CS410: Artificial Intelligence 2020 Fall
Homework 2: Machine Learning Basics and Logistic Regression
Due date: 23:59:59 (GMT +08:00), November 11, 2020

1 True or False Questions

1. Classification and regression are both supervised learning problems.

Answer: True.

2. Overfitting may occur in both classification and regression problems.

Answer: True.

3. Generally, regression is not used in classification problems, but there are also special ones. For example, logistic regression can be used to solve 0/1 classification problems.

Answer: True.

4. The most commonly used evaluations for both regression and classification problems are precision and recall.

Answer: False.

5.(single-choice) After evaluating the model, it is concluded that there is a bias in the model. Which of the following methods may solve this problem:
A. Reduce the number of features in the model
B. Add more features to the model
C. Add more data
D. Both B and C
E. All of the above

Answer: B.

6.(single-choice) To do a two-class prediction problem, first set the threshold to 0.5. Samples with a probability greater than or equal to 0.5 are classified as positive examples (i.e. 1), and samples less than 0.5 are classified as negative examples (i.e. 0). Then, use the threshold $n(n > 0.5)$ to re-divide the sample into positive and negative examples. Which of the following statements is correct ()

(a) Increasing the threshold will not increase the recall
(b) Increasing the threshold will increase the recall
(c) Increasing the threshold will not reduce the precision
(d) Increasing the threshold will reduce the precision

A. (a) B. (b) C.(a) and (c) D.(b) and (d) E. None

Answer: C.

2 Calculation Questions (Please provide the detailed calculation process)

1.Given 100 human photographs with 50 female (labeled as 1) and 50 male (labeled as 0), an algorithm predicts 45 to be male, 55 to be female. Among the 45 male predictions, 5 predictions are not correct. Among the 55 female predictions, 10 predictions are not correct.
Please calculate the Accuracy, Precision, Recall and F-measure of this algorithm.

Answer: The tabular of TP, TN, FP, and FN for predicting whether **male** or not is:

	Prediction 1	Prediction 0
Label 1	40	10
Label 0	5	45

Therefore, we have:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = 0.85 = 85\% \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \approx 0.8889 = 88.89\% \tag{2}$$

$$Recall = \frac{TP}{TP + FN} = 0.8 = 80\% \tag{3}$$

$$F1score = \frac{2 \times Precision \times Recall}{Precision + Recall} \approx 0.8421 = 84.21\% \tag{4}$$

2. Assuming that the probability of a series of samples being divided into positive classes has been obtained, the following figure is an example. There are 10 test samples in the figure. The ”Class” column indicates the true label of each test sample (1 represents positive sample, 0 represents negative sample), ”Prediction” represents the probability of each test sample belonging to a positive sample.
Please calculate the AUC score.

Class	Prediction
1	0.92
1	0.88
0	0.76
1	0.67
1	0.59
1	0.43
0	0.38
0	0.36
1	0.29
0	0.2

Answer: We use each predicted value as the threshold to draw the ROC curve, and calculate the AUC score. Predicted value which is **no less than** the current threshold is predicted to be 1.

- **Threshold = 0.2:** $TPR = 1, FPR = 1$
- **Threshold = 0.29:** $TPR = 1, FPR = 0.75$
- **Threshold = 0.36:** $TPR = 0.83, FPR = 0.75$
- **Threshold = 0.38:** $TPR = 0.83, FPR = 0.5$
- **Threshold = 0.43:** $TPR = 0.83, FPR = 0.25$
- **Threshold = 0.59:** $TPR = 0.67, FPR = 0.25$
- **Threshold = 0.67:** $TPR = 0.5, FPR = 0.25$
- **Threshold = 0.76:** $TPR = 0.33, FPR = 0.25$
- **Threshold = 0.88:** $TPR = 0.33, FPR = 0$
- **Threshold = 0.92:** $TPR = 0.17, FPR = 0$

The ROC curve is as follows:

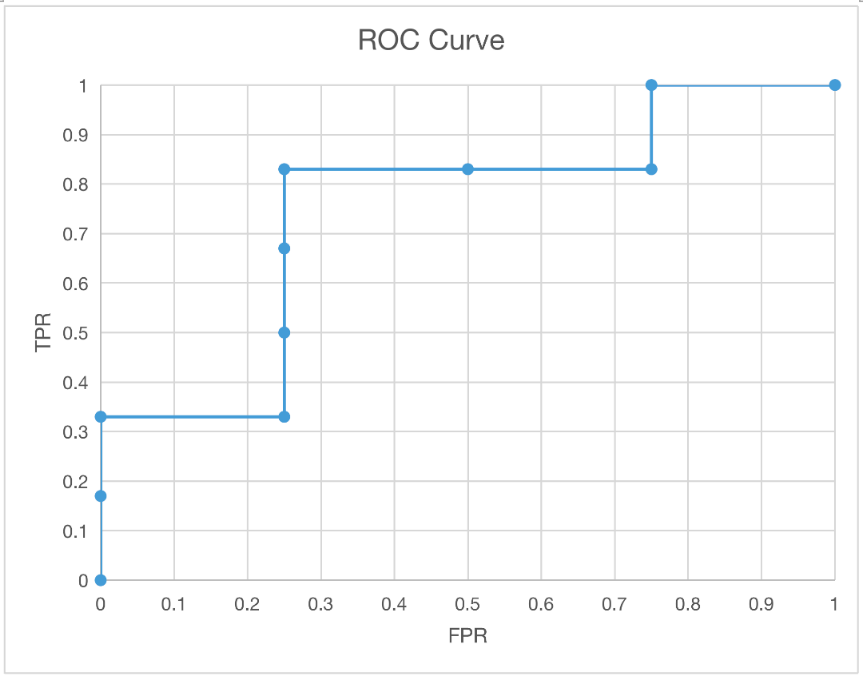

Figure 1: The ROC Curve of Problem 2

$$\begin{aligned} AUC =& 0.33 \times 0.25 + 0.83 \times 0.5 + 1 \times 0.25 \\ =& 0.7475 \end{aligned} \tag{1}$$

Therefore, we have the AUC score is 0.7475.

3 Short Answer Questions

1. How to judge whether overfitting has occurred?

   Answer: The model shows high variance and low bias on the training set. That is, the generalization error is relatively large, and the generalization ability is poor

2. What are the reasons for overfitting and how to solve it?

   Answer: The reasons for overfitting include:

   - The sample size is too small, or the sampling method is wrong.
   - Too many parameters, the complexity of the model is too high.
   - Too large noise in samples.

   Solution for overfitting:

   - Enlarge the size of sampling.
   - Reduce the complexity of the model.
   - Regularization, increase the regularization coefficient.

3. Please list common regularization methods and their comparison.

   Answer: Common regularization methods include **L1 Regularization** and **L2 Regularization**.

   - **Similarities:** L1 and L2 both consist of coefficients in the model. They can both ease the overfitting by increasing the regularization coefficient.
   - **Differences:** L1 uses absolute sum, while L2 uses sum of square. The regularization precision of L1 is higher than L2, while the regularization range of L2 is larger than L1.

4. Please describe the difference between generative and discriminative models.

   Answer: Differences:

   - Discriminative model models the dependence of unobserved variables on observed ones, while genarative model models the joint probabilistic distribution of data.
   - Discrimiative model is deterministic and probabilistic, while generative model has some hidden parameters or variables, and do the conditional inference.

4 Programming

1. National Institute of Diabetes and Digestive and Kidney Diseases wants to establish a model that automatically diagnose whether a Pima Indian has diabetes and provides a dataset as the material of the establishment of the model (Please see Supplementary materials). The dataset contains information like Pregnancies, Glucose, Blood Pressure, Skin Thickness, Insulin, BMI, Diabetes Pedigree Function, Age, of female Pima Indians over 21. In the dataset, outcome labels as 1 means that the person has diabetes. Otherwise, she does not have.

Please implement Logistic Regression to establish this model, and use gradient descent to optimize the model. Change the learning rate gradient descent to plot accuracy-learning rate graph.

Answer: We use Logistic Regression with **TensorFlow** tool to solve this problem. The model satisfies:

- Increase learning rate from 0.0000001 to 0.0005, each step is 0.000005 (except for the first iteration).
- For each train in a certain learning rate, the epoch is 5000.
- The whole data set is used for training, and the accuracy is calculated based on the training set.

The result is as follows:

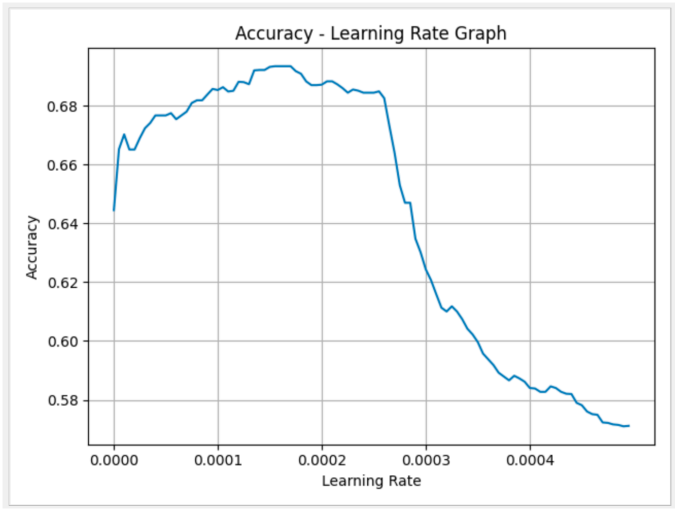

Figure 2: The Accuracy-Learning Rate Graph of Problem 4

Notice that when the learning rate is around $0.00015 - 0.00016$, we can get the relatively highest accuracy.

The reason is that when the learning rate is too low, it's not enough to fully get the best accuracy, while when the learning rate is too high, the model cannot converge in 5000 epochs.

The **Python** source code is given in **Appendix A**.

A Appendix A: Source Code

```
import tensorflow as tf
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

LR = 0.0003
EPOCH = 5000

readIn = pd.read_csv("assignment 2-supp.csv", header=0)
data = readIn.values

train_X = data[:,:-1]
train_Y = data[:,-1:]
feature_num = len(train_X[0])
sample_num = len(train_X)

accuracyArr = []
lrArr = []

input = tf.placeholder(tf.float32, name="input")
label = tf.placeholder(tf.float32, name="label")

def train_model(lr = LR):
    W = tf.Variable(tf.zeros([feature_num, 1]))
    b = tf.Variable([0.1])

    db = tf.matmul(input, tf.reshape(W, [-1, 1])) + b
    prediction = tf.sigmoid(db, name="prediction")
    cost0 = label * tf.log(prediction)
    cost1 = (1 - label) * tf.log(1 - prediction)
    cost = (cost0 + cost1) / -sample_num
    loss = tf.reduce_sum(cost)
    optimizer = tf.train.GradientDescentOptimizer(lr).minimize(loss)

    init = tf.group(tf.global_variables_initializer(),
            tf.local_variables_initializer())
    with tf.Session() as sess:
        sess.run(init)
        avgLoss = 0
        avgAccuracy = 0
        for epoch in range(EPOCH):
            feed_dict = {"input:0": train_X, "label:0": train_Y}
            _, tmpLoss = sess.run([optimizer, loss], feed_dict)
            avgLoss += tmpLoss

            tmpAmount = 0
            pred = sess.run(prediction, feed_dict)
            for i in range(0, sample_num, 1):
                if((pred[i] < 0.5 and train_Y[i] == 0)
                    or (pred[i] >= 0.5 and train_Y[i] == 1)):
                    tmpAmount = tmpAmount + 1
            avgAccuracy += float(tmpAmount / sample_num)

            if epoch % 100 == 0:
                avgLoss = 0
                avgAccuracy = 0

        return float(avgAccuracy / 100)

for i in range(1, 5000, 50):
    lr = (i - 1) * 0.0000001
    if(i == 1):
        lr = i * 0.0000001

    accuracy_ = train_model(lr)
    accuracyArr.append(accuracy_)
    lrArr.append(lr)

    print("Epoch: ", int((i - 1) / 50), "|",
        "Learning Rate = %.7f" % lr, "|",
        "Accuracy = %.4f" % accuracy_)

plt.plot(lrArr, accuracyArr)
plt.grid(True)
plt.title('Accuracy - Learning Rate Graph')
plt.xlabel('Learning Rate')
plt.ylabel('Accuracy')
plt.axis('tight')
plt.xticks(np.arange(0, 0.0005, 0.0001))
plt.show()
```