In [1]:
```
import numpy as np
```

In [2]:
```
import pandas as pd
```

In [3]:
```
from matplotlib import pyplot as plt
```

In [4]:
```
import seaborn as sb
```

In [5]:
```
from sklearn.model_selection import train_test_split
```

In [6]:
```
from sklearn.preprocessing import MinMaxScaler
```

In [8]:
```
from sklearn import metrics
```

In [9]:
```
from sklearn.svm import SVC
```

In [10]:
```
from xgboost import XGBClassifier
```

In [11]:
```
from sklearn.linear_model import LogisticRegression
```

In [12]:
```
import warnings
warnings.filterwarnings('ignore')
```

In [13]:
```
df=pd.read_csv('WineQT.csv')
```

In [15]:
```
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                 11.0                  34.0   0.9978  3.51       0.56   
1                 25.0                  67.0   0.9968  3.20       0.68   
2                 15.0                  54.0   0.9970  3.26       0.65   
3                 17.0                  60.0   0.9980  3.16       0.58   
4                 11.0                  34.0   0.9978  3.51       0.56   

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

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1143 entries, 0 to 1142
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         1143 non-null   float64
 1   volatile acidity      1143 non-null   float64
 2   citric acid           1143 non-null   float64
 3   residual sugar        1143 non-null   float64
 4   chlorides             1143 non-null   float64
 5   free sulfur dioxide   1143 non-null   float64
 6   total sulfur dioxide  1143 non-null   float64
 7   density               1143 non-null   float64
 8   pH                    1143 non-null   float64
 9   sulphates             1143 non-null   float64
 10  alcohol               1143 non-null   float64
 11  quality               1143 non-null   int64  
 12  Id                    1143 non-null   int64  
dtypes: float64(11), int64(2)
memory usage: 116.2 KB
```

In [18]:
```
df.shape
```

Out[18]:
```
(1143, 13)
```

In [19]:
```
df.describe()
```

Out[19]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide
count	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000	1143.000000
mean	8.311111	0.531339	0.268364	2.532152	0.086933	15.615486	45.914698
std	1.747595	0.179633	0.196686	1.355917	0.047267	10.250486	32.782130
min	4.600000	0.120000	0.000000	0.900000	0.012000	1.000000	6.000000
25%	7.100000	0.392500	0.090000	1.900000	0.070000	7.000000	21.000000
50%	7.900000	0.520000	0.250000	2.200000	0.079000	13.000000	37.000000
75%	9.100000	0.640000	0.420000	2.600000	0.090000	21.000000	61.000000
max	15.900000	1.580000	1.000000	15.500000	0.611000	68.000000	289.000000

In [20]:
```
df.isna().any
```

Out[20]:

```
<bound method NDFrame._add_numeric_operations.<locals>.any of       fixed acidity  volatile acidity  citric acid  residual sugar  chlorides  \
0             False             False        False           False      False
1             False             False        False           False      False
2             False             False        False           False      False
3             False             False        False           False      False
4             False             False        False           False      False
...             ...               ...          ...             ...        ...
1138          False             False        False           False      False
1139          False             False        False           False      False
1140          False             False        False           False      False
1141          False             False        False           False      False
1142          False             False        False           False      False

      free sulfur dioxide  total sulfur dioxide  density     pH  sulphates  \
0                   False                 False    False  False      False
1                   False                 False    False  False      False
2                   False                 False    False  False      False
3                   False                 False    False  False      False
4                   False                 False    False  False      False
...                   ...                   ...      ...    ...        ...
1138                False                 False    False  False      False
1139                False                 False    False  False      False
1140                False                 False    False  False      False
1141                False                 False    False  False      False
1142                False                 False    False  False      False

      alcohol  quality     Id
0       False    False  False
1       False    False  False
2       False    False  False
3       False    False  False
4       False    False  False
...       ...      ...    ...
1138    False    False  False
1139    False    False  False
1140    False    False  False
1141    False    False  False
1142    False    False  False

[1143 rows x 13 columns]>
```

In [21]:

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

Out[21]:

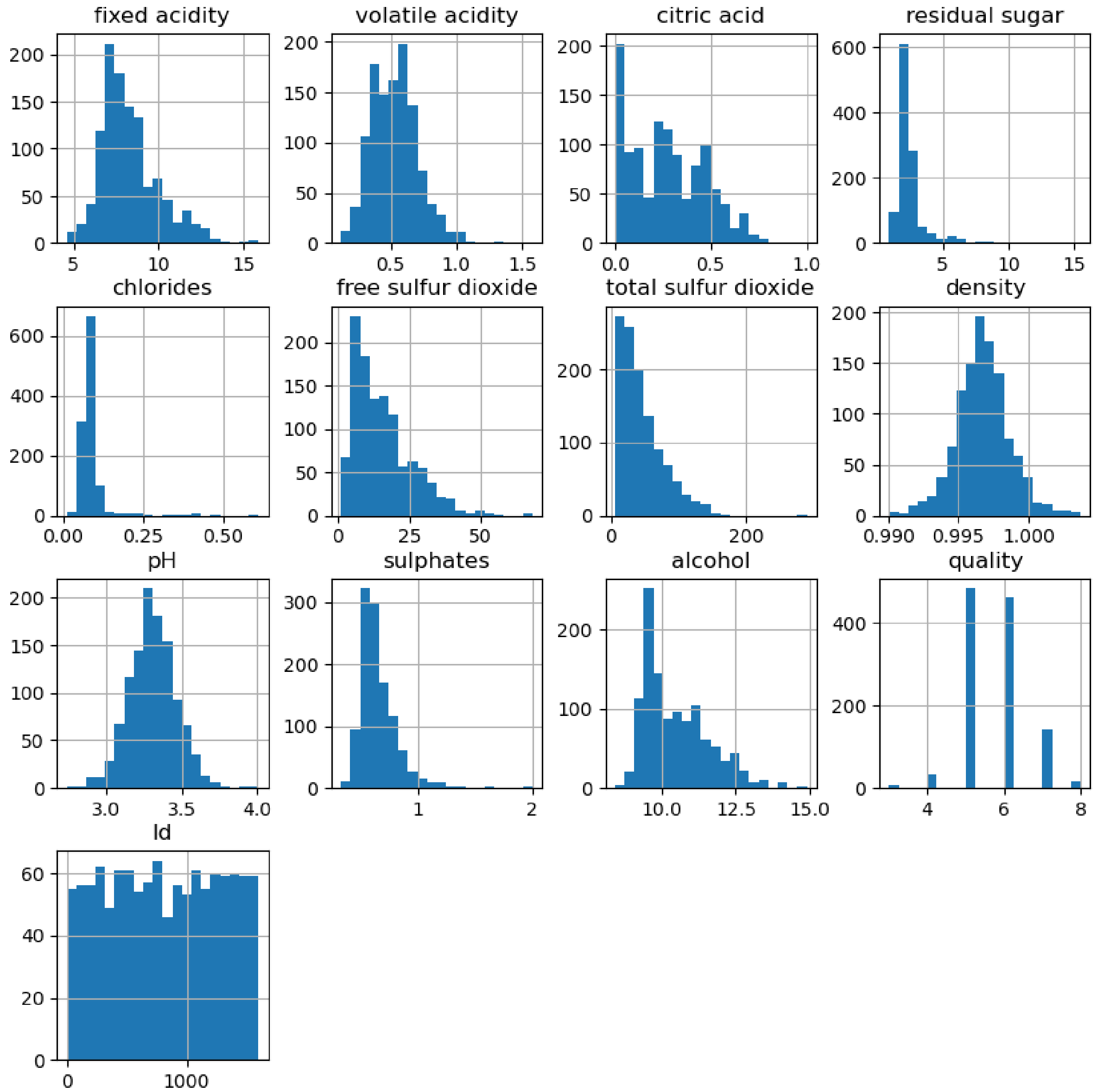
```
fixed acidity           0
volatile acidity        0
citric acid             0
residual sugar          0
chlorides               0
free sulfur dioxide     0
total sulfur dioxide    0
density                 0
pH                      0
sulphates               0
alcohol                 0
quality   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

In [22]:

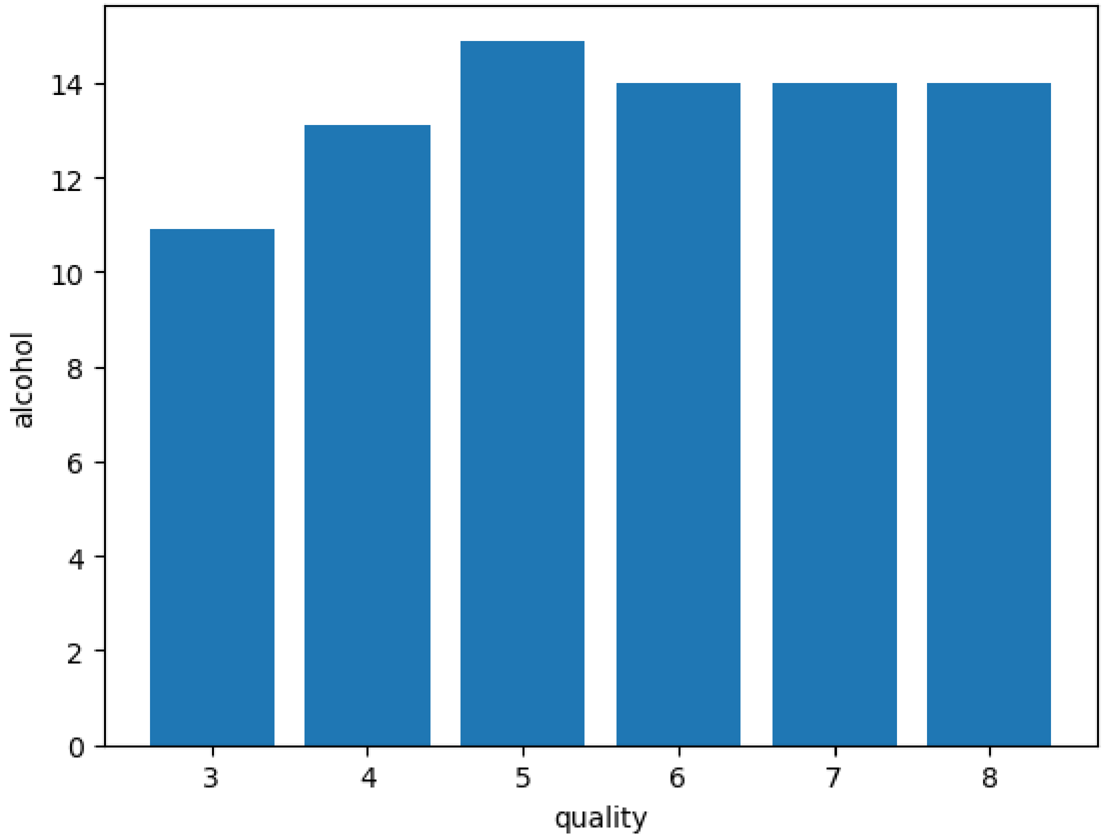
```
df.columns
```

Out[22]:
```
Index(['fixed acidity', 'volatile acidity', 'citric acid', 'residual sugar',
       'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density',
       'pH', 'sulphates', 'alcohol', 'quality', 'Id'],
      dtype='object')
```

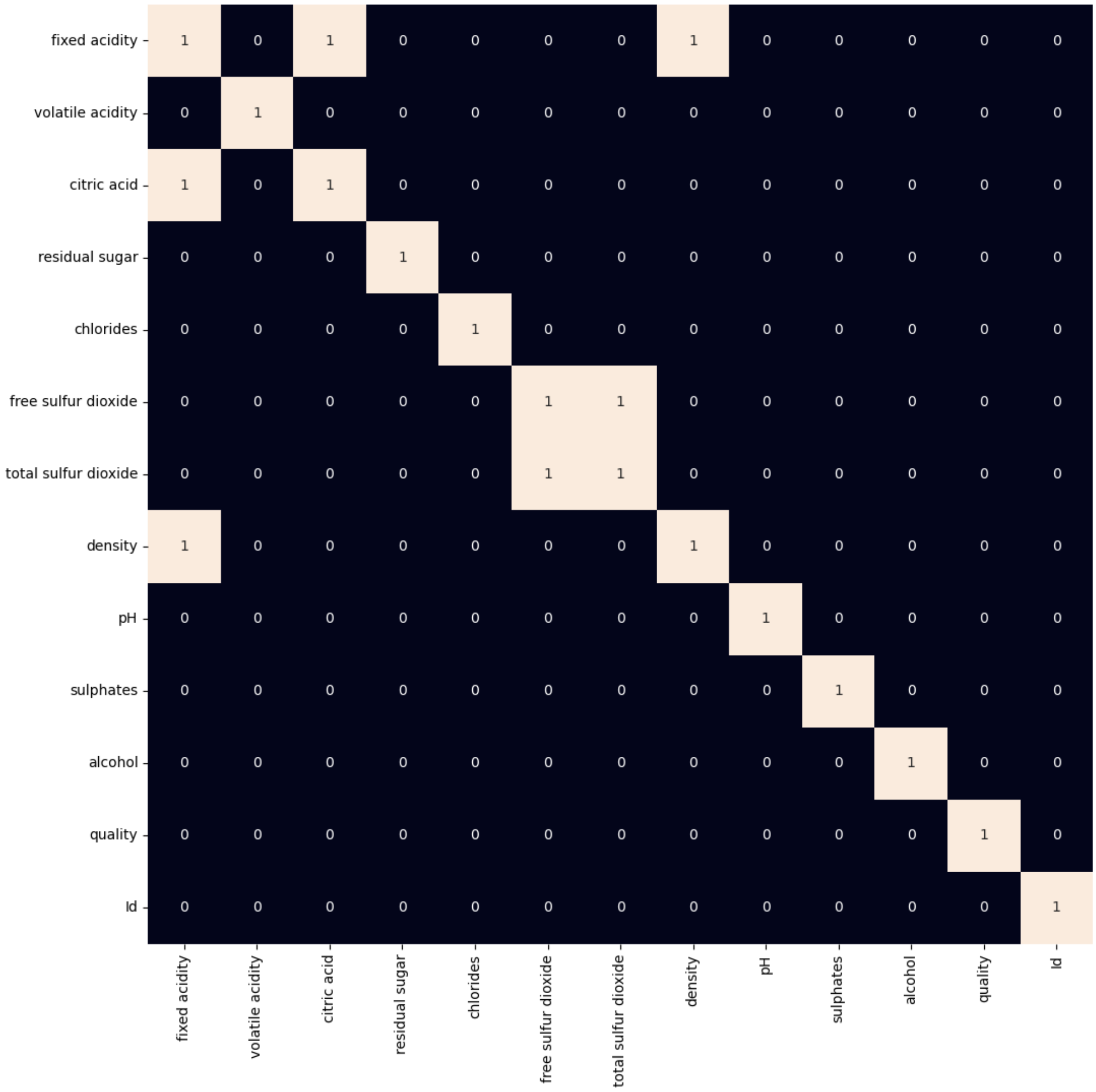
In [23]:
```python
df.hist(bins=20,figsize=(10,10))
plt.show()
```



In [24]:
```python
plt.bar(df['quality'],df['alcohol'])
plt.xlabel('quality')
plt.ylabel('alcohol')
plt.show()
```



In [26]:
```
plt.figure(figsize=(12,12))
sb.heatmap(df.corr()>0.6,annot=True,cbar=False)
plt.show()
```



In [ ]:
```
#From the above heat map we can conclude that the 'total sulphur dioxide' and 'free
```

In [28]:
```
df=df.drop('total sulfur dioxide',axis=1)
```

In [29]:
```
#model development
```

In [30]:
```
df['best quality'] = [1 if x > 5 else 0 for x in df.quality]
```

In [31]:
```
df.replace({'white': 1, 'red': 0}, inplace=True)
```

In [32]:
```
features = df.drop(['quality', 'best quality'], axis=1)
target = df['best quality']

xtrain, xtest, ytrain, ytest = train_test_split(
    features, target, test_size=0.2, random_state=40)

xtrain.shape, xtest.shape
```

Out[32]:
```
((914, 11), (229, 11))
```

In [33]:
```
norm = MinMaxScaler()
xtrain = norm.fit_transform(xtrain)
xtest = norm.transform(xtest)
```

In [34]:
```
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(x
    print('Validation Accuracy : ', metrics.roc_auc_score(
        ytest, models[i].predict(xtest)))
    print()
```

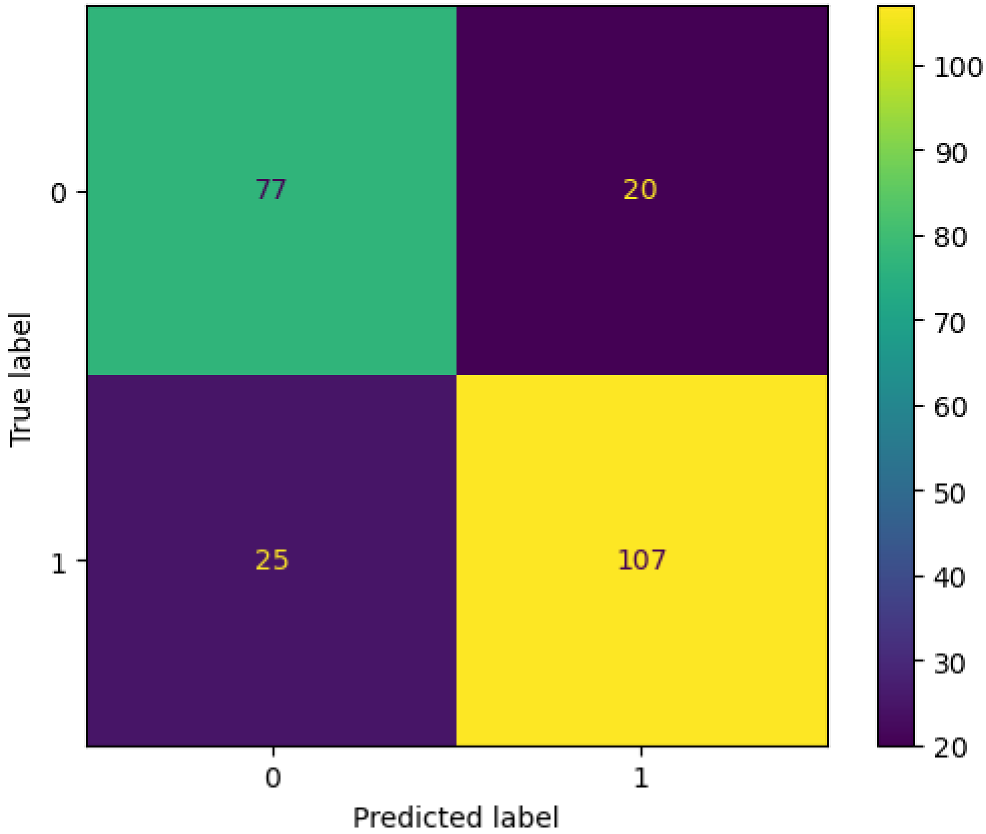
```
LogisticRegression() :
Training Accuracy :  0.7546950559364851
Validation Accuracy :  0.7255154639175256

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 :  0.8022102467978757

SVC() :
Training Accuracy :  0.7648213641284736
Validation Accuracy :  0.7358247422680412
```

In [ ]:
```
#Model Evaluation
From the above accuracies we can say that Logistic Regression and SVC() classifier
```

In [35]:
```
metrics.plot_confusion_matrix(models[1], xtest, ytest)
plt.show()
```



In [36]:
```
print(metrics.classification_report(ytest,
                                    models[1].predict(xtest)))
```

```
              precision    recall  f1-score   support

           0       0.75      0.79      0.77        97
           1       0.84      0.81      0.83       132

    accuracy                           0.80       229
   macro avg       0.80      0.80      0.80       229
weighted avg       0.81      0.80      0.80       229
```

In [ ]: