Homework 2

CSCI 5525: Machine Learning

Due on Oct 10th 11am (before class)

Please type in your info:

- **Name**: Jiemin Yang
- **Student ID**: 5481181
- **Email**: jiemi001@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. The XOR data set is + (0,1), +(1,0),-(1,1),-(0,0). So according to the definition

$$w^T X_i + B < 0 \qquad Y_i = -1 w^T X_i + B > 0 \qquad Y_i = 1$$

So if we define + represent y=1, - represent y=1. Then

$$\begin{cases} b < 0 & (1) \\ w_1 + w_2 + b < 0 & (2) \end{cases} \begin{cases} w_1 + b > 0 \\ w_2 + b > 0 \rightarrow w_1 + w_2 + 2b > 0 & (3) \end{cases}$$

Combine (2),(3)

$$\rightarrow -2b < w_1 + w_2 < -b \rightarrow -2b < -b \quad (*)$$

Since from (1) we get b¡0, so it's unlikely $-2b < -b$, in this case, we conclude that the XOR data set is not linear 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. **1** Since $k_1(x, y), k_2(x, y)$are valid kernel,

$$k_1(X, Y) = \Phi_1(x)^t\Phi_1(y) k_2(X, Y) = \Phi_2(x)^t\Phi_2(y) k_1(X, Y)k_2(X, Y) = \Phi_1(x)^t\Phi_1(y)\Phi_2(x)^t\Phi_2(y) = \sum_i \Phi_1(x_i)^t\Phi_1($$

Since $K(x, y)$ is the inner product of the same function, so we conclude that K(x,y) is also positive semi-definite, so it's valid kernel.

2 By the properties of positive semi-definite, we assume $k_1(x, y) = G_1$ $k_2(x, y) = G_2$. For any non-zero $C.C^tG_1C > 0, C^tG_2C > 0$. $k_1(x, y)+k_2(x, y) = C^tG_1C+C^tG_2C = C^t(G_1+G_2)C > 0$. In another word $k_1(x, y)+k_2(x, y) = C^tGC = k(x, y) > 0$ Which satisfy the definition of positive semi-definite matrix. There $k_(x, y)$is also 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. setting soft-margin SVM

$$\min_w \frac{1}{2}\|w\|^2 + c\sum_{i=1}^{n} \varepsilon_i \begin{cases} \forall y_i(w^t x_i + b)1 - \varepsilon_i \\ \forall \varepsilon_i > 0 \end{cases} \implies \begin{cases} \forall 1 - \varepsilon_i - y_i(w^t x_i + b)0 \\ \forall - \varepsilon_i 0 \end{cases}$$

1dual objective:

$$L(w^*, \varepsilon^*, b^*, \lambda^*, \alpha^*) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{n} \lambda_i(1 - \varepsilon_i - y_i(w^T x_i + b)) - \sum_{i=1}^{n} \alpha_i \varepsilon_i + c\sum_{i=1}^{n} \varepsilon_i$$

According to stationnarity:

$$\begin{cases} \frac{\partial L}{\partial \varepsilon} : -\lambda_i - \alpha_i + c = 0 \to c = \lambda_i^* + \alpha^* \\ \frac{\partial L}{\partial w} : w^* - \sum_{i=1}^{n} \lambda^* y_i x_i = 0 \to w^* = \sum_{i=1}^{n} \lambda_i^* y_i x_i \\ \frac{\partial L}{\partial b} : -\sum_{i=1}^{n} \lambda_i^* y_i = 0 \to \sum_{i=1}^{n} \lambda_i^* y_i = 0 \end{cases}$$

Complementary slackness:

$$\sum_{i=1}^{n} \lambda_i^*(1 - \varepsilon_i^* - y_i(w^{T*} x_i + b^*)) = 0 \quad \sum_{i=1}^{n} \alpha_i^* \varepsilon_i^* = 0$$

Support point would be points where $\lambda_i^* > 0$ which requires $1 - \varepsilon_i^* - y_i(w^{T*} x_i + b^*) = 0$. Plug in $\lambda_i^*(> 0)$ we get:

$$w^* = \sum_{i=1}^{n} \lambda_i^* y_i x_i (\lambda_i^* > 0). b^* = \frac{1 - \varepsilon_i}{y_i} - w^{T*} x_i$$

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, $1 \leq k \leq K$, 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 $1 \leq k \leq K$, 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 $1 \leq k_1, k_2 \leq K$.

1

$$S((W')^T x)_k = \frac{exp((w_k+v)^t x)}{\sum_j exp(w+v)^T x} = \frac{exp(v^t x) exp(w^t x)}{exp(v^t x)\sum_j exp(w_j^t x)} = \frac{exp(w^t x)}{\sum_j exp(w_j^t x)} = S(w^t x)_k$$

Thus $S(W^T x) = S((W')^T x)$.

2 Suppose $v = -w_k$ then $w' = w + v \leftrightarrow$ fitting $w_k = 0_d$, from the proof of problem 1, $S(W^T x) = S((W')^T x)$, so the probability matrix would remain the same, thus the likelihood of training data would not differ when use maximum likelihood estimation.

3 It represents the relative risk(probability ratio) of labeled as $k1$ and labeled as $k2$ when features $\mathbf{x}$hold.

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).

test.png train.png validation.png

Your answer. 2
Used soft margin SVM to maximize margin when find hyper plane. Mainly solved in dual space by converting the minimization to a maximization problem be constructing the Langrangian and then using the KKT conditions. And then solve dual optimization problem to find support vector so that get w By solving

$$\max_{\alpha\,\lambda} \sum_i \lambda_i - \frac{1}{2} \sum_{i,j \in [n]} \lambda_i \lambda_j y_i y_j x_i^T x_j such that for all i : C = \lambda_i + \alpha_i \lambda_i, \alpha_i 0$$

we get support point i with $\lambda_i > 0$. Then we plug in this point with corresponding λ_i, we get w^* by solving $w^* = \sum_{i:\lambda_i^* > 0} y_i \lambda^* x_i$.

3
for c in range [0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]
test accuracy:[0.494 0.49 0.492 0.492 0.488 0.492 0.492] tra$\in$ *accuracy* : [0.5160.5150.5170.5160.5140.5170.515]
validationaccuracy : [0.4980.5070.5120.4970.5120.4970.515]

graphicx
4
On average ,train set has the highest accuracy,test set has the lowest accuracy which make sense. Cause model trained in train set.
The choice of C result slightly different accuracy but it doesn't vary much. It would be better to choose C between 0.001, 0.01, since the result it better in all of the 3 set.

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 4part 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).

Your answer. validation mean accuracy: 0.4790487421383648
test mean accuracy: 0.5089999999999999
train mean accuracy:0.48641514168790045
the result of kenerl it's slightly better than linear SVM, cause it's expand the feature thus capture more information of features. So it's predict more accurate.

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).

Your answer. Loss function The multi-classification problem using cross-entropy loss which compare between true label vector(probability) with estimated probability vector.

$$l_c e(\tilde{y}, \hat{y}) = -f(x)_y + ln \sum_{j=1}^{k} exp(f(x)_j)$$

So for mini batches process, firstly shuffled the data set and then, I separate data in to 59 pieces with 1024 samples for each mini-batches. Then sequentially calculating gradient descent using mini-batches data to train the model. Each 59 laps recorded as a epoch, executing 100 epoch to get final W.

accuracy:0.0087

```
[[0.000e+00 1.109e+03 1.100e+01 2.000e+00 2.000e+00 3.000e+00 2.000e+00 5.000e+00 9.000e+00
8.000e+00]
e+00 1.100e+01 9.300e+02 3.500e+01 1.300e+01 6.000e+00 9.000e+00 4.000e+01 1.600e+01
3.000e+00

e+00 0.000e+00 6.000e+00 8.460e+02 3.000e+00 1.500e+01 0.000e+00 8.000e+00 8.000e+00
9.000e+00

e+00 0.000e+00 5.000e+00 0.000e+00 8.700e+02 1.000e+01 7.000e+00 5.000e+00 7.000e+00
2.400e+01

e+00 2.000e+00 3.000e+00 7.300e+01 1.000e+00 7.650e+02 1.800e+01 3.000e+00 2.900e+01
1.700e+01

e+01 3.000e+00 1.100e+01 6.000e+00 1.900e+01 1.600e+01 8.960e+02 0.000e+00 1.300e+01
3.000e+00

e+00 2.000e+00 9.000e+00 1.300e+01 6.000e+00 8.000e+00 3.000e+00 9.220e+02 1.000e+01
2.500e+01
```

e+00 7.000e+00 3.600e+01 2.100e+01 2.100e+01 3.300e+01 3.000e+00 3.000e+00 8.350e+02 2.600e+01

e+02 1.000e+00 2.100e+01 1.400e+01 4.700e+01 3.600e+01 2.000e+01 4.100e+01 4.700e+01 8.940e+02

e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00 0.000e+00
]