Support Vector Machines

Kuliah 6 - STA1382 Teknik Pembelajaran Mesin

Septian Rahardiantoro

Outline

- Pengantar
- Hyperplane?
- Maximal Margin Classifier
- Support Vector Classifier
- Support Vector Machines

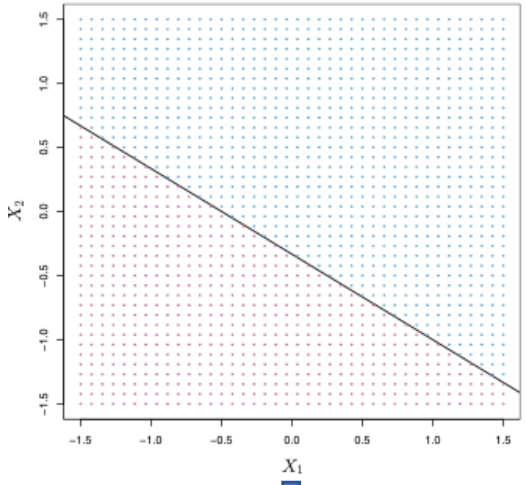
Pengantar

- *Support vector machine* (SVM) merupakan sebuah pendekatan untuk klasifikasi yang dikembangkan dalam komunitas ilmu komputer pada 1990-an dan semakin populer sejak saat itu.

- SVM telah terbukti berkinerja baik dalam berbagai pengaturan, dan sering dianggap sebagai salah satu pengklasifikasi "out of the box" terbaik.

- SVM adalah generalisasi dari pengklasifikasi sederhana dan intuitif yang disebut *marginal margin classifier* dan pengembangannya *support vector classifier*

Ide dasar dan konteks dalam SVM



hyperplane

marginal margin classifier — pengklasifikasi sederhana dan intuitif

support vector classifier — pengembangan dari *marginal margin classifier* yang dapat diterapkan dalam rentang kasus yang lebih luas

support vector machines — Pengembangan lebih lanjut dari *support vector classifier* untuk mengakomodasi batas kelas non-linear.

Hyperplane?

- Dalam ruang berdimensi p, hyperplane adalah subruang bidang datar berdimensi $p - 1$
  - Misalkan dalam 2 dimensi, hyperplane adalah subruang datar satu dimensi → suatu garis lurus
  - Dalam 3 dimensi, hyperplane adalah subruang datar dua dimensi → suatu bidang datar
- Pada ruang berdimensi p, hyperplane didefinisikan sebagai:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p = 0$$

  - untuk setiap $\boldsymbol{X} = (X_1, X_2, \ldots, X_p)^T$
  - Maka untuk 2 dimensi, hyperplane → $\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$ (garis lurus), untuk $\boldsymbol{X} = (X_1, X_2)^T$

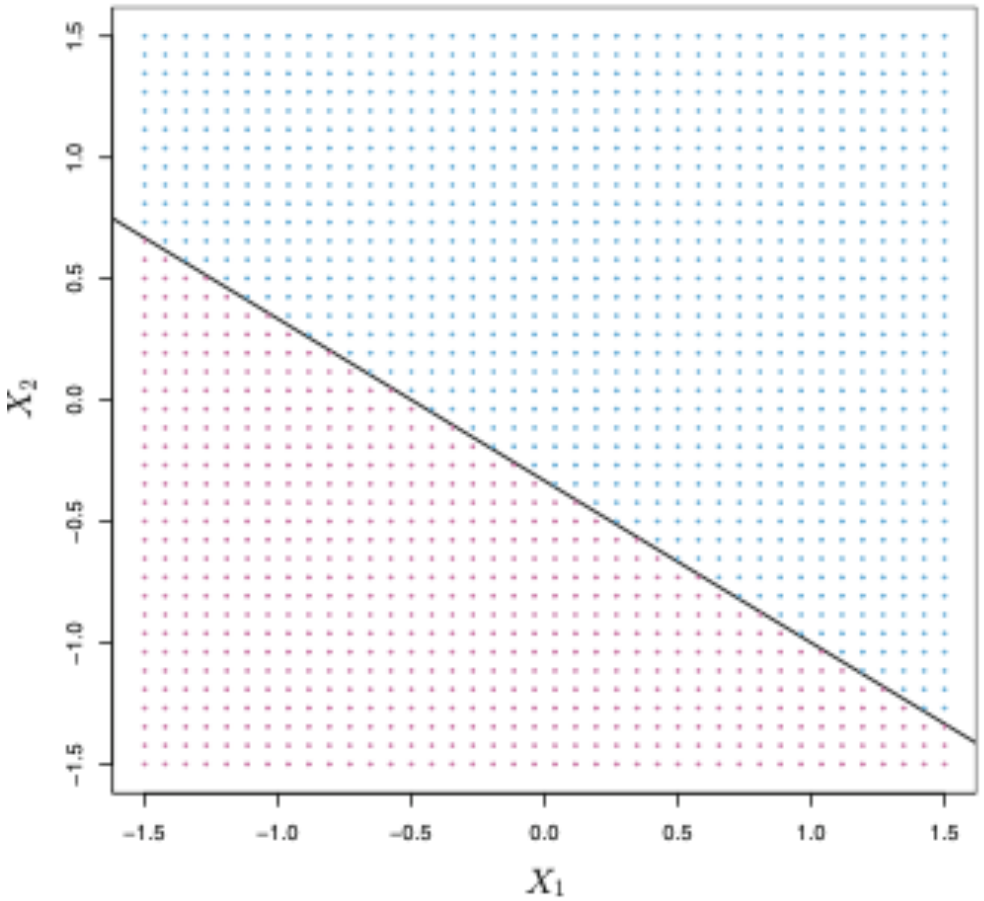
- Pada ruang berdimensi p, anggap:

$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p > 0$ ⟶ $\boldsymbol{X}$ jatuh pada sisi $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p > 0$

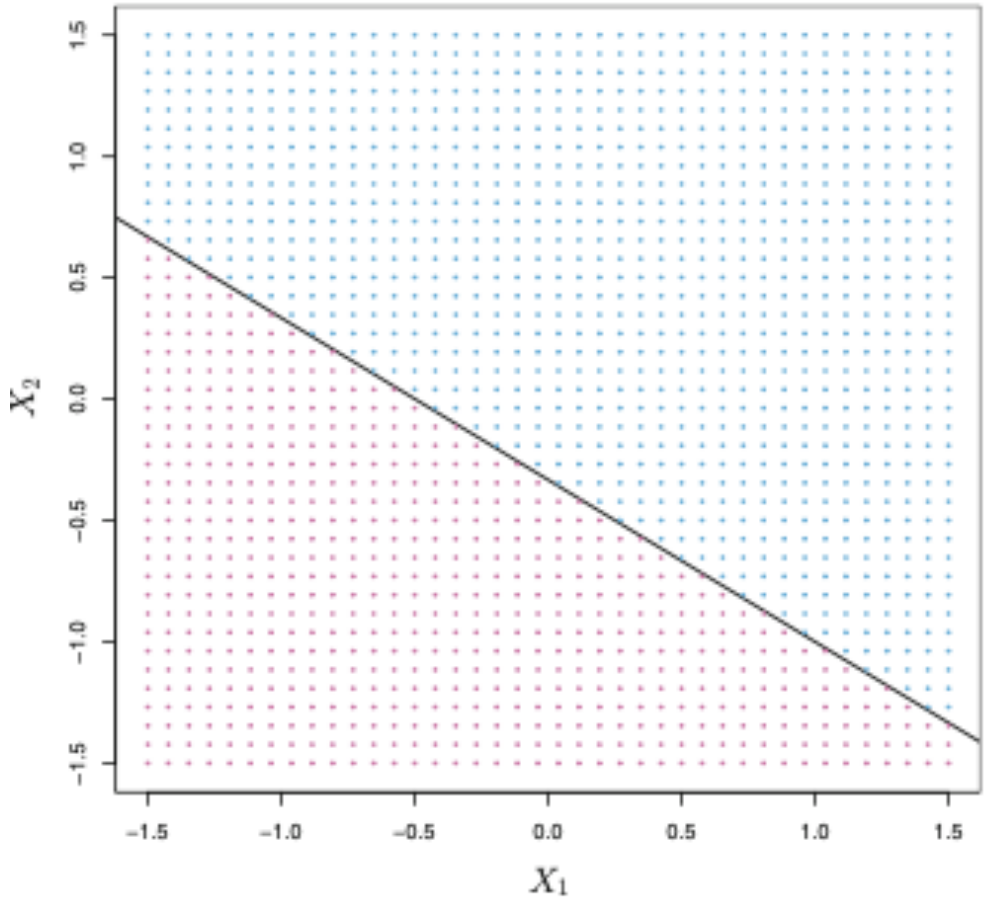
$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p < 0$ ⟶ $\boldsymbol{X}$ jatuh pada sisi yang lainnya

- Jadi kita bisa menganggap hyperplane membagi ruang berdimensi p menjadi dua bagian.

- Seseorang dapat dengan mudah menentukan di sisi mana dari hyperplane suatu titik terletak dengan hanya menghitung tanda $\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$



Contoh sebuah hyperplane dalam ruang dua dimensi



Contoh sebuah hyperplane dalam ruang dua dimensi

Hyperplane $1 + 2X_1 + 3X_2 = 0$

Daerah biru adalah himpunan titik dimana $1 + 2X_1 + 3X_2 > 0$

Daerah ungu adalah himpunan titik dimana $1 + 2X_1 + 3X_2 < 0$

Klasifikasi Menggunakan Pemisahan Hyperplane

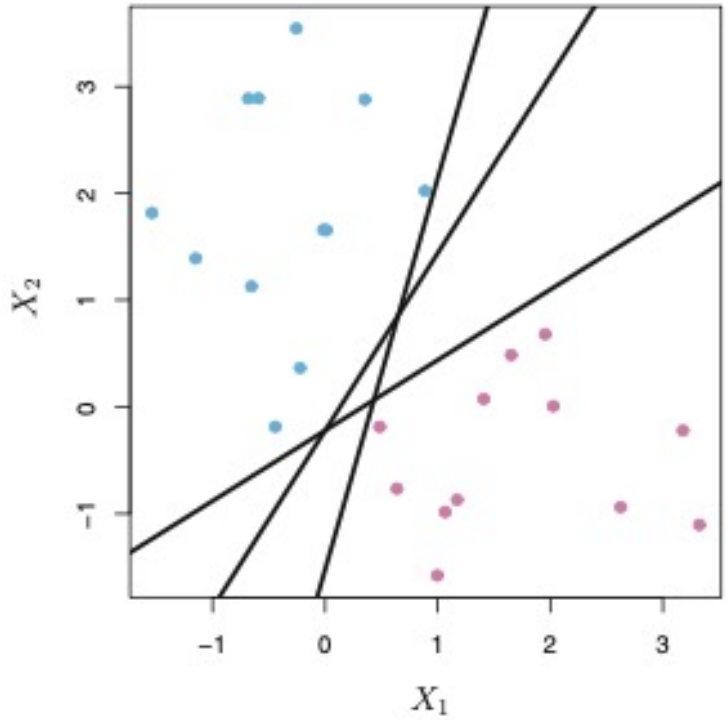
- Sekarang misalkan diketahui matriks data training $\boldsymbol{X}$ berdimensi $n \times p$, yang terdiri dari n observasi dalam ruang dimensi-p

$$\boldsymbol{x}_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, \boldsymbol{x}_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix} \longrightarrow \text{pada kelas } y_1, \ldots, y_n \in \{-1,1\}$$

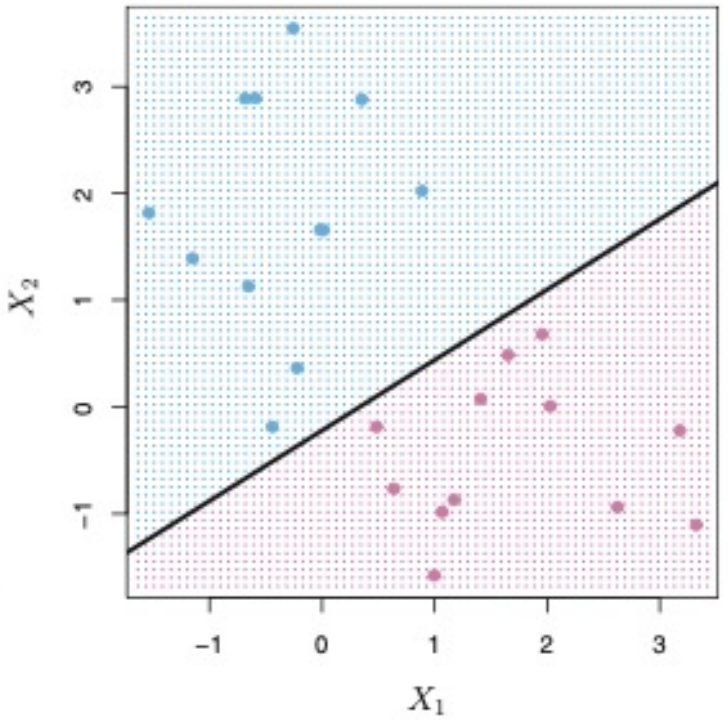
- Tujuannya adalah mengembangkan pengklasifikasi berdasarkan data training yang akan mengklasifikasikan pengamatan data testing dengan benar menggunakan pengukuran fiturnya.

- Sekarang kita akan melihat pendekatan baru yang didasarkan pada konsep memisahkan hyperplane.

- Misalkan dimungkinkan untuk membuat hyperplane yang memisahkan observasi pelatihan dengan sempurna sesuai dengan label kelasnya.



Ada dua kelas pengamatan, ditunjukkan dengan warna biru (1) dan ungu (-1), yang masing-masing memiliki pengukuran pada dua variabel. Tiga hyperplanes yang memisahkan, dari banyak kemungkinan, ditampilkan dalam warna hitam.



Kanan: Sebuah hyperplane pemisah ditampilkan dalam warna hitam. Kisi biru dan ungu menunjukkan aturan keputusan yang dibuat oleh pengklasifikasi berdasarkan hyperplane pemisah ini: observasi uji yang termasuk dalam bagian biru akan diduga ke kelas biru, dan observasi uji yang termasuk dalam bagian ungu akan diduga ke kelas ungu.

- Misalkan terdapat 2 kelas pengamatan: kelas warna biru sebagai $y_i = 1$ dan kelas warna ungu sebagai $y_i = -1$

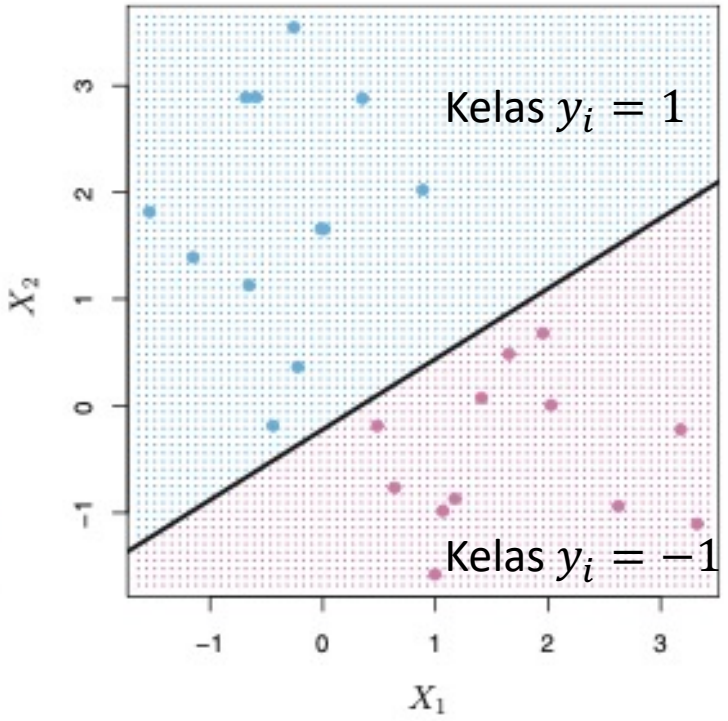
  maka hyperplane pemisah dapat didefinisikan sebagai

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} > 0 \text{ jika } y_i = 1$$
$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} < 0 \text{ jika } y_i = -1$$

- Secara ekuivalen, hyperplane pemisah memiliki properti

$$y_i\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) > 0 \text{ untuk setiap } i = 1,2,3 \ldots, n$$

- Jika terdapat hyperplane pemisah, kita dapat menggunakannya untuk membuat classifier yang sangat alami: observasi data testing diberi kelas tergantung pada sisi mana hyperplane itu berada.



Misalkan dengan pengamatan data testing $\boldsymbol{x}^*$

Maka $\boldsymbol{x}^*$ diklasifikasikan berdasarkan tanda dari

$$f(\boldsymbol{x}^*) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}$$

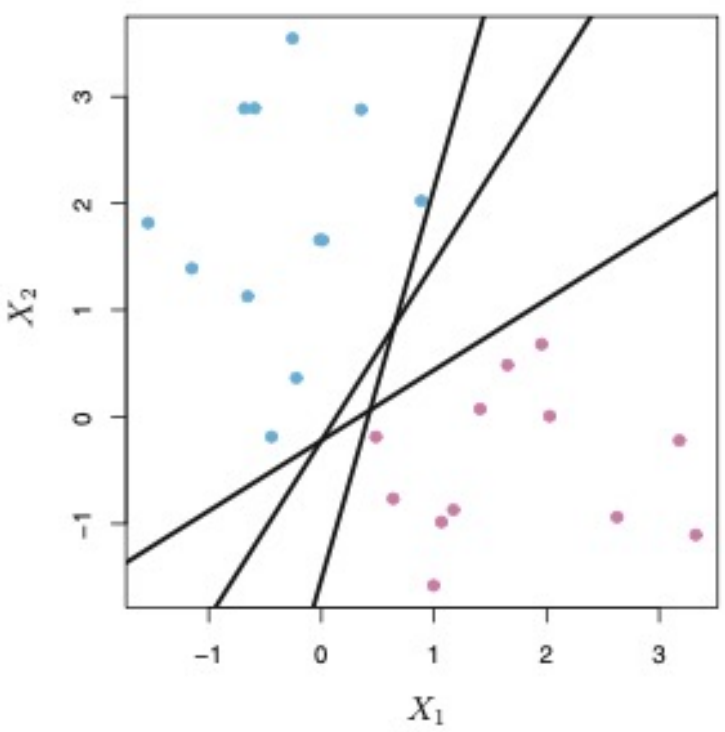
Jika $f(\boldsymbol{x}^*) > 0$, maka $\boldsymbol{x}^*$ diduga masuk ke kelas 1
Jika $f(\boldsymbol{x}^*) < 0$, maka $\boldsymbol{x}^*$ diduga masuk ke kelas -1

Perhatikan, kita juga dapat melihat magnitude (ukuran) dari $f(\boldsymbol{x}^*)$

- Jika $f(\boldsymbol{x}^*)$ jauh dari 0 → $\boldsymbol{x}^*$ berada jauh dari hyperplane → jadi kita bisa percaya diri tentang kelas untuk $\boldsymbol{x}^*$
- Jika $f(\boldsymbol{x}^*)$ mendekati 0 → $\boldsymbol{x}^*$ berada dekat atau disekitar hyperplane → jadi kurang yakin tentang kelas untuk $\boldsymbol{x}^*$

Ide inilah yang nantinya diterapkan pada konsep *maximal margin classifier*

Maximal Margin Classifier



Tiga kemungkinan hyperplanes pemisah

- Untuk membuat pengklasifikasi berdasarkan hyperplane pemisah, kita harus memiliki cara yang masuk akal untuk memutuskan hyperplane pemisah mana yang akan digunakan.
- *Maximal margin hyperplane* (atau *optimal separating hyperplane*) merupakan hyperplane pemisah yang terjauh dari pengamatan data training.
- Yaitu, kita dapat menghitung jarak (tegak lurus) dari setiap observasi data training ke hyperplane pemisah yang diberikan; jarak terkecil adalah jarak minimal dari pengamatan ke hyperplane, dan dikenal sebagai margin.

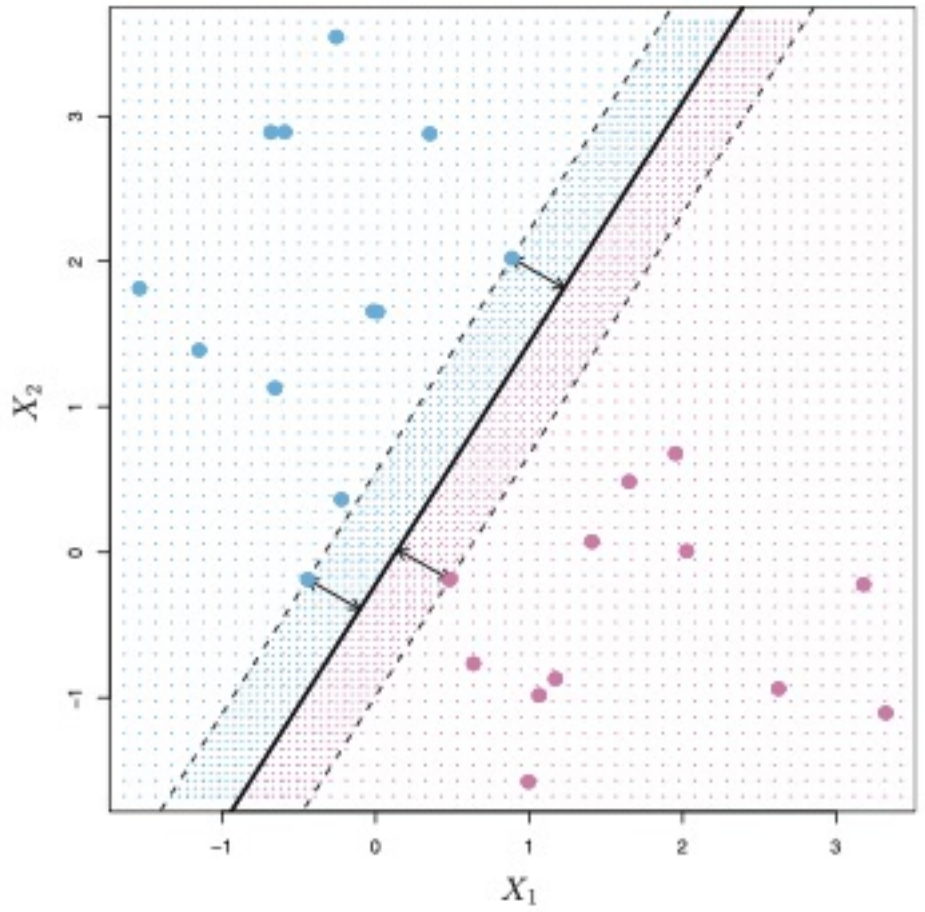
- *Maximal margin hyperplane* adalah hyperplane pemisah yang memiliki margin terbesar—artinya, hyperplane tersebut memiliki jarak minimum terjauh ke pengamatan data training.

- Kemudian dapat diklasifikasikan pengamatan data testing berdasarkan sisi mana dari *maximal margin hyperplane* itu terletak. → Ini dikenal sebagai ***maximal margin classifier***.

- Harapannya, pengklasifikasi yang memiliki margin besar pada data training juga akan memiliki margin besar pada data testing, dan karenanya akan mengklasifikasikan pengamatan data testing dengan benar.

- Meskipun *maximal margin classifier* sering berhasil, hal ini juga dapat menyebabkan overfitting ketika p besar.

- Jika $\beta_0, \beta_1, \beta_2, \dots, \beta_p$ adalah koefisien *maximal margin hyperplane*, maka *maximal margin classifier* mengklasifikasikan pengamatan data testing $\boldsymbol{x}^*$ berdasarkan tanda $f(\boldsymbol{x}^*) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}$



Ada dua kelas pengamatan, ditunjukkan dengan warna biru dan ungu.

- *Maximal margin hyperplane* ditampilkan sebagai garis solid berwarna hitam.
- Margin adalah jarak dari garis solid ke salah satu garis putus-putus.
- Dua titik biru dan titik ungu yang terletak pada garis putus-putus adalah vektor pendukung (*support vectors*), dan jarak dari titik tersebut ke margin ditunjukkan dengan panah.
- Grid ungu dan biru menunjukkan aturan keputusan yang dibuat oleh pengklasifikasi berdasarkan hyperplane pemisah ini.
- *Maximal margin hyperplane* mewakili garis tengah dari "lempengan" terluas yang dapat kita sisipkan di antara kedua kelas.
- Menariknya, *maximal margin hyperplane* bergantung langsung pada *support vectors*, tetapi tidak pada pengamatan lain: pergerakan ke salah satu pengamatan lain tidak akan mempengaruhi hyperplane pemisah, asalkan pergerakan pengamatan tidak menyebabkannya melintasi batas yang ditetapkan oleh margin.

Konstruksi *Maximal Margin Classifier*

- Secara singkat, *maximal margin hyperplane* adalah solusi untuk masalah optimisasi

$$\underset{\beta_0,\beta_1,\ldots,\beta_p}{\text{maximize}}\, M$$

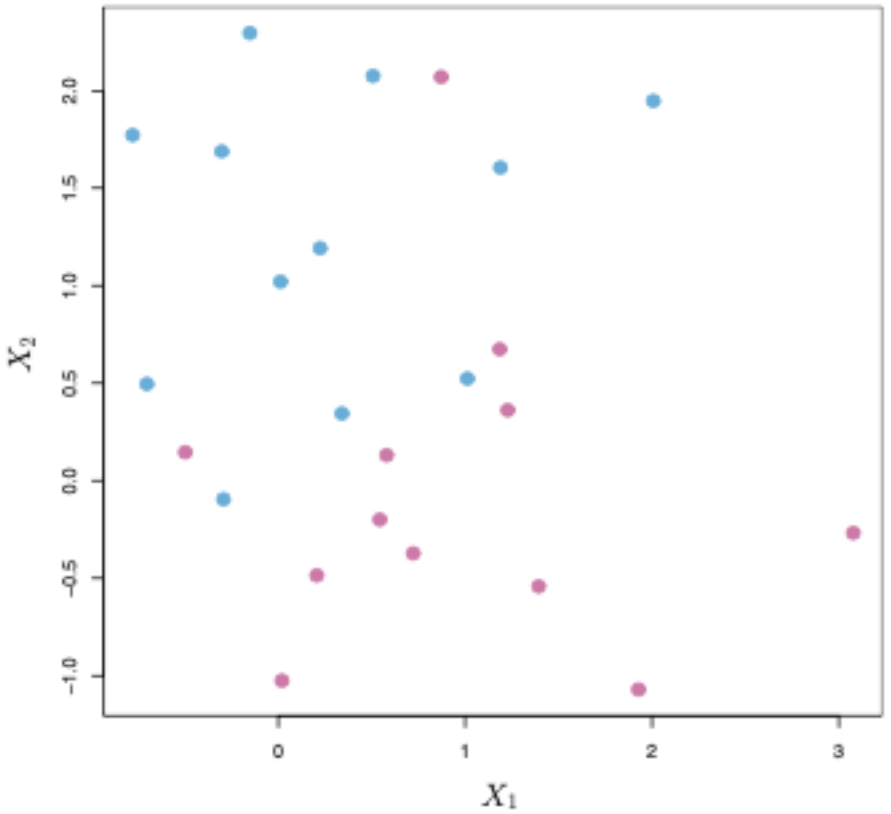
$$\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M \;\; \forall\, i = 1, \ldots, n.$$

Dua kendala ini memastikan bahwa setiap pengamatan berada di sisi yang benar dari hyperplane dan setidaknya berjarak M dari hyperplane.

Oleh karena itu, M merepresentasikan margin hyperplane kita, dan masalah optimisasi memilih $\beta_0, \beta_1, \beta_2, \ldots, \beta_p$ untuk memaksimalkan M. Ini persis definisi *maximal margin hyperplane*!

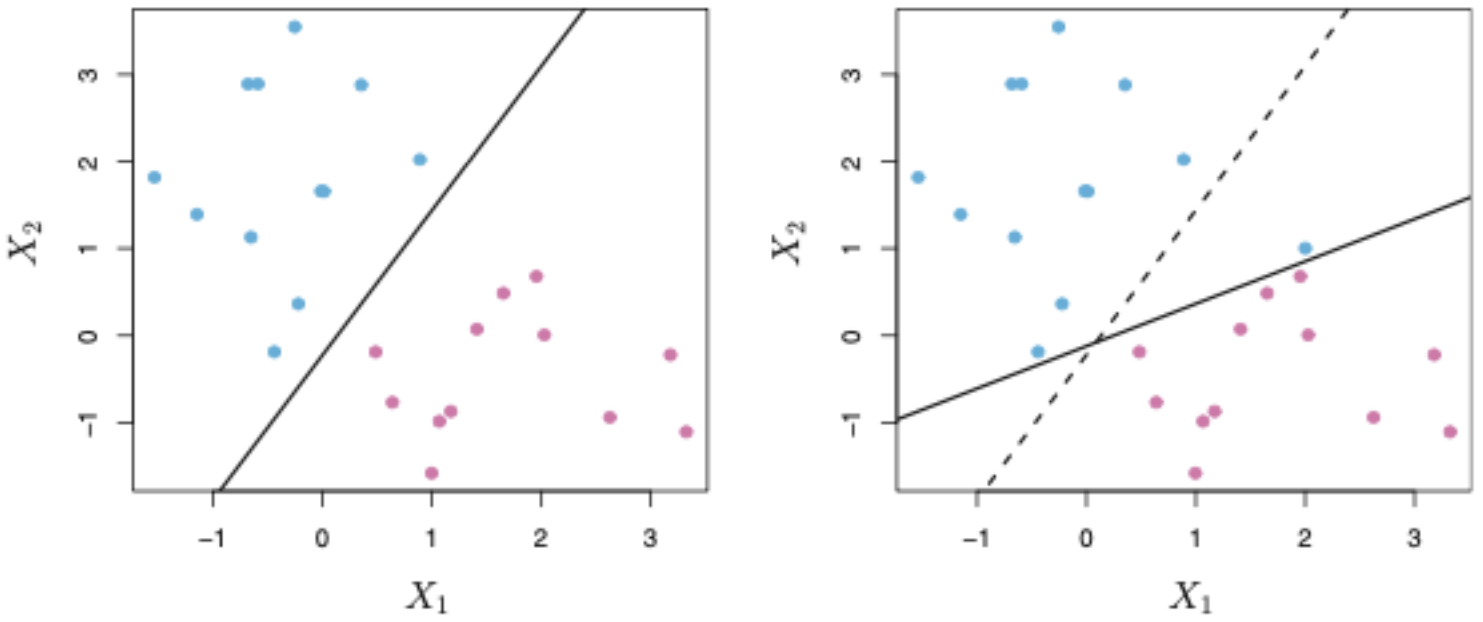
Kasus yang Tidak Dapat Dipisahkan



Ada dua kelas pengamatan, ditunjukkan dengan warna biru dan ungu. Dalam hal ini, kedua kelas tidak dapat dipisahkan oleh hyperplane, sehingga *maximal margin classifier* tidak dapat digunakan.

- *Maximal margin classifier* adalah cara yang sangat alami untuk melakukan klasifikasi, jika terdapat hyperplane pemisah.
- Namun, seperti pada kasus di samping, tidak ada hyperplane pemisah, sehingga tidak ada *maximal margin classifier.*
- Generalisasi dari *maximal margin classifier* ke kasus yang tidak dapat dipisahkan dikenal sebagai *support vector classifier*.

Support Vector Classifier



- Kiri: Dua kelas pengamatan ditampilkan dalam warna biru dan ungu, bersama dengan *maximal margin hyperplane* .

- Kanan: Pengamatan biru tambahan telah ditambahkan, mengarah ke pergeseran dramatis pada *maximal margin hyperplane* yang ditampilkan sebagai garis solid. Garis putus-putus menunjukkan *maximal margin hyperplane* yang diperoleh tanpa adanya titik tambahan ini.

- *Maximal margin hyperplane* sangat sensitif terhadap perubahan dalam pengamatan tunggal menunjukkan bahwa mungkin telah overfit data training.

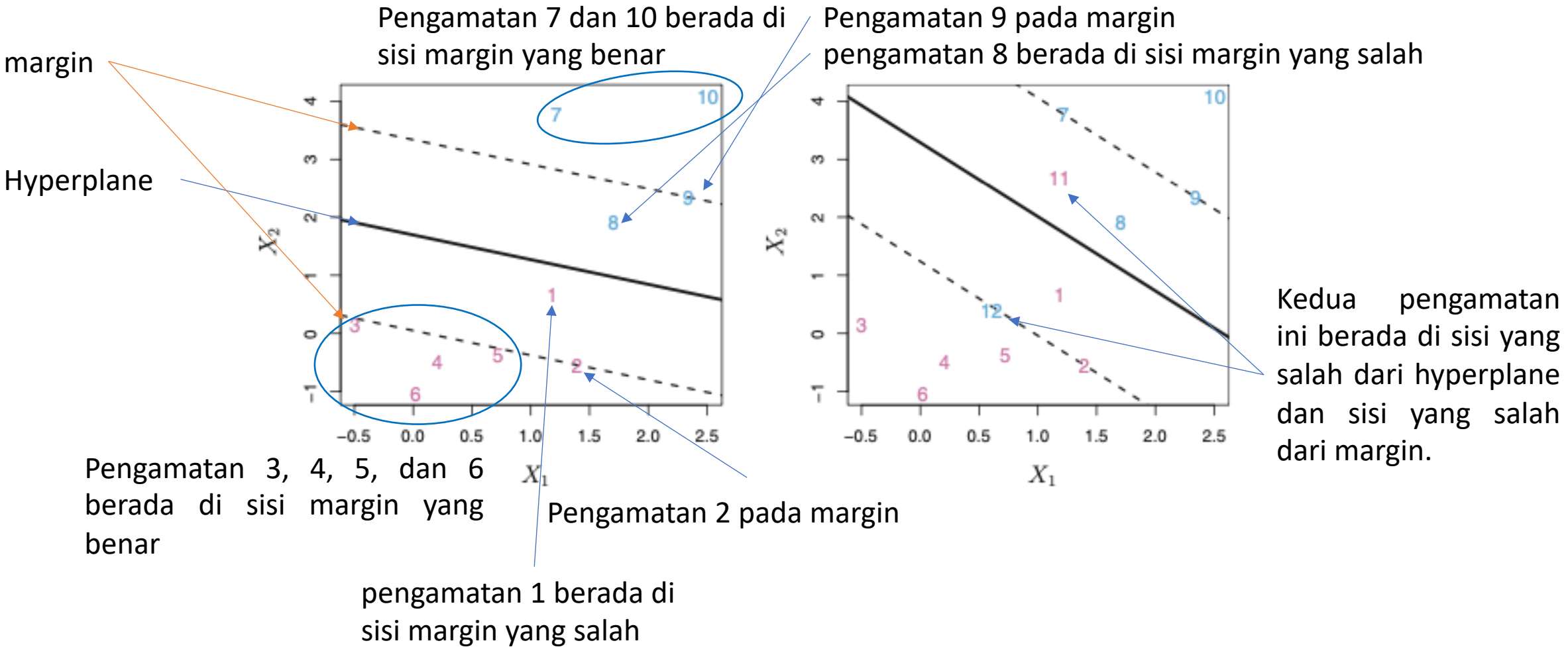
Support Vector Classifier

- Dalam hal ini, kita mungkin bersedia mempertimbangkan classifier berdasarkan hyperplane yang tidak memisahkan dua kelas dengan sempurna, untuk kepentingan
  - ketahanan (robustness) yang lebih besar untuk pengamatan individu,
  - klasifikasi yang lebih baik dari sebagian besar pengamatan data training

- Artinya, mungkin bermanfaat untuk salah mengklasifikasikan beberapa pengamatan data training untuk melakukan pekerjaan yang lebih baik dalam mengklasifikasikan pengamatan yang tersisa (data testing).

- ***Support Vector Classifier***, terkadang disebut *soft margin classifier*, melakukan hal ini dengan tepat.
  - Alih-alih mencari margin sebesar mungkin sehingga setiap pengamatan tidak hanya berada di sisi yang benar dari hyperplane tetapi juga di sisi yang benar dari margin, metode ini membiarkan beberapa pengamatan berada di sisi yang salah dari margin, atau bahkan di sisi yang salah dari hyperplane. (Marginnya lunak (*soft*) karena bisa dilanggar oleh beberapa pengamatan data training.)

- Pengamatan tidak hanya berada di sisi yang salah dari margin, tetapi juga di sisi yang salah dari hyperplane.
- Padahal, ketika tidak ada hyperplane yang memisahkan, situasi seperti itu tidak bisa dihindari.
- Pengamatan di sisi yang salah dari hyperplane sesuai dengan pengamatan data training yang salah diklasifikasikan oleh *support vector classifier*.



Tidak ada observasi yang berada di sisi yang salah dari hyperplane.

- *Support vector classifier* mengklasifikasikan pengamatan data testing tergantung pada sisi mana dari hyperplane itu terletak.

- Hyperplane dipilih untuk memisahkan dengan benar sebagian besar observasi data training ke dalam dua kelas, tetapi mungkin salah mengklasifikasikan beberapa observasi.

- Secara singkat, *support vector classifier* adalah solusi untuk masalah optimisasi

$$\underset{\beta_0,\beta_1,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n}{\text{maximize}} \quad M$$

$$\text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 = 1,$$

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) \geq M(1 - \epsilon_i),$$

$$\epsilon_i \geq 0, \quad \sum_{i=1}^{n} \epsilon_i \leq C,$$

lebar margin; diusahakan membuat kuantitas ini sebesar mungkin

nonnegative tuning parameter

Slack variabel (variabel "kendur") yang memungkinkan pengamatan individu berada di sisi yang salah dari margin atau hyperplane

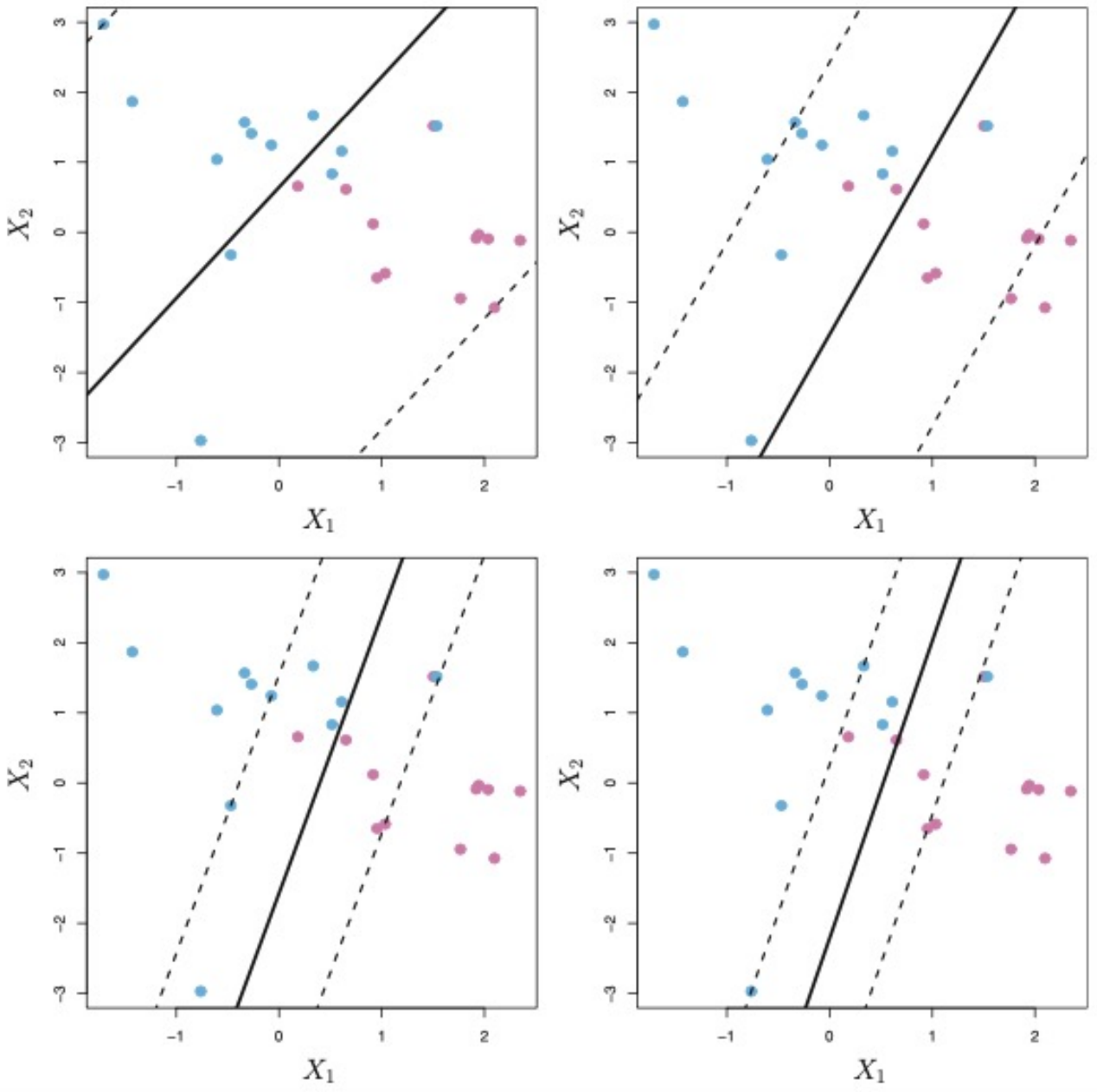
$\epsilon_i = 0$ → pengamatan ke-i terletak pada sisi margin yang benar

$\epsilon_i > 0$ → pengamatan ke-i terletak pada sisi margin yang salah

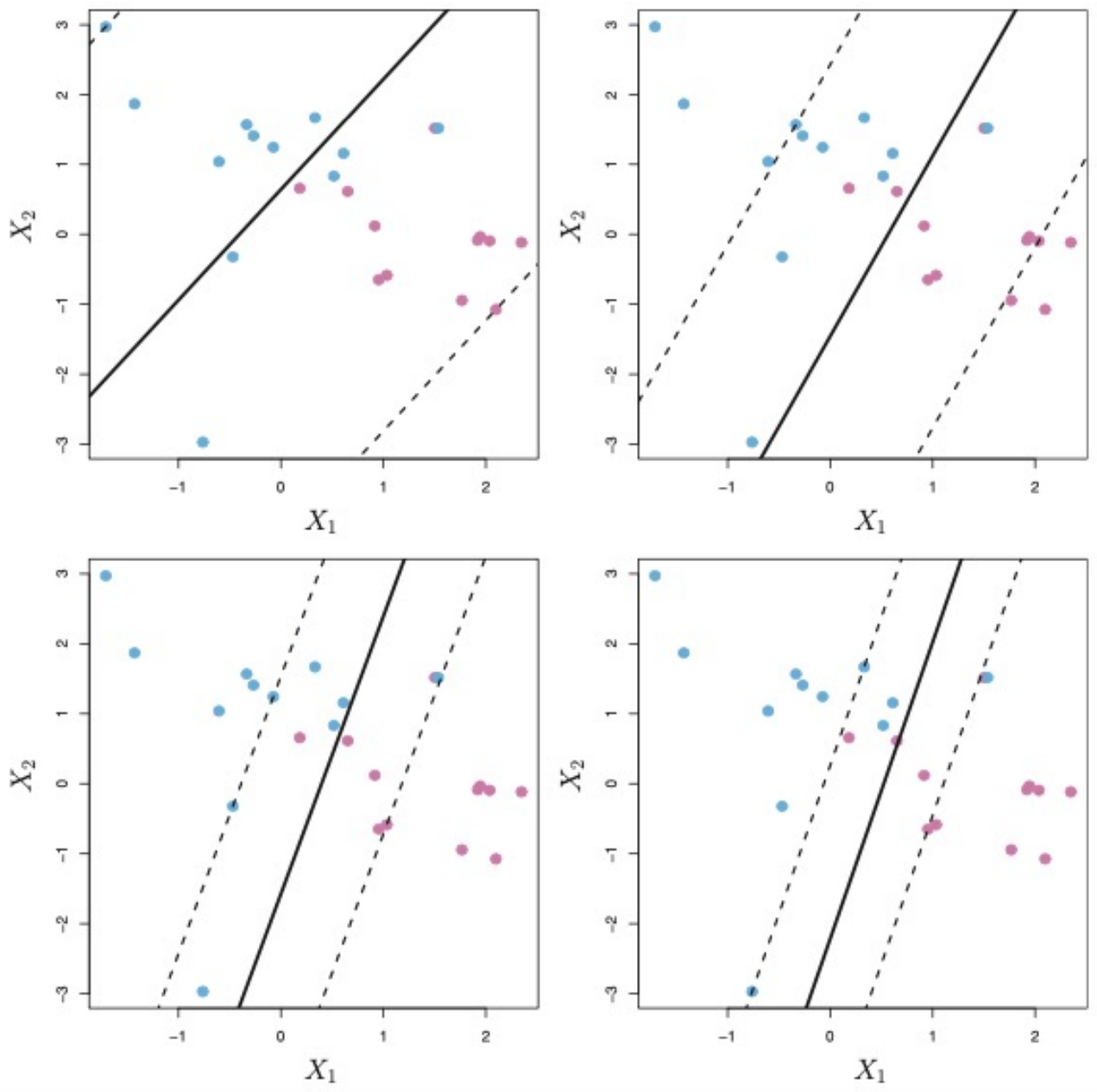
$\epsilon_i > 1$ → pengamatan ke-i terletak pada sisi hyperplane yg salah

Ketika C kecil, dicari margin sempit yang jarang dilanggar; ini sama dengan pengklasifikasi yang sangat sesuai dengan data, yang mungkin memiliki bias rendah tetapi ragam tinggi.

Di sisi lain, ketika C lebih besar, marginnya lebih lebar dan diizinkan lebih banyak pelanggaran; ini sama dengan menyesuaikan data dengan lebih mudah dan mendapatkan pengklasifikasi yang berpotensi lebih bias tetapi mungkin memiliki ragam yang lebih rendah.



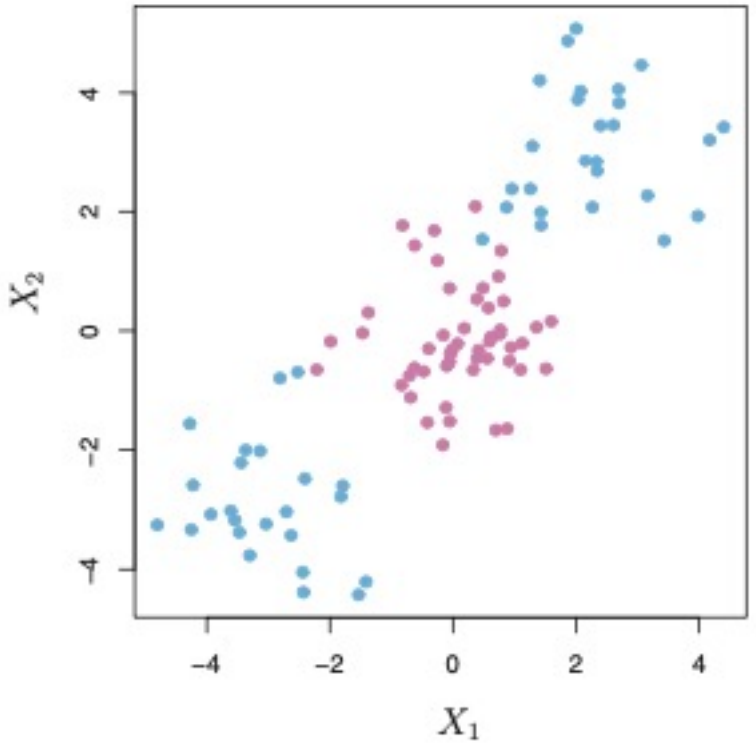
- Pengelompokan *support vector classifier* menggunakan empat nilai berbeda dari tuning parameter C.
- Nilai C terbesar digunakan di panel kiri atas, dan nilai yang lebih kecil digunakan di panel kanan atas, kiri bawah, dan kanan bawah.
- Ketika C besar, maka ada toleransi yang tinggi untuk pengamatan berada di sisi margin yang salah, sehingga marginnya akan besar.
- Saat C berkurang, toleransi untuk pengamatan berada di sisi yang salah dari margin berkurang, dan margin menyempit.



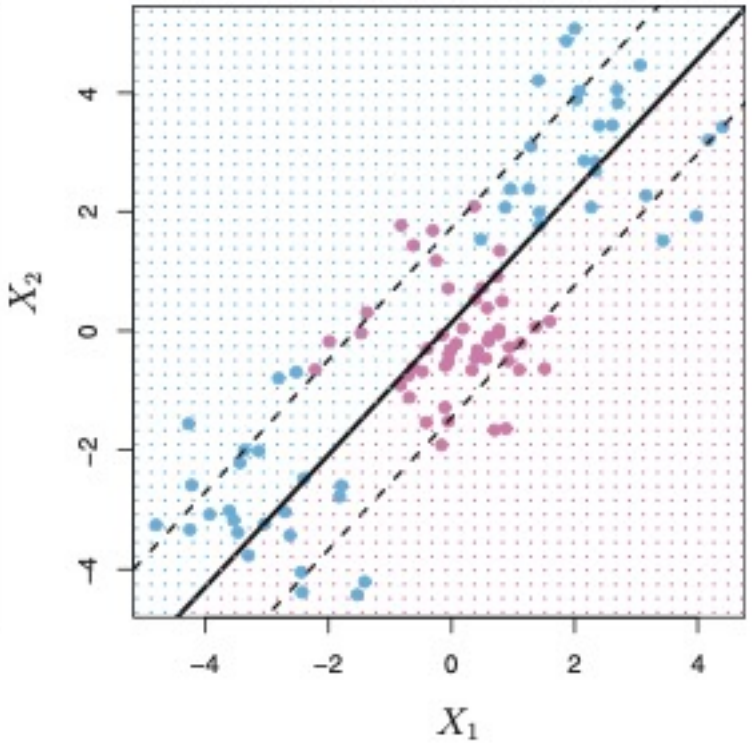
- Pengamatan yang terletak langsung di margin, atau di sisi yang salah dari margin kelasnya, dikenal sebagai vektor pendukung (*support vector*).

- Panel kiri atas: pengklasifikasi ini memiliki ragam rendah (karena banyak pengamatan adalah vektor pendukung (*support vector*)) tetapi berpotensi memiliki bias yang tinggi.

- Sebaliknya, jika C kecil, maka akan ada lebih sedikit vektor pendukung (*support vector*) dan karenanya pengklasifikasi yang dihasilkan akan memiliki bias yang rendah tetapi ragam yang tinggi. Panel kanan bawah mengilustrasikan pengaturan ini, dengan hanya delapan vektor pendukung (*support vector*).



Pengamatan dibagi menjadi dua kelas, dengan batas non-linier di antara keduanya.



Support vector classifier mencari batas linier, dan akibatnya kinerjanya sangat buruk.

Support Vector Machines

- *Support vector classifier* adalah pendekatan alami untuk klasifikasi dalam pengaturan dua kelas, jika batas antara kedua kelas adalah linier. Namun, dalam praktiknya kita terkadang dihadapkan pada batasan kelas yang tidak linier.

- Dalam kasus *support vector classifier*, kita dapat mengatasi masalah kemungkinan batas non-linear antar kelas dengan cara memperbesar ruang fitur menggunakan fungsi polinomial kuadrat, kubik, dan bahkan orde lebih tinggi dari prediktor (fungsi polynomial dari variabel prediktor).

- Tidak sulit untuk melihat bahwa ada banyak cara yang memungkinkan untuk memperbesar ruang fitur, dan jika kita tidak hati-hati, kita bisa mendapatkan banyak fitur. Kemudian perhitungan akan menjadi tidak terkendali. ***Support vector machines***, memungkinkan untuk memperbesar ruang fitur yang digunakan oleh *support vector classifier* dengan cara yang mengarah pada komputasi yang efisien.

Support Vector Machines

- ***Support vector machine*** (SVM) adalah pengembangan dari *support vector classifier* yang dihasilkan dari memperbesar ruang fitur dengan cara tertentu, yakni menggunakan kernel.

- Kita mungkin ingin memperbesar ruang fitur kita untuk mengakomodasi batas non-linear antar kelas. Pendekatan kernel yang diuraikan di sini hanyalah sebuah pendekatan komputasi yang efisien untuk menjalankan ide ini.

- Kernel: fungsi yang mengukur kesamaan dua pengamatan

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j} \longrightarrow \text{Kernel linier untuk } \textit{support vector classifier}$$

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^{d} \longrightarrow \text{Kernel polinomial dengan derajat } d$$

- Pada dasarnya sama dengan fitting *support vector classifier* dalam ruang berdimensi lebih tinggi yang melibatkan polinomial derajat d, bukan di ruang fitur aslinya. Ketika *support vector classifier* digabungkan dengan kernel non-linear seperti kernel polinomial, classifier yang dihasilkan dikenal sebagai ***support vector machine***.

Support Vector Machines

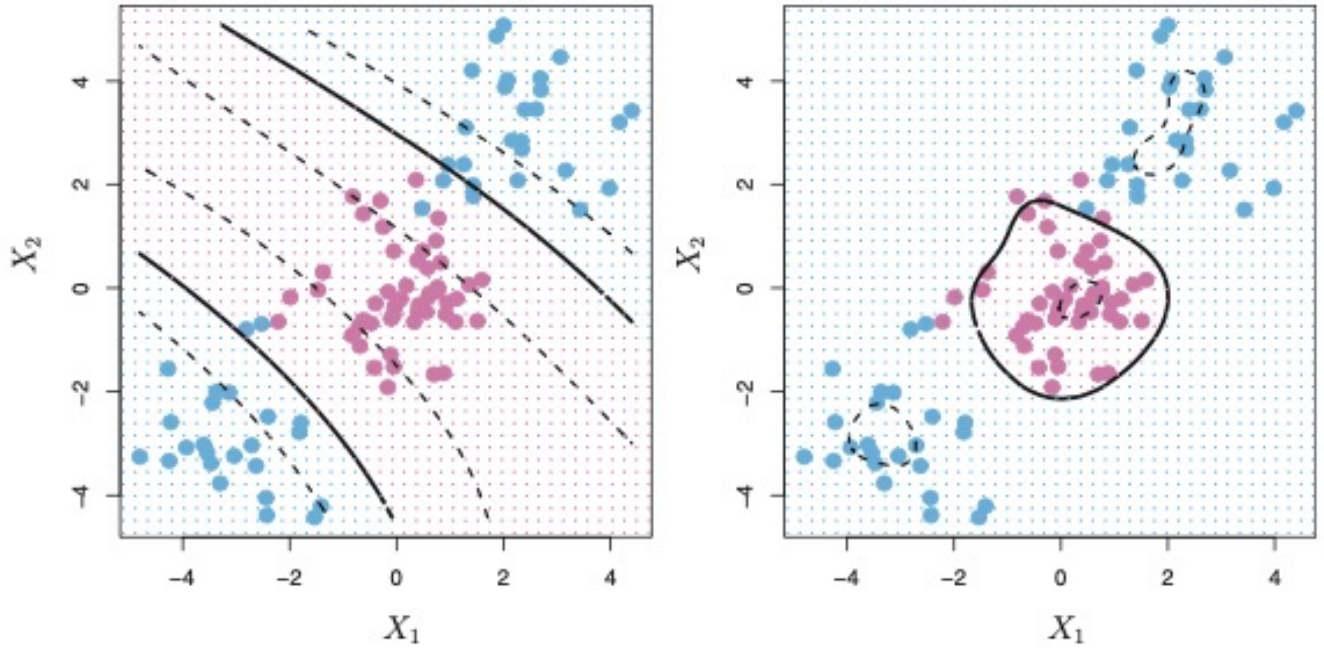
- Sehingga, pada kasus non-linier, fungsi SVM-nya menjadi:

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i)$$

- Pilihan kernel lainnya yang popular adalah *radial kernel*

$$K(x_i, x_{i'}) = exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right)$$

SVM dengan kernel polinomial derajat 3 diterapkan pada data non-linear

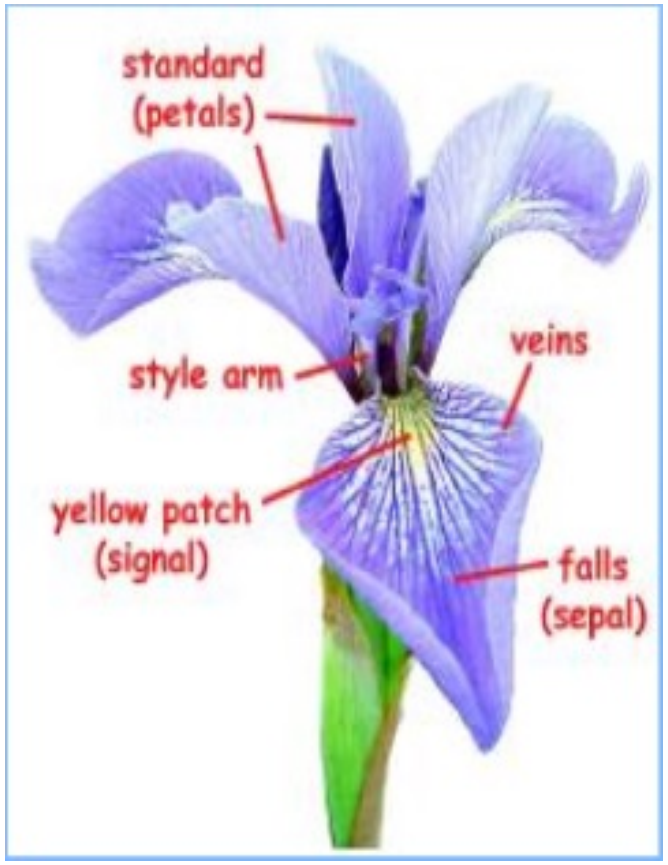


SVM dengan kernel radial diterapkan. Dalam contoh ini, salah satu kernel mampu menangkap batas keputusan yang lebih tepat

Aplikasi di R

- Using IRIS data set.
- We have features of three kinds of flowers: Setosa, Versicolor, and Virginica.
- We need to discriminate between these three kinds of flower throufh their features so we will take some of three features as training of SVM and some of these features as a testing of this classifier to know the ability of this classifier to discriminate between these three kinds of flowers.

- The features, which are collected of these flowers, are: Sepal length, Sepal width, Petal length, and Petal width.



Load the data set

```
data(iris)
head(iris)

# to see the description
help(iris)

summary(iris)
```

Create training and testing data set

- Make filtration of the iris Data Set by selecting some rows for testing .

  testidx<-which(1:length(iris[,1])%%5==0)

  testidx

```
> testidx<-which(1:length(iris[,1])%%5==0)
> testidx
 [1]   5  10  15  20  25  30  35  40  45  50  55  60  65  70  75
[16]  80  85  90  95 100 105 110 115 120 125 130 135 140 145 150
```

Create training and testing data set

- Make selection of the rows which will be used to make training of The SVM. If you note that row 10 and 20 is not in the list of training matrix because these lines are selected as part of other rows which will be used for testing .

```
iristrain<-iris[-testidx,]

iristrain

head(iristrain)
```

```
> head(iristrain)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
7          4.6         3.4          1.4         0.3  setosa
```

Create training and testing data set

- Make selection of the rows which will be used for testing . If you note that rows 10 and 20 in the list of these rows because these rows are used for testing.

```
iristest<-iris[testidx,]
head(iristest)
```

```
> head(iristest)
   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
5           5.0         3.6          1.4         0.2  setosa
10          4.9         3.1          1.5         0.1  setosa
15          5.8         4.0          1.2         0.2  setosa
20          5.1         3.8          1.5         0.3  setosa
25          4.8         3.4          1.9         0.2  setosa
30          4.7         3.2          1.6         0.2  setosa
```

- Load the package.

```
library(e1071)
```

- Make training of SVM classifier as follows:

```
model <-svm(Species~., data=iristrain)
print(model)
summary(model)
```

Make Prediction

- Using the SVM in prediction by testing the model with a new data. In this step, we will see the ability of SVM in discriminating between different types of Data of three kinds of flowers : Setosa , Versicolor and Virginica.

```
prediction<-predict(model,iristest[,-5])

prediction
```

```
> prediction
         5         10         15         20         25
    setosa     setosa     setosa     setosa     setosa
        30         35         40         45         50
    setosa     setosa     setosa     setosa     setosa
        55         60         65         70         75
versicolor versicolor versicolor versicolor versicolor
        80         85         90         95        100
versicolor versicolor versicolor versicolor versicolor
       105        110        115        120        125
 virginica  virginica  virginica versicolor  virginica
       130        135        140        145        150
 virginica  virginica  virginica  virginica  virginica
Levels: setosa versicolor virginica
```

Confusion Matrix

- The last Step, we will see whether the SVM classifier can predict with all types of flowers based on the testing data which we gave to it or not.
- As we can see, the SVM classifier can predict with the Setosa and Versicolor flowers perfectly where the classification accuracy is 100% , but it can't predict the Virginica flower perfectly where the classification accuracy is 90%.

table(iristest$Species,prediction)

```
> table(iristest$Species,prediction)
            prediction
             setosa versicolor virginica
  setosa         10          0         0
  versicolor      0         10         0
  virginica       0          1         9
```

Latihan

1. Hyperplane pada ruang 3-dimensi adalah ...
2. Pengembangan lebih lanjut dari support vector classifier untuk mengakomodasi batas kelas non-linear disebut dengan ...
3. Pengamatan yang terletak pada wilayah margin pada SVM, disebut dengan ...
4. $y_i\left(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}\right) > 0$ merupakan definisi dari …
5. Hyperplane pemisah yang memiliki memiliki jarak minimum margin terjauh ke pengamatan data training, disebut dengan ...

REMINDER Tugas Kelompok – Sesi UTS

- Buatlah proposal penelitian mengenai Projek Kelompok-nya, yang di dalamnya berisi
  1. Judul berdasarkan topik projek yang ditentukan
  2. Latar belakang dan tujuan
  3. Data dan peubah-peubah yang digunakan
  4. Metodologi (rencana tahapan analisis data yang dilakukan)

- Selain proposal penelitian dalam format makalah, dikumpulkan juga file presentasi dalam powerpoint.

- File proposal penelitian dan file presentasi diupload pada class.ipb.ac.id

- Batas waktu pengiriman adalah **Hari Minggu, tanggal 3 Maret 2024 jam 23:59 WIB**

Terima kasih ☺