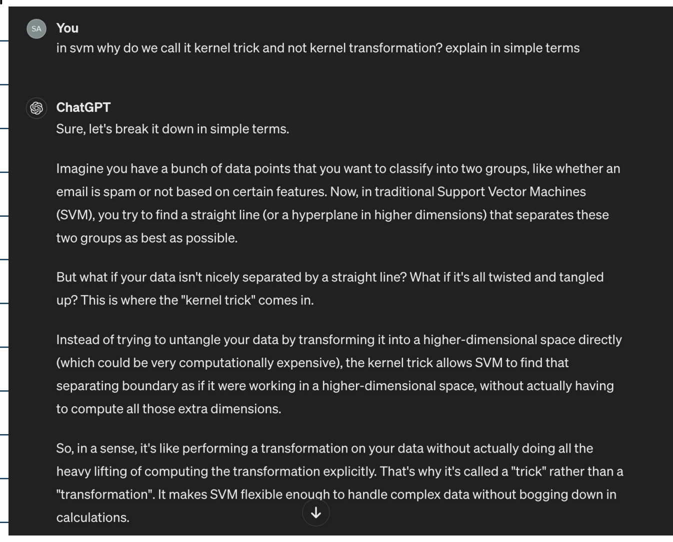
(SVC + Kernels) = SVM Non-linear data

Thursday 2 May 2024 3:24 PM

Kernel trick → Why is it called a trick & not a transformation?


You
in svm why do we call it kernel trick and not kernel transformation? explain in simple terms

ChatGPT
Sure, let's break it down in simple terms.

Imagine you have a bunch of data points that you want to classify into two groups, like whether an email is spam or not based on certain features. Now, in traditional Support Vector Machines (SVM), you try to find a straight line (or a hyperplane in higher dimensions) that separates these two groups as best as possible.

But what if your data isn't nicely separated by a straight line? What if it's all twisted and tangled up? This is where the "kernel trick" comes in.

Instead of trying to untangle your data by transforming it into a higher-dimensional space directly (which could be very computationally expensive), the kernel trick allows SVM to find that separating boundary as if it were working in a higher-dimensional space, without actually having to compute all those extra dimensions.

So, in a sense, it's like performing a transformation on your data without actually doing all the heavy lifting of computing the transformation explicitly. That's why it's called a "trick" rather than a "transformation". It makes SVM flexible enough to handle complex data without bogging down in calculations.

1) input $\xrightarrow[\text{kernels}]{}$ higher dimension

2) kernels are designed in such a way that the data is linearly separable in higher dimensions.

3) SVM

4) project down to lower dimension

Types of kernels
- linear
- rbf
- polynomial
- sigmoid

} hyperparameter

Mathematics of SVM

1) Solving constraint optimization problems

e.g. $\underset{x,y}{\operatorname{argmax}}\ x^2y$ s.t. $x^2+y^2=1$

→ The gradient of a function at a point is a vector that points in the direction of the steepest ascent/maximum change of the function at that point. The magnitude of the gradient vector is equal to the rate of increase of the function in that direction.

→ Contour lines of a function are curves that connect points where the function has the same value.

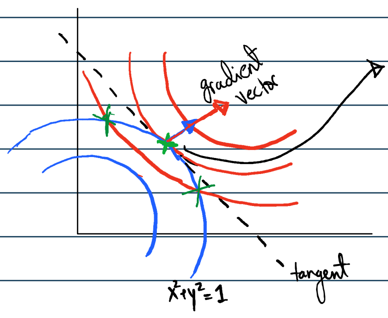
* The gradient at a point is ⊥ to the contour line passing through that point.

* The gradient points in the direction where the function increases most rapidly.

* The magnitude of the gradient indicates how steeply the function is increasing.

let $f(x,y)=x^2y$ & $g(x,y)=x^2+y^2$

Contour plots of $f(x,y)$ & $g(x,y)$



→ point which maximizes $f(x,y)$ as well as satisfies the constraint,
At this point

$$\boxed{\nabla f(x,y) = \underbrace{\lambda}_{\text{scalar}} \nabla g(x,y)}$$

direction ✓
magnitude ✗

→ langange's multiplier

$$\nabla f(x,y) = \begin{bmatrix} \partial f/\partial x \\ \partial f/\partial y \end{bmatrix} = \begin{bmatrix} 2xy \\ x^2 \end{bmatrix} \qquad \nabla g(x,y) = \begin{bmatrix} \partial g/\partial x \\ \partial g/\partial y \end{bmatrix} = \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

$$\begin{bmatrix} 2xy \\ x^2 \end{bmatrix} = \lambda \begin{bmatrix} 2x \\ 2y \end{bmatrix}$$

$2xy = \lambda 2x$ — ①
$x^2 = \lambda 2y$ — ②
$x^2 + y^2 = 1$ — ③

$$\underbrace{y = \pm \frac{1}{\sqrt{3}} \qquad x = \pm \sqrt{\frac{2}{3}}}$$

4 set of points
which satisfy
the constraint

→ Out of these 4 points $\left(\sqrt{\frac{2}{3}}, \frac{1}{\sqrt{3}}\right)$ & $\left(-\sqrt{\frac{2}{3}}, \frac{1}{\sqrt{3}}\right)$
result in max value for $f(x,y)$.

2) Lagrange multiplier

$\underset{x,y}{\text{argmax}}\ x^2y$ s.t. $x^2+y^2=1$

↓ Constraint optimization problem
↓ optimization problem

$$L(x,y,\lambda) = \underset{x,y}{\text{argmax}}\ f(x,y) - \lambda(g(x,y)-1)$$

$\frac{\partial L}{\partial x} = 0$ → $2xy = \lambda 2x$

$\frac{\partial L}{\partial y} = 0$ → $x^2 = \lambda 2y$

$\frac{\partial L}{\partial \lambda} = 0$ → $x^2+y^2=1$

3) SVM in n-Dimensions

equation of hyperplane: $w^T x + b = 0$

$$\boxed{\underset{w,b}{\text{arg min}} \left(\frac{\|w\|^2}{2} + C\,\frac{1}{n}\sum_{i=1}^{n} \xi_i \right) \text{ s.t. } \left\{ y_i(w^T x + b) \geq 1 - \xi_i \right\}}$$

soft margin formulation in n-D

4) Constrained optimization problems with inequality

e.g. $\boxed{\min_x f(x) = x^2 \text{ s.t. } x - 1 \leq 0}$

→ **KKT conditions**

→ They generalize the method of Lagrange multipliers to handle inequality constraints.

→ They generalize the method of Lagrange multipliers to handle inequality constraints.

→ KKT conditions play a key role in deriving the dual problem from the primal problem.

Primal | Lagrangian

e.g. $\boxed{\min_x f(x) = x^2 \ \text{s.t.}\ x-1 \leq 0} \longrightarrow \boxed{L(x,\lambda) = x^2 - \lambda(x-1)}$

Conditions:

1) $\frac{\partial L}{\partial x} = 0 \quad \frac{\partial L}{\partial \lambda} = 0$ 2) $x-1 \leq 0$

3) $\lambda \geq 0$ 4) $\lambda(x-1) = 0$

x & λ which satisfy these 4 conditions will be the solution to our problem

5) Duality

→ optimization theory

→ The primal problem is the original optimization problem that you are trying to solve. It involves finding the minimum or maximum of a particular objective function, subject to certain constraints.

→ The dual problem (→ easier to solve.) is a related optimization problem that is derived from the primal problem. It provides a lower or upper bound on the solution to the primal problem.

→ If strong duality holds, then solving the dual problem can directly give the solution to the primal problem.

6) SVM dual problem derivation (Hard margin SVM)

Primal form

training data $n \times m$

$$\underset{w,b}{\text{argmin}} \frac{\|w\|^2}{2} \quad \text{s.t.} \ \underbrace{y_i(w^Tx_i + b) \geq 1 \ \forall i}_{}$$

↳ for every row 1 constraint

Lagrangian

$$L(w,b,\alpha) = \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \alpha_i \left[y_i(w^Tx_i + b) - 1 \right]$$

$$L(w,b,\alpha) = \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \alpha_i y_i w x_i - \sum_{i=1}^{n} \alpha_i y_i b + \sum_{i=1}^{n} \alpha_i$$

$$\frac{\partial L}{\partial w} = \frac{2w}{2} - \sum_{i=1}^{n} \alpha_i y_i x_i = 0 \qquad \frac{\partial L}{\partial b} = -\sum_{i=1}^{n} \alpha_i y_i = 0$$

① $\boxed{w = \sum_{i=1}^{n} \alpha_i y_i x_i}$ ② $\boxed{\sum_{i=1}^{n} \alpha_i y_i = 0}$

sub ① & ② into Lagrangian form

$$L(w,b,\alpha) = \frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i x_i\right)\left(\sum_{j=1}^{n} \alpha_j y_j x_j\right) - \sum_{i=1}^{n} \alpha_i y_i x_i \left(\sum_{j=1}^{n} \alpha_j y_j x_j\right) + \sum_{i=1}^{n} \alpha_i$$

$$= -\frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i y_i x_i\right)\left(\sum_{j=1}^{n} \alpha_j y_j x_j\right) + \sum_{i=1}^{n} \alpha_i$$

primal dual
min → max
max → min

$$\boxed{\underset{\alpha_i}{\text{argmax}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \quad \text{subject to } \alpha_i \geq 0 \,,\ \sum_{i=1}^{n} \alpha_i y_i = 0}$$

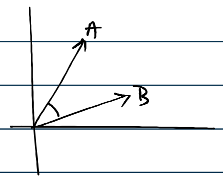
↳ dual form
easier to solve
kernel friendly

1) $\alpha > 0$ only for support vectors ; $\alpha = 0$ for non-support vectors

in dual form only support vectors are considered

7) Similarity

cosine similarity = $\frac{A \cdot B}{\|A\|\|B\|}$



if $\|A\| = 1$ & $\|B\| = 1$

$= A \cdot B$ ↳ dot product of A & B

$$\underset{\alpha_i}{\text{argmax}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) \quad \text{subject to } \alpha_i \geq 0 \,,\ \sum_{i=1}^{n} \alpha_i y_i = 0$$

→ similarity

↳ maximize the similarity of support vectors based on their sign.

8) Kernel SVM

$$\underset{\alpha_i}{\text{argmax}} \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i \cdot x_j) \quad \text{subject to } \alpha_i \geq 0 \,,\ \sum_{i=1}^{n} \alpha_i y_i = 0$$

$K(x_i, x_j)$ → kernel → similarity b/w x_i & x_j

if kernel function is $x_i \cdot x_j$ → linear SVM

→ if in the similarity part of the dual form we use some other form of mathematical function to calculate the similarity we call it kernel SVM.

9) Polynomial Kernel

$K(x_i, x_j) = (r + x_i \cdot x_j)^d$ → expansion of this will contain polynomial terms which mimics transforming our data into higher dimension without actually applying feature transformation. This way, we save memory space. → While the squared terms cater only to cicular kind of shapes. But with the help of the interaction term we will be able to map a wider variety of shapes.

10) RBF kernel

$K(x_i, x_j) = e^{-\gamma \|x_i - x_j\|^2}$ where $\gamma = 1/2\sigma^2$

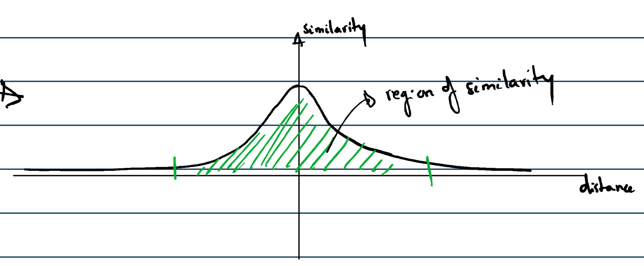
↑ hyperparameter

$K \propto \frac{1}{dist}$

↑ similarity

$\gamma\uparrow$ $\sigma\downarrow$ similarity region↓ (low bias high variance) overfitting
$\gamma\downarrow$ $\sigma\uparrow$ similarity region↑ (high bias low variance) underfitting
} bias variance tradeoff

1) Non-linear transformations
2) Local decisions
3) Flexibility
4) Universal Approximation Property



4) Universal Approximation Property

5) general purpose

* features must be normalized (b/w -1 & +1) ??

→ Infinite dimensional mapping: The RBF kernel implicitly maps input data to a ∞ dimensional feature space, which allows for even greater flexibility in forming decision boundaries. → The reason lies in e^x being expressed as infinite series.

$$e^x = \sum_{k=0}^{\infty} \frac{x^k}{k!}$$

→ Sum of infinite polynomial kernels.

→ Similarity with KNN

$$\boxed{K \text{ similar to } \sigma \propto \frac{1}{\lambda}}$$

11) Custom kernels are possible