customer service request analysis python project

June 9, 2021

[1]:
```
#import library
import pandas as pd
import numpy as np
import matplotlib as mpl
from matplotlib import pyplot as plt
%matplotlib inline
plt.style.use(['fivethirtyeight'])
mpl.rcParams['lines.linewidth'] = 3
import warnings
warnings.filterwarnings("ignore")
import scipy.stats as stats
import statsmodels.api as sm
from statsmodels.formula.api import ols
```

[2]:
```
# read data set 311 NYC service request.
df = pd.read_csv('311_Service_Requests_from_2010_to_Present.csv', header=0,
    sep=',', parse_dates=['Created Date', 'Closed Date', 'Resolution Action␣
 ↪Updated Date'],index_col='Unique Key')
```

[3]:
```
#Request_Closing_Time
def prepareData(df):
    df['Resolution_Time'] = (df['Closed Date'] - df['Created Date']).dt.
 ↪total_seconds() ####days/3600
    df_clean=df[df['Resolution_Time'].notnull()]
    df_perfect = df_clean[df_clean['Closed Date'] >= df_clean['Created Date']]
    df_perfect['Day of Week'] = df_perfect['Created Date'].dt.dayofweek
    df_perfect['Day of Month'] = df_perfect['Created Date'].dt.day
    df_perfect['Month'] = df_perfect['Created Date'].dt.month
    df_perfect['Year'] = df_perfect['Created Date'].dt.year
    df_perfect=df_perfect[df_perfect.Borough!='Unspecified']
    return df_perfect
```

[4]:
```
#shape
df_perfect = prepareData(df)
df_perfect.shape
```

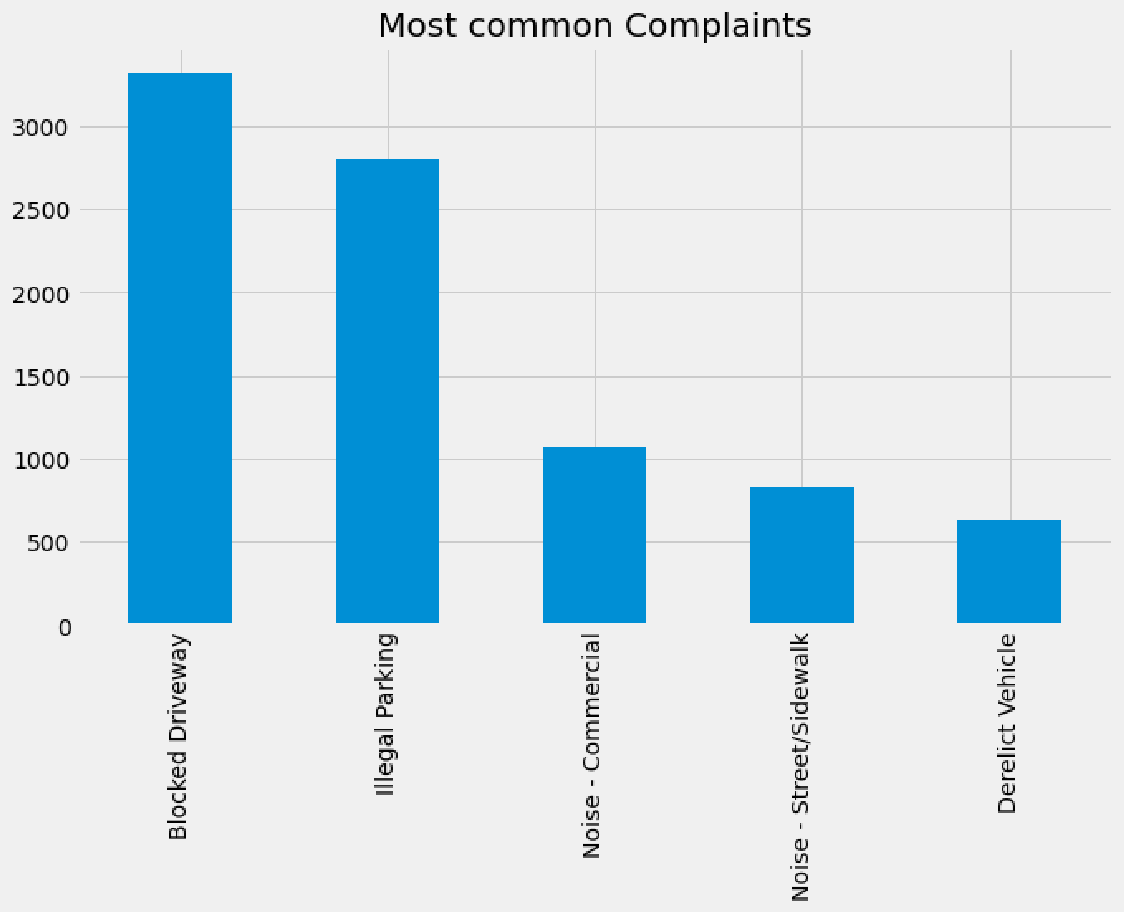
[4]:
```
(9859, 57)
```

[5]:

```
#Major Insights Patterns
#Most frequent Complaints
(df_perfect['Complaint Type'].value_counts()).head().plot(kind='bar',
                    figsize=(10,6), title = 'Most common Complaints')
```

[5]:

```
<AxesSubplot:title={'center':'Most common Complaints'}>
```

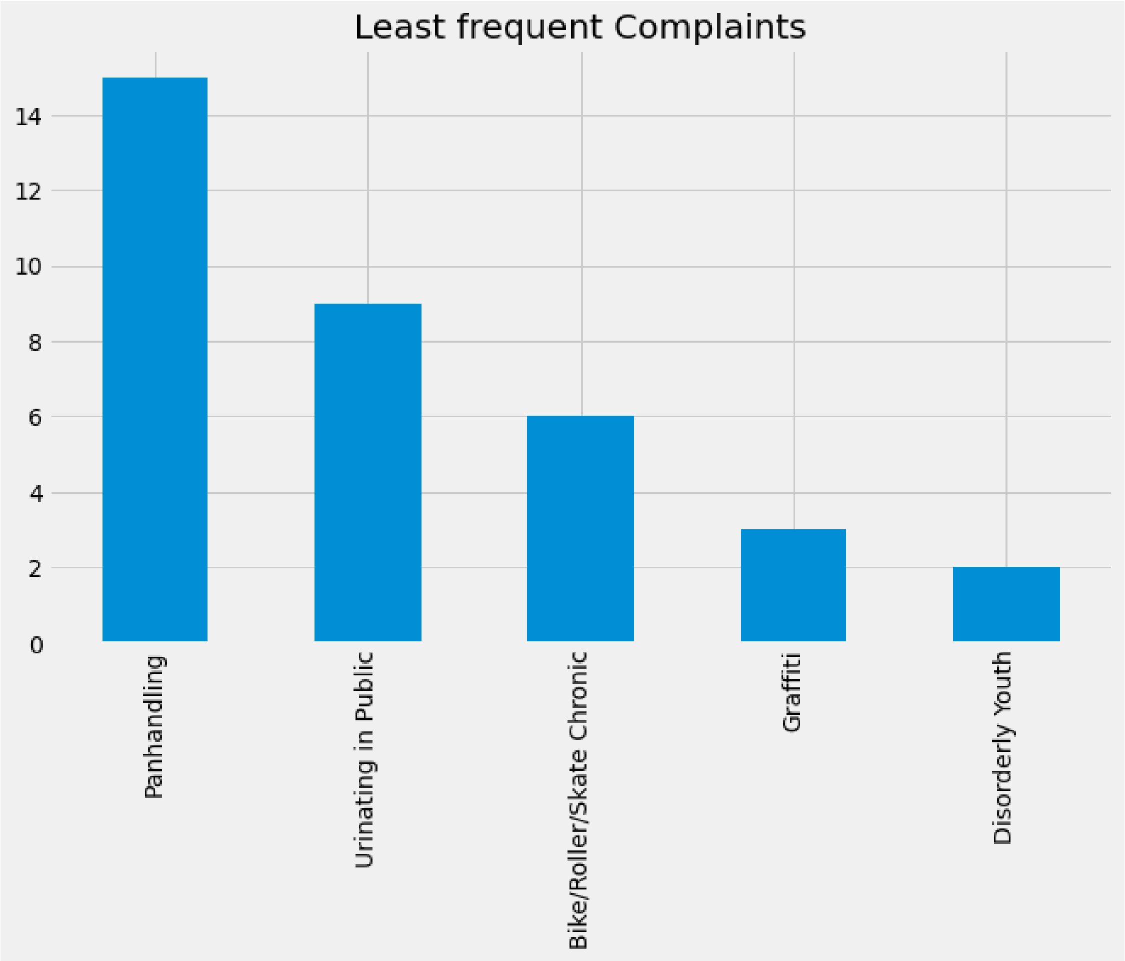


[6]:

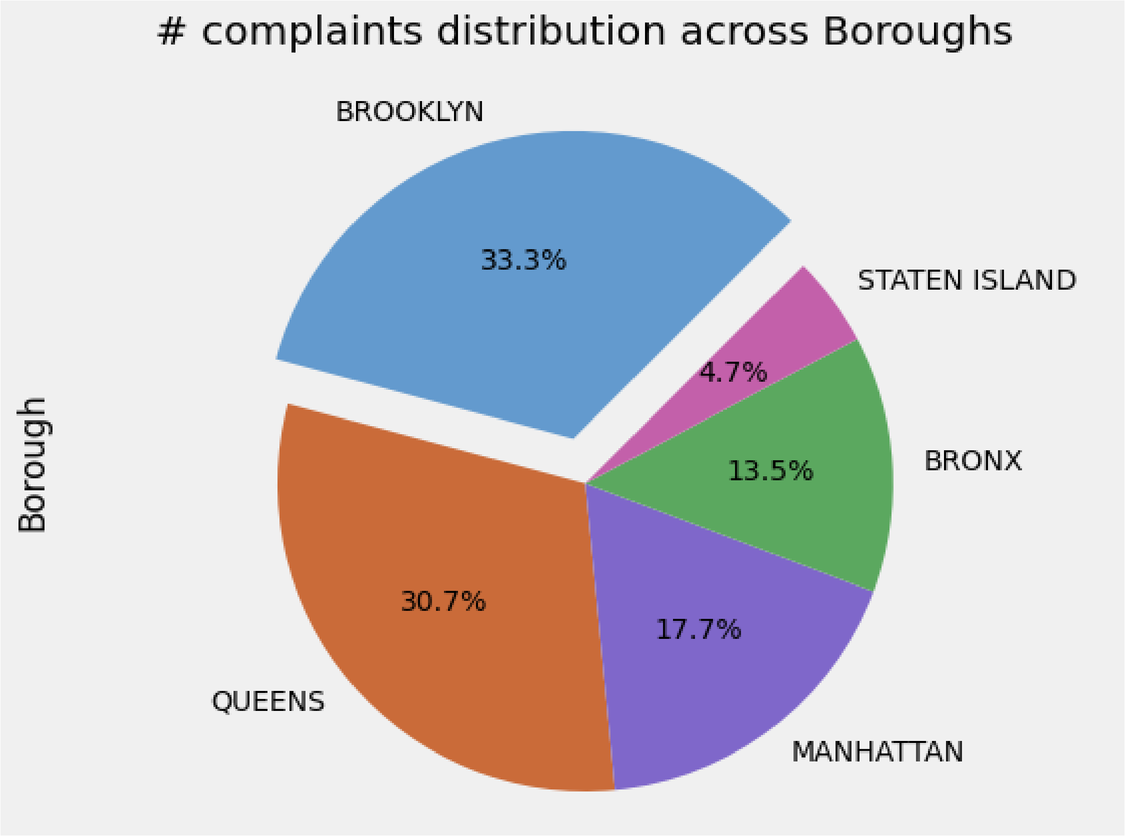
```
#Least frequent Complaints
(df_perfect['Complaint Type'].value_counts()).tail().plot(kind='bar',
                    figsize=(10,6), title = 'Least frequent Complaints')
```

[6]:

```
<AxesSubplot:title={'center':'Least frequent Complaints'}>
```



```
[7]: # complaints distribution across Boroughs
     colors = ['#639ace','#ca6b39','#7f67ca','#5ba85f','#c360aa','#a7993f','#cc566a']
     df_perfect['Borough'].value_counts().plot(kind='pie',autopct='%1.1f%%',
                                 explode = (0.15, 0, 0, 0, 0), startangle=45,␣
      ↪shadow=False, colors = colors,
                                 figsize = (8,6))
     #plt.legend(title='BOROUGH', loc='upper right', bbox_to_anchor=(1.5,1))
     plt.axis('equal')
     plt.title('# complaints distribution across Boroughs\n')
     plt.tight_layout()
     plt.show()
```



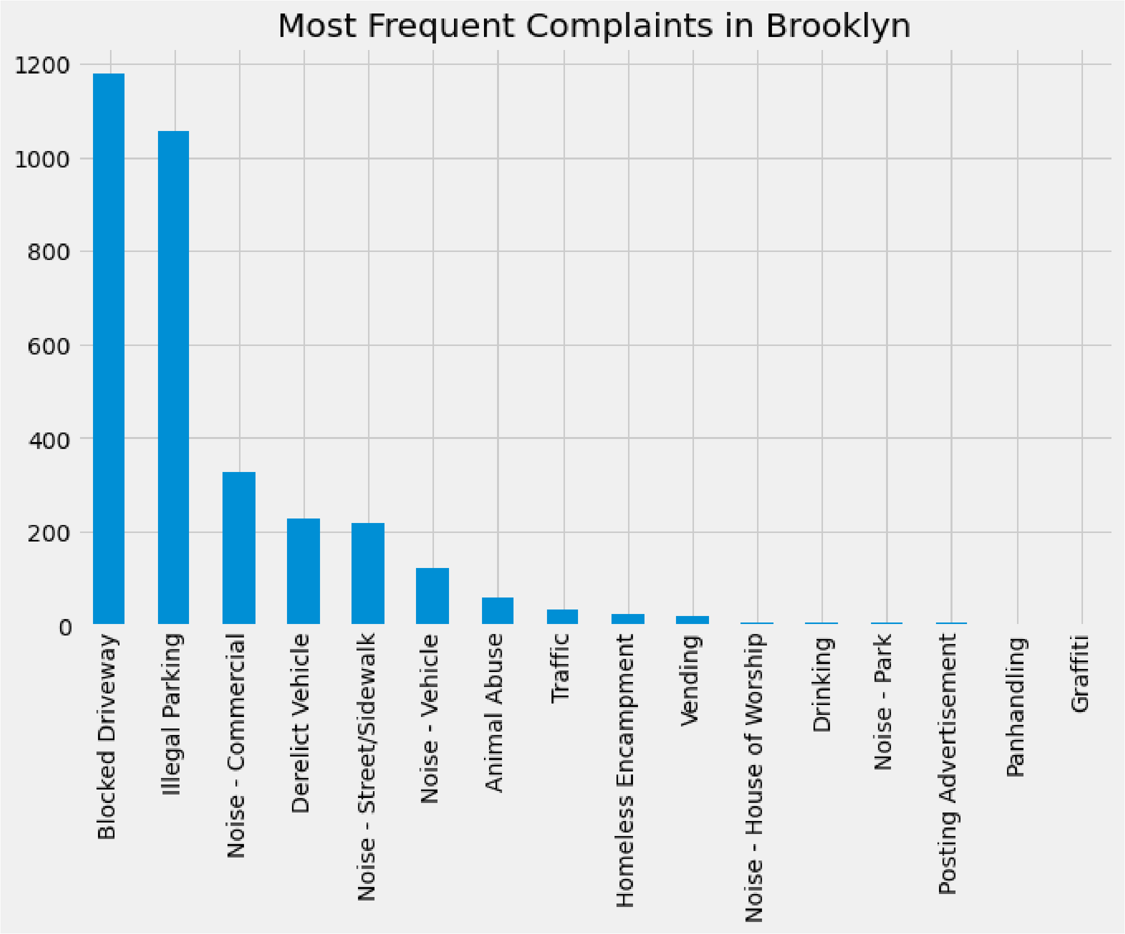
[8]:
```
#Analysis for Brooklyn borough which has highest number of complains
df_Brooklyn = df_perfect[df_perfect['Borough']=='BROOKLYN']
```

[9]:
```
#shape
df_Brooklyn.shape
```

[9]:
```
(3287, 57)
```

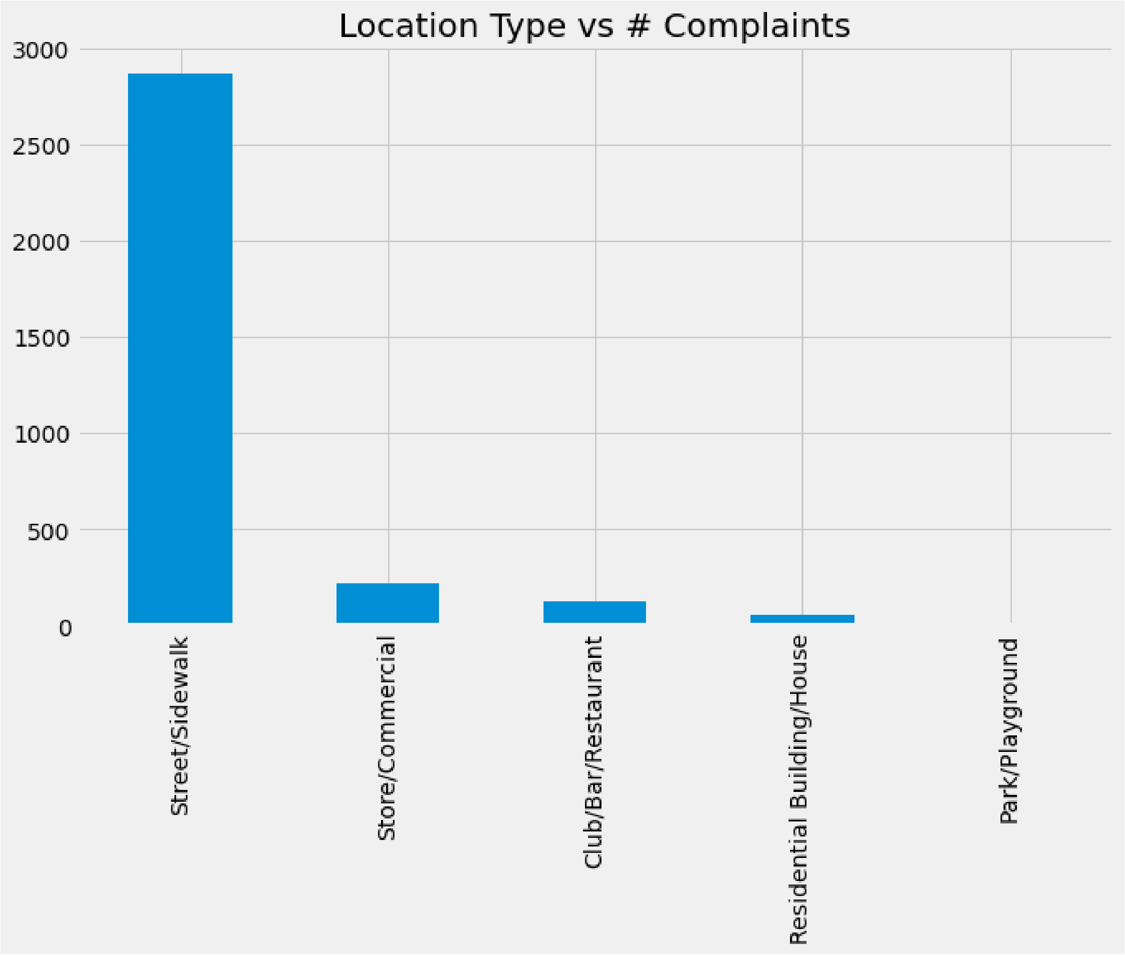
[10]:
```
(df_Brooklyn['Complaint Type'].value_counts()).head(25).plot(kind='bar',
                                                    figsize=(10,6),title =␣
 ↪'Most Frequent Complaints in Brooklyn')
```

[10]:
```
<AxesSubplot:title={'center':'Most Frequent Complaints in Brooklyn'}>
```



```
[11]: #location type vs complaints
      (df_Brooklyn['Location Type'].value_counts()).head().plot(kind='bar',
                                                      figsize=(10,6),title =␣
       ↪'Location Type vs # Complaints')
```

```
[11]: <AxesSubplot:title={'center':'Location Type vs # Complaints'}>
```



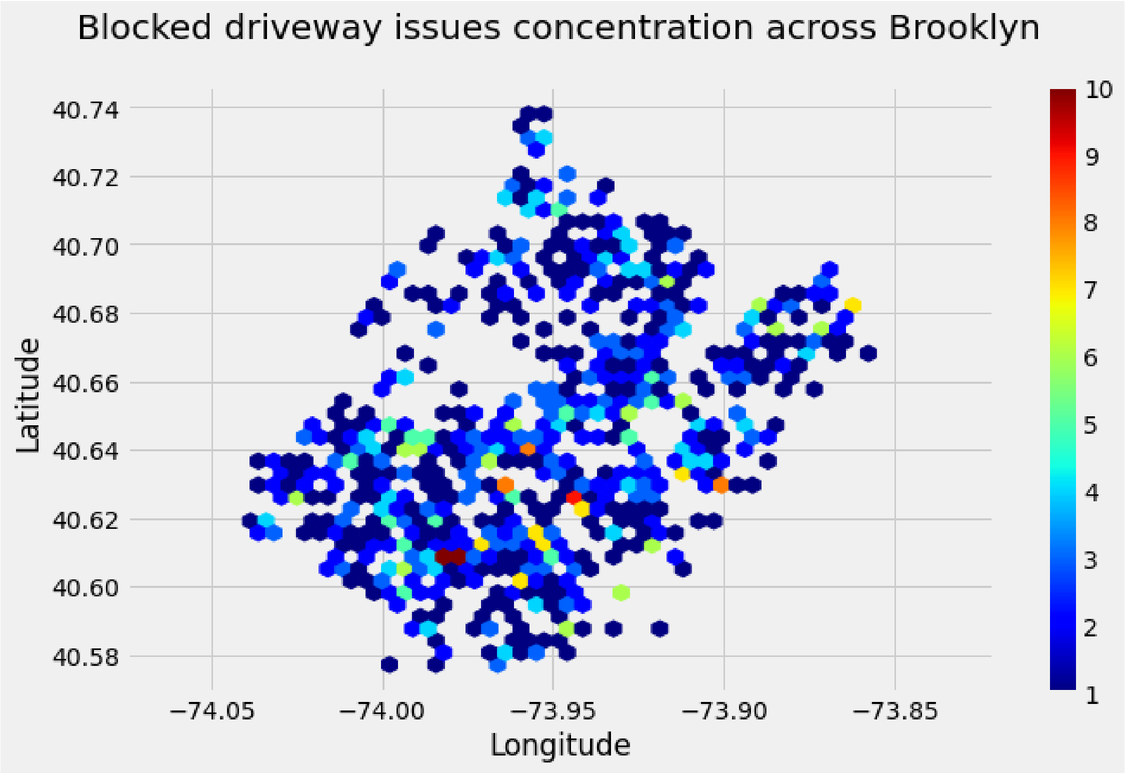
```
[13]: #Anlysis of Most Frequent complaint in Brooklyn
      df_perfect[df_perfect['Complaint Type'] == 'Blocked Driveway']['Descriptor'].
       ↪value_counts()
```

```
[13]: No Access        2519
      Partial Access    798
      Name: Descriptor, dtype: int64
```

```
[14]: df_Brook_blocked = df_Brooklyn[df_Brooklyn['Complaint Type'] == 'Blocked␣
       ↪Driveway']
```

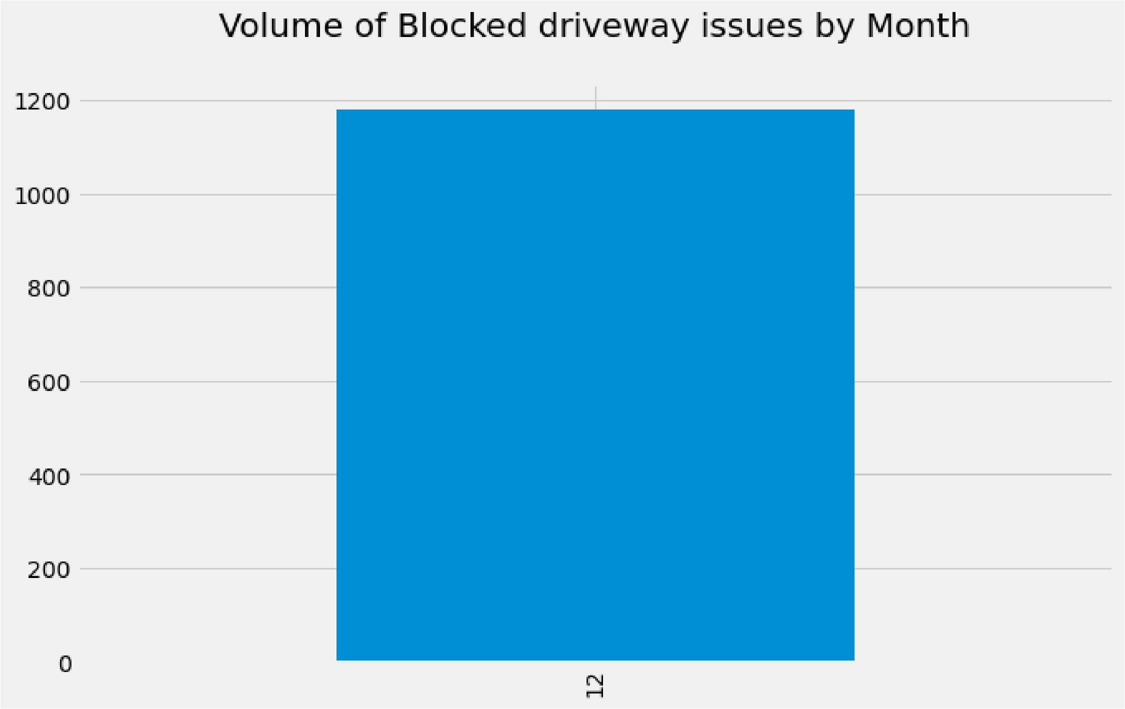
```
[15]: df_Brook_blocked.plot(
          kind='hexbin', x='Longitude', y='Latitude', gridsize=40,title = 'Blocked␣
       ↪driveway issues concentration across Brooklyn\n',
          colormap='jet', mincnt=1, figsize=(10,6)).axis('equal')
```

```
[15]: (-74.04753093769894, -73.84859863230106, 40.5689863275, 40.746125542499996)
```



```
[16]: df_Brook_blocked['Month'].value_counts().plot(kind = 'bar',figsize=(10,6),␣
       ↪title = 'Volume of Blocked driveway issues by Month\n')
```

```
[16]: <AxesSubplot:title={'center':'Volume of Blocked driveway issues by Month\n'}>
```



```
[17]: df_avg_res_time_city = df_perfect.groupby(['City','Complaint Type']).
       ↪Resolution_Time.mean()
      #df_perfect.sort_values('Complaint Type').groupby('City')
      #
      df_avg_res_time_city.head(25)
```

```
[17]: City       Complaint Type
      ARVERNE    Animal Abuse                  5782.500000
                 Illegal Parking              14391.500000
                 Noise - Commercial            5437.000000
      ASTORIA    Animal Abuse                  6719.400000
                 Bike/Roller/Skate Chronic     5101.000000
                 Blocked Driveway             14975.940594
                 Derelict Vehicle             15840.200000
                 Drinking                      6425.000000
                 Homeless Encampment          18789.000000
                 Illegal Parking              13755.479167
                 Noise - Commercial            7852.344828
                 Noise - Street/Sidewalk       9775.666667
                 Noise - Vehicle               8426.777778
      BAYSIDE    Blocked Driveway             10679.000000
                 Derelict Vehicle              7374.600000
                 Illegal Parking               9052.714286
                 Noise - Street/Sidewalk       2031.000000
```

```
           Noise - Vehicle                7054.000000
BELLEROSE  Blocked Driveway               3763.250000
           Derelict Vehicle              55215.000000
           Noise - Commercial             3309.000000
           Noise - Street/Sidewalk       58141.000000
BRONX      Animal Abuse                  45139.710526
           Bike/Roller/Skate Chronic     18530.500000
           Blocked Driveway              27582.763912
Name: Resolution_Time, dtype: float64
```

```
[18]: #Average response time in seconds across complaint types
      df_avg_res_time = df_perfect.groupby('Complaint Type').Resolution_Time.mean().
       ↪sort_values(ascending=True)
      df_avg_res_time.head(21)
```

```
[18]: Complaint Type
      Disorderly Youth              2659.000000
      Posting Advertisement         4817.935484
      Noise - House of Worship      6178.588235
      Urinating in Public           8637.000000
      Bike/Roller/Skate Chronic    10423.000000
      Vending                      10961.470588
      Noise - Commercial           11826.484565
      Noise - Street/Sidewalk      12947.630695
      Panhandling                  13098.733333
      Noise - Vehicle              14086.066838
      Traffic                      14492.788991
      Graffiti                     14967.666667
      Homeless Encampment          15202.952055
      Illegal Parking              15373.488929
      Blocked Driveway             17532.457642
      Animal Abuse                 19210.459350
      Noise - Park                 19496.656250
      Drinking                     23259.475000
      Derelict Vehicle             24489.738437
      Name: Resolution_Time, dtype: float64
```

```
[19]: df_dis_youth = df_perfect[df_perfect['Complaint Type']=='Disorderly Youth']
      df_dis_youth = df_dis_youth.loc[:,['Resolution_Time']]
      df_dis_youth.head()
      #df_dis_youth.columns
      #df_avg_res_time  = df_avg_res_time.to_frame()
      #df_avg_res_time.columns()
      #df_dis_youth.Complaint Type.unique()
```

```
[19]:             Resolution_Time
      Unique Key
```

```
32274507             713.0
32244468            4605.0
```

[20]:

```
df_noise_veh = df_perfect[df_perfect['Complaint Type']=='Noise - Vehicle']
df_noise_veh = df_noise_veh.loc[:,['Resolution_Time']]
df_noise_veh.head()
#df_noise_veh.columns
#df_noise_veh.info()
```

[20]:

```
            Resolution_Time
Unique Key
32307159            22949.0
32308722             7254.0
32308107            11319.0
32308108            10937.0
32306622             2615.0
```

[21]:

```
df_type_res = df_perfect.loc[:, ['Complaint Type','Resolution_Time']]
df_type_res.head()
df_type_res.columns
```

[21]:

```
Index(['Complaint Type', 'Resolution_Time'], dtype='object')
```

[22]:

```
# stats f_oneway functions takes the groups as input and returns F and P-value
fvalue, pvalue = stats.f_oneway(df_dis_youth, df_noise_veh)
pvalue
```

[22]:

```
array([0.45036375])
```

[23]:

```
df_post_ad = df_perfect[df_perfect['Complaint Type']=='Posting Advertisement']
df_post_ad = df_post_ad.loc[:,['Resolution_Time']]
df_post_ad.head()
```

[23]:

```
            Resolution_Time
Unique Key
32306752             7596.0
32307464             7745.0
32308949             7834.0
32307323             8042.0
32306034             8137.0
```

[24]:

```
df_der_veh = df_perfect[df_perfect['Complaint Type']=='Derelict Vehicle']
df_der_veh = df_der_veh.loc[:,['Resolution_Time']]
df_der_veh.head()
```

[24]:

```
            Resolution_Time
Unique Key
```

```
32309424             37763.0
32306497             14221.0
32305124              4913.0
32308002             14879.0
32305798              2712.0
```

[25]:

```
# stats f_oneway functions takes the groups as input and returns F and P-value
fvalue, pvalue = stats.f_oneway(df_post_ad, df_der_veh)
pvalue
```

[25]:

```
array([2.02322069e-05])
```

[26]:

```
# get ANOVA table for complain type and resolution time

# reshape the d dataframe suitable for statsmodels package
df_perfect['Complaint_Type']=df_perfect['Complaint Type']
df_type_res = df_perfect.loc[:, ['Complaint_Type','Resolution_Time']] ␣
 ↪#Complaint Type
# Ordinary Least Squares (OLS) model
model = ols('Resolution_Time ~ Complaint_Type', data=df_type_res).fit()
anova_table = sm.stats.anova_lm(model, typ=2)
anova_table
```

[26]:

```
                      sum_sq      df          F        PR(>F)
Complaint_Type  1.012747e+11    18.0  10.934509  9.822678e-32
Residual        5.063188e+12  9840.0        NaN           NaN
```

[27]:

```
df_city_type = pd.crosstab(df_perfect.City , df_perfect.Complaint_Type)
```

[29]:

```
# chi-squared test with similar proportions
from scipy.stats import chi2_contingency
from scipy.stats import chi2
# contingency table
table = df_city_type
#print(table)
stat, p, dof, expected = chi2_contingency(table)
print('dof=%d' % dof)
print(expected)
# interpret test-statistic
prob = 0.95
critical = chi2.ppf(prob, dof)
print('probability=%.3f, critical=%.3f, stat=%.3f' % (prob, critical, stat))
if abs(stat) >= critical:
    print('Dependent (reject H0)')
else:
    print('Independent (fail to reject H0)')
# interpret p-value
```

```python
alpha = 1.0 - prob
print('significance=%.3f, p=%.3f' % (alpha, p))
if p <= alpha:
    print('Dependent (reject H0)')
else:
    print('Independent (fail to reject H0)')
```

```
dof=774
[[1.24759103e-01 3.04290496e-03 1.68221929e+00 3.17983568e-01
  1.01430165e-03 2.02860331e-02 1.52145248e-03 7.40440207e-02
  1.42002231e+00 5.42144234e-01 8.62156405e-03 1.62288265e-02
  4.22963789e-01 1.97281672e-01 7.60726240e-03 3.14433513e-02
  5.52794401e-02 4.56435744e-03 6.89725124e-02]
 [5.33968962e+00 1.30236332e-01 7.19989857e+01 1.36096967e+01
  4.34121108e-02 8.68242215e-01 6.51181661e-02 3.16908409e+00
  6.07769551e+01 2.32037732e+01 3.69002941e-01 6.94593772e-01
  1.81028502e+01 8.44365554e+00 3.25590831e-01 1.34577543e+00
  2.36596004e+00 1.95354498e-01 2.95202353e+00]
 [7.73506441e-01 1.88660108e-02 1.04297596e+01 1.97149812e+00
  6.28867025e-03 1.25773405e-01 9.43300538e-03 4.59072928e-01
  8.80413835e+00 3.36129425e+00 5.34536971e-02 1.00618724e-01
  2.62237549e+00 1.22314636e+00 4.71650269e-02 1.94948778e-01
  3.42732529e-01 2.82990161e-02 4.27629577e-01]
 [2.24566386e-01 5.47722893e-03 3.02799473e+00 5.72370423e-01
  1.82574298e-03 3.65148595e-02 2.73861446e-03 1.33279237e-01
  2.55604017e+00 9.75859621e-01 1.55188153e-02 2.92118876e-02
  7.61334821e-01 3.55107009e-01 1.36930723e-02 5.65980323e-02
  9.95029922e-02 8.21584339e-03 1.24150522e-01]
 [3.31609697e+01 8.08804138e-01 4.47133888e+02 8.45200325e+01
  2.69601379e-01 5.39202759e+00 4.04402069e-01 1.96809007e+01
  3.77441931e+02 1.44101937e+02 2.29161173e+00 4.31362207e+00
  1.12423775e+02 5.24374683e+01 2.02201035e+00 8.35764276e+00
  1.46932752e+01 1.21320621e+00 1.83328938e+01]
 [8.20166345e+01 2.00040572e+00 1.10589096e+03 2.09042398e+02
  6.66801907e-01 1.33360381e+01 1.00020286e+00 4.86765392e+01
  9.33522670e+02 3.56405619e+02 5.66781621e+00 1.06688305e+01
  2.78056395e+02 1.29692971e+02 5.00101430e+00 2.06708591e+01
  3.63407039e+01 3.00060858e+00 4.53425297e+01]
 [1.99614565e-01 4.86864794e-03 2.69155087e+00 5.08773709e-01
  1.62288265e-03 3.24576529e-02 2.43432397e-03 1.18470433e-01
  2.27203570e+00 8.67430774e-01 1.37945025e-02 2.59661223e-02
  6.76742063e-01 3.15650675e-01 1.21716198e-02 5.03093620e-02
  8.84471042e-02 7.30297190e-03 1.10356020e-01]
 [9.23217365e-01 2.25174967e-02 1.24484228e+01 2.35307841e+00
  7.50583223e-03 1.50116645e-01 1.12587484e-02 5.47925753e-01
  1.05081651e+01 4.01186733e+00 6.37995740e-02 1.20093316e-01
  3.12993204e+00 1.45988437e+00 5.62937418e-02 2.32680799e-01
 4.09067857e-01 3.37762451e-02 5.10396592e-01]
[4.54123136e+00 1.10761741e-01 6.12327822e+01 1.15746019e+01
 3.69205802e-02 7.38411604e-01 5.53808703e-02 2.69520235e+00
 5.16888123e+01 1.97340501e+01 3.13824932e-01 5.90729283e-01
 1.53958819e+01 7.18105285e+00 2.76904351e-01 1.14453799e+00
 2.01217162e+00 1.66142611e-01 2.51059945e+00]
[2.44527843e+00 5.96409372e-02 3.29714981e+01 6.23247794e+00
 1.98803124e-02 3.97606248e-01 2.98204686e-02 1.45126281e+00
 2.78324374e+01 1.06260270e+01 1.68982655e-01 3.18084998e-01
 8.29009027e+00 3.86672076e+00 1.49102343e-01 6.16289685e-01
 1.08347703e+00 8.94614058e-02 1.35186124e+00]
[3.31859215e+00 8.09412719e-02 4.47470332e+01 8.45836292e+00
 2.69804240e-02 5.39608480e-01 4.04706360e-02 1.96957095e+00
 3.77725936e+01 1.44210366e+01 2.29333604e-01 4.31686784e-01
 1.12508368e+01 5.24769246e+00 2.02353180e-01 8.36393143e-01
 1.47043311e+00 1.21411908e-01 1.83466883e+00]
[7.73506441e-01 1.88660108e-02 1.04297596e+01 1.97149812e+00
 6.28867025e-03 1.25773405e-01 9.43300538e-03 4.59072928e-01
 8.80413835e+00 3.36129425e+00 5.34536971e-02 1.00618724e-01
 2.62237549e+00 1.22314636e+00 4.71650269e-02 1.94948778e-01
 3.42732529e-01 2.82990161e-02 4.27629577e-01]
[4.99036413e-02 1.21716198e-03 6.72887717e-01 1.27193427e-01
 4.05720661e-04 8.11441323e-03 6.08580992e-04 2.96176083e-02
 5.68008926e-01 2.16857693e-01 3.44862562e-03 6.49153058e-03
 1.69185516e-01 7.89126686e-02 3.04290496e-03 1.25773405e-02
 2.21117760e-02 1.82574298e-03 2.75890050e-02]
[6.56232884e+00 1.60056801e-01 8.84847348e+01 1.67259357e+01
 5.33522670e-02 1.06704534e+00 8.00284004e-02 3.89471549e+00
 7.46931737e+01 2.85167867e+01 4.53494269e-01 8.53636271e-01
 2.22478953e+01 1.03770159e+01 4.00142002e-01 1.65392028e+00
 2.90769855e+00 2.40085201e-01 3.62795415e+00]
[1.74662745e+00 4.26006694e-02 2.35510701e+01 4.45176996e+00
 1.42002231e-02 2.84004463e-01 2.13003347e-02 1.03661629e+00
 1.98803124e+01 7.59001927e+00 1.20701897e-01 2.27203570e-01
 5.92149305e+00 2.76194340e+00 1.06501674e-01 4.40206918e-01
 7.73912161e-01 6.39010042e-02 9.65615174e-01]
[3.01917030e+00 7.36383000e-02 4.07097069e+01 7.69520235e+00
 2.45461000e-02 4.90922000e-01 3.68191500e-02 1.79186530e+00
 3.43645400e+01 1.31198905e+01 2.08641850e-01 3.92737600e-01
 1.02357237e+01 4.77421645e+00 1.84095750e-01 7.60929100e-01
 1.33776245e+00 1.10457450e-01 1.66913480e+00]
[2.24566386e-01 5.47722893e-03 3.02799473e+00 5.72370423e-01
 1.82574298e-03 3.65148595e-02 2.73861446e-03 1.33279237e-01
 2.55604017e+00 9.75859621e-01 1.55188153e-02 2.92118876e-02
 7.61334821e-01 3.55107009e-01 1.36930723e-02 5.65980323e-02
 9.95029922e-02 8.21584339e-03 1.24150522e-01]
[9.98072827e-01 2.43432397e-02 1.34577543e+01 2.54386855e+00
 8.11441323e-03 1.62288265e-01 1.21716198e-02 5.92352166e-01
 1.13601785e+01 4.33715387e+00 6.89725124e-02 1.29830612e-01
 3.38371032e+00 1.57825337e+00 6.08580992e-02 2.51546810e-01
 4.42235521e-01 3.65148595e-02 5.51780099e-01]
[9.73121006e-01 2.37346587e-02 1.31213105e+01 2.48027183e+00
 7.91155290e-03 1.58231058e-01 1.18673293e-02 5.77543361e-01
 1.10761741e+01 4.22872502e+00 6.72481996e-02 1.26584846e-01
 3.29911756e+00 1.53879704e+00 5.93366467e-02 2.45258140e-01
 4.31179633e-01 3.56019880e-02 5.37985597e-01]
[1.52206106e+00 3.71234405e-02 2.05230754e+01 3.87939953e+00
 1.23744802e-02 2.47489603e-01 1.85617203e-02 9.03337052e-01
 1.73242722e+01 6.61415965e+00 1.05183081e-01 1.97991683e-01
 5.16015823e+00 2.40683639e+00 9.28086013e-02 3.83608885e-01
 6.74409169e-01 5.56851608e-02 8.41464652e-01]
[6.93660615e+00 1.69185516e-01 9.35313926e+01 1.76798864e+01
 5.63951719e-02 1.12790344e+00 8.45927579e-02 4.11684755e+00
 7.89532407e+01 3.01432194e+01 4.79358961e-01 9.02322751e-01
 2.35167867e+01 1.09688609e+01 4.22963789e-01 1.74825033e+00
 3.07353687e+00 2.53778274e-01 3.83487169e+00]
[5.98843696e-01 1.46059438e-02 8.07465260e+00 1.52632113e+00
 4.86864794e-03 9.73729587e-02 7.30297190e-03 3.55411299e-01
 6.81610711e+00 2.60229232e+00 4.13835075e-02 7.78983670e-02
 2.03022619e+00 9.46952024e-01 3.65148595e-02 1.50928086e-01
 2.65341313e-01 2.19089157e-02 3.31068060e-01]
[6.23795517e-01 1.52145248e-02 8.41109646e+00 1.58991784e+00
 5.07150827e-03 1.01430165e-01 7.60726240e-03 3.70220103e-01
 7.10011157e+00 2.71072117e+00 4.31078203e-02 8.11441323e-02
 2.11481895e+00 9.86408358e-01 3.80363120e-02 1.57216756e-01
 2.76397201e-01 2.28217872e-02 3.44862562e-01]
[1.64682016e+00 4.01663455e-02 2.22052947e+01 4.19738310e+00
 1.33887818e-02 2.67775636e-01 2.00831727e-02 9.77381073e-01
 1.87442946e+01 7.15630388e+00 1.13804646e-01 2.14220509e-01
 5.58312202e+00 2.60411806e+00 1.00415864e-01 4.15052237e-01
 7.29688609e-01 6.02495182e-02 9.10437164e-01]
[2.61994117e+00 6.39010042e-02 3.53266051e+01 6.67765493e+00
 2.13003347e-02 4.26006694e-01 3.19505021e-02 1.55492443e+00
 2.98204686e+01 1.13850289e+01 1.81052845e-01 3.40805356e-01
 8.88223958e+00 4.14291510e+00 1.59752510e-01 6.60310376e-01
 1.16086824e+00 9.58515062e-02 1.44842276e+00]
[1.47215742e+00 3.59062785e-02 1.98501876e+01 3.75220611e+00
 1.19687595e-02 2.39375190e-01 1.79531393e-02 8.73719444e-01
 1.67562633e+01 6.39730196e+00 1.01734456e-01 1.91500152e-01
 4.99097272e+00 2.32792372e+00 8.97656963e-02 3.71031545e-01
 6.52297393e-01 5.38594178e-02 8.13875647e-01]
[4.36407344e+01 1.06440815e+00 5.88440308e+02 1.11230652e+02
 3.54802718e-01 7.09605437e+00 5.32204077e-01 2.59005984e+01
 4.96723806e+02 1.89642053e+02 3.01582311e+00 5.67684349e+00
 1.47952734e+02 6.90091287e+01 2.66102039e+00 1.09988843e+01
 1.93367481e+01 1.59661223e+00 2.41265848e+01]
```

```
[3.99229131e-01 9.73729587e-03 5.38310173e+00 1.01754742e+00
 3.24576529e-03 6.49153058e-02 4.86864794e-03 2.36940866e-01
 4.54407141e+00 1.73486155e+00 2.75890050e-02 5.19322446e-02
 1.35348413e+00 6.31301349e-01 2.43432397e-02 1.00618724e-01
 1.76894208e-01 1.46059438e-02 2.20712040e-01]
[2.47023025e+00 6.02495182e-02 3.33079420e+01 6.29607465e+00
 2.00831727e-02 4.01663455e-01 3.01247591e-02 1.46607161e+00
 2.81164418e+01 1.07344558e+01 1.70706968e-01 3.21330764e-01
 8.37468303e+00 3.90617710e+00 1.50623796e-01 6.22578355e-01
 1.09453291e+00 9.03742773e-02 1.36565575e+00]
[1.27254285e+00 3.10376306e-02 1.71586368e+01 3.24343240e+00
 1.03458769e-02 2.06917537e-01 1.55188153e-02 7.55249011e-01
 1.44842276e+01 5.52987118e+00 8.79399533e-02 1.65534030e-01
 4.31423065e+00 2.01227305e+00 7.75940765e-02 3.20722183e-01
 5.63850289e-01 4.65564459e-02 7.03519627e-01]
[1.62186834e+00 3.95577645e-02 2.18688508e+01 4.13378639e+00
 1.31859215e-02 2.63718430e-01 1.97788822e-02 9.62572269e-01
 1.84602901e+01 7.04787504e+00 1.12080333e-01 2.10974744e-01
 5.49852926e+00 2.56466173e+00 9.88944112e-02 4.08763566e-01
 7.18632721e-01 5.93366467e-02 8.96642662e-01]
[1.84643473e+00 4.50349934e-02 2.48968455e+01 4.70615681e+00
 1.50116645e-02 3.00233289e-01 2.25174967e-02 1.09585151e+00
 2.10163303e+01 8.02373466e+00 1.27599148e-01 2.40186632e-01
 6.25986408e+00 2.91976874e+00 1.12587484e-01 4.65361599e-01
 8.18135714e-01 6.75524901e-02 1.02079318e+00]
[6.03834060e+00 1.47276600e-01 8.14194137e+01 1.53904047e+01
 4.90922000e-02 9.81844000e-01 7.36383000e-02 3.58373060e+00
 6.87290800e+01 2.62397809e+01 4.17283700e-01 7.85475200e-01
 2.04714474e+01 9.54843290e+00 3.68191500e-01 1.52185820e+00
 2.67552490e+00 2.20914900e-01 3.33826960e+00]
[4.24180951e-01 1.03458769e-02 5.71954559e+00 1.08114413e+00
 3.44862562e-03 6.89725124e-02 5.17293843e-03 2.51749670e-01
 4.82807587e+00 1.84329039e+00 2.93133178e-02 5.51780099e-02
 1.43807688e+00 6.70757683e-01 2.58646922e-02 1.06907394e-01
 1.87950096e-01 1.55188153e-02 2.34506542e-01]
[8.23410082e-01 2.00831727e-02 1.11026473e+01 2.09869155e+00
 6.69439091e-03 1.33887818e-01 1.00415864e-02 4.88690537e-01
 9.37214728e+00 3.57815194e+00 5.69023228e-02 1.07110255e-01
 2.79156101e+00 1.30205903e+00 5.02079318e-02 2.07526118e-01
 3.64844305e-01 3.01247591e-02 4.55218582e-01]
[9.73121006e-01 2.37346587e-02 1.31213105e+01 2.48027183e+00
 7.91155290e-03 1.58231058e-01 1.18673293e-02 5.77543361e-01
 1.10761741e+01 4.22872502e+00 6.72481996e-02 1.26584846e-01
 3.29911756e+00 1.53879704e+00 5.93366467e-02 2.45258140e-01
 4.31179633e-01 3.56019880e-02 5.37985597e-01]
[2.04604930e+00 4.99036413e-02 2.75883964e+01 5.21493052e+00
 1.66345471e-02 3.32690942e-01 2.49518207e-02 1.21432194e+00
 2.32883660e+01 8.89116543e+00 1.41393650e-01 2.66152754e-01
```

```
  6.93660615e+00 3.23541941e+00 1.24759103e-01 5.15670961e-01
  9.06582818e-01 7.48554620e-02 1.13114920e+00]
 [2.59498935e+00 6.32924232e-02 3.49901613e+01 6.61405822e+00
  2.10974744e-02 4.21949488e-01 3.16462116e-02 1.54011563e+00
  2.95364641e+01 1.12766001e+01 1.79328532e-01 3.37559590e-01
  8.79764682e+00 4.10345877e+00 1.58231058e-01 6.54021706e-01
  1.14981235e+00 9.49386347e-02 1.43462826e+00]
 [3.74277310e-01 9.12871488e-03 5.04665788e+00 9.53950705e-01
  3.04290496e-03 6.08580992e-02 4.56435744e-03 2.22132062e-01
  4.26006694e+00 1.62643270e+00 2.58646922e-02 4.86864794e-02
  1.26889137e+00 5.91845015e-01 2.28217872e-02 9.43300538e-02
  1.65838320e-01 1.36930723e-02 2.06917537e-01]
 [1.16525003e+01 2.84207323e-01 1.57119282e+02 2.96996653e+01
  9.47357744e-02 1.89471549e+00 1.42103662e-01 6.91571153e+00
  1.32630084e+02 5.06362714e+01 8.05254083e-01 1.51577239e+00
  3.95048179e+01 1.84261081e+01 7.10518308e-01 2.93680901e+00
  5.16309971e+00 4.26310985e-01 6.44203266e+00]
 [4.99036413e-01 1.21716198e-02 6.72887717e+00 1.27193427e+00
  4.05720661e-03 8.11441323e-02 6.08580992e-03 2.96176083e-01
  5.68008926e+00 2.16857693e+00 3.44862562e-02 6.49153058e-02
  1.69185516e+00 7.89126686e-01 3.04290496e-02 1.25773405e-01
  2.21117760e-01 1.82574298e-02 2.75890050e-01]
 [9.73121006e-01 2.37346587e-02 1.31213105e+01 2.48027183e+00
  7.91155290e-03 1.58231058e-01 1.18673293e-02 5.77543361e-01
  1.10761741e+01 4.22872502e+00 6.72481996e-02 1.26584846e-01
  3.29911756e+00 1.53879704e+00 5.93366467e-02 2.45258140e-01
  4.31179633e-01 3.56019880e-02 5.37985597e-01]
 [2.32051932e+00 5.65980323e-02 3.12892788e+01 5.91449437e+00
  1.88660108e-02 3.77320215e-01 2.82990161e-02 1.37721878e+00
  2.64124151e+01 1.00838827e+01 1.60361091e-01 3.01856172e-01
  7.86712648e+00 3.66943909e+00 1.41495081e-01 5.84846333e-01
  1.02819759e+00 8.48970484e-02 1.28288873e+00]
 [3.19383305e+00 7.78983670e-02 4.30648139e+01 8.14037935e+00
  2.59661223e-02 5.19322446e-01 3.89491835e-02 1.89552693e+00
  3.63525713e+01 1.38788924e+01 2.20712040e-01 4.15457957e-01
  1.08278730e+01 5.05041079e+00 1.94745917e-01 8.04949792e-01
  1.41515367e+00 1.16847550e-01 1.76569632e+00]]
probability=0.950, critical=839.833, stat=4983.525
Dependent (reject H0)
significance=0.050, p=0.000
Dependent (reject H0)
```

```
[ ]:
```