Feature Selection With Multi Layer Perceptron Using Approximate L_0-norm and Redundancy Control

A thesis submitted for the partial fulfillment of
the conditions for the award of the degree **M.Tech. Computer Science.**

by

Ankit Varshney
CS1710

Supervised by
Prof. Nikhil R. Pal
Electronics and Communication Sciences Unit

Indian Statistical Institute, Kolkata, India

July, 2019

M.Tech. (Computer Science) Thesis Completion Certificate

Student: Ankit Varshney (CS1710)
Topic: Feature Selection With Multi Layer Perceptron Using Approximate L_0-norm and Redundancy Control
Supervisior: Prof. Nikhil R. Pal

This is to certify that the thesis titled **Feature Selection With Multi Layer Perceptron Using Approximate L_0-norm and Redundancy Control** submitted by **Ankit Varshney** in partial fulfillment for the award of the degree of Master of Technology is a bonafide record of work carried out by him under my supervision. The thesis has fulfilled all the requirements as per the regulations of this Institute and, in my opinion, has reached the standard needed for submission. The results contained in this thesis have not been submitted to any other university for the award of any degree or diploma.

Prof. Nikhil R. Pal	Date

To my family and the professors of ISI. . .

Acknowledgements

First, I would like to thank my dissertation supervisor, Prof. Nikhil R. Pal, for accepting my request to work with him and his constant support and guidance. I also want to thank him for encouraging me to work hard. I want to thank all my labmates and seniors of Computational Intelligence Lab for there valuable suggestions and comments.

Abstract

In this thesis, we develop three new methods for feature selection with Multi Layer Perceptron(MLP) neural networks. In each method, we use a two-step approach. First, we train a MLP network for a given dataset. Second, we introduce feature selector variables and form an optimization problem based on some penalty on these feature selector variables and some measure of redundancy. Then, we optimize the problem using gradient descent method to find nearly optimal values of the feature selector variables while keeping the weights of the MLP networks fixed. First method, which we call Feature Selection with MLP using Approximate L_0-norm and Global Redundancy Control (FSMLP-AL-GRC) uses penalty based on an approximate L_0-norm and global redundancy, i.e., redundancy that is calculated with features values, without considering the class information. For second method, we first define a new redundancy measure that uses class label information while calculating redundancy, we call it class-level redundancy. This method make use of class level redundancy measure along with an approximate L_0-norm based penalty. We call it Feature Selection with MLP using Approximate L_0-norm and Class-level Redundancy Control (FSMLP-AL-CRC). Last method is a variant of method two. Here, we replace each feature selector variable with some non-linear bounded function that always lies between 0 and 1, this function act as feature attenuating gates. We call this method Gated Feature Selection with MLP using Class-level Redundancy Control (Gated-FSMLP-CRC). We test these methods experimentally on different data sets. We also present results on Sonar data using method Gated-FSMLP-CRC without keeping the weights of MLP fixed during the learning process.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Modern technologies have made it possible to assess objects from different perspectives at the same time. This causes the generation of high dimensional datasets, which increase the time and memory requirements as well as increase difficulties of analyzing the data. Different features may have different effects on data analysis - it may have positive or negative influences [9]. We often face this problem while classifying high dimensional datasets. Here, feature selection can be very useful, which is the process of choosing a smaller subset of *"good/useful"* features from a larger set of features [11]. Feature selection not only helps in handling large datserts but it also preserve the original semantics of data, thereby offering interoperability by domain expert [5].

Feature selection methods are usually classified into three categories: filter methods, wrapper methods, and embedded methods. In filter methods, feature selection is performed by some characteristic of data and the procedure is independent of any classifier. On the other hand, in wrapper methods, feature selection is dependent on the performance of a classifier. This makes wrapper methods time-consuming. In embedded methods, feature selection is performed with learning of the classifier in an integrated fashion.

Generally, finding an optimal set of features requires an exhaustive search on all subsets of features, which can be computationally impossible to do in real applications. Even in wrapper methods, some sub optimal heuristics are used to look at fewer subsets through a guided search. Using forward and backward selection type methods may not be very useful as they do not consider the interaction between features properly. Hence, in our view integrated approach can be proved to be most useful, these methods do not require you to search all possible subsets and they can consider the interaction between features during the selection procedure. Our proposed methods also work with this philosophy.

Before going to more details and chapters, we would like to discuss different types of features and what kind of feature we like to select. We want to select useful features and discard features with poor prediction power or features that can confuse the learning process. Keeping this goal in mind, we can classify features into four groups [3]: 1) essential features— these features are necessary irrespective of the modeling tool that we use; 2) bad or derogatory features—these features must be discarded irrespective of the modeling tool that we use; 3) indifferent features—these features neither help nor cause any problem in decision making; and 4) redundant features— these are useful features, which are dependent on each other, such as two highly correlated features. Thus, all of the redundant features are not necessary; only some are needed to solve the problem.

Our proposed methods come under the framework of Maximum Relevance and Minimum Redundancy-type methods as it try to select most relevant features while controlling the level of redundancy [7]. Removing irrelevant and redundant features will decrease feature numbers and enhance the performances of classifiers [13]. Other than this, feature selection methods can also be divided into two categories- Unsupervised and Supervised feature selection methods. The term unsupervised is used for approaches where only feature values are used during feature selection procedure. On the other hand supervised feature selection methods use extra information such as class label information. This work mainly focuses on supervised feature selection. recently semi-supervised feature selction methods have been introduced, which uses combination of labeled and unlabeled data [15].

1.1 Problem Statement

The objective of this work is to propose embedded methods of feature selection using multi layer perceptron. The proposed methods should select a smaller subset of relevant features with low redundancy while maintaining the performance, i.e., the accuracy of MLP.

1.2 Outline of This Thesis

In Chapter 2, we discuss some related works. Chapter 3 presents the proposed methods. In Chapter 4, we describe the experiments to validate this work. We conclude in Chapter 5.

Chapter 2

Related Work

Multi-layer perceptrons (MLPs) are frequently used as classifiers because of there high discriminating capabilities. Various embedded feature selection schemes with MLPs [4] have also been proposed. We will be dicussing few methods for feature selection grouped according to the philosophy.

2.1 Unsupervised approaches

The unsupervised approach as discussed by the authors in [1] uses a greedy method for feature selection. Let $\mathcal{S}$ be the set of indices of the selected r features and X be the data matrix of size $n \times f$. The authors have used a novel feature selection criterion by introducing a minimization problem using function $F(\mathcal{S})$, $F(\mathcal{S}) = ||X - P^{\mathcal{S}}X||_F^2$, where $||.||_F$ is the Frobenius norm and $P^{\mathcal{S}}$ is an $n \times n$ projection matrix which projects the columns of X onto the span of the columns corresponding to the selected features. We want to select a set S with dimension r for which $F(\mathcal{S})$ is minimized.

In [2], authors proposed a method to select r features with controlled redundancy such that the topology of original dataset can be maintained in the reduced dimension. This method uses Sammon's error as a measure of preservation of topology. In this method, extra feature selector variables have been introduced corresponding to each feature. Let $X \in \mathbb{R}^{n \times p}$ be the dataset, i.e., $X = \{\mathbf{x_1}, \mathbf{x_2}, \ldots, \mathbf{x_n}\}$

where $\forall i \in \{1, 2 \dots n\}, \mathbf{x_i} = (x_{i1}, x_{i2}, \dots, x_{ip})^T$. We want to select r features and let $\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_p)$ be extra introduced feature selector variables. Selection of features is done by optimizing a function, E, with respect to $\boldsymbol{\beta}$s, E is sum of four terms as,

$$E = SE + B \times PF_1 + C \times PF_2 + D \times PF_3$$

with

$$SE = \frac{1}{\sum_{i<j}} \sum_{i<j} \frac{(d^*_{ij} - d_{ij})^2}{d^*_{ij}}$$

$$PF_1 = \frac{1}{(p-r)^2} \left(\sum_{i=1}^{p} e^{-\beta_i^2} - r \right)^2$$

$$PF_2 = \frac{4}{p} \sum_{i=1}^{p} e^{-\beta_i^2} (1 - e^{-\beta_i^2})$$

$$PF_3 = \frac{1}{p(p-1)} \sum_{i=1}^{p} e^{-\beta_i^2} \sum_{j \neq i} e^{-\beta_j^2} \rho_{ij}^2$$

where, $B, C, D \in \mathbb{R}_+$ are coefficients of penalty terms; d^*_{ij} is Euclidean distance between $\mathbf{x_i}$ and $\mathbf{x_j}$; $d_{ij} = \sqrt{\sum_{k=1}^{p} e^{-2\beta_k^2} (x_{ik} - x_{jk})^2}$ and ρ_{ij} is the dependency between features i and j. After minimization E by gradient descent method, using the values of $e^{-\beta_i^2}, i = 1, 2 \dots p$ and some threshold, features are selected.

2.2 Sparsity based approaches

For instance, in [10] the authors introduced an extra set of variables that plays the role of feature selectors. Their method is divided into two main steps. First, they trained an MLP on a given dataset, and then, they introduced a variable corresponding to each feature to form a minimization problem as described below.

$$\underset{\boldsymbol{\beta}}{\text{minimize}} \left\{ \frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(\mathbf{t}_i, (x_{i1} \times \beta_1, x_{i2} \times \beta_2, \dots, x_{iF} \times \beta_F)^T) + \lambda \sum_{i=1}^{F} |\beta_i| \right\} \tag{2.1}$$

Here, $\boldsymbol{\beta} = (\beta_1, \beta_2, \dots, \beta_F)^T \in \mathbb{R}^F$ is the set of feature selectors; F is the dimension of the data; n is the is the nuumber of samples; $\mathcal{L}(\cdot)$ is the loss function of the MLP;

$\mathbf{t}_i \in \{0,1\}^F$ is the target vector for the ith sample; $\mathbf{x}_i = (x_{i1}, x_{i2}, \dots, x_{iF}) \in \mathbb{R}^F$ is the ith training sample; and $\lambda \in \mathbb{R}_+$ is a regularizing coefficient. For a given threshold $\tau \in \mathbb{R}$, the ith feature is selected, if $|\beta_i| > \tau$. After selecting features this process is repeated on reduced set of features until stopping criteria is met. The current work indeed is inspired by this work [10].

In [8], the authors discussed an approximate L_0-norm minimization method for compressed sensing by solving the following problem:

$$\begin{aligned} &\underset{\mathbf{x}}{\text{maximize}} \;\; \mathcal{F}(\mathbf{x}) \\ &\text{subject to} \;\; \mathbf{y} = A\mathbf{x} \\ &\text{where} \;\; \mathcal{F}(\mathbf{x}) = \sum_{i=1}^{n} e^{-x_i^2/2a^2}, \end{aligned} \tag{2.2}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the d dimensional signal vector, $\mathbf{y} \in \mathbb{R}^{d'}$ is the d' dimensional reconstructed signal vector, A is a $d' \times d$ transformation matrix, and $a \rightarrow 0$ is a very small quantity. Therefore, the following hold:

$$\lim_{a \rightarrow 0} \mathcal{F}(\mathbf{x}) = \sum_{i=1}^{n} I[x_i = 0] = \sum_{i=1}^{n} (1 - I[x_i \neq 0]) = n - L_0(\mathbf{x}), \tag{2.3}$$

where $I[\cdot]$ is the indicator function. In this fashion, the authors [8] used $(n - \mathcal{F}(\mathbf{x}))$ as an approximation of the L_0-norm of $\mathbf{x}$, i.e., $L_0(\mathbf{x})$. We extend this idea in the current work for a smooth approximation of L_0-norm in the penalty function for selecting features.

2.3 Controlled Redundancy based approaches

In [6], authors proposed a two-stage feature selection approach based on mutual information. They have used the maximum relevance and minimum redundancy as the selection criterion. They first select the feature that has the highest mutual information value with the class labels. Then, from the remaining set, their intention is to choose the second feature that has the highest relevance with the class labels (i.e., mutual information) as well as the minimum dependency with the already selected features. So, at any step, if already $m-1$ features are chosen forming the set S_{m-1}, then from the remaining set $F - S_{m-1}$, they choose the next feature by optimizing the following criterion:

$$\underset{x_j \in F - S_{m-1}}{\text{maximize}} \left(MI(x_j, c) - \frac{1}{m-1} \sum_{x_i \in S_{m-1}} MI(x_i, x_j) \right)$$

Here, F is the set of features. Thus, this scheme selects one feature at a time. Given number of features to be selected say r , it selects the feature subsets $S_1, S_2, \ldots, S_r$. From the selected sets, choose the set that yields better cross-validation accuracy with a classifier.

In [4], the authors proposed a method of feature selection with MLP using a penalty that considers redundancy of features. Rather than introducing linear unbounded variables, they multiplied the value of each feature by a nonlinear gate function $f(\cdot) \in [0,1]$. They minimized the following total error (TE):

$$TE = \frac{1}{n} \sum_{i=1}^{n} ||\mathbf{t}_i - \mathcal{M}(\mathbf{x}_i)||_2^2 + \frac{\lambda}{F(F-1)} \sum_{i=1}^{F} f(\beta_i) \sum_{\substack{j=1 \\ j \neq i}}^{F} f(\beta_j) \phi(X_i, X_j) \tag{2.4}$$

where $\mathcal{M}(\cdot) \in [0,1]^c$ is the output from the MLP, c is the number of classes, $\phi(\cdot)$ is the dependency between two features, and X_f denotes the fth feature. Here, $f(\cdot)$ can be chosen as $f(\beta_i) = e^{-\beta_i^2}$ or $f(\beta_i) = 1/(1+e^{-\beta_i})$. We note here, that $\phi(x_i, x_j)$ will be computed using X_i and X_j, where $X_i \in \mathbb{R}^F$ is the vector containing the ith feature value of all training data points. Hence, for notational convenience in place of $\phi(x_i, x_j)$, we should use $\phi(X_i, X_j)$. In the current work, we also extend this concept [4] of redundancy and define a new redundancy measure, and then, use that for feature selection.

Chapter 3

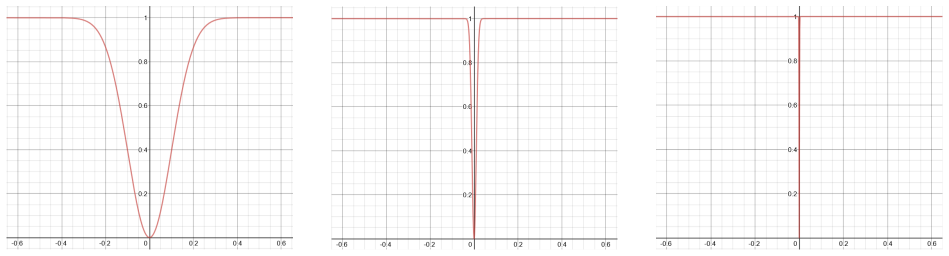
The Proposed Methods

In this Chapter, first, we discuss smooth approximation of L_0-norm that we have used in first two methods. Than we will discuss each method one by one with neccessary details and problem formulation.

3.1 A Smooth Approximation of L_0-norm

Generally, embedded feature selection methods that use neural networks, use L_1-norm or L_2-norm on feature selector variables to reduce the total number of selected features [12], [16]. But when L_0-norm based penalty is used, it provides us with a good sparse subset of features, as it directly reduces the number of selected features by making more feature selector variable zero. L_1-norm or L_2-norm are used because L_2-norm is differentiable and L_1-norm is continuous whereas L_0-norm is not even continuous and there is an issue in using L_0-norm based penalties - L_0-norm based penalty minimization is NP-hard in nature, even for simple classification losses [14]. We may, however, use a smooth approximation of L_0-norm, and then, may apply a gradient-based method for optimization. With this philosophy, we use a smooth approximation of L_0-norm as a regularizer.

The L_0-norm of a d dimensional vector $\mathbf{w} = (w_1, w_2, \ldots, w_d)^T$, i.e., $L_0(\mathbf{w})$ is defined



(a) $a = 0.1$ (b) $a = 0.01$ (c) $a = 0.001$

Figure 3.1: Plot of $\left(1 - e^{-x^2/2a^2}\right)$ with different values of a.

as

$$L_0(\mathbf{w}) = \sum_{i=1}^{d} I[w_i \neq 0]. \tag{3.1}$$

Now, we define a smooth approximation of $L_0(\mathbf{w})$, denoted by $L_0^a(\mathbf{w})$ as follows:

$$L_0^a(\mathbf{w}) = \sum_{i=1}^{d} (1 - e^{-w_i^2/2a^2}) \tag{3.2}$$

The approximation in (3.2), is dependent on the variable a and its behavior for different values of a is shown in Fig. 3.1. Note that, as $a \to 0$, $L_0^a(\mathbf{w}) \to L_0(\mathbf{w})$. The explanation behind this is as follows:

$$\lim_{a \to 0} L_0^a(\mathbf{w}) = \sum_{i=1}^{n} \lim_{a \to 0} (1 - e^{-w_i^2/2a^2}) = \sum_{i=1}^{n} I[w_i \neq 0] = L_0(\mathbf{w}) \tag{3.3}$$

3.2 Method 1, Feature Selection with MLP using Approximate L_0-norm and Global Redundancy Control (FSMLP-AL-GRC)

Let $\mathcal{X} = \{\mathcal{X}_i, \mathcal{X}_j, \mathcal{X}_k\}$ be a set of three features, such that, $\mathcal{X}_i$, $\mathcal{X}_j$, and $\mathcal{X}_k$ are strongly *related* to (dependent on) each other. Dependency may either be linear or nonlinear in nature. Here, The term *"related"*, means that if we know the value of any of the features, we can predict the values of the remaining two features. Consequently, if we have any one of the three features, we do not need the other two. $\mathcal{X}$, therefore, is a set of redundant features.

In this method we want to select features that reduce the number of sets of redundant features and hence reduce overall redundancy in the system. We also want to keep useful features, i.e., features that are relevant for the classification purpose. Hence, this method comes under the framework of Maximum-Relevance Minimum-Redundancy. We will use the redundancy measure that is used by authors in [4] as discussed in chapter 2, with slight modification and with combination of equation (3.2).

Let $X = \{X_1, X_2, \ldots, X_F\}$ denotes input data and $\mathbf{y} = \{y_1, y_2, \ldots, y_n\}$ denotes the set of class label for each sample. Then using $(X, \mathbf{y})$ we train a MLP neural network. Let the MLP take an input a vector of size F. We calculate loss between output of MLP and target vector, for example cross-entropy loss, mean-square loss etc. Let this loss function denoted by $\mathcal{L}(.)$. The optimization function that we have proposed as a part of method 1 is as,

$$\begin{aligned} E_{\lambda,\mu}(\beta_1, \beta_2, \ldots, \beta_F) =& \frac{1}{n}\sum_{i=1}^{n} \mathcal{L}(\mathbf{t}_i, (x_{i1} \times \beta_1, x_{i2} \times \beta_2, \ldots, x_{iF} \times \beta_F)^T) \\ &+ \frac{\mu}{F}\sum_{i=1}^{F}\left(1 - e^{-\beta_i^2/2a^2}\right) \\ &+ \frac{\lambda}{F(F-1)}\sum_{i=1}^{F}\beta_i^2\left(\sum_{\substack{j=1 \\ j\neq i}}^{F}\beta_j^2\phi(X_i, X_j)\right) \end{aligned} \tag{3.4}$$

Optimal values of feature selectors are,

$$\beta^* = \arg\min_{\beta} E_{\lambda,\mu}(\beta_1, \beta_2, ..., \beta_F)$$

From the value of β^* we can select features, i.e., low value of any component denotes rejection of the corresponding feature. We will refer this model as Feature Selection with MLP using Approximate L_0-norm and Global Redundancy Control (FSMLP-AL-GRC). We used the term global redundancy control here because as we can see that the redundancy measure $\phi(.)$ only considers feature values without considering the local correspondence to different classes. But we can use class label information while calculating redundancy and by this motivation we will now discuss our 2^{nd} proposed method.

3.3 Method 2, Feature Selection with MLP using Approximate L_0-norm and Class-level Redundancy Control (FSMLP-AL-CRC)

Let for feature X_i and class c, X_i^c denotes the data of feature X_i that corresponds/belongs to class c, thus $X_i = \cup_{c=1}^{C} X_i^c$. Then, Class-level redundancy is defined as,

The Class-level redundancy between feature X_i and X_j is defined as,

$$\phi_{CL}(X_i, X_j) = \frac{1}{C} \sum_{c=1}^{C} \phi(X_i^c, X_j^c)$$

where C denotes total number of classes.

Now, how to use this idea of redundancy to develop an appropriate optimization function for feature selection? Here, if we want to use this redundancy measure we need to have some extra variable that also tells us that which part within a feature is important or not, as some feature may be redundant for one class but may not be redundant for the another class. Here the objective is not to just reject features but to get some class specific information.

So, we have introduced a feature selector matrix $\beta_{C\times F}$ where C is total number of classes and F is total number of features. And, each element of this matrix say β_{cf}, denotes the status of X_f^c, whether this part of dataset is relevant or not.

Using this matrix $\beta_{C\times F}$ and loss function $\mathcal{L}(.)$, we develop a new optimization function that is a combination of different ideas that was discussed earlier. The optimization function is defined as,

$$E_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\beta_{C\times F}) = \frac{1}{C}\sum_{i=1}^{C}\frac{1}{n_i}\sum_{j=1}^{n_i}\mathcal{L}(\mathbf{t}_i,(x_1^{ij}\times\beta_{i1},x_2^{ij}\times\beta_{i2},\dots,x_F^{ij}\times\beta_{iF})^T)$$
$$+\frac{1}{C}\sum_{i=1}^{C}\frac{\mu_i}{F}\sum_{j=1}^{F}\left(1-e^{-\beta_{ij}^2/2a^2}\right)$$
$$+\frac{1}{C}\sum_{i=1}^{C}\frac{\lambda_i}{F(F-1)}\sum_{j=1}^{F}\beta_{ij}^2\sum_{\substack{k=1\\k\neq j}}^{F}\beta_{ik}^2\phi(X_j^i,X_k^i) \quad (3.5)$$

where n_i denotes number of samples in class i, $\mathbf{t}_i \in \{0,1\}^F$ is the target vector for the ith class, x_k^{ij} represents value of feature X_k that belongs to j^{th} sample of class i data, $\boldsymbol{\mu} = (\mu_1,\mu_2,\dots,\mu_C) \in \mathbb{R}^C$ and $\boldsymbol{\lambda} = (\lambda_1,\lambda_2,\dots,\lambda_C) \in \mathbb{R}^C$ vectors of regularizing coefficients.
Optimal values of feature selector matrix is,

$$\beta_{C,F}^* = \arg\min_{\beta_{C,F}} E_{\boldsymbol{\lambda},\boldsymbol{\mu}}(\beta_{C\times F})$$

After getting $\beta_{C,F}^*$, we will select the features for which its corresponding column contain at least one non-zero value. Because it shows its high relevance for at least one class. We will call this framework, Feature Selection with MLP using approximate L_0-norm and Class-level redundancy control (FSMLP-AL-CRC). Next we, will extensively compare our two proposed method, i.e., FSMLP-AL-GRC and FSMLP-AL-CRC. This comparison is necessary as the first method uses already known redundancy measure and second one uses our proposed redundancy measure.

3.4 Global vs Class-level Redundancy Control

In this section, we will compare various properties of FSMLP-AL-GRC model, as given by equation (3.4) and FSMLP-AL-CRC model, as given by equation (3.5). We will be mainly considering the Pearson's correlation coefficient for comparison purposes but the ideas can be extended to some non-linear dependency measure such as mutual information. For simplicity we will only consider the effect of redundancy terms, i.e., last term in equations (3.4) and (3.5), because we are only discussing the effect of global redundancy and class-level redundancy here. Hence, consider $\forall i \in \{1, 2 \ldots F\}, \mu_i = 0 = \mu$ in equations (3.4) and (3.5). Now we compare these methods and make the following remarks:

Class-level redundancy framework is as good as global redundancy framework for handling highly (globally) correlated features:

As, equation (3.5) contains more free variables, so we can take $\lambda_1 = \lambda_2 = ... = \lambda_C = \lambda$ and $\forall j \in \{1, 2 \ldots F\}, \beta_{1,j} = \beta_{2,j} = ... = \beta_{C,j} = \beta_j$. So, (3.4) and (3.5) mainly differ in the dependency measure as one uses global and another uses class-level redundancy. But, next we will discuss, the effect of class-level is as good as global for highly globally correlated features.

Let feature X_i and X_j are highly linearly correlated features that means $|\rho(X_i, X_j)| \approx 1$ i.e., $\exists a \neq 0$, $X_i \approx aX_j + d$. Now, due to this $\forall c \in \{1, 2 \ldots C\}, X_i^c \approx aX_j^c + d$ and hence, $|\rho(X_i^c, X_j^c)| \approx 1$. So,

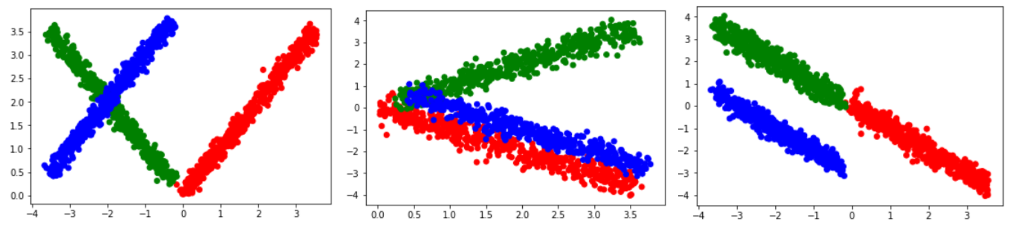
$$\phi_{CL}(X_i, X_j) = \frac{1}{C}\sum_{c=1}^{C} \phi(X_i^c, X_j^c) = \frac{1}{C}\sum_{c=1}^{C} |\rho(X_i^c, X_j^c)| \approx \frac{1}{C}\sum_{c=1}^{C} 1 = 1$$

This means, if in the global redundancy control framework a highly correlated feature has tendency to get rejected, then this tendency will be preserved in class-level redundancy control framework.

There exists datasets for which global redundancy control framework doesn't work but class-level does:

Consider a three dimensional data set with features $\{X_1, X_2, X_3\}$, various views of data are shown in figure 3.2 and figure 3.3, where different colors represent different classes.

As, we can see only feature X_1 and X_3 are enough for 100% accurate classification,



(a) X_1-vs.-X_2 (b) X_2-vs.-X_3 (c) X_1-vs.-X_3

Figure 3.2: Pair-wise view of the features in $\{X_1, X_2, X_3\}$

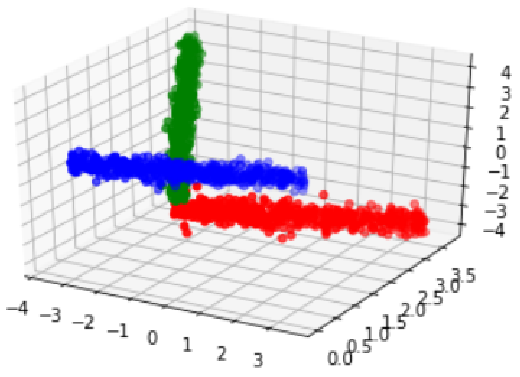


Figure 3.3: Three-dimensional view of the synthetic data

see figure 3.2(b). Global redundancy between different pairs is as,

$$|\rho(X_1, X_2)| = 0.08590776$$

$$|\rho(X_2, X_3)| = 0.19083253$$

$$|\rho(X_3, X_1)| = 0.72072376$$

Suppose, if we use global redundancy control framework for this dataset. As feature X_2's correlation with X_1 and X_3 are low, so we have to choose large regularizing coefficient to eliminate X_2 but then X_3 may also get removed as its correlation with X_1 is high. We have tested it for various coefficient of penalty term and tried to choose top 2 features but feature X_2 always gets selected and feature X_3 always gets rejected. Where as, class-level redundancy control framework works very well and chooses features (X_1, X_3) because here class-level redundancy for all pairs is approximately equal, so it does not favor any feature. The class-level redundancies is shown Table 3.1. Hence, only loss based penalty is effective and helps in rejecting feature X_2. We have tested it and it is found to be true experimentally.

Table 3.1: Different class-level redundancies for synthetic data

	c=0 (Red class)	c=1 (Green class)	c=2 (Blue class)
$\lvert\rho(X_1^c, X_2^c)\rvert$	0.99083612	0.98955383	0.98866423
$\lvert\rho(X_2^c, X_3^c)\rvert$	0.9515664	0.9470847	0.9584851
$\lvert\rho(X_3^c, X_1^c)\rvert$	0.96333222	0.95429516	0.96633191

In global redundancy framework, one of the highly correlated features can be removed without affecting the performance of MLP but this is not true in class-level redundancy framework:

We will show that if two features are highly globally related than we can remove one of them without affecting output of MLP, whereas this is not true for class-level redundancy. Suppose two features X_i and X_j are highly linearly correlated i.e., $|\rho(X_i, X_j)| = 1$ or $\exists a \neq 0, d, X_i = aX_j + d$. Consider any node k in the first hidden layer of the MLP. The output of this node is $\sigma(x_i w_{ik} + x_j w_{jk} + b_k + e)$, where σ is some activation function and e is remaining term of the net. Now, as $x_i = ax_j + d$, this output becomes $\sigma(x_j(aw_{ik} + w_{jk}) + d + b_k + e)$. We can consider $aw_{ik} + w_{jk}$ as new weight w'_{jk} and $d + b_k$ as new bias b'_k to form $\sigma(x_j w'_{jk} + b'_k + e)$. These new weights can be learned by the MLP. This essentially shows that if two features are highly globally correlated than one of them is not useful in influencing the output of the MLP for classification.

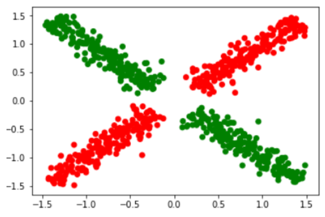


Figure 3.4: View of X_1-vs.-X_2

The above is not true for highly class-level related features. Consider the example of a two dimensional dataset $\{X_1, X_2\}$, as shown in Figure 3.4, where the red and green colors represent different classes. Linear class-level redundancy between X_1

and X_2 is high, $|\rho(X_1^{red}, X_2^{red})| = 0.99$ and $|\rho(X_1^{green}, X_2^{green})| = 0.99$, but as we can see in Figure 3.4, no feature alone is sufficient for the classification task.

3.5 Benefits of FSMLP-AL-CRC method

Their are various benefits of the method/model described using equation (3.5), we will discuss some of them below:

- As, we have already discussed in the previous sections, that this new method with newly proposed redundancy measure is as good as the popularly used and studied global redundancy control. And even on some data-sets it works better as demonstrated in Section 3.4 by a synthetic data-set.
- Minimizing equation (3.5), we get the feature selector matrix $\beta_{C \times F}$. We can answer various questions with this matrix. Suppose we want to understand for a particular class which features are important then we can look at the row in $\beta_{C \times F}$ that corresponds to this class and look for non-zero values of β_{cf}. This can be very important for some applications. For example, suppose for a particular cancer class, we want to understand features relevant to this particular class. Using FSMLP-AL-CRC framework we can get features that are particularly relevant to this cancer class. Previously it was not possible as for all classes we have same set of selected features, their is no difference at the class level.
- We can select top $k < F$ features from each class. In this case we can get representative for each class. This increases the discrimination power as compared to global redundancy control framework because previously in global mechanism if we choose top k features or if use some threshold to select features then we might get lots of features that are important to one class and might not get any feature that is important for other class.

3.6 Method 3, Gated Feature Selection with MLP using Class-level Redundancy Control (Gated-FSMLP-CRC)

We can use the clever idea of penalty with non-linear gate function as discussed by authors in [2] and combine it to idea presented in method 2 to get a new optimization function as,

$$\begin{aligned} E_{\lambda,\mu}(\beta_{C\times F}) = \frac{1}{C}\sum_{i=1}^{C}\frac{1}{n_i}\sum_{j=1}^{n_i}\mathcal{L}(\mathbf{t}_i, (x_1^{ij}\times e^{-\beta_{i1}^2}, x_2^{ij}\times e^{-\beta_{i2}^2}, \ldots, x_F^{ij}\times e^{-\beta_{iF}^2})^T) \\ + \frac{1}{C}\sum_{i=1}^{C}\frac{\mu_i}{F}\sum_{j=1}^{F}e^{-\beta_{ij}^2}(1-e^{-\beta_{ij}^2}) \\ + \frac{1}{C}\sum_{i=1}^{C}\frac{\lambda_i}{F(F-1)}\sum_{j=1}^{F}e^{-\beta_{ij}^2}\sum_{\substack{k=1\\k\neq j}}^{F}e^{-\beta_{ik}^2}\phi(X_j^i, X_k^i) \end{aligned} \quad (3.6)$$

And we want to minimize it to get,

$$\boldsymbol{\beta}^*_{\boldsymbol{C\times F}} = \underset{\beta_{C\times F}}{\arg\min}\, E_{\lambda,\mu}(\beta_{C\times F})$$

First and last term in equation (3.6) are easy to explain as we have just replaced unrestricted linear β_{ij} variable with restricted non-linear function $e^{-\beta_{ij}^2}$, which always lies between 0 and 1. This is called gate function as it behaves like a gate for a feature, 1 indicate selection and 0 indicate rejection of that part of feature.

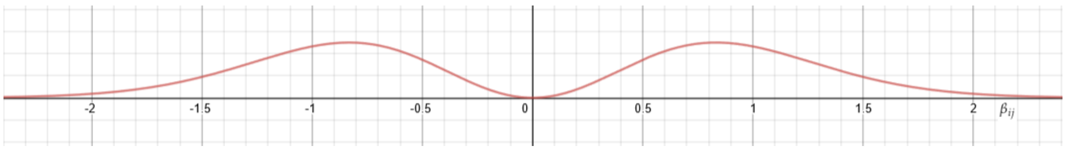


Figure 3.5: The curve of $e^{-\beta_{ij}^2}(1-e^{-\beta_{ij}^2})$

Now, we try to understand the use of second term. Consider the plot of this term in figure 3.5. So, this term will help to make either absolute value of β_{ij} high or low; for high β_{ij}, $e^{-\beta_{ij}^2}$ will be low (approx. 0) and for low β_{ij} it will be high (approx. 1). This represents selection and rejection in a better way. In this model rather than looking at the value of β_{ij} we will look at $e^{-\beta_{ij}^2}$ for feature selection. We will call the method given by equation (3.6) as Gated-FSMLP-CRC.

3.7 Learning rule

We will use gradient descent method for updating values of βs with each iteration using the following learning rule,

3.7.1 For FSMLP-AL-GRC

Consider the error $E_{\lambda,\mu}$ in equation (3.4), then learning rule for $\boldsymbol{\beta}$ with learning rate η is as,
$\forall l \in [F]$,

$$\beta_l = \beta_l - \eta \frac{\partial E}{\partial \beta_l} \tag{3.7}$$

where

$$\frac{\partial E}{\partial \beta_l} = \frac{1}{n}\sum_{i=1}^{n} x_{il}\mathcal{L}'(\mathbf{t}_i, (x_{i1} \times \beta_1, x_{i2} \times \beta_2, \dots, x_{iF} \times \beta_F)^T) + \frac{\mu\beta_l}{a^2F}e^{-\beta_i^2/2a^2} + \frac{2\lambda\beta_l}{F(F-1)}\sum_{\substack{j=1 \\ j\neq i}}^{F}\beta_j^2\phi(X_i, X_j)$$

Here, let I_l represents lth input to MLP than,

$$\mathcal{L}'(\mathbf{t}_i, (x_{i1} \times \beta_1, x_{i2} \times \beta_2, \dots, x_{iF} \times \beta_F)^T) = \left.\frac{\partial \mathcal{L}}{\partial I_l}\right|_{(\mathbf{t}_i,(x_{i1}\times\beta_1, x_{i2}\times\beta_2,\dots,x_{iF}\times\beta_F)^T)}$$

3.7.2 For FSMLP-AL-CRC

Consider the error $E_{\lambda,\mu}$ in equation (3.5), then learning rule to update βs with learning rate η is as,
$\forall l \in [C], m \in [F]$,

$$\beta_{lm} = \beta_{lm} - \eta \frac{\partial E}{\partial \beta_{lm}} \tag{3.8}$$

where

$$\frac{\partial E}{\partial \beta_{lm}} = \frac{1}{n_l C}\sum_{j=1}^{n_l} x_m^{lj}\mathcal{L}'(\mathbf{t}_l, (x_1^{lj}\times\beta_{l1}, x_2^{lj}\times\beta_{l2},\ldots,x_F^{lj}\times\beta_{lF})^T) + \frac{\mu_l\beta_{lm}}{a^2FC}e^{-\frac{\beta_{lm}^2}{2a^2}} + \frac{2\lambda_l\beta_{lm}}{F(F-1)C}\sum_{\substack{k=1\\k\neq m}}^{F}\beta_{lk}^2\phi_{CL}(X_m^l, X_k^l)$$

Here, if I_m represents m^{th} input to MLP, then

$$\mathcal{L}'(\mathbf{t}_l, (x_1^{lj}\times\beta_{l1}, x_2^{lj}\times\beta_{l2},\ldots,x_F^{lj}\times\beta_{lF})^T) = \left.\frac{\partial\mathcal{L}'}{\partial I_m}\right|_{(\mathbf{t}_l,(x_1^{lj}\times\beta_{l1},x_2^{lj}\times\beta_{l2},\ldots,x_F^{lj}\times\beta_{lF})^T)}$$

3.7.3 For Gated-FSMLP-CRC

Similarly, consider the error $E_{\lambda,\mu}$ in equation (3.6), then learning rule to update βs with learning rate η is as,

$\forall l \in [C], m \in [F]$,

$$\beta_{lm} = \beta_{lm} - \eta\frac{\partial E}{\partial \beta_{lm}} \tag{3.9}$$

where

$$\frac{\partial E}{\partial \beta_{lm}} = \frac{-1}{n_l C}\sum_{j=1}^{n_l}\left(2x_{lj}\beta_{lm}e^{-\beta_{lm}^2}\mathcal{L}'(\mathbf{t}_i, (x_1^{ij}\times e^{-\beta_{i1}^2}, x_2^{ij}\times e^{-\beta_{i2}^2},\ldots,x_F^{ij}\times e^{-\beta_{iF}^2})^T)\right) - \frac{2\mu_l\beta_{lm}}{FC}e^{-\beta_{lm}^2} + \frac{4\mu_l\beta_{lm}}{FC}e^{-2\beta_{lm}^2} - \frac{2\lambda_l\beta_{lm}e^{-\beta_{lm}^2}}{F(F-1)C}\sum_{\substack{k=1\\k\neq m}}^{F}e^{-\beta_{lk}^2}\phi_{CL}(X_m^l, X_k^l)$$

Here, if I_m represents m^{th} input to MLP, then

$$\mathcal{L}'(\mathbf{t}_i, (x_1^{ij}\times e^{-\beta_{i1}^2}, x_2^{ij}\times e^{-\beta_{i2}^2},\ldots,x_F^{ij}\times e^{-\beta_{iF}^2})^T) = \left.\frac{\partial\mathcal{L}}{\partial I_m}\right|_{(\mathbf{t}_i,(x_1^{ij}\times e^{-\beta_{i1}^2},x_2^{ij}\times e^{-\beta_{i2}^2},\ldots,x_F^{ij}\times e^{-\beta_{iF}^2})^T)}$$

Chapter 4

Results

For all our experiments, we use just one hidden layer with a fixed number of hidden nodes for each dataset. To make our results reliable, we use two level stratified cross-validation mechanism, except for SRBCT dataset. We have used the following scheme. In the outer level, first, we randomly partition the data into five folds with equal number of data points with respect to each class (to the extent possible), $X = X_1 \cup X_2 \ldots X_5$, $X_i \cap X_j = \emptyset, \forall i \neq j$. One of the folds, say X_j , is kept out for testing. While, the remaining four folds' data, $Y = \cup_{i \neq j} X_i$, are now used for selection of features as well as for designing a network to test the effectiveness of the selected features on the data left out in the outer loop, i.e., on X_j. This is repeated for all $j = 1, 2 \ldots 5$.

In the inner loop, we use only Y and perform two tasks. First, we use it to train the MLP. We divide Y into five folds as $Y = Y_1 \cup Y_2 \ldots Y_5$. We use one fold as validation data say Y_j and rest for training the MLP. We keep the weights until the epoch where we get maximum accuracy on Y_j. We repeat this $\forall j \in \{1, 2, 3, 4, 5\}$ and keep the MLP which has the maximum validation accuracy among these five folds. We perform feature selection with data Y. After the features are selected, we project Y on the selected feature space and let us call the projected version of Y as Y'. To assess how good these selected features are, we train MLP using the selected features, i.e., using the data set Y', as we trained before feature selection. Now, we test this MLP on X'_j, where X'_j is the projected version of X_j that was left out in the outer loop. This process is repeated $\forall j \in \{1,2,3,4,5\}$ in the outer loop to get the accuracy using the selected features. Finally, the entire process can be repeated few times, we repeated twice, every time using a different random partition in the outer loop. We report the average accuracy. For all the methods, we have used stochastic gradient descent rather than updating βs in one iteration with complete data.

4.1 Performance on different datasets

We have generated results for following datasets: Iris (4 features, 3 classes, 150 samples), WDBC (30 features, 2 classes, 569 samples), SRBCT (2308 features, 4 classes, 83 samples), Glass (9 features, 6 classes, 214 samples) and Sonar(60 features, 2 classes, 208 samples). During generation of all results we have fixed the number of hidden nodes in MLP, which is written in brackets with name of the data set.

4.1.1 For FSMLP-AL-GRC

This method uses the equation (3.4), we fixed $a = 0.35$. First, each component in $\boldsymbol{\beta}$ is assigned 1 initially, we select top features by looking at highest values in $\boldsymbol{\beta}$. The penaly factors, features selected(S), average train(Tr) and test accuracies(Te) according to the experimentation procedure are summarized in Table 4.1.

If we look at the results in Table 4.1, for example for Glass data with coefficient $\mu = 3$ and $\lambda = 20$, we get approximately the same test accuracy with nearly all features (eight features) and with four features (approx. 50%). And for SRBCT data only 10 features (out of 2308 features) are enough to get more than 95% accuracy. For the WDBC data when we use L_0-norm, in four of the six cases the test performance is better. Similar is the case for the SRBCT data set.

4.1.2 For FSMLP-AL-CRC

This method uses the equation (3.5). We have fixed $a = 0.35$, keep $\forall i \in \{1,2\ldots C\}, \lambda_i = \lambda$ and $\forall i \in \{1,2\ldots C\}, \mu_i = \mu$. First, each value in $\beta_{C\times F}$ is assigned 1 initially, after

Table 4.1: Results for method FSMLP-AL-GRC

DataSet	μ	λ	**S**	**Tr**	**Te**
Iris Data(3)	0	7	1	91.4	89.7
			2	96.5	92.8
	1	5	1	92.8	89.5
			2	95.4	94.6
WDBC(10)	0	15	1	94.5	88.4
			2	92.5	90.4
			4	96.5	92.8
			6	94.7	91.3
			8	95.3	94.6
			10	98.2	96.4
	1	10	1	92.5	87.7
			2	94.6	89.5
			4	96.5	94.4
			6	96.8	92.3
			8	94.7	96.9
			10	97.1	98.3
Sonar(15)	0	50	1	66.2	48.6
			2	74.5	71.4
			4	79.2	69.4
			6	80.4	74.2
			8	88.7	90.6
			10	84.3	85.4
	4	40	1	68.3	52.4
			2	77.3	68.4
			4	72.4	75.3
			6	85.4	77.9
			8	90.1	88.6
			10	92.2	89.3
SRBCT(10)	0	250	1	44.5	33.33
			2	67.2	66.5
			4	90.3	88.8
			6	95.4	85.3
			8	96.9	85.9
			10	99.2	95.8
	10	200	1	32.6	39.4
			2	54.8	60.3
			4	92.5	90.2
			6	96.7	85.8
			8	98.2	95.3
			10	98.4	98.5
Glass(15)	0	30	1	45.2	46.7
			2	66.5	60.1
			4	80.2	68.8
			6	84.9	72.6
			8	84.2	70.2
	3	20	1	55.3	42.8
			2	62.1	61.3
			4	75.8	68.1
			6	83.7	70.1
			8	81.2	68.3

feature selection procedure, we select a few top features by looking at highest values in each row corresponding to a class. The union of these features for all classes will be set of selected features for the dataset. The penaly factors, how many top features we select per class, the average number of total feature selected(S), average train(Tr) and test accuracies(Te) for different data sets are summarized in Table 4.2.

Results in Table 4.2 suggest that for SRBCT data we can get 100% training accuracy and more than 95% testing accuracy using only 10 features. Table 4.2 revels that the total number of features selected is 4 (number of classes) times the number of top features we select from each class. This means that there is no overlap between the sets of features selected for different classes. This suggests that different classes are characterized by different sets of features. This is natural, but global feature selector methods cannot reveal such information. We can see this clearly by plotting the data of selected features, we will discuss this nature of selected features in the next section.

4.1.3 For Gated-FSMLP-CRC

This method uses the equation (3.6), we keep $\forall i \in \{1, 2 \ldots C\}, \lambda_i = \lambda$ and $\forall i \in \{1, 2 \ldots C\}, \mu_i = \mu$. First, each value in $\beta_{C \times F}$ is assigned 0.1 initially, as it makes Gate variable open i.e., $e^{-\beta_{ij}} = 0.99 \approx 1$. We select features with respect to each class by looking at values in each row corresponding to a class, such that $e^{-\beta_{ij}} > 0.9$. Union of these features for all classes will be set of selected features for the dataset. The average of total number of feature selected(S), the average train(Tr) and test accuracies(Te) for different values of coefficient of penalty are shown in Table 4.3.

For WDBC data set results suggest that approximately 9 features are enough to get more than 95% testing accuracy. For SRBCT data only 7 features (out of 2308 features) are enough to get 95% testing accuracy whereas with 2104 features we get 99% test accuracy, ie., only 4% reduction in test accuracy with more than 99% reduction in total number of selected features.

Table 4.2: Results for method FSMLP-AL-CRC

DataSet	μ	λ	**TOP**	**S**	**Tr**	**Te**
Iris Data(3)	0	15	1	2.3	95.3	92.5
			2	3.2	96.3	95.7
	1	10	1	2.1	96.3	94.3
			2	3.8	98.2	93.7
WDBC(10)	0	30	1	1.7	90.4	86.3
			2	3.4	95.7	90.3
			4	6.9	96.5	92.8
			6	10.2	96.6	93.9
			8	11.8	97.3	94.2
			10	14.3	98.9	97.3
	2	20	1	1.8	92.3	88.6
			2	3.2	93.5	91.5
			4	6.1	96.5	92.1
			6	9.7	97.5	93.8
			8	11.9	98.7	95.3
			10	13.9	99.1	93.4
Sonar(15)	0	100	1	2	70.3	62.4
			2	4	78.3	66.8
			4	7.4	92.2	81.3
			6	11.4	80.2	71.6
			8	15.6	84.3	84.6
			10	17.3	87.2	93.2
	4	50	1	2.3	71.4	60.3
			2	3.6	76.5	70.2
			4	7.1	79.2	76.4
			6	10.9	84.3	78.3
			8	14.7	89.2	84.6
			10	18.3	90.1	80.4
SRBCT(10)	0	5000	1	4	95.5	90.4
			2	8	98.5	93.5
			4	16	99.5	90.2
			6	24	100	93.7
			8	31.7	99.8	95.6
			10	40	100	97.4
	1	2000	1	4	94.5	93.8
			2	8	95.5	94.5
			4	16	98.3	96.3
			6	24	100	97.2
			8	32	100	98.1
			10	40	100	97.6
Glass(15)	0	20	1	4.9	64.4	53.2
			2	7.6	70.3	66.2
			4	8.2	76.8	73.8
			6	8.4	84.2	74.5
			8	9	78.7	69.3
	5	10	1	4.8	65.4	58.2
			2	7.8	72.4	62.4
			4	8.1	77.1	68.8
			6	8.4	82.3	74.5
			8	8.8	80.2	70.3

Table 4.3: Results for method Gated-FSMLP-CRC

DataSet	μ	λ	**S**	**Tr**	**Te**
Iris Data(5)	1	2	3.8	98.6	98.4
	1	5	3.1	97.4	96.3
	1	7	2.4	94.9	92.5
	1	10	1.1	95.4	90.5
WDBC(10)	1	10	9.1	96.5	95.3
	1	20	7.2	94.9	92.6
	2	30	5.3	94.6	89.5
	5	50	2.9	92.3	87.1
Sonar(15)	2	20	4.3	82.1	76.5
	3	25	2.9	72.6	66.7
	2	30	2.2	70.2	68.5
	2	50	0.8	54.6	44.5
SRBCT(10)	5	50	2106.4	99.6	99.5
	5	500	985.6	98.4	96.1
	5	5000	110.6	100	97.5
	3	100000	6.7	99.3	95.4
Glass(15)	1	5	8.6	84.3	72.1
	1	7	7.1	82.5	73.2
	2	7	6.4	76.8	70.5
	1	10	3.2	70.2	66.7

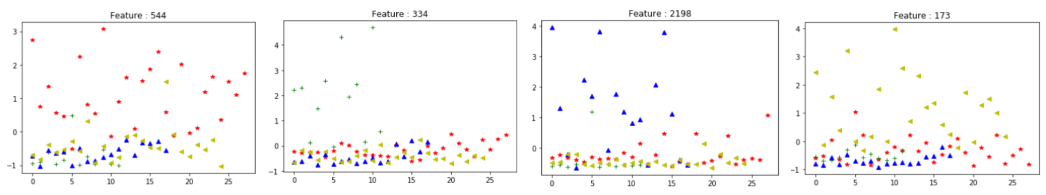
We have also generated results on one data set (Sonar) with a modification in the learning algorithm. Previously, we keep weights of MLP fixed while we learn values of feature selecting gates but for result in Table 4.4, we also update weights of MLP together with feature selector variables in each iteration using stochastic gradient descent. These results generated using modified learning process shows that for Sonar data, we get only 4% reduction in testing accuracy by 80% reduction in total number of features selected.

Table 4.4: Results for Gated-FSMLP-CRC without keeping weights of MLP fixed

DataSet	μ	λ	**S**	**Tr**	**Te**
Sonar(15)	5	500	0	-	-
	5	100	4.3	71.3	68.2
	5	50	11.4	79.4	80.7
	5	10	50.8	88.2	84.2

4.2 Nature of selected features

We have manually plotted some selected features to understand the nature of the selection. For example consider the Figure 4.1. Feature set $\{173, 334, 544, 2198\}$ is selected for the SRBCT data by applying the method FSMLP-AL-CRC and selecting top one feature from each class according to feature selector variables. So we got good representative features for each class. The discriminating power of these features is very high as each represent or help discriminating one class.

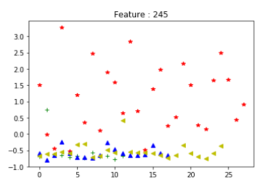


(a) Feature for Class 1 (b) Feature for Class 2 (c) Feature for Class 3 (d) Feature for Class 4

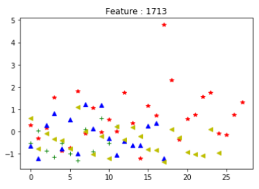
Figure 4.1: Plot of representative features for different classes as selected by class-level redundancy framework, i.e., using method FSMLP-AL-CRC on SRBCT dataset

Now, consider set of top four features selected using method FSMLP-AL-GRC, i.e., $\{245, 1713, 1861, 1910\}$ for SRBCT dataset as shown in Figure 4.2. As we can see it does not select features that represent all four classes. Features 245, 1910 have good discriminating power for first(red) class, fourth(yellow) class respectively. Feature 1861 has moderate discriminating power for third(blue) class but none of four features has good discriminating power for third(green) class. It may happen that the interaction of these features may lead to good accuracy and can discriminate all classes. But as compared to class-level redundancy framework, global redundancy framework may not give good discriminating features for each class with less number of selected features.

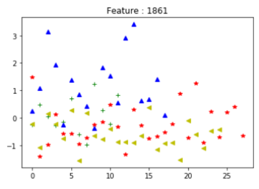
Similarly, consider WDBC dataset. It contains only two classes, so each feature that is good for discriminating one automatically becomes good for others. So, there must exists cases where only one feature is selected for both classes. During experimentation we found such a feature (feature number 22), see plot in Figure 4.3.



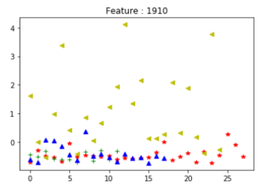
(a) Feature 245



(b) Feature 1713



(c) Feature 1861



(d) Feature 1910

Figure 4.2: Plot of top four features selected using global redundancy framework, i.e., using method FSMLP-AL-GRC on SRBCT dataset

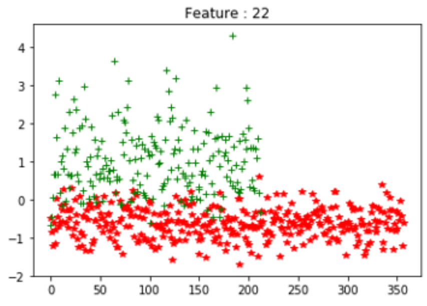


Figure 4.3: Plot of representative feature for both classes of WDBC dataset

Chapter 5

Conclusion and Future Scope

In this thesis, we presented three different feature selection methods that comes in the framework of Maximum Relevance and Minimum Redundancy method. We first described various methods available and how we got the motivation for this work. We than explained the first method that uses approximate L_0-norm based penalty in combination of global redundancy based penalty. Next, we define a new way of dealing with redundancy and incorporate it to give method two that also considers the class-level redundancy. We gave an extensive comparison of this new redundancy control framework, i.e., class-level, with the global redundancy framework. We have explained different benefits of using class-level redundancy. After this we gave a third method which is a variant of method two. This method helps in differentiating the selected and rejected feature set properly by using gate functions, for which the required threshold is easy to find. We discussed the updating rules for all three methods. Then we presented results generated by two level cross validation mechanism on five datsets.

In future, we plan to modify the algorithm in such a way that rather than keeping fixed value of a in approximation of L_0-norm, its value can be changed with iterations or according to the values of βs. We can try to give a general method for assigning values to different hyper parameters like number of hidden nodes in MLP before or after feature selection, various coefficients of penalties etc. We can try to study the convergence of proposed methods.

Bibliography

[1] A. K. FARAHAT, A. G., AND KAMEL, M. S. An efficient greedy method for unsupervised feature selection. *IEEE 11th International Conference on Data Mining* (2011), 161–170.

[2] BANERJEE, M., AND PAL, N. R. Unsupervised feature selection with controlled redundancy (ufescor). *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING 27*, 12 (2015), 3390–3403.

[3] CHAKRABORTY, D., AND PAL, N. R. Selecting useful groups of features in a connectionist framework. *IEEE Transactions on Neural Networks 19*, 3 (2008), 381–396.

[4] CHAKRABORTY, R., AND PAL, N. R. Feature selection using a neural framework with controlled redundancy. *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 26*, 1 (2015), 35–50.

[5] CHEN, X., YUAN, G., WANG, W., NIE, F., CHANG, X., AND HUANG, J. Z. Local adaptive projection framework for feature selection of labeled and unlabeled data. *IEEE Transactions on Neural Networks and Learning Systems 29*, 12 (Dec 2018), 6362–6373.

[6] H. PENG, F. L., AND DING, C. Feature selection based on mutual information of criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 8 (2005), 1226–1238.

[7] HANCHUAN PENG, FUHUI LONG, AND DING, C. Feature selection based on mutual information criteria of max-dependency, max-relevance, and min-

redundancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence 27*, 8 (Aug 2005), 1226–1238.

[8] HYDER, M., AND MAHATA, K. An approximate l0-norm minimization algorithm for compressed sensing. *IEEE International Conference on Acoustics, Speech and Signal Processing* (2009), 3365–3368.

[9] J. GUI, Z. SUN, S. J. D. T. T. T. Feature selection based on structured sparsity: A comprehensive study. *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 28*, 7 (2017), 1490–1507.

[10] KAI SUN, SHAO-HSUAN HUANG, D. S.-H. W., AND JANG, S.-S. Design and application of a variable selection method for multilayer perceptron neural network with lasso. *IEEE TRANSACTIONS ON NEURAL NETWORKS AND LEARNING SYSTEMS 28*, 6 (2017), 1386–1396.

[11] L. WANG, Y. WANG, Q. C. Feature selection methods for big data bioinformatics: A survey from the search perspective. *Methods 111* (2016), 21–31.

[12] LI, F., ZURADA, J. M., LIU, Y., AND WU, W. Input layer regularization of multilayer feedforward neural networks. *IEEE Access 5* (2017), 10979–10985.

[13] LIU, T., WU, G., YU, J., GUO, Y., WANG, Y., SHI, Z., AND CHEN, L. A mrmrmsrc feature selection method for radiomics approach. In *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)* (July 2017), pp. 616–619.

[14] LIU, Z., AND LI, G. Efficient regularized regression with l0-penalty for variable selection and network construction. *Computational and Mathematical Methods in Medicine 2016* (2016).

[15] XU, J., TANG, B., HE, H., AND MAN, H. Semisupervised feature selection based on relevance and redundancy criteria. *IEEE Transactions on Neural Networks and Learning Systems 28*, 9 (Sep. 2017), 1974–1984.

[16] ZHAO, L., HU, Q., AND WANG, W. Heterogeneous feature selection with multi-modal deep neural networks and sparse group lasso. *IEEE Transactions on Multimedia 17*, 11 (Nov 2015), 1936–1948.