**36662 Data Mining Final Report**

**Team #1, 2018 Spring**

**Han Bao, Tianyi Lan, Sijia Liu, Jiayu Zhu**

1. **Introduction**

This report documents the analysis done on the dataset containing 133 patients from two classes (a control class and a case class) with measurements on the same set of 248 genes which are selected from a much larger set of genes. The analysis of interest is to use both supervised and unsupervised learning methods to effectively assign class labels to each patient. The report is divided into mainly two sections: the unsupervised learning, which did not use true labels and after dimension reduction, found two best models for clustering patients into two classes, and the supervised learning, which used the true labels to find two best classification models that assign class labels to new patients.

1. **Unsupervised Learning**

Before selecting specific clustering methods, some exploratory data analysis would help yield some useful information about the dataset itself. For this section of the report, the data available consists of measurements of the same 248 genes for 133 patients from two classes (a control class and a case class). To unravel some potential interesting structure of this high-dimensional dataset, principle component analysis (PCA) was implemented to first reduce the dimensions and select certain features so that the structure of the data is easier to visualize and interpret.

* 1. **Principle Component Analysis**

**Figure 1** plots the variances of all variables for the training set (100 patients randomly selected from the 133 total patients), it’s worth noticing that there are some extremely large variances for several variables whereas most of the variables have variances below 1000. This means that the variables have substantially different numerical ranges. Those variables with much larger ranges would also have larger variances. Since it’s desirable to use PCA for feature selection and PCA is a method that maximizes variance projections, extremely large variances would dominate the associations between the variables in the sample. Therefore, having no interest in the different ranges of variables, scaling is desired to standardize the data so that all variables make the same contribution to the model by setting the length of each variable axis to unit variance.

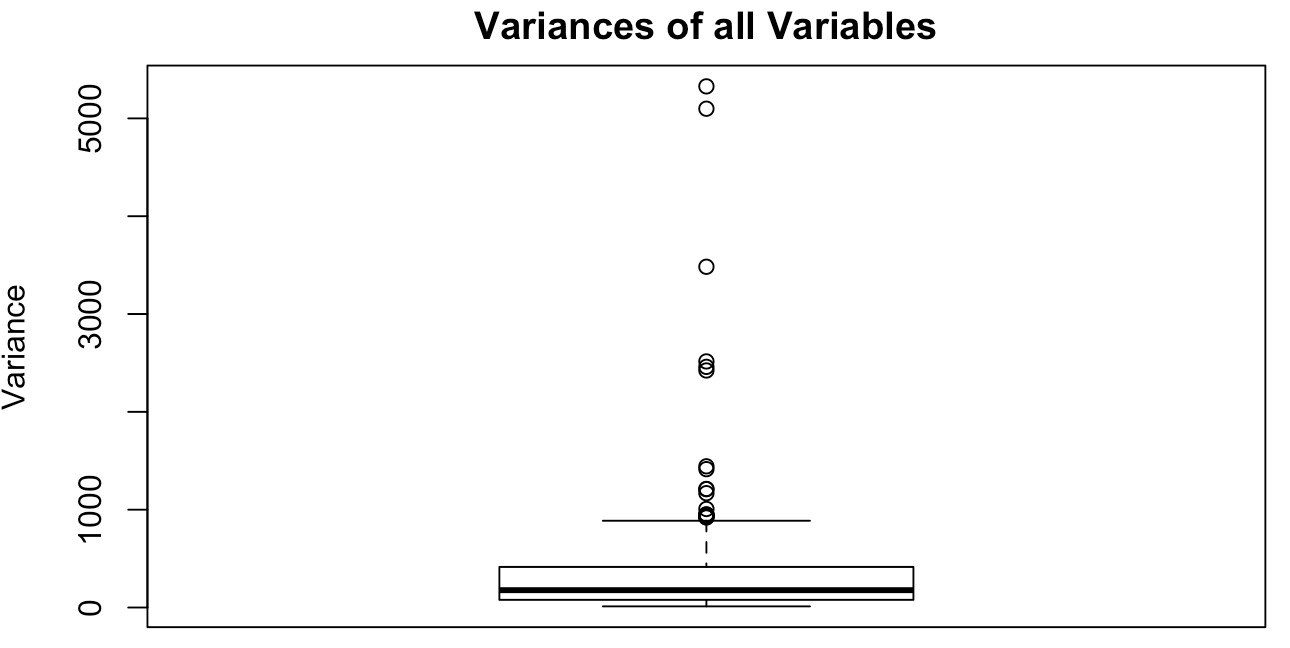


Figure 1. Boxplot of Variances for 248 Features in Training Data

On the other hand, the original variables didn’t have zero means, so to improve the interpretability and orthogonalize the covariance matrix, a mean-centering procedure was also applied. With both pre-treatments (scaling and centering) used, the data was transformed into a more suitable form for analysis. As shown in **Figure 2**, the left plot shows the first two scores of PCA without standardizing whereas the plot on the right displays the normalized PC scores with scaling and centering. It’s clear that without these two procedures, the two clusters were not separated vertically by 0 and the scores both had very wide ranges. Such problem was mediated by scaling and centering as the plot on the right showed fairly clear separation of the two classes.

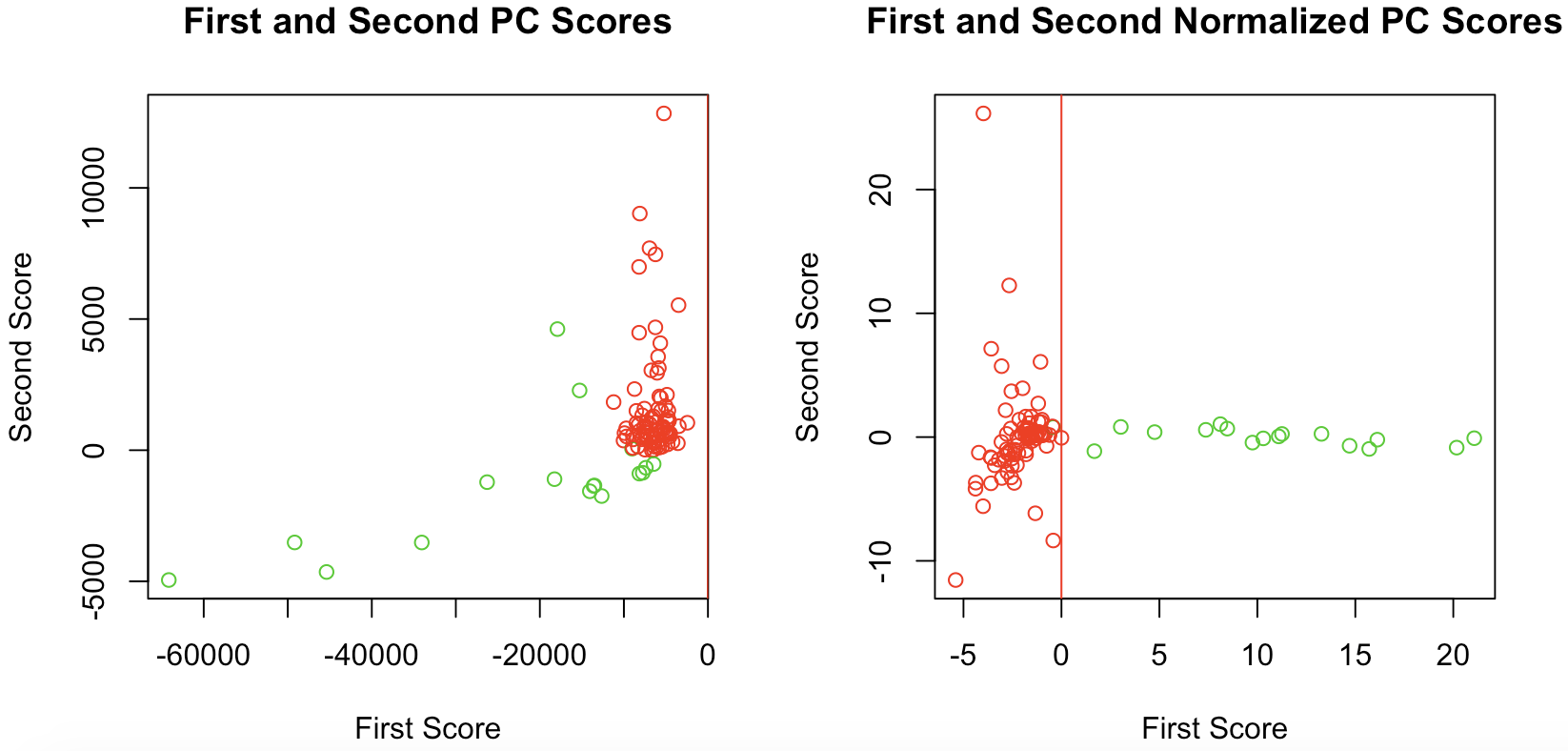


Figure 2. Comparison between Results of PCA with and without Standardization

A further attempt to improve the PCA result was made by trying the whitening operation. This operation takes the data in the eigenbasis and divides every dimension by the eigenvalue to normalize the scale. However, there was no apparent improvement in the score plot after whitening, therefore the whitening operation was excluded from the final analysis and the analysis was moved forward to clustering.

* 1. **K-means Clustering Methods**

With the two classes well separated after applying PCA and the known number of clustering centers of 2, a fast and robust clustering method would be K-means. As **Table 1** shows, three models were conducted using the k-means clustering with different input values. The Hamming error rate was calculated for both the training set, which contained a random sample of size 100 from the 133 patients, and the validation set, which consisted of the 33 patients left.

Table 1. Hamming Error Rates for K-Means Clustering with/without PCA

|  |  |  |  |
| --- | --- | --- | --- |
| Model | K-means with First Two PC Scores | K-means with the First PC Score | K-means without PCA |
| Training Hamming Error Rate (%) | 3~22 | 3 | 12 |
| Validation Hamming Error Rate (%) | 3.03~15.15 | 3.03 | 12.12 |

The best k-means model appeared to be the one using only the first PC score, with the lowest training and validation Hamming error rates, 3% and 3.03% respectively. The k-means model without PCA had highest Hamming error rates in both the training and testing set, which made it the least accurate model among the three. The k-means model with the first and second PC scores had a very unstable performance as the error rate ranged from 3% to 22% at different tries. Besides, the lowest it ever accomplished was about the same as the model with only the first PC score. According to Occam’s razor, the model with only first PC score was chosen to be one of the final two best models based on its simplicity, stability and satisfactory performance with consistently low Hamming error rates. Note that comparing the one using the first PC score and the one without PCA, the benefit of using PCA before conducting k-means clustering is evident as the Hamming error rates decrease notably.

* 1. **Alternative Approaches**

Several alternative approaches were also applied to improve the prediction performance. Another appropriate clustering method is hierarchical clustering, motivated by the fact that there existed several data points around the border of the two classes which made clear separation difficult. In the hope that a different bottom-up approach that constantly looks for the two most similar items might yield different results, a hierarchical clustering model using Ward’s minimum variance method was attempted to find compact and spherical clusters. Looking at the Hamming error rates for hierarchical clustering model, the training error rate was relatively low, only 4%, comparable to the model with the first PC score. However, the validation error rate was as high as 12.12%, indicating a poor prediction performance.

Instead of using PCA, another attempt involved the unsupervised learning with random forest, which was used to construct a joint distribution that roughly describes the complete data (133 patients) for the purpose of dimension reduction. For those 133 real observations labeled as 1, this algorithm constructed 133 simulated observations and labelled them as 0. The random forest classifiers then distinguished between these two labels, therefore the important features selected by this random forest process can described the distribution of the real observations and the real observations were converted into a feature map.

Two clustering methods were then applied with euclidean distance on this feature map. With K-means method, the misclassification rate ranged from 0.075%~20% for the whole 133 transformed data points and this model was not selected due to its instability. Alternatively, as suggested by Tao Shi (2005), the partition around metroids (PAM) clustering method was used to make stable clusters. Notice that here the random forest procedure cannot be re-run with the same dataset, thus no validation error rate was reported. This model yielded a Hamming error rate of 3% for our whole dataset and was therefore chosen to be the other one of the final two best models.

* 1. **Summary**

For the unsupervised learning section of this report, PCA was used as a first step for feature selection for easier interpretability and visualization. Standardization procedures, scaling and mean-centering were used to transform the data in order to avoid the domination of variables with higher variances and orthogonalize the covariance matrix. Afterwards, six clustering models were tested utilizing K-means with different input values, hierarchical clustering analysis and unsupervised random forest with PAM. The best two models were selected to be the K-means model with the first PC score and the PAM clustering model with random forest on account of their stability and lowest Hamming error rates.

1. **Supervised Learning**

In Supervised Learning, each data point is a pair consisting of a set of observed input features and a desired output. Due to high dimensionality in the data (248 features for 133 observations), a reasonable approach for this two-class classification problem is using Shrinkage methods which applies regularization to the regression model for the purpose of feature selection. Another applicable method is random forest.

* 1. **Shrinkage Methods**

As two popular shrinkage methods, both ridge regression and lasso regression shrink the regression coefficients by applying a penalty term. The ridge and lasso coefficients minimize the penalized residual sum of squares:

Ridge:

Lasso:

Here, is a tuning parameter that controls the strength of the penalty or regularization. The larger the value of λ, the greater the amount of shrinkage. One substantial difference between ridge and lasso regressions is that lasso coefficients can be shrunk to exactly zero due to the nature of penalty, while ridge coefficients can only be shrunk towards zero. By using lasso regressions, we assume that the true is sufficiently sparse, so that there is only one sufficiently sparse solution that can be found by -penalization method.

In order to choose the best tuning parameter for prediction, K-fold cross-validation was conducted to estimate the expected prediction error. Specifically, all 133 observations were randomly assigned to 10 folds with about equal sizes. Each time, the models were trained on all but the fold and then validated on the fold, iterating over Note that due to the aforementioned issue of unequal scales of 248 predictor variables, normalization before cross-validation was considered a necessity for ridge and lasso regressions, which could be achieved automatically with *R* *glmnet* package. Then for each , the average prediction error was computed, which yielded the cross-validation error curves like this:

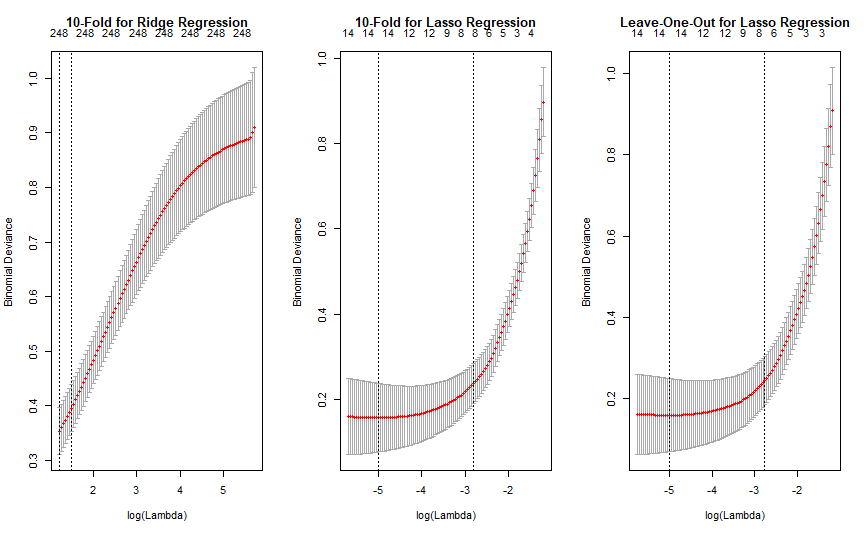


Figure 3. 10-Fold and Leave-One-Out Cross-Validation Error Curves for Ridge and Lasso

Generally, the best can be chosen based on the usual rule which minimizes the cross-validation errors, or the one standard error (SE) rule which tends to pick a more regularized model compared to the usual rule. In addition to 10-fold cross-validation, the leave-one-out cross-validation was also conducted for lasso regression, because it would be more robust and also computationally acceptable for a relatively small testing dataset. Here, misclassification errors were computed for for both ridge and lasso regressions on all 133 observations:

Table 2. Best Values of and Misclassification Errors for Ridge and Lasso Regressions

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Model | Ridge  (usual rule)  10-fold | Ridge  (one SE rule)  10-fold | Lasso  (usual rule)  10-fold | Lasso  (one SE rule)  10-fold | Lasso  (usual rule)  leave-one-out | Lasso  (one SE rule  leave-one-out |
| λ | 3.209 | 4.444 | 0.005 | 0.060 | 0.007 | 0.063 |
| Misclassification Errors (%) | 4.51 | 5.26 | 0 | 1.5 | 0 | 1.5 |

As **Table 2** shows, lasso regressions outperformed ridge regressions with lower misclassification errors. Specifically, both lasso with λ = 0.005 (the usual rule using 10-fold cross-validation) and lasso with λ = 0.007 (the usual rule using leave-one-out cross-validation) had the best performance with no misclassification errors. In addition, as discussed above, lasso regression with λ chosen by leave-one-out cross-validation was favored by the nature of small testing dataset. However, which rule should be applied to choose the best λ largely depends on the testing data. Hence, the best shrinkage model selected is lasso regression with the best λ chosen by leaving-one-out cross-validation, using whichever rule that minimizes misclassification errors.

* 1. **Model-Free Methods**

Generally, regression models require certain assumptions of the data distribution for good prediction performances. There are also several model-free methods that perform well in prediction without any assumptions. Specifically, three model-free methods include the k-nearest-neighbors, classification trees and random forests.

As one of the most simple and straightforward classification methods, the k-nearest-neighbors method tries to predict a new input according to its labeled k-nearest neighbors. Here, the number of nearest neighbors is a tuning parameter. Since the purpose is to minimize prediction errors, the best can be chosen by 10-fold cross-validation.

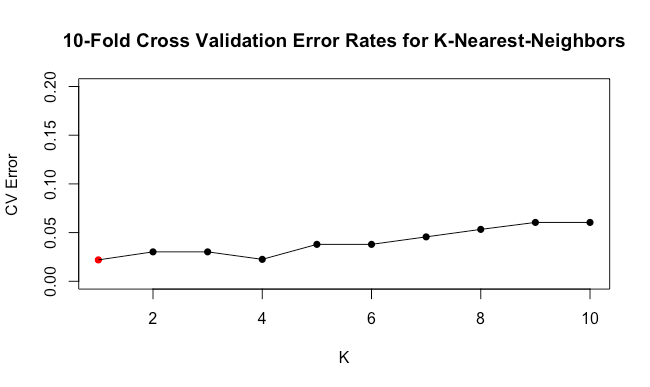


Figure 4. 10-Fold Cross-Validation Error Curve for K-Nearest-Neighbors

As **Figure 4** shows, when = 1, the cross-validation error rate reaches its minimum. Thus, a 1-nearest-neighbor method was applied to 100 observations that were randomly selected as training data and the remaining 33 observations as testing data. The misclassification error rate was 0%, indicating no errors in the classification. However, this method didn’t provide much insight into the relationship between predictors and classes, and hence could be computationally expensive.

The second model-free method is classification trees, which are commonly used in classification problems and interpretable. Classification trees first divide up the feature space into several rectangle regions and then classify a new point by placing it into one region. A tree with tree size being 3 was first grown using the CART algorithm. In order to find the best tree size, cross-validation on this pruned tree was conducted:

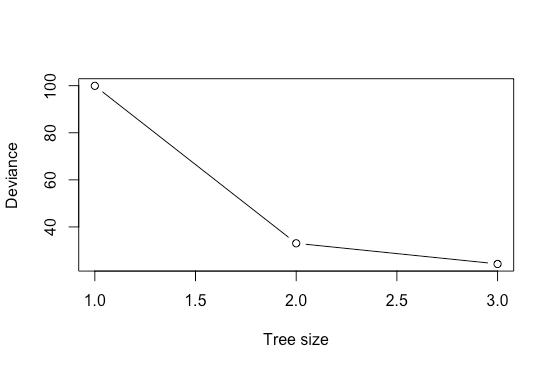


Figure 5. 10-Fold Cross-Validation Error Curve for Pruned Classification Trees

As **Figure 5** suggests, the best classification tree has a tree size of 3 where the deviance reaches its minimum. Same as the -nearest-neighbors, the misclassification error rate of this best tree was also 0%.

As another popular classification method, random forests builds a large collection of de-correlated classification trees and then averages them to reduce the variance. There are two tuning parameters for a random forests, i.e. the number of trees to grow and the number of variables that are randomly sampled as candidates at each split. Generally, the performance improves as increases and more trees are grown. Here, is a reasonable choice to assure the performance in a small testing dataset. By default, is the square root of all predictor variables, and hence is set at

Table 3. Tuning Parameters and Misclassification Error Rates for K-Nearest-Neighbors,

Classification Trees and Random Forests

|  |  |  |  |
| --- | --- | --- | --- |
| Model | *K*-Nearest  -Neighbors | Classification Trees | Random Forests |
| Tuning Parameter | # Neighbors (*k*) = 1 | Tree Size = 3 | # Trees (*n*) = 500  # Variables (*m*) = 16 |
| Misclassification Errors (%) | 0.0 | 0.0 | 0.0 |

* 1. **Summary**

For the supervised learning question, both shrinkage models and model-free methods were applied to this high-dimensional dataset. First, both ridge and lasso regressions were fitted and then tuned by 10-fold cross-validation. The leave-one-out cross-validation was also applied to the lasso regression. The best shrinkage models with no misclassification errors were the lasso regressions with the tuning parameter *λ* chosen by 10-fold and leave-one-out cross-validation respectively, using the usual rule. Then, three model-free methods, the *k*-nearest-neighbors, classification trees and random forests, were applied to the data. The 10-fold cross-validation was also conducted to choose the best values for tuning parameters of the *k*-nearest-neighbors and classification trees. Based on misclassification error comparison, model robustness and flexibility, the best two models for supervised learning were considered to be the lasso regression tuned by leave-one-out cross-validation and random forests, with both misclassification errors being 0%.

***Reference***

Tao Shi (2005), Unsupervised Learning With Random Forest Predictors, *Journal of Computational and Graphical Statistics, Volume 15, Number 1, Pages 118–138.*