Homework 2

2020 Spring CSCI 5525: Machine Learning

Due on March 1st 11:59 p.m.

Please type in your info:

- **Name**:Arvind Renganathan
- **Student ID**:5595189
- **Email**:renga016@umn.edu
- **Collaborators, and on which problems:**

Homework Policy. (1) You are encouraged to collaborate with your classmates on homework problems, but each person must write up the final solutions individually. You need to fill in above to specify which problems were a collaborative effort and with whom. (2) Regarding online resources, you should **not**:

- Google around for solutions to homework problems,
- Ask for help on online.
- Look up things/post on sites like Quora, StackExchange, etc.

Submission. Submit a PDF using this LaTeX template for written assignment part and submit .py files for all programming part. You should upload all the files on Canvas.

Written Assignment

Instruction. For each problem, you are required to write down a full mathematical proof to establish the claim.

Problem 1. Separability.

(**3 points**) Formally show that the XOR data set (see Lecture 4 note) is not linearly separable.
Hint: A data set $\{(x_i, y_i)_{i=1}^N\}$ where $y_i \in \{-1, 1\}$ is linearly separable if $\exists \mathbf{w} \in \mathbb{R}^d$ and $\exists b \in \mathbb{R}$ s.t.

$$\text{sign}(\langle \mathbf{w}, x_i \rangle + b) = y_i \qquad \forall i$$

Your answer. Assume there exists a predictor of $\mathbf{w} = [w1 \quad w2]^\top$ and b that linearly separates the XOR data set (Assume the data set has four points $(0,0),(1,0),(0,1),(1,1)$ whose labels are $-1,+1,+1,-1,$ respectively) and classifies the data corroctly. Then it implies the following:
$1 : w1(0) + w2(0) + b = b \leq 0$
$2 : w1(1) + w2(0) + b = w1 + b > 0$
$3 : w1(0) + w2(1) + b = w2 + b > 0$
$4 : w1(1) + w2(1) + b = w1 + w2 + b \leq 0$
Summing equations (2) and (3) we get $w1+w2+2b > 0$. Compare it with equation (4) we can find that $b > 0$, and this contradict with equation (1), which is $b \leq 0$. So,by contradiction XOR data set is not Linearly separable.

Problem 2. Kernels.

(**4 points**) As you learned in the class, Kernels provide a powerful method to traverse between kernel space and feature space. Using Kernel's properties (mention the properties used):

- (**2 points**) Show that $K(x,y) = K_1(x,y)K_2(x,y)$ is a valid kernel where K_1 and K_2 are valid kernels.
- (**2 points**) Show that the function $K(x,y) = K_1(x,y)+K_2(x,y)$ is a valid kernel function where $x, y \in \mathbb{R}$.

Your answer. a)lets say K1 and K2 are kernels having feature map $\phi1$ and $\phi2$ respectively.and
$K_1(x,y)K_2(x,y) = \left(\Phi_1\left(x_1\right)^\top \Phi_1\left(y_1\right)\right)\left(\Phi_2\left(x_1\right)^\top \Phi_2\left(y_1\right)\right)$
$= \left(\sum_{n=1}^{\infty} f_n(x)f_n(y)\right)\left(\sum_{m=1}^{\infty} g_m(x)g_m(y)\right)$
$= \sum_{n=1}^{\infty} \left(f_n(x)f_n(y)\right)\left(g_m(x)g_m(y)\right)$
$= \sum_{n,m=1}^{\infty} \left(f_n(x)g_m(x)\right)\left(f_n(y)g_m(y)\right)$
$= \sum_{n,m=1}^{\infty} f_{n,m}(x)g_{n,m}(y)$
$= \Phi_3(x)\Phi_3(y)$
As $K(x,y) = \Phi_i(x)\Phi_i(y)$ for some (potentially infinite) set of basis functions is a kernal function $\Phi_3(x)\Phi_3(y)$=K(x, y)(hence proved)

b)$K(x,y) = K_1(x,y) + K_2(x,y)$
Suppose $\mathbf{K}$ is a kernel so $K_{ij} = K\left(x_i, x_j\right) \in \mathbb{R}^{n\times n}$.
Let For any vector $\mathbf{c} \in \mathbb{R}^n$:

$$\begin{aligned}
\mathbf{c}^\top \mathbf{K}\mathbf{c} &= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i K\left(x_i, x_j\right) c_j \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n} c_i \left(K_1\left(x_i, x_j\right) + K_2\left(x_i, x_j\right)\right) c_j \\
&= \underbrace{\sum_{i=1}^{n}\sum_{j=1}^{n} c_i K_1\left(x_i, x_j\right) c_j}_{\geq 0} + \underbrace{\sum_{i=1}^{n}\sum_{j=1}^{n} c_i K_2\left(x_i, x_j\right) c_j}_{\geq 0} \\
&\geq 0
\end{aligned}$$

Therefore, $K(x,y) = K_1(x,y) + K_2(x,y)$ is a valid kernel.

Problem 3. Derivation of SVM Solution.

(**5 points**) In the lecture, we derived the soft-margin SVM solution without the intercept term. Now derive the solution with the intercept term b by going through Lagrange duality. Here the predictor will be $\hat{f}(x) = \text{sign}(\mathbf{w}^\intercal x + b)$. You should use the same conventions for the scalar and vector variables as used in the course.

- What is the Lagrange dual objective?
- State the two relevant KKT conditions: stationarity and complementary slackness.
- What is the condition for which one would get support vectors?
- What is the optimum **w** and b?

Note: Be concise in answering the questions above.

Your answer. $\hat{f}(x) = \text{sign}\left(\mathbf{w}^\top x + b\right)$
Primal problem:

$$\min_{b,\mathbf{w},\boldsymbol{\xi}} \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^{N}\xi_i$$

such that: $y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i
$\xi_i > 1-$ misclassification $\xi_i > 0-$ the data is correctly classified but lies in the margin.
With lagrangian multiplieres we get Lagrangian Dual Objective:
Lagrange dual objective:

$$\mathcal{L}(b,\mathbf{w},\boldsymbol{\xi},\boldsymbol{\lambda},\boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} + C\sum_{i=1}^{N}\xi_i + \sum_{i=1}^{N}\lambda_i\left(1 - \xi_i - y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right)\right) + \sum_{i=1}^{N}\alpha_i\left(-\xi_i\right)$$

subject to $\lambda_i \geq 0$ and $\alpha_i \geq 0$ for all i
Complementary slackness

$$\mathcal{L}(b,\mathbf{w},\boldsymbol{\xi},\boldsymbol{\lambda},\boldsymbol{\alpha}) = \frac{1}{2}\mathbf{w}^\top\mathbf{w} = 0, \sum_{i=1}^{N}\lambda_i\left(1 - \xi_i - y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right)\right) = 0, \sum_{i=1}^{N}\alpha_i\left(-\xi_i\right) = 0$$

so from above support vectors can be calculated by using value of x and y where
$\sum_{i=1}^{N}\lambda_i\left(1 - \xi_i - y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right)\right) = 0$
Apply Stationarity so $\Delta_w\mathcal{L}(b,\mathbf{w},\boldsymbol{\xi},\boldsymbol{\lambda},\boldsymbol{\alpha}) = \Delta_w(\frac{1}{2}\mathbf{w}^\top\mathbf{w}+C\sum_{i=1}^{N}\xi_i+\sum_{i=1}^{N}\lambda_i\left(1 - \xi_i - y_i\left(\mathbf{w}^\top\mathbf{x}_i + b\right)\right)+$
$\sum_{i=1}^{N}\alpha_i\left(-\xi_i\right)) = 0$
From $\frac{\partial L}{\partial w} = 0$ we get
$\mathbf{w} = \sum_{i=1}^{N} y_i\lambda_i x_i$

From $\frac{\partial L}{\partial \xi_i} = 0$ we get
$C - \lambda_i - \alpha_i = 0$

From $\frac{\partial L}{\partial b} = 0$ we get
$\sum_{i=1}^{N}\lambda_i y_i = 0$

So as we can see optimum w can be calculated by
$\mathbf{w} = \sum_{i=1}^{N} y_i \lambda_i x_i$

and b by
$\sum_{i=1}^{N} \lambda_i y_i = 0$
which we get from support vector calculation as $b = y_i - w^T X_i$
The dual Can be defined as $L(\mathbf{w}, \sigma, \lambda, \alpha) = \sum_i \lambda_i - \frac{1}{2} \sum_{i,j \in [n]} \lambda_i \lambda_j y_i y_j x_i^\top x_j$ such that $c = \lambda_i + \alpha_i$ and $\lambda_i, \alpha_i >= 0$

Programming Assignment

Instruction. For each problem, you are required to report descriptions and results in the PDF and submit code as python file (.py) (as per the question).

- **Python** version: Python 3.
- Please follow PEP 8 style of writing your Python code for better readability in case you are wondering how to name functions & variables, put comments and indent your code
- **Packages allowed**: numpy, pandas, matplotlib, cvxopt
- **Submission**: Submit report, description and explanation of results in the main PDF and code in .py files.
- Please PROPERLY COMMENT your code in order to have utmost readability
- Please provide the required functions for each problem.
- There shall be NO runtime errors involved.
- There would be PENALTY if any of the above is not followed

Problem 4. Linear SVM dual form.

(**15 Points**)
SVM can be implemented in either primal or dual form. In this assignment, your goal is to implement a linear SVM in dual form using slack variables. **The quadratic program has a box-constraint on each Lagrangian multiplier α_i, $0 \leq \alpha_i \leq C$.**

For doing this you would need an optimizer. Use an optmizer cvxopt which can be easily installed in your environment either through pip or conda. You can refer to the cvxopt document for more details about quadratic programming: https://cvxopt.org/userguide/coneprog.html#quadratic-programming.

1. (**7 Points**) Implement SVM.

   Please at least include the following functions:

   - weight = svmfit(X, y, c) to train your model
   - label = predict(X, weight) to predicts the labels using the model
   - train_accuracy, cv_accuracy, test_accuracy = k_fold_cv(train_data, test_data, k, c) to realize the k-fold cross validation

   You have to use this ”hw2data.csv” dataset for this assignment. The labels are in the last column. It is a 2 class classification problem. Split this dataset into 80-20 % split and hold out the last 20% to be used as test dataset. Then, you have to implement k=10 fold cross validation on the first 80% of the data as split above.

   Report the test performance (performance on held out test data) of your trained model for the following cases and provide your reasoning (describing the result is not explanation but you must explain the variation ’why’ the result is the way it is):

2. (**2 Points**) Summarize and explain the methods and equations you implemented.

3. (**3 Points**) Vary C in the range [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000] and report the average train, validation and test accuracy as C varies. Provide the plots.

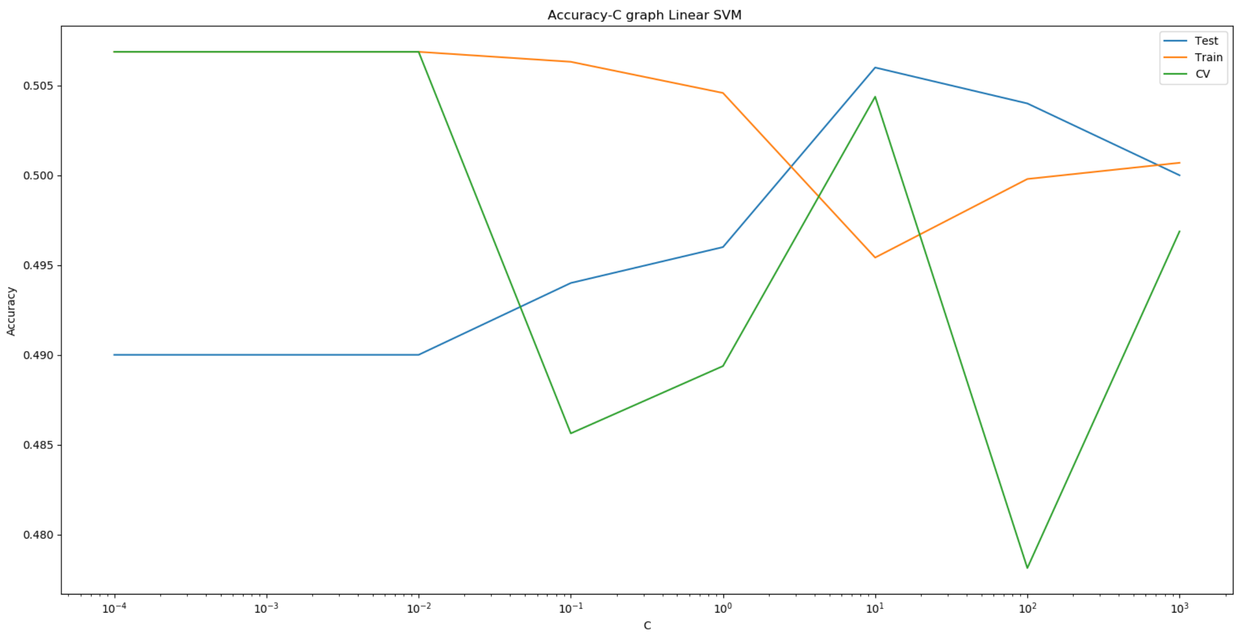
4. (**3 Points**) Explain the train, validation and test performance with C? Reason about it. As a designer, what value of C (and on what basis) would you choose for your model - explain.

Submission: Submit all plots requested and generated while performing hyperparameter tuning and explanations in latex PDF. Submit your program in a file named (hw2_svm.py).

Your answer. 2.The program was implemented using Soft Margin SVM , as was mentioned in the assignment cvoxpt was used as the convex optimization tool to solve Lagrangian dual problem for soft margin svm. Equation used were
$b = y_i[0] - w^\top X_i[0]$ $y_i = w^\top X_i + b$ where optimized w is w $= \sum_{i=1}^{N} y_i \lambda_i x_i$

The above w value results in our y label
$y_j = \sum_{i=1}^{N} y_i \lambda_i x_i^\top x_i + b$ methods implemented were svmfit to train our model where it returns the weight vector and intercept term. In predict i predict the label using given weight vector and the incoming X value.

3.



4.
So we can see in general as C becomes larger accuracy increases it is because as C increase we penalize margin violation more leading to thinner margin and greater accuracy but as we can see after 10 it starts going down again signifying that out model is overfitting as we can see from training error going down. So the best C would be a c between 1 and 10.

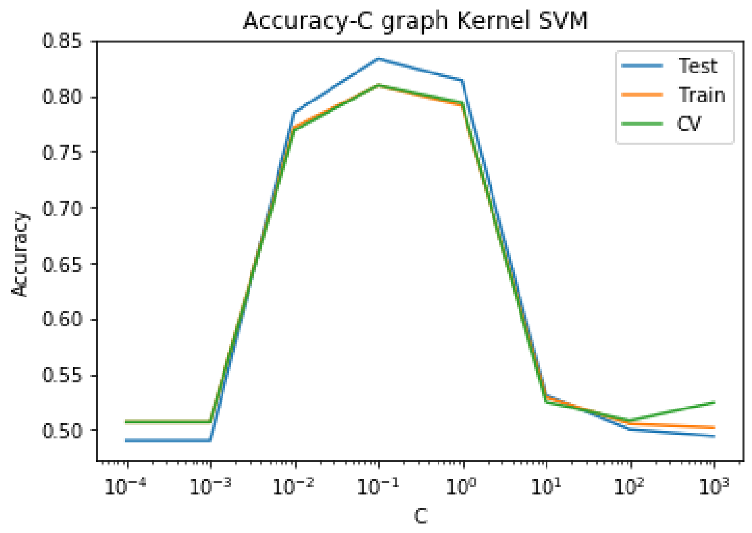
Problem 5. Kernel SVM.

(**10 Points**)

1. (**7 Points**)Implement a Kernel SVM for a generic kernel. Please at least include the following functions:
   - α = rbf_svm_train(X, y, c, σ) to train your model
   - label = predict(test_X, train_X, train_y, α, σ) to predicts the labels using the model
2. (**3 Points**)Now use the linear SVM implemented in Problem 4 and RBF-SVM (by making use of aforementioned Kernel SVM code in part 1) on the data set (”hw2data.csv”). Implement rbf_svm_train and rbf_svm_predict functions. Use the cross validation similar to Problem 4 to select the best hyper-parameters. Please report the error rate as C and σ vary. Since RBF kernel involves the second hyper-parameter you may consider using heat map to plot it. Then, report validation and test error for each fold - plot it. Compare the accuracies achieved using linear SVM and RBF-SVM, briefly explain the difference.

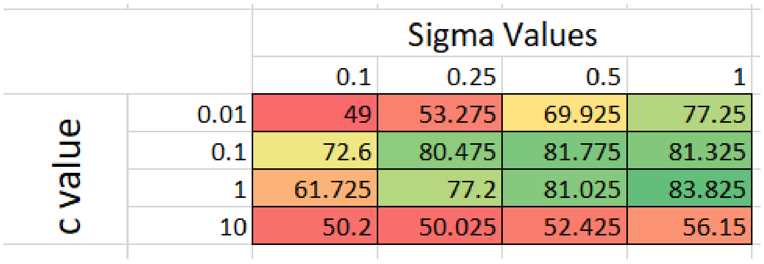
Submission: Submit all plots requested and generated and explanations in latex PDF. Submit your program in files named (hw2_kernel_svm.py).

Your answer. 2.
Accuracy vs C graph for sigma = 0.25



From the above graph we can see that accuracy rates for rbf kernal svm are significantly better than plain svm.This is because the RBF kernel does the feature transformation and increases the model complexity. Therefore, it would achieve better accuracy than linear model (moreover, because this is a not a linear separable data set, the difference is more obvious).

HeatMap of accuracy for different C and σ We can see from the above heatmap that as sigma



		Sigma Values			
		0.1	0.25	0.5	1
c value	0.01	49	53.275	69.925	77.25
	0.1	72.6	80.475	81.775	81.325
	1	61.725	77.2	81.025	83.825
	10	50.2	50.025	52.425	56.15

increases our accuracy goes up at the same time upto an extant which is mostly till C =1 accuracy also goes up gradually on increasing C it starts overfitting after that value.

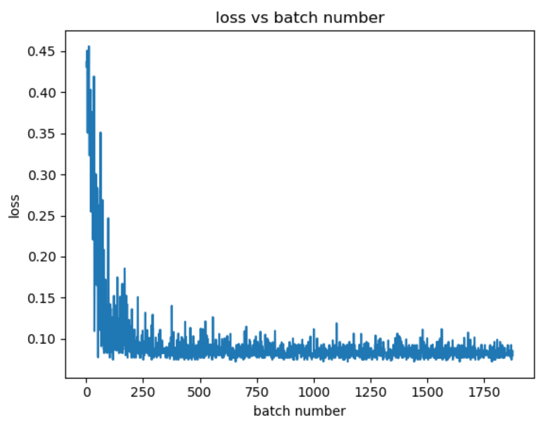
Problem 6. Multi-class logistic regression (Soft-max regression).

(**13 Points**) This problem is about multi class classification and you will use MNIST dataset. In the hw2 folder, we have put the mnist files in csv format. There are two .zip files: mnist_train and mnist_test. Use mnist_train for training your model and mnist_test to test. First column in these files are the digit labels.

1. (**7 Points**) Implement a multi-class logistic regression for classifying MNIST digits. Refer the class notes on classifying multi classes. You should use mini-batches for training the model. Save the final trained weights of your model in a file and submit it. Please at least include the following functions:
   - weight = mnist_train(X, y, learning_rate) to train your model
   - label = mnist_predict(weight, X) to predicts the labels using the model
2. (**4 Points**) Report and briefly explain the loss function, how you train update weights and how you train with mini-batches progresses. Please describe the pseudocode of mini-batches progresses and show the derivation of loss and gradient.
3. (**2 Points**) Report confusion matrix and accuracy for the test data.

Submission: Submit all plots requested and generated along with explanations in latex PDF. Submit your program in files named (hw2_multi_logistic.py).

Answer 6 2)The loss function implemented here is the cross-entropy loss function. As shown in the figure, the loss function converges to a steady value. There are some noise in the curve due to the stochastic process. It is supposed to theoretically reduce when larger batch size comes into play.



PseudoCode
Basic process is the following

```
1.First we assume an initial set of model parameter theta and decide on total no of epochs to run
2.Next for each epoch we create mini batchs
3.For each mini batch we do the following steps
4.We make prediction on mini batch.
5.Using the above prediction we compute error in prediction from our existing model parameters
6.Next we compute gradient for the mini batch partially w.r.t theta
7.using the gradient value we update theta/weight by following equation
8.theta = theta - learningrate*gradient.
9.after updating we move back to next epoch and follow from step 2 again until we finish with no of epochs.
```

3)Confusion matrix is the following

```
[[ 910    0    8    0    0   27   27    1    7    0]
 [    0 1025   29    1    1   20    4    0   55    0]
 [  30   15  861   14   14    2   36   14   42    4]
 [  18   10   62  719    0   76   15   11   87   12]
 [   7    7    7    0  738    0   36    1   24  162]
 [  41   30   22   62   18  626   32    9   28   24]
 [  40    9   26    0    8   21  848    0    6    0]
 [  11   30   47    0    9    0    4  829   28   69]
 [  18    3   27   33    7   29   20    8  803   26]
 [  24    6   15    6   50   11    5   18   34  840]]
```

Accuracy was found to be 81.9981199819982%

Problem 7. Multi-class SVM classifier.

(**10 Points**) This problem is about multi-class classification using SVM and you will use mfeat dataset, which contains descriptors from MNIST for reducing the data dimensionality. There are 64 attributes and label y in mfeat_train.csv and mfeat_test.csv files.

One way to let binary classifier be capable to deal with multi-class classification task is One v.s. All strategy. The pseudocode is shown below:

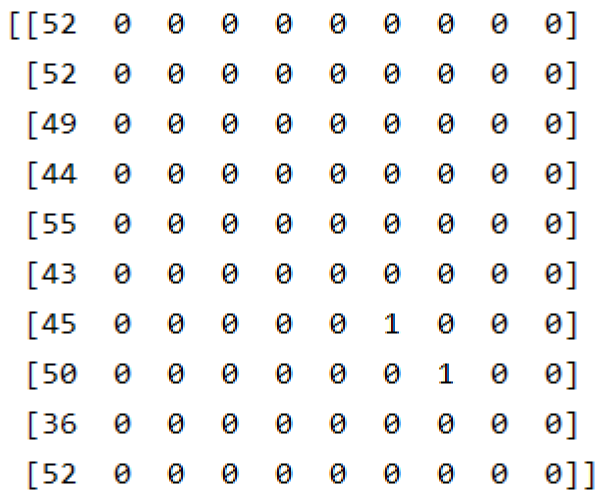
Algorithm 1: One-versus-All

```
  Input: Training set: $S = (\mathbf{x}_i, y_i)$, Binary classification algorithm $A$
  Output: The multicalss hypothesis defined by $h(\mathbf{x}) \in \text{argmax}_{i \in \mathcal{Y}} h_i(\mathbf{x})$
1 for each $i \in \mathcal{Y}$ do
2     $S_i \leftarrow (\mathbf{x}_j, (-1)^{\mathbb{1}[y_j \neq i]})$.
3     $h_i \leftarrow A(S_i)$
4 return $h(\mathbf{x}) \in \text{argmax}_{i \in \mathcal{Y}} h_i(\mathbf{x})$
```

1. (**7 Points**) Implement a multi-class SVM for classifying MNIST digits using One v.s. All strategy. Refer the class notes on classifying multi classes. You should use kernel SVM implemented in Problem 5. Save the final trained weights of your model in a file and submit it.
   - α = mnist_svm_train(X, y, c, σ) to train your model, here α could be a multi-dimensional array or matrix
   - label = mnist_svm_predict(test_X, train_X, train_y α, σ) to predicts the labels using the model
2. (**3 Points**) Report confusion matrix and accuracy for the test data. Apply soft-max regression (in Problem 6) on this dataset. Compared with the result, does the performance improved? Please explain why or why not.

Submission: Submit all plots requested and generated along with explanations in latex PDF. Submit your program in files named (hw2_multi_svm.py).

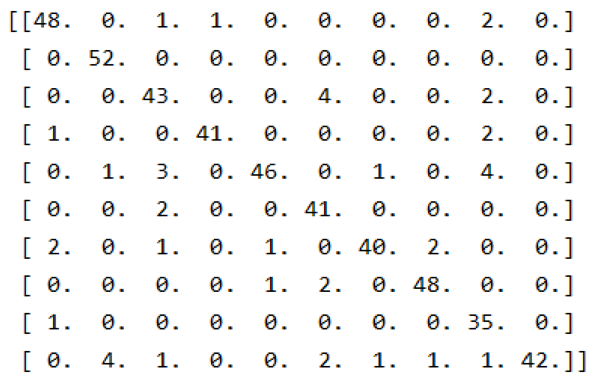
Answer 7 2) Confusion Matrix for test data for Multi class SVM is the following



```
[[52  0  0  0  0  0  0  0  0  0]
 [52  0  0  0  0  0  0  0  0  0]
 [49  0  0  0  0  0  0  0  0  0]
 [44  0  0  0  0  0  0  0  0  0]
 [55  0  0  0  0  0  0  0  0  0]
 [43  0  0  0  0  0  0  0  0  0]
 [45  0  0  0  0  0  1  0  0  0]
 [50  0  0  0  0  0  0  1  0  0]
 [36  0  0  0  0  0  0  0  0  0]
 [52  0  0  0  0  0  0  0  0  0]]
```

Accuracy is 11.25 %

Confusion Matrix for test data for Softmax reg. is the following



```
[[48.  0.  1.  1.  0.  0.  0.  0.  2.  0.]
 [ 0. 52.  0.  0.  0.  0.  0.  0.  0.  0.]
 [ 0.  0. 43.  0.  0.  4.  0.  0.  2.  0.]
 [ 1.  0.  0. 41.  0.  0.  0.  0.  2.  0.]
 [ 0.  1.  3.  0. 46.  0.  1.  0.  4.  0.]
 [ 0.  0.  2.  0.  0. 41.  0.  0.  0.  0.]
 [ 2.  0.  1.  0.  1.  0. 40.  2.  0.  0.]
 [ 0.  0.  0.  0.  1.  2.  0. 48.  0.  0.]
 [ 1.  0.  0.  0.  0.  0.  0.  0. 35.  0.]
 [ 0.  4.  1.  0.  0.  2.  1.  1.  1. 42.]]
```

Accuracy is 90.83333333333333%

Yes as we can see the performance has improved because in rbf kernal case we are using something akin to a sigmoid kernal but in the other solution hat that multi class logistic we are using Softmax.With softmax it is ensured that the sum of outputs along channels (as per specified dimension) is 1 i.e., they are probabilities. Sigmoid just makes output between 0 to 1.