CSYE 7245 – Big Data Systems & Intelligence Analytics

Assignment No 01

Flight Delay & Cancellation Data 2015

Ninad Gadre

INDEX

ABSTRACT

In an ideal world, your travel scenario would go something like this: you show up at the airport, you check in and get your boarding passes, you get through security, you board your plane, and you take off. If you're an occasional traveler, perhaps every one of your flights has followed this pattern. But what happens when it doesn't?

There are certain events entirely beyond your control which could impact your flight. When flight delays and cancellations occur at airports, they are usually due to one of the nine following reasons. Here's a look at why these events affect your flight, and what you should do if they happen to you.

On occasion, delaying or canceling a flight is the only way an airline carrier can maintain high safety standards. In these challenging situations, simply knowing more about options a passenger has, is an important step toward getting the passengers travel plans back on track. When complications such as inclement weather, air traffic control problems or mechanical issues arise, the concern for passenger safety will always outweigh the desire to remain on schedule.

In this assignment we would try and analyze flight cancelation and delay data to provide suggestion for travel agencies in booking flights

INTRODUCTION

A **flight delay** is when an airline flight takes off and/or lands later than its scheduled time. The Federal Aviation Administration (FAA) considers a flight to be delayed when it is 30 minutes later than its scheduled time. A **cancellation** occurs when the airline does not operate the flight at all for a certain reason.

In the United States, when flights are canceled or delayed, passengers may be entitled to compensation due to rules obeyed by every flight company. This rule usually specifies that passengers may be entitled to certain reimbursements, including a free room if the next flight is the day after the canceled one, a choice of reimbursement, rerouting, phone calls, and refreshments. When a flight is delayed, the FAA allocates slots for takeoffs and landings based on which flight is scheduled first. The Transportation Department imposes a fine of up to $27,500 per passenger for planes left on the tarmac for more than three hours without taking off (four hours for international flights). In the United States, passengers are not entitled to compensation when a delay occurs, not even a cut of fees airlines must pay federal authorities for long delays. Airlines are required to pay for lodging costs of passengers if the delay or a cancellation is through their own fault, but not if the cause is beyond their control, such as weather.

Causes

Since 2003, the United States Bureau of Transportation Statistics has been keeping track of the causes of flight delays

Some of the causes of flight delays or cancellation are as follows:

- Maintenance problems with the aircraft
- Fueling
- Inclement weather, such as thunderstorm, hurricane, or blizzard
- Airline glitches. The top cause of flight delays, according to a USA TODAY analysis.
- Congestion in air traffic
- Late arrival of the aircraft to be used for the flight from a previous flight
- Security issues

Effects

Cost to airlines

In the United States, the Federal Aviation Administration estimates that flight delays cost airlines $22 billion yearly. Airlines are forced to pay federal authorities when they hold planes on the tarmac for more than three hours for domestic flights or more than four hours for international flights

Cost to passengers

Flight delays are an inconvenience to passengers. A delayed flight can be costly to passengers by making them late to their personal scheduled events and commitments. A passenger who is delayed on a multi-plane trip could miss a connecting flight. Anger and frustration can occur in delayed passengers.

CODE WITH DOCUMENTATION

PART A

#Import all the required libraries

import datetime, warnings, scipy

import pandas as pd

import numpy as np

import math

warnings.filterwarnings("ignore")

#we will create relative path to get data from csv files

import os

path = os.getcwd()+"\\flights.csv"

df = pd.read_csv(path, low_memory=False)

df.head()

airline_path = os.getcwd()+"\\airlines.csv"

airport_path = os.getcwd()+"\\airports.csv"

airlines_data = pd.read_csv(airline_path)

airports_data = pd.read_csv(airport_path)

airlines_data.head()

airports_data.head()

#We will make a copy of flights dataframe into other dataframe and reduce the size of data by slicing it

df_two = df.copy()

flight_data = df_two[:461502]

flight_data.head()

#Let us drop the columns that we don't require while applying algorithm

colsToDrop0 = ['CANCELLATION_REASON', 'AIR_SYSTEM_DELAY', 'SECURITY_DELAY', 'AIRLINE_DELAY',

'LATE_AIRCRAFT_DELAY', 'WEATHER_DELAY']

colsToDrop1 = ['TAXI_OUT', 'WHEELS_OFF', 'AIR_TIME', 'WHEELS_ON', 'TAXI_IN', 'ARRIVAL_TIME']

```
colsToDrop2 = ['FLIGHT_NUMBER', 'TAIL_NUMBER','YEAR']
df.drop(colsToDrop0, axis=1, inplace=True)
df.drop(colsToDrop1, axis=1, inplace=True)
df.drop(colsToDrop2, axis=1, inplace=True)
df.head()
def get_hour(x):
  try: # If departure time has >2 digits
    return int(str(x)[:-2])
  except: # if departure time has only <=2 digits: the hour is 0, time is 0:xx
    return 0

df['Hour']= df['DEPARTURE_TIME'].apply(get_hour)
#To get the hour from Departure Delay for analysis
def get_correctHour(x):
  if(len(str(x))==4):
    return int(str(x)[:2])
  elif(len(str(x))==3):
    return int(str(x)[:1])
  elif(len(str(x))==2 and x>23 and x<60):
    return 0
  elif(len(str(x))==2 and x>59 and x<90):
    return 1
  elif(len(str(x))==2 and x>89 and x<100):
    return 2
  else:
    return x

df['HOUR']= df['Hour'].apply(get_correctHour)
```

#There's an issue with the ORIGIN_AIRPORT and DESTINATION_AIRPORT features for October 2015. The problem is described in this post on Kaggle. The approach to fix it is drawn from this Kaggle kernel.

##In this section we will try and fix the discrepancy in the ORIGIN_AIRPORT AND DESTINATION_AIRPORT for the month of October

path2 = os.getcwd()+"\L_AIRPORT.csv" **#This is the lookup file provided by dat.gov website having 3-Letter codes for airport**

df_threeLetterCode = pd.read_csv(path2)

path3 = os.getcwd()+"\L_AIRPORT_ID.csv" **#This is the lookup file provided by dat.gov website having 5-digit codes for airport**

df_fiveDigitCode = pd.read_csv(path3)

```
codesToDrop = ['BSM','NYL']

df_threeLetterCode = df_threeLetterCode[~df_threeLetterCode['Code'].isin(codesToDrop)]

threeLetterCodes = list(df_threeLetterCode['Code'])

df2 = df_threeLetterCode.set_index('Description')

df3 = df_fiveDigitCode.set_index('Code')


df_airports = df[['ORIGIN_AIRPORT','DESTINATION_AIRPORT']]

df_October = df_airports.loc[~df_airports['ORIGIN_AIRPORT'].isin(threeLetterCodes) |

                    ~df_airports['DESTINATION_AIRPORT'].isin(threeLetterCodes)]


def fixOctoberAirports(airport):

    if len(airport) != 3:

        index = int(airport)

        descriptionAsKey = df3.loc[index]['Description']

        newCode = df2.loc[descriptionAsKey]['Code']

        return newCode

    else:

        return airport
```

```
print("Fixing Origin_Airport")
fixed_origin_airport = df['ORIGIN_AIRPORT'].apply(fixOctoberAirports)
fixed_origin_airport_fly = flight_data['ORIGIN_AIRPORT'].apply(fixOctoberAirports)

print("Fixing Dest_Airport")
fixed_dest_airport = df['DESTINATION_AIRPORT'].apply(fixOctoberAirports)
fixed_dest_airport_fly = flight_data['DESTINATION_AIRPORT'].apply(fixOctoberAirports)


df['ORIGIN_AIRPORT'] = fixed_origin_airport
df['DESTINATION_AIRPORT'] = fixed_dest_airport
flight_data['ORIGIN_AIRPORT'] = fixed_origin_airport_fly
flight_data['DESTINATION_AIRPORT'] = fixed_dest_airport_fly


df[df['MONTH']==10].head()
```

#General Info

```
number_of_delayed = flights_data["DEP_DELAY"].apply(lambda s: 1 if s!=0 else 0);
print("Total number of flights: "+str(len(flights_data)))
print("Number of cancelled flights: "+str(sum(flights_data["CANCELLED"])))
print("Number of delayed flights: "+str(sum(number_of_delayed)))


print("Number of not cancelled flights: "+str(len(flights_data)-sum(flights_data["CANCELLED"])))
print("Number of not delayed flights: "+str(len(flights_data)-sum(number_of_delayed)))
# print("The number of missing data: "+str(flights_data['DEPARTURE_TIME'].isnull().sum()));
```

print("Percentage of cancelled flights: "+str((sum(flights_data["CANCELLED"])*1.0/len(flights_data))*100)+"%")

print("Percentage of delayed flights: "+str((sum(number_of_delayed)*1.0/len(flights_data))*100)+"%")

#We will try and match the Origin Airport for all Carrier which will give us from where it operates using a lambda func

flight_data["AIRLINE_NAME"]=flight_data.apply(lambda x: airlines_data.loc[airlines_data['IATA_CODE'] == x["AIRLINE"],"AIRLINE"].values[0],axis=1)

flight_data[["AIRLINE_NAME","AIRLINE","ORIGIN_AIRPORT"]].head()

flight_data["ON_TIME"]=flight_data["ARRIVAL_DELAY"].apply(lambda row: 1 if row==0 else 0)

print(len(flight_data["AIRLINE_DELAY"]))

print("ON_TIME: "+str(flight_data["ON_TIME"].sum()))

missing_data_info={};

for column in flight_data.columns:

    missing_data_info[column]=flight_data[column].isnull().sum()

missing_data_info_sorted = sorted(missing_data_info.items(), key=operator.itemgetter(1))

missing_data_info_sorted

cancelled_flights = flight_data

grouped_cancelled_flights=cancelled_flights[["AIRLINE","AIRLINE_NAME","CANCELLED","ON_TIME"]].groupby(['AIRLINE','AIRLINE_NAME']).sum().reset_index()

grouped_cancelled_flights["FLIGHTS_COUNT"]=cancelled_flights[["AIRLINE","AIRLINE_NAME","ON_TIME"]].groupby(['AIRLINE','AIRLINE_NAME']).count().reset_index()["ON_TIME"]

grouped_cancelled_flights["CANCELLED_PERCENTAGE"]=grouped_cancelled_flights["CANCELLED"]*1.0/ grouped_cancelled_flights["FLIGHTS_COUNT"]*100

grouped_cancelled_flights["ON_TIME_PERCENTAGE"]=grouped_cancelled_flights["ON_TIME"]*1.0/grouped_cancelled_flights["FLIGHTS_COUNT"]*100

grouped_cancelled_flights[["AIRLINE","AIRLINE_NAME","FLIGHTS_COUNT","CANCELLED","ON_TIME","CANCELLED_PERCENTAGE","ON_TIME_PERCENTAGE"]].sort_values(by=['CANCELLED_PERCENTAGE'],ascending=[False])

airlines_data["FLIGHTS_COUNT"]=get_airline_information("FLIGHTS_COUNT",airlines_data,grouped_cancelled_flights)

airlines_data["ON_TIME"]=get_airline_information("ON_TIME",airlines_data,grouped_cancelled_flights)

airlines_data["ON_TIME_PERCENTAGE"]=get_airline_information("ON_TIME_PERCENTAGE",airlines_data,grouped_cancelled_flights) airlines_data.sort_values(by="ON_TIME_PERCENTAGE",ascending=False)

#Let us plot a pier chart which will give us the Carrier with most OnTIME performance

airlines_data["ON_TIME"].plot.pie(labels=airlines_data["AIRLINE"],autopct='%.2f', fontsize=20, figsize=(10, 10),colors=['r','g','b','w','y'])

#Mean delay for each airlines

airlines_data.sort_values(by=["MEAN_DEPARTURE_DELAY"],ascending=False).plot(x="AIRLINE",y="MEAN_DEPARTURE_DELAY",kind='bar')

#In tabular form for all the carrier with thier ON-Time performance and Mean Delay

airlines_data["MEAN_DEPARTURE_DELAY"]=get_airline_information("DEPARTURE_DELAY",airlines_data,positive_delayed_flight_grouped)

airlines_data[["AIRLINE","ON_TIME_PERCENTAGE","MEAN_DEPARTURE_DELAY"]].sort_values(by="MEAN_DEPARTURE_DELAY",ascending=True).head()

#Delay by Airlines

positive_delayed_flight=flight_data

positive_delayed_flight=positive_delayed_flight[positive_delayed_flight['DEPARTURE_DELAY']>=0]

positive_delayed_flight_grouped=positive_delayed_flight[["AIRLINE","AIRLINE_NAME","DEPARTURE_DELAY"]].groupby(["AIRLINE",'AIRLINE_NAME']).mean().reset_index()

#Ranking according to cancellation percentage

airlines_data["CANCELLED_PERCENTAGE"]=get_airline_information("CANCELLED_PERCENTAGE",airlines_data,grouped_cancelled_flights)

airlines_data.sort_values(by=["CANCELLED_PERCENTAGE"],ascending=False).plot(x="AIRLINE",y="CANCELLED_PERCENTAGE",kind='bar')

#Now we will try to clean the data to apply algorithn

missing_values = df.isnull().sum(axis=0)

missing_values

df['CANCELLED'].value_counts()

df['DIVERTED'].value_counts()

#Find num diverted or cancelled

print("Total number of flights diverted or cancelled: ", (89884 + 15187))

#Total number of flights diverted or cancelled: 105071 As you can see in the cells above, the number of missing values in the ARRIVAL_DELAY column directly matches the number of flights diverted/cancelled. There are no other missing values, so we'll proceed with our analysis.

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(12, 8))

df.loc[df['CANCELLED']==1 ,'ELAPSED_TIME'].plot.hist(bins= 30, ax=axes[0,0], alpha=.3, title= 'Scheduled Flight time in minutes')

df.loc[df['CANCELLED']==0 ,'ELAPSED_TIME'].plot.hist(bins= 30, ax=axes[0,0], alpha=.3)

df.loc[df['CANCELLED']==1 ,'DISTANCE'].plot.hist(bins= 30, ax=axes[0,1], alpha=.3, title= 'Distance in miles')

df.loc[df['CANCELLED']==0 ,'DISTANCE'].plot.hist(bins= 30, ax=axes[0,1], alpha=.3)

df.loc[df['CANCELLED']==1 ,'ARRIVAL_DELAY'].plot.hist(bins= 30, ax=axes[1,1], alpha=.3, title= 'Arrival delay in minutes')

df.loc[df['CANCELLED']==0 ,'ARRIVAL_DELAY'].plot.hist(bins= 30, ax=axes[1,1], alpha=.3)

df.loc[df['CANCELLED']==1 ,'DEPARTURE_DELAY'].plot.hist(bins= 40, ax=axes[1,0], alpha=.3, title= 'Departure delay in minutes')

df.loc[df['CANCELLED']==0 ,'DEPARTURE_DELAY'].plot.hist(bins= 40, ax=axes[1,0], alpha=.3)

#To compare cancelled and not cancelled flight we will compare using a line chart

plt.plot(cancel_by_month.iloc[0], label= 'Not Canceled')

plt.plot(cancel_by_month.iloc[1], label= 'Canceled')

plt.legend(loc=1)

plt.title('Frequency by month of the year')

PART B

#We will import all the libraries required to apply certain algorithms

import sklearn

from sklearn.pipeline import Pipeline

from sklearn import base

from sklearn.preprocessing import OneHotEncoder, LabelEncoder

```
from sklearn.model_selection import train_test_split, GridSearchCV

from sklearn.neighbors import KNeighborsRegressor

from sklearn.ensemble import RandomForestRegressor

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score, precision_score, recall_score
```

#We will drop the unnecessary columns

```
colsToDrop3 = ['DAY', 'DAY_OF_WEEK', 'ORIGIN_AIRPORT', 'DESTINATION_AIRPORT',
'SCHEDULED_DEPARTURE', 'SCHEDULED_ARRIVAL',

        'DIVERTED','SCHEDULED_DEPARTURE', 'DEPARTURE_DELAY', 'SCHEDULED_TIME']

df.drop(colsToDrop3, axis=1, inplace=True)

df.head()

#colsToDrop4 = ['Hour']

#df.drop(colsToDrop4, axis=1, inplace=True)

airline_List = ['AA', 'UA', 'B6', 'AS', 'WN', 'DL']

airline_List

df1 = df[df['AIRLINE'] == 'AA']

df2 = df[df['AIRLINE'] == 'UA']

df3 = df[df['AIRLINE'] == 'B6']

df4 = df[df['AIRLINE'] == 'AS']

df5 = df[df['AIRLINE'] == 'WN']

df6 = df[df['AIRLINE'] == 'DL']

frames = [df1,df2,df3,df4,df5,df6]

df_concat = pd.concat(frames)
```

#We will limit the data to top ailine carrier

```
#colsToDrop4 = ['Hour']

#df.drop(colsToDrop4, axis=1, inplace=True)

airline_List = ['AA', 'UA', 'B6', 'AS', 'WN', 'DL']

airline_List

df1 = df[df['AIRLINE'] == 'AA']
```

df2 = df[df['AIRLINE'] == 'UA']

df3 = df[df['AIRLINE'] == 'B6']

df4 = df[df['AIRLINE'] == 'AS']

df5 = df[df['AIRLINE'] == 'WN']

df6 = df[df['AIRLINE'] == 'DL']

frames = [df1,df2,df3,df4,df5,df6]

df_concat = pd.concat(frames)

#We will transform the rows into columns

CT = ColumnSelectTransformer(categorical_variables=['AIRLINE','MONTH', 'HOUR'],

numeric_variables=['DISTANCE'])

df_feature = CT.fit_transform(df_concat)

columns = list(df_feature.columns)

print(columns)

df_feature.head()

#We will use the train_test_split to split the data into test and train datasets

X = df_concat.drop('CANCELLED', axis=1)

y = df_concat['CANCELLED']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size= 1000, random_state=42)

#max_depth= 10, n_estimators = 10, max_features= 10

name -> (line format, classifier)

from sklearn.metrics import f1_score, confusion_matrix, accuracy_score

CLASS_MAP ={'Random Forest': (':', RandomForestClassifier(min_samples_split=20)),}

for name, (line_fmt, clf) in CLASS_MAP.items():

  # train model

  model = Pipeline([

    ('ColSelect', CT),# ColumnSelectTransformer

    ('clf', clf)   # classifier

```
    ])

    model.fit( X_train, y_train)
    # predict probability on test data
    preds = model.predict_proba(X_test)
    pred = pd.Series(preds[:, 1])

    # Calcualte FPR, TPR for plotting ROC curve
    fpr, tpr, thresholds = roc_curve(y_test, pred)
    auc_score = auc(fpr, tpr)

    train_pred = model.predict(X_train)
    test_pred = model.predict(X_test)

    print(name + ': Train Accuracy', accuracy_score(y_train, train_pred) , #sum(train_pred==
y_train)/len(y_train),
          ', Test Accuracy', accuracy_score(y_test, test_pred))

    #print('Confusion matrix of train data:')
    #print(confusion_matrix(y_train, train_pred))
    print('Confusion matrix of test data:')
    print(confusion_matrix(y_test, test_pred))
from sklearn.pipeline import Pipeline
CT = ColumnSelectTransformer(categorical_variables=['AIRLINE','MONTH', 'HOUR'],
                             numeric_variables=['DISTANCE'])
#clf = LogisticRegression()
clf = RandomForestClassifier(min_samples_split=20)
pipe = Pipeline([
        ('ColSelect', CT ),# ColumnSelectTransformer
```

```
    ('clf', clf)      # classifier
  ])
```

```
pipe.fit(df_concat, df_concat['CANCELLED'])
pipe.named_steps['clf'].feature_importances_
```

 # Randomly sample 10 rows from data for test prediction

```
test_num = 10
test = df_concat.sample(test_num )
test.head()
```

#To graphically represent our result we will plot it on a graph

```
plt.plot( m['CANCELLED']['mean']['AA'], label='AA')
plt.plot( m['CANCELLED']['mean']['AS'], label='AS')
plt.plot( m['CANCELLED']['mean']['B6'], label='B6')
plt.plot( m['CANCELLED']['mean']['DL'], label='DL')
plt.plot( m['CANCELLED']['mean']['UA'], label='UA')
plt.plot( m['CANCELLED']['mean']['WN'], label='WN')
plt.legend(loc=1)
plt.title('Proportion delayed for different months by airline')
```

RESULTS

1.

```
df.head()
```

	MONTH	DAY	DAY_OF_WEEK	AIRLINE	ORIGIN_AIRPORT	DESTINATION_AIRPORT	SCHEDULED_DEPARTURE	DEPARTURE_TIME	DEPARTI
0	1	1	4	AS	ANC	SEA	5	2354.0	-11.0
1	1	1	4	AA	LAX	PBI	10	2.0	-8.0
2	1	1	4	US	SFO	CLT	20	18.0	-2.0
3	1	1	4	AA	LAX	MIA	20	15.0	-5.0
4	1	1	4	AS	SEA	ANC	25	24.0	-1.0

There's an issue with the ORIGIN_AIRPORT and DESTINATION_AIRPORT features for October 2015. The problem is described in this post on Kaggle. The approach to fix it is drawn from this Kaggle kernel.

```
df[df['MONTH']==10].head()
```

	MONTH	DAY	DAY_OF_WEEK	AIRLINE	ORIGIN_AIRPORT	DESTINATION_AIRPORT	SCHEDULED_DEPARTURE	DEPARTURE_TIME	DI
4385712	10	1	4	AA	14747	11298	5	15.0	1(
4385713	10	1	4	DL	14771	13487	5	16.0	11
4385714	10	1	4	NK	12889	13487	5	2400.0	-5
4385715	10	1	4	AA	12892	13303	10	7.0	-3
4385716	10	1	4	AA	14771	11057	10	8.0	-2

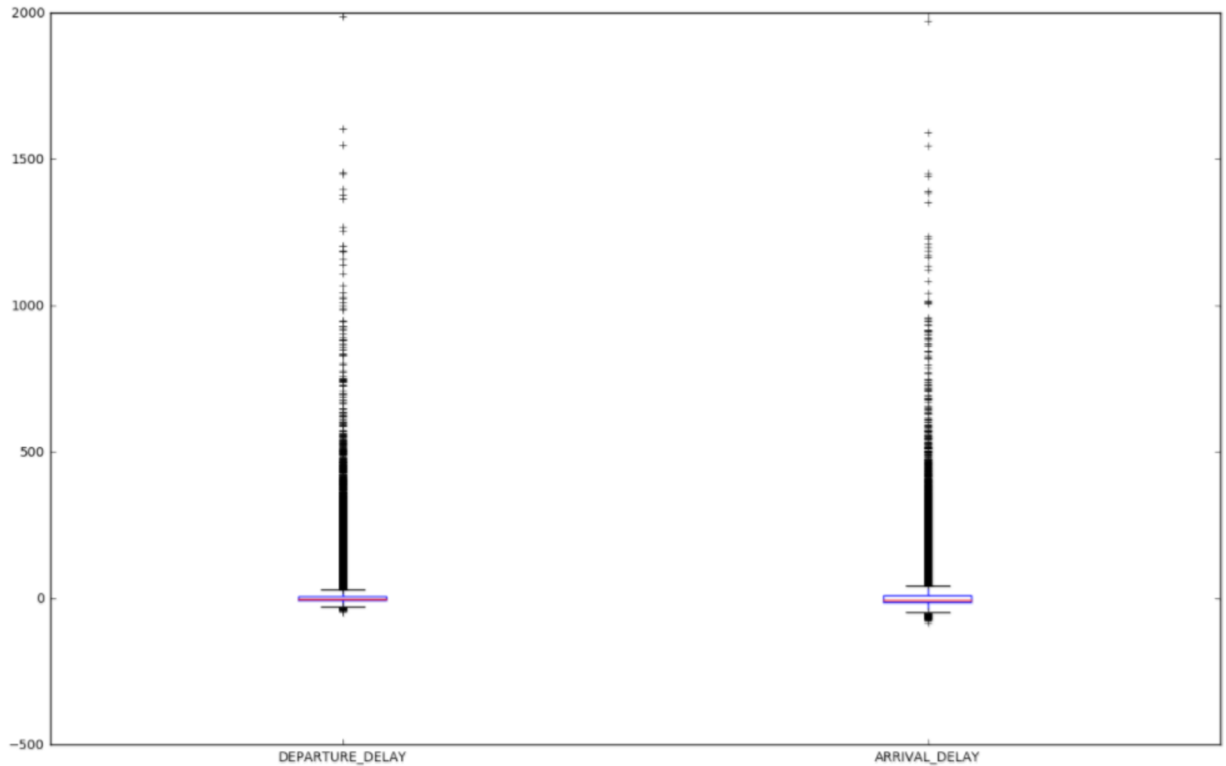
As from above picture we can observe that for the month of October the ORIGIN_AIRPORT & DESTINATION_AIRPORT have 5-digit codes, after converting we get the following result

```
df[df['MONTH']==10].head()
```

```
Fixing Origin_Airport
Fixing Dest_Airport
```

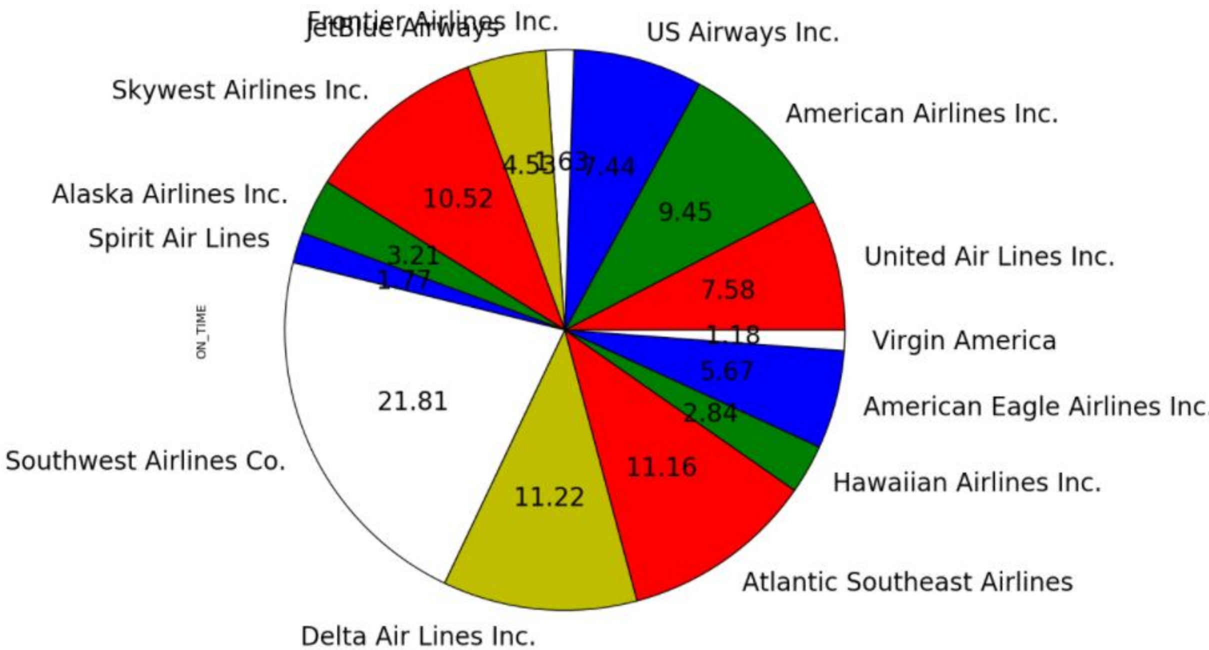
	MONTH	DAY	DAY_OF_WEEK	AIRLINE	ORIGIN_AIRPORT	DESTINATION_AIRPORT	SCHEDULED_DEPARTURE	DEPARTURE_TIME	DI
4385712	10	1	4	AA	SEA	DFW	5	15.0	1(
4385713	10	1	4	DL	SFO	MSP	5	16.0	11
4385714	10	1	4	NK	LAS	MSP	5	2400.0	-5
4385715	10	1	4	AA	LAX	MIA	10	7.0	-3
4385716	10	1	4	AA	SFO	CLT	10	8.0	-2

2.



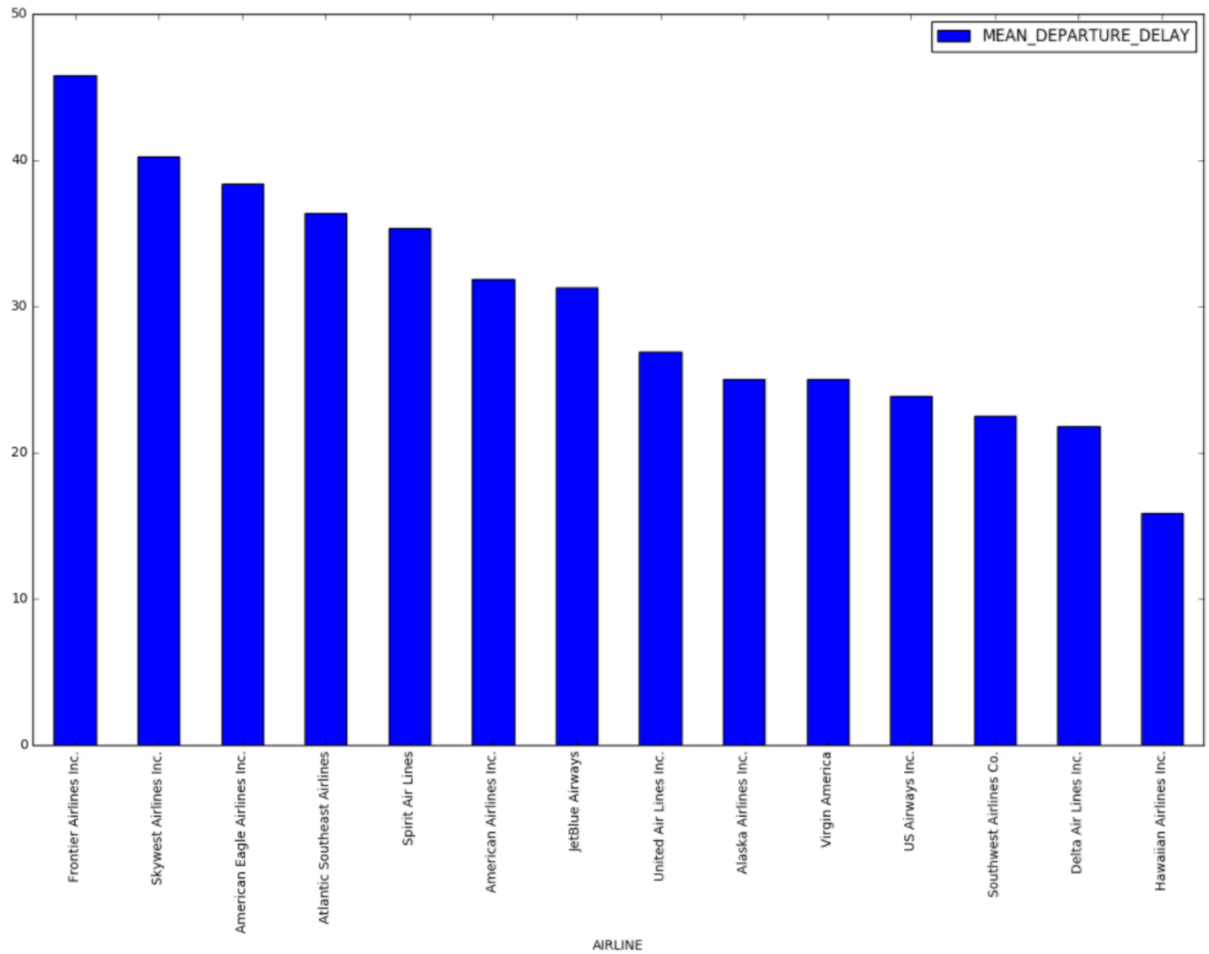
We notice from the previous plot that there are some negative values and that means there are some flights took off before few minutes before the exact time. We are going to call that flights ahead_flights and the other one delayed_flights

3.



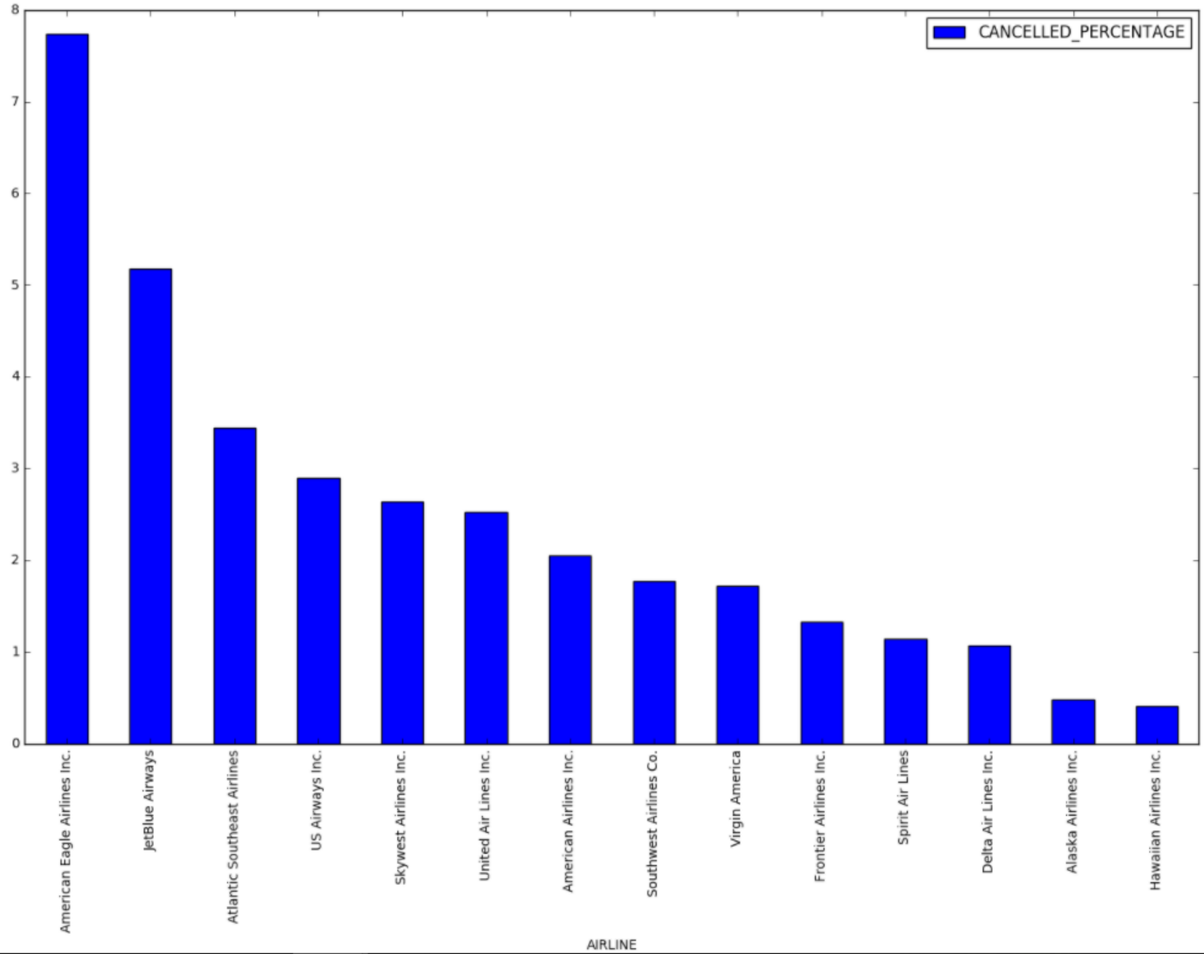
From the above result we can say that Southwest Airlines has the best ONTIME Performance

4.



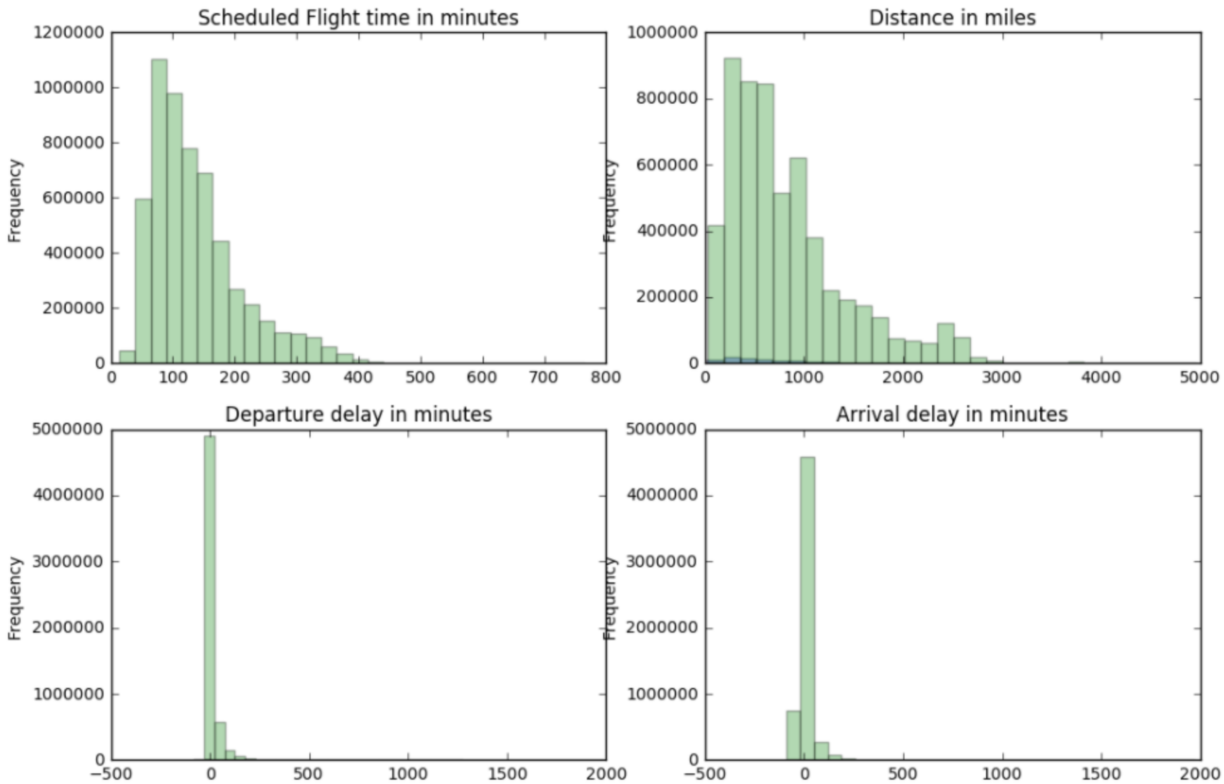
From the above graph we can say that Frontier Airlines has the highest amount of flights that departed late in 2015

5.



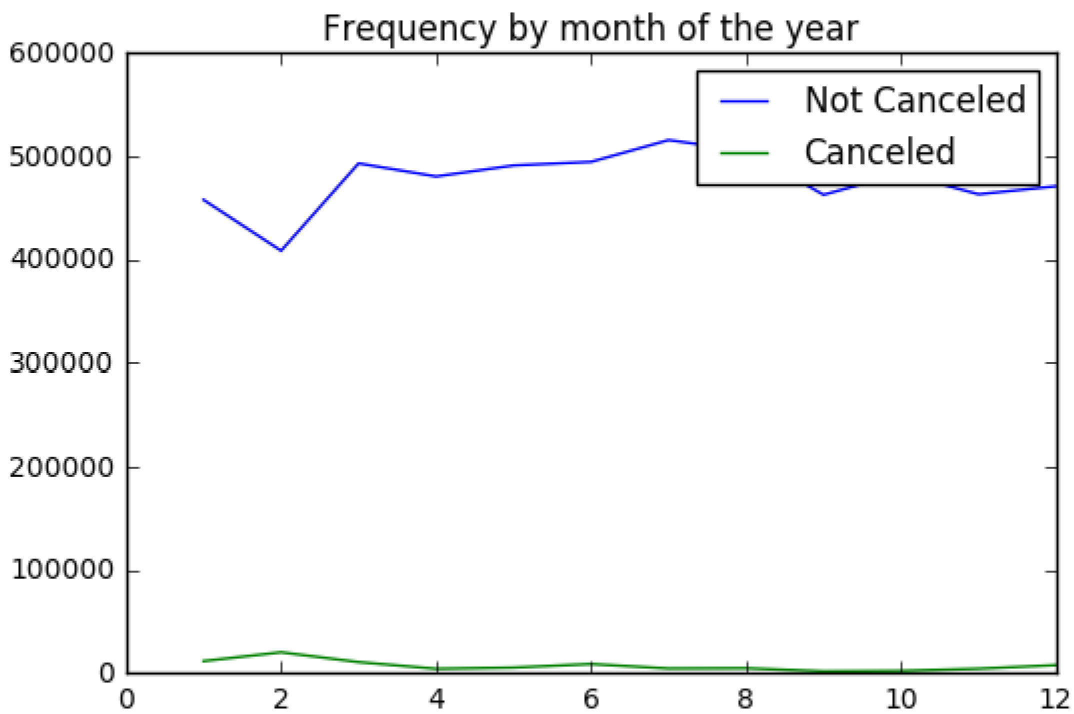
The American Eagle Airlines has highest percentage pf flight cancellation in 2015 while Alaska Airlines and Hawaiian Airlines has least amount of cancelled flights in 2015

6.



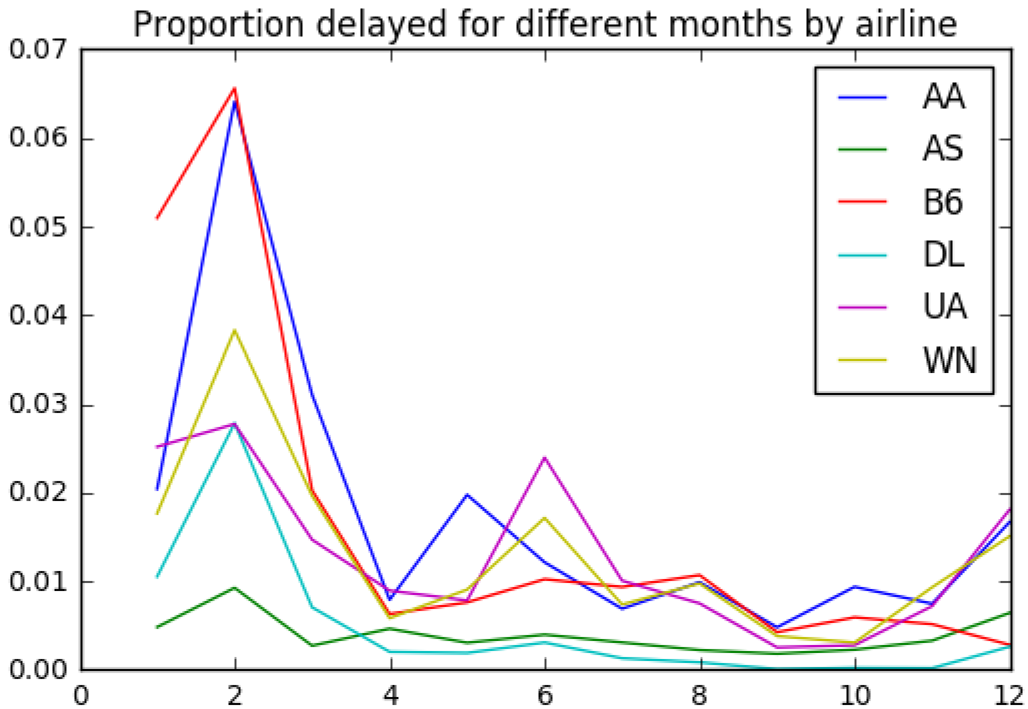
Scheduled Flight time in minutes
Frequency
Distance in miles
Frequency
Departure delay in minutes
Frequency
Arrival delay in minutes
Frequency

7.



The above graph gives us the comparison of Cancelled and Not Cancelled flights in 2015 according to every month

8.



After applying Random Forest Classifier Algorithm, we can predict that Alaska Airlines have least amount of cancellations and delay throughout 2015.

DISCUSSIONS

According to the Exploratory Analysis, we can conclude that Alaska Airline has the least amount of delays and cancellations throughout 2015.

Why Random Forest Algorithm was used?

When fitting a classifier to training data, an important concern is to avoid *overfitting*, in order to keep the generalization error under control. This is a measure of the accuracy of the classifier when applied to previously unseen data. In the case of a random forest for example, parameters specify what happens at each node of each decision tree, whereas hyper-parameters specify the number of trees and the maximum depth of a tree. I used scikit learn's random forest classifier to build my prediction model along with other packages to assist in evaluating and cross-validating my results.

REFERENCES

https://www.kaggle.com/usdot/flight-delays

https://www.transportation.gov/

http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/