TRACK 1 ASSIGNMENT

Wine quality prediction model

- By Aditya verma
- DATE 2/6/2025

DATASET USED

Wine quality.csv

https://drive.google.com/file/d/1C9wDZcosuFLvc54gtKia_yQiAdsaK-SA/view?usp=classroom_web&authuser=0

Aim

To make the model to test the quality of wine

Wine quality prediction model

imports

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

0] ✓ 0.0s

dataset reading

```
df = pd.read_csv('winequality-red - winequality-red.csv')
```

1] ✓ 0.0s Python

dataset information

```
df.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
 11  quality               1599 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

Add Markdown Cell

Checking the columns

```
df.head()
```

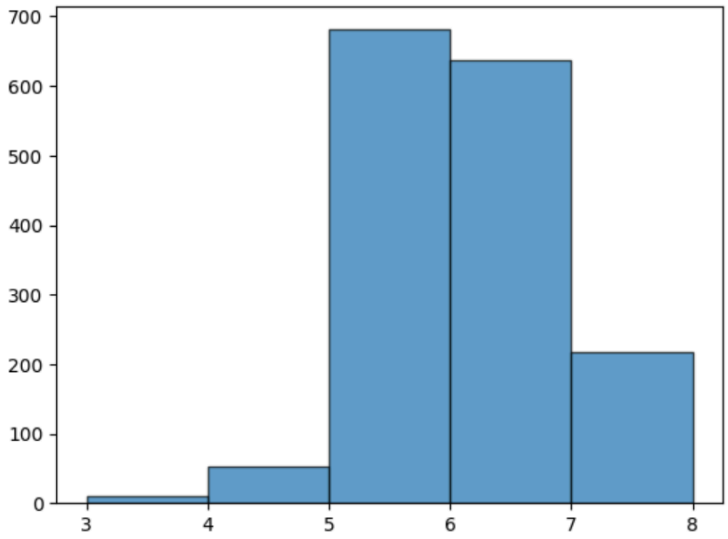
✓ 0.0s Python

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
0	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5
1	7.8	0.88	0.00	2.6	0.098	25.0	67.0	0.9968	3.20	0.68	9.8	5
2	7.8	0.76	0.04	2.3	0.092	15.0	54.0	0.9970	3.26	0.65	9.8	5
3	11.2	0.28	0.56	1.9	0.075	17.0	60.0	0.9980	3.16	0.58	9.8	6
4	7.4	0.70	0.00	1.9	0.076	11.0	34.0	0.9978	3.51	0.56	9.4	5

```
plt.hist(df['quality'], bins=5, edgecolor='black', alpha=0.7)
```

✓ 0.2s Pyth

```
(array([ 10.,  53., 681., 638., 217.]),
 array([3., 4., 5., 6., 7., 8.]),
 <BarContainer object of 5 artists>)
```



Check for the null values

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

✓ 0.0s

```
fixed acidity         0
volatile acidity      0
citric acid           0
residual sugar        0
chlorides             0
free sulfur dioxide   0
total sulfur dioxide  0
density               0
pH                    0
sulphates             0
alcohol               0
quality               0
dtype: int64
```

Conversion into datframe

```
df=pd.DataFrame(df)
```

0.0s Python

Splitting the test set and train set

```
import sklearn as sk
from sklearn.model_selection import train_test_split
X = df.drop('quality', axis=1)
y = df['quality']
X.info()

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

0.0s P

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1599 entries, 0 to 1598
Data columns (total 11 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1599 non-null   float64
 1   volatile acidity      1599 non-null   float64
 2   citric acid           1599 non-null   float64
 3   residual sugar        1599 non-null   float64
 4   chlorides             1599 non-null   float64
 5   free sulfur dioxide   1599 non-null   float64
 6   total sulfur dioxide  1599 non-null   float64
 7   density               1599 non-null   float64
 8   pH                    1599 non-null   float64
 9   sulphates             1599 non-null   float64
 10  alcohol               1599 non-null   float64
dtypes: float64(11)
```

Applying the linear regression on the model

```
from sklearn.linear_model import LinearRegression
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
from sklearn.metrics import  r2_score
r2 = r2_score(y_test, y_pred)
```

0.0s Python

```
print(f'accuracy in linear regerssion: {r2:.2f}')
```

0.0s Python

```
accuracy in linear regerssion: 0.40
```

using standard scalar

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
model_scaled = LinearRegression()
model_scaled.fit(X_train_scaled, y_train)
y_pred_scaled = model_scaled.predict(X_test_scaled)
r2_scaled = r2_score(y_test, y_pred_scaled)
print(f'accuracy in standard scalar: {r2_scaled:.2f}')
```

✓ 0.0s

```
accuracy in standard scalar: 0.40
```

+ Code + Markdown

lets convert the regression model into classification model

```
y_train = [1 if x >= 6 else 0 for x in y_train]
y_test = [1 if x >= 6 else 0 for x in y_test]
```

✓ 0.0s Python

```
y_train
```

✓ 0.0s Python

```
[1,
 1,
 1,
 0,
 0,
 0,
 1,
 1,
 1
```

Applying the Random forest classifier

```
from sklearn.ensemble import RandomForestClassifier
rf_model = RandomForestClassifier( n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)
y_pred_rf = rf_model.predict(X_test)
from sklearn.metrics import accuracy_score
rf_accuracy = accuracy_score(y_test, y_pred_rf)

print(f'Accuracy of Random Forest Classifier: {rf_accuracy:.2f}')
```

✓ 0.7s Python

```
Accuracy of Random Forest Classifier: 0.79
```

Applying the logistic regression

```
from sklearn.linear_model import LogisticRegression
logistic_model = LogisticRegression(max_iter=1000)
logistic_model.fit(X_train, y_train)
y_pred_logistic = logistic_model.predict(X_test)
logistic_accuracy = accuracy_score(y_test, y_pred_logistic)
print(f'Accuracy of Logistic Regression: {logistic_accuracy:.2f}')
```

0.2s

Accuracy of Logistic Regression: 0.74

Applying the gradientboosting classifier

```
from sklearn.ensemble import GradientBoostingClassifier
gb_model = GradientBoostingClassifier(n_estimators=100, random_state=42)
gb_model.fit(X_train, y_train)
y_pred_gb = gb_model.predict(X_test)
gb_accuracy = accuracy_score(y_test, y_pred_gb)
print(f'Accuracy of Gradient Boosting Classifier: {gb_accuracy:.2f}')
```

0.9s

Accuracy of Gradient Boosting Classifier: 0.76

applying svm model

```
from sklearn import svm
svm_model = svm.SVC(kernel='linear', random_state=42)
svm_model.fit(X_train, y_train)
y_pred_svm = svm_model.predict(X_test)
svm_accuracy = accuracy_score(y_test, y_pred_svm)
print(f'Accuracy of SVM Classifier: {svm_accuracy:.2f}')
```

1.4s

Accuracy of SVM Classifier: 0.73

CONCLUSION

The accuracy in the various algorithms are

1. Regression :-

   Linear regression - 0.40

   Standard scalar - 0.40

2. Classification :-

   Random forest classifier - 0.79

   Logistic regression - 0.74

   Gradient boosting classifier - 0.76

   Svm model - 0.73

Link of the notebook :

task2.ipynb

Or

https://colab.research.google.com/drive/1r81tYMih-w8hNJAsCCaZNLFB8qoltKqK?usp=sharing