Direktorat Jenderal Pendidikan Tinggi, Riset, dan, Teknologi
Kementerian Pendidikan, Kebudayaan, Riset, dan Teknologi
Republik Indonesia

DIKTI SIGAP MELAYANI

Kampus Merdeka INDONESIA JAYA

IARC

MICROCREDENTIAL: ASSOCIATE DATA SCIENTIST

01 November – 10 Desember 2021

Pertemuan ke-11

Membangun Model 2 (Regresi Non Linier, dan Klasifikasi)

Profil Pengajar: Nama Lengkap dan Gelar Akademik

Contak Pengajar:

Ponsel:

xxxxxx

Email:

xxxxxxx

Jabatan Akademik: .....

Latar Belakang Pendidikan:

- S1: .....
- S2: .....
- S3: .....

Riwayat/Pengalaman Pekerjaan:

- Dosen
- Xxxx
- Xxxx
- Xxxx
- xxxx

KODE UNIT : **J.62DMI00.013.1**

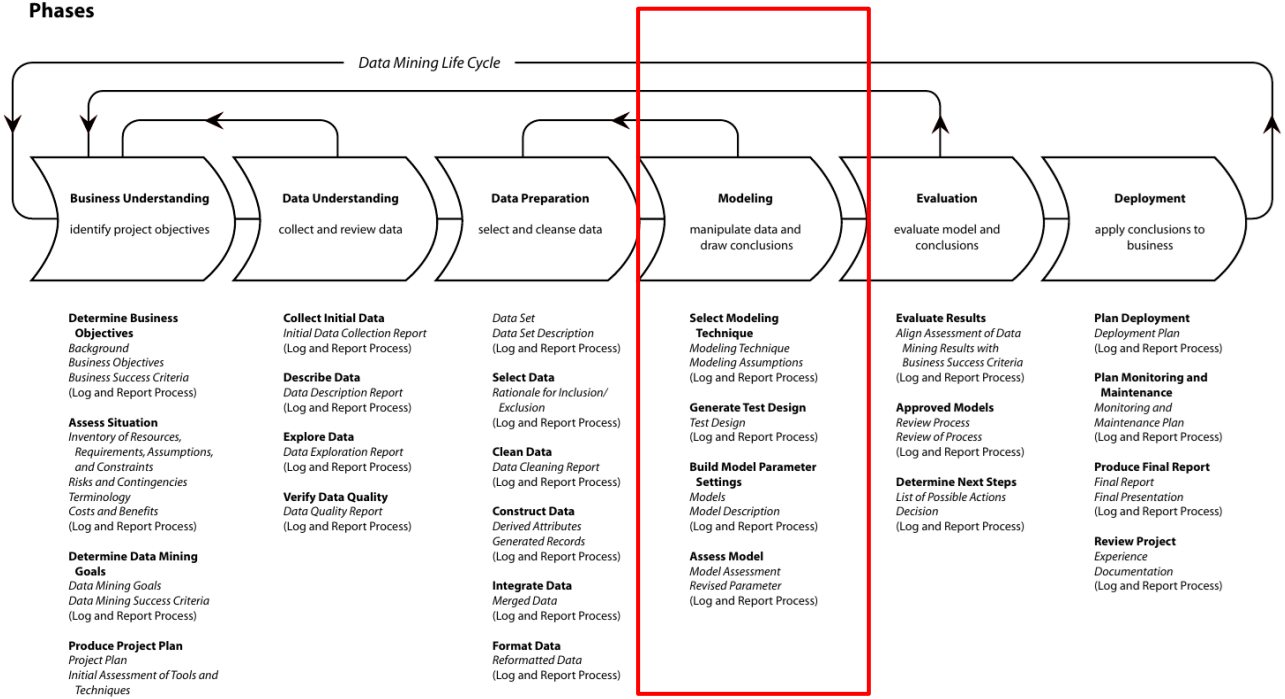
JUDUL UNIT : **Membangun Model**

DESKRIPSI UNIT : Unit kompetensi ini berhubungan dengan pengetahuan, keterampilan, dan sikap kerja yang dibutuhkan dalam membangun model.

ELEMEN KOMPETENSI	KRITERIA UNJUK KERJA
1. Menyiapkan parameter model	1.1 Parameter-parameter yang sesuai dengan model diidentifikasi. 1.2 Nilai toleransi parameter evaluasi pengujian ditetapkan sesuai dengan tujuan teknis.
2. Menggunakan *tools* pemodelan	2.1 ***Tools* untuk membuat model** diidentifikasi sesuai dengan tujuan teknis *data science*. 2.2 Algoritma untuk teknik pemodelan yang ditentukan dibangun menggunakan *tools* yang dipilih. 2.3 Algoritma pemodelan dieksekusi sesuai dengan **skenario pengujian** dan *tools* untuk membuat model yang telah ditetapkan. 2.4 **Parameter model** algoritma dioptimasi untuk menghasilkan **nilai parameter evaluasi** yang sesuai dengan skenario pengujian.

1. Konteks variabel
   1.1 Termasuk di dalam skenario pengujian adalah komposisi *data training* dan *data testing*, cara pemilihan data *training* dan data testing seperti *percentage splitting, random selection,* atau *cross validation.*
   1.2 Yang dimaksud dengan parameter model di antaranya arsitektur model, banyaknya *layer* atau simpul, *learning rate* untuk *neural network,* nilai *k* untuk *k-means*, nilai *pruning* untuk *decision tree.*
   1.3 Nilai parameter evaluasi adalah nilai ambang batas *(threshold)* yang bisa diterima.
   1.4 Yang dimaksud dengan *tools* pemodelan di antaranya perangkat lunak *data science* di antaranya: *rapid miner, weka*, atau *development* untuk bahasa pemrograman tertentu seperti *python* atau R.

Phases



Generic Tasks
Specialized Tasks
(Process Instances)

a visual guide to CRISP-DM methodology

SOURCE CRISP-DM 1.0
http://www.crisp-dm.org/download.htm
DESIGN Nicole Leaper
http://www.nicoleleaper.com

Definisi Kursus

Pelatihan ini menjelaskan regresi dan bagaimana membangun model (regresi), yaitu:
a. menyiapkan parameter model
b. menggunakan tools pemodelan

Selanjutnya menjelaskan algoritma dan menggunakan Regresi Non-Linier, dan performansi regresi dengan Python dan Scikit-learn.

Selain itu pembahasan tambahan adalah mengenai klasifikasi.

Capaian Pembelajaran

Peserta dapat menjelaskan, menyiapkan, dan mengimplementasikan model regresi dengan algoritma: Regresi Polinomial dan Non-linier. Beserta pengukuran performansinya menggunakan Python dan Scikit-learn.

Peserta juga mendapatkan pemahaman mengenai algoritma untuk melakukan klasifikasi.

Tujuan Pembelajaran

Peserta mempelajari pengertian, cara menyiapkan, dan cara implementasi model regresi dengan algoritma Regresi Polinomial dan Non-linier. Beserta pengukuran performansinya menggunakan Python dan Scikit-learn.

Peserta mendapatkan pembahasan tambahan mengenai algoritma untuk melakukan klasifikasi.

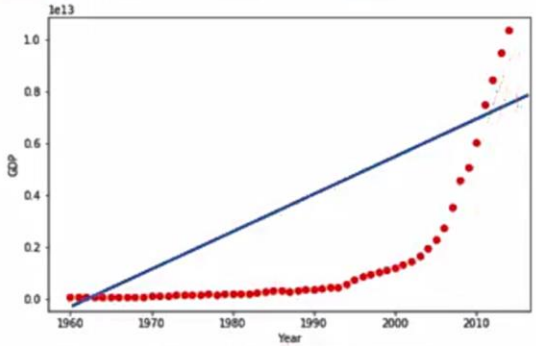
Referensi

- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. and Vanderplas, J., 2011. Scikit-learn: Machine learning in Python. the Journal of machine Learning research, 12, pp.2825-2830.
- Varoquaux, G., Buitinck, L., Louppe, G., Grisel, O., Pedregosa, F. and Mueller, A., 2015. Scikit-learn: Machine learning without learning the machinery. *GetMobile: Mobile Computing and Communications*, *19*(1), pp.29-33.
- https://scikit-learn.org/stable/index.html

Regresi Non Linier

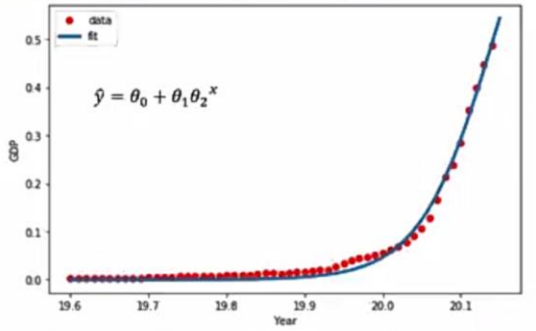
Mengapa Regresi Non-Linier Diperlukan?

	Year	Value
0	1960	5.918412e+10
1	1961	4.955705e+10
2	1962	4.668518e+10
3	1963	5.009730e+10
4	1964	5.906225e+10
5	1965	6.970915e+10
6	1966	7.587943e+10
7	1967	7.205703e+10
8	1968	6.999350e+10
9	1969	7.871882e+10
...	.....	.....



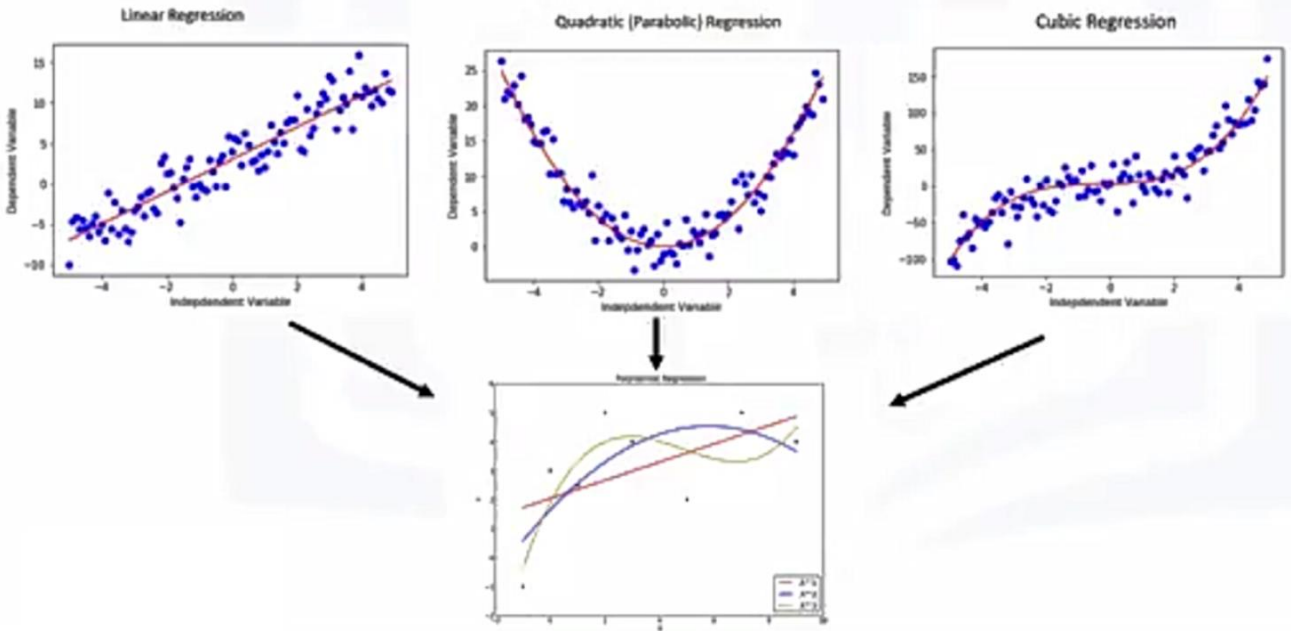
- Tidak setiap data menunjukkan hubungan linier
- Sehingga error akan besar ketika dipaksakan menggunakan model linier

	Year	Value
0	1960	5.918412e+10
1	1961	4.955705e+10
2	1962	4.668518e+10
3	1963	5.009730e+10
4	1964	5.906225e+10
5	1965	6.970915e+10
6	1966	7.587943e+10
7	1967	7.205703e+10
8	1968	6.999350e+10
9	1969	7.871882e+10
...	.....	.....



- Penggunaan model non-linier tampak mempunyai error yang lebih kecil

Tipe Regresi



Regresi Polinomial

- Beberapa data yang berbentuk kurva dapat dimodelkan dengan regresi linier
- Contoh:

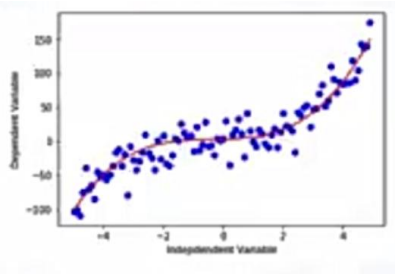
$$\hat{y} = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$$

- Model regresi polinomial dapat ditransformasikan menjadi model regresi linier.

$$x_1 = x$$
$$x_2 = x^2$$
$$x_3 = x^3$$

$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$ -> dapat diselesaikan dengan least squares

Regresi linier variabel jamak



Regresi Non-Linier

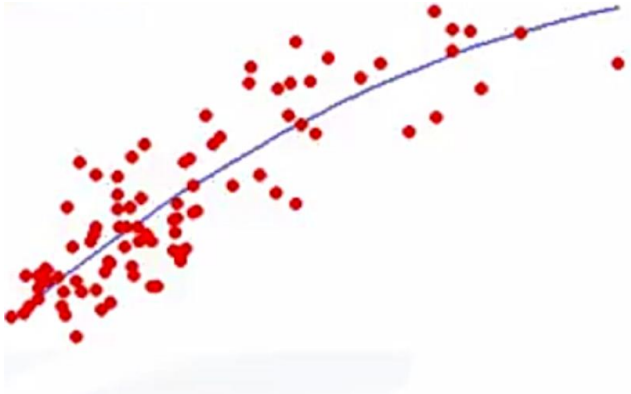
- Memodelkan hubungan tidak linier antara variabel tak bebas dengan himpunan variabel bebas
- ŷ berupa fungsi non-linier dari parameter θ dan fitur x.

$$\hat{y} = \theta_0 + \theta_2{}^2 x$$

$$\hat{y} = \theta_0 + \theta_1 \theta_2{}^x$$

$$\hat{y} = \log(\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3)$$

$$\hat{y} = \frac{\theta_0}{1 + \theta_1{}^{(x-\theta_2)}}$$



Regresi Linier atau Non-Linier?

Cara untuk mengetahui apakah permasalahan cocok diselesaikan dengan regresi linier atau non linier

- Pengamatan visual atas data (visualisasi)
- Pengamatan akurasi hasil pemodelan

Cara untuk memodelkan data apabila visualisasi mengindikasikan non-linier

- Regresi polinomial
- Regresi non-linier
- Transformasi data non-linier menjadi linier

Lab

- Jalankan file Jupyter Notebook untuk Regresi Non Linier

Support Vector Regression

Support Vector Regression

- SVR memberi fleksibilitas untuk menentukan seberapa besar kesalahan yang dapat diterima dalam model dan akan menemukan garis yang sesuai (atau hyperplane dalam dimensi yang lebih tinggi) agar sesuai dengan data.
- Berbeda dengan Least Square biasa, fungsi tujuan SVR adalah untuk meminimalkan koefisien — lebih khusus lagi, *l2*-norm vektor koefisien — bukan *squared error*.

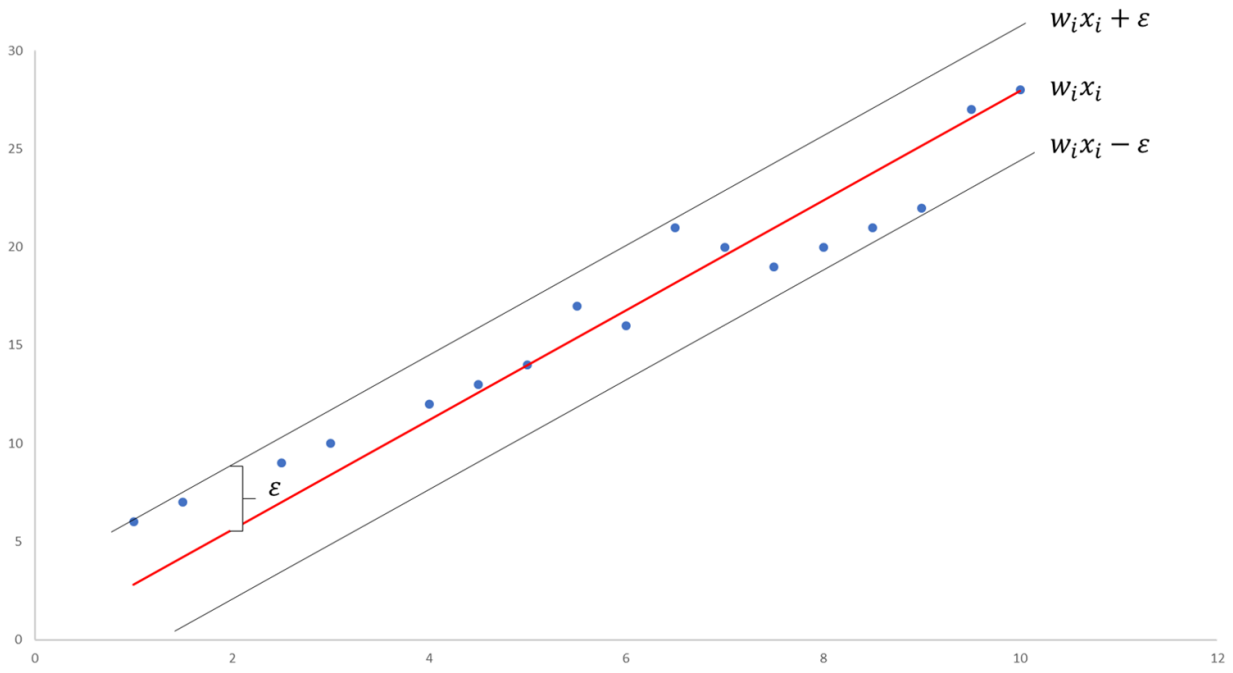
Support Vector Regression

Minimize

$$MIN\ \frac{1}{2}||w||^2$$

Constraint

$$|y_i - w_i x_i| \leq \varepsilon$$



Konsep SVR adalah dengan meminimalkan fungsi objektif yakni koefisien dengan konstrain margin sebagaimana tampak secara visual pada Gambar 15.

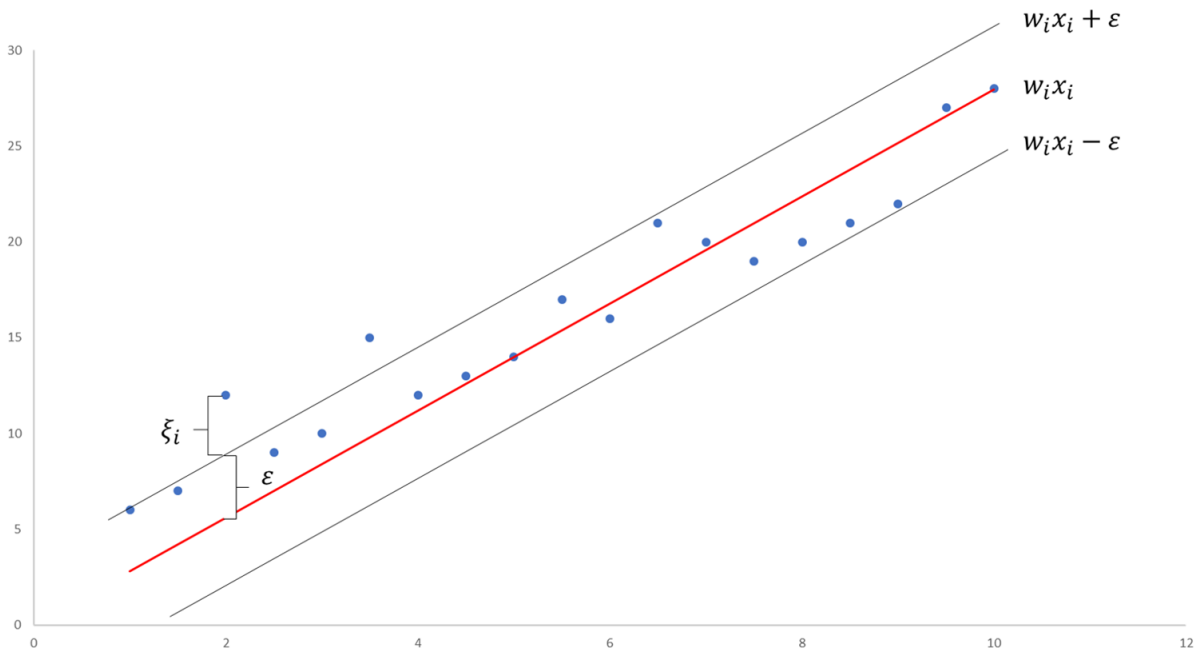
Support Vector Regression with Slack Variable

Minimize

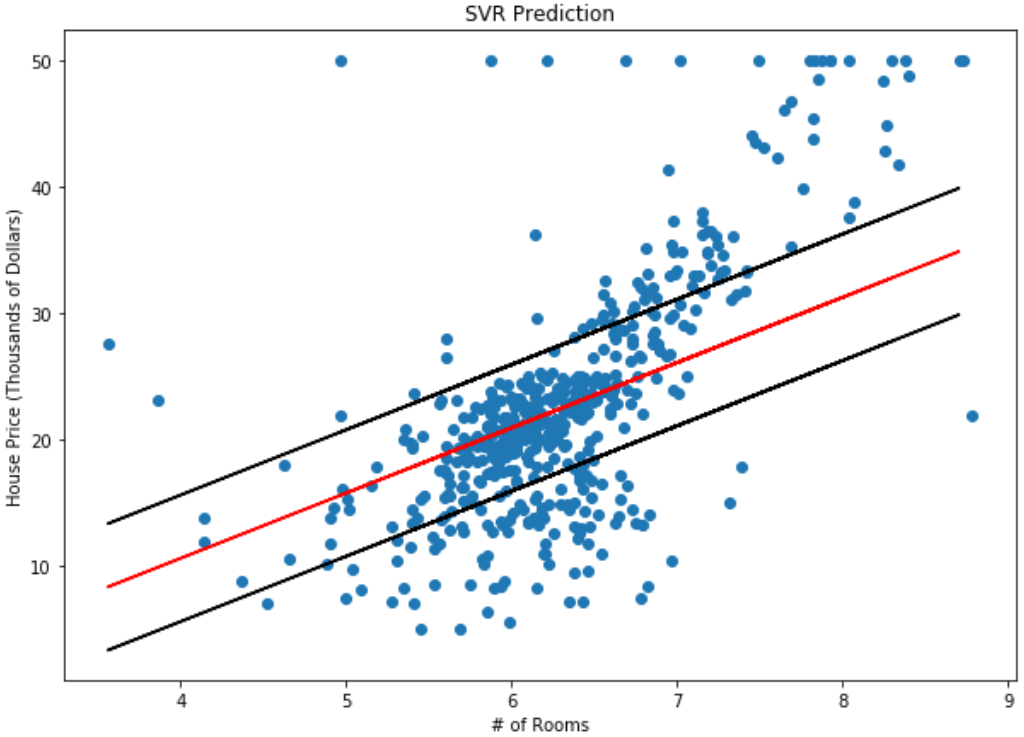
$$MIN\ \frac{1}{2}\left|\left|w\right|\right|^2 + C\sum_{i=1}^{n}|\xi_i|$$

Constraint

$$|y_i - w_i x_i| \leq \varepsilon + |\xi_i|$$

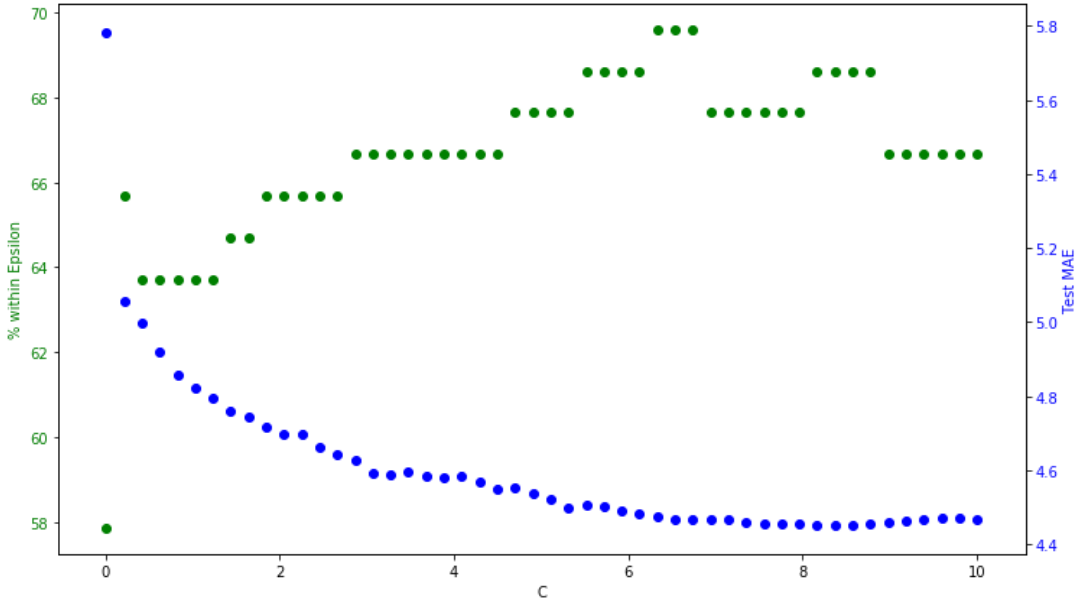


Efek C



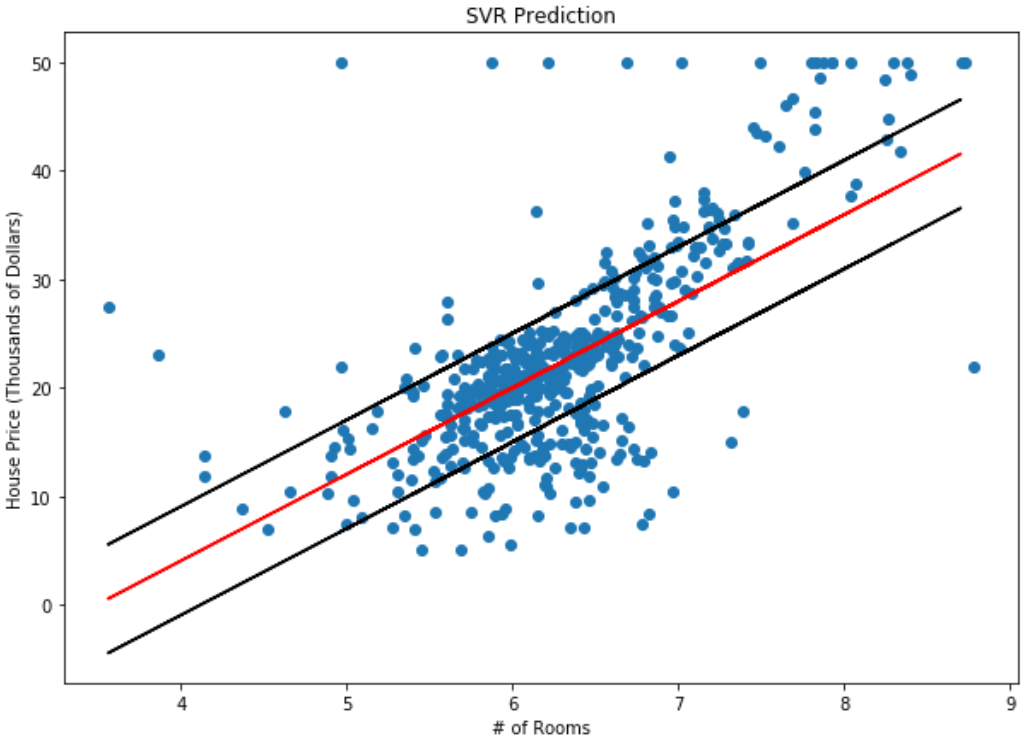
SVR Prediction of Boston Housing Prices with ϵ=5, C=1.0

Mencari C Terbaik



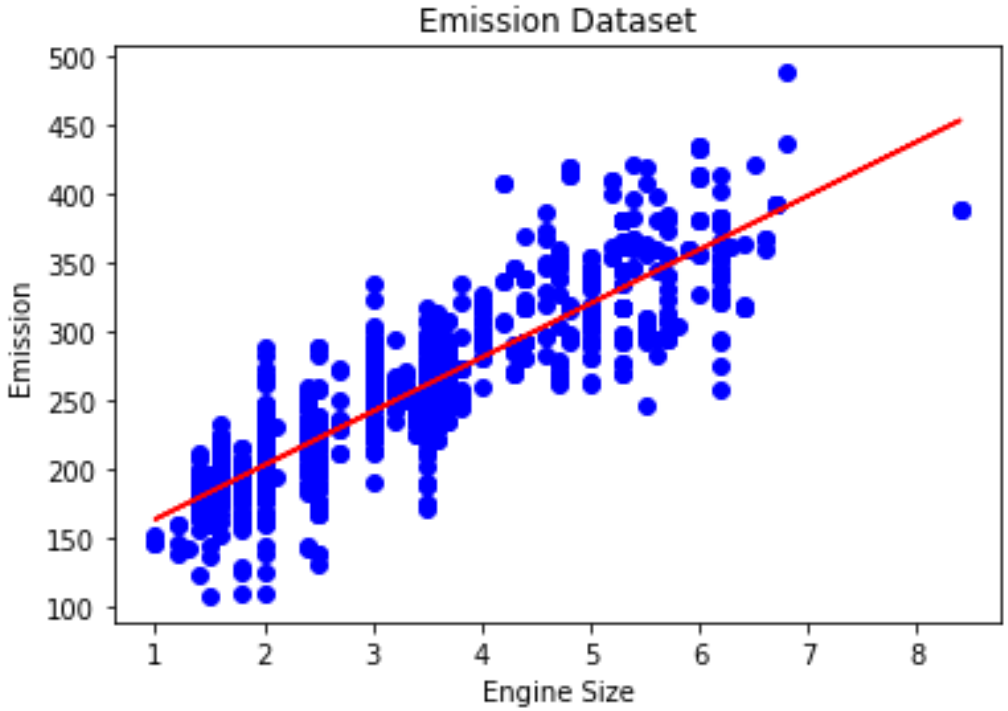
GridSearch for C

Efek C



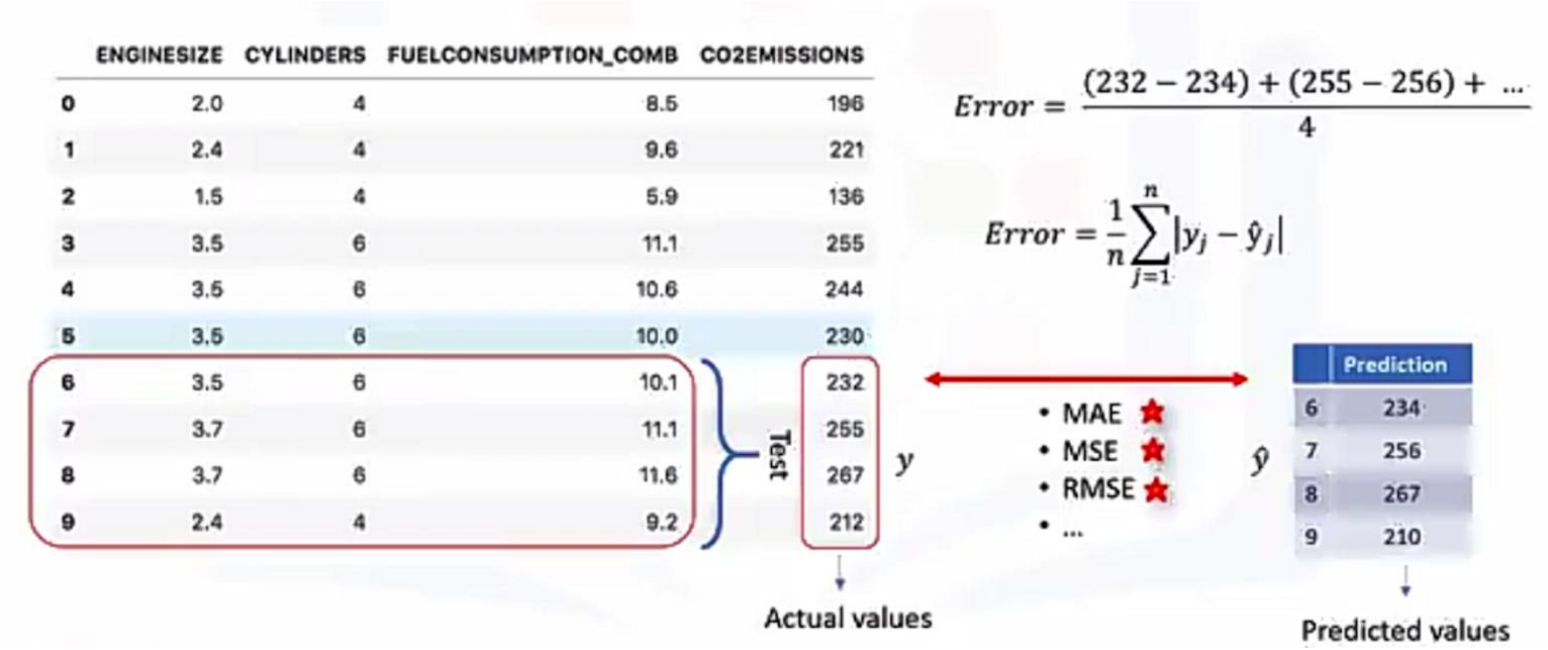
SVR Prediction of Boston Housing Prices with ϵ=5, C=6.13

Aplikasi SVR pada Regresi Sederhana Emission Dataset



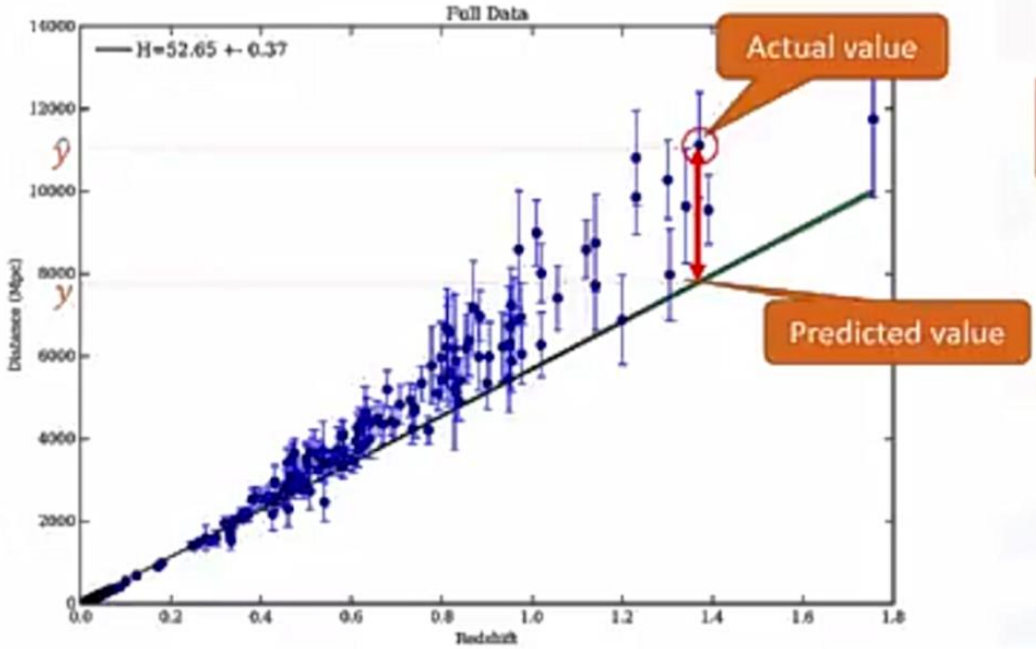
Metriks Evaluasi

Error



- Metrik evaluasi yang paling umum digunakan adalah Error yang terdiri dari berbagai formula antara lain Mean Absolute Error sebagaimana pada Gambar 23.
- Selain itu terdapat Mean Squared Error, Root Mean Squared Error, dan lainnya.

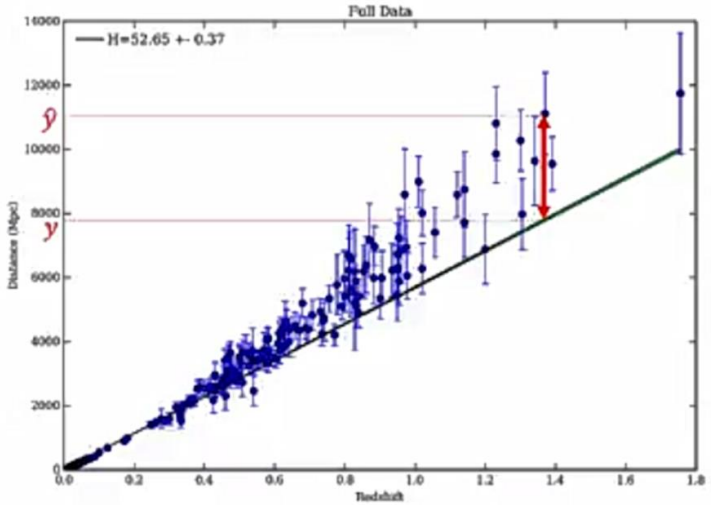
Error dari Model



Error: measure of how far the data is from the fitted regression line.

- Error didefinisikan sebagai ukuran sejauh mana data aktual terhadap garis regresi yang sudah dicocokan sebagaimana diilustrasikan pada Gambar 24.
- Garis regresi tersebut memuat nilai prediksi sedangkan titik-titik biru yang ada adalah nilai aktualnya.

Error dari Model



$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}$$

$$RAE = \frac{\sum_{j=1}^{n}|y_j - \hat{y}_j|}{\sum_{j=1}^{n}|y_j - \bar{y}|}$$

$$RSE = \frac{\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}{\sum_{j=1}^{n}(y_j - \bar{y})^2}$$

$$R^2 = 1 - RSE$$

- MSE, MAE, RMSE lebih disukai digunakan untuk membandingkan kinerja antara model regresi yang berbeda.
- Menggunakan MSE jika nilainya tidak terlalu besar atau menggunakan MAE saat tidak ingin menghukum kesalahan prediksi yang besar
- R-Square dan Adjusted R-Square lebih baik digunakan untuk menjelaskan kesesuaian model. Harus selalu ada keseimbangan karena nilai R2 yang sangat rendah berarti model underfitting sedangkan nilai R2 yang sangat tinggi berarti model overfitting.

Lab

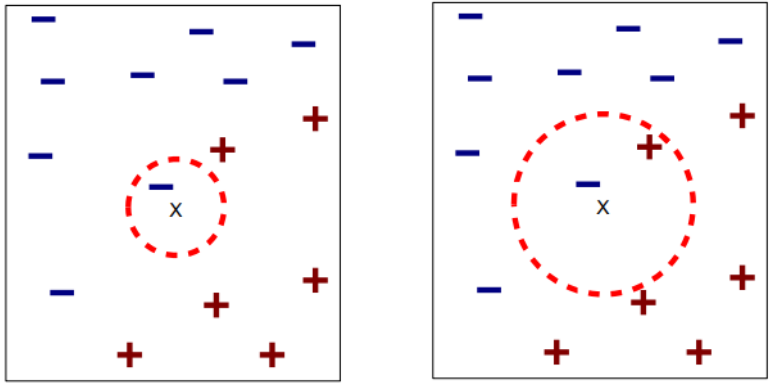
- Jalankan file Jupyter Notebook untuk Support Vector Regression

Klasifikasi

Outline

- **Membangun Model Klasifikasi :**
  - Algoritma klasifikasi yang diimplementasi menggunakan library, yaitu: k-NN, Decision Tree, Naïve Bayes, Support Vector Machine
  - Matriks Performansi Klasifikasi

k – Nearest Neighbor (k-NN) Classifier



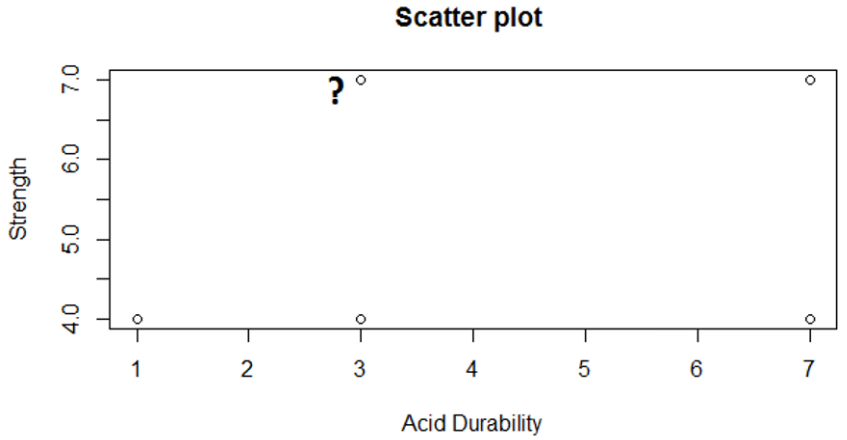
(a) 1-nearest neighbor (b) 2-nearest neighbor

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	?

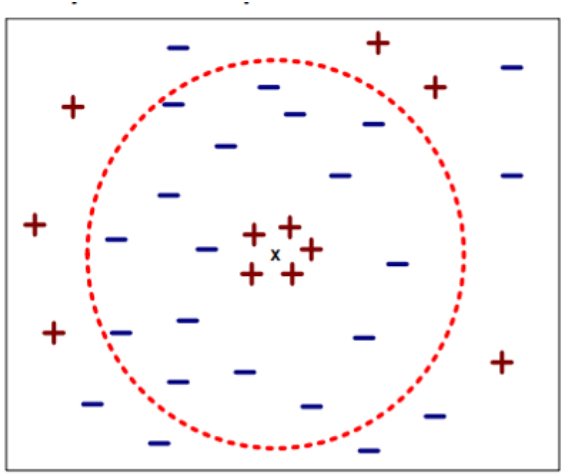
Contoh kasus (lanjutan)

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 neighbor 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*

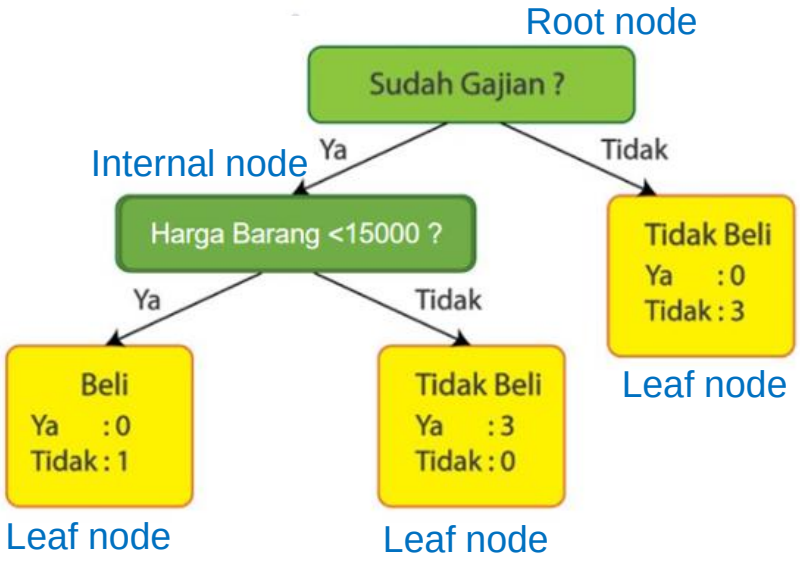
Hands On (lanjutan)

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 neighbors
```

Decision Tree Classifier



- Membangun *decision tree* (pohon keputusan) dari data latih
- *Output* dari pohon keputusan => *rule*. *Rule* tersebut digunakan untuk klasifikasi
- Pohon keputusan, terdiri dari :
  - Root Node
  - Internal Node
  - Leaf Node

Decision Tree Classifier

Kebutuhan Primer	Sudah Gajian	Harga Barang	Beli
Ya	Ya	7000	Tidak
Ya	Tidak	12000	Tidak
Tidak	Ya	18000	Ya
Tidak	Ya	35000	Ya
Ya	Ya	38000	Ya
Ya	Tidak	50000	Tidak
Tidak	Tidak	83000	Tidak

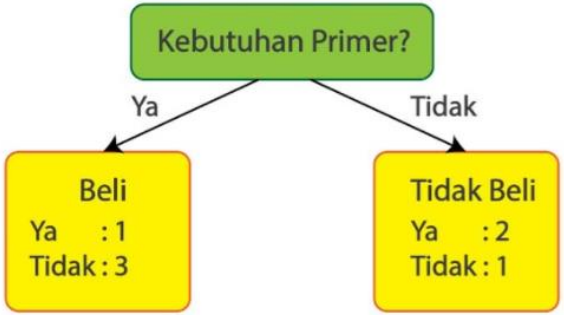
- Data latih terdiri dari 7 data. Data terdiri dari 3 atribut.
- Terdapat 2 kelas : Beli dan Tidak Beli

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

- Menghitung Gini Index (GI) untuk setiap atribut
- Menentukan root berdasarkan nilai GI. Root adalah atribut dengan nilai GI terkecil.
- Ulangi langkah 1 dan 2 untuk level berikutnya pada tree hingga nilai GI = 0.

Gini Index/Impurity untuk Atribut "Kebutuhan Primer"

Kebutuhan Primer	Sudah Gajian	Harga Barang	Beli
Ya	Ya	7000	Tidak
Ya	Tidak	12000	Tidak
Tidak	Ya	18000	Ya
Tidak	Ya	35000	Ya
Ya	Ya	38000	Ya
Ya	Tidak	50000	Tidak
Tidak	Tidak	83000	Tidak



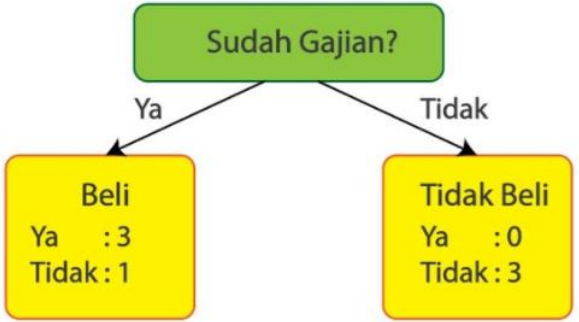
Gini Impurity
$= 1 - (1 / 4)^2 - (3 / 4)^2$
$= 0.38$

Gini Impurity
$= 1 - (2 / 3)^2 - (1 / 3)^2$
$= 0.44$

Total Gini Impurity
$= (4/7) * 0.38 + (3/7) * 0.44$
$= 0.4$

Gini Index/Impurity untuk Atribut "Sudah Gajian"

Kebutuhan Primer	Sudah Gajian	Harga Barang	Beli
Ya	Ya	7000	Tidak
Ya	Tidak	12000	Tidak
Tidak	Ya	18000	Ya
Tidak	Ya	35000	Ya
Ya	Ya	38000	Ya
Ya	Tidak	50000	Tidak
Tidak	Tidak	83000	Tidak



Gini Impurity
$= 1 - (3 / 4)^2 - (1 / 4)^2$
$= 0.38$

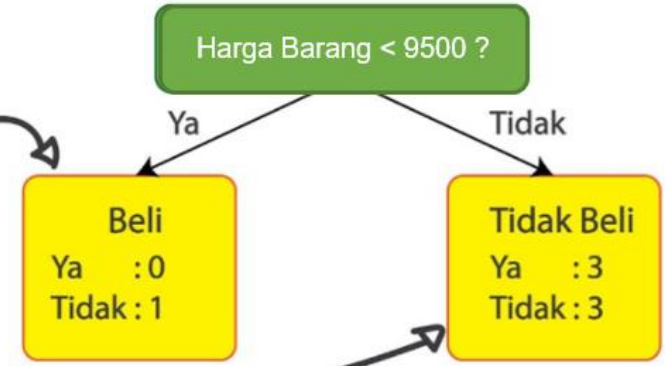
Gini Impurity
$= 1 - (0 / 3)^2 - (3 / 3)^2$
$= 0$

Total Gini Impurity
$= (4/7) * 0.38 + (3/7) * 0$
$= 0.22$

Gini Index/Impurity untuk Atribut "Harga Barang"

rata-rata dari 7000 dan 12000

Harga Barang	Beli
7000	Tidak
9500	
12000	Tidak
15000	
18000	Ya
26500	
35000	Ya
36500	
38000	Ya
44000	
50000	Tidak
66500	
83000	Tidak



Gini Impurity
$= 1 - (0 / 1)^2 - (1 / 1)^2$
$= 0$

Gini Impurity
$= 1 - (3 / 6)^2 - (3 / 6)^2$
$= 0.5$

Total Gini Impurity untuk "Harga Barang < 9500?"
$= (1/7) * 0 + (6/7) * 0.5$
$= 0.43$

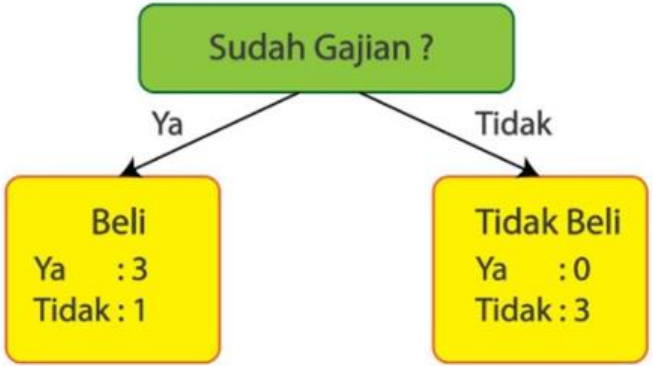
Gini Index/Impurity untuk Atribut "Harga Barang"

Harga Barang	Beli
7000	Tidak
12000	Tidak
18000	Ya
35000	Ya
38000	Ya
50000	Tidak
83000	Tidak

	Gini Impurities
9500	0.43
15000	0.34
26500	0.48
36500	0.48
44000	0.34
66500	0.43

Penentuan Root

- Gini Index “Kebutuhan Primer” : 0.44
- **Gini Index “Sudah gajian” : 0.22**
- Gini Index “Harga barang < 15000” : 0.34
- Atribut yang terpilih sebagai root adalah ”Sudah Gajian”
- Dilakukan proses perhitungan Gini Index bagi left node untuk attribut ”Kebutuhan Primer” dan ”Sudah Gajian”

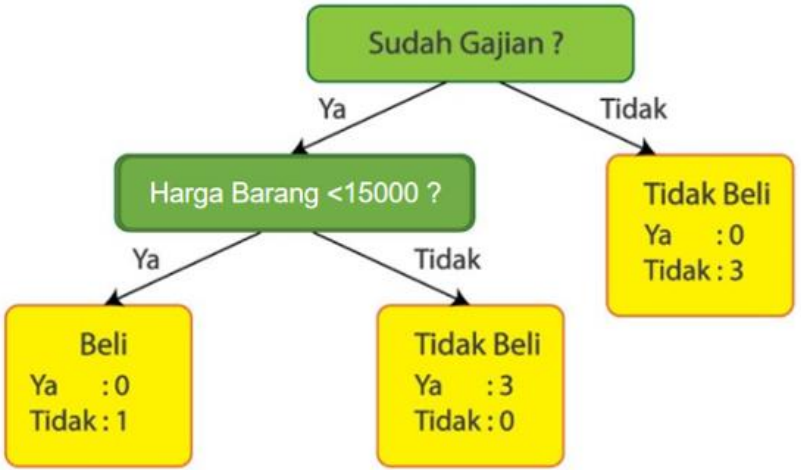


Leaf node

Merupakan *internal node* sehingga memerlukan perhitungan *Gini Index*

Decision Tree Classifier final untuk data latih

Kebutuhan Primer	Sudah Gajian	Harga Barang	Beli
Ya	Ya	7000	Tidak
Ya	Tidak	12000	Tidak
Tidak	Ya	18000	Ya
Tidak	Ya	35000	Ya
Ya	Ya	38000	Ya
Ya	Tidak	50000	Tidak
Tidak	Tidak	83000	Tidak



Hands On

```
from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics

dt = DecisionTreeClassifier(
  max_depth = None,
  min_samples_split = 2
)

dt.fit(X_train, y_train)
y_pred = dt.predict(X_test)
score = metrics.accuracy_score(y_test, y_pred)
print("Akurasi dengan menggunakan Decision Tree: ", score)
```

Decision tree classifier

Proses training data latih, prediksi data uji, perhitungan akurasi

```
Akurasi dengan menggunakan Decision Tree:  0.9555555555555556
```

Output akurasi yang dihasilkan oleh *Decision Tree classifier*

Hands On (lanjutan)

Parameter	Keterangan	Contoh Nilai
max_depth	Maksimum kedalaman tree yang dibentuk	- Bilangan Integer (1,2,3,4,….) - Nilai default : None (jika None maka tree akan terus mendalam sampai seluruh Gini Impurity = 0)
min_samples_split	Minimum sampel yang diperlukan agar node dapat ‘split’	- Bilangan Integer (1,2,3,4,….) - Nilai default : 2

```
# Import Decision Tree Classifier dari sklearn
from sklearn.tree import DecisionTreeClassifier

dt = DecisionTreeClassifier(
      max_depth=4, # menentukan kedalaman maksimum tree = 4
      min_samples_split = 2 # menentukan 2 minimal sampel agar node bisa split
    )
```

Kelebihan dan Kekurangan Decision Tree Classifier

- Pohon keputusan dapat memberikan penjelasan dari proses klasifikasi yang dilakukan berupa rule yang dihasilkan
- Proses pembangunan pohon keputusan pada data numerik menjadi lebih rumit dan memungkinkan terdapat informasi yang hilang
- Pohon keputusan dapat tumbuh menjadi sangat kompleks pada data yang rumit.

Naïve Bayes Classifier

- Membangun probabilistik model untuk klasifikasi
  - Discriminative Model

$$P(C \mid \mathbf{X}) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$$

  - Generative Model

$$P(\mathbf{X} \mid C) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$$

- MAP Classification Rule
  - MAP : Maximum A Posterior
  - Assign $\boldsymbol{x}$ to c^* if $P(C = c^* \mid \mathbf{X} = \mathbf{x}) > P(C = c \mid \mathbf{X} = \mathbf{x}) \quad c \neq c^*, \; c = c_1, \cdots, c_L$
- Generative classification dengan MAP rule
  - Menerapkan Bayesian Rule

$$P(C \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid C)P(C)}{P(\mathbf{X})} \propto P(\mathbf{X} \mid C)P(C)$$

Contoh Kasus

- Terdapat 14 data latih. Data memiliki 4 atribut (Outlook, Temperature, Humidity dan Wind)
- Data latih terdiri dari dua kelas (kelas YES dan kelas NO untuk bermain tenis)
- Diperlukan klasifikasi untuk menentukan kelas dari data uji, apakah bermain tenis (YES) atau tidak (NO).

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Contoh Kasus

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Proses Pelatihan

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/14$

Contoh Kasus

Proses Pelatihan

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/14$

- Data baru,

  $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)

- Look up tabel

  $P(\text{Outlook}=Sunny \mid \text{Play}=Yes) = 2/9$
  $P(\text{Temperature}=Cool \mid \text{Play}=Yes) = 3/9$
  $P(\text{Huminity}=High \mid \text{Play}=Yes) = 3/9$
  $P(\text{Wind}=Strong \mid \text{Play}=Yes) = 3/9$
  $P(\text{Play}=Yes) = 9/14$

  $P(\text{Outlook}=Sunny \mid \text{Play}=No) = 3/5$
  $P(\text{Temperature}=Cool \mid \text{Play}==No) = 1/5$
  $P(\text{Huminity}=High \mid \text{Play}=No) = 4/5$
  $P(\text{Wind}=Strong \mid \text{Play}=No) = 3/5$
  $P(\text{Play}=No) = 5/14$

- MAP rule

  $P(Yes \mid \mathbf{x}')$: $[P(Sunny \mid Yes)P(Cool \mid Yes)P(High \mid Yes)P(Strong \mid Yes)]P(\text{Play}=Yes) = 0.0053$
  $P(No \mid \mathbf{x}')$: $[P(Sunny \mid No)\ P(Cool \mid No)P(High \mid No)P(Strong \mid No)]P(\text{Play}=No) = 0.0206$

  Diketahui $P(Yes \mid \mathbf{x}') < P(No \mid \mathbf{x}')$, maka label $\mathbf{x}'$ adalah "*No*".

Hands On

```
[ ]  from sklearn import naive_bayes
     from sklearn import metrics

     nb = naive_bayes.BernoulliNB()

     nb.fit(X_train, y_train)
     y_pred = nb.predict(X_test)
     score = metrics.accuracy_score(y_test, y_pred)
     print("Akurasi dengan menggunakan Naive Bayes: ", score)

Akurasi dengan menggunakan Naive Bayes:  0.2222222222222222
```

Naïve Bayes Classifier

Proses training data latih, prediksi data uji, perhitungan akurasi

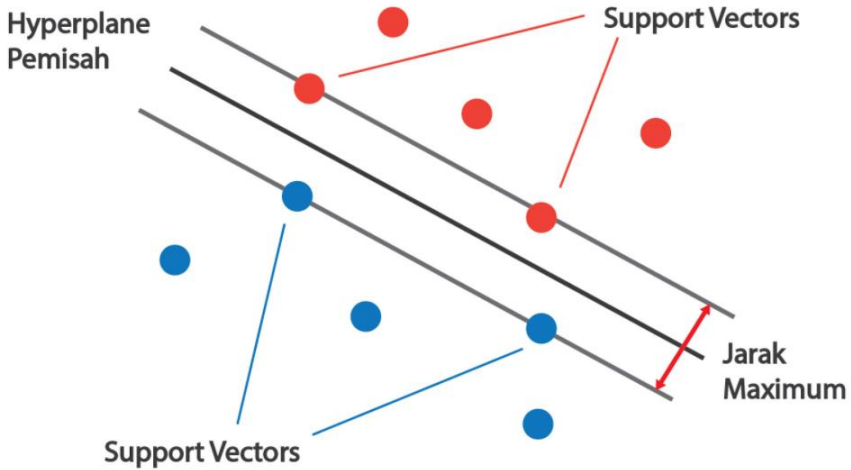
Output akurasi yang dihasilkan oleh *Naïve Bayes Classifier*

Kelebihan dan Kekurangan Naïve Bayes Classifier

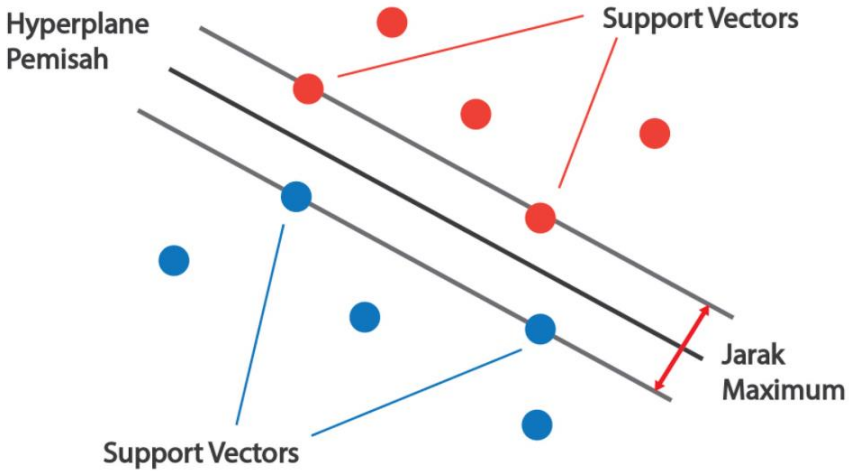
- Naïve Bayes Classifier bekerja lebih baik pada training data yang kecil
- Proses pembangunan Naïve Bayes Classifier pada data numerik menjadi lebih rumit dan memungkinkan terdapat informasi yang hilang
- Pada Naïve Bayes Classifier, diasumsikan bahwa satu fitur dengan fitur yang lain saling independen, hal ini mungkin tidak selalu terjadi pada kasus nyata

Support Vector Machine

- Ide utama dari SVM adalah untuk menemukan hyperplane terbaik yang memisahkan 2 daerah keputusan dengan baik
- Hyperplane adalah sebuah fungsi yang dapat digunakan sebagai pemisah antar kelas



Support Vector Machine



Algoritma SVM :

- Ide Fitur-fitur dari data dipetakan ke dalam ruang dimensi lebih tinggi menggunakan fungsi kernel (linear, polynomial, radial basis function, sigmoid).
- Mencari hyperplane terbaik yang memisahkan data yang telah dipetakan dalam ruang dimensi tinggi.
- Hyperplane terbaik dapat diperoleh dengan memaksimalkan jarak hyperplane dengan titik data terdekat (Support Vectors)

Kernel pada Support Vector Machine

- Fitur-fitur dari data dipetakan ke dalam ruang dimensi tinggi menggunakan fungsi kernel (linear, polynomial, sigmoid, atau radial basis function)
- Dengan menggunakan fungsi kernel, akan dihasilkan fitur-fitur baru yang akan digunakan untuk mengklasifikasi (tidak lagi menggunakan fitur-fitur lama)

Kernel *Linear*

$$K(x_i, x_j) = x_i^T x_j$$

Kernel *Polynomial*

$$K(x_i, x_j) = (\gamma x_i^T x_j + r)^d, \quad \gamma > 0$$

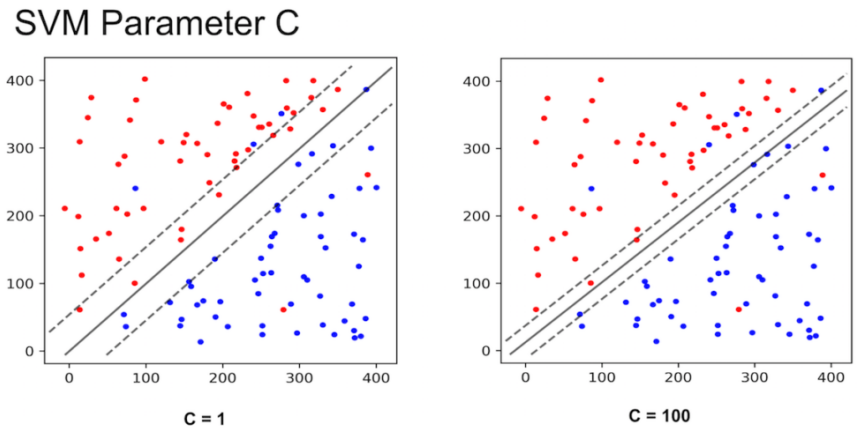
Kernel *Sigmoid*

$$K(x_i, x_j) = \tanh(x_i^T x_j + r)$$

Kernel *Radial Basis Function*

$$K(x_i, x_j) = \exp\left(-\gamma \left|\left|x_i^T - x_j\right|\right|^2\right), \quad \gamma > 0$$

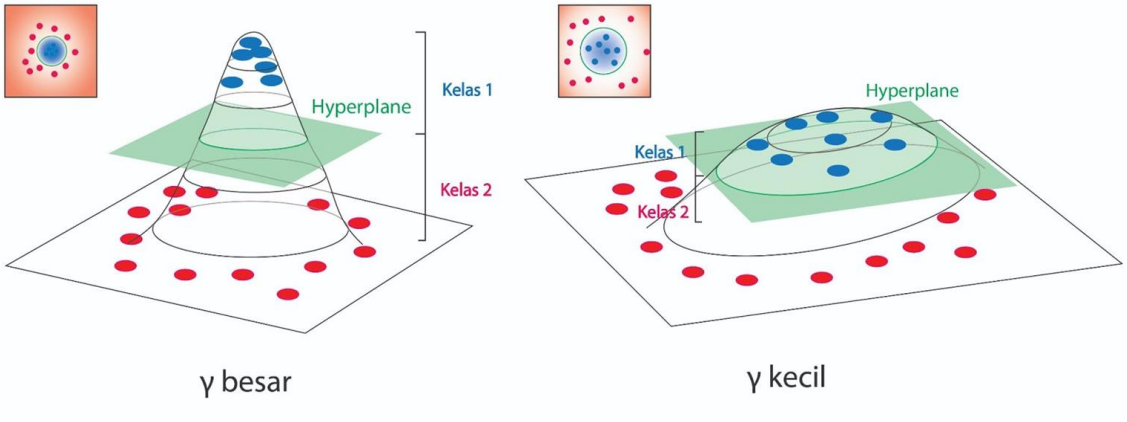
Parameter C pada Support Vector Machine

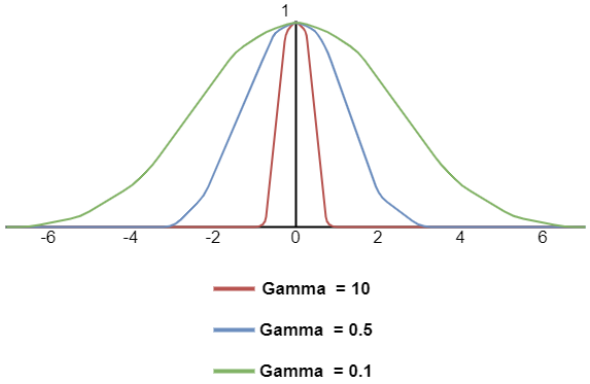


Sumber: https://learnopencv.com/svm-using-scikit-learn-in-python/

- Nilai C yang terbaik harus dicari agar SVM terjadi keseimbangan antara jarak semaksimal dan kesalahan klasifikasi seminimal mungkin.
- Nilai C yang tinggi = model akan menerima lebih banyak penalti ketika model gagal mengklasifikasikan data dengan tepat.
- Nilai C yang rendah = model akan mentolerir data yang salah diklasifikasikan.
- Nilai C yang rendah biasanya baik digunakan pada data yang memiliki banyak noise, sebaliknya nilai C yang tinggi biasanya baik digunakan pada data yang memiliki sedikit noise.

Parameter γ pada Support Vector Machine





Nilai γ yang besar akan menghasilkan kelengkungan yang tajam pada ruang dimensi tinggi. Sebaliknya, nilai γ yang kecil akan menghasilkan ruang dimensi tinggi yang lebih landai.

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

Tools / Lab Online

- Jupyter Notebook
- Google Collabs

Quiz / Tugas

- Quiz dapat diakses melalui https://spadadikti.id/

Terima kasih