

Tugas

Data dan PRE-PROCESSING

1. Carilah data bebas

https://drive.google.com/file/d/1KUILYhe_ITM_jg5NQfPwXqiPd8oW2r8V/view?usp=sharing

2. Link Google Collab

https://colab.research.google.com/drive/1XyracCw224b1AReaPgmRhLctCkuE_16a?usp=sharing

A. Matlab

- 1) Mengimport dan menampilkan data dari file CSV

```
% Membaca data
data = readtable('day.csv')
```

Output :

data =													
731x16 table													
instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual
1	2011-01-01	1	0	1	0	6	0	2	0.34417	0.36362	0.80583	0.16045	331
2	2011-01-02	1	0	1	0	0	0	2	0.36348	0.35374	0.69609	0.24854	131
3	2011-01-03	1	0	1	0	1	1	1	0.19636	0.1894	0.43727	0.24831	120
4	2011-01-04	1	0	1	0	2	1	1	0.2	0.21212	0.59044	0.1603	108
5	2011-01-05	1	0	1	0	3	1	1	0.22696	0.22927	0.43696	0.1869	82
:	:	:	:	:	:	:	:	:	:	:	:	:	:
727	2012-12-27	1	1	12	0	4	1	2	0.25417	0.22664	0.65292	0.35013	247
728	2012-12-28	1	1	12	0	5	1	2	0.25333	0.25505	0.59	0.15547	644
729	2012-12-29	1	1	12	0	6	0	2	0.25333	0.2424	0.75292	0.12438	159
730	2012-12-30	1	1	12	0	0	0	1	0.25583	0.2317	0.48333	0.35075	364
731	2012-12-31	1	1	12	0	1	1	2	0.21583	0.22349	0.5775	0.15485	439

- 2) Menginput 3 record(baris) dari data tersebut dan menyimpannya dalam variable baru yang bernama variable **data_day**

```
% Menginput 3 record dari data dan menyimpannya dalam variabel baru.
data_day = data(1:3, :);
disp("data day = ")
disp(data_day)
```

1:3 berarti Anda ingin mengambil baris 1 hingga 3 (sebagai rentang baris), dan **:** digunakan untuk memilih semua kolom dalam tabel.

Output :

data day =													
instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual
1	2011-01-01	1	0	1	0	6	0	2	0.34417	0.36362	0.80583	0.16045	331
2	2011-01-02	1	0	1	0	0	0	2	0.36348	0.35374	0.69609	0.24854	131
3	2011-01-03	1	0	1	0	1	1	1	0.19636	0.1894	0.43727	0.24831	120

- 3) Memilih beberapa atribut dari data tersebut

Disini saya mengambil 3 atribut yakni casual, registered, dan cnt pada kolom ke-14, 15, dan 16

```
% Mengambil 3 atribut dari data
selected_atribut = data_day(:, [14, 15, 16]);
disp(selected_atribut);
```

Output :

casual	registered	cnt
331	654	985
131	670	801
120	1229	1349

4) Melakukan PreProcessing dari data dengan atribut yang dipilih

- Deteksi Outlier

```
% Deteksi outlier
outlier = isoutlier(selected_atribut)
```

Output :

```
outlier =

3x3 logical array

     1     0     0
     0     0     0
     0     1     0
```

Dari hasil deteksi di atas, terlihat bahwa menghasilkan matriks logika yang berisi nilai **1**. Ini artinya terdapat data yang dianggap sebagai outlier

- Penanganan Outlier

Karena data terdapat outlier, maka data tersebut harus kita bersihkan terlebih dahulu. Ada beberapa cara untuk menangani outlier.

Cara pertama : dengan mengganti data yang outlier dengan 0

```
% Penanganan Outlier
% Mengganti outlier dengan 0
data_outlier_cleaned = filloutliers(selected_atribut, 0)
```

Mengapa outlier diganti dengan 0?

1. Dapat membantu membersihkan data dari gangguan yang tidak diinginkan karena nilai 0 tidak akan memiliki dampak besar pada statistik

2. Nilai 0 akan mungkin menjadikan data lebih stabil dan akurat dalam kasus-kasus tertentu.

Output :

```
data_outlier_cleaned =  
  
3x3 table  
  
  casual  registered  cnt  
-----  
    0      654      985  
  131      670      801  
  120       0     1349
```

Output menunjukkan bahwa data outlier yang terdapat pada kolom casual dan registered telah diganti menjadi 0

Cara kedua : Mengganti outlier dengan data terdekat

```
% Mengganti outlier dengan data terdekat  
data_outlier_cleaned2 = filloutliers(selected_atribut, 'nearest', 'DataVariables', {'casual', 'registered'})
```

Output :

```
data_outlier_cleaned2 =  
  
3x3 table  
  
  casual  registered  cnt  
-----  
    131      654      985  
    131      670      801  
    120      670     1349
```

Output menunjukkan bahwa data outlier yang terdapat pada kolom casual dan registered telah diganti menjadi nilai yang sama dengan nilai yang didekatnya.

Cara ketiga : Menghapus data yang memuat outlier

```
% Menghilangkan data outlier  
data_outlier_cleaned3 = rmoutliers(selected_atribut)
```

Output :

```
data_outlier_cleaned3 =
```

casual	registered	cnt
131	670	801

- Deteksi Missing Value

Melakukan deteksi missing value pada data yang outliernya telah diganti

```
% Deteksi Missing Value
missing1 = ismissing(data_outlier_cleaned)
```

Output :

```
missing1 =
```

3x3 logical array		
0	0	0
0	0	0
0	0	0

Melakukan deteksi missing value pada data yang outlier nya telah dihapus

```
missing2 = ismissing(data_outlier_cleaned3)
```

Output :

```
missing2 =
```

1x3 logical array		
0	0	0

Dari kedua deteksi missing value yang dilakukan, output menunjukkan tidak adanya missing value value pada data sehingga tidak perlu dilakukan langkah-langkah penanganan terhadap missing value

5) Normalisasi

```
% NORMALISASI
% Normalisasi pada data data yang sudah oke
normalisasi = normalize(data_outlier_cleaned, 'zscore')
```

Output :

```
normalisasi =  
  
3x3 table  
  
   casual  registered  cnt  
-----  
-1.1514    0.5563   -0.21514  
0.65138    0.59815   -0.87492  
0.50001   -1.1544    1.0901
```

Output di atas menunjukkan bahwa data yang telah dinormalisasi sehingga nilainya berada dalam rentang 0 hingga 1. Rentang dalam normalisasi ini bisa diubah sesuai dengan kebutuhan.

B. PYTHON

- 1) Mengimport berbagai library yang diperlukan

```
[129] import numpy as np  
      import pandas as pd  
      import matplotlib.pyplot as plt  
      import seaborn as sns  
      from scipy import stats
```

- 2) Membaca data menggunakan library Pandas dan menginput 3 record (mengambil 3 baris data)

```
[ ] data_day = pd.read_csv("day.csv")  
    df = pd.DataFrame(data_day)  
  
    #Inputkan 3 record dari data t  
    three_record = df.head(3)  
    print(three_record)
```

Output :

	instant	dteday	season	yr	mnth	holiday	weekday	workingday	weathersit	temp	atemp	hum	windspeed	casual	registered	cnt
0	1	2011-01-01	1	0	1	0	6	0	2	0.344167	0.363625	0.805833	0.160446	331	654	985
1	2	2011-01-02	1	0	1	0	0	0	2	0.363478	0.353739	0.696087	0.248539	131	670	801
2	3	2011-01-03	1	0	1	0	1	1	1	0.196364	0.189405	0.437273	0.248309	120	1229	1349

- 3) Memilih beberapa atribut dari data tersebut

Disini saya mengambil 3 atribut yakni casual, registered, dan cnt.

```
multiple_atribut = df.loc[:, ['casual', 'registered', 'cnt']]  
print(multiple_atribut)
```

Output :

	casual	registered	cnt
0	331	654	985
1	131	670	801
2	120	1229	1349

4) Lakukan PreProcessing dari data dengan atribut yang dipilih

- Deteksi Outlier

Sebelum melakukan deteksi outlier, kita definisikan kembali kolom-kolom yang ingin dideteksi outliernya

```
[ ] # Pilih kolom-kolom yang ingin Anda deteksi outlier-nya
kolom_target = ['casual', 'registered', 'cnt']
```

Fungsi untuk mendeteksi outlier pada kolom yang telah dipilih menggunakan metode Zscore

```
[133] # Definisikan fungsi untuk mendeteksi outlier
def detect_outliers(data, kolom):
    outliers = []
    threshold = 3
    mean = np.mean(data[kolom])
    std = np.std(data[kolom])
    z_scores = np.abs(stats.zscore(data[kolom]))
    kolom_outliers = data[z_scores > threshold]
    return kolom_outliers

# Lakukan deteksi outlier untuk setiap kolom yang dipilih
outlier_datapoints = {}
for kolom in kolom_target:
    outlier_datapoints[kolom] = detect_outliers(data_day, kolom)

# Tampilkan hasil deteksi outlier untuk setiap kolom
for kolom, outlier_data in outlier_datapoints.items():
    print(f"Outliers in {kolom}:")
    print(outlier_data)
    print()
```

threshold = 3: artinya nilai ambang batas yang digunakan dalam metode Z-score. Nilai defaultnya adalah 3, yang berarti titik data dianggap sebagai outlier jika Z-score-nya melebihi 3 atau lebih dari 3 deviasi standar dari rata-rata

mean = np.mean(data[kolom]) dan **std = np.std(data[kolom])**: untuk menghitung rata-rata (mean) dan deviasi standar (standard deviation) dari kolom yang sedang dianalisis.

z_scores = np.abs(stats.zscore(data[kolom])): menghitung Z-score untuk setiap titik data dalam kolom dengan menggunakan rumus Z-score standar. standar.

kolom_outliers = data[z_scores > threshold]: Di sini, Anda membuat variabel **kolom_outliers** yang berisi semua baris data dari kolom yang memiliki Z-score di atas ambang batas **threshold**.

Output :

```
Outliers in casual:
  instant  dteday  season  yr  mnth  holiday  weekday  workingday  \
184      185  2011-07-04    3    7      1         1         0
441      442  2012-03-17    1    3      0         6         0
462      463  2012-04-07    2    4      0         6         0
504      505  2012-05-19    2    5      0         6         0
512      513  2012-05-27    2    5      0         0         0
532      533  2012-06-16    2    6      0         6         0
623      624  2012-09-15    3    9      0         6         0
644      645  2012-10-06    4   10      0         6         0

  weathersit    temp    atemp    hum  windspeed  casual  registered  \
184         2  0.726667  0.665417  0.637917  0.081479   3065       2978
441         2  0.514167  0.505046  0.755833  0.110704   3155       4681
462         1  0.437500  0.426129  0.254167  0.274871   3252       3605
504         1  0.600000  0.566908  0.456250  0.083975   3410       4884
512         1  0.690000  0.641425  0.697083  0.215171   3283       3308
532         1  0.631667  0.594708  0.504167  0.166667   2963       4739
623         1  0.608333  0.585867  0.501667  0.247521   3160       5554
644         1  0.554167  0.538521  0.664167  0.268025   3031       4934

  cnt
184  6043
441  7836
462  6857
504  8294
512  6591
532  7702
623  8714
644  7965

Outliers in registered:
Empty DataFrame
Columns: [instant, dteday, season, yr, mnth, holiday, weekday, workingday, weathersit, temp, atemp, hum, windspeed, casual, registered, cnt]
Index: []

Outliers in cnt:
Empty DataFrame
Columns: [instant, dteday, season, yr, mnth, holiday, weekday, workingday, weathersit, temp, atemp, hum, windspeed, casual, registered, cnt]
Index: []
```

Output di atas menunjukkan bahwa pada kolom casual, terdapat beberapa data yang dianggap sebagai outlier. Data outlier ini dicantumkan dalam daftar yang mencakup kolom-kolom lainnya yang juga memiliki data outlier yang sesuai.

Sedangkan pada kolom registered dan cnt, tidak ada data yang dianggap sebagai outlier. Oleh karena itu, outputnya adalah "Empty DataFrame," yang berarti tidak ada data yang melebihi ambang batas outlier.

- **Penanganan Outlier**

Melakukan penanganan outlier dengan menghapus outlier berdasarkan nilai Z-score. Hasil data yang telah dihapus outliernya akan disimpan dalam DataFrame baru bernama **new_data_day**

```
[134] # Hitung z-score untuk kolom yang dipilih
      z_scores = stats.zscore(data_day[kolom_target])

      # Ambil nilai absolut dari z-score
      abs_z_scores = np.abs(z_scores)

      # Buat kondisi untuk menghapus outlier
      filtered_entries = (abs_z_scores < 3).all(axis=1)

      # Buat DataFrame baru tanpa outlier
      new_data_day = data_day[filtered_entries]

      # Tampilkan DataFrame baru tanpa outlier
      print(new_data_day)
```

abs_z_scores = np.abs(z_scores) : Mengambil nilai absolut dari setiap nilai Z-score agar nilai positif dan negatif Z-score tetap dianggap sebagai outlier jika mereka berjarak dari rata-rata dalam arah yang berlawanan.

filtered_entries = (abs_z_scores < 3).all(axis=1): Kondisi ini menghasilkan larik Boolean di mana setiap baris dataset akan bernilai **True** jika semua nilai Z-score dalam baris tersebut kurang dari 3 (ambang batas outlier yang telah ditetapkan), dan **False** jika ada setidaknya satu nilai Z-score yang lebih besar atau sama dengan 3 dalam baris tersebut. Ini dilakukan dengan menggunakan metode **.all(axis=1)** untuk memeriksa setiap baris.

Output :

```

0      instant  dteday  season  yr  mnth  holiday  weekday  workingday  \
1      2      2.0      1      0      1      0      0      0      0
2      3      3.0      1      0      1      0      1      1      1
3      4      4.0      1      0      1      0      2      1      1
4      5      5.0      1      0      1      0      3      1      1
..      ...      ...      ...  ..      ...      ...      ...      ...
726     727    362.0      1      1     12      0      4      1
727     728    363.0      1      1     12      0      5      1
728     729    364.0      1      1     12      0      6      0
729     730    365.0      1      1     12      0      0      0
730     731    366.0      1      1     12      0      1      1

      weathersit      temp      atemp      hum  windspeed  casual  registered  \
0      2  0.344167  0.363625  0.805833  0.160446      331      654
1      2  0.363478  0.353739  0.696087  0.248539      131      670
2      1  0.196364  0.189405  0.437273  0.248309      120     1229
3      1  0.200000  0.212122  0.590435  0.160296      108     1454
4      1  0.226957  0.229270  0.436957  0.186900       82     1518
..      ...      ...      ...      ...      ...      ...      ...
726     2  0.254167  0.226642  0.652917  0.350133      247     1867
727     2  0.253333  0.255046  0.590000  0.155471      644     2451
728     2  0.253333  0.242400  0.752917  0.124383      159     1182
729     1  0.255833  0.231700  0.483333  0.350754      364     1432
730     2  0.215833  0.223487  0.577500  0.154846      439     2290

      cnt
0      985
1      801
2     1349
3     1562
4     1600
..      ...
726    2114
727    3095
728    1341
729    1796
730    2729
```


- Menampilkan kembali atribut/kolom yang telah dipilih dan telah dibersihkan outlier nya

Karena dari output code sebelumnya menampilkan seluruh kolom/atribut data, sedangkan yang kita deteksi dan bersihkan outliernya hanya beberapa atribut, maka agar terlihat lebih jelas, kita bisa mengambil kolom/atribut yang telah dipilih sebelumnya saja.

```
[14] new_data_columns = new_data_day.loc[:, ['casual', 'registered', 'cnt']]
      print(new_data_columns)
```

Output :

```

      casual  registered  cnt
0         331         654  985
1         131         670  801
2         120        1229 1349
3         108        1454 1562
4          82        1518 1600
..      ...      ...  ...
726        247        1867 2114
727        644        2451 3095
728        159        1182 1341
729        364        1432 1796
730        439        2290 2729

[723 rows x 3 columns]
```

- Deteksi missing value

Mendeteksi missing value dari beberapa atribut/kolom yang telah dipilih dan telah bersih dari outlier

```
[ ] new_data_day.isna().sum()
```

Output :

```
casual      0
registered  0
cnt         0
dtype: int64
```

Output di atas menunjukkan bahwa tidak ditemukan adanya missing value sehingga tidak perlu dilakukan langkah-langkah penanganan terhadap missing value

5) Normalisasikan data – data tersebut

- Mengimport library yang diperlukan untuk melakukan normalisasi

```
[136] from sklearn import preprocessing
```

Melakukan normalisasi menggunakan Min-Max Scalling. Ini akan mengubah data sehingga nilainya berada dalam rentang tertentu, biasanya antara 0 dan 1.

```
min_max_scaler = preprocessing.MinMaxScaler()  
np_scaled = min_max_scaler.fit_transform(new_data_columns)  
df_normalized = pd.DataFrame(np_scaled)  
  
print("\nData hasil normalisasi:\n", df_normalized)
```

Output :

```
Data hasil normalisasi:  
      0      1      2  
0  0.115317  0.091539  0.112856  
1  0.045216  0.093849  0.091293  
2  0.041360  0.174560  0.155514  
3  0.037154  0.207046  0.180476  
4  0.028041  0.216286  0.184929  
..      ...      ...      ...  
718  0.085875  0.266676  0.245166  
719  0.225026  0.350996  0.360131  
720  0.055030  0.167774  0.154576  
721  0.126884  0.203869  0.207899  
722  0.153172  0.327751  0.317239  
  
[723 rows x 3 columns]
```

Output di atas menunjukkan bahwa dataset ini telah diubah sedemikian rupa sehingga semua nilai dalam dataset berada dalam rentang antara 0 dan 1.

Setiap kolom dalam dataset yang telah dinormalisasi memiliki nilai dalam rentang tersebut, yang berarti nilai-nilai tersebut sekarang berada pada skala yang serupa dan tidak memiliki variabilitas besar dalam rentang nilai.