VALIDATION MODEL OF CLASSIFICATION

NADILA IMAARAH

D4 SAINS DATA TERAPAN A

3323600015

APA ITU VALIDATION ?

Validation dalam machine learning adalah proses untuk mengevaluasi performa model sebelum digunakan pada data baru. Tujuannya adalah memastikan bahwa model yang dilatih tidak hanya bekerja baik pada data latih tetapi juga dapat melakukan generalisasi dengan baik terhadap data yang belum pernah dilihat sebelumnya.

Validation biasanya dilakukan dengan membagi dataset menjadi training set dan validation set, lalu mengukur performa model menggunakan metrik tertentu seperti akurasi, presisi, recall, atau F1-score. Ada beberapa metode validation, seperti Hold-Out Method, K-Fold Cross Validation, dan Leave-One-Out Cross Validation (LOO-CV).

01 MEMBACA DATASET

MENGIMPOR LIBRARY

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split, KFold, LeaveOneOut, StratifiedKFold
from sklearn.preprocessing import MinMaxScaler
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
```

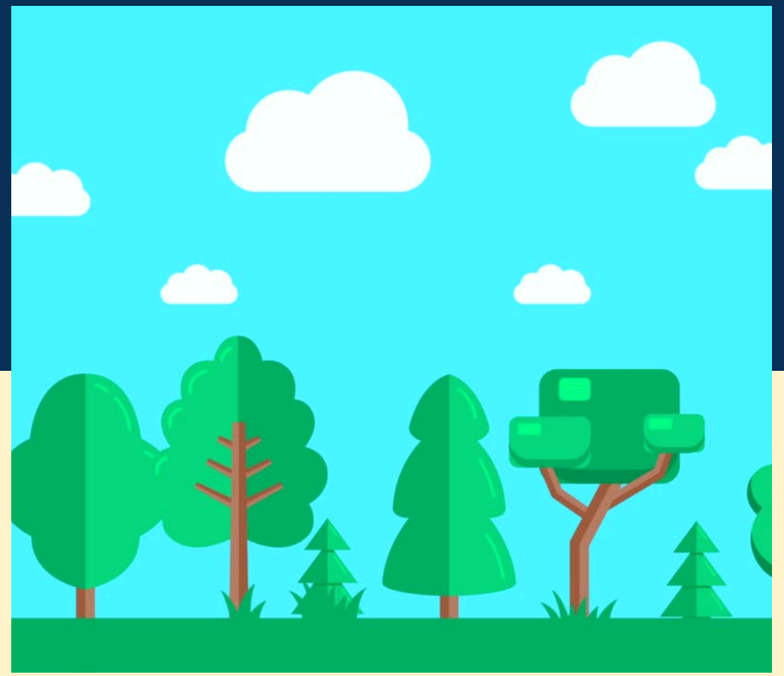
LOAD DATASET

```
dataset = pd.read_csv("titanic.csv")
dataset.head()
```

OUTPUT

	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

Dataset yang digunakan adalah dataset titanic yang terdiri dari beberapa fitur yakni passengerid,survived,pclass,name,sex,age, Sibsp,parch,ticket,fare,cabin,dan embarked. Namun untuk klasifikasi saat ini tidak akan menggunakan semua fitur ini namun hanya mengambil beberapa fitur saja



HOLD OUT METHOD

02

Hold-Out Method adalah metode validasi paling sederhana, di mana dataset dibagi menjadi dua bagian, yaitu training set untuk melatih model dan testing set untuk menguji model.

A

MEMBAGI DATASET

```
train_ho, test_ho = train_test_split(dataset,
                                     test_size=0.3, random_state=42)
```

Pada tahap ini dilakukan split data dengan menggunakan hold out method yakni membagia dataset menjadi 2 bagian yakni 30% data tes dan 70% sisanya adalah data train

B MENGAMBIL FITUR DARI DATA TRAIN

MENGAMBIL FITUR SEX,AGE,PCLASS DAN FARE

```
train_data_ho = train_ho[['Sex', 'Age', 'Pclass', 'Fare']].copy()
test_data_ho = test_ho[['Sex', 'Age', 'Pclass', 'Fare']].copy()
```

HANDLING MISSING VALUE

```
train_data_ho['Age'] = train_data_ho.groupby('Pclass')['Age'].transform(lambda x: x.fillna(x.mean()))
test_data_ho['Age'] = test_data_ho.groupby('Pclass')['Age'].transform(lambda x: x.fillna(x.mean()))
```

MENGAMBIL KOLOM SURVIVED SEBAGAI TARGET

```
# Mengambil kolom 'Survived' sebagai target
label_train_ho = train_ho[['Survived']]
label_test_ho = test_ho[['Survived']]
```

Pada tahap ini saya mengambil beberapa fitur yakni sex, age, pclass, dan fare dari data train. Kemudian saya handling missing value pada fitur age. Untuk mengisi missing value di fitur age ini saya menggunakan rata-rata yang berdasarkan pclassnya masing-masing

C

ENCODING

```
train_data_ho["Sex"] = train_data_ho["Sex"].map({"male": 0, "female": 1})
test_data_ho["Sex"] = test_data_ho["Sex"].map({"male": 0, "female": 1})
```

Pada tahap ini saya melakukan encoding pada fitur sex, karena fitur sex berbentuk kategorikal disini saya akan merubahnya menjadi numerik yakni 1 dan 0 untuk mempermudah analisis lanjutan

D NORMALISASI

```
1
2   scaler = MinMaxScaler()
3   train_ho_scaled = scaler.fit_transform(train_data_ho)
4   test_ho_scaled = scaler.transform(test_data_ho)
```

Sebelum melakukan klasifikasi di sini saya melakukan normalisasi agar skala dari setiap fitur memiki skala yang seragam dan tidak memiliki selisih yang jauh. Untuk normalisasi ini saya menggunakan Min Max. Min Max fit_transform ini digunakan untuk normalisasi pada data training dan transform digunakan untuk normalisasi data tes

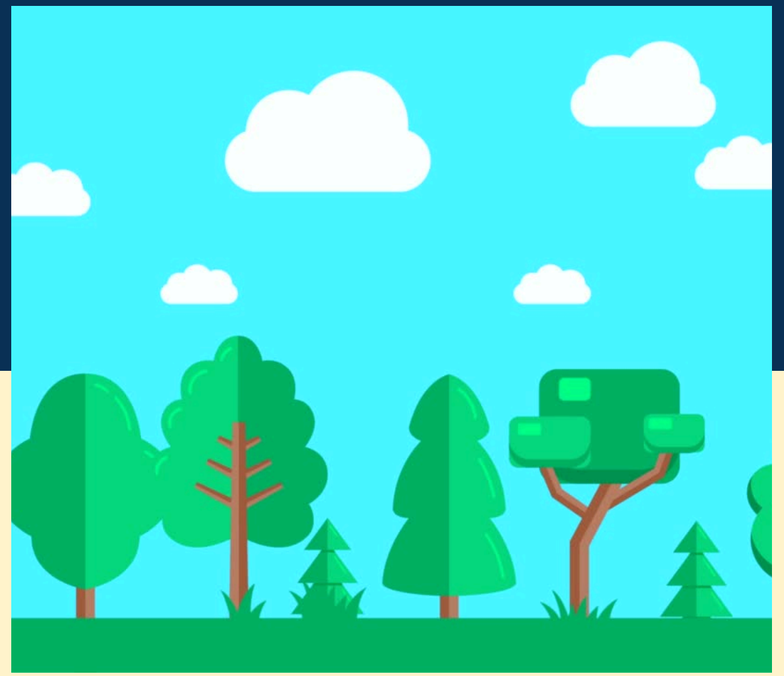
E

KLASIFIKASI KNN

```
knn = KNeighborsClassifier(n_neighbors=3)
knn.fit(train_ho_scaled, label_train_ho)
y_pred_ho = knn.predict(test_ho_scaled)
error_ratio_ho = 1 - accuracy_score(label_test_ho, y_pred_ho)
print(f"Error Ratio Hold-Out: {error_ratio_ho}")
```

```
Error Ratio Hold-Out: 0.1865671641791044
```

Setelah data melewati normalisasi, selanjutnya di tahap ini akan dilakukan klasifikasi. Klasifikasi ini menggunakan KNN dengan k sebanyak 3. Setelah mengklasifikasi ,maka akan dihitung error rasio dengan cara 1 dikurangi akurasi. Pada klasifikasi ini mendapatkan error rasio sebesar 0.1866



K-FOLD METHOD

03

Leave-One-Out Cross Validation (LOO-CV) adalah metode validasi yang lebih akurat tetapi lebih mahal secara komputasi. Dalam LOO-CV, setiap sampel dalam dataset secara bergantian dijadikan data uji, sementara sampel lainnya digunakan sebagai data latih. Proses ini diulang sebanyak jumlah total sampel (N), sehingga setiap sampel pernah menjadi data uji satu kali. Setelah semua iterasi selesai, hasil evaluasi dirata-ratakan untuk mendapatkan estimasi performa model

K-FOLD METHOD

```python
kf = KFold(n_splits=10, shuffle=True, random_state=42)
error_ratios_kfold = []

for train_idx, test_idx in kf.split(dataset):
    # Membagi dataset menjadi train dan test
    train_kf, test_kf = dataset.iloc[train_idx], dataset.iloc[test_idx]

    # Mengambil fitur dan mengisi missing value Age dengan rata-rata berdasarkan Pclass
    train_data_kf = train_kf[['Sex', 'Age', 'Pclass', 'Fare']].copy()
    test_data_kf = test_kf[['Sex', 'Age', 'Pclass', 'Fare']].copy()

    train_data_kf['Age'] = train_data_kf.groupby('Pclass')['Age'].transform(lambda x: x.fillna(x.mean()))
    test_data_kf['Age'] = test_data_kf.groupby('Pclass')['Age'].transform(lambda x: x.fillna(x.mean()))

    # Mengambil kolom 'Survived' sebagai label
    label_train_kf = train_kf[['Survived']]
    label_test_kf = test_kf[['Survived']]

    #Encoding pada kolom sex
    train_data_kf["Sex"] = train_data_kf["Sex"].map({"male": 0, "female": 1})
    test_data_kf["Sex"] = test_data_kf["Sex"].map({"male": 0, "female": 1})
```

Pada tahap ini saya melakukan klasifikasi namun dengan menggunakan metode split data yang berbeda. Di sini menggunakan k-fold cross validation dengan fold sebanyak 10 yang artinya dataset akan di bagi menjadi 10 bagian secara acak kedalam data train dan data tes

Di sini saya menggunakan looping for untnuk melakukan pemisahan antara fitur dan target dan handling missing value yakni pada kolom age dengan menggunakan rata-rata berdasarkan masing-masing pclass.Selain itu juga melakukan encoding pada kolom sex, dengan merubah isi kolom sex menjadi numerik yakni menjadi angka 0 dan 1. Setelah itu fitur survived dijadikan label dalam klasifikasi in

KLASIFIKASI KNN

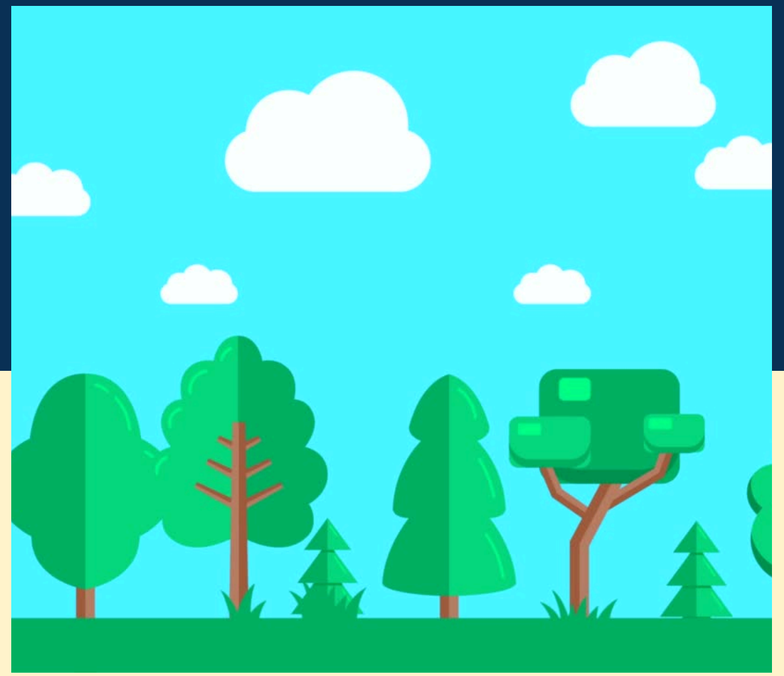
```python
# Normalisasi Min-Max
    scaler = MinMaxScaler()
    train_kf_scaled = scaler.fit_transform(train_data_kf)
    test_kf_scaled = scaler.transform(test_data_kf)

    # Klasifikasi K-NN dan menghitung error ratio
    knn = KNeighborsClassifier(n_neighbors=3)
    knn.fit(train_kf_scaled, label_train_kf.values.ravel())
    y_pred_kf = knn.predict(test_kf_scaled)
    error_ratio_kf = 1 - accuracy_score(label_test_kf, y_pred_kf)
    error_ratios_kfold.append(error_ratio_kf)

# Menghitung rata-rata error ratio
mean_error_ratio_kfold = np.mean(error_ratios_kfold)
print(f"Error Ratio K-Fold (k=10): {mean_error_ratio_kfold}")
```

```
Error Ratio K-Fold (k=10): 0.19189762796504367
```

Setelah encoding barulah dilakukan normalisasi, sama dengn hold out method tadi yakni menggunakan minmax scaller. Selanjutnya dilakukan klasifikasi dengan mengunakan KNN dengan k sebanayak 3. Setelah mengklasifikasikan ,maka akan dihitung error rasio dengan cara 1 dikurangi akurasi. Pada klasifikasi ini mendapatkan error rasio sebesar 0.19189762796504367



LEAVE-ONE-OUT

03

K-Fold Cross Validation merupakan alternatif yang lebih efisien dibanding LOO-CV. Dalam metode ini, dataset dibagi menjadi K lipatan (folds) yang sama besar. Model dilatih dengan K-1 lipatan dan diuji dengan 1 lipatan yang tersisa. Proses ini diulang sebanyak K kali, sehingga setiap lipatan menjadi data uji satu kali. Hasil dari setiap iterasi dirata-ratakan untuk memperoleh estimasi akurasi model.

LEAVE-ONE-OUT METHOD

```python
loo = LeaveOneOut()
error_ratios_loo = []

for train_idx, test_idx in loo.split(dataset):
    # Membagi dataset menjadi train dan test
    train_loo, test_loo = dataset.iloc[train_idx], dataset.iloc[test_idx]

    # Mengambil fitur dan mengisi missing value Age dengan rata-rata berdasarkan Pclass
    train_data_loo = train_loo[['Sex', 'Age', 'Pclass', 'Fare']].copy()
    test_data_loo = test_loo[['Sex', 'Age', 'Pclass', 'Fare']].copy()

    train_data_loo['Age'] = train_data_loo.groupby('Pclass')['Age'].transform(lambda x: x.fillna(x.mean()))
    test_data_loo['Age'] = test_data_loo['Age'].fillna(train_data_loo['Age'].mean())

    # Mengambil kolom 'Survived' sebagai label
    label_train_loo = train_loo[['Survived']]
    label_test_loo = test_loo[['Survived']]

    #Encoding pada kolom sex
    train_data_loo["Sex"] = train_data_loo["Sex"].map({"male": 0, "female": 1})
    test_data_loo["Sex"] = test_data_loo["Sex"].map({"male": 0, "female": 1})
```

Pada tahap ini saya melakukan klasifikasi namun dengan menggunakan metode split data yang berbeda. Di sini menggunakan Leave-One-Out Cross Validation (LOO-CV) cara kerja metode ini yakni setiap sampel dalam dataset digunakan sebagai data uji satu per satu, sementara sisanya digunakan sebagai data latih.

Di sini saya menggunakan looping for untnuk melakukan pemisahan antara fitur dan target,serta handling missing value yakni pada kolom age dengan menggunakan rata-rata berdasarkan masing-masing pclass. Selain itu juga melakukan encoding pada kolom sex, dengan merubah isi kolom sex menjadi numerik yakni menjadi angka 0 dan 1. Setelah itu fitur survived dijadikan label dalam klasifikasi ini

KLASIFIKASI KNN

```
1
2       # Normalisasi Min-Max
3       scaler = MinMaxScaler()
4       train_loo_scaled = scaler.fit_transform(train_data_loo)
5       test_loo_scaled = scaler.transform(test_data_loo)
6
7       # Klasifikasi K-NN dan menghitung error ratio
8       knn = KNeighborsClassifier(n_neighbors=3)
9       knn.fit(train_loo_scaled, label_train_loo.values.ravel())
10      y_pred_loo = knn.predict(test_loo_scaled)
11      error_ratio_loo = 1 - accuracy_score(label_test_loo, y_pred_loo)
12      error_ratios_loo.append(error_ratio_loo)
13
14  # Menghitung rata-rata error ratio
15  mean_error_ratio_loo = np.mean(error_ratios_loo)
16  print(f"Error Ratio LOO: {mean_error_ratio_loo}")
```

```
Error Ratio LOO: 0.19079685746352412
```

Setelah encoding barulah dilakukan normalisasi, sama dengn hold out method tadi yakni menggunakan minmax scaller. Selanjutnya dilakukan klasifikasi dengan mengunakan KNN dengan k sebanyak 3. Setelah mengklasifikasikan ,maka akan dihitung error rasio dengan cara 1 dikurangi akurasi. Pada klasifikasi ini mendapatkan error rasio sebesar 0.19079685746352412

KESIMPULAN

Dari ketiga metode validation, metode Hold Out Method memiliki error ratio yang paling kecil dibandingkan dengan LOO ataupun K-Fold yakni dengan nilai error rasio sebesar 0.1865671641791044 6, sedangkan nilai kfold sebesar 0.19189762796504367 dan nilai dari LOO sebesar 0.1907968574635241 2.

Kemudian K-Fold Validation menghasilkan nilai yang hampir sama dengan Leave-One-Out. Dari ketiga metode tersebut LOO adalah metode yang lama, namun hasilnya cukup efektif karena mencoba keseluruhan data

TERIMA KASIH