exploration_flights_dataset

August 6, 2021

1 Flights Dataset Exploration

1.1 by Tokhir

1.2 Preliminary Wrangling

This dataset reports flights in the United States, including carriers, arrival and departure delays, and reasons for delays, 2008. This dataset consist of slightly more 7 million rows and 29 columns.

In [1]:
```
# import all packages and set plots to be embedded inline
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sb

%matplotlib inline
```

In [2]:
```
df = pd.read_csv('2008.csv.bz2')
```

In [3]:
```
df.shape
```

Out[3]:
```
(7009728, 29)
```

In [4]:
```
# data view
df.head()
```

Out[4]:
```
   Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime  ArrTime  \
0  2008      1           3          4   2003.0        1955   2211.0
1  2008      1           3          4    754.0         735   1002.0
2  2008      1           3          4    628.0         620    804.0
3  2008      1           3          4    926.0         930   1054.0
4  2008      1           3          4   1829.0        1755   1959.0

   CRSArrTime UniqueCarrier  FlightNum       ...        TaxiIn  TaxiOut  \
0        2225            WN        335       ...           4.0      8.0
1        1000            WN       3231       ...           5.0     10.0
2         750            WN        448       ...           3.0     17.0
```

```
3        1100              WN       1746          ...             3.0      7.0
4        1925              WN       3920          ...             3.0     10.0

   Cancelled  CancellationCode  Diverted  CarrierDelay WeatherDelay NASDelay  \
0          0               NaN         0           NaN          NaN      NaN
1          0               NaN         0           NaN          NaN      NaN
2          0               NaN         0           NaN          NaN      NaN
3          0               NaN         0           NaN          NaN      NaN
4          0               NaN         0           2.0          0.0      0.0

   SecurityDelay  LateAircraftDelay
0            NaN                NaN
1            NaN                NaN
2            NaN                NaN
3            NaN                NaN
4            0.0               32.0

[5 rows x 29 columns]
```

In [5]:
```
# describtive statistic
df.describe()
```

Out[5]:
```
            Year         Month    DayofMonth     DayOfWeek       DepTime  \
count  7009728.0  7.009728e+06  7.009728e+06  7.009728e+06  6.873482e+06
mean      2008.0  6.375130e+00  1.572801e+01  3.924182e+00  1.333830e+03
std          0.0  3.406737e+00  8.797068e+00  1.988259e+00  4.780689e+02
min       2008.0  1.000000e+00  1.000000e+00  1.000000e+00  1.000000e+00
25%       2008.0  3.000000e+00  8.000000e+00  2.000000e+00  9.280000e+02
50%       2008.0  6.000000e+00  1.600000e+01  4.000000e+00  1.325000e+03
75%       2008.0  9.000000e+00  2.300000e+01  6.000000e+00  1.728000e+03
max       2008.0  1.200000e+01  3.100000e+01  7.000000e+00  2.400000e+03

         CRSDepTime       ArrTime    CRSArrTime     FlightNum  \
count  7.009728e+06  6.858079e+06  7.009728e+06  7.009728e+06
mean   1.326086e+03  1.481258e+03  1.494801e+03  2.224200e+03
std    4.642509e+02  5.052251e+02  4.826728e+02  1.961716e+03
min    0.000000e+00  1.000000e+00  0.000000e+00  1.000000e+00
25%    9.250000e+02  1.107000e+03  1.115000e+03  6.220000e+02
50%    1.320000e+03  1.512000e+03  1.517000e+03  1.571000e+03
75%    1.715000e+03  1.909000e+03  1.907000e+03  3.518000e+03
max    2.359000e+03  2.400000e+03  2.400000e+03  9.743000e+03

       ActualElapsedTime        ...             Distance        TaxiIn  \
count       6.855029e+06        ...         7.009728e+06  6.858079e+06
mean        1.273224e+02        ...         7.263870e+02  6.860852e+00
std         7.018731e+01        ...         5.621018e+02  4.933649e+00
min         1.200000e+01        ...         1.100000e+01  0.000000e+00
25%         7.700000e+01        ...         3.250000e+02  4.000000e+00
```

```
50%         1.100000e+02       ...          5.810000e+02  6.000000e+00
75%         1.570000e+02       ...          9.540000e+02  8.000000e+00
max         1.379000e+03       ...          4.962000e+03  3.080000e+02

            TaxiOut     Cancelled      Diverted  CarrierDelay  WeatherDelay  \
count  6.872670e+06  7.009728e+06  7.009728e+06  1.524735e+06  1.524735e+06
mean   1.645305e+01  1.960618e-02  2.463006e-03  1.577206e+01  3.039031e+00
std    1.133280e+01  1.386426e-01  4.956753e-02  4.009912e+01  1.950287e+01
min    0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
25%    1.000000e+01  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
50%    1.400000e+01  0.000000e+00  0.000000e+00  0.000000e+00  0.000000e+00
75%    1.900000e+01  0.000000e+00  0.000000e+00  1.600000e+01  0.000000e+00
max    4.290000e+02  1.000000e+00  1.000000e+00  2.436000e+03  1.352000e+03

           NASDelay  SecurityDelay  LateAircraftDelay
count  1.524735e+06   1.524735e+06       1.524735e+06
mean   1.716462e+01   7.497434e-02       2.077098e+01
std    3.189495e+01   1.837940e+00       3.925964e+01
min    0.000000e+00   0.000000e+00       0.000000e+00
25%    0.000000e+00   0.000000e+00       0.000000e+00
50%    6.000000e+00   0.000000e+00       0.000000e+00
75%    2.100000e+01   0.000000e+00       2.600000e+01
max    1.357000e+03   3.920000e+02       1.316000e+03

[8 rows x 24 columns]
```

In [6]:
```python
# dtypes of variables
df.dtypes
```

Out[6]:
```
Year                   int64
Month                  int64
DayofMonth             int64
DayOfWeek              int64
DepTime              float64
CRSDepTime             int64
ArrTime              float64
CRSArrTime             int64
UniqueCarrier         object
FlightNum              int64
TailNum               object
ActualElapsedTime    float64
CRSElapsedTime       float64
AirTime              float64
ArrDelay             float64
DepDelay             float64
Origin                object
Dest                  object
Distance               int64
```

```
TaxiIn                 float64
TaxiOut                float64
Cancelled                int64
CancellationCode        object
Diverted                 int64
CarrierDelay           float64
WeatherDelay           float64
NASDelay               float64
SecurityDelay          float64
LateAircraftDelay      float64
dtype: object
```

1.3 Data Wrangling

- Define

Drop unnecessary columns

- Code

```
In [7]: df.drop(['UniqueCarrier', 'TaxiIn', 'TaxiOut', 'FlightNum', 'CRSArrTime', 'CarrierDelay'
```

- Test

```
In [8]: df.head()
```

```
Out[8]:    Year  Month  DayofMonth  DayOfWeek  DepTime  CRSDepTime  ArrTime TailNum  \
        0  2008      1           3          4   2003.0        1955   2211.0  N712SW
        1  2008      1           3          4    754.0         735   1002.0  N772SW
        2  2008      1           3          4    628.0         620    804.0  N428WN
        3  2008      1           3          4    926.0         930   1054.0  N612SW
        4  2008      1           3          4   1829.0        1755   1959.0  N464WN

           ActualElapsedTime  CRSElapsedTime  AirTime  DepDelay Origin Dest  Distance  \
        0              128.0           150.0    116.0       8.0    IAD  TPA       810
        1              128.0           145.0    113.0      19.0    IAD  TPA       810
        2               96.0            90.0     76.0       8.0    IND  BWI       515
        3               88.0            90.0     78.0      -4.0    IND  BWI       515
        4               90.0            90.0     77.0      34.0    IND  BWI       515

           Cancelled CancellationCode  Diverted
        0          0              NaN         0
        1          0              NaN         0
        2          0              NaN         0
        3          0              NaN         0
        4          0              NaN         0
```

- Define

Using replace function replace the numbers used to indicate the month and days of the week with verbal designation.

- Code

```
In [9]: df['Month'].replace([1,2,3,4,5,6,7,8,9,10,11,12], ['January', 'February', 'March', 'Apri
        df['DayOfWeek'].replace([1,2,3,4,5,6,7], ['Monday', 'Tuesday', 'Wednesday', 'Thursday',
```

- Test

```
In [10]: df['Month'].value_counts()

Out[10]: July         627931
         March        616090
         August       612279
         June         608665
         May          606293
         January      605765
         April        598126
         February     569236
         October      556205
         December     544958
         September    540908
         November     523272
         Name: Month, dtype: int64

In [11]: df['DayOfWeek'].value_counts()

Out[11]: Wednesday    1039665
         Monday       1036201
         Friday       1035166
         Thursday     1032224
         Tuesday      1032049
         Sunday        976887
         Saturday      857536
         Name: DayOfWeek, dtype: int64
```

1.3.1 What is the structure of your dataset?

This dataset represents information about approximately seven billion flights inside the country(US). The main part of the variables is integer and float numbers but more than half of them is information about datetime in integer and float format.

1.3.2 What is/are the main feature(s) of interest in your dataset?

The main features in the data are the distributions of months, day of weeks, day of months, distance, and the main reasons for cancellation.

1.3.3 What features in the dataset do you think will help support your investigation into your feature(s) of interest?

The variables related to date, distance, reasons for cancellation might be to help my investigations.

1.4 Univariate Exploration

In the first part of the analysis, I will look at visualizations with only one variable(univariate exploration). This will be the beginning of the analysis to move on to more complex visualizations. This exploration is meant to see how individual variables behave. I want to get answers to the following questions:

How many flights are there every month, every day of the week, every day of the month?

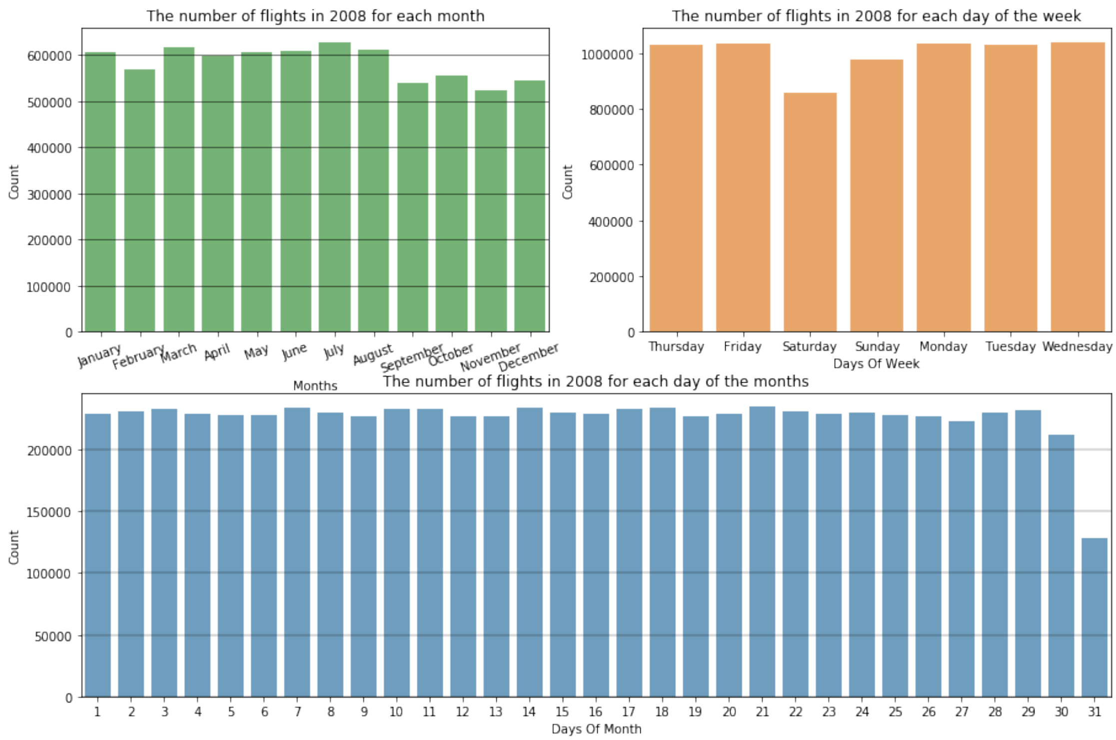
Most frequent flight distance

Most common reasons for canceled flights

2 I

```
In [12]: # creating a function to avoid repetition during visualizations.
         def countplot_func(df, x_axis, color_type):
             return (sb.countplot(data = df, x= x_axis, color = color[color_type], alpha = 0.7))
```

```
In [13]: # plotting
         plt.figure(figsize=(15,10))
         color = sb.color_palette()

         plt.subplot(2,2,1)
         countplot_func(df, 'Month', 2);
         plt.grid(axis='y', linewidth = 0.5, color= 'black')
         plt.xticks(rotation = 20)
         plt.xlabel('Months')
         plt.ylabel('Count')
         plt.title('The number of flights in 2008 for each month')

         plt.subplot(2,2,2)
         countplot_func(df, 'DayOfWeek', 1);
         plt.xlabel('Days Of Week')
         plt.ylabel('Count')
         plt.title('The number of flights in 2008 for each day of the week')

         plt.subplot(2,1,2)
         countplot_func(df, 'DayofMonth', 0);
         plt.grid(axis='y', linewidth = 0.3, color= 'black');
         plt.xlabel('Days Of Month');
         plt.ylabel('Count')
         plt.title('The number of flights in 2008 for each day of the months');
```



I created a bar charts for months, day of weeks and day of months, since the first two are a categorical variables.

These charts are uniformly distributed apart from some drops in each graph. he last graph shows that on every day of any month, approximately the same number of flights takes place, not counting the 30th and 31st. This is due to the alternation of 30 and 31 numbers and also due to leap and common years. As for the days of the week and months, the smallest number of flights takes place on weekends and on last four months of the year.

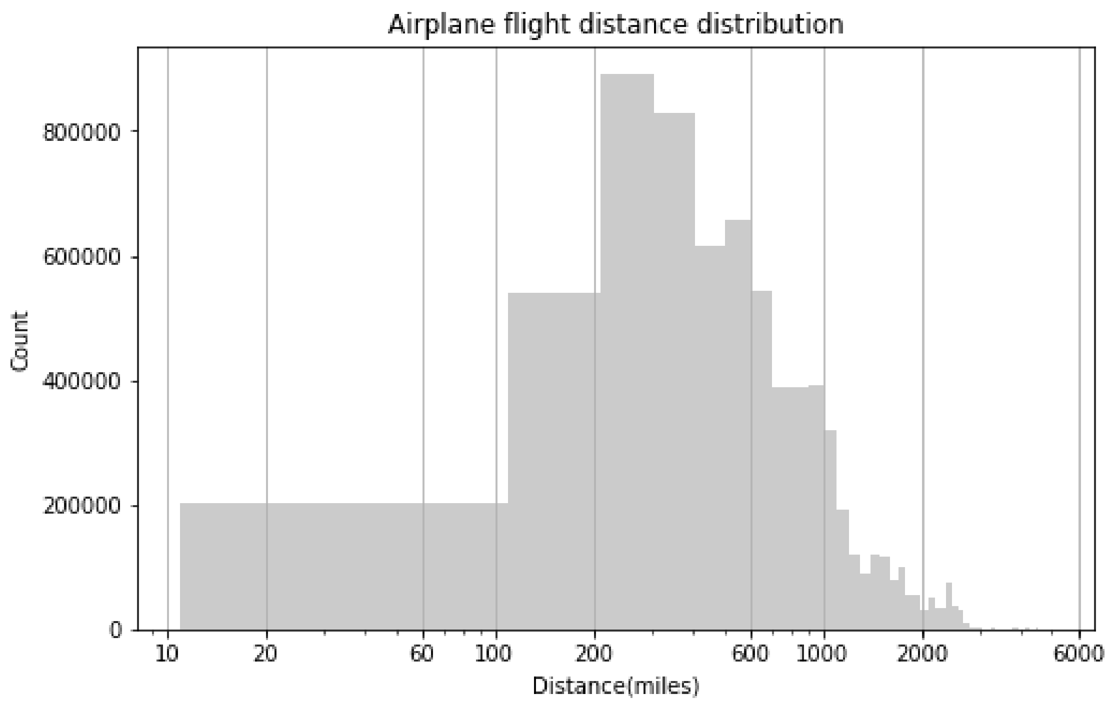
---

3 II

In [14]:
```
# plotting
def flight_distance_distr():
    plt.figure(figsize=(8,5))
    color = sb.color_palette()

    sb.distplot(df['Distance'], color=color[7], kde=False);
    plt.xscale('log')
    ticks = [10, 20, 60, 100, 200, 600, 1000, 2000, 6000]
    plt.xticks(ticks, ticks)
    plt.grid(axis='x');
    plt.title('Airplane flight distance distribution')
```

```
        plt.xlabel('Distance(miles)')
        plt.ylabel('Count');
```

```
In [15]: flight_distance_distr();
```



I created a histogram for distance, since it is a numeric variable. My initial plots show that distance follows a highly right-skewed distribution, because of this I used a log scaling. Under a log scale, i see that the data is roughtly unimodal, with one large peak somewhere between 200 and 300.

---

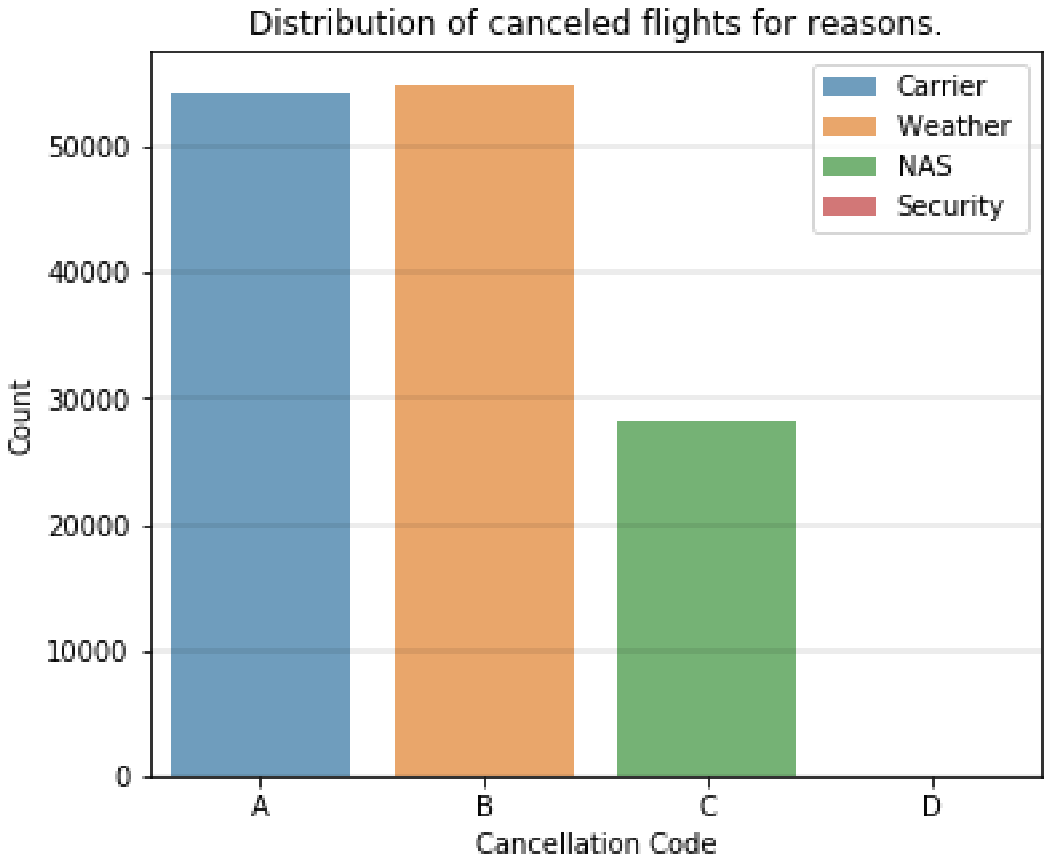
4 III

```
In [16]: # setting data order
         cancellation_code = ['A','B','C','D']
         order = pd.api.types.CategoricalDtype(categories=cancellation_code, ordered=True)
         df['CancellationCode'] = df['CancellationCode'].astype(order);
```

```
In [17]: # plotting
         def canceled_flights_distr():
             plt.figure(figsize=(6,5))
             ax = sb.countplot(data=df, x= 'CancellationCode', hue = 'CancellationCode', alpha =
             plt.grid(axis='y', linewidth = 0.15, color = 'black');
             plt.ylabel('Count')
             plt.xlabel('Cancellation Code');
```

```
    plt.title('Distribution of canceled flights for reasons.');
    ax.legend(['Carrier','Weather ','NAS', 'Security']);
```

```
In [18]: canceled_flights_distr();
```



Since these features are categorical, i produced bar chart here. In addition, since the columns are not ordinal, i sorted them alphabetically.

The bar chart show that the main reason for cancellations is weather. After a little lag, cancellations due to carrier follows. It may seem that the last variable(security) is missing on the graph, but due to its very small value, it is not visible. Below I have provided statistics on the number of cancellations. It can be seen that the difference is huge.

```
In [19]: df.CancellationCode.value_counts()

Out[19]: B    54904
         A    54330
         C    28188
         D       12
         Name: CancellationCode, dtype: int64
```

4.0.1 Discuss the distribution(s) of your variable(s) of interest. Were there any unusual points? Did you need to perform any transformations?

After creating visualizations, i approximately got the results that I expected. In the second graph, when I was considering at the distance distribution, i was using the log function because the tail of the histogram was too long. On the last chart when i consedered at the number of types of cancellation reasons i expected much more cancellations due to security. After calculations i got that the probability of flight cancellation due to security is 0.003%.

4.0.2 Of the features you investigated, were there any unusual distributions? Did you perform any operations on the data to tidy, adjust, or change the form of the data? If so, why did you do this?

I have deleted almost half of the columns in this dataset that I will not be using in this analysis. Also I figured that I would not change the data type for the date to make it easier for me to compose visualizations in the future.

hanged the numeric data of months and days of the week to verbal data.

4.1 Bivariate Exploration

In the second part of the analysis, I will look at visualizations with two variables(bivariate exploration). Here we will look at the relationship between two variables depending on their type. I asked the following questions:

At which airports the most frequent flight cancellations occur?

Is there a relationship between the departure time of the aircraft and the distance?

how canceled flights were distributed by months?

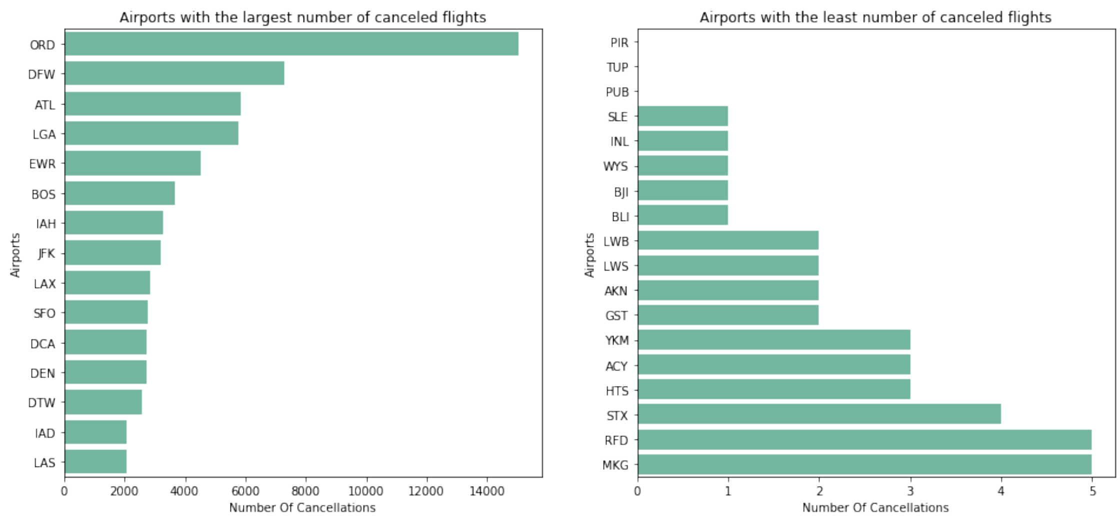
5 IV

```
In [20]: most_cancell = df.groupby('Origin', as_index = False).count()[['Origin', 'CancellationC
         least_cancel = df.groupby('Origin', as_index = False).count()[['Origin', 'CancellationC
```

```
In [21]: # creating a function to avoid repetition during visualizations.
         def barplot_func(df, top_air):
             return sb.barplot(data = df.head(top_air), x = 'CancellationCode', y = 'Origin', co
```

```
In [22]: # plotting
         plt.figure(figsize=(16,7))
         color1 = sb.color_palette("Set2")[0]

         plt.subplot(1,2,1)
         barplot_func(most_cancell, 15);
         plt.title('Airports with the largest number of canceled flights');
         plt.ylabel('Airports')
```

```
plt.xlabel('Number Of Cancellations')


plt.subplot(1,2,2)
barplot_func(least_cancel, 18);
plt.title('Airports with the least number of canceled flights');
plt.ylabel('Airports');
plt.xlabel('Number Of Cancellations');
```

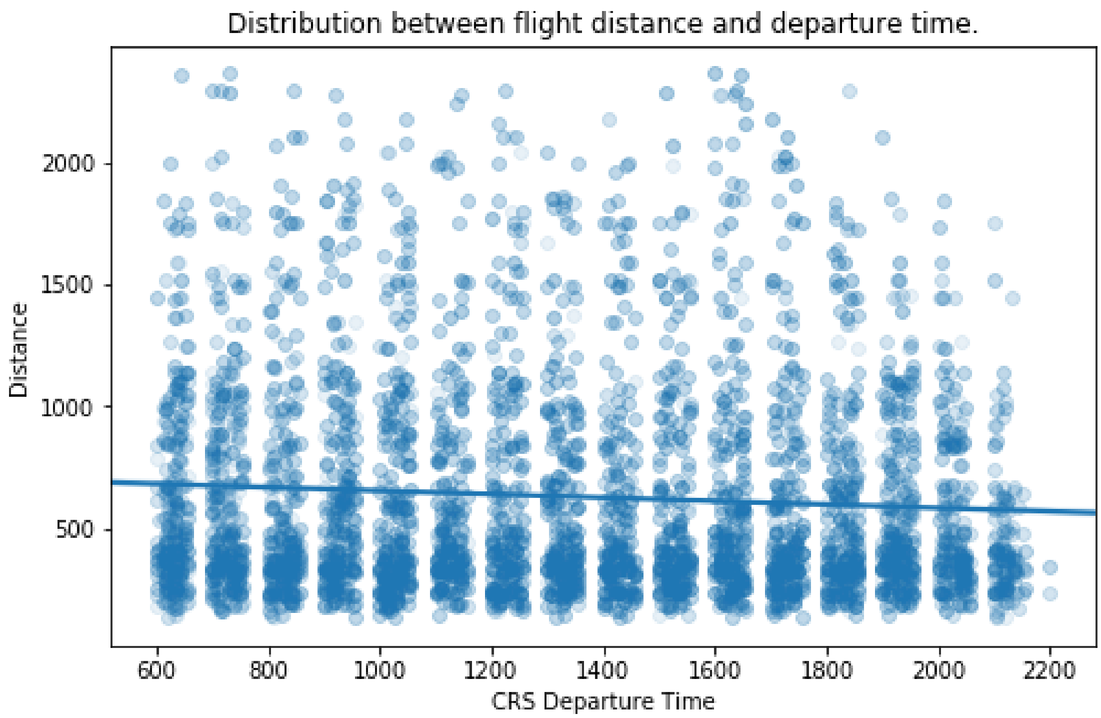


I created a bar chart here, since i have one categorical and one quantitative variable, where each categorical data has a corresponding number.

---

6 V

```
In [23]: # plotting
         def distr_flight_distan_dep_time():
             plt.figure(figsize=(8,5))
             sb.regplot(data=df.head(10000), x= 'CRSDepTime', y = 'Distance', scatter_kws={'alph
             plt.title('Distribution between flight distance and departure time.');
             plt.xlabel('CRS Departure Time');

In [24]: distr_flight_distan_dep_time()
```



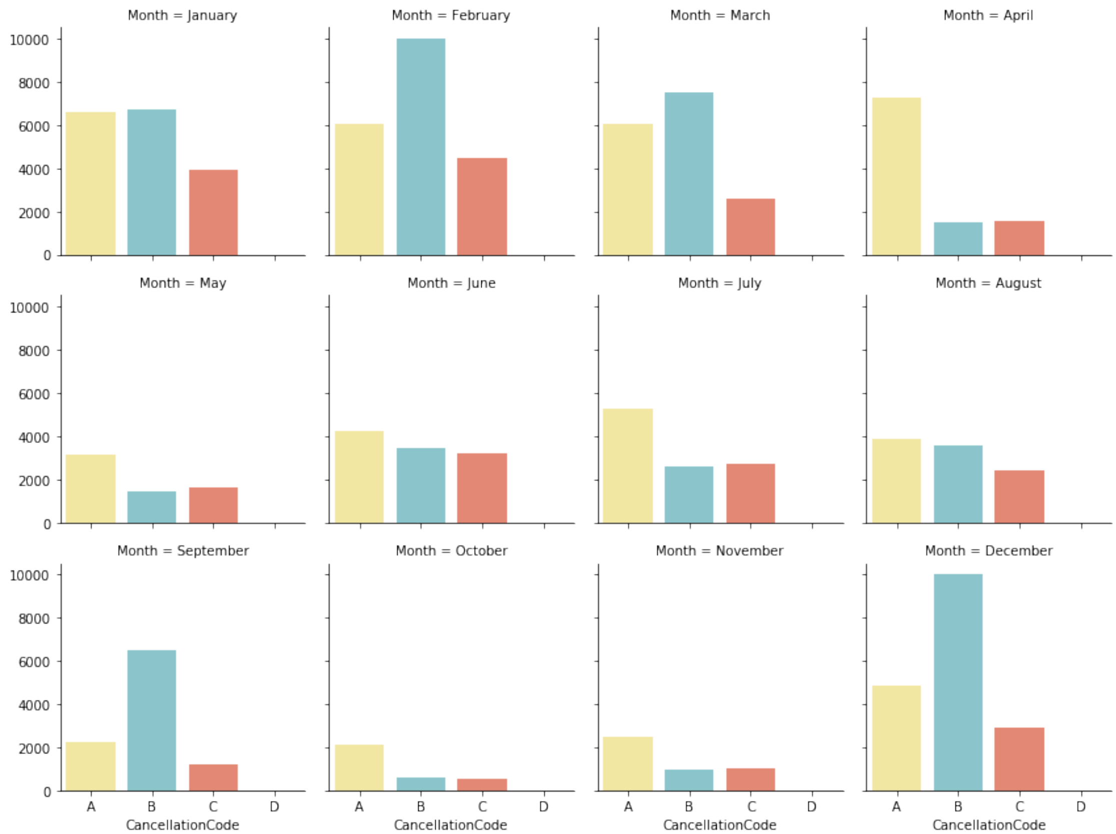
Since these are both numeric variables, i created a scatterplot.

---

7 VI

First method

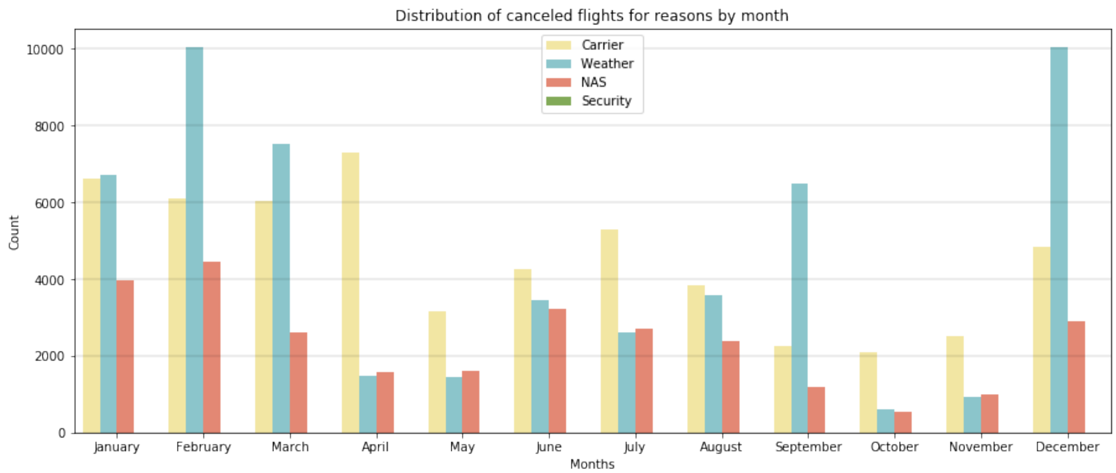
```
In [25]: # plotting
         def distr_canceled_flights_by_month1():
             g = sb.FacetGrid(data=df, col='Month', col_wrap=4)
             g.map(sb.countplot, 'CancellationCode', palette = ['#ffef96', '#80ced6', '#f57c61',

In [26]: distr_canceled_flights_by_month1();

/opt/conda/lib/python3.6/site-packages/seaborn/axisgrid.py:703: UserWarning: Using the countplot
  warnings.warn(warning)
```



Second method

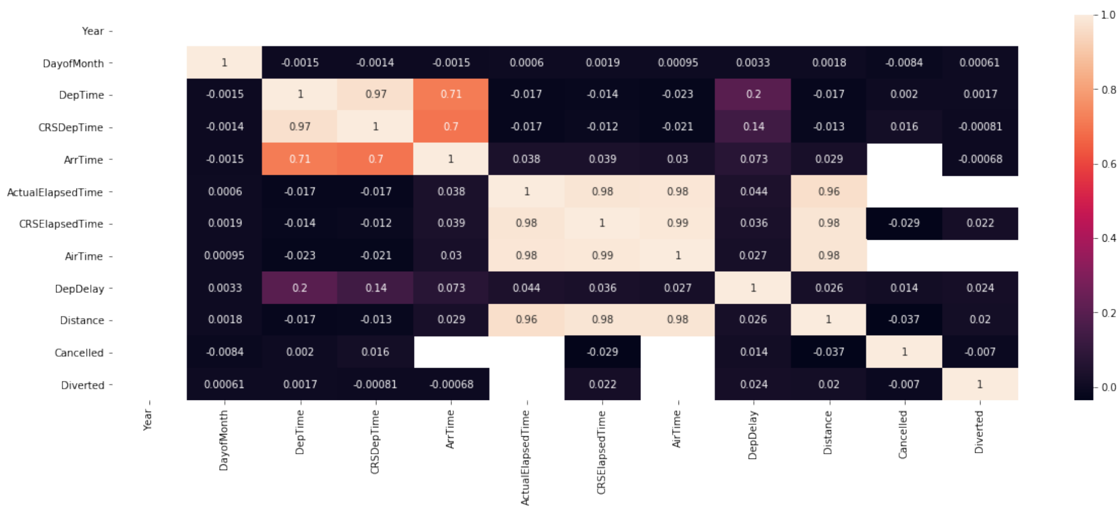
```
In [27]: # plotting
         def distr_canceled_flights_by_month2():
             plt.figure(figsize=(15,6))
             a = sb.color_palette("tab10")

             ax = sb.countplot(data = df, x = 'Month', hue = 'CancellationCode', palette = ['#ff
             plt.grid(axis='y', linewidth = 0.15, color = 'black');
             ax.legend(['Carrier','Weather ','NAS', 'Security'], loc = 'upper center');
             plt.title('Distribution of canceled flights for reasons by month');
             plt.ylabel('Count');
             plt.xlabel('Months');
```

```
In [28]: distr_canceled_flights_by_month2();
```



Since these features are categorical, i produced bar chart here.

In [29]:
```
plt.figure(figsize=(20,7))
sb.heatmap(df.corr(), annot=True);
```



7.0.1 Talk about some of the relationships you observed in this part of the investigation. How did the feature(s) of interest vary with other features in the dataset?

In first graphs i considered airports with higher and lower rates of cancelled flights. Flights were canceled due to carrier(A), weather(B), NAS(C)(https://aspmhelp.faa.gov/index/Types_of_Delay.html - info about NAS) and security(D). By a huge margin in the list of the most frequent canceled flights is the airport Chicago O'Hare International Airport(15000 canceled flights). It is followed by Dallas Fort Worth International Airport(7500) and Hartsfield – Jackson Atlanta International Airport(5800).

In the following graph, I examined the relationship between distance and scheduled departure time. I wanted to know if the departure time(morning, afternoon, evening, night) of the plane depends on the distance. Having received the results, i realized that there is no connection between them and the distance does not depend on what time the plane takes off.

In the last graph I wanted to know the distribution of the flight cancellation because of weather in each month. As expected, the most frequent flight cancellations occur in winter (December and February). Starting from April and until the end of the summer, flight cancellations due to weather are more moderate. Surprisingly, the least number of flight cancellations due to weather was observed in two out of three months of autumn. To make the visualization more informative, i also added the rest of the reasons for the cancellation of flights.

7.0.2 Did you observe any interesting relationships between the other features (not the main feature(s) of interest)?

At the very end, I created a heatmap of the relationships of all variables.

7.1 Multivariate Exploration

In the third part of the analysis, I will look at visualizations with three and more variables(multivariate exploration). In this part there will be more complex visualizations, more informative graphs. This is where encodings come in with the help of color, size and shape. I asked the following questions:

What is the relationship between the departure, arrival of the aircraft and the distance it travels?

What days are the most frequent departure delays in average (in minutes)?

Which days and which flights were most often redirected?

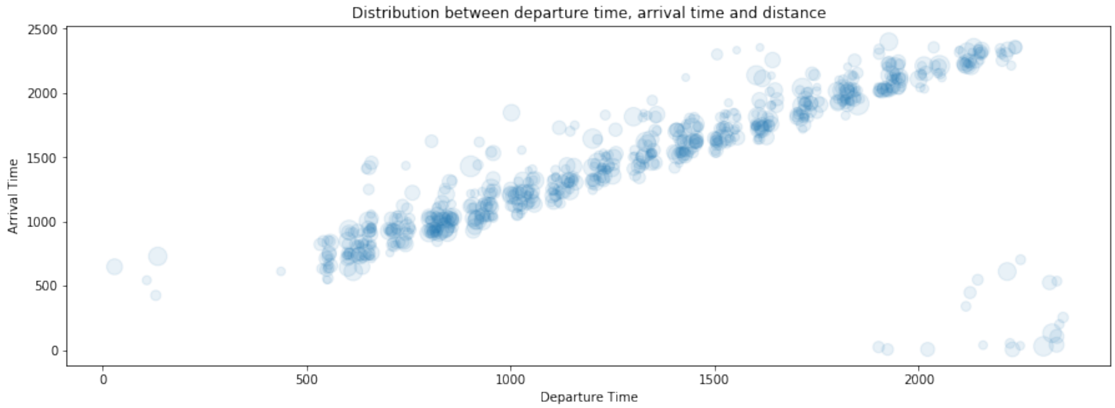
What the heatmap will look like for delayed flights (average in minutes) by months in the top 20 airports by this indicator?

8 VII

In [30]:
```
# plotting
def distr_dep_time_arr_time_distance():
    sample = df.sample(700)
    plt.figure(figsize=(15,5))
    ax = sb.regplot(data=sample, x='DepTime', y= 'ArrTime', marker='o',
                scatter_kws={'s': df['AirTime'], 'alpha':0.1}, fit_reg=False, x_jitter=0
    plt.title('Distribution between departure time, arrival time and distance');
```

```
        plt.ylabel('Arrival Time');
        plt.xlabel('Departure Time');
```

```
In [31]: distr_dep_time_arr_time_distance();
```



Since these are both numeric variables, i choise a scatterplot.

---

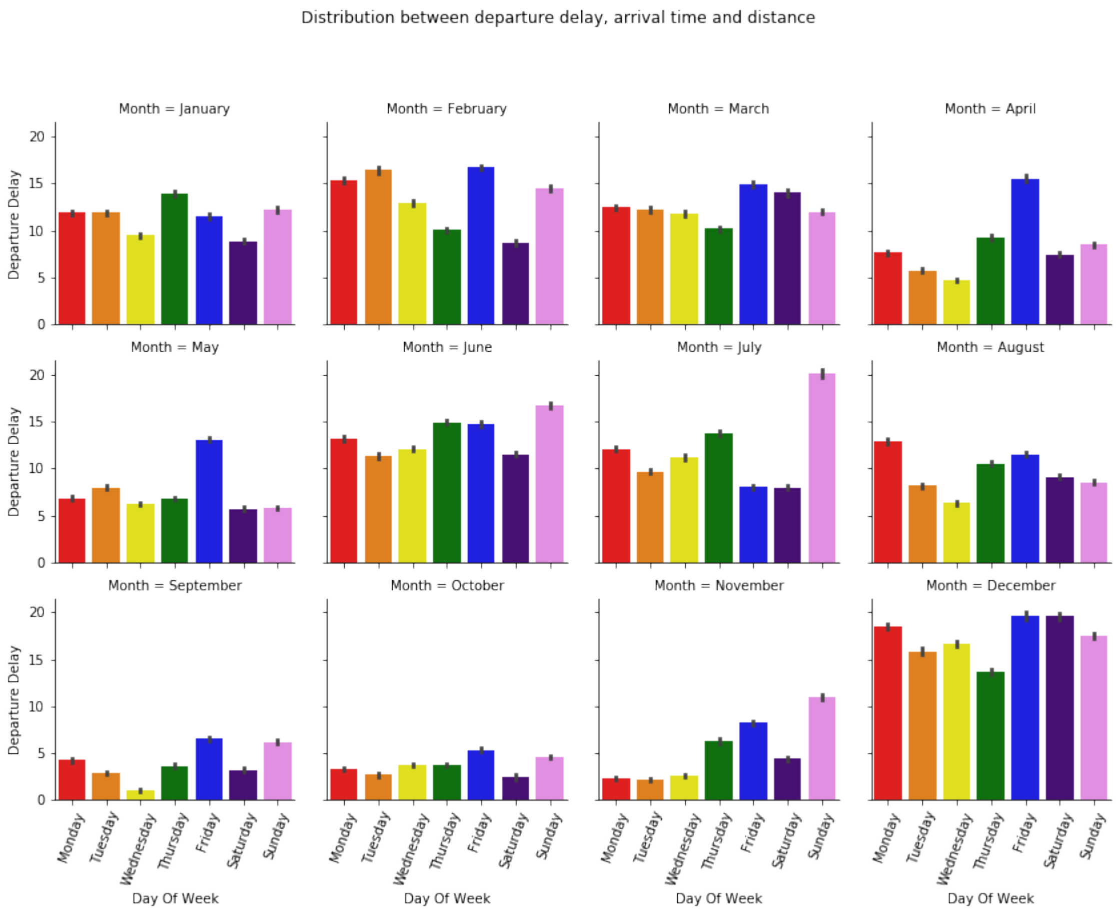
9 VIII

```
In [32]: # setting data order
         d_of_week = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunda
         order = pd.api.types.CategoricalDtype(categories=d_of_week, ordered=True)
         df['DayOfWeek'] = df['DayOfWeek'].astype(order);
```

```
In [33]: # plotting
         def distr_dep_delay_arr_time_distance():
             g = sb.FacetGrid(data=df, col='Month', col_wrap=4)
             g.map(sb.barplot, 'DayOfWeek', 'DepDelay', palette = ['#FF0000','#FF8000','#FFFF00'
             g.set_xticklabels(rotation = 70);
             g.set_xlabels('Day Of Week')
             g.set_ylabels('Departure Delay')
             g.fig.subplots_adjust(top=0.85)
             g.fig.suptitle('Distribution between departure delay, arrival time and distance');
```

```
In [34]: distr_dep_delay_arr_time_distance();
```

```
/opt/conda/lib/python3.6/site-packages/seaborn/axisgrid.py:703: UserWarning: Using the barplot f
  warnings.warn(warning)
```



---

10 IX

In [35]:
```
# concatenation of two columns into one to get flight directions
df['Flight'] = df['Origin']+ '-' + df['Dest']
div_flights = df[df.Diverted == 1]
div_flights_top = div_flights.groupby(['Flight'], as_index = False).count()[['Flight',
div_flights_top = div_flights_top.head(30)
div_flights_top_df = div_flights[div_flights.Flight.isin(div_flights_top.Flight)]
```

In [36]:
```
# plotting
def distr_flights_divertedflights_day_of_week():
    g=sb.FacetGrid(data=div_flights_top_df, col='Flight', col_wrap=5)
    g.map(sb.countplot, 'DayOfWeek')
    g.set_xticklabels(rotation = 70)
    g.fig.subplots_adjust(top=0.85)
    g.fig.suptitle('Distribution between flights, diverted flights and day of week');
```

In [37]: distr_flights_divertedflights_day_of_week();

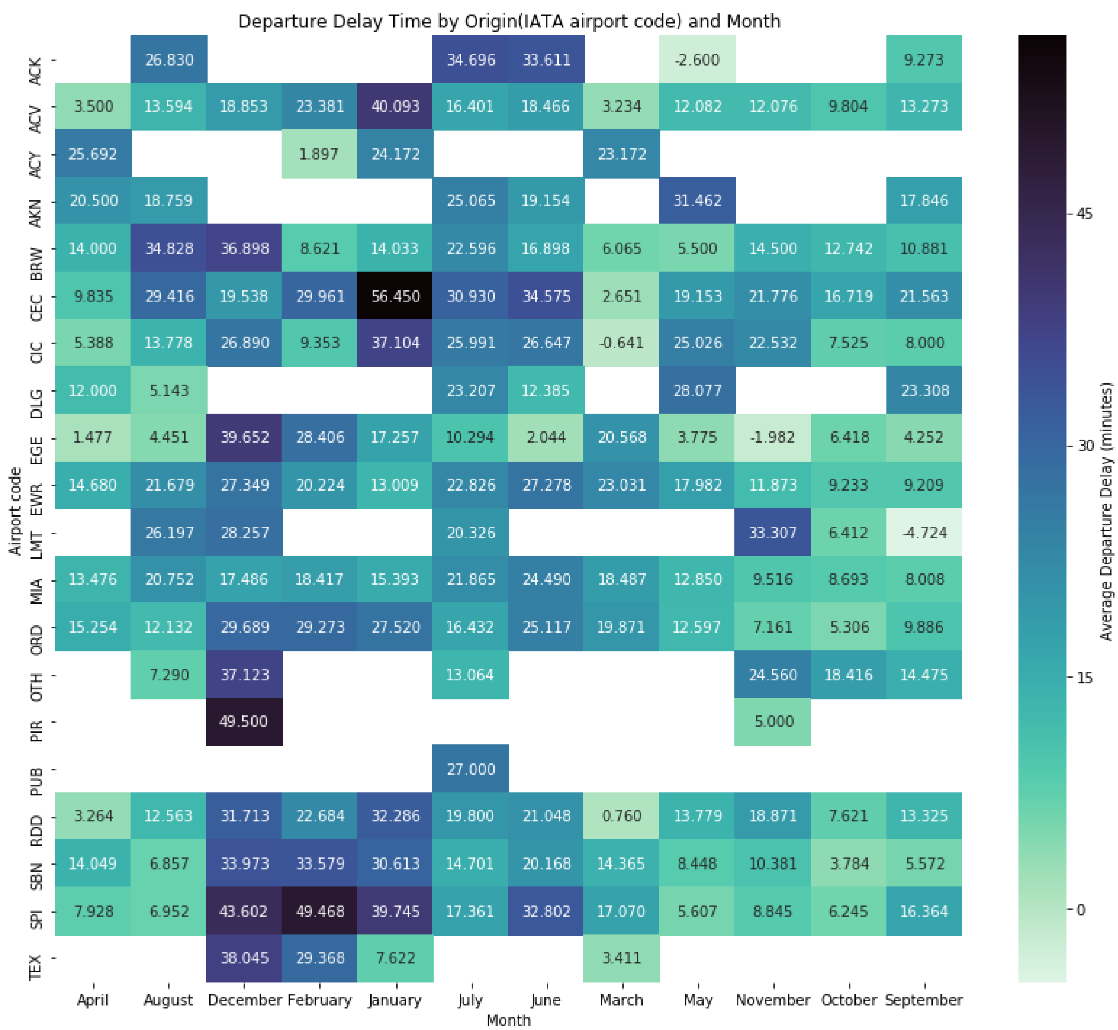
```
/opt/conda/lib/python3.6/site-packages/seaborn/axisgrid.py:703: UserWarning: Using the countplot
  warnings.warn(warning)
```

11 X

```
In [38]: # preparing a dataframe for using a heatmap
         airpor_delay_20 = df.groupby(['Origin']).mean()['DepDelay'].sort_values(ascending=False
         airpor_delay_20_df = df[df.Origin.isin(airpor_delay_20)]
         delay_means = airpor_delay_20_df.groupby(['Origin', 'Month']).mean()['DepDelay']
         delay_means = delay_means.reset_index(name='ArrDelay_avg')
         delay_means_pivot = delay_means.pivot(index='Origin', columns='Month', values='ArrDelay
```

```
In [39]: # plotting
         def dep_delay_by_Origin_and_Month():
             plt.figure(figsize=[14,12])
             sb.heatmap(delay_means_pivot, cmap='mako_r', annot=True, fmt='0.3f', cbar_kws={'lab
             plt.title('Departure Delay Time by Origin(IATA airport code) and Month')
             plt.ylabel('Airport code');
```

```
In [40]: dep_delay_by_Origin_and_Month();
```



11.0.1 Talk about some of the relationships you observed in this part of the investigation. Were there features that strengthened each other in terms of looking at your feature(s) of interest?

In the first visualization, data show an uphill pattern as you move from left to right, this indicates a positive relationship between X and Y. As the X-values increase, the Y-values tend to increase. I looked at the relationship between departure, arrival and distance traveled. The data in the lower right corner may appear to be outliers, but it is not. The reason for this is that the plane took off in the evening and landed the very next day.

In the second graph, i looked at departure delays in minutes for each day of the week throughout the year. As you can see from the visualization, the months with the highest number of departure delays coincide with the months with the highest number of canceled flights. This makes sense because the months with the most canceled flights will roughly be the months with the most departure delays.

In the third visualization i looked at the top 30 redirected flights by destination forwarding by day of the week. The most frequent diverted flight is a flight from ORG to LGA(66). - The most frequent redirected flights on Monday and Sunday are SLC-SUN(15,23), - The most frequent redirected flighte on Tuesday and are Thursday ATL-LGA(17,17), - The most frequent redirected flight on Wednesday is ATL-DFW(16), - The most frequent redirected flight on Friday is DFW-LGA(18), - The most frequent redirected flight on Saturday is LAX-JFK(15)

In the last visualization, i looked at the time of the delay in the departure of the plane(top 20 airports for this indicator) and the airport of departure by months. To accomplish this I used a heatmap. On this heat map, light areas represent less time spent, while dark areas represent more time. You can see that at some airports, delays were only a few months, but due to their magnitude, they hit the top twenty. More than half of the airports experience delays every month on average. Top 3 longest delays seen in Winter.

11.0.2 Were there any interesting or surprising interactions between features?

I was surprised when I saw that in the first days of the week there were very few delays. I thought there would be a fair amount of flight delays at the start of the new working week.