Data-driven Customizable Travel Insurance

Project Team: 1. Aakash Desai

2. Jinali Gandhi

3. Kanchan Markandeya

4. Srishti Kabtiyal

ORIGINAL WORK STATEMENT

We the undersigned certify that the actual composition of this proposal was done by us and is original work.

Signatures:

Name	**Signature**
Aakash Desai	AD
Jinali Gandhi	JG
Kanchan Markandeya	KM
Srishti Kabtiyal	SK

Table of Contents

I. EXECUTIVE SUMMARY

The U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics (BTS) tracks the on-time performance of domestic flights operated by large air carriers. This information is being used by large insurance companies to fundamentally change the way travel insurance is priced and perceived by the public. Berkshire Hathaway Travel Protection is one such travel insurance product that utilizes data to create uniquely priced insurance products based on the likelihood of a flight being delayed. Hence, it has become more important than ever to accurately predict the likelihood of delay of flights. Also, it is important to understand the various factors that significantly cause delay of flights along with the length of delay. Inspired from BHTP, our team has undertaken the task of predicting flight delays from a large data set acquired from the US Department of Transportation.

II. DATA DESCRIPTION

Data Source: Our data source was acquired from Kaggle.com and was generated by US Department of Transportation's Bureau of Transportation Statistics. It comprised of flight logs for all domestic flights in the USA for the year 2015.

The various fields in our data set are described below:

Sr No.	Field	Description
1	Year	Self-Explanatory
2	Month	
3	Day	
4	Day of Week	
5	Airline	
6	Flight Number	
7	Tail Number	Registration Number of Aircraft
8	Origin Airport	ICAO code of source
9	Destination Airport	ICAO code of destination
10	Scheduled Departure	Scheduled time of departure from gate

11	Actual Departure Time	Denotes when aircraft left gate
12	Departure Delay	Difference of above two fields (if any)
13	Taxi Out	Time needed for aircraft to taxi to runway
14	Wheels Off	Actual time of take-off
15	Scheduled Time	Planned gate to gate time for flight
16	Elapsed Time	Actual gate to gate time for flight
17	Airtime	Amount of time in air
18	Distance	Self-Explanatory
19	Wheels On	Actual time of landing
20	Taxi In	Time needed to taxi to gate
21	Scheduled Arrival	Scheduled time for arrival at gate
22	Actual arrival time	Actual time for arrival at gate
23	Arrival Delay	Number of minutes flight is delayed
24	Diverted	Binary variable denoting if flight was diverted
25	Cancelled	Binary variable denoting if flight was cancelled
26	Cancellation Reason	Categorical variable denoting reason of flight delay (if known)
27	Air System Delay	Amount of delay caused due to air traffic congestion in minutes
28	Security Delay	Amount of delay caused due to security clearance in minutes
29	Airline Delay	Amount of delay caused due to airline faults in minutes
30	Late Aircraft Delay	Amount of delay caused due to late aircraft delay in minutes
31	Weather Delay	Amount of delay caused due to weather in minutes

The dataset consisted of a total of **5.9 million** records with 31 variables each with some instances having occurrence of null values.

III. RESEARCH QUESTIONS

The fundamental goal of this project was to create knowledgeable and actionable insight that travel insurance companies can use to price different flights differently depending on their likelihood to be delayed. This likelihood would translate to a higher probability of payout for the insurance company. This insight could be used by the insurance company to hike their premium on said variable combinations, routes or flights. Since this project relied heavily on the payout models of companies like Berkshire Hathaway Travel Insurance and similar companies, we used the models and created our prediction models to mirror these actual business needs.

Travel Insurance companies usually have a set/variable payout associated with fixed intervals of delay. BHTP, for example, has a $100 payout associated with a two-hour delay and offer to rebook a passenger on the next available flight on the passenger's choice of airline if the flight is delayed by four hours or more. This means that travel insurance majors need to be able to tell whether a flight is likely to be delayed by two hours or more and then also predict whether a flight is likely to be delayed by four hours or more. These two predictions have direct business value to insurance companies as it gives them insight as to how often they may have to payout a passenger for a certain set of circumstances or flights. This information can be used as a robust baseline for determining premiums for flights, increasing premiums on flights with high likelihood of delay and decreasing premiums for flights with low likelihood of delay.

Initially, we created models that would attempt to predict our dependent variable, i.e. **Arrival Delay** as a numerical outcome. This was simple in terms of actual model creation since the data already existed in the required format and the model could be created directly. However, we quickly realized that the models we were creating had two-fold problems. Firstly, it was very difficult to predict a numerical outcome with no model achieving good predictive ability. Secondly, the business driver of this problem did not demand such high degree of accuracy in determining delay. Hence, a business compromise was desired. Hence, we switched our approach by bucketing the dependent variable into 15 minute buckets of delay. We than ran numerical outcome predictions on these buckets. These models were closer to the desired business value of the problem. However, our models could not generate good levels of accuracy even after the outcome was bucketed.

Considering the results that we were generating, we switched the problem from a numerical outcome prediction to a classification problem. We started to generate models to predict whether a flight would be delayed by two hours or not and generating models to predict whether the flight would be delayed by four hours or not.

IV. METHODOLOGY

Initially, we performed a data cleaning operation to eliminate any columns that did not add value or were posing a hindrance to the functioning of the predictive model that we were trying to build. This data cleaning involved the following steps:

- **Elimination of Null values:** This stage involved the elimination of all tuples that had one or more attributes associated with null occurrences or missing data.
- **Data sub-setting based on month and routes of the flights:** Considering the massive size of the data set, we decided to sub set the data set by initially only analyzing the month of January on all combination of routes for the 15 busiest airports in the US.
- **Bucketing of Time variables**: The time variables were converted from an absolute time to buckets specifying the hour of day. This would be a more interpretable outcome.
- **Elimination of Cause variables:** Certain extreme occurrences of delay were logged in the data set along with the cause of said delay such as security emergency, weather delay, etc. These variables were eliminated from the analyses as it is impossible to know of these conditions before hand and take any intelligent decision on pricing of premiums based upon them. Hence, they were removed.
- **Binning of dependent variable:** The dependent variables were binned or bucketed based upon which business scenario was catered to by the model in question.

We employed all the below specified models when performing our analyses.

1. Linear Regression

- Forward & Backward Selection
- Best Subset Selection
- Ridge Regression
- Lasso

Initially, we ran linear regression with all the variables on our training dataset with dependent variable AIRLINE_DELAY bucketed into 15 min timeframes. The regression output on training set had an adjusted R-square of 15%. After fitting the training model on validation set, we got mean square error of 1.72. Since the training model explained only 15% of the variation, we had to use feature selection methods to find which variables were important contributors to the fit of the model. We first ran forward selection method and got a subset of 9 variables that fit the model best. Similarly, for backward selection and best subset selection, we got best subset of 9 variables too but they were slightly different from the best coefficient results from forward selection. After fitting regression using only the feature selection variables and predicting delays on validation set, we still got MSE of around 1.73. This proved that the model was not performing any better with feature selection as well. We then tried running regularization models of lasso and ridge regression which use a shrinkage factor to push coefficients of unimportant features to 0 or towards 0 respectively. Using the non-zero variables from lasso and ridge regression, we predicted delays on validation set but surprisingly still no notable improvements in MSE.

2. Regression Trees

Finally, we created regression tree for the full variable set and got best pruned tree with 3 terminal nodes and decision nodes for variables TAXI_IN and TAXI_OUT. Since the taxi times are not usually known beforehand and exploring the predictions just on the basis of these two does not make sense, we decided to change the approach to classification. Also, the MSE for regression trees was 1.94 which further fuelled our decision to change to classification.

3. Classification Trees

Initially, we attempted to run a simple classification tree with all the relevant independent variables as possible predictors. However, this did not yield a good tangible outcome with many misclassifications occurring. This was when we discovered that data sparsity of our success class was a major problem. The number of delayed flights in the data set were such a small portion of the data set at large that any model we tried to fit was not able to learn enough of the features of the delayed flights to predict them properly. Hence, we attempted different approached to solve the problem of data sparsity.

- **Over Sampling**

In this approach, we over-sampled the delayed flights in our training data set so that a ratio of 1:1 is achieved. In such a data set, the algorithm and model can learn about the features of delayed flights in more detail.

- **Under Sampling**

In this approach, we use a similar methodology as the above method, but use less of the non-delayed flights to achieve a similar result.

- **Random Data Synthesis**

Data synthesis is an approach by which new tuples of the success class are artificially created using data points in the actual data set. It is often known to perform much better in majority of the situations.

- **K-Nearest Neighbors**

We ran a simple K-NN classification and attempted to predict whether a flight will be delayed by two hours or not. Also, we built a K-NN model to predict whether a flight will be delayed by four hours or not. All variables under consideration were first scaled to produce an optimal result. The K-NN algorithm was also run over-sampled and under-sampled data. This helped the model to learn about features leading to delay of flights. We restricted the number of destination and origin airports to comply with the specification of the library functions that perform K-NN in R. The variables under consideration for the classification were DAY_OF_WEEK, AIRLINE, DESTINATION_AIRPORT, SCHEDULED_DEPARTURE, SCHEDULED_ARRIVAL and ORIGIN_AIRPORT. These variables were specifically chosen due to their availability to predict delays prior to sale of insurance.

- Association Rules

The idea behind computing association rules was to understand the relationship (association) between independent attributes that cause the delay of flights. For the analysis, we used the complete data set consisting of 5.8 million rows. We made a selection of variables, based on their feasibility to be converted into a binary matrix. Some of the factors we took into consideration for the analysis are '*month*', '*day of week'*, '*airline*', '*origin airport'* and so on. Considering further detailed variables such as '*airtime'*, would have created an extensive matrix.

After computing the association rules, following are the top rules:

- If Chicago is either the origin airport or the destination airport, then there is a high chance of delay.
- If the scheduled departure is at 6pm in the month of June, then it is likely to be a delay.
- If the scheduled arrival of a flight is 8pm then it is likely to be delayed.
- A United Airlines flight usually has a high tendency to be delayed.

V. RESULTS AND FINDINGS

1. Linear Regression

- Forward & Backward Selection
- Best Subset Selection
- Ridge Regression
- Lasso

As discussed earlier, the regression output on training set had an adjusted R-square of 15%. After fitting the training model on validation set, we got mean square error of 1.72. After fitting regression using only the feature selection variables and predicting delays on validation set, we still got MSE of around 1.73. This proved that the model was not performing any better with feature selection as well. We then tried running regularization models of lasso and ridge regression which use a shrinkage factor to push coefficients of unimportant features to 0 or towards 0 respectively. Using the non-zero variables from lasso and ridge regression, we predicted delays on validation set but surprisingly still no notable improvements in MSE.

2. Regression Trees

We got the best pruned tree with 3 terminal nodes and decision nodes for variables TAXI_IN and TAXI_OUT. Since the taxi times are not usually known beforehand and exploring the predictions just on the basis of these two does not make sense, we decided to change the approach to classification. Also, the MSE for regression trees was 1.94 which further fuelled our decision to change to classification.

3. Classification Trees

The classification trees generated the following outputs. The built models have accuracy ratings ranging from 55% to 65%. However, this is an incorrect measure of model performance. The use of false positive and true positive rate is far more usable and understandable. Using this metric, the tree created with an over sampling approach (predicting 2 hour delays) generated the best model.

4. K-Nearest Neighbors

The results of the KNN classifier are showcased below. It is quite clear from the below results that KNN holds little to no business value. It performed very poorly. Although the model had a high accuracy, it achieved this accuracy by classifying all flight as on-time. Clearly, this does yield any business value to the central question of interest.

5. Association Rules

Following were the top rules we computed using association rules approach:

- If Chicago is either the origin airport or the destination airport, then there is a high chance of delay.
- If the scheduled departure is at 6pm in the month of June, then it is likely to be a delay.
- If the scheduled arrival of a flight is 8pm then it is likely to be delayed.
- A United Airlines flight usually has a high tendency to be delayed.

VI. CONCLUSION

In conclusion, it is evident from the above analysis that this was indeed a very challenging data set to work with, especially considering the nature of the question that we had posed for analyses. A lot of insurance industries such as home insurance, car insurance etc. have been capitalizing on data mining techniques for a long time, because of which they have been able to provide customized insurance plans to their customers. Whereas, the travel insurance industry is still working on flat rate insurance plans. This needs to change.

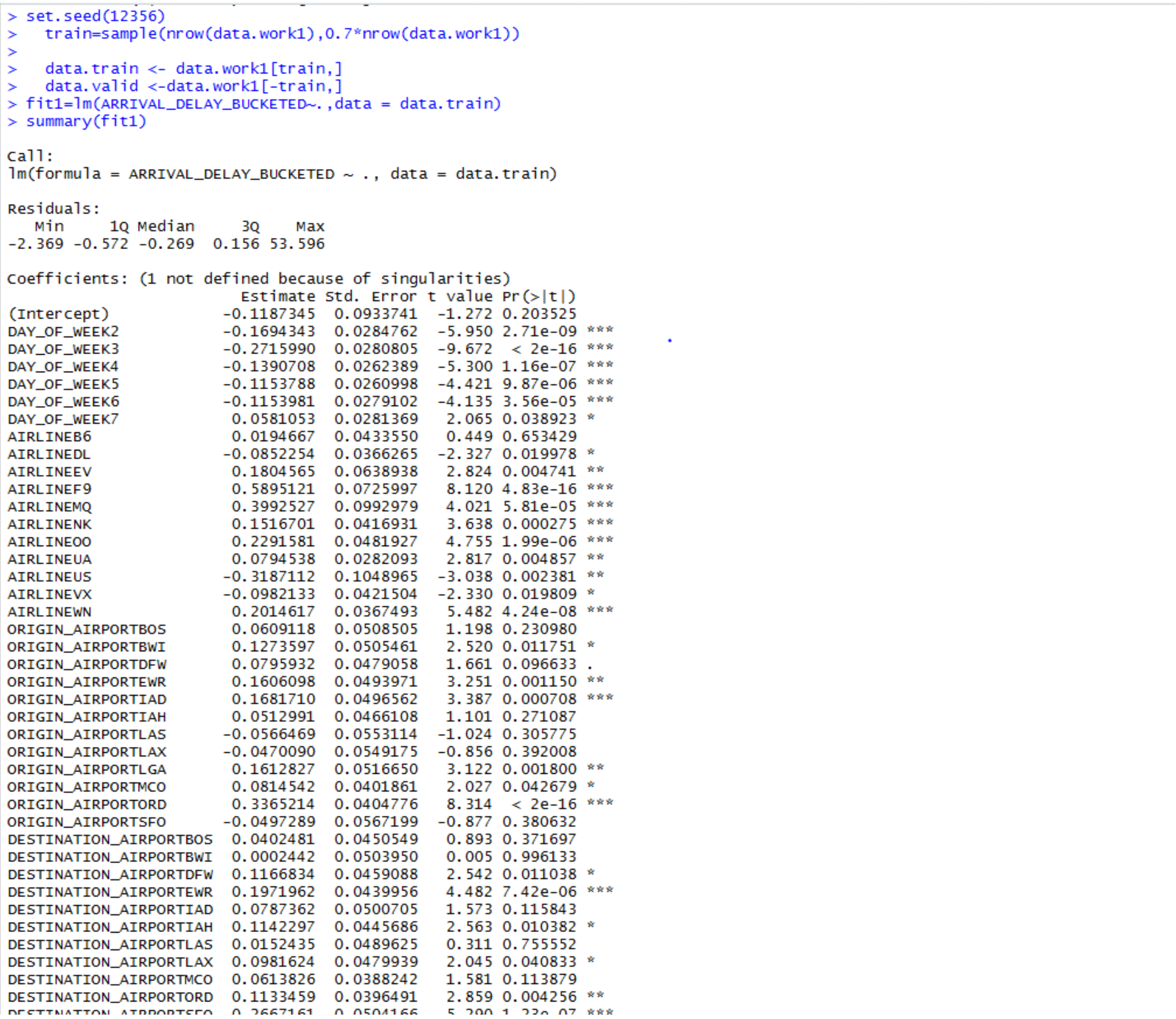
The airline travel industry is approximately an 800 billion-dollars-a-year industry, which means that there is a lot of scope for profit making and utilizing data mining techniques to improve upon business processes.

VII. LIMITATIONS AND IMPROVEMENTS

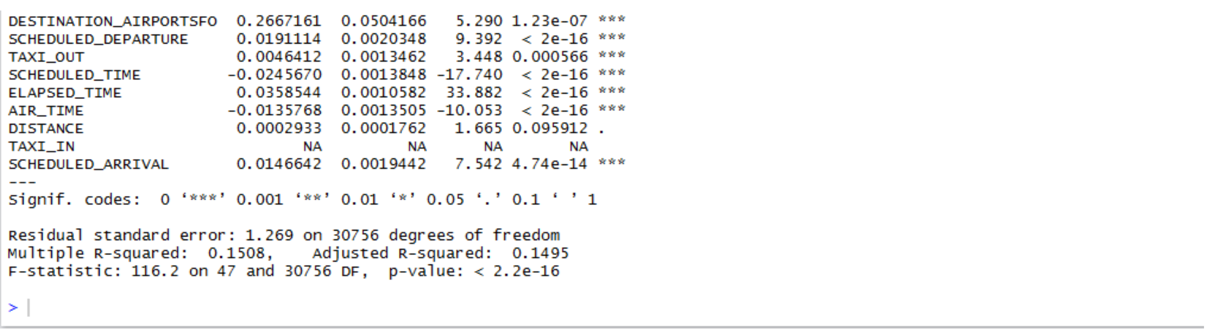
In our opinion, the most glaring problem that we faced was the sparsity of the data itself. The success class, i.e. delayed flights, constituted a very small proportion of the data set at large. This did not allow any model to accurately learn about the various features so as to predict the delayed flights well. We did attempt to mitigate and solve this problem by creating a balanced training data set using over-sampling and data synthesis. This did improve the performance of all our classifiers from a business perspective (TPR + FPR). However, there is much improvement to be made in the model before it is implemented to dynamically and accurately price premiums. There is a wealth of academic research that deals with the handling of imbalanced/sparse datasets. Some of these approaches could be used to design better models. Considering the size of the data set, we could also have utilized Big Data tools such as Hadoop to analyze the data.

VIII. APPENDIX

1. Linear Regression Model

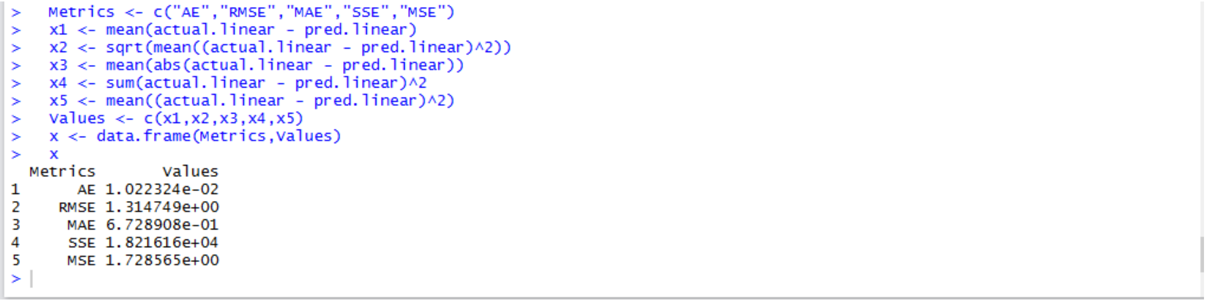


```
> set.seed(12356)
>   train=sample(nrow(data.work1),0.7*nrow(data.work1))
> 
>   data.train <- data.work1[train,]
>   data.valid <-data.work1[-train,]
> fit1=lm(ARRIVAL_DELAY_BUCKETED~.,data = data.train)
> summary(fit1)

Call:
lm(formula = ARRIVAL_DELAY_BUCKETED ~ ., data = data.train)

Residuals:
   Min      1Q Median      3Q     Max 
-2.369 -0.572 -0.269  0.156 53.596 

Coefficients: (1 not defined because of singularities)
                        Estimate Std. Error t value Pr(>|t|)    
(Intercept)           -0.1187345  0.0933741  -1.272 0.203525    
DAY_OF_WEEK2          -0.1694343  0.0284762  -5.950 2.71e-09 ***
DAY_OF_WEEK3          -0.2715990  0.0280805  -9.672  < 2e-16 ***
DAY_OF_WEEK4          -0.1390708  0.0262389  -5.300 1.16e-07 ***
DAY_OF_WEEK5          -0.1153788  0.0260998  -4.421 9.87e-06 ***
DAY_OF_WEEK6          -0.1153981  0.0279102  -4.135 3.56e-05 ***
DAY_OF_WEEK7           0.0581053  0.0281369   2.065 0.038923 *  
AIRLINEB6              0.0194667  0.0433550   0.449 0.653429    
AIRLINEDL             -0.0852254  0.0366265  -2.327 0.019978 *  
AIRLINEEV              0.1804565  0.0638938   2.824 0.004741 ** 
AIRLINEF9              0.5895121  0.0725997   8.120 4.83e-16 ***
AIRLINEMQ              0.3992527  0.0992979   4.021 5.81e-05 ***
AIRLINENK              0.1516701  0.0416931   3.638 0.000275 ***
AIRLINEOO              0.2291581  0.0481927   4.755 1.99e-06 ***
AIRLINEUA              0.0794538  0.0282093   2.817 0.004857 ** 
AIRLINEUS             -0.3187112  0.1048965  -3.038 0.002381 ** 
AIRLINEVX             -0.0982133  0.0421504  -2.330 0.019809 *  
AIRLINEWN              0.2014617  0.0367493   5.482 4.24e-08 ***
ORIGIN_AIRPORTBOS      0.0609118  0.0508505   1.198 0.230980    
ORIGIN_AIRPORTBWI      0.1273597  0.0505461   2.520 0.011751 *  
ORIGIN_AIRPORTDFW      0.0795932  0.0479058   1.661 0.096633 .  
ORIGIN_AIRPORTEWR      0.1606098  0.0493971   3.251 0.001150 ** 
ORIGIN_AIRPORTIAD      0.1681710  0.0496562   3.387 0.000708 ***
ORIGIN_AIRPORTIAH      0.0512991  0.0466108   1.101 0.271087    
ORIGIN_AIRPORTLAS     -0.0566469  0.0553114  -1.024 0.305775    
ORIGIN_AIRPORTLAX     -0.0470090  0.0549175  -0.856 0.392008    
ORIGIN_AIRPORTLGA      0.1612827  0.0516650   3.122 0.001800 ** 
ORIGIN_AIRPORTMCO      0.0814542  0.0401861   2.027 0.042679 *  
ORIGIN_AIRPORTORD      0.3365214  0.0404776   8.314  < 2e-16 ***
ORIGIN_AIRPORTSFO     -0.0497289  0.0567199  -0.877 0.380632    
DESTINATION_AIRPORTBOS 0.0402481  0.0450549   0.893 0.371697    
DESTINATION_AIRPORTBWI 0.0002442  0.0503950   0.005 0.996133    
DESTINATION_AIRPORTDFW 0.1166834  0.0459088   2.542 0.011038 *  
DESTINATION_AIRPORTEWR 0.1971962  0.0439956   4.482 7.42e-06 ***
DESTINATION_AIRPORTIAD 0.0787362  0.0500705   1.573 0.115843    
DESTINATION_AIRPORTIAH 0.1142297  0.0445686   2.563 0.010382 *  
DESTINATION_AIRPORTLAS 0.0152435  0.0489625   0.311 0.755552    
DESTINATION_AIRPORTLAX 0.0981624  0.0479939   2.045 0.040833 *  
DESTINATION_AIRPORTMCO 0.0613826  0.0388242   1.581 0.113879    
DESTINATION_AIRPORTORD 0.1133459  0.0396491   2.859 0.004256 ** 
```



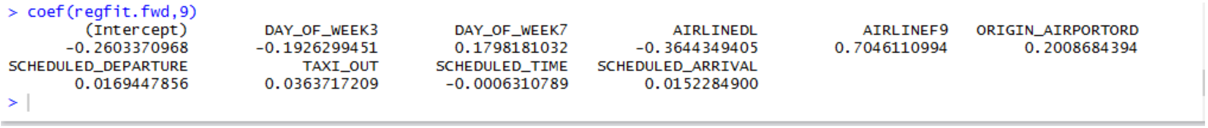
```
DESTINATION_AIRPORTSFO 0.2667161  0.0504166   5.290 1.23e-07 ***
SCHEDULED_DEPARTURE    0.0191114  0.0020348   9.392  < 2e-16 ***
TAXI_OUT               0.0046412  0.0013462   3.448 0.000566 ***
SCHEDULED_TIME        -0.0245670  0.0013848 -17.740  < 2e-16 ***
ELAPSED_TIME           0.0358544  0.0010582  33.882  < 2e-16 ***
AIR_TIME              -0.0135768  0.0013505 -10.053  < 2e-16 ***
DISTANCE               0.0002933  0.0001762   1.665 0.095912 .  
TAXI_IN                       NA         NA      NA       NA    
SCHEDULED_ARRIVAL      0.0146642  0.0019442   7.542 4.74e-14 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.269 on 30756 degrees of freedom
Multiple R-squared:  0.1508,	Adjusted R-squared:  0.1495 
F-statistic: 116.2 on 47 and 30756 DF,  p-value: < 2.2e-16

> |
```

1.1 MSE for linear regression



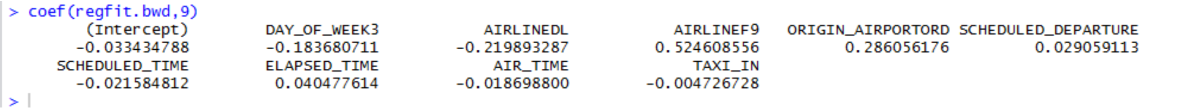
```
>   Metrics <- c("AE","RMSE","MAE","SSE","MSE")
>   x1 <- mean(actual.linear - pred.linear)
>   x2 <- sqrt(mean((actual.linear - pred.linear)^2))
>   x3 <- mean(abs(actual.linear - pred.linear))
>   x4 <- sum(actual.linear - pred.linear)^2
>   x5 <- mean((actual.linear - pred.linear)^2)
>   Values <- c(x1,x2,x3,x4,x5)
>   x <- data.frame(Metrics,Values)
>   x
  Metrics       Values
1      AE 1.022324e-02
2    RMSE 1.314749e+00
3     MAE 6.728908e-01
4     SSE 1.821616e+04
5     MSE 1.728565e+00
> |
```

1.2 Forward selection variable subset



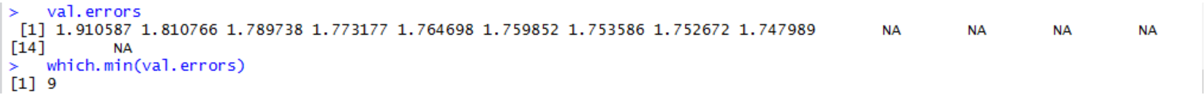
```
> coef(regfit.fwd,9)
        (Intercept)        DAY_OF_WEEK3        DAY_OF_WEEK7          AIRLINEDL          AIRLINEF9   ORIGIN_AIRPORTORD
      -0.2603370968       -0.1926299451        0.1798181032      -0.3644349405       0.7046110994        0.2008684394
SCHEDULED_DEPARTURE            TAXI_OUT      SCHEDULED_TIME  SCHEDULED_ARRIVAL
       0.0169447856        0.0363717209       -0.0006310789       0.0152284900
> |
```

1.3 Backward selection variable subset

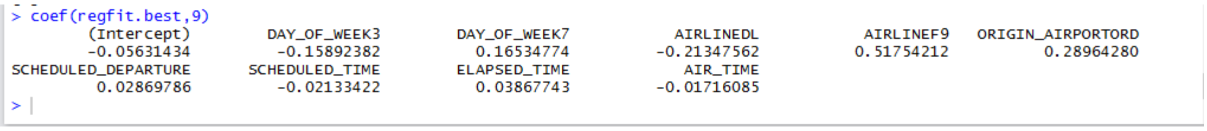


```
> coef(regfit.bwd,9)
        (Intercept)        DAY_OF_WEEK3           AIRLINEDL           AIRLINEF9   ORIGIN_AIRPORTORD SCHEDULED_DEPARTURE
       -0.033434788        -0.183680711        -0.219893287         0.524608556         0.286056176         0.029059113
     SCHEDULED_TIME        ELAPSED_TIME            AIR_TIME             TAXI_IN
       -0.021584812         0.040477614        -0.018698800        -0.004726728
> |
```

1.4 Best subset selection / validation error

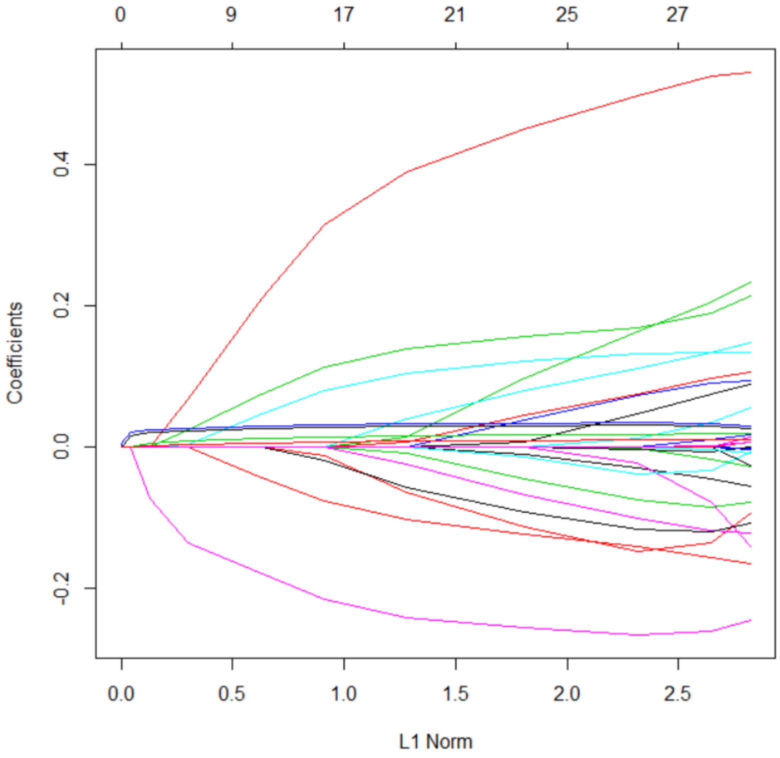


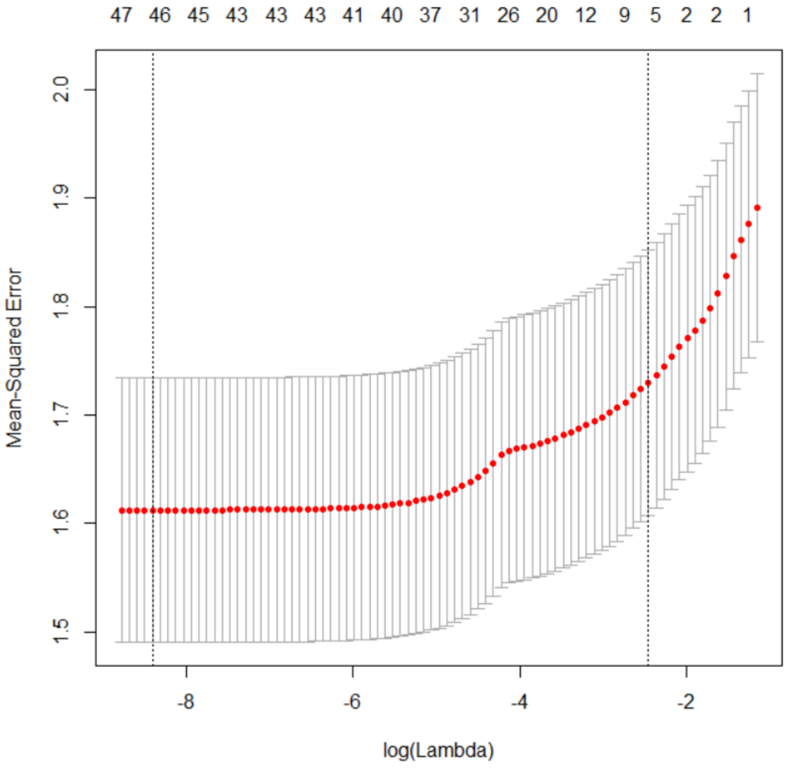
```
>   val.errors
 [1] 1.910587 1.810766 1.789738 1.773177 1.764698 1.759852 1.753586 1.752672 1.747989       NA       NA       NA       NA
[14]       NA
>   which.min(val.errors)
[1] 9
```



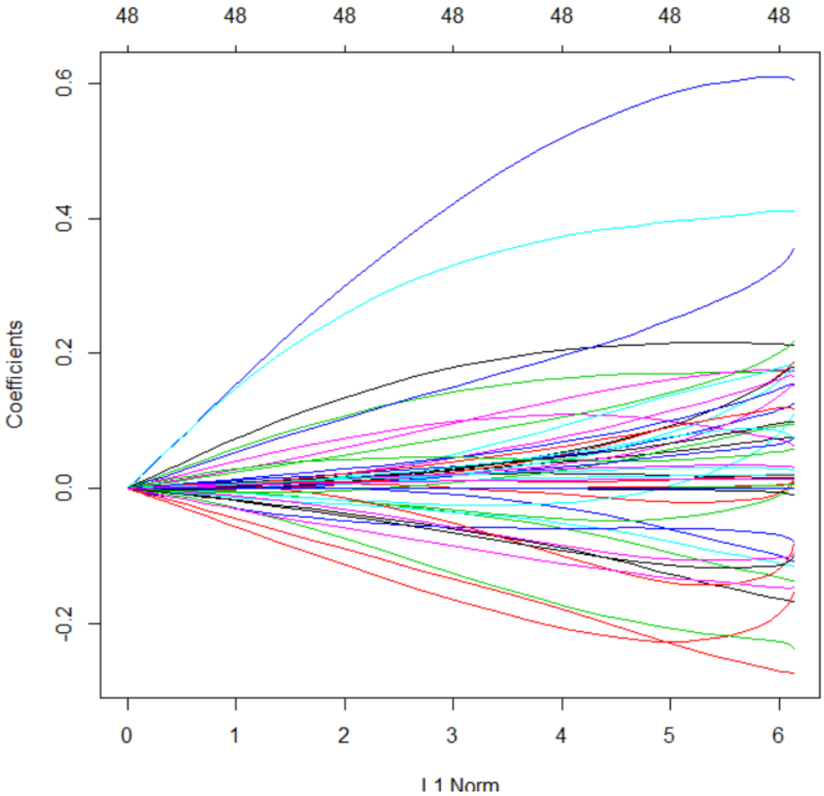
```
> coef(regfit.best,9)
        (Intercept)        DAY_OF_WEEK3        DAY_OF_WEEK7           AIRLINEDL           AIRLINEF9   ORIGIN_AIRPORTORD
        -0.05631434         -0.15892382          0.16534774         -0.21347562          0.51754212          0.28964280
SCHEDULED_DEPARTURE      SCHEDULED_TIME        ELAPSED_TIME            AIR_TIME
         0.02869786         -0.02133422          0.03867743         -0.01716085
> |
```

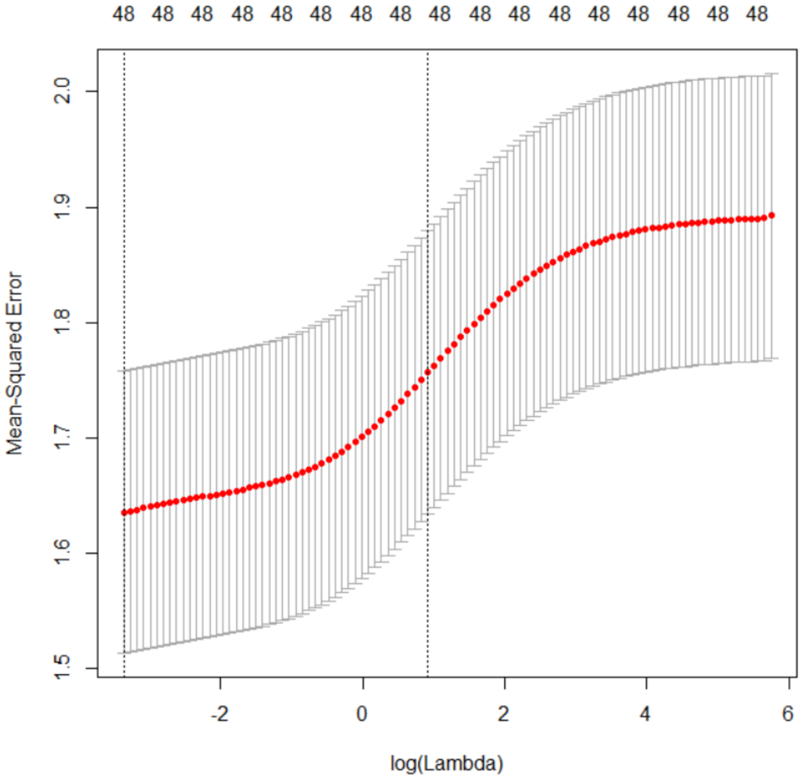
1.5 Lasso regression



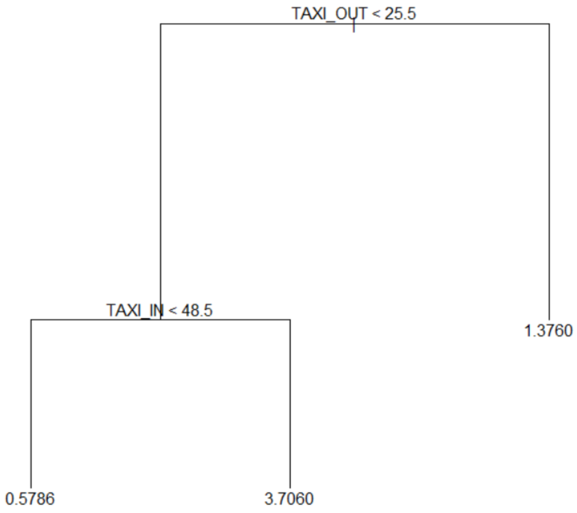


1.6 Ridge regression

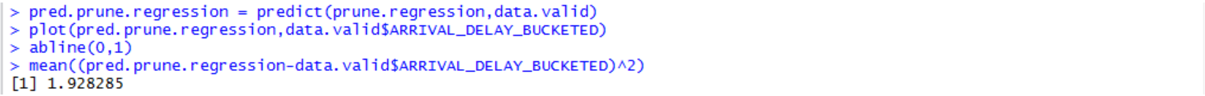




1.7 Best pruned tree

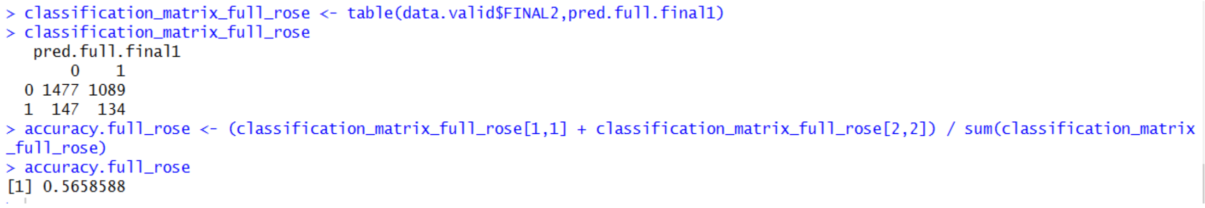


1.8 MSE for best pruned tree-

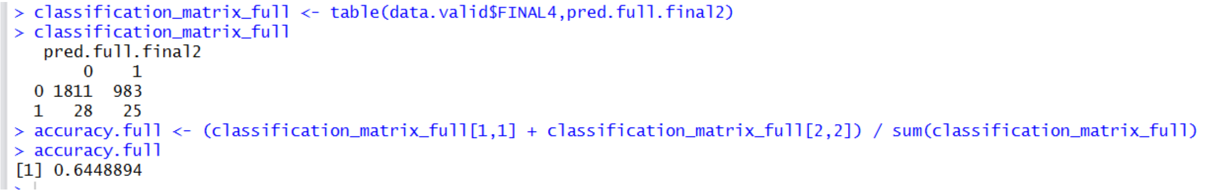


```
> pred.prune.regression = predict(prune.regression,data.valid)
> plot(pred.prune.regression,data.valid$ARRIVAL_DELAY_BUCKETED)
> abline(0,1)
> mean((pred.prune.regression-data.valid$ARRIVAL_DELAY_BUCKETED)^2)
[1] 1.928285
```

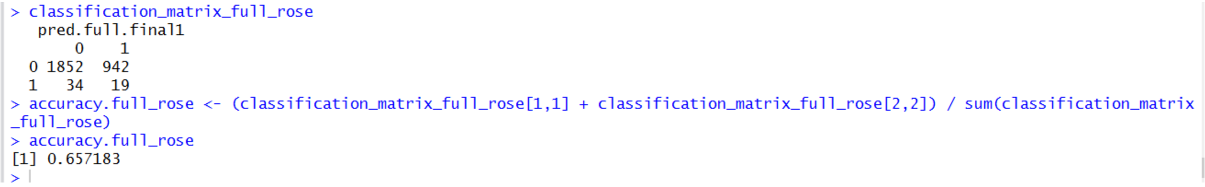
2. Classification



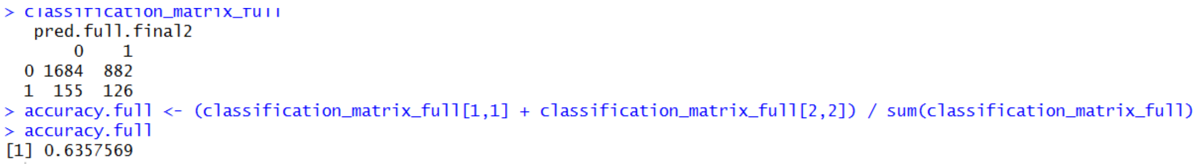
```
> classification_matrix_full_rose <- table(data.valid$FINAL2,pred.full.final1)
> classification_matrix_full_rose
   pred.full.final1
       0    1
  0 1477 1089
  1  147  134
> accuracy.full_rose <- (classification_matrix_full_rose[1,1] + classification_matrix_full_rose[2,2]) / sum(classification_matrix
_full_rose)
> accuracy.full_rose
[1] 0.5658588
```



```
> classification_matrix_full <- table(data.valid$FINAL4,pred.full.final2)
> classification_matrix_full
   pred.full.final2
       0    1
  0 1811  983
  1   28   25
> accuracy.full <- (classification_matrix_full[1,1] + classification_matrix_full[2,2]) / sum(classification_matrix_full)
> accuracy.full
[1] 0.6448894
```

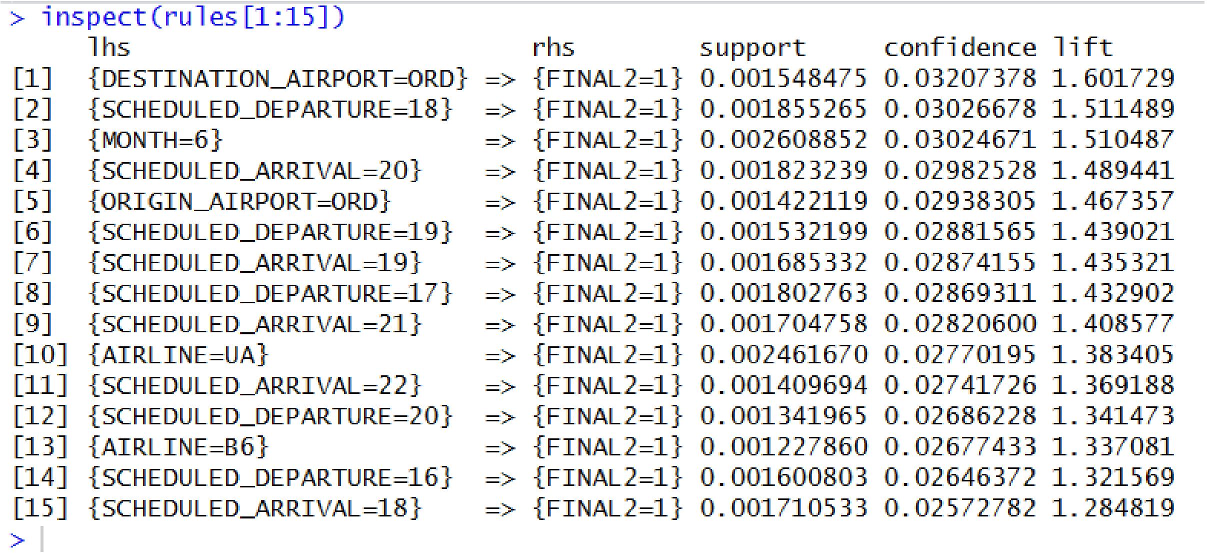


```
> classification_matrix_full_rose
   pred.full.final1
       0    1
  0 1852  942
  1   34   19
> accuracy.full_rose <- (classification_matrix_full_rose[1,1] + classification_matrix_full_rose[2,2]) / sum(classification_matrix
_full_rose)
> accuracy.full_rose
[1] 0.657183
>
```



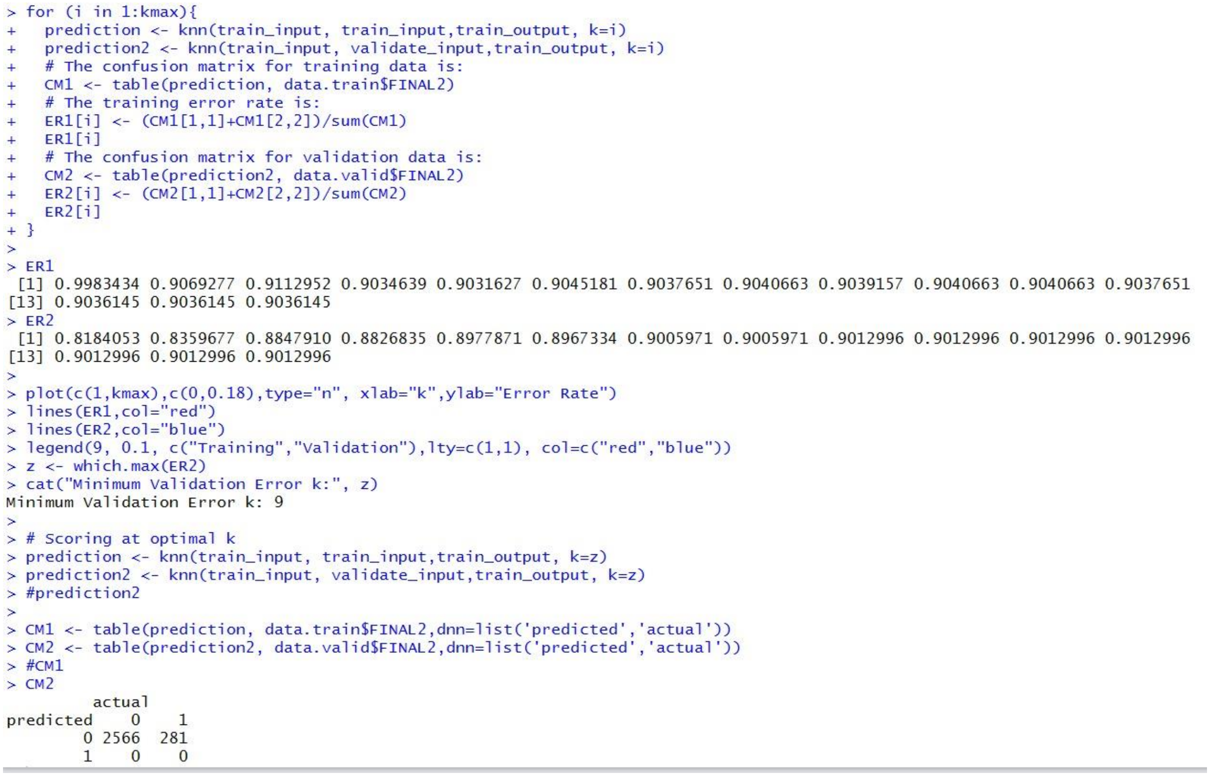
```
> classification_matrix_full
   pred.full.final2
       0    1
  0 1684  882
  1  155  126
> accuracy.full <- (classification_matrix_full[1,1] + classification_matrix_full[2,2]) / sum(classification_matrix_full)
> accuracy.full
[1] 0.6357569
```

3. Association Rules



```
> inspect(rules[1:15])
     lhs                             rhs         support     confidence lift
[1]  {DESTINATION_AIRPORT=ORD} => {FINAL2=1} 0.001548475 0.03207378 1.601729
[2]  {SCHEDULED_DEPARTURE=18}  => {FINAL2=1} 0.001855265 0.03026678 1.511489
[3]  {MONTH=6}                 => {FINAL2=1} 0.002608852 0.03024671 1.510487
[4]  {SCHEDULED_ARRIVAL=20}    => {FINAL2=1} 0.001823239 0.02982528 1.489441
[5]  {ORIGIN_AIRPORT=ORD}      => {FINAL2=1} 0.001422119 0.02938305 1.467357
[6]  {SCHEDULED_DEPARTURE=19}  => {FINAL2=1} 0.001532199 0.02881565 1.439021
[7]  {SCHEDULED_ARRIVAL=19}    => {FINAL2=1} 0.001685332 0.02874155 1.435321
[8]  {SCHEDULED_DEPARTURE=17}  => {FINAL2=1} 0.001802763 0.02869311 1.432902
[9]  {SCHEDULED_ARRIVAL=21}    => {FINAL2=1} 0.001704758 0.02820600 1.408577
[10] {AIRLINE=UA}              => {FINAL2=1} 0.002461670 0.02770195 1.383405
[11] {SCHEDULED_ARRIVAL=22}    => {FINAL2=1} 0.001409694 0.02741726 1.369188
[12] {SCHEDULED_DEPARTURE=20}  => {FINAL2=1} 0.001341965 0.02686228 1.341473
[13] {AIRLINE=B6}              => {FINAL2=1} 0.001227860 0.02677433 1.337081
[14] {SCHEDULED_DEPARTURE=16}  => {FINAL2=1} 0.001600803 0.02646372 1.321569
[15] {SCHEDULED_ARRIVAL=18}    => {FINAL2=1} 0.001710533 0.02572782 1.284819
> 
```

4. KNN



```
> for (i in 1:kmax){
+   prediction <- knn(train_input, train_input,train_output, k=i)
+   prediction2 <- knn(train_input, validate_input,train_output, k=i)
+   # The confusion matrix for training data is:
+   CM1 <- table(prediction, data.train$FINAL2)
+   # The training error rate is:
+   ER1[i] <- (CM1[1,1]+CM1[2,2])/sum(CM1)
+   ER1[i]
+   # The confusion matrix for validation data is:
+   CM2 <- table(prediction2, data.valid$FINAL2)
+   ER2[i] <- (CM2[1,1]+CM2[2,2])/sum(CM2)
+   ER2[i]
+ }
>
> ER1
 [1] 0.9983434 0.9069277 0.9112952 0.9034639 0.9031627 0.9045181 0.9037651 0.9040663 0.9039157 0.9040663 0.9040663 0.9037651
[13] 0.9036145 0.9036145 0.9036145
> ER2
 [1] 0.8184053 0.8359677 0.8847910 0.8826835 0.8977871 0.8967334 0.9005971 0.9005971 0.9012996 0.9012996 0.9012996 0.9012996
[13] 0.9012996 0.9012996 0.9012996
>
> plot(c(1,kmax),c(0,0.18),type="n", xlab="k",ylab="Error Rate")
> lines(ER1,col="red")
> lines(ER2,col="blue")
> legend(9, 0.1, c("Training","Validation"),lty=c(1,1), col=c("red","blue"))
> z <- which.max(ER2)
> cat("Minimum Validation Error k:", z)
Minimum Validation Error k: 9
>
> # Scoring at optimal k
> prediction <- knn(train_input, train_input,train_output, k=z)
> prediction2 <- knn(train_input, validate_input,train_output, k=z)
> #prediction2
>
> CM1 <- table(prediction, data.train$FINAL2,dnn=list('predicted','actual'))
> CM2 <- table(prediction2, data.valid$FINAL2,dnn=list('predicted','actual'))
> #CM1
> CM2
         actual
predicted    0    1
        0 2566  281
        1    0    0
```

IX. REFERENCES

- Our data source was acquired from Kaggle.com and was generated by US Department of Transportation's Bureau of Transportation Statistics.

  2015 Flight Delays and Cancellations: https://www.kaggle.com/usdot/flight-delays

- https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classification-problems/

- http://www-bcf.usc.edu/~gareth/ISL/ISLR%20Sixth%20Printing.pdf