Introduction

In this project I have analyzed some 911 call data from Kaggle. The data contains the following fields:

- lat : String variable, Latitude
- lng: String variable, Longitude
- desc: String variable, Description of the Emergency Call
- zip: String variable, Zipcode
- title: String variable, Title
- timeStamp: String variable, YYYY-MM-DD HH:MM:SS
- twp: String variable, Township
- addr: String variable, Address
- e: String variable, Dummy variable (always 1)

First we will have to make some changes in the dataset so we can perform the visualizations.

Creating new features

In the titles column there are "Reasons/Departments" specified before the title code. These are EMS, Fire, and Traffic. By using .apply() with a custom lambda expression I have created a new column called "Reason" that contains this string value.

For example, if the title column value is EMS: BACK PAINS/INJURY , the Reason column value would be EMS.

```
In [10]: df['Reason'] = df['title'].apply(lambda title: title.split(':')[0])

         What is the most common Reason for a 911 call based off of this new column?

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

Out[11]: EMS        48877
         Traffic    35695
         Fire       14920
         Name: Reason, dtype: int64
```

Figure 1: It shows how we created a reason field with the help of lambda function.

TimeStamp

Now we will convert the time field which is currently a string into the Timestamp. We will also grab the date , month , year of the TimeStamp field and we will map days of week to each date using map() function .

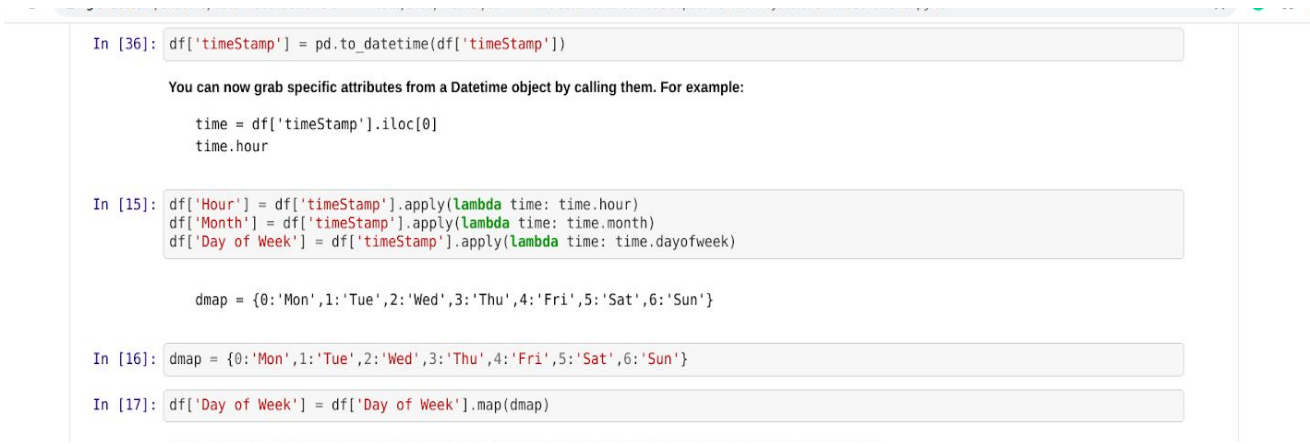


Figure 2:It shows the Convert shows conversion into DateTime field and mapping of days

Now our dataset is ready and we can perform visualization.I have used matplotlib and seaborn Libraries for visualization.

Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib. Matplotlib was originally written by John D. Hunter, since then it has an active development community, and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012, and further joined by Thomas Caswell.

Seaborn

Seaborn is a library for making statistical graphics in Python. It is built on top of matplotlib and closely integrated with pandas data structures. Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots.

Operations Performed:

1. Count Plot

2.Simple Plot

3.Line Plot

4.Heat Map

5.Cluster Map

1. Count Plot:

A count plot can be thought of as a histogram across a categorical, instead of quantitative, variable. The basic API and options are identical to those for barplot(), so you can compare counts across nested variables.Input data can be passed in a variety of formats, including:

- Vectors of data represented as lists, numpy arrays, or pandas Series objects passed directly to the x, y, and/or hue parameters.
- A “long-form” DataFrame, in which case the x, y, and hue variables will determine how the data are plotted.
- A “wide-form” DataFrame, such that each numeric column will be plotted.
- An array or list of vectors.

```
sns.countplot(x='Day of Week',data=df,hue='Reason',palette='viridis')
```

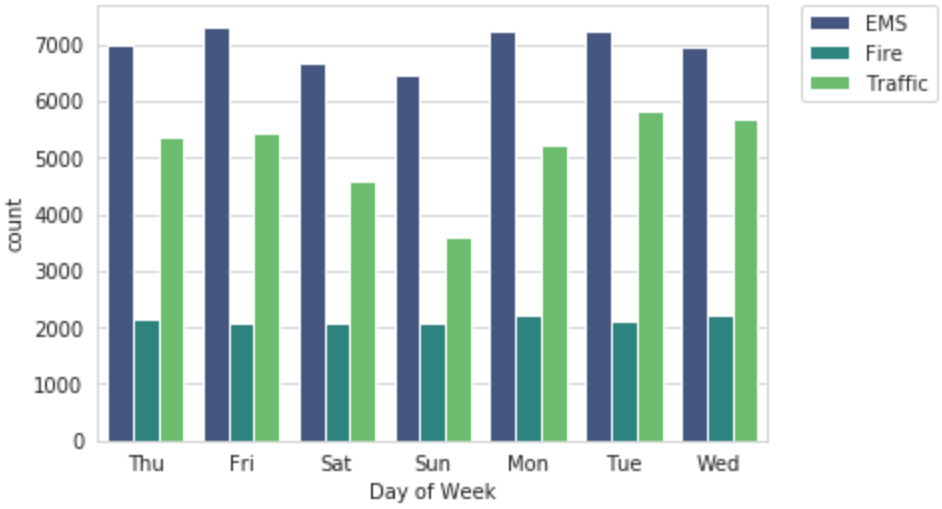


Figure 3: It shows count plot with the help of bar graph

2. Simple Plot:

We use plot() method to invoke the required plot. We can also plot multiple sets of data by passing in multiple sets of arguments of X and Y axis in the plot() method.

```
byMonth['twp'].plot()
```

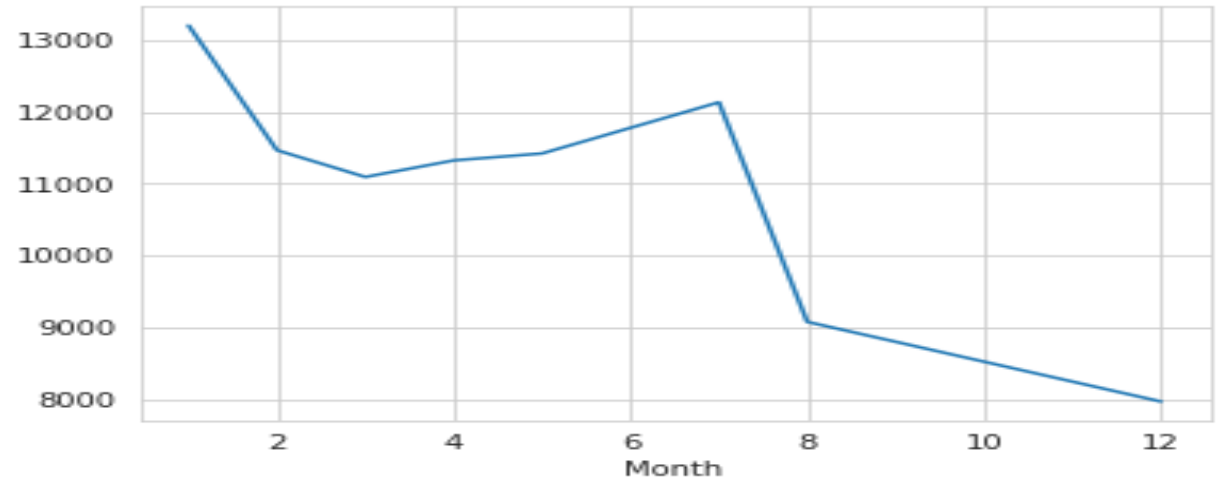


Figure 4: It shows the line plot between month and twp

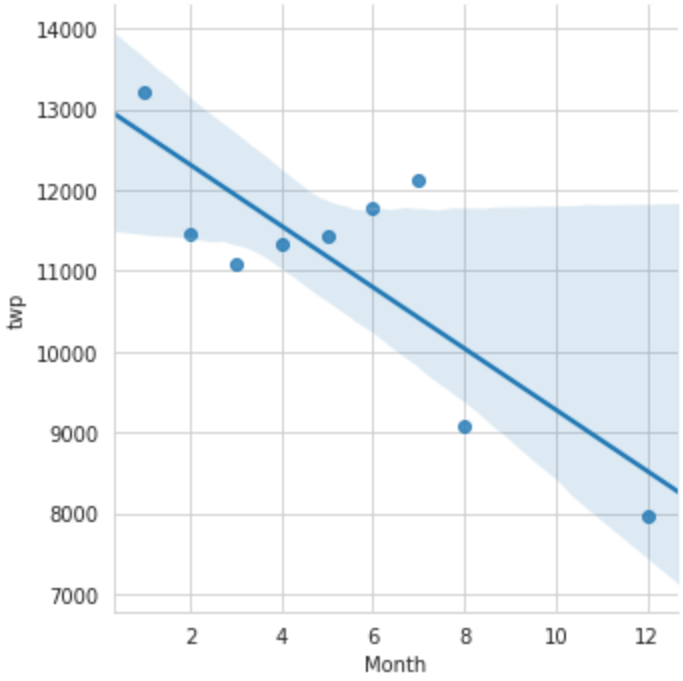
3. Line Plot:

The line plot (**lmplot**) is one of the most basic plots. It shows a line on a 2 dimensional plane. You can plot it with seaborn or matplotlib depending on your preference.

Plot data and regression model fits across FacetGrid. This function combines regplot() and Facetgrid. It is intended as a convenient interface to fit regression models across conditional subsets of a dataset.

When thinking about how to assign variables to different facets, a general rule is that it makes sense to use hue for the most important comparison, followed by col and row. However, always think about your particular dataset and the goals of the visualization you are creating.

```
sns.lmplot(x='Month',y='twp',data=byMonth.reset_index())
```



4. Heat Map:

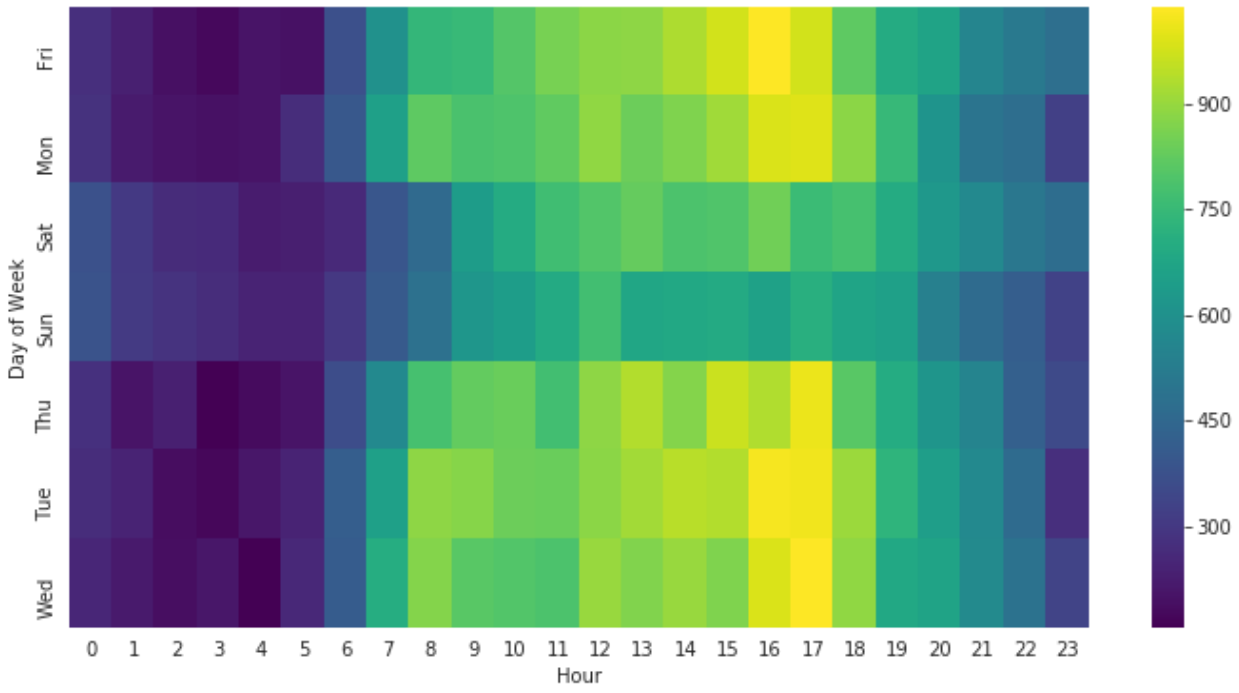
The heatmap is a way of representing the data in a 2-dimensional form. The data values are represented as colors in the graph. The goal of the heatmap is to provide a colored visual summary of information.

The values in the x axis and y axis for each block in the heatmap are called tick labels. The tick labels are added by default. If we want to remove the tick labels, we can set the xticklabel or ytickelabel attribute of seaborn heatmap to False.

We can add a label in the x axis by using the xlabel attribute of Matplotlib. You can change the color of seaborn heatmap by using the color map using the cmap attribute of the heatmap.

```
plt.figure(figsize=(12,6))

sns.heatmap(dayHour,cmap='viridis')
```



5. Cluster Map:

Plot a matrix dataset as a hierarchically-clustered heatmap. The returned object has a savefig method that should be used if you want to save the figure object without clipping the dendrograms.

Parameters:

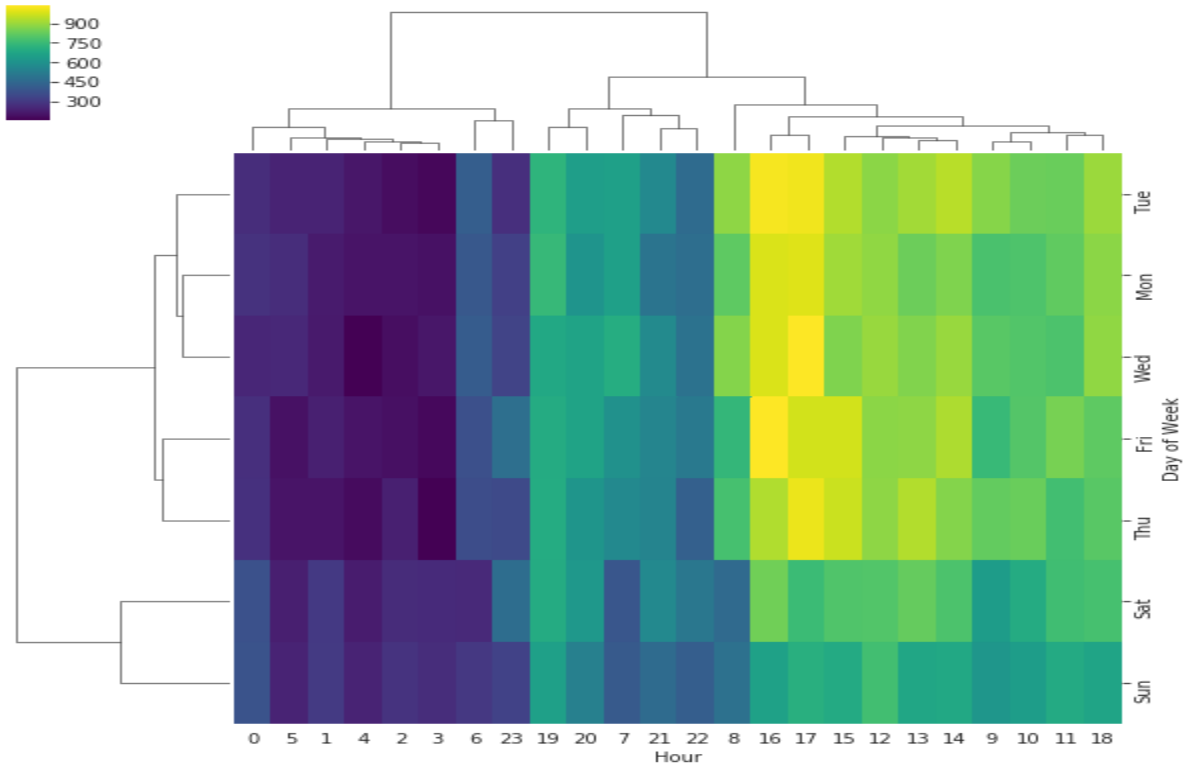
data: pandas.DataFrame. Rectangular data for clustering. Cannot contain NAs.

pivot_kws: dict, optional. If *data* is a tidy dataframe, can provide keyword arguments for pivot to create a rectangular dataframe.

method: str, optional. Linkage method to use for calculating clusters.

metric: str, optional. Distance metric to use for the data. See scipy.spatial.distance.pdist documentation for more options.

```
sns.clustermap(dayHour,cmap='viridis')
```



Conclusion: Different data mining operations can be performed using the Matplotlib tool. Matplotlib is a visualization tool that offers different components called widgets that offer techniques like simple data visualization, subset selection and pre-processing.