Car Price Estimation Code

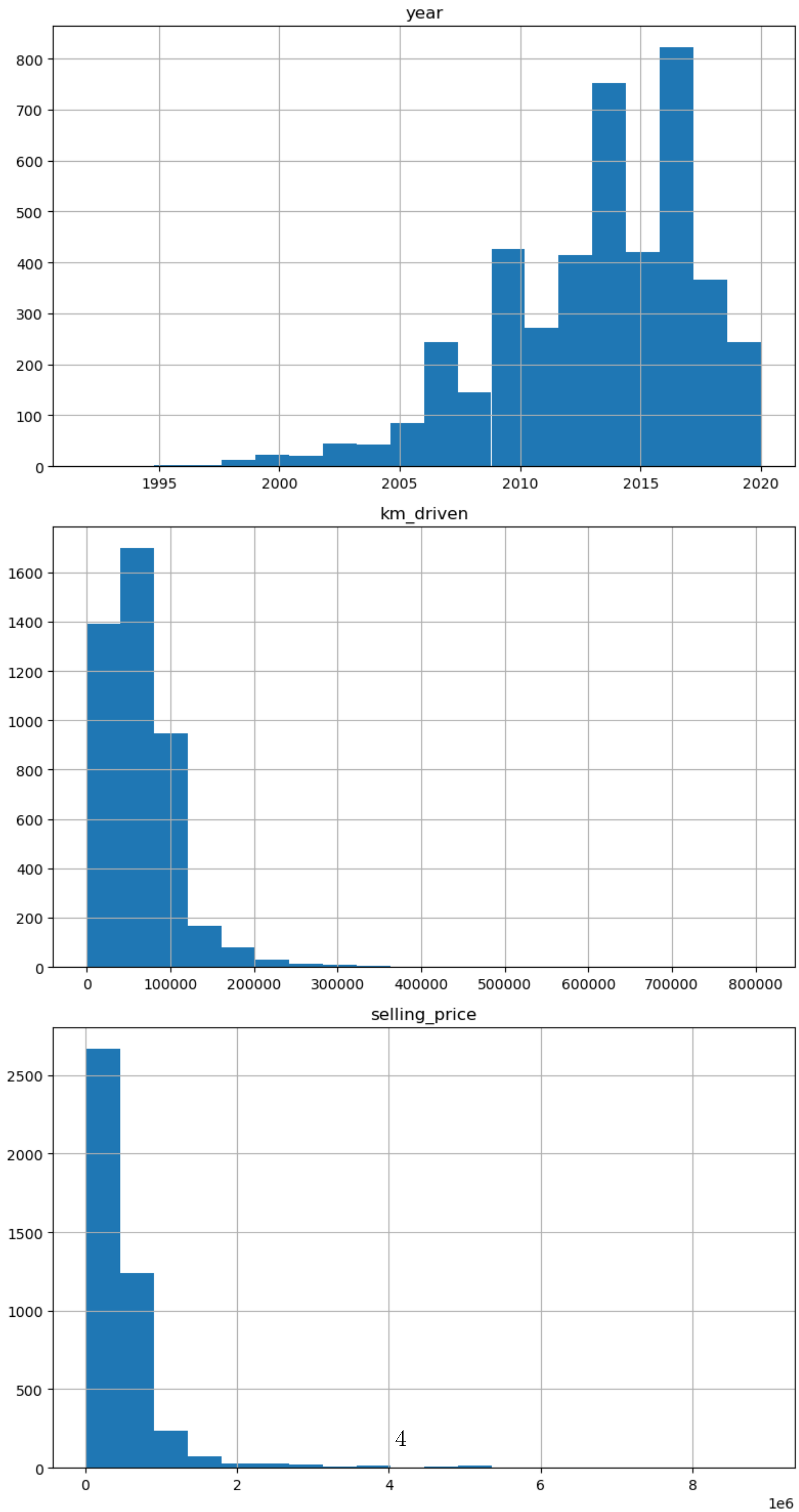
March 1, 2024

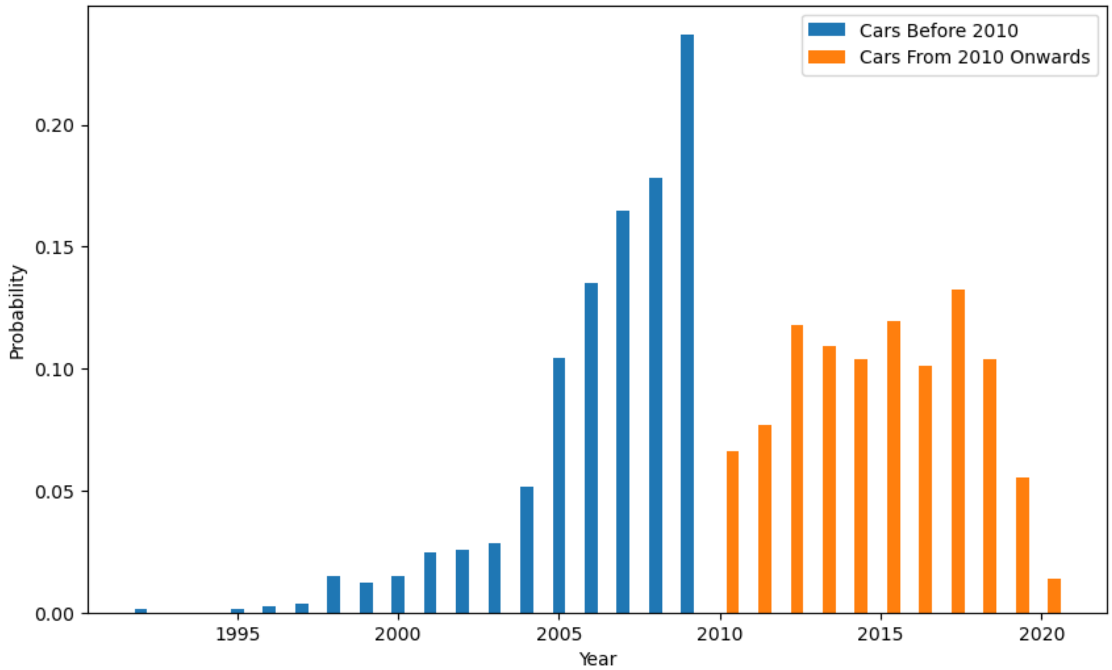
[2]:
```python
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, r2_score
import numpy as np
from statsmodels.distributions.empirical_distribution import ECDF

# Load the dataset
file_path = '/Users/nickblackford/Downloads/CAR DETAILS FROM CAR DEKHO.csv'
car_data = pd.read_csv(file_path)

# Convert categorical variables to numeric format using one-hot encoding
car_data_encoded = pd.get_dummies(car_data[['year', 'km_driven', 'fuel',␣
↪'transmission', 'selling_price']], drop_first=True)

# Histograms for each variable (excluding binary variables)
variables_for_histogram = ['year', 'km_driven', 'selling_price']
fig, axes = plt.subplots(len(variables_for_histogram), 1, figsize=(8, 15))
for i, var in enumerate(variables_for_histogram):
    car_data_encoded[var].hist(ax=axes[i], bins=20)
    axes[i].set_title(var)
plt.tight_layout()

# Descriptive statistics
variables = ['year', 'km_driven', 'selling_price', 'fuel_Diesel',␣
↪'transmission_Manual']
descriptive_stats = car_data_encoded[variables].describe().transpose()
descriptive_stats['mode'] = car_data_encoded[variables].mode().iloc[0]
print(descriptive_stats)

# PMF comparison
pmf_pre_2010 = car_data_encoded[car_data_encoded['year'] < 2010]['year'].
↪value_counts(normalize=True).sort_index()
pmf_2010_onwards = car_data_encoded[car_data_encoded['year'] >= 2010]['year'].
↪value_counts(normalize=True).sort_index()
```

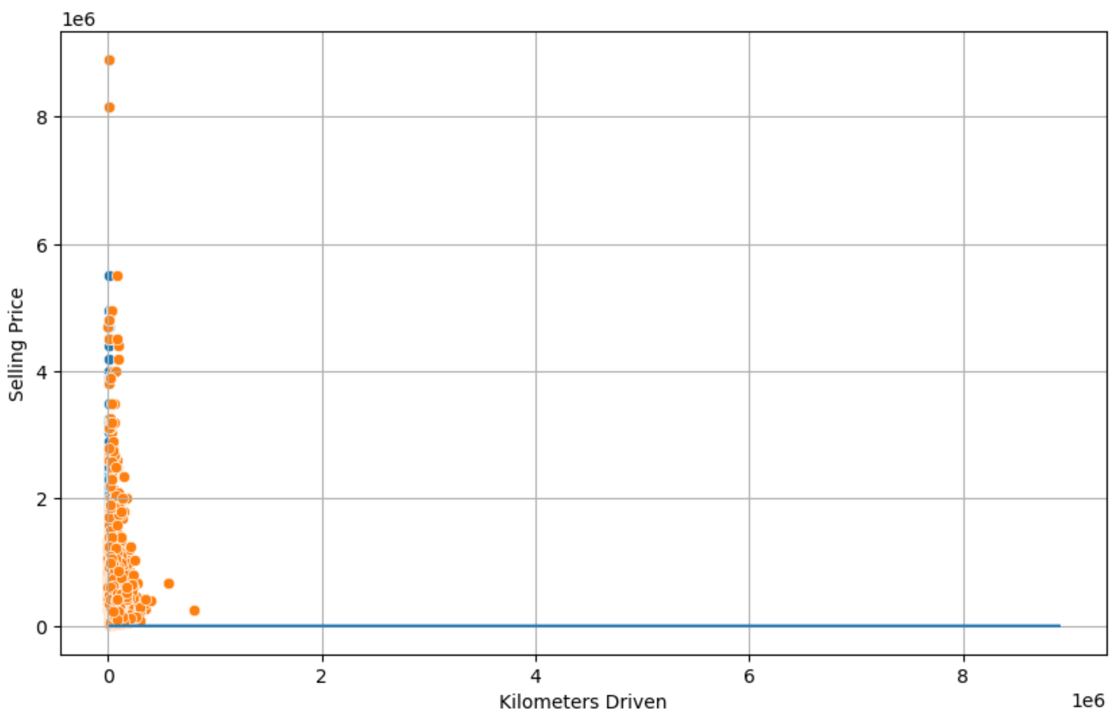
```python
plt.figure(figsize=(10, 6))
plt.bar(pmf_pre_2010.index, pmf_pre_2010.values, width=0.4, label='Cars Before 2010', align='center')
plt.bar(pmf_2010_onwards.index + 0.4, pmf_2010_onwards.values, width=0.4, label='Cars From 2010 Onwards', align='center')
plt.xlabel('Year')
plt.ylabel('Probability')
plt.legend()

# CDF of selling price
ecdf_selling_price = ECDF(car_data_encoded['selling_price'])
plt.figure(figsize=(10, 6))
plt.plot(ecdf_selling_price.x, ecdf_selling_price.y)
plt.xlabel('Selling Price')
plt.ylabel('CDF')
plt.grid(True)

# Scatter plots
sns.scatterplot(x='year', y='selling_price', data=car_data_encoded)
plt.xlabel('Year')
plt.ylabel('Selling Price')
sns.scatterplot(x='km_driven', y='selling_price', data=car_data_encoded)
plt.xlabel('Kilometers Driven')
plt.ylabel('Selling Price')

# Regression analysis
# Simple linear regression
X = car_data_encoded[['year']]
y = car_data_encoded['selling_price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print(f"Simple Linear Regression RMSE: {mean_squared_error(y_test, y_pred, squared=False)}")
print(f"R-squared: {r2_score(y_test, y_pred)}")

# Multiple linear regression
X_multi = car_data_encoded[['year', 'km_driven', 'fuel_Diesel', 'transmission_Manual']]
y_multi = car_data_encoded['selling_price']
X_train_multi, X_test_multi, y_train_multi, y_test_multi = train_test_split(X_multi, y_multi, test_size=0.2, random_state=42)
model_multi = LinearRegression()
model_multi.fit(X_train_multi, y_train_multi)
```

```python
y_pred_multi = model_multi.predict(X_test_multi)
print(f"Multiple Linear Regression RMSE: {mean_squared_error(y_test_multi,␣
↪y_pred_multi, squared=False)}")
print(f"R-squared: {r2_score(y_test_multi, y_pred_multi)}")
```

```
               count           mean            std      min        25%  \
year          4340.0    2013.090783       4.215344   1992.0    2011.00
km_driven     4340.0   66215.777419   46644.102194      1.0   35000.00
selling_price 4340.0  504127.311751  578548.736139  20000.0  208749.75

                   50%       75%        max    mode
year            2014.0    2016.0     2020.0    2017
km_driven      60000.0   90000.0   806599.0   70000
selling_price 350000.0  600000.0  8900000.0  300000
Simple Linear Regression RMSE: 505615.85076330346
R-squared: 0.1622795527600407
Multiple Linear Regression RMSE: 429812.45879370486
R-squared: 0.3946371643695866
```


year
800
700
600
500
400
300
200
100
0
1995
2000
2005
2010
2015
2020
km_driven
1600
1400
1200
1000
800
600
400
200
0
0
100000
200000
300000
400000
500000
600000
700000
800000
selling_price
2500
2000
1500
1000
500
0
0
2
4
6
8
1e6


Cars Before 2010
Cars From 2010 Onwards
0.00
0.05
0.10
0.15
0.20
Probability
1995
2000
2005
2010
2015
2020
Year


1e6
0
2
4
6
8
Selling Price
0
2
4
6
8
Kilometers Driven
1e6

[ ]:

```

```