
Right on Time

Smart Air Travels

By: Le Gu, Jacky Zhang

Initiatives

According to a study conducted by UC Berkeley researchers, flight delays put a \$32.9 billion hole in the U.S. economy in 2007, and more than half of that cost is borne by us, the passengers.

This project aims to better understand air travel experiences by analyzing flight on-time performance statistics. By exploring flight delays, the project hope to reveal delay trends by airport, airline, aircraft models and reasons for delay etc.



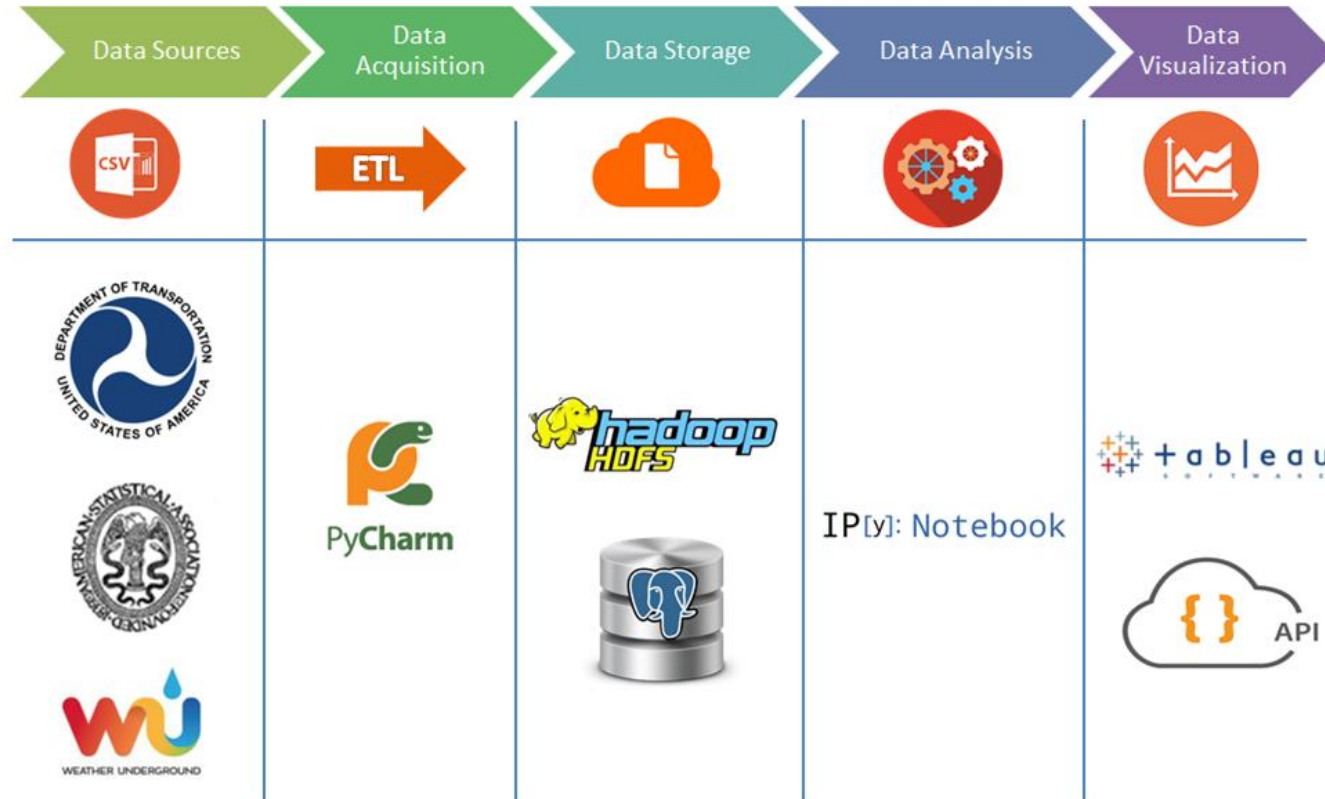
Data Sources

The United States Department of Transportation tracks on-Time performances of major US carriers. The data set contains flights departure, arrival details for US domestic flights since 1987. This project would take the most recent 2 years' data (6 GB) for analysis.

Statistical Computing Organization provides aircraft-related information such as manufacturer, issue year, model type etc and airport geographical information.

www.wunderground.com provides historical airport weather data.

Data Architecture

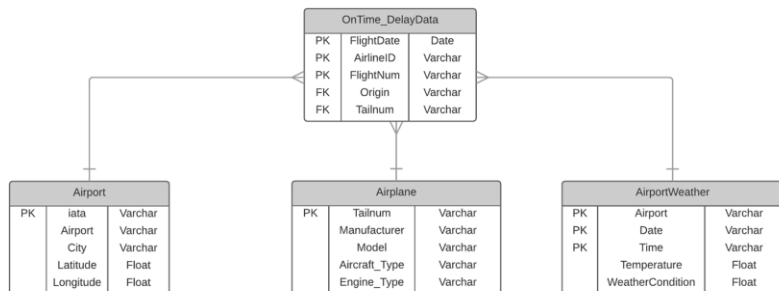


PostgreSQL

Structural and tabular data

Data retrieval optimization:

- Primary & Foreign Keys, Indexes
- Table partitionings



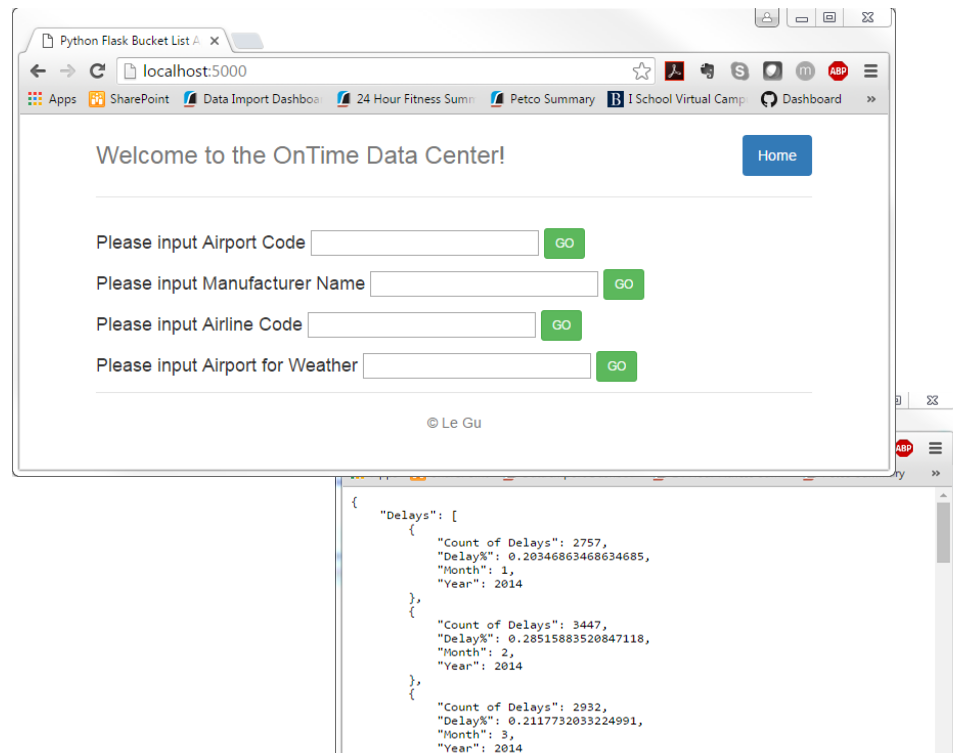
```
root@ip-172-31-57-133:~# psql -U postgres -d ontime
psql (8.4.20)
Type "help" for help.

ontime=# \dt
          List of relations
Schema |      Name      | Type | Owner
-----+-----+-----+-----
public | airplane_data  | table | w205
public | airport_data   | table | w205
public | airport_weather | table | w205
public | ontime_2014_01 | table | w205
public | ontime_2014_02 | table | w205
public | ontime_2014_03 | table | w205
public | ontime_2014_04 | table | w205
public | ontime_2014_05 | table | w205
public | ontime_2014_06 | table | w205
public | ontime_2014_07 | table | w205
public | ontime_2014_08 | table | w205
public | ontime_2014_09 | table | w205
public | ontime_2014_10 | table | w205
public | ontime_2014_11 | table | w205
public | ontime_2014_12 | table | w205
public | ontime_2015_01 | table | w205
public | ontime_2015_02 | table | w205
public | ontime_2015_03 | table | w205
public | ontime_2015_04 | table | w205
public | ontime_2015_05 | table | w205
public | ontime_2015_06 | table | w205
public | ontime_2015_07 | table | w205
public | ontime_2015_08 | table | w205
public | ontime_2015_09 | table | w205
public | ontime_2015_10 | table | w205
public | ontime_2015_11 | table | w205
public | ontime_2015_12 | table | w205
--More--
```

REST API & Flask Web Framework

- Serve Data from PostgreSQL Database through REST API using Python.
- Web interface based on Python Flask

```
w205@ip-172-31-57-133:~/project_api
'CA', u'06', u'California', u'91', u'0900', u'0855', u'-5.00', u'0.00', u'0.00',
u'-1', u'0900-0959', u'17.00', u'0912', u'1230', u'7.00', u'1230', u'1237', u'7
.00', u'7.00', u'0.00', u'0', u'1200-1259', u'0.00', u'', u'0.00', u'390.00', u'
402.00', u'378.00', u'1.00', u'2475.00', u'10', None, None, None, None, None, u'
', None, None, u'0', None, None, None, None, u'', None, None, u'', None, None, u'
', u'', u'', None, None, u'', None, None, u'', u'', u'', None, None, u'', None,
None, u'', u'', u'', None, None, u'', None, None, u'', u'', u'', None, None, u'
', None, None, u'', u'', None)]
* Running on http://0.0.0.0:8080/ (Press CTRL+C to quit)
* Restarting with stat
[(2015, 1, 1, 1, 4, datetime.date(2015, 1, 1), u'AA', u'19805', u'AA', u'N787AA'
, u'1', u'12478', u'1247802', u'31703', u'JFK', u'New York, NY', u'NY', u'36', u
'New York', u'22', u'12892', u'1289203', u'32575', u'LAX', u'Los Angeles, CA', u
'CA', u'06', u'California', u'91', u'0900', u'0855', u'-5.00', u'0.00', u'0.00',
u'-1', u'0900-0959', u'17.00', u'0912', u'1230', u'7.00', u'1230', u'1237', u'7
.00', u'7.00', u'0.00', u'0', u'1200-1259', u'0.00', u'', u'0.00', u'390.00', u'
402.00', u'378.00', u'1.00', u'2475.00', u'10', None, None, None, None, None, u'
', None, None, u'0', None, None, None, None, u'', None, None, u'', None, None, u'
', u'', u'', None, None, u'', None, None, u'', u'', u'', None, None, u'', None,
None, u'', u'', u'', None, None, u'', None, None, u'', u'', u'', None, None, u'
', None, None, u'', u'', None)]
* Debugger is active!
* Debugger pin code: 816-658-900
```



Tableau

- Access remote PostgreSQL database from Hive Thrift Server
- Join different data sources in Tableau for analysis and visualization

```
w205@ip-172-31-57-133:~  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
(api) [w205@ip-172-31-57-133 ~]$  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: Found binding in [jar:file:/home/w205/spark15/lib/spark-assembly-1.5.0-hadoop2.6.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]  
SLF4J: Class path contains multiple SLF4J bindings.  
SLF4J: Found binding in [jar:file:/usr/lib/zookeeper/lib/slf4j-log4j12-1.7.5.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: Found binding in [jar:file:/home/w205/spark15/lib/spark-assembly-1.5.0-hadoop2.6.0.jar!/org/slf4j/impl/StaticLoggerBinder.class]  
SLF4J: See http://www.slf4j.org/codes.html#multiple_bindings for an explanation.  
SLF4J: Actual binding is of type [org.slf4j.impl.Log4jLoggerFactory]
```

Tableau - OnTimeProject

File Data Server Window Help

ontime_2015_01+ (ontime)

Connected to PostgreSQL

Server
54.164.201.65

Database
ontime

Table
Enter table name

ontime_2014_01
ontime_2014_02
ontime_2014_03
ontime_2014_04
ontime_2014_05
ontime_2014_06
ontime_2014_07
ontime_2014_08
ontime_2014_09
ontime_2014_10
ontime_2014_11
ontime_2014_12
ontime_2015_01
ontime_2015_02
ontime_2015_03

ontime_2015_01

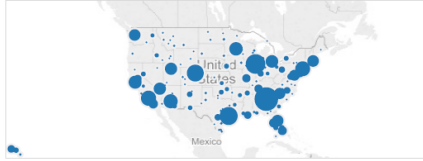
airplane_data
airport_data
airport_weather

Sort fields Data source order

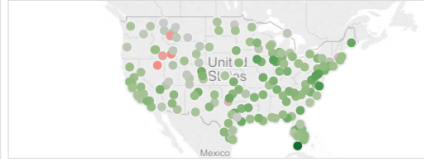
Abc	Abc	Abc	Abc	Abc	Abc	Abc
airplane_data	airplane_data	airplane_data	airplane_data	airplane_data	airplane_data	airplane_data
Tailnum (Airplane ...	Type	Manufacturer	Issue Date	Model	Status	Aircraft Type
N634AA	Corporation	BOEING	12/20/1990	757-223	Valid	Fixed Wing Multi-Engi...
N634AA	Corporation	BOEING	12/20/1990	757-223	Valid	Fixed Wing Multi-Engi...
N634AA	Corporation	BOEING	12/20/1990	757-223	Valid	Fixed Wing Multi-Engi...
N634AA	Corporation	BOEING	12/20/1990	757-223	Valid	Fixed Wing Multi-Engi...
N634AA	Corporation	BOEING	12/20/1990	757-223	Valid	Fixed Wing Multi-Engi...

Tableau

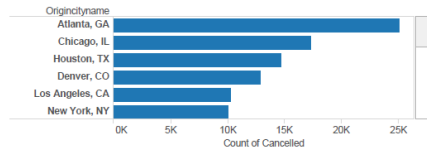
Total Flight by Airport



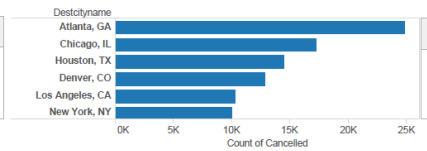
Avg Delay by Airport (minute)



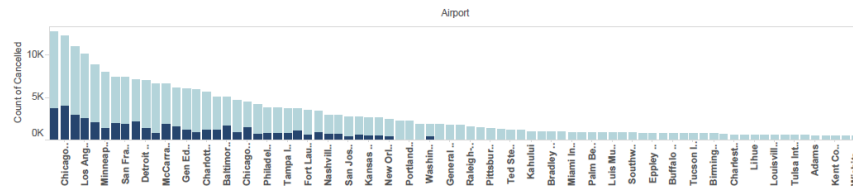
Top Origin City



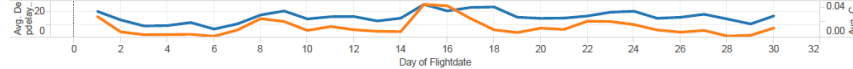
Top Destination City



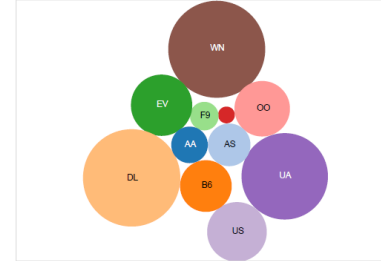
Total Airport Delay vs Total Flights



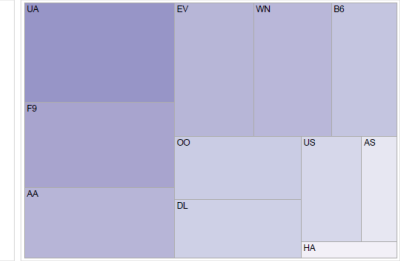
Delay and Cancellation in a Month (percentage)



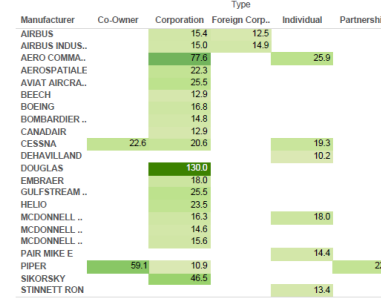
Carrier Total Distance



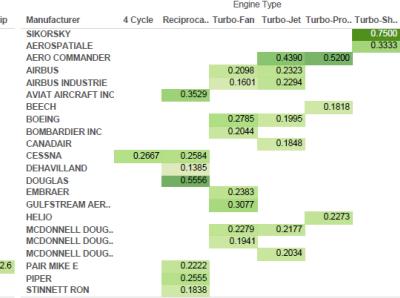
Carrier Avg Delays (minute)



Manufacturer, Plan Model vs Delays (minute)



Manufacturer, Engine vs Delays (percentage)



Data Analysis

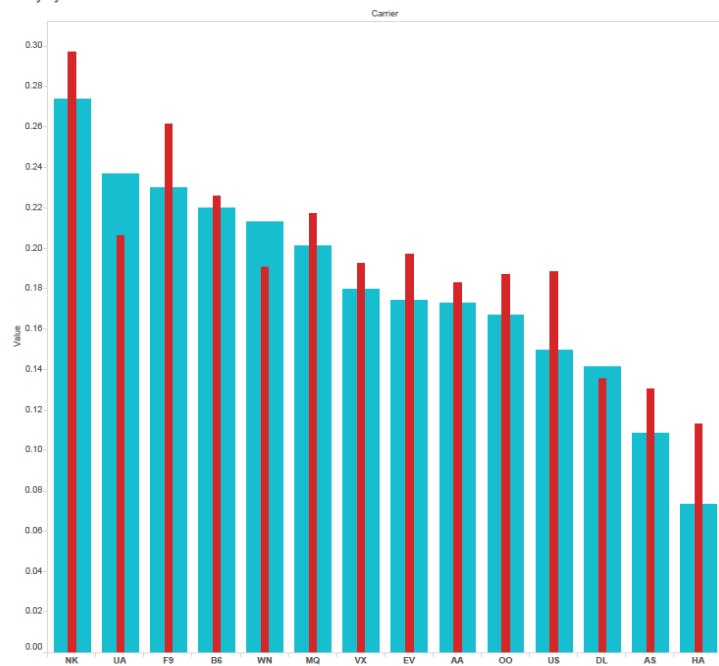
1. Which airlines have the most delays?
2. Do longer flights have more delays?
3. When is the worst time to travel, in terms of delays expected?
4. Which are the worst months the travel, in terms of delays expected?

Predictive Model:

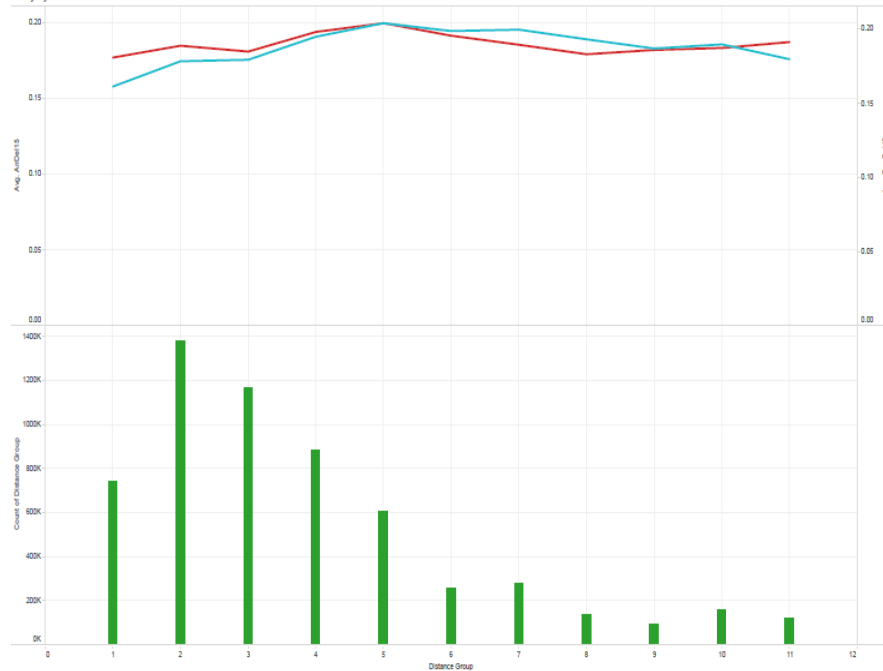
We seek to build a prediction model that gives the users an estimate of expected delay given information about their flight.

Exploratory Analysis - Tableau

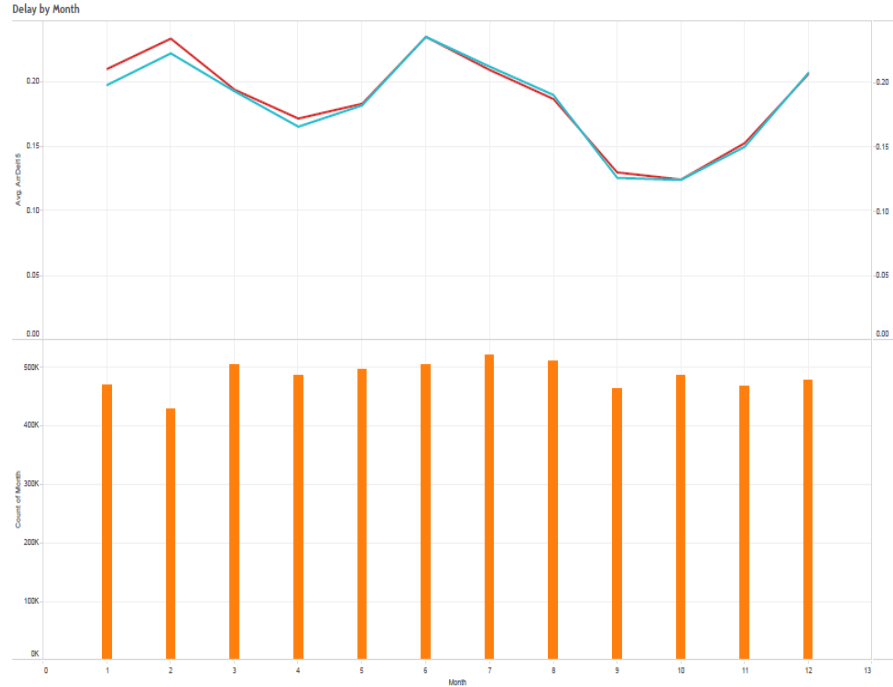
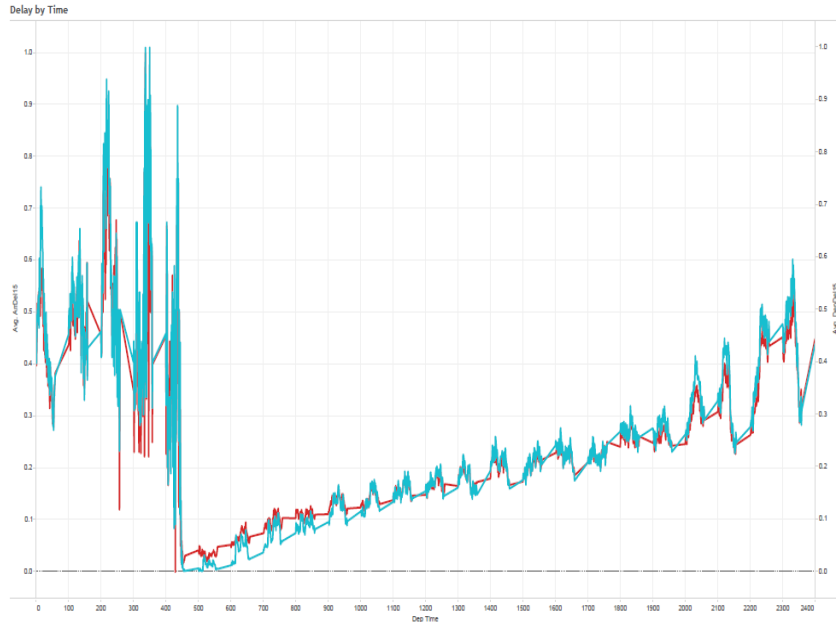
Delay by Carrier



Delay by Distance



Exploratory Analysis - Tableau



Prediction Model

Used IPython notebook to connect to Postgres database

Used features identified in exploratory analysis

Random forest classifier

- Data size: 2 million records
- Input variables: 10; Year, Month, DayOfMonth, DayOfWeek, DepTime, Dest, Origin, Carrier, DistanceGroup, Holiday
- Output variable: DepDel15 (Binary)
- Accuracy to beat: ~80%
- Accuracy: 83.5%
- F1-Score: 30.7

```
In [3]: conn = psycopg2.connect(database="ontime", user="postgres", host="localhost", port="5432")

In [4]: delay = pd.read_sql_query("SELECT year, quarter, month, dayofmonth, dayofweek, flightdate, deptime, carrier, tailnum, \n
                                origin, dest, distance, distancegroup, depdelayminutes, depdel15, arrdelayminutes, \n
                                arrdel15 FROM ontime_data LIMIT 2000000", conn)
```



```
In [40]: predict = np.asarray([2015, 3, 4, 3, 1220, 217, 83, 10, 11, 0])

print ("Probability of delay:" , rf.predict_proba(predict.reshape(1,10))[0][1])

('Probability of delay:', 0.72004257916726822)
```

	Feature Importance
DepTime	0.373418
Origin	0.147665
Dest	0.139933
DayOfMonth	0.092986
Carrier	0.073735
DistanceGroup	0.072669
Month	0.052429
DayOfWeek	0.044109
Holiday	0.003054
Year	0.000000

Next Steps

Solution Scalability:

- In order to serve concurrent users, we will need to build EC2 clusters that allow the parallel processing and offer higher IOPS.
- As data get accumulated in the Postgres database, the query optimizer would be able to leverage the table partitioning and the performance should not be greatly affected.

Solution Evolution:

- Twitter data sentiment analysis: introduce real-time stream processing with Twitter developer API, apply sentiment analysis to track public opinions on flight delays.
- Data Pipeline Scheduler: incorporate AWS data pipeline API to schedule runs in a automated fashion, such as real-time twitter data collection, nightly weather data web scraping, and monthly data feeds acquisition and model retrain.