Shanghai Jiao Tong University
School of Electronic and Electrical Engineering

Neural Network Theory and Applications Assignment 2

Tianyue Cao
119034910071

Contents

Problems

In this assignment, the support vector machine (SVM) and the Min-Max-Module SVM will be used to deal with multi-class classification problems. SVMs usually handle binary classification tasks. If there are several classes to recognize, some strategies are needed. The most common methods are one-vs-one strategy, one-vs-rest strategy, and part-vs-part strategy.

Two problems are given below. The dataset used in this homework is the SJTU Emotion EEG Dataset (SEED), which is a three-class classification problem. The dataset can be downloaded from this **link**. Four files are included in the .zip file: train_data.npy, train_label.npy, test_data.npy, and test_label.npy, and 37367 samples are included in the training data, and 13588 samples in the test data.

Problem 1:

Solving the three-class classification problem in the given dataset using SVM classifiers and the one-vs-rest strategy. SVM classifiers are provided in **LibSVM** package and other machine learning libraries (**sklearn**). You can use these libraries to solve this problem.

Notice: the SVM provided in these third-party modules can handle multi-class classification. However, you are required to write the one-vs-rest strategy by yourself in this assignment

Problem 2:

Solving the three-class classification problem using Min-Max-Module SVM and part-vs-part task decomposition method. You should divide the three-class problem into three two-class problems using one-vs-rest method and then decompose these imbalance two-class problems into balance two-class problems following random task decomposition and task decomposition with prior knowledge strategies. Please compare the performance of SVMs obtained in Problem One and the Min-Max-Module SVMs here.

CHAPTER 1
Basic Information

1.1 Environments

In this assignment, the hardware and software environments I used are shown as following:

- **Device:** MacBook Pro (13-inch, 2017, Four Thunderbolt 3 Ports)
- **Processor:** 3.1 GHz Intel Core i5
- **Operation System:** macOS Catalina 10.15.4
- **Requirements:** Python 3.6, scikit-learn 0.20.1, Numpy 1.15.4, matplotlib, pickle

1.2 Datasets

The dataset is the SJTU Emotion EEG Dataset (SEED), which is a three-class classification problem. The training set contains 37367 samples and the test set contains 13588 sample. The sample distribution of training data is shown as following:

- **Number of samples of class -1:** 12320
- **Number of samples of class 0:** 12144
- **Number of samples of class 1:** 12903

The number of samples of each class is approximately the same, Each sample is represented by a 310 dimension vector.

CHAPTER 2

Solution for Problem 1

In problem 1, I use **one-vs-rest** strategy to train 3 2-class SVM classifiers, comparing the positive scores in three classifiers to get the final score and class of a sample. I use internal function in **scikit-learn** library to implement SVM and have tried **linear, RBF and polynomial** kernel. And get the results under different hyperparameters at last.

2.1 One-vs-rest SVM Classifier

Since the problem is a 3 class problem, there are one-vs-rest, one-vs-one, part-vs-part strategies to divide the problem to multiple subproblems. In this problem, I use one-vs-rest strategy to divide the problem to three 2-class SVM classification problems.

2.1.1 Dataset Division

As shown in Section 1.2, there are three labels $\{-1, 0, 1\}$ in this task. I split the training set to two parts as shown in Table 2.1 to train three classifiers, set the label of the corresponding class to the label in table.

Then I get 3 group of datasets, using one-vs-rest strategy. For detailed implementation, I use the function in numpy: `np.where(train_label == 1,1,0)` to set elements equal to 1 to 1, others 0.

	class -1	**class 0**	**class 1**
classifier 1	1	0	0
classifier 2	0	1	0
classifier 3	0	0	1

Table 2.1. Label division for three one-vs-rest classifiers.

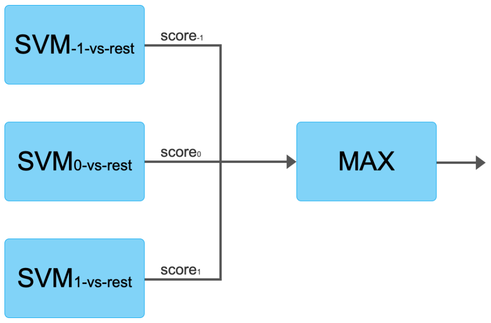

Figure 2.1. Problem 1 model structure.

2.1.2 Model

The model structure is shown in Figure 2.1. Using the three group of dataset division, I setup three SVM classifiers, and train them separately. Each classifier outputs a score for the related class. Then choose the best score for each testing sample, and classify the sample to the corresponding class.

For detailed implementation, I use the function `sklearn.svm.SVC(kernel, C, class_weight)` to setup the SVM classifiers. `kernel` is an argument to choose the kernel SVM classsifier use, `C` is the penalty parameter, `class_weight` is a parameter to set diffrent penalty parameters C for different classes, if it is set to 'balanced', then the weight is automatically ajusted to the inversely proportional to the class frequency of the input data. Also, to improve the robustness of the model, I use `sklearn.calibration.CalibratedClassifierCV` to do cross validation. The cross validation generator is used to estimate the calibration of training samples and test samples for each split model parameter. Then the probability of folding prediction is averaged.

2.2 Experiments & Results

2.2.1 Model Evaluation

In this problem, I use **accuracy, precision, recall, f1-score, and ROC curve** to evaluate the performance of the model.

First, define 4 parameters:

- **True Positive (TP):** prediction is positive, ground truth is positive.
- **False Positive (FP):** prediction is positive, ground truth is negative.

- **True Negative (TN):** prediction is negative, ground truth is negative.
- **False Negative (FN):** prediction is negative, ground truth is positive.

Then the evaluation indicators are defined as:

- **accuracy:** $\frac{TP+TN}{TP+TN+FP+FN}$
- **precision:** $\frac{TP}{TP+FP}$
- **recall:** $\frac{TP}{TP+FN}$
- **f1-score:** $\frac{2}{1/presicion+1/recall}$

Since it is a multi-class problem, each indicator above has **micro, macro and weighted** average. For micro, samples are used for average; for macro, classes are used for average; and for weighted, classes with weight are used for averge.

ROC curve is a curve is the line drawn at each point with the false probability p (Y / N) obtained by the subject under different judgment standards as the abscissa and the hit probability p (Y / Sn) as the ordinate under specific stimulation conditions. The area under the curve is called AUC. ROC curve and AUC can be used to evaluate the performace of classifiers.

2.2.2 Experiments Setup

I have tried different kernels (linear, RBF, polynomial) and different penalty parameter C (1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1) in experiments. Other parameters of the SVM classifier is set to be default. During training, I use 5-fold cross validation to choose model.

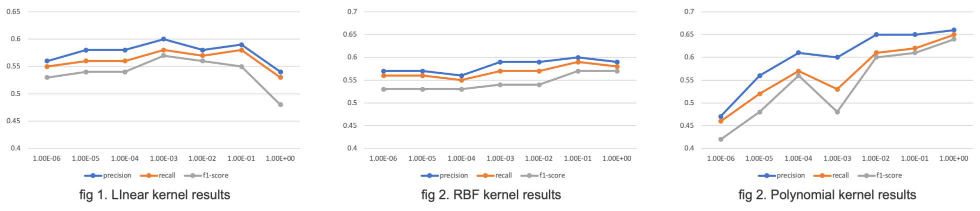

Figure 2.2. Weighted average precision, recall, f1-score for different kernels and penalty parameters.

2.2.3 Results & Analysis

For different kernels and parameter C, I use grid search to find the satisfied parameter. Weighted average precision, recall, f1-score for different kernels and penalty parameters are shown in Figure 2.2. The ROC curves of different parameters are shown in Figure 2.3.

According to Figure 2.2, for linear kernel, model with penalty parameter C=1e-3 can achieve the best performance; For RBF kernel, model with penalty parameter C=0.1 can achieve the best performance; For polynomial kernel, model with penalty parameter C=1 can achieve the best performance. Among all the kernels, polynomial kernel with C=1 can get the best performance. Also, polynomial kernel SVM classifiers' results with different Cs are widely distributed, while results with RBF kernel changes little.

According to Figure 2.3, the best parameters combination can also be conformed. Polynomial kernel with C=1 can get the best performance, and the average AUC is 0.8. While using linear kernel, as is shown in fig a1-7, with C increases, the performance of class 1 increases and class -1 and class 0 decreases. While using RBF kernel, the performance changes little with different Cs. While using polynomial kernel, AUC increases with C increases.

By grid search, I got the best parameter combination for this task: kernel **polynomial**, C=**1**. Accuracy of the model is 0.65, and the classification report is shown in Table 2.2. According to the table and ROC curve, the classifier performs best on class 1 and worst on class 0.

class	**precision**	**recall**	**f1-score**	**support**
-1	0.74	0.52	0.61	4480
0	0.58	0.55	0.57	4416
1	0.65	0.87	0.74	4692
micro avg	0.65	0.65	0.65	13588
macro avg	0.66	0.65	0.64	13588
weighted avg	0.66	0.65	0.64	13588

Table 2.2. Classification report for polynomial kernel and C=1.

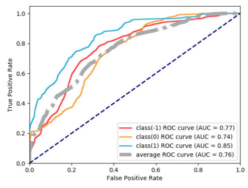

fig a.1. ROC curve for SVM (linear kernel, C=1e-6)

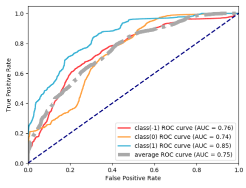

fig b.1. ROC curve for SVM (rbf kernel, C=1e-6)

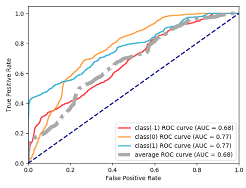

fig c.1. ROC curve for SVM (polynomial kernel, C=1e-6)

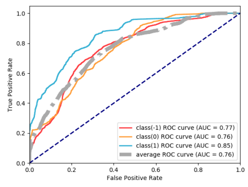

fig a.2. ROC curve for SVM (linear kernel, C=1e-5)

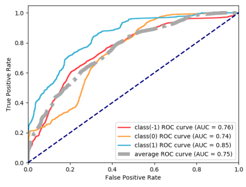

fig b.2. ROC curve for SVM (rbf kernel, C=1e-5)

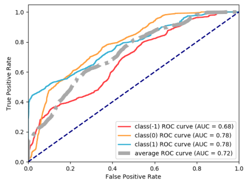

fig c.2. ROC curve for SVM (polynomial kernel, C=1e-5)

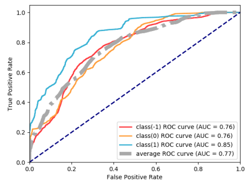

fig a.3. ROC curve for SVM (linear kernel, C=1e-4)

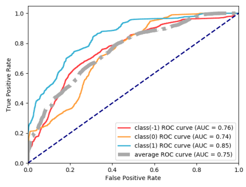

fig b.3. ROC curve for SVM (rbf kernel, C=1e-4)

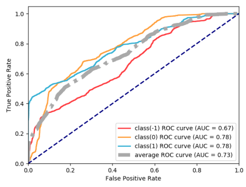

fig c.3. ROC curve for SVM (polynomial kernel, C=1e-4)

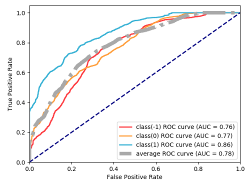

fig a.4. ROC curve for SVM (linear kernel, C=1e-3)

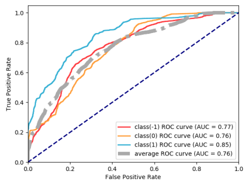

fig b.4. ROC curve for SVM (rbf kernel, C=1e-3)

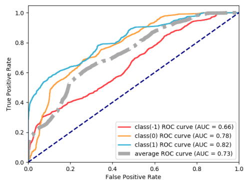

fig c.4. ROC curve for SVM (polynomial kernel, C=1e-3)

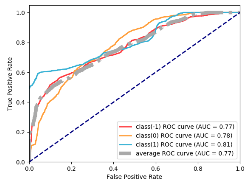

fig a.5. ROC curve for SVM (linear kernel, C=1e-2)

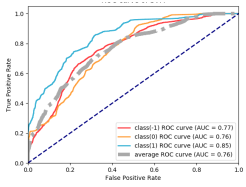

fig b.5. ROC curve for SVM (rbf kernel, C=1e-2)

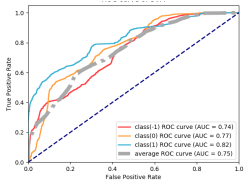

fig c.5. ROC curve for SVM (polynomial kernel, C=1e-2)

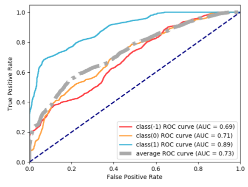

fig a.6. ROC curve for SVM (linear kernel, C=1e-1)

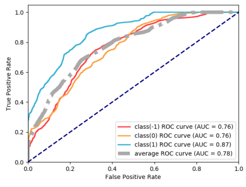

fig b.6. ROC curve for SVM (rbf kernel, C=1e-1)

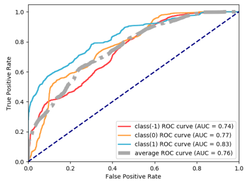

fig c.6. ROC curve for SVM (polynomial kernel, C=1e-1)

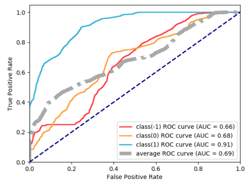

fig a.7. ROC curve for SVM (linear kernel, C=1)

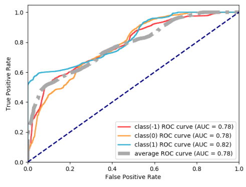

fig b.7. ROC curve for SVM (rbf kernel, C=1)

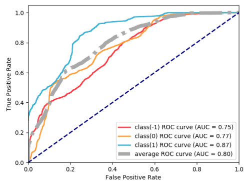

fig c.7. ROC curve for SVM (polynomial kernel, C=1)

Figure 2.3. ROC curve and AUC for different parameters.

CHAPTER 3

Solution for Problem 2

In problem 1, one-vs-rest strategy is used for dividing the task to three two-class classifiers. To decompose the problem, I use Min-Max-Module SVM and part-vs-part task decomposition method. And two kinds or decomposition strategies are used, random decomposition and decomposition based on prior knowledge. The prior knowledge I used is the gender information of the testing people.

3.1 Min-Max SVM Decomposed Randomly

The model is shown in Figure 3.1. I randomly split the training dataset into 8 equal parts, get the minimum score as the score for the corresponding class. Then get the maximum score as the classification score, and set the class to corresponding label. The prediction score $p(x_i)$ is shown in (3.1):

$$p(x_i) = \max(\min_{j=0}^{7} p_{0j}(x_i), \min_{j=0}^{7} p_{1j}(x_i), \min_{j=0}^{7} p_{2j}(x_i)) \tag{3.1}$$

And the corresponding k of the maximum of $\min_{j=0}^{7} p_{kj}(x_i)$, is the output class label.

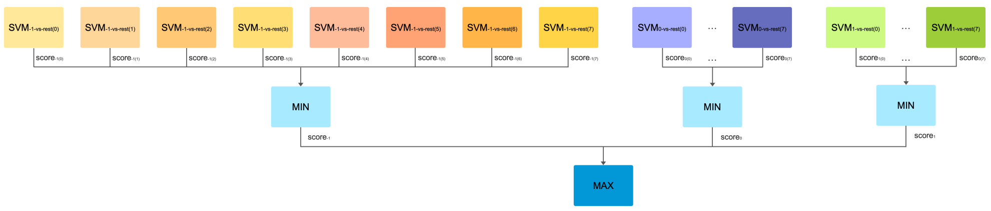

Figure 3.1. The randomly decompose Min-Max SVM model.

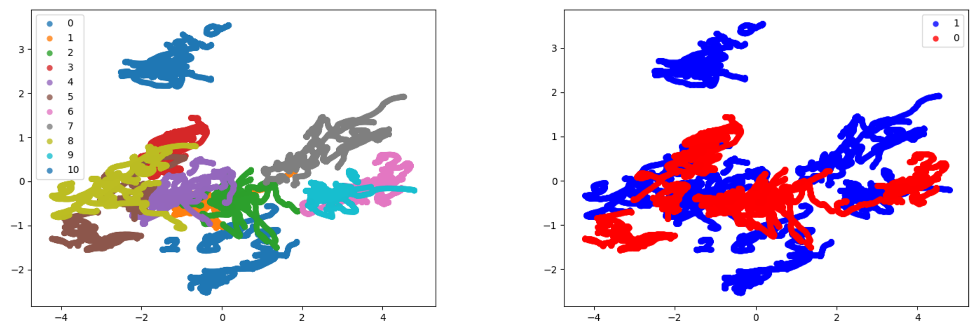

fig a. The distribution of training data by testing ID.

fig b. The distribution of training data by gender.

Figure 3.2. Training data distribution after PCA. Fig a shows the distribution by testing person ID, and fig b shows the distribution by gender.

3.2 Min-Max-Module SVM Based on Prior Knowledge

3.2.1 Prior Knowledge

Data is collected from 15 people, 11 in training dataset and 4 in testing dataset. The gender of these people is shown in Table 3.1. To visulize the data in training set, I use PCA to lower the data dimension to 2, and the distribution of training data is shown in Figure 3.2. Fig a shows the distribution of training data grouped by testing person ID, and fig b shows the distribution of training data grouped by gender. So in this section, I split the data into 2 parts, male and female.

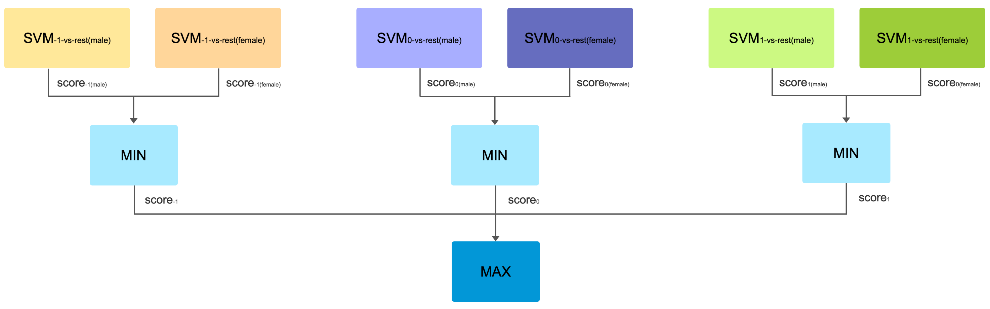

Figure 3.3. The MIn-Max SVM model decomposed by prior knowledge.

	testing ID	gender (male 1, female 0)
training	1	1
	2	0
	3	0
	4	0
	5	1
	6	0
	7	0
	8	1
	9	1
	10	1
	11	1
testing	12	1
	13	0
	14	1
	15	0

Table 3.1. Prior knowledge for dataset.

3.2.2 Model

The model structure is shown in Figure 3.3. For each class, train a classifier for male and a classifire for female, choose the minimum score as the corresponding class score, and then the maximum of three classes is chosen as the final classification result. The prediction score $p(x_i)$ is shown in (3.2):

$$p(x_i) = \max(\min_{j\in\{0,1\}} p_{0j}(x_i), \min_{j\in\{0,1\}} p_{1j}(x_i), \min_{j\in\{0,1\}} p_{2j}(x_i)) \tag{3.2}$$

And the corresponding k of the maximum of $\min_{j\in\{0,1\}} p_{2j}(x_i)$, is the output class label.

3.3 Results & Analysis

The parameter tuning process is similar to Chapter 2, the best performance parameters for random decomposition is **linear** kernel and **C=1e-3**, and the best performance parameters for decomposition based on prior knowledge is **polynomial** kernel and **C=1**.

In this section, I compare the results of basic method in Chapter 2, random decomposition Min-Max SVM, and decomposition based on prior knowledge Min-Max SVM.

The classification report of differnt methods are shown in Table 2.2, Table 3.2, Table 3.3. The ROC curve and AUC of different methods are shown in Figure 3.4.

class	precision	recall	f1-score	support
-1	0.55	0.39	0.46	4480
0	0.52	0.35	0.42	4416
1	0.54	0.86	0.66	4692
micro avg	0.54	0.54	0.54	13588
macro avg	0.54	0.53	0.51	13588
weighted avg	0.54	0.54	0.51	13588

Table 3.2. Classification report for random decomposition Min-Max SVM (linear kernel, C=1e-3).

class	precision	recall	f1-score	support
-1	0.66	0.55	0.60	4480
0	0.63	0.49	0.55	4416
1	0.65	0.89	0.75	4692
micro avg	0.65	0.65	0.65	13588
macro avg	0.65	0.64	0.63	13588
weighted avg	0.65	0.65	0.63	13588

Table 3.3. Classification report for decomposition by prior knowledge Min-Max SVM (polynomial kernel, C=1).

model	precision	recall	f1-score	AUC
model 1	0.66	0.65	0.64	0.80
model 2	0.54	0.54	0.51	0.75
model 3	0.65	0.65	0.63	0.78

Table 3.4. Weighted average precision, recall, f1-score, and AUC for 3 SMV models.

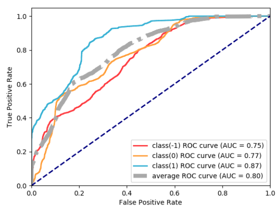

fig c. ROC curve for basic one-vs-rest SVM (polynomial kernel, C=1).

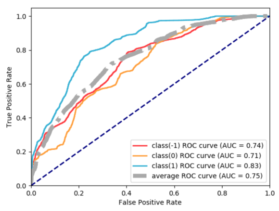

fig b. ROC curve for random decomposition Min-Max SVM (linear kernel, C=1e-3).

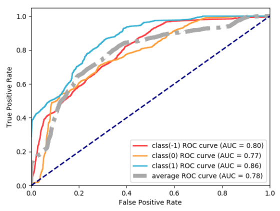

fig c. ROC curve for decomposition by prior knowledge Min-Max SVM (polynomial kernel, C=1).

Figure 3.4. ROC curve for 3 SMV models.

For convenience, note the basic one-vs-rest SVM model as **model 1**, the random decomposition Min-Max SVM model as **model 2**, and the decomposition by prior knowledge Min-Max SVM model as **model 3**. As is shown in Table 3.4 and Figure 3.4, model 1 can achieve the best average performance. Model 2 achieves the worst average performance. The reason may be after random decomposition of the dataset, the performance of each classifier decreases, then the final performance decreases. Model 3 performs much better than model 2, which means that decomposition by prior knowledge can lead to great increasing. Also, though the average point of model 1 is higher than model 3, model 3 can get more balanced and robust results by Figure 3.4.

All in all, decomposition based on prior knowledge and Min-Max-Module are important strategies for large scale problems, which may increase the performance.

Conclusion

In this assignment, I solve a 3-class classifiaction problem using SVM classifier. In problem 1, an one-vs-rest strategy is used to decompose the problem into three 2-class classification problems. In problem 2, I use some decomposition strategies to perform Min-Max-Module, including random decomposition and decomposition by prior. The prior knowledge is the gender of the testing people.

By tuning the parameters, I finally get precision, recall, f1-score and AUC for problem 1, and get precision, recall, f1-score and AUC for random decomposition strategy and precision, recall, f1-score and AUC for prior knowledge decompositon strategy. Though in this problem, problem 2 cannot get better result than problem 1, but decomposition and Min-Max-Module are important strategies for large scale problems. And prior knowledge can help to decide the decomposition strategy and get better performance.