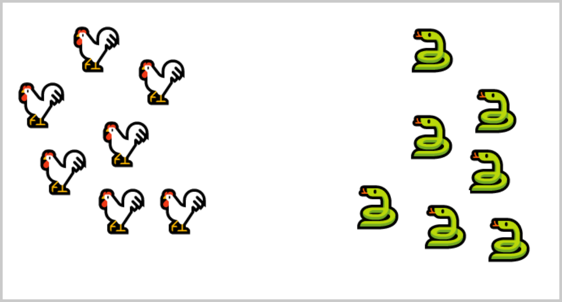
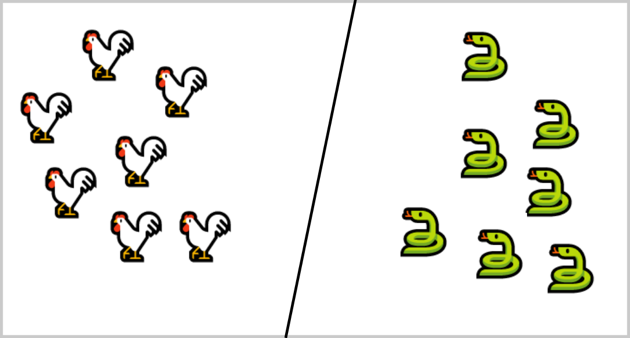
Modul berikut akan membahas tentang SVM pada kasus klasifikasi. Untuk memahami bagaimana algoritma support vector machine atau SVM bekerja pada kasus klasifikasi, bayangkan kita memiliki sebuah kebun binatang mini. Di kebun binatang tersebut terdapat dua jenis binatang yaitu, ayam hias dan ular. Sebagai seorang ML engineer, kita ingin mengembangkan sebuah model yang mampu membedakan antara ayam dan ular piton agar bisa menempatkan kedua hewan tersebut di kandang yang berbeda. Kita tentunya tak mau menyatukan ayam dan ular dalam satu kandang yang sama ya.

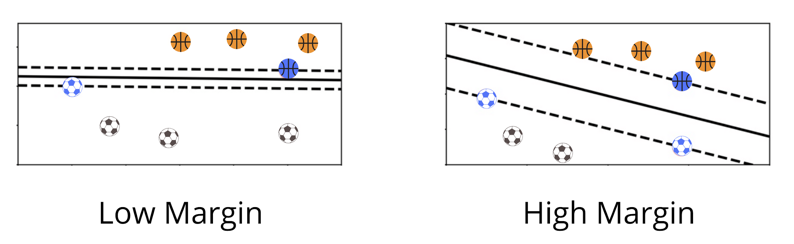
[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Kita bisa membuat sebuah model klasifikasi yang memisahkan antara kedua kelas tersebut menggunakan Support Vector Machine. Menurut Aurelien Geron dalam buku *Hands on Machine Learning*, SVM bekerja dengan membuat decision boundary atau sebuah bidang yang mampu memisahkan dua buah kelas. Pada masalah ini decision boundary yang mampu memisahkan kelas ayam dan kelas ular adalah sebuah garis lurus yang dapat dilihat pada gambar.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Lantas bagaimana SVM membuat sebuah decision boundary tersebut?

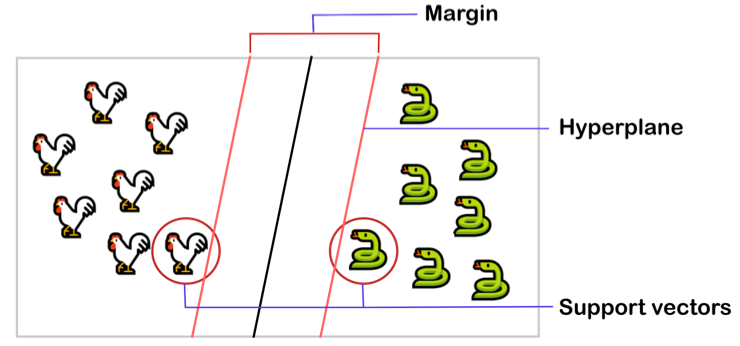
Pertama SVM mencari support vector pada setiap kelas. Support vector adalah sampel dari masing-masing kelas yang memiliki jarak paling dekat dengan sampel kelas lainnya. Pada contoh dataset bola basket dan bola kaki di bawah, support vector adalah bola basket dan bola kaki yang memiliki warna biru.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Setelah support vector ditemukan, SVM menghitung margin. Margin bisa kita anggap sebagai jalan yang memisahkan dua kelas. Margin dibuat berdasarkan support vector di mana support vector bekerja sebagai batas tepi jalan, atau sering kita kenal sebagai bahu jalan. SVM mencari margin terbesar atau jalan terlebar yang mampu memisahkan kedua kelas.

Pada dataset bola basket dan bola kaki di atas SVM akan memilih margin di sebelah kanan karena ‘jalan’ atau margin pada gambar sebelah kanan lebih lebar dari ‘jalan’ di sebelah kiri. Oleh karena itu, gambar sebelah kanan disebut sebagai *high margin classification* dan gambar di sebelah kiri disebut *low margin classification.*

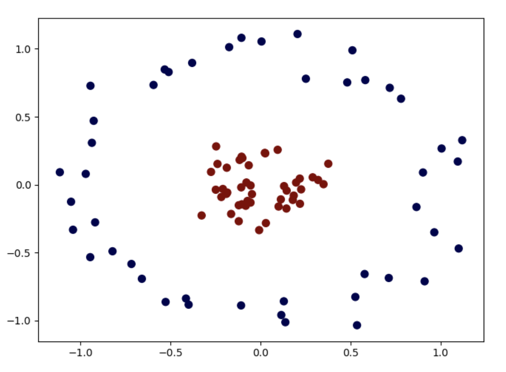
Kembali lagi ke kasus klasifikasi ayam dan ular, sampel ayam dan ular yang berada dalam lingkaran merah adalah support vector. Kemudian kita mencari jalan terlebar dari 2 support vector. Setelah menemukan jalan terlebar, decision boundary lalu digambar berdasarkan jalan tersebut.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Decision boundary adalah garis yang membagi jalan atau margin menjadi 2 bagian yang sama besar. Hyperplane adalah bidang yang memisahkan kedua kelas, sedangkan margin adalah lebar ‘jalan’ yang membagi kedua kelas. Selamat! Sekarang Anda sudah paham bagaimana support vector machine bekerja dalam masalah klasifikasi.

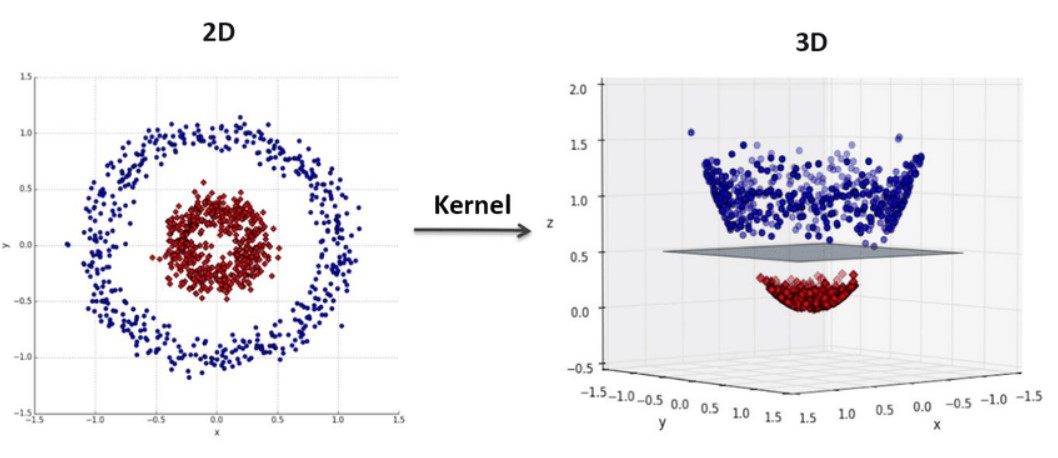
**SVM Klasifikasi non Linier**

Sebelumnya kita sudah belajar tentang support vector classifier untuk kasus linear. Support vector classifier bekerja dengan mencari margin terbesar, atau jalan terlebar yang mampu untuk memisahkan 2 buah kelas. Permasalahannya, data di lapangan jauh lebih kompleks dibanding data ayam hias dan ular seperti di atas. Bagaimana jika data yang kita miliki terlihat seperti ini?

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Data di atas merupakan data yang tidak bisa dipisahkan secara linier sehingga kita menyebutnya sebagai data non-linear. Pada data non-linear, decision boundary yang dihitung algoritma SVM bukan berbentuk garis lurus. Meski cukup rumit dalam menentukan decision boundary pada kasus ini, tapi kita juga mendapatkan keuntungan, yaitu, bisa menangkap lebih banyak relasi kompleks dari setiap data poin yang tersebar.

Untuk data seperti di atas, Support Vector Classifier menggunakan sebuah metode yaitu “kernel trick” sehingga data dapat dipisahkan secara linier. Apa itu trik kernel? Ia adalah sebuah metode untuk mengubah data pada dimensi tertentu (misal 2D) ke dalam dimensi yang lebih tinggi (3D) sehingga dapat menghasilkan hyperplane yang optimal. Perhatikan gambar berikut [[11]](https://arxiv.org/pdf/2004.06077.pdf).

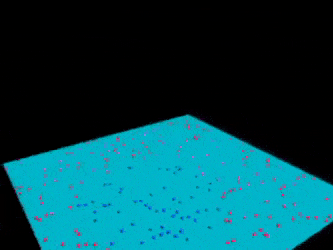
[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Bagaimana trik kernel bekerja?

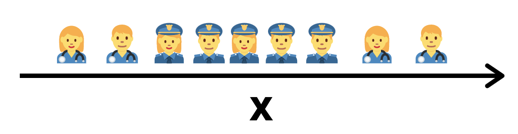
Pertama, kita perlu menghitung skor jarak dari dua titik data, misal x\_i dan x\_j. Skor akan bernilai lebih tinggi untuk titik data yang lebih dekat, dan sebaliknya. Lalu kita gunakan skor ini untuk memetakan data pada dimensi yang lebih tinggi (3D). Teknik ini berguna untuk mengurangi waktu dan sumber daya komputasi, terutama untuk data berjumlah besar. Hal ini juga mencegah kebutuhan akan proses transformasi yang lebih kompleks. Itulah mengapa teknik ini sering disebut sebagai trik kernel.

Seperti yang ditunjukkan gambar di atas, pemetaan titik data dari ruang 2D menjadi 3D menggunakan fungsi kernel. Titik-titik merah yang sebelumnya berada di tengah sekarang berada di dalam bidang vertikal dengan posisi lebih rendah setelah diubah menjadi ruang 3D. Titik data yang sebelumnya sulit dipisahkan, sekarang dapat dengan mudah dipisahkan dengan teknik kernel.

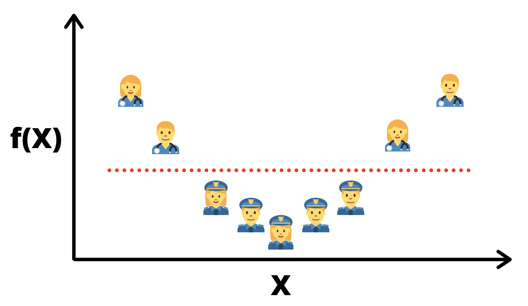
Animasi yang dibuat oleh Udiprod [[12]](https://www.youtube.com/watch?v=3liCbRZPrZA) berikut dapat membantu kita untuk melihat bagaimana data 2 dimensi -seperti pada gambar sebelumnya- diubah ke dalam ruang 3 dimensi, sehingga data dapat dipisahkan secara linier.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Bagaimana, sampai sini sudah paham, kan? Kami akan memberikan satu contoh lagi. Mari perhatikan ilustrasi berikut tentang bagaimana trik kernel bekerja pada data yang lebih sederhana.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Data di atas adalah data 1 dimensi dengan 2 buah kelas yaitu dokter dan polisi. Data di atas bukan data linier karena kita tak dapat menggambar satu garis lurus untuk memisahkan 2 kelas yang ada. Bagaimana cara kita bisa menggambar garis lurus yang bisa memisahkan 2 kelas tersebut? Betul, kita akan menggunakan trik kernel untuk mengubah data tersebut ke dalam dimensi yang lebih tinggi seperti ke dalam bidang 2 dimensi.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

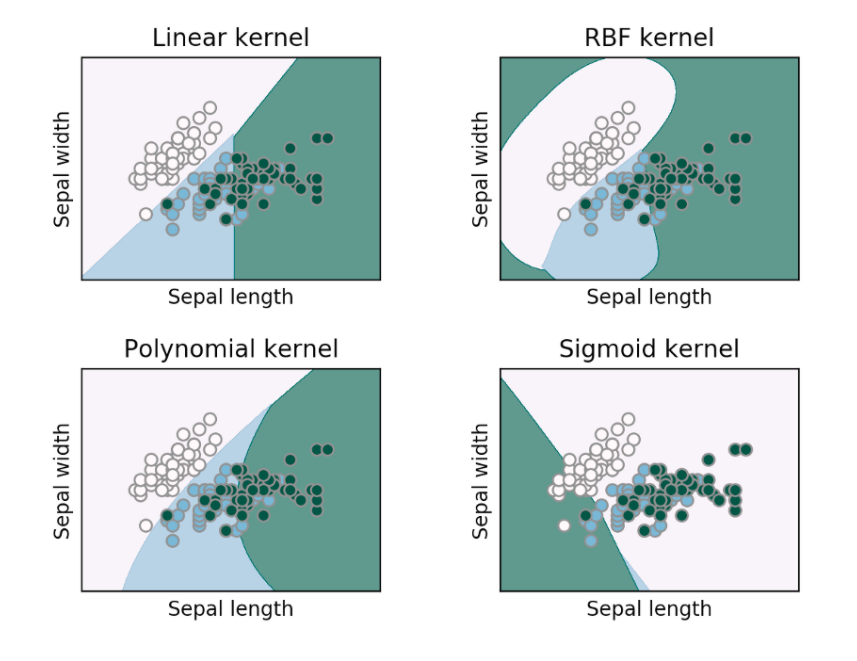
Ketika data sudah diubah ke dalam bidang 2 dimensi, sebuah garis lurus bisa digambar untuk memisahkan 2 kelas. Trik kernel menggunakan fungsi matematis yang bisa mengubah data dari dimensi tertentu ke dimensi yang lebih tinggi sehingga kelas-kelas pada data dapat dipisah secara linier.

Bagaimana, mudah bukan? Berikut adalah beberapa fungsi kernel yang perlu Anda ketahui.

* [Linear](https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC)
* [RBF](https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html#sphx-glr-auto-examples-svm-plot-rbf-parameters-py)*(Radial Basis Function)* atau*Gaussian kernel*
* Polinomial
* Sigmoid

Tertarik mengetahui fungsi matematis di balik kernel? Kunjungi [tautan](https://data-flair.training/blogs/svm-kernel-functions/) berikut yah.

Untuk melihat bagaimana pemisahan data dan seperti apa bentuk hyperplane pada masing-masing fungsi kernel, perhatikan gambar yang dibuat oleh H. Marius[[13]](https://towardsdatascience.com/multiclass-classification-with-support-vector-machines-svm-kernel-trick-kernel-functions-f9d5377d6f02)berikut.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Sampai di sini kita telah mempelajari bagaimana SVM bekerja untuk menyelesaikan permasalahan klasifikasi. Lantas, bagaimana SVM Classifier ini diterapkan dalam kehidupan sehari-hari?

Salah satu contoh aplikasi SVM dalam kehidupan sehari-hari adalah fitur deteksi wajah. Pengenalan wajah telah berkembang menjadi topik penelitian utama dalam bidang *computer vision.* Sistem pengenalan wajah memiliki banyak fitur/kelas (individu) dan hanya memiliki beberapa data gambar (sampel) per orang, bahkan kadang hanya ada satu sampel pelatihan untuk setiap orang. Permasalahan dengan jumlah fitur lebih banyak dari jumlah sampel ini efektif jika dipecahkan dengan algoritma SVM.

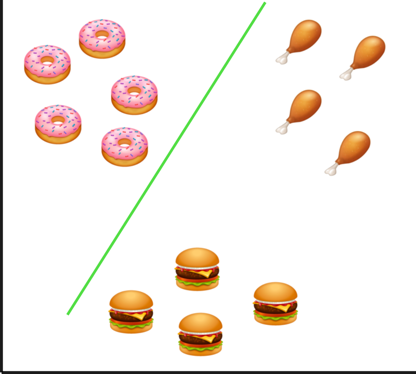
Aplikasi lain SVM yang juga menarik adalah di bidang bioinformatika. Dalam bidang komputasi biologi, deteksi homologi jarak jauh pada protein *(protein remote homology detection)* adalah permasalahan yang umum. Metode paling efektif untuk menyelesaikan permasalahan ini adalah dengan SVM. Dalam beberapa tahun terakhir, algoritma SVM telah diterapkan secara luas untuk proses deteksi ini. Fungsi kernel pada SVM digunakan untuk mengidentifikasi sekuens biologis dan membantu menemukan kesamaan antara urutan protein yang berbeda.

**SVM untuk Klasifikasi Multi-kelas**

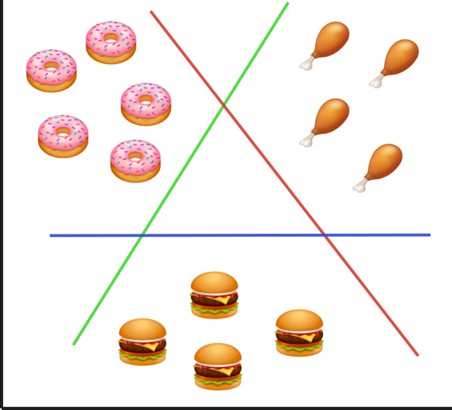
SVM sejatinya merupakan binary classifier atau model untuk klasifikasi 2 kelas. Namun, SVM juga dapat dipakai untuk klasifikasi multi-kelas menggunakan suatu teknik yaitu “one-vs-rest”.

Pada masalah klasifikasi multi-kelas, SVM melakukan klasifikasi biner untuk masing-masing kelas. Model kemudian memisahkan kelas tersebut dari semua kelas lainnya, menghasilkan model biner sebanyak jumlah kelasnya. Untuk membuat prediksi, semua proses klasifikasi biner dijalankan pada tahap uji.

Sebagai contoh, jika ada 3 buah kelas: donat, ayam, dan burger, SVM akan melakukan 3 kali klasifikasi. Pertama, membangun pemisah antara kelas donat dan kelas bukan donat.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)

Kemudian membangun pemisah antara kelas ayam dan kelas bukan ayam, lalu pemisah antara kelas burger dan bukan kelas burger. Teknik inilah yang disebut dengan “One-vs-Rest”.

[](https://www.dicoding.com/academies/184/tutorials/8447?from=8442)