#necessary libraries

import numpy as np

import pandas as pd

from sklearn.preprocessing import StandardScaler

wine\_file = pd.read\_csv('/kaggle/input/wine-quality/Wine Quality.csv') #loading the dataset

print(wine\_file.shape) #checking the shape(row x col) of the dataset

print(wine\_file.columns) #checking the features

# Removing Null values:

wine\_file.isnull().sum() #checking the total number of null values in features.

#It is observed that the number of null values is relatively small; therefore, removing the rows containing null values is preferable to removing columns. As a result, the .dropna() method should be applied.

print("Before removing null values:", wine\_file.shape)

wine\_file = wine\_file.dropna(axis = 0, subset = ['fixed acidity']) #removing null rows for the 'fixed acidity' feature.

wine\_file = wine\_file.dropna(axis = 0, subset = ['volatile acidity']) #removing null rows for the 'volatile acidity' feature.

wine\_file = wine\_file.dropna(axis = 0, subset = ['citric acid']) #removing null rows for the 'critic acid' feature.

wine\_file = wine\_file.dropna(axis = 0, subset = ['residual sugar']) #removing null rows for the 'residual sugar' feature.

wine\_file = wine\_file.dropna(axis = 0, subset = ['chlorides']) #removing null rows for the 'chlorides' feature.

wine\_file = wine\_file.dropna(axis = 0, subset = ['pH']) #removing null rows for the 'pH' feature.

wine\_file = wine\_file.dropna(axis = 0, subset = ['sulphates']) #removing null rows for the 'sulphates' feature.

print("After removing null values:", wine\_file.shape)

wine\_file.isnull().sum()

# Removing Duplicates

# Duplicate values can distort the representation of data. These duplicates may skew the data, leading to inaccurate or misleading results in the model

dup = wine\_file.duplicated() # identifying duplicate rows

sum\_dup = dup.sum() #total number of duplicate rows

print(f"Number of duplicate values present are:",sum\_dup)

wine\_file2 = wine\_file.drop\_duplicates() #removes duplicate rows

print('Duplicates removed!')

dup2 = wine\_file2.duplicated()

sum\_dup2 = dup2.sum()

print(f"Number of duplicate values present are:",sum\_dup2)

#Handling Categorical Data

wine\_file2['type'].unique() # retrieves the unique values from the 'type' column

# As the retrieval values are Nominal, instead of scaling, encoding methods need to be applied. This is why, applying a one-hot-encoding method.

wine\_file2 = pd.get\_dummies(wine\_file2, columns=['type'], drop\_first=True) #creating dummy variables for categorical data

wine\_file2.head()

wine\_file2['type\_white'].unique() #Instead of having non-ordered 'type' values, the values are now represented with conditions or states.

#Feature Scaling

# Seperating the target column from the features as scaling brings all the input features to a common scale so that feature contributes equally to the learning. The target should not be affected by this as it is what the model tries to predict.

target\_val = wine\_file2['quality']

vals = wine\_file2.drop(columns=['quality'])

print(vals.describe())

# Observing the dataset, it is vigilant that residual sugar, total sulfur dioxide, chlorides columns contain a significant variance and different scale units. So, scaling these features may help to understand these features more.

sel\_cols = ['residual sugar','chlorides', 'total sulfur dioxide', 'free sulfur dioxide']

scaler = StandardScaler() #as the dataset contains features of different scale with high variance.

vals[sel\_cols] = scaler.fit\_transform(vals[sel\_cols])

print(vals.describe())

wine\_file3 = pd.concat([vals, target\_val], axis=1)

print(wine\_file3.head())

wine\_file3.shape

# Correlation

# Correlation helps to understand how much a certain feature is similar to other features. If two features display high correlation, they affect the model similarly. So, one of it can be removed while considering it redundant.

corr\_matx = wine\_file3.corr() #applying correlation

print(corr\_matx)

import seaborn as sns

sns.heatmap(corr\_matx, cmap = 'YlGnBu')

# Dropping one of the highly correlated values based on the generated heatmap

data\_dropped = wine\_file3.drop('free sulfur dioxide', axis=1)

# Checking the dataset after removing correlated columns

print(data\_dropped.head())