|  |  |
| --- | --- |
| Project Title | **Electric Vehicle Sales by State in India** |
| Tools | Python, ML, SQL, Excel |
| Domain | Data Analyst |
| Project Difficulties level | intermediate |

Dataset : Dataset is available in the given link. You can download it at your convenience.

[Click](https://drive.google.com/file/d/14_cAtKqyS_nlzAhdhftC7NDlnBx_4YBS/view?usp=sharing) [here](https://drive.google.com/file/d/14_cAtKqyS_nlzAhdhftC7NDlnBx_4YBS/view?usp=sharing) [to](https://drive.google.com/file/d/14_cAtKqyS_nlzAhdhftC7NDlnBx_4YBS/view?usp=sharing) [download](https://drive.google.com/file/d/14_cAtKqyS_nlzAhdhftC7NDlnBx_4YBS/view?usp=sharing) [data](https://drive.google.com/file/d/14_cAtKqyS_nlzAhdhftC7NDlnBx_4YBS/view?usp=sharing) [set](https://drive.google.com/file/d/14_cAtKqyS_nlzAhdhftC7NDlnBx_4YBS/view?usp=sharing)

# About Dataset

This dataset is valuable for analysts, data scientists, and researchers aiming to understand electric vehicle (EV) adoption trends across India. It is versatile and ideal for geographic market segmentation, trend analysis, and predictive modeling. By offering insights into regional EV sales patterns, the dataset supports strategic decision-making in market planning and infrastructure investment.

The data was meticulously scraped from the Clean Mobility Shift website, and then thoroughly preprocessed to ensure accuracy and relevance. All null values have been removed, and the dataset has been cleaned to prepare it for immediate use in exploration, visualization, and analytical projects. It is particularly valuable for market trend analysis, infrastructure planning, and policy development within the EV sector.

The dataset is provided in CSV format and is ready for analysis.

Included Files:

EV\_Dataset.csv: Contains state-level data on EV sales, including vehicle types and categories, offering a comprehensive view of EV distribution across Indian states.

Key Features:

State: Names of Indian states with recorded EV sales data.

Vehicle Type: Classifications of vehicles, such as two-wheelers and four-wheelers. Vehicle Category: Further classification into segments like commercial and passenger vehicles.

Electric\_Vehicle\_Sales\_Quantity: The number of EVs sold per state, essential for analyzing adoption trends.

**Example: You can get the basic idea how you can create a project from here**

|  |
| --- |
| **Electric Vehicle Sales by State in India: Machine Learning Project (3-Year**  **Experience Level)**  This project aims to analyze and predict the sales of Electric Vehicles (EV) by state in India using machine learning. The dataset contains the following columns:   * **Year**: The year of the sales. * **Month\_Name**: The month in which sales occurred. * **Date**: The specific date of the sales. * **State**: The state in India where the sales occurred. * **Vehicle\_Class**: The class of the vehicle (e.g., sedan, SUV, etc.). * **Vehicle\_Category**: The category of the vehicle (e.g., commercial, passenger). * **Vehicle\_Type**: The type of the vehicle (e.g., 2-wheeler, 4-wheeler). * **EV\_Sales\_Quantity**: The quantity of EV sales.   **Steps Involved:**  1. **Data Collection**: Load and inspect the dataset. |

|  |
| --- |
| 1. **Data Preprocessing**: Handle missing values, convert date formats, and perform feature engineering. 2. **Exploratory Data Analysis (EDA)**: Visualize trends and relationships between variables. 3. **Feature Engineering**: Create new features from the date column and encode categorical variables. 4. **Modeling**: Build a regression model to predict EV sales. 5. **Evaluation**: Evaluate the model performance and interpret the results. 6. **Visualization**: Visualize the results and trends using graphs and charts.   **Python Code: Step-by-Step**  **Step 1: Data Collection**  Start by loading the dataset. For this example, let's assume the dataset is in CSV format.  # Import necessary libraries import pandas as pd import numpy as np  # Load the dataset df = pd.read\_csv('ev\_sales\_india.csv')  # Display the first few rows of the dataset print(df.head())  **Step 2: Data Preprocessing**  Handle missing values and convert the date column to a proper datetime format. |

|  |
| --- |
| # Convert 'Date' column to datetime format df['Date'] = pd.to\_datetime(df['Date'])  # Check for missing values print(df.isnull().sum())  # Fill missing values (if any) using median for numerical columns or mode for categorical columns df['EV\_Sales\_Quantity'].fillna(df['EV\_Sales\_Quantity'].median()  , inplace=True)  df.fillna(df.mode().iloc[0], inplace=True)  **Step 3: Exploratory Data Analysis (EDA)**  Visualize trends in EV sales over time, across states, and vehicle categories.  import matplotlib.pyplot as plt import seaborn as sns  # Plot EV sales over the years plt.figure(figsize=(10, 6)) sns.lineplot(data=df, x='Year', y='EV\_Sales\_Quantity', hue='State') plt.title('EV Sales by State over the Years') plt.show()  # Plot sales by vehicle category |

|  |
| --- |
| plt.figure(figsize=(10, 6)) sns.barplot(x='Vehicle\_Category', y='EV\_Sales\_Quantity', data=df, ci=None) plt.title('EV Sales by Vehicle Category') plt.show()  **Step 4: Feature Engineering**  Create new features such as month and day from the Date column and encode  categorical variables.  # Extract Month and Day from the Date column df['Month'] = df['Date'].dt.month df['Day'] = df['Date'].dt.day  # Encode categorical variables using one-hot encoding df\_encoded = pd.get\_dummies(df, columns=['State', 'Vehicle\_Class', 'Vehicle\_Category', 'Vehicle\_Type'], drop\_first=True)  # Drop unnecessary columns like Date, Month\_Name (if already extracted into numerical values)  df\_encoded.drop(['Date', 'Month\_Name'], axis=1, inplace=True)  **Step 5: Modeling**  Use a regression model (e.g., Random Forest Regressor) to predict EV sales. |

|  |
| --- |
| from sklearn.model\_selection import train\_test\_split from sklearn.ensemble import RandomForestRegressor from sklearn.metrics import mean\_squared\_error  # Split the data into features and target variable X = df\_encoded.drop('EV\_Sales\_Quantity', axis=1) y = df\_encoded['EV\_Sales\_Quantity']  # Split the dataset into training and testing sets X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)  # Instantiate the model model = RandomForestRegressor(n\_estimators=100, random\_state=42)  # Train the model model.fit(X\_train, y\_train)  # Make predictions y\_pred = model.predict(X\_test)  # Evaluate the model mse = mean\_squared\_error(y\_test, y\_pred) rmse = np.sqrt(mse) |

|  |
| --- |
| print(f'Root Mean Squared Error: {rmse}')  **Step 6: Model Evaluation**  Check how well the model performs on the test set.  # Plot actual vs predicted sales plt.figure(figsize=(10, 6)) plt.scatter(y\_test, y\_pred) plt.title('Actual vs Predicted EV Sales') plt.xlabel('Actual EV Sales') plt.ylabel('Predicted EV Sales') plt.show()  # Check feature importance importance = model.feature\_importances\_ feature\_importance = pd.Series(importance, index=X\_train.columns).sort\_values(ascending=False)  # Plot the most important features plt.figure(figsize=(10, 6)) feature\_importance.plot(kind='bar') plt.title('Feature Importance') plt.show()  **Step 7: Conclusion**  The machine learning model helps in understanding the factors affecting Electric |
| Vehicle sales across different states and predicting future sales based on historical data. Feature importance gives insight into which factors (e.g., State, Vehicle Category) have the highest impact on sales.  **Explanation:**   * **Data Preprocessing**: Cleaned the dataset and handled missing values. * **Feature Engineering**: Created new columns from the Date column and   encoded categorical variables.   * **Modeling**: Built a Random Forest Regressor model to predict EV sales and evaluated its performance using RMSE (Root Mean Squared Error). * **Visualization**: Visualized sales trends and feature importance using bar plots and scatter plots. |

**NOTE :**

1. **this project is only for your guidance, not exactly the same you have to create. Here I am trying to show the way or idea of what steps you can follow and how your projects look. Some projects are very advanced (because it will be made with the help of flask, nlp, advance ai, advance DL and some advanced things ) which you can not understand .**
2. **You can make or analyze your project with yourself, with your idea, make it more creative from where we can get some information and understand about our business. make sure what overall things you have created all things you understand very well.**

**Example: You can get the basic idea how you can create a project from here**

**Sample code with output**

Importing

all

the

Required

Libraries.

[¶](https://www.kaggle.com/code/deepeshkansotia/ev-sales-in-india-eda#Importing-all-the-Required-Libraries.)

In

[1]:

import

pandas

as

pd

import

numpy

as

np

import

matplotlib.pyplot

as

plt

import

seaborn

as

sns

Reading

the

Given

Dataset

as

a

Pandas

Dataframe.

In

[2]:

df

=

pd

.

read\_csv(

'EV\_Dataset.csv'

)

Basic

Dataset

Overview

In

[3]:

df

.

shape

*#*

*so*

*the*

*data*

*contains*

*96845*

*rows*

*and*

*8*

*columns.*

Out[3]:

(96845

,

8)

In

[4]:

df

.

columns

.

nunique()

*#*

*8*

*unique*

*columns*

*are*

*present*

*in*

*total.*

Out[4]:

8

In

[5]:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| df.columns *# column names are as below:*  Out[5]:  Index(['Year', 'Month\_Name', 'Date', 'State', 'Vehicle\_Class',  'Vehicle\_Category', 'Vehicle\_Type',  'EV\_Sales\_Quantity'], dtype='object') In [6]:  df.head() *# first 5 rows of the dataset.*  Out[6]:   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | Yea  r | Month\_  Name | Date | State | Vehicle\_Class | Vehicle\_  Category | Vehicle  \_Type | EV\_Sales  \_Quantity | | 0 | 201  4.0 | jan | 1/1/  201  4 | Andhra Prades h | ADAPTED  VEHICLE | Others | Others | 0.0 | | 1 | 201  4.0 | jan | 1/1/  201  4 | Andhra Prades h | AGRICULTUR  AL TRACTOR | Others | Others | 0.0 | | 2 | 201  4.0 | jan | 1/1/  201  4 | Andhra Prades h | AMBULANCE | Others | Others | 0.0 | | 3 | 201  4.0 | jan | 1/1/  201  4 | Andhra Prades h | ARTICULATE  D VEHICLE | Others | Others | 0.0 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 4 | 201  4.0 | jan | 1/1/  201  4 | Andhra Prades h | BUS | Bus | Bus | 0.0 |   In [7]:  df.tail() *# last 5 rows of the dataset.*  Out[7]:   |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | |  | Yea  r | Month\_  Name | Date | State | Vehicle\_Cla ss | Vehicle\_  Category | Vehicle  \_Type | EV\_Sales  \_Quantity | | 96  84  0 | 202  3.0 | dec | 12/1/  2023 | Anda man &  Nicob ar  Island | MOTOR  CAR | 4-Wheele  rs | 4W\_Pe  rsonal | 1.0 | | 96  84  1 | 202  3.0 | dec | 12/1/  2023 | Anda man &  Nicob ar  Island | MOTOR  CYCLE/SC  OOTER-US  ED FOR  HIRE | 2-Wheele  rs | 2W\_Sh  ared | 5.0 | | 96  84  2 | 202  3.0 | dec | 12/1/  2023 | Anda man &  Nicob ar  Island | OMNI BUS | Bus | Bus | 0.0 | |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 96  84  3 | 202  3.0 | dec | 12/1/  2023 | Anda man &  Nicob ar  Island | THREE  WHEELER  (GOODS) | 3-Wheele  rs | 3W\_G  oods | 0.0 |
| 96  84  4 | 202  3.0 | dec | 12/1/  2023 | Anda man &  Nicob ar  Island | THREE  WHEELER  (PASSENG  ER) | 3-Wheele  rs | 3W\_Sh  ared | 0.0 |

In [8]:

df.Year.value\_counts() *# so the data contains information 2014 to present.*

Out[8]:

Year

|  |  |
| --- | --- |
| 2019.0 | 10315 |
| 2023.0 | 10279 |
| 2018.0 | 10225 |
| 2022.0 | 10021 |
| 2017.0 | 9799 |
| 2016.0 | 9348 |
| 2021.0 | 9249 |
| 2015.0 | 9052 |

2014.0

9022

2020.0

8675

2024.0

860

Name:

count,

dtype:

int64

In

[9]:

df

.

State

.

value\_counts()

*#*

*almost*

*every*

*state*

*and*

*UT*

*are*

*present*

*in*

*the*

*data.*

Out[9]:

State

Maharashtra

4912

Karnataka

4830

Uttar

Pradesh

4557

Rajasthan

4552

Gujarat

4517

West

Bengal

4196

Tamil

Nadu

4063

Odisha

4027

Haryana

3842

Kerala

3666

Chhattisgarh

3590

Madhya

Pradesh

3587

Andhra

Pradesh

3457

Assam

3114

Uttarakhand

3045

|  |  |
| --- | --- |
| Himachal Pradesh | 2980 |
| Punjab | 2950 |
| Jharkhand | 2773 |
| Bihar | 2544 |
| Jammu and Kashmir | 2292 |
| Arunachal Pradesh | 2285 |
| Goa | 2139 |
| DNH and DD | 1927 |
| Delhi | 1871 |
| Meghalaya | 1867 |
| Puducherry | 1832 |
| Manipur | 1632 |
| Nagaland | 1588 |
| Tripura | 1564 |
| Mizoram | 1557 |
| Chandigarh | 1554 |
| Sikkim | 1246 |
| Andaman & Nicobar Island | 1226 |
| Ladakh | 1063 |

Name: count, dtype: int64 In [10]: df.Vehicle\_Class.value\_counts() *# below are the class of vehicles being sold in the Indian market.*

Out[10]:

Vehicle\_Class

MOTOR CAR 4111

M-CYCLE/SCOOTER 4101

GOODS CARRIER 4096

MOTOR CAB 3985

BUS 3813

...

SEMI-TRAILER (COMMERCIAL) 18

X-RAY VAN 12

MOTOR CYCLE/SCOOTER-WITH TRAILER 9

MODULAR HYDRAULIC TRAILER 3

MOTOR CARAVAN 3

Name: count, Length: 73, dtype: int64 In [11]: df.Vehicle\_Category.value\_counts() *# Below are the category of vehicles being sold in Indian markets.*

Out[11]:

Vehicle\_Category

Others 54423

2-Wheelers 13121

3-Wheelers 11491

Bus 9119

4-Wheelers 8691

Name: count, dtype: int64

|  |  |  |
| --- | --- | --- |
| In [12]: df.Vehicle\_Type.value\_counts() *# Below are the types of vehicles being sold in Indian markets.*  Out[12]:  Vehicle\_Type  Others 54423  2W\_Personal 11700  Bus 7026  4W\_Shared 4580  4W\_Personal 4111  3W\_Shared 3786  3W\_Goods 3208  Institution Bus 2093  3W\_Shared\_LowSpeed 1951  3W\_Goods\_LowSpeed 1517  2W\_Shared 1421  3W\_Personal 1029  Name: count, dtype: int64 In [13]: df.drop(columns=['Year']).describe() *# Basic statistics related to the EV sales in India Quantity wise.*  Out[13]:   |  |  | | --- | --- | |  | EV\_Sales\_ | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | |  | Quantity | | cou  nt | 96845.0000  00 | | me an | 37.108896 | | std | 431.566675 | | min | 0.000000 | | 25  % | 0.000000 | | 50  % | 0.000000 | | 75  % | 0.000000 | | ma  x | 20584.0000  00 |   Checking for Duplicates and Missing Values.  In [14]:  check\_duplicates=df.duplicated().sum() print(check\_duplicates)  0  *No Duplicates present.*  In [15]:  check\_missing\_values=df.isnull().sum() |

print

(

check\_missing\_values

)

Year

0

Month\_Name

0

Date

0

State

0

Vehicle\_Class

0

Vehicle\_Category

0

Vehicle\_Type

0

EV\_Sales\_Quantity

0

dtype:

int64

*No*

*missing*

*values*

*present.*

Checking

if

the

Datatypes

are

correct

or

not

In

[16]:

df

.

info()

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| <class 'pandas.core.frame.DataFrame'> RangeIndex: 96845 entries, 0 to 96844  Data columns (total 8 columns): | | | |  |
| # | Column | Non-Null Count | | Dtype |
| --- | ------ | -------------- | | ----- |
| 0 | Year | 96845 | non-null | float64 |
| 1 | Month\_Name | 96845 | non-null | object |
| 2 | Date | 96845 | non-null | object |
| 3 | State | 96845 | non-null | object |
| 4 | Vehicle\_Class | 96845 | non-null | object |
| 5 | Vehicle\_Category | 96845 | non-null | object |
| 6 | Vehicle\_Type | 96845 | non-null | object |
| 7 | EV\_Sales\_Quantity | 96845 | non-null | float64 |

dtypes:

float64(2),

object(6)

memory

usage:

5.9+

MB

*The*

*column*

*Year*

*is*

*given*

*as*

*Float,*

*but*

*we*

*should*

*convert*

*it*

*to*

*int.*

In

[17]:

df[

'Year'

]

=

df[

'Year'

]

.

astype(

int

)

*The*

*column*

*Date*

*is*

*given*

*as*

*Object,*

*but*

*it*

*should*

*be*

*in*

*Datetime*

*format*

In

[18]:

df[

'Date'

]

=

pd

.

to\_datetime(df[

'Date'

]

,

errors

=

'coerce'

)

*Similarly*

*converting*

*other*

*columns*

*to*

*their*

*relevant*

*datatypes.*

In

[19]:

categorical\_columns

=

[

'Month\_Name'

,

'State'

,

'Vehicle\_Class'

,

'Vehicle\_Category'

,

'Vehicle\_Type'

]

df[categorical\_columns]

=

df[categorical\_columns]

.

astype(

'category'

)

Checking

once

again

In

[20]:

df

.

info()

*#*

*now*

*everything*

*is*

*well*

*organised.*

<

class

'pandas.core.frame.DataFrame'>

|  |
| --- |
| RangeIndex: 96845 entries, 0 to 96844 Data columns (total 8 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----   1. Year 96845 non-null int64 2. Month\_Name 96845 non-null category 3. Date 96845 non-null datetime64[ns] 4. State 96845 non-null category 5. Vehicle\_Class 96845 non-null category 6. Vehicle\_Category 96845 non-null category 7. Vehicle\_Type 96845 non-null category 8. EV\_Sales\_Quantity 96845 non-null float64 dtypes: category(5), datetime64[ns](1), float64(1), int64(1) |

memory

usage:

2.7

MB

Data

Visualisation

In

[21]:

plt

.

figure(figsize

=

(

6

,

4

))

plt

.

title(

'Yearly

Analysis

of

EV

Sales

in

India'

)

sns

.

lineplot(x

=

'Year'

,

y

=

'EV\_Sales\_Quantity'

,

data

=

df,

marker

=

'o'

,

color

=

'b'

)

plt

.

xlabel(

'Year'

)

plt

.

ylabel(

'EV

Sales'

)

;

In

[22]:

plt

.

figure(figsize

=

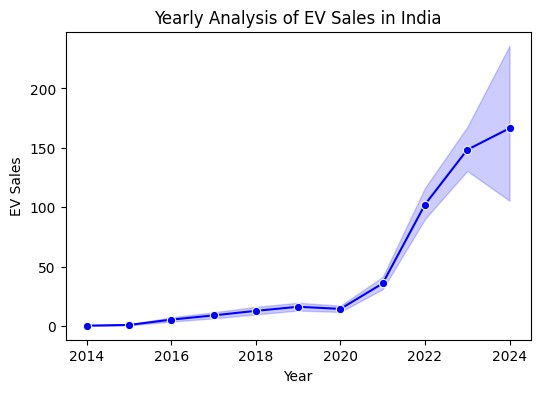
(

6

,

4

))



plt

.

title(

'Monthly

Analysis

of

EV

Sales

in

India'

)

sns

.

lineplot(x

=

'Month\_Name'

,

y

=

'EV\_Sales\_Quantity'

,

data

=

df,

marker

=

'o'

,

color

=

'r'

)

plt

.

xlabel(

'Month'

)

plt

.

ylabel(

'EV

Sales'

)

;

In

[61]:

plt

.

figure(figsize

=

(

6

,

7

))

plt

.

title(

'State-Wise

Analysis

of

EV

Sales'

)

sns

.

barplot(y

=

'State'

,

x

=

'EV\_Sales\_Quantity'

,

data

=

df,

hue

=

'State'

,

palette

=

'bright'

)

plt

.

xlabel(

'States'

)

plt

.

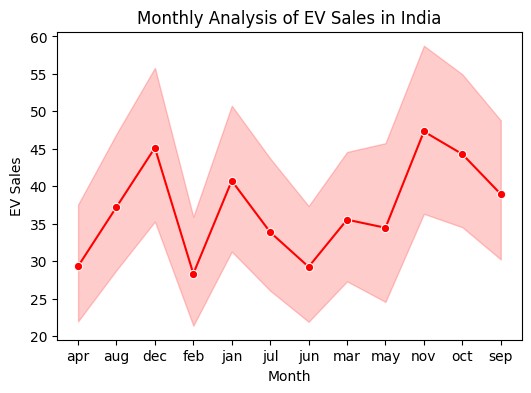
ylabel(

'EV

Sales'

)

;



In

[59]:

plt

.

figure(figsize

=

(

15

,

4

))

sns

.

barplot(x

=

'Vehicle\_Class'

,

y

=

'EV\_Sales\_Quantity'

,data

=

df,

hue

=

'Vehicle\_Class'

,

palette

=

'bright'

)

plt

.

title(

'Analysis

by

Vehicle

Class'

)

plt

.

xlabel(

'Vehicle

Class'

)

plt

.

ylabel(

'EV

Sales'

)

plt

.

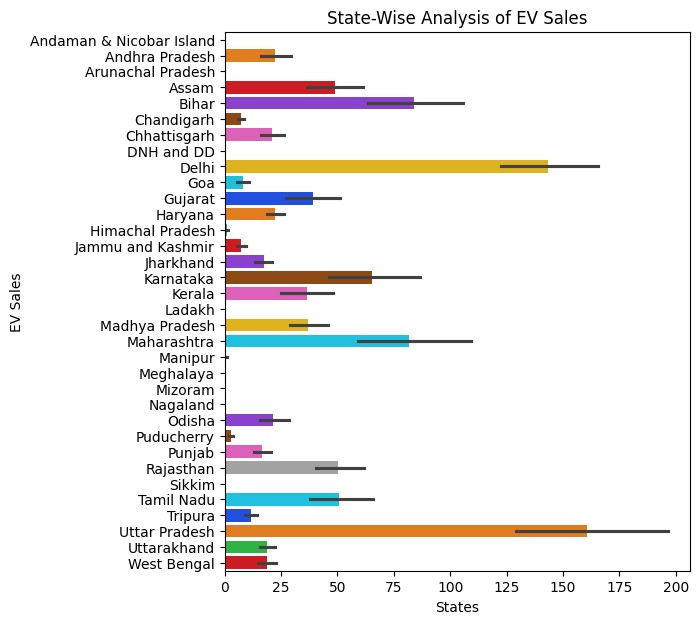
xticks(rotation

=

90

;

)



In

[66]:

plt

.

figure(figsize

=

(

6

,

4

))

sns

.

barplot(x

=

'Vehicle\_Category'

,

y

=

'EV\_Sales\_Quantity'

,data

=

df,

hue

=

'Vehicle\_Category'

,

palette

=

'bright'

)

plt

.

title(

'Analysis

by

Vehicle

Category'

)

plt

.

xlabel(

'Vehicle

Category'

)

plt

.

ylabel(

'EV

Sales'

)

plt

.

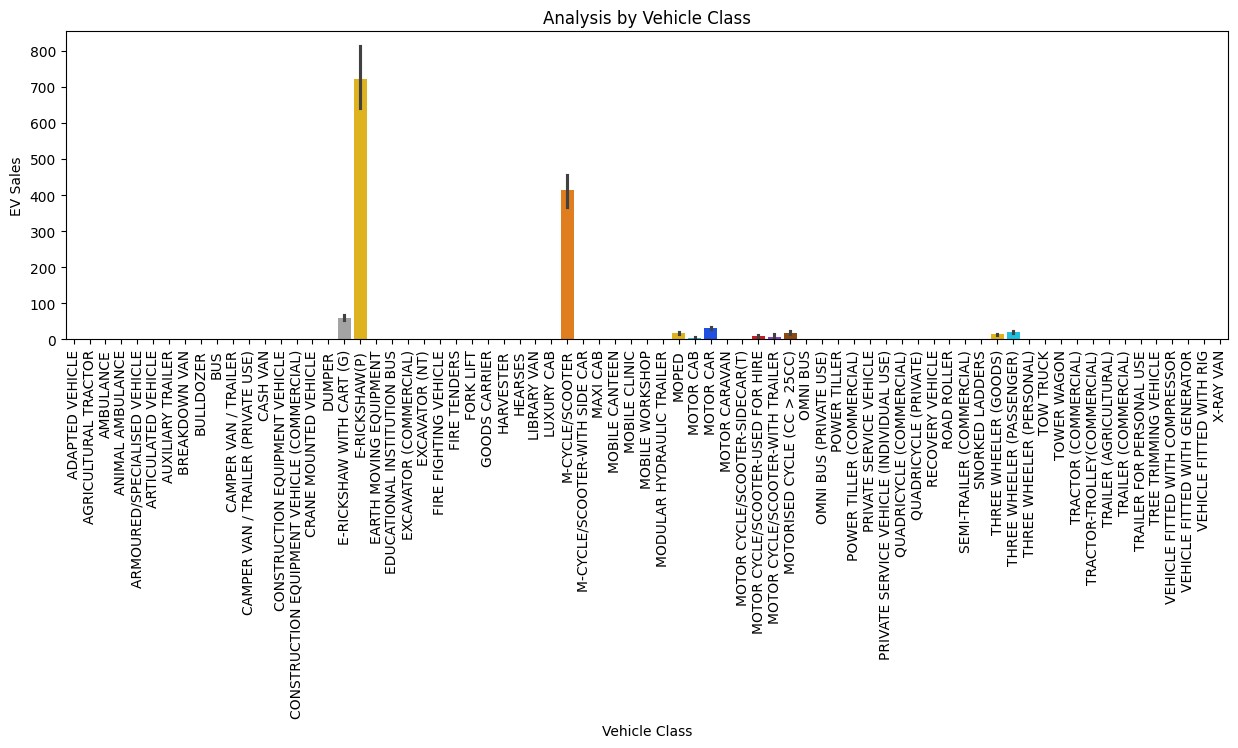
xticks(rotation

=

0

;

)



In

[68]:

plt

.

figure(figsize

=

(

6

,

4

))

sns

.

barplot(x

=

'Vehicle\_Type'

,

y

=

'EV\_Sales\_Quantity'

,data

=

df,

hue

=

'Vehicle\_Type'

,

palette

=

'bright'

)

plt

.

title(

'Analysis

by

Vehicle

Type'

)

plt

.

xlabel(

'Vehicle

Type'

)

plt

.

ylabel(

'EV

Sales'

)

plt

.

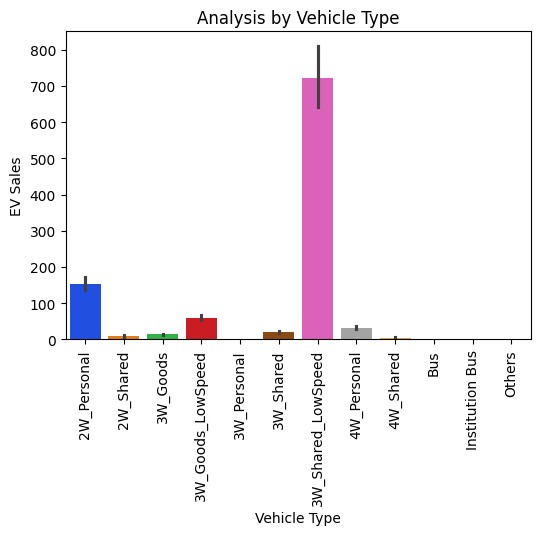
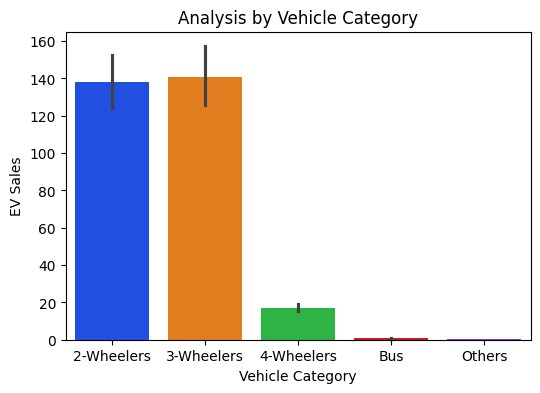
xticks(rotation

=

90

;

)



[Reference](https://github.com/Chaganti-Reddy/EVMarket-India) [link](https://github.com/Chaganti-Reddy/EVMarket-India)