Haozhen Ni
Huimin Zhang

Midterm Project Report

STATS 101C

Ashlyn Jew
Team: MLA

1 Introduction

The aim of this project is to build a model to accurately predict the class of cancer driver genes. The use of this model would be a powerful tool in cancer diagnosis. For this task, we were given a training dataset that included various features of the genes as predictors. Our goal was to classify the genes into three different classes of genes: tumor suppressor genes (TSGs), oncogenes (OGs), or neutral genes (NGs). We utilized various statistical methods such as linear discriminant analysis, K-nearest neighbors regression, and multinomial logistic regression for this classification and prediction challenge.

2 Methodology

2.1 Data Preprocessing

We first looked through the dataset to see if there were any obvious incomplete data, outliers, or noisy data. After searching through the dataset, we found observations that included 0 values in multiple columns where nonzero values typically appeared. We also created boxplots and observed that there were many outliers in the dataset. Therefore, we decided to test out various ways of dealing with the inconsistent data. The approaches we took to clean the data included deleting the outlying rows with multiple 0 values, replacing the 0 values and the outliers with the column's mean value, and replacing the 0 values with the column's median value. After building models using identical predictors on the different cleaned datasets, we found that not modifying the data actually performed slightly better, or were not significantly different, than if we did modify them. Therefore, we decided to deal with the outliers after choosing a model.

2.2 Model Description

Our best-performing model included 20 predictors. We selected our predictors by finding the predictors most positively correlated with a class of 1 (OG) and combined them with the predictors most positively correlated with a class of 2 (TSG). After combining these sets of predictors, we built a linear model with them and selected the predictors that had a variance inflation factor (VIF) of less than 5 in order to minimize multicollinearity within the model. Our final model includes the following predictors:

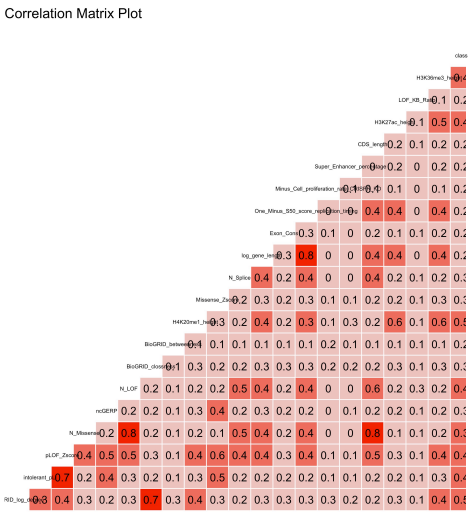

Figure 1: Non-Multicolinearity

Predictors	& Predictors
BioGRID_log_degree	N_Splice
intolerant_pLI	log_gene_length
pLOF_Zscore	Exon_Cons
N_Missense	One_Minus_S50_score_replication_timing
ncGERP	Minus_Cell_proliferation_rate_CRISPR_KD
N_LOF	Super_Enhancer_percentage
BioGRID_clossness	CDS_length
BioGRID_betweenness	H3K27ac_height
H4K20me1_height	LOF_KB_Ratio
Missense_Zscore	H3K36me3_height

Figure 2: Predictors for Final Model

With these predictors in our model, we used K-nearest neighbor (KNN), Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA), and Multinomial Logistic Regression (LR) classification methods to classify each observation in the test dataset. After many trials, we consistently found that LDA was the best classification method using our model.

2.3 Model Evaluation

The best evaluation metric value with this model is a Weighted Categorization Accuracy (WCA) value of 0.67486.

2.4 Model Performance Justification

Using 10-fold cross validation, we had a validation accuracy (1 - Validation Error Rate) of 0.932596. This means our validation error rate, which we want to minimize, is 0.067404. As shown in the figure below, we also plotted the Receiver Operating Characteristic (ROC) curve. The area under the curve is approximately 0.9, which indicates that LDA is a good classifier for our chosen model.

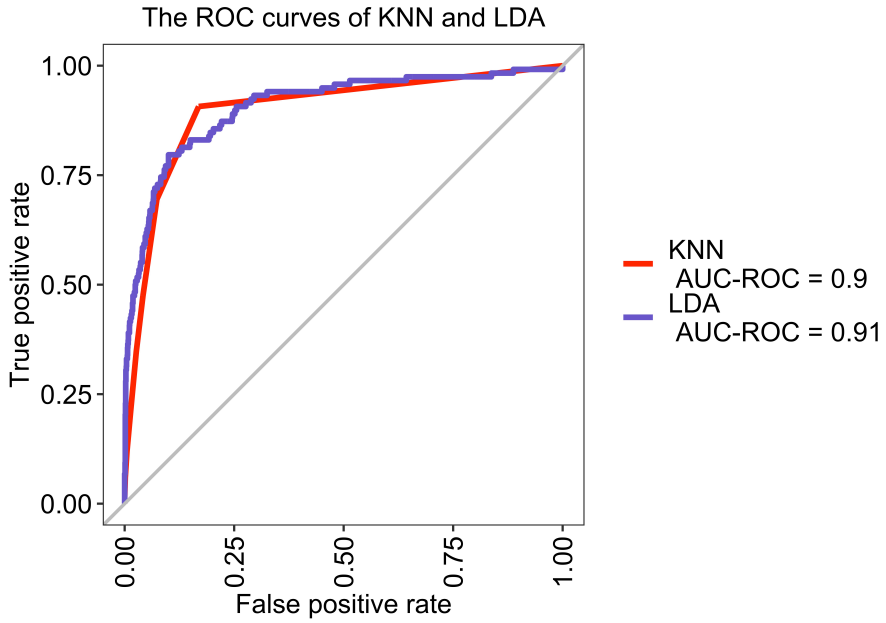

Figure 3: ROC curve

	AUC	ACC	FPR	TPR
k-Nearest Neighbors	0.90	0.9154212	0.005	0.1186441
Linear Discriminant Analysis	0.91	0.9361729	0.014	0.4237288

Figure 4: ROC table

We also evaluated our classification performance using a confusion matrix. Our resulting confusion matrix is as follows:

```
Confusion Matrix and Statistics

          Reference
Prediction   a   b   c
         a 845  39  28
         b   3   9  11
         c   4   2  11
```

Figure 5: Confusion-Matrix

3 Conclusion

Our best WCA value is 0.67486, which indicates that our model performed moderately well. We have used 33 submissions to test each variation of models we built, but found that we were only able to get a WCA score within the 0.6 - 0.7 range. Some decisions that may have contributed to this mediocre performance are not properly cleaning the data, finding correlation based on numeric class rather than factor class, and strict filtering. We believe that improvement in feature selection would lead to getting a better WCA score. Some areas that we can investigate are why choosing variables on absolute correlations result in poorer performance, how to better deal with multicollinearity other than using VIF, and testing if using more variables and less strict dimension reduction methods would result in better performance.

Statement of Contributions

Ashlyn Jew worked on writing the report and model comparison:

I tried building variations of our model using different classification methods and comparing whether it was a good decision to replace outlying 0 values with column medians, remove the outlying 0 values, or leave the data as-is. I also compared results when choosing variables using absolute values of correlations against non-absolute correlations. I also wrote a majority of the report, including the Introduction, Methodology, and Conclusion sections.

Huimin Zhang worked on coding and built models:

I tried to find predictors by using correlation, p-value, boxplots, and checking multicollinearity. I tried to compare the difference of using absolute value of correlation and not absolute value of correlation to find predictors. I tried to select predictors based on p-value ¡ 0.05, and then used VIF ¡ 5 to remove those high correlation predictors. I also tried to use VIF ¡ 5 to remove those high correlation predictors first, then chose predictors based on p-value ¡ 0.05 from those remaining predictors. I tried to split the data set to class 0/1 data set and class 0/2 data set and used Logistic Regression on both to find predictors that have p-value ¡ 0.5 in both models. Finally, I used a method that is based on the predictors having the top 49 values of correlation with class 1 and class 2 to find predictors.

Haozhen Ni worked on coding and data cleaning and make graphs:

In order to find the variable with the strongest correlation with "class", I tried to use ANOVA tables, correlation plots, scatter plots and also boxplots. In addition, I tried to process training data by replacing all 0 with mean and also finding outliers in all variables and then replacing them with mean. After finding variables that can be used by using aov() and cor() functions, I only retained variables where VIF ¡ 7 to reduce multicollinearity. After discovering that different predictors had different predictive abilities for class = 1 and class = 2, I divided data into class = 1 and class = 2 data sets for subsequent modeling. Also, I'm responsible for the graphs in the final report.

Appendix

```
set.seed(1)

cancer <- read.csv("training.csv")

# Find the predictors that have the top 49 values of positive
correlation with class 1 and 2

cor_1 <- order(cor(cancer$class == 1, cancer), decreasing =
TRUE)[2:50]
cor_2 <- order(cor(cancer$class == 2, cancer), decreasing =
TRUE)[2:50]

# The column indices of the predictors that both in the top 49 values
of correlation with class 1 and 2

cor_id <- cor_1[which(cor_1 %in% cor_2)]

# Create new dataset with predictors that have high positive
correlation with class 1 and 2

cor_cancer <- cancer[, c(cor_id, 99)]

# Create a new dataset that without multicollinearity
library(car)

cor_cancer_lm <- lm(class ~ . , data = cor_cancer)
summary(cor_cancer_lm)

# Select predictors with VIF < 5
cor_cancer_sb <- subset(cor_cancer, select =
c(which(vif(cor_cancer_lm) < 5)))
cor_cancer_sb <- data.frame(cor_cancer_sb, "class" = cancer$class)

cor_cancer_lm_sl <- lm(class ~ . , data = cor_cancer_sb)
summary(cor_cancer_lm_sl)

# Need to change the class level 0/1/2 to character to use caret
package

cor_cancer_sb$class <- as.factor(cor_cancer_sb$class)
levels(cor_cancer_sb$class) <- c("a","b","c")

# Split training dataset into 70 training, 30 validation
library(caret)
trainIndex <- createDataPartition(cor_cancer_sb$class, p = 0.7, list
= FALSE)

cancer.training <- cor_cancer_sb[trainIndex,]
cancer.test <- cor_cancer_sb[-trainIndex,]
```

```
# Conduct 10-fold Cross Validation
train_control <- trainControl(method = "cv", number = 10,
                              classProbs = TRUE,
                              savePredictions = TRUE)

# Fit a LDA model
LDAfit <- train(class ~ .,
                data = cancer.training, method = "lda",
                preProc = c("center", "scale"),
                trControl = train_control)

# Evaluate LDA using 10-fold Cross Validation
LDAfit

# Evaluate using ROC curve
library(MLeval)

res <- evalm(list(LDAfit), gnames = c('LDA'))
res$roc

# Evaluate using confusion matrix
predLDA <- predict(LDAfit, newdata = cancer.test)
confusionMatrix(data = predLDA, reference = cancer.test$class)

# Load the test data
cancer_test <- read.csv("test.csv")

# Output the test result using the LDA model
predLDA_TS <- predict(LDAfit, newdata = cancer_test)
cancer_test$class <- predLDA_TS

# Change the levels of class back to 0/1/2
levels(cancer_test$class) <- c(0, 1, 2)
LDA_output <- cancer_test[, c(1, 99)]
```