Final Project (4 people)

Aishwarya Guda(aguda), Xinya Li(xinyal), Mrinalini Samanta(mrinalis), Richa Singh(richasin)

Aishwarya Guda, Xinya Li, Mrinalini Samanta, Richa Singh

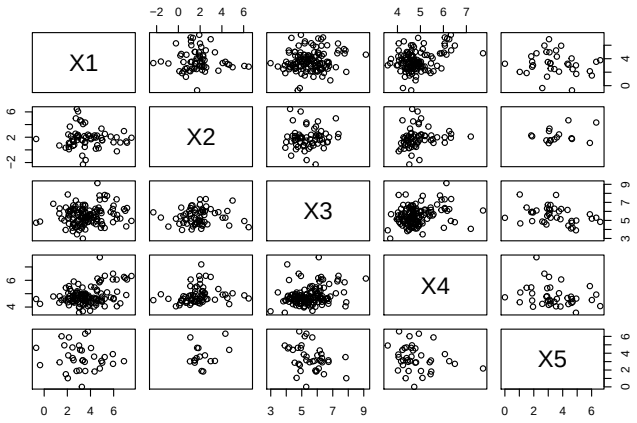
Introduction

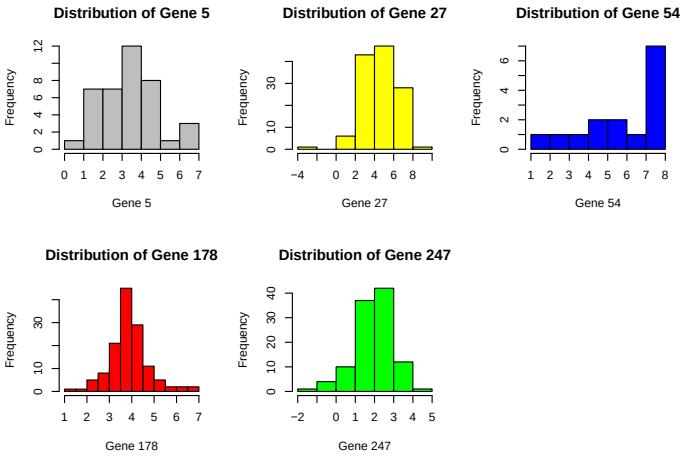
For our data set, we have a variety of 248 genes which respond to either a control or a response case. Of the 133 samples, 22 are response cases, while 111 are control cases (it is important to note that these groups are not equally represented). Our motivations for determining which genes cause a significant response led us to explore both unsupervised learning and supervised learning techniques. In order to choose which one of our models for each method were useful, we assessed our models on accuracy rates (using both cross validation and hamming errors) as well as model interpretability. We ran our models for both supervised learning and unsupervised learning on variety of high dimensional datasets and reduced dimensional datasets to compare our results in determining, which methods would be most accurate and useful.

Exploratory Data Analysis

To initially get a sense of the distributions of a few genes, we randomly sampled a few and plotted the histograms of their log distributions. We plotted on a log scale for improved visibility in determining their individual distributions. From our histograms we can see that genes 5, 178, and 247 are roughly normally distributed. While, the log distribution of gene 27 is slightly skewed left and gene 54 has more of a uniform distribution but the right tail is very heavily weighted.

To assess the correlation between our genes we used a pairs plot. From the pairs plot of the first 5 genes, we see very few are highly correlated. In testing for multicollinearity among all the 248 genes we ran a correlation matrix. Checking for multicollinearity helped in our feature selection process for dimension reduction and it helped in improving the validity of our models.





Unsupervised Learning

For unsupervised learning we considered both hierarchical clustering and k-means. For hierarchical clustering we considered complete linkage, average linkage, and single linkage. However, before performing these clustering methods we looked into reducing our dimensionality to improve our clustering results.

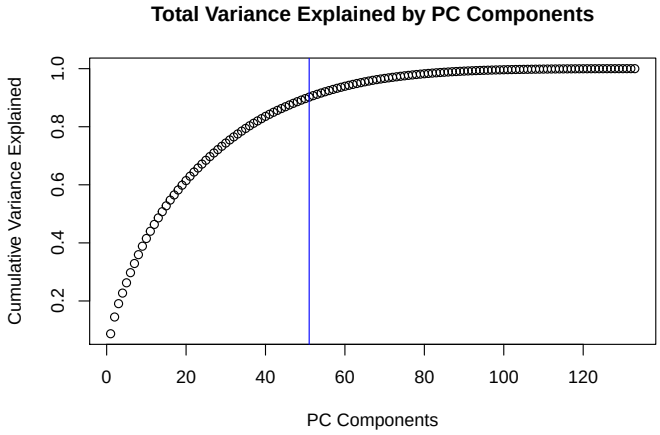
Dimensionality Reduction

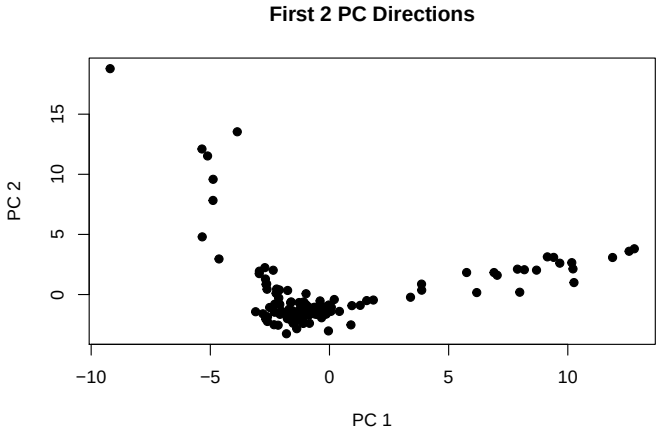
We explored multiple methods for reducing the dimensionality, one of which included feature selection. We initially tried to remove highly correlated variables. After observing pearson correlations of the initial data we found relatively high correlations (greater than 0.6). We tried different threshold values to choose to remove highly correlated variables and after trying different threshold values, we ended up using a threshold of 0.7. When removing variables that had an absolute pearson correlation of greater than 0.7, we were able to significantly reduce the dimensionality to 133 rows by 176 columns, compared to 133 rows by 248 columns, initially.

Once we initially removed the highly correlated variables, we also performed PCA on the reduced matrix, to determine which principle components explained the majority of the variance. To the get variance explained by each PC direction we squared the eigenvalues.

From our plot of variance explained by principal components we see that the top 51 principle components explains 90% of the variance. Since 90% is a large majority of the variance explained we decided to use the first 51 principle components. Thus, using PCA after removing the highly correlated variables helped to reduce dimensionality to 133 rows x 51 columns.

To get a sense of our initial data structure, we plotted the first 2 principle components of the reduced data. From the plot of our initial data we can see that a binary clustering is not very obvious from our data structure.



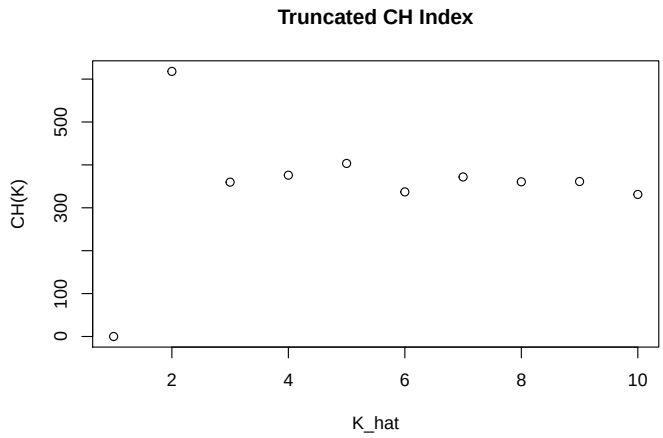


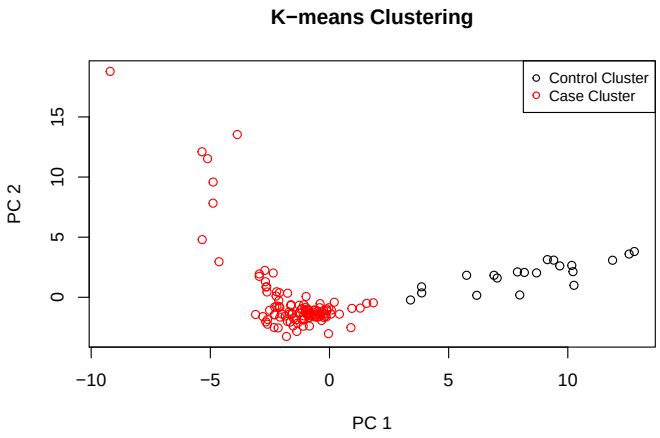
K-means

We considered k-means clustering because it does not involve assumptions about normality. From our exploratory data analysis all variables were not normally distributed. But k-means clustering does assume that the variance of the distribution of each attribute (variable) is spherical, all variables have the same variance, and each cluster has roughly an equal number of observations. From our EDA we

found that the majority of the observations were for the control case. Since this assumption is violated, we may not get the correct clustering for k-means.

To determine the optimal number of clusters we should use in our k-means algorithm we used the maximum ch.index (2) on the reduced data. After choosing a cluster size of 2 from running the ch.index, we ran k-means on the reduced matrix:



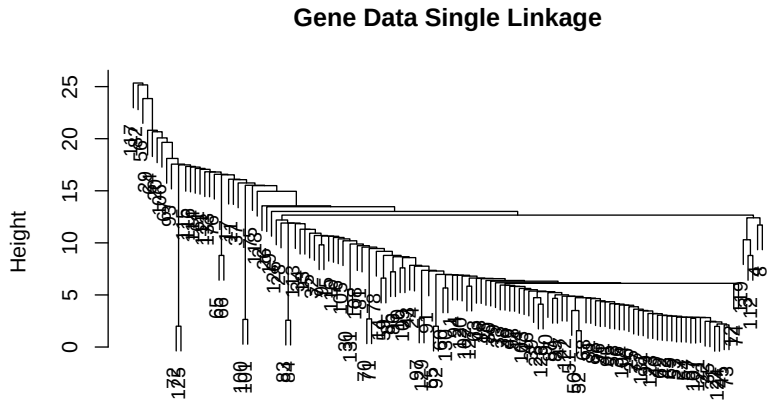


We compared our results on the reduced matrix to the original matrix. For the reduced matrix we had a within cluster variation of 8.5%, while for the original matrix it was 43.1%. And the hamming clustering error for the reduced matrix was 1.5% whereas for the original matrix it was 12%. Thus from these misclassification rates we can tell that the dimension reduction improved accuracy rates.

Hierarchical Clustering

In considering hierarchical clustering one assumption is that the data is independent (which we addressed by removing highly correlated variables in our dimension reduction). Some advantages to hierarchical clustering is that it doesn??t require the number of clusters and it doesn??t require the initial guess of cluster centers (whereas k-means does). We ran single, complete, and average linkage for the hierarchical clustering on both the reduced data and the original data. Our most accurate model hierarchical model on the reduced data was the single linkage model (with a hamming error of 15%). Our most accurate hierarchical model on the original data was our average linkage (with a hamming error of 12.7%).

Of our two most accurate hierarchical models, we chose single linkage on the reduced data because it has no inversions and a good cut interpretation. Whereas average linkage is not interpretable. One drawback of single linkage is there is a potential for chaining (clusters can be too spread out and not compact enough), whereas average linkage is balanced. Since we had a max ch index of 2, we chose to cut the tree with 2 being the number of chosen clusters. This makes sense as our original data has only two groups for classification. Our hierarchical single linkage plot can be seen below.



Comparison of the Two Methods

From both the k-means plot and the hierarchical plot, it is easier to interpret the k-means clustering output. However, the k-means algorithm had more assumptions, some of which we may have violated (equal cluster size). Whereas for hierarchical clustering we did not violate any assumptions. The hamming error for k-means on the reduced data was significantly lower (1.5%) than the hamming error for single linkage on the reduced data (12.7%).

We can see that the hamming errors of hierarchical clustering methods on the data with removed correlated variables are much lower.

We can see that according to error rates, the best two unsupervised prediction methods are k-means clustering with reduced data (with a hamming error rate of 0.015) and hierarchical clustering with reduced data (with a hamming error rate of 0.128). In general, a matrix reduced to lower dimensions significantly helps prediction.

Supervised Learning

With supervised learning, we tested a variety of classification models. To decide on the models that we wanted to test, we first listed out all supervised models that we learned in this class. Then, we narrowed it down to the classification models only. From there, we decided on a set of a few models that we thought would work well based on the characteristics of this data that we observed from our EDA.

The models that we tested included GLM, LDA, the Tree Model, and the Bagging Model. For each of these, we used cross validation to split our data into training and testing data on multiple folds. We tested each model on different data sets which we obtained from the different methods of dimension reduction that we performed.

Error Calculations

As mentioned above, we noticed immediately that our given classification labels were biased heavily towards 0 (the control case). Approximately $\frac{4}{5}$ of the data belonged to the 0 (control group), while $\frac{1}{5}$ belonged to the 1 (response group). We wanted to make sure the classifiers we built would not act ??lazy?? and classify most of the predictions as the control case just because that was the majority classification label in the training set. To account for this, we decided to oversample while building each of our classifiers. This way, we forced our model to pay more attention to the response class (the 1s).

Since we used cross validation to calculate our model error rates, we decided to oversample in each step of the cross validation. First, we split the data into a training and test set (depending on the current fold of the k-fold cross validation). From said training set, we oversampled from the underrepresented class (the 1s). To do this, we sampled with replacement from the data points belonging to the underrepresented class until we had about an equal number of data points for both the 1s and 0s. This way, our model was built on a more well balanced dataset. This greatly improved our error rates.

Dimension Reduction

With the availability of the y-labels for supervised learning, we could test LASSO feature selection for the data. Another method that we tested was removing highly correlated features. For each of these methods we applied PCA and tested our models on all these different subsets of the data. We also compared everything to the results on the original data itself as well.

For LASSO feature selection, we used glmnet to create a sparse matrix, where the entries of the sparse matrix represent the coefficients of the features that are significant for the data. To remove the features that were highly correlated, we removed variables that had an absolute pearson correlation of greater than 0.7 as for the unsupervised models.

After PCA on both of these methods, we decided to keep the features that explained about 90% of the variance. For the LASSO feature selection method after PCA, this was approximately 18 features. For the correlation-reduced data, after PCA we kept 51 features. We also tested all our models on data obtained from PCA alone, resulting in 133 features, and the full original data set with 248 features.

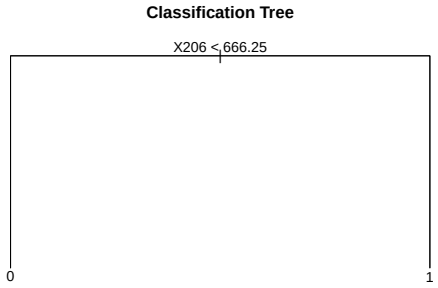
Clustering Methods

From testing our GLM, LDA, Tree, and Bagging models on each of these 5 data sets, we were able to compare the model performances and rule out the feature selection methods that removed too many important features from the data such as with LASSO, and Correlation-Reduced with PCA, or kept too many unnecessary features such as with the original data. In the end, we found that the top two methods for feature selection that we wanted to test further were Correlation-Reduced feature selection and PCA on the original data.

Classification Tree

The tree that best classified the data was the pruned tree from the correlation-reduced data. We pruned the tree by conducting cross-validation on the size of the tree to come up with the best size for the final

tree that we used for our predictions. The ??best-size?? parameter here is a tuning parameter for the trees.



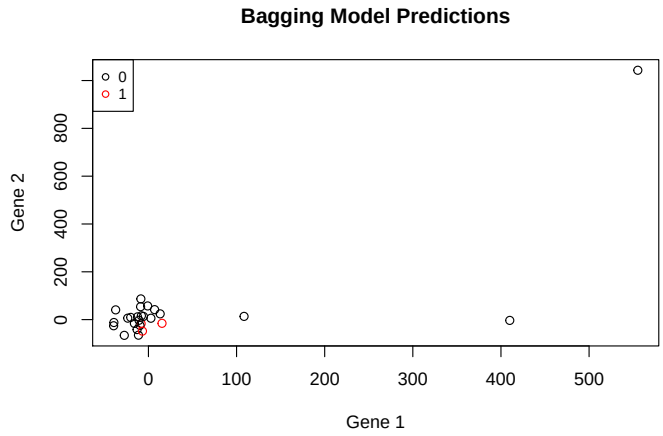
Model Accuracy

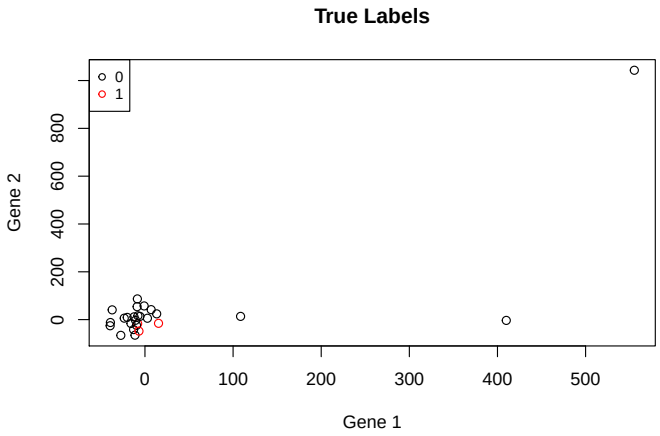
The summary of the tree model tells us that there are two terminal nodes, or leaves for this data. We are also getting a misclassification rate of 0. When we look at the tree itself, the node that it is splitting on is X206. The split criterion is X206 < 666.25, and the number of observations in that branch is 140 with a deviance of 0 . The prediction for this branch is 0, and the prediction for the rest of the data points is 1. The CV Error for the tree on the Correlation-reduced data was 0.74% and the hamming error was approximately 0%. For comparison, the tree model on the just PCA-reduced data gave a CV error of 4.5% and a hamming error of 11.5%.

For the classification problem at hand, this means that about 0.74% of the patients selected were misclassified between the case and control groups. With a hamming error of 0%, we can see that both errors indicate that close to 0 patients were misclassified. For these reasons, our final tree model, which was also our best model overall, was the Pruned Tree model on the Correlation-Reduced data.

Bagging - Random Forest

The bagging model we created uses random forests to bootstrap samples and aggregate the results of classification trees. Some of the advantages of this are similar to that of the regular tree model in that it does not have any model assumptions, however, this model has the disadvantage that it does not actually output any trees. Thus, the model is not interpretable, which takes away from the effectiveness of the model.





From these plots of the first two genes, we can see an example of the tree model classification that most of the predictions are correct when compared to the true labels. This makes sense as the average cross validation error that we got from our tree model was approximately 1.5%, which fluctuates slightly due to the stochastic nature of Random Forest models. We also got a hamming error of 3.8% for this model. For comparison, our bagging model on the PCA-reduced data was 9.7%and the hamming error was 7.6%.

In the context of this problem, the random forest model tells us that there was approximately a 1.5% misclassification error of patients into the control and case groups as indicated by the CV Errors and approximately 3.8% hamming error. For these reasons, our final bagging model was the Random Forest model on the Correlation-Reduced data.

Conclusion

In comparing both of our unsupervised models to our supervised models we believe that the classification tree is our overall best model. This is because it is useful for interpretability, does not involve any model assumptions, and has the lowest misclassification error rate. In instances where we need low misclassification rates and good interpretability our classification tree performs best. We were also able to adjust for the different weighting of the classes present in the data during our cross-validation error calculations, which helped in adjusting our accuracy rates. We are able to use these reweighting methods during cross-validation in our supervised models, when we were unable to do so in our unsupervised models. Thus, our supervised models are more flexible for data structures like the one we assessed.

Note: Error rates for all reported models may change based on different runnings of the code.