911 Calls Capstone Project

Data and Setup

Importing Required Packages

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

Reading in the csv file and creating Data Frame

```
from google.colab import drive
drive.mount('/content/drive')
df = pd.read_csv('./drive/MyDrive/Dataset/911.csv')
df.head()
```

```
Drive already mounted at /content/drive; to attempt to forcibly
remount, call drive.mount("/content/drive", force_remount=True).

         lat        lng
desc  \
0  40.297876 -75.581294  REINDEER CT & DEAD END;  NEW HANOVER; Station
...
1  40.258061 -75.264680  BRIAR PATH & WHITEMARSH LN;  HATFIELD
TOWNSHIP...
2  40.121182 -75.351975  HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-
St...
3  40.116153 -75.343513  AIRY ST & SWEDE ST;  NORRISTOWN; Station
308A;...
4  40.251492 -75.603350  CHERRYWOOD CT & DEAD END;  LOWER POTTSGROVE;
S...

       zip                    title            timeStamp
twp  \
0  19525.0   EMS: BACK PAINS/INJURY  2015-12-10 17:40:00        NEW
HANOVER
1  19446.0  EMS: DIABETIC EMERGENCY  2015-12-10 17:40:00  HATFIELD
TOWNSHIP
2  19401.0      Fire: GAS-ODOR/LEAK  2015-12-10 17:40:00
NORRISTOWN
3  19401.0   EMS: CARDIAC EMERGENCY  2015-12-10 17:40:01
NORRISTOWN
4      NaN          EMS: DIZZINESS  2015-12-10 17:40:01   LOWER
POTTSGROVE
```

```
                            addr  e
0      REINDEER CT & DEAD END  1
1  BRIAR PATH & WHITEMARSH LN  1
2                    HAWS AVE  1
3          AIRY ST & SWEDE ST  1
4    CHERRYWOOD CT & DEAD END  1
```

Checking the info() of the df

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99492 entries, 0 to 99491
Data columns (total 9 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   lat        99492 non-null  float64
 1   lng        99492 non-null  float64
 2   desc       99492 non-null  object 
 3   zip        86637 non-null  float64
 4   title      99492 non-null  object 
 5   timeStamp  99492 non-null  object 
 6   twp        99449 non-null  object 
 7   addr       98973 non-null  object 
 8   e          99492 non-null  int64  
dtypes: float64(3), int64(1), object(5)
memory usage: 6.8+ MB
```

Basic Questions

What are the top 5 zipcodes for 911 calls?

```
df['zip'].value_counts().head(5)
```

```
19401.0    6979
19464.0    6643
19403.0    4854
19446.0    4748
19406.0    3174
Name: zip, dtype: int64
```

What are the top 5 townships (twp) for 911 calls?

```
df['twp'].value_counts().head(5)
```

```
LOWER MERION    8443
ABINGTON        5977
NORRISTOWN      5890
UPPER MERION    5227
```

```
CHELTENHAM      4575
Name: twp, dtype: int64
```

How many unique title codes are there?

```
df['title'].unique
```

```
<bound method Series.unique of 0             EMS: BACK PAINS/INJURY
1            EMS: DIABETIC EMERGENCY
2                Fire: GAS-ODOR/LEAK
3             EMS: CARDIAC EMERGENCY
4                     EMS: DIZZINESS
                        ...
99487    Traffic: VEHICLE ACCIDENT -
99488    Traffic: VEHICLE ACCIDENT -
99489               EMS: FALL VICTIM
99490           EMS: NAUSEA/VOMITING
99491    Traffic: VEHICLE ACCIDENT -
Name: title, Length: 99492, dtype: object>
```

Creating new features

In the titles column there are "Reasons/Departments" specified before the title code. These are EMS, Fire, and Traffic. Using .apply()to create a new column called "Reason" that contains this string value.

```
def createReason(x):
  x = x.split(":")
  return x[0]

df['Reason'] = df['title'].apply(createReason)
df.head()
```

```
         lat        lng                                                
desc  \
0  40.297876 -75.581294  REINDEER CT & DEAD END;  NEW HANOVER; Station 
...
1  40.258061 -75.264680  BRIAR PATH & WHITEMARSH LN;  HATFIELD 
TOWNSHIP...
2  40.121182 -75.351975  HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-
St...
3  40.116153 -75.343513  AIRY ST & SWEDE ST;  NORRISTOWN; Station 
308A;...
4  40.251492 -75.603350  CHERRYWOOD CT & DEAD END;  LOWER POTTSGROVE; 
S...

       zip                    title            timeStamp                
twp  \
0  19525.0   EMS: BACK PAINS/INJURY  2015-12-10 17:40:00        NEW 
HANOVER
```

```
1  19446.0  EMS: DIABETIC EMERGENCY  2015-12-10 17:40:00  HATFIELD
TOWNSHIP
2  19401.0      Fire: GAS-ODOR/LEAK  2015-12-10 17:40:00
NORRISTOWN
3  19401.0   EMS: CARDIAC EMERGENCY  2015-12-10 17:40:01
NORRISTOWN
4      NaN          EMS: DIZZINESS  2015-12-10 17:40:01   LOWER
POTTSGROVE

                           addr  e Reason
0      REINDEER CT & DEAD END  1    EMS
1  BRIAR PATH & WHITEMARSH LN  1    EMS
2                    HAWS AVE  1   Fire
3          AIRY ST & SWEDE ST  1    EMS
4    CHERRYWOOD CT & DEAD END  1    EMS
```

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

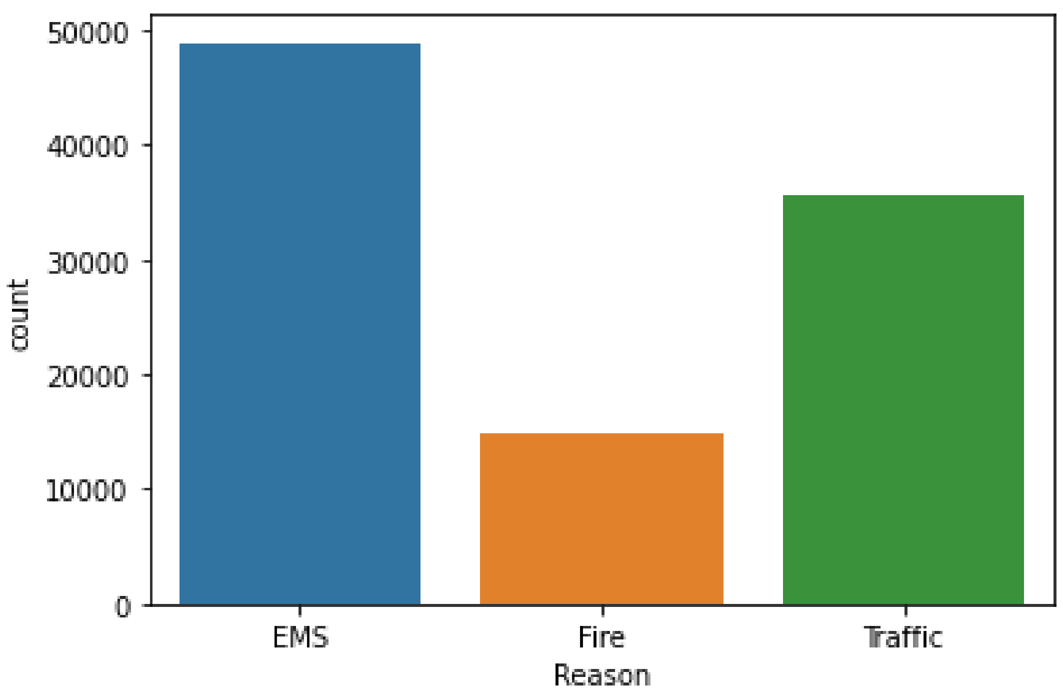
```
df['Reason'].value_counts()
```

```
EMS        48877
Traffic    35695
Fire       14920
Name: Reason, dtype: int64
```

Creating a countplot of 911 calls by Reason.

```
import seaborn as sns

sns.countplot(data=df, x='Reason')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe9d1680ed0>
```



What is the data type of the objects in the timeStamp column?

```
df['timeStamp'].dtype
```

```
dtype('O')
```

Using pd.to_datetime to convert the column from strings to DateTime objects.

```
df['timeStamp'] = pd.to_datetime(df['timeStamp'])
df.head()
```

```
         lat        lng
desc  \
0  40.297876 -75.581294  REINDEER CT & DEAD END;  NEW HANOVER; Station
...
1  40.258061 -75.264680  BRIAR PATH & WHITEMARSH LN;  HATFIELD
TOWNSHIP...
2  40.121182 -75.351975  HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-
St...
3  40.116153 -75.343513  AIRY ST & SWEDE ST;  NORRISTOWN; Station
308A;...
4  40.251492 -75.603350  CHERRYWOOD CT & DEAD END;  LOWER POTTSGROVE;
S...

       zip                    title           timeStamp
twp  \
0  19525.0   EMS: BACK PAINS/INJURY 2015-12-10 17:40:00        NEW
HANOVER
```

```
1  19446.0  EMS: DIABETIC EMERGENCY 2015-12-10 17:40:00  HATFIELD 
TOWNSHIP
2  19401.0      Fire: GAS-ODOR/LEAK 2015-12-10 17:40:00 
NORRISTOWN
3  19401.0   EMS: CARDIAC EMERGENCY 2015-12-10 17:40:01 
NORRISTOWN
4      NaN           EMS: DIZZINESS 2015-12-10 17:40:01   LOWER 
POTTSGROVE

                         addr  e Reason  
0      REINDEER CT & DEAD END  1    EMS  
1  BRIAR PATH & WHITEMARSH LN  1    EMS  
2                    HAWS AVE  1   Fire  
3          AIRY ST & SWEDE ST  1    EMS  
4    CHERRYWOOD CT & DEAD END  1    EMS
```

Now that the timestamp column is actually DateTime objects, using .apply() to create 3 new columns called Hour, Month, and Day of Week.

```
df['Hour'] = df['timeStamp'].apply(lambda x: x.hour)
df['Month'] = df['timeStamp'].apply(lambda x: x.month)
df['DayOfWeek'] = df['timeStamp'].apply(lambda x: x.dayofweek)

df.head()
```

```
         lat        lng                                               
desc  \
0  40.297876 -75.581294  REINDEER CT & DEAD END;  NEW HANOVER; Station 
...   
1  40.258061 -75.264680  BRIAR PATH & WHITEMARSH LN;  HATFIELD 
TOWNSHIP...   
2  40.121182 -75.351975  HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-
St...   
3  40.116153 -75.343513  AIRY ST & SWEDE ST;  NORRISTOWN; Station 
308A;...   
4  40.251492 -75.603350  CHERRYWOOD CT & DEAD END;  LOWER POTTSGROVE; 
S...   

       zip                    title           timeStamp                
twp  \
0  19525.0   EMS: BACK PAINS/INJURY 2015-12-10 17:40:00        NEW 
HANOVER   
1  19446.0  EMS: DIABETIC EMERGENCY 2015-12-10 17:40:00  HATFIELD 
TOWNSHIP   
2  19401.0      Fire: GAS-ODOR/LEAK 2015-12-10 17:40:00 
NORRISTOWN   
3  19401.0   EMS: CARDIAC EMERGENCY 2015-12-10 17:40:01 
NORRISTOWN   
4      NaN           EMS: DIZZINESS 2015-12-10 17:40:01   LOWER 
POTTSGROVE
```

```
                         addr  e Reason  Hour  Month  DayOfWeek
0      REINDEER CT & DEAD END  1    EMS    17     12          3
1  BRIAR PATH & WHITEMARSH LN  1    EMS    17     12          3
2                    HAWS AVE  1   Fire    17     12          3
3          AIRY ST & SWEDE ST  1    EMS    17     12          3
4    CHERRYWOOD CT & DEAD END  1    EMS    17     12          3
```

Using the .map() with dictionary to map the actual string names to the day of the week

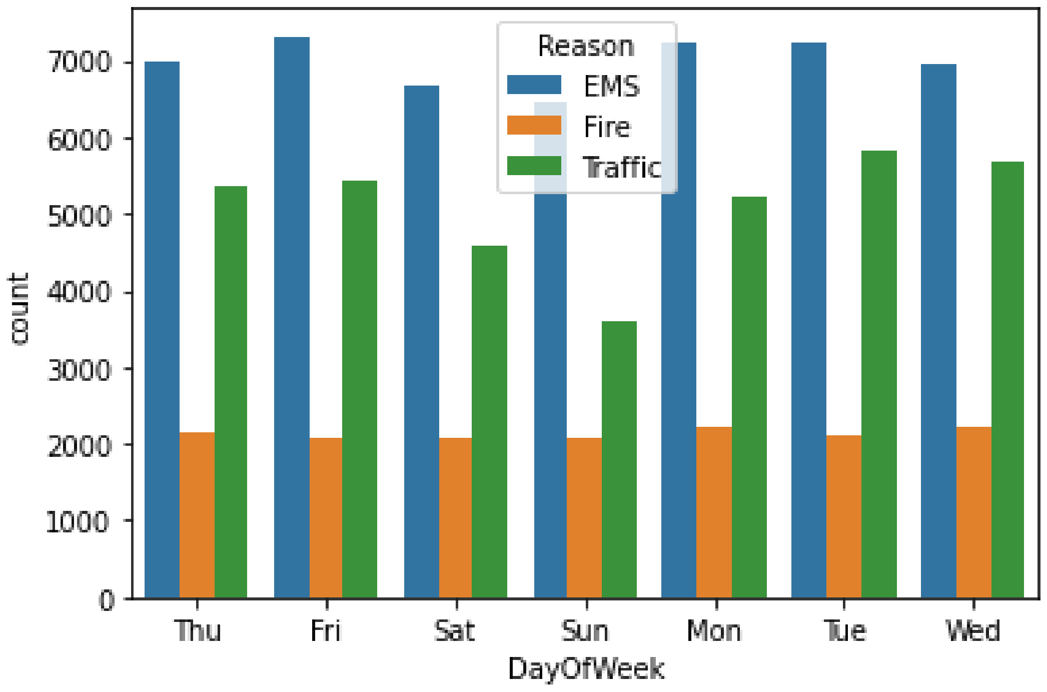
```
dmap = {0:'Mon',1:'Tue',2:'Wed',3:'Thu',4:'Fri',5:'Sat',6:'Sun'}
df['DayOfWeek'] = df['DayOfWeek'].map(dmap)

df.head()
```

```
         lat        lng                                              
desc  \
0  40.297876 -75.581294  REINDEER CT & DEAD END;  NEW HANOVER; Station 
...   
1  40.258061 -75.264680  BRIAR PATH & WHITEMARSH LN;  HATFIELD 
TOWNSHIP...   
2  40.121182 -75.351975  HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-
St...   
3  40.116153 -75.343513  AIRY ST & SWEDE ST;  NORRISTOWN; Station 
308A;...   
4  40.251492 -75.603350  CHERRYWOOD CT & DEAD END;  LOWER POTTSGROVE; 
S...   

       zip                    title           timeStamp                
twp  \
0  19525.0   EMS: BACK PAINS/INJURY 2015-12-10 17:40:00        NEW 
HANOVER   
1  19446.0  EMS: DIABETIC EMERGENCY 2015-12-10 17:40:00  HATFIELD 
TOWNSHIP   
2  19401.0      Fire: GAS-ODOR/LEAK 2015-12-10 17:40:00         
NORRISTOWN   
3  19401.0   EMS: CARDIAC EMERGENCY 2015-12-10 17:40:01         
NORRISTOWN   
4      NaN           EMS: DIZZINESS 2015-12-10 17:40:01   LOWER 
POTTSGROVE   

                         addr  e Reason  Hour  Month DayOfWeek  
0      REINDEER CT & DEAD END  1    EMS    17     12       Thu  
1  BRIAR PATH & WHITEMARSH LN  1    EMS    17     12       Thu  
2                    HAWS AVE  1   Fire    17     12       Thu  
3          AIRY ST & SWEDE ST  1    EMS    17     12       Thu  
4    CHERRYWOOD CT & DEAD END  1    EMS    17     12       Thu  
```

Creating a countplot of the Day of Week column with the hue based off of the Reason column.

```
sns.countplot(data=df, x='DayOfWeek', hue='Reason')
```

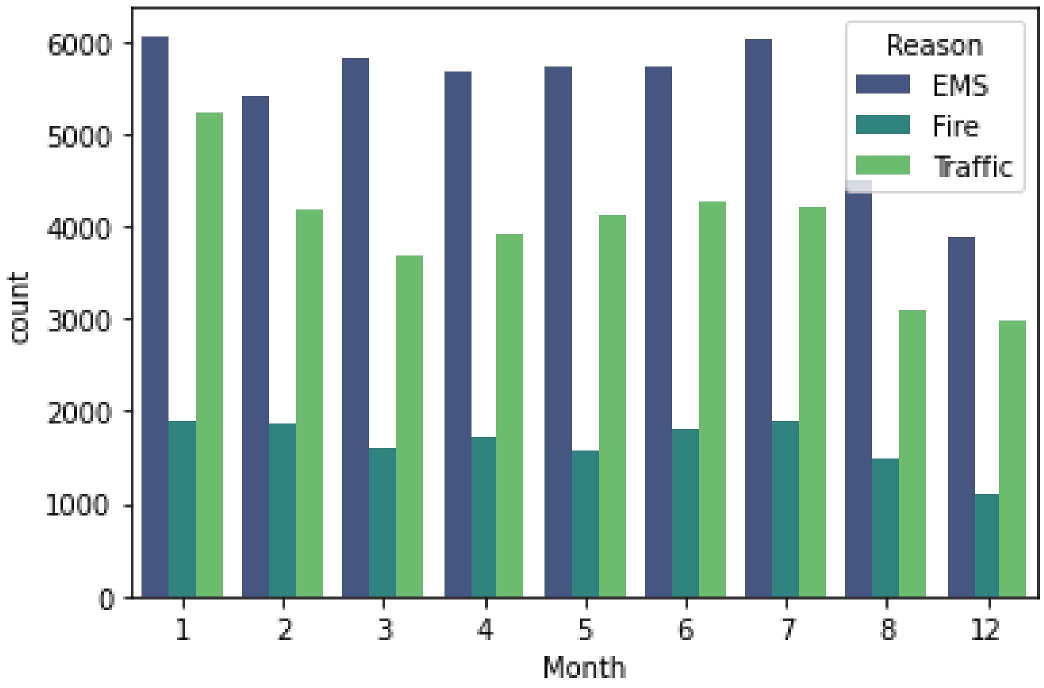
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe9d05ffb90>
```



Now doing the same for Month

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

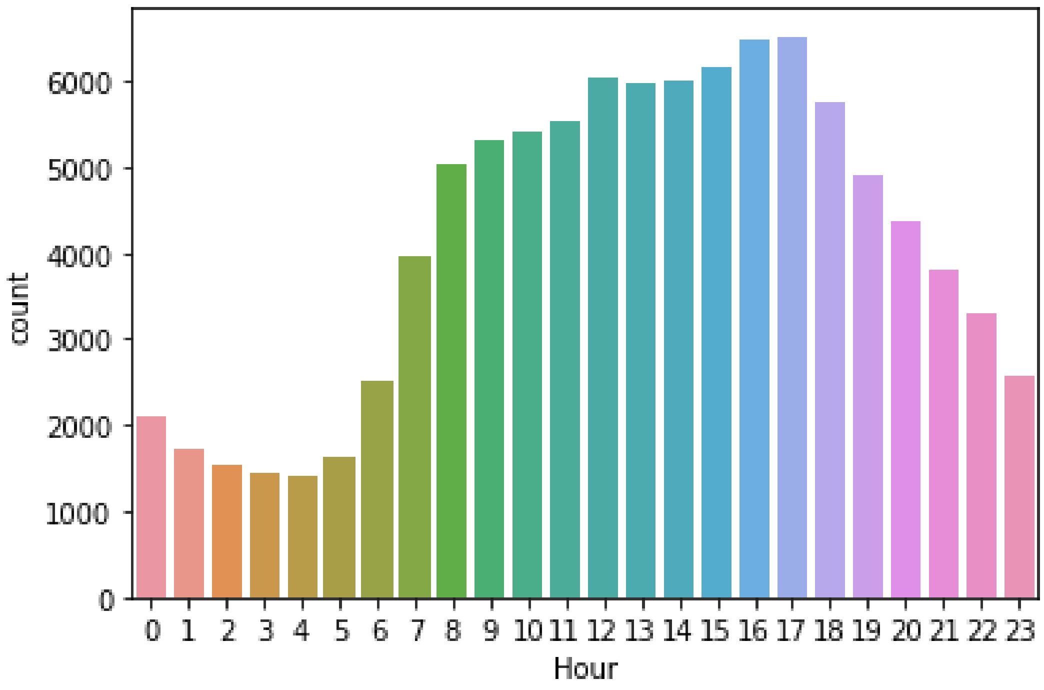
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe9d168e250>
```



Which Hour Has the most calls?

```
sns.countplot(data=df, x='Hour')
```

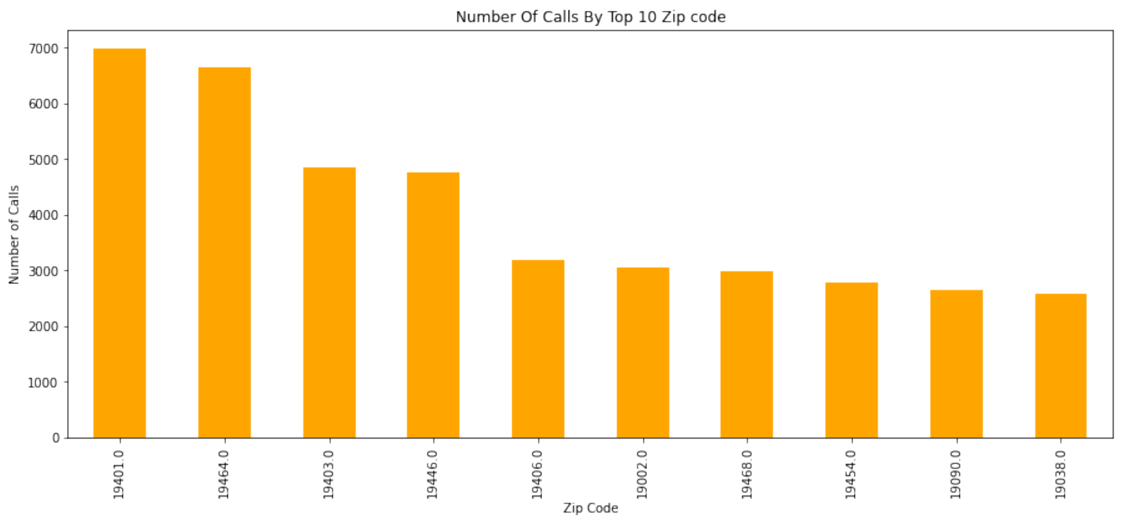
```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe9d0405a90>
```



Which Zip code Receives More Calls?

```
df['zip'].value_counts().head(10).plot(kind='bar', figsize=(15,6),
color='orange')
plt.title("Number Of Calls By Top 10 Zip code")
plt.xlabel('Zip Code')
plt.ylabel('Number of Calls')
```

```
Text(0, 0.5, 'Number of Calls')
```



Observations

1. 19401 Zip code receives the most 911 calls.
2. 911 receives the most calls from LOWER MERION.
3. 911 receives most calls for EMS with 48877 calls in the dataset.
4. Wednesdays record most calls for EMS.
5. Tuesday record most calls for traffic.
6. Mondays record most calls for Fire.
7. January and July have received the highest and almost same number of calls for EMS.
8. Most calls for traffic were received in January.
9. Most calls for fire were received in July.
10. 911 receives most calls between 15, 16, and 17 hours of the day.