



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 [1]: # import library pandas
import pandas as pd
# Import library scipy
import scipy
# Import library winsorize dari scipy
from scipy.stats.mstats import winsorize
# Import library trima dari scipy
from scipy.stats.mstats import trim
# 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 [2]: # load dataset Iris_Unclean
df_unclean = pd.read_csv('Iris_unclean.csv')
```

```
In [3]: # tampilkan dataset
df_unclean
```

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
0	NaN	3.5	1.4	0.2	Iris-setosa
1	4.9	2000.0	1.4	0.2	Iris-setosa
2	4.7	3.2	-1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...

	SepalLengthCm	SepalWidthCm	PetalLengthCm	PetalWidthCm	Species
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

In [4]:

```
# hitung jumlah nilai null pada dataset
df_unclean.isnull().sum()
```

Out[4]:

SepalLengthCm	2
SepalWidthCm	0
PetalLengthCm	0
PetalWidthCm	0
Species	0
dtype: int64	

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 [5]:

```
# load dataset Iris_Unclean
df = pd.read_csv("Iris_unclean.csv")
```

In [6]:

```
# ambil 10 data teratas SepalLengthCm, kemudian tampilkan
df = df['SepalLengthCm'][0:10]
df
```

Out[6]:

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 [7]: # mengganti missing value dengan mean(), kemudian masukkan ke variabel  
df = df.fillna(df.mean())
```

```
In [8]: # tampilkan 10 data teratas SepalLengthCm setelah handle missing value dengan  
df.head(10)
```

```
Out[8]: 0    4.8625  
1    4.9000  
2    4.7000  
3    4.6000  
4    5.0000  
5    5.4000  
6    4.8625  
7    5.0000  
8    4.4000  
9    4.9000  
Name: SepalLengthCm, dtype: float64
```

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

```
In [10]: # ambil 10 data teratas SepalLengthCm, kemudian tampilkan  
df = df["SepalLengthCm"][0:10]  
df
```

```
Out[10]: 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]: # melakukan imputasi nilai suka-suka (Arbitrary), masukkan ke dalam variabel  
df = df.fillna(99)
```



```
In [12]: # tampilkan 10 data teratas SepalLengthCm setelah handle missing value dengan  
df.head(10)
```



```
Out[12]: 0    99.0  
1    4.9  
2    4.7  
3    4.6  
4    5.0  
5    5.4  
6    99.0  
7    5.0  
8    4.4  
9    4.9  
Name: SepalLengthCm, dtype: float64
```

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

```
In [14]: # tampilkan 10 data teratas kolom SepalLengthCm  
df = df_unclean['SepalLengthCm'][0:10]  
df
```

```
Out[14]: 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 [15]: # Import SimpleImputer dari sklearn.impute  
  
from sklearn.impute import SimpleImputer  
  
# Mengatasi missing value dengan frequent category / modus  
  
imp = SimpleImputer(strategy='most_frequent')
```

```
In [16]: # Tampilkan hasil imputasi "SepalLengthCm"  
  
imp.fit_transform(df_unclean[['SepalLengthCm']])
```

```
Out[16]: array([[5. ,  
   4.9],  
  [4.7],  
  [4.6],  
  [5. ],  
  [5.4],  
  [5. ],  
  [5. ],  
  [4.4],  
  [4.9],  
  [5.4],  
  [4.8],  
  [4.8],  
  [4.3],  
  [5.8],  
  [5.7],  
  [5.4],  
  [5.1],  
  [5.7],  
  [5.1],  
  [5.4],  
  [5.1],  
  [4.6],  
  [5.1],  
  [4.8],  
  [5. ],  
  [5. ],  
  [5.2],  
  [5.2],  
  [4.7],  
  [4.8],  
  [5.4],  
  [5.2],  
  [5.5],  
  [4.9],  
  [5. ],  
  [5.5],  
  [4.9],  
  [4.4],  
  [5.1],  
  [5. ],  
  [4.5],  
  [4.4],  
  [5. ],  
  [5.1],  
  [4.8],  
  [5.1],  
  [4.6],
```

```
[5.3],  
[5. ],  
[7. ],  
[6.4],  
[6.9],  
[5.5],  
[6.5],  
[5.7],  
[6.3],  
[4.9],  
[6.6],  
[5.2],  
[5. ],  
[5.9],  
[6. ],  
[6.1],  
[5.6],  
[6.7],  
[5.6],  
[5.8],  
[6.2],  
[5.6],  
[5.9],  
[6.1],  
[6.3],  
[6.1],  
[6.4],  
[6.6],  
[6.8],  
[6.7],  
[6. ],  
[5.7],  
[5.5],  
[5.5],  
[5.8],  
[6. ],  
[5.4],  
[6. ],  
[6.7],  
[6.3],  
[5.6],  
[5.5],  
[5.5],  
[6.1],  
[5.8],  
[5. ],  
[5.6],  
[5.7],  
[5.7],  
[6.2],  
[5.1],  
[5.7],  
[6.3],  
[5.8],  
[7.1],  
[6.3],  
[6.5],  
[7.6],  
[4.9],  
[7.3],  
[6.7],  
[7.2],  
[6.5],  
[6.4],  
[6.8],
```

```
[5.7],  
[5.8],  
[6.4],  
[6.5],  
[7.7],  
[7.7],  
[6. ],  
[6.9],  
[5.6],  
[7.7],  
[6.3],  
[6.7],  
[7.2],  
[6.2],  
[6.1],  
[6.4],  
[7.2],  
[7.4],  
[7.9],  
[6.4],  
[6.3],  
[6.1],  
[7.7],  
[6.3],  
[6.4],  
[6. ],  
[6.9],  
[6.7],  
[6.9],  
[5.8],  
[6.8],  
[6.7],  
[6.7],  
[6.3],  
[6.5],  
[6.2].
```

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

```
In [18]: # tampilkan 10 data teratas SepalLengthCm  
  
df = df_unclean['SepalLengthCm'][0:10]  
df
```

```
Out[18]: 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 [19]: # Membuat imputer random sample dengan random state = 5  
  
imputer = RandomSampleImputer(random_state=5)  
  
# Cocokan imputer ke data  
  
imputer.fit(df_unclean)  
  
# Ubah data dengan imputer masukkan ke dalam variable  
  
test_t = imputer.transform(df_unclean)
```

```
In [20]: # Tampilkan data hasil imputasi data "SepalLengthCm"  
  
test_t['SepalLengthCm'].head(10)
```

```
Out[20]: 0      5.8  
1      4.9  
2      4.7  
3      4.6  
4      5.0  
5      5.4  
6      6.9  
7      5.0  
8      4.4  
9      4.9  
Name: SepalLengthCm, dtype: float64
```

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 data dan tampilkan
4. Winsorize data dengan batas nilai terendah 10% dan batas nilai tinggi 20%
5. Tampilkan hasil winsorize

```
In [21]: # Import library scipy
```

```
from scipy.stats.mstats import winsorize
```

```
In [22]: # Load data Iris_AfterClean
```

```
data = pd.read_csv('Iris_AfterClean.csv')
```

```
# Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel a
```

```
a = data['SepalLengthCm'][:10]
```

```
a
```

```
Out[22]: 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 [23]: # 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 scipy
2. Load data Iris_AfterClean
3. Ambil 10 data teratas "SepalLengthCm", kemudian masukkan ke dalam variabel data dan tampilkan
4. Trimming data dengan batas nilai terendah 2 dan batas nilai tinggi 5
5. Tampilkan hasil trimming

```
In [24]: # Import library trima dari scipy
```

```
from scipy.stats.mstats import trima
```

In [25]:

```
# Load data Iris_AfterClean
data = pd.read_csv('Iris_AfterClean.csv')

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

```
# 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 scaling

In [27]:

```
# Load data Iris_AfterClean

data = pd.read_csv('Iris_AfterClean.csv')

# Ambil 10 data teratas SepalLengthCm dan SepalWidthCm

data = data[['SepalLengthCm', 'SepalWidthCm']].head(10)
data
```

Out [27]:

	SepalLengthCm	SepalWidthCm
0	4.6	3.1
1	5.0	3.6

	SepalLengthCm	SepalWidthCm
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

In [28]:

```
# Menghitung mean
means = data.mean()

# menghitung max - min
max_min = data.max() - data.min()

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

In [29]:

```
# Tampilkan hasil scaling
# Tampilkan hasil scaling

train_scaled
```

Out [29]:

	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

In [30]:

```
# Load data Iris_AfterClean

data = pd.read_csv('Iris_AfterClean.csv')

# Ambil 10 data teratas SepalLengthCm dan SepalWidthCm

data = data[['SepalLengthCm', 'SepalWidthCm']].head(10)
data
```

Out [30]:

	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 [31]:

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

# Buat objek scaler
scaler = StandardScaler()

# Sesuaikan scaler dengan data
scaler.fit(data)

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

In [32]:

```
# Tampilkan hasil

train_scaled
```

Out [32]: 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],  
[-0.3260225 , -1.23812891])
```

In []: