120450016 - Masayu Franstika
Sains Data - RB

LAPORAN AKHIR

-TUGAS BESAR-

STATISTIKA SAINS DATA

Dosen Pengampu Mata kuliah :
Mika Alvionita Sitinjak, M.Si
Riksa Meidy Karim, M.Sc

Institut Teknologi Sumatera- 2022

TABLE OF CONTENT

Dataset yang digunakan pada Tugas Besar ini adalah dataset covid_uts.csv

Ada 4 Pokok Pembahasan utama dalam Laporan ini yaitu :

1. Data Wrangling
2. Data Visualization
3. Data Processing
4. Model Implementation

DATA WRANGLING

Data wrangling adalah proses pembersihan, penataan, dan pengayaan data mentah ke dalam format yang diinginkan untuk menghasilkan pengambilan keputusan yang lebih baik dalam waktu yang lebih singkat. Berikut query yang saya gunakan :

- head() : digunakan untuk menampilkan data awal atau data teratas pada dataframe. *Default*-nya jika kita tidak memberikan argumen di dalam tanda kurung (), data yang akan ditampilkan adalah 5 baris teratas. Namun, kita juga dapat menentukan berapa baris data yang ingin ditampilkan dengan memberikan argumen berupa bilangan integer

```
df.head() # menampilkan 5 data pertama
```

	Negara	Tanggal	Varian	N_Positif
0	Egypt	2020-05-11	Alpha	0
1	Egypt	2020-05-11	Beta	0
2	Egypt	2020-05-11	Gamma	0
3	Egypt	2020-05-11	Mu	0
4	Egypt	2020-05-11	Omicron	0

- tail() ; digunakan untuk menampilkan data terbawah pada dataframe:

```
df.tail() # menampilkan 5 data terakhir
```

	Negara	Tanggal	Varian	N_Positif
1485	Malaysia	2021-12-27	Alpha	0
1486	Malaysia	2021-12-27	Beta	0
1487	Malaysia	2021-12-27	Gamma	0
1488	Malaysia	2021-12-27	Mu	0
1489	Malaysia	2021-12-27	Omicron	5

- unique() : digunakan untuk menampilkan nilai unik dari suatu kolom

```
negaras = df['Negara'].unique()
print('Jumlah Negara: ', len(negaras))
for neg in negaras:
    print('-',neg)
```

```
Jumlah Negara:  7
- Egypt
- Finland
- Germany
- Indonesia
- Italy
- Japan
- Malaysia
```

```
varians = df['Varian'].unique()
print('Jumlah Varian Covid: ' , len(varians))
for v in varians:
    print('-',v)
```

```
Jumlah Varian Covid:  5
- Alpha
- Beta
- Gamma
- Mu
- Omicron
```

DATA WRANGLING

- melt() : digunakan untuk mengembalikan kondisi data yang sudah dilakukan pivot menjadi sebelum pivot

```
pd.melt(df) #untuk mengubah kolom menjadi baris
```

	variable	value
0	Negara	Egypt
1	Negara	Egypt
2	Negara	Egypt
3	Negara	Egypt
4	Negara	Egypt
...	...	...
5955	N_Positif	0
5956	N_Positif	0
5957	N_Positif	0
5958	N_Positif	0
5959	N_Positif	5

5960 rows × 2 columns

- rename() : digunakan untuk mengganti nama dari kolom yang kita inginkan sebagai contoh disini saya me rename kolom varian menjadi variant

```
df.rename(columns={'Varian': 'Variant'}, inplace=True) #untuk mengganti nama kolom
df
```

	Negara	Tanggal	Variant	N_Positif
0	Egypt	2020-05-11	Alpha	0
1	Egypt	2020-05-11	Beta	0
2	Egypt	2020-05-11	Gamma	0
3	Egypt	2020-05-11	Mu	0
4	Egypt	2020-05-11	Omicron	0
...	...	...	...	...
1485	Malaysia	2021-12-27	Alpha	0
1486	Malaysia	2021-12-27	Beta	0
1487	Malaysia	2021-12-27	Gamma	0
1488	Malaysia	2021-12-27	Mu	0
1489	Malaysia	2021-12-27	Omicron	5

- sort_values() : digunakan untuk mencari data terbesar hingga terkecil

```
df.sort_values('Negara', ascending=False) #untuk mencara data dari terbesar hingga ke terkecil
```

	Negara	Tanggal	Variant	N_Positif
1489	Malaysia	2021-12-27	Omicron	5
1332	Malaysia	2020-10-26	Gamma	0
1356	Malaysia	2021-01-04	Beta	0
1355	Malaysia	2021-01-04	Alpha	1
1354	Malaysia	2020-12-21	Omicron	0
...	...	...	...	...
135	Egypt	2021-05-31	Alpha	3
134	Egypt	2021-05-17	Omicron	0
133	Egypt	2021-05-17	Mu	0
132	Egypt	2021-05-17	Gamma	0
0	Egypt	2020-05-11	Alpha	0

DATA WRANGLING

- describe() : digunakan untuk menampilkan deskriptif statistik data. Hanya kolom yang bertipe numerik yang akan ditampilkan statistiknya:

```
df.describe() # menghitung statistik
```

	N_Positif
count	1490.000000
mean	133.230872
std	1067.331772
min	0.000000
25%	0.000000
50%	0.000000
75%	0.000000
max	18317.000000

- info() : digunakan untuk menampilkan informasi detail tentang dataframe, seperti jumlah baris data, nama-nama kolom berserta jumlah data dan tipe datanya, dan sebagainya

```
df.info() # untuk mengaetahui informasi data
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1490 entries, 0 to 1489
Data columns (total 4 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Negara     1490 non-null   object
 1   Tanggal    1490 non-null   object
 2   Variant    1490 non-null   object
 3   N_Positif  1490 non-null   int64
dtypes: int64(1), object(3)
memory usage: 46.7+ KB
```

- groupby() : untuk melakukan perhitungan kelompok berdasarkan nilai unik sesuai kolom yang dipilih atau dapat juga digunakan untuk menggabungkan kolom berdasarkan nama kolom yang sesuai dengan query

```
df2=df.groupby(['Negara', 'Variant']).sum() #menggabungkan data berdasarkan kolom yang sama
df2
```

Negara	Variant	N_Positif
Egypt	Alpha	29
	Beta	0
	Gamma	0
	Mu	0
	Omicron	1
Finland	Alpha	6800
	Beta	1213
	Gamma	19
	Mu	5
	Omicron	0
Germany	Alpha	104138
	Beta	2303
	Gamma	858
	Mu	17
	Omicron	2270
Indonesia	Alpha	81
	Beta	22
	Gamma	0
	Mu	0
	Omicron	130
Italy	Alpha	26877
	Beta	116
	Gamma	2488
	Mu	83
	Omicron	526
Japan	Alpha	49841
	Beta	101
	Gamma	120
	Mu	3
	Omicron	150
Malaysia	Alpha	33
	Beta	273
	Gamma	0
	Mu	0
	Omicron	17

DATA WRANGLING

- reset_index() : digunakan untuk me-reset indeks yang telah ter-set dan menjadikan indeksnya default, yaitu berupa bilangan integer yang dimulai dari 0

```
df3 = df2.query("Variant=='Omicron' | Variant=='Alpha'| Variant=='Beta' ") #mencari data yang sesuai dengan kriteria
df3.reset_index(inplace=True)
df3
```

	Negara	Variant	N_Positif
0	Egypt	Alpha	29
1	Egypt	Beta	0
2	Egypt	Omicron	1
3	Finland	Alpha	6800
4	Finland	Beta	1213
5	Finland	Omicron	0
6	Germany	Alpha	104138
7	Germany	Beta	2303
8	Germany	Omicron	2270
9	Indonesia	Alpha	81
10	Indonesia	Beta	22
11	Indonesia	Omicron	130
12	Italy	Alpha	26877
13	Italy	Beta	116
14	Italy	Omicron	526
15	Japan	Alpha	49841
16	Japan	Beta	101
17	Japan	Omicron	150
18	Malaysia	Alpha	33
19	Malaysia	Beta	273
20	Malaysia	Omicron	17

DATA VISUALIZATION

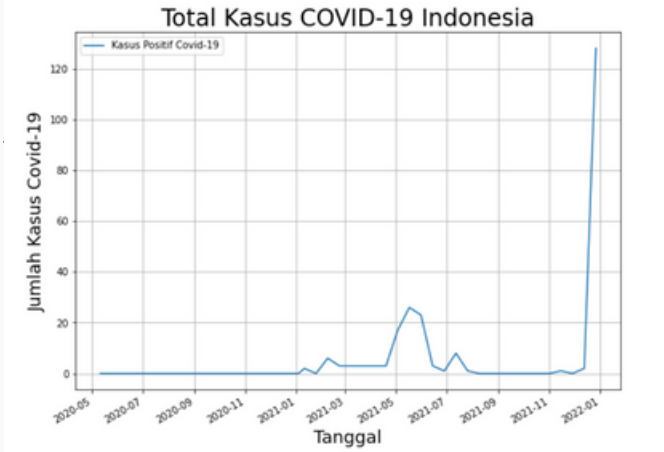
Data Visualization adalah tampilan berupa grafis atau visual dari informasi dan data. Dengan kata lain, data visualization mengubah kumpulan data menjadi hal lebih sederhana untuk ditampilkan. Dengan menggunakan elemen visual tersebut, pembaca akan lebih mudah memahami tren, outliers, dan pola dalam suatu data.

Disini saya akan meng plot kan total kasus covid - 19 di Indonesia dengan menggunkan query sebagai berikut :

```
df_idn = df.groupby(['Negara','Tanggal']).sum() #menggabungkan data berdasarkan kolom yang sama
df_idn = df_idn.loc[['Indonesia']].reset_index() #mencari data yang sesuai dengan kriteria
df_idn['Tanggal'] = pd.to_datetime(df_idn['Tanggal']) #mengubah data tanggal menjadi format datetime
df_idn = df_idn.set_index('Tanggal') #mengubah kolom tanggal menjadi index
```

```
plt.figure( figsize=(10,7.5) ) #untuk mengatur ukuran plot
ax = plt.gca()
df_idn.plot(ax=ax)
ax.grid()
ax.set_xlabel('Tanggal',fontsize=18)
ax.set_ylabel('Jumlah Kasus Covid-19',fontsize=18)
ax.legend(['Kasus Positif Covid-19'])
ax.set_title('Total Kasus COVID-19 Indonesia',fontsize=24)
plt.show()
```

Dari query diatas didapatkan hasil plot seperti gambar dibawah :



Dari grafik diatas dapat kita lihat bahwa jumlah kasus covid - 19 di Indonesia pada bulan Mei tahun 2020 tercatat 0 kasus dan kasus yang tertinggi berada pada bulan November 2021. Untuk mengecek kebenaran dari grafik diatas maka dapat kita lakukan pemanggilan df_idn dan didapatkan hasil sebagai berikut :

df_idn

Tanggal	Negara	N_Positif
2020-05-11	Indonesia	0
2020-05-25	Indonesia	0
2020-06-08	Indonesia	0
2020-06-22	Indonesia	0
2020-07-06	Indonesia	0
2020-07-20	Indonesia	0
2020-08-03	Indonesia	0
2020-08-17	Indonesia	0
2020-08-31	Indonesia	0
2020-09-14	Indonesia	0
2020-09-28	Indonesia	0
2020-10-12	Indonesia	0
2020-10-26	Indonesia	0
2020-11-09	Indonesia	0
2020-11-23	Indonesia	0
2020-12-07	Indonesia	0
2020-12-21	Indonesia	0
2021-01-04	Indonesia	0
2021-01-11	Indonesia	2
2021-01-25	Indonesia	0
2021-02-08	Indonesia	6
2021-02-22	Indonesia	3
2021-03-08	Indonesia	3
2021-03-22	Indonesia	3
2021-04-05	Indonesia	3
2021-04-19	Indonesia	3
2021-05-03	Indonesia	17
2021-05-17	Indonesia	26
2021-05-31	Indonesia	23
2021-06-14	Indonesia	3
2021-06-28	Indonesia	1
2021-07-12	Indonesia	8
2021-07-26	Indonesia	1
2021-08-09	Indonesia	0
2021-08-23	Indonesia	0
2021-09-06	Indonesia	0
2021-09-20	Indonesia	0
2021-10-04	Indonesia	0
2021-10-18	Indonesia	0
2021-11-01	Indonesia	0
2021-11-15	Indonesia	1
2021-11-29	Indonesia	0
2021-12-13	Indonesia	2
2021-12-27	Indonesia	128

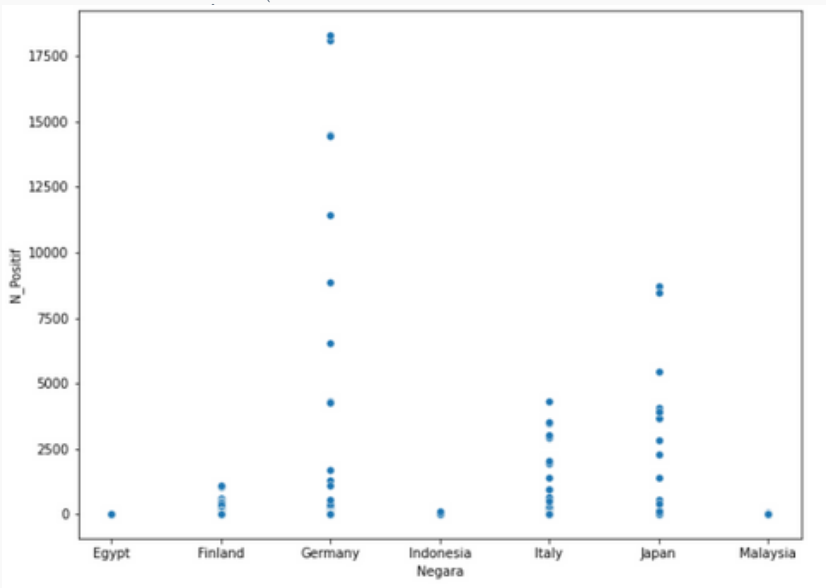
DATA VISUALIZATION

Dari hasil pengecekan data diatas terbukti bahwa data kasus covid di Indonesia terbesar berada pada bulan Desember tahun 2021 yaitu sebanyak 128 kasus.

Berikutnya disini saya akan menampilkan diagram scatterplot dari N_Positif setiap Negara, dengan menggunkan query sebagai berikut :

```
sns.scatterplot(data=df, x=df['Negara'], y=df['N_Positif']); #menampilkan grafik scatterplot
```

Dari query diatas didapatkan hasil diagram seperti gambar dibawah :



Dari hasil diagram diatas dapat kita lihat secara visual bahwa jumlah kasus Covid-19 terbesar adalah di negara Germany yaitu lebih dari 17500 kasus Covid-19 kemudian disusul dengan Japan, Italy, Finland, Indonesia, Malaysia, dan yang terendah adalah Egypt. Untuk mengetahui kebenaran dari diagram diatas dapat dilakukan data processing untuk mengetahui jumlah sebenarnya.

DATA PROCESSING

Data Processing merupakan proses pengumpulan data dan dikonversi menjadi sebuah informasi yang bermanfaat dan dapat digunakan.

Disini saya akan melakukan pemrosesan data untuk mengetahui jumlah kasus Covid-19 dari setiap Negara dengan Variant Covid-19 yaitu Omicron, Alpha, Beta dengan menggunakan query sebagai berikut :

```
processing = df3[['Negara']]
negara = processing['Negara'].unique()

omicrons = df3[df3['Variant']=='Omicron']
o = omicrons[['N_Positif']].values.flatten()
alpha = df3[df3['Variant']=='Alpha']
a= alpha[['N_Positif']].values.flatten()
beta = df3[df3['Variant']=='Beta']
b = beta[['N_Positif']].values.flatten()

datas = {
'Negara': negara,
'Omicron': o,
'Alpha':a,
'Beta':b,
}
df_new = pd.DataFrame(datas)
df_new
```

Dan didapatkan hasil sebagai beriku :

	Negara	Omicron	Alpha	Beta
0	Egypt	1	29	0
1	Finland	0	6800	1213
2	Germany	2270	104138	2303
3	Indonesia	130	81	22
4	Italy	526	26877	116
5	Japan	150	49841	101
6	Malaysia	17	33	273

Dari data diatas dapat kita ketahui bahwa jumlah kasus tertinggi Covid-19 adalah Gemany, Japan, Italy, Finland, Indonesia, Malaysia dan Egypt. Artinya bahwa diagram pada pembahasan sebelumnya terbukti kebenarannya.

Kemudian saya melakukan perhitngan Similarity dengan mendefinisikan similarity sebagai jarak r yang dihitung menggunakan Euclidean Distance Pada Negara Finland, Indonesia, Italy, Japan, dan Malaysia. Dan membandingkan Manakan Negara yang paling similar dengan Japan dengan query sebagai berikut :

```
def euclid(x,y):

    return np.linalg.norm(x-y)
lneg = len(temps)
d_matrix = [ [ round(euclid(temps[i],temps[j]),2 ) for j in range(lneg) ] for i in range(lneg)]
```

```
negara1 = df4[['Negara']].values.flatten()
df5 = pd.DataFrame( d_matrix,columns = negara1, index = negara1 )
df5
```

DATA PROCESSING

Dari query diatas didapat kan hasil berupa matrix 5 negara sebagai berikut :

	Finland	Indonesia	Italy	Japan	Malaysia
Finland	0.00	6824.98	20113.83	43055.62	6832.00
Indonesia	6824.98	0.00	26799.09	49760.07	279.42
Italy	20113.83	26799.09	0.00	22967.08	26849.28
Japan	43055.62	49760.07	22967.08	0.00	49808.47
Malaysia	6832.00	279.42	26849.28	49808.47	0.00

Setelah didapatkan hasil diatas , maka kita akan mencari manakah Negara yang paling Similar degan Japan dengan menggunkan query berikut :

```
d1 = df5.loc['Japan']
d2 = df5.columns.values
def similarity(d):
    return 1/(d+0.001)
d3 = map( lambda x,y:[ similarity(x) ] + [y] ,d1,d2)
d4 = list(d3)
d4.sort(key=lambda x:x[0],reverse=True)
d4 = d4[1:]
# Urutan Similarity:
d4
```

kemudian didapatkan data yang paling similar dengan Japan adalah Italy, Finland, Indonesia dan malaysia. Dapat dilihat pada gambar dibawah ini :

```
[[4.3540578796234485e-05, 'Italy'],
 [2.3225771148440757e-05, 'Finland'],
 [2.0096434347933307e-05, 'Indonesia'],
 [2.0076906195333723e-05, 'Malaysia']]
```

MODEL IMPLEMENTASI

Metode Regresi Logistik : Suatu metode analisis statistika untuk mendeskripsikan hubungan antara variabel terikat yang memiliki dua kategori atau lebih dengan satu atau lebih peubah bebas berskala kategori atau kontinu.

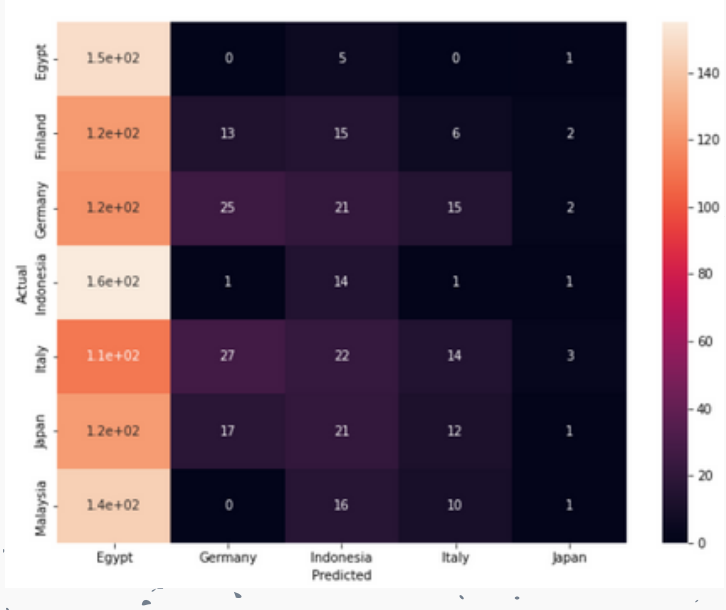
```
X = df[['N_Positif']]
y = df['Negara']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.8, random_state = 0)

logistic_regression = LogisticRegression(solver='liblinear')
logistic_regression.fit(X_train, y_train)
y_pred = logistic_regression.predict(X_test)

confusion_matrix = pd.crosstab(y_test, y_pred, rownames=['Actual'], colnames=['Predicted'])
sns.heatmap(confusion_matrix, annot=True)
```

Pada Model Implementasi Metode Regresi Logistik saya mendefinisikan x sebagai N_Positif dan y sebagai Negara dan saya mengambil secara acak y_test sebesar 80% . Kemudian saya plot kan kemudian didapatkan hasil heatmap sebagai berikut :



Selanjutnya saya melakukan pemanggilan x_test dengan cara print(X_test) dan didapatkan hasil sebagai berikut :

```
      N_Positif
9             0
354           0
1227          1
907           0
575          31
...         ...
832           0
951           7
1144          0
1037          0
1196          3

[1192 rows x 1 columns]
```

MODEL IMPLEMENTASI

Selanjutnya saya melakukan pemanggilan y_pred dengan cara print(y_test) dan didapatkan hasil sebagai berikut :

```
['Egypt' 'Egypt' 'Indonesia' ... 'Egypt' 'Egypt' 'Indonesia']
```

Kemudian saya melakukan perhitungan akurasi dengan matrics.accuracy_score dan didapatkan hasil akurasi sebesar : 0.17030201342281878 dengan syntax sebagai beriku :

```
print('Akurasi: ', metrics.accuracy_score(y_test, y_pred))
```

Langkah selanjutnya mencari classification_report dengan y_teset dan predictions kemudian mencari accuracy_score dengan menggunkan y_test dan predictions dan didapatkan hasil sebagai berikut :

```
model=LogisticRegression()
model.fit(X_train,y_train)
predictions=model.predict(X_test)
```

```
print(classification_report(y_test,predictions))
print(accuracy_score(y_test,predictions))
```

```
              precision    recall  f1-score   support

       Egypt       0.16      0.96      0.28       155
     Finland       0.00      0.00      0.00       159
     Germany       0.29      0.11      0.16       183
   Indonesia       0.14      0.08      0.10       172
       Italy       0.26      0.11      0.15       177
       Japan       0.23      0.03      0.06       175
    Malaysia       0.00      0.00      0.00       171

    accuracy                           0.17      1192
   macro avg       0.16      0.18      0.11      1192
weighted avg       0.16      0.17      0.11      1192

Akurasi :  0.174496644295302
```

Accuracy Merupakan rasio prediksi Benar (positif dan negatif) dengan keseluruhan data dari data diatas didapatkan accuracy sebesar 0.174496644295302. Precission Merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan hasil yang diprediksi positf Dari hasil data diatas dapat kita lihat bahwa Precison yang paling besar pada Negara Germany sebesar 0,29 artinya bahwa terbukti Kasus Covid-19 terbesar pada dataset yang kita miliki adalah Germany. Recall (Sensitifitas) Merupakan rasio prediksi benar positif dibandingkan dengan keseluruhan data yang benar positif Kemudian Recall terbesar dengan hasil 0,96 pada Negara Egypt. F1 Score merupakan perbandingan rata-rata presisi dan recall yang dibobotkan dan F1-Score yang paling besar Adalah Negara Egypt dengan hasil 0.28 artinya bahwa Prediksi Covid-19 keseluruhan dibandingkan dengan kasus Covid-19 sebenarnya pada data tersebut benar bahwa Egypt memiliki kasus terkecil dibandingkan Germany dan Negara lainnya.