Predictive Modeling of COVID-19

Aziz Asomiddinov, Tom Youngblood

August 21, 2024

Introduction

Abstract

This R-Markdown file contains an exploratory data analysis of Kaggle's COVID-19 Dataset, and the process of fitting two machine learning models, LDA and QDA, to determine the variables associated with death by COVID-19.

There are five sections:

1. Data Wrangling
2. Exploratory Data Analysis
3. Variable Selection
4. Predictive Modeling
5. Conclusions

Data

The dataset, Kaggle's 'COVID-19 Dataset', can be found at this link: https://www.kaggle.com/datasets/meirnizri/covid19-dataset.

The content of the dataset contains 21 features and 1,048,576 unique patients. The features are listed below (source: Kaggle user meirnizri):

- sex: 1 for female and 2 for male.
- age: of the patient.
- classification: covid test findings. Values 1-3 mean that the patient was diagnosed with covid in different.
- degrees. 4 or higher means that the patient is not a carrier of covid or that the test is inconclusive.
- patient type: type of care the patient received in the unit. 1 for returned home and 2 for hospitalization.
- pneumonia: whether the patient already have air sacs inflammation or not.
- pregnancy: whether the patient is pregnant or not.
- diabetes: whether the patient has diabetes or not.
- copd: Indicates whether the patient has Chronic obstructive pulmonary disease or not.
- asthma: whether the patient has asthma or not.
- inmsupr: whether the patient is immunosuppressed or not.
- hypertension: whether the patient has hypertension or not.

- cardiovascular: whether the patient has heart or blood vessels related disease.
- renal chronic: whether the patient has chronic renal disease or not.
- other disease: whether the patient has other disease or not.
- obesity: whether the patient is obese or not.
- tobacco: whether the patient is a tobacco user.
- usmr: Indicates whether the patient treated medical units of the first, second or third level.
- medical unit: type of institution of the National Health System that provided the care.
- intubed: whether the patient was connected to the ventilator.
- icu: Indicates whether the patient had been admitted to an Intensive Care Unit.
- date died: If the patient died indicate the date of death, and 9999-99-99 otherwise.

Data Wrangling

The data is loaded and cleaned in this section.

Loading Data

Libraries must be loaded first.

```r
library(ggplot2)
library(dplyr)
```

```
##
## 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
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1
## v readr     2.1.5

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become error
```

```r
library(tidyr)
library(MASS)
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:dplyr':
```

```
##
##     select
```

```r
library(ggpubr)
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
library(car)
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:purrr':
##
##     some
##
## The following object is masked from 'package:dplyr':
##
##     recode
```

```r
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(leaps)
```

The data is loaded in the cell below.

```r
data <- read_csv('Covid Data.csv')
```

```
## Rows: 1048575 Columns: 21
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): DATE_DIED
## dbl (20): USMER, MEDICAL_UNIT, SEX, PATIENT_TYPE, INTUBED, PNEUMONIA, AGE, P...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(data)
```

```
## # A tibble: 6 x 21
##   USMER MEDICAL_UNIT   SEX PATIENT_TYPE DATE_DIED  INTUBED PNEUMONIA   AGE
##   <dbl>        <dbl> <dbl>        <dbl> <chr>        <dbl>     <dbl> <dbl>
## 1     2             1     1             1 03/05/2020      97          1    65
## 2     2             1     2             1 03/06/2020      97          1    72
## 3     2             1     2             2 09/06/2020       1          2    55
## 4     2             1     1             1 12/06/2020      97          2    53
## 5     2             1     2             1 21/06/2020      97          2    68
## 6     2             1     1             2 9999-99-99       2          1    40
## # i 13 more variables: PREGNANT <dbl>, DIABETES <dbl>, COPD <dbl>,
## #   ASTHMA <dbl>, INMSUPR <dbl>, HIPERTENSION <dbl>, OTHER_DISEASE <dbl>,
## #   CARDIOVASCULAR <dbl>, OBESITY <dbl>, RENAL_CHRONIC <dbl>, TOBACCO <dbl>,
## #   CLASIFFICATION_FINAL <dbl>, ICU <dbl>
```

Data Cleaning: Binary Response Variable Creation

As seen in the table above, there is no binary response variable for the intended outcome variable, death. In the code below, the binary variable, 'DEATH', is created.

```
# If DATE_DIED not 9999-99-99, patient did not die
data$DIED <- ifelse(data$DATE_DIED != '9999-99-99', 1, 0)
data$DIED <- as.factor(data$DIED)
```

Data Cleaning: Removal of Missing Values

Values of 97, 98, and 99 indicate missing data; these values are immediately visible in the table above. Missing data values are removed in the cell below.

```
# Drop date died
data <- data[,!names(data) %in% c("DATE_DIED", "USMER")]

# Rows before missing value removal
original_nrows <- nrow(data)
cat('Number of rows before missing data removal: ', original_nrows, '\n')
```

```
## Number of rows before missing data removal:  1048575
```

```
# Missing Value Removal
data <- data <- data[!apply(data, 1, function(row) any(row %in% c(97, 98, 99))), ]

# Drop variables that are problematic later in the analysis
data <- data[,!names(data) %in% c("SEX", "PATIENT_TYPE", "CLASSIFICATION_FINAL")]

# Rows after missing value removal
new_nrows <- nrow(data)
cat('Number of rows after data removal: ', new_nrows, '\n')
```

```
## Number of rows after data removal:  76832
```

```
# Rows removed
removed <- original_nrows - new_nrows
cat('Number of observations removed: ', removed, '\n')
```

```
## Number of observations removed:  971743
```

Data Cleaning: Datatype Conversion

Many of the categorical variables in the data-set are classified as type . The LDA and QDA algorithms from the MASS library may treat these variables as ordinal, which they are not. The code-cell below converts the necessary variables to .

All variables besides those listed below were converted to factor:

- age: of the patient.
- date died: If the patient died indicate the date of death, and 9999-99-99 otherwise.
- usmr: Indicates whether the patient treated medical units of the first, second or third level.
    - usmr was not converted to factor as its data is not useful in factor form, according to R output.

```r
# Convert necessary variables to factor
data$PNEUMONIA <- as.factor(data$PNEUMONIA)
data$PREGNANT <- as.factor(data$PREGNANT)
data$DIABETES <- as.factor(data$DIABETES)
data$COPD <- as.factor(data$COPD)
data$ASTHMA <- as.factor(data$ASTHMA)
data$INMSUPR <- as.factor(data$INMSUPR)
data$HIPERTENSION <- as.factor(data$HIPERTENSION)
data$CARDIOVASCULAR <- as.factor(data$CARDIOVASCULAR)
data$RENAL_CHRONIC <- as.factor(data$RENAL_CHRONIC)
data$OTHER_DISEASE <- as.factor(data$OTHER_DISEASE)
data$OBESITY <- as.factor(data$OBESITY)
data$TOBACCO <- as.factor(data$TOBACCO)
#data$USMER < as.factor(data$USMER)
data$MEDICAL_UNIT <- as.factor(data$MEDICAL_UNIT)
data$INTUBED <- as.factor(data$INTUBED)
data$ICU <- as.factor(data$ICU)

str(data)
```

```
## tibble [76,832 x 18] (S3: tbl_df/tbl/data.frame)
##  $ MEDICAL_UNIT        : Factor w/ 13 levels "1","2","3","4",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ INTUBED             : Factor w/ 2 levels "1","2": 2 2 2 2 1 1 2 2 2 2 ...
##  $ PNEUMONIA           : Factor w/ 2 levels "1","2": 1 2 2 1 1 1 2 2 2 2 ...
##  $ AGE                 : num [1:76832] 40 37 25 80 58 48 25 24 25 30 ...
##  $ PREGNANT            : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ DIABETES            : Factor w/ 2 levels "1","2": 2 1 2 2 2 1 2 2 2 2 ...
##  $ COPD                : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ASTHMA              : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ INMSUPR             : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ HIPERTENSION        : Factor w/ 2 levels "1","2": 2 1 2 1 1 2 2 2 2 2 ...
##  $ OTHER_DISEASE       : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 1 ...
##  $ CARDIOVASCULAR      : Factor w/ 2 levels "1","2": 2 2 2 2 1 2 2 2 2 2 ...
##  $ OBESITY             : Factor w/ 2 levels "1","2": 2 1 2 2 1 2 2 2 2 2 ...
##  $ RENAL_CHRONIC       : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ TOBACCO             : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ CLASIFFICATION_FINAL: num [1:76832] 3 3 3 3 7 7 7 7 7 7 ...
##  $ ICU                 : Factor w/ 2 levels "1","2": 2 2 2 1 1 1 2 2 2 2 ...
##  $ DIED                : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Background: LDA and QDA

Linear Discriminant Analysis (LDA) and **Quadratic Discriminant Analysis (QDA)** are both classification tasks that predict the probability of a categorical outcome variable belonging to a specific class.

A class is made up of the results of each categories; for example, the categorical variable DIED has two classes:

- Class 1: (DIED = 1)
- Class 2: (DIED = 0)

Connection between LDA, QDA, and Logistic Regression: All three methods, LDA, QDA, and Logistic Regression, attempt to predict the probability of a categorical outcome variable based on a set of input variables. The primary difference between the three forms of regression lie in their assumptions:

- Logistic Regression: Does not have any distributional assumptions, but requires a categorical outcome variable.
- LDA: Assumes that the predictor variables are normally distributed, that the predictor variables share the same covariance matrix, and that the outcome variable is categorical. This produces a linear decision boundary.
- QDA: A version of LDA allows each class to have its own covariance matrix. This produces a quadratic decision boundary.

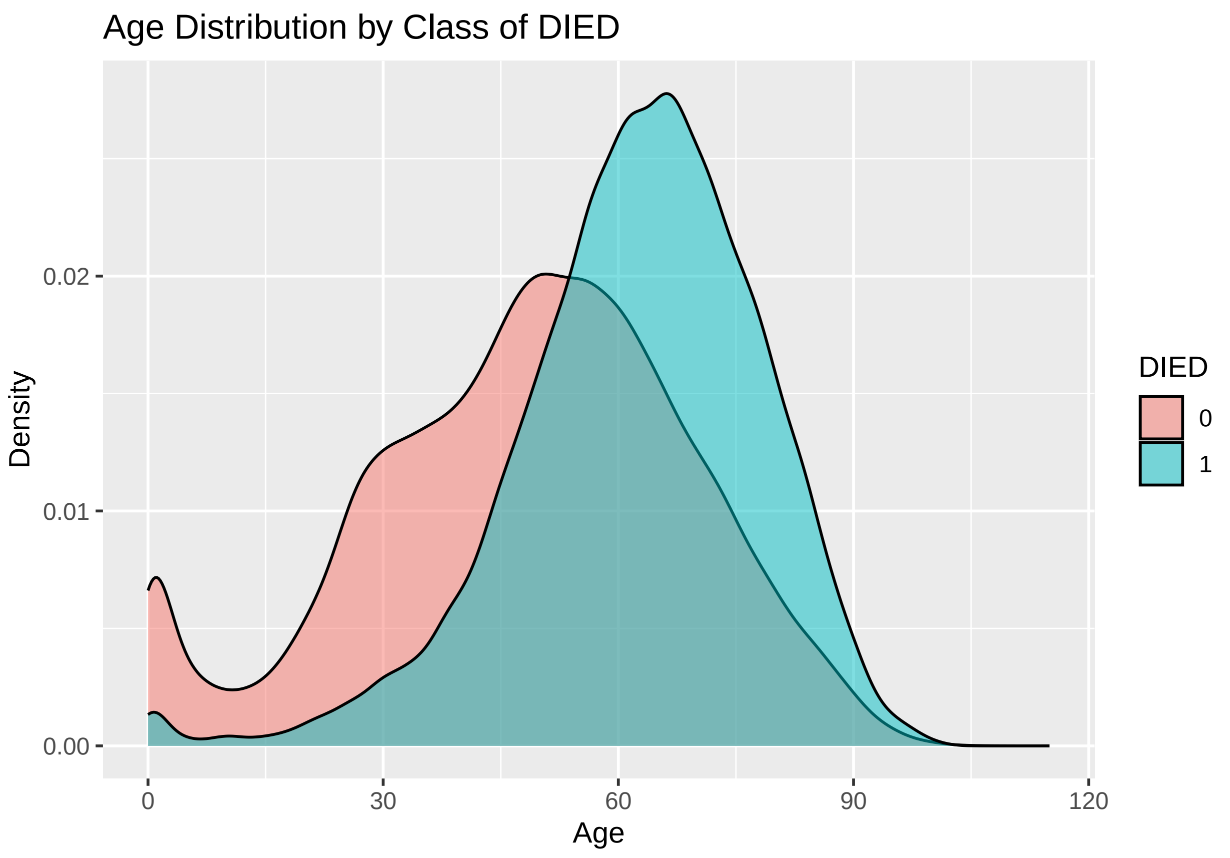
Decision Boundary: The function that separates classes.

Exploratory Data Analysis: Introductory Visualizations

Density Plot of Age and Class of DIED

The code chunk below produces density plots. The area is of each density plot represents the distribution of age corresponding to DIED=0 (survied) or DIED=1 (died), while the color of the density plot's area represents the class (DIED=0 or DIED=1).

```r
# Density Plot
ggplot(data, aes(x = AGE, fill = DIED)) +
  geom_density(alpha = 0.5) +
  labs(title = 'Age Distribution by Class of DIED', x='Age', y='Density')
```



```r
# Get the means of the ages of died and didn't die
died_age <- data %>% filter(DIED==1)
surv_age <- data %>% filter(DIED==0)
mean(died_age$AGE)
```

```
## [1] 62.44085
```

```r
mean(surv_age$AGE)
```

```
## [1] 48.51227
```

Both distributions have a relatively bell shaped curve, suggesting normality. The age of patients who died has a mean around 62.44, which is greater than the age of patients who did not die, whose distribution had a mean of 48.5.

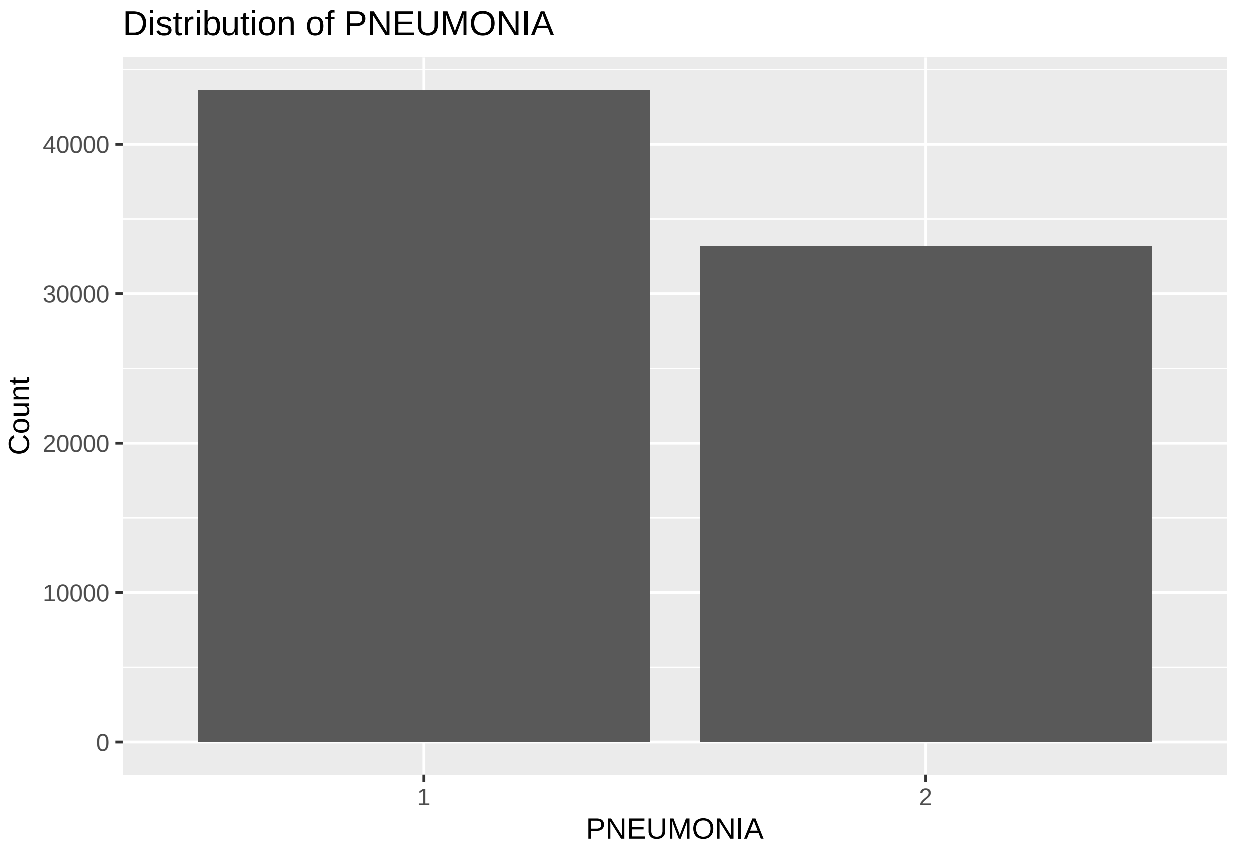
Distributions of Binary Predictors

In the graphs below, the distributions of pre-existing conditions are often associate with death by COVID-19 are shown.

```r
# Create function to use for each variable
bar <- function(data, var) {
  ggplot(data = data, aes_string(x = var)) +
    geom_bar() +
    labs(title = paste("Distribution of", var), x = var, y = "Count"
    )
}
```

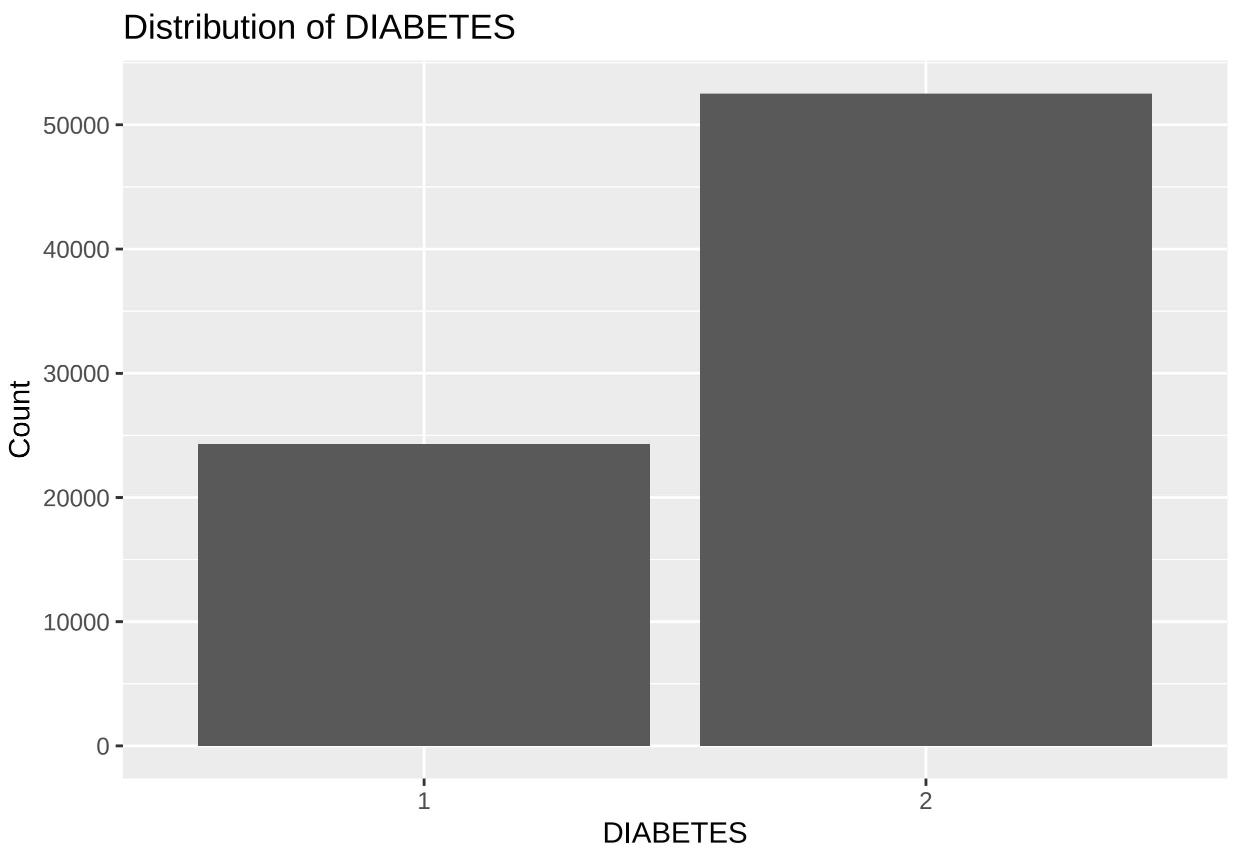
```r
# Bar plot for PNEUMONIA
bar(data, 'PNEUMONIA')
```

```
## Warning: `aes_string()` was deprecated in ggplot2 3.0.0.
## i Please use tidy evaluation idioms with `aes()`.
## i See also `vignette("ggplot2-in-packages")` for more information.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



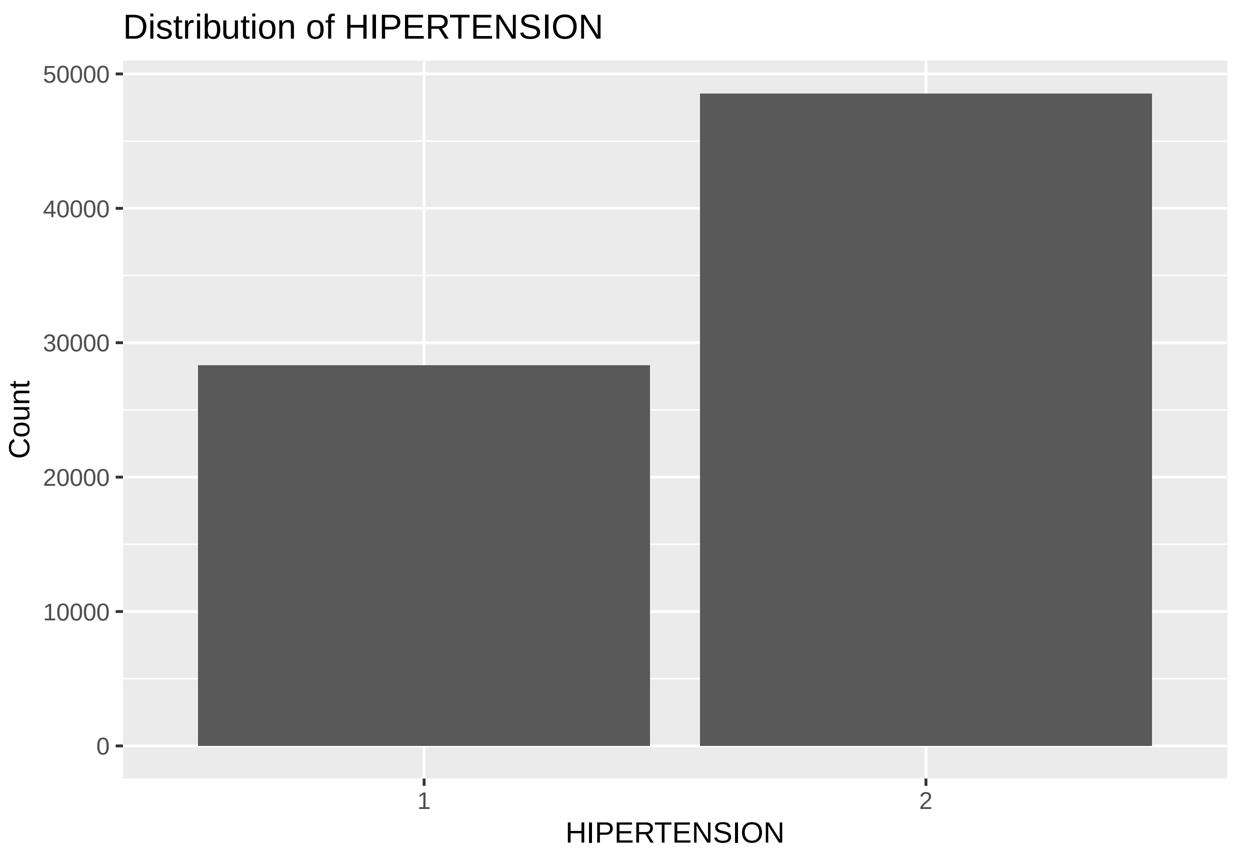
There were more patients with with pneumonia than without pneumonia.

```
# Bar plot for DIABETES
bar(data, 'DIABETES')
```



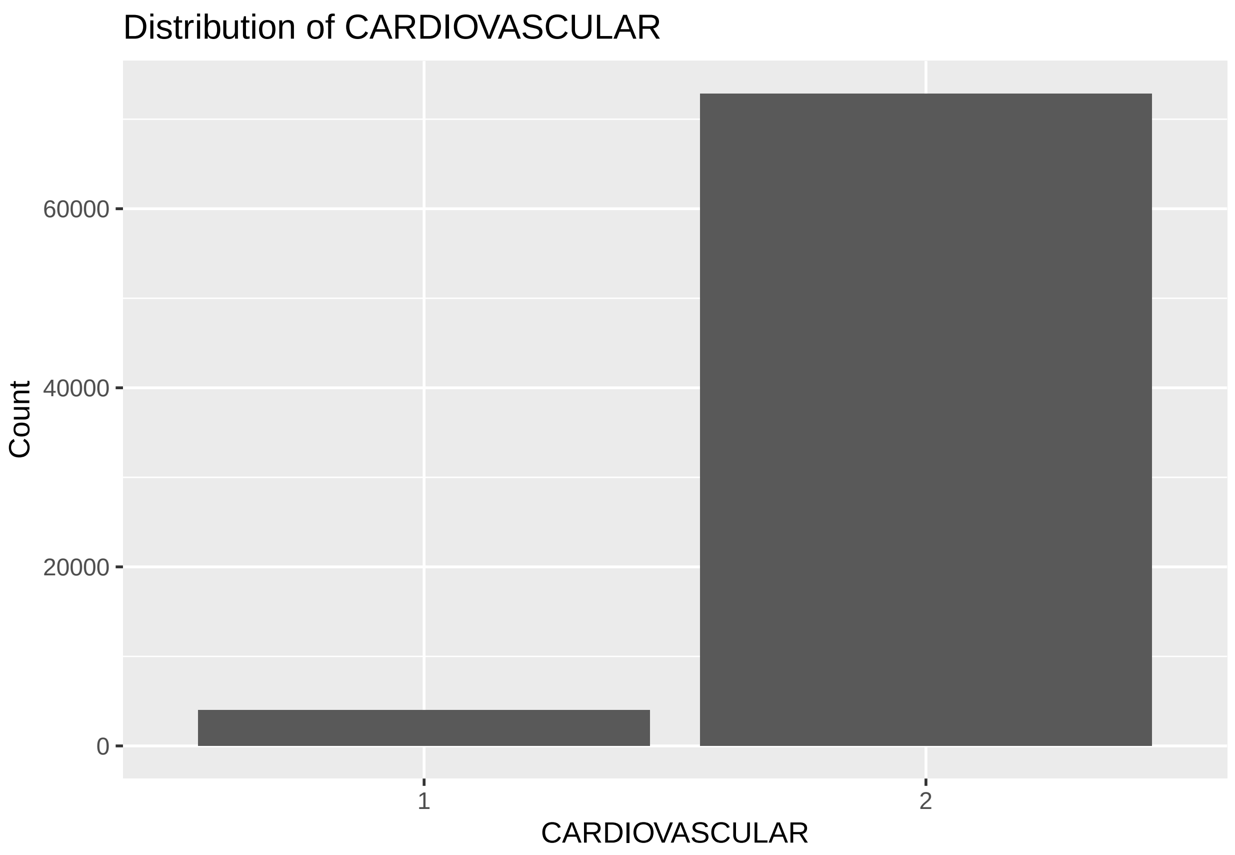
Significantly More patients did not have diabetes than did.

```r
# Bar plot for HIPERTENSION
bar(data, 'HIPERTENSION')
```



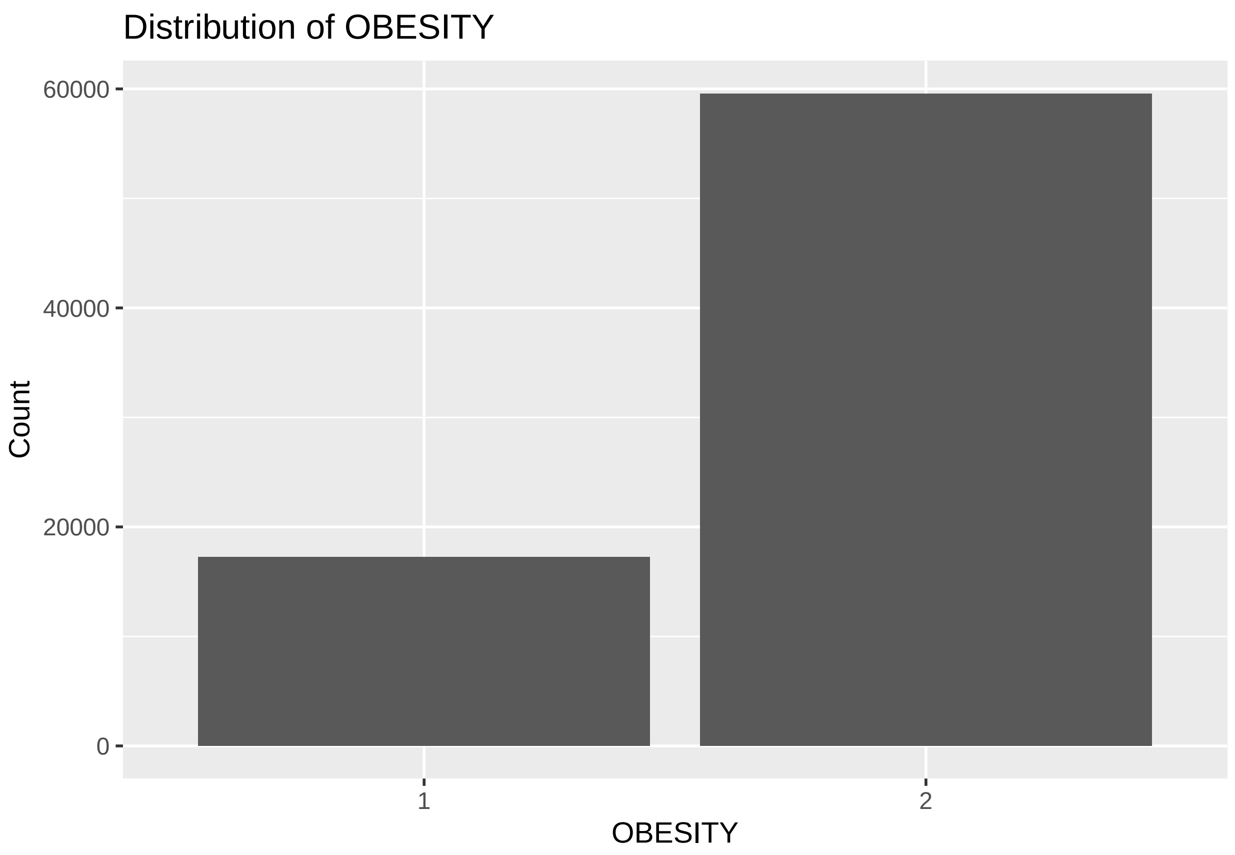
The majority of patients did not have hypertension

```r
# Bar plot for CARDIOVASCULAR
bar(data, 'CARDIOVASCULAR')
```



The vast majority of patients did not have cardiovascular disease

```
# Bar plot for OBESITY
bar(data, 'OBESITY')
```



The majority of patients were not obese.

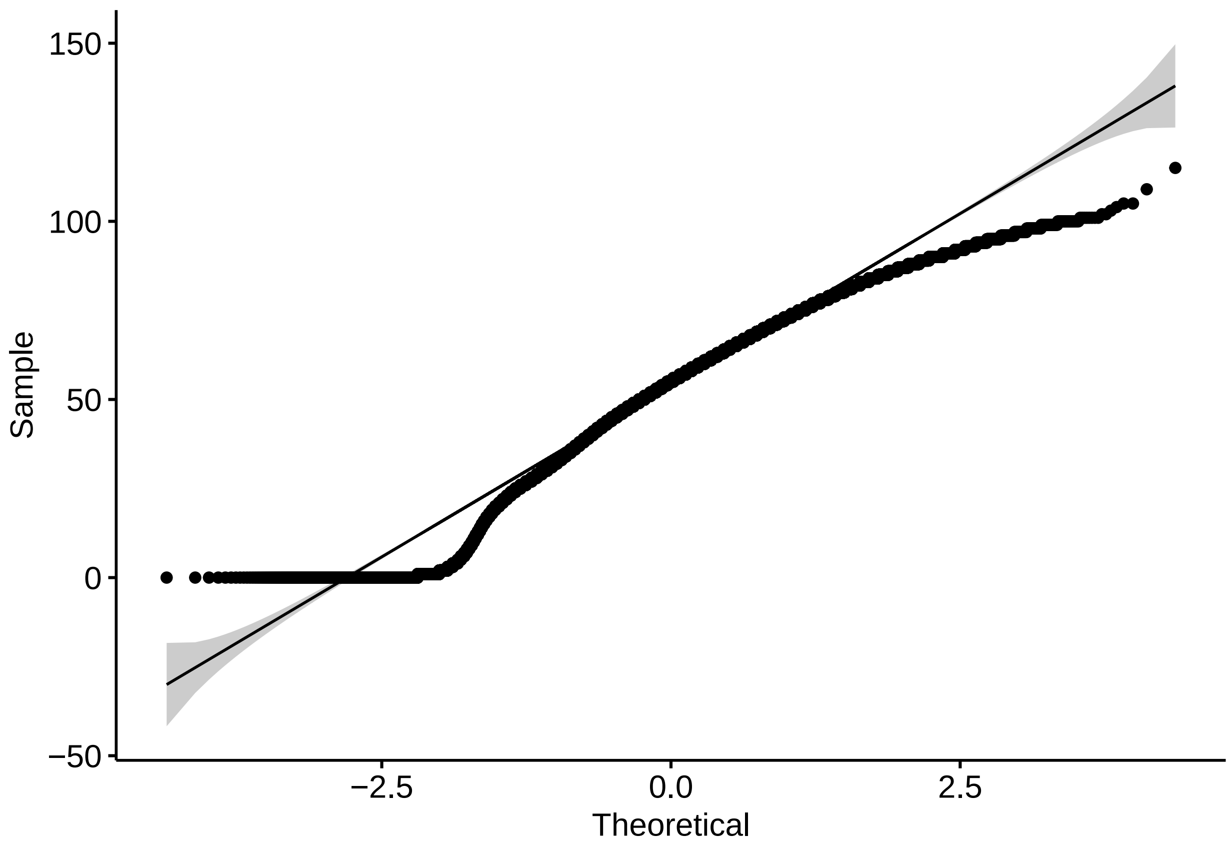
Exploratory Data Analysis: Assumptions

The Primary Assumptions of LDA and QDA are:

1. The outcome variable must be categorical; our outcome variable, **DIED**, is categorical. This was ensured in the code above.
2. LDA and QDA perform optimally when the predictor variables are continuous and normally distributed.
3. LDA and QDA assume that the variance is constant among classes in the outcome variable (homoscedasticity).
4. LDA and QDA assume that the variables are independent.

Assumption 2: Continuous Predictor variables are Normally Distributed

```r
# Normality of continuous variables
ggqqplot(data$AGE)
```



The one continuous predictor variable, age, may be normally distributed, but it is unclear. A Shapiro-Test is necessary.

```
# Normality of continuous variables
shapiro.test(sample(data$AGE,size=5000))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  sample(data$AGE, size = 5000)
## W = 0.97707, p-value < 2.2e-16
```

The output of the Shapiro-Wilk test provides evidence that age is not normally distributed. As age is the only continuous variable, it is unlikely that its distribution will greatly affect the model.

Assumption 3: The variance is constant among classes in the outcome variable (homoscedasticity)

```
# Normality of continuous variables of DIED
leveneTest(AGE ~ DIED, data = data)
```

```
## Levene's Test for Homogeneity of Variance (center = median)
##          Df F value    Pr(>F)
## group     1  2338.6 < 2.2e-16 ***
##       76830
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The assumption of homoscedasticity, which is the most important assumption in LDA and QDA analysis, is not violated.

Assumption 4: Sample measurements are independent from each-other

The design of the experiment suggests that the variables are independent.

Variable Selection

In this section, the variables that are most closely associated with COVID-19 death are defined. We must eliminate variables with any multi-collinearity or near-zero-variance first.

```
# Calculate VIF (from car package)
multi_col_model <- lm(DIED ~ ., data = data)
```

```
## Warning in model.response(mf, "numeric"): using type = "numeric" with a factor
## response will be ignored

## Warning in Ops.factor(y, z$residuals): '-' not meaningful for factors
```

```
vif(multi_col_model)
```

```
## Warning in Ops.factor(r, 2): '^' not meaningful for factors

## Warning in cov2cor(v): diag(V) had non-positive or NA entries; the non-finite
## result may be dubious

##                      GVIF Df GVIF^(1/(2*Df))
## MEDICAL_UNIT          NaN 12             NaN
## INTUBED               NaN  1             NaN
## PNEUMONIA             NaN  1             NaN
## AGE                   NaN  1             NaN
## PREGNANT              NaN  1             NaN
## DIABETES              NaN  1             NaN
## COPD                  NaN  1             NaN
## ASTHMA                NaN  1             NaN
## INMSUPR               NaN  1             NaN
## HIPERTENSION          NaN  1             NaN
## OTHER_DISEASE         NaN  1             NaN
## CARDIOVASCULAR        NaN  1             NaN
## OBESITY               NaN  1             NaN
## RENAL_CHRONIC         NaN  1             NaN
## TOBACCO               NaN  1             NaN
## CLASIFFICATION_FINAL  NaN  1             NaN
## ICU                   NaN  1             NaN
```

```
# Calculate the near-zero-variance (from caret)
near_zero_var <- nearZeroVar(data, saveMetrics = TRUE)
near_zero_var <- rownames(near_zero_var[near_zero_var$nzv == TRUE,])
near_zero_var
```

```
## [1] "PREGNANT" "COPD"     "ASTHMA"   "INMSUPR"  "TOBACCO"
```

The variables found to have multi-collinearity, near-zero-variance, or other negative traits are removed in the chunk below.

```
data <- data[,!names(data) %in% c("MEDICAL_UNIT", "USMER", "CLASIFFICATION_FINAL", "PREGNANT", "COPD",
str(data)
```

```
## tibble [76,832 x 12] (S3: tbl_df/tbl/data.frame)
##  $ INTUBED       : Factor w/ 2 levels "1","2": 2 2 2 2 1 1 2 2 2 2 ...
##  $ PNEUMONIA     : Factor w/ 2 levels "1","2": 1 2 2 1 1 1 2 2 2 2 ...
##  $ AGE           : num [1:76832] 40 37 25 80 58 48 25 24 25 30 ...
##  $ DIABETES      : Factor w/ 2 levels "1","2": 2 1 2 2 2 1 2 2 2 2 ...
##  $ INMSUPR       : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ HIPERTENSION  : Factor w/ 2 levels "1","2": 2 1 2 1 1 2 2 2 2 2 ...
##  $ OTHER_DISEASE : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 1 ...
##  $ CARDIOVASCULAR: Factor w/ 2 levels "1","2": 2 2 2 2 1 2 2 2 2 2 ...
##  $ OBESITY       : Factor w/ 2 levels "1","2": 2 1 2 2 1 2 2 2 2 2 ...
##  $ RENAL_CHRONIC : Factor w/ 2 levels "1","2": 2 2 2 2 2 2 2 2 2 2 ...
##  $ ICU           : Factor w/ 2 levels "1","2": 2 2 2 1 1 1 2 2 2 2 ...
##  $ DIED          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
```

Variable Selection

One of the most important parts of the model building process is variable selection. In this section, best subsets variable selection was used. As best subsets selection does not work on classes of 'lda' or 'qda', a logistic regression is used.

```
# LDA variable selection
full_model <- glm(DIED ~ ., data = data, family = binomial, trace = TRUE)
```

```
## Deviance = 73590.06 Iterations - 1
## Deviance = 72859.59 Iterations - 2
## Deviance = 72848.71 Iterations - 3
## Deviance = 72848.71 Iterations - 4
## Deviance = 72848.71 Iterations - 5
```

```
step_model <- stepAIC(full_model, direction = "both")
```

```
## Start:  AIC=72872.71
## DIED ~ INTUBED + PNEUMONIA + AGE + DIABETES + INMSUPR + HIPERTENSION +
##     OTHER_DISEASE + CARDIOVASCULAR + OBESITY + RENAL_CHRONIC +
##     ICU
##
## Deviance = 83583.53 Iterations - 1
## Deviance = 83303.29 Iterations - 2
## Deviance = 83301.75 Iterations - 3
## Deviance = 83301.75 Iterations - 4
## Deviance = 74634.23 Iterations - 1
## Deviance = 74051.46 Iterations - 2
## Deviance = 74045.44 Iterations - 3
## Deviance = 74045.44 Iterations - 4
## Deviance = 74045.44 Iterations - 5
## Deviance = 78048.07 Iterations - 1
## Deviance = 77904.45 Iterations - 2
## Deviance = 77904.38 Iterations - 3
## Deviance = 77904.38 Iterations - 4
## Deviance = 73745.06 Iterations - 1
## Deviance = 73037.75 Iterations - 2
## Deviance = 73027.88 Iterations - 3
## Deviance = 73027.87 Iterations - 4
## Deviance = 73027.87 Iterations - 5
## Deviance = 73593.73 Iterations - 1
## Deviance = 72863.08 Iterations - 2
## Deviance = 72852.19 Iterations - 3
## Deviance = 72852.19 Iterations - 4
## Deviance = 72852.19 Iterations - 5
## Deviance = 73617.48 Iterations - 1
## Deviance = 72882.38 Iterations - 2
## Deviance = 72871.45 Iterations - 3
## Deviance = 72871.45 Iterations - 4
## Deviance = 72871.45 Iterations - 5
## Deviance = 73623.12 Iterations - 1
## Deviance = 72892.78 Iterations - 2
## Deviance = 72881.92 Iterations - 3
## Deviance = 72881.92 Iterations - 4
## Deviance = 72881.92 Iterations - 5
## Deviance = 73606.91 Iterations - 1
## Deviance = 72886.08 Iterations - 2
## Deviance = 72875.61 Iterations - 3
## Deviance = 72875.61 Iterations - 4
## Deviance = 72875.61 Iterations - 5
## Deviance = 73590.18 Iterations - 1
## Deviance = 72867.91 Iterations - 2
## Deviance = 72857.56 Iterations - 3
## Deviance = 72857.55 Iterations - 4
## Deviance = 72857.55 Iterations - 5
## Deviance = 73692.83 Iterations - 1
## Deviance = 72976.64 Iterations - 2
## Deviance = 72966.29 Iterations - 3
## Deviance = 72966.29 Iterations - 4
## Deviance = 72966.29 Iterations - 5
## Deviance = 73705.94 Iterations - 1
## Deviance = 72989.14 Iterations - 2
## Deviance = 72978.81 Iterations - 3
## Deviance = 72978.8 Iterations - 4
## Deviance = 72978.8 Iterations - 5
##                   Df Deviance   AIC
## <none>                  72849 72873
## - INMSUPR          1    72852 72874
## - OBESITY          1    72858 72880
## - HIPERTENSION     1    72871 72893
## - CARDIOVASCULAR   1    72876 72898
## - OTHER_DISEASE    1    72882 72904
## - RENAL_CHRONIC    1    72966 72988
## - ICU              1    72979 73001
## - DIABETES         1    73028 73050
## - PNEUMONIA        1    74045 74067
## - AGE              1    77904 77926
## - INTUBED          1    83302 83324
```

```r
summary(step_model)
```

```
##
## Call:
## glm(formula = DIED ~ INTUBED + PNEUMONIA + AGE + DIABETES + INMSUPR +
##     HIPERTENSION + OTHER_DISEASE + CARDIOVASCULAR + OBESITY +
##     RENAL_CHRONIC + ICU, family = binomial, data = data, trace = TRUE)
```

```
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)      -0.2836398  0.0833094  -3.405 0.000662 ***
## INTUBED2         -2.5257761  0.0278591 -90.663  < 2e-16 ***
## PNEUMONIA2       -0.6657838  0.0194902 -34.160  < 2e-16 ***
## AGE               0.0388904  0.0005833  66.676  < 2e-16 ***
## DIABETES2        -0.2723666  0.0202993 -13.418  < 2e-16 ***
## INMSUPR2         -0.0881166  0.0470832  -1.872 0.061275 .
## HIPERTENSION2    -0.0998585  0.0209256  -4.772 1.82e-06 ***
## OTHER_DISEASE2   -0.2144426  0.0369573  -5.802 6.54e-09 ***
## CARDIOVASCULAR2   0.2045357  0.0396731   5.156 2.53e-07 ***
## OBESITY2         -0.0646925  0.0217211  -2.978 0.002898 **
## RENAL_CHRONIC2   -0.3926455  0.0359270 -10.929  < 2e-16 ***
## ICU2              0.4190061  0.0371452  11.280  < 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: 95191  on 76831  degrees of freedom
## Residual deviance: 72849  on 76820  degrees of freedom
## AIC: 72873
##
## Number of Fisher Scoring iterations: 5
```

According to the step-wise selection, all variables are highly statistically significant predictors of death by COVID-19. This is likely due to to previous heavy pruning of variables earlier with tests such as tests for multi-collinearity and non-zero-variance.

As the model's deviance did not drop by removing variables, as shown by the output of the stepAIC function, it is not necessary to remove any variables. Furthermore, these variables have already been tested for multi-collinearity.

Fitting the LDA Model

In the code below, the LDA Model is fit on the variables output by stepAIC.

```r
# Set the seed
set.seed(1)

# Fit the LDA model
lda_model <- lda(DIED ~ ., data = data)

# Make predictions
lda_predictions <- predict(lda_model)

# Confusion matrix
lda_conf <- table(Predicted = lda_predictions$class, Actual = data$DIED)
lda_conf
```

```
##          Actual
## Predicted     0     1
##         0 50114 14478
##         1  2864  9376
```

Fitting the QDA Model

In the code below, the QDA Model is fit on the variables output by stepAIC.

```
# Set the seed
set.seed(1)

# Fit the LDA model
qda_model <- qda(DIED ~ ., data = data)

# Make predictions
qda_predictions <- predict(qda_model)

# Confusion matrix
qda_conf <- table(Predicted = qda_predictions$class, Actual = data$DIED)
qda_conf
```

```
##          Actual
## Predicted     0     1
##         0 45155 12095
##         1  7823 11759
```

These models confirm that lda and qda work with the data. In the section below, they are refit on testing data, and their accuracy is tested.

Splitting the Data into Training Data and Testing Data

In order to test the accuracy, among other metrics, of the model, the train-test paradigm will be used.

```
set.seed(1)

# Establish n
n <- nrow(data)

# Train test split
tts <- rep(0:1,c(round(n*.3), n-round(n*.3)))

# Get the TTS Split
tts.split <- sample(tts, n)

# Visualize the split (0 = Testing, 1 = Training)
table(tts.split)
```

```
## tts.split
##     0     1
## 23050 53782
```

In the table above, the train test split is established. Approximately 70% (53782) of the data is allocated as testing data, while the other 30% (23050) is allocated as training data.

In the code-chunk below, the training and testing data is established

```
# Establish training and testing data
training_data <- data[tts.split==1, ]
testing_data <- data[tts.split==0, ]
```

Refitting the Model with the Train-Test Paradigm and Testing Model Accuracy

In the code chunk below, the models are fit on the training and testing data.

```
# Fit the LDA and QDA model on the training data
lda_model <- lda(DIED ~ ., data = training_data)
qda_model <- qda(DIED ~ ., data = training_data)

cat('\n','LDA MODEL', '\n')
```

```
##
##  LDA MODEL
```

```
lda_model
```

```
## Call:
## lda(DIED ~ ., data = training_data)
##
## Prior probabilities of groups:
##         0         1
## 0.6904541 0.3095459
##
## Group means:
##    INTUBED2 PNEUMONIA2      AGE DIABETES2  INMSUPR2 HIPERTENSION2
## 0 0.9499111  0.5027737 48.56038 0.7315668 0.9600097     0.6896914
## 1 0.6210356  0.2791326 62.51880 0.5721408 0.9570519     0.5003003
##   OTHER_DISEASE2 CARDIOVASCULAR2  OBESITY2 RENAL_CHRONIC2      ICU2
## 0      0.9379275       0.9532773 0.7905154      0.9486993 0.9365810
## 1      0.9276790       0.9371696 0.7414104      0.9121216 0.8886353
##
## Coefficients of linear discriminants:
##                          LD1
## INTUBED2        -2.423660588
## PNEUMONIA2      -0.484540151
## AGE              0.026002456
## DIABETES2       -0.206220719
## INMSUPR2        -0.089375827
## HIPERTENSION2   -0.115837326
## OTHER_DISEASE2  -0.153715891
## CARDIOVASCULAR2  0.147251773
## OBESITY2        -0.008263488
## RENAL_CHRONIC2  -0.309078156
## ICU2             0.309781544
```

```
cat('\n','QDA MODEL', '\n')
```

```
##
##  QDA MODEL
```

```
qda_model
```

```
## Call:
## qda(DIED ~ ., data = training_data)
##
## Prior probabilities of groups:
##         0         1
## 0.6904541 0.3095459
##
## Group means:
##    INTUBED2 PNEUMONIA2      AGE DIABETES2  INMSUPR2 HIPERTENSION2
## 0 0.9499111  0.5027737 48.56038 0.7315668 0.9600097     0.6896914
## 1 0.6210356  0.2791326 62.51880 0.5721408 0.9570519     0.5003003
##   OTHER_DISEASE2 CARDIOVASCULAR2  OBESITY2 RENAL_CHRONIC2      ICU2
## 0      0.9379275       0.9532773 0.7905154      0.9486993 0.9365810
## 1      0.9276790       0.9371696 0.7414104      0.9121216 0.8886353
```

Then the training models are used to make predictions on the testing data.

```r
# Make predictions on testing data
lda_model_predictions <- predict(lda_model, testing_data)$class
qda_model_predictions <- predict(qda_model, testing_data)$class
```

The confusion matrix for each model is visualized below

```r
# Visualize the tables
print("LDA MODEL")
```

```
## [1] "LDA MODEL"
```

```r
lda_conf <- table(predicted_deaths = lda_model_predictions, actual_deaths = testing_data$DIED)
lda_conf
```

```
##                 actual_deaths
## predicted_deaths     0     1
##                0 15014  4414
##                1   830  2792
```

```r
cat("\n")
```

```r
print("QDA MODEL")
```

```
## [1] "QDA MODEL"
```

```r
qda_conf <- table(predicted_deaths = qda_model_predictions, actual_deaths = testing_data$DIED)
qda_conf
```

```
##                 actual_deaths
## predicted_deaths     0     1
##                0 13502  3706
##                1  2342  3500
```

```r
cat("\n")
```

Chosing Best Model

Determining Model Accuracy

In the code below, the best model is chosen based on its accuracy, recall, and precision.

```r
# Get metrics for LDA
lda_tn <- lda_conf[1,1]
lda_fn <- lda_conf[1,2]
lda_fp <- lda_conf[2,1]
lda_tp <- lda_conf[2,2]
```

```r
# Get metrics for QDA
qda_tn <- qda_conf[1,1]
qda_fn <- qda_conf[1,2]
qda_fp <- qda_conf[2,1]
qda_tp <- qda_conf[2,2]

# Get accuracy for LDA
lda_accuracy <- sum(lda_tp + lda_tn) / sum(lda_tn + lda_fn + lda_fp + lda_tp)

# Get accuracy for QDA
qda_accuracy <- sum(qda_tp + qda_tn) / sum(qda_tn + qda_fn + qda_fp + qda_tp)

# Get recall for LDA
lda_recall <- lda_tp / sum(lda_tp, lda_fp)

# Get recall for QDA
qda_recall <- qda_tp / sum(qda_tp, qda_fp)

# Get precision for LDA
lda_precision <- lda_tp / sum(lda_tp, lda_fp)

# Get precision for QDA
qda_precision <- qda_tp / sum(qda_tp , lda_fp)

# Print
cat('LDA ACCURACY: ', lda_accuracy, '\n')
```

```
## LDA ACCURACY:  0.7724946
```

```r
cat('QDA ACCURACY: ', qda_accuracy, '\n')
```

```
## QDA ACCURACY:  0.7376139
```

```r
cat('\n')
```

```r
cat('LDA RECALL: ', lda_recall, '\n')
```

```
## LDA RECALL:  0.7708448
```

```r
cat('QDA RECALL: ', qda_recall, '\n')
```

```
## QDA RECALL:  0.5991099
```

```r
cat('\n')
```

```r
cat('LDA PRECISION: ', lda_precision, '\n')
```

```
## LDA PRECISION:  0.7708448
```

```r
cat('QDA PRECISION: ', qda_precision, '\n')
```

```
## QDA PRECISION:  0.8083141
```

```r
cat('\n')
```

As the LDA model has higher accuracy and higher recall, it is chosen as the best model, and will be assessed moving forward.

The LDA model is 77% accurate in making classifications, meaning that 77% of all predictions made by the model will be correct.

Understanding the Best Model

The coefficients of the best model are output below.

```
# Printing the coefficients of the model
lda_model$scaling
```

```
##                         LD1
## INTUBED2        -2.423660588
## PNEUMONIA2      -0.484540151
## AGE              0.026002456
## DIABETES2       -0.206220719
## INMSUPR2        -0.089375827
## HIPERTENSION2   -0.115837326
## OTHER_DISEASE2  -0.153715891
## CARDIOVASCULAR2  0.147251773
## OBESITY2        -0.008263488
## RENAL_CHRONIC2  -0.309078156
## ICU2             0.309781544
```

The coefficients printed above are linear discriminant coefficients. Each linear discriminant coefficient indicates how much the variable contributes to the observation's class determination, 'DIED' = 0 or 'DIED' = 1.

The coefficients indicate that variables are positively or negatively associated with death from COVID-19.

Positive Association With Death by COVID-19 - AGE - CARDIOVASCULAR - ICU

Negative Association With Death by COVID-19 - INTUBED - PNEUMONIA - DIABETES - INMSUPR - HIPERTENSION - OTHER_DISEASE - OBESITY - RENAL_CHRONIC

The 'Cut Off' Parameter and Measures of Accuracy

LDA/QDA 'Cut Off' Parameter

The **Cut Off* parameter in LDA and QDA models determines the point (probability) at which an observation is considered one class of the outcome variable or another. The Cut Off parameter, by default, is set to 0.5.

For instance, in this example, the Cut Off parameter determines whether or not an observation, output by the classification model, is classified as 'DIED' == 1 (died) or 'DIED' == 0 (survived). If the Cut Off parameter is set to 0.5, all observations resulting in a probability of 50% and higher of death, as predicted by the LDA or QDA model, would have their outcome variable's class, 'DIED', set to 1.

Affect of Cut Off On Recall

Increasing the Cut Off parameter forces less parameters to be considered positive. In this scenario, increasing the Cut Off parameter would cause less deaths to be predicted.

Therefore, more true positives may be considered negative. This causes an increased number of false negatives, reducing the recall.

Affect of Cut Off On Precision

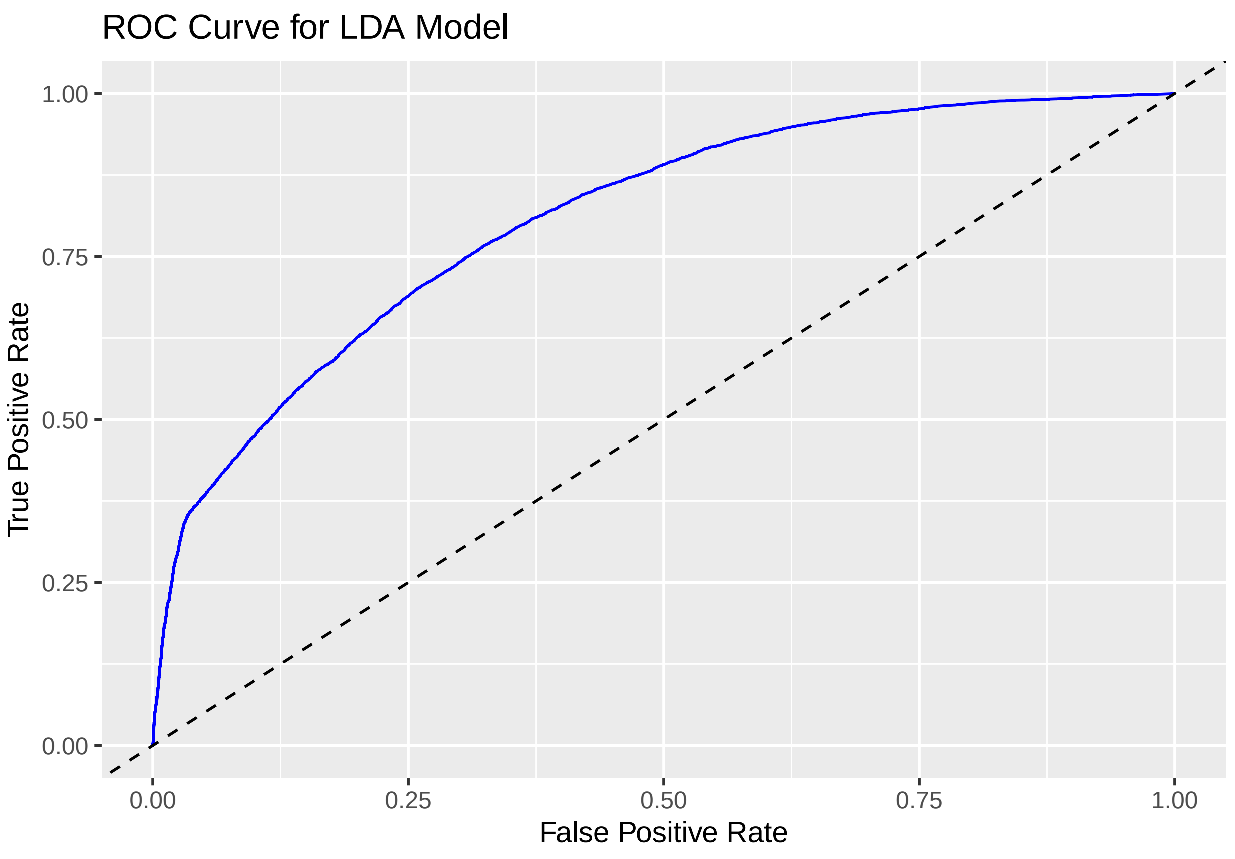
Alternatively, increasing the Cut Off parameter can increase precision by reducing the number of false positives.

ROC Curve and Conclusion

```r
# Cut Off
cutoff <- 0.5

# Fix lda_model_predictions variable
lda_model_predictions <- predict(lda_model, testing_data)

# ROC Curve Step 1: Get probabilities for positive class
lda_probabilities <- lda_model_predictions$posterior[,2]

# ROC Curve Step 2: Create ROC curve
lda_roc <- roc(testing_data$DIED, lda_probabilities)
```

```
## Setting levels: control = 0, case = 1

## Setting direction: controls < cases
```

```r
# ROC Curve Step 3: Create dataframe with columns tpr and fpr
lda_roc_df <- data.frame(tpr = lda_roc$sensitivities, fpr = 1 - lda_roc$specificities)

# ROC Curve Step 4: Graph
ggplot(lda_roc_df, aes(x = fpr, y = tpr)) +
  geom_line(color = 'blue') +
  labs(title = "ROC Curve for LDA Model",
       x = "False Positive Rate",
       y = "True Positive Rate") +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed")
```



The blue ROC Curve represents the True Positive Rate against the False Positive Rate; it is compared to the dashed line, which represents typical random chance (an AUC of 0.5).

The greater the area under the curve (the closer the function gets to the top left corner of the graph), the more accurate the model is.

The area under the curve is calculated below.

```r
# Area under curve computation
auc(lda_roc)
```

```
## Area under the curve: 0.8074
```

The Area Under the Curve Calculation is 0.8, meaning that the model performs 30% (calculation: 0.80-0.50) better than random chance.

In the case of this analysis, it is more concerning to make a false negative than a false positive, as the outcome of a false negative may be death without treatment. If a false positive is predicted, the patient can take actions to prevent death by COVID-19; if a flase negative is produced, a patient cannot.

Conclusions

By convention, an AUC score from 0.8-0.9 is considered excellent, meaning that the model's prediction of death by COVID-19 is 'excellent'. The accuracy of the model is 77%, meaning that 77% of all predictions will be correct.