Introduction to Machine Learning

Fall 2021

University of Science and Technology of China

Lecturer: Jie Wang | Homework 5

Posted: Nov. 30, 2021 | Due: Dec. 13, 2021

Notice, to get the full credits, please show your solutions step by step.

Exercise 1: Entropy and Uncertainty

1. The **relative entropy** of two vectors $\mathbf{u}$, $\mathbf{v} \in \mathbb{R}^n_{++}$ defined as

$$f(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{n} u_i \log\left(\frac{u_i}{v_i}\right).$$

The **Kullback-Leibler divergence** between $\mathbf{u}$, $\mathbf{v} \in \mathbb{R}^n_{++}$ is defined as

$$D_{kl}(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{n} u_i \log\left(\frac{u_i}{v_i}\right) - u_i + v_i.$$

   (a) Show that the relative entropy is convex on $\mathbb{R}^n_{++} \times \mathbb{R}^n_{++}$.

   (b) Show that D_{kl} is convex on $\mathbb{R}^n_{++} \times \mathbb{R}^n_{++}$.

   (c) Show the *information inequality*: $D_{kl}(\mathbf{u}, \mathbf{v}) \geq 0$ for all $\mathbf{u}$, $\mathbf{v} \in \mathbb{R}^n_{++}$. Also show that $D_{kl}(\mathbf{u}, \mathbf{v}) = 0$ if and only if $\mathbf{u} = \mathbf{v}$. Hence the Kullback-Leibler divergence can be used as a measure of deviation between two positive vectors.

   Hint: The Kullback-Leibler divergence can be expressed as

$$D_{kl}(\mathbf{u}, \mathbf{v}) = g(\mathbf{u}) - g(\mathbf{v}) - \nabla g(\mathbf{v})^\top (\mathbf{u} - \mathbf{v}),$$

   where $g(\mathbf{u}) = \sum_{i=1}^{n} u_i \log u_i$ is the negative entropy of $\mathbf{u}$.

   (d) Show that the relative entropy and the Kullback-Leibler divergence are the same when $\mathbf{u}$ and $\mathbf{v}$ are probability vectors, i.e., satisfy $\mathbf{1}^\top \mathbf{u} = \mathbf{1}^\top \mathbf{v} = 1$.

2. The **Shannon entropy** of a discrete random variable X defined as

$$H(X) = -\sum_{x} p_X(x) \log p_X(x),$$

where $p_X(x) = P(X = x)$ is the probability mass function of X.

The **joint entropy** of two random variables X and Y is similarly defined as

$$H(X, Y) = -\sum_{x} \sum_{y} p_{X,Y}(x, y) \log p_{X,Y}(x, y),$$

where $p_{X,Y}(x, y) = P(X = x, Y = y)$ is the joint probability mass function of X, Y.

The **conditional entropy** of two random variables X and Y is defined as

$$H_Y(X) = \sum_{y} p_Y(y) H_{Y=y}(X),$$

where $H_{Y=y}(X) = -\sum_x p_{X|Y}(x \mid y) \log p_{X|Y}(x \mid y)$.

The **mutual information** of two random variables X and Y is defined as

$$I(X,Y) = \sum_x \sum_y p_{X,Y}(x,y) \log \left(\frac{p_{X,Y}(x,y)}{p_X(x) p_Y(y)} \right).$$

(a) Show that $H(X,Y) = H(Y) + H_Y(X)$. Interpret the equality.

(b) Show that $H_Y(X) \leq H(X)$. When does the equality hold?

(c) Show that for any function $g : \mathbb{R} \to \mathbb{R}$ such that $g(X)$ is a random variable, there holds $H(g(X)) \leq H(X)$. What does this inequality mean?

(d) Show that

$$I(X,Y) = H(X) + H(Y) - H(X,Y) = H(X) - H_Y(X).$$

Interpret the equality.

Solution: ■

Exercise 2: Random Forests (Optional)

Of all the well-known learning methods, **decision trees** come closest to meeting the requirements for serving as an off-the-shelf procedure for data mining, and have emerged as the most popular learning method for data mining due to many advantages. However, trees have one aspect that prevents them from being the ideal tool for predictive learning, namely inaccuracy. They seldom provide predictive accuracy comparable to the best that can be achieved with the data at hand. In other words, they work great with the data used to create them, but they are not flexible when encountering new samples.

The good news is that **random forests** (Breiman, 2001), as shown in Algorithm 1, combine the simplicity of decision trees with flexibility resulting in a vast improvement in accuracy, thus implemented in a variety of packages.

Algorithm 1 Random Forest for Regression or Classification

1: **Input:** The training data with p attributes, the number of random-forest trees B, the size of one bootstrap sample N, the minimum node size n_{min}, and the number of attributes selected randomly each time m.

2: **for** $b = 1$ to B **do**

(a) Draw a sample Z^* of size N **with replacement** (which is called a bootstrap sample) from the training data.

(b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

i. Select m attributes at random from the p attributes.

ii. Pick the best attribute/split-point among the m.

iii. Split the node into two child nodes.

End for

3: To make a prediction at a new point x:
For *Regression:* $\hat{f}_{rf}^{B}(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x)$.
For *Classification:* Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree T_b. Then $\hat{C}_{rf}^{B}(x) = \text{majority vote}\{\hat{C}_b(x)\}_{b=1}^{B}$.

Random forests is a substantial modification of the so-called **bagging** or **bootstrap aggregation**, which builds a large collection of *de-correlated* trees, and then averages them.

1. Given B i.d. (identically distributed, but not necessarily independent) random variables with positive pairwise correlation ρ and variance σ^2, show that the variance of their average is

$$\rho\sigma^2 + \frac{1-\rho}{B}\sigma^2. \tag{1}$$

This appears to fail if ρ is negative; diagnose the problem in this case.

2. As B increases, the second term of Eq. (1) disappears, but the first remains, and hence the size of the correlation of pairs of bagged trees limits the benefits of averaging. The idea in random forests is to improve the variance reduction of bagging by reducing

the correlation between the trees, without increasing the variance too much. This is achieved in the tree-growing process through random selection of the input variables.

How does the size of m effect the correlation between any pair of random-forest trees? Intuitively explain your claim.

3. Not all estimators can be improved by the bootstrap aggregation. For example, bagging does not change *linear* estimates like the sample mean and its variance. What's more, the pairwise correlation between bootstrapped means is about 50%, but ρ is typically small for bootstrapped trees.

   Suppose $X_1, \ldots, X_N$ are i.i.d. with means μ and variances σ^2. Let $\bar{X}_1^*$ and $\bar{X}_2^*$ be two bootstrap realizations of the sample mean, i.e.,

   $$\bar{X}_i^* = \frac{1}{n} \sum_{j=1}^{n} X_j^i, \; i = 1, 2,$$

   where $\{X_j^i\}_{j=1}^n$, $i = 1, 2$ are the bootstrap samples randomly selected with replacement from $\{X_i\}_{i=1}^N$. Show that the correlation between $\bar{X}_1^*$ and $\bar{X}_2^*$ is $\frac{n}{2n-1} \approx 50\%$.

4. The leave-one-out cross-validation (LOOCV) is a special case of the cross-validation. For each instance, LOOCV uses all other instances as a training set and the selected instance as a single-item test set. For random forests, we can also use out-of-bag (OOB) samples to estimate the prediction error. Specifically, for each observation $z_i = (x_i, y_i)$, construct its random forest predictor by averaging only those trees corresponding to *bootstrap samples in which z_i did not appear*.

   Show that as the number of bootstrap samples B gets large, the OOB error estimate for a random forest approaches its leave-one-out cross-validation error estimate, and that in the limit, the identity is exact.

Solution: ■

Exercise 3: Bayes Theorem

Recall that Bayes Theorem relates conditional probabilities of the form $P(A \mid B)$ with conditional probabilities of the form $P(B \mid A)$, in which the order of the conditioning is reversed.

1. Let $A_1, A_2, \ldots, A_n$ be disjoint events that form a partition of the sample space, and assume that $P(A_i) > 0$ for all i. Then, for any event B such that $P(B) > 0$, please show that
$$P(A_i \mid B) = \frac{P(A_i)P(B \mid A_i)}{P(A_1)P(B \mid A_1) + \cdots + P(A_n)P(B \mid A_n)}.$$

2. (**The False-Positive Puzzle.**) A test for a certain rare disease is assumed to be correct 95% of the time: if a person has the disease, the test results are positive with probability 0.95, and if the person does not have the disease, the test results are negative with probability 0.95. A random person drawn from a certain population has probability 0.001 of having the disease. Given that one person was just tested positive, what is the probability that the person really had this disease?

3. Suppose you are lost in a park. Tourists comprise two-thirds of the visitors to the park, and give a correct answer to requests for directions with probability $\frac{3}{4}$. (Answers to repeated questions are independent, even if the question and the person are the same.) The others are unfriendly locals, who always give false answers when you ask for directions.

   (a) You ask a passer-by whether the exit from the Park is East or West. The answer is East. What is the probability this is correct?

   (b) You ask the same person again, and receive the same reply. What is the probability this is correct?

   (c) You ask the same person for the third time, and receive the same reply. What is the probability this is correct?

   (d) Had the third answer been West instead, what is the probability that East is correct?

   (e) Suppose that Mr. Bayes is in the same position as you were in the previous problems, but he has no reason to believe that, with probability ε, East is the correct answer. Please show that whatever answer first received, Mr. Bayes continues to believe that East is correct with probability ε.

   (f) If the first two replies are the same, i.e., either East-East or West-West, please show that Mr. Bayes continues to believe that East is correct with probability ε.

Solution: ■

Exercise 4: Programming Exercise: Naive Bayes Classifier

We provide you with a data set that contains spam and non-spam emails ("hw5_nb.zip"). Please use the Naive Bayes Classifier to detect the spam emails. Finish the following exercises by programming. You can use your favorite programming language.

1. Remove all the tokens that contain non-alphabetic characters.
2. Train the Naive Bayes Classifier on the training set according to Algorithm 2.
3. Test the Naive Bayes Classifier on the test set according to Algorithm 3. You may encounter a problem that the likelihood probabilities you calculate approach 0. How do you deal with this problem?
4. Compute the confusion matrix, accuracy, precision, recall, and F-score.
5. Without the Laplace smoothing technique, complete the steps again.

Algorithm 2 Training Naive Bayes Classifier

Input: The training set with the labels $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}$.

$\mathcal{V} \leftarrow$ the set of distinct words and other tokens found in $\mathcal{D}$
for each target value c in the labels set $\mathcal{C}$ **do**
  $\mathcal{D}_c \leftarrow$ the training samples whose labels are c
  $P(c) \leftarrow \frac{|\mathcal{D}_c|}{|\mathcal{D}|}$
  $T_c \leftarrow$ a single document by concatenating all training samples in $\mathcal{D}_c$
  $n_c \leftarrow |T_c|$
  for each word w_k in the vocabulary $\mathcal{V}$ **do**
    $n_{c,k} \leftarrow$ the number of times the word w_k occurs in T_c
    $P(w_k|c) = \frac{n_{c,k}+1}{n_c+|\mathcal{V}|}$
  end for
end for

Algorithm 3 Testing Naive Bayes Classifier

Input: An email $\mathbf{x}$. Let x_i be the i^{th} token in $\mathbf{x}$. $\mathcal{I} = \emptyset$.

for $i = 1, \ldots, |\mathbf{x}|$ **do**
  if $\exists\, w_{k_i} \in \mathcal{V}$ such that $w_{k_i} = x_i$ **then**
    $\mathcal{I} \leftarrow \mathcal{I} \cup i$
  end if
end for
predict the label of $\mathbf{x}$ by

$$\hat{y} = \arg\max_{c \in \mathcal{C}} P(c) \prod_{i \in \mathcal{I}} P(w_{k_i}|c)$$

Solution: ■

Exercise 5: Logistic Regression and Newton's Method

Given the training data $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, where $\mathbf{x}_i \in \mathbb{R}^d$ and $y_i \in \{0, 1\}$. Let

$$\mathcal{I}^+ = \{i : i \in [n], y_i = 1\},$$
$$\mathcal{I}^- = \{i : i \in [n], y_i = 0\},$$

where $[n] = \{1, 2, \ldots, n\}$. We assume that $\mathcal{I}^+$ and $\mathcal{I}^-$ are not empty.
Then, we can formulate the logistic regression of the form.

$$\min_{\mathbf{w}} \; L(\mathbf{w}) = -\frac{1}{n}\sum_{i=1}^{n}\left(y_i \log\left(\frac{\exp(\langle \mathbf{w}, \overline{\mathbf{x}}_i\rangle)}{1+\exp(\langle \mathbf{w}, \overline{\mathbf{x}}_i\rangle)}\right) + (1-y_i)\log\left(\frac{1}{1+\exp(\langle \mathbf{w}, \overline{\mathbf{x}}_i\rangle)}\right)\right), \quad (2)$$

where $\mathbf{w} \in \mathbb{R}^{d+1}$ is the model parameter to be estimated and $\overline{\mathbf{x}}_i^\top = (1, \mathbf{x}_i^\top)$.

1. (a) Suppose that the training data is strictly linearly separable, that is, there exists $\hat{\mathbf{w}} \in \mathbb{R}^{d+1}$ such that

$$\langle \hat{\mathbf{w}}, \overline{\mathbf{x}}_i\rangle > 0, \, \forall \, i \in \mathcal{I}^+,$$
$$\langle \hat{\mathbf{w}}, \overline{\mathbf{x}}_i\rangle < 0, \, \forall \, i \in \mathcal{I}^-.$$

   Show that problem (2) has no solution.

   (b) Suppose that the training data is NOT linearly separable, that is, for all $\mathbf{w} \in \mathbb{R}^{d+1}$, there exists $i \in [n]$ such that

$$\langle \mathbf{w}, \overline{\mathbf{x}}_i\rangle < 0, \text{ if } i \in \mathcal{I}^+,$$

   or

$$\langle \mathbf{w}, \overline{\mathbf{x}}_i\rangle > 0, \text{ if } i \in \mathcal{I}^-.$$

   Show that problem (2) always admits a solution.

2. Suppose that $\overline{\mathbf{X}} = (\overline{\mathbf{x}}_1, \overline{\mathbf{x}}_2, \ldots, \overline{\mathbf{x}}_n)^\top \in \mathbb{R}^{n\times(d+1)}$ and $\mathbf{rank}(\overline{\mathbf{X}}) = d + 1$. Show that $L(\mathbf{w})$ is strictly convex, i.e., for all $\mathbf{w}_1 \neq \mathbf{w}_2$,

$$L(t\mathbf{w}_1 + (1-t)\mathbf{w}_2) < tL(\mathbf{w}_1) + (1-t)L(\mathbf{w}_2), \forall \, t \in (0, 1).$$

3. In real applications, a widely-used method to learn the parameters' values of logistic regression is to solve the optimization problem in (2) with a regularization term, e.g.,

$$\min_{\mathbf{w}} F(\mathbf{w}) = L(\mathbf{w}) + \frac{\lambda}{2}\|\mathbf{w}\|_2^2, \; \lambda > 0.$$

   The Newton's method is an iterative method for optimization problems. We use the Newton's method to fit the regularized logistic regression by the following algorithm.

Algorithm 4 Newton's Method for Logistic Regression

1: **Input:** The twice-differentiable objective function $F(\mathbf{w})$, the initial point $\mathbf{w_0}$, the degree of precision ϵ.
2: Calculate the gradient $\mathbf{g}(\mathbf{w}) = \nabla F(\mathbf{w})$ and the Hessian matrix $\mathbb{H}(\mathbf{w})$ of the input $F(\mathbf{w})$.
3: **while** $\|\mathbf{g}_k(\mathbf{w}_k)\|_2 \geq \epsilon$

(a) Let $\mathbb{H}(\mathbf{w}_k) = \mathbb{H}_k$ and $\mathbf{g}(\mathbf{w}_k) = \mathbf{g}_k$ to simplify notations. Calculate the Hessian matrix $\mathbb{H}_k$, and the let $\mathbf{w}_{k+1} = \mathbf{w}_k - \mathbb{H}_k^{-1}\mathbf{g}_k$.

(b) $k = k + 1$.

(c) Calculate the gradient $\mathbf{g}_{k+1}$.

4: **Output:** $\hat{\mathbf{w}}$,the first point satisfying $\|\hat{\mathbf{g}}\|_2 < \epsilon$.

(a) Please calculate the gradient $\mathbf{g}(\mathbf{w})$ and the Hessian matrix $\mathbb{H}(\mathbf{w})$ of the regularized Logistic regression.

(b) Please show that the Hessian matrix $\mathbb{H}(\mathbf{w})$ is invertible.

(c) (Bonus) Please show the local convergence of Newton's method in logistic regression, i.e.,

$$\frac{\|\mathbf{w}_{k+1} - \mathbf{w}^*\|}{\|\mathbf{w}_k - \mathbf{w}^*\|^2} < B,$$

for some $B \in \mathbb{R}$, if the initial point is close enough to $\mathbf{w}^* = \arg\min_{\mathbf{w}} F(\mathbf{w})$.

Solution: ■

Exercise 6: Convergence of Stochastic Gradient Descent for Convex Function

Consider an optimization problem

$$\min_{\mathbf{w}} F(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}), \tag{3}$$

where the objective function F is continuously differentiable and strongly convex with convexity parameter $\mu > 0$. Suppose that the gradient of F, i.e., ∇F, is Lipschitz continuous with Lipschitz constant L, and F can attain it minimum F^* at $\mathbf{w}^*$. We use the stochastic gradient descent(SGD) algorithm introduced in Lecture 12 to solve the problem (3). Let the solution sequence generated by SGD be $(\mathbf{w}_k)$.

1. Please show that $\forall \mathbf{w} \in \mathbf{dom}\ F$, the following inequality

$$F(\mathbf{w}) - F^* \leq \frac{1}{2\mu}\|\nabla F(\mathbf{w})\|^2 \tag{4}$$

   holds, and interpret the role of strong convexity based on this.

2. Recall that with a fixed stepsize $\alpha \in [0, \frac{1}{LM_G}]$ where M_G (as well as the following M) is a parameter regarding the upper bound of the variance of stochastic gradient in SGD, the sequence $(\mathbb{E}[F(\mathbf{w}_k)])$ generated by SGD converges to a neighborhood of F^* with a linear rate, i.e,

$$\mathbb{E}_{\xi_0:\xi_{k-1}}[F(\mathbf{w}_k) - F^*] \leq \frac{LM}{2\mu}\alpha + (1-\mu\alpha)^k(F(\mathbf{w}_0) - F^* - \frac{LM}{2\mu}\alpha) \xrightarrow{\text{linear}} \frac{LM}{2\mu}\alpha.$$

   This means that the expected optimality gap, i.e., $\mathbb{E}_{\xi_0:\xi_{k-1}}[F(\mathbf{w}_k) - F^*]$, fails to converge to zero. In order to alleviate this problem, we consider a strategy of diminishing stepsize α_k. Suppose that the SGD method is run with a stepsize sequence (α_k) such that, for all $k \in \mathbb{N}$,$\alpha_k = \frac{\beta}{\gamma+k}$ for some $\beta > \frac{1}{\mu}$ and $\gamma > 0$ satisfying $\alpha_0 \leq \frac{1}{LM_G}$. Please show that $\forall k \in \mathbb{N}$, we have

$$\mathbb{E}_{\xi_0:\xi_{k-1}}[F(\mathbf{w}_k) - F^*] \leq \frac{\tau}{\gamma + k},$$

   where $\tau = \max\{\frac{\beta^2 LM}{2(\beta\mu - 1)}, \gamma(F(\mathbf{w}_0) - F^*)\}$.

3. In practice, for the same problem, SGD enjoys less time cost but more iteration steps than gradient descent methods and may suffer from non-convergence. As a trade-off between SGD and gradent descent approaches, consider using mini-batch samples to estimate the full gradient. Taking k^{th} iteration as an example, instead of picking a single sample, we randomly select a subset $\mathcal{S}_k$ of the sample indices to compute the update direction

$$\mathbf{g}_k(\xi_k) = \frac{1}{|\mathcal{S}_k|}\sum_{i \in \mathcal{S}_k} \nabla f_i(\mathbf{w}_k)$$

where ξ_k is the selected samples. For simplicity, suppose that the mini-batches in all iterations are of constant size, i.e., $|\mathcal{S}_k| = n_m$, and the stepsize α is fixed. Please show that for mini-batch SGD, there holds

$$\mathbb{E}_{\xi_0:\xi_{k-1}}[F(\mathbf{w}_k) - F^*] \leq \frac{LM}{2\mu n_m}\alpha + (1-\mu\alpha)^k(F(\mathbf{w}_0) - F^* - \frac{LM}{2\mu n_m}\alpha) \xrightarrow{\text{linear}} \frac{LM}{2\mu n_m}\alpha.$$

Moreover, point out the advantage of mini-batch SGD compared to SGD in terms of the number of the iteration step.

4. Notice that in real applications, F is not always strongly convex. Let F be convex and continuously differentiable, and the second moment of stochastic gradient $\mathbf{g}$ be bounded, i.e.,

$$\mathbb{E}_\xi[\|\mathbf{g}(\xi)\|_2^2] \leq G^2.$$

We denote $(\mathbf{w}_k)$ as a sequence generated by SGD algorithm with a fixed stepsize α. Besides, define $\tilde{\mathbf{w}}_K = \frac{1}{K+1}\sum_{k=0}^K \mathbf{w}_k$ and $F^* = F(\mathbf{w}^*)$.

(a) If X is a random variable and $h : \mathbb{R} \to \mathbb{R}$ is a convex function, please show that

$$h(\mathbb{E}[X]) \leq \mathbb{E}[h(X)].$$

(b) Suppose that the stochastic gradient at k^{th} iteration is $\mathbf{g}_k$. Please show that

$$\mathbb{E}_{\xi_0:\xi_k}[F(\mathbf{w}_k) - F^*] \leq \mathbb{E}_{\xi_0:\xi_k}[\langle \mathbf{g}_k, \mathbf{w}_k - \mathbf{w}^*\rangle].$$

(c) Please show that

$$\mathbb{E}_{\xi_0:\xi_k}[F(\mathbf{w}_k) - F^*] \leq \frac{1}{2\alpha}\mathbb{E}_{\xi_0:\xi_k}[\|\mathbf{w}_k - \mathbf{w}^*\|_2^2 - \|\mathbf{w}_{k+1} - \mathbf{w}^*\|_2^2 + \alpha^2\|\mathbf{g}_k\|_2^2].$$

(d) Please show that

$$\mathbb{E}_{\xi_0:\xi_K}[F(\tilde{\mathbf{w}}_K) - F^*] \leq \frac{\|\mathbf{w}_0 - \mathbf{w}^*\|_2^2 + \alpha^2 G^2(K+1)}{2\alpha(K+1)} \to \frac{\alpha G^2}{2}.$$

Solution: ■

Exercise 7: Programming Exercise: Logistic Regression

We provide you with a dataset of handwritten digits[1] that contains a training set of 60000 examples and a test set of 2022 examples ("hw5_lr.mat"). Each image in this dataset has 28×28 pixels and the associated label is the handwritten digit—that is, an integer from the set $\{0, 1, \cdots, 9\}$—in the image. In this exercise, you need to build a logistic regression classifier to *predict if a given image has the handwritten digit* 6 *in it or not.* You can use your favorite programming language to finish this exercise.

1. Normalize the data matrix and please find a Lipschitz constant of $\nabla L(\mathbf{w})$, where $L(\mathbf{w})$ is the objective function of the logistic regression after normalizing and $\mathbf{w}$ is the model parameter to be estimated.

2. (a) Use the gradient descent algorithm (GD), which is a special case of ISTA introduced in Lecture 09, and SGD to train the logistic regression classifier on the training set, respectively. Evaluate the classification accuracy on the training set after each iteration. Stop the iteration when Accuracy $\geq 90\%$ or total steps are more than 5000. Please plot the accuracy of these two classifiers (the one trained by GD and the other trained by SGD) versus the iteration step on one graph.

   (b) Compare the total iteration counts and the total time cost of the two methods (GD and SGD), respectively. Please report your result.

   (c) Compare the confusion matrix, precision, recall and F1 score of the two classifiers (the one trained by GD and the other trained by SGD). Please report your result.

   (d) Use GD and SGD to train the logistic regression classifier with a 2-norm regularization term. Note that other experimental setup details is in line with 2.(a). Please plot the accuracy of these two classifiers (the one trained by GD and the other trained by SGD) versus the iteration step on one graph and compare the confusion matrix, precision, recall and F1 score of the two classifiers.

3. (a) The training set is imbalanced as the majority class has roughly nine times more images than the minority class. Imbalanced data can hurt the performance of the classifiers badly. Thus, please undersample the majority class such that the numbers of images in the two classes are roughly the same.

   (b) Use GD and SGD to train the logistic regression classifier on the new training set after undersampling. Stop the iteration when Accuracy $\geq 90\%$ or total steps are more than 5000.

   (c) Evaluate the two classifiers (the one trained with GD on the original training set and the other trained on the new training set after undersampling) on the test set. Compare the confusion matrix, precision, recall and F1 score of the two classifiers. Please report your result.

Solution: ■

[1] This dataset is modified from the MNITS dataset: http://yann.lecun.com/exdb/mnist/