LAPORAN DATA WRANGLING C
A DEEP DIVE INTO DATA WRANGLING WITH PYTHON

DISUSUN OLEH:

Mohamad Ibnu Fajar Maulana (21083010106)

DOSEN PENGAMPU:

KARTIKA MAULIDA HINDRAYANI, S.Kom, M.Kom

PROGRAM STUDI SAINS DATA

FAKULTAS ILMU KOMPUTER

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"

JAWA TIMUR

2022

DAFTAR ISI

T04 - A Deep Dive into Data Wrangling with Python

1. Load dataset yang kamu punya boleh sumbernya dari Kaggle dll
2. Temukan missing values pada dataset tersebut
3. Buat kolom baru dengan menggunakan teknik subsetting
4. Temukan outliers pada dataset tersebut
5. Gunakan fungsi groupBy

Input-Output:

1. Load Dataset

   Dataset yang digunakan pada laporan ini mengangkat permasalahan harga penutupan saham dan nilai jualnya, data tersebut diambil dari website:

   https://www.kaggle.com/code/haidhiangkawijana/prediksi-saham-lstm/notebook

```
[1]: !pip install xlrd
     Requirement already satisfied: xlrd in c:\users\gaming 3\anaconda3\lib\site-packages (2.0.1)
```

- Pada Gambar diatas jika tipe file xls bisa menggunakan *!pip install xlrd*, akan tetapi dataset yang saya gunakan memiliki tipe file csv, berikut caranya:

```
[2]: import pandas as pd
     df = pd.read_csv("./DaftarSaham.csv")
     df
```

- Gambar ditas kita menggunakan import library python yakni *import pandas as pd* guna menampilkan sebuah DataFrame(csv). Kemudian data tersebut menghasilkan output:

[2]:

	Code	Name	ListingDate	Shares	ListingBoard	Sector	LastPrice	MarketCap	MinutesFirstAdded	MinutesLastUpdated	HourlyFirstAdded	HourlyLastUpdated	DailyFirstAdded	DailyLastUpdated
0	AALI	Astra Agro Lestari Tbk.	1997-12-09	1924688333	Utama	Consumer Non-Cyclicals	8250.0	1.587868e+13	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2001-04-16	2022-09-30
1	ABBA	Mahaka Media Tbk.	2002-04-03	3935892857	Pengembangan	Consumer Cyclicals	214.0	8.422811e+11	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2002-04-03	2022-09-30
2	ABDA	Asuransi Bina Dana Arta Tbk.	1989-07-06	620806680	Pengembangan	Financials	6025.0	3.740360e+12	2021-11-01 09:00:00	2022-09-23 15:59:00	2020-04-16 09:00:00	2022-09-23 16:00:00	2001-04-16	2022-09-30
3	ABMM	ABM Investama Tbk.	2011-12-06	2753165000	Utama	Industrials	3960.0	1.090253e+13	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2011-12-06	2022-09-30
4	ACES	Ace Hardware Indonesia Tbk.	2007-11-06	17150000000	Utama	Consumer Cyclicals	610.0	1.046150e+13	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2007-11-06	2022-09-30
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
805	YULE	Yulie Sekuritas Indonesia Tbk.	2004-12-10	1785000000	Pengembangan	Financials	2100.0	3.748500e+12	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2004-12-10	2022-09-30
806	ZBRA	Dosni Roha Indonesia Tbk.	1991-08-01	2510706263	Pengembangan	Industrials	570.0	1.431103e+12	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2001-04-16	2022-09-30
807	ZINC	Kapuas Prima Coal Tbk.	2017-10-16	25250000000	Pengembangan	Basic Materials	71.0	1.792750e+12	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2017-10-16	2022-09-30
808	ZONE	Mega Perintis Tbk.	2018-12-12	870171478	Pengembangan	Consumer Cyclicals	1045.0	9.093292e+11	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2018-12-12	2022-09-30
809	ZYRX	Zyrexindo Mandiri Buana Tbk.	2021-03-30	1333333350	Pengembangan	Technology	474.0	6.320000e+11	2021-11-01 09:00:00	2022-09-30 15:59:00	2021-03-30 09:00:00	2022-09-30 16:00:00	2021-03-30	2022-09-30

810 rows × 14 columns

106_Mohamad Ibnu Fajar Maulana

- Pada gambar diatas merupakan DataFrame saham-saham yang ada di Indonesia, dengan 810 baris dan 14 kolom data.

2. Menemukan Missing Value

- Untuk menemukan missing value kita dapat melakukan kodingan seperti gambar dibawah ini:

```
[4]: #Memeriksa missing value setiap variabel
     df.isnull().any()
```

- Gambar diatas menggunakan method *df.isnull().any()* yang merupakan sebuah method pada objek pandas DataFrame yang digunakan untuk memeriksa apakah terdapat nilai null atau kosong pada setiap kolom DataFrame. Kemudian mendapatkan output sebagai berikut:

```
[4]: Code                 False
     Name                 False
     ListingDate          False
     Shares               False
     ListingBoard         False
     Sector               False
     LastPrice             True
     MarketCap             True
     MinutesFirstAdded     True
     MinutesLastUpdated    True
     HourlyFirstAdded      True
     HourlyLastUpdated     True
     DailyFirstAdded       True
     DailyLastUpdated      True
     dtype: bool
```

- Gambar diatas terdapat True-False yang artinya menentukan kolom mana yang terdapat *missing value*. Kolom yang dilabeli *True* artinya terdapat *missing value*
- Penjelasan lebih lanjut alangkah baiknya melakukan kode seperti dibawah ini:

```
[53]: for i in df.columns:
          miss = df[i].isnull().sum()
          if miss>0:
              print('\033[1m', '\33[31m')
              print("{} terdapat {} missing value(s)".format(i,miss))
          else:
              print('\033[1m', '\33[34m')
              print("{} tidak terdapat missing value!".format(i))
```

- Gambar tersebut merupakan kode yang mengandung sebuah loop yang digunakan untuk memeriksa apakah terdapat nilai null atau kosong pada setiap kolom DataFrame yang disimpan pada variabel df.
  - Pertama, kode mengambil setiap nama kolom dari DataFrame menggunakan fungsi df.columns.

106_Mohamad Ibnu
Fajar Maulana

- Kemudian, kode memeriksa jumlah nilai null pada setiap kolom dengan menggunakan fungsi df[i].isnull().sum(). Fungsi ini menghitung jumlah nilai null pada kolom yang sedang diperiksa.
- Jika jumlah nilai null pada kolom tersebut lebih besar dari nol, maka kode akan mencetak pesan yang menunjukkan bahwa terdapat nilai null pada kolom tersebut menggunakan format string dan warna merah (dengan escape sequence \033[1m dan \33[31m). Pesan ini akan mencantumkan nama kolom dan jumlah nilai null pada kolom tersebut.
- Jika jumlah nilai null pada kolom tersebut sama dengan nol, maka kode akan mencetak pesan yang menunjukkan bahwa tidak ada nilai null pada kolom tersebut menggunakan format string dan warna biru (dengan escape sequence \033[1m dan \33[34m).

Dengan menggunakan loop ini, kita dapat dengan mudah memeriksa setiap kolom pada DataFrame untuk mengetahui apakah terdapat nilai null pada kolom tersebut.

- Kemudian mendapatkan output:

```
Code tidak terdapat missing value!
Name tidak terdapat missing value!
ListingDate tidak terdapat missing value!
Shares tidak terdapat missing value!
ListingBoard tidak terdapat missing value!
Sector tidak terdapat missing value!
LastPrice terdapat 45 missing value(s)
MarketCap terdapat 45 missing value(s)
MinutesFirstAdded terdapat 45 missing value(s)
MinutesLastUpdated terdapat 45 missing value(s)
HourlyFirstAdded terdapat 23 missing value(s)
HourlyLastUpdated terdapat 23 missing value(s)
DailyFirstAdded terdapat 5 missing value(s)
DailyLastUpdated terdapat 5 missing value(s)
```

- Gambar diatas mengintepretasikan kolom-kolom yang terdapat *missing value* (berwarna merah) dan kolom-kolom yang tidak terdapat *missing value*(berwarna biru) seperti penjelasan sebelumnya, silahkan simak baik-baik hasil outputnya dan laporan ini.

❖ Mengatasi *Missing Value*

- Untuk mengatasi *missing value* lakukan kodingan dibawah ini:

```
[54]: #Drop missing value rows using dropna() function
      #Read the data
      df = df.dropna()
      df
```

- Pada gambar diatas menggunakan *df.dropna()* adalah sebuah method pada objek pandas DataFrame yang digunakan untuk menghapus baris atau kolom yang mengandung nilai null atau kosong. Method ini akan mengembalikan DataFrame baru dengan nilai null dihapus. Kemudian mendapatkan output:

[56]:

	Code	Name	ListingDate	Shares	ListingBoard	Sector	LastPrice	MarketCap	MinutesFirstAdded	MinutesLastUpdated	HourlyFirstAdded	HourlyLastUpdated	DailyFirstAdded	DailyLastUpdated
0	AALI	Astra Agro Lestari Tbk.	1997-12-09	1924688333	Utama	Consumer Non-Cyclicals	8250.0	1.587868e+13	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2001-04-16	2022-09-30
1	ABBA	Mahaka Media Tbk.	2002-04-03	3935892857	Pengembangan	Consumer Cyclicals	214.0	8.422811e+11	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2002-04-03	2022-09-30
2	ABDA	Asuransi Bina Dana Arta Tbk.	1989-07-06	620806680	Pengembangan	Financials	6025.0	3.740360e+12	2021-11-01 09:00:00	2022-09-23 15:59:00	2020-04-16 09:00:00	2022-09-23 16:00:00	2001-04-16	2022-09-30
3	ABMM	ABM Investama Tbk.	2011-12-06	2753165000	Utama	Industrials	3960.0	1.090253e+13	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2011-12-06	2022-09-30
4	ACES	Ace Hardware Indonesia Tbk.	2007-11-06	17150000000	Utama	Consumer Cyclicals	610.0	1.046150e+13	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2007-11-06	2022-09-30
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
805	YULE	Yulie Sekuritas Indonesia Tbk.	2004-12-10	1785000000	Pengembangan	Financials	2100.0	3.748500e+12	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2004-12-10	2022-09-30
806	ZBRA	Dosni Roha Indonesia Tbk.	1991-08-01	2510706263	Pengembangan	Industrials	570.0	1.431103e+12	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2001-04-16	2022-09-30
807	ZINC	Kapuas Prima Coal Tbk.	2017-10-16	25250000000	Pengembangan	Basic Materials	71.0	1.792750e+12	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2017-10-16	2022-09-30
808	ZONE	Mega Perintis Tbk.	2018-12-12	870171478	Pengembangan	Consumer Cyclicals	1045.0	9.093292e+11	2021-11-01 09:00:00	2022-09-30 15:59:00	2020-04-16 09:00:00	2022-09-30 16:00:00	2018-12-12	2022-09-30
809	ZYRX	Zyrexindo Mandiri Buana Tbk.	2021-03-30	1333333350	Pengembangan	Technology	474.0	6.320000e+11	2021-11-01 09:00:00	2022-09-30 15:59:00	2021-03-30 09:00:00	2022-09-30 16:00:00	2021-03-30	2022-09-30

765 rows × 14 columns

- Gambar diatas sudah menghapus beberapa baris yang terdapat *missing value* yang tadinya berjumlah 810 baris dan 14 kolom menjadi 765 baris dan 14 kolom

3. Subsetting DataFrame

- Lakukan kodingan seperti dibawah ini untuk melakukan subsetting DataFrame saham:

```
[31]: df_subset = df.loc[
       [i for i in range(0,5)],
          ['Code','Name','ListingDate','Shares','LastPrice', 'MarketCap']]
      df_subset
```

- Gambar diatas odingan tersebut merupakan sebuah kode untuk membuat DataFrame baru yang terdiri dari lima baris pertama dari DataFrame df dan kolom-kolom tertentu yaitu 'Code', 'Name', 'ListingDate', 'Shares', 'LastPrice', dan 'MarketCap'.

Secara detail, kode tersebut melakukan beberapa hal sebagai berikut:

➢ *df.loc*: method untuk melakukan indexing DataFrame dengan label (misalnya nama kolom atau nama indeks).

106_Mohamad Ibnu Fajar Maulana

- *[i for i in range(0,5)]*: sebuah list comprehension yang digunakan untuk membuat list dari nilai indeks baris yang akan dipilih. Pada contoh ini, list tersebut berisi nilai indeks baris dari 0 sampai dengan 4, yaitu lima baris pertama dari DataFrame df.
- *['Code','Name','ListingDate','Shares','LastPrice', 'MarketCap']*: list yang berisi nama-nama kolom yang akan dipilih. Hanya kolom dengan nama yang tercantum dalam list ini yang akan diambil untuk DataFrame baru.
- *df_subset*: variabel yang digunakan untuk menyimpan DataFrame baru yang terdiri dari lima baris pertama dan kolom-kolom tertentu dari DataFrame.

- Kemudian mendapatkan output:

[31]:

	Code	Name	ListingDate	Shares	LastPrice	MarketCap
0	AALI	Astra Agro Lestari Tbk.	1997-12-09	1924688333	8250.0	1.587868e+13
1	ABBA	Mahaka Media Tbk.	2002-04-03	3935892857	214.0	8.422811e+11
2	ABDA	Asuransi Bina Dana Arta Tbk.	1989-07-06	620806680	6025.0	3.740360e+12
3	ABMM	ABM Investama Tbk.	2011-12-06	2753165000	3960.0	1.090253e+13
4	ACES	Ace Hardware Indonesia Tbk.	2007-11-06	17150000000	610.0	1.046150e+13

- Gambar diatas merupakan data baru dan hasil dari teknik subsetting.
- Kemudian buat data baru kembali untuk mendeskripsikan data harga penutupan saham dan nilai market saham perusahaan dengan code *df.describe*, seperti gambar dibawah ini:

[32]:
```
df_subset = df.loc[
 [i for i in range(0, 48)],
    ['LastPrice', 'MarketCap']]
df_subset.describe()
```

- Kemudian mendapatkan output:

[32]:

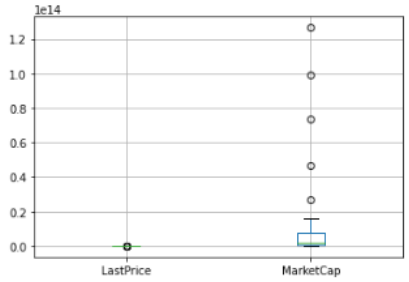
	LastPrice	MarketCap
count	48.000000	4.800000e+01
mean	1389.166667	1.054186e+13
std	2154.504508	2.525972e+13
min	50.000000	5.324000e+10
25%	179.750000	3.835330e+11
50%	356.000000	1.805281e+12
75%	1421.250000	7.479988e+12
max	8300.000000	1.266644e+14

- Gambar diatas dengan teknik subsetting sehingga jadi data baru dan juga mendiskripsikan secara statistik pada kolom *LastPrice*(harga penutupan saham) dan kolom *MarketCap*(nilai saham pada pasar).

106_Mohamad Ibnu Fajar Maulana

- Kemudian perbandingan 2 kolom tersebut dapat divisualisasikan boxplot akan tetapi, tetap menggunakan teknik subsetting, dengan kodingan dibawah ini:

```
[33]: import matplotlib as plt
      boxplot = df_subset.boxplot()
```



- Gambar diatas merupakan *input-output* untuk memvisualisasikan perbandingan harga penutupan saham dan nilai saham pada pasar.

4. Outliers

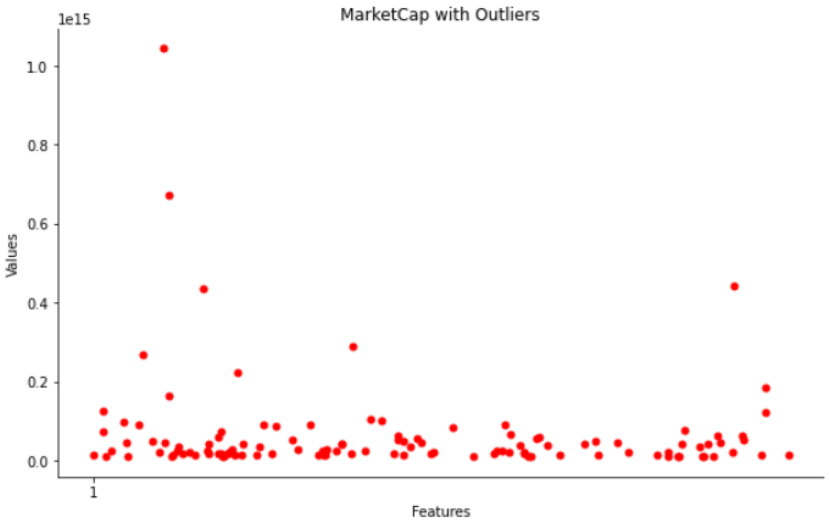
- Lakukan kodingan dibawah ini untuk menemukan outliers pada kolom *MarketCap*:

```
[39]: import pandas as pd
      import matplotlib.pyplot as plt

      # Load data from CSV file
      df = pd.read_csv('DaftarSaham.csv')

      # Define the column(s) to check for outlier
      column = 'MarketCap'

      # Calculate IQR for the selected column
      q1 = df[column].quantile(0.25)
      q3 = df[column].quantile(0.75)
      iqr = q3 - q1

      # Define upper and lower limits to identify outliers
      lower_limit = q1 - 1.5 * iqr
      upper_limit = q3 + 1.5 * iqr

      # Find the outlier values
      outliers = df[(df[column] < lower_limit) | (df[column] > upper_limit)]

      # Create a boxplot with outliers
      fig, ax = plt.subplots(figsize=(10, 6))
      ax.boxplot(df[column], showfliers=True)

      # Add a red marker for each outlier value
      ax.plot(list(outliers.index), outliers[column], 'ro', markersize=5)

      # Set the title and labels
      ax.set_title("Boxplot with Outliers")
      ax.set_xlabel("Features")
      ax.set_ylabel("Values")

      # Remove the top and right spines
      ax.spines['top'].set_visible(False)
      ax.spines['right'].set_visible(False)

      # Show the plot
      plt.show()
```

- Kode di atas digunakan untuk melakukan deteksi dan visualisasi outlier pada data pasar saham. Secara detail, kode tersebut melakukan beberapa hal sebagai berikut:
  - *import pandas as pd dan import matplotlib.pyplot as plt*: Mengimport modul pandas dan matplotlib untuk membaca data dan membuat visualisasi.
  - *df = pd.read_csv('DaftarSaham.csv')*: Membaca data dari file CSV yang bernama DaftarSaham.csv dan menyimpannya ke dalam DataFrame df.
  - *column = 'MarketCap'*: Mengambil kolom MarketCap sebagai kolom yang akan diperiksa terhadap outlier.

- *q1 = df[column].quantile(0.25), q3 = df[column].quantile(0.75), dan iqr = q3 - q1*: Menghitung nilai kuartil pertama (Q1), kuartil ketiga (Q3), dan interquartile range (IQR) pada kolom MarketCap.
- *lower_limit = q1 - 1.5 * iqr dan upper_limit = q3 + 1.5 * iqr*: Menghitung nilai batas bawah (lower limit) dan batas atas (upper limit) untuk mendeteksi nilai yang dianggap sebagai outlier pada kolom MarketCap.
- *outliers = df[(df[column] < lower_limit) | (df[column] > upper_limit)]*: Mencari nilai-nilai yang dianggap sebagai outlier pada kolom MarketCap.
- *fig, ax = plt.subplots(figsize=(10, 6)) dan ax.boxplot(df[column], showfliers=True)*: Membuat boxplot dari kolom MarketCap yang menampilkan nilai outlier.
- *ax.plot(list(outliers.index), outliers[column], 'ro', markersize=5)*: Menandai nilai-nilai outlier dengan tanda titik merah pada boxplot.
- *ax.set_title("Boxplot with Outliers"), ax.set_xlabel("Features"), dan ax.set_ylabel("Values")*: Memberikan judul dan label pada sumbu-sumbu grafik.
- *ax.spines['top'].set_visible(False) dan ax.spines['right'].set_visible(False)*: Menghilangkan sumbu atas dan kanan pada grafik.
- *plt.show()*: Menampilkan boxplot yang telah dibuat.

- Kemudian mendapatkan output:



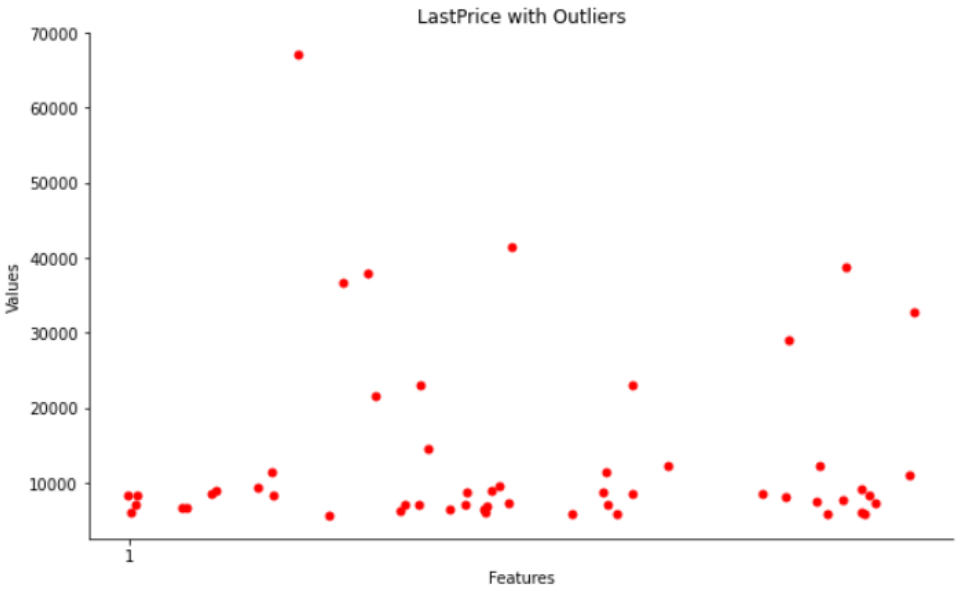
106_Mohamad Ibnu Fajar Maulana

- Gambar diatas terlihat beberapa outliers pada *MarketCap* namun, tidak terlalu menyebar yang artinya nilai pasar yang terjadi pada beberapa saham saling bersaing satu sama lain.
- Kemudian kita harus menemukan outliers pada kolom *LastPrice* karena saling berhubungan dengan kolom *MarketCap.*
- Lakukan kodingan seperti dibawah ini untuk menemukan outliers:

[60]:
```python
import pandas as pd
import matplotlib.pyplot as plt

# Load data from CSV file
df = pd.read_csv('DaftarSaham.csv')

# Define the column(s) to check for outlier
column = 'LastPrice'

# Calculate IQR for the selected column
q1 = df[column].quantile(0.15)
q3 = df[column].quantile(0.85)
iqr = q3 - q1

# Define upper and lower limits to identify outliers
lower_limit = q1 - 1.5 * iqr
upper_limit = q3 + 1.5 * iqr

# Find the outlier values
outliers = df[(df[column] < lower_limit) | (df[column] > upper_limit)]

# Create a boxplot with outliers
fig, ax = plt.subplots(figsize=(10, 6))
ax.boxplot(df[column], showfliers=True)

# Add a red marker for each outlier value
ax.plot(list(outliers.index), outliers[column], 'ro', markersize=5)

# Set the title and labels
ax.set_title("LastPrice with Outliers")
ax.set_xlabel("Features")
ax.set_ylabel("Values")

# Remove the top and right spines
ax.spines['top'].set_visible(False)
ax.spines['right'].set_visible(False)

# Show the plot
plt.show()
```

- Gambar diatas sama dengan penjelasan untuk menemukan outliers pada kolom *MarketCap*, sama-sama menggunakan konsep IQR (interquartile range) dan definisi upper dan lower limits. Upper dan lower limits didefinisikan sebagai 1,5 * IQR di atas Q3 dan di bawah Q1. Data yang di luar upper dan lower limits dianggap sebagai outlier. Kemudian outlier ditandai dengan titik merah pada boxplot dan ditampilkan pada plot.
- Yang membedakan adalah nilai quartilnya yang mana pada *MarketCap* memiliki nilai $quartil\ (q1) = 0.25\ dan\ (q3) = 0.75$, sedangkan untuk *LastPrice* memiliki nilai $quartil\ (q1) = 0.15\ dan\ (q3) = 0.85$, sehingga mendapatkan output outliers seperti gambar dibawah ini:

106_Mohamad Ibnu
Fajar Maulana



- Gambar diatas memvisualisasikan titik-titik merah pada outliers *LastPrice* yang artinya harga penutupan pada saham saling berbeda secara signifikan satu sama lain.

5. Fungsi GroupBy
   - Load data baru dengan teknik subsetting yang sudah dijelaskan diatas, seperti gambar dibawah ini:

```
[40]: df_subset = df.loc[[i for i in range (10)],\
                         ['ListingBoard','Sector','LastPrice']]
      df_subset
```

   - Gambar diatas memanggil 10 data dan menggabungkan kolom *ListingBoard, Sector,* dan *LastPrice*.
   - Kemudian mendapatkan output:

[40]:

	ListingBoard	Sector	LastPrice
0	Utama	Consumer Non-Cyclicals	8250.0
1	Pengembangan	Consumer Cyclicals	214.0
2	Pengembangan	Financials	6025.0
3	Utama	Industrials	3960.0
4	Utama	Consumer Cyclicals	610.0
5	Utama	Infrastructures	176.0
6	Utama	Properties & Real Estate	81.0
7	Pengembangan	Consumer Non-Cyclicals	7175.0
8	Utama	Infrastructures	715.0
9	Utama	Financials	8300.0

- Kemudian membuat DataFrame dengan method *GroupBy*, dengan kode seperti gambar dibawah ini:

```
[41]: bySector = df_subset.groupby('Sector')
      bySector
```

```
[41]: <pandas.core.groupby.generic.DataFrameGroupBy object at 0x000001FE472F1C10>
```

- Gambar diatas berdasarkan kolom *Sector*
- Kemudian kita menghitung rata-rata harga penutupan saham berdasarkan *Sector*, dengan kode *BySector.mean()*. Sebagai berikut:

```
[42]: print('\033[1m', '\33[32m')
      print(" Data kelompok berdasarkan sektor dan harga rata rata penutupan saham")
      bySector.mean()
```

- Sehingga mendapatkan output:

```
 Data kelompok berdasarkan sektor dan harga rata rata penutupan saham
```

[42]:

Sector	LastPrice
Consumer Cyclicals	412.0
Consumer Non-Cyclicals	7712.5
Financials	7162.5
Industrials	3960.0
Infrastructures	445.5
Properties & Real Estate	81.0

- Gambar diatas berdasarkan sektornaya seperti sektor industri, keunagan dll (silahkan lihat gambar) dapat diketahui harga rata -rata penutupan saham.
- Selanjutnya kita dapat menghitung total harga penutupan saham dari berbagai sektor dengan kode *bySector.sum()*, sebagai berikut:

```
[43]: print('\033[1m', '\33[32m')
      print("Data kelompok berdasarkan sektor dan total harga penutupan saham")
      bySector.sum()
```

- Kemudian mendapatkan output:

```
Data kelompok berdasarkan sektor dan total harga penutupan saham
```

[43]:

Sector	LastPrice
Consumer Cyclicals	824.0
Consumer Non-Cyclicals	15425.0
Financials	14325.0
Industrials	3960.0
Infrastructures	891.0
Properties & Real Estate	81.0

- Setelah itu, kita melihat pendeskripsian secara statistik pada sektor keuangan(Financials), dengan kode berikut ini:

```
[44]: pd.DataFrame(bySector.describe().loc['Financials'])
```

- Kemudian mendapatkan hasil:

[44]:

		Financials
LastPrice	count	2.000000
	mean	7162.500000
	std	1608.667927
	min	6025.000000
	25%	6593.750000
	50%	7162.500000
	75%	7731.250000
	max	8300.000000

- Pada gambar diatas pendeskripsian nilai min-max, rata rata dan standar deviasi pada sektor keuangan.
- Kemudian jika ingin mengetahui informasi terkait saham pada bursa efek dalam tahap pengembangan atau utama maka lakukan kodingan dibawah ini dengan menggunakan metode *Groupby:*

```
[45]: df_subset.groupby('ListingBoard').describe()\
      .loc[['Pengembangan','Utama']]
```

- Setelah itu mendapatkan data baru atau informasi baru mengenai daftar atau board saham pada bursa efek secara statistik.

[63]:

	LastPrice							
	count	mean	std	min	25%	50%	75%	max
ListingBoard								
Pengembangan	3.0	4471.333333	3731.526542	214.0	3119.5	6025.0	6600.0	7175.0
Utama	7.0	3156.000000	3737.927456	81.0	393.0	715.0	6105.0	8300.0

- Kemudian kita menggunakan metode *Groupby* untuk membuat informasi atau data baru berdasarkan nama, sektor dan harga penutupan sahamnya secara statistik dengan kode berikut:

```
[64]: byNameSector=df.groupby(['Name','Sector'])
      byNameSector.describe()['LastPrice']
```

- Kemudian mendapatkan hasil informasi baru secara statistik sebagai berikut:

[64]:

Name	Sector	count	mean	std	min	25%	50%	75%	max
ABM Investama Tbk.	Industrials	1.0	3960.0	NaN	3960.0	3960.0	3960.0	3960.0	3960.0
AKR Corporindo Tbk.	Energy	1.0	1350.0	NaN	1350.0	1350.0	1350.0	1350.0	1350.0
Ace Hardware Indonesia Tbk.	Consumer Cyclicals	1.0	610.0	NaN	610.0	610.0	610.0	610.0	610.0
Ace Oldfields Tbk.	Industrials	1.0	80.0	NaN	80.0	80.0	80.0	80.0	80.0
Acset Indonusa Tbk.	Infrastructures	1.0	176.0	NaN	176.0	176.0	176.0	176.0	176.0
...	...	...	...	...	...	...	...	...	...
XL Axiata Tbk.	Infrastructures	1.0	2460.0	NaN	2460.0	2460.0	2460.0	2460.0	2460.0
Yanaprima Hastapersada Tbk	Basic Materials	1.0	765.0	NaN	765.0	765.0	765.0	765.0	765.0
Yelooo Integra Datanet Tbk.	Consumer Cyclicals	1.0	90.0	NaN	90.0	90.0	90.0	90.0	90.0
Yulie Sekuritas Indonesia Tbk.	Financials	1.0	2100.0	NaN	2100.0	2100.0	2100.0	2100.0	2100.0
Zyrexindo Mandiri Buana Tbk.	Technology	1.0	474.0	NaN	474.0	474.0	474.0	474.0	474.0

810 rows × 8 columns

- Pada gambar diatas ktia melihat informasi beberapa perusahan dan sektornya yang memiliki harga penutupan saham secara statistik, jadi kita bisa mengetahui nilai minimum dan maksimum, rata-rata dan sebagainya mengenai harga penutupan saham. Akan tetapi, nilai standar deviasi harga penutupan saham(LastPrice) terdapat *missing value*.
- Tenang saja walaupun terdapat *missing value* hal tersebut bisa diatasi dengan kode skrip berikut ini:

[66]:
```
byNameSector = df.groupby(['Name','Sector'])
mean = byNameSector.describe()['LastPrice']['mean'].mean()
byNameSector.describe()['LastPrice'].fillna(mean)
```

- *.fillna(mean)* pada kode tersebut digunakan untuk mengisi nilai yang hilang (NaN) pada suatu objek dengan nilai rata-rata (mean) dari objek tersebut. Dalam kasus ini, fillna(mean) digunakan untuk mengisi nilai-nilai NaN pada hasil keluaran describe()['LastPrice'] yang sebelumnya telah diambil hanya kolom 'mean'-nya saja. Oleh karena itu, setiap nilai NaN pada kolom 'mean' akan diganti dengan nilai rata-rata dari kolom 'mean' tersebut. Dalam hal ini, mean mengacu pada rata-rata nilai dari kolom 'mean' hasil keluaran sebelumnya.

Dengan demikian, penggunaan fillna(mean) pada kode di atas bertujuan untuk memperbaiki kelengkapan data sehingga data dapat digunakan secara lebih akurat dan lengkap dalam analisis selanjutnya.

- Sehingga mendapatkan output:

[66]:

		count	mean	std	min	25%	50%	75%	max
Name	**Sector**								
ABM Investama Tbk.	**Industrials**	1.0	3960.0	1591.681046	3960.0	3960.0	3960.0	3960.0	3960.0
AKR Corporindo Tbk.	**Energy**	1.0	1350.0	1591.681046	1350.0	1350.0	1350.0	1350.0	1350.0
Ace Hardware Indonesia Tbk.	**Consumer Cyclicals**	1.0	610.0	1591.681046	610.0	610.0	610.0	610.0	610.0
Ace Oldfields Tbk.	**Industrials**	1.0	80.0	1591.681046	80.0	80.0	80.0	80.0	80.0
Acset Indonusa Tbk.	**Infrastructures**	1.0	176.0	1591.681046	176.0	176.0	176.0	176.0	176.0
...	...	...	...	...	...	...	...	...	...
XL Axiata Tbk.	**Infrastructures**	1.0	2460.0	1591.681046	2460.0	2460.0	2460.0	2460.0	2460.0
Yanaprima Hastapersada Tbk	**Basic Materials**	1.0	765.0	1591.681046	765.0	765.0	765.0	765.0	765.0
Yelooo Integra Datanet Tbk.	**Consumer Cyclicals**	1.0	90.0	1591.681046	90.0	90.0	90.0	90.0	90.0
Yulie Sekuritas Indonesia Tbk.	**Financials**	1.0	2100.0	1591.681046	2100.0	2100.0	2100.0	2100.0	2100.0
Zyrexindo Mandiri Buana Tbk.	**Technology**	1.0	474.0	1591.681046	474.0	474.0	474.0	474.0	474.0

810 rows × 8 columns

- Nilai standar deviasi tersedia pada harga penutupan saham(LastPrice)

106_Mohamad Ibnu Fajar Maulana