# Pertemuan 6 — Naive Bayes dan Support Vector Machine

Menyederhanakan peluang, lalu memperlebar margin

Hendri Karisma

Semester Ganjil 2026/2027

Program Studi Teknik Informatika, STMIK Tazkia

# Alur pertemuan hari ini

**Bagian A — Naive Bayes**

1. Dari Bayes optimal ke naive Bayes
2. Asumsi independensi bersyarat
3. Contoh terhitung pada PlayTennis
4. Klasifikasi teks dan penyaring *spam*
5. Peluang nol dan *smoothing*
6. Mengapa asumsi yang salah tetap bekerja

**Bagian B — Support Vector Machine**

7. Hyperplane mana yang paling baik
8. Margin maksimum dan *support vector*
9. *Soft margin* dan parameter $C$
10. *Kernel trick* dan RBF
11. Menyetel $C$ dan $\gamma$
12. Perbandingan keduanya

**Sasaran pertemuan ini**

Pertemuan 5 berakhir dengan kesimpulan yang pahit: pengklasifikasi Bayes optimal adalah yang terbaik yang mungkin ada, tetapi ongkosnya membuatnya tak terpakai. Hari ini dua jalan keluar yang sangat berbeda: naive Bayes tetap di dalam kerangka peluang dan menebus ongkos itu dengan satu asumsi yang berani; SVM meninggalkan peluang dan bertanya soal geometri.

# Dari Bayes optimal ke naive Bayes

# Pengingat: mengapa Bayes optimal terlalu mahal

Pertemuan 5 memberi kita aturan keputusan terbaik secara teori, yaitu memilih kelas dengan peluang posterior terbesar:

$$v_{MAP} = \arg\max_{v \in V} P(v \mid a_1, \ldots, a_d) = \arg\max_{v \in V} P(a_1, \ldots, a_d \mid v)\, P(v)$$

Persoalannya ada pada suku $P(a_1, \ldots, a_d \mid v)$: menaksirnya berarti menghitung frekuensi **setiap kombinasi nilai atribut** di dalam setiap kelas.

- Pada PlayTennis — Outlook (3 nilai), Temperature (3), Humidity (2), Wind (2) — ada $3 \times 3 \times 2 \times 2 = 36$ kombinasi, sekitar 70 bilangan untuk dua kelas, dari 14 baris data.
- Pada teks dengan kosakata 10.000 kata biner, kombinasinya $2^{10000}$.

**Intinya**

Masalahnya bukan rumusnya, melainkan **jumlah parameter** yang harus dipelajari. Kebanyakan kombinasi tidak pernah muncul di data latih, sehingga taksirannya nol atau sangat berderau.

# Satu asumsi yang mengubah segalanya

**Independensi bersyarat**

Diberikan kelasnya, nilai satu atribut dianggap tidak memberi informasi apa pun tentang nilai atribut lain:

$$P(a_1, a_2, \ldots, a_d \mid v) = \prod_{i=1}^{d} P(a_i \mid v)$$

Perhatikan kata **bersyarat**: kita tidak mengatakan atribut-atribut itu independen secara umum, hanya bahwa setelah kelasnya diketahui kaitan di antara mereka diabaikan. Pada PlayTennis ini jelas keliru — Outlook dan Humidity tetap berkaitan walaupun kelasnya sudah diketahui. Imbalannya besar: jumlah parameter turun dari perkalian menjadi **penjumlahan**, dari $3 \times 3 \times 2 \times 2$ menjadi $3 + 3 + 2 + 2 = 10$ taksiran per kelas, dan untuk teks dari $2^{10000}$ menjadi $2 \times 10.000$. Itulah tukar tambahnya — sedikit kebenaran model diserahkan demi sesuatu yang benar-benar dapat dipelajari dari data yang ada.

# Aturan naive Bayes dan cara melatihnya

## Pengklasifikasi naive Bayes

$$v_{NB} = \arg\max_{v \in V} P(v) \prod_{i=1}^{d} P(a_i \mid v)$$

**Melatih** berarti mengisi dua jenis tabel frekuensi, sekali saja, dengan satu lintasan atas data:

- **Peluang awal** $\hat{P}(v) = n_v/n$, yaitu porsi data yang berkelas $v$.
- **Peluang bersyarat** $\hat{P}(a_i \mid v) = n_{v,a_i}/n_v$, yaitu di antara data berkelas $v$, berapa bagian yang bernilai $a_i$ pada atribut ke-$i$.

Tidak ada pencarian, tidak ada iterasi, tidak ada *learning rate*. Yang dilakukan hanyalah **menghitung** — salah satu dari sedikit model yang tidak perlu dioptimasi secara numerik.

Sumber: Mitchell, *Machine Learning*, Bab 6.9.

# Contoh terhitung: PlayTennis

# Data latih yang sudah kita kenal dari Pertemuan 3

| Outlook | Temp | Humidity | Wind | Play | Outlook | Temp | Humidity | Wind | Play |
|---|---|---|---|---|---|---|---|---|---|
| Sunny | Hot | High | Weak | No | Sunny | Mild | High | Weak | No |
| Sunny | Hot | High | Strong | No | Sunny | Cool | Normal | Weak | Yes |
| Overcast | Hot | High | Weak | Yes | Rain | Mild | Normal | Weak | Yes |
| Rain | Mild | High | Weak | Yes | Sunny | Mild | Normal | Strong | Yes |
| Rain | Cool | Normal | Weak | Yes | Overcast | Mild | High | Strong | Yes |
| Rain | Cool | Normal | Strong | No | Overcast | Hot | Normal | Weak | Yes |
| Overcast | Cool | Normal | Strong | Yes | Rain | Mild | High | Strong | No |

Empat belas baris, dua kelas: **Yes** sebanyak 9 baris dan **No** sebanyak 5 baris. Pertanyaan hari ini: apa keputusan naive Bayes untuk hari baru

$$\langle \text{Outlook} = \text{Sunny},\ \text{Temp} = \text{Cool},\ \text{Humidity} = \text{High},\ \text{Wind} = \text{Strong} \rangle\ ?$$

Kombinasi persis ini **tidak ada** di data latih. Pendekatan tabel penuh akan berhenti di situ; naive Bayes tetap bisa menjawab karena ia hanya perlu potongan-potongannya.

# Langkah 1: kumpulkan tabel frekuensinya

| Taksiran yang diperlukan | Kelas Yes | Kelas No |
|---|---|---|
| $P(v)$ | $9/14 = 0{,}643$ | $5/14 = 0{,}357$ |
| $P(\text{Outlook} = \text{Sunny} \mid v)$ | $2/9 = 0{,}222$ | $3/5 = 0{,}600$ |
| $P(\text{Temp} = \text{Cool} \mid v)$ | $3/9 = 0{,}333$ | $1/5 = 0{,}200$ |
| $P(\text{Humidity} = \text{High} \mid v)$ | $3/9 = 0{,}333$ | $4/5 = 0{,}800$ |
| $P(\text{Wind} = \text{Strong} \mid v)$ | $3/9 = 0{,}333$ | $3/5 = 0{,}600$ |

Cara membaca satu barisnya: di antara 9 hari berkelas Yes ada 2 hari yang Sunny, sehingga $P(\text{Sunny} \mid \text{Yes}) = 2/9$; di antara 5 hari berkelas No ada 3 hari yang Sunny, sehingga $P(\text{Sunny} \mid \text{No}) = 3/5$. Penyebutnya berbeda karena keduanya dihitung **di dalam kelasnya masing-masing**, bukan atas seluruh 14 baris.

**Intinya**

Semua yang dibutuhkan model ada di tabel ini: sembilan bilangan, bukan tujuh puluh.

# Langkah 2: kalikan, lalu bandingkan

Untuk kelas **Yes**:

$$P(\text{Yes}) \prod_i P(a_i \mid \text{Yes}) = 0{,}643 \times 0{,}222 \times 0{,}333 \times 0{,}333 \times 0{,}333 = \mathbf{0{,}0053}$$

Untuk kelas **No**:

$$P(\text{No}) \prod_i P(a_i \mid \text{No}) = 0{,}357 \times 0{,}600 \times 0{,}200 \times 0{,}800 \times 0{,}600 = \mathbf{0{,}0206}$$

Karena $0{,}0206 > 0{,}0053$, maka $v_{NB} = \text{No}$: hari itu sebaiknya tidak bermain tenis. Kedua bilangan itu belum berupa peluang, tetapi penyebutnya sama untuk kedua kelas, sehingga cukup dinormalkan bila kita memerlukan angka peluang: $P(\text{No} \mid \text{hari itu}) \approx 0{,}0206/(0{,}0206 + 0{,}0053) = 0{,}795$.

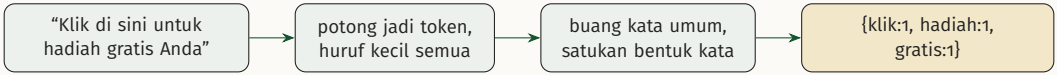
**Intinya**

Perhatikan siapa yang menentukan: Humidity berbanding 0,800 lawan 0,333 dan Outlook 0,600 lawan 0,222. Dua atribut itulah yang mendorong keputusan ke arah No.

# Naive Bayes untuk klasifikasi teks

# Bagaimana sebuah surel menjadi vektor

Teks tidak punya kolom; kita harus membuatnya. Model yang paling lazim dipakai bersama naive Bayes adalah **bag-of-words**: dokumen diwakili kata-kata yang muncul di dalamnya, sedangkan urutannya dibuang.




Setiap kata menjadi satu atribut. Dengan kosakata 20.000 kata, setiap surel adalah titik di ruang berdimensi 20.000 — dan hampir semua koordinatnya nol.

**Mengapa naive Bayes cocok di sini**

Dimensinya sangat tinggi sehingga model lain mudah *overfitting*, sedangkan naive Bayes hanya punya sedikit parameter per dimensi dan cukup menyentuh kata yang benar-benar muncul. Melatihnya pun sekadar menghitung, sehingga penyaring dapat dilatih ulang tiap malam.

# Menaksir $P(\text{kata} \mid \text{kelas})$ pada korpus mini

Pada varian **multinomial**, satu posisi kata dianggap satu penarikan dari kantong kata milik kelas itu, sehingga taksirannya memakai jumlah kemunculan dibagi total token kelas:

$$\hat{P}(w \mid c) = \frac{\text{jumlah kemunculan } w \text{ di seluruh dokumen kelas } c}{\text{total token pada kelas } c}$$

**Kelas SPAM — 3 surel, 10 token**

- “menang hadiah gratis”
- “gratis hadiah klik sekarang”
- “klik gratis menang”

Hitungan: gratis 3, hadiah 2, menang 2, klik 2, sekarang 1.

**Kelas BUKAN SPAM — 3 surel, 10 token**

- “rapat besok pagi”
- “jadwal rapat proyek”
- “kirim laporan proyek besok”

Hitungan: rapat 2, besok 2, proyek 2, pagi 1, jadwal 1, kirim 1, laporan 1.

Kosakata gabungannya berisi $|V| = 12$ kata, dan $P(\text{spam}) = P(\text{bukan}) = 3/6 = 0{,}5$. Contoh taksiran mentahnya: $\hat{P}(\text{gratis} \mid \text{spam}) = 3/10 = 0{,}3$, sedangkan $\hat{P}(\text{rapat} \mid \text{spam}) = 0/10 = 0$.

# Masalah peluang nol

Kita ingin menilai surel baru: **“gratis rapat besok”**. Dengan taksiran mentah tadi:

$$\text{SPAM:}\quad 0{,}5 \times \underbrace{0{,}3}_{\text{gratis}} \times \underbrace{0{,}0}_{\text{rapat}} \times \underbrace{0{,}0}_{\text{besok}} = 0$$

$$\text{BUKAN:}\quad 0{,}5 \times \underbrace{0{,}0}_{\text{gratis}} \times \underbrace{0{,}2}_{\text{rapat}} \times \underbrace{0{,}2}_{\text{besok}} = 0$$

Kedua kelas memberi nol, dan model tidak dapat memutuskan apa pun.

- Penyebabnya perkalian: **satu** faktor nol membatalkan seluruh bukti dari faktor lain.
- Kata yang belum pernah muncul di suatu kelas itu hal **biasa**: dengan kosakata 20.000 kata, sebagian besar sel di tabel itu memang kosong.
- Taksiran 0/10 mengatakan “mustahil”, padahal yang benar hanya “belum pernah terlihat”.

**Intinya**

Frekuensi mentah rapuh justru pada data yang sedikit. Perbaikannya disebut *smoothing*: geser sedikit setiap taksiran menjauh dari nol.

# Dua bentuk smoothing yang perlu Anda kuasai

### Laplace, atau *add-one*

$$\hat{P}(a_i \mid v) = \frac{n_{v,a_i} + 1}{n_v + k}$$

dengan $k$ = banyaknya nilai yang mungkin pada atribut itu (teks: $k = |V|$).

Tambahkan satu pengamatan semu pada setiap kemungkinan nilai.

### m-estimate

$$\hat{P}(a_i \mid v) = \frac{n_{v,a_i} + m\,p}{n_v + m}$$

dengan $p$ = peluang awal yang kita yakini, dan $m$ = besar keyakinan itu diukur dalam "jumlah data semu".

Laplace adalah kasus khusus dengan $p = 1/k$ dan $m = k$.

Tafsirannya mudah diingat: kita berpura-pura sudah pernah melihat $m$ contoh tambahan yang tersebar menurut keyakinan awal $p$. Bila $n_v$ besar tambahan itu tidak terasa; bila $n_v$ kecil, tambahan itulah yang menyelamatkan kita dari nol.

### Intinya

Nilai $m$ mengatur seberapa berani kita mempercayai data yang sedikit — semakin kecil datanya, semakin pantas $m$ diperbesar.

# Smoothing dengan angka nyata

**Pada PlayTennis.** Tidak ada hari Overcast yang berkelas No, sehingga $\hat{P}(\text{Overcast} \mid \text{No}) = 0/5 = 0$ dan hari Overcast otomatis dinilai Yes. Atribut Outlook punya $k = 3$ nilai, sehingga

$$\text{Laplace: } \frac{0+1}{5+3} = 0{,}125 \qquad \text{m-estimate } (p = \tfrac{1}{3},\, m = 3): \ \frac{0 + 3 \cdot \frac{1}{3}}{5+3} = 0{,}125$$

Keduanya sama, karena di sini Laplace memang m-estimate dengan $p = 1/3$ dan $m = 3$. Taksiran yang tidak nol ikut bergeser: $P(\text{Sunny} \mid \text{No})$ dari 0,600 menjadi 0,500, ditarik ke arah seragam tetapi tidak dirusak.

**Pada korpus surel** ($n_c = 10$ token, $|V| = 12$, penyebut menjadi $10 + 12 = 22$):

| | SPAM | | BUKAN SPAM | |
|---|---|---|---|---|
| **Kata** | mentah | Laplace | mentah | Laplace |
| gratis | $3/10 = 0{,}300$ | $4/22 = 0{,}182$ | $0/10 = 0$ | $1/22 = 0{,}045$ |
| klik, hadiah | $2/10 = 0{,}200$ | $3/22 = 0{,}136$ | $0/10 = 0$ | $1/22 = 0{,}045$ |
| rapat | $0/10 = 0$ | $1/22 = 0{,}045$ | $2/10 = 0{,}200$ | $3/22 = 0{,}136$ |

# Keputusan penyaring spam, dan mengapa dihitung dalam log

Surel baru: **“gratis klik hadiah”**. Dengan taksiran Laplace di slide sebelumnya:

$$\text{SPAM:}\quad 0{,}5 \times \tfrac{4}{22} \times \tfrac{3}{22} \times \tfrac{3}{22} = 1{,}69 \times 10^{-3} \qquad \text{BUKAN:}\quad 0{,}5 \times \tfrac{1}{22} \times \tfrac{1}{22} \times \tfrac{1}{22} = 4{,}70 \times 10^{-5}$$

Perbandingannya $36 : 1$, sehingga $P(\text{SPAM} \mid \text{surel}) = 36/37 = 0{,}973$ dan surel itu ditandai spam. Bandingkan dengan “gratis rapat besok” yang tadi macet di nol: kini ia memberi $1{,}88 \times 10^{-4}$ untuk SPAM lawan $4{,}23 \times 10^{-4}$ untuk BUKAN, jadi diputuskan bukan spam — masuk akal, karena dua dari tiga katanya kata kerja kantor.

**Satu keharusan rekayasa**

Surel sepanjang 200 kata berarti mengalikan 200 bilangan yang masing-masing sekitar $10^{-4}$. Hasilnya di bawah batas *floating point* dan menjadi nol di komputer. Karena itu implementasi nyata selalu bekerja dalam logaritma, $\log P(v) + \sum_i \log P(a_i \mid v)$: urutan kelasnya tidak berubah karena logaritma naik monoton, tetapi penjumlahan jauh lebih aman daripada perkalian.

# Mengapa asumsi yang salah tetap bekerja

# Peringkat sudah cukup; kalibrasinya yang rusak

Asumsi independensi bersyarat hampir selalu salah — pada teks, “gratis” dan “hadiah” jelas berkorelasi di dalam kelas spam. Namun naive Bayes tetap salah satu penyaring spam paling tangguh.

- **Yang dibutuhkan hanya urutan.** Keputusan diambil dengan $\arg\max$. Selama kelas yang benar tetap di peringkat teratas, besaran peluangnya boleh melenceng jauh.
- **Menghitung ganda sering berpihak pada kelas yang benar.** Ketika dua atribut berkorelasi, buktinya dihitung dua kali — dan pembesaran itu terjadi pada kelas yang memang didukung bukti tersebut.
- **Sedikit parameter berarti sedikit ragam.** Biasnya besar tetapi variansnya kecil; pada data latih yang sedikit itu pertukaran yang menguntungkan.

**Tetapi hati-hati memakai angkanya**

Keluarannya bukan peluang yang layak dipercaya. Naive Bayes terkenal *overconfident*: ia gemar memberi 0,999 atau 0,001. Bila peluang itu dipakai untuk keputusan berbiaya, kalibrasilah dulu, misalnya dengan *Platt scaling*.

# Tiga varian, dan ongkos yang sangat murah

| Varian | Bentuk fitur | Taksiran $P(a_i \mid v)$ |
|---|---|---|
| **Gaussian** | Numerik kontinu (suhu, nilai transaksi) | Sebaran normal per kelas: cukup taksir rerata $\mu_{i,v}$ dan simpangan $\sigma_{i,v}$ |
| **Multinomial** | Hitungan kemunculan (frekuensi kata, TF-IDF) | Porsi token: $(n_{v,w} + \alpha)/(n_v + \alpha\lvert V\rvert)$ |
| **Bernoulli** | Biner: ada atau tidak ada | Porsi dokumen yang memuat kata itu; ketidakhadiran kata **ikut** dihitung sebagai bukti |

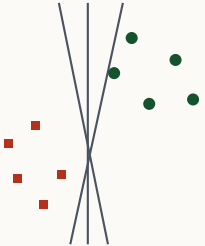
Untuk surel yang panjangnya beragam, multinomial biasanya lebih baik; untuk dokumen pendek seperti judul atau pesan singkat, Bernoulli sering menang karena ketiadaan kata juga informatif.

## Kembali ke bahasan efisiensi Pertemuan 1

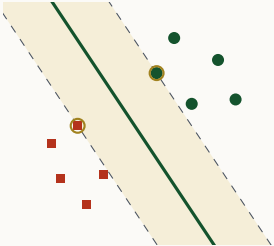
Melatih naive Bayes berbiaya $O(nd)$ — satu lintasan atas $n$ data dengan $d$ fitur, sekadar mengisi tabel hitungan. Menjawab satu pertanyaan berbiaya $O(d)$. Tidak ada model di mata kuliah ini yang lebih murah. Karena itu ia hampir selalu layak dijadikan *baseline*: bila model rumit Anda tidak mampu mengalahkannya, kerumitan itu belum terbayar.

# Support Vector Machine: mencari margin terlebar

# Banyak garis bisa memisahkan; mana yang dipilih?



Semuanya benar di data latih



Margin terlebar; dua titik bertanda

## Intinya

Ketiga garis di kiri punya galat latih nol, jadi ukuran “berapa yang salah” tidak membantu memilih. SVM menambahkan kriteria baru: pilih garis yang **jaraknya ke titik terdekat dari kedua kelas paling besar**. Garis yang menempel pada data akan berpindah sisi begitu ada sedikit pergeseran; garis yang berjarak lega punya ruang untuk salah sedikit tanpa berubah keputusan.

# Menulis hyperplane, jarak, dan margin

Sebuah **hyperplane** dinyatakan oleh vektor bobot $w$ dan bias $b$, yaitu $w^\top x + b = 0$. Vektor $w$ menunjuk arah tegak lurus permukaan itu, $b$ menggesernya menjauh dari titik asal, dan keputusannya diambil dari tandanya: $\hat{y} = \operatorname{sign}(w^\top x + b)$.

- **Jarak satu titik ke hyperplane:** $\operatorname{jarak}(x) = |w^\top x + b| \,/\, \|w\|$. Pembilangnya mengukur seberapa jauh dari nol nilai fungsinya; pembagian oleh $\|w\|$ mengubahnya menjadi jarak sesungguhnya.
- **Penskalaan kanonik.** Pasangan $(w, b)$ dan $(2w, 2b)$ menggambarkan hyperplane yang sama. Kebebasan itu dipakai untuk menetapkan $|w^\top x + b| = 1$ pada titik terdekat, sehingga jarak titik terdekat menjadi $1/\|w\|$.
- **Lebar margin** adalah jarak antara batas $w^\top x + b = +1$ dan $w^\top x + b = -1$, yaitu $2/\|w\|$.

**Intinya**

Memperlebar margin berarti **memperkecil** $\|w\|$. Dari sinilah bentuk optimasinya lahir: tujuan geometris diubah menjadi tujuan aljabar yang dapat diminimalkan.

# Bentuk optimasi, dan mengapa hanya sedikit titik yang berperan

**Hard-margin SVM**

$$\min_{w,\,b} \ \tfrac{1}{2}\|w\|^2 \qquad \text{dengan kendala} \qquad y_i\,(w^\top x_i + b) \geq 1, \quad i = 1, \ldots, n$$

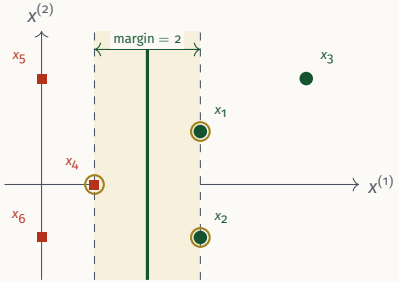
Kendala itu memuat label $y_i \in \{-1, +1\}$ sekaligus, sehingga satu pertidaksamaan berlaku untuk kedua kelas: setiap titik harus berada di sisi yang benar **dan** sejauh setidaknya satu satuan kanonik. Fungsi tujuannya cembung dengan kendala linier, sehingga minimumnya tunggal.

- Titik yang kendalanya **tepat terpenuhi**, $y_i(w^\top x_i + b) = 1$, disebut **support vector**: ia berdiri di batas margin dan menahan hyperplane di tempatnya.
- Titik yang kendalanya berlebih tidak berpengaruh sama sekali. Hapus atau geser sedikit, solusinya tetap.
- Pada bentuk dual hanya support vector yang punya $\alpha_i > 0$, dan $f(x) = \sum_{i \in SV} \alpha_i y_i\, x_i^\top x + b$. Karena itu modelnya ringkas.

# Contoh terhitung: enam titik, dua dimensi

| Titik | $y$ | $y(w^\top x + b)$ | |
|---|---|---|---|
| $x_1 = (3, 1)$ | $+1$ | 1 | **SV** |
| $x_2 = (3, -1)$ | $+1$ | 1 | **SV** |
| $x_3 = (5, 2)$ | $+1$ | 3 | |
| $x_4 = (1, 0)$ | $-1$ | 1 | **SV** |
| $x_5 = (0, 2)$ | $-1$ | 2 | |
| $x_6 = (0, -1)$ | $-1$ | 2 | |

Solusinya $w = (1, 0)$ dan $b = -2$, sehingga $\|w\| = 1$ dan lebar margin $2/\|w\| = 2$. Hyperplane-nya garis $x^{(1)} = 2$.




**Mengapa $w = (1, 0)$ dan bukan yang lain?** Pilihan $w = (2, 0)$, $b = -4$ memberi hyperplane yang sama persis dan semua kendala tetap terpenuhi, tetapi $\frac{1}{2}\|w\|^2 = 2$ bukan 0,5, dan marginnya terbaca $2/2 = 1$. Sebaliknya $w = (0{,}5,\ 0)$, $b = -1$ melanggar kendala karena di $x_1$ nilainya $0{,}5 < 1$. Titik $x_3$, $x_5$, $x_6$ boleh dibuang tanpa mengubah jawaban; membuang $x_4$ akan menggeser hyperplane.

# Soft margin: variabel slack dan parameter $C$

Data nyata jarang terpisah bersih. Satu titik keliru label sudah membuat kendala $y_i(w^\top x_i + b) \geq 1$ mustahil dipenuhi. Jalan keluarnya: izinkan pelanggaran, tetapi kenakan biaya.

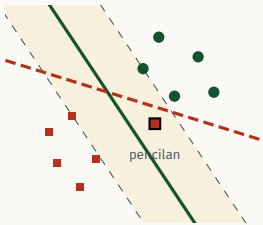
**Soft-margin SVM**

$$\min_{w,\,b,\,\xi} \ \tfrac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \qquad \text{dengan} \qquad y_i(w^\top x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

Setiap $\xi_i$ mengukur seberapa jauh titik ke-$i$ menerobos batas margin, sehingga $\xi_i = \max(0,\, 1 - y_i(w^\top x_i + b))$ — persis *hinge loss*. Bila $\xi_i = 0$ titik itu aman di luar margin; bila $0 < \xi_i \leq 1$ ia masuk margin tetapi masih di sisi yang benar; bila $\xi_i > 1$ ia sudah salah klasifikasi.

Contoh dengan angka slide sebelumnya: bila ada $x_7 = (1{,}2,\ 0)$ berlabel $+1$, maka $y_7(w^\top x_7 + b) = 1{,}2 - 2 = -0{,}8$, sehingga $\xi_7 = 1{,}8$ dan dendanya $C \times 1{,}8$.

# Apa yang terjadi bila $C$ besar atau kecil



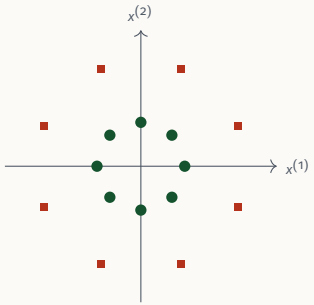

| | $C$ **besar** | $C$ **kecil** |
|---|---|---|
| Prioritas | Jangan ada pelanggaran | Margin harus lebar |
| Gambar | Garis merah menampung pencilan; margin menyempit | Garis hijau membiarkan pencilan salah; margin tetap lega |
| Bias–varians | Varians tinggi, bias rendah | Bias tinggi, varians rendah |
| Risiko khas | *Overfitting* | *Underfitting* |
| Ekstrem | $C \to \infty$: kembali hard margin | $C \to 0$: margin melebar tanpa peduli label |

**Intinya**

$C$ adalah pengendali **regularisasi** pada SVM, dan arah bacanya berlawanan dengan kebiasaan: $C$ besar berarti regularisasi **lemah**. Nilainya tidak dapat ditebak dari teori; ia harus dicari dengan validasi silang pada data Anda sendiri.

# Kernel trick: memisahkan di ruang yang lebih tinggi

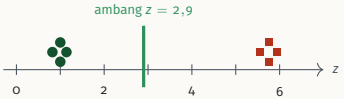
# Ada data yang tidak ada garisnya




Dua kelas dalam dua cincin

Tidak ada satu pun garis lurus yang memisahkan cincin dalam dari cincin luar. Tetapi tambahkan satu fitur baru, yaitu jarak kuadrat titik itu dari pusat:

$$\phi(x) = \big(x^{(1)},\ x^{(2)},\ z\big), \qquad z = (x^{(1)})^2 + (x^{(2)})^2$$



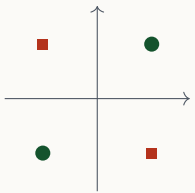

Di ruang baru itu satu ambang sudah memisahkan keduanya: cincin dalam berkumpul di $z \approx 1$, cincin luar di $z \approx 5{,}8$.

Inilah gagasan pokoknya: **ketidakterpisahan bisa merupakan sifat ruangnya, bukan sifat datanya**.

# XOR: pemetaan eksplisit yang bisa Anda periksa sendiri

| $x^{(1)}$ | $x^{(2)}$ | $y$ | $x^{(1)}x^{(2)}$ |
|---|---|---|---|
| $+1$ | $+1$ | $+1$ | $+1$ |
| $-1$ | $-1$ | $+1$ | $+1$ |
| $+1$ | $-1$ | $-1$ | $-1$ |
| $-1$ | $+1$ | $-1$ | $-1$ |

Di dua dimensi aslinya, mustahil. Tetapi dengan $\phi(x) = (x^{(1)},\ x^{(2)},\ x^{(1)}x^{(2)})$ kolom terakhir persis setanda dengan $y$, sehingga $w = (0, 0, 1)$ dan $b = 0$ memisahkannya sempurna.



XOR di ruang asli

## Mengapa disebut *trick*

Menghitung $\phi(x)$ terang-terangan bisa sangat mahal, bahkan tak berhingga dimensinya. Tetapi bentuk dual SVM hanya memerlukan **hasil kali dalam** antar pasangan data, $\phi(x_i)^\top \phi(x_j)$, bukan $\phi(x)$ itu sendiri. Bila ada fungsi $K(x_i, x_j)$ yang memberi nilai itu langsung dari $x$ di ruang asli, kita memperoleh semua keuntungan dimensi tinggi tanpa sekali pun pergi ke sana. Fungsi itulah **kernel**.

# Tiga kernel yang dipakai sehari-hari

| Kernel | Rumus | Kapan dipakai |
|---|---|---|
| **Linear** | $K(x, z) = x^\top z$ | Fitur sudah banyak dan informatif, misalnya teks TF-IDF. Paling cepat dan paling mudah dibaca. |
| **Polinomial** | $K(x, z) = (\gamma\, x^\top z + r)^d$ | Ketika interaksi antar fitur penting sampai derajat tertentu. Derajat di atas 3 mudah tidak stabil secara numerik. |
| **RBF** (Gaussian) | $K(x, z) = \exp\big(-\gamma \lVert x - z \rVert^2\big)$, $\gamma > 0$ | Pilihan bawaan bila bentuk batasnya tidak diketahui; padanan ruang fiturnya berdimensi tak berhingga. |

**Arti $\gamma$ pada RBF.** Nilai kernel adalah kemiripan yang meluruh terhadap jarak, dan $\gamma$ menentukan seberapa cepat peluruhan itu. Untuk dua titik berjarak $\lVert x - z \rVert^2 = 4$:

| $\gamma$ | 0,01 | 0,1 | 1 | 10 |
|---|---|---|---|---|
| $K(x, z)$ | 0,961 | 0,670 | 0,018 | $4 \times 10^{-18}$ |

Jadi $\gamma$ kecil berarti setiap data latih berpengaruh luas dan batasnya mulus; $\gamma$ besar berarti pengaruhnya hanya di sekitar dirinya sendiri, sehingga batasnya menjadi pulau-pulau kecil.

# Menyetel, memperluas, dan memilih

# Menyetel $C$ dan $\gamma$

| Parameter | Terlalu kecil — gejala underfit | Terlalu besar — gejala overfit |
|---|---|---|
| $C$ | Galat latih dan validasi sama-sama tinggi; batasnya terlalu lurus. | Galat latih mendekati nol tetapi galat validasi naik; batasnya berliku mengikuti titik tunggal. |
| $\gamma$ | Model berperilaku seperti kernel linear; kantong kecil tidak tertangkap. | Hampir semua data latih menjadi support vector; akurasi latih 100% sedangkan validasi runtuh. |

## Cara mencarinya

- Cari pada **skala logaritmik**: rentang yang sudah terbukti adalah $C \in \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$ dan $\gamma \in \{2^{-15}, 2^{-13}, \ldots, 2^{3}\}$.
- *Grid search* dengan validasi silang 5 lipatan: kasar dulu, lalu perhalus.
- $C$ dan $\gamma$ berinteraksi, jadi harus dicari sebagai **pasangan**.

## Tidak boleh dilewatkan

SVM **wajib** memakai penskalaan fitur, karena $\|x - z\|^2$ akan dikuasai fitur bersatuan besar. Bila penghasilan (puluhan juta) dan usia (puluhan) dicampur, suku usia praktis hilang. Parameter penskalaannya dihitung **hanya dari data latih**.

# Lebih dari dua kelas, dan ketika targetnya bilangan

## Multikelas

SVM pada dasarnya biner. Untuk $K$ kelas dipakai gabungan beberapa SVM biner.

- **One-vs-Rest** — latih $K$ model, masing-masing "kelas ini melawan semua lainnya". Murah, tetapi datanya menjadi tidak seimbang dan skor antar model harus dibandingkan dengan hati-hati.
- **One-vs-One** — latih $K(K-1)/2$ model untuk setiap pasangan kelas, lalu putuskan dengan suara terbanyak. Modelnya lebih banyak tetapi masing-masing lebih kecil; ini bawaan `SVC` di scikit-learn.

## Support Vector Regression

Gagasan margin dapat dipinjam untuk regresi.

- Alih-alih memisahkan kelas, SVR mencari pita selebar $\epsilon$ di sekitar fungsi, dan galat yang masih di dalam pita itu **tidak didenda sama sekali**.
- Hanya titik di luar pita yang menjadi support vector dan membentuk model.
- Efeknya: SVR tahan terhadap pencilan kecil, dan bersama kernel RBF ia dapat mencocokkan hubungan tak linier.

Pada $K = 10$ kelas, One-vs-One berarti 45 model. Karena biaya latih SVM tumbuh lebih cepat daripada linier terhadap jumlah data, 45 model kecil sering justru lebih cepat daripada 10 model yang memakai seluruh data.

# Naive Bayes dan SVM berhadapan langsung

| | **Naive Bayes** | **Support Vector Machine** |
|---|---|---|
| Jenis model | Generatif dan probabilistik; memodelkan $P(x \mid v)$ lalu membalikkannya | Diskriminatif dan geometris; langsung mencari batas pemisah |
| Asumsi utama | Fitur independen bersyarat diberikan kelas; jelas keliru tetapi menyederhanakan | Tidak berasumsi tentang sebaran; hanya bahwa margin lebar itu baik |
| Data yang cocok | Dimensi sangat tinggi dan jarang, fitur kategorikal atau hitungan, data latih sedikit | Data menengah ($n$ ribuan sampai puluhan ribu), fitur numerik, batas rumit |
| Biaya latih | $O(nd)$ — satu lintasan menghitung frekuensi | $O(n^2)$ sampai $O(n^3)$ untuk kernel; berat begitu $n$ melewati ratusan ribu |
| Biaya inferensi | $O(d)$ — menjumlahkan $d$ logaritma | $O(\lvert SV\rvert \times d)$ untuk kernel; $O(d)$ bila kernelnya linear |
| Keterbacaan | Tinggi; kontribusi tiap kata atau atribut dapat dibaca langsung | Rendah dengan kernel; cukup terbaca bila linear karena $w$ dapat diperiksa |
| Keluaran peluang | Ada, tetapi buruk kalibrasinya | Tidak ada secara alami; perlu *Platt scaling* yang berbiaya tambahan |
| Penskalaan fitur | Tidak perlu | Wajib |
| Hiperparameter | Praktis hanya $\alpha$ pada smoothing | $C$, pilihan kernel, dan $\gamma$ atau $d$ — perlu *grid search* |

# Jadi kapan memilih yang mana

**Pilih naive Bayes bila**

- Anda perlu *baseline* hari ini juga
- Datanya teks berdimensi tinggi dan jarang, atau data latihnya sedikit dibanding jumlah fiturnya
- Model harus dilatih ulang sangat sering
- Anda perlu menjelaskan mengapa satu surel ditandai spam

**Pilih SVM bila**

- Data berukuran menengah dan akurasi lebih berharga daripada waktu latih
- Batas antar kelas tampak tidak linier
- Fiturnya jelas saling berkaitan, sehingga asumsi naive Bayes merugikan
- Anda punya anggaran waktu untuk *grid search*

**Intinya**

Keduanya bukan pesaing yang saling menggantikan, melainkan dua titik berbeda pada pertukaran bias–varians dan biaya–akurasi dari Pertemuan 1. Untuk teks, urutan yang lazim: naive Bayes dulu, lalu SVM linear pada TF-IDF, baru RBF.

Sumber: Mitchell Bab 6; Bishop Bab 7; Hastie dkk. Bab 12.

# Tugas untuk pertemuan berikutnya

1. Baca Mitchell Bab 6.9–6.10 dan Hastie Bab 12.1–12.3.
2. **Pengklasifikasi teks dengan naive Bayes.** Ambil satu himpunan data teks berlabel, misalnya SMS Spam Collection atau `20 Newsgroups` empat kategori.
   - Bangun bag-of-words, latih `MultinomialNB`, laporkan akurasi, presisi, *recall*, dan matriks kebingungan untuk tiga nilai $\alpha$.
   - Tampilkan sepuluh kata dengan bukti terkuat bagi tiap kelas.
3. **SVM dengan beberapa kernel.** Pada data yang sama atau satu data numerik pilihan Anda, latih SVM dengan kernel linear, polinomial ($d = 2$ dan $d = 3$), dan RBF.
   - *Grid search* atas $C$ dan $\gamma$ pada skala logaritmik dengan validasi silang 5 lipatan; catat akurasi, jumlah support vector, dan waktu latih, lalu ulangi sekali tanpa penskalaan fitur untuk melihat pengaruhnya.
4. Tulis satu halaman kesimpulan: model mana yang Anda pakai di produksi, dengan alasan yang menyebut **akurasi maupun ongkos**.

**Pertemuan depan:** kNN yang menunda semua pekerjaan sampai ada pertanyaan, dan regresi linier yang justru punya jawaban tertutup.

# Terima kasih

**Pertanyaan?**