Support Vector Machine (SVM)

Tuesday, November 20, 2018 12:25

For the glory of God

Introduction

- When data is unlabelled, supervised learning is not possible but an unsupervised learning approach is required.

  e.g. Clustering is an unsupervised learning technique and classification is a supervised learning technique.

- A Support Vector Machine (SVM) is one of the best supervised learning algorithms for classification and so forth.

  ↳ It was extremely popular around the time it was developed in the 1990s.

- A SVM is sometimes more powerful than Neural Network (NN) for complex non-linear problems.

- The original SVM algorithm was invented in 1963.

- In 1992, the authors suggested a way to create non-linear classifiers by applying the kernel tricks. (Maximum margin)

- In 1995, the soft margin was proposed by one of Google employees.

What is a Support Vector Machine?

- In machine learning, a SVM is supervised learning model that analyzes data used for classification and regression.

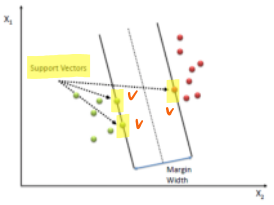
  ↳ It's mostly used for classification.

- A SVM performs classification by finding the hyperplane that maximizes the margin between the two classes.

  (Note that it actually can extend to multi-dimensional classes)

- The vectors that define the hyperplane are called as 'support vectors'. This is where the name came from.

  e.g.



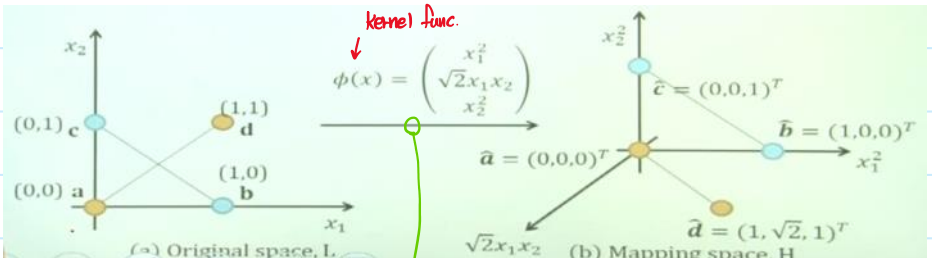
; We see three support vectors in this example.

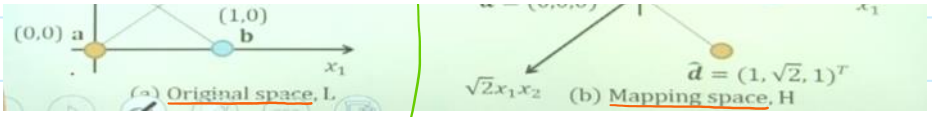
This contributes the SVM with more popularity.

- SVM uses a technique called as the kernel trick to transform given data and then based on these transformations.

  it finds an optimal boundary between the possible outputs.

  e.g. Suppose that there are non-linearly separable data sets;

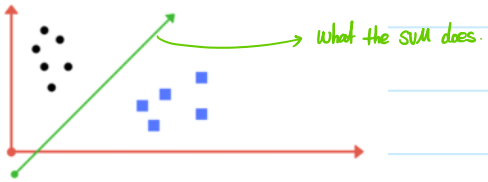




The non-linear separable case can be linearly separable when we increase the basis space.

- In short, separation of classes is what the SVM does.

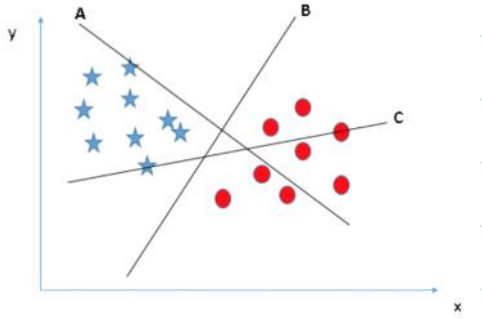
e.g.



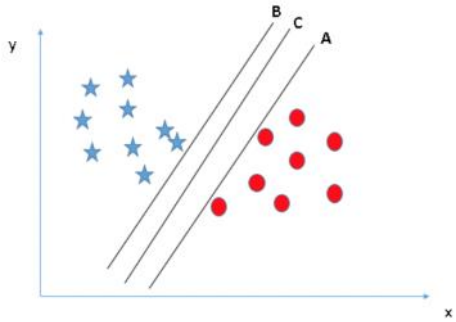
How does the SVM work ?

a) Intuitive explanation

- Basically, the objective of the SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points.
- Let's say that we have the dataset as below ;



- We would choose the B if we remember a thumb rule to identify the hyperplane that distinctly classifies the dataset.
- Then, how about this ?
  - It is obvious that three hyperplanes are exactly separating two classifiers without any mis-classification ;



note that ; (Neural Network may be stuck in local min.)

the beauty of SVM is that if the dataset is linearly separable, there is a unique global solution in the optimization problem using quadratic programming.

↳ It is needed to avoid local minimum issues.

- In this case, we need to think about the thumb rule ;

- A SVM performs classification by finding the hyperplane that maximizes the margin between the two classes.

∴ C would be selected !

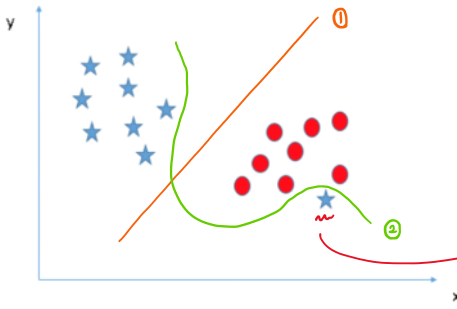
What is the significance of maximization on the margin ?

1) If we select a hyperplane having low margin, i.e. A or B,
there is high chance of mis-classification.

2) Maximizing the margin distance provides some reinforcement
so that future data points can be classified with more confidence.

· So far, we have discussed about the linear decision boundary case only.

: What if we add one data point such that we are unable to separate linearly ?



① : SVM with linear classification

② : SVM with non-linear classification

well, we can actually ignore the point.

· You may be wondering which one is correct ?

- The answer is that both are correct.

It may use the hinge loss function if necessary.

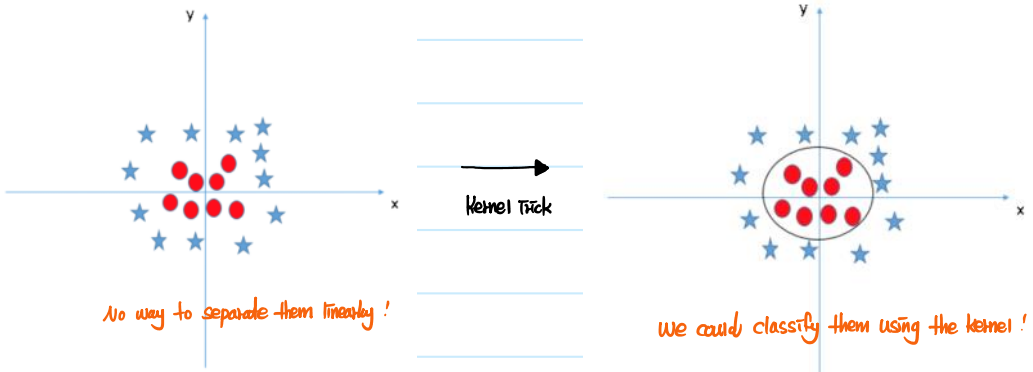
- However, keep in mind that there is trade-off.

1) ① tolerates some outlier points but it's fast.

2) ② tries to achieve 0 tolerance with perfect partition but it's slow.

: In real world, there may be millions of data taking lots of time.

· As we discussed, there is a case where we can't classify them with linearly separable way such as :



No way to separate them linearly !

We could classify them using the kernel !

· We might be able to do SVM with non-linear classification as seen before.

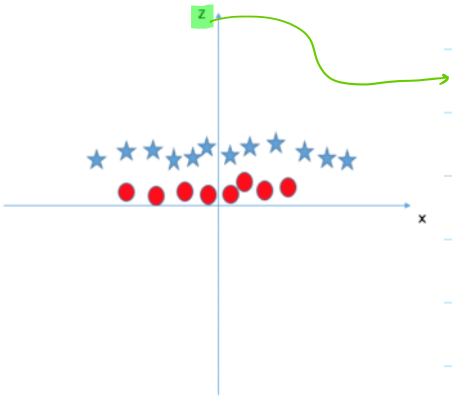
· We might be able to do SVM with non-linear classification as seen before.

- Then, how would that be possible ? ↓ the decision boundary doesn't have to be a straight line.

: It is possible by introducing the kernel.

↳ The method extends to patterns that are not linearly separable by transformations of original data to map into new space using the kernel function.

· For example, the original dataset can be transformed as following ;



→ we define a kernel function $z = x^2 + y^2$.

Here, a kernel is defined as ;

- It is a function which takes low dimensional input space and transform it to a high dimensional space.

· So, types of kernel functions ?

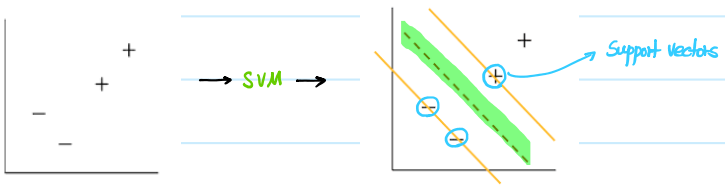
Some common kernels are following :

- Polynomial(homogeneous)
  - $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i \cdot \boldsymbol{x}_j)^d$
- Polynomial(inhomogeneous)
  - $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i \cdot \boldsymbol{x}_j + 1)^d$
- Gaussian kernel function, a.k.a. Radial Basis Function
  - $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\gamma \|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right)$
    - For $\gamma > 0$. Sometimes parameterized using $\gamma = \frac{1}{2\sigma^2}$
- Hyperbolic tangent, a.k.a. Sigmoid Function
  - $k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\kappa \boldsymbol{x}_i \cdot \boldsymbol{x}_j + c)$
  - For some(not every) $\kappa > 0$ and $c < 0$

b) Mathematical explanation (MIT AI lecture)

· As we discussed, what the SVM does is to answer the question as following ;

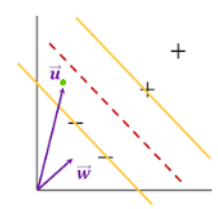
" How do we divide the space with decision boundaries ? "



· From now on, let's dive into how SVM divide the space mathematically.

· First, let's think about a vector $\vec{w}$ { constrained to be perpendicular to the decision boundary
It can be having any lengths

· Also, let's say that we have an unknown point in the space, which can be a vector $\vec{u}$ from origin.



• : unknown point

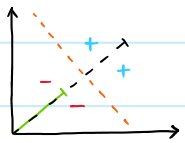
- Here, we are interested if the unknown is right side of the decision boundary or left side.

↳ In order for that, we are going to project the vector $\vec{u}$ onto $\vec{w}$ vector such that ;

↳ If the distance further out we go, the answer would be right side of the boundary.

↳ If the distance isn't far away, the answer would be left side.

e.g.



− : Left

- : Right

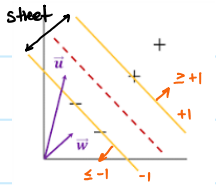
- In other words,

$\vec{w} \cdot \vec{u} \geq c \Leftrightarrow \vec{w} \cdot \vec{u} + b \geq 0$; Then positive sample (+) ; where $c = -b$ ①

c: some constant

" Decision Rule "

- Now, let's think about two points. They must follow ;



$\vec{w} \cdot \vec{x}_+ + b \geq 1$

$\vec{w} \cdot \vec{x}_- + b \leq -1$

; x_+ is a sample from positive space but still unknown

- It seems like that we have two equations, which may be painful.

↳ Let's make the math be easy by introducing a new variable y_i such that ;

↳ $y_i = 1$ if + samples

$y_i = -1$ if − samples

- Let us multiply two equations by y_i ; then, we will see a magic !

$y_i (\vec{w} \cdot \vec{x}_+ + b) \geq 1 \Leftrightarrow \vec{w} \cdot \vec{x}_i + b \geq 1$

$y_i (\vec{w} \cdot \vec{x}_- + b) \leq -1 \Leftrightarrow -\vec{w} \cdot \vec{x}_i - b \leq -1 \Leftrightarrow \vec{w} \cdot \vec{x}_i + b \geq 1$

$\therefore \vec{w} \cdot \vec{x}_i + b \geq 1$; They are identical !

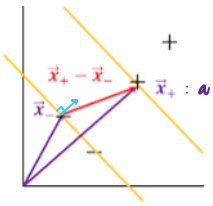
- Hence, we have

$y_i (\vec{w} \cdot \vec{x}_i + b) - 1 \geq 0$

from the equation, regarding the support vectors, we have

$$y_i(\vec{w}\cdot\vec{x}_i + b) - 1 = 0 \quad \text{; this is a sort of constraint} \qquad \text{②}$$

- okay, it's time to recall the goal of SVM, which was to maximize the width of street ;
  - SVM performs classification by finding the hyperplane that maximizes the margin between the two classes.



It can be defined as (Difference vector) · (Unit vector to be perpendicular)

$\vec{x}_+$: a vector from origin to a point located in positive area

↗ : unit normal vector

- If we can remember, we defined the perpendicular vector as $\vec{w}$

$$\therefore \text{width} = (\vec{x}_+ - \vec{x}_-)\cdot\frac{\vec{w}}{\|\vec{w}\|} \quad \text{; It will turn out a scalar which is the width}$$

- Now, let's use ② equation to rearrange a numerator of the width ;

$$\text{width} = \frac{\vec{x}_+\cdot\vec{w}}{\|\vec{w}\|} - \frac{\vec{x}_-\cdot\vec{w}}{\|\vec{w}\|}$$

$$= \frac{1-b}{\|\vec{w}\|} - \frac{(-1-b)}{\|\vec{w}\|} \quad \text{; think about } y_i = +1 \text{ if positive sample}$$

$$= \frac{1-b+1+b}{\|\vec{w}\|}$$

$$= \frac{2}{\|\vec{w}\|}$$

$$\therefore \text{width} = \frac{2}{\|\vec{w}\|} \qquad \text{③}$$

- So, we want to maximize the width ;

$$\max \frac{2}{\|\vec{w}\|} \Leftrightarrow \max \frac{1}{\|\vec{w}\|} \Leftrightarrow \min \|\vec{w}\| \Leftrightarrow \min \frac{1}{2}\|\vec{w}\|^2$$

⇒ why do we do this ?

: For sake of mathematics !

- Now, it feels like that we are formulating the optimization problem for Support vector machine !

$$\text{Minimize} \quad \frac{1}{2}\|\vec{w}\|^2$$

$$\text{Subject to} \quad y_i(\vec{w}\cdot\vec{x}_i + b) - 1 \geq 0$$

$\Leftrightarrow -\left[y_i (\vec{w} \cdot \vec{x}_i + b) - 1 \right] \leq 0$; Generalize the constraint optimization problem

· Generally speaking,

- Solving unconstrained optimization problem may be easier than solving constrained optimization problem.

· There are two types of methods to handle constrained optimization problem
  - Direct method
  - Indirect method
    - Exterior penalty function
    - Interior penalty function
    - Augmented Lagrangian

· In this case, we are going to use a Lagrangian function such that the constraint could be ignored ; hence, unconstrained prob.

$$J = f(x) + \sum_{j=1}^{m} \lambda_j g_j(x) + \sum_{k=1}^{l} \lambda_{m+k} h_k(x)$$; they are all in generalized forms

$$\Leftrightarrow \boxed{J = \frac{1}{2}\|\vec{w}\|^2 - \sum_{j=1}^{m} \lambda_j \left[y_j (\vec{w} \cdot \vec{x}_j + b) - 1 \right]} \quad ④$$; λ_j is a lagrangian multiplier

· As we know, we can have a minimum by taking derivative of J making it equal to zero.

· Since J is a function of both $\vec{w}$ and b,

$$\frac{dJ}{d\vec{w}} = \vec{w} - \sum_{j=1} \lambda_j y_j x_j = 0 \qquad \therefore \boxed{\vec{w} = \sum_{j=1} \lambda_j y_j x_j} \quad ⑤$$

$$\frac{dJ}{db} = -\sum_{j=1} \lambda_j y_j = 0 \qquad \therefore \boxed{\sum_{j=1} \lambda_j y_j = 0} \quad ⑥$$

· As can be seen, the math will sing a song !

: The vector $\vec{w}$ is a linear sum of the samples.

↳ It actually didn't have to be like the format. It could be very complicated.

· As the last step, let us substitute ⑤ into ④.

$$J = \frac{1}{2}\left(\sum \lambda_i y_i \vec{x}_i\right)\left(\sum \lambda_j y_j \vec{x}_j\right) - \sum \lambda_i y_i \vec{x}_i \cdot \sum \lambda_j y_j \vec{x}_j - \sum \lambda_i y_i b + \sum \lambda_i$$

($\sum \lambda_i y_i b = 0$ (∵ ⑥))

$$\Leftrightarrow \boxed{J = \sum \lambda_i - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j \vec{x}_i \cdot \vec{x}_j} \quad ⑦$$

It is discovered that the optimization problem is only related to a dot product of samples.

· In a similar way, it is observed that the decision rule is also only related to the dot product.

$\vec{w} \cdot \vec{w} + b \geq 0$; Then positive sample (+)

$\Leftrightarrow \sum \lambda_i y_i \vec{x}_i \cdot \vec{w} + b \geq 0$

- okay, it's done. Let's summarize it.

  - Let's say that we really want to recognize the hand writing for some reasons.

  - The problem may consist of either yes or no classification problems.

  - Neural network may be considered as the best option for the problem.

  - However, the NN might be stuck in the local minimum.

  - On the other hand, the SVM may be better than NN because it's always formulated as convex optimization problem.

    ⇒ If you find an optimum via SVM, it is guaranteed that it's a global optimum.

- The purpose of SVM is to classify the design space by defining the hyperplane where the width is maximized.

  - In order to maximize the width, we formulate the optimization problem with Lagrangian.

  - From the equation ⑦, we may be able to calculate the Lagrangian multipliers.

  - From the equation ⑤, we then can calculate $\vec{w}$.

  - From the equation ②, we can calculate b.

  - Therefore, we may be able to draw the best straight line in the space.

- What if we have a space where we can't separate linearly ?

  - As we discussed, we are going to use any of kernel functions.

  - And then, we are going to use the same technique.

Pros and cons

Pros and Cons associated with SVM

- **Pros:**
  - It works really well with clear margin of separation
  - It is effective in high dimensional spaces.
  - It is effective in cases where number of dimensions is greater than the number of samples.
  - It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

- **Cons:**
  - It doesn't perform well, when we have large data set because the required training time is higher
  - It also doesn't perform very well, when the data set has more noise i.e. target classes are overlapping

SVM regression

- In fact, most of the examples of SVMs are related to classification.

- However, SVM technique is actually able to be used for regression. (Sometimes, it is called as Support Vector Regressor)

- Then, how does it work ? Let us explain it intuitively !

- Recall that ;

  - In SVM, we are trying to maximize the margin by formulating the problem as a convex optimization problem ; $\min. \frac{1}{2}w^2$

· The goal of SVM is actually done both in the classification and regression.

a) Classification : Maximize the margin such that data points are correctly classified as much as possible.

b) Regression : Maximize the margin such that the data points deviate less than the required accuracy ε from $f(x)$.

↳ Let us talk a little bit more about this case.

· In regression case,

- The goal is to find a function $f(x) = wx + b$ (Red line) under the condition

- The condition is that $f(x)$ is within a required accuracy ε of every data point.

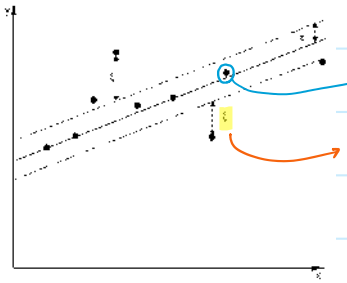
e.g. $|y(x) - f(x)| \leq \varepsilon$; where ε is the distance between the red and the grey line.

So, what is the significance of this value ?

↳ In the same way as with classification approach, we seek and optimize the generalization bounds given for regression.

: The regression relies on the loss function that { no costs if the data points are within ε / Yes costs if the data points are outside of the boundary

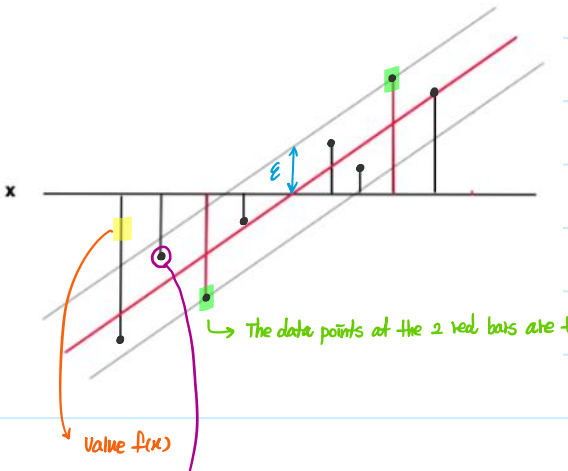
Called as epsilon intensive func.



→ The cost is zero because it is inside the band.

→ Cost of the error on the training point

· Let us take an example.



↳ The data points at the 2 red bars are the support vectors in this case.

Value $f(x)$

Unlike SVM classification, the point is actually located in inside of the zone ? How come ?

- In classification case, we create a safety boundary from both sides of the hyperplane.

- In regression, we want the data point to be as close as possible to the hyperplane.

· So, why is it good idea ?

: Usually, linear regression tries to fit a line to data by minimizing a cost function. SVM does do it as well.

↳ However, in SVM, we actually can deploy a non-linear kernel : it means we will be able to create non-linear regression model.

In addition, as we discussed, there is absence of local minimum.