INFO I 590 - Big Data Applications and Analytics

Performance Analysis of Airline Carriers based on their delay pattern

Name	email	GitHub name
Rashmi Rayala	rashraya@umail.iu.edu	rashraya
Paventhan Vivekanandan	pvivekan@umail.iu.edu	pvivekan

Project Goals and Objectives:

1] To analyse the delays in departures of airline carriers and to provide the passengers with delay percentage of different carriers at the time of booking tickets.

2] To build different predictive models (Logistic Regression, RandomForestClassifier, GaussianNB, VotingClassifier, AdaBoostClassifier, GradientBoostingClassifier, ExtraTreesClassifier) using Python's Scikit-learn machine learning package and to compare their performance.

Tasks performed to complete project deliverables:

1] Pre-processing of raw dataset
2] Reducing the dataset using Pig
3] Classification of delayed airline records using Python's Scikit-learn machine learning packages
4] Performance comparison of different Scikit-learn machine learning packages
5] Calculation of different airline delays in percentage

Techniques used and technical difficulties resolved:

Preprocessing of raw dataset:

The raw dataset consists of over 5 GB of airline data. It has around 33 attributes. Most of the fields in the original dataset are double quoted. But we faced difficulties in processing the double quoted fields while using Pig. The pig script didn't recognize double quoted fields and also caused problems in processing field's separated by commas. For example, the pig script recognized the field "Los Angeles, LA" as two separate fields namely "Los Angeles" and "LA". So we removed the double quotes. We also replaced the fields "Los Angeles, LA" to 'Los Angeles LA'.

Reducing the dataset using Pig:

The dataset 1 contains airline delay records from 2011 to 2015. The dataset 2 contains weather information from 2011 to 2015. We chose only the flight records departing from New York (JFK) and reaching Los Angeles (LAX). We also filtered the records of flights that were cancelled. Pig script is used to reduce the original dataset and also to join the two datasets (airline delay and weather dataset). We also created one more sub dataset from the original dataset having New York (JFK) as origin and Chicago (ORD) as destination for analysing the impact of snow (at destination) in classification of delay records. After reducing the dataset, the size decreased from 5 GB to 10 MB.

Classification of delayed airline records using Python's Scikit-learn machine learning packages:

We used the following Scikit-learn packages for classification of delay records.

1] LogisticRegression
2] RandomForestClassifier
3] AdaBoostClassifier
4] VotingClassifier
5] GradientBoostingClassifier
6] ExtraTreesClassifier

We compared the performance of these algorithms in our reduced dataset having JFK as origin and LAX as destination.

Calculation of different airline delays in percentage:

We focused on the following 5 carriers to analyse their performance based on departure delay.

1] American Airlines [AA]
2] Delta Airlines [DL]
3] JetBlue Airlines [B6]
4] United Airlines [UA]
5] Virgin America [VX]

Results:

We generated the following parameters:

1] Confusion matrix
2] Precision
3] Recall
4] F1 score
5] Accuracy

Confusion matrix generation

The confusion matrix is presented as follows:

	0	1
0	True Negative (TN)	False Positive (FP)
1	False Negative (FN)	True Positive (TP)

Precision

The precision is computed using precision_recall_fscore_support module.
Precision = TP / (TP + FP)

Recall

The recall is computed using precision_recall_fscore_support module.
Recall = TP / (TP + FN)

F1 Score

The F1 measure is computed using precision_recall_fscore_support module.
F1 = 2TP / (2TP + FP + FN)

Accuracy

The accuracy is computed using accuracy_score module.
Accuracy = (TP + TN) / (TP + FP + TN + FN)

Scikit-learn Algorithms

We generated the above parameters for all the six algorithms with three datasets as follows.

Case 1: Dataset 1 (without the weather data)
Case 2: Dataset 1 (without the weather data and with categorical data converted to binary values using OneHotEncoder)
Case 3: Dataset 1 and 2 combined (with weather information for origin (JFK) added)
Case 4: Dataset 1 and 2 combined (with weather information for origin (JFK) and destination (LAX) added).

The program execution results are presented as follows:

LogisticRegression:

Case 1

```
<-------  LogisticRegression -------->
Confusion matrix:
      0     1
0  4374  3237
1   474   955

[-] Precision = 0.23
[-] Recall = 0.67
[-] F1 score = 0.34
[-] Accuracy = 0.59
<------------------------------------>
```

Case 2

```
<-------  LogisticRegression -------->
Confusion matrix:
      0     1
0  4620  2991
1   547   882

[-] Precision = 0.23
[-] Recall = 0.62
[-] F1 score = 0.33
[-] Accuracy = 0.61
<------------------------------------>
```

Case 3

```
<-------  LogisticRegression -------->
Confusion matrix:
      0     1
0  4779  2832
1   562   867

[-] Precision = 0.23
[-] Recall = 0.61
[-] F1 score = 0.34
[-] Accuracy = 0.62
<------------------------------------>
```

Case 4

```
<-------  LogisticRegression -------->
Confusion matrix:
      0     1
0  4719  2767
1   523   881

[-] Precision = 0.24
[-] Recall = 0.63
[-] F1 score = 0.35
[-] Accuracy = 0.63
<------------------------------------>
```

Here we can see the improvement in accuracy for higher cases.

RandomForestClassifier:

Case 1

```
<-------  RandomForestClassifier -------->
Confusion matrix:
      0    1
0  7007  604
1  1202  227

[-] Precision = 0.27
[-] Recall = 0.16
[-] F1 score = 0.20
[-] Accuracy = 0.80
<---------------------------------------->
```

Case 2

```
<-------  RandomForestClassifier -------->
Confusion matrix:
      0    1
0  7372  239
1  1318  111

[-] Precision = 0.32
[-] Recall = 0.08
[-] F1 score = 0.12
[-] Accuracy = 0.83
<---------------------------------------->
```

Case 3

```
<-------  RandomForestClassifier -------->
Confusion matrix:
      0    1
0  7406  205
1  1314  115

[-] Precision = 0.36
[-] Recall = 0.08
[-] F1 score = 0.13
[-] Accuracy = 0.83
<------------------------------------->
```

Case 4

```
<-------  RandomForestClassifier -------->
Confusion matrix:
      0    1
0  7313  173
1  1316   88

[-] Precision = 0.34
[-] Recall = 0.06
[-] F1 score = 0.11
[-] Accuracy = 0.83
<------------------------------------->
```

For this model the accuracy remains the same for higher cases but improvement in precision can be observed. A decrease in recall and F1 measurement for higher cases can also be seen.

<u>AdaBoostClassifier:</u>

Case 1

```
<-------  AdaBoostClassifier -------->
Confusion matrix:
      0    1
0  7524  87
1  1356  73

[-] Precision = 0.46
[-] Recall = 0.05
[-] F1 score = 0.09
[-] Accuracy = 0.84
<------------------------------------>
```

Case 2

```
<-------  AdaBoostClassifier -------->
Confusion matrix:
      0    1
0  7502  109
1  1351   78

[-] Precision = 0.42
[-] Recall = 0.05
[-] F1 score = 0.10
[-] Accuracy = 0.84
<------------------------------------>
```

Case 3

```
<-------  AdaBoostClassifier -------->
Confusion matrix:
      0    1
0  7423  188
1  1290  139

[-] Precision = 0.43
[-] Recall = 0.10
[-] F1 score = 0.16
[-] Accuracy = 0.84
<------------------------------------>
```

Case 4

```
<-------  AdaBoostClassifier -------->
Confusion matrix:
      0    1
0  7215  271
1  1230  174

[-] Precision = 0.39
[-] Recall = 0.12
[-] F1 score = 0.19
[-] Accuracy = 0.83
<------------------------------------>
```

In this model the accuracy almost remains the same for all the cases. The precision is observed to be decreasing and the recall and F1 score is observed to be increasing with higher cases.

<u>VotingClassifier:</u>

Case 1

```
<-------  VotingClassifier -------->
Confusion matrix:
      0    1
0  7603    8
1  1418   11

[-] Precision = 0.58
[-] Recall = 0.01
[-] F1 score = 0.02
[-] Accuracy = 0.84
<---------------------------------->
```

Case 2

```
<-------  VotingClassifier -------->
Confusion matrix:
      0    1
0  7357  254
1  1283  146

[-] Precision = 0.36
[-] Recall = 0.10
[-] F1 score = 0.16
[-] Accuracy = 0.83
<---------------------------------->
```

Case 3

```
<-------  VotingClassifier -------->
Confusion matrix:
      0    1
0  7362  249
1  1237  192

[-] Precision = 0.44
[-] Recall = 0.13
[-] F1 score = 0.21
[-] Accuracy = 0.84
<---------------------------------->
```

Case 4

```
<-------  VotingClassifier -------->
Confusion matrix:
      0    1
0  7220  266
1  1224  180

[-] Precision = 0.40
[-] Recall = 0.13
[-] F1 score = 0.19
[-] Accuracy = 0.83
<---------------------------------->
```

The accuracy remains almost same for this model. The precision decreases and recall and F1 score increases for higher cases.

ExtraTreesClassifier:

Case 1

```
<-------  ExtraTreesClassifier -------->
Confusion matrix:
      0    1
0  7003  608
1  1195  234

[-] Precision = 0.28
[-] Recall = 0.16
[-] F1 score = 0.21
[-] Accuracy = 0.80
<-------------------------------------->
```

Case 2

```
<-------  ExtraTreesClassifier -------->
Confusion matrix:
      0    1
0  7221  390
1  1286  143

[-] Precision = 0.27
[-] Recall = 0.10
[-] F1 score = 0.15
[-] Accuracy = 0.81
<-------------------------------------->
```

Case 3

```
<-------  ExtraTreesClassifier -------->
Confusion matrix:
      0    1
0  7380  231
1  1315  114

[-] Precision = 0.33
[-] Recall = 0.08
[-] F1 score = 0.13
[-] Accuracy = 0.83
<-------------------------------------->
```

Case 4

```
<-------  ExtraTreesClassifier -------->
Confusion matrix:
      0    1
0  7234  252
1  1296  108

[-] Precision = 0.30
[-] Recall = 0.08
[-] F1 score = 0.12
[-] Accuracy = 0.83
<-------------------------------------->
```

For this model improvement in accuracy can be seen for higher cases. Precision is high for the dataset containing weather information for origin. Recall and F1 score is high for case 1 (dataset 1).

GradientBoostingClassifier:

Case 1

```
<-------  GradientBoostingClassifier -------->
Confusion matrix:
      0    1
0  7526  85
1  1351  78

[-] Precision = 0.48
[-] Recall = 0.05
[-] F1 score = 0.10
[-] Accuracy = 0.84
<-------------------------------------------->
```

Case 2

```
<-------  GradientBoostingClassifier -------->
Confusion matrix:
      0    1
0  7510  101
1  1353   76

[-] Precision = 0.43
[-] Recall = 0.05
[-] F1 score = 0.09
[-] Accuracy = 0.84
<-------------------------------------------->
```

Case 3

```
<-------  GradientBoostingClassifier -------->
Confusion matrix:
      0    1
0  7368  243
1  1271  158

[-] Precision = 0.39
[-] Recall = 0.11
[-] F1 score = 0.17
[-] Accuracy = 0.83
<-------------------------------------------->
```

Case 4

```
<-------  GradientBoostingClassifier -------->
Confusion matrix:
      0    1
0  7234  252
1  1241  163

[-] Precision = 0.39
[-] Recall = 0.12
[-] F1 score = 0.18
[-] Accuracy = 0.83
<-------------------------------------------->
```

In this model, the accuracy is almost same but precision is decreasing. The recall and F1 score are increasing for higher cases.

Delay Percentage computation for different airlines:

Here we have taken five carriers (American Airlines, Delta Airlines, JetBlue Airlines, United Airlines, Virgin America) for which we are computing the delay percentage. The dataset includes delay records from 2011 to 2015. The route we are considering for our case is from New York (JFK) to Los Angeles (LAX) airports.

The ticket fare for the five carriers (for route New York (JFK) to Los Angeles (LAX)) were taken manually from "http://www.cheapflights.com".

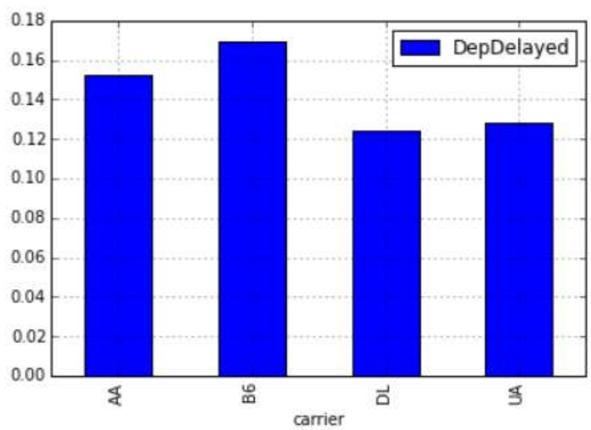
The execution result is as follows:

```
<================ Flight Details for JFK (New York) to LAX (Los Angeles) =================>

>>>>>>>>>>>> American Airlines Delay Percent >>>>>>>>>>>>>
Percent of AA carriers delayed in 2011 = 15.2027867629%
Percent of AA carriers delayed in 2012 = 15.4520713638%
Percent of AA carriers delayed in 2013 = 16.5571205008%
Percent of AA carriers delayed in 2014 = 13.1234866828%
Percent of AA carriers delayed in 2015 = 12.9194630872%

Delayed AA Carriers (2011-2015) = 2578
Total AA Carriers (2011-2015) = 17631
Percent of AA carriers delayed in (2011-2015) = 14.6219726618%

Average ticket fare for AA [JFK -> LAX (Quickest)] = 589$

>>>>>>>>>>>> Delta Airlines Delay Percent >>>>>>>>>>>>>
Percent of DL carriers delayed in 2011 = 12.3960968558%
Percent of DL carriers delayed in 2012 = 10.5799373041%
Percent of DL carriers delayed in 2013 = 14.0449438202%
Percent of DL carriers delayed in 2014 = 21.0922787194%
Percent of DL carriers delayed in 2015 = 29.7552836485%

Delayed DL Carriers (2011-2015) = 2058
Total DL Carriers (2011-2015) = 12264
Percent of DL carriers delayed in (2011-2015) = 16.7808219178%

Average ticket fare for DL [JFK -> LAX (Quickest)] = 588$

>>>>>>>>>>>> JetBlue Airlines Delay Percent >>>>>>>>>>>>>
Percent of B6 carriers delayed in 2011 = 16.934487021%
Percent of B6 carriers delayed in 2012 = 16.4242424242%
Percent of B6 carriers delayed in 2013 = 19.3107546049%
Percent of B6 carriers delayed in 2014 = 15.0468384075%
Percent of B6 carriers delayed in 2015 = 11.7111995452%

Delayed B6 Carriers (2011-2015) = 1333
Total B6 Carriers (2011-2015) = 8418
Percent of B6 carriers delayed in (2011-2015) = 15.8351152293%

Average ticket fare for B6 [JFK -> LAX (Quickest)] = 556$

>>>>>>>>>>>> United Airlines Delay Percent >>>>>>>>>>>>>
Percent of UA carriers delayed in 2011 = 12.8130217028%
Percent of UA carriers delayed in 2012 = 14.5320197044%
Percent of UA carriers delayed in 2013 = 15.6709108717%
Percent of UA carriers delayed in 2014 = 15.0813208477%
Percent of UA carriers delayed in 2015 = 12.6686656672%

Delayed UA Carriers (2011-2015) = 1397
Total UA Carriers (2011-2015) = 9831
Percent of UA carriers delayed in (2011-2015) = 14.2101515614%

Average ticket fare for UA [JFK -> LAX (Quickest)] = 546$

>>>>>>>>>>>> Virgin America Airlines Delay Percent >>>>>>>>>>>>>
Percent of VX carriers delayed in 2011 = No Records Found !!!%
Percent of VX carriers delayed in 2012 = 16.5967365967%
Percent of VX carriers delayed in 2013 = 18.0695847363%
Percent of VX carriers delayed in 2014 = 11.8984664199%
Percent of VX carriers delayed in 2015 = 11.4627887083%

Delayed VX Carriers (2011-2015) = 1037
Total VX Carriers (2011-2015) = 6987
Percent of VX carriers delayed in (2011-2015) = 14.8418491484%

Average ticket fare for VX [JFK -> LAX (Quickest)] = 546$
```

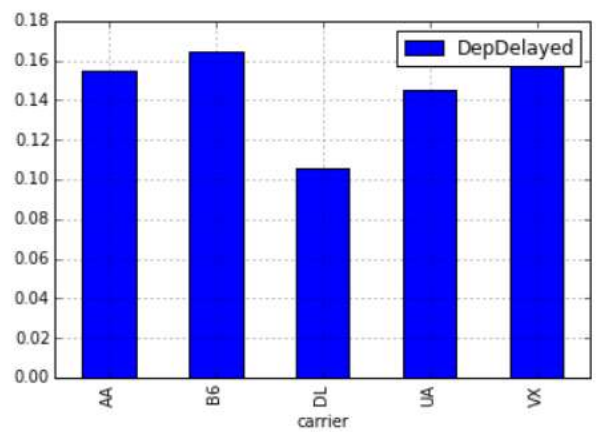
From the above results we can infer that Virgin America Airlines is the best to travel as it has least delay percentage and the price of the ticket is also reasonable. If we provide this kind of useful data for passengers while they are booking tickets, it would be very helpful for them. If someone has to travel in an emergency condition they will choose Virgin America Airways as the delay percentage is very little and the price of ticket is also reasonable.

<u>Graph representation:</u>

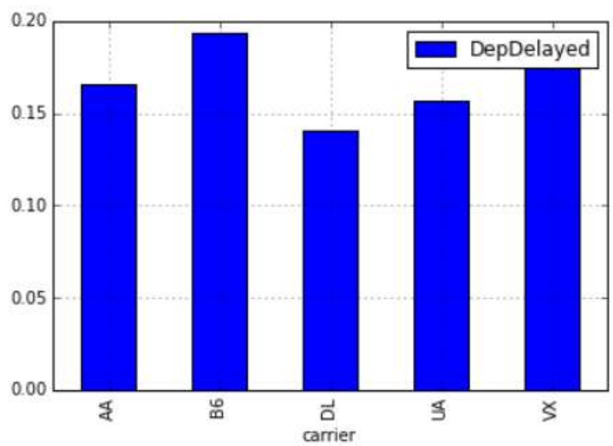
Delay by carrier (2011)



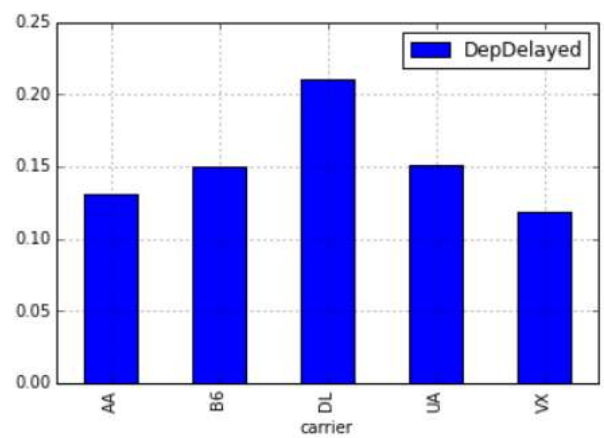
Delay by carrier (2012)



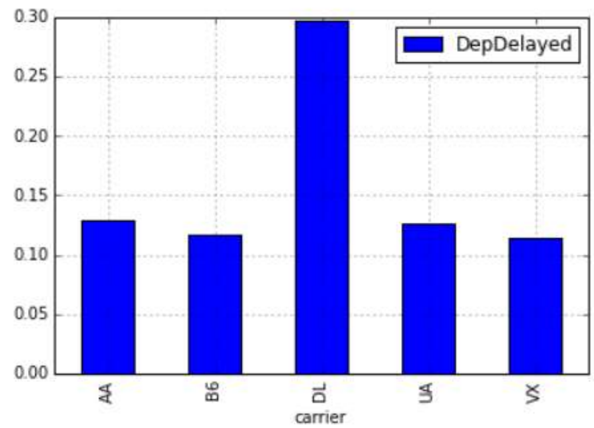
Delay by carrier (2013)



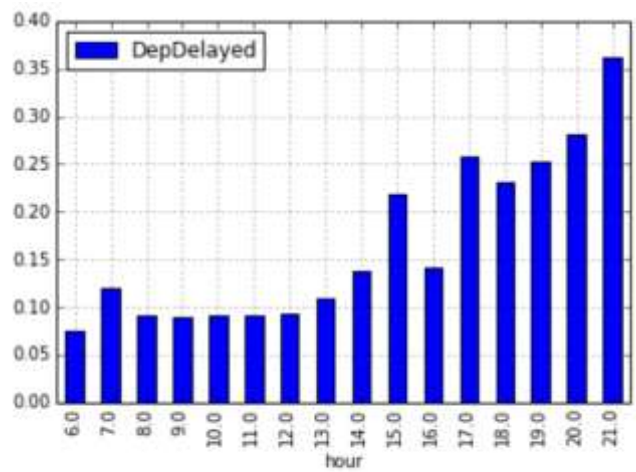
Delay by carrier (2014)



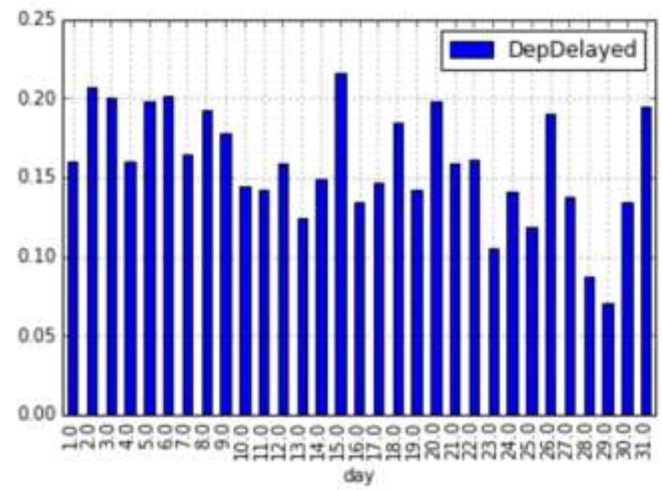
Delay by carrier (2015)



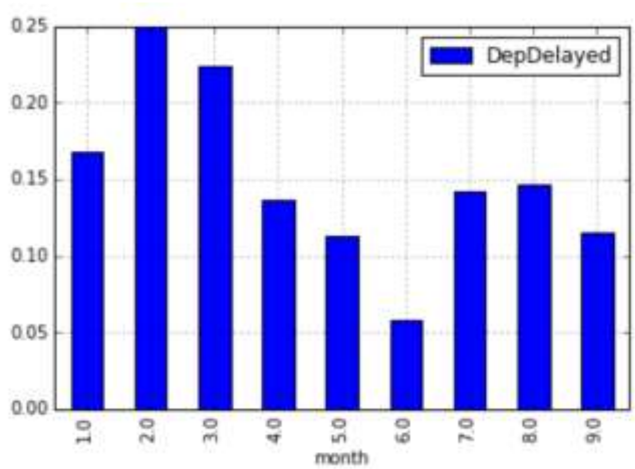
Delay by hour (2015)



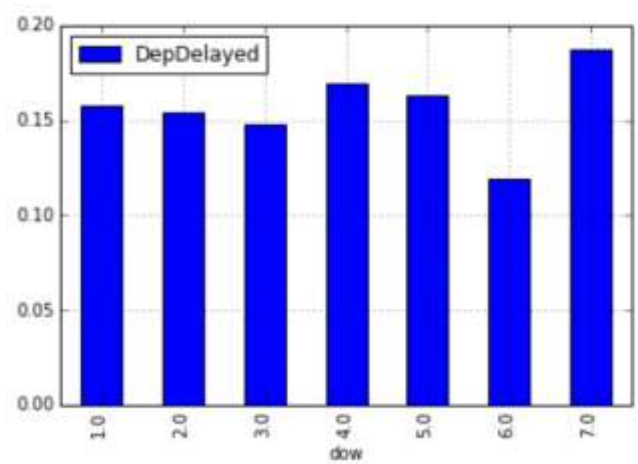
Delay by day (2015)



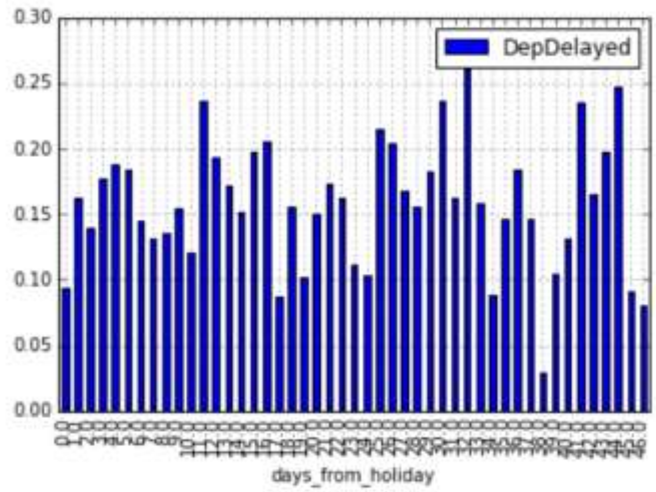
Delay by month (2015)



Delay by day of week (2015)



Delay by days from holiday (2015)



<u>Recommendations for future improvements:</u>

This software can be improved in lot of ways as follows:

1] The accuracy can be improved further by adding or removing any other attributes from the original dataset.

2] Ticket fare dataset can be obtained and merged with the reduced dataset and the fluctuation in the ticket fares can be compared against the performance of different airlines.

3] Passengers travelling long distance will be affected adversely when there is any delay because that will increase the possibility of them missing a connecting flight. Longer routes (including international routes) can be merged into the dataset and the connecting flights can be taken into account and more options can be provided to the passengers at the time of booking.

<u>Reproducibility:</u>

The reported results are reproducible.

<u>Development environment:</u>

OS Version – Ubuntu 15.10 (Wily Werewolf)
Kernel Version – 4.2.0-16-generic
Architecture – AMD 64-bit
Browser used (for Ipython Notebook) – Mozilla Firefox 41.0.2
Python 2.7.10
Apache Pig version 0.15.0
Ipython Notebook 2.3.0
Scikit-learn packages
Hadoop
Ultraedit text editor

Instructions to run the software:

The code is available in GitHub (https://github.iu.edu/pvivekan/Flight_Delay_Prediction)
We have created separate directories for all the functionalities described. For example the delay percentage computation is pushed as separate directory and the delay prediction using scikit-learn packages are pushed as six different directories. To run a particular module (or directory) just copy and paste the code (in order) in the Ipython Notebook executing on the Firefox web browser (or any other compatible browser). We have also uploaded the results to the GitHub repository.

Further instructions are available in the Readme file in GitHub.

Datasets:

Dataset 1:
Airline data (2011-2015)
http://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp

YEAR	2011 - 2015
MONTH	1 - 12
DAY_OF_MONTH	1 - 31
DAY_OF_WEEK	1 - 7
UNIQUE_CARRIER	Unique carrier number of flight
TAIL_NUM	Tail number of plane
FL_NUM	Flight number
ORIGIN_AIRPORT_ID	Airport id of origin city
ORIGIN	Origin city code
ORIGIN_CITY_NAME	Origin city name
DEST_AIRPORT_ID	Airport id of destination city
DEST	Destination city code
DEST_CITY_NAME	Destination city name
CRS_DEP_TIME	Scheduled arrival time (local, hhmm)
DEP_TIME	actual departure time (local, hhmm)
DEP_DELAY	departure delay, in minutes
TAXI_OUT	Taxi out time
TAXI_IN	Taxi in time
CRS_ARR_TIME	scheduled arrival time (local, hhmm)
ARR_TIME	actual arrival time (local, hhmm)
ARR_DELAY	arrival delay, in minutes
CANCELLED	Was the flight cancelled or not?
CANCELLATION_CODE	reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
DIVERTED	diverted = 1, not diverted =0
CRS_ELAPSED_TIME	Scheduled elapsed time
ACTUAL_ELAPSED_TIME	Actual elapsed time
AIR_TIME	Time of flight in air

DISTANCE	Distance from source to destination
CARRIER_DELAY	Delay caused by carrier
WEATHER_DELAY	Delay caused by weather
NAS_DELAY	Delay caused by National Airspace System (NAS)
SECURITY_DELAY	Delay caused due to security reasons
LATE_AIRCRAFT_DELAY	Delay caused by late arrival of flight

Dataset 2:

Weather data (2011-2015)
http://www.ncdc.noaa.gov/cdo-web/datasets/

ID	Unique station identification code
YEAR/MONTH/DAY	date in YYYYMMDD format
ELEMENT	element indicator
DATA VALUE	data value for element
M-FLAG	Measurement Flag
Q-FLAG	Quality Flag
S-FLAG	Source Flag
OBS-TIME	Observation time

References:

1] http://nbviewer.ipython.org/github/ofermend/IPython-notebooks/blob/master/blog-part-1.ipynb

2] http://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236&DB_Short_Name=On-Time

3] ftp://ftp.ncdc.noaa.gov/pub/data/ghcn/daily/by_year/

4] http://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp

5] http://www.ncdc.noaa.gov/cdo-web/datasets/

6] http://scikit-learn.org/stable/supervised_learning.html#supervised-learning

7] http://scikit-learn.org/stable/modules/ensemble.html

8] http://scikit-learn.org/stable/modules/gaussian_process.html

9] http://scikit-learn.org/stable/modules/svm.html

10] https://en.wikipedia.org/wiki/Precision_and_recall

11] https://en.wikipedia.org/wiki/Accuracy_and_precision

12] https://en.wikipedia.org/wiki/F1_score

13] http://www.airfarewatchdog.com/pages/3799702/airline-letter-codes/

14] http://www.cheapflights.com