Kelompok Tan 45

Final Project

Compfest

Tema : “Kolaborasi Data untuk Kemanusiaan”

1. Perumusan Masalah:
2. Bagaimanakah sebaran bencana alam yang terjadi di Indonesia?
3. Bencana alam apakah yang memiliki frekuensi tertinggi terjadi di Indonesia?
4. Kapan bencana alam terjadi dengan frekuensi tertinggi?
5. Daerah manakah yang berpotensi rentan terkena bencana?
6. Apakah ketersediaan rumah sakit/puskesmas sudah memadai di daerah rawan bencana?
7. Hipotesis

Kota-kota besar seperti Jakarta akan lebih banyak mengalami bencana alam, data bencana alam yang terjadi di kota-kota besar akan lebih sering ditemui pada dataset Peta Bencana karena pengaplikasian sistem pengumpulan data yang didasarkan pada laporan masyarakat. Hal ini didukung oleh latar belakang pendidikan dan pengetahuan masyarakat.

Bencana alam yang banyak terjadi kemungkinan adalah banjir, karena bencana alam ini bukan hanya dipengaruhi oleh faktor cuaca namun juga oleh perilaku manusianya.

1. Metodologi dan Variabel

Identifikasi Variabel

Variabel yang diangkat pada *final project* ini yaitu :  
1. Variabel bebas adalah faktor sebab (variabel X) : Peta Bencana

2. Variabel terikat adalah faktor akibat (variabel Y) : Puskesmas Bekasi dan Jakarta

Metodologi

Metodologi yang digunakan adalah metode analisis data kuantitatif. Data yang digunakan merupakan data yang dapat diukur atau dinumerikan, sehingga data ini dapat diolah dengan teknik statistik dan komputasi atau dapat divisualisasikan melalui metode yang digunakan. Selain itu, data ini juga diolah dengan metode *Descriptive Analytics* yang bertujuan untuk mendapatkan gambaran umum dari data yang telah terkumpul untuk membantu menjawab dari rumusan masalah yang ada.

1. Analisis Awal
2. Dataset yang digunakan adalah peta bencana, puskesmas di Jakarta, dan puskesmas di Bekasi
3. Terdapat kolom dengan data yang berformat json dan perlu di pharsing
4. Terdapat penambahan kolom pada dataset setelah pharsing sebanyak 5 kolom dan 1 baris
5. Terdapat 15 data bertipe objek dan 12 data bertipe float
6. Terdapat missing value pada setiap dataset
7. Terdapat 8 data duplikat
8. Missing value diganti dengan 0 untuk tipe data integer/float dan '-' untuk tipe data kategori atau objek
9. Beberapa kolom yang tidak penting dieliminasi dari dataset
10. Konversi atribut bertipe objek menjadi datetime64, integer, dan kategori
11. Mean pada kolom points tidak dapat digunakan untuk mereplace missing value
12. Analisis Mendalam
13. Dimensi pada data terbaru sebanyak 4236 baris dan 24 kolom
14. Terdapat atribut bertipe float sebanyak 10, object 8, category 5, dan datetime 1
15. Terdapat 3 sumber data pelaporan bencana yaitu grasp, qlue, dan detik
16. Sumber pelaporan terbanyak yaitu dari grasp
17. Status pelaporan sudah terkonfirmasi semua
18. Kesimpulan dan Rekomendasi
19. Persebaran bencana alam di Indonesia banyak terjadi di Pulau Jawa
20. Bencana alam lebih sering terjadi pada tahun 2020 dibandingkan tahun 2021.
21. Banjir merupakan bencana alam dengan frekuensi tertinggi yang terjadi di Indonesia dan hampir terjadi di seluruh wilayah di Indonesia.
22. Kedalaman banjir yang paling tinggi adalah setinggi 2 meter yang terjadi di 14 daerah di Indonesia.
23. 5 daerah yang paling rentan terkena bencana alam di Indonesia adalah Jakarta Timur, Jakarta Barat, Bekasi, Jakarta Utara, dan Jakarta Selatan. Bencana yang rawan melanda kelima daerah tersebut adalah banjir.
24. Ketersediaan fasilitas kesehatan (puskesmas) di 5 daerah paling rawan terkena bencana sudah mencukupi kebutuhan daerah tersebut ketika terjadi bencana berdasarkan perbandingan keterjadian bencana dengan banyaknya puskesmas yang ada. Korban-korban bencana alam akan terevakuasi lebih cepat tanpa khawatir tidak akan mendapatkan fasilitas kesehatan.

Dataset yang digunakan:

1.Peta Bencana

2. Puskesmas Bekasi

3. Puskesmas DKI Jakarta

# import packages

import pandas as pd

import numpy as np

import matplotlib as plt

from datetime import datetime

import matplotlib.pyplot as plt

import seaborn as sns

import folium

from folium import Marker, GeoJson

from folium.plugins import MarkerCluster, HeatMap

Library:

Pandas: membaca dan mengolah data

Numpy: operasi matematika dan statistika

Matplotlib: visualisasi data

Seaborn: visualisasi data

Folium: spatial analisis dan visualisasi

datetime: mengubah tipe data menjadi format datetime

**Data Wrangling dan Pre-Processing**

Load Dataset:

df = pd.read\_csv('/work/PetaBencana.csv')

df\_new = pd.read\_csv('/work/PetaBencana\_Convert.csv')

pb = pd.read\_csv('/work/Puskesmas Bekasi.csv')

pj = pd.read\_csv('/work/Puskesmas Jakarta.csv')

**EXPLORE DATA**

**#Melihat pengaruh parsing dari kolom report\_dat dan tags**

**display(df.shape) #ada 4235 baris dan 22 kolom**

**display(df\_new.shape) #Bertambah satu baris dan 5 kolom yakni menjadi 4236 baris 27 kolom**

Karena didapatkan kolom yang masih dalam format json, maka kami melakukan parsing kolom tersebut menggunakan power query melalui excel. Sehingga didapatkan penambahan kolom.

display(df.columns)

display(df\_new.columns) #yakni penambahan pada kolom report\_type, flood\_depth, points, district\_id,

#local\_area\_id, dan instance\_region\_code

df\_new.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 4236 entries, 0 to 4235

Data columns (total 27 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

0 pkey 4235 non-null float64

1 date 4235 non-null object

2 time 4235 non-null object

3 source 4235 non-null object

4 status 4235 non-null object

5 url 3570 non-null object

6 image\_url 3334 non-null object

7 disaster\_type 4235 non-null object

8 report\_type 3382 non-null object

9 flood\_depth 3155 non-null float64

10 points 792 non-null float64

11 district\_id 2316 non-null float64

12 local\_area\_id 2369 non-null float64

13 instance\_region\_code 4213 non-null object

14 title 188 non-null object

15 text 3830 non-null object

16 ID\_Desa 4227 non-null float64

17 ID\_Kec 4227 non-null float64

18 ID\_Kab 4227 non-null float64

19 ID\_Prov 4227 non-null float64

20 Desa 4227 non-null object

21 Kecamatan 4227 non-null object

22 Kabupaten 4228 non-null object

23 Provinsi 4228 non-null object

24 long 4235 non-null float64

25 lat 4235 non-null float64

26 Unnamed: 26 0 non-null float64

dtypes: float64(12), object(15)

memory usage: 893.7+ KB

Melihat type data dan melihat data yang null.

#Mengetahui jumlah tipe tiap data

pd.value\_counts(df\_new.dtypes)#ditemukan hanya ada dua tipe data yakni objek dan float

object 15

float64 12

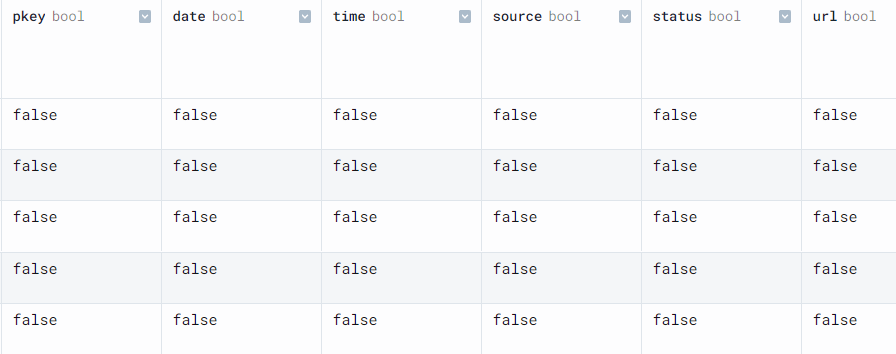
Terdapat 15 object dan 12 float.

**Missing Value**

#data new

new\_missing\_value = df\_new.isnull()

new\_missing\_value.head() #Melihat secara sekilas bagaimana missing value pada dataframe

****

false menunjukkan data tidak kosong, true menunjukkan missing value.

# Count missing value in each column Peta Bencana

null\_columns = df\_new.isnull().sum().sort\_values(ascending=False)

null\_columns

title 4048

points 3444

district\_id 1920

local\_area\_id 1867

flood\_depth 1081

image\_url 902

report\_type 854

url 666

text 406

instance\_region\_code 23

ID\_Desa 9

ID\_Kec 9

ID\_Kab 9

ID\_Prov 9

Desa 9

Kecamatan 9

Kabupaten 8

Provinsi 8

disaster\_type 1

status 1

date 1

source 1

time 1

long 1

lat 1

pkey 1

Melihat missing value melalui akumulasi, dan membuat perencanaan aksi terhadap missing value tersebut berdasarkan jumlahnya.

# Count missing value in each column Puskesmas Jakarta

null\_columns1 = pj.isnull().sum().sort\_values(ascending=False)

null\_columns1

nama\_puskesmas 1023

Kabupaten 1

alamat\_puskesmas 0

# Count missing value in each column Peta Bencana

null\_columns2 = pb.isnull().sum().sort\_values(ascending=False)

null\_columns2

KETERANGAN 43

NO 0

KECAMATAN 0

PUSKESMAS / KELURAHAN 0

ALAMAT 0

KONDISI BAIK 0

KONDISI KURANG BAIK 0

terdapat 43 missing value pada kolom KETERANGAN.

df\_new['points'] = df\_new['points'].fillna(0)

df\_new['ID\_Desa'] = df\_new['ID\_Kec'].fillna(0)

df\_new['ID\_Kec'] = df\_new['ID\_Kec'].fillna(0)

df\_new['ID\_Kab'] = df\_new['ID\_Kab'].fillna(0)

df\_new['ID\_Prov'] = df\_new['ID\_Prov'].fillna(0)

df\_new['long'] = df\_new['long'].fillna(0)

df\_new['lat'] = df\_new['lat'].fillna(0)

df\_new['flood\_depth'] = df\_new['flood\_depth'].fillna(0)

df\_new['district\_id'] = df\_new['district\_id'].fillna(0)

df\_new['local\_area\_id'] = df\_new['local\_area\_id'].fillna(0)

df\_new['source'] = df\_new['source'].fillna('-')

df\_new['status'] = df\_new['status'].fillna('-')

df\_new['url'] = df\_new['url'].fillna('-')

df\_new['image\_url'] = df\_new['image\_url'].fillna('-')

df\_new['disaster\_type'] = df\_new['disaster\_type'].fillna('-')

df\_new['report\_type'] = df\_new['report\_type'].fillna('-')

df\_new['instance\_region\_code'] = df\_new['instance\_region\_code'].fillna('-')

df\_new['text'] = df\_new['text'].fillna('-')

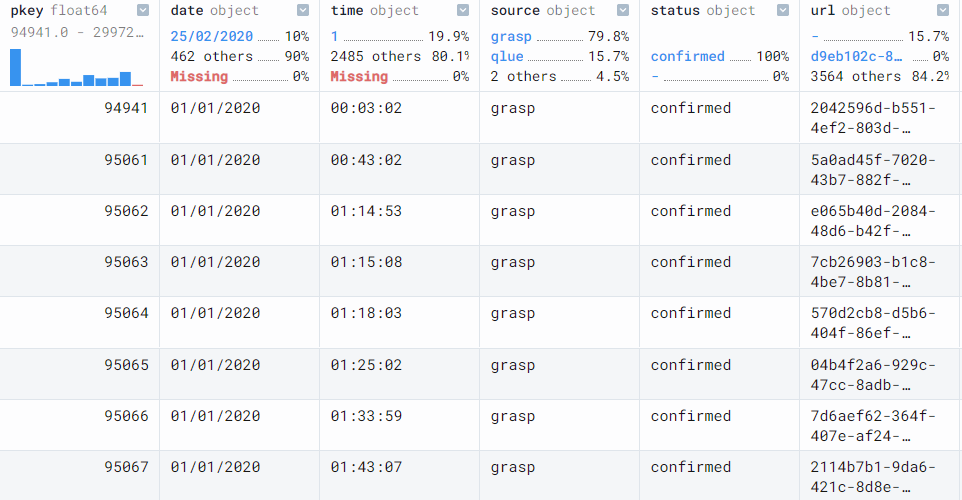
df\_new['Desa'] = df\_new['Desa'].fillna('-')

df\_new['Kecamatan'] = df\_new['Kecamatan'].fillna('-')

df\_new['Kabupaten'] = df\_new['Kabupaten'].fillna('-')

df\_new['Provinsi'] = df\_new['Provinsi'].fillna('-')

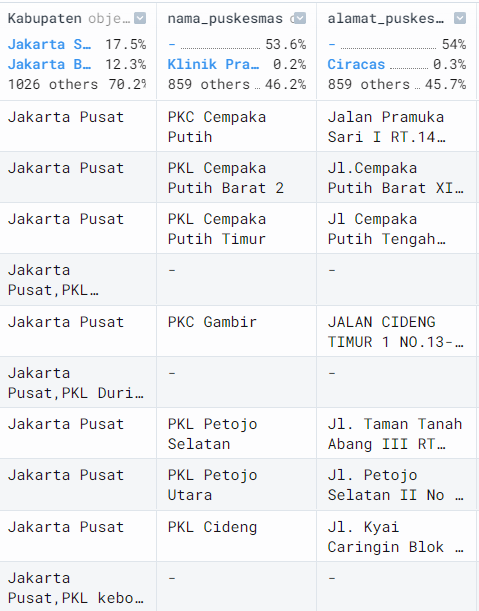
df\_new



pj['Kabupaten'] = pj['Kabupaten'].fillna('-')

pj['nama\_puskesmas'] = pj['nama\_puskesmas'].fillna('-')

pj



Aksi yang dilakukan terhadap missing value, yakni mengisi setiap missing value dengan ‘-’ untuk tipe data non-numeric, dan ‘0’ untuk data numeric.

df\_new['date'] = df\_new['date'].astype('datetime64[ns]')

df\_new['source'] = df\_new['source'].astype('category')

df\_new['status'] = df\_new['status'].astype('category')

df\_new['report\_type'] = df\_new['report\_type'].astype('category')

df\_new['disaster\_tyoe'] = df\_new['disaster\_type'].astype('category')

df\_new['instance\_region\_code'] = df\_new['instance\_region\_code'].astype('category')

df\_new.info()

Column Non-Null Count Dtype

--- ------ -------------- -----

0 pkey 4235 non-null float64

1 date 4235 non-null datetime64[ns]

2 time 4235 non-null object

3 source 4236 non-null category

4 status 4236 non-null category

5 url 4236 non-null object

6 image\_url 4236 non-null object

7 disaster\_type 4236 non-null object

8 report\_type 4236 non-null category

9 flood\_depth 4236 non-null float64

10 points 4236 non-null float64

11 district\_id 4236 non-null float64

12 local\_area\_id 4236 non-null float64

13 instance\_region\_code 4236 non-null category

14 title 188 non-null object

15 text 4236 non-null object

16 ID\_Desa 4236 non-null float64

17 ID\_Kec 4236 non-null float64

18 ID\_Kab 4236 non-null float64

19 ID\_Prov 4236 non-null float64

20 Desa 4236 non-null object

21 Kecamatan 4236 non-null object

22 Kabupaten 4236 non-null object

23 Provinsi 4236 non-null object

24 long 4236 non-null float64

25 lat 4236 non-null float64

26 Unnamed: 26 0 non-null float64

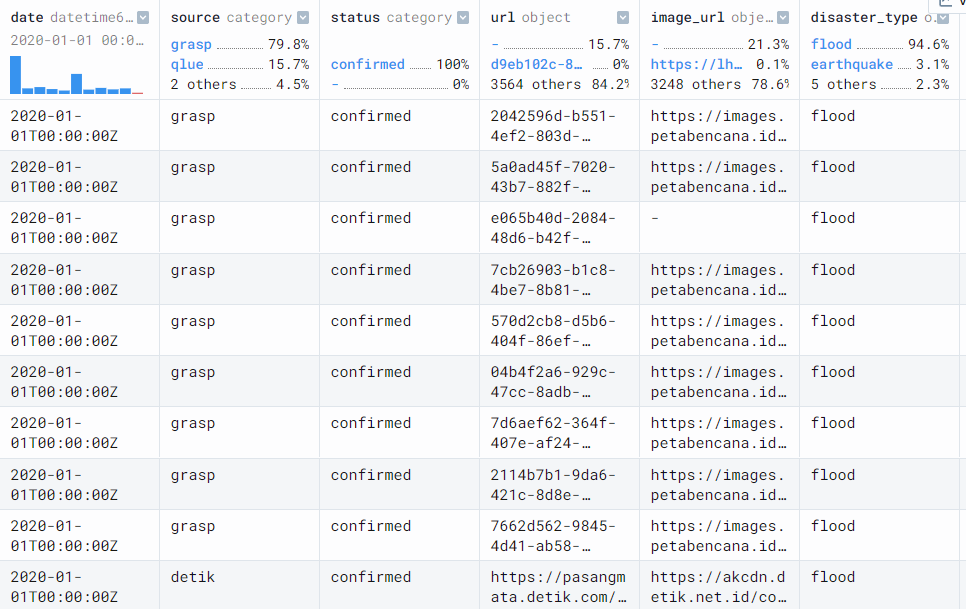
27 disaster\_tyoe 4236 non-null category

dtypes: category(5), datetime64[ns](1), float64(12), object(10)

Mengubah beberapa tipe data menjadi datetime dan category, untuk keperluan olah data.

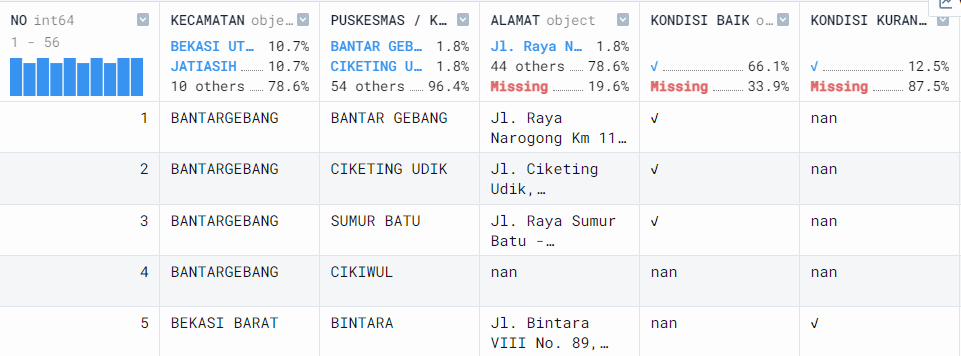
df\_news = df\_new.drop(columns=['title','time','pkey','Unnamed: 26'])

df\_news



pb\_new = pb.drop(columns=['KETERANGAN'])

pb\_new



Menghapus kolom yang terdapat banyak missing value dan tidak terpakai pada analisis.

#Menghitung data yang duplicate pada data yang baru

pd.value\_counts(df\_news.duplicated()).dropna() #ada 8 data yang duplicate

False 4228

True 8

Melihat jumlah data yang duplikat. Ada 8 data yang duplikat.

df\_news.isnull().sum()

date 1

source 0

status 0

url 0

image\_url 0

disaster\_type 0

report\_type 0

flood\_depth 0

points 0

district\_id 0

local\_area\_id 0

instance\_region\_code 0

text 0

ID\_Desa 0

ID\_Kec 0

ID\_Kab 0

ID\_Prov 0

Desa 0

Kecamatan 0

Kabupaten 0

Provinsi 0

long 0

lat 0

disaster\_tyoe 0

Melihat data yang kosong pada tiap kolom.Terdapat satu data duplikat pada kolom date.

#cek mean data points

mean=df\_news['points'].mean()

mean

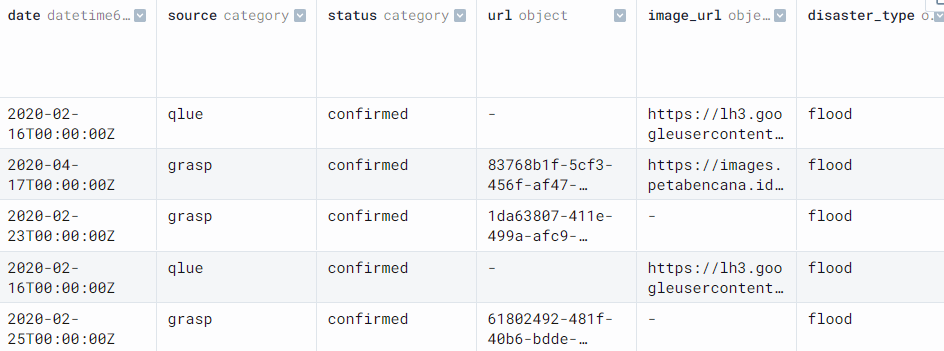
0.1789423984891407

Rencananya mean pada kolom points ini akan digunakan untuk me-*replace* missing value pada kolom tersebut, namun karena angkanya tidak memuaskan maka tidak dipakai.

**EXPLORATORY DATA ANALYSIS (EDA)**

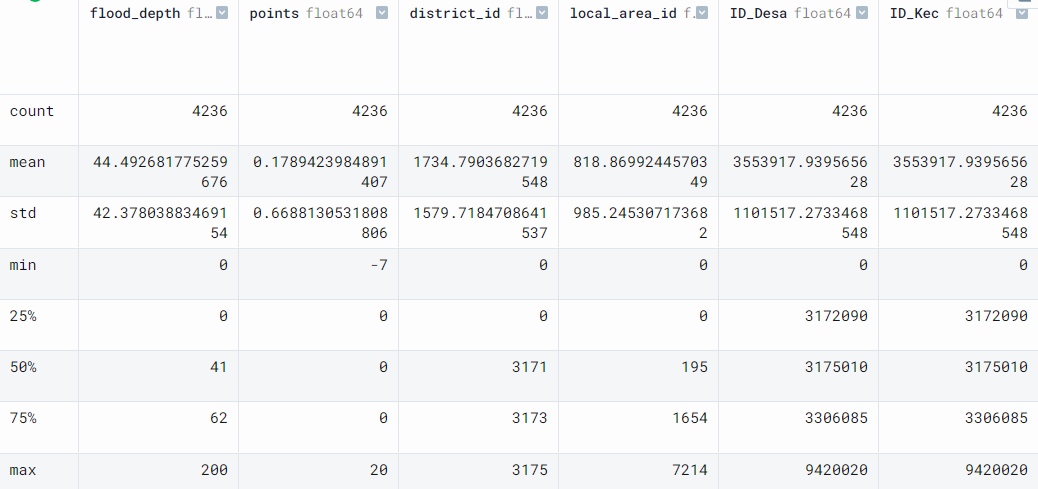
**Structure Investigation**

df\_news.sample(5)



Mengambil 5 data sampel dari data frame untuk ditampilkan.

df\_news.describe()



Melihat summary data secara cepat seperti jumlah, rata-rata, quartil, dan simpangan baku.

df\_news.tail()



Menampilkan 5 data terbawwah dari data frame.

df\_news.dtypes

date datetime64[ns]

source category

status category

url object

image\_url object

disaster\_type object

report\_type category

flood\_depth float64

points float64

district\_id float64

local\_area\_id float64

instance\_region\_code category

text object

ID\_Desa float64

ID\_Kec float64

ID\_Kab float64

ID\_Prov float64

Desa object

Kecamatan object

Kabupaten object

Provinsi object

long float64

lat float64

disaster\_tyoe category

Melihat tipe data setelah dilakukan preprocessing.

df\_news['disaster\_type'].unique()

array(['flood', 'earthquake', 'wind', 'fire', 'volcano', 'haze', '-'],

dtype=object)

df\_news['source'].unique()

['grasp', 'detik', 'qlue', '-']

Categories (4, object): ['grasp', 'detik', 'qlue', '-']

df\_news['points'].unique()

array([ 0., 1., 2., -1., 4., -2., 8., 5., 3., -3., 6., -7., 20.])

df\_news['status'].unique()

['confirmed', '-']

Categories (2, object): ['confirmed', '-']

df\_news['Provinsi'].unique()

array(['Daerah Khusus Ibukota Jakarta', 'Jawa Barat', 'Banten',

'Jawa Tengah', 'Jawa Timur', 'Kalimantan Timur',

'Sumatera Selatan', 'Daerah Istimewa Yogyakarta', 'Lampung',

'Nusa Tenggara Barat', 'Sumatera Utara', 'Gorontalo',

'Kalimantan Selatan', 'Sulawesi Selatan',

'Kepulauan Bangka Belitung', 'Aceh', 'Bali', 'Riau',

'Kalimantan Barat', 'Bengkulu', 'Sumatera Barat', 'Papua Barat',

'Kalimantan Tengah', 'Sulawesi Tengah', 'Maluku', 'Sulawesi Barat',

'Sulawesi Utara', 'Jambi', 'Kepulauan Riau', 'Papua',

'Sulawesi Tenggara', '-', 'Nusa Tenggara Timur',

'Kalimantan Utara', 'Maluku Utara'], dtype=object)

df\_news['report\_type'].unique()

['flood', '-', 'structure', 'road', 'wind', 'fire', 'volcano', 'haze']

Categories (8, object): ['flood', '-', 'structure', 'road', 'wind', 'fire', 'volcano', 'haze']

Melihat unique value (data yang mengisi kolom), pada beberapa kolom.

df\_news['source'].value\_counts()

grasp 3382

qlue 665

detik 188

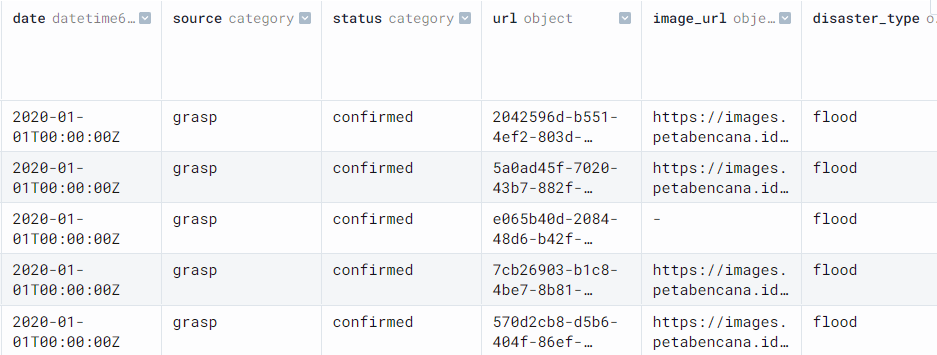
- 1

Melihat jumlah sumber data pelaporan bencana alam didapatkan.

**Structure of non-numerical features**

# Display non-numerical features

df\_news.select\_dtypes(exclude="number").head()



Menampilkan data yang tidak mengandung numerical.

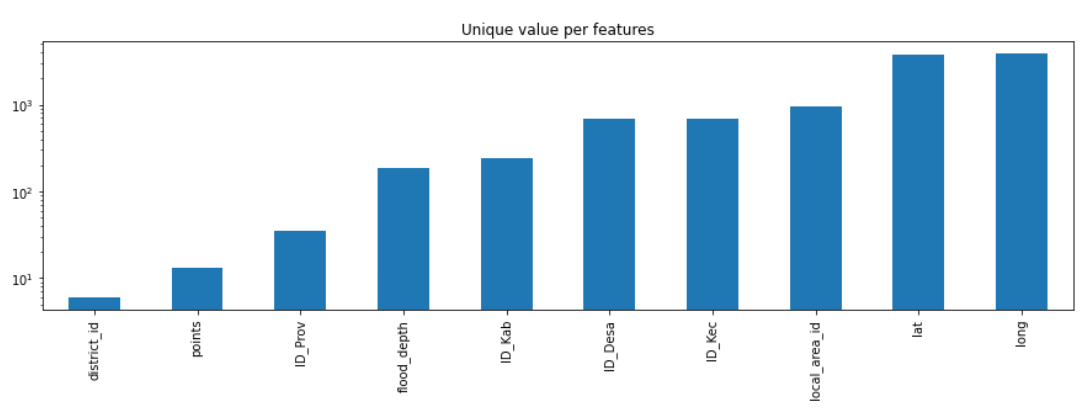
**Structure of numerical features**

# For each numerical feature compute number of unique entries

unique\_val = df\_news.select\_dtypes(include="number").nunique().sort\_values()

# Plot information with y-axis in log-scale

unique\_val.plot.bar(logy=True, figsize=(15,4), title="Unique value per features")



Memvisualisasikan data numeric menggunakan bar-chart.

**Conclusion of Structure Investigation**

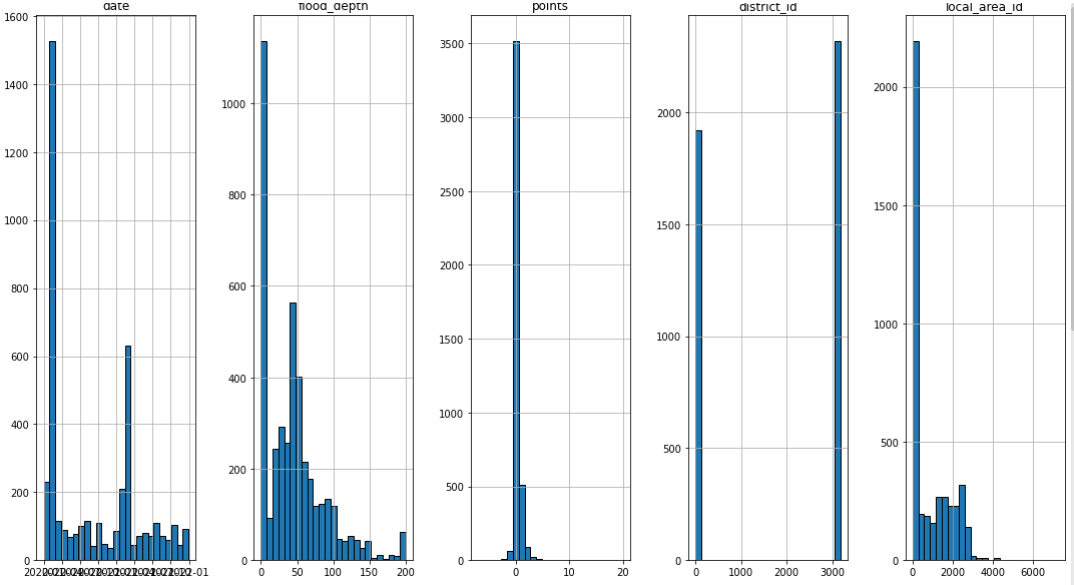
1. we know that we have 4236 samples and 24 atributes
2. as

**Content Investigation**

# Plots the histogram for each numerical feature in a separate subplot

df\_news.hist(bins=25, figsize=(15,25), layout=(-1,5), edgecolor='black')

plt.tight\_layout()

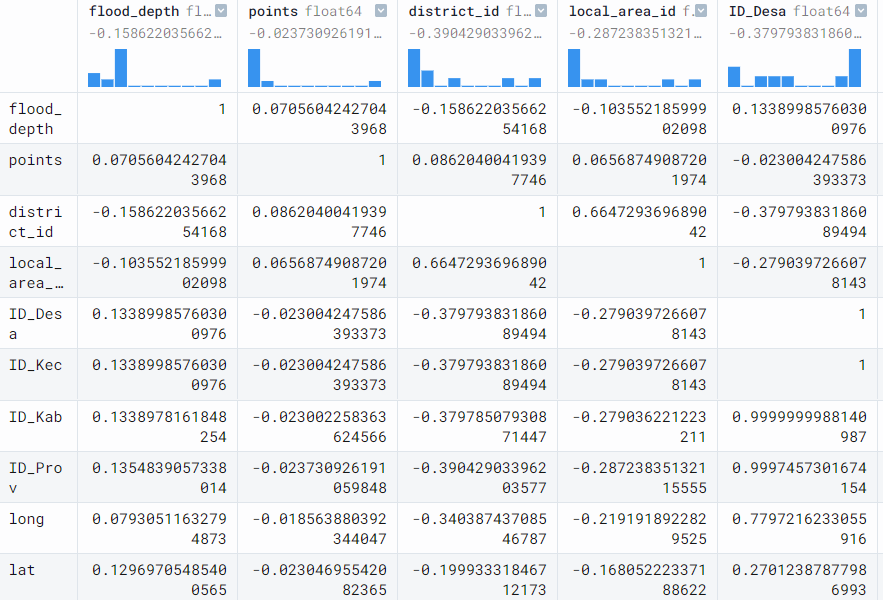


Menampilkan data numeric pada setiap subplot dalam histogram

# korelasi

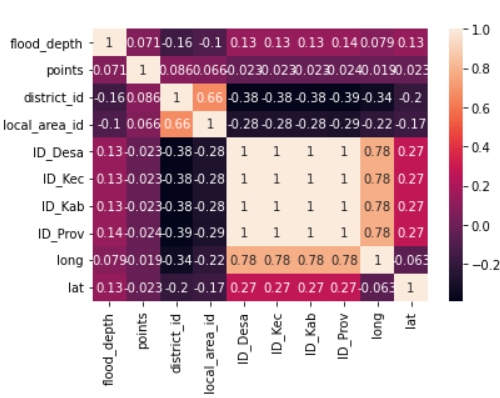
korelasi = df\_news.corr()

korelasi



Melihat korelasi melalui tabel.

sns.heatmap(korelasi, annot=True)



Membuat visualisasi dari korelasi, menggunakan heatmap.

loc\_center = [df\_news['lat'].mean(), df\_news['long'].mean()]

df\_test = df\_news[['lat', 'long']]

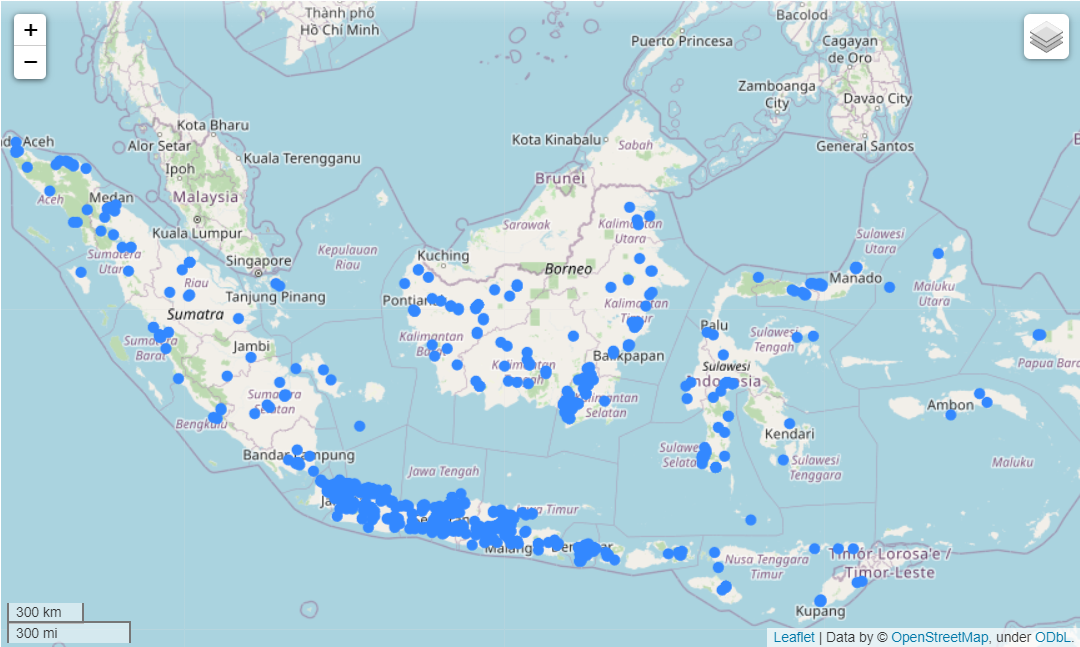
map1 = folium.Map(location = loc\_center, tiles='Openstreetmap', zoom\_start = 5, control\_scale=True)

for index, loc in df\_test.iterrows():

folium.CircleMarker([loc['lat'], loc['long']], radius=2, weight=5, popup=df\_news['disaster\_type'][index]).add\_to(map1)

folium.LayerControl().add\_to(map1)

map1



Menampilkan sebaran bencana di Indonesia menggunakan visualisasi map.

map2 = folium.Map(location = loc\_center, tiles='Stamen Terrain', zoom\_start = 5, control\_scale=True)

df\_test = df\_news[['lat', 'long','disaster\_type']]

for index, loc in df\_test.iterrows():

if loc['disaster\_type']=='flood':

color = 'blue'

elif loc['disaster\_type']=='earthquake':

color = "yellow"

elif loc['disaster\_type']=='wind':

color = 'white'

elif loc['disaster\_type']=='volcano':

color = 'red'

elif loc['disaster\_type']=='fire':

color = 'orange'

elif loc['disaster\_type']=='haze':

color = 'purple'

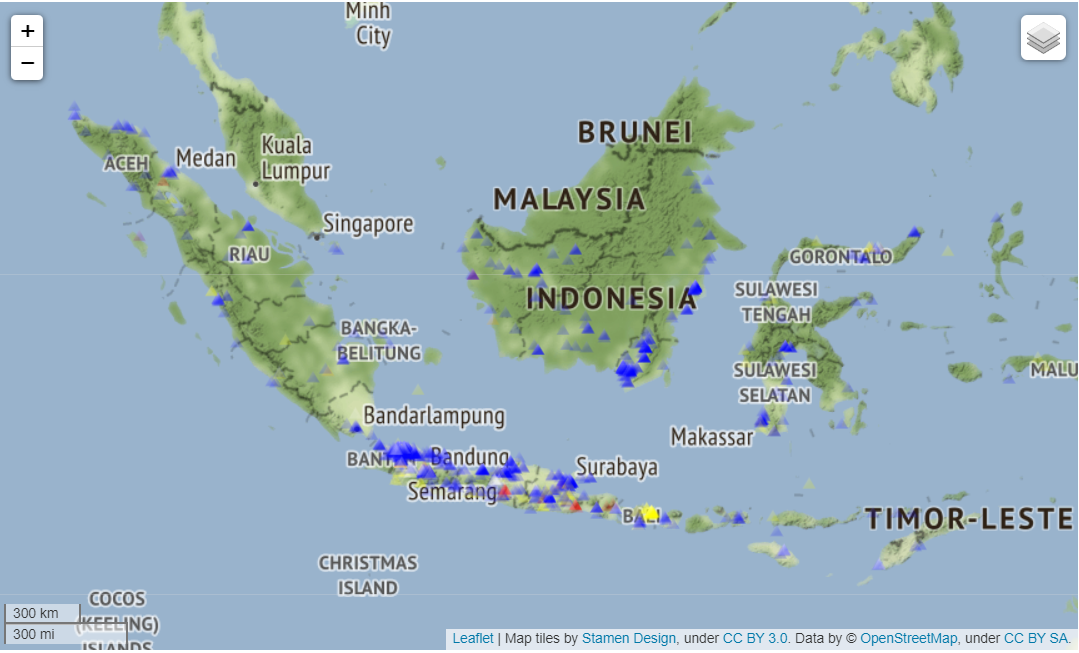
else:

color = 'black'

folium.RegularPolygonMarker([loc['lat'], loc['long']], popup = df\_news['disaster\_type'][index], fill\_color=color, color=False, number\_of\_sides=3, radius=6, bold=True, rotation=30).add\_to(map2)

folium.LayerControl().add\_to(map2)

map2



Memvisualisasikan sebaran bencana alam di Indonesia berdasarkan tipe bencana.

# Memvisualisasikan sebaran bencana alam yang terjadi di Indonesia

print(df\_news.value\_counts('disaster\_type'))#we know that flood is the common disaster that happened in Indonesia based on the data

dffin = df\_news['disaster\_type'].apply(lambda x : x.split(',')[0])

plt.figure(figsize=(8,6))

sns.countplot(y=dffin)

plt.title('Bencana alam yang terjadi di Indonesia')

plt.show()

disaster\_type

flood 4008

earthquake 131

wind 64

volcano 18

fire 12

haze 2

- 1

Menghitung dan melihat jumlah terjadinya bencana dengan tabel dan visualisasi.

# Terjadi banjir di 100 daerah berbeda pada tgl 01-01-2020

dffin1=df\_news.groupby('date')['disaster\_type'].value\_counts().head()

dffin1

#pada hari yang sama didapatkan pelaporan banjir terbanyak dalam satu hari

date disaster\_type

2020-01-01 flood 100

2020-01-02 flood 10

2020-01-03 flood 9

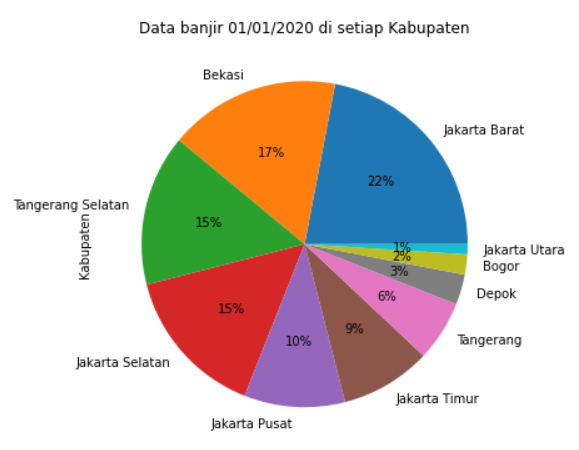
2020-01-04 flood 1

2020-01-05 flood 2

Melihat jumlah terbanyak terjadinya bencana banjir dalam satu hari.Ternyata pada tanggal 01/01/2020 pernah terjadi bencana alam banjir sebanyak 100 pelaporan kasus.

dffin2=df\_news[(df\_news['date']=='2020-01-01') & (df\_news['disaster\_type']=='flood')]

print(dffin2['Kabupaten'].value\_counts())

dffin2['Kabupaten'].value\_counts().plot(kind='pie', title='Data banjir 01/01/2020 di setiap Kabupaten', figsize=(8,6), autopct='%1.0f%%')

plt.show()

Jakarta Barat 22

Bekasi 17

Tangerang Selatan 15

Jakarta Selatan 15

Jakarta Pusat 10

Jakarta Timur 9

Tangerang 6

Depok 3

Bogor 2

Jakarta Utara 1

Melihat rincian dari 100 pelaporan bencana alam banjir pada tanggal 01/01/2020. Didapatkan pada hari itu paling banyak terjadi bencana banjir di daerah Jakarta Barat dengan total pelaporan kasus sebanyak 22 kasus.

df\_news['flood\_depth'].max() #banjir dengan kedalaman tertinggi setinggi 2 meter

200.0

Mencari besar data kedalaman tertinggi, yakni 2 meter.

print(df\_news[df\_news['flood\_depth']==200.0].value\_counts().sum()) # terdapat 14 daerah dengan kedalaman banjir paling dalam

dffin3=df\_news[(df\_news['flood\_depth']==200.0)&(df\_news['Kabupaten'])]

dffin3[['date','Desa','Kecamatan','Kabupaten','disaster\_type']].sort\_values('Kabupaten') #pada data desa gambir terjadi bencana yang sama pada tanggal yang sama di tahun yang berbeda



Menampilkan kolom tertentu pada data yang mengalami banjir dengan kedalaman 2 meter. Terdapat 14 data.

dffin4 = df\_news['Kabupaten'].value\_counts(ascending=False).head(5)

print(dffin4)

dffin4.plot(kind = 'pie', title='5 Daerah terkena bencana terbanyak', figsize=(8,6), autopct='%1.0f%%')

plt.show()

Jakarta Timur 681

Jakarta Barat 527

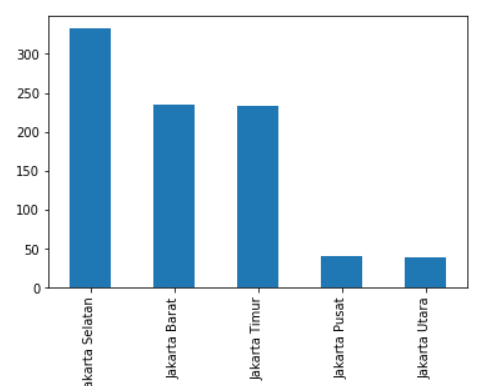
Bekasi 473

Jakarta Utara 408

Jakarta Selatan 375

Melihat daerah yang paling sering terjadi bencana Alam

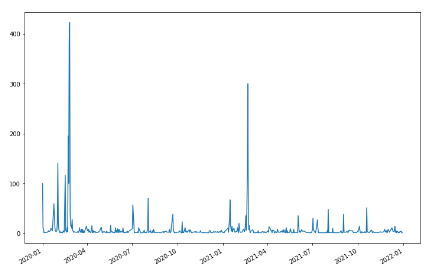
pj['Kabupaten'].value\_counts().head(5).plot(kind='bar')



Tampilan lain dari visualisasi daerah yang sering terjadi bencana alam di Indonesia.

# Menampilkan waktu terjadinya bencana

df\_news['date'].value\_counts().plot(kind='line', figsize = (12,8))



Menampilkan visualisasi dari waktu terjadinya bencana, berdasarkan tanggal.

!pip install openpyxl

with pd.ExcelWriter("D:\tan45\tan45.xlsx") as writer:

df\_news.to\_excel(writer, sheet\_name='dataset')

pb['Alamat'].value\_counts()

Perintah untuk membantu mengunduh data yang sudah di clean menjadi file.xls, untuk diolah di tableau

DAFTAR PUSTAKA

Nayu.(2020).*Airport Traffic Visualization geopandas & Plotly✈.* Diakses pada 18 Agustus 2022, dari<https://www.kaggle.com/code/nayuts/airport-traffic-visualization-geopandas-plotly>

Devakumar. (2022). *GeoPandas.* Diakses pada 18 Agustus 2022, dari<https://www.kaggle.com/code/imdevskp/geopandas>

<https://stackoverflow.com/questions/20680272/parsing-a-json-string-which-was-loaded-from-a-csv-using-pandas>

<https://stackoverflow.com/questions/32375471/pandas-convert-strings-to-time-without-date>

<https://stackoverflow.com/questions/20680272/parsing-a-json-string-which-was-loaded-from-a-csv-using-pandas>

<https://kumparan.com/berita_viral/perbedaan-waktu-pagi-siang-sore-senja-petang-malam-dini-hari-dan-subuh-1yDy7XXywXI/4>

Pankaj. (2022). Python string to datetime - strptime(). Diakses pada 18 Agustus 2022, dari

<https://www.digitalocean.com/community/tutorials/python-string-to-datetime-strptime>

Data to Fish. (2021). *How to Replace Values in Pandas DataFrame.* Diakses pada 19 Agustus 2022, dari<https://datatofish.com/replace-values-pandas-dataframe/#:~:text=Suppose%20that%20you%20want%20to,value'%2C...%5D>)

<https://stackoverflow.com/questions/32375471/pandas-convert-strings-to-time-without-date>

Abdishakur. (2018). *How to convert latitude/longtitude columns in CSV to Geometry Column using Python.* Diakses pada 19 Agustus 2022, dari <https://shakasom.medium.com/how-to-convert-latitude-longtitude-columns-in-csv-to-geometry-column-using-python-4219d2106dea>

Nisa.(2022). *Descriptive Analytics (Pengertian, Manfaat, Langkah, dan Contohnya)*. Diakses pada 18 Agustus 2022, dari <https://inmarketing.id/descriptive-analytics-adalah.html>

Retnowardhani, Astari dan Tiswa Ramdani.(2019). *Jenis - Jenis Data Analytics*. Diakses pada 18 Agustus 2022, dari <https://mmsi.binus.ac.id/2019/11/26/jenis-jenis-data-analytics/>