Homework 2

CSCI 5525: Machine Learning

Due on Oct 10th 11am (before class)

Please type in your info:

- **Name**: Chih-Tien Kuo
- **Student ID**: 5488927
- **Email**: kuo00013@umn.edu
- **Collaborators, and on which problems**: Wei-Yu Chen, Problem 7

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.

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

Answer

Proof. Assume there exists a predictor of $\mathbf{w} = \begin{bmatrix} w1 & w2 \end{bmatrix}^\intercal$ 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 correctly. Then it implies the following:

$$point1 : w1(0) + w2(0) + b = b \leq 0 \tag{1}$$
$$point2 : w1(1) + w2(0) + b = w1 + b > 0 \tag{2}$$
$$point3 : w1(0) + w2(1) + b = w2 + b > 0 \tag{3}$$
$$point4 : w1(1) + w2(1) + b = w1 + w2 + b \leq 0 \tag{4}$$

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, 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 $x, y \in \mathbb{R}$.

Answer

(a) Suppose $\mathbf{K}$ is a gram matrix which $K_{ij} = K(x_i, x_j) \in \mathbb{R}^{n \times n}$. For any vector $\mathbf{c} \in \mathbb{R}^n$:

$$\begin{aligned}
\mathbf{c}^\intercal \mathbf{K} \mathbf{c} &= \sum_{i=1}^{n} \sum_{j=1}^{n} c_i K(x_i, x_j) c_j \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} c_i (K_1(x_i, x_j) K_2(x_i, x_j)) c_j \\
&= \sum_{i=1}^{n} \sum_{j=1}^{n} c_i (K_1(x_i, x_j) (\sum_{k=1}^{n} R_{ki} R_{kj})) c_j \\
&\text{Because } K_2(x_i, x_j) \text{ is symmetric, it can be factorized into } R^\intercal R \\
&= \sum_{k=1}^{n} \sum_{i=1}^{n} \sum_{j=1}^{n} (R_{ki} c_i) K_1(x_i, x_j) (R_{kj} c_j) \\
&= \sum_{k=1}^{n} \underbrace{(\mathbf{Rc})^\intercal \mathbf{K_1} (\mathbf{Rc})}_{\geq 0} \\
&\geq 0
\end{aligned}$$

(b) Suppose $\mathbf{K}$ is a gram matrix which $K_{ij} = K(x_i, x_j) \in \mathbb{R}^{n \times n}$. For any vector $\mathbf{c} \in \mathbb{R}^n$:

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

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 $\mathbf{w}$ and b?

Note: Be concise in answering the questions above.

Answer primal problem:

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

such that: $y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

Lagrange dual objective:

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

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

Applying stationarity:

1.

$$\frac{\partial \mathcal{L}}{\partial \xi_i} = 0 \Rightarrow C - \lambda_i - \alpha_i = 0$$

We can remove α_i by replacing $\alpha_i = C - \lambda_i$ to the constraint on dual variable, and the constraint becomes $C - \lambda_i \geq 0$ and $\lambda_i \geq 0 \Rightarrow 0 \leq \lambda_i \leq C$. We can also eliminate slack variable $\boldsymbol{\xi}$, since that $\sum_{i=1}^{N}(C - \lambda_i - \alpha_i)\xi_i = 0$. Therefore, the simplified dual objective now becomes:

$$\mathcal{L}(b, \mathbf{w}, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{w}^\intercal\mathbf{w} + \sum_{i=1}^{N}\lambda_i(1 - y_i(\mathbf{w}^\intercal\mathbf{x}_i + b))$$

subject to $0 \leq \lambda_i \leq C$ for all i

2.

$$\frac{\partial \mathcal{L}}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N}\lambda_i y_i = 0$$

From the equation above, we can now remove b from the dual objective because $\sum_{i=1}^{N}\lambda_i y_i b = 0$. Then the dual objective can be further simplified to the following:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}) = \frac{1}{2}\mathbf{w}^\intercal\mathbf{w} + \sum_{i=1}^{N}\lambda_i(1 - y_i(\mathbf{w}^\intercal\mathbf{x}_i))$$

subject to $0 \leq \lambda_i \leq C$ for all i, and $\sum_{i=1}^{N}\lambda_i y_i = 0$

3.

$$\frac{\partial \mathcal{L}}{\partial w_i} = 0 \Rightarrow w_i - \sum_{i=1}^{N}\lambda_i y_i x_{ij} = 0$$

From the equation above, we can get the $\boxed{\text{optimal } \mathbf{w} = \sum_{i=1}^{N}\lambda_i y_i \mathbf{x}_i}$, and the dual objective would become:

$$\begin{aligned}
\mathcal{L}(\mathbf{w}, \boldsymbol{\lambda}) &= \frac{1}{2}\mathbf{w}^\intercal\mathbf{w} + \sum_{i=1}^{N}\lambda_i - \mathbf{w}^\intercal\mathbf{w} \\
&= -\frac{1}{2}\|\sum_{i=1}^{N}\lambda_i y_i \mathbf{x}_i\|^2 + \sum_{i=1}^{N}\lambda_i \\
&= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \mathbf{x}_i^\intercal\mathbf{x}_j + \sum_{i=1}^{N}\lambda_i
\end{aligned}$$

subject to $0 \leq \lambda_i \leq C$ for all i, $\sum_{i=1}^{N}\lambda_i y_i = 0$, and $\mathbf{w} = \sum_{i=1}^{N}\lambda_i y_i \mathbf{x}_i$

So the dual optimization becomes

$$\begin{aligned} & \max_{\boldsymbol{\lambda}} \quad -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j + \sum_{i=1}^{N}\lambda_i \\ \Leftrightarrow \quad & \min_{\boldsymbol{\lambda}} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \mathbf{x}_i^\intercal \mathbf{x}_j - \sum_{i=1}^{N}\lambda_i \\ & \text{subject to } 0 \le \lambda_i \le C \text{ for all i, } \sum_{i=1}^{N}\lambda_i y_i = 0, \text{ and } \mathbf{w} = \sum_{i=1}^{N}\lambda_i y_i \mathbf{x}_i \end{aligned}$$

Support vector is any point i with $\lambda_i > 0$, and the optimal weight $\mathbf{w} = \sum_{\lambda_i>0}\lambda_i y_i \mathbf{x}_i$, which is the linear combination of support vectors.

By applying complementary slackness, we will have:

$$\lambda_i(1 - \xi_i - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b)) = 0 \tag{5}$$

$$(C - \lambda_i)\xi_i = 0 \tag{6}$$

For any support vector i ($\lambda_i > 0$), equation (5) would become

$$1 - \xi_i - y_i(\mathbf{w}^\intercal \mathbf{x}_i + b) = 0 \Rightarrow y_i - y_i\xi_i - (\mathbf{w}^\intercal \mathbf{x} + b) = 0 \tag{7}$$

$$\Rightarrow b = y_i - y_i\xi_i - \mathbf{w}^\intercal \mathbf{x} \tag{8}$$

For any support vector s which is unbounded ($\lambda_s < C$), equation (6) would become

$$\xi_s = 0 \tag{9}$$

Then we can solve b by combining equations (8) and (9), which will give us the optimal b as follow

$$\boxed{b = y_s - \mathbf{w}^\intercal \mathbf{x}_s}$$

where point s is any unbounded support vector.

Problem 4. Softmax Regression

(**6 points**) Consider softmax regression with K classes, no bias terms, and inputs $x_i \in \mathbb{R}^d$. $w^1, ..., w^K \in \mathbb{R}^d$ denote the weight vectors corresponding to each class, and $W \in \mathbb{R}^{d\times K}$ is the weight matrix with w_k, $1 \le k \le K$, as its columns. According to softmax regression model:

$$p(Y = c|x, W) = S(W^T x)_k$$

where $S(W^T x)_k = \frac{\exp(\langle w_k, x\rangle)}{\sum_{k'}\exp(\langle w_{k'}, x\rangle)}$

1. (**2 points**) Let $v \in \mathbb{R}^d$ be some fixed vector. For $1 \le k \le K$, let $w'_k = w_k + v$ and W' is the corresponding weight matrix. Prove $S(W^T x) = S((W')^T x)$
2. (**3 points** Suppose we train a softmax regression model on a some given dataset, once by fitting all weight vectors and once by fixing $w_K = 0_d$ and fitting the remaining weight vectors. Does the likelihood of the training data differ if we use maximum likelihood estimation?
3. (**1 point**) Interpret the below ratio

$$\frac{S(W^T x)_{k1}}{S(W^T x)_{k2}}$$

for $1 \le k_1, k_2 \le K$.

Answer

1.

$$\begin{aligned} S((W')^\intercal x)_k &= \frac{\exp\left(\langle w_k + v, x_i\rangle\right)}{\sum_{k'} \exp\left(\langle w_{k'} + v, x_i\rangle\right)} \\ &= \frac{\exp\left(\langle w_k, x_i\rangle\right)\exp\left(\langle v, x_i\rangle\right)}{\sum_{k'} \exp\left(\langle w_{k'}, x_i\rangle\right)\exp\left(\langle v, x_i\rangle\right)} \\ &= \frac{\exp\left(\langle w_k, x_i\rangle\right)}{\sum_{k'} \exp\left(\langle w_{k'}, x_i\rangle\right)} \\ &= S(W^\intercal x)_k \end{aligned}$$

2. From the previous problem, we can see that adding a fixed vector to every w doesn't affect the softmax function. Fixing w_K to 0 is the same as adding $-w_K$ to every w_j where $j \in \{1, 2, ..., K\}$. So the likelihood doesn't change.

3. The ratio implies how more likely would the label be k_1 over k_2. If the ratio is larger than 1, it means that the predicted label would have higher probability to be k_1 rather than k_2, and so on.

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

Problem 5. Linear SVM dual form.

(**20 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, $0 <= \alpha_i <= C$.**

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.

1. (**10 Points**) Implement SVM. Write a training function 'svmfit' to train your model and a prediction function 'predict' that predicts the labels using the model.

   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. (**4 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. (**4 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).

Answer

1. Please see the submitted file hw2_svm.py.
2. I trained the model using soft-margin SVM, and used the quadratic programming function from cvxopt to solve the Lagrange dual problem. The equations I used for solving $\mathbf{w}$ and b was derived in problem 3.
3. Below are the tables of average train, validation, test accuracy

lin_svm_train

C	accuracy
0.0001	0.505625
0.001	0.505625
0.01	0.5179166666666666
0.1	0.559375
1	0.56875
10	0.5685416666666667
100	0.5685416666666667
1000	0.5685416666666667

Figure 1: Train accuracy

lin_svm_valid

C	accuracy
0.0001	0.505625
0.001	0.505625
0.01	0.49874999999999997
0.1	0.5362500000000001
1	0.54375
10	0.54375
100	0.54375
1000	0.54375

Figure 2: Validation accuracy

lin_svm_test

C	accuracy
0.0001	0.485
0.001	0.485
0.01	0.495
0.1	0.5305
1	0.5389999999999999
10	0.5389999999999999
100	0.5389999999999999
1000	0.5389999999999999

Figure 3: Test accuracy

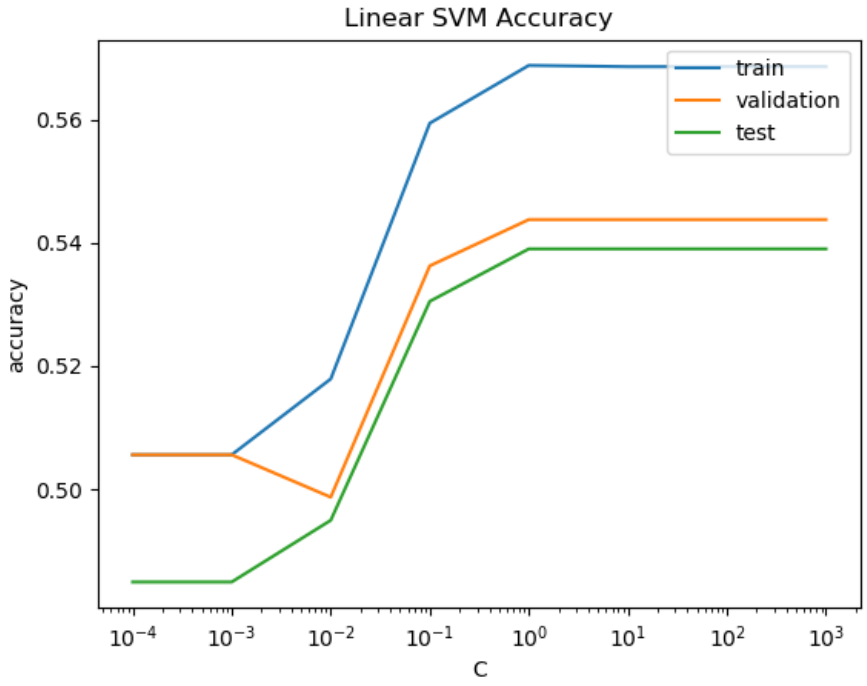

Figure 4: Linear SVM accuracy

4. As C becomes larger, the accuracy also becomes larger. This is because as C increases, the margin violation is penalized more (and you will get thinner margin). Therefore, the accuracy would increase.

   I would select C between 1 and 10 for this problem, because when C is too small, there is not enough margin penalization, and will lead to lower accuracy. However, when C is too large, there may be overfitting (it can't be seen from this problem because it's a XOR data set, but for other linear separable data set with higher noise, overfitting may happen with large C).

Problem 6. Kernel SVM.

(**7 Points**)

1. Implement a Kernel SVM for a generic kernel.
2. 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 and 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).

Answer From the tables and plot below, we can see that kernel SVM has much higher accuracy than linear 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).

kernel_svm_test

C	accuracy
0.0001	0.499
0.001	0.499
0.01	0.9727499999999999
0.1	0.9644999999999999
1	0.97575
10	0.984
100	0.99275
1000	0.9962499999999999

Figure 5: Train accuracy

kernel_svm_valid

C	accuracy
0.0001	0.475
0.001	0.475
0.01	0.9668749999999999
0.1	0.9650000000000001
1	0.974375
10	0.9806250000000002
100	0.9850000000000001
1000	0.9893750000000001

Figure 6: Validation accuracy

kernel_svm_test

C	accuracy
0.0001	0.499
0.001	0.499
0.01	0.9727499999999999
0.1	0.9644999999999999
1	0.97575
10	0.984
100	0.99275
1000	0.9962499999999999

Figure 7: Test accuracy

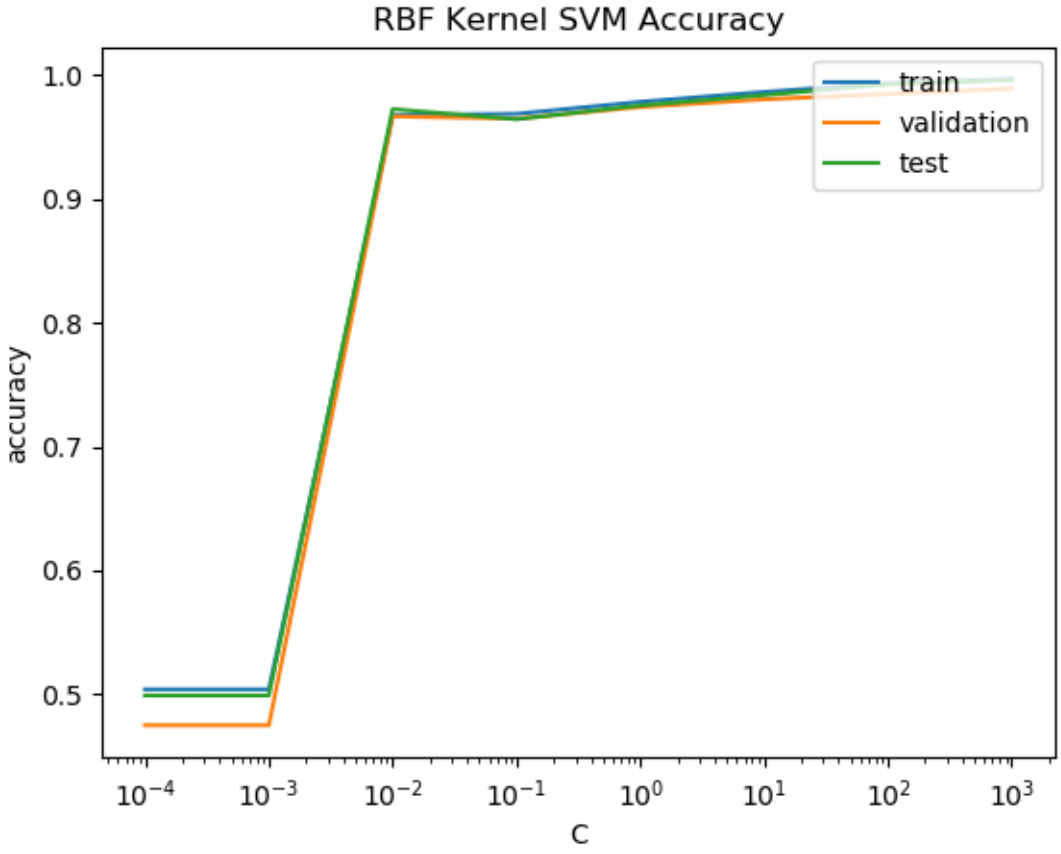

Figure 8: RBF SVM accuracy

Problem 7. Multi-class logistic regression.

(**15 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. The code should have at least two functions: multi_logistic_train and multi_logistic_predict.

2. (**3 Points**) Report and briefly explain loss function as training with mini-batches progresses.

3. (**3 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

1. Please see the submitted file hw2_multi_logistic.py.

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 will reduce with larger batch size, and vice versa.

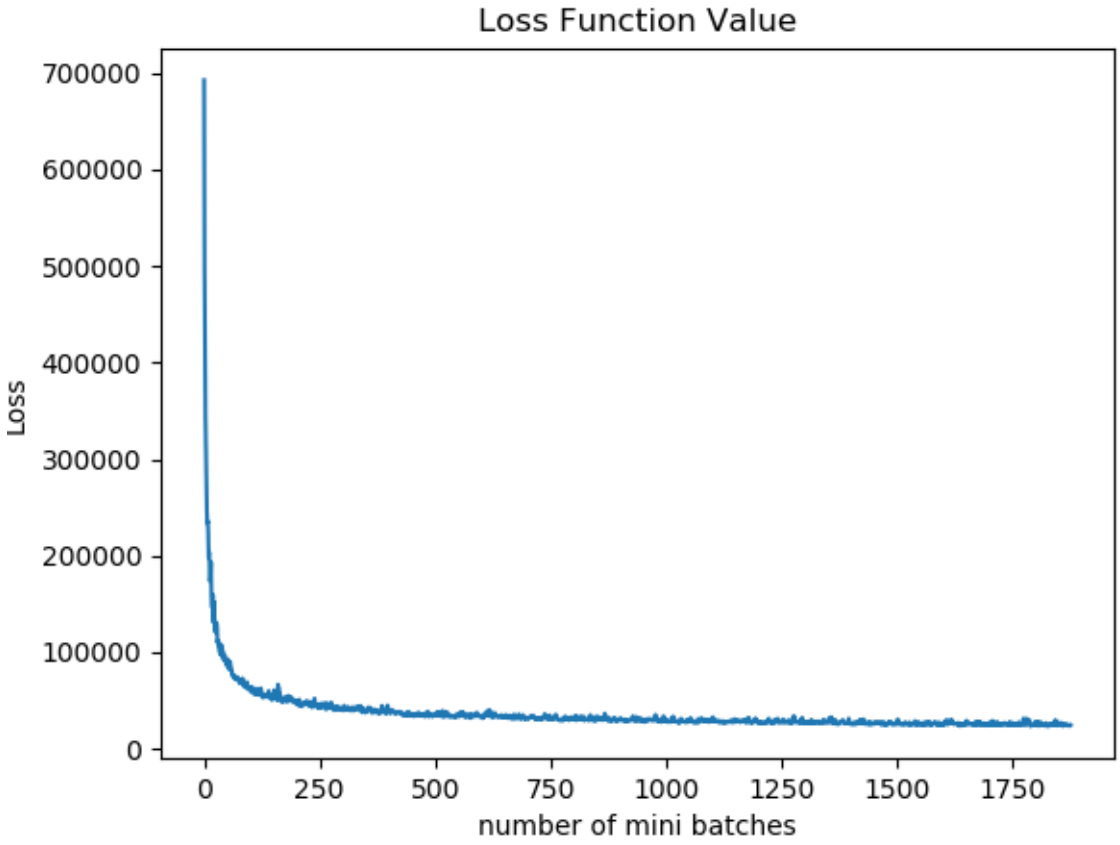

Figure 9: Loss function value

3. Confusion matrix and accuracy for train and test data are as below.

```
train accuracy: 0.8956833333333334
test accuracy: 0.8952
train data confusion matrix:
[[5693    0   31    9   13   85   40   11   30   11]
 [   2 6510   41   30    8   50   11   21   62    7]
 [  60   49 5287   99   64   47  119   78  116   39]
 [  50   30  197 4996    8  566   33   56  113   82]
 [  10   28   33    3 5218   28  126   34   42  320]
 [  82   25   66   83   47 4789  107   17  129   76]
 [  58   20   57    3   30  106 5609    8   16   11]
 [  24   38   95   30   69   16    4 5663   24  302]
 [  55  142  167  126   17  429  116   23 4665  111]
 [  37   35   23   44  147   75    7  212   58 5311]]
test data confusion matrix:
[[ 956    0    4    1    0    8    7    2    2    0]
 [   0 1101    7    2    1    4    4    3   12    1]
 [  13    6  911   22    6    4   17   14   30    9]
 [   8    0   30  828    1   87    3    8   31   14]
 [   3    6    5    1  870    4   31    7    6   49]
 [  16    4    5   19    7  787   15    7   26    6]
 [  15    3    7    2    7   21  900    1    2    0]
 [   2   11   30    7    8    2    0  924    3   41]
 [  13    9   14   18    8   70   24   13  782   23]
 [   7   10    2    8   28   16    3   30   12  893]]
```

Figure 10: RBF SVM accuracy and confusion matrix