bicycle_relation_Poisson_reg_INCOMPLETE

April 24, 2018

In [2]:
```
import pandas as pd
import numpy as np
```

In [3]:
```
df = pd.read_csv('nyc.csv', parse_dates=['Date'])
df.head()
```

Out[3]:

	Unnamed: 0	Date	Day	High Temp (řF)	Low Temp (řF)	\
0	0	2016-04-01	2016-04-01 00:00:00	78.1	66.0	
1	1	2016-04-02	2016-04-02 00:00:00	55.0	48.9	
2	2	2016-04-03	2016-04-03 00:00:00	39.9	34.0	
3	3	2016-04-04	2016-04-04 00:00:00	44.1	33.1	
4	4	2016-04-05	2016-04-05 00:00:00	42.1	26.1	

	Precipitation	Brooklyn Bridge	Manhattan Bridge	Williamsburg Bridge	\
0	0.01	1704.0	3126	4115.0	
1	0.15	827.0	1646	2565.0	
2	0.09	526.0	1232	1695.0	
3	0.47 (S)	521.0	1067	1440.0	
4	0	1416.0	2617	3081.0	

	Queensboro Bridge	Total
0	2552.0	11497
1	1884.0	6922
2	1306.0	4759
3	1307.0	4335
4	2357.0	9471

In [4]:
```
df.isnull().any()
```

Out[4]:
```
Unnamed: 0             False
Date                   False
Day                    False
High Temp (řF)         False
Low Temp (řF)          False
Precipitation          False
Brooklyn Bridge        False
Manhattan Bridge       False
Williamsburg Bridge    False
```

```
Queensboro Bridge      False
Total                  False
dtype: bool
```

```
In [5]: day = pd.to_datetime(df['Date'])
        brooklyn = df['Brooklyn Bridge']
        manhattan = df['Manhattan Bridge']
        william = df['Williamsburg Bridge']
        queen = df['Queensboro Bridge']
        high_temp = df['High Temp (řF)']
        low_temp = df['Low Temp (řF)']
        total = df['Total']
        d = []
        for i in day:
            d.append(str(i)[8:10])
        day = pd.DataFrame(d,columns=['Day'])
        day.head()
```

```
Out[5]:   Day
        0  01
        1  02
        2  03
        3  04
        4  05
```

```
In [10]: print('Avarage byc. passes from  manhattan', manhattan.mean(),'in ',len(day),'days')
```

```
('Avarage byc. passes from  manhattan', 4049.5333333333333, 'in ', 210, 'days')
```

Temperature correlation with bycle load on bridge

```
In [89]: high_temp.corr(total), low_temp.corr(total)
```

```
Out[89]: (0.7433437005618655, 0.4921241940792782)
```

```
In [131]: (high_temp.corr(total) + low_temp.corr(total)) /2
```

```
Out[131]: 0.6177339473205719
```

```
In [161]: avg = {}
          avg['Brooklyn']= np.average(brooklyn)
          avg['Manhattan']= np.average(manhattan)
          avg['Williamsburg']= np.average(william)
          avg['Queensboro']= np.average(queen)
```

```
In [148]: print('Max: %r , Min: %r Average: %.8r' %(max(brooklyn),min(brooklyn),avg['Brooklyn']))
```

```
Max: 3871.0 , Min: 504.0 Average: 2269.633
```

```
In [149]: print('Max: %r , Min: %r Average: %.8r' %(max(manhattan),min(manhattan),avg['Manhattan
```

```
Max: 6951 , Min: 997 Average: 4049.533
```

```
In [150]: print('Max: %r , Min: %r Average: %.8r' %(max(william),min(william),avg['Williamsburg'
```

```
Max: 7834.0 , Min: 1440.0 Average: 4862.466
```

```
In [152]: print('Max: %r , Min: %r Average: %.8r' %(max(queen),min(queen),avg['Queensboro']))
```

```
Max: 5032.0 , Min: 1306.0 Average: 3352.866
```

```
In [173]: for key, value in sorted(avg.iteritems(), key=lambda (k,v): (v,k),reverse=True):
              print "Bicycle load on %s Bridge : %.8s" % (key, value)
```

```
Bicycle load on Williamsburg Bridge : 4862.466
Bicycle load on Manhattan Bridge : 4049.533
Bicycle load on Queensboro Bridge : 3352.866
Bicycle load on Brooklyn Bridge : 2269.633
```