⚡ Baseline & ML | Electric Vehicle Prediction 🚗 🔋

📥 Imports and Setup

In [3]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from scipy import stats
import folium
```

In [4]:

```
data= pd.read_csv(r"C:\Users\chitt\Downloads\Electric_Vehicle_Population_Data (1
data
```

Out[4]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Mod
0	2T3YL4DV0E	King	Bellevue	WA	98005.0	2014	TOYOTA	RAV
1	5YJ3E1EB6K	King	Bothell	WA	98011.0	2019	TESLA	MODEL
2	5UX43EU02S	Thurston	Olympia	WA	98502.0	2025	BMW	X
3	JTMAB3FV5R	Thurston	Olympia	WA	98513.0	2024	TOYOTA	RAV PRIM
4	5YJYGDEE8M	Yakima	Selah	WA	98942.0	2021	TESLA	MODEL
...	...	...	...	...	...	...	...	
232225	5YJ3E1EA3K	King	Renton	WA	98058.0	2019	TESLA	MODEL
232226	1GKB0RDC1R	Snohomish	Snohomish	WA	98290.0	2024	GMC	HUMME EV SU

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Mod
232227	7SAYGDED3R	King	Redmond	WA	98033.0	2024	TESLA	MODEL
232228	JTMEB3FV5P	Chelan	Leavenworth	WA	98826.0	2023	TOYOTA	RAV PRIM
232229	5YJYGDEE3M	Kitsap	Bremerton	WA	98312.0	2021	TESLA	MODEL

232230 rows × 17 columns

In [5]:
```
data.head(10)
```

Out[5]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type
0	2T3YL4DV0E	King	Bellevue	WA	98005.0	2014	TOYOTA	RAV4	Battery Electric Vehicle (BEV)
1	5YJ3E1EB6K	King	Bothell	WA	98011.0	2019	TESLA	MODEL 3	Battery Electric Vehicle (BEV)
2	5UX43EU02S	Thurston	Olympia	WA	98502.0	2025	BMW	X5	Plug-in Hybrid Electric Vehicle (PHEV)
3	JTMAB3FV5R	Thurston	Olympia	WA	98513.0	2024	TOYOTA	RAV4 PRIME	Plug-in Hybrid Electric Vehicle (PHEV)
4	5YJYGDEE8M	Yakima	Selah	WA	98942.0	2021	TESLA	MODEL Y	Battery Electric Vehicle (BEV)
5	3C3CFFGE1G	Thurston	Olympia	WA	98501.0	2016	FIAT	500	Battery Electric Vehicle (BEV)
6	5YJ3E1EA4J	Snohomish	Marysville	WA	98271.0	2018	TESLA	MODEL 3	Battery Electric Vehicle (BEV)
7	5YJ3E1EA3K	King	Seattle	WA	98102.0	2019	TESLA	MODEL 3	Battery Electric Vehicle (BEV)
8	1N4AZ0CP5E	Thurston	Yelm	WA	98597.0	2014	NISSAN	LEAF	Battery Electric

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Model	Electric Vehicle Type
									Vehicle (BEV)
9	5YJSA1S25F	Thurston	Yelm	WA	98597.0	2015	TESLA	MODEL S	Battery Electric Vehicle (BEV)

In [6]:
```
data.iloc[0]
```

Out[6]:
```
VIN (1-10)
2T3YL4DV0E
County
King
City
Bellevue
State
WA
Postal Code
98005.0
Model Year
2014
Make
TOYOTA
Model
RAV4
Electric Vehicle Type                                                   Battery Ele
ctric Vehicle (BEV)
Clean Alternative Fuel Vehicle (CAFV) Eligibility          Clean Alternative Fu
el Vehicle Eligible
Electric Range
103.0
Base MSRP
0.0
Legislative District
41.0
DOL Vehicle ID
186450183
Vehicle Location                                                          POINT
(-122.1621 47.64441)
Electric Utility                                       PUGET SOUND ENERGY INC||CI
TY OF TACOMA - (WA)
2020 Census Tract
53033023604.0
Name: 0, dtype: object
```

In [7]:
```
data.tail()
```

Out[7]:

	VIN (1-10)	County	City	State	Postal Code	Model Year	Make	Mod
232225	5YJ3E1EA3K	King	Renton	WA	98058.0	2019	TESLA	MODEL
232226	1GKB0RDC1R	Snohomish	Snohomish	WA	98290.0	2024	GMC	HUMME EV SU
232227	7SAYGDED3R	King	Redmond	WA	98033.0	2024	TESLA	MODEL
232228	JTMEB3FV5P	Chelan	Leavenworth	WA	98826.0	2023	TOYOTA	RAV PRIM
232229	5YJYGDEE3M	Kitsap	Bremerton	WA	98312.0	2021	TESLA	MODEL

In [8]:

```
data.shape
```

Out[8]:

```
(232230, 17)
```

In [9]:

```
data.columns
```

Out[9]:

```
Index(['VIN (1-10)', 'County', 'City', 'State', 'Postal 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', 'Electric Utility', '2020 Census Tract'],
      dtype='object')
```

In [10]:

```
data.dtypes
```

```
Out[10]:  VIN (1-10)                                             object
          County                                                 object
          City                                                   object
          State                                                  object
          Postal Code                                           float64
          Model Year                                              int64
          Make                                                   object
          Model                                                  object
          Electric Vehicle Type                                  object
          Clean Alternative Fuel Vehicle (CAFV) Eligibility      object
          Electric Range                                        float64
          Base MSRP                                             float64
          Legislative District                                  float64
          DOL Vehicle ID                                          int64
          Vehicle Location                                       object
          Electric Utility                                       object
          2020 Census Tract                                     float64
          dtype: object
```

```
In [11]:  data['Electric Utility'].mode()[0]
```

```
Out[11]:  'PUGET SOUND ENERGY INC||CITY OF TACOMA - (WA)'
```

```
In [12]:  data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 232230 entries, 0 to 232229
Data columns (total 17 columns):
 #   Column                                             Non-Null Count   Dtype  
---  ------                                             --------------   -----  
 0   VIN (1-10)                                         232230 non-null  object 
 1   County                                             232226 non-null  object 
 2   City                                               232226 non-null  object 
 3   State                                              232230 non-null  object 
 4   Postal Code                                        232226 non-null  float64
 5   Model Year                                         232230 non-null  int64  
 6   Make                                               232230 non-null  object 
 7   Model                                              232230 non-null  object 
 8   Electric Vehicle Type                              232230 non-null  object 
 9   Clean Alternative Fuel Vehicle (CAFV) Eligibility  232230 non-null  object 
 10  Electric Range                                     232203 non-null  float64
 11  Base MSRP                                          232203 non-null  float64
 12  Legislative District                               231749 non-null  float64
 13  DOL Vehicle ID                                     232230 non-null  int64  
 14  Vehicle Location                                   232219 non-null  object 
 15  Electric Utility                                   232226 non-null  object 
 16  2020 Census Tract                                  232226 non-null  float64
dtypes: float64(5), int64(2), object(10)
memory usage: 30.1+ MB
```

```
In [13]:  data.describe().T
```

Out[13]:

	count	mean	std	min	25%	
Postal Code	232226.0	9.818017e+04	2.489408e+03	1.731000e+03	9.805200e+04	9.81260
Model Year	232230.0	2.021354e+03	2.994884e+00	1.999000e+03	2.020000e+03	2.02300
Electric Range	232203.0	4.675600e+01	8.437360e+01	0.000000e+00	0.000000e+00	0.00000
Base MSRP	232203.0	8.038090e+02	7.246597e+03	0.000000e+00	0.000000e+00	0.00000
Legislative District	231749.0	2.888098e+01	1.490450e+01	1.000000e+00	1.700000e+01	3.20000
DOL Vehicle ID	232230.0	2.343671e+08	6.831418e+07	4.385000e+03	2.034737e+08	2.51271
2020 Census Tract	232226.0	5.298177e+10	1.507814e+09	1.001020e+09	5.303301e+10	5.30330

In [14]:
```
data.isnull().sum()
```

Out[14]:
```
VIN (1-10)                                             0
County                                                 4
City                                                   4
State                                                  0
Postal Code                                            4
Model Year                                             0
Make                                                   0
Model                                                  0
Electric Vehicle Type                                  0
Clean Alternative Fuel Vehicle (CAFV) Eligibility      0
Electric Range                                        27
Base MSRP                                             27
Legislative District                                 481
DOL Vehicle ID                                         0
Vehicle Location                                      11
Electric Utility                                       4
2020 Census Tract                                      4
dtype: int64
```

In [15]:
```
data.duplicated().sum()
```

Out[15]:
```
0
```

In [16]:
```
data.nunique()
```

Out[16]:
```
VIN (1-10)                                            13560
County                                                  209
City                                                    786
State                                                    49
Postal Code                                             950
Model Year                                               21
Make                                                     46
Model                                                   170
Electric Vehicle Type                                     2
Clean Alternative Fuel Vehicle (CAFV) Eligibility         3
Electric Range                                          109
Base MSRP                                                31
Legislative District                                     49
DOL Vehicle ID                                       232230
Vehicle Location                                        948
Electric Utility                                         76
2020 Census Tract                                      2191
dtype: int64
```

In [20]:
```
data.mode().iloc[0]
```

Out[20]:
```
VIN (1-10)
7SAYGDEE6P
County
King
City
Seattle
State
WA
Postal Code
98052.0
Model Year
2023.0
Make
TESLA
Model
MODEL Y
Electric Vehicle Type                                                   Battery
Electric Vehicle (BEV)
Clean Alternative Fuel Vehicle (CAFV) Eligibility    Eligibility unknown as bat
tery range has not b...
Electric Range
0.0
Base MSRP
0.0
Legislative District
41.0
DOL Vehicle ID
4385
Vehicle Location                                                           POIN
T (-122.13158 47.67858)
Electric Utility                                         PUGET SOUND ENERGY INC
||CITY OF TACOMA - (WA)
2020 Census Tract
53033028200.0
Name: 0, dtype: object
```

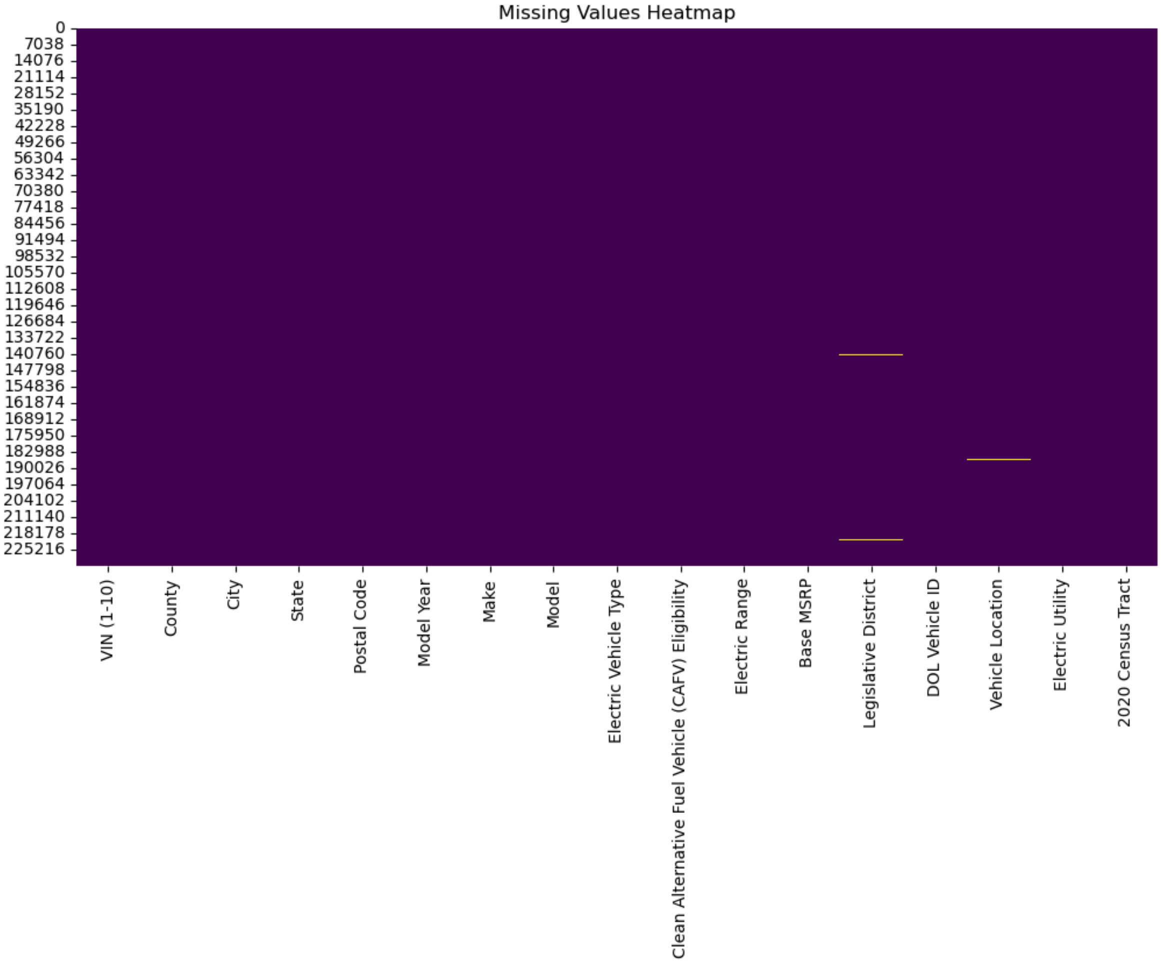
🧹 Data Cleaning

In [23]:
```
# Check for missing values
missing_values = data.isnull().sum()
missing_values = missing_values[missing_values>0].sort_values(ascending= False)
print("Missing values:")
print(missing_values)
```

```
Missing values:
Legislative District    481
Electric Range           27
Base MSRP                27
Vehicle Location         11
County                    4
City                      4
Postal Code               4
Electric Utility          4
2020 Census Tract         4
dtype: int64
```

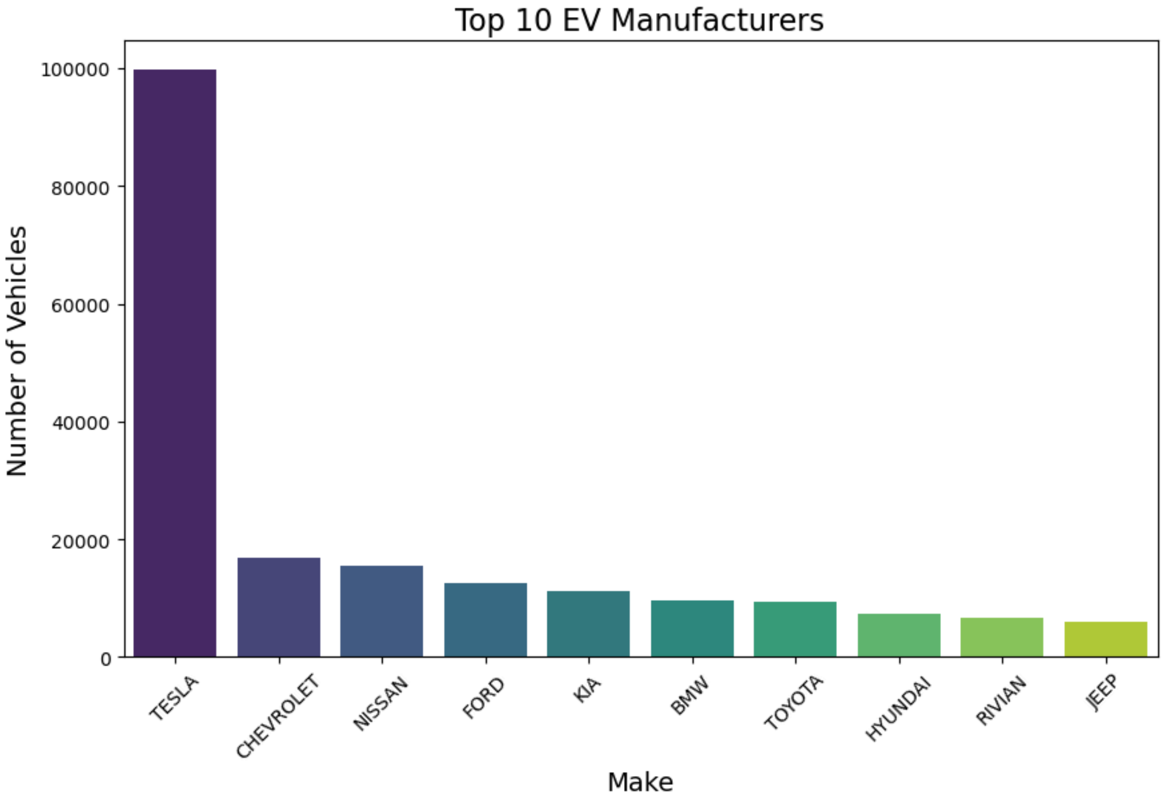
📊 Exploratory Data Analysis (EDA)

In [28]:
```
# Visualizing missing values
plt.figure(figsize = (12, 6))
sns.heatmap(data.isnull(), cbar=False, cmap="viridis")
plt.title("Missing Values Heatmap")
plt.show()
```



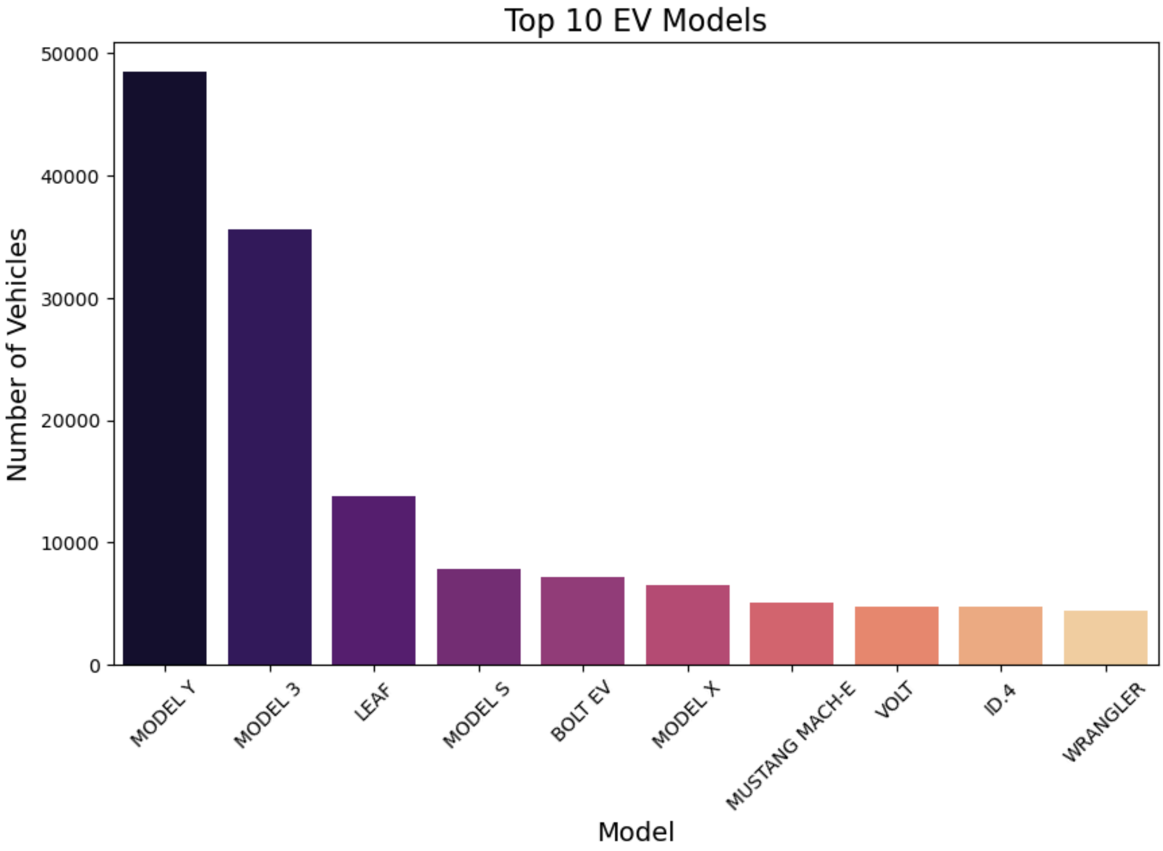
In [32]:

```python
# Top 10 Markes
top_makes = data['Make'].value_counts().nlargest(10)

import warnings
warnings.filterwarnings('ignore')

# Plot
plt.figure(figsize=(10, 6))
sns.barplot(x=top_makes.index, y=top_makes.values, palette="viridis")
plt.title('Top 10 EV Manufacturers', fontsize=16)
plt.xlabel('Make', fontsize=14)
plt.ylabel('Number of Vehicles', fontsize=14)
plt.xticks(rotation=45)
plt.show()
```

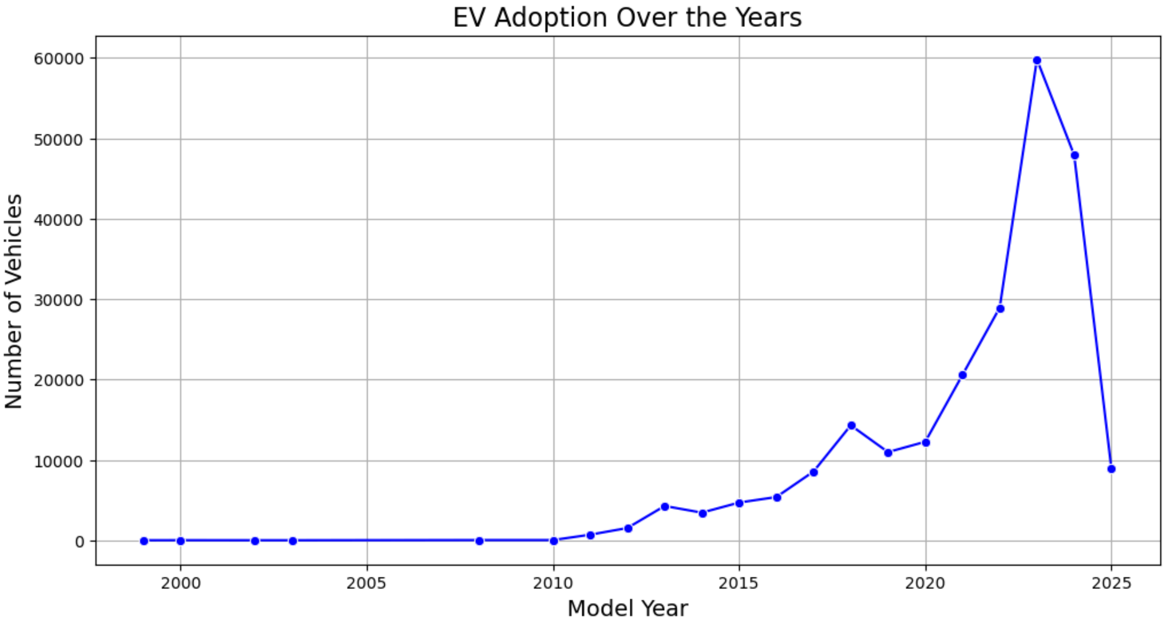


In [36]:

```
# Top  10 Models
top_models = data['Model'].value_counts().nlargest(10)

#Plot
plt.figure(figsize=(10,6))
sns.barplot(x=top_models.index, y=top_models.values, palette="magma")
plt.title('Top 10 EV Models', fontsize=16)
plt.xlabel('Model', fontsize=14)
plt.ylabel('Number of Vehicles', fontsize=14)
plt.xticks(rotation=45)
plt.show()
```

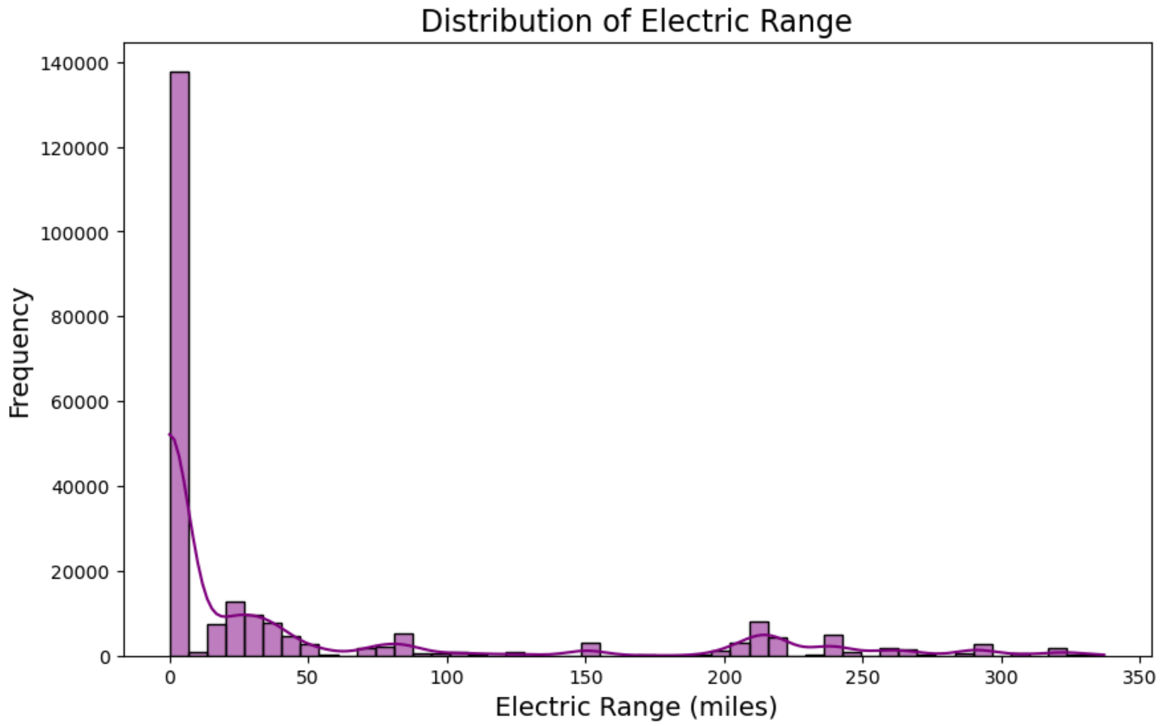


In [38]:
```python
# EVs by Year
evs_by_year = data['Model Year'].value_counts().sort_index()

# Plot
plt.figure(figsize=(12,6))
sns.lineplot(x=evs_by_year.index, y=evs_by_year.values, marker ='o', color='b')
plt.title('EV Adoption Over the Years', fontsize=16)
plt.xlabel('Model Year', fontsize=14)
plt.ylabel('Number of Vehicles', fontsize=14)
plt.grid(True)
plt.show()
```

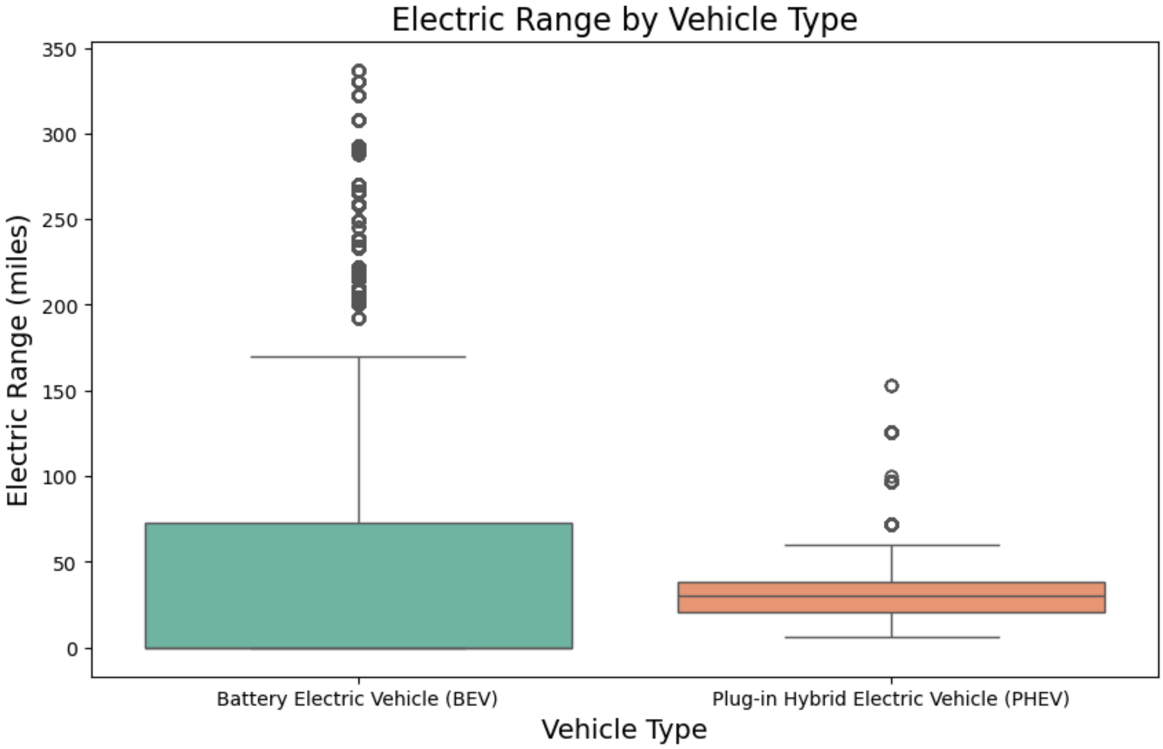


In [40]:
```python
# Plot Distribution of Electric Range
plt.figure(figsize=(10,6))
```

```
sns.histplot(data['Electric Range'], bins=50, kde=True, color='purple')
plt.title('Distribution of Electric Range', fontsize=16)
plt.xlabel('Electric Range (miles)', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.show()
```

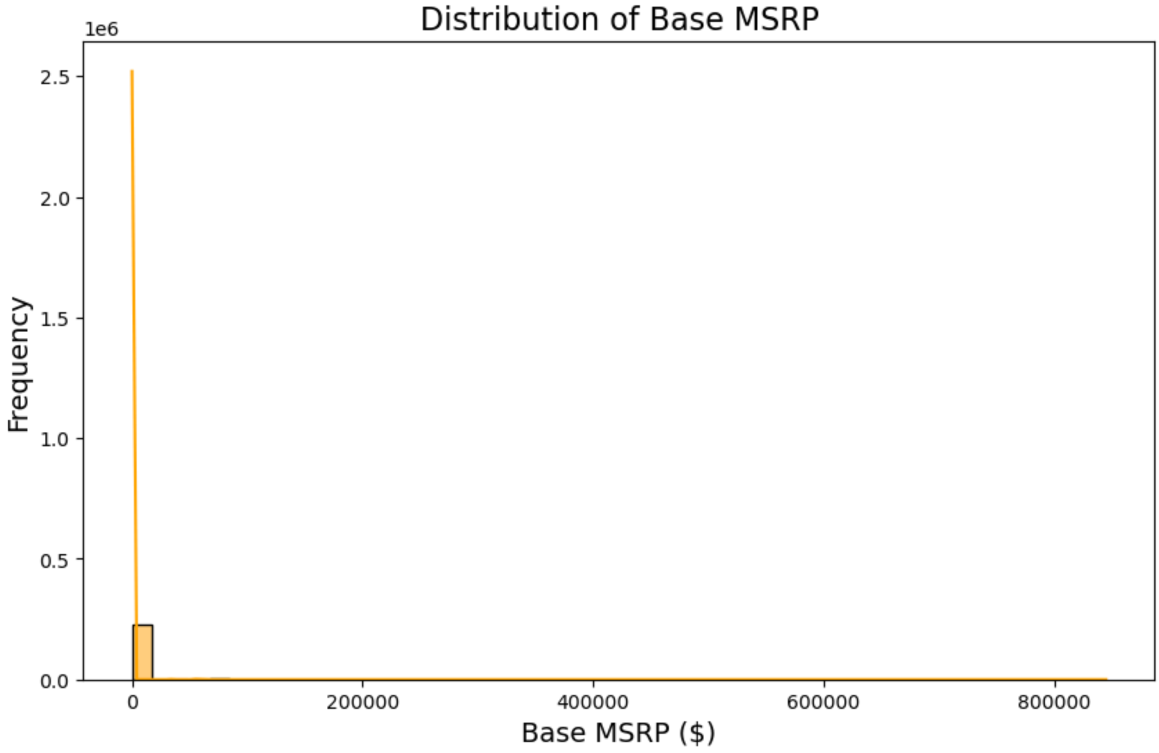


```
In [42]: # Boxplot of Electric Range by Vehicle Type
plt.figure(figsize=(10, 6))
sns.boxplot(x='Electric Vehicle Type', y='Electric Range', data=data, palette="S
plt.title('Electric Range by Vehicle Type', fontsize=16)
plt.xlabel('Vehicle Type', fontsize=14)
plt.ylabel('Electric Range (miles)', fontsize=14)
plt.show()
```



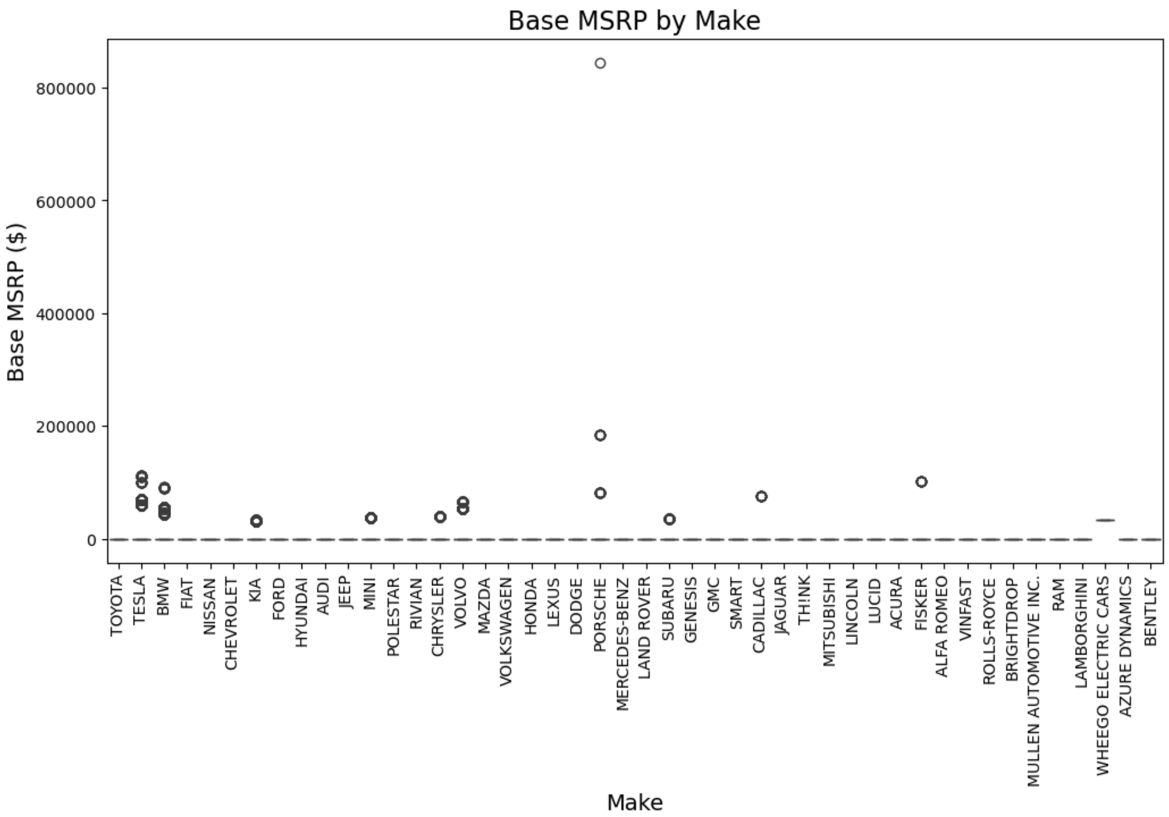
In [44]:

```
# Plot distribution of Base MSRP
plt.figure(figsize=(10, 6))
sns.histplot(data['Base MSRP'], bins=50, kde=True, color='orange')
plt.title('Distribution of Base MSRP', fontsize=16)
plt.xlabel('Base MSRP ($)', fontsize=14)
plt.ylabel('Frequency', fontsize=14)
plt.show()
```

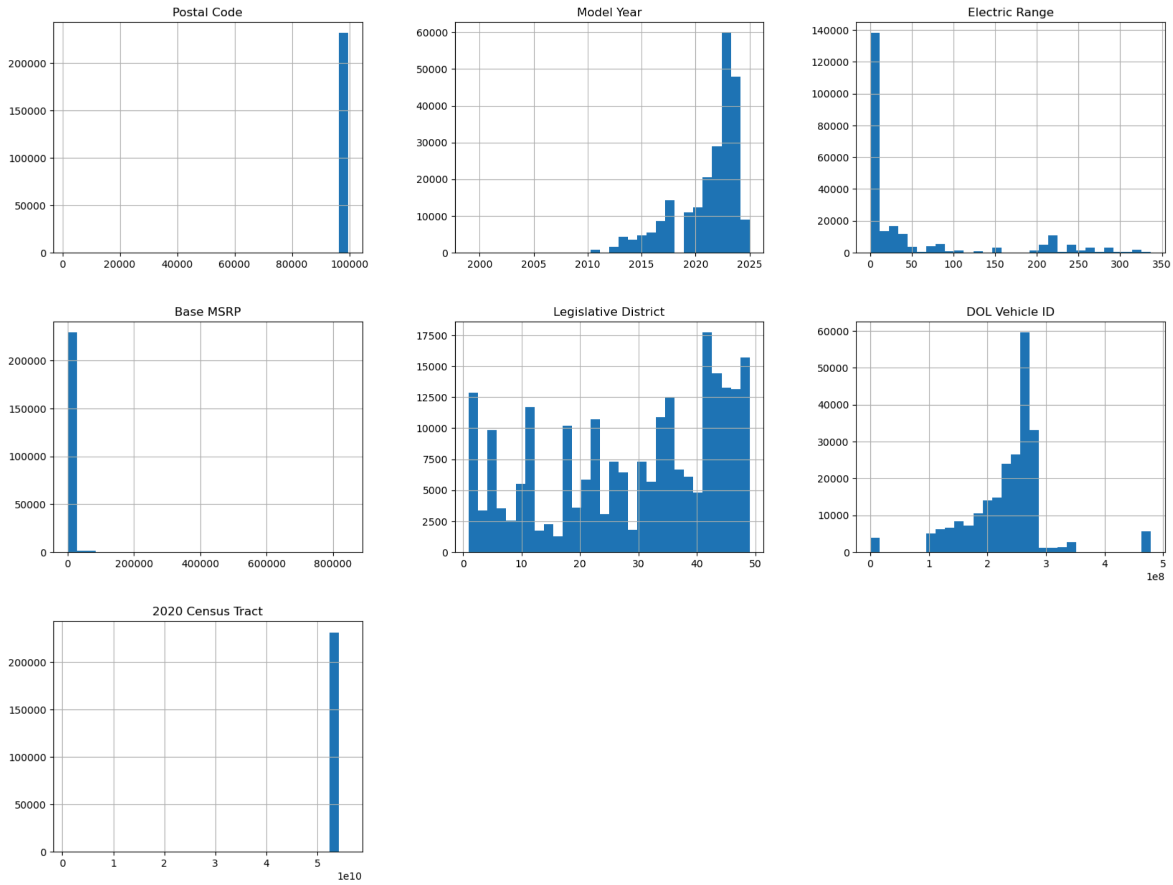


In [46]:

```
# Boxplot of Base MSRP by Make
plt.figure(figsize=(12, 6))
sns.boxplot(x='Make', y='Base MSRP', data=data, palette="coolwarm")
plt.title('Base MSRP by Make', fontsize=16)
plt.xlabel('Make', fontsize=14)
plt.ylabel('Base MSRP ($)', fontsize=14)
plt.xticks(rotation=90)
plt.show()
```



In [54]:
```
# Plot histograms for each feature
data.hist(bins=30, figsize=(20,15))
plt.show()
```



In [58]:
```
# Mapping Electric Vehicles by Location
map_center = [47.5, -122.2]  # Approximate center for Washington State
m = folium.Map(location=map_center, zoom_start=8)
```

```
for _, row in data.dropna(subset=["Vehicle Location"]).sample(500).iterrows():
    try:
        lat, lon = row["Vehicle Location"].replace("POINT (", "").replace(")", "
        folium.CircleMarker(
            location=[float(lon), float(lat)],
            radius=2,
            color='blue',
            fill=True,
            fill_color='blue'
        ).add_to(m)
    except:
        pass

m.save("electric_vehicles_map.html")  # Save the map as an HTML file
print("Map of vehicles saved as 'electric_vehicles_map.html'")
```

```
Map of vehicles saved as 'electric_vehicles_map.html'
```

In [67]:
```
from sklearn.model_selection import train_test_split, cross_val_score, GridSearc
from sklearn.preprocessing import OneHotEncoder, StandardScaler
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.impute import SimpleImputer
from sklearn.dummy import DummyRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

In [61]:
```
numeric_features = ["Model Year", "Electric Range", "Base MSRP"]
categorical_features = ["Make", "Model", "Electric Vehicle Type", "Clean Alterna
```

In [69]:
```
# Numeric Transformer

numeric_transformer = Pipeline(steps=[
    ("imputer", SimpleImputer(strategy="median")),
    ("scaler", StandardScaler())
])
```

In [71]:
```
# Categorical Transformer
categorical_transformer = Pipeline(steps=[
    ("imputer", SimpleImputer(strategy="most_frequent")),
    ("onehot", OneHotEncoder(handle_unknown="ignore"))
])
```

In [73]:
```
# Column Transformer
preprocessor = ColumnTransformer(
    transformers=[
        ("num", numeric_transformer, numeric_features),
        ("cat", categorical_transformer, categorical_features)
    ]
)
```

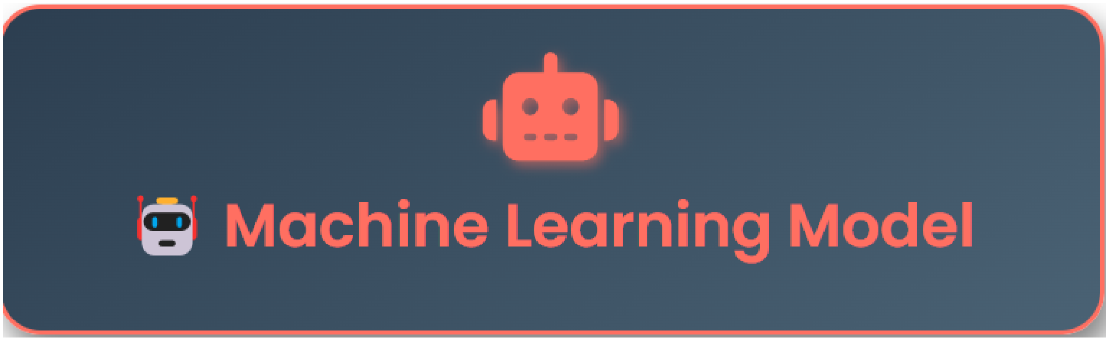
In [75]:
```
# Splitting the data
X = data[numeric_features + categorical_features]
y = data["Electric Range"].fillna(0)  # Target variable with missing values repl
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

📊 Baseline Model

In [78]:
```
# Baseline Model (Dummy Regressor)
dummy_regressor = DummyRegressor(strategy="mean")
dummy_regressor.fit(X_train, y_train)
y_pred_dummy = dummy_regressor.predict(X_test)
```

In [80]:
```
# Evaluate Baseline Model
dummy_mse = mean_squared_error(y_test, y_pred_dummy)
dummy_r2 = r2_score(y_test, y_pred_dummy)
print(f"Baseline Model - MSE: {dummy_mse:.2f}, R2 Score: {dummy_r2:.2f}")
```

```
Baseline Model - MSE: 7102.71, R2 Score: -0.00
```



🤖 Machine Learning Model

In [83]:
```
models = {
    "Linear Regression": LinearRegression(),
    "Random Forest": RandomForestRegressor(n_estimators=100, random_state=42)
}

for name, model in models.items():
    pipeline = Pipeline(steps=[
        ("preprocessor", preprocessor),
        ("model", model)
    ])
    scores = cross_val_score(pipeline, X_train, y_train, cv=5, scoring='r2')
    print(f"{name} - Cross-Validation R2 Score: {np.mean(scores):.2f}")
```

```
Linear Regression - Cross-Validation R2 Score: 1.00
Random Forest - Cross-Validation R2 Score: 1.00
```

🎛️ Hyperparameter Tuning

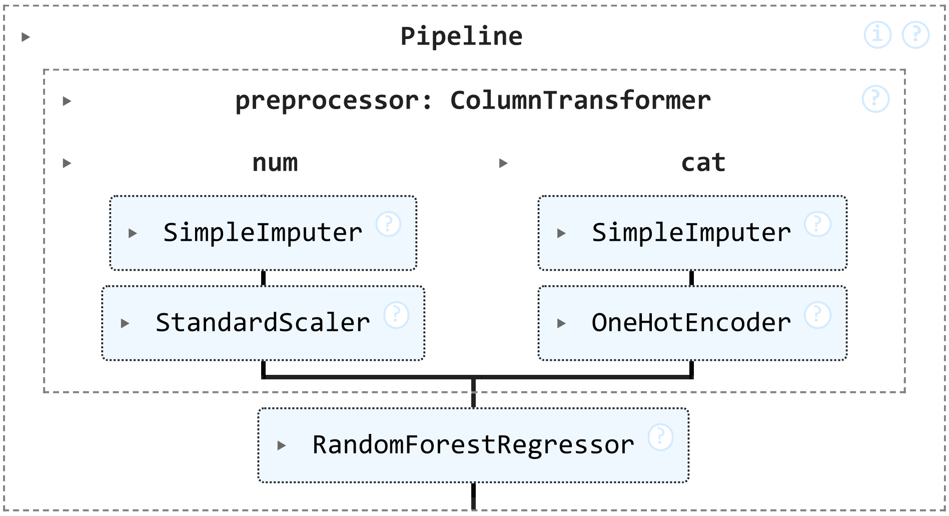
In [86]:
```
# Hyperparameter Tuning for Random Forest
param_grid = {
    "model__n_estimators": [50, 100, 200],
    "model__max_depth": [None, 10, 20]
}

grid_search = GridSearchCV(Pipeline(steps=[("preprocessor", preprocessor), ("mod
grid_search.fit(X_train, y_train)
print(f"Best Parameters: {grid_search.best_params_}")
```

```
Best Parameters: {'model__max_depth': None, 'model__n_estimators': 50}
```

🚂 Train the Best Model on the Full Training Set

In [91]:
```
# Train Best Model
best_model = grid_search.best_estimator_
best_model.fit(X_train, y_train)
```

Out[91]:



🔍 Make Predictions on the Validation Set

In [94]:
```
# Make Predictions
y_pred_best = best_model.predict(X_test)
```

🏁 Compare Baseline and Best Model

In [97]:
```
# Evaluate Best Model
best_mse = mean_squared_error(y_test, y_pred_best)
best_r2 = r2_score(y_test, y_pred_best)
print(f"Best Model - MSE: {best_mse:.2f}, R2 Score: {best_r2:.2f}")
```

```
Best Model - MSE: 0.00, R2 Score: 1.00
```

In [99]:
```
# Compare Baseline and Best Model
print(f"Improvement in R2 Score: {best_r2 - dummy_r2:.2f}")
```

```
Improvement in R2 Score: 1.00
```