In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as mp
from sklearn import *
```

In [3]:
```
TrafficData = pd.read_csv("/Users/sammonaghan/Desktop/TrafficCongestionD
ata/Traffic_Volume_Counts__2014-2018_.csv")
```

In [4]:
```
grouped = TrafficData.groupby("ID")
meanGrouped = grouped.mean()

print meanGrouped.idxmax()

print(meanGrouped.loc[138,'Segment ID'])
#Most populated street is same at 12:00-1:00 AM as 12:00-1:00 PM

##last = meanGrouped['Segment ID'].count()

##l =[]

##for i in range(1,last):
    ##l.append(i)

##for i in l:
    ##row  = i
    ##roadName = meanGrouped.loc[i, "Roadway Name"]
    ##if roadName == 'Geo Washington':
        ##print(true)
for col_name in meanGrouped:
    print col_name
```

```
Segment ID        139
12:00-1:00 AM     138
1:00-2:00AM       162
2:00-3:00AM       162
3:00-4:00AM       162
4:00-5:00AM       138
5:00-6:00AM       138
6:00-7:00AM       138
7:00-8:00AM       138
8:00-9:00AM       138
9:00-10:00AM      138
10:00-11:00AM     138
11:00-12:00PM     138
12:00-1:00PM      138
1:00-2:00PM       138
2:00-3:00PM       138
3:00-4:00PM       138
4:00-5:00PM       138
5:00-6:00PM       138
6:00-7:00PM       138
7:00-8:00PM       138
8:00-9:00PM       138
9:00-10:00PM      138
10:00-11:00PM     138
11:00-12:00AM     138
dtype: int64
134465.14285714287
Segment ID
12:00-1:00 AM
1:00-2:00AM
2:00-3:00AM
3:00-4:00AM
4:00-5:00AM
5:00-6:00AM
6:00-7:00AM
7:00-8:00AM
8:00-9:00AM
9:00-10:00AM
10:00-11:00AM
11:00-12:00PM
12:00-1:00PM
1:00-2:00PM
2:00-3:00PM
3:00-4:00PM
4:00-5:00PM
5:00-6:00PM
6:00-7:00PM
7:00-8:00PM
8:00-9:00PM
9:00-10:00PM
10:00-11:00PM
11:00-12:00AM
```

In [5]:
```
##test loc method
print(meanGrouped.loc[138,"12:00-1:00PM"])
```

```
2730.746031746032
```

In [6]:
```
##Rush hours: 8-10 am and 4-7 Pm
##I could also average the mean data between 8-10 and 4-7(or 4-6)

rushMorning1 = meanGrouped.loc[138,"8:00-9:00AM"]
rushAfter1 = meanGrouped.loc[138,"4:00-5:00PM"]

print(rushMorning1,rushAfter1)
```

```
(2650.6984126984125, 2918.84126984127)
```

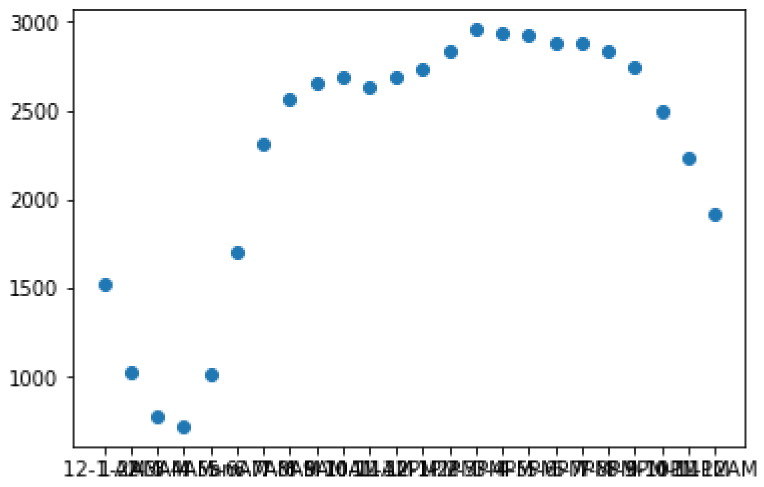
In [7]:
```
grouped2 = TrafficData.groupby('Roadway Name')
meanGrouped2 = grouped2.mean()
meanGrouped2['Roadway Name', 138]
```

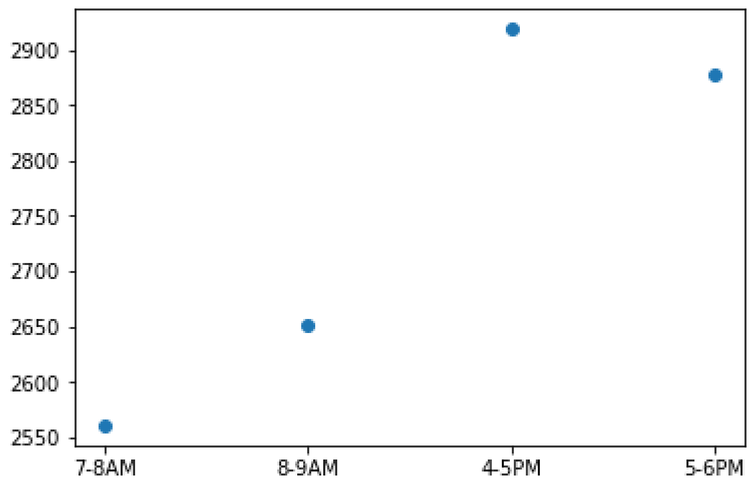
```
---------------------------------------------------------------------------
----
KeyError                                  Traceback (most recent call l
ast)
<ipython-input-7-cbd26e181133> in <module>()
      1 grouped2 = TrafficData.groupby('Roadway Name')
      2 meanGrouped2 = grouped2.mean()
----> 3 meanGrouped2['Roadway Name', 138]

/usr/local/lib/python2.7/site-packages/pandas/core/frame.pyc in __getit
em__(self, key)
   2925             if self.columns.nlevels > 1:
   2926                 return self._getitem_multilevel(key)
-> 2927             indexer = self.columns.get_loc(key)
   2928             if is_integer(indexer):
   2929                 indexer = [indexer]

/usr/local/lib/python2.7/site-packages/pandas/core/indexes/base.pyc in
get_loc(self, key, method, tolerance)
   2657                 return self._engine.get_loc(key)
   2658             except KeyError:
-> 2659                 return self._engine.get_loc(self._maybe_cast_in
dexer(key))
   2660         indexer = self.get_indexer([key], method=method, tolera
nce=tolerance)
   2661         if indexer.ndim > 1 or indexer.size > 1:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObj
ectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObj
ectHashTable.get_item()

KeyError: ('Roadway Name', 138)
```

In [8]:
```
df = pd.DataFrame(meanGrouped)
print df["RoadWay Name"]
```

```
---------------------------------------------------------------------------
----
KeyError                                  Traceback (most recent call l
ast)
<ipython-input-8-1580c33a1c9d> in <module>()
      1 df = pd.DataFrame(meanGrouped)
----> 2 print df["RoadWay Name"]

/usr/local/lib/python2.7/site-packages/pandas/core/frame.pyc in __getit
em__(self, key)
   2925             if self.columns.nlevels > 1:
   2926                 return self._getitem_multilevel(key)
-> 2927             indexer = self.columns.get_loc(key)
   2928             if is_integer(indexer):
   2929                 indexer = [indexer]

/usr/local/lib/python2.7/site-packages/pandas/core/indexes/base.pyc in
get_loc(self, key, method, tolerance)
   2657                 return self._engine.get_loc(key)
   2658             except KeyError:
-> 2659                 return self._engine.get_loc(self._maybe_cast_in
dexer(key))
   2660         indexer = self.get_indexer([key], method=method, tolera
nce=tolerance)
   2661         if indexer.ndim > 1 or indexer.size > 1:

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/index.pyx in pandas._libs.index.IndexEngine.get_loc()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObj
ectHashTable.get_item()

pandas/_libs/hashtable_class_helper.pxi in pandas._libs.hashtable.PyObj
ectHashTable.get_item()

KeyError: 'RoadWay Name'
```

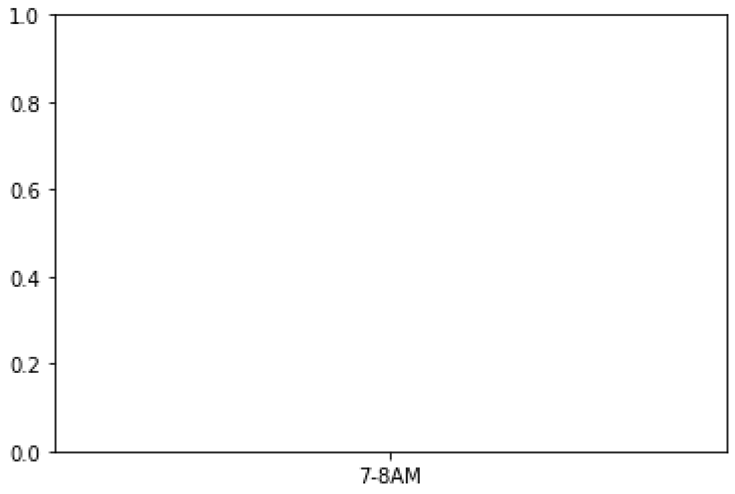
In [11]:
```
hours = ["12-1 AM", "1-2AM","2-3AM","3-4AM","4-5am","5-6AM","6-7AM","7-8
AM","8-9AM","9-10AM","10-11AM","11-12PM","12-1PM","1-2PM","2-3PM","3-4P
M","4-5PM","5-6PM","6-7PM", "7-8PM","8-9PM","9-10PM","10-11PM","11-12AM"
]
rushHours = ["7-8AM", "8-9AM", "4-5PM", "5-6PM"]
trafficHours = [meanGrouped.loc[138,"12:00-1:00 AM"], meanGrouped.loc[13
8,"1:00-2:00AM"],meanGrouped.loc[138,"2:00-3:00AM"],meanGrouped.loc[138,
"3:00-4:00AM"],meanGrouped.loc[138,"4:00-5:00AM"],meanGrouped.loc[138,
"5:00-6:00AM"],meanGrouped.loc[138,"6:00-7:00AM"],meanGrouped.loc[138,
"7:00-8:00AM"],meanGrouped.loc[138,"8:00-9:00AM"],meanGrouped.loc[138,
"9:00-10:00AM"],meanGrouped.loc[138,"10:00-11:00AM"],meanGrouped.loc[138
,"11:00-12:00PM"],meanGrouped.loc[138, "12:00-1:00PM"], meanGrouped.loc[
138,"1:00-2:00PM"],meanGrouped.loc[138,"2:00-3:00PM"],meanGrouped.loc[13
8,"3:00-4:00PM"],meanGrouped.loc[138,"4:00-5:00PM"],meanGrouped.loc[138,
"5:00-6:00PM"],meanGrouped.loc[138,"6:00-7:00PM"],meanGrouped.loc[138,
"7:00-8:00PM"],meanGrouped.loc[138,"8:00-9:00PM"],meanGrouped.loc[138,
"9:00-10:00PM"],meanGrouped.loc[138,"10:00-11:00PM"],meanGrouped.loc[138
,"11:00-12:00AM"]]
rushTrafficHours = [meanGrouped.loc[138,"7:00-8:00AM"],meanGrouped.loc[1
38,"8:00-9:00AM"],meanGrouped.loc[138,"4:00-5:00PM"],meanGrouped.loc[138
,"5:00-6:00PM"]]

plt.scatter(hours, trafficHours)
plt.show()
plt.scatter(rushHours, rushTrafficHours)
plt.show()

##Histogram
plt.hist(rushHours, bins = rushTrafficHours)
plt.show()
```


3000
2500
2000
1500
1000


2900
2850
2800
2750
2700
2650
2600
2550
7-8AM
8-9AM
4-5PM
5-6PM

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call l
ast)
<ipython-input-11-46ea6f67a3fc> in <module>()
     10 
     11 ##Histogram
---> 12 plt.hist(rushHours, bins = rushTrafficHours)
     13 plt.show()

/usr/local/lib/python2.7/site-packages/matplotlib/pyplot.pyc in hist(x,
bins, range, density, weights, cumulative, bottom, histtype, align, ori
entation, rwidth, log, color, label, stacked, normed, hold, data, **kwa
rgs)
   3135                       histtype=histtype, align=align, orientati
on=orientation,
   3136                       rwidth=rwidth, log=log, color=color, labe
l=label,
-> 3137                       stacked=stacked, normed=normed, data=dat
a, **kwargs)
   3138     finally:
   3139         ax._hold = washold

/usr/local/lib/python2.7/site-packages/matplotlib/__init__.pyc in inner
(ax, *args, **kwargs)
   1865                         "the Matplotlib list!)" % (label_namer,
func.__name__),
   1866                         RuntimeWarning, stacklevel=2)
-> 1867             return func(ax, *args, **kwargs)
   1868 
   1869         inner.__doc__ = _add_data_doc(inner.__doc__,

/usr/local/lib/python2.7/site-packages/matplotlib/axes/_axes.pyc in his
t(***failed resolving arguments***)
   6637             # this will automatically overwrite bins,
   6638             # so that each histogram uses the same bins
-> 6639             m, bins = np.histogram(x[i], bins, weights=w[i], **
hist_kwargs)
   6640             m = m.astype(float)  # causes problems later if i
t's an int
   6641             if mlast is None:

/usr/local/lib/python2.7/site-packages/numpy/lib/histograms.pyc in hist
ogram(a, bins, range, normed, weights, density)
    788     a, weights = _ravel_and_check_weights(a, weights)
    789 
--> 790     bin_edges, uniform_bins = _get_bin_edges(a, bins, range, we
ights)
    791 
    792     # Histogram is an integer or a float array depending on the
weights.

/usr/local/lib/python2.7/site-packages/numpy/lib/histograms.pyc in _get
_bin_edges(a, bins, range, weights)
    431         if np.any(bin_edges[:-1] > bin_edges[1:]):
    432             raise ValueError(
--> 433                 '`bins` must increase monotonically, when an ar
```

```
ray')
    434
    435     else:

ValueError: `bins` must increase monotonically, when an array
```



In [ ]:
```
from sklearn import datasets, linear_model

regr = linear_model.LinearRegression()

regr.fit(rushHours,rushTrafficHours)
```

In [ ]: