A Thesis

Submitted in partial fulfillment of the requirements
for the degree of

Doctor of Philosophy
by
Sheena Leeza Verghese
(023459)

Title:
High Dimensional Low Sample Size Data Analysis: A Probabilistic Graphical Model based Approach for Feature Extraction and Classification

Supervisors:
Dr Iman Yi Liao
Dr Tomas Maul
Dr Chong Siang Yew

School of Computer Science
The University of Nottingham Malaysia Campus
Semenyih (Malaysia)
8 June 2020

Abstract

High dimensional small sample sized (HDLSS) datasets are datasets which contain many features but a limited number of samples. High dimensional low sample size datasets are commonly found in microarray data and medical imaging (Hall *et al.*, 2005). Most algorithms were not created with high dimensional low sample size data in mind. Due to this, predictions made with these datasets do not perform well. Classification algorithms select and discriminate against features which give good prediction. Most classification algorithms assume the Euclidean space. In the Euclidean space, all the data points form an equidistant simplex under the HDLSS asymptotic if the data points are drawn independently from a distribution. Hence, using Euclidean based distance metrics as commonly done by metric learning and K-Nearest Neighbour becomes a hard task. Classifiers such as Support Vector Machines (SVM) have issues with data piling. Support vectors tend to pile on one another causing difficulty in finding the optimal hyperplane. Algorithms such as neural networks and manifold learning require reasonably large amounts of samples to accurately model the data. Further, neural networks, though they can capture the intrinsic dimensionality of the data, they have a black box problem which reduces interpretability of the model. In medical and biological fields, it is desirable to be able to map backwards to the input features that contribute to the prediction. Classifiers also have a tendency to perform poorly due to the misrepresentation of the structure of the data when the density estimation degenerates. The degeneration of the estimation occurs due to the lack of samples. Therefore, it can be seen that that the challenges that are faced by HDLSS data for classification are two fold: i) high dimensionality of the data ii)low sample size of the data.

In our research, we only deal with the high dimensionality problem of HDLSS data. Generating more samples which is usually done to increase the samples size is not in the scope of this research. In order to deal with high dimensionality for classification, common techniques in the literature include i) preprocessing the feature space through the use of feature extraction or selection prior to classification ii) using models with latent structures to capture the intrinsic dimensionality of the data. Feature

selection and extraction methods that have been previously explored deal with high dimensional asymptotics. This means that although the dimensionality is high, the number of samples is larger than the dimensionality size. In the Euclidean space, unlike points in HDLSS data which become equidistant, the data points lie on a narrow strip on the Euclidean sphere. Classical feature extraction methods such as Principle Component Analysis (PCA) which works well under high dimensional asymptotic becomes unstable after a certain point as the eigenvalues can become zero in the HDLSS case (Bishop, 2006). Due to the nature of the data which tends to have correlated features, finding higher order interactions may be potent. Brown *et al.* (2012) studied the application of information theory based feature selection methods and found that only some methods perform well in HDLSS circumstances. In light of this, we use Correlation Explanation (CorEx) (Ver Steeg and Galstyan, 2014a) which is reported to be able to handle high dimensional low sample size data through the use of marginals rather than the full probability distribution of the data to estimate the probability density function.

Correlation Explanation is an unsupervised probabilistic graphical model based on information theory. Unlike Neural Networks, it is also easily interpretable. Using CorEx, we deal with the high dimensionality challenge in HDLSS data for classification by i) proposing feature selection and extraction under the discriminant classification model ii) proposing an inference algorithm under the generative classification model. In order to fulfil the two proposed methods, we needed to model the CorEx algorithm such that class labels can be incorporated. We call this model CorEx-C. The CorEx-C model is able to accurately determine features that are related to the class label with HDLSS data provided its assumptions are not violated. Using the CorEx-C model, the feature extraction and selection methods that were proposed is competitive with the feature selection methods that work with HDLSS data. In the literature,the current feature selection and extraction methods are catered mainly for high dimensional data. We show through the comparison between some existing methods that not all methods can be directly extended to HDLSS settings. This proposed feature selection and extraction method extends the research to aim for HDLSS settings. It also brings into view that more research is required to circumvent the curse of dimensionality in the HDLSS setting. Finally, our proposed inference algorithm (CorEx-Ci) was constructed using mean field approximation. Current classification techniques generally require preprocessing the data with dimensionality reduction when the dimensions are very high. They are also known to have many drawbacks in the HDLSS setting as they were not built with HDLSS data in mind. The exception to this would be Distance

Weighted Discriminant and Maximal Data Piling, which were created for the HDLSS domain.These methods were proposed in literature as an extension to the SVM algorithm for HDLSS data. Our proposed inference algorithm aims to tackle inferencing for HDLSS data from the vantage point of probabilistic graphical model under the generative classification model umbrella. It performs comparatively with other classification techniques. We ran controlled experiments using simulated data to test the capability of a few different classification models and CorEx-Ci when sample size varies and the number of features varies. The increased sample space and feature space only causes slight changes to the performance of CorEx-Ci. However, CorEx-Ci has a slightly higher asymptotic error compared to other classification algorithms possibly due to the violation of the data generation assumption. A future work would be expand this assumption and add constraints to the training phase to accommodate it.

Acknowledgements

I am grateful to many who have helped me and supported me throughout my PhD journey.

First and foremost, I am indebted to God Almighty for His abounding grace and mercy upon me, without which, none of these would be possible.

I am very thankful to my primary supervisor Dr. Iman Yi Liao for her guidance and patience throughout this whole process. I am grateful for her teaching me the many skills required to complete this project. I am also grateful to my secondary supervisors Dr. Tomas Maul and Dr. Chong Siang Yew whose inputs and insights are invaluable.

I would like to thank the Ministry of Education of Malaysia who through the Fundamental Research Grant Scheme (FRGS/1/2014/ICT07/UNIM/02/1) provided the necessary funding for this project.

I would like to thank my parents. My father, Verghese Thirumala, and my mother, Susan Varughese, for inspiring me through their support, determination, and Godliness. I would also like appreciate my brother, Shane, who was a source of relief during the stressful times, and for putting up with my idiosyncrasies throughout this journey.

Lastly, I am deeply appreciative of my colleagues and friends for their camaraderie and encouragement through this process. Thank you.

Sheena Leeza Verghese
University of Nottingham, Malaysia Campus
8 June 2020

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

A high dimensional low sample size (HDLSS) dataset is a dataset where the dimensionality increases while the number of samples available is fixed or increases but at a much slower pace than the dimensionality. High dimensional low sample size can be found with medical images, microarray datasets and some of the healthcare datasets. Advances in modern technology allows images to have higher resolution, hence increasing the dimensionality of the dataset. However, in cases such as medical imaging and gene expression, the number of samples obtained is low as the cost of obtaining samples can be high. Hall *et al.* (2005) claim that in the fields of genomics, chemometrics and functional analysis, having HDLSS datasets is becoming common.

Data collection sometimes known as data harvesting is done readily by different organisations. With improved communication and sharing of information, experts from different fields can create a portfolio containing events that might be deemed useful. However, data harvesting done through the web could threaten the privacy of many individuals. Either way data is being amassed readily causing many of the information to be redundant or unimportant to the task at hand. As such the role of data mining is to discern the different patterns that exist within the collected data to identify important patterns.

Classification algorithms are a part of the supervised learning framework. The goal of classification is to assign a class, c to an input vector, $\vec{X}$. The input space is divided by using decision boundaries in order to assign a class label. Most classifiers are being built to deal with high dimensional data. In general, most algorithms assume the Euclidean space. High dimensional data tend to lie on a narrow strip within the Euclidean sphere. However, the setting of a high dimensional low sample size data is different. The data points drawn independently from a distribution becomes equidistant in the HDLSS case (Hall *et al.*, 2005). The methods such as K-Nearest Neighbour and

Support Vector Machines (SVM) which are able deal with high dimensional data well do not work as well in the high dimensional low sample size data setting. K-Nearest Neighbour suffers due to points being equidistant while SVM suffers from data piling near the hyperplane. State-of-the-art classifiers such as deep neural networks require large quantities of samples to train the model. Some studies on using small samples have been conducted with deep neural networks in (Feng *et al.*, 2019). However, the features were reduced in dimension using PCA. The effectiveness of PCA reduces as the sample size decreases and the dimensions increases (Bishop, 2006). Neural networks especially deep neural networks also have a high variance problem when the sample size is small (Liu *et al.*, 2017a). Extracting the underlying structure of a dataset can enable better understanding of the dataset. It is also able to better discriminate among classes. Information theoretic methods aim at learning the intrinsic patterns which exist within the data and capturing it to be used for classification purposes.

Unlike neural network models which require a function approximator to estimate the probability distribution, directed probabilistic graphical models can directly estimate the probability distribution. Directed graphical models also allow one-to-one mapping between input variables and latent variables. This gives flexibility to interpret the data. Density estimation methods used in general require a reasonable sample size (Silverman, 1986). The lack of samples can cause the degeneration of the density estimation method. This in turn reduces the accuracy of representation of data learnt. Hence, there is a need to improve and explore the current capability of machine learning techniques to accommodate HDLSS datasets. The main focus of this research is to treat the class labels as one of the input features, how probabilistic graphical models can be used to select and extract features, and directly perform inferencing. Information theory allows the modelling of higher order interactions between the input features. This is useful especially in high dimensional situations when there may be high levels of redundancy. Probabilistic graphical models based on information theory deals with non-linear relationships within data. It also has the advantage of better interpretation of the data which is desirable in the medical field. In light of this, Correlation Explanation (CorEx) which was formulated by Ver Steeg and Galstyan (2014a) is used as a case study in this research. Correlation Explanation is an unsupervised graphical model based on total correlation. Total correlation is a generalization of mutual information to accommodate multivariate data. CorEx is reported to learn the intrinsic structure of a data even when the sample size is low. This is due to it requiring less data to estimate the density through the use of marginals rather than the full probability.

Classification algorithms can be broken into two broad categories, i.e. discriminant classifiers and generative classifiers. A discriminant classifier learns a direct mapping of the posterior probability, $p(c|\vec{x})$ from the input data. Meanwhile, a generative classifier learns to model the joint probability before computing the posterior probability. Generative models have high computational complexity as it tries to learn the joint probability of the data. The lower sample size would mean that the joint probability may be far from the population distribution. Assumptions are made in order to make the algorithms tractable. A problem that arises from learning the posterior probability for HDLSS data through the use of covariance matrix is that the covariance matrix generated may not be accurate (Ahn *et al.*, 2007). This is due to the lack of a reasonable sample size. Further, data points have a tendency of piling (Marron *et al.*, 2007) or being equidistant (Beyer *et al.*, 1999) from each other.

Hence, it can be seen that there are two broad classes through which challenges due to HDLSS can be dealt with. The first challenge of HDLSS is its high dimensionality. Existing methods that deal with the challenge include feature selection (Brown *et al.*, 2012), manifold learning (Li *et al.*, 2019), metric learning (Yang and Jin, 2006), and the state-of-the-art representation learning (Mahmud *et al.*, 2018). Principle Component Analysis (PCA) is a method commonly used to reduce the dimensionality of the data. In literature (Jung *et al.*, 2009), PCA is studied to determine under which circumstances the estimation of its statistical distribution can be used for HDLSS data such that the variance is low. It does not look into the intrinsic dimensionality of the data. Metric learning, manifold learning, and state-of-the-art representation such as neural networks learn intrinsic dimensionality of the data. Mahmud *et al.* (2018) showed that using variational autoencoder (VAE) as a preprocessing step to reduce the input dimension works better than some of the other feature extraction methods such as PCA and fast Independent Component Analysis (fastICA). The drawback with VAE is that it is slightly difficult to map the interactions between features that may be related to the performance of the classifier. Metric learning and manifold learning methods on the other hand require a reasonable sample size in order to learn an accurate representation of data. Collecting more data to do so can be very expensive especially in medical fields.

The second challenge is the low sample size which results in less accurate sample distribution and related statistics having large variances. Existing methods to deal with the problem include upsampling (Maldonado *et al.*, 2019) and generating artificial samples (Yang *et al.*, 2011). Random upsampling using algorithms such as SMOTE can lead to overfitting of any classification model when the number of samples is ex-

tremely small. This is due to very small variability between samples in the training set when the algorithms are used. Hence, the second challenge is not directly addressed in this research as some preliminary results with variational autoencoders have shown that it is difficult to assess the quality of the random samples generated as to whether they would reflect the true data distribution when the real data sample size is very small. Thus the second challenge is out of scope. The research focus of this thesis is the first challenge.

1.0.1 Research Statement

This research tries to classify HDLSS data through learning the intrinsic structure of the data that is related to the class labels.

High dimensional low sample size data are becoming more common as data is being collected. Examples of data with HDLSS asymptotics include medical images such as MRI and CT scans and microarray gene datasets. Collecting sufficient samples are expensive. Many common classification algorithms are not built with HDLSS data in mind. They suffer problems such as overfitting, data piling, and data points being equidistant from each other. HDLSS data has two main challenges, namely its high dimensionality and its low sample size. We will be focusing on the challenge of the high dimensionality of the HDLSS data. Moreover, high dimensional low sample size data usually has a lower intrinsic structure.

In the literature, two common methods are enforced to deal with high dimensionality, i.e. feature selection and extraction as a preprocessing step to classification or using latent variable models to capture intrinsic dimensionality of the data. Classical feature extraction methods such as PCA are unstable as the principle components are only defined until the number of features (Bishop, 2006). State-of-the-art latent models such as Neural Networks require large amounts of data to accurately model the data. Hence there is a need to study and understand classification for HDLSS data.

In order to deal with the high dimensionality of HDLSS data, we use Correlation Explanation (CorEx) as our case study. We extend the CorEx model to include the class label which we call as CorEx-C. We conduct feature extraction and selection under the discriminative classification framework. We also use CorEx-C to perform inferencing through the use of mean field approximation. Mean field approximation is a variational inference method. It makes the assumption that the latent features are independent of each other.

1.0.2 Aims & Objectives

The overall aim of this research is to classify HDLSS data through learning an intrinsic structure or representation of the data that is related to class labels. The specific objectives include:-

1. To design a graphical model that is able to represent and capture the relationship of observed data (features & labels) and its intrinsic hidden structure.

2. To propose a feature extraction/selection method based on the proposed graphical model for HDLSS data classification under the discriminative classification framework.

3. To propose an inference algorithm based on the proposed graphical model for HDLSS data classification under the generative classification framework.

4. To evaluate the proposed graphical model and classification algorithms with both simulated data as well as benchmark datasets.

1.0.3 Contributions

The contributions of this thesis based on the aims and objectives are as follows:-

1. A graphical model based on CorEx but with class label as an additional feature, which we name as CorEx-C model is proposed. We exploited the modularity of CorEx between features and latent nodes to model a mixture of distributions. We show that CorEx-C is able to learn features which are strongly connected to the class label, if the data generation assumptions made holds.

2. A feature extraction method and a feature selection method based on the connection between class label and hidden nodes and features through hidden nodes respectively is proposed. The methods were run using the CorEx-C model. Experimental results have shown that the two methods are comparable to the state-of-the-art information theoretic based feature extraction and selection methods such as Maximum Relevance Minimum Redundancy, Infomax Independent Component Analysis, and Kernel Entropy Component Analysis.

3. A Bayesian inference method based on the Mean Field inference concept for a trained CorEx-C model is proposed. Experimental results on both simulated data and benchmark datasets have shown that the proposed inference method is

comparable to discriminative methods. Control experiments were done to evaluate the trend shown when the dimensionality of the original feature space increases. It was seen that redundancy is bad only if the redundant features are not relevant to the class labels. Through the simulated data, we show that adding relevant features as redundancy causes a bias which could help improve classification performance. Experiments were also run to understand the ability of the classification methods when sample size increases.

1.0.4 Scope & Limitation

The scope and limitations of this research includes the following:-

1. The current research primarily concerns binary classification problems, but the proposed methods can be easily extended to multi-class classification problems.

2. The current research does not consider generating artificial samples to upscale the sample size. There have been some preliminary experiments on employing variational autoencoder on very small sample size data ($n \leq 100$) but the generated results were not representative of the sample distribution.

3. The current research has instantiated the idea based on CorEx as the case study method. CorEx was selected as it is able to handle high dimensional data as well as data with small sample sizes. However, it is possible to explore and implement the concept on other unsupervised graphical models. An unsupervised method can be directly converted to a supervised method if the class labels are available.

1.0.5 Organisation of the Thesis

Chapter 2 of the thesis consists of the literature review. Some analysis of the geometry of HDLSS data is reviewed along with common machine learning methods. The basics of probability graphical model is also introduced in this chapter. Chapter 3 discusses the methodologies used to propose CorEx-C, using CorEx-C for feature extraction and selection under the discriminant classification umbrella, and the formulation of CorEx-Ci under the generative classification umbrella. The results are analysed in Chapter 4. Chapter 5 gives the conclusion and lists out some future work.

Chapter 2

Literature Review

2.1 Analysis of High Dimensional Low Sample Size Dataset

High dimensional low sample size datasets occur due to the vast harvesting of data that exists. The data harvested are often times irrelevant or redundant to the problem at hand. As such it is said to be hit by the curse of dimensionality as the data is extremely high dimensional while the training of the data is restricted due to low sample size. The classical asymptotic analysis deals with an increasing number of samples, n with the dimensionality, d being fixed (Ahn, 2006). However, there exist other forms of asymptotics. They are: -

1. $n \to \infty, d \to \infty$ and $d/n \to \gamma \in (0, \infty)$. As the number of samples, n increase so does the number of dimensions, d. It is often also assumed that the ratio of d/n goes to some constant (Ahn, 2006). This case of asymptotics are the ones that are related to high dimensional datasets.

2. n is fixed, $d \to \infty$: The number of samples remains constant even as the number of dimensions increase. Such a case is usually termed as high dimensional low sample size dataset (Hall *et al.*, 2005).

It is the latter case that will be studied in this thesis.

The understanding of the structure of HDLSS data will form the basis of our critic throughout the thesis.

2.1.1 Geometrical Structure of High Dimensional Low Sample Size Dataset

According to Hall *et al.* (2005), the geometrical structure of random samples from a Gaussian distribution $\mathcal{N}(\mu,\sigma^2)$ for HDLSS datasets does not lie near the population mean. Normally distributed Gaussian states that most samples should concentrate around the centre. That is to say that approximately 68% of the data lie between $\mu-\sigma$ and $\mu+\sigma$ where μ is the mean and σ is the standard deviation. Therefore, for a 10-dimensional dataset it is approximated that only 0.02% of the data will lie within the hypersphere graph for high dimensional low sample size dataset. This points towards the fact that as the dimensionality increases, the samples become farther away from each other, increasing the sparsity of the dataset.

Geometric Analysis of High Dimensional Low Sample Size Data

The increase in dimensionality, d, when the number of samples, n, is fixed causes the pairwise distance between data vectors to become constant (Hall *et al.*, 2005). This would mean that the data in HDLSS setting lies only in random rotations of a regular simplex in $\mathbb{R}^d$. Due to the fact that it is lying at the vertices of a regular simplex, the distance between the data points will be almost equidistant from each other. In Euclidean space, the geometric distance between any two points is the same as between that point and other points (Hall *et al.*, 2005). As such, many of the machine learning algorithms that use Euclidean distances to classify or cluster HDLSS data set will not perform as well. The pairwise distances between each point form an n-simplex (National and Recherche, 1986). In a standard Gaussian representation- let $Z(d) = \{Z^{(1)},...,Z^{(d)}\}$ be a d-dimensional random vector where $N(0,I)$, where I is the identity covariance matrix - the Euclidean distance using the simple delta method calculation is given by (Hall *et al.*, 2005),

$$||Z|| = (\sum_{k=1}^{d} Z^{(k)^2})^{\frac{1}{2}} = d^{\frac{1}{2}} + O_p(1) \tag{2.1}$$

Eq. (2.1) shows that the data tends to lie near the expanding sphere as $d \to \infty$ (Hall *et al.*, 2005). By expanding Eq. (2.1) to two independent vectors, $Z_1(d)$ and $Z_2(d)$ from the standard normal,the following equation is produced by Hall *et al.* (2005) using the delta method,

$$||Z_1 - Z_2|| = (2d)^{\frac{1}{2}} + O_p(1) \tag{2.2}$$

as $d \to \infty$. This shows that the data points tend to be deterministically apart. By looking at the angles between the data point, the pairwise angles are almost equal (Hall *et al.*, 2005). The angles are given by Hall *et al.* (2005) using the delta method,

$$ang(Z_1, Z_2) = \frac{1}{2}\pi + O_p(d^{-\frac{1}{2}}) \tag{2.3}$$

where $\mathcal{O}_p$ (Big-O) is the stochastic boundedness. The pairwise angles point to the fact that they are approximately perpendicular to each other (Hall *et al.*, 2005).

Both Eq.(2.2) and Fig.(2.3) hold for pairwise points, hence showing that pairwise distances are almost the same. Since the pairwise distances are almost the same, it can then be said that the points lie at the vertices of an equilateral triangle for the case where the intrinsic dimensionality is 3.

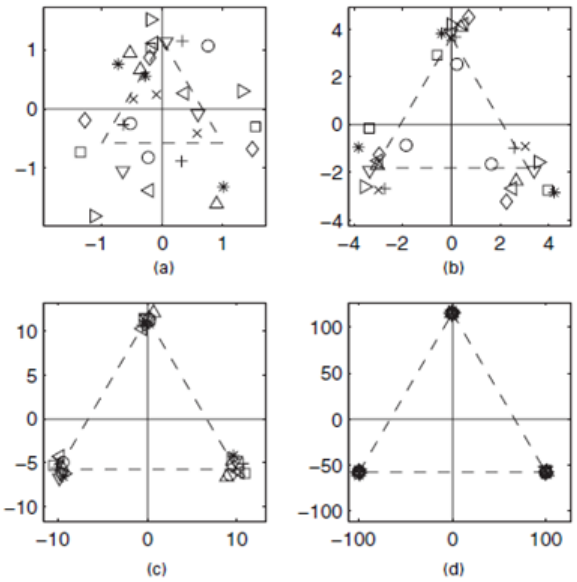


Figure 2.1: Geometric representation and convergence of 3-simplex when d increases by a)2 b)20 c) 200 d) 20,000. This illustration is taken from Hall *et al.* (2005).

2.1.2 Analysis of Covariance Matrix in High Dimensional Low Sample Size Settings

The HDLSS dataset does not have enough data to correctly and accurately estimate the covariance matrix. The covariance matrix is used to detect the underlying distribution of the data. Due to this, efforts to produce meaningful estimates or predictions becomes a difficult task (Ahn *et al.*, 2007). The matrices of HDLSS data sets have been found to be singular and the covariance matrix is not full rank (Hall *et al.*, 2005). A matrix that is singular is not invertible, meaning that the determinant of the matrix will be 0 and the eigenvalues may also be 0 (Ahn, 2006). This could mean that its columns

may be a combination of other columns in the dataset. Hence, the columns may not be independent of each other and are possibly correlated to each other. An implication of this may be that unlike most other types of datasets, the HDLSS datasets cannot 'sphere the data' (Hall *et al.*, 2005). 'Sphering the data' would mean to multiply the data vector by the square root inverse of the covariance matrix (Hall *et al.*, 2005). Therefore, statistical methods that are used for analysis which are based upon covariance data may not work very well for datasets that are high dimensional and low sample size.

Ahn *et al.* (2007) states that the lack of data samples in the dataset can lead to a wrongly perceived geometry of the underlying covariance structure. The learning of an incorrectly generated covariance structure can lead to poor performance of a dataset. Bentler (1996) states that sample size is critical as most statistics is based on the assumption that the number of samples available will become arbitrarily large (Bentler, 1996). This is clearly not the case with HDLSS datasets where the number of samples remain fixed. Hence, the traditional methods of statistical analysis of datasets require a reasonably large number of data samples which is not available with HDLSS datasets.

2.1.3 Analysis of Statistical Tests for Sphericity

Sphericity is a condition where the variances of the differences between all possible parts of a group is equal. It is an important assumption made by statistical tests such as ANOVA. Violation of the sphericity assumption by HDLSS data would mean that statistical tests cannot be reliably used. Mauchly's test of sphericity is the formal method to test the assumption of sphericity. Mauchly's test (Mauchly, 1940) for sphericity was used in Ahn *et al.* (2012) to test the departure from it. The hypothesis used is given as follows,

$$H_0 : \sum = \sigma^2 I \tag{2.4}$$

vs

$$H_1 : \sum \neq \sigma^2 I \tag{2.5}$$

where σ is the standard deviation, is not specified and $\sum$ is the covariance matrix in a multivariate Gaussian distribution with mean, $\mu = 0$. H_0 is the null hypothesis, the test for sphericity while H_1 is the test for non-sphericity (Ahn *et al.*, 2012). These hypothesis formed is an ANOVA method, which is validation through a repeated measure of analysis of variance. If the likelihood of the first test is more than a certain value (a constant is chosen accordingly by the researcher who tests the hypothesis), then the sphericity holds as the as the null hypothesis is not rejected (Ahn *et al.*, 2012).

The maximum likelihood formula used to test this is given by,

$$R_1 = \frac{|A|^{\frac{1}{2}n}}{(tr(A)/d)^{\frac{1}{2}dn}} \tag{2.6}$$

where $A = nS$, $S = \frac{1}{n}XX^T$, X is a $d \times n$ matrix with d, dimensions, tr is the trace of the matrix and n is the sample size. The likelihood criterion proposed by Maulchy, however, if S is singular, this test cannot be used as it is degenerate. As such, a different criterion was used to test the sphericity condition. It is given as follows (John, 1971),

$$U = \frac{1}{d}tr[(S-I)^2] \tag{2.7}$$

where I is the identity matrix. This method holds when $d >> n$.

The consequence of violating the sphericity is an increase in the number of Type I errors, i.e., false positive errors. Hence, the conventional multivariate analysis does not work for HDLSS because it was developed under the $n \to \infty$ with d fixed framework. Under the normal data asymptotics, when the test holds, it is easy detect the deviation from sphericity. However, when even the test degenerates, the authors believe new algorithms should be explored and formulated to handle it.

The other hypothesis testing problem is to test whether the underlying covariance matrix is equal to the given positive definite matrix, $\sum_0$. This condition can be tested using ANOVA on the following hypothesis:

$$H_0 : \sum = I \tag{2.8}$$

vs

$$H_1 : \sum \neq I \tag{2.9}$$

by multiplying the data by $\sum_0^{-\frac{1}{2}}$. If the null hypothesis H_0 holds, it means that the underlying covariance is equal to the given positive definite matrix. This would imply that the eigenvalues are all more than zero, which is an important trait for Principle Component Analysis (PCA) as it uses the first few eigenvectors to characterise the dataset. On the other hand, if H_1 holds, the underlying covariance is not equal to the positive definite matrix. This means that the eigenvalues could be zero. This could also mean that a variable could be linearly dependant on other variables, leading to a possible correlation among the data. Nagao (1973) derived V as the equivalent of U to test the second hypothesis. V is given by:

$$V = \frac{1}{d}tr[(S-I)^2] \tag{2.10}$$

While U holds for when $d >> n$ in the test for sphericity, V does not hold under the $d >> n$ conditions to test the second hypothesis. V tends to degenerate when d

exceeds the number of samples in the dataset. Ledoit and Wolf (2002) introduced a modified version of V which holds even when $d >> n$. This is given by,

$$W = \frac{1}{d} tr[(S-I)^2] - \frac{d}{n}[\frac{1}{d} tr(S)]^2 + \frac{d}{n} \tag{2.11}$$

As such, it can be seen that new techniques of analysis have to be produced in order to test the capabilities of HDLSS datasets to the fullest. Statistical methods do not always extend towards the HDLSS asymptotics. Most of the tests and analysis done were mainly under the fixed d and increasing n framework where $n >> d$. New analytical methods have to be created for the HDLSS framework as done by Ledoit and Wolf (2002). It should also be noted that any method that requires the use of the inverse of the covariance matrix will not work due to the singularity of the covariance matrix. This would mean that the Moore-Penrose inverse would have to be applied.

2.2 Investigation of Classification Methods in High Dimensional Low Sample Size Conditions

Classification tasks can be modeled using either a discriminative classifier or a generative classifier. Discriminative classifiers learn the posterior, $p(c|X)$ directly from mapping the input X to the class labels c. Meanwhile, generative classifiers learn a model of the joint probability, $p(X,c)$, where $c \in C$, $X \in (X_1, X_2, \ldots, X_d)$, and $d \in D$ is the number of dimensions. Predictions are then made using Bayes' rule to pick the most likely class label. Discriminant models focus on separating the decision boundary between classes while generative models focus on capturing the underlying data generation process (Tu, 2007). According to Ng and Jordan (2002), discriminative models achieve a lower asymptotic error rate compared to generative models as given by the following proposition:

$$\epsilon(h_{\text{Dis,inf}}) \leq \epsilon(h_{\text{Gen,inf}}) \tag{2.12}$$

where $\epsilon(h_{\text{Dis,inf}})$ is the error of any population version of any discriminative classifier. $\epsilon(h_{\text{Gen,inf}})$ refers to the error of any population version of the generative classifier. This proposition forms the basis of the belief that discriminant classifiers are better than generative classifiers. An assumption made here is that the classifiers have finite Vapnik–Chervonenkis (VC) dimensions. The sample complexity for discriminant classifiers are said to be lower bound by $\omega(n)$, where n is the sample size. Generative models requires lower amounts of training samples ($\mathcal{O}(\log n)$) to approach $h_{\text{Gen,inf}}$. This

implies that with lower amounts of training samples ($\mathcal{O}(\log n)$), a generative model should be able to reach its asymptotic error at a faster rate than discriminative models. This indicates that as the sample size increases, the discriminative model should perform better than the generative model (due to the fact that it has lower asymptotic error). Asymptotic error is the standard error when the number of samples approach infinity, $n \to \infty$. It has to be noted that Ng and Jordan (2002) made the propositions for the generative models by considering Naive Bayes which decouples into independent closed form optimization problem. This is not necessarily the case for all generative models (Liang and Jordan, 2008). Generative models generally make simplified assumptions in order to learn the data generation process, e.g. conditional independence assumption of Naive Bayes. However, generative models try learn a richer structure of the classes compared to discriminative models.

Examples of generative models include Naive Bayes and graphical models. Support Vector Machines (SVM), K-Nearest neighbour, and traditional neural networks are discriminative models.

2.2.1 Support Vector Machines (SVM)

SVMs work as a binary classifier. Multi-class classification can be done with SVM by combining multiple binary classifiers (Bishop, 2006). SVM uses the concept of maximal margin to classify the data. A maximal margin is the sum of the distances from the separating hyperplane from the closest example of class a to the closest example in class b (Bishop, 2006). The larger the margin between these classes the better the generalization error of the classifier, i.e. the generalization error tends to become lower. This is because when the margin is bigger the ability to correctly predict the class a point belongs to increases (Bishop, 2006).

Support Vectors are points that lie on the closest to the hyperplane (Bishop, 2006). The geometric locality of the hyperplane does not change even with the change in data samples, provided that the support vectors do not change (Klement *et al.*, 2008). This would mean that once the support vectors have been identified, the remaining data points can be discarded as new data points are compared to the support vectors. This allows for saving where the usage of memory is concerned (Zhang and Lin, 2011). It is the distance between support vectors that allow a decision about the classification of a new point, x can be made.

SVM mainly works with linearly separable data (Bishop, 2006). However, with the use of kernels, SVM can be made to work with non-linear data(Bishop, 2006). SVM transforms the data into high dimensional feature space in which classes become lin-

early separable (i.e. Projection of data into higher dimensional spaces). This can cause problems if the original data is itself a high dimensional data with low sample size, as the projection of this data into a higher dimensional space can lead to increased sparsity of data in the higher dimensional space.

In a high dimensional space, SVMs should perform well. This can be seen by looking at the generalization error bounds that was introduced by Boser *et al.* (1992) for Large Margin Classifiers. The error bound ϵ is given by :

$$\epsilon = \tilde{O}(\frac{1}{n}(\frac{R^2}{\gamma^2} + log\frac{1}{\delta})) \tag{2.13}$$

where n is the number of training samples, γ is the margin between parallel planes, $(R,\delta) \in \mathbb{R}^+$ with $0 < \delta \leq 1$ and $\tilde{O}$ which is the variant of the big-O function which ignores the logarithmic factors $\exists k : f(x) \in O(g(x)log^k g(x)$. The generalization error is only based upon the sample size and not the dimensionality of the data. It is inversely dependent on the sample size, n and the inverse of the square of the margin γ.

SVMs generally uses Fisher's consistency, meaning which when the number of samples approaches infinity, $n \to \infty$, its decision rule converges to the Bayes discrimination rule (Qiao, 2013). However, in the HDLSS setting, the angle between the direction of the SVM and the angle between the direction of the Bayes' rule is generally large (Qiao, 2013). This would mean that the accuracy of classification of SVMs in HDLSS settings does not perform as well as it should. The variability of the sampling distances of the SVM direction becomes large implying that a consistent generalizability of a HDLSS dataset using SVMs becomes poor.

It was found by Marron *et al.* (2007) that SVMs are prone to overfitting in the HDLSS setting. A condition known as data piling occurs due to the overfitting of the SVM (Marron *et al.*, 2007). The clumping of data samples in a particular point when using support vector machines leads to a phenomenon known as data-piling (Qiao, 2013). In a support vector machine, a large number of support vectors have been found to be projected onto a single point (Ahn *et al.*, 2012). This factor further pushed the SVM away from making accurate classification for a given new data, X^*.

Two methods that address the data piling phenomenon are Distance Weighted Discriminant (DWD) and Maximal Data Piling (MDP). Distance Weighted Discriminant is a method which reduces data piling by replacing the margin based criterion of SVMs by using a different function of distances such as by optimizing the sum of the inverse distances (Marron *et al.*, 2007). DWD is sensitive to imbalanced classes in a dataset unlike SVMs. Maximal Data Piling on the other hand, maximises the piling of data by finding the support vectors that form the largest distance between the piles

among the classes (Ahn *et al.*, 2012). It finds the maximum direction vector between class while searching for piles that would minimize the scatter within class. In other words, MDP often acts as Fisher's Linear Discriminant Analysis would in non-HDLSS settings (Ahn *et al.*, 2012).

As such it can be seen that the classical SVMs do not work as well as they should, due to data piling in the case of HDLSS data. This could mean that the distance between the samples or support vectors becomes indistinguishable.

2.2.2 K-Nearest Neighbour (KNN)

K-Nearest Neighbour is a classification algorithm whereby a new point x which is yet to receive a class label will query the closest point y to assign the label of y to x (Thirumuruganathan, 2010). Distance metrics are used to query for the closest point(s). Due to the geometric structure of HDLSS data as discussed in section 2.1.1, querying for a point closest to the data becomes a hard task in the Euclidean space. The distance between the nearest points converges to the distance between the farthest points (Beyer *et al.*, 1999). This fact will be further aggravated in a scenario whereby the points in a dataset lie in a circle making the determination of the nearest neighbour with a high-level confidence a very difficult task (Beyer *et al.*, 1999).

Metric Learning

Metric Learning focuses on automating the process of selecting and learning distance functions (Kulis *et al.*, 2013). It is often used in conjunction with the KNN algorithm. In the metric learning community, the application of Mahalanobis distance refers to whitening of the data and then applying Euclidean distance (Kulis *et al.*, 2013).

Given $X = [\vec{x}_1, \cdots, \vec{x}_n]$ is a matrix of all data points, the Euclidean distance is given as $||\vec{x}_i - \vec{x}_j||_2 = \sqrt{(\vec{x}_i - \vec{x}_j)^T(\vec{x}_i - \vec{x}_j)}$ where ${\vec{x}_i}^T \vec{x}_j$ is the canonical inner product. Whitening a random vector $\vec{w}$ with mean μ and covariance σ produces $\vec{w^*} = \sigma^{\frac{-1}{2}}(\vec{w} - \mu)$. The Euclidean distance with the whitened variables is the Mahalanobis distance. The Mahalanobis distance has the form:

$$d_A(\vec{x}_i, \vec{x}_j) = (\vec{x}_i - \vec{x}_j)^T A(\vec{x}_i - \vec{x}_j) \tag{2.14}$$

where A is a positive semi-definite matrix.

Some examples of metric learning algorithms are Adaptive Kernel Metric Nearest Neighbour Classification, Information Theoretic Metric Learning (ITML), Large Margin Distance Metric Learning (LMDML), and Sparse Distance Metric Learning.

1. Adaptive Kernel Metric Nearest Neighbour (Jing Peng *et al.*, 2002) classification uses a quasiconformal kernel for nearest neighbour. It basically adjusts the RBF Kernel by introducing weights based on local consistency of class labels and label uncertainty. The quasiconformal mapping is defined as: $\hat{K}(\vec{x},\vec{x}_0) = c(\vec{x})c(\vec{x}_0)K(\vec{x},\vec{x}_0)$, where $\vec{x}$ is a sample and $\vec{x}_0$ is a testing sample, and $c(\vec{x})$ is a positive function of $\vec{x}$. The distance metric is then defined as:

$$D(\vec{x},\vec{x}_0) = c(\vec{x})^2 - 2c(\vec{x})c(\vec{x}_0)(1-K(\vec{x},\vec{x}_0)) + c(\vec{x}_0)^2 \tag{2.15}$$

assuming $K(\vec{x},\vec{x}) = 1, \forall \vec{x}$. The weight function $c(x)$ is defined as $c(\vec{x}) = \frac{P(j_{\hat{m}}|\vec{x})}{P(j_m|x_0)}$, where $j_m = \arg\max_j P(j|\vec{x}_0)$, j is the class label at point $\vec{x}$ and $j_{\hat{m}} = 1 - j_m$ by Domeniconi and Gunopulos (2002). Through Taylor Expansion, it is simplified to a weighted combination of Chi-Squared distance and Mahalanobis distance. Chi-squared metric requires a sufficiently large sample size to produce accurate inference (McHugh, 2013) which is not the case for HDLSS data.

2. Neighbourhood Component Analysis (Goldberger *et al.*, 2005) learns a Mahalanobis distance for KNN classifier by maximising the performance of leave-one-out cross validation. It assumes distance metric to be in the form of $Q = A^T A$, where A is a matrix. Given a data point $\vec{x}_i$, the soft neighbour is given as the probability that $\vec{x}_j$ is selected as the neighbour and has the same class label as $\vec{x}_i$, p_{ij}.

$$p_{ij} = \frac{\exp(-||A\vec{x}_i - A\vec{x}_j||^2)}{\sum_{k\neq i}\exp(-||A\vec{x}_i - A\vec{x}_k||^2)} \tag{2.16}$$

where the set of points that share the same class with data point $\vec{x}_i$ be denoted by $C_i = \{j|c_i = c_j\}$. The expected number of correctly classified points is given as

$$f(A) = \sum_{i=1}^{n}(\sum_{j\in C_i} p_{ij}) \tag{2.17}$$

The drawback to this formulation is that it cannot scale to large dimensionality because its objective function is differentiated w.r.t. distance matrix. The parameters in matrix A is quadratic to the number of features thereby causing overfitting if the training sample is insufficient. It also makes the computation of matrices with large dimensions intractable.

3. Large Margin Distance Metric Learning (LMDML) employs margin maximization in learning a distance metric in order to improve the K-Nearest Neighbour classifier. The use of semidefinite programming to overcome the positive semidefinite constraint is intractable in large datasets. Hence, (Nguyen *et al.*, 2018), uses stochastic gradient descent to overcome this problem.

The cost function, $e(L)$, of LMDML is given as :

$$e(L) = \sum_{ij} \nu_{ij} ||L(\vec{x}_i - \vec{x}_j)||_2^2 + C \sum_{ijl} \nu_{ij}(1 + y_{il})[1 + ||L(\vec{x}_i - \vec{x}_j)||^2 - ||L(\vec{x}_i - \vec{x}_l)||^2]_+ \tag{2.18}$$

where $[z]_+ = \max(z, 0)$ denotes the standard hinge loss, $\nu_{ij} \in \{0,1\}$ indicates whether $\vec{x}_i$ is a neighbour of $\vec{x}_j$, $y_{ij} \in \{0,1\}$ indicates whether labels $y_i = y_j$, $\vec{x}_l$ is an imposter neighbour, and the constant $C > 0$. It is the second term of the cost function which incorporates the idea for margin (Yang and Jin, 2006). The cost function ensures that the distance between the classes are as large as possible.

4. Information Theoretic Metric Learning (ITML) (Davis *et al.*, 2007) minimizes the LogDet divergence w.r.t. linear constraints. It aims at satisfying the similarity and dissimilarity constraints while staying as close as possible to the Euclidean distance.It is that it is equivalent to minimizing the KL divergence between two multivariate Gaussian distributions parameterized by M and M_0, where M_0 is usually set to identity matrix. The objective function is given as:

$$\begin{aligned} &\min_{M \in S_+^d} D_{ld}(M, M_0) + \gamma \sum_{i,j} \varepsilon_{ij} \\ &\text{s.t. } d_M^2(\vec{x}_i, \vec{x}_j) \le u + \varepsilon_{ij} \qquad \forall (\vec{x}_i, \vec{x}_j) \in S \\ &\qquad d_M^2(\vec{x}_i, \vec{x}_j) \ge v - \varepsilon_{ij} \qquad \forall (\vec{x}_i, \vec{x}_j) \in D \end{aligned} \tag{2.19}$$

where D_{ld} is the LogDet divergence, $u, v \in \mathbb{R}$ are threshold parameters, S is the set of similarity, D is the set of dissimilarity, S_+^d is the cone symmetric positive semi-definite $d \times d$ real valued matrix, ε is the slack variable, and $\gamma \ge 0$ the trade-off parameter. For high dimensional datasets, feature selection was first applied to get a lower mapping before ITML was applied. This two-step process would improve the performance of HDLSS data sets, provided the feature selection method is stable under the HDLSS asymptotic.

5. Sparse Distance Metric Learning (Qi *et al.*, 2009) is a metric algorithm that was specially designed for HDLSS data. In order to avoid overfitting , LogDet and l_1 was used as the regularizers. The objective function is given by

$$\min_M tr(M_0^{-1} M) - \log\det M + \gamma ||M||_{1,\text{off}} + \nu L(S, D) \quad \text{s.t.} M \ge 0 \tag{2.20}$$

where $M \ge 0$ is the positive semi-definite constraint, γ is the balance parameter trading off between sparsity prior and the M_0 prior, and $L(S, D)$ is a loss function

defined on the sets of similarity S and dissimilarity D constraints. ν is a positive balance parameter trading off between the loss function and the regularizer and $||M||_{1,\text{off}}$ refers to applying l_1 on the off-diagonal (Qi *et al.*, 2009). The reason this metric learning algorithm is able to handle HDLSS data is that the l_1 regularizer shrinks the less important features to zero. It essentially is doing some form of feature selection while learning the distance metric.

Metric learning methods work best if the number of dimensionality is less than the sample size. KNN algorithm when used in conjunction with metric learning is still aversely affected. In order to circumvent it, like the sparse distance metric learning algorithm, either some form of constraint has to be added to enable feature selection or explicit preprocessing has to be applied.

2.2.3 Naive Bayes

Naive Bayes is a generative classifier. It is considered naive because of the conditional independence assumption. The conditional independence assumption usually does not hold up against real world datasets. The equation is given by:

$$p(\vec{x}|c) = p(c)\prod_{i=1}^{d} p(x_i|c) \tag{2.21}$$

Sordo and Zeng (2005), compared SVM, Decision Tree, and Naive Bayes classifiers for text classification data with sample size ranging from approximately 150 samples to 8500 samples, and a dimension of size 194. They found that the Naive Bayes classifier performs better than the SVM and Decision Tree when the sample size is less than 340. However, as the sample size increased, SVM and Decision Tree have a higher accuracy. This finding by Sordo and Zeng (2005) complements that of Ng and Jordan (2002). The plots by Sordo and Zeng (2005) show a faster convergence of accuracy for Naive Bayes compared to SVM or Decision Tree. Naive Bayes is less affected by the curse of dimensionality because of the conditional independence assumption. The assumption allows the features to decouple. This would mean that parameter estimates are done in 1 dimensional space.

As seen, classical classifiers were not made with HDLSS data in mind. As such, more research is required to develop algorithms which can hold up under the HDLSS setting.

2.3 Feature Selection and Extraction

Feature selection and extraction methods are used to reduce the dimensionaity of the dataset. Feature extraction and selection methods are commonly categorized based on meta-heuristic strategies. Meta-heuristic strategies refers to the categorization of methods based on filter, wrapper and embedded methods. Guyon and Elisseeff (2003) discussed feature selection methods in terms of meta-heuristics for producing better prediction results and provided a general guideline on feature selection methods. Chandrashekar and Sahin (2014) provided a generic introduction to feature extraction and feature selection based on meta-heuristic methods. Blum and Langley (1997) on the other hand, focused on selecting relevant features as well as relevant samples through the meta-heuristic framework. Miao and Niu (2016) explored feature selection methods for improving performance of clustering algorithms, and compared between supervised and unsupervised feature selection methods. Brown *et al.* (2012) focused on filter based information theoretic feature selection methods and created a unifying framework to compare the methods. Bolón-Canedo *et al.* (2014) compared some feature selection methods on microarray data under the meta-heuristic framework .

Other categorizations of feature extraction and selection methods also exist. Li *et al.* (2017) discussed feature selection methods based on different types of data. Wang *et al.* (2016) provided search based strategies in bio-informatics. Cunningham and Ghahramani (2015) reviewed feature selection and extraction methods that are linear in nature. Ding *et al.* (2012) categorized feature extraction and selection methods based on different sub-fields in machine learning including statistical analysis, information theory, artificial neural networks and optimization based algorithms. Xue *et al.* (2016) reviewed feature selection methods based on evolutionary computing, and outlined its strengths and weaknesses.

While feature extraction and selection is used within the field of computing, it can also be applied to benefit other fields. Hence, awareness needs to be raised in other fields. Khalid *et al.* (2014) surveyed feature extraction and feature selection methods in the context of eye diseases to create awareness of the usage of the methods to ophthalmologists. Saeys *et al.* (2007) on the other hand, reviewed feature selection techniques for sub-fields within bio-informatics.

As with all areas, there are challenges, open problems and future directions to explore with feature extraction and selection methods. Liu *et al.* (2010) discusses some future directions such as developing computational theories to keep pace with the rapid mining of data, the usage of symbolic learning and statistical learning in tandem, and developing efficient approaches to estimate feature relevance and extract

features in HDLSS spaces. Storcheus *et al.* (2015) discusses some challenges such as the trade-off between supervised and unsupervised feature extraction and selection methods, the scalability of current feature extraction and selection algorithms for HDLSS data, and the relationship between convex and non-convex feature extraction methods. Hence, it can be seen that there is a need to study and extend feature selection and extraction methods in the HDLSS domain.

2.3.1 Principle Component Analysis (PCA)

Principle Component Analysis (PCA) is widely used to perform feature selection and reduction. PCA is defined as the orthogonal projection of data onto a lower dimensional linear space (principle subspace) such that the variance of the projected data is maximized(Bishop, 2006). It chooses the features with the largest variance spread. This will allow maximum coverage of the features as well as reduce the number of features that are correlated to each other (Bishop, 2006). The selection criterion of the the principle components can be done using the eigenvalue (Bishop, 2006). The use of the eigenvalue is done such that each component should have at least one variable's worth of variability (i.e. The eigenvalue has to be more than 1).

The eigen-decomposition used in PCA of a given population covariance matrix is given by $\sum_d$ is :

$$\sum_d = U_d \Lambda_d U_d^T \tag{2.22}$$

where Λ_d is the diagonal matrix of eigenvalues, U_d is the orthogonal matrix composed of eigenvectors of the matrix and U_d^T is the transpose of U_d.

The variation between the sample eigenvalue and population eigenvalue should be as minimal as possible to capture a more accurate structure of the data (Jung *et al.*, 2009). This is also known as the consistency of the covariance matrix whereby the direction of the eigenvector of the sample eigenvector is consistent or the same with that of the population eigenvector (Jung *et al.*, 2009). An inconsistent direction between the sample and population eigenvector points to a reduced representation of the dataset structure. In the case of high dimensional low sample sized settings, the eigenvalues of the sample covariance matrix displays tendencies of being the identity matrix (Jung *et al.*, 2009). This implies strong inconsistencies in the directions of principle components among the sample covariance matrix(Jung *et al.*, 2009).

Cross data matrix can give feasible estimates of eigenvalues in the HDLSS setting (Yata and Aoshima, 2010). The cross data matrix is given by:

$$S_{D(1)} = n^{-1} X_1^T X_2 \tag{2.23}$$

where X_1 and X_2 are two $d \times n$ matrix which are i.i.d. $S_{D(1)}$ is the dual matrix of the sample covariance matrix.

Let the sphered $d \times n$ matrix be given as:

$$Z_i = \Lambda^{-\frac{1}{2}} U^T X_i \tag{2.24}$$

where $i = 1,2$ then, the cross data matrix becomes,

$$S_{D(1)} = n^{-1} \sum_{j=1}^{d} \lambda Z_{1j} Z_{2j}^T \tag{2.25}$$

. By applying singular value decomposition on the cross data matrix,

$$S_{D(1)} = \sum_{j=1}^{n} \hat{\lambda}_j \hat{a}_{j(1)} \hat{a}_{j(2)}^T \tag{2.26}$$

where $\hat{\lambda}$ is the singular value of $S_{D(1)}$ and $\hat{a}_{j(1)}$ and $\hat{a}_{j(2)}$ is the unit left or right of the singular vector corresponding to $\hat{\lambda}$.

The steps for estimating the eigenvalues are given by Yata and Aoshima (2010) as follows:

1. Calculating the cross data matrix.

2. Calculate the singular values, $\hat{\lambda}$ of the cross data matrix to determine the estimate of the eigenvalues.

Using cross data matrix method gives some improvements to the estimation of the the eigenvalues of the PCA allowing it to better represent the HDLSS data.

Another method that has been proposed for HDLSS datasets is to impose a sparsity constraint when the maximal eigenvector is sparse (Shen *et al.*, 2013). This would produce a sparse PCA (SPCA). Sparse PCA is found to be slightly more consistent as compared to the conventional PCA (Shen *et al.*, 2013). Given that α is the spike index whereby it is the measure of dominance of the first eigenvalue and β is the sparsity index which finds the number of non-zero values of the maximal eigenvector, SPCA is said to be consistent when $\alpha \leq 1$ and $\alpha > \beta$ (Shen *et al.*, 2013). However, SPCA becomes strongly inconsistent when $\alpha < \beta$. It is said to be marginally inconsistent when $\alpha = \beta$. Although the SPCA is more consistent than the classical PCA, it still has its inconsistencies (Shen *et al.*, 2013).

Hence, it can be seen that for HDLSS data whereby the number of dimensions, d, is more than the sample size,n, the effectiveness of the application of the classical PCA reduces. When the number of features increases above a certain limit of the linear subspace the points lie in, the principle components are only defined for the limit, m

number of features (Bishop, 2006). According to Bishop (2006), the eigenvalues will tend to be zero for at least $d - m$ number of features. Therefore, it can be said that classical PCA may not work as well for HDLSS datasets as it only works efficiently up till the number of sample size available and will not be able to account for many of the features.

2.3.2 Autoencoder

Autoencoders are a form of multilayer perceptron that work in an unsupervised manner. The goal of an autoencoder is to learn a hypothesis such that it is close to the identity function (Bengio *et al.*, 2015). This is to say that the $h_{w,b}(x) \approx x$. An autoencoder is also sometimes known as a bottleneck neural network. Autoencoders can take two forms, namely being undercomplete and overcomplete (Bengio *et al.*, 2015). An autoencoder which is undercomplete has fewer number of hidden nodes than input nodes while an overcomplete autoencoder has as many or more hidden nodes than input nodes. The goal of the undercomplete neural network is to learn salient features that can be found in the data whereas with an overcomplete autoencoder, certain rules or conditions such as sparsity needs to be enforced to ensure that salient features can be learnt (Bengio *et al.*, 2015).

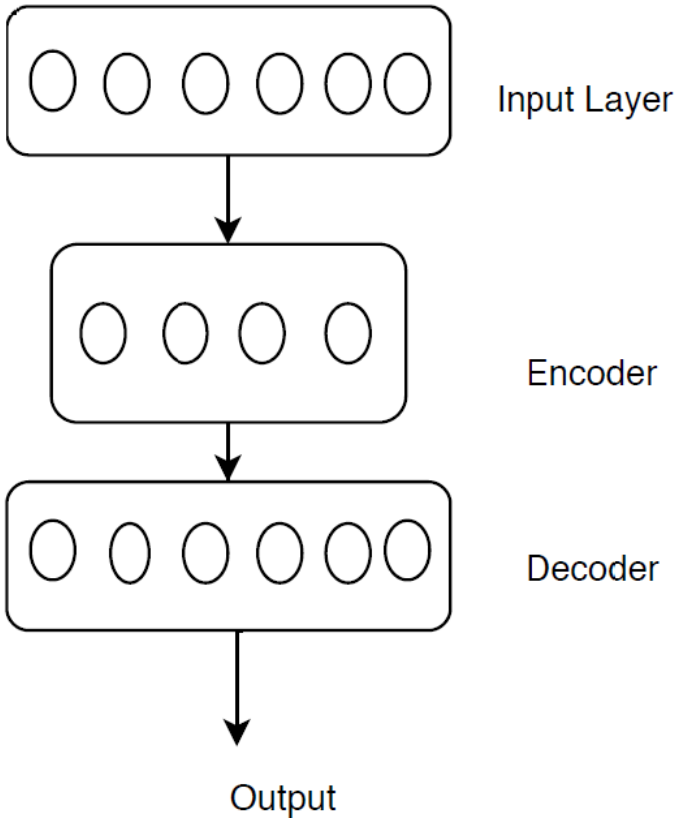


Figure 2.2: The figure shows an undercomplete autoencoder. Its encoder contains fewer nodes than its input. The decoder reconstructs the input from the data that was compressed.

Figure 2.2 above shows an undercomplete neural network since it has fewer hidden units than input units.

There are a few different categories of autoencoders. They are:-

1. Sparse Auntoencoder: In order to enforce the sparsity constraint, the activation of the hidden unit is set to as close to the non-firing criterion(in the case of the tanh activation function, -1) as possible (Ng, 2011)

   For inputs (x_1, x_2, x_3), that are drawn from a distribution D, that is i.i.d, the sparsity constraint can be written as (Ng, 2011):-

   $$E_{x\sim D}[a_i^{(l)}] = \rho \tag{2.27}$$

   where $E_{x\sim D}$ denotes the expected value of the activation function $a_i^{(l)}$, and ρ is the activation value the neurons should have (Ng, 2011). As an example in the case of a tanh activation function, $\rho = -0.9$ to ensure that it is close to being as inactive as possible (Ng, 2011).

2. Denoising autoencoder: A denoising autoencoder (DAE) is often used to learn features from a set of inputs that are corrupted with noise (Bengio *et al.*, 2015). The goal of the autoencoder is to predict the uncorrupted data as its output (Bengio *et al.*, 2015). It takes in a training pair $(x, \hat{x})$, where x is the original data sample and $\hat{x}$ is the corrupted data. The corrupted data was constructed from the original data by applying some noise through a corruption process $C(\hat{x}|x)$ (Bengio *et al.*, 2013).

3. Stacked Autoencoder: A stacked autoencoder works by pretraining each layer one at a time. So then the first hidden layer and the second hidden layers are pretrained in a greedy layer-wise manner. This process continues until the final layer is pretrained. All the layers are combined together thereafter to create a stacked autoencoder.

4. Variational Autoencoder: A variational autoencoder is different from the standard autoencoders in that it can be used to generate samples. Their latent space is continuous. Instead of outputting one vector after the encoder phase, it outputs two vectors, i.e. mean and standard deviation. The samples that are modelled using the mean and standard deviations are then used to train the decoder. In order to ensure that the encodings are close to each other but still distinct, Kullback–Leibler divergence is minimized.

Xue *et al.* (2017) studied the effect of insufficient data for stacked autoencoders. It was shown that the lack in samples causes a drastic drop in performance of the dataset. Data augmentation and transfer learning was used to test whether there would be any improvement in accuracy. Their findings show that while data augmentation does not improve the accuracy, transferring useful knowledge does improve classification accuracy. Liu *et al.* (2017b) proposed a sample expansion method which corrupts the data to 0. Using this method, they are able to generate sufficient samples to train a stacked autoencoder. Reasonable sample size is require for autoencoders to perform well. Preliminary experiments were conducted to test the feasibility of autoencoders under the HDLSS setting. It was found that it does not perform as well as some of the other methods tested. One drawback of neural network based methods is the number of parameters to optimise in order to attain a reasonable result. The experiment setup and results can be found in Appendix A.

2.3.3 Information Theoretic Feature Extraction Methods

In supervised information theoretic based feature extraction, features with high MI w.r.t. the class labels are generally selected for classification. Some popular supervised information theoretic feature extraction methods include Maximization of Mutual Information (MMI) by Torkkola (2003), Maximization of Mutual Information by Leiva-Murillo and Artés-Rodríguez (2007), Renyi's Entropy Discriminant Analysis (REDA) by Yuan and Hu (2009), and Information Discriminant Analysis (IDA) (Nenadic, 2007). In unsupervised information theoretic based feature extraction methods, features are extracted such that information within the data is maximally retained with minimal redundancy among the extracted features. Independent Component Analysis (ICA) (Aapo Hyvärinen, 1997), and Kernel Entropy Component Analysis (KECA) (Jenssen, 2010) are two representatives of feature extraction methods. Brown *et al.* (2012) did a study on the effects of small sample size on information theory based feature selection methods. They found that some of the methods which does not balance the effects of redundancy and relevancy such as CIFE degenerates. They created a probabilistic framework to incorporate terms such as redundancy and relevancy. Taking motivation from them, we show that the information theory based feature extraction methods follow the general pattern of minimizing or maximising $D_\alpha - \beta$, where D_α stands for a divergence measure with order of α, and β a penalty function imposed onto the respective objective functions. Using the operational definition from Harremoës (2006), the general pattern indicates finding a compression between two probabilistic mixtures w.r.t some additional cost or penalty function. The penalty function allows a less noisy and redundant set of features to be extracted.

Maximization of Mutual Information

MMI (Torkkola, 2003) learns discriminant feature transforms using mutual information between the class labels and transformed data. It uses quadratic divergence measures. Its objective is to find the distribution which maximises the quadratic mutual information. The Parzen-window estimation method was used to estimate the density of the probability distribution function (PDF). A spherical Gaussian kernel was used with the Parzen density estimator.

The quadratic mutual information is defined (Torkkola, 2003) as follows:

$$I(C,Y) = V_{IN} + V_{ALL} - 2V_{BTW} \tag{2.28}$$

where V_{IN}, V_{ALL}, V_{BTW} are information potentials.

It is then given as:

$$I(C,Y) = \sum_c \int_y p(c,y)^2 dy + \sum_c \int_y P(c)^2 p(y)^2 dy - 2\sum_c \int_y p(c,y)P(c)p(y)dy \quad (2.29)$$

where Y is the transformed continuous variable such that $Y = g(W,X)$ where W is a weight matrix and g is the transformation function, c is a class label such that $c \in C$ and P is a discrete distribution.

The objective function to maximise the mutual information is given by:

$$W = \text{argmax}_W(I(C,Y)) \quad (2.30)$$

subject to $W^T W = \mathbf{I}$ where $\mathbf{I}$ is the identity function. Gradient ascent was performed on the objective function. The objective function can be rewritten into the general form as follows:

$$\max D_2(p(y,c)||p(y)p(c)) \quad (2.31)$$

where the divergence is a quadratic divergence measure given by order of $\alpha = 2$. In this case, it only takes into consideration the mutual information. β is set to 0 as it does not consider any other factors. Through the general framework, it can be seen that a compression between the class labels and the input data is being found.

The code used for the experiments can be found at https://github.com/ktorkkola/mmi-on-line.

Conditional Infomax Feature Extraction(CIFE)

CIFE selects features by ensuring reduction of class-relevant redundancies and the mutual information among selected features given the class label is maximised (Lin and Tang, 2006). Shannon's entropy was used and the objective function for an unselected feature, X_k maximises the following function:-

$$J(X_k) = I(X_k;C) - \sum_{X_j \in S} I(X_j;X_k) + \sum_{X_j \in S} I(X_j;X_k|C) \quad (2.32)$$

where C is the class label, and S the sample feature set that has been selected. Parzen windows with a Gaussian kernel were used to estimate the pdf.

The objective function can be rewritten in a general form in terms of Renyi's divergence as follows:

$$\max_{X_k} D_1(p(X_k,C)||p(X_k)P(C)) - \beta \quad (2.33)$$

where D_1, taking $\lim_{\alpha \to 1}$, KL divergence is a special case of Renyi's divergence, and $\beta = \sum_{X_j \in S} I(X_j;X_k) - \sum_{X_j \in S} I(X_j;X_k|C)$ denotes the class-relevant redundancy. It compresses information between class label and the input data whilst ensuring that any

redundancy among selected features is minimized without losing the feature information.

The code used in this experiment is available from Li *et al.* (2016).

Maximum Relevance Minimum Redundancy (MRMR)

The MRMR method was one of the best performing feature selection methods for extremely small sample size datasets (Brown *et al.*, 2012). An unselected feature X_k is to be selected by maximizing the following objective function:

$$J(X_k) = I(X_k;C) - \frac{1}{|S|}\sum_{X_j \in S} I(X_j;X_k) \tag{2.34}$$

where S is the set of selected features. The density estimation is done using Parzen windows. MRMR balances the effects of redundancy and relevance while CIFE does not (Brown *et al.*, 2012).

The objective function can be rewritten as follows:

$$\max_{X_k} D_1(p(X_k,C)||p(X_k)p(C)) - \beta \tag{2.35}$$

where D_1 is Renyi's divergence and taking $\lim_{\alpha\to 1}$, KL divergence is a special case of Renyi's divergence and $\beta = \frac{1}{|S|}\sum_{X_j \in S} I(X_j;X_k)$ which represents redundancy. The main difference between equation (2.34) and equation (2.35) is the term $\sum_{X_j \in S} I(X_j;X_k)$ that was used in MRMR to compensate for information loss in CIFE.

The code used in this paper can be obtained from Li *et al.* (2016).

Kernel Entropy Component Analysis(KECA)

KECA (Jenssen, 2010) reveals the structure of the dataset by relating to Renyi's entropy, estimated via kernel matrix and Parzen windowing. KECA is an unsupervised feature extraction method. Renyi's quadratic entropy is given by:

$$H_2(p) = -\log \int p^2(x)\,dx \tag{2.36}$$

where $p(x)$ is estimated by the Parzen window density estimator given by:

$$\hat{p}(x) = \frac{1}{d}\sum_{x_i \in X} k_\sigma(x,x_i) \tag{2.37}$$

where $k_\sigma(x,x_i)$ takes the form of the Gaussian kernel, where x_i is a sample in $X = X_1, X_2, \ldots, X_d$. Using the sample mean approximation of the expectation and by removing the log from Renyi's entropy as it is a monotonic function, the following is obtained:

$$V(p) = \frac{1}{d^2}\mathbf{1}^T K\mathbf{1} \tag{2.38}$$

where $\mathbf{1}$ is a $(d \times 1)$ vector where each element equals to 1 and K equals to $k_\sigma(x_i, x_{i'})$ where $i' \neq i$ and d is the number of dimensions.

If the Parzen window function used is positive semi-definite, the Renyi's entropy estimator is composed of all the projections onto the kernel PCA axes. Kernel PCA selects eigenvalues and eigenvectors based on the magnitude of the eigenvalues while KECA does dimensionality reduction by projecting the data on the kernel PCA axis which contributes most to the entropy estimate of the data Jenssen (2010).

KECA's transformation is given by the minimisation of the function

$$V(p) - V_k(p) \tag{2.39}$$

where a subset of k eigenvalues are selected to represent the transformation.

Since the objective function is derived from Renyi's quadratic entropy, it can be rewritten as:

$$\max D_{2-\alpha}(P_{diag}||P_k \times P_k) - H_2(X) \tag{2.40}$$

where Renyi's entropy is defined in terms of Renyi's divergence as $D_{2-\alpha}(P_{k_{diag}}||P_k \times P_k)$, P_{diag} is the distribution of (X, X), and when $\alpha \to 1$, the right-hand side of the divergence becomes the mutual information between X and itself $\beta = H_2(X)$ where H_2 is Renyi's entropy with order of $\alpha = 2$. This shows that the compression is based upon self-information. It ensures that minimal loss of information occurs during the compression.

The code is available online at https://github.com/IPL-UV/simFeat and has been used in the comparative study here.

Independent Component Analysis

Independent Component Analysis (ICA) (Comon, 1994) assumes that a random observed vector, X, is a mixture of independent elements of a random vector S, given by:

$$X = AS \tag{2.41}$$

where A is a mixing matrix. ICA tries to find the unmixing matrix W, so that will give Y, the best approximation of S.

$$Y = WX \tag{2.42}$$

To measure the independence between the sources and the mixture, minimization of mutual information can be used (Aapo Hyvärinen, 1997). The matrix W is selected such that the mutual information between components of Y is minimized. ICA that uses minimization of mutual information is known as InfoMax ICA.

The minimization of mutual information of $Y = Y_1, \ldots, Y_n$ as its objective function can be written in terms of the general form using Renyi's divergence as :

$$\max D_1(p(Y)||\prod_{i=1}^{n} p(Y_i)) \tag{2.43}$$

where $Y_i \in Y$. β is defined as 0 in this case. Using the operational definition, it can be observed as the compression of the information among the components.

The code is available online at https://github.com/alvarouc/ica and was used in this study.

Information Theory based feature extraction and selection methods are able to capture higher order information about the data. This indicates the possibility of supplementing the lack of samples with the additional information captured in the HDLSS setting.

2.3.4 Manifold Learning

The main aim of manifold learning is to learn the underlying low dimensional manifold where geometric relationships between most observed data are preserved (Huo *et al.*, 2007). It is essentially a method to reduce the feature space.

Given a dataset $X = \{\vec{x}_1, \ldots, \vec{x}_n\} \in \mathbb{R}^M$ find points $\vec{y}_1, \ldots, \vec{y}_n \in \mathbb{R}^m$ where $M \geq m$, such that $\vec{y}_i$ represents $\vec{x}_i$ in lower manifold space where $\mathcal{M}$ is a manifold embedded in $\mathbb{R}_m$. Some of the common manifold methods include ISOMAP, Laplacian Eigenmaps, and Locally Linear Embedding(LLE).

ISOMAP (Tenenbaum *et al.*, 2000) is an algorithm that assumes that isometric properties are preserved and it should be observed in both its intrinsic embedded space and observation space. It preserves geodesic interpoint distances. ISOMAP makes the premise that Euclidean distance is deceptive in high dimensional spaces (Huo *et al.*, 2007). However, it searches for the nearest neighbour relations via Euclidean distance, $d_x(i, j)$, in its input space X. Further representation is done using a graph, G over the data points, with weight $d_x(i, j)$ assigned to the corresponding edges. The pairwise geodesic distance on manifold, $\mathcal{M}$ is the shortest path in the graph. For neighbouring points, the initial distance is considered a good approximation to their geodesic distance. Whereas for far away points, the sequence between the neighbouring points are added up. Multidimensional Scaling algorithm is applied to the matrix

of geodesic distance to construct the embedding. The usage of Euclidean distance to initialise the adjacency graph could lead to the weights being all the same due to points being equidistant for HDLSS data.

An enhanced version of ISOMAP which is supervised was suggested by (Ribeiro *et al.*, 2008). In this version, once the lower dimensional manifold embedding is found, a dissimilarity measure, D is used to integrate the class label (Chao *et al.*, 2019).

$$D(x_i, x_j) = \begin{cases} (\frac{a-1}{a})^{\frac{1}{2}} & \text{if} c_i = c_j \\ a^{\frac{1}{2}} - d_0 & \text{if} c_i \neq c_j \end{cases} \tag{2.44}$$

where $a = \frac{1}{e^{(-d_{ij}^2/\sigma)}}$, d_{ij} is a distance measure, σ is a smoothness parameter, and $0 \leq d_0 \leq 1$ is a constant.

Locally Linear Embedding and Laplacian Eigenmap focuses on preserving local neighbour structure. Locally Linear Embedding (LLE) (Roweis and Saul, 2000) establishes the mapping relationship between the observed data and the corresponding low dimensional data. Each sample in the observation space is a linearly weighted average of of its neighbours. It first finds n nearest neighbours for each data point x_i in a dataset. It assumes that each data point and neighbour lie on a locally linear patch of the manifold. Linear coefficients are added to reconstruct each data point from its neighbour. The sum of the squared distance between points and reconstruction error is given by:

$$\begin{aligned} q(W) &= ||x_i - \sum_{j=1}^{K} W_{ij} x_{ij}||^2, \\ \text{s.t.} \sum_{j=1}^{n} W_{ij} &= 1 \\ W_{ij} &= 0 \text{ iff } x_j \neq \text{neighbour to } x_i \end{aligned} \tag{2.45}$$

where the weight matrix $W = [W_{ij}]^{n \times n}$ summarizes the contribution of the j^{th} neighbour to the reconstruction of the i^{th} point. These constraints make them invariant to rotation, rescaling, and translation. The weight W_{ij} is chosen by minimising the cost function given in Eq.(2.45). W_{ij} reflects the intrinsic geometric properties of the data point. The lower manifold is then reconstructed by using the final form of the weight matrix, $Y = \text{argmin}_Y q(Y)$. LLE is made into a supervised manifold learning metric (Zhang, 2009) by modifying the dissimilarity used to rank the nearest neighbour.

Laplacian Eigenmaps (LE) (Belkin and Niyogi, 2003) is a locality preserving non-linear dimension reduction method. The adjacency graph is plotted using the criterion such as nearest neighbour or squared Euclidean distance. The edges are weighted with

either rigid weights (i.e. 1 if connected, 0 otherwise) or using soft weights (i.e. interval of [0,1]). Once the adjacency map is connected, the generalized eigen decomposition problem is calculated and solved. $Lf = \lambda Df$, where D is the diagonal weight matrix, the entries of which are the column sum of W, $D_{ii} = \sum_j W_{ji}$ and L is the Laplacian matrix. λ is the eigenvalues and f the eigenvectors.

Raducanu and Dornaika (2012) borrowed the idea from LDA to propose a supervised LE. They minimized the margin between the homogeneous data and maximised the margin between the heterogeneous data. They modified the weight matrix in order to do so. Zheng *et al.* (2008) integrated the label information by optimizing the weight matrix. The weight matrix was optimized using the labels after the similarity matrix was formed from the local neighbourhood relations.

Manifold learning is able to learn the intrinsic dimensionality of a dataset. However, it does require sufficient amounts of data to be able to accurately reduce the dimensionality of the data.

2.4 Investigation of Intrinsic Structure Models for HDLSS Conditions

2.4.1 Fundamentals of Probabilistic Models

Graphical models are used to describe the dependence and independence between variables which are modelled as distributions. It enables visual interactions between variables. It can be defined in terms of directed or undirected graphical models. A graph $G(V,E)$ has vertices also known as nodes, V, which represents the random variables and edges, E, which represents the probabilistic relationships between the random variables. The graph, G, models the joint distribution and the decomposition of its factors. Given a joint distribution of 3 random variables $X = \{x_1, x_2, x_3\}$ where the lowercase x refers to the values of a random variable, using the product rule of probability (Bishop, 2006), the joint distribution decomposes to:-

$$p(x) = p(x_3|x_1, x_2)\, p(x_1, x_2) \tag{2.46}$$

Applying the product rule of probability for a second time to Eq. (2.46) will give

$$p(x) = p(x_3|x_1, x_2)\, p(x_1|x_2)\, p(x_1) \tag{2.47}$$

This would produce the following graphical structure:

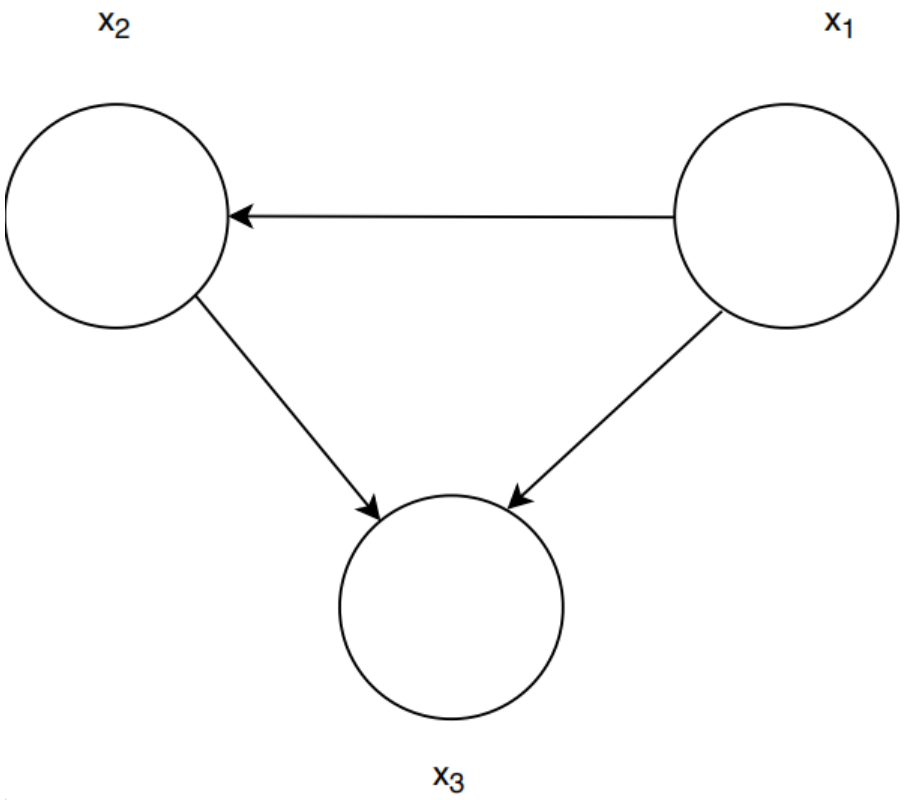


Figure 2.3: The figure shows the visual representation of the decomposition of the conditional distribution in Eq.(2.47).

Typically, a graphical model has two tasks:

1. Inferencing: Inferencing is the querying of some information regarding some unobserved random variables through a certain graphical model, M. There are 3 types of problems which are solved using inferencing.

   - Marginal inference: $P(X_1)$
   - Posterior inference: $P(X_1|X_2 = x_2)$
   - Maximum-a-posteriori inference: argmax $P(X_1|X_2 = x_2)$

   Inferencing methods can be further divided based on the methodology, i.e exact inferencing or approximate inferencing.

2. Learning: Estimating the plausible graphical model, M, from the observed data, X. In the case of Bayesian networks, model estimation is done through $P(M|X)$. It becomes an inferencing problem.

In our work, we will be focusing on directed graphical models.

Bayesian Network

Bayesian Networks are also known as Directed Acyclic Graphs (DAG). Bayesian Networks represents a set of random variables as causal relationship. Bayesian Networks can be grouped into 3 graph structures according to the conditional independence criteria.

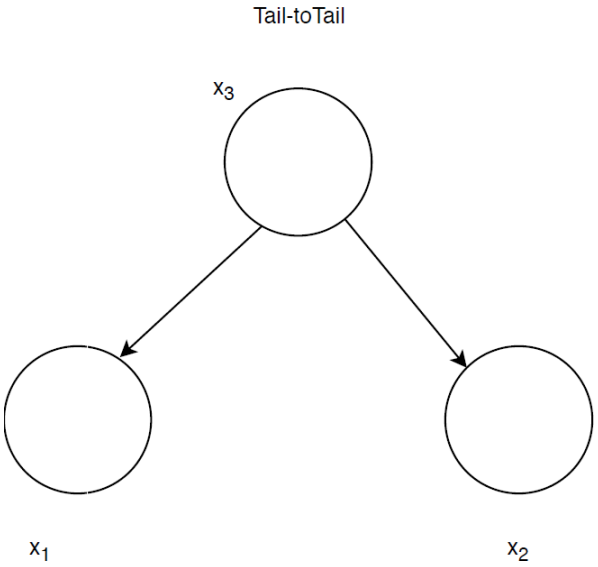


Figure 2.4: The figure shows the connection between a tail-to-tail graph.

1. In the case that x_3 is observed in Fig.(2.4), it follows that $x_1 \perp\!\!\!\perp x_2 \mid x_3$
2. In the case that none of the variables are observed in Fig.(2.4), it follows that $x_1 \not\perp\!\!\!\perp x_2 \mid \emptyset$

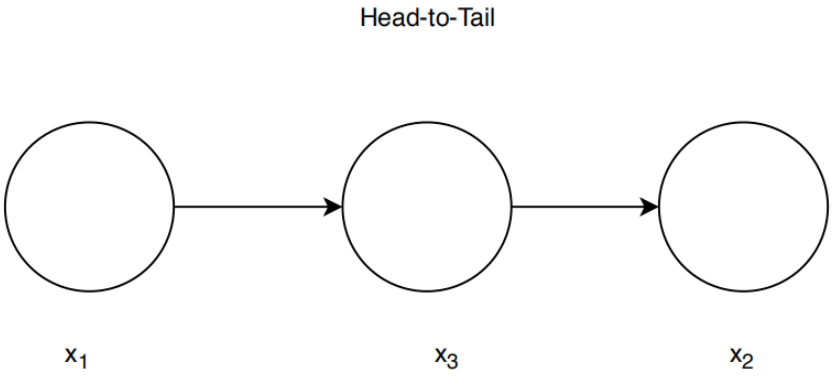


Figure 2.5: The figure shows the connection between a head-to-tail graph.

1. In the case x_3 is observed in Fig.(2.5), it follows that $x_1 \perp\!\!\!\perp x_2 \mid x_3$

2. In the case none of the random variables are observed in Fig.(2.5), it follows that $x_1 \not\perp\!\!\!\perp x_2 \mid \emptyset$

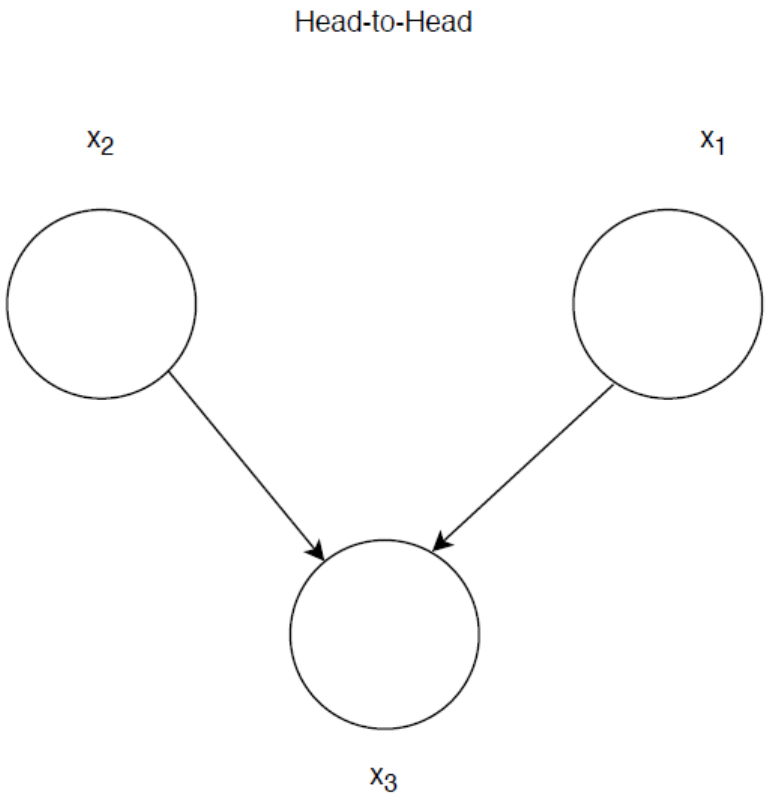


Figure 2.6: The figure shows the connection between a head-to-head graph.

1. In the case that x_3 is observed in Fig.(2.6), it follows that $x_1 \not\perp\!\!\!\perp x_2 \mid x_3$

2. In the case that none of the random variables are observed in Fig.(2.6), it follows that $x_1 \perp\!\!\!\perp x_2 \mid \emptyset$

A d-separated graph implies conditional independence, i.e. $x_1 \perp\!\!\!\perp x_2 \mid x_3$. x_1 is said to be d-separated from x_2 by x_3 if all the paths are blocked. A path is blocked in the case of head-to-tail or tail-to-tail when the node is in x_3. In the case of head-to-head graph, a path is blocked if the observed node is not in set x_3. The d-separation property in graphical models simplifies the structures in the graphical model and the computation required to perform inference and learning (Bishop, 2006).

When dealing with Bayesian Networks, the main goal is generally to learn the posterior belief of the latent variable, Y given the observed variable X, due to its applica-

bility in areas of diagnosis and prediction. This is given by:

$$P(Y|X) = \frac{P(Y,X)}{\sum_y P(Y=y|X)} \tag{2.48}$$

However, computing $P(Y = y|X)$ is NP hard. Hence, various inferencing techniques combined with assumptions made through the d-separation property are used in order to lower the computational complexity.

As mentioned, there are two broad categories of inferencing for probabilistic graphical models, i.e. exact inferencing and approximate inferencing. Exact inference can be hard to compute due to high time complexity especially as the feature space grows. In this case, approximate inference is used instead. An example of exact inference method is variable elimination. Mean field inference is a type of approximate inference method.

Variable Elimination

Variable Elimination is a form of exact inference. It assumes a causal relationship between variables. This is to say, $X_1 \rightarrow X_2 \rightarrow X_3 \rightarrow X_4 \rightarrow X_5$. By exploiting this structure, the joint probability can be defined as

$$P(\vec{X}) = \sum_{x_1} P(X_1) \cdot P(X_2|X_1) \sum_{x_2:x_5} P(X_3|X_2) \cdot P(X_4|X_3) \cdot P(X_5|X_4) \tag{2.49}$$

The summation process eliminates the variables. The process of summation is repeated for each variable.

In general, the algorithm is given as:

$$P(X_1,\ldots,X_5) = \phi_{X_1}(X_1) \cdot \phi_{X_2}(X_1,X_2) \cdot \phi_{X_3}(X_2,X_3) \cdot \cdots \cdot \phi_{X_5}(X_4,X_5) \tag{2.50}$$

$$\psi(X_1,X_2) = \phi_{X_1} \cdot \phi_{X_2} \tag{2.51}$$

Marginalising X_1:

$$\tau_1(X_2) = \sum_{x_1} \psi(X_1,X_2) \tag{2.52}$$

These equations are repeated for all ϕ, where ϕ is a factor.

Mean Field Inference

Mean Field Inference is a variational inferencing method. A variational inference infers the conditional distribution over a set of latent variables given the samples. It is also known as Bayesian inference. The posterior,

$$P(Y|X,\alpha) = \frac{P(Y,X|\alpha)}{\int_Y P(Y,X|\alpha)} \tag{2.53}$$

has an integral that cannot be computed analytically. Instead, we choose a family of distributions to approximate the true distribution. Mean field inferencing assumes the independence over the latent variables. Given q is the approximated probability, the joint probability of the latent variables, Y can be factorised as follows:

$$q(Y_1, \ldots, Y_M) = \prod_{j=1}^{M} q(Y_j) \tag{2.54}$$

Inferencing in Bayesian Network also corresponds to its learning. Hence, evidence lower bound (ELBO) is used to compute the similarity between the two distributions. Decomposing ELBO using the mean field variational inferencing method, Bayesian network maximizes over

$$\text{argmax}\, L = \log P(X_{1:n}) + \sum_{j}^{M} E[\log P(Y_j | Y_{1:j-1}, X_{1:n})] - E_{q_j}[\log q(Y_j)] \tag{2.55}$$

where Y is the latent variable.

Bayesian network uses Bayesian inference to find the posterior distribution. Bayes' theorem is used to update the hypothesis of the posterior distribution. Bayes' theorem is given by:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)} \tag{2.56}$$

where Y is the hypothesis and X is the observed data.

2.4.2 Neural Networks

Neural networks are a type of graphical model. Neural networks was introduced by McCulloch and Pits in the 1940s (Gershenson, 2003). A two layer network with a perceptron that was able to learn classification problems through the usage of weights was later introduced by Rosenblatt in the 1950s (Gershenson, 2003). Artificial neural networks have drawn inspiration from biological neural networks. It acts to mimic the processes that occur in the biological neuron by viewing its nodes as artificial neurons (Gershenson, 2003).

The concept of artificial neural networks is not as sophisticated as the biological neurons. In an artificial neural network, the inputs which act as synapses are multiplied by weights which are then computed through a mathematical function to determine the activation of the neuron (Karpathy, 2016). The output can be computed using other functions such as the identity function based on the purpose of the neurons.

FeedFoward Neural Networks

A feedforward neural network is a network that only performs forward pass (Gershenson, 2003). This means that there is no back-propagation nor error correction.

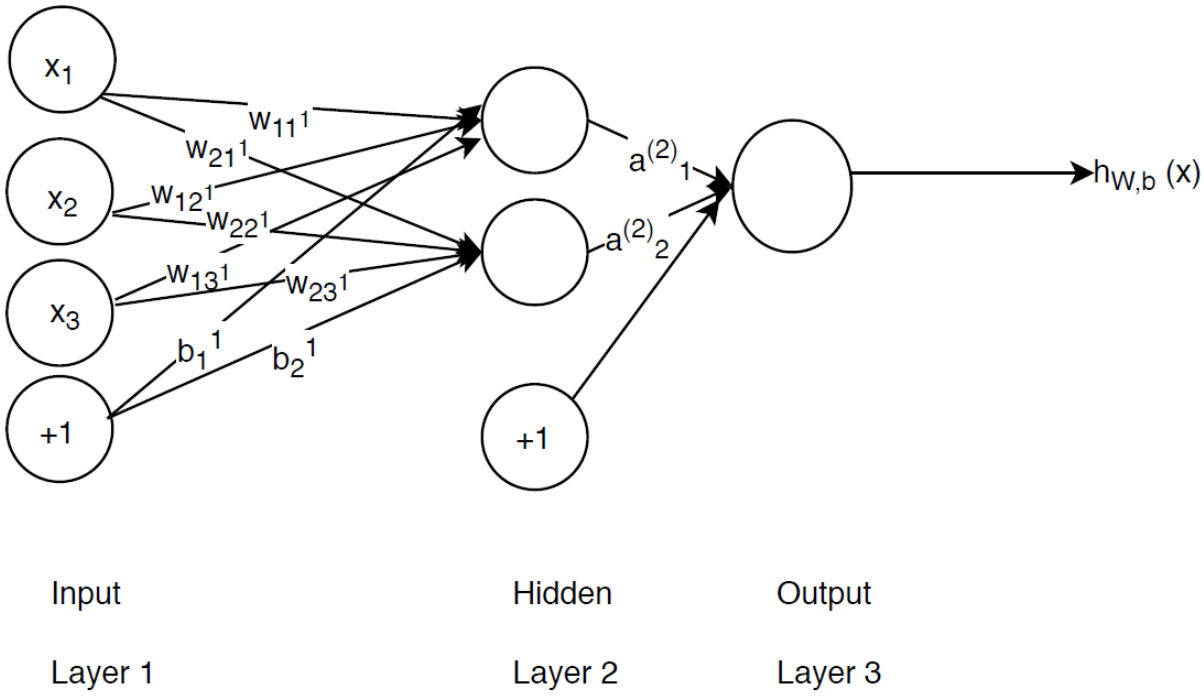


Figure 2.7: The figure shows a neural network with an input, a hidden layer and an output. The hidden layer contains 2 hidden nodes. This form of neural network is known as a bottleneck neural network.

x_1, x_2, x_3 are the inputs. A neural network can take many inputs. In this example we are only going to be looking at 3 inputs. $h_{w,b}(x)$ is the output value produced (Ng, 2011). The bias for unit i which is associated in layer $l+1$ is given by b_i^l. The weight or parameter that is associated with unit i in the layer $l+1$ and unit j in the layer l, is denoted by w_{ij}^l. a_i^l is the activation function, which is also known as the output value of unit i in layer l (Ng, 2011).

For $l = 1$, the activation value is given by $a_i^{(1)} = x_i$ where x_i corresponds to the inputs (Ng, 2011).

$$a_1^{(2)} = f(w_{11}^1 x_1 + w_{12}^1 x_2 + w_{13}^1 x_3 + b_1^{(1)}) = f(z_1^{(2)}) \tag{2.57}$$

$$a_2^{(2)} = f(w_{21}^1 x_1 + w_{22}^1 x_2 + w_{23}^1 x_3 + b_1^{(1)}) = f(z_2^{(2)}) \tag{2.58}$$

$$h_{w,b}(x) = a_1^{(3)} = f(w_{11}^2 a_1^{(2)} + w_{12}^2 a_{(2)}^2 + w_{13}^2 a_3^{(2)} + b_2^{(1)}) = f(z_1^{(3)}) \tag{2.59}$$

The notation f can be viewed as a tanh function or any other type of activation function such as the sigmoid function.

Backpropagation

Backpropagation is usually used to correct or lower the error by updating the weights and/or biases. Backpropagation is also known as stochastic gradient descent (Ng, 2011). The backpropagation algorithm is used as a backwards pass which can automate the computation of the error correction (Ng, 2011).

The algorithm can be given as (Ng, 2011):-

---

for all $(x^{(i)}, y^{(i)}), i = 1,2,3..$ **do**

Update weight $w_{jk}^{(l)} = w_{jk}^{(l)} - \alpha \frac{\delta}{\delta w_{jk}^{l}} J$,

Update the bias term $b_j^{(l)} = b_j^{(l)} - \alpha \frac{\delta}{\delta b_j^{l}} J$

where $x^{(i)}$ is the input value ,$y^{(i)}$ is the output value or class corresponding to the input, α is the learning rate and J is the cost function with respect to the respective training example.

$J = \frac{1}{2}||h_{w,b}(x) - y||^2 + \frac{\lambda}{2} \sum_l \sum_i \sum_j (w_{ij}^{(l)})^2$

The second part of the cost function is the weight decay term (Ng, 2011). The weight decay term in this case is the squared error penalty on the value of parameter, w. This is the term which drives $w \to 0$. This function helps to prevent overfitting (Ng, 2011). The initialising of the weight, w cannot be all set to 0 as the hidden neurons will have the same values all the time. The value of $\delta_i^{(l)}$ measures the responsibility of each node for the error in the output. λ is the weight decay parameter.

The algorithm for the neural network is as follows (Ng, 2011):

A feedfoward pass is done once to compute the activation functions of all the neurons.

for all nodes in the output layer (outermost only) denoted by n_l **do**

$\delta^{n_l} = -(y_i - a_i^{n_l}.f'(z_i^{(n_l)}))$

y_i is the target value and $a_i^{n_l}$ is the actual value.

for all layer $l = n_{l-1}, n_{l-2}, .., 2$ **do**

$\delta_i^l = -(\sum w_{ij}^{(l)} \delta_j^{(l+1)}.f'(z_i^{(l)})$

update the parameters l, the weight w and the bias, b

$w_{ij}^{(l)} = w_{ij}^{(l)} - \alpha(a_j^{(l)} \delta_i^{(l+1)} + \lambda w_{ij}^{(l)})$

$b_i^{(l)} = b_i^{(l)} - \alpha \delta_i^{(l+1)}$

Deep Neural Network

A deep neural network is different from the shallow networks in that it works on depth. A neural network system which has 4 or more layers is considered a deep neural network (Chris Nicholson, 2016). Each layer in a deep neural network trains on the output from the previous layer to learn salient features. As the depth of the neural network gets deeper, the more abstract the features learnt. Deep learning can be used as an automatic feature extraction method. Some of the deep learning networks include deep autoencoder (sometimes known as stacked autoencoder) and convolutional neural networks. Deep neural networks generally require a large number of samples in order to tune the hyperparameters. It's complexity increases as the dimensions increase.

A convolutional neural network's connections are restricted such that only a small subset of inputs and hidden units are connected (Bengio *et al.*, 2015). This will reduce the number of hyperparameters that is required to be tuned, hence reducing the number of samples required. However, it produces other complexities such as the explainability of the data which is desirable in many domains. Traditionally, neural networks that use matrix multiplication requires a matrix of parameters to indicate the relationship of each parameter describing the interaction between the input and output layers. Using kernels, its size can be reduced compared to the original input size as the kernels should be able to identify important features such as edges (Bengio *et al.*, 2015).

Parameter sharing means that a parameter can be used for more than one function in a model. Traditional neural networks have separate weights for each node, whereas having one parameter for multiple functions is somewhat similar to having a tied weight (Le *et al.*, 2010). Tied weights are weights that are applied to an input which is tied or has the same value applied in other functions in the model (Le *et al.*,

2010). Learning just one set of parameters instead of multiple separate sets of parameters has the added advantage of using less storage space (memory) as well as reduced complexity of the network.

Recent developments in deep learning for the use of HDLSS data was done by Liu *et al.* (2017a) through their model Deep Neural Pursuit (DNP). In their study, in order to deal with the extremely high dimensionality, they set the weights to 0 initially and graft in nodes. l_1 norm constraint is used on the input weights. This is essentially feature selection integrated with neural networks. A similar concept to this is the dropout regulariser. Dropout as its name suggests, aims to drop units from a neural networks model along with the incoming and outgoing connections. This is to prevent the neurons from co-adapting with each other (Hinton, 2014). Dropout can reduce overfitting and gives a better performance than when just using normal regularizers such as max-norm on the neural network models. The dropout method aims to reduce overfitting and provides a new way to combine different neural network architectures together. One of the effect of dropout is that it is able to induce sparsity. This ensures that most of the neurons become non-firing. However, unlike dropout which starts with a partially connected network, Liu *et al.* (2017a) starts off with no connections. While deep neural networks show some promise under the HDLSS structure, more work needs to go into making neural networks more interpretable.

2.4.3 Correlation Explanation

Correlation Explanation (CorEx) was first introduced in NIPS in 2014 by Ver Steeg and Galstyan (2014a). It is an information-theoretic method to learn representations of high dimensional data. It is able to work even with small sample sizes. The reason for this is that the estimation of the probability distribution is through the marginal $p(x)$ rather than estimating the full probability (Ver Steeg and Galstyan, 2015). By estimating the marginal, Chernoff bound guarantees that the estimation error decreases exponentially. This implies less samples are required to give a good estimate. CorEx also does not require any prior knowledge. It searches for latent factors that are able to explain correlations among the data using total correlation in a hierarchical fashion. It learns in an unsupervised manner.

Formally, given that X is discrete, the aim is to learn hidden factors $Y = \{y_1, y_2, .., y_m\}$ where m is the number of latent features. Each of the latent features can take a number of discrete values, k. The total correlation of X after being conditioned on the latent variable Y is given by:

$$TC(X;Y) = TC(X) - TC(X|Y) \tag{2.60}$$

Eq. (2.60) measures the extent to which the latent features Y explains the correlations in X. Hence the higher the value of $TC(X;Y)$ the more Y explains X. Optimizing eq. (2.60) which searches for each latent factor Y gives us :

$$\max_{p(y_j|x)} TC(X;Y) \text{ s.t. } |Y| = k \tag{2.61}$$

The objective function $TC(X;Y)$ can be rewritten to conform to a pattern of minimising Renyi's divergence with a penalty function, β, as follows:

$$\min D_1(p(X|Y)||\prod_{i=1}^{d} p(X_i|Y)) - TC(X) \tag{2.62}$$

where $\beta = TC(X)$ and $x_i \in X$. This shows that CorEx looks for the compression between the original group of features after being conditioned on the latent features but ensuring that as much information from the original is preserved. One advantage which CorEx has over neural networks is that features that contribute to the connections can be determined.

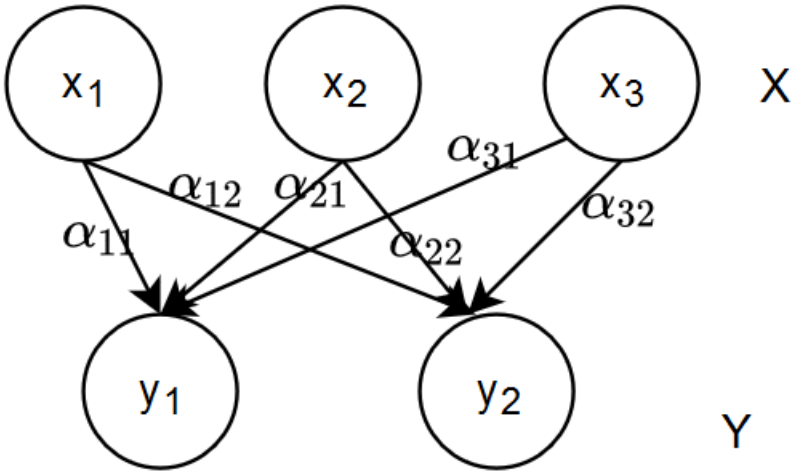


Figure 2.8: The figure shows the structure of CorEx.

CorEx takes in values from each feature in a dataset into each corresponding node in X. This means that each feature in the dataset will be associated with an input node in CorEx. Let x_1, x_2, and x_3 be 3 input features in the dataset, and y_1, and y_2 are two hidden nodes in the CorEx structure. The maximum number of latent nodes, Y, which can be defined are upper bounded by the number of input feature in X. Each factor y_j can be considered a function of x_i, i.e. $y_j = f_j(x_i)$.The α-values are initialised using Dirichlet distribution. The α-values represent the degree of influence each y_j has on each x_i. The relationship between the input features can be characterised by

the strength of the α-values and the connectivity to a hidden node. At the very beginning, the model has full connectivity and the $TC(X;Y) = 0$ because nothing is learnt. Through various iterations, it stochastically updates the α-values until all the hidden nodes have found the input variables for which the correlations can be explained, i.e. Eq. (2.62)is minimized.

In the literature, Correlation Explanation has been used to analyse gene expression data whereby the data is split into two sets according to class information (Pepke and Ver Steeg, 2017). The model was trained with samples representing only one class. The second class was then run through the already trained model. A histogram was used to map the features which the factor labels showed the greatest difference between the classes when CorEx was applied. It was used as a clustering type algorithm and its use for classification has not been fully explored. A formulation called anchored CorEx was introduced by Gallagher *et al.* (2016) for topic modelling. It uses information bottleneck to preserve document relevance. Words were ranked according to the mutual information to the topic and chosen to be used for document classification. Hence, it can be seen that CorEx has the potential to be extended for classification purposes.

Unlike Neural Networks, the full potential of Correlation Explanation as a supervised model has not been fully explored. It also has the potential to work under HDLSS circumstances due to the formulation of the estimation of the probability density distribution. Further, Correlation Explanation also uses total correlation which would be able to capture higher order statistics. This would be useful to learn underlying relationships between features. The relationships between the features can be captured and weighed. This feature of CorEx, forms an additional benefit. As such, we have chosen to select CorEx as our case study method.

2.5 Summary

The geometric structure of data points that are drawn independently from a distribution converges to become equidistant in the Euclidean space as the feature space increases (Hall *et al.*, 2005). This means that in HDLSS settings, algorithms which use Euclidean distance may not be able to perform well. Examples of such algorithms are K-Nearest Neighbour and Metric Learning. Generally, estimation of covariance matrix requires the number of samples to exceed the number of features. Hence, when the number of samples are not sufficient, the covariance structure could be incorrectly generated which could lead to poor performance of a dataset. Classification algorithms

which use covariance matrix may not be able to accurately model the HDLSS data (Ahn *et al.*, 2007). Statistical methods such as ANOVA which is commonly used in medical and biomedical fields may not be able to give accurate statistical estimations if the sphericity assumption is violated by the HDLSS data. Methods to test if the sphericity assumption holds for a given data can degenerate for HDLSS conditions. As such, in the literature, some study has been conducted to create better tests of sphericity to cater for HDLSS data (John, 1971). Through this, we see that blindly using algorithms on HDLSS data without understanding its nature can lead to unreliable results.

Taking the stride from studies held for sphericity tests, classification algorithms have to be studied and extended to suit HDLSS data. There are two common classifier models, i.e. discriminant models and generative models. Generative models require less samples than discriminative models as it achieves its asymptotic error at a faster rate, but as the sample size increases, the discriminative model should be able to perform better (Ng and Jordan, 2002). SVMs deal relatively well with high dimensional data but is not able to perform well in the HDLSS asymptotic. This is because of the data piling phenomenon which causes the samples to pile reducing the discriminant ability of SVMs (Marron *et al.*, 2007). Neural networks and manifold learning on the other hand require a large amount of samples in order to produce accurate results. HDLSS data generally lack the samples required to support these algorithms. Neural networks are also not able to interpret data as transparently as other methods. This can become a problem in medical fields requires explainability of features which contribute to the output. Naive Bayes is better able to handle the curse of dimensionality compared to other algorithms as it decouples its features. Hence, there is a need to build classifiers or inferencing algorithms catering for HDLSS data.

A common tactic to deal with high dimensionality is to conduct dimensionality reduction to preprocess the data. Common dimensionality reduction technique such as PCA are not able to perform well in HDLSS settings. This is due to inconsistencies among the directions of the principal components (Shen *et al.*, 2013). While a few alternatives to the PCA algorithm were proposed in literature, they are still not able to fully solve the problem. Autoencoders are able to discover compact representation of datasets. However, they require large amounts of data to accurately discover the representation. In literature, a drop in performance of autoencoders was observed when the sample size decreases (Xue *et al.*, 2017). Information theory based dimensionality reduction methods are able to capture higher order statistics. Brown *et al.* (2012) reported that some feature selection method such as MRMR is able to perform well in HDLSS data while some others are not able. Using dimensionality reduction methods

that are not able to perform in HDLSS setting to preprocess the data before classification is done is counter-intuitive. This is because it would cause the performance to degrade. Hence, there is a need to study feature extraction and selection methods which are able to perform in HDLSS setting. Feature selection and extraction methods should also be built for HDLSS data.

In order to address the research gap we use CorEx to propose a feature extraction and selection methods under the discriminant classification model and an inference algorithm under the generative classification model. We selected CorEx as its probability distribution is calculated in such a manner as to take into consideration the HDLSS asymptotic. The estimation of probability density distribution via Parzen Window, kernel density functions, or the use of covariance matrix, causes inconsistencies within the model. Density estimation methods generally require more samples than features to produce meaningful estimations (Silverman, 1986). Further, CorEx does not have the black box effect which is undesirable for medical data. However, CorEx is an unsupervised method. Hence, we extend CorEx to include the class label by allowing a mixture of distributions to be modelled. We take advantage of their density estimation and explore the possibilities and problems which arise when extending the model for supervised learning.

Chapter 3

Research Methodology

3.1 Overview

Fig.(3.1) shows the overview of the workflow of this PhD. In the figure, we show how the different objectives of this PhD are met. CorEx was extended to include the class label. This we call as CorEx-C. The use of CorEx-C enabled us to investigate the possibility of using it under both the discriminant and generative classification umbrella. We also compared the effectiveness of the features extracted through CorEx-C against features extracted through CorEx under the discriminant classification banner. We then benchmarked methods proposed for both discriminant classification and generative classification against existing classifiers. The benchmarking process was done using simulated data and real world data.

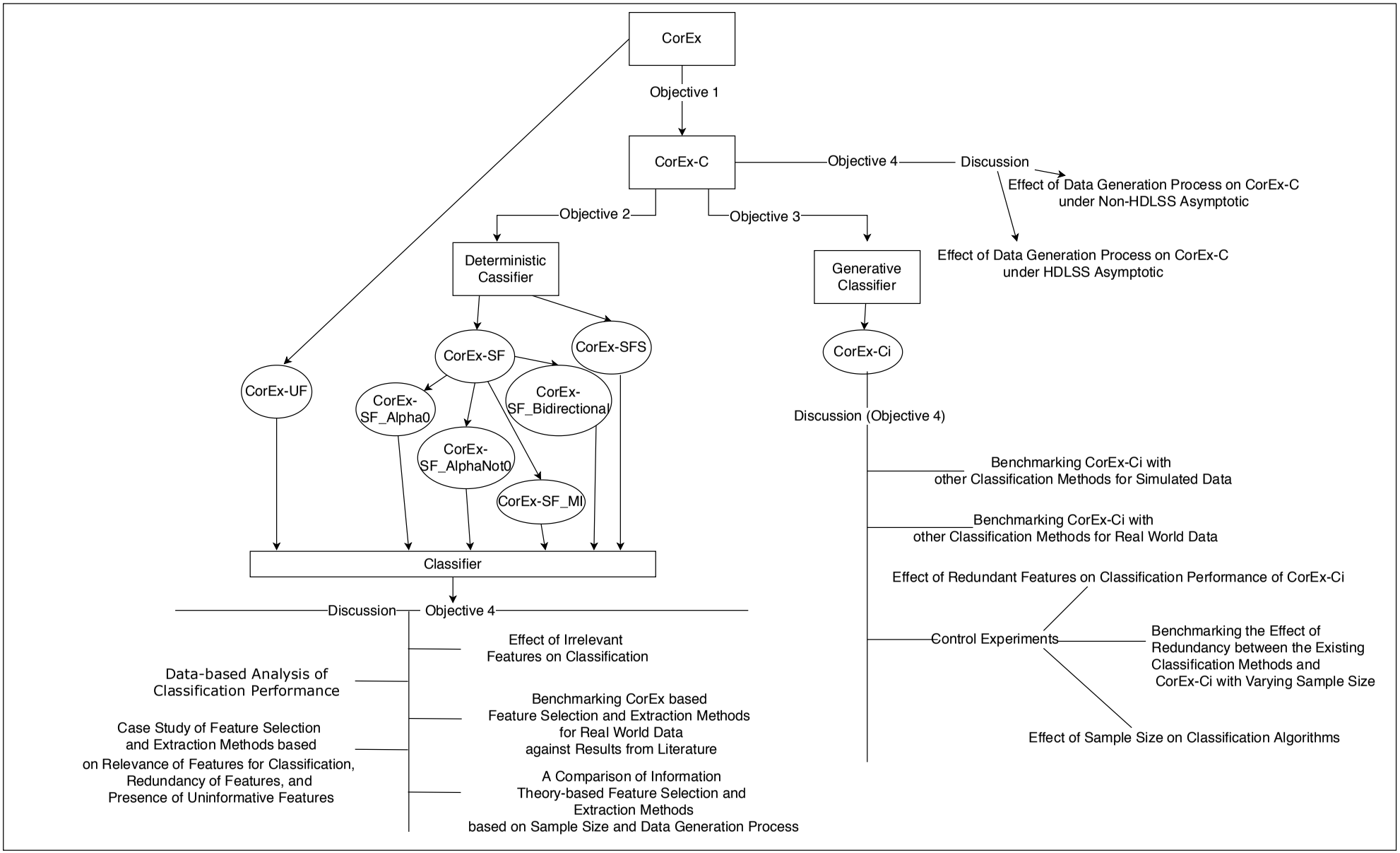


Figure 3.1: The figure shows the overview of the process involved during the course of the PhD research and how each of the objectives stated in Chapter 2 are met.

3.2 CorEx-C

We propose CorEx-C which is a graphical model based on CorEx which takes into account the class label. The purpose of this section is to extend an unsupervised graphical model to allow for supervised learning. The class label is incorporated as a part of the iterative optimization process of Eq.(2.61). Previously, both Pepke and Ver Steeg (2017) and Ver Steeg and Galstyan (2014b) only proposed CorEx models which either consider the input layer to be continuous or discrete. CorEx uses Lagrangian optimization to find the solution to $p(y_j|x^l)$. The equation is given by:

$$p(y_j|x^l) = \frac{1}{Z(x)} p(y) \prod_{i=1}^{d} \left(\frac{p(y_j|x_i)}{p(y)}\right)^{\alpha_{i,j}} \tag{3.1}$$

where the α is estimated by:

$$\alpha_{i,j} = \frac{I(x_i, y_j|y_1, \ldots, y_{j-1})}{I(x_i, y_j)} \tag{3.2}$$

Using Bayes' theorem, $p(y_j|x_i)$ can be further decomposed into,

$$p(y_j|x_i) = \frac{p(x_i|y_j)p(y_j)}{p(x_i)} \tag{3.3}$$

$p(x_i|y_j)$ is parameterized in Pepke and Ver Steeg (2017) as a normal distribution in the first hierarchical layer and as a discrete distribution in the proceeding layers.

The estimation of the parameters of the normal distribution, $\mu_{i,j}$ and $\sigma_{i,j}$ for each value k for which y_j can take, through an iterative scheme is as follows:

$$\mu_{i,j} = \frac{\frac{1}{N}\sum_l p(y_j|x^l)x_i^l}{p(y_j)} \tag{3.4}$$

$$\sigma_{i,j} = \frac{\sqrt{\frac{1}{N}\sum_l p(y_j|x^l)(x_i - \mu_{i,j})^2}}{p(y_j)} \tag{3.5}$$

where $l \in \{1, \ldots, N\}$.

$p(y_j)$ is calculated iteratively as Pepke and Ver Steeg (2017):

$$p(y_j) = \frac{1}{N}\sum_l p(y_j|x^l) \tag{3.6}$$

In the discrete case, the distribution is estimated directly from the contingency table of counts (Pepke and Ver Steeg, 2017) as follows:

$$p(x_i|y_j) = \frac{1}{N}\sum_l^N p(y_j|x^l)\delta_{x_i^l, x_i} \tag{3.7}$$

where x_i^l is the range of values that x_i can take that has been observed and δ denotes Kronecker's delta.

We take advantage of the modular structure by which the distribution of the data $p(x_i|y_j)$ is formulated in Eq.(3.1) to apply a mixture of distributions.

The parametrization of $p(x_i|y_j)$ was modified to reflect the mixture of distributions as follows:

$$p(x_i|y_j) = \begin{cases} \text{Normal Distribution}, & \text{if } i \neq c \\ \text{Discrete Distribution}, & \text{otherwise} \end{cases} \tag{3.8}$$

where c indicates class label.

This parametrization is then used to solve for Eq.(3.3).

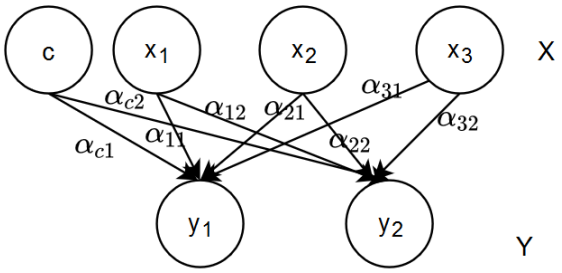


Figure 3.2: The figure shows the CorEx-C structure. It is similar to the original CorEx structure. The difference lies in the parameterization of the input layer.

The main structure of CorEx-C remains the same as CorEx, however, in the case of CorEx-C, class label is included as an additional input node as shown in Fig.(3.2). The distribution in the level of the input features will be calculated through Eq.(3.8). If the parameterization of $p(x_i|y_j)$ is not modified, then in order to include the class label, the data will need to be discretized. This will cause some information loss to occur. In the case of HDLSS data where samples are scarce, retaining as much of the integrity of the data is desired to minimise error. If the class label is modelled as a continuous variable, it would be a misrepresentation of the structure of the class label.

The pseudocode for the implementation is given as follows:

Procedure 1 Pseudocode implementing the change to reflect the mixture of distributions in Correlation Explanation. It is similar to the pseudocode in (Ver Steeg and Galstyan, 2014a). The difference between CorEx and CorEx-C lies in the initial matrix representation as well as the computation of marginals to allow for mixture of distribution to be modelled.

Input: A matrix of size $n \times d$ representing n samples, $d-1$ continuous random variables, and 1 discrete random variable (class)

Set: Set m, the number of latent variables, Y, and k so that $|Y_j| = k$

Output: Parameters $\alpha_{i,j}$, $p(y_j|x_i)$, $p(y_j)$, $p(y|\vec{x}^l)$, for $i \in \mathbb{N}_d$, $j \in \mathbb{N}_m$, $l \in \mathbb{N}_n$, $y \in \mathbb{N}_k$, $x_i \in X$

Randomly initialize $\alpha_{i,j}, p(y_j|x^l)$;

repeat

Estimate marginals, $p(y_j)$, $p(y_j|x_i)$ using Eq.(3.6) and Eq.(3.3). The marginal for Eq.(3.3) is calculated based on Eq.(3.8).

Calculate $I(x_i, y_j)$ from marginals

Update α through Eq.(3.2)

Calculate $p(y|x^l), l = 1 \ldots n$ using Eq.(3.1)

until Convergence

3.3 Discriminant Classification Model

We investigate feature selection and extraction under the discriminant classification model framework for HDLSS settings. We test some existing information theory based feature selection and extraction methods in HDLSS settings. We also propose to use CorEx as a feature extractor in two forms, i.e., an unsupervised feature extractor which will be abbreviated to CorEx-UF and a supervised version coined as CorEx-SF. It is also tested as a supervised feature selector which is named as CorEx-Supervised Feature Selection (CorEx-SFS). In the case of CorEx-SFS and CorEx-SF, we will be using the CorEx-C model.

CorEx-C based feature extraction and selection methods are benchmarked against the accuracy produced by MMI by Torkkola (2003), KECA, and Infomax ICA. We look at Maximum Relevance Minimum Redundancy (MRMR) (Peng *et al.*, 2005) and Conditional Infomax Feature Extraction(CIFE) (Lin and Tang, 2006) as two baseline methods for comparison among supervised feature extraction methods with HDLSS data, with the former being ranked as one of the top performers and the latter among the worst (Brown *et al.*, 2012). For a more detailed description of the comparisons of

the performance of information theory based feature selection methods for small sample size datasets, we direct the readers to Brown *et al.* (2012).

3.3.1 CorEx-UF

CorEx-UF extracts the values of the latent variables obtained from running CorEx on the input data as the transformed feature space for classification. The use of the most likely value, k, of each $Y_j \in Y$ for each sample is based on the assumption that CorEx learns nearly deterministic functions of X, hence reflecting its true distribution (Ver Steeg and Galstyan, 2014a). The values of the latent factors for each sample is used as an encoding of the input data for classification.

The selection of the number of nodes to be set in a layer is initially chosen at random if prior knowledge is not available. We hypothesize that the representative chosen should be small enough to retain as much generating information regarding the dataset and not overly small that too much information is lost when feature extraction is done. CorEx-UF was built as a comparator for the supervised versions of CorEx-based feature extraction methods.

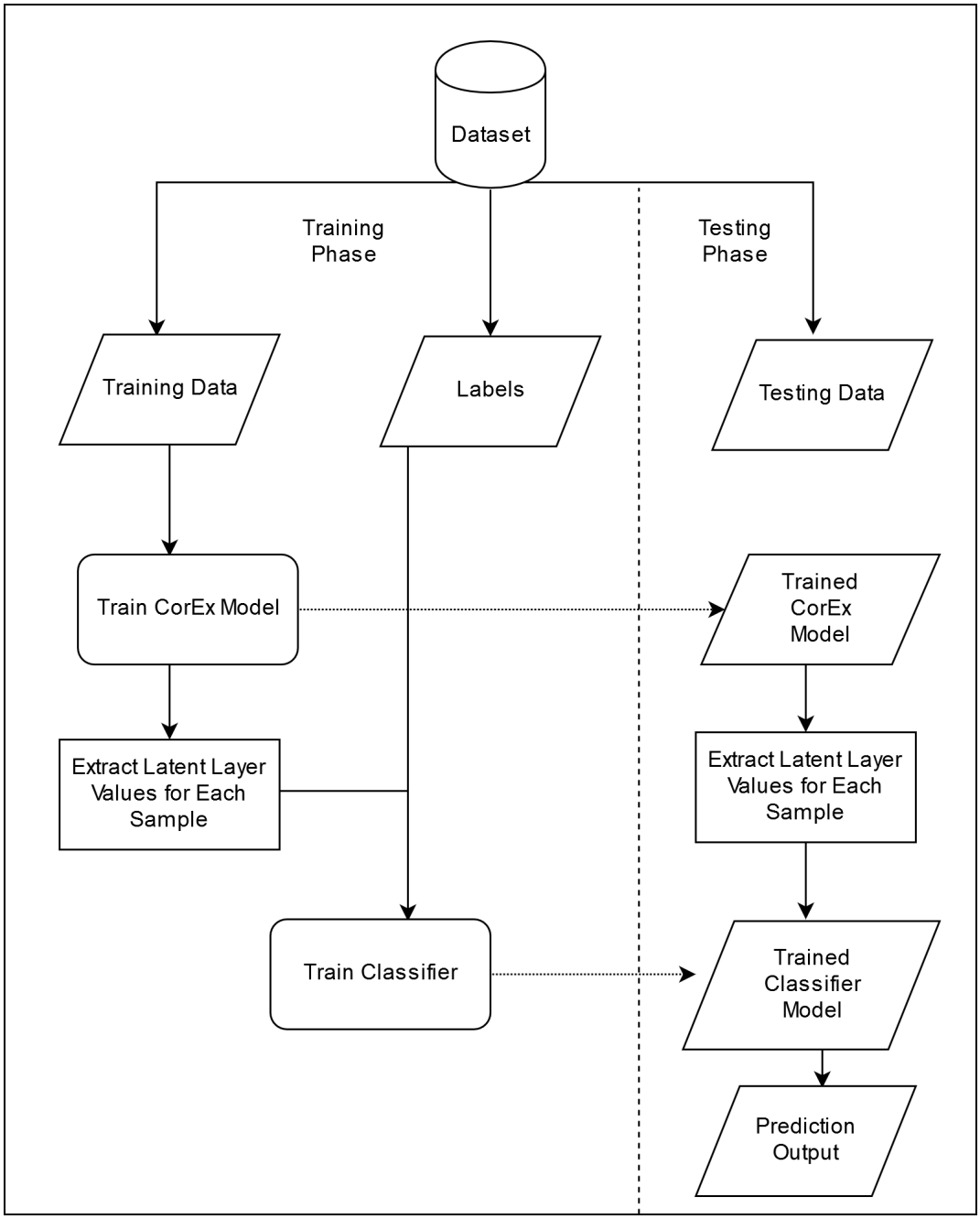


Figure 3.3: The figure shows the process for CorEx-UF. CorEx-UF uses the CorEx model. The data is split into training and testing set respectively. The training data is used to train the CorEx model. The testing data is then passed through the trained CorEx model. The output values produced for each hidden node is used as the input features for classification.

3.3.2 CorEx-SF

CorEx-SF adds the class labels as an additional input node during the training stage of CorEx, i.e. CorEx-C. Instead of learning a generic representation of the data

as with CorEx-UF, it aims to facilitate the learning of latent variables that reduce its redundancy whilst maintaining relevance to the class labels. During the testing stage of CorEx we set the class labels as missing. The values of the latent variables are used for classification.

We threshold the latent features learnt in order to understand the features which are important for classification. We tested the thresholding in 4 ways:

1. CorEx-SF_AlphaNot0: The latent factors which satisfy the condition $\alpha_{c,j} > 0$ are selected, where $\alpha_{c,j}$ is the connection between the input class feature and the latent factor j. They are then sorted in descending order of $\alpha_{c,j}$. This method was formulated so that only the latent features that are related to the class label will be extracted.

Procedure 2 Pseudocode of the implementation for CorEx-SF_AlphaNot0

for $j = 1 \dots m$ **do**
  if $\alpha_{c,j} > 0$ **then**
    $S \leftarrow j$
$S_s \leftarrow$ sort(S) in descending order of $\alpha_{c,j}$
$t \leftarrow$ threshold S_s
Classify(t)

2. CorEx-SF_Alpha0: The latent factors which satisfy the condition $\alpha_{c,j} \geq 0$ are selected. They are sorted in descending order. Values of $\alpha_{c,j} = 0$ are sorted in descending order according to the total correlation captured by the latent node. The higher the total correlation, the more the information captured by the node. This method was built to investigate whether additional features that are not considered relevant to the class label can contribute to discriminating the samples for classification.

Procedure 3 Pseudocode of the implementation for CorEx-SF_Alpha0

for $j = 1 \ldots m$ **do**
  if $\alpha_{c,j} > 0$ **then**
    $S \leftarrow j$
  else
    $H \leftarrow j$
$S_s \leftarrow$ sort(S) in descending order of $\alpha_{c,j}$
$H_s \leftarrow$ sort(H) in descending order of Total Correlation.
$S_s \leftarrow S_s ++ H_s$
$t \leftarrow$ threshold S_s
Classify(t)

3. CorEx-SF_MI: The latent factors which satisfies the condition $\alpha_{c,j} \times MI(x_c, y_j) \geq 0$ are selected, where x_c and y_j correspond to the class node and latent node. They are sorted in descending order. Values of $\alpha_{c,j} \times MI(x_c, y_j) = 0$ was sorted in descending order according to the total correlation. This method is done as a weighted method to study whether exaggerating the contribution of features that are considered relevant and diluting the contribution of features that are weak will have any impact for classification.

Procedure 4 Pseudocode of the implementation for CorEx-SF_MI

for $j = 1 \ldots m$ **do**
  if $\alpha_{c,j} \times MI(x_c, y_j) > 0$ **then**
    $S \leftarrow j$
  else
    $H \leftarrow j$
$S_s \leftarrow$ sort(S) in descending order of $\alpha_{c,j}$
$H_s \leftarrow$ sort(H) in descending order of Total Correlation.
$S_s \leftarrow S_s ++ H_s$
$t \leftarrow$ threshold S_s
Classify(t)

4. CorEx-SF_Bidirectional: The latent factor, h, with the strongest $\alpha_{c,j}$ value are selected. The input features which are connected to the latent node, h are thresholded based on $\alpha_{i,h} > 0$ and $i \neq c$. The selected set of input features,i_G, are then traced backwards to the latent factor which has the largest magnitude of $\alpha_{i_g,j}$

where $i_g \in i_G$. This method was built to understand whether selecting latent factors that are strongest connected to the input features that are relevant will allow for a better representation of the data.

Procedure 5 Pseudocode of the implementation for CorEx-SF_Bidirectional

$h \leftarrow \text{argmax}_j \alpha_{c,j}$
$i_G = []$
for $i = 1 \ldots d-1$ **do**
  if $\alpha_{i,h} > 0$ **then**
    $i_G \leftarrow i$
for $i_g \in i_G$ **do**
  for $j = 1 \ldots m$ **do**
    $t \leftarrow \text{argmax}_j \alpha_{i_g,j}$
Classify(t)

The top 10%, 30%, 50%, and 80% of the latent factors that were sorted are selected. 100% indicates all latent factors are selected.

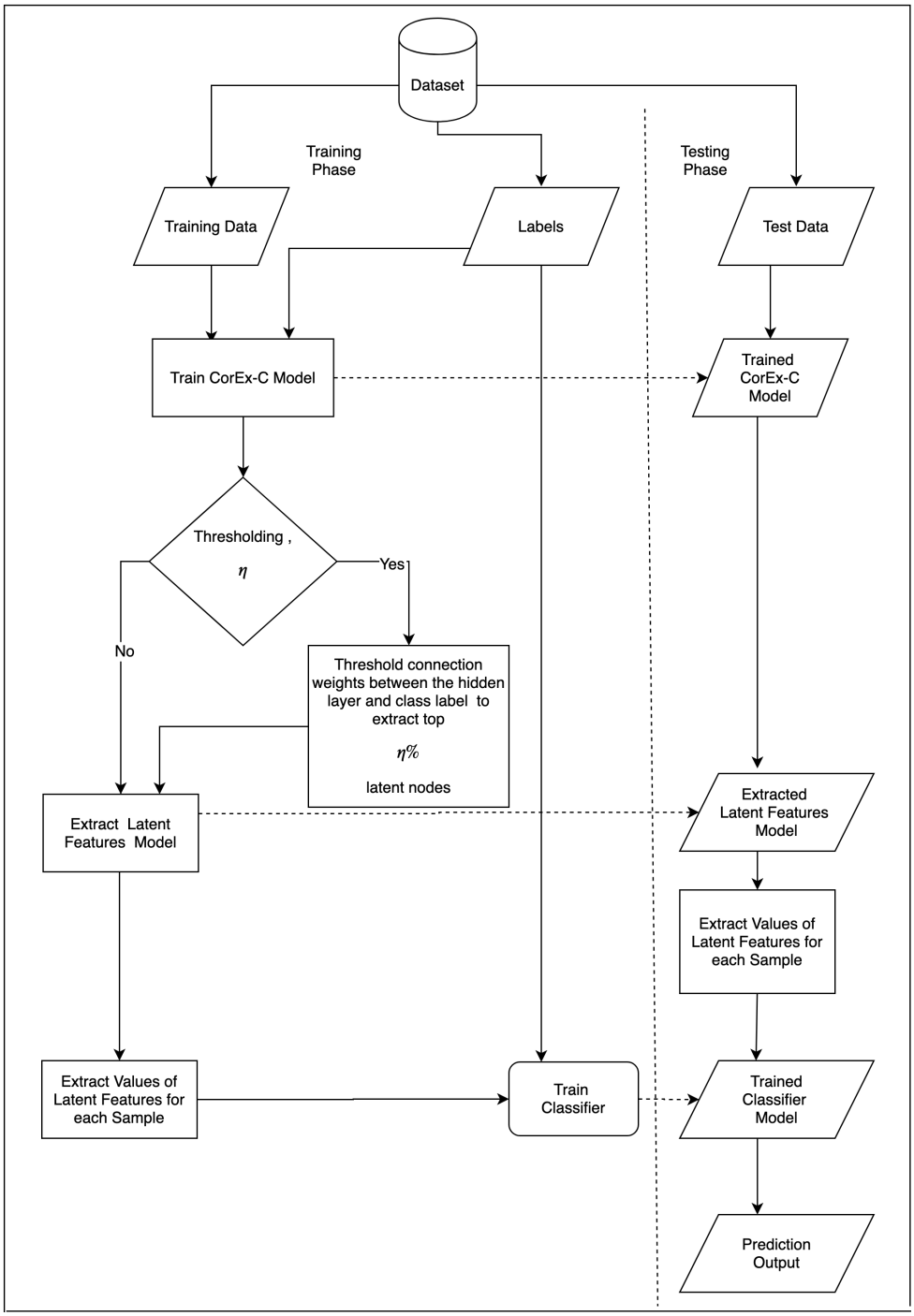


Figure 3.4: The figure shows the process for CorEx-SF. The data is split into training and testing set respectively. The training data trains the CorEx-C model. The testing data is then passed through the trained CorEx-C model. A further thresholding of the values to be used for classification can be done through the choice of CorEx-SF_MI, CorEx-SF_Alpha0, CorEx-SF_AlphaNot0, and CorEx-SF_Bidirectional. The output values produced for each hidden node is used as the input features for classification.

3.3.3 CorEx-SFS

CorEx-SFS selects features from the original input space by considering the nodes which are related to the class labels. We used the CorEx-C model to train the data. The difference between CorEx-SF and CorEx-SFS is that the features are selected from the input space for CorEx-SFS while CorEx-SF uses the transformed nodes as its feature space. We hypothesize that CorEx-SF should perform better than CorEx-SFS as selecting features could cause the loss of information that comes from features considered less important to the classification task.

CorEx-SFS is thresholded based upon $s = \text{argmax}_j \alpha_{c,j}$. The input features are then selected based on $\alpha_{i,s} > 0$ and $i \neq c$. The input features are used for classification.

Procedure 6 Pseudocode for CorEx-C based supervised feature selection

$s \leftarrow \text{argmax}_j \alpha_{c,j}$
for $i = 1 \ldots d-1$ **do**
  if $\alpha_{i,s} > 0$ **then**
    $t \leftarrow i$
Classify(t)

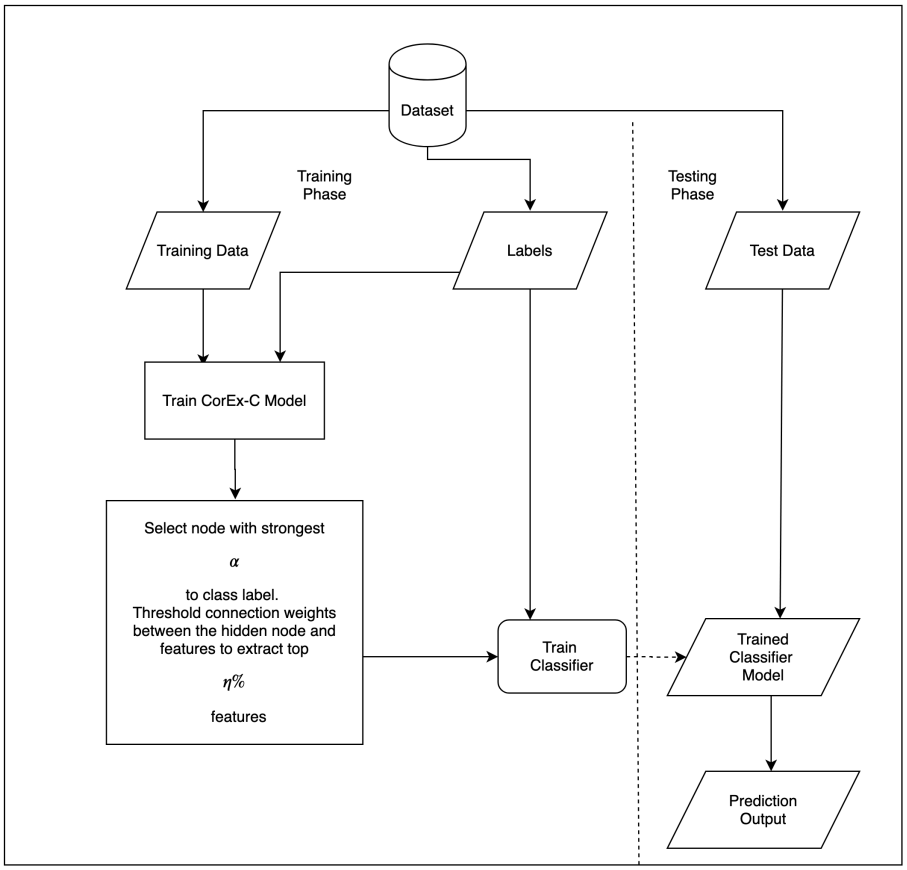


Figure 3.5: The figure shows the process for CorEx-SFS. The training data trains the CorEx-C model. The testing data is then passed through the trained CorEx-C model. Input features with the strongest α-value to the hidden node which is connected to the class label were selected as the input for the classification task.

3.4 Generative Classification Model

CorEx-Ci uses the CorEx-C model along with Bayesian inferencing coupled with mean field approximation. The purpose of this method is to investigate the possibility of extending the CorEx-C method under the generative classification model umbrella. We test whether it can perform in HDLSS settings and how well it performs compared to common classification methods such as SVM, Naive Bayes, and KNN in HDLSS settings.

3.4.1 Posterior Estimation

CorEx-C takes as input a set of data, which is denoted as X such that $\{(x_1, c_1), (x_2, c_2), .., (x_n, c_n)\} \in X$, where $\{c\} \in C$ corresponds to the class label of instance

x_i and n is the total sample available in the data.

$$X \xrightarrow{\text{CorEx}} Y, \alpha \tag{3.9}$$

Equation 3.9 shows that data, X when run through CorEx will output a set of weights, α to connect the input feature with the hidden nodes and Y discrete values which each node can take.

When a new instance, $\vec{x^*}$ is supplied, the posterior is calculated through an objective function which is obtained using Bayes' theorem to assign a class label. Bayes' theorem is mathematicaly defined as:

$$P(X|Y) = \frac{P(Y|X)P(X)}{P(Y)} \tag{3.10}$$

Hence, we calculate the posterior as follows:

$$\text{argmax}_c \; p(c|\vec{x^*}) = \sum_{\vec{y}} p(c|\vec{y}) \prod_{j=1}^{M} p(y_j|\vec{x^*}) \tag{3.11}$$

where c is a possible class label and x^* is the instance of the new data. The objective function looks for the most probable class label for the new data.

The CorEx structure follows the tail-to-tail structure as shown in Fig.(2.4). This implies that when the latent structure is observed, the input variables are conditionally independent of each other. The indicator function α prunes the connections between the each feature and the latent nodes. Through the use of α, the relationship between each of the latent variables and features are weighted. Hence, through continuous application of Bayes' theorem and the use of the indicator function to weight the relationship between the class label and the latent variables, we formalised the components of Eq.(3.11) to give :-

$$p(c|\vec{y}) = \prod_{j=1}^{M} p(c|y_j)^{\alpha_{c,j}} \tag{3.12}$$

Eq. (3.12) can be further optimized by:-

$$p(c|y_j) = \frac{p(y_j|c)p(c)}{p(y_j)} \tag{3.13}$$

In the case of $p(y_j|\vec{x^*})$, it is the same as $p(y_j|x^l)$, i.e. Eq.(2.61) with the only difference that during inferencing, the class label will be considered missing. CorEx ignores features where the class labels are missing. Putting everything together, we would obtain

$$\text{argmax}_c p(c|\vec{x^*}) = \sum_{\vec{y}} \prod_{j=1}^{M} \left(\frac{p(y_j|c)p(c)}{p(y_j)} \right)^{\alpha_{c,j}} \prod_{j=1}^{M} p(y_j|\vec{x^*}) \tag{3.14}$$

The complexity of this inferencing method is $\mathcal{O}(n)$ because we use all samples to compute the estimate. Due to the fact that the sample size is small, the time complexity is relatively low but will grow linearly as the number of samples increase. The number of latent nodes selected is done by running random search over a set of values. The experiments were seeded in order to produce consistent results.

Procedure 7 Pseudocode implementing the inference process for CorEx-Ci.

Input: A matrix of size $n \times d$ representing n samples, $d-1$ continuous random variables, and 1 discrete random variable (class label)

Set: Set m, the number of latent variables, Y, and k so that $|Y_j| = k$

Output: Parameters $\alpha_{i,j}$, $p(y_j|x_i)$, $p(y_j)$, $p(y|\vec{x}^l)$, for $i \in \mathbb{N}_d$, $j \in \mathbb{N}_m$, $l \in \mathbb{N}_n$, $y \in \mathbb{N}_k$, $x_i \in X$

Randomly initialize $\alpha_{i,j}, p(y_j|x^l)$;

repeat

Estimate marginals, $p(y_j)$, $p(y_j|x_i)$ using Eq.(3.6) and Eq.(3.3)

Calculate $I(x_i, y_j)$ from marginals

Update α through Eq.(3.2)

Calculate $p(y|x^l), l = 1 \ldots n$ using Eq.(3.1)

until Convergence

Inference Step:

for $n_t = 1 \ldots n_{\text{test}}$ **do** where n_{test} is the number of testing samples

Estimate $p(c)$ from training samples

Calculate $p(y_j|\vec{x^*_{n_t}})$ using Eq.(3.1).

Calculate $p(y_j)$ using Eq.(3.6)

Calculate $p(y_j|x^*_{i,n_t})$ using Eq.(3.3)

$c_{n_t} \leftarrow \text{argmax}_c p(c|\vec{x^*_{n_t}})$

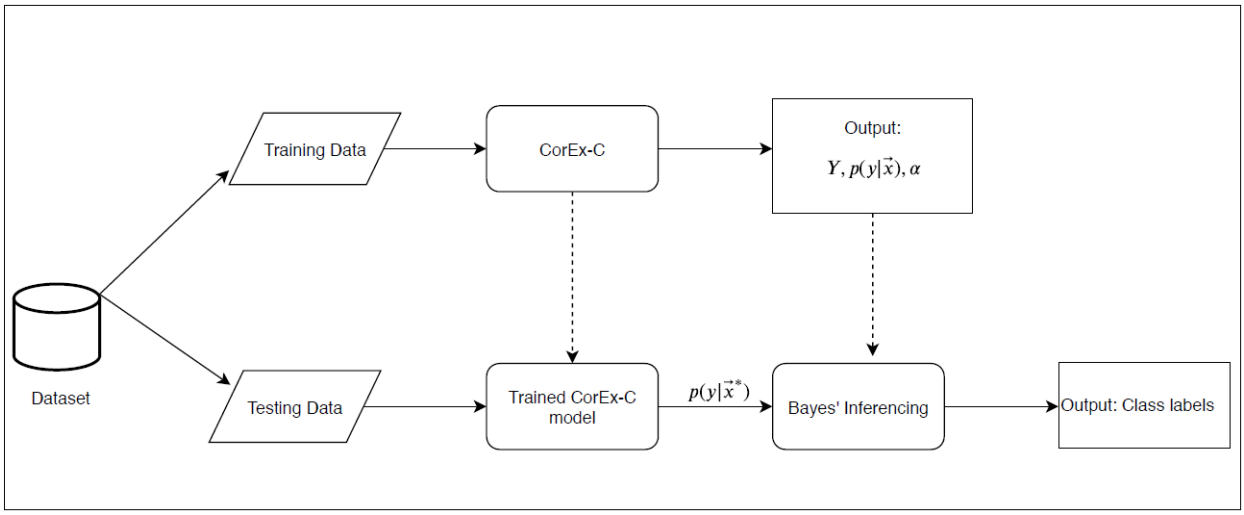


Figure 3.6: The figure shows the process for CorEx-Ci.

3.4.2 Control Experiments for CorEx-Ci

We ran two sets of control experiments in order to understand the nature of CorEx-Ci better. The simulated data was used is as described in section 3.5. We ran the experiment 10 times and plotted the average of the runs. We also compared the performance against the performance of other classifiers such as SVM with Linear Kernel, SVM with RBF Kernel, Naive Bayes, and K-Nearest Neighbour. The two sets of experiment run were:

1. We varied the input dimensions of the simulated data while keeping the sample size fixed, which follows the HDLSS asymptotic.

   We tested three variations of differing the dimension size. They are:-

   (a) Keeping the variables that are directly related to the class label the same while increasing redundancy among the uninformative variables.

   (b) Keeping the uninformative variables the same while increasing the redundancy among the informative variables.

   (c) Increasing the redundancy among both the informative and uninformative variables.

2. Vary the sample size, but keep the dimensions the same. This follows the classical asymptotic under which most classification methods would perform well.

Plots were made to view the trend each of the different classifiers portray.

3.5 Simulated Datasets

Two simulated datasets were generated. The datasets were simulated to further understand how the proposed methods work and to what extent in a controlled manner. It is also used to test under which conditions CorEx-based methods will work. The data simulation process used follows that of ancestral sampling. Ancestral sampling samples the variables in order of parent to child (Bishop, 2006). The child variable is sampled conditioned on its parents in the graph. The data was generated using the $Y \rightarrow X \rightarrow C$ structure and $Y \rightarrow X$, $Y \rightarrow C$ structure. Y is the latent structure, X is the input feature set and C is the class label. For the $Y \rightarrow X \rightarrow C$ generation, it indicates that $C \perp\!\!\!\perp Y \mid X$ while $Y \rightarrow X$, $Y \rightarrow C$ indicates that $C \perp\!\!\!\perp X \mid Y$. In both the generation process, Y is independent of each other. CorEx makes the assumption that the data follows the form of $Y \rightarrow X$, $Y \rightarrow C$. On the other hand, many classifiers try to model the relationship such that X has an explicit relationship with C. Hence, through these generated data, we investigate to what extent the CorEx-C model captures the probability in terms of $P(C|X)$.

The data generation process was done as follows: 10 nodes of latent variables were first generated using Bernoulli distribution with different probabilities. This was done in order to simulate the latent variables modeled by CorEx. We have used Bernoulli distribution as we set the $y = \{0, 1\}$ as done in CorEx. For each $y \in \{y_1, \ldots, y_{10}\}$ we generated 50000 samples. In order to sample X, μ and σ of the normal distribution was parameterized as a function of Y to model the conditional probability $P(X|Y)$. For each latent variable y, 10 x were generated with 50000 samples each. Therefore, the total number of features in the dataset is 100, not including the class label. The class labels were generated in the following manner:

1. $Y \rightarrow X$, $Y \rightarrow C$ (YC): Only y_1 was used. An additive noise was added. The class label was a pulse variable of y_1

2. $Y \rightarrow X \rightarrow C$ (YXC): The class labels are generated only from the first 10 xs (i.e. only x simulated from y_1). This was done to keep it consistent with the $Y \rightarrow X, Y \rightarrow C$ generation process. The class label is a non-linear combination of the values between $x_1 \cdots x_{10}$. The values were then passed through the inverse-logit function and thresholded using the criteria $< 0.5 \rightarrow c = 0$, while $>= 0.5 \rightarrow c = 1$.

The simulated datasets were split into 60% training samples and 40% testing samples. The number of samples from each class are balanced in the training set. The training sample size was also downsampled to 80 samples using random downsam-

pling in order to simulate the HDLSS condition. The training set for both the training sample size is kept the same. The testing set is an imbalanced set.

In the case of $Y \rightarrow X, Y \rightarrow C$, we expect that the data may be hard to classify. This is because the class labels are not directly linked to the input variables. Many of the classification and inferencing methods require some form of metadata between the input variables and the class variables. In this case, the relationship is implicit.

3.6 Classification Performance Metric

Accuracy and Matthew's Correlation Coefficient (MCC) was chosen to measure the performance of the classifier.

3.6.1 Accuracy

Classification accuracy is the ratio of the number of correct samples to the total number of predictions made. Mathematically, it is defined as:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total Samples}} \tag{3.15}$$

where where TP is true positive, TN is true negative.

Accuracy is able to give a general evaluation of how well the model has been trained. However, if the samples are imbalanced, the accuracy could be skewed. This is because it could easily predict the majority class well but perform poorly on the minority class.

3.6.2 Matthew's Correlation Coefficient

Matthew's Correlation Coefficient (MCC) is used as a metric to check the robustness of the generalisation capabilities of the classifiers. MCC takes into account the false negatives and false positives when measuring the performance of a binary classification task. This will give a better evaluation of the classifiers performance in the case of imbalance data. MCC can take a value between -1 to 1. MCC is defined as:-

$$\text{MCC} = \frac{\text{TP} * \text{TN} - \text{FP} * \text{FN}}{\sqrt{(\text{TP} + \text{FP}) * (\text{TP} + \text{FN}) * (\text{TN} + \text{FP}) * (\text{TN} + \text{FN})}} \tag{3.16}$$

where TP is true positive, TN is true negative, FP is false positive and FN is false negative. If the denominator computes to zero, then it is set to 1 so that the computation does not become undefined.

Chapter 4

Results & Discussions

4.1 CorEx-C

We plotted heatmaps of the connection between the class label and the input features. The intensity of the heatmaps are based on the α value between the input features and the latent variables. The purpose of this experiment is to test the effect of different data generation process on CorEx-C model. Further, we also explore its effect given a non-HDLSS scenario. It is seen that the higher the sample size, the more accurate the representation. It should also be noted that with CorEx-C under the Y->X, Y->C assumption, regardless of the sample size, the class label is able to bias the relevant features fairly accurately.

The horizontal label (x-axis) in the heatmap represents the input features. The first 10 x in the dataset, i.e. $x1 \dots x10$ are relevant to the class label. The remaining input features are irrelevant to the class label. The last x plotted in the heatmap, i.e. $x101$, is the class label. The vertical-axis (y-axis) displays the hidden nodes that the CorEx-C model was trained with. Through the nodes, we can visualise the relationship between variables. As an example, Fig.4.3 shows that the class label ($x101$) is strongly connected to node 8 (y-axis). Through this node, we can also visualise with the aid of the heatmap, input node $x1$ is strongly related to the class label. The latent nodes are arranged in descending order of total correlation between the features. This means that features connected to latent node 0 would have the highest total correlation among them. The colour key indicates the strength of the relationship between the input features and the latent variable, with 0-light grey- indicating no relationship and 1 -black- indicating very strong relationship.

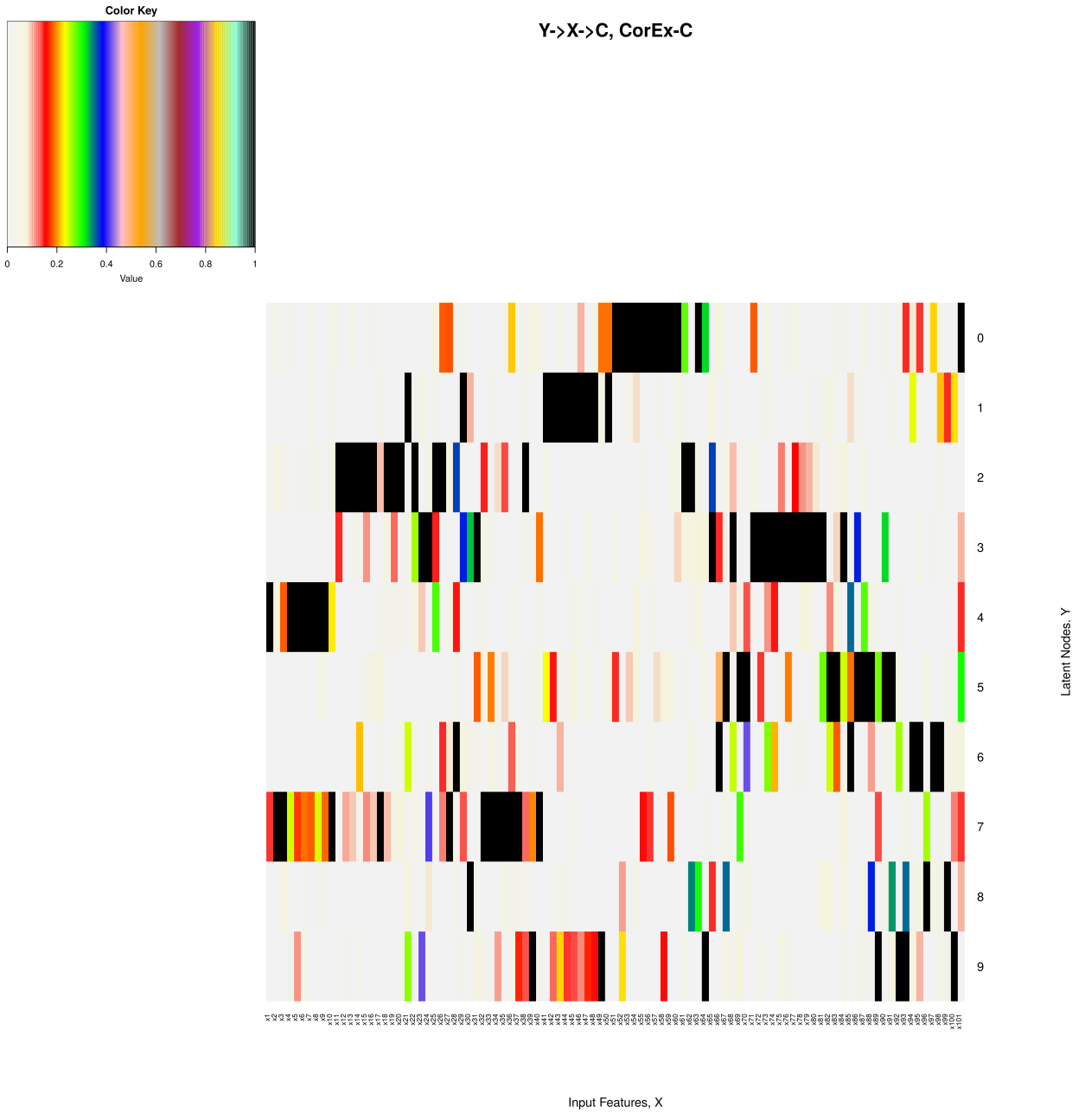


Figure 4.1: The figure shows the heatmap of the α values between the latent variables and the input features with 80 training samples $Y \rightarrow X \rightarrow C$.

4.1.1 Effect of Data Generation Process on CorEx-C Model under HDLSS Asymptotic

The original CorEx model made the assumption that the data generated should follow the $Y \rightarrow X, Y \rightarrow C$ process in order for relationships between features to be discovered accurately. The CorEx-C model does not change the assumptions made by the original CorEx model. Through the visualisation of the heatmaps, we investigate the behaviour of the CorEx-C model under two different data generation processes.

Data Generation Process: Y->X->C

Fig.(4.1) shows a completely incorrect grouping of features with the class label at hidden node 0. This shows the inability of the CorEx-C model to accurately capture relationships with high confidence. It should be noted that latent nodes 4 and 7 shows a very weak relationship between the class label and the relevant features. As such, in the case when the data generation process violates the CorEx assumption, the algorithm does not perform well. However, it should be noted that it is still able to capture some of the relevant nodes given that the generation process is violated.

Data Generation Process: Y->X, Y->C

Fig.(4.2) shows that the relationship between the class label and most of the relevant input features have been captured. The relationship that was captured shows a high confidence. This is because features were captured by nodes with higher total correlation. This shows that if the data generation assumption holds, CorEx-C is able to detect useful and meaningful relationships. Another difference between relationship captured by CorEx-C between the two data generation process is the quality of the features captured. This is to say that the features captured by the CorEx-C model as related to the class label, under this data generation process, is less noisy. Hence, the features extracted through this model will be more representative and lowers the chance of skewing the classifiers due to the presence of noise.

4.1.2 Effect of Data Generation Process on the CorEx-C Model under Non-HDLSS Asymptotic

While the CorEx-C model is able to capture relationships under HDLSS asymptotics, we further explore its ability when abundant samples are available.

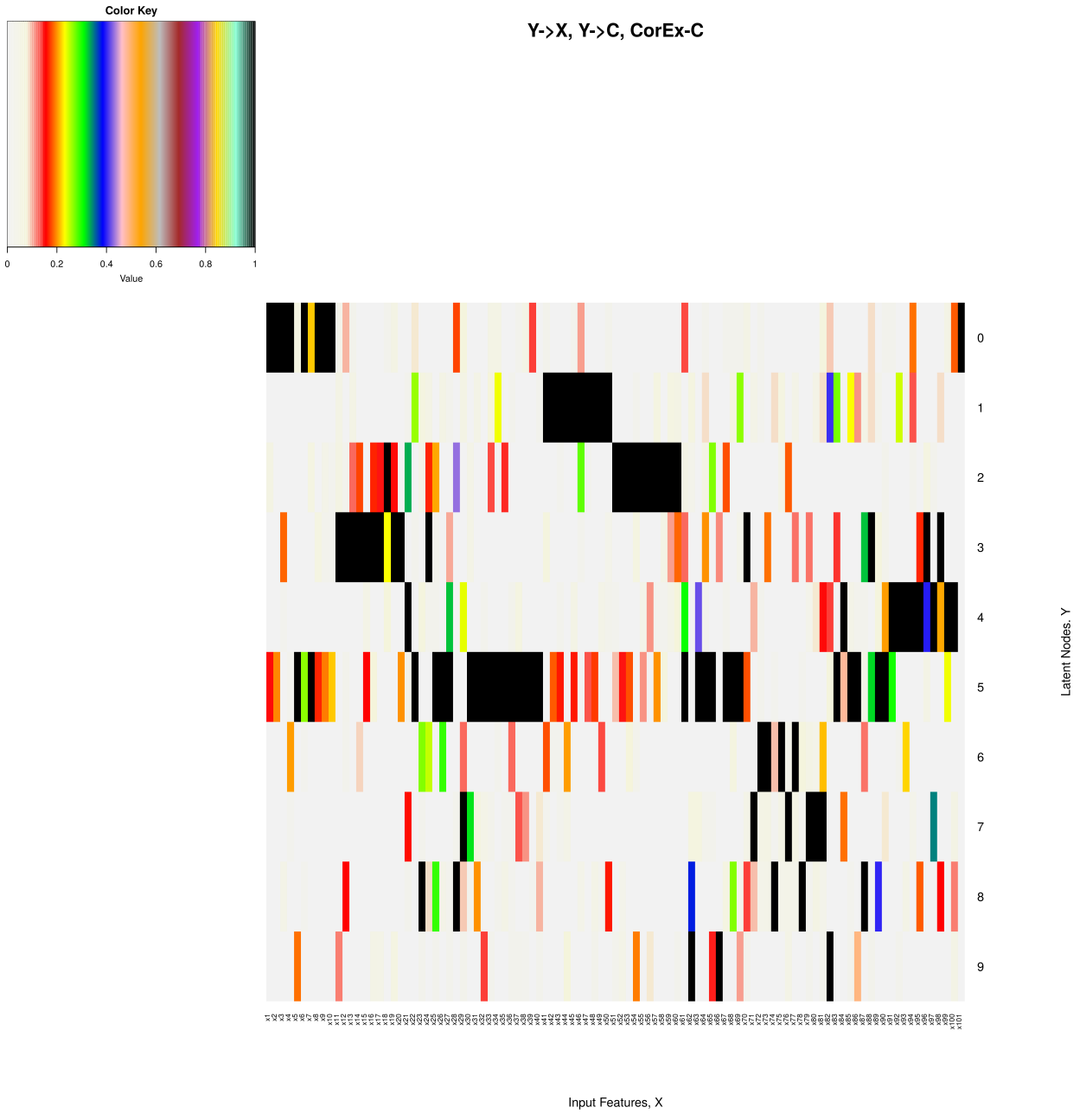


Figure 4.2: The figure shows the heatmap of the α values between the latent variables and the input features with 80 training samples $Y \to X$, $Y \to C$.

Data Generation Process: Y->X->C

Fig. (4.3) shows that the class label is able to detect some of the relevant features via latent node 8. The heatmap shows more features to be accurately grouped as related to the class label under the non-HDLSS scenario as compared to having a very small sample size. Less irrelevant features have also been captured as being related to the class label in contrast to when the sample size is small. A larger sample size allows for a more precise probability to be computed. Statistical methods generally do well when sample size is sufficiently large. However, as seen, there are still a number of other relevant features which are connected through node 7 which are considered irrelevant to the class label by CorEx-C. This also drives the point that large sample size does not circumvent all the obstacles posed when there are other factors involved such as the violation of the data generation assumption.

Data Generation Process: Y->X, Y->C

Fig.(4.4) shows that all the features that are relevant to the class label is connected to each other. It shows improvement compared to when the sample size is small. The higher the sample size, the more precise the model as all the hyperparameters can be trained sufficiently. However, the improvement is not as drastic as seen compared to the heatmaps in Fig.(4.1) and Fig.(4.3), when the data generation assumption was violated. It shows that CorEx-C model is relatively robust in capturing relevant features even in HDLSS settings provided the assumptions are not violated. The reason for this is due to the optimization done through the original CorEx model which CorEx-C leverages upon.

4.1.3 Discussion

Through the heatmaps, it can be seen that adding the class label can indeed map the discriminant features with the class labels in the case of $Y \rightarrow X$, $Y \rightarrow C$. While CorEx-C allows for changes in the modelling of the distribution of data, it does not change the core assumptions made by CorEx itself. CorEx strictly assumes that all the observations are induced by the latent variables. The latent variables are assumed to be independent of each other. In this case, as shown by Fig.(4.4)and Fig.(4.2), the assumption holds. However, in the case where the class labels are induced by other input features, CorEx-C is able to capture the relationship only with abundant data as shown from Fig.(4.3) and Fig.(4.1). Fig.4.1 is strongly connected to irrelevant features but is able to detect weak relationships with features that are relevant to the class label.

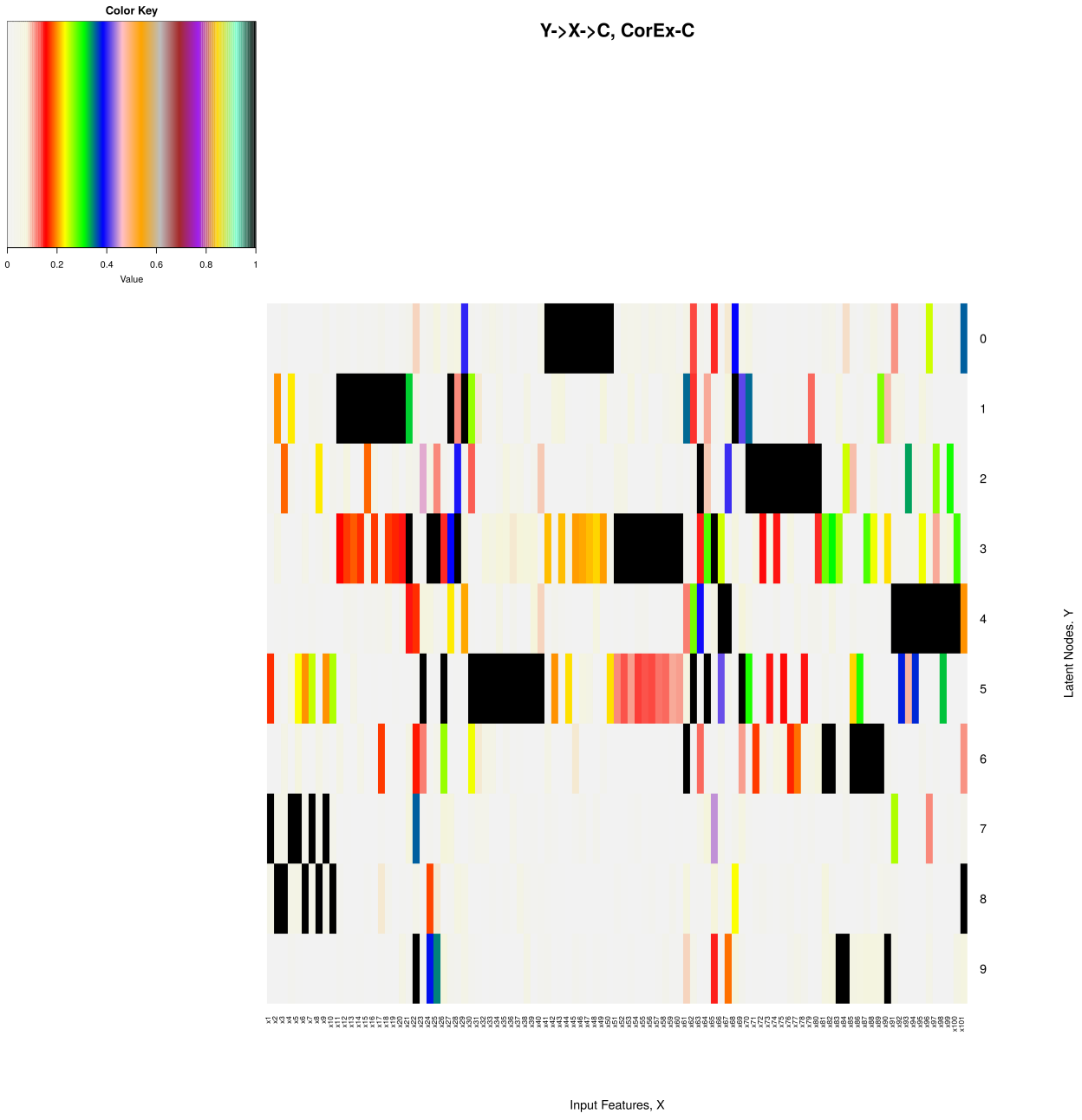


Figure 4.3: The figure shows the heatmap of the α values between the latent variables and the input features with 30000 training samples $Y \rightarrow X \rightarrow C$.

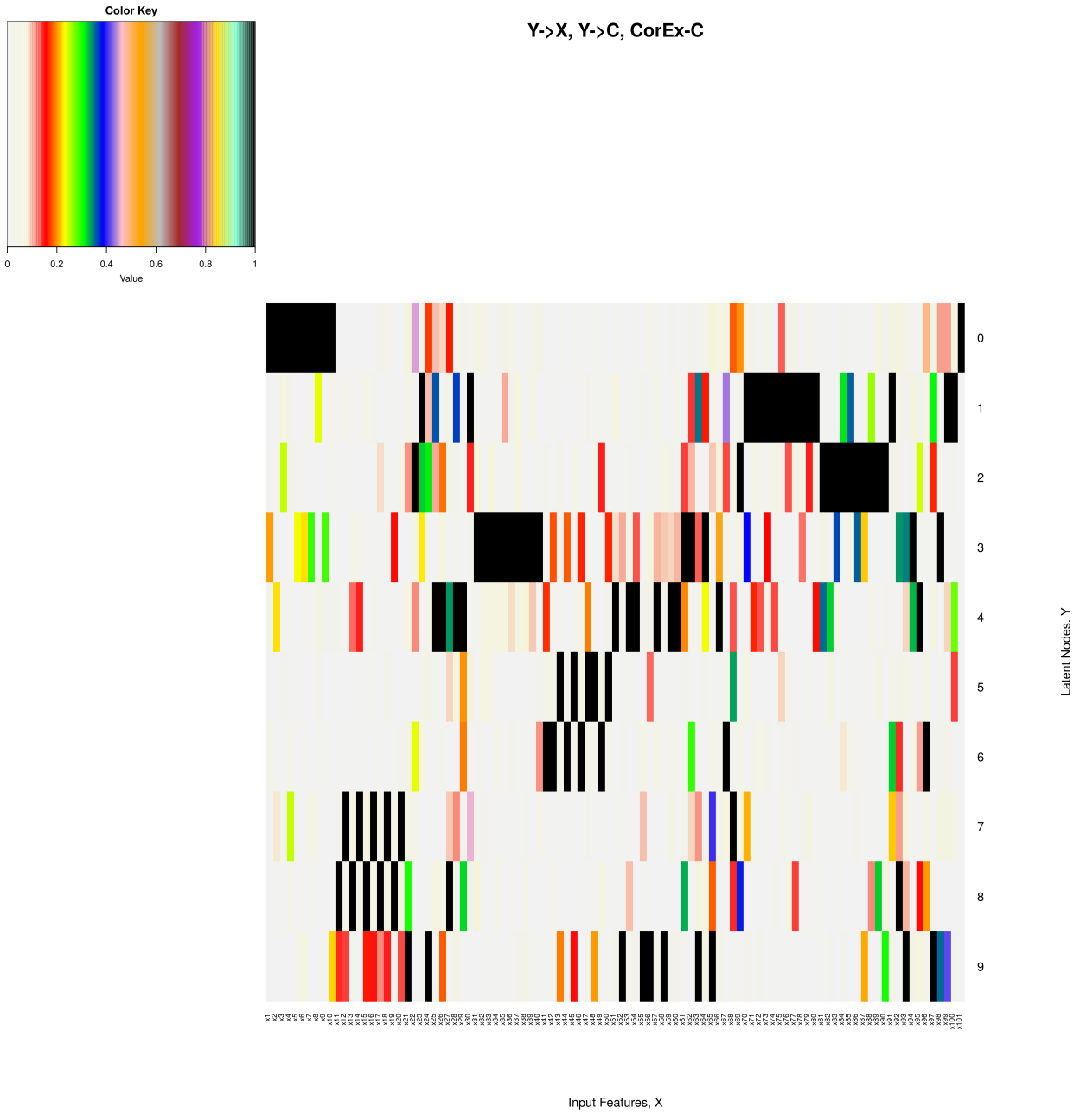


Figure 4.4: The figure shows the heatmap of the α values between the latent variables and the input features with 30000 training samples $Y \rightarrow X$, $Y \rightarrow C$.

While it has to be noted that not all data necessarily follow this assumption, we make the assumption that the datasets in the real world can be linked to an ancestor variable (i.e. latent variable). For example, in the medical field when a disease is diagnosed, the values obtained do not cause the factors of the disease, but the characteristics of the disease causes the test to react in a certain manner. Therefore, we consider this a reasonable assumption to work within. However, we acknowledge that the assumption made should be expanded and is a future direction to be explored.

4.2 Discriminant Classification Method

The experiments in this section were conducted to analyse how well CorEx-C based method works under the discriminant classification umbrella. In order to do so, the feature selection and extraction methods were first trained on the training set and then applied to the test set. Classification was done after the preprocessing step. The classifiers used were SVM with Linear Kernel, SVM with Radial Basis Function (RBF) kernel, K-Nearest Neighbour(KNN) with k=5 and the Gaussian Naive Bayes classifier. The SVM classifiers were chosen as they work well with high dimensional data. However, in the case of HDLSS data, SVMs tend to overfit and exhibit symptoms of data piling (Ahn and Marron, 2010). This causes a large number of support vectors to be projected onto a single point (Ahn *et al.*, 2007) making it difficult for separation between classes. K-Nearest neighbour approach was chosen because it is known to be hit by the curse of dimensionality in HDLSS settings. This is because the distance between the nearest points converges to the distance between the furthest points (Beyer *et al.*, 1999). Using KNN we can then test if a feature extraction method can overcome the curse of dimensionality by reducing the convergence of the distance between the points. The Naive Bayes classifier on the other hand holds the assumption that the input features are independent of each other, given the class labels. Gaussian Naive Bayes was used to allow for computation with real valued attributes. HDLSS data is said to contain redundancies between features, therefore this opens an interesting possibility to test if the redundancy among the data has been reduced and its effect on the classifiers. The classifiers can be found and are implemented in Python using the scikit-learn package (Pedregosa *et al.*, 2011). The experiments run are seeded so that it would be able to produce consistent results.

Throughout this chapter, a particular feature extraction or selection method being competitive with another feature extraction or selection method in lieu of classification performance will be referred to as "feature extraction or selection method

performing competitively with each other". E.g. CorEx-SF_Alpha0 is competitive with CorEx-SFS. The performance of the feature selection or extraction method is based on the classification performance of the features that are extracted and selected by the methods. Classification performance refers to both the accuracy and MCC values.

4.2.1 Data-based Analysis of Classification Performance for Simulated Data

The experiments were conducted using the simulated data discussed in Chapter 3.5. We compared the methods against some of the information theory based feature selection and extraction methods described in Chapter 2.3.3. We set the number of features to be selected and extracted to 10. This was done as there were only 10 features that were relevant to the class label. The number of latent features were also set to 10 as there were 10 latent features that generated all the data.

We also compared them against the classifiers' performances on the raw data, which is the full dataset without any preprocessing applied. Further, we tested the performance after preprocessing against the features which are relevant for the classification task. The relevant features are annotated as R.F. in the tables. We also benchmarked the performance of CorEx-based feature extraction and selection methods with MRMR and CIFE for each of the dataset.

80 Y->X, Y->C

Table 4.1: A table of comparison of accuracy of CorEx-based methods for 80 samples under the $Y \rightarrow X, Y \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	51.32%	35.51%	51.95%	59.74%
CorEx-SF_AlphaNot0 (80%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_Alpha0 (10%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_Alpha0 (20%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_Alpha0 (30%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_Alpha0 (50%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_Alpha0 (80%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_Bidirectional	**59.06%**	**59.06%**	58.36%	59.06%
CorEx-SF_MI (10%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_MI (20%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_MI (30%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_MI (50%)	**59.06%**	**59.06%**	59.06%	59.06%
CorEx-SF_MI (80%)	**59.06%**	**59.06%**	59.08%	59.06%
CorEx-SFS	52.43%	35.51%	48.50%	59.47%
CorEx-UF	58.94%	35.69%	**64.29%**	**62.09%**
R.F.	53.63%	35.52%	50.70%	59.00%

Table 4.2: A table of comparison of MCC for CorEx-based methods for 80 samples under the $Y \to X, Y \to C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.0229	0.0105	0.0002	0.1475
CorEx-SF_AlphaNot0 (80%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_Alpha0 (10%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_Alpha0 (20%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_Alpha0 (30%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_Alpha0 (50%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_Alpha0 (80%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_Bidirectional	**0.1838**	**0.1838**	0.1795	**0.1838**
CorEx-SF_MI (10%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_MI (20%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_MI (30%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_MI (50%)	**0.1838**	**0.1838**	0.1838	**0.1838**
CorEx-SF_MI (80%)	**0.1838**	**0.1838**	0.1813	**0.1838**
CorEx-SFS	0.1062	0.0105	0.0002	0.1623
CorEx-UF	0.1835	0.0324	-0.0348	-0.1168
R.F.	0.1654	0.0105	**0.1940**	0.1819

Table 4.3: A table of comparison of accuracy for feature extraction and selection methods for 80 samples under the $Y \to X, Y \to C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	51.32%	35.51%	**51.95%**	**59.74%**
CIFE	**53.63%**	35.52%	50.70%	59.00%
Infomax ICA	50.86%	35.50%	46.60%	41.17%
KECA	35.50%	35.50%	48.59%	49.28%
MMI	49.37%	**36.46%**	47.76%	43.50%
MRMR	**53.63%**	35.52%	50.70%	59.00%
CorEx-SFS	52.43%	35.51%	48.50%	59.47%
R.F	**53.63%**	35.52%	50.70%	59.00%

Table 4.4: A table of comparison of MCC values for feature extraction and selection methods for 80 samples under the $Y \rightarrow X, Y \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.0229	**0.0105**	0.0002	0.1475
CIFE	**0.1654**	**0.0105**	**0.1940**	**0.1819**
Infomax ICA	0.0031	0.0000	-0.0130	-0.0056
KECA	0.0000	0.0000	-0.0016	-0.0082
MMI	-0.0114	0.0082	0.0058	-0.0060
MRMR	**0.1654**	**0.0105**	**0.1940**	**0.1819**
CorEx-SFS	0.1062	**0.0105**	0.0002	0.1623
R.F.	**0.1654**	**0.0105**	**0.1940**	**0.1819**

The heatmap in Fig.(4.2) for the simulated data using the Y->X, Y->C data generation process with 80 samples show that the class label is connected to most of the relevant features for the classification task. CorEx-SFS performs comparably with the raw data. The MCC values indicates that CorEx-SFS allows the classifiers to be more robust to generalisation compared to the raw data. This could be due to the reduction in noisy features selected. CorEx-SFS is less robust and accurate compared to R.F. because it has not selected all the relevant features.

CorEx-SF performs better than R.F. both in terms of accuracy and ability for data to generalise for all the classifiers except KNN. This shows that under the Y->X, Y->C generative process, the features extracted does encode the relationship between the input features to be used for classification. CorEx-UF on the other hand gives a better accuracy than CorEx-SF and CorEx-SFS for the Naive Bayes and KNN classifiers. However, when comparing the MCC values, it can be seen that CorEx-UF produces a less robust classifier model than its supervised counterparts.

Comparing the CorEx-SFS methods with the other information theory based feature extraction and selection methods, it gives better accuracy than MMI for all the classifiers except SVM with RBF Kernel. However, when comparing the MCC values, CorEx-SFS gives better generalisation than MMI for SVM with RBF Kernel. Features selected through CorEx-SFS leads to a slightly more robust classifier compared to Infomax ICA and KECA in this case. MRMR and CIFE on the other hand performs better or at par with CorEx-SFS, whereas CorEx-SF performs competitively with MRMR and CIFE.

30000 Y->X, Y->C

Table 4.5: A table of comparison of accuracy for CorEx-based methods for 30000 samples under the $Y \rightarrow X, Y \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	53.66%	**61.06%**	50.04%	**59.04%**
CorEx-SF_AlphaNot0 (80%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_Alpha0 (10%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_Alpha0 (20%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_Alpha0 (30%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_Alpha0 (50%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_Alpha0 (80%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_Bidirectional	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_MI (10%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_MI (20%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_MI (30%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_MI (50%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SF_MI (80%)	**59.02%**	59.02%	**59.02%**	59.02%
CorEx-SFS	53.50%	**61.06%**	49.74%	**59.04%**
CorEx-UF	**59.02%**	50.93%	49.91%	59.02%
R.F.	53.01%	60.31%	58.23%	59.02%

Table 4.6: A table of comparison of MCC for CorEx-based methods for 30000 samples under the $Y \to X, Y \to C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.1301	-0.1401	0.0120	**0.1827**
CorEx-SF_AlphaNot0 (80%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_Alpha0 (10%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_Alpha0 (20%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_Alpha0 (30%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_Alpha0 (50%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_Alpha0 (80%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_Bidirectional	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_MI (10%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_MI (20%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_MI (30%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_MI (50%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SF_MI (80%)	**0.1822**	**0.1822**	0.1822	0.1822
CorEx-SFS	0.1259	-0.1401	0.0052	**0.1827**
CorEx-UF	**0.1822**	0.0299	0.0079	0.1822
R.F.	0.1300	0.1803	**0.1835**	0.1822

Table 4.7: A table of comparison of accuracy for feature extraction and selection methods for 30000 samples under the $Y \to X, Y \to C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	**53.66%**	**61.06%**	50.04%	**59.04%**
CIFE	53.01%	60.31%	**58.23%**	59.02%
Infomax ICA	49.06%	51.63%	50.44%	50.27%
KECA	50.64%	48.01%	50.31%	50.34%
MMI	50.23%	49.94%	49.85%	50.60%
MRMR	53.01%	60.31%	**58.23%**	59.02%
CorEx-SFS	53.50%	**61.06%**	49.74%	**59.04%**
R.F	53.01%	60.31%	**58.23%**	59.02%

Table 4.8: A table of comparison of MCC values for feature extraction and selection methods for 30000 samples under the $Y \rightarrow X, Y \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	**0.1301**	-0.1401	0.0120	**0.1827**
CIFE	0.1300	**0.1803**	**0.1835**	0.1822
Infomax ICA	0.0016	0.0025	0.0050	-0.0093
KECA	-0.0021	0.0042	0.0039	-0.0124
MMI	0.0058	-0.0018	-0.0031	0.0025
MRMR	0.1300	**0.1803**	**0.1835**	0.1822
CorEx-SFS	0.1259	-0.1401	0.0052	**0.1827**
R.F.	0.1300	**0.1803**	**0.1835**	0.1822

CorEx-SF performs competitively with R.F. both in terms of accuracy and generalisation capability. This is because all the relevant features have been extracted through the CorEx-C model. CorEx-SFS performs marginally better than CorEx-SF under the SVM with RBF classifier accuracy-wise. However, the ability for the classifier to generalise improves with the use of features extracted by CorEx-SF. CorEx-SF performs bettter than CorEx-UF even as the number of samples increase. This shows that the encoding of the class label can lead to the discovery of the relevant structure of the data in relation to classification.

CorEx-SFS is competitive accuracy-wise with MRMR and CIFE for all the classifiers except KNN. Taking a closer look at the generalisation capability of the classifiers, MRMR and CIFE performs better than CorEx-SFS. On the other hand, CorEx-SF performs competitively with MRMR and CIFE for all classifiers.

The performance of features extracted through CorEx-SF is fairly stable regardless of the number of samples available in the dataset under the Y->X, Y->C condition. This shows that CorEx-SF can be used in HDLSS conditions under the Y->X, Y->C data generation assumption.

80 Y->X->C

Table 4.9: A table of comparison of CorEx-based methods for 80 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	52.56%	64.51%	51.37%	59.74%
CorEx-SF_AlphaNot0 (10%)	41.14%	41.14%	41.14%	41.14%
CorEx-SF_AlphaNot0 (30%)	47.73%	47.73%	47.73%	47.73%
CorEx-SF_AlphaNot0 (50%)	46.77%	53.57%	60.19%	53.57%
CorEx-SF_AlphaNot0 (80%)	49.46%	52.09%	46.57%	53.07%
CorEx-SF_Alpha0 (10%)	41.14%	41.14%	41.14%	41.14%
CorEx-SF_Alpha0 (20%)	47.73%	47.73%	47.73%	47.73%
CorEx-SF_Alpha0 (30%)	61.22%	47.73%	59.14%	47.73%
CorEx-SF_Alpha0 (50%)	49.51%	52.09%	54.04%	52.68%
CorEx-SF_Alpha0 (80%)	50.67%	57.84%	53.09%	53.07%
CorEx-SF_Bidirectional	51.22%	62.37%	49.21%	53.04%
CorEx-SF_MI (10%)	41.14%	41.14%	41.14%	41.14%
CorEx-SF_MI (20%)	41.14%	54.49%	41.14%	41.14%
CorEx-SF_MI (30%)	41.14%	54.49%	42.44%	41.14%
CorEx-SF_MI (50%)	41.14%	64.51%	42.38%	47.04%
CorEx-SF_MI (80%)	51.84%	62.56%	50.82%	49.22%
CorEx-SFS	50.05%	64.51%	52.13%	49.41%
CorEx-UF	35.68%	64.37%	64.34%	63.57%
R.F.	**85.52%**	**68.48%**	**72.64%**	**78.82%**

Table 4.10: A table of comparison of MCC for CorEx-based methods for 80 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.0139	0.0000	0.0096	0.2117
CorEx-SF_AlphaNot0 (10%)	-0.0060	-0.0060	-0.0060	-0.0060
CorEx-SF_AlphaNot0 (30%)	-0.0040	-0.0040	-0.0040	-0.0040
CorEx-SF_AlphaNot0 (50%)	0.0538	0.0294	0.1376	0.0294
CorEx-SF_AlphaNot0 (80%)	0.0596	0.0651	0.0312	0.0409
CorEx-SF_Alpha0 (10%)	-0.0060	-0.0060	-0.0060	-0.0060
CorEx-SF_Alpha0 (20%)	-0.0040	-0.0040	-0.0040	-0.0040
CorEx-SF_Alpha0 (30%)	0.1115	-0.0040	0.1654	-0.0040
CorEx-SF_Alpha0 (50%)	0.0387	0.0651	0.0722	0.0513
CorEx-SF_Alpha0 (80%)	0.0546	0.0486	0.0417	0.0409
CorEx-SF_Bidirectional	0.0497	0.0517	0.0048	0.0408
CorEx-SF_MI (10%)	-0.0060	-0.0060	-0.0060	-0.0060
CorEx-SF_MI (20%)	-0.0060	-0.0154	-0.0060	-0.0060
CorEx-SF_MI (30%)	-0.0060	-0.0154	-0.0199	-0.0060
CorEx-SF_MI (50%)	-0.0060	0.0000	-0.0109	-0.0073
CorEx-SF_MI (80%)	0.0619	0.0651	0.0157	0.0076
CorEx-SFS	0.0170	0	0.0033	0.0344
CorEx-UF	0.0319	-0.0273	-0.0306	-0.0723
R.F.	**0.7223**	**0.2530**	**0.4558**	**0.6081**

Table 4.11: A table of comparison of accuracy for feature extraction and selection methods for 80 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	52.56%	64.51%	51.37%	59.74%
CIFE	**85.52%**	**68.48%**	**72.64%**	**78.82%**
Infomax ICA	50.77%	64.51%	50.90%	47.64%
KECA	64.51%	64.51%	49.27%	51.80%
MMI	52.57%	63.12%	50.55%	54.94%
MRMR	**85.52%**	**68.48%**	**72.64%**	**78.82%**
CorEx-SFS	50.05%	64.51%	52.13%	49.41%
R.F	**85.52%**	**68.48%**	**72.64%**	**78.82%**

Table 4.12: A table of comparison of MCC values for feature extraction and selection methods for 80 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.0139	0.0000	0.0096	0.2117
CIFE	**0.7223**	**0.2530**	**0.4558**	**0.6081**
Infomax ICA	-0.0035	0.0000	0.0022	0.0007
KECA	0.0000	0.0000	-0.0003	-0.0028
MMI	-0.0171	-0.0107	-0.0162	-0.0057
MRMR	**0.7223**	**0.2530**	**0.4558**	**0.6081**
CorEx-SFS	0.0170	0.0000	0.0033	0.0344
R.F.	**0.7223**	**0.2530**	**0.4558**	**0.6081**

As seen from the heatmap in Fig. (4.1), the class label is only mildly related to features that are relevant for classification. This is due to the violation of the CorEx assumption. Thus, it is to be expected that the classification result after preprocessing through CorEx-C does not perform well. The accuracy of the CorEx-SF values increase when the threshold size increases. The accuracy of the classification models decreases as the thresholding values of CorEx-SF decrease. This is because the bigger the threshold value, the more the relevant features which are mildly related will be selected. The classifiers would be able to generalise better when there are more relevant features. However, it has to be noticed that after a certain threshold level, the generalisation capability fluctuates. This could be because more irrelevant information is captured just as more relevant features are captured as the threshold size increases. There is a trade-off that needs to be balanced when increasing the threshold size to ensure that too many irrelevant features are not selected while extracting the relevant features.

CorEx-SF_MI performs worse than the other CorEx-SF methods because CorEx-SF_MI dilutes the weak relationships captured and strengthens the strong relationships. In this case, it will not perform well because the relevant features are detected as having weak relationship with the class label. R.F performs better than the CorEx-based feature extraction and selection methods. This is because the relationships learnt are not accurate. CorEx-SF_Alpha0 with a threshold of 50% can perform marginally better than CorEx-UF and CorEx-SFS MCC-wise.

CorEx-SFS is competitive with KECA accuracy-wise for all the classifiers except for SVM with Linear Kernel and Naive Bayes. The MCC values show that CorEx-SFS is slightly more robust than KECA, MMI, and Infomax ICA. CIFE and MRMR performs much better than CorEx-SFS.

30000 Y->X->C

Table 4.13: A table of comparison of accuracy for CorEx-based methods for 30000 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	88.26%	64.51%	50.08%	67.72%
CorEx-SF_AlphaNot0 (10%)	59.03%	59.03%	35.50%	59.03%
CorEx-SF_AlphaNot0 (30%)	59.03%	59.03%	51.79%	59.03%
CorEx-SF_AlphaNot0 (50%)	59.03%	59.03%	53.71%	59.03%
CorEx-SF_AlphaNot0 (80%)	59.03%	59.03%	54.32%	59.15%
CorEx-SF_Alpha0 (10%)	59.03%	59.03%	35.50%	59.03%
CorEx-SF_Alpha0 (20%)	59.03%	59.03%	51.79%	59.03%
CorEx-SF_Alpha0 (30%)	59.03%	59.03%	54.44%	59.03%
CorEx-SF_Alpha0 (50%)	59.03%	59.03%	53.99%	59.03%
CorEx-SF_Alpha0 (80%)	59.03%	59.03%	52.98%	59.10%
CorEx-SF_Bidirectional	59.03%	59.03%	53.15%	59.08%
CorEx-SF_MI (10%)	58.03%	58.03%	35.50%	58.03%
CorEx-SF_MI (20%)	58.03%	58.03%	35.50%	58.03%
CorEx-SF_MI (30%)	58.03%	58.03%	54.19%	58.03%
CorEx-SF_MI (50%)	58.03%	58.03%	54.94%	58.03%
CorEx-SF_MI (80%)	58.03%	58.03%	53.24%	58.03%
CorEx-SFS	88.28%	64.51%	50.69%	**67.83%**
CorEx-UF	62.37%	54.26%	53.48%	61.58%
R.F.	**88.39%**	**83.14%**	**89.78%**	67.15%

Table 4.14: A table of comparison of MCC values for CorEx-based methods for 30000 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.7554	0.0000	0.0033	0.4182
CorEx-SF_AlphaNot0 (10%)	0.1868	0.1868	0.0000	0.1868
CorEx-SF_AlphaNot0 (30%)	0.1868	0.1868	0.0382	0.1868
CorEx-SF_AlphaNot0 (50%)	0.1868	0.1868	0.1162	0.1868
CorEx-SF_AlphaNot0 (80%)	0.1868	0.1868	0.1115	0.1870
CorEx-SF_Alpha0 (10%)	0.1868	0.1868	0.0000	0.1868
CorEx-SF_Alpha0 (20%)	0.1868	0.1868	0.0382	0.1868
CorEx-SF_Alpha0 (30%)	0.1868	0.1868	0.0920	0.1868
CorEx-SF_Alpha0 (50%)	0.1868	0.1868	0.1096	0.1868
CorEx-SF_Alpha0 (80%)	0.1868	0.1868	0.0651	0.1851
CorEx-SF_Bidirectional	0.1868	0.1868	0.0559	0.1860
CorEx-SF_MI (10%)	0.1665	0.1665	0.0000	0.1665
CorEx-SF_MI (20%)	0.1665	0.1665	0.0000	0.1665
CorEx-SF_MI (30%)	0.1665	0.1665	0.0875	0.1665
CorEx-SF_MI (50%)	0.1665	0.1665	0.0829	0.1665
CorEx-SF_MI (80%)	0.1665	0.1665	0.0197	0.1665
CorEx-SFS	0.7554	0.0000	0.0169	**0.4206**
CorEx-UF	0.2647	0.0089	0.0071	0.1440
R.F.	**0.7604**	**0.6317**	**0.7884**	0.4197

Table 4.15: A table of comparison of accuracy for feature extraction and selection methods for 30000 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	88.26%	64.51%	50.08%	67.72%
CIFE	**88.39%**	**83.14%**	**89.78%**	67.15%
Infomax ICA	54.20%	52.55%	49.84%	52.26%
KECA	51.35%	49.43%	49.58%	52.48%
MMI	47.29%	50.60%	49.98%	48.23%
MRMR	**88.39%**	**83.14%**	**89.78%**	67.15%
CorEx-SFS	88.28%	64.51%	50.69%	**67.83%**
R.F.	**88.39%**	**83.14%**	**89.78%**	67.15%

Table 4.16: A table of comparison of MCC values for feature extraction and selection methods for 30000 samples under the $Y \rightarrow X \rightarrow C$ generation process.

Dataset	SVM:Linear	SVM:RBF	KNN (k=5)	Naive Bayes
Raw Data	0.7554	0.0000	0.0033	0.4182
CIFE	**0.7604**	**0.6317**	**0.7884**	0.4197
Infomax ICA	-0.0064	0.0025	-0.0004	0.0014
KECA	-0.0058	-0.0042	-0.0037	-0.0004
MMI	-0.0589	-0.0050	-0.0003	-0.0511
MRMR	**0.7604**	**0.6317**	**0.7884**	0.4197
CorEx-SFS	0.7554	0.0000	0.0169	**0.4206**
R.F.	**0.7604**	**0.6317**	**0.7884**	0.4197

As seen in Fig.(4.3) it can be seen that only some of the relevant features were strongly detected as related to the class label. The accuracy will be lower for CorEx-C based feature extraction and selection methods. CorEx-SFS is competitive with R.F. under the SVM with Linear Kernel and Naive Bayes classifier. CorEx-SFS gives a slightly better performance than CorEx-UF for all the classifiers. With 30000 samples, CorEx-SF based methods improves its performance compared to when only 80 samples were used but CorEx-SF based methods perform worse than R.F. for all classifiers. The reason is that only some relevant features have been extracted.

CorEx-SFS gives a better accuracy than MMI, KECA, and Infomax ICA for all classifiers. CorEx-SFS when benchmarked with MRMR and CIFE, shows that it is competitive when SVM Linear Kernel and Naive Bayes is used while it performs worse when used with KNN and SVM with RBF Kernel. This could be because SVM with Linear Kernel and Naive Bayes perform linear classification and the data can be linearly separated. This is an improvement over the classification performance when the sample size is very low for the $Y \rightarrow X \rightarrow C$ data generation process.

It can be seen that CorEx-SFS and CorEx-SF become more stable as the number of samples increases. While CorEx-SFS becomes competitive with R.F., CorEx-SF is still performing badly. Hence, under the Y->X->C condition, CorEx-based methods do not perform well.

Summary

1. If the data generation assumption made by CorEx is not violated, CorEx-based feature extraction methods are competitive with R.F., MRMR, and CIFE.

2. If the data generation assumption made by CorEx is violated, MRMR and CIFE generally performs better than the CorEx-based feature selection and extraction methods. However, as the sample size increases, CorEx-SFS starts becoming competitive with MRMR, CIFE, and R.F..

3. CorEx-based feature extraction and selection methods, regardless of the data generation process, performs better than Infomax ICA, MMI, and KECA with the simulated dataset.

4.2.2 Data-based Analysis of Classification Performance for Real World Data

The datasets used in this work are Alon's Colon Cancer, Gravier's Breast Cancer, Golub's Leukemia and Gisette datasets. The datasets were balanced by randomly down-sampling the majority class to match the number of samples in the minority class. This is to ensure that the effects of imbalanced data do not skew the results produced from the experiment. The datasets were chosen based on the ratio of attributes to instances under the HDLSS regime.

Alon

The dataset was used in Alon *et al.* (1999) and contains gene expression levels of 40 tumours and 22 normal colon tissues for 6500 human genes obtained with an affymetrix oligonucleotide array. To avoid problems relating to bias of class imbalance, the dataset was down-sampled to contain 22 samples from each class. It is a binary classification problem.

Gravier

This breast cancer dataset was used in Gravier, Eleonore *et al.* (2010). Genomic hybridization arrays were used to observe 168 patients over a 5-year period of which 111 patients had no event after diagnosis and 57 others showed early metastasis. The dataset was downsampled to contain 57 samples from each of the classes.

Gisette

The Gisette dataset (Guyon *et al.*, 2005) is a subset of MNIST from samples of classes 4 and 9. The original Gisette dataset contains 13500 samples. For this experiment, Gisette was down sampled and balanced to contain only 4000 samples.

Golub

Golub data (Golub *et al.*, 1999) contains 72 instances of leukaemia patients who were diagnosed with either acute lymphoblastic leukaemia (ALL) and patients with acute myeloid leukaemia (AML). Random down-sampling was done on the majority class to match the number of instances of the minority class. The total number of instances was brought down to 50.

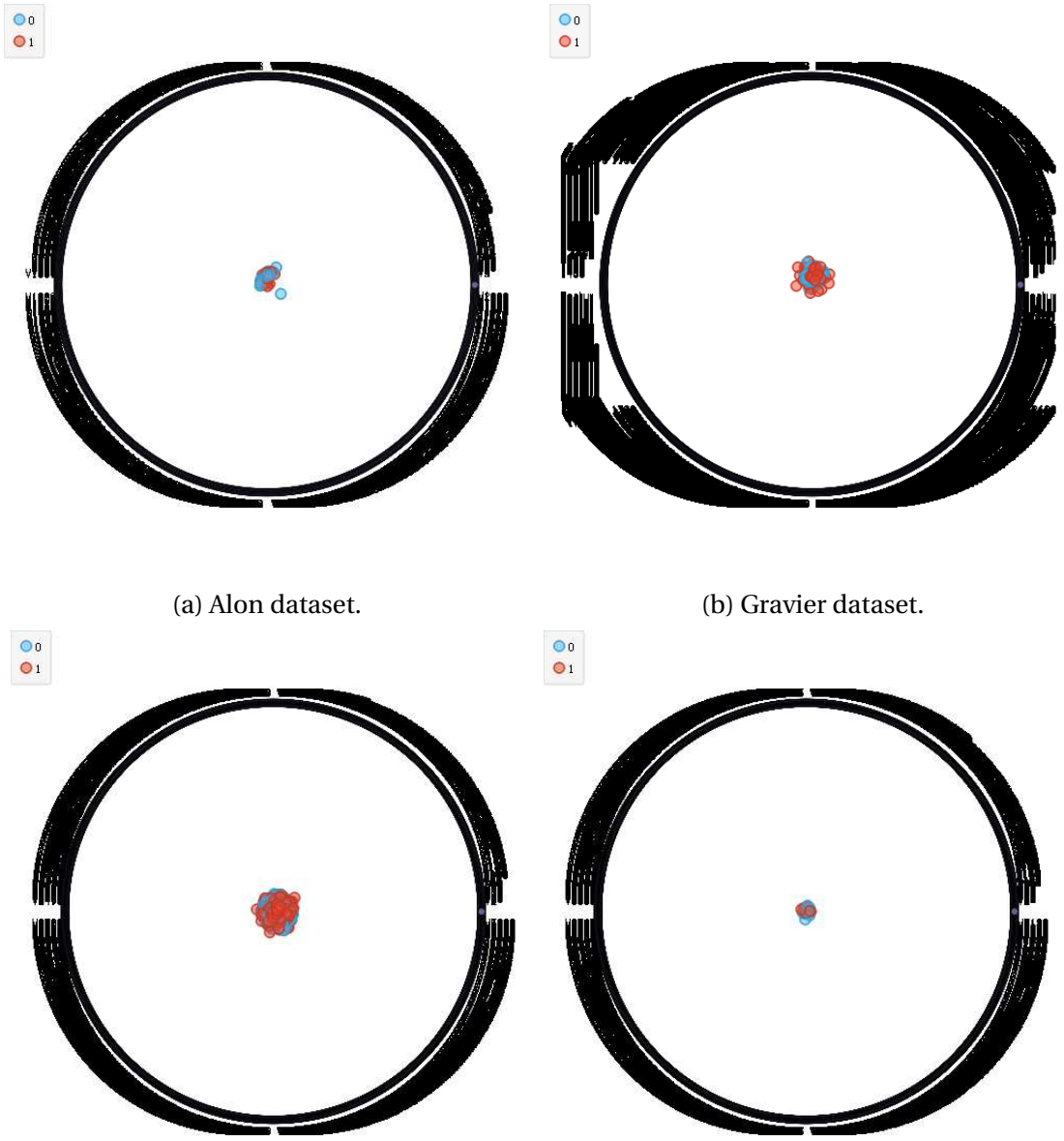
Table 4.17: The table shows the attributes of the datasets used in the experiments.

Dataset	# Features	# Samples	#Class
Alon	2000	44	2
Gisette	5000	4000	2
Golub	7129	50	2
Gravier	2905	114	2

The datasets were split into training and test sets. All the datasets with the exception of the Golub dataset were split such that 60% of the data was placed into the training set and 40% of the data into the test set. The Golub dataset has a 50% training sample and 50% testing sample split due to its large number of dimensions which leads to a longer computational time.

The number of features extracted by CorEx-UF, CorEx-SF, CIFE, and MRMR are 150 dimensions and 515 dimensions respectively for all the datasets. The number of features chosen is still high dimensional in comparison to the number of samples per class. The objective of the experiments is to show that reducing the number of dimensions can help improve the performance of the classifiers. The number of features extracted by KECA and ICA on the other hand is in accordance to the number of samples available in the dataset for all datasets except Gisette which was extracted with 150 and 515 features. This is because these methods are not able to handle cases where the transformed data is higher than the number of samples. In the case of MMI the number of extracted dimensions that we have chosen is $n-1$ where n is the number of sample instances for all datasets except in the case of the Gisette dataset. The reason $n-1$ dimensions were chosen rather than n is due to the rank deficiency when n dimensions are used. A smaller number of dimensions could be chosen, however, to be consistent with KECA and ICA which uses n dimensions, we have chosen to use $n-1$ dimensions. MMI was initialised using Linear Discriminant Analysis (LDA). In the case of the Gisette dataset, 150 and 515 features were chosen for all the feature extraction methods as the number of sample instances exceeds the dimensions at 150 and 515.

Visualisation of the Real World Datasets



(a) Alon dataset.

(b) Gravier dataset.

(c) Gisette dataset

(d) Golub dataset

Figure 4.5: Visualisations of the datasets used using Radviz generated by scOrange software (Demšar *et al.*, 2013). Radviz visualisation tool is based upon dimensional anchors which are used to aid visualisation of high dimensional datasets in a 2D projection. The concept was introduced by Hoffman *et al.* (1999).

Graphs were plotted using Radviz in order to view the geometric distribution of the samples in the high dimensional feature space. Radviz considers features to be dimensional anchors which are independent of one another (Kuntal *et al.*, 2014). The visualisations aids in checking if the data points pile on each other (Ahn and Marron, 2010)

or if the data points converges to be equidistant from each other (Hall *et al.*, 2005). The graphs in Figure 4.5 shows that the data points for all the graphs seem to pile on one another. This would mean that the separating hyperplane would not very easily be able to discriminate between classes. The graph shows that the datasets have some of the geometrical structure of HDLSS datasets. Hence, investigating the performance of the classifiers in order to benchmark on these data under HDLSS umbrella is justifiable.

In this section, we compare the performance of features extracted or selected through CorEx-based methods on different real world datasets. We benchmark the performance against the raw data. We also investigate the effect of the number of nodes used on the performance of the classifiers for each of the datasets.

Golub Dataset

Table 4.18: A table of comparison of accuracy of CorEx-based feature extraction and selection methods for Golub dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	96.00%	48.00%	**96.00%**	**96.00%**
CorEx-SF 150 Dim (100%)	96.00%	56.00%	**96.00%**	68.00%
CorEx-SF_Alpha0 150 Dim (10%)	96.00%	**88.00%**	**96.00%**	92.00%
CorEx-SF_Alpha0 150 Dim (30%)	96.00%	80.00%	**96.00%**	76.00%
CorEx-SF_Alpha0 150 Dim (50%)	92.00%	**88.00%**	**96.00%**	72.00%
CorEx-SF_Alpha0 150 Dim (80%)	92.00%	72.00%	**96.00%**	68.00%
CorEx-SF_AlphaNot0 150 Dim (10%)	48.00%	48.00%	60.00%	48.00%
CorEx-SF_AlphaNot0 150 Dim (30%)	48.00%	48.00%	60.00%	48.00%
CorEx-SF_AlphaNot0 150 Dim (50%)	48.00%	48.00%	60.00%	48.00%
CorEx-SF_AlphaNot0 150 Dim (80%)	48.00%	48.00%	60.00%	48.00%
CorEx-SF_Bidirectional 150 Dim	92.00%	64.00%	**96.00%**	60.00%
CorEx-SF_MI 150 Dim (10%)	96.00%	**88.00%**	**96.00%**	92.00%
CorEx-SF_MI 150 Dim (30%)	96.00%	80.00%	**96.00%**	76.00%
CorEx-SF_MI 150 Dim (50%)	92.00%	**88.00%**	**96.00%**	72.00%
CorEx-SF_MI 150 Dim (80%)	92.00%	72.00%	**96.00%**	68.00%
CorEx-SFS 150 Dim	**100.00%**	48.00%	**96.00%**	**96.00%**
CorEx-UF 150 Dim	92.00%	68.00%	76.00%	76.00%
CorEx-SF 515 Dim (100%)	96.00%	72.00%	**96.00%**	84.00%
CorEx-SF_Alpha0 515 Dim (10%)	96.00%	**88.00%**	92.00%	88.00%
CorEx-SF_Alpha0 515 Dim (30%)	92.00%	80.00%	**96.00%**	76.00%
CorEx-SF_Alpha0 515 Dim (50%)	92.00%	76.00%	92.00%	76.00%
CorEx-SF_Alpha0 515 Dim (80%)	96.00%	84.00%	**96.00%**	76.00%
CorEx-SF_AlphaNot0 515 Dim (10%)	48.00%	48.00%	40.00%	60.00%
CorEx-SF_AlphaNot0 515 Dim (30%)	48.00%	48.00%	40.00%	60.00%
CorEx-SF_AlphaNot0 515 Dim (50%)	48.00%	48.00%	40.00%	60.00%
CorEx-SF_AlphaNot0 515 Dim (80%)	48.00%	48.00%	40.00%	60.00%
CorEx-SF_Bidirectional 515 Dim	80.00%	60.00%	76.00%	56.00%
CorEx-SF_MI 515 Dim (10%)	96.00%	**88.00%**	92.00%	88.00%
CorEx-SF_MI 515 Dim (30%)	92.00%	80.00%	**96.00%**	76.00%
CorEx-SF_MI 515 Dim (50%)	92.00%	76.00%	92.00%	76.00%
CorEx-SF_MI 515 Dim (80%)	96.00%	84.00%	**96.00%**	76.00%
CorEx-SFS 515 Dim	96.00%	48.00%	**96.00%**	**96.00%**
CorEx-UF 515 Dim	92.00%	72.00%	**96.00%**	80.00%

Table 4.19: A table of comparison of MCC of CorEx-based feature extraction and selection methods for Golub dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	0.9231	0.0000	**0.9231**	**0.9231**
CorEx-SF 150 Dim (100%)	0.9226	0.2833	0.9226	0.3590
CorEx-SF_Alpha0 150 Dim (10%)	0.9226	0.7806	0.9226	0.8397
CorEx-SF_Alpha0 150 Dim (30%)	0.9226	0.6138	0.9226	0.5424
CorEx-SF_Alpha0 150 Dim (50%)	0.8397	**0.7845**	0.9226	0.4470
CorEx-SF_Alpha0 150 Dim (80%)	0.8498	0.5399	0.9226	0.3590
CorEx-SF_AlphaNot0 150 Dim (10%)	0.0000	0.0000	0.1987	0.0000
CorEx-SF_AlphaNot0 150 Dim (30%)	0.0000	0.0000	0.1987	0.0000
CorEx-SF_AlphaNot0 150 Dim (50%)	0.0000	0.0000	0.1987	0.0000
CorEx-SF_AlphaNot0 150 Dim (80%)	0.0000	0.0000	0.1987	0.0000
CorEx-SF_Bidirectional 150 Dim	0.8397	0.2942	0.9226	0.2065
CorEx-SF_MI 150 Dim (10%)	0.9226	0.7806	0.9226	0.8397
CorEx-SF_MI 150 Dim (30%)	0.9226	0.6138	0.9226	0.5424
CorEx-SF_MI 150 Dim (50%)	0.8397	**0.7845**	0.9226	0.4470
CorEx-SF_MI 150 Dim (80%)	0.8498	0.5399	0.9226	0.3590
CorEx-SFS 150 Dim	**1.0000**	0.0000	0.9226	**0.9231**
CorEx-UF 150 Dim	0.8498	0.4804	0.5849	0.5424
CorEx-SF 515 Dim (100%)	0.9231	0.5399	0.9226	0.6903
CorEx-SF_Alpha0 515 Dim (10%)	0.9231	0.7628	0.8397	0.7628
CorEx-SF_Alpha0 515 Dim (30%)	0.8397	0.6138	0.9226	0.5192
CorEx-SF_Alpha0 515 Dim (50%)	0.8516	0.5290	0.8498	0.5192
CorEx-SF_Alpha0 515 Dim (80%)	0.9231	0.7206	0.9226	0.5192
CorEx-SF_AlphaNot0 515 Dim (10%)	0.0000	0.0000	-0.2202	0.2202
CorEx-SF_AlphaNot0 515 Dim (30%)	0.0000	0.0000	-0.2202	0.2202
CorEx-SF_AlphaNot0 515 Dim (50%)	0.0000	0.0000	-0.2202	0.2202
CorEx-SF_AlphaNot0 515 Dim (80%)	0.0000	0.0000	-0.2202	0.2202
CorEx-SF_Bidirectional 515 Dim	0.6026	0.2202	0.5849	0.1307
CorEx-SF_MI 515 Dim (10%)	0.9231	0.7628	0.8397	0.7628
CorEx-SF_MI 515 Dim (30%)	0.8397	0.6138	0.9226	0.5192
CorEx-SF_MI 515 Dim (50%)	0.8516	0.5290	0.8498	0.5192
CorEx-SF_MI 515 Dim (80%)	0.9231	0.7206	0.9226	0.5192
CorEx-SFS 515 Dim	0.9226	0.0000	**0.9231**	0.9226
CorEx-UF 515 Dim	0.8516	0.5399	0.9226	0.6591

As seen from Table (4.18), CorEx-SF performs better or at par with the raw data for all the classifiers except KNN. This shows that CorEx-SF is able to extract important information related to the class label and minimise information loss through its compression. SVM with RBF Kernel does not perform well for the Golub raw data whereas SVM with Linear Kernel performs well. The reason behind this is that the need to map the dataset into a much higher dimension than what it already is in may not be there (Hsu *et al.*, 2003). This would mean that non-linear mapping by the RBF Kernel will not improve the performance as it increases the sparsity and complexity of the data space. CorEx-SF for Golub dataset with 150 nodes shows an underperformed generalization on SVM with RBF Kernel and KNN when the threshold value is set at 100% but improves significantly as the threshold values get smaller through the use of CorEx-SF_Alpha0 and CorEx-SF_MI. A reason for this could be that less irrelevant features are being captured as the threshold size decreases. CorEx-SFS can be seen to perform competitively with CorEx-SF, while CorEx-UF performs worse compared to CorEx-SFS for all classifiers except for SVM with RBF Kernel. Using 150 nodes, show the best performance across the different feature selection and extraction methods for the Golub dataset.

Alon Dataset

Table 4.20: The table shows the comparison of accuracy values of CorEx-based feature extraction and selection methods for Alon dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	88.89%	50.00%	66.67%	66.67%
CorEx-SF 150 Dim (100%)	77.78%	88.89%	**77.78%**	83.33%
CorEx-SF_Alpha0 150 Dim (10%)	88.89%	88.89%	50.00%	77.78%
CorEx-SF_Alpha0 150 Dim (30%)	66.67%	83.33%	50.00%	66.67%
CorEx-SF_Alpha0 150 Dim (50%)	83.33%	83.33%	61.11%	72.22%
CorEx-SF_Alpha0 150 Dim (80%)	77.78%	88.89%	**77.78%**	72.22%
CorEx-SF_AlphaNot0 150 Dim (10%)	61.11%	61.11%	61.11%	61.11%
CorEx-SF_AlphaNot0 150 Dim (30%)	61.11%	61.11%	61.11%	61.11%
CorEx-SF_AlphaNot0 150 Dim (50%)	61.11%	61.11%	61.11%	61.11%
CorEx-SF_AlphaNot0 150 Dim (80%)	61.11%	61.11%	61.11%	66.67%
CorEx-SF_Bidirectional 150 Dim	66.67%	55.56%	50.00%	55.56%
CorEx-SF_MI 150 Dim (10%)	88.89%	88.89%	50.00%	**88.89%**
CorEx-SF_MI 150 Dim (30%)	72.22%	72.22%	50.00%	55.56%
CorEx-SF_MI 150 Dim (50%)	83.33%	83.33%	61.11%	72.22%
CorEx-SF_MI 150 Dim (80%)	77.78%	83.33%	**77.78%**	83.33%
CorEx-SFS 150 Dim	77.78%	50.00%	61.11%	77.78%
CorEx-UF 150 Dim	**88.90%**	**88.90%**	61.11%	83.33%
CorEx-SF 515 Dim (100%)	55.56%	55.56%	50.00%	50.00%
CorEx-SF_Alpha0 515 Dim (10%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_Alpha0 515 Dim (30%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_Alpha0 515 Dim (50%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_Alpha0 515 Dim (80%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_AlphaNot0 515 Dim (10%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_AlphaNot0 515 Dim (30%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_AlphaNot0 515 Dim (50%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_AlphaNot0 515 Dim (80%)	50.00%	50.00%	50.00%	50.00%
CorEx-SF_Bidirectional 515 Dim	50.00%	61.11%	50.00%	55.56%
CorEx-SF_MI 515 Dim (10%)	55.56%	55.56%	50.00%	50.00%
CorEx-SF_MI 515 Dim (30%)	55.56%	55.56%	50.00%	50.00%
CorEx-SF_MI 515 Dim (50%)	55.56%	55.56%	50.00%	50.00%
CorEx-SF_MI 515 Dim (80%)	55.56%	55.56%	50.00%	50.00%
CorEx-SFS 515 Dim	88.89%	50.00%	66.67%	66.67%
CorEx-UF 515 Dim	61.11%	61.11%	61.11%	61.11%

Table 4.21: A table of comparison of MCC values of CorEx-based feature extraction and selection methods for Alon dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	0.7778	0.0000	0.4472	0.4472
CorEx-SF 150 Dim (100%)	0.5698	**0.7977**	**0.6202**	0.6708
CorEx-SF_Alpha0 150 Dim (10%)	**0.7977**	**0.7977**	0.0000	0.5698
CorEx-SF_Alpha0 150 Dim (30%)	0.3721	0.6708	0.0000	0.3721
CorEx-SF_Alpha0 150 Dim (50%)	0.7071	0.6708	0.3536	0.4472
CorEx-SF_Alpha0 150 Dim (80%)	0.5556	**0.7977**	**0.6202**	0.4714
CorEx-SF_AlphaNot0 150 Dim (10%)	0.3536	0.3536	0.3536	0.3536
CorEx-SF_AlphaNot0 150 Dim (30%)	0.3536	0.3536	0.3536	0.3536
CorEx-SF_AlphaNot0 150 Dim (50%)	0.3536	0.3536	0.3536	0.3536
CorEx-SF_AlphaNot0 150 Dim (80%)	0.3536	0.3536	0.3536	0.4472
CorEx-SF_Bidirectional 150 Dim	0.3419	0.1491	0.0000	0.1240
CorEx-SF_MI 150 Dim (10%)	**0.7977**	0.7778	0.0000	**0.7977**
CorEx-SF_MI 150 Dim (30%)	0.4714	0.4714	0.0000	0.1491
CorEx-SF_MI 150 Dim (50%)	0.7071	0.6708	0.3536	0.4472
CorEx-SF_MI 150 Dim (80%)	0.5556	0.6708	**0.6202**	0.6708
CorEx-SFS 150 Dim	0.5556	0.0000	0.2673	0.5556
CorEx-UF 150 Dim	0.7778	**0.7977**	0.3536	0.6708
CorEx-SF 515 Dim (100%)	0.1491	0.1491	0.0000	0.0000
CorEx-SF_Alpha0 515 Dim (10%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_Alpha0 515 Dim (30%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_Alpha0 515 Dim (50%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_Alpha0 515 Dim (80%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_AlphaNot0 515 Dim (10%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_AlphaNot0 515 Dim (30%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_AlphaNot0 515 Dim (50%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_AlphaNot0 515 Dim (80%)	0.0000	0.0000	0.0000	0.0000
CorEx-SF_Bidirectional 515 Dim	0.0000	0.3536	0.0000	0.1240
CorEx-SF_MI 515 Dim (10%)	0.1491	0.1491	0.0000	0.0000
CorEx-SF_MI 515 Dim (30%)	0.1491	0.1491	0.0000	0.0000
CorEx-SF_MI 515 Dim (50%)	0.1491	0.1491	0.0000	0.0000
CorEx-SF_MI 515 Dim (80%)	0.1491	0.1491	0.0000	0.0000
CorEx-SFS 515 Dim	0.7778	0.0000	0.4472	0.4472
CorEx-UF 515 Dim	0.3536	0.3536	0.3536	0.3536

For the Alon dataset, CorEx-UF performs well with 150 nodes but its classification performance degenerates when the number of nodes is 515. This could be due to the fact that for each dataset, a different number of dimension is optimal to encode the information contained within the different datasets. This logic runs along the line of the intrinsic dimensionality of a dataset which refers to each dataset having its own true dimensionality. CorEx-SF performs competitively with CorEx-UF when 150 nodes are used. CorEx-SFS with 515 nodes works as well as the raw data but performs better than the raw data when 150 nodes are used.

It can be seen from Table (4.20) that KNN, SVM with RBF Kernel and Naive Bayes are hit by the curse of dimensionality when raw data is used. This conclusion is drawn from the fact that with feature extraction, the quality of the generalization of the classifier has improved. When looking at the effects of CorEx-SF_MI and CorEx-SF_Alpha0 with 150 nodes with the threshold set at 10%, the performance of the classifiers tend to improve with the exception of Naive Bayes. The Naive Bayes classifier works as well as random guessing with threshold 10%. CorEx-SF with 150 nodes perform competitively and in some cases, better than the raw data. This shows that relevant relationships and encodings between the features and the class label are learnt. CorEx-SF based methods with 150 nodes performs better than CorEx-SFS with 150 nodes. This is due to the reduction of complexity of the data that needs to be modelled as the redundancy reduces when feature extraction is done compared to feature selection. However, in the case of 515 nodes, CorEx-SFS performs better than CorEx-SF based methods.

Gravier Dataset

Table 4.22: A table of comparison of accuracy of CorEx-based feature extraction and selection methods for Gravier dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	70.21%	40.43%	**76.60%**	61.70%
CorEx-SF 150 Dim (100%)	70.21%	76.60%	57.44%	**82.98%**
CorEx-SF_Alpha0 150 Dim (10%)	61.70%	61.70%	61.70%	63.30%
CorEx-SF_Alpha0 150 Dim (30%)	61.70%	72.34%	68.09%	76.60%
CorEx-SF_Alpha0 150 Dim (50%)	63.83%	78.72%	70.21%	72.34%
CorEx-SF_Alpha0 150 Dim (80%)	72.34%	76.60%	70.21%	76.60%
CorEx-SF_AlphaNot0 150 Dim (10%)	61.70%	61.70%	61.70%	61.70%
CorEx-SF_AlphaNot0 150 Dim (30%)	61.70%	61.70%	61.70%	61.70%
CorEx-SF_AlphaNot0 150 Dim (50%)	61.70%	61.70%	61.70%	61.70%
CorEx-SF_AlphaNot0 150 Dim (80%)	61.70%	61.70%	61.70%	61.70%
CorEx-SF_Bidirectional 150 Dim	70.21%	27.66%	72.34%	27.66%
CorEx-SF_MI 150 Dim (10%)	55.32%	55.32%	55.32%	57.45%
CorEx-SF_MI 150 Dim (30%)	55.32%	74.47%	57.45%	74.46%
CorEx-SF_MI 150 Dim (50%)	57.45%	78.72%	57.45%	74.47%
CorEx-SF_MI 150 Dim (80%)	65.96%	**80.85%**	57.45%	78.72%
CorEx-SFS 150 Dim	57.45%	46.81%	55.32%	59.57%
CorEx-UF 150 Dim	68.09%	78.72%	**76.60%**	74.47%
CorEx-SF 515 Dim (100%)	65.96%	72.34%	57.45%	74.47%
CorEx-SF_Alpha0 515 Dim (10%)	46.81%	57.45%	53.19%	65.96%
CorEx-SF_Alpha0 515 Dim (30%)	61.70%	72.34%	57.45%	70.21%
CorEx-SF_Alpha0 515 Dim (50%)	**76.60%**	74.47%	57.45%	72.34%
CorEx-SF_Alpha0 515 Dim (80%)	**76.60%**	**80.85%**	57.44%	74.47%
CorEx-SF_AlphaNot0 515 Dim (10%)	63.82%	65.96%	68.09%	57.45%
CorEx-SF_AlphaNot0 515 Dim (30%)	61.70%	59.57%	51.06%	63.83%
CorEx-SF_AlphaNot0 515 Dim (50%)	57.45%	61.70%	57.45%	63.30%
CorEx-SF_AlphaNot0 515 Dim (80%)	53.19%	63.83%	61.70%	65.96%
CorEx-SF_Bidirectional 515 Dim	**76.60%**	76.60%	63.83%	76.60%
CorEx-SF_MI 515 Dim (10%)	48.94%	59.57%	53.19%	70.21%
CorEx-SF_MI 515 Dim (30%)	61.70%	74.47%	55.32%	74.47%
CorEx-SF_MI 515 Dim (50%)	70.21%	76.60%	55.32%	74.47%
CorEx-SF_MI 515 Dim (80%)	68.09%	74.47%	57.45%	76.60%
CorEx-SFS 515 Dim	61.70%	59.57%	61.70%	59.57%
CorEx-UF 515 Dim	74.47%	76.60%	51.06%	70.21%

Table 4.23: A table of comparison of MCC values of CorEx-based feature extraction and selection methods for Gravier dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	0.3724	0.0000	**0.5306**	0.1790
CorEx-SF 150 Dim (100%)	0.4706	0.5187	0.3731	**0.6451**
CorEx-SF_Alpha0 150 Dim (10%)	0.1920	0.1920	0.2346	0.2699
CorEx-SF_Alpha0 150 Dim (30%)	0.1920	0.4310	0.3915	0.5063
CorEx-SF_Alpha0 150 Dim (50%)	0.2556	0.5529	0.4459	0.4115
CorEx-SF_Alpha0 150 Dim (80%)	0.4595	0.5105	0.4459	0.5063
CorEx-SF_AlphaNot0 150 Dim (10%)	0.1920	0.1920	0.1920	0.1920
CorEx-SF_AlphaNot0 150 Dim (30%)	0.1920	0.1920	0.1920	0.1920
CorEx-SF_AlphaNot0 150 Dim (50%)	0.1920	0.1920	0.1920	0.1920
CorEx-SF_AlphaNot0 150 Dim (80%)	0.1920	0.1920	0.1920	0.1920
CorEx-SF_Bidirectional 150 Dim	0.4078	-0.4437	0.4213	-0.4437
CorEx-SF_MI 150 Dim (10%)	0.2907	0.2907	0.2907	0.2521
CorEx-SF_MI 150 Dim (30%)	0.2907	0.5336	0.3731	0.4627
CorEx-SF_MI 150 Dim (50%)	0.3223	0.5678	0.3731	0.4591
CorEx-SF_MI 150 Dim (80%)	0.4088	0.6064	0.3731	0.5568
CorEx-SFS 150 Dim	0.1018	0.1436	0.1299	0.1402
CorEx-UF 150 Dim	0.4396	0.5678	**0.5306**	0.4662
CorEx-SF 515 Dim (100%)	0.3582	0.4595	0.2835	0.4662
CorEx-SF_Alpha0 515 Dim (10%)	-0.0445	0.2226	0.0952	0.3061
CorEx-SF_Alpha0 515 Dim (30%)	0.2705	0.4437	0.2030	0.3661
CorEx-SF_Alpha0 515 Dim (50%)	**0.5306**	0.4945	0.2030	0.4192
CorEx-SF_Alpha0 515 Dim (80%)	**0.5306**	**0.6175**	0.2258	0.4662
CorEx-SF_AlphaNot0 515 Dim (10%)	0.2860	0.3582	0.4135	0.2258
CorEx-SF_AlphaNot0 515 Dim (30%)	0.2346	0.2368	0.1217	0.2556
CorEx-SF_AlphaNot0 515 Dim (50%)	0.1828	0.2514	0.2258	0.2429
CorEx-SF_AlphaNot0 515 Dim (80%)	0.0952	0.3252	0.3173	0.3061
CorEx-SF_Bidirectional 515 Dim	0.5061	0.5061	0.2860	0.5061
CorEx-SF_MI 515 Dim (10%)	0.0247	0.2592	0.1139	0.3816
CorEx-SF_MI 515 Dim (30%)	0.2514	0.4806	0.1920	0.4662
CorEx-SF_MI 515 Dim (50%)	0.3934	0.5306	0.1920	0.4662
CorEx-SF_MI 515 Dim (80%)	0.3728	0.4945	0.2835	0.5120
CorEx-SFS 515 Dim	0.1590	0.1142	0.1590	0.1014
CorEx-UF 515 Dim	0.4806	0.5187	0.2228	0.3704

For the Gravier dataset, it can be seen from Table (4.22) that CorEx-UF performs competitively with CorEx-SF_MI and raw data. It has to be noticed however, that CorEx-UF with 150 nodes for the SVM with Linear Kernel performs marginally worse than that of the raw data. A closer look at the MCC score in Table (4.23) shows that CorEx-UF with 150 nodes for SVM with Linear Kernel is able to produce a slightly better model than that of the raw data. This shows that CorEx-UF has less false positives and negatives making the model more generalized and robust than the raw data. CorEx-SF shows improvement over the raw data. This shows that CorEx-C based methods can learn relevant features which are related to the class label with the Gravier dataset. CorEx-SF performs better than CorEx-SFS. The different number of nodes used, different CorEx-based methods perform well. There is no one clear advantage of using either 150 or 515 nodes. However, this would point to the fact that there may be a different number of nodes which may produce more accurate performance for this dataset.

Gisette Dataset

Table 4.24: The table shows the comparison of accuracy of CorEx-based feature extraction and selection methods for the Gisette dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	**96.88%**	50.00%	74.06%	95.81%
CorEx-SF 150 Dim (100%)	93.19%	95.06%	89.13%	97.00%
CorEx-SF_Alpha0 150 Dim (10%)	89.56%	93.19%	87.19%	92.56%
CorEx-SF_Alpha0 150 Dim (30%)	90.06%	91.63%	87.44%	92.00%
CorEx-SF_Alpha0 150 Dim (50%)	92.00%	93.38%	88.69%	93.00%
CorEx-SF_Alpha0 150 Dim (80%)	92.44%	94.56%	87.63%	95.06%
CorEx-SF_AlphaNot0 150 Dim (10%)	64.56%	64.56%	64.56%	64.56%
CorEx-SF_AlphaNot0 150 Dim (30%)	64.56%	67.94%	63.44%	63.50%
CorEx-SF_AlphaNot0 150 Dim (50%)	72.69%	80.75%	70.31%	77.94%
CorEx-SF_AlphaNot0 150 Dim (80%)	72.63%	80.19%	71.06%	76.69%
CorEx-SF_Bidirectional 150 Dim	75.13%	82.63%	71.44%	81.38%
CorEx-SF_MI 150 Dim (10%)	89.69%	93.31%	87.25%	93.88%
CorEx-SF_MI 150 Dim (30%)	89.50%	92.38%	87.56%	92.25%
CorEx-SF_MI 150 Dim (50%)	92.00%	93.38%	88.69%	93.00%
CorEx-SF_MI 150 Dim (80%)	92.44%	94.56%	87.63%	95.05%
CorEx-SFS 150 Dim	92.31%	79.69%	85.94%	93.88%
CorEx-UF 150 Dim	93.06%	95.00%	89.06%	96.44%
CorEx-SF 515 Dim (100%)	92.13%	94.75%	89.75%	96.88%
CorEx-SF_Alpha0 515 Dim (10%)	46.81%	57.45%	53.19%	65.96%
CorEx-SF_Alpha0 515 Dim (30%)	88.88%	89.63%	87.88%	83.00%
CorEx-SF_Alpha0 515 Dim (50%)	87.69%	89.38%	87.56%	83.44%
CorEx-SF_Alpha0 515 Dim (80%)	91.81%	94.06%	89.19%	95.56%
CorEx-SF_AlphaNot0 515 Dim (10%)	59.88%	60.81%	59.00%	57.50%
CorEx-SF_AlphaNot0 515 Dim (30%)	84.81%	86.44%	82.69%	84.19%
CorEx-SF_AlphaNot0 515 Dim (50%)	86.31%	87.44%	82.38%	84.44%
CorEx-SF_AlphaNot0 515 Dim (80%)	88.75%	89.50%	83.50%	87.94%
CorEx-SF_Bidirectional 515 Dim	92.00%	67.31%	89.81%	68.06%
CorEx-SF_MI 515 Dim (10%)	89.25%	90.31%	87.69%	89.75%
CorEx-SF_MI 515 Dim (30%)	88.63%	89.63%	88.38%	86.75%
CorEx-SF_MI 515 Dim (50%)	91.19%	92.75%	**90.25%**	91.94%
CorEx-SF_MI 515 Dim (80%)	91.88%	94.00%	88.88%	95.38%
CorEx-SFS 515 Dim	**96.88%**	50.00%	74.06%	95.81%
CorEx-UF 515 Dim	93.81%	**95.31%**	88.69%	**97.13%**

Table 4.25: The table shows the Matthew's Correlation Coefficient values of CorEx-based feature extraction and selection methods for Gisette dataset.

Classifier	SVM: Linear	SVM:RBF	Naive Bayes	KNN(k=5)
Raw Data	**0.93750**	0.00000	0.53340	0.91651
CorEx-SF 150 Dim (100%)	0.86378	0.90127	0.78486	0.94003
CorEx-SF_Alpha0 150 Dim (10%)	0.79125	0.86386	0.74473	0.85174
CorEx-SF_Alpha0 150 Dim (30%)	0.80133	0.83276	0.75040	0.84076
CorEx-SF_Alpha0 150 Dim (50%)	0.84004	0.86760	0.77533	0.86039
CorEx-SF_Alpha0 150 Dim (80%)	0.84878	0.89127	0.75409	0.90125
CorEx-SF_AlphaNot0 150 Dim (10%)	0.29688	0.29688	0.29688	0.29688
CorEx-SF_AlphaNot0 150 Dim (30%)	0.29688	0.36917	0.26964	0.27150
CorEx-SF_AlphaNot0 150 Dim (50%)	0.45429	0.61512	0.40664	0.55912
CorEx-SF_AlphaNot0 150 Dim (80%)	0.45639	0.60400	0.42135	0.53432
CorEx-SF_Bidirectional 150 Dim	0.50256	0.65434	0.42956	0.62893
CorEx-SF_MI 150 Dim (10%)	0.79380	0.86649	0.74634	0.87796
CorEx-SF_MI 150 Dim (30%)	0.77256	0.79252	0.74863	0.73644
CorEx-SF_MI 150 Dim (50%)	0.84004	0.86760	0.77533	0.86039
CorEx-SF_MI 150 Dim (80%)	0.84793	0.89127	0.75409	0.90125
CorEx-SFS 150 Dim	0.92167	0.63876	0.71876	0.87768
CorEx-UF 150 Dim	0.86155	0.90005	0.78417	0.92928
CorEx-SF 515 Dim (100%)	0.84250	0.89504	0.79842	0.93757
CorEx-SF_Alpha0 515 Dim (10%)	-0.04453	0.22582	0.09515	0.30615
CorEx-SF_Alpha0 515 Dim (30%)	0.77750	0.79266	0.75803	0.66162
CorEx-SF_Alpha0 515 Dim (50%)	0.75378	0.78750	0.75244	0.67203
CorEx-SF_Alpha0 515 Dim (80%)	0.83627	0.88128	0.78760	0.91131
CorEx-SF_AlphaNot0 515 Dim (10%)	0.19770	0.21625	0.18474	0.15003
CorEx-SF_AlphaNot0 515 Dim (30%)	0.69645	0.72878	0.65379	0.68646
CorEx-SF_AlphaNot0 515 Dim (50%)	0.72628	0.74885	0.64755	0.68934
CorEx-SF_AlphaNot0 515 Dim (80%)	0.77509	0.79000	0.67005	0.75956
CorEx-SF_Bidirectional 515 Dim	0.84007	0.34701	0.79977	0.36260
CorEx-SF_MI 515 Dim (10%)	0.78504	0.80630	0.75401	0.79502
CorEx-SF_MI 515 Dim (30%)	0.79006	0.84795	0.75316	0.73644
CorEx-SF_MI 515 Dim (50%)	0.82380	0.85501	**0.80698**	0.83875
CorEx-SF_MI 515 Dim (80%)	0.83751	0.88001	0.78162	0.90755
CorEx-SFS 515 Dim	**0.93750**	0.00000	0.53340	0.91651
CorEx-UF 515 Dim	0.87625	**0.90626**	0.77376	**0.94255**

For the Gisette dataset, as seen in Table (4.24), the raw data performs better than the features extracted using CorEx-based methods for SVM with Linear Kernel. The accuracy decreases as the dimensions decrease when thresholding is done with CorEx-SF_MI and CorEx-SF_Alpha0 for SVM with Linear Kernel. The Naive Bayes though like the SVM with Linear Kernel, is a linear classifier, it does not perform as well with raw data. After feature extraction with CorEx-UF, CorEx-SF_MI, and CorEx-SF_Alpha0, the Naive Bayes classifier is able to perform better, demonstrating that relationships exists among the different input features. The input features were not conditionally independent of each other given the class label. The decrease in accuracy with the Naive Bayes classifier as the threshold percentage of CorEx-SF decreases could be due to the loss of information since the features captured are independent. CorEx-SF performs marginally better than CorEx-UF. This demonstrates that the class label encoding within CorEx-SF performs as a bias during classification of the Gisette dataset. There is no clear distinction of the number of nodes to choose as different methods perform better with either 150 or 515 nodes. This is because different relations are learnt when different number of nodes are selected to represent the intrinsic structure of the data.

Summary

1. Using features extracted and selected through CorEx-based methods show improvements in the classification performance when compared against the raw data.

2. The ideal number of nodes to be used varies from data-to-data as well as the method used to extract or select the features. This is due to the difference in relationship learnt with different number of nodes. In order to find the optimal number of nodes, random search or grid-like search over the number of hidden nodes may need to be applied.

3. Different feature selection and extraction method may be better for different data.

4.2.3 Effect of Irrelevant Features on Classification

In order to test the effect of the presence of irrelevant features on classifiers, we formulated the CorEx-SF_AlphaNot0 and CorEx-SF_Alpha0. CorEx-SF_AlphaNot0 only extracts features which are considered relevant to the class label whereas the latter

extracts features that are relevant but also includes some irrelevant features as additional noise. We test the premise of the effect of irrelevant features on the accuracy and the generalisation capability of classifiers. Under this, we further test the premise both when the data is HDLSS and when there exists a reasonably large number of sample size for each data generation process.

Some of the insights we hope to gain are :-

1. Would additional irrelevant features help in discriminating between class labels by contributing some higher level interaction?

2. Would the irrelevant features extracted by CorEx-SF cause an averse reaction to the classification task?

Effect of Irrelevant Features on the $Y \rightarrow X \rightarrow C$ Data Generation Process

Under the $Y \rightarrow X \rightarrow C$ data generation process with 80 samples, it can be seen that having a larger threshold for CorEx-SF_AlphaNot0 gives a better accuracy. This is because a larger number of informative features to the class label are being selected. This pattern of improved accuracy is also observed when the threshold of CorEx-SF_Alpha0 is increased. In fact the accuracy increases when CorEx-SF_Alpha0 is used compared to CorEx-SF_AlphaNot0. Taking a closer look at the heatmap, we understand that some of the features that are actually relevant to the class label is deemed irrelevant. Hence, by selecting these additional features through CorEx-SF_Alpha0, features that are relevant is represented. However, when looking through the MCC scores we notice that there exists a trade-off between the number of relevant features captured and the increasing influence of the irrelevant features as the number of features extracted increases. A balance between the both is required to model a robust classifier. Too many irrelevant features causes the robustness of the classifier to reduce.

With 30000 training samples, CorEx-SF_AlphaNot0 and CorEx-SF_Alpha0 performs at par accuracy-wise with each other when the threshold is 10% and 30%, and 10% and 20% respectively. The reason for this is that at these thresholds only features that are strongly connected to the class label is selected. The features selected being the same, will give the same accuracy and MCC score. As the threshold for CorEx-SF_AlphaNot0 increases for KNN classifier, the accuracy also increases. This is to be expected as more features that are relevant to the class label is being extracted as seen through the heatmap in Fig.(4.3). In the case of CorEx-SF_Alpha0, as the threshold increases, the accuracy and the robustness of the classifiers with the exception of KNN remain constant. In the case of KNN, the performance increases, but beyond a cer-

tain threshold of features extracted through CorEx-SF_Alpha0, the classification performance (i.e. accuracy and MCC score) drops although informative features are still being captured. This alludes to the existence of a trade-off between the amount of relevant information captured against the increasing complexity of the classification model as the number of irrelevant information learnt also increases. However, an exact trade-off cannot be determined as different dataset may have different structures and assumptions.

It can be seen from the vantage point of accuracy that the more relevant the features extracted the better the accuracy. However, a trade-off between increasing number of relevant features or information which could, in the case of imperfect extraction, also increase irrelevant information, needs to be managed. Hence, an overpowering presence of irrelevant features will aversely affect the classification in the case of $Y \rightarrow X \rightarrow C$ data generation process.

Effect of Irrelevant Features on the $Y \rightarrow X$, $Y \rightarrow C$ Data Generation Process

The increase in the number of irrelevant features captured does not affect the accuracy nor the robustness of the classifiers. This could be because in both cases (80 samples and 30000 samples) as represented through the heatmap, captures a perfect set of features or information that is relevant. Unlike the case of $Y \rightarrow X \rightarrow C$ data generation process, the data generation assumption is not broken. This would lead to a better estimation of the parameters that influence connections between the hidden nodes and the input nodes. This would also mean that the quality of the data extracted is better which leads to a more stable classification performance regardless of the size of the training set and the amount of irrelevant features. However, although the performance is not aversely affected in this case, it should be noted that the complexity of the data increases as the number of irrelevant features are extracted. We speculate that after a certain threshold, albeit much higher, the stability of the classifier will degrade as the quality of the data and relationships extracted decreases.

Effect of Irrelevant Features on Real World Data

The real world datasets (i.e. Alon, Golub, Gisette, and Gravier) shows that capturing irrelevant features are helpful for classification performance. This could be due to some relevant features being captured as irrelevant to the class label as seen with the $Y \rightarrow X \rightarrow C$ data generation process. However, the possibility of the data having features which are generated through both the process cannot be eliminated. Methods to deal with such hybrid data generation process remains an open problem.

A question which arises from this is whether CorEx-UF would be better than CorEx-SF-based methods as features that are considered irrelevant may be related to the class label and improve the classification performance. Upon further investigation, we discover that CorEx-SF based methods generally gives a better classification performance than CorEx-UF. A comparison between CorEx-SF with a threshold of 100%, i.e. all the features are extracted when compared to CorEx-UF shows CorEx-SF being able to perform better in some cases. This shows that encapsulating the relationship between the class label and the input features can help to create a more discriminant feature set. A reason for this is that hidden feature Y explains as much of the input feature X. Hence, by having the class label as part of the input features, CorEx-C tries to learn as much of the higher-order relationship that exists. The structure and relationship learnt through CorEx-C model is different from that of the CorEx model. This is validated by the classification performance of features extracted from both these models.

Summary

1. In general, if the assumptions are not broken, the transformed features extracted are consistent regardless of the additional irrelevant features that contribute to the extraction.

2. In both cases ($Y \rightarrow X$, $Y \rightarrow C$ and $Y \rightarrow X \rightarrow C$), the stronger the influence of the relevant features, the better the classification performance.

3. In the case that the assumptions are broken, a trade-off between capturing the relevant information and the contribution from irrelevant features should be balanced in order to attain the best classification performance.

4.2.4 Case Study of Feature Selection and Extraction Methods based on Relevance of Features for Classification, Redundancy of Features, and Presence of Uninformative Features

The methods we formulated for feature extraction and selection can be broadly categorised into four groups:

1. Case 1: Methods for which the features selected and extracted are relevant, have minimal redundancy, but contains uninformative features. The algorithms which fall under this category are CorEx-SF_Bidirectional, CorEx-SF_Alpha0,

and CorEx-SF_MI. These methods capture the relevant features by causing a bias in the connection using the class label.

2. Case 2: Methods for which features selected and extracted are relevant, with minimal redundancy and no uninformative features extracted. CorEx-SF_AlphNot0 is an algorithm that fall under this category. It extracts the features from the transformed feature space that are considered to have a relationship with the class label.

3. Case 3: Methods which perform pure compression of the data, i.e. reduction of redundancy but not necessarily selection or extraction of relevant features. This category of method ensures that there is minimal information loss while reducing the redundancy. Methods that fall into this category are generally unsupervised methods. A method that falls under this category is CorEx-UF.

4. Case 4: Methods which extract only relevant features, minimises selection of uninformative features, but may consist redundant features. CorEx-SFS falls in this category as it only selects features that are considered relevant to the class label. However, it does not remove redundant features.

Throughout this section, we will denote features extracted or selected by methods in a particular case being competitive when classification is applied as, e.g. "Case 1 is competitive with Case 2".

A Comparison of Classification Performance between Methods within Case 1

Ensemble method was used to compare between the different feature extraction methods. The best classification performance and the threshold parameter associated to each of feature extraction method for the data were used to compare between the methods for each classifier.

Under the $Y \rightarrow X, Y \rightarrow C$ data generation process with 80 samples, all methods under the category produces the same accuracy and MCC score. The reason for this phenomenon can be justified as follows: CorEx-SF_Bidirectional connects the strongest node, then remaps each features to the node that has the largest α value. From the heatmap, it is observed that it selects the same node as CorEx-SF_AlphaNot0. This is because the hidden node connected to the class label perfectly captures most of the features which are related to the class label. CorEx-SF_MI also performs the same as the other methods because only one node actually captures the class label. So it just dilutes the influence from the other nodes. Hence, they perform similarly. The

additional nodes which are not relevant to the class label which are extracted by CorEx-SF_Alpha0 as the threshold size is increased does not have an impact on the classification performance under this data generation process. Hence, its performance matches that of the other extraction methods in this category. As the sample size is increased to 30000 for the $Y \rightarrow X, Y \rightarrow C$ data generation process, the accuracy improves but the ranking of the performance remains the same between the different methods.

Under the $Y \rightarrow X \rightarrow C$ data generation process, with 80 samples, CorEx-SF_Alpha0 extracted features which gave a better classification performance than CorEx-SF_MI. The reason is that many of the relevant features are captured as weakly related while others are not considered to be related. As CorEx-SF_MI dilutes the weak connection and exaggerates the strong connections, the features that are actually strongly connected but which are not related to the class label becomes more dominant than the informative features which are considered weakly related. This is also evidenced as the threshold size increases, the performance of the classifiers using the extracted features start improving. CorEx-SF_Alpha0 and CorEx-SF_MI both show increasing robustness as the threshold increases. This is due to fact that more relevant features are captured as threshold size increases. CorEx-SF_Bidirectional gives a weaker generalisation of the classifiers than CorEx-SF_Alpha0. A reason for this is that if the data is at least weakly captured as being relevant, CorEx-SF_Bidirectional will remap through the connections to select the node that represents the corresponding input features the best. This would also mean that the node that is considered strongest connected to the class label may be discarded as the input features that are considered related to the class label may have a stronger relationship with other hidden nodes which are not considered related to the class label. This may cause loss of information which reduces the performance of CorEx-SF_Bidirectional. With 30000 samples, under the $Y \rightarrow X \rightarrow C$ data generation process, the performance of the classifiers show an improvement using the features selected and extracted compared to when only 80 samples are available. Unlike the case of 80 samples, CorEx-SF_Bidirectional performs competitively with CorEx-SF_Alpha0 with 30000 samples.

Hence, in conclusion, under the $Y \rightarrow X, Y \rightarrow C$ data generation process, the classification performance for this category of methods are similar. In the case of $Y \rightarrow X \rightarrow C$, CorEx_Alpha0 performs competitively if not better than CorEx-SF_Bidirectional and CorEx_MI.

Effect of Sample Size and Different CorEx based Feature Extraction and Selection Methods on the Classifiers

Both sample size and the process of extraction can have different effects on different classifiers. This is because different classifiers have different assumptions and criteria for classification. From this study, we hope to understand the factors that effect the classifiers in terms of redundancy, presence of uninformative features, and relevance of features to class labels.

We investigate the case for SVM with RBF Kernel and KNN which are non-linear methods. SVM with Linear Kernel, although is a non-linear method, performs linear classification. Naive Bayes is also a linear classifier.

We created an ensemble of feature selection and extraction methods for each case. The feature selection or extraction methods in a case is combined. The ensemble chooses the feature set from the output of the feature selection or extraction method which gives the best performance as the final output in a particular case. The classification performance of the feature set chosen for each case, is used as the basis of the discussion in this section. Case 1 denotes the ensemble of the feature selection and extraction methods that fall under Case 1. The same is true for the other cases.

SVM:RBF Kernel

The classification performance of $Y \rightarrow X, Y \rightarrow C$ data generation process with 80 samples is the same with the algorithms in Case 1 and Case 2. The reason is there is a common subset of features which are relevant to the class labels that are extracted. Case 4 performs poorly although like its feature extraction counterpart, it captures relevant relationships between the features and the class label. A reason for this would be that since the dataset is actually linearly separable, the lack of a direct class encoding as existing with CorEx-SF based methods makes it harder for the RBF Kernel to separate the data. Unlike Case 1 and Case 2, Case 4 does not reduce the redundancy among the features. This could increase the complexity of the data to be modelled by the classifier thereby contributing to its weaker performance. Case 3 has a performance that is worse than Case 1 and Case 2 MCC-wise. This could be because there is a difference in the connections learnt between the nodes compared to CorEx-C based methods which would change the encoding of data within the nodes. As the sample size increases, the performance of the classifier also improved. The feature extraction methods under Case 1 and Case 2 perform comparatively, while Case 4 has worse MCC score than Case 1 and Case 2. Case 3 has a better generalisation capability than Case 4.

In the case of $Y \rightarrow X \rightarrow C$ data generation process with 80 samples, the accuracy of Case 1 and Case 4 is better than Case 3, which is better than Case 2. Taking a closer

look at the MCC scores, we see that the method in Case 3 and Case 4 are overfitting while Case 1 and Case 2 performs better. The MCC scores also show that both Case 1 and Case 2 perform comparatively with each other. As the sample size increases, the performance of Case 1 and Case 2 converges to the ranking of methods as seen with the $Y \rightarrow X, Y \rightarrow C$ data generation process when there were 80 samples. This shows that in the event of abundance of data and even if the data generation process is violated, there is a potential for the methods to work similar to when under the HDLSS case without violation of the data generation assumption.

Hence, in general, we find that methods which minimises redundancy performs better for SVM with RBF Kernel. The relevance to class label allows better performance when the data assumptions made by CorEx-C is not violated.

SVM:Linear Kernel

The classification performance for $Y \rightarrow X, Y \rightarrow C$ data generation process with 80 samples is similar for Case 1 and Case 2. The additional encoding of the class labels learnt through the connections of CorEx-C allows the classifier to be equally discriminant. Case 4 does not perform as well as Case 1 and Case 2. Features extracted through Case 3 performs slightly worse than features extracted through Case 1 and 2. However, it can still be considered as being competitive. A reason could be because the linear separability of the data allows for this classifier to perform reasonably well even when the data is not encoded with the class labels. As the sample size increased to 30000, Case 1, Case 2, and Case 3 perform similarly. This is because there is sufficient samples to accurately model the algorithm. Case 4 performs worse than Case 1, Case 2, and Case 3.

In the case of $Y \rightarrow X \rightarrow C$ data generation process with 80 samples, Case 1 performs better than Case 2 and Case 4. However, it should be noted that under both Case 1 and Case 2, some of the methods (CorEx-SF_Alpha0 and CorEx-SF_AlphaNot0) performs similarly when the threshold size is small. This is because the same features are being extracted. Case 3 performs badly compared to Case 1 and Case 2 as there is no biasing factor. Case 4 performs worse than Case 1, Case 2, and Case 3 MCC-wise possibly due to the presence of redundant features. The redundant features can result in the number of features being selected to be of higher dimension than when the features are extracted. As the sample size increases, the disparity between Case 1 and Case 2 decreases. This is because, as with SVM with RBF Kernel, the stability of the model increases when the sample size increases. Case 3 shows improvement when sample size increased but gives a weaker performance than Case 4. Case 4 performs better than the other methods as the sample size increased. Case 4 selects whole features while Case 1

and 2 extracts only information from features that are considered relevant to the class label. However, due to violation of the assumption made by the data generation process, not all relevant information may be extracted. The effect of redundant features also diminishes as the sample size increases as seen with the performance of Case 4.

Hence, in general, we find that methods that maximises relevance to class label and reduces redundancy performs better than those which do not have either of the criteria. However, as the sample size increases the effect of redundant features diminishes. This is seen as Case 4 improves its performance and becomes competitive if not better than the rest of the methods.

KNN

In the case of $Y \rightarrow X, Y \rightarrow C$ data generation process with 80 samples, Case 1 and Case 2 have similar accuracy while Case 3 gives a better accuracy. Taking a closer look at the MCC score, it can be seen that Case 3 has very poor generalisation capabilities in comparison with all the other methods. The accuracy of Case 3 shows that the model has learnt some inherent structure that is causing it to overfit on the training set. As the sample size increases, the classification performance of Case 3 improves whereas Case 1 and Case 2 does not show much change in the MCC score. However, Case 1 and Case 2 still performs better than Case 3. One reason for this could be that the relationships discovered with lower samples are close to the ideal set of features as it is the case with the larger sample size. Case 4 performs poorly compared to the other methods even though it shows improvement in classification performance when abundant samples are available.

In the case of $Y \rightarrow X \rightarrow C$ data generation process with 80 samples, as the threshold size increases, the performance improves for Case 1 and Case 2 until a certain threshold before dropping in performance. The reason for this is that the relevant relationships are increasingly captured as the threshold size increases. After a certain threshold, there starts becoming too many irrelevant information which could skew the classifier away from the class label, hence reducing the classification performance. Case 4 performs worse than Case 1 and Case 2. Case 3 performs better than Case 2 accuracy-wise but reduces the generalisation capability of the classifier. As the sample size increases, the robustness of the classifiers also improves. However, Case 1, Case 2, and Case 4 produces a more robust classifier than Case 3 when the sample size is larger. This shows that the class encoding acts as an added weighting to the features to discriminate between samples from different classes when KNN is used.

Hence, in general, we find that methods that reduces redundancy performs better than those which do not reduce the redundancy of the data. The reduction of redun-

dancy is very important for KNN as it reduces the complexity of the data. We postulate that the features slowly move away from being equidistant from each other when the redundancy decreases provided the samples are sampled independently from a distribution.

Naive Bayes

Case 1 and Case 2 performs equally well for $Y \rightarrow X, Y \rightarrow C$ data generation process with 80 samples. Case 3 and Case 4 shows a better accuracy but lower robustness of the classifier compared to Case 1 and 2. A reason why Case 1 and Case 2 could be similar for Naive Bayes could be that if x_i is not related to c, then the $P(x_i|c)$ value would be small. This reduces the influence of the node. Hence, the selection or extraction of the features which are not related to the class label does not make much of a difference. As the sample size increased, the robustness of Case 3 improves to the level of the methods in Case 1 and Case 2. A reason for this could be because there are sufficient samples of the features extracted to correct the bias generated by the irrelevant features that existed with lower sample sizes. Case 4 on the other hand performs better than all the others when the sample size increased.

In the case of $Y \rightarrow X \rightarrow C$ data generation process with 80 samples, the accuracy shows that Case 3 is better than Case 1 and 2. Taking a closer look at the MCC score, it showcases that using features extracted through Case 3 gives a worse generalisation of the classifier. Insufficient samples could cause the classifier to learn a biased model when no class encoding is captured. As the sample size increased, Case 2 performs similarly with Case 1. Case 4 performs better than Case 1 and Case 2. It is possibly because there is less information loss when whole features are selected whereas Case 1 and Case 2 only captures partial relationships as being relevant to the class label. Case 3 has lower robustness than Case 1 and Case 2. Naive Bayes only strongly biases the features that are related to the class label, which is captured by Case 1 and Case 2.

Hence, in general, we find that reduction of redundancy is a very important factor along with relevance to class label for improvement of the performance of the classification ability of Naive Bayes. However, when the sample size increases, the effect of redundancy on the Naive Bayes classifier reduces.

Summary

For the different classifiers, regardless of the data generation process, Case 1 and Case 2 seem to perform slightly better than the other cases. This shows that supervised methods may be able to detect relevant relationships better than unsupervised methods especially under HDLSS conditions. Case 4 has the ability to perform well under the $Y \rightarrow X \rightarrow C$ condition when the number of samples are large. The presence of re-

dundant features do not improve the performance of the classifiers but may add to the complexity of the data being modelled. This in turn can cause the classifiers to perform worse. Reducing redundancy seem to be a key factor for improving classification performance under HDLSS conditions. On the other hand, too many uninformative features can bias the model away from features that are important for classification. This is seen with Case 1 as the threshold size increases. However, when the data generation assumption is violated, some relevant features may be considered uninformative. In such a case, the features can be captured by Case 1 and help to improve the performance of the classifiers.

Benchmarking the Cases on Real World Dataset

The cases perform differently on different datasets. Different classifiers also have performances based on the number of nodes defined. This is to be expected as the different datasets have different structures.

1. Golub Dataset: Generally, Golub dataset works better with Case 1 or Case 4 compared to Case 3. It works worst with Case 2. The reason for this could be due to some information being relevant but not captured as being related to the class label. Case 2 purely captures the relevant relationships to be classified. Case 4, although it captures only the relevant relationship, due to it selecting the original features, any noise that is present is captured and will also help to improve the robustness of the classifier. However, a high noise content in a feature can have an averse effect on classification performance.

2. Alon Dataset: Although Case 3 is able to perform well with some of the classifiers, it can be seen through the MCC score that CorEx-C based methods are more robust with the exception of Case 2.

3. Gravier Dataset: Generally, Case 1 performs the best for Gravier dataset while Case 2 and Case 4 performs badly. Although all 3 cases includes relevant features, it can be seen that having some uninformative features can help to improve the robustness of the classifiers. Case 3 can perform competitively with Case 1. It also contains uninformative features for the classification task.

4. Gisette Dataset: Case 1 performs the best. Case 2 on the other hand performs the worst. While having relevant features are important, it is equally important to have some uninformative features in the dataset. This allows for variations to be learnt. The variations help the classifiers to be more robust.

In general, the relevance of the features selected or extracted to the class label is important. However, some variations of the data that are introduced through uninformative information can also help the classification performance. Another reason for the performance of Case 2 to be bad with real world datasets could be due to the datasets having a mix of data generation process. Dealing with mix of data generation process is a future direction which should be explored to further improve the CorEx-C structure. Another reason for the varied performance could be because the number of hidden nodes defined is not optimal for each of the datasets. Selecting the optimal number of hidden nodes is an open problem that is not addressed in this thesis.

4.2.5 A Comparison of Different Information Theory-based Feature Selection and Extraction Methods

CIFE, KECA, MRMR, MMI, and ICA are different information-theory based feature extraction and selection methods that were used to benchmark against CorEx-based feature selection and extraction methods. The information-theory based feature extraction and selection methods were also used to test how well it works with HDLSS datasets. The methods can be categorised into four categories based on the general framework formulation discussed in Chapter 2. The four common categories are methods which minimises redundancy and maximises relevance, methods which maximises relevance, methods which minimises redundancy, and methods which minimises redundancy and information loss. The first two categories consist of methods which are supervised while the latter two consist of methods which are unsupervised. Through these, we wish to make a comparison of methods within each groups and also between the groups. We also investigate the differences in performance between supervised and unsupervised methods.

The method which fall under the categories are:

1. Case 1: This category consists of methods that minimises redundancy and maximises class relevance. The methods include CIFE, MRMR, and CorEx-SF based methods.

2. Case 2: This category consists of methods that maximises class relevance. Unlike Case 1, it does not consider reducing the redundancy in the dataset. MMI and CorEx-SFS fall in this category.

3. Case 3: This category reduces the redundancy without giving importance to the relevance of the features. ICA falls in this category.

4. Case 4: This category minimises redundancy and loss of information. The methods include KECA and CorEx-UF. Both these methods contrasts the information captured before and after extraction.

We investigate the effect of sample size as well as data generation process on the methods within each case and between the cases.

Table 4.26: A table of comparison of accuracy of feature extraction and selection methods for Golub dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	88.00%	**48.00%**	88.00%	84.00%
CIFE 150 Dim	84.00%	**48.00%**	64.00%	72.00%
KECA	48.00%	**48.00%**	52.00%	60.00%
MMI	28.00%	32.00%	28.00%	40.00%
InfoMax ICA	72.00%	**48.00%**	48.00%	80.00%
MRMR 515 Dim	88.00%	**48.00%**	88.00%	76.00%
MRMR150 Dim	76.00%	**48.00%**	84.00%	84.00%
CorEx-SFS 515 Dim	96.00%	**48.00%**	**96.00%**	**96.00%**
CorEx-SFS 150 Dim	**100.00%**	**48.00%**	**96.00%**	**96.00%**
Raw Data	96.00%	**48.00%**	**96.00%**	**96.00%**

Table 4.27: A table of comparison of MCC of feature extraction and selection methods for Golub dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN (k=5)
CIFE 515 Dim	0.7613	**0.0000**	0.7845	0.7140
CIFE 150 Dim	0.6795	**0.0000**	0.2774	0.4576
KECA	0.0000	**0.0000**	0.0385	0.1987
MMI	-0.4387	-0.3974	-0.4387	-0.1961
InfoMax ICA	0.5204	**0.0000**	0.0000	0.6210
MRMR 515 Dim	0.7613	**0.0000**	0.7806	0.5290
MRMR150 Dim	0.5230	**0.0000**	0.6864	0.6795
CorEx-SFS 515 Dim	0.9226	**0.0000**	**0.9231**	0.9226
CorEx-SFS 150 Dim	**1.0000**	**0.0000**	0.9226	**0.9231**
Raw Data	0.9231	**0.0000**	**0.9231**	**0.9231**

Table 4.28: The table shows the accuracy of feature extraction and selection methods for Alon dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	**88.89%**	50.00%	66.67%	72.22%
CIFE 150 Dim	72.22%	50.00%	66.67%	**77.78%**
KECA	72.22%	**72.22%**	**94.44%**	66.67%
MMI	33.33%	38.89%	27.78%	38.89%
InfoMax ICA	**88.89%**	50.00%	50.00%	72.22%
MRMR 515 Dim	**88.89%**	50.00%	66.67%	72.22%
MRMR 150 Dim	66.67%	50.00%	66.67%	**77.78%**
CorEx-SFS 515 Dim	**88.89%**	50.00%	66.67%	66.67%
CorEx-SFS 150 Dim	77.78%	50.00%	61.11%	**77.78%**
Raw Data	**88.89%**	50.00%	66.67%	66.67%

Table 4.29: The table shows the comparison of MCC values for feature extraction and selection methods for Alon dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	**0.7778**	0.0000	0.4472	0.5345
CIFE 150 Dim	0.4472	0.0000	0.7071	**0.5698**
KECA	0.5345	**0.5345**	**0.8944**	0.3721
MMI	-0.3721	-0.2673	-0.4472	-0.2357
InfoMax ICA	**0.7778**	0.0000	0.0000	0.4714
MRMR 515 Dim	**0.7778**	0.0000	0.4472	0.5345
MRMR 150 Dim	0.3333	0.0000	0.4472	**0.5698**
CorEx-SFS 515 Dim	**0.7778**	0.0000	0.4472	0.4472
CorEx-SFS 150 Dim	0.5556	0.0000	0.2673	0.5556
Raw Data	**0.7778**	0.0000	0.4472	0.4472

Table 4.30: A table of comparison of accuracy of feature extraction and selection methods for Gravier dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	65.96%	40.42%	76.60%	61.70%
CIFE 150 Dim	55.32%	40.43%	70.21%	59.57%
KECA	**70.21%**	40.43%	55.32%	**72.34%**
MMI	29.79%	40.43%	34.42%	40.43%
InfoMax ICA	55.32%	40.43%	59.57%	57.45%
MRMR 515 Dim	65.96%	40.43%	**78.72%**	59.57%
MRMR 150 Dim	57.45%	40.43%	76.60%	61.70%
CorEx-SFS 515 Dim	61.70%	**59.57%**	61.70%	59.57%
CorEx-SFS 150 Dim	57.45%	46.81%	55.32%	59.57%
Raw Data	**70.21%**	40.43%	76.60%	61.70%

Table 4.31: A table of comparison of MCC of feature extraction and selection methods for Gravier dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	0.2731	0.0000	0.5306	**0.1790**
CIFE 150 Dim	-0.0270	0.0000	0.3934	0.0000
KECA	0.3661	0.0000	-0.1737	0.4213
MMI	-0.3724	0.0000	-0.2602	-0.0411
InfoMax ICA	0.1299	0.0000	0.0000	-0.1215
MRMR 515 Dim	0.0722	0.0000	0.5306	**0.1790**
MRMR 150 Dim	0.2932	0.0000	**0.5813**	0.0000
CorEx-SFS 515 Dim	0.1590	0.1142	0.1590	0.1014
CorEx-SFS 150 Dim	0.1018	**0.1436**	0.1299	0.1402
Raw Data	**0.3724**	0.0000	0.5306	**0.1790**

Table 4.32: A table of comparison of accuracy of feature extraction and selection methods for Gisette dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	89.44%	50.00%	85.56%	92.25%
CIFE 150 Dim	88.25%	50.00%	86.56%	90.31%
KECA 515 Dim	56.63%	56.63%	85.25%	94.88%
KECA 150 Dim	56.63%	56.63%	83.88%	95.13%
MMI 515 Dim	53.44%	53.81%	54.25%	49.19%
MMI 150 Dim	50.06%	50.50%	52.00%	50.63%
InfoMax ICA 515 Dim	95.38%	50.00%	77.81%	89.69%
InfoMax ICA 150 Dim	94.63%	50.00%	**91.88%**	95.44%
MRMR 515 Dim	93.44%	50.00%	81.13%	95.63%
MRMR 150 Dim	91.93%	51.00%	80.50%	95.13%
CorEx-SFS 515 Dim	**96.88%**	50.00%	74.06%	**95.81%**
CorEx-SFS 150 Dim	92.31%	**79.69%**	85.94%	93.88%
Raw Data	**96.88%**	50.00%	74.06%	**95.81%**

Table 4.33: A table of comparison of MCC of feature extraction and selection methods for Gisette dataset.

Classifier	SVM:Linear	SVM: RBF	Naive Bayes	KNN(k=5)
CIFE 515 Dim	0.7888	0.0000	0.7139	0.8451
CIFE 150 Dim	0.7652	0.0000	0.7371	0.8063
KECA 515 Dim	0.2128	0.2128	0.7267	0.8976
KECA 150 Dim	0.2128	0.2128	0.7002	0.9026
MMI 515 Dim	0.0689	0.0786	0.0851	-0.0163
MMI 150 Dim	0.0013	0.0217	0.0436	0.0125
InfoMax ICA 515 Dim	0.9077	0.0000	0.5991	0.7943
InfoMax ICA 150 Dim	0.8941	0.0000	**0.8461**	0.9090
MRMR 515 Dim	0.8688	0.0000	0.6619	0.9128
MRMR 150 Dim	0.8388	0.0900	0.6480	0.9025
CorEx-SFS 515 Dim	**0.9375**	0.0000	0.5333	**0.9165**
CorEx-SFS 150 Dim	0.8463	**0.6388**	0.7188	0.8777
Raw Data	**0.9375**	0.0000	0.5334	**0.9165**

Effect of Data Generation Process and Sample Size on Case 1

The comparison for CorEx-SF based methods are done using an ensamble method whereby the method with the best classification performance is used to compare with the other methods.

Under the $Y \rightarrow X, Y \rightarrow C$ data generation process with 80 samples, MRMR and CIFE performed comparatively with each other while CorEx-SF performs better. However, it should be noted that in the case of KNN, MRMR and CIFE are more robust although the accuracy produced is lower than CorEx-SF. As the sample size increased to 30000, features extracted by MRMR and CIFE gives better classification performance than when the sample size is small. CorEx-SF does not show any change in performance. The features extracted by CorEx-SF performs competitively with CIFE and MRMR.

Under the $Y \rightarrow X \rightarrow C$ data generation process with 80 samples on the other hand shows that features extracted through CorEx-SF performs poorly with the classifiers while MRMR and CIFE performs better. As the sample size increases, the performances of all three algorithms improve. MRMR and CIFE perform comparatively with each other while still being better than CorEx-SF methods. CorEx-SF based methods do not perform well under this data generation process due to violation of data assumption.

Effect of Data Generation Process and Sample Size on Case 2

Under the $Y \rightarrow X, Y \rightarrow C$ process with 80 samples, CorEx-SFS performs better than MMI. MMI does not perform well as Parzen Window degenerates when there are insufficient samples. In order for the density estimate by the Parzen Window kernel to be accurate, the region in which the samples fall under should be relatively small. With the Parzen Window function, each sample contributes to the density estimation if it lies close enough to the resulting density at a particular point. However, the contribution to the density remains constant if the samples fall in the same hypercube. This is the case for HDLSS data in the Euclidean space, which would explain the degeneration of the Parzen Window density estimation method. Parzen window used with MMI uses Gaussian kernels. The resulting density will not be smooth and will include discontinuities. This causes problems because values which falls in-between ranges will not be accurately represented using the density estimation method.

As the sample size increases, the features extracted with MMI improves the classification performance. CorEx-SFS performs better than MMI even as the sample size increases. However in the case of SVM with RBF Kernel, CorEx-SFS performs worse

than MMI. One reason could be the due to the data being linearly separable and so the need to extrapolate it to higher dimensions would not be there as relevant features are extracted. MMI although better than CorEx-SFS for SVM with RBF Kernel, does not perform well either.

Under the $Y \rightarrow X \rightarrow C$ process with 80 samples, CorEx-SFS and MMI both perform competitively. However, the performance of both CorEx-SFS and MMI is worse under the data generation process of $Y \rightarrow X \rightarrow C$ compared to the data generation process of $Y \rightarrow X, Y \rightarrow C$. As the sample size grows, CorEx-SFS shows improvement in classification performance while MMI does not. Hence, for both these algorithms, they work better with data generated through the $Y \rightarrow X, Y \rightarrow C$ data generation process.

Effect of Data Generation Process and Sample Size on Case 3

Under the $Y \rightarrow X, Y \rightarrow C$ data generation process, the performance of the of the classifiers improved with the exception of Naive Bayes and SVM with Linear Kernel when the sample size increased for ICA. This could be because of the loss of linearity of the data after extraction was done.

Under the $Y \rightarrow X \rightarrow C$ data generation process, the performance is comparative for all the classifiers regardless of the sample size. This could be because it searches for the original source. Minimal change in performance will be seen if the data is representative of the sources even as the sample size increases.

Effect of Data Generation Process and Sample Size on Case 4

Under the $Y \rightarrow X, Y \rightarrow C$ data generation process, with 80 samples, CorEx-UF is more robust than KECA with SVM with Linear Kernel and SVM with RBF Kernel. As the sample size increased the performance of the classifiers improved for both CorEx-UF and KECA. In both cases, the performance is competitive with each other.

Under the $Y \rightarrow X \rightarrow C$ data generation process, with 80 samples, KECA performs competitively with CorEx-UF. Although there is no direct violation of the data generation assumption, the manner in which the information is extracted matters to the classifiers. This is seen as the information captured by CorEx-UF and KECA under both data generation processes will be the same as no class label is explicitly involved. However, their performance differs from that of $Y \rightarrow X, Y \rightarrow C$ depending on the generation of the class label. This shows that including the class label during pre-processing helps to identify structure of underlying data in relation to the class label. As the sample size increases, both show improvement in classification performance. However, CorEx-UF produces a more robust classifier.

Discussion

In general, we observed that supervised feature extraction and selection methods give a better performance compared to unsupervised feature extraction methods. The additional information provided by class label helps discriminate between features that are relevant and irrelevant to classification. This reduces bias from the other features, allowing classifiers to produce a more accurate model of the data.

Real world datasets on the other hand demonstrated the following patterns among the methods within Case 1,2, and 4:

1. Case 1: In the case of real world datasets that were tested, CorEx-SF performed better than MRMR and CIFE for all the cases except for Alon. For the Alon dataset, when 515 hidden nodes were used with CorEx-C, MRMR and CIFE performed better. When 150 nodes were used, CorEx-SF based methods performed competitively with CIFE and MRMR. As can be seen, CorEx-SF based methods generally perform better than CIFE and MRMR.

2. Case 2: In the case of the real world datasets, it can be seen that CorEx-SFS performs better than MMI. One of the reason could be because Parzen Window that is used for density estimation in MMI degenerates under HDLSS conditions. CorEx-SFS also has the added advantage of being a feature selection method whereby information from a feature is fully captured, whereas a feature extraction method, there is a possibility of information loss with MMI due to only partial information being extracted and considered as relevant to the task at hand.

3. Case 4: For real world datasets, CorEx-UF generally performed better than KECA. This is with the exception of Alon dataset with the Naive Bayes classifier. Different data is structured differently and will have different methods that will work better. This is the case for all algorithms.

In order to compare between categories, we created an ensemble with the methods for each of the cases. The output for each of the ensemble is based on the classification performance of the methods in each case for the data. The output from each of the ensemble are compared with each other. The different categories show that Case 1 performs better than the other cases regardless of the sample size. This shows that reducing redundancy and ensuring class-based relevance produces the best results. This is followed by methods which ensure relevance although redundancy still exists within the dataset. However, it should be noted that MMI doesn't conform to the pattern. We also observed that Case 3 tends to be better than Case 4. This could be because ICA

tries to look for the original sources, while Case 4 tries to look for a lower compression which may not be related to the original structure of the data.

In conclusion, it can be seen that KECA, ICA, MRMR, CIFE, and CorEx-based methods have the ability to deal with HDLSS data. However, KECA and ICA have a limitation in the number of features that can be extracted. They are only able to extract n features, where $n \in N$ and N is the number of samples available. The number of recoverable sources in conventional ICA is generally bounded by the number of samples available. KECA on the other hand is not able to select more than n features due to the use of the kernel which transforms the data into an $N \times N$ space representing the inner product space (Gram matrix). CorEx-SFS is more flexible in the sense that the number of features that can be selected is limited only by the number of the original dimensions. CorEx-UF and CorEx-SF on the other hand is only upper bounded by the number of latent variables defined. However, the maximum number of hidden nodes which can be defined is bounded by the number of input features.

4.2.6 Benchmarking CorEx based Feature Selection and Extraction Methods for Real World Data against Results from Literature

In order to further benchmark the CorEx based methods, comparisons between the accuracy of the results obtained and the results of the datasets in literature was made. The comparisons are as follows:

1. Golub Dataset:

   According to Golub *et al.* (1999), it was also found that the informative genes selected between the range of 10 to 200 was found to give an accuracy of 100% when majority voting by the genes were used for classification. This shows that the genes in Golub are able to discriminate against the classes well which is also reflected in the results we have obtained as observed by CorEx-SFS.

2. Alon Dataset:

   In Alladi *et al.* (2008), 50 genes were selected using the t-statistics for classification using SVM with a variety of kernels and achieved an accuracy between the range of 60% to 83%. Our results show that CorEx-SF_MI and CorEx-SF_Alpha0 with 150 node at 10% thresholding is able to outperform the results obtained while CorEx-SF_MI and CorEx-SF_Alpha0 with 150 nodes with other threshold values perform competitively with the literature when SVM with Linear and RBF Kernel were used.

3. Gravier Dataset:

   In the literature, Gravier dataset's accuracy using SVM with Linear Kernel selected using heat map (Huertas and Juarez-Ramirez, 2016) is competitive with CorEx-UF and CorEx-SF_Alpha0 with threshold 80% for both 515 and 150 nodes.

4. Gisette Dataset:

   The standard Gisette dataset with the original 13500 samples attains an accuracy in the range of 95%-97% when SVM with Linear Kernel is used (Eskandari and Akbas, 2017). CorEx-UF trained using only 2400 samples was still able to competitively attain an accuracy of 93.81%. In Deng *et al.* (2016), using Landmark Spectral Clustering (LSC) on the standard Gisette, an accuracy of 95% was obtained. Classification using KNN after the down sized Gisette was run with CorEx-UF gave an accuracy of 97.13%. This shows that CorEx is competitive with the methods available in the literature.

Through our experiments and results obtained from literature, we observe that CorEx based methods are able to select or extract features which can give a competitive if not better accuracy for classification.

4.3 Generative Classification Method

This section describes the results and analysis of generative classification method for HDLSS data. The inferencing formulated for CorEx-C is compared and benchmarked against other classification methods. The datasets were split into 60% training, 40% testing data for all the datasets except the Golub dataset which was split into 50% training and 50% testing. Random grid search was used with the real world data as knowledge on the actual number of intrinsic dimension was not known. The experiments were held to test how well CorEx-C coupled with the posterior estimation would work as a generative model for classification of HDLSS data.

4.3.1 Benchmarking CorEx-Ci with other Classification Methods for Simulated Data

Table 4.34: Accuracy of the simulated data using different machine learning algorithms.

Dataset	SVM Linear	SVM RBF	KNN	Naive Bayes	CorEx-Ci
80_YC	51.32%	35.51%	51.95%	**59.74%**	58.21%
80_YXC	52.56%	**64.51%**	51.37%	59.74%	56.27%
30000_YC	53.66%	**61.06%**	50.04%	59.04%	58.51%
30000_YXC	**88.26%**	64.51%	50.08%	67.72%	60.62%

Table 4.35: The table shows the Matthew's Correlation Explanation values for the simulated data for the different machine learning algorithms.

Dataset	SVM Linear	SVM RBF	KNN	Naive Bayes	CorEx-Ci
80_YC	0.022866	0.010491	0.000196	**0.147491**	0.056300
80_YXC	0.013865	0.000000	0.009644	**0.211711**	0.043639
30000_YC	0.130116	-0.140109	0.011994	**0.182718**	0.070699
30000_YXC	**0.755395**	0.000000	0.003326	0.418229	0.047342

As seen from tables (4.34 and 4.35), we show that under small sample size, CorEx-Ci performs comparably. It also produces a more robust classifier compared to SVM with Linear Kernel and RBF Kernel when the sample size is small. As the sample size increased, the robustness of the classifiers also increased with the exception of KNN. It could be possible that there were samples which were less discriminative as the sample size increased. Naive Bayes performs the best on the smaller sample sizes compared to the other classification algorithms. The decoupling of the features conditioned on the class means that each distribution is calculated as a one dimensional distribution. This helps to alleviate the curse of dimensionality. It also means that less samples are required to estimate the parameters of the Naive Bayes classifier.

4.3.2 Benchmarking CorEx-Ci with other Classification Methods for Real World Data

The datasets used in this work are Alon's Colon Cancer, Gravier's Breast Cancer, and Golub's Leukemia. The datasets were balanced by randomly down-sampling the

majority class to match the number of samples in the minority class. We also used Sonar dataset, Pima Indians Diabetes, and Wisconsin Breast Cancer (Diagnostic).

Pima Indians Diabetes

The Pima Indians Diabetes dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The dataset is used to predict whether patients have diabetes. This dataset can be obtained from the Kaggle website (Rossi and Ahmed, 2015). The dataset has a mixture of discrete features and continuous features. The PIMA dataset also consists of missing values, which we have imputed using the mean value.

Sonar

The Sonar dataset predicts whether an object is a rock or a mine. It does so by the strength of the sonar returned from various angles. This dataset is obtained from the UCI repository (Dua and Graff, 2017).

Wisconsin Breast Cancer (Diagnostic)

The Wisconsin Breast Cancer (Diagnostic) (WBCD) is used to predict if a growth in benign or malignant. The features were computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. This dataset is obtained from the UCI repository (Dua and Graff, 2017).

Table 4.36: The table shows the attributes of the datasets used in the experiments.

Dataset	# Features	# Samples	#Class
Alon	2000	44	2
Golub	7129	50	2
Gravier	2905	114	2
Pima	9	768	2
Sonar	61	208	2
WBCD	32	568	2

A random search grid was used to search for the number of latent features that gave the best results. The number of latent variables used in this experiment are 5 latent nodes for Alon dataset, 5 latent node for the Golub dataset, 11 latent nodes for the Gravier dataset, 3 latent nodes for the WBCD dataset, 47 latent nodes for the Sonar

dataset, and 8 latent nodes for the Pima dataset. The datasets were split in to 60% training set and 40% testing set with the exception of the Golub dataset which was split 50% training set and 50% testing set.

Discussion

Table 4.37: Accuracy of the different datasets for each of the different machine learning algorithms.

Dataset	SVM Linear	SVM RBF	KNN	Naive Bayes	CorEx-Ci
Sonar	60.24%	46.99%	46.99%	56.63%	**72.29**%
WBCD	**94.30**%	68.42%	**94.30**%	90.79%	**94.30**%
PIMA	**80.46**%	69.71%	72.31%	78.83%	68.73%
Alon	**88.89**%	50.00%	66.67%	66.67%	77.78%
Golub	**96.00**%	48.00%	**96.00**%	**96.00**%	68.00%
Gravier	70.21%	40.43%	**76.60**%	61.70%	74.47%

Table 4.38: Table shows the Matthew's Correlation Coefficient values for the datatsets for the different machine learning algorithms.

Dataset	SVM Linear	SVM RBF	KNN	Naive Bayes	CorEx-Ci
Sonar	0.294727	-0.070680	-0.062910	0.293592	**0.426901**
WBCD	**0.868592**	0	0.866819	0.784813	0.866819
PIMA	0.51998	0	0.36270	0.5098	0.41180
Alon	**0.777778**	0	0.447214	0.447214	0.569803
Golub	**0.923077**	0	**0.923077**	**0.923077**	0.454257
Gravier	0.372444	0	**0.530573**	0.178988	0.459105

The experiments show that CorEx-Ci is competitive against other machine learning algorithms among the Sonar and WBCD datasets. It is performs comparably with the Gravier dataset. For the Golub dataset and PIMA dataset, it performs relatively poorly in comparison to the other classifiers. A reason for this could be that the data generation is a mixture since CorEx-Ci is able to give a positive MCC value albeit not as high as the others. For the Alon dataset, it performs better than KNN, SVM with RBF Kernel, and Naive Bayes but performs worse than SVM with Linear Kernel. As with all classifiers, CorEx-Ci does not work well with all datasets but only with datasets which meets its assumptions. CorEx-Ci shows some promise in terms of prediction. However, in

order to achieve a better accuracy, we speculate the need to condition the structure of the CorEx-C with stronger bias towards the class label during training. A future work would be to incorporate a stronger regularizer as an added constraint for training. Expanding the CorEx assumption would also be an important step to allow flexibility to allow a larger class of data to be applied successfully.

4.3.3 Control Experiments

The control experiments tests the sensitivity of CorEx-Ci towards redundant features and its response to varying sample size.

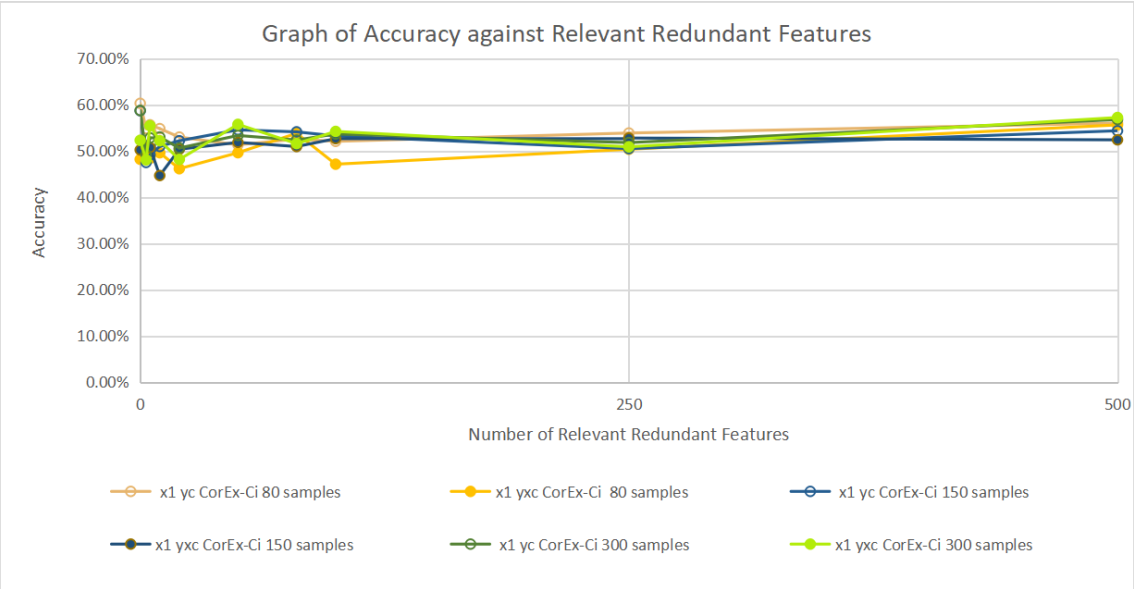
The redundant features were varied based on three criteria:

1. Relevant Redundant Features: The features that are considered relevant to classification is duplicated and a linear transformation is added to the new features. These features contain no new information although it is relevant to the class label.

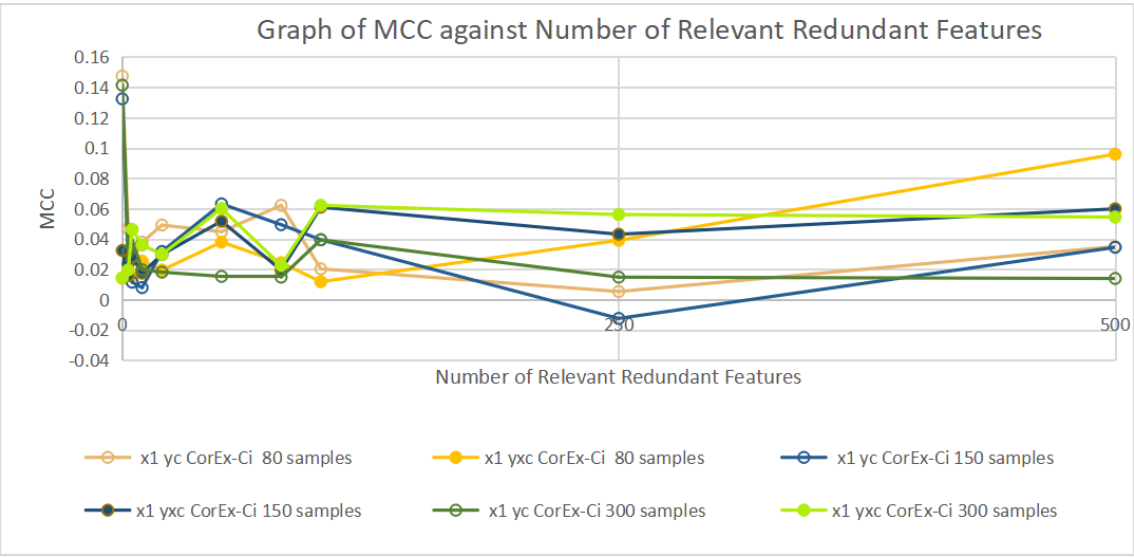
2. Irrelevant Redundant Features: The features that are not related to the class labels are duplicated and a linear transformation is applied. These features are added noise which could bias the modelling of the classification away from the actual model.

3. Relevant and Irrelevant Redundant Features: All features regardless of whether they are related to the class label or not are duplicated and a linear transformation applied. The ratio of relevant to irrelevant features remain the same as the original ratio.

Effect of Redundant Features on the Classification Performance of CorEx-Ci

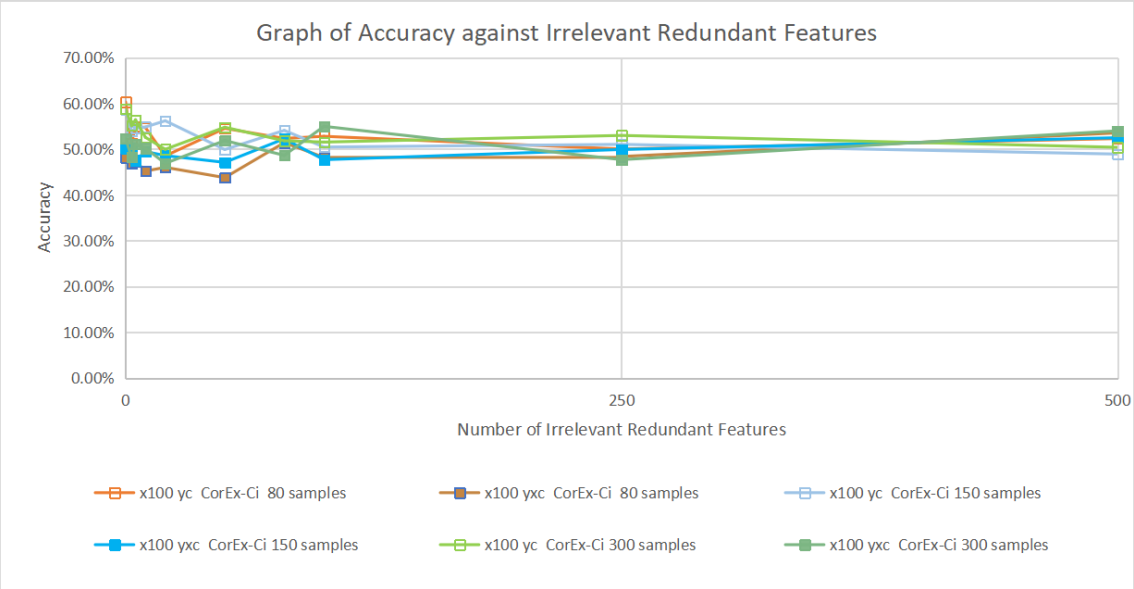


(a) Accuracy

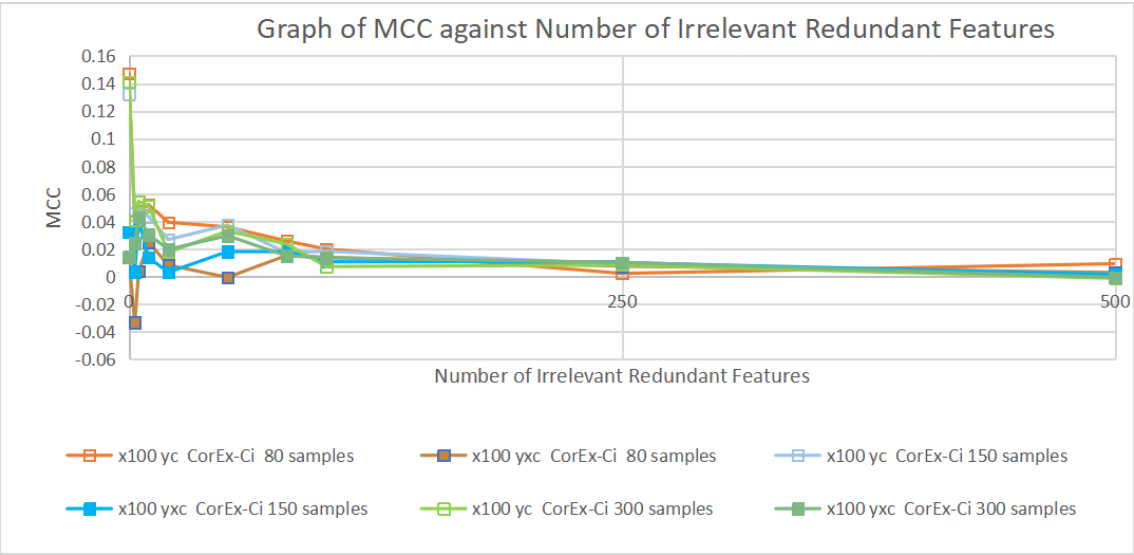


(b) MCC

Figure 4.6: The figure shows the trend when the redundant features which are relevant to the class label increases. x1 refers to the relevant features being increased in redundancy through linear transformation.

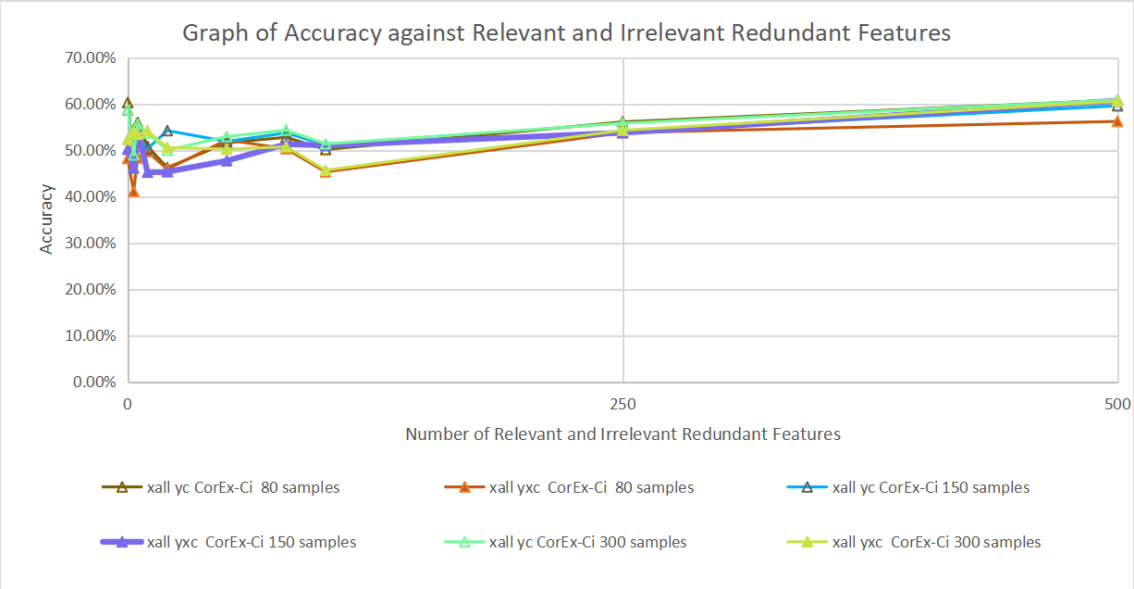


(a) Accuracy

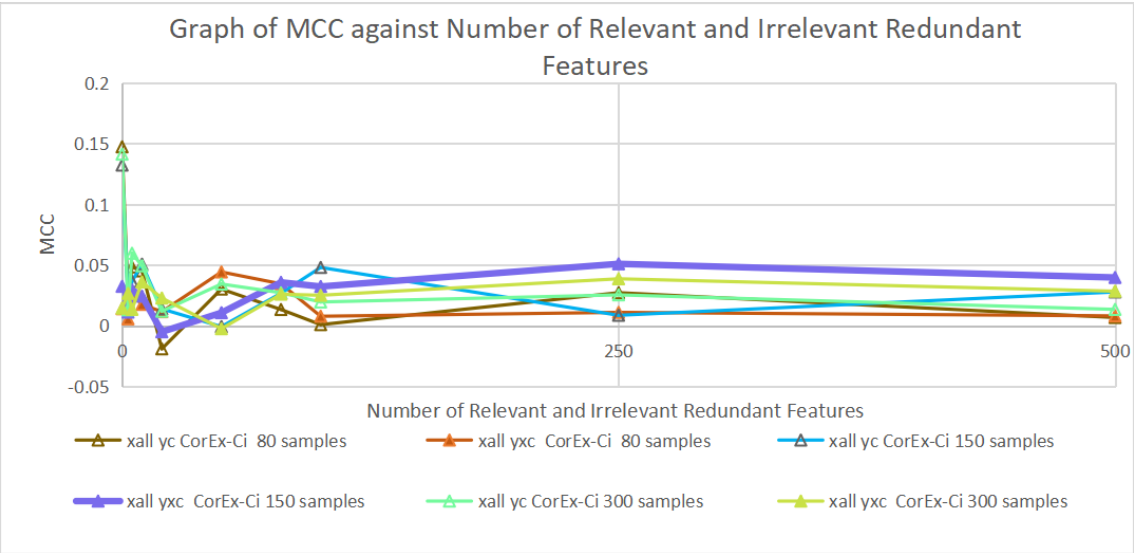


(b) MCC

Figure 4.7: The figure shows the trend when the redundant features which are irrelevant to the class label increases. x100 refers to the irrelevant features being increased in redundancy through linear transformation.



(a) Accuracy



(b) MCC

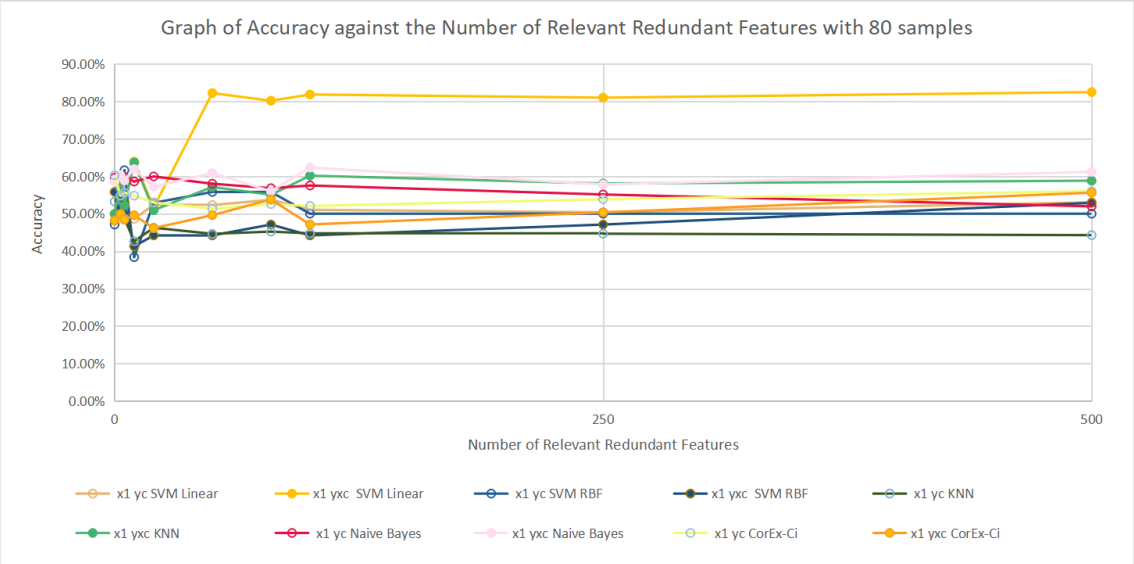
Figure 4.8: The figure shows the trend when the redundant features which are both relevant and irrelevant to the class label increases at a constant rate. xall refers to both the relevant and irrelevant features being increased in redundancy through linear transformation.

Figs.(4.6, 4.7, 4.8) compares the sensitivity of CorEx-Ci to different types of redundant features. Figs.(4.6, 4.7, 4.8) shows that the data generated using the $Y \rightarrow X, Y \rightarrow C$ decreases both in accuracy and stability as the number of redundant features increases. However, in the case of data generated through the $Y \rightarrow X \rightarrow C$ process, when the redundancy is increased by increasing the relevant features and keeping the irrelevant features at a constant, we find that the accuracy and generalisation capability of CorEx-

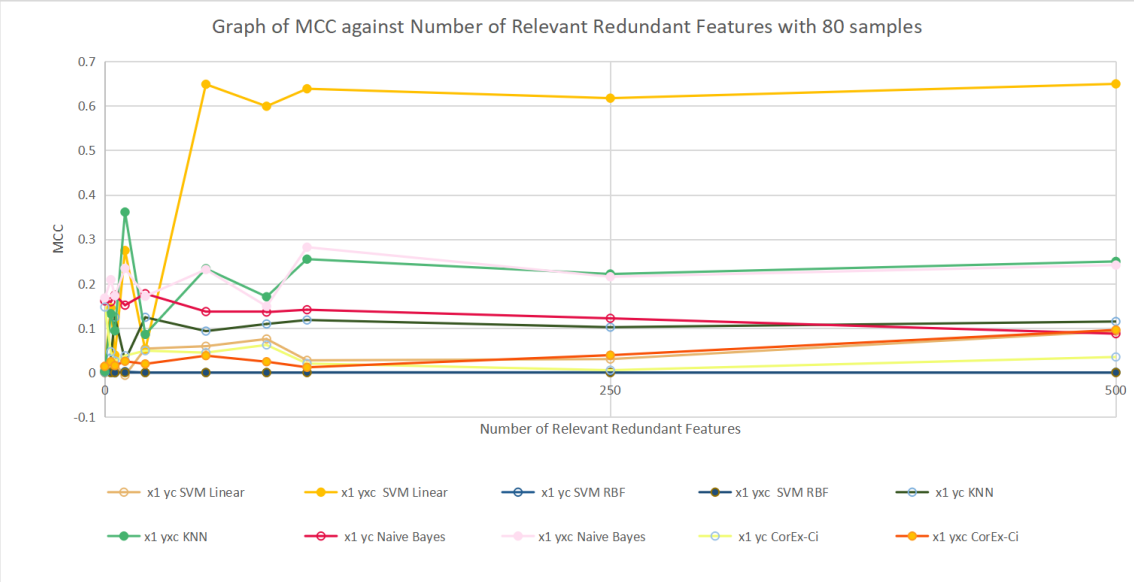
Ci increases. We increased the number of relevant features by applying linear transformation to dataset. Fig.(4.6b) also shows that the increase in generalisation capability is more prominent with smaller samples. A reason for this could be that the overwhelming number of relevant redundant features is able to bias the training with minimal adversarial samples.

The redundancy in Fig.(4.7) is increased by applying linear transformations to the pre-existing irrelevant features in the dataset. As expected, as the number of irrelevant features increased, the generalisation capability and accuracy of the CorEx-Ci also decreases. When redundancy is increased by increasing both relevant and irrelevant features at the same rate, we find that the accuracy improves slightly for $Y \rightarrow X \rightarrow C$ generation process. The generalisation capability peaks slightly initially and converges to the original generalisation capability. This could be due to the fact that the ratio of relevant and irrelevant features are constant, hence cancelling out each others effect on the dataset.

Benchmarking the Effect of Redundancy between the Existing Classification Methods and CorEx-Ci with Varying Sample Size

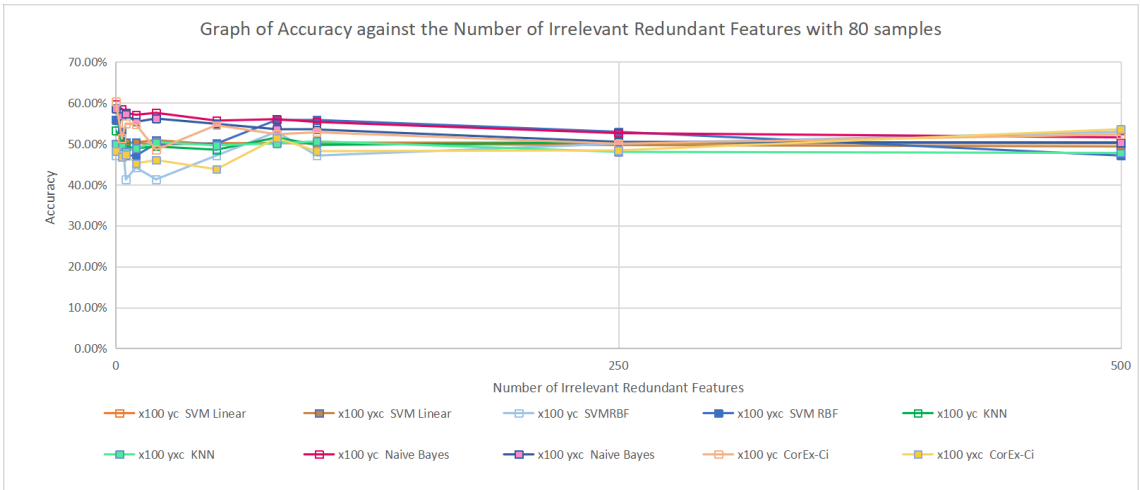


(a) Accuracy

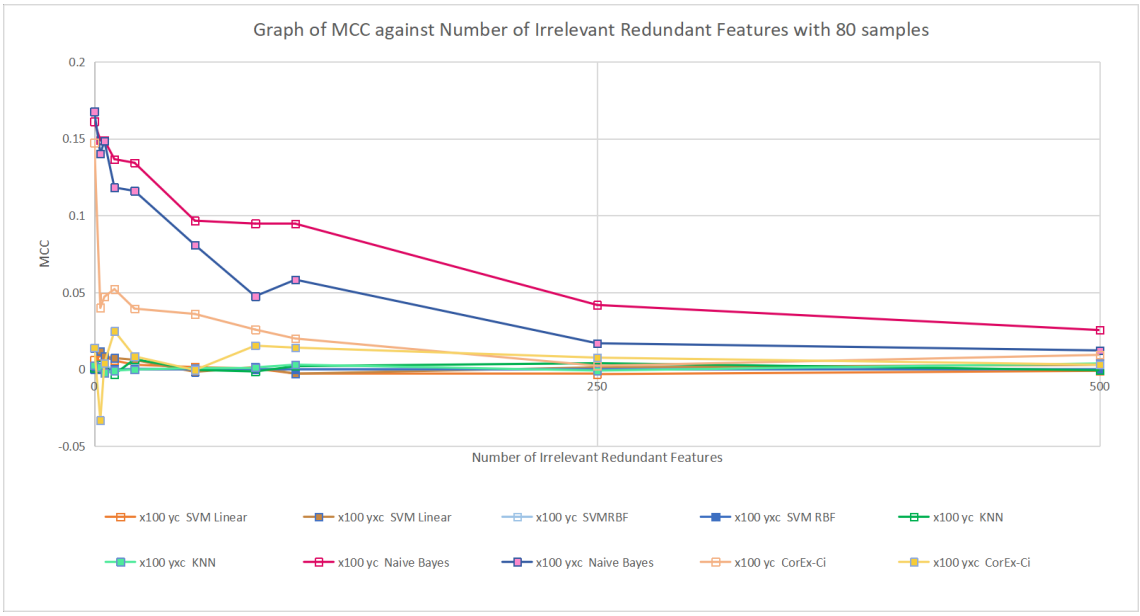


(b) MCC

Figure 4.9: The figure shows the trend when the redundant features which are relevant to the class label increases when sample size is 80. x1 refers to relevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

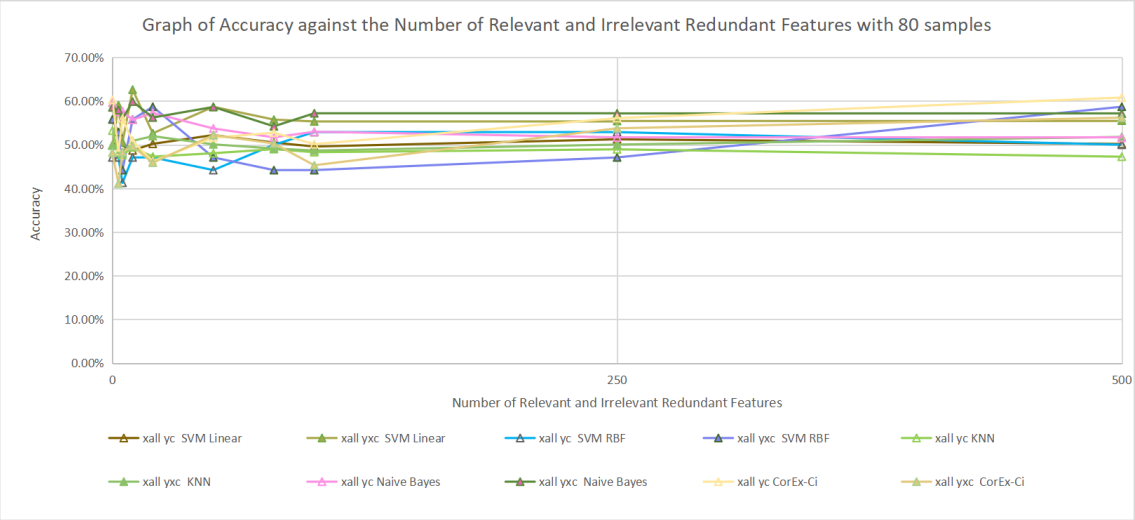


(a) Accuracy

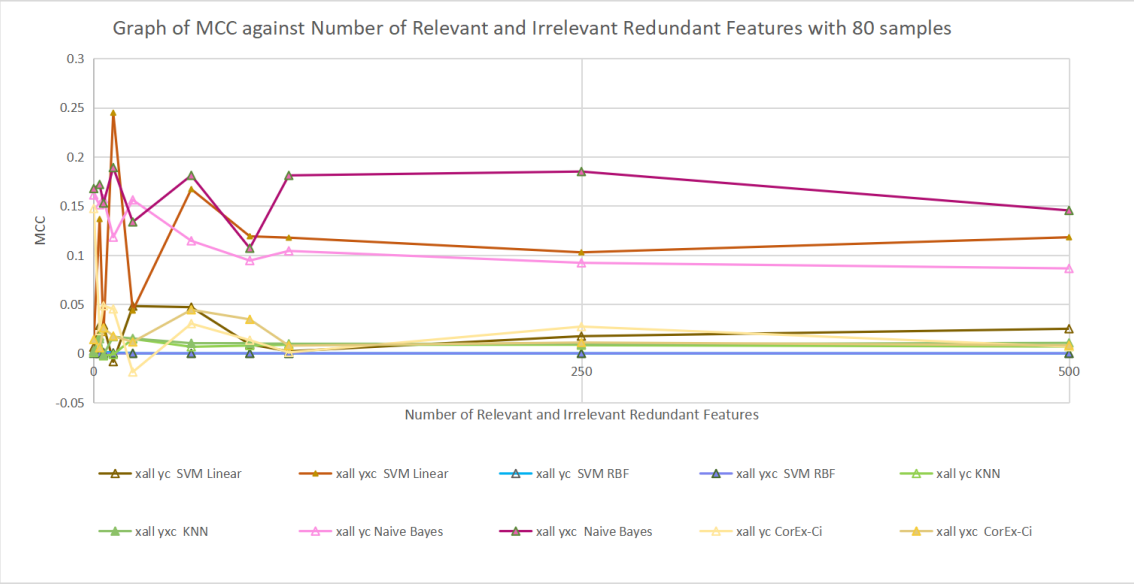


(b) MCC

Figure 4.10: The figure shows the trend when the redundant features which are irrelevant to the class label increases when the sample size is 80. x100 refers to irrelevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

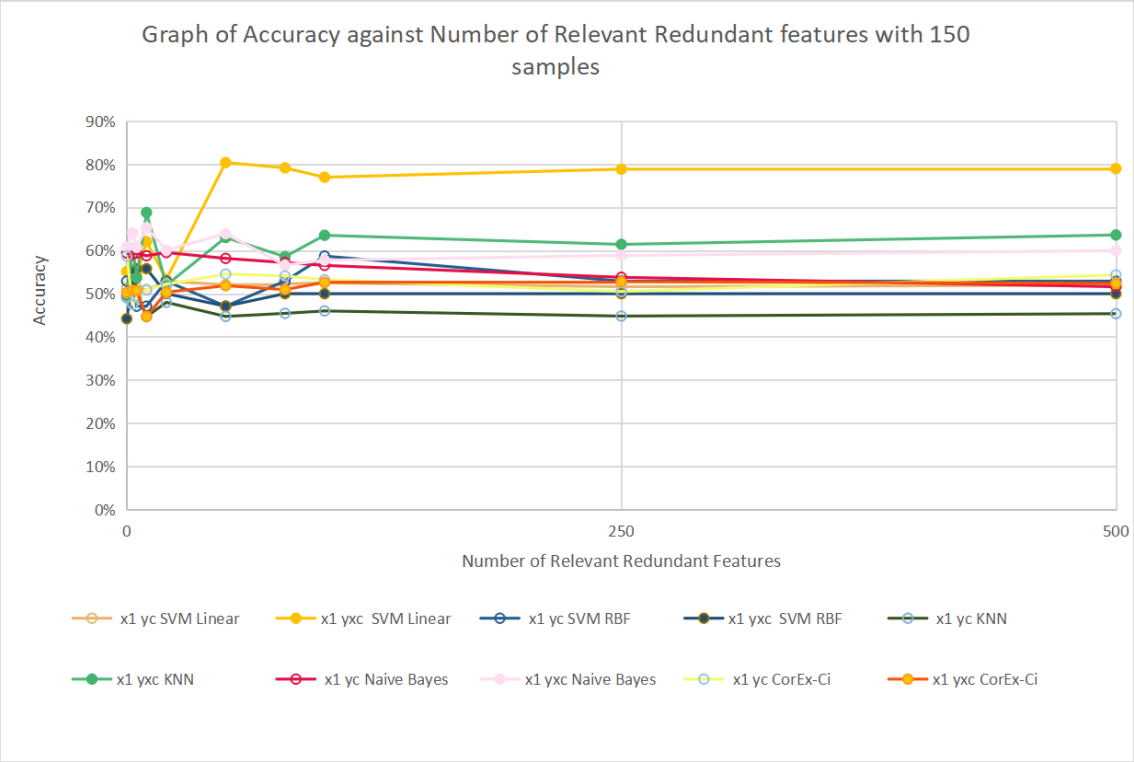


(a) Accuracy

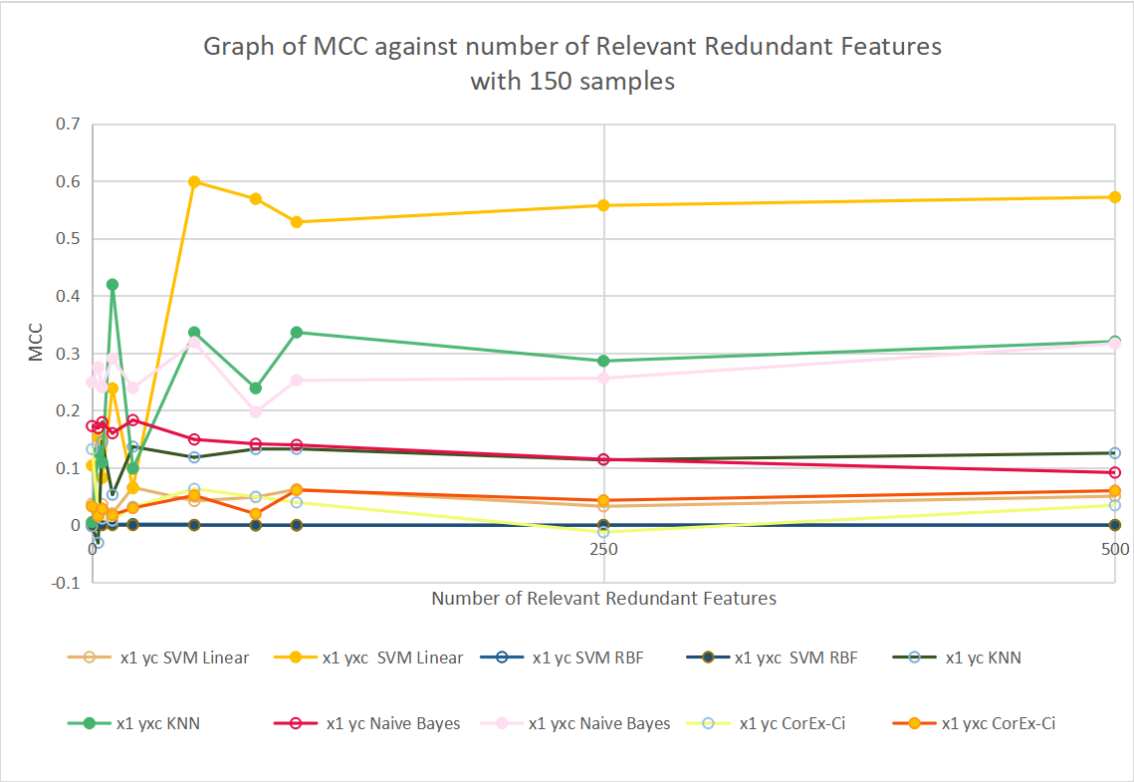


(b) MCC

Figure 4.11: The figure shows the trend when the number of relevant and irrelevant redundant features to the class label increases when the sample size is 80. xall refers to both relevant and irrelevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

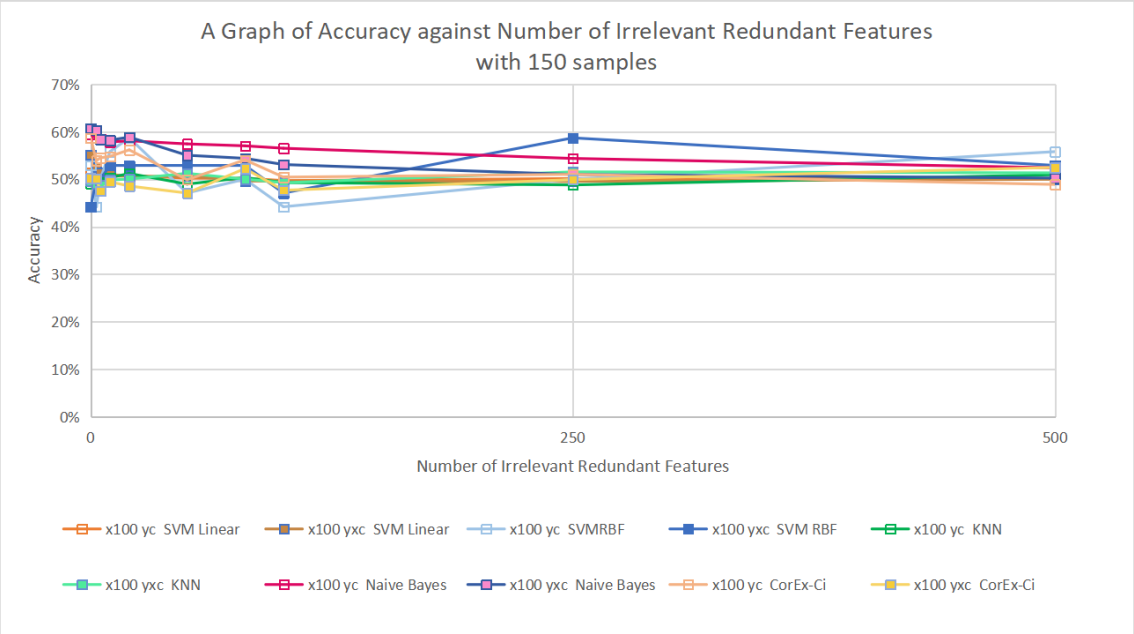


(a) Accuracy

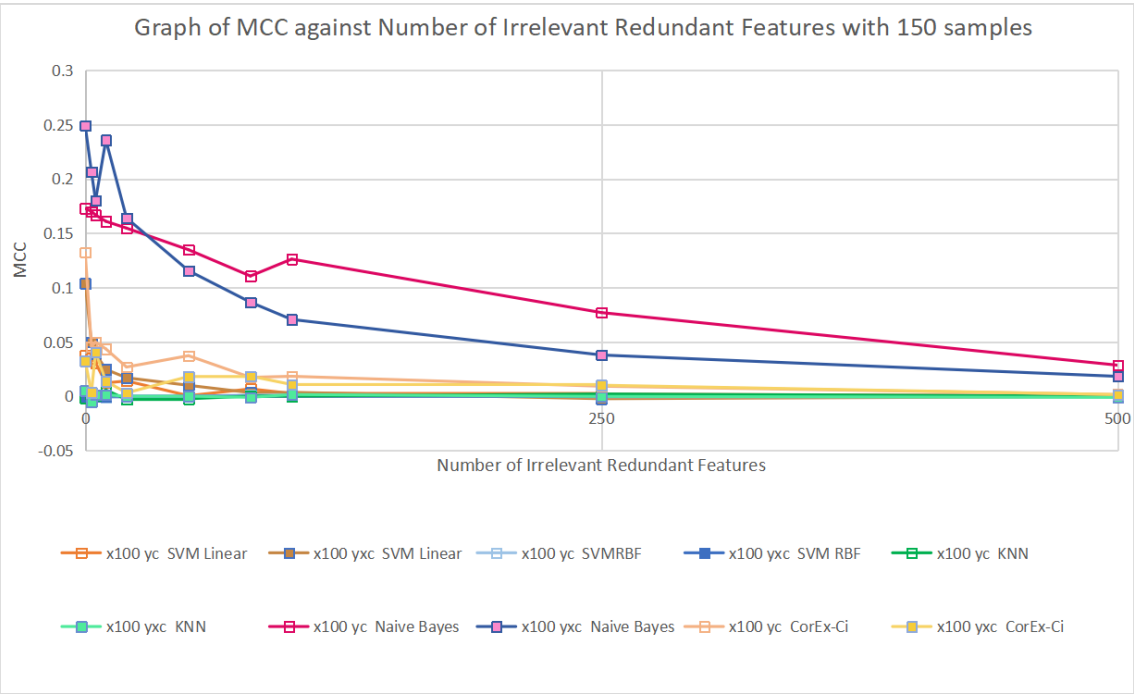


(b) MCC

Figure 4.12: The figure shows the trend when the redundant features which are relevant to the class label increases when the sample size is 150. x1 refers to relevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

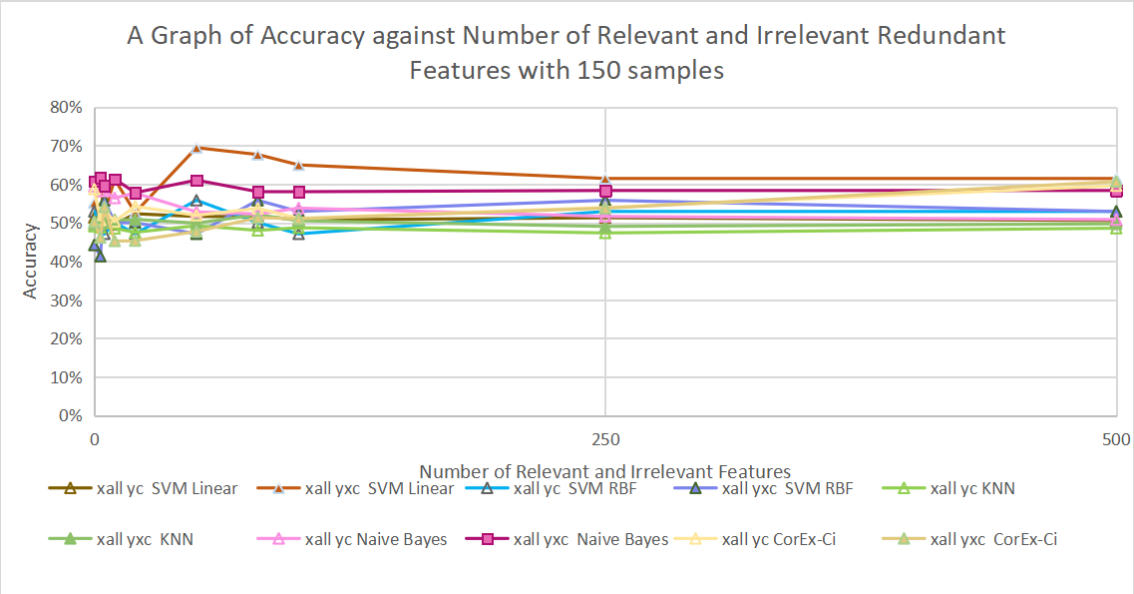


(a) Accuracy

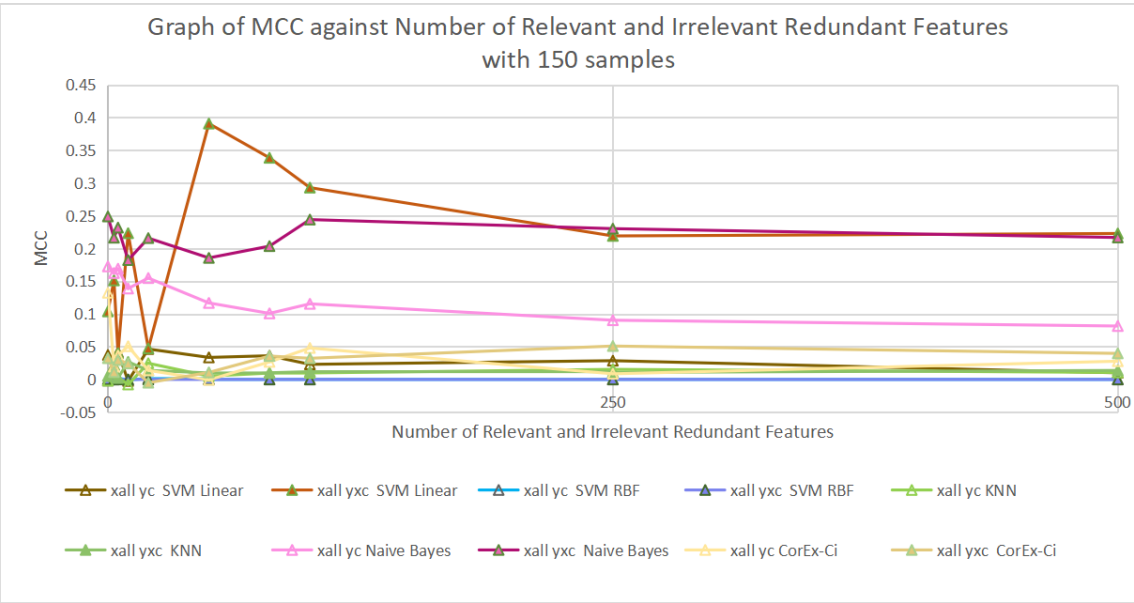


(b) MCC

Figure 4.13: The figure shows the trend when the redundant features which are irrelevant to the class label increases when the sample size is 150. x100 refers to irrelevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

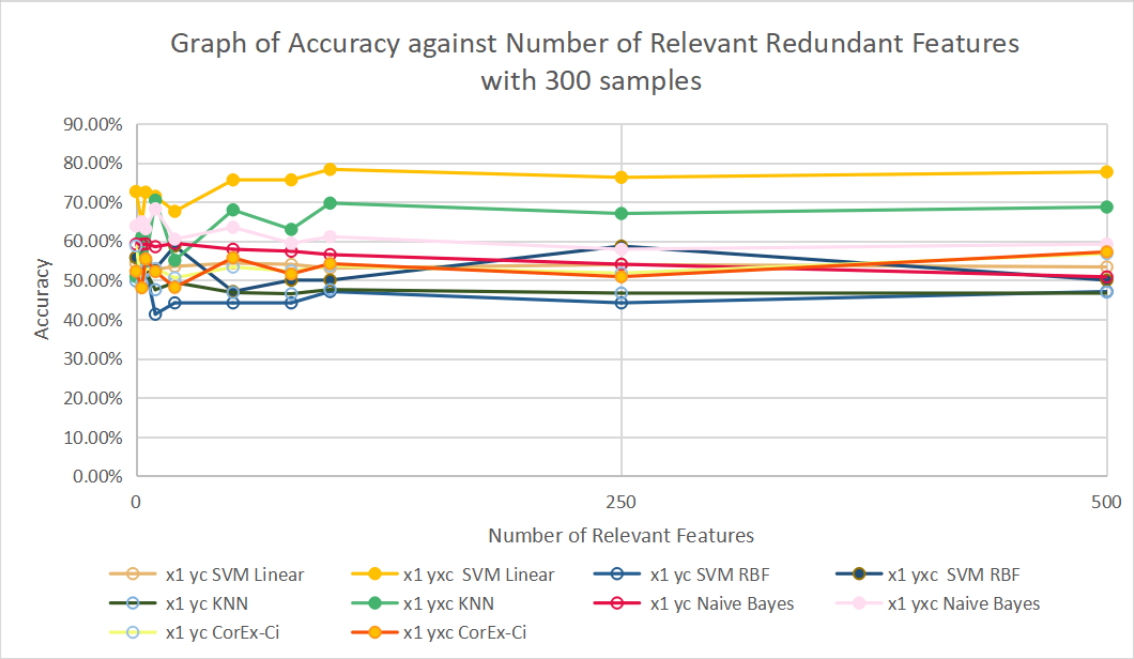


(a) Accuracy

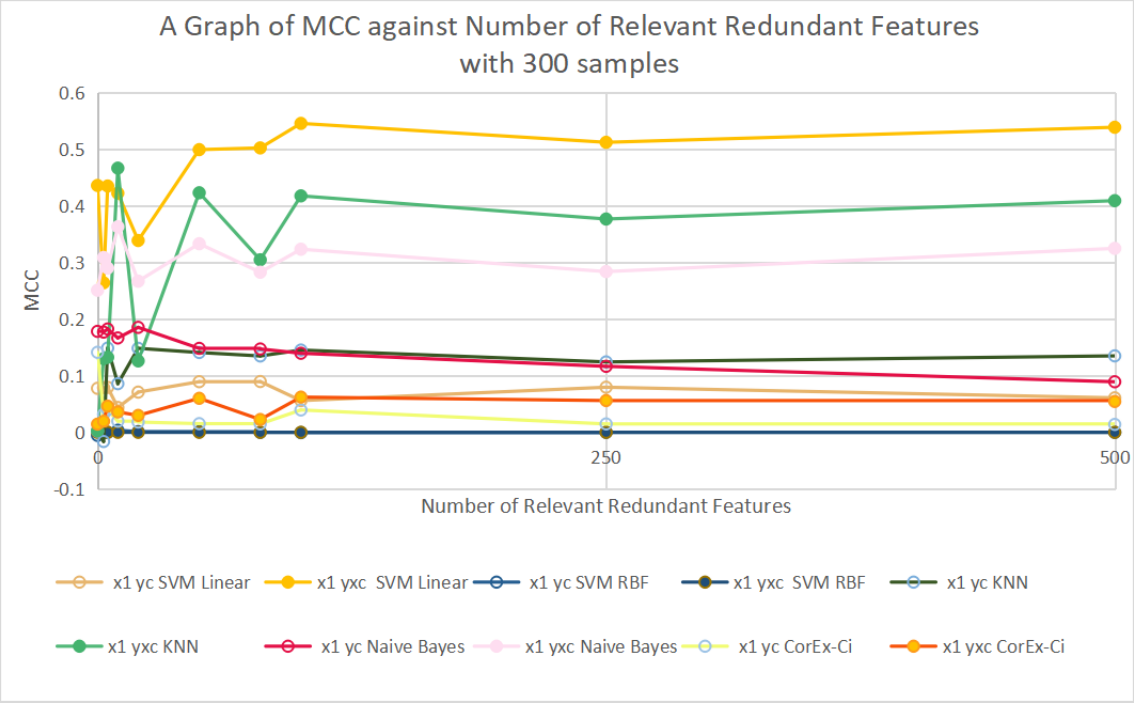


(b) MCC

Figure 4.14: The figure shows the trend when the number of relevant and irrelevant redundant features to the class label increases when the sample size is 150. xall refers to both relevant and irrelevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

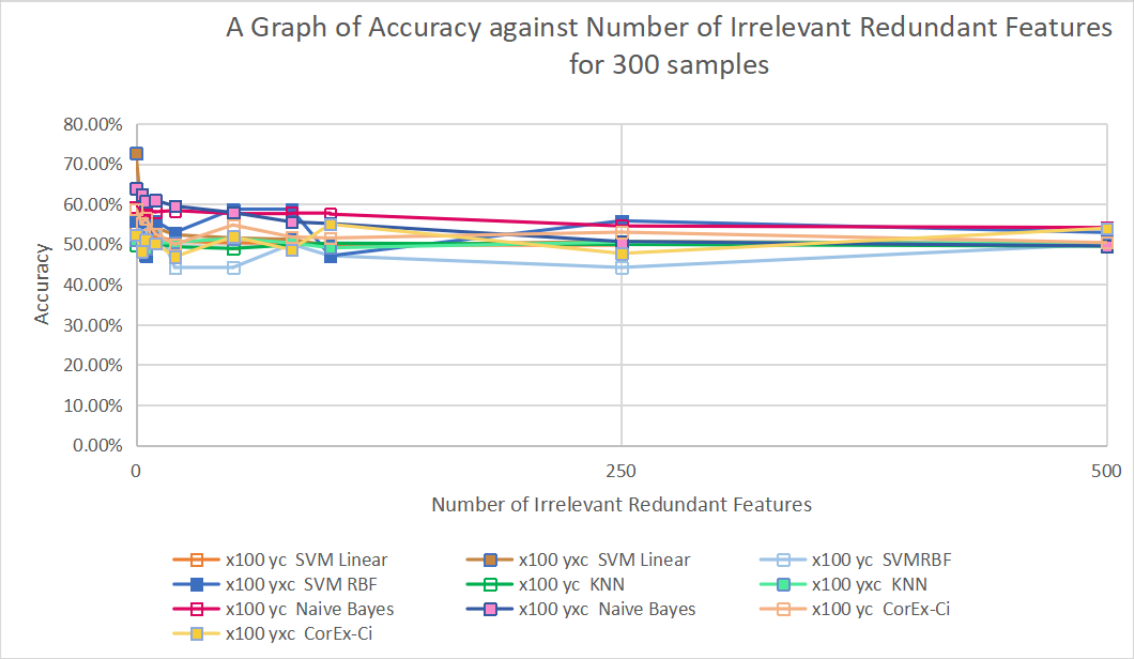


(a) Accuracy

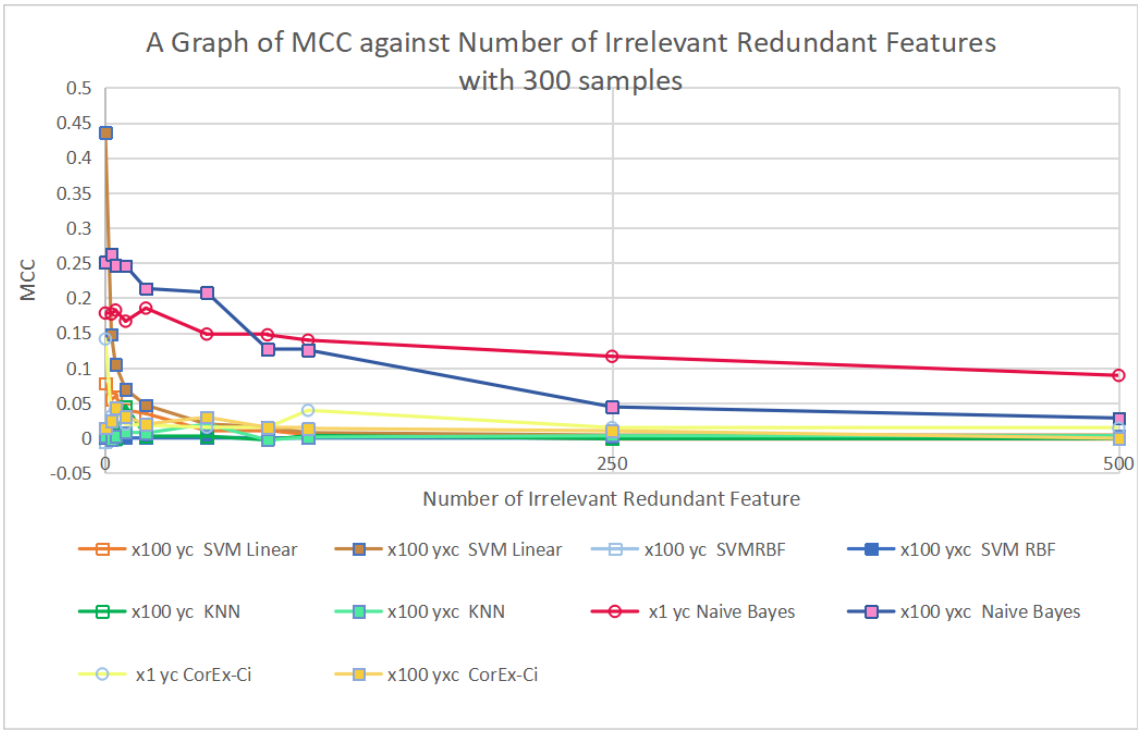


(b) MCC

Figure 4.15: The figure shows the trend when redundant features which are relevant to the class label increases when the sample size is 300. x1 refers to relevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

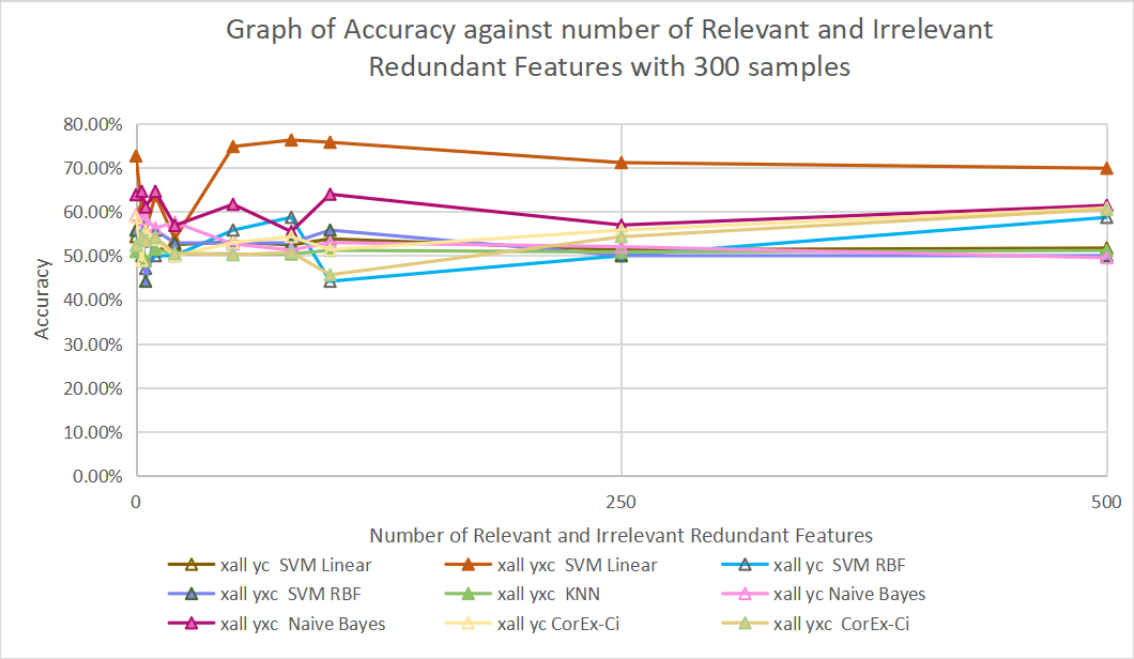


(a) Accuracy

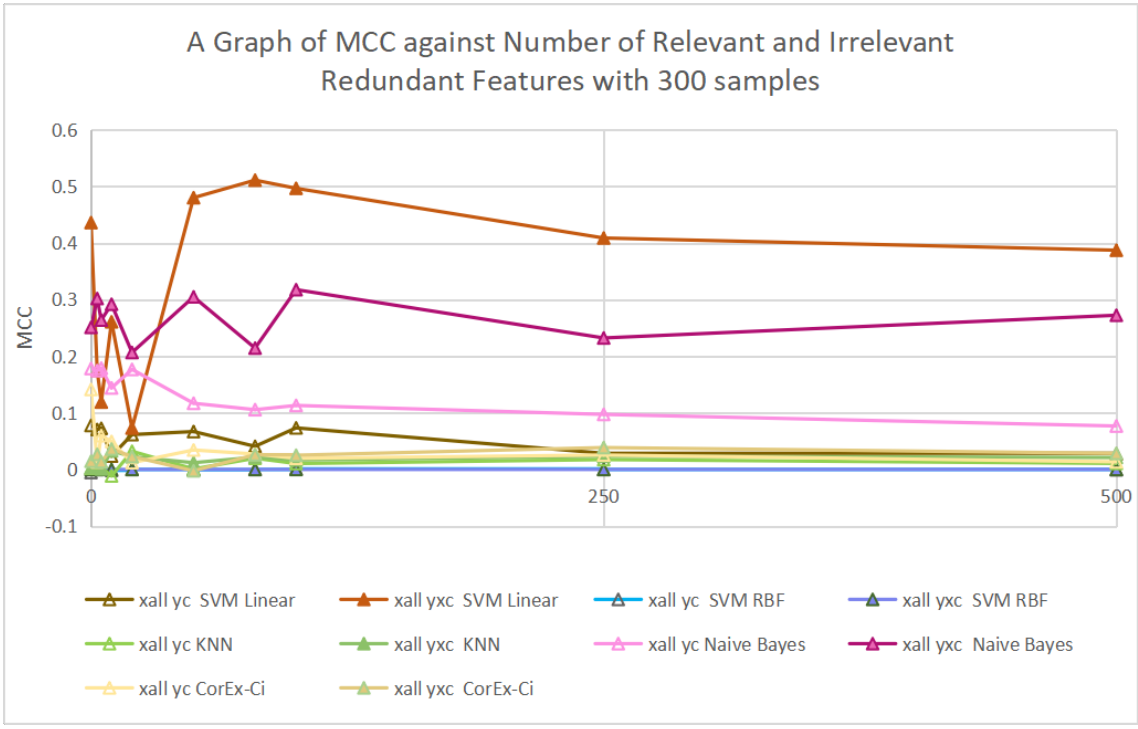


(b) MCC

Figure 4.16: The figure shows the trend when redundant features which are irrelevant to the class label increases when the sample size is 300. x100 refers to irrelevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.



(a) Accuracy



(b) MCC

Figure 4.17: The figure shows the trend when the number of relevant and irrelevant redundant features to the class label increases when the sample size is 300. xall refers to both relevant and irrelevant features being increased in redundancy through linear transformation. The plots compare the trend by different classifiers.

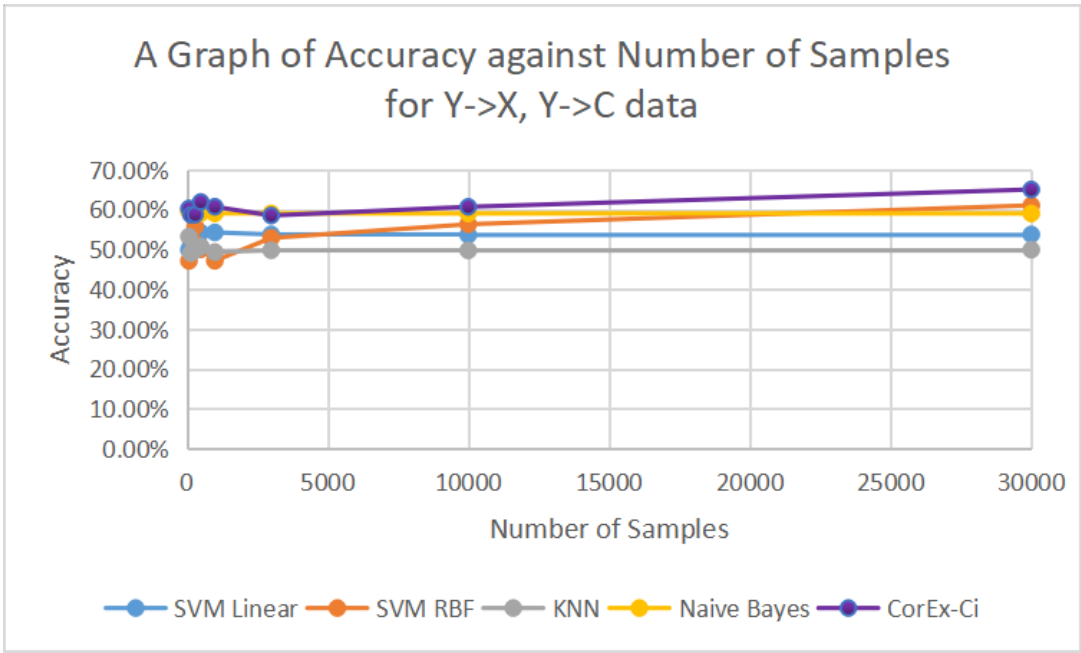
We plotted the trend when the number of redundant features are increased for SVM with Linear Kernel, SVM with RBF Kernel, K-Nearest Neighbour, and Naive Bayes. As with CorEx-Ci, we find that when the number of relevant features increase, the accuracy and generalisation capability of the classifiers also increases with the exception of SVM with RBF Kernel. SVM with Linear Kernel shows the most increase in performance as the feature size increases with the addition of redundant features which are relevant to the class label. This could also be due to the fact that linear transformation was applied in order to increase the dimensionality of the dataset. We also see that like CorEx-Ci, SVM with Linear Kernel shows a higher increase in performance with smaller sample size (i.e. 80 samples compared to 150 and 300 samples) as the number of relevant but redundant features increases. Naive Bayes and KNN on the other hand shows higher increase in performance as the sample size increases. We find this trend very interesting as it shows that under the HDLSS asymptotic, if most of the features are relevant to the class label, the classifiers will still be able to give a reasonable performance. This would most likely be because the additional features are able to overpower the other features to bias the classifiers. However, the increase in performance is higher for the other classifiers compared to CorEx-Ci. This shows that while the additional relevant features can skew the data, CorEx-Ci is not as influenced by it as the other classifiers. A reason for this would be that there is very little increase in new information captured through total correlation by CorEx-Ci.

Figs.(4.10,4.13,4.16) shows that as the number of irrelevant features increase, the generalisation capability of the classifiers decreases. Naive Bayes is the most robust followed by CorEx-Ci against irrelevant features as they decreased in performance less than the others. CorEx-Ci applied on the dataset generated through the $Y \rightarrow X, Y \rightarrow C$ process performs marginally better than KNN, SVM with Linear Kernel, and SVM with RBF Kernel. SVM with Linear Kernel is able to withstand against irrelevant features better as the number of samples increase. This is seen in through the plots in Figs. (4.10,4.13,4.16). Figs. (4.11,4.14,4.17) show that the performance peaks slightly at lower sizes of redundant features before decreasing and converging in performance as the number of redundant features increase. A reason would be due to the cancelling out of effects as ratio of redundancy among the relevant and irrelevant features remain the same.

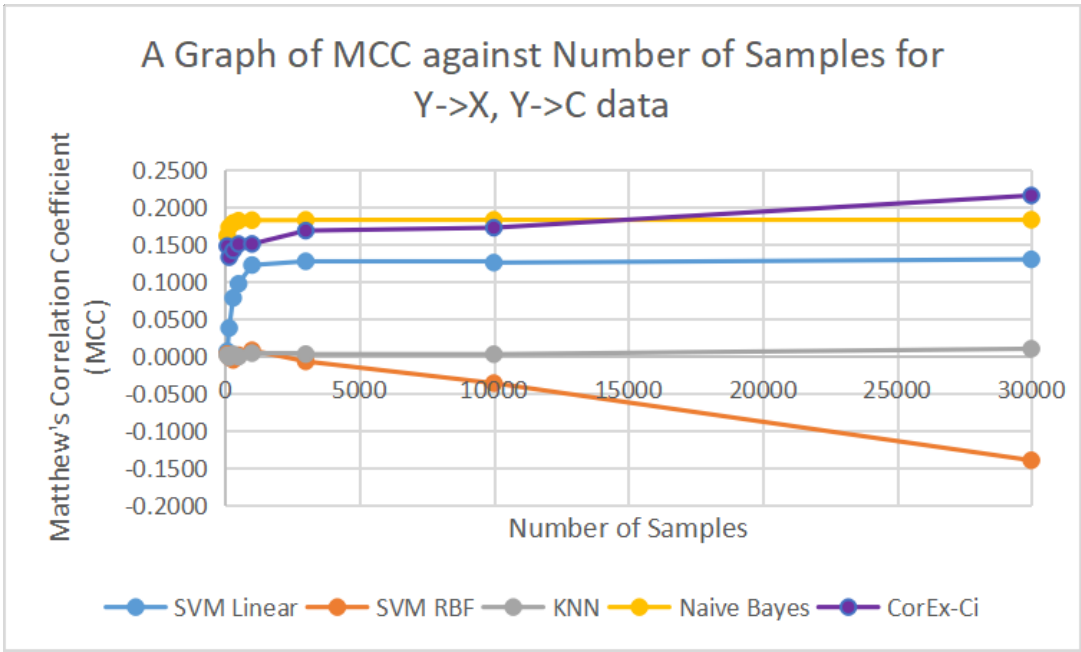
In most cases, too much redundancy is seen as a bad thing for classifiers as it is seen as reducing the effectiveness of the classifier and increasing the complexity of the data. This reflected by many feature selection methods which tries to reduce redundancy (Liang *et al.*, 2019). Redundant variables very seldom add new information to

the data especially if they are collinear in nature. However, through our simulated experiments, we show that if the redundancy within the dataset is relevant to the class label, the accuracy and robustness of the classifier can improve. It is only in the case of irrelevant features found in abundance that the classification performance starts to degrade. The reason for this is that the irrelevant features bias the model away from the actual representation of relationship between the class and relevant features. Relevant features on the other hand bias the model towards the relevant features.

Effect of Sample Size on Classification Algorithms

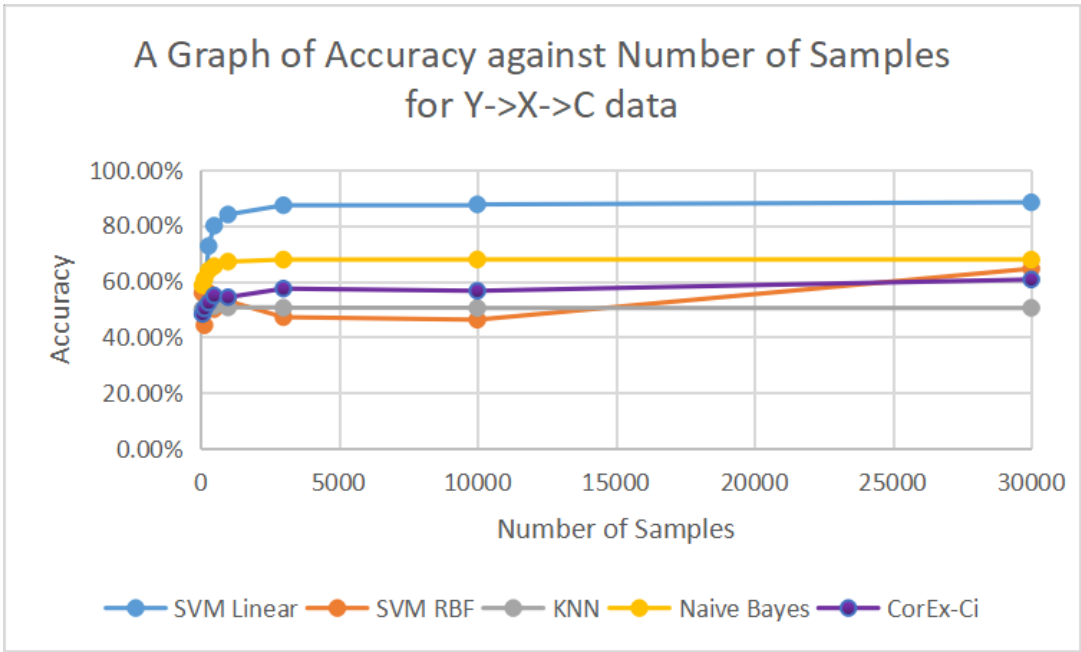


(a) Accuracy

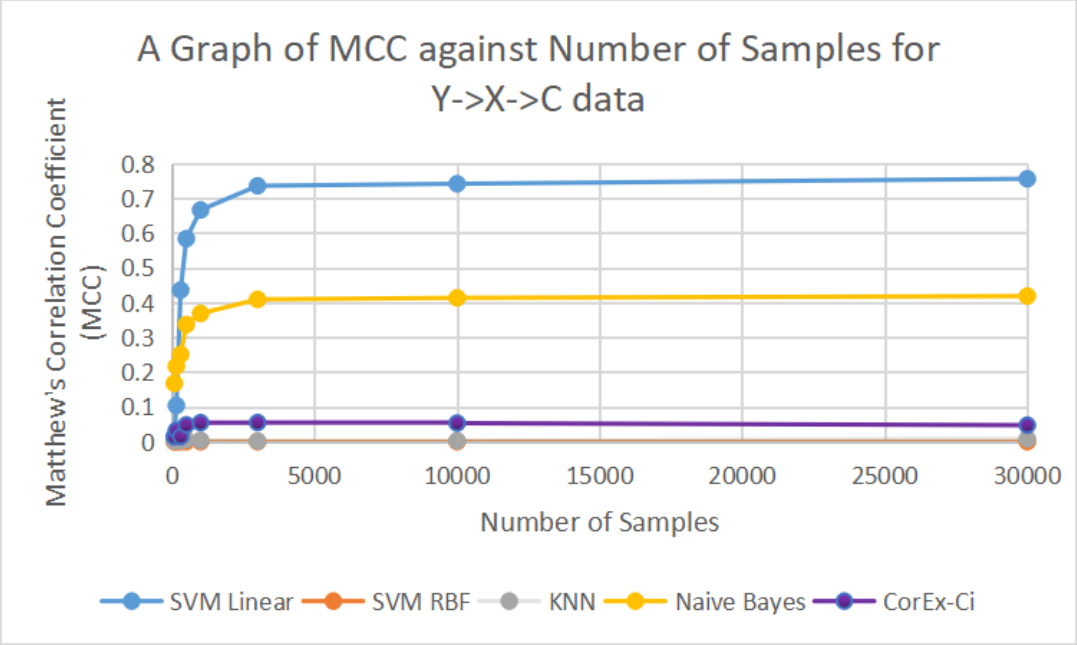


(b) MCC

Figure 4.18: The figure shows the trend when the number of samples increases when the input dimension remains fixed for data generated through $Y \rightarrow X, Y \rightarrow C$ process.



(a) Accuracy



(b) MCC

Figure 4.19: The figure shows the trend when the number of samples increases when the input dimension remains fixed for data generated through $Y \rightarrow X \rightarrow C$ process.

Figs. (4.18) and (4.19) shows the trend when the number of samples increases but the number of input dimensions remain fixed. In the case of data generated through the $Y \rightarrow X, Y \rightarrow C$, the accuracy peaks before converging. While SVM with RBF Kernel shows improvement in terms of accuracy its robustness decreases as the sample size

increases. The generalisation capability of KNN does not improve while SVM with Linear Kernel, Naive Bayes, and CorEx-Ci shows slight improvement.

In the case of $Y \rightarrow X \rightarrow C$ it can be seen that SVM with Linear Kernel shows the greatest improvement in accuracy and generalisation capability as the sample size increased. Although the accuracy of SVM with RBF Kernel increased, its generalisation capability remains constant. Both CorEx-Ci and KNN did not show much improvement as the sample size increased. In the case of CorEx-Ci, only slight improvement was seen most probably because only some of the relevant features have been captured even as the sample size increased as shown by the heatmap in Fig.(4.3).

Hence, the control experiments show that CorEx-Ci is less sensitive to redundant features compared to the other classifiers except Naive Bayes. The performance of CorEx-Ci improves only slightly when the sample size increases compared to the other classifiers when the data generation assumption is met. This shows the robustness of CorEx-Ci compared to the other classifiers when the sample size is varied. It is indicative of the fact that CorEx-Ci reaches its asymptotic error at a faster rate than the other classification methods.

4.4 Summary

In this chapter, we tested how the CorEx-C model responds to the different data generation processes. We find that the CorEx-C model is more stable with $Y \rightarrow X \rightarrow C$ data generation process. We also found that when the sample size is high, the effects of the violation of the data generation assumption of CorEx-C diminishes.

Under the discriminant classification umbrella we test CorEx-based feature extraction and selection methods. We find that methods which reduces redundancy has an advantage over methods which do not reduce redundancy in HDLSS situations. Due to this, CorEx-UF, although an unsupervised method is competitive and at times better than CorEx-SFS. However, we also note that the effect of redundancy diminishes as the sample size increases when the data generation assumption is violated. We also saw that when the data generation process is not violated, the increase in sample size does not change the effect of redundancy on the dataset. This is also further shown with the control experiments to test the effect of redundancy and sample size on CorEx-Ci. CorEx-SF performs better than the CorEx-UF and CorEx-SFS provided the data generation assumption is not violated. When benchmarking the data, we find that in general, CorEx-SF based methods are competitive and at times better than MRMR and

CIFE provided the data generation assumption is not violated. Expanding CorEx-C to reduce the restriction imposed by the data generation assumption is a future work.

Under the generative classification model, we show that CorEx-Ci has a slightly higher asymptotic error. This also means that the sample size bears little to no effect on CorEx-Ci. This indicates that the error rate is the same with lower number of samples as it is with higher number of samples. Expanding the learning of the model for CorEx-Ci would be a future direction to investigate in order to reduce the asymptotic error. We note that CorEx-Ci is more robust to redundancy compared to some of the other classification methods. Robustness to redundancy is an important factor as HDLSS data often contains redundant information. This is a consideration that needs to be addressed as more research is done with HDLSS data.

Hence, we show that CorEx-based methods can be expanded to be used for classification for HDLSS data but more research needs to be done in order to fully tap into its potential. We also show that without generating more samples, HDLSS data can be handled.

Chapter 5

Conclusion & Future Works

High dimensional low sample size (HDLSS) data is becoming more prevalent. Research in this area is important as HDLSS data is commonly found with medical and microarray data. Many algorithms in the field of machine learning and data mining require a reasonably large amount data in order to perform well. A large number of samples will ensure that statistical significance is maintained. The algorithms are usually catered for high dimensional data or data with large sample size especially with data being more freely available with boom of big data. Unlike high dimensional data, HDLSS data has a slightly different behaviour in the Euclidean space. The asymptotic nature of HDLSS means that the number of samples does not increase or increases at a very slow pace in comparison to the growth of the number of dimensions.

In this thesis, HDLSS data is classified by exploiting the intrinsic structure of the data in a supervised manner through the use of probabilistic graphical models. There are two problems that arise with HDLSS data, i.e. its high dimensionality and its low sample size. Literature generally addresses these as separate problems. However, as seen through HDLSS datasets, the problems are not necessarily mutually exclusive of each other. We focused our efforts in dealing with the high dimensionality aspect of HDLSS datasets, but keeping in mind its sample size. Throughout this thesis, we focused on classification of HDLSS datasets through discriminant classification and generative classification. These methods are two categories of classifiers.

We aimed at learning the intrinsic structure of the data through the use of probabilistic graphical models. Classification methods help to discriminate and identify categories of data. The categories of data is also known as class label. Classical classifiers are built for high dimensional data which has a huge sample size, or data which is not high dimensional but contains sufficient samples. The sufficiency of samples for the classifiers is an open problem and is not addressed in this thesis. The assumption

made throughout this thesis is that HDLSS has sample size lower than the dimension and is not sufficient to produce a statically significant result with the classifiers.

Support Vector Machines (SVM) do not perform well with HDLSS datasets. This is because data piling occurs. Data piling clumps the support vectors making it difficult for the hyperplane to segregate between the classes. Classifiers such as K-Nearest Neighbour (KNN) do not perform well using Euclidean distances because the pairwise distance between points becomes equidistant from each other for features drawn independently from a probability distribution. As most classifiers make the i.i.d. assumption, this would be the case for distance based methods. Metric Learning techniques are often integrated with KNN to learn distance functions suitable for the data. However, it makes the assumption of having sufficient samples. Further, it uses a process known as whitening which requires the squaring of data. This uses the covariance matrix. The covariance matrix is not accurate for HDLSS data as it has low sample size. Naive Bayes is a method that is least affected by the curse of dimensionality as its conditional independence assumption decouples the features. This would mean that Naive Bayes works in a 1-dimensional space. Neural networks are used as a classification method. It learns the intrinsic structure of the data through its hidden nodes. However, it requires large amounts of data to accurately learn the structure. Another drawback with Neural Networks is the black box effect. It lacks the ability to explain the factors which are related to classification. The explainability of artificial intelligence is required in medical fields. The density estimation done within a classifier can degenerate due to lack of samples. This causes the structure of the data to be misrepresented. As seen, classification methods were not created to cater for HDLSS data.

Correlation Explanation (CorEx) is an unsupervised probabilistic graphical model based algorithm which uses total correlation as part of its objective function. It has a novel method to estimate the probability distribution which works under HDLSS conditions. It learns the intrinsic structure of the data through its hidden variables. Unlike neural networks, it also is able to map the features which contributes to the intrinsic structure. In order to address the problems that arise from the lack of stability of the estimation of density of probability distribution for classification, we extend CorEx to support class labels. To further meet our objective for classification of HDLSS data, we extended CorEx with class labels to encapsulate both the discriminant classification and the generative classification case.

Our contribution is this thesis is three-fold in order to meet our aims and objectives for the research:

1. Extending CorEx to include the class label. This graphical model proposed is known as CorEx-C.

2. Exploring the use of CorEx-C under the discriminant classification umbrella by proposing supervised feature extraction (CorEx-SF) and supervised feature selection (CorEx-SFS) methods. The performance of the proposed methods were tested through the use of classifiers.

3. Exploring the use of CorEx-C under the generative classification umbrella by proposing CorEx-Ci, which is an extension to CorEx-C by including an inferencing algorithm.

In this thesis, in order to extend CorEx to support classification, we exploited the modular structure of CorEx to enable a mixture distribution to be modeled. We call this structure CorEx-C. The modification to the probability distribution is an important step in order to ensure that the class label is compatible with the CorEx structure. Without this step, the class label would either be considered continuous or the remainder of the dataset would have to be discretized. Both cases incorporates an erroneous assumption of the data. A further investigation on the CorEx-C structure show that the key assumptions made by CorEx itself does not change. However, it adds an additional category of assumption of the data generation process that was not required to be handled by CorEx as no distinction would have been made. In order to test the additional category of assumption, we simulated data through ancestral sampling technique. We tested it under two data generation assumption, i.e. $Y \rightarrow X \rightarrow C$ and $Y \rightarrow X, Y \rightarrow C$, where Y is the latent variable, X is the input variables, and C is the class label. We found that given the data generation assumption made by the original CorEx model, i.e. $Y \rightarrow X, Y \rightarrow C$, the CorEx-C model is able to accurately learn the relationship between the relevant variables and the class label. However, in the case of $Y \rightarrow X \rightarrow C$, the data generation assumption is violated. Hence, the model is only able to detect weak relationships between the relevant features and the class label when the sample size is large as shown by the heatmaps. The case of $Y \rightarrow X \rightarrow C$ was not required to be handled by CorEx as any class label used would fall under X. A future direction which can be explored is the effective learning of the CorEx structure in the case of $Y \rightarrow X \rightarrow C$.

In order to extend CorEx-C for discriminant classification, feature selection and extraction methods were applied to HDLSS data and classified using pre-existing discriminant classifiers such as Support Vector Machines and K-Nearest Neighbour. As the dimensions of the HDLSS dataset are high but the sample size low, learning higher order information can be advantageous to supplement the the lack of samples. Infor-

mation theory based methods have the ability to learn higher order relationships. Feature selection and extraction methods which deal with information theory such as Infomax Information Component Analysis (Infomax ICA), Maximization of Mutual Information (MMI), Maximum Redundancy Minimum Relevance (MRMR), Conditional Infomax Feature Extraction (CIFE), and Kernel Entropy Component Analysis (KECA) was investigated and compared in this thesis. The information theory based dimensionality reduction methods can be structured using divergence methods through $D_\alpha - \beta$ framework. The framework consists of two components, i.e. the divergence factor as well as the penalty imposed during dimensionality reduction.

We extended the CorEx-C model to enable feature extraction and selection to test it under the discriminant classification framework. We compared our results with some existing feature selection and extraction methods. We benchmarked it against Conditional Infomax Feature Extraction and Maximum Relevance Minimum Redundancy. The CorEx-C based feature extraction and selection methods are competitive with MRMR and CIFE. We also discovered that some of the methods such as Maximization of Mutual Information is not able to perform under the HDLSS setting. Other methods such as Kernel Entropy Component Analysis and Independent Component Analysis is able to perform reasonably well with HDLSS data. However, both KECA and ICA have a limit on the number of features that can be extracted which is based on the sample size of the data. Our method has the benefit of not having that limitation. However, we note that there is an ideal number of features known as the intrinsic dimensionality that produces the optimal result for each dataset. We further compared the methods based on the number of samples. We find that CorEx-C based dimensionality reduction methods are more robust compared to the others as the features extracted for classification are not impacted by the change in number of samples. The classification performance showed little to no change with the increase in samples. Other methods start performing competitively with CorEx-C as the number of samples increases. This shows that the features extracted through CorEx-C can be used for discriminant classification under HDLSS conditions. We also tested our method against CorEx based unsupervised feature extraction to understand any benefits of including the class label into CorEx. Our findings suggest that CorEx-C based methods perform better provided the data generation assumption is not violated.

Finally, to tackle the problem from a generative model viewpoint, we proposed an inference method based on Bayes Theorem and mean field approximation called CorEx-Ci. Generative models have a tendency of reaching the asymptotic error rate at a higher speed than discriminative classifiers. This proved to be the case with the

proposed inference model. CorEx-Ci model tends to have a slightly higher asymptotic error. The control experiments we ran tested the potential and characteristics of CorEx-Ci in terms of sensitivity towards redundant variables as well as varying sample sizes. We found that the increase in sample size causes very small changes in the performance of CorEx-Ci. However, the increase of sample size drastically improves the performance of the classical classifiers such as SVM with Linear Kernel and Naive Bayes. This shows that CorEx-Ci is robust towards the number of sample size. We also show through our plots that SVM with Linear Kernel improves greatly when the number of redundant variables which are related to class label increases. We believe the relevant but redundant features to the class label creates a large bias to skew the classifier towards the class label. CorEx-Ci only shows very slight improvement, which indicates it being robust against the bias introduced by the redundant variables. The high error rate of CorEx-Ci is a major limitation of our work although it can also be considered a blessing as it indicates a stable capturing of relevant information regardless of the sample size.

The future direction CorEx-Ci poses is to lower the asymptotic error. One method to do so would be to incorporate additional bias or priors into the learning process of CorEx-C model. An open problem to this work is how to detect the generation process behind the dataset in order to identify if the dataset fulfils the CorEx assumption. A method to circumvent this would be to expand the assumption to include $Y \rightarrow X \rightarrow C$ data generation process. A further direction which can be explored would be to expand CorEx-C to work more discriminatively as neural networks do. With the addition of error propagation, CorEx-C has the potential performing at the level of neural networks. Unlike the neural network, it does not need as many samples and it is also more easily explainable. The method undertaken in this thesis can be expanded to other unsupervised graphical models if the class label is available. However, the ability of those models to extend to HDLSS asymptotic will have to be tested. Hence, by extending the CorEx model to incorporate the class label, we were able to extend and test its abilities under the discriminant classification model and the generative classification model. This thesis has explored and investigated classification for HDLSS datasets through learning an intrinsic structure of the data which is related to the class label. More research to expand the research in this field is required to fully mature this field.

Appendix A

Preliminary Experiment based on Autoencoder

A preliminary experiment was conducted using shallow autoencoders to test the possibility of it being used for HDLSS settings. The autoencoders were programmed using KERAS which was built using Theano backend. All experiments were seeded to produce consistent results each time it is run. The datasets used for this experiment were the Gravier dataset and Alon dataset whose properties can be found in Table (4.17).

The autoencoder used is an undercomplete autoencoder. This would mean that the number nodes in the hidden layers is lower than the number of input dimensionality. Stochastic gradient descent was used as an optimizer.

The structure of the autoencoder used is as follows:

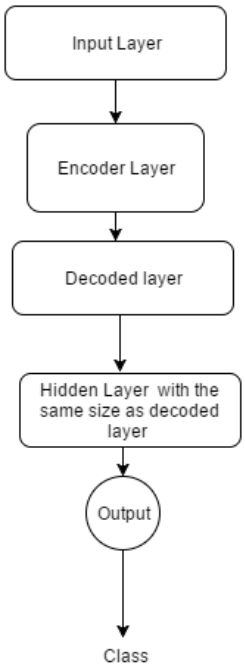


Figure A.1: The autoencoder used is a shallow autoencoder which is fitted with a neural network classifier to gain an accuracy reading. The classifier is directly attached to the decoder.

The activation functions used were Rectified Linear Units (ReLU) and the Sigmoid function.

ReLU is given by:

$$f(a) = max(0, a) \tag{A.1}$$

where $a = Wx + b$, W is the weight and b is the bias.

Sigmoid is given by:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{A.2}$$

The Table (A.1) shows that the autoencoder which uses ReLU as the activation function performs only as well as the control classifier. The use of sigmoid on the other hand allows for better accuracy compared to ReLu.

The accuracy of the Alon dataset only improves at best the same as the raw data as seen in Fig.(A.2). As seen, many parameters need to be tuned to obtain the best performance for each dataset. Shallow autoencoder requires a reasonable amount of data in order to be able to achieve a much better accuracy. Applying deep autoencoder will increase the complexity of the network but may not necessarily give improvements as large amounts of data are required.

Input size	No. of Epoch	No. of Nodes in Encoder	Activation	Accuracy
2905	1000	515	ReLu,ReLu, Sigmoid,Sigmoid	44.00%
2905	1000	1215	ReLu,ReLu, Sigmoid,Sigmoid	44.00%
2905	100	515	ReLu,ReLu, Sigmoid,Sigmoid	44.00%
2905	100	1215	ReLu,ReLu, Sigmoid,Sigmoid	44.00%
2905	50	515	ReLu,ReLu, Sigmoid,Sigmoid	44.00%
2905	50	1215	ReLu,ReLu, Sigmoid,Sigmoid	44.00%
2905	1000	515	Sigmoid,Sigmoid, Sigmoid,Sigmoid	53.65%
2905	1000	1215	Sigmoid,Sigmoid, Sigmoid,Sigmoid	53.65%
2905	100	515	Sigmoid,Sigmoid, Sigmoid,Sigmoid	53.65%
2905	100	1215	Sigmoid,Sigmoid, Sigmoid,Sigmoid	53.65%
2905	50	515	Sigmoid,Sigmoid, Sigmoid,Sigmoid	53.65%
2905	50	1215	Sigmoid,Sigmoid, Sigmoid,Sigmoid	53.65%
2905	1000	Classifier only(Control)	Sigmoid,Sigmoid	44.00%

Table A.1: Table of accuracy for the Gravier dataset based on preliminary tests using autoencoder.

Input size	No. of Epoch	No. of Nodes in Encoder	Activation	Accuracy
2000	1000	515	ReLu,ReLu, Sigmoid,Sigmoid	76.47%
2000	1000	1215	ReLu,ReLu, Sigmoid,Sigmoid	76.47%
2000	100	515	ReLu,ReLu, Sigmoid,Sigmoid	58.82%
2000	100	1215	ReLu,ReLu, Sigmoid,Sigmoid	58.82%
2000	50	515	ReLu,ReLu, Sigmoid,Sigmoid	58.82%
2000	50	1215	ReLu,ReLu, Sigmoid,Sigmoid	58.82%
2000	1000	200	Sigmoid,Sigmoid, Sigmoid,Sigmoid	76.47%
2000	1000	515	Sigmoid,Sigmoid, Sigmoid,Sigmoid	70.58%
2000	1000	1215	Sigmoid,Sigmoid, Sigmoid,Sigmoid	64.70%
2000	100	515	Sigmoid,Sigmoid, Sigmoid,Sigmoid	58.82%
2000	100	1215	Sigmoid,Sigmoid, Sigmoid,Sigmoid	58.82%
2000	50	515	Sigmoid,Sigmoid, Sigmoid,Sigmoid	58.82%
2000	50	1215	Sigmoid,Sigmoid, Sigmoid,Sigmoid	58.82%
2000	1000	Classifier only(Control)	Sigmoid,Sigmoid	76.47%

Table A.2: The table gives the performance accuracy of the stacked autoencoders for the Alon dataset based on preliminary tests using autoencoder.

References

Aapo Hyvärinen, A., 1997, "Independent component analysis by minimization of mutual information," *Helsinki University of Technology, Laboratory of Computer and Information Science, Finland, Report A* **46**

Ahn, J., 2006, *High dimensional, low sample size data analysis*, Ph.D. thesis (University of North Carolina at Chapel Hill).

Ahn, J., Lee, M. H., and Yoon, Y. J., 2012, "Clustering High Dimension , Low Sample Size," **22**, 443–464.

Ahn, J., and Marron, J., 2010, "The maximal data piling direction for discrimination," *Biometrika* **97**, 254–259.

Ahn, J., Marron, J., Muller, K. M., and Chi, Y.-Y., 2007, "The high-dimension, low-sample-size geometric representation holds under mild conditions," *Biometrika* **94**, 760–766.

Alladi, S. M., Shinde Santosh, P., Ravi, V., and Murthy, U. S., 2008, "Colon cancer prediction with genetic profiles using intelligent techniques," *Bioinformation* **3**, 130.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., and Levine, A. J., 1999, "Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays," *Proceedings of the National Academy of Sciences* **96**, 6745–6750.

Belkin, M., and Niyogi, P., 2003, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural computation* **15**, 1373–1396.

Bengio, Y., Goodfellow, I. J., and Courville, A., 2015, "Deep Learning,"

Bengio, Y., Yao, L., Alain, G., and Vincent, P., 2013, "Generalized denoising auto-encoders as generative models," *NIPS*

Bentler, P., Peter M.Dudgeon, 1996, "Covariance structure analysis: Statistical practice, theory, and directions," *Annual Review of Psychology* **47**, 563–592.

Beyer, K., Goldstein, J., Ramakrishnan, R., and Shaft, U., 1999, "When is "nearest neighbor" meaningful?." in *International conference on database theory* (Springer). pp. 217–235.

Bishop, C. M., 2006, *Pattern Recognition and Machine Learning (Information Science and Statistics)* (Springer-Verlag New York, Inc., Secaucus, NJ, USA). ISBN 0387310738

Blum, A. L., and Langley, P., 1997, "Selection of relevant features and examples in machine learning," *Artificial intelligence* **97**, 245–271.

Bolón-Canedo, V., Sánchez-Marono, N., Alonso-Betanzos, A., Benítez, J. M., and Herrera, F., 2014, "A review of microarray datasets and applied feature selection methods," *Information Sciences* **282**, 111–135.

Boser, B. E., Guyon, I. M., and Vapnik, V. N., 1992, "A training algorithm for optimal margin classi,"

Brown, G., Pocock, A., Zhao, M.-J., and Luján, M., 2012, "Conditional likelihood maximisation: a unifying framework for information theoretic feature selection," *Journal of machine learning research* **13**, 27–66.

Chandrashekar, G., and Sahin, F., 2014, "A survey on feature selection methods," *Computers & Electrical Engineering* **40**, 16–28.

Chao, G., Luo, Y., and Ding, W., 2019, "Recent advances in supervised dimension reduction: A survey," *Machine Learning and Knowledge Extraction* **1**, 341–358.

Chris Nicholson, A. G., 2016, "Introduction to deep neural networks,"

Comon, P., 1994, "Independent component analysis, a new concept?." *Signal processing* **36**, 287–314.

Cunningham, J. P., and Ghahramani, Z., 2015, "Linear dimensionality reduction: Survey, insights, and generalizations," *The Journal of Machine Learning Research* **16**, 2859–2900.

Davis, J. V., Kulis, B., Jain, P., Sra, S., and Dhillon, I. S., 2007, "Information-theoretic metric learning," in *Proceedings of the 24th international conference on Machine learning* (ACM). pp. 209–216.

Demšar, J., Curk, T., Erjavec, A., Črt Gorup, Hočevar, T., Milutinovič, M., Možina, M., Polajnar, M., Toplak, M., Starič, A., Štajdohar, M., Umek, L., Žagar, L., Žbontar, J., Žitnik, M., and Zupan, B., 2013, "Orange: Data mining toolbox in python," *Journal of Machine Learning Research* **14**, 2349–2353.

Deng, Z., Zhu, X., Cheng, D., Zong, M., and Zhang, S., 2016, "Efficient knn classification algorithm for big data," *Neurocomputing* **195**, 143–148.

Ding, S., Zhu, H., Jia, W., and Su, C., 2012, "A survey on feature extraction for pattern recognition," *Artificial Intelligence Review* **37**, 169–180.

Domeniconi, C., and Gunopulos, D., 2002, "Adaptive nearest neighbor classification using support vector machines," in *Advances in neural information processing systems*, pp. 665–672.

Dua, D., and Graff, C., 2017, "UCI machine learning repository,"

Eskandari, S., and Akbas, E., 2017, "Supervised infinite feature selection," *arXiv preprint arXiv:1704.02665*

Feng, S., Zhou, H., and Dong, H., 2019, "Using deep neural network with small dataset to predict material defects," *Materials & Design* **162**, 300–310.

Gallagher, R. J., Reing, K., Kale, D., and Steeg, G. V., 2016, "Anchored correlation explanation: Topic modeling with minimal domain knowledge," *arXiv preprint arXiv:1611.10277*

Gershenson, C., 2003, "Artificial neural networks for beginners,"

Goldberger, J., Hinton, G. E., Roweis, S. T., and Salakhutdinov, R. R., 2005, "Neighbourhood components analysis," in *Advances in neural information processing systems*, pp. 513–520.

Golub, T. R., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P., Coller, H., Loh, M. L., Downing, J. R., Caligiuri, M. A., Bloomfield, C. D., and Lander, E. S., 1999 Oct., "Molecular classification of cancer: class discovery and class prediction by gene expression monitoring.." *Science* **286**, 531–537.

Gravier, Eleonore, Pierron, G., Vincent-Salomon, A., gruel, N., Raynal, V., Savignoni, A., De Rycke, Y., Pierga, J.-Y., Lucchesi, C., Reyal, F., Fourquet, A., Roman-Roman, S.,

Radvanyi, F., Sastre-Garau, X., Asselain, B., and Delattre, O., 2010 Sep., "A prognostic DNA signature for T1T2 node-negative breast cancer patients.." *Genes, Chromosomes and Cancer* **49**, 1125–1125.

Guyon, I., and Elisseeff, A., 2003, "An introduction to variable and feature selection," *Journal of machine learning research* **3**, 1157–1182.

Guyon, I., Gunn, S., Ben-Hur, A., and Dror, G., 2005, "Result analysis of the nips 2003 feature selection challenge," in *Advances in neural information processing systems*, pp. 545–552.

Hall, P., Marron, J. S., and Neeman, A., 2005, "Geometric representation of high dimension, low sample size data," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* **67**, 427–444.

Harremoës, P., 2006, "Interpretations of renyi entropies and divergences," *Physica A: Statistical Mechanics and its Applications* **365**, 57–62.

Hinton, G., 2014, "Dropout : A Simple Way to Prevent Neural Networks from Overfitting," **15**, 1929–1958.

Hoffman, P., Grinstein, G., and Pinkney, D., 1999, "Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations," in *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management* (ACM). pp. 9–16.

Hsu, C.-W., Chang, C.-C., Lin, C.-J., *et al.*, 2003, "A practical guide to support vector classification,"

Huertas, C., and Juarez-Ramirez, R., 2016, "Automatic threshold search for heat map based feature selection: A cancer dataset analysis," *World Academy of Science, Engineering and Technology, International Journal of Computer, Electrical, Automation, Control and Information Engineering* **10**, 1341–1347.

Huo, X., Ni, X. S., and Smith, A. K., 2007, "A survey of manifold-based learning methods," *Recent advances in data mining of enterprise data*, 691–745.

Jenssen, R., 2010, "Kernel entropy component analysis," *IEEE transactions on pattern analysis and machine intelligence* **32**, 847–860.

Jing Peng, Heisterkamp, D. R., and Dai, H. K., 2002 Aug, "Adaptive kernel metric nearest neighbor classification," in *Object recognition supported by user interaction for service robots*, Vol. 3, pp. 33–36 vol.3.

John, S., 1971 apr, "Some optimal multivariate tests," *Biometrika* **58**, 123–127.

Jung, S., Marron, J. S., *et al.*, 2009, "Pca consistency in high dimension, low sample size context," *The Annals of Statistics* **37**, 4104–4130.

Karpathy, A., 2016, "Cs231n convolutional neural networks for visual recognition,"

Khalid, S., Khalil, T., and Nasreen, S., 2014, "A survey of feature selection and feature extraction techniques in machine learning," in *Science and Information Conference (SAI), 2014* (IEEE). pp. 372–378.

Klement, S., Mamlouk, A. M., and Martinetz, T., 2008, "Reliability of Cross-Validation for SVMs in High-Dimensional , Low Sample Size Scenarios," , 41–50.

Kulis, B., *et al.*, 2013, "Metric learning: A survey," *Foundations and Trends® in Machine Learning* **5**, 287–364.

Kuntal, B. K., Ghosh, T. S., and Mande, S. S., 2014, "Igloo-plot: a tool for visualization of multidimensional datasets," *Genomics* **103**, 11–20.

Le, Q. V., Ngiam, J., Chen, Z., Chia, D., Koh, P. W., and Ng, A. Y., 2010, "Tiled convolutional neural networks,"

Ledoit, O., and Wolf, M., 2002, "Some hypothesis tests for the covariance matrix when the dimension is large compared to the sample size," *Annals of Statistics*, 1081–1102.

Leiva-Murillo, J. M., and Artés-Rodríguez, A., 2007, "Maximization of mutual information for supervised linear feature extraction," *IEEE Transactions on Neural Networks* **18**, 1433–1441.

Li, B., Li, Y.-R., and Zhang, X.-L., 2019, "A survey on laplacian eigenmaps based manifold learning methods," *Neurocomputing* **335**, 336–351.

Li, J., Cheng, K., Wang, S., Morstatter, F., Robert, T., Tang, J., and Liu, H., 2016, "Feature selection: A data perspective," *arXiv:1601.07996*

Li, J., Cheng, K., Wang, S., Morstatter, F., Trevino, R. P., Tang, J., and Liu, H., 2017, "Feature selection: A data perspective," *ACM Computing Surveys (CSUR)* **50**, 94.

Liang, J., Hou, L., Luan, Z., and Huang, W., 2019, "Feature selection with conditional mutual information considering feature interaction," *Symmetry* **11**, 858.

Liang, P., and Jordan, M. I., 2008, "An asymptotic analysis of generative, discriminative, and pseudolikelihood estimators," in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08 (ACM, New York, NY, USA). pp. 584–591.

Lin, D., and Tang, X., 2006, "Conditional infomax learning: an integrated framework for feature extraction and fusion," *Computer Vision–ECCV 2006*, 68–82.

Liu, B., Wei, Y., Zhang, Y., and Yang, Q., 2017a, "Deep neural networks for high dimension, low sample size data.." in *IJCAI*, pp. 2287–2293.

Liu, H., Motoda, H., Setiono, R., and Zhao, Z., 2010, "Feature selection: An ever evolving frontier in data mining," in *Feature Selection in Data Mining*, pp. 4–13.

Liu, J., Wang, X., Cheng, Y., and Zhang, L., 2017b, "Tumor gene expression data classification via sample expansion-based deep learning," *Oncotarget* **8**, 109646.

Mahmud, M. S., Fu, X., Huang, J. Z., and Masud, M. A., 2018, "High-dimensional limited-sample biomedical data classification using variational autoencoder," in *Australasian Conference on Data Mining* (Springer). pp. 30–42.

Maldonado, S., López, J., and Vairetti, C., 2019, "An alternative smote oversampling strategy for high-dimensional datasets," *Applied Soft Computing* **76**, 380–389.

Marron, J. S., Todd, M. J., and Ahn, J., 2007, "Distance-Weighted Discrimination," *Journal of the American Statistical Association* **102**, 1267–1271.

Mauchly, J. W., 1940, "Significance Test for Sphericity of a Normal n-Variate Distribution," *The Annals of Mathematical Statistics* **11**, 204–209.

McHugh, M. L., 2013, "The chi-square test of independence," *Biochemia medica: Biochemia medica* **23**, 143–149.

Miao, J., and Niu, L., 2016, "A survey on feature selection," *Procedia Computer Science* **91**, 919–926.

Nagao, H., 1973, "On some test criteria for covariance matrix," *The Annals of Statistics* **1**, 700–709.

National, C., and Recherche, D., 1986, "Ultrametricity for physicists," **58**

Nenadic, Z., 2007, "Information discriminant analysis: Feature extraction with an information-theoretic objective," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **29**, 1394–1407.

Ng, A., 2011, "CS294A Lecture Notes Sparse Autoencoder," , 1–19.

Ng, A. Y., and Jordan, M. I., 2002, "On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes," in *Advances in neural information processing systems*, pp. 841–848.

Nguyen, B., Morell, C., and De Baets, B., 2018, "Scalable large-margin distance metric learning using stochastic gradient descent," *IEEE Transactions on Cybernetics*, 1–12.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E., 2011, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research* **12**, 2825–2830.

Peng, H., Long, F., and Ding, C., 2005, "Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-redundancy," *IEEE Transactions on pattern analysis and machine intelligence* **27**, 1226–1238.

Pepke, S., and Ver Steeg, G., 2017, "Comprehensive discovery of subsample gene expression components by information explanation: therapeutic implications in cancer," *BMC medical genomics* **10**, 12.

Qi, G.-J., Tang, J., Zha, Z.-J., Chua, T.-S., and Zhang, H.-J., 2009, "An efficient sparse metric learning in high-dimensional space via l1-penalized log-determinant regularization," in *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09 (ACM, New York, NY, USA). pp. 841–848.

Qiao, X., 2013, "Flexible High-dimensional Classification Machines and Their Asymptotic Properties,"

Raducanu, B., and Dornaika, F., 2012, "A supervised non-linear dimensionality reduction approach for manifold learning," *Pattern Recognition* **45**, 2432–2444.

Ribeiro, B., Vieira, A., and das Neves, J. C., 2008, "Supervised isomap with dissimilarity measures in embedding learning," in *Iberoamerican Congress on Pattern Recognition* (Springer). pp. 389–396.

Rossi, R. A., and Ahmed, N. K., 2015, "The network data repository with interactive graph analytics and visualization," in *AAAI*

Roweis, S. T., and Saul, L. K., 2000, "Nonlinear dimensionality reduction by locally linear embedding," *science* **290**, 2323–2326.

Saeys, Y., Inza, I., and Larrañaga, P., 2007, "A review of feature selection techniques in bioinformatics," *bioinformatics* **23**, 2507–2517.

Shen, D., Shen, H., and Marron, J. S., 2013, "Consistency of sparse PCA in High Dimension , Low Sample Size contexts," *Journal of Multivariate Analysis* **115**, 317–333.

Silverman, B. W., 1986, *Density estimation for statistics and data analysis*, Vol. 26 (CRC press).

Sordo, M., and Zeng, Q., 2005, "On sample size and classification accuracy: A performance comparison," in *Biological and Medical Data Analysis*, edited by Oliveira, J. L., Maojo, V., Martín-Sánchez, F., and Pereira, A. S. (Springer Berlin Heidelberg, Berlin, Heidelberg). pp. 193–201.

Storcheus, D., Rostamizadeh, A., and Kumar, S., 2015, "A survey of modern questions and challenges in feature extraction," in *Feature Extraction: Modern Questions and Challenges*, pp. 1–18.

Tenenbaum, J. B., De Silva, V., and Langford, J. C., 2000, "A global geometric framework for nonlinear dimensionality reduction," *science* **290**, 2319–2323.

Thirumuruganathan, S., 2010, "A detailed introduction to k-nearest neighbor (knn) algorithm,"

Torkkola, K., 2003, "Feature extraction by non-parametric mutual information maximization," *Journal of machine learning research* **3**, 1415–1438.

Tu, Z., 2007, "Learning generative models via discriminative approaches," in *2007 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE). pp. 1–8.

Ver Steeg, G., and Galstyan, A., 2014a, "Discovering structure in high-dimensional data through correlation explanation," in *Advances in Neural Information Processing Systems 27*, edited by Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. D., and Weinberger, K. Q. (Curran Associates, Inc.). pp. 577–585.

Ver Steeg, G., and Galstyan, A., 2014b, "Discovering structure in high-dimensional data through correlation explanation," in *Advances in Neural Information Processing Systems*, pp. 577–585.

Ver Steeg, G., and Galstyan, A., 2015, "Maximally informative hierarchical representations of high-dimensional data," in *Artificial Intelligence and Statistics*, pp. 1004–1012.

Wang, L., Wang, Y., and Chang, Q., 2016, "Feature selection methods for big data bioinformatics: A survey from the search perspective," *Methods* **111**, 21–31.

Xue, B., Zhang, M., Browne, W. N., and Yao, X., 2016, "A survey on evolutionary computation approaches to feature selection," *IEEE Transactions on Evolutionary Computation* **20**, 606–626.

Xue, J., Chan, P. P. K., and Hu, X., 2017 July, "Experimental study on stacked autoencoder on insufficient training samples," in *2017 International Conference on Wavelet Analysis and Pattern Recognition (ICWAPR)*, pp. 223–229.

Yang, J., Yu, X., Xie, Z.-Q., and Zhang, J.-P., 2011, "A novel virtual sample generation method based on gaussian distribution," *Knowledge-Based Systems* **24**, 740–748.

Yang, L., and Jin, R., 2006, "Distance metric learning: A comprehensive survey," *Michigan State Universiy* **2**, 4.

Yata, K., and Aoshima, M., 2010, "Effective PCA for high-dimension , low-sample-size data with singular value decomposition of cross data matrix," *Journal of Multivariate Analysis* **101**, 2060–2077.

Yuan, X.-T., and Hu, B.-G., 2009, "Robust feature extraction via information theoretic learning," in *Proceedings of the 26th annual international conference on machine learning* (ACM). pp. 1193–1200.

Zhang, L., and Lin, X., 2011, "Some considerations of classification for high dimension low-sample size data," doi:\bibinfo{doi}{10.1177/0962280211428387}

Zhang, S.-q., 2009, "Enhanced supervised locally linear embedding," *Pattern Recognition Letters* **30**, 1208–1218.

Zheng, F., Chen, N., and Li, L., 2008, "Semi-supervised laplacian eigenmaps for dimensionality reduction," in *2008 International Conference on Wavelet Analysis and Pattern Recognition*, Vol. 2 (IEEE). pp. 843–849.