Hands-On

Hands-On ini digunakan pada kegiatan Microcredential Associate Data Scientist 2021

Pertemuan 9

Pertemuan 9 (sembilan) pada Microcredential Associate Data Scientist 2021 menyampaikan materi mengenai Mengkonstruksi Data

Pada Tugas Mandiri Pertemuan 9

silakan Anda kerjakan Latihan 1 s/d 10. Output yang anda lihat merupakan panduan yang dapat Anda ikuti dalam penulisan code :)

Latihan (1)

Melakukan import library yang dibutuhkan

In [49]:
```
# import library pandas
import pandas as pd

# Import library scipy
import scipy as sp

# Import library winsorize dari scipy
from scipy.stats.mstats import winsorize

# Import library trima dari scipy
from scipy.stats.mstats import trima

# Import library RandomSampleImputer dari feature engine imputation
from feature_engine.imputation import RandomSampleImputer

# import library StandardScaler dari sklearn
from sklearn.preprocessing import StandardScaler
```

Latihan (2)

Menghitung nilai null pada dataset :

1. Load dataset Iris_Unclean
2. Tampilkan dataset
3. Hitung jumlah nilai null pada dataset

In [ ]:
```
# load dataset Iris_Unclean
df = pd.read_csv('Iris_unclean.csv')
```

In [ ]:
```
# tampilkan dataset
df
```

In [ ]:
```
# hitung jumlah nilai null pada dataset
df.isna().sum()
```

Latihan (3)

Melakukan handle missing value dengan Imputasi Mean:

1. Load dataset Iris_Unclean
2. Ambil 10 data teratas "SepalLengthCm", kemudian tampilkan
3. Mengganti missing value Imputasi dengan mean, kemudian masukkan ke variable
4. Tampilkan 10 data teratas "SepalLengthCm" setelah handle missing value dengan Imputasi mean()

In [ ]:
```
# load dataset Iris_Unclean
df = pd.read_csv('Iris_unclean.csv')
```

In [ ]:
```
# ambil 10 data teratas SepalLengthCm, kemudian tampilkan

df = df['SepalLengthCm'][0:10]
df
```

In [ ]:
```
# mengganti missing value dengan mean(), kemudian masukkan ke variabel

df = df.fillna(df.mean())
```

In [ ]:
```
# tampilkan 10 data teratas SepalLengthCm setelah handle missing value dengan imputasi mean

df
```

Latihan (4)

Melakukan handle missing value dengan nilai suka-suka (Arbitrary):

1. Load dataset Iris_Unclean
2. Ambil 10 data teratas "SepalLengthCm", kemudian tampilkan
3. Mengganti missing value dengan imputasi nilai suka-suka (Arbitrary), kemudian masukkan ke variable
4. Tampilkan 10 data teratas "SepalLengthCm" setelah handle missing value dengan nilai suka-suka

In [ ]:
```
# load dataset Iris_Unclean
df = pd.read_csv('Iris_unclean.csv')
```

In [ ]:
```
# ambil 10 data teratas SepalLengthCm, kemudian tampilkan

df = df['SepalLengthCm'][0:10]
df
```

In [ ]:
```
# melakukan imputasi nilai suka-suka (Arbitrary), masukkan ke dalam variabel

df = df.fillna(99)
```

In [ ]:
```
# tampilkan 10 data teratas SepalLengthCm setelah handle missing value dengan nilai suka-suka (arbitary)

df
```

Latihan (5)

Melakukan handle missing value dengan frequent category / modus:

1. Load dataset Iris_Unclean
2. Ambil 10 data teratas "SepalLengthCm", kemudian tampilkan
3. Mengganti missing value dengan frequent category / modus
4. Tampilkan hasil imputasi "SepalLengthCm" setelah handle dengan frequent category / modus

In [2]:
```
# load dataset Iris_Unclean
df = pd.read_csv('Iris_unclean.csv')
```

In [3]:
```
# tampilkan 10 data teratas kolom SepalLengthCm

df = df['SepalLengthCm'][0:10]
df
```

Out[3]:
```
0    NaN
1    4.9
2    4.7
3    4.6
4    5.0
5    5.4
6    NaN
7    5.0
8    4.4
9    4.9
Name: SepalLengthCm, dtype: float64
```

In [4]:
```
df = pd.DataFrame(df)
df
```

Out[4]:

	SepalLengthCm
0	NaN
1	4.9
2	4.7
3	4.6
4	5.0
5	5.4
6	NaN
7	5.0
8	4.4
9	4.9

In [5]:
```
# Import SimpleImputer dari sklearn.impute
from sklearn.impute import SimpleImputer

# Mengatasi missing value dengan frequent category / modus
imp = SimpleImputer(strategy='most_frequent')
```

In [6]:
```
# Tampilkan hasil imputasi "SepalLengthCm"

imp.fit_transform(df)
```

Out[6]:
```
array([[4.9],
       [4.9],
       [4.7],
       [4.6],
       [5. ],
       [5.4],
       [4.9],
       [5. ],
       [4.4],
       [4.9]])
```

Latihan (6)

Melakukan handle missing value dengan Imputasi Random Sample:

1. Load dataset Iris_Unclean
2. Tampilkan 10 data teratas
3. Membuat imputer random sample dengan random state = 5
4. Cocokan imputer ke data
5. Ubah data dengan imputer masukkan ke dalam variable
6. Tampilkan hasil imputasi data "SepalLengthCm"

In [7]:
```
# load dataset Iris_Unclean
df = pd.read_csv('Iris_unclean.csv')
```

In [8]:
```
# tampilkan 10 data teratas kolom SepalLengthCm

df = df['SepalLengthCm'][0:10]
df
```

Out[8]:
```
0    NaN
1    4.9
2    4.7
3    4.6
4    5.0
5    5.4
6    NaN
7    5.0
8    4.4
9    4.9
Name: SepalLengthCm, dtype: float64
```

In [11]:
```
df = pd.DataFrame(df)
df
```

Out[11]:

	SepalLengthCm
0	NaN
1	4.9
2	4.7
3	4.6
4	5.0
5	5.4
6	NaN
7	5.0
8	4.4
9	4.9

In [12]:
```
# Membuat imputer random sample dengan random state = 5
imputer = RandomSampleImputer(random_state = 5)

# Cocokan imputer ke data
imputer.fit(df)

# Ubah data dengan imputer masukkan ke dalam variable
test_t = imputer.transform(df)
```

In [13]:
```
# Tampilkan data hasil imputasi data "SepalLengthCm"
test_t
```

Out[13]:

	SepalLengthCm
0	5.0
1	4.9
2	4.7
3	4.6
4	5.0
5	5.4
6	4.4
7	5.0
8	4.4
9	4.9

Latihan (7)

Melakukan Winsorizing

1. Import library winsorize dari scipy
2. Load data Iris_AfterClean
3. Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel datan tampilkan
4. Winsorize data dengan batas nilai terendah 10% dan batas nilai tinggi 20%
5. Tampilkan hasil winsorize

In [14]:
```
# Import library scipy
import scipy as sp
```

In [25]:
```
# Load data Iris_AfterClean
data = pd.read_csv('Iris_AfterClean.csv')

# Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel data tampilkan
a = data['SepalLengthCm'][0:10]
a
```

Out[25]:
```
0    4.6
1    5.0
2    5.4
3    4.9
4    5.4
5    4.8
6    4.8
7    4.3
8    5.8
9    5.4
Name: SepalLengthCm, dtype: float64
```

In [26]:
```
# Winsorize data dengan batas nilai terendah 10% dan batas nilai tinggi 20%
wins = winsorize(a, limits=[0.1, 0.2])

# Tampilkan hasil winsorize
print(wins)
```

```
[4.6 5.  5.4 4.9 5.4 4.8 4.8 4.6 5.4 5.4]
```

Latihan (8)

Melakukan Trimming

1. Import library trima dari scopy
2. Load data Iris_AfterClean
3. Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel datan tampilkan
4. Trimming data dengan batas nilai terendah 2 dan batas nilai tinggi 5
5. Tampilkan hasil trimming

In [31]:
```
# Load data Iris_AfterClean
data = pd.read_csv('Iris_AfterClean.csv')

# Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel data tampilkan
a = data['SepalLengthCm'][0:10]
a
```

Out[31]:
```
0    4.6
1    5.0
2    5.4
3    4.9
4    5.4
5    4.8
6    4.8
7    4.3
8    5.8
9    5.4
Name: SepalLengthCm, dtype: float64
```

In [28]:
```
# Trimming data dengan batas nilai terendah 2 dan batas nilai tinggi 5

trims = trima(a, limits=(2,5))

# Tampilkan hasil trimming

print(trims)
```

```
[4.6 5.0 -- 4.9 -- 4.8 4.8 4.3 -- --]
```

Latihan (9)

Melakukan Scaling: Normalisasi

1. Load data Iris_AfterClean
2. Ambil 10 data teratas SepalLengthCm dan SepalWidthCm
3. Menghitung mean data
4. Menghitung max - min pada data
5. Menerapkan transformasi ke data
6. Tampilkan hasil scalling

In [43]:
```
# Load data Iris_AfterClean
data = pd.read_csv('Iris_AfterClean.csv')

# Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel data tampilkan
a = pd.DataFrame(data[['SepalLengthCm', 'SepalWidthCm']][0:10])
a
```

Out[43]:

	SepalLengthCm	SepalWidthCm
0	4.6	3.1
1	5.0	3.6
2	5.4	3.9
3	4.9	3.1
4	5.4	3.7
5	4.8	3.4
6	4.8	3.0
7	4.3	3.0
8	5.8	4.0
9	5.4	3.9

In [44]:
```
#Menghitung mean
means = a.mean(axis = 0)

# menghitung max - min
max_min = a.max(axis = 0) - a.min(axis = 0)

# menerapkan transformasi ke data
train_scaled = (a - means) / max_min
```

In [45]:
```
# Tampilkan hasil scalling

train_scaled
```

Out[45]:

	SepalLengthCm	SepalWidthCm
0	-0.293333	-0.37
1	-0.026667	0.13
2	0.240000	0.43
3	-0.093333	-0.37
4	0.240000	0.23
5	-0.160000	-0.07
6	-0.160000	-0.47
7	-0.493333	-0.47
8	0.506667	0.53
9	0.240000	0.43

Latihan (10)

Melakukan Scaling: Standardisasi

1. Load data Iris_AfterClean
2. Ambil 10 data teratas SepalLengthCm dan SepalWidthCm
2. Import library StandardScaler dari sklearn
3. Membuat objek scaler
4. Sesuaikan scaler dengan data
5. Mengubah data
6. Tampilkan hasil scalling dengan standarisasi

In [46]:
```
# Load data Iris_AfterClean
data = pd.read_csv('Iris_AfterClean.csv')

# Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel data tampilkan
a = pd.DataFrame(data[['SepalLengthCm', 'SepalWidthCm']][0:10])
a
```

Out[46]:

	SepalLengthCm	SepalWidthCm
0	4.6	3.1
1	5.0	3.6
2	5.4	3.9
3	4.9	3.1
4	5.4	3.7
5	4.8	3.4
6	4.8	3.0
7	4.3	3.0
8	5.8	4.0
9	5.4	3.9

In [47]:
```
from sklearn.preprocessing import StandardScaler

#Buat objek scaler
scaler = StandardScaler()

#Sesuaikan scaler dengan data
scaler.fit(a)

#Mengubah data kereta
train_scaled = scaler.transform(a)
```

In [48]:
```
# Tampilkan hasil
train_scaled
```

Out[48]:
```
array([[-1.02464215, -0.97469723],
       [-0.09314929,  0.34246119],
       [ 0.83834358,  1.13275625],
       [-0.3260225 , -0.97469723],
       [ 0.83834358,  0.60589288],
       [-0.55889572, -0.18440218],
       [-0.55889572, -1.23812892],
       [-1.7232618 , -1.23812892],
       [ 1.76983644,  1.39618793],
       [ 0.83834358,  1.13275625]])
```

In [ ]:
```

```