Assignment 4

Ai and ml

Name: **JASTI PAVAN**

Registration Number: **21BDS0309**

Task 1

```
import pandas as pd

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

```
<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
```

```
df.shape
```

```
(1599, 12)
```

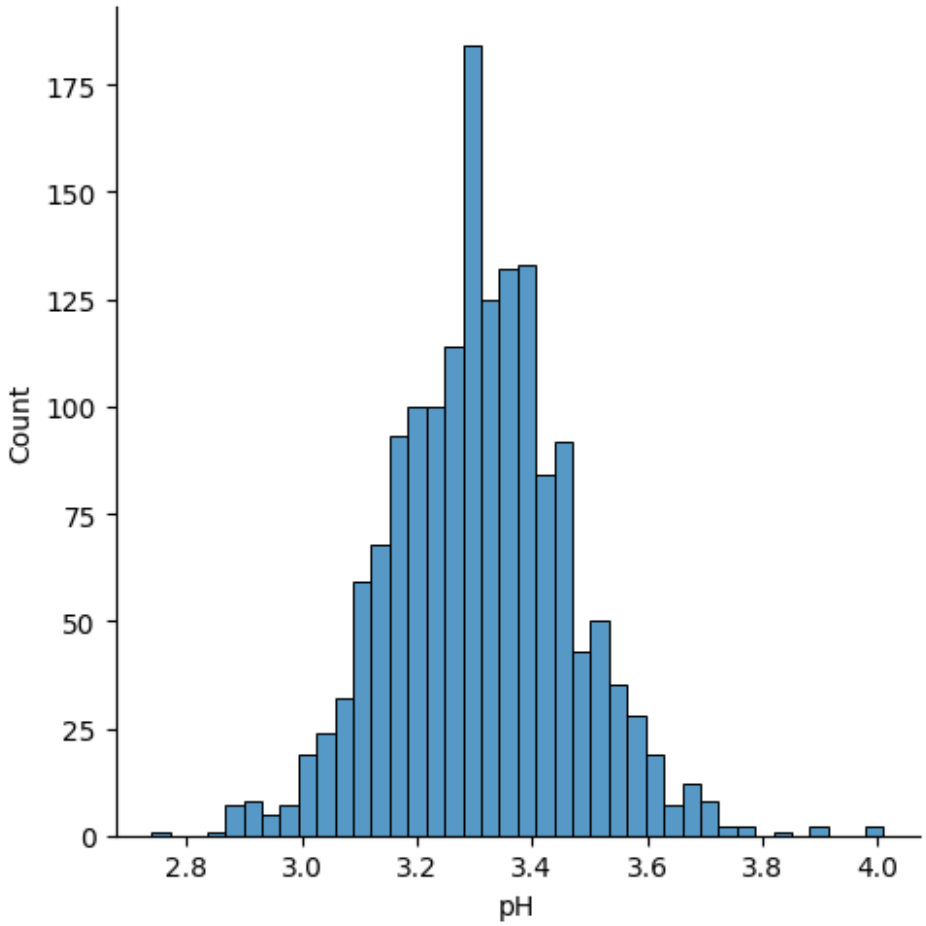
Task 2

Univariate

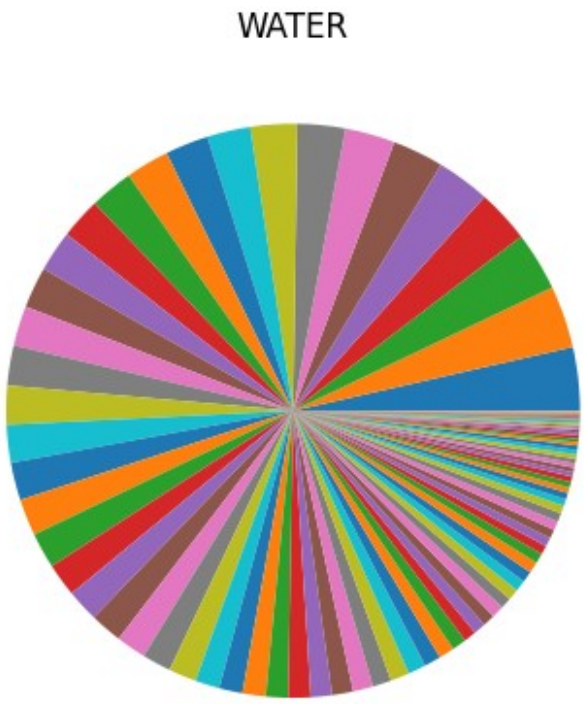
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

sns.displot(df.pH)
```

```
<seaborn.axisgrid.FacetGrid at 0x7bea221bfa90>
```



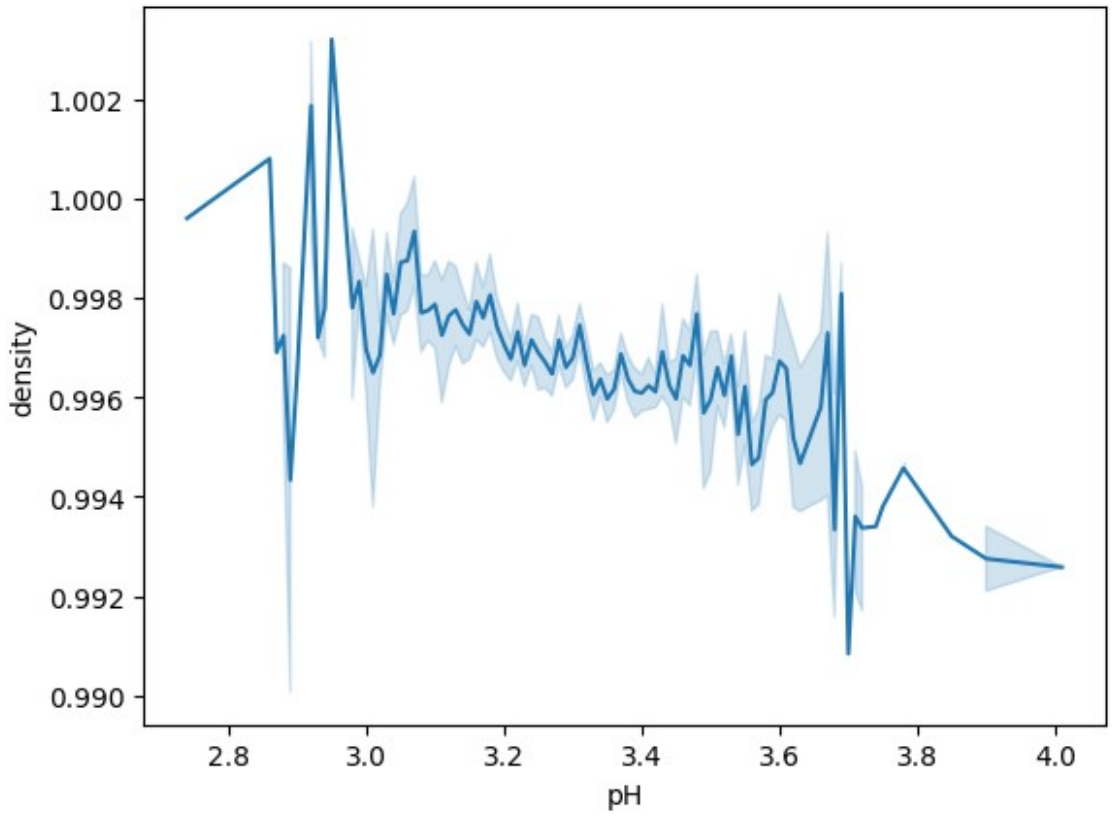

```
plt.pie(df.pH.value_counts())
plt.title('WATER')
plt.show()
```



Bivariate

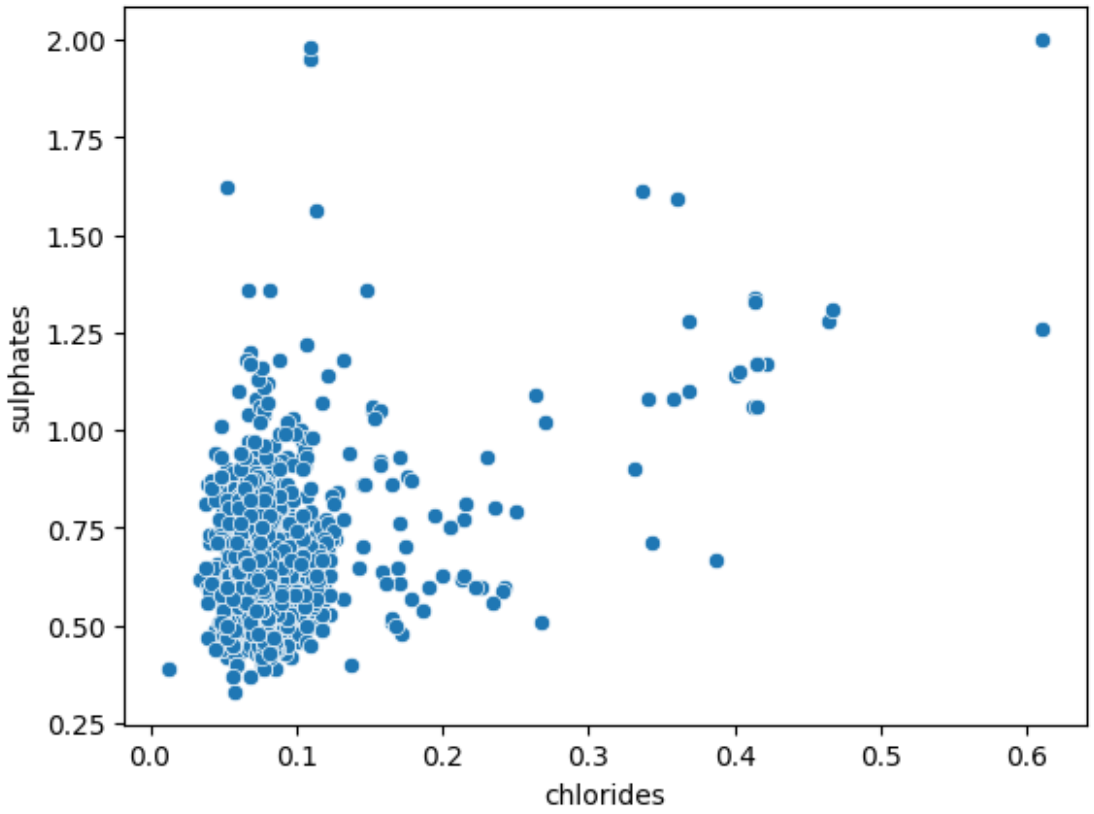
```
sns.lineplot(x=df.pH,y=df.density)
```

```
<Axes: xlabel='pH', ylabel='density'>
```




```
sns.scatterplot(x=df.chlorides, y=df.sulphates)
```

```
<Axes: xlabel='chlorides', ylabel='sulphates'>
```



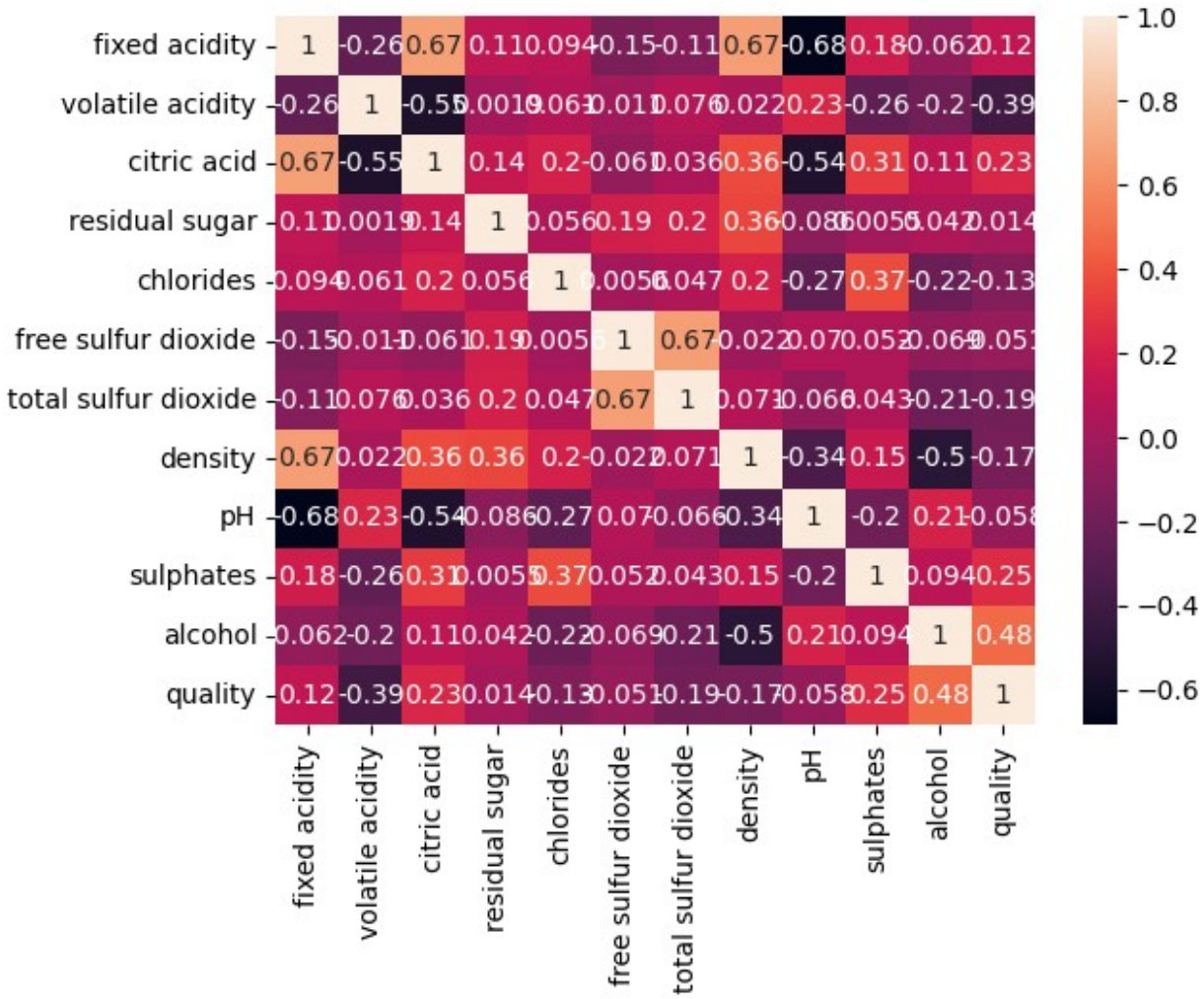
Multivariate

```
sns.pairplot(df)
```

```
Output hidden; open in https://colab.research.google.com to view.
```

```
sns.heatmap(df.corr(),annot=True)
```

```
<Axes: >
```



Task 3

```
from sklearn.linear_model import LinearRegression

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

```
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
```

```
df.value_counts()

fixed acidity  volatile acidity  citric acid  residual sugar
chlorides  free sulfur dioxide  total sulfur dioxide  density  pH
sulphates  alcohol  quality
7.5            0.510             0.02         1.7             0.084
13.0                31.0                 0.99538  3.36  0.54
10.5     6          4
6.7            0.460             0.24         1.7             0.077
18.0                34.0                 0.99480  3.39  0.60
10.6     6          4
7.2            0.360             0.46         2.1             0.074
24.0                44.0                 0.99534  3.40  0.85
11.0     7          4
               0.695             0.13         2.0             0.076
12.0                20.0                 0.99546  3.29  0.54
10.1     5          4
8.3            0.650             0.10         2.9             0.089
17.0                40.0                 0.99803  3.29  0.55
9.5      5          3

..
7.5            0.430             0.30         2.2             0.062
6.0                 12.0                 0.99495  3.44  0.72
11.5     7          1
               0.420             0.32         2.7             0.067
7.0                 25.0                 0.99628  3.24  0.44
10.4     5          1
                                  0.31         1.6             0.080
15.0                42.0                 0.99780  3.31  0.64
9.0      5          1
               0.410             0.15         3.7             0.104
29.0                94.0                 0.99786  3.14  0.58
9.1      5          1
15.9           0.360             0.65         7.5             0.096
22.0                71.0                 0.99760  2.98  0.84
14.9     5          1
Length: 1359, dtype: int64

df.nunique()

fixed acidity            96
volatile acidity        143
citric acid              80
residual sugar           91
chlorides               153
free sulfur dioxide      60
total sulfur dioxide    144
density                 436
pH                       89
```

```
sulphates               96
alcohol                 65
quality                  6
dtype: int64
```

```
y=df['pH']
y.head()
```

```
0    3.51
1    3.20
2    3.26
3    3.16
4    3.51
Name: pH, dtype: float64
```

```
X=df.drop(columns=['quality'],axis=1)
X.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
0      9.4
1      9.8
2      9.8
3      9.8
4      9.4
```

```
from sklearn.preprocessing import StandardScaler
scale=StandardScaler()

X_scaled=pd.DataFrame(scale.fit_transform(X),columns=X.columns)
X_scaled.head()
```

```
   fixed acidity  volatile acidity  citric acid  residual sugar
chlorides  \
0      -0.528360          0.961877    -1.391472       -0.453218  -
0.243707
1      -0.298547          1.967442    -1.391472        0.043416
0.223875
2      -0.298547          1.297065    -1.186070       -0.169427
0.096353
3       1.654856         -1.384443     1.484154       -0.453218  -
0.264960
4      -0.528360          0.961877    -1.391472       -0.453218  -
0.243707

   free sulfur dioxide  total sulfur dioxide   density        pH
sulphates  \
0            -0.466193             -0.379133  0.558274  1.288643  -
0.579207
1             0.872638              0.624363  0.028261 -0.719933
0.128950
2            -0.083669              0.229047  0.134264 -0.331177  -
0.048089
3             0.107592              0.411500  0.664277 -0.979104  -
0.461180
4            -0.466193             -0.379133  0.558274  1.288643  -
0.579207

    alcohol
0 -0.960246
1 -0.584777
2 -0.584777
3 -0.584777
4 -0.960246
```

```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, random_state=0)

x_train.shape
```

```
(1279, 11)
```

```
x_test.shape
```

```
(320, 11)
```

```
x_test.head()
```

```
      fixed acidity  volatile acidity  citric acid  residual sugar
chlorides  \
1109       1.425044         -0.323013     0.816598       -0.311323
1.775397
1032      -0.126188          1.632254    -1.391472        1.107633
0.160114
1002       0.448342         -1.328579     0.303093       -0.346797  -
0.520005
487        1.080326          0.654620     0.457144       -0.524166  -
0.732542
979        2.229387         -0.434742     1.124700       -0.807957  -
0.264960

      free sulfur dioxide  total sulfur dioxide   density        pH  \
1109             1.063900              0.593954  0.770280 -0.914312
1032            -1.039977             -0.987312  0.950485  0.316751
1002            -0.274931             -0.591995 -0.840962 -0.331177
487             -1.039977             -0.987312  0.770280 -0.914312
979             -1.231239             -1.230584  0.081262 -1.173483

      sulphates   alcohol
1109   0.601055  0.353895
1032  -0.756246 -0.772512
1002   1.073160  1.198701
487   -1.405390 -0.397043
979   -0.166115 -0.021574
```

```
y_test
```

```
1109    3.17
1032    3.36
1002    3.26
487     3.17
979     3.13
        ...
794     3.17
813     3.44
1322    3.18
704     3.29
1023    3.27
Name: pH, Length: 320, dtype: float64
```

```
from sklearn.linear_model import LinearRegression
lr=LinearRegression()

lr.fit(x_train, y_train)
```

```
LinearRegression()
```

```
y_predict=lr.predict(x_test)
y_predict
```

```
array([3.17, 3.36, 3.26, 3.17, 3.13, 3.13, 3.12, 3.47, 3.3 , 3.27, 3.2
,
       3.14, 3.29, 3.74, 3.5 , 3.34, 3.33, 3.15, 3.55, 3.38, 3.35,
3.27,
       3.36, 3.3 , 3.57, 3.27, 3.23, 3.35, 3.2 , 3.9 , 3.14, 3.31,
3.53,
       3.5 , 3.52, 3.26, 3.46, 3.69, 3.62, 3.35, 3.33, 3.37, 3.49,
3.02,
       3.3 , 3.19, 3.16, 3.3 , 3.58, 3.36, 3.41, 3.38, 3.38, 3.3 ,
3.32,
       3.35, 3.3 , 3.09, 3.23, 3.27, 3.68, 3.32, 3.32, 3.56, 3.51,
3.42,
       3.29, 2.92, 3.19, 3.07, 3.3 , 3.04, 3.38, 3.17, 3.34, 3.58,
3.46,
       3.45, 3.35, 3.3 , 3.1 , 3.39, 3.51, 3.14, 3.36, 3.4 , 3.44,
3.25,
       3.42, 3.44, 3.26, 3.22, 3.62, 3.35, 3.4 , 3.3 , 3.15, 3.19,
3.26,
       3.04, 3.23, 3.38, 3.29, 3.34, 3.46, 3.39, 3.46, 3.41, 3.42,
3.23,
       3.56, 3.1 , 3.23, 3.29, 3.33, 3.37, 3.25, 3.27, 3.45, 3.36,
3.23,
       2.92, 3.03, 3.36, 3.27, 3.3 , 3.23, 3.13, 3.36, 3.06, 3.4 ,
3.35,
       3.32, 3.22, 3.28, 3.29, 3.06, 3.  , 2.88, 3.36, 3.39, 3.19,
3.16,
       3.67, 3.17, 3.3 , 3.23, 3.33, 3.18, 3.34, 3.42, 3.39, 3.41,
3.29,
       3.61, 3.46, 3.25, 3.28, 3.44, 3.38, 3.48, 3.42, 3.32, 3.5 ,
3.14,
       3.25, 3.54, 3.41, 3.18, 3.27, 3.1 , 3.35, 3.48, 3.15, 3.52,
3.21,
       3.18, 3.34, 3.71, 3.16, 3.32, 3.38, 3.41, 3.2 , 3.39, 3.2 ,
3.26,
       3.29, 3.25, 3.42, 3.33, 2.94, 3.18, 3.29, 3.46, 3.28, 3.2 ,
3.39,
       3.28, 3.16, 3.19, 3.41, 3.26, 3.31, 3.26, 3.59, 3.24, 3.38,
3.36,
       3.42, 3.29, 3.25, 3.11, 3.18, 3.49, 3.47, 3.04, 3.39, 3.38,
3.28,
       3.51, 3.16, 3.2 , 3.03, 3.36, 3.12, 3.22, 3.56, 3.16, 3.39,
3.53,
       3.57, 3.35, 3.23, 3.13, 3.42, 3.07, 3.33, 3.36, 3.22, 3.48, 3.4
,
       3.36, 3.39, 3.24, 3.26, 3.29, 3.29, 3.  , 3.39, 3.19, 3.44,
3.23,
       3.26, 3.4 , 3.36, 3.21, 3.11, 3.48, 3.07, 3.44, 3.68, 3.14,
```

```
3.08,
       3.34, 3.36, 3.6 , 3.28, 3.28, 3.33, 3.14, 3.17, 3.24, 3.9 ,
3.22,
       3.39, 3.14, 3.21, 3.39, 3.05, 3.14, 3.27, 3.17, 3.26, 3.26,
3.55,
       3.36, 3.27, 3.34, 3.15, 3.49, 3.5 , 3.41, 3.19, 3.26, 3.38, 3.4
,
       3.29, 3.35, 3.15, 3.44, 3.56, 3.12, 3.31, 3.35, 3.61, 3.45,
3.48,
       3.37, 3.39, 3.36, 3.42, 3.66, 3.19, 3.2 , 3.17, 3.44, 3.18,
3.29,
       3.27])
```

```
y_predict1=lr.predict(x_train)
y_predict1
```

```
array([3.39, 3.13, 3.26, ..., 3.29, 3.3 , 3.25])
```

```
profit=pd.DataFrame({'Actual_pH':y_test,'Predicted_pH':y_predict})
profit
```

	Actual_pH	Predicted_pH
1109	3.17	3.17
1032	3.36	3.36
1002	3.26	3.26
487	3.17	3.17
979	3.13	3.13
...	...	...
794	3.17	3.17
813	3.44	3.44
1322	3.18	3.18
704	3.29	3.29
1023	3.27	3.27

[320 rows x 2 columns]

Task 4

```
from sklearn import metrics

print(metrics.r2_score(y_test,y_predict))

1.0

print(metrics.r2_score(y_train,y_predict1))

1.0

print(metrics.mean_squared_error(y_test,y_predict))

3.457429436163966e-31
```

```
print(np.sqrt(metrics.mean_squared_error(y_test,y_predict)))
```

```
5.879991017139368e-16
```

Task 5

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

```
lr.predict([[7.4, 0.70, 0.00, 1.9, 0.076, 11.0, 34.0, 0.9978, 3.51,
0.56, 9.4 ]])/100000
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439:
UserWarning: X does not have valid feature names, but LinearRegression
was fitted with feature names
  warnings.warn(

array([3.85284021e-05])
```

```
lr.predict([[7.8,0.88,0.00,2.6,0.098,25.0,67.0,0.9968,3.20,0.68,9.8]])
/100000

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439:
UserWarning: X does not have valid feature names, but LinearRegression
was fitted with feature names
  warnings.warn(

array([3.80499538e-05])

lr.predict([[7.8,0.88,0.00,2.6,0.098,25.0,67.0,0.9968,3.20,0.68,9.8]])
/100000

/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439:
UserWarning: X does not have valid feature names, but LinearRegression
was fitted with feature names
  warnings.warn(

array([3.80499538e-05])
```

---------- ***DONE*** ----------