# 1st Dataset- Electric_Vehicle_Population_Data.csv

1. Importing the requisite libraries

```
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans

from warnings import filterwarnings
filterwarnings('ignore')
```

2. Reading the dataset

In this dataset(Electric_Vehicle_Population_Data.csv),we will try to find out the Base MSRP of the vehicle so that it will be easy to ascertain income group of our customer

```
#Reading the dataset
df=pd.read_csv('Electric_Vehicle_Population_Data.csv')
df.head()
```

3. Shape and columns

```
#Checking the shape
df.shape
```

```
(79767, 15)
```

```
#Columns name
df.columns
```

```
Index(['VIN (1-10)', 'County', 'City', 'State', 'ZIP Code', 'Model Year',
       'Make', 'Model', 'Electric Vehicle Type',
       'Clean Alternative Fuel Vehicle (CAFV) Eligibility', 'Electric Range',
       'Base MSRP', 'Legislative District', 'DOL Vehicle ID',
       'Vehicle Location'],
      dtype='object')
```

Description for each column

1. Electric Range:
   - Description: Represents the electric range of the vehicle, indicating the distance it can travel on electric power alone.
   - Type: Numeric (Continuous)

2. Model Year:
   - Description: The year the model of the vehicle was manufactured.
   - Type: Numeric (Discrete)

3. City:
   - Description: The city where the vehicle data was recorded.
   - Type: Categorical (Nominal)

4. State:
   - Description: The state where the vehicle data was recorded.
   - Type: Categorical (Nominal)

5. Make:
   - Description: The manufacturer or brand of the vehicle.
   - Type: Categorical (Nominal)

6. Model:
   - Description: The specific model of the vehicle.
   - Type: Categorical (Nominal)

7. Electric Vehicle Type:
   - Description: Indicates the type of electric vehicle, e.g., Battery Electric Vehicle (BEV), Plug-in Hybrid Electric Vehicle (PHEV).
   - Type: Categorical (Nominal)

8. Clean Alternative Fuel Vehicle (CAFV) Eligibility:
   - Description: Specifies whether the vehicle is eligible for Clean Alternative Fuel Vehicle incentives.
   - Type: Categorical (Nominal)

9. Base MSRP:
   - Description: The Manufacturer's Suggested Retail Price (MSRP) of the vehicle before any additional options.
   - Type: Numeric (Continuous)

4. Checking info

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 79767 entries, 0 to 79766
Data columns (total 15 columns):
 #   Column                                             Non-Null Count  Dtype
---  ------                                             --------------  -----
 0   VIN (1-10)                                         79767 non-null  object
 1   County                                             79762 non-null  object
 2   City                                               79767 non-null  object
 3   State                                              79767 non-null  object
 4   ZIP Code                                           79767 non-null  int64
 5   Model Year                                         79767 non-null  int64
 6   Make                                               79767 non-null  object
 7   Model                                              79767 non-null  object
 8   Electric Vehicle Type                              79767 non-null  object
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  79767 non-null  object
 10  Electric Range                                     79767 non-null  int64
 11  Base MSRP                                          79767 non-null  int64
 12  Legislative District                               79621 non-null  float64
 13  DOL Vehicle ID                                     79767 non-null  int64
 14  Vehicle Location                                   79763 non-null  object
dtypes: float64(1), int64(5), object(9)
memory usage: 9.1+ MB
```

There are 6 numerical and 9 categorical columns

5. Based on our domain knowledge we will drop several columns we are not required for our analysis
   VIN (1-10),County,ZIP Code,Legislative District,DOL Vehicle ID,Vehicle Location are the columns we will drop.

```
df=df.drop(['VIN (1-10)','County','ZIP Code','Legislative District','DOL Vehicle ID','Vehicle Location'],axis=1)
```

6.

```
#Again checking null values
df.isnull().sum()
```

```
City                                               0
State                                              0
Model Year                                         0
Make                                               0
Model                                              0
Electric Vehicle Type                              0
Clean Alternative Fuel Vehicle (CAFV) Eligibility  0
Electric Range                                     0
Base MSRP                                          0
dtype: int64
```

## 7. Summary of Numeric Columns

```
#Summary of Numeric columns
df.describe().T
```

| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **Model Year** | 4478.0 | 2015.765967 | 2.411569 | 2008.0 | 2014.0 | 2016.0 | 2018.0 | 2020.0 |
| **Electric Range** | 4478.0 | 116.360652 | 90.705254 | 11.0 | 26.0 | 93.0 | 208.0 | 265.0 |
| **Base MSRP** | 4478.0 | 52446.504020 | 22510.219260 | 28500.0 | 34600.0 | 52900.0 | 69900.0 | 845000.0 |

- 1.The max cars are from year 2020.
- 2.The minimum no. of cars are from year 2008
- 3.The average electric range of cars is 116 miles.
- 4.The maximum and minimum ranges of the car are 265 and 11 miles respectively.
- 5.The mean Base MSRP of the car is USD 52446.504.
- 6.The max and min Base MSRP of the car are USD 845000 and USD 28500.

## 8. Summary of Categorical columns

```
#summary of categorical column
df.describe(include=object).T
```

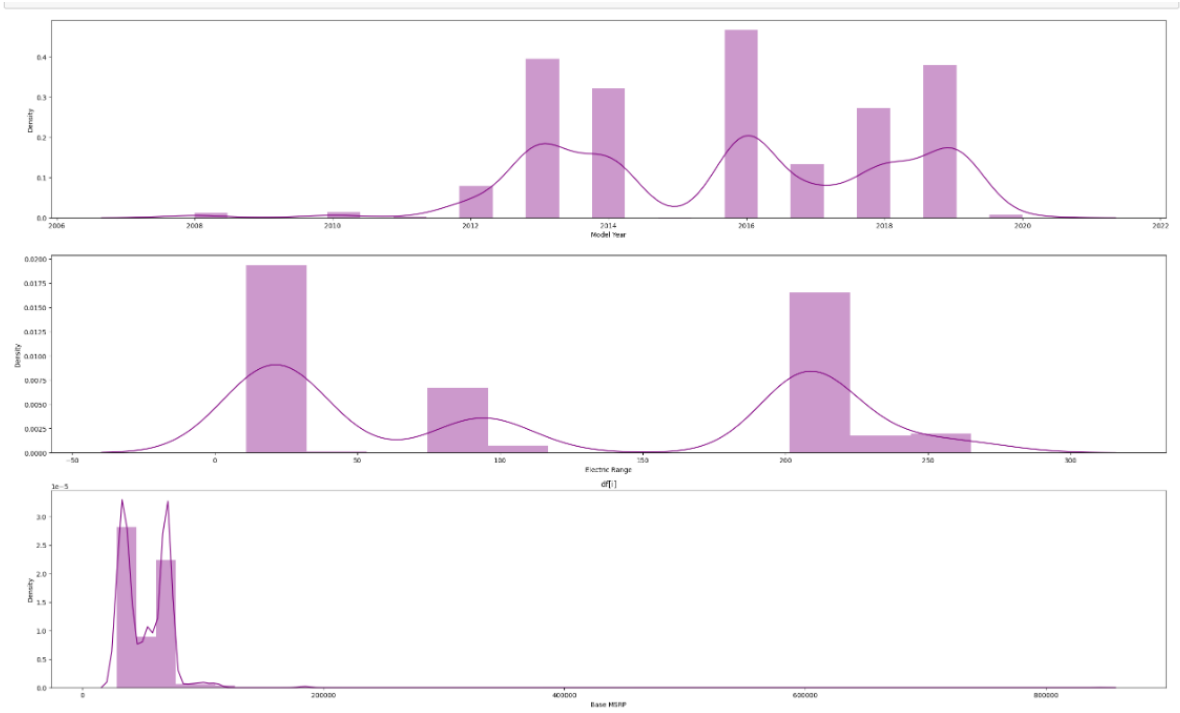
| | count | unique | top | freq |
|---|---|---|---|---|
| **City** | 4478 | 258 | SEATTLE | 753 |
| **State** | 4478 | 8 | WA | 4466 |
| **Make** | 4478 | 13 | TESLA | 1751 |
| **Model** | 4478 | 25 | MODEL S | 1690 |
| **Electric Vehicle Type** | 4478 | 2 | Battery Electric Vehicle (BEV) | 2625 |
| **Clean Alternative Fuel Vehicle (CAFV) Eligibility** | 4478 | 2 | Clean Alternative Fuel Vehicle Eligible | 2773 |

- 1.There are 258 unique cities in the dataset.
- 2.There are 8 unique states in the dataset.
- 3.The dataset contains 25 different models.
- 4.There are 13 different make of the cars.

## 9. Checking the Skewness

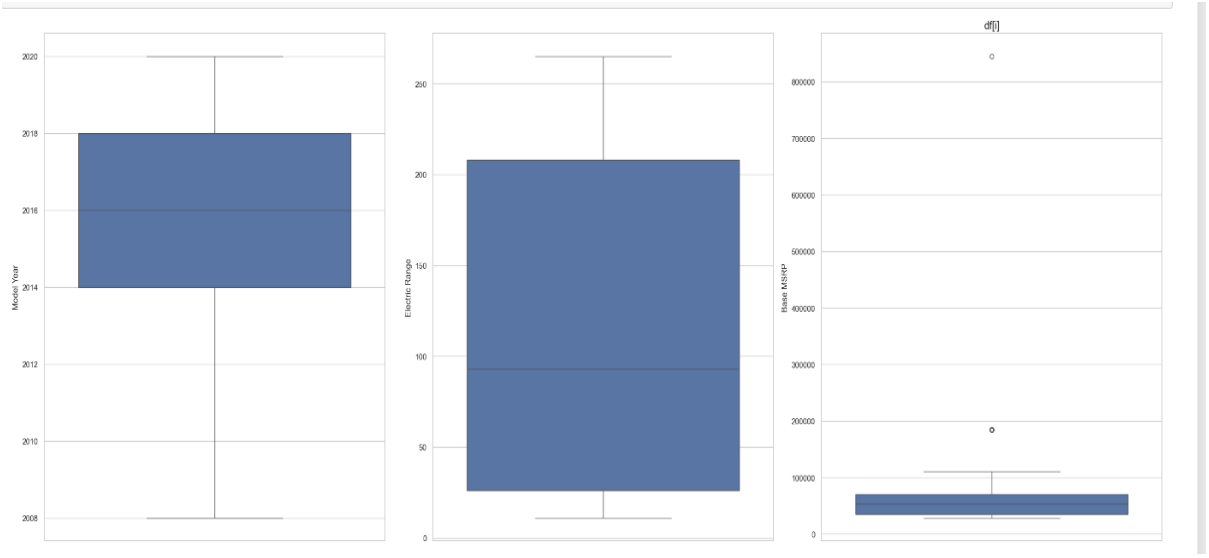
```
from scipy.stats import skew
skewness1 = skew(df['Base MSRP'])
skewness2 = skew(df['Electric Range'])
skewness3 = skew(df['Model Year'])
print(f'Skewness of the column Base MSRP: {skewness1}')
print(f'Skewness of the column Electric Range: {skewness2}')
print(f'Skewness of the column Model Year: {skewness3}')
```

```
Skewness of the column Base MSRP: 10.358325940922008
Skewness of the column Electric Range: 0.14248164661221469
Skewness of the column Model Year: -0.21879122007865393
```



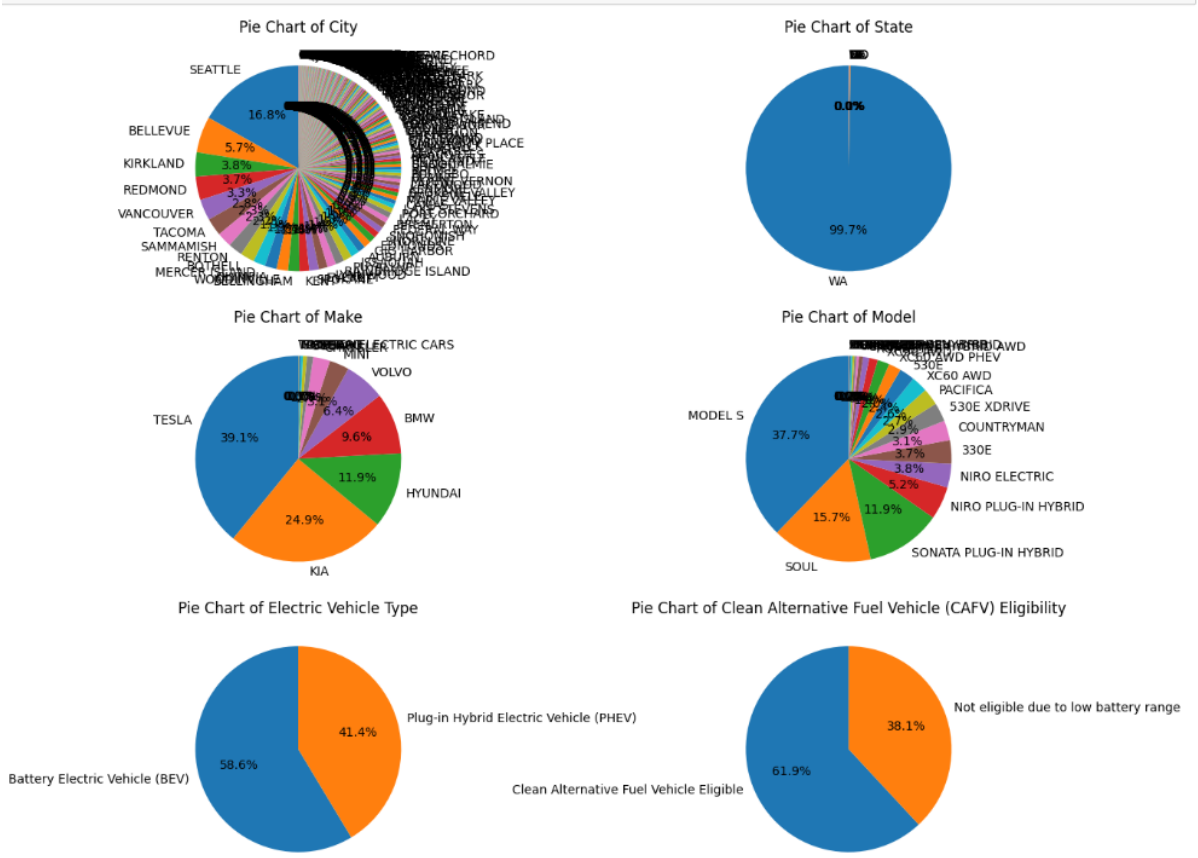

- Based on the above data,the column Base MSRP is right skewed as it values deviates too much from -0.5 to 0.5.
- Other two columns are Electric Range and Model Year are symmetric in nature

## 10. Checking for Outliers



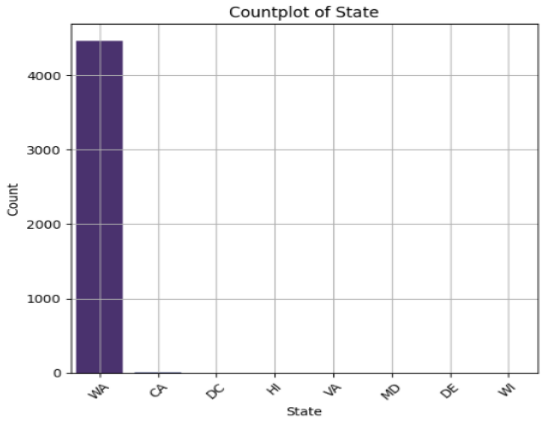

Only column Base MSRP has outliers that to only two positive outliers.

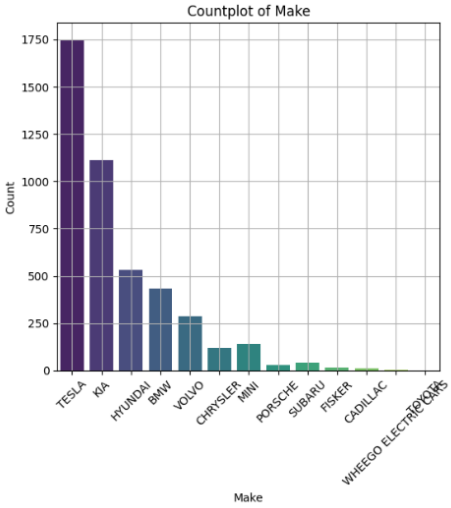
## 11. Analysis of Categorical Variables

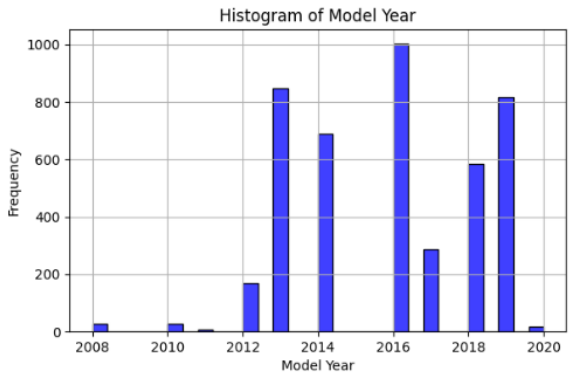


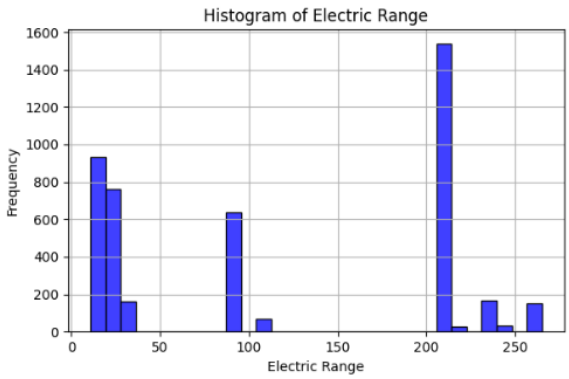

- 1. 57.6% of the vehicle are BEV type and 41.4% of vehicle are of PHEV type
- 2. 61.9% of vehicles are eligible for Clean Alternative Fuel Vehicle
- 3. 37.5% of vehicles are Model S
- 4. A total of 39.1% of cars are of Tesla.
- 5. 99.7% of vehicles are of WA state.

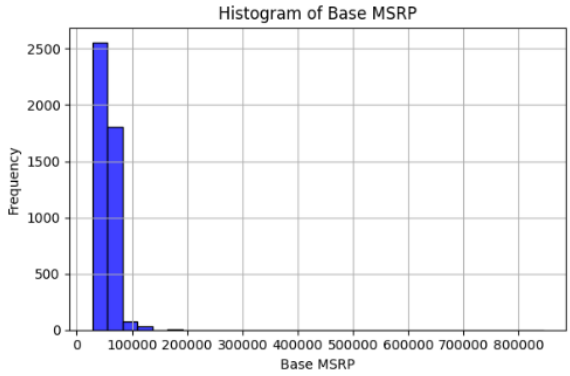
## 12. Other Plots

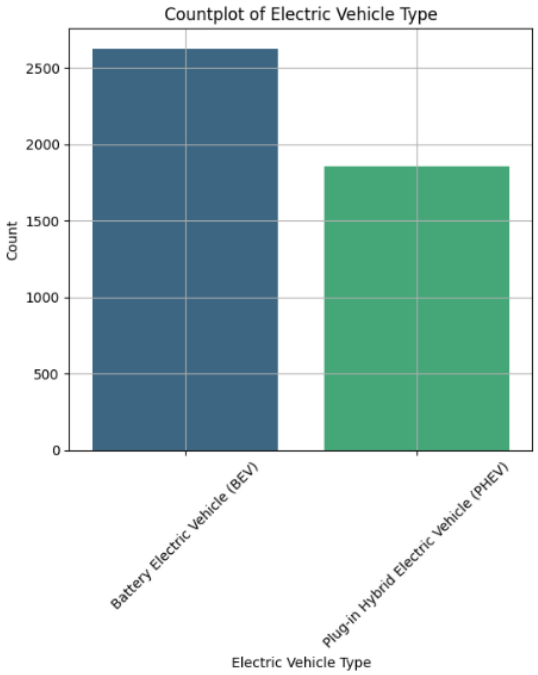


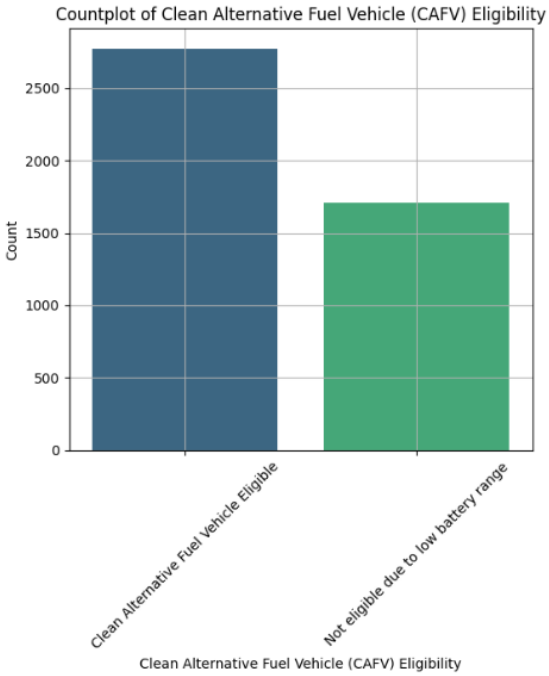





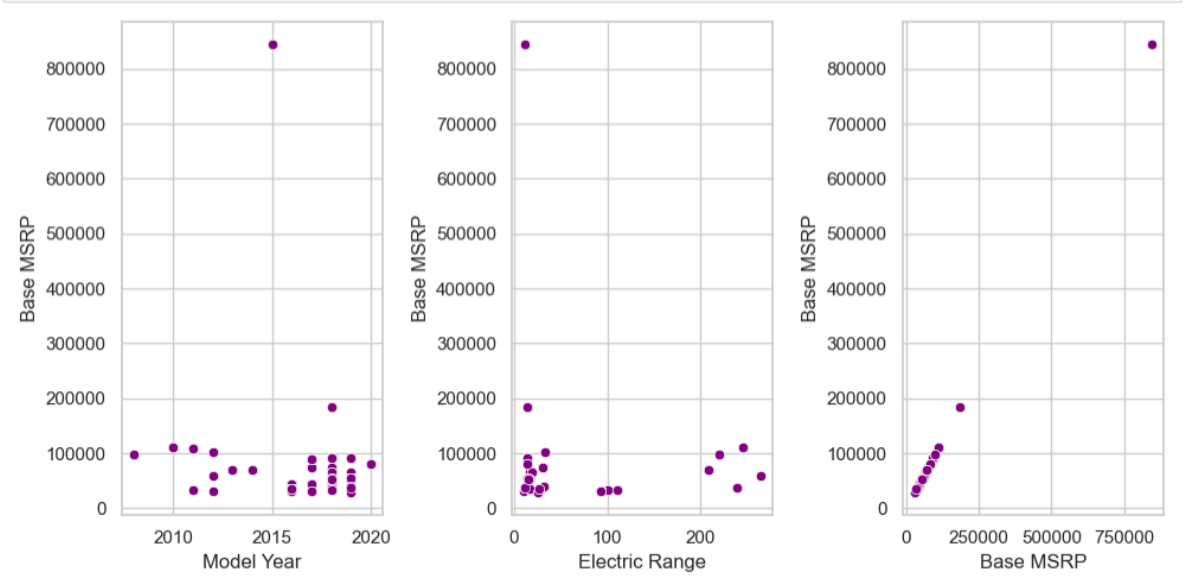




- Most of the vehicle are Battery Electric Vehicle.
- More than 2500 vehicles are BEV type and less than 2000 are PHEV type.
- More than 2500 vehicles are of Clean Alternative Fuel Vehicle Eligible type.
- Nearly 1750 vehicles are not eligibe for CAFV due ti low battery range.
- Tesla has most no. of cars around 1750 in the dataset.
- Out of all models Tesla's model S remains the best selling car.
- Most EV cars were sold in state of WA.

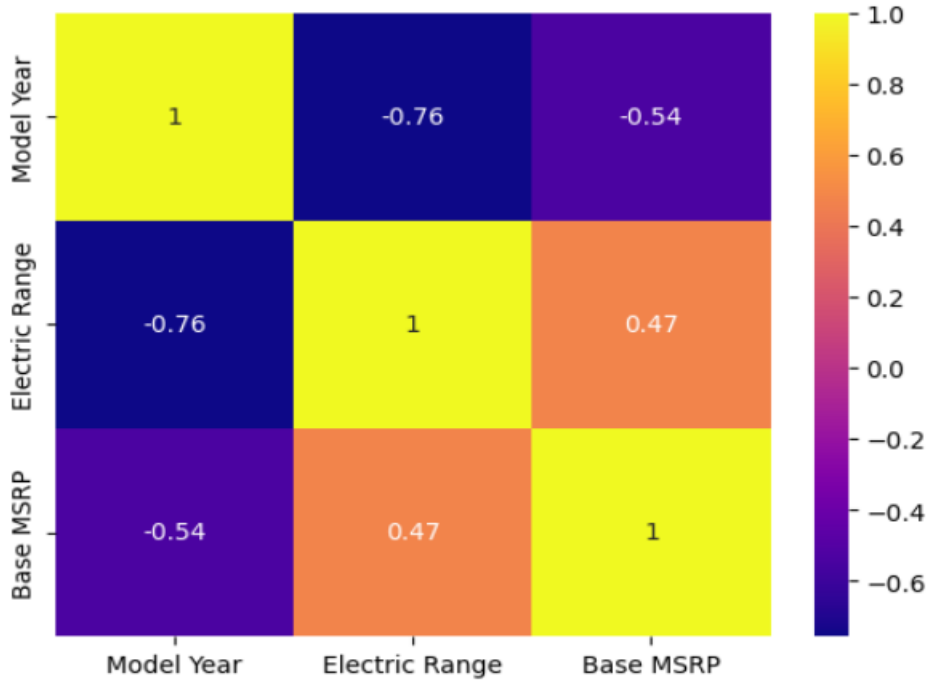
## 13. Scatter Plot




- A car launched in 2015 is priced highest over 800000 USD and has a range of less than 50 miles.
- Most of the cars are priced below 100000 USD.
- Year 2018 saw the launch of most cars

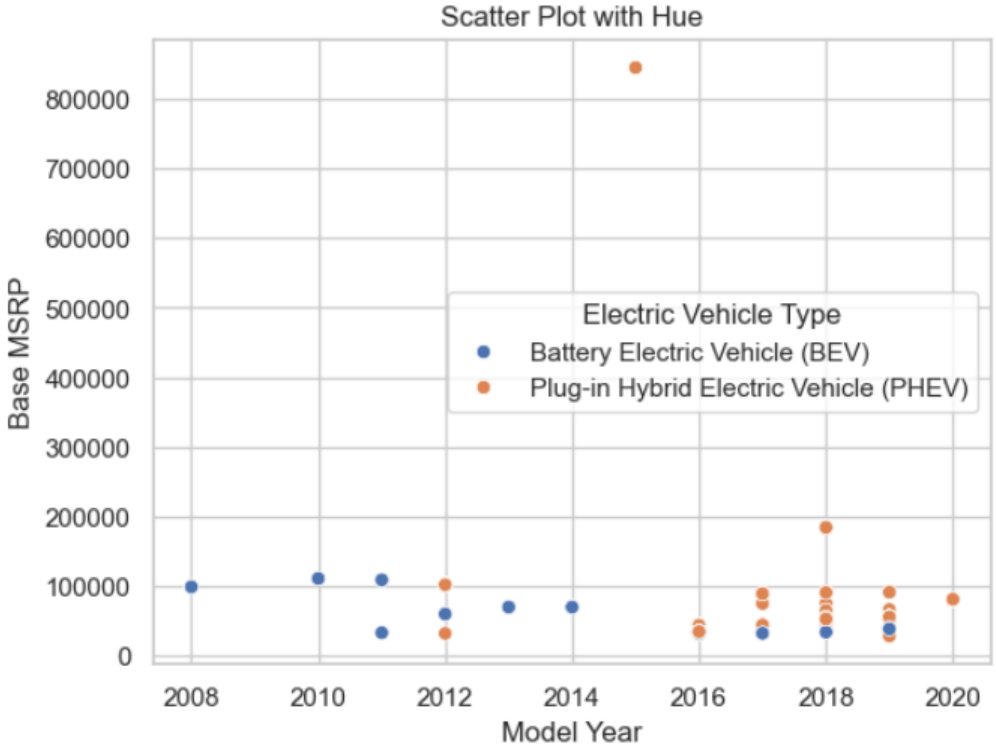
## 14. Correlation

| | Model Year | Electric Range | Base MSRP |
|---|---|---|---|
| **Model Year** | 1.000000 | -0.756519 | -0.542602 |
| **Electric Range** | -0.756519 | 1.000000 | 0.474184 |
| **Base MSRP** | -0.542602 | 0.474184 | 1.000000 |

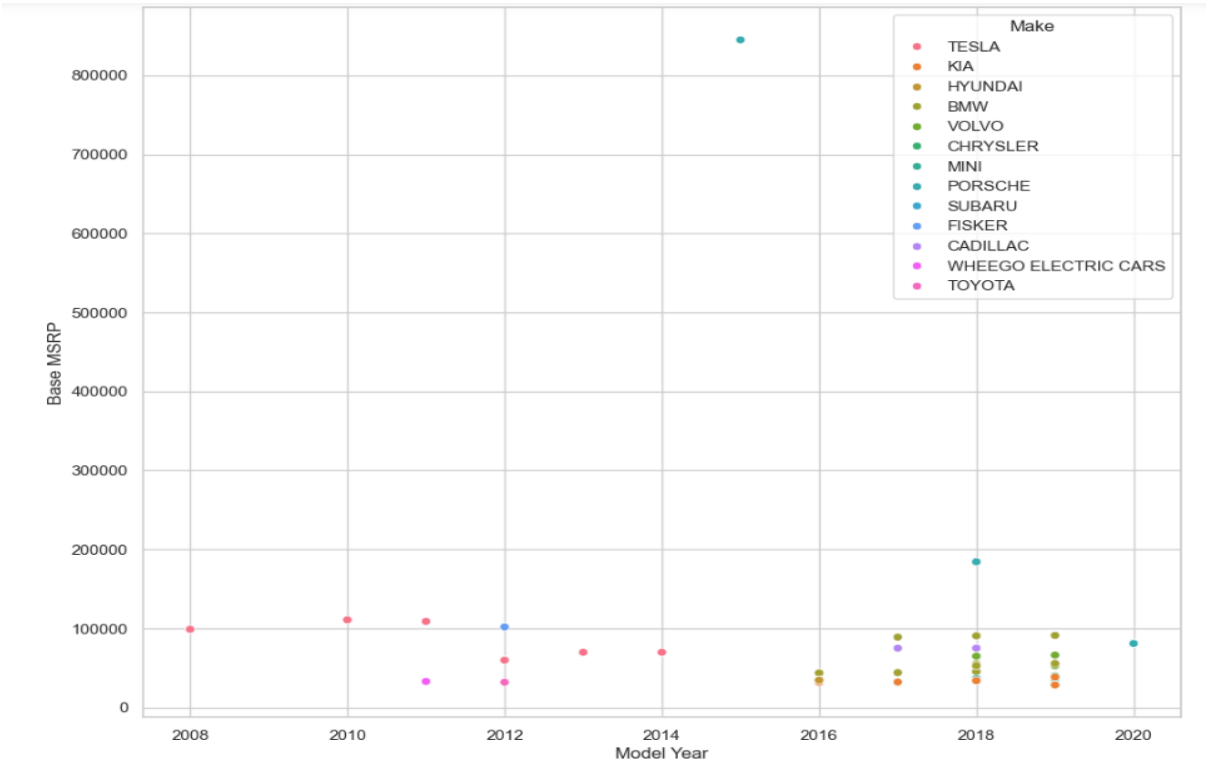



Electric Range and Model Year columns are highly negatively correlated

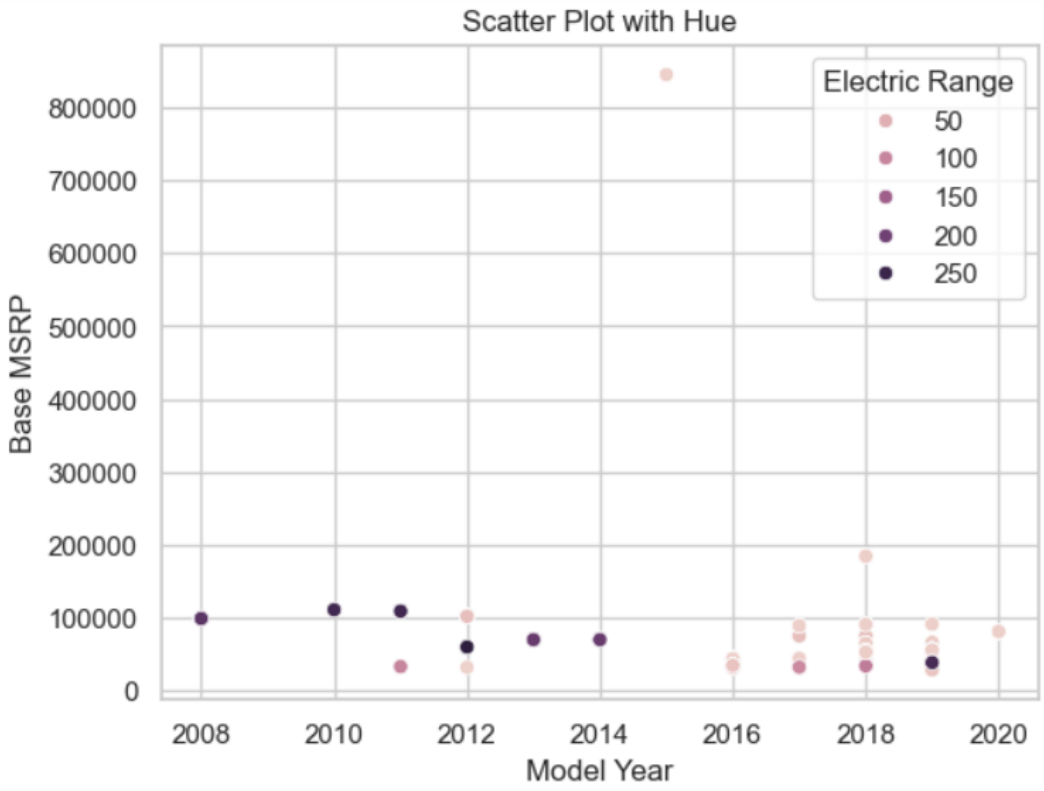
## 15. Multivariate Analysis




- Most Plug-in Hybrid Electric vehicles were launched in year 2018




- In year 2008,the first electric priced at 100000 USD was launched
- In year 2015,The costliest car priced above 800000 USD




- The first car launched in year 2008 was priced at 100000 USD.
- Most of cars sold were priced below 100000 USD

## 16. Outlier Treatment

```
q1=df.quantile(0.25)
q3=df.quantile(0.75)
iqr=q3-q1
ul=q3+1.5*iqr #upper limit
ll=q1-1.5*iqr  #lower limit

df_cleaned=df[~((df<(ll))|(df>(ul))).any(axis=1)]
df_cleaned
```

```
df_n=df_cleaned
```

```
#Shape after outlier treatment
df_n.shape
```

```
(4467, 9)
```

```
df.shape
```

```
(4478, 9)
```

```
df_num=df_n.select_dtypes(include=np.number)
df_cat=df_n.select_dtypes(exclude=np.number)
```

## 17. Scaling the data

```
from sklearn.preprocessing import StandardScaler

# Assuming df_n is your DataFrame
numerical_columns = ['Model Year', 'Electric Range', 'Base MSRP']
df_num1 = df_n[numerical_columns]

# Separate the 'Model Year' column for later concatenation
model_year = df_num['Model Year']
df_num1 = df_num.drop(columns=['Model Year', 'Base MSRP'])  # Exclude the target variable and 'Model Year' column

# Standardize the numerical columns
scaler = StandardScaler()
X_scaled = scaler.fit_transform(df_num1)

# Create a DataFrame with the scaled features
df_scaled = pd.DataFrame(X_scaled, columns=df_num1.columns)

# Concatenate the 'Model Year' column to the scaled DataFrame
df_scaled['Model Year'] = model_year.reset_index(drop=True)

df_scaled
```

## 18. Encoding

```
df_en= pd.get_dummies(df_cat,drop_first=True)
```

```
df_scaled.reset_index(drop=True, inplace=True)
df_en.reset_index(drop=True, inplace=True)
```

```
df_f=pd.concat([df_scaled,df_en],axis=1)
df_f
```

## 19. Modelling

```
#Step 1:Separate x and y
X = df_f.drop(columns=['Electric Range'])
y=df_num['Base MSRP']
```

```
#Step 2: Split the data in train and test
from sklearn.model_selection import train_test_split

X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.30,random_state=42)
```

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(3126, 301)
(1341, 301)
(3126,)
(1341,)
```

```
#Step 3:Calling the model
from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score,mean_squared_error

lr=LinearRegression()

#Step 4: Fit the model
model_lr=lr.fit(X_train,y_train)

#Step 5:Use Prediction
pred_train=model_lr.predict(X_train)
pred_test=model_lr.predict(X_test)

r2_train=r2_score(y_train,pred_train)
r2_test=r2_score(y_test,pred_test)

print("R2 Train: ",r2_train)
print("R2 Test: ",r2_test)

#Step 6:Model Performance of model
print("MSE Train: ",mean_squared_error(y_train,pred_train))
print("MSE Test: ",mean_squared_error(y_test,pred_test))

print("RMSE Train: ",np.sqrt(mean_squared_error(y_train,pred_train)))
print("RMSE Test: ",np.sqrt(mean_squared_error(y_test,pred_test)))
```

## Output

```
R2 Train:  0.9936806538339503
R2 Test:  -1.2065987725565532e+19
MSE Train:  2046940.467490403
MSE Test:  4.0734043546859425e+27
RMSE Train:  1430.7132722842837
RMSE Test:  63823227391647.49
```

- Based on above scores,linear regression is not performing well

## Using other models

```
from sklearn.ensemble import RandomForestRegressor
```

```
#doing the RF with original data with all features
rfo=RandomForestRegressor()

# build the model and find the r2score with RMSE

rfo = rfo.fit(X_train,y_train)

pred_train_rfo = rfo.predict(X_train)

pred_test_rfo=rfo.predict(X_test)
```

```
r2_train=r2_score(y_train,pred_train_rfo)
r2_test=r2_score(y_test,pred_test_rfo)

print("R2 Train: ",r2_train)
print("R2 Test: ",r2_test)
# Mean Squared Error
# print("MSE Train :", mean_squared_error(ytrain,pred_train_rf))
# print("MSE Test  :", mean_squared_error(ytest,pred_test_rf))


print("RMSE Train :", np.sqrt(mean_squared_error(y_train,
                                                 pred_train_rfo)))

print("RMSE Test  :", np.sqrt(mean_squared_error(y_test,
                                                 pred_test_rfo)))
```

```
R2 Train:  0.9994985165454182
R2 Test:  0.9911029388015473
RMSE Train : 403.03692565283814
RMSE Test  : 1733.087983337699
```

- Based on R2 score ,it can be seen that model is learning as well as performing well on seen and unseen data.

## Now using Grid Search CV

```
%%time
from sklearn.model_selection import GridSearchCV

param_grid = {
    'n_estimators': [50, 100, 200, 300],
    'max_depth': [None, 10, 20, 30],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'max_features': ['auto', 'sqrt', 'log2'],
    'bootstrap': [True, False]
}

grid_search = GridSearchCV(RandomForestRegressor(), param_grid, cv=5)
grid_search.fit(X_train, y_train)

best_rf = grid_search.best_estimator_
```

```
CPU times: total: 8min 26s
Wall time: 25min 57s
```

```
print(best_rf)
```

```
RandomForestRegressor(bootstrap=False, max_features='sqrt', n_estimators=300)
```

```
%%time
from sklearn.ensemble import RandomForestRegressor

# Create a RandomForestRegressor with the best hyperparameters
best_rf = RandomForestRegressor(n_estimators=300,max_features='sqrt',bootstrap=False)

# Fit the model to your training data
best_rf.fit(X_train, y_train)

# Make predictions
pred_train_rf = best_rf.predict(X_train)
pred_test_rf = best_rf.predict(X_test)

# Evaluate the model with R-squared and RMSE
r2_train = r2_score(y_train, pred_train_rf)
r2_test = r2_score(y_test, pred_test_rf)

print("R2 Train:", r2_train)
print("R2 Test:", r2_test)

rmse_train = np.sqrt(mean_squared_error(y_train, pred_train_rf))
rmse_test = np.sqrt(mean_squared_error(y_test, pred_test_rf))

print("RMSE Train:", rmse_train)
print("RMSE Test:", rmse_test)
```

```
R2 Train: 0.999982481218315
R2 Test: 0.9959261297249378
RMSE Train: 75.33008946375027
RMSE Test: 1172.7377897153938
CPU times: total: 1.25 s
Wall time: 2.68 s
```

- The R2 scores are very close to 1, indicating that the model explains the variance in the target variable extremely well, both on the training and test sets.
- The Root Mean Squared Error (RMSE) values are low, especially on the training set, suggesting that the model's predictions are close to the actual values. The RMSE on the test set is also reasonable, given the scale of the target variable.
- The model generalizes well to unseen data, as evidenced by the minimal difference between the training and test set performance.
- The R2 scores indicate that the model captures almost all the variability in the target variable.
- The RMSE values, particularly on the training set, are low, indicating good predictive accuracy.

# 2nd Data Set-EV Stats-1.csv

## 1.Reading the data

```
df=pd.read_csv('EV Stats-1.csv')
df.head()
```

## 2. Checking the info

```
#checking the info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 35 entries, 0 to 34
Data columns (total 10 columns):
 #   Column                                                    Non-Null Count  Dtype
---  ------                                                    --------------  -----
 0   Sl. No                                                    34 non-null     object
 1   State                                                     34 non-null     object
 2   Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules  31 non-null     float64
 3   Two Wheelers (Category L2 (CMVR))                         31 non-null     float64
 4   Two Wheelers (Max power not exceeding 250 Watts)          31 non-null     float64
 5   Three Wheelers (Category L5 slow speed as per CMVR)       34 non-null     object
 6   Three Wheelers (Category L5 as per CMVR)                  31 non-null     float64
 7   Passenger Cars (Category M1 as per CMVR)                  31 non-null     float64
 8   Buses                                                     31 non-null     float64
 9   Total in state                                            31 non-null     float64
dtypes: float64(7), object(3)
memory usage: 2.9+ KB
```

There are 7 numerical and 3 categorical columns

## 3. Checking the shape and details about columns

```
#Shape of data
df.shape
```

```
(35, 10)
```

There are 35 rows and 10 columns

1. Sl. No:

- Description: Serial number or index of the data entries.

2. State:

- Description: The name of the state.

3. Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules):

- Description: Count of two-wheelers in category L1 & L2 as per Central Motor Vehicles Rules.

4. Two Wheelers (Category L2 (CMVR)):

- Description: Count of two-wheelers in category L2 as per Central Motor Vehicles Rules.

5. Two Wheelers (Max power not exceeding 250 Watts):

- Description: Count of two-wheelers with a maximum power not exceeding 250 Watts.

6. Three Wheelers (Category L5 slow speed as per CMVR):

- Description: Count of three-wheelers in category L5 with slow speed as per Central Motor Vehicles Rules.

7. Three Wheelers (Category L5 as per CMVR):

- Description:Count of three-wheelers in category L5 as per Central Motor Vehicles Rules.

8. Passenger Cars (Category M1 as per CMVR):

- Description: Count of passenger cars in category M1 as per Central Motor Vehicles Rules.

9. Buses:

- Description: Count of buses.

10. Total in state:

- Description: Total count of vehicles in the state.

## 4. Null values after data cleaning

```
#Based on the obeservation of the data.We need to drop rows from 31 to 35
df=df[:31]
```

```
#Checking the null values
df.isnull().sum()
```

```
Sl. No                                                              0
State                                                               0
Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules  0
Two Wheelers (Category L2 (CMVR))                                   0
Two Wheelers (Max power not exceeding 250 Watts)                    0
Three Wheelers (Category L5 slow speed as per CMVR)                 0
Three Wheelers (Category L5 as per CMVR)                            0
Passenger Cars (Category M1 as per CMVR)                            0
Buses                                                               0
Total in state                                                      0
dtype: int64
```

There are no null values present

## 5. Summary of Numerical and Categorical Column

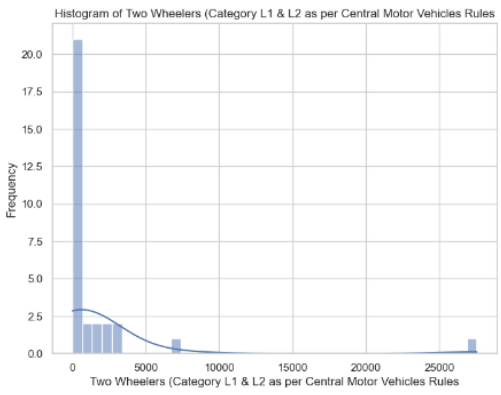
| | count | mean | std | min | 25% | 50% | 75% | max |
|---|---|---|---|---|---|---|---|---|
| **Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules** | 31.0 | 1777.354839 | 5000.160994 | 0.0 | 22.0 | 463.0 | 1089.5 | 27549.0 |
| **Two Wheelers (Category L2 (CMVR))** | 31.0 | 907.677419 | 2518.917378 | 0.0 | 19.0 | 228.0 | 767.5 | 14069.0 |
| **Two Wheelers (Max power not exceeding 250 Watts)** | 31.0 | 7260.516129 | 20008.445495 | 0.0 | 94.0 | 2148.0 | 6639.0 | 112538.0 |
| **Three Wheelers (Category L5 slow speed as per CMVR)** | 31.0 | 25.096774 | 73.990700 | 0.0 | 0.0 | 0.0 | 7.0 | 389.0 |
| **Three Wheelers (Category L5 as per CMVR)** | 31.0 | 46.451613 | 131.491150 | 0.0 | 0.0 | 1.0 | 37.5 | 720.0 |
| **Passenger Cars (Category M1 as per CMVR)** | 31.0 | 6811.032258 | 18941.726543 | 1.0 | 236.5 | 997.0 | 5487.5 | 105571.0 |
| **Buses** | 31.0 | 1.741935 | 6.016465 | 0.0 | 0.0 | 0.0 | 0.0 | 27.0 |
| **Total in state** | 31.0 | 16829.870968 | 46327.212405 | 6.0 | 665.0 | 4234.0 | 14951.5 | 260863.0 |

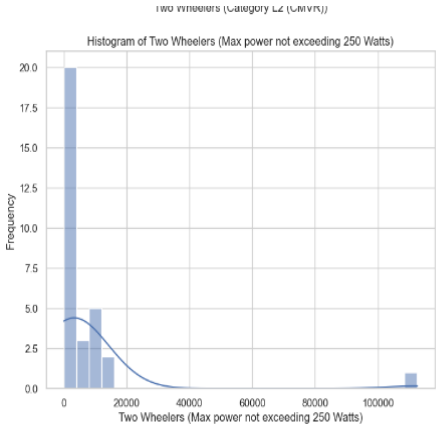
| | Sl. No | State |
|---|---|---|
| **count** | 31 | 31 |
| **unique** | 31 | 31 |
| **top** | 1 | Andhra Pradesh |
| **freq** | 1 | 1 |

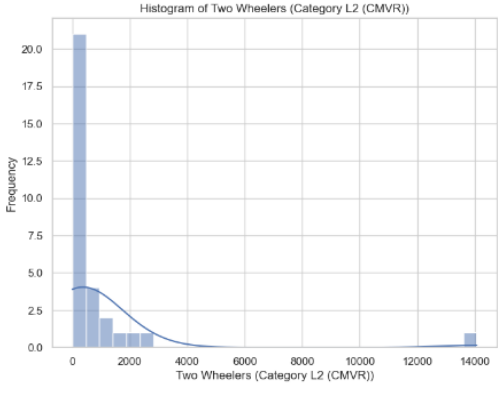
- 1.There are maximum of 27 busus.
- 2.There are max of 14069 Two Wheelers of Category L2 (CMVR).
- 3.There are max of 112538 Two Wheelers of Max power not exceeding 250 Watts
- 4.There are total of 260863 vehicles in all states combined.

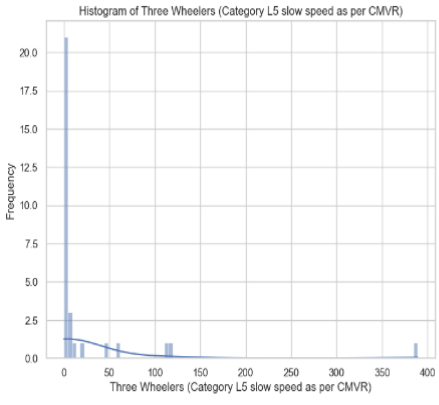
---

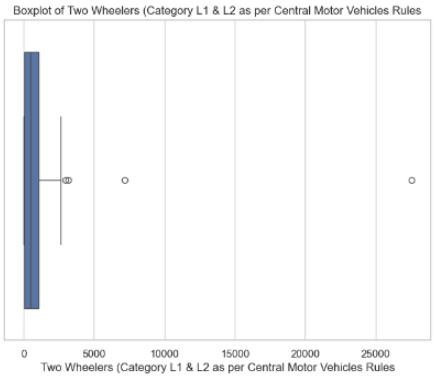
## 6. Plots

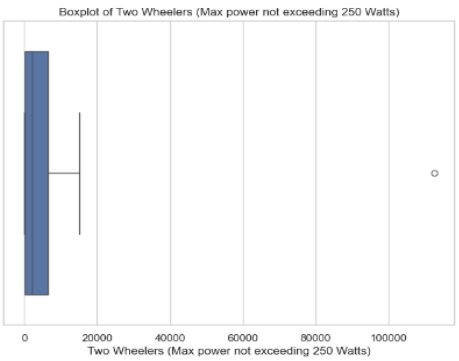


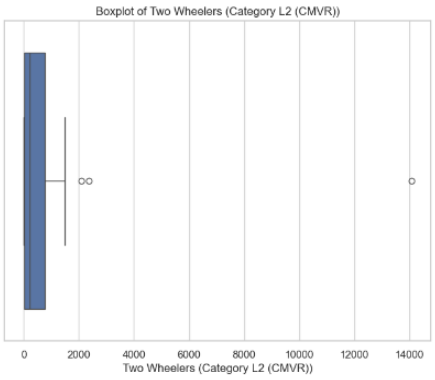



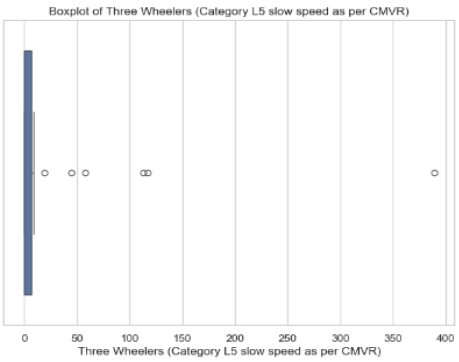



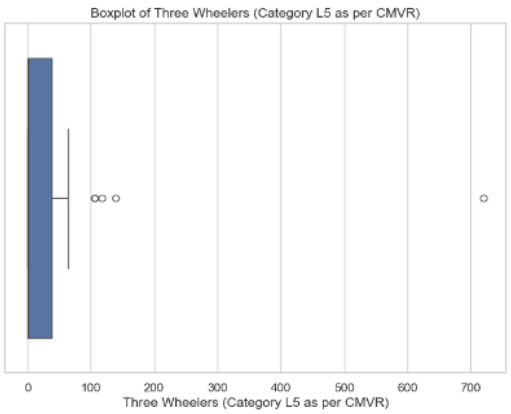



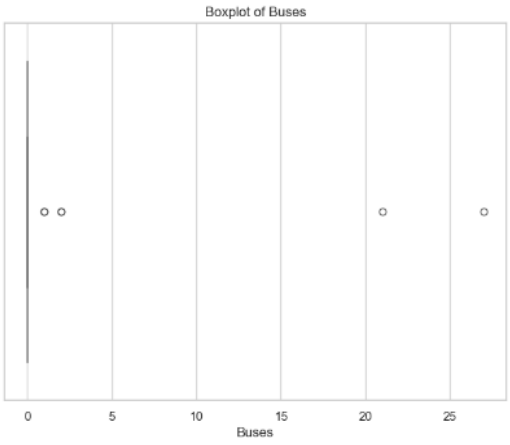

Boxplot of Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules)
0
5000
10000
15000
20000
25000
Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules)

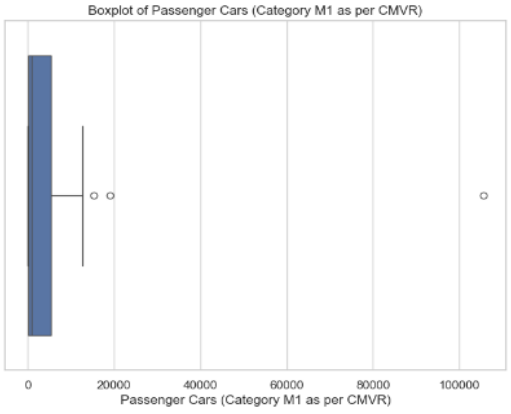

Boxplot of Two Wheelers (Max power not exceeding 250 Watts)
0
20000
40000
60000
80000
100000
Two Wheelers (Max power not exceeding 250 Watts)

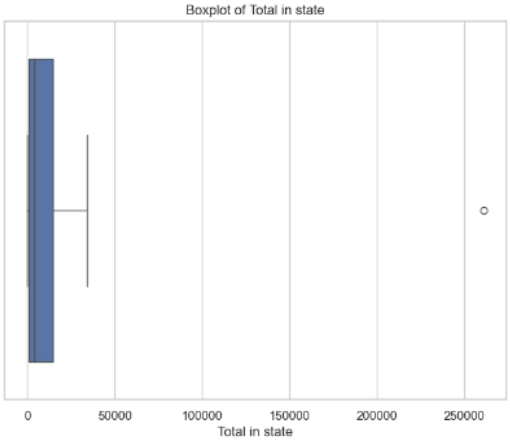

Boxplot of Two Wheelers (Category L2 (CMVR))
0
2000
4000
6000
8000
10000
12000
14000
Two Wheelers (Category L2 (CMVR))


Boxplot of Three Wheelers (Category L5 slow speed as per CMVR)
0
50
100
150
200
250
300
350
400
Three Wheelers (Category L5 slow speed as per CMVR)


Boxplot of Three Wheelers (Category L5 as per CMVR)
0
100
200
300
400
500
600
700
Three Wheelers (Category L5 as per CMVR)

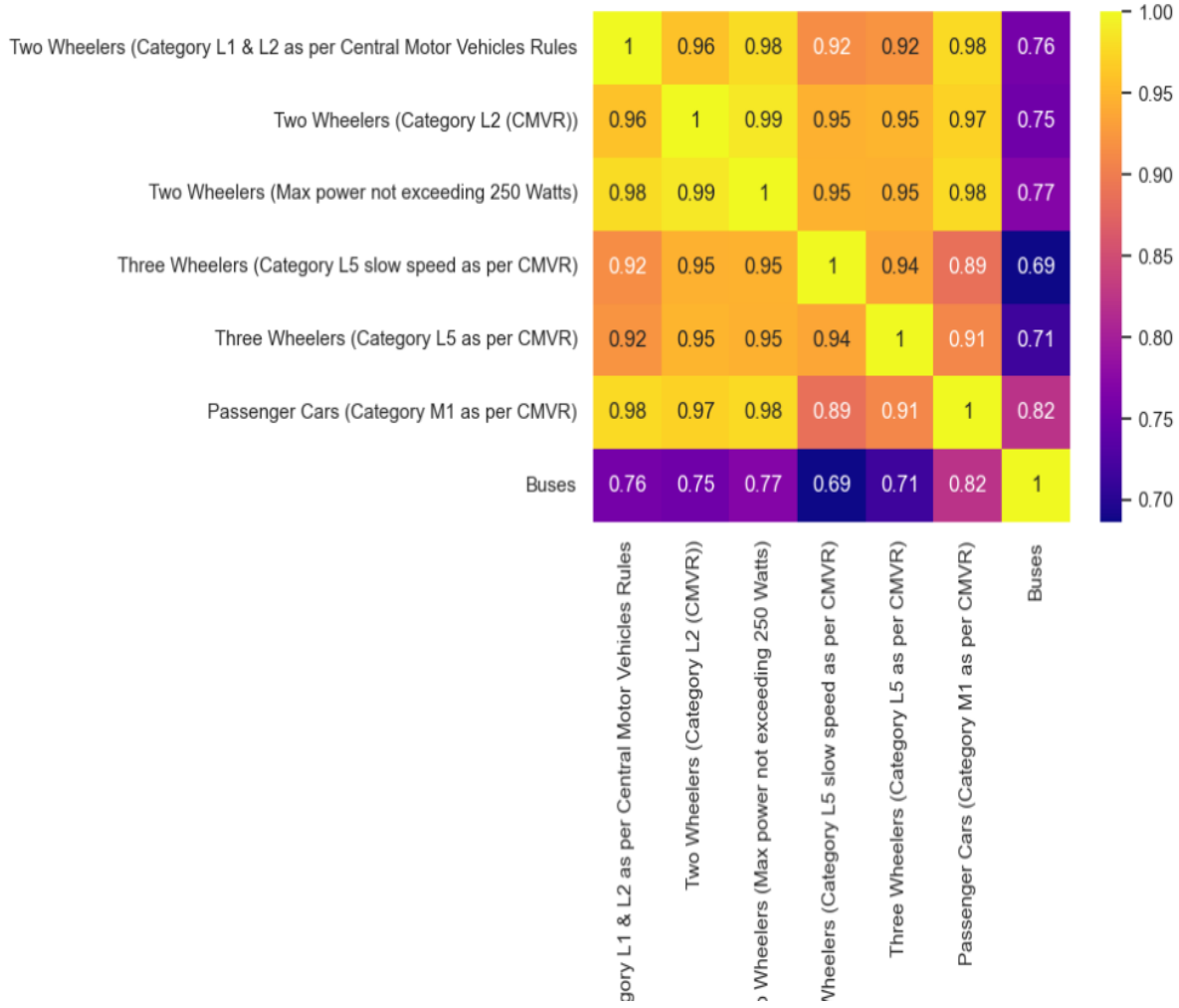

Boxplot of Buses
0
5
10
15
20
25
Buses


Boxplot of Passenger Cars (Category M1 as per CMVR)
0
20000
40000
60000
80000
100000
Passenger Cars (Category M1 as per CMVR)


Boxplot of Total in state
0
50000
100000
150000
200000
250000
Total in state

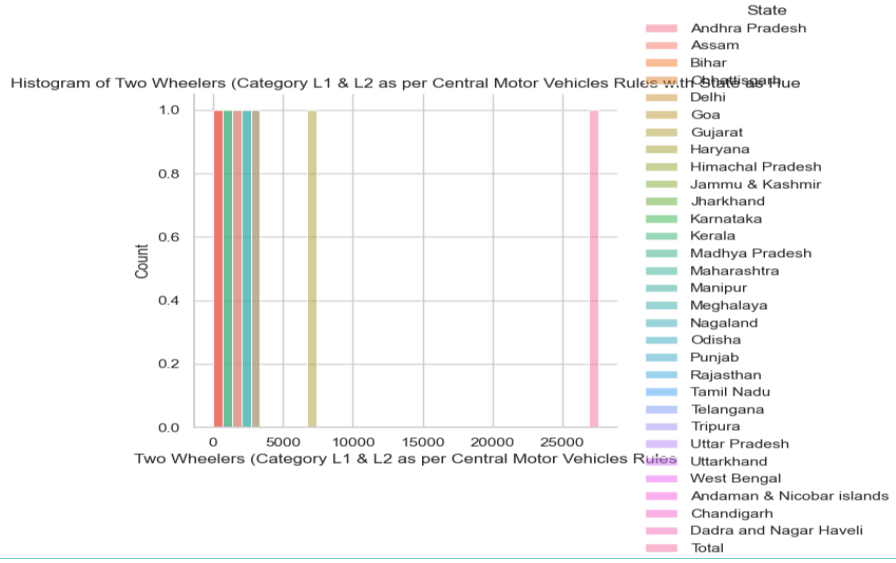
- 1.Most of the state does not have any buses.
- 2.There are only 1 state which has more than 20 buses.
- 3.There is only 2 state which have passenger car of EV type closer to 20000.
- 4.There are only 3 states which have Three Wheelers of Category L5 above 100.
- 5.There are only 2 states which have Three Wheelers of Category L5 slow speed above 100.
- 6.There are only 2 states which have Two Wheelers Category L2 above 2000.
- 7.There is only one state which have Two Wheelers (Category L1 and L2) above 5000.

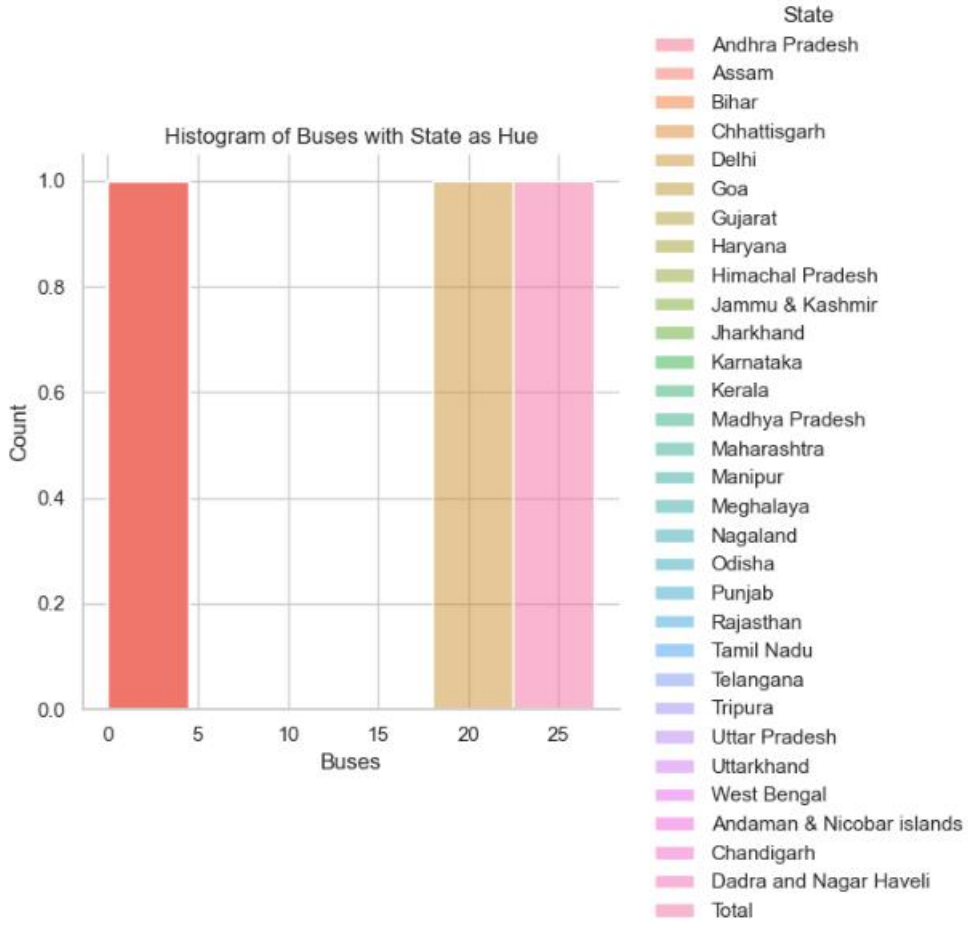
## 7. Heatmap

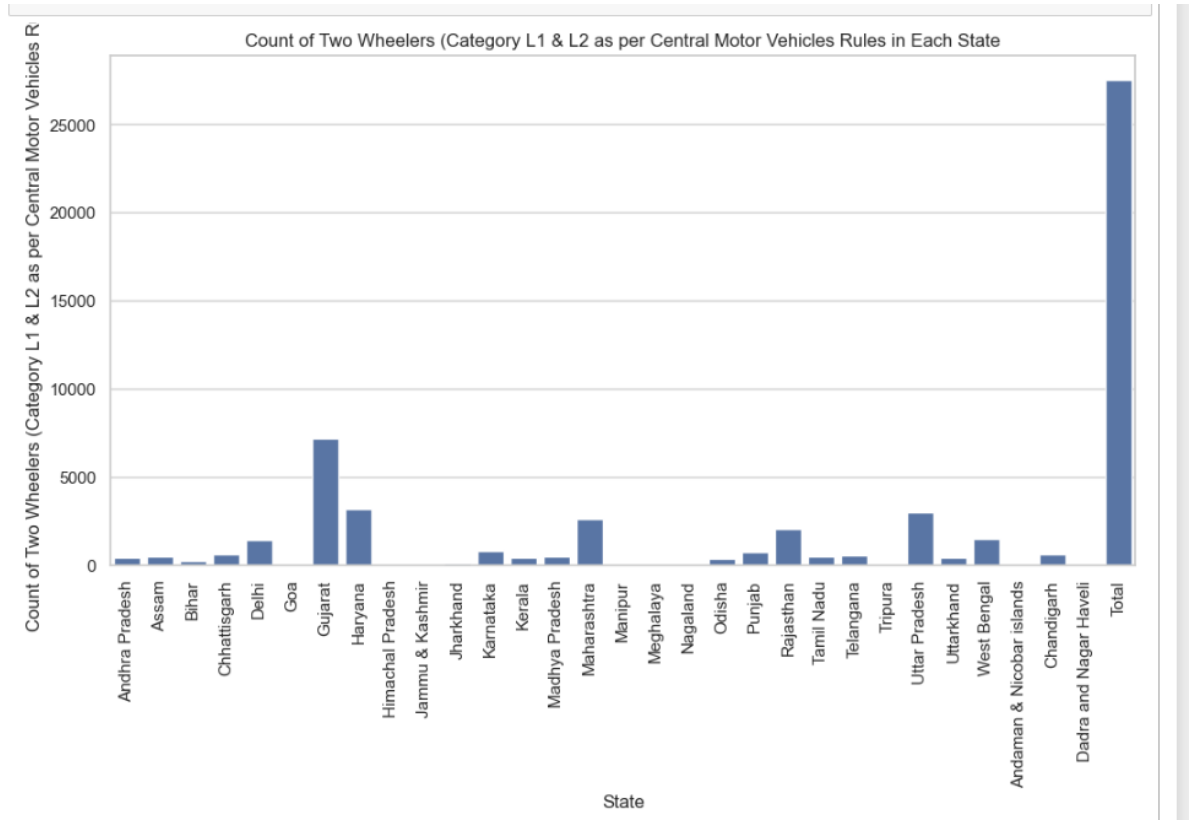


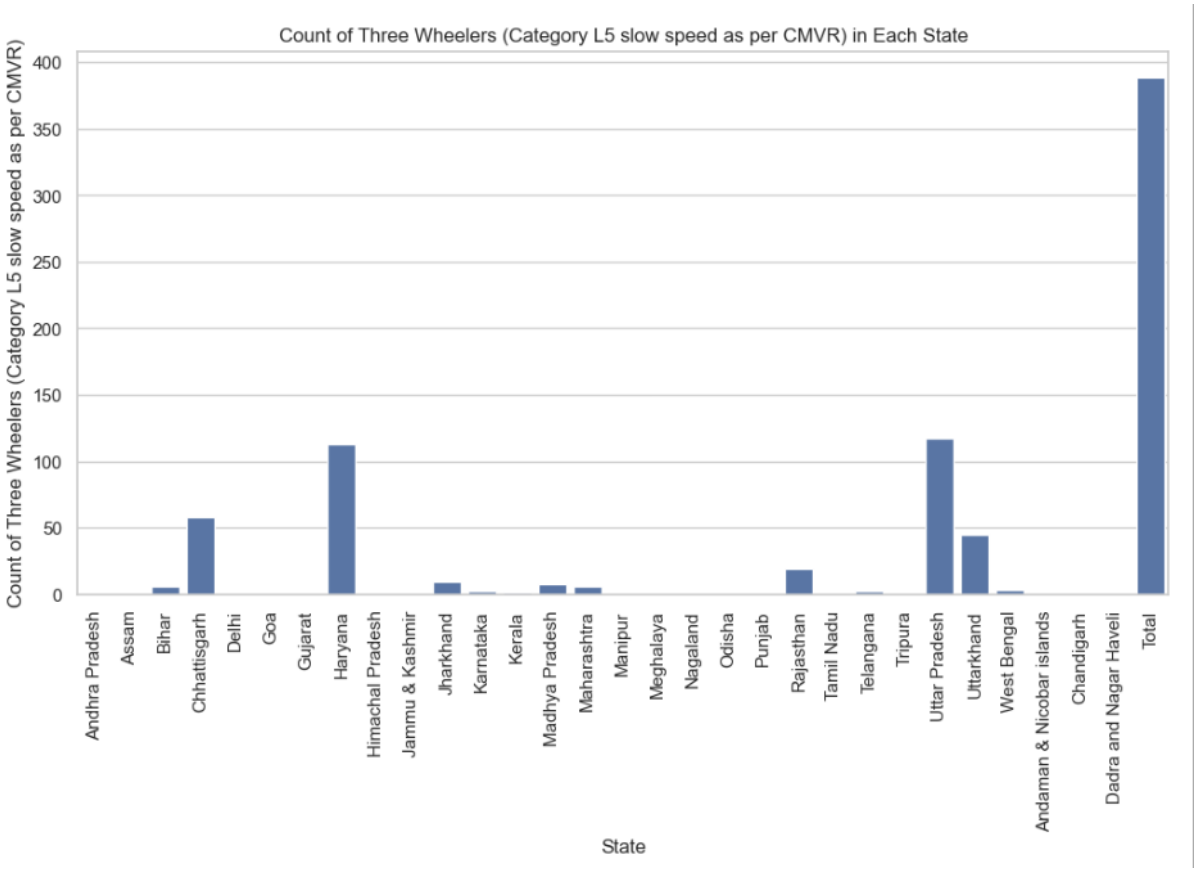

- 1.All the columns are highly correalted
- 2.The Buses column andThree Wheelers (Category L5 slow speed) is least highly coorelated to the tune of 0.69.
- 3.The column Two Wheelers (Max power not exceeding 250 Watts) and Two Wheelers (Category L2 (CMVR) has highest correaltion of 0.99

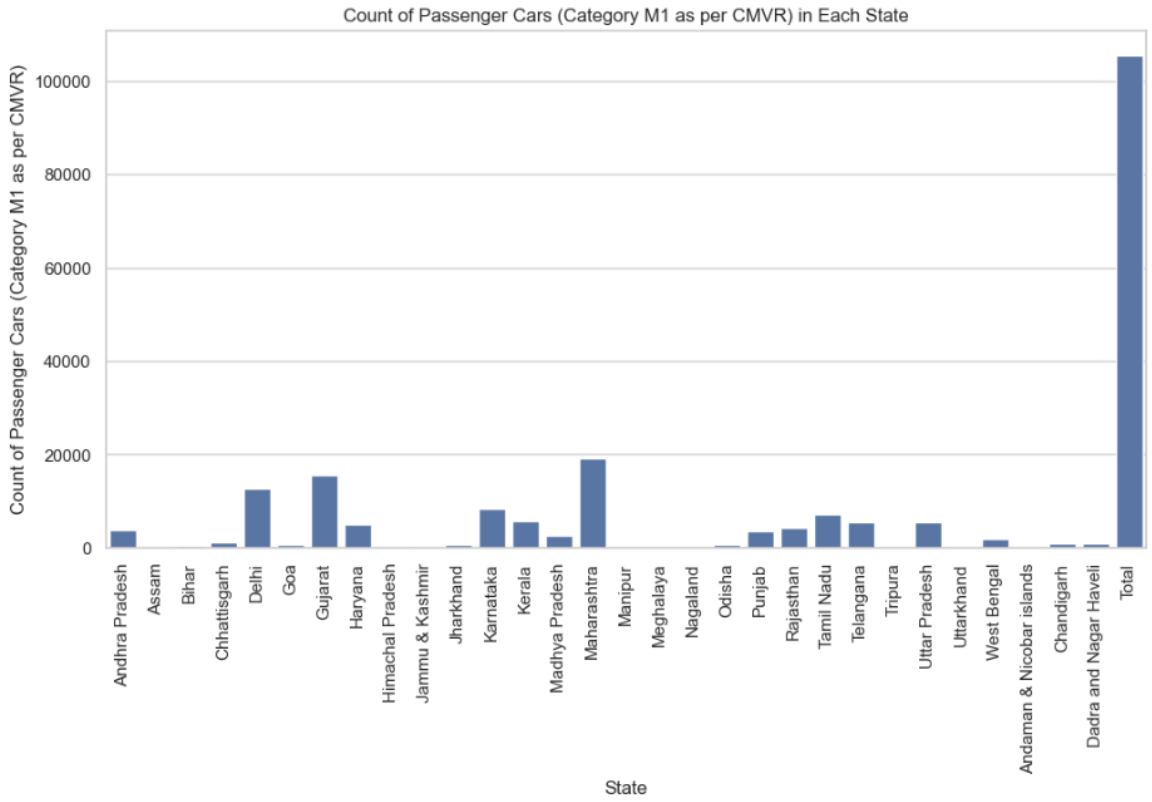
# **8.** Various Plots

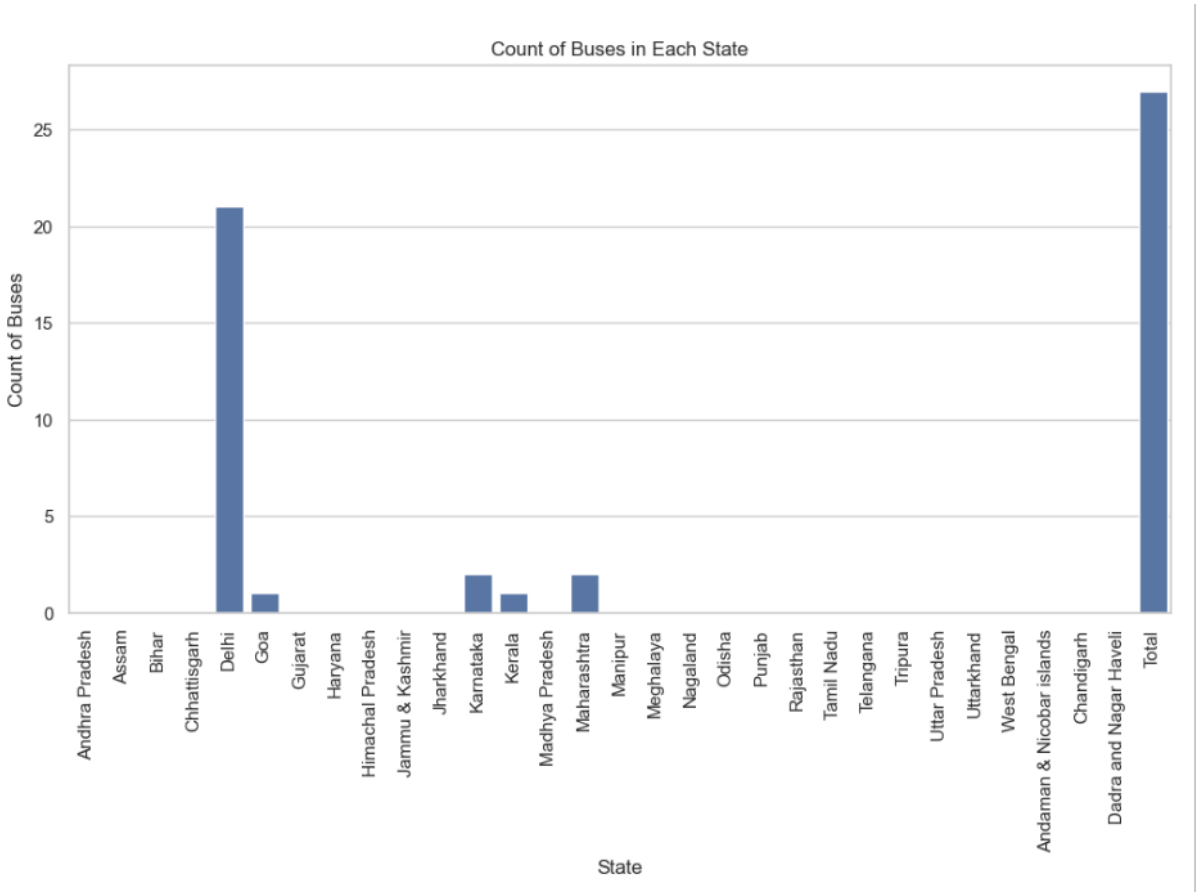


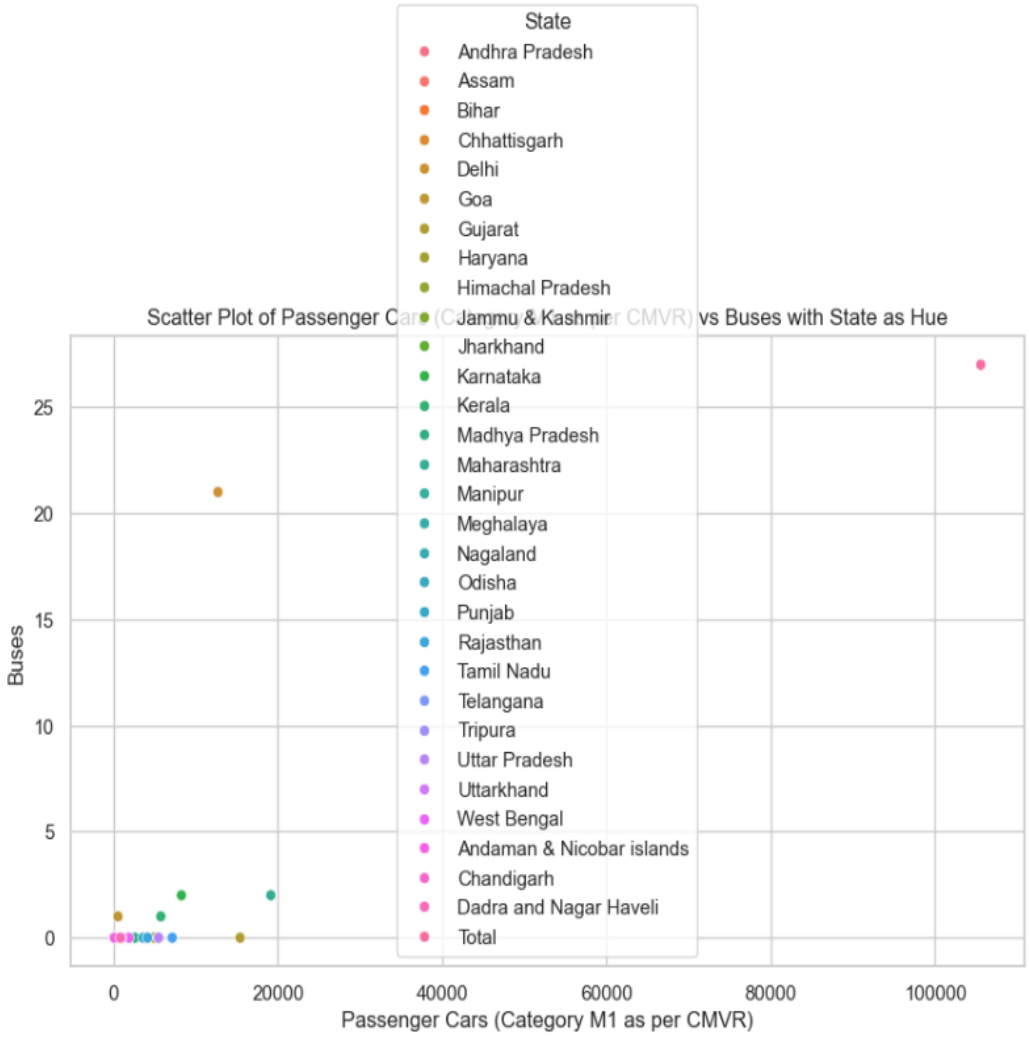

Count of Two Wheelers (Category L1 & L2 as per Central Motor Vehicles Rules in Each State
Count of Two Wheelers (Category L1 & L2 as per Central Motor Vehicles R
25000
20000
15000
10000
5000
0
Andhra Pradesh
Assam
Bihar
Chhattisgarh
Delhi
Goa
Gujarat
Haryana
Himachal Pradesh
Jammu & Kashmir
Jharkhand
Karnataka
Kerala
Madhya Pradesh
Maharashtra
Manipur
Meghalaya
Nagaland
Odisha
Punjab
Rajasthan
Tamil Nadu
Telangana
Tripura
Uttar Pradesh
Uttarkhand
West Bengal
Andaman & Nicobar islands
Chandigarh
Dadra and Nagar Haveli
Total
State


Count of Three Wheelers (Category L5 slow speed as per CMVR) in Each State
Count of Three Wheelers (Category L5 slow speed as per CMVR)
400
350
300
250
200
150
100
50
0
Andhra Pradesh
Assam
Bihar
Chhattisgarh
Delhi
Goa
Gujarat
Haryana
Himachal Pradesh
Jammu & Kashmir
Jharkhand
Karnataka
Kerala
Madhya Pradesh
Maharashtra
Manipur
Meghalaya
Nagaland
Odisha
Punjab
Rajasthan
Tamil Nadu
Telangana
Tripura
Uttar Pradesh
Uttarkhand
West Bengal
Andaman & Nicobar islands
Chandigarh
Dadra and Nagar Haveli
Total
State


Count of Passenger Cars (Category M1 as per CMVR) in Each State
Count of Passenger Cars (Category M1 as per CMVR)
0
20000
40000
60000
80000
100000
Andhra Pradesh
Assam
Bihar
Chhattisgarh
Delhi
Goa
Gujarat
Haryana
Himachal Pradesh
Jammu & Kashmir
Jharkhand
Karnataka
Kerala
Madhya Pradesh
Maharashtra
Manipur
Meghalaya
Nagaland
Odisha
Punjab
Rajasthan
Tamil Nadu
Telangana
Tripura
Uttar Pradesh
Uttarkhand
West Bengal
Andaman & Nicobar islands
Chandigarh
Dadra and Nagar Haveli
Total
State


Count of Buses in Each State
Count of Buses
0
5
10
15
20
25
Andhra Pradesh
Assam
Bihar
Chhattisgarh
Delhi
Goa
Gujarat
Haryana
Himachal Pradesh
Jammu & Kashmir
Jharkhand
Karnataka
Kerala
Madhya Pradesh
Maharashtra
Manipur
Meghalaya
Nagaland
Odisha
Punjab
Rajasthan
Tamil Nadu
Telangana
Tripura
Uttar Pradesh
Uttarkhand
West Bengal
Andaman & Nicobar islands
Chandigarh
Dadra and Nagar Haveli
Total
State

- 1.Gujrat the highest no. of Two Wheelers (Category L1 & L2 ) of more than 5000.
- 2.There are 10 states and union territories combined that does not have Two Wheelers (Category L1 & L2 ).
- 3.Uttar Pradesh has the highest no. of Two Wheelers of (Category L2 (CMVR).
- 4.Uttar Pradesh has the highest no. of Two Wheelers (Max power not exceeding 250 Watts).
- 5.Uttar Pradesh has the highest no. of Three Wheelers.
- 6.Maharastra has highest count of Passenger cars (Category M1 as per CMVR).
- 7.Delhi has the highest no. of buses of EV type numbered above 20.
- 8.There is total of little over 25 busus of EV type in whole country.




- The no. of buses are much less as compared to no. of EV cars in all states.