assignment-4

NAME: Shuvam Jena(21BAI1131)
MAIL:shuvam.jena2021@vitstudent.ac.in

[1]:

#Grapes to Greatness: Machine Learning in Wine Quality Prediction

- **Task 1 : Load the Dataset**

[2]:

[2]:		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	0.997 8	3.51	0.56	
1	25.0	67.0		3.20	0.68	

		0.996 8		
2	15.0	54.0 0.997 0	3.26	0.65
3	17.0	60.0 0.998 0	3.16	0.58
4	11.0	34.0 0.997 8	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

- **Task 2 : Data preprocessing including visualization**

[3]:

[3]: (1599, 12)

[4]:

```
<class
'pandas.core.frame.DataFram
e'>RangeIndex: 1599 entries,
0 to 1598 Data columns
(total 12 columns):
 #   Column              Non-Null Count  Dtype
```

#	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

[5]:

```
dtypes: float64(11), int64(1)
memory usage: 150.0 KB
```

```
[5]: 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
```

[6]:

[6]:		fixed_acidity	volatile_acidity	citric_acid	residual_sugar \
	count	1599.000000	1599.000000	1599.000000	1599.000000

	mean	8.319637	0.527821	0.270976	2.538806
	std	1.741096	0.179060	0.194801	1.409928
	min	4.600000	0.120000	0.000000	0.900000
	25%	7.100000	0.390000	0.090000	1.900000
	50%	7.900000	0.520000	0.260000	2.200000
	75%	9.200000	0.640000	0.420000	2.600000
	max	15.900000	1.580000	1.000000	15.500000

	chlorides	free_sulfur_dioxide	total_sulfur_dioxide	density	\
count	1599.000000	1599.000000	1599.000000	1599.000000	
mean	0.087467	15.874922	46.467792	0.996747	
std	0.047065	10.460157	32.895324	0.001887	
min	0.012000	1.000000	6.000000	0.990070	
25%	0.070000	7.000000	22.000000	0.995600	
50%	0.079000	14.000000	38.000000	0.996750	
75%	0.090000	21.000000	62.000000	0.997835	
max	0.611000	72.000000	289.000000	1.003690	

	pH	sulphates	alcohol	quality
count	1599.000000	1599.000000	1599.000000	1599.000000

mean	3.311113	0.658149	10.422983	5.636023
std	0.154386	0.169507	1.065668	0.807569
min	2.740000	0.330000	8.400000	3.000000
25%	3.210000	0.550000	9.500000	5.000000
50%	3.310000	0.620000	10.200000	6.000000
75%	3.400000	0.730000	11.100000	6.000000
max	4.010000	2.000000	14.900000	8.000000

[7]:

[7]:	fixed_acidity	volatile_acidity	citric_acid	\
fixed_acidity	1.000000	-0.256131	0.671703	
volatile_acidity	-0.256131	1.000000	-0.552496	
citric_acid	0.671703	-0.552496	1.000000	
residual_sugar	0.114777	0.001918	0.143577	
chlorides	0.093705	0.061298	0.203823	
free_sulfur_dioxide	-0.153794	-0.010504	-0.060978	
total_sulfur_dioxide	-0.113181	0.076470	0.035533	
density	0.668047	0.022026	0.364947	
pH	-0.682978	0.234937	-0.541904	
sulphates	0.183006	-0.260987	0.312770	
alcohol	-0.061668	-0.202288	0.109903	
quality	0.124052	-0.390558	0.226373	

	residual_sugar	chlorides	free_sulfur_dioxide \
fixed_acidity	0.114777	0.093705	-0.153794
volatile_acidity	0.001918	0.061298	-0.010504
citric_acid	0.143577	0.203823	-0.060978
residual_sugar	1.000000	0.055610	0.187049

chlorides	0.055610	1.000000	0.005562
free_sulfur_dioxide	0.187049	0.005562	1.000000
total_sulfur_dioxide	0.203028	0.047400	0.667666
density	0.355283	0.200632	−0.021946
pH	−0.085652	−0.265026	0.070377
sulphates	0.005527	0.371260	0.051658
alcohol	0.042075	−0.221141	−0.069408
quality	0.013732	−0.128907	−0.050656

	total_sulfur_dioxide	density	pH	sulphates	\
fixed_acidity	−0.113181	0.668047	−0.682978	0.183006	
volatile_acidity	0.076470	0.022026	0.234937	−0.260987	
citric_acid	0.035533	0.364947	−0.541904	0.312770	
residual_sugar	0.203028	0.355283	−0.085652	0.005527	
chlorides	0.047400	0.200632	−0.265026	0.371260	
free_sulfur_dioxide	0.667666	−0.021946	0.070377	0.051658	
total_sulfur_dioxide	1.000000	0.071269	−0.066495	0.042947	
density	0.071269	1.000000	−0.341699	0.148506	
pH	−0.066495	−0.341699	1.000000	−0.196648	
sulphates	0.042947	0.148506	−0.196648	1.000000	
alcohol	−0.205654	−0.496180	0.205633	0.093595	
quality	−0.185100	−0.174919	−0.057731	0.251397	

	alcohol	quality

fixed_acidity	−0.061668	0.124052
volatile_acidity	−0.202288	−0.390558
citric_acid	0.109903	0.226373
residual_sugar	0.042075	0.013732
chlorides	−0.221141	−0.128907
free_sulfur_dioxide	−0.069408	−0.050656
total_sulfur_dioxide	−0.205654	−0.185100
density	−0.496180	−0.174919
pH	0.205633	−0.057731
sulphates	0.093595	0.251397
alcohol	1.000000	0.476166
quality	0.476166	1.000000

[8]:

[8]:

```
quality                 1.000000
alcohol                 0.476166
```

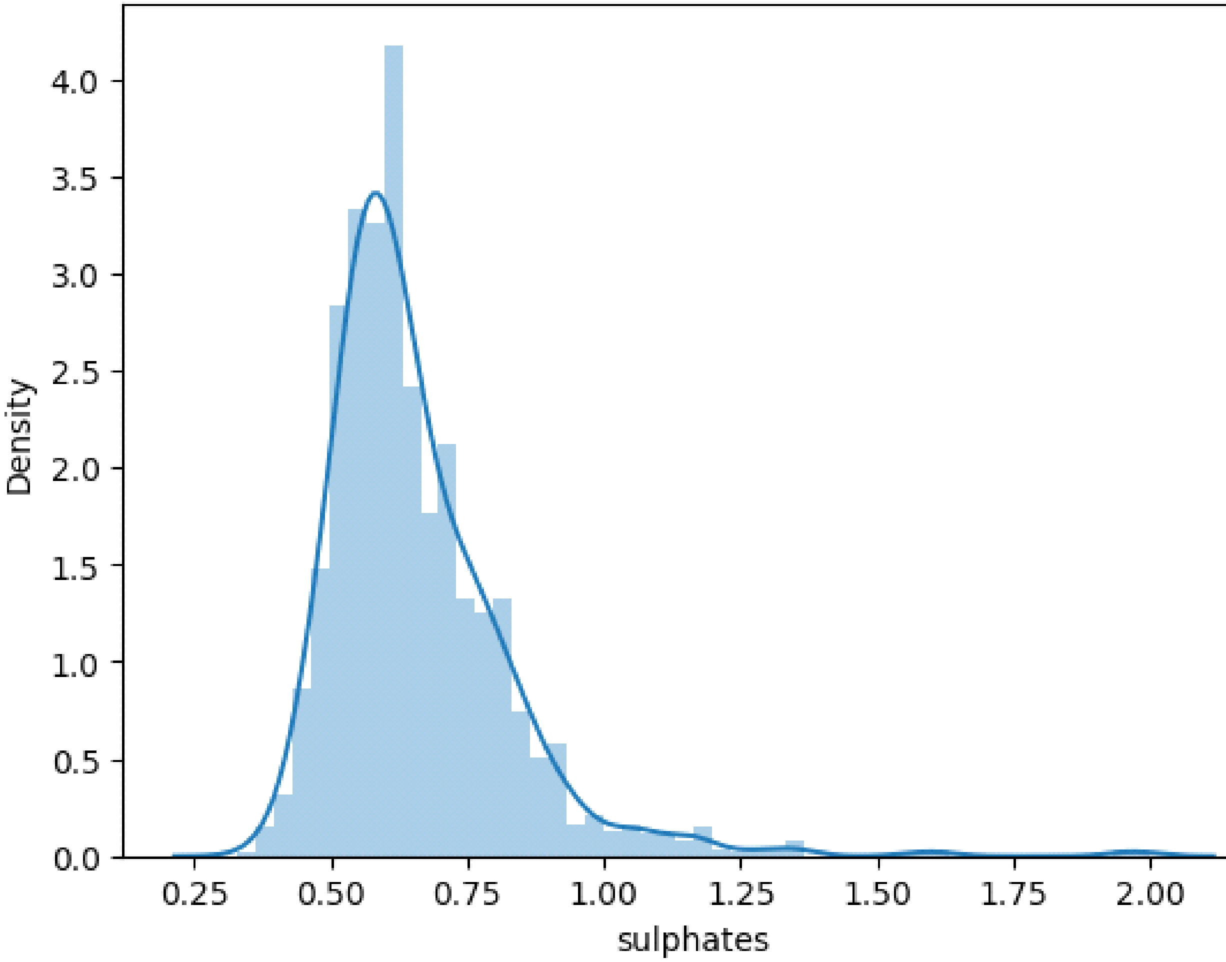
sulphates	0.251397
citric_acid	0.226373
fixed_acidity	0.124052
residual_sugar	0.013732
free_sulfur_dioxide	−0.050656
pH	−0.057731
chlorides	−0.128907
density	−0.174919
total_sulfur_dioxide	−0.185100
volatile_acidity	−0.390558
Name: quality, dtype:	float64

[9]:

Univariate Analysis

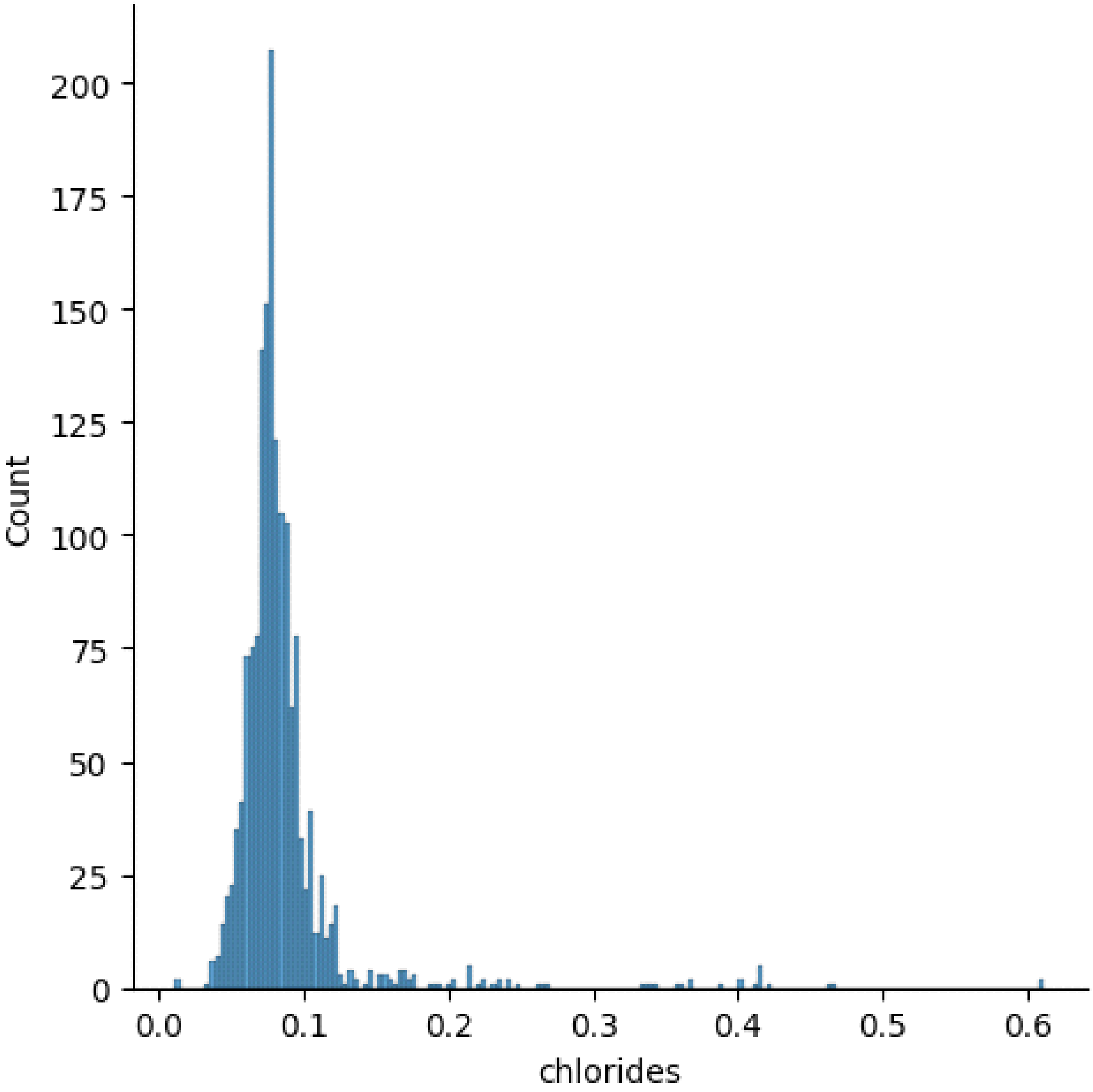
```
<ipython-input-9-8b271c44c149>:1:  UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please
see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5
751

  sns.distplot(df.sulphates)
```

```
  • : <Axes: xlabel='sulphates', ylabel='Density'>
```


4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Density
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
sulphates

[10]:

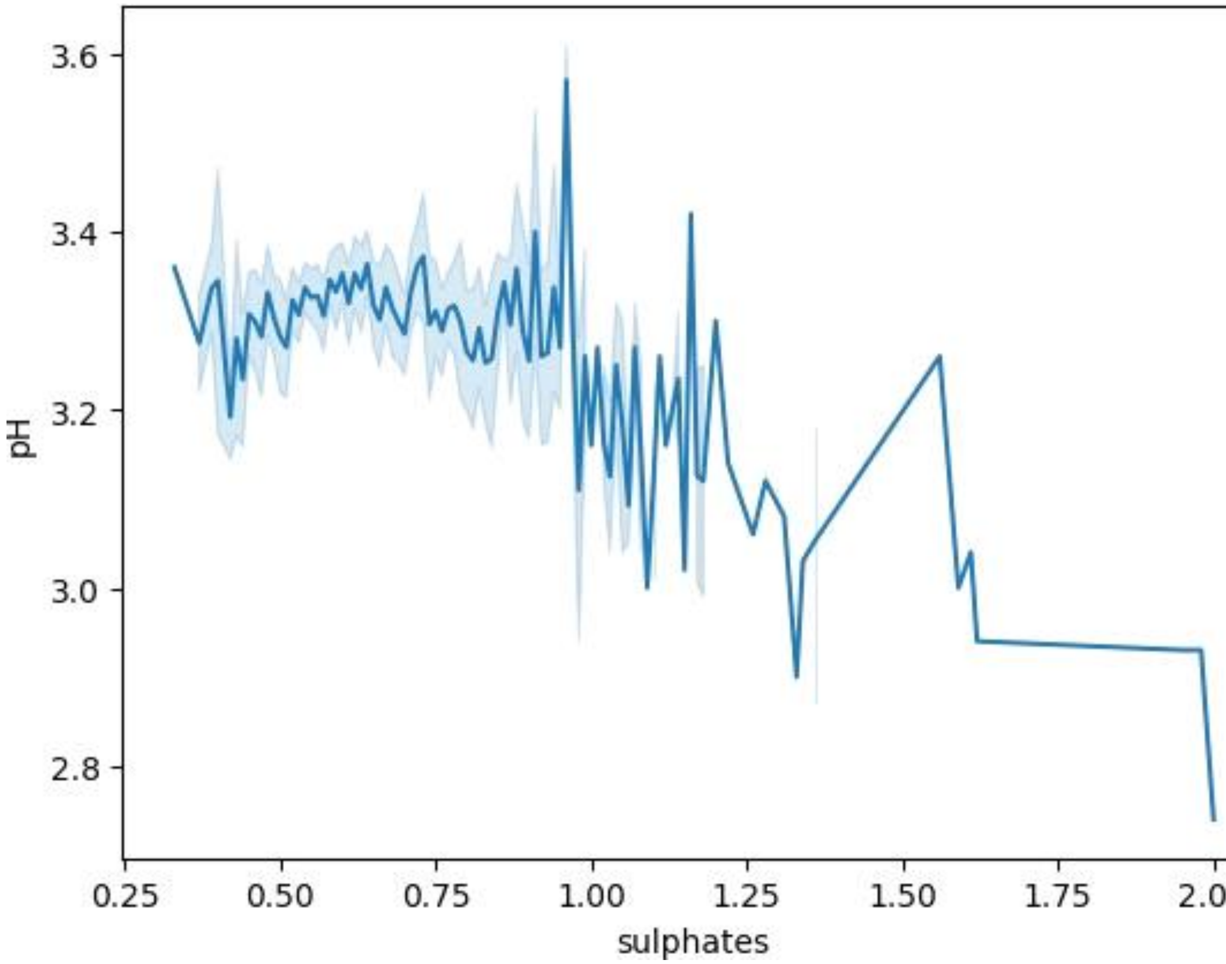
- : <seaborn.axisgrid.FacetGrid at 0x7ddd8a543160>



Bivariate Analysis

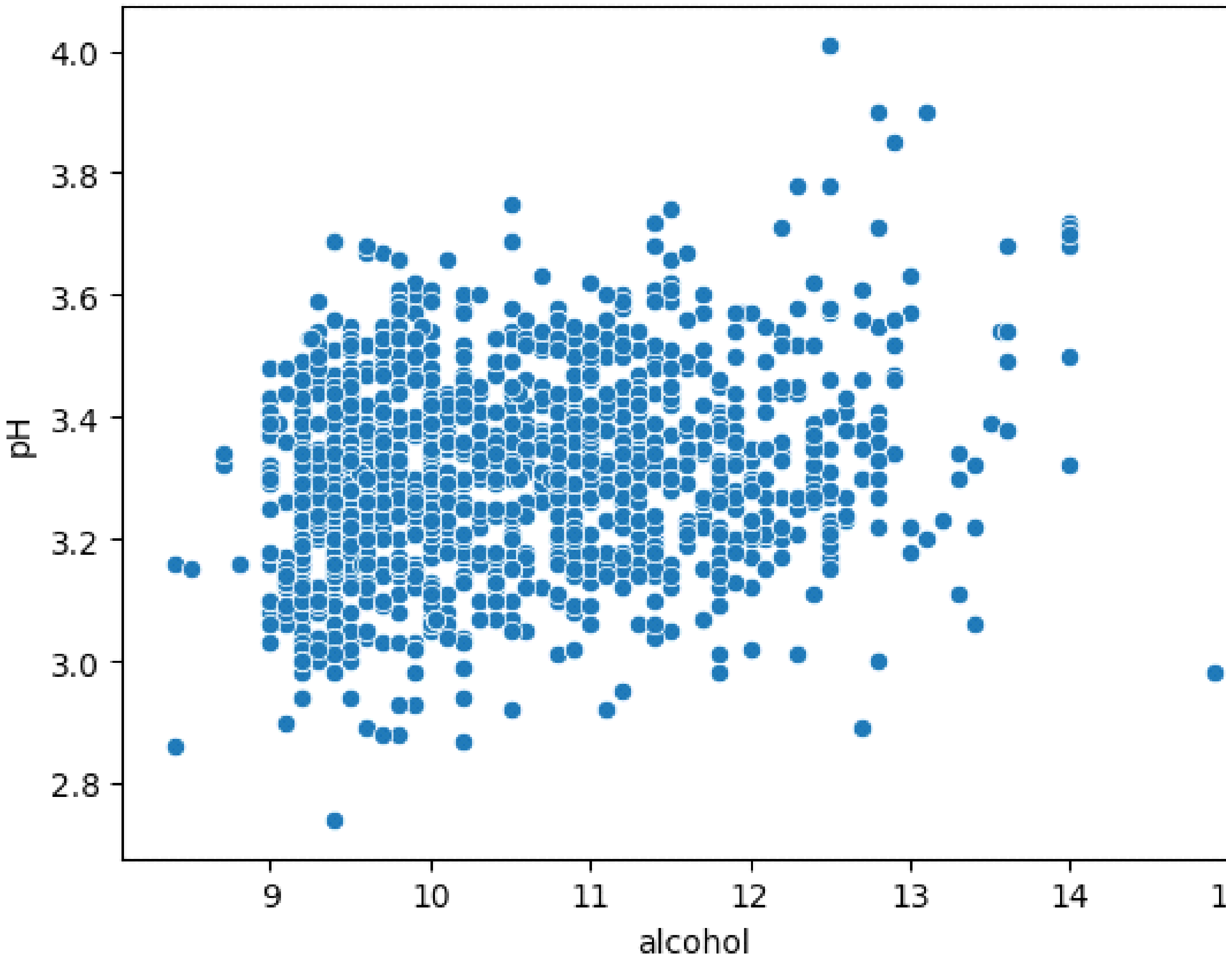
[11]:

- : <Axes: xlabel='sulphates', ylabel='pH'>



[12]:

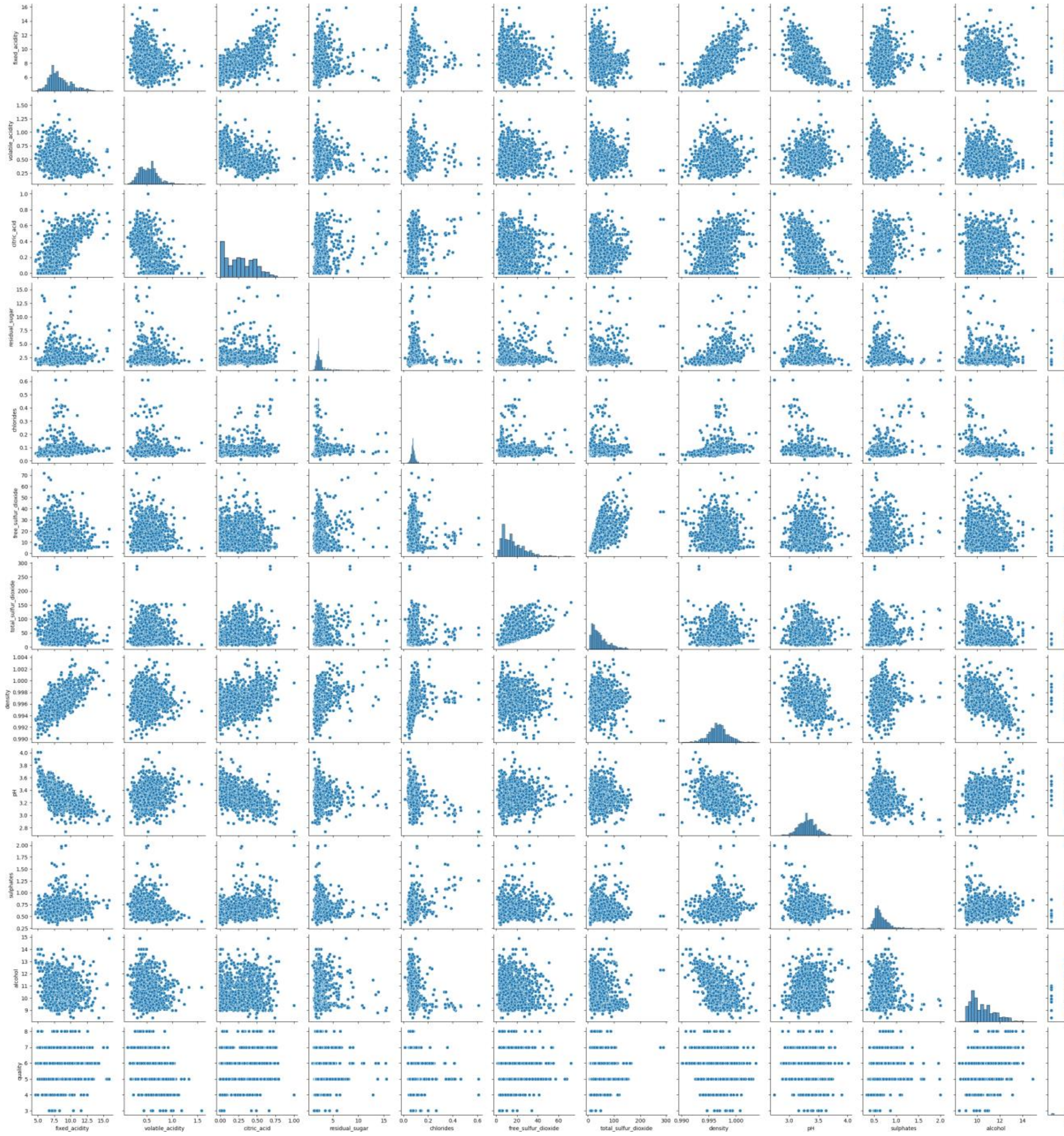
- : <Axes: xlabel='alcohol', ylabel='pH'>



Multivariate Analysis

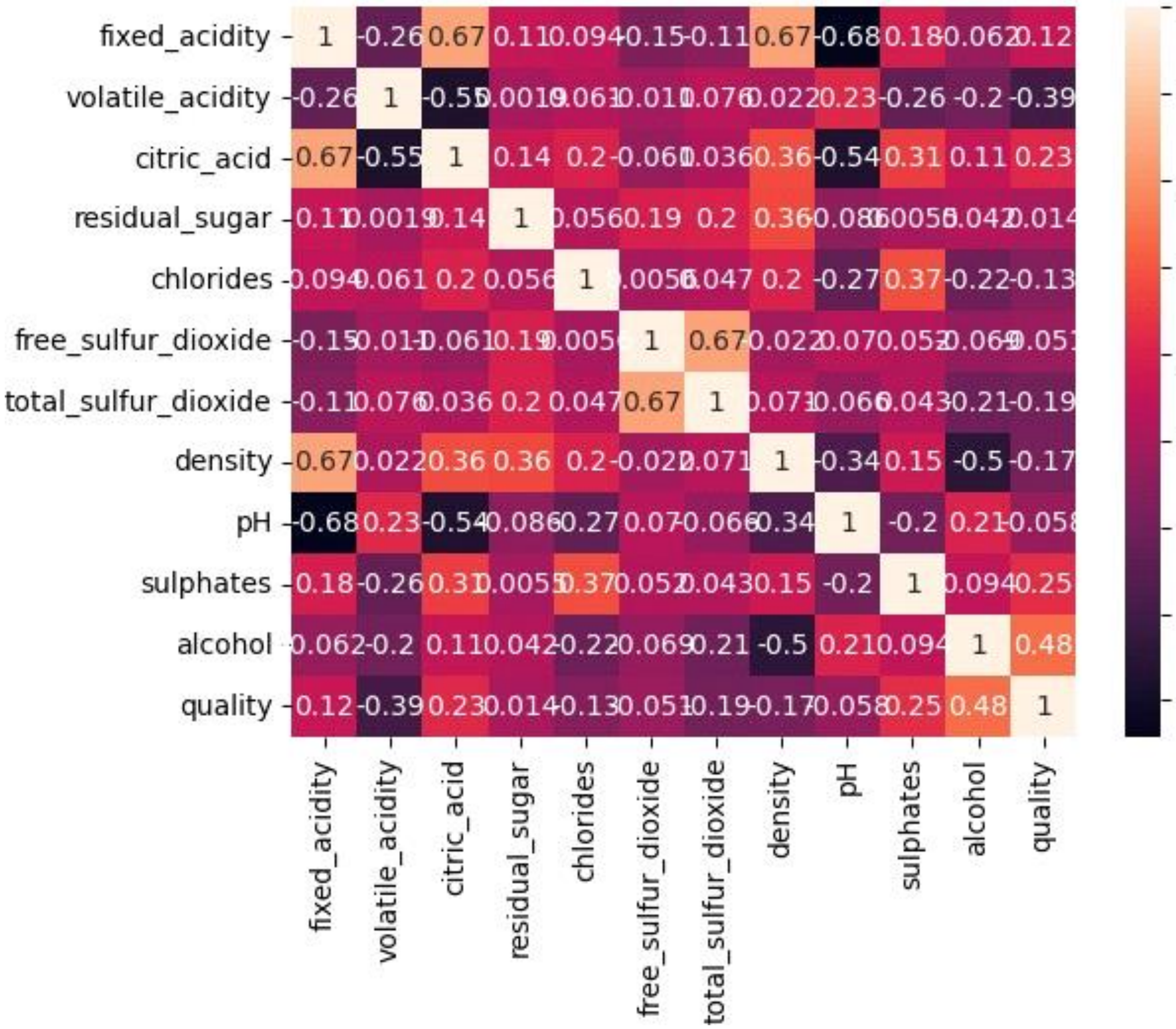
[13]:

- : <seaborn.axisgrid.PairGrid at 0x7ddd4f583280>


fixed_acidity
6
8
10
12
14
16
volatile_acidity
0.25
0.50
0.75
1.00
1.25
1.50
citric_acid
0.0
0.2
0.4
0.6
0.8
1.0
residual_sugar
2.5
5.0
7.5
10.0
12.5
15.0
chlorides
0.0
0.1
0.2
0.3
0.4
0.5
0.6
free_sulfur_dioxide
0
10
20
30
40
50
60
70
total_sulfur_dioxide
0
50
100
150
200
250
300
density
0.990
0.992
0.994
0.996
0.998
1.000
1.002
1.004
pH
2.8
3.0
3.2
3.4
3.6
3.8
4.0
sulphates
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
alcohol
9
10
11
12
13
14
15
quality
3
4
5
6
7
8
5.0 7.5 10.0 12.5 15.0
fixed_acidity
0.5 1.0 1.5
volatile_acidity
0.00 0.25 0.50 0.75 1.00
citric_acid
5 10 15
residual_sugar
0.0 0.2 0.4 0.6
chlorides
0 20 40 60
free_sulfur_dioxide
0 100 200 300
total_sulfur_dioxide
0.990 0.995 1.000
density
3.0 3.5 4.0
pH
0.5 1.0 1.5 2.0
sulphates
10 12 14
alcohol

[14]:

- : <Axes: >



Outlier Detection and removal by percentile method & IQR MEthod

[16]:

[16]:	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

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

alcohol quality

```
# Removing outliers from fixed_acidity column
```

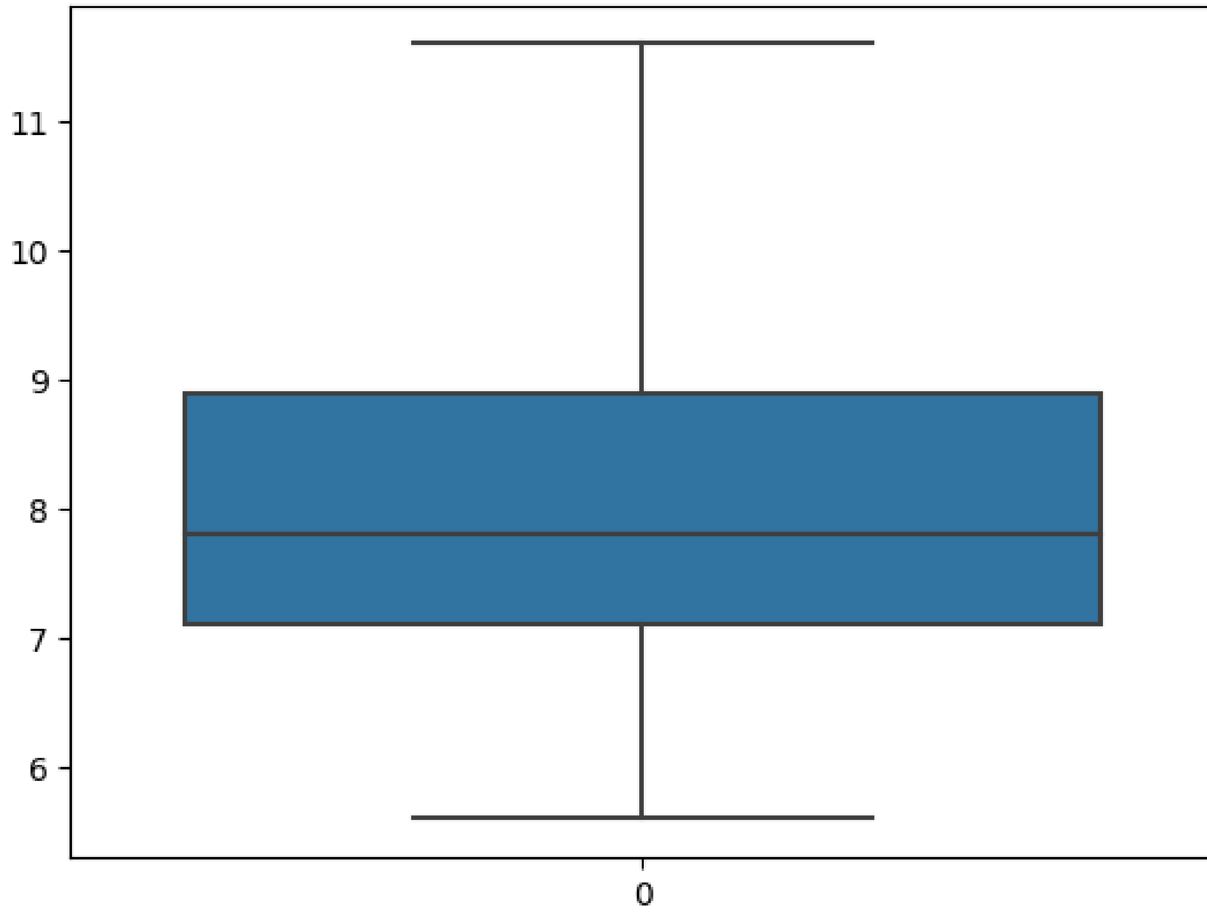
```
f1   =  df.fixed_acidity.quantile(0.25)  #Q1
f3   =  df.fixed_acidity.quantile(0.75)  #Q3
IQR_f = f3 - f1
upper_limit_f  =  f3+(1.5)*(IQR_f)
lower_limit_f  =  f1-(1.5)*(IQR_f)
print(f1)
print(f3) print(IQR_f)
print(upper_limit_f)
print(lower_limit_f)
```

[49]:

[51]:

```
7.1
8.9
1.8000000000000007
11.600000000000001
4.3999999999999999
```

[51]: <Axes: >

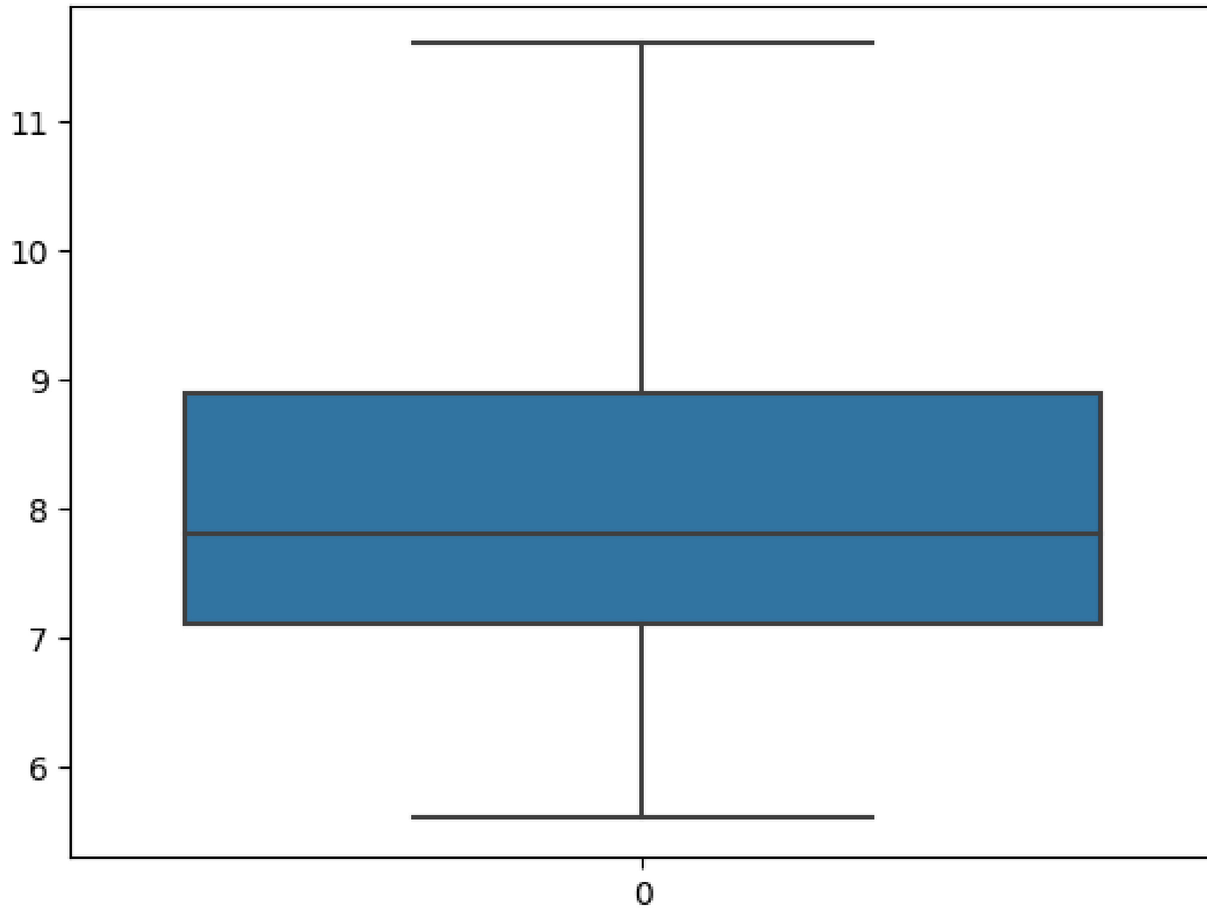


[47]:

```
5.6
11.6
```

[48]:

[48]: <Axes: >



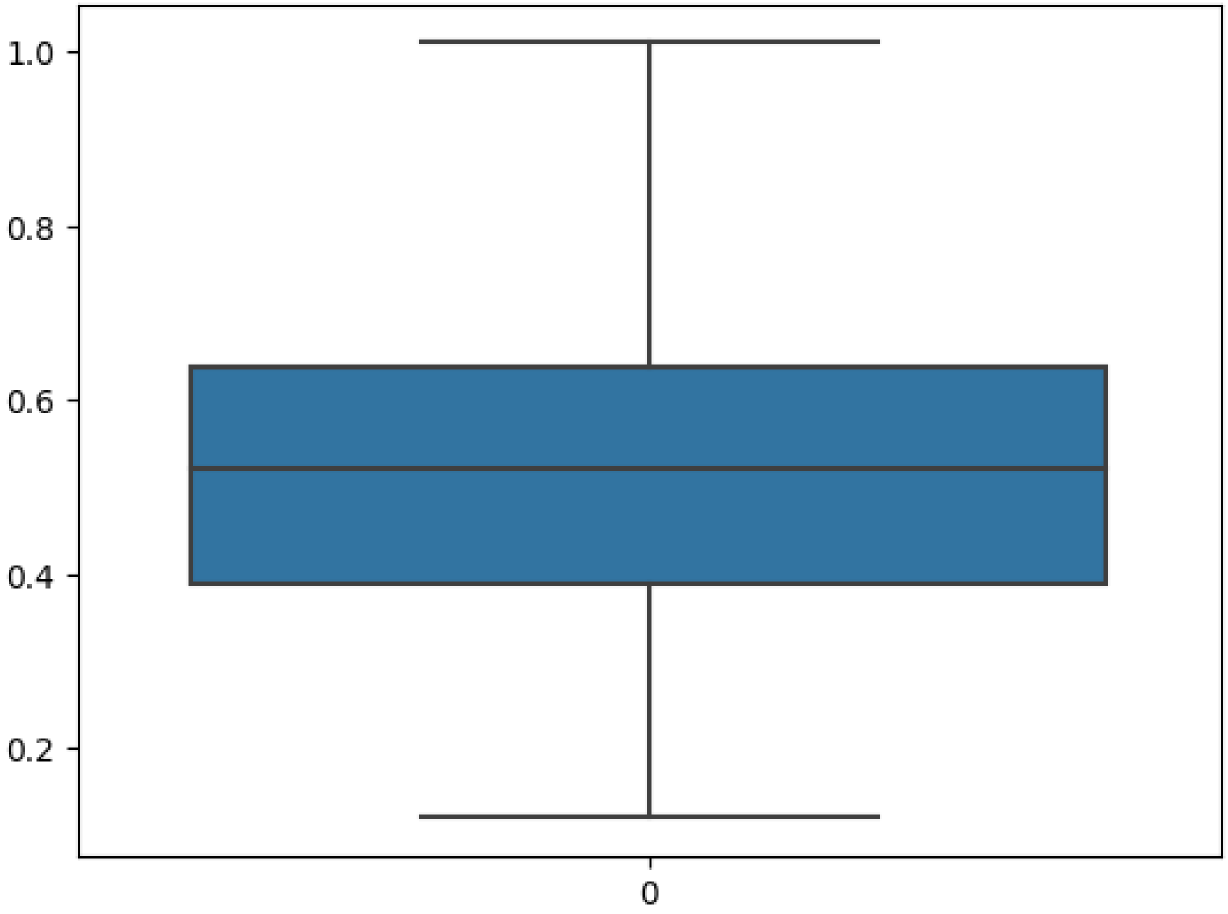
```
# Removing outliers from volatile_acidity column

v1 = df.volatile_acidity.quantile(0.25)    #Q1
v3  =  df.volatile_acidity.quantile(0.75)  #Q3
IQR_v = v3 - v1
upper_limit_v  =  v3+(1.5)*(IQR_v)
lower_limit_v  =  v1-(1.5)*(IQR_v)
print(v1)
print(v3) print(IQR_v)
print(upper_limit_v)
print(lower_limit_v)
```

[22]:

[23]:

```
0.3925
0.64
0.2475
1.01125
0.021250000000000047
```

[23]: `<Axes: >`

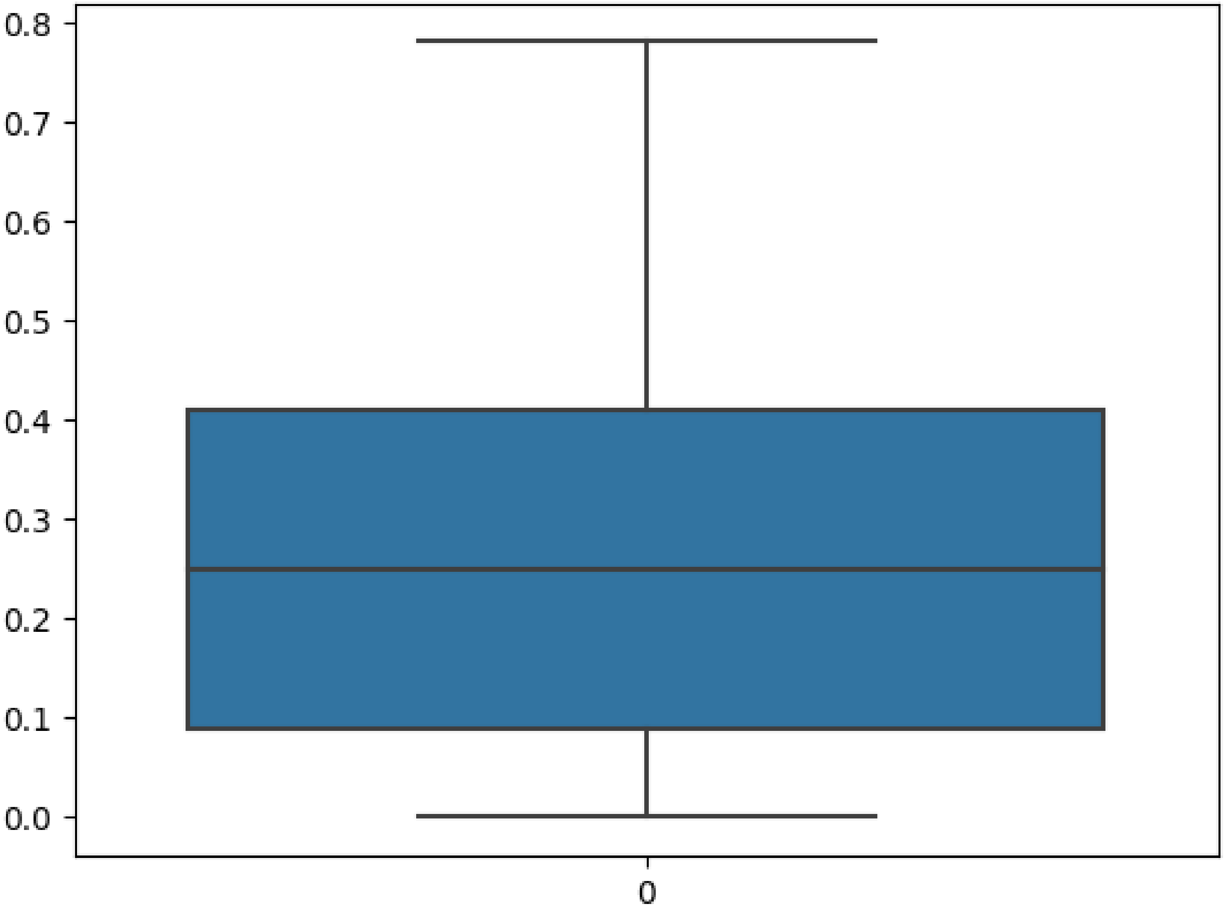


```
# Removing outliers from citric_acid column

c1   =  df.citric_acid.quantile(0.25)  #Q1
c3   =  df.citric_acid.quantile(0.75)  #Q3
IQR_c = c3 - c1
upper_limit_c  =  c3+(1.5)*(IQR_c)
lower_limit_c  =  c1-(1.5)*(IQR_c)
print(c1)
print(c3) print(IQR_c)
print(upper_limit_c)
print(lower_limit_c)
  [24]:
```

```
0.09
0.41
0.31999999999999995
0.8899999999999999
-0.3899999999999999
```

[25]:

[25]:
```
<Axes: >
```



```
# Removing outliers from residual_sugar column

r1 = df.residual_sugar.quantile(0.25) #Q1
r3 = df.residual_sugar.quantile(0.75) #Q3
IQR_r = r3 - r1
upper_limit_r = r3+(1.5)*(IQR_r)
lower_limit_r = r1-(1.5)*(IQR_r)
print(r1)
print(r3) print(IQR_r)
print(upper_limit_r)
print(lower_limit_r)
```

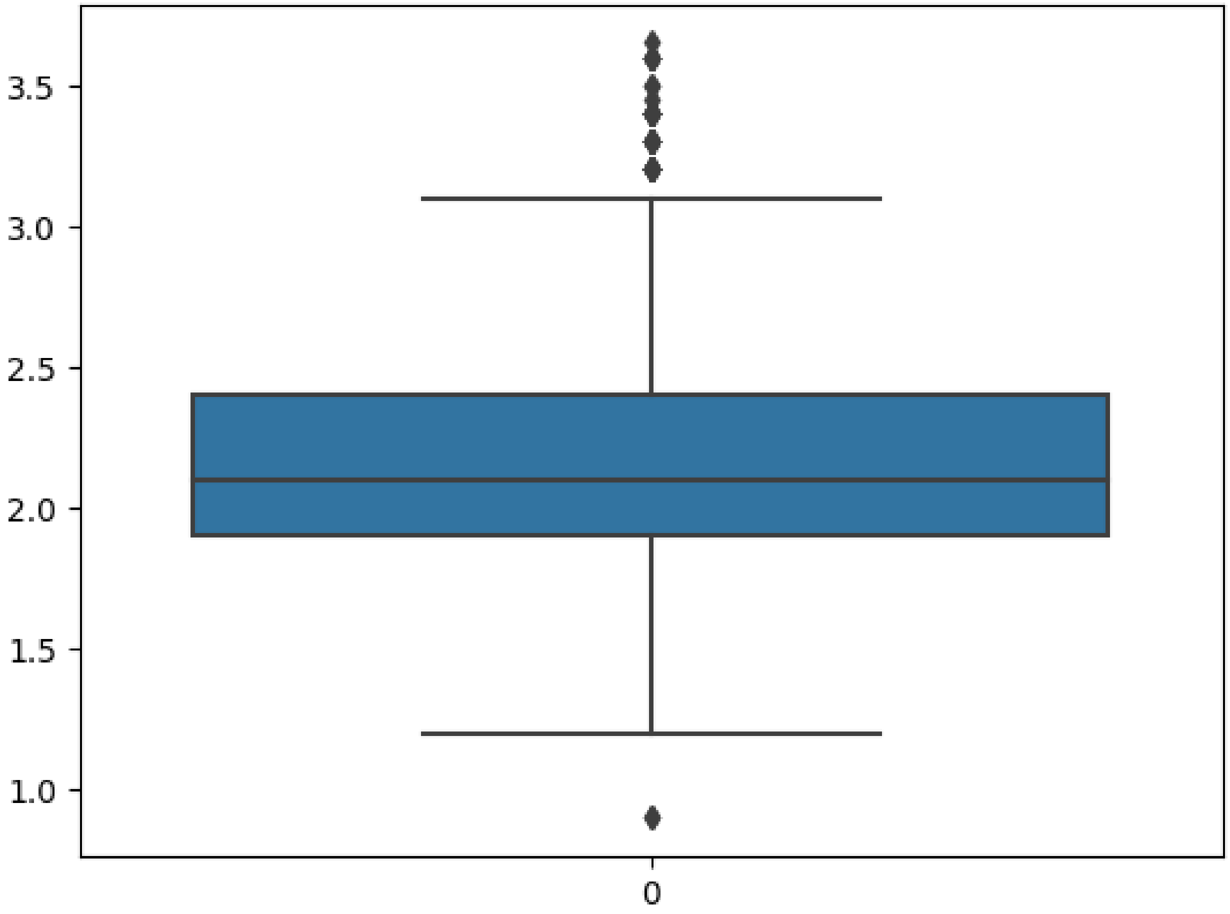
[26]:

```
1.9
2.6
0.7000000000000002
3.6500000000000004


0.8499999999999996
```

[27]:

[27]:

```
<Axes: >
```

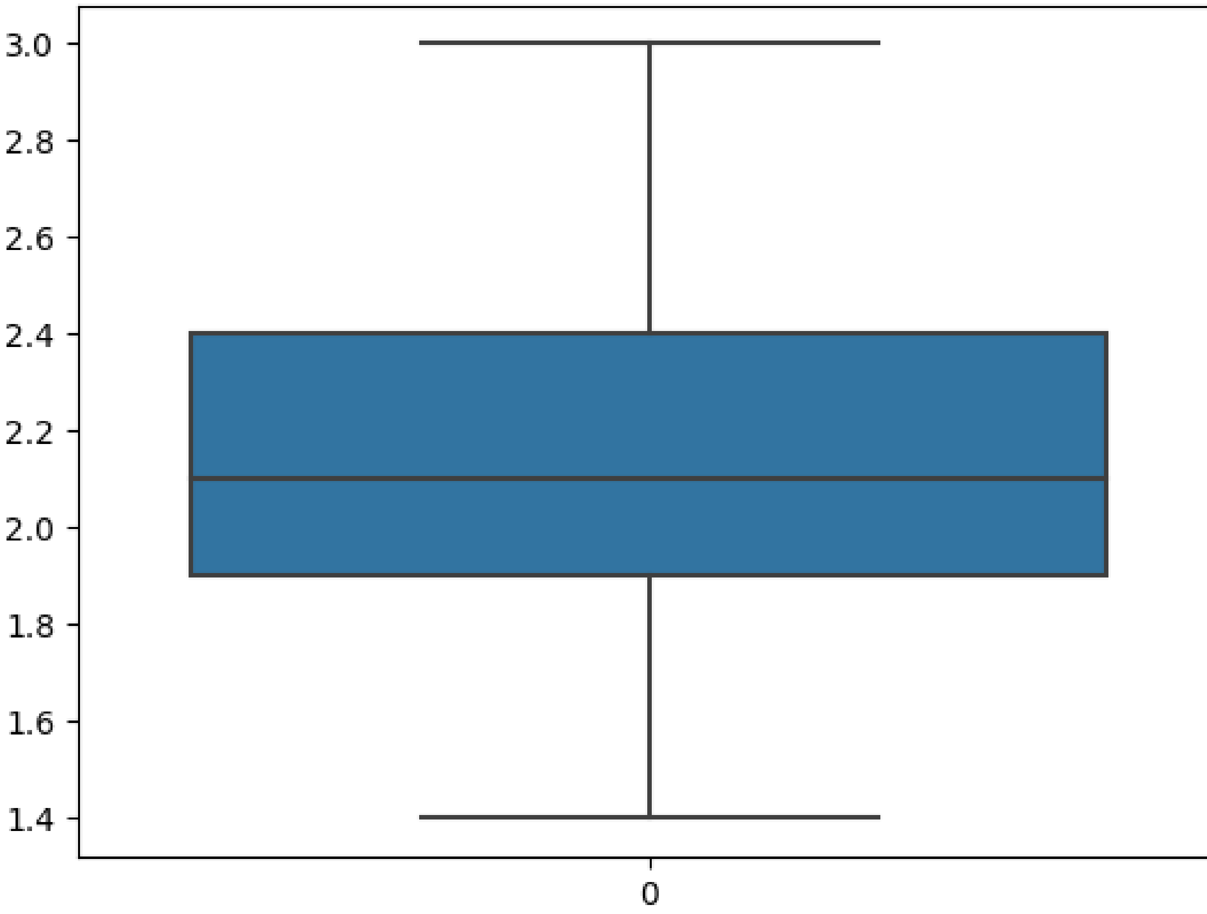


[34]:

```
1.4
3.0159999999999854
```

[35]:

[35]: <Axes: >



```
# Removing outliers from chlorides column

ch1 = df.chlorides.quantile(0.25) #Q1
ch3 = df.chlorides.quantile(0.75) #Q3
IQR_ch = ch3 - ch1
upper_limit_ch = ch3+(1.5)*(IQR_ch)
lower_limit_ch = ch1-(1.5)*(IQR_ch)
print(ch1)
print(ch3)
print(IQR_ch)
print(upper_limit_ch)
print(lower_limit_ch)
```

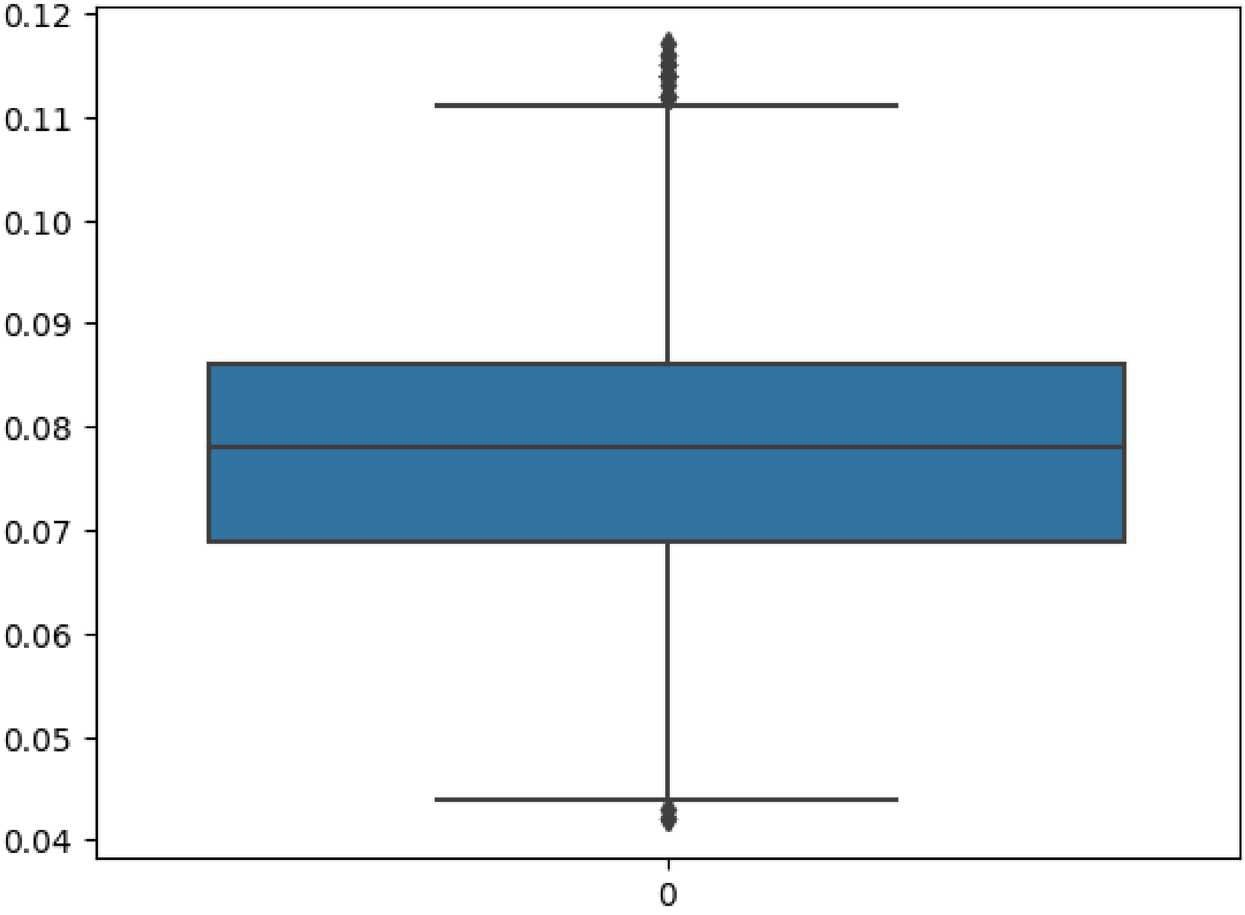
[36]:

[37]:

```
0.07
0.089
0.018999999999999999
0.11749999999999998
0.041500000000000002
```

[37]:

```
<Axes: >
```

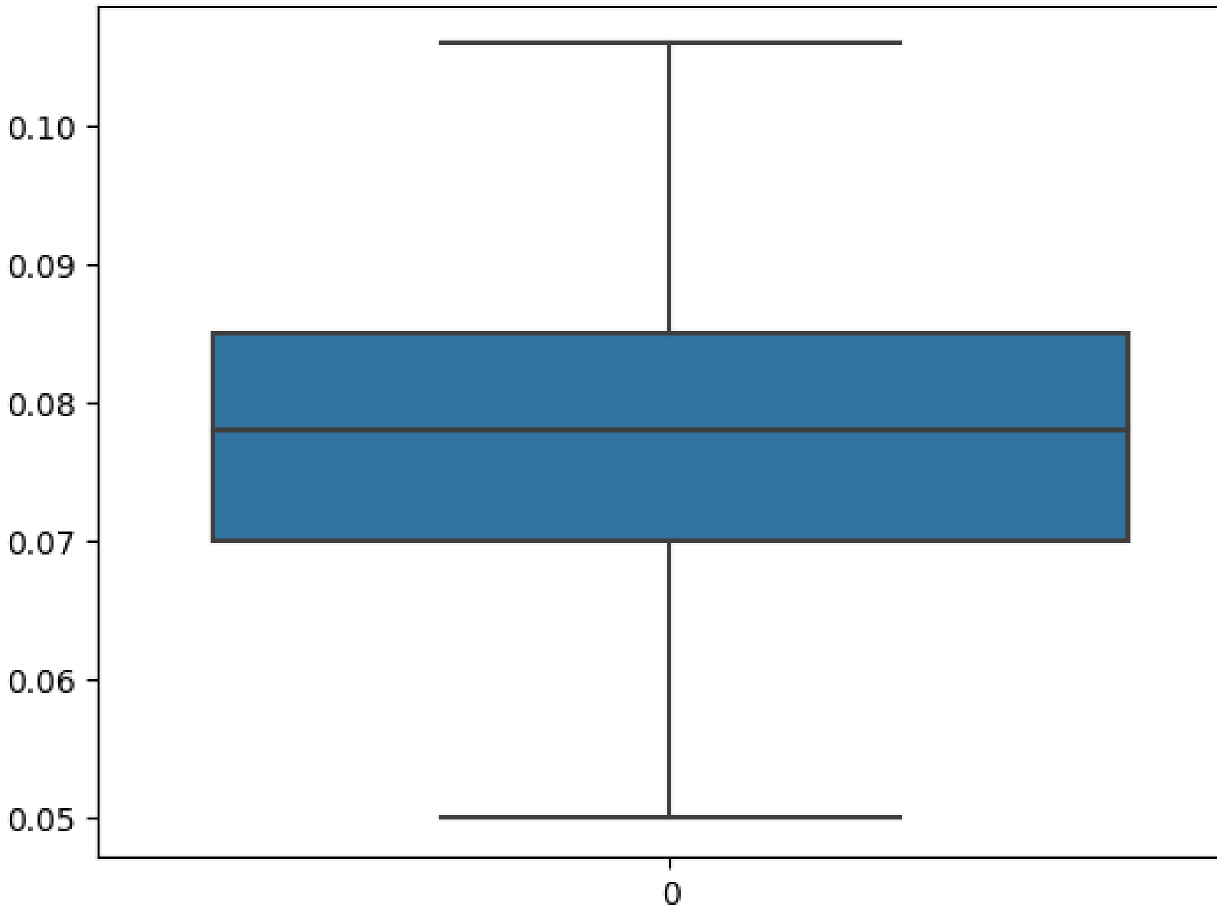


[44]:

```
0.04989000000000004
0.106
```

[45]:

[45]: <Axes: >



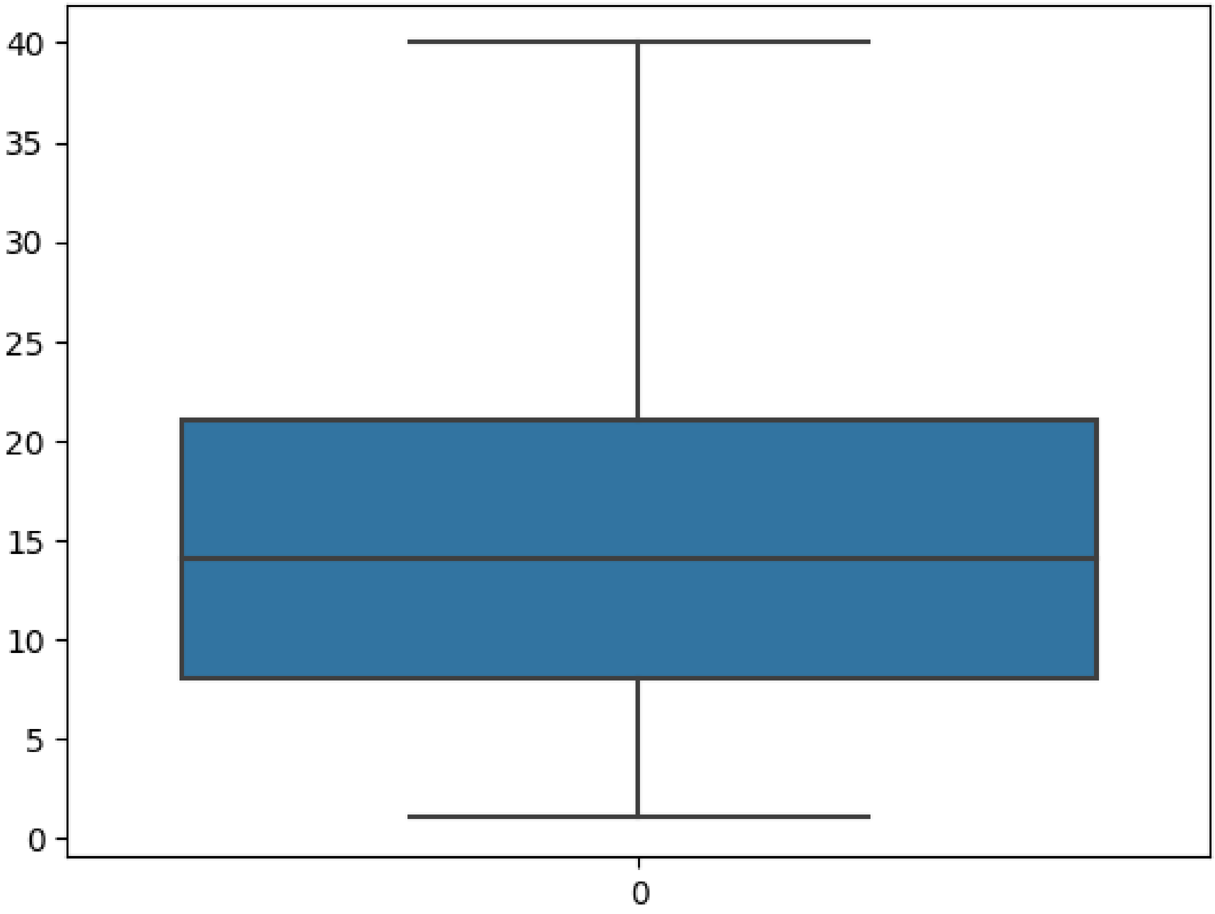
```
# Removing outliers from free_sulfur_dioxide column

fs1 = df.free_sulfur_dioxide.quantile(0.25) #Q1
fs3 = df.free_sulfur_dioxide.quantile(0.75) #Q3
IQR_fs = fs3 - fs1
upper_limit_fs = fs3+(1.5)*(IQR_fs)
lower_limit_fs = fs1-(1.5)*(IQR_fs)
print(fs1)
print(fs3) print(IQR_fs)
print(upper_limit_fs)
print(lower_limit_fs)
```

[52]:

[53]:

```
8.0
21.0
13.0
40.5
-11.5
```

[53]:
```
<Axes: >
```



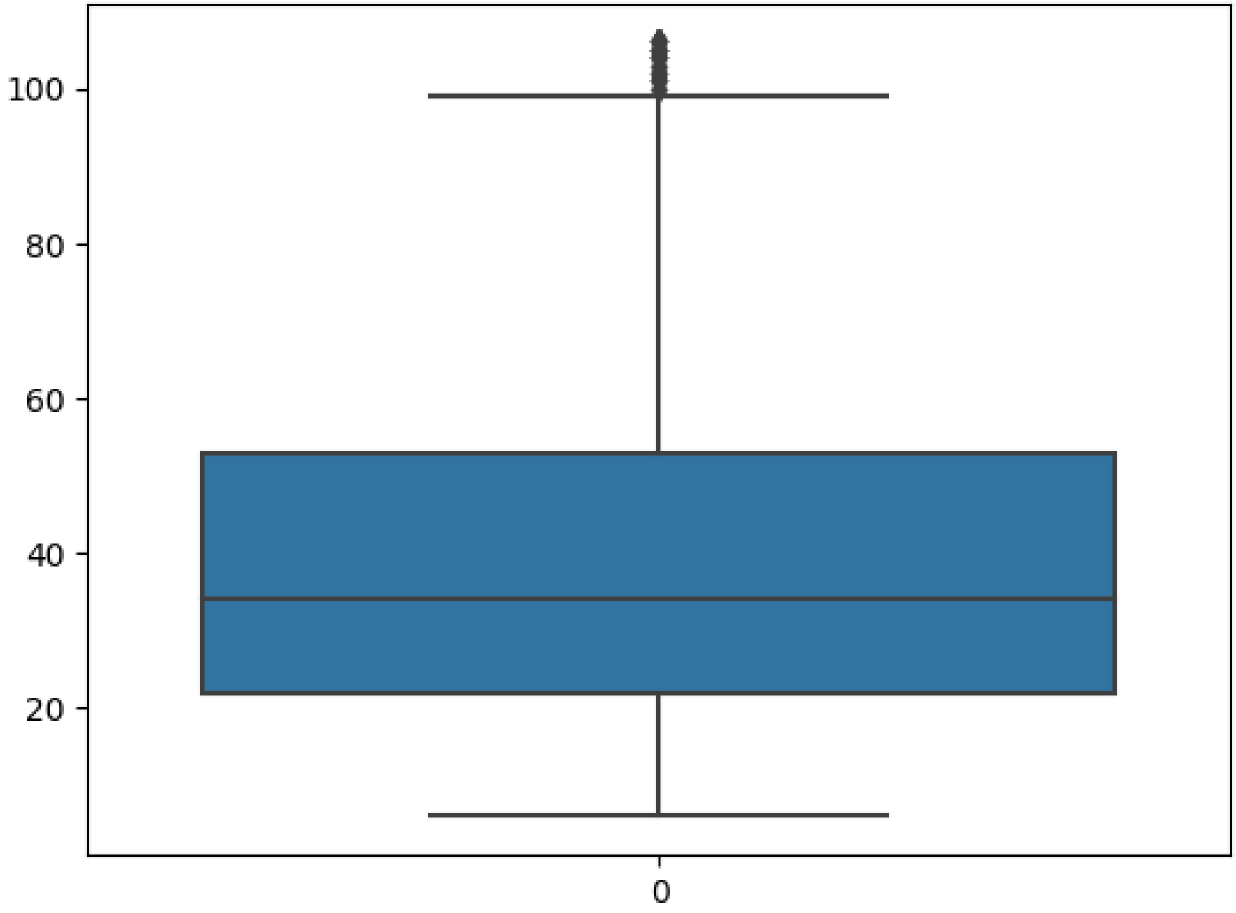
```
# Removing outliers from total_sulfur_dioxide column

ts1 = df.total_sulfur_dioxide.quantile(0.25) #Q1 ts3
= df.total_sulfur_dioxide.quantile(0.75) #Q3 IQR_ts
= ts3 - ts1
upper_limit_ts = ts3+(1.5)*(IQR_ts)
lower_limit_ts = ts1-(1.5)*(IQR_ts)
print(ts1)
print(ts3) print(IQR_ts)
print(upper_limit_ts)
print(lower_limit_ts)
  [54]:
```

```
23.0
57.0
34.0
108.0
-28.0
```

[55]:

[55]:

```
<Axes: >
```

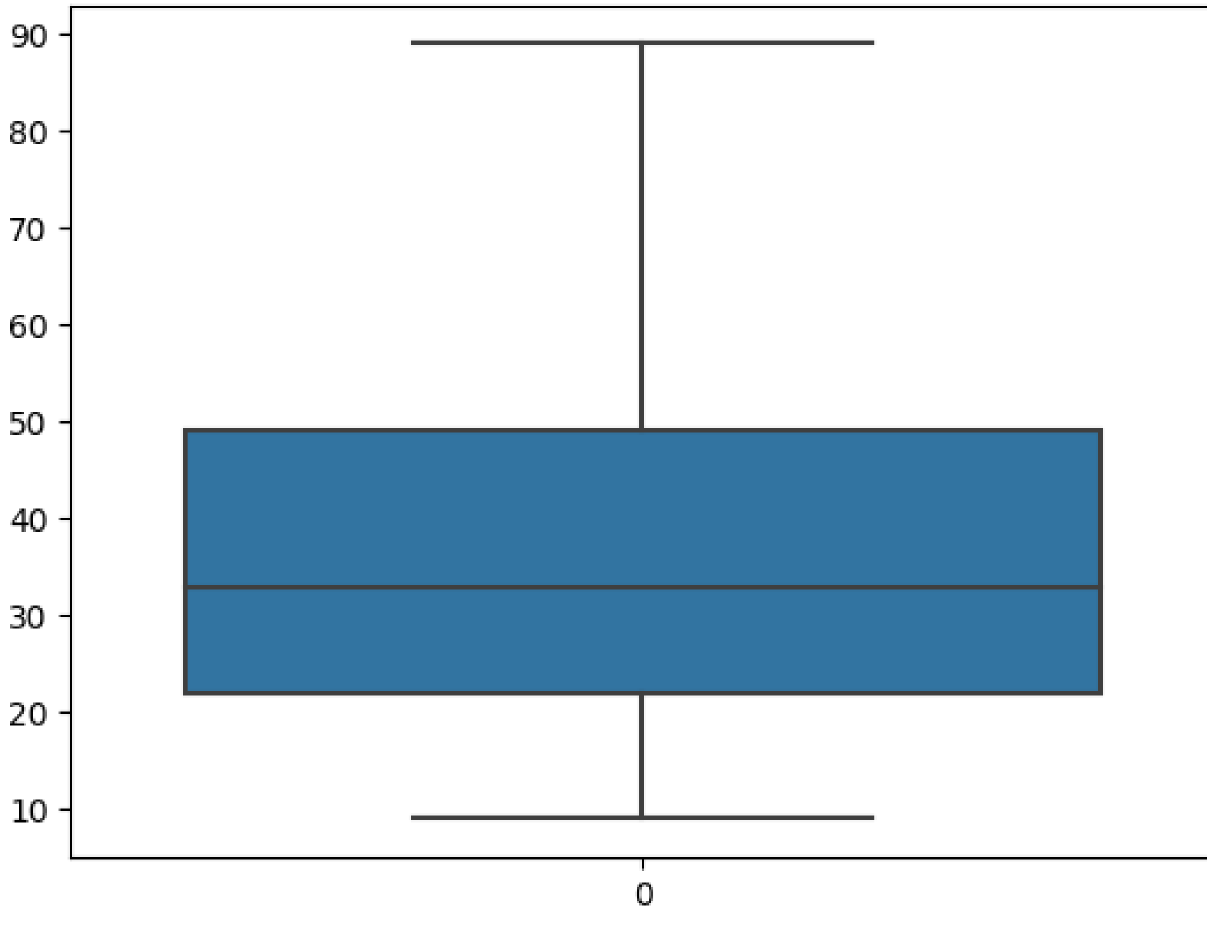


[60]:

```
9.0
89.0
```

[61]:

[61]: <Axes: >



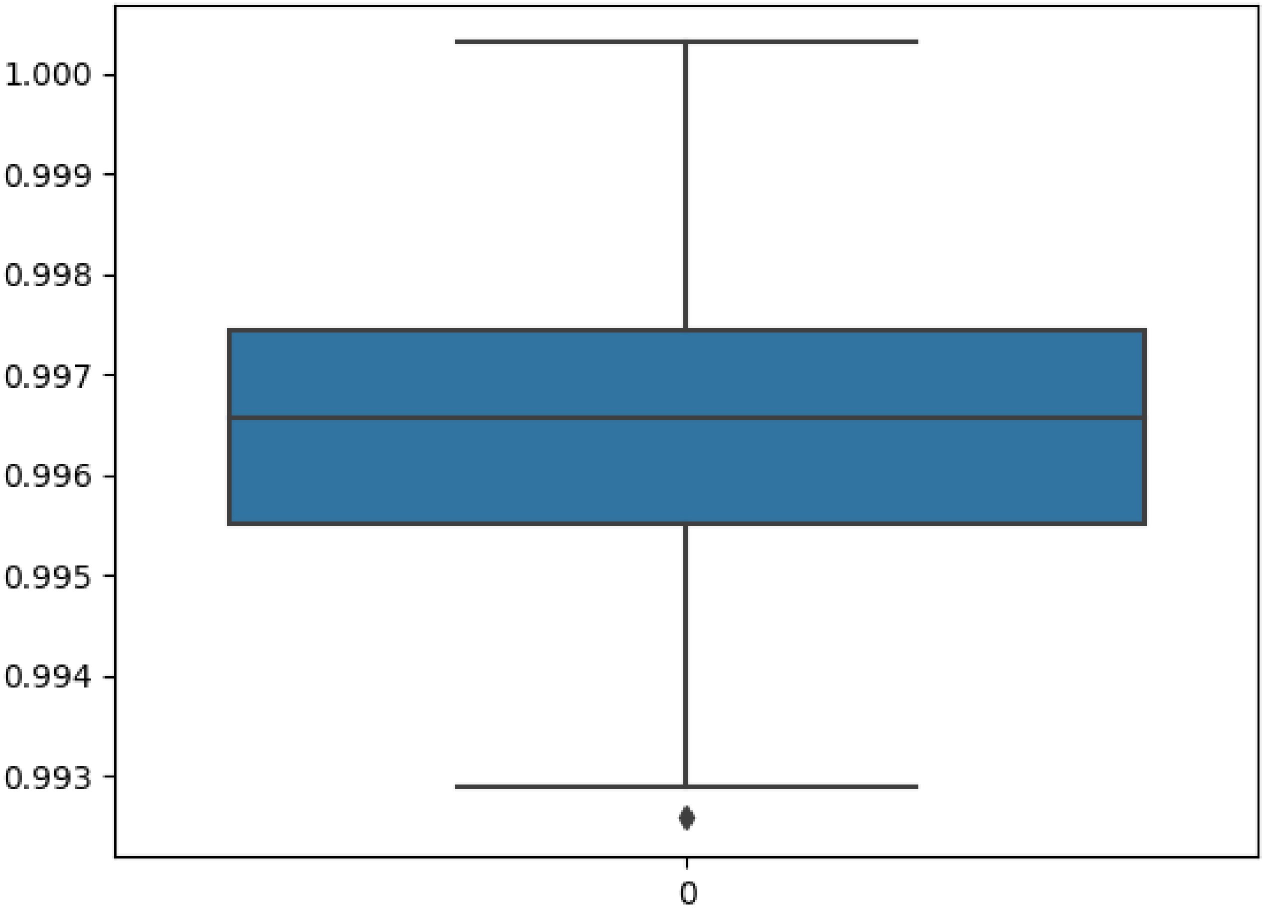
```
# Removing outliers from density column

d1 = df.density.quantile(0.25) #Q1
d3 = df.density.quantile(0.75) #Q3
IQR_d = d3 - d1
upper_limit_d = d3+(1.5)*(IQR_d)
lower_limit_d = d1-(1.5)*(IQR_d)
print(d1)
print(d3) print(IQR_d)
print(upper_limit_d)
print(lower_limit_d)
```

[62]:

[63]:

```
0.9955
0.99745
0.0019499999999998963
1.0003749999999998
0.9925750000000002
```

[63]: <Axes: >

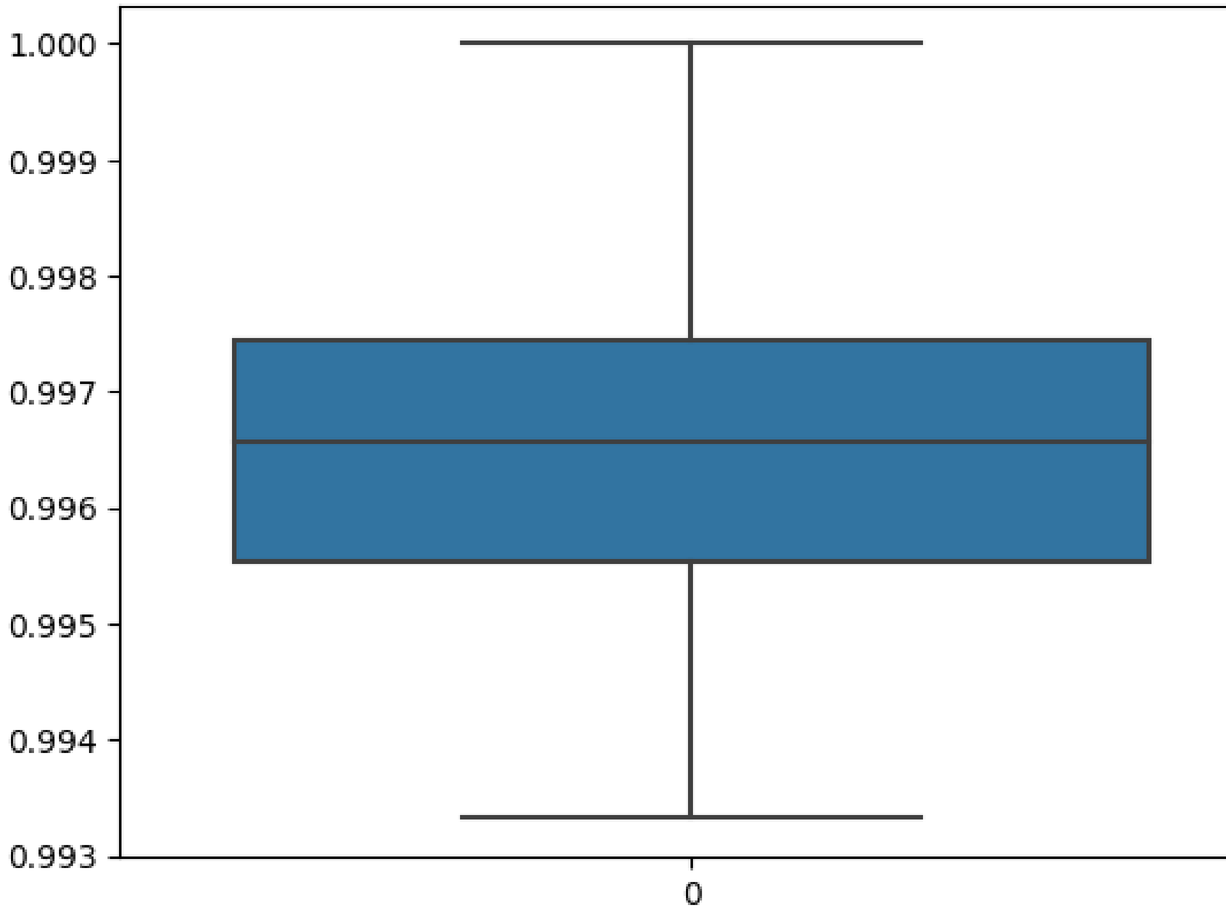


[64]:

```
0.9933132
1.0
```

[65]:

[65]: <Axes: >



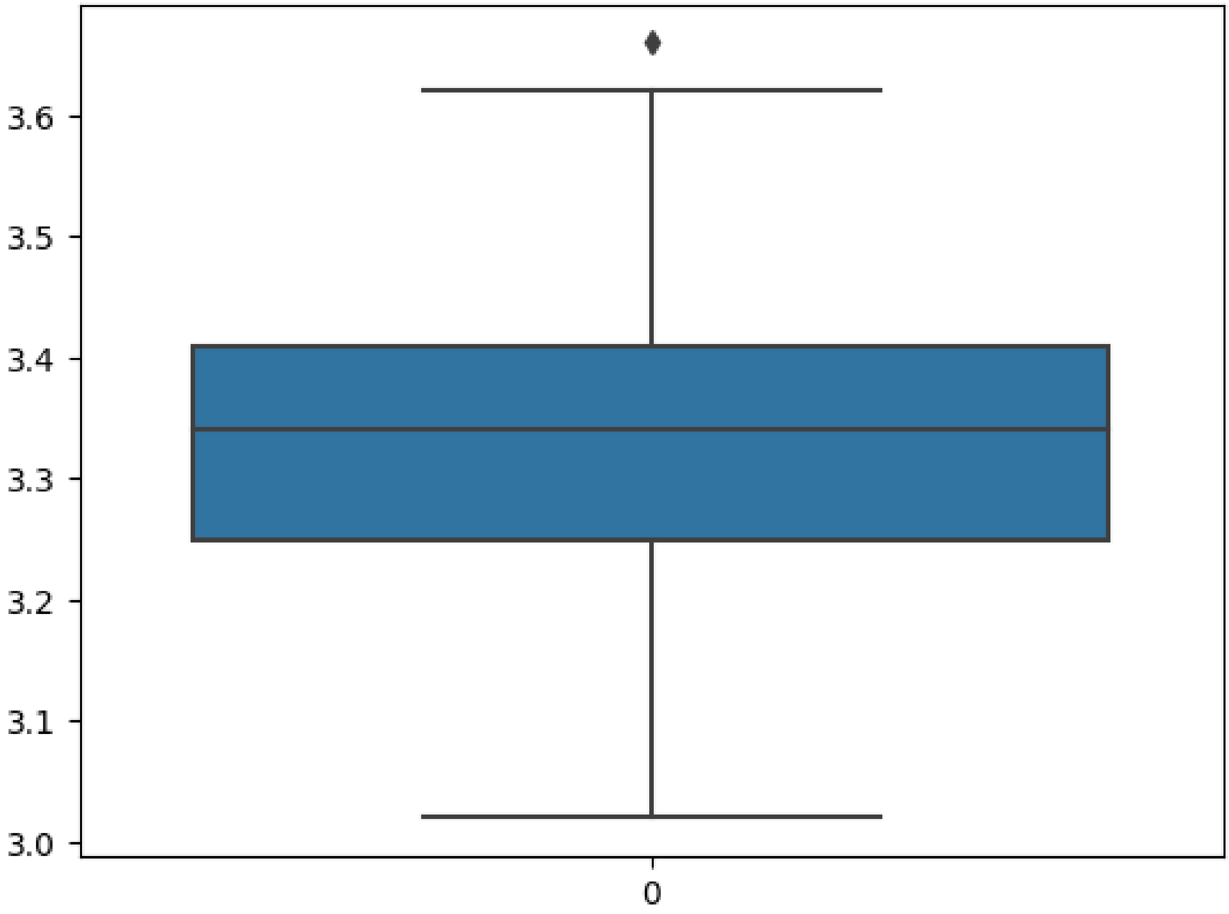
```
# Removing outliers from pH column

pH1 = df.pH.quantile(0.25) #Q1
pH3 = df.pH.quantile(0.75) #Q3
IQR_pH = pH3 - pH1
upper_limit_pH                    =
pH3+(1.5)*(IQR_pH)    lower_limit_pH
= pH1-(1.5)*(IQR_pH) print(pH1)
print(pH3)
print(IQR_pH)
print(upper_limit_pH)
print(lower_limit_pH)
```

[66]:

[67]:

```
3.2425
3.41
0.16749999999999998
3.66125
2.99125
```

[67]: <Axes: >

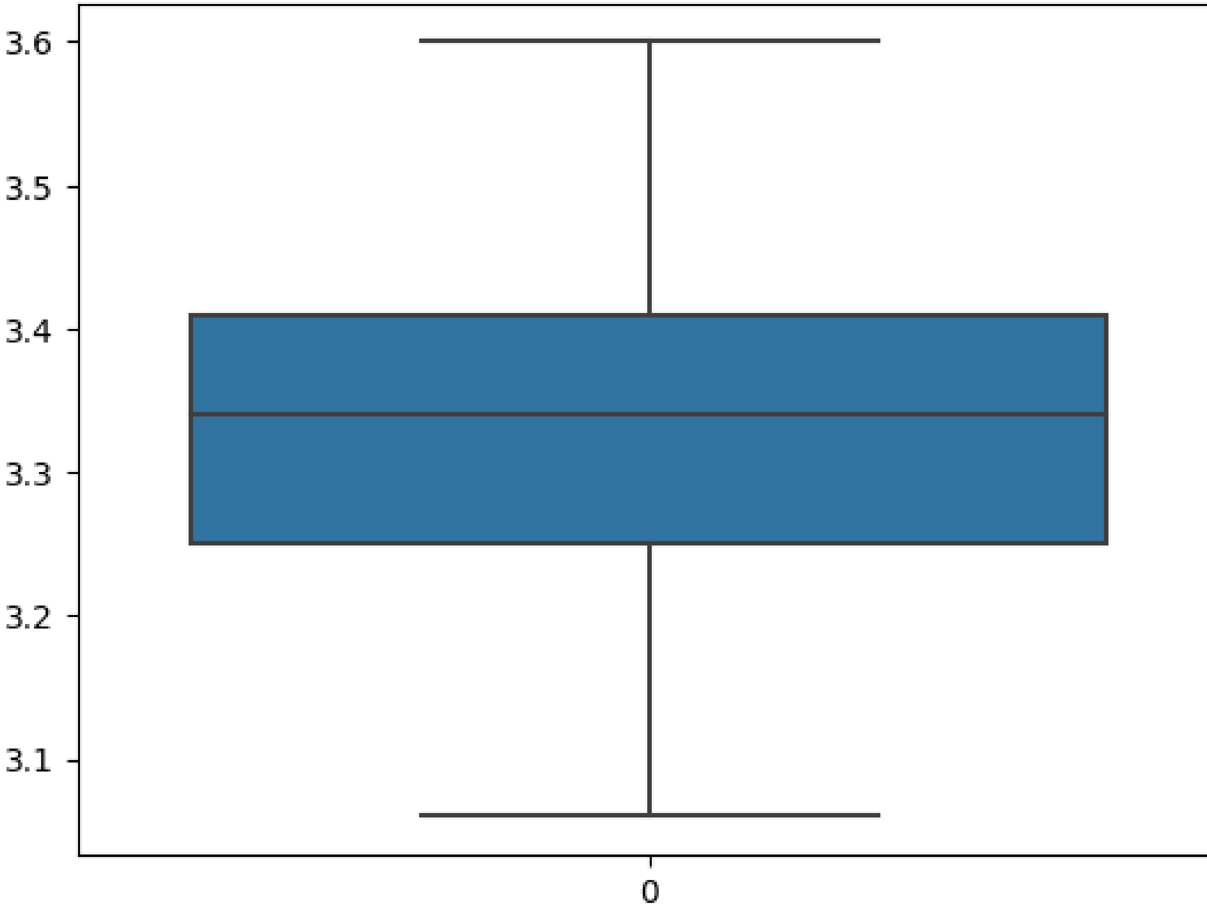


[68]:

```
3.06
3.6066
```

[69]:

[69]: <Axes: >

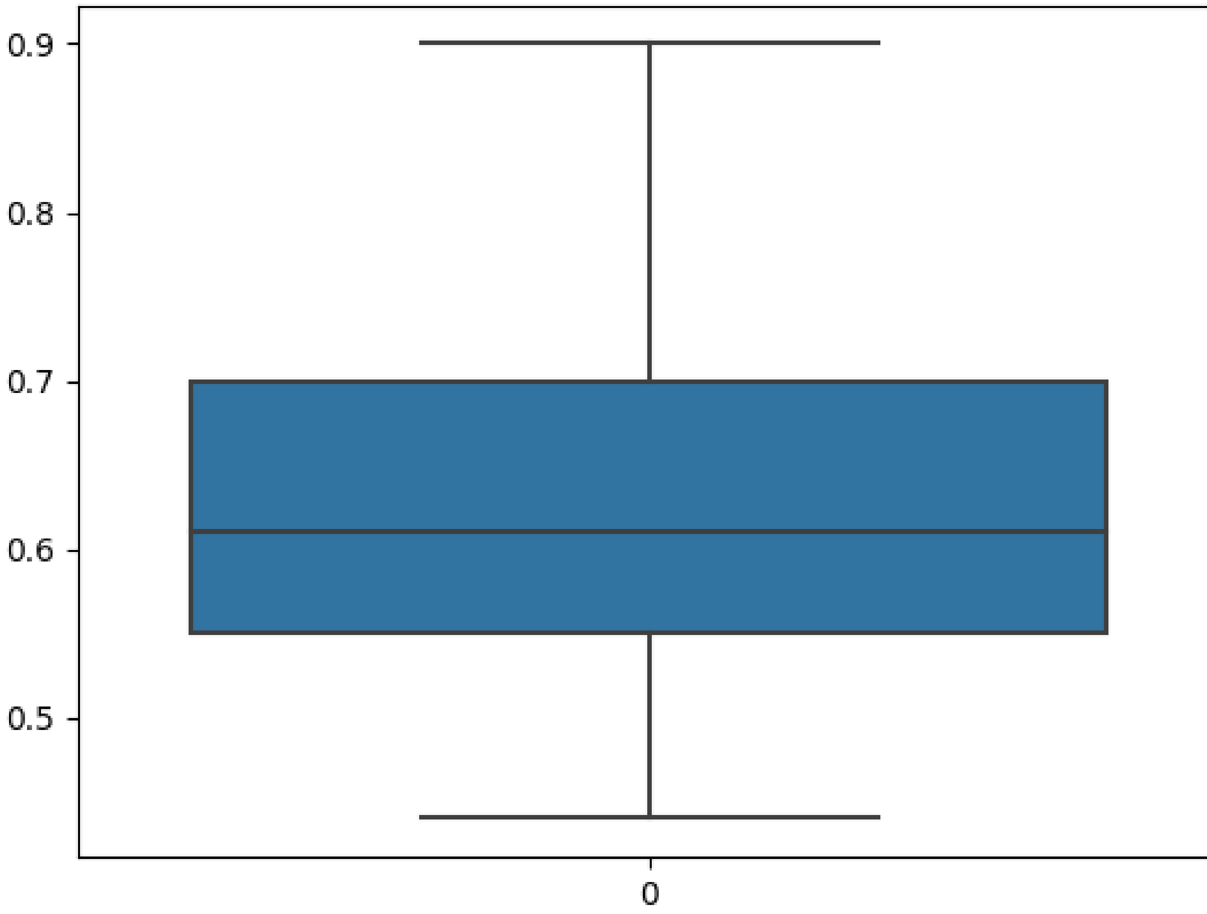


[74]:

```
0.44
0.9
```

[75]:

[75]: <Axes: >

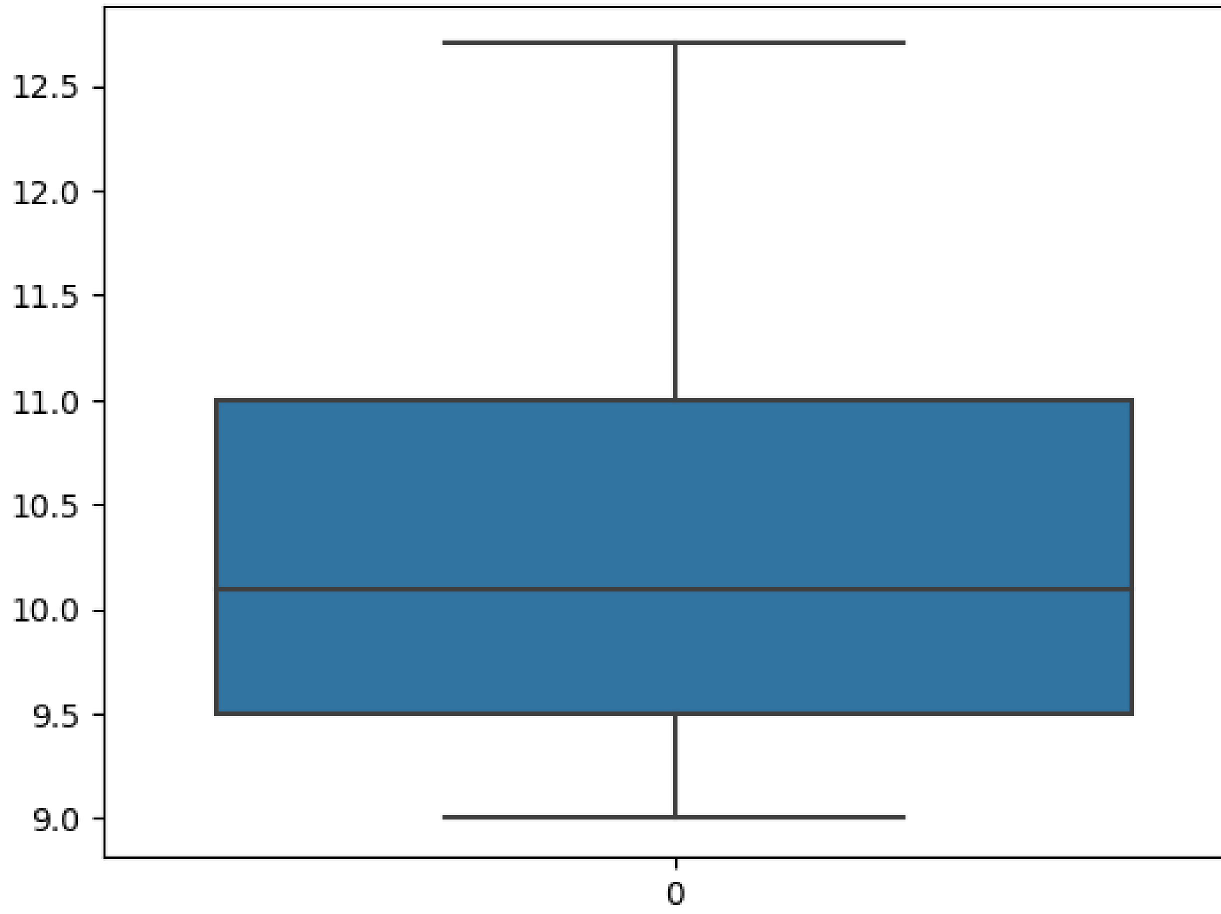


[76]:

```
9.0
12.724
```

[77]:

[77]: <Axes: >



[233]:

Therefore all the outliers are removed

- **Task - 3 : Machine Learning Model Building**

[233]:		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 0.997 8	3.51	0.56
1	25.0	67.0 0.996 8	3.20	0.68
2	15.0	54.0 0.997 0	3.26	0.65
3	17.0	60.0 0.998 0	3.16	0.58
4	11.0	34.0 0.997 8	3.51	0.56

[234]:

alcohol

```
    0
   9.
    4
    1
   9.
    8
    2
   9.
    8
    3
   9.
    8
    4
   9.
    4
```

[234]:

```
0          5
   1

   5
   2

   5
   3

   6
   4

   5
    Name:  quality,  dtype:  int64
```

Label Binarisation (Conidering alcohol quality > 7 as good and assigning ‘1’ to it else assigning‘0’)

[235]:

[236]:

[237]:

```
0       0
1       0
2       0
3       0
4       0
       ..
1593    0
1594    0
1595    0
1596    0
1597    0
Name: quality, Length: 866, dtype: int64
```

[238]:

```
[238]: (692, 11)
```

[239]:

```
[239]: (174, 11)
```

[240]:

[242]:

```
(866,) (692,) (174,)
```

- **Decision Tree Classifier**

```
[242]: DecisionTreeClassifier(criterion='entropy', max_depth=2)
```

[243]:

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

[245]:

- **Task - 4 : Evaluating the model (Decision tree classifier)**

[246]:

[247]:

```
Testing Accuracy =
                        0.8793103448275
862 Training Accuracy =
                        0.8916184971098
265
```

- **Random Forest Classifier**

[247]: RandomForestClassifier(criterion='entropy', n_estimators=200)

[248]:

- **Task - 4 : Evaluating Random Forest Model**

[249]:

```
[251]:
```

```
Testing Accuracy =
                          0.942528735632
1839Training Accuracy =          1.0
```

- **Naive Bayesian Classification Model**

```
[251]: GaussianNB()
```

```
[252]:
```

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

- **Task - 4 : Evaluating Naive Bayesian Classification Model**

```
[254]:
```

```
[254]: 0.8850574712643678
```

- **Accuracies of all the algorithms used in model nuilding phase :**

Decision Tree Classification : 87.93 %

- **Random Forset Classification : 94.25 %**

Naive Bayesian Classification : 88.50 %

[262]:

-
- **Conclusion : Random Forest Classifier Model is best suited for the wine qualitydataset.**
- **Task - 5 : Test with random observation**

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

[262]: array([0])

[263]:

According to “decision tree classifier” model, the above random observation gives prediction “array([0])” i.e., bad quality alcohol

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

[263]: array([0])

[264]:

According to “Random Forest classifier” model, the above random observation givesprediction “array([0])” i.e., bad quality alcohol

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439:
UserWarning: X doesnot have valid feature names, but GaussianNB was
fitted with feature names
  warnings.warn(
```

[264]:
```
array([0])
```

According to “Naive Bayesian classifier” model, the above random observation givesprediction “array([0])” i.e., bad quality alcohol

- **CONCLUSION : For the same random observation, all the three modelsgave the ”alchohol quality is BAD”**

1 The End !!!!