华南师范大学
SOUTH CHINA NORMAL UNIVERSITY

实验报告

课程名称：数据挖掘课程项目

实验题目：通过 HAMD 量表评估抑郁症

指导老师：樊小毛

组　　别：10

成　　员：唐高智 简楚帆 林宏拓 王铭杰 李辉龙

学　　院：计算机学院

专　　业：计算机科学与技术

班　　级：18 级计算机科学与技术 3 班

时　　间：2021 年 1 月 6 日

Abstract

Depression is one of the most common mental disorders, and studies have shown that acupuncture has significant efficacy in treating depression and assessing the degree of depression in patients using the 17-item HAMD scale, however, assessment is difficult for general practitioners, and the establishment of an effective model for direct assessment of depression through EEG signals can effectively address the problem of depression assessment and inadequate medical resources.

In order to develop a system that can directly assess depression through EEG signals, this paper investigates several indicators that affect patient depression based on acupuncture treatment data provided by the hospital, first pre-processing the data and subsequently exploring the data comprehensively in order to better understand the situation of this data. Subsequently, the raw data were characterized in order to better utilize valid information and eliminate useless information. Then the depression was classified and fitted using supervised learning and semi-supervised learning, respectively, and the evaluation index values of each model were obtained, and finally a comprehensive evaluation was performed based on the results of each model.

For data preprocessing and data exploration, this paper first explored the rhythm amplitude index and rhythm index before acupuncture and found that they have a certain linear relationship. To verify the conjecture, this paper calculated the correlation coefficient of each rhythm and drew a heat map, and concluded that the rhythm fluctuation index and rhythm index are indeed linear. The patients with/without depression were then marked as 1/0, and the data were subjected to relevant statistics, such as the proportion of distribution with and without depression and the distribution of HAMD scores, etc. It was found that 63.04% of the patients did not suffer from depression, as well as the largest proportion of patients with scores around 15, which helped us to understand in detail the number of patients with depression in the sample and the HAMD scores in the sample. Immediately after we calculated the distribution of each feature (e.g., FP1-A1 δ rhythm ,μV) and drew histograms and kernel function density estimates for some of the data, which collectively showed that all features were roughly normally distributed. Finally, in order to make full use of the dataset, we stitched together the post-needling EEG indicators with the pre-needling ones to form a new dataset, thus facilitating the subsequent semi-supervised learning.

For feature engineering, in order to filter the most representative data, we selected several models for feature engineering processing, which are: logistic regression, SVM-derived models (SVC corresponding to classification and SVR corresponding to regression), decision trees, random forests, GBDT-derived models (for classification GradientBoostingClassifier for classification and GradientBoostingRegressor for regression), etc. We use these models to test the data and use some basic evaluation metrics such as MSE and R2-Score to evaluate the effectiveness of the selected features,

and finally filter the optimal data. By comparing the results before and after feature engineering, we can see that after feature engineering, MSE is significantly lower and R2-Score is significantly higher as well as accuracy in terms of fitting; in terms of classification, accuracy and F1-Score are significantly higher.

For model building, in this paper, we first construct a supervised learning algorithm and train the pre-processed dataset. First, we utilize the more common algorithms in integrated learning, including the random forest algorithm in the averaging method, the Adaboost algorithm and the gradient tree enhancement algorithm in the augmentation method. The goal of the integrated approach is to combine the predictions of several basic estimators built with a given learning algorithm to improve the generalizability/robustness of a single estimator. In addition, to enhance the robustness of the model, we introduced a voting strategy, built voting classifiers, trained through these models, and predicted the output based on the total number of votes for each output class. After a ten-fold cross-check, it can be found that the Votingclf algorithm obtained the best evaluation results in both fitting and classification with an accuracy of 0.7134 and an F1-Score of 0.6841, followed immediately by plotting the confusion matrix and regression scatter plot with an MSE value of 29.16 and an R2-Score value of 0.34.

For unsupervised learning, the TSVM algorithm is used in this paper. First, the labeled and unlabeled samples are assigned to the training set, and then an initial SVM is trained using the set of labeled samples, and then the unlabeled samples are labeled using this learner, so that all samples are labeled, and the SVM is retrained based on these labeled samples, and then the error-prone samples are found and adjusted continuously. After that, the SVM is continuously adjusted by finding error-prone samples. After a ten-fold cross-check, it can be found that the TSVM model obtained the best evaluation results in terms of both fitting and classification when the values and values were initialized to 1.5 and 0.001, respectively, with an accuracy of 0.88 and an F1-Score of 0.769. Immediately after plotting the confusion matrix and the regression scatter plot, the MSE value was 16.84 and the R2-Score value was 0.24. The results show that semi-supervised learning outperforms supervised learning in all classification problems, and in the fitting problem, although the MSE values are excellent, the R2-Score is not as good as supervised learning.

In this paper, the results of each model are evaluated by finding their accuracy_score, Precision_score, Recall_score, F1_score, and, if necessary, by performing k-fold tests and drawing ROC plots for the models.

Finally, this paper evaluates supervised learning and semi-supervised learning, summarizes their advantages and disadvantages, and makes a later prospect: modeling unsupervised learning with deep learning, and then finding the optimal model.

Keywords: depression assessment; data feature engineering processing; supervised learning; support vector machine semi-supervised learning

Content

1. Contributions of members

- 唐高智：Responsible for the semi-supervised learning.
- 简楚帆：Responsible for the feature engineering.
- 林宏拓：Responsible for the supervised classification.
- 王铭杰：Responsible for the supervised regression.
- 李辉龙：Responsible for the data exploration and concatenation.

2. Introduction

2.1 Background

Depression is one of the most common mental disorders, which seriously affects patients' study, work and normal social life. At present, the treatment of depression mainly includes antidepressant drugs and traditional Chinese medicine acupuncture. Because antidepressant drug treatment is often accompanied by drug side effects, drug dependence, and poor compliance. At the same time, studies have shown that acupuncture has significant effects in the treatment of depression. In this project, a hospital uses acupuncture to treat depression and uses the 17-item HAMD scale to assess the degree of depression in patients. Since the HAMD-17 scale needs to be evaluated by professional doctors after training, it is difficult for doctors in community hospitals and township hospitals to be competent. This project intends to directly assess depression through EEG signals, which is not only more objective, but also can overcome the problem of insufficient medical resources in community hospitals or township hospitals.

2.2 Our Work

In the project, we divide the work into six steps. First, we conduct data exploration to spy on the data, then perform data preprocessing, and filter the data through feature engineering. At last, we use supervised learning and semi-supervised learning to model as well as add the data after acupuncture for training.

3. Data exploration and preprocessing

3.1 Assumption

We explored the rhythm amplitude index and rhythm index before acupuncture, and found that they have a certain linear relationship.

The steps are like this:we draw a scatter plot of the δ rhythm in the rhythm amplitude index, μV and the δ rhythm in the rhythm index. Intuitively, combined with the correlation coefficient to figure out, we guess that they are correlated (other columns, such as θ , μV rhythm and θ rhythm are the same).

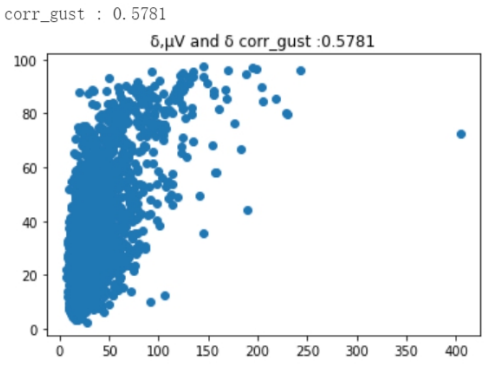

Fig. 2.1. Scatter plot of the δ rhythm

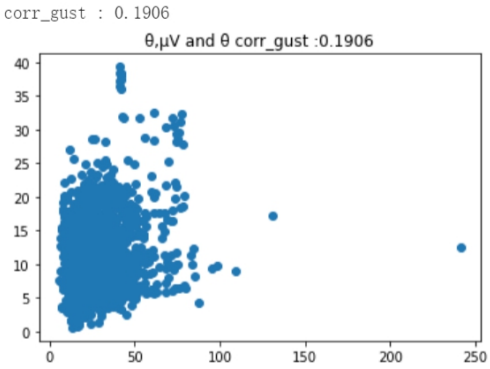

Fig. 2.2. Scatter plot of the θ rhythm

After that, we want to verify our conjecture further. Then we find the correlation of (δ rhythm, μV), (θ rhythm, μV), (α rhythm, μV), (β (LF) rhythm, μV), δ rhythm, θ rhythm, α rhythm β (LF) rhythm Coefficients and give the heat maps of them.The heat map is often used to display the correlation coefficient matrix of a set of variables. Through the heat map, we can intuitively feel the difference in the magnitude of the value. Therefore, we can be sure that the rhythm volatility index and the rhythm index are indeed linearly related.

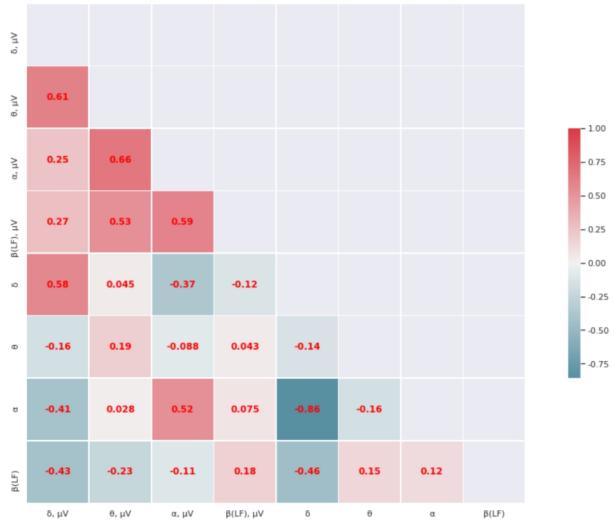

Fig. 2.3. Heatmap of the indicators' correlation coefficient

3.2 Statistics

We conduct some statistical experiments. Through the preprocessing of the array, we classify patients with depression as 0, and mark patients without depression as 1. Get the pie chart below. We found that the proportion of people who did not suffer from depression was 63.04%. By counting the distribution of HAMD, we found that the number of people with a score of 15 points or so is the largest. In this way, we have a detailed understanding of how many people in the sample suffer from depression and the HAMD score in the sample. This helps us train and understand the model.

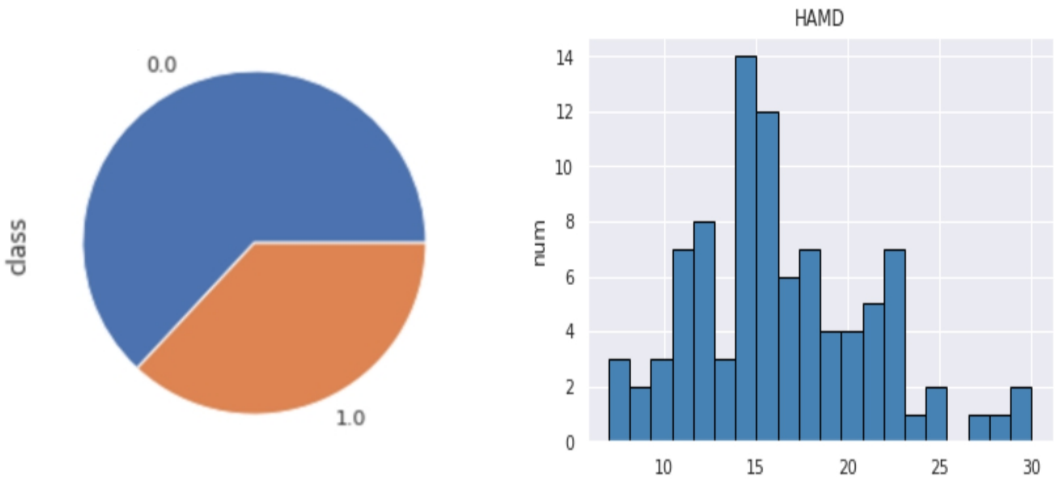

Fig. 2.4. Statistics of label and HAMD

3.3 Explore the characteristics of columns

We draw the histogram of features (such as FP1-A1 δ rhythm, μV) and the kernel function density estimation chart, observe and estimate the general sample distribution. Because we know that the value of the probability density function at a sample point represents the relative probability that the predicted distribution of that point is consistent with the measured sample distribution.We found that some features conform to normal distribution, and some are skewed. We will answer this question during data preprocessing.

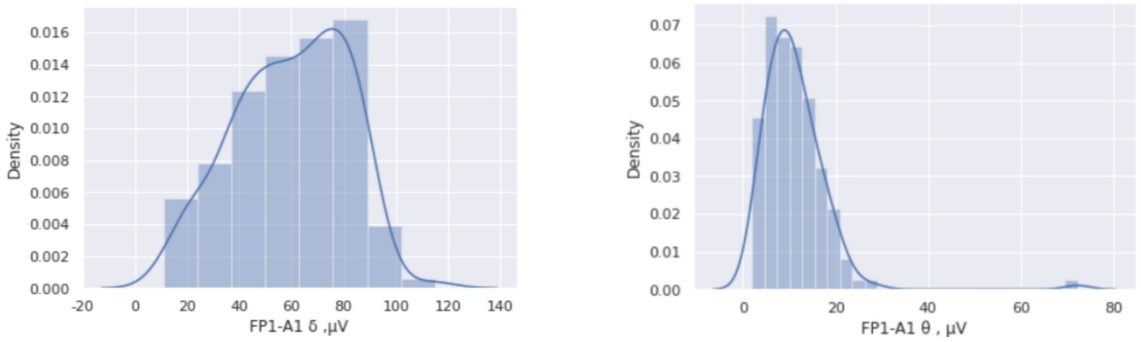

Fig. 2.5. Distribution of some indicators

3.4 Concatenate the data

In order to make full use of the data set, we concatenated the EEG indicators after acupuncture with those before acupuncture to form a new data set to facilitate semi-supervised learning.

The following chart shows the situation after concatenation.

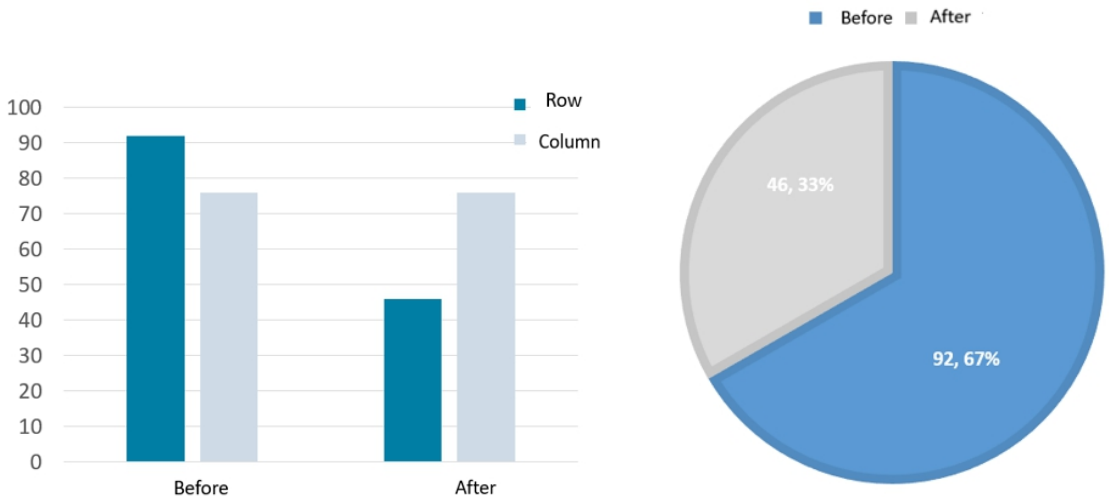

Fig. 2.6. Situation of the data after concatenation

4. Feature Engineering

4.1 The Metrics of Feature Selection

In order to test whether the filtered features are effective, we use some basic metrics to evaluate the effectiveness of the selected features. which will be introduced below.

4.1.1 Mean Squared Error (MSE)

If a vector of n predictions is generated from a sample of n data points on all variables, and n is the vector of observed values of the variable being predicted, with $\widehat{Y}$ being the predicted values (e.g. as from a least-squares fit), then the within-sample MSE of the predictor is computed as

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \widehat{Y}_i\right)^2 \quad (3\text{-}1)$$

4.1.2 R2-Score

If a dataset has n values marked $y_1, ..., y_n$ (collectively known as y_i or as a vector $y = [y_1, ..., y_n]^T$), each associated with a fitted value $f_1, ...f_n$. Define the residuals as $e_i = y_i - f_i$, if $\bar{y}$ is the mean of the observed data:

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i \quad (3\text{-}2)$$

then the variability of the data set can be measured with two sum of squares formulas:

- The total sum of squares:

$$SS_{tot} = \sum_i (y_i - \bar{y})^2 \quad (3\text{-}3)$$

- The sum of squares of residuals:

$$SS_{res} = \sum_i (y_i - f_i)^2 \quad (3\text{-}4)$$

- Coefficient of determination(R2-Score):

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}} \quad (3\text{-}5)$$

4.1.3 Accuracy

Accuracy represents the ratio of the correctly classified samples to total samples. For instance, there are a total of 138 samples in test-set; if our model successfully predict 100 samples, then the accuracy equals to $\frac{100}{138} = 0.725$.

4.1.4 Confusion matrix and F1-Score

We need to introduce the confusion matrix first before introducing the related metrics. As shown in the Figure 2.3, the confusion matrix is a table with two rows and two columns that reports the number of false positives (FP), false negatives (FN), true positives (TP) and true negatives (TN). TP represents the sample is positive and is also predicted to be positive correctly while FN represents the sample belongs to the positive class while it's predicted to be negative. Likewise, TN represents the sample is negative

and is also predicted to be negative properly while FP represents the sample belongs to the negative class while it's predicted to be positive.

		Actual class	
		P	N
Predicted class	P	TP	FP
	N	FN	TN

Fig. 3.1. Confusion matrix

Based on the above discussion, we can define the precision and recall as follows:

$$Precision = \frac{TP}{TP+FP} \tag{3-6}$$

$$Recall = \frac{TP}{TP+FN} \tag{3-7}$$

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{3-8}$$

Among the above indicators, precision metric represents the proportion of the samples predicted to be positive are actually positive samples. Recall metric represents the proportion of the positive examples in the dataset were predicted correctly. F_1 is an index used to measure the accuracy of dichotomous model in statistics, which considers both precision and recall.

4.2 The Methods of Feature Selection

In order to construct good features on the original data, it is very important to carry out feature engineering for the project. The types of feature engineering are as follows:

1. Filter

Score each feature according to divergence or relevance, set the threshold or the number of features to be selected, and then select the feature based on the rules.

2. Wrapper

According to the objective function (usually the effective score of prediction), select several features or exclude several features in each iteration.

3. Embedding

At first, some machine learning algorithms and models are used for training, and the weight coefficients of each feature are obtained, so that the features can be selected from large to small according to the coefficient.

We chose the embedding method for feature engineering, which has the advantage of method diversity and automatic selecting. The embedded model includes logistic regression, support vector machine derivative model (support vector regressor and classifier), decision tree, random forest and gradient boosting decision tree derivative model (gradient boosting decision tree regressor and classifier).

4.3 Comparison of results after feature selection

In order to effectively prove that the data after feature selection are more effective than the original data, we take the following steps to verify.

Step 1. Save the data filtered by different embedded models.

Step 2. Use different models to evaluate these data and observe the metrics of the models.

Step 3. Comprehensively observe the evaluation indicators of each model to select the best data for regression and classification.

In order to view the results after feature selection more intuitively, we illustrate them by the following diagrams.

According to the figure below, we can see that the MSE after feature selection is obviously lower, while R_2 is higher than the original data.

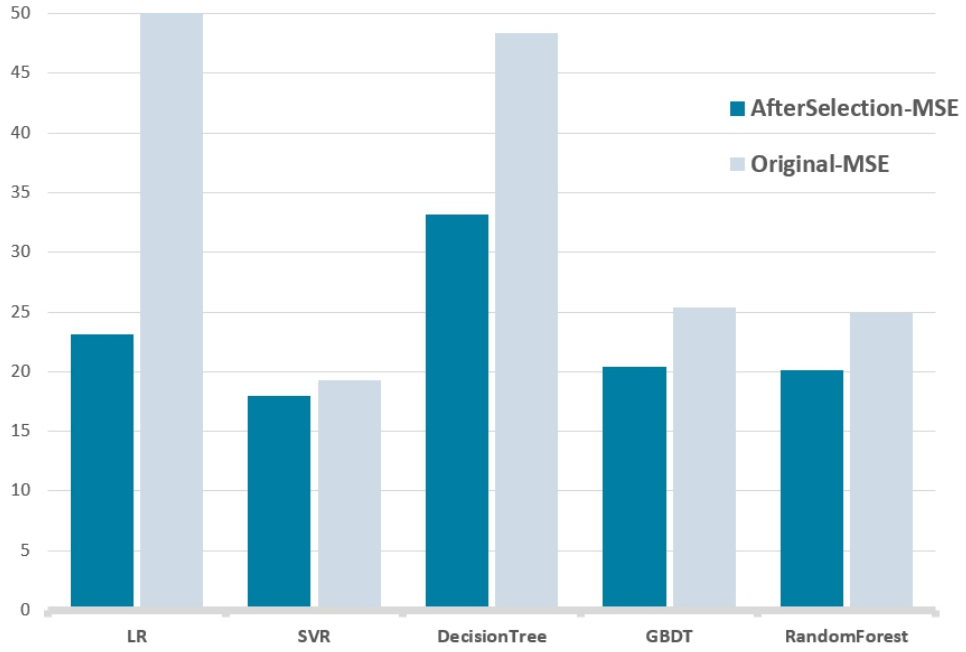

Fig. 3.2. Comparison of MSE before and after feature selection

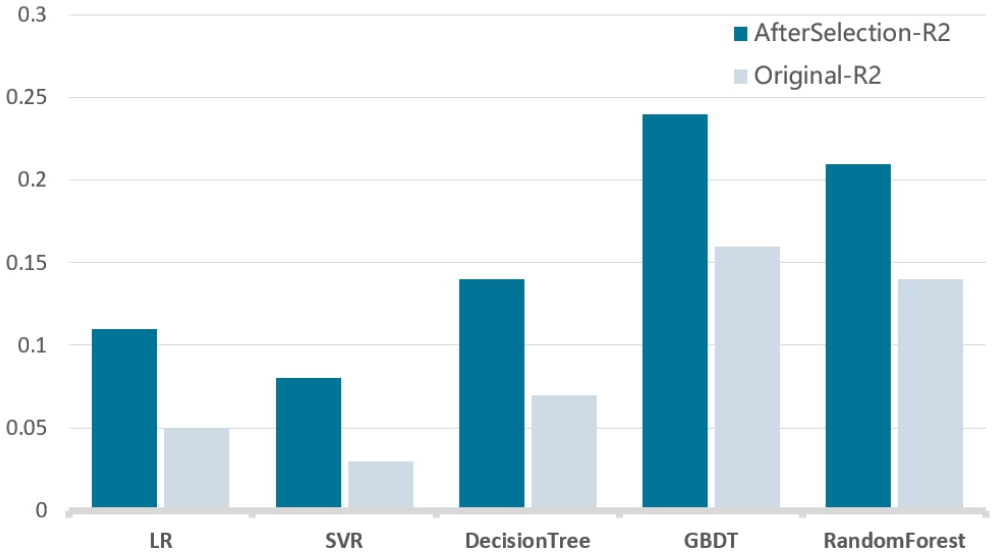

Fig. 3.3. Comparison of R_2 before and after feature selection

In terms of classification, according to the chart, it can be found that the accuracy and F_1 after feature selection are higher than the original data.

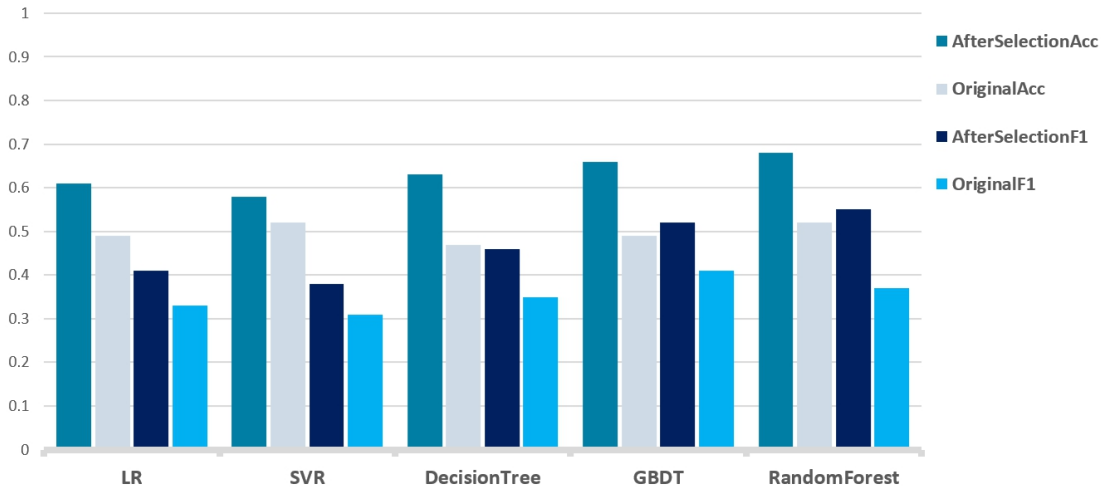

Fig. 3.4. Comparison of classification results before and after feature selection

5. Supervised Learning

5.1 Integrated model based on the tree model

Since a part of the original dataset is labeled, we first use supervised learning for data mining and then compare the obtained results with semi-supervised learning in the next step.

At first, we utilize the more common algorithms in ensemble learning, including Random Forest in averaging methods, Adaboost and Gradient Tree Boosting in boosting methods. The goal of ensemble methods is to combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator.

- **Random Forest**

A random forest is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. Here are the 4 steps to construct a random forest:

Step 1. If there are N samples, N samples are randomly selected for replacement (one sample is randomly selected each time, and then returned to continue selection). The selected N samples are used to train a decision tree as the samples at the root node of the decision tree.

Step 2. When each sample has M attributes, when each node of the decision tree needs to be split, randomly select m attributes from these M attributes and satisfy the condition m << M. Then a certain strategy (such as information gain) is used from these m attributes to select one attribute as the split attribute of the node.

Step 3. In the process of decision tree formation, each node must be split according to step 2 (it is easy to understand that if the next attribute selected by the node is the attribute that was just used when the parent node was split, then the node has reached If the leaf node is removed, there is no need to continue to split). Until it can no longer be divided. Note that there is no pruning during the entire decision tree formation process.

Step 4. Follow steps 1 to 3 to build a large number of decision trees, which constitutes a random forest.

● Adaboost

An AdaBoost classifier / regressor is a meta-estimator that begins by fitting a classifier / regressor on the original dataset and then fits additional copies of the classifier / regressor on the same dataset but where the weights of incorrectly classified instances are adjusted such that subsequent classifiers focus more on difficult cases. The algorithm of Adaboost is as follows:

Step 1. First, the weight distribution of the training data is initialized. Each training sample is given the same weight at the beginning: 1/N.

$$D_1 = \left(w_{11}, w_{12} \cdots w_{1i} \cdots, w_{1N}\right), w_{1i} = \frac{1}{N}, i = 1, 2, \cdots, N \tag{4-1}$$

Step 2. Perform multiple iterations with m = 1, 2, ... , M denotes the number of rounds of the iteration

a). Learning using a training dataset with a weight distribution Dm to obtain the basic classifier (choose the threshold that allows the lowest error rate to design the basic classifier)

$$G_m(x): \quad \chi \to \{-1, +1\} \tag{4-2}$$

b). Calculate the classification error rate of Gm(x) on the training data set

$$e_m = P\left(G_m\left(x_i\right) \neq y_i\right) = \sum_{i=1}^{N} w_{mi} I\left(G_m\left(x_i\right) \neq y_i\right) \tag{4-3}$$

c). Calculate the coefficient of Gm(x), am indicates the importance of Gm(x) in the final classifier (purpose: to obtain the weight of the basic classifier in the final classifier).

$$\alpha_m = \frac{1}{2} \log \frac{1 - e_m}{e_m} \tag{4-4}$$

d. Update the weight distribution of the training data set (purpose: get the new weight distribution of the samples) for the next iteration

$$D_{m+1} = \left(w_{m+1,1}, w_{m+1,2} \cdots w_{m+1,i} \cdots, w_{m+1,N}\right) \tag{4-5}$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m} \exp\left(-\alpha_m y_i G_m\left(x_i\right)\right), i = 1, 2, \cdots, N \tag{4-6}$$

Step 3. Combine each weak classifier

$$G(x) = \text{sign}(f(x)) = \text{sign}\left(\sum_{m=1}^{M} \alpha_m G_m(x)\right) \tag{4-7}$$

- **Gradient Boosting Decision Tree**

The basic idea of Gradient Boosting is to serially generate multiple weak learners, each with the goal of fitting the negative gradient of the loss function of the previously accumulated model so that the cumulative model loss after adding the weak learner is reduced in the direction of the negative gradient.

Algorithm of GBDT:

1. Initialize :

$$F_0(x) = \arg\min_{h \in H} Loss(y_i, h(x_i)) \tag{4-8}$$

2. For m=1:M Do:

Compute the negative gradient:

$$g_m = -\frac{\partial Loss(y, Fm-1(x))}{\partial F_{m-1}(x)} \tag{4-9}$$

Fit a weak learner which minimize:

$$\sum_{i=1}^{N} (g_m^i - h(x_i))^2 \tag{4-10}$$

Update:

$$F_m = F_{m-1} + vh(x) \tag{4-11}$$

3.Return:

$$F(x) = F_M(x) \tag{4-12}$$

Then, we use a Logistic Regression / Linear Regression as a comparison reference to determine the excellence of integrated learning.

5.2 Voting Classifier / Regressor

In order to enhance the robustness of the model, we also introduce the voting strategy, which basic idea is to select the class with the highest output among all machine learning algorithms. A Voting Classifier is a machine learning model that trains on an ensemble of numerous models and predicts an output (class) based on their highest probability of chosen class as the output.

It simply aggregates the findings of each classifier passed into Voting Classifier and predicts the output class based on the highest majority of voting. The idea is instead of creating separate dedicated models and finding the accuracy for each them, we create a

single model which trains by these models and predicts output based on their combined majority of voting for each output class.

Voting Classifier supports two types of votings.

- **Hard Voting:**

In hard voting, the predicted output class is a class with the highest majority of votes i.e the class which had the highest probability of being predicted by each of the classifiers. Suppose three classifiers predicted the output class(A, A, B), so here the majority predicted A as output. Hence A will be the final prediction.

- **Soft Voting:**

In soft voting, the output class is the prediction based on the average of probability given to that class. Suppose given some input to three models, the prediction probability for class A = (0.30, 0.47, 0.53) and B = (0.20, 0.32, 0.40). So the average for class A is 0.4333 and B is 0.3067, the winner is clearly class A because it had the highest probability averaged by each classifier.

5.3 Results

5.3.1 Classification result

According to the chart below, it can be found that the accuracy, precision and F_1 of Voting Classifier are the highest.

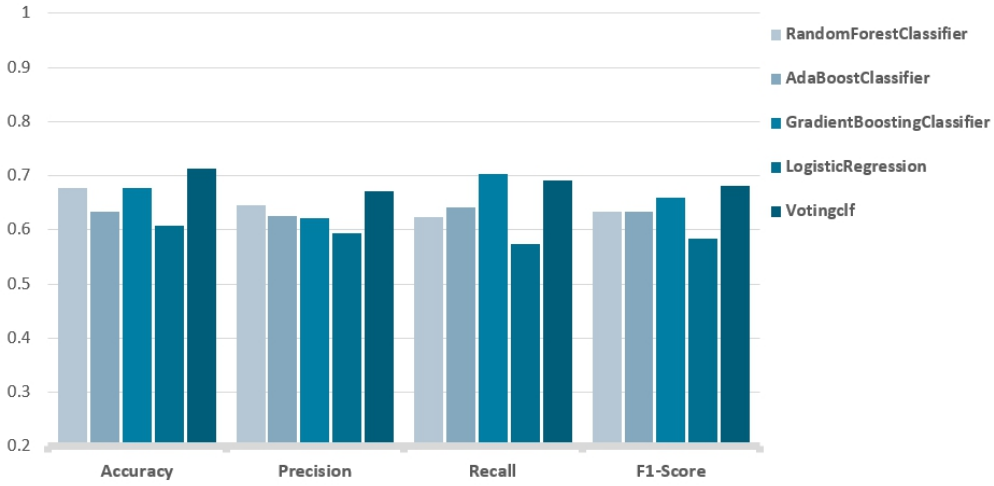

Fig. 4.1. Comparison of classification results of each model

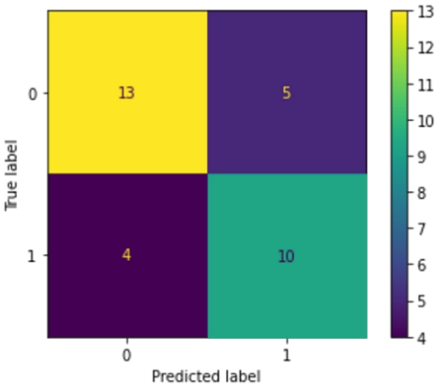

Fig. 4.2. Confusion matrix of the final model

5.3.2 Regression result

We plot the scatter plots of some of the regression results, and we can see from the graphical features that the fit of votingregressor is better.

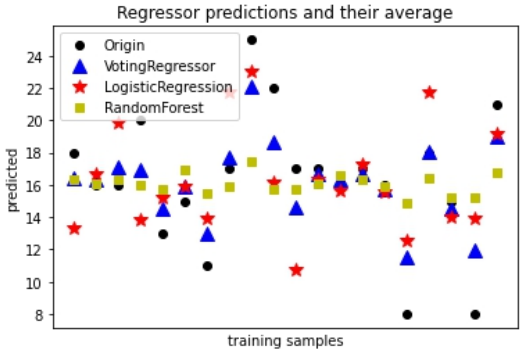

Fig. 4.3. Comparison of scatter plots of regression results for each model

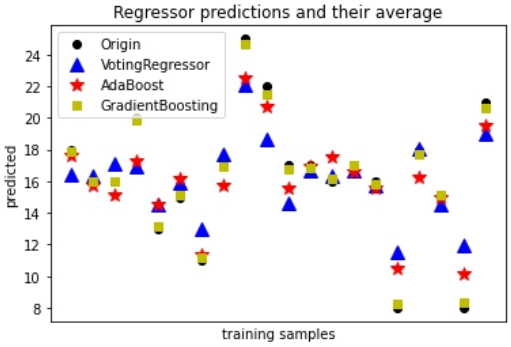

Fig. 4.4. Comparison of scatter plots of regression results for each model

For the evaluation metrics part of the regression model, it is also found that the votingregressor has the lowest mse and the highest r2score.

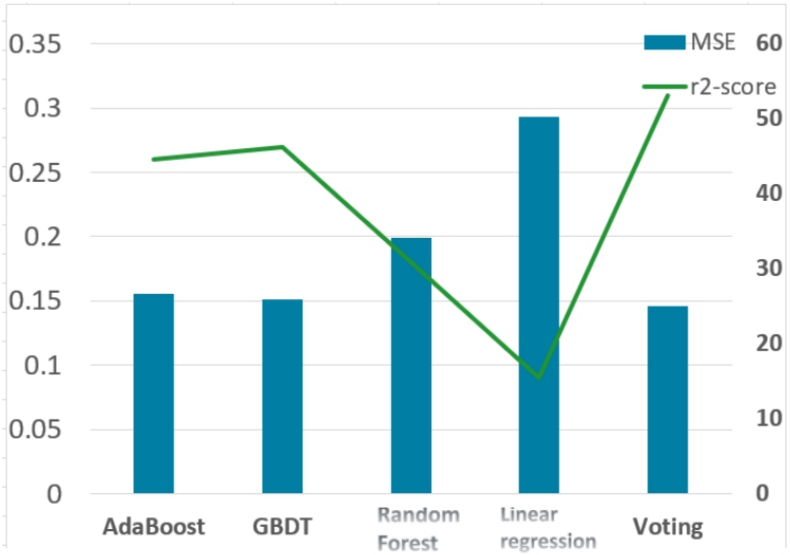

Fig. 4.5. Comparison of mse and r2score of each model

6. Semi-Supervised Learning

6.1 Introduction to semi-supervised learning

Traditional supervised learning usually requires a large number of labeled data samples as training examples, but in many real-world problems, it is easy to obtain a large number of data samples, but it is labor-intensive to provide labels for the data. For example, in computer-aided medical image analysis, a large number of medical images can be obtained from hospitals, but it is unrealistic to expect medical experts to identify all the lesions in the images. So, when only a small amount of labeled data is available, can the learning performance be improved by using a large amount of unlabeled data? In this case, semi-supervised learning is more suitable for real-world applications and has become a major research hotspot in machine learning in the last decade or so. This method can improve the learning performance by exploiting a large amount of unlabeled data when only a small amount of labeled data is available.

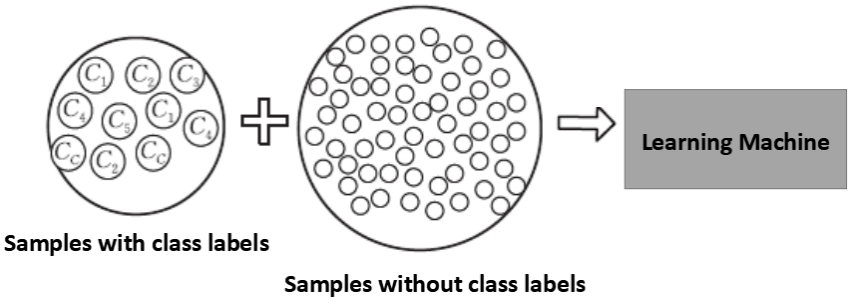

Fig. 5.1. Semi-supervised learning example diagram

6.2 Semi-supervised support vector machines (TSVM)

6.2.1 Introduction of TSVM

If there is a large amount of unlabeled data in the dataset (as in Figure 5.2), how should the decision boundary be determined? A decision boundary learned using only labeled data (as in Figure 5.2) will cut through the dense unlabeled data, and if we assume that the two classes are completely separate, then that decision boundary is not what we want; the decision boundary we want is the solid black line in Figure 5.3.

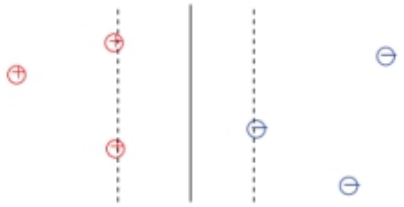

Fig. 5.2. SVM decision boundary

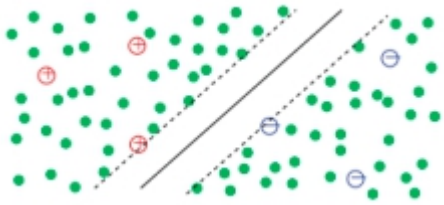

Fig. 5.3. TSVM decision boundary

6.2.2 Establishment of TSVM algorithm

SVM in supervised learning tries to find a division hyperplane that maximizes the interval between the two support vectors, i.e., the "maximum division interval" idea. For semi-supervised learning, S3VM considers that the hyperplane needs to pass through a region of low data density.

The objective function of the complete S3VMs can be written:

$$\min_{w,b} \sum_{i=1}^{l} \max\left(1 - y_i\left(\mathbf{w}^{\top}\mathbf{x}_i + b\right), 0\right) + \lambda_1 \|\mathbf{w}\|^2 + \lambda_2 \sum_{j=l+1}^{l+u} \max\left(1 - \left|\mathbf{w}^{\top}\mathbf{x}_j + b\right|, 0\right) \quad (5\text{-}1)$$

Subject to.

$$\frac{1}{u}\sum_{j=l+1}^{l+u} \mathbf{w}^{\top}\mathbf{x}_j + b = \frac{1}{l}\sum_{i=1}^{l} y_i \quad (5\text{-}2)$$

where w is the parameter vector that determines the direction and scale of the decision boundary and b is the offset.

However, the objective function of S3VMs algorithm is not convex, i.e., he has multiple locally optimal solutions, which is a computational difficulty in solving our objective function. The learning algorithm may fall into a suboptimal local optimal solution instead of a global optimal solution, and one of the research hotspots of S3VMs is to efficiently find an approximate optimal solution.

TSVM is the most famous representative of semi-supervised support vector machines, whose core idea is to try to find suitable labeled assignments for unlabeled samples to

maximize the interval after hyperplane division. The SVM is retrained based on these labeled samples, and then continuously adjusted by finding error-prone samples.

The objective function of the complete TSVM can be written as:

$$\min_{\mathbf{w},b,\hat{\mathbf{y}},\xi} \frac{1}{2}\|\mathbf{w}\|_2^2 + C_l\sum_{i=1}^{l}\xi_i + C_u\sum_{i=l+1}^{m}\xi_i \tag{5-3}$$

where ξ_i is the slack variable and the hinge loss is used.

The following constraints are satisfied.

$$\begin{aligned} &s.t. \\ &y_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b\right) \geqslant 1-\xi_i, \quad i=1,2,\ldots,l \\ &\hat{y}_i\left(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b\right) \geqslant 1-\xi_i, \quad i=l+1,l+2,\ldots,m \\ &\xi_i \geqslant 0, \quad i=1,2,\ldots,m \end{aligned} \tag{5-4}$$

The TSVM algorithm steps are as follows:

Input: Marked sample set $D_l = \left\{\left(\mathbf{x}_1, y_1\right), \left(\mathbf{x}_2, y_2\right), \ldots, \left(\mathbf{x}_l, y_l\right)\right\}$

Unlabeled sample set $D_u = \left\{\mathbf{x}_{l+1}, \mathbf{x}_{l+2}, \ldots, \mathbf{x}_{l+u}\right\}$

Compromise parameters C_l, C_u.

Process:

1: Train a SVM_l with D_l;

2: Using SVM_l to predict the sample in D_u, we get $\hat{\mathbf{y}} = \left(\hat{y}_{l+1}, \hat{y}_{l+2}, \ldots, \hat{y}_{l+u}\right)$

3: Initialize $C_u \ll C_l$

4: **while** $C_u < C_l$ **do**

5: Solving equation (5-3) based on $D_l, D_u, \hat{\mathbf{y}}, C_l, C_u$ yields $(\mathbf{w}, b), \boldsymbol{\xi}$

6: **while** $\exists\left\{i, j \middle| \left(\hat{y}_i\hat{y}_j < 0\right) \wedge \left(\xi_i > 0\right) \wedge \left(\xi_j > 0\right) \wedge \left(\xi_i + \xi_j > 2\right)\right\}$ do

7: $\hat{y}_i = -\hat{y}_i$

8: $\hat{y}_j = -\hat{y}_j$

9: Resolving equation (5-3) based on $D_l, D_u, \hat{\mathbf{y}}, C_l, C_u$ yields $(\mathbf{w}, b), \boldsymbol{\xi}$

10: **end while**

11: $C_u = \min\{2C_u, C_l\}$

12: **end while**

Output: Prediction results for unlabeled samples: $\hat{\mathbf{y}} = (\hat{y}_{l+1}, \hat{y}_{l+2}, \ldots, \hat{y}_{l+u})$

The operation of initializing the SVM is performed in the first step, and the unlabeled samples are unlabeled in step two. The larger the relaxation variable in the while loop indicates that the closer to the hyperplane, the easier it is to split the error, and in the eleventh step we gradually increase C_u. The output is the result of the final adjustment.

6.3 Results

6.3.1 Classification result

In this experiment, after a 12-fold cross-test, we know that the model achieves optimal results for an initial value of C_l of 1.5 and an initial value of C_u of 0.001.

In addition the optimal parameters of the SVM algorithm are as follows:

```
SVC( C=1.5,  break_ties=False,  cache_size=200,  class_weight=None,
        coef0=0.0, decision_function_shape='ovr', degree=3, gamma='scale',
         kernel='linear', max_iter=-1, probability=False,
        random_state=None, shrinking=True, tol=0.001, verbose=False)
```

Under the optimal parameters, the values of the four major evaluation indicators of the model are shown in Figure 5.4.

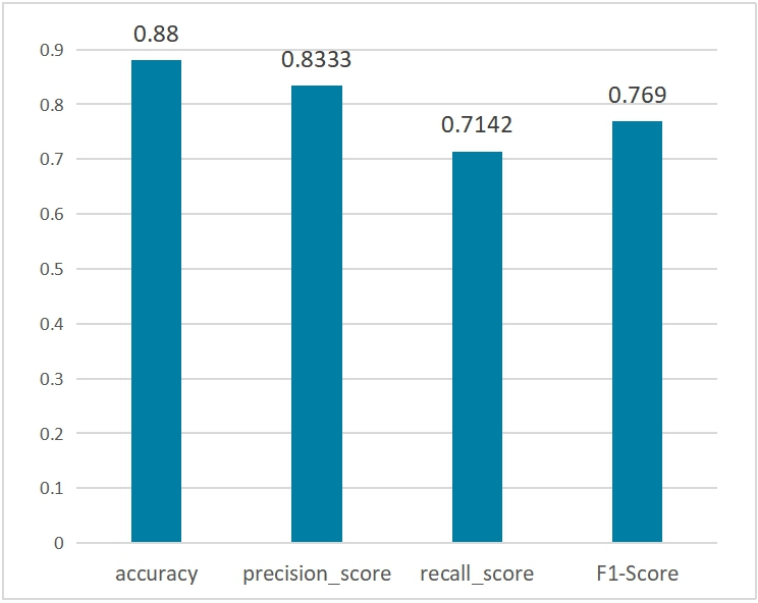

Fig. 5.4. Values of four major evaluation metrics under the TSVM semi-supervised learning optimal model

The confusion matrix of the optimal model results is shown in Figure 5.5.

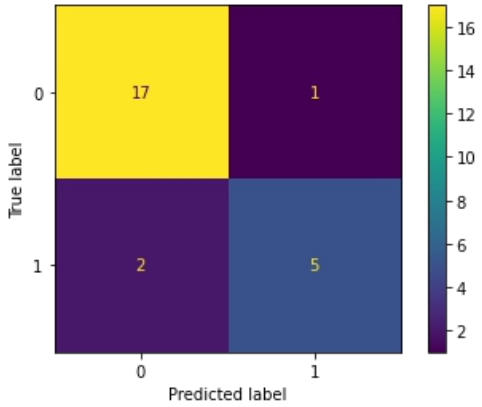

Fig. 5.5. The confusion matrix of the optimal model results

6.3.2 Regression result

The optimal parameters are the same as those shown in 5.3.1.

The MSE and R2-Score were calculated from the predicted results, and the results are shown in Table 5.1

Table 5.1 MSE and R2-Score values of regression analysis results

Evaluation Indicators	Value
MSE	16.84
R2-Score	0.24

7. Strength and weakness

7.1 Strengths

- **Explainable:** We used multiple embeddings to perform feature engineering and test the optimal data across multiple models.
- **Theoretical Support:** We both use supervised learning and semi-supervised learning to experiment and make comparisons between them.
- **Intuitive:** We have visualized processes and results to make our work better understood.

7.2 Weaknesses

- **Data Imbalance:** In the data set, there is a certain imbalance in the sample size of the two categories, which should be balanced by using data enhancement technology.
- **Temporary optimality:** In the given dataset, the ratio of unlabeled samples to labeled samples is 1:2. When we perform supervised learning or semi-supervised learning, the allocation of the training set, test set and unlabeled set is constrained by the size of the current dataset, in other words, if the given dataset has more than 10,000 samples, this problem can be ignored, however, the current dataset has only about 100 samples.

8. Conclusion

In terms of supervised learning, after a ten-fold cross-check, it can be found that the Votingclf algorithm obtains the best evaluation results in terms of both fitting and classification with an accuracy of 0.7134 and an F1-Score of 0.6841, followed by a confusion matrix and a regression scatter plot with an MSE value of 29.16 and an R2-Score value of 0.34.

In terms of unsupervised learning, after a ten-fold cross-test, it can be found that the TSVM model obtained the best evaluation results in terms of both fitting and classification when the values and values were initialized to 1.5 and 0.001, respectively, with an accuracy of 0.88 and an F1-Score of 0.769, followed by plotting the confusion matrix and regression scatter plot with an MSE value of 16.84 and an R2-Score value of 0.04. By plotting the confusion matrix and regression scatter plot, the MSE value was 16.84 and the R2-Score value was 0.24. The results show that the semi-supervised learning outperforms the supervised learning in the classification problem, and the R2-Score is not as good as

the supervised learning in the fitting problem, although the MSE value is excellent.

Finally, we hope to integrate semi-supervised learning with deep learning in the future to find the optimal model.

Reference

[1] Mingers J. An empirical comparison of pruning methods for decision tree induction[J]. Machine learning, 1989, 4(2): 227-243.

[2] Myles A J, Feudale R N, Liu Y, et al. An introduction to decision tree modeling[J]. Journal of Chemometrics: A Journal of the Chemometrics Society, 2004, 18(6): 275-285.

[3] David W Hosmer, Stanley Lemeshow. Applied logistic regression. Technometrics. 2000.

[4] Rosenblatt, Frank. x. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC, 1961.

[5] Breiman, L., Friedman, J. Olshen, R. and Stone C. Classification and Regression Trees, Wadsworth, 1984.

[6] Cortes, C. and Vapnik, V. Support vector networks. Machine Learning, 20, 273-297, 1995.

[7] Friedman J H. Greedy function approximation: a gradient boosting machine[J]. Annals of statistics, 2001: 1189-1232.

[8] Tsoumakas G, Katakis I, Vlahavas I. Effective voting of heterogeneous classifiers[C]//European Conference on Machine Learning. Springer, Berlin, Heidelberg, 2004: 465-476.

[9] Chapelle O, Scholkopf B, Zien A. Semi-supervised learning (chapelle, o. et al., eds.; 2006)[book reviews][J]. IEEE Transactions on Neural Networks, 2009, 20(3): 542-542.

[10] Xu P, Lu W, Wang B. A semi-supervised learning framework for gas chimney detection based on sparse autoencoder and TSVM[J]. Journal of Geophysics and Engineering, 2019, 16(1): 52-61.