

Team AIRy - COMPFEST 15  
Data Science Academy

# Banjir Jakarta 2018 dan 2020

Model Regresi Pengungsi Tertinggi dan Prediksi  
Clustering Resiko Wilayah Banjir di DKI Jakarta



# Anggota Team

01.



**M. Alvero Johansyah**

02.



**M. Irfansyah**

03.



**Rifki Prakasya**

# List of Content

06

**Kesimpulan Saran**

Kesimpulan serta saran terkait permasalahan

01

**Rumusan Masalah**

Permasalahan yang ingin dicari jawabannya

02

**Hipotesis**

Jawaban sementara dari permasalahan

03

**Data Assesment & Preprocessing**

Penjelasan data yang digunakan dan proses preprocessing data

04

**Exploratory Data Analysis (EDA)**

Penjelasan mengenai EDA dari data

05

**Feature Engineering & Modelling**

Melakukan modeling terhadap data dan feature engineering

# List of Content

01

## Rumusan Masalah

Permasalahan yang ingin dicari jawabannya

# Rumusan Masalah

Terdapat Tiga rumusan masalah:

1. Bagaimana hubungan antara jumlah pengungsi tertinggi dengan jumlah luka ringan korban dalam keadaan banjir di DKI Jakarta Tahun 2020?
2. Bagaimana prediksi kondisi banjir DKI Jakarta per kota administrasi pada bulan januari 2021?
3. Bagaimana prediksi keadaan kelembaban, temperatur curah hujan, dan hari hujan DKI Jakarta yang memengaruhi kondisi banjir berdasarkan kota administrasi pada bulan januari 2019?



# List of Content

02

## Hipotesis

Jawaban sementara dari  
permasalahan

# Hipotesis

Kami memiliki dugaan awal atas jawaban dari rumusan masalah yang telah dirumuskan sebagai berikut.

1. Hubungan antara jumlah pengungsi tertinggi dengan jumlah luka ringan korban pada bencana banjir jakarta tahun 2020 sangat baik yang menunjukkan bahwa semakin tinggi jumlah pengungsi maka makin tinggi juga korban yang mengalami luka ringan
2. Kondisi banjir untuk setiap kota administrasi di wilayah DKI Jakarta pada bulan januari 2021 mengalami kenaikan dalam hal ketinggian air dan jumlah terdampak jiwa
3. Kondisi kelembaban, temperatur curah hujan, dan hari hujan yang memengaruhi kondisi banjir untuk setiap kota administrasi di wilayah DKI Jakarta pada bulan januari 2019 mengalami kenaikan dalam hal ketinggian air dan faktor alam



# List of Content

03

Data Assesment &  
Preprocessing

Penjelasan data yang  
digunakan dan proses  
preprocessing data

# Data Assesment & Preprocessing

## Informasi DataSet

Kami mengumpulkan dataset Data Kejadian Bencana Banjir di Provinsi DKI Jakarta Tahun 2018 dan 2020 setiap bulannya sebagai dataset acuan. Dataset ini berisi 18 Fitur dan 931 Baris.

Kemudian kami memutuskan untuk mendapatkan informasi lain untuk mengkaji lebih lanjut mengenai banjir dki jakarta berupa:

1. Jumlah Penduduk DKI Jakarta Tahun 2018 dan 2020 berdasarkan kota
2. Kelembapan, Temperatur, Curah Hujan, dan Banyaknya Hari Hujan perBulan di DKI Jakarta 2018

Kab/Kota	Jumlah Penduduk Menurut Kabupaten/Kota di Provinsi DKI Jakarta (Jiwa)		
	2020	2021	2022
Kep Seribu	27 749	28 240	28 240
Jakarta Selatan	2 226 812	2 233 855	2 244 812
Jakarta Timur	3 037 139	3 056 300	3 083 139
Jakarta Pusat	1 056 896	1 066 460	1 079 896
Jakarta Barat	2 434 511	2 440 073	2 448 511
Jakarta Utara	1 778 981	1 784 753	1 793 981
DKI Jakarta	10 562 088	10 609 681	10 679 088

Sumber:  
<https://jakarta.bps.go.id/indicator/12/1270/1/jumlah-penduduk-menurut-kabupaten-kota-di-provinsi-dki-jakarta-.html>

### Data Kejadian Bencana Banjir di Provinsi DKI Jakarta Tahun 2018

Dataset ini berisi tentang Data Rekapitulasi Bulanan Kejadian Banjir di Provinsi DKI Jakarta Tahun 2018.

Penjelasan mengenai variabel pada dataset ini :

1. kota\_administrasi : nama kota di Provinsi DKI Jakarta yang terkena dampak banjir
2. kecamatan : nama kecamatan yang terkena dampak banjir
3. kelurahan : nama kelurahan yang terkena dampak banjir
4. rw : nama RW yang terkena dampak banjir
5. jumlah\_terdampak\_rw : jumlah RW yang terkena dampak banjir
6. jumlah\_terdampak\_rt : jumlah RT yang terkena dampak banjir
7. jumlah\_terdampak\_kk : jumlah kepala keluarga yang terkena dampak banjir
8. jumlah\_terdampak\_jiwa : jumlah orang yang terkena dampak banjir
9. ketinggian\_air : ketinggian air pada saat kejadian banjir (cm)
10. tanggal\_kejadian : tanggal kejadian banjir
11. lama\_genangan : lama genangan saat banjir (hari)
12. jumlah\_meninggal : jumlah korban meninggal saat kejadian banjir
13. jumlah\_hilang : jumlah korban hilang saat kejadian banjir
14. jumlah\_luka\_berat : jumlah korban luka berat saat kejadian banjir
15. jumlah\_luka\_ringan : jumlah korban luka ringan saat kejadian banjir
16. jumlah\_pengungsingan : jumlah pengungsian banjir
17. jumlah\_tempat\_pengungsian : jumlah tempat pengungsian korban banjir
18. nilai\_kerugian : jumlah nilai kerugian akibat kejadian banjir

Sumber: [Data.jakarta.go.id](https://Data.jakarta.go.id)



Sumber:  
<https://statistik.jakarta.go.id/tabel/kelembapan-temperatur-curah-hujan-dan-hari-hujan-di-dki-jakarta-2018/>

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Import Modul & Dataset

```
▶ import pandas as pd
  from sklearn.linear_model import LinearRegression
  from sklearn.model_selection import train_test_split
  from sklearn.metrics import mean_squared_error

[ ] januari = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-januari.csv')
februari = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-februari.csv')
maret = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-maret.csv')
april = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-april.csv')
mei = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-meい.csv')
juni = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-juni.csv')
juli = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-juli.csv')
agustus = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-agustus.csv')
september = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-september.csv')
oktober = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-oktober.csv')
november = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-november.csv')
desember = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2020-bulan-desember.csv')
```

Import Modul dan dataset kejadian banjir DKI Jakarta 2020 dari bulan januari sampai desember.

### Menggabungkan beberapa data frame menjadi satu

```
[ ] # List DataFrame yang sudah Anda baca
dataframes = [januari, februari, maret, april, mei, juni, juli, agustus, oktober, november]

# Menggabungkan DataFrame menjadi satu DataFrame
df = pd.concat(dataframes)

# Reset indeks DataFrame yang telah digabungkan
df.reset_index(drop=True, inplace=True)
```

Gabungkan beberapa dataset menjadi satu dataframe dari januari hingga desember.

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Dataframe yang baru

```
[ ] df
```

	kota_administrasi	kecamatan	kelurahan	rw	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	ketinggian_air	tanggal_kejadian	lama_genangan	jumlah_meninggal	jumlah_hilang
0	Jakarta Pusat	JOHAR BARU	JOHAR BARU	RW 01, 02, 06, 08	4	4	35	140	10 s/d 30 cm	tgl. 01 Januari	0	0	
1	Jakarta Pusat	KEMAYORAN	GUNUNG SAHARI SELATAN	RW 01, 02, 07, 08	4	4	0	0	10 s/d 70 cm	tgl. 01 Januari	0	0	
2	Jakarta Pusat	KEMAYORAN	SERDANG	RW 01, 02, 06, 07	4	4	0	0	10 s/d 30 cm	tgl. 01 Januari	1	0	
3	Jakarta Pusat	SAWAH BESAR	PASAR BARU	RW 02, 03, 04, 05, 06, 07, 08	7	7	155	625	10 s/d 70 cm	tgl. 01 Januari	0	0	
4	Jakarta Pusat	TANAH ABANG	BENDUNGAN HILIR	RW 07, 09	2	2	52	195	31 s/d 70 cm	tgl. 01 Januari	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...
926	Jakarta Timur	KRAMAT JATI	BALEKAMBANG	RW :03	1	1	0	0	20 cm	2020-11-08	0	0	
927	Jakarta Timur	KRAMAT JATI	CAWANG	RW :03	1	1	0	0	50 cm	2020-11-12	0	0	
928	Jakarta Timur	KRAMAT JATI	CAWANG	RW :05	1	1	0	0	30 cm	2020-11-12	0	0	
929	Jakarta Timur	KRAMAT JATI	CAWANG	0	0	1	0	0	30 cm	2020-11-12	0	0	
930	Jakarta Timur	PASAR REBO	GEDONG	RW :010	1	1	0	0	20 cm	2020-11-25	0	0	

931 rows x 18 columns

### Ukuran data frame

```
[ ] df.shape
```

(931, 18)

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Informasi mengenai data

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 931 entries, 0 to 930
Data columns (total 18 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   kota_administrasi    931 non-null   object  
 1   kecamatan          931 non-null   object  
 2   kelurahan          931 non-null   object  
 3   rw                 931 non-null   object  
 4   jumlah_terdampak_rw 931 non-null   int64  
 5   jumlah_terdampak_rt 931 non-null   int64  
 6   jumlah_terdampak_kk 931 non-null   object  
 7   jumlah_terdampak_jiwa 931 non-null   int64  
 8   ketinggian_air      931 non-null   object  
 9   tanggal_kejadian    931 non-null   object  
 10  lama_genangan      931 non-null   int64  
 11  jumlah_meninggal    931 non-null   int64  
 12  jumlah_hilang       931 non-null   int64  
 13  jumlah_luka_berat   931 non-null   int64  
 14  jumlah_luka_ringan  931 non-null   int64  
 15  jumlah_pengungsi_tertinggi 931 non-null   int64  
 16  jumlah_tempat_pengungsian 931 non-null   int64  
 17  nilai_kerugian     931 non-null   int64  
dtypes: int64(11), object(7)
memory usage: 131.0+ KB
```

### Nama kolom pada data

```
df.columns
```

```
Index(['kota_administrasi', 'kecamatan', 'kelurahan', 'rw',
       'jumlah_terdampak_rw', 'jumlah_terdampak_rt',
       'jumlah_terdampak_kk',
       'jumlah_terdampak_jiwa', 'keterangan_air', 'tanggal_kejadian',
       'lama_genangan', 'jumlah_meninggal', 'jumlah_hilang',
       'jumlah_luka_berat', 'jumlah_luka_ringan',
       'jumlah_pengungsi_tertinggi',
       'jumlah_tempat_pengungsian', 'nilai_kerugian'],
      dtype='object')
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Head data

```
[ ] df.head()
```

	kota_administrasi	kecamatan	kelurahan	rw	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	ketinggian_air	tanggal_kejadian	lama_genangan	jumlah_meninggal	jumlah_hilang	jumlah_luka_berat	jumlah
0	Jakarta Pusat	JOHAR BARU	JOHAR BARU RW 01, 02, 06, 08	4	4	35	140	10 s/d 30 cm	tg. 01 Januari	0	0	0	0	0	0
1	Jakarta Pusat	KEMAYORAN	GUNUNG SAHARI SELATAN RW 01, 02, 07, 08	4	4	0	0	10 s/d 70 cm	tg. 01 Januari	0	0	0	0	0	0
2	Jakarta Pusat	KEMAYORAN	SERDANG RW 01, 02, 06, 07	4	4	0	0	10 s/d 30 cm	tg. 01 Januari	1	0	0	0	0	0
3	Jakarta Pusat	SAWAH BESAR	PASAR BARU RW 02, 03, 04, 05, 06, 07, 08	7	7	155	625	10 s/d 70 cm	tg. 01 Januari	0	0	0	0	0	0
4	Jakarta Pusat	TANAH ABANG	BENDUNGAN HILIR RW 07, 09	2	2	52	195	31 s/d 70 cm	tg. 01 Januari	0	0	0	0	0	0

5 data awal

### Tail data

```
[ ] df.tail()
```

	kota_administrasi	kecamatan	kelurahan	rw	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	ketinggian_air	tanggal_kejadian	lama_genangan	jumlah_meninggal	jumlah_hilang	jumlah_luka_berat	jumlah
926	Jakarta Timur	KRAMAT JATI	BALEKAMBANG RW 03	1	1	0	0	20 cm	2020-11-08	0	0	0	0	0	0
927	Jakarta Timur	KRAMAT JATI	CAWANG RW 03	1	1	0	0	50 cm	2020-11-12	0	0	0	0	0	0
928	Jakarta Timur	KRAMAT JATI	CAWANG RW 05	1	1	0	0	30 cm	2020-11-12	0	0	0	0	0	0
929	Jakarta Timur	KRAMAT JATI	CAWANG 0	0	1	0	0	30 cm	2020-11-12	0	0	0	0	0	0
930	Jakarta Timur	PASAR PINGGIR	GEDONG RW 1	1	1	0	0	20 cm	2020-11-25	0	0	0	0	0	0

5 data akhir

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Cek missing value

```
df.isna().sum()  
kota_administrasi      0  
kecamatan            0  
kelurahan             0  
rw                   0  
jumlah_terdampak_rw  0  
jumlah_terdampak_rt  0  
jumlah_terdampak_kk  0  
jumlah_terdampak_jiwa 0  
ketinggian_air        0  
tanggal_kejadian      0  
lama_genangan        0  
jumlah_meninggal      0  
jumlah_hilang         0  
jumlah_luka_berat    0  
jumlah_luka_ringan    0  
jumlah_pengungsi_tertinggi 0  
jumlah_tempat_pengungsian 0  
nilai_kerugian        0  
dtype: int64
```

Tidak ada missing value / data yang kosong dari dataframe

### Jumlah data di setiap kolom

```
df.sum()  
<ipython-input-205-7e5fdb616c56>:1: FutureWarning: The default value of numeric_only in DataFrame.sum is deprecated. In a future version, it will defa  
df.sum()  
kota_administrasi      Jakarta Pusat  
kecamatan            JOHAR BARUKEMAYORAN KEMAYORAN SAWAH BESARTANAH A...  
kelurahan             JOHAR BARUGUNUNG SAHARI SELATANSERDANGPASAR BA...  
rw                   RW 01, 02, 06, 08  
jumlah_terdampak_rw  1783  
jumlah_terdampak_rt  3256  
jumlah_terdampak_kk  148898  
jumlah_terdampak_jiwa 10 s/d 30 cm  
ketinggian_air        10 s/d 30 cm  
tanggal_kejadian      10 s/d 30 cm  
lama_genangan        10 s/d 30 cm  
jumlah_meninggal      10 s/d 30 cm  
jumlah_hilang         10 s/d 30 cm  
jumlah_luka_berat    10 s/d 30 cm  
jumlah_luka_ringan   10 s/d 30 cm  
jumlah_pengungsi_tertinggi 10 s/d 30 cm  
jumlah_tempat_pengungsian 10 s/d 30 cm  
nilai_kerugian        10 s/d 30 cm  
dtype: object
```

Mengecek jumlahan data

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Describe data

```
] df.describe()
```

	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_jiwa	lama_genangan	jumlah_meninggal	jumlah_hilang	jumlah_luka_berat	jumlah_luka_ringan
count	931.000000	931.000000	931.000000	931.000000	931.0	931.0	931.0	931.000000
mean	1.915145	3.497315	159.933405	0.020408	0.0	0.0	0.0	96.831364
std	2.396300	5.497482	706.077628	0.155930	0.0	0.0	0.0	372.279960
min	0.000000	1.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000
25%	0.000000	1.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000
50%	1.000000	1.000000	0.000000	0.000000	0.0	0.0	0.0	0.000000
75%	2.000000	3.000000	16.500000	0.000000	0.0	0.0	0.0	0.000000
max	15.000000	46.000000	13450.000000	2.000000	0.0	0.0	0.0	4461.000000

### Cek data duplikat

```
[ ] df.duplicated().sum()
```

77

Dapat dilihat terdapat 77 data yang duplikat

## Data Assesment & Preprocessing-Bagian data cleaning

### Rumusan Masalah 1 & 2

#### Menghapus data duplikat

```
[ ] # Menghapus data duplikat dari DataFrame  
df.drop_duplicates(inplace=True)  
  
# Reset indeks DataFrame setelah menghapus duplikat  
df.reset_index(drop=True, inplace=True)
```

#### Cek data duplikat lagi

```
df.duplicated().sum() #mengecek kembali data yang duplikat
```

0

# Data Assesment & Preprocessing-Bagian data cleaning

## Rumusan Masalah 1 & 2

### Menghapus kolom yang 100% kosong / tidak memiliki nilai / tidak begitu penting

```
[ ] df = df.drop(columns=['jumlah_meninggal', 'jumlah_hilang', 'jumlah_luka_berat', 'nilai_kerugian', 'tanggal_kejadian', 'rw', 'kecamatan', 'kelurahan'], axis=1)  
# menghapus kolom yang 100% kosong / tidak memiliki nilai / tidak begitu penting
```

### Mengubah kolom ketinggian air menjadi integer menggunakan rata-rata ketinggian air

```
[ ] import re  
  
# Membuat fungsi untuk mendapatkan rata-rata dari string ketinggian air  
def get_average_height(height_str):  
    # Menggunakan regular expression untuk mengekstraksi angka dari string  
    heights = re.findall(r'\d+', height_str)  
    # Mengubah angka menjadi tipe integer  
    heights = [int(height) for height in heights]  
    # Menghitung rata-rata ketinggian air  
    average_height = sum(heights) / len(heights)  
    return average_height  
  
# Menggunakan fungsi get_average_height untuk mengubah fitur ketinggian air menjadi numerik  
df['ketinggian_air'] = df['ketinggian_air'].apply(get_average_height)  
  
[ ] df['ketinggian_air'] = df['ketinggian_air'].astype('int64')  
  
[ ] import numpy as np  
  
df['jumlah_terdampak_kk'] = pd.to_numeric(df['jumlah_terdampak_kk'], errors='coerce').fillna(0).astype('int64')
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

Dataframe yang baru setelah proses cleaning

```
[ ] df
```

	kota_administrasi	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	jumlah_luka_ringan	jumlah_pengungsi_tertinggi	jumlah_tempat_pengungsian	lama_genangan	ketinggian_air
0	Jakarta Pusat	4	4	35	0	0	0	0	0	0
1	Jakarta Pusat	4	4	0	0	0	0	0	0	0
2	Jakarta Pusat	4	4	0	0	0	0	0	0	0
3	Jakarta Pusat	7	7	155	0	0	0	0	0	0
4	Jakarta Pusat	2	2	52	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...
849	Jakarta Timur	1	1	0	0	0	0	0	0	0
850	Jakarta Timur	1	1	0	0	0	0	0	0	0
851	Jakarta Timur	1	1	0	0	0	0	0	0	0
852	Jakarta Timur	0	1	0	0	0	0	0	0	0
853	Jakarta Timur	1	1	0	0	0	0	0	0	0

854 rows x 10 columns

Informasi mengenai data setelah proses cleaning

```
[ ] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 854 entries, 0 to 853
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   kota_administrasi    854 non-null   object 
 1   jumlah_terdampak_rw  854 non-null   int64  
 2   jumlah_terdampak_rt  854 non-null   int64  
 3   jumlah_terdampak_kk  854 non-null   int64  
 4   jumlah_terdampak_jiwa 854 non-null   int64  
 5   ketinggian_air       854 non-null   int64  
 6   lama_genangan       854 non-null   int64  
 7   jumlah_luka_ringan  854 non-null   int64  
 8   jumlah_pengungsi_tertinggi 854 non-null   int64  
 9   jumlah_tempat_pengungsian 854 non-null   int64  
dtypes: int64(9), object(1)
memory usage: 66.8+ KB
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

Mengetahui data unik dari kolom kota administrasi

```
# Mengetahui data unik dari kota administrasi

unique_data = df['kota_administrasi'].unique()
print(unique_data)

['Jakarta Pusat' 'Jakarta Utara' 'Jakarta Barat' 'Jakarta Selatan'
 'Jakarta Timur' 'Kepulauan Seribu' 'Jakarta Urata']
```

Jumlah data di setiap kolom setelah proses cleaning

```
df.sum()

kota_administrasi      Jakarta Pusat Jakarta Pusat Jakarta Pusat...
kecamatan             JOHAR BARUKEMAYORAN KEMAYORAN SAWAH BESARTANAH A...
kelurahan            JOHAR BARUGUNUNG SAHARI SELATAN SERDANG PASAR BA...
jumlah_terdampak_rw    1783
jumlah_terdampak_rt    3179
jumlah_terdampak_kk    41666
jumlah_terdampak_jiwa  148898
keterangan_air         36580
lama_genangan          19
jumlah_luka_ringan     90150
jumlah_pengungsian      1327
jumlah_tempat_pengungsian 12
dtype: object
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Mengubah data yang typo

```
df['kota_administrasi'] = df['kota_administrasi'].replace('Jakarta Urata', 'Jakarta Utara')
```

### Mengubah Kepulauan Seribu

```
#Hapus Kota Administrasi Kepulauan Seribu  
df = df[df['kota_administrasi'] != 'Kepulauan Seribu']
```

### Mengetahui data unik dari kolom kota administrasi

```
# Mengetahui data unik dari kota administrasi  
  
unique_data = df['kota_administrasi'].unique()  
print(unique_data)  
  
['Jakarta Pusat' 'Jakarta Utara' 'Jakarta Barat' 'Jakarta Selatan'  
'Jakarta Timur' 'Kepulauan Seribu']
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

Membuat dataframe jumlah penduduk sesuai dengan kota administrasi

```
# MEMBUAT DATAFRAME JUMLAH PENDUDUK JAKARTA 2020 BERDASARKAN KOTA ADMINISTRASI

data = {
    'Kota Administrasi': ['Jakarta Selatan', 'Jakarta Timur', 'Jakarta Pusat', 'Jakarta Barat', 'Jakarta Utara'],
    'Jumlah Penduduk': [2226812, 3037139, 1056896, 2434511, 1778981]
}

df_penduduk = pd.DataFrame(data)

#SUMBER: https://jakarta.bps.go.id/indicator/12/1270/1/jumlah-penduduk-menurut-kabupaten-kota-di-provinsi-dki-jakarta-.html
```

Mensubtitusikan data jumlah penduduk ke data frame

```
mapping = {
    'Jakarta Selatan': 2226812,
    'Jakarta Timur': 3037139,
    'Jakarta Pusat': 1056896,
    'Jakarta Barat': 2434511,
    'Jakarta Utara': 1778981
}

df['jumlah_penduduk'] = df['kota_administrasi'].replace(mapping)
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Tampilan dataframe setelah proses mapping

df	kota_administrasi	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	ketinggian_air	lama_genangan	jumlah_luka_ringan	jumlah_luka_serious	jumlah_luka_berat
0	Jakarta Pusat	4	4	35	140	20	0			
1	Jakarta Pusat	4	4	0	0	40	0			
2	Jakarta Pusat	4	4	0	0	20	1			
3	Jakarta Pusat	7	7	155	625	40	0			
4	Jakarta Pusat	2	2	52	195	50	0			
...	...	...	...	...	...	...	...	...	...	...
849	Jakarta Timur	1	1	0	0	20	0			
850	Jakarta Timur	1	1	0	0	50	0			
851	Jakarta Timur	1	1	0	0	30	0			
852	Jakarta Timur	0	1	0	0	30	0			
853	Jakarta Timur	1	1	0	0	20	0			

### Mengecek Outlier

```
#Mengecek outlier

from scipy import stats

# Menghitung z-score untuk setiap kolom numerik dalam dataframe
z_scores = np.abs(stats.zscore(df.select_dtypes(include=np.number)))

# Menentukan batas z-score untuk mengidentifikasi outlier
threshold = 3

# Menampilkan data yang dianggap sebagai outlier
outliers = df[(z_scores > threshold).any(axis=1)]
print(outliers)
```

## Data Assesment & Preprocessing

### Rumusan Masalah 1 & 2

#### Menghapus Outlier

```
#Menghapus Outlier
```

```
df = df[ (z_scores <= threshold).all(axis=1) ]
```

Jumlah data di setiap kolom setelah menghapus outlier

```
df.sum() #dapat dilihat lama genangan dan jumlah tempat pengungsian bernilai 0
```

kota_administrasi	Jakarta Pusat				
jumlah_terdampak_rw	1267				
jumlah_terdampak_rt	1986				
jumlah_terdampak_kk	13322				
jumlah_terdampak_jiwa	49361				
ketinggian_air	30745				
lama_genangan	0				
jumlah_luka_ringan	31250				
jumlah_pengungsi_tertinggi	361				
jumlah_tempat_pengungsian	0				
jumlah_penduduk	1898793041				

dtype: object

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Menghapus kolom yang jumlah datanya 0

```
df = df.drop(columns=['lama_genangan', 'jumlah_tempat_pengungsian'], axis=1)
```

### Mengecek Outlier lagi

```
#cek outlier lagi

# Menghitung z-score untuk setiap kolom numerik dalam dataframe
z_scores = np.abs(stats.zscore(df.select_dtypes(include=np.number)))

# Menentukan batas z-score untuk mengidentifikasi outlier
threshold = 3

# Menampilkan data yang dianggap sebagai outlier
outliers = df[(z_scores > threshold).any(axis=1)]
print(outliers)
```

### Menghapus Outlier lagi

```
#Menghapus Outlier lagi

df = df[(z_scores <= threshold).all(axis=1)]
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1 & 2

### Mengecek Outlier lagi

```
#cek outlier lagi

# Menghitung z-score untuk setiap kolom numerik dalam dataframe
z_scores = np.abs(stats.zscore(df.select_dtypes(include=np.number)))

# Menentukan batas z-score untuk mengidentifikasi outlier
threshold = 3

# Menampilkan data yang dianggap sebagai outlier
outliers = df[(z_scores > threshold).any(axis=1)]
print(outliers)
```

### Menghapus Outlier lagi

```
#Menghapus Outlier lagi

df = df[(z_scores <= threshold).all(axis=1)]
```

Jumlah data di setiap kolom  
setelah menghapus outlier

```
df.sum()

kota_administrasi      Jakarta Pusat
jumlah_terdampak_rw    Jakarta Utara
jumlah_terdampak_rt      Jakarta...
                           ...
jumlah_terdampak_kk      850
jumlah_terdampak_jiwa    1284
keterangan_air           2814
jumlah_luka_ringan       8941
jumlah_pengungsi_tertinggi 24767
jumlah_penduduk            4275
dtype: object
```

# Data Assesment & Preprocessing

## Rumusan Masalah 2

### Import Modul

```
import pandas as pd
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
```

### Memilih fitur-fitur yang akan digunakan

```
# Pilih fitur-fitur yang akan digunakan
features = ['jumlah_terdampak_rw', 'jumlah_terdampak_rt', 'jumlah_terdampak_kk',
            'jumlah_terdampak_jiwa', 'ketinggian_air', 'jumlah_luka_ringan', 'jumlah_pengungsi_tertinggi', 'jumlah_penduduk']
```

# Data Assesment & Preprocessing

## Rumusan Masalah 2

### Clustering dengan K-Means

```
# Lakukan clustering dengan K-means
kmeans = KMeans(n_clusters=5, random_state=42)
df['cluster'] = kmeans.fit_predict(df[features])
```

### Split data

```
# Split data menjadi atribut (X) dan target (y)
X = df[features]
y = df['cluster']
```

### Standarisasi

```
# Standardisasi Data dengan Standard Scaler
from sklearn.preprocessing import StandardScaler
Scaler=StandardScaler()
X=Scaler.fit_transform(X)

X[0:3]
X.shape
```

### Split data

```
# Split data menjadi train set dan test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

# Data Asesment & Preprocessing

## Rumusan Masalah 3

### Import Modul & Dataset

```
1 import pandas as pd

1 januari = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-januari.csv')
2 februari = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-februari.csv')
3 maret = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-maret.csv')
4 april = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-april.csv')
5 mei = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-mei.csv')
6 juni = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-juni.csv')
7 oktober = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-oktober.csv')
8 november = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-november.csv')
9 desember = pd.read_csv('data-kejadian-bencana-banjir-di-provinsi-dki-jakarta-tahun-2018-bulan-desember.csv')

1 # load dataset baru yang merupakan data kelembapan, temperatur, dll
2 tempdf = pd.read_csv('kelembapan Temperatur Curah Hujan dan Hari Hujan di DKI Jakarta 2018.csv')
3 tempdf
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Membuat List Data Kelembapan, Udara, Curah Hujan, dan Hari Hujan Untuk Setiap Bulan

```
1 #januari
2 rku_januari = []
3 ru_januari = []
4 ch_januari = []
5 hh_januari = []
6 for i in range(januari.shape[0]):
7     rku_januari.append(tempdf['Rata-Rata Kelembapan Udara (%)'][0])
8     ru_januari.append(tempdf['Rata-Rata Udara (.C)'][0])
9     ch_januari.append(tempdf['Curah Hujan (mm)'][0])
10    hh_januari.append(tempdf['Hari Hujan'][0])
11
12 #februari
13 rku_februari = []
14 ru_februari = []
15 ch_februari = []
16 hh_februari = []
17 for i in range(februari.shape[0]):
18     rku_februari.append(tempdf['Rata-Rata Kelembapan Udara (%)'][1])
19     ru_februari.append(tempdf['Rata-Rata Udara (.C)'][1])
20     ch_februari.append(tempdf['Curah Hujan (mm)'][1])
21     hh_februari.append(tempdf['Hari Hujan'][1])
22
23 #maret
24 rku_maret = []
25 ru_maret = []
26 ch_maret = []
27 hh_maret = []
28 for i in range(maret.shape[0]):
29     rku_maret.append(tempdf['Rata-Rata Kelembapan Udara (%)'][2])
30     ru_maret.append(tempdf['Rata-Rata Udara (.C)'][2])
31     ch_maret.append(tempdf['Curah Hujan (mm)'][2])
32     hh_maret.append(tempdf['Hari Hujan'][2])
33
34 #april
35 rku_april = []
36 ru_april = []
37 ch_april = []
38 hh_april = []
39 for i in range(april.shape[0]):
40     rku_april.append(tempdf['Rata-Rata Kelembapan Udara (%)'][3])
41     ru_april.append(tempdf['Rata-Rata Udara (.C)'][3])
42     ch_april.append(tempdf['Curah Hujan (mm)'][3])
43     hh_april.append(tempdf['Hari Hujan'][3])
44
```

```
45 #mei
46 rku_mei = []
47 ru_mei = []
48 ch_mei = []
49 hh_mei = []
50 for i in range(meい.shape[0]):
51     rku_mei.append(tempdf['Rata-Rata Kelembapan Udara (%)'][4])
52     ru_mei.append(tempdf['Rata-Rata Udara (.C)'][4])
53     ch_mei.append(tempdf['Curah Hujan (mm)'][4])
54     hh_mei.append(tempdf['Hari Hujan'][4])
55
56 #juni
57 rku_juni = []
58 ru_juni = []
59 ch_juni = []
60 hh_juni = []
61 for i in range(juni.shape[0]):
62     rku_juni.append(tempdf['Rata-Rata Kelembapan Udara (%)'][5])
63     ru_juni.append(tempdf['Rata-Rata Udara (.C)'][5])
64     ch_juni.append(tempdf['Curah Hujan (mm)'][5])
65     hh_juni.append(tempdf['Hari Hujan'][5])
66
67 #oktober
68 rku_oktober = []
69 ru_oktober = []
70 ch_oktober = []
71 hh_oktober = []
72 for i in range(oktober.shape[0]):
73     rku_oktober.append(tempdf['Rata-Rata Kelembapan Udara (%)'][9])
74     ru_oktober.append(tempdf['Rata-Rata Udara (.C)'][9])
75     ch_oktober.append(tempdf['Curah Hujan (mm)'][9])
76     hh_oktober.append(tempdf['Hari Hujan'][9])
77
78 #november
79 rku_november = []
80 ru_november = []
81 ch_november = []
82 hh_november = []
83 for i in range(november.shape[0]):
84     rku_november.append(tempdf['Rata-Rata Kelembapan Udara (%)'][10])
85     ru_november.append(tempdf['Rata-Rata Udara (.C)'][10])
86     ch_november.append(tempdf['Curah Hujan (mm)'][10])
87     hh_november.append(tempdf['Hari Hujan'][10])
88
```

```
89 #desember
90 rku_desember = []
91 ru_desember = []
92 ch_desember = []
93 hh_desember = []
94 for i in range(desember.shape[0]):
95     rku_desember.append(tempdf['Rata-Rata Kelembapan Udara (%)'][11])
96     ru_desember.append(tempdf['Rata-Rata Udara (.C)'][11])
97     ch_desember.append(tempdf['Curah Hujan (mm)'][11])
98     hh_desember.append(tempdf['Hari Hujan'][11])
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Membuat Dataframe dari List-List Yang Telah Dibuat dan Disatukan Dengan Data Banjir pada Bulan Yang Bersesuaian

```
1 #januari
2 rku_januari = pd.DataFrame(data=rku_januari, columns=['Rata-Rata Kelembapan Udara (%)'])
3 ru_januari = pd.DataFrame(data=ru_januari, columns=['Rata-Rata Udara (.C)'])
4 ch_januari = pd.DataFrame(data=ch_januari, columns=['Curah Hujan (mm)'])
5 hh_januari = pd.DataFrame(data=hh_januari, columns=['Hari Hujan'])
6
7 januaridf = pd.concat([januari, rku_januari, ru_januari, ch_januari, hh_januari], axis=1)
8
9
10 #februari
11 rku_februari = pd.DataFrame(data=rku_februari, columns=['Rata-Rata Kelembapan Udara (%)'])
12 ru_februari = pd.DataFrame(data=ru_februari, columns=['Rata-Rata Udara (.C)'])
13 ch_februari = pd.DataFrame(data=ch_februari, columns=['Curah Hujan (mm)'])
14 hh_februari = pd.DataFrame(data=hh_februari, columns=['Hari Hujan'])
15
16 februaridf = pd.concat([februari, rku_februari, ru_februari, ch_februari, hh_februari], axis=1)
17
18 #maret
19 rku_maret = pd.DataFrame(data=rku_maret, columns=['Rata-Rata Kelembapan Udara (%)'])
20 ru_maret = pd.DataFrame(data=ru_maret, columns=['Rata-Rata Udara (.C)'])
21 ch_maret = pd.DataFrame(data=ch_maret, columns=['Curah Hujan (mm)'])
22 hh_maret = pd.DataFrame(data=hh_maret, columns=['Hari Hujan'])
23
24 maretfdf = pd.concat([maret, rku_maret, ru_maret, ch_maret, hh_maret], axis=1)
```

```
26 #april
27 rku_april = pd.DataFrame(data=rku_april, columns=['Rata-Rata Kelembapan Udara (%)'])
28 ru_april = pd.DataFrame(data=ru_april, columns=['Rata-Rata Udara (.C)'])
29 ch_april = pd.DataFrame(data=ch_april, columns=['Curah Hujan (mm)'])
30 hh_april = pd.DataFrame(data=hh_april, columns=['Hari Hujan'])
31
32 aprildf = pd.concat([april, rku_maret, ru_maret, ch_maret, hh_maret], axis=1)
33
34 #mei
35 rku_mei = pd.DataFrame(data=rku_mei, columns=['Rata-Rata Kelembapan Udara (%)'])
36 ru_mei = pd.DataFrame(data=ru_mei, columns=['Rata-Rata Udara (.C)'])
37 ch_mei = pd.DataFrame(data=ch_mei, columns=['Curah Hujan (mm)'])
38 hh_mei = pd.DataFrame(data=hh_mei, columns=['Hari Hujan'])
39
40 meidf = pd.concat([mei, rku_mei, ru_mei, ch_mei, hh_mei], axis=1)
41
42 #juni
43 rku_juni = pd.DataFrame(data=rku_juni, columns=['Rata-Rata Kelembapan Udara (%)'])
44 ru_juni = pd.DataFrame(data=ru_juni, columns=['Rata-Rata Udara (.C)'])
45 ch_juni = pd.DataFrame(data=ch_juni, columns=['Curah Hujan (mm)'])
46 hh_juni = pd.DataFrame(data=hh_juni, columns=['Hari Hujan'])
47
48 junidf = pd.concat([juni, rku_juni, ru_juni, ch_juni, hh_juni], axis=1)
49
```

```
50 #oktober
51 rku_oktober = pd.DataFrame(data=rku_oktober, columns=['Rata-Rata Kelembapan Udara (%)'])
52 ru_oktober = pd.DataFrame(data=ru_oktober, columns=['Rata-Rata Udara (.C)'])
53 ch_oktober = pd.DataFrame(data=ch_oktober, columns=['Curah Hujan (mm)'])
54 hh_oktober = pd.DataFrame(data=hh_oktober, columns=['Hari Hujan'])
55
56 oktoberdf = pd.concat([oktober, rku_oktober, ru_oktober, ch_oktober, hh_oktober], axis=1)
57
58 #november
59 rku_november = pd.DataFrame(data=rku_november, columns=['Rata-Rata Kelembapan Udara (%)'])
60 ru_november = pd.DataFrame(data=ru_november, columns=['Rata-Rata Udara (.C)'])
61 ch_november = pd.DataFrame(data=ch_november, columns=['Curah Hujan (mm)'])
62 hh_november = pd.DataFrame(data=hh_november, columns=['Hari Hujan'])
63
64 novemberberdf = pd.concat([november, rku_november, ru_november, ch_november, hh_november], axis=1)
65
66 #desember
67 rku_desember = pd.DataFrame(data=rku_desember, columns=['Rata-Rata Kelembapan Udara (%)'])
68 ru_desember = pd.DataFrame(data=ru_desember, columns=['Rata-Rata Udara (.C)'])
69 ch_desember = pd.DataFrame(data=ch_desember, columns=['Curah Hujan (mm)'])
70 hh_desember = pd.DataFrame(data=hh_desember, columns=['Hari Hujan'])
71
72 desemberberdf = pd.concat([desember, rku_desember, ru_desember, ch_desember, hh_desember], axis=1)
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Menggabungkan Dataframe Sebelumnya Menjadi Satu

```
1 # List DataFrame yang sudah Anda baca
2 dataframes = [januaridf, februaridf, maretddf, aprildf, meidf, junidf, oktoberdf, novemberdf, desemberdf]
3
4 # Menggabungkan DataFrame menjadi satu DataFrame
5 df = pd.concat(dataframes)
6
7 # Reset indeks DataFrame yang telah digabungkan
8 df.reset_index(drop=True, inplace=True)
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Dataframe Yang Akan Digunakan

kota_administrasi	kecamatan	kelurahan	rw	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	ketinggian_air	tanggal_kejadian	...	jumlah_hilang	jumlah_luka_berat	jumlah_punggsi_tertinggi	jumlah_tempat_punggsian	nilai_kerugian
0 Jakarta Barat	PENJARINGAN	KAMAL MUARA	01, 04	2	12	0	0	20 s/d 30 cm	tgl. 02	...	0	0	0	0	0
1 Jakarta Utara	KALIDERES	KAMAL	1, 4	2	3	0	0	10 s/d 25 cm	tgl. 03, 04, 05, 18, 31	...	0	0	0	0	0
2 Jakarta Utara	GROGOL PETAMBURAN	JELAMBAR BARU	1	1	2	0	0	15 cm	tgl. 03	...	0	0	0	0	0
3 Jakarta Utara	GROGOL PETAMBURAN	JELAMBAR	7	1	1	0	0	20 cm	tgl. 03	...	0	0	0	0	0
4 Jakarta Utara	KEMBANGAN	KEMBANGAN UTARA	3, 4, 5, 6	4	2	0	0	5 s/d 20 cm	tgl. 18	...	0	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
130 Jakarta Selatan	KEC. PESANGGRAHAN	KEL. ULUJAMI	1	1	8	0	0	20 s/d 40 cm	tgl. 11	...	0	0	0	0	0
131 Jakarta Timur	KEC. CAKUNG	KEL. RAWA TERATE	5	1	1	0	0	10 s/d 100 cm	tgl. 3	...	0	0	0	0	0
132 Jakarta Timur	KEC. CAKUNG	CAKUNG TIMUR	9	1	4	210	800	20 cm	tgl. 3	...	0	0	0	0	0
133 Jakarta Timur	KEC. MAKASAR	KEL. CIPINANG MELAYU	4	1	1	0	0	10 s/d 30 cm	tgl. 6	...	0	0	0	0	0
134 Jakarta Timur	KEC. CIRACAS	KEL. RAMBUTAN	03, 05	2	4	0	0	10 s/d 50	tgl. 11	...	0	0	0	0	0

135 rows × 22 columns

### Statistik Dataframe

	jumlah_terdampak_rw	jumlah_terdampak_rt	jumlah_terdampak_kk	jumlah_terdampak_jiwa	lama_genangan	jumlah_meninggal	jumlah_hilang	jumlah_luka_berat	jumlah_luka_ringan	jumlah_punggsi_tertinggi	jumlah_tempat_punggsian	nilai_kerugian
count	135.000000	135.000000	135.000000	135.000000	135.000000	135.0	135.000000	135.0	135.000000	135.000000	135.0	135.0
mean	2.748148	7.874074	67.340741	242.244444	0.007407	0.0	0.007407	0.0	115.755556	0.459259	0.0	0.0
std	2.457758	14.925401	190.169805	699.523189	0.086068	0.0	0.086068	0.0	450.404735	1.605787	0.0	0.0
min	1.000000	0.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.0
25%	1.000000	1.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.0
50%	2.000000	3.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.0
75%	4.000000	8.000000	0.000000	0.000000	0.000000	0.0	0.000000	0.0	0.000000	0.000000	0.0	0.0
max	16.000000	143.000000	1066.000000	4000.000000	1.000000	0.0	1.000000	0.0	3200.000000	10.000000	0.0	0.0

### Informasi Dataframe

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 135 entries, 0 to 134
Data columns (total 22 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   kota_administrasi 135 non-null   object  
 1   kecamatan         135 non-null   object  
 2   kelurahan         135 non-null   object  
 3   rw                135 non-null   object  
 4   jumlah_terdampak_rw 135 non-null   int64  
 5   jumlah_terdampak_rt 135 non-null   int64  
 6   jumlah_terdampak_kk 135 non-null   int64  
 7   jumlah_terdampak_jiwa 135 non-null   int64  
 8   ketinggian_air     135 non-null   object  
 9   tanggal_kejadian   135 non-null   object  
 10  lama_genangan     135 non-null   int64  
 11  jumlah_meninggal  135 non-null   int64  
 12  jumlah_hilang     135 non-null   int64  
 13  jumlah_luka_berat 135 non-null   int64  
 14  jumlah_luka_ringan 135 non-null   int64  
 15  jumlah_punggsi_tertinggi 135 non-null   int64  
 16  jumlah_tempat_punggsian 135 non-null   int64  
 17  nilai_kerugian    135 non-null   int64  
 18  Rata-Rata Kelembapan Udara (%) 135 non-null   object  
 19  Rata-Rata Udara (-C)   135 non-null   object  
 20  Curah Hujan (mm)    135 non-null   object  
 21  Hari Hujan        135 non-null   object  
dtypes: int64(12), object(10)
memory usage: 23.3+ KB
```

# Data Assesment & Preprocessing

## Rumusan Masalah 1

### Cek missing value

```
1 df.isna().sum()  
  
kota_administrasi          0  
kecamatan                  0  
kelurahan                  0  
rw                          0  
jumlah_terdampak_rw        0  
jumlah_terdampak_rt        0  
jumlah_terdampak_kk        0  
jumlah_terdampak_jiwa       0  
ketinggian_air              0  
tanggal_kejadian            0  
lama_genangan              0  
jumlah_meninggal             0  
jumlah_hilang                0  
jumlah_luka_berat            0  
jumlah_luka_ringan           0  
jumlah_pengungsi_tertinggi      0  
jumlah_tempat_pengungsian      0  
nilai_kerugian               0  
Rata-Rata Kelembapan Udara (%) 0  
Rata-Rata Udara (.C)          0  
Curah Hujan (mm)              0  
Hari Hujan                   0  
dtype: int64
```

### Cek data duplikat

```
1 df.duplicated().sum()
```

0

### Cek jumlah data

```
1 df.sum()  
  
kota_administrasi          Jakarta Barat  
kecamatan                  Jakarta Utara  
kelurahan                  Jakarta Utara  
rw                          PENJARINGAN  
jumlah_terdampak_rw        KALIDERES  
jumlah_terdampak_rt        GROGOL PETAMBURANG  
jumlah_terdampak_kk        KAMAL MUARA  
jumlah_terdampak_jiwa       KAMAL JELAMBAR  
ketinggian_air              BARU JELAMBARKEMBANGAN  
tanggal_kejadian             01, 041, 4173, 4, 5, 602, 05, 06, 07, 09, 12, ...  
lama_genangan              371  
jumlah_meninggal             1063  
jumlah_hilang                9091  
jumlah_luka_berat             32703  
jumlah_luka_ringan           20 s/d 30 cm  
jumlah_pengungsi_tertinggi      10 cm  
jumlah_tempat_pengungsian      15 cm  
nilai_kerugian                 5 s/d 15 cm  
Rata-Rata Kelembapan Udara (%) 20 cm  
Rata-Rata Udara (.C)          15 cm  
Curah Hujan (mm)              10 cm  
Hari Hujan                   6 cm  
dtype: object
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Menghapus kolom yang 100% kosong / tidak memiliki nilai / tidak begitu penting

```
1 # menghapus kolom yang 100% kosong / tidak memiliki nilai / tidak begitu penting
2 df = df.drop(columns=['jumlah_meninggal', 'lama_genangan', 'jumlah_hilang', 'jumlah_luka_berat', 'jumlah_tempat_pengungsian', 'nilai_kerugian', 'tanggal_kejadian'])
```

### Mengubah kolom ketinggian air menjadi integer menggunakan rata-rata ketinggian air

```
1 import re
2
3 # Membuat fungsi untuk mendapatkan rata-rata dari string ketinggian air
4 def get_average_height(height_str):
5     # Menggunakan regular expression untuk mengekstraksi angka dari string
6     heights = re.findall(r'\d+', height_str)
7     # Mengubah angka menjadi tipe integer
8     heights = [int(height) for height in heights]
9     # Menghitung rata-rata ketinggian air
10    average_height = sum(heights) / len(heights)
11    return average_height
12
13 # Menggunakan fungsi get_average_height untuk mengubah fitur ketinggian air menjadi numerik
14 df['ketinggian_air'] = df['ketinggian_air'].apply(get_average_height)
```

# Data Asesment & Preprocessing

## Rumusan Masalah 3

### Mengubah tipe data kolom menjadi numerik

```
1 # Mengubah tipe data kolom menjadi numerik
2 df['ketinggian_air'] = df['ketinggian_air'].astype('int64')
3
4
5 df['Rata-Rata Kelembapan Udara (%)'] = df['Rata-Rata Kelembapan Udara (%)'].str.replace(',', '.').astype(float)
6 df['Rata-Rata Udara (.C)'] = df['Rata-Rata Udara (.C)'].str.replace(',', '.').astype(float)
7 df['Curah Hujan (mm)'] = df['Curah Hujan (mm)'].str.replace(',', '.').astype(float)
8 df['Hari Hujan'] = df['Hari Hujan'].str.replace(',', '.').astype(float)
9
10 df['Rata-Rata Kelembapan Udara (%)'] = df['Rata-Rata Kelembapan Udara (%)'].round().astype(int)
11 df['Rata-Rata Udara (.C)'] = df['Rata-Rata Udara (.C)'].round().astype(int)
12 df['Curah Hujan (mm)'] = df['Curah Hujan (mm)'].round().astype(int)
13 df['Hari Hujan'] = df['Hari Hujan'].round().astype(int)
14
15 # Menampilkan informasi DataFrame setelah perubahan tipe data
16 print(df.info())
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Melihat nilai unik dari kolom kota administrasi

```
1 # Mengetahui data unik dari kota administrasi  
2  
3 unique_data = df['kota_administrasi'].unique()  
4 print(unique_data)  
  
['Jakarta Barat' 'Jakarta Utara' 'Jakarta Pusat' 'Jakarta Selatan'  
'Jakarta Timur']
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Membuat Dataframe Jumlah Penduduk Jakarta 2018 Berdasarkan Kota Administrasi

```
1 # MEMBUAT DATAFRAME JUMLAH PENDUDUK JAKARTA 2018 BERDASARKAN KOTA ADMINISTRASI
2
3 data = {
4     'Kota Administrasi': ['Jakarta Selatan', 'Jakarta Timur', 'Jakarta Pusat', 'Jakarta Barat', 'Jakarta Utara'],
5     'Jumlah Penduduk': [2246137, 2916018, 924686, 2559362, 1797292]
6 }
7
8 df_penduduk = pd.DataFrame(data)
9
10 #SUMBER: https://jakarta.bps.go.id/indicator/12/1270/1/jumlah-penduduk-menurut-kabupaten-kota-di-provinsi-dki-jakarta-.html
```

1 df_penduduk		
	Kota Administrasi	Jumlah Penduduk
0	Jakarta Selatan	2246137
1	Jakarta Timur	2916018
2	Jakarta Pusat	924686
3	Jakarta Barat	2559362
4	Jakarta Utara	1797292

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Buat Kolom Jumlah Penduduk dengan Mapping Sesuai Dengan Kora Administrasi

```
1 mapping = {  
2     'Jakarta Selatan': 2246137,  
3     'Jakarta Timur': 2916018,  
4     'Jakarta Pusat': 924686,  
5     'Jakarta Barat': 2559362,  
6     'Jakarta Utara': 1797292  
7 }  
8  
9 df['jumlah_penduduk'] = df['kota_administrasi'].replace(mapping)
```

# Data Assesment & Preprocessing

## Rumusan Masalah 3

### Mengecek Outlier

```
1 #Mengecek outlier
2 import pandas as pd
3 import numpy as np
4 from scipy import stats
5
6 # Menghitung z-score untuk setiap kolom numerik dalam dataframe
7 z_scores = np.abs(stats.zscore(df.select_dtypes(include=np.number)))
8
9 # Menentukan batas z-score untuk mengidentifikasi outlier
10 threshold = 5
11
12 # Menampilkan data yang dianggap sebagai outlier
13 outliers = df[(z_scores > threshold).any(axis=1)]
14 print(outliers)
```

### Menghapus Outlier

```
1 #Menghapus Outlier
2
3 df = df[(z_scores <= threshold).all(axis=1)]
```

### Output

	kota_administrasi	jumlah_terdampak_rw	jumlah_terdampak_rt	\
16	Jakarta Barat	16	143	
29	Jakarta Selatan	2	6	
41	Jakarta Timur	8	55	
46	Jakarta Timur	6	27	
104	Jakarta Selatan	1	1	
	jumlah_terdampak_kk	jumlah_terdampak_jiwa	ketinggian_air	\
16	0	0	25	
29	600	2993	155	
41	983	3074	115	
46	1066	3584	145	
104	825	4000	55	
	jumlah_luka_ringan	jumlah_pengungsi_tertinggi	\	
16	0	0		
29	3200	7		
41	1392	10		
46	557	4		
104	0	0		

	Rata-Rata Kelembapan Udara (%)	Rata-Rata Udara (°C)	Curah Hujan (mm)	\
16	82	28	484	
29	82	28	484	
41	82	28	484	
46	82	28	484	
104	76	29	192	
	Hari Hujan	jumlah_penduduk		
16	24	2559362		
29	24	2246137		
41	24	2916018		
46	24	2916018		
104	13	2246137		

# List of Content

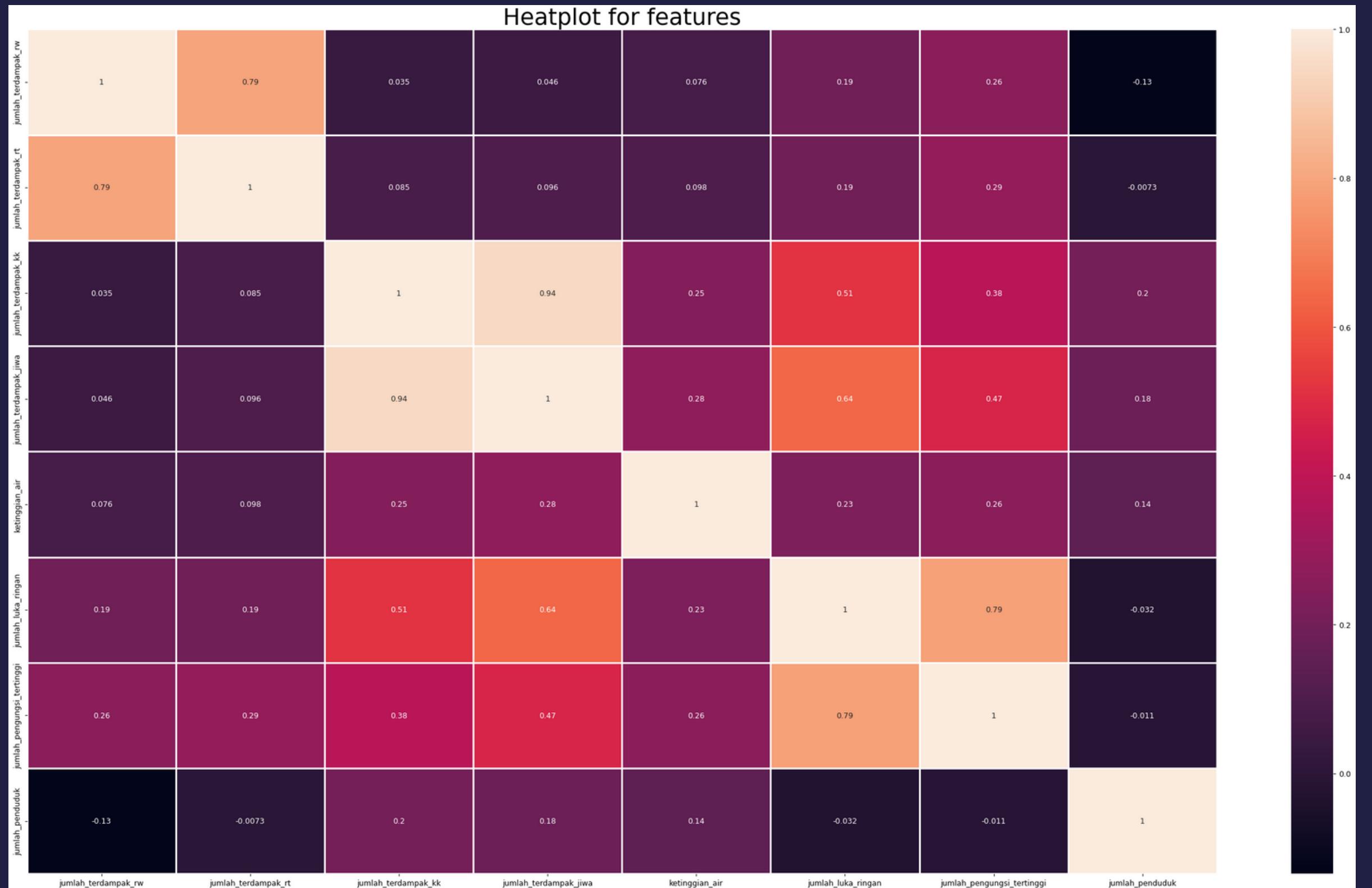
04

Exploratory Data Analysis (EDA)

Penjelasan mengenai EDA dari data

# EDA Rumusan Masalah 1 & 2

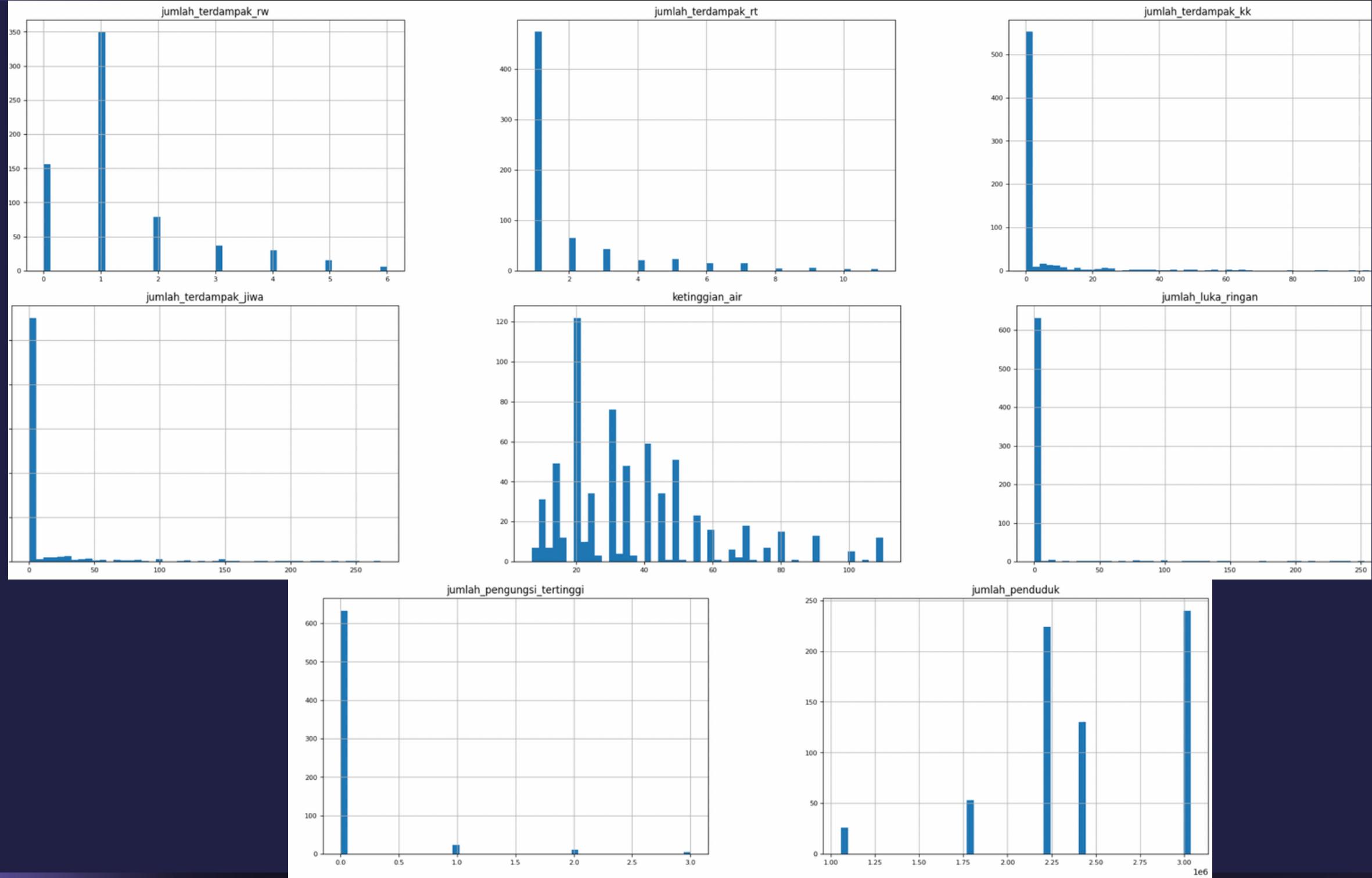
## Heat Map



Disini kami melakukan visualisasi data berupa heatmap untuk melihat hubungan antar masing-masing fitur dan mana sekiranya hubungan masing-masing fitur yang memiliki korelasi berbanding lurus ataupun korelasi yang berbanding terbalik. semakin nilai heatmap mendekati 1, maka semakin berbanding lurus, begitu juga sebaliknya. Dari sini dapat dilihat salah satu yang memiliki korelasi tinggi adalah jumlah pengungsi tertinggi dan jumlah luka ringan, sedangkan ketinggian air tidak ada memiliki hubungan yang terlalu kuat dengan yang lain.

# EDA Rumusan Masalah 1 & 2

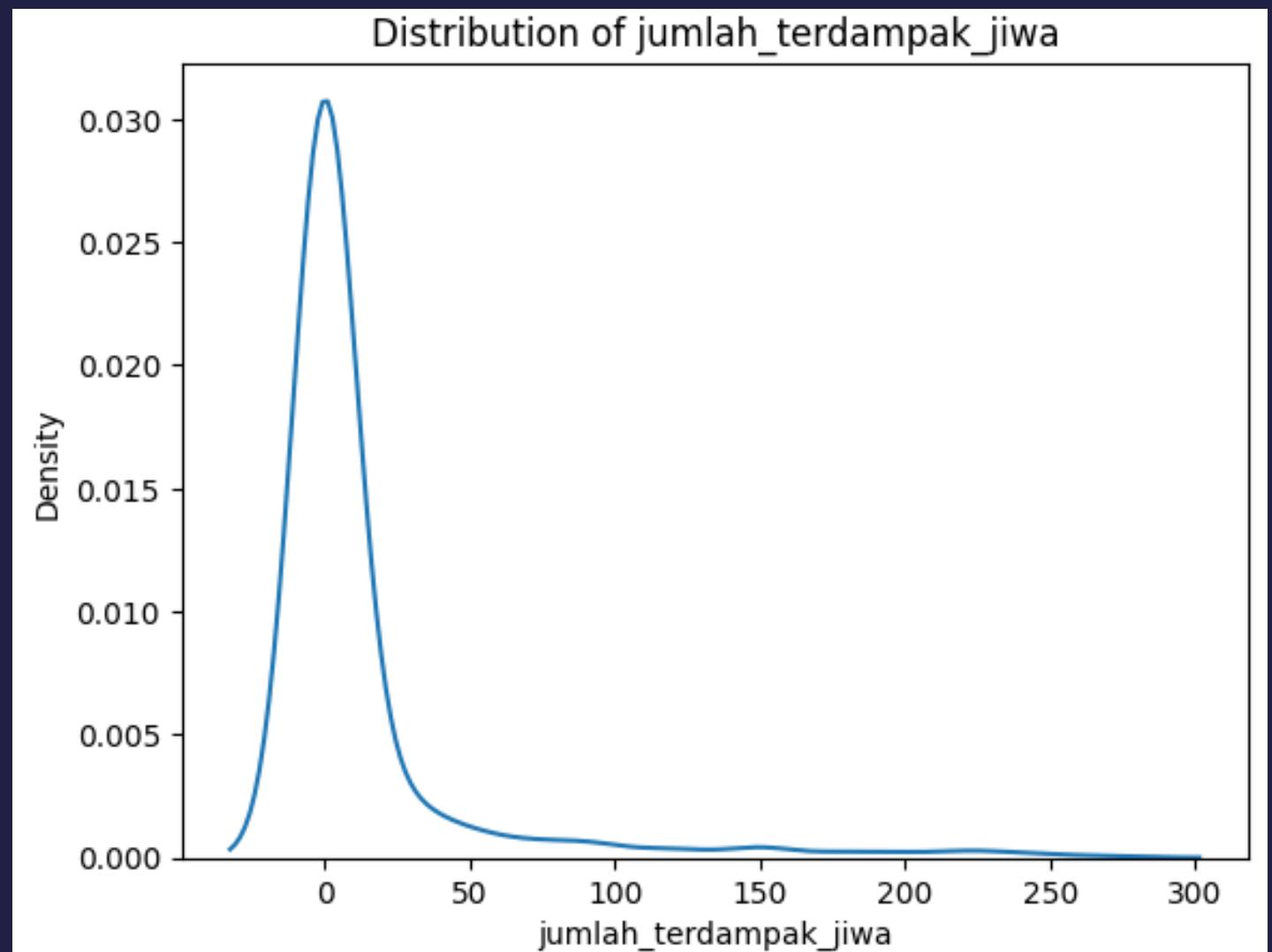
## Distribusi Dataset dalam Barplot



Disini kami melakukan visualisasi data berupa barplot untuk melihat masing-masing distribusi dataset dari masing-masing fitur, seperti ketinggian air, jumlah terdampak jiwa, dan lain-lain

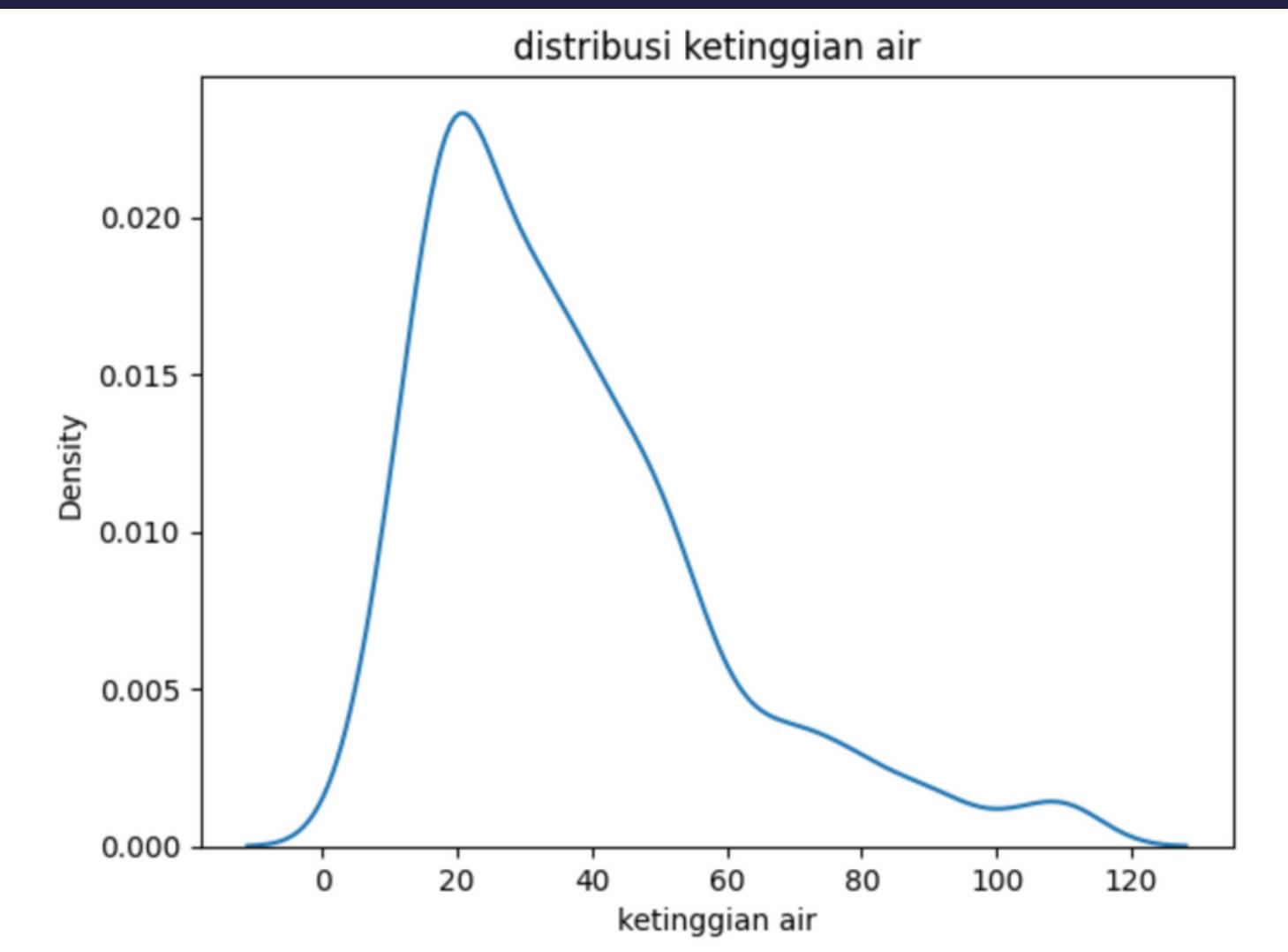
# EDA Rumusan Masalah 1 & 2

## Distribusi jumlah terdampak jiwa



Distribusi jumlah terdampak jiwa dominan di angka 0 yang berarti hampir tidak ada yang terdampak jiwa pada banjir 2020

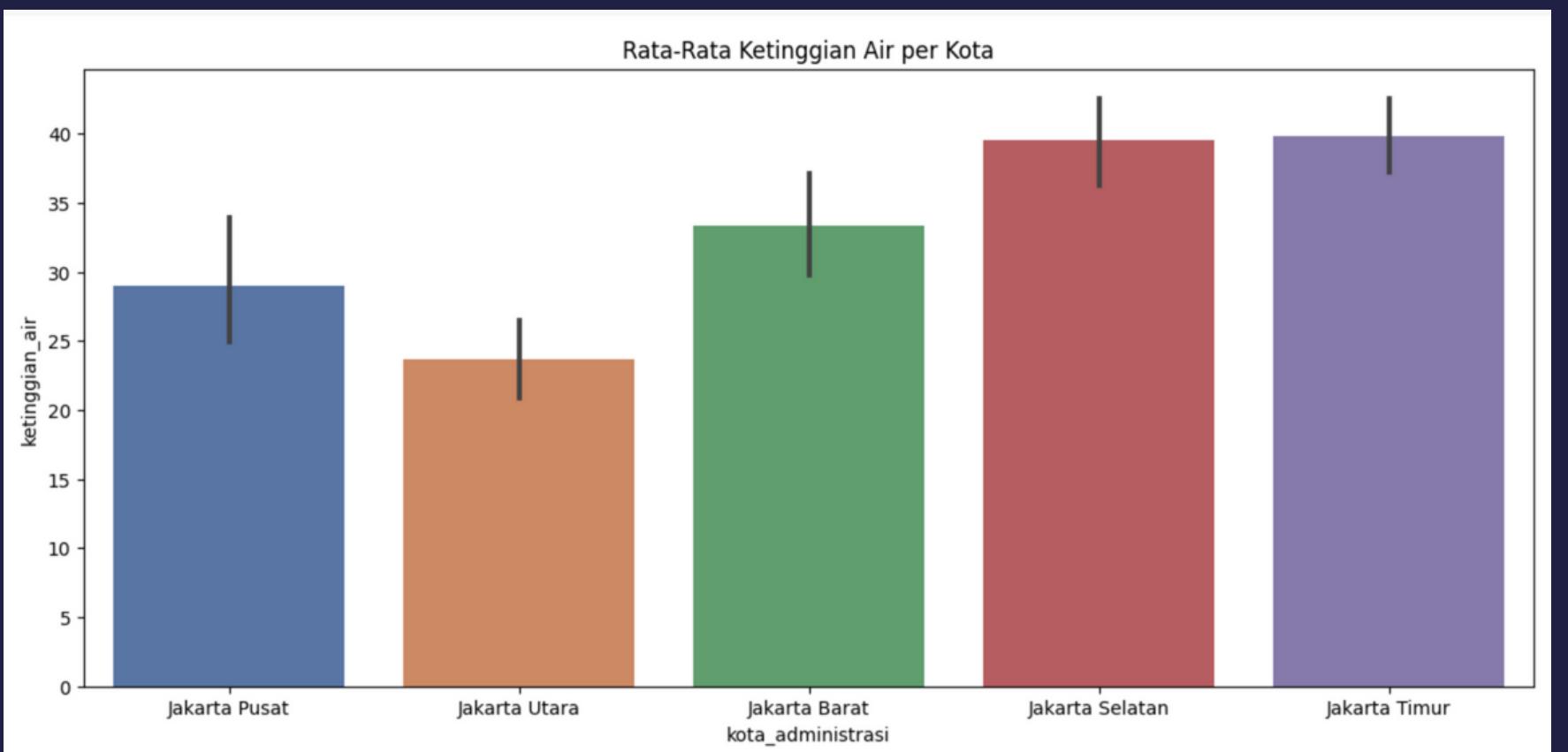
## Distribusi Ketinggian Air



Distribusi ketinggian air dominan berada pada saat ketinggian air di sekitar 20-30cm.

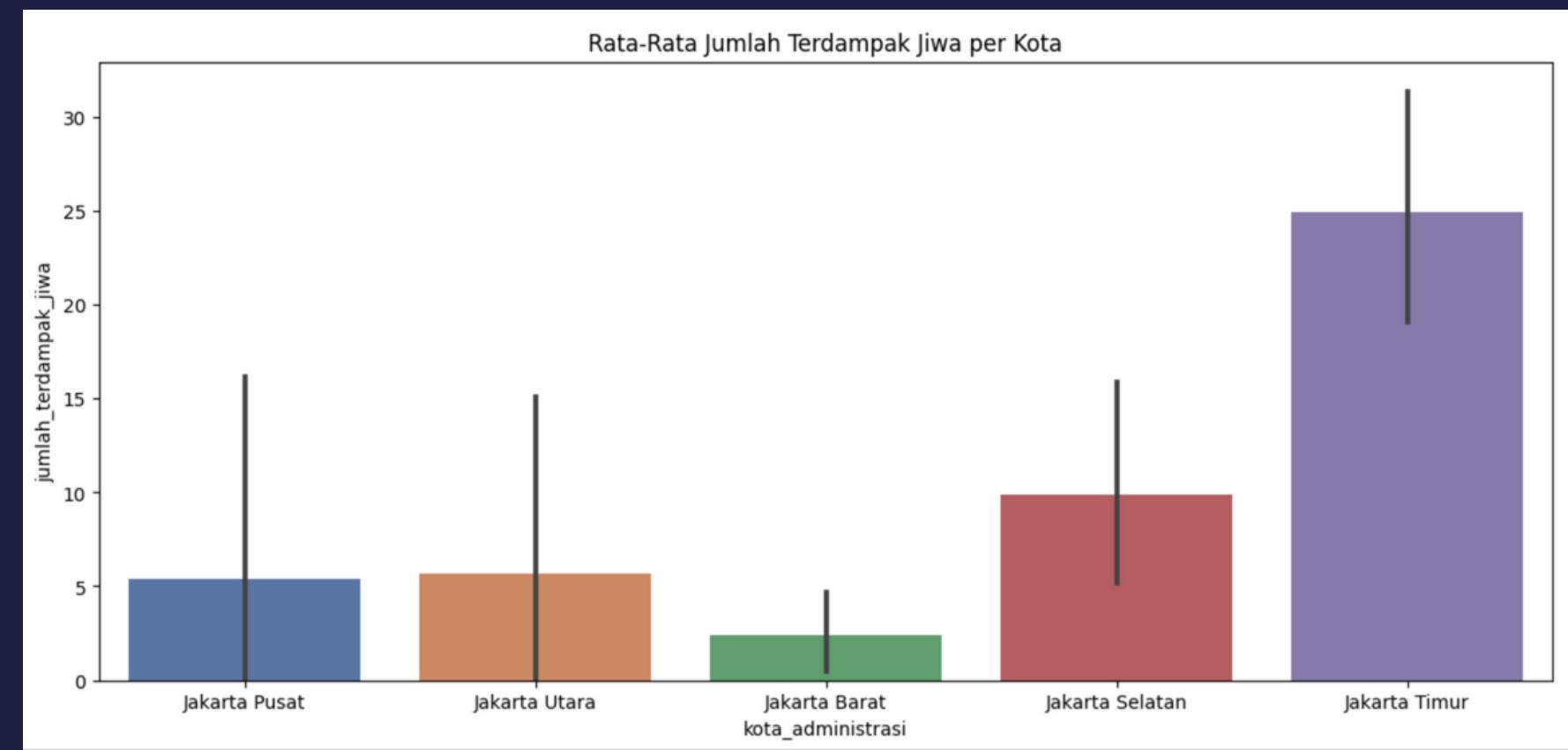
## EDA Rumusan Masalah 3

### Bar Plot Ketinggian Air



Rata-Rata Ketinggian Air per Kota Administrasi tertinggi berada pada jakarta selatan dan jakarta timur, sedangkan terendah berada pada jakarta utara.

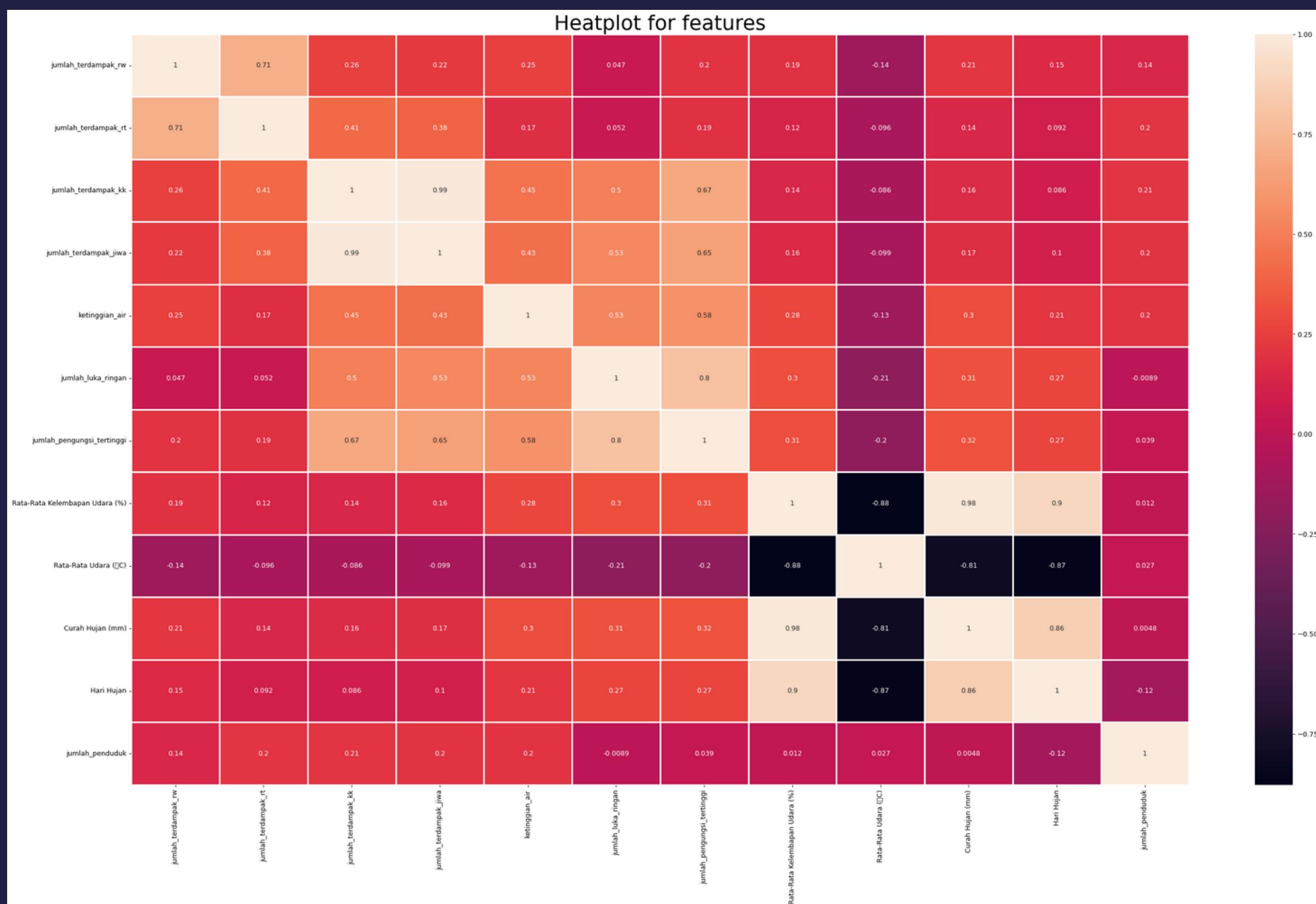
### Bar Plot Jumlah Terdampak Jiwa



Jakarta Timur menjadi jumlah yang paling banyak terdampak jiwa sedangkan jakarta barat menjadi yang paling sedikit jumlah yang paling banyak terdampak jiwa.

# EDA Rumusan Masalah 3

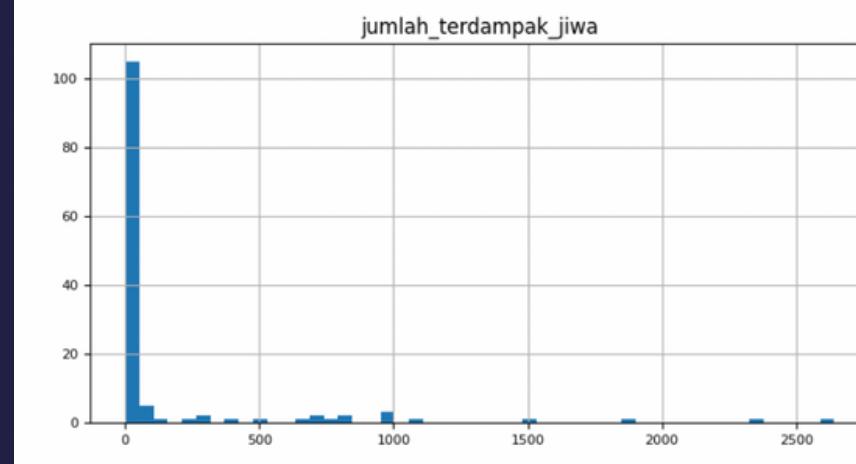
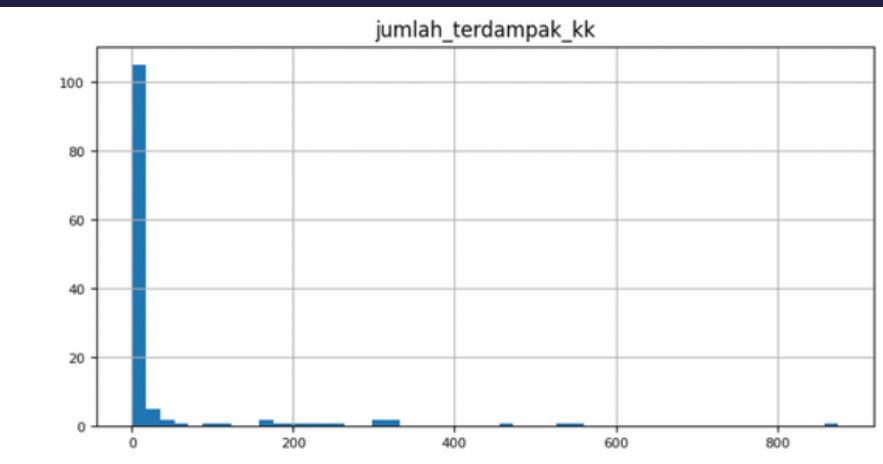
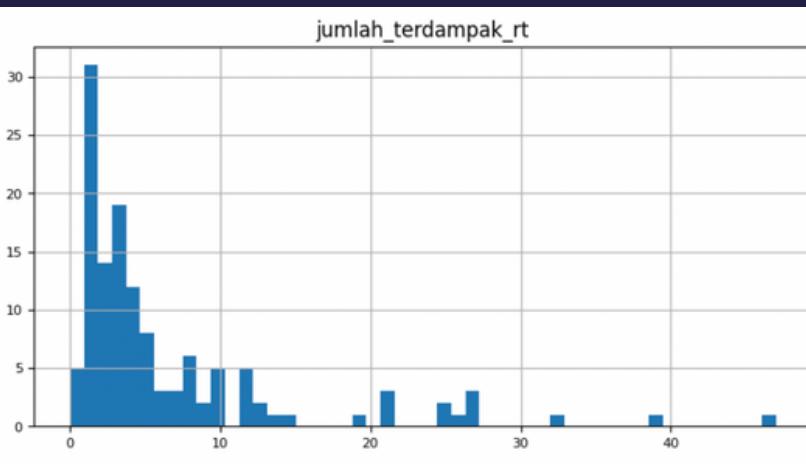
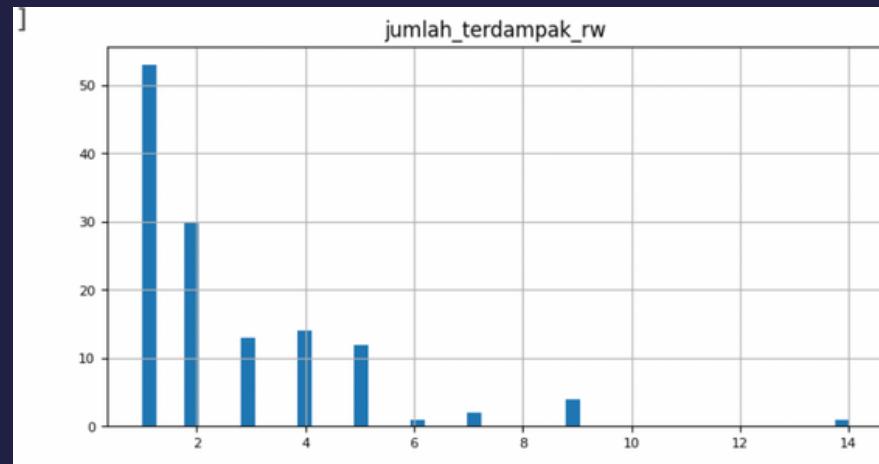
## Heat Map



Disini kami melakukan visualisasi data berupa heatmap untuk melihat hubungan antar masing-masing fitur dan mana sekiranya hubungan masing-masing fitur yang memiliki korelasi berbanding lurus ataupun korelasi yang berbanding terbalik. semakin nilai heatmap mendekati 1, maka semakin berbanding lurus, begitu juga sebaliknya. Dari sini dapat dilihat salah satu yang memiliki korelasi tinggi adalah jumlah pengungsi tertinggi dan jumlah luka ringan, dan juga pengungsi tertinggi dengan jumlah\_terdampak\_jawa

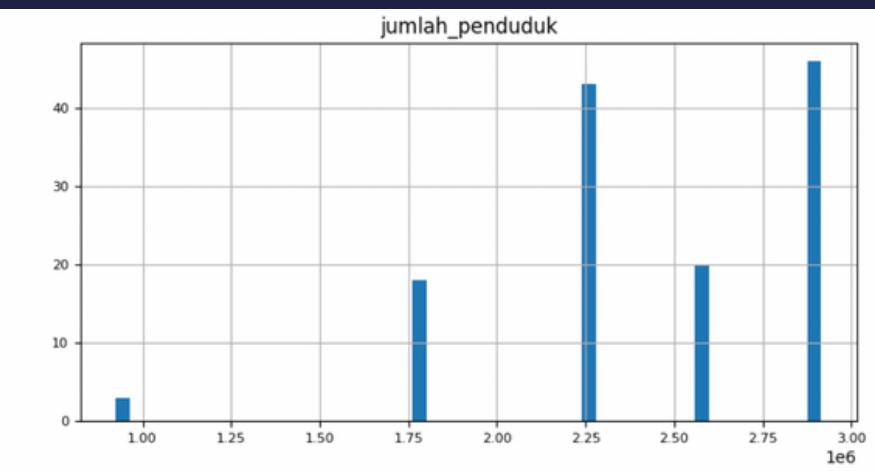
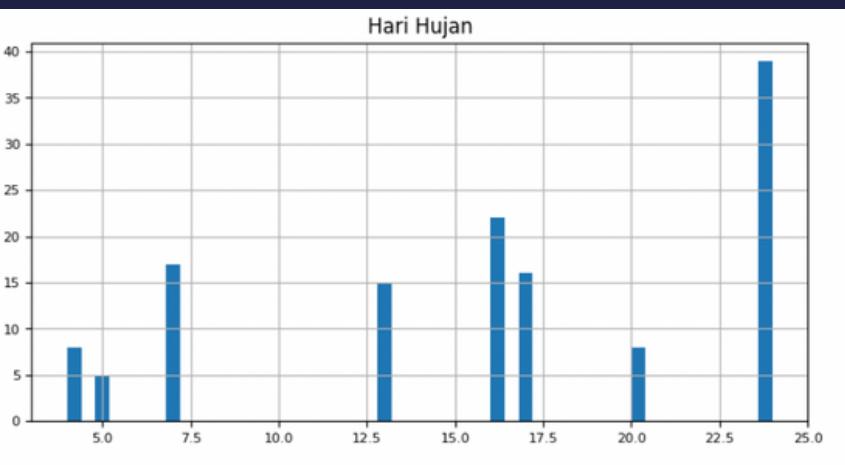
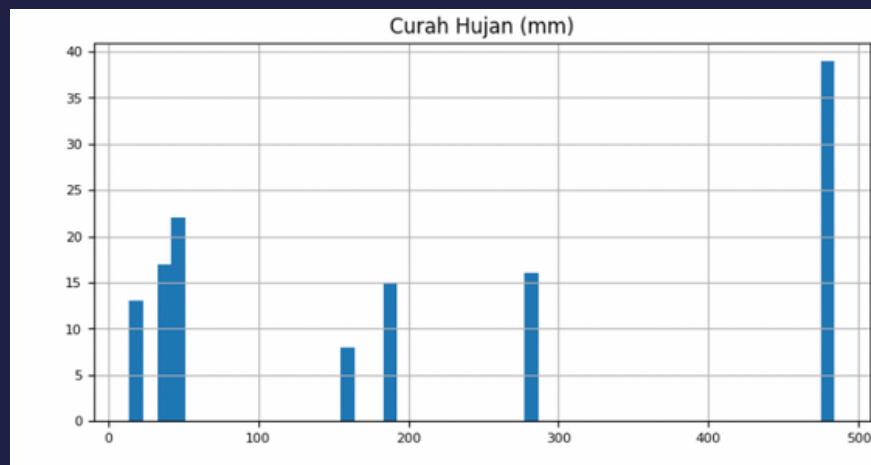
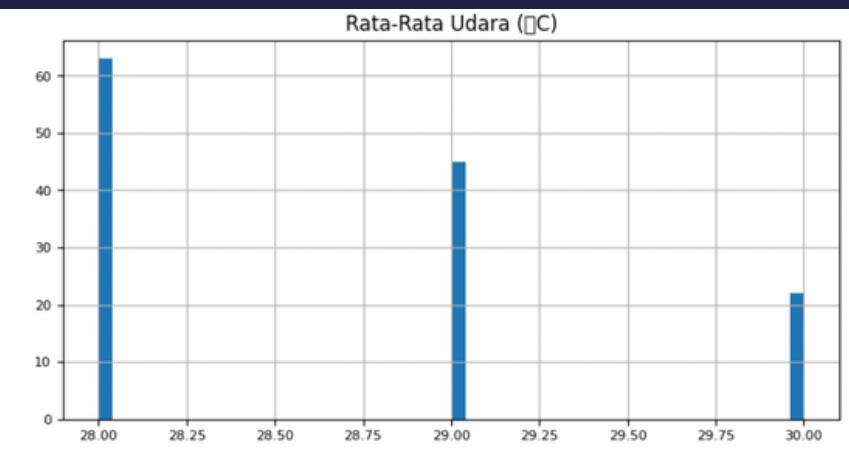
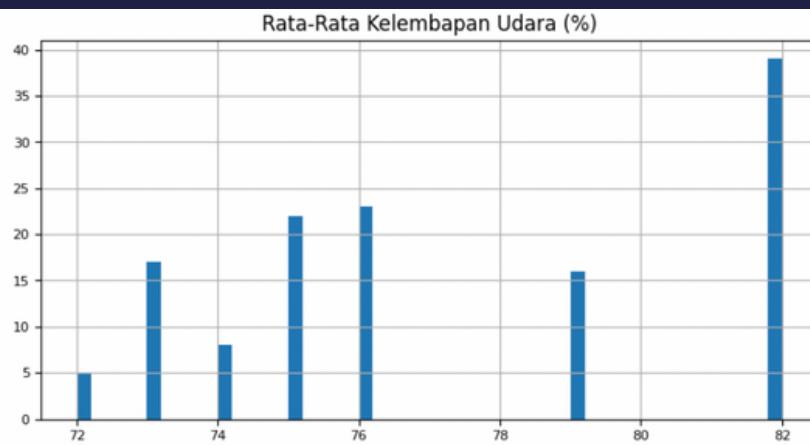
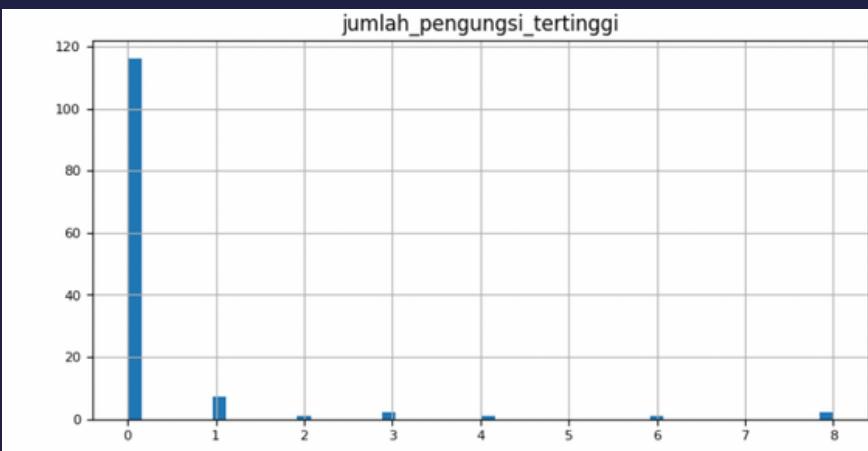
# EDA Rumusan Masalah 3

## Distribusi Dataset dalam Barplot



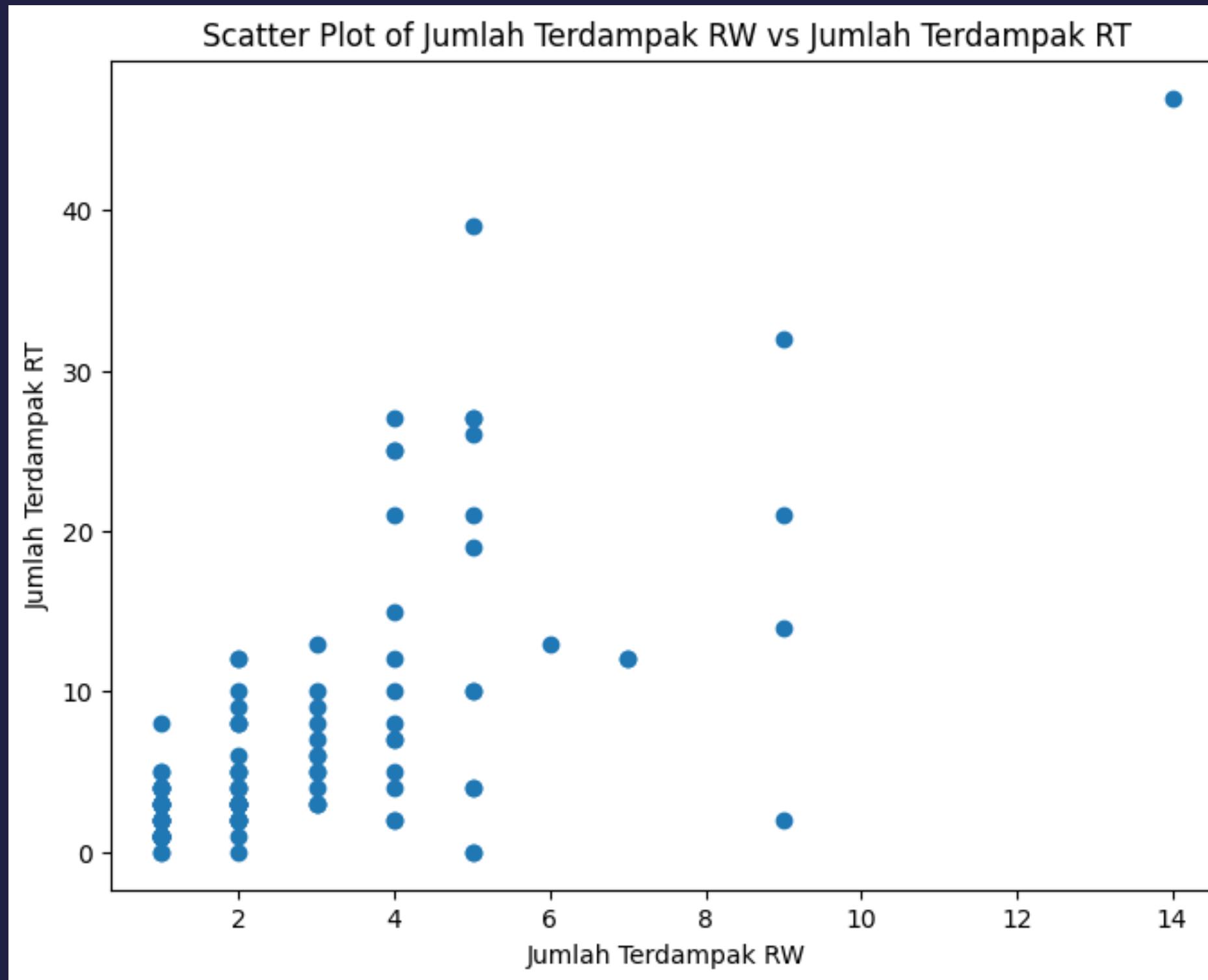
# EDA Rumusan Masalah 3

## Distribusi Dataset dalam Barplot



## EDA Rumusan Masalah 3

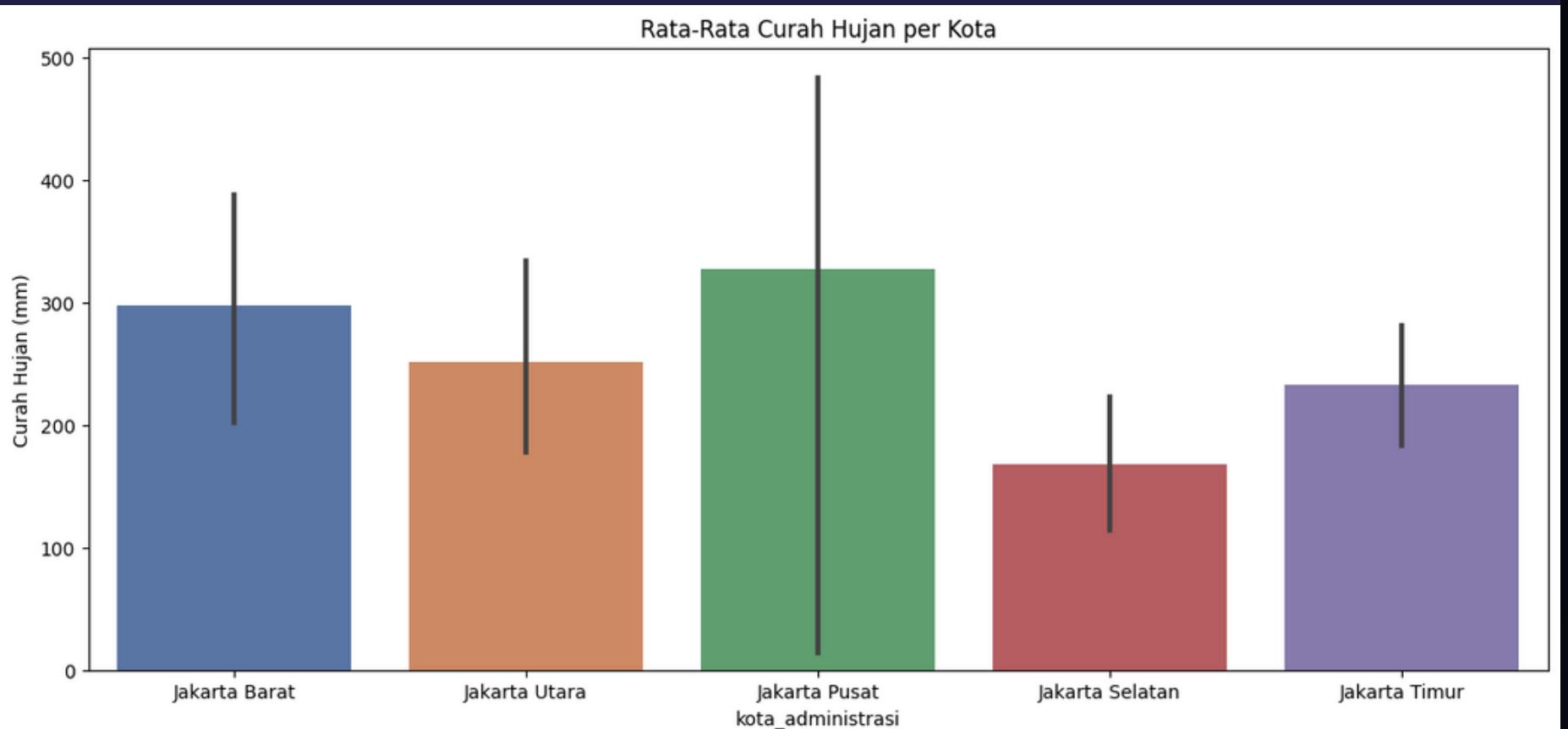
### Scatter Plot



Korelasi yang cukup berbanding lurus antara jumlah terdampak rt dan jumlah terdampak rw.

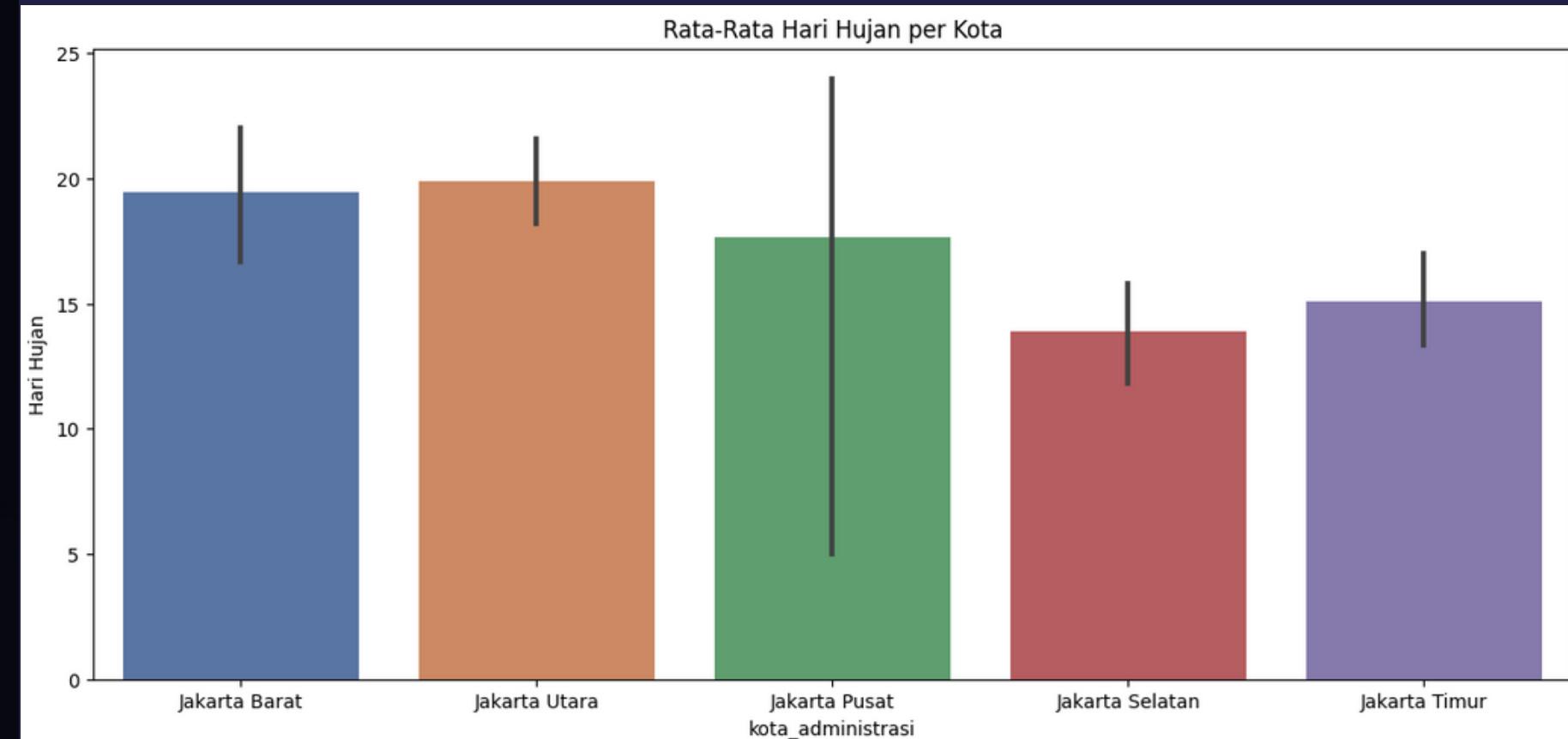
# EDA Rumusan Masalah 3

## Bar Plot Curah Hujan



Curah Hujan tertinggi dialami oleh jakarta pusat, sedangkan curah hujan terendah dialami oleh jakarta selatan

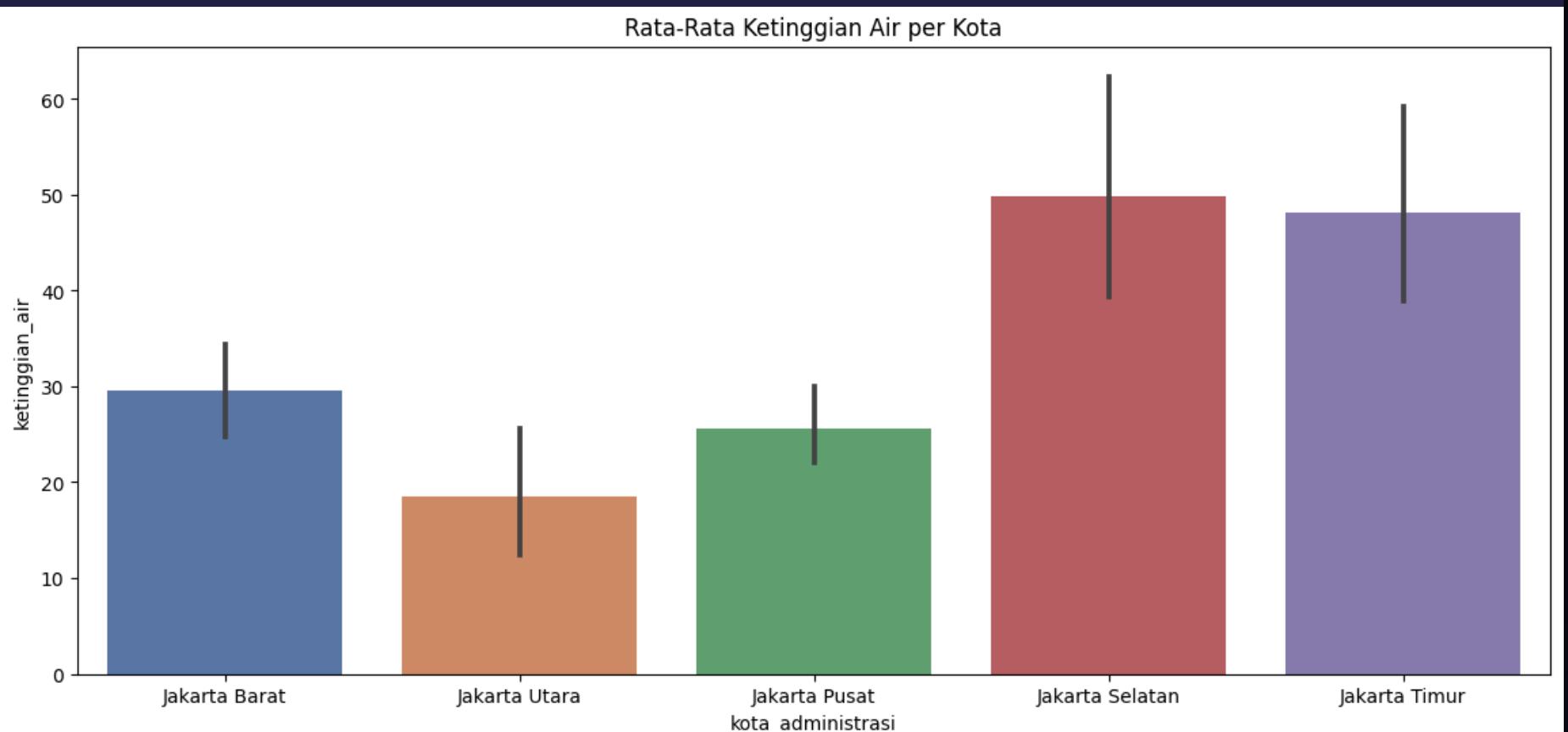
## Bar Plot Hari Hujan



dapat dilihat kota yang paling sering terkena hujan setiap bulannya adalah jakarta utara dan jakarta barat, disusul jakarta pusat, sedangkan jakarta timur dan jakarta selatan paling sedikit diantara yang lain

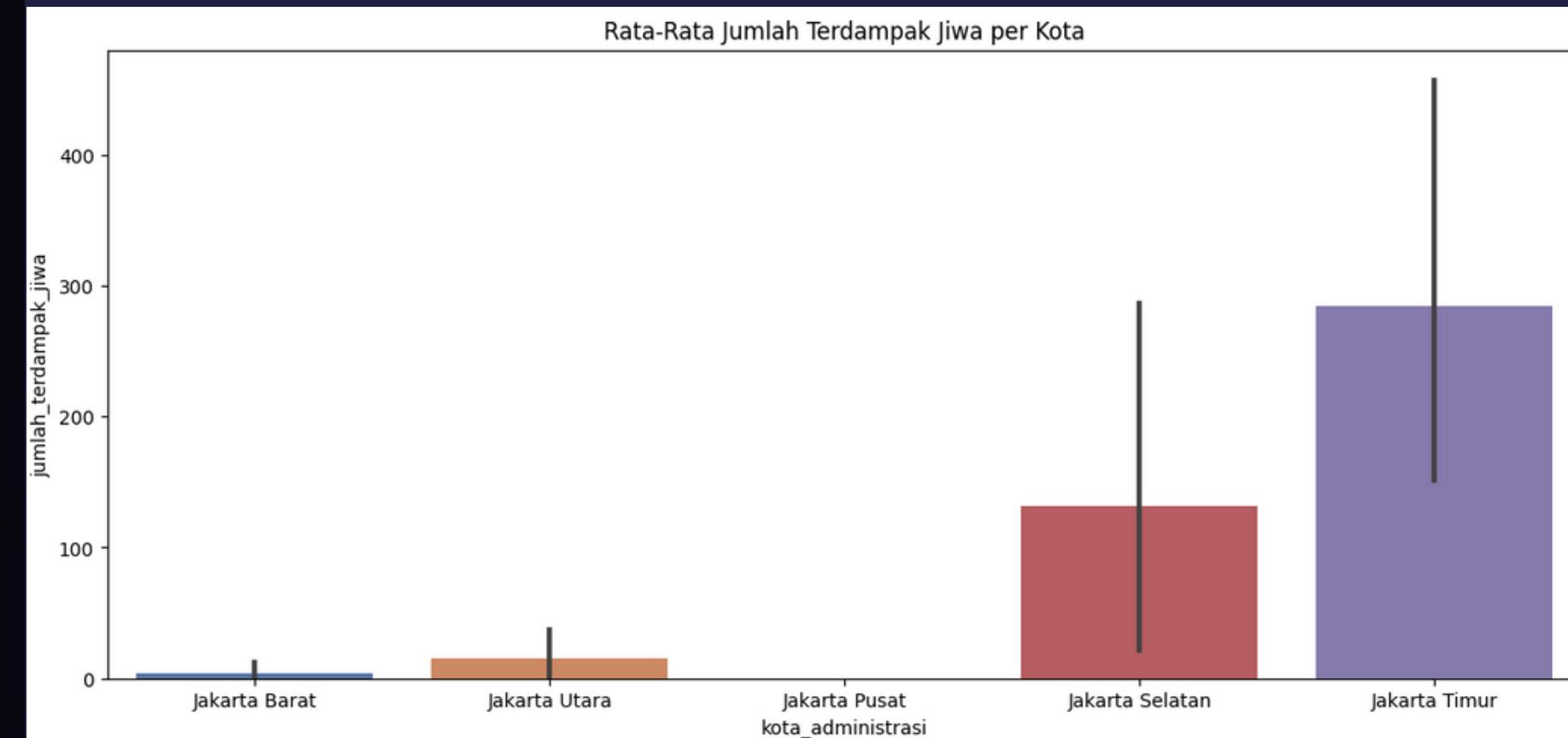
# EDA Rumusan Masalah 3

## Bar Plot Ketinggian Air



Rata-Rata Ketinggian Air per Kota Administrasi tertinggi berada pada jakarta selatan dan jakarta timur, sedangkan terendah berada pada jakarta utara.

## Bar Plot Jumlah Terdampak Jiwa



Jakarta Timur menjadi jumlah yang paling banyak terdampak jiwa sedangkan jakarta barat dan menjadi yang paling sedikit jumlah yang paling banyak terdampak jiwa.

# List of Content

05

**Feature  
Engineering &  
Modelling**

Melakukan modeling  
terhadap data dan feature  
engineering

# Model Rumusan Masalah 1

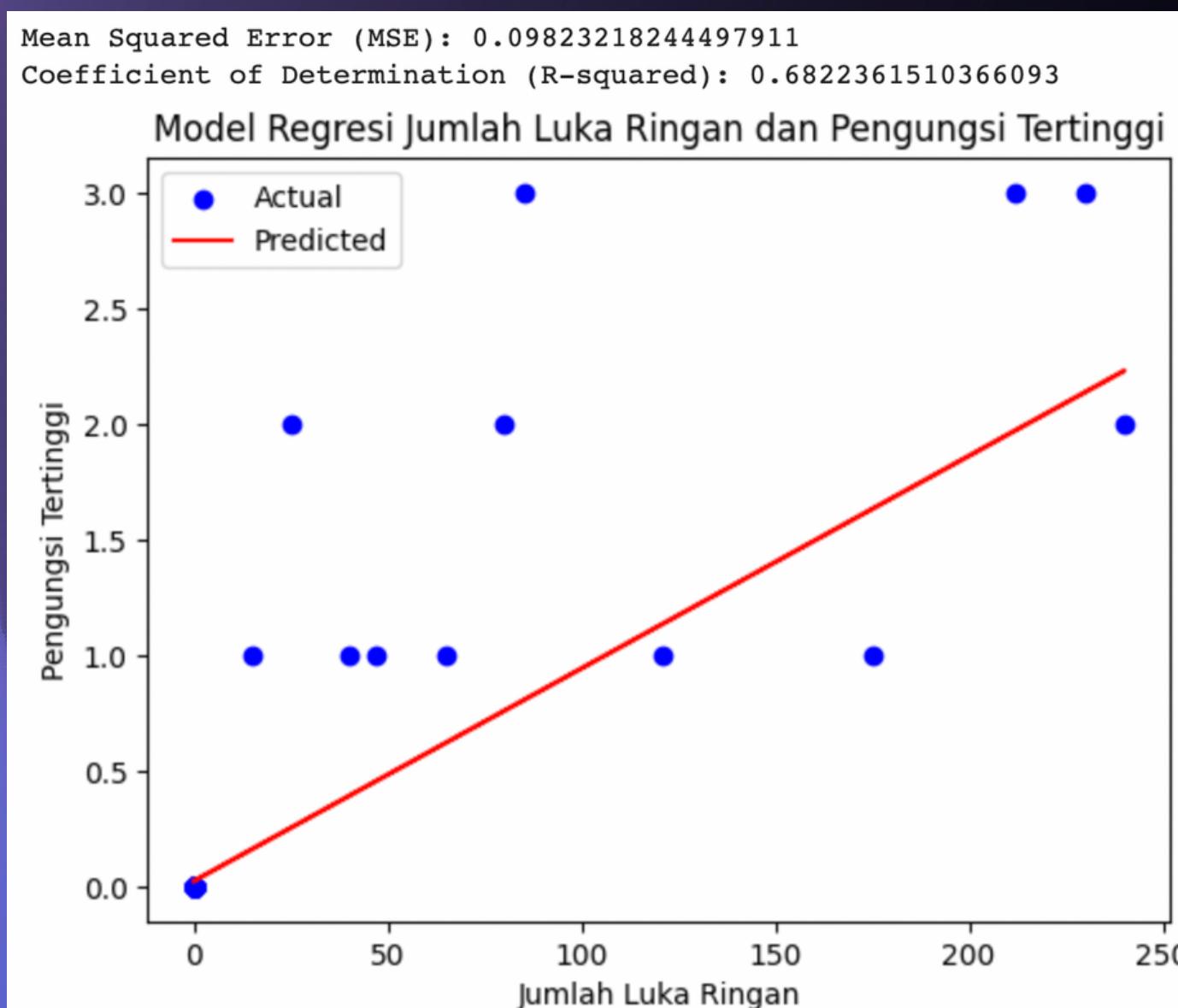
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import mean_squared_error, r2_score

#Pisahkan data ke dalam fitur (X) dan target (y)
X = df['jumlah_luka_ringan'].values.reshape(-1, 1)
y = df['jumlah_pengungsi_tertinggi'].values
#Bagi data menjadi set pelatihan dan set pengujian
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
#Lakukan scaling pada fitur menggunakan StandardScaler
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
#model regresi linear
model = LinearRegression()
#Latih model menggunakan data pelatihan yang telah di-scaled
model.fit(X_train_scaled, y_train)
#Prediksi ketinggian air menggunakan data pengujian yang telah di-scaled
y_pred = model.predict(X_test_scaled)
#Evaluasi model dengan menghitung metrik seperti mean squared error (MSE) dan coefficient of determination (R-squared):
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print("Mean Squared Error (MSE):", mse)
print("Coefficient of Determination (R-squared):", r2)
```

## Model Rumusan Masalah 1

```
#visualisasi hasil prediksi dan data aktual  
plt.scatter(X_test, y_test, color='blue', label='Actual')  
plt.plot(X_test, y_pred, color='red', label='Predicted')  
plt.xlabel('Jumlah Luka Ringan')  
plt.ylabel('Pengungsi Tertinggi')  
plt.title('Model Regresi Jumlah Luka Ringan dan Pengungsi Tertinggi')  
plt.legend()  
plt.show()
```

### OUTPUT



Semakin kecil nilai MSE, semakin baik performa model dalam memprediksi jumlah terdampak jiwa. Dalam kasus ini, nilai MSE yang relatif kecil menunjukkan bahwa model memiliki tingkat kesalahan yang rendah.

Dalam kasus ini, nilai R-squared sebesar 0.6822 menunjukkan bahwa model mampu menjelaskan sekitar 68.22% variasi jumlah terdampak jiwa berdasarkan fitur-fitur yang digunakan dalam model.

meskipun model regresi linear yang di bangun memiliki tingkat kesalahan yang rendah (MSE yang kecil), namun kemampuan model dalam menjelaskan variasi jumlah terdampak jiwa belum sepenuhnya optimal

## Model Rumusan Masalah 2

```
[ ] import pandas as pd
from sklearn.cluster import KMeans
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# Pilih fitur-fitur yang akan digunakan
features = ['jumlah_terdampak_rw', 'jumlah_terdampak_rt', 'jumlah_terdampak_kk',
            'jumlah_terdampak_jiwa', 'ketinggian_air', 'jumlah_luka_ringan', 'jumlah_pengungsi_tertinggi', 'jumlah_penduduk']

# Preprocessing Data
# Lakukan clustering dengan K-means
kmeans = KMeans(n_clusters=5, random_state=42)
df['cluster'] = kmeans.fit_predict(df[features])

# Split data menjadi atribut (X) dan target (y)
X = df[features]
y = df['cluster']

# Standardisasi Data dengan Standard Scaler
from sklearn.preprocessing import StandardScaler
Scaler=StandardScaler()
X=Scaler.fit_transform(X)

X[0:3]
X.shape

# Split data menjadi train set dan test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
# Buat model Random Forest
model = RandomForestClassifier()
model.fit(X_train, y_train)

# Lakukan prediksi pada test set
y_pred = model.predict(X_test)

# Evaluasi model dengan akurasi
accuracy = accuracy_score(y_test, y_pred)
print("Akurasi:", accuracy)
```

Akurasi: 0.9777777777777777

Pada rumusan masalah ini setelah dilakukan clustering untuk masing-masing kota, dilakukan random forest classifier yang nantinya akan menentukan bagaimana keadaan / kondisi fitur-fitur yang ada berdasarkan kota administrasi pada dataset banjir DKI Jakarta 2020

Didapatkan akurasi dari model randomforest classifier sebesar 0.97

```

# Definisikan pemetaan nama cluster
nama_cluster = {
    1: 'Jakarta Utara',
    2: 'Jakarta Timur',
    3: 'Jakarta Selatan',
    4: 'Jakarta Pusat',
    5: 'Jakarta Barat'
}

# Looping untuk mendefinisikan setiap cluster
for i, center in enumerate(cluster_centers):
    cluster_id = i + 1
    print("Cluster", cluster_id)
    for j, feature in enumerate(features):
        print(f"{feature}: {round(center[j])}") # Mengonversi angka menjadi integer
    print(f"Nama Cluster: {nama_cluster.get(cluster_id)}")
    print("-----")

```

Dapat dilihat bahwa model ini adalah berarti untuk memprediksi dampak dan resiko yang akan terjadi untuk setiap kota di DKI Jakarta perbulannya.

resiko dapat berupa jumlah yang terdampak terluka, hingga ketinggian air di setiap kotanya.

## Model Rumusan Masalah 2

### OUTPUT

```

Cluster 1
jumlah_terdampak_rw: 1
jumlah_terdampak_rt: 2
jumlah_terdampak_kk: 1
jumlah_terdampak_jiwa: 2
ketinggian_air: 33
jumlah_luka_ringan: 3
jumlah_pengungsi_tertinggi: 0
jumlah_penduduk: 2434511
Nama Cluster: Jakarta Utara
-----
Cluster 2
jumlah_terdampak_rw: 1
jumlah_terdampak_rt: 2
jumlah_terdampak_kk: 8
jumlah_terdampak_jiwa: 25
ketinggian_air: 40
jumlah_luka_ringan: 5
jumlah_pengungsi_tertinggi: 0
jumlah_penduduk: 3037139
Nama Cluster: Jakarta Timur
-----
Cluster 3
jumlah_terdampak_rw: 2
jumlah_terdampak_rt: 2
jumlah_terdampak_kk: 1
jumlah_terdampak_jiwa: 6
ketinggian_air: 24
jumlah_luka_ringan: 7
jumlah_pengungsi_tertinggi: 0
jumlah_penduduk: 1778981
Nama Cluster: Jakarta Selatan
-----
```

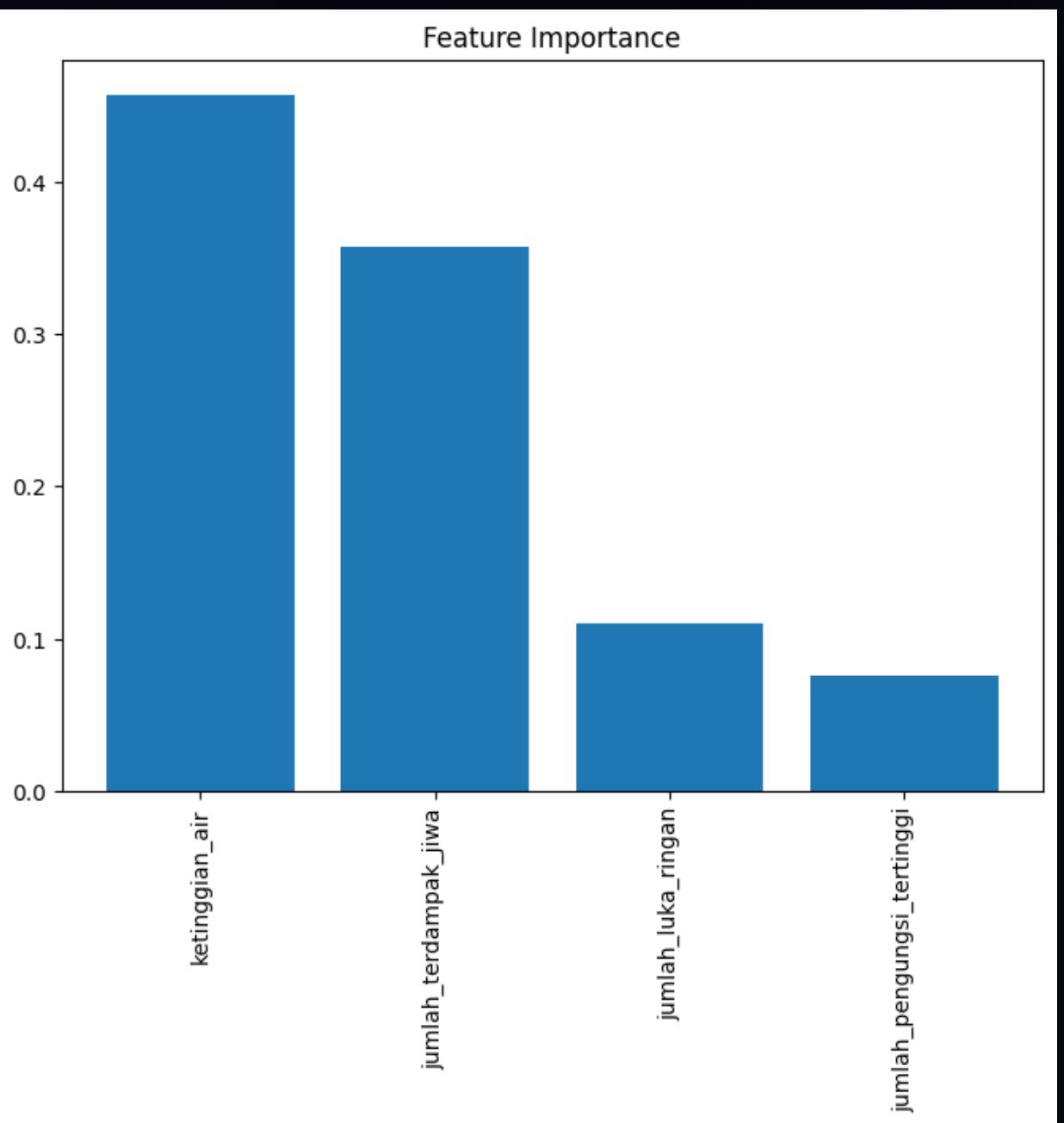
```

Cluster 4
jumlah_terdampak_rw: 2
jumlah_terdampak_rt: 2
jumlah_terdampak_kk: 1
jumlah_terdampak_jiwa: 5
ketinggian_air: 29
jumlah_luka_ringan: 7
jumlah_pengungsi_tertinggi: 0
jumlah_penduduk: 1056896
Nama Cluster: Jakarta Pusat
-----
Cluster 5
jumlah_terdampak_rw: 1
jumlah_terdampak_rt: 2
jumlah_terdampak_kk: 3
jumlah_terdampak_jiwa: 10
ketinggian_air: 40
jumlah_luka_ringan: 9
jumlah_pengungsi_tertinggi: 0
jumlah_penduduk: 2226812
Nama Cluster: Jakarta Barat
-----
```

# Feature Engineering

## Rumusan Masalah 2

### OUTPUT



Selain itu, kamu melakukan feature selection dan didapat ketinggian air memiliki fitur yang cukup penting dalam model dengan yang tidak begitu penting adalah jumlah pengungsi

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt
from sklearn.cluster import KMeans

# Pilih fitur-fitur yang akan digunakan
selected_features = ['jumlah_terdampak_jiwa', 'keterangan_air', 'jumlah_luka_ringan', 'jumlah_pengungsi_tertinggi']

# Preprocessing Data
# Lakukan clustering dengan K-means
kmeans = KMeans(n_clusters=5, random_state=42)
df['cluster'] = kmeans.fit_predict(df[features])

# Split data menjadi atribut (X) dan target (y)
X = df[selected_features]
y = df['cluster']

# Standardisasi Data dengan Standard Scaler
from sklearn.preprocessing import StandardScaler
Scaler=StandardScaler()
X=Scaler.fit_transform(X)

X[0:3]
X.shape

# Split data menjadi train set dan test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inisialisasi model Random Forest
model = RandomForestClassifier()

# Melatih model
model.fit(X_train, y_train)

# Mengambil fitur-fitur terpenting
feature_importance = model.feature_importances_

# Membuat dataframe fitur-fitur terpenting
feature_importance_df = pd.DataFrame({'Feature': selected_features, 'Importance': feature_importance})

# Mengurutkan fitur berdasarkan tingkat penting
feature_importance_df = feature_importance_df.sort_values('Importance', ascending=False)

# Mengatur ukuran figure
plt.figure(figsize=(8, 6))

# Membuat plot barchart
plt.bar(feature_importance_df['Feature'], feature_importance_df['Importance'])

# Mengatur label pada sumbu x
plt.xticks(rotation=90)

# Memberi judul pada plot
plt.title('Feature Importance')

# Menampilkan plot
plt.show()
```

## Model Rumusan Masalah 3

```
1 import pandas as pd
2 from sklearn.cluster import KMeans
3 from sklearn.ensemble import RandomForestClassifier
4 from sklearn.model_selection import train_test_split
5 from sklearn.metrics import accuracy_score
6
7 # Pilih fitur-fitur yang akan digunakan
8 features = ['Rata-Rata Kelembapan Udara (%)', 'Rata-Rata Suhu Udara (°C)', 'jumlah_terdampak_jiwa', 'ketinggian_air', 'jumlah_penduduk',
9 | | | | | 'Curah Hujan (mm)', 'Hari Hujan']
10
11 # Preprocessing Data
12 # Lakukan clustering dengan K-means
13 kmeans = KMeans(n_clusters=5, random_state=42)
14 df['cluster'] = kmeans.fit_predict(df[features])
15
16 # Split data menjadi atribut (X) dan target (y)
17 X = df[features]
18 y = df['cluster']
19
20 # Standardisasi Data dengan Standard Scaler
21 from sklearn.preprocessing import StandardScaler
22 Scaler=StandardScaler()
23 X=Scaler.fit_transform(X)
24
25 X[0:3]
26 X.shape
27
28 # Split data menjadi train set dan test set
29 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
30
```

Pada rumusan masalah ini setelah dilakukan clustering untuk masing-masing kota, dilakukan random forest classifier yang nantinya akan menentukan bagaimana keadaan / kondisi fitur-fitur yang digunakan berdasarkan kota administrasi pada dataset banjir DKI Jakarta 2018

untuk kali ini fitur yang digunakan berupa faktor alam seperti kelembapan, suhu udara, ketinggian air, curah hujan, dan hari hujan.

## Model Rumusan Masalah 3

```
31 # Buat model Random Forest
32 model = RandomForestClassifier()
33 model.fit(X_train, y_train)
34
35 # Lakukan prediksi pada test set
36 y_pred = model.predict(X_test)
37
38 # Evaluasi model dengan akurasi
39 accuracy = accuracy_score(y_test, y_pred)
40 print("Akurasi:", accuracy)
```

### OUTPUT

Akurasi: 0.9615384615384616

## Model Rumusan Masalah 3

```
1 # Definisikan pemetaan nama cluster
2 cluster_centers = kmeans.cluster_centers_
3 nama_cluster = {
4     1: 'Jakarta Timur',
5     2: 'Jakarta Utara',
6     3: 'Jakarta Selatan',
7     4: 'Jakarta Pusat',
8     5: 'Jakarta Barat'
9 }
10
11 # Looping untuk mendefinisikan setiap cluster
12 for i, center in enumerate(cluster_centers):
13     cluster_id = i + 1
14     print("Cluster", cluster_id)
15     for j, feature in enumerate(features):
16         print(f"{feature}: {round(center[j])}") # Mengonversi angka menjadi integer
17     print(f"Nama Cluster: {nama_cluster.get(cluster_id)}")
18     print("-----")
```

Dapat dilihat bahwa model ini adalah berarti untuk memprediksi kondisi alam yang memengaruhi banjir serta resiko yang akan terjadi untuk setiap kota di DKI Jakarta perbulannya.

resiko dapat berupa ketinggian air dan jumlah hari hujan dan kondisi alam berupa kelembapan, curah hujan, dan lain-lain.

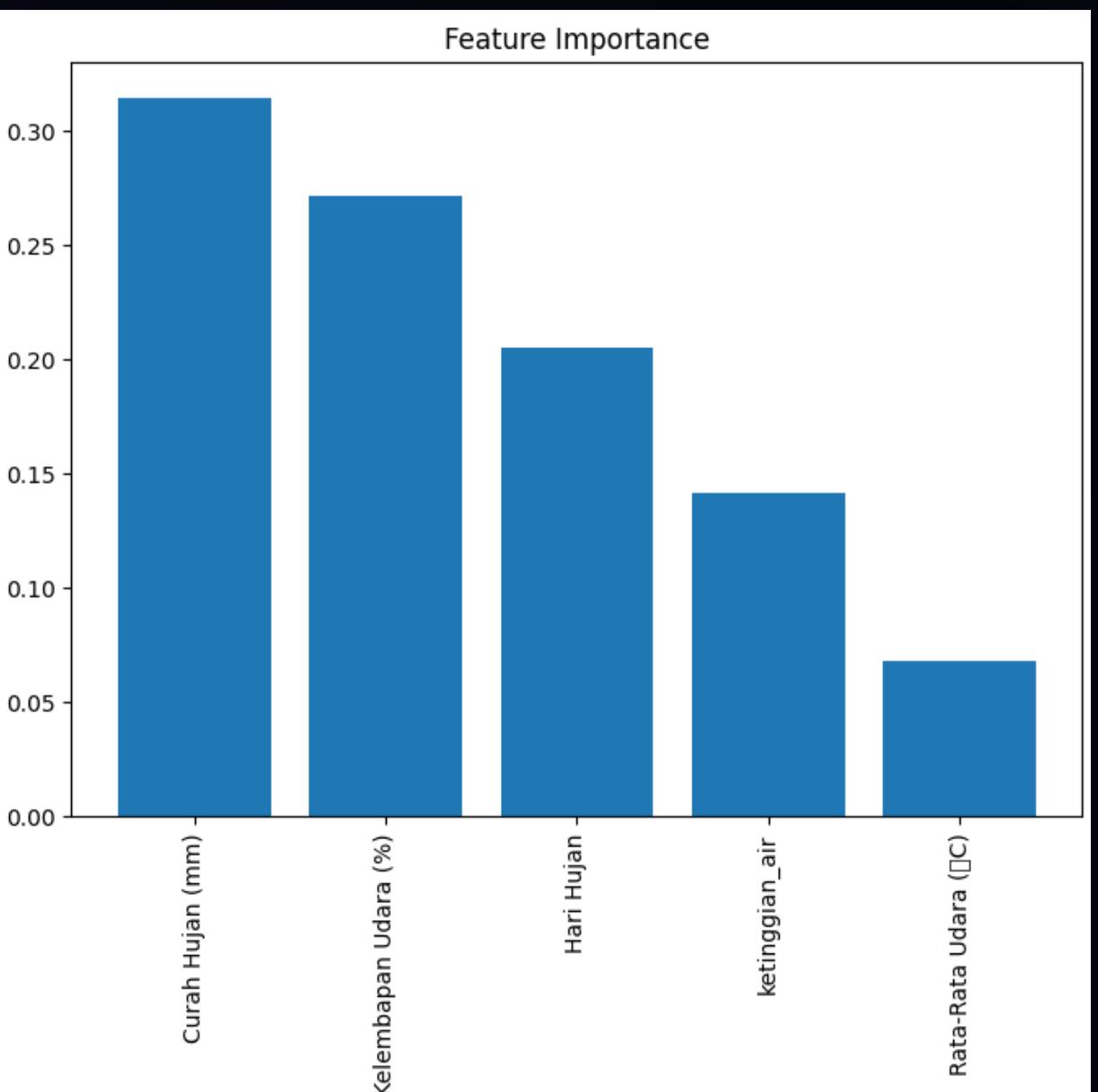
## OUTPUT

```
Cluster 1
Rata-Rata Kelembapan Udara (%): 77
Rata-Rata Udara (.C): 29
jumlah_terdampak_jiwa: 284
ketinggian_air: 48
jumlah_penduduk: 2916018
Curah Hujan (mm): 233
Hari Hujan: 15
Nama Cluster: Jakarta Timur
-----
Cluster 2
Rata-Rata Kelembapan Udara (%): 78
Rata-Rata Udara (.C): 28
jumlah_terdampak_jiwa: 15
ketinggian_air: 18
jumlah_penduduk: 1797292
Curah Hujan (mm): 252
Hari Hujan: 20
Nama Cluster: Jakarta Utara
-----
Cluster 3
Rata-Rata Kelembapan Udara (%): 76
Rata-Rata Udara (.C): 29
jumlah_terdampak_jiwa: 131
ketinggian_air: 50
jumlah_penduduk: 2246137
Curah Hujan (mm): 168
Hari Hujan: 14
Nama Cluster: Jakarta Selatan
-----
Cluster 4
Rata-Rata Kelembapan Udara (%): 79
Rata-Rata Udara (.C): 29
jumlah_terdampak_jiwa: 0
ketinggian_air: 26
jumlah_penduduk: 924686
Curah Hujan (mm): 327
Hari Hujan: 18
Nama Cluster: Jakarta Pusat
-----
Cluster 5
Rata-Rata Kelembapan Udara (%): 79
Rata-Rata Udara (.C): 28
jumlah_terdampak_jiwa: 4
ketinggian_air: 30
jumlah_penduduk: 2559362
Curah Hujan (mm): 298
Hari Hujan: 19
Nama Cluster: Jakarta Barat
```

# Feature Engineering

## Rumusan Masalah 3

### OUTPUT



```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
import matplotlib.pyplot as plt

# Pilih fitur-fitur yang akan digunakan
selected_features = ['Rata-Rata Kelembapan Udara (%)', 'Rata-Rata Suhu Udara (°C)', 'Ketinggian Air', 'Curah Hujan (mm)', 'Hari Hujan']

# Preprocessing Data
# Lakukan clustering dengan K-means
kmeans = KMeans(n_clusters=5, random_state=42)
df['cluster'] = kmeans.fit_predict(df[features])

# Split data menjadi atribut (X) dan target (y)
X = df[selected_features]
y = df['cluster']

# Standardisasi Data dengan Standard Scaler
from sklearn.preprocessing import StandardScaler
Scaler=StandardScaler()
X=Scaler.fit_transform(X)

X[0:3]
X.shape

# Split data menjadi train set dan test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inisialisasi model Random Forest
model = RandomForestClassifier()

# Melatih model
model.fit(X_train, y_train)

# Mengambil fitur-fitur terpenting
feature_importance = model.feature_importances_

# Membuat dataframe fitur-fitur terpenting
feature_importance_df = pd.DataFrame({'Feature': selected_features, 'Importance': feature_importance})

# Mengurutkan fitur berdasarkan tingkat penting
feature_importance_df = feature_importance_df.sort_values('Importance', ascending=False)

# Mengatur ukuran figure
plt.figure(figsize=(8, 6))

# Membuat plot barchart
plt.bar(feature_importance_df['Feature'], feature_importance_df['Importance'])

# Mengatur label pada sumbu x
plt.xticks(rotation=90)

# Memberi judul pada plot
plt.title('Feature Importance')

# Menampilkan plot
plt.show()
```

Selain itu, kamu melakukan feature selection dan didapat curah hujan memiliki fitur yang cukup penting dalam model dengan yang tidak begitu penting adalah rata-rata suhu udara.

# List of Content

05

Kesimpulan  
&  
Saran

Kesimpulan serta saran  
terkait permasalah

# Kesimpulan

Berdasarkan model yang telah dibuat, didapatkan kesimpulan sebagai berikut.

1. Hubungan antara jumlah pengungsi tertinggi dengan jumlah luka ringan korban pada bencana banjir jakarta tahun 2020 memiliki hubungan yang **belum optimal** hal ini ditunjukkan dengan nilai **R-Square** dalam model regresi yang bernilai **0,68**
2. Kondisi banjir untuk setiap kota administrasi di wilayah DKI Jakarta pada bulan januari 2021 **terdapat kenaikan** maupun **penurunan** dalam hal ketinggian air dan jumlah terdampak jiwa
3. Kondisi kelembaban, temperatur curah hujan, dan hari hujan yang memengaruhi kondisi banjir untuk setiap kota administrasi di wilayah DKI Jakarta pada bulan januari 2019 **mengalami kenaikan** maupun **penurunan**
4. Model Clustering Wilayah Banjir DKI Jakarta 2020 memiliki fitur terpenting dan paling berpengaruh berupa **ketinggian air dan jumlah terdampak jiwa**.
5. Model Clustering Wilayah Banjir DKI Jakarta 2018 yang dipengaruhi faktor alam memiliki fitur terpenting dan paling berpengaruh berupa **Curah Hujan dan Kelembapan Udara**.



# Saran

Berikut adalah saran/ rekomendasi yang berkaitan dengan permasalahan

1. Dalam Observasi Data dapat dilakukan pencarian dataset lebih dalam untuk mengkaji permasalahan banjir terutama untuk clustering wilayah banjir dan model regresi dengan fitur-fitur yang berkaitan kuat, seperti ketinggian tanah, Kepadatan penduduk, dan lain-lain.
2. Pada tahapan data cleaning, didapatkan bahwa masih terdapat outlier yang ada pada dataset. namun, untuk penghapusannya diperlukan metode yang lebih tepat dan sesuai untuk akhirnya model prediksi dapat berjalan dengan baik.



**Team AIRy - COMPFEST 15**

**Data Science Academy**

**Terima  
Kasih**

...

**Team AIRy**

---