In [ ]:
```
# Import all the packages in a new code block before its use
## Type your registration number and Name
Registration_Number = "22011103110"
Name = "Deepthi I"
```

LAB 1 (26-12-2023)

The following topics will be tested/reviewed in this notebook.

- Basic I/O in python
- Functions and Classes in Python
- Descision statements and Loops
- Numpy and Matplotlib
- pandas
- scikit-learn

Basic I/O in python

Write python program for the following:

- print the result of $\sqrt{b^2 - 4ac}$
- accept a string and print the reverse of the string

In [ ]:
```
import math
a = float(input('Enter a: '))
b = float(input('Enter b: '))
c = float(input('Enter c: '))
s = (b**2) - (4*a*c)
d = math.sqrt(s)

if(d>0):
    print("root exits: " , d)
if(d==0):
    print("equal roots: ", d)
if(d<0):
    print("root doesnt exists")
```

```
root exits:  29.866369046136157
```

In [ ]:
```
str = input("enter a string: " )
ans = str[::-1]
print(ans)
```

```
olleg
```

Decision statements and Loops

Write python program for the following:

- Given a list, check how many times an element is present in the list

- Given two numbers a, b and b>a, find how many even numbers, odd numbers and prime numbers are there.

In [ ]:
```
lst = [3,5,2,4,5,4,3,3,3]
for i in lst:
    x = lst.count(i)
    print(i,x)
```

```
3 4
5 2
2 1
4 2
5 2
4 2
3 4
3 4
3 4
```

In [ ]:
```
a=100
b=200
even = 0
odd =0
prime =0

for i in range (a, b):
    if (i%2 == 0):
        even = even +1

    if(i>1):
        for num in range(2,i):
            if(i%num == 0):
              prime = prime + 1
    if(i%2 != 0):
        odd = odd +1


print(even)
print(odd)
print(prime)
```

```
50
50
413
```

Functions and Classes

Write python functions for the following:

- accept a string (a row of comma separated values) and returns a list whose elemetns are the values from the string
- given a list of names return the total characters in that list

Write class for the following:

- Complex number arithmetic (+,*) Try operator overloading

In [ ]:
```
def stringtolist(input_str):
    return input_str.split(',')



user_input = input("enter a string: ")
result_list = stringtolist(user_input)
print("result: " , result_list)
```

```
result:  ['d', 'g', 'g', 'h', 'y']
```

In [ ]:
```
def total_characters(names_list):
    return sum(len(name) for name in names_list)

names = input("enter the list of names: ")
result = total_characters(names)
print("total characters in the list: ", result)
```

```
total characters in the list:  24
```

In [ ]:
```
class complexnumber:
    def __init__(self , real,imag):
        self.real = real
        self.imag = imag

    def __add__(self,other):
        return(complexnumber(self.real + other.real, self.imag + other.imag))


    def __mul__(self,other):
        real_part = self.real * other.real - self.imag* other.imag
        imag_part = self.real * other.imag +self.imag * other.real
        return complexnumber(real_part ,imag_part)


    def __str__(self):
        return f"{self.real} + {self.imag}i"


c1 = complexnumber(2,3)
c2 = complexnumber(1,-1)
sum_result = c1+c2
product_result= c1*c2
print("sum: ", sum_result)
print("product: ", product_result)
```

```
sum:  3 + 2i
product:  5 + 1i
```

Numpy and Matplotlib

TODO

Numpy is a python package that has well-optimised numerical algorithms like matrices, arrays etc.,

- Create a list of numbers and convert it into numpy array
- Create a 10x10 matrix (D) where each element of the matrix is given by the formula

$$D_{ij} = i + j + 1 - ij$$

- Compute the maximum value of the element in the matrix
- go through a short tutorial on numpy

Matplotlib is package that specialises in producing scientific grade plots.

TODO

- Create two random arrays y,z using numpy
- create an array x equal to the size of y the elements of the array x goes from 1 to size(y)
- Plot (x,y) and (x,z)
- Try different features

In [ ]:
```
import numpy as np
numbers = [1,2,3,4,5,6]
numpyarray = np.array(numbers)
print("list: " , numbers)
print("numpy array: " , numpyarray)
```

```
list:  [1, 2, 3, 4, 5, 6]
numpy array:  [1 2 3 4 5 6]
```

In [ ]:
```
import numpy as np
rows , cols =  10, 10
d = np.zeros((rows, cols) , dtype = int)
for i in range(rows):
    for j in  range(cols):
        d[i,j] = i+j+1-i*j
print("matrix d: ")
print(d)
```

```
matrix d:
[[  1   2   3   4   5   6   7   8   9  10]
 [  2   2   2   2   2   2   2   2   2   2]
 [  3   2   1   0  -1  -2  -3  -4  -5  -6]
 [  4   2   0  -2  -4  -6  -8 -10 -12 -14]
 [  5   2  -1  -4  -7 -10 -13 -16 -19 -22]
 [  6   2  -2  -6 -10 -14 -18 -22 -26 -30]
 [  7   2  -3  -8 -13 -18 -23 -28 -33 -38]
 [  8   2  -4 -10 -16 -22 -28 -34 -40 -46]
 [  9   2  -5 -12 -19 -26 -33 -40 -47 -54]
 [ 10   2  -6 -14 -22 -30 -38 -46 -54 -62]]
```

In [ ]:
```
import numpy as np
rows, cols = 10,10
d = np.zeros((rows, cols) , dtype = int)
for i in range(rows):
        for j in  range(cols):
           d[i,j] = i+j+1-i*j

max_value = np.max(d)
print("maximum value in the matrix: ",max_value)
```

```
maximum value in the matrix:  10
```

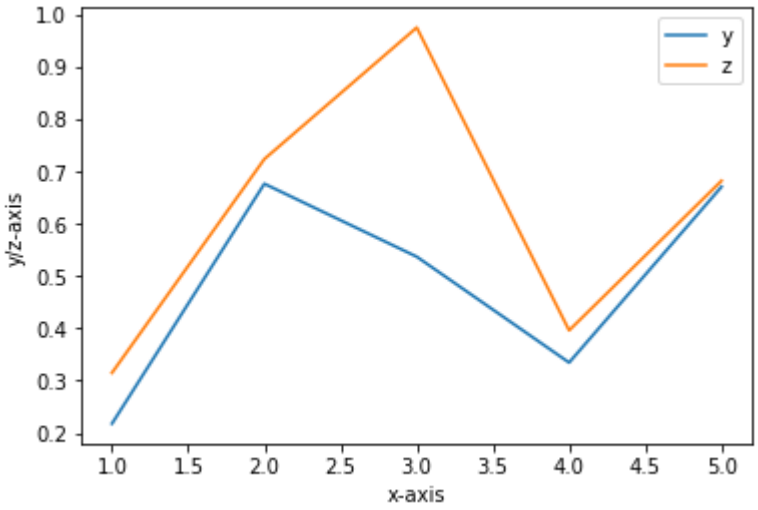
In [ ]:
```
import numpy as np
y =np.random.rand(5)
z = np.random.rand(5)

print("array y: ", y)
print("array z: " , z)
```

```
array y:  [0.21765949 0.67631198 0.5368631  0.33441293 0.67053516]
array z:  [0.31478887 0.72354306 0.97500457 0.39600813 0.68197503]
```

In [ ]:
```
import numpy as np
y_size = len(y)
x = np.arange(1,y_size +1)
print("array x:" , x)
```

```
array x: [1 2 3 4 5]
```

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
y_size = len(y)
z_size = len(z)
plt.plot(range(1,y_size+1), y, label ='y')
plt.plot(range(1,z_size+1), z, label ='z')
plt.xlabel('x-axis')
plt.ylabel('y/z-axis')
plt.legend()
plt.show()
```



pandas

pandas is a python package that is useful for processing tabular data. Usually, if the format of the data is in a .csv, .xls, etc., then pandas is handy in processing.

TODO

- Own a unique dataset to workwith from Free Dataset. You may also choose any dataset from INDIAN DATASETS, US Govt data
- Download it in CSV file
- Clean the dataset i.e., check for any N/A values or empty values and take appropriate action on the column. (Check how to clean dataset using pandas)

- Process the data using padas library (use describe() routine to check the statistical significance of each feature in your dataset)
- Use scatterplot (of pandas) to study how a pair of features are related
- Create a new column from three existing columns. For example if x1,x2,x3 denote three already existing columns then create a new column x4 = 0.56* (x3)*2 + 0.3* (x2)**2 + 0.2 * (x1)

In [ ]:
```
import pandas as pd
column_names = ['VIn','County','City','State','Postal Code','Model Year','Make',
vehicle_data= pd.read_csv('Electric_Vehicle_Population_Data.csv', names= column_
vehicle_data.head()
```

Out[ ]:

	VIn	County	City	State	Postal Code	Model Year	Make	Model	Electr Vehicl Typ
0	5YJYGDEF5L	Thurston	Lacey	WA	98516.0	2020	TESLA	MODEL Y	Batter Electr Vehic (BE
1	1N4BZ1CP1K	King	Sammamish	WA	98074.0	2019	NISSAN	LEAF	Batter Electr Vehic (BE
2	5YJXCDE28G	King	Kent	WA	98031.0	2016	TESLA	MODEL X	Batter Electr Vehic (BE
3	JHMZC5F37M	Kitsap	Poulsbo	WA	98370.0	2021	HONDA	CLARITY	Plug-i Hybri Electr Vehic (PHE
4	WA1F2AFY4P	Thurston	Olympia	WA	98501.0	2023	AUDI	Q5 E	Plug-i Hybri Electr Vehic (PHE

In [ ]:
```
print(vehicle_data.info())
print(vehicle_data.describe())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 163003 entries, 0 to 163002
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   Dtype  
---  ------                                             --------------   -----  
 0   VIn                                                163003 non-null  object 
 1   County                                             162999 non-null  object 
 2   City                                               162999 non-null  object 
 3   State                                              163003 non-null  object 
 4   Postal Code                                        162999 non-null  float64
 5   Model Year                                         163003 non-null  int64  
 6   Make                                               163003 non-null  object 
 7   Model                                              163003 non-null  object 
 8   Electric Vehicle Type                              163003 non-null  object 
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  163003 non-null  object 
 10  Electric Range                                     163003 non-null  int64  
 11  Base MSRP                                          163003 non-null  int64  
 12  Legislative District                               162637 non-null  float64
 13  DOL Vehicle ID                                     163003 non-null  int64  
 14  Vehicle Location                                   162994 non-null  object 
 15  Electric Utility                                   162999 non-null  object 
 16  2020 Census Tract                                  162999 non-null  float64
dtypes: float64(3), int64(4), object(10)
memory usage: 21.1+ MB
None
         Postal Code     Model Year  Electric Range      Base MSRP  \
count  162999.000000  163003.000000   163003.000000  163003.000000   
mean    98170.717422    2020.258449       63.382183    1198.344632   
std      2467.998984       3.005057       94.323062    8825.505678   
min      1730.000000    1997.000000        0.000000       0.000000   
25%     98052.000000    2018.000000        0.000000       0.000000   
50%     98122.000000    2021.000000       13.000000       0.000000   
75%     98370.000000    2023.000000       84.000000       0.000000   
max     99577.000000    2024.000000      337.000000  845000.000000   

       Legislative District  DOL Vehicle ID  2020 Census Tract  
count         162637.000000    1.630030e+05       1.629990e+05  
mean              29.226861    2.153918e+08       5.297368e+10  
std               14.841717    7.874180e+07       1.612977e+09  
min                1.000000    4.385000e+03       1.081042e+09  
25%               18.000000    1.762441e+08       5.303301e+10  
50%               33.000000    2.209718e+08       5.303303e+10  
75%               42.000000    2.495753e+08       5.305307e+10  
max               49.000000    4.792548e+08       5.603300e+10  
```

In [ ]:
```
print(vehicle_data['Model'].value_counts())
```

```
MODEL Y             31640
MODEL 3             28848
LEAF                13264
MODEL S              7670
BOLT EV              6279
                    ...  
XM                      1
HUMMER EV PICKUP        1
S-10 PICKUP             1
BENTAYGA                1
MX-30                   1
Name: Model, Length: 136, dtype: int64
```

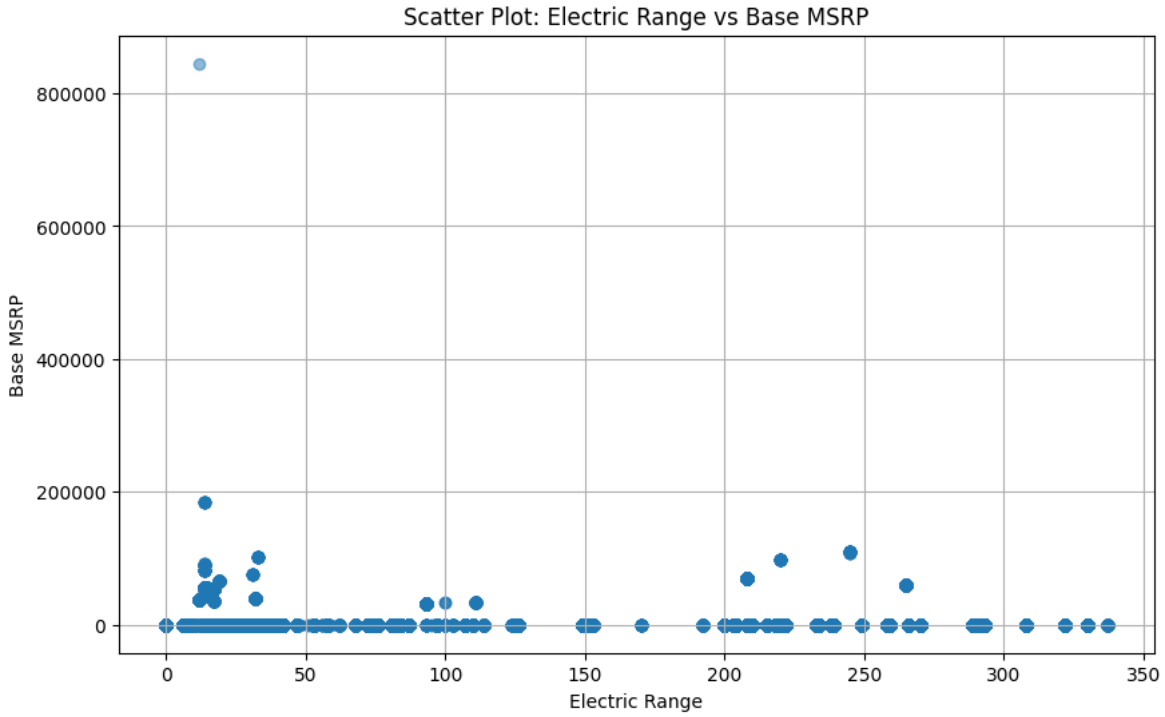
In [ ]:
```
vehicle_data.dropna(inplace=True)
```

In [ ]:
```
duplicate_rows = vehicle_data.duplicated()
print("Number of duplicate rows:", duplicate_rows.sum())
```

```
Number of duplicate rows: 0
```

In [ ]:
```
vehicle_data.drop_duplicates(inplace=True)
```

In [ ]:
```
import pandas as pd

# Assuming you have already defined column_names and loaded your data
```



In [ ]:
```
column_names = ['VIn','County','City','State','Postal Code','Model Year','Make',
vehicle_data = pd.read_csv('Electric_Vehicle_Population_Data.csv', names=column_

import matplotlib.pyplot as plt

# Scatter plot of 'Electric Range' vs 'Base MSRP'
plt.figure(figsize=(10, 6))
plt.scatter(vehicle_data['Electric Range'], vehicle_data['Base MSRP'], alpha=0.5
plt.title('Scatter Plot: Electric Range vs Base MSRP')
plt.xlabel('Electric Range')
plt.ylabel('Base MSRP')
plt.grid(True)
plt.show()
```

In [ ]:
```
import pandas as pd


df = pd.read_csv('Electric_Vehicle_Population_Data.csv')

df['x4'] = 0.56 * (df['Electric Range'])**2 + 0.3 * (df['Base MSRP'])**2 + 0.2 *

print(df.head())
```

```
   VIN (1-10)    County       City State  Postal Code  Model Year    Make  \
0  5YJYGDEF5L  Thurston      Lacey    WA      98516.0        2020   TESLA
1  1N4BZ1CP1K      King  Sammamish    WA      98074.0        2019  NISSAN
2  5YJXCDE28G      King       Kent    WA      98031.0        2016   TESLA
3  JHMZC5F37M    Kitsap    Poulsbo    WA      98370.0        2021   HONDA
4  WA1F2AFY4P  Thurston    Olympia    WA      98501.0        2023    AUDI

     Model                    Electric Vehicle Type  \
0  MODEL Y          Battery Electric Vehicle (BEV)
1     LEAF          Battery Electric Vehicle (BEV)
2  MODEL X          Battery Electric Vehicle (BEV)
3  CLARITY  Plug-in Hybrid Electric Vehicle (PHEV)
4     Q5 E  Plug-in Hybrid Electric Vehicle (PHEV)

  Clean Alternative Fuel Vehicle (CAFV) Eligibility  Electric Range  \
0           Clean Alternative Fuel Vehicle Eligible             291
1           Clean Alternative Fuel Vehicle Eligible             150
2           Clean Alternative Fuel Vehicle Eligible             200
3           Clean Alternative Fuel Vehicle Eligible              47
4             Not eligible due to low battery range              23

   Base MSRP  Legislative District  DOL Vehicle ID  \
0          0                  22.0       124535071
1          0                  45.0       102359449
2          0                  33.0       228682037
3          0                  23.0       171566447
4          0                  22.0       234923230

                 Vehicle Location  \
0  POINT (-122.7474291 47.0821119)
1  POINT (-122.0313266 47.6285782)
2  POINT (-122.2012521 47.3931814)
3     POINT (-122.64177 47.737525)
4     POINT (-122.89692 47.043535)

                              Electric Utility  2020 Census Tract        x4
0                        PUGET SOUND ENERGY INC       5.306701e+10  47425.76
1  PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)       5.303303e+10  12609.00
2  PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)       5.303303e+10  22406.60
3                        PUGET SOUND ENERGY INC       5.303509e+10   1241.64
4                        PUGET SOUND ENERGY INC       5.306701e+10    300.64
```

SCIKIT-LEARN

TODO

- In the above dataset that you have taken, If one of your columns has non-numerical value but it is categorical (Gender, birthplace, etc,.) Use scikit-learn to encode using numerical values.
- Explore various scaling techniques available in scikit-learn, Atleast try two of the scalers with your dataset
- Check what is "Polynomial features" in scikit-learn and try it with your dataset.

In [ ]:
```
import pandas as pd

non_numerical_categorical_columns = df.select_dtypes(include=['object']).columns
```

```
print("Non-numerical categorical columns:", non_numerical_categorical_columns)
```

```
Non-numerical categorical columns: Index(['VIN (1-10)', 'County', 'City', 'Stat
e', 'Make', 'Model',
       'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Vehicle Location',
       'Electric Utility'],
      dtype='object')
```

In [ ]:
```
import pandas as pd
from sklearn.preprocessing import LabelEncoder


categorical_columns = ['VIN (1-10)', 'County', 'City', 'State', 'Make', 'Model',
                       'Electric Vehicle Type',
                       'Clean Alternative Fuel Vehicle (CAFV) Eligibility',
                       'Vehicle Location',
                       'Electric Utility']

df_categorical = df[categorical_columns]


label_encoder = LabelEncoder()


for column in df_categorical.columns:
    df[column] = label_encoder.fit_transform(df[column])
print(df.head())
```

```
   VIN (1-10)  County  City  State  Postal Code  Model Year  Make  Model  \
0        4145     167   302     43      98516.0        2020    33     80
1        1574      79   536     43      98074.0        2019    27     75
2        4030      79   291     43      98031.0        2016    33     79
3        4724      81   485     43      98370.0        2021    14     25
4        7167     167   435     43      98501.0        2023     1     96

   Electric Vehicle Type  Clean Alternative Fuel Vehicle (CAFV) Eligibility  \
0                      0                                                  0
1                      0                                                  0
2                      0                                                  0
3                      1                                                  0
4                      1                                                  2

   Electric Range  Base MSRP  Legislative District  DOL Vehicle ID  \
0             291          0                  22.0       124535071
1             150          0                  45.0       102359449
2             200          0                  33.0       228682037
3              47          0                  23.0       171566447
4              23          0                  22.0       234923230

   Vehicle Location  Electric Utility  2020 Census Tract        x4
0               563                72       5.306701e+10  47425.76
1               325                73       5.303303e+10  12609.00
2               364                73       5.303303e+10  22406.60
3               532                72       5.303509e+10   1241.64
4               585                72       5.306701e+10    300.64
```

In [ ]:
```
import pandas as pd
from sklearn.preprocessing import MinMaxScaler, StandardScaler
numerical_columns = df.select_dtypes(include=['float64', 'int64']).columns
```

```
df_numerical = df[numerical_columns]
min_max_scaler = MinMaxScaler()
```

```
   Postal Code  Model Year  Electric Range  Base MSRP  Legislative District  \
0     0.989157    0.851852        0.863501        0.0              0.437500
1     0.984639    0.814815        0.445104        0.0              0.916667
2     0.984200    0.703704        0.593472        0.0              0.666667
3     0.987664    0.888889        0.139466        0.0              0.458333
4     0.989003    0.962963        0.068249        0.0              0.437500

   DOL Vehicle ID  2020 Census Tract            x4  Postal Code  Model Year  \
0        0.259845           0.946026  2.214001e-07     0.139904   -0.086005
1        0.213573           0.945407  5.886255e-08    -0.039189   -0.418778
2        0.477157           0.945407  1.046014e-07    -0.056612   -1.417099
3        0.357980           0.945445  5.795502e-09     0.080747    0.246768
4        0.490180           0.946026  1.402565e-09     0.133826    0.912315

   ...  VIN (1-10)  County  City  State  Make  Model  Electric Vehicle Type  \
0  ...        4145     167   302     43    33     80                      0
1  ...        1574      79   536     43    27     75                      0
2  ...        4030      79   291     43    33     79                      0
3  ...        4724      81   485     43    14     25                      1
4  ...        7167     167   435     43     1     96                      1

   Clean Alternative Fuel Vehicle (CAFV) Eligibility  Vehicle Location  \
0                                                  0               563
1                                                  0               325
2                                                  0               364
3                                                  0               532
4                                                  2               585

   Electric Utility
0                72
1                73
2                73
3                72
4                72

[5 rows x 26 columns]
```

In [ ]:
```
standard_scaler = StandardScaler()
df_numerical_minmax = pd.DataFrame(min_max_scaler.fit_transform(df_numerical), c
df_numerical_standard = pd.DataFrame(standard_scaler.fit_transform(df_numerical)
df_scaled = pd.concat([df_numerical_minmax, df_numerical_standard, df[categorica
print(df_scaled.head())
```

```
   Postal Code  Model Year  Electric Range  Base MSRP  Legislative District  \
0     0.989157    0.851852        0.863501        0.0              0.437500
1     0.984639    0.814815        0.445104        0.0              0.916667
2     0.984200    0.703704        0.593472        0.0              0.666667
3     0.987664    0.888889        0.139466        0.0              0.458333
4     0.989003    0.962963        0.068249        0.0              0.437500

   DOL Vehicle ID  2020 Census Tract            x4  Postal Code  Model Year  \
0        0.259845           0.946026  2.214001e-07     0.139904   -0.086005
1        0.213573           0.945407  5.886255e-08    -0.039189   -0.418778
2        0.477157           0.945407  1.046014e-07    -0.056612   -1.417099
3        0.357980           0.945445  5.795502e-09     0.080747    0.246768
4        0.490180           0.946026  1.402565e-09     0.133826    0.912315

   ...  VIN (1-10)  County  City  State  Make  Model  Electric Vehicle Type  \
0  ...        4145     167   302     43    33     80                      0
1  ...        1574      79   536     43    27     75                      0
2  ...        4030      79   291     43    33     79                      0
3  ...        4724      81   485     43    14     25                      1
4  ...        7167     167   435     43     1     96                      1

   Clean Alternative Fuel Vehicle (CAFV) Eligibility  Vehicle Location  \
0                                                  0               563
1                                                  0               325
2                                                  0               364
3                                                  0               532
4                                                  2               585

   Electric Utility
0                72
1                73
2                73
3                72
4                72

[5 rows x 26 columns]
```

In [ ]:

```
import pandas as pd
from sklearn.preprocessing import PolynomialFeatures
numerical_columns = df.select_dtypes(include=['float64', 'int64']).columns
df_numerical = df[numerical_columns]
df_numerical.fillna(0, inplace=True)
degree = 2
poly = PolynomialFeatures(degree=degree)
df_poly_features = pd.DataFrame(poly.fit_transform(df_numerical), columns=poly.g
df_poly = pd.concat([df_poly_features, df[categorical_columns]], axis=1)
print(df_poly.head())
```

```
C:\Users\Deepthi\AppData\Local\Temp\ipykernel_6184\3243916778.py:11: SettingWithC
opyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  df_numerical.fillna(0, inplace=True)
```

```
     1  Postal Code  Model Year  Electric Range  Base MSRP  \
0  1.0      98516.0      2020.0           291.0        0.0
1  1.0      98074.0      2019.0           150.0        0.0
2  1.0      98031.0      2016.0           200.0        0.0
3  1.0      98370.0      2021.0            47.0        0.0
4  1.0      98501.0      2023.0            23.0        0.0

   Legislative District  DOL Vehicle ID  2020 Census Tract        x4  \
0                  22.0     124535071.0       5.306701e+10  47425.76
1                  45.0     102359449.0       5.303303e+10  12609.00
2                  33.0     228682037.0       5.303303e+10  22406.60
3                  23.0     171566447.0       5.303509e+10   1241.64
4                  22.0     234923230.0       5.306701e+10    300.64

   Postal Code^2  ...  VIN (1-10)  County  City  State  Make  Model  \
0   9.705402e+09  ...        4145     167   302     43    33     80
1   9.618509e+09  ...        1574      79   536     43    27     75
2   9.610077e+09  ...        4030      79   291     43    33     79
3   9.676657e+09  ...        4724      81   485     43    14     25
4   9.702447e+09  ...        7167     167   435     43     1     96

   Electric Vehicle Type  Clean Alternative Fuel Vehicle (CAFV) Eligibility  \
0                      0                                                  0
1                      0                                                  0
2                      0                                                  0
3                      1                                                  0
4                      1                                                  2

   Vehicle Location  Electric Utility
0               563                72
1               325                73
2               364                73
3               532                72
4               585                72

[5 rows x 55 columns]
```