A Discussion of Statistical and Machine Learning Methods for determination of the most significant risk factors for developing COPD

Zoe Parker Cates (11182963)

Yaindrila Barua (11318333)

Leila Rabiei Fard (11301719)

Isaac Dante Asamoah (11319281)

Mohammad Mahmudul Huq (11243856)

11/28/2021

Table of Contents

1 Background

Chronic obstructive pulmonary disease (COPD) refers to a group of chronic and progressive lung diseases. Responsible for 6% of deaths in 2019, COPD is the third leading cause of death worldwide [1]. More relevantly, an estimated 16.6% of Canadians have COPD [2]. COPD is a progressive disease; symptoms including breathlessness, cough, and fatigue can greatly interfere with quality of life. While many risk factors are preventable, including exposure to airborne toxins, many risk factors such as asthma and genetic conditions are unavoidable [3].

Single nucleotide polymorphisms (SNPs) represent one form of an unavoidable genetic risk factor. While SNPs are abundant and mostly harmless within the human genome, studies have identified several which increase the risk of developing COPD [4, 5].

The aim of this study is to verify what lifestyle and genetic factors are most associated with COPD diagnosis and develop machine learning models that can accurately predict the presence of disease based on these factors.

2 Data Description [6]

The data set under consideration contains samples from individuals ranging in age from 40 – 69. The information collected concerns each sample's Asthma, COPD, and Cancer status, as well as several variables relevant to the risk factors for each disease.

The data set comes pre-divided into one training and one testing set. Our models were trained using the training data set provided. A summary of the data follows:

Training set: n = 112,151 participants

Variable	**Type**	**Possible Values**	**Mean**	**Median**	**SD**	**Variance**
Sex (F=64334, M=47817)	Binomial	0 or 1	N/A	N/A	N/A	N/A
Body Mass Index	Continuous	14.32 to 67.38	27.31	26.60	4.7099	22.1828
Age	Discrete	40 to 69	56.65	58.00	7.9200	62.7266
Smoking Status	Discrete	0, 1, or 2	N/A	N/A	N/A	N/A
Forced Expiratory Volume	Continuous	-4.8300 to 4.9960	0.4068	0.3860	1.0875	1.1826
Asthma Status	Binomial	0 or 1	N/A	N/A	N/A	N/A
COPD Status	Binomial	0 or 1	N/A	N/A	N/A	N/A
Cancer Status	Binomial	0 or 1	N/A	N/A	N/A	N/A
SNP1 through SNP500	Discrete	0, 1, or 2	N/A	N/A	N/A	N/A

In the cases of the binomial and discrete variables:

Sex:	Female = 0, Male = 1
Smoking Status:	Never = 0, Previous = 1, Current=2
Asthma Status:	No Asthma = 0, Asthma = 1
COPD Status:	No COPD = 0, COPD = 1
Cancer Status:	No Cancer = 0, Cancer = 1

The data set was cleaned such that the column "patid" (patient ID) and all samples containing a value of NA were omitted. We have also removed the columns for Asthma and Cancer statuses as this study only concerns COPD status.

2.1 Definition of outcome variable

The outcome variable is y: the presence or absence of COPD.

$$y = \begin{cases} 1 & if\ a\ person\ has\ the\ disease \\ 0 & if\ a\ person\ \ does\ not\ have\ the\ disease \end{cases}$$

2.2 Definition & Selection of predictor variables

505 predictor variables are included:

Age (x_1):	The age of the individual in years
Sex (x_2):	The sex of the individual
Smoking Status (x_3):	Whether the individual is an active smoker
Forced expiratory volume (x_4):	The volume of air the individual can expel in a breath
Body mass index (x_5):	The individual's weight (kg) divided by their height (m)
SNP1-500 ($x_{6\text{-}500}$):	Top 500 genotyping variants (snips)

These predictors may be divided into two groups: Clinical and Genetic. Clinical features include Sex, BMI, FEV1Z, Smoking Status, and Age. As shown in Figures 1 and 2, the highest correlation can be seen between COPD status and FEV1Z. In contrast, the lowest correlation is shown between COPD status and Sex.

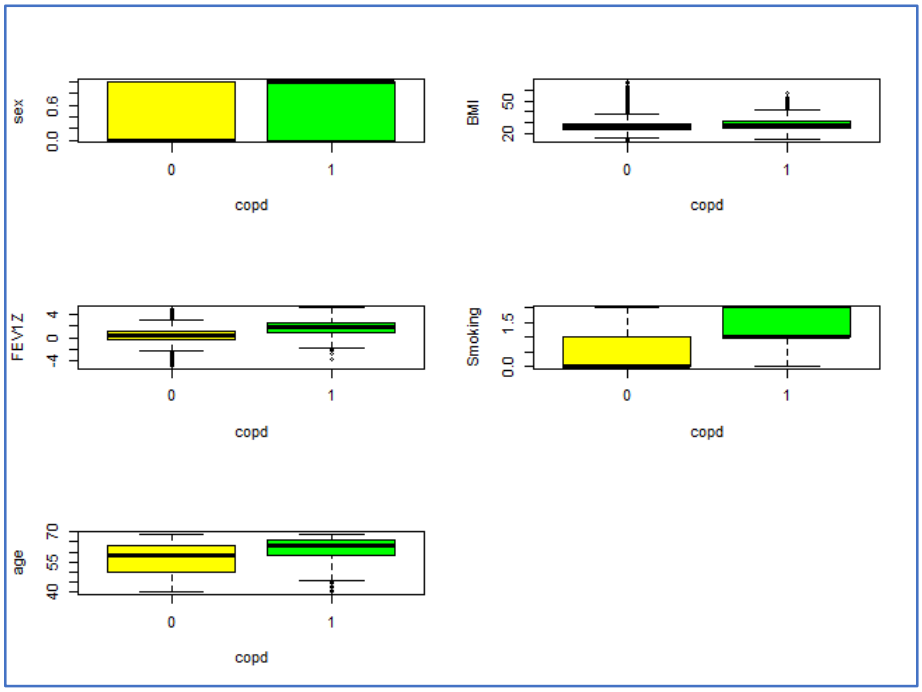


Figure 1: Association between clinical features and COPD.

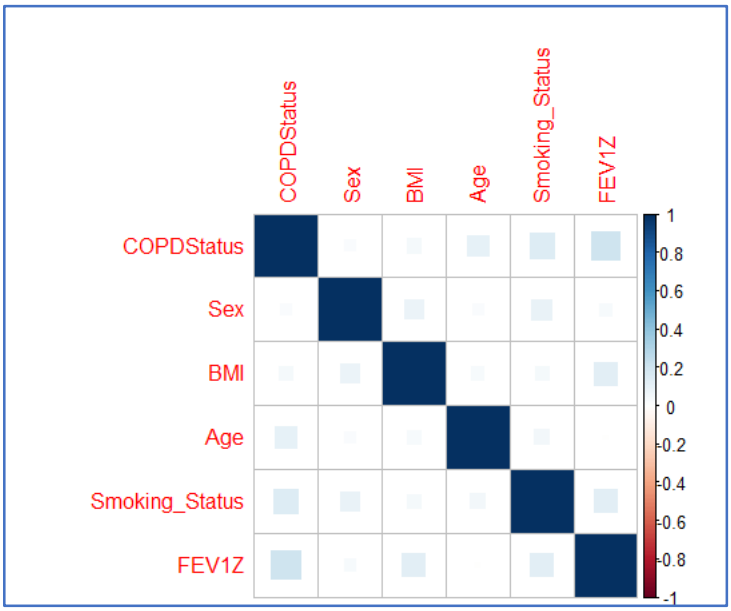


Figure 2: Correlation between clinical features and COPD.

500 SNPs are included as Genetic features. Given the large number of observations and the large number of genetic features, we will investigate the SNPs that are the strongest predictors of COPD status and exclude the rest from our analysis.

We hypothesize that Age, Sex, Smoking Status, Forced Expiratory Volume (FEV1), and Body Mass Index (BMI) are highly correlated with the outcome variable (absence or presence of COPD). However, genetics may also contribute to COPD onset in an individual. SNP variables that contribute to an accurate prediction are included in the final model.

2.3 Outcome variable

The outcome variable in the final project is y: the presence or absence of COPD:

$$y_i = \begin{cases} 1 & if\ a\ person\ has\ the\ disease \\ 0 & if\ a\ person\ \ does\ not\ have\ the\ disease \end{cases}$$

3 Statistical Methods

In this section we consider two statistical methods: logistic regression and classification tree.

3.1 Logistic Regression

Logistic regression is an effective means for classifying data. In this section, we apply a logistic model to predict the presence or absence of COPD.

3.1.1 Significant SNPs

First, we will identify significant SNPs. We will use a t-test and a logistic model to find SNPs that reliably predict COPD.

3.1.1.1 Significant SNPs identified by T-test

If we apply a t-test to the data, we find 20 significant SNPs: SNP29, SNP62, SNP144, SNP164, SNP210, SNP217, SNP220, SNP274, SNP320, SNP332, SNP333 SNP336, SNP337, SNP338, SNP339, SNP340, SNP341, SNP342, SNP393, and SNP458.

3.1.1.2 Significant SNPs identified by Logistic Model

If we apply a logistic model to the data, we find 27 significant SNPs: SNP28, SNP29, SNP56, SNP62, SNP106, SNP143, SNP144, SNP164, SNP205, SNP217, SNP220, SNP222, SNP229, SNP269, SNP320, SNP332, SNP333, SNP336, SNP337, SNP338, SNP339, SNP340, SNP341, SNP342, SNP393, SNP442, and SNP458.

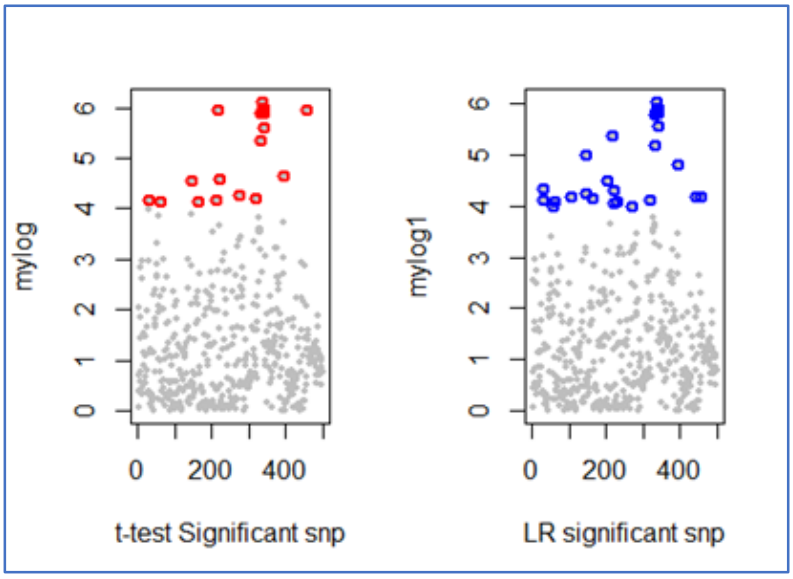


Figure 3: Significant SNPs identified by t-test (left) and logistic regression (right).

The results from the t-test and logistic model have 18 SNPs in common. SNP210 and SNP274 are uniquely identified by the t-test. SNP28, SNP56, SNP106, SNP143, SNP205, SNP222, SNP229, SNP269, and SNP442 are unique to the logistic model.

3.1.2 Fitting the Logistic Model

Let us define the Clinical variables x_1, x_2, x_3, x_4, and x_5 as age, sex, smoking status, FEV1Z, and BMI, respectively. We will include these Clinical variables as well as the significant SNPs in the logistic model. The model will be trained using 5-fold cross-validation.

3.1.2.1 Using significant SNPs taken from t-test to fit the model

Let x_6^t to x_{25}^t represent the 20 SNPs identified by the t-test:

x_6^t =SNP29, x_7^t =SNP62, x_8^t =SNP144, x_9^t =SNP164, x_{10}^t =SNP210, x_{11}^t =SNP217, x_{12}^t =SNP220, x_{13}^t =SNP274, x_{14}^t =SNP320, x_{15}^t =SNP332, x_{16}^t =SNP333 x_{17}^t =SNP336, x_{18}^t =SNP337, x_{19}^t =SNP338, x_{20}^t =SNP339, x_{21}^t =SNP340, x_{22}^t =SNP341, x_{23}^t =SNP342, x_{24}^t =SNP393, x_{25}^t =SNP458

The Clinical variables are also included. The standard logistic regression function for predicting the outcome given the above 25 predictors is a curve defined as

$$P = \frac{\exp(y)}{1 + \exp(y)}$$

Where

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6^t + \cdots + \beta_{25} x_{25}^t$$

and

P is the probability of the event occurring given the $x_1, x_2, x_3, x_4, x_5, x_6^t, \ldots, x_{25}^t$.

Mathematically, this is written as $P(COPD = 1 | x_1, x_2, x_3, x_4, x_5, x_6^t, \ldots, x_{25}^t)$.

Now the p-values can be used to determine which variables are significant. We fit the model as follows:

$$\begin{aligned} y = (-1.116e+01) &+ (9.792\text{e}-02\ )\, x_1 + (8.424\text{e}-01)x_3 + (9.366e-01)x_4 \\ &+ (2.550e-02)x_5 + (-1.249\text{e}-01)x_7^t + (1.551\text{e}-01)x_8^t \\ &+ (-1.246\text{e}-01)x_9^t + (-1.915\text{e}-01)x_{10}^t + (-1.304\text{e}-01)\, x_{11}^t \\ &+ (-1.220\text{e}-01)x_{12}^t + (-3.568\text{e}-01)x_{13}^t + (-9.256\text{e}-02)x_{14}^t \\ &+ (1.383\text{e}-01)x_{24}^t + (-5.342\text{e}-01)\, x_{25}^t \end{aligned}$$

Confusion matrix

To determine the accuracy of this model fitted using the Clinical variables and the significant SNPs identified by a t-test, we compute the confusion matrix:

To determine the percentage of correct predictions (accuracy), the following formula is used:

$$\frac{54278 + 120}{54278 + 120 + 63 + 1614} = 97\%$$

The overall predictions of the model are 97% accurate.

Additionally, the model correctly predicted negatives (COPD Status = 0) 99.9% of the time:

$$\frac{54278}{54278 + 63} = 99.9\%$$

However, the model only correctly predicted the positives 7% of the time:

$$\frac{120}{120 + 1614} = 7\%$$

Therefore, this model will not reliably predict a positive COPD diagnosis but can reliably predict a negative COPD status.

ROC plot

The receiver operating characteristics (ROC) plot displays the false positive rate vs the true positives rate of the model's predictions, as in Figure 4. The area under the curve (AUC) is an effective measurement of the overall performance [7]. In this case a large AUC is observed, indicating a good performance.

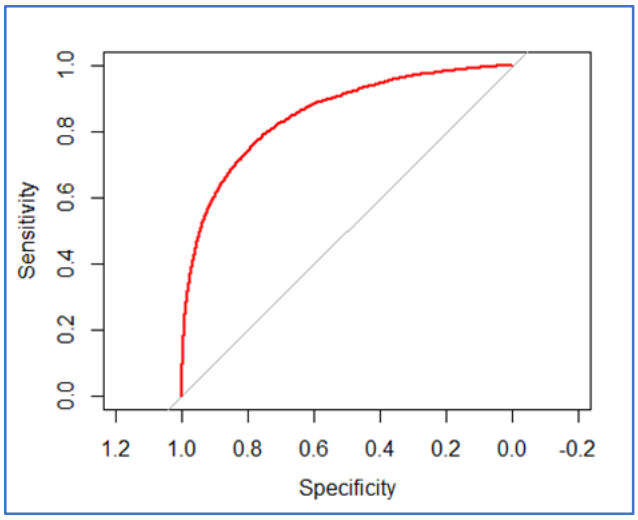


Figure 4: AUC for the model fitted using SNPs identified by the t-test.

3.1.2.2 Using significant SNPs taken from logistic model to fit the model

Let x_6^l to x_{32}^l represent 27 significant genotyping variants (SNPs) found by logistic model:

x_6^l =SNP28, x_7^l =SNP29, x_8^l =SNP56, x_9^l =SNP62, x_{10}^l =SNP106, x_{11}^l =SNP143, x_{12}^l =SNP144, x_{13}^l =SNP164, x_{14}^l =SNP205, x_{15}^l =SNP217, x_{16}^l =SNP220 x_{17}^l =SNP222, x_{18}^l =SNP229, x_{19}^l =SNP269, x_{20}^l =SNP320, x_{21}^l =SNP332, x_{22}^l =SNP333, x_{23}^l =SNP336, x_{24}^l =SNP337, x_{25}^l =SNP338, x_{26}^l =SNP339, x_{27}^l =SNP340, x_{28}^l =SNP341, x_{29}^l =SNP342, x_{30}^l =SNP393, x_{31}^l =SNP442, x_{32}^l =SNP458

The 5 Clinical variables are also included. The logistic regression function for predicting the outcome given the 32 predictors is a curve defined as

$$P = \frac{\exp(y)}{1 + \exp(y)}$$

Where

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 + \beta_6 x_6^l + \cdots + \beta_{25} x_{32}^l$$

and

P is the probability of event occurring given the $x_1, x_2, x_3, x_4, x_5, x_6^t, \dots, x_{25}^t$.

Mathematically, this is written as $P(COPD = 1 | x_1, x_2, x_3, x_4, x_5, x_6^l, \dots, x_{32}^l)$.

The p-values can now be used to determine which variables are significant. We fit the model as follows:

$$\begin{aligned} y = (-1.154\text{e}+01) &+ (9.796\text{e}-02)\,x_1 + (8.416\text{e}-01)x_3 + (9.367\text{e}-01)x_4 \\ &+ (2.544\text{e}-02)x_5 + (1.312\text{e}-01)x_8^l + (-1.266\text{e}-01)\,x_9^l \\ &+ (8.150\text{e}-01)\,x_{10}^l + (-1.233\text{e}-01)x_{13}^l + (3.302\text{e}-01)\,x_{14}^l \\ &+ (-1.304\text{e}-01)x_{15}^l + (-1.208\text{e}-01)\,x_{16}^l + (1.521\text{e}-01)x_{17}^l \\ &+ (4.745\text{e}-01)x_{18}^l + (3.003\text{e}-01)x_{19}^l + (-9.329\text{e}-02)x_{20}^l \\ &+ (1.368\text{e}-01)x_{30}^l + (4.456\text{e}-01)x_{31}^l + (-5.374\text{e}-01)x_{32}^l \end{aligned}$$

Confusion matrix

To determine the accuracy of this model fitted using the Clinical variables and the significant SNPs identified by the logistic model, we compute the confusion matrix:

To determine the percentage of correct predictions (accuracy), the following formula is used:

$$\frac{54272 + 125}{54272 + 125 + 69 + 1609} = 97\%$$

Additionally, the model correctly predicts the negatives (COPD Status = 0) 99.9% of the time:

$$\frac{54272}{54272 + 69} = 99.9\%$$

Similar to the model trained using the SNPs identified by t-test, the model using the SNPs identified by logistic regression only predicted the positives correctly 7% of the time:

$$\frac{125}{125 + 1609} = 7\%$$

ROC plot

Figure 5 displays the ROC for the model fitted with the SNPs found by the logistic regression. Again, a large AUC is observed, indicating a good performance.

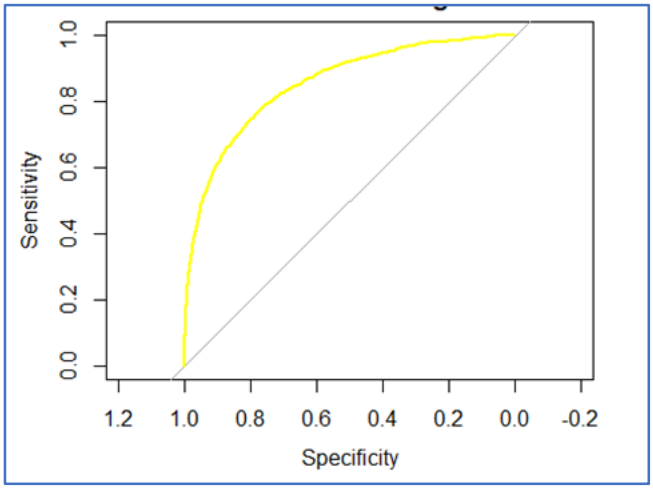


Figure 5: AUC for the model fitted using SNPs identified by logistic regression.

3.1.2 Comparison of two AUCs

An AUC of 0.5 means that a model cannot make predictions more accurately than random guessing. The AUCs for our model using SNPs identified by the t-test and for our model using SNPS identified by logistic regression both have an AUC of 0.815, indicating a good performance.

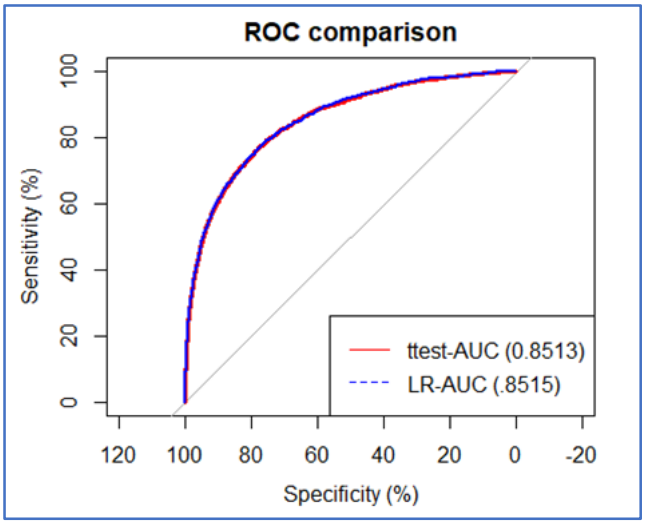


Figure 6: Comparison of the two AUCs from Figures 4 (red) and 5 (blue).

3.1.3 Logistic Regression Visualization

Visualization of regression classification is an important step for understanding the results of the model. Figure 7 contains plots of the top 3 predictors identified by the model: FEV1Z, smoking status, and age.

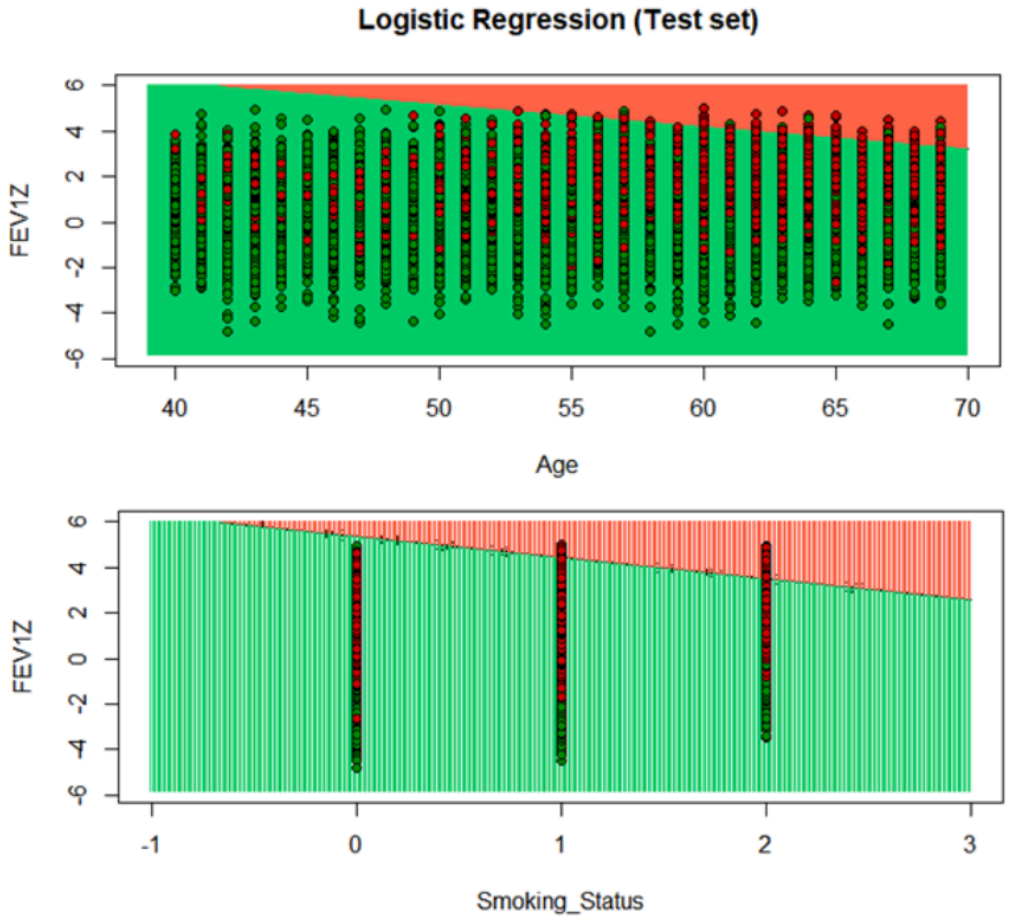


Figure 7: Visualization of logistic regression results using significant clinical variables (a) age and smoking status (b) FEV1Z and smoking status.

As categorized by logistic regression, the green and red areas in Figure 7 represent the areas for non-COPD status and COPD status, respectively. Similarly, the green and red dots depict COPD negative and COPD positive cases from the test set, respectively. In both cases, logistic regression predicts COPD negative status (COPD status = 0) with a high degree of accuracy – there are rarely any green dots located in the red zone. However, many red dots appear in the green zone, indicating the presence of many false negative results. The plots also demonstrate that COPDStatus increases with FEV1Z, Smoking_status and Age. Given that the logistic regression model is linear, there is a clear linear separation between the red and green zones despite the abundant number of red dots in the green zone. Application of a non-linear model to this dataset would result in better prediction of COPD.

3.2 Classification Tree

We will now apply classification tree algorithms to verify which Clinical and Genetic variables are the best predictors of COPD diagnosis. Below, we discuss an implementation of random forest as well as an implementation of a classification tree and assess the performance of each.

3.2.1 Random Forest using R library(randomForest) for Classification

The parameters of this random forest model are as follows: ntree=100, maxnodes=4, mtry=30. These values are restrictive, but due to the large size of the dataset, increasing these values results in too long of a computation time.

For this model, the training_data was randomly split into two halves. One half was used for training, while the other was used for testing. The resulting confusion matrix is shown below:

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 54342  1734
         1     0     0

               Accuracy : 0.9691
                 95% CI : (0.9676, 0.9705)
    No Information Rate : 0.9691
    P-Value [Acc > NIR] : 0.5064
```

This random forest model had an overall accuracy of ~97%. The prediction accuracy for negative samples (COPD status = 0) was 100%, and the prediction accuracy for positive samples was 0%. Additionally, the accuracy is not significantly better than the no information rate (NIR). The NIR is the proportion of the data belonging the majority class – in this case, 97% of our data belongs to the negative (COPD status = 0) class [8]. This result leads us to the next section: *Problem Solving using Undersampling*.

3.2.2 Problem Solving using Undersampling

From the previous section, the random forest model received a 97% accuracy. Consider the following scenario: in our training dataset, there are ~55000 negatives and ~1700 positives. If we assign all 56700 values a negative status, we will be correct 55000/56700*100 = 97% of the time. If we assign all 56700 values a positive status, we will be correct 1700/56700*100 = 3% of the time.

Given the fact that the percentages are the same for the random forest and the imagined scenario, as well as the fact that the random forest model only predicted negative outcomes (COPD status = 0), we suspect that the proportion of negatives to positives in the training data is inadequate; that is, there are too many negatives compared to positives and random forest is basing too much of its decision-making on the negative samples.

The solution to this is undersampling. We undersample the negative training data in order to create a training data set with the same amount of negative samples as positive samples, which we then use to train a new random forest model. The new model has the following confusion matrix:

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1135  657
         1  630 1108

               Accuracy : 0.6354
                 95% CI : (0.6193, 0.6513)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16
```

Intuitively, this result makes more sense. All 4 quadrants of the confusion matrix are represented, meaning the model is now predicting both negatives and positives instead of just negatives. This model has an accuracy of 63.5%. Additionally, the accuracy is significantly different than the NIR. While the overall accuracy is now worse than the NIR, the model better predicts positive diagnoses than the logistic model from section 3.1.

To improve this model, we increase ntrees and maxnodes. This is now possible because undersampling has resulted in a smaller dataset, meaning the computational time required to build the larger model is manageable. We generate the following confusion matrix:

```
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1331  460
         1  434 1305

               Accuracy : 0.7467
                 95% CI : (0.7321, 0.761)
    No Information Rate : 0.5
    P-Value [Acc > NIR] : <2e-16
```

Increasing the number of nodes has improved the accuracy by nearly 10%. Additional pruning of the model could be carried out to improve the accuracy even more. However, we are now interested in the features extracted by random forest. As seen in Figure 8, the top three variables in all random forest models are FEV1Z, smoking status, and age. This result matches the most important predictors identified during logistic regression (Figure 7).

To conclude this section, undersampling did result in a model that better predicted positive COPD diagnoses. However, when we undersample, we throw away a large portion of usable data. The result is a tradeoff between overall accuracy and prediction accuracy for an individual class (in this case, positive COPD samples).

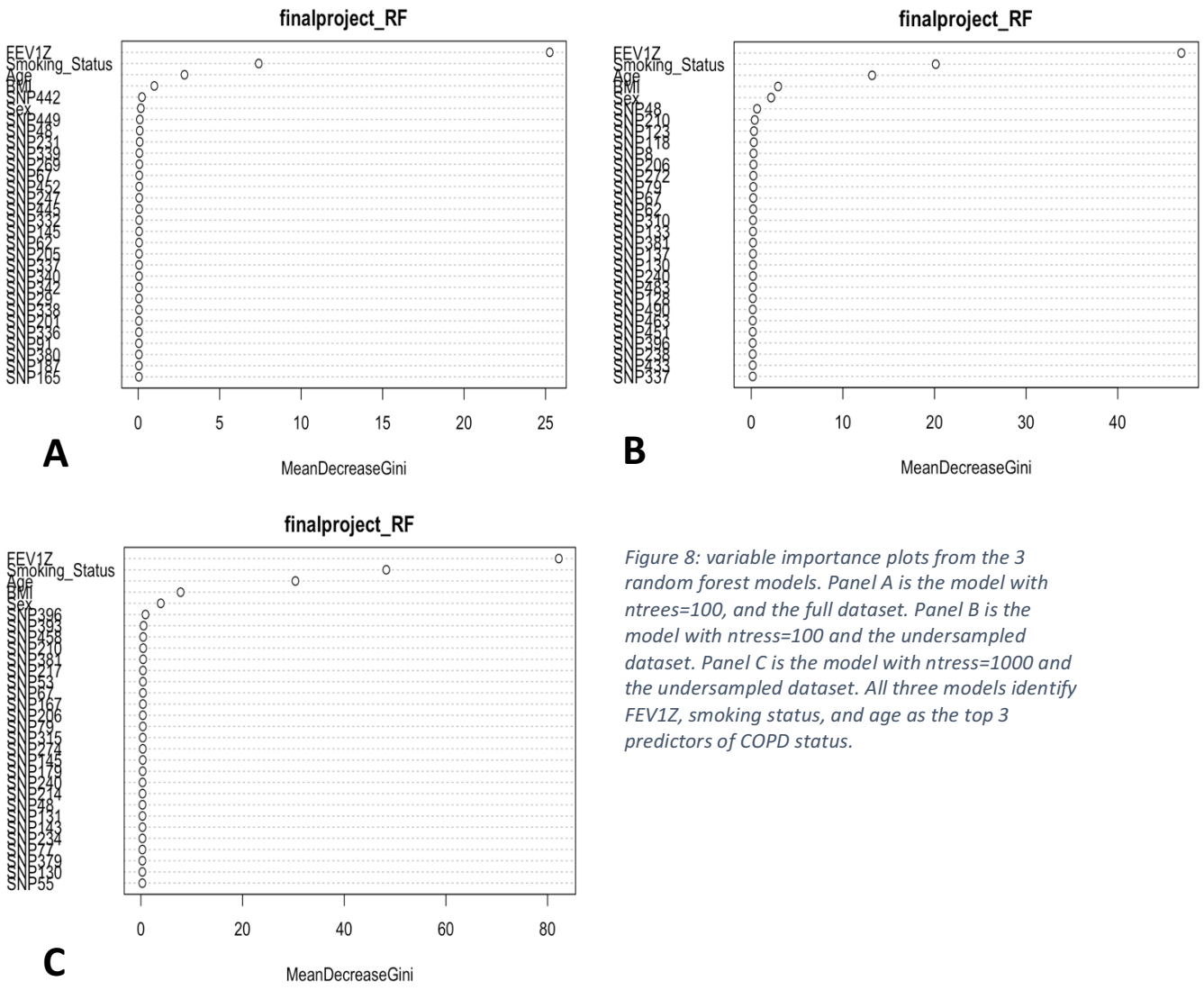


Figure 8: variable importance plots from the 3 random forest models. Panel A is the model with ntrees=100, and the full dataset. Panel B is the model with ntress=100 and the undersampled dataset. Panel C is the model with ntress=1000 and the undersampled dataset. All three models identify FEV1Z, smoking status, and age as the top 3 predictors of COPD status.

3.2.3 Classification tree using R library(rparts)

Library(randomForest) in R creates an effective model and is simple enough to use. However, the implementation is very slow. As the dataset grows larger, the time required to build and prune the random forest model quickly becomes unmanageable. Therefore, we discuss a second implementation of a classification tree using library(rparts).

First, we fit an unpruned tree using all the predictors. The unpruned model included 124 out of 505 variables in tree construction. After using the model to make predictions and computing the confusion matrix, the overall accuracy is 96.11%, the sensitivity is 98.70 %, and the specificity is 14.88%.

Figure 9 contains a plot of the complexity parameter (CP) with the corresponding tree size against relative error. This plot visualizes the number of trees used at each CP and the corresponding error produced, informing us which CP is best for model optimization. More than 206 trees were used when CP=0. The relative error is lowest when CP=0.002784.

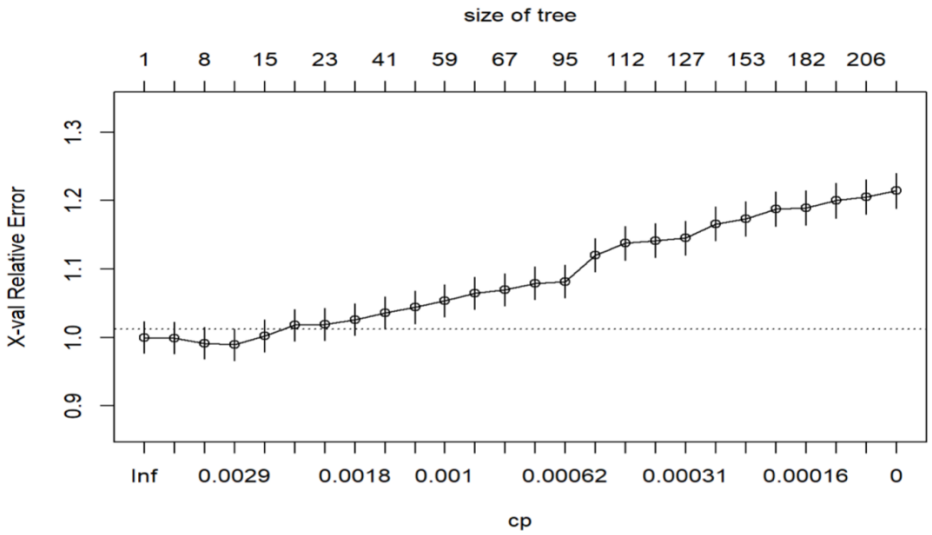


Figure 8: the CP plot for the unpruned model.

Setting CP=0.002784 results in an optimized model that uses only 37 out of 505 variables for tree construction. As is consistent with all other model results in this paper, the 3 most important predictors identified in the below variable importance plot are FEV1Z, smoking status, and age. The confusion matrix for this model reports an overall accuracy of 96.95%, a sensitivity of 99.83%, and a specificity of 6.69 %. Therefore, the pruned model predicts absence of COPD with higher accuracy than the unpruned model. However, neither model is a good predictor of COPD presence.

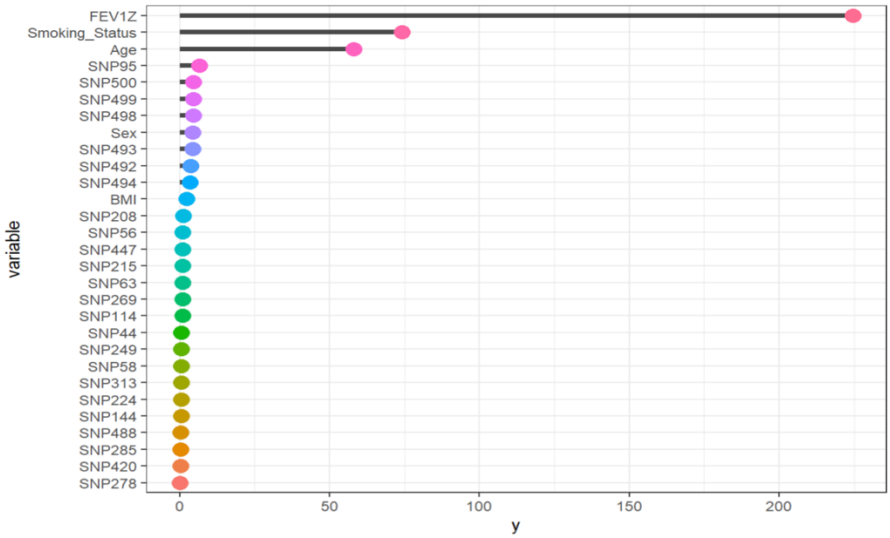


Figure 9: the variable importance plot for the pruned rparts model.

4 Discussion

There are several key outcomes for this project. First, both the t-test and logistic regression are effective methods for determining important predictors of a given response. In our project, 18/22 significant t-test SNPs and 18/27 significant logistic regression SNPs were identical. Additionally, the SNPs predicted by both the t-test and logistic regression contributed to an accurate model.

The library(randomForest) model provided easily interpretable output. However, this implementation is very slow. The amount of time required to build, prune, and optimize a random forest model using this library was unmanageable with the size of this entire data set. Additionally, the first tree built using this library was a poor predictor of COPD presence – in fact, it predicted COPD absence for every test sample. This issue was mitigated using undersampling, which resulted in lower overall accuracy, but much better accuracy when predicting COPD presence. Given unlimited computation resources, we could train and prune a library(randomForest) model using a dataset with less drastic undersampling to obtain higher prediction accuracy.

The library(rparts) implementation of a classification tree is a faster algorithm than library(randomForest), and therefore is computationally less expensive. This makes it a feasible option for large datasets requiring feature extracting and classification. As well, this package comes with a variety of clear and attractive plotting options. The model built using this package achieved accuracy similar to that of the logistic and random forest model (without undersampling).

With the above information in mind, future work would involve building a classification tree using rparts, as the implementation requires less time and memory than random forest. However, some degree of undersampling would be applied to the dataset to ensure reasonable prediction capability for COPD presence. Regarding this particular dataset, it is important to note the tradeoff between overall prediction accuracy and accurate prediction of COPD presence.

5 Conclusion

Numerous studies have utilized machine learning methodologies for biomarker detection and disease prediction. As demonstrated by our report, machine learning models have the potential to rapidly detect diseases based on common risk factors using methods that would be otherwise impossible to carry out by hand. Most exciting is the fact that all models fitted in this project agree that the most important predictors of COPD are the Clinical variables, along with a handful of SNPs. Regardless of the model used, the top 3 predictors of COPD were FEV1Z, smoking status, and age. We therefore conclude this project, having successfully isolated the strongest predictors of COPD.

Citations

[1] World Health Organization. (2021, June 22). *Chronic obstructive pulmonary disease (COPD)*. https://www.who.int/. https://www.who.int/news-room/fact-sheets/detail/chronic-obstructive-pulmonary-disease-(copd)

[2] Evans, J., Chen, Y., Camp, P. G., Bowie, D. M., & McRae, L. (2014). Estimating the prevalence of COPD in Canada: Reported diagnosis versus measured airflow obstruction. *Health reports, 25*(3), 3–11.

[3] World Health Organization. (2020, December 9). *The top 10 causes of death.* https://www.who.int/. https://www.who.int/news-room/fact-sheets/detail/the-top-10-causes-of-death

[4] Li, L. J., Gao, L. B., Lv, M. L., Dong, W., Su, X. W., Liang, W. B., & Zhang, L. (2011). Association between SNPs in pre-miRNA and risk of chronic obstructive pulmonary disease. *Clinical Biochemistry, 44*(10–11), 813–816. https://doi.org/10.1016/j.clinbiochem.2011.04.021

[5] Kim, W. J., Hoffman, E., Reilly, J., Hersh, C., DeMeo, D., Washko, G., & Silverman, E. K. (2010). Association of COPD candidate genes with computed tomography emphysema and airway phenotypes in severe COPD. *European Respiratory Journal, 37*(1), 39–43. https://doi.org/10.1183/09031936.00173009

[6] This section *Data Description* was initially written for our October Project Proposal and reworked for this Final Report.

[7] James, G., Witten, D., Hastie, T., & Tibshirani, R. (2021). *An Introduction to Statistical Learning: with Applications in R* (2nd ed. 2021 ed.). Springer.

[8] Signorell, A. (2021, November 23). *Confusion Matrix And Associated Statistics*. https://Rdrr.Io/. https://rdrr.io/cran/DescTools/man/Conf.html

Additionally, we would like to acknowledge the lecture notes and R examples provided to us throughout this course. We relied on this material for background knowledge about statistical techniques, machine learning methods, and applications in R.

Supplementary Material

In this section, we include our R code for the t-test, logistic regression, and two implementations of random forest.

Appendix A – Logistic Regression R Code

Project R files

Yaindrila Barua (11318333)

11/24/2021

Reading data and sorting out the variables we will use.

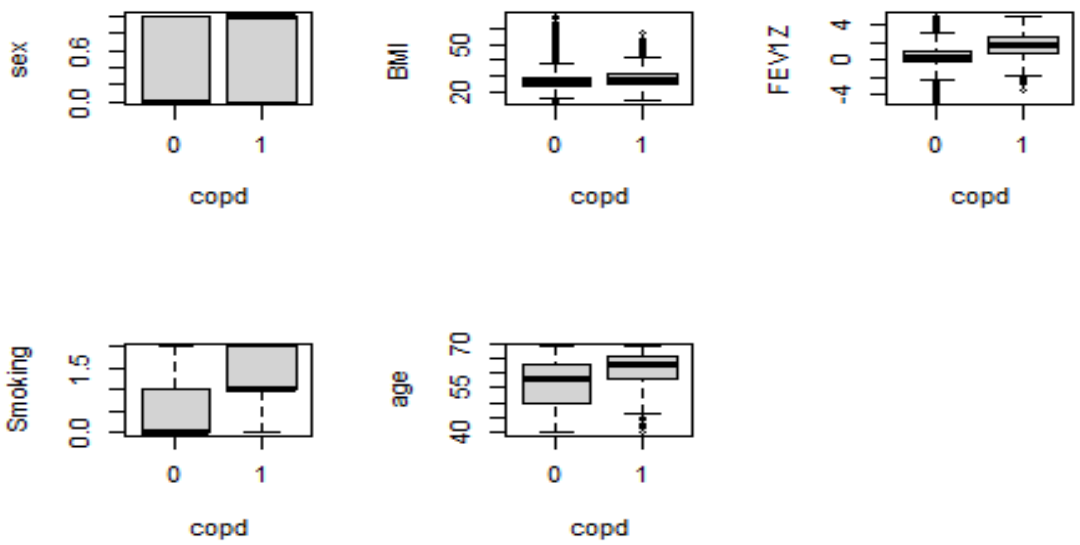
```
set.seed(1031)
# function to report significant snps based on Bonferroni adjustment
Sig.snp <- function(mypvalues){
  inn <- length(mypvalues)
  iID <- (mypvalues < 0.05/inn)  #Bonferroni adjustment
  if (sum(iID) > 0 ) {
    return(list(pvalue = mypvalues[iID], positions = c(1:inn)[iID], ids = iID
))}
  else{ print("No significant snp found!") }
}

load("E:/USASK/stat 846/Project/train.rda")
load("E:/USASK/stat 846/Project/test.rda")
any(is.na(mytrain)) # No missing values

## [1] FALSE

data=mytrain[,-(6:8)] # excluding the disease columns
df<-data.frame(mytrain[,7],data)
train=df[,-507] # altered train data for analysis
cor(train[,1:6]) ## Clinical data

##                COPDStatus        Sex        BMI          Age Smoking_Statu
s
## COPDStatus     1.00000000 0.02828802 0.04603819  0.107479211     0.1427130
1
## Sex            0.02828802 1.00000000 0.08827282  0.027685997     0.0923058
4
## BMI            0.04603819 0.08827282 1.00000000  0.038842402     0.0432793
6
## Age            0.10747921 0.02768600 0.03884240  1.000000000     0.0524604
4
## Smoking_Status 0.14271301 0.09230584 0.04327936  0.052460444     1.0000000
0
## FEV1Z          0.20485604 0.03189211 0.12647549 -0.003188012     0.1279488
0
##                       FEV1Z
## COPDStatus      0.204856042
## Sex             0.031892112
## BMI             0.126475490
```

```
## Age             -0.003188012
## Smoking_Status   0.127948795
## FEV1Z            1.000000000

#head(train)

snp=train[,7:506] ## extracting the snp's

snpcol<-colnames(train[,7:506])
copd<-train[,1]
summary(mytrain[,1:8])

##       Sex              BMI             Age        Smoking_Status
##  Min.   :0.0000   Min.   :14.32   Min.   :40.00   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:24.07   1st Qu.:50.00   1st Qu.:0.0000
##  Median :0.0000   Median :26.60   Median :58.00   Median :0.0000
##  Mean   :0.4264   Mean   :27.31   Mean   :56.65   Mean   :0.5393
##  3rd Qu.:1.0000   3rd Qu.:29.71   3rd Qu.:63.00   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :67.38   Max.   :69.00   Max.   :2.0000
##      FEV1Z          AsthmaStatus      COPDStatus       CancerStatus
##  Min.   :-4.8300   Min.   :0.0000   Min.   :0.00000   Min.   :0.000000
##  1st Qu.:-0.2930   1st Qu.:0.0000   1st Qu.:0.00000   1st Qu.:0.000000
##  Median : 0.3860   Median :0.0000   Median :0.00000   Median :0.000000
##  Mean   : 0.4068   Mean   :0.1347   Mean   :0.03148   Mean   :0.005849
##  3rd Qu.: 1.0790   3rd Qu.:0.0000   3rd Qu.:0.00000   3rd Qu.:0.000000
##  Max.   : 4.9960   Max.   :1.0000   Max.   :1.00000   Max.   :1.000000

table(train[,1])

##
##      0      1
## 108621   3530
```

Associaction of the clinical data to the COPD

Visualization for clinical features:

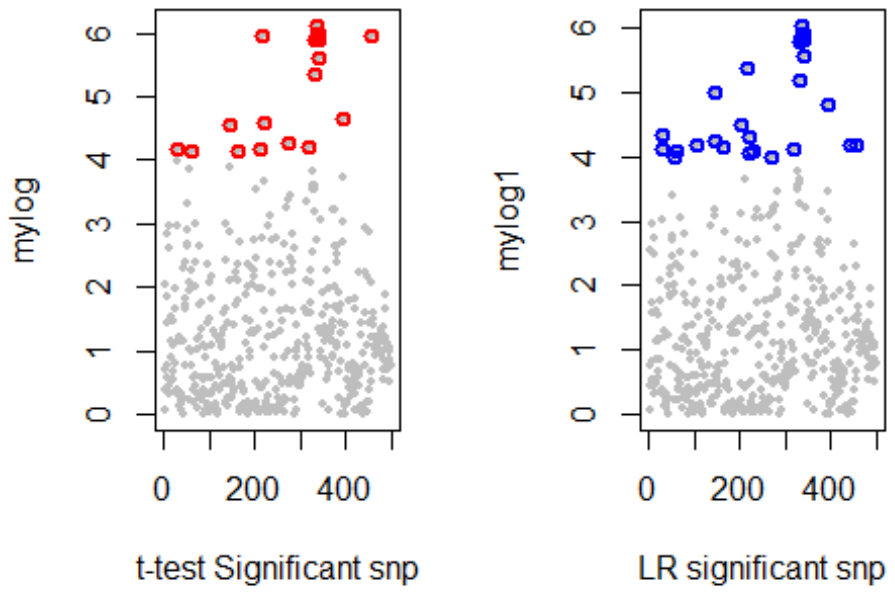
par(mfrow=c(3,3)) boxplot(train$Sex\ train$COPDStatus, xlab="copd", ylab="sex")
boxplot(train$BMI\ train$COPDStatus, xlab="copd", ylab="BMI")
boxplot(train$FEV1Z\ train$COPDStatus, xlab="copd", ylab="FEV1Z")
boxplot(train$Smoking_Status\ train$COPDStatus, xlab="copd", ylab="Smoking")
boxplot(train$Age\ train$COPDStatus, xlab="copd", ylab="age")



finding significant snp's by ttest and logistic model

```
#For t-test to find significant snp's
myttest <- data.frame(snpcol, myp = rep(NA, 500))

for (ii in 1:500){
  myttest[ii, 2] <- t.test(train[,snpcol[ii]]~copd )$p.value
}
myresult1 <- Sig.snp(myttest[, 2])
mylog <- -log(myttest[, 2], base = 10)
sum(myresult1$ids)

## [1] 20

snp.order <- myresult1$positions[order(myresult1$pvalue)]
print(snp.order) #20 genes

##  [1] 338 339 217 458 336 337 342 341 333 340 332 393 220 144 274 320  29 2
10 164
## [20]  62

#logistic regression
mylogist <- data.frame(snpcol, myp = rep(NA, 500))

for (jj in 1:500){
```

```
  myfit <- glm(copd ~ train[, snpcol[jj]], family="binomial")
  mylogist[jj, 2]<- coef(summary(myfit))[2, 4]
}
myresult2 <- Sig.snp(mylogist[, 2])

sum(myresult2$ids)

## [1] 27

snp.order.glm <- myresult2$positions[order(myresult2$pvalue)]
print(snp.order.glm)  #27 genes

##  [1] 338 339 336 337 342 341 333 340 217 332 144 393 205  29 220 143 106 4
42 458
## [20] 164  28 320  62 229 222  56 269

#plot the significant snps
mylog1 <- -log(mylogist[, 2], base = 10)

par(mfrow=c(1,2))
plot(mylog, pch = 19, cex = 0.5, col = "gray", xlab="t-test Significant snp")
points(myresult1$positions, mylog[myresult1$ids], col = "red", lwd=2)
plot(mylog1, pch = 19, cex = 0.5, col = "gray", xlab="LR significant snp")
points(myresult2$positions, mylog1[myresult2$ids], col = "blue", lwd=2)
```



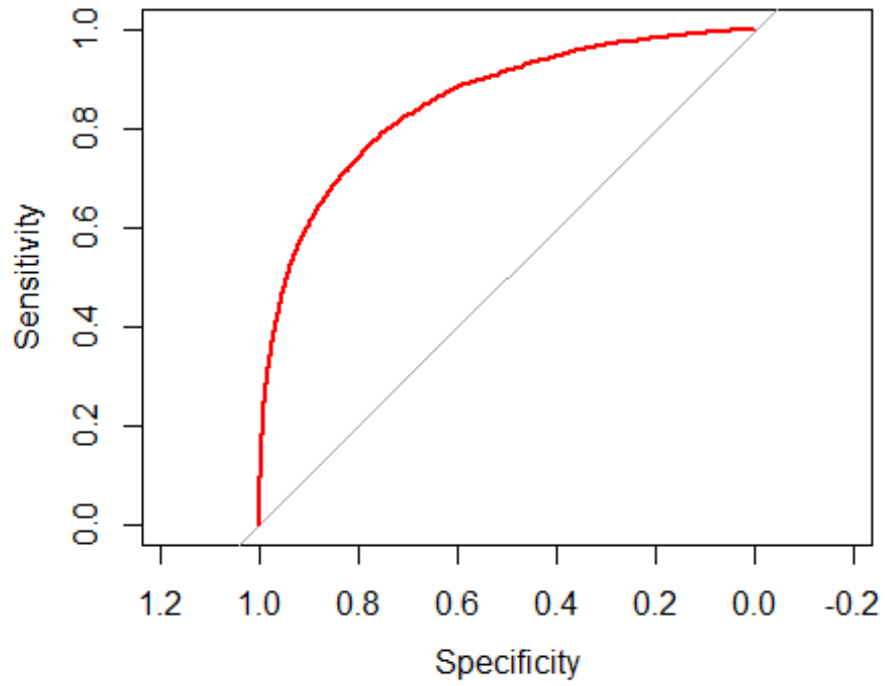
Training and test data set

```
##Spliting data into 50-50%

set.seed(2021)
split = createDataPartition(y=train[,1],p = 0.5,list = F)
train_data = train[split,]
test_data = train[-split,]

#data with snp's
snp_train=train_data[,7:506]
snp_test=test_data[,7:506]
```

Using significant SNP's in Logistic Regression we got from t-test single snp approach

```
sig.train<-data.frame(train_data[,1:6],snp_train[,myresult1$positions])
fit1<-glm(sig.train$COPDStatus~., data=sig.train, family="binomial")
summary(fit1)

##
## Call:
## glm(formula = sig.train$COPDStatus ~ ., family = "binomial",
##     data = sig.train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7420  -0.2366  -0.1440  -0.0865   3.8440
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.116e+01  2.984e-01 -37.396  < 2e-16 ***
## Sex            -4.128e-04  5.179e-02  -0.008 0.993641
## BMI             2.550e-02  4.888e-03   5.216 1.83e-07 ***
## Age             9.792e-02  4.032e-03  24.284  < 2e-16 ***
## Smoking_Status  8.424e-01  3.707e-02  22.725  < 2e-16 ***
## FEV1Z           9.366e-01  2.366e-02  39.590  < 2e-16 ***
## SNP29           5.557e-02  3.602e-02   1.543 0.122860
## SNP62          -1.249e-01  4.711e-02  -2.652 0.008003 **
## SNP144          1.551e-01  5.644e-02   2.748 0.005998 **
## SNP164         -1.246e-01  3.627e-02  -3.435 0.000593 ***
## SNP210         -1.915e-01  8.668e-02  -2.210 0.027134 *
## SNP217         -1.304e-01  5.471e-02  -2.384 0.017104 *
## SNP220         -1.220e-01  4.635e-02  -2.633 0.008474 **
## SNP274         -3.568e-01  1.573e-01  -2.269 0.023265 *
## SNP320         -9.256e-02  3.850e-02  -2.404 0.016201 *
## SNP332         -6.764e-02  6.116e-02  -1.106 0.268726
## SNP333          4.972e-02  1.991e-01   0.250 0.802829
## SNP336         -1.729e-03  3.419e-01  -0.005 0.995965
## SNP337         -9.301e-02  2.591e-01  -0.359 0.719589
## SNP338         -5.951e-02  3.553e-01  -0.168 0.866967
## SNP339         -3.275e-01  5.744e-01  -0.570 0.568585
```

```
## SNP340          4.207e-01  4.718e-01   0.892 0.372624
## SNP341          8.614e-03  5.556e-01   0.016 0.987630
## SNP342         -8.466e-02  1.352e-01  -0.626 0.531062
## SNP393          1.383e-01  3.953e-02   3.497 0.000470 ***
## SNP458         -5.342e-01  1.665e-01  -3.209 0.001334 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15894  on 56075  degrees of freedom
## Residual deviance: 12272  on 56050  degrees of freedom
## AIC: 12324
##
## Number of Fisher Scoring iterations: 7

sig.test<-data.frame(test_data[,1:6], snp_test[,myresult1$positions])

pred<-predict(fit1, sig.test, type="response")
fit1.pred<-rep("0", dim(sig.test)[1])
fit1.pred[pred>0.5]="1"
fit1.pred=factor(fit1.pred)
table(fit1.pred, sig.test$COPDStatus)

##
## fit1.pred     0     1
##         0 54278  1614
##         1    63   120

test_accuracy=mean(fit1.pred==sig.test$COPDStatus)
test_accuracy

## [1] 0.9700936

##finding AUC for LR
library(pROC)
auc(sig.test$COPDStatus, pred)

## Area under the curve: 0.8513

plot(roc(sig.test$COPDStatus, pred, direction="<"), col="red")
```




```
most.significant.predictors<-which((coefficients(summary(fit1))[,4])<0.05)
most.significant.predictors
```

```
##    (Intercept)             BMI             Age Smoking_Status           FEV1Z
##              1               3               4               5               6
##          SNP62         SNP144         SNP164         SNP210         SNP217
##              8               9              10              11              12
##         SNP220         SNP274         SNP320         SNP393         SNP458
##             13              14              15              25              26
```

```
#require(GGally)
#ggscatmat(sig.train, color="COPDStatus")


#odds ratio for the predictors
or.sig.snp<-NULL
for(i in 2:26){
  or.sig.snp[i]<-exp(coefficients(summary(fit1))[,1])[i]
  }
or.sig.snp
```

```
##  [1]        NA 0.9995873 1.0258249 1.1028746 2.3219568 2.5511722 1.0571481
##  [8] 0.8825516 1.1677641 0.8828637 0.8256957 0.8777042 0.8851195 0.6998799
## [15] 0.9115986 0.9345940 1.0509804 0.9982723 0.9111885 0.9422230 0.7207443
## [22] 1.5229550 1.0086517 0.9188258 1.1482655 0.5861338
```

Using significant SNP's in Logistic Regression we got from Logistic Model single snp approach

```
sig.train.lr<-data.frame(train_data[,1:6],snp_train[,myresult2$positions])
fit2<-glm(sig.train.lr$COPDStatus~., data=sig.train.lr, family="binomial")
summary(fit2)

##
## Call:
## glm(formula = sig.train.lr$COPDStatus ~ ., family = "binomial",
##     data = sig.train.lr)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9240  -0.2357  -0.1426  -0.0856   3.8559
##
## Coefficients:
##                  Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -1.154e+01  3.976e-01 -29.021  < 2e-16 ***
## Sex            -1.220e-03  5.187e-02  -0.024 0.981238
## BMI             2.544e-02  4.894e-03   5.198 2.02e-07 ***
## Age             9.796e-02  4.039e-03  24.254  < 2e-16 ***
## Smoking_Status  8.416e-01  3.714e-02  22.661  < 2e-16 ***
## FEV1Z           9.367e-01  2.371e-02  39.514  < 2e-16 ***
## SNP28           1.045e-01  1.338e-01   0.781 0.434841
## SNP29           1.535e-01  1.333e-01   1.152 0.249466
## SNP56           1.312e-01  4.302e-02   3.049 0.002297 **
## SNP62          -1.266e-01  4.717e-02  -2.684 0.007267 **
## SNP106          8.150e-01  3.920e-01   2.079 0.037618 *
## SNP143         -3.845e-04  1.228e-01  -0.003 0.997503
## SNP144          1.538e-01  1.255e-01   1.225 0.220411
## SNP164         -1.233e-01  3.633e-02  -3.395 0.000686 ***
## SNP205          3.302e-01  1.072e-01   3.080 0.002069 **
## SNP217         -1.304e-01  5.481e-02  -2.379 0.017343 *
## SNP220         -1.208e-01  4.643e-02  -2.602 0.009260 **
## SNP222          1.521e-01  6.231e-02   2.442 0.014611 *
## SNP229          4.745e-01  1.646e-01   2.883 0.003940 **
## SNP269          3.003e-01  8.905e-02   3.372 0.000746 ***
## SNP320         -9.329e-02  3.852e-02  -2.422 0.015443 *
## SNP332         -6.271e-02  6.128e-02  -1.023 0.306120
## SNP333          5.041e-02  1.992e-01   0.253 0.800192
## SNP336         -2.820e-02  3.371e-01  -0.084 0.933313
## SNP337         -9.597e-02  2.572e-01  -0.373 0.709031
## SNP338         -7.579e-02  3.488e-01  -0.217 0.828002
## SNP339         -3.097e-01  5.611e-01  -0.552 0.580960
## SNP340          4.180e-01  4.734e-01   0.883 0.377266
## SNP341          4.578e-02  5.503e-01   0.083 0.933690
## SNP342         -9.664e-02  1.354e-01  -0.714 0.475266
## SNP393          1.368e-01  3.960e-02   3.454 0.000552 ***
## SNP442          4.456e-01  1.591e-01   2.800 0.005109 **
```

```
## SNP458         -5.374e-01  1.665e-01  -3.227 0.001252 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15894  on 56075  degrees of freedom
## Residual deviance: 12229  on 56043  degrees of freedom
## AIC: 12295
##
## Number of Fisher Scoring iterations: 7
```

```
sig.test.lr<-data.frame(test_data[,1:6], snp_test[,myresult2$positions])
pred2<-predict(fit2, sig.test.lr, type="response")
fit2.pred<-rep("0", dim(sig.test.lr)[1])
fit2.pred[pred2>0.5]="1"
fit2.pred=factor(fit2.pred)
table(fit2.pred, sig.test.lr$COPDStatus)
```

```
##
## fit2.pred     0     1
##         0 54272  1609
##         1    69   125
```

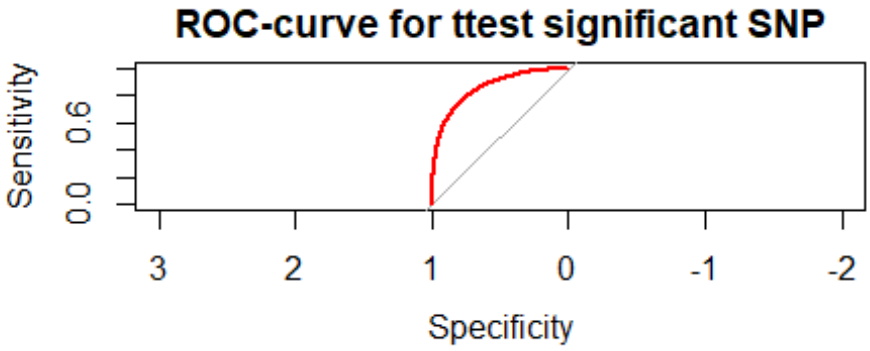
```
test_accuracy=mean(fit2.pred==sig.test.lr$COPDStatus)
test_accuracy
```

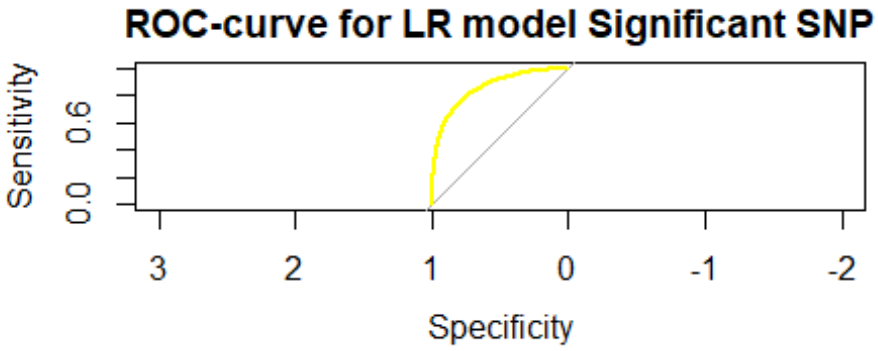
```
## [1] 0.9700758
```

```
significant<-which((coefficients(summary(fit2))[,4])<0.05)
auc(sig.test.lr$COPDStatus, pred2)
```

```
## Area under the curve: 0.8515
```

```
par(mfrow=c(2,1))
plot(roc(sig.test$COPDStatus, pred, direction="<"), col="red", main="ROC-curv
e for ttest significant SNP")
plot(roc(sig.test.lr$COPDStatus, pred2, direction="<"), col="yellow", main="R
OC-curve for LR model Significant SNP")
```






```
#odds ratio for the predictors with 27 snp's
or.sig.snp.lr<-NULL
for(i in 2:33){
  or.sig.snp.lr[i]<-exp(coefficients(summary(fit2))[,1])[i]
  }
or.sig.snp.lr
```

```
##  [1]        NA 0.9987808 1.0257658 1.1029163 2.3201796 2.5515258 1.1101248
##  [8] 1.1659065 1.1401631 0.8810742 2.2591916 0.9996156 1.1662503 0.8839621
## [15] 1.3912146 0.8777401 0.8861864 1.1643266 1.6072417 1.3502639 0.9109288
## [22] 0.9392132 1.0516975 0.9721904 0.9084907 0.9270097 0.7336419 1.5189179
## [29] 1.0468470 0.9078793 1.1465776 1.5613735 0.5842921
```

The OR's greater than 1 means, those SNP's are risk factors and others less than 1 me ans they are the protective ones for COPD development.

##LDA prediction test

library(MASS)

lda.train <- lda(sig.train$COPDStatus ~ ., data = sig.train)

predlda <- predict(lda.train, sig.test, type = "response")

```
levels(predlda$class) <- c("0", "1")
table(predlda$class, sig.test$COPDStatus) ##96.80% accuracy



qda.train <- qda(sig.train$COPDStatus ~ ., data = sig.train)
predqda <- predict(qda.train, sig.test, type = "response")
levels(predqda$class) <- c("0", "1")
table(predqda$class, sig.test$COPDStatus) ##94.43% accuracy
```

Appendix B – Logistic Visualization R Code

Visualization of logistic regression

Mohammad Mahmudul Huq

11/28/2021

```
##############################################
#VISUALIZATION Logistic regression

#Importing the dataset

load('train.rda')

trdata = mytrain

load('test.rda')

tstdata = mytest2

#data preprocessing for visualization



#Removing Sex, BMI, CancerStatus, AsthmaStatus and Genetic data

tr_C_b = trdata[-c(1,2,6,8, 9:509)]

#Removing Smoking_Status

tr_C_b_A_F = tr_C_b[-c(2)]

#Splitting into training and test sets

library(caTools)
set.seed(123)
split = sample.split(tr_C_b_A_F$COPDStatus, SplitRatio = 0.5)
training_A_F = subset(tr_C_b_A_F, split == TRUE)
test_A_F = subset(tr_C_b_A_F, split == FALSE)

#Fitting into logistic regression
classifier_A_F = glm(formula = COPDStatus~.,
                     family = binomial, data = training_A_F)

summary(classifier_A_F)
```

```
##
## Call:
## glm(formula = COPDStatus ~ ., family = binomial, data = training_A_F)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6090  -0.2473  -0.1583  -0.0988   3.9652
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -10.359408   0.251277  -41.23   <2e-16 ***
## Age           0.100206   0.004012   24.98   <2e-16 ***
## FEV1Z         1.035218   0.022664   45.68   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15682  on 56074  degrees of freedom
## Residual deviance: 12667  on 56072  degrees of freedom
## AIC: 12673
##
## Number of Fisher Scoring iterations: 7
```

```
#Predicting the test set results

prob_pred = predict(classifier_A_F, type = 'response', newdata = test_A_
F[-3])

y_pred = ifelse(prob_pred>0.5,1,0)


#Building confusion matrix

cm_A_F = table(test_A_F[,3], y_pred)

cm_A_F
```

```
##    y_pred
##         0     1
##   0 54281    30
##   1  1705    60
```

```
# Visualising the Test set results

#Downloading ElemStatLearn

packageurl <- "https://cran.r-project.org/src/contrib/Archive/ElemStatLe
arn/ElemStatLearn_2015.6.26.tar.gz"
install.packages(packageurl, repos=NULL, type="source")
```

```
## Installing package into 'C:/Users/Rubiyet/Documents/R/win-library/4.
1'
## (as 'lib' is unspecified)
```

```
library(ElemStatLearn)
set = test_A_F
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Age', 'FEV1Z')
prob_set = predict(classifier_A_F, type = 'response', newdata = grid_se
t)
y_grid = ifelse(prob_set > 0.5, 1, 0)
plot(set[, -3],
     main = 'Logistic Regression (Test set)',
     xlab = 'Age', ylab = 'FEV1Z',
     xlim = range(X1), ylim = range(X2))
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add
 = TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'tomato', 'springg
reen3'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'red3', 'green4'))
```

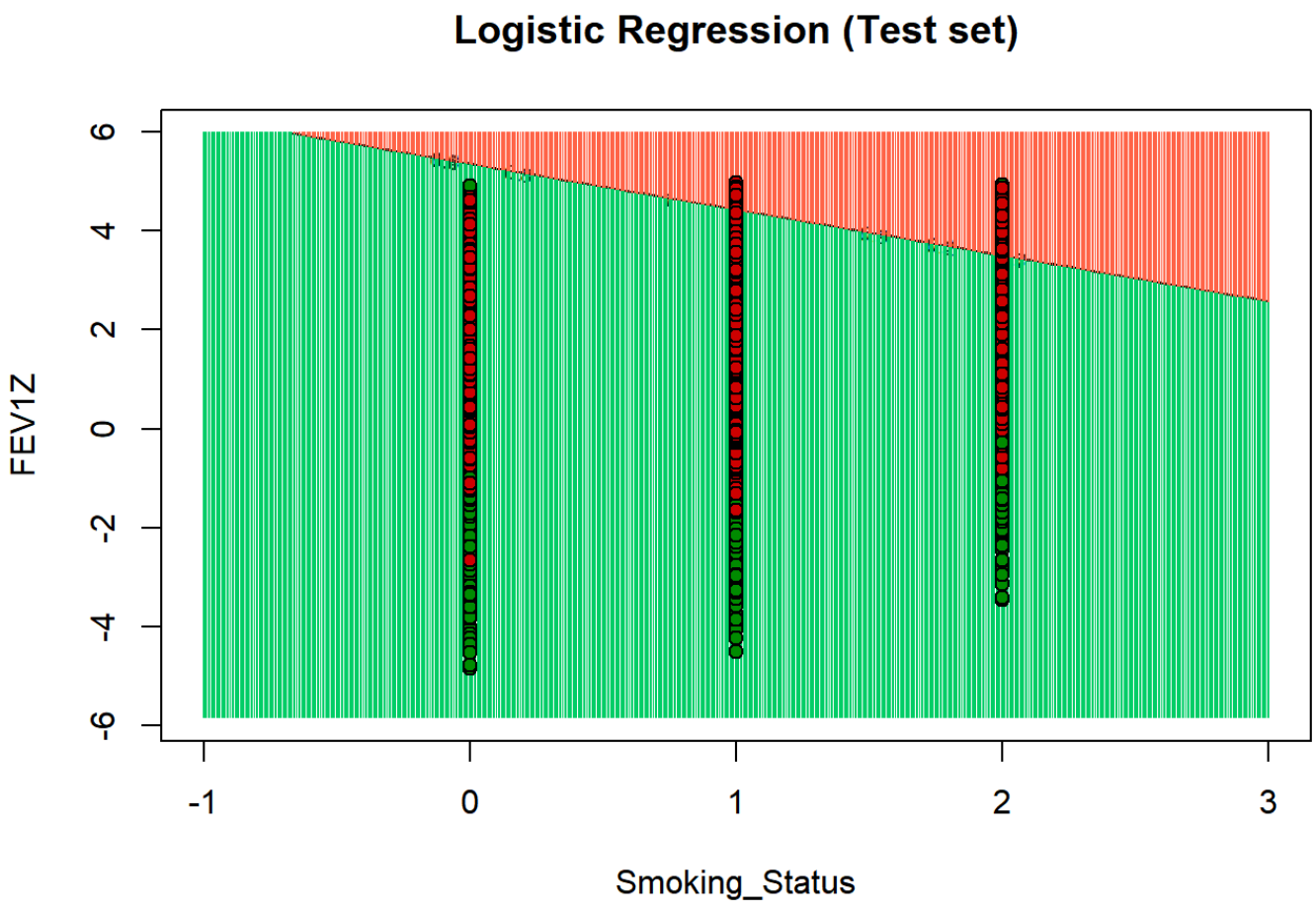

```
#Removing Age

tr_C_b_S_F = tr_C_b[-c(1)]

#Splitting into training and test sets

library(caTools)
set.seed(123)
split = sample.split(tr_C_b_S_F$COPDStatus, SplitRatio = 0.5)
training_S_F = subset(tr_C_b_S_F, split == TRUE)
test_S_F = subset(tr_C_b_S_F, split == FALSE)

#Fitting into logistic regression
classifier_S_F = glm(formula = COPDStatus~.,
                     family = binomial, data = training_S_F)

summary(classifier_S_F)
```

```
##
## Call:
## glm(formula = COPDStatus ~ ., family = binomial, data = training_S_F)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7919  -0.2428  -0.1628  -0.1117   4.0865
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)    -4.99827    0.05218  -95.79   <2e-16 ***
## Smoking_Status  0.86373    0.03468   24.90   <2e-16 ***
## FEV1Z           0.93396    0.02263   41.26   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 15682  on 56074  degrees of freedom
## Residual deviance: 12816  on 56072  degrees of freedom
## AIC: 12822
##
## Number of Fisher Scoring iterations: 7
```

```
#Predicting the test set results

prob_pred = predict(classifier_S_F, type = 'response', newdata = test_S_
F[-3])

y_pred = ifelse(prob_pred>0.5,1,0)


#Building confusion matrix

cm_S_F = table(test_S_F[,3], y_pred)

cm_S_F
```

```
##    y_pred
##         0     1
##   0 54282    29
##   1  1715    50
```

```
# Visualising the Test set results

library(ElemStatLearn)
set = test_S_F
X1 = seq(min(set[, 1]) - 1, max(set[, 1]) + 1, by = 0.01)
X2 = seq(min(set[, 2]) - 1, max(set[, 2]) + 1, by = 0.01)
grid_set = expand.grid(X1, X2)
colnames(grid_set) = c('Smoking_Status', 'FEV1Z')
prob_set = predict(classifier_S_F, type = 'response', newdata = grid_se
t)
y_grid = ifelse(prob_set > 0.5, 1, 0)
plot(set[, -3],
     main = 'Logistic Regression (Test set)',
     xlab = 'Smoking_Status', ylab = 'FEV1Z',
     xlim = range(X1), ylim = range(X2))
contour(X1, X2, matrix(as.numeric(y_grid), length(X1), length(X2)), add
 = TRUE)
points(grid_set, pch = '.', col = ifelse(y_grid == 1, 'tomato', 'springg
reen3'))
points(set, pch = 21, bg = ifelse(set[, 3] == 1, 'red3', 'green4'))
```



Appendix C – Random Forest R Code

STAT 846 Random Forest

Zoe Parker Cates

28/11/2021

First, we have to remove any NA entries as well as the columns for Asthma Status, Cancer Status, and patient ID.

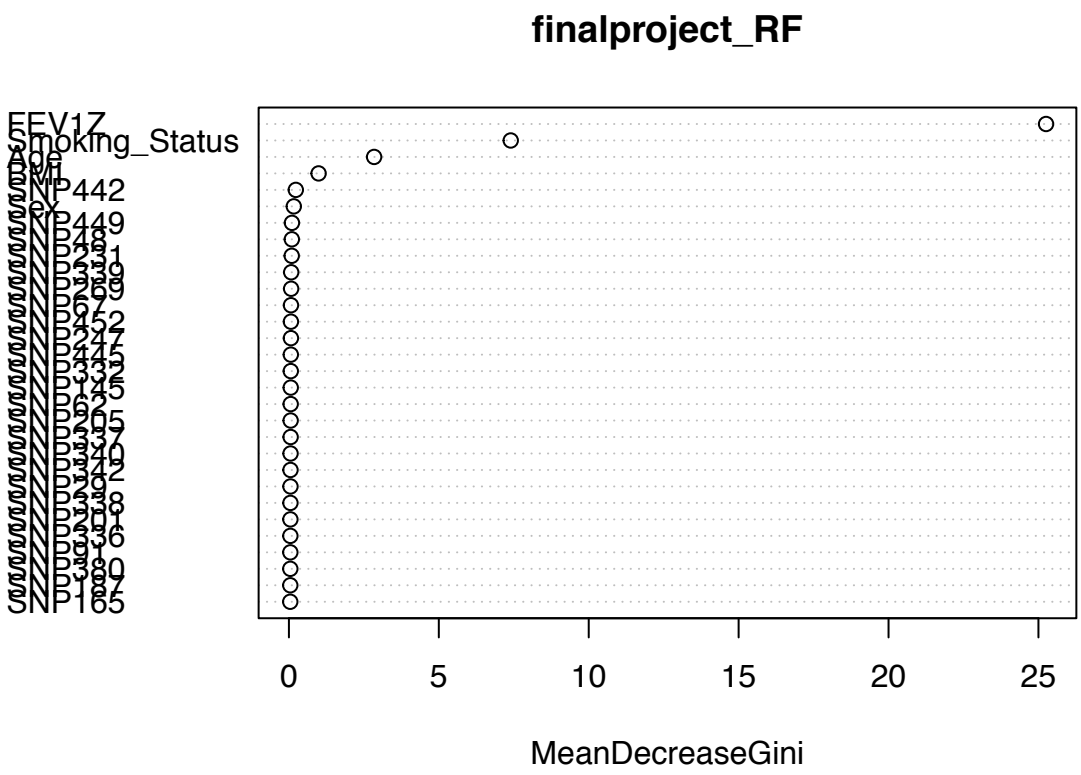
```
# read in data and clean
train1 <- get(load("train.rda"))
test <- get(load("test.rda"))

na.omit(train1)
na.omit(test)

train1 <- train1[, names(train1) != "patid"]
train1 <- train1[, names(train1) != "AsthmaStatus"]
train1 <- train1[, names(train1) != "CancerStatus"]
```

Next we will separate the provided training data (112151 samples) into a training subset and a testing subset.

Then we can build the random forest model.

```
# separate into testing and training
set.seed(2021)
training_data <- sample(1:nrow(train1), nrow(train1)/2)
testing_data <- train1[-training_data, ]


# build tree
set.seed(2021)
n_tree = 100
finalproject_RF <- randomForest(factor(COPDStatus) ~ ., data=train1,
                                subset=training_data, maxnodes=4,
                                mtry=30, ntree=n_tree)


# test tree
finalproject_predict <- predict(finalproject_RF, newdata = testing_data)
```

```r
# plot results
varImpPlot(finalproject_RF)
```




```r
# generate confusion matrix
testPRED = as.factor(finalproject_predict)
testCOPD = as.factor(testing_data$COPDStatus)
confusionMatrix(testPRED, testCOPD)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 54342  1734
##          1     0     0
##
##                Accuracy : 0.9691
##                  95% CI : (0.9676, 0.9705)
##     No Information Rate : 0.9691
##     P-Value [Acc > NIR] : 0.5064
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
```

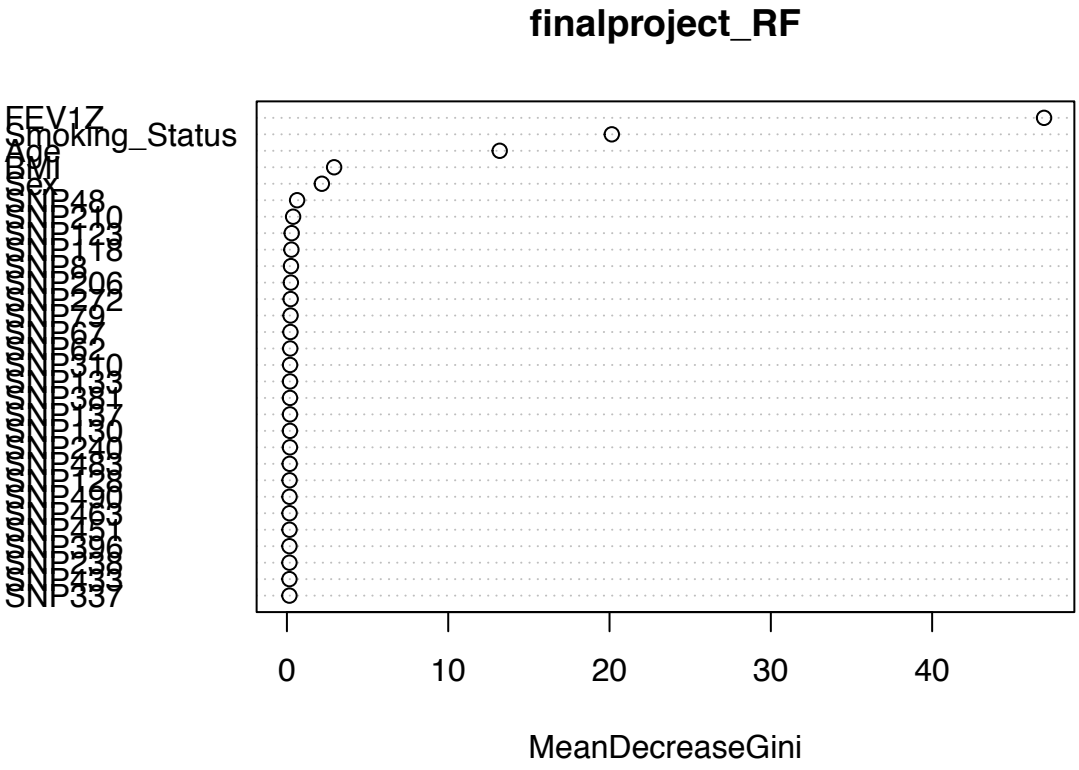
```
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.9691
##          Neg Pred Value :    NaN
##              Prevalence : 0.9691
##          Detection Rate : 0.9691
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

Discussion

This is a problem. The random forest model appears to be predicting only one class (0, meaning negative for COPD) all the time. This may be because we have 54342 cases of 0 and only 1734 cases of 1. To combat this problem, we will try undersampling and training a new model.

```
# separate into testing and training
set.seed(2021)


# undersample the negative data
split_data <- split(train1, train1$COPDStatus)
training_data_undersample <- sample(1:nrow(split_data$"0"), 3530)
new_train_neg <- split_data$"0"[training_data_undersample, ]

# training data
training_data_negative <- sample(1:nrow(new_train_neg), 1765)
training_data_positive <- sample(1:nrow(split_data$"1"), 1765)

# testing data
testing_data_negative <- new_train_neg[-training_data_negative, ]
testing_data_positive <- split_data$"1"[-training_data_positive, ]

# combine negative and positive data
training_combo <- rbind(new_train_neg[training_data_negative, ],
                        split_data$"1"[training_data_positive, ])
testing_combo <- rbind(testing_data_negative, testing_data_positive)


# build tree
set.seed(2021)
n_tree = 100
finalproject_RF <- randomForest(factor(COPDStatus) ~ ., data=training_combo,
                                maxnodes=4, mtry=30, ntree=n_tree)


# test tree
finalproject_predict <- predict(finalproject_RF, newdata = testing_combo)


# plot results
```

```r
varImpPlot(finalproject_RF)
```




```r
# generate confusion matrix
testPRED = as.factor(finalproject_predict)
testCOPD = as.factor(testing_combo$COPDStatus)
confusionMatrix(testPRED, testCOPD)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1250  482
##          1  515 1283
##
##                Accuracy : 0.7176
##                  95% CI : (0.7024, 0.7324)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.4351
##
##  Mcnemar's Test P-Value : 0.3108
##
##             Sensitivity : 0.7082
##             Specificity : 0.7269
```

```
##          Pos Pred Value : 0.7217
##          Neg Pred Value : 0.7136
##              Prevalence : 0.5000
##          Detection Rate : 0.3541
##    Detection Prevalence : 0.4907
##       Balanced Accuracy : 0.7176
##
##        'Positive' Class : 0
##
```

Because we are now working with a data set of reduced size, we can optimize the tree for better performance. Here we increase ntree and maxnodes.

```
# build a larger tree
set.seed(2021)
n_tree = 1000
finalproject_RF <- randomForest(factor(COPDStatus) ~ ., data=training_combo,
                                maxnodes=15, mtry=30, ntree=n_tree)


# test larger tree
finalproject_predict <- predict(finalproject_RF, newdata = testing_combo)


# plot results
varImpPlot(finalproject_RF)
```


```
# create confusion matrix
# I found an answer to an error I was having with formatting the following
# 2 lines: https://stackoverflow.com/a/65942203
testPRED = as.factor(finalproject_predict)
testCOPD = as.factor(testing_combo$COPDStatus)
confusionMatrix(testPRED, testCOPD)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1331  460
##          1  434 1305
##
##                Accuracy : 0.7467
##                  95% CI : (0.7321, 0.761)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.4935
##
##  Mcnemar's Test P-Value : 0.4031
##
##             Sensitivity : 0.7541
##             Specificity : 0.7394
##          Pos Pred Value : 0.7432
```

```
##          Neg Pred Value : 0.7504
##              Prevalence : 0.5000
##          Detection Rate : 0.3771
##    Detection Prevalence : 0.5074
##       Balanced Accuracy : 0.7467
##
##        'Positive' Class : 0
##
```

##Results

The final model has an accuracy of ~75%. This is better than the first model. In this case undersampling was necessary to increase the prediction accuracy for positive samples. However, the trade-off is a decrease in overall accuracy.

This code produced with assistance from professor-provided lab for random forest and homework 4 solution.

Appendix D – Classification Tree R Code

A Discussion of Statistical and Machine Learning Methods for determination of the most significant risk factors for developing COPD

Zoe Parker Cates (11182963)

Yaindrila Barua (11318333)

Leila Rabiei Fard (11301719)

Isaac Dante Asamoah (11319281)

Mohammad Mahmudul Huq (11243856)

28/11/2021

```
library(tree);library(ggthemes);
library("ggplot2");library(caret);library(rpart.plot);library(dplyr)

## Loading required package: lattice

## Loading required package: rpart

##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union

library(rpart);library(tidyverse);library(knitr);library(kableExtra)

## Registered S3 method overwritten by 'cli':
##   method     from
##   print.tree tree

## -- Attaching packages --------------------------------------- tidyverse
1.3.1 --

## v tibble  3.1.4     v purrr   0.3.4
## v tidyr   1.1.4     v stringr 1.4.0
## v readr   2.0.2     v forcats 0.5.1

## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
```

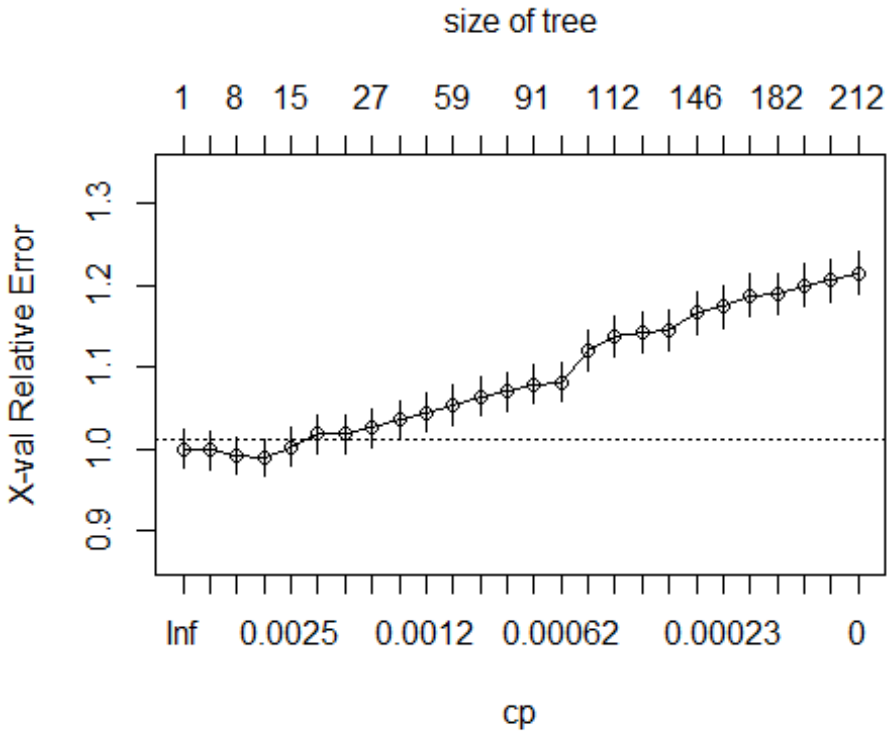
```
##
## Attaching package: 'kableExtra'

## The following object is masked from 'package:dplyr':
##
##     group_rows

load("C:/Users/NEW USER/Desktop/Acturial/Data/test.rda")
load("C:/Users/NEW USER/Desktop/Acturial/Data/train.rda")

mytrain$COPDStatus = as.factor(ifelse(mytrain$COPDStatus <= 0, "Neg", "Pos"))

mytrain <- mytrain[, names(mytrain) != "patid"]
mytrain <- mytrain[, names(mytrain) != "AsthmaStatus"]
mytrain <- mytrain[, names(mytrain) != "CancerStatus"]

#Split data
set.seed(2021)
train= sample(1:nrow(mytrain), nrow(mytrain) / 2)
train.data=mytrain[train,]
test.data=mytrain[-train,]

#str(mytrain)

rpart_model <- rpart(COPDStatus ~ ., data = train.data, method = 'class',
                     control = rpart.control(cp = 0))

summary(rpart_model, cp=1)

## Call:
## rpart(formula = COPDStatus ~ ., data = train.data, method = "class",
##     control = rpart.control(cp = 0))
##   n= 56075
##
##              CP nsplit rel error    xerror       xstd
## 1  5.707127e-03      0 1.0000000 1.0000000 0.02321550
## 2  3.619154e-03      4 0.9771715 0.9988864 0.02320300
## 3  3.062361e-03      7 0.9660356 0.9916481 0.02312155
## 4  2.783964e-03      9 0.9599109 0.9894209 0.02309642
## 5  2.227171e-03     14 0.9459911 1.0022272 0.02324049
## 6  1.948775e-03     15 0.9437639 1.0178174 0.02341451
## 7  1.855976e-03     22 0.9270601 1.0189310 0.02342688
## 8  1.670379e-03     26 0.9187082 1.0261693 0.02350712
## 9  1.299183e-03     40 0.8919822 1.0356347 0.02361159
## 10 1.113586e-03     44 0.8853007 1.0439866 0.02370333
## 11 8.908686e-04     58 0.8674833 1.0534521 0.02380681
## 12 8.351893e-04     63 0.8630290 1.0645880 0.02392789
## 13 8.042564e-04     66 0.8602450 1.0695991 0.02398214
## 14 6.959911e-04     90 0.8324053 1.0790646 0.02408424
## 15 5.567929e-04     94 0.8296214 1.0818486 0.02411418
```

```
## 16 4.175947e-04    107 0.8223831 1.1202673 0.02452297
## 17 3.977092e-04    111 0.8207127 1.1375278 0.02470408
## 18 3.479955e-04    118 0.8179287 1.1414254 0.02474477
## 19 2.783964e-04    126 0.8151448 1.1453229 0.02478537
## 20 2.386255e-04    145 0.8084633 1.1659243 0.02499872
## 21 2.227171e-04    152 0.8067929 1.1731626 0.02507318
## 22 1.855976e-04    157 0.8056793 1.1876392 0.02522133
## 23 1.391982e-04    181 0.8012249 1.1893096 0.02523836
## 24 1.113586e-04    185 0.8006682 1.1998886 0.02534589
## 25 9.279881e-05    205 0.7984410 1.2054566 0.02540228
## 26 0.000000e+00    211 0.7978842 1.2143653 0.02549219
##
## Variable importance
##          FEV1Z            Age Smoking_Status            BMI         SNP387
##             17              7              6              3              1
##         SNP177          SNP67         SNP331         SNP330         SNP176
##              1              1              1              1              1
##          SNP12          SNP80         SNP326          SNP81         SNP130
##              1              1              1              1              1
##          SNP95         SNP245         SNP325         SNP296         SNP262
##              1              1              1              1              1
##         SNP323         SNP276         SNP398         SNP306         SNP498
##              1              1              1              1              1
##         SNP499         SNP500         SNP345
##              1              1              1
##
## Node number 1: 56075 observations
##   predicted class=Neg  expected loss=0.03202853  P(node) =1
##     class counts: 54279  1796
##    probabilities: 0.968 0.032
```

```
kk=printcp(rpart_model)%>%kable
```

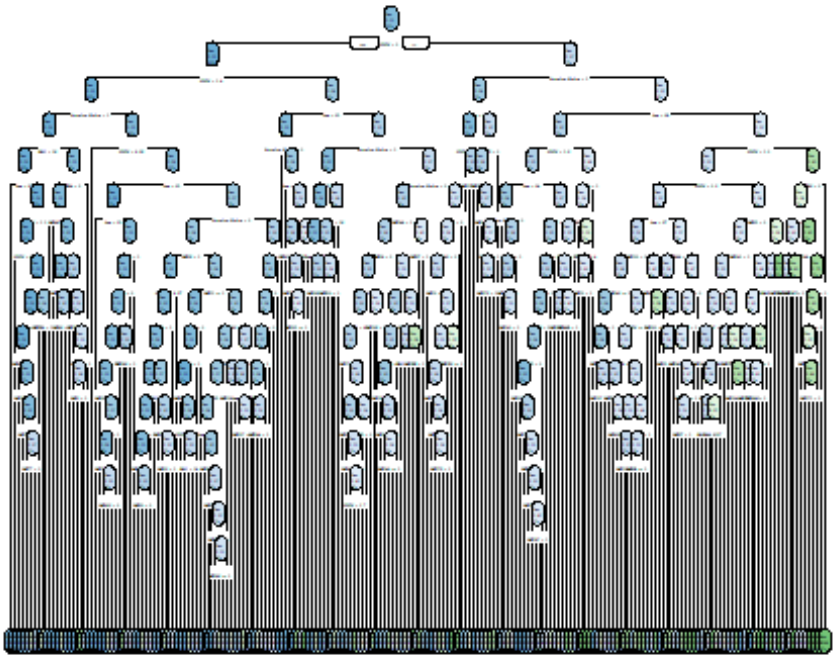
```
##
## Classification tree:
## rpart(formula = COPDStatus ~ ., data = train.data, method = "class",
##     control = rpart.control(cp = 0))
##
## Variables actually used in tree construction:
##   [1] Age            BMI            FEV1Z          Sex
Smoking_Status
##   [6] SNP104         SNP107         SNP11          SNP114         SNP116
##  [11] SNP12          SNP120         SNP13          SNP130         SNP134
##  [16] SNP136         SNP137         SNP138         SNP143         SNP149
##  [21] SNP17          SNP170         SNP171         SNP176         SNP177
##  [26] SNP18          SNP180         SNP184         SNP187         SNP19
##  [31] SNP194         SNP195         SNP196         SNP197         SNP206
##  [36] SNP211         SNP222         SNP225         SNP227         SNP229
##  [41] SNP230         SNP233         SNP237         SNP24          SNP241
##  [46] SNP244         SNP245         SNP25          SNP257         SNP26
```

```
##  [51] SNP261         SNP262         SNP263         SNP264         SNP265
##  [56] SNP266         SNP268         SNP269         SNP276         SNP281
##  [61] SNP285         SNP29          SNP294         SNP296         SNP3
##  [66] SNP306         SNP31          SNP312         SNP320         SNP322
##  [71] SNP326         SNP327         SNP331         SNP334         SNP337
##  [76] SNP345         SNP348         SNP349         SNP350         SNP351
##  [81] SNP353         SNP356         SNP359         SNP36          SNP360
##  [86] SNP370         SNP374         SNP384         SNP387         SNP39
##  [91] SNP395         SNP397         SNP401         SNP408         SNP409
##  [96] SNP412         SNP424         SNP425         SNP44          SNP447
## [101] SNP449         SNP451         SNP453         SNP455         SNP46
## [106] SNP48          SNP498         SNP5           SNP53          SNP56
## [111] SNP61          SNP65          SNP67          SNP69          SNP76
## [116] SNP77          SNP80          SNP81          SNP83          SNP86
## [121] SNP94          SNP95          SNP96          SNP97
##
## Root node error: 1796/56075 = 0.032029
##
## n= 56075
##
##            CP nsplit rel error  xerror     xstd
## 1  5.7071e-03      0   1.00000 1.00000 0.023216
## 2  3.6192e-03      4   0.97717 0.99889 0.023203
## 3  3.0624e-03      7   0.96604 0.99165 0.023122
## 4  2.7840e-03      9   0.95991 0.98942 0.023096
## 5  2.2272e-03     14   0.94599 1.00223 0.023240
## 6  1.9488e-03     15   0.94376 1.01782 0.023415
## 7  1.8560e-03     22   0.92706 1.01893 0.023427
## 8  1.6704e-03     26   0.91871 1.02617 0.023507
## 9  1.2992e-03     40   0.89198 1.03563 0.023612
## 10 1.1136e-03     44   0.88530 1.04399 0.023703
## 11 8.9087e-04     58   0.86748 1.05345 0.023807
## 12 8.3519e-04     63   0.86303 1.06459 0.023928
## 13 8.0426e-04     66   0.86024 1.06960 0.023982
## 14 6.9599e-04     90   0.83241 1.07906 0.024084
## 15 5.5679e-04     94   0.82962 1.08185 0.024114
## 16 4.1759e-04    107   0.82238 1.12027 0.024523
## 17 3.9771e-04    111   0.82071 1.13753 0.024704
## 18 3.4800e-04    118   0.81793 1.14143 0.024745
## 19 2.7840e-04    126   0.81514 1.14532 0.024785
## 20 2.3863e-04    145   0.80846 1.16592 0.024999
## 21 2.2272e-04    152   0.80679 1.17316 0.025073
## 22 1.8560e-04    157   0.80568 1.18764 0.025221
## 23 1.3920e-04    181   0.80122 1.18931 0.025238
## 24 1.1136e-04    185   0.80067 1.19989 0.025346
## 25 9.2799e-05    205   0.79844 1.20546 0.025402
## 26 0.0000e+00    211   0.79788 1.21437 0.025492

plotcp(rpart_model)
```



The summary shows us that the variable FEV1Z is by far the most important for determining the COPD status.

```
##rpart_model
rpart.plot(rpart_model)

## Warning: labs do not fit even at cex 0.15, there may be some overplotting
```



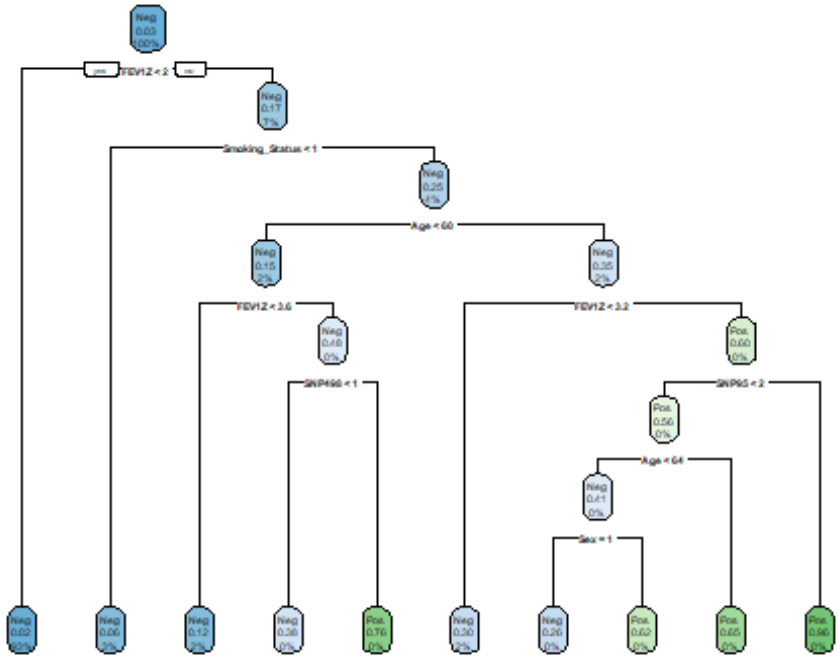
```
pred <- predict(rpart_model, newdata=test.data,type="class")

tab=table(pred,test.data$COPDStatus)
tab

##
## pred    Neg   Pos
##   Neg 53634  1476
##   Pos   708   258

100-(sum(diag(tab))/sum(tab))*100

## [1] 3.894714

confusionMatrix(tab)

## Confusion Matrix and Statistics
##
##
## pred    Neg   Pos
##   Neg 53634  1476
##   Pos   708   258
##
##                Accuracy : 0.9611
##                  95% CI : (0.9594, 0.9626)
##     No Information Rate : 0.9691
##     P-Value [Acc > NIR] : 1
##
```

```
##                   Kappa : 0.1728
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9870
##             Specificity : 0.1488
##          Pos Pred Value : 0.9732
##          Neg Pred Value : 0.2671
##              Prevalence : 0.9691
##          Detection Rate : 0.9565
##    Detection Prevalence : 0.9828
##       Balanced Accuracy : 0.5679
##
##        'Positive' Class : Neg
##
```

```
min_cp = rpart_model$cptable[which.min(rpart_model$cptable[,"xerror"]),"CP"]
min_cp
```

```
## [1] 0.002783964
```

```
rpart_prune = prune(rpart_model, cp = min_cp)

rpart.plot(rpart_prune)
```



```
pred <- predict(rpart_prune, newdata=test.data,type="class")
```

```
tab=table(pred,test.data$COPDStatus)
tab
```

```
##
## pred    Neg   Pos
##   Neg 54252  1618
##   Pos    90   116
```

```
100-(sum(diag(tab))/sum(tab))*100
```

```
## [1] 3.045866
```

```
confusionMatrix(tab)
```

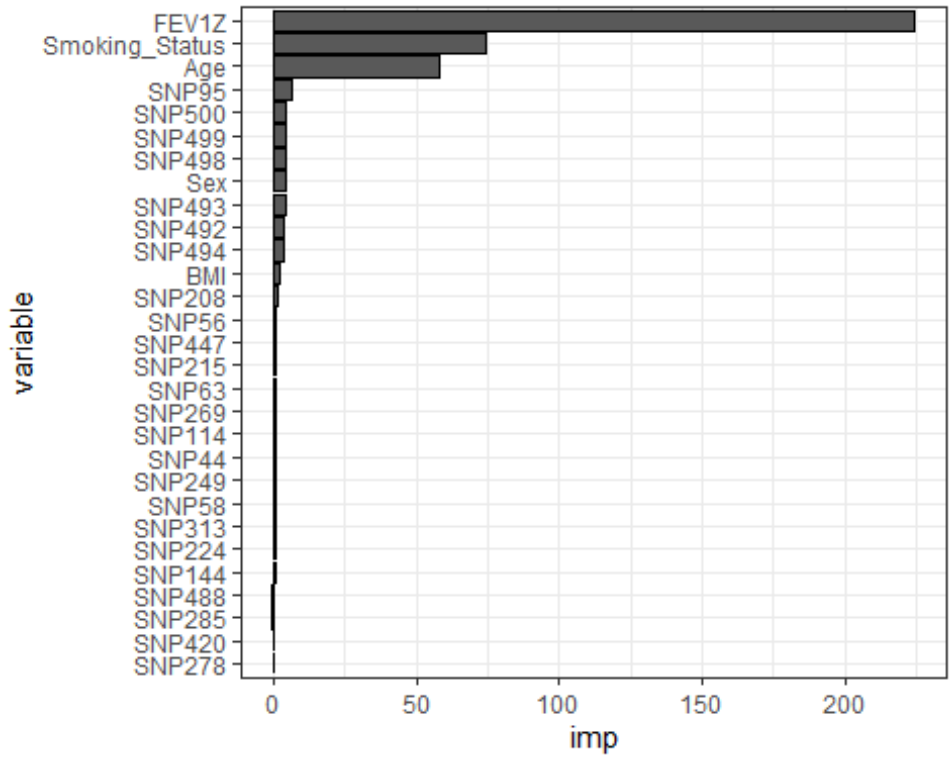
```
## Confusion Matrix and Statistics
##
##
## pred    Neg   Pos
##   Neg 54252  1618
##   Pos    90   116
##
##                Accuracy : 0.9695
##                  95% CI : (0.9681, 0.9709)
##     No Information Rate : 0.9691
##     P-Value [Acc > NIR] : 0.2677
##
##                   Kappa : 0.1138
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9983
##             Specificity : 0.0669
##          Pos Pred Value : 0.9710
##          Neg Pred Value : 0.5631
##              Prevalence : 0.9691
##          Detection Rate : 0.9675
##    Detection Prevalence : 0.9963
##       Balanced Accuracy : 0.5326
##
##        'Positive' Class : Neg
##
```

```
implot <- as.data.frame(rpart_prune$variable.importance)
implot
```

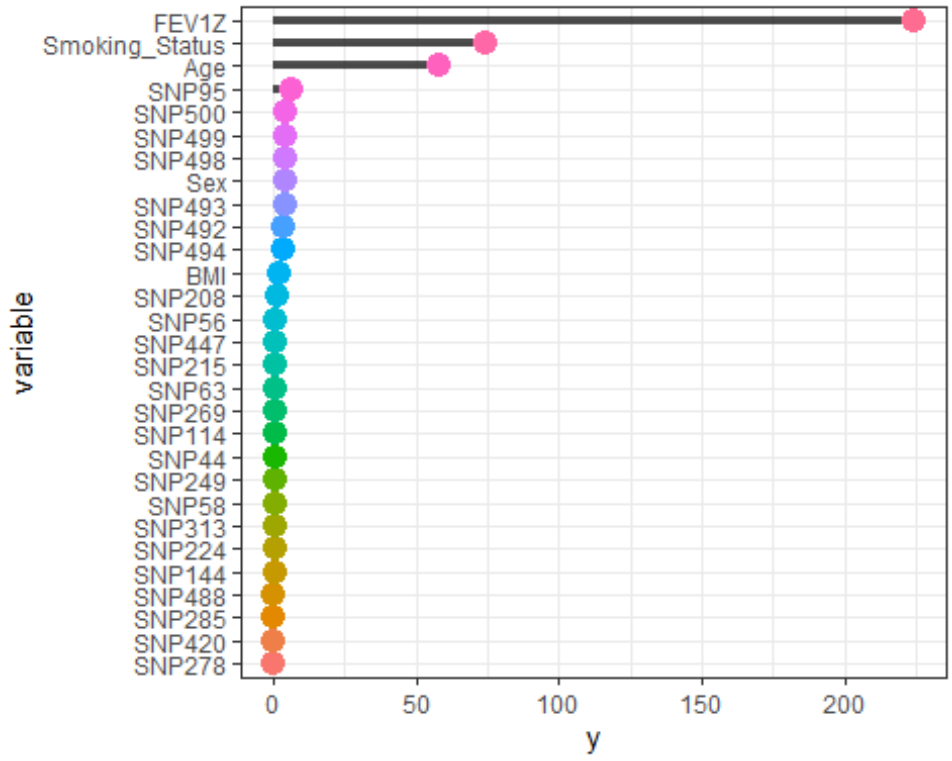
```
##                rpart_prune$variable.importance
## FEV1Z                               224.5602457
## Smoking_Status                       74.2785094
## Age                                  58.1168961
## SNP95                                 6.5576735
## SNP498                                4.6276537
```

```
## SNP499                                  4.6276537
## SNP500                                  4.6276537
## Sex                                     4.4165915
## SNP493                                  4.4072892
## SNP492                                  3.7461959
## SNP494                                  3.5258314
## BMI                                     2.3803753
## SNP208                                  1.2082147
## SNP56                                   1.0660738
## SNP215                                  0.9493115
## SNP447                                  0.9493115
## SNP114                                  0.9137776
## SNP269                                  0.9137776
## SNP63                                   0.9137776
## SNP249                                  0.5002972
## SNP44                                   0.5002972
## SNP58                                   0.4954106
## SNP313                                  0.4541264
## SNP144                                  0.4288261
## SNP224                                  0.4288261
## SNP488                                  0.3715579
## SNP285                                  0.3573551
## SNP420                                  0.3035523
## SNP278                                  0.1517761
```

```
df <- data.frame(imp = rpart_prune$variable.importance)
df2 <- df %>%
  tibble::rownames_to_column() %>%
  dplyr::rename("variable" = rowname) %>%
  dplyr::arrange(imp) %>%
  dplyr::mutate(variable = forcats::fct_inorder(variable))
ggplot2::ggplot(df2) +
  geom_col(aes(x = variable, y = imp),
           col = "black", show.legend = F) +
  coord_flip() +
  scale_fill_grey() +
  theme_bw()
```




```
ggplot2::ggplot(df2) +
  geom_segment(aes(x = variable, y = 0, xend = variable, yend = imp),
               size = 1.5, alpha = 0.7) +
  geom_point(aes(x = variable, y = imp, col = variable),
             size = 4, show.legend = F) +
  coord_flip() +
  theme_bw()
```




```
plotcp(rpart_prune)
```