Technological Institute of the Philippines

Computer Engineering Department

Quezon city Campus

Seatwork 11.1 Exploratory Data Analysis for Machine Learning

Course: CPE 311	Program: BSCpE
Course Title: Computational Thinking with Python	**Date Performed:** April 27 , 2024
Section: BSCPE22S3	**Date Submitted:** April 27, 2024
Student Name: Juliann Vincent B. Quibral	**Instructor's Name:** Engr. Roman Richard

Linear Regression Analysis

```
pip install ucimlrepo
```

```
Collecting ucimlrepo
  Downloading ucimlrepo-0.0.6-py3-none-any.whl (8.0 kB)
Installing collected packages: ucimlrepo
Successfully installed ucimlrepo-0.0.6
```

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from sklearn.preprocessing import StandardScaler
from scipy import stats
```

```
from ucimlrepo import fetch_ucirepo
```

```
# fetch dataset
automobile = fetch_ucirepo(id=10)

# data (as pandas dataframes)
X = automobile.data.features
y = automobile.data.targets

# metadata
print(automobile.metadata)

# variable information
print(automobile.variables)
```

```
{'uci_id': 10, 'name': 'Automobile', 'repository_url': 'https://archive.ics.uci.edu/dataset/10/automobile', 'data_ur
                 name     role         type demographic  \
0               price  Feature   Continuous        None
1         highway-mpg  Feature   Continuous        None
2            city-mpg  Feature   Continuous        None
3            peak-rpm  Feature   Continuous        None
4          horsepower  Feature   Continuous        None
5   compression-ratio  Feature   Continuous        None
6              stroke  Feature   Continuous        None
7                bore  Feature   Continuous        None
8         fuel-system  Feature  Categorical        None
9         engine-size  Feature   Continuous        None
10   num-of-cylinders  Feature      Integer        None
11        engine-type  Feature  Categorical        None
12        curb-weight  Feature   Continuous        None
13             height  Feature   Continuous        None
14              width  Feature   Continuous        None
15             length  Feature   Continuous        None
16         wheel-base  Feature   Continuous        None
17    engine-location  Feature       Binary        None
18       drive-wheels  Feature  Categorical        None
19         body-style  Feature  Categorical        None
20       num-of-doors  Feature      Integer        None
21         aspiration  Feature       Binary        None
22          fuel-type  Feature       Binary        None
23               make  Feature  Categorical        None
24  normalized-losses  Feature   Continuous        None
```

```
25          symboling   Target      Integer        None
```

		description	units	missing_values
0		continuous from 5118 to 45400	None	yes
1		continuous from 16 to 54	None	no
2		continuous from 13 to 49	None	no
3		continuous from 4150 to 6600	None	yes
4		continuous from 48 to 288	None	yes
5		continuous from 7 to 23	None	no
6		continuous from 2.07 to 4.17	None	yes
7		continuous from 2.54 to 3.94	None	yes
8	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi		None	no
9		continuous from 61 to 326	None	no
10	eight, five, four, six, three, twelve, two		None	no
11	dohc, dohcv, l, ohc, ohcf, ohcv, rotor		None	no
12		continuous from 1488 to 4066	None	no
13		continuous from 47.8 to 59.8	None	no
14		continuous from 60.3 to 72.3	None	no
15		continuous from 141.1 to 208.1	None	no
16		continuous from 86.6 120.9	None	no
17		front, rear	None	no
18		4wd, fwd, rwd	None	no
19	hardtop, wagon, sedan, hatchback, convertible		None	no
20		four, two	None	yes
21		std, turbo	None	no
22		diesel, gas	None	no
23	alfa-romero, audi, bmw, chevrolet, dodge, hond...		None	no
24		continuous from 65 to 256	None	yes
25		-3, -2, -1, 0, 1, 2, 3	None	no

1 X.dtypes

```
price               float64
highway-mpg           int64
city-mpg              int64
peak-rpm            float64
horsepower          float64
compression-ratio   float64
stroke              float64
bore                float64
```

```
fuel-system           object
engine-size            int64
num-of-cylinders       int64
engine-type           object
curb-weight            int64
height               float64
width                float64
length               float64
wheel-base           float64
engine-location       object
drive-wheels          object
body-style            object
num-of-doors         float64
aspiration            object
fuel-type             object
make                  object
normalized-losses    float64
dtype: object
```

```
1 print(automobile.metadata)
```

```
{'uci_id': 10, 'name': 'Automobile', 'repository_url': 'https://archive.ics.uci.edu/dataset/10/automobile', 'data_url':
```

```
1 print(automobile.variables)
```

```
                 name     role         type demographic  \
0               price  Feature   Continuous        None
1         highway-mpg  Feature   Continuous        None
2            city-mpg  Feature   Continuous        None
3            peak-rpm  Feature   Continuous        None
4          horsepower  Feature   Continuous        None
5   compression-ratio  Feature   Continuous        None
6              stroke  Feature   Continuous        None
7                bore  Feature   Continuous        None
8         fuel-system  Feature  Categorical        None
9         engine-size  Feature   Continuous        None
10  num-of-cylinders  Feature      Integer        None
11        engine-type  Feature  Categorical        None
12        curb-weight  Feature   Continuous        None
```

13	height	Feature	Continuous	None			
14	width	Feature	Continuous	None			
15	length	Feature	Continuous	None			
16	wheel-base	Feature	Continuous	None			
17	engine-location	Feature	Binary	None			
18	drive-wheels	Feature	Categorical	None			
19	body-style	Feature	Categorical	None			
20	num-of-doors	Feature	Integer	None			
21	aspiration	Feature	Binary	None			
22	fuel-type	Feature	Binary	None			
23	make	Feature	Categorical	None			
24	normalized-losses	Feature	Continuous	None			
25	symboling	Target	Integer	None			

	description	units	missing_values
0	continuous from 5118 to 45400	None	yes
1	continuous from 16 to 54	None	no
2	continuous from 13 to 49	None	no
3	continuous from 4150 to 6600	None	yes
4	continuous from 48 to 288	None	yes
5	continuous from 7 to 23	None	no
6	continuous from 2.07 to 4.17	None	yes
7	continuous from 2.54 to 3.94	None	yes
8	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi	None	no
9	continuous from 61 to 326	None	no
10	eight, five, four, six, three, twelve, two	None	no
11	dohc, dohcv, l, ohc, ohcf, ohcv, rotor	None	no
12	continuous from 1488 to 4066	None	no
13	continuous from 47.8 to 59.8	None	no
14	continuous from 60.3 to 72.3	None	no
15	continuous from 141.1 to 208.1	None	no
16	continuous from 86.6 120.9	None	no
17	front, rear	None	no
18	4wd, fwd, rwd	None	no
19	hardtop, wagon, sedan, hatchback, convertible	None	no
20	four, two	None	yes
21	std, turbo	None	no
22	diesel, gas	None	no
23	alfa-romero, audi, bmw, chevrolet, dodge, hond...	None	no
24	continuous from 65 to 256	None	yes
25	-3, -2, -1, 0, 1, 2, 3	None	no

```
print(automobile.data.columns)
```

```
None
```

```
from ucimlrepo import fetch_ucirepo

# Fetch dataset
automobile = fetch_ucirepo(id=10)

if automobile is None:
    print("Error: Dataset loading failed.")
else:
    print("Dataset loaded successfully.")
```

```
Dataset loaded successfully.
```

```
if automobile is not None:
    print("Dataset columns:")
    print(automobile.data.columns)
    print("\nDataset metadata:")
    print(automobile.metadata)
```

```
Dataset columns:
None

Dataset metadata:
{'uci_id': 10, 'name': 'Automobile', 'repository_url': 'https://archive.ics.uci.edu/dataset/10/automobile', 'data_url':
```

Load the dataset

```
automobile = fetch_ucirepo(id=10)
X = automobile.data.features
y = automobile.data.target
```

Data Preprocessing/Wraingling

Check for missing values:

```
1 print(X_encoded.isnull().sum())
```

```
price                   4
highway-mpg             0
city-mpg                0
peak-rpm                2
horsepower              2
                       ..
make_toyota             0
make_volkswagen         0
make_volvo              0
engine-location_front   0
engine-location_rear    0
Length: 68, dtype: int64
```

Encoding Categorical Variables:

```
1 X_encoded = pd.get_dummies(X, columns=['fuel-system', 'engine-type', 'drive-wheels', 'body-style', 'aspiration', 'fuel-t
2
3 print(X_encoded.shape)
4 print(X_encoded.dtypes)
```

```
(205, 68)
price                  float64
highway-mpg              int64
city-mpg                 int64
peak-rpm               float64
horsepower             float64
                        ...
make_toyota               bool
```

```
make_volkswagen             bool
make_volvo                  bool
engine-location_front       bool
engine-location_rear        bool
Length: 68, dtype: object
```

Feature Scaling:

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

numeric_columns = X_encoded_full.select_dtypes(include=['float64', 'int64']).columns
X_scaled = X_encoded_full.copy()
X_scaled[numeric_columns] = scaler.fit_transform(X_encoded_full[numeric_columns])
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:768: UserWarning: pandas.DataFrame with sparse colu
  warnings.warn(
/usr/local/lib/python3.10/dist-packages/sklearn/utils/validation.py:768: UserWarning: pandas.DataFrame with sparse colu
  warnings.warn(
```

Exploratory Data Analysis (EDA):

Descriptive Statistics:

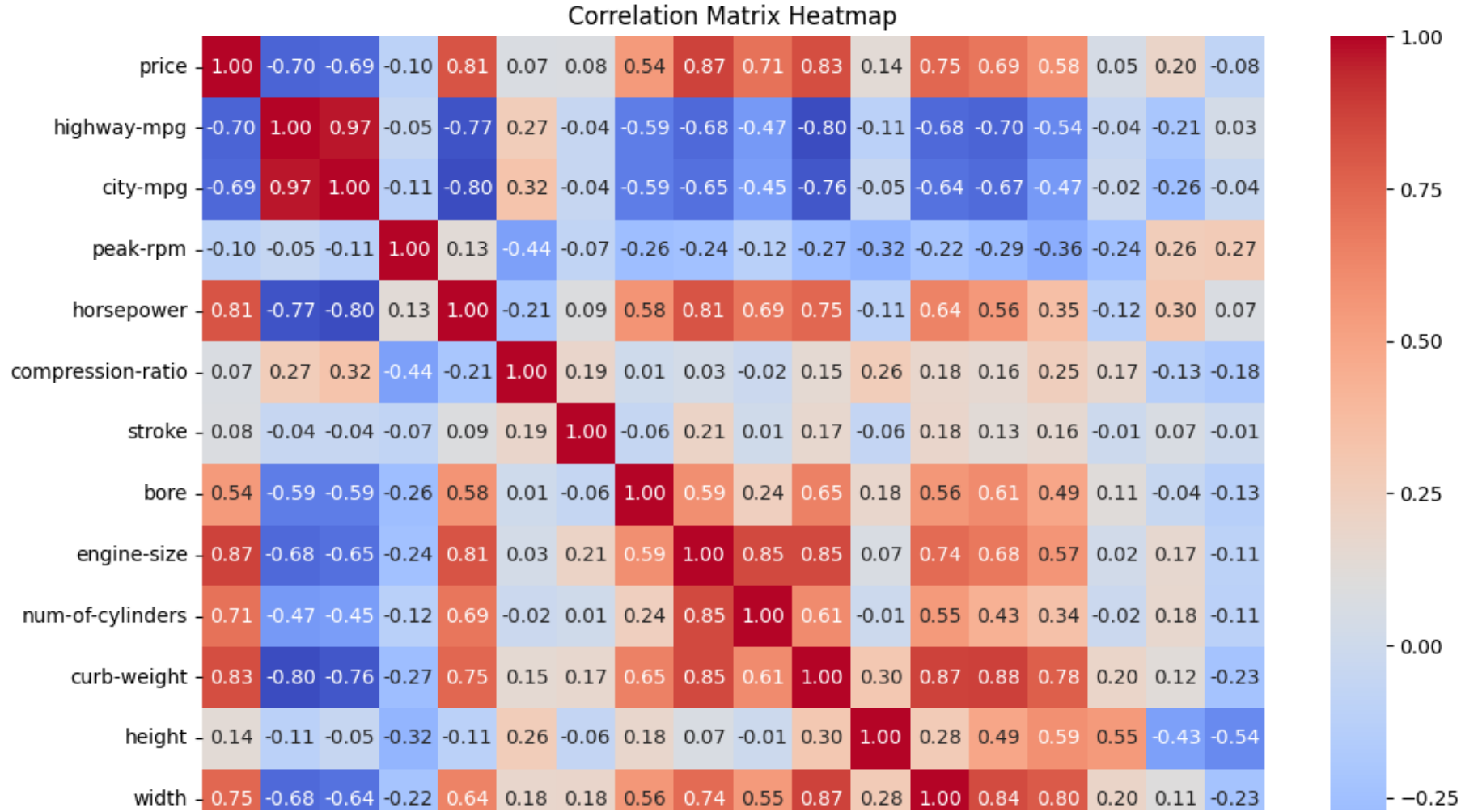
```
X.describe()
```

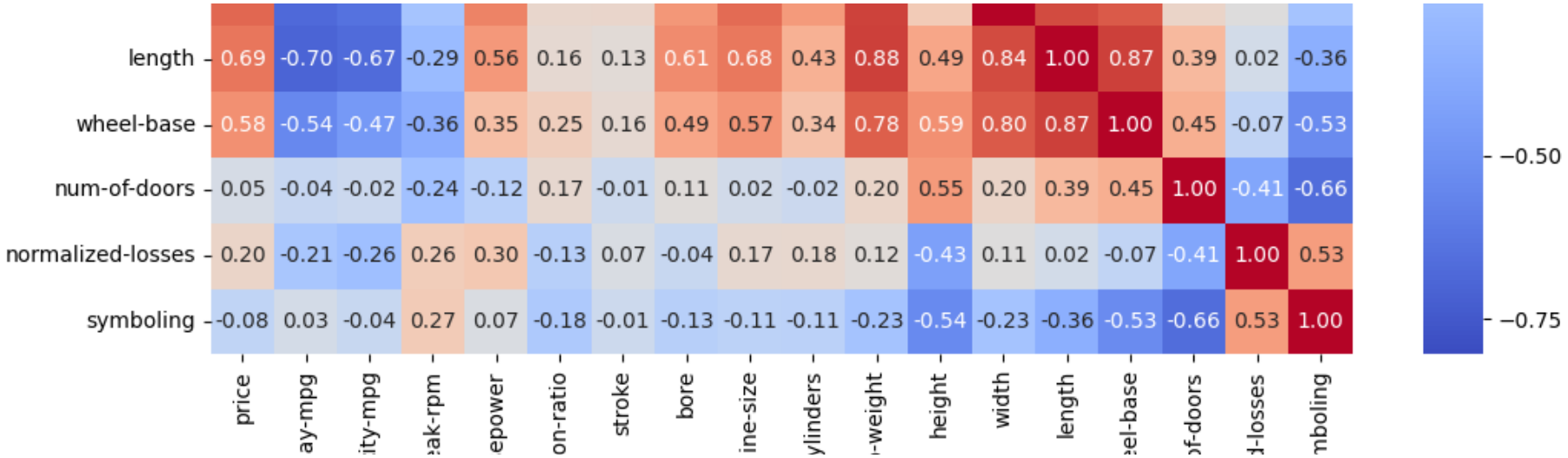
	price	highway-mpg	city-mpg	peak-rpm	horsepower	compression-ratio	stroke	bore	engine-size	cy
count	201.000000	205.000000	205.000000	203.000000	203.000000	205.000000	201.000000	201.000000	205.000000	20
mean	13207.129353	30.751220	25.219512	5125.369458	104.256158	10.142537	3.255423	3.329751	126.907317	
std	7947.066342	6.886443	6.542142	479.334560	39.714369	3.972040	0.316717	0.273539	41.642693	
min	5118.000000	16.000000	13.000000	4150.000000	48.000000	7.000000	2.070000	2.540000	61.000000	
25%	7775.000000	25.000000	19.000000	4800.000000	70.000000	8.600000	3.110000	3.150000	97.000000	
50%	10295.000000	30.000000	24.000000	5200.000000	95.000000	9.000000	3.290000	3.310000	120.000000	
75%	16500.000000	34.000000	30.000000	5500.000000	116.000000	9.400000	3.410000	3.590000	141.000000	
max	45400.000000	54.000000	49.000000	6600.000000	288.000000	23.000000	4.170000	3.940000	326.000000	1

Correlation Analysis:

```
print(data_combined.dtypes)

data_numeric = data_combined.select_dtypes(include=['float64', 'int64'])

correlation_matrix = data_numeric.corr()

plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title("Correlation Matrix Heatmap")
plt.show()
```

```
height              float64
width               float64
length              float64
wheel-base          float64
engine-location      object
drive-wheels         object
body-style           object
num-of-doors        float64
aspiration           object
fuel-type            object
make                 object
normalized-losses   float64
symboling             int64
dtype: object
```





Simple Linear Regression:

Split Data into Training and Testing Sets:

```
1 from sklearn.model_selection import train_test_split
2
3 X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2, random_state=42)
```

```
from sklearn.linear_model import LinearRegression

model = LinearRegression()

model.fit(X_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

Fit Linear Regression Model:

```
from sklearn.metrics import mean_squared_error, r2_score

# Make predictions
y_pred = model.predict(X_test)

# Evaluate the model
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print(f"Mean Squared Error: {mse}")
print(f"R-squared: {r2}")
```

```
Mean Squared Error: 0.6859129086119733
R-squared: 0.5320537340191854
```

Logistic Regression Analysis

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
from ucimlrepo import fetch_ucirepo

# fetch dataset
wine = fetch_ucirepo(id=109)

# data (as pandas dataframes)
X = wine.data.features
y = wine.data.targets

# metadata
print(wine.metadata)

# variable information
print(wine.variables)
```

```
{'uci_id': 109, 'name': 'Wine', 'repository_url': 'https://archive.ics.uci.edu/dataset/109/wine', 'data_url': 'https://
                          name     role         type demographic  \
0                        class   Target  Categorical        None
1                      Alcohol  Feature   Continuous        None
2                    Malicacid  Feature   Continuous        None
3                          Ash  Feature   Continuous        None
4              Alcalinity_of_ash  Feature   Continuous        None
5                    Magnesium  Feature      Integer        None
6                Total_phenols  Feature   Continuous        None
7                   Flavanoids  Feature   Continuous        None
8           Nonflavanoid_phenols  Feature   Continuous        None
9                Proanthocyanins  Feature   Continuous        None
10               Color_intensity  Feature   Continuous        None
11                           Hue  Feature   Continuous        None
12  0D280_0D315_of_diluted_wines  Feature   Continuous        None
13                       Proline  Feature      Integer        None

   description units missing_values
0         None  None             no
1         None  None             no
```

```
2          None  None              no
3          None  None              no
4          None  None              no
5          None  None              no
6          None  None              no
7          None  None              no
8          None  None              no
9          None  None              no
10         None  None              no
11         None  None              no
12         None  None              no
13         None  None              no
```

Load the Dataset:

```
1 df = pd.DataFrame(data=X, columns=wine.variables['name'][1:])
2 df['class'] = y
```

Data Inspection:

```
1 print(df.head())
2
3 print(df.describe())
4
5 print(df.info())
```

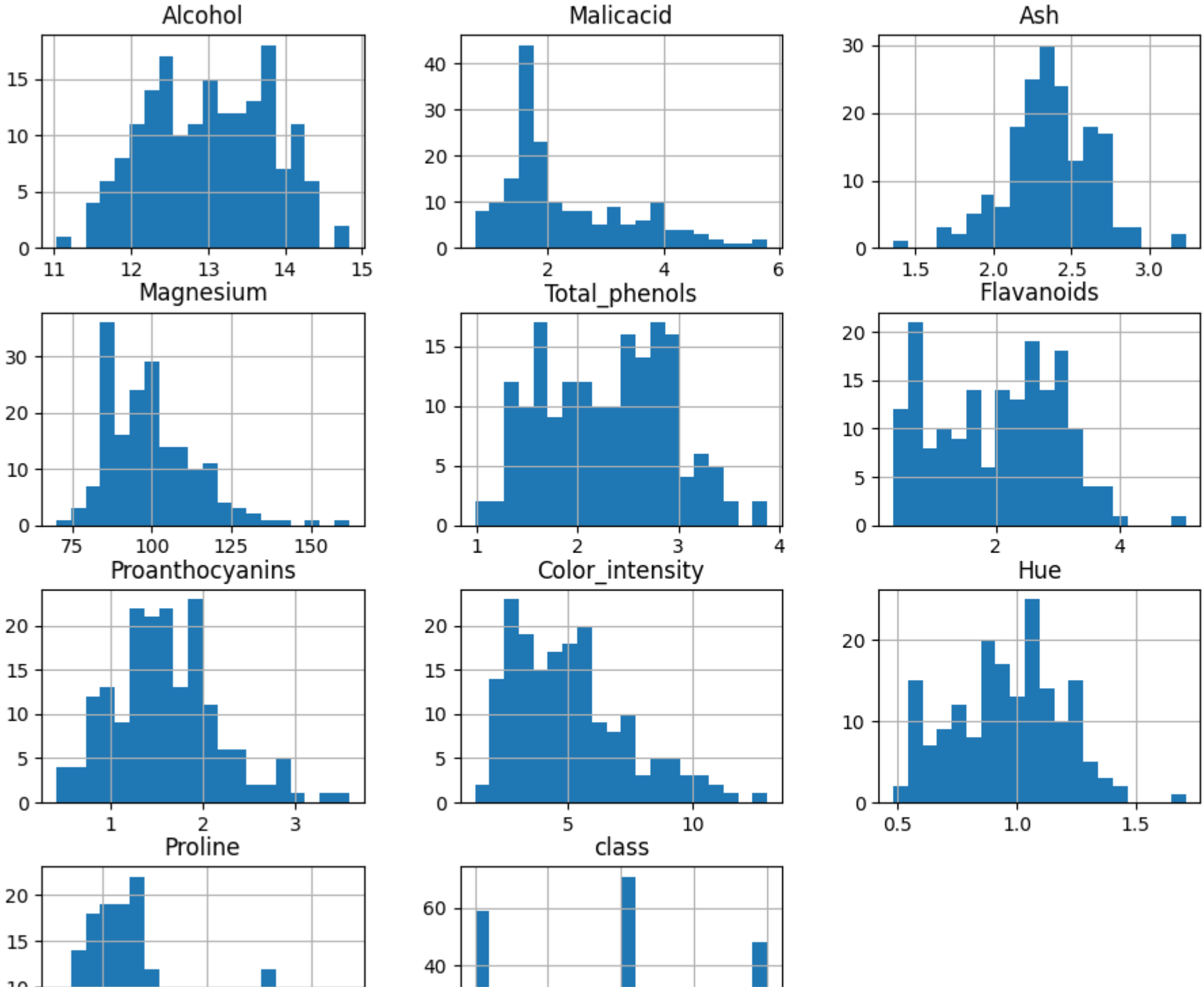
```
mean        2.295112    2.029270              0.361854         1.590899
std         0.625851    0.998859              0.124453         0.572359
min         0.980000    0.340000              0.130000         0.410000
25%         1.742500    1.205000              0.270000         1.250000
50%         2.355000    2.135000              0.340000         1.555000
75%         2.800000    2.875000              0.437500         1.950000
max         3.880000    5.080000              0.660000         3.580000

name   Color_intensity         Hue  0D280_0D315_of_diluted_wines      Proline  \
count       178.000000  178.000000                    178.000000   178.000000
mean          5.058090    0.957449                      2.611685   746.893258
std           2.318286    0.228572                      0.709990   314.907474
min           1.280000    0.480000                      1.270000   278.000000
25%           3.220000    0.782500                      1.937500   500.500000
50%           4.690000    0.965000                      2.780000   673.500000
75%           6.200000    1.120000                      3.170000   985.000000
max          13.000000    1.710000                      4.000000  1680.000000

name        class
count  178.000000
```

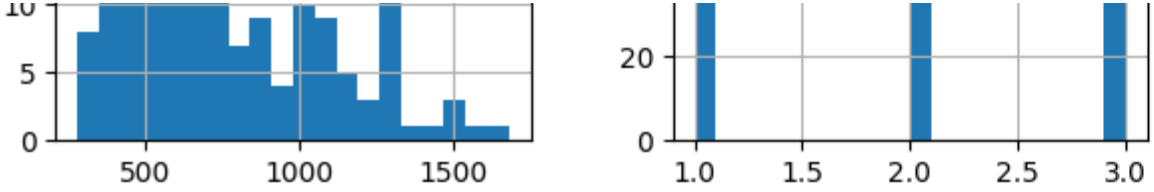
```
 10  Hue                           178 non-null    float64
 11  0D280_0D315_of_diluted_wines  178 non-null    float64
 12  Proline                       178 non-null    int64
 13  class                         178 non-null    int64
dtypes: float64(11), int64(3)
```

Exploratory Data Analysis (EDA):

numerical feature

```
df.hist(bins=20, figsize=(15, 10))
plt.show()
```

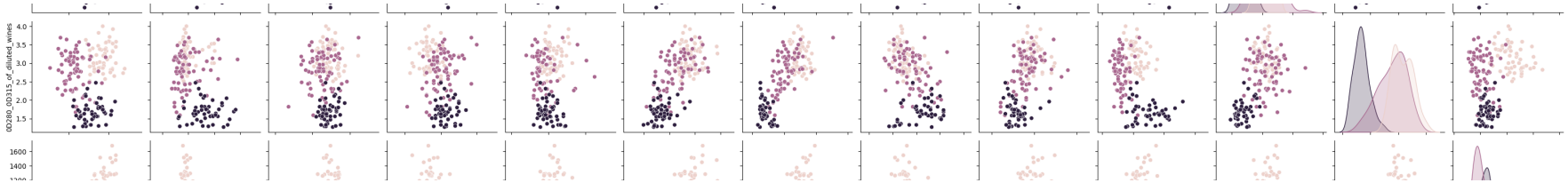

Alcohol
Malicacid
Ash
Magnesium
Total_phenols
Flavanoids
Proanthocyanins
Color_intensity
Hue
Proline
class



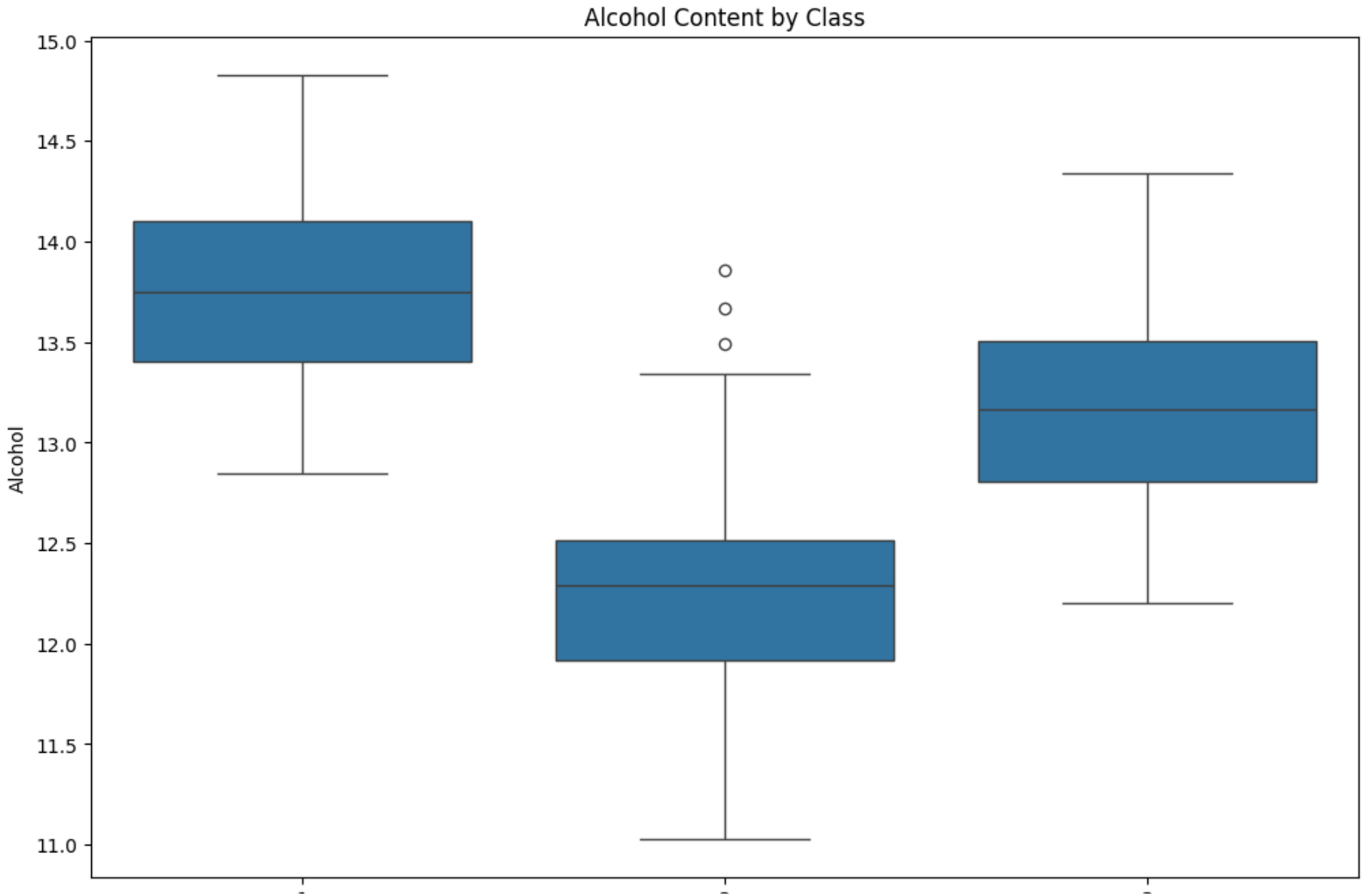
visualize relationships between features

```
1 sns.pairplot(df, hue='class')
2 plt.show()
```


Alcohol
Malicacid
Ash
Alcalinity_of_ash
Magnesium
Total_phenols
Flavanoids
Nonflavanoid_phenols
Proanthocyanins
Color_intensity
Hue
class
1
2
3




```
plt.figure(figsize=(12, 8))
sns.boxplot(x='class', y='Alcohol', data=df)
plt.title('Alcohol Content by Class')
plt.show()
```



Data Pre-processing:

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

```
name
Alcohol                         0
Malicacid                       0
Ash                             0
Alcalinity_of_ash               0
Magnesium                       0
Total_phenols                   0
Flavanoids                      0
Nonflavanoid_phenols            0
Proanthocyanins                 0
Color_intensity                 0
Hue                             0
0D280_0D315_of_diluted_wines    0
Proline                         0
class                           0
dtype: int64
```

```
scaler = StandardScaler()
df_scaled = scaler.fit_transform(df.drop('class', axis=1))
```

```
from sklearn.preprocessing import LabelEncoder

le = LabelEncoder()
```