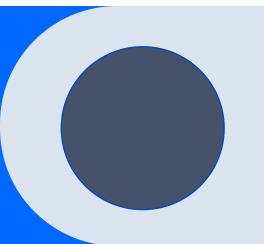
Praktikum Data Mining Minggu Ke-4



Faris Saifullah 3124640034

```
#no.1
import pandas as pd
dataset = pd.read_csv('titanic.csv')
print(dataset.head())
```

Analisa:

Menampilkan dataset dari file data titanic.csv

```
PassengerId Survived Pclass \
                                                         Age SibSp
                        Braund, Mr. Owen Harris
Cumings, Mrs. John Bradley (Florence Briggs Th...
                         Heikkinen, Miss. Laina
    Futrelle, Mrs. Jacques Heath (Lily May Peel)
                                                female 35.0
                        Allen, Mr. William Henry
Parch
                Ticket
                           Fare Cabin Embarked
             A/5 21171 7.2500
      STON/02. 3101282 7.9250
                       53.1000 C123
                373450
                        8.0500
```



```
#no.2
dataset = pd.read_csv('titanic_test.csv')
print(dataset.head())
```

Analisa:

Menampilkan dataset dari file data titanic_test.csv

```
PassengerId Pclass
                                                                     Sex
                                                Kelly, Mr. James
                                                                    male
       892
                                Wilkes, Mrs. James (Ellen Needs)
        893
                                                                  female
        894
                                       Myles, Mr. Thomas Francis
                                                                    male
        895
                                                Wirz, Mr. Albert
                                                                    male
        896
                    Hirvonen, Mrs. Alexander (Helga E Lindqvist)
                                                                  female
                    Ticket
                               Fare Cabin Embarked
                    330911
                            7.8292
47.0
                    363272
                                      NaN
                    240276
27.0
                    315154
                             8.6625
22.0
                 1 3101298 12.2875
                                                 s
```



```
#no.3
train_data = pd.read_csv('titanic.csv')
train_data_features = train_data[['Age', 'Fare']]
pos_missing_train = train_data_features[train_data_features.isna().any(axis=1)].index.tolist()
train_data_cleaned = train_data_features.dropna()
print("Data setelah missing values dihilangkan:")
print(train_data_cleaned.head())
print("\nPosisi missing values yang dihilangkan:")
print(pos_missing_train[:5])
```

Analisa:

Menghilangkan baris data yang terdapat missing values pada file titanic.csv

```
Data setelah missing values dihilangkan:
    Age    Fare
0 22.0 7.2500
1 38.0 71.2833
2 26.0 7.9250
3 35.0 53.1000
4 35.0 8.0500

Posisi missing values yang dihilangkan:
[5, 17, 19, 26, 28]
```

Output

```
test_data = pd.read_csv('titanic_test.csv')
test_data_features = test_data[['Age', 'Fare']]
pos_missing_test = test_data_features[test_data_features.isna().any(axis=1)].index.tolist()
test_data_cleaned = test_data_features.dropna()
print("Data setelah missing values dihilangkan:")
print(test_data_cleaned.head())
print("\nPosisi missing values yang dihilangkan:")
print(pos_missing_test[:5])
```

Analisa:

Menghilangkan baris data yang terdapat missing values pada file titanic_test.csv

```
Data setelah missing values dihilangkan:
Age Fare
0 34.5 7.8292
1 47.0 7.0000
2 62.0 9.6875
3 27.0 8.6625
4 22.0 12.2875

Posisi missing values yang dihilangkan:
[10, 22, 29, 33, 36]
```



Output

```
#no.5

df = pd.read_csv('titanic.csv')

train_label = df['Survived'].dropna()

print("Data pada kolom 'Survived' yang tidak memiliki nilai kosong:")

print(train_label.head())
```

Analisa:

Menampilkan data kolom 'Survived' yang tidak memiliki nilai kosong



Output

```
#no.6
df_test = pd.read_csv('titanic_testlabel.csv')
test_label = df_test.dropna()
print("Data pada titanic_testlabel.csv yang tidak memiliki nilai kosong:")
print(test_label.head())
```

		ic_testlabel.csv Survived	yang	tidak	memiliki	nilai	kosong:
0	892	0					
1	893	1					
2	894	0					
3	895	0					
4	896	1					

Analisa:

Menampilkan data yang tidak memiliki nilai kosong pada file titanic_testlabel.csv



```
#no.7

df = pd.read_csv('titanic.csv')

df_numeric = df.select_dtypes(include=['float64', 'int64']).copy()

min_values = df_numeric.min()

max_values = df_numeric.max()

new_min = 0

new_max = 1

train_data = (df_numeric - min_values) * (new_max - new_min) / (max_values - min_values) + new_min

print("Data setelah normalisasi (0-1):")

print(train_data.head())

print("\nNilai minimum setiap atribut sebelum normalisasi:")

print(min_values)

print("\nNilai maksimum setiap atribut sebelum normalisasi:")

print(max_values)
```

Analisa:

Menormalisasi kolom yang berada di file titanic.csv

```
Data setelah normalisasi (0-1):
   PassengerId Survived Pclass
                                      Age SibSp
                                                            Fare
     0.000000
                            1.0 0.271174 0.125
                                                   0.0 0.014151
     0.001124
                    1.0
                            0.0 0.472229 0.125
                                                        0.139136
     0.002247
                            1.0 0.321438 0.000
                                                        0.015469
     0.003371
                            0.0 0.434531 0.125
                                                        0.103644
      0.004494
                                                   0.0 0.015713
                            1.0 0.434531 0.000
Nilai minimum setiap atribut sebelum normalisasi:
PassengerId
              1.00
Survived
              0.00
Pclass
               1.00
Age
              0.42
SibSp
               0.00
Parch
               0.00
Fare
              0.00
dtype: float64
Nilai maksimum setiap atribut sebelum normalisasi:
PassengerId
               891.0000
Survived
                1.0000
Pclass
                3.0000
               80.0000
Age
SibSp
                8.0000
Parch
                6,0000
Fare
               512.3292
dtype: float64
```

```
#no.8

df = pd.read_csv('titanic_test.csv')

df_numeric = df.select_dtypes(include=['float64', 'int64']).copy()

min_values = df_numeric.min()

max_values = df_numeric.max()

new_min = 0

new_max = 1

train_data = (df_numeric - min_values) * (new_max - new_min) / (max_values - min_values) + new_min

print("Data setelah normalisasi (0-1):")

print(train_data.head())

print("\nNilai minimum setiap atribut sebelum normalisasi:")

print(min_values)

print("\nNilai maksimum setiap atribut sebelum normalisasi:")

print(max_values)
```

Analisa:

Menormalisasi kolom yang berada di file titanic_test.csv

```
Data setelah normalisasi (0-1):
   PassengerId Pclass
                                                      Fare
                             Age SibSp
                                           Parch
                   1.0 0.452723 0.000 0.000000
                                                  0.015282
      0.002398
                   1.0 0.617566 0.125
                                                  0.013663
      0.004796
                   0.5 0.815377 0.000
      0.007194
                   1.0 0.353818 0.000
                                                  0.016908
      0.009592
                   1.0 0.287881 0.125 0.111111 0.023984
Nilai minimum setiap atribut sebelum normalisasi:
PassengerId
               892.00
Pclass
                1.00
                 0.17
Age
SibSp
                 0.00
Parch
                 0.00
Fare
                 0.00
dtype: float64
Nilai maksimum setiap atribut sebelum normalisasi:
PassengerId
               1309.0000
Pclass
                  3.0000
Age
                 76,0000
SibSp
                  8.0000
Parch
                  9.0000
Fare
                512,3292
dtype: float64
```



```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report
df_train = pd.read_csv('titanic.csv')
df_test = pd.read_csv('titanic_test.csv')
df_train = df_train.select_dtypes(include=['float64', 'int64']).dropna()
df_test = df_test.select_dtypes(include=['float64', 'int64']).dropna()
X_train = df_train.drop('Survived', axis=1)
y_train = df_train['Survived']
X_test = df_test.drop('Survived', axis=1, errors='ignore')
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
for k in range(1, 11):
   print(f"\nKlasifikasi dengan k = {k}")
   knn = KNeighborsClassifier(n_neighbors=k)
   knn.fit(X_train_scaled, y_train)
   y_pred = knn.predict(X_test_scaled)
   print("Hasil klasifikasi:")
   print(classification_report(y_train, knn.predict(X_train_scaled)))
   if 'Survived' in df_test.columns:
        print("Classification report untuk test_data:")
       print(classification_report(df_test['Survived'], y_pred))
```

Pecision Pecal f1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-score Support Pecal F1-scor	Klasifikasi dengan k = 1 Hasil klasifikasi:					Klasifikasi dengan k = 6 Hasil klasifikasi:					
1 1.60 1.60 1.60 290 accuracy macro avg 1.60 1.60 1.60 1.60 1.60 290 accuracy macro avg 1.60 1.60 1.60 1.60 714 accuracy macro avg 1.60 1.60 1.60 714 accuracy macro avg 1.60 1.60 1.60 714 accuracy macro avg 6.76 6.75 714 accuracy macro avg 6.83 6.77 6.75 714 accuracy macro avg 6.81 6.81 6.57 6.75 714 accuracy macro avg 6.81 6.81 6.81 6.81 6.81 6.81 6.87 714 accuracy macro avg 6.80 6.80 6.80 714 accuracy macro avg 6.81 6.81 6.81 6.81 6.81 6.81 6.81 714 accuracy macro avg 6.81 6.81 6.81 6.81 6.81 714 accuracy macro avg 6.81 6.81 6.81 6.81 6.81 714 accuracy macro avg 6.83 6.82 6.83 6.82 6.82 6.82 6.82 6.82 6.82 6.82 6.82 6.82 6.83 6.82			recall	f1-score	support		precision	recall	f1-score	support	
1 1.00 1.00 1.00 290		1 00	1 00	1 00	424	0	0.75	0.91	0.82	424	
accuracy are and a second seco	_					1	0.81	0.57	0.67	290	
Securacy 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 714 1.00 1.00 1.00 714 1.00 1.00 1.00 714 1.00 1.00 1.00 714 1.00 1.00 1.00 1.00 714 1.00 1.00 1.00 1.00 714 1.00 1.00 1.00 1.00 1.00 714 1.00 1.00 1.00 1.00 1.00 714 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 714 1.00 1.0	•	1.00	1.00	1.00	250						
### ### ### ### ### ### ### ### ### ##	accuracy			1.00	714		0.70				
Klasifikasi dengan k = 2		1.00	1.00			_					
Classifikasi dengan k = 2 Hasil klasifikasi dengan k = 2 Hasil klasifikasi: precision recall f1-score support	_					weighted avg	0.78	0.77	0.76	714	
Hasil klasifikasi Precision Precisio											
Hasifikasi:	Vinetfilmet d	angan k - 3									
Precision Prec						Hasil klasifikasi:					
B B B B B B B B B B	masii kiasifi		recall	f1-score	support		precision	recall	f1-score	support	
8		precision	receil	11-30016	suppor c	_					
1 1.00 0.58 0.73 290 accuracy across of the second of the	9	0.78	1.00	0.87	424						
accuracy macro avg 0.89 0.79 0.80 714 weighted avg 0.87 0.83 0.82 714 weighted avg 0.87 0.83 0.82 714 weighted avg 0.87 0.83 0.82 714 weighted avg 0.87 0.83 0.84 0.84 424 1 0.77 0.75 0.76 290 accuracy macro avg 0.80 0.81 0.81 714 weighted avg 0.81 0.81 0.81 714 weighted avg 0.83 0.84 0.84 424 1 0.77 0.75 0.76 290 accuracy macro avg 0.80 0.80 0.80 714 weighted avg 0.81 0.81 0.81 714 weighted avg 0.83 0.86 0.86 714 weighted avg 0.85 0.86 0.86 714 weighted avg 0.87 0.87 714 macro avg 0.80 0.80 0.80 714 714 weighted avg 0.81 0.81 0.81 714 Weighted avg 0.83 0.86 0.86 714 weighted avg 0.83 0.86 0.86 714 714 Weighted avg 0.83 0.86 0.86 290 accuracy macro avg 0.76 0.84 0.80 424 1 0.83 0.86 0.86 290 accuracy macro avg 0.75 0.73 0.75 714 Weighted avg 0.78 0.77 714 Weighted avg 0.78 0.77 714 Meighted avg 0.78 0.78 0.77 714 Meighted avg 0.78 0.78 0.77 714 macro avg 0.75 0.73 0.75 714 Meighted avg 0.78 0.79 0.77 714 macro avg 0.76 0.78 0.75 714 Meighted avg 0.78 0.79 0.77 714 macro avg 0.76 0.77 0.77 714 macro avg 0.78 0.79 0.77 714 weighted avg 0.76 0.75 0.75 714 macro avg 0.78 0.79 0.77 714 weighted avg 0.76 0.75 0.75 0.75 714 macro avg 0.78 0.79 0.77 714 macro avg 0.78 0.79 0.79 0.79 0.79 0.79 0.79 0.79 0.79	_					1	0.74	0.64	0.69	290	
accuracy	_								0.70	744	
macro avg	accuracy			0.83	714	_	0.76	0.74			
Klasifikasi dengan k = 3 Hasil klasifikasi: precision recall f1-score support		0.89	0.79	0.80	714	_					
Hasil klasifikasi dengan k = 3	weighted avg	0.87	0.83	0.82	714	weighted avg	0.76	0.76	0.76	/14	
Hasil klasifikasi dengan k = 3											
Hasil klasifikasi dengan k = 3						Klasifikasi d	lengan k = 8				
Hasil klasifikasi:											
## Precision recall f1-score support ## 8	Hasil klasifi	kasi:				HOSII KIOSIII		recall	f1-score	support	
8		precision	recall	f1-score	support		precision	100011	11-30010	Suppor C	
8						a	9.74	0.89	0.81	424	
1 0.77 0.75 0.76 290 accuracy	0	0.83	0.84	0.84	424	_					
accuracy	1	0.77	0.75	0.76	290	_					
accuracy						accuracy			0.75	714	
### weighted avg							0.76	0.72	0.73	714	
Klasifikasi dengan k = 4 Hasil klasifikasi: precision recall f1-score support	_					_				714	
Hasil klasifikasi dengan k = 4 Hasil klasifikasi:	weighted avg	0.81	0.81	0.81	714						
Hasil klasifikasi dengan k = 4 Hasil klasifikasi:											
Hasil klasifikasi: precision recall f1-score support 0 0.75 0.92 0.83 424 1 0.73 0.61 0.67 290 accuracy macro avg 0.79 0.74 0.75 714 weighted avg 0.78 0.77 0.76 714 Klasifikasi dengan k = 5 Hasil klasifikasi: precision recall f1-score support Klasifikasi dengan k = 5 Hasil klasifikasi: precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.78 0.74 0.79 0.76 714 ### 0 0.79 0.86 0.82 424 1 0.78 0.74 0.89 0.81 424 1 0.76 0.67 0.72 290 accuracy macro avg 0.75 0.75 714 ### 0 0.74 0.89 0.81 424 1 0.78 0.75 714 ### 0 0.74 0.89 0.81 424 1 0.78 0.75 714 ### 0 0.74 0.89 0.81 424 ### 0 0.74 0.89 0.81 424 ### 0 0.74 0.89 0.81 424 ### 0 0.75 714 ### 0 0.76 0.75 714 ### 0 0.76 0.75 714 ### 0 0.77 0.77 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.75 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.75 0.74 714 ### 0 0.76 0.75 0.76 0.75 0.74 714 ### 0 0.76 0.76 0.77 0.77 0.77 0.76 0.76 0.						Klasifikasi d	lengan k = 9				
precision recall f1-score support 0 0.75 0.92 0.83 424 1 0.73 0.61 0.67 290 accuracy 0.77 714 macro avg 0.79 0.74 0.75 714 weighted avg 0.78 0.77 0.76 714 Klasifikasi dengan k = 5 Hasil klasifikasi: precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.78 0.54 0.64 290 accuracy 0.75 0.73 0.75 714 Klasifikasi dengan k = 5 Hasil klasifikasi: precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.78 0.54 0.64 290 accuracy 0.75 714 accuracy 0.75 0.75 714 Macro avg 0.75 0.75 714 Macro avg 0.75 0.75 0.75 714 Macro avg 0.75 0.75 0.75 714 Macro avg 0.75 0.75 714 Macro avg 0.75 0.75 0.75 714 Macro avg 0.75 0.75 714 Macro avg 0.75 0.75 0.75 714 Macro avg 0.78 0.77 0.77 714 Macro avg 0.78 0.77 0.77 714 Macro avg 0.78 0.77 0.77 714 Macro avg 0.78 0.75 0.75 0.74 714 Macro avg 0.75 0.75 0.74 714						Hasil klasifi					
## 8	Masii Klasiti						precision	recall	f1-score	support	
8 8.75 8.92 8.83 424 1 8.73 8.61 8.67 298 accuracy 8.77 714 macro avg 8.79 8.74 8.75 714 weighted avg 8.78 8.77 8.76 714 Klasifikasi dengan k = 5 Hasil klasifikasi:		precision	recall	+1-score	support						
1 0.83 0.56 0.66 290 accuracy 0.77 714 macro avg 0.79 0.74 0.75 714 weighted avg 0.78 0.77 0.76 714 Klasifikasi dengan k = 5 Hasil klasifikasi: precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.76 0.67 0.72 290 accuracy 0.75 0.73 0.73 714 Weighted avg 0.75 0.75 0.75 714 Klasifikasi dengan k = 10 Hasil klasifikasi: precision recall f1-score support 0 0.74 0.89 0.81 424 1 0.78 0.54 0.64 290 accuracy 0.75 714 accuracy 0.75 714 macro avg 0.75 0.75 714 weighted avg 0.76 0.72 0.73 714 weighted avg 0.76 0.72 0.73 714 weighted avg 0.76 0.72 0.73 714 weighted avg 0.76 0.75 0.75 0.74 714		0.75	0.00	0.03	424	0	0.76	0.84	0.80	424	
accuracy	_					1	0.73	0.61	0.67	290	
accuracy 8.77 714 macro avg 8.79 9.74 8.75 714 weighted avg 9.78 9.77 8.76 714 Klasifikasi dengan k = 5 Hasil klasifikasi:	•	0.03	0.50	0.00	250						
macro avg 0.79 0.74 0.75 714 macro avg 0.75 0.75 0.75 714 weighted avg 0.78 0.77 0.76 714 weighted avg 0.75 0.75 0.75 714 Klasifikasi dengan k = 5 Hasil klasifikasi: Precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.78 0.81 424 1 0.76 0.67 0.72 290 0.78 0.74 0.89 0.81 424 1 0.76 0.67 0.72 290 0.78 0.54 0.64 290 accuracy 0.75 0.75 714 macro avg 0.74 0.89 0.81 424 1 0.76 0.67 0.72 290 0.75 0.75 0.75 714 macro avg 0.76 0.75 0.73 714 0.75 0.75 0.73	accuracy			9.77	714	_					
Weighted avg 0.78 0.77 0.76 714 Weighted avg 0.73 0.73 0.75 714 Klasifikasi dengan k = 5 Hasil klasifikasi: precision Klasifikasi dengan k = 10 Hasil klasifikasi: precision Hasil klasifikasi: precision 0.74 0.89 0.81 424 0 0.79 0.86 0.82 424 1 0.78 0.54 0.64 290 accuracy macro avg 		9.79	9.74			_					
Klasifikasi dengan k = 5 Hasil klasifikasi: precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.76 0.67 0.72 290 accuracy	_					weighted avg	0.75	0.75	0.75	714	
Hasil klasifikasi dengan k = 5		00									
Hasil klasifikasi dengan k = 5						Vinetfilmet e	tongon k - 10				
Hasil klasifikasi: precision recall f1-score support 0 0.79 0.86 0.82 424 1 0.76 0.67 0.72 290 accuracy 0.78 714 macro avg 0.78 0.77 0.77 714 weighted avg 0.76 0.75 0.74 714 weighted avg 0.76 0.75 0.74 714	Klasifikasi d	engan k = 5	_								
9 0.79 0.86 0.82 424 1 0.78 0.54 0.64 290 1 0.76 0.67 0.72 290 accuracy 0.78 714 macro avg 0.78 0.78 0.77 714 weighted avg 0.76 0.75 0.74 714	Hasil klasifikasi:				Masii Kiasiti		recall	£1-score	support		
0 0.79 0.86 0.82 424 1 0.78 0.54 0.64 290 1 0.76 0.67 0.72 290 accuracy 0.78 714 macro avg 0.76 0.72 0.73 714 macro avg 0.78 0.77 714 weighted avg 0.76 0.75 0.74 714		precision	recall	f1-score	support		precision	100011	11 30010	Suppor C	
0 0.79 0.86 0.82 424 1 0.78 0.54 0.64 290 1 0.76 0.67 0.72 290 accuracy 0.78 714 macro avg 0.76 0.72 0.73 714 macro avg 0.78 0.77 714 weighted avg 0.76 0.75 0.74 714						9	0.74	0.89	0.81	424	
1 0.76 0.67 0.72 290 accuracy 0.75 714 accuracy 0.78 714 macro avg 0.76 0.72 0.73 714 macro avg 0.78 0.77 714 weighted avg 0.76 0.75 0.74 714	0	0.79	0.86	0.82	424						
accuracy 0.78 714 macro avg 0.76 0.72 0.73 714 macro avg 0.76 0.72 0.73 714 weighted avg 0.76 0.75 0.74 714	1	0.76	0.67	0.72	290	-				230	
accuracy 0.78 714 macro avg 0.76 0.72 0.73 714 macro avg 0.76 0.72 0.73 714 weighted avg 0.76 0.75 0.74 714						accuracy			0.75	714	
macro avg 0.78 0.77 0.77 714 weighted avg 0.76 0.75 0.74 714							0.76	0.72			
weighted avg 0.78 0.78 714										714	
	weighted avg	0.78	0.78	0.78	714						

```
#no.10
import pandas as pd
from sklearn.model selection import train test split
from sklearn.preprocessing import StandardScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import classification_report, accuracy_score
df train = pd.read csv('titanic.csv')
df test = pd.read csv('titanic test.csv')
df_train = df_train.select_dtypes(include=['float64', 'int64']).dropna()
df_test = df_test.select_dtypes(include=['float64', 'int64']).dropna()
X_train = df_train.drop('Survived', axis=1)
y_train = df_train['Survived']
X_test = df_test.drop('Survived', axis=1, errors='ignore')
y_test = df_test['Survived'] if 'Survived' in df_test.columns else nonzero
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
results = {}
for k in range(1, 11):
    print(f"\nKlasifikasi dengan k = {k}")
    knn = KNeighborsClassifier(n neighbors=k)
    knn.fit(X train scaled, y train)
    y_pred = knn.predict(X_test_scaled)
    results[k] = y_pred
    if y_test is not None:
        accuracy = accuracy_score(y_test, y_pred)
        error_ratio = 1 - accuracy
        print(f"Rasio error untuk k = {k}: {error_ratio:.4f}")
    else:
        print("Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung."),
```

```
Klasifikasi dengan k = 1
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 2
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 3
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 4
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 5
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 6
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 7
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 8
Kolom 'Survived' tidak ada di df test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 9
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
Klasifikasi dengan k = 10
Kolom 'Survived' tidak ada di df_test, jadi rasio error tidak dapat dihitung.
```