**Project 2: Electric Vehicle Sales Prediction Project (India)**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestRegressor

from sklearn.metrics import mean\_squared\_error

Step 1: Data Collection

Load the dataset

Please ensure 'EV\_Dataset.csv' is in the same directory as this script

df = pd.read\_csv('EV\_Dataset.csv')

print("First Five Rows:

", df.head())

Step 2: Data Preprocessing

Convert 'Date' column to datetime

if 'Date' in df.columns:

df['Date'] = pd.to\_datetime(df['Date'])

Fill missing values for numeric and categorical separately

df['EV\_Sales\_Quantity'] = df['EV\_Sales\_Quantity'].fillna(df['EV\_Sales\_Quantity'].median())

df.fillna(df.mode().iloc[0], inplace=True)

Step 3: Exploratory Data Analysis (EDA)

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()

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

Extract month and day if 'Date' exists

if 'Date' in df.columns:

df['Month'] = df['Date'].dt.month

df['Day'] = df['Date'].dt.day

Encode categorical variables

categorical\_cols = ['State', 'Vehicle\_Class', 'Vehicle\_Category', 'Vehicle\_Type']

for col in categorical\_cols:

if col in df.columns:

df = pd.get\_dummies(df, columns=[col], drop\_first=True)

Drop unneeded columns

for col in ['Date', 'Month\_Name']:

if col in df.columns:

df = df.drop(col, axis=1)

Step 5: Modeling

X = df.drop('EV\_Sales\_Quantity', axis=1)

y = df['EV\_Sales\_Quantity']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

model = RandomForestRegressor(n\_estimators=100, random\_state=42)

model.fit(X\_train, y\_train)

y\_pred = model.predict(X\_test)

Step 6: Evaluation

mse = mean\_squared\_error(y\_test, y\_pred)

rmse = np.sqrt(mse)

print(f'Root Mean Squared Error: {rmse}')

# Actual vs Predicted

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()