MAIA

Classification of Alzheimer's Disease using MRIs and Gene Expression Data
Statistical Learning and Data Mining (91940)

Sheikh Adilina
Università degli Studi di Cassino e del Lazio Meridionale
sheikh.adilina@studentmail.unicas.it
Registration Number : 0063769

1 Introduction

For the challenge 2, we were given three binary classification problems to identify if the patient iscognitively normal (CN) or has mild cognitive impairment (MCI) or the Alzheimer's Disease (AD). We were asked to train a classification model for each of the problem and test it on the corresponding test dataset. The main task was to choose the classifier that achieves the best performance classification results on the test datasets.

2 Materials Used

We received three train datasets (ADCNtrain.csv, ADMCItrain.csv, MCICNtrain.csv) and three corresponding testdatasets (ADCNtest.csv, ADMCItest.csv, MCICNtest.csv). The first column of each of the datasets has ('Subject_id') indicating the ID of each observation. The last column is 'Label', which was the target of the datasets. The first column of the testing dataset is the same as the training dataset, however, the testing dataset we received did not contain the 'Label' column. The rest of the columns are feature that have already been extracted from MRIs and Gene Expression.

3 Pre-Processing

Below I have explained the techniques that I tried on the datset before passing thme into the classification models:

1. Firstly, I deleted the first column which was the 'Subject_id'.
2. Second, I used the variance inflation factor (or VIF) to identify the predictors that have co-linearity problem. Then I deleted all those predictors from the dataset. The function is shown in Figure 1a.
3. Lastly, I used correlation matrix to find out the predictor with correlation more than 75% and then I deleted those predictors. The function is shown in Figure 1b.
4. Even after removing so many features, some models were still overfitting due to the huge number of predictors compared to the total number of data. So, I carried out the lasso regression feature selection as it makes the coefficients of the less important predictors to become zero hence eliminating them.

In addition to the above mentioned methods, I also tried to carry out feature selection using Principal Component Analysis (PCA) but before doing that I studied the pattern of the features. After plotting

several principal component plots, I found out that the features with most variance are projected in such a way that they are inseparable and hence none of the classification algorithms will be able to classify them properly. So I decided not to use PCA. Some examples of PCA plots for each dataset are given in Figures: 3 (ADCN Dataset), 4 (ADMCI Dataset), 5 (MCICN Dataset).

```
########### Co-Linearity ###########
computing_colinearity <- function(x) {
  colinearity <- vifcor(x)
  colinearity
}
```

(a) Function to compute the VIF

```
########### Correlation Matrix ###########
computing_correlation_matrix <- function(x) {
  cor_mat <- cor(train_data_adcn)
  index <- findCorrelation(cor_mat, .75)
  to_be_removed <- colnames(cor_mat)[index]
  x <- x[!names(x) %in% to_be_removed]
  dim(x)
  return(x)
}
```

(b) Function to compute the Correlation Matrix

4 Classifiers Used

In this report, I have reported the experimental results of K Nearest Neighbour, Linear Discriminant Analysis, Logistic Regression and Support Vector Machine. The functions I wrote for each of the classifiers are shown in Figure: 2. For each classifier I centered and scaled the data and fitted the data in the model. 10-fold cross validation was carried out to obtain the training result. The same centering and scaling of data was done on the test dataset while obtaining the prediction.

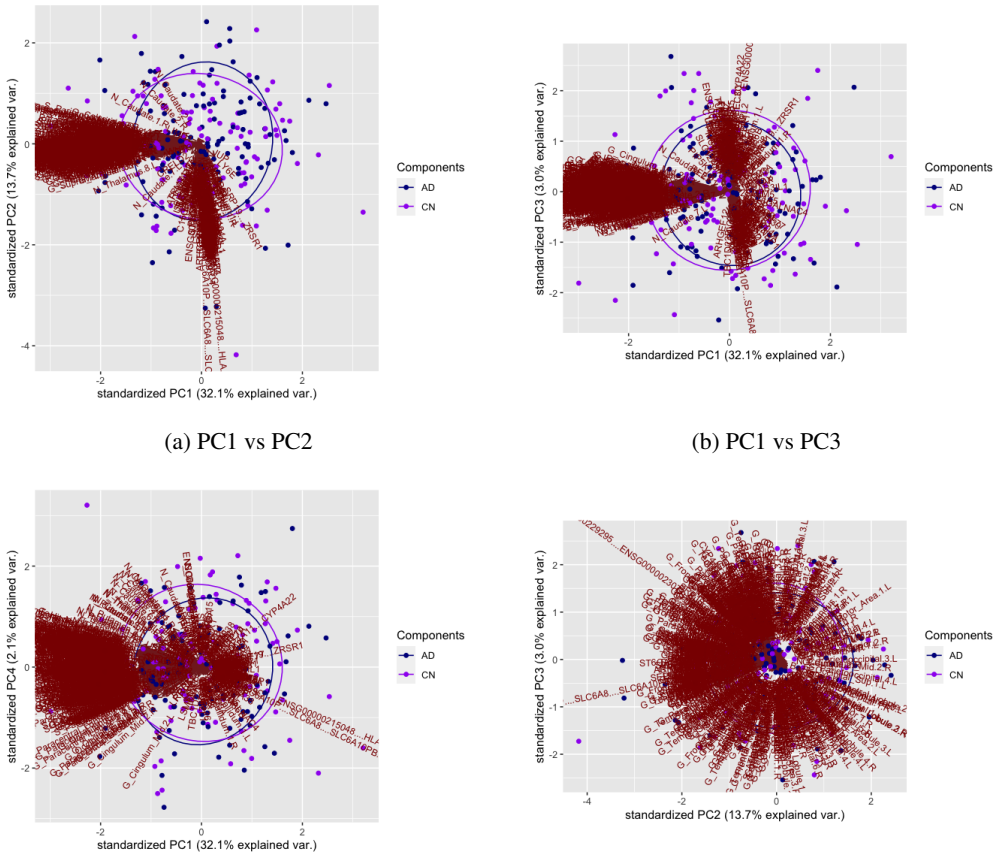
```
########### Training the model and printing evaluation score ###########

ctrl <- trainControl(method="cv", number = 10)
######### FITTING K NEAREST NEIGHBOUR #########
fitting_knn_model <- function(x){
  knn_fit <- train(Labels ~ ., data = x, method="knn", trControl = ctrl,
                   preProcess = c("center", "scale"))
  return(knn_fit)
}

######### FITTING LINEAR DISCRIMINATIVE ANALYSIS #########
fitting_lda_model <- function(x){
  lda_fit <- train(Labels ~ ., data = x, method="lda", trControl = ctrl,
                   preProcess = c("center", "scale"))
  return(lda_fit)
}

######### FITTING LOGISTIC REGRESSION #########
fitting_lr_model <- function(x){
  lr_fit <- train(Labels ~ ., data = x, method="glm", trControl = ctrl,
                   preProcess = c("center", "scale"))
  return(lr_fit)
}
######### FITTING SUPPORT VECTOR MACHINE #########
fitting_svm_model <- function(x){
  svm_fit <- train(Labels ~ ., data = x, method="svm", trControl = ctrl,
                   kernel = 'linear', preProcess = c("center", "scale"))
  return(svm_fit)
}
```

Figure 2: Function to Fit the Models



(a) PC1 vs PC2

(b) PC1 vs PC3

(c) PC1 vs PC4

(d) PC2 vs PC3

Figure 3: Plots of combination of multiple PCA components on the ADCN Dataset

5 ADCN Dataset

In this section, I will explain how I preprocessed, trained and evaluated the ADCN Dataset.

5.1 Pre-Processing and Feature Selection

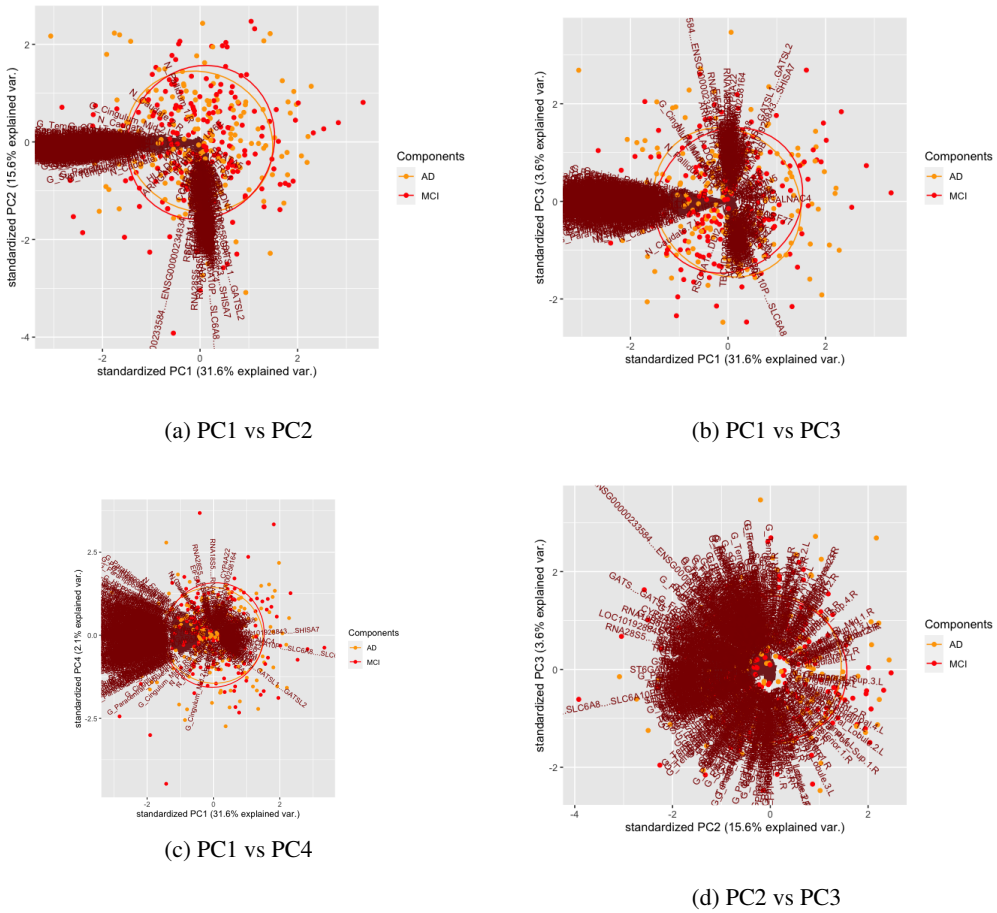
1. The result of the VIF index showed that among 566 predictors, 33 of the input variables have co-linearity problem. I removed all the 33 predictors from the training set.
2. After deleting the correlated features (as mentioned in Section 3) I had 287 predictors left.
3. After carrying out lasso regression feature selection (as mentioned in Section 3) I had 39 predictors left. The convergence of the predictors is shown in Figure 6a

5.2 Results

The best performance was obtained by using Support Vector Machine (SVM) classifier. I did not choose Logistic Regression (LR) as the best classifier even when the values are higher than SVM is because LR is clearly overfitting the data. Table:1 shows the performance of all the classifiers on the training dataset.

6 ADMCI Dataset

In this section, I will explain how I preprocessed, trained and evaluated the ADMCI Dataset.



(a) PC1 vs PC2

(b) PC1 vs PC3

(c) PC1 vs PC4

(d) PC2 vs PC3

Figure 4: Plots of combination of multiple PCA components on the ADMCI Dataset

Table 1: Performance of the Classifiers on ADCN Training Dataset

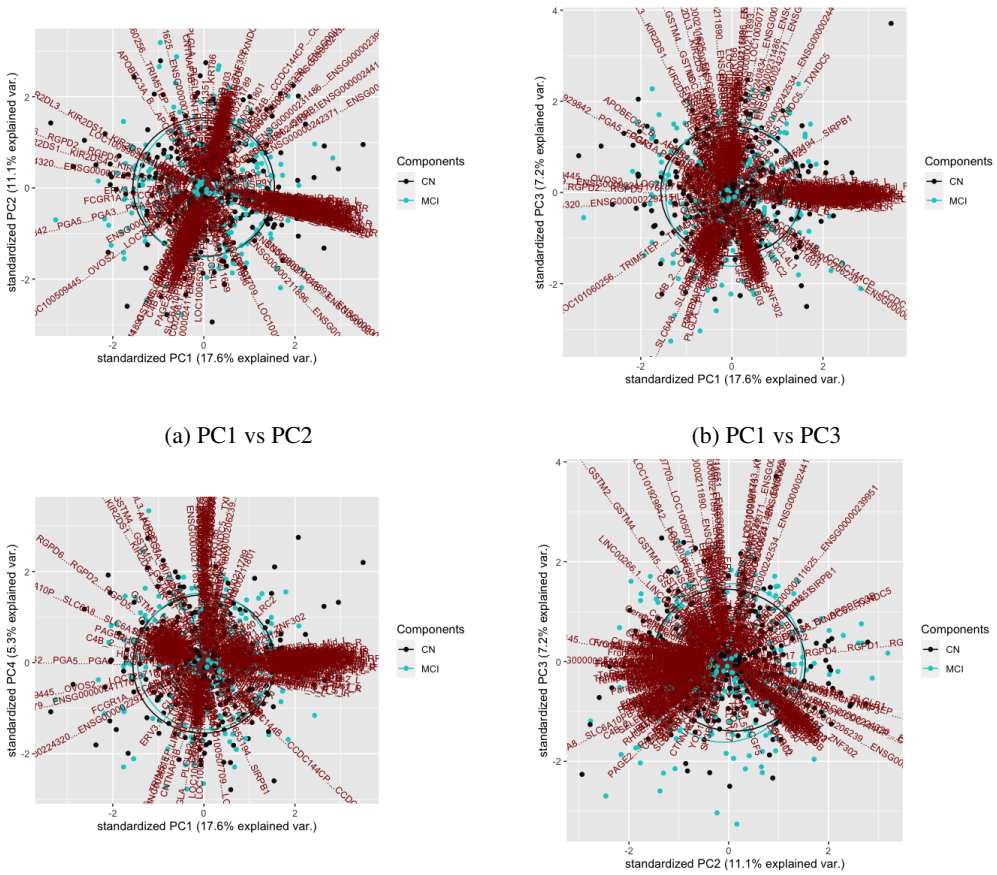
Method	**MCC Score**	**auPR Score**	**Accuracy**
K Nearest Neighbour	0.5606952	0.6886765	0.8932971
Linear Discriminant Analysis	0.8947383	0.9361765	0.9105072
Logistic Regression	1	1	0.8940217
Support Vector Machine	**0.9827306**	**0.9852941**	0.8929348

6.1 Pre-Processing and Feature Selection

1. The result of the VIF index showed that among 596 predictors, 40 of the input variables have co-linearity problem. I removed all the 40 predictors from the training set.
2. After deleting the correlated features (as mentioned in Section 3) I had 292 predictors left.
3. After carrying out lasso regression feature selection (as mentioned in Section 3) I had 60 predictors left. The convergence of the predictors is shown in Figure 6b

6.2 Results

The best performance was obtained by using Linear Discriminant Analysis. I would not choose Logistic Regression or Support Vector Machine (SVM) as the better classifiers because both of them have 100% performance, which means they are clearly overfitting. Table:2 shows the performance of all the classifiers on the training dataset.



(a) PC1 vs PC2

(b) PC1 vs PC3

(c) PC1 vs PC4

(d) PC2 vs PC3

Figure 5: Plots of combination of multiple PCA components on the MCICN Dataset

Table 2: Performance of the Classifiers on ADMCI Training Dataset

Method	**MCC Score**	**auPR Score**	**Accuracy**
K Nearest Neighbour	0.5242435	0.6470588	0.9199785
Linear Discriminant Analysis	**0.950973**	**0.9691262**	0.9601634
Logistic Regression	1	1	0.9576631
Support Vector Machine	1	1	0.9600175

7 MCICN Dataset

In this section, I will explain how I preprocessed, trained and evaluated the MCICN Dataset.

7.1 Pre-Processing and Feature Selection

1. The result of the VIF index showed that among 421 predictors, 58 of the input variables have co-linearity problem. I removed all the 40 predictors from the training set.

2. After deleting the correlated features (as mentioned in Section 3) I had 196 predictors left.

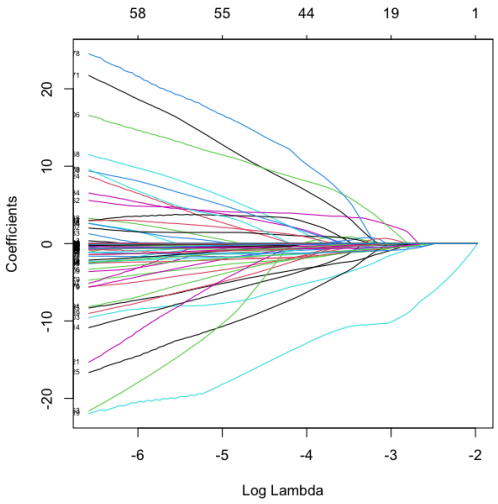
3. In case of MCICN Dataset, lasso regression feature selection deteriorated the performance of the classifier, so I skipped this feature selection.

7.2 Results

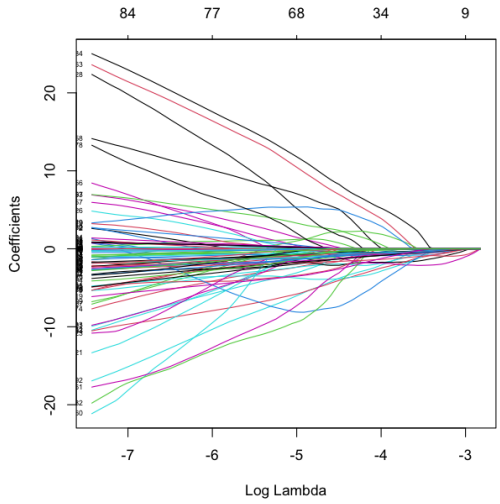
The best performance was clearly obtained by using Support Vector Machine. Table:3 shows the performance of all the classifiers on the training dataset.

Table 3: Performance of the Classifiers on MCICN Training Dataset

Method	**MCC Score**	**auPR Score**	**Accuracy**
K Nearest Neighbour	0.2810311	0.6107164	0.5884512
Linear Discriminant Analysis	0.5840996	0.7887865	0.5609428
Logistic Regression	0.6316319	0.8115205	0.5513805
Support Vector Machine	**0.7300473**	**0.8642544**	0.5517845



(a) Plot of Lasso Regression on ADCN Dataset



(b) Plot of Lasso Regression on ADMCI Dataset