```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
df=pd.read_csv('/content/water_quality.csv')
df
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	T
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	
...	...	...	...	...	...	...	...	...	
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	
3272	7.808856	193.553212	17329.802160	8.061362	NaN	392.449580	19.903225	NaN	
3273	9.419510	175.762646	33155.578218	7.350233	NaN	432.044783	11.039070	69.845400	
3274	5.126763	230.603758	11983.869376	6.303357	NaN	402.883113	11.168946	77.488213	
3275	7.874671	195.102299	17404.177061	7.509306	NaN	327.459760	16.140368	78.698446	

3276 rows × 10 columns

```
df.head()
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	Turb
0	NaN	204.890455	20791.318981	7.300212	368.516441	564.308654	10.379783	86.990970	2.96
1	3.716080	129.422921	18630.057858	6.635246	NaN	592.885359	15.180013	56.329076	4.50
2	8.099124	224.236259	19909.541732	9.275884	NaN	418.606213	16.868637	66.420093	3.05
3	8.316766	214.373394	22018.417441	8.059332	356.886136	363.266516	18.436524	100.341674	4.62
4	9.092223	181.101509	17978.986339	6.546600	310.135738	398.410813	11.558279	31.997993	4.07

```
df.tail()
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalomethanes	T
3271	4.668102	193.681735	47580.991603	7.166639	359.948574	526.424171	13.894419	66.687695	
3272	7.808856	193.553212	17329.802160	8.061362	NaN	392.449580	19.903225	NaN	
3273	9.419510	175.762646	33155.578218	7.350233	NaN	432.044783	11.039070	69.845400	
3274	5.126763	230.603758	11983.869376	6.303357	NaN	402.883113	11.168946	77.488213	
3275	7.874671	195.102299	17404.177061	7.509306	NaN	327.459760	16.140368	78.698446	

```
df.shape
```

```
(3276, 10)
```

```
df.dtypes
```

```
ph                float64
Hardness          float64
Solids            float64
Chloramines       float64
Sulfate           float64
Conductivity      float64
Organic_carbon    float64
Trihalomethanes   float64
Turbidity         float64
Potability          int64
dtype: object
```

```
df.isna().sum()
```

```
ph                491
```

```
Solids              0
Chloramines         0
Sulfate           781
Conductivity        0
Organic_carbon      0
Trihalomethanes   162
Turbidity           0
Potability          0
dtype: int64
```

```
df.describe()
```

	ph	Hardness	Solids	Chloramines	Sulfate	Conductivity	Organic_carbon	Trihalometha
count	2785.000000	3276.000000	3276.000000	3276.000000	2495.000000	3276.000000	3276.000000	3114.000
mean	7.080795	196.369496	22014.092526	7.122277	333.775777	426.205111	14.284970	66.396
std	1.594320	32.879761	8768.570828	1.583085	41.416840	80.824064	3.308162	16.175
min	0.000000	47.432000	320.942611	0.352000	129.000000	181.483754	2.200000	0.738
25%	6.093092	176.850538	15666.690297	6.127421	307.699498	365.734414	12.065801	55.844
50%	7.036752	196.967627	20927.833607	7.130299	333.073546	421.884968	14.218338	66.622
75%	8.062066	216.667456	27332.762127	8.114887	359.950170	481.792304	16.557652	77.337
max	14.000000	323.124000	61227.196008	13.127000	481.030642	753.342620	28.300000	124.000

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3276 entries, 0 to 3275
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   ph               2785 non-null   float64
 1   Hardness         3276 non-null   float64
 2   Solids           3276 non-null   float64
 3   Chloramines      3276 non-null   float64
 4   Sulfate          2495 non-null   float64
 5   Conductivity     3276 non-null   float64
 6   Organic_carbon   3276 non-null   float64
 7   Trihalomethanes  3114 non-null   float64
 8   Turbidity        3276 non-null   float64
 9   Potability       3276 non-null   int64  
dtypes: float64(9), int64(1)
memory usage: 256.1 KB
```

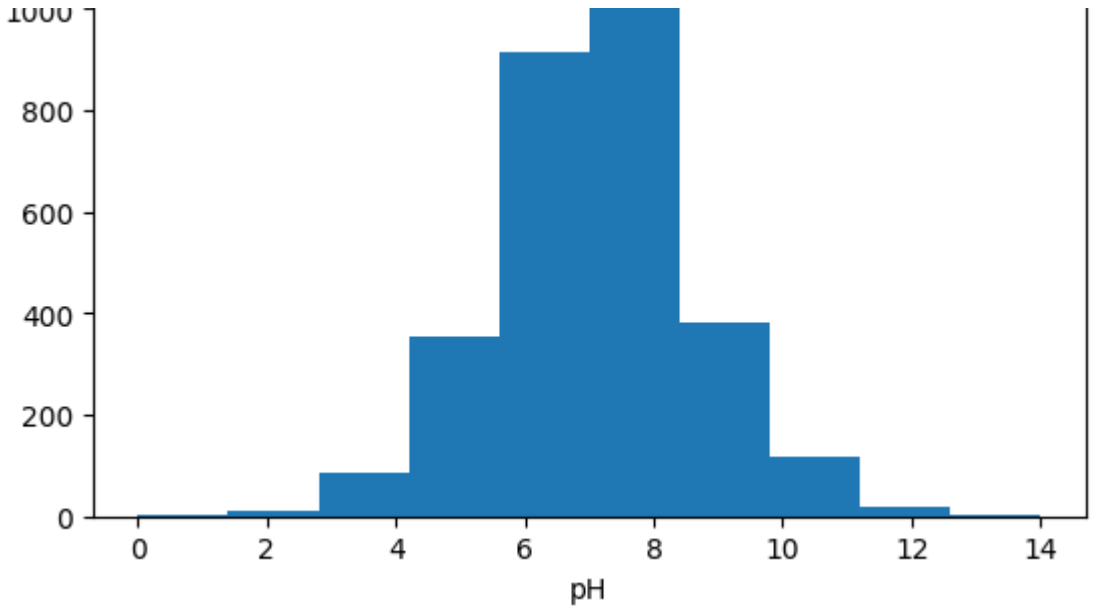
```
# Filling the missing values
df['ph'].fillna(df['ph'].mean(),inplace=True)
df['Sulfate'].fillna(df['Sulfate'].mean(),inplace=True)
df['Trihalomethanes'].fillna(df['Trihalomethanes'].mean(),inplace=True)
```

```
df.isna().sum()
```

```
ph                0
Hardness          0
Solids            0
Chloramines       0
Sulfate           0
Conductivity      0
Organic_carbon    0
Trihalomethanes   0
Turbidity         0
Potability        0
dtype: int64
```

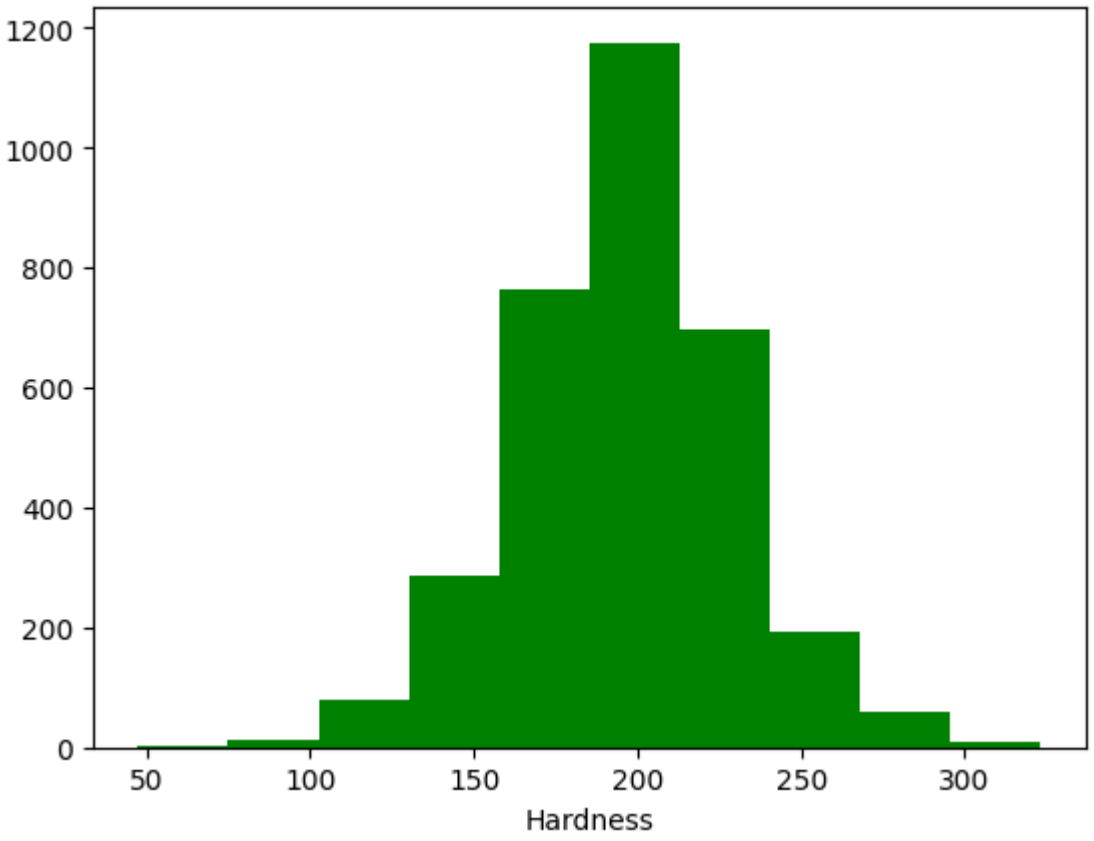
```
plt.hist(df['ph'])
plt.xlabel('pH')
```

```
Text(0.5, 0, 'pH')
```



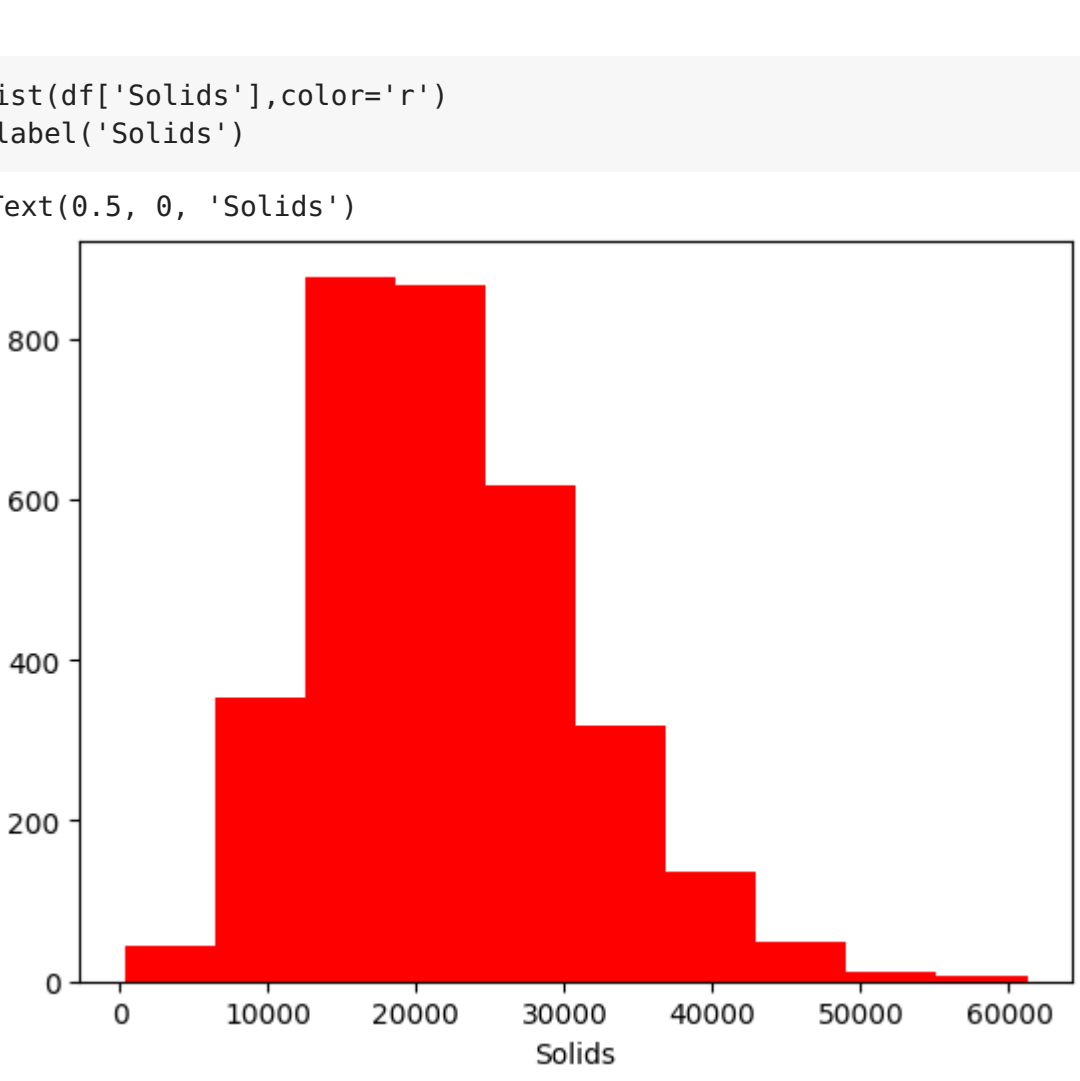

```
plt.hist(df['Hardness'],color='g')
plt.xlabel('Hardness')
```

```
Text(0.5, 0, 'Hardness')
```



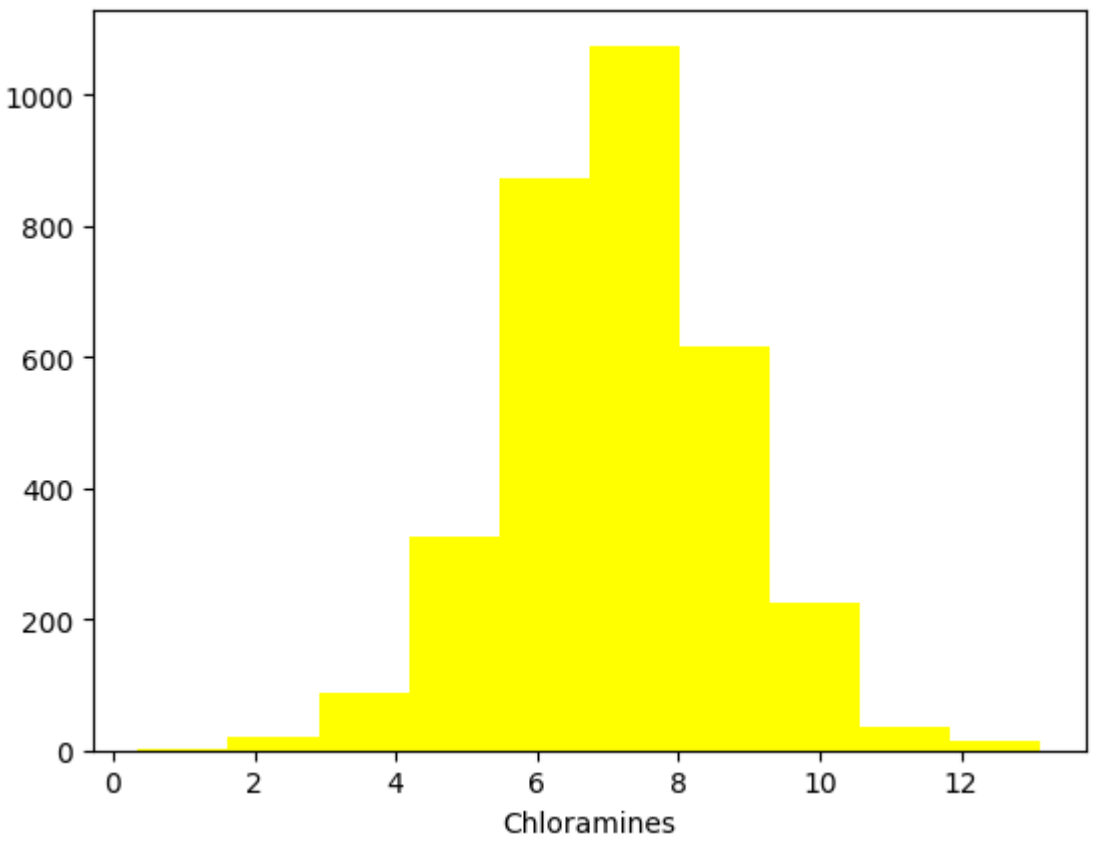

```
plt.hist(df['Solids'],color='r')
plt.xlabel('Solids')
```

```
Text(0.5, 0, 'Solids')
```



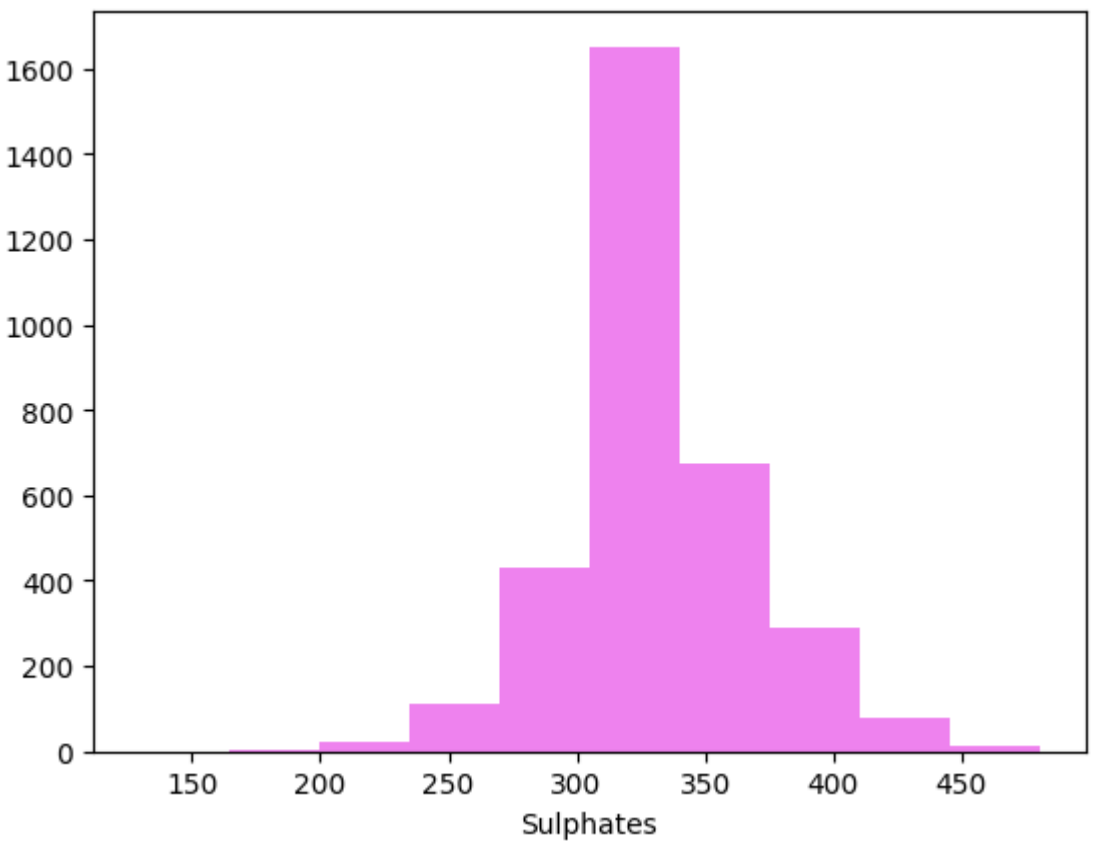

```
plt.hist(df['Chloramines'],color='yellow')
plt.xlabel('Chloramines')
```

```
Text(0.5, 0, 'Chloramines')
```



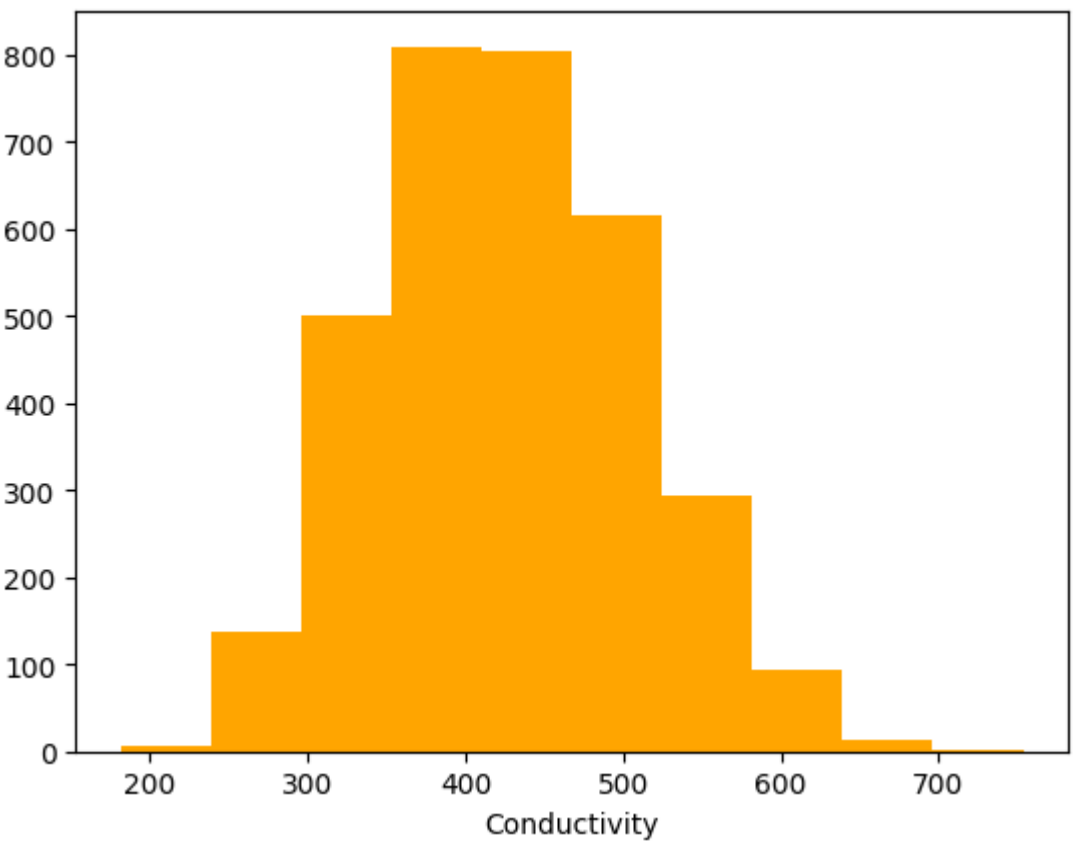

```
plt.hist(df['Sulfate'],color='violet')
plt.xlabel('Sulphates')
```

```
Text(0.5, 0, 'Sulphates')
```



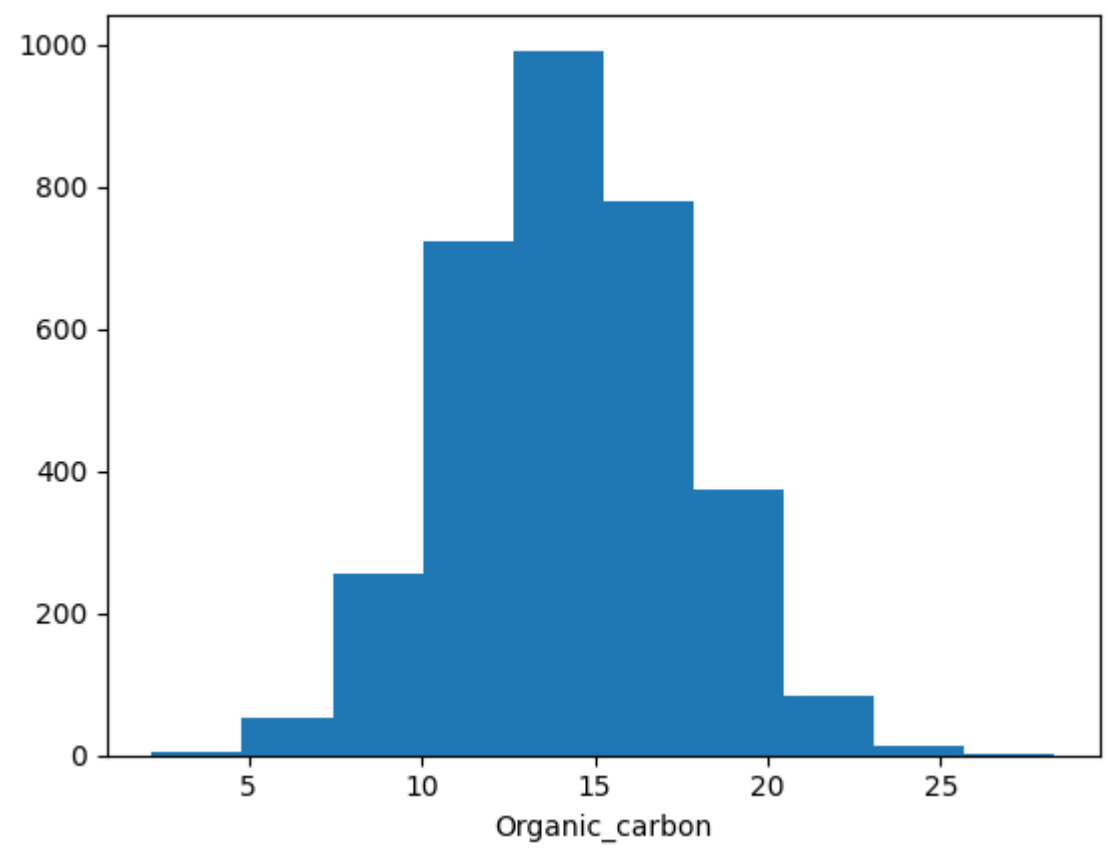

```
plt.hist(df['Conductivity'],color='orange')
plt.xlabel('Conductivity')
```

```
Text(0.5, 0, 'Conductivity')
```



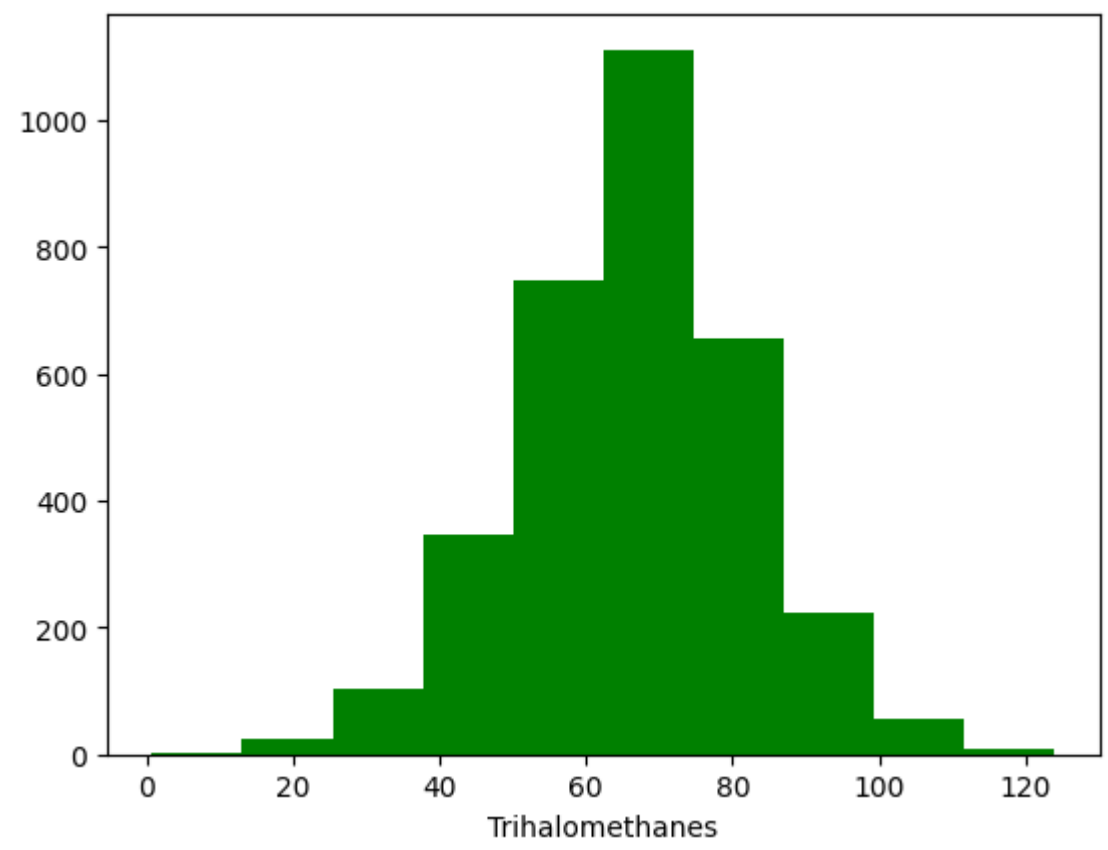
```
plt.hist(df['Organic_carbon'])
plt.xlabel('Organic_carbon')
```

```
Text(0.5, 0, 'Organic_carbon')
```



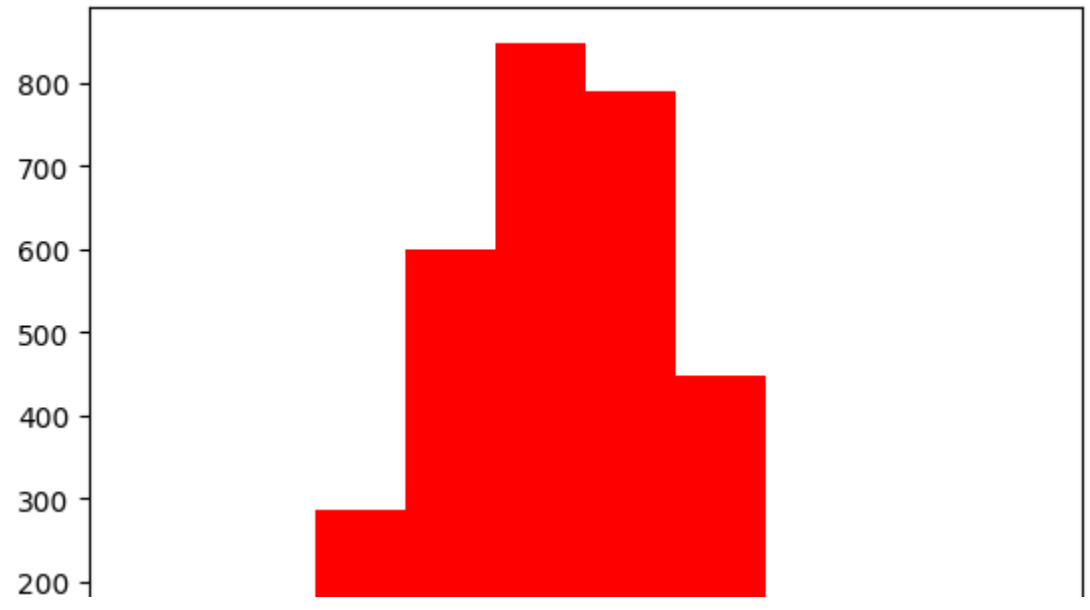

```
plt.hist(df['Trihalomethanes'],color='g')
plt.xlabel('Trihalomethanes')
```

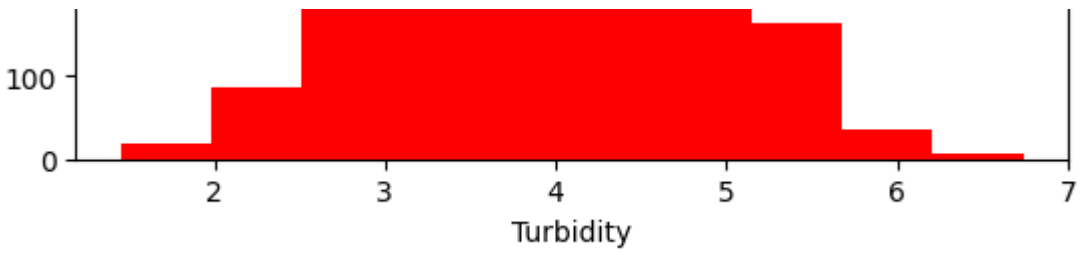
```
Text(0.5, 0, 'Trihalomethanes')
```



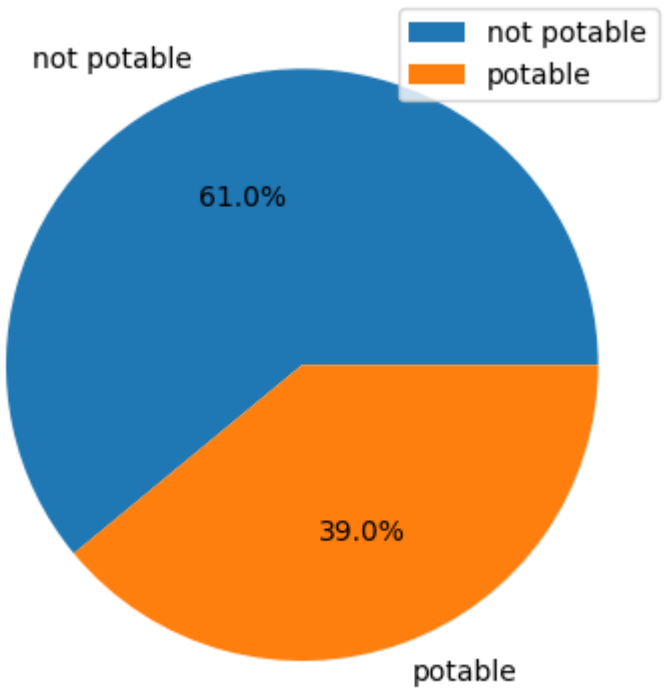

```
plt.hist(df['Turbidity'],color='r')
plt.xlabel('Turbidity')
```

```
Text(0.5, 0, 'Turbidity')
```






```
df1=df['Potability'].value_counts()
potability=['not potable','potable']
plt.pie(df1,labels=potability,autopct='%1.1f%%')
plt.legend()
plt.show()
df1
```




```
0    1998
1    1278
Name: Potability, dtype: int64
```

Separating input and output samples

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

```
(array([[7.08079450e+00, 2.04890455e+02, 2.07913190e+04, ...,
         1.03797831e+01, 8.69909705e+01, 2.96313538e+00],
        [3.71608008e+00, 1.29422921e+02, 1.86300579e+04, ...,
         1.51800131e+01, 5.63290763e+01, 4.50065627e+00],
        [8.09912419e+00, 2.24236259e+02, 1.99095417e+04, ...,
         1.68686369e+01, 6.64200925e+01, 3.05593375e+00],
        ...,
        [9.41951032e+00, 1.75762646e+02, 3.31555782e+04, ...,
         1.10390697e+01, 6.98454003e+01, 3.29887550e+00],
        [5.12676292e+00, 2.30603758e+02, 1.19838694e+04, ...,
         1.11689462e+01, 7.74882131e+01, 4.70865847e+00],
        [7.87467136e+00, 1.95102299e+02, 1.74041771e+04, ...,
         1.61403676e+01, 7.86984463e+01, 2.30914906e+00]]),
 array([0, 0, 0, ..., 1, 1, 1]))
```

Training and testing data

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
```

```
x_train
```

```
array([[7.08079450e+00, 1.88445469e+02, 2.87916144e+04, ...,
        1.05756901e+01, 6.32353650e+01, 3.22837922e+00],
       [7.20343885e+00, 1.68445358e+02, 2.28264847e+04, ...,
        1.64106541e+01, 6.45059226e+01, 6.38916101e+00],
       [7.08079450e+00, 2.42827588e+02, 2.92980743e+04, ...,
        5.42664993e+00, 6.63962929e+01, 3.52258617e+00],
```

```
           ...,
           [6.63036382e+00, 1.86761088e+02, 3.09390232e+04, ...,
            1.38157569e+01, 8.67531175e+01, 3.49058757e+00],
           [8.78316762e+00, 2.18032840e+02, 1.61835866e+04, ...,
            1.63545202e+01, 4.71009821e+01, 4.27413688e+00],
           [6.69815387e+00, 1.98286268e+02, 3.46758628e+04, ...,
            1.21766783e+01, 6.63962929e+01, 3.75817965e+00]])
```

```
x_test
```

```
    array([[7.08079450e+00, 1.83521107e+02, 2.04612527e+04, ...,
            2.01790289e+01, 6.70199032e+01, 4.88663379e+00],
           [6.64315871e+00, 1.88913541e+02, 3.28738200e+04, ...,
            1.47068103e+01, 6.78448489e+01, 4.56219767e+00],
           [7.84605793e+00, 2.24058877e+02, 2.32641100e+04, ...,
            1.34067375e+01, 4.30751865e+01, 2.48796865e+00],
           ...,
           [7.28873666e+00, 2.33977810e+02, 3.16170754e+04, ...,
            1.22253418e+01, 3.18020242e+01, 3.53138466e+00],
           [7.24348215e+00, 1.96982379e+02, 2.97720192e+04, ...,
            1.41549548e+01, 5.06550008e+01, 4.76357851e+00],
           [6.65364952e+00, 1.72584512e+02, 3.48164445e+04, ...,
            1.58724745e+01, 6.79768694e+01, 4.87140619e+00]])
```

```
y_train
```

```
    array([0, 1, 1, ..., 0, 0, 1])
```

```
y_test
```

```
    array([0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0,
           0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0,
           0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0,
           1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1,
           0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0,
           0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1,
           0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0,
           1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1,
           0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1,
           0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0,
           0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
           0, 0, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1,
           0, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1,
           0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1,
           0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0,
           0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0,
           0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0,
           0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0,
           0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
           1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1,
           0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1,
           0, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 1, 0,
           1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0,
           0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1,
           1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0,
           1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1,
           1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0,
           1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1,
           0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0,
           0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0,
           0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1,
           1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
           1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0,
           0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
           0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
           0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
           0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
           0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0])
```

Normalisation

```
from sklearn.preprocessing import MinMaxScaler
scaler=MinMaxScaler()
```

```
scaler=MinMaxScaler()
scaler.fit(x_train)
x_train=scaler.transform(x_train)
x_test=scaler.transform(x_test)
```

```
x_train
```

```
array([[0.50577104, 0.46049122, 0.46386091, ..., 0.25926802, 0.47537151,
        0.33090165],
       [0.51453135, 0.38037276, 0.36526119, ..., 0.50312205, 0.48634123,
        0.93332328],
       [0.50577104, 0.67834057, 0.47223236, ..., 0.04408003, 0.50266227,
        0.38697532],
       ...,
       [0.47359742, 0.45374375, 0.49935618, ..., 0.39467646, 0.67841861,
        0.38087663],
       [0.62736912, 0.57901528, 0.2554584 , ..., 0.50077611, 0.3360708 ,
        0.53021533],
       [0.47843956, 0.49991248, 0.56112371, ..., 0.32617631, 0.50266227,
        0.4318777 ]])
```

```
x_test
```

```
array([[0.50577104, 0.44076471, 0.32616544, ..., 0.66060947, 0.50804638,
        0.6469527 ],
       [0.47451134, 0.46236627, 0.53133711, ..., 0.43191524, 0.51516878,
        0.58511758],
       [0.56043271, 0.60315498, 0.37249485, ..., 0.37758277, 0.30131296,
        0.18978488],
       ...,
       [0.52062405, 0.64288924, 0.51056394, ..., 0.32821004, 0.20398296,
        0.38865225],
       [0.51739158, 0.49468922, 0.48006636, ..., 0.40885217, 0.36675541,
        0.62349928],
       [0.47526068, 0.3969538 , 0.56344743, ..., 0.48063052, 0.51630862,
        0.64405044]])
```

Model creation

```
from sklearn.svm import SVC
model=SVC()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
y_pred
```

```
array([0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
```

```
0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])
```

```
y_test
```

```
array([0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0,
       0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0,
       0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0,
       1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1,
       0, 0, 1, 1, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1,
       0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 0, 1,
       0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1,
       1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1,
       1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 1,
       1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 1,
       0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1,
       0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1,
       1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0,
       0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0,
       0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0])
```

Performance evaluation

```
from sklearn.metrics import confusion_matrix,accuracy_score
result=confusion_matrix(y_test,y_pred)
score=accuracy_score(y_test,y_pred)
score
```

```
0.6856561546286877
```

```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

```
                  precision    recall  f1-score   support

               0       0.68      0.95      0.79       617
               1       0.75      0.23      0.35       366

        accuracy                           0.69       983
       macro avg       0.71      0.59      0.57       983
    weighted avg       0.71      0.69      0.63       983
```