Customer Service Requests Analysis

311 Service Requests from 2010 to Present

DESCRIPTION:

You have been asked to perform data analysis of service request (311) calls from New York City. You have also been asked to utilize data wrangling techniques to understand the pattern in the data and visualize the major types of complaints.

PERFORM THE FOLLOWING STEPS:

INITIALLY IMPORT ALL LIBRARY

- ✓ *import numpy as np*
- ✓ *import pandas as pd*
- ✓ *import matplotlib.pyplot as plt*
- ✓ *import seaborn as sns*
- ✓ *import warnings warnings.filterwarnings('ignore')*

IMPORT THE DATA SET BY USING PANDAS LIBRARY

- ✓ *calls = pd.read_csv('311_Service_Requests_from_2010_to_Present.csv')*

1.UNDERSTAND THE DATASET

- ✓ *calls.head(): This will display the first few rows of the dataset.*
- ✓ *calls. shape: This will give you the number of rows and columns in the dataset.*
- ✓ *calls.info (): This will display information about each column, such as its data type and the number of non-null values*
- ✓ *calls.describe(): This will give you a summary of statistics for each numeric column in the dataset.*

1.1 UNDERSTANDING THE SHAPE OF THE DATASET

- ✓ *To identify the shape of a dataset, we can use the. shape attribute in pandas*
- ✓ *This will output a tuple containing the number of rows and columns in the dataset in the format (rows, columns)*
- ✓ *Our output is (300698, 53)*

1.2 IDENTIFY VARIABLES WITH NULL VALUES

- ✓ *To identify variables with null values in a dataset, you can use the isnull() function in pandas*
- ✓ *This function returns a Boolean Data Frame showing which values are missing (True) and which values are present (False)*
- ✓ *There are two type we can identify*
- ✓ *1.calls.isnull().sum()*
- ✓ *2.calls.isna().sum()*
- ✓ *Total null values from data set also we can find by using this attribute*
- ✓ *calls.isna().sum().sum()*

2. PERFORM BASIC DATA EXPLORATORY ANALYSIS

2.1 UTILIZE MISSING VALUE TREATMENT

- ✓ *Total 14 columns are having almost Null values (Unnecessary Column)*
- ✓ *we should remove those columns before treating null value using dropna or fillna*
- ✓ *Because if use dropna all rows are deleting*
- ✓ *Example calls['Garage Lot Name'] has 364558 null values if we use dropna pandas attribute all row were deleting*
- ✓ *Eventually we could not able to Imputing missing values because unnecessarily added with our dataset*

2.1 STPE1- SOME SELF -STUDY TECHNIQUE TO DROP USELESS COLUMNS

- ✓ *So below some self-study technique I used*
- ✓ *Cut-off values should occur or must be remove the column*
- ✓ *Total row * 90% = result*
- ✓ *364558 * 90% = 328102.2 data must be occur in columns or otherwise we will remove the column to get better result*

2.1 STPE2 - FOCUSING NA VALUES FOR FILLING VALUES

- ✓ *I divided my Data set into two types one Numerical columns and another one is Categorical columns*
- ✓ *for Categorical Na values I am using filna by mode function (attribute from pandas)*
- ✓ *for Numerical Na values I am using filna by Interpolate function (attribute from pandas)*
- ✓ *Closed date has Na values It is really import column to find the Average Response Time if using Interpolate function for numerical could not able get accurate result so that in Closed Date Na value removed from the data set*

2.2 ANALYSE THE DATE COLUMN AND REMOVE THE ENTRIES IF IT HAS AN INCORRECT TIMELINE

- ✓ *There are two columns have date format stored values*
- ✓ *Created Date*
- ✓ *Closed Date*

2.2 STEPS INVOLVED TO REMOVE INCORRECT DATE FORMAT TIMELINE

- ✓ *Step - 1 Convert the date column to datetime format*
- ✓ *Step - 2 Analyse the date column with describe()*
- ✓ *Step-3- Checking the outlier by using Conditional (Start date and end date)*

2.1.1 DRAW A FREQUENCY PLOT FOR CITY-WISE COMPLAINTS

- ✓ *Solved this task by two ways*
- ✓ *Histplot from seaborn and Bar Plot from Matplot.pylot*

2.2.2 DRAW SCATTER PLOT FOR COMPLAINT CONCENTRATION ACROSS BROOKLYN

- ✓ *Steps Involved to Draw a Scatter plot for Concentration across Brooklyn*
- ✓ *step1- one hot labelling for City column with new variable*
- ✓ *step2- adding only Brooklyn column into original data set*
- ✓ *step3 -Created a variable and used conditional Brooklyn as 1*
- ✓ *now we have all columns under Brooklyn city*
- ✓ *Note - scatter plot will happen based on numerical*
- ✓ *for plotting it require two numerical columns*
- ✓ *for sns it requires x= numerical and y = categorical*
- ✓ *for hexbin will works with only Floating-point values*

3. Find major types of complaints

3.1 Plot Bar Graph of Count vs Compliant Types

- ✓ *To plot a bar graph of count versus categorical column values*
- ✓ *We can use seaborn's count plot() function*

3.2 Find the top 10 types of complaints

- ✓ *We can also use the nlargest() function to find the top n types from a categorical column in a pandas dataframe*

3.3 Display the types of complaints in each city in separate dataset

- ✓ *Steps Involved for this task*
- ✓ *Step 1- we need to create group by for city column*
- ✓ *Step 2- Create a dictionary of Data Frames, with one Data Frame for each category*
- ✓ *Step 3- Print the unique subcategories for each category*
- ✓ *Step 4 -filling Na values by using numpy.np*
- ✓ *(Could not able create dataframe due to same length value error)*
- ✓ *Step 5 - Creating Dataframe from dictionary key value pair along with nan by pandas as new data set*

4.Visualize the major types of complaints in each city

- ✓ *pd.crosstab() is a pandas function that is used to create a cross-tabulation*
- ✓ *Table between two or more categorical variables*
- ✓ *It is a convenient way to summarize and analyse the relationships between categorical variables in a dataset*

5.Check if the average response time across various types of Complaints

- ✓ *Steps involved for this task*
- ✓ *Step1 - Subtraction for create date column and closed Date column*
- ✓ *Step 2 - convert the 'Time Difference' column to seconds*
- ✓ *(Because without converting the output was in days only or it was floating point values)*
- ✓ *Step 3 - First I grouped by Complaint type column and then I used aggregate function for checking average response*

- ✓ *Step 4 - pd.to_timedelta() is a pandas function*
- ✓ *(For easily readable like human readable*
- ✓ *it is used to convert a numeric value (in seconds, minutes, hours, or days) to a time delta object*
- ✓ *Time delta objects represent a duration of time*
- ✓ *like datetime objects which represent a specific point in time)*

5.1 Using Plot for Checking Average Time Response