In [2]:

```
__author__ = "NehaAVarshney"
__email__ = "varshney.n@husky.neu.edu"
```

Table of Contents

The Problem Data Overview

In this data set, the city of Seattle only receives 911 calls for four reasons

- *Latte Spills* i.e a hot latte spills all over your lap
- *Beavers Attack* i.e unsuspecting passersbys
- *Seal Attacks*
- *Marshawn Lynch Sightings* i.e people get very excited and choose to call 911 for some reason
- **More Insight is shared in Answers of the Questions Section**

Import Packages

In [3]:

```
# Importing the neccessary modules for data manipulation and visual representati
on

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib as matplot
import seaborn as sns
%matplotlib inline

# import plotly.plotly as py
```

Read the Data

In [4]:

```
incident_calls_df= pd.read_excel("rev data for test.xlsx")
```

In [5]:

```
# Examining the dataset
print ("Contains Incident Type with respect to loctations for total incidents : 
" + str(len(incident_calls_df)))
incident_calls_df.head()
```

```
Contains Incident Type with respect to loctations for total incident
s : 1514
```

Out[5]:

	Type	Latitude	Longitude	Report Location
0	Beaver Accident	47.6992	-122.2167	(47.6291923608656, -122.186728398282)
1	Beaver Accident	47.6977	-122.2164	(47.5576821104334, -122.156421437319)
2	Beaver Accident	47.6967	-122.2131	(47.6167258135906, -122.173139389518)
3	Beaver Accident	47.6971	-122.2178	(47.5370517340417, -122.197755316941)
4	Beaver Accident	47.6925	-122.2127	(47.6124577512516, -122.14272010056)

Data Quality Check

There are no null values in this data set, but if there is null value: we should impute it. Otherwise, it will impact the result of our analysis.

In [6]:

```
# checking to see if there are any missing values in our data set
incident_calls_df.isnull().any()
```

Out[6]:

```
Type               False
Latitude           False
Longitude          False
Report Location    False
dtype: bool
```

This summarize the information of dtype of index and column, MAKING IT FEASIBLE FOR FEATURE ENGINEERING non null values and memory usage.

In [7]:

```
incident_calls_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1514 entries, 0 to 1513
Data columns (total 4 columns):
Type                  1514 non-null object
Latitude              1514 non-null float64
Longitude             1514 non-null float64
Report Location       1514 non-null object
dtypes: float64(2), object(2)
memory usage: 47.4+ KB
```

Renaming the features to make the code more informative

In [8]:

```
incident_calls_df = incident_calls_df.rename(columns={'Type': 'Incident_Type',
                                    'Latitude':'Incident_Latitude',
                                    'Longitude':'Incident_Longitude',
                                     'Report Location': 'Report_Station_Location
'
                                   })

incident_calls_df.head(3)
```

Out[8]:

	Incident_Type	Incident_Latitude	Incident_Longitude	Report_Station_Location
0	Beaver Accident	47.6992	-122.2167	(47.6291923608656, -122.186728398282)
1	Beaver Accident	47.6977	-122.2164	(47.5576821104334, -122.156421437319)
2	Beaver Accident	47.6967	-122.2131	(47.6167258135906, -122.173139389518)

Feature Engineering

Adding a new feature:

- [Aggregating the latitude and longitude as Incident_Location]

In [9]:

```
incident_calls_df["Incident_Location"] = "("+incident_calls_df["Incident_Latitud
e"].map(str)+","+incident_calls_df["Incident_Longitude"].map(str)+")"
```

In [10]:

```
incident_calls_df['frequency'] = incident_calls_df['Incident_Location'].map(inci
dent_calls_df['Incident_Location'].value_counts())
```

In [11]:

```
incident_calls_df.to_excel("incident_original.xlsx")
```

In [12]:

```
incident_calls_df.sort_values(by=['frequency'], ascending=False).head(3)
```

Out[12]:

	Incident_Type	Incident_Latitude	Incident_Longitude	Report_Station_Location	Incident_L
1205	Marshawn Lynch Sighting	47.5199	-122.2684	(47.519853, -122.268364)	(47.5199,-1:
1202	Marshawn Lynch Sighting	47.5199	-122.2684	(47.519853, -122.268364)	(47.5199,-1:
1208	Marshawn Lynch Sighting	47.5199	-122.2684	(47.519853, -122.268364)	(47.5199,-1:

Descriptive Statistics

Getting the Description of all the features in our Dataset

Summarizing the descriptive statistics of the Features.

1) ***Incident_Type*** :

As provided in the problem statement, there are 4 unique types of reasons to call 911 and the heighest occuring number is 508 for a type of reason

2) ***Incident_Longitude and Incident_Latitude***:

I understood the dispersion of the data points around the mean, and got to know that the Longitude and Latitude values are not not much spread out (as Std is ~0.5 and ~0.09)

3) ***Incident_Location and Report_Station_Location***:

I understood that, there are some repetitive locations i.e locations from where call have been made also, the report locations

Curious Step : to check if the reason and report station for all the calls from the repetitive location is same or not, same for the report station location.

Reason to this thought is : If the calls are for the same reason, we can definitely segment the location with reason type.

- Samples for Incident Location, where multiple calls have been made from

In [13]:

```
df_column_headers= list(incident_calls_df)
```

In [14]:

```
for column1 in df_column_headers:
    print("Description for Feature:  ",column1)
    print(incident_calls_df[str(column1)].describe())
    print("\n")
```

```
Description for Feature:   Incident_Type
count                 1514
unique                   4
top        Beaver Accident
freq                   508
Name: Incident_Type, dtype: object


Description for Feature:   Incident_Latitude
count    1514.000000
mean       47.618480
std         0.051916
min        47.500200
25%        47.586008
```

```
50%        47.608487
75%        47.672450
max        47.732000
Name: Incident_Latitude, dtype: float64


Description for Feature:   Incident_Longitude
count    1514.000000
mean     -122.284465
std         0.089676
min      -122.469940
25%      -122.355300
50%      -122.301850
75%      -122.185650
max      -122.140100
Name: Incident_Longitude, dtype: float64


Description for Feature:   Report_Station_Location
count                          1514
unique                         1468
top       (47.519853, -122.268364)
freq                              9
Name: Report_Station_Location, dtype: object


Description for Feature:   Incident_Location
count                     1514
unique                    1478
top       (47.5199,-122.2684)
freq                         9
Name: Incident_Location, dtype: object


Description for Feature:   frequency
count    1514.000000
mean        1.143989
std         0.955248
min         1.000000
25%         1.000000
50%         1.000000
75%         1.000000
max         9.000000
Name: frequency, dtype: float64
```

Info() Analysis:

Checking for Inconsistencies in data

In [15]:

```
# Checking the type of our features.
incident_calls_df.dtypes
```

Out[15]:

```
Incident_Type                object
Incident_Latitude           float64
Incident_Longitude          float64
Report_Station_Location      object
Incident_Location            object
frequency                     int64
dtype: object
```

In [16]:

```
#How many incidents are there in the dataset
incident_calls_df.shape
```

Out[16]:

```
(1514, 6)
```

Exploring The Data

What is the most common reason for calling 911?

Steps Performed:

- Calculated the 911 call count for each reason
- Calculated the rate for each reason i.e (number of samples for each type / total number of samples)
- i.e What is rate of calling 911 for each reason?

Here, Taking the incident type with maximum number of 911 calls as the most common reason.

***Answer: *Beaver Spills* is the most common reason to call 911, with total calls of 508 i.e ~34% of the 911 calls were for Beaver Accident.

In [17]:

```
total_incidents= len(incident_calls_df)
incident_count = incident_calls_df.Incident_Type.value_counts()
incident_rate = incident_calls_df.Incident_Type.value_counts()/total_incidents

print("Printing the count of reasons, i.e total reports under each reason")
print("\n")
print(incident_count)
print("\n")
print("Normalizing the value count as rate, to get the clear and relative unders
tanding")
print("\n")
print(incident_rate)
```

```
Printing the count of reasons, i.e total reports under each reason


Beaver Accident            508
Latte Spills               416
Marshawn Lynch Sighting    324
Seal Attack                266
Name: Incident_Type, dtype: int64


Normalizing the value count as rate, to get the clear and relative u
nderstanding


Beaver Accident            0.335535
Latte Spills               0.274769
Marshawn Lynch Sighting    0.214003
Seal Attack                0.175694
Name: Incident_Type, dtype: float64
```

Display these results graphically

Answer I have chosen countplot from seaborn library to display the count of each observations in *categorical bin (Incident Types)* using bars

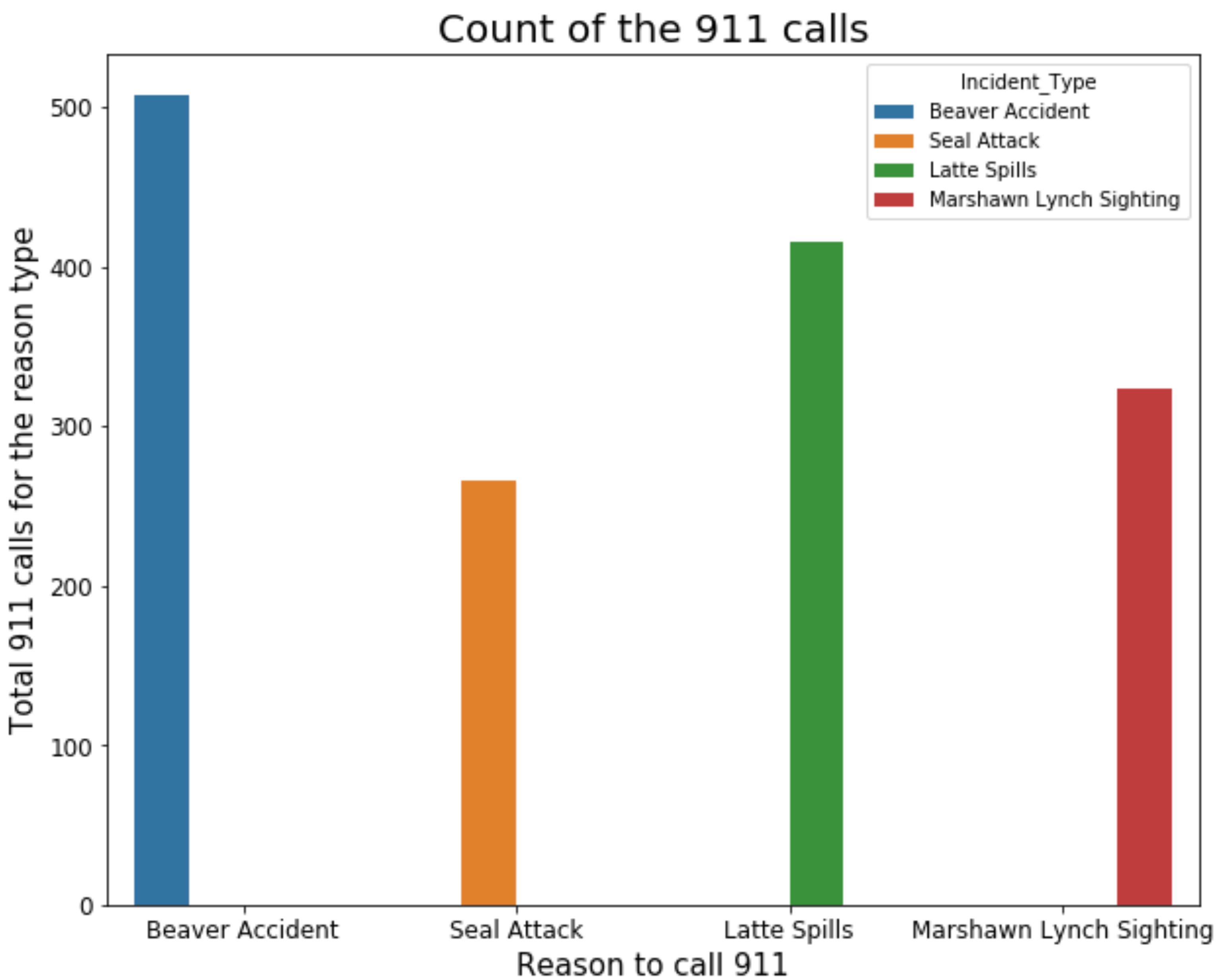
Interpretation from Graph : **Breaver Accident** has the highest count of 911 calls. Therefore, being the most common reason for 911 call.

In [18]:

```
# resizing the plot size
plt.figure(figsize=(10,8))
# Plotting the countplot from seaborn library
ax=sns.countplot(x="Incident_Type", data=incident_calls_df, hue="Incident_Type")


# Setting the title, increasing font size
ax.axes.set_title("Count of the 911 calls",fontsize=20)
# Setting the x and y label i.e name along axis, increasing font size
ax.set_xlabel("Reason to call 911",fontsize=15)
ax.set_ylabel("Total 911 calls for the reason type",fontsize=15)
# Increasing the fontsize of values of each label
ax.tick_params(labelsize=12)

plt.show
```

Out[18]:

```
<function matplotlib.pyplot.show(*args, **kw)>
```



Visualizations and Mislabelling

creating a graph of the 911 calls using the 'Latitude' and 'Longitude' (graph type is up to you) (differentiate call type using colors)

_**Answer** : Created Scatter Plot of seaborn library to represent the reason of call for each Latitude and Longitude.

Scatter plots show how much one variable is affected by another i.e here, the relation of reason of the calls to latitiude and longitude of the incident.

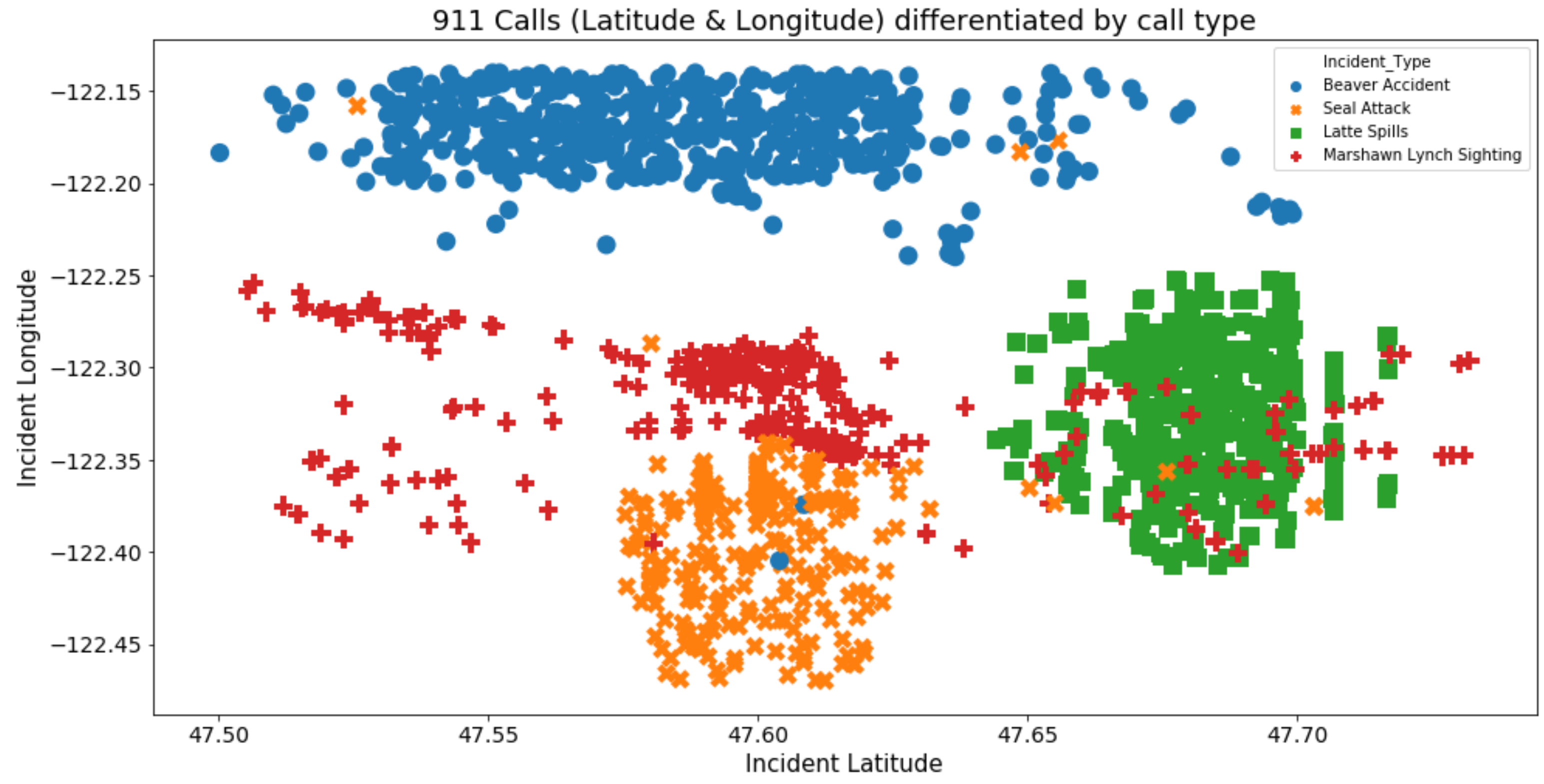
This Graph represents that most of the Incident Types are SEGMENTED on the Latitude and Longitude, i.e interpreted by the same color data points plotted together for respective latitude and longitude cordinates

In [19]:

```
import matplotlib.pyplot as plt
import matplotlib as matplot
import seaborn as sns
%matplotlib inline

cmap=plt.cm.rainbow
# resizing the plot size
plt.figure(figsize=(16,8))

norm = matplot.colors.BoundaryNorm(np.arange(0,10,1), cmap.N)

ax = sns.scatterplot(x="Incident_Latitude", y="Incident_Longitude",
                     hue="Incident_Type",style="Incident_Type", s= 150,
                     norm = norm, data=incident_calls_df, edgecolor='none')

ax.set_title("911 Calls (Latitude & Longitude) differentiated by call type", fon
tsize =18)
ax.set_xlabel("Incident Latitude", fontsize = 15)
ax.set_ylabel("Incident Longitude",fontsize=15)

# Increasing the fontsize of values of each label
ax.tick_params(labelsize=14)

plt.show
```

Out[19]:

```
<function matplotlib.pyplot.show(*args, **kw)>
```



- [This is analysed by the k means clustering algorithm]()

- ***Are there any Data Points that look mislabeled?***

Answer : Yes, potentially there is some mislabeled data as interepreted by the graph above.

Summary of the above graph:

- There are some Seal Attack data points in the cluster of Beaver Accident, they are potentially mislabeled
- There are few Beaver Accident data pointsin the segment of Seal Attack
- There is a Big Overlap between Latte Spills and Marshawn Lynch Sighting

If there is more information about the data, I can do further interpretations regarding these data points

K-Means Impelmentation

If we were to use only 'Latitude' and 'Longitude', could we make an intelligent decision as to why a resident dialed 911? (In other words, if we take off the labels - can we still determine which category a 911 call would most likely fall into?)

Yes, if we were to use only 'Latitude' and 'Longitude', we could make an intelligent decision as to why a resident dialed 911.

The segmentation of call type as seen in the question 2.A, clearly indicates that latitude and longitude can be used to to decide on the reason of the call.

For example, I am a 911 responder, if I know that calls from location A are due to beaver accident. Now, I got a new call which is from the next building of location A. It will be very probable that this call is for beaver accident too.

Also, in the analysis it is seen that: In case when, multiple calls are made from one location, they have reported the same reason. Therefore, if we take off the labels - can we still determine which category a 911 call would most likely fall into.

Algorithm : ***K-Means Clustering***

Reason being, The Variance(spread of information) of a reason of the call is mostly retained in a cluster, i.e in a segment

Description : ***It can be used as algorithm for Summarization of data, dividing a group of data points into clusters, where each point in the cluster is similar to each other and dissimilar to data points from other clusters.***

Reason To Choosing It :

- We can detect the area with multiple calls as a proxy of incidents
- We can infer the Hot Spots of the Incident (Reason)
- This will be a good sign as we can infer the probable location of the specific attack
- Will get to know the overall Dense Area between all the reasons of calls
- Also, in the below analysis, I found out that there are locations from where reason of same type were reported many times

Analysing the reason, in case of multiple calls from same location i.e location as a proxy of attack(reason)

Note: The below functions are just the sample of the idea of how this can be achieved. *They can be simplified further!!*

Summarizing the below part of analysis:

It is seen in the obtained samples(via code snipped below) that:

- There are 14 locations from where, multiple calls were made to report
- But, the many calls from the same location were placed to report the same reason each time, as seen below.
- Therefore, we can apply segementation(clustering) to intelligently retrieve the reason of call from seeing the location
- To quantifiably convert this analysis into result, I will implement K-means clustering

In [20]:

```
# Object Oriented Code Structure, increasing modularity of the code

#Function to get the list of locations with more than one call
def get_list_incident_location(list_incident):
    len_incident=len(list_incident)
    list_more_reports=[]

    for j in range(0,len_incident-4):
        if(j%2!=0):
            list_more_reports.append(list_incident[j])
    return list_more_reports

# Function to print the samples of the above locations.
#Reason? To check the integrity of calls made from one location

# df['frequency'] = df['county'].map(df['county'].value_counts())

def print_samples_incident(l1):
    for j in l1:
        print("\n")
        print("The value for reason(incident) location", j)
        print("\n")
        print(incident_calls_df.loc[incident_calls_df['Incident_Location'] == j]
)

        print("----------------------------------------------------")
```

In [21]:

```
# getting the count for each incident location
df_Incident=incident_calls_df.groupby('Incident_Location').Incident_Location.cou
nt()

#Getting the list of the incident locations, where the count is > 1i.e have been
repeated
# the output was the list of list, so working with data modelling



list1 = [df_Incident[df_Incident>1]][0]


list1= str(list1).split()
# print(list1)

list_incident_location= get_list_incident_location(list1)

print("Number of locations with multiple calls is: ", len(list_incident_location
))
```

```
Number of locations with multiple calls is:  14
```

In [22]:

```
print_samples_incident(list_incident_location)
```

```
The value for reason(incident) location (47.5199,-122.2684)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1200  Marshawn Lynch Sighting            47.5199           -122.2684
1201  Marshawn Lynch Sighting            47.5199           -122.2684
1202  Marshawn Lynch Sighting            47.5199           -122.2684
1203  Marshawn Lynch Sighting            47.5199           -122.2684
1204  Marshawn Lynch Sighting            47.5199           -122.2684
1205  Marshawn Lynch Sighting            47.5199           -122.2684
1206  Marshawn Lynch Sighting            47.5199           -122.2684
1207  Marshawn Lynch Sighting            47.5199           -122.2684
1208  Marshawn Lynch Sighting            47.5199           -122.2684

         Report_Station_Location    Incident_Location  frequency
1200  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1201  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1202  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1203  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1204  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1205  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1206  (47.519853, -122.268364)  (47.5199,-122.2684)          9
1207  (47.519853, -122.268364)  (47.5199,-122.2684)          9
```

```
1208  (47.519853, -122.268364)  (47.5199,-122.2684)          9
---------------------------------------------------
```

The value for reason(incident) location (47.5233,-122.27)

```
                Incident_Type  Incident_Latitude  Incident_Longitude  \
1352  Marshawn Lynch Sighting            47.5233             -122.27
1363  Marshawn Lynch Sighting            47.5233             -122.27

       Report_Station_Location  Incident_Location  frequency
1352  (47.523307, -122.269986)  (47.5233,-122.27)          2
1363  (47.523307, -122.269986)  (47.5233,-122.27)          2
---------------------------------------------------
```

The value for reason(incident) location (47.5389,-122.2822)

```
                Incident_Type  Incident_Latitude  Incident_Longitude  \
1269  Marshawn Lynch Sighting            47.5389           -122.2822
1341  Marshawn Lynch Sighting            47.5389           -122.2822

       Report_Station_Location    Incident_Location  frequency
1269  (47.538884, -122.282179)  (47.5389,-122.2822)          2
1341  (47.538884, -122.282179)  (47.5389,-122.2822)          2
---------------------------------------------------
```

The value for reason(incident) location (47.5857,-122.3342)

```
                Incident_Type  Incident_Latitude  Incident_Longitude  \
1210  Marshawn Lynch Sighting            47.5857           -122.3342
1264  Marshawn Lynch Sighting            47.5857           -122.3342
1355  Marshawn Lynch Sighting            47.5857           -122.3342

       Report_Station_Location    Incident_Location  frequency
1210  (47.585705, -122.334198)  (47.5857,-122.3342)          3
1264  (47.585733, -122.334198)  (47.5857,-122.3342)          3
1355  (47.585747, -122.334198)  (47.5857,-122.3342)          3
---------------------------------------------------
```

The value for reason(incident) location (47.5972,-122.1521)

```
       Incident_Type  Incident_Latitude  Incident_Longitude  \
446  Beaver Accident            47.5972           -122.1521
```

```
496  Beaver Accident            47.5972          -122.1521

                  Report_Station_Location    Incident_Location  fre
quency
446   (47.5971754244305, -122.15207690789)  (47.5972,-122.1521)
2
496  (47.5972017146939, -122.152079011347)  (47.5972,-122.1521)
2
---------------------------------------------------
```

The value for reason(incident) location (47.6002,-122.3305)

```
                Incident_Type  Incident_Latitude  Incident_Longitude
\
1254  Marshawn Lynch Sighting            47.6002           -122.3305
1301  Marshawn Lynch Sighting            47.6002           -122.3305
1367  Marshawn Lynch Sighting            47.6002           -122.3305

       Report_Station_Location    Incident_Location  frequency
1254  (47.600194, -122.330541)  (47.6002,-122.3305)          3
1301  (47.600194, -122.330541)  (47.6002,-122.3305)          3
1367  (47.600194, -122.330541)  (47.6002,-122.3305)          3
---------------------------------------------------
```

The value for reason(incident) location (47.607,-122.2894)

```
                Incident_Type  Incident_Latitude  Incident_Longitude
\
1192  Marshawn Lynch Sighting             47.607           -122.2894
1193  Marshawn Lynch Sighting             47.607           -122.2894
1194  Marshawn Lynch Sighting             47.607           -122.2894
1195  Marshawn Lynch Sighting             47.607           -122.2894
1196  Marshawn Lynch Sighting             47.607           -122.2894
1197  Marshawn Lynch Sighting             47.607           -122.2894
1198  Marshawn Lynch Sighting             47.607           -122.2894
1199  Marshawn Lynch Sighting             47.607           -122.2894

      Report_Station_Location   Incident_Location  frequency
1192  (47.607026, -122.28944)  (47.607,-122.2894)          8
1193  (47.607026, -122.28944)  (47.607,-122.2894)          8
1194  (47.607026, -122.28944)  (47.607,-122.2894)          8
1195  (47.607026, -122.28944)  (47.607,-122.2894)          8
1196  (47.607026, -122.28944)  (47.607,-122.2894)          8
1197  (47.607026, -122.28944)  (47.607,-122.2894)          8
1198  (47.607026, -122.28944)  (47.607,-122.2894)          8
1199  (47.607026, -122.28944)  (47.607,-122.2894)          8
---------------------------------------------------
```

```
The value for reason(incident) location (47.6098,-122.3378)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1182  Marshawn Lynch Sighting            47.6098           -122.3378
1305  Marshawn Lynch Sighting            47.6098           -122.3378

      Report_Station_Location    Incident_Location  frequency
1182  (47.60975, -122.337793)  (47.6098,-122.3378)          2
1305  (47.60975, -122.337793)  (47.6098,-122.3378)          2
---------------------------------------------------


The value for reason(incident) location (47.6107,-122.3387)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1289  Marshawn Lynch Sighting            47.6107           -122.3387
1295  Marshawn Lynch Sighting            47.6107           -122.3387

       Report_Station_Location    Incident_Location  frequency
1289  (47.610743, -122.338702)  (47.6107,-122.3387)          2
1295  (47.610743, -122.338702)  (47.6107,-122.3387)          2
---------------------------------------------------


The value for reason(incident) location (47.6112,-122.3376)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1231  Marshawn Lynch Sighting            47.6112           -122.3376
1306  Marshawn Lynch Sighting            47.6112           -122.3376
1330  Marshawn Lynch Sighting            47.6112           -122.3376

       Report_Station_Location    Incident_Location  frequency
1231  (47.611207, -122.337592)  (47.6112,-122.3376)          3
1306  (47.611207, -122.337592)  (47.6112,-122.3376)          3
1330  (47.611207, -122.337592)  (47.6112,-122.3376)          3
---------------------------------------------------


The value for reason(incident) location (47.6134,-122.3465)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1227  Marshawn Lynch Sighting            47.6134           -122.3465
1319  Marshawn Lynch Sighting            47.6134           -122.3465

       Report_Station_Location    Incident_Location  frequency
```

```
1227  (47.613375, -122.346513)  (47.6134,-122.3465)          2
1319  (47.613375, -122.346513)  (47.6134,-122.3465)          2
---------------------------------------------------


The value for reason(incident) location (47.617,-122.3234)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1184  Marshawn Lynch Sighting             47.617           -122.3234
1185  Marshawn Lynch Sighting             47.617           -122.3234
1186  Marshawn Lynch Sighting             47.617           -122.3234
1187  Marshawn Lynch Sighting             47.617           -122.3234
1188  Marshawn Lynch Sighting             47.617           -122.3234
1189  Marshawn Lynch Sighting             47.617           -122.3234
1190  Marshawn Lynch Sighting             47.617           -122.3234
1191  Marshawn Lynch Sighting             47.617           -122.3234

         Report_Station_Location   Incident_Location  frequency
1184  (47.616984, -122.323442)  (47.617,-122.3234)          8
1185  (47.616984, -122.323442)  (47.617,-122.3234)          8
1186  (47.616984, -122.323442)  (47.617,-122.3234)          8
1187  (47.616984, -122.323442)  (47.617,-122.3234)          8
1188  (47.616984, -122.323442)  (47.617,-122.3234)          8
1189  (47.616984, -122.323442)  (47.617,-122.3234)          8
1190  (47.616984, -122.323442)  (47.617,-122.3234)          8
1191  (47.616984, -122.323442)  (47.617,-122.3234)          8
---------------------------------------------------


The value for reason(incident) location (47.7068,-122.3232)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1259  Marshawn Lynch Sighting            47.7068           -122.3232
1307  Marshawn Lynch Sighting            47.7068           -122.3232

         Report_Station_Location    Incident_Location  frequency
1259  (47.706831, -122.323224)  (47.7068,-122.3232)          2
1307  (47.706795, -122.323223)  (47.7068,-122.3232)          2
---------------------------------------------------


The value for reason(incident) location (47.7269,-122.3477)


                Incident_Type  Incident_Latitude  Incident_Longitude  \
1318  Marshawn Lynch Sighting            47.7269           -122.3477
1340  Marshawn Lynch Sighting            47.7269           -122.3477
```

```
      Report_Station_Location    Incident_Location  frequency
1318  (47.726934, -122.347715)  (47.7269,-122.3477)          2
1340  (47.726934, -122.347715)  (47.7269,-122.3477)          2
----------------------------------------------------
```

Does the algorithm chosen utilize Euclidean distance? Should we be concerned that 'Latitude' and 'Longitude' are not necessarily Euclidean?

Answer: Yes the algorithm chosen i.e K-MEANS utilizes the Euclidean Distance.

Why Distance matters in K-Means Algorithm: Distance measures how similar two elements are, thus influencing the shape of the clusters.

k-means generates clusters based on the Euclidean distance between points—meaning the straight-line distance between two pins in the map.

But as we know, the Earth isn’t flat so this approximation will affect the clusters being generated

Therefore, **Yes we should be concerned that 'Latitude' and 'Longitude' are not Euclidean.**

Instead, we should be using Geographical (spatial) distance i.e the distance measured along the surface of the earth i.e calculating lengths of the shortest curve between two points along the surface of the Earth.

Hierarchical clustering, PAM, CLARA, and DBSCAN are the popular examples of using spatial distances.

displaing the results of algorithm, along with the associated code

Import Necessary Libraries

In [23]:

```
import pylab as pl
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA
```

Number of Clusters?

The number of clusters, explains “most” of the variance in the data, and choose it as the optimum value of k i.e clusters

Here, *Taking cluster size as 4, as there are 4 types of reasons.*

In the situation when number of types or segments are not provided, we apply the elblow method to calculate the optimum number of clusteres i.e the Optimum value of K.

In [24]:

```
# from sklearn.cluster import KMeans # Importing KMeans
kmeans_1 = KMeans(n_clusters=4)
# Using fit_predict to cluster the dataset
df_coord = incident_calls_df[['Incident_Longitude','Incident_Latitude','Incident
_Type']]
# predictions = kmeans_1.fit_predict(X)
```

In [25]:

```
df_coord.head()
```

Out[25]:

	Incident_Longitude	Incident_Latitude	Incident_Type
0	-122.2167	47.6992	Beaver Accident
1	-122.2164	47.6977	Beaver Accident
2	-122.2131	47.6967	Beaver Accident
3	-122.2178	47.6971	Beaver Accident
4	-122.2127	47.6925	Beaver Accident

In [26]:

```
from sklearn.cluster import KMeans # Importing KMeans
# Creating an instance of KMeans with 4 clusters
kmeans_1 = KMeans(n_clusters=4)
# Using fit_predict to cluster the dataset
X = df_coord[['Incident_Longitude','Incident_Latitude']].values
predictions = kmeans_1.fit_predict(X)
```

In [27]:

```
clustered = pd.concat([df_coord.reset_index(),
                       pd.DataFrame({'Cluster':predictions})],
                      axis=1)
```

In [28]:

```
clustered.drop('index', axis=1, inplace=True)
```

In [29]:

```
conditions = [
    clustered['Cluster'] == 0,
    clustered['Cluster'] == 1,
    clustered['Cluster'] == 2,
    clustered['Cluster'] == 3,
    ]
choices = ['Beaver Accident','Latte Spills', 'Marshawn Lynch Sighting', 'Seal At
tack' ]
clustered['Reason Type'] = np.select(conditions, choices, default='black')
```

In [30]:

```
clustered.sort_values(by=['Cluster']).head(2)
```

Out[30]:

	Incident_Longitude	Incident_Latitude	Incident_Type	Cluster	Reason Type
0	-122.2167	47.6992	Beaver Accident	0	Beaver Accident
657	-122.3288	47.6551	Latte Spills	0	Beaver Accident

In [31]:

```
clustered[clustered['Cluster']==2].head(2)
```

Out[31]:

	Incident_Longitude	Incident_Latitude	Incident_Type	Cluster	Reason Type
11	-122.240000	47.636600	Beaver Accident	2	Marshawn Lynch Sighting
975	-122.309264	47.592783	Marshawn Lynch Sighting	2	Marshawn Lynch Sighting

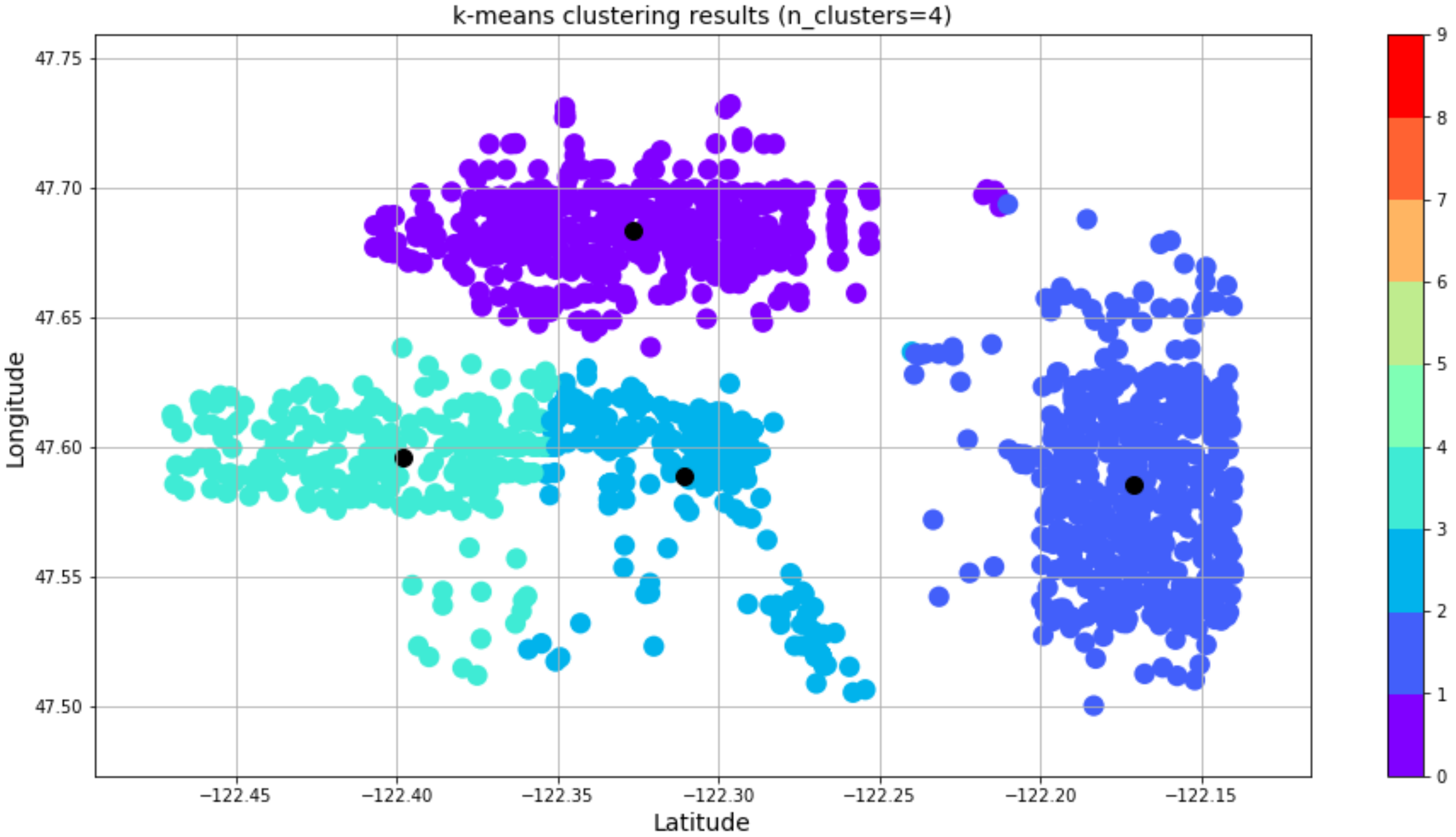
In [32]:

```
print(kmeans_1.inertia_)
# measures how spread out clusters are: lower the better
```

```
2.4637602937849774
```

In [33]:

```
fig = plt.figure(figsize=(16,8))
cmap=plt.cm.rainbow
norm = matplot.colors.BoundaryNorm(np.arange(0,10,1), cmap.N)
plt.scatter(clustered['Incident_Longitude'], clustered['Incident_Latitude'], c=c
lustered['Cluster'],
            cmap=cmap, norm=norm, s=150, edgecolor='none')
plt.colorbar(ticks=np.linspace(0,9,10))
centers = kmeans_1.cluster_centers_
plt.scatter(centers[:, 0], centers[:, 1], c='black', s=100, alpha=1);
#plt.xlim(2.28, 2.375)
#plt.ylim(48.84, 48.89)
plt.xlabel('Latitude', fontsize=14)
plt.ylabel('Longitude', fontsize=14)
plt.title('k-means clustering results (n_clusters=4)', fontsize=14)
plt.grid()
plt.show()
```



the number of correct categorizations

- [Answer 3.D]()

In [34]:

```
list_category = ["Beaver Accident","Latte Spills","Marshawn Lynch Sighting","Sea
l Attack"]

for l in list_category:
    print(l)
    b = clustered[clustered["Incident_Type"]==l]
    b_cluster = b["Cluster"]
    print(b_cluster.value_counts())
```

```
Beaver Accident
1    499
0      6
3      2
2      1
Name: Cluster, dtype: int64
Latte Spills
0    416
Name: Cluster, dtype: int64
Marshawn Lynch Sighting
2    258
0     47
3     19
Name: Cluster, dtype: int64
Seal Attack
3    240
2     18
0      5
1      3
Name: Cluster, dtype: int64
```

Wrongly classified values i.e categorised in a different cluster

- Here is the analysis

In [35]:

```
wrong = 6+2+1+0+47+19+18+5+3
```

In [36]:

```
wrong
```

Out[36]:

```
101
```

In [37]:

```
correct = len(clustered) - wrong
```

In [38]:

```
print("The Total Correctly categorised values are ",correct)
```

```
The Total Correctly categorised values are  1413
```

3.D.Answer: **The rightly classified values are: 1413**

How did I analyse this, let's take an example for the Seal Attack

In [39]:

```
ct_seal= pd.crosstab(clustered["Incident_Type"]=="Seal Attack", clustered["Clust
er"])
```

In [40]:

```
ct_seal
```

Out[40]:

Cluster	0	1	2	3
Incident_Type				
False	469	499	259	21
True	5	3	18	240

The row with true represents the distribution of the Seal Attack value in the clusters Incident Type False, means it is not a value for the Seal Attack

- It says that, 240 Seal Attack values are in cluster 3, 26 values are wrongly classified of seal attack in clusters 0,1,2

What insight can we extract from this analysis?

- Latte Spills is the cluster with least spread out distance
- Marshawn Lynch Sighting is potentially more spread out cluster (15% values in Latte and 5% as Seal Attack)
- This is also supported by the, scatter plot in 2.A, there is a overlap between red and green i.e Latte Spills and Marshawn Lynch Sighting
- To infer with the problem statement, in come cases people called out to 911 due to latte spills but maybe did not close the reason for some reason.Or this maybe a error in dataset. Cross Checking with business person is the best idea here
- Seal Attacks are also some times reported as the Marshawn Lynch Sighting

Thank You!

Neha A Varshney