Brief Introduction to Machine Learning

Yuh-Jye Lee

Lab of Data Science and Machine Intelligence
Dept. of Applied Math. at NCTU

August 29, 2016

What is Machine Learning?

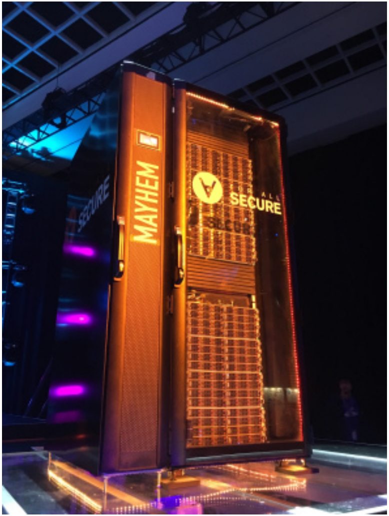
Representation + Optimization + Evaluation

Pedro Domingos, A few useful things to know about machine learning, Communications of the ACM, Vol. 55 Issue 10, 78-87, October 2012

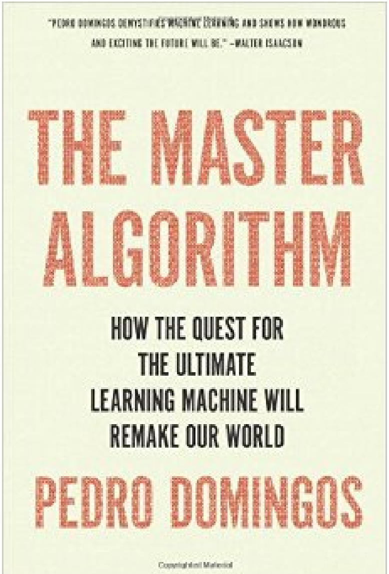
AlphaGo


nature
THE INTERNATIONAL WEEKLY JOURNAL OF SCIENCE
At last — a computer program that can beat a champion Go player PAGE 484
ALL SYSTEMS GO
CONSERVATION
SONGBIRDS À LA CARTE
Illegal harvest of millions of Mediterranean birds
PAGE 452
RESEARCH ETHICS
SAFEGUARD TRANSPARENCY
Don't let openness backfire on individuals
PAGE 459
POPULAR SCIENCE
WHEN GENES GOT 'SELFISH'
Dawkins's calling card forty years on
PAGE 462
NATURE.COM/NATURE
28 January 2016 £10
Vol. 529, No. 7587

Mayhem Wins DARPA Cyber Grand Challenge



The Master Algorithm



The Master Algorithm

The Plan of My Lecture

- Only forcus on *Supervised Learning*
- Will give you four basic algorithms
  - Online Perceptron Algorithm
  - Support Vector Machines
  - k-Nearest Neighbor
  - Naive Bayes Classifier
- Basic Concept of Learning Theorey
- Evaluation

Binary Classification Problem (A Fundamental Problem in Data Mining)

- Find a decision function (classifier) to discriminate two categories data sets.
- Supervised learning in Machine Learning
    - Decision Tree, *Deep* Neural Network, k-NN and Support Vector Machines, etc.
- Discrimination Analysis in Statistics
    - Fisher Linear Discriminator
- Successful applications:
    - Marketing, Bioinformatics, Fraud detection

Binary Classification Problem

Given a training dataset

$$S = \{(x^i, y_i) | x^i \in \mathbb{R}^n, y_i \in \{-1, 1\}, i = 1, \ldots, \ell\}$$

$$x^i \in A_+ \Leftrightarrow y_i = 1 \ \& \ x^i \in A_- \Leftrightarrow y_i = -1$$

Main Goal:

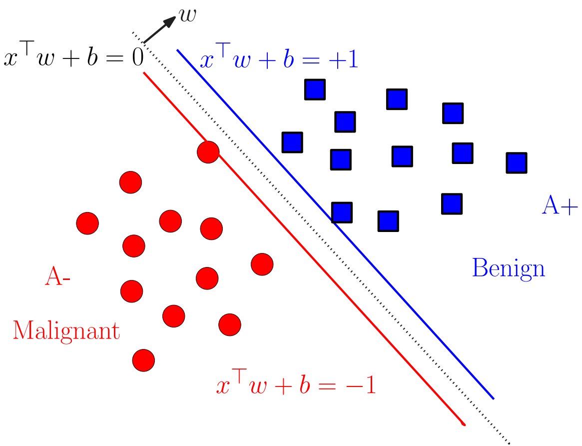
Predict the unseen class label for new data

Find a function $f : \mathbb{R}^n \to \mathbb{R}$ by learning from data

$$f(x) \geq 0 \Rightarrow x \in A_+ \text{ and } f(x) < 0 \Rightarrow x \in A_-$$

The simplest function is linear: $f(x) = w^\top x + b$

Binary Classification Problem
Linearly Separable Case



People of ACM: David Blei, (Sept. 9, 2014)

The recipient of the 2013 ACM- Infosys Foundation Award in the Computing Sciences, he is joining Columbia University this fall as a Professor of Statistics and Computer Science, and will become a member of Columbia's Institute for Data Sciences and Engineering.

What is the most important recent innovation in machine learning?

[A]: One of the main recent innovations in ML research has been that we (the ML community) can now scale up our algorithms to massive data, and I think that this has fueled the modern renaissance of ML ideas in industry. The main idea is called *stochastic optimization*, which is an adaptation of an *old algorithm invented by statisticians in the 1950s*.

What is the most important recent innovation in machine learning?

[A]: *In short, many machine learning problems can be boiled down to trying to find parameters that maximize (or minimize) a function*. A common way to do this is "gradient ascent," iteratively following the steepest direction to climb a function to its top. This technique requires repeatedly calculating the steepest direction, and the problem is that this calculation can be expensive. *Stochastic optimization* lets us use *cheaper approximate calculations*. It has transformed modern machine learning.

Linear Learning Machines

- The simplest function is linear: $f(x) = w^\top x + b$
- Finding this simplest function via an on-line and mistake-driven procedure
- Update the weight vector and bias when there is a misclassified point

Perceptron Algorithm (Primal Form) Rosenblatt, 1956

- Given a training dataset S, and initial weight vector $w^0 = \mathbf{0}$ and the bias $b_0 = 0$

  Repeat:

  for $i = 1$ *to* ℓ

      if $y_i(\langle w^k \cdot x^i \rangle + b_k) \leq 0$ *then*

          $w^{k+1} \leftarrow w^k + \eta y_i x^i$

          $b_{k+1} \leftarrow b_k + \eta y_i R^2$

          $k \leftarrow k + 1$

      end if

  Until no mistakes made within the for loop

  Return: $k, (w^k, b_k)$.

$$R = \max_{1 \leq i \leq \ell} \|x^i\|$$

- What is k ?

$y_i(\langle w^{k+1} \cdot x^i \rangle + b_{k+1}) > y_i(\langle w^k \cdot x^i \rangle) + b_k$? $w^{k+1} \longleftarrow w^k + \eta y_i x^i$ and $b_{k+1} \longleftarrow b_k + \eta y_i R^2$

$$
\begin{aligned}
y_i(\langle w^{k+1} \cdot x^i \rangle + b_{k+1}) &= y_i(\langle (w^k + \eta y_i x^i) \cdot x^i \rangle + b_k + \eta y_i R^2) \\
&= y_i(\langle w^k \cdot x^i \rangle + b_k) + y_i(\eta y_i(\langle x^i \cdot x^i \rangle + R^2)) \\
&= y_i(\langle w^k \cdot x^i \rangle + b_k) + \eta(\langle x^i \cdot x^i \rangle + R^2)
\end{aligned}
$$

$$
\boxed{R = \max_{1 \le i \le \ell} \|x^i\|}
$$

$y_i(\langle w^{k+1} \cdot x^i \rangle + b_{k+1}) > y_i(\langle w^k \cdot x^i \rangle) + b_k$? $w^{k+1} \longleftarrow w^k + \eta y_i x^i$ and $b_{k+1} \longleftarrow b_k + \eta y_i R^2$

$$\begin{aligned} y_i(\langle w^{k+1} \cdot x^i \rangle + b_{k+1}) &= y_i(\langle (w^k + \eta y_i x^i) \cdot x^i \rangle + b_k + \eta y_i R^2) \\ &= y_i(\langle w^k \cdot x^i \rangle + b_k) + y_i(\eta y_i(\langle x^i \cdot x^i \rangle + R^2)) \\ &= y_i(\langle w^k \cdot x^i \rangle + b_k) + \eta(\langle x^i \cdot x^i \rangle + R^2) \end{aligned}$$

$$\boxed{R = \max_{1 \leq i \leq \ell} \|x^i\|}$$

Perceptron Algorithm Stop in Finite Steps

Theorem(Novikoff)
Let S be a non-trivial training set, and let

$$R = \max_{1 \le i \le \ell} \|x^i\|$$

Suppose that there exists a vector w_{opt} *such that* $\|w_{opt}\| = 1$ and

$$y_i(\langle w_{opt} \cdot x^i \rangle + b_{opt}) \ge \gamma \text{ for } 1 \le i \le \ell.$$

Then the number of mistakes made by the on-line perceptron algorithm on S is almost $(\frac{2R}{\gamma})^2$.

Perceptron Algorithm (Dual Form) $w = \sum_{i=1}^{\ell} \alpha_i y_i x^i$

Given a linearly separable training set S and $\alpha = 0$, $\alpha \in \mathbb{R}^{\ell}$, $b = 0$, $R = \max_{1 \leq i \leq \ell} \|x_i\|$.

Repeat: *for* $i = 1$ *to* ℓ

if $y_i(\sum_{j=1}^{\ell} \alpha_j y_j \langle x^j \cdot x^i \rangle + b) \leq 0$ *then*

$\alpha_i \leftarrow \alpha_i + 1$; $b \leftarrow b + y_i R^2$

end if

end for

Until no mistakes made within the for loop return: (α, b)

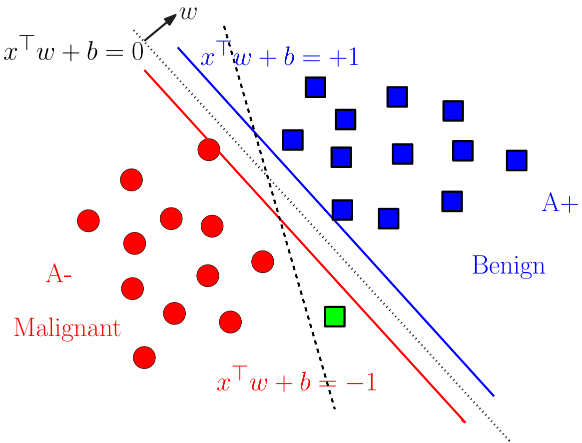
What We Got in the Dual Form of Perceptron Algorithm?

- The number of updates equals: $\sum_{i=1}^{\ell} \alpha_i = \|\alpha\|_1 \leq (\frac{2R}{\gamma})^2$
- $\alpha_i > 0$ implies that the training point (x_i, y_i) has been misclassified in the training process at least once.
- $\alpha_i = 0$ implies that removing the training point (x_i, y_i) will not affect the final results.
- The training data only appear in the algorithm through the entries of the Gram matrix, $G \in \mathbb{R}^{\ell \times \ell}$ which is defined below:

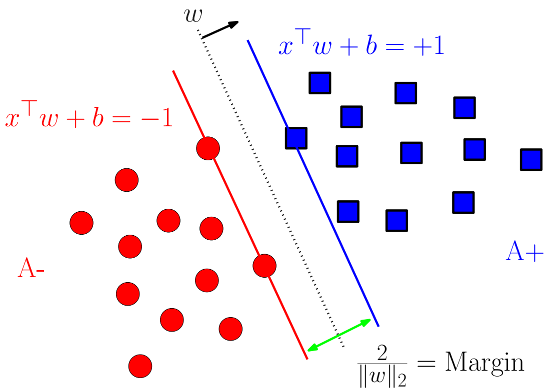
$$G_{ij} = \langle x_i, x_j \rangle$$

Outline

Binary Classification Problem
Linearly Separable Case



Support Vector Machines
Maximizing the Margin between Bounding Planes



Why Use Support Vector Machines? Powerful tools for Data Mining

- SVM classifier is an optimally defined surface
- SVMs have a good geometric interpretation
- SVMs can be generated very efficiently
- Can be extended from linear to nonlinear case
  - Typically nonlinear in the input space
  - Linear in a higher dimensional "feature space"
  - Implicitly defined by a kernel function
- Have a sound theoretical foundation
  - Based on Statistical Learning Theory

Why We Maximize the Margin? (Based on Statistical Learning Theory)

- The Structural Risk Minimization (SRM):
  - The expected risk will be less than or equal to empirical risk (training error)+ VC (error) bound
- $\|w\|_2 \propto$ *VC bound*
- min *VC bound* $\Leftrightarrow$ min $\frac{1}{2}\|w\|_2^2 \Leftrightarrow$ max *Margin*

Summary the Notations

Let $S = \{(x^1, y_1), (x^2, y_2), \ldots, (x^\ell, y_\ell)\}$ be a training dataset and represented by matrices

$$A = \begin{bmatrix} (x^1)^\top \\ (x^2)^\top \\ \vdots \\ (x^\ell)^\top \end{bmatrix} \in \mathbb{R}^{\ell \times n}, D = \begin{bmatrix} y_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & y_\ell \end{bmatrix} \in \mathbb{R}^{\ell \times \ell}$$

$A_i w + b \geq +1, \ \ for\ D_{ii} = +1$
$A_i w + b \leq -1, \ \ for\ D_{ii} = -1$, equivalent to $D(Aw + \mathbf{1}b) \geq \mathbf{1}$,
where $\mathbf{1} = [1, 1, \ldots, 1]^\top \in \mathbb{R}^\ell$

Support Vector Classification (Linearly Separable Case, Primal)

The hyperplane (w, b) is determined by solving the minimization problem:

$$\min_{(w,b)\in\mathbb{R}^{n+1}} \frac{1}{2}\|w\|_2^2$$

$$D(Aw + \mathbf{1}b) \geq \mathbf{1},$$

It realizes the maximal margin hyperplane with geometric margin

$$\gamma = \frac{1}{\|w\|_2}$$

Support Vector Classification (Linearly Separable Case, Dual Form)

The dual problem of previous MP:

$$\max_{\alpha \in R^{\ell}} \quad \mathbf{1}^{\top}\alpha - \frac{1}{2}\alpha^{\top} D A A^{\top} D \alpha$$

subject to

$$\mathbf{1}^{\top} D \alpha = 0, \alpha \geq \mathbf{0}$$

Applying the KKT optimality conditions, we have $w = A^{\top} D \alpha$. But where is b ?

Don't forget

$$\mathbf{0} \leq \alpha \perp D(Aw + \mathbf{1}b) - \mathbf{1} \geq \mathbf{0}$$

Dual Representation of SVM

(Key of Kernel Methods: $w = A^\top D\alpha^* = \sum_{i=1}^{\ell} y_i \alpha_i^* A_i^\top$)

The hypothesis is determined by (α^*, b^*)

$$\begin{aligned} h(x) &= sgn(\langle x \cdot A^\top D\alpha^* \rangle + b^*) \\ &= sgn(\sum_{i=1}^{\ell} y_i \alpha_i^* \langle x^i \cdot x \rangle + b^*) \\ &= sgn(\sum_{\alpha_i^* > 0} y_i \alpha_i^* \langle x^i \cdot x \rangle + b^*) \end{aligned}$$

Remember : $A_i^\top = x_i$

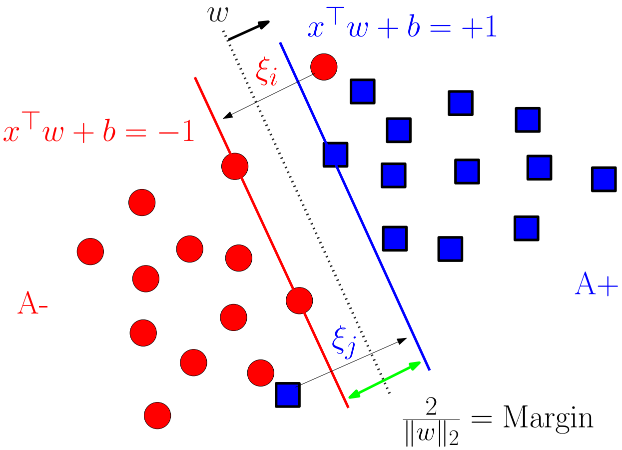
Soft Margin SVM (Nonseparable Case)

- If data are not linearly separable
  - Primal problem is infeasible
  - Dual problem is unbounded above
- Introduce the slack variable for each training point

$$y_i(w^\top x^i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad \forall i$$

- The inequality system is always feasible e.g.

$$w = \mathbf{0}, \quad b = 0, \quad \xi = \mathbf{1}$$



w
$x^\top w + b = +1$
ξ_i
$x^\top w + b = -1$
A-
A+
ξ_j
$\frac{2}{\|w\|_2} = \text{Margin}$

Robust Linear Programming
Preliminary Approach to SVM

$$\begin{aligned} \min_{w,b,\xi} \quad & \mathbf{1}^\top \xi \\ \text{s.t.} \quad & D(Aw + \mathbf{1}b) + \xi \geq \mathbf{1} \quad (LP) \\ & \xi \geq \mathbf{0} \end{aligned}$$

where ξ is nonnegative slack(*error*) vector

- The term $\mathbf{1}^\top \xi$, 1-norm measure of *error* vector, is called the *training error*
- For the linearly separable case, at solution of(LP): $\xi = \mathbf{0}$

Support Vector Machine Formulations (Two Different Measures of Training Error)

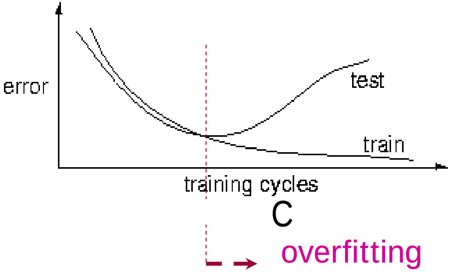
2-Norm Soft Margin:

$$\min_{(w,b,\xi)\in\mathbb{R}^{n+1+\ell}} \quad \frac{1}{2}\|w\|_2^2 + \frac{C}{2}\|\xi\|_2^2$$
$$D(Aw + \mathbf{1}b) + \xi \geq \mathbf{1}$$

1-Norm Soft Margin (Conventional SVM)

$$\min_{(w,b,\xi)\in\mathbb{R}^{n+1+\ell}} \quad \frac{1}{2}\|w\|_2^2 + C\mathbf{1}^\top\xi$$
$$D(Aw + \mathbf{1}b) + \xi \geq \mathbf{1}$$
$$\xi \geq \mathbf{0}$$

Tuning Procedure
How to determine C ?



The final value of parameter is one with the maximum testing set correctness!

1-Norm SVM (Different Measure of Margin)

1-Norm SVM:

$$\min_{(w,b,\xi)\in\mathbb{R}^{n+1+\ell}} \quad \| w \|_1 + C\mathbf{1}^\top \xi$$
$$D(Aw + \mathbf{1}b) + \xi \geq \mathbf{1}$$
$$\xi \geq \mathbf{0}$$

Equivalent to:

$$\min_{(s,w,b,\xi)\in\mathbb{R}^{2n+1+\ell}} \quad \mathbf{1}s + C\mathbf{1}^\top \xi$$
$$D(Aw + \mathbf{1}b) + \xi \geq \mathbf{1}$$
$$-s \leq w \leq s$$
$$\xi \geq \mathbf{0}$$

Good for feature selection and similar to the LASSO

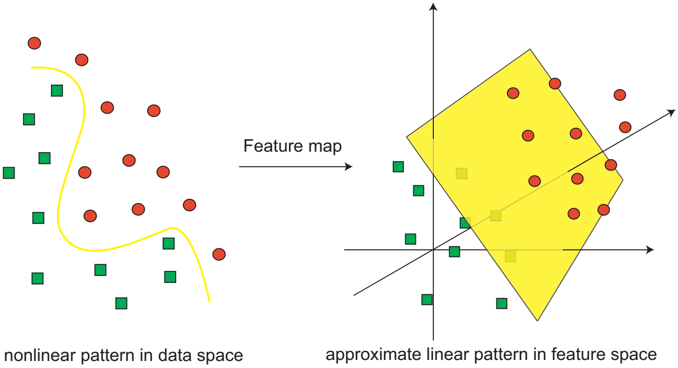
Outline

Two-spiral Dataset (94 white Dots & 94 Red Dots)

Learning in Feature Space (Could Simplify the Classification Task)

- Learning in a high dimensional space could degrade generalization performance
  - This phenomenon is called *curse of dimensionality*
- By using a *kernel function*, that represents the inner product of training example in feature space, we never need to explicitly know the nonlinear map
  - Even do not know the dimensionality of feature space
- There is no free lunch
  - Deal with a huge and dense kernel matrix
    - Reduced kernel can avoid this difficulty

$$X \overset{\Phi}{- - - \longrightarrow} F$$



Linear Machine in Feature Space

Let $\phi : X \longrightarrow F$ be a nonlinear map from the input space to some feature space
The classifier will be in the form(*primal*):

$$f(x) = (\sum_{j=1}^{?} w_j \phi_j(x)) + b$$

Make it in the *dual* form:

$$f(x) = (\sum_{i=1}^{\ell} \alpha_i y_i \langle \phi(x^i) \cdot \phi(x) \rangle) + b$$

Kernel:Represent Inner Product in Feature Space

Definition: A kernel is a function $K : X \times X \longrightarrow \mathbb{R}$ such that *for all* $x, z \in X$

$$K(x,z) \quad = \quad \langle \phi(x) \cdot \phi(z) \rangle$$

where $\phi : X \longrightarrow F$
The classifier will become:

$$f(x) \quad = \quad (\sum_{i=1}^{\ell} \alpha_i y_i K(x^i, x)) + b$$

Outline

Instance-based Learning

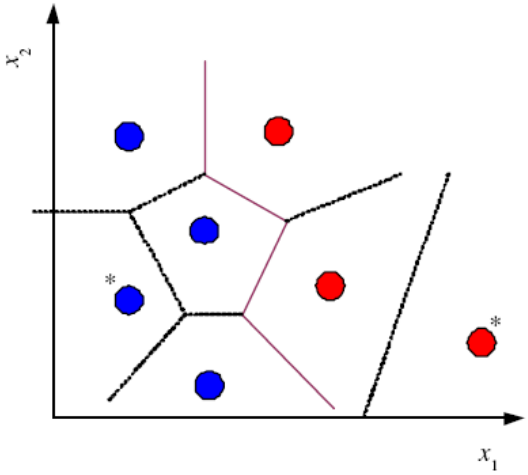
- Fundamental philosophy: Two instances that are *close to each other* or *similar to each other* they should share with the same label
- Also known as *memory-based learning* since what they do is store the training instances in a lookup table and *interpolate* from these.
- It requires memory of $\mathcal{O}(N)$
- Given an input similar ones should be found and finding them requires computation of $\mathcal{O}(N)$
- Such methods are also called *lazy learning* algorithms. Because they do NOT compute a model when they are given a training set but postpone the computation of the model until they are given a new test instance (query point)

k-Nearest Neighbors Classifier

- Given a query point x^o, we find the k training points $x^(i), \; i = 1, 2, \ldots, k$ *closest* in *distance* to x^o
- Then classify using *majority vote* among these k neighbors.
- Choose k as an odd number will avoid the tie. Ties are broken at random
- If all attributes (features) are real-valued, we can use Euclidean distance. That is $d(x, x^o) = \|x - x^o\|_2$
- If the attribute values are *discrete*, we can use *Hamming distance*, which counts the number of *nonmatching* attributes

$$d(x, x^o) = \sum_{j=1}^{n} \mathbf{1}(x_j \neq x_j^o)$$

1-Nearest Neighbor Decision Boundary (Voronoi)



Distance Measure

- Using different distance measurements will give very different results in k-NN algorithm.
- Be careful when you compute the distance
- We might need to *normalize* the scale between different attributes. For example, yearly income vs. daily spend
- Typically we first standardize each of the attributes to have mean zero and variance 1

$$\hat{x}_j = \frac{x_j - \mu_j}{\sigma_j}$$

Learning Distance Measure

- Finding a distance function $d(x^i, x^j)$ such that if x^i and x^j are belong to the *class* the distance is *small* and if they are belong to the *different classes* the distance is large.
- Euclidean distance: $\|x^i - x^j\|_2^2 = (x^i - x^j)^\top (x^i - x^j)$
- Mahalanobis distance: $d(x^i, x^j) = (x^i - x^j)^\top M (x^i - x^j)$ where M is a positive semi-definited matrix.

$$(x^i - x^j)^\top M (x^i - x^j) = (x^i - x^j)^\top L^\top L (x^i - x^j)$$

$$= (Lx^i - Lx^j)^\top (Lx^i - Lx^j)$$

- The matrix L can be with the size $k \times n$ and $k << n$

Reference

- C. J. C Burges. "A Tutorial on Support Vector Machines for Pattern Recognition", Data Mining and Knowledge Discovery, Vol. 2, No. 2, (1998) 121-167.
- N. Cristianini and J. Shawe-Taylor. "An Introduction to Support Vector Machines", Cambridge University Press,(2000).