Effects of Smoking, Age, and Number of Blood Platelets on Chance of Death After Heart Failure

2023-20-19

Authors: Ian Murday, Norvell Bartow, Rayan Hassan

```
library(dplyr)
library(tidyr)
library(ggplot2)
library(readr)
library(e1071)
library(glmnet)
library(randomForest)
library(rpart)
library(rpart.plot)

df =
read.csv("C:/Users/RAYAN/OneDrive/Desktop/heart_failure_clinical_records_data
set.csv", header = TRUE, sep = ",")
```

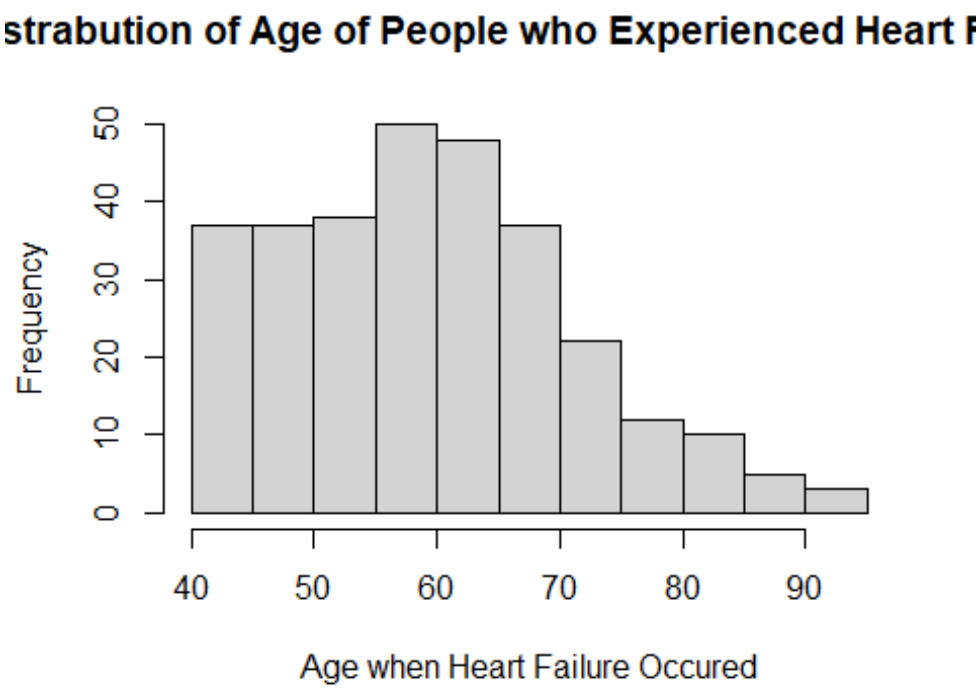
Introduction

Heart failure is a cardiovascular condition that affects millions of individuals worldwide. It is characterized by the heart's inability to pump blood effectively, leading to several problems and sometimes even death. Many factors can affect and contribute to its severity. However, in this study we first intend to look at how age, number of blood platelets and whether a person smokes affect someone's chances of living after having experienced heart failure using graphics. Then, we plan to use logistic regression, a decision tree, naive bayes, and a random forest in order to create an accurate model that predicts the death event. For our machine learning models we plan to us all variables in our data set except time in order to figure out which machine learning method can create the most accurate model.

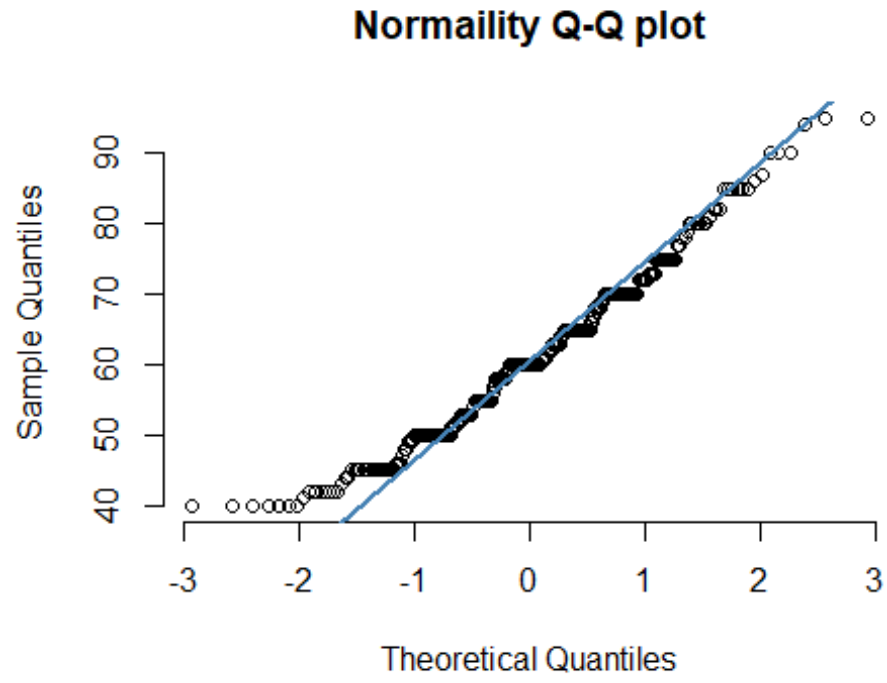
Distrabution of Quantatative Variables

Before starting an analysis of how age, number of blood platelets and whether the person smokes affect the chance of living after heart failure, it is important to check the distribution, symmetry, and normality of our quantitative variables. In the case of this study, age and blood platelets are the quantitative variables.

```
hist(
  df$age,
  main = "Distrabution of Age of People who Experienced Heart Failure",
  xlab = "Age when Heart Failure Occured"
  )
```

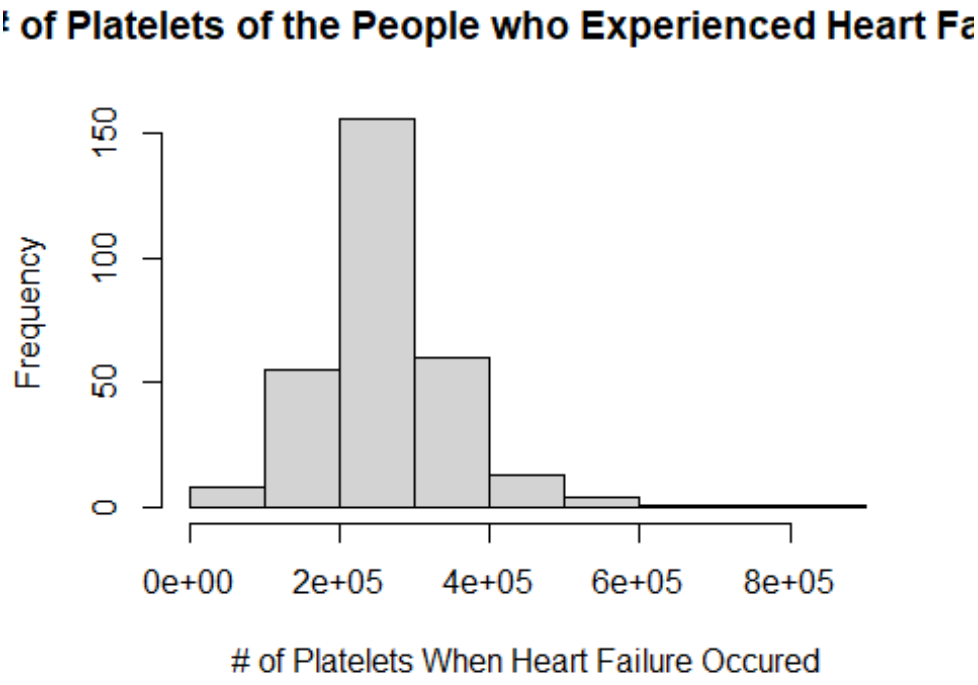



```r
qqnorm(df$age, pch = 1, frame = FALSE, main = "Normaility Q-Q plot")
qqline(df$age, col = "steelblue", lwd = 2)
```

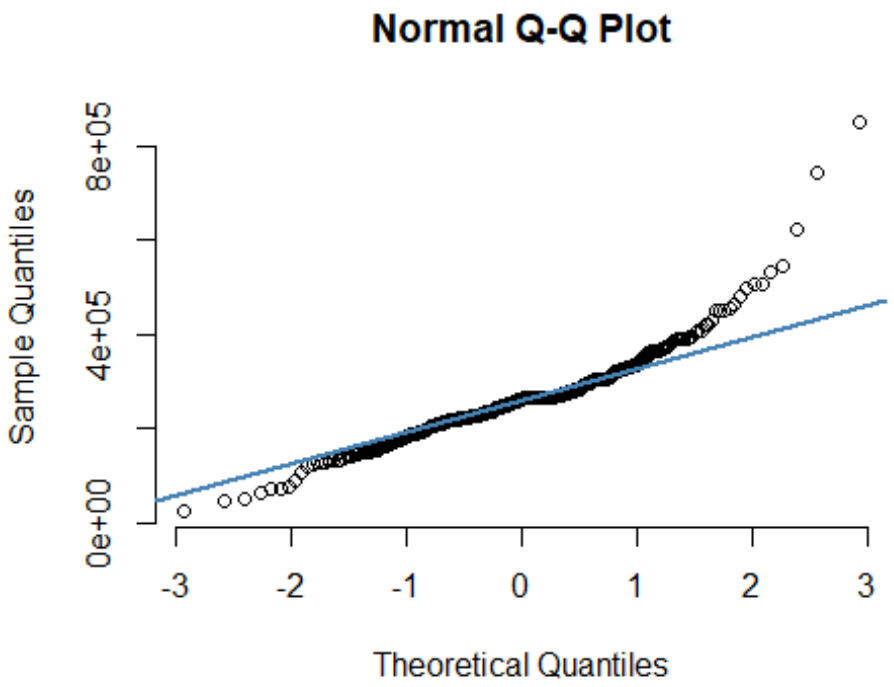


Above is a histogram and a QQ plot of the age of all of the people in our dataset who experienced heart failure. As can be seen in the histogram, our data doesn't appear to be very symmetric as it is right-skewed. Furthermore, it looks like outliers occur in the upper age ranges. Lastly, as it can be seen from the histogram and the QQ plot, this variable does not follow a normal distribution. The histogram doesn't show a normal bell shaped curve which would indicate normality. Also, on the QQ plot there is a long tail on the lower end, a short tail on the upper end, and the points zig zag the reference line throughout.

```
hist(
  df$platelets,
  main = "# of Platelets of the People who Experienced Heart Failure",
  xlab = "# of Platelets When Heart Failure Occured"
  )
```




```
qqnorm(df$platelets, pch = 1, frame = FALSE)
qqline(df$platelets, col = "steelblue", lwd = 2)
```



The blood platelets distributions can be seen above. The histogram seems to have a much more symmetrical distribution than age. It can be seen above in the histogram that there are certainly a lot of outliers on the upper end of the spectrum. But, where the density is the highest, it is symmetrical. That being said, this variable still isn't normal. On the QQ plot there is a curved tail at the end of the plot showing a non normal distribution.

Analysis

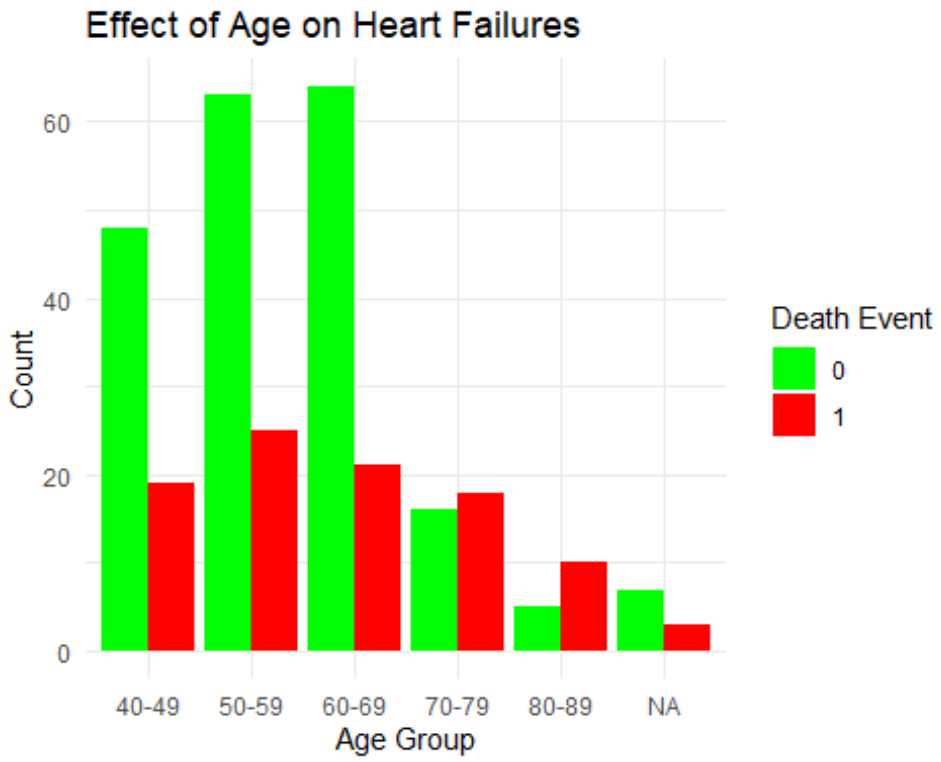
Below is the structure of the data that we retrieved off of kaggle. Death event is our response variable while age, smoking, and platelets are our explanatory variables.

```
df %>%
  select(age, smoking, platelets, DEATH_EVENT)%>%
  head(10)
```

```
##    age smoking platelets DEATH_EVENT
## 1   75       0    265000           1
## 2   55       0    263358           1
## 3   65       1    162000           1
## 4   50       0    210000           1
## 5   65       0    327000           1
## 6   90       1    204000           1
## 7   75       0    127000           1
## 8   60       1    454000           1
## 9   65       0    263358           1
## 10  80       1    388000           1
```

The code below is used to generate a histogram showing the effect of age on death by heart failure. Note that the labels have been divided into 6 age groups to help better visualize the data.

```
df$age_group <- cut(df$age, breaks = seq(40, 90, by = 10), labels = c("40-
49", "50-59", "60-69", "70-79", "80-89"))

ggplot(df, aes(x = age_group, fill = factor(DEATH_EVENT))) +
  geom_bar(position = "dodge") +
  labs(
    x = "Age Group",
    y = "Count",
    fill = "Death Event",
    title = "Effect of Age on Heart Failures"
  ) +
  scale_fill_manual(values = c("0" = "green", "1" = "red")) +
  theme_minimal()
```



At first glance, the histogram above seems to show that in total, older people are not likely to die from heart failures as the red bars (corresponding to deaths) for age groups "70-79" and "80-89" are shorter than the ones corresponding to younger groups. However, this is misleading because the variable that should be taken into account is the ratio of deaths to non deaths for each age group. When comparing the green bars (corresponding to non-deaths) to the red ones, one can have a better understanding of the results. The red bars for age groups "70-79" and "80-89" are longer than the green ones corresponding to these same age groups. That means that older people are more likely to die from heart failures.

Where as for younger age groups, the green bars are considerably longer than the red ones, signaling that younger people are not as likely to die from heart failures.

Let us study the ratio of deaths between older and younger individuals based on a specified age threshold. In this case, the threshold is chosen to be 65 as it seems reasonable looking at the histogram. In order to do that, the following code first categorizes individuals into "old" and "young" groups using the defined age threshold. Then, it calculates the ratio of deaths for each group, providing insights into whether age is a significant factor in heart failure-related deaths.

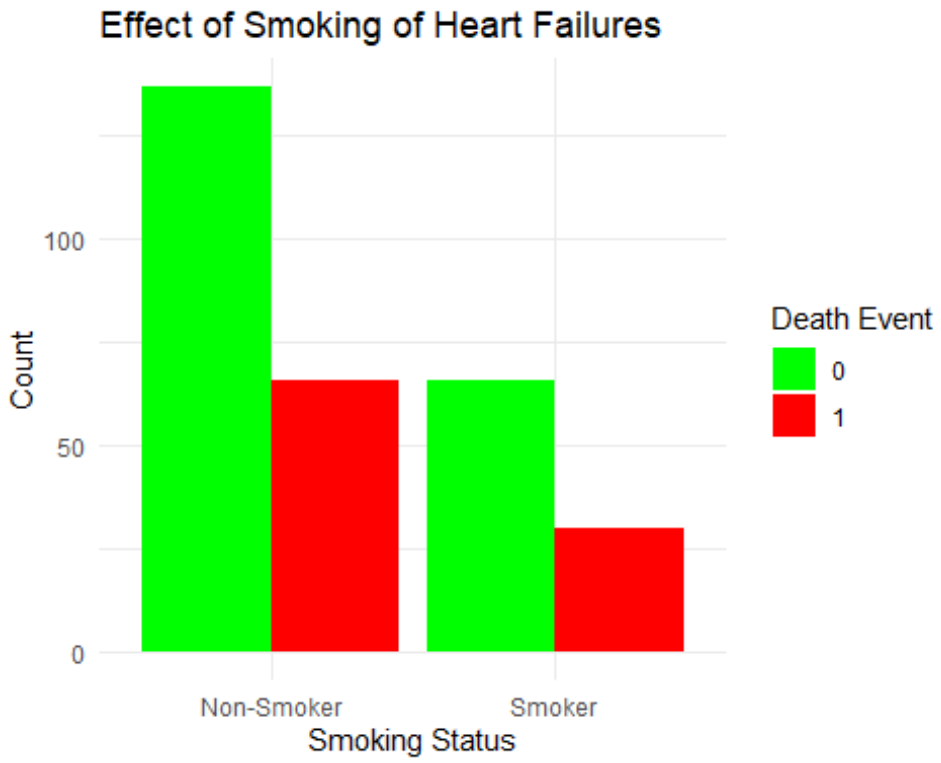
```
age_threshold <- 65

df <- df %>%
  mutate(age_group = ifelse(age >= age_threshold, 'old', 'young'))

death_ratio <- df %>%
  group_by(age_group, DEATH_EVENT) %>%
  summarise(count = n()) %>%
  spread(DEATH_EVENT, count, fill = 0) %>%
  mutate(ratio = `1` / (`0` + `1`))

print(death_ratio)
```

```
## # A tibble: 2 × 4
## # Groups:   age_group [2]
##   age_group   `0`   `1` ratio
##   <chr>     <dbl> <dbl> <dbl>
## 1 old          65    50 0.435
## 2 young       138    46 0.25
```

The results above show that 43.5% of people older than 65 years old die from heart failures, whereas only 25% of people younger than 65 die of that same reason. These results clearly show that age is a significant factor associated with heart failure-related deaths.

Another factor that we studied is smoking. The following code creates a histogram to visualize the effect of smoking on heart failure deaths.

```
ggplot(df, aes(x=factor(smoking, labels = c("Non-Smoker", "Smoker")),
fill=factor(DEATH_EVENT))) + geom_bar(position="dodge") +
  labs(
    x="Smoking Status",
    y="Count",
    fill = "Death Event",
    title = "Effect of Smoking of Heart Failures"
  ) +
  scale_fill_manual(values= c("0"="green","1"="red"))+theme_minimal()
```

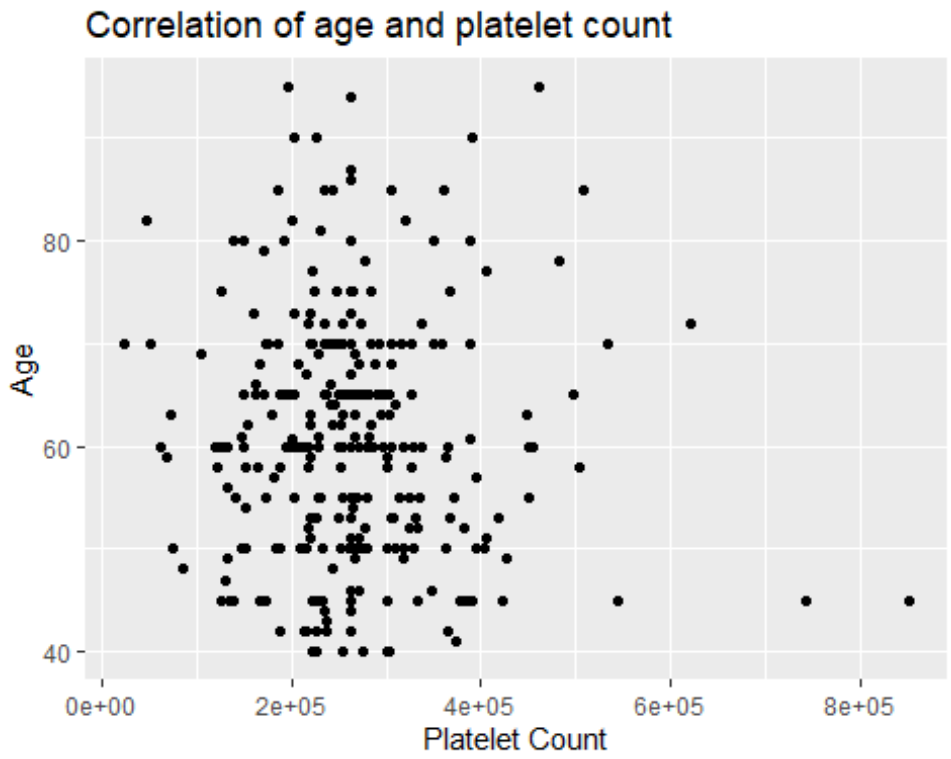


Looking at the histogram, the ratios of heart failure-related deaths for smokers and non-smokers look almost equal. To verify that with numbers, a similar code calculates those ratios.

```
smoking_death_ratio <- df %>%
  group_by(smoking, DEATH_EVENT) %>%
  summarise(count = n()) %>%
  pivot_wider(names_from = DEATH_EVENT, values_from = count, values_fill = 0)
%>%
  mutate(ratio = `1` / (`0` + `1`))

print(smoking_death_ratio)

## # A tibble: 2 × 4
## # Groups:   smoking [2]
##   smoking   `0`   `1` ratio
##     <int> <int> <int> <dbl>
## 1       0   137    66 0.325
## 2       1    66    30 0.312
```

As seen in the results above, the ratios are indeed almost the same (around 32%). This shows that there is no significant difference in the likelihood of heart failure-related deaths between smokers and non-smokers. Although these results might sound inaccurate, it is important to keep in mind that they are representative of the specific data set used which in itself could have limitations such as the size.

```
data <- ggplot(data = df, aes(platelets, age))
 data + geom_point() + labs(x = "Platelet Count",y = "Age",
                            title = "Correlation of age and platelet count")
```



The graph above was used to show if there was any correlation between a person's age and their platelet counts. As we examine the scatter plot we can see that the majority of the individuals studied in this data set have a platelet count of around 200,000 to 300,000. There are definitely some outliers, which we mainly see in the younger individuals of the group. However, despite the outliers, there are around 200,000 to 300,000 platelets in all of the age groups, 40 years old to 90 years old. This has led us to believe that there is not a correlation between someone's age and the number of platelets that they have.

```
df$platelet_group <- cut(df$platelets, breaks = seq(25100, 850000, by =
200000))
ggplot(df, aes(x = platelets, fill = factor(DEATH_EVENT))) +
  geom_boxplot(position = "dodge") +
  labs(x = "platelets group", y = "count", fill = "Death Event", title =
"Effect of Platelets on Heart Failure") +
  scale_fill_manual(values = c("0" = "green", "1" = "red"))
```

The graph above is showing the correlation between the number of platelets that a person has and its correlation with dying from heart disease. As we look at the graph we see that the results look almost 50/50 with a large outlier of a death event occurring in the middle of the pack. Due to this we can infer that the number of platelets do not increase the chances of a person dying from heart failure.

Another way to determine if variables are significant predictors is to build a model with said variables and look at their significance. Below is a model using smoking, age, and blood platelets to try and predict whether a person died from their heart failure.

```
modelDeathEvent = glm(DEATH_EVENT ~ age + platelets + smoking , data = df,
family = "binomial")
summary(modelDeathEvent)

##
## Call:
## glm(formula = DEATH_EVENT ~ age + platelets + smoking, family =
"binomial",
##     data = df)
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.373e+00  7.987e-01  -4.223 2.41e-05 ***
## age          4.683e-02  1.109e-02   4.222 2.42e-05 ***
## platelets   -9.587e-07  1.385e-06  -0.692    0.489
## smoking     -7.671e-02  2.753e-01  -0.279    0.781
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 375.35  on 298  degrees of freedom
## Residual deviance: 355.42  on 295  degrees of freedom
## AIC: 363.42
##
## Number of Fisher Scoring iterations: 4
```

The significance of the independent variables in this model comes at no surprise after the analysis. Both platelets and whether a person smokes seemed to be random when compared against the death event and that shows in this model. Platelets have a p-value of 0.49 in the model and whether a person smokes has a p-value of 0.78 which are both very high p-values. Furthermore, there were certainly age grouping that had higher death rates after heart failure which is why it would make sense to see age's p-value much below 0.01. So, through our analysis and the model above, it can be confidently concluded that the number of blood platelets and smoking are not significant predictors to predict death when someone experiences heart failure. Also, we can confidently conclude that the age of the person that experienced heart failure is a significant predictor to predict death when someone experiences heart failure.

Now that we have determined which of the predictors are significant, it is important to evaluate the model that is supposed to predict death. To do this we will save the logit of the predicted probability from the model.Then, categorize whether the person lives or passes by splitting probability at 0.5. Lastly, a table will be printed in order to interpret the results.

```
df$pred_death = predict(modelDeathEvent)
df$prob = exp(df$pred_death)/(1+exp(df$pred_death))
df$predOut = ifelse(df$prob > 0.5, 1, 0)
table(df$predOut, df$DEATH_EVENT)

##
##       0   1
##   0 195  77
##   1   8  19
```

In order to find accuracy of this model, we must sum the number of times the model predicted a death when a death occurred with the times the model predicted heart failure did not cause death and a death did not occur, all over the total number of observations. This works out to be 0.716 meaning our model predicts the death event of an observation from the data set correctly 71.6% of the time. This is a very mediocre accuracy, it isn't good but it isn't necessarily bad.

Machine Learning Models

```
dim(df)

## [1] 299  18

names(df)
```

```
##  [1] "age"                       "anaemia"
##  [3] "creatinine_phosphokinase" "diabetes"
##  [5] "ejection_fraction"        "high_blood_pressure"
##  [7] "platelets"                "serum_creatinine"
##  [9] "serum_sodium"             "sex"
## [11] "smoking"                  "time"
## [13] "DEATH_EVENT"              "age_group"
## [15] "platelet_group"           "pred_death"
## [17] "prob"                     "predOut"
```

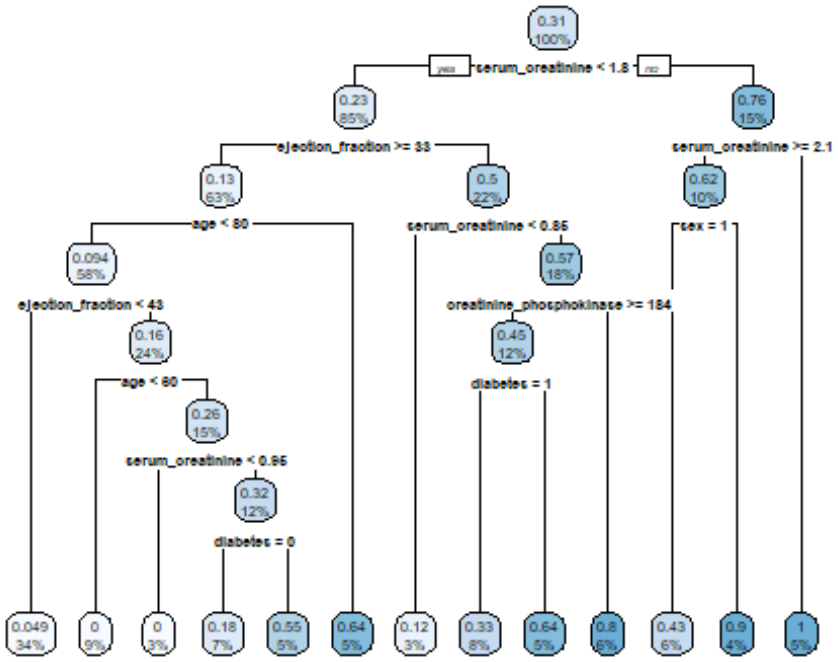
```
glimpse(df)
```

```
## Rows: 299
## Columns: 18
## $ age                       <dbl> 75, 55, 65, 50, 65, 90, 75, 60, 65, 80,
75, 6…
## $ anaemia                   <int> 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1,
1, …
## $ creatinine_phosphokinase <int> 582, 7861, 146, 111, 160, 47, 246, 315,
157, …
## $ diabetes                  <int> 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0,
0, …
## $ ejection_fraction         <int> 20, 38, 20, 20, 20, 40, 15, 60, 65, 35,
38, 2…
## $ high_blood_pressure       <int> 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1,
1, …
## $ platelets                 <dbl> 265000, 263358, 162000, 210000, 327000,
20400…
## $ serum_creatinine          <dbl> 1.90, 1.10, 1.30, 1.90, 2.70, 2.10, 1.20,
1.1…
## $ serum_sodium              <int> 130, 136, 129, 137, 116, 132, 137, 131,
138, …
## $ sex                       <int> 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1,
0, …
## $ smoking                   <int> 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1, 0, 0,
0, …
## $ time                      <int> 4, 6, 7, 7, 8, 8, 10, 10, 10, 10, 10, 10,
11,…
## $ DEATH_EVENT               <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, …
## $ age_group                 <chr> "old", "young", "old", "young", "old",
"old",…
## $ platelet_group            <fct> "(2.25e+05,4.25e+05]",
"(2.25e+05,4.25e+05]",…
## $ pred_death                <dbl> -0.11439362, -1.04951214, -0.56069986, -
1.232…
## $ prob                      <dbl> 0.4714327, 0.2593188, 0.3633855,
0.2257389, 0…
## $ predOut                   <dbl> 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0, …
```

```
data2<- df %>%
  select(DEATH_EVENT, smoking, sex, serum_sodium, serum_sodium,
serum_creatinine, platelets, high_blood_pressure, ejection_fraction,
diabetes, creatinine_phosphokinase, anaemia, age) %>% drop_na()

set.seed(100)
train2 <- df %>% sample_frac(size = 0.8, fac=DEATH_EVENT)
test2 <- df %>% setdiff(train2)
```

The code above displays all variables and a glimpse of what the data they contain looks like. In the following section we will be including all variables expect time in the machine learning models in attempt to create the most accurate model. Time is neglected from all models because it doesn't have a description on kaggle where the data set was downloaded from and all models preform better without it. As seen above, the data has been split into 80% for training and 20% for testing.

```
library(rpart)
library(rpart.plot)
form_full = as.formula(DEATH_EVENT ~ smoking + sex + serum_sodium +
serum_creatinine + platelets + high_blood_pressure + ejection_fraction +
diabetes + creatinine_phosphokinase + anaemia + age)

predictors <- model.matrix(form_full, data = train2)
mod_tree <- rpart(form_full,data=train2)
rpart.plot(mod_tree)
```



Above is a decision tree that was trained using all of the variables in the data set. We are usualy able to conclude what variables are most important in a model based on how high the variable is in the decision tree. Above we see that Serum Creatinine levels, ejection fraction, and age are the most important variables in this data set. We neglected two of these variables in our first analysis which is one of the reasons why the first logistic regression model is so poor. Due to the nature of the data set the accuracy and sensitivity statistics weren't able to be retrieved for the decision tree but it is still important to observe which variables are important using the tree. It isn't vital that these performance statistics were able to be retrieved since a random forst is included in this analysis later on.

Another model that we trained is Naive Bayes. The training and testing data used is the same as the other models. To visualize how the model performed, we have created a confusion matrix.

```
mod_nb <- naiveBayes(DEATH_EVENT ~ .,train2)

confusion_matrix <- function(data,y,mod){
  confusion_matrix <- data %>%
  mutate(pred = predict(mod, newdata = data, type = "class"),
         y=y) %>%
  select(y,pred) %>% table()
}
misclass <- function(confusion){
  misclass <- 1- sum(diag(confusion))/sum(confusion)
  return(misclass)
}

confusion = confusion_matrix(test2, test2$DEATH_EVENT, mod_nb)
confusion
```

```
##    pred
## y    0  1
##   0 31  7
##   1 12 10
```

```
misclas = misclass(confusion)
misclas
```

```
## [1] 0.3166667
```

The confusion matrix above shows the false positives, true positives, false negatives and true negatives. From the table, we can see that among 60 testing samples, 12 are false negatives, meaning they were predicted '0' when they are actually '1'. Similarly, 7 samples are false positives, meaning they were predicted '1' when they should have been predicted '0'. The misclassification rate is therefore around 31.7% as shown in the results. Which corresponds to a testing accuracy of around 68.3%.

Below we built a logistic regression model but this time we included all of the variables in order to make the model more accurate.

```
regressionModel = glm(DEATH_EVENT ~ smoking + sex + serum_sodium +
serum_sodium + serum_creatinine + platelets + high_blood_pressure +
ejection_fraction + diabetes + creatinine_phosphokinase + anaemia + age, data
= train2, family = "binomial")
summary(regressionModel)
```

```
##
## Call:
## glm(formula = DEATH_EVENT ~ smoking + sex + serum_sodium + serum_sodium +
##     serum_creatinine + platelets + high_blood_pressure + ejection_fraction
+
##     diabetes + creatinine_phosphokinase + anaemia + age, family =
"binomial",
##     data = train2)
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               4.953e+00  5.415e+00   0.915 0.360352
## smoking                   5.172e-01  4.096e-01   1.263 0.206647
## sex                      -5.400e-01  3.994e-01  -1.352 0.176422
## serum_sodium             -6.513e-02  3.936e-02  -1.655 0.098026 .
## serum_creatinine          8.014e-01  2.271e-01   3.529 0.000417 ***
## platelets                 1.265e-07  1.775e-06   0.071 0.943219
## high_blood_pressure       4.682e-01  3.485e-01   1.344 0.179090
## ejection_fraction        -6.222e-02  1.632e-02  -3.814 0.000137 ***
## diabetes                  2.197e-01  3.368e-01   0.652 0.514170
## creatinine_phosphokinase  3.087e-04  1.601e-04   1.928 0.053854 .
## anaemia                   4.807e-01  3.437e-01   1.399 0.161914
## age                       6.086e-02  1.492e-02   4.079 4.52e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 295.79  on 238  degrees of freedom
## Residual deviance: 228.49  on 227  degrees of freedom
## AIC: 252.49
##
## Number of Fisher Scoring iterations: 5
```

As can be seen by the significance of each explanatory variable, Serum Creatinine levels, age, and ejection fraction are the three most important variables in this model for predicting the death event. It is also important to note that Creatinine Phosphokinase levels also was just barley not significant in this logistic regression model. We expected these variables to all be the most significant because of there positioning in the decision tree. It is important to look at the accuracy of this model.

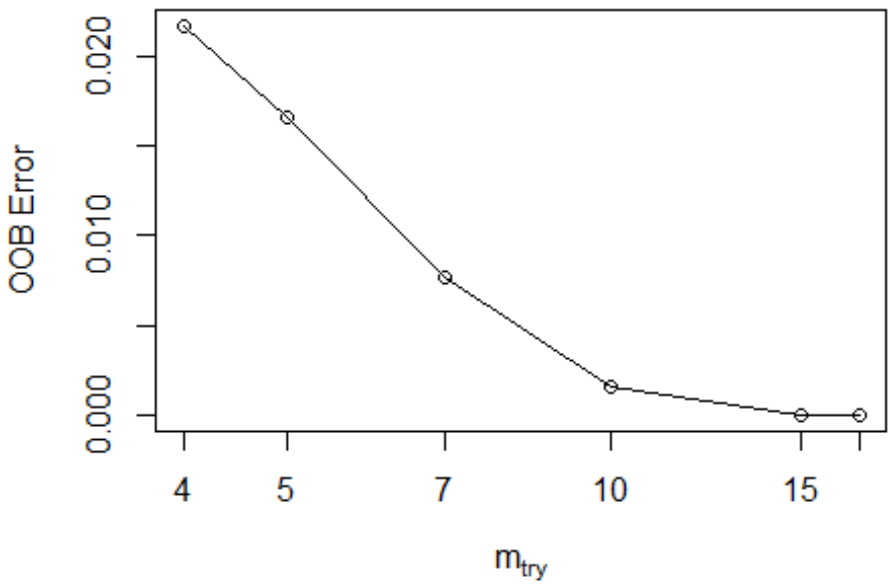
```
holdTest = test2
holdTest$pred_death = predict(regressionModel, holdTest)
holdTest$prob = exp(holdTest$pred_death)/(1+exp(holdTest$pred_death))
```

```
holdTest$predOut = ifelse(holdTest$prob > 0.5, 1, 0)
table(holdTest$predOut, holdTest$DEATH_EVENT)

##
##      0  1
##   0 32 14
##   1  6  8
```

As we can see from the confusion matrix above, the model has about a 67% accuracy. This is a slight decrease from what we got from our original logistic regression model. This is a surprise considering that we only added more variables and three more variables that were significant predictors. This may be due to the fact that are validation data set is so small that results can be very variable. This model has a true positive rate of about 0.57 and true negative rate of about 0.70. This model ended up not being very impressive at predicting the death event.

Below we will construct a Random Forrest model.

```
mtry <- tuneRF(test2[-1],test2$DEATH_EVENT, ntreeTry=1000,
               stepFactor=1.5,improve=0.01, trace=TRUE, plot=TRUE)

## mtry = 5  OOB error = 0.01661305
## Searching left ...
## mtry = 4     OOB error = 0.02170601
## -0.306564 0.01
## Searching right ...
## mtry = 7     OOB error = 0.007651123
## 0.5394509 0.01
## mtry = 10    OOB error = 0.001519266
## 0.8014323 0.01
## mtry = 15    OOB error = 3.306102e-05
## 0.9782388 0.01
## mtry = 17    OOB error = 7.265344e-07
## 0.9780244 0.01
```




```
best.m <- mtry[mtry[, 2] == min(mtry[, 2]), 1]
print(mtry)

##    mtry     OOBError
## 4     4 2.170601e-02
## 5     5 1.661305e-02
## 7     7 7.651123e-03
## 10   10 1.519266e-03
## 15   15 3.306102e-05
## 17   17 7.265344e-07

print(best.m)

## [1] 17
```

The first thing we did when creating our Random Forrest model was to tune the data. We did this by looking for the most optimal mtyr value. We did this my comparing the mtyr value to the OOBError. When comparing the mtyr value to the OOBError we found that mtyr 12 was the most optimal.

```
form_full<-
as.formula(DEATH_EVENT~age+anaemia+creatinine_phosphokinase+diabetes+ejection
_fraction+high_blood_pressure+platelets+serum_creatinine+serum_sodium+sex+smo
king)
mod_rf <- randomForest(formula = form_full,data=train2,ntree = 1000,mtry =
12)
mod_rf
```

```
##
## Call:
##  randomForest(formula = form_full, data = train2, ntree = 1000,      mtry
= 12)
##                Type of random forest: regression
##                      Number of trees: 1000
## No. of variables tried at each split: 11
##
##           Mean of squared residuals: 0.1748895
##                     % Var explained: 18.18
```

Using the most optimal mtyr that we found above, we then ran the random forrest model. This told us that the mean square of residuals was 0.18 and the percent variance explained is 18.98

```
library(tibble)
importance(mod_rf)%>%
  as.data.frame()%>%
  rownames_to_column()%>%
  arrange(desc(IncNodePurity))
```

```
##                     rowname IncNodePurity
## 1          serum_creatinine    13.2651994
## 2         ejection_fraction     8.7939057
## 3                       age     6.6842236
## 4  creatinine_phosphokinase     6.1783611
## 5                 platelets     5.6753250
## 6              serum_sodium     3.4237414
## 7                       sex     0.9085800
## 8                  diabetes     0.7222326
## 9       high_blood_pressure     0.5878404
## 10                 anaemia     0.5469382
## 11                 smoking     0.4388638
```

```
table(train2$DEATH_EVENT)/length(train2$DEATH_EVENT)
```

```
##
##         0         1
## 0.6903766 0.3096234
```

After running the random forest model we looked at the accuracy and error rate of the data. We found that the data had a 69% accuracy rate and a 33% error rate. We see that this model is not balanced and we can bet that the error rate a person will have a death event is 31%

```
predictors <- model.matrix(form_full, data = train2)
cv.fit <- cv.glmnet(predictors, train2$DEATH_EVENT, family = "binomial", type
= "class")
lambda_opt=cv.fit$lambda.1se
mod_lr2 <- glmnet(predictors, train2$DEATH_EVENT, family = "binomial", lambda
= lambda_opt)
```

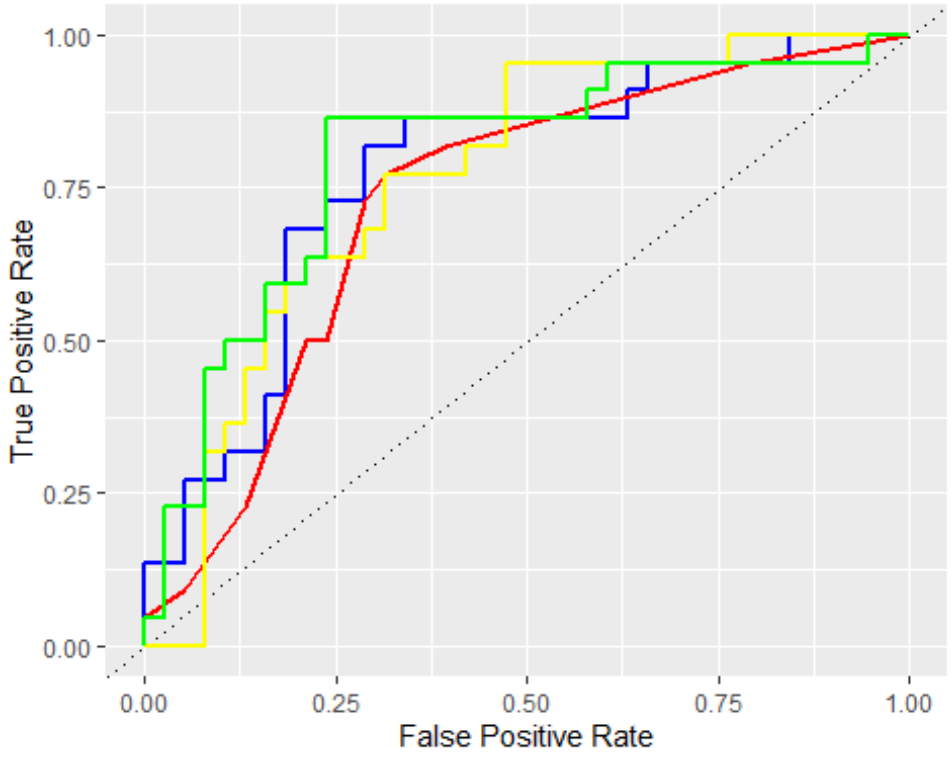
```
y_lr = predict(mod_lr2, newx = model.matrix(form_full, data = test2), type =
"class")
confusion_lr = table(test2$DEATH_EVENT, y_lr)
confusion_lr

##    y_lr
##      0  1
##   0 35  3
##   1 15  7

tpr_lr = confusion_lr[2,2]/sum(confusion_lr[2,]); tpr_lr

## [1] 0.3181818

tnr_lr = confusion_lr[1,1]/sum(confusion_lr[1,]); tnr_lr

## [1] 0.9210526
```

We also looked at the true positive and negative rate of the model and found that the model had a true positive rate of 0.32 and a true negative rate of 0.92.

To summarize all these results, we generated ROC curves for each model.

```
library(ROCR)

## Warning: package 'ROCR' was built under R version 4.3.2

roc_data <- function(test,y_test,model,type){
  prob = model %>%
    predict(newdata=test, type=type) %>%
    as.data.frame()
  if (type == "raw") {
  pred_prob = prediction(prob[,2], y_test)
  }
  else {
  pred_prob = prediction(prob[,1], y_test)
  }
  perf = performance(pred_prob, 'tpr', 'fpr')
  perf_df = data.frame(perf@x.values, perf@y.values)
  names(perf_df)=c('fpr','tpr')
  return(perf_df)
}

roc_data_lr <- function(test,y_test,model,type, newx){
  prob = model %>%
    predict(newdata=test, newx=newx, type=type) %>%
    as.data.frame()
  pred_prob = prediction(prob[,1], y_test)
  perf = performance(pred_prob, 'tpr', 'fpr')
  perf_df = data.frame(perf@x.values, perf@y.values)
  names(perf_df)=c('fpr','tpr')
  return(perf_df)
```

```
}

perf_df_dt = roc_data(test2, test2$DEATH_EVENT, mod_tree, "matrix")
perf_df_rf = roc_data(test2, test2$DEATH_EVENT, mod_rf, "response")
perf_df_nb = roc_data(test2, test2$DEATH_EVENT, mod_nb, "raw")
perf_df_lr = roc_data_lr(test2, test2$DEATH_EVENT, mod_lr2, "response",newx =
model.matrix(form_full, data = test2))



p <- ggplot(data =perf_df_nb, aes(x=fpr, y=tpr))+
  geom_line(color="blue",linewidth=1)+
  geom_abline(intercept=0,slope=1,lty=3)+
  geom_line(data =perf_df_dt,color="red",linewidth=1) +
  geom_line(data =perf_df_rf,color="yellow",linewidth=1) +
  geom_line(data =perf_df_lr,color="green",linewidth=1) +
  labs(x='False Positive Rate', y='True Positive Rate')

p
```



Above are a blue, red, yellow and green ROC curve corresponding respectively to the Naive Bayes, Decision Tree, Random Forest and Logistic Regression models. As we can see, there is no significant difference in their performances but we can tell for instance that the Decision Tree model performed most poorly as the area below the curve (AUC) seems to be the smallest, which would indicate a lower accuracy. It is hard to compare the AUCs of the other curves as they look almost equal, but this is expected as their accuracies were almost

the same with 67% for logistic regression, 68% for Naive Bayes and 69% for Random Forest.

Conclusion

Our goal in this study was to first look at how age, number of blood platelets and whether a person smokes affects someone's chance of living after having experienced heart failure. Then, to develop various machine learning models in order to create the best possible model to predict the death event. We have concluded that whether someone smokes and their number of blood platelets are not correlated with whether someone dies after heart failure. But, age does have a strong correlation at predicting the death event. On the other hand, after analyzing all four of the machine learning models that we evaluated for performance, we can conclude that the Random Forest is the best model for predicting the death even of someone that has heart failure. We also learned that ejection fraction and serum creatinine were both very important predictors in all of our models. None of the models that were developed were extremely compelling and accurate, so it would be important for further research to be done on confounding variables in order to create a more accurate model.