

Tugas Besar IF2220 Probabilitas dan Statistika

Penarikan Kesimpulan dan Pengujian Hipotesis

- 13519197- Muhammad Jafer Gurdari
- 13519206- Muhammad Fawwad Naabigh

```
In [1]: # Import libraries
import pandas as pd
import matplotlib.pyplot as plt
from scipy import stats
import seaborn as sns
import numpy as np
import math

In [2]: #read file
column_list = ["Id", "Daerah", "SumbuUtama", "SumbuKecil", "Keunikan", "AreaBulatan", "Diameter", "KadarAir", "Kelling", "Bulatan", "Ransum", "Kelas"]
df = pd.read_csv("Gandum.csv", names=column_list)

Out [2]:
```

Id	Daerah	SumbuUtama	SumbuKecil	Keunikan	AreaBulatan	Diameter	KadarAir	Kelling	Bulatan	Ransum	Kelas
0	5781	128.288875	58.470848	0.890095	5954	85.793926	0.874090	316.756	0.724041	2.194068	1
1	2	4176	109.348294	49.837688	0.890098	4277	72.918093	0.596231	260.346	0.774227	2.194068
2	3	4555	114.427991	52.151207	0.890105	4706	76.155155	0.776641	279.606	0.732159	2.194158
3	4	4141	108.701191	49.457349	0.890409	4236	72.611879	0.633190	260.478	0.766980	2.197877
4	5	5273	122.747869	55.757848	0.890876	5431	81.937733	0.669842	302.730	0.723031	2.201446

1. Menulis deskripsi statistika (Descriptive Statistics) dari semua kolom pada data yang bersifat numerik, terdiri dari mean, median, modus, standar deviasi, variansi, range, nilai minimum, maksimum, kuartil, IQR, skewness dan kurtosis. Boleh juga ditambahkan deskripsi lain.

Fungsi untuk mendapatkan descriptive statistics

```
In [3]: def addToData(Tabel, namaKolom):
    data = []
    data.append(df[namaKolom].mean())
    data.append(df[namaKolom].median())
    # data.append(df[namaKolom].mode())
    data.append(df[namaKolom].std())
    data.append(df[namaKolom].var(skipna = True)) # variance
    data.append(df[namaKolom].min())
    data.append(df[namaKolom].max())
    data.append(df[namaKolom].skew())
    # skewness
    data.append(df[namaKolom].kurt())

    # kurtosis
    data.append(df[namaKolom].kurt())
    data.append(data)

    # pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])

def printModus(namaKolom):
    # Print modus
    modus = df[namaKolom].mode()
    print("Modus :")
    for item in modus:
        print("-", item)

def printQuartile(namaKolom):
    # Quartile
    # First quartile (Q1)
    Q1 = np.percentile(df[namaKolom], 25)

    # Third quartile (Q3)
    Q3 = np.percentile(df[namaKolom], 75)

    # Interquartile range (IQR)
    IQR = Q3 - Q1

    # Quartile
    print("Quartile 25% :", Q1)
    print("Quartile 75% :", Q3)
    print("Interquartile : ", IQR)

PERHATIAN untuk kolom yang setiap datanya unik, tidak dituliskan modus, karena dinilai tidak informatif
```

Daerah

```
In [4]: # DataFrame daerah
Tabel = []
addToData(Tabel, "Daerah")
printModus("Daerah")
printQuartile("Daerah")
dfDaerah = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfDaerah

Modus :
- 4992
- 4881
- 5642
- 6083
Quartile 25% : 4042.75
Quartile 75% : 5495.5
Interquartile : 1452.75

Out [4]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	4801.246	4735.0	986.395491	972976.065615	2522	7453	4931	0.238144	-0.434531

SumbuUtama

Modus tidak dimasukkan karena setiap data unik

```
In [5]: Tabel = []
addToData(Tabel, "SumbuUtama")
printModus("SumbuUtama")
dfSumbuUtama = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfSumbuUtama

Quartile 25% : 104.11609817499999
Quartile 75% : 129.046792025
Interquartile : 24.93069385000001

Out [5]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	116.045171	115.40514	18.2682626	334.254412	74.133114	227.928593	153.795469	0.761529	4.330534

SumbuKecil

Modus tidak dimasukkan karena setiap data unik

```
In [6]: Tabel = []
addToData(Tabel, "SumbuKecil")
printModus("SumbuKecil")
dfSumbuKecil = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfSumbuKecil

Quartile 25% : 51.1935763325
Quartile 75% : 56.3251578825
Interquartile : 5.131581650000001

Out [6]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	53.715246	53.731199	4.071075	16.57365	39.906517	68.9777	29.071182	-0.010828	0.475568

Keunikan

Modus tidak dimasukkan karena setiap data unik

```
In [7]: Tabel = []
addToData(Tabel, "Keunikan")
printModus("Keunikan")
dfKeunikan = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfKeunikan

Quartile 25% : 0.8636757527500001
Quartile 75% : 0.907377917
Interquartile : 0.04390216424999993

Out [7]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	0.878764	0.890045	0.036586	0.001339	0.719916	0.914001	0.194085	-1.623472	2.917256

AreaBulatan

```
In [8]: Tabel = []
addToData(Tabel, "AreaBulatan")
printModus("AreaBulatan")
dfAreaBulatan = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfAreaBulatan

Modus :
- 3802
- 4913
Quartile 25% : 4170.25
Quartile 75% : 5654.25
Interquartile : 1484.0

Out [8]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	4937.048	4857.0	1011.686255	1.023529e+06	2579	7720	5141	0.25756	-0.408685

Diameter

```
In [9]: Tabel = []
addToData(Tabel, "Diameter")
printModus("Diameter")
dfDiameter = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfDiameter

Modus :
- 71.29356396
- 78.8325579
- 84.75622403
- 88.00634154
Quartile 25% : 0.57263245725
Quartile 75% : 0.7266333445000001
Interquartile : 0.15400088725000001

Out [9]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	77.771158	63.764277	8.059867	64.913111	56.666558	97.41383	40.747172	0.022725	-0.468455

KadarAir

```
In [10]: Tabel = []
addToData(Tabel, "KadarAir")
printModus("KadarAir")
dfKadarAir = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfKadarAir

Modus :
- 0.735849057
- 0.824404762
Quartile 25% : 0.57263245725
Quartile 75% : 0.7266333445000001
Interquartile : 0.15400088725000001

Out [10]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	0.648372	0.626117	0.094367	0.008905	0.409927	0.878999	0.468972	0.459661	-0.743226

Kelling

```
In [11]: Tabel = []
addToData(Tabel, "Kelling")
printQuartile("Kelling")
dfKelling = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfKelling

Quartile 25% : 255.89299999999998
Quartile 75% : 306.0625
Interquartile : 50.179500000000002

Out [11]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	281.479722	280.0455	37.335402	1393.932221	197.015	488.837	291.822	0.733627	2.272885

Bulatan

```
In [12]: Tabel = []
addToData(Tabel, "Bulatan")
printQuartile("Bulatan")
dfBulatan = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfBulatan

Quartile 25% : 0.731990728
Quartile 75% : 0.79636096975
Interquartile : 0.06437024175000006

Out [12]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	0.761737	0.761288	0.061702	0.003807	0.17459	0.904748	0.730158	-3.599237	29.975096

Ransum

```
In [13]: Tabel = []
addToData(Tabel, "Ransum")
printModus("Ransum")
dfRansum = pd.DataFrame(Tabel, columns=["Mean", "Median", "Std", "Variance", "Min", "Max", "Range", "Skewness", "Kurtosis"])
dfRansum

Quartile 25% : 1.98393879075
Quartile 75% : 2.38161221825
Interquartile : 0.3976734275

Out [13]:
```

	Mean	Median	Std	Variance	Min	Max	Range	Skewness	Kurtosis
0	2.150915	2.193599	0.249767	0.062383	1.440796	2.464809	1.024013	-0.658188	-0.428656

2. Membuat Visualisasi plot distribusi, dalam bentuk histogram dan boxplot untuk setiap kolom numerik. Berikan uraian penjelasan kondisi setiap kolom berdasarkan kedua plot tersebut.

Dari seluruh kolom, semuanya merupakan kolom numerik kecuali kolom **Kelas** yang merupakan kolom kategorik (**Kelas 1** atau **Kelas 2**)

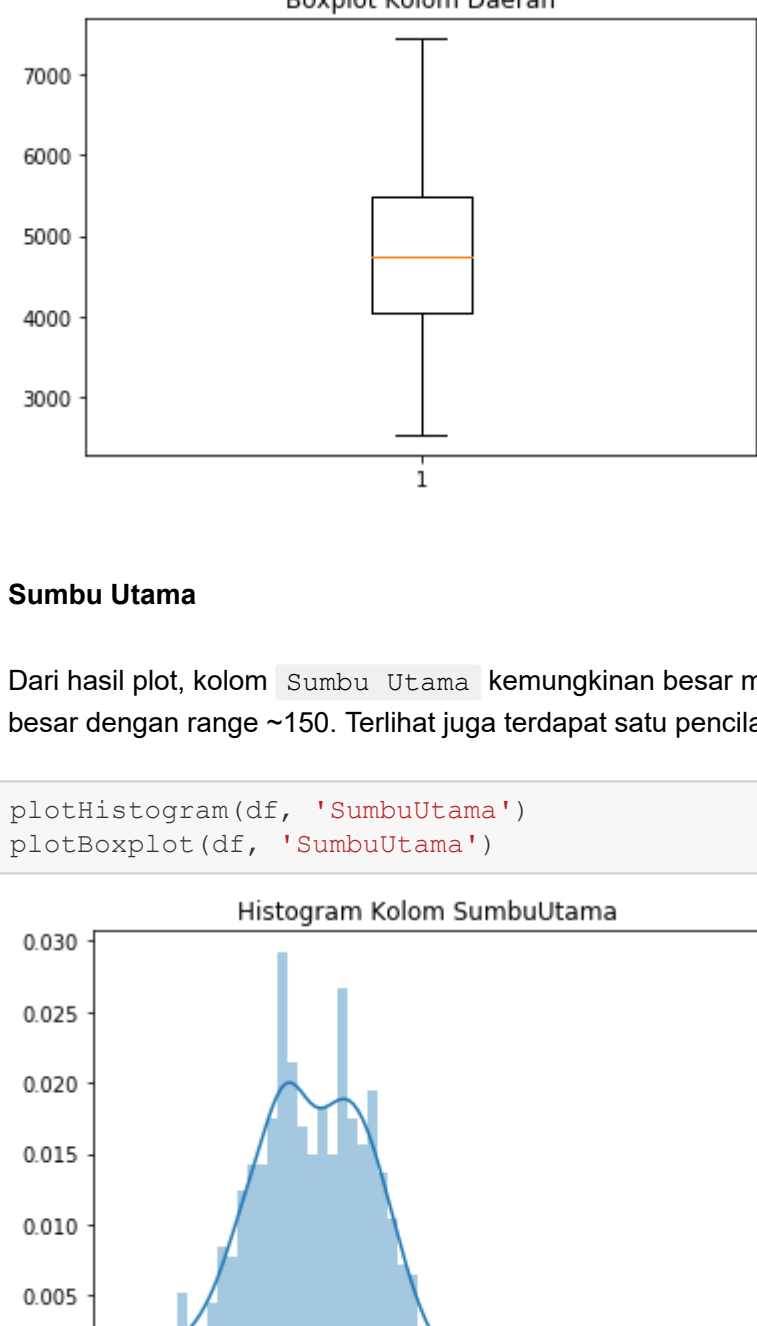
```
In [14]: # Histogram
sns.distplot(df[kolom],
             sns.distplot(df[kolom], kde = True, bins = 50).set(title="Histogram " + "Kolom " + kolom)
             plt.show()

# Boxplot
def plotBoxplot(df, kolom):
    plt.title("Boxplot " + "Kolom " + kolom)
    plt.boxplot(df[kolom])
    plt.show()
```

Daerah

Dari hasil plot didapat bahwa kolom **Daerah** memiliki bimodal distribution dengan range ~5000 dan tidak memiliki pencilan.

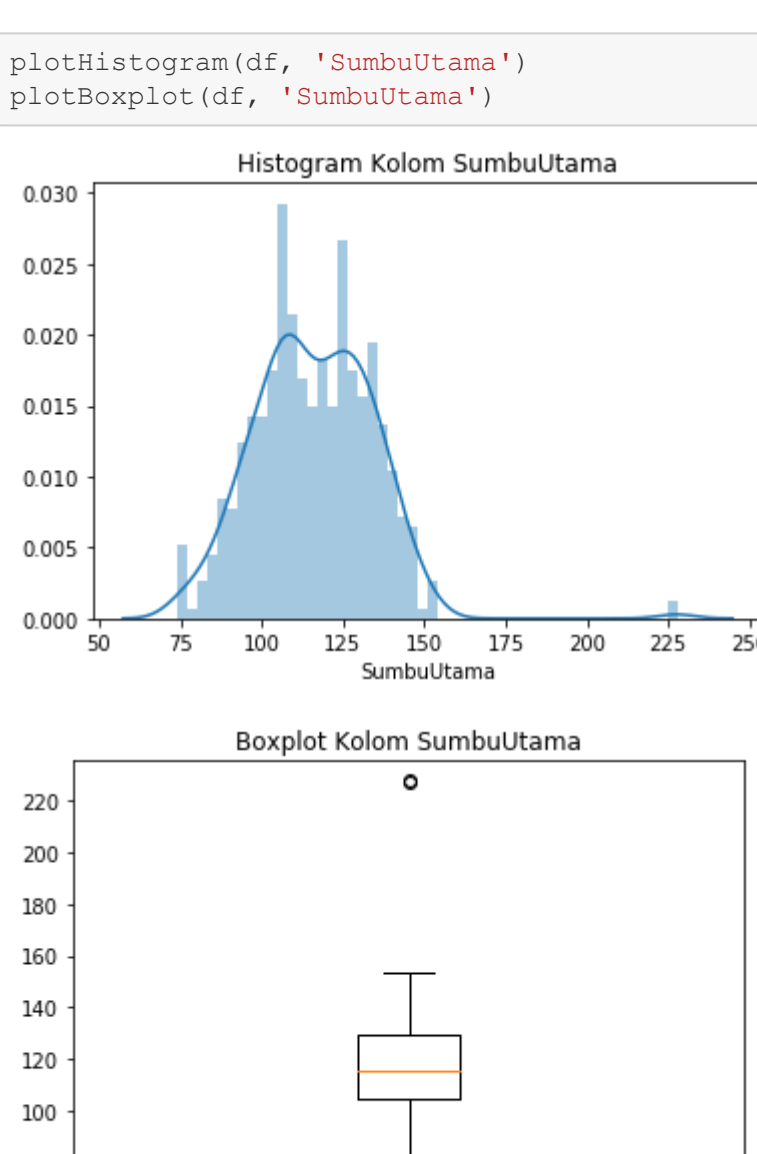
```
In [15]: plotHistogram(df, 'Daerah')
plotBoxplot(df, 'Daerah')
```



Sumbu Utama

Dari hasil plot, kolom **Sumbu Utama** kemungkinan besar memiliki nilai kurtosis yang relatif tinggi dengan sebaran data yang tidak begitu besar dengan range ~150. Terlihat juga terdapat satu pencilan.

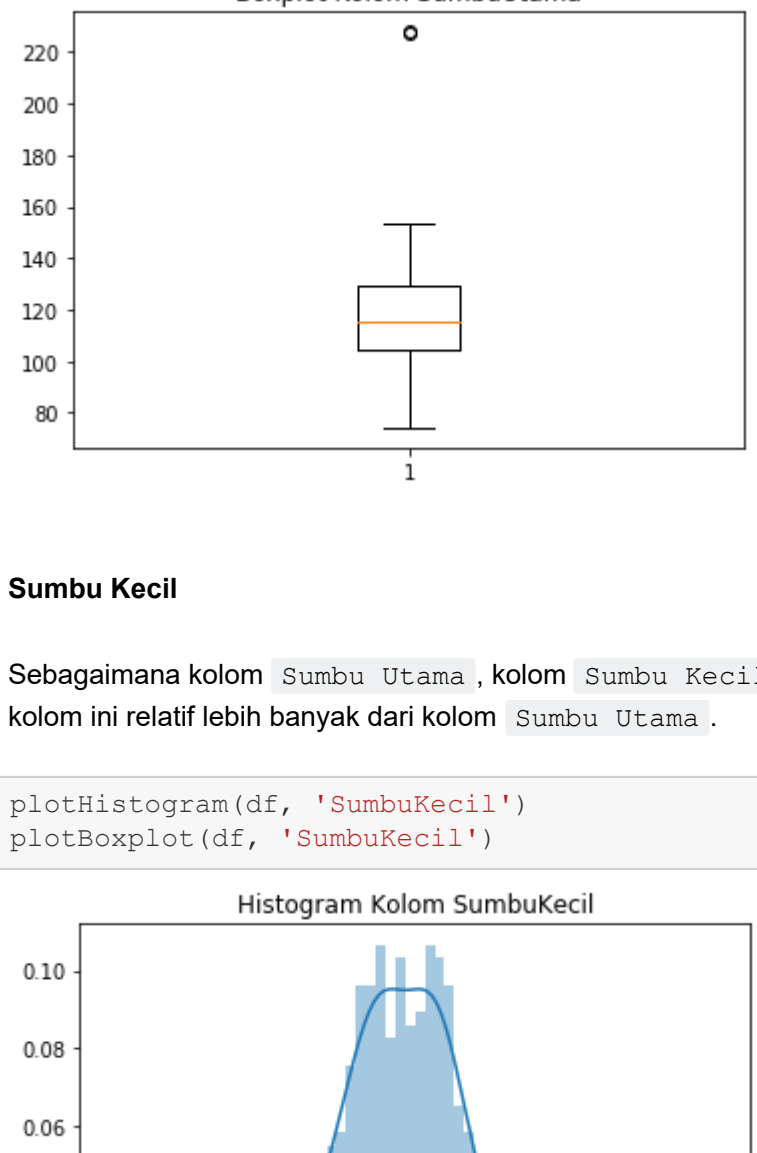
```
In [16]: plotHistogram(df, 'SumbuUtama')
plotBoxplot(df, 'SumbuUtama')
```



Sumbu Kecil

Sebagaimana kolom **Sumbu Utama**, kolom **Sumbu Kecil** memiliki kurtosis yang relatif tinggi dengan range ~30. Pencilan yang dimiliki kolom ini relatif lebih banyak dari kolom **Sumbu Utama**.

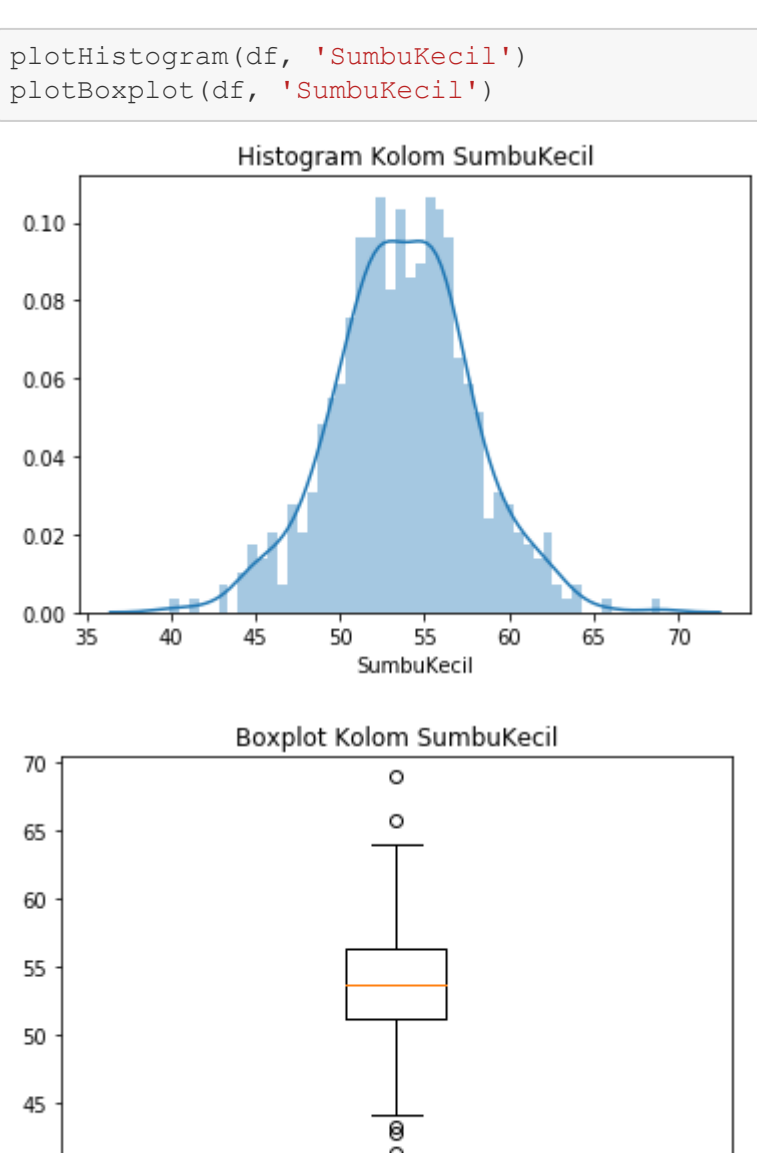
```
In [17]: plotHistogram(df, 'SumbuKecil')
plotBoxplot(df, 'SumbuKecil')
```



Keunikan

Terlihat jelas bahwa kolom **Keunikan** memiliki bentuk skewed-left dengan pencilan yang cukup banyak di tail-nya.

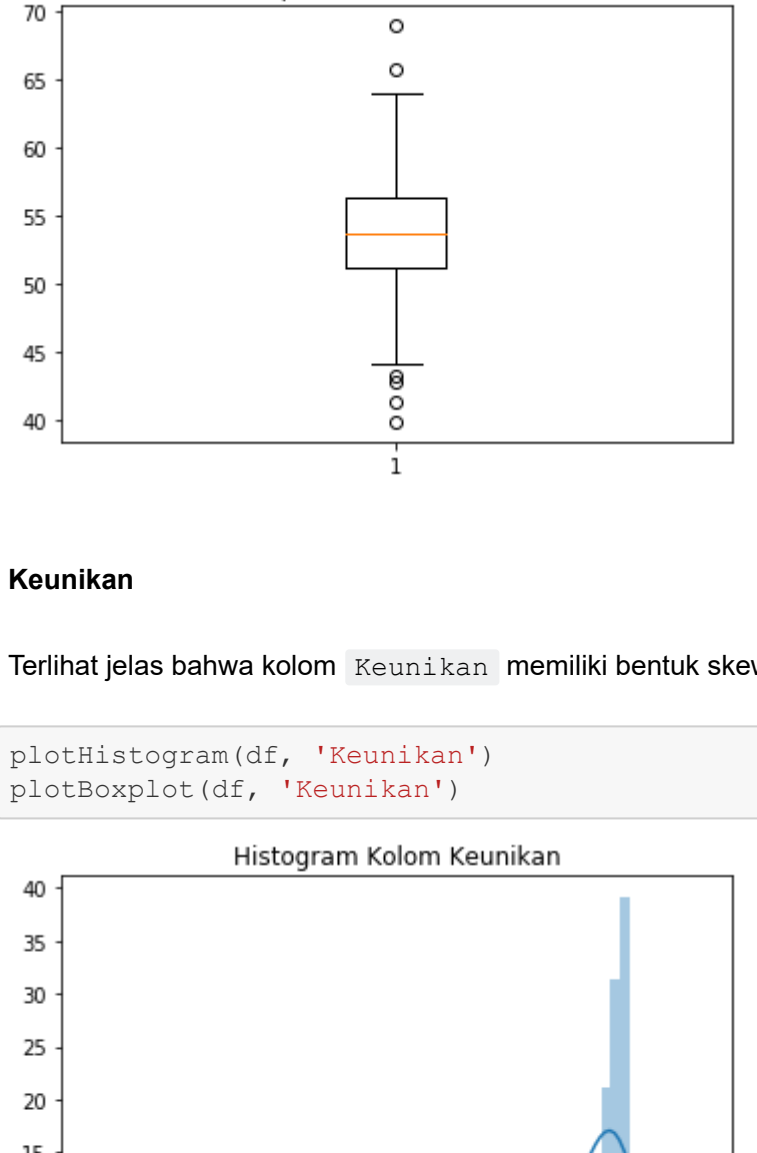
```
In [18]: plotHistogram(df, 'Keunikan')
plotBoxplot(df, 'Keunikan')
```



Area Bulatan

Kolom **Area Bulatan** terlihat memiliki bentuk bimodal dengan range ~5000 dan tidak memiliki pencilan sama sekali.

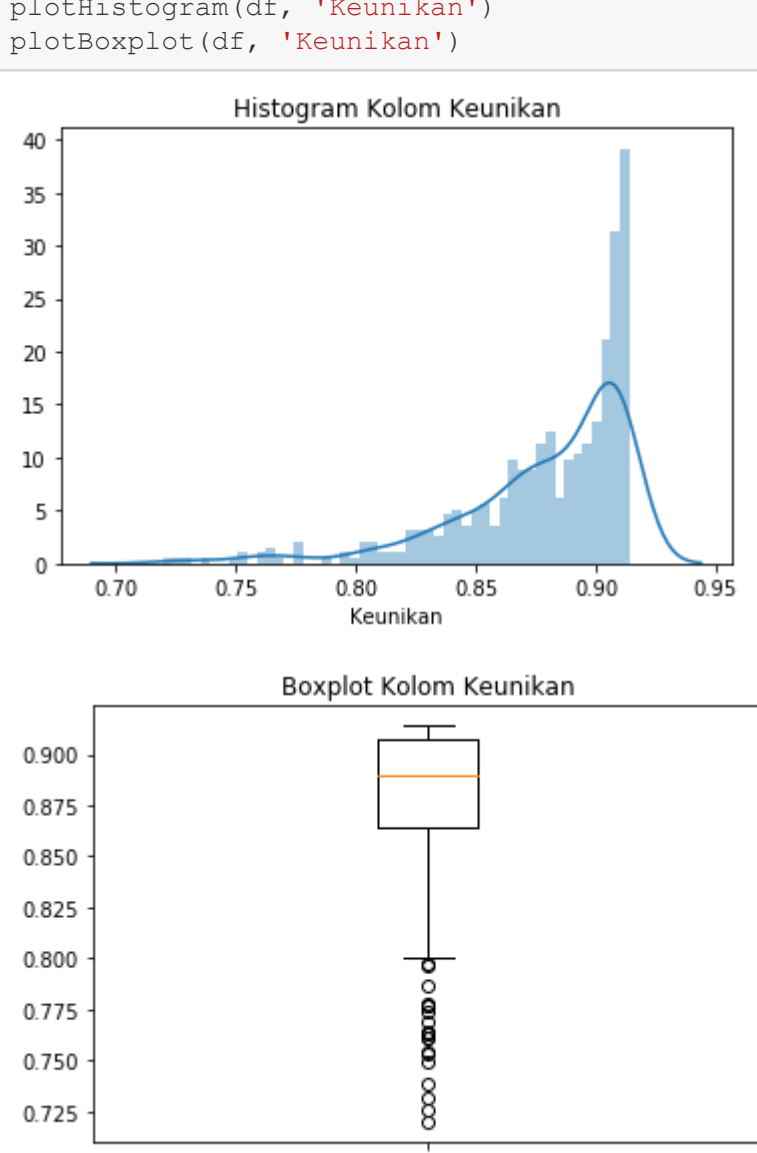
```
In [19]: plotHistogram(df, 'AreaBulatan')
plotBoxplot(df, 'AreaBulatan')
```



Diameter

Kolom **Diameter** terlihat memiliki bentuk bimodal dengan nilai ~75 sebagai modus. Terlihat juga bahwa kolom ini tidak memiliki pencilan.

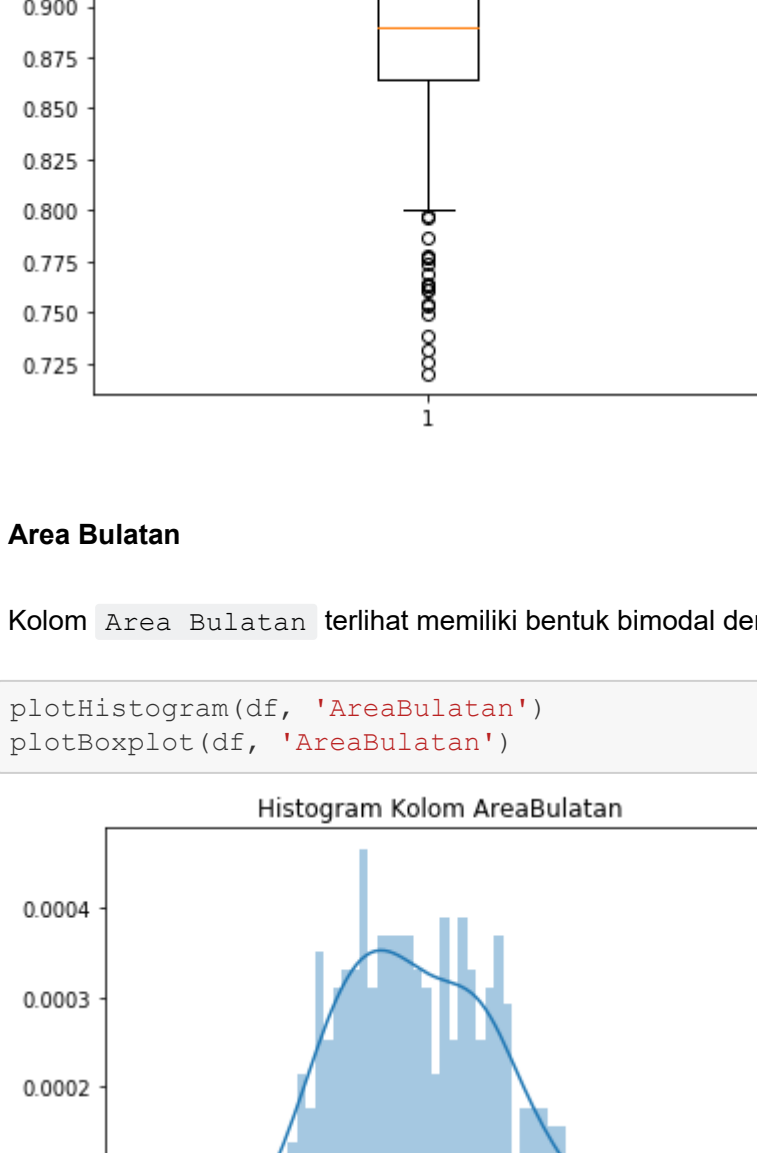
```
In [20]: plotHistogram(df, 'Diameter')
plotBoxplot(df, 'Diameter')
```



Kadar Air

Dari hasil plot, kolom **Kadar Air** tidak memiliki range ~0.5 dan modus di nilai ~0.6. Kolom ini tidak memiliki pencilan.

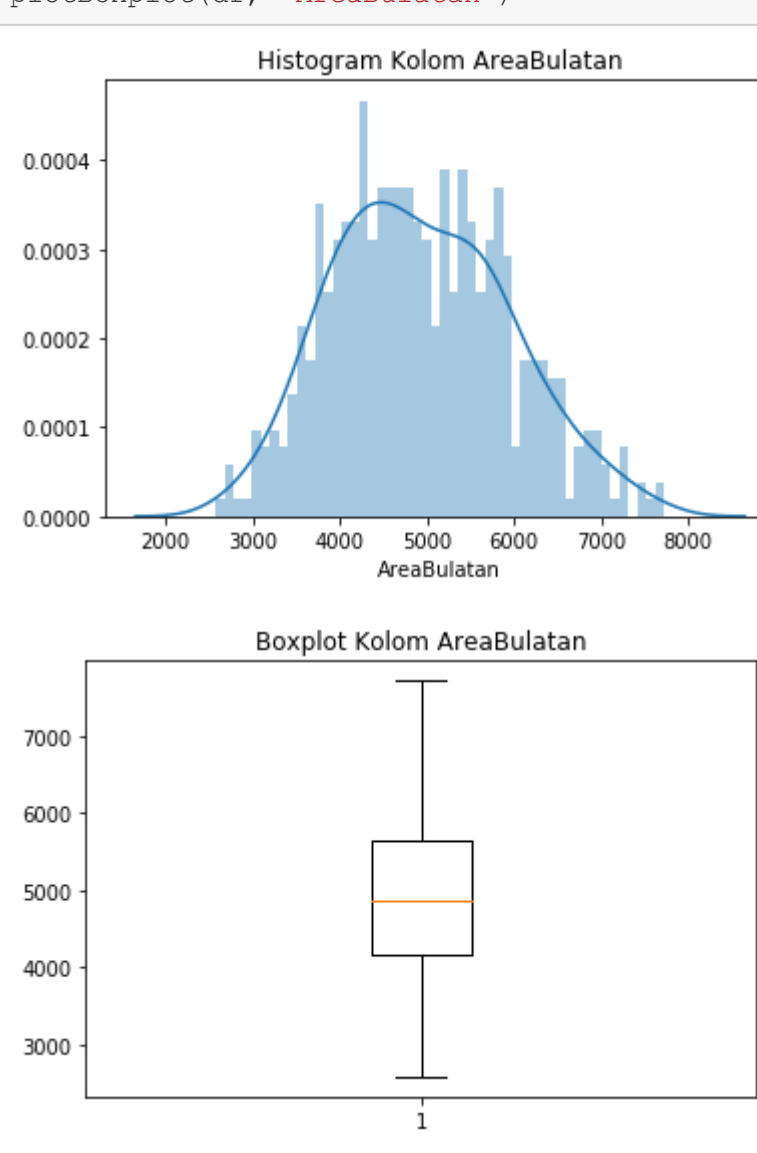
```
In [21]: plotHistogram(df, 'KadarAir')
plotBoxplot(df, 'KadarAir')
```



Kelling

Terlihat pada plot bahwa kolom ini memiliki range 300 dengan beberapa pencilan pada nilai yang relatif tinggi.

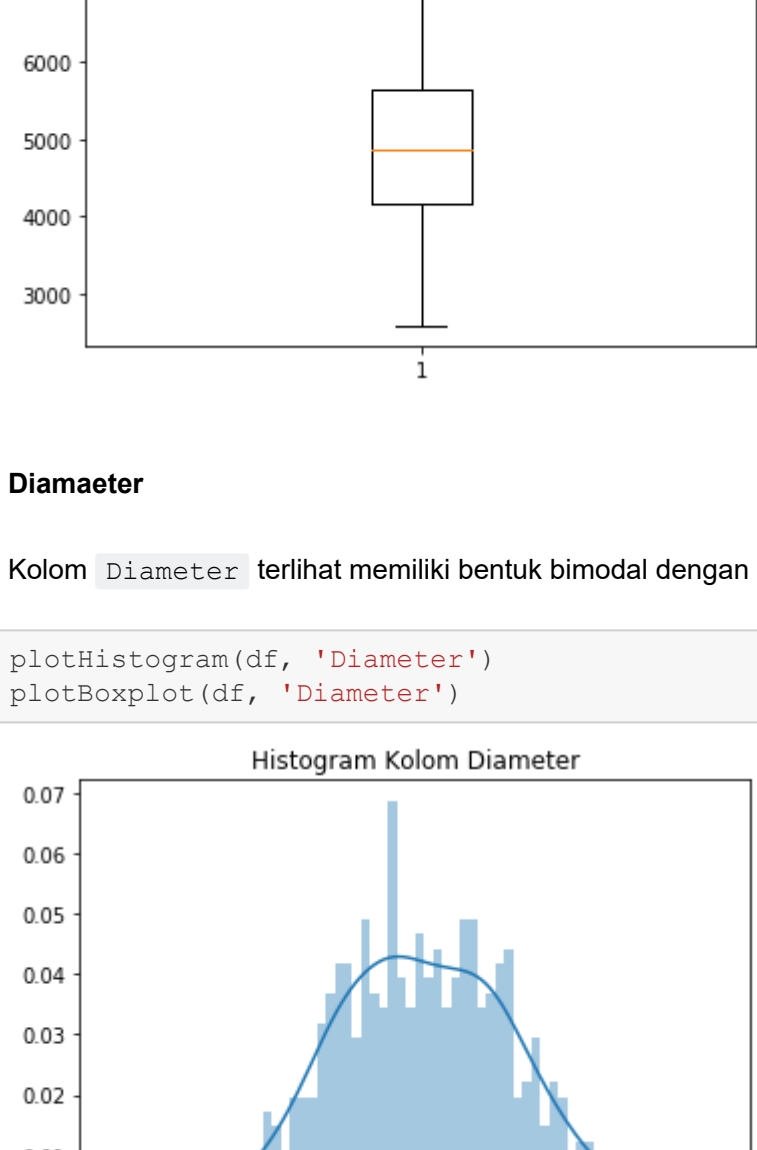
```
In [22]: plotHistogram(df, 'Kelling')
plotBoxplot(df, 'Kelling')
```



Bulatan

Pada plot, dapat dilihat kemungkinan kolom **Bulatan** memiliki kurtosis yang cukup tinggi dan memiliki bentuk skewed-left. Terdapat pula beberapa pencilan pada bagian tail.

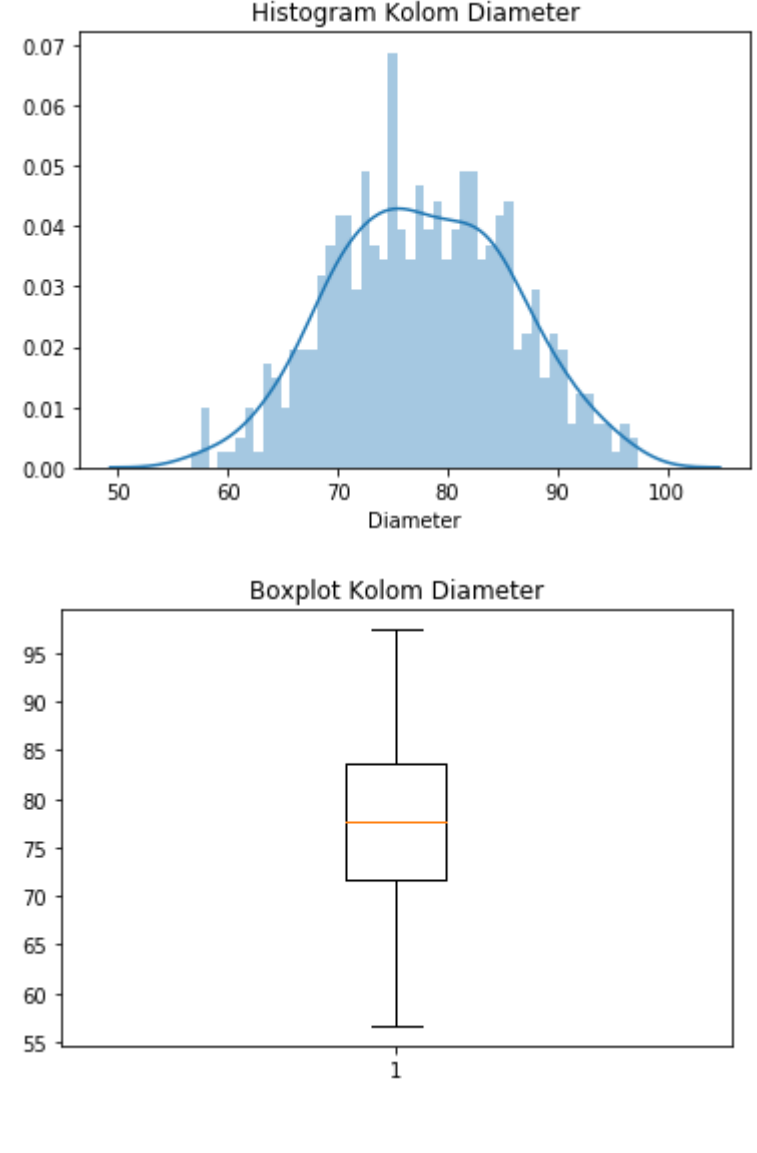
```
In [23]: plotHistogram(df, 'Bulatan')
plotBoxplot(df, 'Bulatan')
```



Ransum

Terlihat jelas bahwa kolom **Ransum** berbentuk skewed-left dengan modus bernilai ~2.4. Range dari kolom ini berkisar pada nilai ~1.

```
In [24]: plotHistogram(df, 'Ransum')
plotBoxplot(df, 'Ransum')
```



3. Menentukan setiap kolom numerik berdistribusi normal atau tidak. Gunakan normality test yang dikaitkan dengan histogram plot.

Normality test yang digunakan adalah Kolmogorov-Smirnov Test yang tersedia pada library scipy dengan fungsi bernama kstest(). Metode penentuan dengan skewness dan kurtosis tidak dilakukan mengingat pengujian dengan metode tersebut hanya akurat jika data berjumlah lebih dari 2000. Metode Kolmogorov-Smirnov yang dilakukan merupakan uji hipotesis dengan

- H_0 : data terdistribusi normal
- H_1 : data tidak terdistribusi normal
- $\alpha = 0.05$

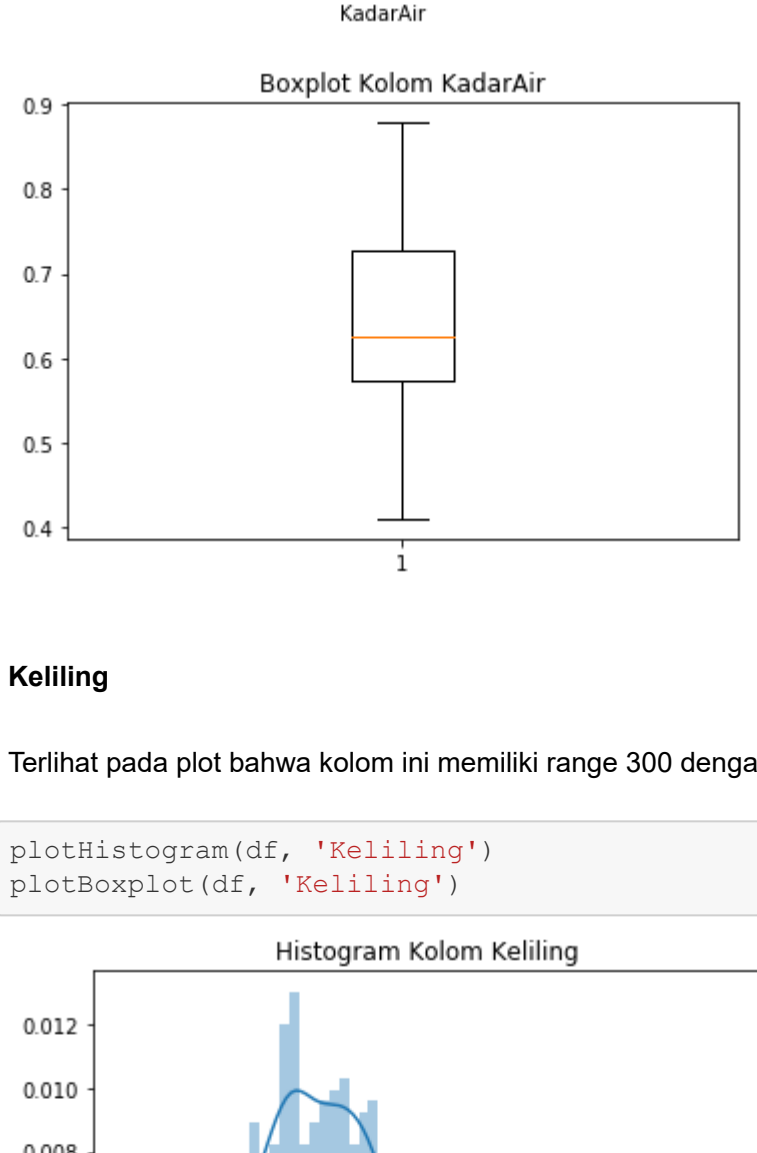
Dengan demikian, jika hasil test lebih besar dari α , maka H_0 gagal ditolak dan data dapat disimpulkan terdistribusi secara normal.

Dari seluruh kolom, semuanya merupakan kolom numerik kecuali kolom **Kelas** yang merupakan kolom kategorik (**Kelas 1** atau **Kelas 2**)

```
In [25]: def checkNormality(df, kolom):
    stat, p = stats.kstest(df[kolom], 'norm')
    print("Nilai p =", p)
    if(p > 0.05):
        print("Kolom " + kolom + " terdistribusi secara normal.")
    else:
        print("Kolom " + kolom + " tidak terdistribusi secara normal.")
```

Daerah

```
In [26]: plotHistogram(df, "Daerah")
checkNormality(df, "Daerah")
```

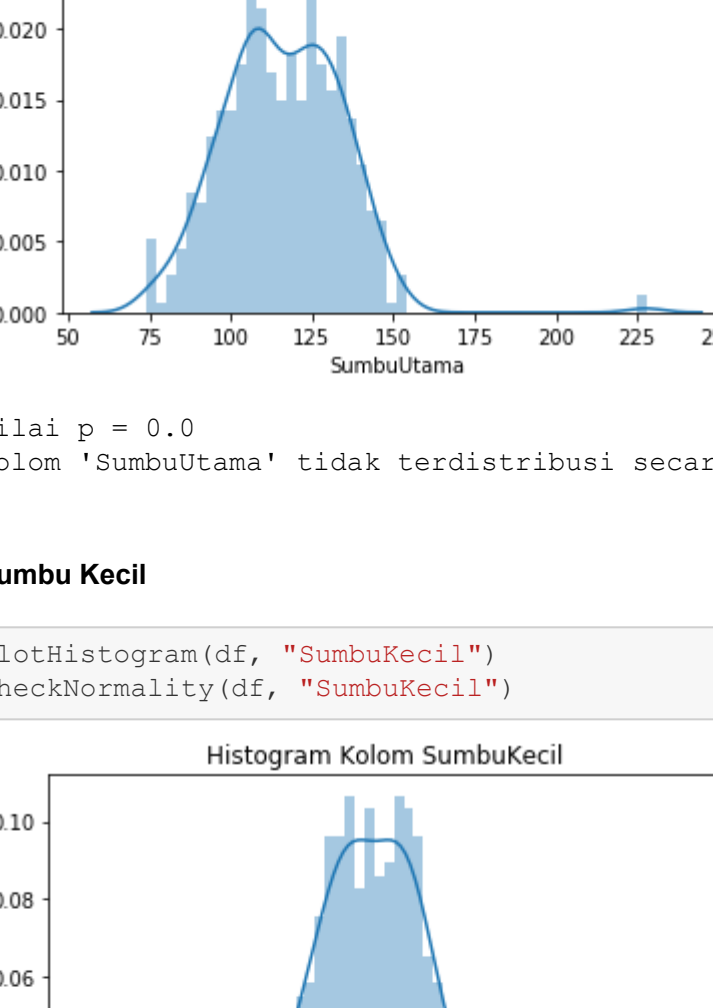


Nilai $p = 0.0$
Kolom 'Daerah' tidak terdistribusi secara normal.

Sumbu Utama

In [27]: plotHistogram(df, "SumbuUtama")
checkNormality(df, "SumbuUtama")

Histogram Kolom SumbuUtama

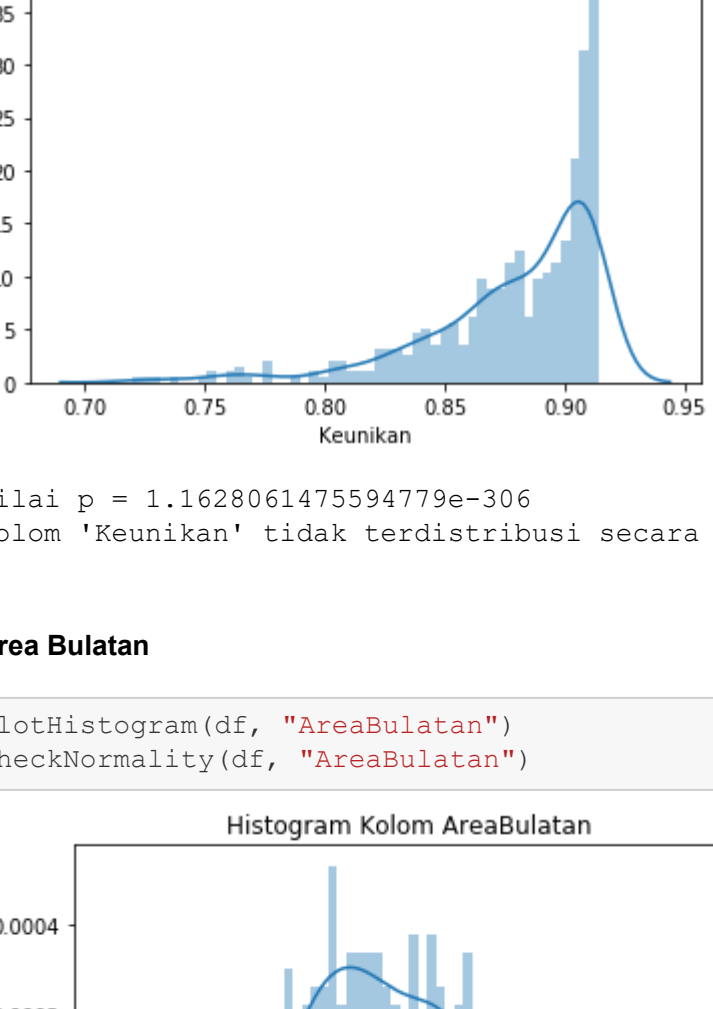


Nilai p = 0.0
Kolom 'SumbuUtama' tidak terdistribusi secara normal.

SumbuKecil

In [28]: plotHistogram(df, "SumbuKecil")
checkNormality(df, "SumbuKecil")

Histogram Kolom SumbuKecil

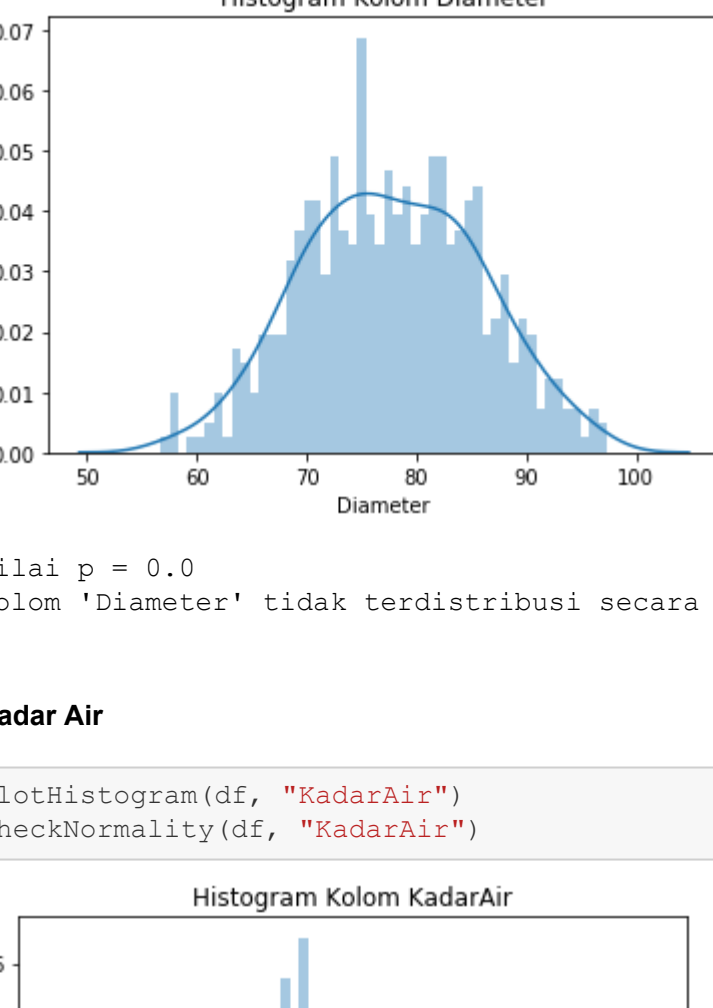


Nilai p = 0.0
Kolom 'SumbuKecil' tidak terdistribusi secara normal.

Keunikan

In [29]: plotHistogram(df, "Keunikan")
checkNormality(df, "Keunikan")

Histogram Kolom Keunikan

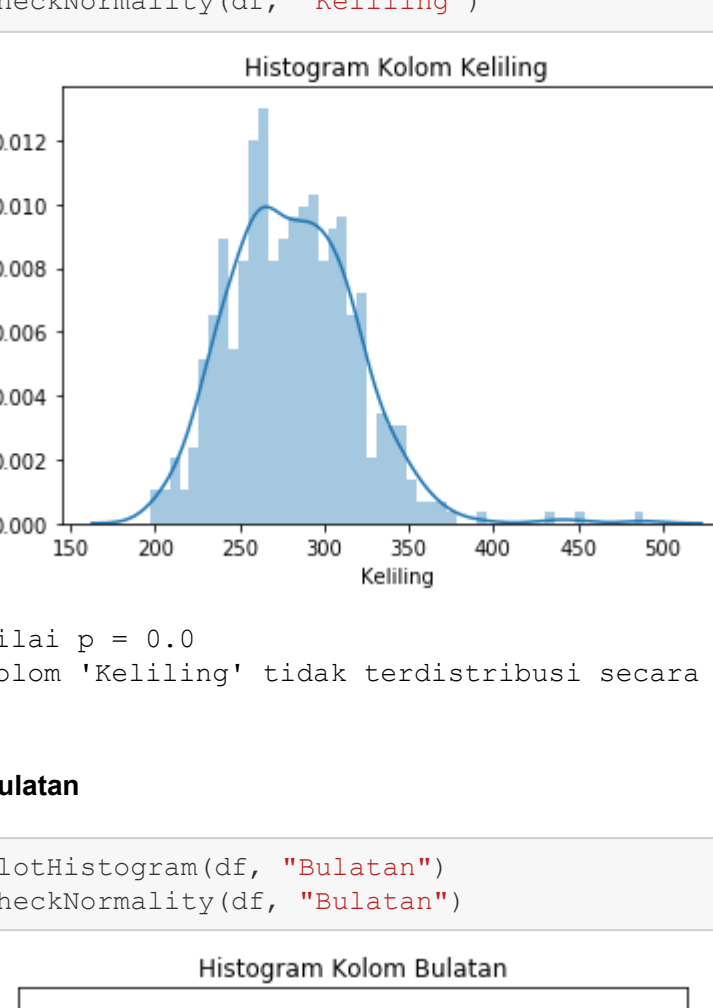


Nilai p = 1.1628061475594779e-306
Kolom 'Keunikan' tidak terdistribusi secara normal.

AreaBulatan

In [30]: plotHistogram(df, "AreaBulatan")
checkNormality(df, "AreaBulatan")

Histogram Kolom AreaBulatan

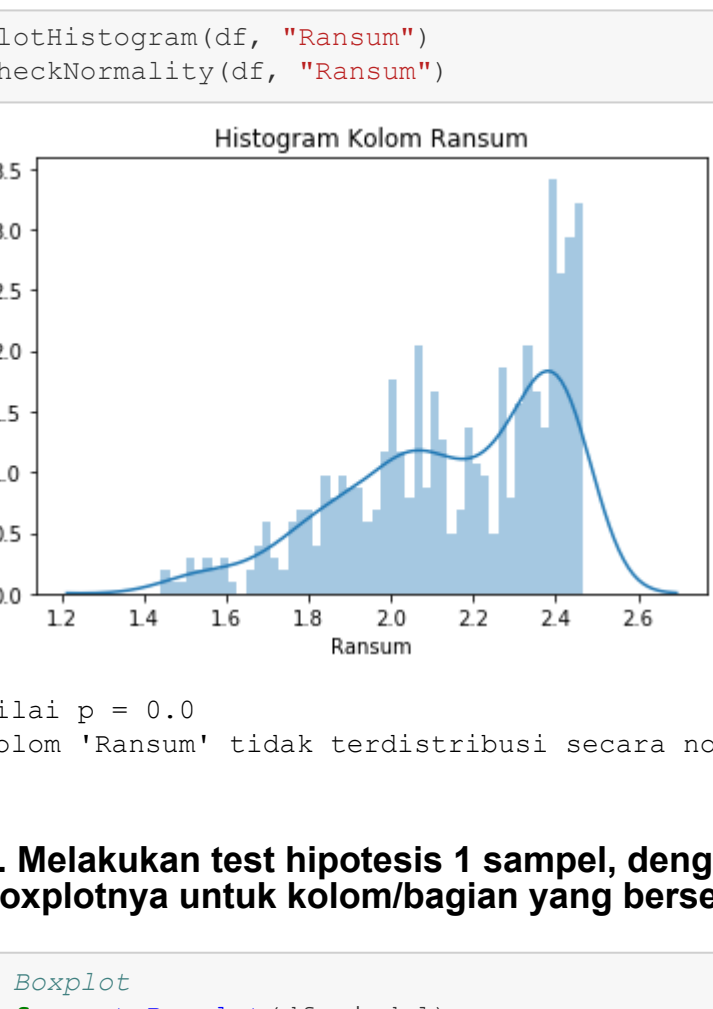


Nilai p = 0.0
Kolom 'AreaBulatan' tidak terdistribusi secara normal.

Diameter

In [31]: plotHistogram(df, "Diameter")
checkNormality(df, "Diameter")

Histogram Kolom Diameter

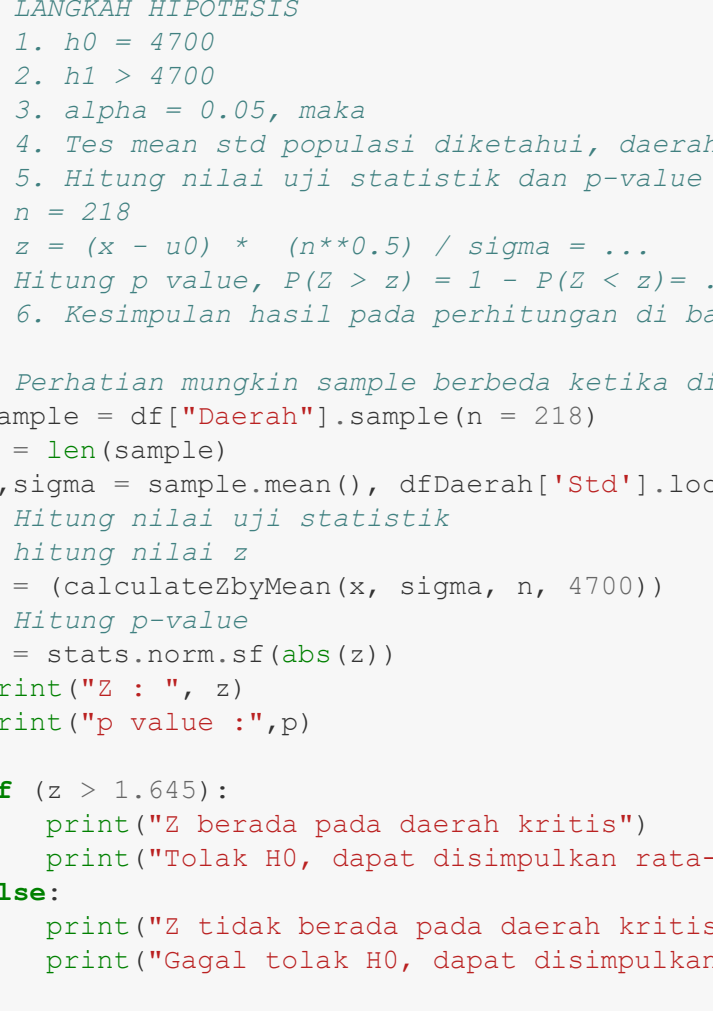


Nilai p = 0.0
Kolom 'Diameter' tidak terdistribusi secara normal.

KadarAir

In [32]: plotHistogram(df, "KadarAir")
checkNormality(df, "KadarAir")

Histogram Kolom KadarAir

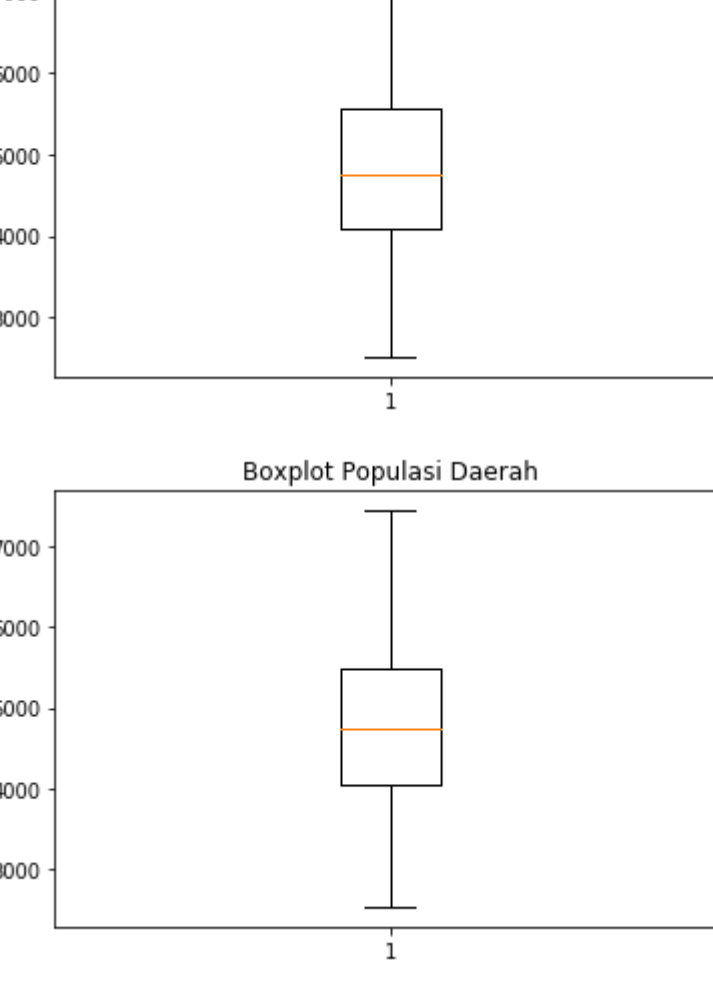


Nilai p = 1.68339436051386e-236
Kolom 'KadarAir' tidak terdistribusi secara normal.

Keliling

In [33]: plotHistogram(df, "Keliling")
checkNormality(df, "Keliling")

Histogram Kolom Keliling

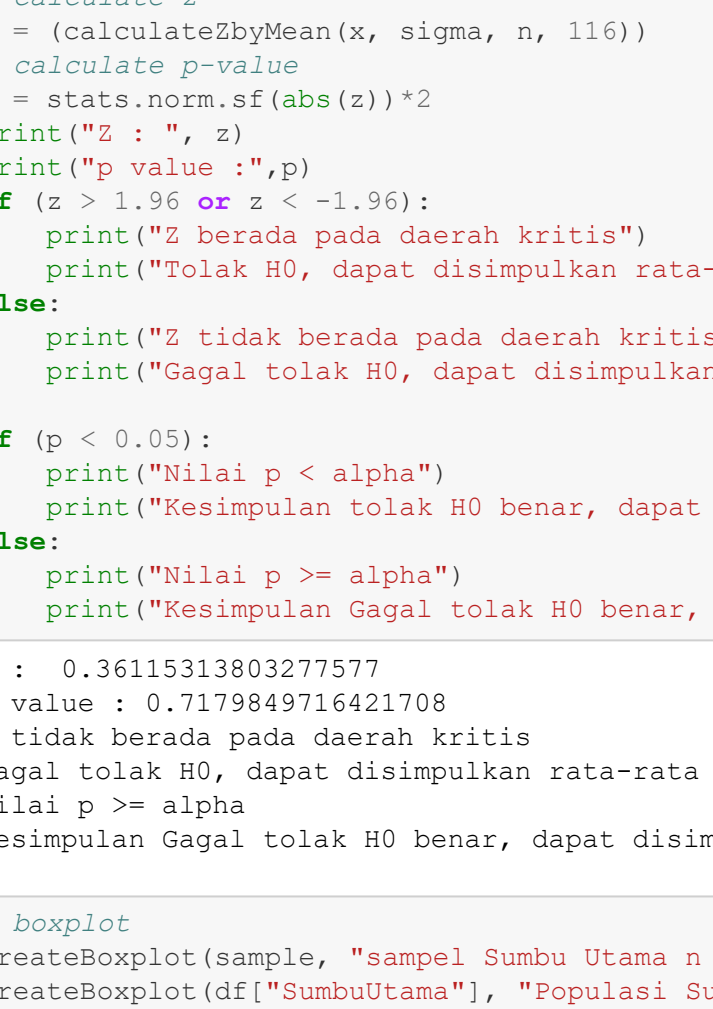


Nilai p = 0.0
Kolom 'Keliling' tidak terdistribusi secara normal.

Bulatan

In [34]: plotHistogram(df, "Bulatan")
checkNormality(df, "Bulatan")

Histogram Kolom Bulatan

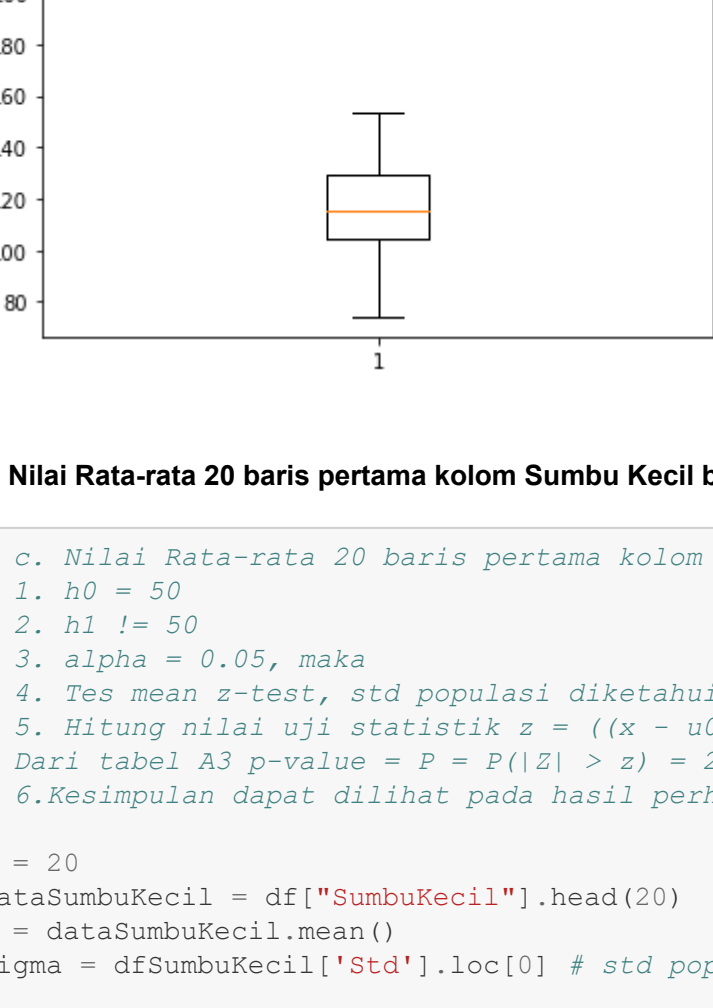


Nilai p = 1.1179956564642067e-269
Kolom 'Bulatan' tidak terdistribusi secara normal.

Ransum

In [35]: plotHistogram(df, "Ransum")
checkNormality(df, "Ransum")

Histogram Kolom Ransum



Nilai p = 0.0
Kolom 'Ransum' tidak terdistribusi secara normal.

4. Melakukan test hipotesis 1 sampel, dengan menuliskan 6 langkah testing dan menampilkan juga boxplotnya untuk kolom/bagian yang berseusulan.

In [36]: # Boxplot
def createBoxplot(df, judul):
 plt.title("Boxplot " + judul)
 plt.boxplot(df)
 plt.show()

def calculateZbyMean(x, sigma, n, u0):
 z = (x - u0) * ((n**0.5)) / sigma
 return z

def getZbyBinomial(x, n, p):
 # 1. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z > 1.96 dan z < -1.96
 # 2. H1 = p < 0.05
 # 3. alpha = 0.05
 # 4. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z > 1.96 dan z < -1.96
 # 5. Hitung nilai uji statistik dan hitung p-value, random sampling n = 218
 # 6. Kesimpulan hasil pada perhitungan di bawah ini
 p = stats.norm.sf(abs(z))
 print("Z : ", z)
 print("p value : ", p)

if (z > 1.96 or z < -1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan rata-rata daerah lebih besar dari 4700")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan rata-rata daerah sama dengan 4700")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata daerah lebih besar dari 4700")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata daerah sama dengan 4700")

Z : 1.9771415250471962
p value : 0.02401281755940076
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan rata-rata daerah lebih besar dari 4700
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata daerah lebih besar dari 4700

In [37]: # n = 218, untuk populasi 500, margin of error 5%

a. Nilai rata-rata Daerah di atas 4700?

In [38]: # LANGKAH HIPOTESIS
1. H0 = 4700
2. H1 = 4700
3. alpha = 0.05, maka
4. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z > 1.96 dan z < -1.96
5. Hitung nilai uji statistik dan hitung p-value, random sampling n = 218
6. Kesimpulan hasil pada perhitungan di bawah ini
6. Berlanjut pada hasil dibawah

Perhatian mungkin sample berbeda ketika di run ulang
sample = df["Daerah"].sample(n = 218)
n = len(sample)
x, sigma = sample.mean(), df["Daerah"].std().loc[0]
Hitung nilai uji statistik
Hitung nilai z
z = (calculateZbyMean(x, sigma, n, 4700))
Hitung p-value
p = stats.norm.sf(abs(z))
print("Z : ", z)
print("p value : ", p)

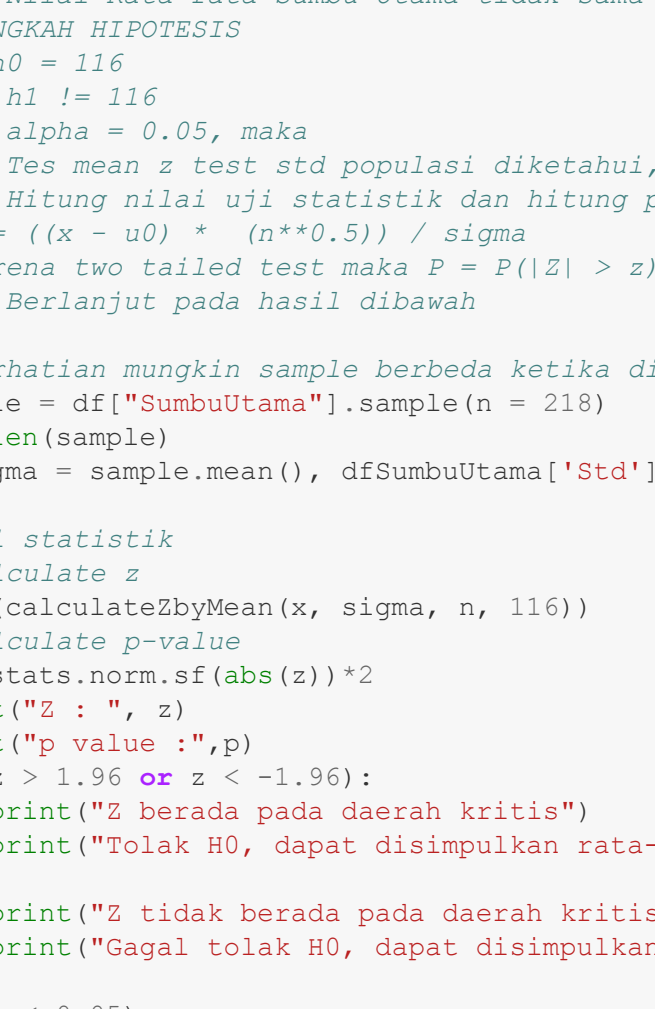
if (z > 1.96 or z < -1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan rata-rata daerah lebih besar dari 4700")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan rata-rata daerah sama dengan 4700")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata daerah lebih besar dari 4700")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata daerah sama dengan 4700")

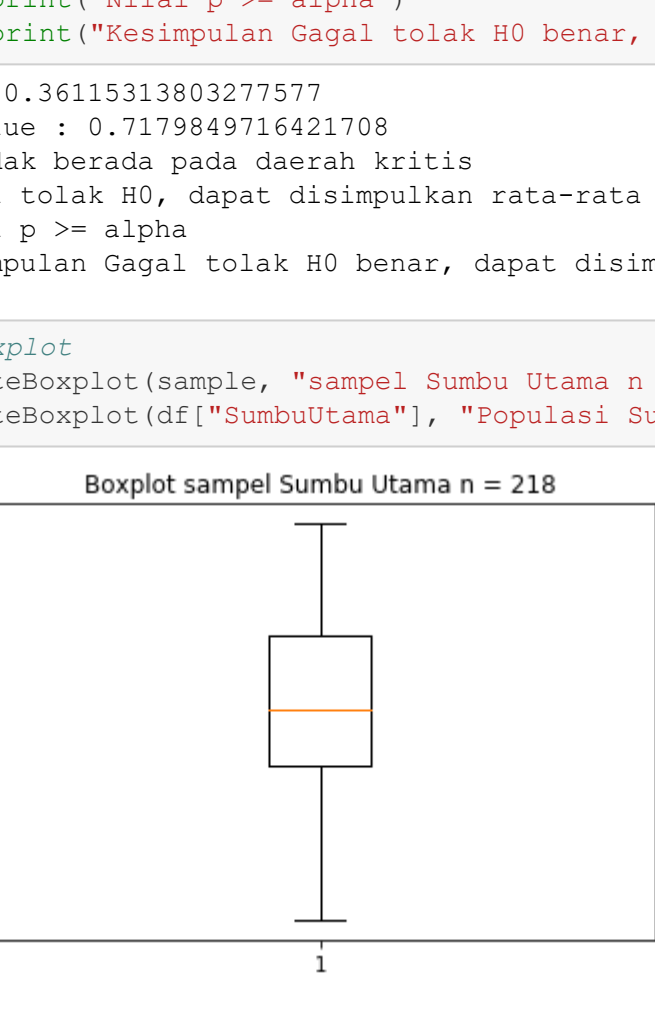
Z : 1.9771415250471962
p value : 0.02401281755940076
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan rata-rata daerah lebih besar dari 4700
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata daerah lebih besar dari 4700

In [39]: # boxplot
createBoxplot(sample, "Random Sampel Daerah n = 218")
createBoxplot(df["Daerah"], "Populasi Daerah")

Boxplot Random Sampel Daerah n = 218



Boxplot Populasi Daerah



b. Nilai Rata-rata Sumbu Utama tidak sama dengan 116?

In [40]: # b. Nilai Rata-rata Sumbu Utama tidak sama dengan 116?
LANGKAH HIPOTESIS
1. H0 = 116
2. H1 = 116
3. alpha = 0.05, maka
4. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z > 1.96 dan z < -1.96
5. Hitung nilai uji statistik dan hitung p-value, random sampling n = 218
6. Kesimpulan hasil pada perhitungan di bawah ini
6. Berlanjut pada hasil dibawah

Perhatian mungkin sample berbeda ketika di run ulang
sample = df["SumbuUtama"].sample(n = 218)
n = len(sample)
x, sigma = sample.mean(), df["SumbuUtama"].std().loc[0]
Uji statistik
calculate z
z = (calculateZbyMean(x, sigma, n, 116))
calculate p-value
p = stats.norm.sf(abs(z))
print("Z : ", z)
print("p value : ", p)

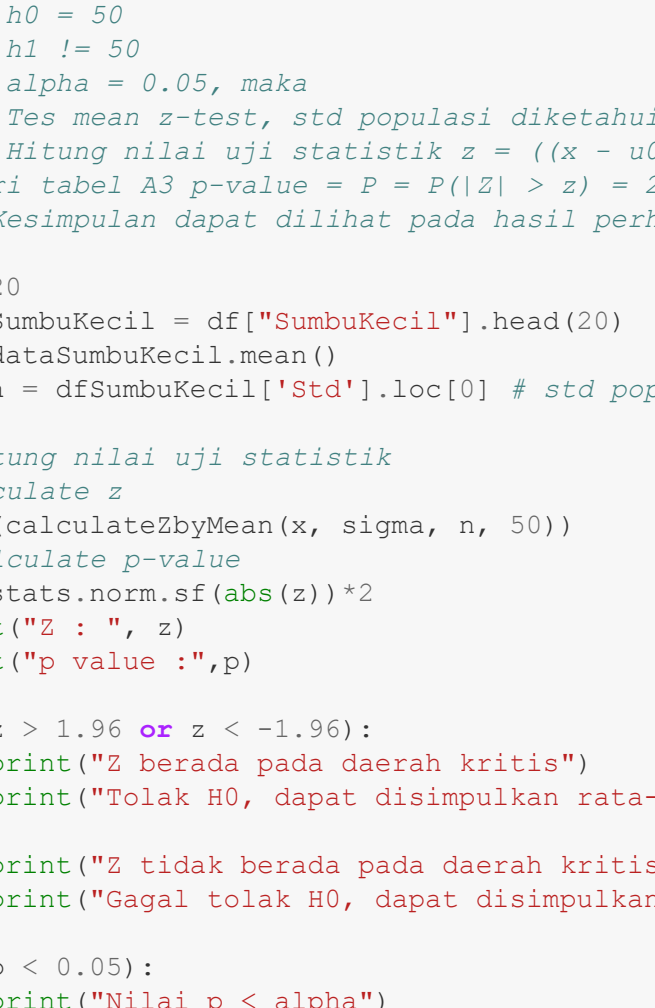
if (z > 1.96 or z < -1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan rata-rata sumbu utama tidak sama dengan 116")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan rata-rata sumbu utama sama dengan 116")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata sumbu utama tidak sama dengan 116")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata sumbu utama sama dengan 116")

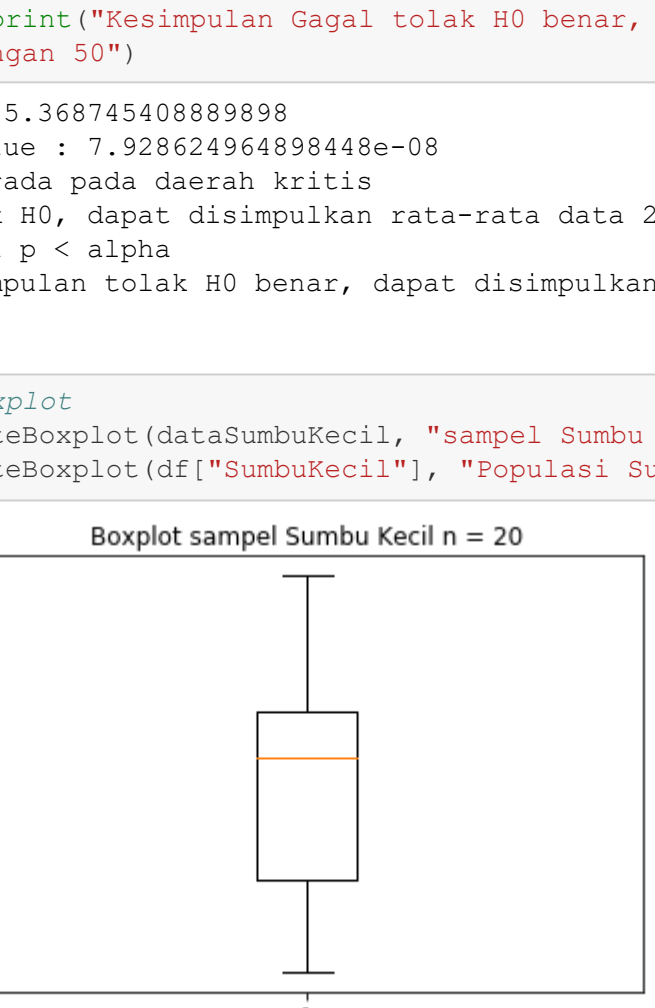
Z : 0.3611511803275777
p value : 0.7179949716421708
Z tidak berada pada daerah kritis
Gagal tolak H0, dapat disimpulkan rata-rata sumbu utama sama dengan 116
Nilai p >= alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata sumbu utama sama dengan 116

In [41]: # boxplot
createBoxplot(sample, "Sampel Sumbu Utama n = 218")
createBoxplot(df["SumbuUtama"], "Populasi Sumbu Utama")

Boxplot sampel Sumbu Utama n = 218



Boxplot Populasi Sumbu Utama



c. Nilai Rata-rata 20 baris pertama kolom Sumbu Kecil bukan 50?

In [42]: # c. Nilai Rata-rata 20 baris pertama kolom Sumbu Kecil bukan 50?
1. H0 = 50
2. H1 = 50
3. alpha = 0.05, maka
4. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z > 1.96 dan z < -1.96
5. Hitung nilai uji statistik dan hitung p-value, random sampling n = 218
6. Kesimpulan hasil pada perhitungan di bawah ini
6. Berlanjut pada hasil dibawah

n = 20
dataSumbuKecil = df["SumbuKecil"].head(20)
x = dataSumbuKecil.mean()
sigma = dfSumbuKecil["Std"].loc[0] # std populasi

Hitung nilai uji statistik
calculate z
z = (calculateZbyMean(x, sigma, n, 50))
calculate p-value
p = stats.norm.sf(abs(z))
print("Z : ", z)
print("p value : ", p)

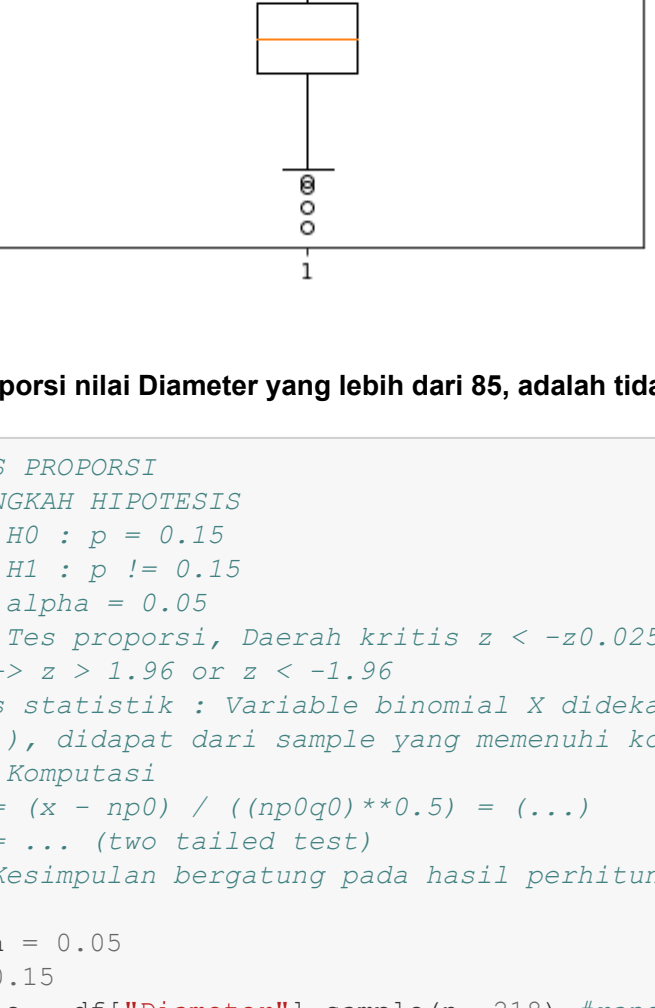
if (z > 1.96 or z < -1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan rata-rata data 20 sumbu kecil pertama tidak sama dengan 50")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan rata 20 sumbu kecil pertama sama dengan 50")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata data 20 sumbu kecil pertama tidak sama dengan 50")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata data 20 sumbu kecil pertama sama dengan 50")

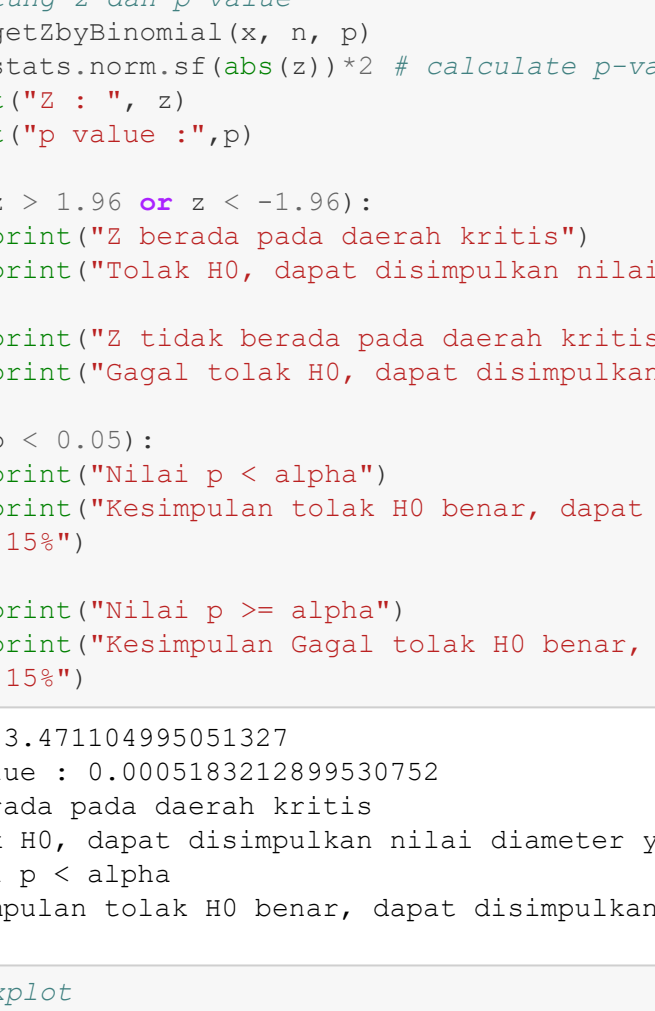
Z : 5.3687454088989898
p value : 7.928624964898448e-08
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan rata-rata data 20 sumbu kecil pertama tidak sama dengan 50
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata data 20 sumbu kecil pertama tidak sama dengan 50

In [43]: # boxplot
createBoxplot(dataSumbuKecil, "Sampel Sumbu Kecil n = 20")
createBoxplot(df["SumbuKecil"], "Populasi Sumbu Kecil")

Boxplot sampel Sumbu Kecil n = 20



Boxplot Populasi Sumbu Kecil



d. Proporsi nilai Diameter yang lebih dari 85, adalah tidak sama dengan 15%?

In [44]: # TES PROPORSI
LANGKAH HIPOTESIS
1. H0 : p = 0.15
2. H1 : p < 0.15
3. alpha = 0.05
4. Tes proporsi, Daerah kritis z < -z(0.05) or z > z(0.05)
--> z > 1.96 or z < -1.96
Tes statistik : Variable binomial X didekati normal dengan p = 0.15 dan random sampling n = 218, x =
5. Komputasi
6. Kesimpulan bergantung pada hasil perhitungan dibawah ini

alpha = 0.05
p = 0.15
sample = df["Diameter"].sample(n= 218) #random sampling n = 218
n = len(sample)
Jumlah data Diameter > 85 = x
x = len(sample.loc[sample > 85])
hitung s dan p-value
s = getZbyBinomial(x, n, p)
p = stats.norm.sf(abs(z)) # calculate p-value
print("Z : ", z)
print("p value : ", p)

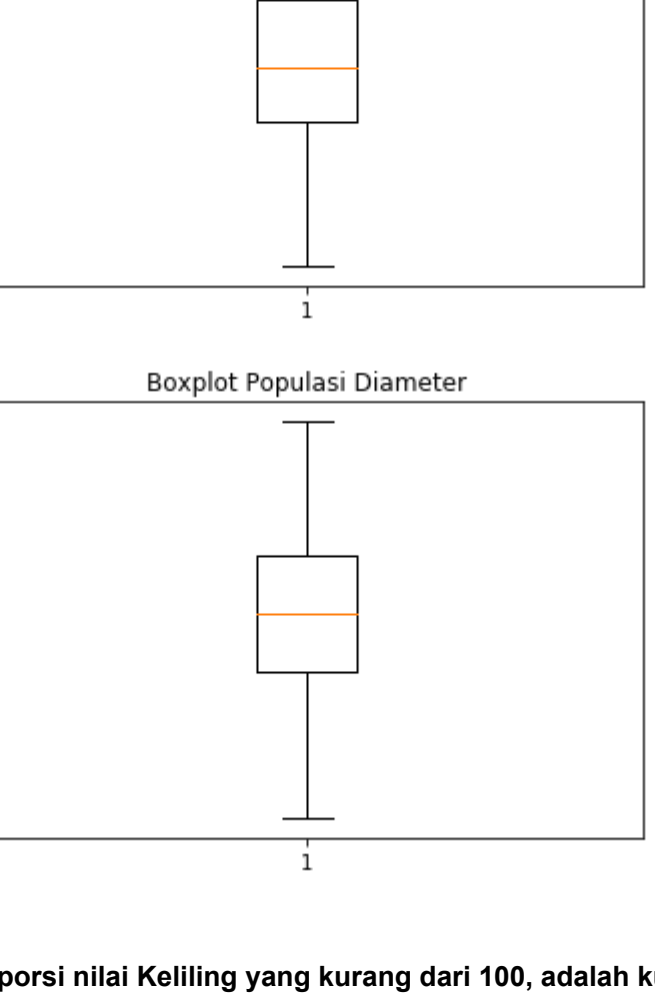
if (z > 1.96 or z < -1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan nilai Diameter yang lebih dari 85 tidak sama dengan 15")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan Nilai diameter yang lebih dari 85 sama dengan 15")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan Nilai diameter yang lebih dari 85 tidak sama dengan 15")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan Nilai diameter yang lebih dari 85 sama dengan 15")

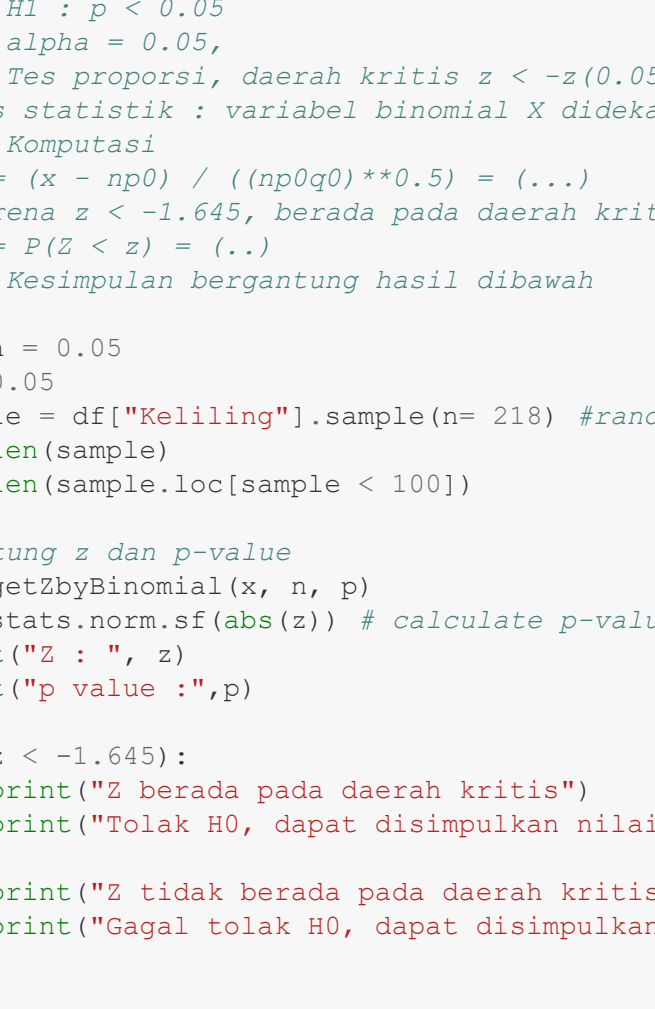
Z : 3.471104995051327
p value : 0.0005183212899530752
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan nilai diameter yang lebih dari 85 tidak sama dengan 15
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan Nilai diameter yang lebih dari 85 tidak sama dengan 15

In [45]: # boxplot
createBoxplot(sample, "Random sampling Diameter n = 218")
createBoxplot(df["Diameter"], "Populasi Diameter")

Boxplot Random sampling Diameter n = 218



Boxplot Populasi Diameter



e. Proporsi nilai Keliling yang kurang dari 100, adalah kurang dari 5%?

In [46]: # TES PROPORSI
LANGKAH HIPOTESIS
1. H0 : p = 0.15
2. H1 : p < 0.05
3. alpha = 0.05
4. Tes proporsi, daerah kritis z < -z(0.05) = -1.645
Tes statistik : Variable binomial X didekati normal dengan p = 0.05, n = 218, x = 0
5. Komputasi
6. Kesimpulan bergantung hasil dibawah

alpha = 0.05
p = 0.05
sample = df["Keliling"].sample(n= 218) #random sampling n = 218
n = len(sample)
x = len(sample.loc[sample < 100])
hitung z dan p-value
z = getZbyBinomial(x, n, p)
p = stats.norm.sf(abs(z)) # calculate p-value
print("Z : ", z)
print("p value : ", p)

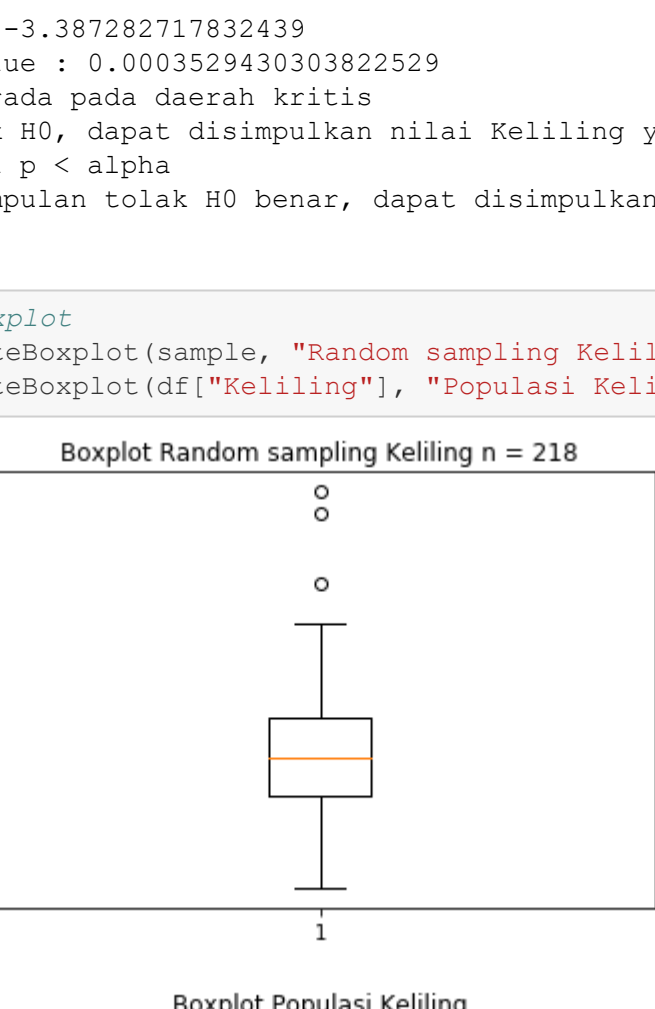
if (z < -1.645):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan nilai Keliling yang kurang dari 100, adalah kurang dari 5")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan nilai Keliling yang kurang dari 100, adalah sama dengan 5")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan nilai Keliling yang kurang dari 100, adalah kurang dari 5")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan nilai Keliling yang kurang dari 100, adalah sama dengan 5")

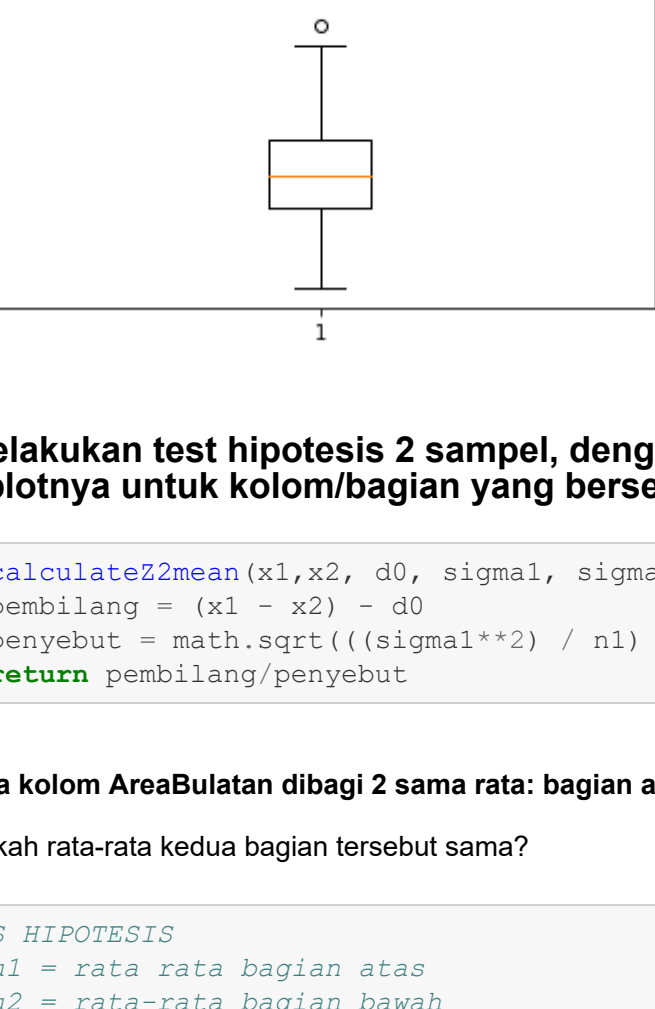
Z : -3.387282717832439
p value : 0.0003529430303822529
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan nilai Keliling yang kurang dari 100, adalah kurang dari 5
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan nilai Keliling yang kurang dari 100, adalah kurang dari 5

In [47]: # boxplot
createBoxplot(sample, "Random sampling Keliling n = 218")
createBoxplot(df["Keliling"], "Populasi Keliling")

Boxplot Random sampling Keliling n = 218



Boxplot Populasi Keliling



5. Melakukan test hipotesis 2 sampel, dengan menuliskan 6 langkah testing dan menampilkan juga boxplotnya untuk kolom/bagian yang berseusulan.

In [49]: def calculateZ2mean(x1,x2, d0, sigma1, sigma2, n1, n2):
 pembilang = (x1 - x2) - d0
 penyebut = math.sqrt(((sigma1**2) / n1) + ((sigma2**2) / n2))
 return pembilang/penyebut

a. Data kolom AreaBulatan dibagi 2 sama rata: bagian awal dan bagian akhir kolom. Benarkah rata-rata kedua bagian tersebut sama?

In [50]: # TES HIPOTESIS
mu1 = rata rata bagian atas
mu2 = rata-rata bagian bawah
1. H0 : mu1 - mu2 = 0
2. H1 : mu1 - mu2 != 0
3. alpha = 0.05
4. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z < -z(0.05) or z > z(0.05)
5. Hitung nilai z dan p-value
6. Kesimpulan bergantung hasil dibawah ini

Perhatian mungkin sample berbeda ketika di run ulang
sample1 = df["AreaBulatan"].head(250)
sample2 = df["AreaBulatan"].tail(250)
d0 = 0
sigma1 = sample1.std()
sigma2 = sample2.std()
Hitung nilai uji statistik
Hitung nilai z
z = (calculateZ2mean(x1,x2, d0, sigma1, sigma2, n1, n2))
calculate p-value
p = stats.norm.sf(abs(z))
print("Z : ", z)
print("p value : ", p)

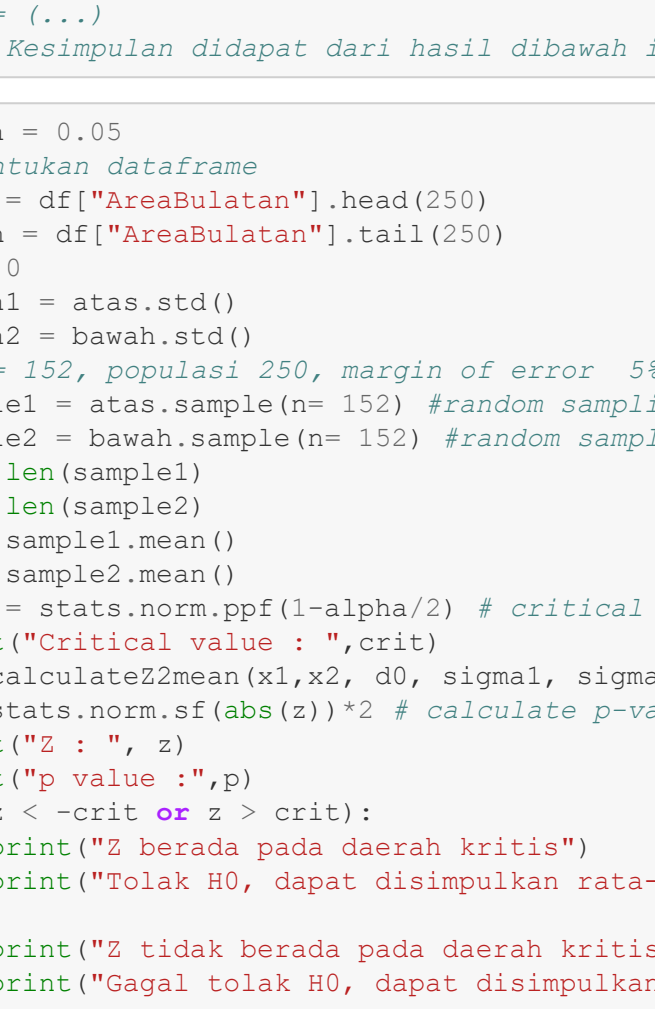
if (z < -1.96 or z > 1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan rata-rata kedua bagian tersebut tidak sama")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan rata-rata kedua bagian tersebut sama")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata kedua bagian tersebut tidak sama")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata kedua bagian tersebut sama")

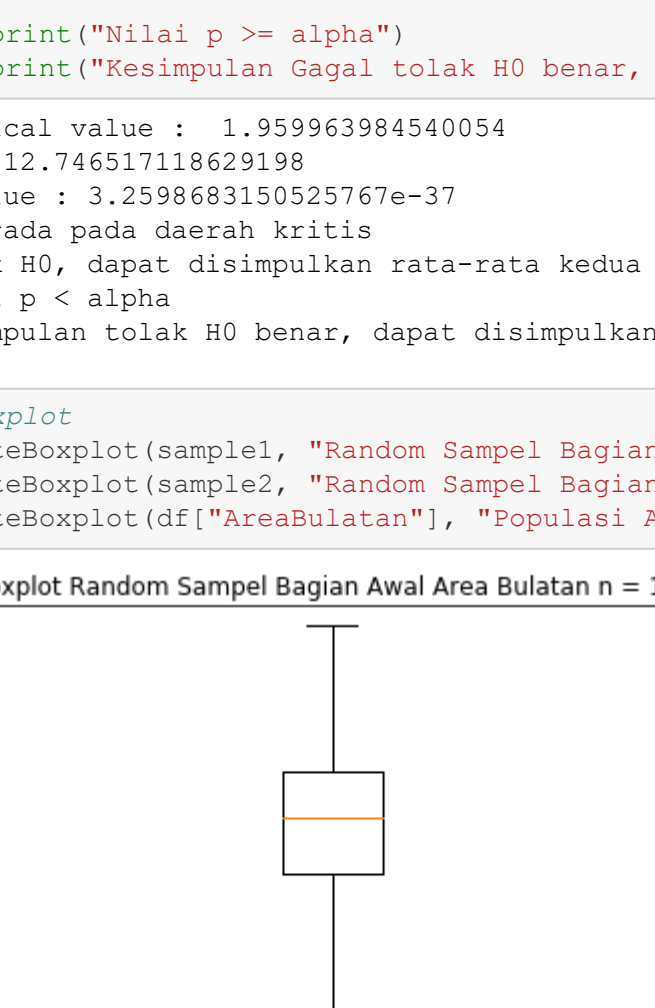
Critical value : 1.959963984540054
Z : 12.746517118629198
p value : 3.259868150525767e-37
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan rata-rata kedua bagian tersebut tidak sama
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata kedua bagian tersebut tidak sama

In [52]: # boxplot
createBoxplot(sample1, "Random Sampel Bagian Awal Area Bulatan n = 152")
createBoxplot(sample2, "Random Sampel Bagian Akhir Area Bulatan n = 152")
createBoxplot(df["AreaBulatan"], "Populasi Area Bulatan")

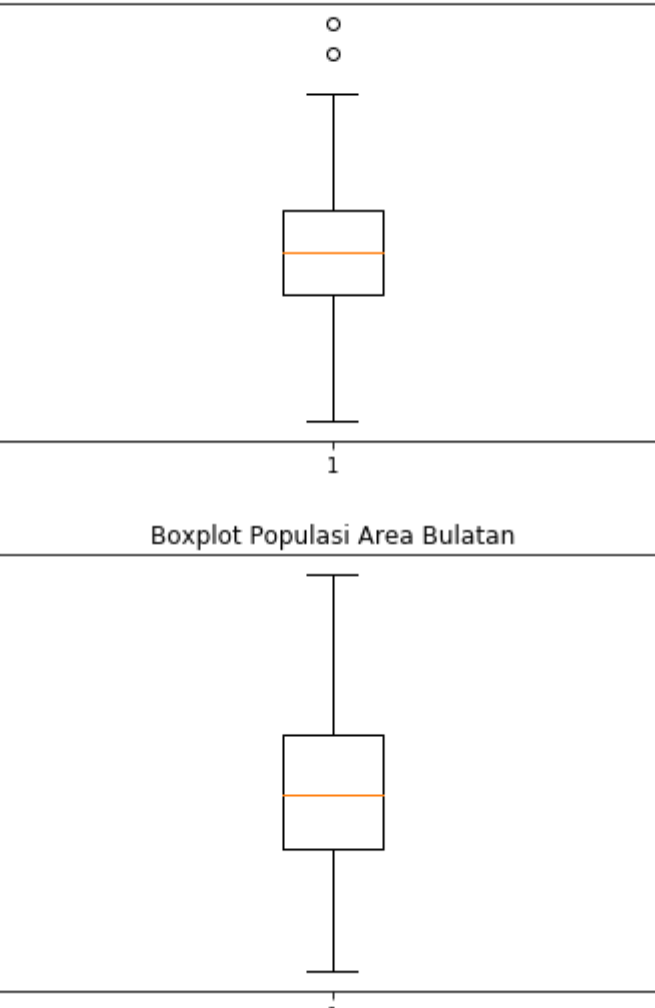
Boxplot Random Sampel Bagian Awal Area Bulatan n = 152



Boxplot Random Sampel Bagian Akhir Area Bulatan n = 152



Boxplot Populasi Area Bulatan



b. Data kolom Kadar Air dibagi 2 sama rata: bagian awal dan bagian akhir kolom. Benarkah rata-rata bagian awal lebih besar dari pada bagian akhir sebesar 0.2?

In [53]: # TES HIPOTESIS
mu1 = rata rata bagian atas
mu2 = rata-rata bagian bawah
1. H0 : mu1 - mu2 = 0.2
2. H1 : mu1 - mu2 != 0.2
3. alpha = 0.05
4. Tes mean z-test, std populasi diketahui, two tailed test daerah kritis z < -z(0.05) or z > z(0.05)
5. Hitung nilai z dan p-value
6. Kesimpulan bergantung hasil dibawah ini

Perhatian mungkin sample berbeda ketika di run ulang
sample1 = df["KadarAir"].head(250)
sample2 = df["KadarAir"].tail(250)
d0 = 0.2
sigma1 = sample1.std()
sigma2 = sample2.std()
Hitung nilai uji statistik
Hitung nilai z
z = (calculateZ2mean(x1,x2, d0, sigma1, sigma2, n1, n2))
calculate p-value
p = stats.norm.sf(abs(z))
print("Z : ", z)
print("p value : ", p)

if (z < -1.96 or z > 1.96):
 print("Z berada pada daerah kritis")
 print("Tolak H0, dapat disimpulkan rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.2")
else:
 print("Z tidak berada pada daerah kritis")
 print("Gagal tolak H0, dapat disimpulkan rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.2")

if (p < 0.05):
 print("Nilai p < alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.2")
else:
 print("Nilai p >= alpha")
 print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.2")

Critical value : 1.959963984540054
Z : 12.746517118629198
p value : 3.259868150525767e-37
Z berada pada daerah kritis
Tolak H0, dapat disimpulkan rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.2
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata bagian awal lebih besar dari bagian akhir sebesar 0.2


```
[54]: alpha = 0.05
# Tentukan dataframe
sigma1 = stats.std()
sigma2 = df["KadarAir"].head(250)
bawah = df["KadarAir"].tail(250)
d0 = 0.2

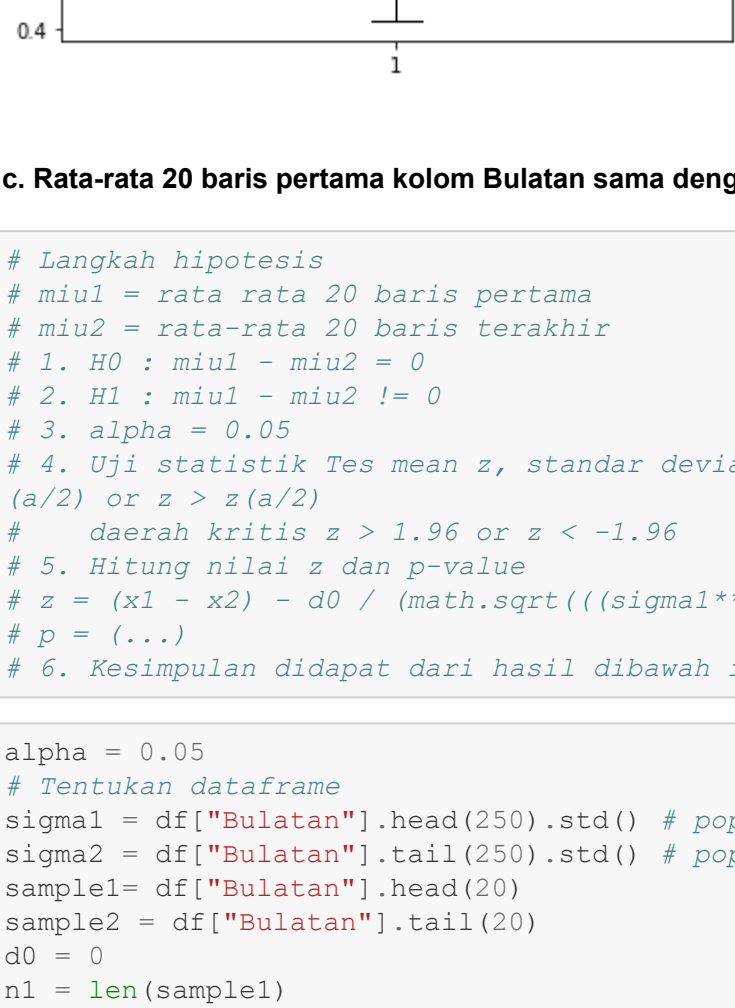
n1 = len(sample1)
n2 = len(sample2)
x1 = sample1.mean()
x2 = sample2.mean()

crit = stats.norm.ppf(1-(alpha/2)) # critical value
print("Critical value :",crit)
z = calculateZ2mean(x1,x2, d0, sigma1, sigma2, n1, n2)
p = stats.norm.sf(abs(z))*2 # calculate p-value
print("P value :",p)
if (z < -crit or z > crit):
    print("Tolak H0, dapat disimpulkan rata-rata bagian awal lebih besar daripada bagian akhir tidak se benar 0.2")
else:
    print("Tolak H0, dapat disimpulkan rata-rata bagian awal lebih besar daripada bagian akhir se benar 0.2")

if (p < 0.05):
    print("Nilai p < alpha")
    print("Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata bagian awal lebih besar daripada bagian akhir tidak sebesar 0.2")
else:
    print("Nilai p > alpha")
    print("Kesimpulan tolak H0 benar, rata-rata bagian awal lebih besar daripada bagian akhir sebesar 0.2")

Critical value : 1.959963984540054
Z : -1.8186931775787083
p value : 0.06895825316348737
Tolak H0, dapat disimpulkan rata-rata bagian awal lebih besar daripada bagian akhir tidak sebesar 0.2
Nilai p < alpha
Kesimpulan tolak H0 benar, dapat disimpulkan rata-rata bagian awal lebih besar daripada bagian akhir tidak sebesar 0.2
```

```
In [55]: # boxplot
createBoxplot(sample1, "Random Sampel Bagian Awal Kadar Air n = 152")
createBoxplot(sample2, "Random Sampel Bagian Akhir Kadar Air n = 152")
createBoxplot(df["KadarAir"], "Populasi Kadar Air")
```



c. Rata-rata 20 baris pertama kolom Bulatan sama dengan 20 baris terakhirnya?

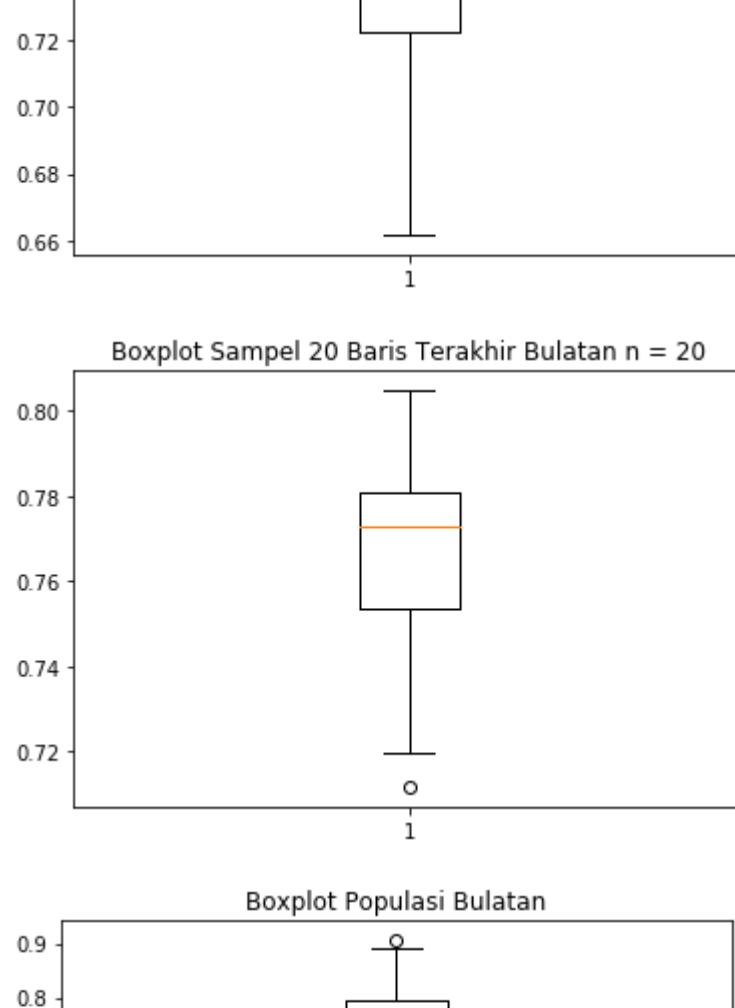
```
In [56]: # Langkah hipotesis
# mu1 = rata-rata 20 baris pertama
# mu2 = rata-rata 20 baris terakhir
# 1. H0 : mu1 = mu2 = 0
# 2. H1 : mu1 - mu2 != 0
# 3. alpha = 0.05
# 4. Uji statistik Tes mean z, standar deviasi populasi diketahui, two tailed test daerah kritis z < -z
# 5. Hitung nilai z dan p-value
# z = (x1 - x2) - d0 / (math.sqrt(((sigma1**2) / n1) + ((sigma2**2) / n2))) = (...)
# p = (...)
# 6. Kesimpulan didapat dari hasil dibawah ini

alpha = 0.05
# Tentukan dataframe
sigma1 = df["Bulatan"].head(250).std() # populasi atas
sigma2 = df["Bulatan"].tail(250).std() # populasi bawah
sample1 = df["Bulatan"].head(20)
sample2 = df["Bulatan"].tail(20)
d0 = 0
n1 = len(sample1)
n2 = len(sample2)
x1 = sample1.mean()
x2 = sample2.mean()
# Hitung titik kritis
crit = stats.norm.ppf(1-(alpha/2)) # critical value
print("Critical value :",crit)

# Hitung z dan p-value
p1 = calculateZ2mean(x1,x2, d0, sigma1, sigma2, n1, n2)
p = stats.norm.sf(abs(z))*2 # calculate p-value
print("P value :",p)
print("P value :",p)
# Kesimpulan
if (z < -crit or z > crit):
    print("Tolak H0, dapat disimpulkan rata-rata 20 baris pertama Bulatan tidak sama dengan 20 baris terakhir")
else:
    print("Tolak H0, dapat disimpulkan rata-rata 20 baris pertama sama dengan rata-rata 20 baris terakhir")
    print("Gagal tolak H0, dapat disimpulkan rata-rata 20 baris pertama sama dengan rata-rata 20 baris terakhir")

Critical value : 1.959963984540054
Z : -1.8186931775787083
p value : 0.06895825316348737
Tolak H0, dapat disimpulkan rata-rata 20 baris pertama sama dengan rata-rata 20 baris terakhir
```

```
In [62]: # boxplot
createBoxplot(sample1, "Sampel 20 Baris Pertama Bulatan n = 20")
createBoxplot(sample2, "Sampel 20 Baris Terakhir Bulatan n = 20")
createBoxplot(df["Bulatan"], "Populasi Bulatan")
```



d. Proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Ransum?

```
In [63]: # TES HIPOTESIS
# s1^2 = variansi bagian awal
# s2^2 = variansi bagian akhir
# 1. H0 : s1^2 = s2^2
# 2. H1 : s1^2 > s2^2
# 3. alpha = 0.05
# 4. Normal, z. Critical region z>=alpha.
# 5. Hitung uji statistik dan p-value
# z = (p1 - p2) / (p*(1-p)*(1/250 + 1/250))**0.5
# p = (...)

# Kesimpulan
crit = stats.norm.isf(0.05)
p1 = (sample1>2).sum()/250
p2 = (sample2>2).sum()/250
p = ((sample1>2).sum() + (sample2>2).sum())/500

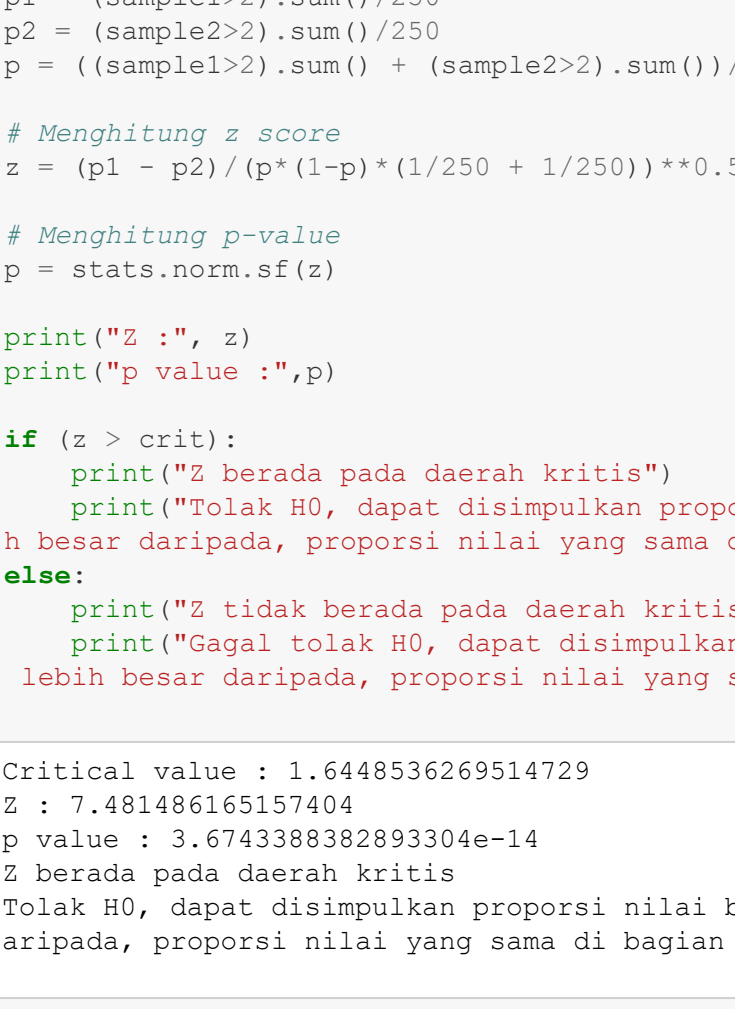
# Menghitung z score
z = (p1 - p2) / (p*(1-p)*(1/250 + 1/250))**0.5

# Menghitung p-value
p = stats.norm.sf(z)

print("P value :",p)
print("P value :",p)
if (z > crit):
    print("Tolak H0, dapat disimpulkan proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Ransum")
else:
    print("Tolak H0, dapat disimpulkan proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Ransum")
    print("Gagal tolak H0, dapat disimpulkan proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Ransum")

Critical value : 1.6448536269514729
Z : 7.481486165157404
p value : 0.00834937751160385
Tolak H0, dapat disimpulkan proporsi nilai bagian awal Ransum yang lebih dari 2, adalah lebih besar daripada, proporsi nilai yang sama di bagian akhir Ransum
```

```
In [65]: # boxplot
createBoxplot(sample1, "Random Sampel Bagian Awal Ransum n = 152")
createBoxplot(sample2, "Random Sampel Bagian Akhir Ransum n = 152")
createBoxplot(df["Ransum"], "Populasi Diameter")
```



e. Bagian awal kolom Diameter memiliki variansi yang sama dengan bagian akhirnya?

```
In [75]: # TES HIPOTESIS
# s1^2 = variansi bagian awal
# s2^2 = variansi bagian akhir
# 1. H0 : s1^2 = s2^2
# 2. H1 : s1^2 > s2^2
# 3. alpha = 0.05
# 4. distribution, uji statistik f dan p-value
# f = s1^2/s2^2
# p-value = P(F < f) = (...)
# 5. Kesimpulan
n1 = 250
n2 = 250
v1 = n1-1
v2 = n2-1

# Mengambil sample dan nilai variansi sample
sample1 = df["Diameter"].head(n1).sample(n=152)
sample2 = df["Diameter"].tail(n2).sample(n=152)
s1_2 = sample1.var()
s2_2 = sample2.var()

# nilai f
f = s1_2/s2_2

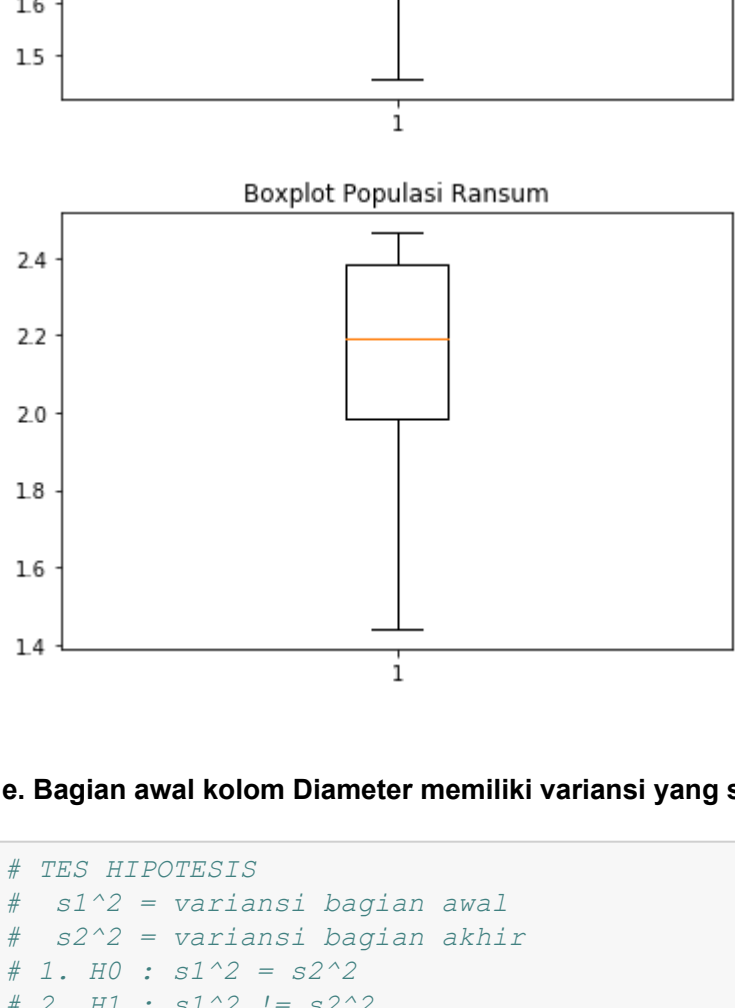
# Daerah kritis
# atas
crit_atas = stats.f.isf(q=0.975, dfn=v1, dfd=v2)
# bawah
crit_bawah = stats.f.isf(q=0.025, dfn=v1, dfd=v2)

# p-value
p = stats.f.sf(f, dfn=v1, dfd=v2)

print("Critical value : f < {0} atau f > {0}'.format(crit_atas, crit_bawah))
print("P value :",p)
print("P value :",p)
if (f < crit_atas or f > crit_bawah):
    print("Tolak H0, dapat disimpulkan variansi bagian awal tidak sama dengan bagian akhir")
else:
    print("Tolak H0, dapat disimpulkan variansi bagian awal tidak sama dengan bagian akhir")
    print("Gagal tolak H0, dapat disimpulkan variansi bagian awal sama dengan bagian akhir")

Critical value : f < 0.7795916576054985 atau f > 0.7795916576054985
F : 1.355571121047759
p value : 0.00834937751160385
Tolak H0, dapat disimpulkan variansi bagian awal tidak sama dengan bagian akhir
```

```
In [76]: # boxplot
createBoxplot(sample1, "Random Sampel Bagian Awal Diameter n = 152")
createBoxplot(sample2, "Random Sampel Bagian Akhir Diameter n = 152")
createBoxplot(df["Diameter"], "Populasi Diameter")
```



6. Test korelasi: tentukan apakah setiap kolom non-target berkorelasi dengan kolom target, dengan menggambarkan juga scatter plot nya. Gunakan correlation test.

Daerah

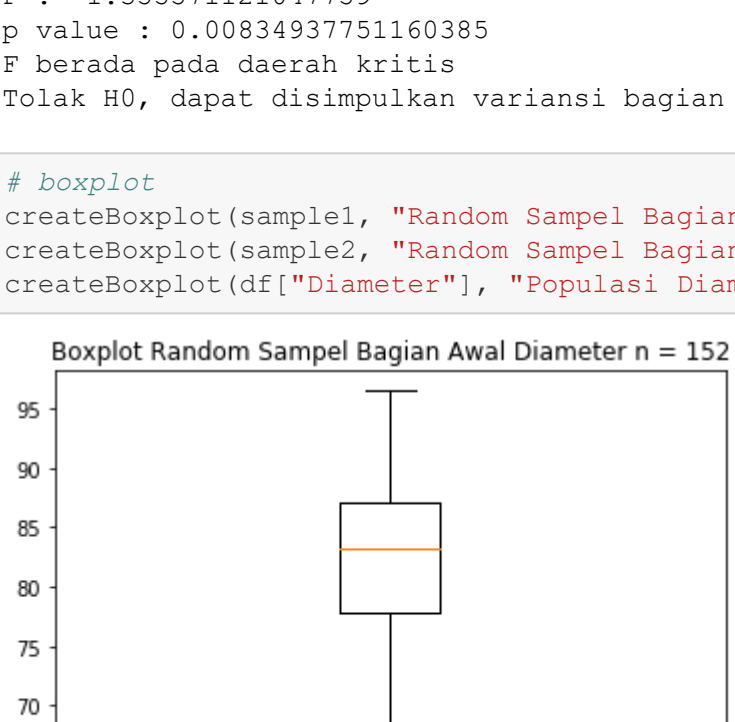
Berdasarkan correlation test didapat nilai correlation sebesar -0.602747 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Daerah berkorelasi negatif secara moderat dengan kolom Kelas. Klam tersebut diperkuat dengan hasil scatter plot yang menunjukkan terdapat kecenderungan untuk nilai kolom Daerah yang mendekati titik ekstrem untuk berada pada klasifikasi kelas yang berbeda. Nilai kolom Daerah yang mendekati nilai minimum cenderung berada pada Kelas 2, sebaliknya nilai yang mendekati nilai maksimum cenderung berada pada Kelas 1

```
In [77]: # Correlation test
df[['Daerah', 'Kelas']].corr()
```

```
Out[77]:
```

	Daerah	Kelas
Daerah	1.000000	-0.602747
Kelas	-0.602747	1.000000

```
In [78]: # Scatter plot
plt.scatter(df[['Daerah']], df[['Kelas']])
plt.show()
```



Sumbu Utama

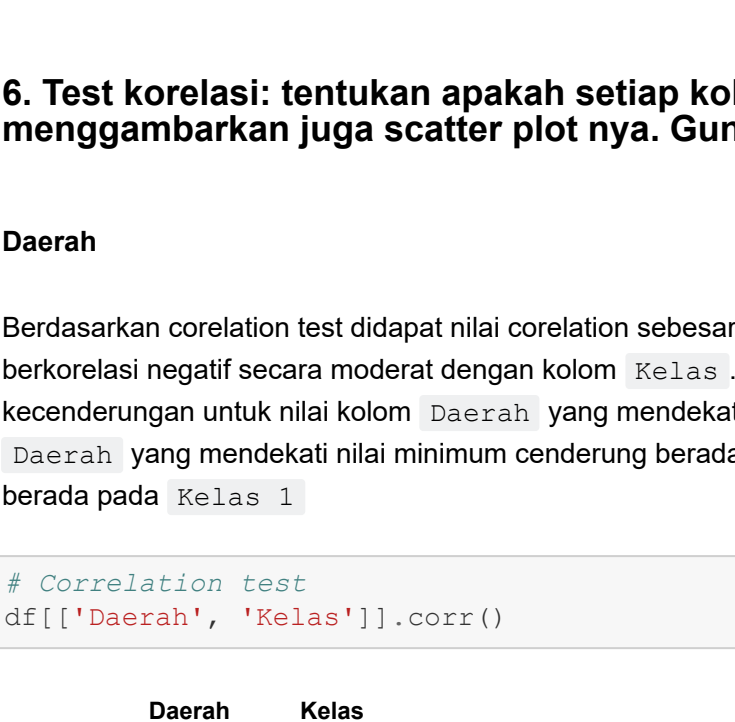
Berdasarkan correlation test didapat nilai correlation sebesar -0.713091 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Sumbu Utama berkorelasi negatif secara cukup kuat dengan kolom Kelas. Pada scatter plot, terlihat bahwa nilai Sumbu Utama yang mendekati nilai minimum akan cenderung berada pada Kelas 2 dan nilai Sumbu Utama yang bernilai lebih dari 140 akan berada pada Kelas 1

```
In [79]: # Correlation test
df[['SumbuUtama', 'Kelas']].corr()
```

```
Out[79]:
```

	SumbuUtama	Kelas
SumbuUtama	1.000000	-0.713091
Kelas	-0.713091	1.000000

```
In [80]: # Scatter plot
plt.scatter(df[['SumbuUtama']], df[['Kelas']])
plt.show()
```



Sumbu Kecil

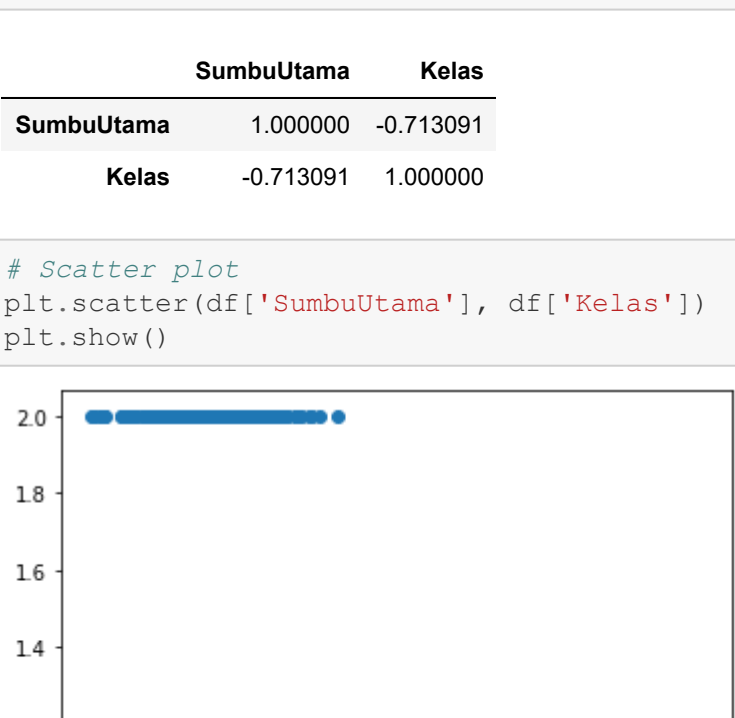
Berdasarkan correlation test didapat nilai correlation sebesar -0.152975 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Sumbu Kecil berkorelasi negatif secara lemah dengan kolom Kelas. Pada scatter plot, tidak begitu terlihat pola yang terbentuk oleh kolom Sumbu Kecil dan Kelas.

```
In [81]: # Correlation test
df[['SumbuKecil', 'Kelas']].corr()
```

```
Out[81]:
```

	SumbuKecil	Kelas
SumbuKecil	1.000000	-0.152975
Kelas	-0.152975	1.000000

```
In [82]: # Scatter plot
plt.scatter(df[['SumbuKecil']], df[['Kelas']])
plt.show()
```



Keunikan

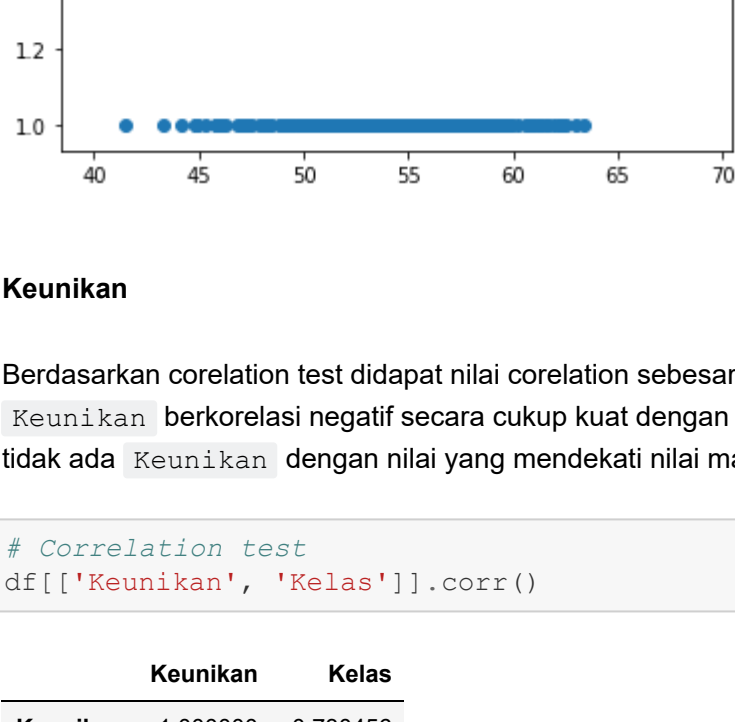
Berdasarkan correlation test didapat nilai correlation sebesar -0.730456 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Keunikan berkorelasi negatif secara moderat dengan kolom Kelas. Pada scatter plot, terlihat sedikit kecenderungan untuk nilai Keunikan yang mendekati nilai minimum dan yang berada di atas 320.

```
In [83]: # Correlation test
df[['Keunikan', 'Kelas']].corr()
```

```
Out[83]:
```

	Keunikan	Kelas
Keunikan	1.000000	-0.730456
Kelas	-0.730456	1.000000

```
In [84]: # Scatter plot
plt.scatter(df[['Keunikan']], df[['Kelas']])
plt.show()
```



Area Bulatan

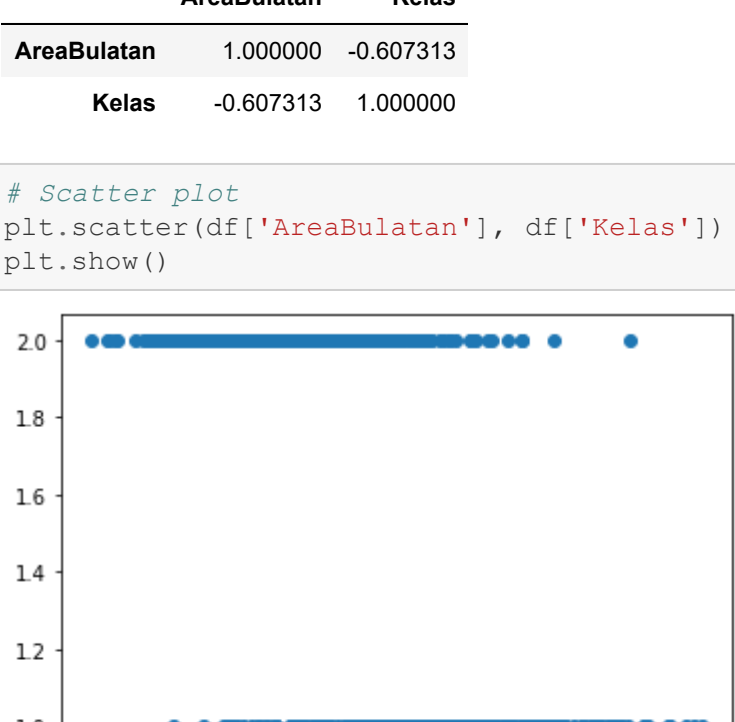
Berdasarkan correlation test didapat nilai correlation sebesar -0.607313 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Area Bulatan berkorelasi negatif secara moderat dengan kolom Kelas. Pada scatter plot, ditunjukkan kecenderungan dan pola hanya terjadi pada nilai Area Bulatan yang mendekati nilai ekstrem.

```
In [85]: # Correlation test
df[['AreaBulatan', 'Kelas']].corr()
```

```
Out[85]:
```

	AreaBulatan	Kelas
AreaBulatan	1.000000	-0.607313
Kelas	-0.607313	1.000000

```
In [86]: # Scatter plot
plt.scatter(df[['AreaBulatan']], df[['Kelas']])
plt.show()
```



Diameter

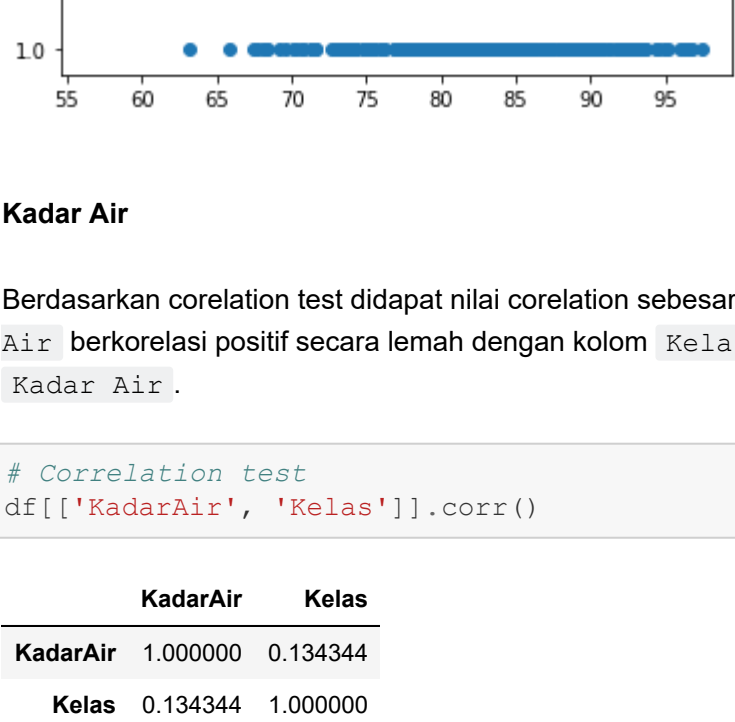
Berdasarkan correlation test didapat nilai correlation sebesar -0.602536 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Diameter berkorelasi negatif secara kuat dengan kolom Kelas. Sebagaimana kolom Area Bulatan, kecenderungan dan pola pada kolom Diameter hanya terjadi pada nilai yang mendekati nilai ekstrem.

```
In [87]: # Correlation test
df[['Diameter', 'Kelas']].corr()
```

```
Out[87]:
```

	Diameter	Kelas
Diameter	1.000000	-0.602536
Kelas	-0.602536	1.000000

```
In [88]: # Scatter plot
plt.scatter(df[['Diameter']], df[['Kelas']])
plt.show()
```



Kadar Air

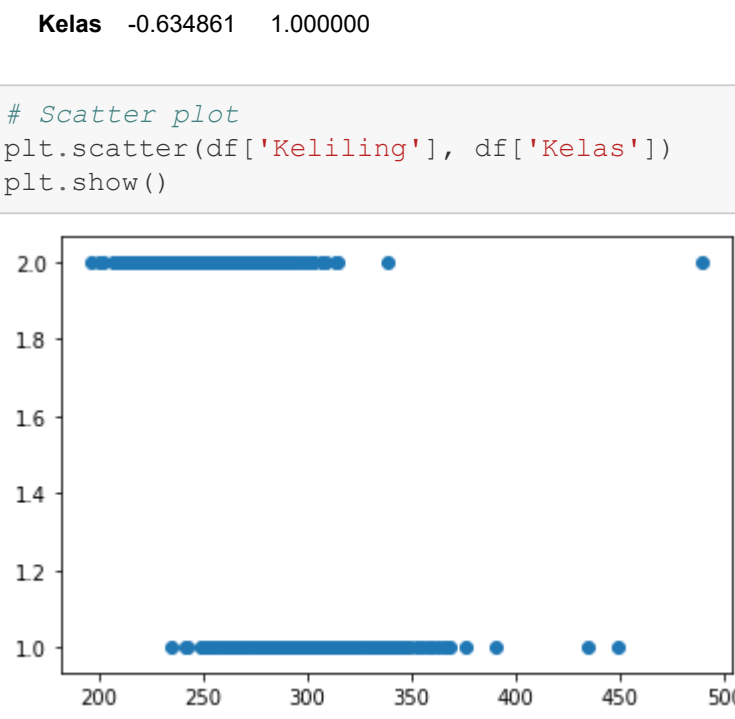
Berdasarkan correlation test didapat nilai correlation sebesar 0.134344 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Kadar Air berkorelasi positif secara lemah dengan kolom Kelas. Pada scatter plot, terlihat tidak ada kecenderungan dan pola untuk kolom Kadar Air.

```
In [89]: # Correlation test
df[['KadarAir', 'Kelas']].corr()
```

```
Out[89]:
```

	KadarAir	Kelas
KadarAir	1.000000	0.134344
Kelas	0.134344	1.000000

```
In [90]: # Scatter plot
plt.scatter(df[['KadarAir']], df[['Kelas']])
plt.show()
```



Keilling

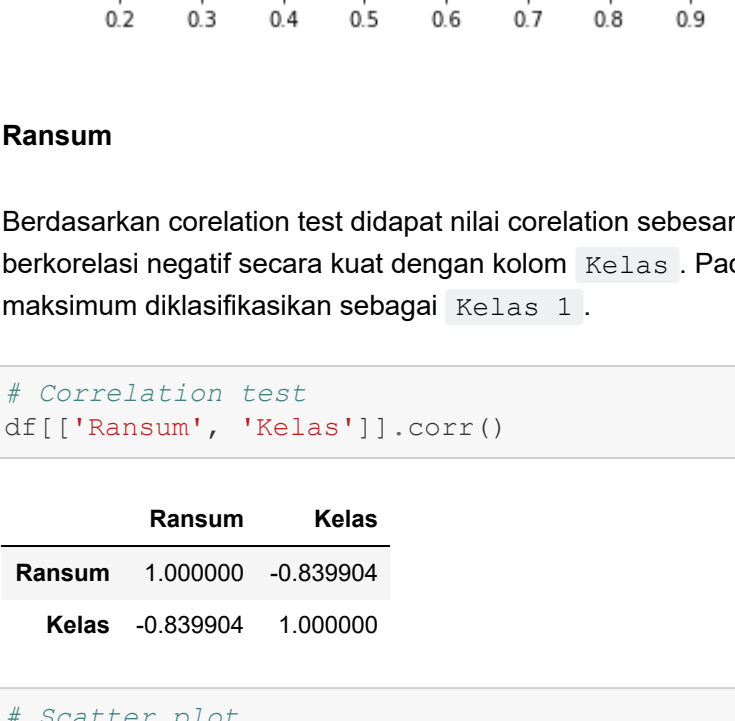
Berdasarkan correlation test didapat nilai correlation sebesar -0.634861 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Keilling berkorelasi negatif secara moderat dengan kolom Kelas. Pada scatter plot, terlihat sedikit kecenderungan untuk nilai Keilling yang mendekati nilai minimum dan yang berada di atas 320.

```
In [91]: # Correlation test
df[['Keilling', 'Kelas']].corr()
```

```
Out[91]:
```

	Keilling	Kelas
Keilling	1.000000	-0.634861
Kelas	-0.634861	1.000000

```
In [92]: # Scatter plot
plt.scatter(df[['Keilling']], df[['Kelas']])
plt.show()
```



Bulatan

Berdasarkan correlation test didapat nilai correlation sebesar 0.545005 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Bulatan berkorelasi positif secara kuat dengan kolom Kelas. Terlihat pada scatter plot bahwa ada sedikit kecenderungan untuk nilai Bulatan yang mendekati nilai maksimum untuk diklasifikasikan sebagai Kelas 2.

```
In [93]: # Correlation test
df[['Bulatan', 'Kelas']].corr()
```

```
Out[93]:
```

	Bulatan	Kelas
Bulatan	1.000000	0.545005
Kelas	0.545005	1.000000

```
In [94]: # Scatter plot
plt.scatter(df[['Bulatan']], df[['Kelas']])
plt.show()
```



Ransum

Berdasarkan correlation test didapat nilai correlation sebesar -0.839904 . Dengan nilai tersebut, dapat disimpulkan bahwa kolom Ransum berkorelasi negatif secara kuat dengan kolom Kelas. Pada scatter plot, terlihat pola bahwa nilai Ransum yang mendekati nilai maksimum diklasifikasikan sebagai Kelas 1.

```
In [95]: # Correlation test
df[['Ransum', 'Kelas']].corr()
```

```
Out[95]:
```

	Ransum	Kelas
Ransum	1.000000	-0.839904
Kelas	-0.839904	1.000000

```
In [96]: # Scatter plot
plt.scatter(df[['Ransum']], df[['Kelas']])
plt.show()
```

