Classification Methods on Airline Delay Data

Load Data

This data set had 500,000+ rows so I reduced it down to roughly 10,000 rows chosen randomly so it doesn't take as long to run. Airlines delay data set found here: https://www.kaggle.com/datasets/ulrikthygepedersen/airlines-delay (https://www.kaggle.com/datasets/ulrikthygepedersen/airlines-delay)

```
df <- read.csv("airlines.csv")
str(df)
```

```
## 'data.frame':    539382 obs. of  8 variables:
##  $ Flight     : num  2313 6948 1247 31 563 ...
##  $ Time       : num  1296 360 1170 1410 692 ...
##  $ Length     : num  141 146 143 344 98 60 239 80 105 108 ...
##  $ Airline    : chr  "DL" "OO" "B6" "US" ...
##  $ AirportFrom: chr  "ATL" "COS" "BOS" "OGG" ...
##  $ AirportTo  : chr  "HOU" "ORD" "CLT" "PHX" ...
##  $ DayOfWeek  : int  1 4 3 6 4 4 4 3 7 3 ...
##  $ Class      : int  0 0 0 0 0 0 0 0 0 0 ...
```

```
i <- sample(1:nrow(df), 0.98*nrow(df), replace=FALSE)
df <- df[-i,]
```

Data Exploration

We are basing our classification on the time of the flight, length of the flight, and the day of the week the flight is on. Row 8 (Class) is our target.

```
df <- df[,c(2, 3, 7, 8)]
df$DayOfWeek <- factor(df$DayOfWeek)
df$Class <- factor(df$Class)
```

Divide into 80/20 train/test.

```
set.seed(1234)
i <- sample(1:nrow(df), 0.80*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

Data Exploration. Class is whether the flight was canceled, with 1 being canceled. DayOfWeek is the day of the week the flight is on, with 1 being Monday. There aren't any N/As so no data manipulation is needed.

```
sapply(train, function(x) sum(is.na(x) == TRUE))
```

```
##      Time    Length DayOfWeek     Class
##         0         0         0         0
```

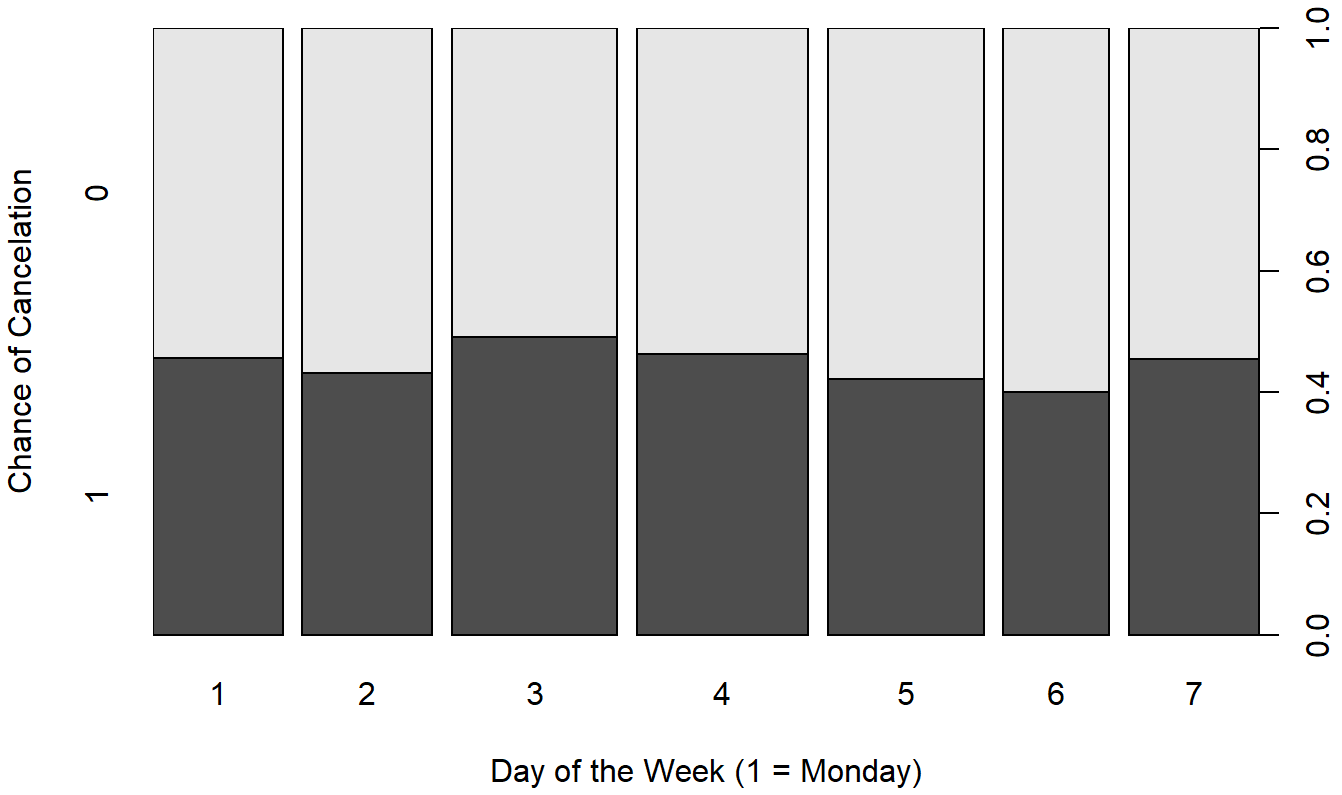
```
summary(train)
```

```
##       Time           Length      DayOfWeek Class
##  Min.   :  30.0   Min.   :  0.0   1:1134    0:4765
##  1st Qu.: 555.0   1st Qu.: 81.0   2:1136    1:3865
##  Median : 790.0   Median :114.0   3:1443
##  Mean   : 798.2   Mean   :131.2   4:1494
##  3rd Qu.:1028.0   3rd Qu.:160.0   5:1363
##  Max.   :1439.0   Max.   :535.0   6: 926
##                                   7:1134
```

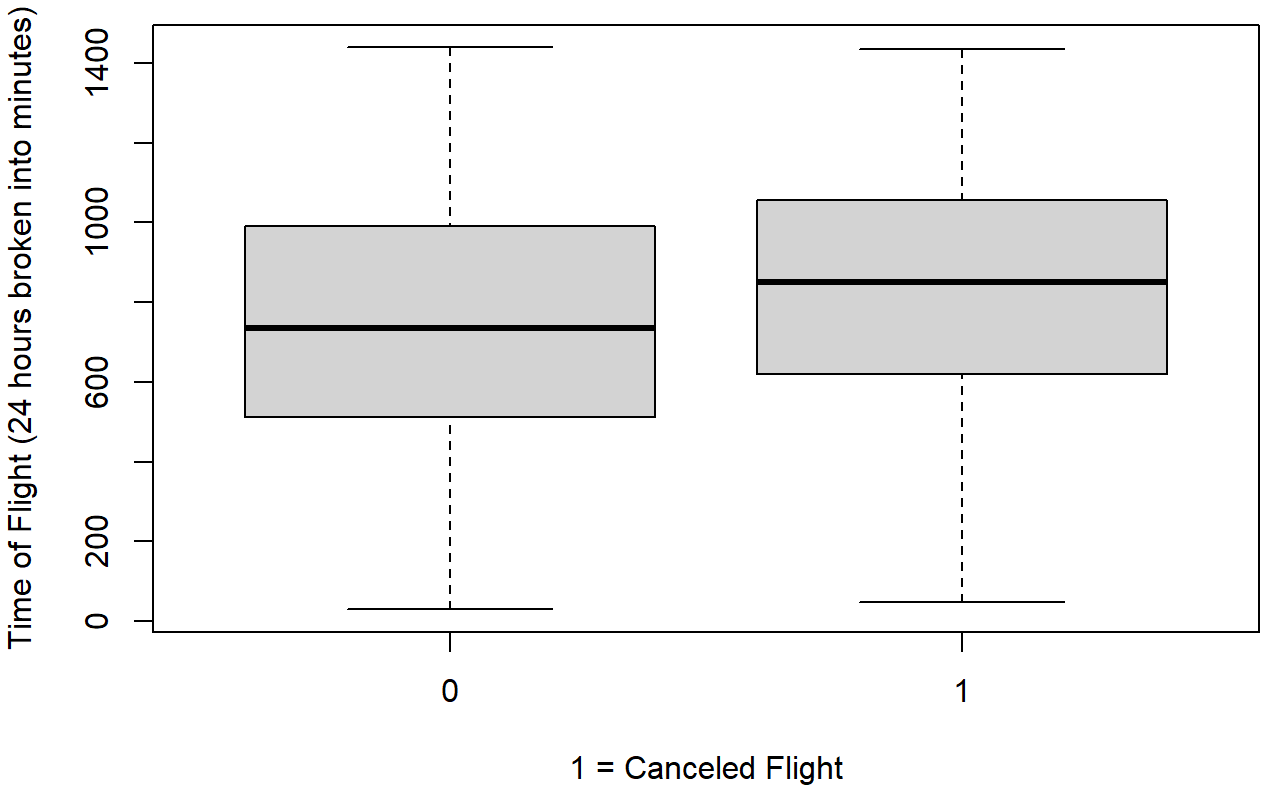
```
table(train$Class)
```

```
##
##    0    1
## 4765 3865
```

```
plot(train$DayOfWeek, train$Class, xlab = "Day of the Week (1 = Monday)", ylab = "Chance of C
ancelation")
```




```
plot(train$Class, train$Time, xlab = "1 = Canceled Flight", ylab = "Time of Flight (24 hours
broken into minutes)")
```



Logistic Regression Model

```
glm1 <- glm(Class~., data=train, family="binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = Class ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5318  -1.0761  -0.9129   1.2213   1.6539
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -1.168e+00  9.738e-02 -11.990  < 2e-16 ***
## Time         1.005e-03  7.909e-05  12.711  < 2e-16 ***
## Length       1.467e-03  3.172e-04   4.624 3.76e-06 ***
## DayOfWeek2  -1.099e-01  8.544e-02  -1.286   0.1984
## DayOfWeek3   1.339e-01  8.043e-02   1.664   0.0961 .
## DayOfWeek4   6.103e-03  7.992e-02   0.076   0.9391
## DayOfWeek5  -1.489e-01  8.188e-02  -1.819   0.0689 .
## DayOfWeek6  -2.240e-01  9.073e-02  -2.469   0.0136 *
## DayOfWeek7  -2.104e-02  8.525e-02  -0.247   0.8051
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 11870  on 8629  degrees of freedom
## Residual deviance: 11660  on 8621  degrees of freedom
## AIC: 11678
##
## Number of Fisher Scoring iterations: 4
```

Above we have a summary of glm1, our simple logistic regression model for predicting whether a flight will be canceled or not given our predictors. Under the coefficients table we have our predictors with some values. The coefficient shows how the likelihood of a flight being canceled changes per unit (i.e. the coefficient of x in the logistic equation). The next column, Std. Error, represents the average distance that our observed values stray from our plotted line. Pr(>|z|) is our p-value associated with the z value column right before it. This shows us how statistically significant that predictor is (with the response variable z). As you can see the days of the week vary a lot, some by several orders of magnitude. We can see that our most significant predictor is the time a flight takes place.

```
probs <- predict(glm1, newdata=test, type="response")
pred <- ifelse(probs>0.5, 1, 0)
t1 <- table(pred, test$Class)
table(pred, test$Class)
```

```
##
## pred   0   1
##    0 930 639
##    1 279 310
```

```
acc <- mean(pred==test$Class)
print(paste("accuracy ", acc))
```

```
## [1] "accuracy  0.574606116774791"
```

```
library(psych)

print(paste("precision ", t1[1,1]/(t1[1,1]+t1[1,2])))
```

```
## [1] "precision  0.592734225621415"
```

```
print(paste("sensitivity ", t1[1,1]/(t1[1,1]+t1[2,1])))
```

```
## [1] "sensitivity  0.769230769230769"
```

```
print(paste("specificty ", t1[2,2]/(t1[2,2]+t1[1,2])))
```

```
## [1] "specificty  0.326659641728135"
```

```
cohen.kappa(t1)
```

```
## Call: cohen.kappa1(x = x, w = w, n.obs = n.obs, alpha = alpha, levels = levels)
##
## Cohen Kappa and Weighted Kappa correlation coefficients and confidence boundaries
##                  lower estimate upper
## unweighted kappa  0.06      0.1  0.14
## weighted kappa    0.06      0.1  0.14
##
##  Number of subjects = 2158
```

As we can see from our classification metrics, our predictions using simple logistic regression were comparable to flipping a coin in accuracy and precision, but tend to predict true positive results fairly often. On the flipside, the ability to predict true negative results was quite poor.

Naive Bayes Model

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

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.5521437 0.4478563
##
## Conditional probabilities:
##    Time
## Y       [,1]     [,2]
##   0 763.4042 285.0996
##   1 841.0122 266.3328
##
##    Length
## Y       [,1]     [,2]
##   0 128.3459 67.44019
##   1 134.7477 70.88509
##
##    DayOfWeek
## Y            1          2          3          4          5          6
##   0 0.12927597 0.13578174 0.15445960 0.16873033 0.16537251 0.11668416
##   1 0.13402329 0.12652005 0.18292367 0.17852523 0.14877102 0.09573092
##    DayOfWeek
## Y            7
##   0 0.12969570
##   1 0.13350582
```

The Naive Bayes Model creates conditional probabilities for each predictor separately, assuming that each feature is independent of one another. The A-priori probabilities indicate the distribution of our data, or how likely any flight in the data set will be delayed or not. Below that are the means and standard deviations of the predictors within each class (not delayed and delayed).

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

```
##
## p1    0   1
##   0 919 642
##   1 290 307
```

```
print(paste("accuracy ", mean(p1==test$Class)))
```

```
## [1] "accuracy  0.568118628359592"
```

```
print(paste("precision ", t[1,1]/(t[1,1]+t[1,2])))
```

```
## [1] "precision  0.588725176169122"
```

```
print(paste("sensitivity ", t[1,1]/(t[1,1]+t[2,1])))
```

```
## [1] "sensitivity  0.760132340777502"
```

```
print(paste("specificty ", t[2,2]/(t[2,2]+t[1,2])))
```

```
## [1] "specificty  0.32349841938883"
```

```
cohen.kappa(t)
```

```
## Call: cohen.kappa1(x = x, w = w, n.obs = n.obs, alpha = alpha, levels = levels)
##
## Cohen Kappa and Weighted Kappa correlation coefficients and confidence boundaries
##                  lower estimate upper
## unweighted kappa 0.047    0.087  0.13
## weighted kappa   0.047    0.087  0.13
##
##  Number of subjects = 2158
```

Using the Naive Bayes model resulted in only marginally higher accuracy, precision, and specificity, but lower sensitivity. Overall I would say it performed slightly better than logistic regression due to these values. Although minute, I believe the reason that the Naive Bayes model performed better than logistic regression is because the predictors were largely independent from each other. Both models however, did not perform very well, as the low kappa correlation indicates that the agreement isn't much better than what would've been obtained by flipping a fair coin. Days of the week is a human concept that has no bearing on inclement weather, the main factor for delayed flights. The length of the flight should have no bearing on delayed flights because ultimately the length is determined once the plane has finished the flight, not whether it is delayed or not. Lastly the time of a flight may not directly have an effect on the weather, but temperature varies throughout the day and would cause pressure differences that may result in hazardous weather, which I believe is the main factor for our slightly better than average accuracy.

Conclusions

As stated previously, the main strength of the Naive Bayes model is that if the independence assumption holds, it will fit the data very well. This is also the main weakness, resulting in a higher bias and lower variance due to this assumption, whether true or false. While Naive Bayes is a generative algorithm, estimating P(Y) and P(X|Y), logistic regression is a discriminative algorithm, directly estimating for P(X|Y) from data. As we could see from our summary we explored, the predictors gave inference regarding the importance of each feature, which isn't offered from Naive Bayes. A downside of logistic regression, and one reason why it didn't support our data very well, is that logistic regression tends to degrade if irrelevant features are included in the model.

Our data included the day of the week and length of the flight, which seemed to be largely irrelevant.

Accuracy is the proportion of true results out of the total results, which is useful when the classes are balanced and data isn't high variance. Precision is the percentage of true positive results out of total positive results. Higher precision is useful when false positives are detrimental. Sensitivity is the percentage of positive results were true positive. Higher sensitivity is useful when we want to achieve positives, but false positives are better than false negatives. Common example is in medicine, when missing a diagnosis is much more dangerous than being misdiagnosed. Specificity is the percentage of negative results were true negatives. The inverse of sensitivity, and when the model is correctly identifying negatives. Cohen Kappa is a calculation that accounts for guessing correctly by chance, i.e. were your results a fluke.