

Praktikum Data Preprocessing

UTS DATA MINING

Ulima Inas Shabrina(2110181048)

# UTS - #Assignment

## Klasifikasi untuk Imputasi

1. **dataset**  $\leftarrow$  **titanic.csv**
2. **data**  $\leftarrow$  ambil dataset kolom fitur (**Sex**, **Age**, **Pclass**, **Fare**, **Survived**).
3. **train\_data**  $\leftarrow$  ambil fitur (**Sex**, **Pclass**, **Fare**, **Survived**) pada **data** yang **Age**  $\neq$  null
4. **train\_label**  $\leftarrow$  ambil fitur (**Age**) pada **data** yang **Age**  $\neq$  null
5. **test\_data**  $\leftarrow$  ambil fitur (**Sex**, **Pclass**, **Fare**, **Survived**) pada **data** yang **Age** = null
6. **train\_data**  $\leftarrow$  lakukan normalisasi pada **train\_data** dengan Min-Max 0-1 (catat nilai min dan max setiap atribut)
7. **test\_data**  $\leftarrow$  lakukan normalisasi pada **test\_data** dengan Min-Max 0-1 (dengan nilai min dan max setiap atribut pada Langkah 6)
8. **class\_result**  $\leftarrow$  Lakukan klasifikasi **test\_data** terhadap **train\_data** dengan 3-NN (output memakai class pada **train\_label**)
9. **data** (**Age**)  $\leftarrow$  lakukan pengisian missing values pada **data** yang **Age** = null dengan nilai **class\_result**
10. **test\_dataset**  $\leftarrow$  **titanic\_test.csv**
11. **train\_data**  $\leftarrow$  ambil fitur (**Sex**, **Age**, **Pclass**, **Fare**) dari **data**
12. **train\_label**  $\leftarrow$  ambil fitur (**Survived**) dari **data**
13. **test\_data**  $\leftarrow$  ambil **test\_dataset** kolom fitur (**Sex**, **Age**, **Pclass**, **Fare**). Hilangkan baris data yang terdapat missing values
14. **test\_label**  $\leftarrow$  **titanic\_testlabel.csv** (urutan sesuai **test\_data** no.13)
15. **train\_data**  $\leftarrow$  lakukan normalisasi pada **train\_data** dengan Min-Max 0-1 (catat nilai min dan max setiap atribut)
16. **test\_data**  $\leftarrow$  lakukan normalisasi pada **test\_data** dengan Min-Max 0-1 (dengan nilai min dan max setiap atribut pada Langkah 15)
17. **class\_result**  $\leftarrow$  Lakukan klasifikasi **test\_data** terhadap **train\_data** dengan 3-NN (output memakai class pada **train\_label**)
18. **error**  $\leftarrow$  Bandingkan hasil klasifikasi **class\_result** dengan **test\_label**. Jika tidak sama berarti error. Hitunglah jumlah error dari seluruh **class\_result**
19. **error\_ratio**  $\leftarrow$  **error** dibagi jumlah **test\_data**, dikali 100 (%)

```
In [1]: #1 dataset <- titanic.csv
# Pada langkah ini dilakukan pembacaan isi dari file titanic.csv dengan menggunakan library pandas dan disimpan
# pada variabel dataset. Total baris pada dataset berjumlah 891 dan total kolomnya 12

import pandas as pd

dataset = pd.read_csv('titanic.csv')
dataset
```

Out[1]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

```
In [2]: #2 data <- ambil dataset kolom fitur (Sex, Age, Pclass, Fare, Survived)
# Langkah ini dilakukan pengambilan kolom fitur Sex, Age, Pclass, Fare, Survived pada variabel dataset yang
# disimpan pada variabel data. Pada langkah ini juga dilakukan perubahan nilai value pada kolom Sex untuk
# value male diubah menjadi 1 dan female menjadi 0. Total baris pada data berjumlah 891 dan total kolomnya 5.

data = pd.DataFrame(dataset, columns=['Sex', 'Age', 'Pclass', 'Fare', 'Survived']).replace(['male', 'female'], [1,0])
data
```

Out [2]:

	Sex	Age	Pclass	Fare	Survived
0	1	22.0	3	7.2500	0
1	0	38.0	1	71.2833	1
2	0	26.0	3	7.9250	1
3	0	35.0	1	53.1000	1
4	1	35.0	3	8.0500	0
...	...	...	...	...	...
886	1	27.0	2	13.0000	0
887	0	19.0	1	30.0000	1
888	0	NaN	3	23.4500	0
889	1	26.0	1	30.0000	1
890	1	32.0	3	7.7500	0

891 rows × 5 columns

```
In [3]: #3 train_data <- ambil fitur (Sex, Pclass, Fare, Survived) pada data yang Age≠null
# mengambil fitur kolom Sex, Pclass, Fare, Survived pada variabel data dimana dilakukan filter pada setiap baris
# untuk kolom Age tidak boleh null dan disimpan pada variabel train_data

train_data = data[data['Age'].notnull()].loc[:, data.columns != 'Age']
train_data
```

Out[3]:

	Sex	Pclass	Fare	Survived
0	1	3	7.2500	0
1	0	1	71.2833	1
2	0	3	7.9250	1
3	0	1	53.1000	1
4	1	3	8.0500	0
...	...	...	...	...
885	0	3	29.1250	0
886	1	2	13.0000	0
887	0	1	30.0000	1
889	1	1	30.0000	1
890	1	3	7.7500	0

714 rows × 4 columns

```
In [4]: #4 train_label <- ambil fitur (Age) pada data yang Age≠null
# mengambil fitur kolom Age pada variabel data dimana dilakukan filter pada setiap baris untuk
# kolom Age tidak boleh null dan disimpan pada variabel train_label

train_label = data[data['Age'].notnull()].loc[:, data.columns == 'Age']
train_label
```

Out[4]:

	Age
0	22.0
1	38.0
2	26.0
3	35.0
4	35.0
...	...
885	39.0
886	27.0
887	19.0
889	26.0
890	32.0

714 rows × 1 columns

```
In [5]: #5 test_data <- ambil fitur (Sex, Pclass, Fare, Survived) pada data yang Age=null
# mengambil fitur kolom selain Age pada variabel data dimana dilakukan filter pada setiap baris
# untuk kolom Age yang bernilai null dan disimpan pada variabel test_data

test_data = data[data['Age'].isnull()].loc[:, data.columns != 'Age']
test_data
```

Out[5]:

	Sex	Pclass	Fare	Survived
5	1	3	8.4583	0
17	1	2	13.0000	1
19	0	3	7.2250	1
26	1	3	7.2250	0
28	0	3	7.8792	1
...	...	...	...	...
859	1	3	7.2292	0
863	0	3	69.5500	0
868	1	3	9.5000	0
878	1	3	7.8958	0
888	0	3	23.4500	0

177 rows × 4 columns

In [6]: *#6 train\_data lakukan normalisasi pada train\_data dengan Min-Max 0-1 (catat nilai min dan max setiap atribut)*  
*# Min dan Max dari setiap kolom pada train\_data dapat diambil dengan menggunakan method agg().*

```
min_max_train_data = train_data.agg([min, max])
train_data = ((train_data - train_data.min()) * (1-0) / ((train_data.max() - train_data.min()) + 0))
print(min_max_train_data)
print('\n-----\n')
print(train_data)
```

	Sex	Pclass	Fare	Survived
min	0	1	0.0000	0
max	1	3	512.3292	1

-----

	Sex	Pclass	Fare	Survived
0	1.0	1.0	0.014151	0.0
1	0.0	0.0	0.139136	1.0
2	0.0	1.0	0.015469	1.0
3	0.0	0.0	0.103644	1.0
4	1.0	1.0	0.015713	0.0
..	...	...	...	...
885	0.0	1.0	0.056848	0.0
886	1.0	0.5	0.025374	0.0
887	0.0	0.0	0.058556	1.0
889	1.0	0.0	0.058556	1.0
890	1.0	1.0	0.015127	0.0

[714 rows x 4 columns]



```
In [7]: #7 test_data <- lakukan normalisasi pada test_data dengan Min-Max 0-1
# variabel new_min_max_train_data merupakan nilai min dan max pada setiap kolom pada train_data setelah
# dilakukan normalisasi pada langkah 6. nilai new_min_max_train_data pada setiap kolom adalah 1 dan 0.
# Nilai 1 dan 0 digunakan sebagai nilai max dan min untuk normalisasi pada test_data

new_min_max_train_data = train_data.agg([min, max])
test_data = (test_data - train_data.min())*(1-0)/(train_data.max() - train_data.min()) + 0
print(new_min_max_train_data)
print('\n-----\n')
print(test_data)
```

	Sex	Pclass	Fare	Survived
min	0.0	0.0	0.0	0.0
max	1.0	1.0	1.0	1.0

```
-----
```

	Sex	Pclass	Fare	Survived
5	1.0	3.0	8.4583	0.0
17	1.0	2.0	13.0000	1.0
19	0.0	3.0	7.2250	1.0
26	1.0	3.0	7.2250	0.0
28	0.0	3.0	7.8792	1.0
..	...	...	...	...
859	1.0	3.0	7.2292	0.0
863	0.0	3.0	69.5500	0.0
868	1.0	3.0	9.5000	0.0
878	1.0	3.0	7.8958	0.0
888	0.0	3.0	23.4500	0.0

[177 rows x 4 columns]

```
In [8]: #8 class_result <- Lakukan klasifikasi test_data terhadap train_data dengan 3-NN (output memakai class pada train_label)
# Klasifikasi dilakukan dengan membuat model dari train_data dengan output class pada train_label.
# Hal tersebut dilakukan dengan menggunakan method fit(). Hasil klasifikasi dapat diketahui dengan
# menggunakan method predict() yang disimpan pada variabel class_result
```

```
from sklearn.neighbors import KNeighborsClassifier
```

```
kNN = KNeighborsClassifier(n_neighbors = 3, weights = 'distance')
kNN.fit(train_data, train_label.astype('int').values.ravel())
class_result = kNN.predict(test_data)
```

```
print(class_result)
```

```
[35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35
 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35
 35 35 35 35 35 35 19 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35
 35 35 35 35 35 35 35 35 35 35 19 35 35 35 35 35 35 35 35 35 19 35 35
 35 19 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35 35
 35 35 35 35 35 19 35 35 35 35 35 35 35 35 35 35 35 35 19 35 35 35 35
 35 35 35 35 35 35 35 35]
```

```
In [9]: #9 data (Age) <- lakukan pengisian missing values pada data yang Age=null dengan nilai class_result
```

```
index = data[['Age']].isnull().any(1).to_numpy().nonzero()[0]
for i in range(len(class_result)) :
    data.loc[index[i], 'Age'] = class_result[i]

print(index, '\n-----\n', data)
```

```
[ 5  17  19  26  28  29  31  32  36  42  45  46  47  48  55  64  65  76
 77  82  87  95 101 107 109 121 126 128 140 154 158 159 166 168 176 180
181 185 186 196 198 201 214 223 229 235 240 241 250 256 260 264 270 274
277 284 295 298 300 301 303 304 306 324 330 334 335 347 351 354 358 359
364 367 368 375 384 388 409 410 411 413 415 420 425 428 431 444 451 454
457 459 464 466 468 470 475 481 485 490 495 497 502 507 511 517 522 524
527 531 533 538 547 552 557 560 563 564 568 573 578 584 589 593 596 598
601 602 611 612 613 629 633 639 643 648 650 653 656 667 669 674 680 692
697 709 711 718 727 732 738 739 740 760 766 768 773 776 778 783 790 792
793 815 825 826 828 832 837 839 846 849 859 863 868 878 888]
```

```
-----
      Sex  Age  Pclass   Fare  Survived
0      1  22.0      3    7.2500         0
1      0  38.0      1   71.2833         1
2      0  26.0      3    7.9250         1
3      0  35.0      1   53.1000         1
4      1  35.0      3    8.0500         0
..  ...  ...  ...  ...  ...
886    1  27.0      2   13.0000         0
887    0  19.0      1   30.0000         1
888    0  35.0      3   23.4500         0
889    1  26.0      1   30.0000         1
890    1  32.0      3    7.7500         0
```

```
[891 rows x 5 columns]
```

```
In [10]: #10 test_dataset <- titanic_test.csv

test_dataset = pd. read_csv('titanic_test.csv')
test_dataset
```

Out[10]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...
413	1305	3	Spector, Mr. Woolf	male	NaN	0	0	A.5. 3236	8.0500	NaN	S
414	1306	1	Oliva y Ocana, Dona. Fermina	female	39.0	0	0	PC 17758	108.9000	C105	C
415	1307	3	Saether, Mr. Simon Sivertsen	male	38.5	0	0	SOTON/O.Q. 3101262	7.2500	NaN	S
416	1308	3	Ware, Mr. Frederick	male	NaN	0	0	359309	8.0500	NaN	S
417	1309	3	Peter, Master. Michael J	male	NaN	1	1	2668	22.3583	NaN	C

418 rows × 11 columns

```
In [11]: #11 train_data <- ambil fitur (Sex, Age, Pclass, Fare) dari data
```

```
train_data = data[['Sex', 'Age', 'Pclass', 'Fare']]  
train_data
```

Out[11]:

	Sex	Age	Pclass	Fare
0	1	22.0	3	7.2500
1	0	38.0	1	71.2833
2	0	26.0	3	7.9250
3	0	35.0	1	53.1000
4	1	35.0	3	8.0500
...	...	...	...	...
886	1	27.0	2	13.0000
887	0	19.0	1	30.0000
888	0	35.0	3	23.4500
889	1	26.0	1	30.0000
890	1	32.0	3	7.7500

891 rows × 4 columns

```
In [12]: #12 train_label <- ambil_fitur (Survived) dari data
```

```
train_label = data[['Survived']]  
train_label
```

Out[12]:

Survived	
0	0
1	1
2	1
3	1
4	0
...	...
886	0
887	1
888	0
889	1
890	0

891 rows × 1 columns

```
In [13]: #13 test_data <- ambil test_dataset kolom fitur (Sex, Age, Pclass, Fare). Hilangkan baris data yang  
# terdapat missing values
```

```
test_data = pd.DataFrame(test_dataset, columns=['Sex', 'Pclass', 'Age', 'Fare']).replace(['male', 'female'], [1,0])  
remove_index = test_data.isnull().any(1).to_numpy().nonzero()[0]  
test_data = test_data.dropna()  
print(remove_index)  
print('\n-----\n')  
print(test_data)
```

```
[ 10  22  29  33  36  39  41  47  54  58  65  76  83  84  85  88  91  93  
102 107 108 111 116 121 124 127 132 133 146 148 151 152 160 163 168 170  
173 183 188 191 199 200 205 211 216 219 225 227 233 243 244 249 255 256  
265 266 267 268 271 273 274 282 286 288 289 290 292 297 301 304 312 332  
339 342 344 357 358 365 366 380 382 384 408 410 413 416 417]
```

```
-----  
  
      Sex  Pclass   Age   Fare  
0         1         3  34.5   7.8292  
1         0         3  47.0   7.0000  
2         1         2  62.0   9.6875  
3         1         3  27.0   8.6625  
4         0         3  22.0  12.2875  
..      ...      ...   ...   ...  
409        0         3   3.0  13.7750  
411        0         1  37.0  90.0000  
412        0         3  28.0   7.7750  
414        0         1  39.0 108.9000  
415        1         3  38.5   7.2500
```

```
[331 rows x 4 columns]
```

```
In [14]: #14 test_label <- titanic_testlabel.csv (urutan sesuai test_data no.13)

test_label = pd.read_csv('titanic_testlabel.csv')
test_label = pd.DataFrame(test_label, columns=['Survived']).drop(remove_index)
test_label
```

Out[14]:

Survived	
0	0
1	1
2	0
3	0
4	1
...	...
409	1
411	1
412	1
414	1
415	0

331 rows × 1 columns



In [15]: *#15 train\_data <- lakukan normalisasi pada train\_data dengan Min-Max 0-1 (catat nilai min dan max setiap atribut)*

```
min_max_train_data = train_data.agg([min, max])
train_data = ((train_data - train_data.min()) * (1-0) / ((train_data.max() - train_data.min()) + 0))
print(min_max_train_data)
print('\n-----\n')
print(train_data)
```

	Sex	Age	Pclass	Fare
min	0	0.42	1	0.0000
max	1	80.00	3	512.3292

-----

	Sex	Age	Pclass	Fare
0	1.0	0.271174	1.0	0.014151
1	0.0	0.472229	0.0	0.139136
2	0.0	0.321438	1.0	0.015469
3	0.0	0.434531	0.0	0.103644
4	1.0	0.434531	1.0	0.015713
..	...	...	...	...
886	1.0	0.334004	0.5	0.025374
887	0.0	0.233476	0.0	0.058556
888	0.0	0.434531	1.0	0.045771
889	1.0	0.321438	0.0	0.058556
890	1.0	0.396833	1.0	0.015127

[891 rows x 4 columns]

```
In [16]: #16 test_data <- lakukan normalisasi pada test_data dengan Min-Max 0-1 (dengan nilai min dan max setiap atribut
# pada Langkah 15)
```

```
# variabel new_min_max_train_data merupakan nilai min dan max pada setiap kolom pada train_data setelah
# dilakukan normalisasi pada langkah sebelumnya. nilai new_min_max_train_data pada setiap kolom adalah 1 dan 0.
# Nilai 1 dan 0 digunakan sebagai nilai max dan min untuk normalisasi pada test_data
```

```
new_min_max_train_data = train_data.agg([min, max])
test_data = (test_data - train_data.min())*(1-0)/(train_data.max() - train_data.min()) + 0
print(new_min_max_train_data)
print('\n-----\n')
print(test_data)
```

	Sex	Age	Pclass	Fare
min	0.0	0.0	0.0	0.0
max	1.0	1.0	1.0	1.0

```
-----
```

	Age	Fare	Pclass	Sex
0	34.5	7.8292	3.0	1.0
1	47.0	7.0000	3.0	0.0
2	62.0	9.6875	2.0	1.0
3	27.0	8.6625	3.0	1.0
4	22.0	12.2875	3.0	0.0
..	...	...	...	...
409	3.0	13.7750	3.0	0.0
411	37.0	90.0000	1.0	0.0
412	28.0	7.7750	3.0	0.0
414	39.0	108.9000	1.0	0.0
415	38.5	7.2500	3.0	1.0

```
[331 rows x 4 columns]
```

```
In [17]: #17 class_result <- Lakukan klasifikasi test_data terhadap train_data dengan 3-NN (output memakai
# class pada train_label)

# Klasifikasi dilakukan dengan membuat model dari train_data dengan output class pada train_label.
# Hal tersebut dilakukan dengan menggunakan method fit(). Hasil klasifikasi dapat diketahui dengan
# menggunakan method predict() yang disimpan pada variabel class_result

from sklearn.metrics import accuracy_score

kNN = KNeighborsClassifier(n_neighbors=3, weights = 'distance')
kNN.fit(train_data, train_label.astype('int').values.ravel())
class_result = kNN.predict(test_data)
```

```
In [18]: #18 error <- Bandingkan hasil klasifikasi class_result dengan test_label. Jika tidak sama berarti error.
# Hitunglah jumlah error dari seluruh class_result

# Jumlah error bisa didapatkan dengan mengurangi jumlah test_data dengan prosentase dari accuracy_score

score = accuracy_score(class_result, test_label)
error = 1 - score
score = round(score * 100, 2)
error = round(error * 100, 2)

print('Score : ', score, '%')
print('Error : ', error, '%')

Score :  61.63 %
Error :  38.37 %
```

```
In [19]: #19 error_ratio <- error dibagi jumlah test_data, dikali 100 (%)

error_ratio = round((error/len(test_label)) * 100, 2)
print('Error Ratio : ', error_ratio, '%')

Error Ratio :  11.59 %
```

Terimakasih