7

Support Vector Classifier

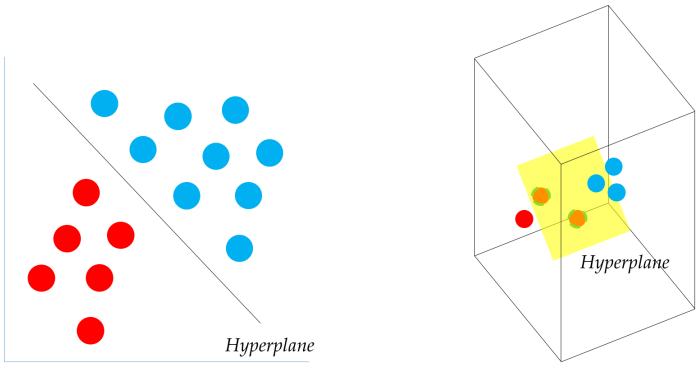
> "More data beats clever algorithms, but better data beats more data."
>
> Peter Norvig

Saat membaca judul bab ini, kamu mungkin berpikir bahwa *support vector machine* akan dibahas. *Support vector classifier* dan *support vector machine* adalah dua hal yang berbeda, walau kedua istilah tersebut sering digunakan pada konteks yang mirip [17]. *Support vector classifier* sesungguhnya adalah konsep yang lebih sederhana. Walaupun bab ini menyinggung kulit *support vector machine*, kami tidak membahas secara rinci. Akan tetapi, kami berharap pembaca mampu mendapatkan intuisi. Kamu dapat menganggap bab ini sebagai kelanjutan cerita bab 5. Referensi utama bab ini adalah buku karangan James et al. [17].

7.1 Maximal Margin Classifier

Ingat kembali kedua bab sebelumnya bahwa model klasifikasi mencari suatu *decision boundary* untuk memisahkan data pada kelas satu dan kelas lainnya. Apabila kamu memiliki data berdimensi dua, maka *decision boundary* yang kita dapat berupa garis. Pada data tiga dimensi, *decision boundary* berupa sebuah bidang (*plane*). Sebagai ilustrasi, lihatlah Gambar 7.1. Secara umum, konsep bidang pemisah disebut sebagai *hyperplane*. Untuk data berdimensi F, bidang pemisah kita memiliki dimensi $F - 1$.

Secara matematis, *hyperplane* didefinisikan pada persamaan 7.1 (dapat ditulis ulang seperti persamaan 7.2), dimana x melambangkan suatu fitur, $\mathbf{x}$ adalah *input* dalam bentuk *feature vector* dan F melambangkan banyaknya fitur. Ingat kembali, ruas kiri pada persamaan adalah bentuk dasar pada model linear. Dengan demikian, kita mengasumsikan bahwa data dapat dipisahkan



Gambar 7.1. Ilustrasi *hyperplane*

secara linear.

$$x_1 w_1 + x_2 w_2 + \cdots + x_F w_F + b = 0 \tag{7.1}$$

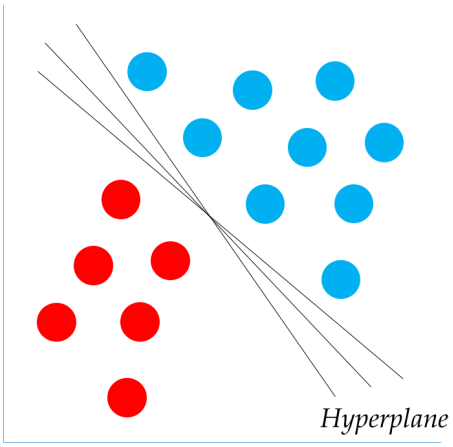
$$\mathbf{x} \cdot \mathbf{w} + b = 0 \tag{7.2}$$

Untuk permasalahan klasifikasi dua kelas, kita dapat memisahkan keputusan berdasarkan letak data pada *hyperplane*, misal di atas atau di bawah *hyperplane* pada Gambar 7.1. Secara lebih matematis, seperti pada persamaan 7.3 dan 7.4. Konsep ini mirip seperti yang sudah dijelaskan pada bab 5 tentang melakukan *binary classification* menggunakan fungsi *sign* dan *thresholding*.

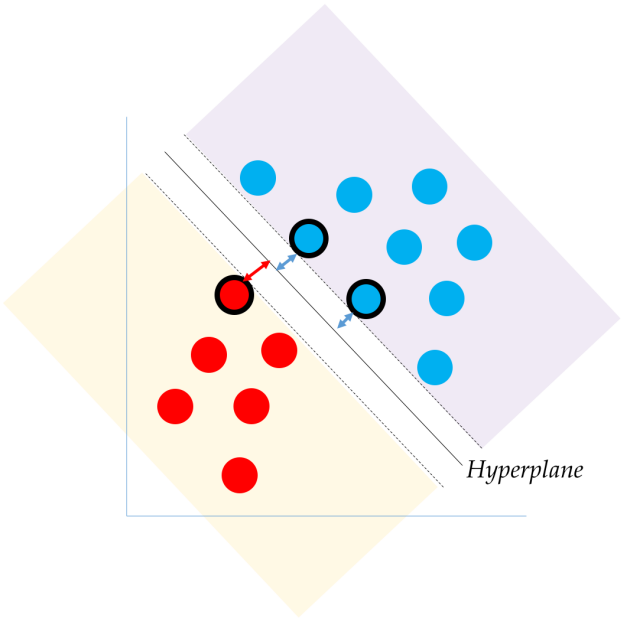
$$\text{if } \mathbf{x} \cdot \mathbf{w} + b > 0, \text{then class } A \tag{7.3}$$

$$\text{if } \mathbf{x} \cdot \mathbf{w} + b < 0, \text{then class } B \tag{7.4}$$

Apabila kita memang mampu memisahkan data dua kelas secara sempurna dengan suatu *hyperplane* (*linearly separable*), pilihan *hyperplane* yang dapat kita buat tidaklah satu. Artinya, kita dapat menggeser-geser garis pembatas, disamping tetap memisahkan data secara sempurna, seperti diilustrasikan pada Gambar 7.2. *Hyperplane* terbaik dari beberapa pilihan yang ada adalah yang memiliki *maximal margin*. Artinya, suatu *hyperplane* yang memiliki jarak terjauh terhadap data pada suatu kelas. Dengan demikian, ada jarak yang besar dari *hyperplane* dengan data. Ilustrasi diberikan pada Gambar 7.3. Kita harap, suatu *hyperplane* yang memiliki *margin* besar dapat melakukan klasifikasi dengan baik pada data yang belum kita lihat, karena kita memberikan *margin* yang besar untuk suatu data baru masuk ke daerah kelas masing-masing. Bentuk lingkaran yang memiliki *border* berwarna hitam pada Gambar 7.4 menandakan data terluar pada masing-masing kelas, dikenal sebagai ***support vectors***. Garis putus-putus melambangkan garis yang dibentuk oleh masing-masing *support vectors* untuk masing-masing kelas (*margin*).



Gambar 7.2. Ilustrasi banyak pilihan *hyperplane*. Garis hitam melambangkan opsi *hyperplane* untuk memisahkan data secara sempurna.

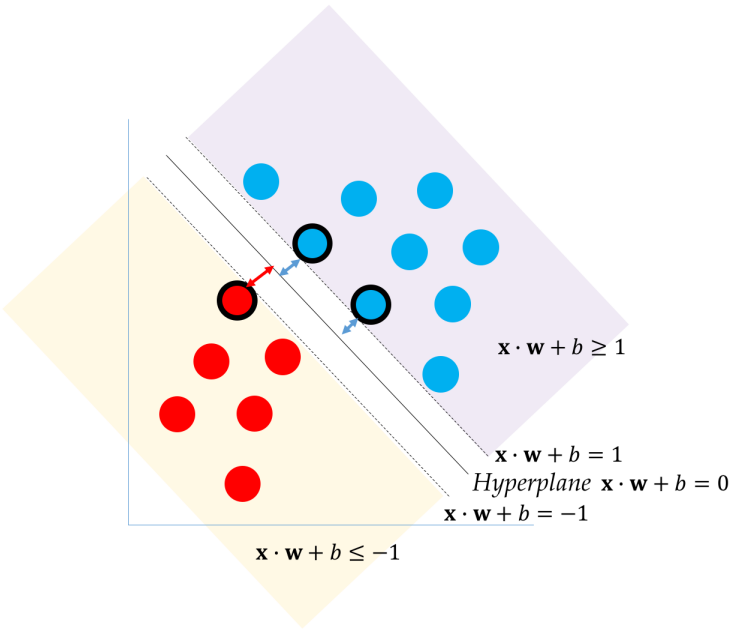


Gambar 7.3. *Maximal Margin Hyperplane*. Bentuk lingkaran yang memiliki *border* berwarna hitam menandakan data terluar pada masing-masing kelas, dikenal sebagai *support vectors*. Garis putus-putus melambangkan garis yang dibentuk oleh masing-masing *support vectors* untuk masing-masing kelas (*margin*)

Apabila kita definisikan (simbolkan) kelas pertama dengan *output* bernilai 1 dan kelas kedua bernilai -1 (ingat kembali materi fungsi *sign*), maka *support vectors* adalah poin $\mathbf{x}$ yang memenuhi kriteria pada persamaan 7.5. Dengan demikian, semua data berlabel 1 dan -1 memenuhi persamaan 7.6, dimana y melambangkan kategori. Hal inilah yang disebut sebagai ***maximal margin classifier***. Kita mencari *support vectors* yang memberikan *hyperplane* yang memiliki *maximal margin*. Lihatlah ilustrasi pada Gambar 7.4!

$$|\mathbf{x} \cdot \mathbf{w} + b| = 1 \tag{7.5}$$

$$y_i(\mathbf{x} \cdot \mathbf{w} + b) \geq 1 \tag{7.6}$$



Gambar 7.4. *Maximal Margin Classifier*

Misalkan kita ambil satu *support vector* dari masing-masing kelas. Pada kelas pertama, ia memenuhi $\mathbf{x}^{c_1} \cdot \mathbf{w} + b = 1$. Pada kelas kedua, ia memenuhi $\mathbf{x}^{c_2} \cdot \mathbf{w} + b = -1$. Apabila kita kurangi kedua persamaan tersebut, kita mendapatkan persamaan 7.7. Persamaan 7.8 memberikan perhitungan *margin* antara *support vectors* dan *hyperplane* yang memberikan nilai maksimal.

$$\mathbf{w} \cdot (\mathbf{x}^{c_1} - \mathbf{x}^{c_2}) = 2 \tag{7.7}$$

$$\frac{\mathbf{w}}{\|\mathbf{w}\|} \cdot (\mathbf{x}^{c_1} - \mathbf{x}^{c_2}) = \frac{2}{\|\mathbf{w}\|} \tag{7.8}$$

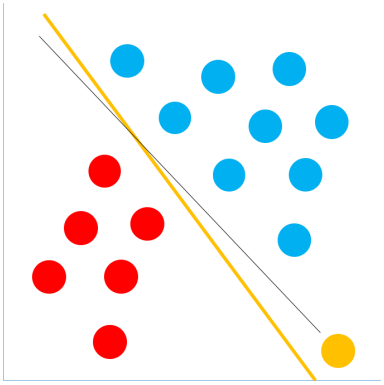
Sekarang, kita formalisasi *maximal margin classifier* secara matematis. Objektifnya adalah memaksimalkan *margin* (persamaan 7.9) dengan menjaga setiap *training data* diklasifikasikan dengan benar (persamaan 7.10).

$$\text{Objective} : \text{maximize Margin} = \frac{2}{\|\mathbf{w}\|} \tag{7.9}$$

$$\text{Subject to} : y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1 \tag{7.10}$$

Tidak sama dengan *model linear* yang sudah dijelaskan sebelumnya, kita ingin mengoptimasi sebuah fungsi sembari memenuhi kendala (*constraint*) dari fungsi lain. Ini adalah bentuk *integer linear programming*, dan solusi untuk *maximal margin classifier* dapat diselesaikan menggunakan *lagrange multiplier*. Untuk detail penyelesaiannya, silahkan membaca sumber lainnya.

Seperti yang kamu sadari, *maximal margin classifier* hanya bergantung pada subset *training data* yang dipilih sebagai *support vectors*. Dengan demikian, metode ini sangat sensitif terhadap tiap-tiap observasi. Satu observasi baru yang menjadi *support vector* dapat merubah *decision boundary*. Kita kenal ini sebagai *overfitting*. Ilustrasi permasalahan ini diberikan pada Gambar 7.5. Selain itu, *maximal margin classifier* mengasumsikan bahwa data bersifat *linearly separable*, walaupun kebanyakan data tidak bersifat demikian.



Gambar 7.5. *Maximal Margin Classifier* sangatlah sensitif terhadap perubahan *training data*. Lingkaran berwarna oranye melambangkan *training data* baru. Akibat kemuculan data baru ini, *decision boundary* awal (garis berwarna hitam) berubah secara cukup dramatis (garis berwarna oranye)

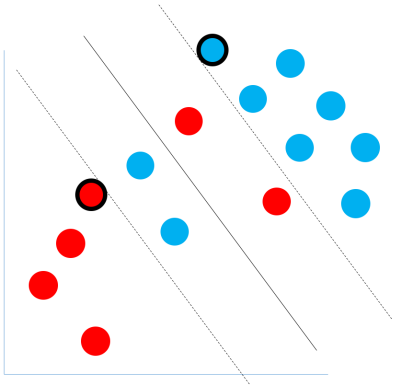
7.2 Support Vector Classifier

Support Vector Classifier adalah ekstensi dari *maximal margin classifier*. Ide utamanya adalah relaksasi kendala pada persamaan 7.10. Sebelumnya, *maximal margin classifier* mengasumsikan bahwa data bersifat *linearly separable* dan dipisahkan secara sempurna. Akan tetapi, kenyataan tidaklah demikian. Ide *support vector classifier* adalah memperbolehkan beberapa data diklasifikasikan dengan salah. Kita modifikasi *constraint* persamaan 7.10 menjadi persamaan 7.11 untuk memperbolehkan model salah mengklasifikasikan data dengan parameter kontrol ϵ, melambangkan apakah suatu observasi boleh berada pada ruang yang tidak tepat. Kita juga dapat membatasi seberapa banyak kesalahan yang dibuat dengan *constraint* baru yang diberikan pada persamaan 7.12.

$$\text{Subject to} : y_i(\mathbf{x}_i \cdot \mathbf{w} + b) \geq 1(1 - \epsilon_i) \tag{7.11}$$

$$\epsilon_i \geq 0; \sum \epsilon_i \leq C \tag{7.12}$$

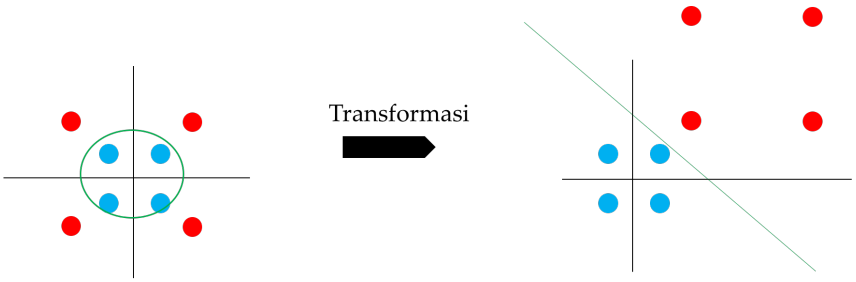
Ilustrasi *support vector classifier* diberikan pada Gambar 7.6. Disini, kita sedikit modifikasi definisi *support vectors* sebagai observasi yang tepat jauh pada *margin* atau pada daerah yang tidak sesuai dengan kelasnya [17].



Gambar 7.6. *Support Vector Classifier*. Lingkaran dengan *border* berwarna hitam melambangkan *support vectors*, garis putus-putus melambangkan *margin*

Walaupun memperbolehkan beberapa observasi boleh tidak berada pada ruang yang tepat (berdasarkan *margin*), *support vector classifier* masih memiliki asumsi bahwa *decision boundary* berbentuk suatu fungsi linear. Ini adalah batasan utama model ini.

7.3 Support Vector Machine



Gambar 7.7. Ilustrasi transformasi data. Garis berwarna hijau melambangkan *decision boundary*

Berhubung banyak *decision boundary* tidak dapat dimodelkan dengan suatu bentuk atau persamaan linear, kita harus memodelkan *decision boundary* sebagai fungsi non-linear. Ekstensi *support vector classifier* adalah ***support vector machine*** yang menggunakan teknik ***kernel***. Suatu fungsi *kernel* mentransformasi data ke ruang (*space* atau dimensi) lainnya (biasanya ke dimensi lebih tinggi). Data yang ditransformasi ini (pada dimensi lebih tinggi), kita harapkan dapat dipisahkan dengan fungsi linear. Apabila kita lihat balik pada dimensi asli, *decision boundary* pada dimensi yang baru memodelkan suatu *decision boundary* non-linear pada dimensi aslinya. Hal ini diilustrasikan pada Gambar 7.7. Fungsi *kernel* ini ada banyak, beberapa yang terkenal diantaranya[1]:

1. Polynomial Kernel (persamaan 7.13)
2. Radial Basis Function Kernel (persamaan 7.14, σ melambangkan varians)

Yang membedakan *support vector machine* dan *support vector classifier* adalah mengintegrasikan fungsi *kernel* pada model.

$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^d \tag{7.13}$$

$$k(x, y) = \exp\left(-\frac{\|x - y\|^2}{2\sigma^2}\right) \tag{7.14}$$

Sebelum membahas bagaimana fungsi *kernel* diintegrasikan pada SVM, kita sedikit bahas kembali *support vector classifier*. Ingat kembali *support vector classifier* mencari *maximal margin* (persamaan 7.9) dengan kendala persamaan 7.11 dan 7.12. Suatu *support vector classifier* yang memenuhi seluruh kendala tersebut dapat direpresentasikan sebagai persamaan 7.15 dimana $\mathbf{x}'$

[1] `https://data-flair.training/blogs/svm-kernel-functions/`

melambangkan suatu data baru, $\mathbf{x}_i$ adalah suatu instans pada *training data* dan N adalah banyaknya *training data*. Operasi $\langle \mathbf{x}', \mathbf{x}_i \rangle$ melambangkan *inner product*. Cara menghitung *inner product* dari dua vektor diberikan pada persamaan 7.16, dimana F melambangkan panjangnya vektor. *Inner product* dapat diinterpretasikan sebagai perhitungan kemiripan dua vektor.

$$f(\mathbf{x}') = \beta_0 + \sum_{i=1}^{N} \alpha_i \langle \mathbf{x}', \mathbf{x}_i \rangle \tag{7.15}$$

$$\langle \mathbf{a}, \mathbf{b} \rangle = \sum_{j=1}^{F} a_j b_j \tag{7.16}$$

Untuk menghitung parameter β dan α pada persamaan 7.15, kita membutuhkan $\binom{N}{2}$ kombinasi pasangan instans yang ada pada *training data*. Akan tetapi, pada saat melewatkan suatu *input* baru pada persamaan tersebut, kita sebenarnya cukup menghitung seberapa kedekatan (kemiripan) antara *input* dan *support vectors*. Hal ini disebabkan α bernilai 0 untuk instans selain *support vectors*, i.e., diatur hanya bernilai *nonzero* untuk *support vectors*. Artinya, keputusan klasifikasi bergantung pada pilihan *support vectors*. Dengan demikian, kita dapat menulis kembali persamaan 7.15 sebagai persamaan 7.17.

$$f(\mathbf{x}') = \beta_0 + \sum_{\mathbf{x}_i \in S} \alpha_i \langle \mathbf{x}', \mathbf{x}_i \rangle \tag{7.17}$$

Ketika menggunakan fungsi *kernel*, kita menghitung kemiripan (*inner product*) antara dua vektor pada dimensi transformasi. Kita mengghanti *inner product* menggunakan suatu fungsi *kernel*, ditulis sebagai persamaan 7.18. Persamaan inilah yang dikenal sebagai ***support vector machine*** (SVM). Untuk memodelkan *non-linear decision boundary*, kita menggunakan fungsi yang bersifat non-linear sebagai *kernel*.

$$f(\mathbf{x}') = \beta_0 + \sum_{\mathbf{x}_i \in S} \alpha_i k(\mathbf{x}', \mathbf{x}_i) \tag{7.18}$$

Untuk memahami SVM lebih dalam, kamu bisa membaca buku karangan Bishop [8].

7.4 Klasifikasi lebih dari dua kelas

Penjelasan pada bab ini berfokus pada *binary classification*. Memang, *maximal margin classifier* dan ekstensinya difokuskan pada permasalahan *binary classification*. Kita dapat mengekstensinya untuk *multi-class classification*. Ada dua teknik yang umumnya digunakan, yaitu *one versus one* dan *one versus all* seperti yang sudah dijelaskan pada bab 5. Kita dapat mendekomposisi *classifier* untuk *multi-label classification* menjadi beberapa *binary classifiers*, seperti yang sudah dijelaskan pada bab 5.

7.5 Tips

Untuk memahami materi yang disampaikan pada bab ini secara lebih dalam, kami menyarankan kamu untuk mempelajari *optimization theory* dan *operation research* (*i.e., integer linear programming*). Penulis harus mengakui tidak terlalu familiar dengan teori-teori tersebut. Sejarah dan perkembangan *support vector machine* dapat dibaca pada paper-paper yang diberikan di pranala `http://www.svms.org/history.html`. Walaupun penjelasan pada bab ini hanya bersifat "kulit"-nya saja, kami harap pembaca mampu mendapatkan intuisi.

Soal Latihan

7.1. Metode *Kernel*
Baca dan jelaskanlah konsep metode *kernel* yang dijelaskan pada buku Bishop [8]!

7.2. Fungsi *Kernel*
Jelaskanlah macam-macam fungsi *kernel* yang sering digunakan untuk *support vector machine*!

7.3. SVM-rank
Walau umumnya digunakan untuk permasalahan klasifikasi, SVM juga dapat diaplikasikan pada permasalahan *ranking* (*learning to rank*), dikenal sebagai SVM-*rank*[2]. Permasalahan ini adalah inti dari *search engine*. Jelaskanlah bagaimana cara kerja SVM-*rank*!

[2] `https://www.cs.cornell.edu/people/tj/svm_light/svm_rank.html`