```
from google.colab import files

# Upload the CSV file (e.g., data.csv)
uploaded = files.upload()
```

Choose Files data.csv

- **data.csv**(text/csv) - 1475504 bytes, last modified: 4/23/2025 - 100% done

```
Saving data.csv to data.csv
```

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.cluster import KMeans
from scipy.stats import ttest_ind

# Load the data
df = pd.read_csv('data.csv')

# View basic info
df.head()
```

| | Make | Model | Year | Engine Fuel Type | Engine HP | Engine Cylinders | Transmission Type | Driven_Wheels | Number of Doors | Market Category | Vehicle Size | Vehicle Style | highwa MP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | BMW | 1 Series M | 2011 | premium unleaded (required) | 335.0 | 6.0 | MANUAL | rear wheel drive | 2.0 | Factory Tuner,Luxury,High-Performance | Compact | Coupe | 2 |
| **1** | BMW | 1 Series | 2011 | premium unleaded (required) | 300.0 | 6.0 | MANUAL | rear wheel drive | 2.0 | Luxury,Performance | Compact | Convertible | 2 |
| **2** | BMW | 1 Series | 2011 | premium unleaded (required) | 300.0 | 6.0 | MANUAL | rear wheel drive | 2.0 | Luxury,High-Performance | Compact | Coupe | 2 |
| **3** | BMW | 1 Series | 2011 | premium unleaded (required) | 230.0 | 6.0 | MANUAL | rear wheel drive | 2.0 | Luxury,Performance | Compact | Coupe | 2 |
| **4** | BMW | 1 Series | 2011 | premium unleaded | 230.0 | 6.0 | MANUAL | rear wheel drive | 2.0 | Luxury | Compact | Convertible | 2 |

Next steps: Generate code with df View recommended plots New interactive sheet

```
#q-1
#H0 (Null Hypothesis): Engine HP does not significantly affect the car price (MSRP).
#H1 (Alternative Hypothesis): Higher Engine HP leads to higher MSRP.
```

```
#Q-2
import pandas as pd

df = pd.read_csv("data.csv")  # Upload this file in Colab using files.upload()

mean = df['Engine HP'].mean()
median = df['Engine HP'].median()
mode = df['Engine HP'].mode()[0]
std_dev = df['Engine HP'].std()

print("Mean:", mean)
print("Median:", median)
print("Mode:", mode)
print("Standard Deviation:", std_dev)
```

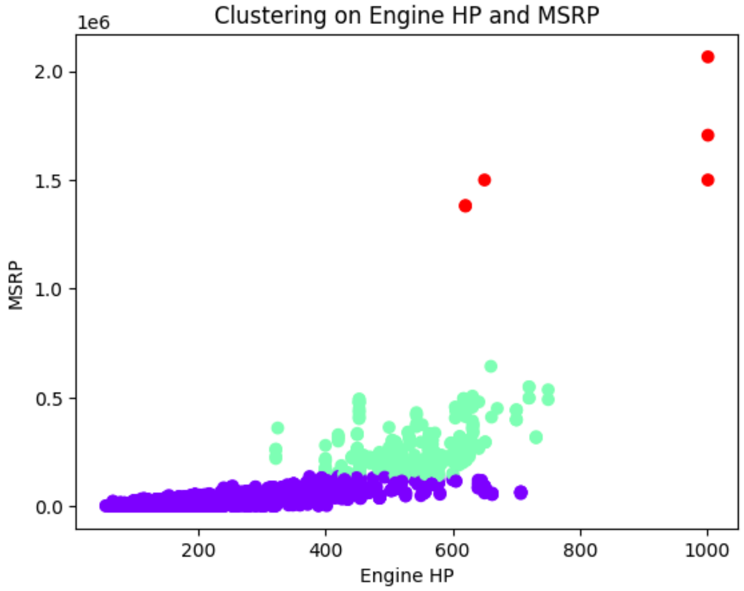
```
Mean: 249.38607007176023
Median: 227.0
Mode: 200.0
Standard Deviation: 109.19187025917257
```

```
#Q-3
print(df.groupby('Vehicle Size')['MSRP'].mean())
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
```

```
X = df[['Engine HP', 'MSRP']].dropna()
kmeans = KMeans(n_clusters=3)
X['Cluster'] = kmeans.fit_predict(X)

plt.scatter(X['Engine HP'], X['MSRP'], c=X['Cluster'], cmap='rainbow')
plt.xlabel("Engine HP")
plt.ylabel("MSRP")
plt.title("Clustering on Engine HP and MSRP")
plt.show()
```

```
Vehicle Size
Compact     34275.336482
Large       53890.500540
Midsize     39035.919049
Name: MSRP, dtype: float64
```



```
#Q-4 Apply Regression Analysi
#from sklearn.linear_model import LinearRegression

df_reg = df[['Engine HP', 'MSRP']].dropna()
X = df_reg[['Engine HP']]
y = df_reg['MSRP']

model = LinearRegression()
model.fit(X, y)

print("Regression Coefficient:", model.coef_[0])
print("Intercept:", model.intercept_)
```

```
Regression Coefficient: 365.28835618918765
Intercept: -50550.63198303285
```

```
#q-5 Perform t-test for Hypothesis Validation
from scipy.stats import ttest_ind

high_hp = df[df['Engine HP'] > 300]['MSRP'].dropna()
low_hp = df[df['Engine HP'] <= 300]['MSRP'].dropna()

t_stat, p_value = ttest_ind(high_hp, low_hp)
print("T-Test Result:\nt-stat =", t_stat, "\np-value =", p_value)
```

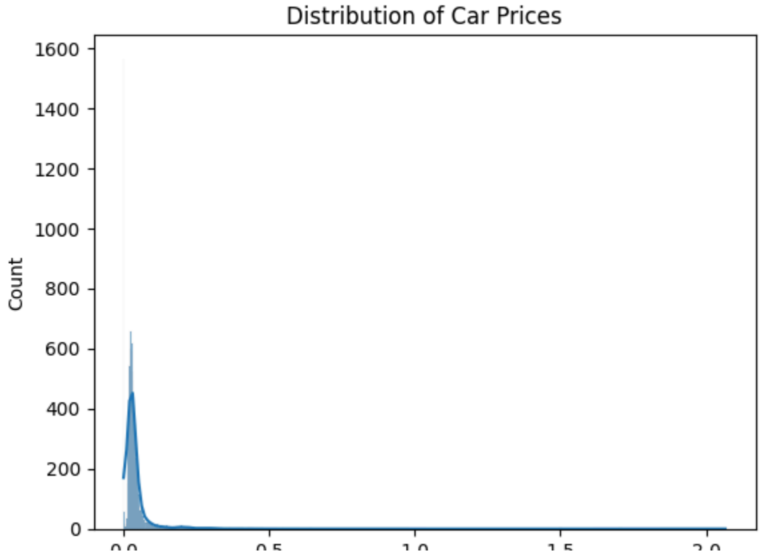
```
T-Test Result:
t-stat = 56.132118469255374
p-value = 0.0
```

```
#Q-6 Visualize Data using seaborn/matplotlib
import seaborn as sns
import matplotlib.pyplot as plt

# Histogram of MSRP
sns.histplot(df['MSRP'], kde=True)
plt.title("Distribution of Car Prices")
plt.show()
```

```
# Boxplot of MSRP by Vehicle Size
sns.boxplot(x='Vehicle Size', y='MSRP', data=df)
plt.title("Car Price by Vehicle Size")
plt.show()

# Regression Plot: Engine HP vs MSRP
sns.regplot(x='Engine HP', y='MSRP', data=df, scatter_kws={'alpha':0.4})
plt.title("Engine HP vs MSRP")
plt.show()
```


Distribution of Car Prices
Count
1600
1400
1200
1000
800
600
400
200
0
0.0
0.5
1.0
1.5
2.0