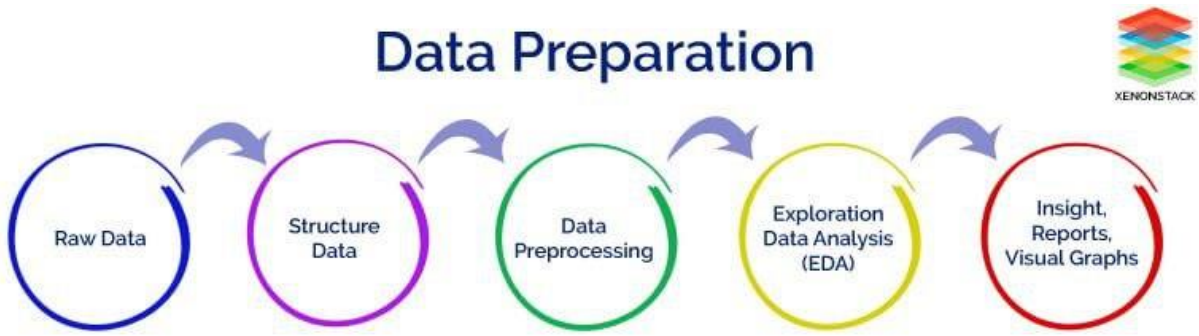
Data Preparation steps:



Data Pre-processing :

Steps taken to pre process the scraped raw data:

1. Ordinal encoded 'Power Train'

2. Label encoded 'Rapid Charge'

3. Used Label Encoder and Standard Scale package for pre processing of the dataset.

Libraries that's are necessary in order to perform data analysis and clustering on the collected data, the following Python libraries are used:

- NumPy: It is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, fourier transform, and matrices.
- 2. Pandas: It is a library written for the Python programming language for data manipulation and analysis
- 3. Matplotlib: It is one of the most popular Python packages used for data visualization. It is a cross-platform library for making 2D plots from data in arrays.

- 4. Seaborn: It is an open-source Python library built on top of matplotlib. It is used for data visualization and exploratory data analysis.
-
- *Importing necessary libraries:*

September 15, 2023

```
: # importing the dependencies
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

Fig 1 : Importing Important Libraries

- *Reading Datasets:*

```
[33]: df1 = pd.read_excel('Ev_charger.xlsx')
      df1.head()
```

[33]:

	Region	2W	3W	4W	Bus	Chargers
0	Uttar Pradesh	9852	42881	458	197	207
1	Maharastra	38558	893	1895	186	317
2	Tamil Nadu	25642	396	426	0	256
3	Karnataka	32844	568	589	57	172
4	Gujarat	22359	254	423	22	228

Fig 2 : Reading Datasets

```
df2 = pd.read_excel('charging_station.xlsx',sheet_name='Table 4', header=1)
df2.head()
```

	State/UT	EV Charging Facility
0	Andhra Pradesh	65
1	Arunachal Pradesh	4
2	Assam	19
3	Bihar	26
4	Chandigarh	4

```
# checking the shape (# of rows and columns) of the datasets
```

Fig 2 : Reading Datasets

- *Analysing the data*:

```
[44]: # checking the shape (# of rows and columns) of the datasets
      print('DF1 Shape: ', df1.shape)
      print('DF2 Shape: ', df2.shape)
```

1

1

```
DF1 Shape:  (24, 6)
DF2 Shape:  (31, 2)
```

Fig 3: Rows and columns of the dataset

```
[37]: d1 = df1.describe()
      d2 = df2.describe()
      display('DATASET 1 ', d1, 'DATASET 2 ', d2, )
```

'DATASET 1 '

	2W	3W	4W	Bus	Chargers
count	24.000000	24.000000	24.000000	24.000000	24.000000

mean	8421.458333	3853.166667	334.041667	28.500000	106.791667
std	10942.261145	8850.690961	476.930628	63.771331	96.623869
min	187.000000	234.000000	12.000000	0.000000	10.000000
25%	848.000000	512.750000	34.750000	0.000000	25.000000
50%	2967.500000	931.000000	129.000000	0.000000	67.500000
75%	10697.750000	2659.250000	434.000000	5.500000	180.250000
max	38558.000000	42881.000000	1895.000000	197.000000	317.000000

'DATASET 2 '

	EV Charging Facility
count	31.000000
mean	49.548387
std	50.768651
min	1.000000
25%	4.000000
50%	26.000000
75%	81.500000
max	174.000000

Fig 4: Information in the dataset

Exploratory Data Analysis:

Exploratory Data Analysis, popularly abbreviated as EDA, is one of the most important steps in the data science pipeline. It is the process of gaining the information present inside the data with the help of summary statistics and visual representations. Keys features of this technique are presented in the below image .We analyzed our dataset using univariate (analyze data over a single variable/column from a dataset), bivariate (analyze data by taking two variables/columns into consideration from a dataset) and multivariate (analyze data by taking more than two variables/columns into consideration from a dataset) analysis.

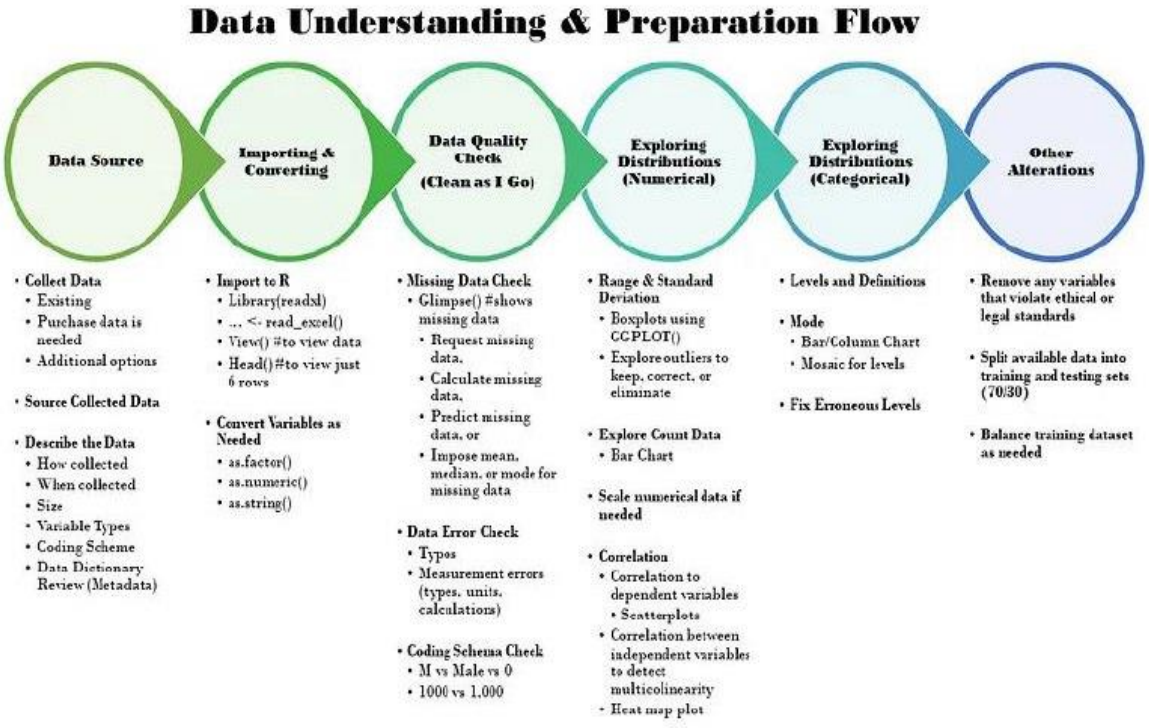


Fig 6: Exploratory Data Analysis

- *Checking for null values in the dataset :*

```
[36]: # checking the info (columns, datatypes, nulls) of the datasets
print('  DATASET 1 ')
print(df1.info())
print('  DATASET 2 ')
print(df2.info())
```

```
  DATASET 1
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 24 entries, 0 to 23
Data columns (total 6 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Region    24 non-null     object
 1   2W        24 non-null     int64
 2   3W        24 non-null     int64
 3   4W        24 non-null     int64
 4   Bus       24 non-null     int64
 5   Chargers  24 non-null     int64
dtypes: int64(5), object(1)
memory usage: 1.3+ KB
None
  DATASET 2
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 31 entries, 0 to 30
Data columns (total 2 columns):
 #   Column
Non-Null Count  Dtype
---  ------
--------------  -----
 0   State/UT
31 non-null     object
 1
EV Charging Facility  31 non-null     int64
dtypes: int64(1), object(1)
memory usage: 628.0+ bytes
None
```

Fig 5: Checking null values

- *Visualization of dataset :*

```
max                                                    174.000000
```

[38]:
```
plt.figure(figsize=(5, 5))
sns.barplot(data=df1, y=df1['Region'].sort_values(ascending=True), x='2W',␣
 ↪palette='viridis')
plt.ylabel('State', fontsize=14, family='serif')
plt.xlabel('Number of EV: 2 Wheelers', family='serif', fontsize=14, labelpad=20)
plt.xticks(family='serif')
plt.yticks(family='serif')
plt.title(label='2 wheelers EV in India', weight=200, family='serif', size=18,␣
 ↪pad=10)
plt.show()
```

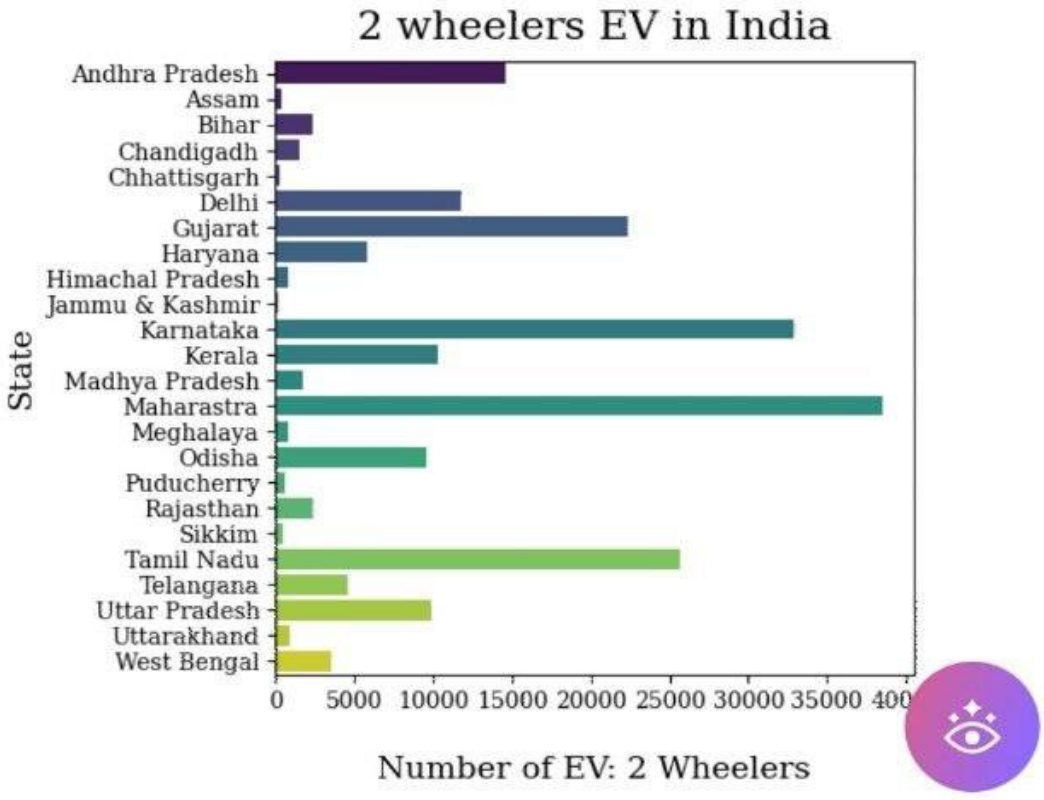


Fig 7: 2 Wheelers EV in India

```
[39]: # 3 wheelers data visualization from dataset 1
plt.figure(figsize=(6, 5))
sns.barplot(data=df1, y=df1['Region'].sort_values(ascending=True), x='3W',␣
 ↪palette='viridis')
plt.ylabel('State', fontsize=14, family='serif')
plt.xlabel('Number of EV: 3 Wheelers', family='serif', fontsize=14, labelpad=20)
plt.xticks(family='serif')
plt.yticks(family='serif')
plt.title(label='3 wheelers EV in India', weight=400, family='serif', size=15,␣
 ↪pad=12)
plt.show()
```

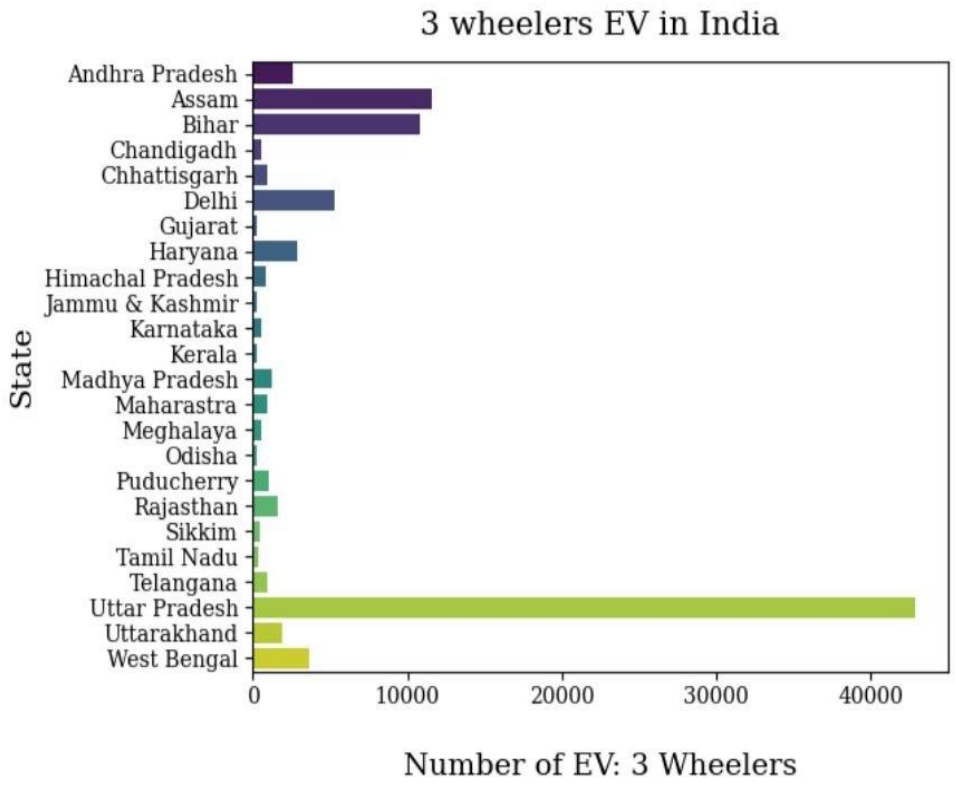


Fig 8: 3 Wheelers EV in India

[40]:
```
# 4 wheelers data visualization from dataset 1
plt.figure(figsize=(6, 5))
sns.barplot(data=df1, y=df1['Region'].sort_values(ascending=True), x='4W',␣
 ↪palette='viridis')
plt.ylabel('State', fontsize=14, family='serif')
plt.xlabel('Number of EV: 4 Wheelers', family='serif', fontsize=14, labelpad=20)
plt.xticks(family='serif')
plt.yticks(family='serif')
plt.title(label='4 wheelers EV in India', weight=400, family='serif', size=15,␣
 ↪pad=12)
plt.show()
```

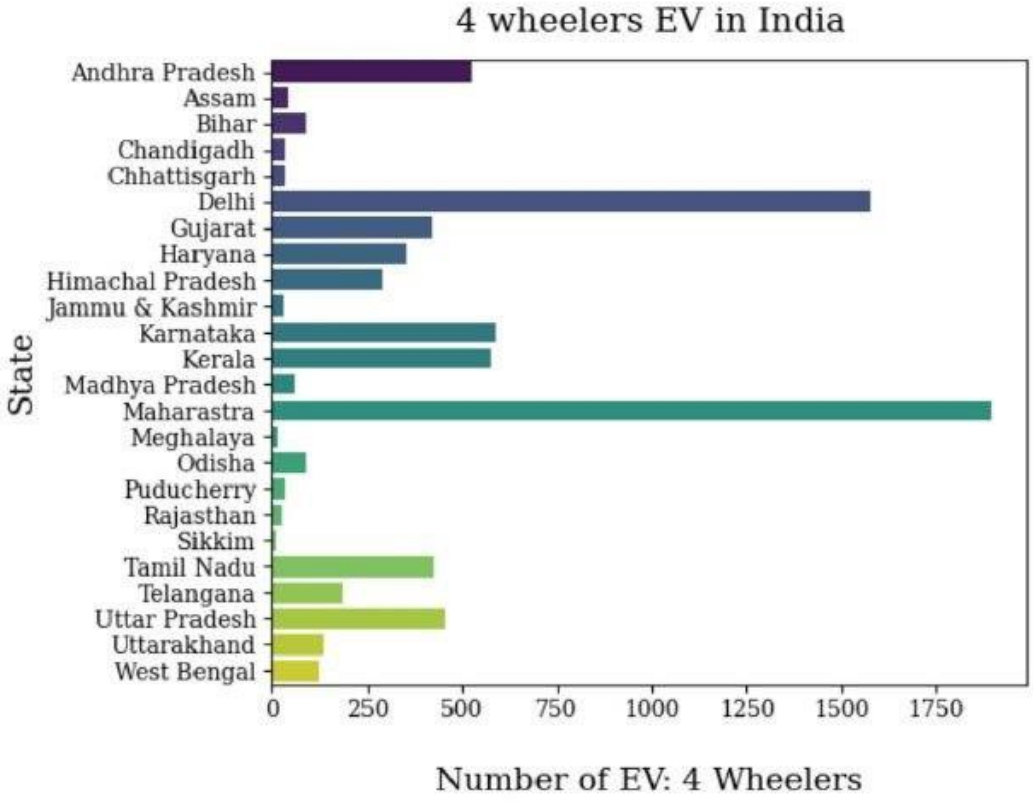


Fig 9: 4 Wheelers EV in India

```
[46]: plt.figure(figsize=(6, 5))
sns.barplot(data=df1, y=df1['Region'].sort_values(ascending=True), x='Bus',␣
  ↪palette='viridis')
plt.ylabel('State', fontsize=14, family='serif')
plt.xlabel('Number of EV: Bus', family='serif', fontsize=14, labelpad=20)
plt.xticks(family='serif')
plt.yticks(family='serif')
plt.title(label='Number of Electric Bus in India', weight=400, family='serif',␣
  ↪size=15, pad=12)
plt.show()
```

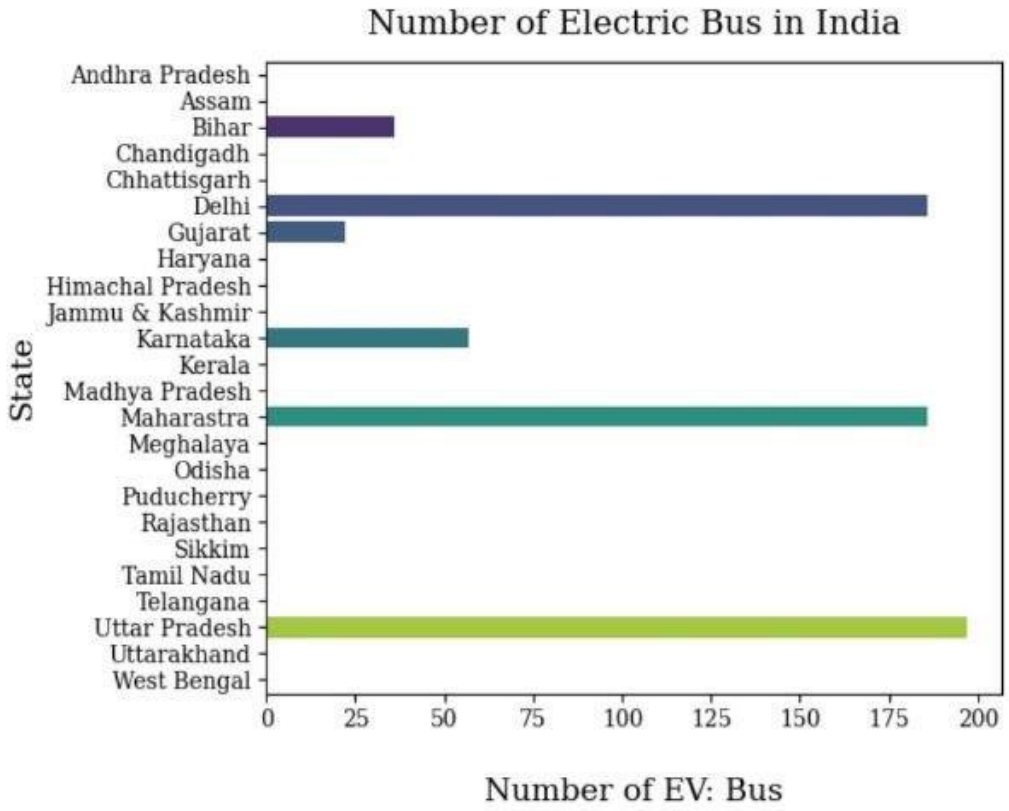



Fig 10: Number Electric Bus in India

```
# charging stations sanctioned visualization from dataset 1
plt.figure(figsize=(6, 5))
sns.barplot(data=df1, y=df1['Region'].sort_values(ascending=True),␣
 ↪x='Chargers', palette='viridis')
plt.ylabel('State', fontsize=14, family='serif')
plt.xlabel('Number of Chargers', family='serif', fontsize=14, labelpad=
```



```
plt.xticks(family='serif')
plt.yticks(family='serif')
plt.title(label='Number of Chargers in India', weight=400, family='serif',␣
 ↪size=15, pad=12)
plt.show()
```

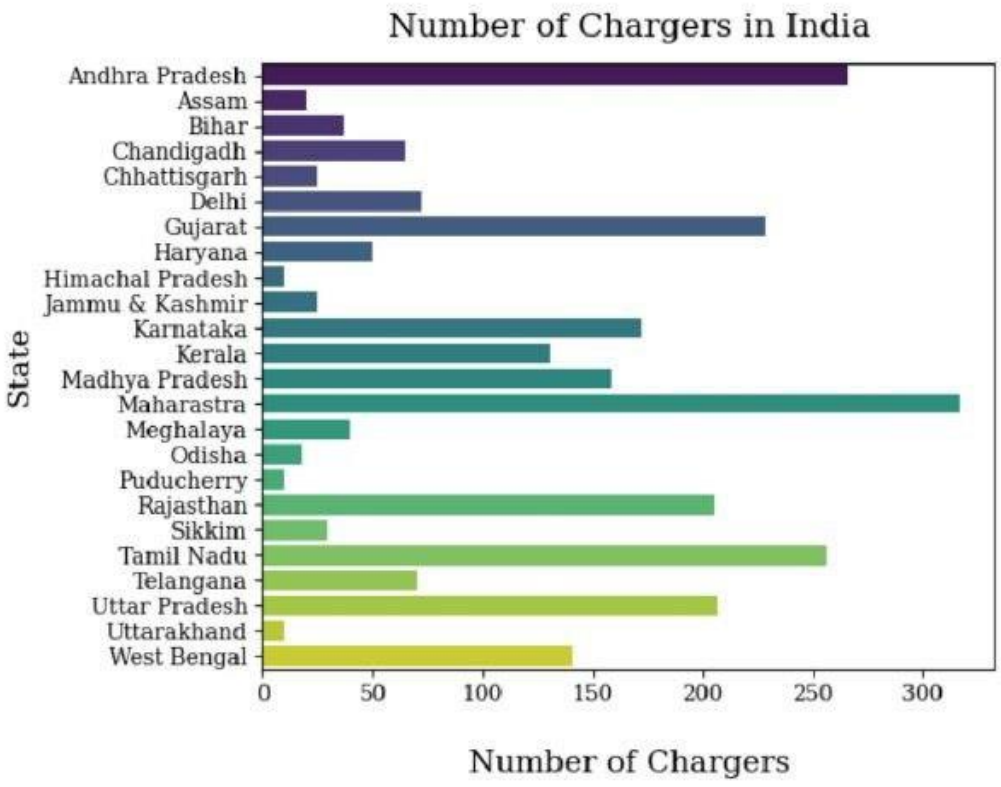


Fig 11: Number of EV Chargers in India

```
[ ]: # retail outlets visualization from dataset - 2
plt.figure(figsize=(6, 8))
sns.pointplot(data=df2, y='State/UT', x='EV Charging Facility', color='orange')
plt.xlabel('Number of Retail Outlets for Charging EVs', family='serif',␣
 ↪size=12, labelpad=10)
plt.ylabel('State/Ut', family='serif', size=12)
plt.tick_params(direction='inout')
plt.xticks(family='serif', size=10)
plt.yticks(family='serif', size=10)
plt.title(label='Available Retail Outlets for Charging EVs in India',␣
 ↪weight=200, family='serif', size=15, pad=12)
plt.show()
```

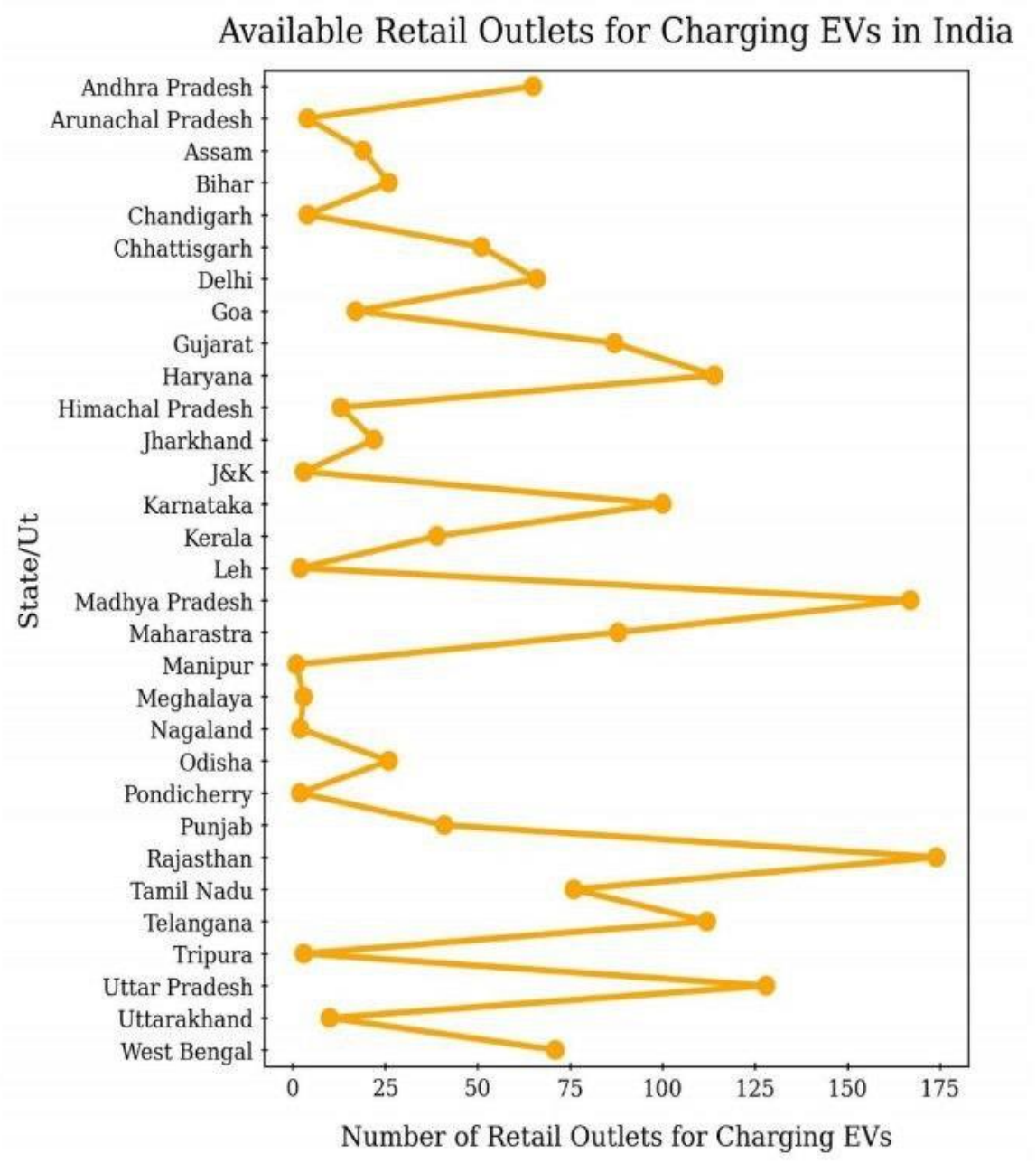


Fig 12: EV Charging Stations