Logistic Regression Analysis: Wine Datasets

Marquez, Keith Leigh Zhen R.

Data Wrangling:

```
pip install ucimlrepo
```

```
Requirement already satisfied: ucimlrepo in /usr/local/lib/python3.10/dist-packages (0.0.6)
```

```
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://archive.ics.uci.e
                            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
```

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

```
wd = pd.concat([X,y], axis=1)
wd
```

	Alcohol	Malicacid	Ash	Alcalinity_of_ash	Magnesium	Total_phenols	Flavanoids
0	14.23	1.71	2.43	15.6	127	2.80	3.06
1	13.20	1.78	2.14	11.2	100	2.65	2.76
2	13.16	2.36	2.67	18.6	101	2.80	3.24
3	14.37	1.95	2.50	16.8	113	3.85	3.49
4	13.24	2.59	2.87	21.0	118	2.80	2.69
...	...	...	...	...	...	...	...
173	13.71	5.65	2.45	20.5	95	1.68	0.61
174	13.40	3.91	2.48	23.0	102	1.80	0.75
175	13.27	4.28	2.26	20.0	120	1.59	0.69
176	13.17	2.59	2.37	20.0	120	1.65	0.68
177	14.13	4.10	2.74	24.5	96	2.05	0.76

178 rows × 14 columns

Next steps: View recommended plots

```
wd.dtypes
```

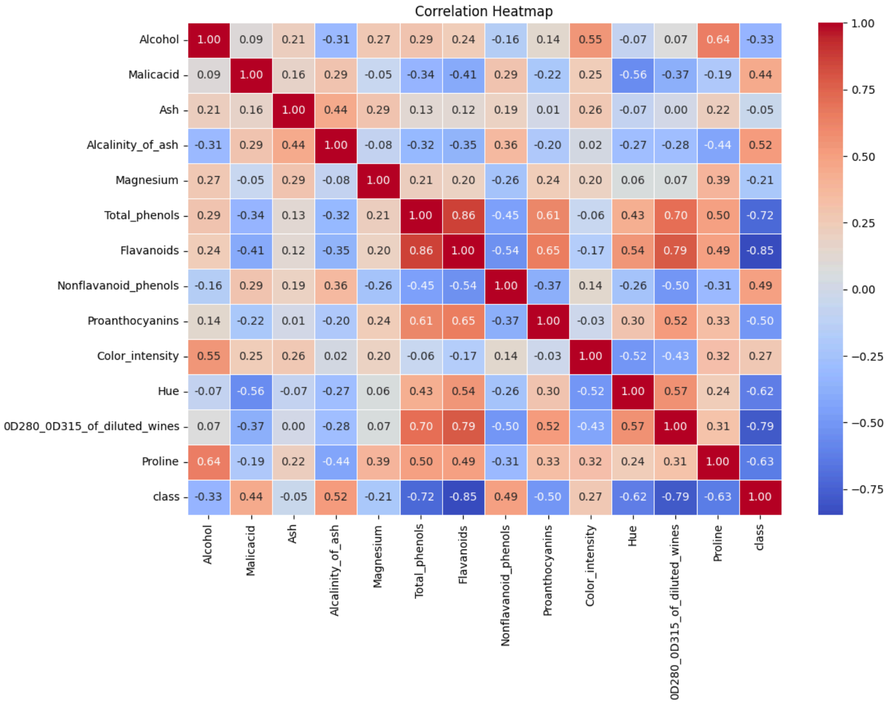
```
Alcohol                         float64
Malicacid                       float64
Ash                             float64
Alcalinity_of_ash               float64
Magnesium                         int64
Total_phenols                   float64
Flavanoids                      float64
Nonflavanoid_phenols            float64
Proanthocyanins                 float64
Color_intensity                 float64
Hue                             float64
0D280_0D315_of_diluted_wines    float64
Proline                           int64
class                             int64
dtype: object
```

```
# Check for duplicates
duplicate_rows = wd.duplicated()

# Count the number of duplicate rows
num_duplicates = duplicate_rows.sum()
print("Number of duplicate rows:", num_duplicates)
```

```
Number of duplicate rows: 0
```

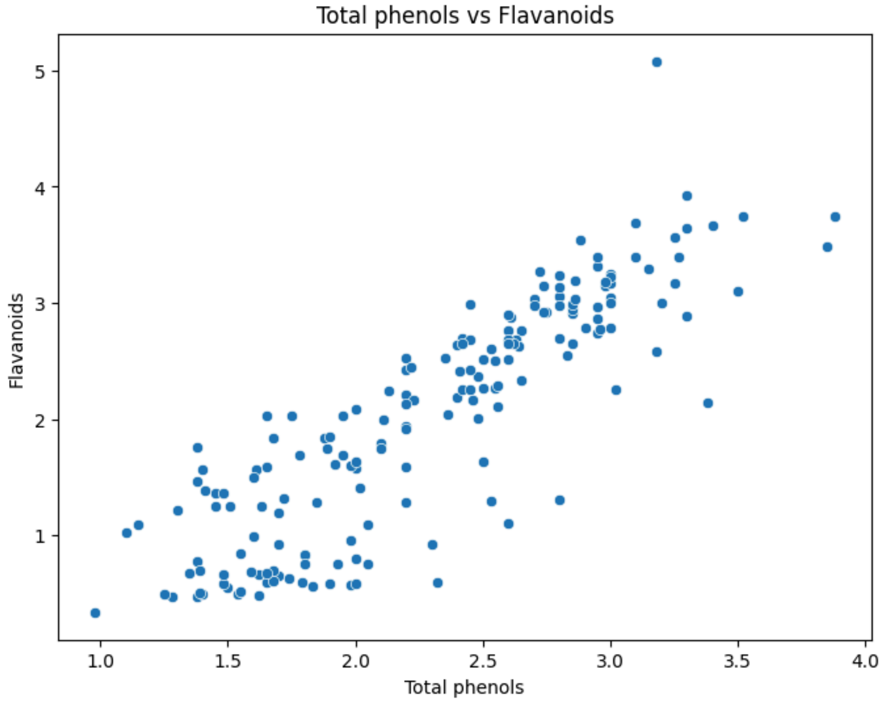
EDA (exploratory data analysis):

```
plt.figure(figsize=(12, 8))
plt.title("Correlation Heatmap")
sns.heatmap(wd.corr(), annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.show()
```



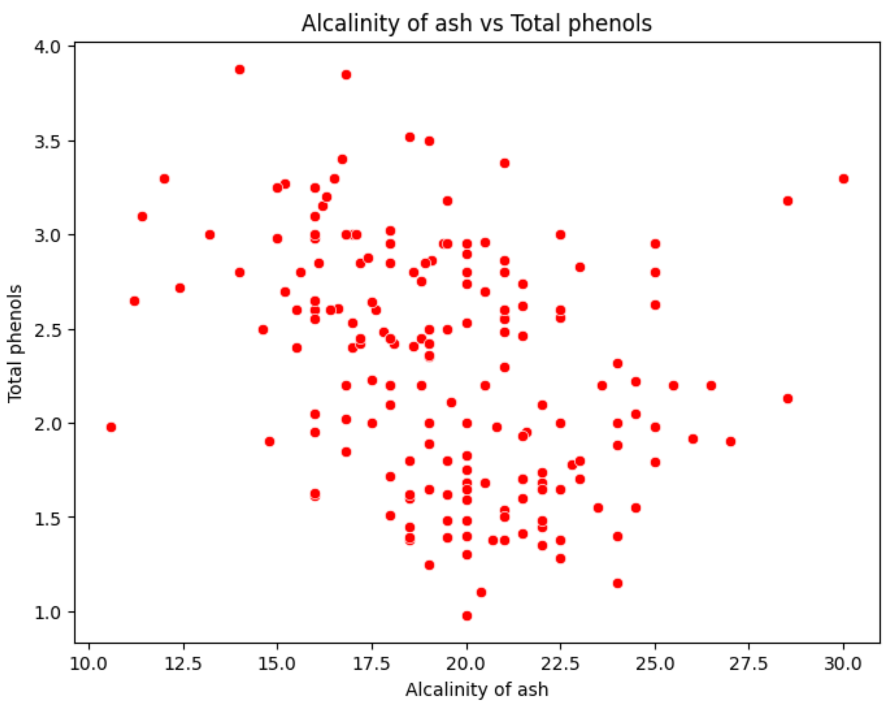
The correlation heatmap reveals relationships between various chemical properties of wine. Key findings include a strong positive correlation between Total phenols and Flavanoids, and between OD280/OD315 and Flavanoids. Conversely, there's a strong negative correlation between Alcalinity of ash and Total phenols. These insights can inform winemakers about factors influencing wine quality and guide production processes. For instance, enhancing total phenols could improve taste, while managing alkalinity of ash can help achieve desired wine profiles. Overall, leveraging such analyses can optimize wine quality by focusing on influential chemical properties.

```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Total_phenols', y='Flavanoids', data=wd)
plt.title('Total phenols vs Flavanoids')
plt.xlabel('Total phenols')
plt.ylabel('Flavanoids')
plt.show()
```



The scatter plot illustrates the relationship between Total Phenols and Flavanoids in a dataset. It indicates a positive correlation, indicating that as Total Phenols increase, so do Flavanoids. Data points are spread out but concentrated at moderate levels of both Phenols and Flavanoids. This suggests that higher Phenols are likely to correspond to higher Flavanoids, possibly indicating a biological or chemical connection.

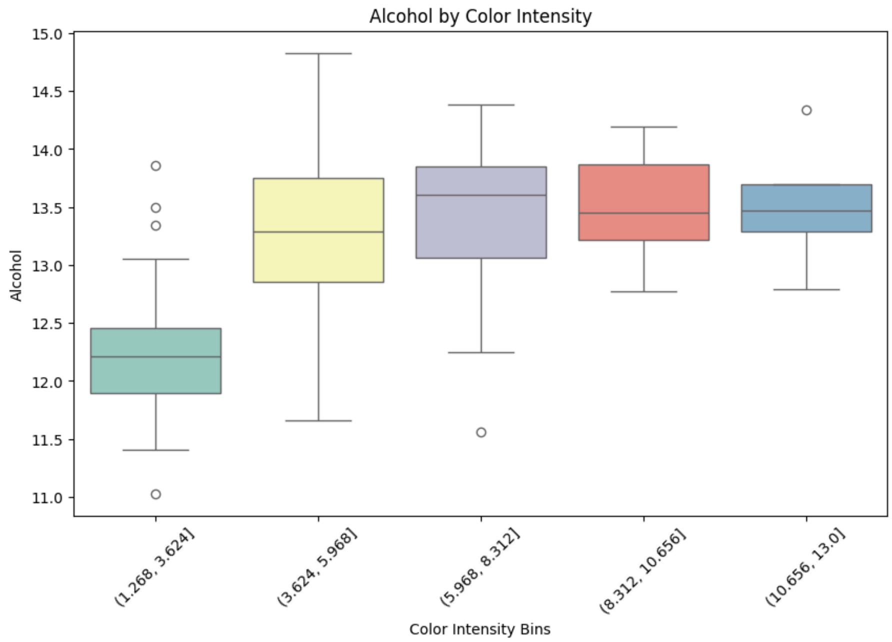
```
plt.figure(figsize=(8, 6))
sns.scatterplot(x='Alcalinity_of_ash', y='Total_phenols', data=wd, color='red')
plt.title('Alcalinity of ash vs Total phenols')
plt.xlabel('Alcalinity of ash')
plt.ylabel('Total phenols')
plt.show()
```



The scatter plot compares Alkalinity of ash (ranging from 10 to 30) on the x-axis with Total phenols (ranging from 1 to 4) on the y-axis. There's no discernible pattern in the data distribution, indicating no clear correlation between the variables. This suggests that the alkalinity of ash may not consistently predict total phenols content.

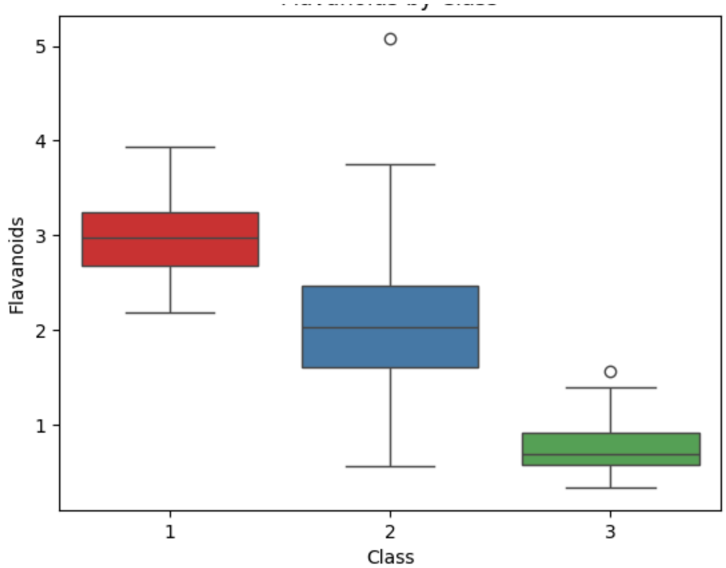
```
plt.figure(figsize=(10, 6))
wd['Color_intensity_bins'] = pd.cut(wd['Color_intensity'], bins=5)
sns.boxplot(x='Color_intensity_bins', y='Alcohol', data=wd, palette='Set3')
plt.title('Alcohol by Color Intensity')
plt.xlabel('Color Intensity Bins')
plt.ylabel('Alcohol')
plt.xticks(rotation=45)
plt.show()
```

```
<ipython-input-10-f60969592d45>:3: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0.

  sns.boxplot(x='Color_intensity_bins', y='Alcohol', data=wd, palette='Set3')
```

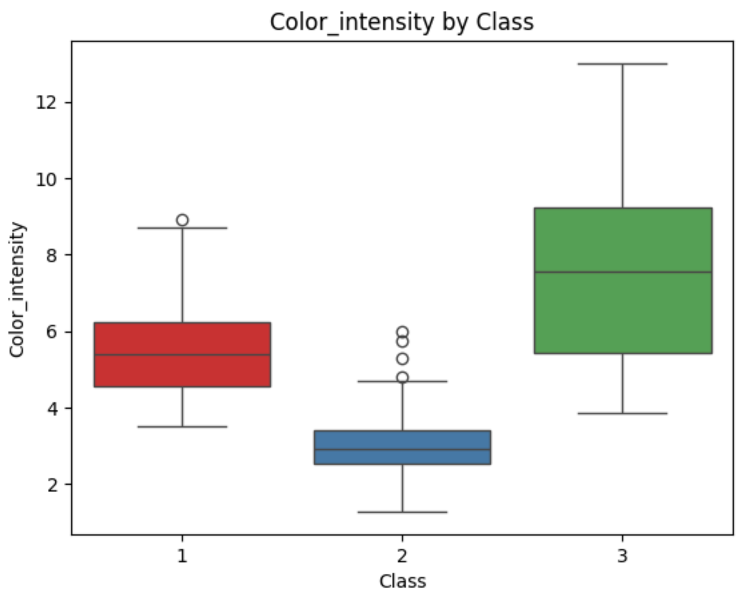


Lower color intensity bins exhibit greater variability in alcohol levels, contrasting with mid-range bins that show consistent distributions. Outliers in higher intensity bins suggest deviations from the general trend, while the highest intensity bin indicates both a higher median alcohol content and significant variability. Overall, while a trend of higher alcohol content with increased color intensity emerges

```
plt.figure(figsize=(12, 8))
features_of_interest = ['Alcohol', 'Total_phenols', 'Flavanoids', 'Color_intensity']
for feature in features_of_interest:
    sns.boxplot(x='class', y=feature, data=wd, palette='Set1')
    plt.title(f'{feature} by Class')
    plt.xlabel('Class')
    plt.ylabel(feature)
    plt.show()
```




```
<ipython-input-11-0c9cd1d7ccc8>:4: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.(

  sns.boxplot(x='class', y=feature, data=wd, palette='Set1')
```



Alcohol by Class: Class 1 (Red) has the highest median alcohol content, indicating consistency. Class 2 (Blue) shows lower median alcohol content with outliers, suggesting variability. Class 3 (Green) has a median alcohol content close to Class 1 but with more variability. Total Phenols by Class: Class 1 generally has higher total phenol levels, with some outliers. Class 2 exhibits more variability in phenol levels. Class 3 has the lowest median phenol levels but also shows outliers. Flavonoids by Class: Class 1 contains significantly higher levels of flavonoids. Class 2 has moderate levels but greater variability and an outlier. Class 3 consistently shows low levels of flavonoids. Color Intensity by Class: Each class has a unique distribution of color intensity. Class 2 shows significant variability, especially with outliers. Class 3 has the highest median intensity.

Logistic Regression Analysis

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
```

Split the data into training and testing sets

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```