

K02-T1-IF2220-13520020

April 14, 2022

```
[ ]: import pandas as pd
import numpy
import scipy
from scipy import stats
import matplotlib.pyplot as plt
import math

#file csv awal ditambahkan dengan:
#id,pH,Hardness,Solids,Chloramines,Sulfate,Conductivity,OrganicCarbon,
#Trihalomethanes,Turbidity,Potability
#pada baris pertama file agar data lebih mudah diakses

dataset = pd.read_csv("water_potability.csv")
print(dataset)
```

	id	pH	Hardness	Solids	Chloramines	Sulfate	\
0	1	8.316766	214.373394	22018.417441	8.059332	356.886136	
1	2	9.092223	181.101509	17978.986339	6.546600	310.135738	
2	3	5.584087	188.313324	28748.687739	7.544869	326.678363	
3	4	10.223862	248.071735	28749.716544	7.513408	393.663396	
4	5	8.635849	203.361523	13672.091764	4.563009	303.309771	
...	
2005	2006	8.197353	203.105091	27701.794055	6.472914	328.886838	
2006	2007	8.989900	215.047358	15921.412018	6.297312	312.931022	
2007	2008	6.702547	207.321086	17246.920347	7.708117	304.510230	
2008	2009	11.491011	94.812545	37188.826022	9.263166	258.930600	
2009	2010	6.069616	186.659040	26138.780191	7.747547	345.700257	
	Conductivity	OrganicCarbon	Trihalomethanes	Turbidity	Potability		
0	363.266516	18.436524	100.341674	4.628771	0		
1	398.410813	11.558279	31.997993	4.075075	0		
2	280.467916	8.399735	54.917862	2.559708	0		
3	283.651634	13.789695	84.603556	2.672989	0		
4	474.607645	12.363817	62.798309	4.401425	0		
...		
2005	444.612724	14.250875	62.906205	3.361833	1		
2006	390.410231	9.899115	55.069304	4.613843	1		
2007	329.266002	16.217303	28.878601	3.442983	1		

2008	439.893618	16.172755	41.558501	4.369264	1
2009	415.886955	12.067620	60.419921	3.669712	1

[2010 rows x 11 columns]

```
[ ]: #Mengerjakan no 2 dan 3 untuk kolom data pH

pHMean = numpy.mean(dataset.pH)
pHMedian = numpy.median(dataset.pH)
pHModus = scipy.stats.mode(dataset.pH)
pHSTD = numpy.std(dataset.pH)
pHVariance = numpy.var(dataset.pH)
pHMin = numpy.min(dataset.pH)
pHMax = numpy.max(dataset.pH)
pHRange = pHMax-pHMin
pHQ1 = numpy.quantile(dataset.pH,0.25)
pHQ2 = numpy.quantile(dataset.pH,0.5)
pHQ3 = numpy.quantile(dataset.pH,0.75)
pHIQR = pHQ3 - pHQ1
pHSkew = scipy.stats.skew(dataset.pH)
pHKurtosis = scipy.stats.kurtosis(dataset.pH)

print("pH mean: " + str(pHMean))
print("pH median: " + str(pHMedian))
print("pH modus: " + str(pHModus))
print("pH standard deviation: " + str(pHSTD))
print("pH variance: " + str(pHVariance))
print("pH min: " + str(pHMin))
print("pH max: " + str(pHMax))
print("pH range: " + str(pHRange))
print("pH Q1: " + str(pHQ1))
print("pH Q2: " + str(pHQ2))
print("pH Q3: " + str(pHQ3))
print("pH IQR: " + str(pHIQR))
print("pH Skew: " + str(pHSkew))
print("pH Kurtosis: " + str(pHKurtosis))

temp, histogram_pH = plt.subplots(1,1)
histogram_pH.hist(dataset.pH)
histogram_pH.set_xlabel("pH")
histogram_pH.set_ylabel("amount")
plt.show();

temp, boxplot_pH = plt.subplots(1,1)
boxplot_pH.boxplot(dataset.pH)
boxplot_pH.set_ylabel("pH")
plt.show();
```

```

#harus melakukan pengecekan data skew dan kurtosis untuk melihat jika data
↳memiliki normal distribution
#nilai skew dan kurtosis harus mendekati 0
#(nilai kurtosis mengikuti definisi fisher sehingga hasil akhir kurtosis pada
↳hasil print kurtosis sudah dikurangi 3)
#(untuk menentukan normal distribution, skew memiliki besar maksimum 0.05 dan
↳kurtosis memiliki besar maksimum 0.3)

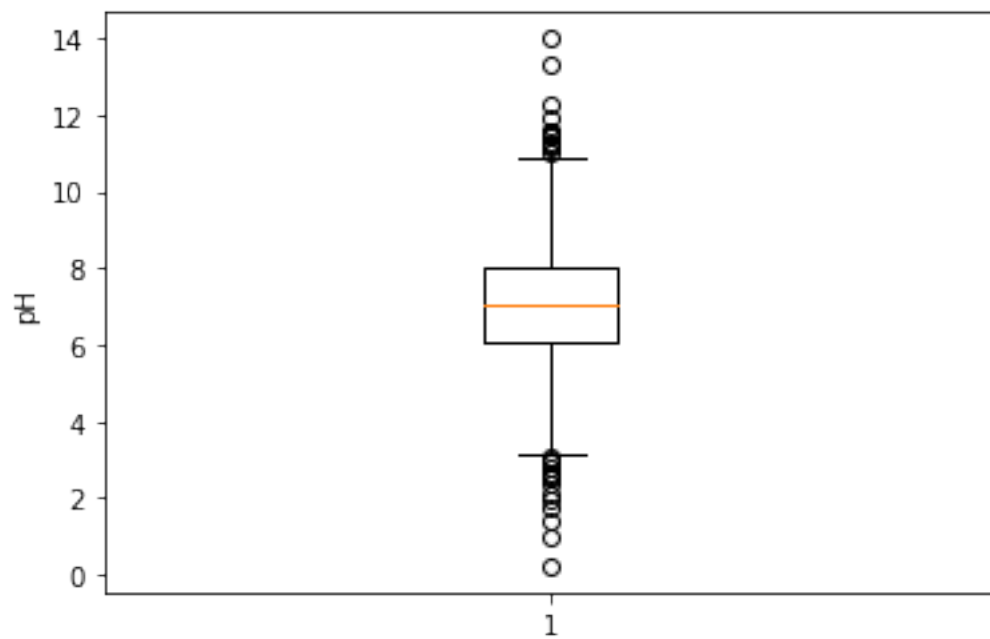
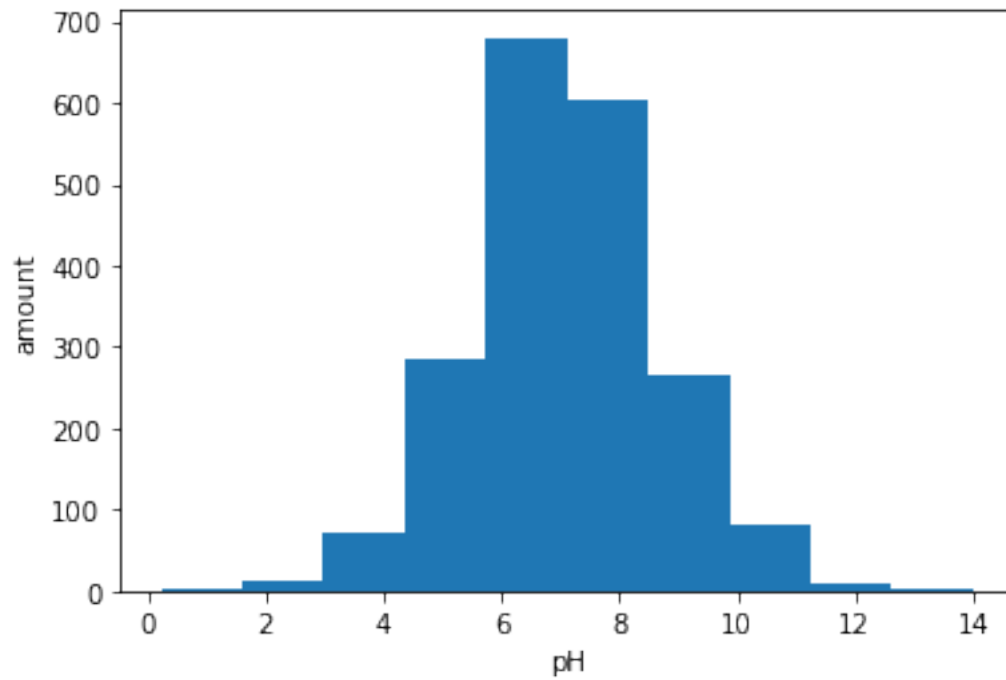
#data pH tidak memiliki distribusi normal karena memiliki nilai kurtosis yang
↳besar
#jika dilihat dari histogram, bagian tengah histogram jauh lebih tinggi
↳daripada bagian sekitar dan naik secara drastis sehingga tidak memiliki
↳distribusi normal

```

```

pH mean: 7.0871927687138205
pH median: 7.029490455474185
pH modus: ModeResult(mode=array([0.22749905]), count=array([1]))
pH standard deviation: 1.572411653857112
pH variance: 2.4724784091856584
pH min: 0.2274990502021987
pH max: 13.999999999999998
pH range: 13.7725009497978
pH Q1: 6.09078502142353
pH Q2: 7.029490455474185
pH Q3: 8.053006240791538
pH IQR: 1.9622212193680078
pH Skew: 0.048498286755236694
pH Kurtosis: 0.6223621582163492

```



```
[ ]: #Mengerjakan no 2 dan 3 untuk kolom data Hardness
```

```
HardnessMean = numpy.mean(dataset.Hardness)
```

```

HardnessMedian = numpy.median(dataset.Hardness)
HardnessModus = scipy.stats.mode(dataset.Hardness)
HardnessSTD = numpy.std(dataset.Hardness)
HardnessVariance = numpy.var(dataset.Hardness)
HardnessMin = numpy.min(dataset.Hardness)
HardnessMax = numpy.max(dataset.Hardness)
HardnessRange = HardnessMax-HardnessMin
HardnessQ1 = numpy.quantile(dataset.Hardness,0.25)
HardnessQ2 = numpy.quantile(dataset.Hardness,0.5)
HardnessQ3 = numpy.quantile(dataset.Hardness,0.75)
HardnessIQR = HardnessQ3 - HardnessQ1
HardnessSkew = scipy.stats.skew(dataset.Hardness)
HardnessKurtosis = scipy.stats.kurtosis(dataset.Hardness)

print("Hardness mean: " + str(HardnessMean))
print("Hardness median: " + str(HardnessMedian))
print("Hardness modus: " + str(HardnessModus))
print("Hardness standard deviation: " + str(HardnessSTD))
print("Hardness variance: " + str(HardnessVariance))
print("Hardness min: " + str(HardnessMin))
print("Hardness max: " + str(HardnessMax))
print("Hardness range: " + str(HardnessRange))
print("Hardness Q1: " + str(HardnessQ1))
print("Hardness Q2: " + str(HardnessQ2))
print("Hardness Q3: " + str(HardnessQ3))
print("Hardness IQR: " + str(HardnessIQR))
print("Hardness Skew: " + str(HardnessSkew))
print("Hardness Kurtosis: " + str(HardnessKurtosis))

temp, histogram_Hardness = plt.subplots(1,1)
histogram_Hardness.hist(dataset.Hardness)
histogram_Hardness.set_xlabel("Hardness")
histogram_Hardness.set_ylabel("amount")
plt.show();

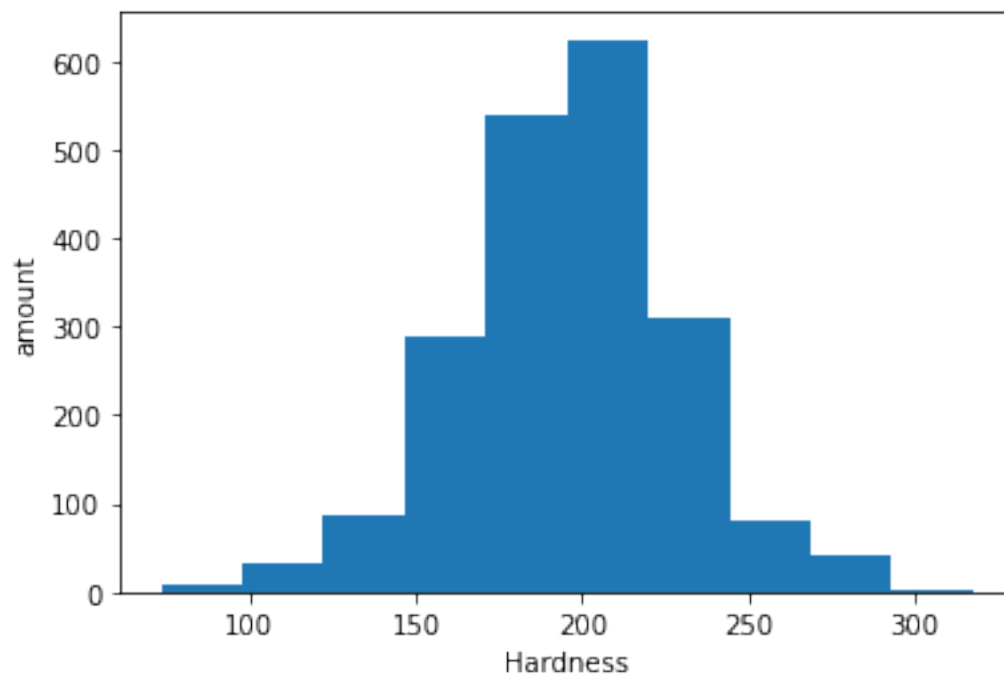
temp, boxplot_Hardness = plt.subplots(1,1)
boxplot_Hardness.boxplot(dataset.Hardness)
boxplot_Hardness.set_ylabel("Hardness")
plt.show();

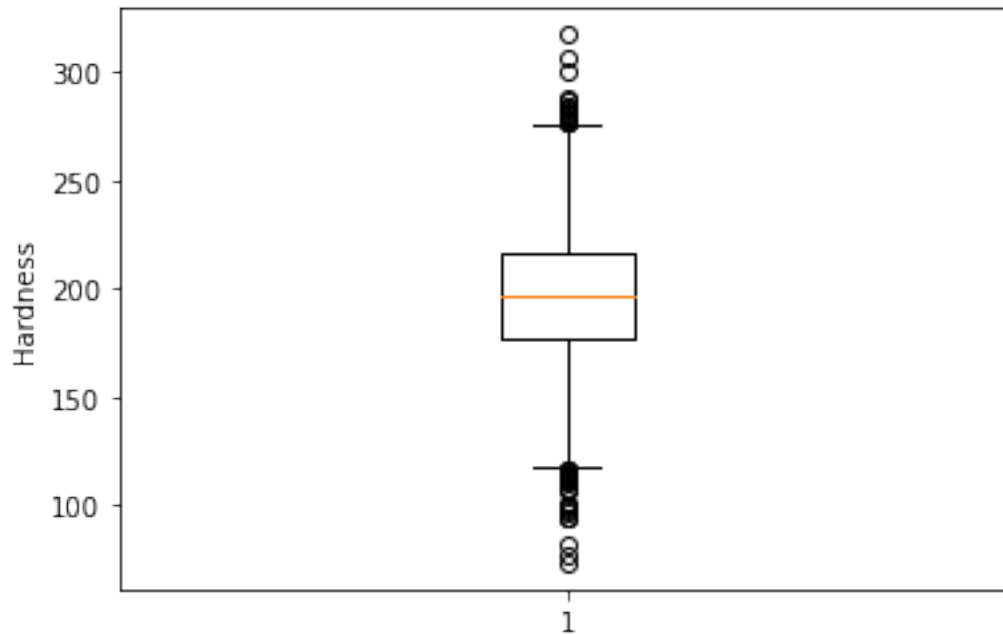
#data Hardness tidak memiliki distribusi normal karena memiliki nilai kurtosis
→yang besar
#jika dilihat dari histogram, bagian tengah histogram jauh lebih tinggi
→daripada bagian sekitar dan naik secara drastis sehingga tidak memiliki
→distribusi normal

```

Hardness mean: 195.96920903783553

Hardness median: 197.20352491941043
Hardness modus: ModeResult(mode=array([73.49223369]), count=array([1]))
Hardness standard deviation: 32.63504465869229
Hardness variance: 1065.0461398748398
Hardness min: 73.4922336890611
Hardness max: 317.33812405558257
Hardness range: 243.84589036652147
Hardness Q1: 176.74065667669896
Hardness Q2: 197.20352491941043
Hardness Q3: 216.44758866727156
Hardness IQR: 39.7069319905726
Hardness Skew: -0.0852573561057953
Hardness Kurtosis: 0.5211906487769769





[]: *#Mengerjakan no 2 dan 3 untuk kolom data Solids*

```
SolidsMean = numpy.mean(dataset.Solids)
SolidsMedian = numpy.median(dataset.Solids)
SolidsModus = scipy.stats.mode(dataset.Solids)
SolidsSTD = numpy.std(dataset.Solids)
SolidsVariance = numpy.var(dataset.Solids)
SolidsMin = numpy.min(dataset.Solids)
SolidsMax = numpy.max(dataset.Solids)
SolidsRange = SolidsMax-SolidsMin
SolidsQ1 = numpy.quantile(dataset.Solids,0.25)
SolidsQ2 = numpy.quantile(dataset.Solids,0.5)
SolidsQ3 = numpy.quantile(dataset.Solids,0.75)
SolidsIQR = SolidsQ3 - SolidsQ1
SolidsSkew = scipy.stats.skew(dataset.Solids)
SolidsKurtosis = scipy.stats.kurtosis(dataset.Solids)

print("Solids mean: " + str(SolidsMean))
print("Solids median: " + str(SolidsMedian))
print("Solids modus: " + str(SolidsModus))
print("Solids standard deviation: " + str(SolidsSTD))
print("Solids variance: " + str(SolidsVariance))
print("Solids min: " + str(SolidsMin))
print("Solids max: " + str(SolidsMax))
print("Solids range: " + str(SolidsRange))
print("Solids Q1: " + str(SolidsQ1))
```

```

print("Solids Q2: " + str(SolidsQ2))
print("Solids Q3: " + str(SolidsQ3))
print("Solids IQR: " + str(SolidsIQR))
print("Solids Skew: " + str(SolidsSkew))
print("Solids Kurtosis: " + str(SolidsKurtosis))

temp, histogram_Solids = plt.subplots(1,1)
histogram_Solids.hist(dataset.Solids)
histogram_Solids.set_xlabel("Solids")
histogram_Solids.set_ylabel("amount")
plt.show();

temp, boxplot_Solids = plt.subplots(1,1)
boxplot_Solids.boxplot(dataset.Solids)
boxplot_Solids.set_ylabel("Solids")
plt.show();

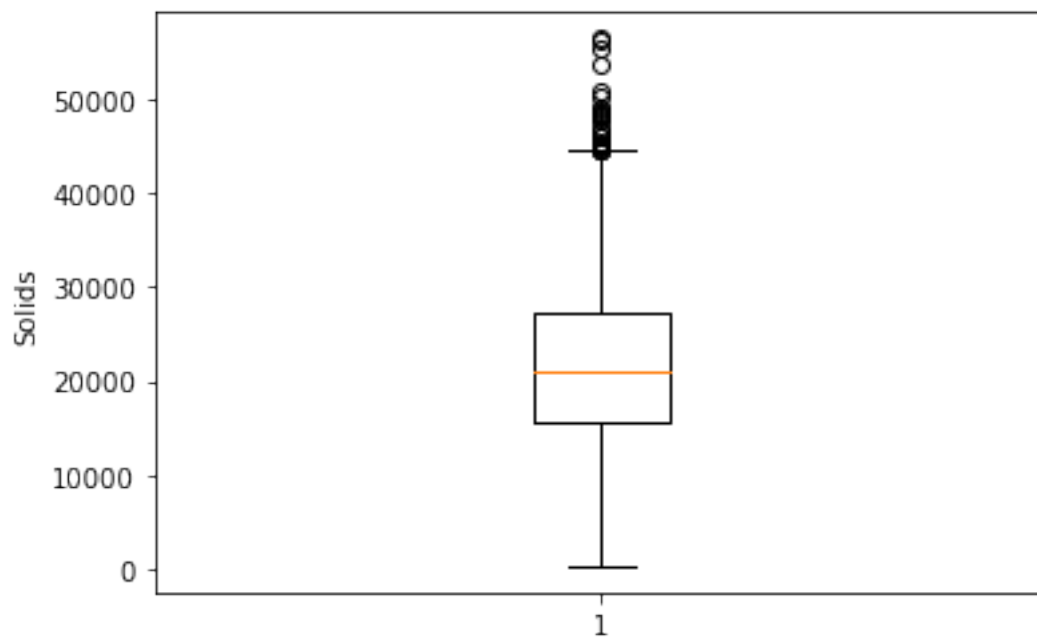
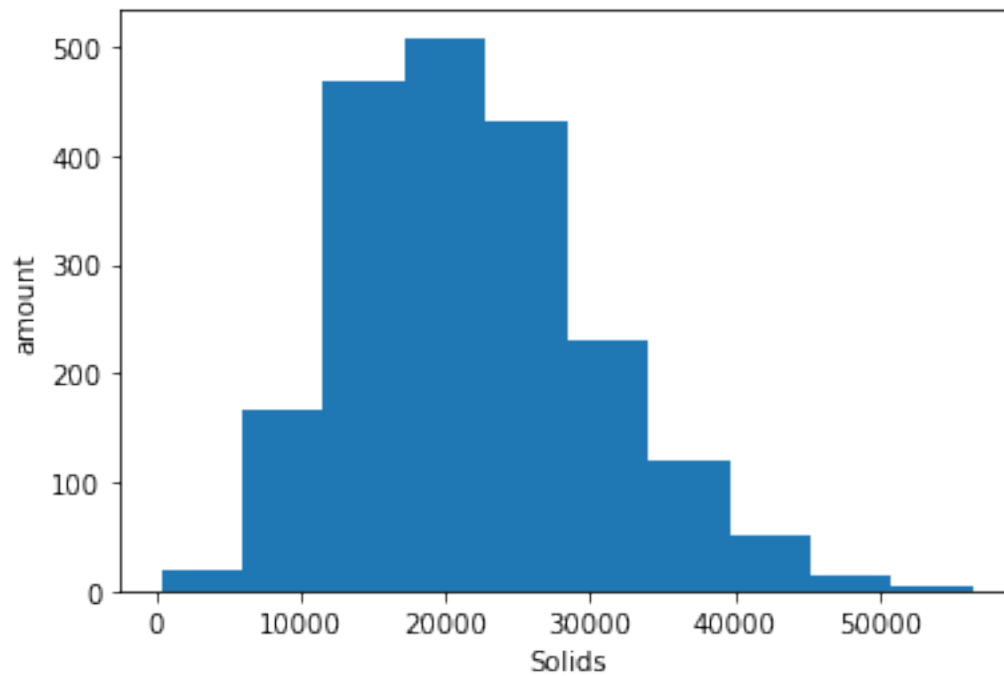
#data Solids tidak memiliki distribusi normal karena memiliki nilai skew yang
↳ besar
#jika dilihat dari histogram, bagian paling tinggi histogram berada di bagian
↳ kiri gambar dan tidak di tengah sehingga tidak memiliki distribusi normal

```

```

Solids mean: 21904.67343905309
Solids median: 20926.88215534375
Solids modus: ModeResult(mode=array([320.94261127]), count=array([1]))
Solids standard deviation: 8623.2520228849
Solids variance: 74360475.45018855
Solids min: 320.942611274359
Solids max: 56488.67241273919
Solids range: 56167.72980146483
Solids Q1: 15614.412961614333
Solids Q2: 20926.88215534375
Solids Q3: 27170.534648603603
Solids IQR: 11556.12168698927
Solids Skew: 0.5905702277342111
Solids Kurtosis: 0.333498156306705

```

```
[ ]: #Mengerjakan no 2 dan 3 untuk kolom data Chloramines
```

```
ChloraminesMean = numpy.mean(dataset.Chloramines)
```

```

ChloraminesMedian = numpy.median(dataset.Chloramines)
ChloraminesModus = scipy.stats.mode(dataset.Chloramines)
ChloraminesSTD = numpy.std(dataset.Chloramines)
ChloraminesVariance = numpy.var(dataset.Chloramines)
ChloraminesMin = numpy.min(dataset.Chloramines)
ChloraminesMax = numpy.max(dataset.Chloramines)
ChloraminesRange = ChloraminesMax-ChloraminesMin
ChloraminesQ1 = numpy.quantile(dataset.Chloramines,0.25)
ChloraminesQ2 = numpy.quantile(dataset.Chloramines,0.5)
ChloraminesQ3 = numpy.quantile(dataset.Chloramines,0.75)
ChloraminesIQR = ChloraminesQ3 - ChloraminesQ1
ChloraminesSkew = scipy.stats.skew(dataset.Chloramines)
ChloraminesKurtosis = scipy.stats.kurtosis(dataset.Chloramines)

print("Chloramines mean: " + str(ChloraminesMean))
print("Chloramines median: " + str(ChloraminesMedian))
print("Chloramines modus: " + str(ChloraminesModus))
print("Chloramines standard deviation: " + str(ChloraminesSTD))
print("Chloramines variance: " + str(ChloraminesVariance))
print("Chloramines min: " + str(ChloraminesMin))
print("Chloramines max: " + str(ChloraminesMax))
print("Chloramines range: " + str(ChloraminesRange))
print("Chloramines Q1: " + str(ChloraminesQ1))
print("Chloramines Q2: " + str(ChloraminesQ2))
print("Chloramines Q3: " + str(ChloraminesQ3))
print("Chloramines IQR: " + str(ChloraminesIQR))
print("Chloramines Skew: " + str(ChloraminesSkew))
print("Chloramines Kurtosis: " + str(ChloraminesKurtosis))

temp, histogram_Chloramines = plt.subplots(1,1)
histogram_Chloramines.hist(dataset.Chloramines)
histogram_Chloramines.set_xlabel("Chloramines")
histogram_Chloramines.set_ylabel("amount")
plt.show();

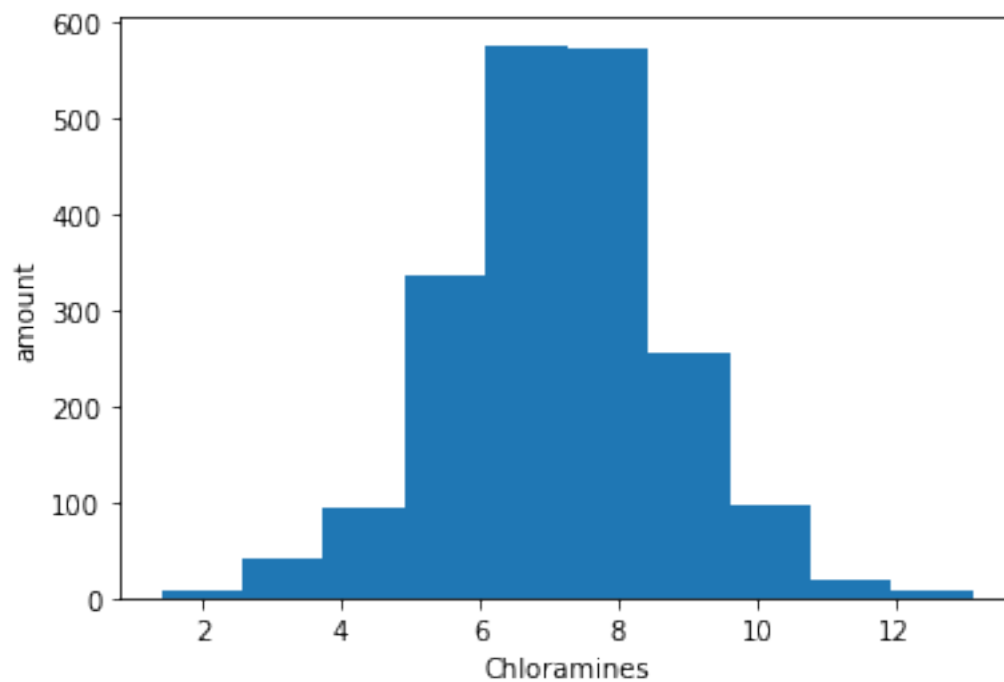
temp, boxplot_Chloramines = plt.subplots(1,1)
boxplot_Chloramines.boxplot(dataset.Chloramines)
boxplot_Chloramines.set_ylabel("Chloramines")
plt.show();

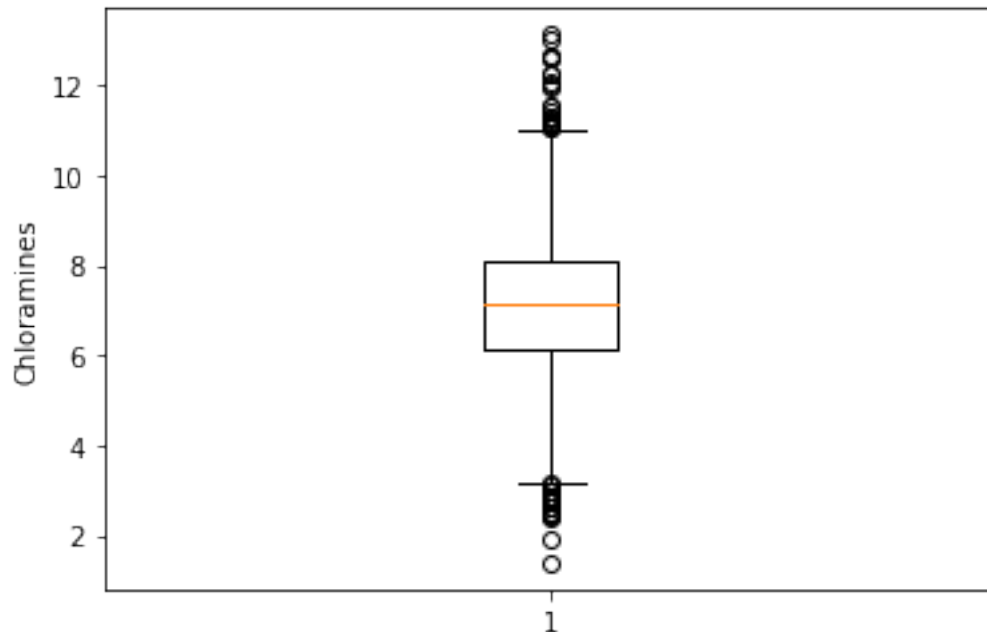
#data Chloramines tidak memiliki distribusi normal karena memiliki nilai
↳kurtosis yang besar
#jika dilihat dari histogram, bagian tengah histogram jauh lebih tinggi
↳daripada bagian sekitar dan naik secara drastis sehingga tidak memiliki
↳distribusi normal

```

Chloramines mean: 7.134322344600092

Chloramines median: 7.1420143046226645
Chloramines modus: ModeResult(mode=array([1.3908709]), count=array([1]))
Chloramines standard deviation: 1.5848197173405134
Chloramines variance: 2.511653536471265
Chloramines min: 1.3908709048851806
Chloramines max: 13.127000000000002
Chloramines range: 11.736129095114823
Chloramines Q1: 6.138326387572855
Chloramines Q2: 7.1420143046226645
Chloramines Q3: 8.109933216133502
Chloramines IQR: 1.9716068285606472
Chloramines Skew: 0.012993791664418169
Chloramines Kurtosis: 0.5454318545555785





```
[ ]: #Mengerjakan no 2 dan 3 untuk kolom data Sulfate

SulfateMean = numpy.mean(dataset.Sulfate)
SulfateMedian = numpy.median(dataset.Sulfate)
SulfateModus = scipy.stats.mode(dataset.Sulfate)
SulfateSTD = numpy.std(dataset.Sulfate)
SulfateVariance = numpy.var(dataset.Sulfate)
SulfateMin = numpy.min(dataset.Sulfate)
SulfateMax = numpy.max(dataset.Sulfate)
SulfateRange = SulfateMax-SulfateMin
SulfateQ1 = numpy.quantile(dataset.Sulfate,0.25)
SulfateQ2 = numpy.quantile(dataset.Sulfate,0.5)
SulfateQ3 = numpy.quantile(dataset.Sulfate,0.75)
SulfateIQR = SulfateQ3 - SulfateQ1
SulfateSkew = scipy.stats.skew(dataset.Sulfate)
SulfateKurtosis = scipy.stats.kurtosis(dataset.Sulfate)

print("Sulfate mean: " + str(SulfateMean))
print("Sulfate median: " + str(SulfateMedian))
print("Sulfate modus: " + str(SulfateModus))
print("Sulfate standard deviation: " + str(SulfateSTD))
print("Sulfate variance: " + str(SulfateVariance))
print("Sulfate min: " + str(SulfateMin))
print("Sulfate max: " + str(SulfateMax))
print("Sulfate range: " + str(SulfateRange))
print("Sulfate Q1: " + str(SulfateQ1))
```

```

print("Sulfate Q2: " + str(SulfateQ2))
print("Sulfate Q3: " + str(SulfateQ3))
print("Sulfate IQR: " + str(SulfateIQR))
print("Sulfate Skew: " + str(SulfateSkew))
print("Sulfate Kurtosis: " + str(SulfateKurtosis))

temp, histogram_Sulfate = plt.subplots(1,1)
histogram_Sulfate.hist(dataset.Sulfate)
histogram_Sulfate.set_xlabel("Sulfate")
histogram_Sulfate.set_ylabel("amount")
plt.show();

temp, boxplot_Sulfate = plt.subplots(1,1)
boxplot_Sulfate.boxplot(dataset.Sulfate)
boxplot_Sulfate.set_ylabel("Sulfate")
plt.show();

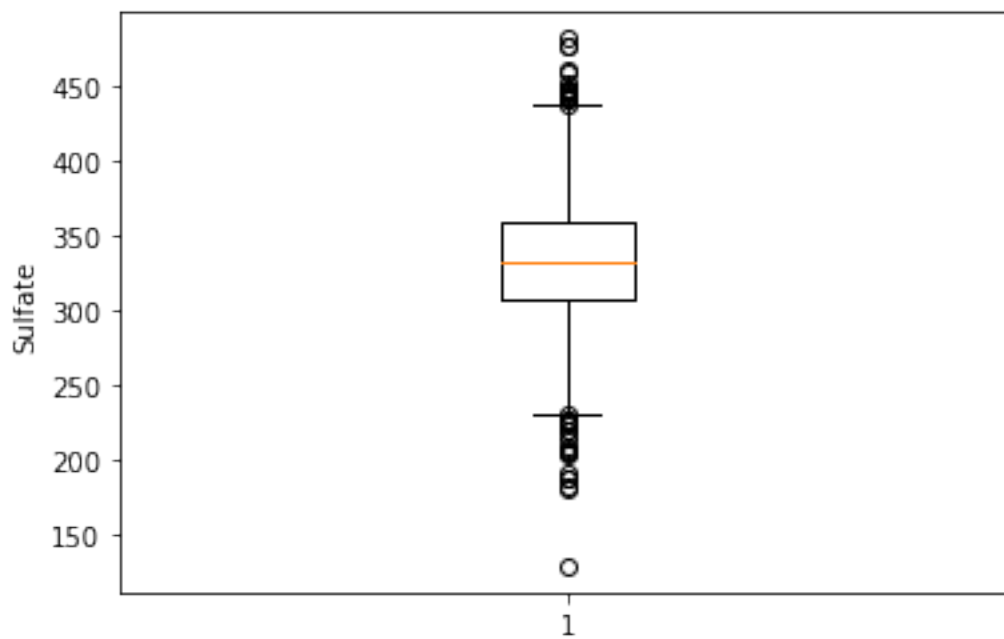
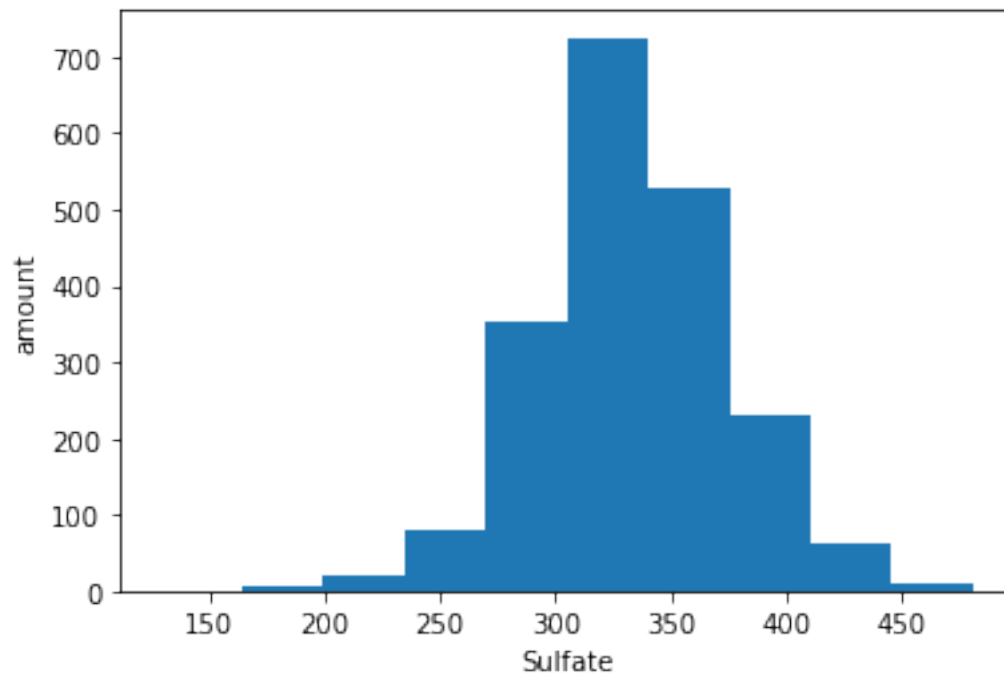
#data Sulfate tidak memiliki distribusi normal karena memiliki nilai kurtosis
↳ yang besar
#jika dilihat dari histogram, bagian tengah histogram jauh lebih tinggi
↳ daripada bagian sekitar
#bentuk histogram juga tidak terlihat simetris

```

```

Sulfate mean: 333.21137641518925
Sulfate median: 332.2141128069568
Sulfate modus: ModeResult(mode=array([129.]), count=array([1]))
Sulfate standard deviation: 41.20085823007217
Sulfate variance: 1697.5107188945055
Sulfate min: 129.00000000000003
Sulfate max: 481.0306423059972
Sulfate range: 352.03064230599716
Sulfate Q1: 307.6269864860709
Sulfate Q2: 332.2141128069568
Sulfate Q3: 359.26814739141554
Sulfate IQR: 51.641160905344634
Sulfate Skew: -0.04569367213282734
Sulfate Kurtosis: 0.7819149219038866

```



```
[ ]: #Mengerjakan no 2 dan 3 untuk kolom data Conductivity
```

```
ConductivityMean = numpy.mean(dataset.Conductivity)
```

```

ConductivityMedian = numpy.median(dataset.Conductivity)
ConductivityModus = scipy.stats.mode(dataset.Conductivity)
ConductivitySTD = numpy.std(dataset.Conductivity)
ConductivityVariance = numpy.var(dataset.Conductivity)
ConductivityMin = numpy.min(dataset.Conductivity)
ConductivityMax = numpy.max(dataset.Conductivity)
ConductivityRange = ConductivityMax-ConductivityMin
ConductivityQ1 = numpy.quantile(dataset.Conductivity,0.25)
ConductivityQ2 = numpy.quantile(dataset.Conductivity,0.5)
ConductivityQ3 = numpy.quantile(dataset.Conductivity,0.75)
ConductivityIQR = ConductivityQ3 - ConductivityQ1
ConductivitySkew = scipy.stats.skew(dataset.Conductivity)
ConductivityKurtosis = scipy.stats.kurtosis(dataset.Conductivity)

print("Conductivity mean: " + str(ConductivityMean))
print("Conductivity median: " + str(ConductivityMedian))
print("Conductivity modus: " + str(ConductivityModus))
print("Conductivity standard deviation: " + str(ConductivitySTD))
print("Conductivity variance: " + str(ConductivityVariance))
print("Conductivity min: " + str(ConductivityMin))
print("Conductivity max: " + str(ConductivityMax))
print("Conductivity range: " + str(ConductivityRange))
print("Conductivity Q1: " + str(ConductivityQ1))
print("Conductivity Q2: " + str(ConductivityQ2))
print("Conductivity Q3: " + str(ConductivityQ3))
print("Conductivity IQR: " + str(ConductivityIQR))
print("Conductivity Skew: " + str(ConductivitySkew))
print("Conductivity Kurtosis: " + str(ConductivityKurtosis))

temp, histogram_Conductivity = plt.subplots(1,1)
histogram_Conductivity.hist(dataset.Conductivity)
histogram_Conductivity.set_xlabel("Conductivity")
histogram_Conductivity.set_ylabel("amount")
plt.show();

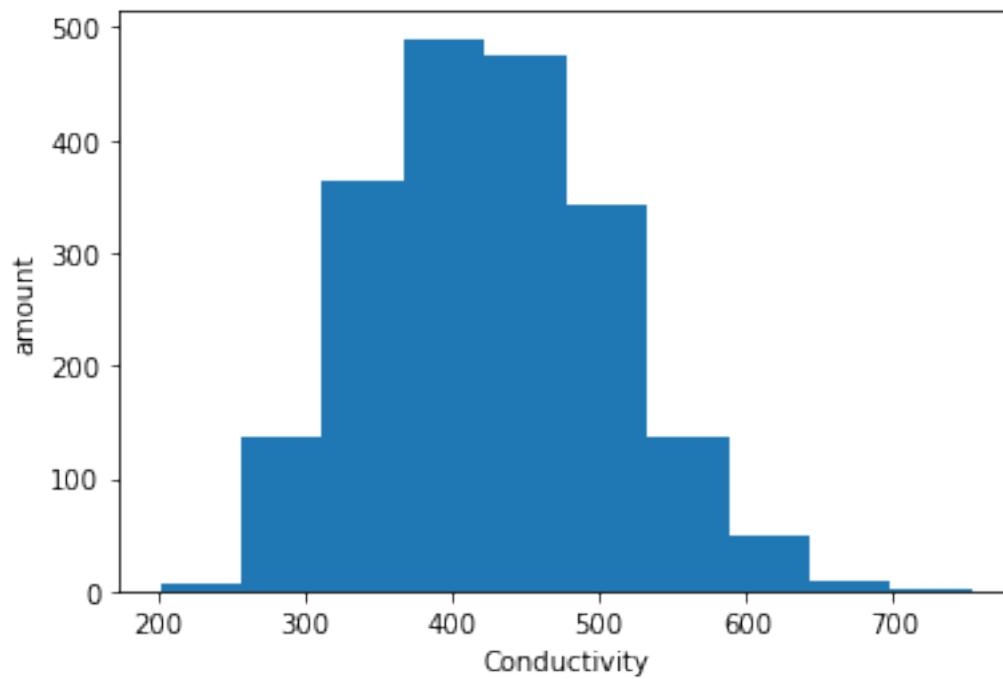
temp, boxplot_Conductivity = plt.subplots(1,1)
boxplot_Conductivity.boxplot(dataset.Conductivity)
boxplot_Conductivity.set_ylabel("Conductivity")
plt.show();

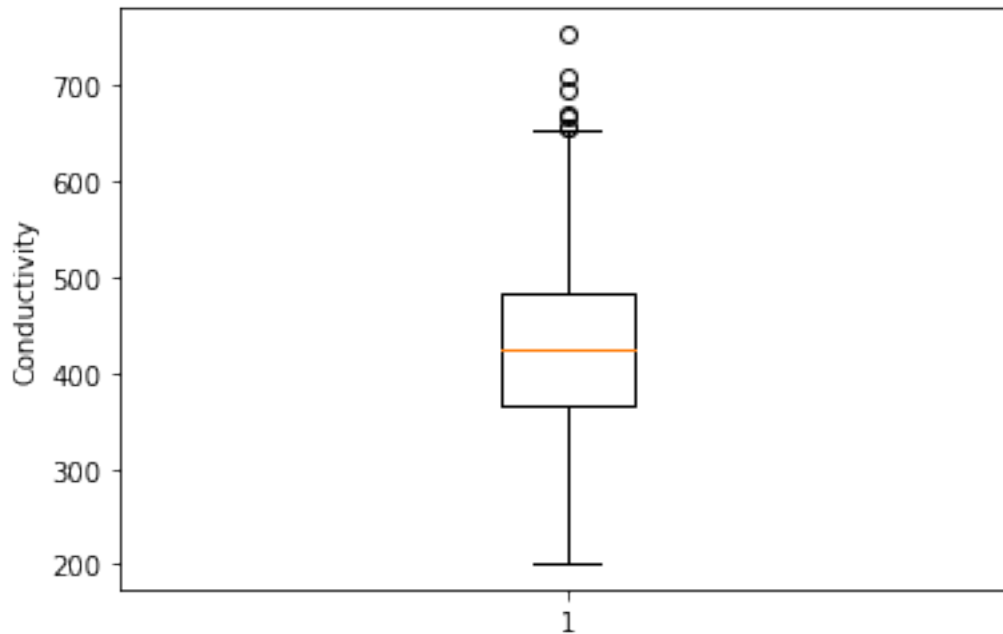
#data Conductivity tidak memiliki distribusi normal karena memiliki nilai skew
→yang besar
#jika dilihat dari histogram, bagian paling tinggi histogram berada di bagian
→kiri gambar dan tidak di tengah sehingga tidak memiliki distribusi normal

```

Conductivity mean: 426.4767083525792
Conductivity median: 423.43837202443706

Conductivity modus: ModeResult(mode=array([201.61973676]), count=array([1]))
Conductivity standard deviation: 80.68179421728163
Conductivity variance: 6509.55191811978
Conductivity min: 201.6197367551575
Conductivity max: 753.3426195583046
Conductivity range: 551.7228828031471
Conductivity Q1: 366.61921929632433
Conductivity Q2: 423.43837202443706
Conductivity Q3: 482.2097724598859
Conductivity IQR: 115.5905531635616
Conductivity Skew: 0.26781228234697935
Conductivity Kurtosis: -0.2395999421551358





[]: *#Mengerjakan no 2 dan 3 untuk kolom data OrganicCarbon*

```
OrganicCarbonMean = numpy.mean(dataset.OrganicCarbon)
OrganicCarbonMedian = numpy.median(dataset.OrganicCarbon)
OrganicCarbonModus = scipy.stats.mode(dataset.OrganicCarbon)
OrganicCarbonSTD = numpy.std(dataset.OrganicCarbon)
OrganicCarbonVariance = numpy.var(dataset.OrganicCarbon)
OrganicCarbonMin = numpy.min(dataset.OrganicCarbon)
OrganicCarbonMax = numpy.max(dataset.OrganicCarbon)
OrganicCarbonRange = OrganicCarbonMax - OrganicCarbonMin
OrganicCarbonQ1 = numpy.quantile(dataset.OrganicCarbon, 0.25)
OrganicCarbonQ2 = numpy.quantile(dataset.OrganicCarbon, 0.5)
OrganicCarbonQ3 = numpy.quantile(dataset.OrganicCarbon, 0.75)
OrganicCarbonIQR = OrganicCarbonQ3 - OrganicCarbonQ1
OrganicCarbonSkew = scipy.stats.skew(dataset.OrganicCarbon)
OrganicCarbonKurtosis = scipy.stats.kurtosis(dataset.OrganicCarbon)

print("OrganicCarbon mean: " + str(OrganicCarbonMean))
print("OrganicCarbon median: " + str(OrganicCarbonMedian))
print("OrganicCarbon modus: " + str(OrganicCarbonModus))
print("OrganicCarbon standard deviation: " + str(OrganicCarbonSTD))
print("OrganicCarbon variance: " + str(OrganicCarbonVariance))
print("OrganicCarbon min: " + str(OrganicCarbonMin))
print("OrganicCarbon max: " + str(OrganicCarbonMax))
print("OrganicCarbon range: " + str(OrganicCarbonRange))
print("OrganicCarbon Q1: " + str(OrganicCarbonQ1))
```

```

print("OrganicCarbon Q2: " + str(OrganicCarbonQ2))
print("OrganicCarbon Q3: " + str(OrganicCarbonQ3))
print("OrganicCarbon IQR: " + str(OrganicCarbonIQR))
print("OrganicCarbon Skew: " + str(OrganicCarbonSkew))
print("OrganicCarbon Kurtosis: " + str(OrganicCarbonKurtosis))

temp, histogram_OrganicCarbon = plt.subplots(1,1)
histogram_OrganicCarbon.hist(dataset.OrganicCarbon)
histogram_OrganicCarbon.set_xlabel("OrganicCarbon")
histogram_OrganicCarbon.set_ylabel("amount")
plt.show();

temp, boxplot_OrganicCarbon = plt.subplots(1,1)
boxplot_OrganicCarbon.boxplot(dataset.OrganicCarbon)
boxplot_OrganicCarbon.set_ylabel("OrganicCarbon")
plt.show();

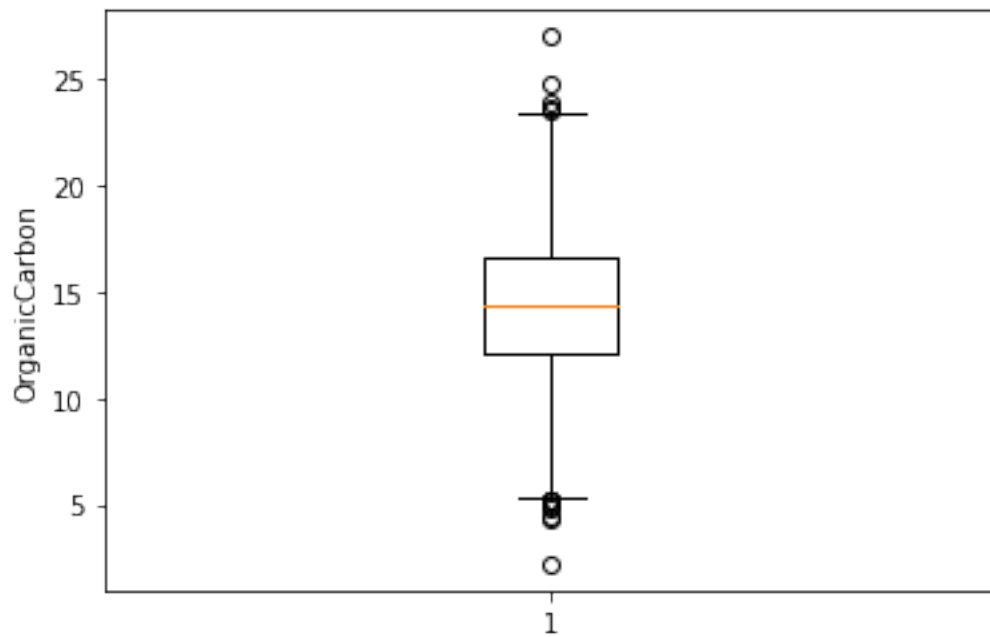
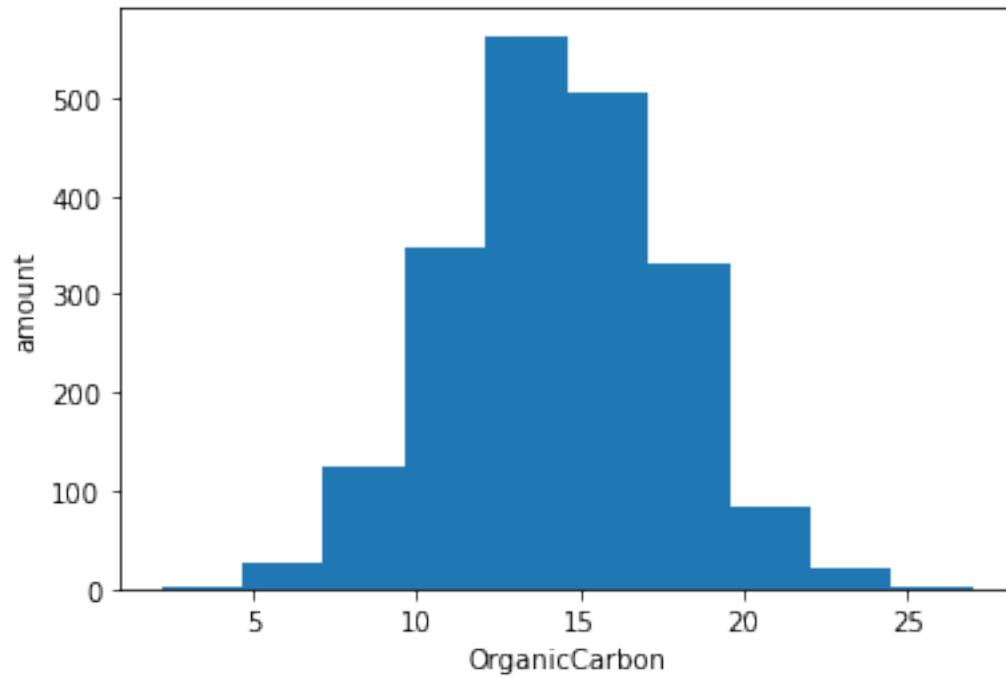
#data OrganicCarbon memiliki distribusi normal karena nilai skew dan kurtosis
↳mendekati 0
#jika dilihat di histogram, bagian paling tinggi berada di tengah, tinggi
↳bagian tengah tidak terlalu drastis dibandingkan dengan data disampingnya

```

```

OrganicCarbon mean: 14.357939902048088
OrganicCarbon median: 14.323285610653329
OrganicCarbon modus: ModeResult(mode=array([2.2]), count=array([1]))
OrganicCarbon standard deviation: 3.32494259280379
OrganicCarbon variance: 11.05524324544079
OrganicCarbon min: 2.1999999999999886
OrganicCarbon max: 27.00670661116601
OrganicCarbon range: 24.80670661116602
OrganicCarbon Q1: 12.122530374047727
OrganicCarbon Q2: 14.323285610653329
OrganicCarbon Q3: 16.683561746173808
OrganicCarbon IQR: 4.561031372126081
OrganicCarbon Skew: -0.02020466379237664
OrganicCarbon Kurtosis: 0.02795769149333216

```



```
[ ]: #Mengerjakan no 2 dan 3 untuk kolom data Trihalomethanes
```

```
TrihalomethanesMean = numpy.mean(dataset.Trihalomethanes)
```

```

TrihalomethanesMedian = numpy.median(dataset.Trihalomethanes)
TrihalomethanesModus = scipy.stats.mode(dataset.Trihalomethanes)
TrihalomethanesSTD = numpy.std(dataset.Trihalomethanes)
TrihalomethanesVariance = numpy.var(dataset.Trihalomethanes)
TrihalomethanesMin = numpy.min(dataset.Trihalomethanes)
TrihalomethanesMax = numpy.max(dataset.Trihalomethanes)
TrihalomethanesRange = TrihalomethanesMax-TrihalomethanesMin
TrihalomethanesQ1 = numpy.quantile(dataset.Trihalomethanes,0.25)
TrihalomethanesQ2 = numpy.quantile(dataset.Trihalomethanes,0.5)
TrihalomethanesQ3 = numpy.quantile(dataset.Trihalomethanes,0.75)
TrihalomethanesIQR = TrihalomethanesQ3 - TrihalomethanesQ1
TrihalomethanesSkew = scipy.stats.skew(dataset.Trihalomethanes)
TrihalomethanesKurtosis = scipy.stats.kurtosis(dataset.Trihalomethanes)

print("Trihalomethanes mean: " + str(TrihalomethanesMean))
print("Trihalomethanes median: " + str(TrihalomethanesMedian))
print("Trihalomethanes modus: " + str(TrihalomethanesModus))
print("Trihalomethanes standard deviation: " + str(TrihalomethanesSTD))
print("Trihalomethanes variance: " + str(TrihalomethanesVariance))
print("Trihalomethanes min: " + str(TrihalomethanesMin))
print("Trihalomethanes max: " + str(TrihalomethanesMax))
print("Trihalomethanes range: " + str(TrihalomethanesRange))
print("Trihalomethanes Q1: " + str(TrihalomethanesQ1))
print("Trihalomethanes Q2: " + str(TrihalomethanesQ2))
print("Trihalomethanes Q3: " + str(TrihalomethanesQ3))
print("Trihalomethanes IQR: " + str(TrihalomethanesIQR))
print("Trihalomethanes Skew: " + str(TrihalomethanesSkew))
print("Trihalomethanes Kurtosis: " + str(TrihalomethanesKurtosis))

temp, histogram_Trihalomethanes = plt.subplots(1,1)
histogram_Trihalomethanes.hist(dataset.Trihalomethanes)
histogram_Trihalomethanes.set_xlabel("Trihalomethanes")
histogram_Trihalomethanes.set_ylabel("amount")
plt.show();

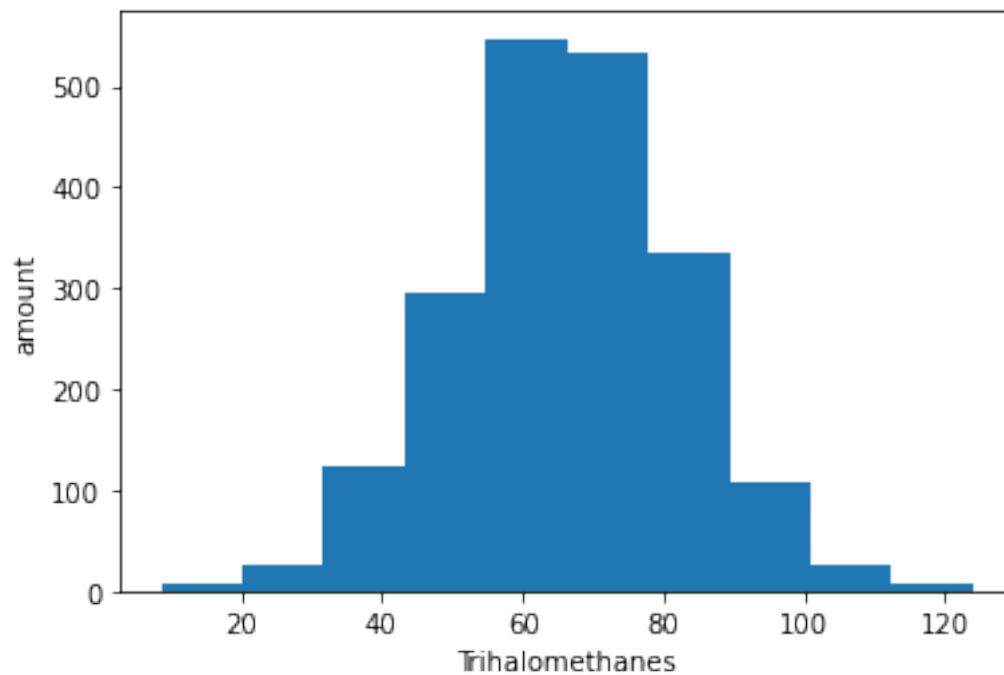
temp, boxplot_Trihalomethanes = plt.subplots(1,1)
boxplot_Trihalomethanes.boxplot(dataset.Trihalomethanes)
boxplot_Trihalomethanes.set_ylabel("Trihalomethanes")
plt.show();

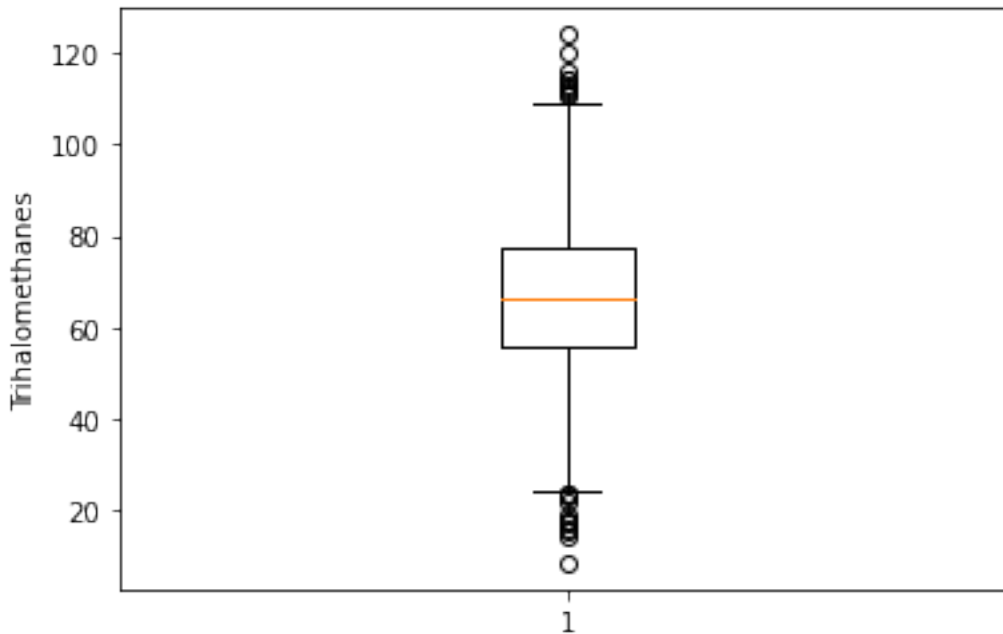
#data Trihalomethanes memiliki distribusi normal karena nilai skew dan kurtosis
↳mendekati 0
#jika dilihat di histogram, bagian paling tinggi berada di tengah, tinggi
↳bagian tengah tidak terlalu drastis dibandingkan dengan data disampingnya

```

Trihalomethanes mean: 66.40071666307463
Trihalomethanes median: 66.48204080309809

Trihalomethanes modus: ModeResult(mode=array([8.57701293]), count=array([1]))
Trihalomethanes standard deviation: 16.077108208788328
Trihalomethanes variance: 258.473408357089
Trihalomethanes min: 8.577012932983806
Trihalomethanes max: 124.0
Trihalomethanes range: 115.4229870670162
Trihalomethanes Q1: 55.94999302803186
Trihalomethanes Q2: 66.48204080309809
Trihalomethanes Q3: 77.2946128060674
Trihalomethanes IQR: 21.344619778035543
Trihalomethanes Skew: -0.051344331277615854
Trihalomethanes Kurtosis: 0.21947880896670435





[]: *#Mengerjakan no 2 dan 3 untuk kolom data Turbidity*

```
TurbidityMean = numpy.mean(dataset.Turbidity)
TurbidityMedian = numpy.median(dataset.Turbidity)
TurbidityModus = scipy.stats.mode(dataset.Turbidity)
TurbiditySTD = numpy.std(dataset.Turbidity)
TurbidityVariance = numpy.var(dataset.Turbidity)
TurbidityMin = numpy.min(dataset.Turbidity)
TurbidityMax = numpy.max(dataset.Turbidity)
TurbidityRange = TurbidityMax-TurbidityMin
TurbidityQ1 = numpy.quantile(dataset.Turbidity,0.25)
TurbidityQ2 = numpy.quantile(dataset.Turbidity,0.5)
TurbidityQ3 = numpy.quantile(dataset.Turbidity,0.75)
TurbidityIQR = TurbidityQ3 - TurbidityQ1
TurbiditySkew = scipy.stats.skew(dataset.Turbidity)
TurbidityKurtosis = scipy.stats.kurtosis(dataset.Turbidity)

print("Turbidity mean: " + str(TurbidityMean))
print("Turbidity median: " + str(TurbidityMedian))
print("Turbidity modus: " + str(TurbidityModus))
print("Turbidity standard deviation: " + str(TurbiditySTD))
print("Turbidity variance: " + str(TurbidityVariance))
print("Turbidity min: " + str(TurbidityMin))
print("Turbidity max: " + str(TurbidityMax))
print("Turbidity range: " + str(TurbidityRange))
print("Turbidity Q1: " + str(TurbidityQ1))
```

```

print("Turbidity Q2: " + str(TurbidityQ2))
print("Turbidity Q3: " + str(TurbidityQ3))
print("Turbidity IQR: " + str(TurbidityIQR))
print("Turbidity Skew: " + str(TurbiditySkew))
print("Turbidity Kurtosis: " + str(TurbidityKurtosis))

temp, histogram_Turbidity = plt.subplots(1,1)
histogram_Turbidity.hist(dataset.Turbidity)
histogram_Turbidity.set_xlabel("Turbidity")
histogram_Turbidity.set_ylabel("amount")
plt.show();

temp, boxplot_Turbidity = plt.subplots(1,1)
boxplot_Turbidity.boxplot(dataset.Turbidity)
boxplot_Turbidity.set_ylabel("Turbidity")
plt.show();

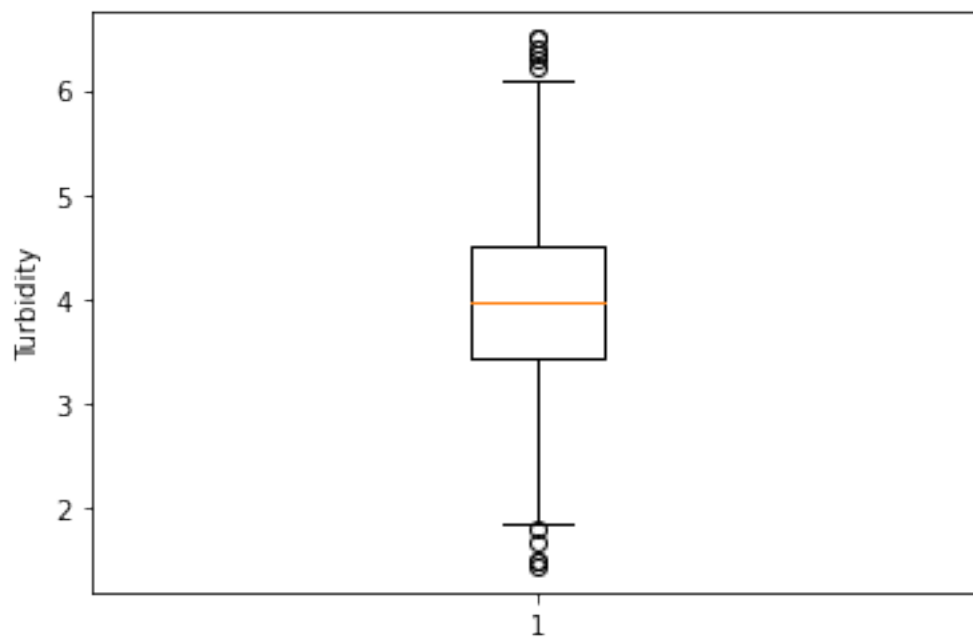
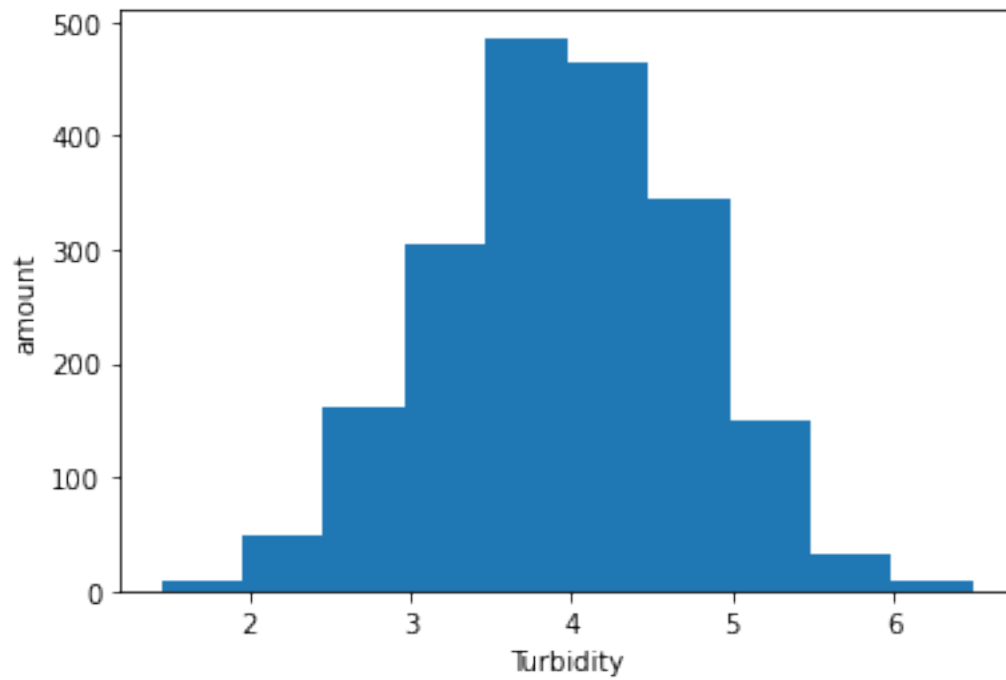
#data Turbidity memiliki distribusi normal karena nilai skew dan kurtosis
↪mendekati 0
#jika dilihat di histogram, bagian paling tinggi berada di tengah, tinggi
↪bagian tengah tidak terlalu drastis dibandingkan dengan data disampingnya

```

```

Turbidity mean: 3.969496912630371
Turbidity median: 3.967373963531836
Turbidity modus: ModeResult(mode=array([1.45]), count=array([1]))
Turbidity standard deviation: 0.7802768695296949
Turbidity variance: 0.6088319931230606
Turbidity min: 1.45
Turbidity max: 6.494748555990993
Turbidity range: 5.044748555990993
Turbidity Q1: 3.442881623557439
Turbidity Q2: 3.967373963531836
Turbidity Q3: 4.5146627202018825
Turbidity IQR: 1.0717810966444437
Turbidity Skew: -0.03224189559762075
Turbidity Kurtosis: -0.05269051630420529

```



```
[ ]: #Nomor 4a
```

```
HO_pH = 7
```



```

pH_result = pHMean
alpha = 0.05
totaldata = 2010
nilai_kritis_pH = scipy.stats.t.ppf(q=1-alpha, df=totaldata-1)
nilai_uji_pH = ((pH_result - H0_pH)/(pHSTD/math.sqrt(totaldata)))
nilai_p_pH = scipy.stats.t.sf(abs(nilai_uji_pH), df=totaldata-1)

print("hasil nilai uji: " + str(nilai_uji_pH))
print("hasil nilai p: " + str(nilai_p_pH))

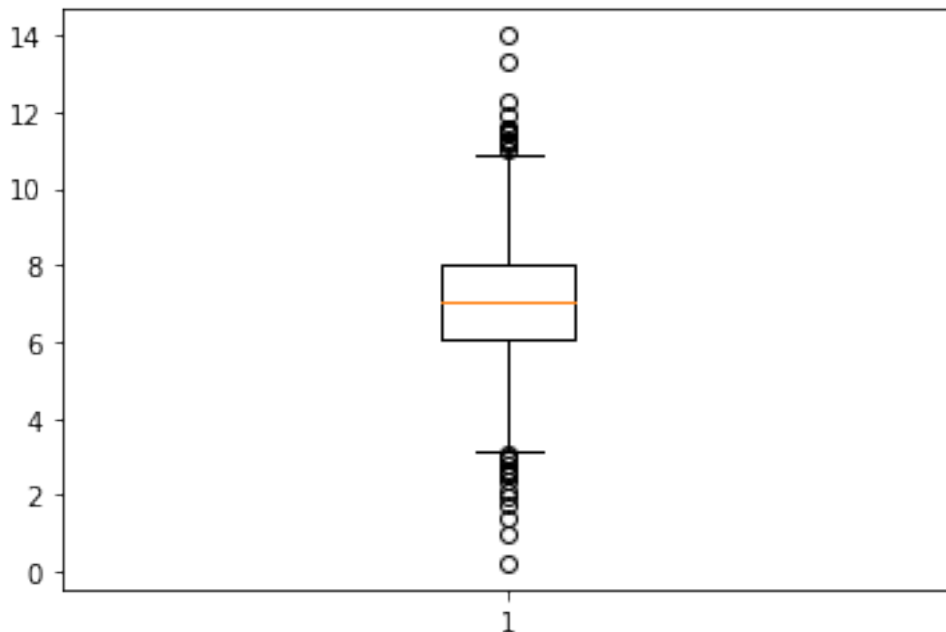
if (nilai_uji_pH > nilai_kritis_pH or nilai_p_pH < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#jika H0 ditolak, maka rata-rata pH > 7

plt.boxplot(dataset.pH)
plt.show()

```

hasil nilai uji: 2.486063648110245
 hasil nilai p: 0.006498604504908631
 H0 ditolak



```
[ ]: #Nomor 4b

H0_Hardness = 205
Hardness_result = HardnessMean
alpha = 0.05
totaldata = 2010
nilai_kritis_Hardness = scipy.stats.t.ppf(q=1-(alpha/2), df=totaldata-1)
nilai_uji_Hardness = ((Hardness_result - H0_Hardness)/(HardnessSTD/math.
    ↳sqrt(totaldata)))
nilai_p_Hardness = (scipy.stats.t.sf(abs(nilai_uji_Hardness), df=totaldata-1))↳
    ↳* 2

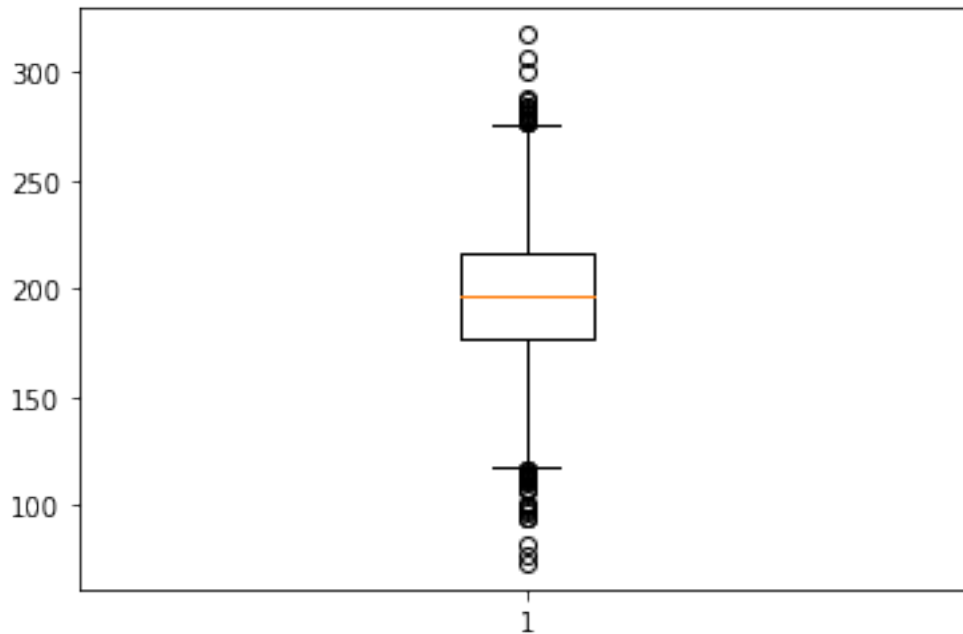
print("hasil nilai uji: " + str(nilai_uji_Hardness))
print("hasil nilai p: " + str(nilai_p_Hardness))

if ((nilai_uji_Hardness > nilai_kritis_Hardness or nilai_uji_Hardness <↳
    ↳-1*nilai_kritis_Hardness) or nilai_p_Hardness < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 ditolak, maka rata-rata Hardness != 205

plt.boxplot(dataset.Hardness)
plt.show()
```

```
hasil nilai uji: -12.406223679245638
hasil nilai p: 4.148029129970065e-34
H0 ditolak
```



[]: *#Nomor 4c*

```
SolidsMean100 = numpy.mean(dataset.Solids[0:100])
SolidsSTD100 = numpy.std(dataset.Solids[0:100])

H0_Solids = 21900
Solids_result = SolidsMean100
alpha = 0.05
totaldata = 100
nilai_kritis_Solids = scipy.stats.t.ppf(q=1-(alpha/2), df=totaldata-1)
nilai_uji_Solids = ((Solids_result - H0_Solids)/(SolidsSTD100/math.
    ↳sqrt(totaldata)))
nilai_p_Solids = (scipy.stats.t.sf(abs(nilai_uji_Solids), df=totaldata-1)) * 2

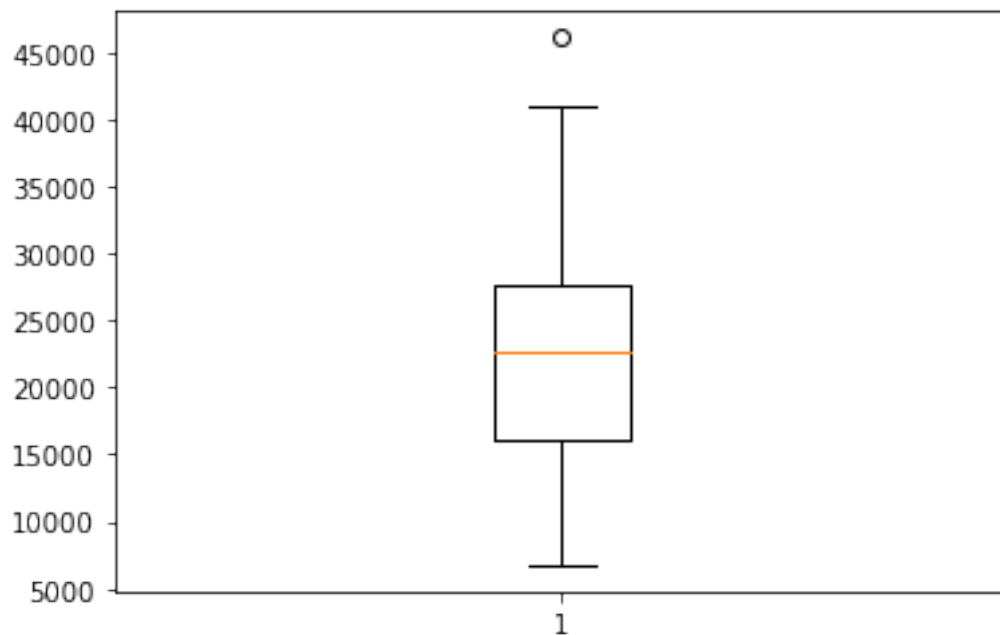
print("hasil nilai uji: " + str(nilai_uji_Solids))
print("hasil nilai p: " + str(nilai_p_Solids))

if ((nilai_uji_Solids > nilai_kritis_Solids or nilai_uji_Solids < -1 *
    ↳nilai_kritis_Solids) or nilai_p_Solids < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#karena nilai uji tidak berada di daerah kritis dan nilai p > 0.05 maka tidak
    ↳cukup data untuk menolak H0
```

```
plt.boxplot(dataset.Solids[0:100])
plt.show()
```

hasil nilai uji: 0.5665194861266858
 hasil nilai p: 0.5723227935487426
 H0 diterima



[]: *#Nomor 4d*

```
HO_Conductivity = 0.1
Conductivity_result = dataset.Conductivity[dataset.Conductivity > 450].count()/
    ↳totaldata
alpha = 0.05
totaldata = 2010
q0_Conductivity = 1-HO_Conductivity
nilai_kritis_Conductivity = scipy.stats.t.ppf(q=1-alpha, df=totaldata-1)
nilai_uji_Conductivity = (Conductivity_result - HO_Conductivity)/math.
    ↳sqrt(HO_Conductivity*q0_Conductivity/totaldata)
nilai_p_Conductivity = (scipy.stats.binom.pmf(dataset.Conductivity[dataset.
    ↳Conductivity > 450].count(), totaldata, HO_Conductivity))

print("hasil nilai uji: " + str(nilai_uji_Conductivity))
print("hasil nilai p: " + str(nilai_p_Conductivity))
```

```

if ((nilai_uji_Conductivity > nilai_kritis_Conductivity or
    nilai_uji_Conductivity < -1*nilai_kritis_Conductivity) or
    nilai_p_Conductivity < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 ditolak, maka proporsi Conductivity != 0.1

```

hasil nilai uji: 1098.4090768015349
 hasil nilai p: 8.039654739499262e-230
 H0 ditolak

```

[ ]: #Nomor 4e

HO_Trihalomethanes = 0.05
Trihalomethanes_result = dataset.Trihalomethanes[dataset.Trihalomethanes < 40].
    count()/totaldata
alpha = 0.05
totaldata = 2010
q0_Trihalomethanes = 1 - HO_Trihalomethanes
nilai_kritis_Trihalomethanes = scipy.stats.t.ppf(q=1-alpha, df=totaldata-1)
nilai_uji_Trihalomethanes = (Trihalomethanes_result - HO_Trihalomethanes)/math.
    sqrt(HO_Trihalomethanes*q0_Trihalomethanes/totaldata)
nilai_p_Trihalomethanes = (scipy.stats.binom.pmf(dataset.
    Trihalomethanes[dataset.Trihalomethanes < 40].count(), totaldata,
    HO_Trihalomethanes))

print("hasil nilai uji: " + str(nilai_uji_Trihalomethanes))
print("hasil nilai p: " + str(nilai_p_Trihalomethanes))

if ((nilai_uji_Trihalomethanes < nilai_kritis_Trihalomethanes) or
    nilai_p_Trihalomethanes < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 ditolak, maka proporsi Trihalomethanes < 0.05

```

hasil nilai uji: 0.5628826416670951
 hasil nilai p: 0.03404367379225386
 H0 ditolak

```

[ ]: #Nomor 5a

SulfateMean1 = numpy.mean(dataset.Sulfate[0:int(len(dataset.Sulfate)/2)])

```

```

SulfateMean2 = numpy.mean(dataset.Sulfate[int(len(dataset.Sulfate)/2):
↳len(dataset.Sulfate)])
SulfateSTD1 = numpy.std(dataset.Sulfate[0:int(len(dataset.Sulfate)/2)])
SulfateSTD2 = numpy.std(dataset.Sulfate[int(len(dataset.Sulfate)/2):len(dataset.
↳Sulfate)])
print("mean first half: " + str(SulfateMean1))
print("mean second half: " + str(SulfateMean2))
print("std first half: " + str(SulfateSTD1))
print("std second half: " + str(SulfateSTD2))

H0_Sulfate = 0
Sulfate_result = SulfateMean1-SulfateMean2
alpha = 0.05
totaldata = 2010
Sulfate_S12 = math.sqrt((((totaldata/2)-1) * (SulfateSTD1**2) + ((totaldata/
↳2)-1) * (SulfateSTD2**2)) / totaldata-2)
nilai_kritis_Sulfate = scipy.stats.t.ppf(q=1-(alpha/2), df=totaldata-2)
nilai_uji_Sulfate = (Sulfate_result-H0_Sulfate)/(Sulfate_S12*math.sqrt((1/
↳(totaldata/2)) + (1/(totaldata/2))))
nilai_p_Sulfate = scipy.stats.t.sf(abs(nilai_uji_Sulfate), df=totaldata) * 2

print("hasil nilai uji: " + str(nilai_uji_Sulfate))
print("hasil nilai p: " + str(nilai_p_Sulfate))

if ((nilai_uji_Sulfate > nilai_kritis_Sulfate or nilai_uji_Sulfate <
↳-1*nilai_kritis_Sulfate) and nilai_p_Sulfate < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 ditolak, maka SulfateMean1 != SulfateMean2

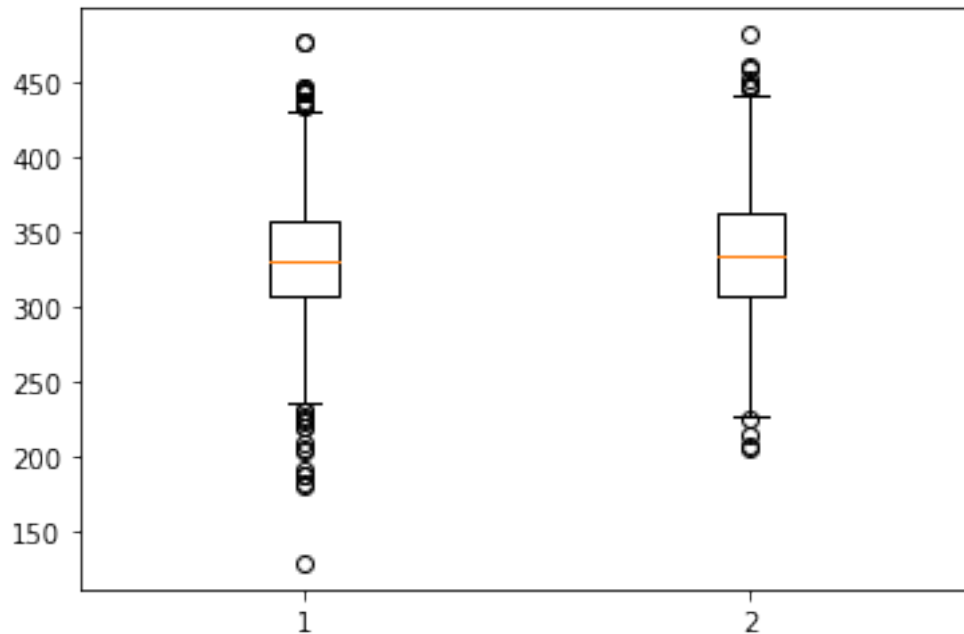
data = [(dataset.Sulfate[0:int(len(dataset.Sulfate)/2)]),(dataset.
↳Sulfate[int(len(dataset.Sulfate)/2):len(dataset.Sulfate)])]
plt.boxplot(data)
plt.show()

```

```

mean first half: 331.3053295054953
mean second half: 335.11742332488234
std first half: 41.31218591361095
std second half: 41.00071588624226
hasil nilai uji: -2.0785647605086353
hasil nilai p: 0.03778395451096237
H0 ditolak

```



[]: *#Nomor 5b*

```
OrganicCarbonMean1 = numpy.mean(dataset.OrganicCarbon[0:int(len(dataset.
    ↳OrganicCarbon)/2)])
OrganicCarbonMean2 = numpy.mean(dataset.OrganicCarbon[int(len(dataset.
    ↳OrganicCarbon)/2):len(dataset.OrganicCarbon)])
OrganicCarbonSTD1 = numpy.std(dataset.OrganicCarbon[0:int(len(dataset.
    ↳OrganicCarbon)/2)])
OrganicCarbonSTD2 = numpy.std(dataset.OrganicCarbon[int(len(dataset.
    ↳OrganicCarbon)/2):len(dataset.OrganicCarbon)])
print("mean first half: " + str(OrganicCarbonMean1))
print("mean second half: " + str(OrganicCarbonMean2))
print("std first half: " + str(OrganicCarbonSTD1))
print("std second half: " + str(OrganicCarbonSTD2))

H0_OrganicCarbon = 0.15
OrganicCarbon_result = OrganicCarbonMean1-OrganicCarbonMean2
alpha = 0.05
totaldata = 2010
OrganicCarbon_S12 = math.sqrt((((totaldata/2)-1) * (OrganicCarbonSTD1**2) +
    ↳((totaldata/2)-1) * (OrganicCarbonSTD2**2)) / totaldata-2)
nilai_kritis_OrganicCarbon = scipy.stats.t.ppf(q=1-alpha, df=totaldata-2)
nilai_uji_OrganicCarbon = (OrganicCarbon_result-H0_OrganicCarbon)/
    ↳((OrganicCarbon_S12*math.sqrt((1/(totaldata/2)) + (1/(totaldata/2)))))
```

```

nilai_p_OrganicCarbon = scipy.stats.t.sf(abs(nilai_uji_OrganicCarbon),
↳df=totaldata) * 2

print("hasil nilai uji: " + str(nilai_uji_OrganicCarbon))
print("hasil nilai p: " + str(nilai_p_OrganicCarbon))

if ((nilai_uji_OrganicCarbon > nilai_kritis_OrganicCarbon) or
↳nilai_p_OrganicCarbon < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 ditolak, maka bagian awal OrganicCarbon lebih besar dari bagian akhir
↳OrganicCarbon
#tetapi jika dilihat dari data, bagian awal OrganicCarbon lebih kecil dari
↳bagian akhir OrganicCarbon

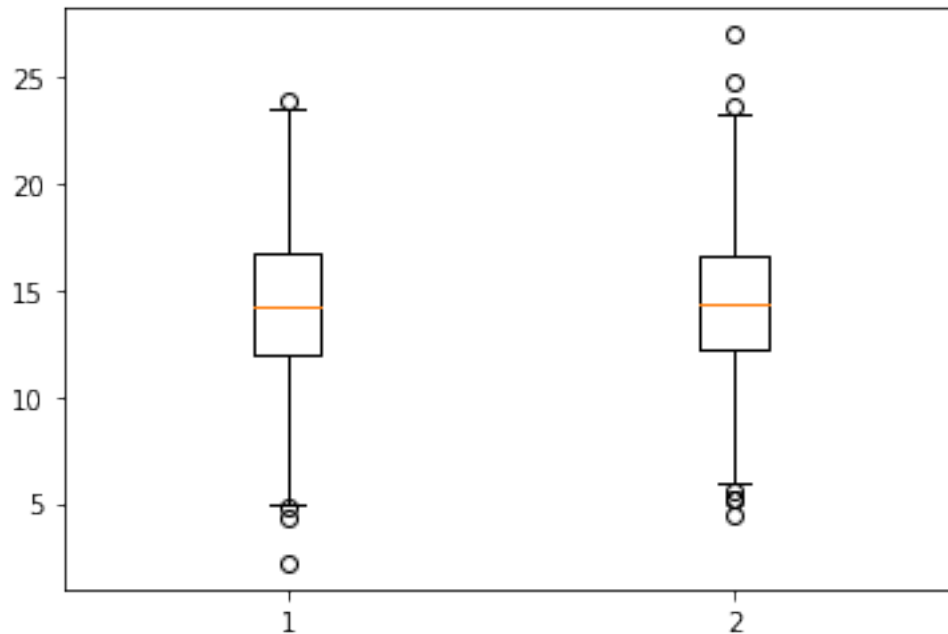
data = [(dataset.OrganicCarbon[0:int(len(dataset.OrganicCarbon)/2)]), (dataset.
↳OrganicCarbon[int(len(dataset.OrganicCarbon)/2):len(dataset.OrganicCarbon)])]
plt.boxplot(data)
plt.show()

```

```

mean first half: 14.253972723723418
mean second half: 14.461907080372761
std first half: 3.3494944109867837
std second half: 3.296931199359838
hasil nilai uji: -2.669590985872705
hasil nilai p: 0.00765550051537214
H0 ditolak

```

[]: *#Nomor 5c*

```
ChloraminesMean1 = numpy.mean(dataset.Chloramines[0:100])
ChloraminesMean2 = numpy.mean(dataset.Chloramines[int(len(dataset.
    ↳Chloramines)-100):len(dataset.Chloramines)])
ChloraminesSTD1 = numpy.std(dataset.Chloramines[0:100])
ChloraminesSTD2 = numpy.std(dataset.Chloramines[int(len(dataset.
    ↳Chloramines)-100):len(dataset.Chloramines)])
print("mean first half: " + str(ChloraminesMean1))
print("mean second half: " + str(ChloraminesMean2))
print("std first half: " + str(ChloraminesSTD1))
print("std second half: " + str(ChloraminesSTD2))

H0_Chloramines = 0
Chloramines_result = ChloraminesMean1-ChloraminesMean2
alpha = 0.05
totaldata = 200
Chloramines_S12 = math.sqrt((((totaldata/2)-1) * (ChloraminesSTD1**2) +
    ↳((totaldata/2)-1) * (ChloraminesSTD2**2)) / (totaldata-2))
nilai_kritis_Chloramines = scipy.stats.t.ppf(q=1-(alpha/2), df=totaldata-2)
nilai_uji_Chloramines = Chloramines_result/(Chloramines_S12*math.sqrt((1/
    ↳(totaldata/2)) + (1/(totaldata/2))))
nilai_p_Chloramines = scipy.stats.t.sf(abs(nilai_uji_Chloramines),
    ↳df=totaldata-2) * 2
```

```

print("hasil nilai uji: " + str(nilai_uji_Chloramines))
print("hasil nilai p: " + str(nilai_p_Chloramines))

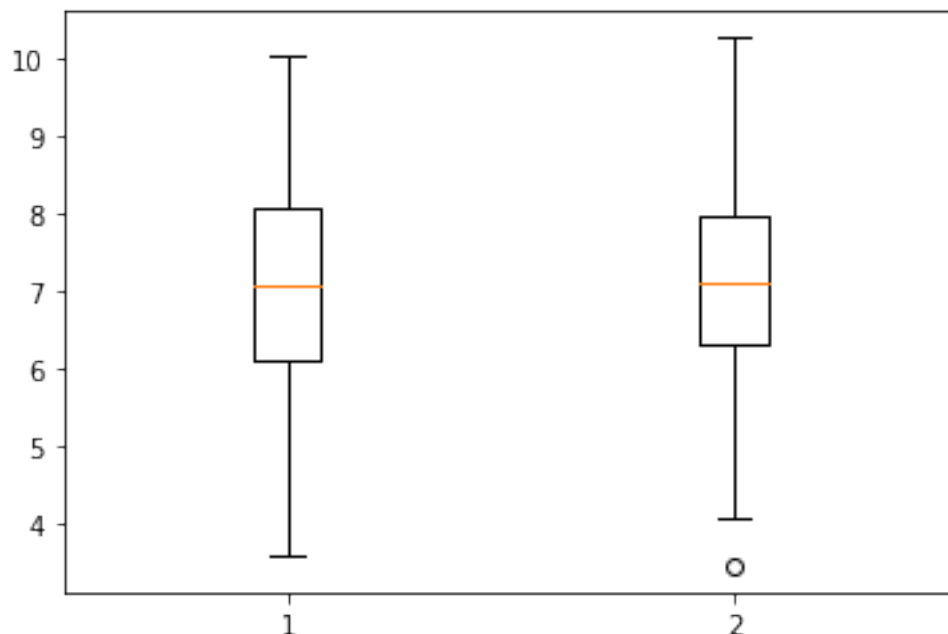
if ((nilai_uji_Chloramines > nilai_kritis_Chloramines or nilai_uji_Chloramines_
    ↳ < -1 * nilai_kritis_Chloramines) or nilai_p_Chloramines < alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 tidak memiliki cukup bukti untuk ditolak sehingga rata-rata 100 data awal_
    ↳ Chloramines sama dengan rata-rata 100 data akhir Chloramines

data = [(dataset.Chloramines[0:100]), (dataset.Chloramines[int(len(dataset.
    ↳ Chloramines)-100):len(dataset.Chloramines)])]
plt.boxplot(data)
plt.show()

```

mean first half: 7.007771140423916
 mean second half: 7.147197636249925
 std first half: 1.4734692281775796
 std second half: 1.3002558770558545
 hasil nilai uji: -0.7094988920428625
 hasil nilai p: 0.478849594192584
 H0 diterima



[]: *#Nomor 5d*

```
datasetTurbidity1 = dataset.Turbidity[0:int(len(dataset.Turbidity)/2)]
datasetTurbidity2 = dataset.Turbidity[int(len(dataset.Turbidity)/2):len(dataset.
    ↳Turbidity)]
TurbidityRatio1 = datasetTurbidity1[datasetTurbidity1 > 4].count() /
    ↳len(datasetTurbidity1)
TurbidityRatio2 = datasetTurbidity2[datasetTurbidity2 > 4].count() /
    ↳len(datasetTurbidity2)
print("Ratio 1: " + str(TurbidityRatio1))
print("Ratio 2: " + str(TurbidityRatio2))

H0_Turbidity = 0
Turbidity_result = TurbidityRatio1 - TurbidityRatio2
alpha = 0.05
totaldata = 2010
p_accen = (datasetTurbidity1[datasetTurbidity1 > 4].count() +
    ↳datasetTurbidity2[datasetTurbidity2 > 4].count()) / totaldata
q_accen = 1-p_accen
nilai_kritis_Turbidity = scipy.stats.t.ppf(q=1-alpha, df=totaldata)
nilai_uji_Turbidity = (Turbidity_result-H0_Turbidity)/(math.
    ↳sqrt(p_accen*q_accen*((1/(totaldata/2)) + (1/(totaldata/2)))))
nilai_p_Turbidity = scipy.stats.t.sf(abs(nilai_uji_Turbidity), df=totaldata)

print("hasil nilai uji: " + str(nilai_uji_Turbidity))
print("hasil nilai p: " + str(nilai_p_Turbidity))

if ((nilai_uji_Turbidity > nilai_kritis_Turbidity) or nilai_p_Turbidity <
    ↳alpha):
    print("H0 ditolak")
else:
    print("H0 diterima")

#H0 tidak memiliki cukup bukti untuk ditolak sehingga ratio Turbidity awal sama
    ↳dengan ratio Turbidity akhir
```

Ratio 1: 0.4835820895522388
Ratio 2: 0.48656716417910445
hasil nilai uji: -0.13388958661778735
hasil nilai p: 0.44675164429687164
H0 diterima

[]: *#Nomor 5e*

```
SulfateVariance1 = numpy.var(dataset.Sulfate[0:int(len(dataset.Sulfate)/2)])
SulfateVariance2 = numpy.var(dataset.Sulfate[int(len(dataset.Sulfate)/2):
    ↳len(dataset.Sulfate)])
```

```

print("Variance 1: " + str(SulfateVariance1))
print("Variance 2: " + str(SulfateVariance2))

H0_Sulfate = 0
Sulfate_result = SulfateVariance1-SulfateVariance2
alpha = 0.05
totaldata = 2010
batas1_Sulfate = scipy.stats.f.ppf(1-(alpha/2),(totaldata/2)-1,(totaldata/2)-1)
batas2_Sulfate = scipy.stats.f.ppf(alpha/2,(totaldata/2)-1,(totaldata/2)-1)
nilai_f_Sulfate = SulfateVariance1/SulfateVariance2

print("batas 1: " + str(batas1_Sulfate))
print("batas 2: " + str(batas2_Sulfate))
print("nilai f: " + str(nilai_f_Sulfate))

if (batas2_Sulfate > batas1_Sulfate):
    if(nilai_f_Sulfate > batas1_Sulfate and nilai_f_Sulfate < batas2_Sulfate):
        print("H0 diterima")
    else:
        print("H0 ditolak")
else:
    if(nilai_f_Sulfate > batas2_Sulfate and nilai_f_Sulfate < batas1_Sulfate):
        print("H0 diterima")
    else:
        print("H0 ditolak")

#Nilai f berada di dalam batas sehingga pengujian diterima

```

```

Variance 1: 1706.6967049607554
Variance 2: 1681.0587031843588
batas 1: 1.1317692392568777
batas 2: 0.883572344355818
nilai f: 1.0152511043950052
H0 diterima

```

```

[ ]: #Nomor 6
#Tes korelasi data Potability dengan kolom data lain
#Menggunakan metode Pearson

Potability_pH_corr = scipy.stats.pearsonr(dataset.Potability, dataset.pH)
print("koefisien korelasi: " +str(Potability_pH_corr[0]))
plt.scatter(dataset.Potability, dataset.pH)
plt.show()

Potability_Hardness_corr = scipy.stats.pearsonr(dataset.Potability, dataset.
↪Hardness)
print("koefisien korelasi: " + str(Potability_Hardness_corr[0]))

```

```

plt.scatter(dataset.Potability, dataset.Hardness)
plt.show()

Potability_Solids_corr = scipy.stats.pearsonr(dataset.Potability, dataset.
    ↪Solids)
print("koefisien korelasi: " + str(Potability_Solids_corr[0]))
plt.scatter(dataset.Potability, dataset.Solids)
plt.show()

Potability_Chloramines_corr = scipy.stats.pearsonr(dataset.Potability, dataset.
    ↪Chloramines)
print("koefisien korelasi: " + str(Potability_Chloramines_corr[0]))
plt.scatter(dataset.Potability, dataset.Chloramines)
plt.show()

Potability_Sulfate_corr = scipy.stats.pearsonr(dataset.Potability, dataset.
    ↪Sulfate)
print("koefisien korelasi: " + str(Potability_Sulfate_corr[0]))
plt.scatter(dataset.Potability, dataset.Sulfate)
plt.show()

Potability_Conductivity_corr = scipy.stats.pearsonr(dataset.Potability, dataset.
    ↪Conductivity)
print("koefisien korelasi: " + str(Potability_Conductivity_corr[0]))
plt.scatter(dataset.Potability, dataset.Conductivity)
plt.show()

Potability_OrganicCarbon_corr = scipy.stats.pearsonr(dataset.Potability, ↪
    ↪dataset.OrganicCarbon)
print("koefisien korelasi: " + str(Potability_OrganicCarbon_corr[0]))
plt.scatter(dataset.Potability, dataset.OrganicCarbon)
plt.show()

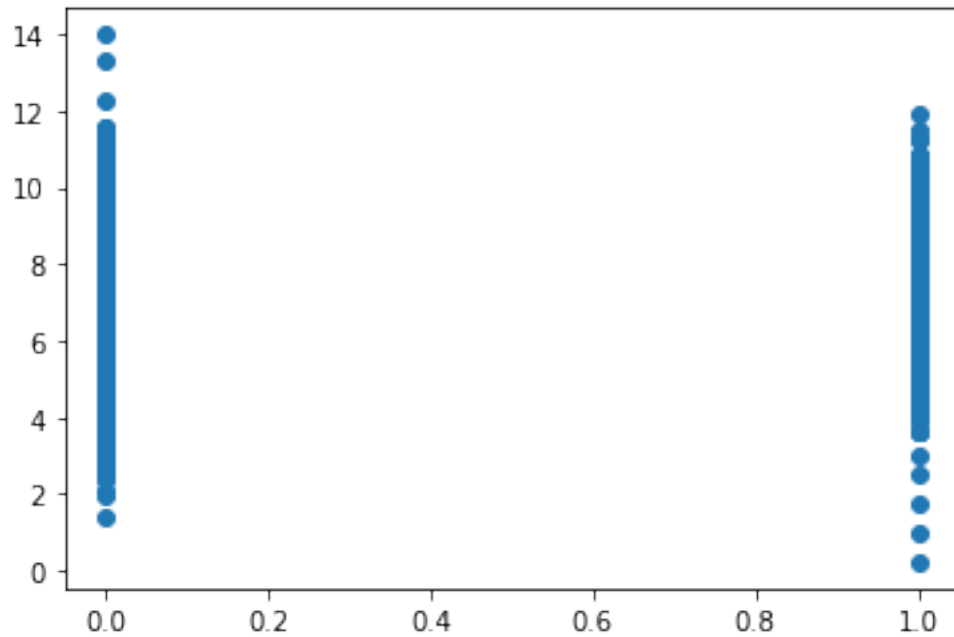
Potability_Trihalomethanes_corr = scipy.stats.pearsonr(dataset.Potability, ↪
    ↪dataset.Trihalomethanes)
print("koefisien korelasi: " + str(Potability_Trihalomethanes_corr[0]))
plt.scatter(dataset.Potability, dataset.Trihalomethanes)
plt.show()

Potability_Turbidity_corr = scipy.stats.pearsonr(dataset.Potability, dataset.
    ↪Turbidity)
print("koefisien korelasi: " + str(Potability_Turbidity_corr[0]))
plt.scatter(dataset.Potability, dataset.Turbidity)
plt.show()

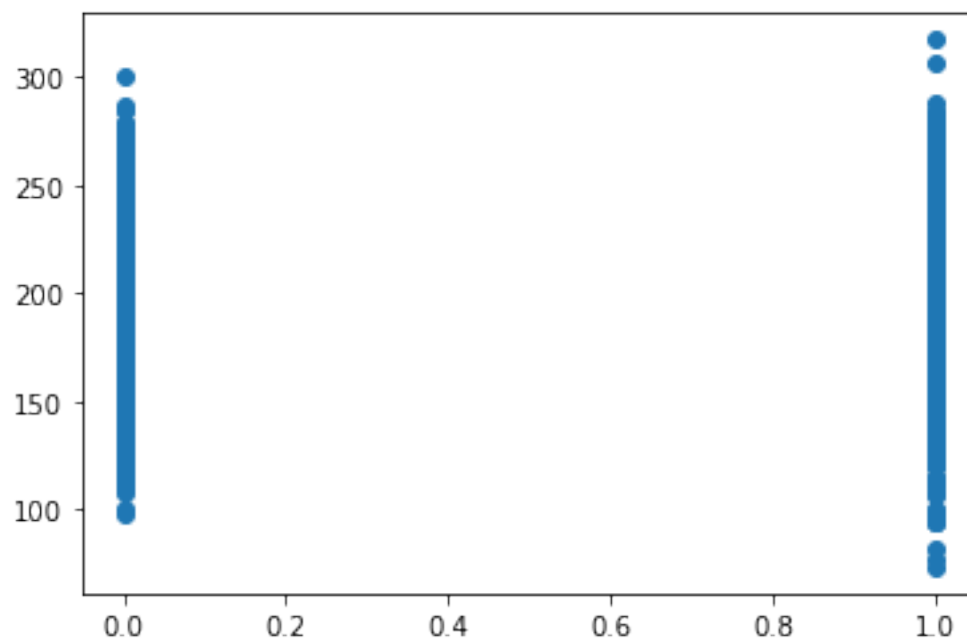
```

#semua koefisien korelasi memiliki nilai mendekati 0 sehingga tidak ada
↪ korelasi antara kolom target dan kolom non target

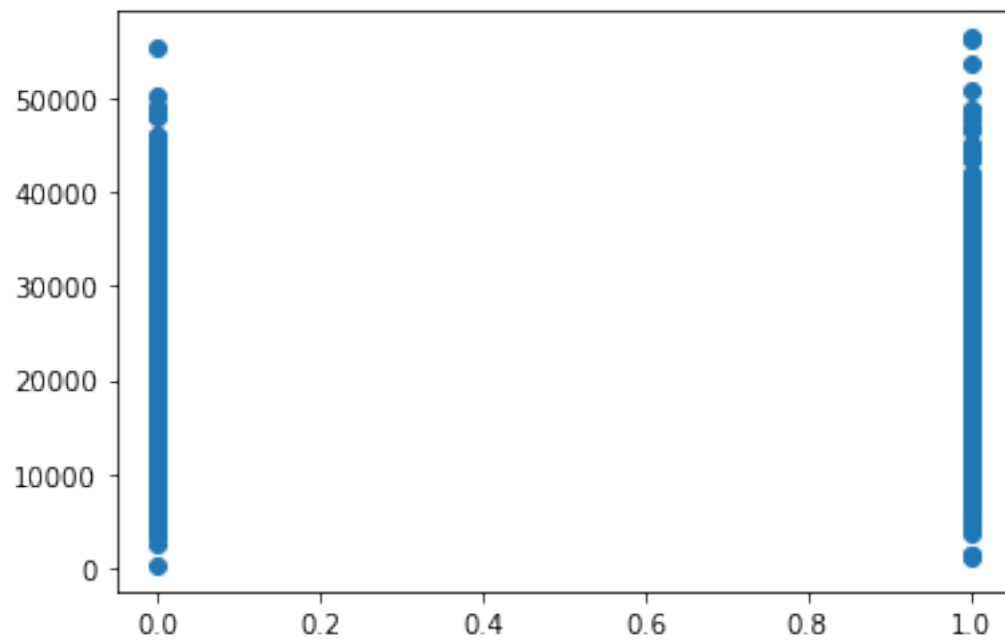
koefisien korelasi: 0.01547509440843326



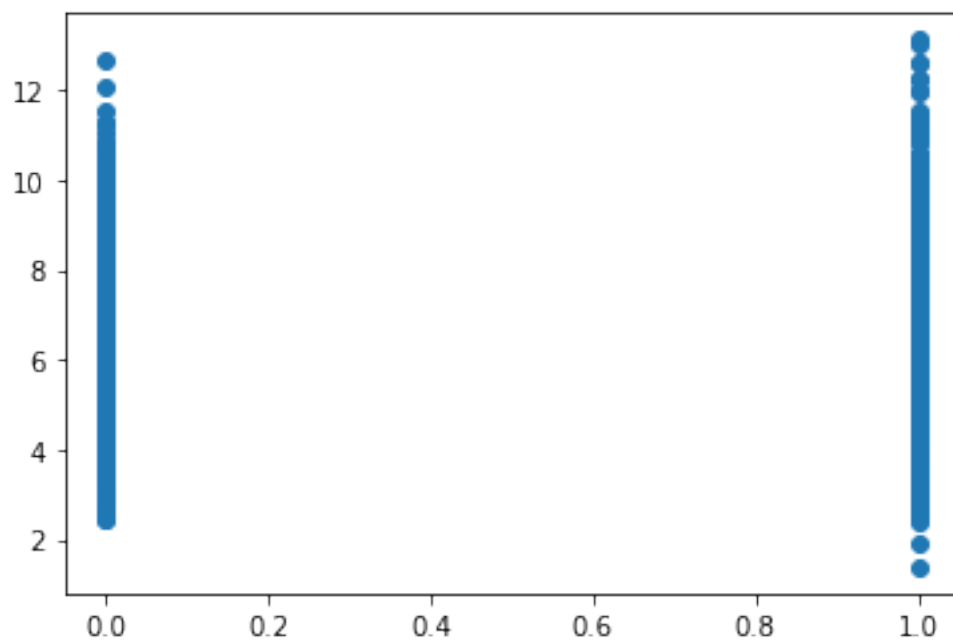
koefisien korelasi: -0.0014631528959479327



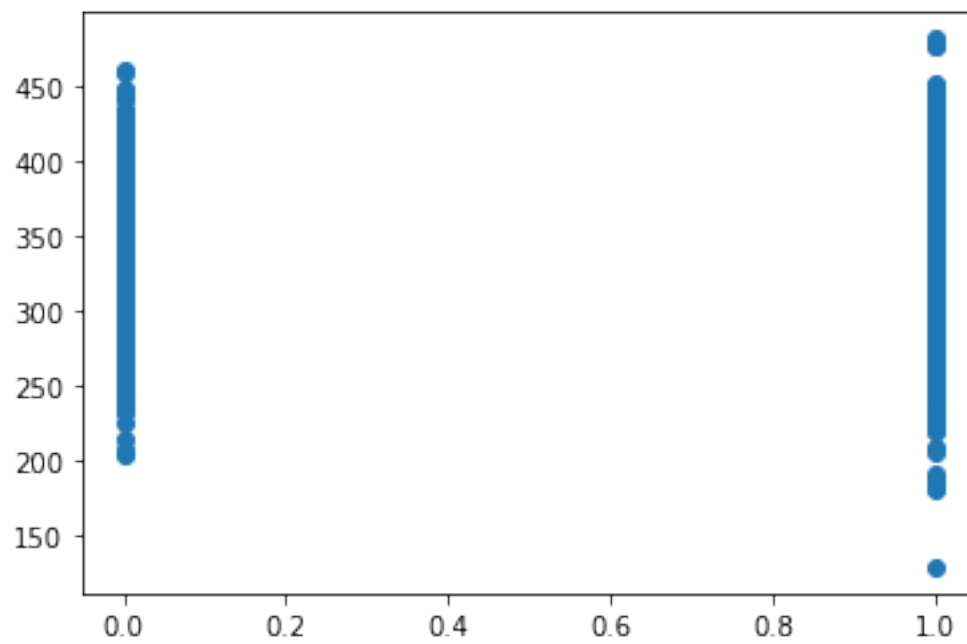
koefisien korelasi: 0.038976578181734174



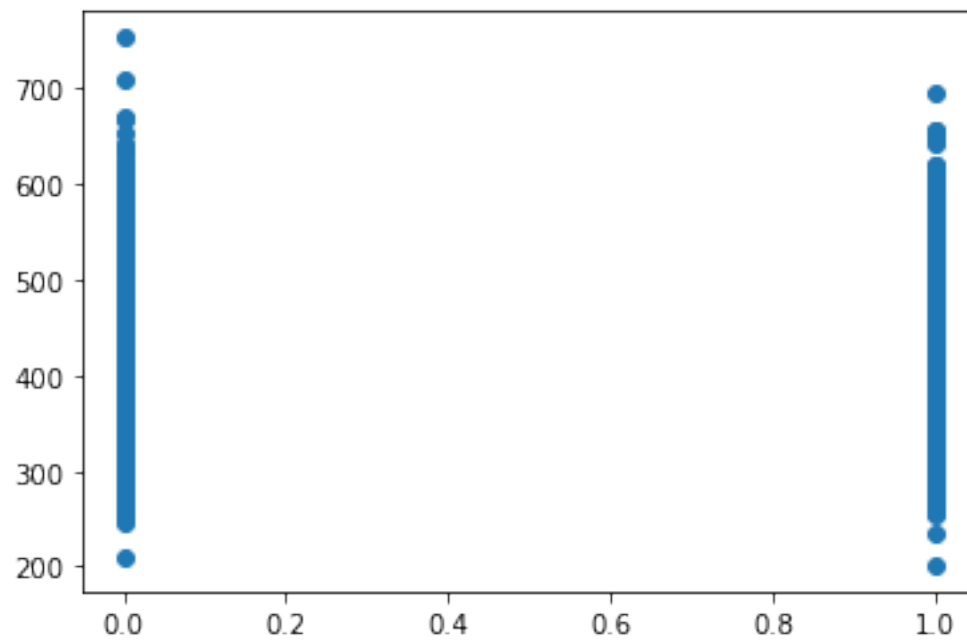
koefisien korelasi: 0.020778921840523837



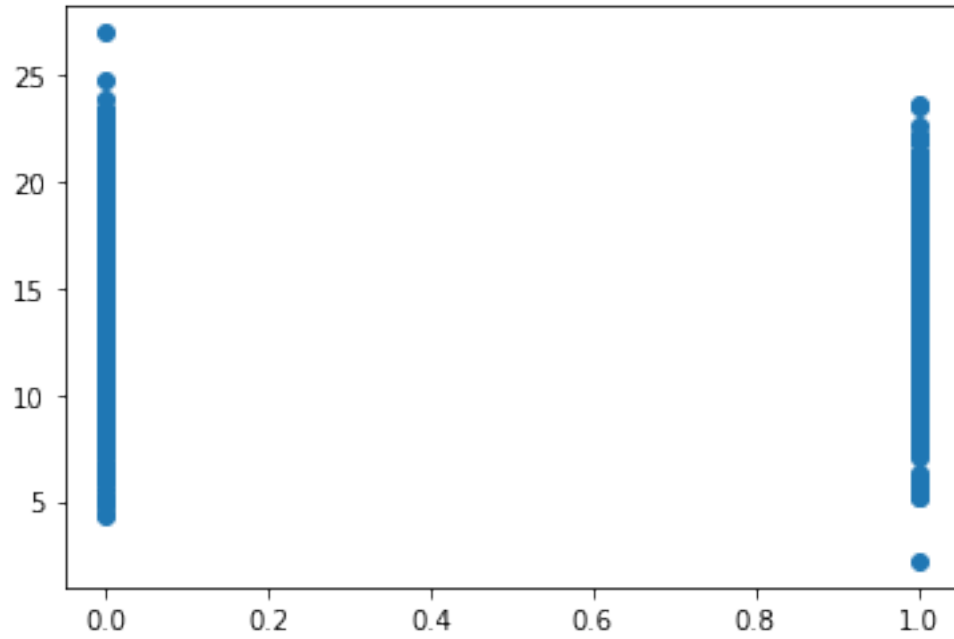
koefisien korelasi: -0.01570316441927358



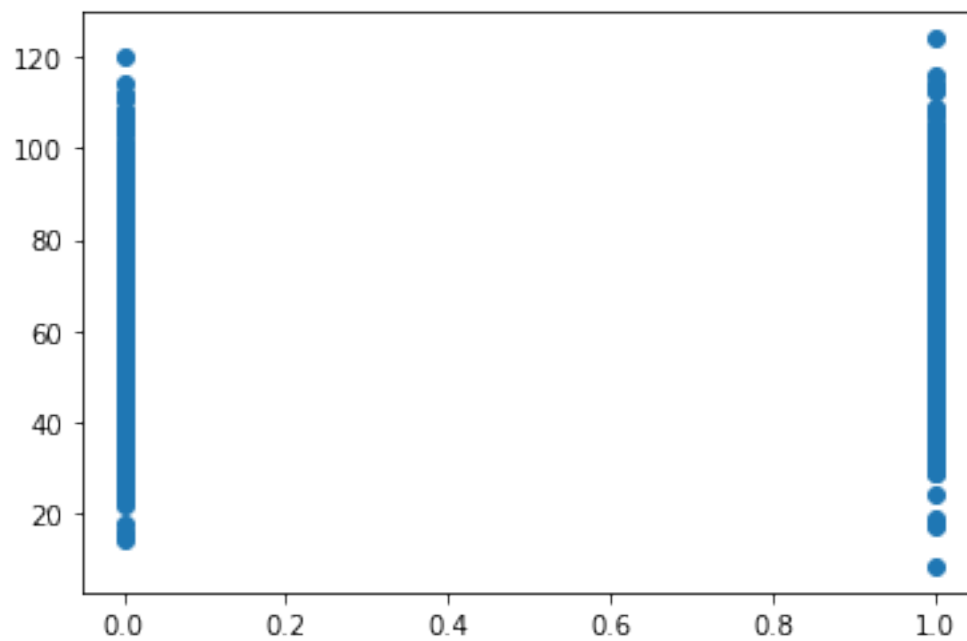
koefisien korelasi: -0.01625712011137684



koefisien korelasi: -0.01548846191074708



koefisien korelasi: 0.009236711064712903



koefisien korelasi: 0.022331042640622363

