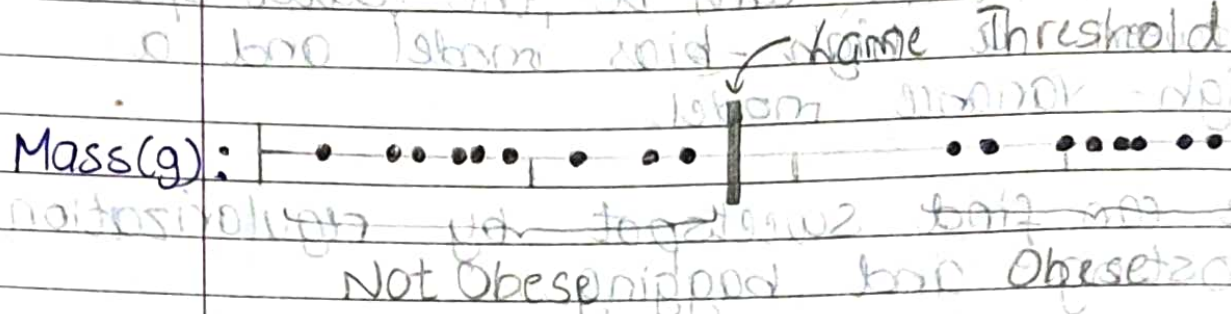
Date: / /
Page No.

SUPPORT VECTOR MACHINES



Suppose we measure the mass of a bunch of mice and have data classifying them as obese and non-obese.

Based on these observations, we can pick a threshold and when we get a new observation, we can use it classify the example as obese or not obese.

But clearly, this threshold lies way closer to the not-obese observations than to the obese observations. So this threshold is pretty lame?

We can do better by using the observations on the edges of each cluster and use the mid-point between them as the threshold. This will classify better.

The shortest distance between the observations and the threshold is called the margin.

Date: / /
Page No.

OR MACHINES

ame Threshold?

Obese

the mass of a bunch
data classifying them
on-obese.

rvations, we can pick
en we get a new
use it classify the
or not obese.

eshold lies way
not-obese observati-
obese observations.
is pretty lame?

y using the observat-
of each cluster and
between them as
will classify better

nce between the obser-
hreshold is called the

Date: / /
Page No.

Mass(g):

Margin Margin

- When the threshold/ decision boundary is halfway and the 2 margins are equal then we get the largest margin to make classification and we are said to be using a Maximal Margin classifier.

Maximal Margin classifiers may seem pretty cool, but what if our training data looks like this -

Mass(g):

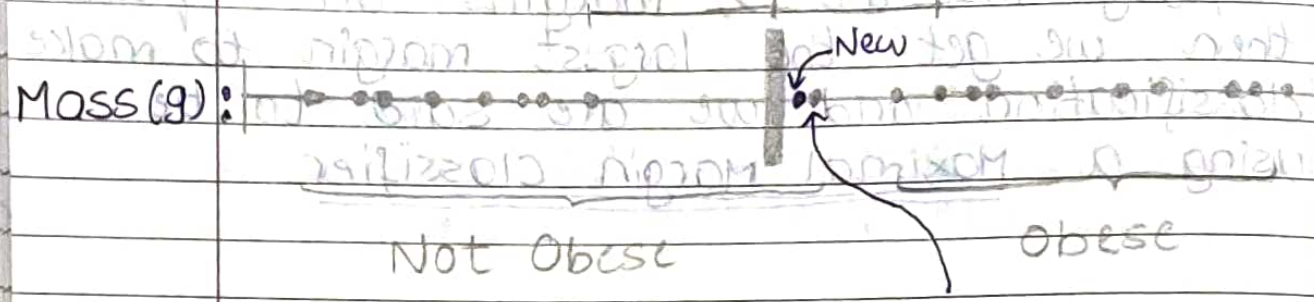
Not obese Obese

In this case, we have a couple of outliers where the observation(s) classified as not obese are super close to the obese observations.
If we use MMC for this, it will be quite close to the obese observations and will or might classify many obese testing examples as not-obese.

- Maximal Margin classifiers are very sensitive to outliers and that makes them pretty lame.

To make a threshold that is not so sensitive to outliers, we must allow misclassifications.

For example, if we put the threshold as shown in the figure:



then we'll misclassify this observation but now when we get a new observation as shown, then it'll be classified as obese which makes sense as it's closer to the obese observations

This is an example of the Bias/Variance Tradeoff that plagues all of machine learning.

This example has higher bias but lower variance.

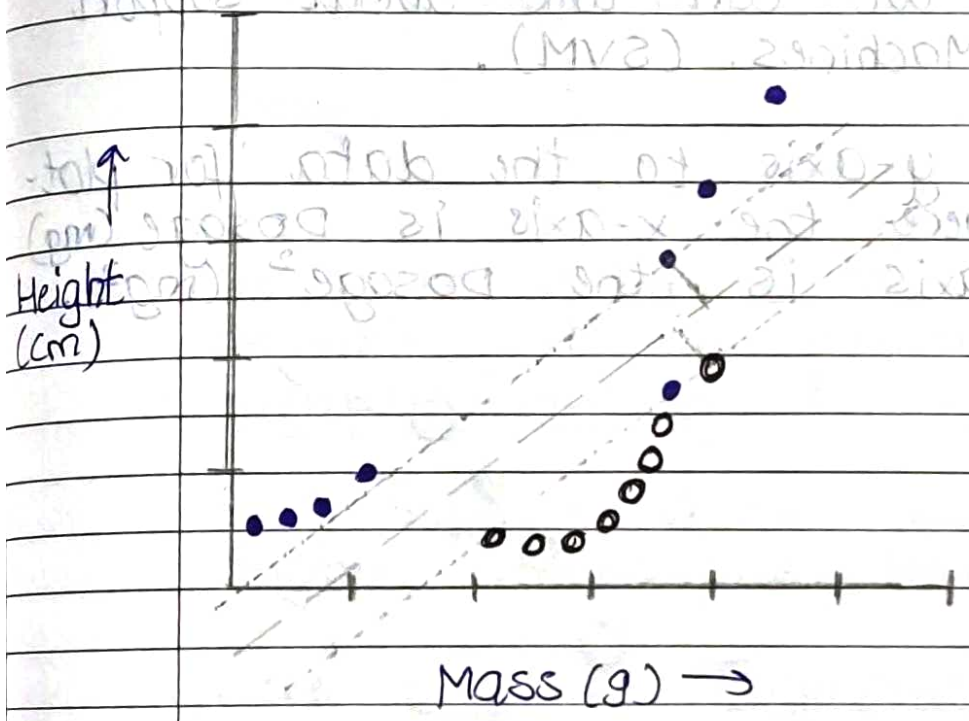
When we allow misclassifications the distance between the threshold and the observations is called soft margin.

And we use cross validation to get the best soft margin.

When we use a soft margin to determine the location of a threshold then we're using a Soft Margin Classifier

AKA a Support Vector Classifier to classify observations.

- The observations on the edge and within the soft margin are called support vectors.

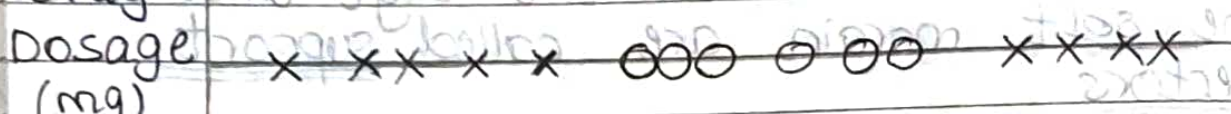


This data is 2-dimensional and thus the support vector classifier will be a line.
In this case we have measured the soft margin from the 2 points shown and allowed one misclassification. Cross validation showed us that this will give us better results in the long run.

Date: / /
Page No.

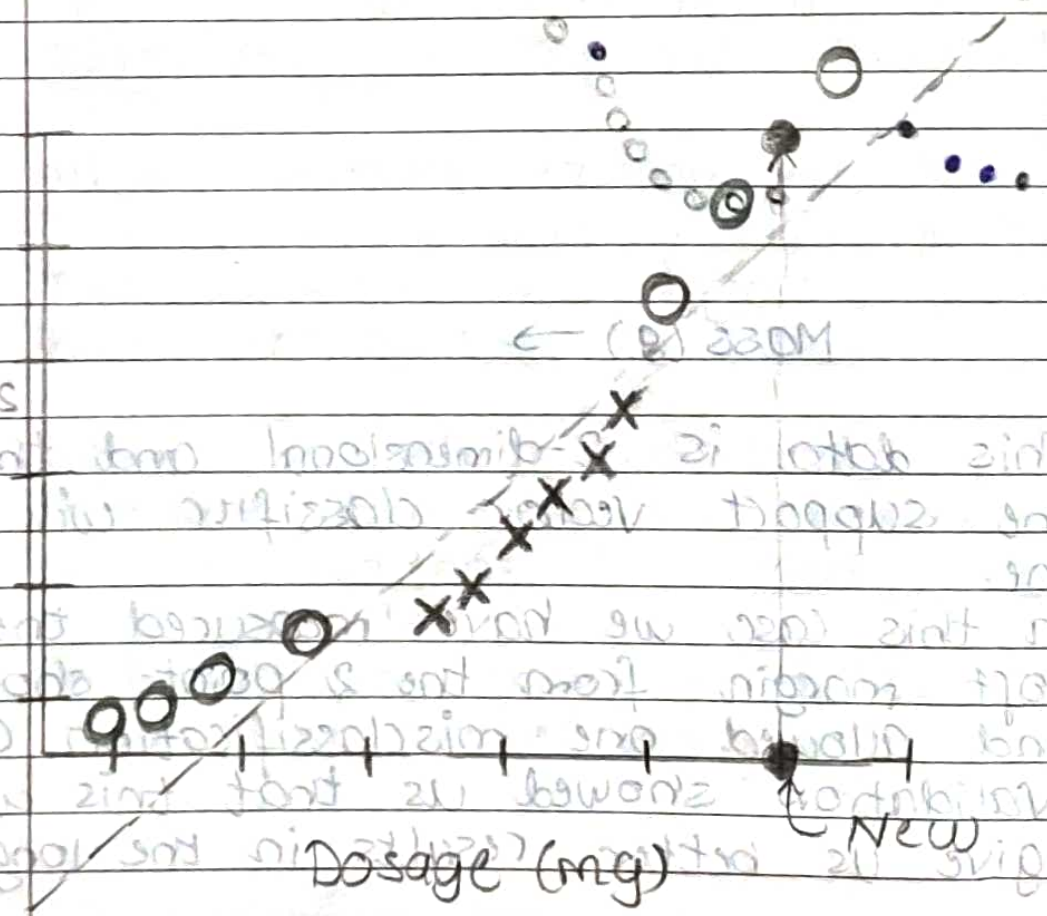
Generally, all flat affine subspaces are called hyperplanes.

Maximal Margin classifiers and Support Vector Classifiers cannot deal with data like this:



So now we can talk about Support Vector Machines. (SVM).

Add a y-axis to the data for plotting, where the x-axis is Dosage (mg) and y-axis is the Dosage^2 (mg^2).



Date: / /
Page No.

And now we can draw a Support Vector Classifier which will be a line separating the cured from the not-cured.

If a new observation had this Dosage as shown then we can calculate the y-axis co-ordinate by squaring the value and classify the observation as not-cured

Q But why did we take Dosage squared on the y-axis? Why not $Dosage^3$? or $\sqrt{Dosage}$?

In order to make the mathematics possible, SVM use something called Kernel Functions to systematically find Support Vector classifiers in higher dimensions. Here's how:

Here we use Polynomial Kernel, which has a parameter, d, which stands for the degree of the polynomial. When $d = 1$, the Polynomial Kernel computes the relationships b/w each pair of observations in 1-D and these relationships are used to build a SVC.

Date: / /
Page No.

The Polynomial Kernel systematically increases dimensions by setting d, the degree of the polynomial and the relationships between each set / pair of observations are used to find a SVC. Lastly, we can find a good value for d with cross validation.

Commonly used Kernel, Radial Kernel also called Radial Basis Function Kernel.

Radial Kernel finds SVC in infinite dimensions.

It behaves like a Weighted Nearest Neighbor model when we deal with testing data. The closest observations have a lot of influence on how we classify the new observation.

Kernel functions only calculate the relationships between every pair of point as if they are in the higher dimension. They don't actually do the transformation. This Trick of performing the calculations of the high-dimensional relationships without actually transforming the data to the higher dimension is called The Kernel Trick

This reduces the amount of compatation required for SVM by avoiding the math that transforms the data from low to high dimensions and also

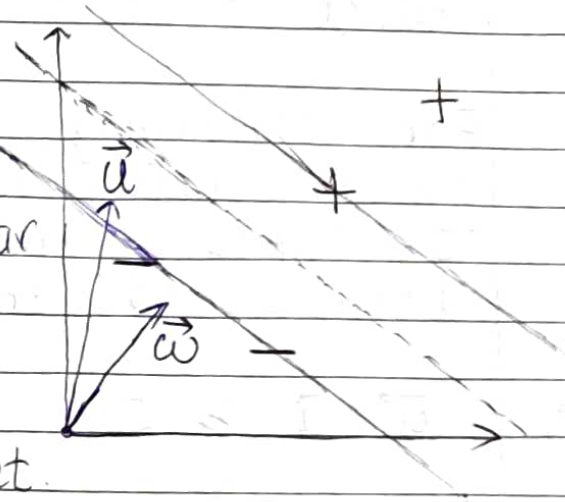
Date: / /
Page No.

make Radial Kernel calculations possible.

Date: / /
Page No.

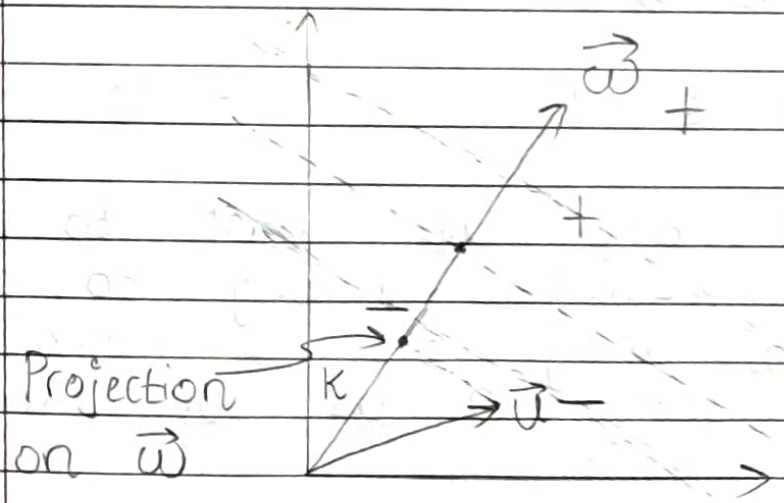
SUPPORT VECTOR MACHINES

Imagine that we have a vector $\vec{w}$ such that it is always perpendicular to the Decision Boundary (Gutter). We don't know about its length yet.



Now, we have an unknown vector $\vec{u}$ and we want to make a prediction whether it belongs to class 1 (+) or -1 (-).

We can figure this out by projecting $\vec{u}$ on $\vec{w}$ and then finding the length of this projection. If this projection crosses the median line, then that means u belongs to the + class and vice versa.



Date: / /
Page No.

Now, the projection of $\vec{u}$ on $\vec{w}$ is

$$\vec{w} \cdot \vec{u} = \vec{u} \cdot \vec{w} = k \text{ (say)}$$

For classifying, this k has to be greater than or less than some value c. (where c may be the perpendicular distance to the median along $\vec{w}$)

$\therefore$ $\vec{w} \cdot \vec{u} \geq c$ (check if true)

$\Rightarrow$ $\boxed{\vec{w} \cdot \vec{u} + b \geq 0}$ DECISION RULE (Then + sample) ($b = -c$)

The problem is, we don't know what is $\vec{w}$ and b. So we need some more constraint(s).

Let's assume we have a + sample x_+ and a - sample x_-. So, we want that

$$\vec{w} \cdot \vec{x}_+ + b \geq 1 \quad - (i)$$

$$\vec{w} \cdot \vec{x}_- + b \leq -1 \quad - (ii)$$

For consistency, we want to be sure that our samples belong to particular class. So, using ± 1 is more definitive than using 0 as our limit / threshold.

Date: / /
Page No.

Introduce y_i such that it's the true class of the i^{th} sample.

$\therefore$ $y_i = +1$ for + samples — (iii)

$y_i = -1$ for − samples. — (iv)

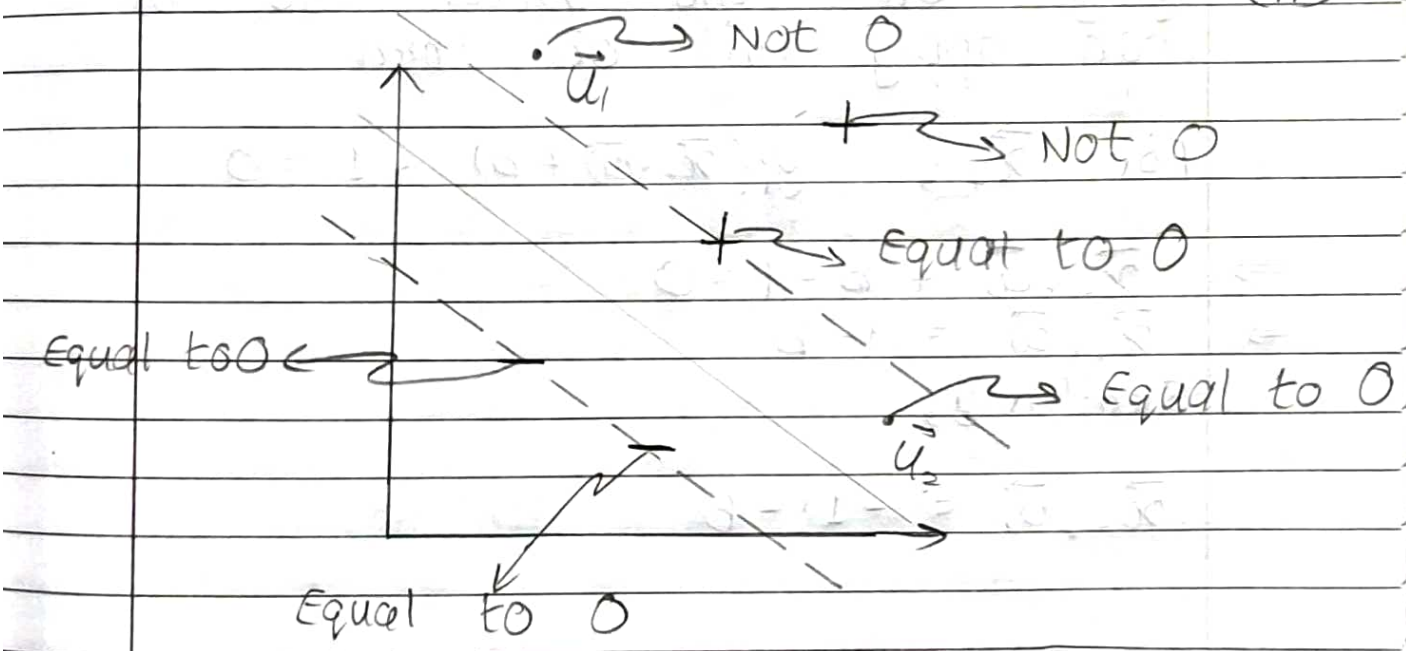
Multiplying (iii) and (iv) with (i) and (ii) respectively.

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

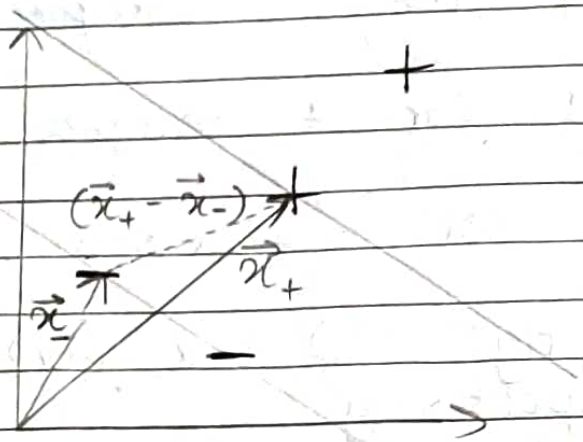
$y_i (\vec{w} \cdot \vec{x}_- + b) \geq 1$ } Same!

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

NOTICE: $\boxed{y_i (\vec{w} \cdot \vec{x}_i + b) - 1 = 0}$ for all x_i's in the gutter / street. — (vi)



Now, let's talk about the width of the gutter / street.



Technically, the width of the gutter is the difference between x_+ and x_- in the direction of $\vec{w}$ (perpendicular to the median, parallel to the width)

$$\text{WIDTH} = (\vec{x}_+ - \vec{x}_-) \cdot \frac{\vec{w}}{|\vec{w}|}$$

Remeber that x_+ and x_- are the distance from origin to the points lying "on" the gutter. So, we can apply (vi) on them

For $\vec{x}_+$, $\quad y_i(\vec{x}_+ \cdot \vec{w} + b) - 1 = 0$

$\Rightarrow \quad \vec{x}_+ \cdot \vec{w} + b - 1 = 0$

$\Rightarrow \quad \vec{x}_+ \cdot \vec{w} = 1 - b$

Similarly,

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

Date: / /
Page No.

i.e. $\text{WIDTH} = (\vec{x}_+ \cdot \vec{\omega} - \vec{x}_- \cdot \vec{\omega}) \frac{1}{|\vec{\omega}|}$

$\text{WIDTH} = \frac{(1-b+1+b)}{|\vec{\omega}|} = \frac{2}{|\vec{\omega}|}$

$$\boxed{\text{WIDTH} = \frac{2}{|\vec{\omega}|}}$$ So, Margin $= \frac{2}{|\vec{\omega}|}$

Now, our aim is to maximize the width of the gutter.

$\Rightarrow$ $\text{MAX}\left(\frac{2}{|\vec{\omega}|}\right) \Rightarrow \text{MIN}\,(|\vec{\omega}|)$

$\Rightarrow$ $\underset{\text{MIN}}{\text{MAX}}\left(\frac{1}{2}|\vec{\omega}|^2\right)$ for mathematical convinience.

Now, we have some equations and some constraints. This means that we'll have to use Lagrange Multiplier

$$\mathcal{L} = \frac{1}{2}|\vec{\omega}|^2 - \sum_i \alpha_i [y_i(\vec{\omega}\cdot\vec{x}_i + b) - 1] \quad - (vii)$$

$$\frac{\partial \mathcal{L}}{\partial \omega} = \vec{\omega} - \sum_i \alpha_i y_i x_i = 0$$

$\Rightarrow$ $$\boxed{\vec{\omega} = \sum_i \alpha_i y_i \vec{x}_i} \quad - (viii)$$

$$\frac{\partial \mathcal{L}}{\partial b} = -\sum_i \alpha_i y_i = 0 \Rightarrow \boxed{\sum_i \alpha_i y_i = 0} \quad - (ix)$$

The decision vector $\vec{w}$ is a linear sum of the samples $\vec{x}_i$'s.

Placing (viii) in (vii)

$$\mathcal{L} = \frac{1}{2}\left(\sum_i \alpha_i y_i \vec{x}_i\right)\left(\sum_j \alpha_j y_j \vec{x}_j\right) - \cdots$$

$$\sum_i \alpha_i y_i x_i \left(\sum_j \alpha_j y_j x_j\right) - \underbrace{\sum_i \alpha_i y_i b}_{0} + \sum_i \alpha_i$$

$$\boxed{\mathcal{L} = -\frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j x_i \cdot x_j + \sum_i \alpha_i}$$

Interestingly, the optimization depends on the sum of the dotted pairs of the samples. We have to find the maximum of this expression which is the job of numerical analysts. :(

The DECISION RULE now,

$$\sum_i \alpha_i y_i \vec{x}_i \cdot \vec{u} + b \geq 0 \quad \text{THEN} +$$

$$\sum_i \alpha_i y_i \vec{x}_i \cdot \vec{u} + b \leq 0 \quad \text{THEN} -$$

NOTE: The lagrange multipliers α_i's for the points on / in the gutter are non-zero, and for points not in the gutter, α_i's are zero.

Date: / /
Page No.

We had something called a CONSTRAINED OPTIMIZATION PROBLEM which is solved using Lagrangian Multiplier method. This is quadratic because :

$$\min f(x) = \frac{1}{2}|\vec{w}|^2 \qquad \text{(Quadratic)}$$

$$g(x): \quad [y_i(\vec{w}.\vec{x}_i) + b] - 1 = 0$$

The key to Lagrangian Multiplier Method is to find intersection of f and g at a tangent point where the gradients of f & g are proportional.

$$\Rightarrow \quad \nabla f(P) = \lambda \nabla g(P)$$
$$g(x) = 0$$

$$\Rightarrow \quad \mathcal{L}(x, \alpha) = f(x) - \alpha g(x)$$
$$\nabla \mathcal{L}(x, \alpha) = 0$$

At a solution p :

- The constraint g and the contour lines of f must be tangent.
- If they are tangent, their gradient vectors are parallel.
- Gradient of g must be 0

$\therefore$ For SVM,

$$\min \mathcal{L}_p = \frac{1}{2}|\vec{w}|^2 - \sum_i \alpha_i y_i (\vec{w}.\vec{x}_i + b) + \sum_i \alpha_i$$

$$\Rightarrow \quad w = \sum_i \alpha_i y_i \vec{x}_i \qquad \sum_i \alpha_i y_i = 0 \quad \text{as before}$$

Date: / /
Page No.

Now, this is the primal problem $\mathcal{L}_P$. We will solve the optimization problem by solving for the dual of this primal problem.

The Langrangian Dual Problem : Instead of minimizing over $\vec{\omega}, b$ subject to constraints involving α's, we can maximize over α (the dual variable) subject to the relations obtained prev. for $\vec{\omega}$ and b

$$\vec{\omega} = \sum_i \alpha_i y_i \vec{x}_i \quad , \quad \sum_i \alpha_i y_i = 0$$

Primal Problem :

$$\min \mathcal{L}_P = \frac{1}{2} |\vec{\omega}|^2 - \sum_i \alpha_i y_i (\vec{\omega} . \vec{x}_i + b) + \sum_i \alpha_i$$

where $\vec{\omega} = \sum_i \alpha_i y_i \vec{x}_i \quad , \quad \sum_i \alpha_i y_i = 0$

Dual Problem :

$$\max \mathcal{L}_D(\alpha_i) = \sum_{i=1} \alpha_i - \frac{1}{2} \sum_i \alpha_i \alpha_j y_i y_j (x_i . x_j)$$

$$\text{s.t.} \quad \sum_i a_i y_i = 0$$

NOTE: We removed the dependence on $\vec{\omega}$ & b by placing in their values in the dual problem expression.

Date: / /
Page No.

Now, why did we do this? Even though the dual problem will give us the same solution as the primal problem but note that now our optimization problem is solving by computing the inner products of x_i and x_j which will be important for solving non-linearly separable classification problems. Earlier, we would have had the burden of computing $\vec{w}$ and b

Solving this DUAL PROBLEM will give us what we got earlier

$$\sum_i \alpha_i y_i = 0 \qquad 0 \leq \alpha_i \leq C$$

and $\vec{w} = \sum_i \alpha_i y_i \vec{x}_i$

Thus $f(x) = \vec{u} \cdot \vec{w} + b = \left(\sum_i \alpha_i y_i \vec{x}_i \cdot \vec{u}\right) + b$

A lot of the α's will be 0 as discussed before which reduces the dimensionality of the solution.

A HINT as to why inner product $\vec{x}_i \cdot \vec{x}_j$ are useful is that it provides the cosine of the angle between two unit vectors = how 'far apart' they are.

Eg: $\vec{x} = [1\ 0]^T \qquad \vec{y} = [0\ 1]^T$

if they are parallel, their inner product is 1 (completely similar)

$\vec{x}^T \vec{y} = \vec{x} \cdot \vec{y} = 1$ and vice versa

Date: / /
Page No.

NON-LINEARLY SEPARABLE DATA :

Map the data to higher dimensions so that a higher dimensional hyperplane can separate the data which is not separable in the lower-dimensional space.

$$\mathcal{L}_D = \sum_i \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j (x_i \cdot x_j)$$

Imagine a function ϕ that maps the data into another space. If we now transform to ϕ, instead of computing this $(\bar{x}_i \cdot \bar{x}_j)$ we will have to compute $\phi(x_i) \cdot \phi(x_j)$ which is very expensive and time consuming.

If there is some "kernel function" K such that $K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$ then we don't need to know or compute ϕ at all. K will define the inner products and/or similarity in the transformed space.

∴ Now, $$\mathcal{L}_D = \sum \alpha_i - \frac{1}{2} \sum_i \sum_j \alpha_i \alpha_j y_i y_j K(x_i, x_j)$$

Some Kernels that are used :

Date: / /
Page No.

a) $K(x_i, x_j) = (x_i \cdot x_j + 1)^p$ where p is a tunable parameter

NOTE: We don't need $\phi(x_i)$ & $\phi(x_j)$

b) $K(\vec{x}, \vec{y}) = e^{-\left\{\frac{|x-y|^2}{2\sigma^2}\right\}}$

c) $K(\vec{x}, \vec{y}) = \tanh(\kappa \vec{x} \cdot \vec{y} - \delta)$

REFER TO An Idiot's guide to Support Vector Machines (SVMs)