CSE 555
Assignment 3
Name: Rajiv Ranjan
Ub id:50249099

1. Write code to train a multi-class support vector classifier with dot-product kernel and 1-norm soft margin using the MNIST training data set. Then reporting the performance using MNIST test data set. There is a hyper-parameter that sets the trade-off between the margin and the training error --- tune this hyper-parameter through cross-validation.

Soln: The code is being sent as an attachment in the file svm1.py and svm2.py

Set up for the code: Cross validation Score has been taken as 5. For validating the training data set through Cross Validation often an odd number is taken.

I have written 2 set of codes: First one is SVM1.py which runs on the entire MNIST data set, as per the requirement a dot-product kernel has been used that means the kernel was selected to be linear. The cache size was defined to be 1000. Also it was required that the hyper parameter C and gama be varied and it was varied. Below is the output for various values of C and gama.

O/p

```
scaling
grid search
Fitting 5 folds for each of 20 candidates, totalling 100 fits
[CV] C=100000.0, gamma=0.03125 .....................................
[CV] C=100000.0, gamma=0.03125 .....................................
[CV] C=100000.0, gamma=0.03125 .....................................
[CV] C=100000.0, gamma=0.03125 .....................................
[CV] .............. C=100000.0, gamma=0.03125, score=0.781745 -25.8min
[CV] C=100000.0, gamma=0.03125 .....................................
[CV] .............. C=100000.0, gamma=0.03125, score=0.785336 -26.0min
[CV] C=100000.0, gamma=0.0625 ......................................
[CV] .............. C=100000.0, gamma=0.03125, score=0.782500 -26.0min
[CV] C=100000.0, gamma=0.0625 ......................................
[CV] .............. C=100000.0, gamma=0.03125, score=0.785090 -26.1min
[CV] C=100000.0, gamma=0.0625 ......................................
[CV] .............. C=100000.0, gamma=0.03125, score=0.784328 -24.9min
[CV] C=100000.0, gamma=0.0625 ......................................
[CV] ............... C=100000.0, gamma=0.0625, score=0.778250 -62.1min
[CV] C=100000.0, gamma=0.0625 ......................................
[CV] ............... C=100000.0, gamma=0.0625, score=0.779753 -62.2min
[CV] C=100000.0, gamma=0.125 .......................................
[CV] ............... C=100000.0, gamma=0.0625, score=0.780841 -62.8min
[CV] C=100000.0, gamma=0.125 .......................................
[CV] ............... C=100000.0, gamma=0.0625, score=0.776494 -63.4min
```

```
[CV] C=100000.0, gamma=0.125 ........................................
[CV] ............... C=100000.0, gamma=0.0625, score=0.680077 -64.5min
[CV] C=100000.0, gamma=0.125 ........................................
[CV] ............... C=100000.0, gamma=0.125, score=0.692795 -147.0min
[CV] C=100000.0, gamma=0.125 ........................................
[CV] ............... C=100000.0, gamma=0.125, score=0.679853 -146.6min
[CV] C=100000.0, gamma=0.25 .........................................
[CV] ............... C=100000.0, gamma=0.125, score=0.688000 -146.7min
[CV] C=100000.0, gamma=0.25 .........................................
[CV] ............... C=100000.0, gamma=0.125, score=0.683721 -142.5min
[CV] C=100000.0, gamma=0.25 .........................................
[CV] ............... C=100000.0, gamma=0.125, score=0.700717 -151.0min
[CV] C=100000.0, gamma=0.25 .........................................
[CV] ................ C=100000.0, gamma=0.25, score=0.441233 -168.4min
[CV] C=100000.0, gamma=0.25 .........................................
[CV] ................ C=100000.0, gamma=0.25, score=0.376487 -160.6min
[CV] C=1000000.0, gamma=0.03125 .....................................
[CV] ............. C=1000000.0, gamma=0.03125, score=0.785090 -16.0min
[CV] C=1000000.0, gamma=0.03125 .....................................
[CV] ................ C=100000.0, gamma=0.25, score=0.419333 -151.4min
[CV] C=1000000.0, gamma=0.03125 .....................................
[CV] ............. C=1000000.0, gamma=0.03125, score=0.785336 -17.1min
[CV] C=1000000.0, gamma=0.03125 .....................................
[CV] ............. C=1000000.0, gamma=0.03125, score=0.782500 -17.1min
[CV] C=1000000.0, gamma=0.03125 .....................................
[CV] ............. C=1000000.0, gamma=0.03125, score=0.781745 -16.0min
[CV] C=1000000.0, gamma=0.0625 ......................................
[CV] ............. C=1000000.0, gamma=0.03125, score=0.784328 -16.4min
[CV] C=1000000.0, gamma=0.0625 ......................................
[CV] ................ C=100000.0, gamma=0.25, score=0.398150 -122.4min
[CV] C=1000000.0, gamma=0.0625 ......................................
[Parallel(n_jobs=-1)]: Done  24 tasks      | elapsed: 508.6min
```

Inference: This shows the various C and gama values used: they were calculated using the below line of code which is self explanatory:
parameters = {'C':10. ** np.arange(1,5), 'gamma':2. ** np.arange(-5, -1)}

Also we see that the code ran for a lot of time approximately 10 hours. Also the best prediction score was found to be around 0.78 i.e around 78 percent accurate, which is good for a linear SVM classifier.

The various gamma values were: 0.25, 0.03125, 0.125, 0.0625.

Various C values was: 10000,100000 etc

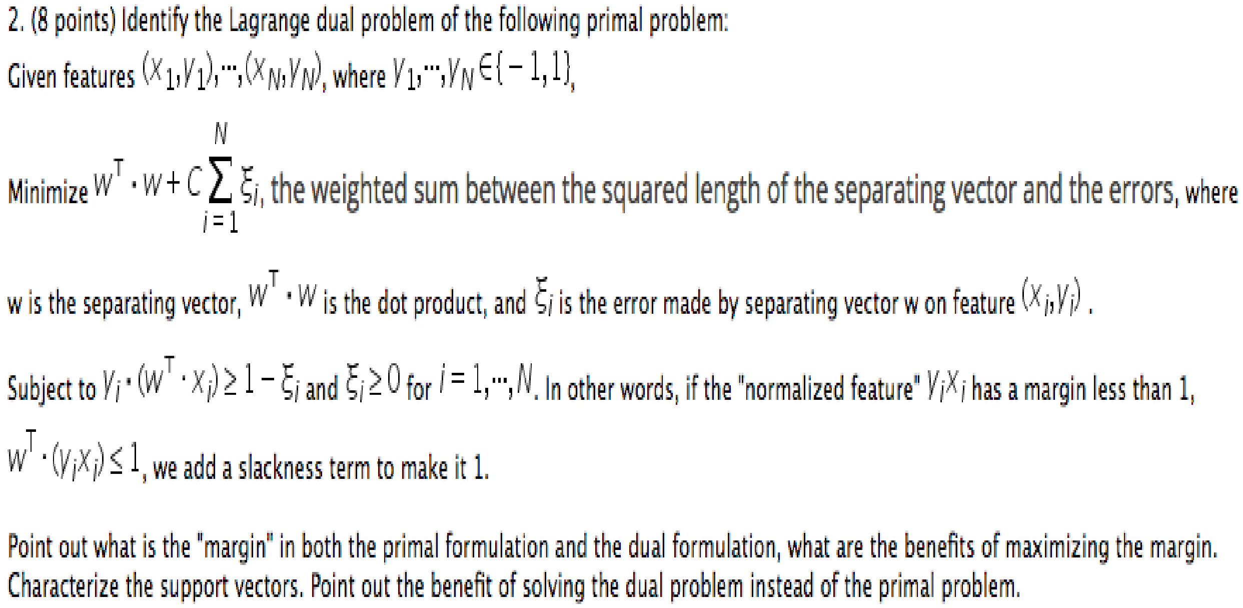
The accuracy score for a total run of 10 hrs has been shared above.

2nd Part: I also wrote a second code to test the accuracy. The code for this is in SVM2.py This code was run a small data set to check how well the classifier can distinguish between 2 different digits, let's see if it can differentiate between 8 and 9 how well. Cross validation score was taken to be: 5
For this purpose, load_digits dataset from the sklearn list of datasets was used.
It is a subset of the MNIST data set.
Also for this optunity package was used.
1)Now firstly it was run with default parameters and the accuracy was found to be 0.7655589359455676.
2)The program was run for all the 3 models linear kernel, polynomial kernel and the rbf kernel.
The various adjusted best hyperparametrs found were :
Optimal parameters {'kernel': 'rbf', 'C': 5.807812500000001, 'coef0': None, 'degree': None, 'logGamma': -3.4678653190880104}
Best Score of tuned SVM: 0.985

But the assignment specifically asks to run for linear i.e. dot product:
So the output for that was also run and could be seen below:

C	coef0	degree	kernel	logGamma	value
1.742604	NaN	NaN	linear	NaN	0.962069
0.117604	NaN	NaN	linear	NaN	0.962069
0.210938	NaN	NaN	linear	NaN	0.962069
1.835938	NaN	NaN	linear	NaN	0.962069
0.024271	NaN	NaN	linear	NaN	0.962069
1.649271	NaN	NaN	linear	NaN	0.962069

The different C hyperparameter which tells how much to avoid misclassifying and the different score values of accuracy are also shown.



2. (8 points) Identify the Lagrange dual problem of the following primal problem:

Given features $(x_1,y_1),\cdots,(x_N,y_N)$, where $y_1,\cdots,y_N \in \{-1,1\}$,

Minimize $w^T \cdot w + C\sum_{i=1}^{N}\xi_i$, the weighted sum between the squared length of the separating vector and the errors, where w is the separating vector, $w^T \cdot w$ is the dot product, and ξ_i is the error made by separating vector w on feature (x_i,y_i) .

Subject to $y_i \cdot (w^T \cdot x_i) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for $i = 1,\cdots,N$. In other words, if the "normalized feature" $y_i x_i$ has a margin less than 1, $w^T \cdot (y_i x_i) \leq 1$, we add a slackness term to make it 1.

Point out what is the "margin" in both the primal formulation and the dual formulation, what are the benefits of maximizing the margin. Characterize the support vectors. Point out the benefit of solving the dual problem instead of the primal problem.

Soln:

Given is $(x_1, y_1), \ldots (x_N, y_N)$ where $y_1, \ldots y_N \in \{-1, 1\}$ is basically N points where $x_1 \ldots x_n$

Soln:- is a feature vector, y_N is a binary class can have value either as -1 or 1

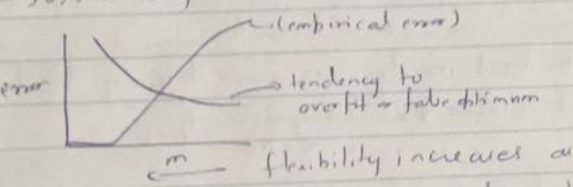
Part 1 · Minimize $w^T \cdot w + C \sum_{i=1}^{N} \xi_i$

where w is the separating vector, $w^T w$ is the dot product & ξ_i is the error made by separating vector w on feature (x_i, y_i)

subject to. $y_i \cdot (w^T \cdot x_i) \geq 1 - \xi_i$

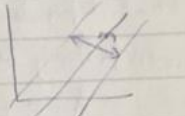
$\xi_i \geq 0$ for $i = 1 \ldots N$

Sol^n:- So, the thing is the actual SVM is a mixture of 2



flexibility increases as we go right

here m is the margin assumed is width of the margin



∴ so, for actual SVM we have already seen all the empirical error, and we now add something that will quantify our tendency to overfit, so coming back to this formula here, this was given the margin m.

$$\underset{w, b, \xi}{\arg\min} \sum_{i=1}^{n} \xi_i \quad \text{such that} \quad y_i(w^T x_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

$$\|w\| \leq \frac{1}{m}$$

so we will consider $\sigma(w) = w^Tw$ or $||w||$ (2)

But suppose we don't know m from advance in that case, I define it as in a form such that

$$\underset{w, b, \xi}{\arg\min} \ \lambda \ \sigma(w) + \sum_i \xi_i \text{ s.t. } y_i(w^Tx_i + b) \geq 1 - \xi_i, \quad \xi_i \geq 0$$

such the length of my normal vector, is λ. so I no longer specify the margin $||m||$, instead I have introduced a new user designed parameter λ, given the parameter λ, I can solve this problem, corresponding to overfitting, term. so this would be a tendency to overfit and as you have seen it increases with decreasing "m" margin bcz $||w|| \leq \frac{1}{m}$, is it is something that increses with $||w||$. And what people mostly use is the quadratic squared euclidian length of the normal vector

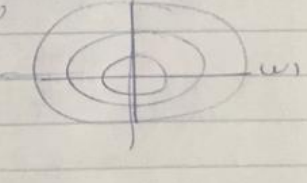
$$\sigma(w) = w^Tw$$

→ So, for the best model, we were trying to find a model that minimizes $\underset{(w,b)}{\arg\min} \sum_{i=1}^{N}$

that depended on w & b. but we know, bcz of over fitting thing, we have to add an extra term $\lambda \cdot \sigma(w)$ to get the most/best classifier

so, important choice is the one that you see here, so this would be the squared l_2 - Norm or another choice would be to just take the L1 norm of w

so, $\sigma(w) = w^Tw$



So, this is Lab-2 ... to do da... collected ... article. So first ... different ... theory ... → Atten... Transf...

so, we will consider, $\sigma(w) = \underset{\downarrow}{w^T w}$ or $\|w\|$ (2)

choosing L2 regularized term

we will derive the optimization equation

-> so the tendency to overfit is the regularization term whereas the other thing here the slack variable (ξ) is the data term

given λ

$$\underset{w,b,\xi}{\operatorname{argmin}} \; \underbrace{\lambda \cdot \sigma(w)}_{\text{regularization tendency to overfit}} + \underbrace{\sum_{i=1}^{n} \xi_i}_{\text{data term}} \quad s.t. \; y_i(w^T x_i + b) \geq 1 - \xi_i \quad \xi_i = 0 \quad i = 1..N \quad \rightarrow (A)$$

- so we have this formula, we have there terms & constraints, we now need to make a specific choice as to the nature of this regularization term, we use L_2 regularization $\sigma(w) = w^T w$ → (B)

so the formula (A) becomes using (B)

$$\underset{w,b,\xi}{\operatorname{argmin}} \; \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i \quad s.t. \quad y_i(w^T x_i + b) \geq 1 - \xi_i; \quad \xi_i \geq 0$$

-> so this formula is in primal domain, now with optimization problems, such ~~the~~ as these, we here have a quadratic program (quadratic objective) function & linear constraints. It would be perfectly fine to optimize it here, but sometimes, we need to go go dual domain for solving certain questions!

-> So, since there are inequalities the constraints, so instead of using the Lagrangian formalism. We will use KKT equations or

So, this is a v
Lab-2 projec
to do data
collected from
article. -> S
S. firstly
different
through
-> Above
Train

So, this is a video
Lab-2 project, whic
to do data analy
collected from 2
article. -- So f
So, firstly
different
through

→ After
found

→ So
di

also called "Lagrangian Inequalities"

-- So, I am setting up a Lagrangian function here which involves both primal variables $(w, b \& \xi)$ and new dual variables the lagrange multiplier $\alpha \& u$.

so now the equation becomes

penalization of the slack variable

$$L(w, b, \xi, \alpha, u) = \frac{1}{2} w^T w + C \sum_{i=1}^{n} \xi_i - \sum_{i=1}^{N} \alpha_i (y_i (w^T x_i + b) - 1 + \xi_i) - \sum_{i=1}^{N} u_i \xi_i$$

regularization term → ① ↳

↳ and these other terms came from ~~from~~ from the constraints of inequalities, so what I have done here is that since I have 1 equation for each observation here and I'm now bringing what is on the RHS to the LHS so I have $y_i (w^T x + b) - 1 + \xi_i$ and I am multiplying this with the Lagrange multiplier α_i, and now since I have equation ① summed over that many Lagrange multipliers α_i's, here

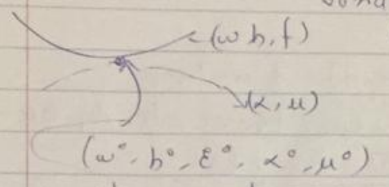
so

$$\sum_{i=1} \alpha_i (y_i (w^T x_i + b) - 1 + \xi_i) - \sum_i u_i \xi_i$$

↳ similarly we have u_i for this

-- so, what it says is that there KKT, we want to look for an optimum, it has to fulfill the following properties, the Lagrangian or this equation here must ~~have~~ be at a minimum w.r.t to the primal variables w, b, & ξ and it must be at a maximum with respect to also an extremum but at a maximum w.r.t to the dual variables and at this saddle point it is minimum w.r.t to a primal maximum w.r.t to dual variables

- optimum given by saddle point : minimum w.r.t. primal variable & maximum w.r.t dual variables



& $\alpha_i, \mu_i \geq 0$

$(w^\circ, b^\circ, \xi^\circ, \alpha^\circ, \mu^\circ)$

- this is where we have an optimum; I can idicate the optimum parameters by the superscript o, there lagrange multipliers have to be non -ve. ie. $\alpha_i, \mu_i \geq 0$, we can go & find the extremum by differenting this Lagrangian w.r.t its primal variables, so we differentiate w.r.t to the normal vector (w), bias (b) & slack varibale (ξ) & we set the derivative to 0

ning (1)

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

↳ (A)

$$\frac{\partial L}{\partial b} = -\sum_i \alpha_i y_i = 0 \qquad = \sum_{i=1}^{N} \alpha_i y_i = 0 \text{ -- (2)}$$

$$\frac{\partial L}{\partial \xi_i} = c - \alpha_i - \mu_i = 0 \qquad = \alpha_i + \mu_i = c \text{ -- (3)}$$

so, the margin is w∈ $w = \sum_{i=1}^{n} \alpha_i y_i x_i$

- Now I'm starting with the first term going . & we will see that that the sum here is going to be sparse, so I don't need all training examples to express my ~~natu~~ normal vector & this is one of the great things about SVM so, we have found the conditions that describe how we can find a minimum w.r.t the primal variables what remains is to maximize the wolfe dual w.r.t the dual variables.

so

$$\underset{\alpha, u}{\operatorname{argmax}} \; \underbrace{L(w^{o}, b^{o}, \xi^{o}}, \alpha, u) = \frac{1}{2} w_o^T w_o$$

optimal values have already been put using equations Ⓐ, ② & ③

so using ①, Ⓐ, ② & ③

we get

$$L(w^{o}, b^{o}, \xi^{o}, \alpha^{o}, u^{o}) = \frac{1}{2} w^T w + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \alpha_i (y_i (w^T x_i + b) - 1 + \xi_i) - \sum_{i=1}^{N} u_i \xi_i$$

solving & substituting the value of from Ⓐ, ② & ③, so

$$\underset{\alpha, u}{\operatorname{argmax}} \; L(w^{o}, b^{o}, \xi^{o}, \alpha, \mu) = \frac{1}{2} w_o^T w_o + \sum_i \xi_i (\alpha_i + u_i) - w_o^T w^{o} - \underbrace{\sum_{i=1}^{N} \alpha_i (y_i b - 1 - \xi_i)}_{} - \sum_{i=1}^{N} u_i \xi_i$$

we still have the term here that dependents i.e. b (④)

$$\underset{\alpha, u}{\operatorname{argmax}} \; L(w^{o}, b^{o}, \xi^{o}, \alpha, u) = \frac{1}{2} w^{oT} w^{o} + \sum_i \alpha_i$$ such $\alpha_i \geq 0$, $u_i \geq 0$

that this term here is quadratic in w & the sum over α_i from the summing one here & this is subject to the equations that we had here, and to the dual variables being non-ve

now using Ⓐ, ② & ③ where we have replace w by $\sum_{i=1}^{N} \alpha_i x_i y_i$

$= \underset{\alpha, u}{\operatorname{argmax}} -\frac{1}{2} \sum_i \sum_j y_i y_j \alpha_i \alpha_j x_i^T x_j + \sum_i \alpha_i$ ④

such that

$\alpha_i \geq 0$

$u_i \geq 0$

& $\alpha_i + u_i = C \left\{ \begin{array}{l} 0 \leq \alpha_i \leq C \\ \sum \alpha_i y_i = 0 \end{array} \right.$

- These along with equation ② & ③ say that the lagrange multiplier α_i, they anyway have to be bounded below by 0, but they must also not becomes larger than C, which was the user-defined constraint that we put into trade off the regularization versus the data term.

- So this is the optimization problem that we need to solve

i.e.

$$d(\alpha, u, w^\circ, b^\circ, \xi^\circ) = \underset{\alpha, u}{\operatorname{argmax}} -\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \overbrace{x_i^T x_j}^{K(x_i, x_j)} + \sum_i \alpha_i$$

such that

$\alpha_i \geq 0$ $\quad \alpha_i + u_i = C \}$

$u_i \geq 0$ $\quad 0 \leq \alpha_i \leq C$

$\sum_{i=1}^{n} \alpha_i y_i = 0$

- so, just like the primal, here also, we have a product of 2 dual variables α_i & α_j but we convert this whole thing in matrix form as follows:- so rather than maximizing, we may wish to minimize to put this to a standard form for a quadratic programming problem, then we have to invert the sign of and we had an inner product between two vectors x_i & x_j. I am inserting these inner products. I am collecting all of these inner products in a kernel matrix, so the element i,j of this kernel matrix is going to be x_i transpose time x_j which is written here, then I have these vectes α summarizing all of my langarium multipliers and one more term in there, we have a missing term here, the sum over the α can be written as 1 transpose alpha, when there 1 is vecto of all 1's

$$= \underbrace{\arg\min_{\alpha}}_{\text{standard QP problem}} + \frac{1}{2}\alpha^T \underline{Y}\, \underset{y}{\bar{K}}\, \bar{Y}\alpha + \bar{1}^T\alpha \qquad \text{such that } 0 \le \alpha_i \le C,\ \alpha y = 0$$

kernel matrix $(K)_{ij} = x_i^T x_j$

diagonal matrix holding labels

→ one thing to note there than I have not mentioned in v'do, I can collect here in a diagonal matrix, i.e. holding the n labels, which must either be +1 or -1. along with the diagonals.

→ so this is something that you can put into any standard QP solver, specifically. here I have underlined in gree the inputs that you need to provide, so, this is what you want to solve for but Y's the labels are given in the training set, the kernel matrix you can compute using this recipe and c is a constant supplied by the user, to specify how strongly do we want to regularize → how closely we want to fit the data, ok - so this you can really put into it's over and once you have done that you got out a collection of α's and now how do you relate them to the solution in the primal domain, you can again remember that (a normal vector w, the simply sum over i α, $w = \sum_{i=1}^{N} \alpha_i x_i y_i$.

So, this is a
Lab-2 project
to do stable
collected from
article. - S
S. firstly
different
through
→ Also
Turn

after optimizing for the alphas here, here you can s insert what you found & then got the normal vector to the primal domain. Now I mentioned earlier that perhaps, we don't need to involve all of the observations indeed at this is important for SVM, & the reason is given by KTT called complementary slackness condition which states that the product of Lagrange multiplier and the equality – inequality here to enforce the product of these two has to be zero, now for this product to be zero

$$\alpha_i (y_i (w^T x_i + b) - 1 - \xi_i) = 0$$

2) part2) Point out what is the "margin" in both the primal formulation and the dual formulation,

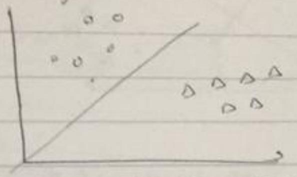
Soln: As suggested in the above derivation

So, we can say $w^Tx+b>0$ $y=+1$ (y is +ve)

If $w^Tx+b<0$ $y=1$ (y is -ve)

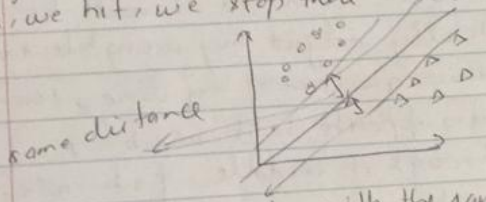
$w^Tx+b=0$ (can by anything

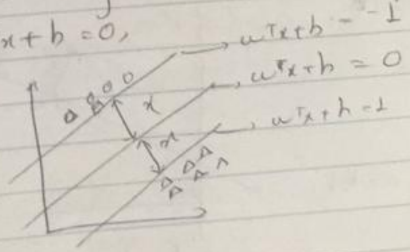
Idea of max^m margin



→ 2 types of points are there ooo & △△

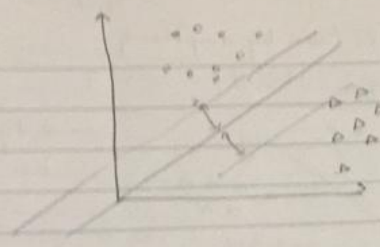
How to define margin ? → First we define the slope, let's say we have the slope, what we do is the first we draw a line with that slope going through origin then we start moving it in either direction, we will move it up little bit ||el & whenever, we hit a point on either side, maybe we will hit a circle first or a △gle first, may be we will hit both at the same time,, but which ever, we hit, we stop there , this is 1 margin, bcz after this point I will start making mistakes & then I draw a symmetri line over the other side with the same distance, then the line i.e on the middle of it (need not go through the origin) will be my decision boundary, so, this will be called $w^Tx+b=0$,





But what...

So, I am defining these as my boundaries, so now I want to find w & b such that this whole distance is maximized

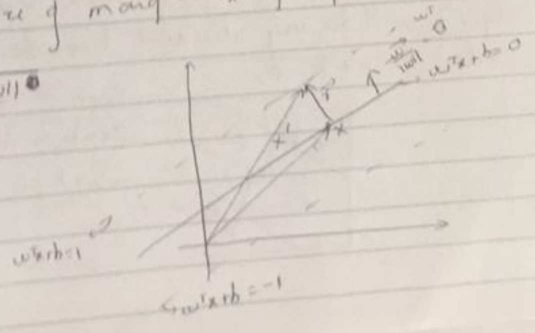


Let's say, that again, let's try to fit w, any ~~w~~ w, then I start going up & down, so with some increment I keep going up & say hey at this point, I hit a +ve point so, this will be my one side margin, then I am going to rescale my w & b, so that the equation of the line ~~is~~ like this $w^Tx+b=c1$ something

→ Remember, if $w^Tx+b=1$ is a line, then you can always multiply it with same constant it will be the same line, so. I basically adjust my weights such that this $w^Tx+b=1$ is the equation of my line, then I look at a line which is opposite to it. $w^Tx+b=-1$, & then the line that goes through it middle of it $w^Tx+b=0$ will be my decision bound.

So, the length/size of margin is going to be

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



in that, 6

$\vec{\gamma} = \gamma \underbrace{\frac{\vec{w}}{\|w\|}}_{magnitude}$ $\hat{u} = \frac{\vec{W}}{\|w\|}$

so $\vec{\gamma}' = \vec{x}' - \vec{x}^0$

$\vec{w}^T\vec{\gamma}' = w^T x' - w^T x^0$

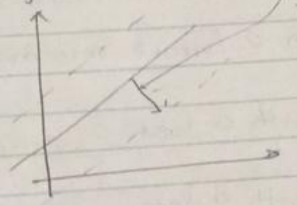
$= \hat{w}^T \gamma \frac{\vec{w}}{\|w\|} = ~~b-e(1-eb)~~ (1-b) - ~~b~~(-b)$

$= \gamma \frac{w^T w}{\|w\|} = +1 \Big]$ $\gamma \|w\| = +1$

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

so, now, when you derive the same quantity for this side

this length = $\frac{1}{\gamma}$



size of the margin = $2\gamma = \frac{2}{\|w\|}$

Now, the task has become like this, we want to find a line $\vec{w}$ & b, which not only separates the 2 types of points

So, we want to learn w, b such that for all points for which $y_i = +1$, $w^T x_i + b > 0$ & for all points x_i for which $y_i = -1$ } zero training err
$w^T x_i + b < 0$

and $\frac{2}{\|w\|}$ is as large as possible / - support vector machine

$y_i(w_i^T x_i + b) > 0$
↳ for this we can write $y_i(w_i^T x_i+b)>0$ to be
bcz if y_i is -ve then we want $(w_i^T x_i + b)$

-ve as well.

How do we learn this w & b?

→ We are going to assume that the data is linearly separable, that is, we accomodate no loss.

in past

An How are we going to learn w&b? → either we try to minimize some loss or maximize some likelihood, here we are going to pose it in a way: we are going to maximize the margin i.e. $\frac{2}{||w||}$ → minimize $||w||$

Optimization: maximize margin while incurring zero training error.

Max $\frac{2}{||w||}$ with 0 loss. → minimize $\frac{||w||}{2}$ with 0 loss.

$\frac{\sqrt{w_1^2+w_2^2}}{2}$

→ Minimize $\frac{||W||_2^2}{2}$ with 0 loss → $\frac{w_1^2+w_2^2}{2}$

Minimize $\frac{W^Tw}{2}$ with 0 loss.

→ This is what the optimization criteria is.

i.e. $\underset{w,b}{\text{minimize}} \; \frac{||w||^2}{2}$

subject to $y_n(w^Tx_n+b) \geq 1$, $n=1,\ldots,N$ (for all the training data points, this should be always ≥ 1

→ so we not only want the data points to be on right side, correct side of the margin, i.e. if it is +ve we want it be on +ve side and if it is -ve, on the -ve side. but we actually want it be to on the other side of margin. We don't want any points inside the margin at all, so that is determined by

$(y_n)(w^Tx_n+b) \geq \boxed{1}$ this

bcz, if this were 0, all I am saying is that, try to find a line such that, maximizes the margin & points are on the correct side of the decision boundary,

So, this is a constrainted optimization, subject to certain contraint.

so, Optimization with N linear inequality constraint
↓ no. of data points

→ So, large margin = small |w|
→ small |w| ⇒ regularized/simple solutions
→ simple solution ⇒ Better generalizability

→ so $\underset{w,b}{\text{minimize}} \ \frac{\|w\|^2}{2}$
subject to $y_n(w^T x_n + b) \geq 1, \ n = 1 \ldots N$

→ so, this is an quadratic objective function to minimized & N constraints

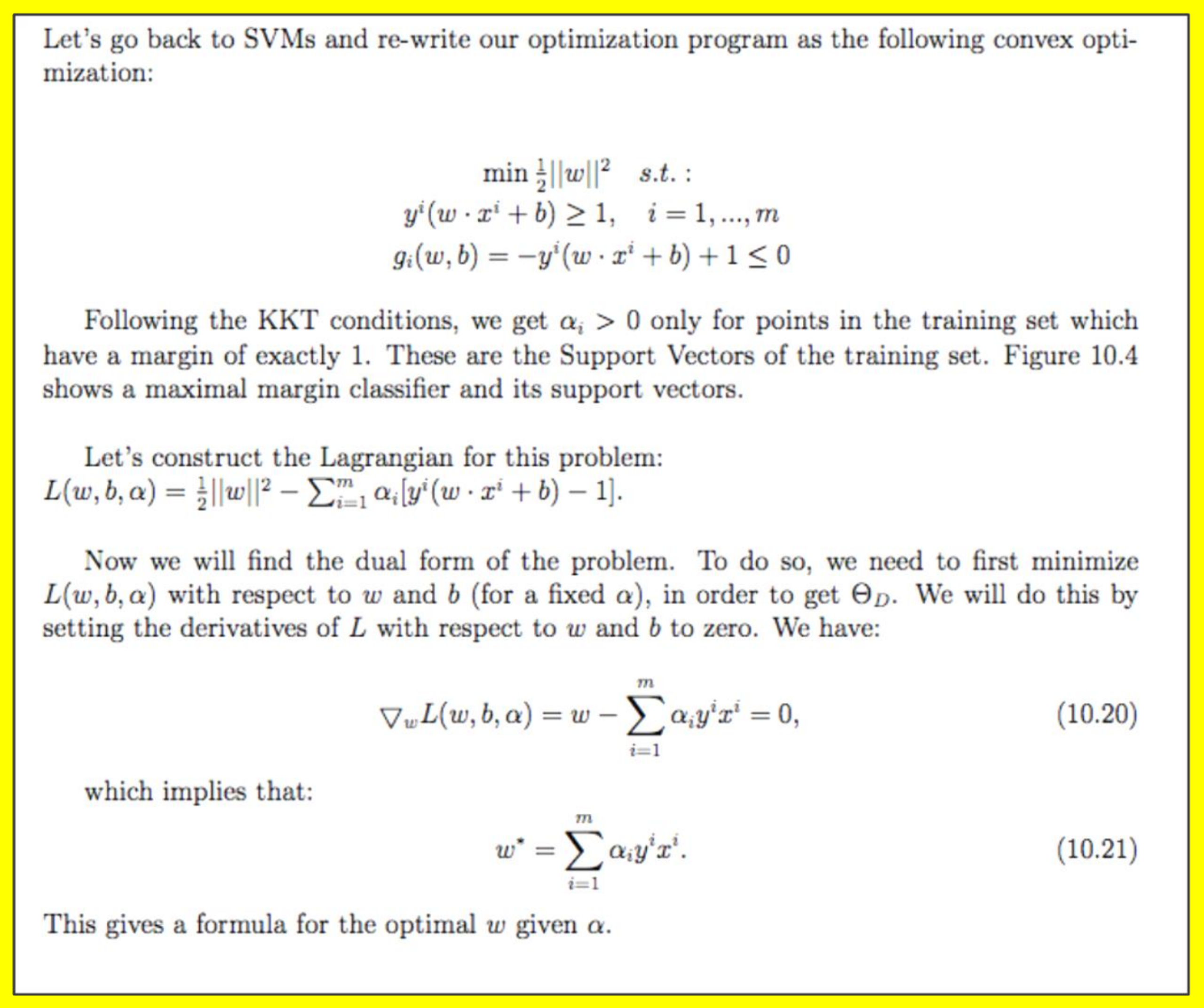
Basic Optimization
$\underset{x,y}{\text{minimize}} \ f(x,y) = 2 - x^2 - 2y^2$
subject to $h(x,y) = x + y - 1 = 0$ } one in equality constraint

How to do it?
An so, there is a notion of Lagrangian multiplier (β), that lets you combine the two equation in L.

so, what you do is that you define a new objective function
which $\underset{x,y,\beta}{\text{minimize}} \ L(x,y,\beta) = f(x,y) - \beta g(x,y)$

→ It has been shown that, if you find a solution to this

∂y $\frac{\partial L}{\partial \beta} = x + y - 1 = 0$



Let's go back to SVMs and re-write our optimization program as the following convex optimization:

$$\min \frac{1}{2}||w||^2 \quad s.t.:$$
$$y^i(w \cdot x^i + b) \geq 1, \quad i = 1, ..., m$$
$$g_i(w, b) = -y^i(w \cdot x^i + b) + 1 \leq 0$$

Following the KKT conditions, we get $\alpha_i > 0$ only for points in the training set which have a margin of exactly 1. These are the Support Vectors of the training set. Figure 10.4 shows a maximal margin classifier and its support vectors.

Let's construct the Lagrangian for this problem:
$L(w, b, \alpha) = \frac{1}{2}||w||^2 - \sum_{i=1}^{m} \alpha_i[y^i(w \cdot x^i + b) - 1]$.

Now we will find the dual form of the problem. To do so, we need to first minimize $L(w, b, \alpha)$ with respect to w and b (for a fixed α), in order to get Θ_D. We will do this by setting the derivatives of L with respect to w and b to zero. We have:

$$\nabla_w L(w, b, \alpha) = w - \sum_{i=1}^{m} \alpha_i y^i x^i = 0, \qquad (10.20)$$

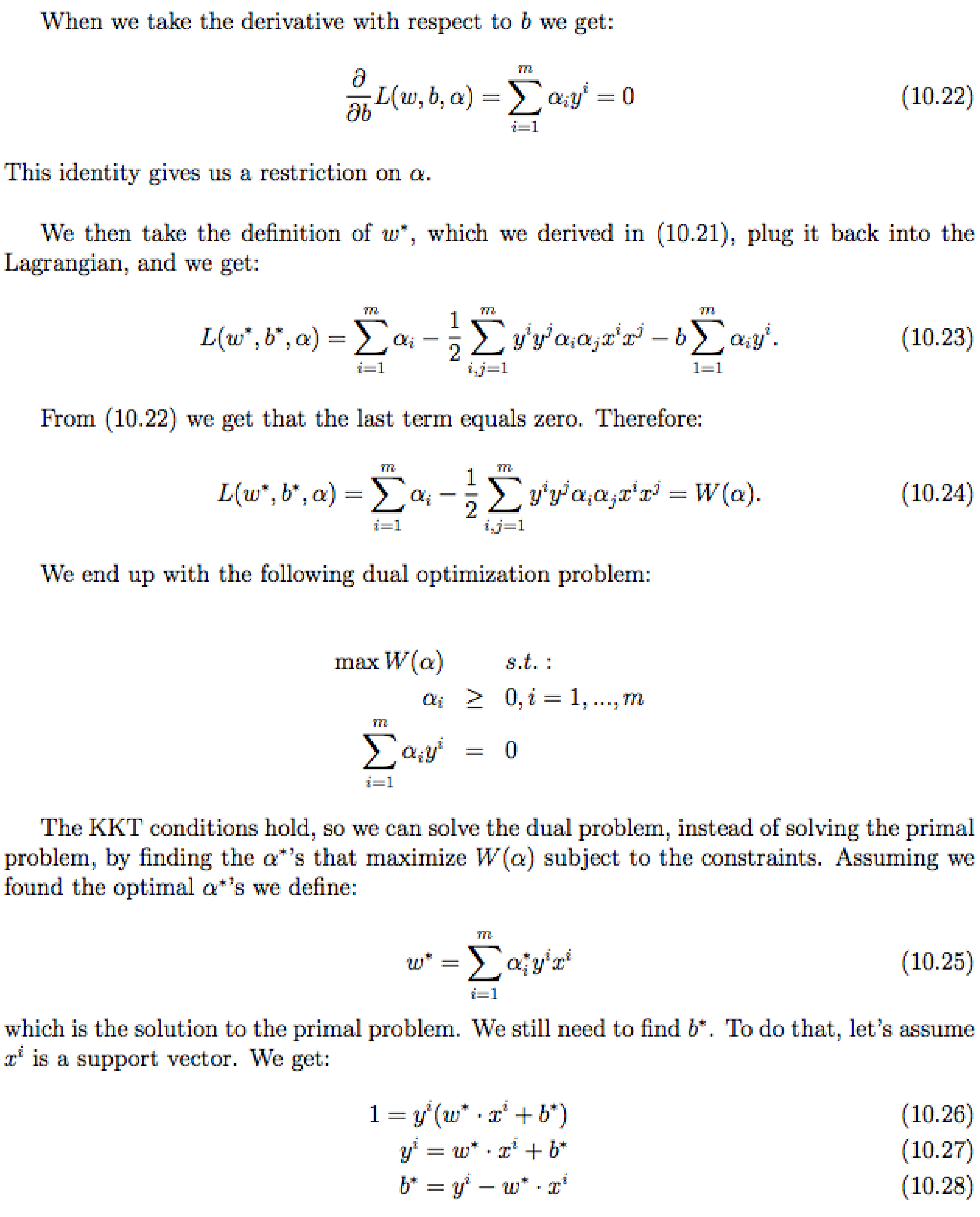
which implies that:

$$w^* = \sum_{i=1}^{m} \alpha_i y^i x^i. \qquad (10.21)$$

This gives a formula for the optimal w given α.

which can be found out by finding w* is the optimum w here because w*=summation of alpha * y* x (all three quantities going from 1 to N). So given alpha we can find the w

. and we know that margin is **2/|w| .**



When we take the derivative with respect to b we get:

$$\frac{\partial}{\partial b} L(w,b,\alpha) = \sum_{i=1}^{m} \alpha_i y^i = 0 \tag{10.22}$$

This identity gives us a restriction on α.

We then take the definition of w^*, which we derived in (10.21), plug it back into the Lagrangian, and we get:

$$L(w^*,b^*,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^i y^j \alpha_i \alpha_j x^i x^j - b \sum_{1=1}^{m} \alpha_i y^i. \tag{10.23}$$

From (10.22) we get that the last term equals zero. Therefore:

$$L(w^*,b^*,\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y^i y^j \alpha_i \alpha_j x^i x^j = W(\alpha). \tag{10.24}$$

We end up with the following dual optimization problem:

$$\begin{aligned} \max W(\alpha) & \quad s.t. : \\ \alpha_i & \geq 0, i = 1, ..., m \\ \sum_{i=1}^{m} \alpha_i y^i & = 0 \end{aligned}$$

The KKT conditions hold, so we can solve the dual problem, instead of solving the primal problem, by finding the α^*'s that maximize $W(\alpha)$ subject to the constraints. Assuming we found the optimal α^*'s we define:

$$w^* = \sum_{i=1}^{m} \alpha_i^* y^i x^i \tag{10.25}$$

which is the solution to the primal problem. We still need to find b^*. To do that, let's assume x^i is a support vector. We get:

$$1 = y^i(w^* \cdot x^i + b^*) \tag{10.26}$$

$$y^i = w^* \cdot x^i + b^* \tag{10.27}$$

$$b^* = y^i - w^* \cdot x^i \tag{10.28}$$

2)Part3 what are the benefits of maximizing the margin.

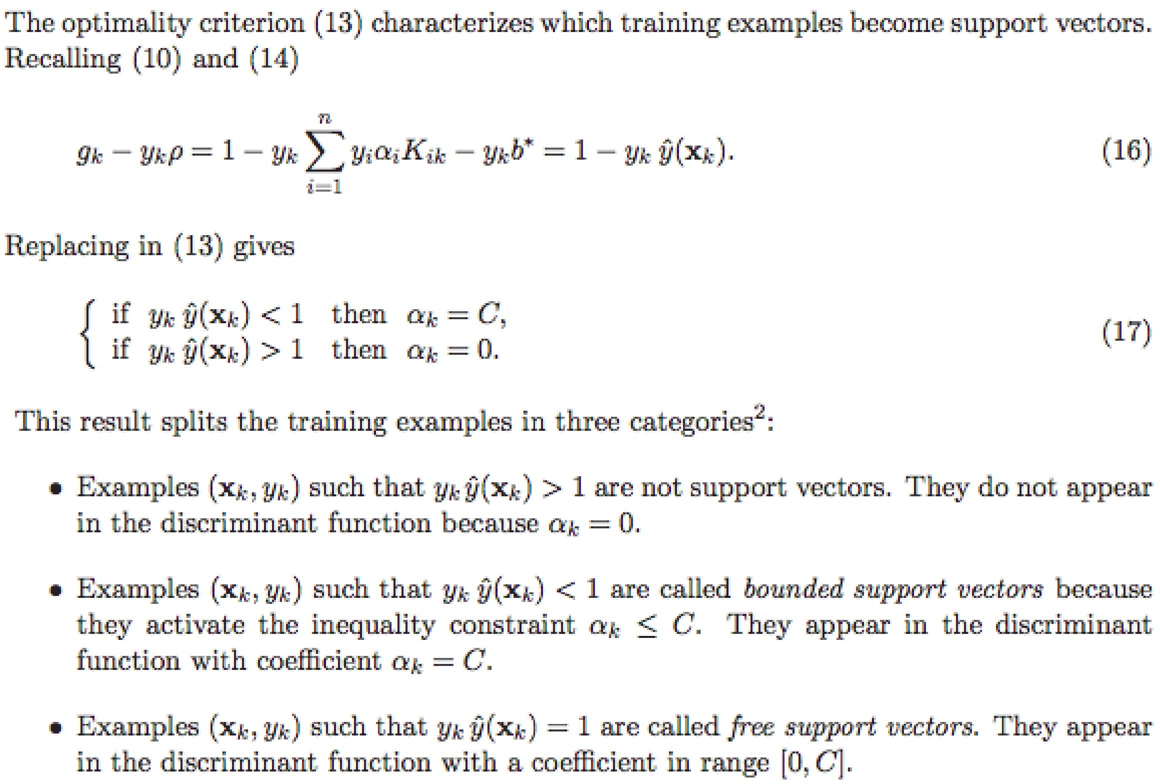
Ans: Because of following reasons:

1)A large margin effectively corresponds to a regularization of SVM weights which prevents overfitting. Hence, we prefer a large margin (or the right margin chosen by cross-validation) because it helps us generalize our predictions and perform better on the test data by not overfitting the model to the training data.

2) “Maximizing the margin seems good because points near the decision surface represent very uncertain classification decisions: there is almost a 50% chance of the classifier deciding either way. A classifier with a large margin makes no low certainty classification decisions. This gives you a classification safety margin: a slight error in measurement or a slight document variation will not cause a mis-classification.”

3) SVM maximizes margin, so the model is slightly more robust (compared to linear regression) but more importantly: SVM supports kernels, so you can model even non-linear relations

2)Part4 Characterize the support vectors.
Soln:



The optimality criterion (13) characterizes which training examples become support vectors. Recalling (10) and (14)

$$g_k - y_k\rho = 1 - y_k \sum_{i=1}^{n} y_i \alpha_i K_{ik} - y_k b^* = 1 - y_k\, \hat{y}(\mathbf{x}_k). \tag{16}$$

Replacing in (13) gives

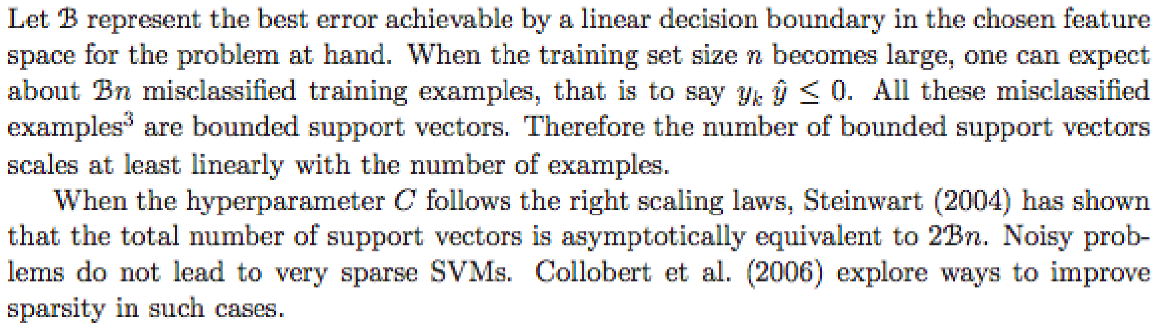
$$\begin{cases} \text{if} \quad y_k\, \hat{y}(\mathbf{x}_k) < 1 \quad \text{then} \quad \alpha_k = C, \\ \text{if} \quad y_k\, \hat{y}(\mathbf{x}_k) > 1 \quad \text{then} \quad \alpha_k = 0. \end{cases} \tag{17}$$

This result splits the training examples in three categories^2:

- Examples $(\mathbf{x}_k, y_k)$ such that $y_k\, \hat{y}(\mathbf{x}_k) > 1$ are not support vectors. They do not appear in the discriminant function because $\alpha_k = 0$.

- Examples $(\mathbf{x}_k, y_k)$ such that $y_k\, \hat{y}(\mathbf{x}_k) < 1$ are called *bounded support vectors* because they activate the inequality constraint $\alpha_k \leq C$. They appear in the discriminant function with coefficient $\alpha_k = C$.

- Examples $(\mathbf{x}_k, y_k)$ such that $y_k\, \hat{y}(\mathbf{x}_k) = 1$ are called *free support vectors*. They appear in the discriminant function with a coefficient in range $[0, C]$.



Let $\mathcal{B}$ represent the best error achievable by a linear decision boundary in the chosen feature space for the problem at hand. When the training set size n becomes large, one can expect about $\mathcal{B}n$ misclassified training examples, that is to say $y_k\,\hat{y} \leq 0$. All these misclassified examples[3] are bounded support vectors. Therefore the number of bounded support vectors scales at least linearly with the number of examples.

When the hyperparameter C follows the right scaling laws, Steinwart (2004) has shown that the total number of support vectors is asymptotically equivalent to $2\mathcal{B}n$. Noisy problems do not lead to very sparse SVMs. Collobert et al. (2006) explore ways to improve sparsity in such cases.

This is also explained below as part of solution of the above mentioned question:

i.e.

$$\underset{\alpha,\mu}{\operatorname{argmax}} -\frac{1}{2}\sum_i\sum_j y_i y_j \alpha_i \alpha_j \overbrace{x_i^T x_j}^{k(x_i,x_j)} + \sum_i \alpha_i \quad \text{such that}$$

$$\left.\begin{array}{l} \alpha_i \geq 0 \\ \mu_i \geq 0 \end{array}\right. \quad \alpha_i + \mu_i = C \left.\right\} 0 \leq \alpha_i \leq C$$

$$\sum_i \alpha_i y_i = 0$$

all this equation is a bit redundant here and in fact this can be written even more succinctly as follows to bring out even more clearly the quadratic structure we have. so this is also a quadratic objective just like, we had in the primal here, we have a product of 2 dual variables α_i & α_j ~~and~~ but we can write this whole thing in matrix form as follows:

so, rather than maximizing we may wish to minimize to put this to standard form for a quadratic programming problem then we have to invert the sign of course and now here we had an inner product between two vectors x_i T & x_j, I am ~~interesting~~ these inner products I'm collecting, all of these inner products in a kernel matrix so the element i,j of this kernel matrix is going to be x_i transpose time x_j which is written here, then I have these vector α summarizing all of my lagrangian multipliers and one more term is there. we have a missing term here the sum over the α can be written as 1 transpose alpha, where, this is a vector of all 1's

$$\underset{\alpha}{\operatorname{argmin}} + \frac{1}{2}\alpha^T \underbrace{\underline{\bar{Y}}\,\underline{\bar{K}}\,\underline{\bar{Y}}}\alpha + \bar{1}^T\alpha \quad \text{s.t. } 0 \le \alpha_i \le \underline{C}, \quad \alpha \cdot y = 0$$

standard QP problem — kernel matrix $(K)_{ij} = x_i^T x_j$ — diagonal matrix holding labels

- one thing that I have not mentioned is y's, I can collect these in a diagonal matrix that is holding the m labels, which must be +1 or -1 along with its diagonals
- so this is something that you can put into any standard QP solver, specifically, here I have underlined in green the inputs that you need to provide so. this is what you want to solve for but y's the labels are given in the training set, the kernel matrix you can compute using this receipe and C is a constant supplied by the user to specify how strongly do we want to regularize or how closely do we want to fit the data. So so this you can really put into it's ~~[illegible]~~ over and once you have done that you get out a collection α's and now how do you relate them to the solution in the primal domain, if you can again remember that for normal vector w, we simply sum over i alpha $w^* = \sum_i \alpha_i y_i x_i$ after optimizing for the alphas here, you can insert what you found. & thus get the normal vector in the primal domain. Now I mentioned earlier that

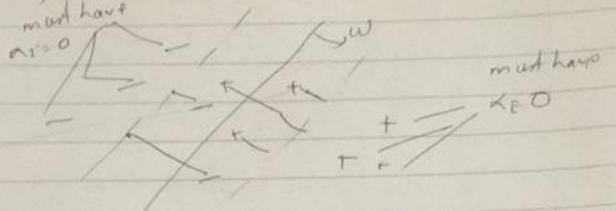
perhaps, we don't need to involve all of the observations, indeed this is one of the interesting perhaps of SVM and the reason is given by KKT condition. called complementary slackness.) condition which states that the product of Lagrange multiplier and the equality or inequality lies to enforce the, product of these two has to be zero in order for this product to be zero .

$$\alpha_i \underbrace{(y_i(w^T x_i + b) - 1 - \xi_i)}_{\text{larger than 'two' for all training points on correct side of their margin}} = 0$$

- obviously one of the two factors has to be zero and let's look at these in turn this is lagrange m ultiplier ; α_i on the other hand. (lagrangrone multiplier) $y_i(w^T x_i + b)$ = was the score to how much I was lying on the correct or wrong side of the decision boundary, there is minus 1, the margin that I want to subtract and if a point lies on the right side of the margin then this thing here cannot be 0; ξ_i, (here, then there will be no slack), so if slack is required and remember that slack appeared in the primal objective function of the program was explicitly asked to minimize . If, so thing will be $\xi_i = 0$ & the term $(y_i(w^T x_i + b) - 1)$ will be some term

larger than 0

this is exactly what makes for sparsity what this very interesting property of SVM, so, the slack of a couple of variables, I have indicated here



this would be the amount of slack or the distance that would has from the correct side of margin ~~+~~ that +ve positive point has from it's side of the margin

Imp
- My normal vecto w, will now be specified as a linear combination of only of the support vectors and not of these other points out here.

$$K_{i,j} = x_i^T x_j$$

- for matrix K is a measure of similarity for observations i & j

-> can use other similarity measures in particular for non vectorial data you will want to use that own similarity measures the only thing that they have to satisfy is that this matrix K should remain +ve semi-definite bcz as long as it is we have a convex optimization problem and a unique optimum if this matrix is not +ve semi-definite then optimization would much, much harder, K

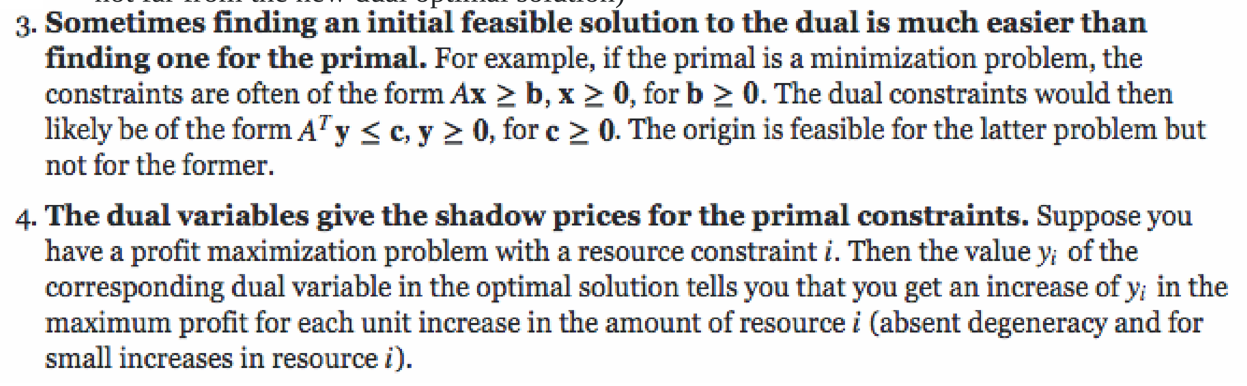
2)Part5

Point out the benefit of solving the dual problem instead of the primal problem.

Soln: Below is the answer:

1) **Understanding the dual problem leads to specialized algorithms for some important classes of linear programming problems.** Examples include the transportation simplex method, the Hungarian algorithm for the assignment problem, and the network simplex method. Even column generation relies partly on duality.

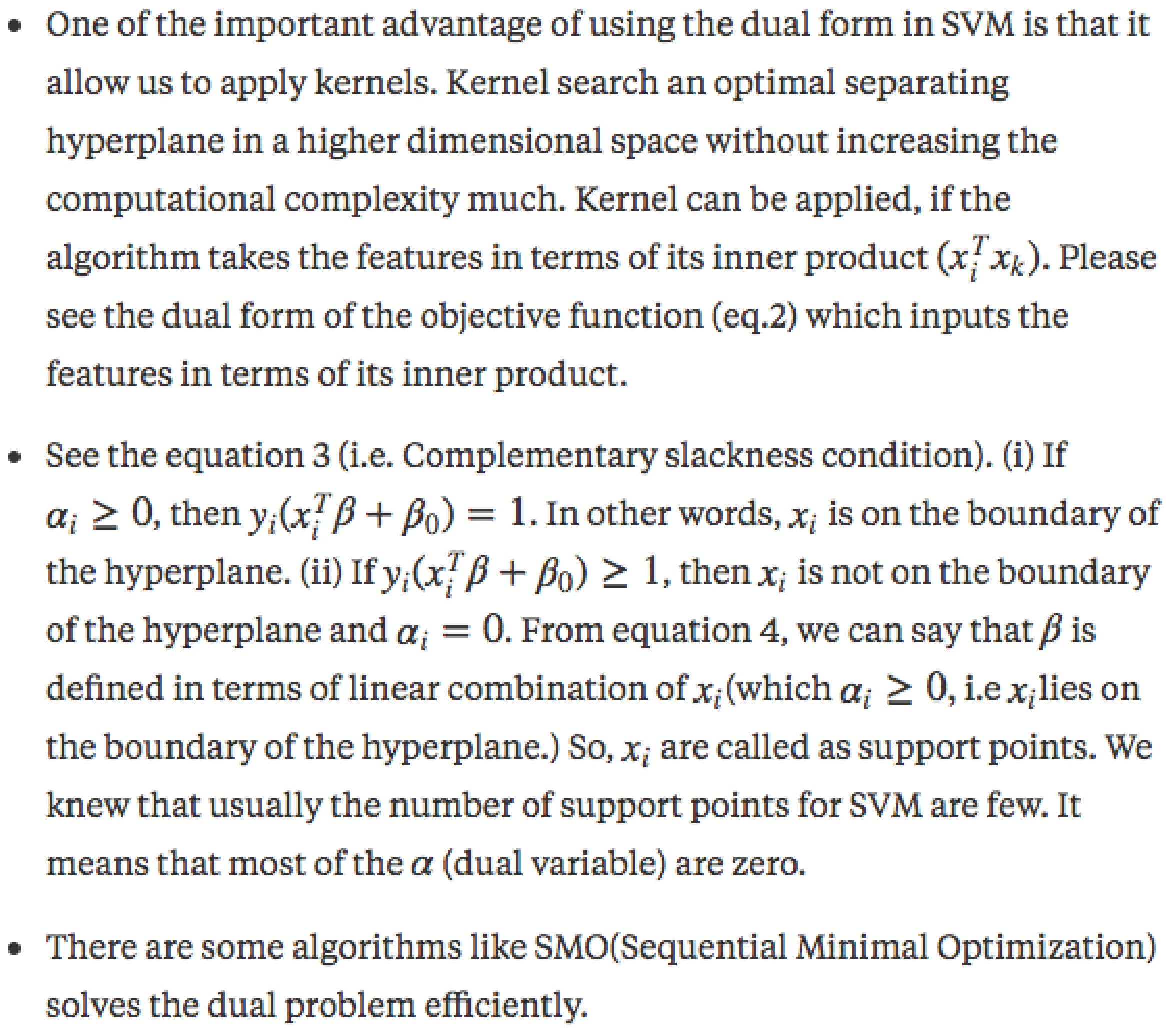
2) **The dual can be helpful for sensitivity analysis.** Changing the primal's right-hand side constraint vector or adding a new constraint to it can make the original primal optimal solution infeasible. However, this only changes the objective function or adds a new variable to the dual, respectively, so the original dual optimal solution is still feasible (and is usually not far from the new dual optimal solution)



3. **Sometimes finding an initial feasible solution to the dual is much easier than finding one for the primal.** For example, if the primal is a minimization problem, the constraints are often of the form $A\mathbf{x} \geq \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, for $\mathbf{b} \geq \mathbf{0}$. The dual constraints would then likely be of the form $A^T\mathbf{y} \leq \mathbf{c}$, $\mathbf{y} \geq \mathbf{0}$, for $\mathbf{c} \geq \mathbf{0}$. The origin is feasible for the latter problem but not for the former.

4. **The dual variables give the shadow prices for the primal constraints.** Suppose you have a profit maximization problem with a resource constraint i. Then the value y_i of the corresponding dual variable in the optimal solution tells you that you get an increase of y_i in the maximum profit for each unit increase in the amount of resource i (absent degeneracy and for small increases in resource i).

5)

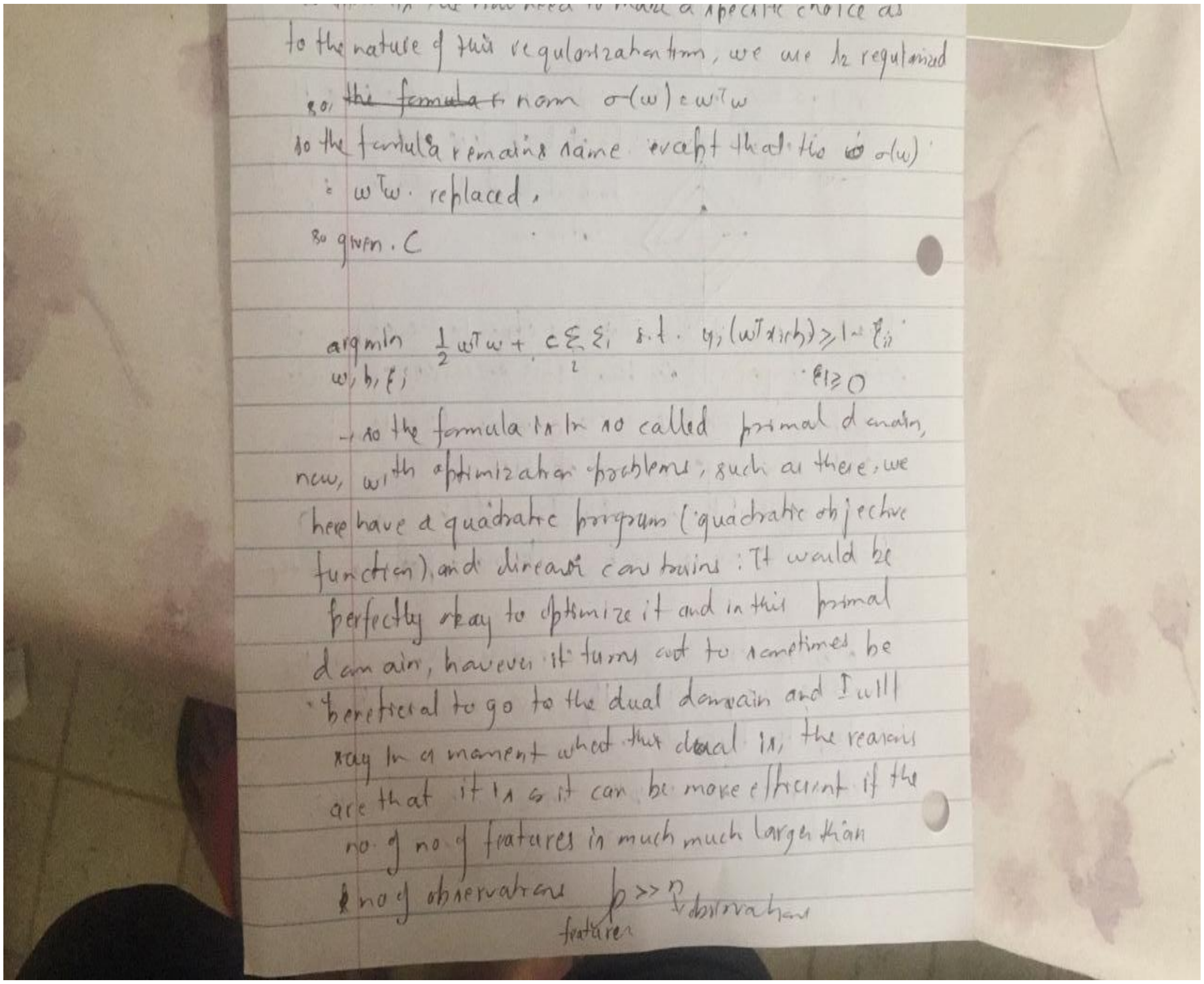


- One of the important advantage of using the dual form in SVM is that it allow us to apply kernels. Kernel search an optimal separating hyperplane in a higher dimensional space without increasing the computational complexity much. Kernel can be applied, if the algorithm takes the features in terms of its inner product $(x_i^T x_k)$. Please see the dual form of the objective function (eq.2) which inputs the features in terms of its inner product.

- See the equation 3 (i.e. Complementary slackness condition). (i) If $\alpha_i \geq 0$, then $y_i(x_i^T \beta + \beta_0) = 1$. In other words, x_i is on the boundary of the hyperplane. (ii) If $y_i(x_i^T \beta + \beta_0) \geq 1$, then x_i is not on the boundary of the hyperplane and $\alpha_i = 0$. From equation 4, we can say that β is defined in terms of linear combination of x_i(which $\alpha_i \geq 0$, i.e $x_i$lies on the boundary of the hyperplane.) So, x_i are called as support points. We knew that usually the number of support points for SVM are few. It means that most of the α (dual variable) are zero.

- There are some algorithms like SMO(Sequential Minimal Optimization) solves the dual problem efficiently.

A proper explanation is below:



is we now need to make a specific choice as to the nature of this regularization term, we use l_2 regularized

so, ~~the formula~~ ~~i~~ norm $\sigma(w) = w^T w$

so the formula remains same except that the $\sigma(w)$: $w^T w$ replaced.

so given C

$$\underset{w,b,\xi_i}{\arg\min} \; \frac{1}{2} w^T w + C \sum_i \xi_i \quad s.t. \quad y_i(w^T x_i + b) \geq 1 - \xi_i$$

$$\xi_i \geq 0$$

→ so the formula is in so called primal domain, now, with optimization problems, such as these, we here have a quadratic program (quadratic objective function) and linear constrains: It would be perfectly okay to optimize it and in this primal domain, however it turns out to sometimes be beneficial to go to the dual domain and I will say in a moment what this dual is, the reasons are that it is it can be more efficient if the no. of no. of features is much much larger than & no of observations $\underset{\text{features}}{p} >> \underset{\text{observations}}{n}$

So, if we have 10,000 features, then each of these inner products here and involving the normal vector

$y_i(w^Tx_i+b)$

will always involve, well 10,000 numbers and even after training, if you want to make predictions, I will always have to take an inner product between my observations will would be 10,000 dimensional and this normal vector and in the dual domain it turns out that we operate with variables with as many variables at most as we have observations, rather than features.

→ In addition, we will discuss, so called kernel trick which allows to use all of this formalism that so far has involved hyperplanes only, to use this formalism also for non-linear classification problems.

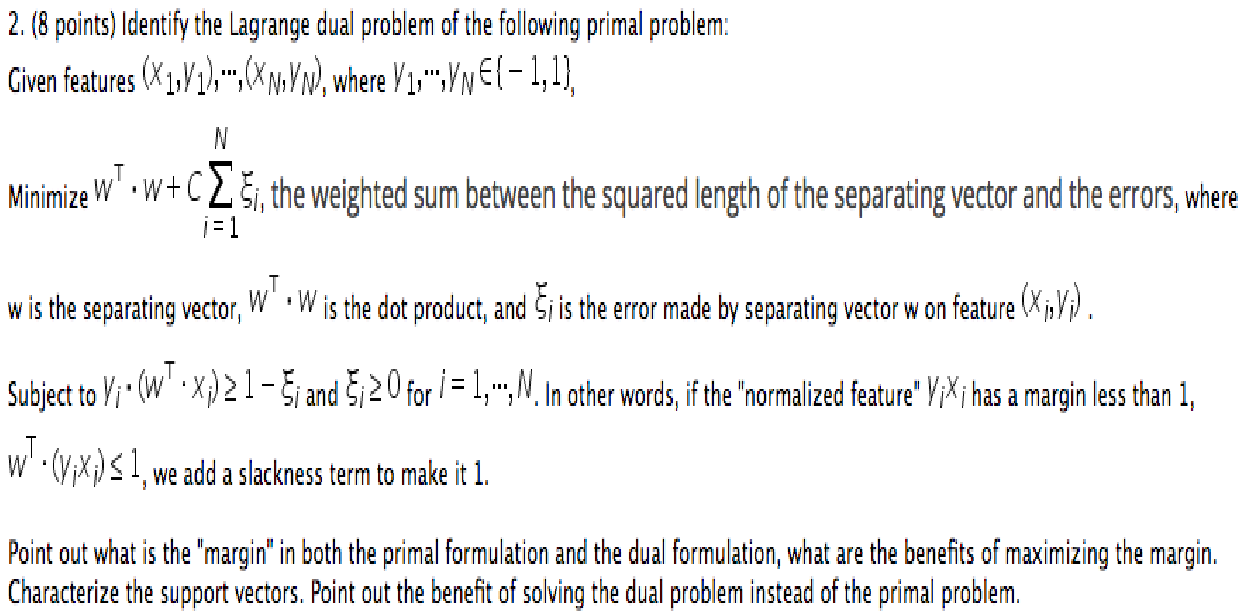
→ But up here if $y_i(w^Tx_i+b) \geq 1-\xi_i$

$\xi_i \geq 0$

were equalities. we could have used the Lagrangian formalism but we have inequalities and the ~~for~~ inequalit the proper approach is to use the KKT equations, also called "lagrange for inequalities".

So, I am setting up a Lagrangian function here: which involves both primal variables and, w, b and ξ, and new dual variables, the langrange multipliers α & μ.

So, I am copying the same terms here



2. (8 points) Identify the Lagrange dual problem of the following primal problem:

Given features $(x_1,y_1),\cdots,(x_N,y_N)$, where $y_1,\cdots,y_N \in \{-1,1\}$,

Minimize $w^T \cdot w + C\sum_{i=1}^{N}\xi_i$, the weighted sum between the squared length of the separating vector and the errors, where w is the separating vector, $w^T \cdot w$ is the dot product, and ξ_i is the error made by separating vector w on feature (x_i,y_i) .

Subject to $y_i \cdot (w^T \cdot x_i) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for $i = 1,\cdots,N$. In other words, if the "normalized feature" $y_i x_i$ has a margin less than 1, $w^T \cdot (y_i x_i) \leq 1$, we add a slackness term to make it 1.

Point out what is the "margin" in both the primal formulation and the dual formulation, what are the benefits of maximizing the margin. Characterize the support vectors. Point out the benefit of solving the dual problem instead of the primal problem.

2nd solution: My second at taking a different approach to the solution of problem 2 of this assignment

Given features $(x_1, y_1), (x_2, y_2), \dots (x_N, y_N)$

where $y_1, \dots y_n \in [-1, 1]$

Minimize the $w^T . w + C \sum_{i=1}^{N} \varepsilon_i$

$$f(x) = C \sum_{i=1}^{N} \varepsilon_i \text{ (say)}$$

dual function $L^*(u) = \min_{\varepsilon \in P} f(\varepsilon) + u^T w(\varepsilon)$

$$= \min_{\varepsilon \in P} \lambda \left[f(\varepsilon) + \omega_1^T g(\varepsilon) \right] + [1-\lambda] \left[f(\varepsilon) + \omega_2^T g(\varepsilon) \right]$$

$$\geq \lambda \left[\min_{\varepsilon \in P} f(\varepsilon) + \omega_1^T w(\varepsilon) \right] + (1-\lambda) \left[\min_{\varepsilon \in P} (f(\varepsilon) + \omega_2^T g(\varepsilon) \right]$$

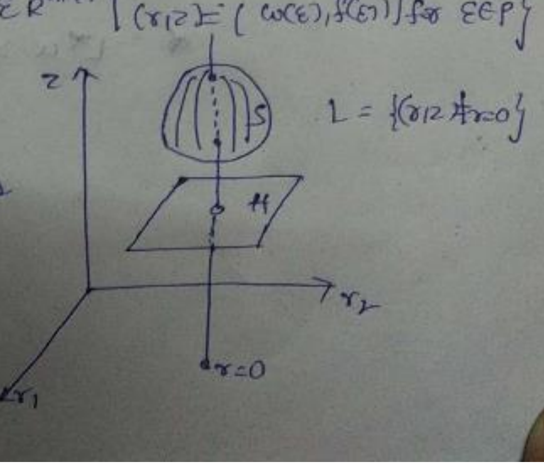
$$= \lambda L^*(u_1) + (1-\lambda) L^*(u_2)$$

let us Consider primal problem from a "resource cost" point of view for each $\varepsilon \in P$ we have array of resource and costs associated with ε

$$\begin{pmatrix} r \\ z \end{pmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_m \\ z \end{bmatrix} = \begin{bmatrix} w_1(\varepsilon) \\ w_2(\varepsilon) \\ \vdots \\ w_m(\varepsilon) \\ f(\varepsilon) \end{bmatrix} = \begin{bmatrix} w(\varepsilon) \\ f(\varepsilon) \end{bmatrix}$$

we call the region S

$$S = \{ (r, z) \in R^{m+1} \mid (r, z) = (w(\varepsilon), f(\varepsilon)) \text{ for } \varepsilon \in P \}$$



The column geometry of primary and dual problem

The feasible solution with lowest cost corresponds to lowest point on intersection of S and L. The ~~feasible~~ lowest value is exactly the value of primal problem.

→ In the figes shown hyperplane H, lies below the regions S suppose we try to find that hyperplane H lying below S whose intersection with L is Large as possible

let formulate it

$$H = H_{w,\alpha} = \{(\gamma, z) \in R^{m+1} \mid z + w^T\gamma = \alpha\}$$

$$L = \{(\gamma z) \in R^{m+1} \mid \gamma = 0\}$$

The intersection $H_{w,\alpha}$ with L is $(H \cap L)$ occur at point $(\gamma, z) = (0, \alpha)$

→ $H_{u,\alpha}$ lying below S whose intersection with L is highest

$$\text{maximum}_{u,\alpha} \quad \alpha$$

Conditions for $\max_{u,\alpha}$ s.t. $H_{u,\alpha}$ lies below S.

$$z + u^T\gamma \geq \alpha \text{ for all } (\gamma, z) \in S$$

$$f(\varepsilon) + w^T w \geq \alpha \text{ for all } \varepsilon \in P$$

$$L^* u \geq \alpha$$

3. (Optional) Formulate the primal problem and derive the dual problem if there are multiple classes.
Ans: The solution remains more and less the same other than a few changes mentioned below:

Bonus Question Solution

Soln:- For a multiclass classification problem, a new multi-class classifier called NHCMC - Nonparallel hyperplanes classifier for multiclass classification

Consider the multiple classification problem with the training set: $T = \{(x_1, y_1), \ldots, (x_l, y_l)\}$, — (14)
where $x_i \in R^n$, $i = 1, \ldots, l$, $y_i \in \{1, \ldots, K\}$ is the corresponding pattern of x_i.

For multiple classification, we seek K nonparallel hyperplanes:

$$(w_k \cdot x) + b_k = 0, \; k = 1, \ldots, K \quad —(15)$$

For convenience, we denote the no. of each class of the training set (14) as l_k and the points belonging to kth class as $A_k \in R^{l_k \times n}$, $k = 1, \ldots, K$. Besides, we define the matrix

$$B_k = [A_1^T, \ldots, A_{k-1}^T, A_{k+1}^T, \ldots, A_K^T]^T$$

as all the points except for the points belonging to kth class

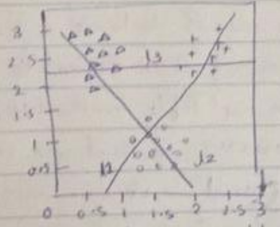


Fig1: A toy example learned by the linear NHCMC

Linear case

The primal Problem

We seek to construct k nonparallel hyplanes (15) by solving the following convex quadratic programming problems (QPPs):

$$\min_{w_k, b_k, \eta_k, \xi_k} \frac{1}{2} C_1 \|w_k\|^2 + \frac{1}{2}\eta_k^T \eta_k + C_2 e_{k2}^T \xi_k$$

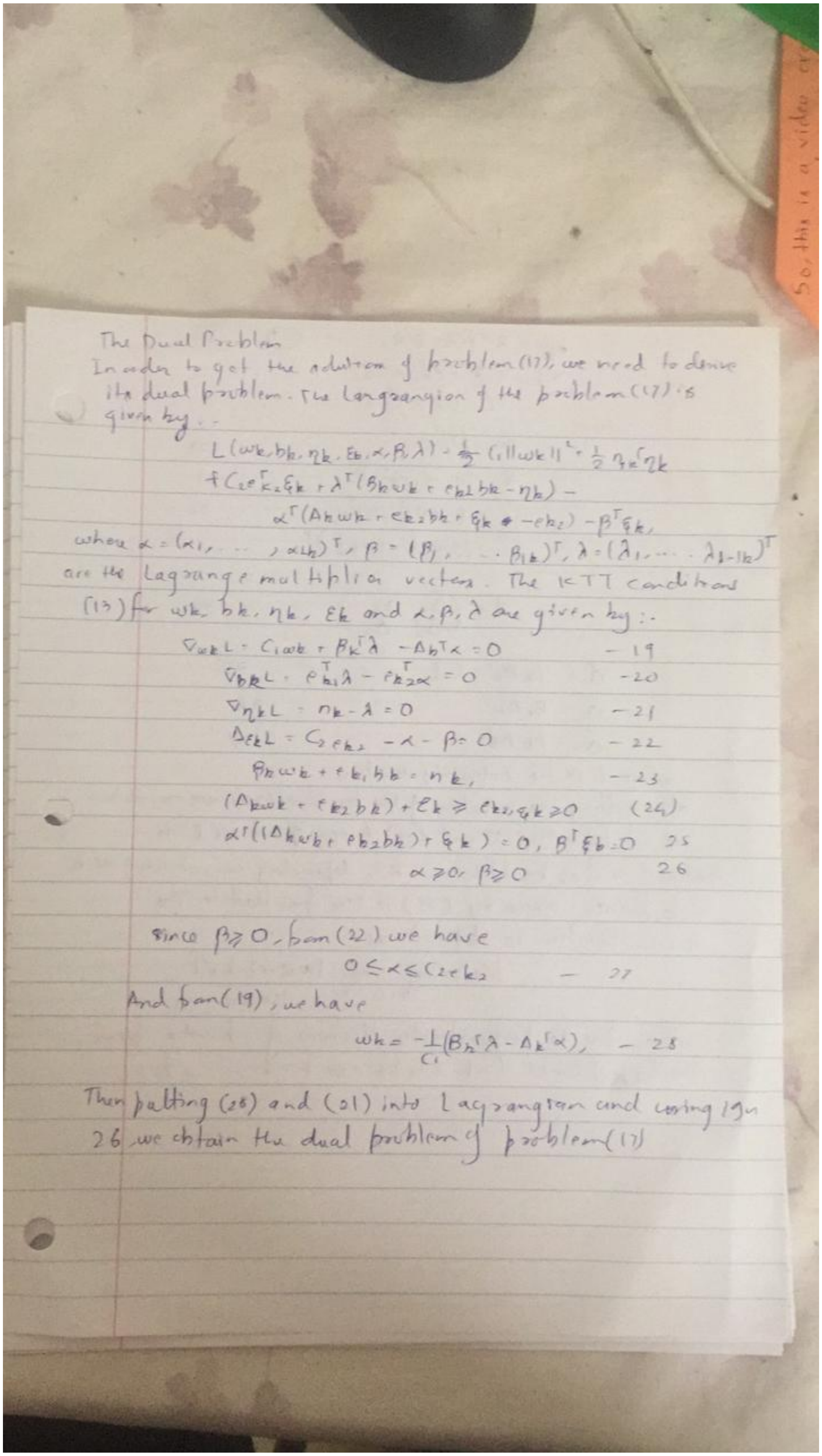
$$\text{s.t.} \quad B_k w_k + e_{k1} b_k = \eta_k,$$

$$(A_k w_k + e_{k2} b_k) + \xi_k \geq e_{k2} \qquad (17)$$

$$\xi_k \geq 0$$

where $\eta_k \in R^{(l-l_k)}$ is a variable and ξ_k is a slack variable, $e_{k1} \in R^{(l-l_k)}$ and $e_{k2} \in R^{l_k}$ are the vectors of one. $C_1 \geq 0$ and $C_2 \geq 0$ are penalty parameter.

In order to illustrate the primal problem of NHCMC, we generated an artificial two dimensional three-class dataset. The geometric interpretation of above problem with $x \in R^2$ is shown in Figure above. where minimizes the sum of the squared distance from the hyperplane of k-1 classes. that is all classes except for those of the k-th class, and the points of the k-th class are far from the ith hyperplane. Take the "*" class in Figure 1 as an example. We hope the hyperplane of the the "*" class l_1 is far from the "*" points and close to the "o" and triangular points. In order to minimize classification, the points of the k-th class are at distance 1 from the hyperplane, and we minimize the sum of error variables with soft margin loss. The differences between multiple birth support vector machine (MBSVM) and NHCMC are that we introduce a regularization term to implement structural risk minimization (SRM) principle and a variable is introduced to make a term of objective function to be constraints. These changes have many +ve effects on original NHCMC.



So, this is a video

The Dual Problem

In order to get the solution of problem (17), we need to derive its dual problem. The Langrangion of the problem (17) is given by:-

$$L(w_k, b_k, \eta_k, \xi_k, \alpha, \beta, \lambda) = \frac{1}{2} C_1 \|w_k\|^2 + \frac{1}{2} \eta_k^T \eta_k$$
$$+ C_2 e_{k_2}^T \xi_k + \lambda^T (B_k w_k + e_{k1} b_k - \eta_k) -$$
$$\alpha^T (A_k w_k + e_{k_2} b_k + \xi_k - e_{k_2}) - \beta^T \xi_k,$$

where $\alpha = (\alpha_1, \dots, \alpha_{l_k})^T$, $\beta = (\beta_1, \dots \beta_{lk})^T$, $\lambda = (\lambda_1, \dots \lambda_{l-lk})^T$ are the Lagrange multiplier vectors. The KTT conditions (13) for w_k, b_k, η_k, ξ_k and α, β, λ are given by:-

$$\nabla_{w_k} L = C_1 w_k + B_k^T \lambda - A_k^T \alpha = 0 \qquad - 19$$
$$\nabla_{b_k} L = e_{k_1}^T \lambda - e_{k_2}^T \alpha = 0 \qquad -20$$
$$\nabla_{\eta_k} L = \eta_k - \lambda = 0 \qquad -21$$
$$\Delta_{\xi_k} L = C_2 e_{k_2} - \alpha - \beta = 0 \qquad -22$$
$$B_k w_k + e_{k_1} b_k = \eta_k, \qquad -23$$
$$(A_k w_k + e_{k_2} b_k) + \xi_k \geqslant e_{k_2}, \xi_k \geqslant 0 \qquad (24)$$
$$\alpha^T((A_k w_k + e_{k_2} b_k) + \xi_k) = 0, \; \beta^T \xi_k = 0 \qquad 25$$
$$\alpha \geqslant 0, \; \beta \geqslant 0 \qquad 26$$

Since $\beta \geqslant 0$, from (22) we have

$$0 \leq \alpha \leq C_2 e_{k_2} \qquad - 27$$

And from (19), we have

$$w_k = -\frac{1}{C_1}(B_k^T \lambda - A_k^T \alpha), \qquad - 28$$

Then putting (28) and (21) into Lagrangian and using 19-26 we obtain the dual problem of problem (17)

$$\min_{\hat{\pi}} \frac{1}{2}\hat{\pi}^T\hat{\Lambda}\hat{\pi} + \hat{k}^T\hat{\pi}, \quad - \quad (29)$$

$$\text{s.t. } e^T_{k_1}\lambda - e^T_{k_2}\alpha = 0,$$

$$\hat{C}_1 \le \hat{\pi} \le \hat{C}_2,$$

where

$$\hat{\pi} = (\lambda^T, \alpha^T)^T \qquad 30$$

$$\hat{k} = (0, -C_1 e^T_{k_2})^T, \qquad 31$$

$$\hat{C}_1 = (-\infty e^T_{k_1}, 0)^T, \qquad 32$$

$$\hat{C}_2 = (+\infty e^T_{k_1}, C_2 e^T_{k_2})^T, \qquad 33$$

$$\hat{\Lambda} = \begin{pmatrix} \hat{Q}_1 & \hat{Q}_2 \\ \hat{Q}_2^T & \hat{Q}_3 \end{pmatrix} \quad - \quad 34$$

$$\hat{Q}_1 = B_k B_k^T + C_1 I \quad - \quad 35$$

$$\hat{Q}_2 = B_k A_k^T \qquad 36$$

$$\hat{Q}_3 = A_k A_k^T \qquad 37$$

where I is the identity matrix

After getting the solution of the problem (29), we can obtain w_k^* and b_k^* with (19) and (23). A new point $x \in R^n$ is assigned to class k ($k \in 1, \dots, K$), depending on which of the K hyperplanes given by (15) it lies farthest to. The decision function is defined as

$$f(x) = \arg\max_{k=1,\dots,K} \frac{|(w_k^* \cdot x) + b_k^*|}{\|w_k^*\|},$$

where |·| is the perpendicular distance of point x from the hyperplanes $(w_k \cdot x) + b_k = 0$, $k = 1, \dots, K$.