**Customer Service Request**

**DESCRIPTION**

***Background of Problem Statement :***

NYC 311's mission is to provide the public with quick and easy access to all New York City government services and information while offering the best customer service. Each day, NYC311 receives thousands of requests related to several hundred types of non-emergency services, including noise complaints, plumbing issues, and illegally parked cars. These requests are received by NYC311 and forwarded to the relevant agencies such as the police, buildings, or transportation. The agency responds to the request, addresses it, and then closes it.

***Problem Objective:***

Perform a service request data analysis of New York City 311 calls. You will focus on the data wrangling techniques to understand the pattern in the data and also visualize the major complaint types.

Domain: Customer Service

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

import matplotlib as mpl

### **Task 1 : Import a 311 NYC service request**

cust\_service = pd.read\_csv(r"C:\\Users\\Workstation\\OneDrive\\Desktop\\Data Science\_Material\\Data Science with Python\\Python\_Projects\\Customer Service Requests Analysis\\311\_Service\_Requests\_from\_2010\_to\_Present.csv", low\_memory=False)

cust\_service.head()

cust\_service.info()

cust\_service.shape

cust\_service['Status'].unique()

cust\_service['Complaint Type'].unique()

cust\_service['Descriptor'].unique()

***Lots of Columns are empty, whereas many columns have 80% Null Values. i.e These series are of no use. So, we need to remove them from our dataset.***

cust\_service.drop(['Agency Name','Incident Address','Street Name','Cross Street 1','Cross Street 2','Intersection Street 1','Intersection Street 2','Address Type','Park Facility Name','Park Borough','School Name','School Number','School Region','School Code','School Phone Number','School Address','School City','School State','School Zip','School Not Found','School or Citywide Complaint','Vehicle Type','Taxi Company Borough','Taxi Pick Up Location','Bridge Highway Name','Bridge Highway Direction','Road Ramp','Bridge Highway Segment','Garage Lot Name','Ferry Direction','Ferry Terminal Name','Landmark', 'X Coordinate (State Plane)','Y Coordinate (State Plane)','Due Date','Resolution Action Updated Date','Community Board','Facility Type','Location'], axis = 1, inplace = True)

***As most of the requests are 'Closed', hence our analysis will include only 'Closed' requests and we will drop other records. Then we will drop the 'Status' column. Selecting not null values of 'Latitude', 'Longitude' and 'Descriptor' column values will furthere remove empty values.***

cust\_service =cust\_service[cust\_service['Status']=='Closed']

#cust\_service.drop(cust\_service['Status'], inplace = True, axis = 1)

cust\_service = cust\_service[(cust\_service['Latitude'].notnull() & cust\_service['Longitude'].notnull() & cust\_service['Descriptor'].notnull())]

cust\_service.info()

## **Task 2 : Read or convert the columns ‘Created Date’ and Closed Date’ to datetime datatype**

cust\_service['Request\_Creation'] = pd.to\_datetime(cust\_service['Created Date'])

cust\_service['Request\_Closed'] = pd.to\_datetime(cust\_service['Closed Date'])

cust\_service.info()

**We have changed ‘Created Date’ and Closed Date’ which was "object" datatype to "datetime" datatype by putting their value in new columns 'Request\_Creation' and 'Request\_Closed'**

cust\_service['Request\_Closing\_Time'] = cust\_service['Request\_Closed'] - cust\_service['Request\_Creation']

#Calculating the response time in minutes

cust\_service['Request\_Closing\_Time'] = cust\_service['Request\_Closing\_Time']/np.timedelta64(1,'m')

cust\_service['Request\_Closing\_Time'].head()

## **Task 3 : Provide major insights/patterns that you can offer in a visual format (graphs or tables); at least 4 major conclusions that you can come up with after generic data mining.**

***City Names are mixed as some names are lowercase and some UpperCase. We will convert all city names to Camel case using below defined function***.

def camel\_case(city):

try:

city = city.split(' ')

city = ' '.join([x.lower().capitalize() for x in city])

if city == 'Unknown':

return np.nan

else:

return city

except:

return np.nan

# Apply camel\_case function to City column

cust\_service['City'] = cust\_service['City'].apply(camel\_case)

cust\_service['City'].value\_counts().head()

***In the preprocessing we have dropped around 2.8 % of the records. Below count shows that we have a very small number of empty values remaining.***

cust\_service.nunique()

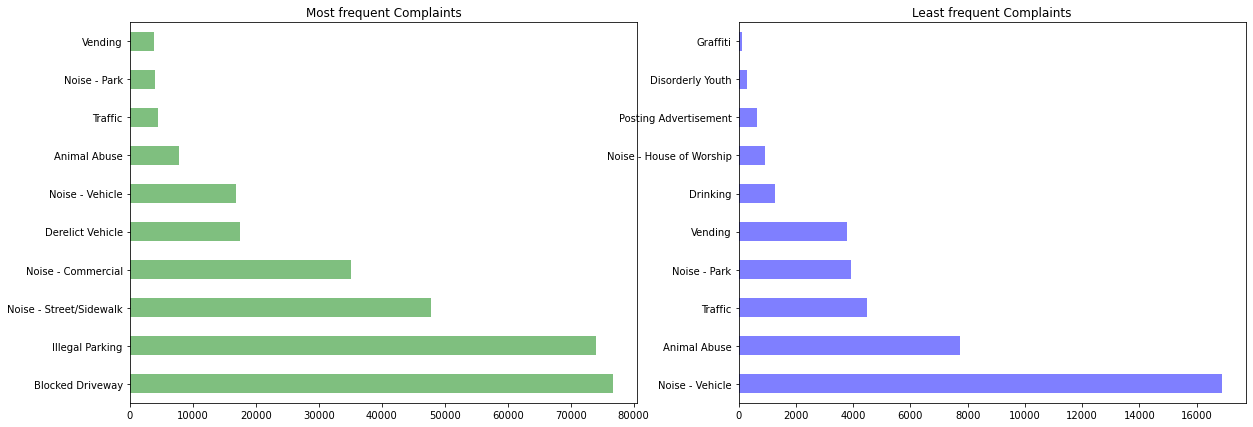
# Data Analysis and Visualization[¶](http://localhost:8888/notebooks/Customer%20Service%20Request%20Project%201.ipynb#Data-Analysis-and-Visualization)

plt.subplot(1, 2, 1)

cust\_service['Complaint Type'].value\_counts().head(10).plot(kind='barh', color = 'Green', alpha = 0.5, figsize=(10,6), title = 'Most frequent Complaints')

plt.subplot(1, 2, 2)

cust\_service['Complaint Type'].value\_counts().tail(10).plot(kind='barh', color = 'Blue', alpha = 0.5, figsize=(10,6), title = 'Least frequent Complaints')

plt.gcf().set\_size\_inches(20, 7)

***Conclusion : Most of the complaints are of 'Driving', 'Parking', 'Noise' which is related to traffic and transporations Least Frequent Complaintsa are of 'Urinating in Public', 'Bike/Roller/Skate Chronic','Panhandling'***

top\_city\_requests = cust\_service['City'].value\_counts().head(10)

top10 = top\_city\_requests.index

txt = {'weight':'bold'}

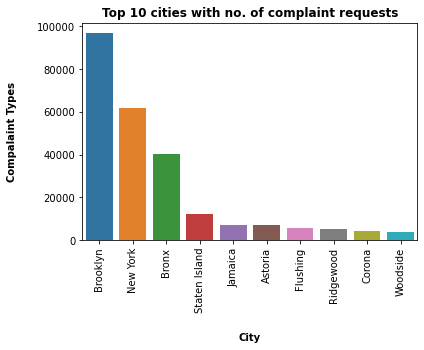
sns.countplot(data = cust\_service[cust\_service.City.isin(top10)],x= 'City', order = top10)

plt.title("Top 10 cities with no. of complaint requests", fontdict = txt)

plt.xticks(rotation = 90)

plt.xlabel('City', fontdict = txt, labelpad = 20)

plt.ylabel('Compalaint Types', fontdict = txt, labelpad = 20)



***Conclusion 2****: Most of the Complaints are from 'Brooklyn', 'New York' and 'Bronx' Cities*

txt = {'weight':'bold'}

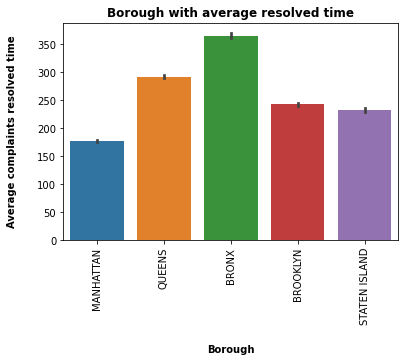
sns.barplot(data = cust\_service,x= 'Borough',y= 'Request\_Closing\_Time')

plt.title("Borough with average resolved time", fontdict = txt)

plt.xticks(rotation = 90)

plt.xlabel('Borough', fontdict = txt, labelpad = 20)

plt.ylabel('Average complaints resolved time', fontdict = txt, labelpad = 20)



***Conclusion 3****: 'Manhattan' has the minimum average complaint response time and 'Bronx' has the maximum average complaint response time*

dt = cust\_service[['Complaint Type', 'Request\_Closing\_Time']]

dt1 = dt.groupby('Complaint Type')['Request\_Closing\_Time'].mean().to\_frame()

dt1 = dt1.sort\_values('Request\_Closing\_Time')

dt1['Complaint Type'] = dt1.index

txt = {'weight':'bold'}

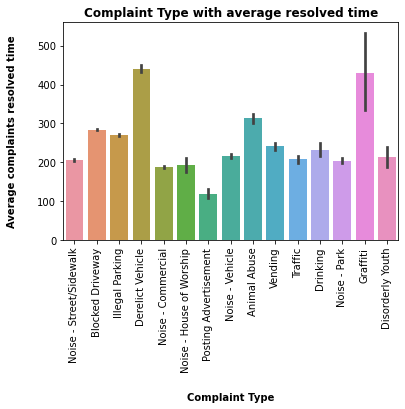
sns.barplot(data = cust\_service ,x= 'Complaint Type',y= 'Request\_Closing\_Time')

plt.title("Complaint Type with average resolved time", fontdict = txt)

plt.xticks(rotation = 90)

plt.xlabel('Complaint Type', fontdict = txt, labelpad = 20)

plt.ylabel('Average complaints resolved time', fontdict = txt, labelpad = 20)



***Conclusion 4****: 'Posting Advertisement' complaints are responded faster and 'Derelict Vehicle Complaints' are responded slower*

top6\_complaints = ['Blocked Driveway','Illegal Parking','Noise - Commercial','Noise - Street/Sidewalk','Derelict Vehicle','Animal Abuse']

comp\_borough = cust\_service.groupby(['Borough','Complaint Type']).size().unstack()

comp\_borough = comp\_borough[top6\_complaints]

comp\_borough

mpl.style.use('fivethirtyeight')

col\_number = 2

row\_number = 3

txt = {'weight' : 'bold'}

fig, axes = plt.subplots(row\_number,col\_number)

for i, (label,col) in enumerate(comp\_borough.iteritems()):

ax = axes[int(i/col\_number), i%col\_number]

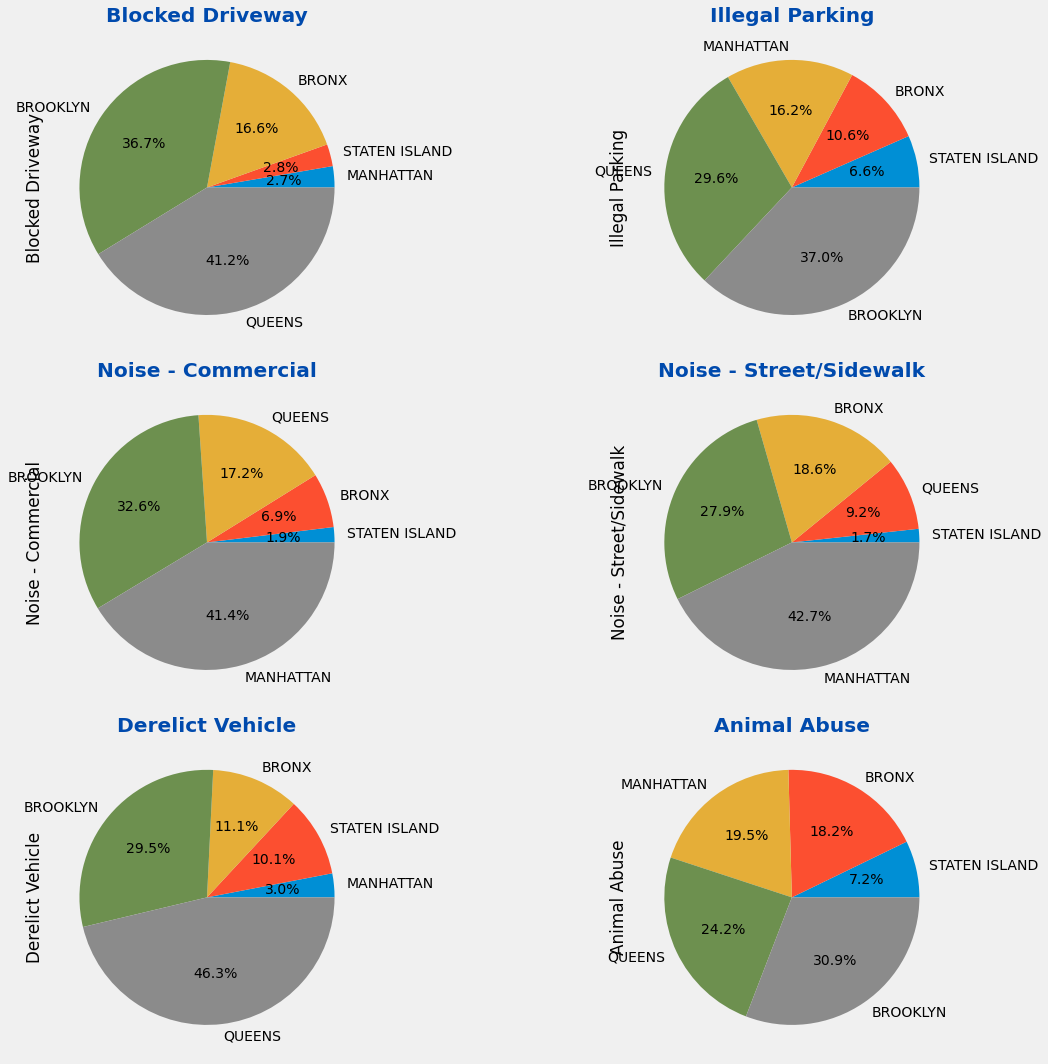
col = col.sort\_values(ascending=True)[:15]

col.plot(kind='pie', ax=ax, autopct = '%1.1f%%' )

ax.set\_title(label, color = '#004aad', fontdict = txt)

plt.gcf().set\_size\_inches(18, 15)

plt.tight\_layout()



***Conclusion - 5***

* *Clearly Manhattan is making most of the noise, followed by Brooklyn.*
* *Brooklyn has the most number of 'Illegal parking' complaints and is also on the top for 'Animal Abuse'. (Savages!)*
* *Queens has highest complaints for 'Blocked Driveway' as well as for 'Derelict Vehicle'.*
* *Lets now breakdown the complaints for each Borough*

mpl.style.use('fivethirtyeight')

borough\_comp = cust\_service.groupby(['Complaint Type','Borough']).size().unstack()

row\_number = 3

col\_number = 2

txt = {'weight' : 'bold'}

fig, axes = plt.subplots(row\_number,col\_number, figsize=(12,12))

for i, (label,col) in enumerate(borough\_comp.iteritems()):

ax = axes[int(i/col\_number), i%col\_number]

col = col.sort\_values(ascending=True)[:15]

col.plot(kind='barh', ax=ax, color = '#ecbfbf')

ax.set\_title(label, fontdict = txt, color = '#841a41')

plt.tight\_layout()



***Conclusion 6***

* *Apart from Manhattan, number of Complaints for 'Blocked Driveway' and 'Illegal Parking' is highest for each Borough.*
* *Manhattan has highest number of Noise complaints. Fortunaltely, parking in Manhattan is better than other Boroughs.*

## **Task 4 :Order the complaint types based on the average ‘Request\_Closing\_Time’, grouping them for different locations**

CityComplaint= cust\_service.groupby(['Complaint Type','Borough']).Request\_Closing\_Time.mean().unstack()

row\_number = 2

col\_number = 3

txt = {'weight':'bold', 'color':'#bc8273'}

fig, axes = plt.subplots(row\_number, col\_number, figsize = (12,8))

for i,(label,col) in enumerate(CityComplaint.iteritems()):

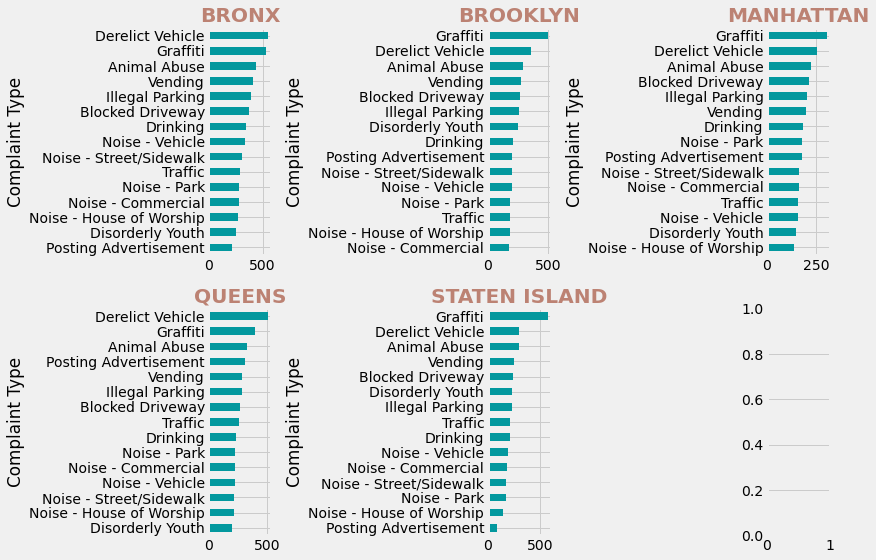
ax = axes[int(i/col\_number), int(i%col\_number)]

col = col.sort\_values(ascending = True)[:15]

col.plot(kind = 'barh', ax = ax, color = '#03989e')

ax.set\_title(label, fontdict = txt)

plt.tight\_layout()



***Conclusion****- Clealry 'Graffiti' complaints are taking a long time to be closed. Could be because the number of Graffiti complaints are very less and officials are focused on more pressing issues.*

* *Manhattan, Bronx and Queens are handling 'Disorderly Youth' complaints very well.*
* *Brooklyn is performing well to close Noise and traffic complaints.*

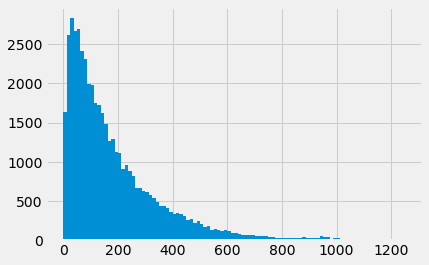
## **Task 5 : Hypothesis Testing**

***Perform a statistical test for the following:***

***Please note: For the below statements you need to state the Null and Alternate and then provide a statistical test to accept or reject the Null Hypothesis along with the corresponding ‘p-value’.***

original = cust\_service[cust\_service['Complaint Type']=='Noise - Street/Sidewalk']['Request\_Closing\_Time']

original.hist(bins=100,range=(0,1250))



original.describe()

data = {}

for complaint in cust\_service['Complaint Type'].unique():

data[complaint] = np.log(cust\_service[cust\_service['Complaint Type']==complaint]['Request\_Closing\_Time'])

for complaint in data.keys():

print(data[complaint].std())

***To conduct our hypothesis test, we will conduct an ANOVA (analysis of variance) test as we have to compare the means of more than two groups. Below conditions should be met before conducting ANOVA.***

* ***All distributions must follow a normal distributions curve. We have verified this after the log transformation***
* ***Standard deviation for all groups must be same. Above output proves that this is true.***
* ***All samples are drawn independently of each other.***

from scipy.stats import f\_oneway

stat, p = f\_oneway(data['Noise - Street/Sidewalk'],data['Blocked Driveway'],data['Illegal Parking'],data['Derelict Vehicle'],

data['Noise - Commercial'])

print('Statistics=%.3f, p=%.3f' % (stat, p))

# interpret

alpha = 0.05

if p > alpha:

print('Same distributions (fail to reject H0)')

else:

print('Different distributions (reject H0)')

***Conclusion****: As our p-value is quite low , hence it is being converted to 0.0 Since our p-value is lowere that our critical p-value, we will conclude that we have enogh evidence to reject pur Null Hypothesis*

**Average response time for all the complaints type is not same.**

## **Task 6 : Are the type of complaint or service requested and location related?**

CType\_Location = pd.crosstab(cust\_service['City'], cust\_service['Complaint Type'])

from scipy.stats import chi2\_contingency

from scipy.stats import chi2

table = CType\_Location

stat, p, dof, expected = chi2\_contingency(table)

print('dof=%d' %dof)

print(expected)

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 HD)')

else:

print('Independent (fail to reject HD)')

# interpret p-value

alpha = 1.0 - prob

print('significance=%.3f, p=%.3f' % (alpha, p))

if p <= alpha:

print('Dependent (reject HD)')

else:

print('Independent (fail to reject HD)')

***Conclusion****: There is no relationship between complaint type and location. By applying chisquare test, p< alpha. our hypothesis went wrong. We can conclude that complaint type/service requested and location are not related.*