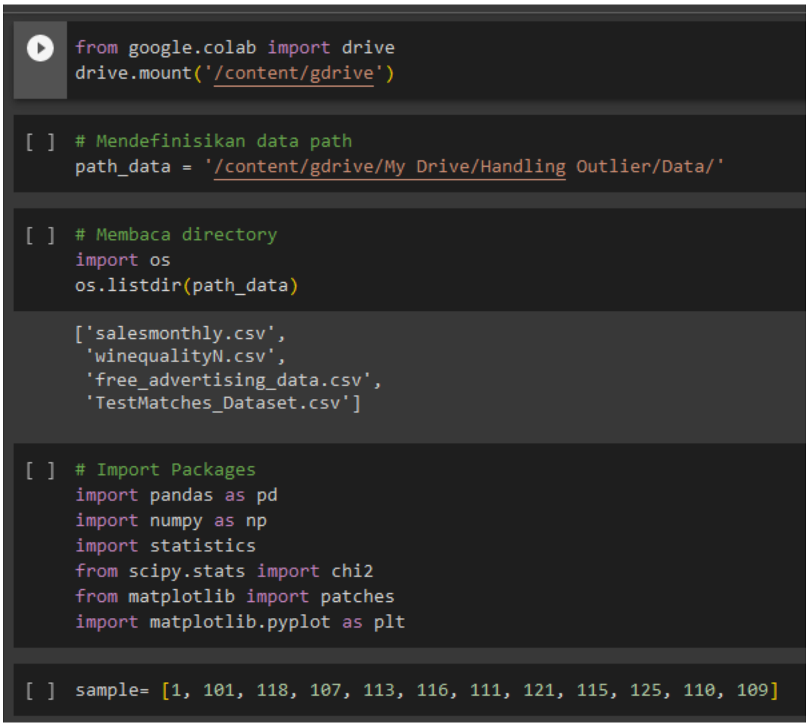
Handling Outlier

Download dataset:
https://drive.google.com/drive/folders/1AjKQsuCSe2hWZwSRNdy70NJmHj7dJ-LU?usp=sharing

1. Load Data



```
from google.colab import drive
drive.mount('/content/gdrive')
```

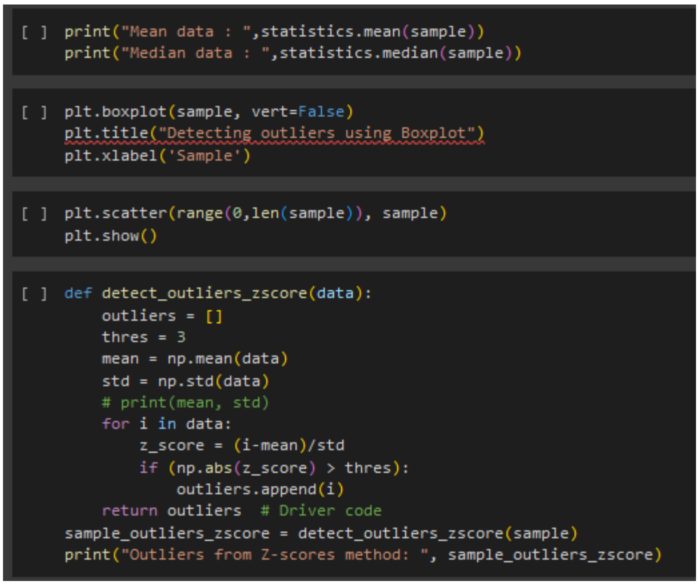
```
[ ] # Mendefinisikan data path
    path_data = '/content/gdrive/My Drive/Handling Outlier/Data/'
```

```
[ ] # Membaca directory
    import os
    os.listdir(path_data)

    ['salesmonthly.csv',
     'winequalityN.csv',
     'free_advertising_data.csv',
     'TestMatches_Dataset.csv']
```

```
[ ] # Import Packages
    import pandas as pd
    import numpy as np
    import statistics
    from scipy.stats import chi2
    from matplotlib import patches
    import matplotlib.pyplot as plt
```

```
[ ] sample= [1, 101, 118, 107, 113, 116, 111, 121, 115, 125, 110, 109]
```

2. Deteksi Outlier

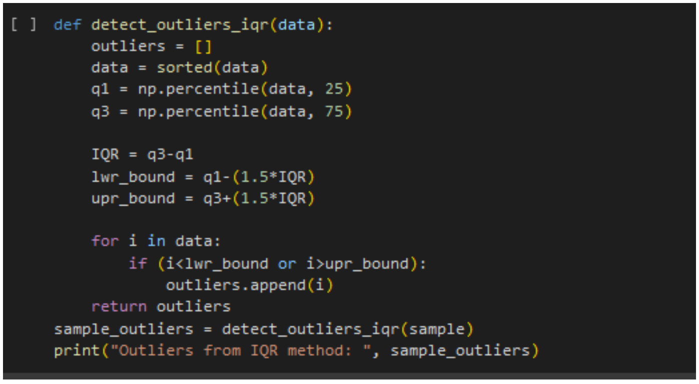


```
[ ] print("Mean data : ",statistics.mean(sample))
    print("Median data : ",statistics.median(sample))
```

```
[ ] plt.boxplot(sample, vert=False)
    plt.title("Detecting outliers using Boxplot")
    plt.xlabel('Sample')
```

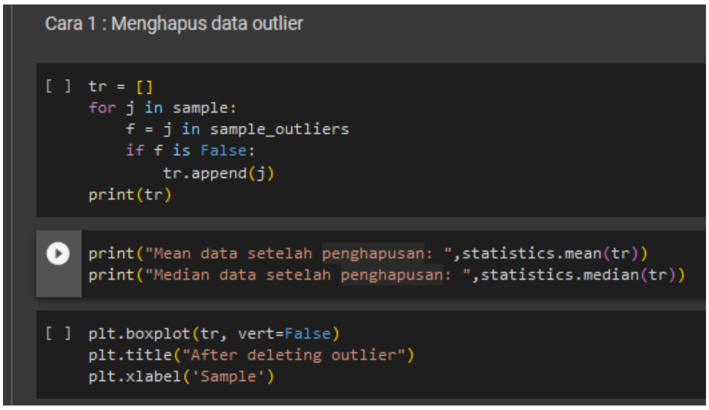
```
[ ] plt.scatter(range(0,len(sample)), sample)
    plt.show()
```

```
[ ] def detect_outliers_zscore(data):
        outliers = []
        thres = 3
        mean = np.mean(data)
        std = np.std(data)
        # print(mean, std)
        for i in data:
            z_score = (i-mean)/std
            if (np.abs(z_score) > thres):
                outliers.append(i)
        return outliers  # Driver code
    sample_outliers_zscore = detect_outliers_zscore(sample)
    print("Outliers from Z-scores method: ", sample_outliers_zscore)
```



```
[ ] def detect_outliers_iqr(data):
        outliers = []
        data = sorted(data)
        q1 = np.percentile(data, 25)
        q3 = np.percentile(data, 75)

        IQR = q3-q1
        lwr_bound = q1-(1.5*IQR)
        upr_bound = q3+(1.5*IQR)

        for i in data:
            if (i<lwr_bound or i>upr_bound):
                outliers.append(i)
        return outliers
    sample_outliers = detect_outliers_iqr(sample)
    print("Outliers from IQR method: ", sample_outliers)
```

3. Handling Outlier dengan sampling

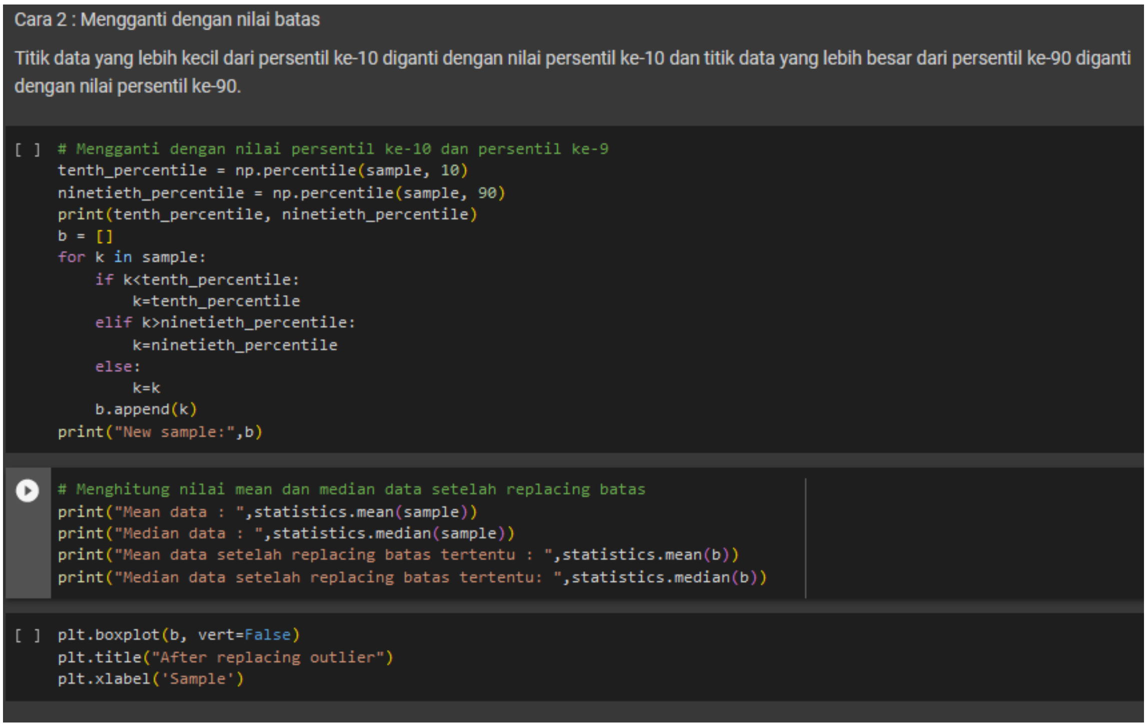


Cara 1 : Menghapus data outlier

```
tr = []
for j in sample:
    f = j in sample_outliers
    if f is False:
        tr.append(j)
print(tr)
```

```
print("Mean data setelah penghapusan: ",statistics.mean(tr))
print("Median data setelah penghapusan: ",statistics.median(tr))
```

```
plt.boxplot(tr, vert=False)
plt.title("After deleting outlier")
plt.xlabel('Sample')
```



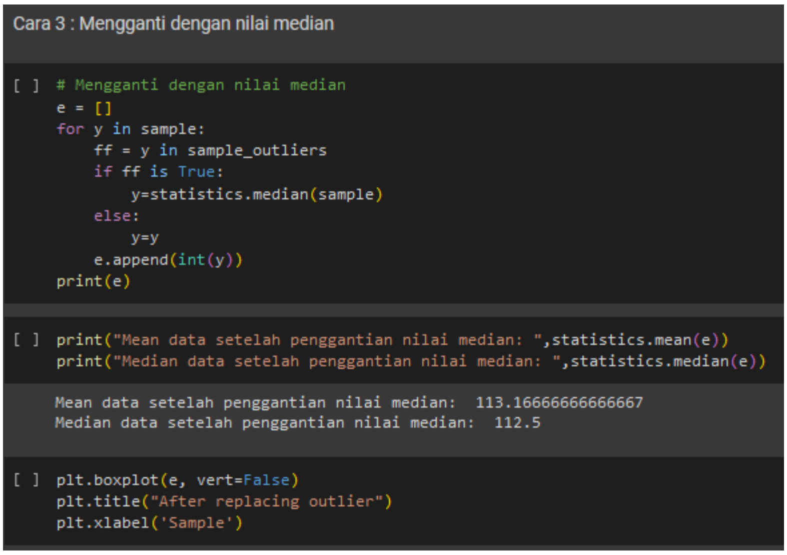
Cara 2 : Mengganti dengan nilai batas

Titik data yang lebih kecil dari persentil ke-10 diganti dengan nilai persentil ke-10 dan titik data yang lebih besar dari persentil ke-90 diganti dengan nilai persentil ke-90.

```
# Mengganti dengan nilai persentil ke-10 dan persentil ke-9
tenth_percentile = np.percentile(sample, 10)
ninetieth_percentile = np.percentile(sample, 90)
print(tenth_percentile, ninetieth_percentile)
b = []
for k in sample:
    if k<tenth_percentile:
        k=tenth_percentile
    elif k>ninetieth_percentile:
        k=ninetieth_percentile
    else:
        k=k
    b.append(k)
print("New sample:",b)
```

```
# Menghitung nilai mean dan median data setelah replacing batas
print("Mean data : ",statistics.mean(sample))
print("Median data : ",statistics.median(sample))
print("Mean data setelah replacing batas tertentu : ",statistics.mean(b))
print("Median data setelah replacing batas tertentu: ",statistics.median(b))
```

```
plt.boxplot(b, vert=False)
plt.title("After replacing outlier")
plt.xlabel('Sample')
```



Cara 3 : Mengganti dengan nilai median

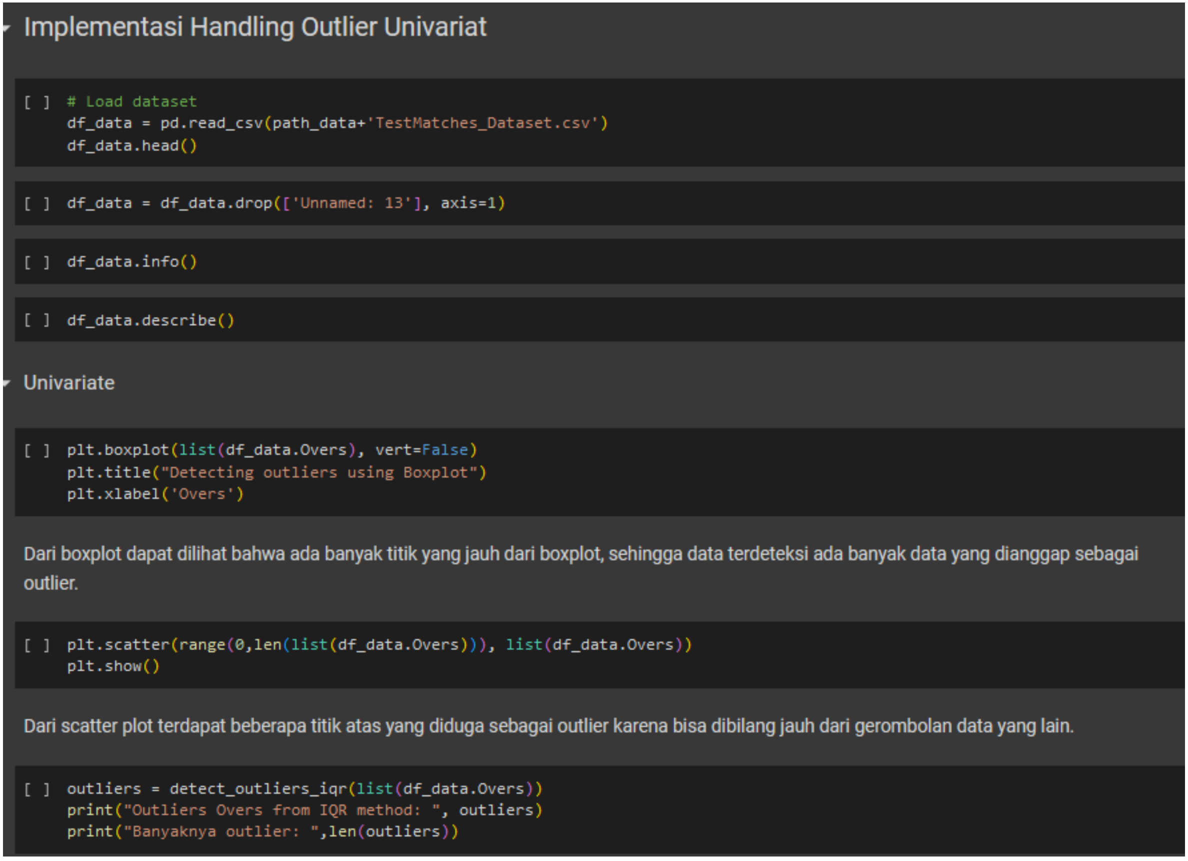
```
# Mengganti dengan nilai median
e = []
for y in sample:
    ff = y in sample_outliers
    if ff is True:
        y=statistics.median(sample)
    else:
        y=y
    e.append(int(y))
print(e)
```

```
print("Mean data setelah penggantian nilai median: ",statistics.mean(e))
print("Median data setelah penggantian nilai median: ",statistics.median(e))
```

```
Mean data setelah penggantian nilai median:  113.16666666666667
Median data setelah penggantian nilai median:  112.5
```

```
plt.boxplot(e, vert=False)
plt.title("After replacing outlier")
plt.xlabel('Sample')
```

4. Handling Outlier Univariat dengan Dataset



Implementasi Handling Outlier Univariat

```
# Load dataset
df_data = pd.read_csv(path_data+'TestMatches_Dataset.csv')
df_data.head()
```

```
df_data = df_data.drop(['Unnamed: 13'], axis=1)
```

```
df_data.info()
```

```
df_data.describe()
```

Univariate

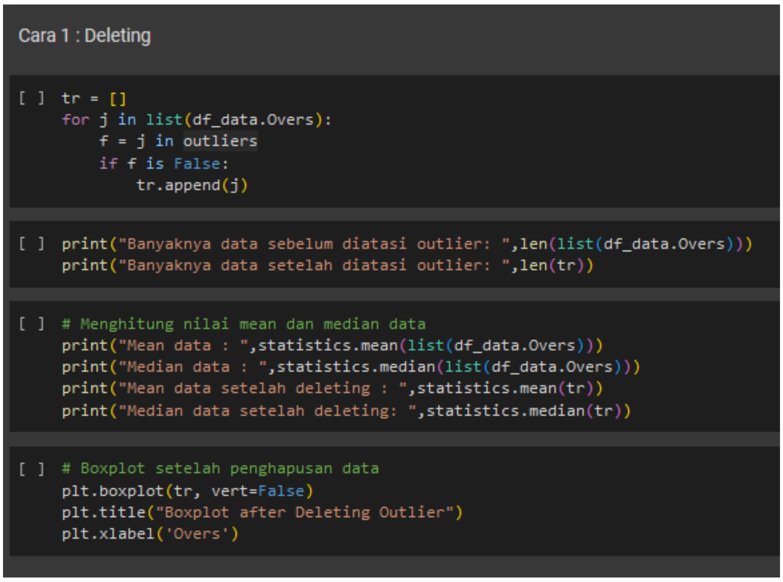
```
plt.boxplot(list(df_data.Overs), vert=False)
plt.title("Detecting outliers using Boxplot")
plt.xlabel('Overs')
```

Dari boxplot dapat dilihat bahwa ada banyak titik yang jauh dari boxplot, sehingga data terdeteksi ada banyak data yang dianggap sebagai outlier.

```
plt.scatter(range(0,len(list(df_data.Overs))), list(df_data.Overs))
plt.show()
```

Dari scatter plot terdapat beberapa titik atas yang diduga sebagai outlier karena bisa dibilang jauh dari gerombolan data yang lain.

```
outliers = detect_outliers_iqr(list(df_data.Overs))
print("Outliers Overs from IQR method: ", outliers)
print("Banyaknya outlier: ",len(outliers))
```



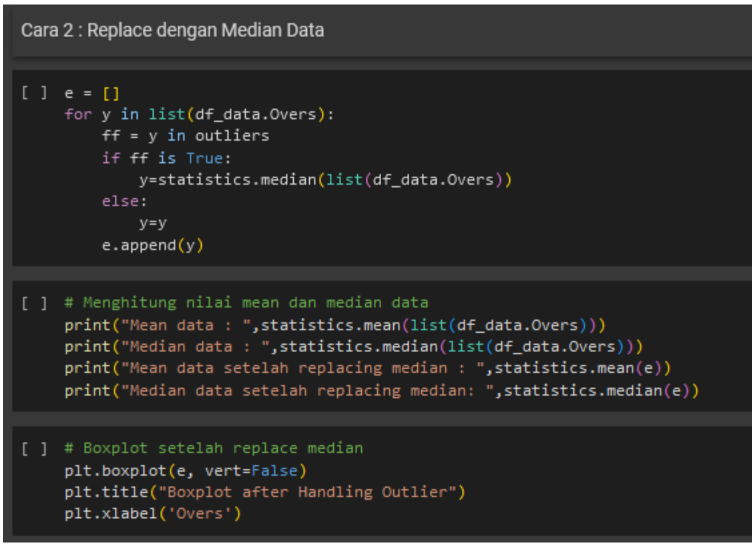
Cara 1 : Deleting

```
tr = []
for j in list(df_data.Overs):
    f = j in outliers
    if f is False:
        tr.append(j)
```

```
print("Banyaknya data sebelum diatasi outlier: ",len(list(df_data.Overs)))
print("Banyaknya data setelah diatasi outlier: ",len(tr))
```

```
# Menghitung nilai mean dan median data
print("Mean data : ",statistics.mean(list(df_data.Overs)))
print("Median data : ",statistics.median(list(df_data.Overs)))
print("Mean data setelah deleting : ",statistics.mean(tr))
print("Median data setelah deleting: ",statistics.median(tr))
```

```
# Boxplot setelah penghapusan data
plt.boxplot(tr, vert=False)
plt.title("Boxplot after Deleting Outlier")
plt.xlabel('Overs')
```



Cara 2 : Replace dengan Median Data

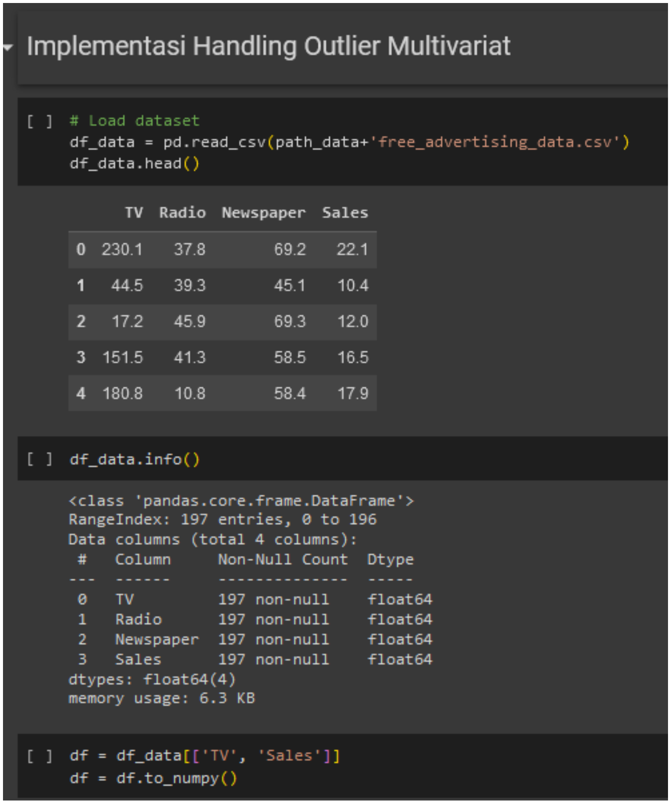
```
e = []
for y in list(df_data.Overs):
    ff = y in outliers
    if ff is True:
        y=statistics.median(list(df_data.Overs))
    else:
        y=y
    e.append(y)
```

```
# Menghitung nilai mean dan median data
print("Mean data : ",statistics.mean(list(df_data.Overs)))
print("Median data : ",statistics.median(list(df_data.Overs)))
print("Mean data setelah replacing median : ",statistics.mean(e))
print("Median data setelah replacing median: ",statistics.median(e))
```

```
# Boxplot setelah replace median
plt.boxplot(e, vert=False)
plt.title("Boxplot after Handling Outlier")
plt.xlabel('Overs')
```

5. Handling Outlier Multivariat

Step 1



Implementasi Handling Outlier Multivariat

```
[ ] # Load dataset
    df_data = pd.read_csv(path_data+'free_advertising_data.csv')
    df_data.head()
```

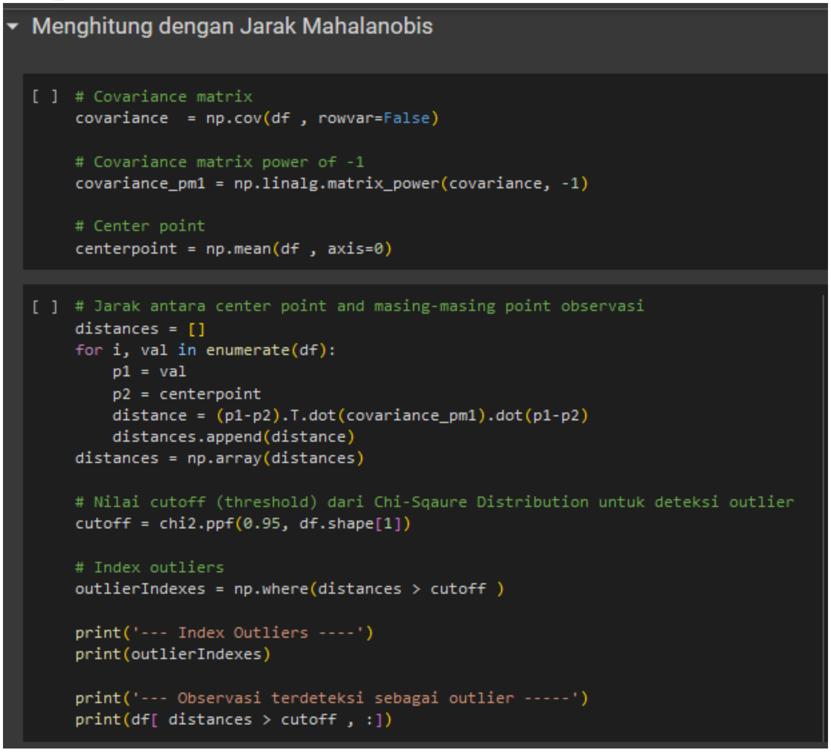
	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9

```
[ ] df_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 197 entries, 0 to 196
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   TV         197 non-null    float64
 1   Radio      197 non-null    float64
 2   Newspaper  197 non-null    float64
 3   Sales      197 non-null    float64
dtypes: float64(4)
memory usage: 6.3 KB
```

```
[ ] df = df_data[['TV', 'Sales']]
    df = df.to_numpy()
```

Step 2

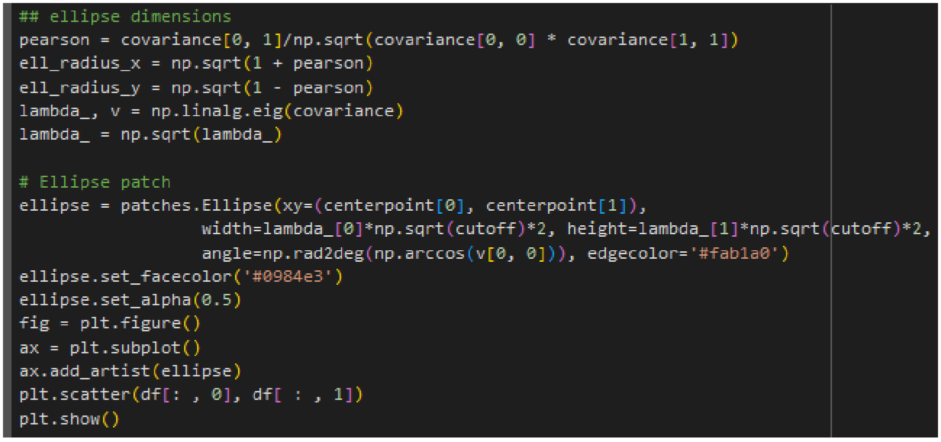


Menghitung dengan Jarak Mahalanobis

```
[ ] # Covariance matrix
    covariance  = np.cov(df , rowvar=False)

    # Covariance matrix power of -1
    covariance_pm1 = np.linalg.matrix_power(covariance, -1)

    # Center point
    centerpoint = np.mean(df , axis=0)
```

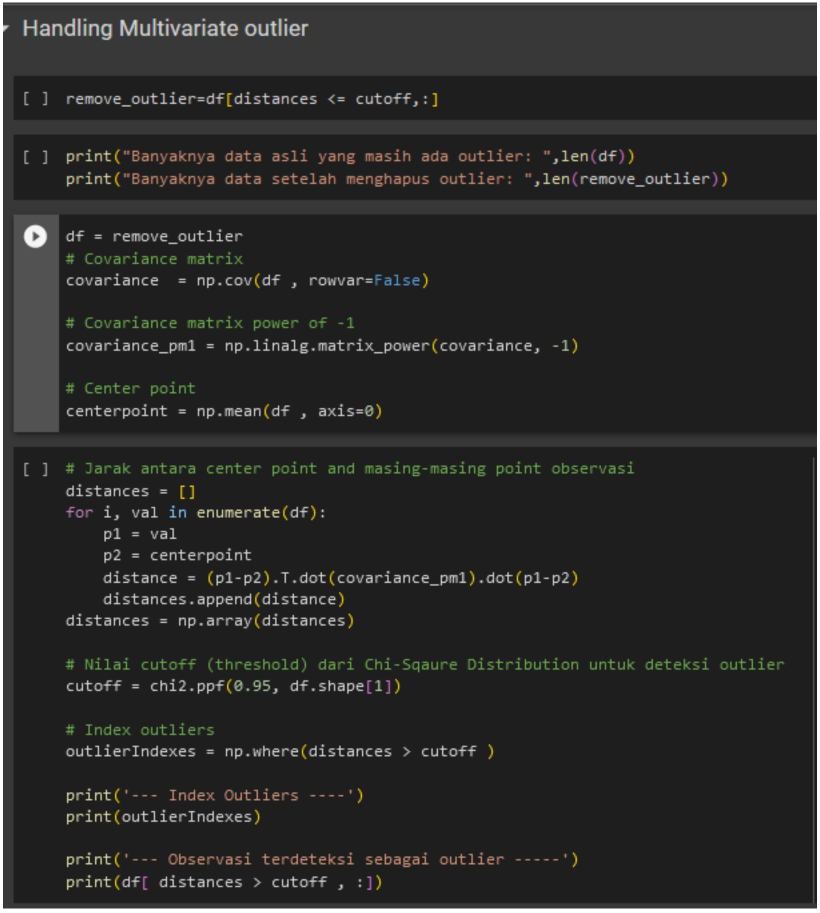
```
[ ] # Jarak antara center point and masing-masing point observasi
    distances = []
    for i, val in enumerate(df):
        p1 = val
        p2 = centerpoint
        distance = (p1-p2).T.dot(covariance_pm1).dot(p1-p2)
        distances.append(distance)
    distances = np.array(distances)

    # Nilai cutoff (threshold) dari Chi-Sqaure Distribution untuk deteksi outlier
    cutoff = chi2.ppf(0.95, df.shape[1])

    # Index outliers
    outlierIndexes = np.where(distances > cutoff )

    print('--- Index Outliers ----')
    print(outlierIndexes)

    print('--- Observasi terdeteksi sebagai outlier -----')
    print(df[ distances > cutoff , :])
```

Step 3



```
## ellipse dimensions
pearson = covariance[0, 1]/np.sqrt(covariance[0, 0] * covariance[1, 1])
ell_radius_x = np.sqrt(1 + pearson)
ell_radius_y = np.sqrt(1 - pearson)
lambda_, v = np.linalg.eig(covariance)
lambda_ = np.sqrt(lambda_)

# Ellipse patch
ellipse = patches.Ellipse(xy=(centerpoint[0], centerpoint[1]),
                  width=lambda_[0]*np.sqrt(cutoff)*2, height=lambda_[1]*np.sqrt(cutoff)*2,
                  angle=np.rad2deg(np.arccos(v[0, 0])), edgecolor='#fab1a0')
ellipse.set_facecolor('#0984e3')
ellipse.set_alpha(0.5)
fig = plt.figure()
ax = plt.subplot()
ax.add_artist(ellipse)
plt.scatter(df[: , 0], df[ : , 1])
plt.show()
```

Step 4 (penyelesaian masalah)



Handling Multivariate outlier

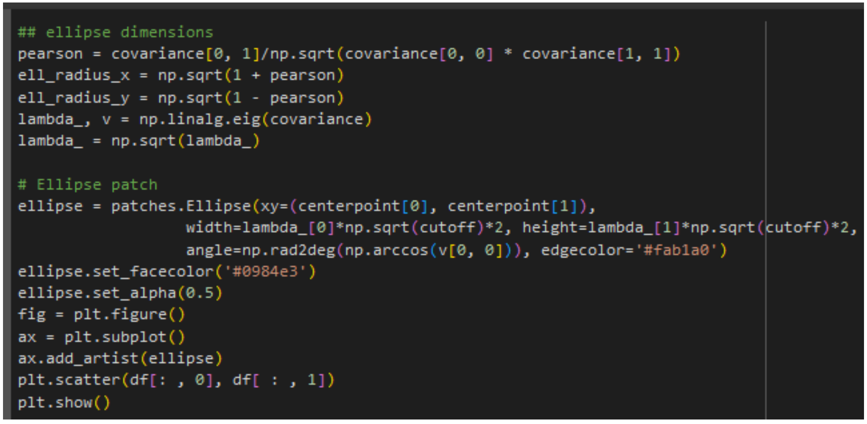
```
[ ] remove_outlier=df[distances <= cutoff,:]
```

```
[ ] print("Banyaknya data asli yang masih ada outlier: ",len(df))
    print("Banyaknya data setelah menghapus outlier: ",len(remove_outlier))
```

```
df = remove_outlier
# Covariance matrix
covariance  = np.cov(df , rowvar=False)

# Covariance matrix power of -1
covariance_pm1 = np.linalg.matrix_power(covariance, -1)

# Center point
centerpoint = np.mean(df , axis=0)
```

```
[ ] # Jarak antara center point and masing-masing point observasi
    distances = []
    for i, val in enumerate(df):
          p1 = val
          p2 = centerpoint
          distance = (p1-p2).T.dot(covariance_pm1).dot(p1-p2)
          distances.append(distance)
    distances = np.array(distances)

    # Nilai cutoff (threshold) dari Chi-Sqaure Distribution untuk deteksi outlier
    cutoff = chi2.ppf(0.95, df.shape[1])

    # Index outliers
    outlierIndexes = np.where(distances > cutoff )

    print('--- Index Outliers ----')
    print(outlierIndexes)

    print('--- Observasi terdeteksi sebagai outlier -----')
    print(df[ distances > cutoff , :])
```

Step 5 (cek kembali elips)



```
## ellipse dimensions
pearson = covariance[0, 1]/np.sqrt(covariance[0, 0] * covariance[1, 1])
ell_radius_x = np.sqrt(1 + pearson)
ell_radius_y = np.sqrt(1 - pearson)
lambda_, v = np.linalg.eig(covariance)
lambda_ = np.sqrt(lambda_)

# Ellipse patch
ellipse = patches.Ellipse(xy=(centerpoint[0], centerpoint[1]),
                  width=lambda_[0]*np.sqrt(cutoff)*2, height=lambda_[1]*np.sqrt(cutoff)*2,
                  angle=np.rad2deg(np.arccos(v[0, 0])), edgecolor='#fab1a0')
ellipse.set_facecolor('#0984e3')
ellipse.set_alpha(0.5)
fig = plt.figure()
ax = plt.subplot()
ax.add_artist(ellipse)
plt.scatter(df[: , 0], df[ : , 1])
plt.show()
```

Perhatian: Jika Titik Biru masih ada diluar lingkaran elips maka lakukan Step 4 dan Step 5 !!!