Option 1: Analyzing `Golub` Dataset

Peter Wu (peterwu)

Introduction

The aim of this project is to explore the `Golub` dataset, applying dimensionality reduction techniques like PCA and sparse PCA and then applying predictive models like penalized logistic regression and XGBoost. We incorporate several variations including using sparse PCA to select the top 25 genes and then run our predictive models and try different hyper-parameters when running our XGBoost algorithm. The dataset we are analyzing, `Golub`, is gene expression data (3051 genes and 72 tumor mRNA samples) from the leukemia microarray study of Golub et al. Of the 72 leukemia patients, 47 were diagnosed with acute lymphoblastic leukemia (ALL) and 25 as acute myeloid leukemia (AML).

For the purposes of this project, we will treat our task as a binary classification task where the input data are the 3051 genes from each of the 72 patients while the response variable is a binary variable that indicates whether a patient has `ALL` (coded 0 in the data), or `AML` (coded 1 in the data). Our goal is to build robust classification models that can perform this classification task and to also have some interpretability when it comes to dataset of which genes are the most important and which are not that important when it comes to classifying patients. The four main methods we employ in this paper are PCA, sparse PCA, penalized logistic regression, and XGBoost. The variations we employ in this paper are that we select the top 100 genes from sparse PCA to apply in our predictive models and the other variation is that when running XGBoost, we try a variety of different parameters for the number of rounds and the maximum depth of each tree.

Methods

The first method of our analysis is to apply dimensionality reduction techniques like PCA and sparse PCA to understand which genes may have a larger weight in predicting `ALL` or `AML`. We want to perform PCA on the dataset and then visualize it in 2D to see if there may be a visual pattern that we can observe to separate the data. Next, by applying sparse PCA, we want to see if this can be an improvement upon our analysis solely using PCA. I expect sparse PCA to work better than PCA because in PCA, the principal components are usually linear combinations of *all* the input variables. This makes is particularly a weak technique when there are a lot of features in consideration when in actuality only a few of them are needed to explain most of the data. In contrast, sparse PCA is able to overcome

this by making the principal components linear combinations that contain *only several* of the most important input variables. Because there are over 3000 input variables, I believe sparse PCA can do feature selection because it will remove the least important features and only use the features that are important in explaining most of the data.

To select the top genes, we summed up the numbers for each gene within the top 4 principal components we got from the sparse PCA analysis, and then used those as the top genes. We could have taken the top genes from only the first principal component for example, but using all 4 principal components gives us a better overall sense of the data. It will help us identify the genes that are showing up overall affecting the response variable the most.

The second method revolves around predictive modeling and it will be applying penalized logistic regression and XGBoost to the dataset. The way we will apply these methods is that we will split the data into 30% training and 70% testing and then train the model on the training and then see what percent accuracy we can get on the testing set. We will apply these methods to both the entire dataset and also the top 100 variables or genes we found from sparse PCA. Furthermore, we will try a variety of different tuning parameters for our XGBoost model to optimize as much as we can from this dataset that has only 72 observations.

Results

First, I performed EDA or exploratory data analysis to get a better grasp of the data. From the given dataset, we can see that we can either treat the problem where we are given the class labels (as either `ALL` or `AML`) and thus the problem would be a binary classification task, or we can treat it as if we do not know these labels and have to find structure within the data, or a clustering task. In this report, we will mainly treat this problem as a classification task.

We analyze more closely the class labels of the problem, whether a patient is diagnosed with acute lymphoblastic leukemia (ALL) or acute myeloid leukemia (AML). The `Golub` dataset given codes `ALL` as a 0 and `AML` as a 1. The table below displays the counts and proportions of the `tumor_type` variable from the `Golub` dataset.

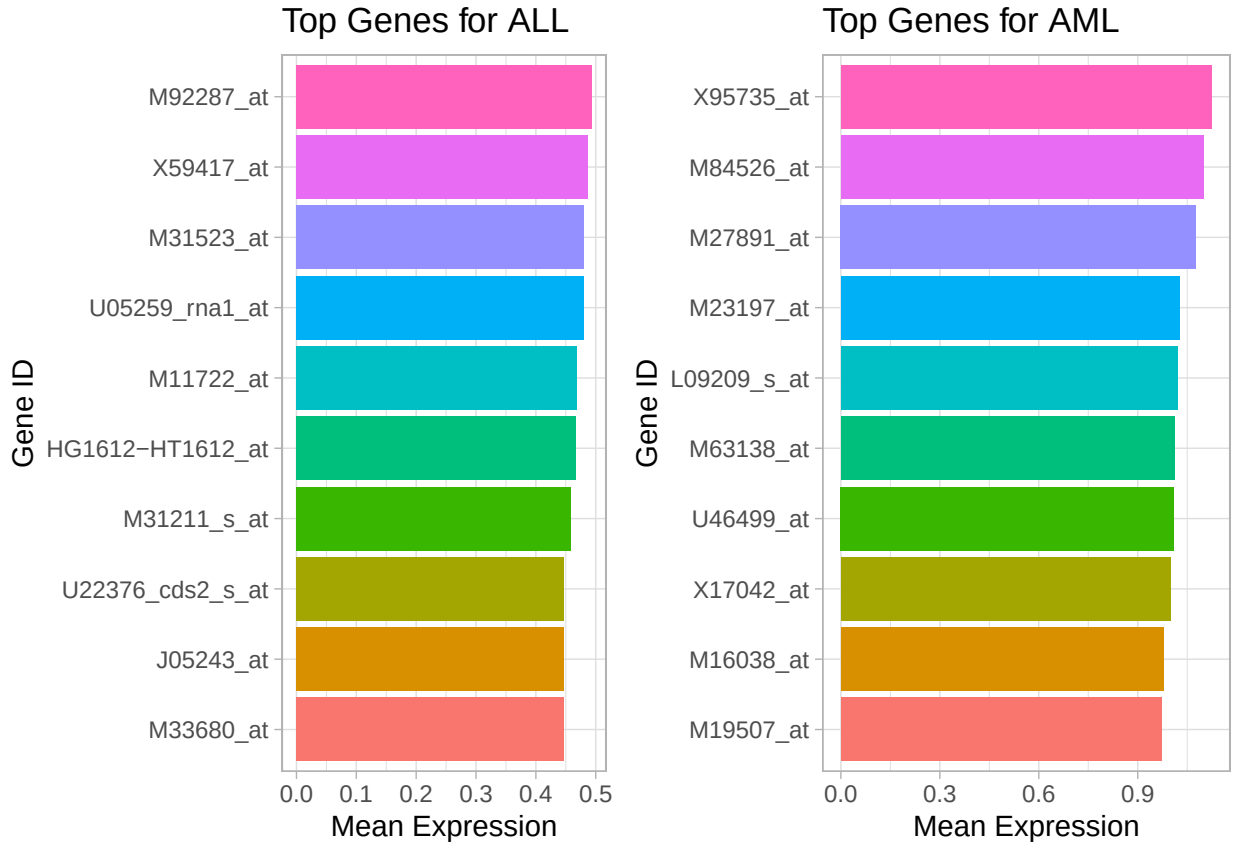
Table 1: Univariate Analysis of Tumor Types in `Golub` Dataset

Tumor Type	Count	Proportion
ALL	47	0.65
AML	25	0.35

From the results in the table, we see that the proportion of `ALL` in the dataset is 65% while the proportion of `AML` is 35%. This tells us that this dataset is not imbalanced where one class label happens to make up something like 90% of the dataset where a model that just

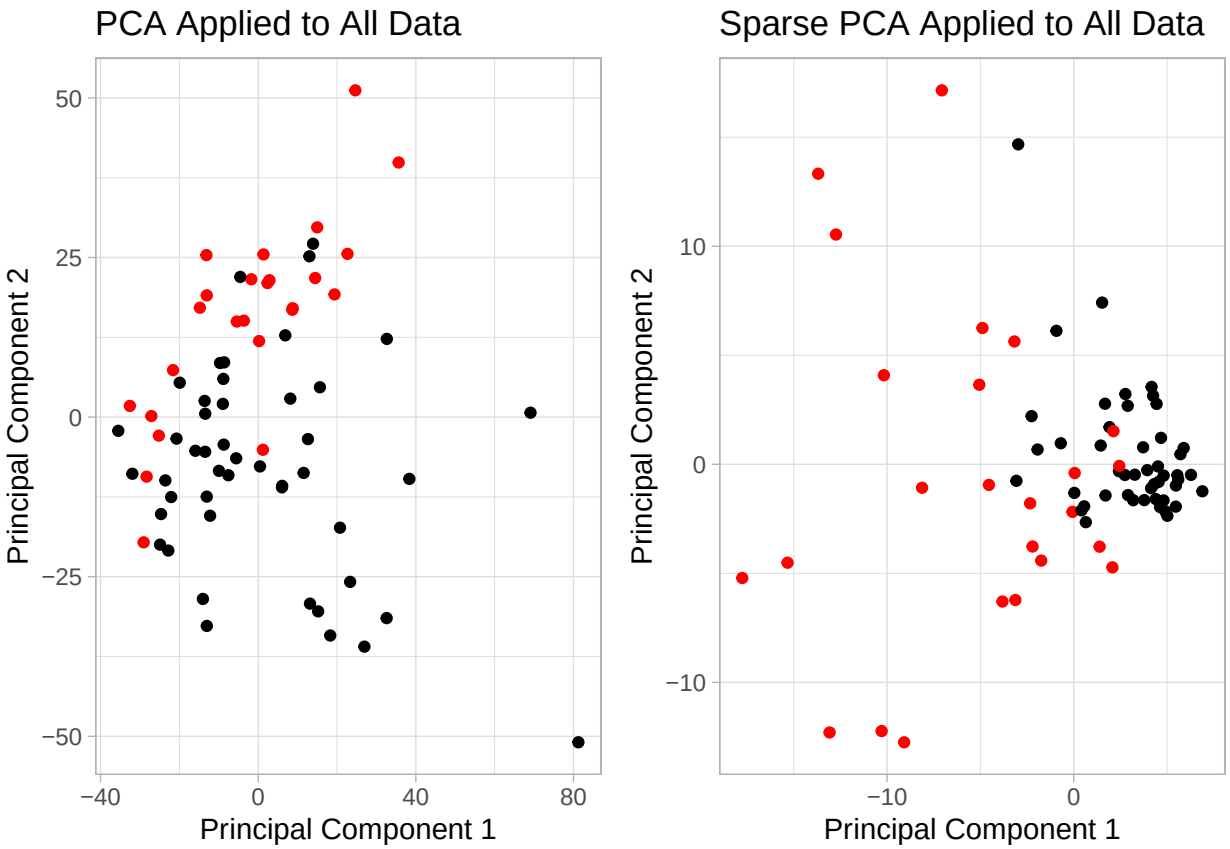
always predicts that class label will hit up to 90% accuracy. After taking a look at the response variable here, `tumor_type`, we shift our attention to the 3051 genes in the dataset.

Because there are so many genes in this dataset, over 3000, we will like to shift our attention to only several, and try to gauge a sense of which genes are the most important in our main task: trying to classify a leukemia patient between `ALL` and `AML`. To do this, we subset our whole dataset into two sets (one with patients that have `ALL` and the others that have `AML`) to find the top average mean gene expressions between the two groups. This will give us the genes that had the highest "weight" between the two class labels. The two plots below display our findings.



These plots give us a rough sense of which genes may interest us as we move to on to more concrete methods in this section.

Our first method we use is PCA and sparse PCA to find structure within the dataset visually first by analyzing it using only the first two principal components. The plot below applies PCA and sparse PCA to the entire dataset, coloring `ALL` and `AML` accordingly. The sparse PCA is done using the top 100 genes. These genes will add variation in the predictive modeling section of the report where we will use this sparse PCA as feature selection and take these top genes as the top variables.

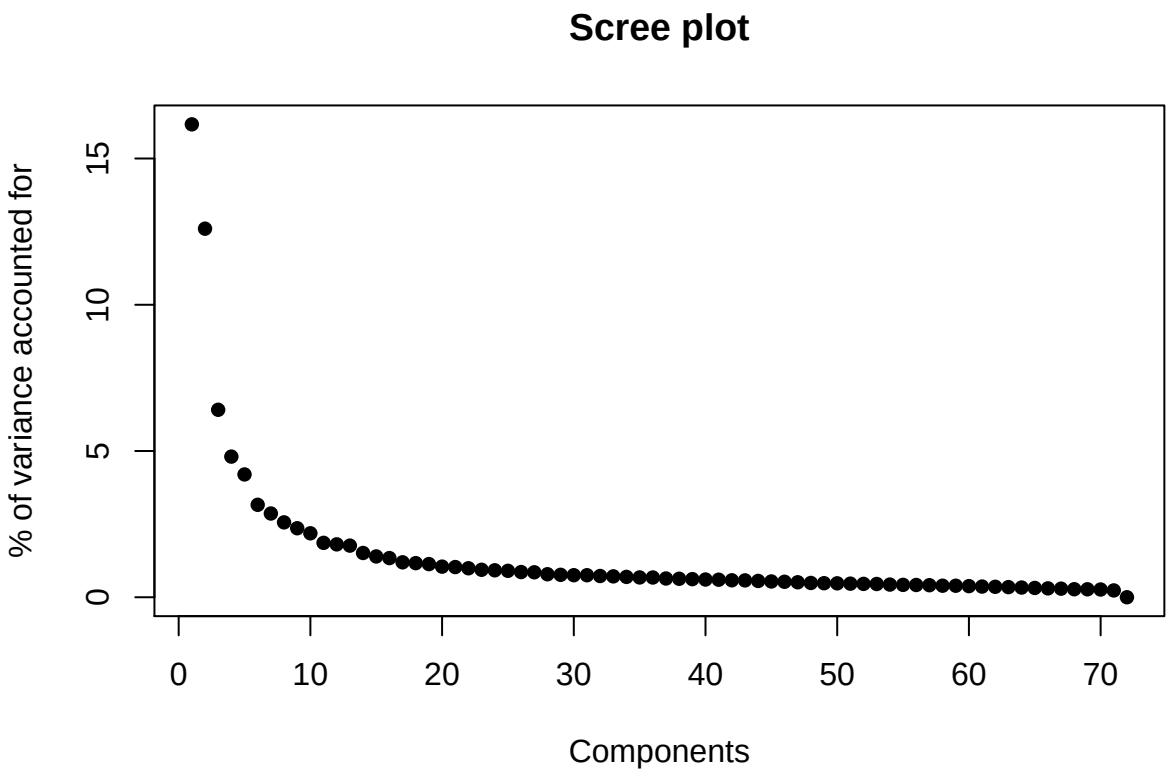


The table below shows the top 10 genes we got from our sparse PCA analysis.

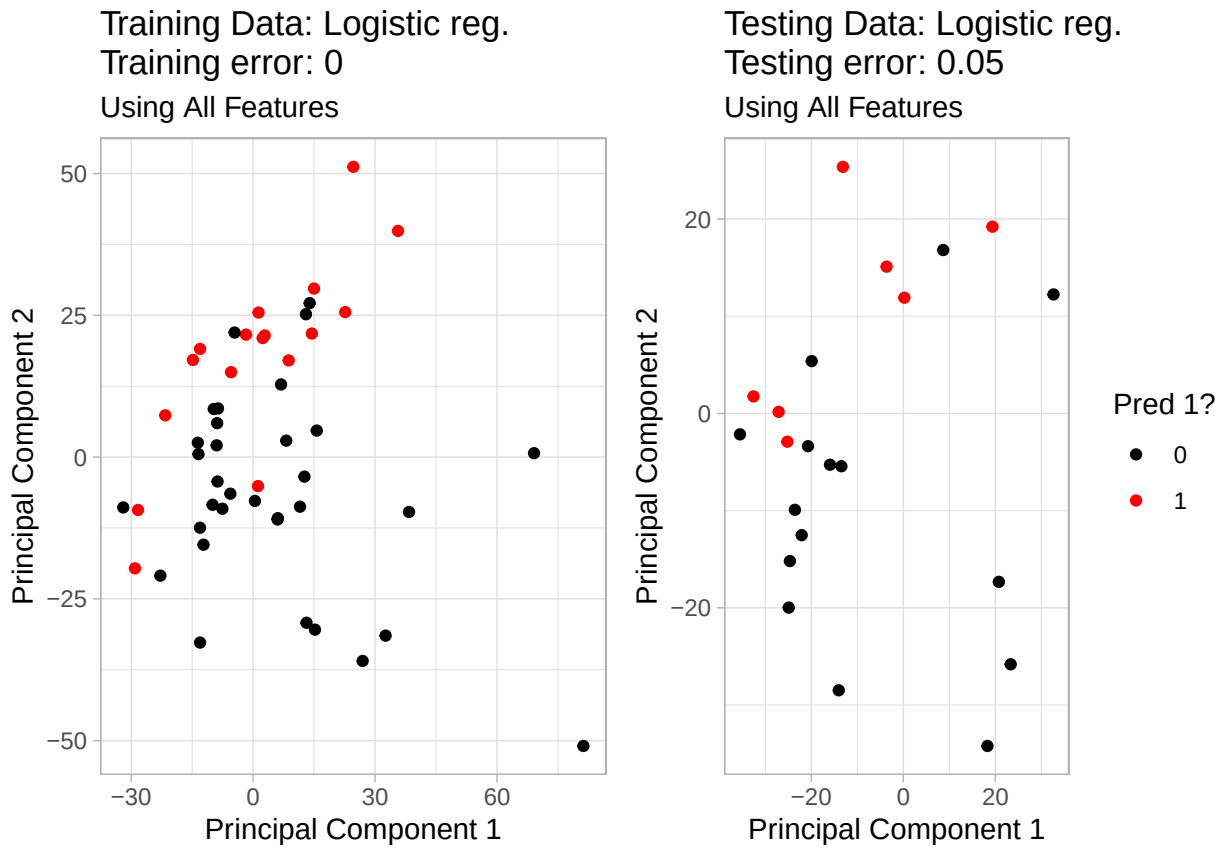
Table 2: Top Genes from Sparse PCA

Gene ID	Sum	Index
M91432_at	0.0648266	1037
U65928_at	0.0642232	1542
L40410_at	0.0634741	698
Y08614_at	0.0617678	2180
U62136_at	0.0616099	1524
D28473_s_at	0.0614748	2422
X78627_at	0.0614746	2032
D87078_at	0.0613621	345
D80006_at	0.0613585	286
L42572_at	0.0612718	713

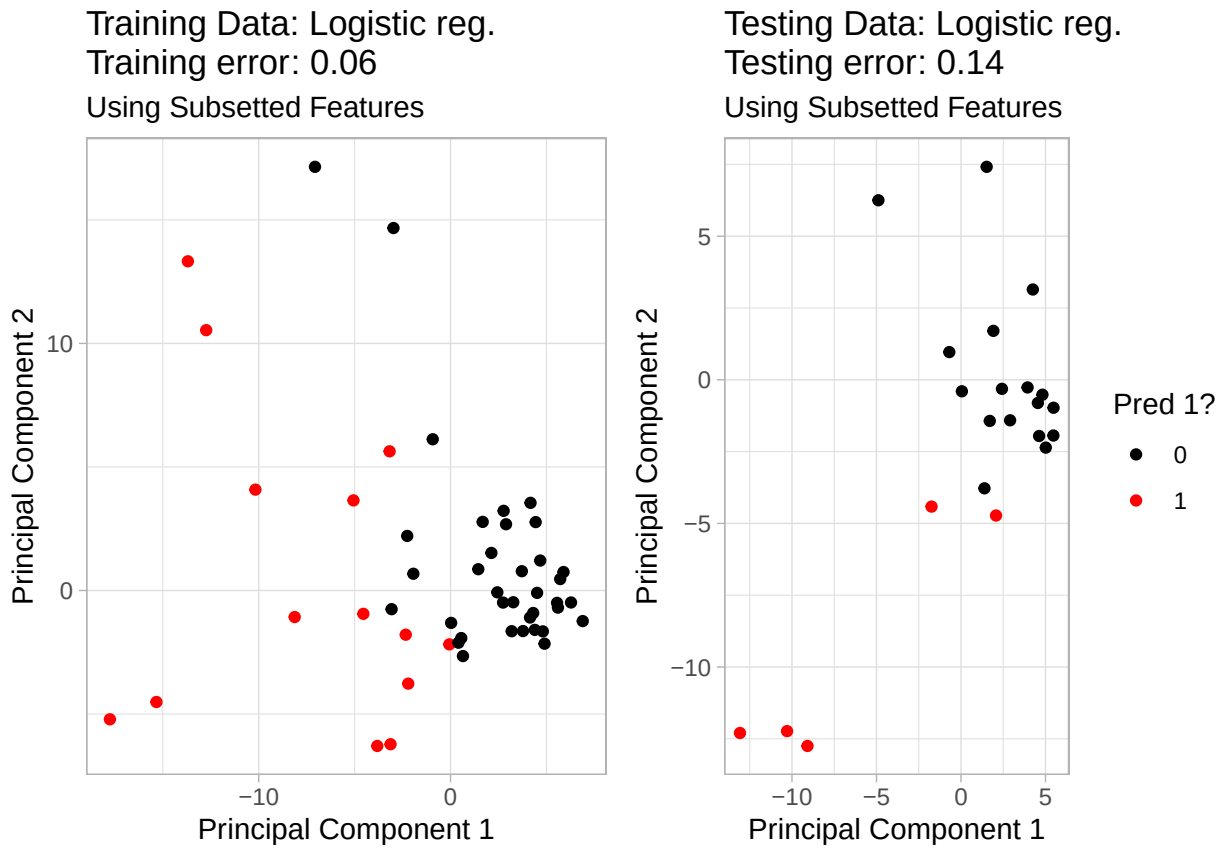
The below left is a scree plot which will display the cumulative proportion of variance accounted for in our data, which gives a sense of the number of principal components needed to describe the data.



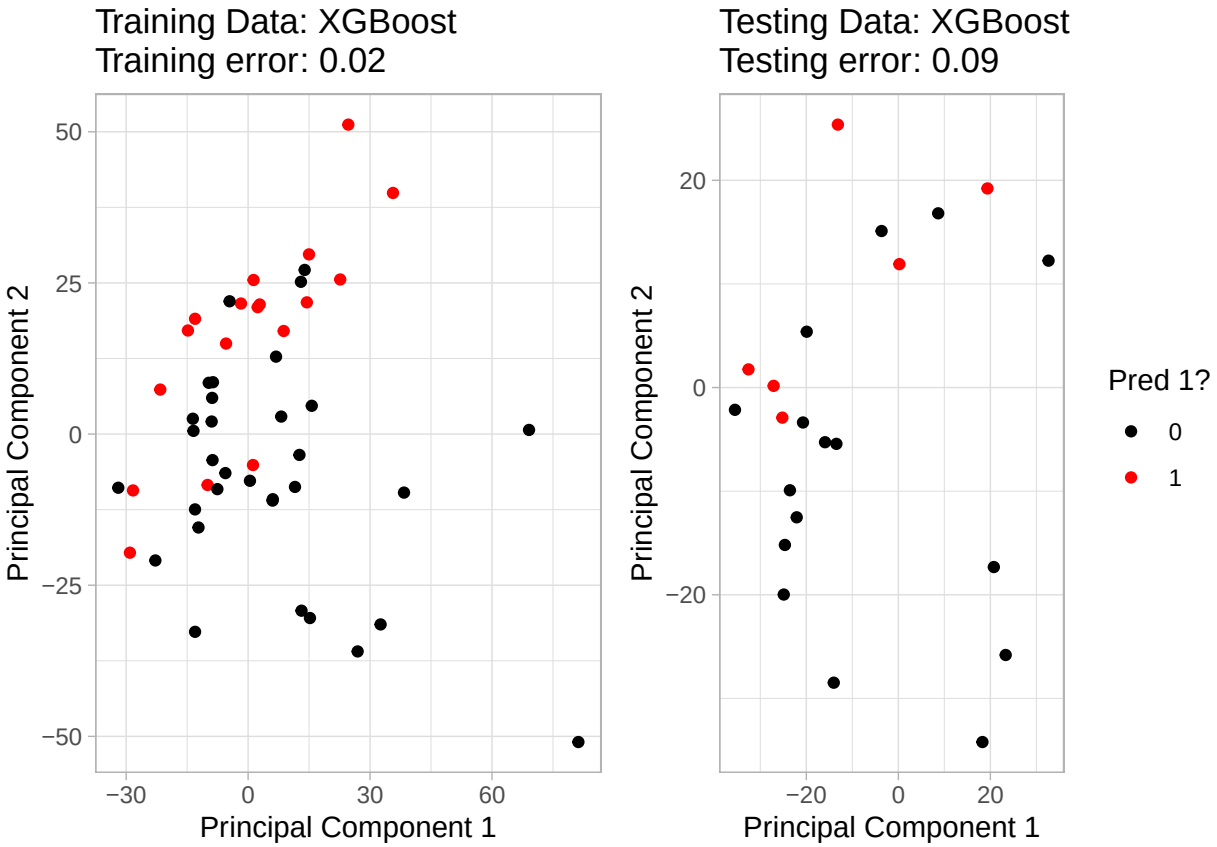
We now continue our Results Section by moving towards predictive models. The below plot is penalized logistic regression (L_1 penalty) applied using the entire dataset.



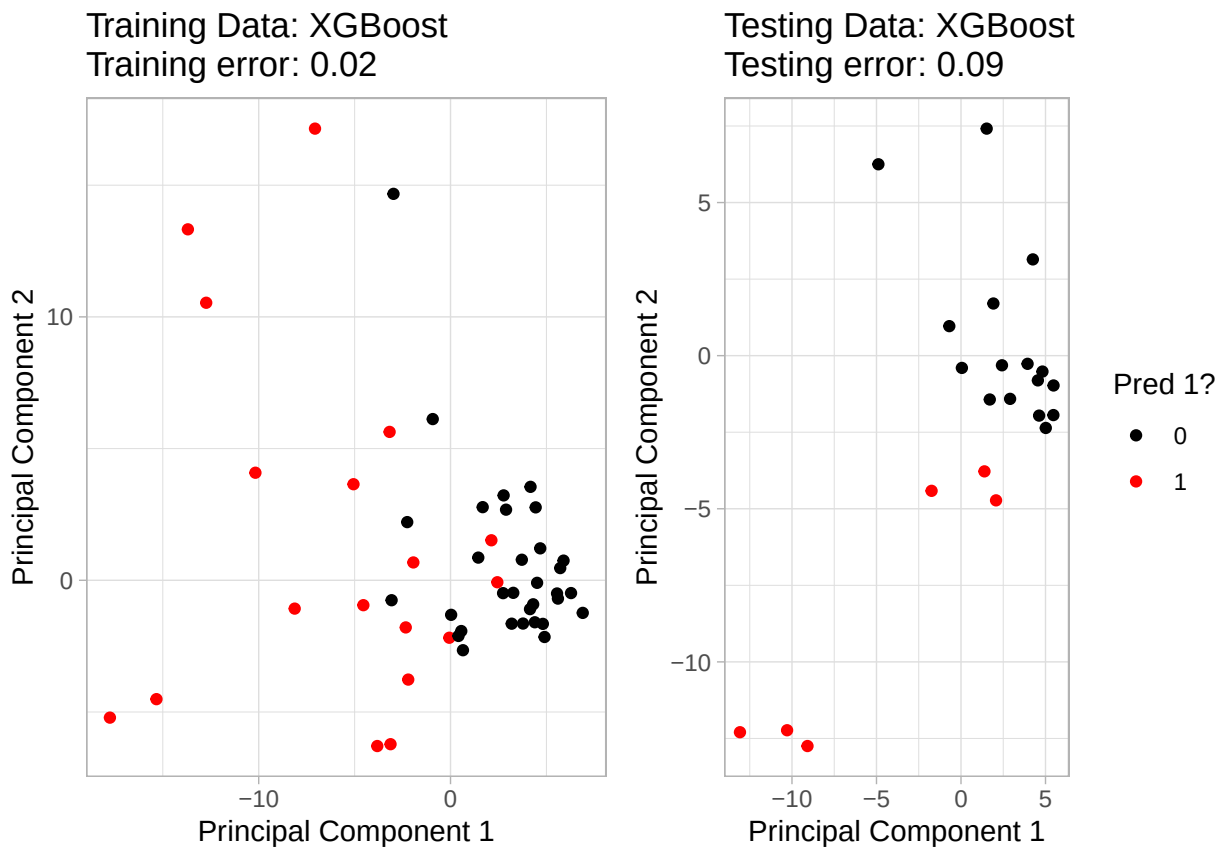
The below plot is penalized logistic regression (L_1 penalty) applied using the subsetted dataset using sparse PCA.



The below plot is XGBoost applied using the entire dataset.



The below plot is XGBoost applied using the subsetted dataset using sparse PCA.



The following below is a table that tries various different parameters on the XGBoost using the subsetted features we got from sparse PCA.

Table 3: Various Parameters for XGBoost

# Rounds	Max Depth	Training Error	Testing Error
1	3	0.02	0.0909091
1	5	0.02	0.0909091
1	7	0.02	0.0909091
5	3	0.02	0.0909091
5	5	0.02	0.0909091
5	7	0.02	0.0909091
9	3	0.00	0.0909091
9	5	0.00	0.0909091
9	7	0.00	0.0909091
13	3	0.00	0.0909091
13	5	0.00	0.0909091
13	7	0.00	0.0909091
17	3	0.00	0.0909091
17	5	0.00	0.0909091
17	7	0.00	0.0909091

# Rounds	Max Depth	Training Error	Testing Error
21	3	0.00	0.0909091
21	5	0.00	0.0909091
21	7	0.00	0.0909091

Discussion

There are plenty of main points we can derive from our analysis. Our work was a binary classification task to label `ALL` and `AML` using 3051 genes that were given for each patient. Our first method was to use PCA to try and find structure in the data. From our ordinary PCA plot, we can see that this method was decent but there was still some considerable overlap between the red and black points (red = 0 or `ALL`, black = 1 or `AML`). We note that this is a particular weakness of PCA because it is trying to build principal components that are linear combinations of *all* the features or genes. In this case, from a practical standpoint with over 3000 genes, there will likely be a chunk of genes that make up most of the variation in the data while there will be a large number of excessive or extra genes that are not really adding much to the variation in the data. We should note however though that that from the plot, the points were starting to be linearly separable after our first PCA iteration, which gives us hope that using sparse PCA will lead to even better results.

After applying PCA, we shift our attention to sparse PCA as a way to build upon our initial PCA slight success. We find that sparse PCA is quite effective at finding linearly separable structures as indicated by the plot where the red points are far off scattered to the left and the black more points are scattered to the right. Sparse PCA was in essence able to do feature selection and find the top genes that explain the variation in the data as opposed to using principal components that were linear combinations of *all* the features.

To add variation in our analysis, we extract the top 100 genes that will based on the sum of the scores for each gene from the first four principal components we got from sparse PCA. These will serve as our feature selection in the future predictive modeling work we do in this report later. In future work, one could play around with a different number of principal components instead of only $K = 4$, and could also select the features from the principal components in different ways, such as only using the first principal component to select the features or counting the number of times a feature appears among the principal components. Furthermore, from the scree plot that displays the percentage of variance accounted for by each principal component, it seems that we hit the elbow at about 10 components where adding more will have minimal affect. This aids us when we are determining how many features and what to use in the predictive modeling sections.

The predictive modeling results are pretty telling. By applying penalized logistic regression to the entire dataset, it does seem like the small sample size might be making the model overfit on the training dataset. When we apply XGBoost, we can see we are still getting about the same results as logistic regression in a 91%-95% accuracy rate on the testing dataset. On the XGBoost, it does appear that using the features we got from sparse PCA did not

affect the overall accuracy all that much. Thus even though the sparse PCA visually looked it was effective at splitting the data, it did not help much which accuracy. We hypothesize this is because our training dataset was only around 20 because the total sample size was about 70. We talk about the next steps and how to improve upon this finding.

In the future, we will likely need more data in order to build a model that can generalize to future, unseen data better. With smaller datasets like the one we used, it is very prone to overfitting on the training dataset. Also we should look at the results from sparse PCA to select features and under the principal components we got from sparse PCA better when selecting them. I believe combining these two traits will yield better results from the predictive models on the accuracy side. We should be encouraged that the visualizations from the sparse PCA helped a lot in separating the data between `ALL` and `AML` better. With more data and more sophisticated feature selection from the sparse PCA (and perhaps using more principal components in the analysis), we can improve our model by not only building a model that has a better accuracy score, but one that will be very good on new, unseen data.