SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

50.007 Machine Learning, Spring 2025
Homework 1

Due 28 February 2025, 11.59pm

In this homework, we would like to look at **Linear Classification** including perceptron algorithm and hinge loss, and **Regression** including linear regression and ridge regression.

# 1 Linear Classification [40 points]

## 1.1 Hinge Loss with Offset [15 points]

Hinge loss is a convex surrogate for the zero-one loss used for linear classification in non-realizable case. In our class, we have already studied hinge loss without an offset. Now, let's consider hinge loss with an offset $\theta_0$. Given a training set $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ is the feature vector and $y_i \in \{-1, +1\}$ is the corresponding label.
(1) [5 points] Please provide the definition of the hinge loss function with the offset $\theta_0$.
(2) [10 points] Describe how to update the weights $\theta$ and $\theta_0$ to minimize the hinge loss function using stochastic sub-gradient descent (SSGD). You must explicitly write out the whole procedure of SSGD.

## 1.2 Code Exercise: Linear Classification for Handwritten Digit Recognition [25 points]

In this exercise, you will work with a real-world dataset from US Postal Service Zip Code Database, which contains $16 \times 16$ pixel images of handwritten zip codes. These images are preprocessed versions of scanned handwritten zip codes and are used for automatic handwritten digit recognition. For this task, you will focus on recognizing only two digits: **1** and **5**.

To simplify the problem, you will use two features: **intensity** and **symmetry**. Digit 5 tends to occupy more black pixels, resulting in higher average pixel intensity compared to digit 1. Additionally, digit 1 is typically symmetric, while digit 5 is not. You will define **asymmetry** as the average difference between an image and its flipped version, and **symmetry** as the negation of asymmetry. This allows digit 1 to have higher symmetry values than digit 5, making it easier to distinguish between these two digits based on these features.
***Instructions***:

- The training and test datasets are provided in the files 'hw1_train_1_5.csv' and 'hw1_test_1_5.csv', respectively. Each row in the CSV files represents an example, where the first value is the symmetry of the image, the second value is the average intensity of the image, and the third value is the label.
- Ensure that the labels are converted: digit **1** becomes +1 and digit **5** becomes -1 for binary classification.

(1) [5 points] **Implement the Perceptron Algorithm for Linear Classification with Offset.** Write the code for the perceptron algorithm (with zero-one loss) to classify the two digits using the features intensity and symmetry. Ensure that your implementation includes an offset (bias term) for the linear classifier.
(2) [5 points] **Train Linear Classifier with Offset using Perceptron Algorithm and Evaluate.** Using the perceptron algorithm you implemented, train the linear classifier with offset on the training set for 1 epoch (*i.e.*, traversing the training set 1 time) and 5 epochs, respectively. For both training runs, report the following:

- The learned weights ($\hat{\theta}$)
- The learned bias ($\hat{\theta}_0$)
- The training error
- The accuracy of the model on the test set

(3) [5 points] **Implement the Stochastic Sub-Gradient Descent Algorithm for Linear Classification with Offset.** Write the code for the stochastic sub-gradient descent (with hinge loss) to classify the two digits using the features intensity and symmetry. Set your stopping criterion as the maximum number of iterations, and fix the learning rate throughout the entire training process.
(4) [5 points] **Train Linear Classifier with Offset using SSGD and Evaluate.** Using the SSGD algorithm you implemented, train the linear classifier with offset on the training set for 5000 iterations, with a learning rate of 0.1, and report the learned weights ($\hat{\theta}$), the learned bias ($\hat{\theta}_0$), the training error, as well as the accuracy of the model on the test set.
(5) [5 points] What are your observations of the differences in training errors between the Perceptron Algorithm (using 5 epochs) and SSGD (using 5000 iterations)?

## 2 Linear Regression [40 points]

In this exercise, you will experiment with linear on a given data set. The inputs are in the file 'hw1_linear_x.dat' and the desired outputs in 'hw1_linear_y.dat'. Note that the input only has one feature, and the output is also one variable.
(1) [10 points] In class, we omit $\theta_0$ for simplicity. Now let us bring $\theta_0$ back, write the cost function - least squares criterion - with offset. Additionally, write the closed-form solution and stochastic gradient descent procedure with offset. (Hint: recall what we discussed in class that how to solve $\theta_0$ together with $\theta$ by adding 1 to the input feature.)
(2) [10 points] Write a function implementing the closed form linear regression formula in your previous answer to obtain the optimal $\theta$ and $\theta_0$ and report them. Plot both the linear regression line with the optimal weights and the data on the same graph. Write a function that will evaluate the training error in terms of empirical risk of the resulting fit and report the error.
(3) [10 points] Write a function to learn the optimal $\theta$ and $\theta_0$ using gradient descent algorithm. Consider setting the learning rate $\eta$ as a constant (0.01) and update it for 5 times. Report $\theta$, $\theta_0$, and training error for the minimum empirical risk. Plot both the linear regression line and the data on the same graph.
(4) [10 points] Write a function to learn the optimal $\theta$ and $\theta_0$ using stochastic gradient descent algorithm. Use the same learning rate but run for 5 epochs (with replacement). Report the optimal $\theta$, $\theta_0$, and training error. Plot both the linear regression line and the data on the same graph.

## 3 Ridge Regression [20 points]

In this exercise, you will explore the effects of $\lambda$ in ridge regression on generalization. You will use 'hw1_ridge_x.dat' as the inputs and 'hw1_ridge_y.dat' as the desired output. Please note that a column vector of 1s is already added to the inputs. Recall from what you learned in class: to find a suitable value for $\lambda$, we set aside a small subset of the provided data set as 'validation set' for estimating the validation loss as a proxty of generalization error. Let the first 10 entries of the data set be the validation set, and the last 40 entries be the training set. Concatenate their features into matrices $X_{val}$ and $X_{train}$, and their responses into vectors $y_{val}$ and $y_{train}$.

(1) [10 points] Write a function ridge regression($X_{train}$, $y_{train}$, $\lambda$) that takes the training features, training responses and regularizing parameter $\lambda$, and outputs the exact solution $\theta$ for ridge regression. Report the resulting value of $\theta$ for $\lambda = 0.15$.

(2) [10 points] Use the following code to plot graphs of the validation loss and training loss as $\lambda$ varies on logarithmic scale from $\lambda = 10^{-5}$ to $\lambda = 10^{0}$. Write down the value of $\lambda$ that minimizes the validation loss.

```
import matplotlib.pyplot as plt
tn = Xt.shape[0]
vn = vX.shape[0]
tloss = []
vloss = []
index = -np.arange(0, 5, 0.1)

for i in index:
w = ridge_regression(tX, tY, 10**i)
tloss = tloss + [np.sum((np.dot(tX, w) - tY)**2) / tn / 2]
vloss = vloss + [np.sum((np.dot(vX, w) - vY)**2) / vn / 2]

plt.plot(index, np.log(tloss), 'r')
plt.plot(index, np.log(vloss), 'b')
```