```
import joblib
import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.preprocessing import LabelEncoder
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.model_selection import RandomizedSearchCV
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
```

```
df = pd.read_csv("Electric_Vehicle_Population_By_County.csv")
```

```
df.head()
```

	Date	County	State	Vehicle Primary Use	Battery Electric Vehicles (BEVs)	Plug-In Hybrid Electric Vehicles (PHEVs)	Electric Vehicle (EV) Total	Non-Electric Vehicle Total	T Vehi
0	September 30 2022	Riverside	CA	Passenger	7	0	7	460	
1	December 31 2022	Prince William	VA	Passenger	1	2	3	188	
2	January 31 2020	Dakota	MN	Passenger	0	1	1	32	
3	June 30 2022	Ferry	WA	Truck	0	0	0	3,575	
4	July 31 2021	Douglas	CO	Passenger	0	1	1	83	

```
df.shape
```

```
(20819, 10)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20819 entries, 0 to 20818
Data columns (total 10 columns):
 #   Column                                    Non-Null Count  Dtype
---  ------                                    --------------  -----
 0   Date                                      20819 non-null  object
 1   County                                    20733 non-null  object
 2   State                                     20733 non-null  object
 3   Vehicle Primary Use                       20819 non-null  object
 4   Battery Electric Vehicles (BEVs)          20819 non-null  object 
 5   Plug-In Hybrid Electric Vehicles (PHEVs)  20819 non-null  object 
 6   Electric Vehicle (EV) Total               20819 non-null  object 
 7   Non-Electric Vehicle Total                20819 non-null  object 
 8   Total Vehicles                            20819 non-null  object 
 9   Percent Electric Vehicles                 20819 non-null  float64
dtypes: float64(1), object(9)
memory usage: 1.6+ MB
```

```
df.isnull().sum()
```

	0
Date	0
County	86
State	86
Vehicle Primary Use	0
Battery Electric Vehicles (BEVs)	0
Plug-In Hybrid Electric Vehicles (PHEVs)	0
Electric Vehicle (EV) Total	0
Non-Electric Vehicle Total	0
Total Vehicles	0
Percent Electric Vehicles	0

dtype: int64

```
# Compute Q1 and Q3
Q1 = df['Percent Electric Vehicles'].quantile(0.25)
Q3 = df['Percent Electric Vehicles'].quantile(0.75)
IQR = Q3 - Q1

# Define outlier boundaries
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
print('lower_bound:', lower_bound)
print('upper_bound:', upper_bound)

# Identify outliers
outliers = df[(df['Percent Electric Vehicles'] < lower_bound) | (df['Percent Electric Vehic
print("Number of outliers in 'Percent Electric Vehicles':", outliers.shape[0])
```

```
lower_bound: -3.5174999999999996
upper_bound: 6.9025
Number of outliers in 'Percent Electric Vehicles': 2476
```

```
# Converts the "Date" column to actual datetime objects
df['Date'] = pd.to_datetime(df['Date'], errors='coerce')

# Removes rows where "Date" conversion failed
df = df[df['Date'].notnull()]

# Removes rows where the target (EV Total) is missing
df = df[df['Electric Vehicle (EV) Total'].notnull()]

# Fill missing values
df['County'] = df['County'].fillna('Unknown')
df['State'] = df['State'].fillna('Unknown')

# Confirm remaining nulls
print("Missing after fill:")
print(df[['County', 'State']].isnull().sum())

df.head()
```

```
Missing after fill:
County    0
State     0
dtype: int64
```

	Date	County	State	Vehicle Primary Use	Battery Electric Vehicles (BEVs)	Plug-In Hybrid Electric Vehicles (PHEVs)	Electric Vehicle (EV) Total	Non-Electric Vehicle Total	Total Vehicles
0	2022-09-30	Riverside	CA	Passenger	7	0	7	460	467
1	2022-12-31	Prince William	VA	Passenger	1	2	3	188	191
2	2020-01-31	Dakota	MN	Passenger	0	1	1	32	33
3	2022-06-30	Ferry	WA	Truck	0	0	0	3,575	3,575
4	2021-07-31	Douglas	CO	Passenger	0	1	1	83	84

```
# Cap the outliers - it keeps all the data while reducing the skew from extreme values.

df['Percent Electric Vehicles'] = np.where(df['Percent Electric Vehicles'] > upper_bound, u
                                  np.where(df['Percent Electric Vehicles'] < lower_bound, lo

# Identify outliers
outliers = df[(df['Percent Electric Vehicles'] < lower_bound) | (df['Percent Electric Vehic
print("Number of outliers in 'Percent Electric Vehicles':", outliers.shape[0])
```

```
Number of outliers in 'Percent Electric Vehicles': 0
```