Classification

Francisco Trejo & Diego Ochoa

Classification

Linear Models for classification are able to make predictions like regression but Classification classifies the data either by positive or negative relative to the linear line passing through the data. Unlike regression that is quantitative, Classification has to be qualitative. Some strengths to these linear models are that it gives out straightforward probabilistic interpretations of the data and it's not computationally expensive. A weakness to these linear models is that it can't handle complex relationships so it tends to underfit.

Link to CSV file

https://www.kaggle.com/datasets/rsrishav/patient-survival-after-one-year-of-treatment (https://www.kaggle.com/datasets/rsrishav/patient-survival-after-one-year-of-treatment)

Read File

```
df <- read.csv("C:/Users/Diego/Downloads/Training_set_advance.csv", header=TRUE)
```

Clean Data and convert factors

```
df <- df[,c(2,5,6,8,17,18)]
df <- df[complete.cases(df[, 1:6]),]
df$Diagnosed_Condition <- factor(df$Diagnosed_Condition)
df$Survived_1_year <- factor(df$Survived_1_year)
#df$Patient_Smoker <- factor(df$Patient_Smoker)
df$Patient_Rural_Urban <- factor(df$Patient_Rural_Urban)
sapply(df, function(x) sum(is.na(x)==TRUE))
```

```
##     Diagnosed_Condition             Patient_Age Patient_Body_Mass_Index
##                       0                       0                       0
##     Patient_Rural_Urban     Number_of_prev_cond         Survived_1_year
##                       0                       0                       0
```

Split to Train/Test and build Logistic Regression

Summary of Logistic Regression is different for Linear Regression. The coefficient tells us the different between the target variable which is surviving after 1 year of treatment and the predictor but in log odds. The negative tells us for example that with the increase of age the logs odds of surviving 1 year after treatment decreases. The Std. Error also seems to be a bit high on our different diagnosed conditions which questions the accuracy of these coefficients. The residual deviance is lower than the Null deviance which comparing the lack of the fit of the overall model and intercept which is has to be lower. The AIC is also pretty high but useful when comparing other models

with lower AIC. Which is probably high because of the amount of predictors we have. The P value shows us what we knew beforehand about the diagnosed condition which may not be a good predictor because they are really close to 1. However, the last 4 predictors seem to be strong with low Std. Error and P value.

```
set.seed(1234)
i <- sample(1:nrow(df), 0.8*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
glm1 <- glm(Survived_1_year~., data=train, family="binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = Survived_1_year ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1201  -1.1126   0.6830   0.8909   1.8985
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               13.224465  80.270801   0.165    0.869
## Diagnosed_Condition1      -9.831279  80.270901  -0.122    0.903
## Diagnosed_Condition2      -9.929866  80.270914  -0.124    0.902
## Diagnosed_Condition3      -9.766818  80.270915  -0.122    0.903
## Diagnosed_Condition4      -9.689218  80.270905  -0.121    0.904
## Diagnosed_Condition5      -9.796074  80.270903  -0.122    0.903
## Diagnosed_Condition6      -9.679172  80.270904  -0.121    0.904
## Diagnosed_Condition7      -9.865398  80.270900  -0.123    0.902
## Diagnosed_Condition8      -9.742558  80.270902  -0.121    0.903
## Diagnosed_Condition9      -9.601421  80.270922  -0.120    0.905
## Diagnosed_Condition10     -9.796962  80.270913  -0.122    0.903
## Diagnosed_Condition11     -9.897949  80.270898  -0.123    0.902
## Diagnosed_Condition12    -10.006774  80.270898  -0.125    0.901
## Diagnosed_Condition13     -9.672890  80.270910  -0.121    0.904
## Diagnosed_Condition14     -9.889451  80.270904  -0.123    0.902
## Diagnosed_Condition15     -9.781576  80.270904  -0.122    0.903
## Diagnosed_Condition16     -9.867493  80.270903  -0.123    0.902
## Diagnosed_Condition17     -9.811811  80.270895  -0.122    0.903
## Diagnosed_Condition18     -9.818058  80.270910  -0.122    0.903
## Diagnosed_Condition19     -9.965227  80.270899  -0.124    0.901
## Diagnosed_Condition20    -11.102028  80.270896  -0.138    0.890
## Diagnosed_Condition21    -10.990324  80.270896  -0.137    0.891
## Diagnosed_Condition22    -11.189381  80.270903  -0.139    0.889
## Diagnosed_Condition23    -11.098948  80.270898  -0.138    0.890
## Diagnosed_Condition24    -11.148133  80.270892  -0.139    0.890
## Diagnosed_Condition25    -11.147863  80.270895  -0.139    0.890
## Diagnosed_Condition26    -11.162808  80.270897  -0.139    0.889
## Diagnosed_Condition27    -11.012439  80.270892  -0.137    0.891
## Diagnosed_Condition28    -11.124710  80.270896  -0.139    0.890
## Diagnosed_Condition29    -11.010616  80.270892  -0.137    0.891
## Diagnosed_Condition30    -11.063578  80.270889  -0.138    0.890
## Diagnosed_Condition31    -11.238423  80.270897  -0.140    0.889
## Diagnosed_Condition32    -11.005659  80.270902  -0.137    0.891
## Diagnosed_Condition33    -10.877716  80.270892  -0.136    0.892
## Diagnosed_Condition34    -10.817687  80.270888  -0.135    0.893
## Diagnosed_Condition35    -10.912538  80.270891  -0.136    0.892
## Diagnosed_Condition36     -9.885983  80.270902  -0.123    0.902
## Diagnosed_Condition37     -9.651059  80.270917  -0.120    0.904
## Diagnosed_Condition38     -9.780608  80.270916  -0.122    0.903
## Diagnosed_Condition39     -9.740073  80.270911  -0.121    0.903
## Diagnosed_Condition40     -9.706392  80.270908  -0.121    0.904
## Diagnosed_Condition41     -9.783157  80.270911  -0.122    0.903
```

```
## Diagnosed_Condition42     -9.742143  80.270912  -0.121    0.903
## Diagnosed_Condition43     -9.650685  80.270912  -0.120    0.904
## Diagnosed_Condition44     -9.692100  80.270920  -0.121    0.904
## Diagnosed_Condition45     -9.780745  80.270905  -0.122    0.903
## Diagnosed_Condition46     -9.667140  80.270913  -0.120    0.904
## Diagnosed_Condition47     -9.737454  80.270910  -0.121    0.903
## Diagnosed_Condition48     -9.708970  80.270913  -0.121    0.904
## Diagnosed_Condition49     -9.710892  80.270905  -0.121    0.904
## Diagnosed_Condition50     -9.584870  80.270924  -0.119    0.905
## Diagnosed_Condition51     -9.829044  80.270901  -0.122    0.903
## Diagnosed_Condition52    -10.042759  80.270901  -0.125    0.900
## Patient_Age               -0.009821   0.000827 -11.876   <2e-16 ***
## Patient_Body_Mass_Index   -0.064228   0.004260 -15.076   <2e-16 ***
## Patient_Rural_UrbanURBAN  -0.572088   0.034262 -16.697   <2e-16 ***
## Number_of_prev_cond       -0.282002   0.020570 -13.709   <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: 25114  on 18977  degrees of freedom
## Residual deviance: 22844  on 18921  degrees of freedom
## AIC: 22958
##
## Number of Fisher Scoring iterations: 10
```

5 R function data exploration and Graphs with training data

```
sprintf("The average patient age addimited is %#.2f and the standard diviation is %#.2f", mean(t
rain$Patient_Age), sd(train$Patient_Age))
```

```
## [1] "The average patient age addimited is 33.31 and the standard diviation is 19.48"
```

```
sprintf("The average patient's body mass index is  %#.2f and the standard diviation is %#.2f", m
ean(train$Patient_Body_Mass_Index), sd(train$Patient_Body_Mass_Index))
```

```
## [1] "The average patient's body mass index is  23.42 and the standard diviation is 3.79"
```

```
sprintf("Columns of data")
```

```
## [1] "Columns of data"
```

```
names(train)
```

```
## [1] "Diagnosed_Condition"     "Patient_Age"
## [3] "Patient_Body_Mass_Index" "Patient_Rural_Urban"
## [5] "Number_of_prev_cond"     "Survived_1_year"
```

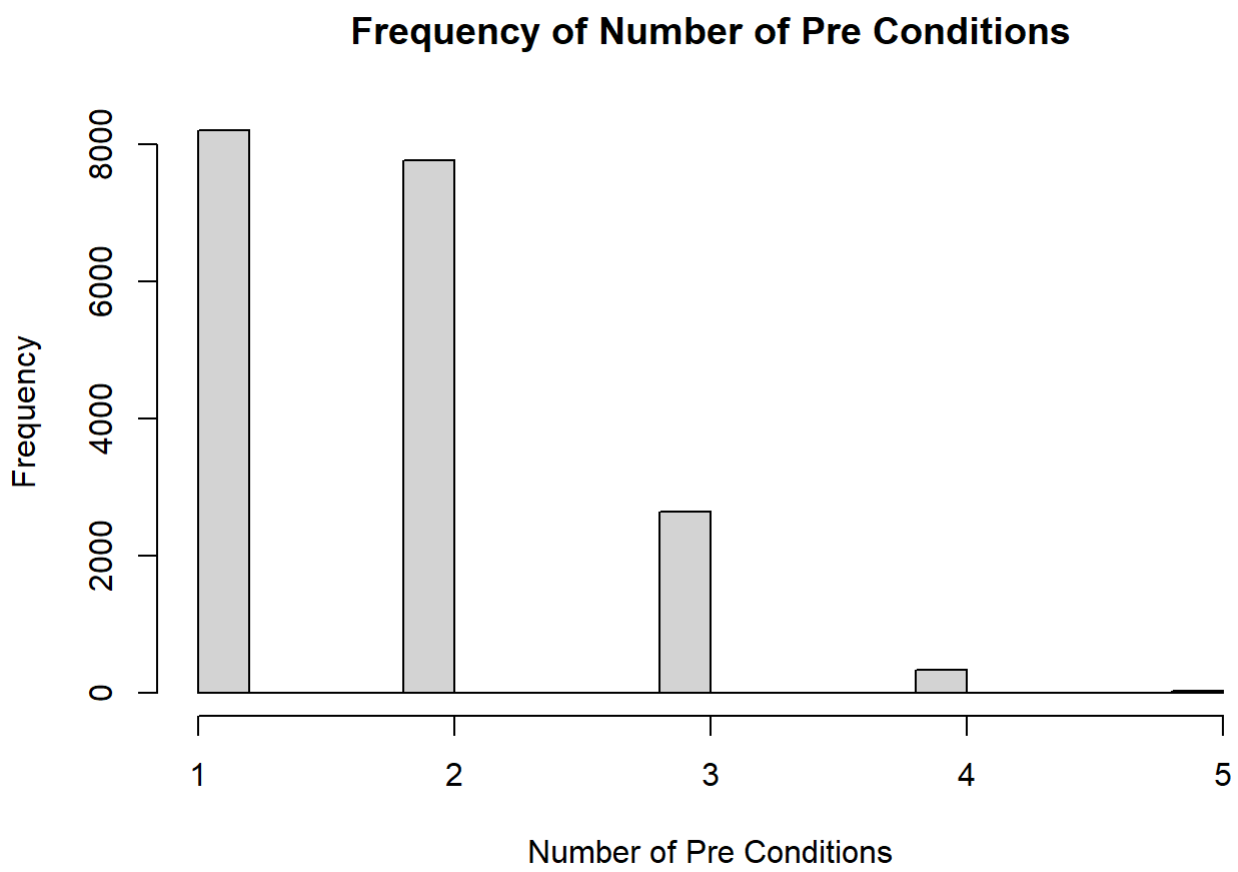
```
sprintf(" Amount of people who didn't survive vs the amount that did")
```

```
## [1] " Amount of people who didn't survive vs the amount that did"
```

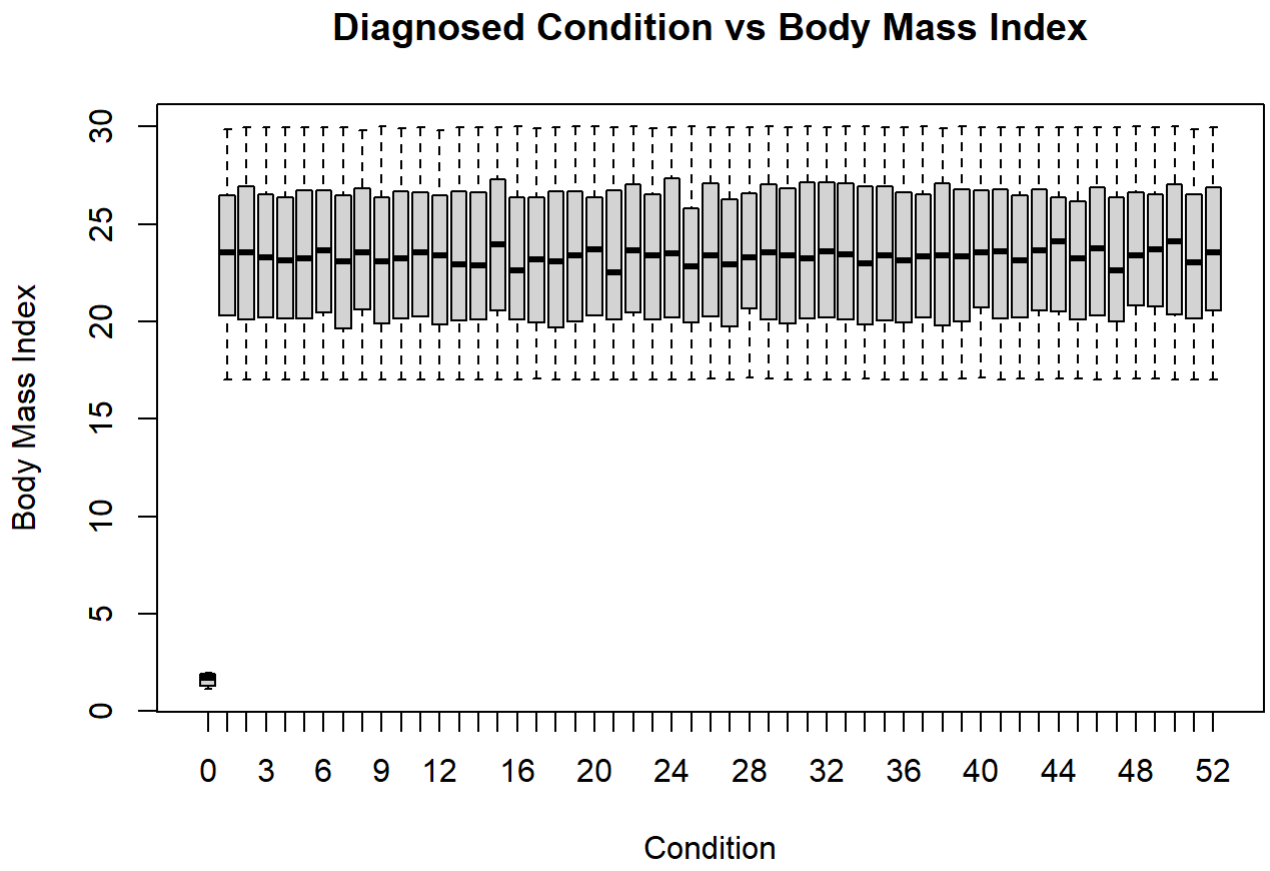
```
summary(train$Survived_1_year)
```

```
##     0     1
##  7120 11858
```

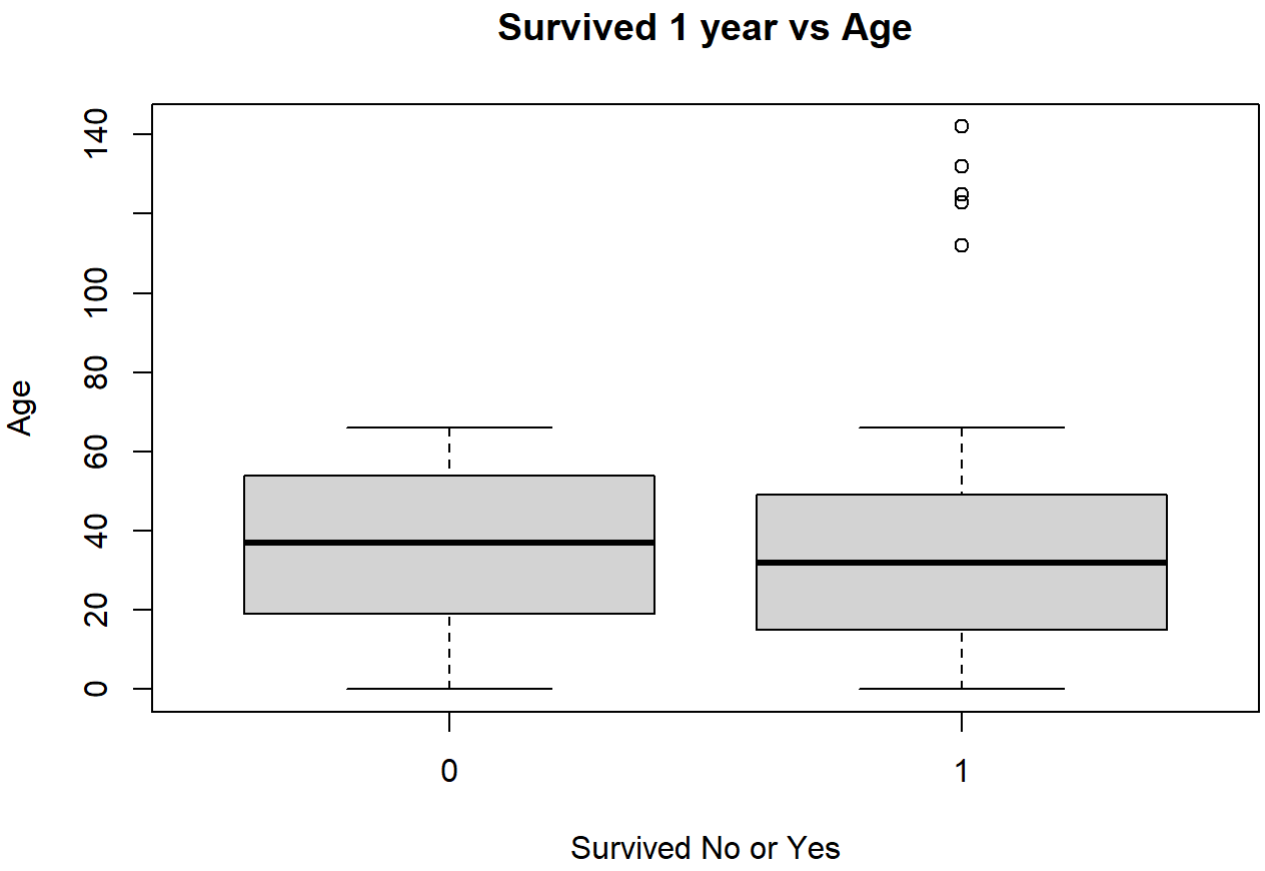
```
hist(train$Number_of_prev_cond, main = "Frequency of Number of Pre Conditions", xlab = "Number o
f Pre Conditions")
```




```
#Graphs
plot(train$Diagnosed_Condition, train$Patient_Body_Mass_Index, main = "Diagnosed Condition vs Bo
dy Mass Index", xlab="Condition", ylab="Body Mass Index")
```




```
plot(train$Survived_1_year, train$Patient_Age , main = "Survived 1 year vs Age", xlab="Survived
No or Yes", ylab="Age")
```



Bayes Model

The Bayes models tells us first the probability of surviving the 1 year treatment which is 62%. Then goes and tells us the conditional probability of each predictor. For the qualitative predictors all the probabilities from the row equal 1 when added. So there is 2.4% probability you would survive 1 year after the treatment if you were diagnosed with Condition 49 compared to the other conditions. The quantitative predictors don't do that but it gives you the average and standard deviation. So the average age of the people who survived 1 year after treatment was 32.

```
library(e1071)
nb1 <- naiveBayes(train$Survived_1_year~., data=train )
nb1
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.3751713 0.6248287
##
## Conditional probabilities:
##    Diagnosed_Condition
## Y              0            1            2            3            4
##   0 0.0000000000 0.0162921348 0.0143258427 0.0139044944 0.0150280899
##   1 0.0005059875 0.0236127509 0.0193961882 0.0202395008 0.0237814134
##    Diagnosed_Condition
## Y              5            6            7            8            9
##   0 0.0153089888 0.0151685393 0.0164325843 0.0162921348 0.0120786517
##   1 0.0218417946 0.0251307134 0.0223477821 0.0246247259 0.0216731321
##    Diagnosed_Condition
## Y             10           11           12           13           14
##   0 0.0139044944 0.0175561798 0.0175561798 0.0134831461 0.0154494382
##   1 0.0208298195 0.0231067634 0.0212514758 0.0227694384 0.0212514758
##    Diagnosed_Condition
## Y             15           16           17           18           19
##   0 0.0155898876 0.0153089888 0.0168539326 0.0146067416 0.0174157303
##   1 0.0231910946 0.0220947883 0.0236127509 0.0207454883 0.0216731321
##    Diagnosed_Condition
## Y             20           21           22           23           24
##   0 0.0292134831 0.0269662921 0.0285112360 0.0279494382 0.0314606742
##   1 0.0121437005 0.0124810255 0.0108787317 0.0120593692 0.0129026817
##    Diagnosed_Condition
## Y             25           26           27           28           29
##   0 0.0297752809 0.0294943820 0.0279494382 0.0290730337 0.0299157303
##   1 0.0118907067 0.0118907067 0.0133243380 0.0123123630 0.0134086693
##    Diagnosed_Condition
## Y             30           31           32           33           34
##   0 0.0314606742 0.0310393258 0.0261235955 0.0269662921 0.0280898876
##   1 0.0137459943 0.0113003879 0.0113003879 0.0139989880 0.0154326193
##    Diagnosed_Condition
## Y             35           36           37           38           39
##   0 0.0285112360 0.0160112360 0.0130617978 0.0134831461 0.0144662921
##   1 0.0146736381 0.0211671445 0.0219261258 0.0199865070 0.0220947883
##    Diagnosed_Condition
## Y             40           41           42           43           44
##   0 0.0144662921 0.0144662921 0.0137640449 0.0139044944 0.0126404494
##   1 0.0232754259 0.0210828133 0.0212514758 0.0237814134 0.0207454883
##    Diagnosed_Condition
## Y             45           46           47           48           49
##   0 0.0151685393 0.0141853933 0.0139044944 0.0136235955 0.0154494382
##   1 0.0230224321 0.0227694384 0.0211671445 0.0212514758 0.0242030697
```

```
##    Diagnosed_Condition
## Y             50           51           52
##   0 0.0123595506 0.0161516854 0.0178370787
##   1 0.0221791196 0.0227694384 0.0199021757
##
##    Patient_Age
## Y       [,1]     [,2]
##   0 35.36699 20.30753
##   1 32.07531 18.86456
##
##    Patient_Body_Mass_Index
## Y       [,1]     [,2]
##   0 23.93387 4.266420
##   1 23.10432 3.434447
##
##    Patient_Rural_Urban
## Y       RURAL     URBAN
##   0 0.6296348 0.3703652
##   1 0.7410187 0.2589813
##
##    Number_of_prev_cond
## Y       [,1]      [,2]
##   0 1.839466 0.8324698
##   1 1.694805 0.7295322
```

Prediting and Evaluating Test Data

We had a slightly higher accuracy for Bayes than we did for Logistical. The reason for that is because Bayes is more generative and the amount of predictors that were factors may have over whelmed the Logistical Regression. The high value of P in these almost 50 dummy predictors may have benefited the Bayes Model.

Logistical

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 1, 0)
acc <- mean(pred==test$Survived_1_year)
print(paste("accuracy = ", acc))
```

```
## [1] "accuracy =  0.675237091675448"
```

```
table(pred, test$Survived_1_year)
```

```
##
## pred    0    1
##    0  705  496
##    1 1045 2499
```

Confusion Matrix

```
library(caret)
```

```
## Loading required package: ggplot2
```

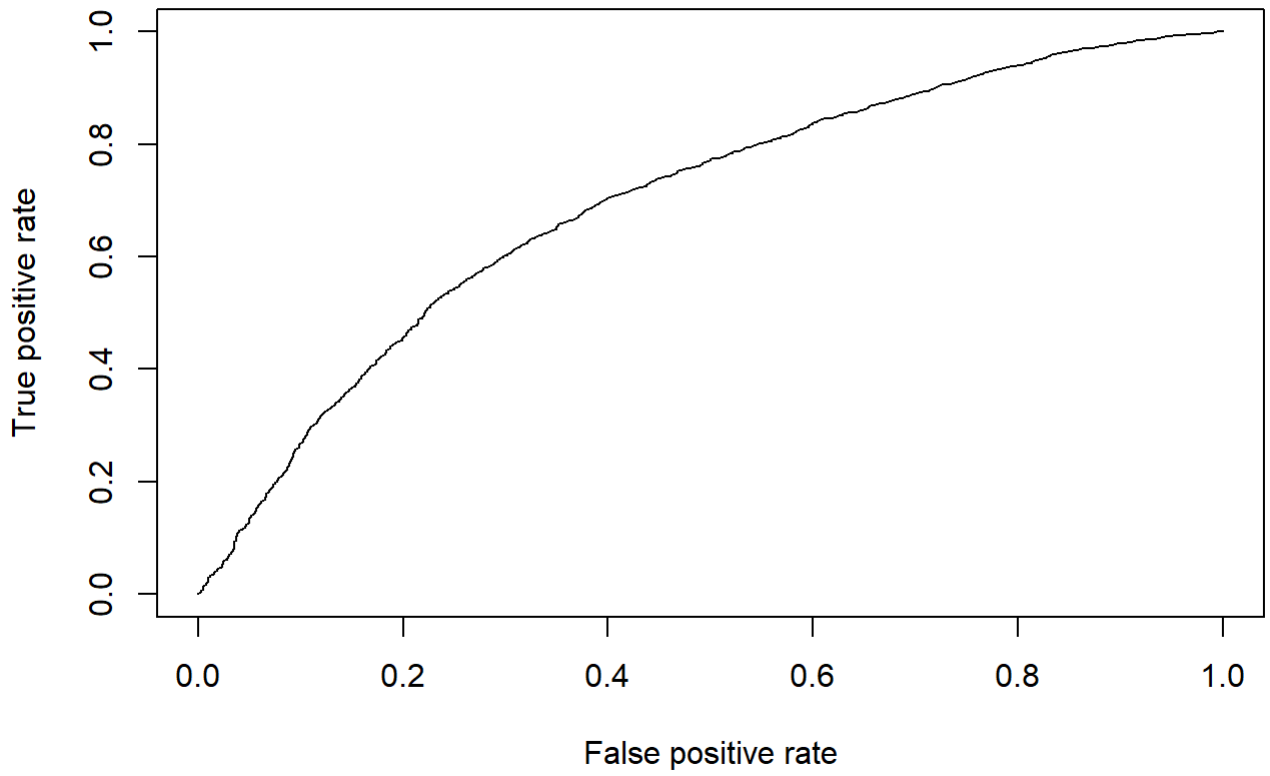
```
## Loading required package: lattice
```

```
confusionMatrix(as.factor(pred), reference=test$Survived_1_year)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0  705  496
##          1 1045 2499
##
##                Accuracy : 0.6752
##                  95% CI : (0.6617, 0.6886)
##     No Information Rate : 0.6312
##     P-Value [Acc > NIR] : 1.23e-10
##
##                   Kappa : 0.2538
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.4029
##             Specificity : 0.8344
##          Pos Pred Value : 0.5870
##          Neg Pred Value : 0.7051
##              Prevalence : 0.3688
##          Detection Rate : 0.1486
##    Detection Prevalence : 0.2531
##       Balanced Accuracy : 0.6186
##
##        'Positive' Class : 0
##
```

ROC

```
library(ROCR)
p <- predict(glm1, newdata=test, type="response")
pr <- prediction(p, test$Survived_1_year)
prf <- performance(pr, measure = "tpr", x.measure = "fpr")
plot(prf)
```




```
auc <- performance(pr, measure = "auc")
auc <- auc@y.values[[1]]
auc
```

```
## [1] 0.6963787
```

Bayes

```
p1 <- predict(nb1, newdata=test, type="class")
table(p1, test$Survived_1_year)
```

```
##
## p1     0    1
##   0  741  417
##   1 1009 2578
```

```
mean(p1==test$Survived_1_year)
```

```
## [1] 0.6994731
```

Strength and Weaknesses

Both Logistical and Bayes have strength and weaknesses. When it comes to data size Bayes tends to work better with short data and Logistical with larger sets of data. However, both do end up converging when the training data heads to infinity. Bayes also runs quickly and works good with high dimensions because it assumes it's independent. The independence can also be double edge sword and affect performance because of it. Another done side is that it guesses on data in the testing set that wasn't seen in the training set. Logistical is computationally inexpensive and binary classification is handled pretty well if they are linearly separable. It also nicely outputs probability. The down side it can under fit because of complex non linear boundaries.

Metrics

One metric is accuracy which tells you how accurate the model correctly predicted. Then there is sensitivity that measures true positive rate and specificity that measures the true negative rate. This shows us how many were misclassified from the classes and it is drawn as a matrix. Then you have Kappa that tries to adjust the accuracy by accounting for the possibility of a correct prediction by chance alone and it's easily calculated. It's the measure of how two qualitative predictors may agree with each other, but that draw back is there is no universal agreement on the interpretation of it. Then there is ROC which is a visualization of the how the machine learning algorithm performed. It helps you see the relationship between true positive and false positive. AUC is the area under that curve. This tells you how well the model was able to distinguish the classes and it gives you a metric between 0 and 1 to evaluate the model.