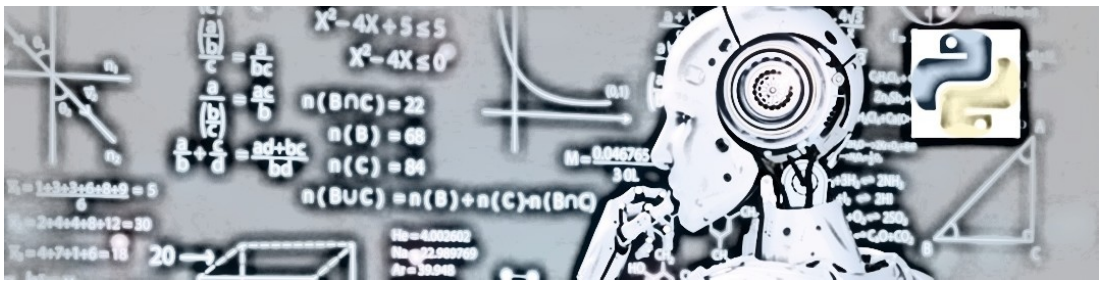

EMERGENCY - CALL 911

EDUARDO MENEZES DE SOUZA AMARANTE

DATA ANALYSIS OF EMERGENCY CALLS IN MONTGOMERY COUNTY, PA

SALVADOR
2021

EMERGENCY – CALL 911
EDUARDO MENEZES DE SOUZA AMARANTE

DATA ANALYSIS OF EMERGENCY CALLS IN MONTGOMERY COUNTY, PA

In this report is presented some details of data exploration, creation of dashboard and conclusions.

SALVADOR
2021

CONTENTS

CONTENTS....3
1. INTRODUCTION....4
2. DATA EXPLORATION....4
2.1 MISSING VALUES TREATMENT....4
2.2 FEATURE ENGINEERING....6
2.3 OUTLIERS TREATMENT....6
3 POWER BI....10
4. CONCLUSION....11

1. INTRODUCTION

The dataset used in this report consist in an open database with data of emergency calls for 911 in Montgomery County, PA. This dataset is available for download in Kaggle. There are three types of emergency calls in this data: fire, traffic and emergency medical service (EMS).
In this report, some details of data analysis will be shared. The principal part of data preparation was made in Python language using the Pandas library and this one was followed by the power query at Power BI.

2. DATA EXPLORATION

The dataset has 663,522 entries distributed in the total of 9 columns, for instance: **lat, lng, desc, zip, title, timeStamp, twp, addr** and **e.** The source of dataset doesn't show details to every column, but the most part of them can be deducted.

lat: latitude of the origin of call;
lng: longitude of the origin of call;
desc: description of calls with address, town, date and time.
zip: zip code of calls area;
title: type of emergency and details of emergency;
twp: town;
addr: address of calls;
e: I didn't understand which it means.

The '**e**' column is filled with the unique value equal 1, so loses the sense of keeping the ones in dataset.

2.1 MISSING VALUES TREATMENT

There are missing values in dataset as can be seen in the picture below.

```
In [5]: df.isnull().sum()

Out[5]: lat              0
        lng              0
        desc             0
        zip          80199
        title            0
        timeStamp        0
        twp            293
        addr             0
        e                0
        dtype: int64
```

Figure 1: Total of missing values by column in dataset.

The total of missing values in **zip column** correspond to 12.08% of dataset. I decided for removing this column due to, in my opinion, the information can be replaced by latitude and

longitude. I decided on keeping the **twp column** without changes in this stage of data exploration because the specific treatment was made in Power Query language.

2.2 FEATURE ENGINEERING

It was necessary make transformations in some columns for preparing this data to make a property visualization. The **timestamp column** contains two important information: date and the time when the person reported the emergency. It is reasonable the split one into two columns: Date and Time columns. Two functions were created to convert their values to datetime format and to extract information of hour. The hour values were placed in a new column.

```
def convert_time(teste):
    novo_time = datetime.datetime.strptime(teste, '%H:%M:%S').time()
    hora = novo_time.hour
    return hora
```

```
df['Hora'] = df['Time'].map(convert_time)
```

```
def convert_data(teste):
    nova_data = datetime.datetime.strptime(teste, '%Y-%m-%d').date()
    nova_data = nova_data.strftime('%m-%d-%Y')
    return nova_data
```

```
df['Data'] = df['Data'].map(convert_data)
```

Figure 2: Function application to convert type of data.

A new column with the information of part of day, AM and PM, was created to be used in dashboard.

```
df['Periodo'] = np.where(  df['Hora'] <12, 'AM', 'PM' )
```

```
df['Periodo'].unique()
```

```
array(['PM', 'AM'], dtype=object)
```

Figure 3: Command used to create the period column.

The type of call and its details is in the column, so I decided to put ones in different columns with the command:

df[['Title', 'Type']] = df['title'].str.split(':', expand=True)

Some columns were renamed while the other ones were dropped of the dataset.

2.3 OUTLIERS TREATMENT

The dataset contains many outliers in longitude and latitude columns. It was made univariate analysis for both columns.

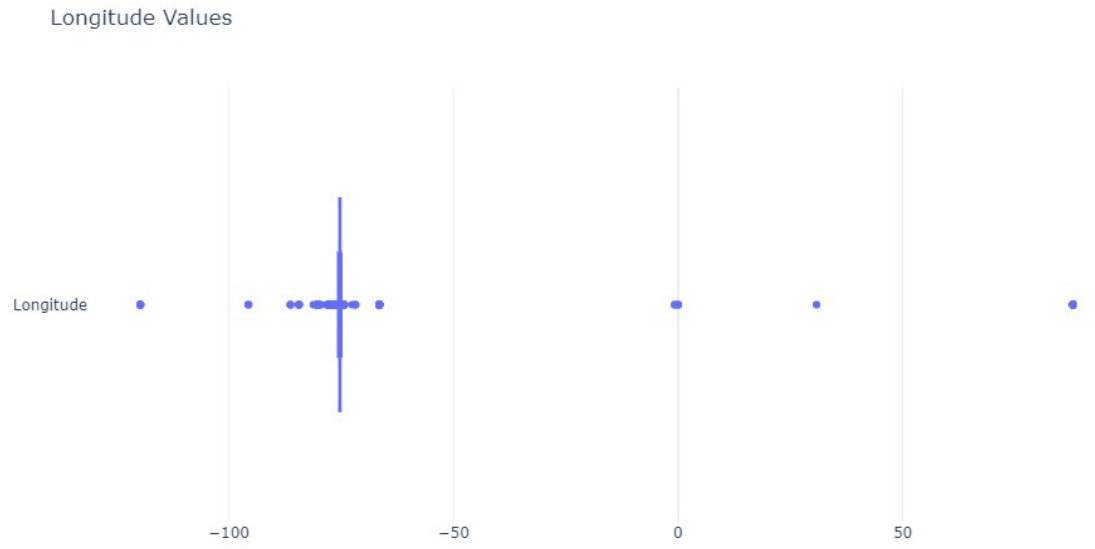

Figure 4: Box plot of longitude values.

As can be seen in the picture above, the values of outliers disrupt the interpretation of values. This behavior can see with latitude values.

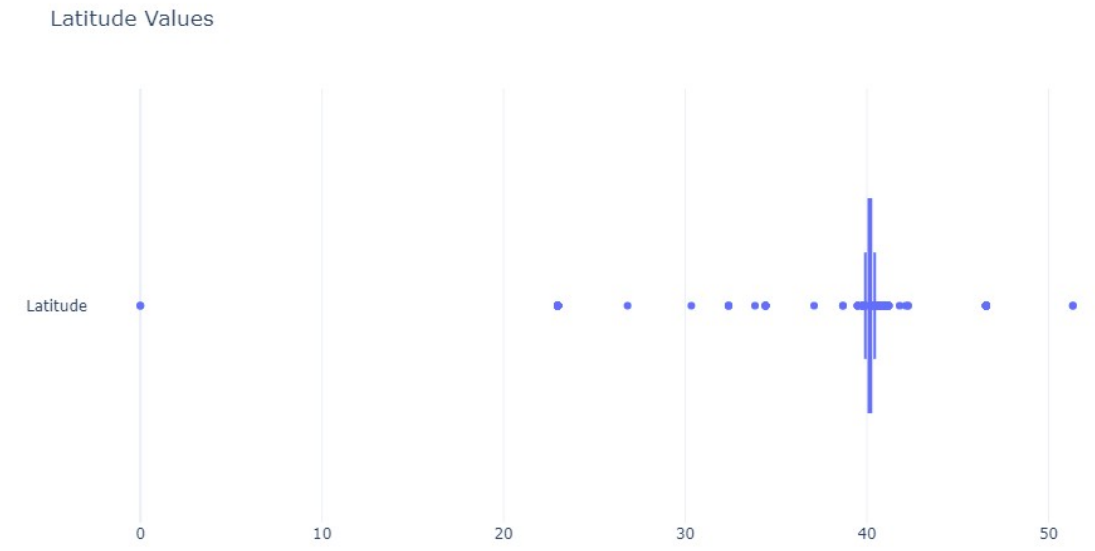

Figure 5: Box plot of latitude values.

When we see these coordinates values and apply them in real situation, we understand that there is something wrong. Every dot in the plot would means the location of someone had a call for 911 number in Montgomery County, PA. However, the presence of outliers regardless the

reason that they are presented in the dataset, they don't make any sense, as we can see the picture below.

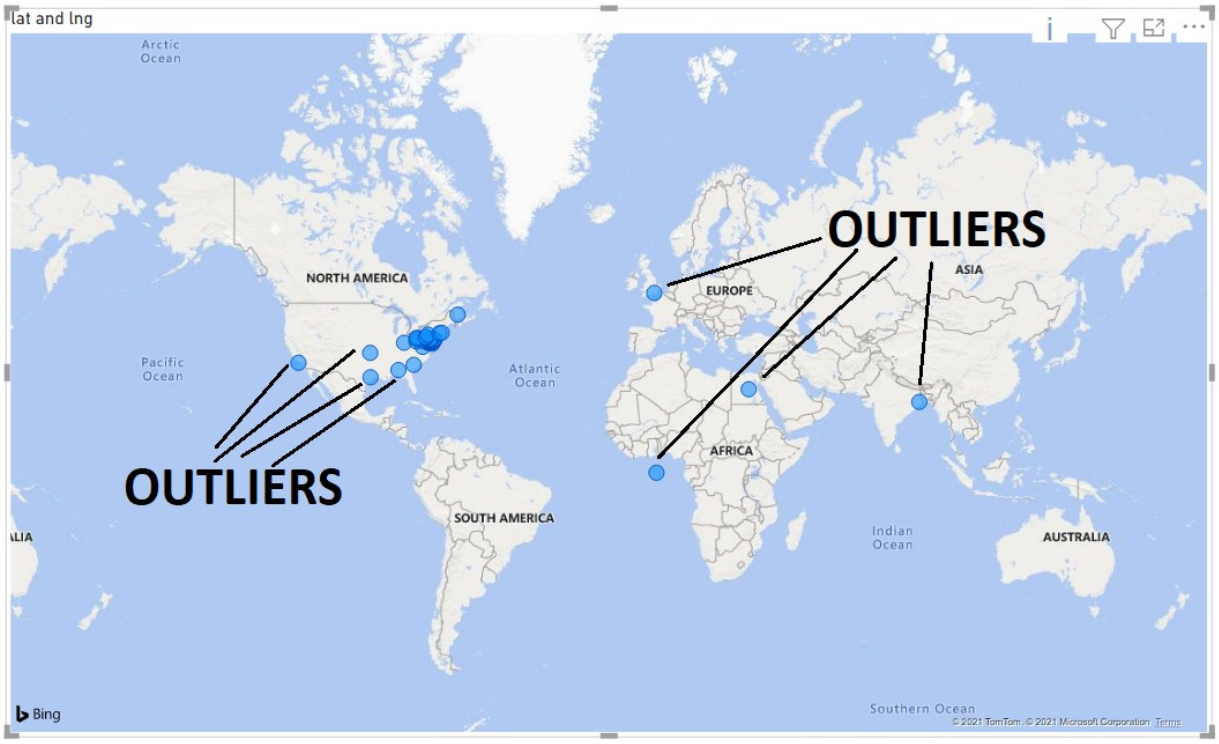

Figure 6: Meaning of outliers in the dataset.

To remove them, I've considered the conditions below.

```
Q1_Lat = df['Lat'].quantile(0.25)
Q3_Lat = df['Lat'].quantile(0.75)

IQR_Lat = Q3_Lat - Q1_Lat

Q1_Lng = df['Lng'].quantile(0.25)
Q3_Lng = df['Lng'].quantile(0.75)

IQR_Lng = Q3_Lng - Q1_Lng

# Defining the cutt off to extrem Outliers

Out_Lat = [Q1_Lat - 3 * IQR_Lat, Q3_Lat + 3 * IQR_Lat]
Out_Lng = [Q1_Lng - 3 * IQR_Lng, Q3_Lng + 3 * IQR_Lng]
```

```
# Dropping them

df.drop(df[df['Lat'] >= Out_Lat[1]].index, inplace=True)
df.drop(df[df['Lat'] <= Out_Lat[0]].index, inplace=True)

df.drop(df[df['Lng'] >= Out_Lng[1]].index, inplace=True)
df.drop(df[df['Lng'] <= Out_Lng[0]].index, inplace=True)
```

Figure 7: Conditions to remove the outliers in dataset.

As expected, the condition kept some outliers in dataset, but it was my decision to keep them because I think some calls made by someone who wasn't inside the limits of town could be redirected to call center available.

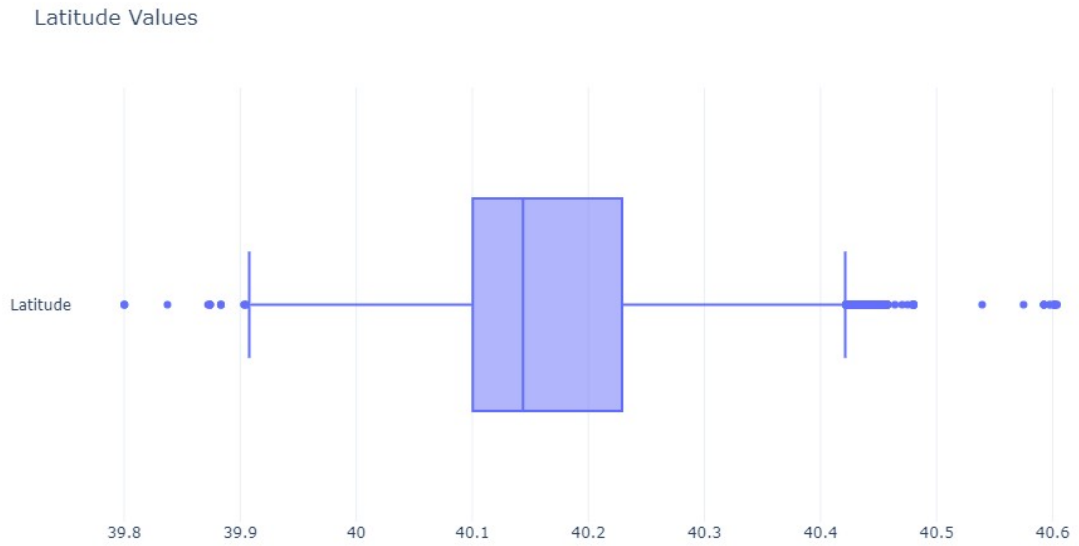

Figure 8: New box plot with the latitudes.

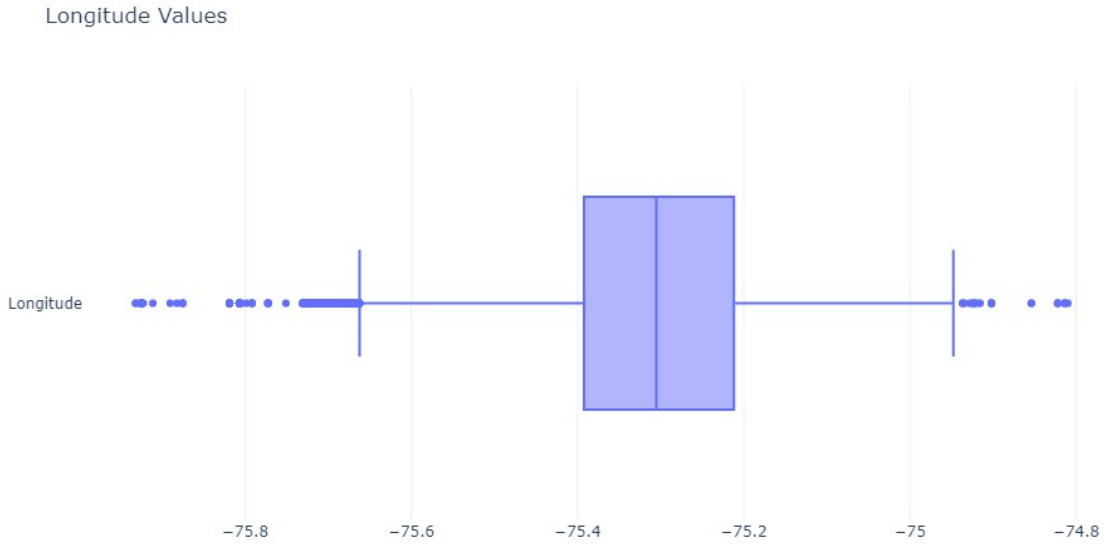

Figure 9: New box plot with the longitudes.

The dataset after the all preparation contains 662,931 entries distributed in 10 columns.

3 POWER BI

In this part of report, I discuss some information at which I extracted after the creation of dashboard. Several information could be extract of dashboard, so I will write about some ones.

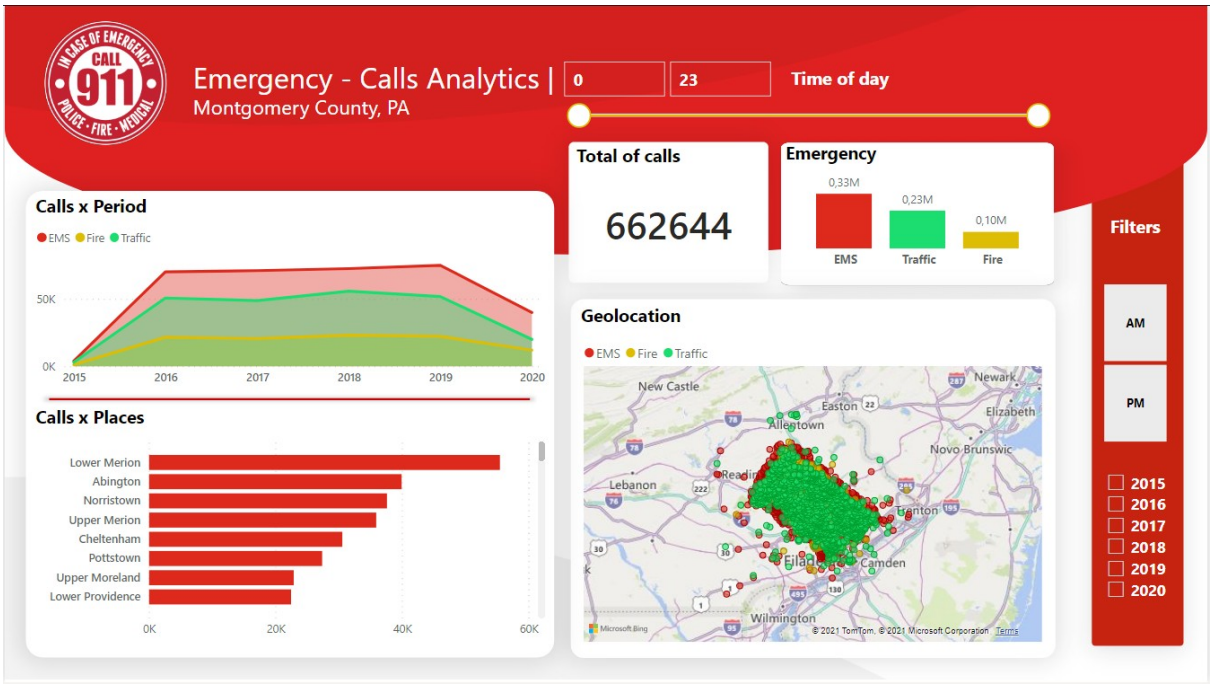

Figure 10: Dashboard of dataset.

The Emergency Medical Service (EMS) is the leader of the calls in the all County from 2015 to 2020. In 2018 there was more calls than other periods.
Almost two thirds of records happened in the PM period in these six years.

I choose only the 2018 to give more details of calls.

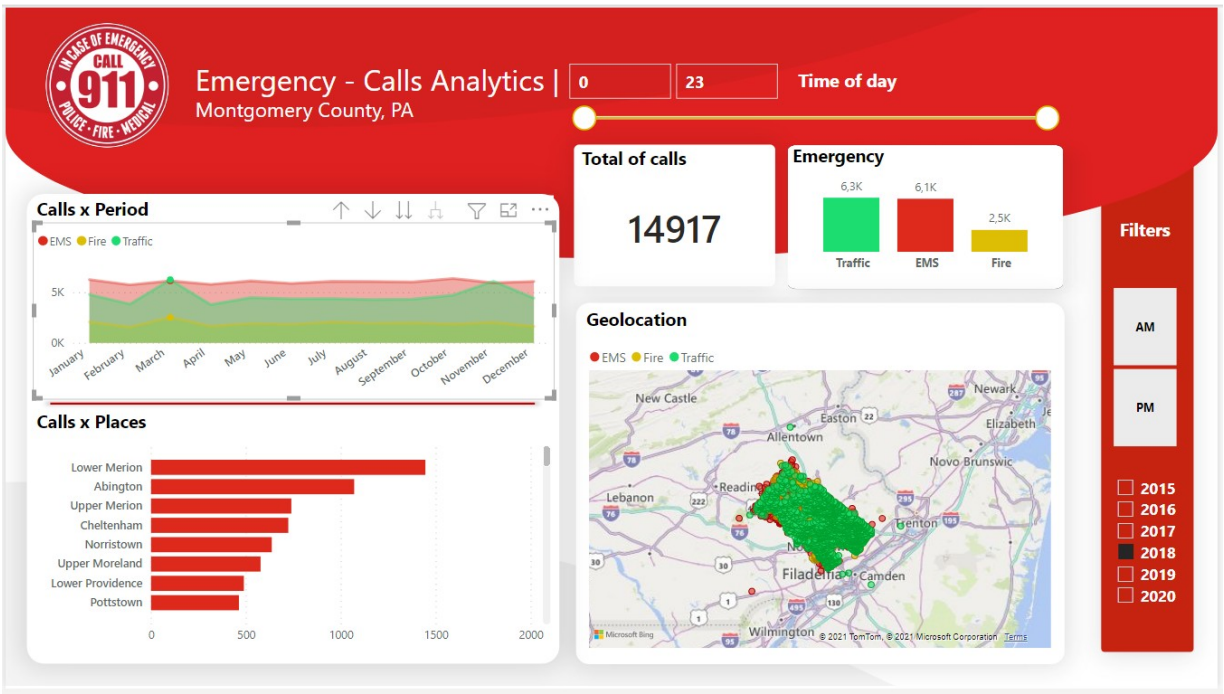

Figure 11: Details of calls in all County in the 2018.

The number of EMS calls were much more in AM period than other kind of call, but in PM the number of occurrences for traffic led the calls. The month with more calls was March.

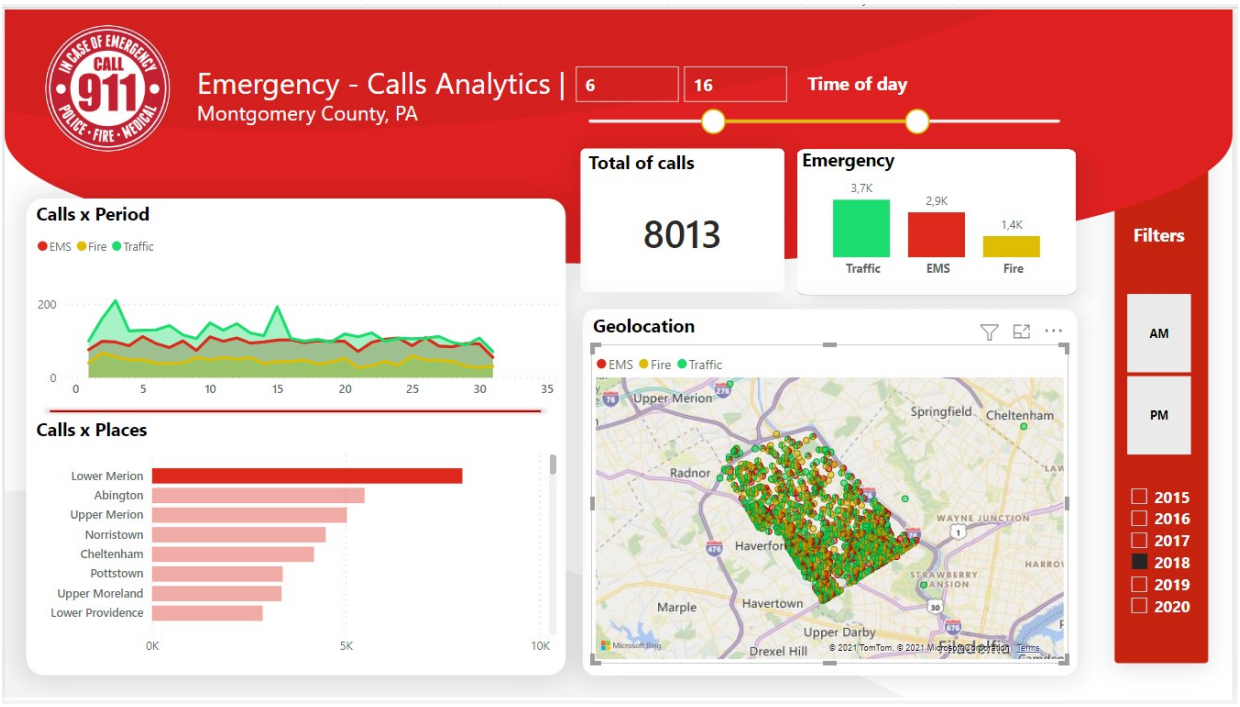

Figure 12: Calls in Lower Merion in the 2018 between 6 AM and 16 PM.

In this video, you can check more details of dashboard. There is no sound in this video.

4. CONCLUSION

This report aims to show some details of data exploration in Python language and some decisions I had to take in the data preparation. The information missing about the data on site didn't damage this report, but it was necessary to search on Google about some aspects of data to get a big picture of data.
The results were presented through the nice dashboard.