

ML-14

ML-2 - Dr.Kavousi

- حل مسأله Karush-Kuhn-Tucker به روش این پرایم و دوچال میباشد
- این مسأله دو چال میباشد، این دو چال میباشند که میتوانند میان آنها تفاوت داشته باشند.
Dual duality gap = 0

- در حل مسأله دوچال میباشد، عبارت w باید دار نزاع میباشد از خط
که میتواند x_i را در میان دو کلاس $y_i = 1$ و $y_i = -1$ بگذارد.

+ Because this will let us solve the problem by computing the
just the inner products of x_i, x_j (which will be very
important later on when we want to solve non-linearity-
separable classification problems)

+ $\{x_1, \dots, x_n\}$ → our data , $+ y_i \in \{1, -1\}$ → the class label
of x_i

+ $\forall i \quad y_i (w^T x_i + b) \geq 1 \rightarrow$ The decision boundary should classify
all points correctly.



+ A constrained optimization problem:

$$\text{Maximize } m = \frac{2}{\|w\|} \rightarrow \text{minimize } \frac{1}{2} \|w\|^2$$

$$\text{Subject to } y_i(w^T x_i + b) \geq 1 \quad \forall i$$

Support Vectors

+ complimentary slackness condition:

$$\alpha_i \geq 0 \Leftrightarrow \mu_i^* [1 - y_i(w_i^T w^* + b^*)] = 0, \forall i$$

مقدار خوبی
مقدار نیز

$$* \text{ if } y_i(w_i^T w^* + b^*) > 1 \rightarrow \mu_i^* = 0$$

$$* \text{ else } \mu_i^* > 0 \rightarrow y_i(w_i^T w^* + b^*) = 1$$

+ support vectors are the set of x_i 's that have $\mu_i^* > 0$ and
for other training data $\mu_i^* = 0$

+ we can transform the problem to its dual:

$$L(w, b, \mu) = \frac{1}{2} w^T w + \sum_{i=1}^l \mu_i [1 - y_i(w^T x_i + b)] \quad w^* = \sum_{i=1}^l \mu_i^* y_i x_i$$

$$L(w, b, \mu) = \sum_{i=1}^l \mu_i - \frac{1}{2} \sum_{i=1}^l \sum_{j=1}^l \mu_i \mu_j y_i y_j (x_i^T x_j)$$

$$\mu_i > 0, \forall i$$

$$\text{s.t. } \sum_{i=1}^l \mu_i y_i = 0$$

+ This is a quadratic programming problem. (QP)

w^*, b^* can be recovered by:

$$w^* = \sum_{i \in S} \mu_i^* y_i x_i$$

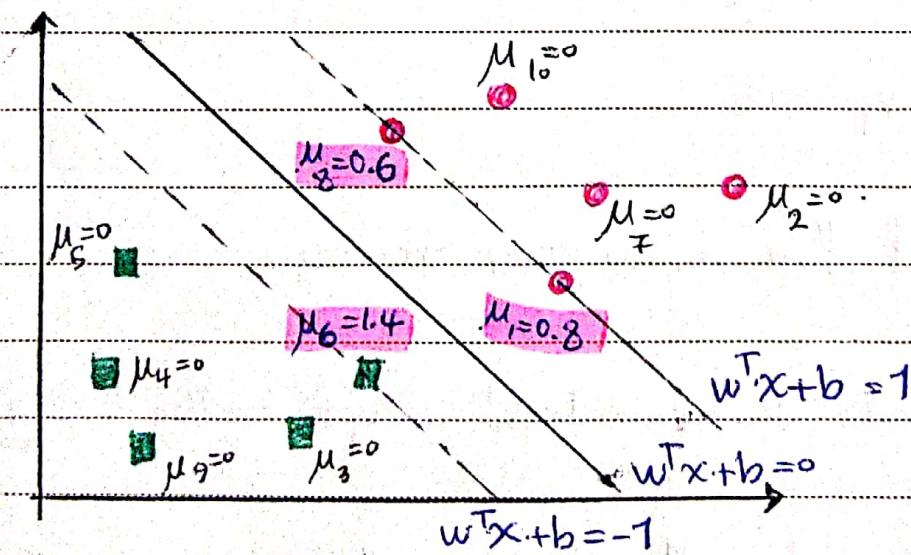
$$b^* = y_i - x_i^T w^*, \mu_i > 0$$

این بزرگتر از مقدار میانگین است + نسبت sign

$$\rightarrow f(x) = \text{sgn} \left(\sum_{i=1}^l y_i \mu_i (x \cdot x_i) + b \right)$$

Decision hyperplane

A Geometrical interpretation

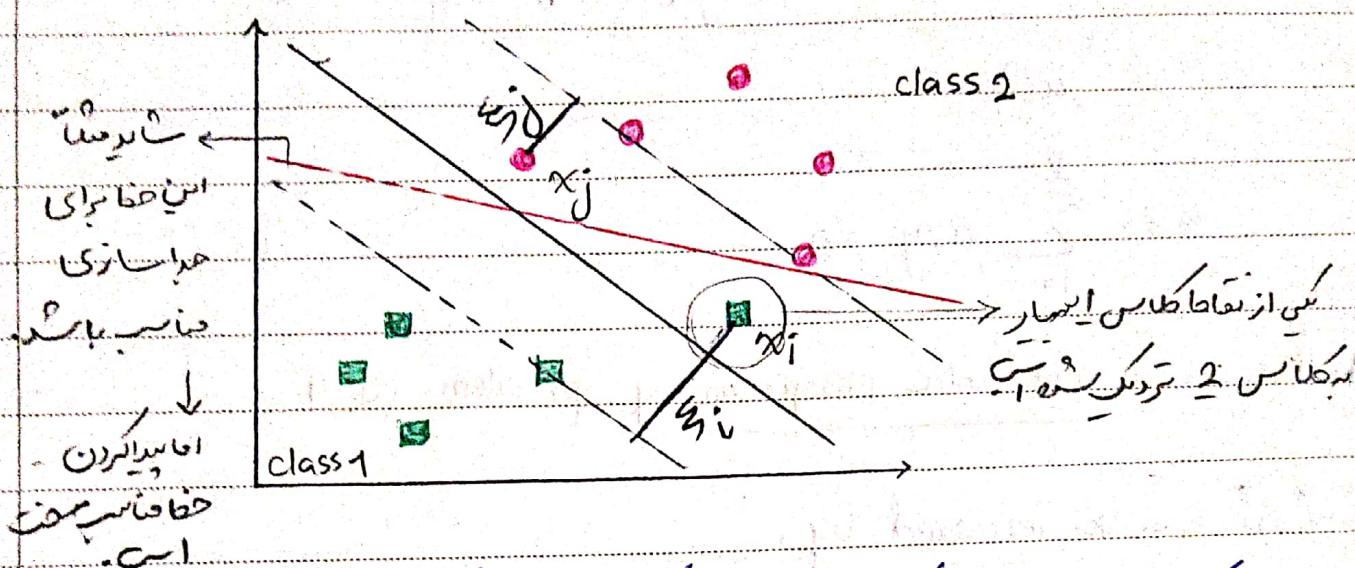


مarge نامه کل بیش از ۱ است و Train سه ماده ای را در چهارچوب می برد.

خوبی برآورده از صفر نباشد و معنی خوبی صفر نباشد.



حال حاضر در این طرز تغییر که مبنی توافقنامه خطا مشخصی برای حدیکردن دو طبقه مبنی شده است (حاشد و سکل نزدیک)



راه حل که در جود دارد: یا اینکه خطی مشخصی با بیندیش ملا برتر کریم شدم (بهمن یا جذب دهن) .
اصفهان می داشتم که خطی کسی با بیندیش نداشته باشد، حساباتی خطای خطا را با این بیندیش
و مطابق نشاند.

پس اگر مخواهیم خطا کیم، باید مقادیری خط را اعمال کرد مبنی معکاری خط را در داده کو
جی بذکریم.

به این ترتیب جی توافقنامه آن محدودیت SVM، ایازنی کشی کشی.

Soft margin Hyperplane

- Define $\xi_i = 0$ if there is no error for x_i .

+ ξ_i are just "slack variables" in optimization theory.

$$\left\{ \begin{array}{l} w^T x_i + b \geq 1 - \xi_i \quad y_i = 1 \\ w^T x_i + b \leq -1 + \xi_i \quad y_i = -1 \\ \xi_i \geq 0 \end{array} \right. \quad \forall i$$



- we want to minimize $\frac{1}{2} \|W\|^2 + C \sum_{i=1}^n E_i$

+ C : trade off parameter between error and margin.

→ the optimization problem becomes:

$$\text{minimize } \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i$$

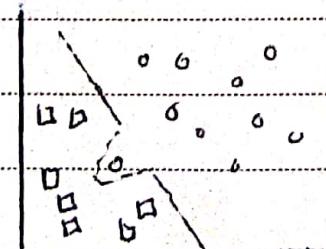
$$\text{s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

آنکه راهنمایی مینماییم که $\sum_{i=1}^n \|w_i\|^2$ را minimize کنیم
 برای این تفاوت را در داده‌ها بازگردانی کنیم

Linear SVM (non-separable case (cont'd))

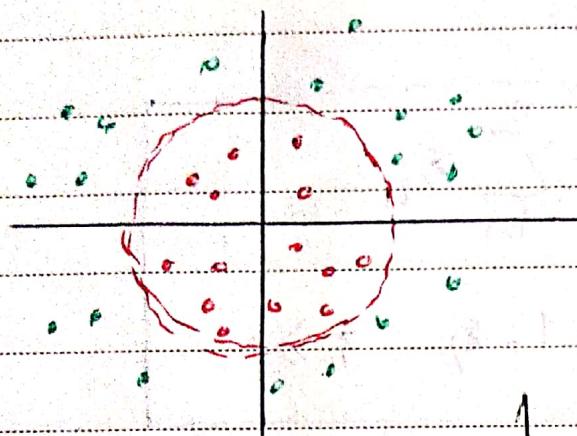
* The constant \underline{C} controls the trade-off between the margin and misclassification errors.

* Aims to prevent outliers from affecting the optimal hyperplane.



Extension to Non-linear Decision Boundary

دان فرض کنید \mathcal{X} مجموعه داده است که \mathcal{X} را $\mathcal{X}_1 \cup \mathcal{X}_2$ تقسیم می‌کند و \mathcal{X}_1 و \mathcal{X}_2 مجموعه‌هایی هستند که $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$ باشند. اگر \mathcal{X}_1 و \mathcal{X}_2 مجموعه‌هایی هستند که $\mathcal{X}_1 \cap \mathcal{X}_2 = \emptyset$ باشند، آن‌ها را مجموعه‌های **linearly separable** می‌نامند.



سریع سک سولوشن / سفاره از شخصیگی -
سریع سک سولوشن / سفاره از شخصیگی -

اجامی Solution سری بزرگیان حی را دری

یکی از solution space کی موجود این است که نقاط مخصوصی خواهد بود (input space) را با -
 نقاط مخصوصی خود برابر نمی کرد . اگر داشتم که نظریت که همچوی تو ایند نقاط را از مخصوصی خواهد بود
 معتقد باشم این تبار سریند، همچوی تو ایند به مخصوصی با این تبار بالاتر خواهد بود اما این توجه بخوبی نیاز
 نظریت که حس تو ایند خطی یا غیرخطی باشد - (linear mapping or non-linear

نمایش کسی خطا، نمایش کسی affine pattern را می‌نماید و سنتک عکس از همین عکس را می‌نماید و همین را خطای خود می‌نامد. اگر نمایش این دو نمایش را می‌نماید و به قصهی حدیدی affine نمایش کند، می‌توانیم این نمایش را با displacement نمایش کنیم. این displacement را $\text{displacement field}$ می‌نامیم. این $\text{displacement field}$ را می‌توانیم با gradient آن را gradient field نمایش کنیم. این gradient field را gradient map می‌نامیم. این gradient map را می‌توانیم با magnitude آن را magnitude map نمایش کنیم. این magnitude map را magnitude field می‌نامیم.

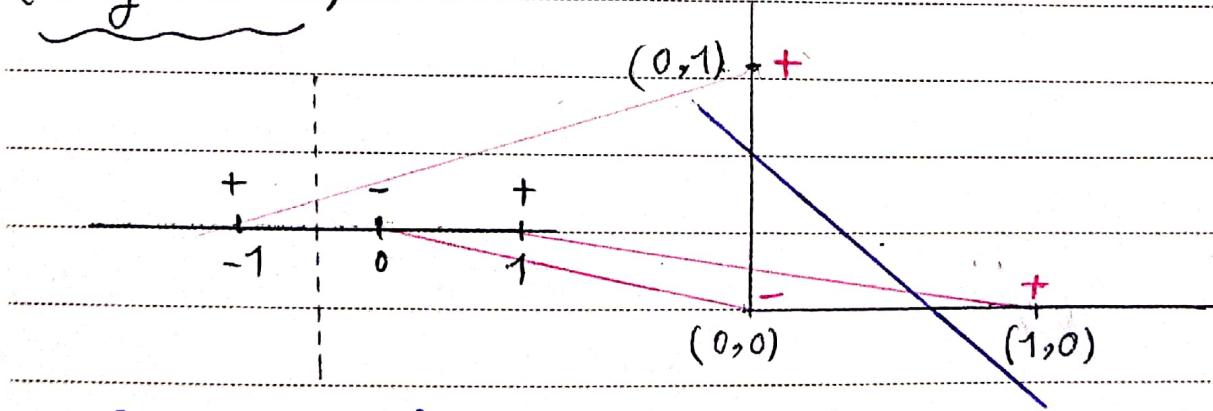


پس نتیجی مکری این اسما را train data, یا پایه نظری از عصایی حذف -
 برای عصایی بالا خود بیان transform کنیم، به این ایندک اعضای مکار را در عصایی
 بالا برای بانتر $\text{linearly separable}$ باشد.
 عصایی جدیدی که بجز از نظریه های آن دستگاه X را عصایی جدید
 به دست آورید.

- input space $\{x_i\}$ \rightarrow $\{f(x_i)\}$ \leftarrow transformation $\in \mathcal{H}$
 - linearly $\{f(x_i)\}$ \rightarrow $\{y_i\}$ \leftarrow $y_i = f(x_i) \cdot w + b$
 - only if $\{x_i\}$ is linearly separable

\vdash non-linear separable \Rightarrow no linear

- project the data to high dimensional space where it is linearly separable and then we can use linear SVM.
(using kernels)



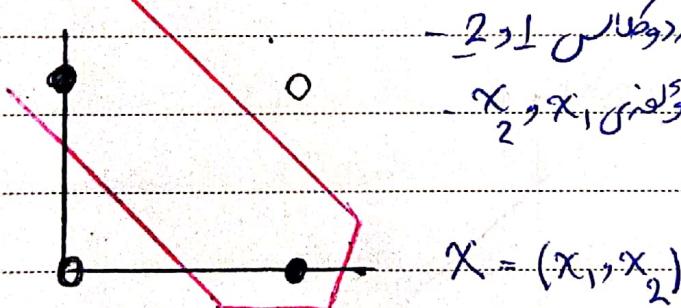
* ممکن باشد این دسته را misclassification نویسند.

حالاً فرض كيـنـد نـظـارـسـنـ اـحـامـ مـنـ دـهـمـ دـاـسـ 3ـنـصـطـ رـاـزـصـتـىـ لـكـ بـهـبـىـ سـرـونـجـدـىـ مـنـ بـرـعـمـ.

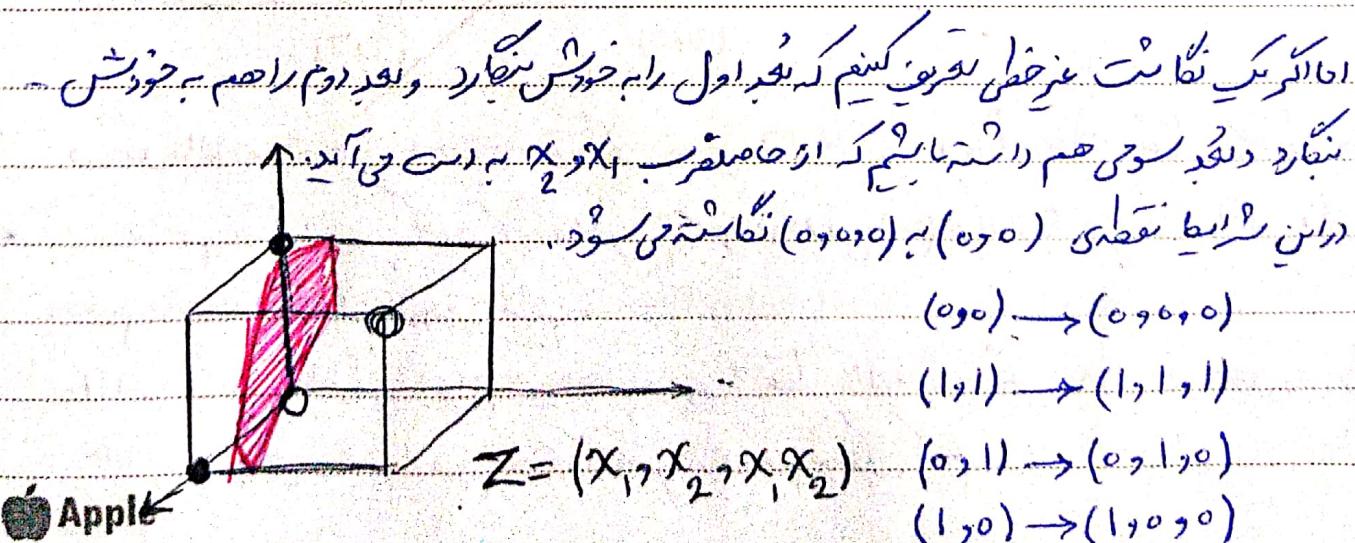
درواقع Kernel، تابس از نکات غریضه ایست که خوب داشتی و داری اینجا هستند و جون و فرم خوب dual lagrangian solution، خوب را خوب داری ایس بود که در پردازش classification برای خوب کرد.

The XOR problem

درین مثال عو نفعی دو عدد را در ترتیب پرید این مثال بی که ای باور Exclusive OR است سبب ایست که آن دو تا ورودی Boolean False ← True آنرا که ایست که آن دو تا ورودی False ← False آنرا که ایست که آن دو تا ورودی True ← F و کسی آنرا که ایست که آن دو تا ورودی Exclusive OR binary که بینم که می سندی ایست که ایست که آن نقطی (۰,۰) را داشم، عدد صفر بجهی کرده اند آنرا (۱,۱) داشم باز هم صفر بجهی کرده اند، و هنوز این دو نقطه در یک مکان حمله بجهی کرده اند آنرا (۰,۱) یا (۱,۰) در ترتیب خوبی ۱ ایست که هنوز این دو نقطه هم در یک مکان هستند.



درین مکله همچنان خوب عنوان کنید که دو مکان ۱ و ۰ را از هم جدا کنند و هر نقطه درین مکانها با دو مختلفین x_1 و x_2 مشخص نماید.



جستجوی محدود برای کلاسیکر دو کلاسی داده ها را در حوزه ای داریم.

- اگر یک object را در فضای اصلی \mathbb{R}^n نمایم، آنچنانچه این object را با دو دسته SVM (support vectors) یا یک دسته lagrangian باید train کرد تا object را در حوزه ای دسته ای کلاسی کرد.

+ possible problem of the transformation:

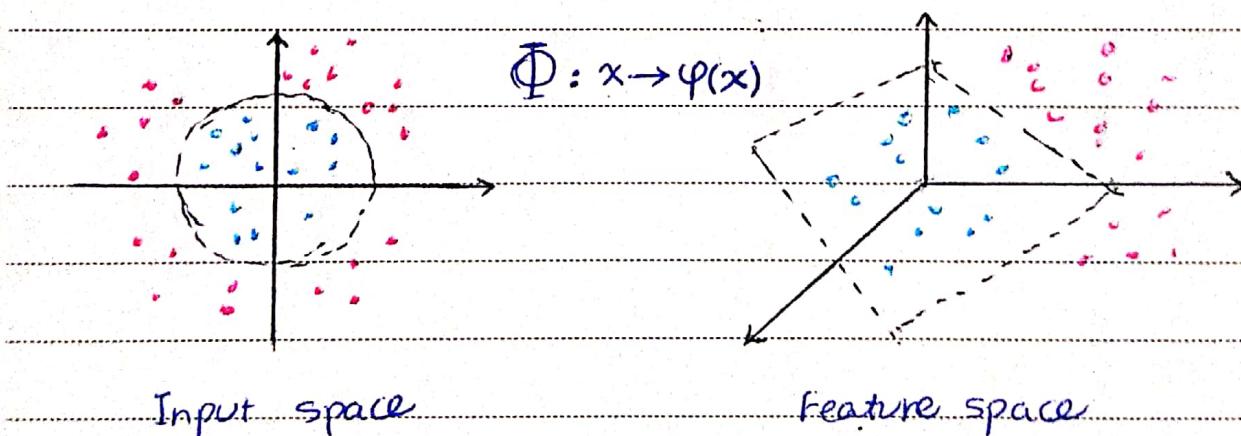
↓ + High computation burden and hard to get a good estimate.

+ SVM solves these two issues simultaneously.

↳ kernel tricks for efficient computation.

minimize $\|w\|^2$ can lead to a "good" classifier.

1/2



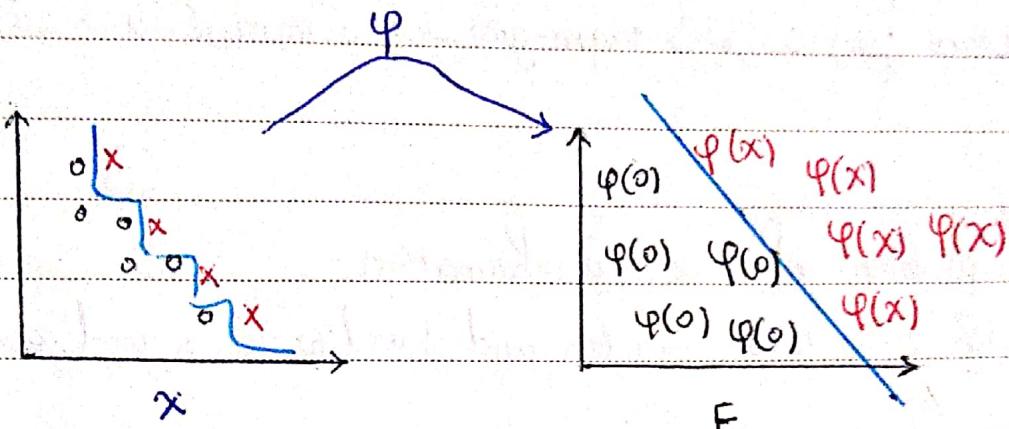
Φ (non-linear mapping) که در SVM non-linear است

$$x \rightarrow \Phi(x_k) = \begin{bmatrix} \varphi_1(x_k) \\ \varphi_2(x_k) \\ \vdots \\ \varphi_n(x_k) \end{bmatrix}$$



* Mapping the data to a sufficiently high dimensional space is likely to cast the data **Linearly separable** in that space.

Ex:



+ linear SVM $\rightarrow g(x) = \sum_{k=1}^n y_k \mu_k (x \cdot x_k) + w_0$

\Downarrow n is the number of training datapoint.

non-linear SVM $\rightarrow g(x) = \sum_{k=1}^n y_k \mu_k (\Phi(x) \cdot \Phi(x_k)) + w_0$

classification rule: $w_1 \text{ if } g(x) > 0 \text{ and } w_2 \text{ if } g(x) < 0$

$\Phi(x) \rightarrow$ is a label for x , $\Phi(x_k) \rightarrow$ training vector of x

$x_k \rightarrow \Phi(x_k) \Rightarrow$ this mapping might be very computationally intensive to compute

+ Is there an efficient way to compute?



Cuml. quadratic cubi $\binom{m+1}{2}$ جملة مربعات المقادير المترافق
 -> pure quadratic term, constant term & quadratic terms
 into quadratic

	1	Constant term	+ Number of terms (assuming m input dim)
$\Phi(x) =$	$\sqrt{2} x_1$	Linear terms	
	$\sqrt{2} x_2$		$= (m+2) - \text{choose } 2$ $= (m+2)(m+1)/2$
	\vdots		
	$\sqrt{2} x_m$		$1/2$
	x_1^2	Pure quadratic terms	
	\vdots		
	x_m^2		
	$\sqrt{2} x_1 x_2$	Quadratic cross terms	
	\vdots		
	$\sqrt{2} x_1 x_{m-1}$		

فرص انتقاء المقادير المترافق m المترافق
 في $\binom{m+1}{2}$ جملة مربعات المقادير المترافق
 (العدد المترافق في المقادير المترافق)
 انتقاء المقادير المترافق المترافق
 في $\binom{m+1}{2}$ جملة مربعات المقادير المترافق

$$\binom{m+2}{2} = \frac{(m+2)(m+1)}{2} = \frac{m^2 + 3m + 2}{2} \leftarrow \text{Cuml. } \frac{m^2}{2}$$

فرص انتقاء المقادير المترافق المترافق
 في $\binom{m+1}{2}$ جملة مربعات المقادير المترافق
 في $\binom{m+1}{2}$ جملة مربعات المقادير المترافق
 في $\binom{m+1}{2}$ جملة مربعات المقادير المترافق

$$\begin{array}{c}
 \Phi(a) * \Phi(b) = \\
 \left[\begin{array}{c} 1 \\ \sqrt{2}a_1 \\ \vdots \\ \sqrt{2}a_m \\ \hline \sqrt{2}a_1 a_2 \\ \vdots \\ \sqrt{2}a_1 a_m \end{array} \right] * \left[\begin{array}{c} 1 \\ \sqrt{2}b_1 \\ \vdots \\ \sqrt{2}b_m \\ \hline \sqrt{2}b_1 b_2 \\ \vdots \\ \sqrt{2}b_1 b_m \end{array} \right] \\
 + \sum_{i=1}^m 2a_i b_i \\
 + \sum_{i=1}^m a_i^2 b_i^2 \\
 + \sum_{i=1}^m \sum_{j=i+1}^m 2a_i a_j b_i b_j
 \end{array}$$

$t \cdot m$ is input dimension $\Rightarrow [a, b]$ are m dim vectors

$$\boxed{\Phi(a) * \Phi(b) = 1 + 2 \sum_{i=1}^m a_i b_i + \sum_{i=1}^m a_i^2 b_i^2 + \sum_{i=1}^m \sum_{j=i+1}^m 2a_i a_j b_i b_j}$$

لکھیں کہ جو دو دلگھ کر جائے تو $(a \cdot b + 1)^2$ کا function ہے جس سے بخوبی نہیں کہا جاتا ہے بلکہ اسے

$$(a \cdot b + 1)^2 = (a \cdot b)^2 + 2a \cdot b + 1 = \left(\sum_{i=1}^m a_i b_i \right)^2 + 2 \sum_{i=1}^m a_i b_i + 1$$

$$= \sum_{i=1}^m \sum_{j=1}^m a_i b_i a_j b_j + 2 \sum_{i=1}^m a_i b_i + 1 =$$

$$= \boxed{\sum_{i=1}^m (a_i b_i)^2 + 2 \sum_{i=1}^m \sum_{j=i+1}^m a_i b_i a_j b_j + 2 \sum_{i=1}^m a_i b_i + 1}$$

لکھیں کہ $(a \cdot b + 1)^2$, $\Phi(a) * \Phi(b)$ میں خواہ ہوں گے

$O(m)$ $O(m^2)$



رس نئی kernel trick مخفی کرد اگرچو اهمیت نطاں را ایام دھم سخت -
رس جوں حصہ برائی سادہ ترین نطاں عرضی (quadratic) اگرچوی مقصود متعجبی -
باید باید $O(m^2)$ حسابات انجام دھم اکثر ضرب داخلی و پرتو را از تعقیل دانتے ہیں لیکن بعد میں حسابات کو کم کر جائے گا۔

$\boxed{O(m)}$ حالا خرض کیسے تو ان $(ab+1)^2$ تحریک کرنے $\leftarrow (a+b+1)^n$ \leftarrow حسابات سخت می گوں۔
اکھات سطح دارہ سے تو باتھر تو ان n تقریبی کرنے و حسابات سخت می گوں۔

$$\downarrow \boxed{(m^n)}$$

اسی طبقے کا kernel یعنی $(a.b+1)^2$ بعقار زندی کھٹھٹھ می گا۔

polynomial	$\Phi(x)$	Q_{KL}	$\Phi(a).\Phi(b)$	Q_{KL}	1/2
quadratic	$m^2/2$	$m^2 n^2/4$	$(a.b+1)^2$	$mn^2/2$	
cubic	$m^3/6$	$m^3 n^2/12$	$(a.b+1)^3$	$mn^2/2$	
Quartic	$m^4/24$	$m^4 n^2/48$	$(a.b+1)^4$	$mn^2/2$	

n : number of training data points

m : input dimension

جو تو ان نئان دار کو حق Kernel کی نویں اسی درخواہم آئیں (باتھر سب جھٹکی صبر بیرون) بخدا، حسابات سب ایسا جو روک

$$K(x,y) = (x \cdot y)^d$$

$$h = \binom{m+d-1}{d}$$

$$\rightarrow m=256, d=4 \rightarrow h=183181376 !!$$



+ A dot product in high dimensional space would require $O(h)$ computations while the kernel requires only $O(m)$ computations.

Ex: $x \in \mathbb{R}^2$, $\Phi(x) = \begin{bmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{bmatrix} \in \mathbb{R}^3$, $k(x, y) = (x \cdot y)^2$

$$(x \cdot y)^2 = (x_1 y_1 + x_2 y_2)^2$$

$$\Phi(x) \cdot \Phi(y) = x_1^2 y_1^2 + 2x_1 y_1 x_2 y_2 + x_2^2 y_2^2 = (x_1 y_1 + x_2 y_2)^2$$

1/2.

نتیجہ میں اس کے نتائج unique Φ ہیں، لیکن ہزاری کے kernel میں اسے جزئی تابع Φ دھردار نہ ہو سکتا۔ Kernel میں مخفی رابطہ موجود ہے۔

$$\Phi(x) = \frac{1}{2} \begin{bmatrix} (x_1^2 - x_2^2) \\ 2x_1x_2 \\ (x_1^2 + x_2^2) \end{bmatrix} \in \mathbb{R}^3 \quad \text{or} \quad \Phi(x) = \begin{bmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{bmatrix} \in \mathbb{R}^3$$

حَدِّيْمِيْتِيْلِيْسْ كِيْرِنْ كِيْرِنْ (Kernell) حَدِّيْمِيْتِيْلِيْسْ كِيْرِنْ كِيْرِنْ (Kernell) $\Phi(x)\Phi(y)$

The kernel trick

compute dot products using a kernel function:

$$K(x, x_K) = \Phi(x) \cdot \Phi(x_K)$$

کرنل (kernel) کے بارے میں خاصی بوداں لارڈ جنری نے بتا دیے ہیں۔



$$g(x) = \sum_{k=1}^n y_k \mu_k (\Phi(x) \cdot \Phi(x_k)) + w_0$$



$$g(x) = \sum_{k=1}^n y_k \mu_k K(x, x_k) + w_0$$

↓
kernel

جیسا کہ kernel اور رسمی کو اپنے کو $\Phi(\cdot)$ کا مکانیکی میں مل جائے تو $K(x, y)$ کو $\Phi(x) \cdot \Phi(y)$ کے طور پر لکھ سکتے ہیں۔ اسی وجہ سے $K(x, y)$ کو $\Phi(\cdot)$ کا کرنل کہا جاتا ہے۔

+ $K(x, y)$ needs to satisfy Mercer condition (in order for $\Phi(\cdot)$ to exist)

+ the mercer's condition does not tell us how to construct $\Phi(\cdot)$ or even what the high dim space is.

+ advantages of kernel trick:

1) no need to know $\Phi(\cdot)$

2) Computations remain feasible even if the feature space has high dim.

Mercer's condition

اگر $K(x, y)$ کو کرنل کہا جائے تو

$$\iint k(x, y) g(x) g(y) dx dy > 0$$

کرنل کو K کہا جائے۔



Ex¹: $K(x, y) = 1$

Fubini's theorem

$$\iint g(x)g(y)dxdy = \int g(x)dx \int g(y)dy = \left(\left(\int g(x)dx \right)^2 \right) > 0$$

↓
non-negative

Ex²: $K(x, y) = (1 + x_1 y_1 + x_2 y_2)^2$

$$\rightarrow \Phi \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = (1, \sqrt{2}x_1, \sqrt{2}x_2, x_1^2, x_2^2, \sqrt{2}x_1 x_2)$$

$$\rightarrow \Phi \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = (1, \sqrt{2}y_1, \sqrt{2}y_2, y_1^2, y_2^2, \sqrt{2}y_1 y_2)$$

$$\rightarrow (\Phi \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}, \Phi \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}) = (1 + x_1 y_1 + x_2 y_2)^2 > 0$$

$$= K(x, y)$$

→ inner product can be computed by K (without knowing Φ)

: over polynomial inner product between features \in Kernel

$$K(x, y) = (x^T y + 1)^d$$

Radial basis function kernel with width σ . : Gaussian kernel

$$K(x, y) = \exp(-\|x - y\|^2 / (2\sigma^2))$$

↓ closely related to radial basis function neural networks.



Year: Month: Day: ()

Subject:

+ sigmoid with parameters k and θ

hyperbolic

$$K(x, y) = \tanh(kx^T y + \theta)$$

* It does not satisfy the mercer condition on all k and θ .

- sigmoid doesn't satisfy the mercer probability \Rightarrow E kernel job
cannot solve linearly separable