Flight Delay

The data set provided contains flight details for 5,819,811 flights for the different months of 2014. The features include categorical variables such as Month, Carrier, Origin Airport Id and Destination Airport Id. Ther are ordinal variables such as Day of month, Day of week and cancelled. There also are numeric variables such as Departure delay, carrier delay, weather delay and distance.

```
require(dplyr)

require(gbm)

load(file="2014flights.Rdata")
```

Feature Engineering

We wish to build a model that would predict the delay time of a flight. Hence, we need remove all the cancelled flights as there cannot be any record of a delay linked to those flights. The carrier delay and weather delay have missing values for most of the observations so it s better to drop those features. Our response is the arrival delay as that would be the final delay for the flight. The arrival delay has a few missing values. There could be two possibilities for this. The first being that there was no delay and hence these were left blank, or the delay was not recored at all. The second case seems to be more likely as the flights that had no delay had a value of 0 and the flights that do not have the arrival delay recorded are also mssing the actual elapsed time and the air time. Hence I removed these observations as imputing zeroes here may have resulted in a weaker model.

```
df <- df %>% filter(CANCELLED==0)
#df<-df[,-c(4,6,8,11,12,13,20,24,25,26,27,28,29)]
df<-df[,-c(4,6,8,9,1,12,13,15,16,20,24,28,29)]
df<-na.omit(df)
format(object.size(df),units='MiB')

## [1] "477.5 MiB"

Y<- df$ARR_DELAY
X<- df[-13]
```

Training and Test set

I split the data into training and testing set for the purpose of getting the estimate of the risk. I split the data randomly in the ration of 70:30.

```r
smp_size <- floor(0.7 * nrow(df))

set.seed(123)
train_ind <- sample(seq_len(nrow(df)), size = smp_size)

X_train <- X[train_ind, ]
X_test <- X[-train_ind, ]

Y_train <- Y[train_ind]
Y_test <- Y[-train_ind]
```

Modelling

I chose the boosting method to generate a predictive model for the follwing reasons. 1] The model can handle more categories than any other modelling technique. 2] I can control the number of iterations easily. 3] The learning rate can be changed to create a model that learns slowly 4] It is easier to utilize all 8 cores of the machine to process in parallel.

An ideal model would be one which would learn slowly over a long perod of time. In other words, a model which has a very low learning rate and a very high number of iterations. This is not possible as we are limited by the computing power available.

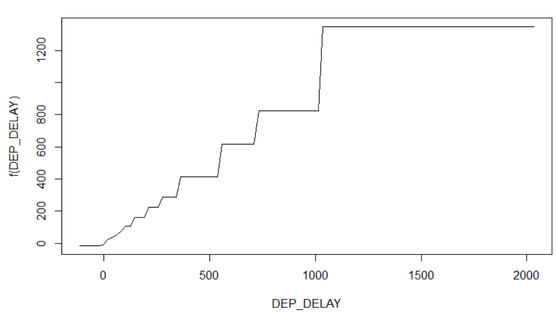
Model 1

I started of with a very basic model. This model performs only 10 iterations with a very high learning rate of 1. I used the gaussian distribution in order to get the mean squared error. The model performs a 3 fold cross validation and utilizes all 8 cores of the machine.

At the end of 10 iterations, the model has a training error of 152.023 which is very high but considering the small amount of iterations, is not very surprising. The model provides the departure delay as the most influential feature. It is highly correlated with the arrival delay, which makes sense as the late a flight leaves its place of origin, the more the chances are that it will arrive late at its destination. The other influential variables are the distance, actual time elapsed, carrier and the destination. These are the features that influence the delay time of a flight. The model performs only as well on the test set as it did on the training set with a risk estimate of 152.023

```
> out.boost<- gbm(Y_train~.,data=X_train,
+               distribution="gaussian",
+               n.trees=10,
+               shrinkage=1,
+               interaction.depth=3,
+               bag.fraction = 0.5,
+               n.minobsinnode = 10,
+               cv.folds = 3,
+               keep.data=TRUE,
+               verbose=TRUE,
+               n.cores=8)
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1        380.2851             nan     1.0000 1162.2218
     2        288.8450             nan     1.0000   91.2287
     3        242.9612             nan     1.0000   44.9896
     4        211.1285             nan     1.0000   31.9552
     5        189.2688             nan     1.0000   21.7915
     6        180.7461             nan     1.0000    8.3922
     7        175.1888             nan     1.0000    5.4617
     8        170.3177             nan     1.0000    4.8614
     9        159.2060             nan     1.0000   11.0357
    10        152.2414             nan     1.0000    6.8032

> boost.sum<-summary(out.boost)
> boost.sum
                                   var     rel.inf
DEP_DELAY                    DEP_DELAY 98.18396193
DISTANCE                      DISTANCE  0.97604584
ACTUAL_ELAPSED_TIME ACTUAL_ELAPSED_TIME  0.61932380
CARRIER                        CARRIER  0.15151396
DEST                              DEST  0.06915447
MONTH                            MONTH  0.00000000
DAY_OF_MONTH              DAY_OF_MONTH  0.00000000
DAY_OF_WEEK                DAY_OF_WEEK  0.00000000
ORIGIN_AIRPORT_ID     ORIGIN_AIRPORT_ID  0.00000000
ORIGIN                          ORIGIN  0.00000000
DEST_AIRPORT_ID         DEST_AIRPORT_ID  0.00000000
DEST_CITY_NAME           DEST_CITY_NAME  0.00000000
DEST_STATE_ABR           DEST_STATE_ABR  0.00000000
CRS_DEP_TIME               CRS_DEP_TIME  0.00000000
AIR_TIME                       AIR_TIME  0.00000000
> out.boost
gbm(formula = Y_train ~ ., distribution = "gaussian", data = X_train,
    n.trees = 10, interaction.depth = 3, n.minobsinnode = 10,
    shrinkage = 1, bag.fraction = 0.5, cv.folds = 3, keep.data = TRUE,
    verbose = TRUE, n.cores = 8)
A gradient boosted model with gaussian loss function.
10 iterations were performed.
The best cross-validation iteration was 10.
There were 15 predictors of which 5 had non-zero influence.
> best.iter<- gbm.perf(out.boost,method = "cv")
> f.predict<- predict(out.boost,X_test,best.iter)
> print(mean((Y_test-f.predict)**2))
[1] 152.023
> most.influential = which(names(X_train)%in%boost.sum[1:1,1])
> plot(out.boost,i.var=most.influential)
```



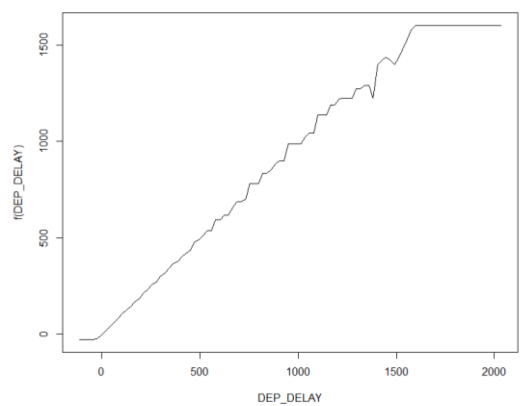
Model 2

The second model I generated performed 300 iterations with a learning rate of 0.5. As expected, the model learns slower than the previous one but the higher number of iterations succesfully form a better model than before.

The training error for this model is 144.9521. The most influential feature is the departure delay once again. The other influencial feature include origin, destination, carrier and month. This model performs better on the test set as well with a estimated risk of 145.8233

```
> out.boost<- gbm(Y_train~.,data=X_train,
+                 distribution="gaussian",
+                 n.trees=300,
+                 shrinkage=0.5,
+                 interaction.depth=3,
+                 bag.fraction = 0.5,
+                 n.minobsinnode = 10,
+                 cv.folds = 3,
+                 keep.data=TRUE,
+                 verbose=TRUE,
+                 n.cores=6)
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
     1       670.9827             nan     0.5000  871.7971
     2       366.3748             nan     0.5000  301.8876
     3       253.5090             nan     0.5000  111.5636
     4       208.8926             nan     0.5000   44.6594
     5       191.7110             nan     0.5000   16.9631
     6       184.3492             nan     0.5000    7.1307
     7       180.4218             nan     0.5000    3.8736
     8       177.3906             nan     0.5000    2.9145
     9       175.6283             nan     0.5000    1.7213
    10       174.3547             nan     0.5000    1.2512
    20       162.1208             nan     0.5000    0.5619
    40       153.5854             nan     0.5000    0.2276
    60       151.0596             nan     0.5000    0.1291
    80       149.2930             nan     0.5000    0.0655
   100       148.4071             nan     0.5000    0.0156
   120       147.9339             nan     0.5000    0.0024
   140       147.3724             nan     0.5000    0.0113
   160       146.9384             nan     0.5000    0.0124
   180       146.6292             nan     0.5000   -0.0008
   200       146.3979             nan     0.5000   -0.0034
   220       145.8849             nan     0.5000    0.0225
   240       145.6933             nan     0.5000    0.0034
   260       145.3223             nan     0.5000   -0.0018
   280       145.0609             nan     0.5000    0.0170
   300       144.9521             nan     0.5000   -0.0054
```

```
> summary(out.boost)
                                          var      rel.inf
DEP_DELAY                           DEP_DELAY 98.92904409
ORIGIN                                 ORIGIN  0.36930104
DEST                                     DEST  0.29825246
CARRIER                               CARRIER  0.26139367
MONTH                                   MONTH  0.06525464
CRS_DEP_TIME                     CRS_DEP_TIME  0.02358437
DAY_OF_WEEK                       DAY_OF_WEEK  0.01897030
DAY_OF_MONTH                     DAY_OF_MONTH  0.01718767
DEST_CITY_NAME                 DEST_CITY_NAME  0.01701176
ORIGIN_AIRPORT_ID           ORIGIN_AIRPORT_ID  0.00000000
ORIGIN_CITY_MARKET_ID ORIGIN_CITY_MARKET_ID  0.00000000
DEST_AIRPORT_ID               DEST_AIRPORT_ID  0.00000000
DEST_AIRPORT_SEQ_ID       DEST_AIRPORT_SEQ_ID  0.00000000
DEST_CITY_MARKET_ID       DEST_CITY_MARKET_ID  0.00000000
DEST_STATE_ABR                 DEST_STATE_ABR  0.00000000
> out.boost
gbm(formula = Y_train ~ ., distribution = "gaussian", data = X_train,
    n.trees = 300, interaction.depth = 3, n.minobsinnode = 10,
    shrinkage = 0.5, bag.fraction = 0.5, cv.folds = 3, keep.data = TRUE,
    verbose = TRUE, n.cores = 6)
A gradient boosted model with gaussian loss function.
300 iterations were performed.
The best cross-validation iteration was 300.
There were 15 predictors of which 9 had non-zero influence.
> best.iter<- gbm.perf(out.boost,method = "cv")
> f.predict<- predict(out.boost,X_test,best.iter)
> print(mean((Y_test-f.predict)**2))
[1] 145.8233
```



Model 3

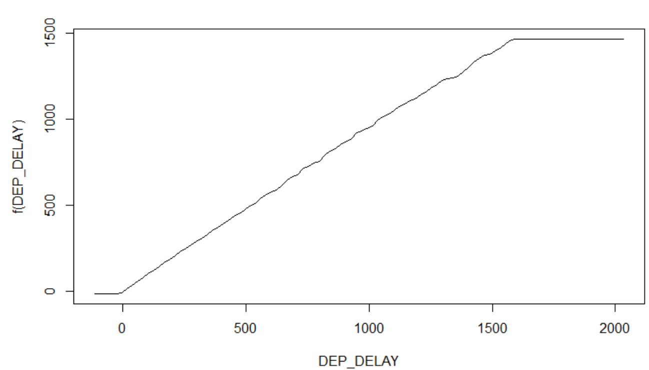
This model is close to the kind of model we would look for through the boosting process. It performs 1000 iterations and learns extremely slow with a learning rate of only 0.01

The training error for this model is 108.2119. The most influential feature is the departure delay. Other important features include actual time elapsed, distance, carrier, origin and destination. The estimated risk via this model is 108.4275.

```
> out.boost<- gbm(Y_train~.,data=X_train,
+                 distribution="gaussian",
+                 n.trees=1000,
+                 shrinkage=0.01,
+                 interaction.depth=3,
```

```
+              bag.fraction = 0.5,
+              n.minobsinnode = 10,
+              cv.folds = 3,
+              keep.data=TRUE,
+              verbose=TRUE,
+              n.cores=8)
Iter   TrainDeviance   ValidDeviance   StepSize   Improve
   900       113.8680             nan     0.0100    0.0519
   920       112.5226             nan     0.0100    0.0921
   940       111.4374             nan     0.0100    0.0756
   960       110.4612             nan     0.0100    0.0558
   980       109.3279             nan     0.0100    0.0505
  1000       108.2119             nan     0.0100    0.0838


> boost.sum<-summary(out.boost)
> boost.sum
                                    var     rel.inf
DEP_DELAY                     DEP_DELAY 96.45393413
ACTUAL_ELAPSED_TIME ACTUAL_ELAPSED_TIME  1.94958108
DISTANCE                       DISTANCE  1.29600429
CARRIER                         CARRIER  0.17870909
ORIGIN                           ORIGIN  0.08555602
DEST                               DEST  0.03621540
MONTH                             MONTH  0.00000000
DAY_OF_MONTH               DAY_OF_MONTH  0.00000000
DAY_OF_WEEK                 DAY_OF_WEEK  0.00000000
ORIGIN_AIRPORT_ID     ORIGIN_AIRPORT_ID  0.00000000
DEST_AIRPORT_ID         DEST_AIRPORT_ID  0.00000000
DEST_CITY_NAME           DEST_CITY_NAME  0.00000000
DEST_STATE_ABR           DEST_STATE_ABR  0.00000000
CRS_DEP_TIME               CRS_DEP_TIME  0.00000000
AIR_TIME                       AIR_TIME  0.00000000
> best.iter<- gbm.perf(out.boost,method = "cv")
> f.predict<- predict(out.boost,X_test,best.iter)
> print(mean((Y_test-f.predict)**2))
[1] 108.4275
> most.influential = which(names(X_train)%in%boost.sum[1:1,1])
> plot(out.boost,i.var=most.influential)
```



So from the three models, we can say that departure delay, actual time elapsed and distance are the three most significant features in predicting the delay time of the flight.