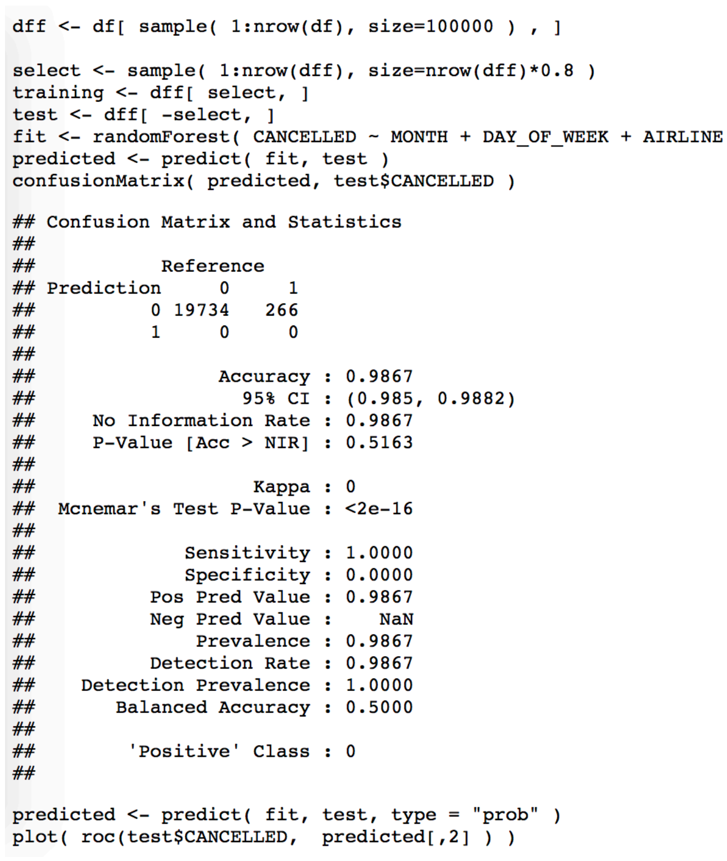
Technical challenge:

Using the Random Forest method for predictive regression modeling, our aim was to best predict the possibility of flight cancellations based on attributes found in the US Department of Transport, 2015 Flight Delays and Cancellations data.

Our approach analyzes information from millions of flights, creating a predictive model that will use past performance to indicate the likelihood of future on-time airline performance. It takes into account departures, arrivals, as well as the frequency of flight cancellations and the reasons why. We want to create a front-end application that will allow users to enter intended travel dates, providing them with a simple to read predictive airline performance score.

The dataset has nearly 6m lines, and both Python and R failed to make a model using the whole set. So we randomly chose 100,000 lines of data to build our RandomForest model. From this set, we used 80% of the data for training and 20% for test. To start, we used ‘months’, ‘day_of_week’ and ‘airline’ as our predictive variables.



```
dff <- df[ sample( 1:nrow(df), size=100000 ) , ]

select <- sample( 1:nrow(dff), size=nrow(dff)*0.8 )
training <- dff[ select, ]
test <- dff[ -select, ]
fit <- randomForest( CANCELLED ~ MONTH + DAY_OF_WEEK + AIRLINE
predicted <- predict( fit, test )
confusionMatrix( predicted, test$CANCELLED )

## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 19734   266
##          1     0     0
##
##                Accuracy : 0.9867
##                  95% CI : (0.985, 0.9882)
##     No Information Rate : 0.9867
##     P-Value [Acc > NIR] : 0.5163
##
##                   Kappa : 0
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.9867
##          Neg Pred Value :    NaN
##              Prevalence : 0.9867
##          Detection Rate : 0.9867
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##

predicted <- predict( fit, test, type = "prob" )
plot( roc(test$CANCELLED,  predicted[,2] ) )
```

The analysis above shows that our model has more than 99% accuracy. At first, it seems fantastic! We made the best model! But there’s a problem…

Out of nearly 6 million flights in 2015, only around 90,000 were cancelled (less than 1.5 % of all flights). This is a classic example of an *imbalanced dataset* (99:1 ratio of non-cancelled versus cancelled). The conventional model evaluation methods don't accurately measure model performance when faced with imbalanced datasets. While working in an imbalanced domain, accuracy is not an appropriate measure to evaluate model performance. The above model will always predict flights will not be cancelled. So we can't just make the model using all the data or smaller subset made by random selection of 100,000 rows.

In order to rectify the situation, we'll make a new dataset containing all the flights that were cancelled (~90k), plus the same number of flights that weren't cancelled(~90k, randomly selected). This new dataset has the same number of cancelled and non-cancelled flights and would be a better choice for making our predictive model.

Let's make our new dataset:

```
dff <- df[ df$CANCELLED == "1" ,  ]
select <- sample( 1:sum( df$CANCELLED == "0" ), size=nrow(dff) )
dff <- rbind( dff, df[ df$CANCELLED == "0" ,  ][ select , ] )

print( sum(dff$CANCELLED=="1") )

## [1] 89884

print( sum(dff$CANCELLED=="0") )

## [1] 89884
```

Like the last model, we use 80% of the new dataset for training and 20% for test:

```
select <- sample( 1:nrow(dff), size=nrow(dff)*0.8 )
training <- dff[ select, ]
test <- dff[ -select, ]
nrow(training)

## [1] 143814

nrow(test)

## [1] 35954
```

We can then use the same three variables as the last model to create our new model ('months', 'day_of_week' and 'airline'). We also calculate the importance of each of these variants in efficacy of the final model:

```
fit <- randomForest( CANCELLED ~ MONTH + DAY_OF_WEEK + AIRLINE, data = training )
imp <- importance(fit)
print(imp)

##             MeanDecreaseGini
## MONTH               5954.484
## DAY_OF_WEEK         1684.112
## AIRLINE             5777.530

predicted <- predict( fit, test, type = "prob" )
plot( roc( test$CANCELLED, predicted[,2] ) )


predicted <- predict( fit, test )
confusionMatrix( predicted, test$CANCELLED )

## Confusion Matrix and Statistics
##
##           Reference
## Prediction     0     1
##          0 12123  4904
##          1  5836 13091
##
##                Accuracy : 0.7013
##                  95% CI : (0.6965, 0.706)
##     No Information Rate : 0.5005
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4025
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.6750
##             Specificity : 0.7275
##          Pos Pred Value : 0.7120
##          Neg Pred Value : 0.6917
##              Prevalence : 0.4995
##          Detection Rate : 0.3372
##    Detection Prevalence : 0.4736
##       Balanced Accuracy : 0.7013
##
##        'Positive' Class : 0
##
```

The ROC curve shows that this model gives us more logical results. The variable ‘month’ has the most effect (importance) on the final result.

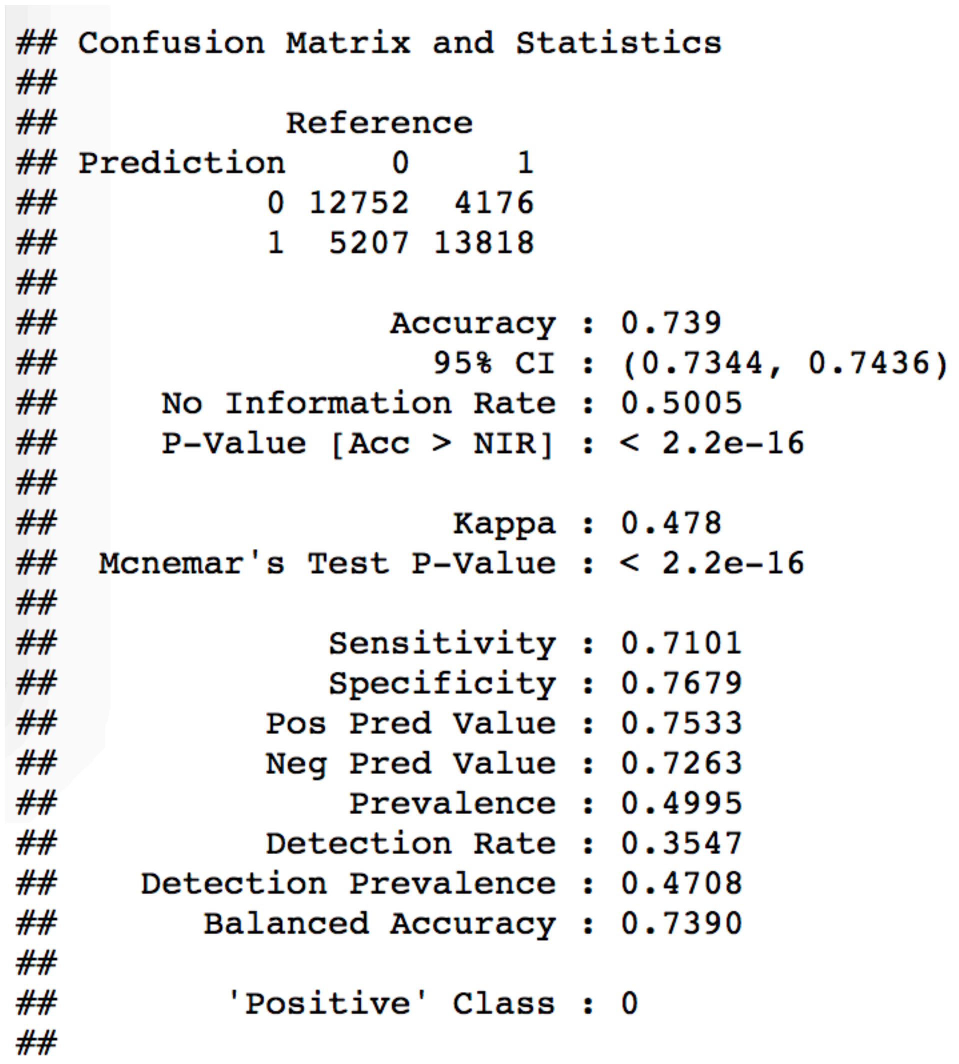
Let’s see if adding more variable can change the accuracy of our RandomForest model:

```
fit <- randomForest( CANCELLED ~ MONTH + DAY_OF_WEEK + AIRLINE + SCHEDULED_DEPARTURE + SCHEDULED_TI
imp <- importance(fit)
print(imp)

##                     MeanDecreaseGini
## MONTH                      10752.534
## DAY_OF_WEEK                 5856.910
## AIRLINE                     9630.864
## SCHEDULED_DEPARTURE        11249.373
## SCHEDULED_TIME             11004.241
## SCHEDULED_ARRIVAL          11668.894

predicted <- predict( fit, test[ !is.na( test$SCHEDULED_TIME ),  ], type = "prob" )
plot( roc( test$CANCELLED[ !is.na( test$SCHEDULED_TIME )  ], predicted[,2] ) )


predicted <- predict( fit, test[ !is.na( test$SCHEDULED_TIME ),  ] )
confusionMatrix( predicted, test$CANCELLED[ !is.na( test$SCHEDULED_TIME )  ] )
```



```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction     0     1
##          0 12752  4176
##          1  5207 13818
##
##                Accuracy : 0.739
##                  95% CI : (0.7344, 0.7436)
##     No Information Rate : 0.5005
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.478
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.7101
##             Specificity : 0.7679
##          Pos Pred Value : 0.7533
##          Neg Pred Value : 0.7263
##              Prevalence : 0.4995
##          Detection Rate : 0.3547
##    Detection Prevalence : 0.4708
##       Balanced Accuracy : 0.7390
##
##        'Positive' Class : 0
##
```

The accuracy is improved by a few percent. The new added variables have as much importance as the variable ‘months’ had in the previous model. The reason that the new model performs only slightly better than the last one might be because of the correlations between the new and old variables.

It was a fun process and we learned a lot. It’s interesting to realize that we’re dealing with a very imbalanced dataset, in which only 1% of the flights were cancelled. So, in order to make a model which could predict the cancellation risk, we had to play with the data to make it balanced. We have an accuracy of 74% for our model which is **very good** with the variables that we had to work with.