UNIVERSITAS AMIKOM YOGYAKARTA

Creative Economy Park

MEMBANGUN MODEL

Mulia Sulistiyono, M.Kom

muliasulistiyono@amikom.ac.id

Learning Objective

Dalam pertemuan ini diharapkan:

A. Mahasiswa mampu melakukan kegitan persiapan pemodelan seperti pembagian data, penyusunan skenario pemodelan

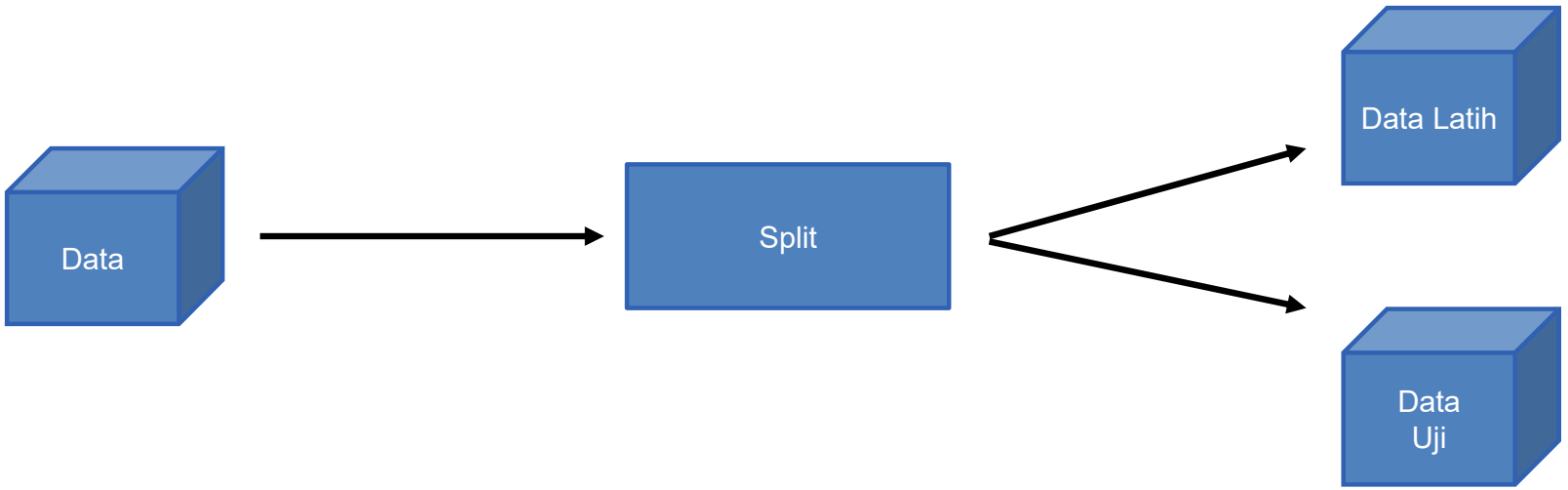
B. Mahasiswa mampu melakukan mampu melakukan proses pemodelan klasifikasi

Outline

- Membangun Skenario Pemodelan :
  - Pembagian data : data latih-uji, *k*-fold cross validation
  - Menentukan Langkah Eksperimen
  - Parameter Evaluasi
- Membangun Model Klasifikasi :
  - Algoritma klasifikasi yang diimplementasi menggunakan library, yaitu : k-NN, Decision Tree, Naïve Bayes, Support Vector Machine dan Boosting
  - Matriks Performansi Klasifikasi

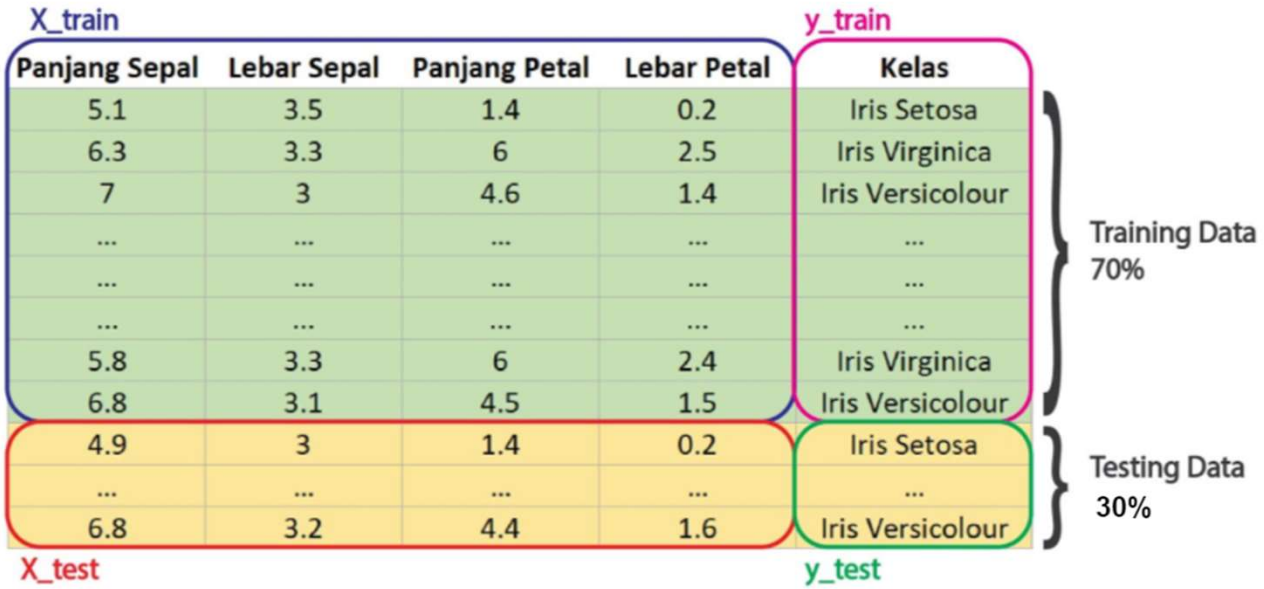
Pembagian Data

- Data dibagi menjadi 2 bagian :
  - Data Latih (*Training Data*) : untuk mengembangkan model
  - Data Uji (*Testing Data*) : untuk Mengukur performansi model



Pembagian Data

- Dataset Iris (https://archive.ics.uci.edu/ml/datasets/iris):
  - Data Latih (*Training Data*) : 70%
  - Data Uji (*Testing Data*) : 30%



Hands On

Data Latih : 70%
Data Uji : 30%

```
[5]  from sklearn.model_selection import train_test_split

     X_train, X_test, y_train, y_test = train_test_split(X, y, train_size = 0.7)
```

```
[6]  print("Banyak data latih setelah dilakukan Train-Test Split: ", len(X_train))
     print("Banyak data uji setelah dilakukan Train-Test Split: ", len(X_test))
```

```
Banyak data latih setelah dilakukan Train-Test Split:  105
Banyak data uji setelah dilakukan Train-Test Split:  45
```

Output, jumlah data latih dan data uji

k-Fold Cross Validation

- *k*-Fold Cross Validation digunakan pada dataset dengan jumlah data yang relatif sedikit
- *k*-Fold Cross Validation dilakukan pada data latih
- Data latih dibagi menjadi *k* bagian kemudian secara iteratif, 1 bagian menjadi data validasi

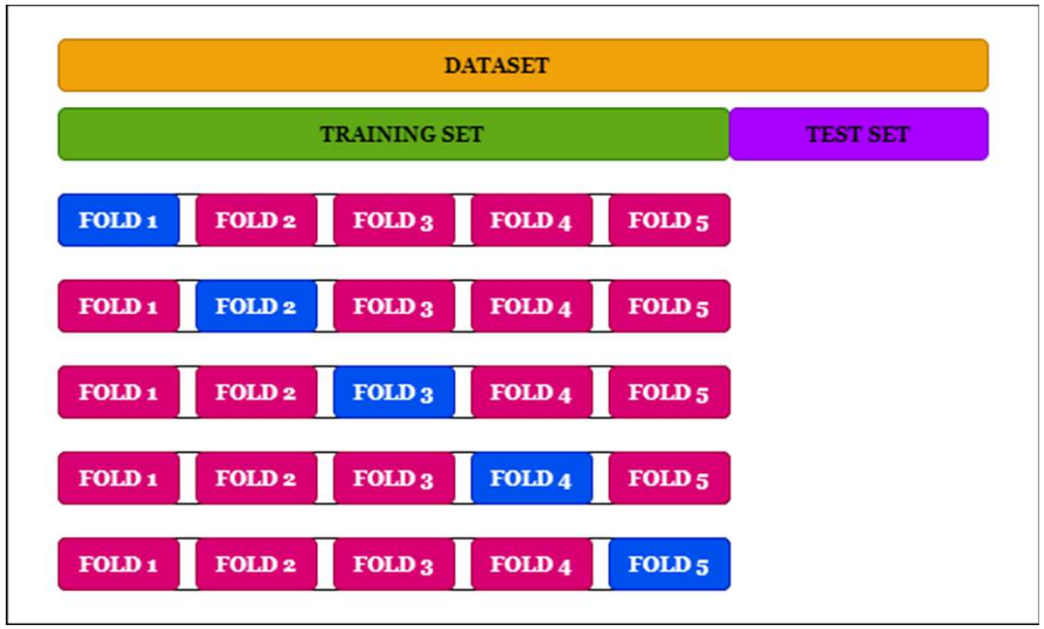


Image Source : https://medium.com/the-owl/k-fold-cross-validation-in-keras-3ec4a3a00538

Hands On

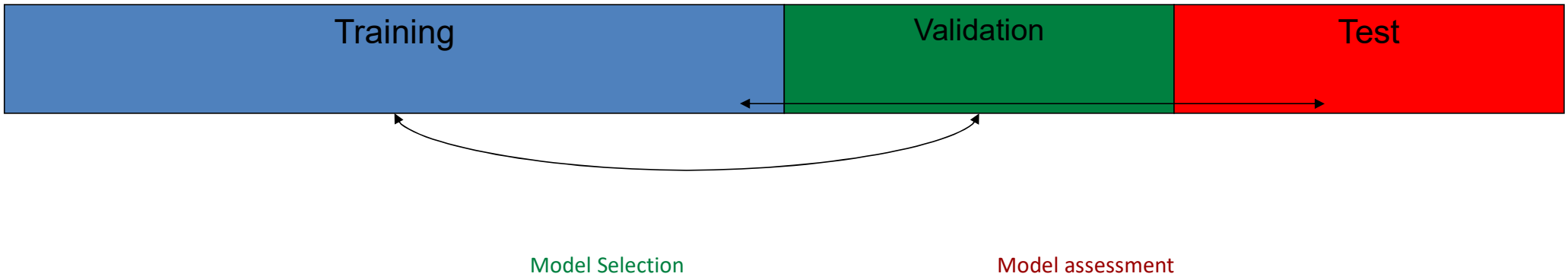
```
[ ]  from sklearn.model_selection import cross_val_score
     from sklearn.svm import SVC

     model = SVC(kernel = 'linear', C = 1)
     scores = cross_val_score(model, X, y, cv = 5)
     print("Akurasi model SVM untuk tiap fold: ", scores)
     print("Akurasi model SVM dengan 5-Fold Cross Validation: ", scores.mean())
```

5 Cross Validation

```
Akurasi model SVM untuk tiap fold:  [0.96666667 1.         0.96666667 0.96666667 1.
Akurasi model SVM dengan 5-Fold Cross Validation:  0.9800000000000001
```

Output akurasi dari setiap fold
Akurasi rata- rata dari seluruh fold

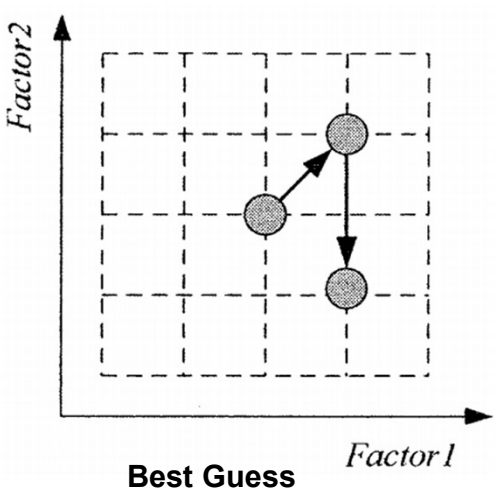
Training – Validation – Testing Data



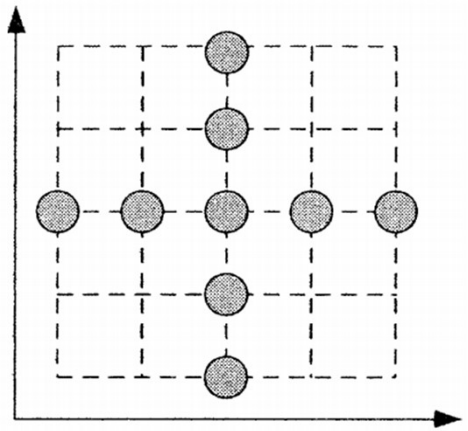
- Model Selection : Mengestimasi performa model - model yang berbeda untuk memilih model yang terbaik, yaitu model dengan minimum error
- Model Assessment : Dari model yang terpilih, mengestimasi error untuk data baru (data uji)

Menentukan Langkah Eksperimen

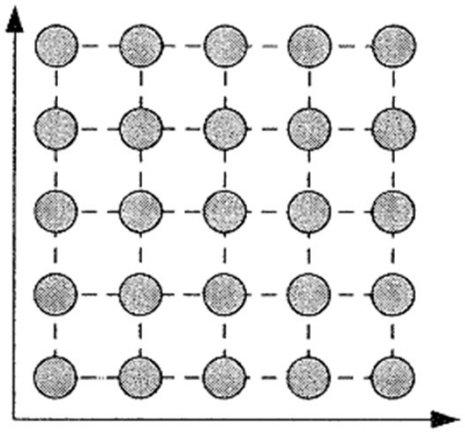
- Setiap metode memiliki parameter tertentu
- Dilakukan eksperimen dengan beberapa variasi parameter
- Parameter yang menghasilkan model performa terbaik akan digunakan selanjutnya
- Beberapa strategi pencarian parameter untuk menghasilkan model terbaik



Best Guess



One Factor at A Time



Grid Search

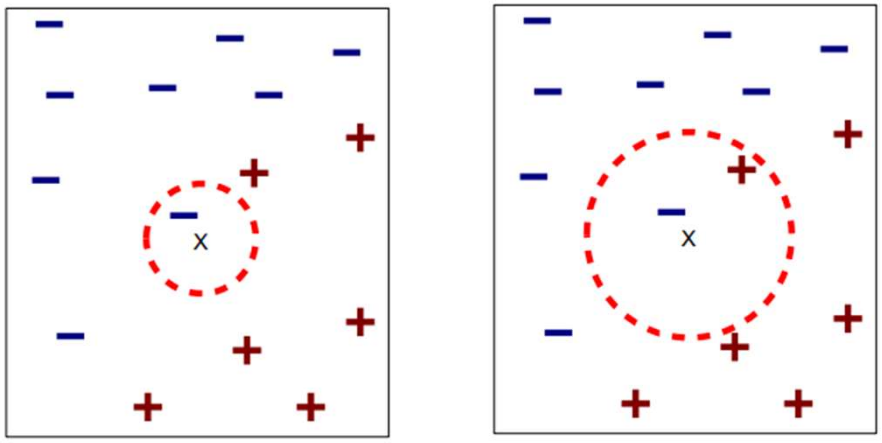
Parameter Evaluasi

- Klasifikasi
  - Akurasi
  - Presisi
  - Recall/Sensitivity
  - Specificity
  - F1-measure
  - …
- Regresi
  - MSE (Mean Squared Error)
  - MAPE (Mean Absolute Percentage Error)
  - …
- Klastering
  - Silhouette Score
  - Davies-Bouldin Index
  - …
- Parameter Evaluasi akan dijelaskan secara detail pada materi berikutnya

Outline

- Membangun Skenario Pemodelan :
  - Pembagian data : data latih-uji, *k*-fold cross validation
  - Menentukan Langkah Eksperimen
  - Parameter Evaluasi
- **Membangun Model Klasifikasi :**
  - Algoritma klasifikasi yang diimplementasi menggunakan library, yaitu : k-NN, Decision Tree, Naïve Bayes, Support Vector Machine dan Boosting
  - Matriks Performansi Klasifikasi

k – Nearest Neighbor (k-NN) Classifier



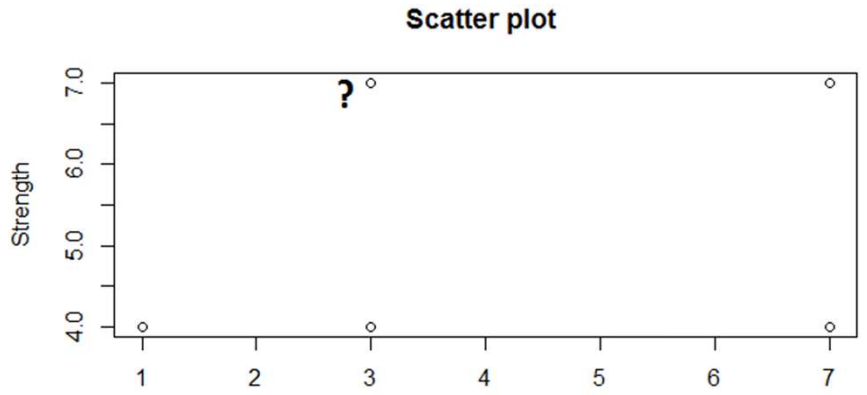
(a) 1-nearest neighbor (b) 2-nearest neighbor

Nearest Neighbor terhadap data baru (x)

- Algoritma :
  - Menentukan nilai k
  - Menghitung jarak antara data baru terhadap semua training data
  - Mengidentifikasi k nearest neighbor
  - Menentukan label/kelas data baru berdasarkan kelas k-nearest neighbor (dapat menggunakan voting)

Contoh kasus

- Terdapat 4 data latih (P1, P2, P3, P4). Data memiliki dua atribut (x1 dan x2)
- Data latih terdiri dari dua kelas (kelas BAD dan kelas GOOD)
- Diperlukan klasifikasi untuk menentukan kelas dari data uji (P5).



Points	X1(Acid Durability)	X2(Strength)	Y(Classification)
P1	7	7	BAD
P2	7	4	BAD
P3	3	4	GOOD
P4	1	4	GOOD
P5	3	7	?

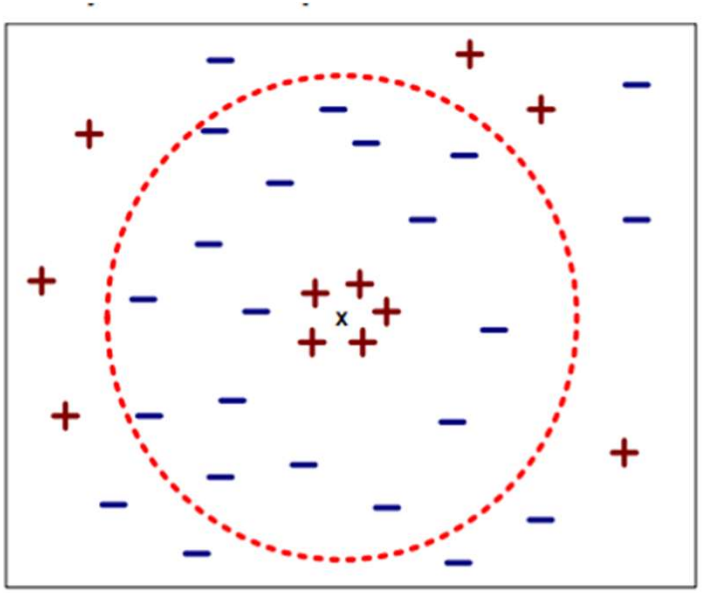
Contoh kasus (lanjutan)

Euclidean Distance of P5(3,7) from	P1	P2	P3	P4
	(7,7)	(7,4)	(3,4)	(1,4)
	Sqrt((7-3) ² + (7-7)²) = $\sqrt{16}$ = 4	Sqrt((7-3) ² + (4-7)²) = $\sqrt{25}$ = 5	Sqrt((3-3) ² + (4-7)²) = $\sqrt{9}$ = 3	Sqrt((1-3) ² + (4-7)²) = $\sqrt{13}$ = 3.60
Class	BAD	BAD	GOOD	GOOD

Points	X1(Durability)	X2(Strength)	Y(Classification)
P1	7	7	BAD
P2	7	4	BAD
P3	3	4	GOOD
P4	1	4	GOOD
P5	3	7	GOOD

Kelebihan dan Kekurangan *k*-NN Classifier

- Cocok untuk data numerik
- Mudah dipahami dan diimplementasikan
- *k*-NN merupakan *lazy learner* (tidak membangun model secara eksplisit)
- Penentuan label/kelas data baru membutuhkan *computational cost* yang cukup tinggi
- Perlu menentukan nilai k yang sesuai
  - Jika k terlalu kecil, sensitif terhadap noise
  - Jika k terlalu besar, nearest neigbor mungkin mencakup data dari kelas lain



Hands On

```
[ ]  from sklearn.neighbors import KNeighborsClassifier
     from sklearn import metrics

     knn = KNeighborsClassifier()

     knn.fit(X_train, y_train)
     y_pred = knn.predict(X_test)
     score = metrics.accuracy_score(y_test, y_pred)
     print("Akurasi dengan menggunakan Nearest Neighbor: ", score)

     Akurasi dengan menggunakan Nearest Neighbor:  0.9777777777777777
```

k-NN Classifier

Proses training data latih, prediksi data uji, perhitungan akurasi

Output akurasi yang dihasilkan *k-NN classifier*

Hands On (lanjutan)

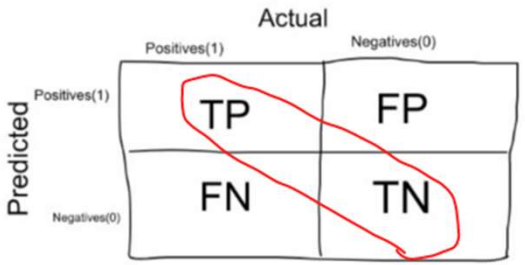
Parameter	Keterangan	Contoh Nilai
n_neighbors	Jumlah Neighbor	- Bilangan Integer (1,2,3,4,....) - Nilai default : 4

```
# Import kNN Classifier dari sklearn
from sklearn.neighbors import KNeighborsClassifier

knn = KNeighborsClassifier(n_neighbors=3) # memilih 3 sebagai banyaknya neighbor
```

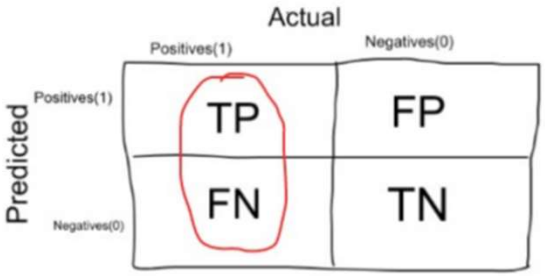
MODEL EVALUATIONS METRICS

- ACCURACY
- PRECISSION
- RECALL
- F1 SCORE

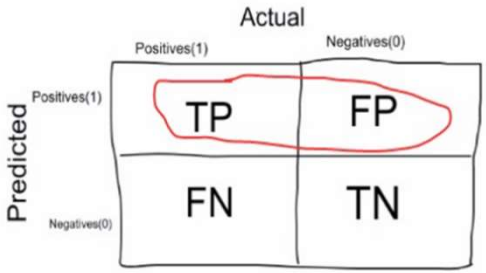
Matriks Performansi Klasifikasi



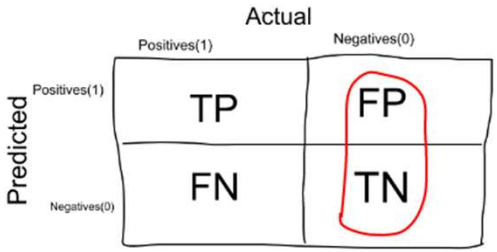
$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$



$$\text{Recall} = \frac{TP}{TP + FN}$$



$$\text{Precision} = \frac{TP}{TP + FP}$$



$$\text{Specificity} = \frac{TN}{TN + FP}$$

$$F_1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Referensi

- CM Bishop, 2006, Pattern Recognition and Machine Learning, Springer
- https://medium.com/the-owl/k-fold-cross-validation-in-keras-3ec4a3a00538
- https://learnopencv.com/svm-using-scikit-learn-in-python/
- https://towardsdatascience.com/boosting-algorithms-explained-d38f56ef3f30
- https://www.analyticsvidhya.com/blog/2015/11/quick-introduction-boosting-algorithms-machine-learning
- https://www.datacamp.com/community/tutorials/adaboost-classifier-python
- https://medium.com/@MohammedS/performance-metrics-for-classification-problems-in-machine-learning-part-i-b085d432082b

Summary

- Perancangan skenario eksperimen merupakan langkah yang dilakukan sebelum membangun model
- Perancangan scenario eksperimen terdiri dari beberapa item :
  - Pembagian Data
  - Strategi langkah eksperimen
  - Parameter evaluasi yang digunakan
- Model klasifikasi dibangun dengan beberapa pilihan algoritma
- Setiap algoritma klasifikasi memiliki kelebihan dan kekurangan
- Perlu dilakukan eksperimen yang komprehensif untuk mendapatkan model yang terbaik bagi suatu kasus/dataset

Tugas Harian

- Gunakan dataset Iris (https://archive.ics.uci.edu/ml/datasets/iris)
- Data dibagi dengan proporsi :
  - Data Latih (70%)
  - Data Uji (30%)
- Terapkan Semua algorithm yang ada dalam hands on dan lakukan analisa perbandingan terhadap performa yang dihasilkan.

Terima Kasih