In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.tree import DecisionTreeClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split

from sklearn.metrics import classification_report

import warnings
warnings.filterwarnings('ignore')
```

In [7]:

```
df=pd.read_csv('winequalitynew.csv')
```

In [8]:

```
df.head()
```

...

In [9]:

```
df.info()
```

...

In [10]:

```
df.describe()
```

...

In [11]:

```
x=df.iloc[:,:-1]
y=df.iloc[:,-1]
```

In [12]:

```
x
```

...

In [13]:

```
y
```

...

In [14]:

```
for col in x:
    plt.figure()
    sns.distplot(x[col])
    plt.show()
```

...

In [15]:

```
for col in x:
    plt.figure()
    sns.boxplot(data=df,x="quality",y=col)
    plt.show()
```

...

In [18]:

```
plt.figure()
sns.countplot(y)
plt.show()
```

...

In [19]:

```
#multicolinaerity
df.corr()
```

...

In [20]:

```
plt.figure(figsize=(8,8))
sns.heatmap(x.corr(),annot=True)
plt.show()
```

...

In [21]:

```
#fixed acidity has coreleation with
#negative with PH
#positive with citric acid,density
```

In [22]:

```
# train_test_split
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=0.3,random_state=1)
```

In [38]:

```
dtc=DecisionTreeClassifier()
dtc.fit(xtrain,ytrain)
```

Out[38]:

```
DecisionTreeClassifier
DecisionTreeClassifier()
```

In [24]:

```
dtc.score(xtrain,ytrain)
```

Out[24]:

```
1.0
```

In [26]:

```
ypred=dtc.predict(xtest)
```

In [28]:

```
print(classification_report(ytest,ypred))
```

...

In [29]:

```
dtc.score(xtest,ytest)
```

Out[29]:

```
0.651356993736952
```

In [39]:

```
dtc1=DecisionTreeClassifier(criterion='entropy')
dtc1.fit(xtrain,ytrain)
```

Out[39]:

```
DecisionTreeClassifier
DecisionTreeClassifier(criterion='entropy')
```

In [35]:

```
ypred=dtc1.predict(xtest)
```

In [36]:

```
print(classification_report(ytest,ypred))
```

...

In [40]:

```
#depth
dtc.get_depth()
```

Out[40]:

```
21
```

In [50]:

```
#15,10,8
dtc3=DecisionTreeClassifier(max_depth=8)
dtc3.fit(xtrain,ytrain)
```

Out[50]:

```
DecisionTreeClassifier
DecisionTreeClassifier(max_depth=8)
```

In [51]:

```
ypred=dtc3.predict(xtest)
print(classification_report(ytest,ypred))
```

...

In [58]:

```
#min_samples_leaf 10,5,8,3
dtc4=DecisionTreeClassifier(min_samples_leaf=3)
dtc4.fit(xtrain,ytrain)
```

Out[58]:

```
DecisionTreeClassifier
DecisionTreeClassifier(min_samples_leaf=3)
```

In [59]:

```
ypred=dtc4.predict(xtest)
print(classification_report(ytest,ypred))
```

...

In [61]:

```
# random over sampling
from imblearn.under_sampling import RandomUnderSampler
from imblearn.over_sampling import RandomOverSampler
```

In [62]:

```
pip install imblearn
```

...

In [63]:

```
ros = RandomOverSampler(random_state=1)
```

In [69]:

```
X_sample2, y_sample2 = ros.fit_resample(xtrain,ytrain)
```

In [70]:

```
X_sample2
```

...

In [71]:

```
 y_sample2.value_counts()
```

...

In [ ]:

```

```