
**Graduate and Extended Studies**

# Airline Delay and Prediction Using Machine Learning

**Submitted by:**

**Team Spartans**

**Sudha Amarnath**

**Mukesh Ranjan Sahay**

**Muthu Kumar**

**Thirumalai Nambi**

**FA19: CMPE-297 Sec 01 - Special Topics**

**Prof. Chandrasekar Vuppalapati**

## Table of contents

# Introduction

One of the biggest downsides of traveling is the delays and cancellation of flights which results in the deteriorating customer satisfaction. These delays are associated with many factors including the weather conditions, connecting flights or technical issues, and these prove costly, both quantitative and qualitative. Also, the growing aviation industry has resulted in the air-traffic congestion which increases the chances of the flights getting delayed. Along with the economic impacts, the flight delays are also associated with the harmful effects on the environment. With all these things going around, it is becoming very challenging to effectively manage the air-traffic and to meet the traveler's expectations.

# Description

Unpredicted flight delays can ruin the extraordinary vacation memories. The airline delay makes the passengers lose their trust from the famous and internationally recognized airlines. An advanced and automated prediction system with a great accuracy must be created that can predict the likely airline delay. This system can save passengers from the hassle and can also help the airlines to run their business smoothly. Apart from the delays resulting due to the weather, security and the limited airspace capacity, one-third of the airline delays were caused by a late-arriving aircraft and thus making it depart late for its next schedule, which is known as delay propagation. Generally, the airlines run their aircraft on a scheduled itinerary daily which requires the transit from a network of airports, a late-arriving aircraft early in the day can significantly impact the upcoming flights. For example, if an aircraft is delayed by one hour in a departure from the first airport, it will almost certainly be late in arriving at its next airport; the late arrival may also result in a late subsequent departure of that aircraft, which will lead to a sequence of late-arriving aircraft delays.

With this in mind, we decided to pursue the issue of predicting fight delays using machine learning. More precisely, our aim was to predict fight delays for flights departing from SFO using information like weather conditions, flight carrier, day of the month, destination airport. We trained our model on the collected datasets. Such a model is useful because:

- The analysis of air delays becomes vital since a better knowledge of their existence, and corresponding triggers, can improve the operational performance of airlines and, consequently airports, by anticipating delay and preparing schedules accordingly.
- Predicting departure delay is vital for customer satisfaction, reduced congestion at airports, preventing the ripple effect of one delay leading to another and also save costs.

# Requirements

## Libraries

Data Processing - pandas, scipy, sklearn and numpy
Exploratory Data Analysis(EDA) - Seaborn, Matplolib, Basemap
Models - regression, classifiers, figures

## Data Sets

Below data set is from Bureau of Transportation and Statistics which has three different data points and one from NOAA SFO WeatherData.

1 - Airline Dataset
2 - Airport Dataset
3 - Flights Dataset
4 - SFO WeatherDelay Dataset

## Colab Link For The Project:

**https://colab.research.google.com/drive/13ngycE7THvdSsCuiI6l7bzTeWTfnWfgS?authuser=1**

## Original Data Sets URL

**https://www.transtats.bts.gov/ONTIME/**

**https://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp**

**https://www.transtats.bts.gov/DL_SelectFields.asp?Table_ID=236&DB_Short_Name=On-Time**

## Amalgamated Datasets

**Flights: https://drive.google.com/open?id=11Qt7LpDbNllgEg5R69TT_yFNEaBJ5WUc**

**Airlines: https://drive.google.com/open?id=1uByCvwq6hqFYUpFWN2HphhEMua3d-J_1**

**Airports:https://drive.google.com/open?id=1XuH_OuJKYGxnvp3caUGKZ_mUsvQA29tM**

# Implementation Approaches

## KDD - Knowledge Discovery

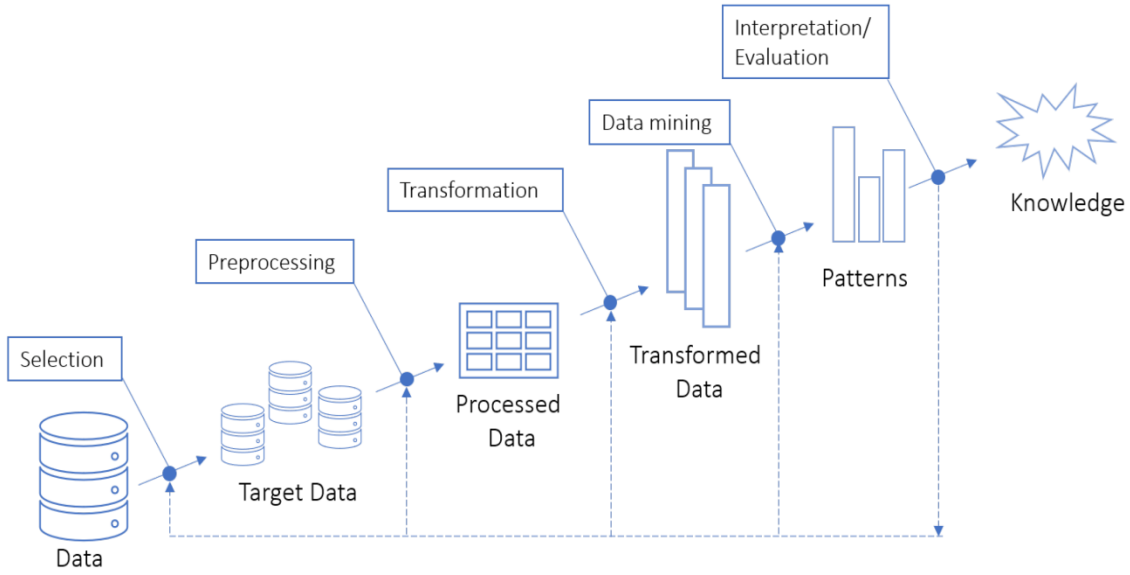



*Figure1:* KDD Process

- Data Cleaning
- Data Integration
  - Airline Data
  - Airport Data
  - Flights Data
  - Weather Data
- Data Mining
  - Analyze the data insights with patterns of data points
- Presentation
  - Geographical Areas Covered by Airlines
- Knowledge
  - Airlines Comparisons for delays
  - Delays in Landing or TakeOff?
  - Relation between Origin Airport and delays?
  - Delay impacted by weather report
- Prediction
  - SFO Airlines departure and arrival delay prediction.

## Machine Learning lifecycle 1-8

1. Configuration of the System : Iterative, Notebook, code structure, data, where will it reside, folders, cloud buckets etc.
2. Data Collection : initial Data Set
3. Set Data Narrative : Set Business Objectives, what use case are you solving for
4. Exploratory Data Analysis and Visualization
    - feature analysis and engineering (for ML, for DL it's feature extraction)
    - Analyze data
    - Visualize data
    - Run Stats: mean, median, mode, correlation, variance
    - .corr
    - pairplot()
    - gini score
5. Data Prep: Curation
    - Feature Selection and Extraction : what are the main features to use in this data set?
    - Data Verification: Do we have enough data?
    - Possibility of Amalgamation1: Add Dataset 2
    - Data Cleansing
    - Data Regularization
    - Data Normalization
6. Unsupervised Exploration : Find relevant Clusters in Your Data
    - How many clusters? Explore different k's…
    - Select Clustering algorithms, run several and compare in a table
    - What does each cluster mean? How do they contribute to your Data Narrative (Story)
    - Measure goodness of your clusters
7. Supervised Training Preparation: Data Curation : label your data set
    - Classify Your Data Sets : Run different classification algorithms
    - Measure Classification Success
    - What regression objectives should we have? Complete your , add to your Data Story
    - Run Regressions using various algorithms
    - Measure Success of Regressions and
    - Compare Regressions in a table

8. Metrics and Evaluation
    - F1, R2, RMSE,
    - Precision, Recall, Accuracy
    - Confusion Matrix

# Data collection and preparation

## Airlines Dataset Dataframe with its shape

Read airline data from airlines Dataset. Features in airline data set -

- IATA_CODE - International Air Transport Association airport code
- AIRLINE - Airline Description

Airlines dataframe dimentions: (18, 2)

| | IATA_CODE | AIRLINE |
|---|---|---|
| 0 | UA | United Air Lines Inc. |
| 1 | AA | American Airlines Inc. |
| 2 | F9 | Frontier Airlines Inc. |
| 3 | B6 | JetBlue Airways |
| 4 | OO | Skywest Airlines Inc. |
| 5 | AS | Alaska Airlines Inc. |
| 6 | NK | Spirit Air Lines |
| 7 | WN | Southwest Airlines Co. |
| 8 | DL | Delta Air Lines Inc. |
| 9 | EV | Atlantic Southeast Airlines |
| 10 | HA | Hawaiian Airlines Inc. |
| 11 | MQ | American Eagle Airlines Inc. |
| 12 | VX | Virgin America |
| 13 | 9E | Endeavor Air |
| 14 | YV | Air Shuttle |
| 15 | OH | Blue Streak |
| 16 | YX | Midwest Airlines |
| 17 | G4 | Allegiant Air |

## Flights dataframe display with column and shape

```
df.columns
```

```
Index(['AIRLINE', 'AIRLINE_DELAY', 'AIR_SYSTEM_DELAY', 'AIR_TIME',
       'ARRIVAL_DELAY', 'ARRIVAL_TIME', 'CANCELLATION_REASON', 'CANCELLED',
       'DAY', 'DAY_OF_WEEK', 'DEPARTURE_DELAY', 'DEPARTURE_TIME',
       'DESTINATION_AIRPORT', 'DISTANCE', 'DIVERTED', 'ELAPSED_TIME',
       'FLIGHT_NUMBER', 'LATE_AIRCRAFT_DELAY', 'MONTH', 'ORIGIN_AIRPORT',
       'SCHEDULED_ARRIVAL', 'SCHEDULED_DEPARTURE', 'SCHEDULED_TIME',
       'SECURITY_DELAY', 'TAIL_NUMBER', 'TAXI_IN', 'TAXI_OUT', 'WEATHER_DELAY',
       'WHEELS_OFF', 'WHEELS_ON', 'YEAR'],
      dtype='object')
```

Dataframe dimensions: (12826280, 31)

| | AIRLINE | AIRLINE_DELAY | AIR_SYSTEM_DELAY | AIR_TIME | ARRIVAL_DELAY | ARRIVAL_TIME | CANCELLATION_REASON | CANCELLED | DAY | DAY_OF_WEEK | DEPARTURE_DELAY |
|---|---|---|---|---|---|---|---|---|---|---|---|
| column type | object | float64 | float64 | float64 | float64 | float64 | object | float64 | int64 | int64 | float64 |
| null values (nb) | 0 | 10409343 | 10409343 | 260009 | 262607 | 231759 | 12599121 | 0 | 0 | 0 | 223899 |
| null values (%) | 0 | 81.1564 | 81.1564 | 2.02716 | 2.04741 | 1.80691 | 98.229 | 0 | 0 | 0 | 1.74563 |

## Airport s information with shape

Airports dataframe dimentions: (322, 7)

| | IATA_CODE | AIRPORT | CITY | STATE | COUNTRY | LATITUDE | LONGITUDE |
|---|---|---|---|---|---|---|---|
| 0 | ABE | Lehigh Valley International Airport | Allentown | PA | USA | 40.65236 | -75.44040 |
| 1 | ABI | Abilene Regional Airport | Abilene | TX | USA | 32.41132 | -99.68190 |
| 2 | ABQ | Albuquerque International Sunport | Albuquerque | NM | USA | 35.04022 | -106.60919 |
| 3 | ABR | Aberdeen Regional Airport | Aberdeen | SD | USA | 45.44906 | -98.42183 |
| 4 | ABY | Southwest Georgia Regional Airport | Albany | GA | USA | 31.53552 | -84.19447 |

## SFO WeatherData dataframe with column and shape

Climate Data Online (CDO) provides free access to NCDC's archive of global historical weather and climate data in addition to station history information. These data include quality controlled

daily, monthly, seasonal, and yearly measurements of temperature, precipitation, wind, and degree days as well as radar data and 30-year Climate Normals.

- Average Temperature
- Maximum temperature
- Minimum temperature

```
sfoorigindf.columns
```

```
Index(['AIRLINE', 'AIRLINE_DELAY', 'AIR_SYSTEM_DELAY', 'AIR_TIME',
       'ARRIVAL_DELAY', 'ARRIVAL_TIME', 'CANCELLATION_REASON', 'CANCELLED',
       'DAY', 'DAY_OF_WEEK', 'DEPARTURE_DELAY', 'DEPARTURE_TIME',
       'DESTINATION_AIRPORT', 'DISTANCE', 'DIVERTED', 'ELAPSED_TIME',
       'FLIGHT_NUMBER', 'LATE_AIRCRAFT_DELAY', 'MONTH', 'ORIGIN_AIRPORT',
       'SCHEDULED_ARRIVAL', 'SCHEDULED_DEPARTURE', 'SCHEDULED_TIME',
       'SECURITY_DELAY', 'TAIL_NUMBER', 'TAXI_IN', 'TAXI_OUT', 'WEATHER_DELAY',
       'WHEELS_OFF', 'WHEELS_ON', 'YEAR', 'DEPARTURE_DELAY_15',
       'ARRIVAL_DELAY_15'],
      dtype='object')
```

SFO weather dataframe dimensions: (638, 22)

| | STATION | NAME | LATITUDE | LONGITUDE | ELEVATION | DATE | AWND | PGTM | PRCP | SNOW | SNWD | TAVG | TMAX | TMIN | WDF2 | WDF5 | WSF2 | WSF5 | WT01 | WT02 | WT03 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | USW00023234 | SAN FRANCISCO INTERNATIONAL AIRPORT, CA US | 37.6197 | -122.3647 | 2.4 | 1/1/18 | 1.79 | NaN | 0.00 | 0.0 | 0.0 | 52 | 58.0 | 45.0 | 350 | 40.0 | 6.9 | 10.1 | 1.0 | NaN | NaN |
| 1 | USW00023234 | SAN FRANCISCO INTERNATIONAL AIRPORT, CA US | 37.6197 | -122.3647 | 2.4 | 1/2/18 | 2.46 | NaN | 0.00 | 0.0 | 0.0 | 54 | 59.0 | 50.0 | 150 | 150.0 | 8.1 | 11.0 | 1.0 | NaN | NaN |
| 2 | USW00023234 | SAN FRANCISCO INTERNATIONAL AIRPORT, CA US | 37.6197 | -122.3647 | 2.4 | 1/3/18 | 2.91 | NaN | 0.30 | 0.0 | 0.0 | 55 | 58.0 | 51.0 | 150 | 140.0 | 12.1 | 14.1 | 1.0 | NaN | NaN |
| 3 | USW00023234 | SAN FRANCISCO INTERNATIONAL AIRPORT, CA US | 37.6197 | -122.3647 | 2.4 | 1/4/18 | 6.04 | NaN | 0.03 | 0.0 | 0.0 | 54 | 63.0 | 52.0 | 180 | 180.0 | 19.9 | 21.9 | 1.0 | NaN | NaN |
| 4 | USW00023234 | SAN FRANCISCO INTERNATIONAL AIRPORT, CA US | 37.6197 | -122.3647 | 2.4 | 1/5/18 | 5.59 | NaN | 0.19 | 0.0 | 0.0 | 59 | 64.0 | 53.0 | 280 | 270.0 | 19.9 | 21.9 | 1.0 | NaN | NaN |

# Data Cleaning

Read airport data from flights Dataset.Clean up all the other features except below

- 'IATA_CODE',
- 'AIRPORT',
- 'CITY',
- 'STATE',

- 'COUNTRY',
- 'LATITUDE',
- 'LONGITUDE'

Airports dataframe dimentions: (322, 7)

| | IATA_CODE | AIRPORT | CITY | STATE | COUNTRY | LATITUDE | LONGITUDE |
|---|---|---|---|---|---|---|---|
| 0 | ABE | Lehigh Valley International Airport | Allentown | PA | USA | 40.65236 | -75.44040 |
| 1 | ABI | Abilene Regional Airport | Abilene | TX | USA | 32.41132 | -99.68190 |
| 2 | ABQ | Albuquerque International Sunport | Albuquerque | NM | USA | 35.04022 | -106.60919 |
| 3 | ABR | Aberdeen Regional Airport | Aberdeen | SD | USA | 45.44906 | -98.42183 |
| 4 | ABY | Southwest Georgia Regional Airport | Albany | GA | USA | 31.53552 | -84.19447 |

Removing unnecessary columns in Flights dataset and converting day, month, year to datetime format.

Flights cleaned dataframe dimentions: (1154518, 11)

| | AIRLINE | ORIGIN_AIRPORT | DESTINATION_AIRPORT | SCHEDULED_DEPARTURE | DEPARTURE_TIME | DEPARTURE_DELAY | SCHEDULED_ARRIVAL | ARRIVAL_TIME | ARRIVAL_DELAY | SCHE |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | UA | FLL | IAH | 2018-01-27 06:15:00 | 06:02:00 | -13.0 | 08:08:00 | 07:56:00 | -12.0 | |
| 1 | UA | SEA | SFO | 2018-01-27 06:18:00 | 06:14:00 | -4.0 | 08:31:00 | 08:13:00 | -18.0 | |
| 2 | UA | DCA | IAH | 2018-01-27 08:30:00 | 08:28:00 | -2.0 | 11:07:00 | 11:08:00 | 1.0 | |
| 3 | UA | LAX | ORD | 2018-01-27 06:50:00 | 06:41:00 | -9.0 | 12:50:00 | 12:42:00 | -8.0 | |
| 4 | UA | JAX | EWR | 2018-01-27 18:24:00 | 18:10:00 | -14.0 | 20:45:00 | 20:21:00 | -24.0 | |

# Data Amalgamation

Combining Flight dataset with Weather dataset with SFO as origin Airport to enrich the data for a valuable outcome.

Amalgamated Flight and SFO Weatherdata dimension: (13850, 29)

| | AIRLINE | ORIGIN_AIRPORT | DESTINATION_AIRPORT | SCHEDULED_DEPARTURE | DEPARTURE_TIME | DEPARTURE_DELAY | SCHEDULED_ARRIVAL | ARRIVAL_TIME | ARRIVAL_DELAY | SCHE |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | UA | SFO | LAS | 2018-01-27 10:42:00 | 10:33:00 | -9.0 | 12:21:00 | 12:08:00 | -13.0 | |
| 1 | UA | SFO | IND | 2018-01-27 12:00:00 | 11:46:00 | -14.0 | 19:13:00 | 18:51:00 | -22.0 | |
| 2 | UA | SFO | RNO | 2018-01-27 22:54:00 | 22:48:00 | -6.0 | 23:59:00 | 23:41:00 | -18.0 | |
| 3 | UA | SFO | HNL | 2018-01-27 13:20:00 | 13:15:00 | -5.0 | 16:59:00 | 17:11:00 | 12.0 | |
| 4 | UA | SFO | DFW | 2018-01-27 10:40:00 | 10:32:00 | -8.0 | 16:15:00 | 15:53:00 | -22.0 | |

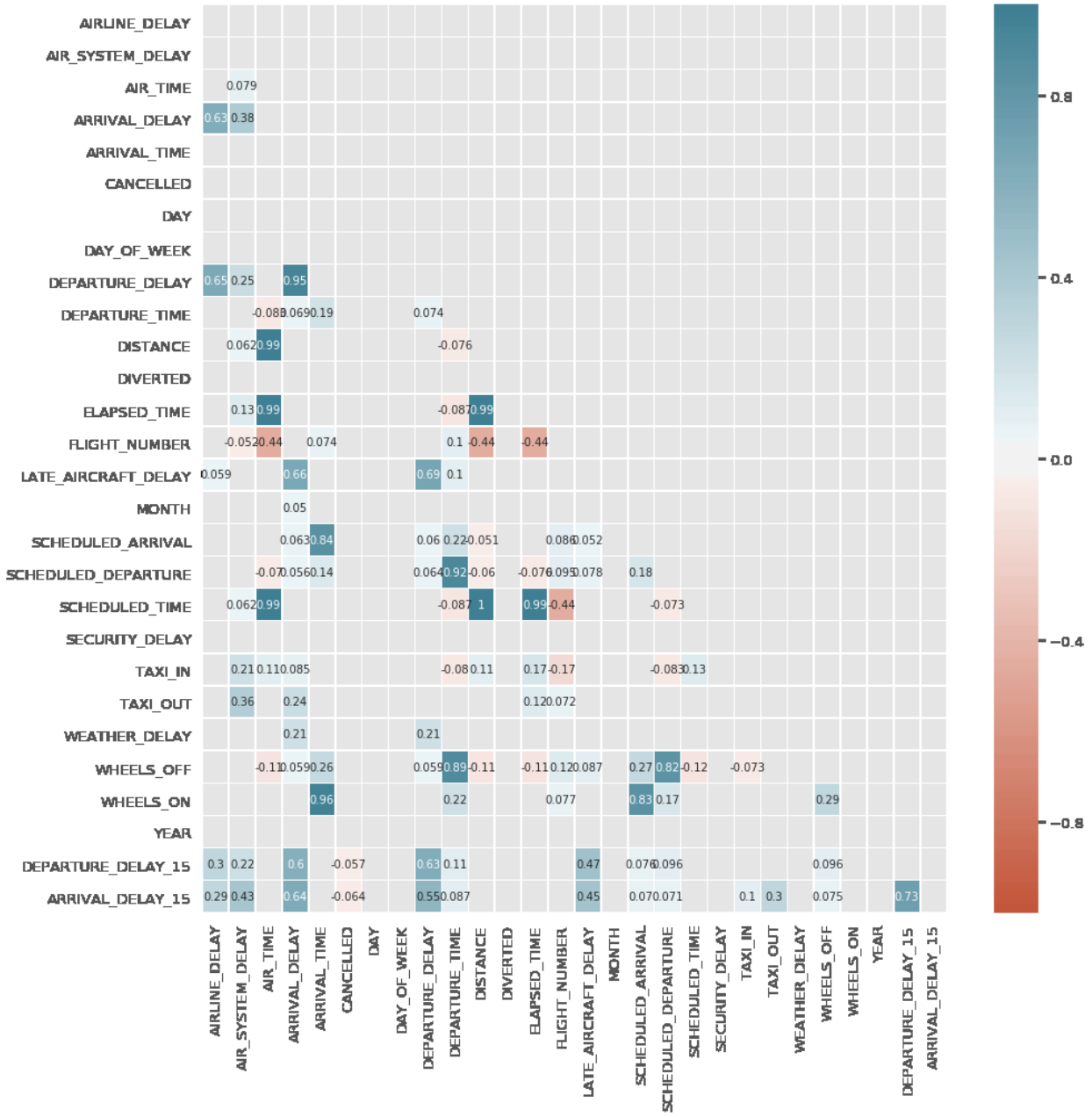
# Correlation heat map

## Correlation heatmap for SFO as origin airport

We created a basic plot by putting the correlation of this dataframe into a Seaborn heatmap. Our main goal is to find the correlation between departure and arrival delays for weather and late-arriving aircrafts, checking for delays >= 15 minutes form SFO as origin airport. The coorelation matrix provide the result that the departure and arrival delay are positively and strongly coorelated which helps us in further evaluation.

```
sfoorigindf.head()
```

| DAY | DAY_OF_WEEK | DEPARTURE_DELAY | DEPARTURE_TIME | DESTINATION_AIRPORT | DISTANCE | DIVERTED | ELAPSED_TIME | FLIGHT_NUMBER | LATE_AIRCRAFT_DELAY | MONTH | ORIGIN_AIRPORT | SCHEDULED_ARRIVAL | SCHEDUL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 27 | 6 | -9.0 | 1033.0 | LAS | 414.0 | 0.0 | 95.0 | 358 | 0.0 | 1 | SFO | 1221 | |
| 27 | 6 | -14.0 | 1146.0 | IND | 1943.0 | 0.0 | 245.0 | 317 | 0.0 | 1 | SFO | 1913 | |
| 27 | 6 | -6.0 | 2248.0 | RNO | 192.0 | 0.0 | 53.0 | 313 | 0.0 | 1 | SFO | 2359 | |
| 27 | 6 | -5.0 | 1315.0 | HNL | 2398.0 | 0.0 | 356.0 | 300 | 0.0 | 1 | SFO | 1659 | |
| 27 | 6 | -8.0 | 1032.0 | DFW | 1464.0 | 0.0 | 201.0 | 294 | 0.0 | 1 | SFO | 1615 | |




# SNS pairplot

Scatterplots for joint relationships and histograms for univariate distributions for the above columns.

```
sns.pairplot(sfoorigindf)
```

# Algorithm Selection

## Target Variable

The target feature is departure delay, which tells the number of minutes a flight is delayed after its scheduled departure time. The original feature had a wide range of integer values, but since our business problem just focuses on predicting whether a flight will be delayed or not and not the actual time by which the flight delayed, only two classes were used: delay>15 and delay<15 minutes.

According to the Federal Aviation Administration (FAA), a flight is considered to be delayed if takes off and/or lands 15 minutes later than its scheduled time. Taking this into account, our target variable was modifieed to take only binary values (0 or 1). 0 representing the flight is not-delayed (departuredelay <= 15) and 1 representing the flight is delayed (departure delay > 15).

## Classificaton Algorithms

### Logistic regression

Logistic Regression has the advantage of being interpretable and computationally inexpensive and thus, we decided to use it as our baseline model, but it can't perform well in a large feature space and large sample sizes with hyperparameter tuning by adding weather features.

### K nearest neighbours

Our baseline model was a linear classifier which didn't perform well and in order to increase the performance, we decided to choose a non-linear classifier. Also, KNN is easy to understand and is an intuitive algorithm but since it gets all of its information from the input's neighbors, localized anomalies affect outcomes significantly, compared to an algorithm that uses a generalized view of the data. k is one of the hyper-parameters to tune along with the choice of distance metric.

### Decision tree

Our decision to apply decision trees was motivated by the implicit feature selection performed by this non-linear classifier. We use entropy as the criterion and use a grid search to tune the parameters min samples split and min samples leaf. They can be interpreted easily but are prone to overfitting and can easily create complex trees that do not generalise well.

### Random Forest

Decision trees have high variance associated with them because of their unstable nature and thus, we decided to proceed to ensemble methods. We performed a grid search over the number of trees and max tree depth. The drawback of using random forests is that it performs worse than decision trees when the feature space is partially sparse.

### Naïve Bayes

An advantage of naive bayes is that it only requires a small amount of training data to estimate the parameters (means and variances of the variables) necessary for classification. Because independent variables are assumed, only the variances of the variables for each class need to be determined and not the entire covariance matrix.

# Function to execute classification algorithms:

```
def run_classification_algorithms(DF, X_train_cols, y_train_col,
algorithms='all'):
    global algorithms_performace
    global algorithms_performace_df
    #if algorithms == 'all':
      #algorithms_performace = []
    if algorithms == 'all' or algorithms == 'Logistic Regression':
      classification_algorithm(DF, X_train_cols, y_train_col,
LogisticRegression(random_state=100), 'Logistic Regression')
    if algorithms == 'all' or algorithms == 'Naive Bayes':
        classification_algorithm(DF, X_train_cols, y_train_col, GaussianNB(),
'Naive Bayes')
    if algorithms == 'all' or algorithms == 'Decision Tree':
        classification_algorithm(DF, X_train_cols, y_train_col,
DecisionTreeClassifier(max_depth=5, random_state=100), 'Decision Tree')
    if algorithms == 'all' or algorithms == 'Random Forest':
        classification_algorithm(DF, X_train_cols, y_train_col,
RandomForestClassifier(max_depth=5,n_estimators=3, random_state=100), 'Random
Forest')
    if algorithms == 'SVM':
        classification_algorithm(DF, X_train_cols, y_train_col,
SVC(random_state = 100), 'SVM')
    if algorithms == 'all' or algorithms == 'KNN':
      classification_algorithm(DF, X_train_cols, y_train_col,
KNeighborsClassifier(n_neighbors = 20, metric = 'minkowski', p = 2), 'KNN')
    algorithms_performace_df = pd.DataFrame(algorithms_performace)
    if algorithms == 'all':
        algorithms_performace_df.plot(x='algorithm', y='accuracy',
kind='bar', color='green')
        print(algorithms_performace_df.head())
```

**Function to calculate metrics**: F1 score, Recall, Precision, R2score, ROCscore, MAE, MSE, RMSE

```
from sklearn.metrics import roc_auc_score

global algorithms_performace_df


def calculate_scores(y_test, y_pred, classifier):
    global algorithms_performace

    accuracyScore = round(accuracy_score(y_test, y_pred) * 100,3)
    f1Score = round(f1_score(y_test, y_pred),3)
    recall = round(recall_score(y_test, y_pred),3)
    precisionScore =  round(precision_score(y_test, y_pred),3)
```

```
    mae= round(metrics.mean_absolute_error(y_test, y_pred),3)
    mse= round(metrics.mean_squared_error(y_test, y_pred),3)
    rmse= round(np.sqrt(metrics.mean_squared_error(y_test, y_pred)),3)
    r2Score = round(r2_score(y_test, y_pred),3)
    cm = confusion_matrix(y_test, y_pred)
    rocScore = round(roc_auc_score(y_test,y_pred), 3)

    print('\n\nPrinting Scores for', classifier, 'Algorithm')
    print('Accuracy:', accuracyScore)
    print('F1 score:', f1Score)
    print('Recall:', recall)
    print('Precision:', precisionScore)
    print('R2Score:', r2Score)
    print('ROCScore:', rocScore)
    print('MAE', mae)
    print('MSE', mse)
    print('RMSE', rmse)
    print('\n clasification report:\n', classification_report(y_test,y_pred))
    print('\n confussion matrix:\n',cm)
    print()

    performance = {'algorithm':classifier, 'accuracy': accuracyScore,
'precision':precisionScore, 'recall':recall, 'fscore':f1Score,
                   'r2score':r2Score, 'rocscore':rocScore, 'mae':mae,
'mse':mse,'rmse':rmse}
    algorithms_performace.append(performance)

    #Confusion Matrix

    fig, ax = plt.subplots()
    sns.heatmap(pd.DataFrame(cm), annot=True, cmap="YlGnBu" ,fmt='g')
    ax.xaxis.set_label_position("top")
    plt.tight_layout()
    plt.title('Confusion matrix for '+classifier, y=1.1)
    plt.ylabel('Actual label')
    plt.xlabel('Predicted label')
    plt.show()

    print('\n\n')
```

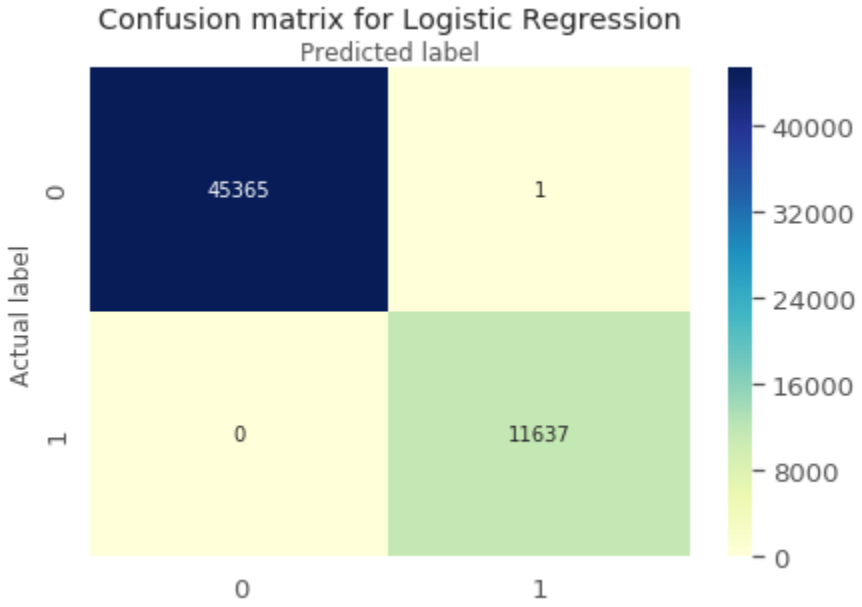
## Results

```
algorithms_performace = []
run_classification_algorithms(sfoorigindf,allcols, 'DEPARTURE_DELAY_15',
'all')
```

## Logistic Regression

```
Printing Scores for Logistic Regression Algorithm
```

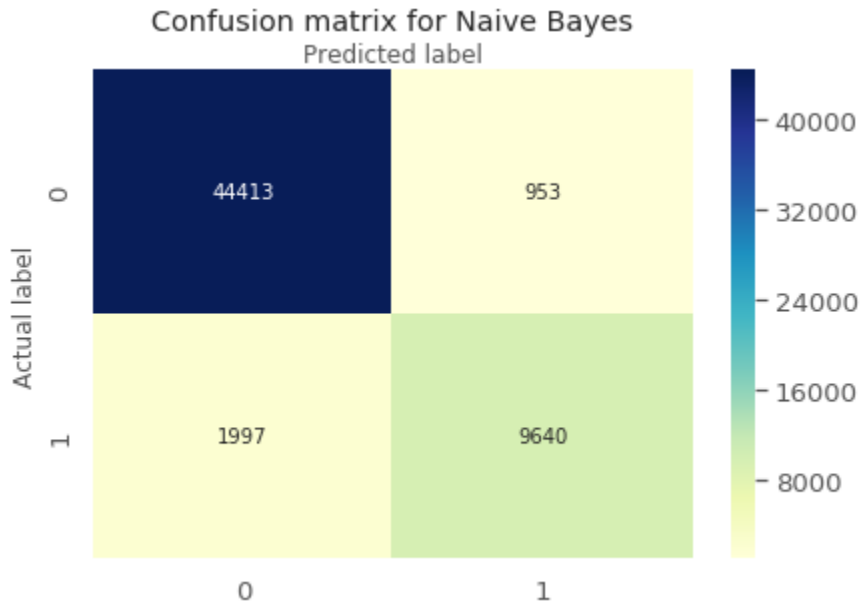
```
Accuracy: 99.998
F1 score: 1.0
Recall: 1.0
Precision: 1.0
R2Score: 1.0
ROCScore: 1.0
MAE 0.0
MSE 0.0
RMSE 0.004

 clasification report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00     45366
           1       1.00      1.00      1.00     11637

    accuracy                           1.00     57003
   macro avg       1.00      1.00      1.00     57003
weighted avg       1.00      1.00      1.00     57003


 confussion matrix:
 [[45365     1]
 [    0 11637]]
```




```
['AIRLINE_DELAY', 'AIR_SYSTEM_DELAY', 'AIR_TIME', 'ARRIVAL_DELAY',
'ARRIVAL_TIME', 'CANCELLED', 'DAY', 'DAY_OF_WEEK', 'DEPARTURE_DELAY',
'DEPARTURE_TIME', 'DISTANCE', 'DIVERTED', 'ELAPSED_TIME', 'FLIGHT_NUMBER',
'LATE_AIRCRAFT_DELAY', 'MONTH', 'SCHEDULED_ARRIVAL', 'SCHEDULED_DEPARTURE',
'SCHEDULED_TIME', 'SECURITY_DELAY', 'TAXI_IN', 'TAXI_OUT', 'WEATHER_DELAY',
'WHEELS_OFF', 'WHEELS_ON', 'YEAR']
DEPARTURE_DELAY_15
```

## Naïve Bayes

```
Printing Scores for Naive Bayes Algorithm
Accuracy: 94.825
F1 score: 0.867
Recall: 0.828
Precision: 0.91
R2Score: 0.681
ROCScore: 0.904
MAE 0.052
MSE 0.052
RMSE 0.227

 clasification report:
               precision    recall  f1-score   support

           0       0.96      0.98      0.97     45366
           1       0.91      0.83      0.87     11637

    accuracy                           0.95     57003
   macro avg       0.93      0.90      0.92     57003
weighted avg       0.95      0.95      0.95     57003


 confussion matrix:
 [[44413   953]
 [ 1997  9640]]
```




```
['AIRLINE_DELAY', 'AIR_SYSTEM_DELAY', 'AIR_TIME', 'ARRIVAL_DELAY',
'ARRIVAL_TIME', 'CANCELLED', 'DAY', 'DAY_OF_WEEK', 'DEPARTURE_DELAY',
'DEPARTURE_TIME', 'DISTANCE', 'DIVERTED', 'ELAPSED_TIME', 'FLIGHT_NUMBER',
'LATE_AIRCRAFT_DELAY', 'MONTH', 'SCHEDULED_ARRIVAL', 'SCHEDULED_DEPARTURE',
```

```
'SCHEDULED_TIME', 'SECURITY_DELAY', 'TAXI_IN', 'TAXI_OUT', 'WEATHER_DELAY',
'WHEELS_OFF', 'WHEELS_ON', 'YEAR']
DEPARTURE_DELAY_15
```

## Decision Tree

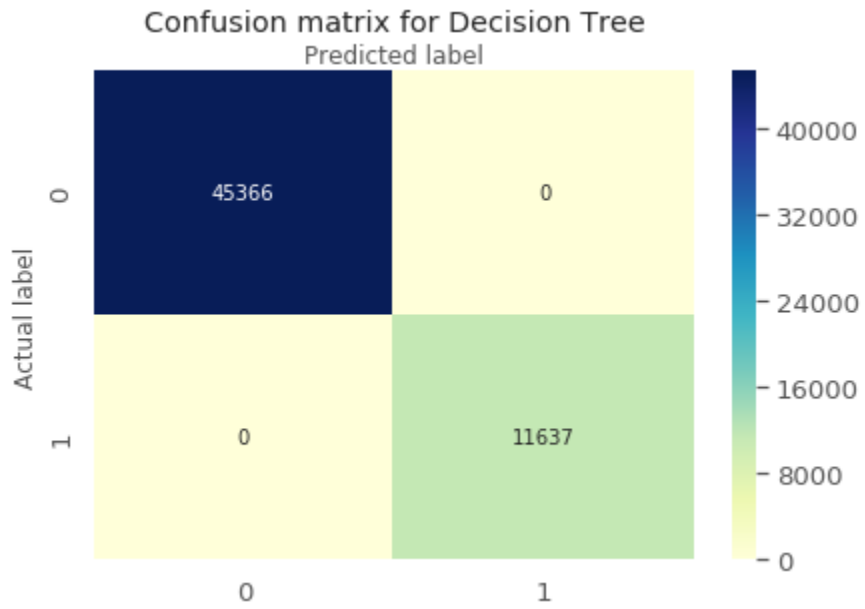
```
Printing Scores for Decision Tree Algorithm
Accuracy: 100.0
F1 score: 1.0
Recall: 1.0
Precision: 1.0
R2Score: 1.0
ROCScore: 1.0
MAE 0.0
MSE 0.0
RMSE 0.0
```

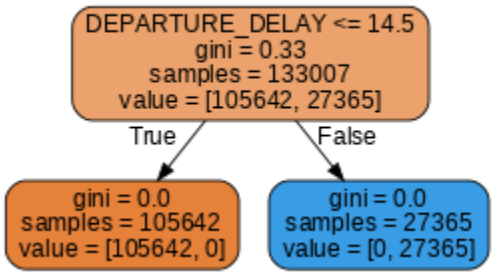
clasification report:

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 45366 |
| 1 | 1.00 | 1.00 | 1.00 | 11637 |
| accuracy | | | 1.00 | 57003 |
| macro avg | 1.00 | 1.00 | 1.00 | 57003 |
| weighted avg | 1.00 | 1.00 | 1.00 | 57003 |

confussion matrix:

```
[[45366     0]
[    0 11637]]
```

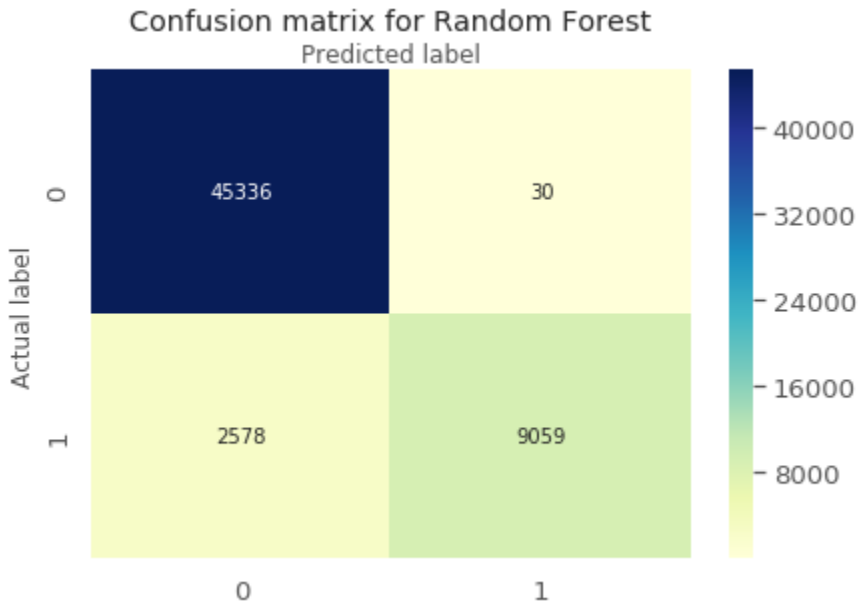


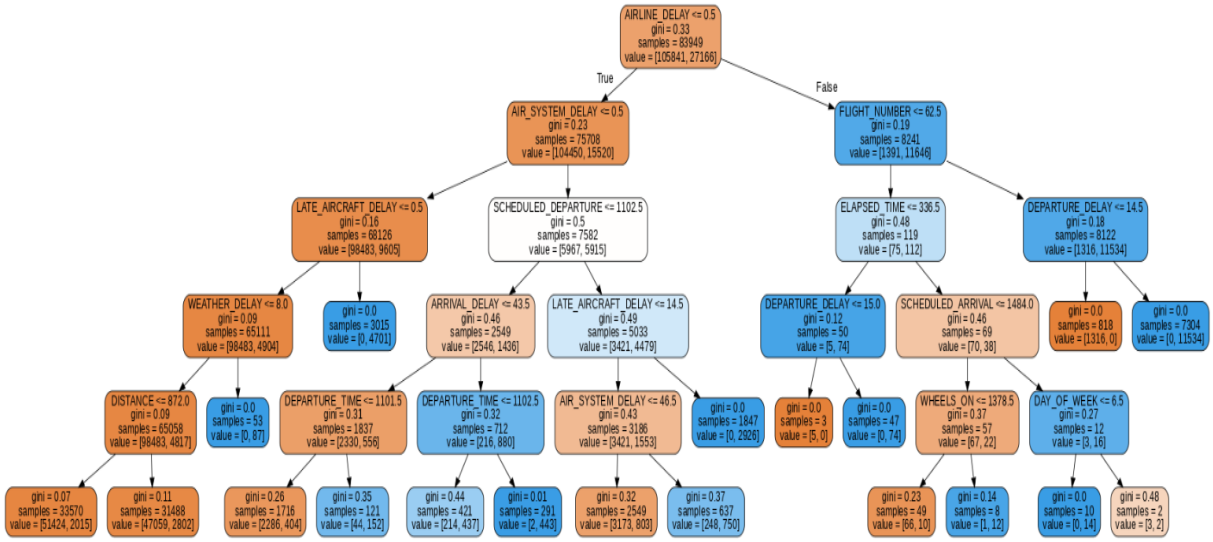



```
['AIRLINE DELAY', 'AIR SYSTEM DELAY', 'AIR TIME', 'ARRIVAL DELAY',
'ARRIVAL_TIME', 'CANCELLED', 'DAY', 'DAY_OF_WEEK', 'DEPARTURE_DELAY',
'DEPARTURE_TIME', 'DISTANCE', 'DIVERTED', 'ELAPSED_TIME', 'FLIGHT_NUMBER',
'LATE_AIRCRAFT_DELAY', 'MONTH', 'SCHEDULED_ARRIVAL', 'SCHEDULED_DEPARTURE',
'SCHEDULED_TIME', 'SECURITY_DELAY', 'TAXI_IN', 'TAXI_OUT', 'WEATHER_DELAY',
'WHEELS_OFF', 'WHEELS_ON', 'YEAR']
DEPARTURE_DELAY_15
```

## Random Forest

```
Printing Scores for Random Forest Algorithm
Accuracy: 95.425
F1 score: 0.874
Recall: 0.778
Precision: 0.997
R2Score: 0.718
ROCScore: 0.889
MAE 0.046
MSE 0.046
RMSE 0.214

 clasification report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97     45366
           1       1.00      0.78      0.87     11637

    accuracy                           0.95     57003
   macro avg       0.97      0.89      0.92     57003
weighted avg       0.96      0.95      0.95     57003


 confussion matrix:
 [[45336    30]
 [ 2578  9059]]
```

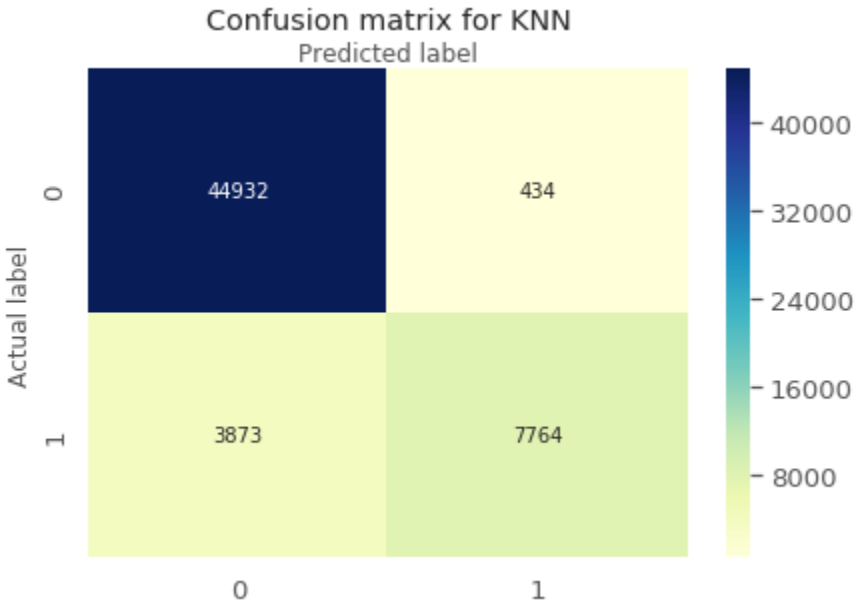



```
['AIRLINE_DELAY', 'AIR_SYSTEM_DELAY', 'AIR_TIME', 'ARRIVAL_DELAY',
'ARRIVAL_TIME', 'CANCELLED', 'DAY', 'DAY_OF_WEEK', 'DEPARTURE_DELAY',
'DEPARTURE_TIME', 'DISTANCE', 'DIVERTED', 'ELAPSED_TIME', 'FLIGHT_NUMBER',
'LATE_AIRCRAFT_DELAY', 'MONTH', 'SCHEDULED_ARRIVAL', 'SCHEDULED_DEPARTURE',
'SCHEDULED_TIME', 'SECURITY_DELAY', 'TAXI_IN', 'TAXI_OUT', 'WEATHER_DELAY',
'WHEELS_OFF', 'WHEELS_ON', 'YEAR']
DEPARTURE_DELAY_15
```

## K-NN

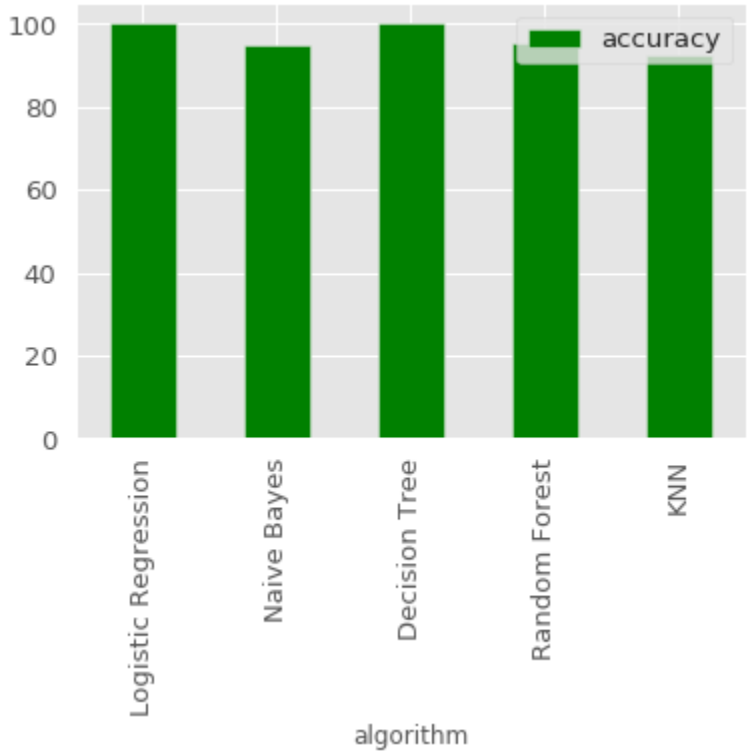
```
Printing Scores for KNN Algorithm
Accuracy: 92.444
F1 score: 0.783
Recall: 0.667
Precision: 0.947
R2Score: 0.535
ROCScore: 0.829
MAE 0.076
MSE 0.076
RMSE 0.275

 clasification report:
              precision    recall  f1-score   support

           0       0.92      0.99      0.95     45366
           1       0.95      0.67      0.78     11637

    accuracy                           0.92     57003
   macro avg       0.93      0.83      0.87     57003
weighted avg       0.93      0.92      0.92     57003


 confussion matrix:
 [[44932   434]
 [ 3873  7764]]
```




| | algorithm | accuracy | precision | recall | fscore | r2score | \ |
|---|---|---|---|---|---|---|---|
| 0 | Logistic Regression | 99.998 | 1.000 | 1.000 | 1.000 | 1.000 | |
| 1 | Naive Bayes | 94.825 | 0.910 | 0.828 | 0.867 | 0.681 | |

```
2         Decision Tree   100.000      1.000   1.000   1.000    1.000
3         Random Forest    95.425      0.997   0.778   0.874    0.718
4                   KNN    92.444      0.947   0.667   0.783    0.535

   rocscore    mae    mse   rmse
0     1.000  0.000  0.000  0.004
1     0.904  0.052  0.052  0.227
2     1.000  0.000  0.000  0.000
3     0.889  0.046  0.046  0.214
4     0.829  0.076  0.076  0.275
```

Barplot of accuracy score of all the above algorithms:




## Accuracy score and metric calculation of algorithms for departure delay

```
departure_performance_df = algorithms_performace_df.copy()
departure_performance_df.head(10)
```

| | algorithm | accuracy | precision | recall | fscore | r2score | rocscore | mae | mse | rmse |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Logistic Regression | 99.998 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.000 | 0.004 |
| 1 | Naive Bayes | 94.825 | 0.910 | 0.828 | 0.867 | 0.681 | 0.904 | 0.052 | 0.052 | 0.227 |
| 2 | Decision Tree | 100.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.000 | 0.000 |
| 3 | Random Forest | 95.425 | 0.997 | 0.778 | 0.874 | 0.718 | 0.889 | 0.046 | 0.046 | 0.214 |
| 4 | KNN | 92.444 | 0.947 | 0.667 | 0.783 | 0.535 | 0.829 | 0.076 | 0.076 | 0.275 |

## Accuracy score and metric calculation of algorithms for arrival delay

```
arrival_performance_df = algorithms_performace_df.copy()
arrival_performance_df.head(10)
```

| | algorithm | accuracy | precision | recall | fscore | r2score | rocscore | mae | mse | rmse |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Logistic Regression | 99.982 | 1.000 | 1.000 | 1.000 | 0.999 | 1.000 | 0.000 | 0.000 | 0.013 |
| 1 | Naive Bayes | 99.974 | 0.999 | 1.000 | 0.999 | 0.998 | 1.000 | 0.000 | 0.000 | 0.016 |
| 2 | Decision Tree | 100.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.000 | 0.000 |
| 3 | Random Forest | 99.988 | 1.000 | 0.999 | 1.000 | 0.999 | 1.000 | 0.000 | 0.000 | 0.011 |
| 4 | KNN | 91.843 | 0.958 | 0.664 | 0.785 | 0.530 | 0.828 | 0.082 | 0.082 | 0.286 |

## Model Evaluation

Decision Tree has the highest Accuracy score for both Departure and Arrival delay>15 minutes, since the True negative and False positive value is zero and the predicted result is nearly equal to the actual result. Among the models applied, we find that most of the classifiers has the highest ROC value (~1), which is a considerable improvement over our Logistic Regression baseline model (1).

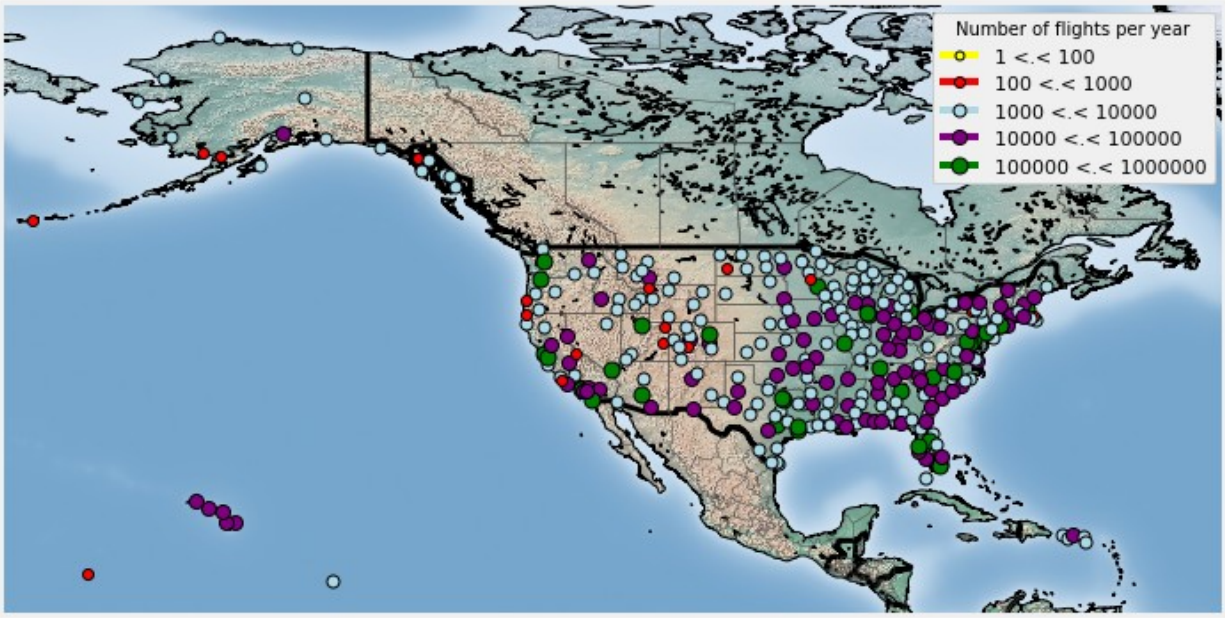
## Model's ability to solve the business problem

The model that we are trying to build is beneficial to the travelers, airlines and the SFO authorities. For the travellers, it would help them plan their schedule accordingly. Airport

authorities and airlines will benefit as they can take into account the possibility of the delay and take preventive measures. As mentioned above, we are using ROC to evaluate our model's performance as it is base rate invariant. ROC is a measure of our model's ability to separate between the two classes and so a good ROC indicates a good class separation capacity. Thus, our model has a reasonable performance and can predict delays with a significant confidence.

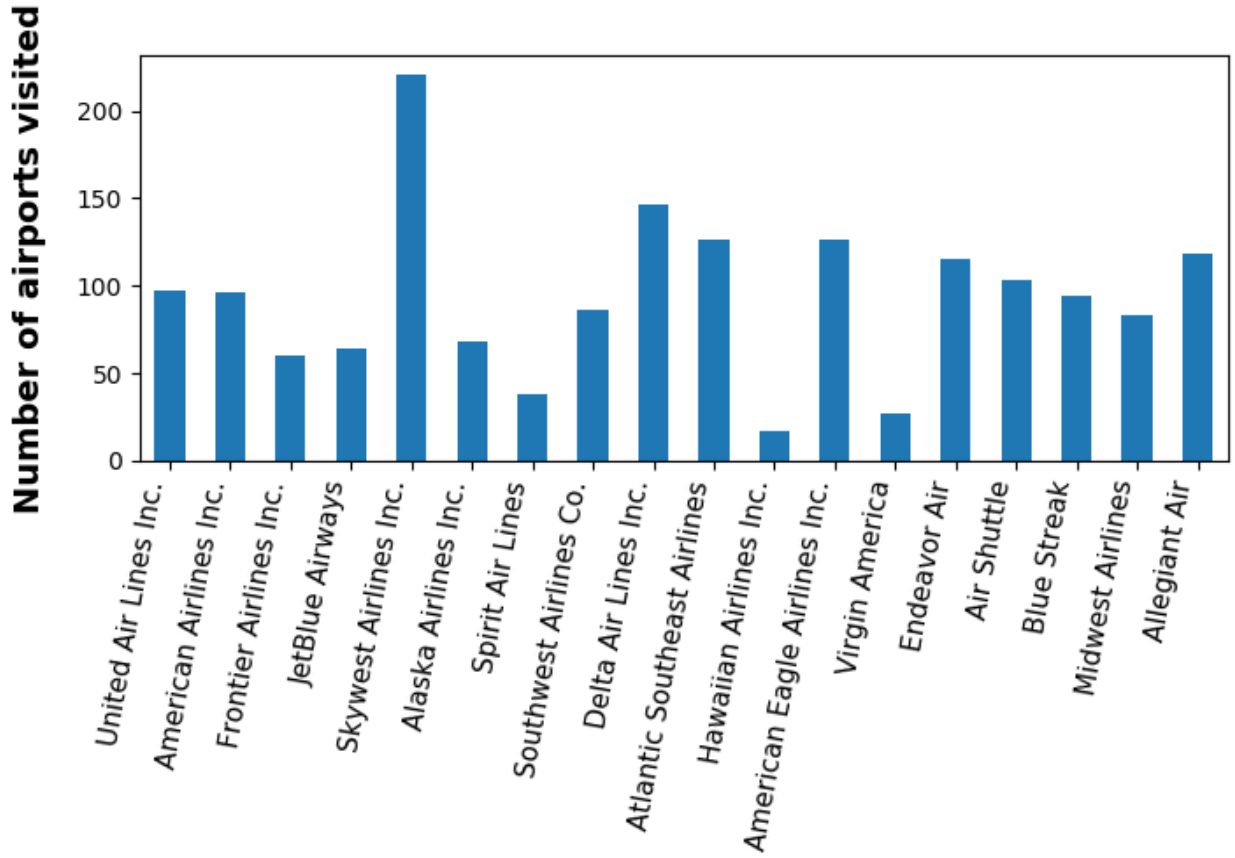
# Analysis

To have a global overview of the geographical area covered in this dataset, we can plot the airports location and indicate the number of flights recorded during year 2018 in each of them:

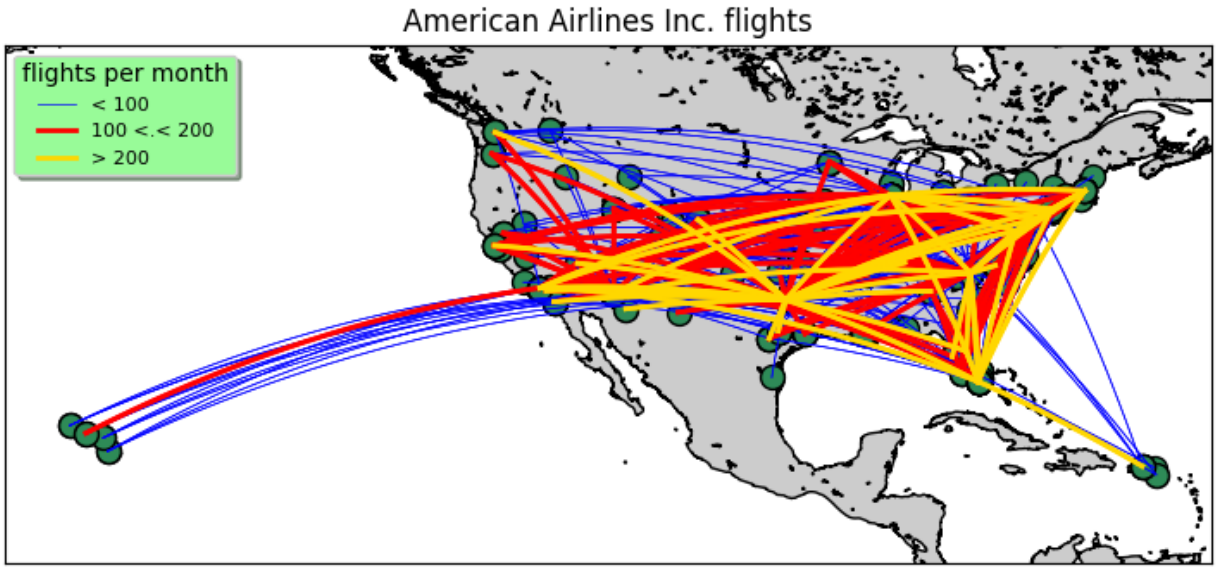


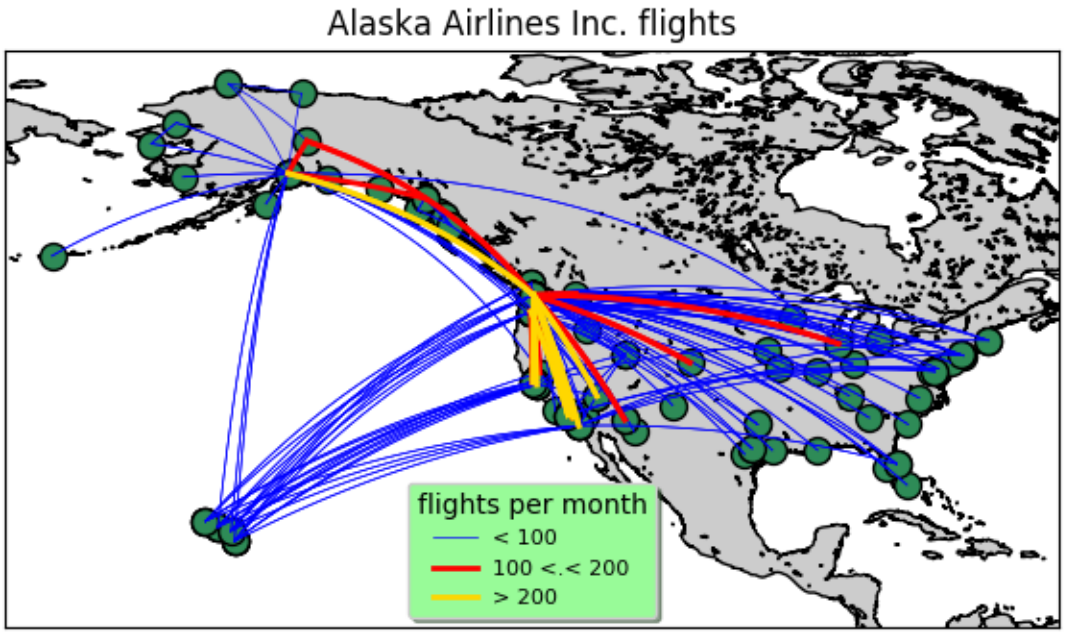

We are plotting departure delay and arrival delay for same airlines between same source and destinations. From the chart, we can deduce that Arrival delay is lesser than the departure delay which means flights are being adjusted with their travel speed to reach destinations on time. Also, most of the delays are happening at during the departure.

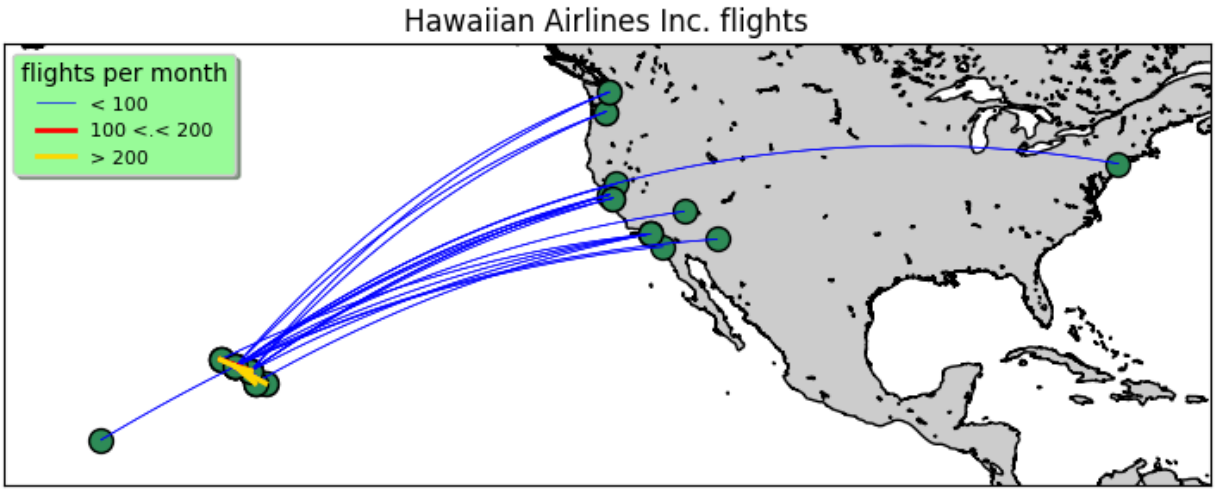
## Number of most visited airports in USA




## Basemap plot for flight paths for American, Alaska, Hawain airlines

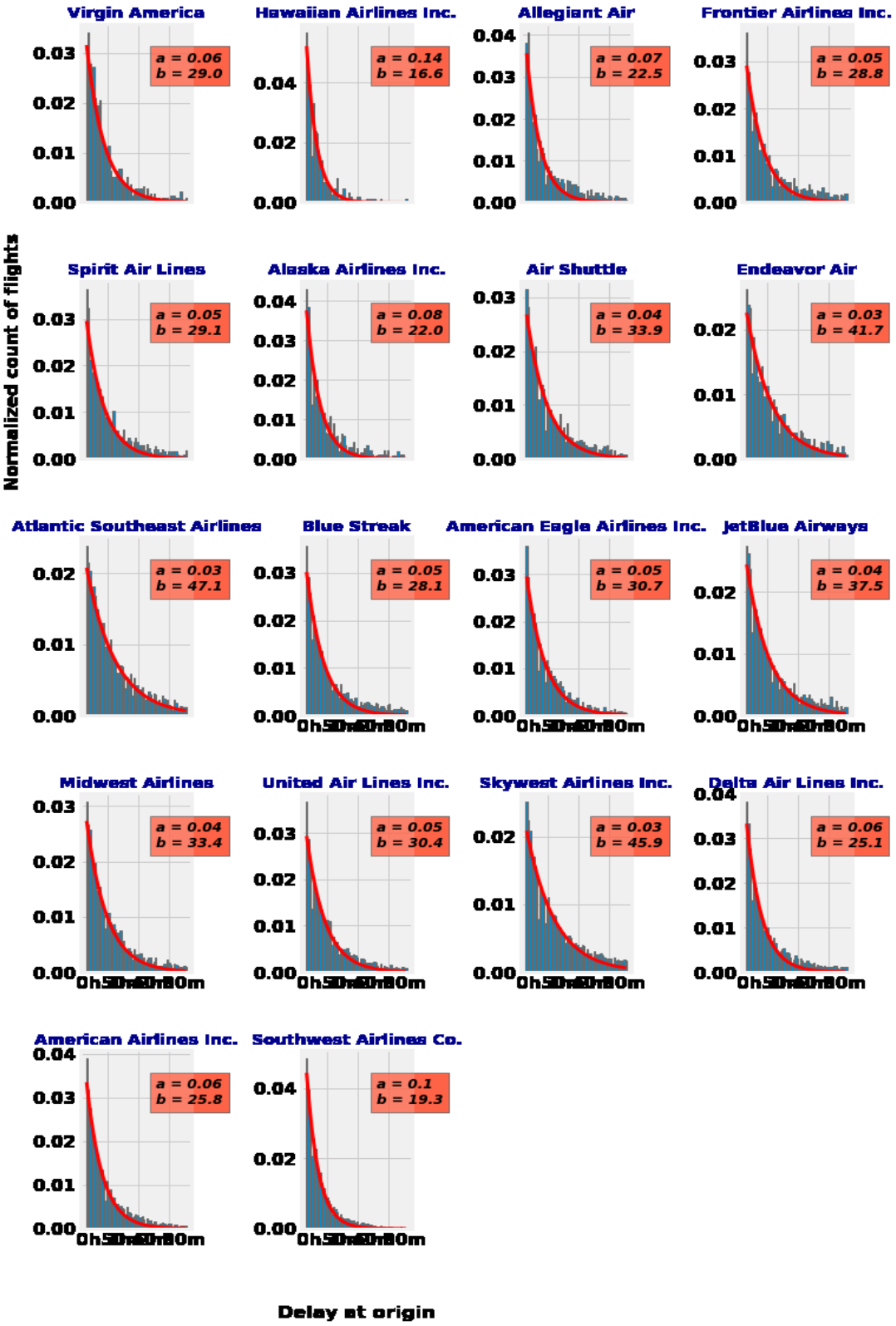




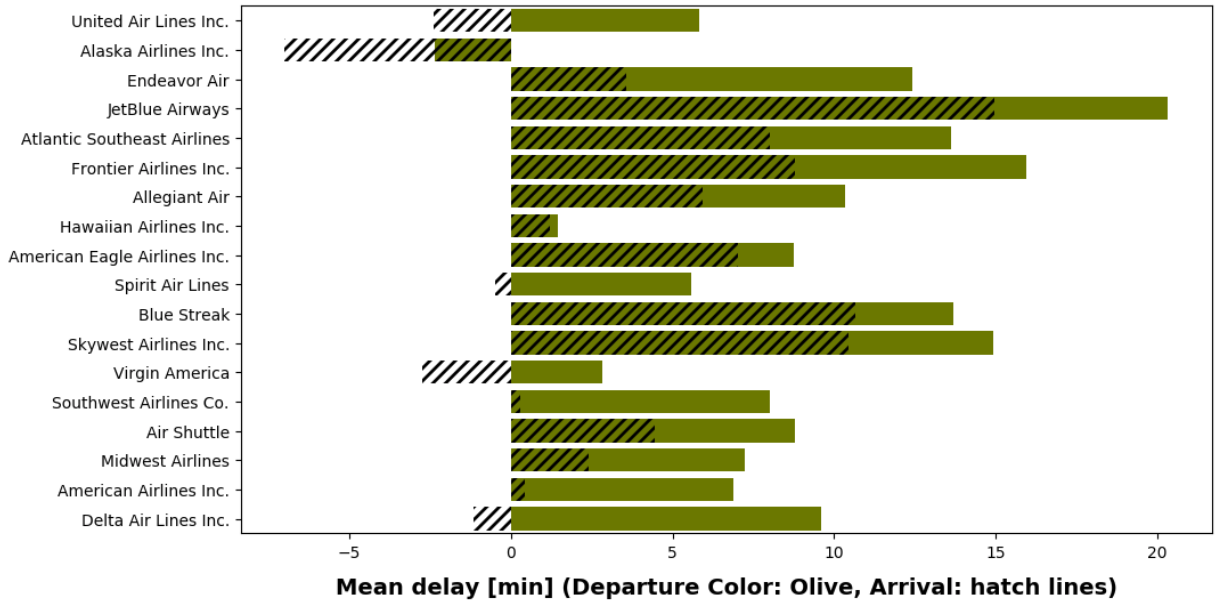




## Delays distribution: establishing the ranking of airlines

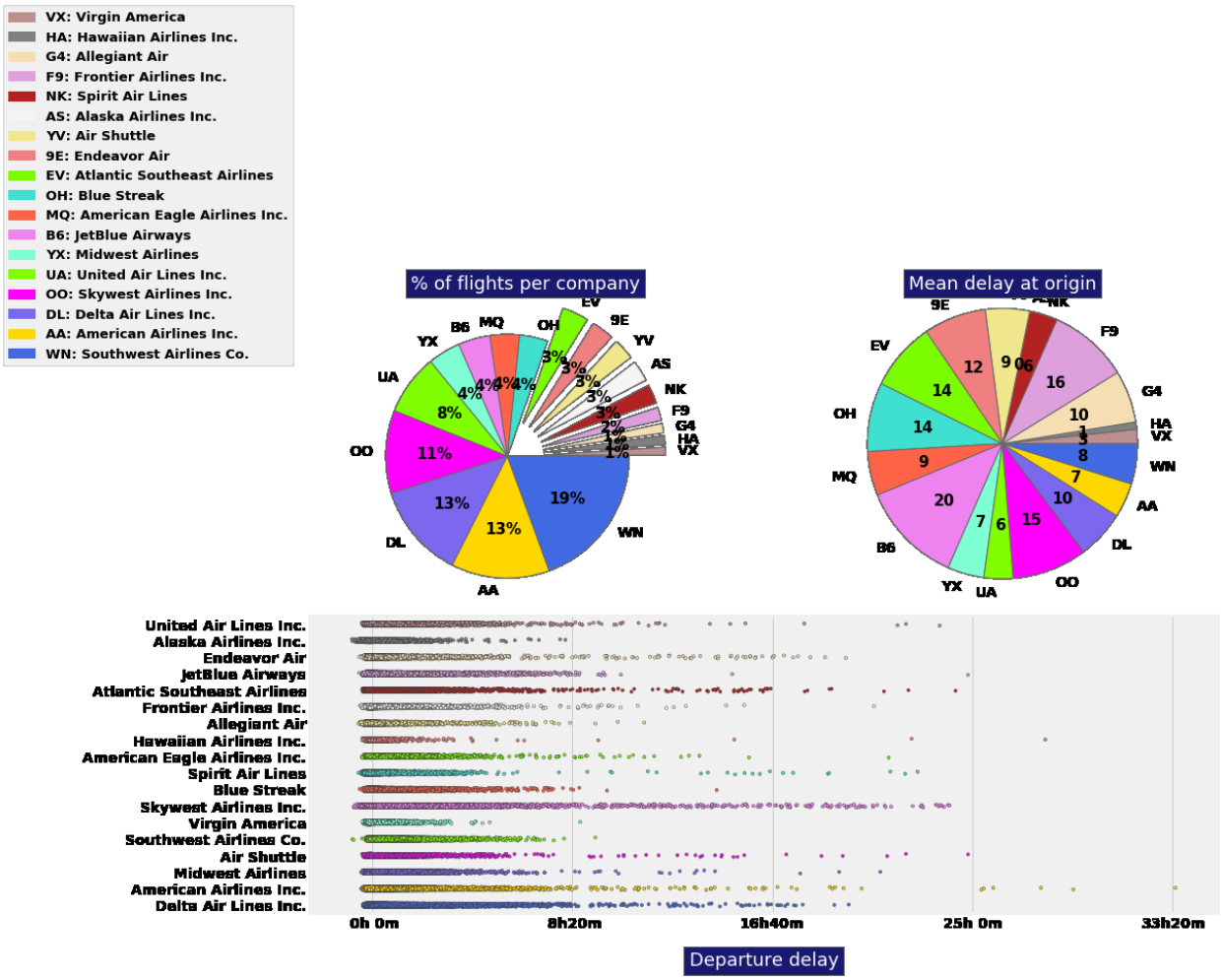
It was shown in the previous section that mean delays behave homogeneously among airlines (apart from two extrem cases) and is around 14±±7 minutes. Then, we saw that this low value is a consequence of the large proportion of flights that take off on time. However, occasionally, large delays can be registred. In this section, I examine more in details the distribution of delays for every airlines:


Virgin America
a = 0.06
b = 29.0
Hawaiian Airlines Inc.
a = 0.14
b = 16.6
Allegiant Air
a = 0.07
b = 22.5
Frontier Airlines Inc.
a = 0.05
b = 28.8
Spirit Air Lines
a = 0.05
b = 29.1
Alaska Airlines Inc.
a = 0.08
b = 22.0
Air Shuttle
a = 0.04
b = 33.9
Endeavor Air
a = 0.03
b = 41.7
Atlantic Southeast Airlines
a = 0.03
b = 47.1
Blue Streak
a = 0.05
b = 28.1
American Eagle Airlines Inc.
a = 0.05
b = 30.7
JetBlue Airways
a = 0.04
b = 37.5
Midwest Airlines
a = 0.04
b = 33.4
United Air Lines Inc.
a = 0.05
b = 30.4
Skywest Airlines Inc.
a = 0.03
b = 45.9
Delta Air Lines Inc.
a = 0.06
b = 25.1
American Airlines Inc.
a = 0.06
b = 25.8
Southwest Airlines Co.
a = 0.1
b = 19.3
Normalized count of flights
Delay at origin

This figure shows the normalised distribution of delays that I modelised with an exponential distribution f(x)=aexp(−x/b)f(x)=aexp(−x/b). The aa et bb parameters obtained to describe each airline are given in the upper right corner of each panel. Note that the normalisation of the distribution implies that ∫f(x)dx∼1∫f(x)dx∼1. Here, we do not have a strict equality since the normalisation applies the histograms but not to the model function. However, this relation entails that the aa et bb coefficients will be correlated with a∝1/ba∝1/b and hence, only one of these two values is necessary to describe the distributions. Finally, according to the value of either aa or bb, it is possible to establish a ranking of the companies: the low values of aa will correspond to airlines with a large proportion of important delays and, on the contrary, airlines that shine from their punctuality will admit hight aa values:




We consider all the flights from all carriers. Here, the aim is to classify the airlines with respect to their punctuality.

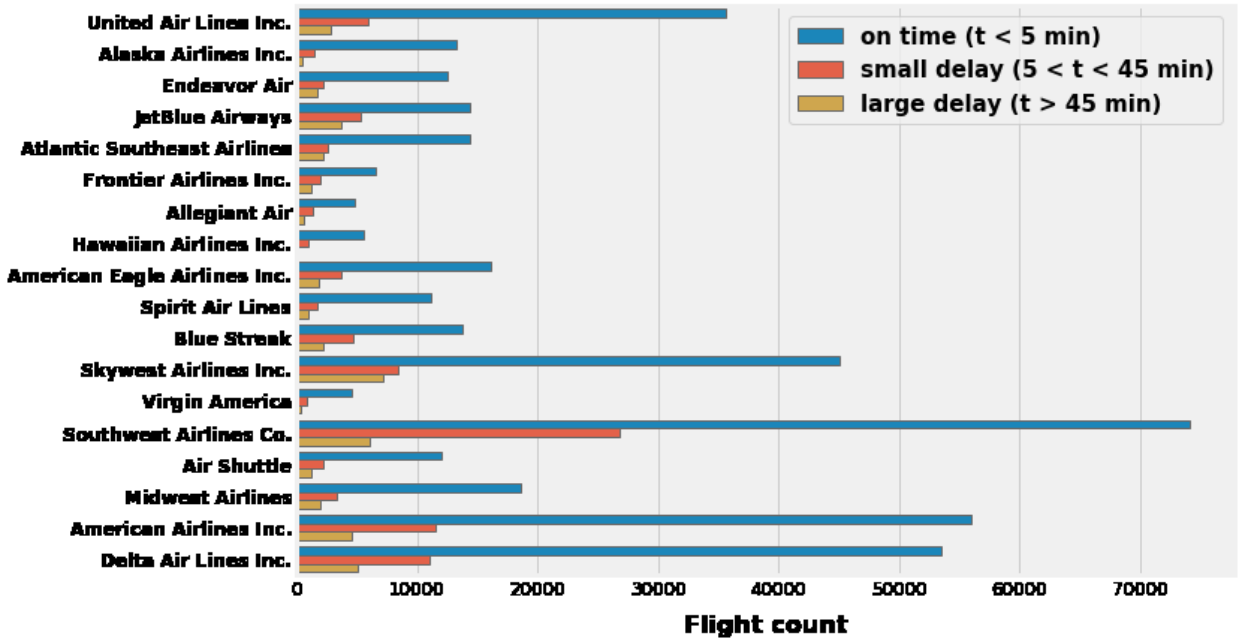



Considering the first pie chart that gives the percentage of flights per airline, we see that there is some disparity between the carriers. For example, *Southwest Airlines* accounts for ∼∼19% of the flights which is similar to the number of flights chartered by the 8 tiniest airlines. However, if we have a look at the second pie chart, we see that here, on the contrary, the differences among airlines are less pronounced. Excluding *Hawaiian Airlines* and *Virgin America* that report extremely low mean delays, we obtain that a value of ∼∼**14±±7 minutes** would correctly represent all mean delays. Note that this value is quite low which mean that the standard for every airline is to respect the schedule.

## How large is the delay for each airlines?

Code to plot the delay for each airlines:

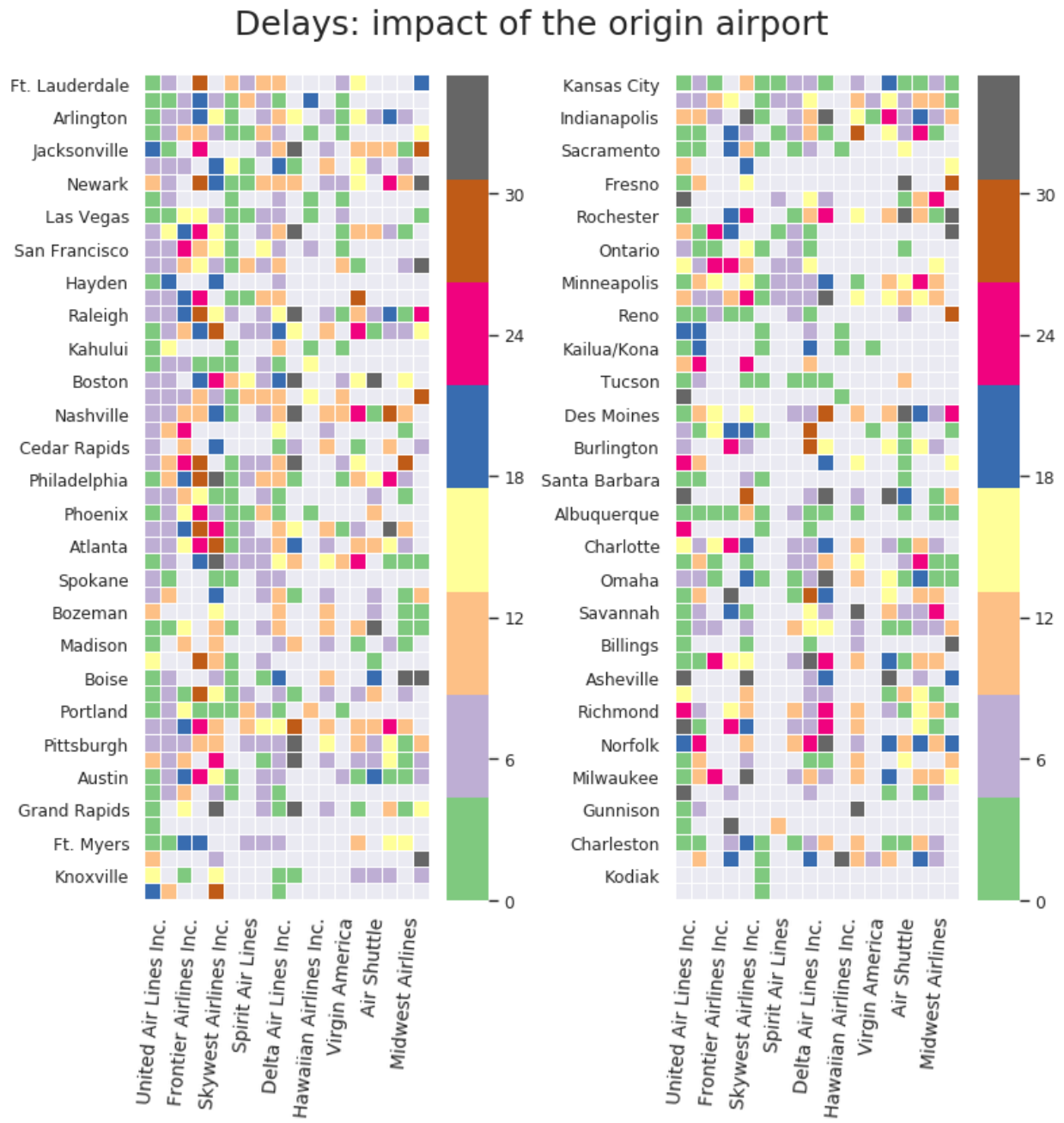
```
# Function that define how delays are grouped
delay_type = lambda x:((0,1)[x > 5],2)[x > 45]
df['DELAY_LEVEL'] = df['DEPARTURE_DELAY'].apply(delay_type)
#_____________________________________________________
fig = plt.figure(1, figsize=(10,7))
ax = sns.countplot(y="AIRLINE", hue='DELAY_LEVEL', data=df)
#_____________________________________________________________________________
________
# We replace the abbreviations by the full names of the companies and set the
labels
labels = [abbr_companies[item.get_text()] for item in ax.get_yticklabels()]
ax.set_yticklabels(labels)
plt.setp(ax.get_xticklabels(), fontsize=12, weight = 'normal', rotation = 0);
plt.setp(ax.get_yticklabels(), fontsize=12, weight = 'bold', rotation = 0);
ax.yaxis.label.set_visible(False)
plt.xlabel('Flight count', fontsize=16, weight = 'bold', labelpad=10)
#_________________
# Set the legend
L = plt.legend()
L.get_texts()[0].set_text('on time (t < 5 min)')
L.get_texts()[1].set_text('small delay (5 < t < 45 min)')
L.get_texts()[2].set_text('large delay (t > 45 min)')
plt.show()
```




South West airlines has more number of delays ranging from 5 to 45 minutes. Skywest airlines has more number of flight delays greater than 45 minutes.

## Mean Delays at Origin Aiports

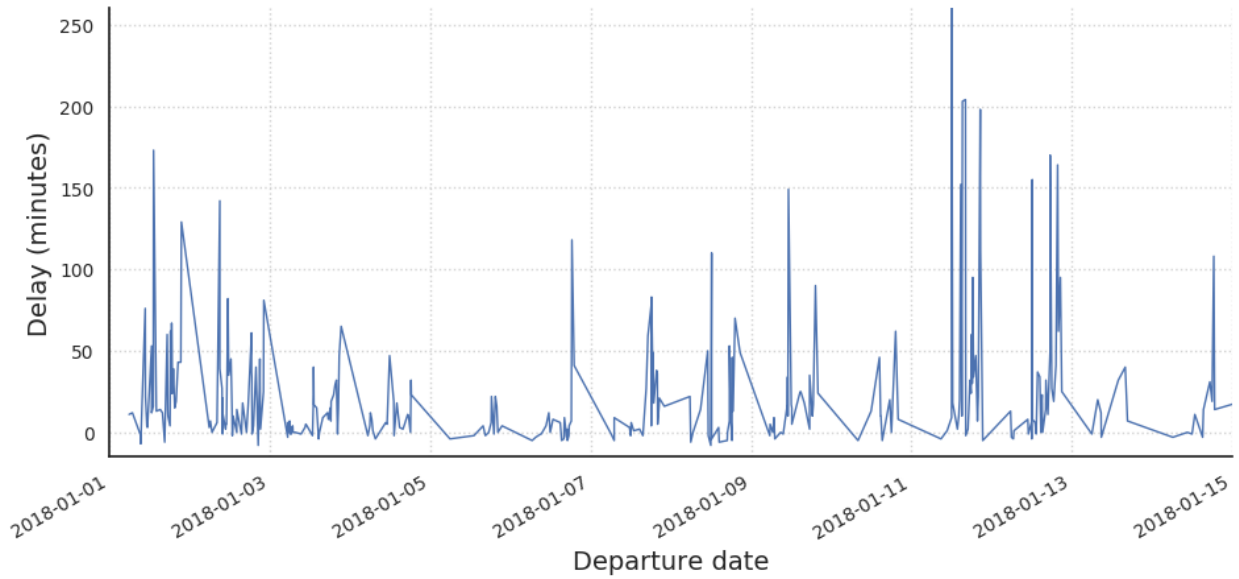
```
airport_mean_delays = pd.DataFrame(pd.Series(df['ORIGIN_AIRPORT'].unique()))
airport_mean_delays.set_index(0, drop = True, inplace = True)

for carrier in abbr_companies.keys():
    df1 = df[df['AIRLINE'] == carrier]
    test =
df1['DEPARTURE_DELAY'].groupby(df['ORIGIN_AIRPORT']).apply(get_stats).unstack
()
    airport_mean_delays[carrier] = test.loc[:, 'mean']
```




This figure allows to draw some conclusions. First, by looking at the data associated with the different airlines, we find the behavior we previously observed: for example, if we consider the right panel, it will be seen that the column associated with *United Airlines* mostly reports large delays, while the column associated with *Delta Airlines* is mainly associated with delays of less than 5 minutes.

Finally, we can deduce from these observations that there is a high variability in average delays, both between the different airports but also between the different airlines. This is important because it implies that in order to accurately model the delays, it will be necessary to adopt a model that is **specific to the company and the home airport** .

## Temporal variability of delays

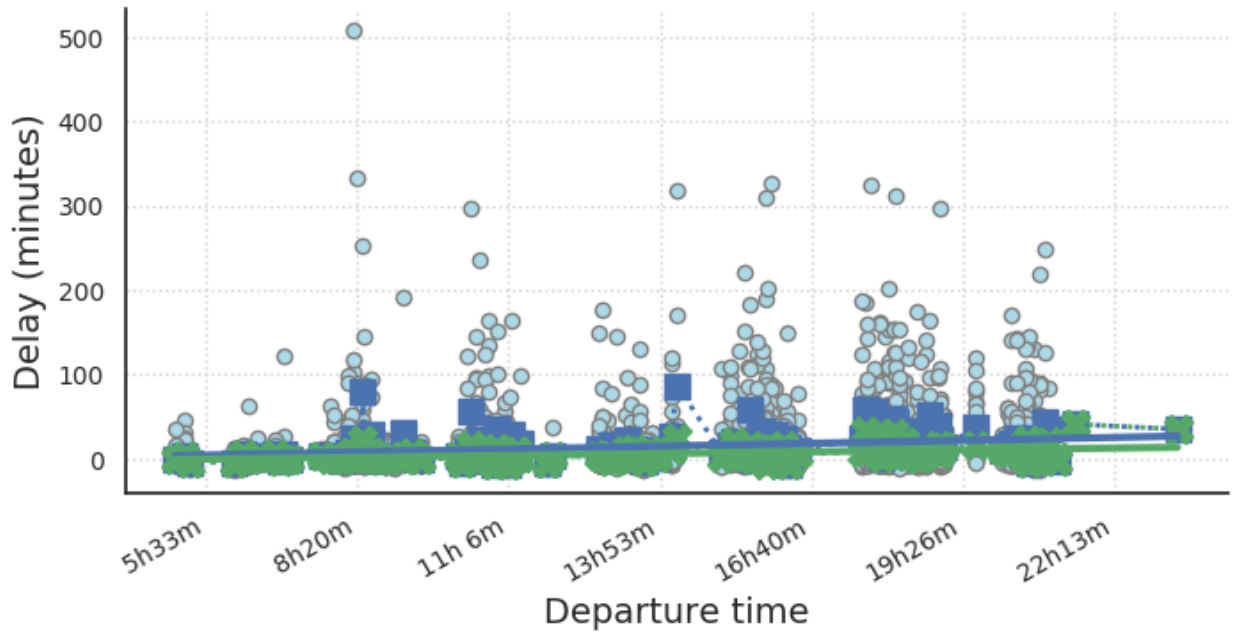



This figure shows the existence of cycles, both in the frequency of the delays but also in their magnitude. In fact, intuitively, it seems quite logical to observe such cycles since they will be a consequence of the day-night alternation and the fact that the airport activity will be greatly reduced (if not inexistent) during the night. This suggests that a **important variable** in the modeling of delays will be **take-off time**. To check this hypothesis, I look at the behavior of the mean delay as a function of departure time, aggregating the data of the current month:

## Linear Regression

Below figure, the points corresponding to the individual flights are represented by the points in gray. The mean of these points gives the mean delays and the mean of the set of initial points corresponds to the blue squares. By removing extreme delays (> 1h), one obtains the average delays represented by the green crosses. Thus, in the first case, the fit (solid blue curve) leads to a prediction which corresponds to an average delay of ∼∼ 10 minutes larger than the predicton obtained in the second case (green curve), and this, at any hour of the day.

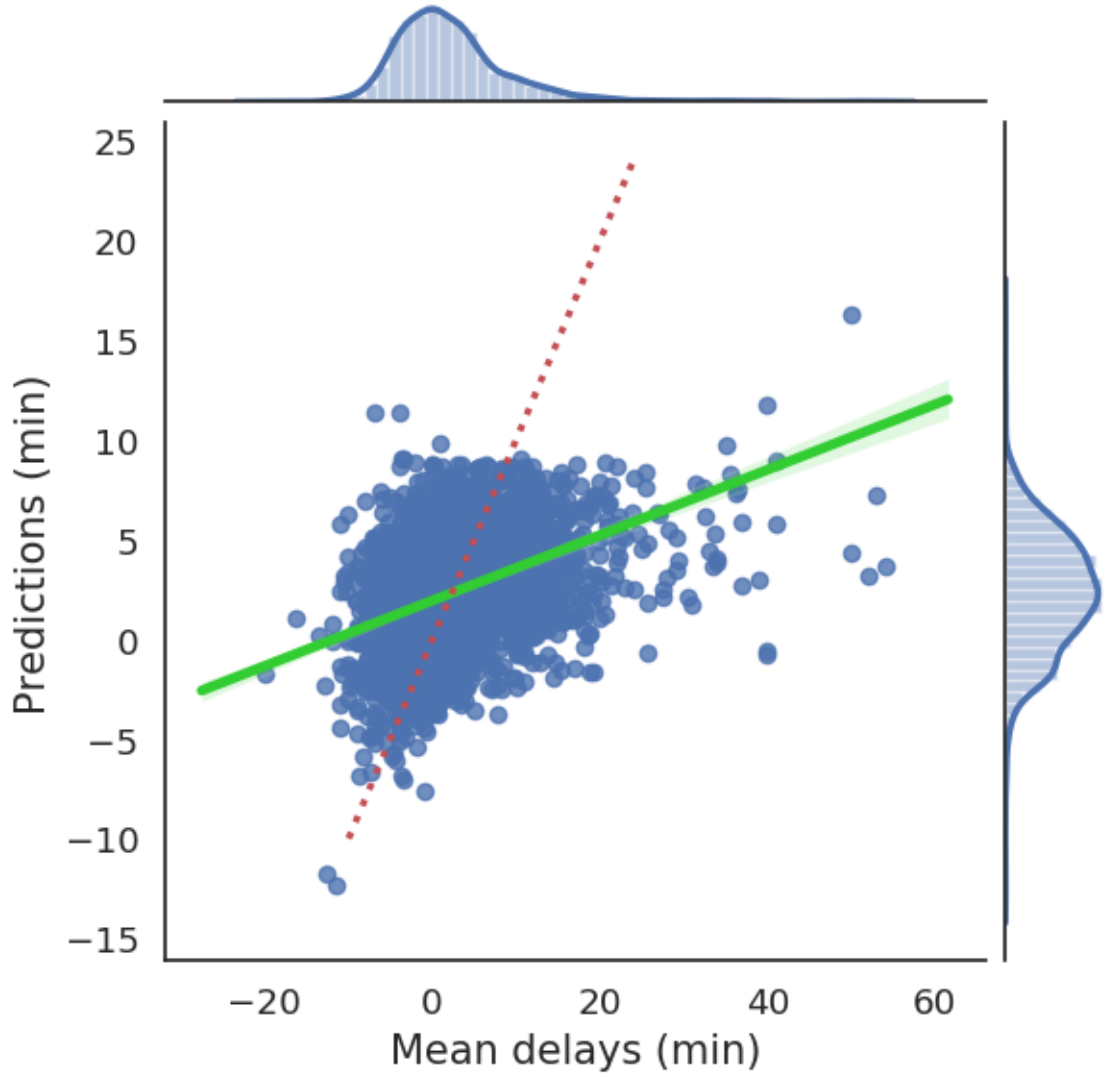
In conclusion, we see here that the way in which we manage the extreme delays will have an important impact on the modeling. Note, however, that the current example corresponds to a *chosen case* where the impact of extreme delays is magnified by the limited number of flights. Presumably, the impact of such delays will be less pronounced in the majority of cases.




The matrices X and Y thus created can be used to perform a linear regression. Calculation of the the MSE score of the fit. In practice, we can have a feeling of the quality of the fit by considering the number of predictions where the differences with real values is greater than 15 minutes. In practice, this model tends to underestimate the large delays, which can be seen in the following figure:

```
lm = linear_model.LinearRegression()
model = lm.fit(X,Y)
predictions = lm.predict(X)
print("MSE =", metrics.mean_squared_error(predictions, Y))

icount = 0
for i, val in enumerate(Y):
    if abs(val-predictions[i]) > 15: icount += 1
'{:.2f}%'.format(icount / len(predictions) * 100)
```

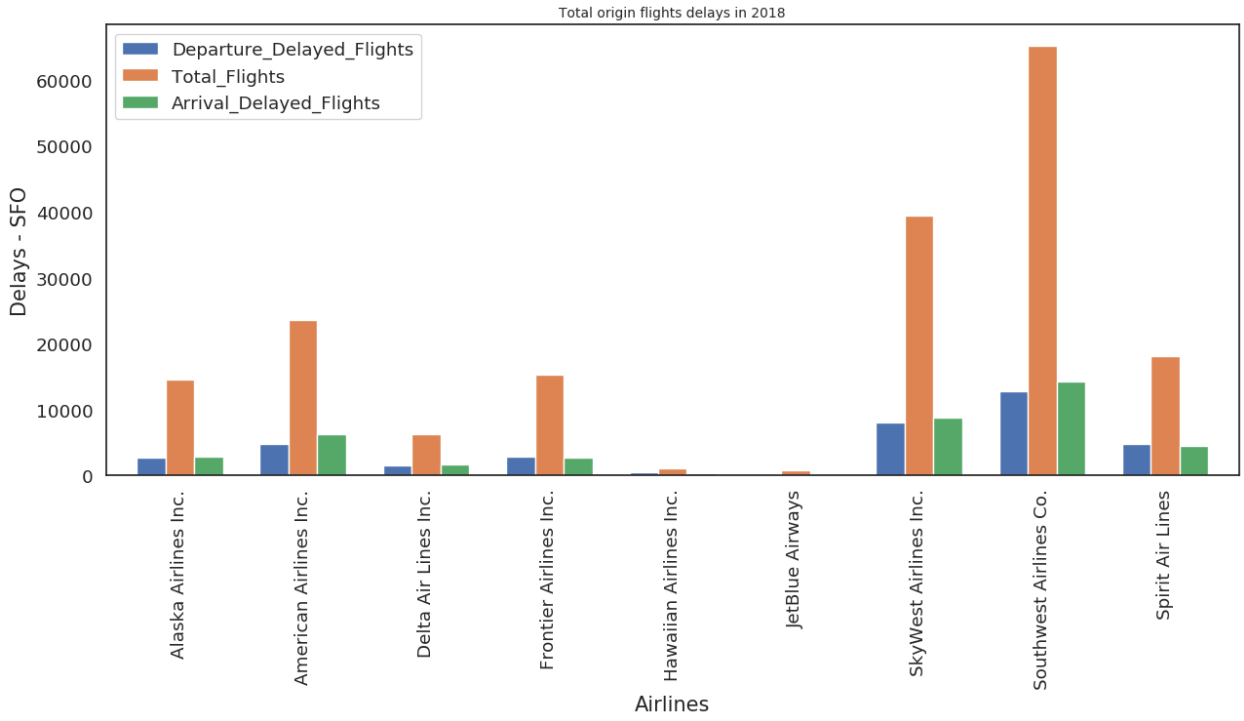
```
MSE = 42.20669284187206

'3.04%'
```



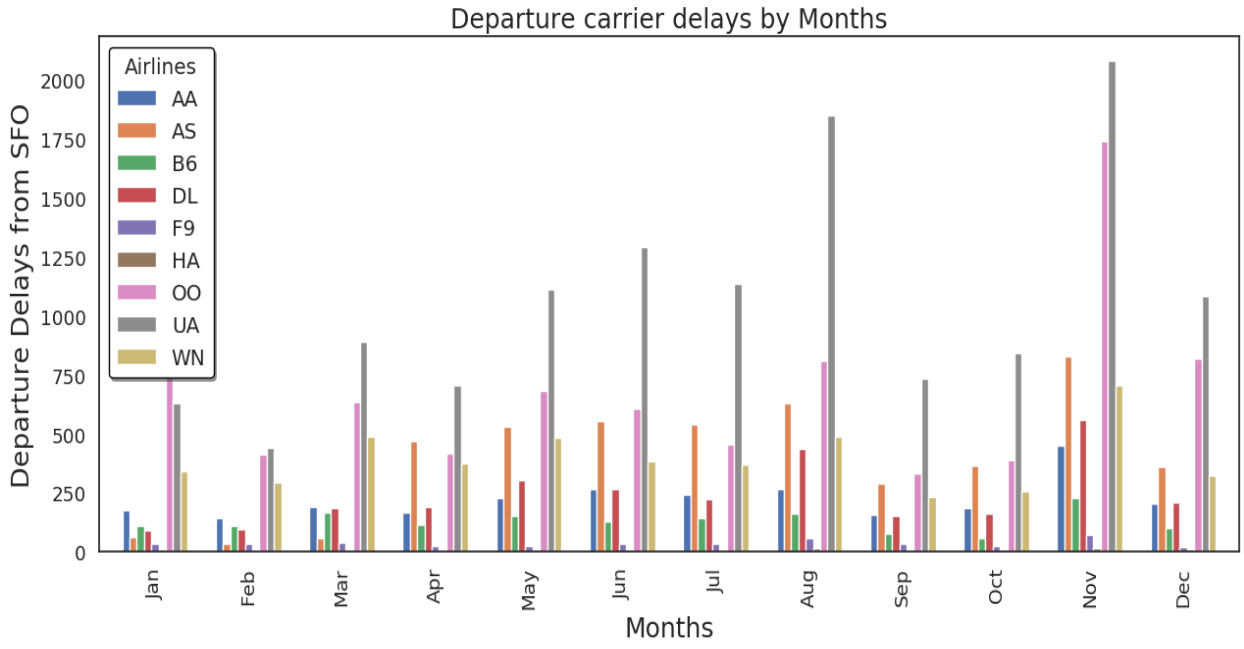

# SFO Flight Delay Analysis

Utilizing the NOAA Weather Statistics database our team will perform a month by month analysis of flight Delay incidents for nine major airline carriers arriving to and departing from the SFO airport for the 2018 year.

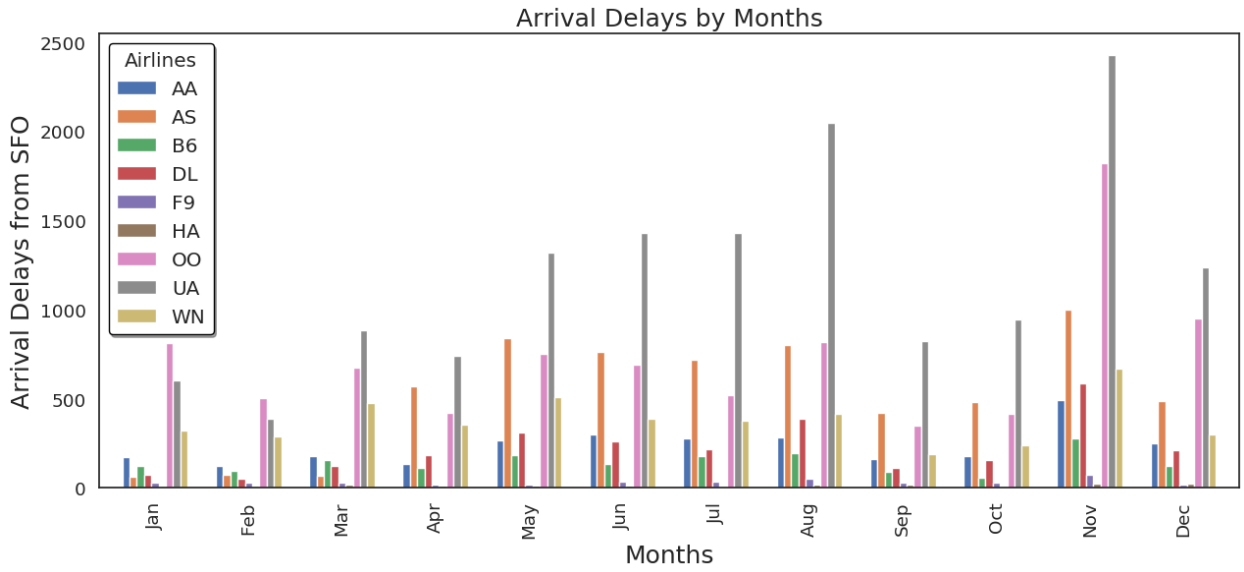
Departure and Arrival delay of total flights vs Airlines to and from SFO Airport.




Departure Delay by months for nine major airlines in SFO Airport.



Arrival Delay by months for nine major airlines in SFO Airport.




## Monthly Delay cause

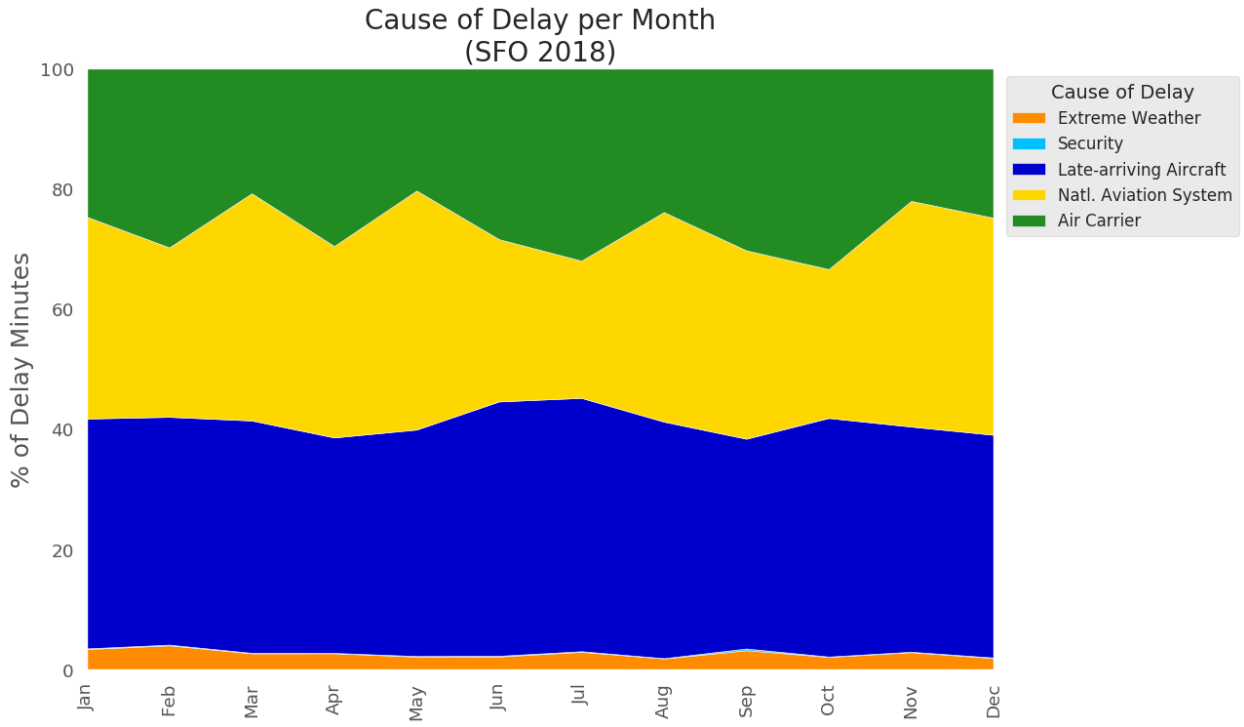
```
months = calendar.month_abbr[1:13]
x_axis = np.arange(0, 12, 1)
cause = ['Extreme Weather','Security','Late-arriving Aircraft','Natl.
Aviation System','Air Carrier']
      #['WEATHER_DELAY', 'SECURITY_DELAY', 'LATE_AIRCRAFT_DELAY',
'AIR_SYSTEM_DELAY', 'AIRLINE_DELAY']

plt.figure(figsize=(12,8))
plt.stackplot(x_axis,
              monthly_percentage_df['WEATHER_DELAY'],
              monthly_percentage_df['SECURITY_DELAY'],
              monthly_percentage_df['LATE_AIRCRAFT_DELAY'],
              monthly_percentage_df['AIR_SYSTEM_DELAY'],
              monthly_percentage_df['AIRLINE_DELAY'],
              colors=['#FF8C00','#00BFFF','#0000CD','#FFD700','#228B22'],
              labels=cause)
plt.xticks(x_axis, months, rotation='vertical')
plt.title('Cause of Delay per Month\n(SFO 2018)', size=20)
plt.ylabel('% of Delay Minutes', size=18)

legend = plt.legend(bbox_to_anchor=(1.28,1), loc="upper right", fontsize=12)
legend.set_title('Cause of Delay', prop={'size':14})

plt.margins(0)
#plt.savefig('images/monthly_delay_cause.png', bbox_extra_artists=(legend,),
bbox_inches='tight')
plt.show()
```

The causes of delay per month for year 2018 can be categorized into Extreme weather delay ~4%, Late-arriving Aircraft delay 20-40%, NAS delay 40-60%, Air carrier delay 80-100%.

**Air Carrier Delay** - Carrier delay is within the control of the air carrier. The aviation data from BTS helps us account for this delay.

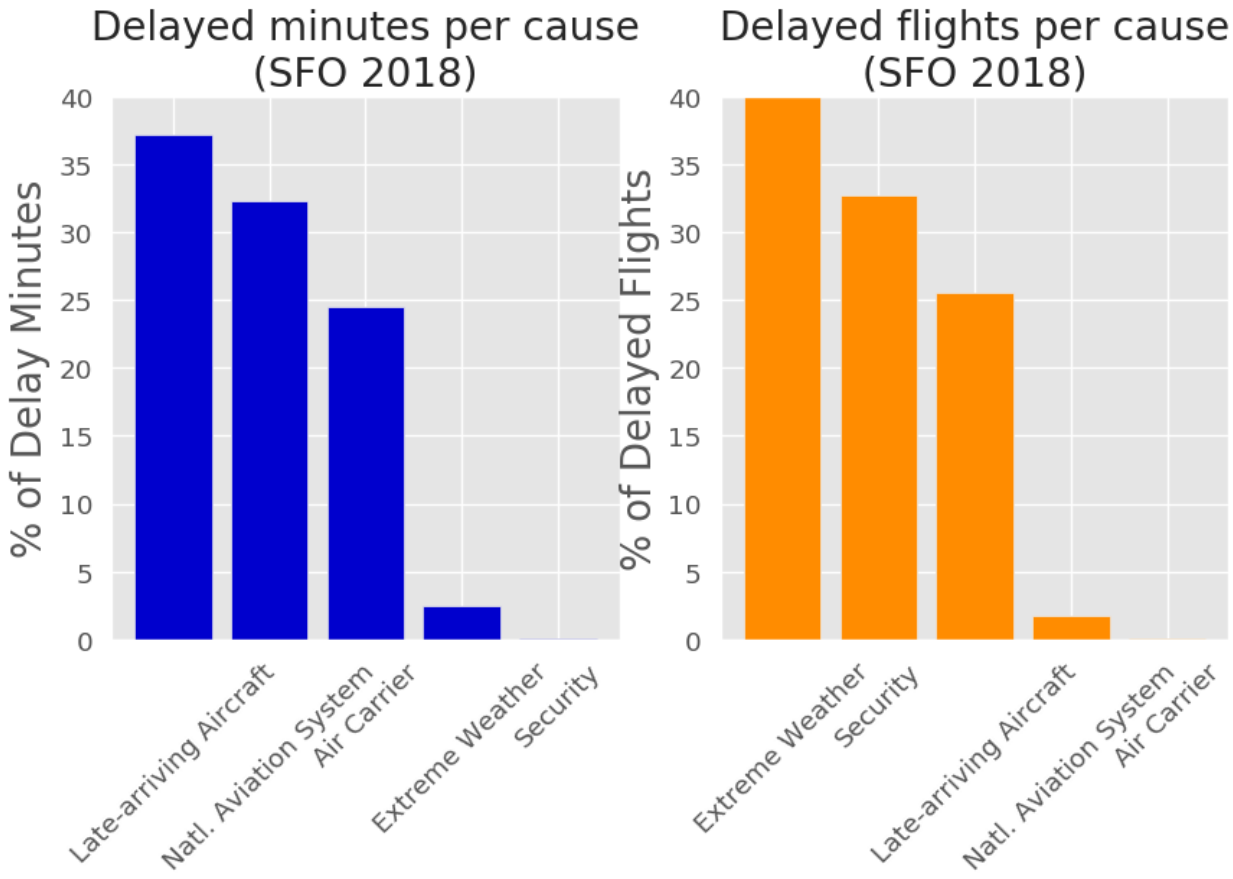
**Weather Delay** - Weather delay is caused by extreme or hazardous weather conditions that are forecasted or occur at the point of departure and point of arrival. To account for this delay, we used the weather data at the point of departure. Data was collected from the National Oceanic and Atmospheric Administration (NOAA) website which included features like wind speed, precipitation (in mm), snow (in mm) and average temperature amongst other binary variables indicating the weather condition.

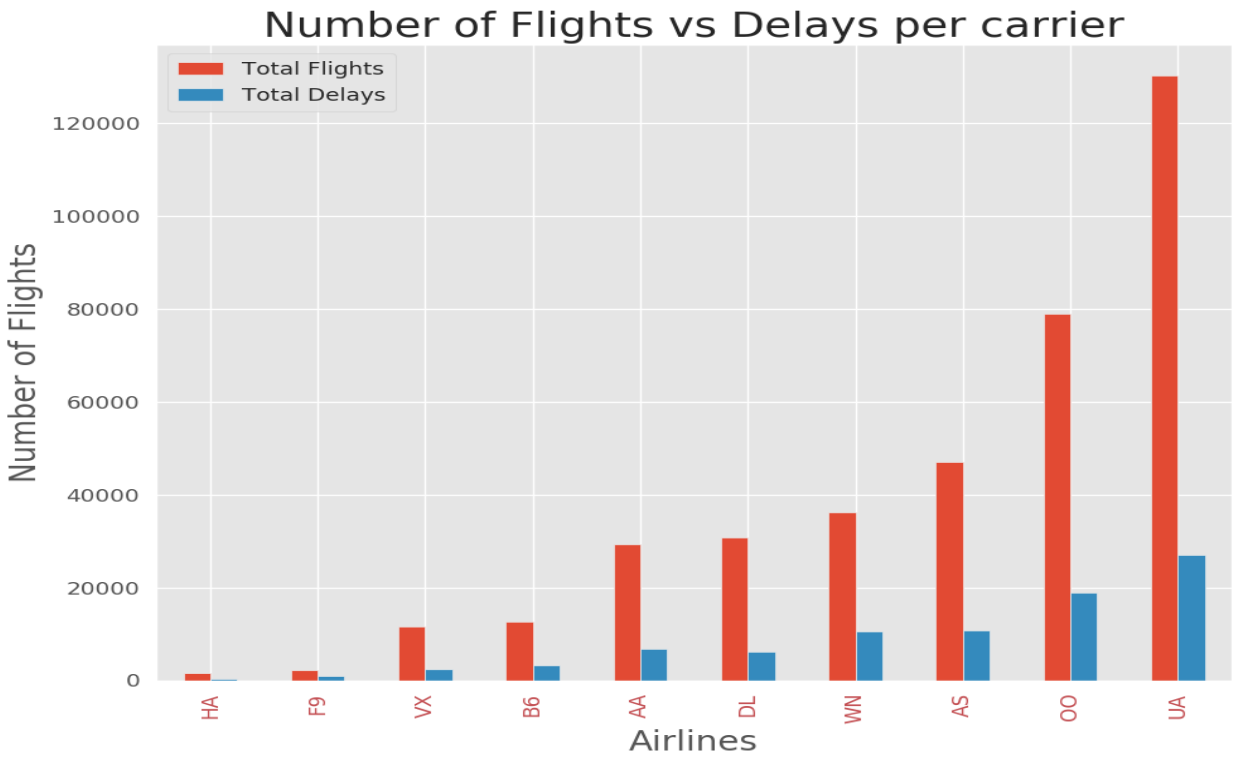
**National Aviation System Delay** - These delays are within the control of the National Airspace System (NAS) and may include: non-extreme weather conditions, airport operations, heavy traffic volume, air traffic control, etc. To account for these delays, we considered heavy traffic volume around the year 2018.



## Delay per cause by year

```
#Delayed Minutes
outbound_delay_minutes = int(outbound_delays['DEPARTURE_DELAY'].sum())
inbound_delay_minutes = int(inbound_delays['ARRIVAL_DELAY'].sum())
total_delay_minutes = outbound_delay_minutes + inbound_delay_minutes

per_cause_delay_minutes = [round(x) for x in all_cause_year.sum().tolist()]

#Percentage delays per cause
percent_minutes_delay = [round((x/total_delay_minutes)*100, 2) for x in
per_cause_delay_minutes]

percent_df = pd.DataFrame()
percent_df['Cause'] = ['Extreme Weather','Security','Late-arriving
Aircraft','Natl. Aviation System','Air Carrier']
percent_df['Percent delay'] = percent_minutes_delay
percent_df = percent_df.sort_values('Percent delay',
ascending=False).reset_index(drop=True)

delay_counts = all_cause_bool.sum()

percent_delay_flight = [round((x/delay_counts.sum())*100, 2) for x in
delay_counts]
percent_delay_flight
percent_flight_df = pd.DataFrame()
percent_flight_df['Cause'] = ['Extreme Weather','Security','Late-arriving
Aircraft','Natl. Aviation System','Air Carrier']
percent_flight_df['Percent flight delays'] = percent_delay_flight
percent_flight_df = percent_flight_df.sort_values('Percent flight delays',
ascending=False).reset_index(drop=True)
plt.figure(figsize=(12,8))
plt.stackplot(x_axis,
              monthly_percentage_df['WEATHER_DELAY'],
              monthly_percentage_df['SECURITY_DELAY'],
              monthly_percentage_df['LATE_AIRCRAFT_DELAY'],
              monthly_percentage_df['AIR_SYSTEM_DELAY'],
              monthly_percentage_df['AIRLINE_DELAY'],
              colors=['#FF8C00','#00BFFF','#0000CD','#FFD700','#228B22'],
              labels=cause)
plt.xticks(x_axis, months, rotation='vertical')
plt.title('Cause of Delay per Month\n(SFO 2018)', size=20)
plt.ylabel('% of Delay Minutes', size=18)

legend = plt.legend(bbox_to_anchor=(1.28,1), loc="upper right", fontsize=12)
legend.set_title('Cause of Delay', prop={'size':14})

plt.margins(0)
#plt.savefig('images/monthly_delay_cause.png', bbox_extra_artists=(legend,),
bbox_inches='tight')
plt.show()
```

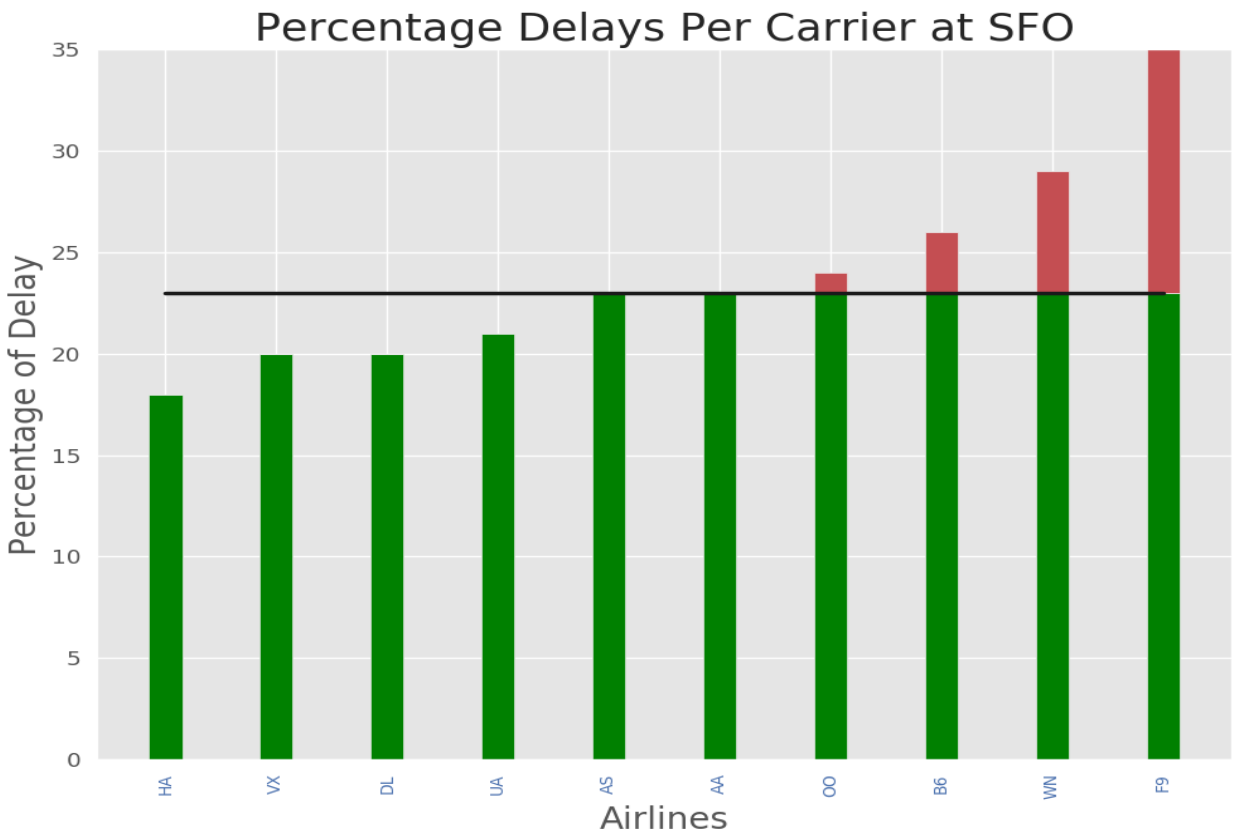

Delayed minutes per cause
(SFO 2018)
% of Delay Minutes
0
5
10
15
20
25
30
35
40
Late-arriving Aircraft
Natl. Aviation System
Air Carrier
Extreme Weather
Security
Delayed flights per cause
(SFO 2018)
% of Delayed Flights
0
5
10
15
20
25
30
35
40
Extreme Weather
Security
Late-arriving Aircraft
Natl. Aviation System
Air Carrier



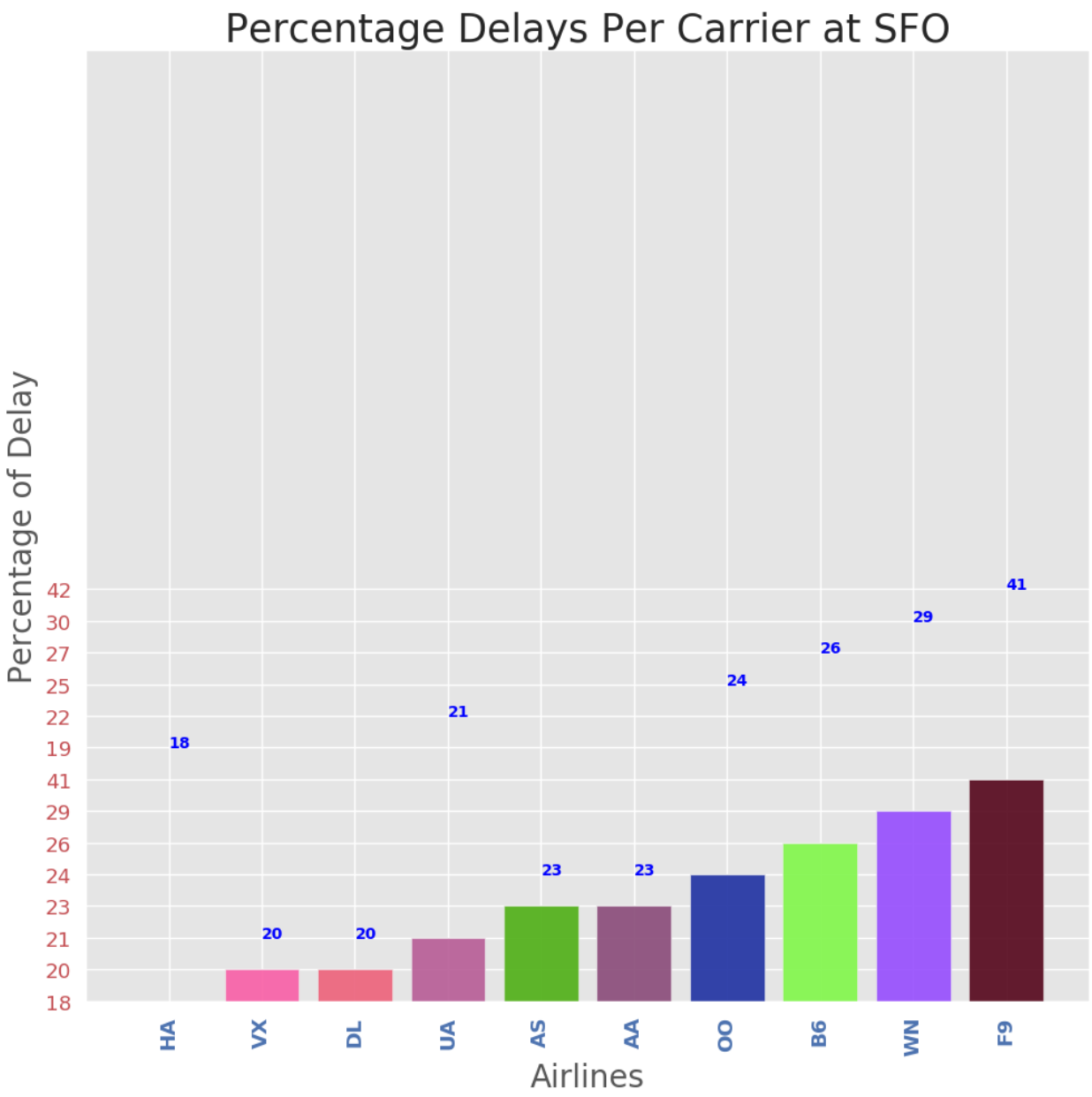

Total number of flights vs total inbound and outbound delays per each carrier. United Airlines has more number of flights and delays, Skywest airlines comes next in the list. Total number of flights and delays are as shown below.

| | AIRLINE | Departure_Delayed_Flights | Total_Flights | Arrival_Delayed_Flights |
|---|---|---|---|---|
| 0 | AA | 2678.0 | 14597 | 2836.0 |
| 1 | AS | 4739.0 | 23503 | 6282.0 |
| 2 | B6 | 1562.0 | 6314 | 1730.0 |
| 3 | DL | 2876.0 | 15322 | 2675.0 |
| 4 | F9 | 443.0 | 1123 | 406.0 |
| 5 | HA | 114.0 | 802 | 175.0 |
| 6 | OO | 8087.0 | 39437 | 8737.0 |
| 7 | UA | 12798.0 | 65081 | 14265.0 |
| 8 | WN | 4750.0 | 18077 | 4523.0 |

Total % of delays for all the airlines with SFO origin airport. Frontier airlines has the highest % of delay followed by Southwest airlines. The mean delay for all the airlines is about 23%.



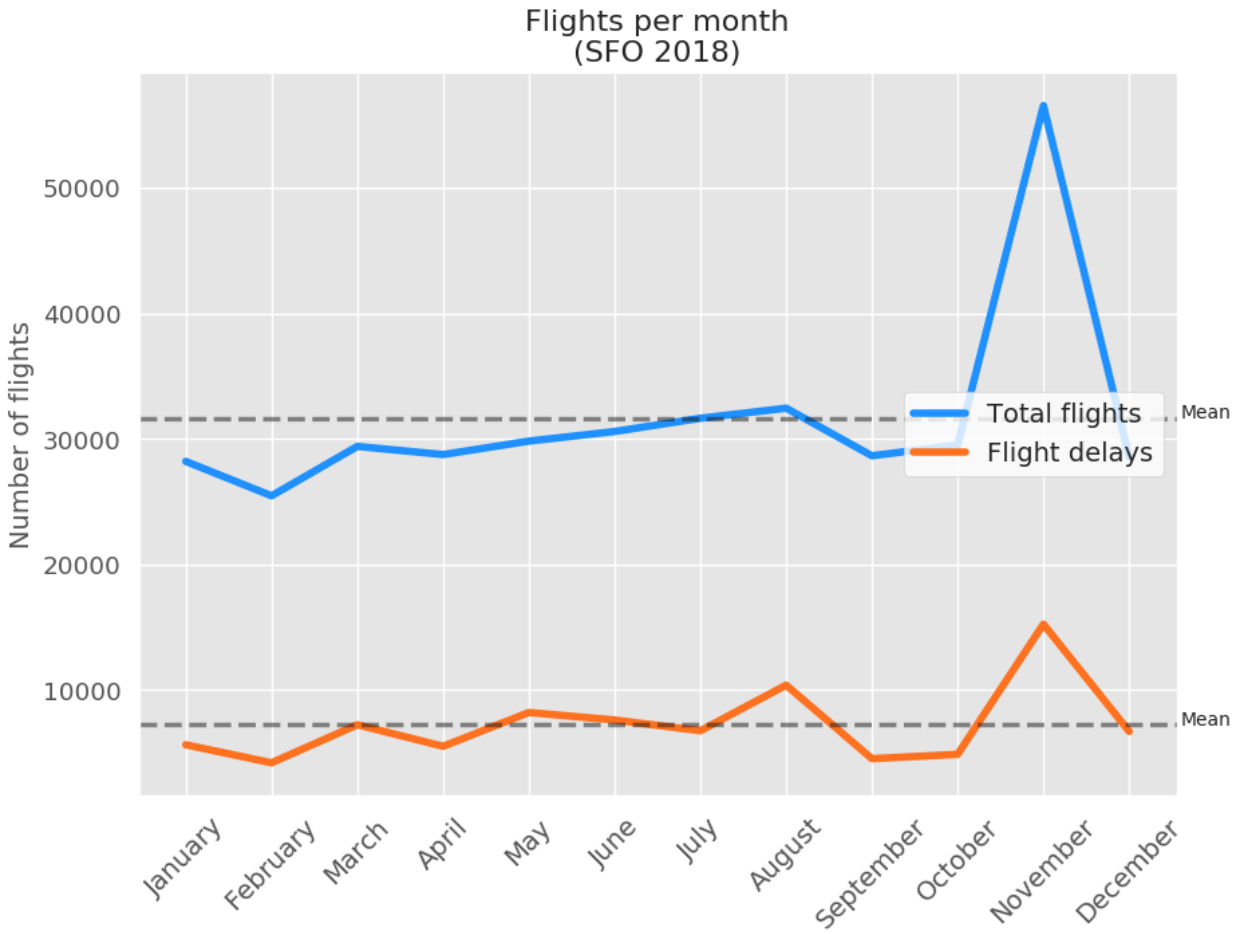

| | Airline | Total Flights | Total Delays | % Delays |
|---|---|---|---|---|
| **4** | HA | 1605 | 292 | 18 |
| **6** | VX | 11505 | 2317 | 20 |
| **9** | DL | 30651 | 6209 | 20 |
| **0** | UA | 130302 | 27013 | 21 |
| **1** | AS | 47022 | 10785 | 23 |
| **8** | AA | 29193 | 6778 | 23 |
| **5** | OO | 78968 | 18914 | 24 |
| **2** | B6 | 12648 | 3242 | 26 |
| **7** | WN | 36156 | 10450 | 29 |
| **3** | F9 | 2246 | 914 | 41 |



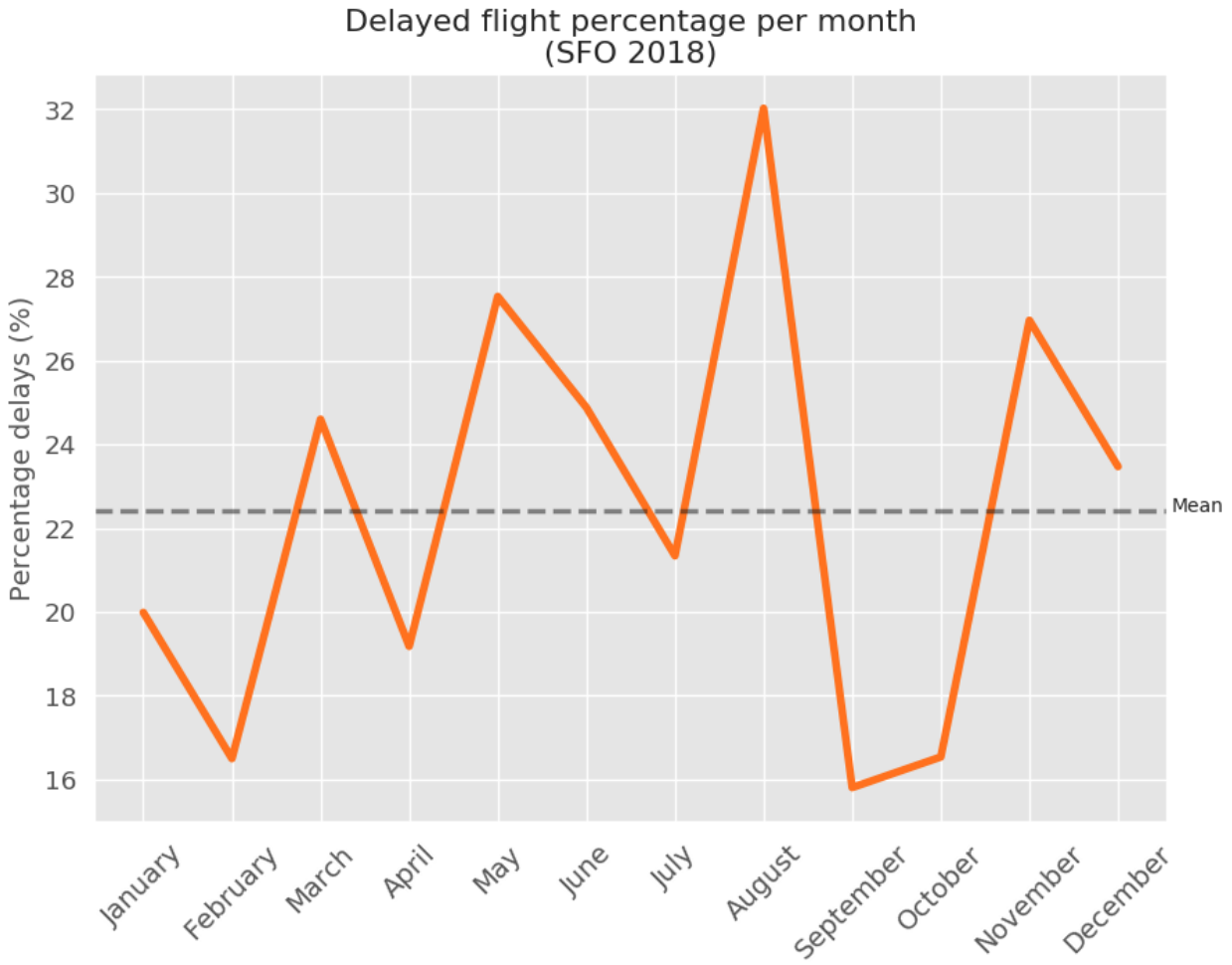

## Calculating inbound and outbound delays >=15 minutes

```
#Outbound Flight Delays
outbound_delays = outbound.loc[outbound['DEPARTURE_DELAY_15'] == 1]
total_outbound_delays = len(outbound_delays['FLIGHT_NUMBER'])

monthly_outbound = outbound_delays.groupby(['MONTH']).sum()
m_outbound_sum = pd.DataFrame(monthly_outbound['DEPARTURE_DELAY_15'])

#Inbound Flight Delays
inbound_delays = inbound.loc[inbound['ARRIVAL_DELAY_15'] == 1]
```

```
total_inbound_delays = len(inbound_delays['FLIGHT_NUMBER'])

monthly_inbound = inbound_delays.groupby(['MONTH']).sum()
m_inbound_sum = pd.DataFrame(monthly_inbound['ARRIVAL_DELAY_15'])
```




The Graph shows larger number of flights and delays in the month of November due to the weather delay caused by storm in the mid-west during the thanksgiving week.




The graph shows a spike in % of delays for the year 2018. As observed there is a spike % delays during the month of July and August because of the summer vacation and during the month February and September the %delay is low as it's a low travel season. The mean % delay is nearly 22%.

# Future Predictions: Departure Delays from SFO origin airport.

Predicting for DAY, DAY_OF_WEEK, MONTH, ORIGIN_AIRPORT_LABEL, DESTINATION_AIRPORT_LABEL, AIRLINE_LABEL for departure delays >=15 minutes

## Prediction Train model: Gaussian Naïve Bayes

```
cnt = 0
nocnt = 0

print('Label: Prediction Value, Probablility, DAY, DAY_OF_WEEK, MONTH,
ORIGIN_AIRPORT_LABEL, DESTINATION_AIRPORT_LABEL, AIRLINE_LABEL')
print('Eg: Predicted Match from SFO on DAY 15 of Month for 10000 X_test set')
for i in range(1,10000):
  sfotestdf = X_test[i:i+1].copy()
  sfotestdf.reset_index(drop=True,inplace=True)

  y_pred = clf.predict(sfotestdf)
  predicedProb = clf.predict_proba(sfotestdf)
  sustr = str(sfotestdf['DAY'][0]) + " " + str(sfotestdf['DAY_OF_WEEK'][0])
+ " " + \
  str(sfotestdf['MONTH'][0])   + " "+
str(sfotestdf['ORIGIN_AIRPORT_LABEL'][0])  + " " +
str(sfotestdf['DESTINATION_AIRPORT_LABEL'][0])  \
  + " " + str(sfotestdf['AIRLINE_LABEL'][0])

  if bool(y_pred) is True and sfotestdf['DAY'][0] == 15:
    if cnt < 5: print(bool(y_pred), predicedProb, sustr)
    cnt = cnt + 1
  else:
    #if nocnt < 5: print(bool(y_pred), predicedProb, sustr)
    nocnt = nocnt + 1

print('Total Number of matches for departure delay > 15 mins = ', cnt)
print('So for the True values, the particular set has predicted delay > 15
mins.')
print('Total Number of matches for departure delay < 15 mins = ', nocnt)
```

## Prediction Result

```
Label: Prediction Value, Probablility, DAY, DAY_OF_WEEK, MONTH, ORIGIN_AIRPORT_LABEL, DESTINATION_AIRPORT_LABEL, AIRLINE_LABEL
Eg: Predicted Match from SFO on DAY 15 of Month for 10000 X_test set
True [[1.19066381e-07 9.99999881e-01]] 15 2 5 0 71 1
True [[0. 1.]] 15 3 8 0 69 7
True [[0. 1.]] 15 4 11 0 19 7
True [[0. 1.]] 15 3 8 0 33 7
True [[5.37002407e-118 1.00000000e+000]] 15 7 7 0 36 3
Total Number of matches for departure delay > 15 mins =  87
So for the True values, the particular set has predicted delay > 15 mins.
Total Number of matches for departure delay < 15 mins =  9912
```

**Predicted Match from SFO on DAY 15 for 10000 X_test set**

| Prediction Value | Probability | DAY | DAY_OF_WEEK | MONTH | ORIGIN_AIRPORT_LABEL | DESTINATION_AIRPORT_LABEL | AIRLINE_LABEL |
|---|---|---|---|---|---|---|---|
| True | [[1.19066381e-07 9.99999881e-01]] | 15 | 2 | 5 | 0 | 71 | 1 |
| True | [[0. 1.]] | 15 | 3 | 8 | 0 | 69 | 7 |
| True | [[0. 1.]] | 15 | 4 | 11 | 0 | 19 | 7 |
| True | [[0. 1.]] | 15 | 3 | 8 | 0 | 33 | 7 |
| True | [[5.37002407e-118 1.00000000e+000]] | 15 | 7 | 7 | 0 | 36 | 3 |

**Total Number of matches for departure delay > 15 mins =  87**
So for the True values, the particular set has predicted delay > 15 mins.
**Total Number of matches for departure delay < 15 mins =  9912**

We have trained a Gaussian Naïve Bayes model to predict flight departure delays from SFO origin airport by considering xtest values ranging from 1-10000, with Departure_Delay_15 as the Ytrain and precting the result for day = 15 as the input for Ypred. The result shows the predicted match of number of departure delays from SFO on day 15, day of the week, month, destination airport and airlines. For True values, the particular set has a delay > 15 mins.

**True count**: **Total Number of matches for departure delay > 15 mins =  87**

**False count**: **Total Number of matches for departure delay < 15 mins =  9912**

Elaborating the predicted result table above.

In the first predicted output, we can predict that on the 15$^{th}$ day of month May ,on Tuesday, there will be a departure delay >=15 of from SFO origin airport to CMX destination airport with United airlines as carrier.

In the second predicted output, we can predict that on the 15th day of month August, on Wednesday, there will be a departure delay >=15 of from SFO origin airport to CMH destination airport with Skywest airlines as carrier.

In the third predicted output, we can predict that on the 15th day of month May ,on Thursday, there will be a departure delay >=15 of from SFO origin airport to ANC destination airport with Skywest airlines as carrier.

From the predicted outputs we can come to the conclusion that most of the flight delays are found in United airlines and Skywest airlines from SFO Origin airport. The predicted results are in favor of our actual analysis from the datasets, plots and algorithms.

# Conclusion

Based on the results, we can conclude that the flight delays can be introduced due to various factors like weather, air traffic, airport location and activities, but the major impact is of the chained delay arising due to the late arriving flights which is propagated to the airport network. The flight delay is also impacted to a greater extent by the originating airport and the airline carrier. Also, datasets related to the airport staff and functionalities can be introduced into the prediction model. Apart from this, various other machine learning and deep learning algorithms are available to be applied for prediction and a comparison can be done to determine the best model depending on the accuracy of the prediction. Airline delay prediction is done for the year 2018 specifically for San Francisco International Airport. The Departure delays are predicted based on the Machine Learning Algorithms – Logistic Regression, Naïve Bayes, Decision Tree, Random Forest, KNN. We chose Gaussian Naïve Bayes model learned for the complete data set for the year 2018. For any day, airline, month, for the Departure delay for the flights from SFO can be predicted based on the algorithm. The comparison of the Accuracy scores, Precision, Recall, and other Sklearn Metrics for the prediction are tabulated below.

| | algorithm | accuracy | precision | recall | fscore | r2score | rocscore | mae | mse | rmse |
|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Logistic Regression | 99.998 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.000 | 0.004 |
| 1 | Naive Bayes | 94.825 | 0.910 | 0.828 | 0.867 | 0.681 | 0.904 | 0.052 | 0.052 | 0.227 |
| 2 | Decision Tree | 100.000 | 1.000 | 1.000 | 1.000 | 1.000 | 1.000 | 0.000 | 0.000 | 0.000 |
| 3 | Random Forest | 95.425 | 0.997 | 0.778 | 0.874 | 0.718 | 0.889 | 0.046 | 0.046 | 0.214 |
| 4 | KNN | 92.444 | 0.947 | 0.667 | 0.783 | 0.535 | 0.829 | 0.076 | 0.076 | 0.275 |

# References


1. Feiteira I. (2018). Predictive Modelling: Flight Delays and Associated Factors, Hartsfield–Jackson Atlanta International Airport. Procedia Computer Science,138,638-645.
2. Chen, Jun & Li, Meng. (2019). Chained Predictions of Flight Delay Using Machine Learning. 10.2514/6.2019-1661.
3. N. Pyrgiotis, K. M. Malone, and A. Odoni, "Modelling delay propagation within an airport network," Transportation Research Part C: Emerging Technologies, vol. 27, pp. 60–75, 2013.
4. K. Gopalakrishnan and H. Balakrishnan, "A comparative analysis of models for predicting delays in air traffic networks," in USA/Europe Air Traffic Management Seminar , 2017
5. Sun Choi (Aerosp. Syst. Design Lab., Georgia Inst. of Technol., Atlanta, GA, United States); Young Jin Kim; Briceno, S.; Mavris, D. Source: 2016 IEEE/AIAA 35th Digital Avionics Systems Conference (DASC). Proceedings, p 6 pp., 2016
6. Thiagarajan, B. (Sri Venkateswara Coll. of Eng., Chennai, India); Srinivasan, L.; Sharma, A.V.; Sreekanthan, D.; Vijayaraghavan, V. Source: 2017 IEEE/AIAA 36th Digital Avionics Systems Conference (DASC). Proceedings, p 6 pp., 2017
7. Proenca, Hugo M. (LIACS, Leiden University, Leiden, Netherlands); Klijn, Ruben; Bäck, Thomas; Van Leeuwen, Matthijs Source: Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018, p 60-67, July 2, 2018, Proceedings of the 2018 IEEE Symposium Series on Computational Intelligence, SSCI 2018
8. Kim, Young Jin (Aerospace Systems Design Laboratory, Georgia Institute of Technology, Atlanta; GA; 30332-0150, United States); Choi, Sun; Briceno, Simon; Mavris, Dimitri Source: AIAA/IEEE Digital Avionics Systems Conference - Proceedings, v 2016-December, December 7, 2016, 35th DASC Digital Avionics Systems Conference 2016, DASC 2016 – Proceedings
9. Chakrabarty, Navoneel, et al. "Flight Arrival Delay Prediction Using Gradient Boosting Classifier." Emerging Technologies in Data Mining and Information Security. Springer, Singapore, 2019. 651-659.
10. Suvojit Manna, Sanket Biswas, Riyanka Kundu, Somnath Rakshit, Priti Gupta, Subhas Burman "A statistical approach to predict flight delay using gradient boosted decision tree", International Conference on Computational Intelligence in Data Science(ICCIDS), 2017
11. Juan Jose Robollo and Hamsa Balakrishnan "Characterization and Prediction of Air Traffic Delays"
12. Yi Ding "Predicting flight delay based on multiple linear regression", IOP Conference Series: Earth and Environmental Science.

13. Sruti Oza, Somya Sharma, Hetal Sangoi, Rutuja Raut, V.C. Kotak ”Flight Delay Prediction System Using Weighted Multiple Linear Regression”, International Journal Of Engineering And Computer Science ISSN:2319-7242, Volume 4 Issue 4 April 2015, Page No. 11668-11677
14. Anish M. Kalliguddi and Aera K. Leboulluec ”Predictive Modeling of Aircraft Flight Delay”, Universal Journal of Management 5(10): 485-491, 2017, DOI: 10.13189/ujm.2017.051003
15. Jianmo Ni, Xinyuan Wang, Ziliang Li ”Flight Delay Prediction using Temporal and Geographical Information”,https://cseweb.ucsd.edu/classes/wi17/ cse258-a/reports/a032.pdf
16. Dong, Yanjie, and Xuehua Wang. ”A new over-sampling approach: random-SMOTE for learning from imbalanced data sets.” International Conference on Knowledge Science, Engineering and Management. Springer, Berlin, Heidelberg, 2011.
17. Li, Jia, Hui Li, and Jun-Ling Yu. ”Application of random-SMOTE on imbalanced data mining.” Business Intelligence and Financial Engineering (BIFE), 2011 Fourth International Conference on. IEEE, 2011.
18. Sina Khanmohammadi, Salih Tutun, Yunus Kucuk ”A New Multilevel Input Layer Artificial Neural Network for Predicting Flight Delays at JFK Airport”, doi.org/10.1016/j.procs.2016.09.321
19. Ball, M., Barnhart, C., Dresner, M., Hansen, M., Neels, K., Odoni, A., Peterson, E., Sherry, L., Trani, A. A., and Zou,B., “Total delay impact study: a comprehensive assessment of the costs and impacts of flight delay in the United States,NEXTOR,” 2010.