Wine Quality Prediction

- LISHA GOEL -

Contents

Introduction

We will predict the quality of wine on the basis of the given features. We use the wine quality dataset available on Internet for free. This dataset has the fundamental features which are responsible for affecting the quality of the wine. By using of several Machine learning models, we will predict the quality of the wine.

Libraries

PANDAS

Useful Library for Data Handling.

NUMPY

Library used for working with arrays.

SKLEARN

This module contains multiple libraries having pre-implemented functions to perform tasks from data preprocessing to model development and evaluation.

XGBOOST

This contains the eXtreme Gradient Boosting machine learning algorithm which is one of the algorithms which helps us to achieve high accuracy on predictions.

Output Screenshots

Read csv

Printing first five rows of dataset

```
[24]: df = pd.read_csv('winequality.csv')
      print(df.head())
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0            7.4              0.70         0.00             1.9      0.076
1            7.8              0.88         0.00             2.6      0.098
2            7.8              0.76         0.04             2.3      0.092
3           11.2              0.28         0.56             1.9      0.075
4            7.4              0.70         0.00             1.9      0.076

   free sulfur dioxide  total sulfur dioxide  density    pH  sulphates  \
0                   11                    34   0.9978  3.51       0.56
1                   25                    67   0.9968  3.20       0.68
2                   15                    54   0.9970  3.26       0.65
3                   17                    60   0.9980  3.16       0.58
4                   11                    34   0.9978  3.51       0.56

   alcohol  quality
0      9.4        5
1      9.8        5
2      9.8        5
3      9.8        6
4      9.4        5
```

Info

Type of data present in each of the columns present in the dataset

```
[25]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 79 entries, 0 to 78
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   fixed acidity         79 non-null     float64
 1   volatile acidity      79 non-null     float64
 2   citric acid           79 non-null     float64
 3   residual sugar        79 non-null     float64
 4   chlorides             79 non-null     float64
 5   free sulfur dioxide   79 non-null     int64
 6   total sulfur dioxide  79 non-null     int64
 7   density               79 non-null     float64
 8   pH                    79 non-null     float64
 9   sulphates             79 non-null     float64
 10  alcohol               79 non-null     float64
 11  quality               79 non-null     int64
dtypes: float64(9), int64(3)
memory usage: 7.5 KB
```

Describe

The descriptive statistical measures of the dataset.

```
In [4]: df.describe().T
```

Out[4]:

	count	mean	std	min	25%	50%	75%	max
fixed acidity	79.0	7.635443	1.042930	4.6000	7.1500	7.7000	8.1000	11.2000
volatile acidity	79.0	0.559557	0.164965	0.2200	0.4150	0.5900	0.6700	1.1300
citric acid	79.0	0.208354	0.181669	0.0000	0.0500	0.1600	0.3100	0.6400
residual sugar	79.0	2.588608	1.497326	1.2000	1.8000	2.1000	2.5000	10.7000
chlorides	79.0	0.096975	0.054618	0.0500	0.0740	0.0840	0.0935	0.3680
free sulfur dioxide	79.0	15.455696	10.435145	3.0000	9.0000	13.0000	17.0000	52.0000
total sulfur dioxide	79.0	52.860759	33.994620	10.0000	25.0000	45.0000	72.0000	148.0000
density	79.0	0.996833	0.001083	0.9934	0.9962	0.9968	0.9976	0.9993
pH	79.0	3.340000	0.135751	3.0400	3.2600	3.3400	3.4150	3.9000
sulphates	79.0	0.656962	0.196066	0.3900	0.5400	0.5800	0.7300	1.5600
alcohol	79.0	9.782278	0.638460	9.0000	9.4000	9.5000	10.1000	13.1000
quality	79.0	5.278481	0.678171	4.0000	5.0000	5.0000	6.0000	7.0000

Exploratory Data Analysis

Is null sum

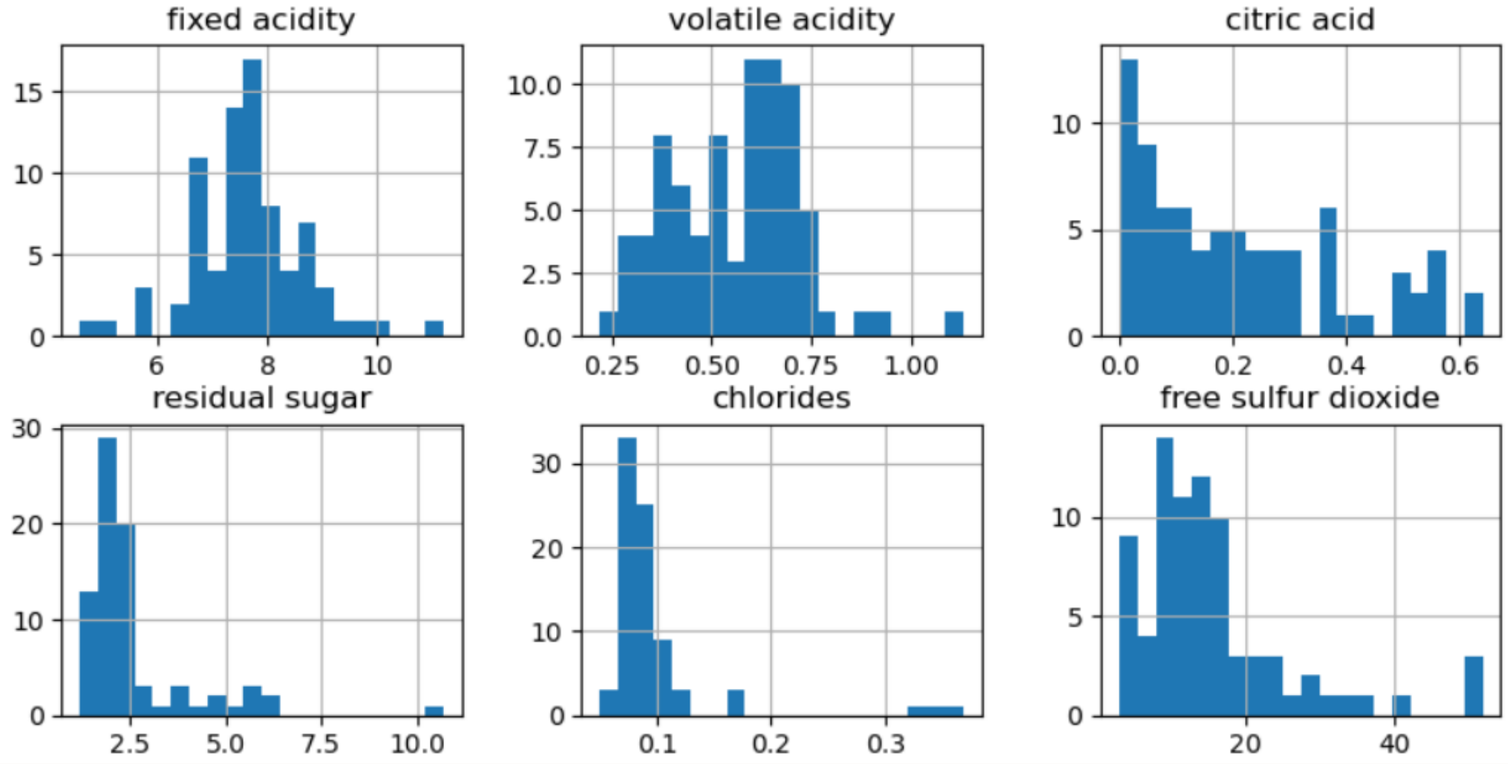
The number of null values in the dataset columns wise.

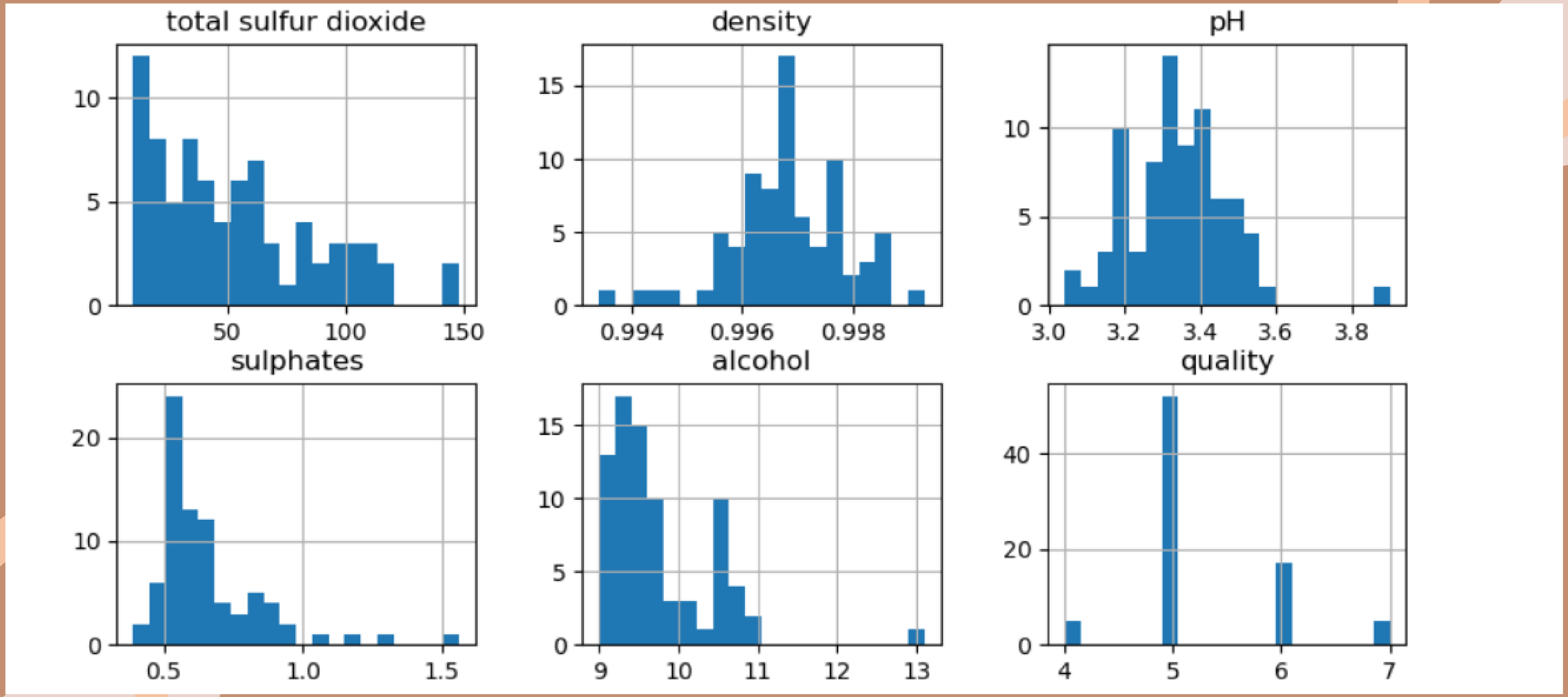
```
In [5]: df.isnull().sum()

Out[5]: 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
```

Histogram

Visualize the distribution of the data with continuous values in the columns of the dataset.

```
In [8]: df.hist(bins=20, figsize=(10, 10))
        plt.show()
```

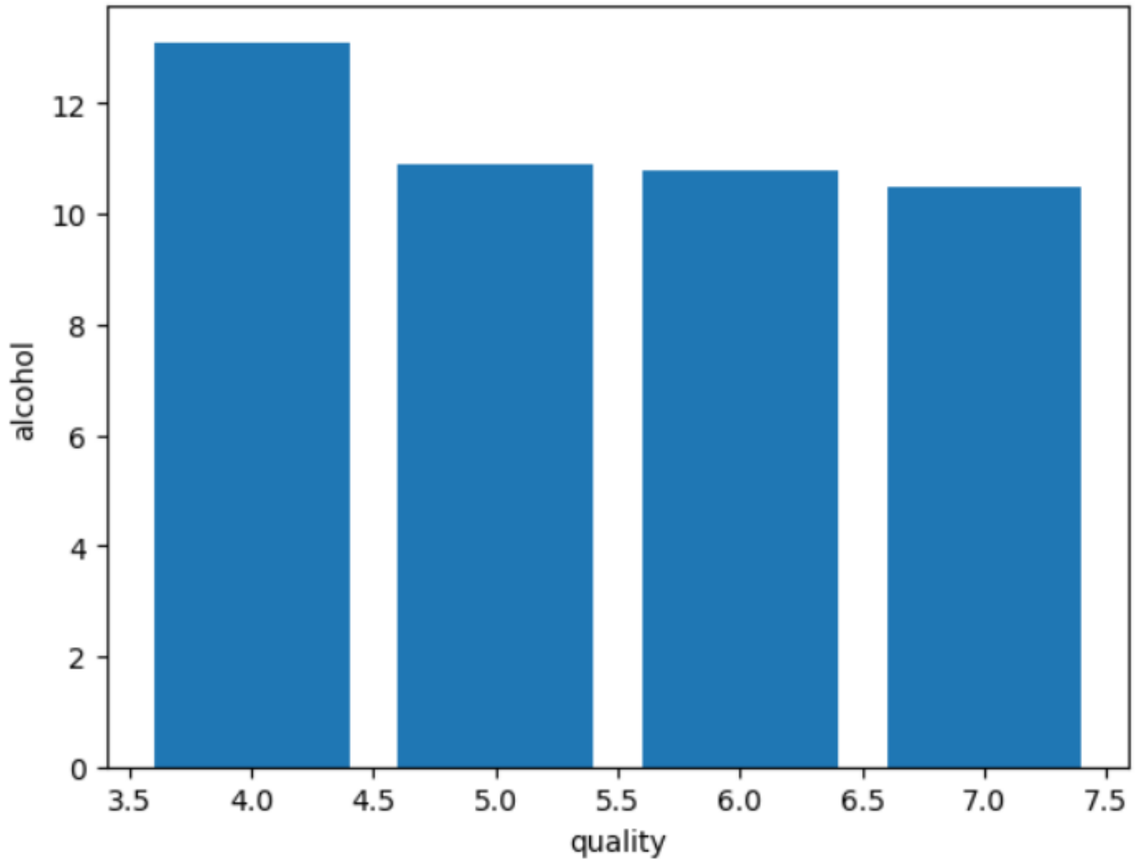



total sulfur dioxide
density
pH
sulphates
alcohol
quality

Count Plot

Visualize the number data for each quality of wine.

```
In [9]: plt.bar(df['quality'], df['alcohol'])
        plt.xlabel('quality')
        plt.ylabel('alcohol')
        plt.show()
```



models

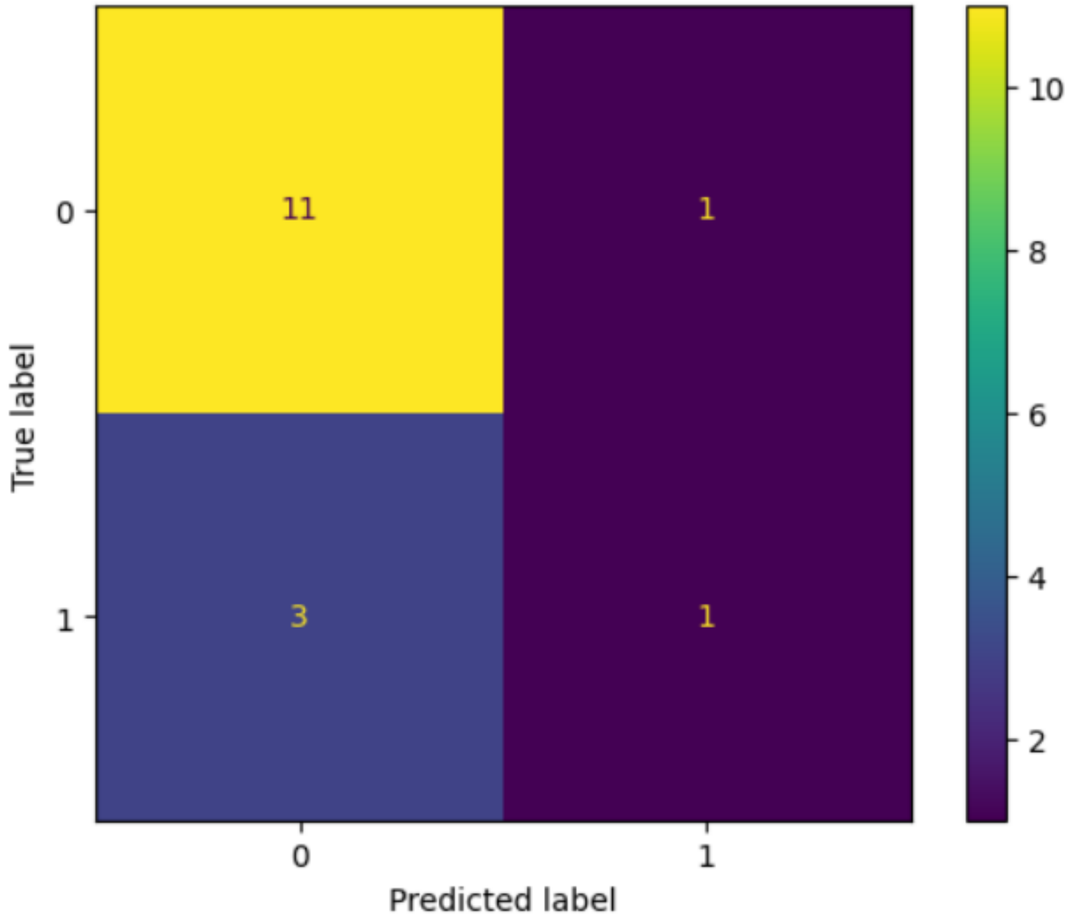
Some basic output

In [16]:
```
models = [LogisticRegression(), XGBClassifier(), SVC(kernel='rbf')]

for i in range(3):
    models[i].fit(xtrain, ytrain)

    print(f'{models[i]} : ')
    print('Training Accuracy : ', metrics.roc_auc_score(ytrain, models[i].predict(xtrain)))
    print('Validation Accuracy : ', metrics.roc_auc_score(
        ytest, models[i].predict(xtest)))
    print()
```

```
LogisticRegression() :
Training Accuracy :  0.5
Validation Accuracy :  0.5

XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, device=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=None, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              multi_strategy=None, n_estimators=None, n_jobs=None,
              num_parallel_tree=None, random_state=None, ...) :
Training Accuracy :  1.0
Validation Accuracy :  0.5833333333333333

SVC() :
Training Accuracy :  0.5277777777777778
Validation Accuracy :  0.5
```

Metrices Plot

In [17]:
```
metrics.plot_confusion_matrix(models[1], xtest, ytest)
plt.show()
```



Thank
you