In [1]:

```python
import pandas as pd
import numpy as np
from scipy import stats
from sklearn.model_selection import train_test_split
```

In [2]:

```python
df = pd.read_csv("winequality.csv")
df
```

Out[2]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	0.45	8.8
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	0.49	9.5
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	0.44	10.1
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	9.9
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	9.9
...	...	...	...	...	...	...	...	...	...	...	...
4893	6.2	0.21	0.29	1.6	0.039	24.0	92.0	0.99114	3.27	0.50	11.2
4894	6.6	0.32	0.36	8.0	0.047	57.0	168.0	0.99490	3.15	0.46	9.6
4895	6.5	0.24	0.19	1.2	0.041	30.0	111.0	0.99254	2.99	0.46	9.4
4896	5.5	0.29	0.30	1.1	0.022	20.0	110.0	0.98869	3.34	0.38	12.8
4897	6.0	0.21	0.38	0.8	0.020	22.0	98.0	0.98941	3.26	0.32	11.8

4898 rows × 12 columns

In [3]:

```python
df.isnull().sum() # no missing value is found
```

Out[3]:

```
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
```

In [4]:

```python
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4898 entries, 0 to 4897
Data columns (total 12 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   fixed acidity         4898 non-null   float64
 1   volatile acidity      4898 non-null   float64
 2   citric acid           4898 non-null   float64
 3   residual sugar        4898 non-null   float64
 4   chlorides             4898 non-null   float64
 5   free sulfur dioxide   4898 non-null   float64
 6   total sulfur dioxide  4898 non-null   float64
 7   density               4898 non-null   float64
 8   pH                    4898 non-null   float64
 9   sulphates             4898 non-null   float64
 10  alcohol               4898 non-null   float64
 11  quality               4898 non-null   int64  
dtypes: float64(11), int64(1)
memory usage: 459.3 KB
```

In [5]:

```
x = df.drop(columns = "alcohol",axis=1)
y = df.alcohol
x
```

Out[5]:

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	quality
0	7.0	0.27	0.36	20.7	0.045	45.0	170.0	1.00100	3.00	0.45	6
1	6.3	0.30	0.34	1.6	0.049	14.0	132.0	0.99400	3.30	0.49	6
2	8.1	0.28	0.40	6.9	0.050	30.0	97.0	0.99510	3.26	0.44	6
3	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	6
4	7.2	0.23	0.32	8.5	0.058	47.0	186.0	0.99560	3.19	0.40	6
...	...	...	...	...	...	...	...	...	...	...	...
4893	6.2	0.21	0.29	1.6	0.039	24.0	92.0	0.99114	3.27	0.50	6
4894	6.6	0.32	0.36	8.0	0.047	57.0	168.0	0.99490	3.15	0.46	5
4895	6.5	0.24	0.19	1.2	0.041	30.0	111.0	0.99254	2.99	0.46	6
4896	5.5	0.29	0.30	1.1	0.022	20.0	110.0	0.98869	3.34	0.38	7
4897	6.0	0.21	0.38	0.8	0.020	22.0	98.0	0.98941	3.26	0.32	6

4898 rows × 11 columns

In [6]:

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size = 0.2,random_state = 1
print("shape of train data is :",x_train.shape,y_train.shape)
print("shape of test data is :",x_test.shape,y_test.shape)
```

```
shape of train data is : (3918, 11) (3918,)
shape of test data is : (980, 11) (980,)
```

In [12]:

```
data = {'Model Name': [], 'MAPE': [],'RMSE':[],'RMSLE':[]}
# Create DataFrame.
```

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

Out[12]:

Model Name	MAPE	RMSE	RMSLE

In [13]:

```
#1.lineqqr regression
from sklearn.linear_model import LinearRegression
model_ler = LinearRegression().fit(x_train,y_train)
pred = model_ler.predict(x_test)

# RMSE
rmse = np.sqrt(np.mean((y_test - pred)**2))

# MAPE
mape = (np.mean(np.abs(((y_test - pred)/y_test))))*100

#RMSLE
rmsle = np.sqrt(np.square(np.log(pred + 1) - np.log(y_test + 1)).mean())

df1 = {'Model Name': 'Linear regression', 'MAPE':mape,'RMSE':rmse,'RMSLE':rmsle}
df = df._append(df1, ignore_index = True)
df
```

Out[13]:

	Model Name	MAPE	RMSE	RMSLE
0	Linear regression	2.947415	0.398478	0.035545

In [19]:

```
#ridge regression
from sklearn.linear_model import Ridge
ridgeReg = Ridge(alpha=0.0005)
ridgeReg.fit(x_train,y_train)
pred = ridgeReg.predict(x_test)

# RMSE
rmse = np.sqrt(np.mean((y_test - pred)**2))

# MAPE
mape = (np.mean(np.abs(((y_test - pred)/y_test))))*100

#RMSLE
rmsle = np.sqrt(np.square(np.log(pred + 1) - np.log(y_test + 1)).mean())

df1 = {'Model Name': 'Ridge regression', 'MAPE':mape,'RMSE':rmse,'RMSLE':rmsle}
df = df._append(df1, ignore_index = True)
df
```

Out[19]:

	Model Name	MAPE	RMSE	RMSLE
0	Linear regression	2.947415	0.398478	0.035545
1	Ridge regression	3.078929	0.411886	0.036452

In [21]:

```
# lasso regression
from sklearn.linear_model import Lasso
lassoReg = Lasso(alpha=0.0005)
lassoReg.fit(x_train,y_train)
pred = lassoReg.predict(x_test)
```

```
# RMSE
rmse = np.sqrt(np.mean((y_test - pred)**2))

# MAPE
mape = (np.mean(np.abs(((y_test - pred)/y_test))))*100

#RMSLE
rmsle = np.sqrt(np.square(np.log(pred + 1) - np.log(y_test + 1)).mean())

df1 = {'Model Name': 'Lasso regression', 'MAPE':mape,'RMSE':rmse,'RMSLE':rmsle}
df = df._append(df1, ignore_index = True)
df
```

Out[21]:

	Model Name	MAPE	RMSE	RMSLE
0	Linear regression	2.947415	0.398478	0.035545
1	Ridge regression	3.078929	0.411886	0.036452
2	Lasso regression	4.805111	0.626084	0.054117

In [22]:

```
from sklearn.tree import DecisionTreeRegressor

dtreg= DecisionTreeRegressor(max_depth=5)
dtreg.fit(x_train,y_train)
pred = dtreg.predict(x_test)

# RMSE
rmse = np.sqrt(np.mean((y_test - pred)**2))

# MAPE
mape = (np.mean(np.abs(((y_test - pred)/y_test))))*100

#RMSLE
rmsle = np.sqrt(np.square(np.log(pred + 1) - np.log(y_test + 1)).mean())

df1 = {'Model Name': 'Decision Tree regression', 'MAPE':mape,'RMSE':rmse,'RMSLE':rmsle
df = df._append(df1, ignore_index = True)
df
```

Out[22]:

	Model Name	MAPE	RMSE	RMSLE
0	Linear regression	2.947415	0.398478	0.035545
1	Ridge regression	3.078929	0.411886	0.036452
2	Lasso regression	4.805111	0.626084	0.054117
3	Decision Tree regression	4.154571	0.572211	0.049829

In [ ]:

In [ ]: