**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}

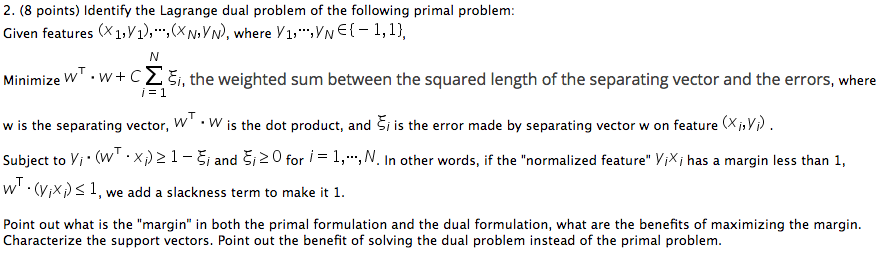
Best Score of tuned SVM: 0.985

But the assignment specifically asks to run for linear i.e. dot product:

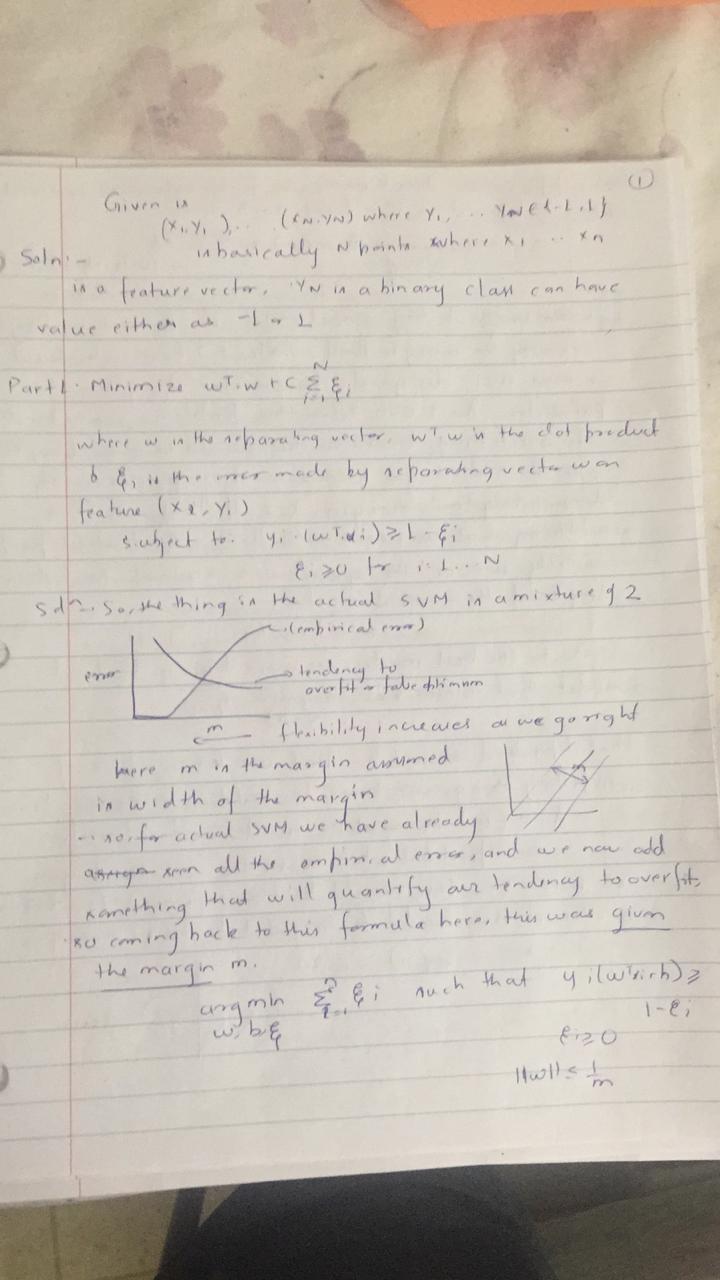
So the output for that was also run and could be seen below:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 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 |

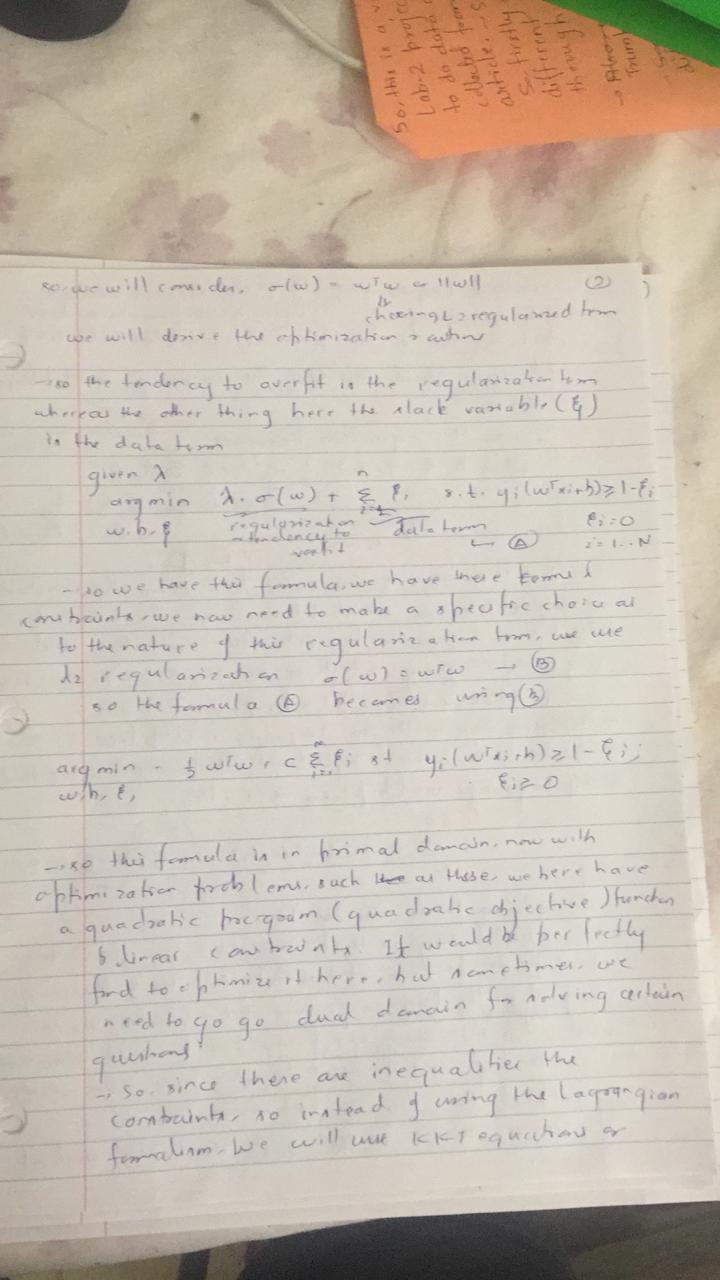
The different C hyperparameter which tells how much to avoid misclassifying and the different score values of accuracy are also shown.

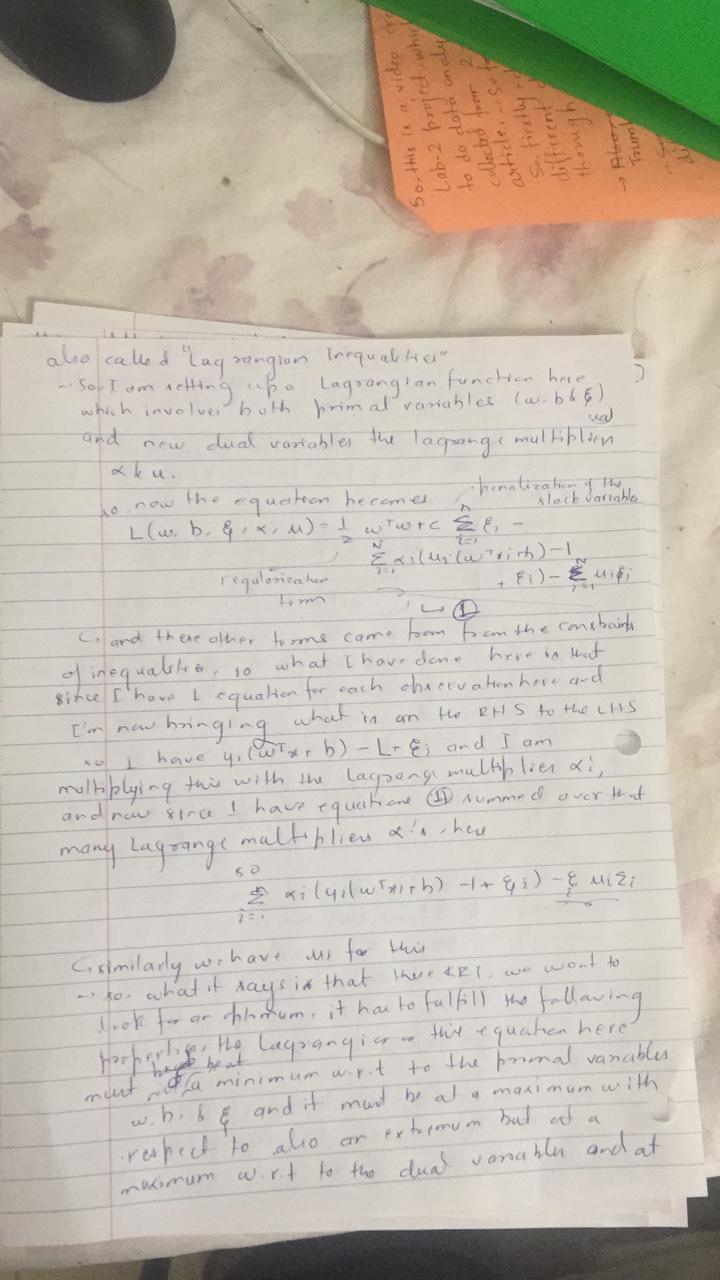


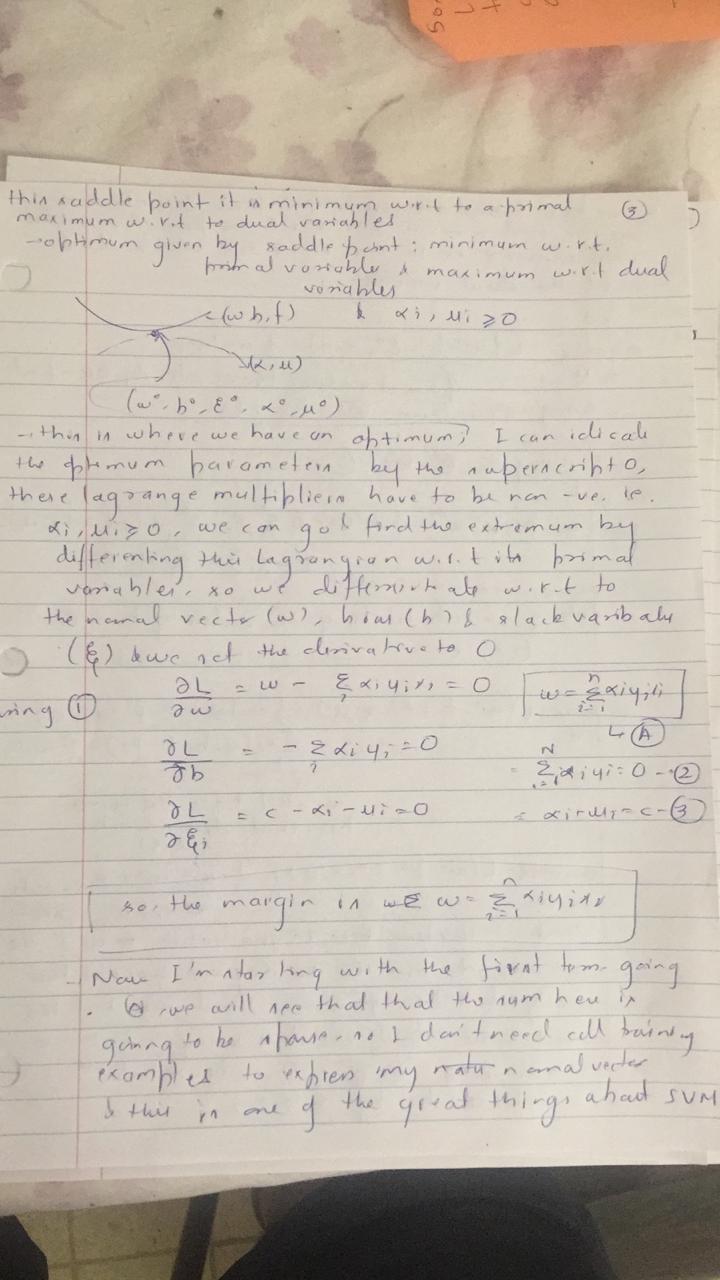
Soln:

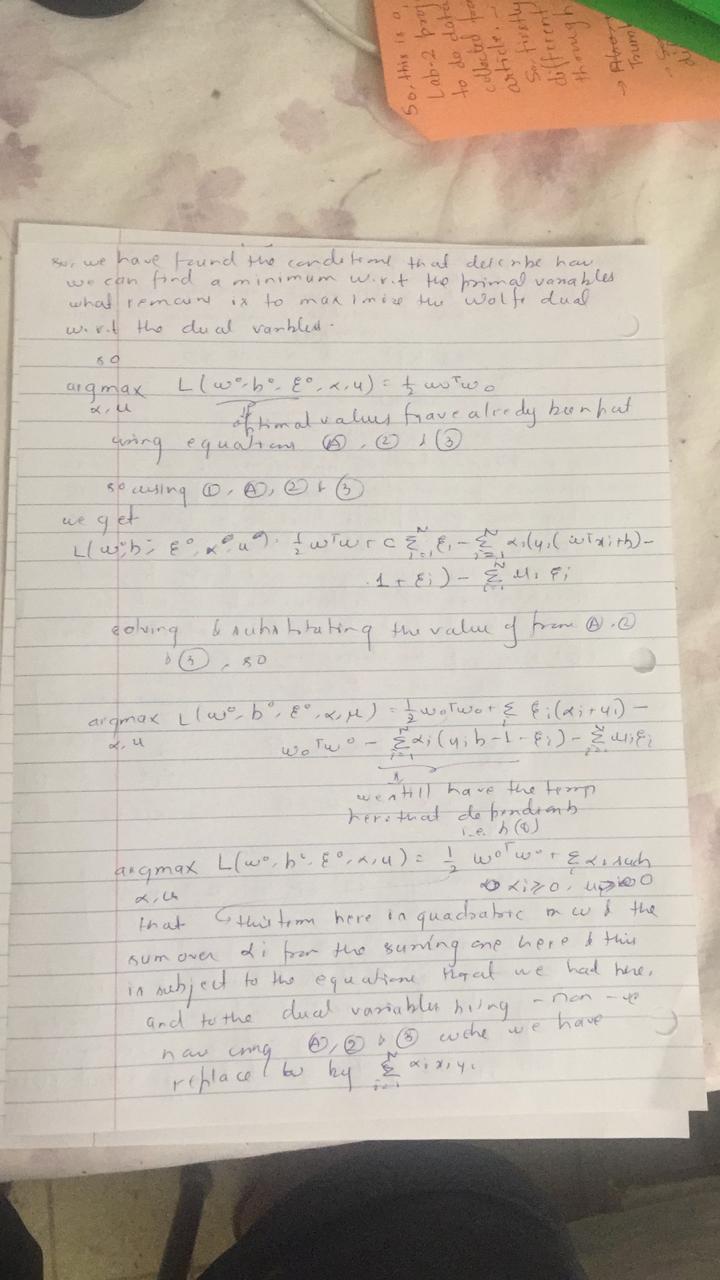


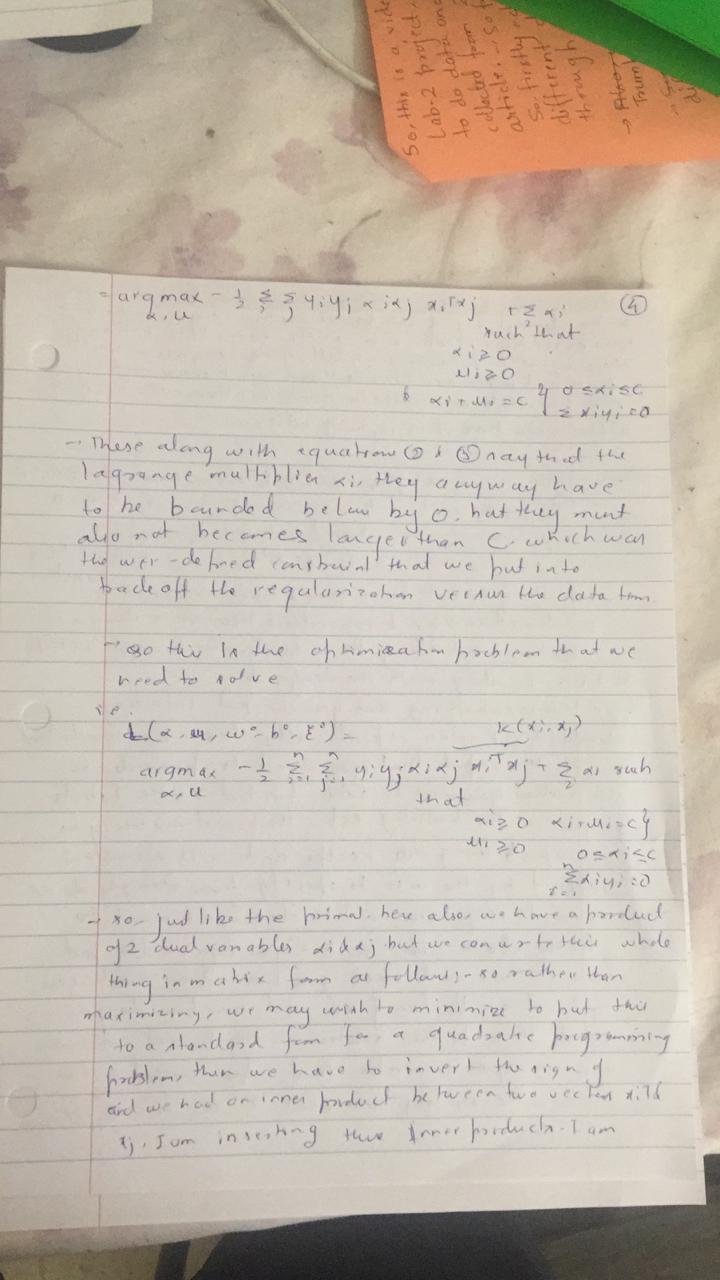


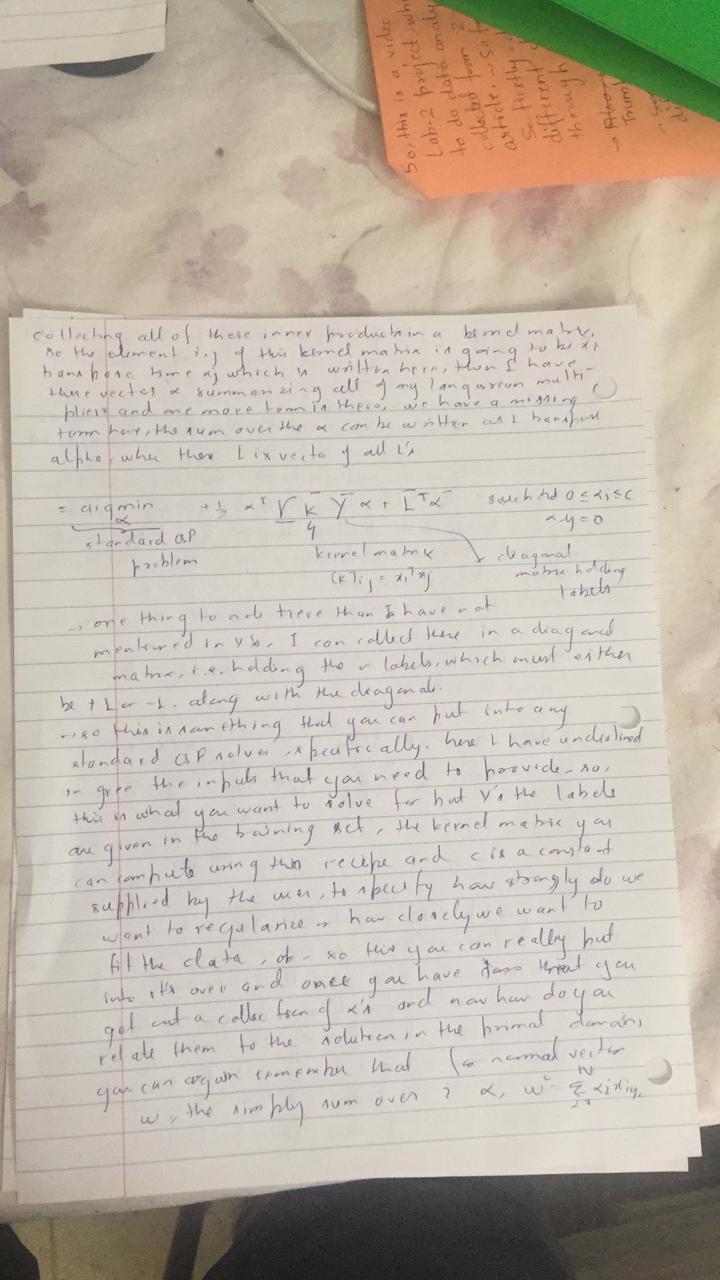


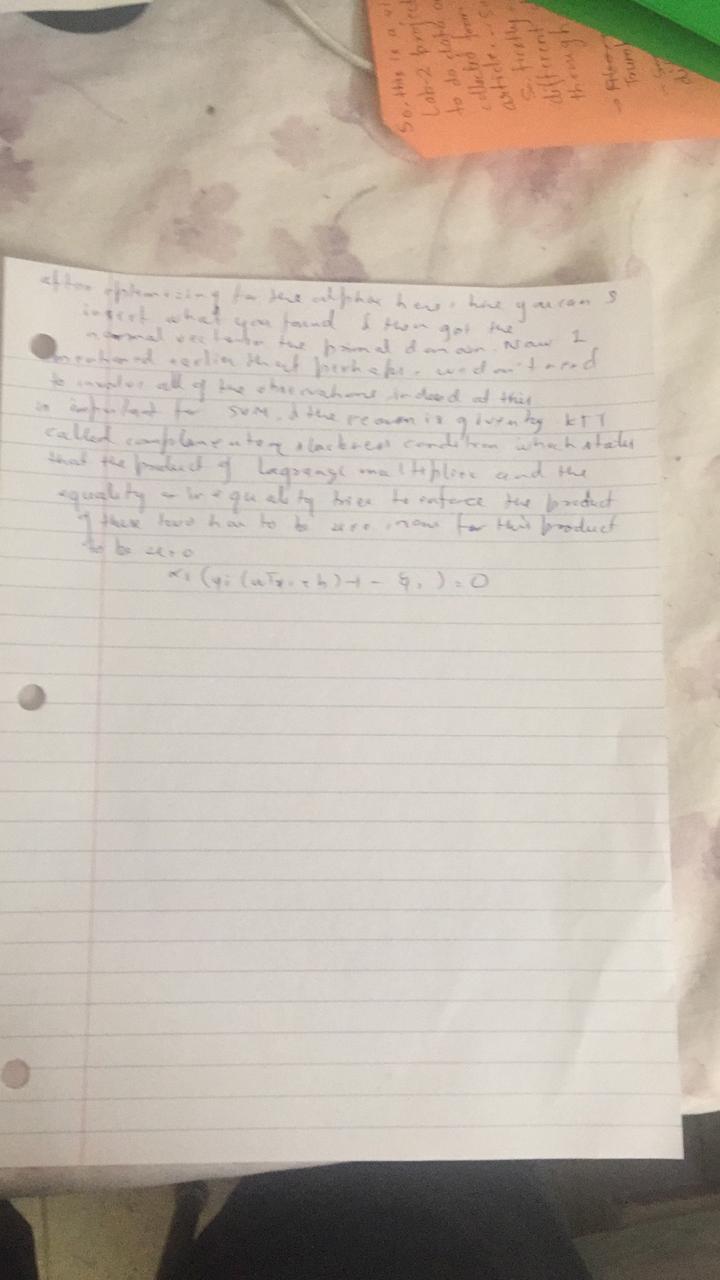






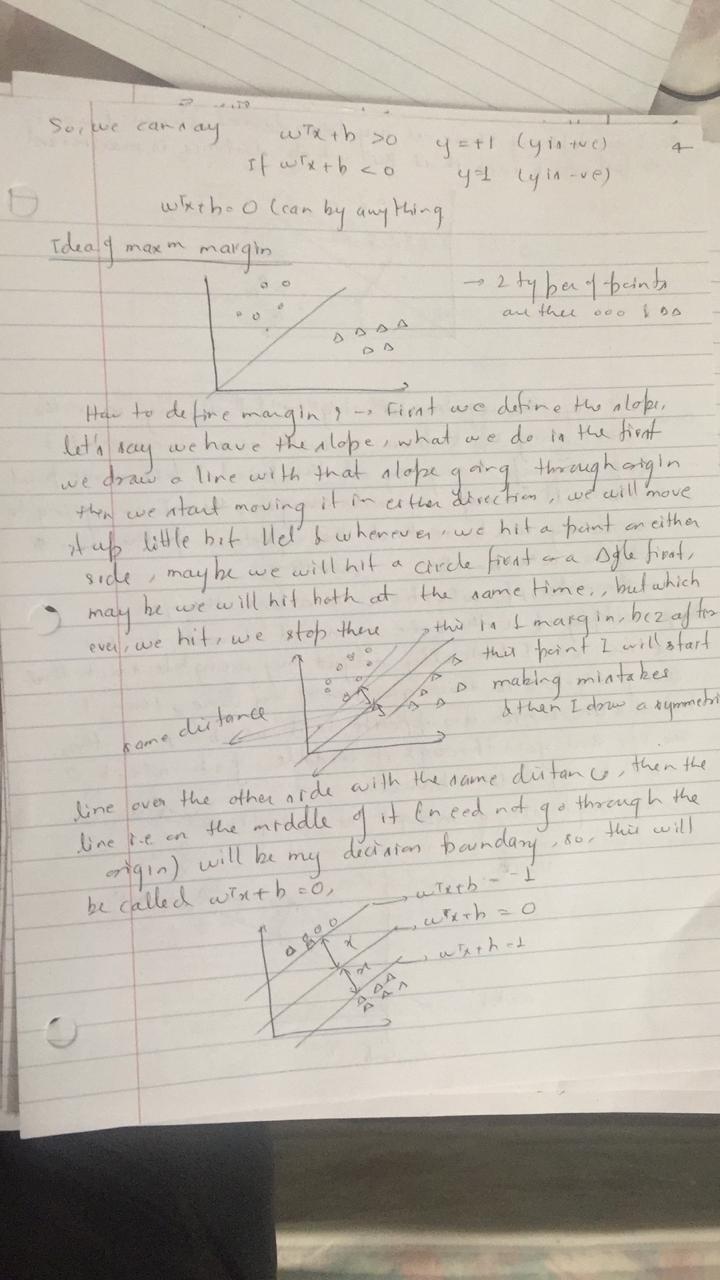


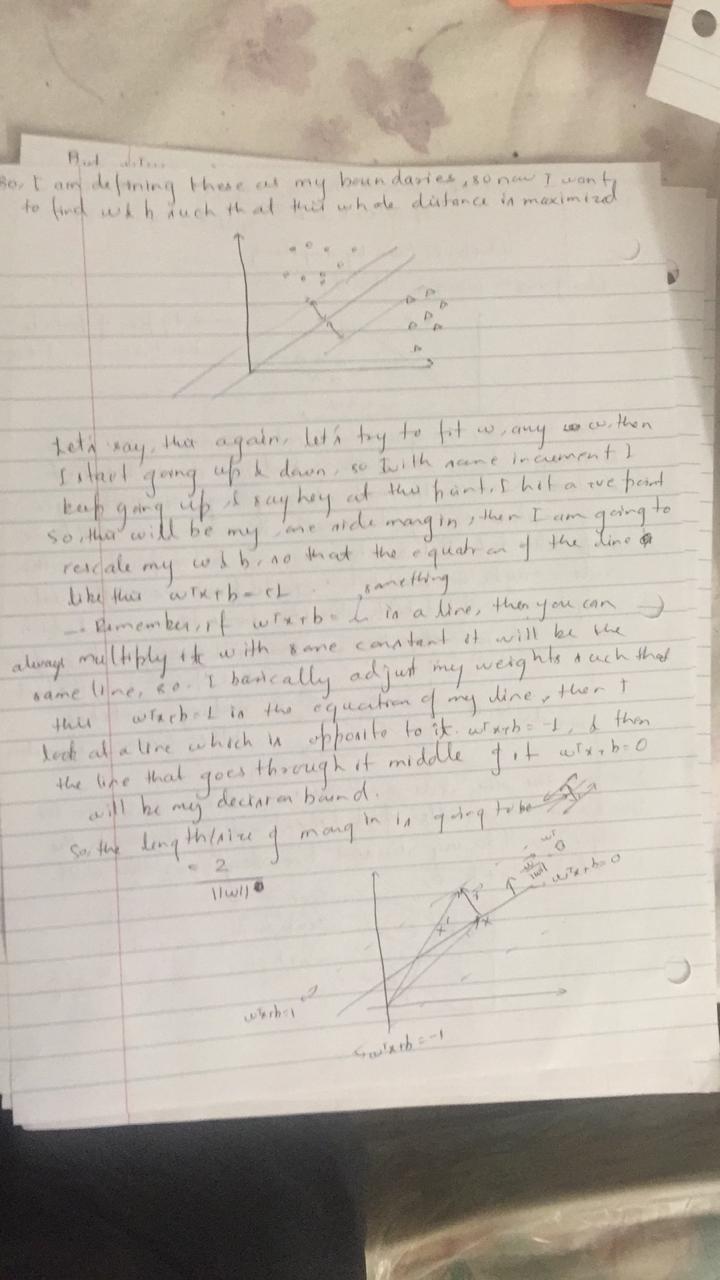


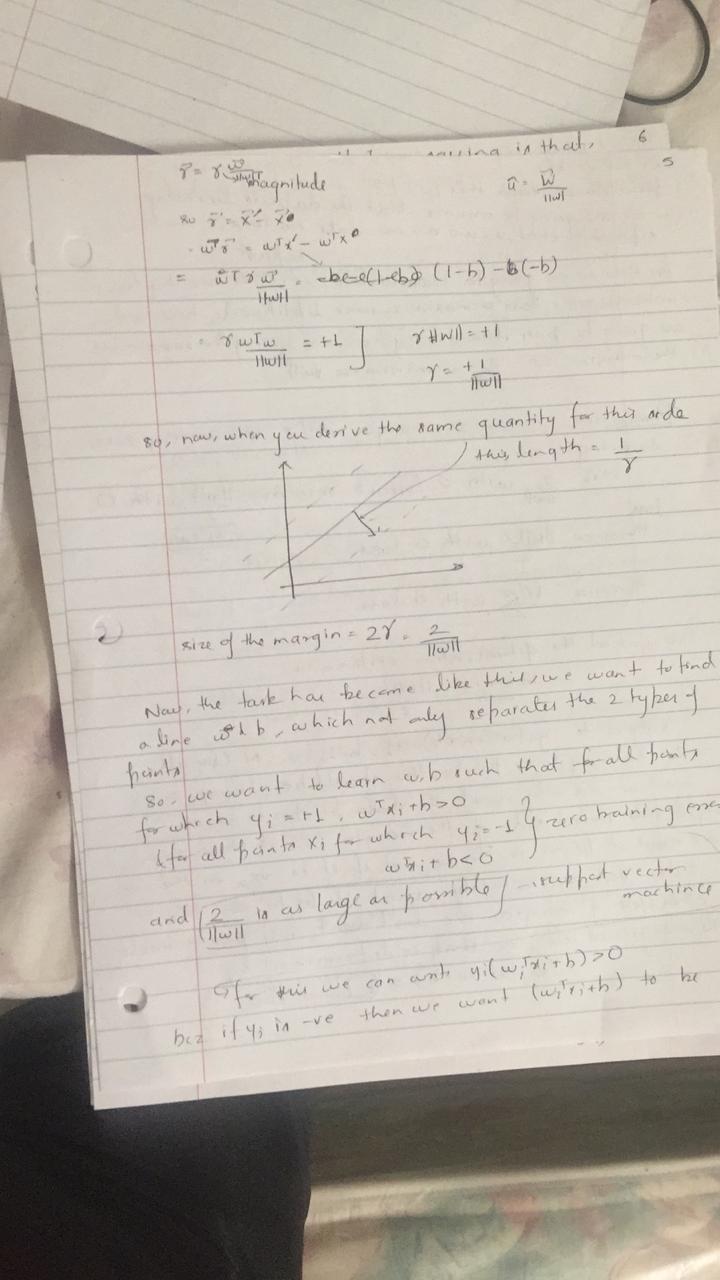


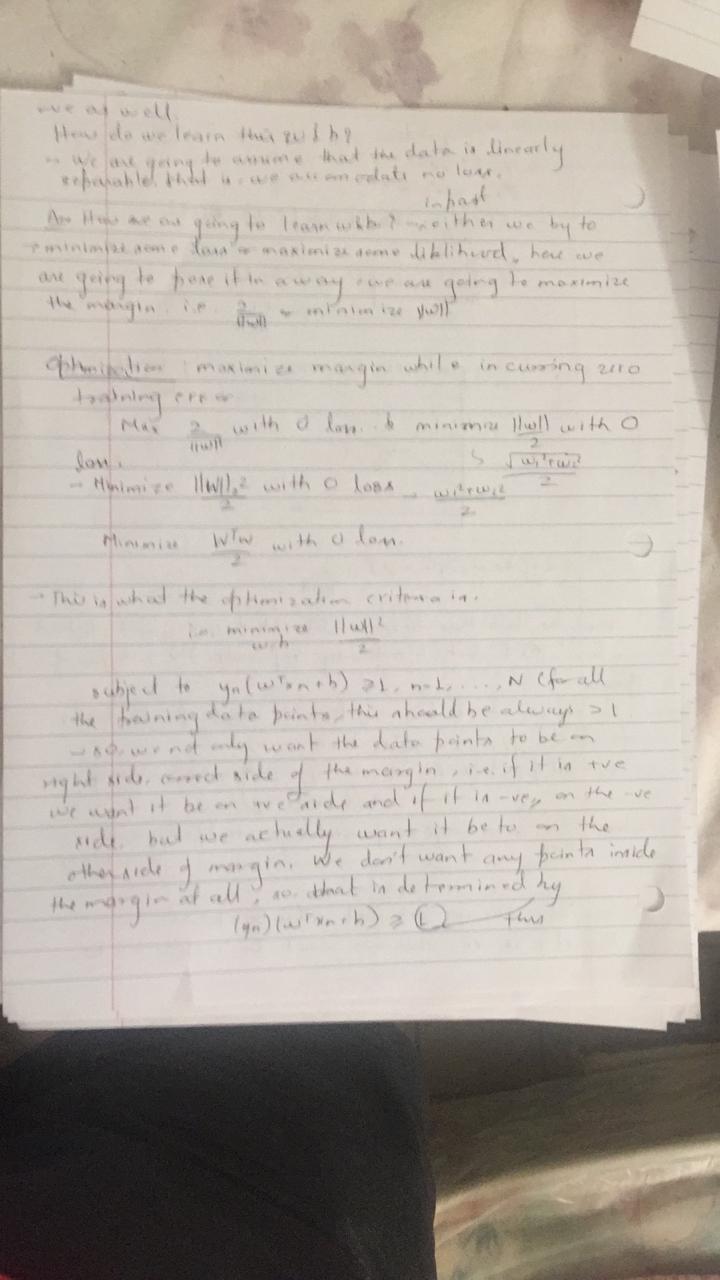
**2) part2) Point out what is the "margin" in both the primal formulation and the dual formulation,**

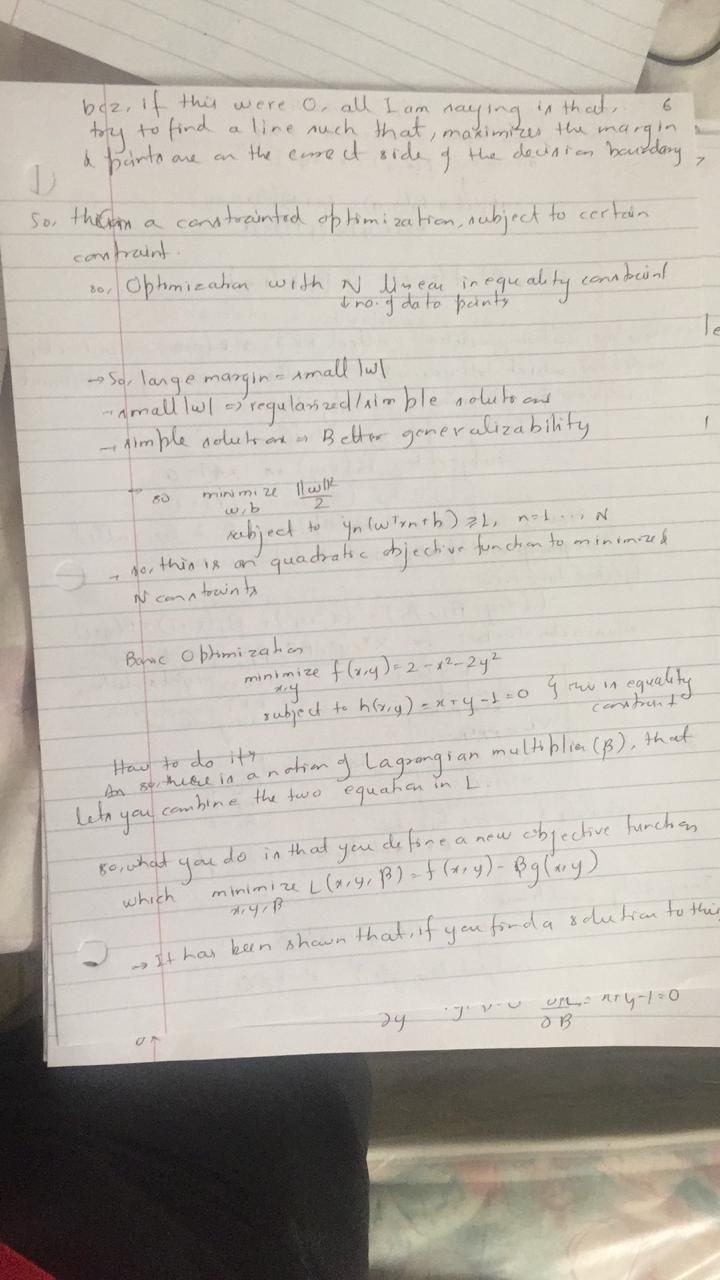
**Soln: As suggested in the above derivation**

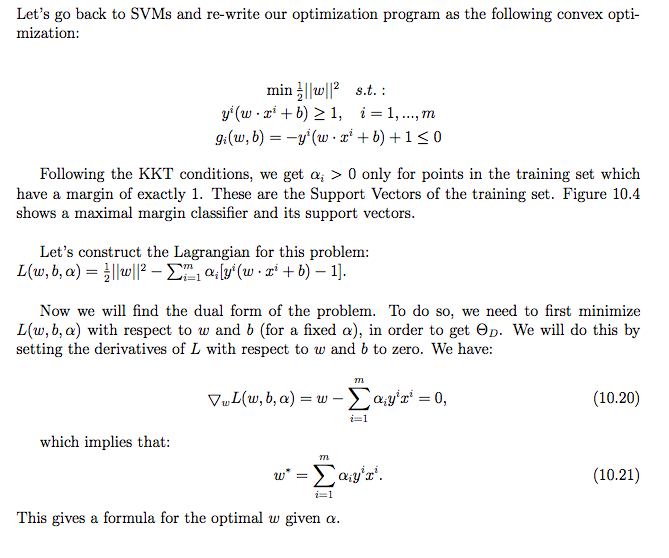






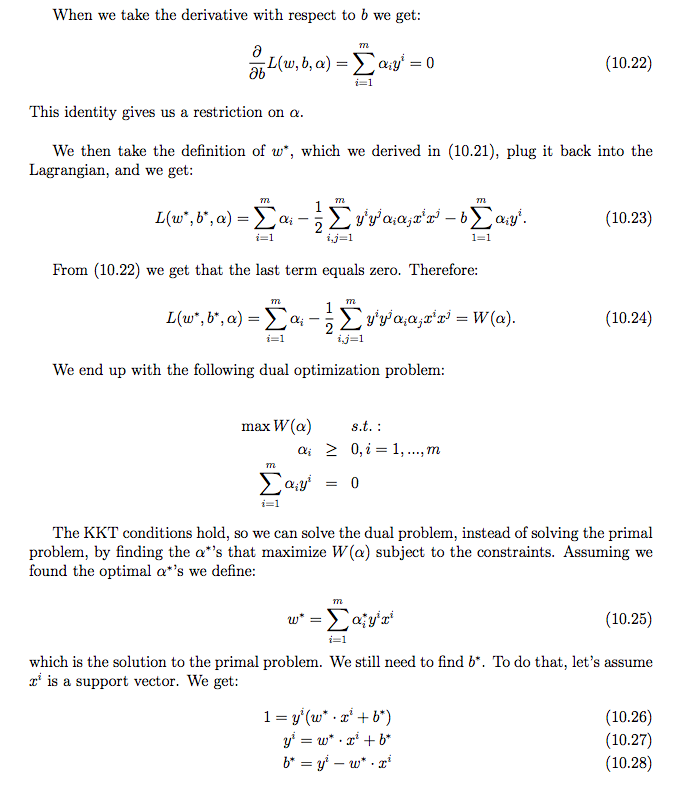






which can be found out by finding w\* is the optimum w here because w\*=summation of alpha \* y\* x (all three quantities going from 1 to N). So given alpha we can find the w

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

****

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

**Ans: Because of following reasons:**

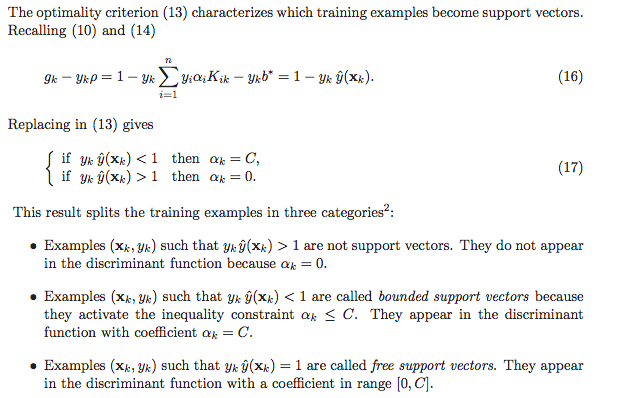
1)A large margin effectively corresponds to a regularization of SVM weights which prevents overfitting. Hence, we prefer a large margin (or the right margin chosen by cross-validation) because it helps us generalize our predictions and perform better on the test data by not overfitting the model to the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

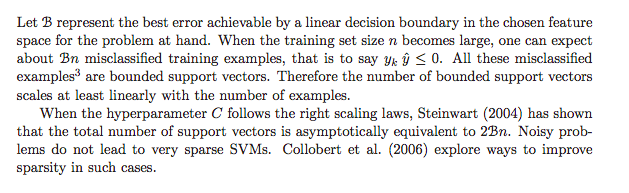
2) “Maximizing the margin seems good because points near the decision surface represent very uncertain classification decisions: there is almost a 50% chance of the classifier deciding either way. A classifier with a large margin makes no low certainty classification decisions. This gives you a classification 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

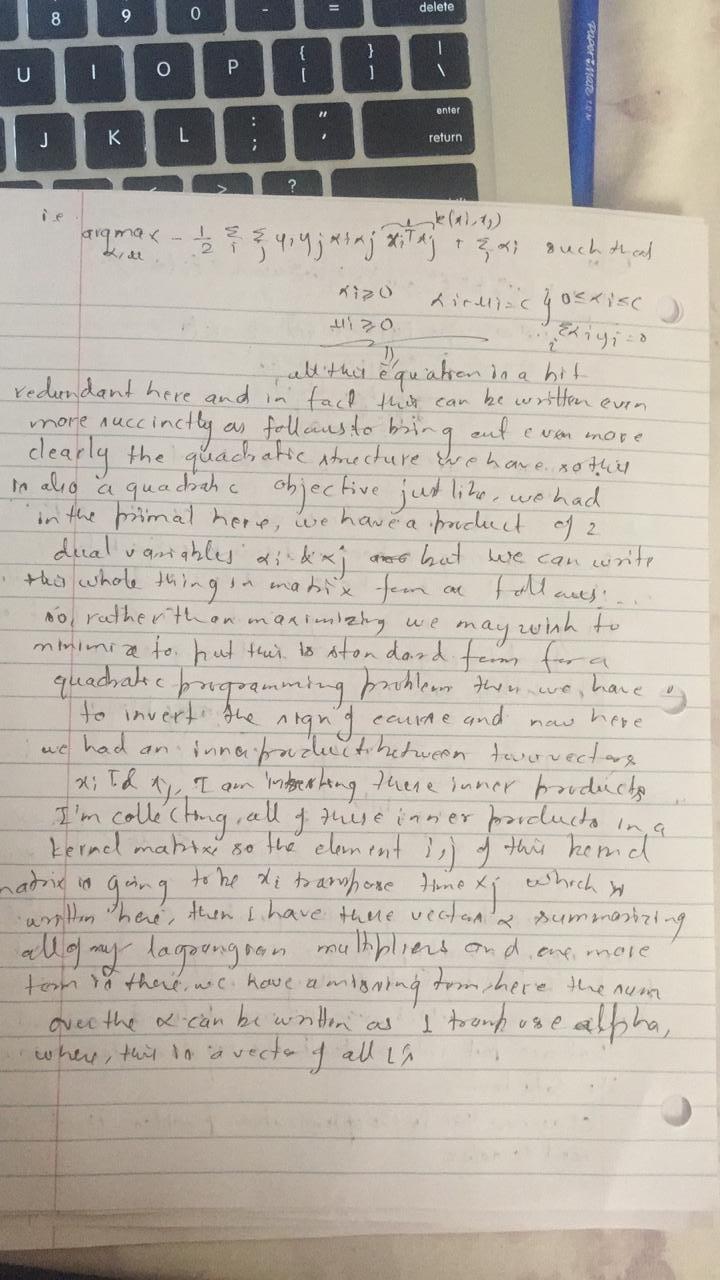
**2)Part4 Characterize the support vectors.**

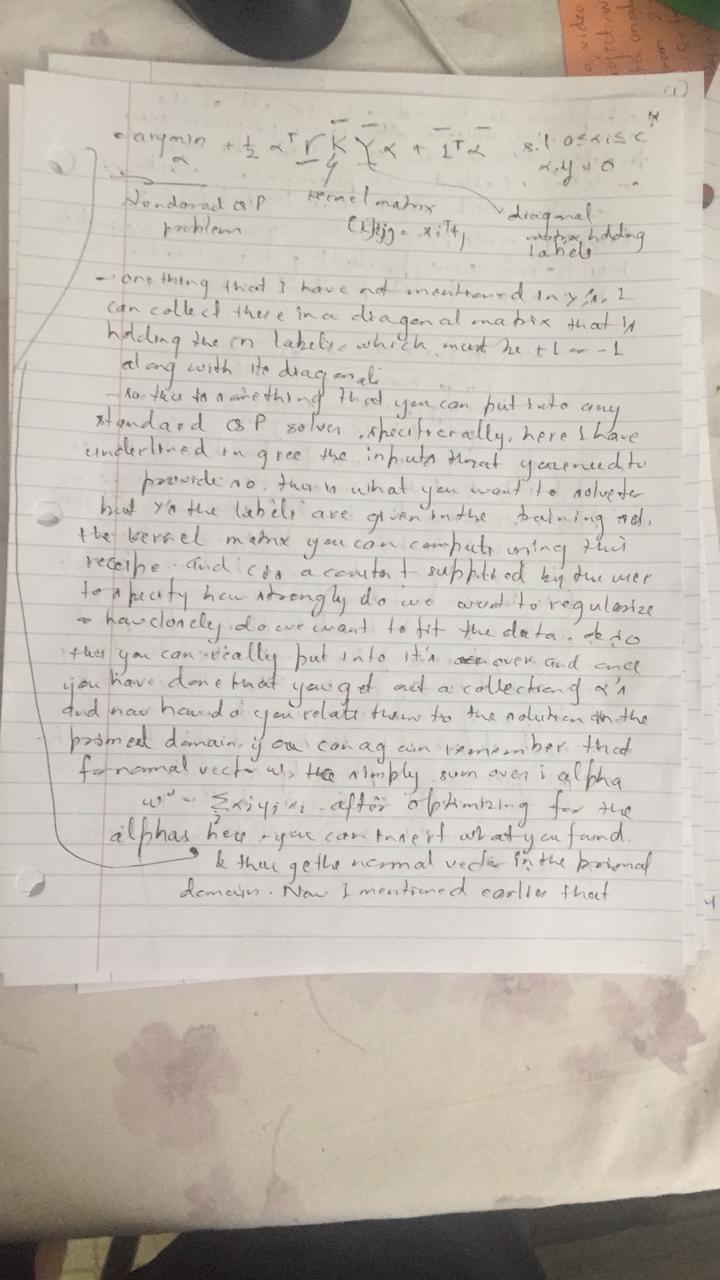
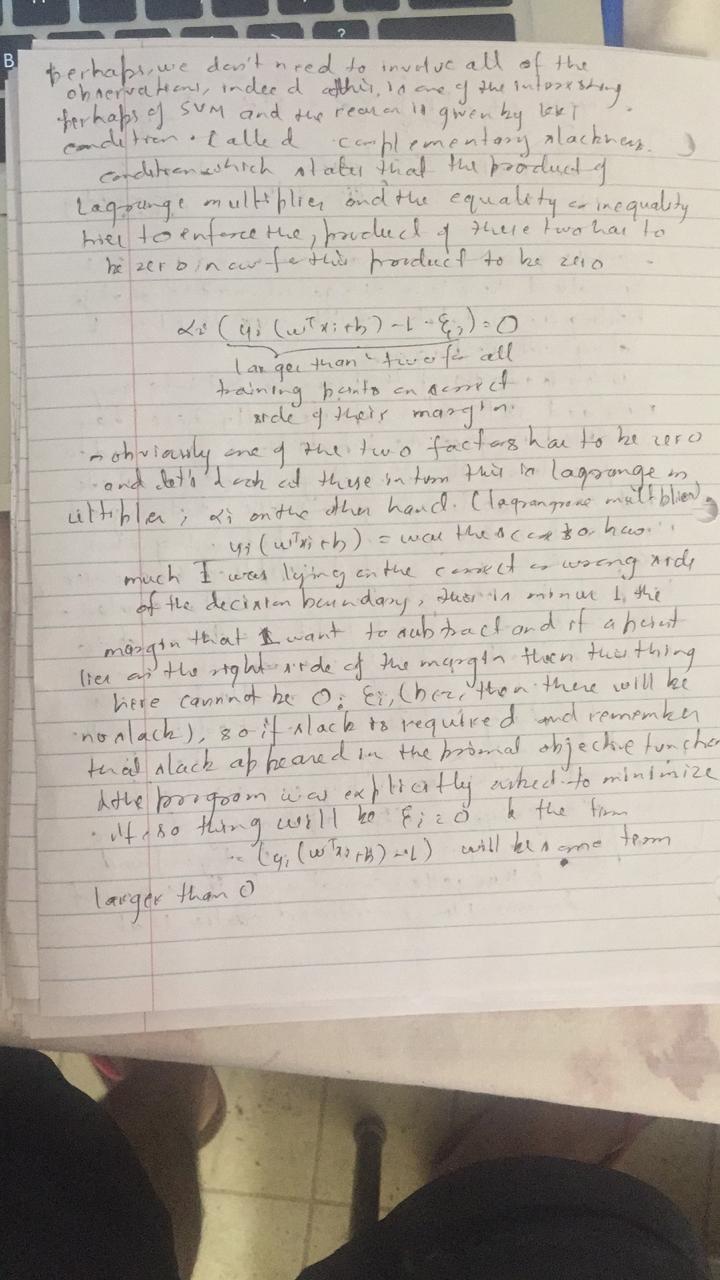
**Soln:**

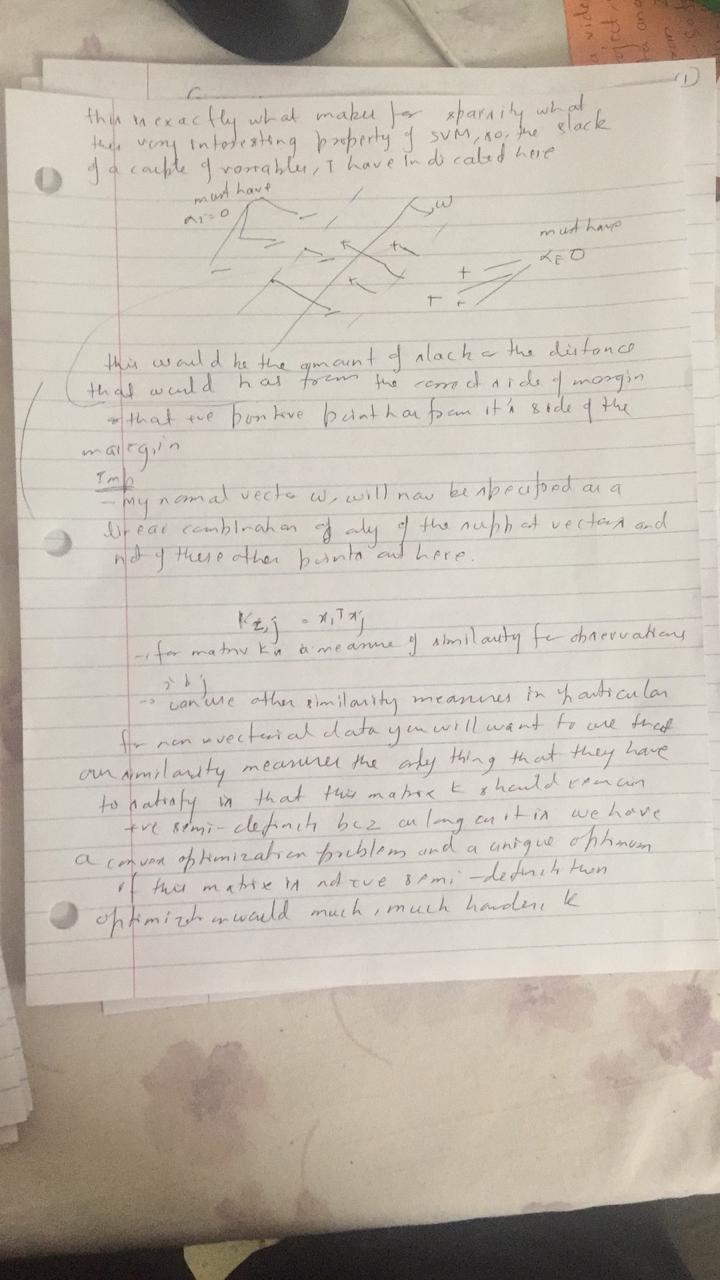




This is also explained below as part of solution of the above mentioned question:

****

**  
**

****

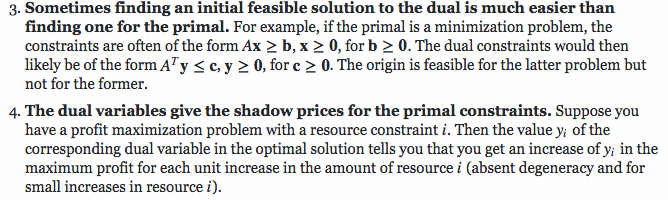
**2)Part5**

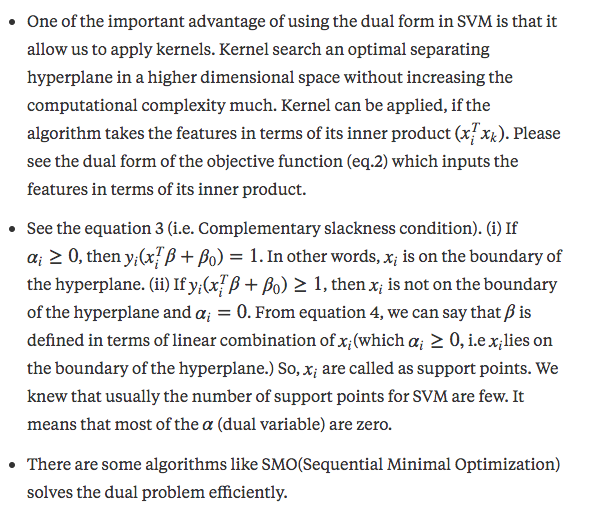
**Point out the benefit of solving the dual problem instead of the primal problem.**

**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.

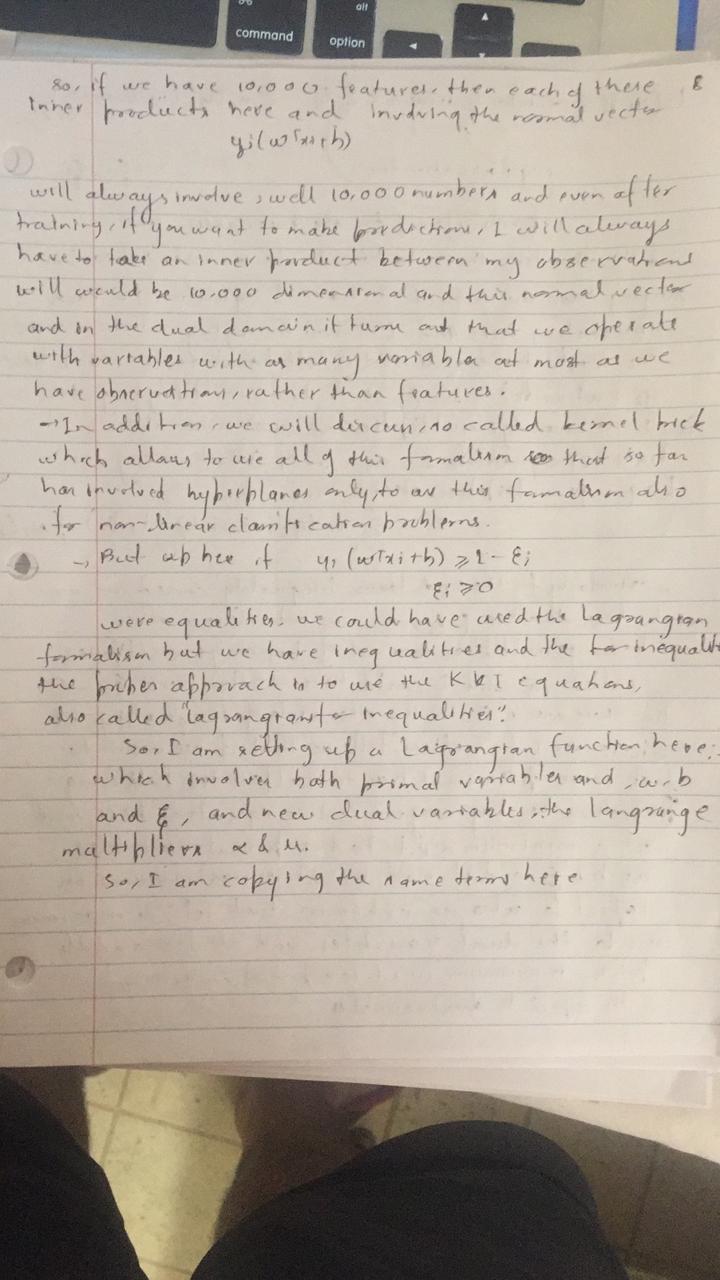
1. **The dual can be helpful for sensitivity analysis.** Changing the primal's right-hand side constraint vector or adding a new constraint to it can make the original primal optimal solution infeasible. However, this only changes the objective function or adds a new variable to the dual, respectively,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

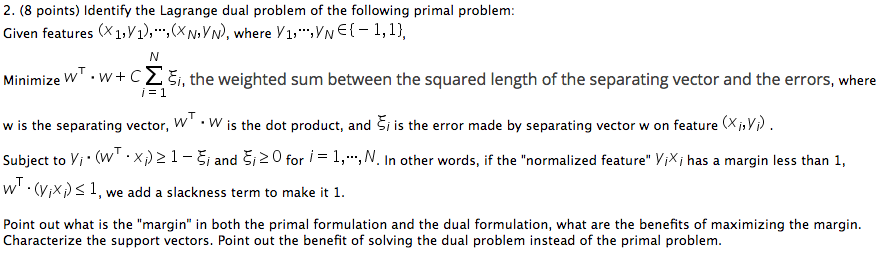
****

**5)**

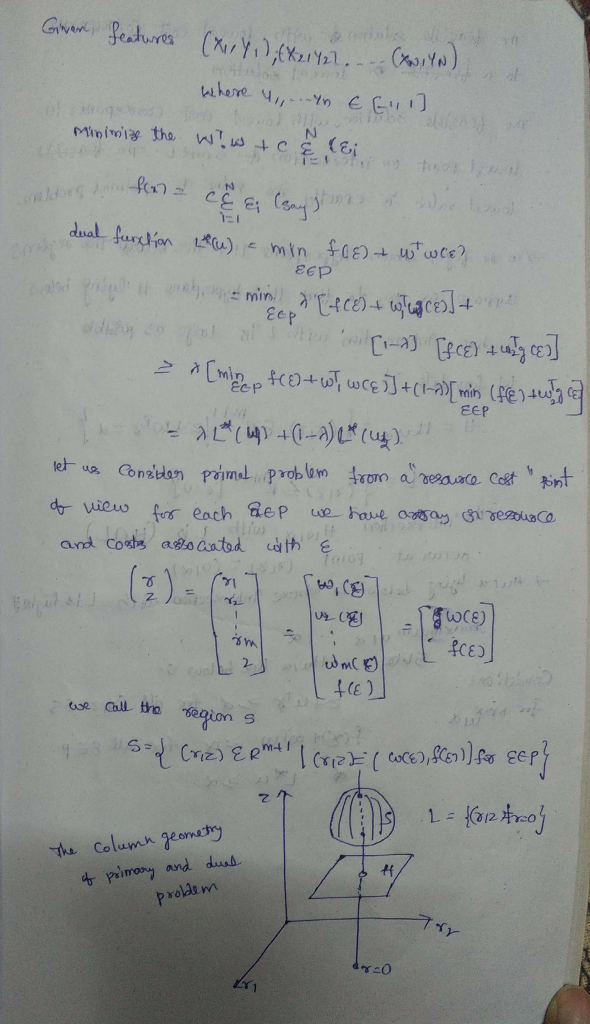
**A proper explanation is below:**

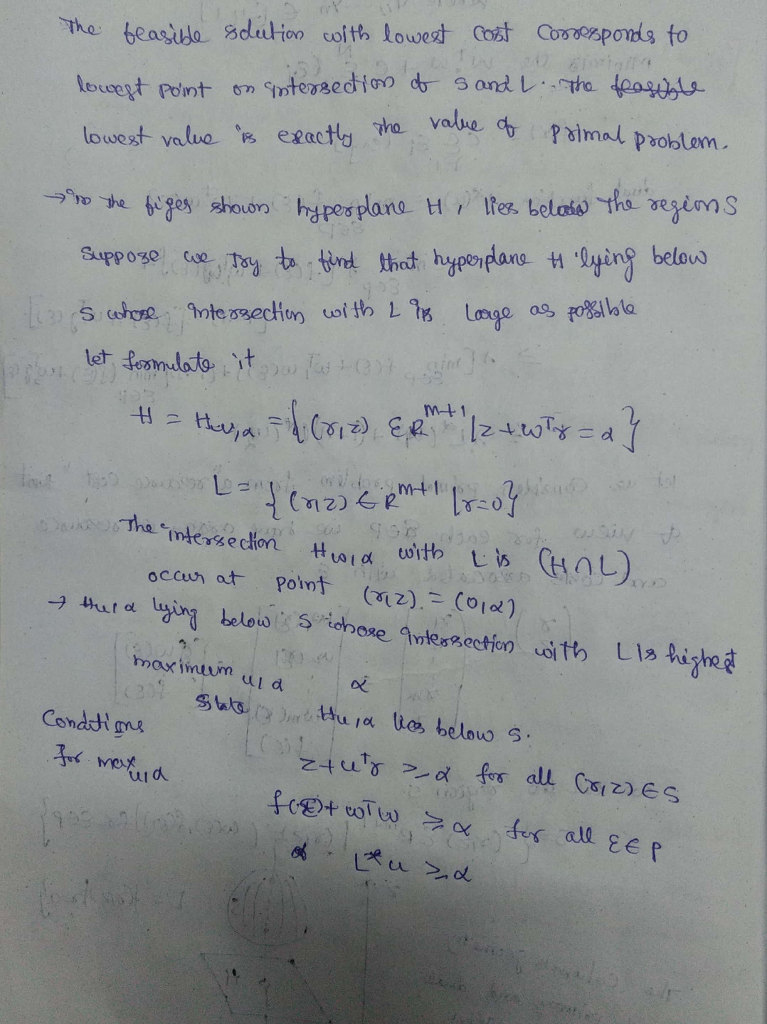




****

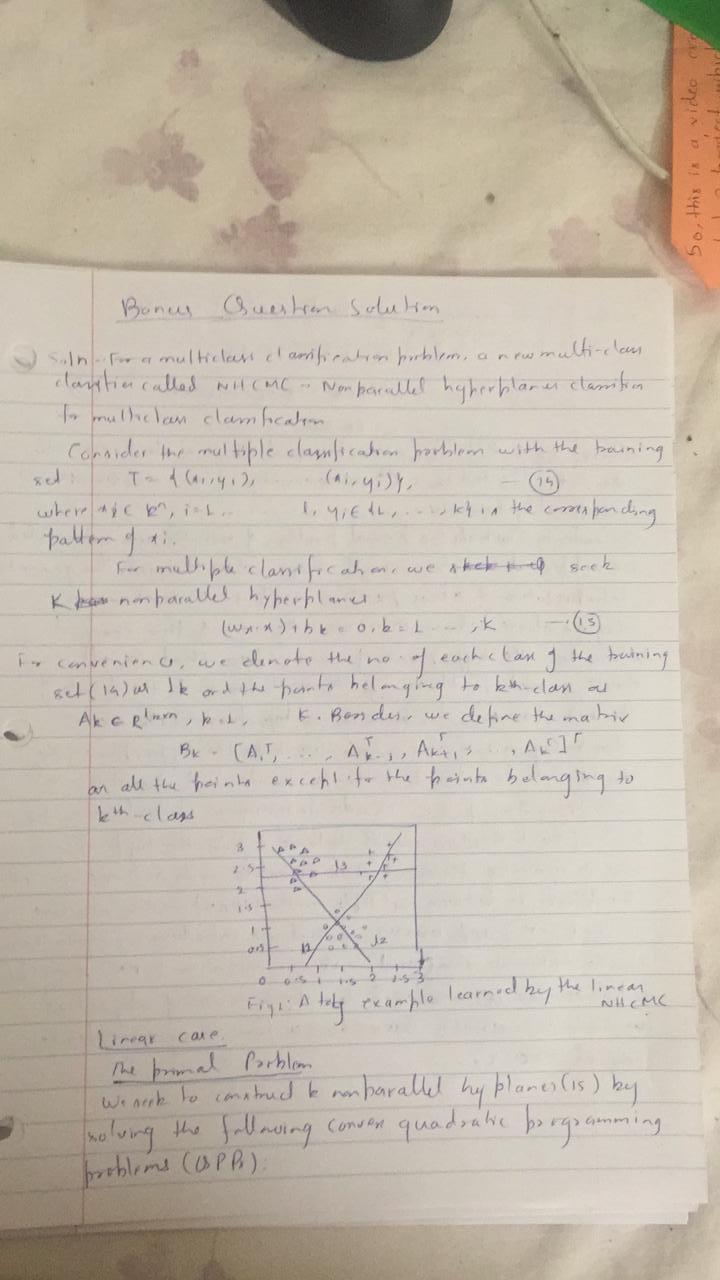
**2nd solution: My second at taking a different approach to the solution of problem 2 of this assignment**

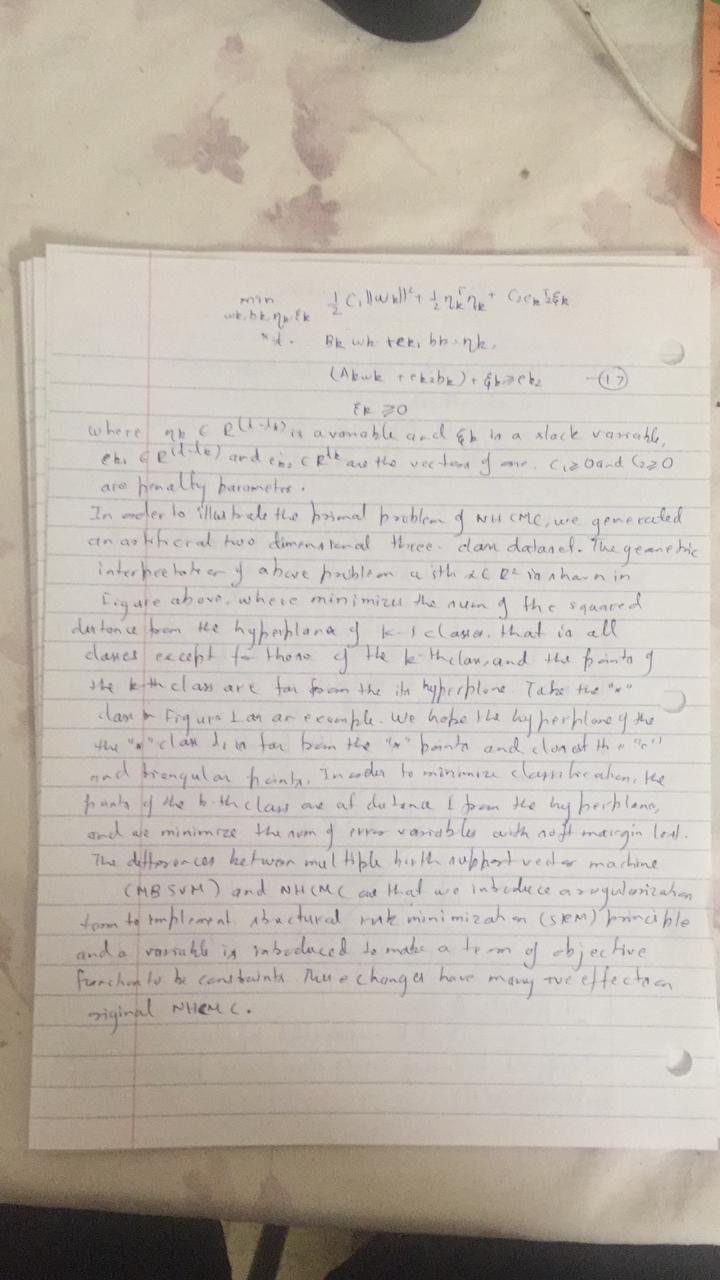
****

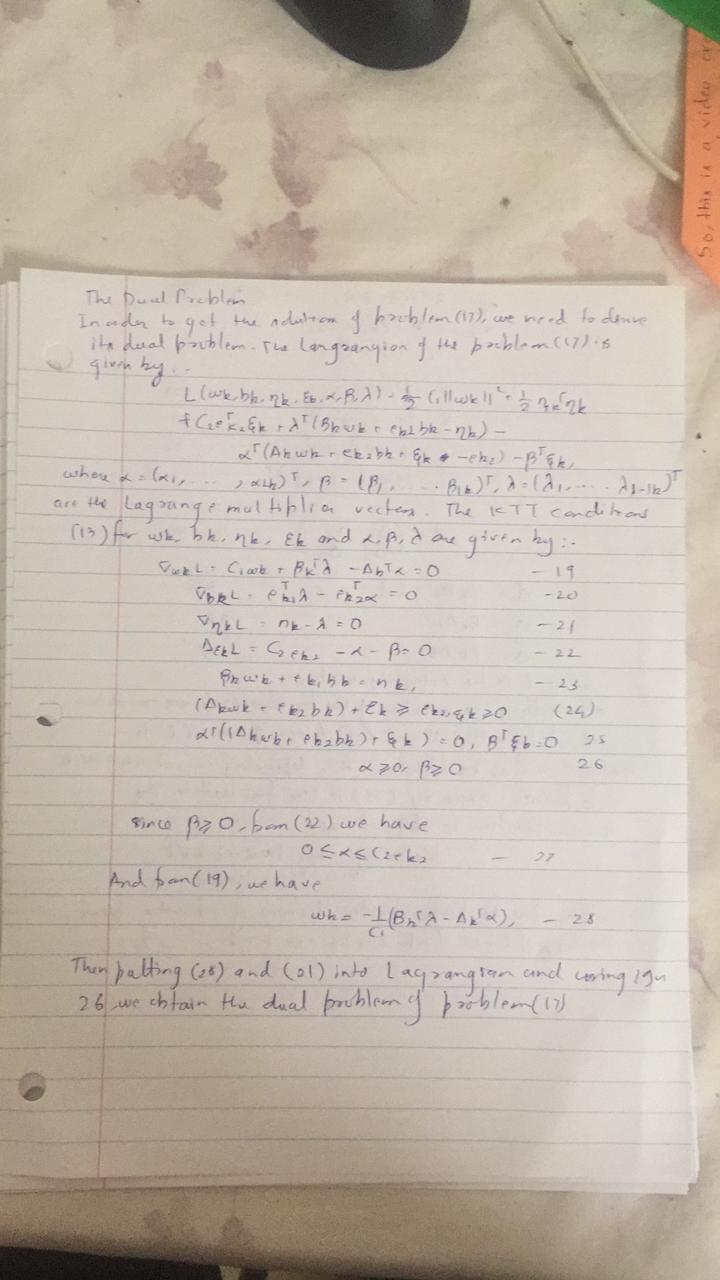
****

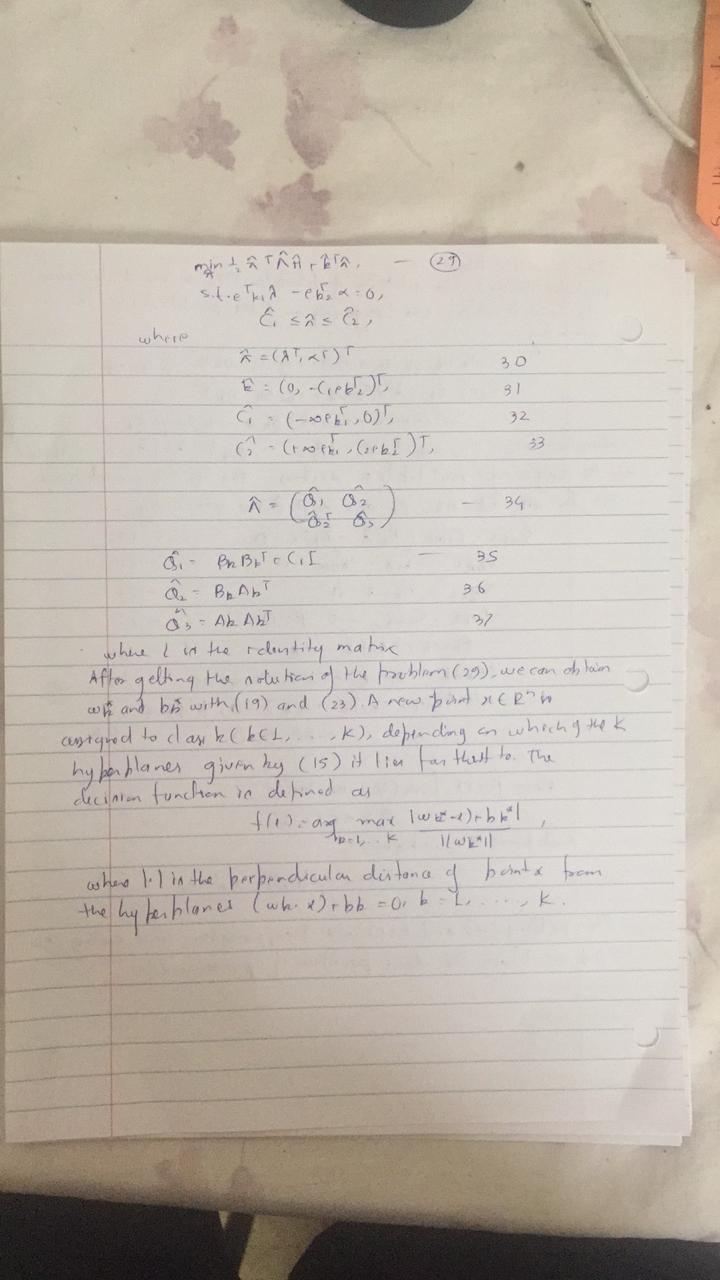
**3. (Optional) Formulate the primal problem and derive the dual problem if there are multiple classes.**

**Ans: The solution remains more and less the same other than a few changes mentioned below:**

****

****

****

****