assignment3

October 23, 2023

1 Data Exploration (10 points):

1.0.1 Load the "mtcars" dataset and describe its structure, including the number of observations and variables.

1.0.2 Explore the dataset by calculating summary statistics and visualizing the data. Create scatter plots to examine the relationships between the independent variables and the target variable (mpg).

[47]:
```
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
from sklearn.metrics import mean_squared_error, r2_score
import seaborn as sb
from scipy import stats
import numpy as np
data = pd.read_csv('mtcars.csv')

# Load the "mtcars" dataset and describe its structure, including the number of
 ↪observations and variables.
data.head()
```

[47]:
```
                model   mpg  cyl   disp   hp  drat     wt   qsec  vs  am  gear  \
0           Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1     4
1       Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1     4
2          Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1     4
3      Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0     3
4   Hornet Sportabout  18.7    8  360.0  175  3.15  3.440  17.02   0   0     3

   carb
0     4
1     4
2     1
3     1
4     2
```

[48]:
```
data.shape
```

[48]:
```
(32, 12)
```

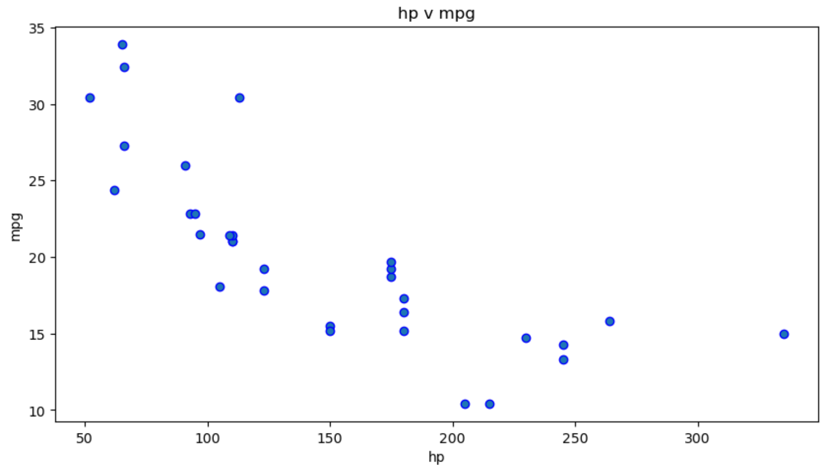
(32, 12) meaning 32 Observations(Rows) and 12 Characteristics(Columns)

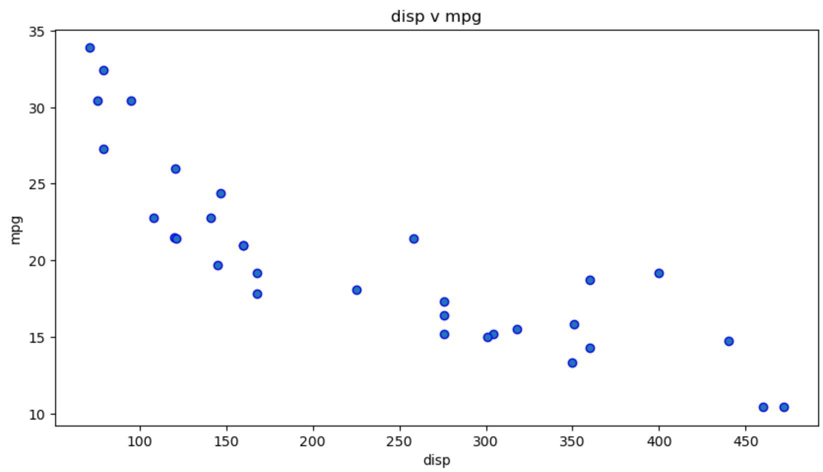
[49]:
```
# Explore the dataset by calculating summary statistics and visualizing the␣
↪data.
data.describe
```

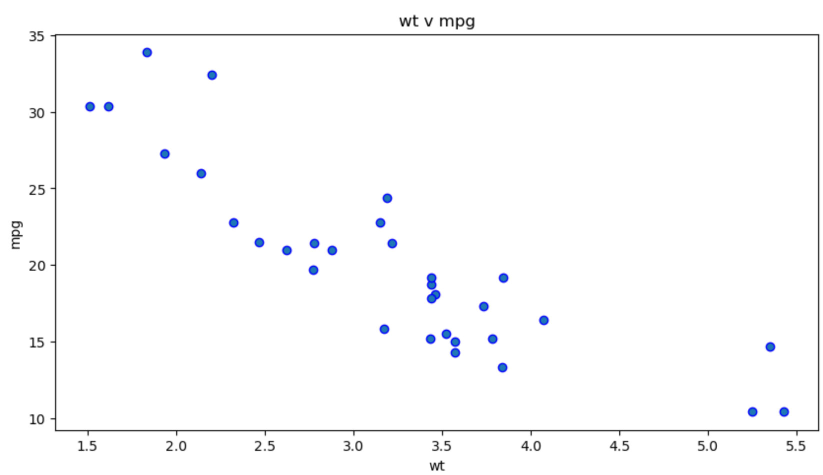
[49]:
```
<bound method NDFrame.describe of                      model   mpg  cyl   disp   hp
drat     wt   qsec  vs   am  \
0             Mazda RX4  21.0    6  160.0  110  3.90  2.620  16.46   0   1
1         Mazda RX4 Wag  21.0    6  160.0  110  3.90  2.875  17.02   0   1
2            Datsun 710  22.8    4  108.0   93  3.85  2.320  18.61   1   1
3        Hornet 4 Drive  21.4    6  258.0  110  3.08  3.215  19.44   1   0
4     Hornet Sportabout  18.7    8  360.0  175  3.15  3.440  17.02   0   0
5               Valiant  18.1    6  225.0  105  2.76  3.460  20.22   1   0
6            Duster 360  14.3    8  360.0  245  3.21  3.570  15.84   0   0
7             Merc 240D  24.4    4  146.7   62  3.69  3.190  20.00   1   0
8              Merc 230  22.8    4  140.8   95  3.92  3.150  22.90   1   0
9              Merc 280  19.2    6  167.6  123  3.92  3.440  18.30   1   0
10            Merc 280C  17.8    6  167.6  123  3.92  3.440  18.90   1   0
11           Merc 450SE  16.4    8  275.8  180  3.07  4.070  17.40   0   0
12           Merc 450SL  17.3    8  275.8  180  3.07  3.730  17.60   0   0
13          Merc 450SLC  15.2    8  275.8  180  3.07  3.780  18.00   0   0
14   Cadillac Fleetwood  10.4    8  472.0  205  2.93  5.250  17.98   0   0
15  Lincoln Continental  10.4    8  460.0  215  3.00  5.424  17.82   0   0
16    Chrysler Imperial  14.7    8  440.0  230  3.23  5.345  17.42   0   0
17             Fiat 128  32.4    4   78.7   66  4.08  2.200  19.47   1   1
18          Honda Civic  30.4    4   75.7   52  4.93  1.615  18.52   1   1
19       Toyota Corolla  33.9    4   71.1   65  4.22  1.835  19.90   1   1
20        Toyota Corona  21.5    4  120.1   97  3.70  2.465  20.01   1   0
21     Dodge Challenger  15.5    8  318.0  150  2.76  3.520  16.87   0   0
22          AMC Javelin  15.2    8  304.0  150  3.15  3.435  17.30   0   0
23           Camaro Z28  13.3    8  350.0  245  3.73  3.840  15.41   0   0
24     Pontiac Firebird  19.2    8  400.0  175  3.08  3.845  17.05   0   0
25            Fiat X1-9  27.3    4   79.0   66  4.08  1.935  18.90   1   1
26        Porsche 914-2  26.0    4  120.3   91  4.43  2.140  16.70   0   1
27         Lotus Europa  30.4    4   95.1  113  3.77  1.513  16.90   1   1
28       Ford Pantera L  15.8    8  351.0  264  4.22  3.170  14.50   0   1
29         Ferrari Dino  19.7    6  145.0  175  3.62  2.770  15.50   0   1
30        Maserati Bora  15.0    8  301.0  335  3.54  3.570  14.60   0   1
31           Volvo 142E  21.4    4  121.0  109  4.11  2.780  18.60   1   1

    gear  carb
0      4     4
```

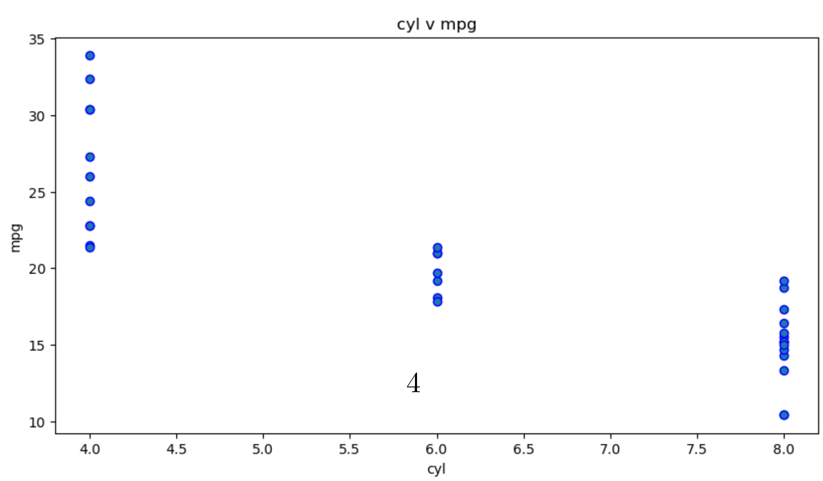
1	4	4
2	4	1
3	3	1
4	3	2
5	3	1
6	3	4
7	4	2
8	4	2
9	4	4
10	4	4
11	3	3
12	3	3
13	3	3
14	3	4
15	3	4
16	3	4
17	4	1
18	4	2
19	4	1
20	3	1
21	3	2
22	3	2
23	3	4
24	3	2
25	4	1
26	5	2
27	5	2
28	5	4
29	5	6
30	5	8
31	4	2 >

[50]:
```
# Create scatter plots to examine the relationships between the independent␣
↪variables and the target variable (mpg).

variables = ['hp','disp','wt','cyl']

# Create a figure with subplots
fig, axes = plt.subplots(len(variables), 1, figsize=(10, 6*len(variables)))

for i, var in enumerate(variables):
    axes[i].scatter(x=var, y='mpg', data=data, edgecolor='b')
    axes[i].set_xlabel(var)
    axes[i].set_ylabel('mpg')
    axes[i].set_title(f'{var} v mpg')

plt.show()
```


hp v mpg
35
30
25
20
15
10
mpg
50
100
150
200
250
300
hp


disp v mpg
35
30
25
20
15
10
mpg
100
150
200
250
300
350
400
450
disp


wt v mpg
35
30
25
20
15
10
mpg
1.5
2.0
2.5
3.0
3.5
4.0
4.5
5.0
5.5
wt


cyl v mpg
35
30
25
20
15
10
mpg
4.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
cyl

As a result we can see that the lower the HorsePower the higher the MPG as the car isn't burning as much. We can see a similar pattern with Displacement, Weight and Cylinder count in relation to the MPG

2 Simple Linear Regression (30 points):

2.0.1 Select one independent variable from the "mtcars" dataset that you believe may have a strong linear relationship with the target variable (mpg).

2.0.2 Implement a simple linear regression model to predict mpg using the selected independent variable.

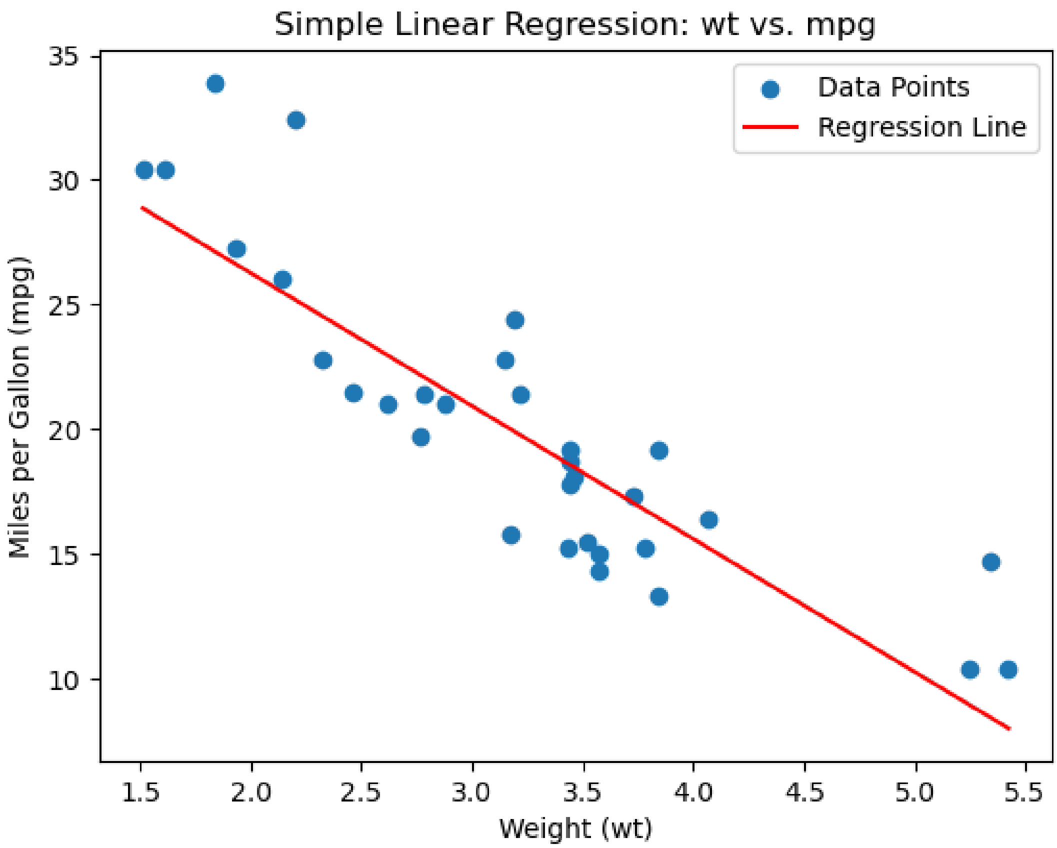
2.0.3 Calculate the model's coefficients (slope and intercept) and evaluate its performance using appropriate regression evaluation metrics (on testing dataset).

[51]:
```
# Select independent variable (X) and target variable (y)
X = data[['wt']]
y = data['mpg']

# Split the data into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)

# Initialize the linear regression model
model = LinearRegression()

# Fit the model on the training data
model.fit(X_train, y_train)

# Calculate the coefficients (slope and intercept)
slope = model.coef_[0]
intercept = model.intercept_

# Predict the target variable on the test set
y_pred = model.predict(X_test)

# Calculate evaluation metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print the coefficients and evaluation metrics
print(f'Slope (Coefficient): {slope}')
print(f'Intercept: {intercept}')
print(f'Mean Squared Error (MSE): {mse}')
print(f'R-squared (R2) Score: {r2}')
```

```
Slope (Coefficient): -5.336941400557082
Intercept: 36.93731031351841
Mean Squared Error (MSE): 12.475985659918848
R-squared (R2) Score: 0.6879761857596269
```

[52]:
```python
# Points for the regression line
x_range = np.linspace(X.min(), X.max(), 100).reshape(-1, 1)
y_range = model.predict(x_range)

# Scatter plot
plt.scatter(X, y, label='Data Points')

# Regression line Plot
plt.plot(x_range, y_range, color='red', label='Regression Line')

# Add labels and legend
plt.xlabel('Weight (wt)')
plt.ylabel('Miles per Gallon (mpg)')
plt.title('Simple Linear Regression: wt vs. mpg')
plt.legend()

# Show plot
plt.show()
```

```
c:\ProgramData\anaconda3\Lib\site-packages\sklearn\base.py:464: UserWarning: X
does not have valid feature names, but LinearRegression was fitted with feature
names
  warnings.warn(
```



3 Multiple Linear Regression (40 points):

3.0.1 Implement a multiple linear regression model using a combination of independent variables from the "mtcars" dataset.

3.0.2 Train the model to predict mpg using multiple features.

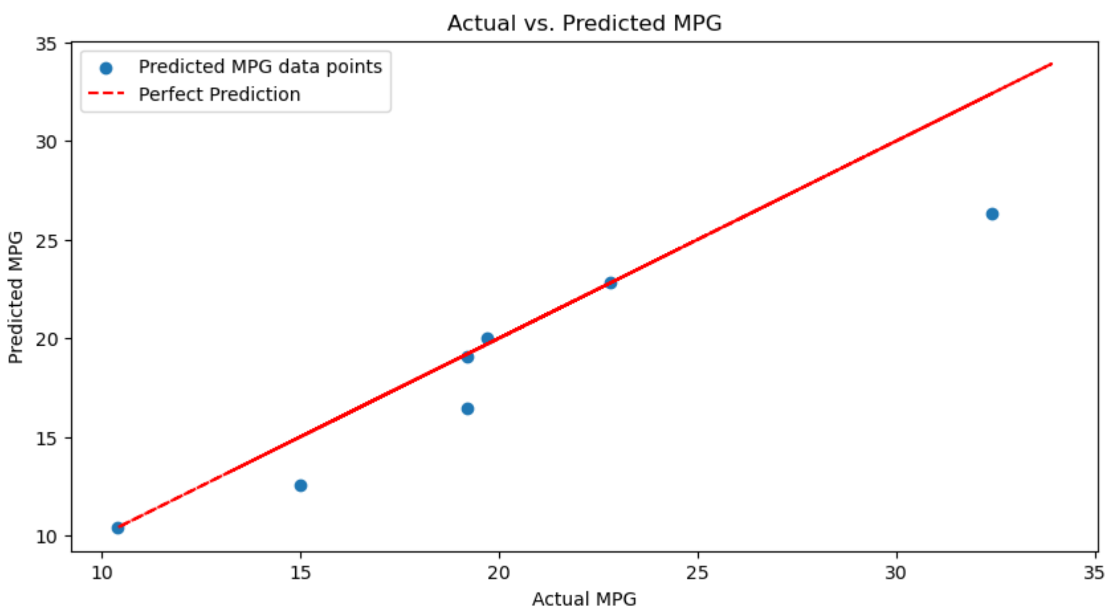
3.0.3 Evaluate the model's performance using appropriate regression evaluation metrics (on testing dataset).

[53]:
```
# Select independent variables (X) and target variable (y)
X = data[['disp', 'hp', 'wt', 'cyl']]  # Using all available features
y = data['mpg']

# Split the data into training and testing sets (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Initialize the linear regression model
model = LinearRegression()
```

```python
# Fit the model on the training data
model.fit(X_train, y_train)

# Predict the target variable on the test set
y_pred = model.predict(X_test)

# Calculate evaluation metrics
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

# Print evaluation metrics
print(f'Mean Squared Error (MSE): {mse}')
print(f'R-squared (R2) Score: {r2}')
```

```
Mean Squared Error (MSE): 7.18935828323054
R-squared (R2) Score: 0.8201944876643273
```

[56]:
```python
# Plot actual vs. predicted to visualise the predictions made by the mlr model
plt.figure(figsize=(10, 5))
plt.scatter(y_test, y_pred, label='Predicted MPG data points')
plt.plot(y, y, color='red', linestyle='--', label='Perfect Prediction')
plt.xlabel("Actual MPG")
plt.ylabel("Predicted MPG")
plt.title("Actual vs. Predicted MPG")
plt.legend()
plt.show()
```



As we can see the models predictions here are more accurate than the SLR with significantly less outliers

4 Discussion and Conclusion 20 points:

4.0.1 Compare the performance and interpretability of the simple linear regression model with the multiple linear regression model. Discuss the trade-offs between simplicity and complexity.

4.0.2 Reflect on the insights gained from the assignment and the implications for predicting fuel efficiency in car models.

4.1 Simple Linear Regression (SLR) Results:

4.1.1 Slope (Coefficient): -5.337

4.1.2 Intercept: 36.937

4.1.3 Mean Squared Error (MSE): 12.476

4.1.4 R-squared (R2) Score: 0.688

4.2 Multiple Linear Regression (MLR) Results:

4.2.1 Mean Squared Error (MSE): 7.189

4.2.2 R-squared (R2) Score: 0.820

4.2.3 Simplicity:

The SLR model is simpler with one independent variable compared to the MLR which is more comples and incomporates multiple independent variables.

4.2.4 Performance:

The MLR model outperforms the SLR model in terms of predictive accuracy. It has a lower MSE (7.189 vs. 12.476) and a higher R-squared score (0.820 vs. 0.688). This indicates that the MLR model provides a better fit to the data.

4.2.5 In Practice:

In practice, if simplicity and interpretability are crucial (e.g., for decision-making in the automotive industry), the SLR model might be preferred. It provides a clear understanding of how weight influences fuel efficiency. If maximizing predictive accuracy is the primary goal and interpretability is less critical, the MLR model might be chosen.

4.3 Insights and Implications:

4.3.1 Feature Selection:

The inclusion of additional features (disp, hp, cyl) in the MLR model significantly improves accuracy meaning that considering multiple factors is important for accurately predicting fuel efficiency.

4.3.2 Model Evaluation:

The MLR model demonstrated superior performance in this case. However, it's crucial to validate its performance on a broader range of data to ensure its generalizability.

4.3.3 Trade-offs:

The choice between simplicity (SLR) and complexity (MLR) depends on the specific goals and requirements of the problem. Balancing model complexity with interpretability is crucial.

Overall, this analysis highlights the importance of considering model complexity, interpretability, and performance when building predictive models, especially in domains like the automotive industry where accurate predictions are essential for decision-making, advertisements and price predictions.