Experiment with non-linear classifiers

Problem 2

1. Introduction to the model

1.1 Xgboost

XGBoost(eXtreme Gradient Boosting), also known as extreme gradient Boost tree, is one of boosting algorithm. For classification or regression problems, the effect is very good. In a variety of data competition, but also widely used in the industry, mainly because of its excellent effect, easy to use, fast speed and other advantages.xgb is an implementation way of boosting algorithm, which aims to reduce the deviation, that is, reduce the error of the model. Therefore, it adopts multiple base learners, each of which is relatively simple to avoid overfitting. The next learner is to learn the difference between the results of the previous base learner and the actual value. Through the learning of multiple learners, the difference between the model value and the actual value is constantly reduced.

1.2 Random forest

As in bagging, we build several decision trees on bootstrapped training samples. But when building these decision trees, each time a split in a tree is considered, a random sample of $p < d$ features (predictors) is chosen as split candidates from the full set of all d features. The split is allowed to use only one of those p features. A fresh sample of features is taken at each split.

1.3 SVM

SVM is a type two classification model. Its basic idea is to find the separation hyperplane with the largest interval in the feature space so that the data can be efficiently dichotomy. Specifically, there are three cases (it is a linear model without kernel function, but it will be upgraded to a nonlinear model after adding) : When the training samples are linearly separable, a linear classifier, namely linear separable support vector machine, is learned by maximizing the hard interval. When the training data is approximately linearly scalable, relaxation variables are introduced and a linear classifier, namely linear support vector machine, is learned by maximizing the soft interval. When the training data is linearly non-separable, the nonlinear support vector machine is learned by using kernel technique and soft interval maximization.

2. Model training strategy

For each model, we will use hyperparameter tuning to select the optimal combination of parameters for each model. The training steps are as follows:

(1) Select the hyperparameter combination and value range of each model;
(2) Initialize the model;
(3) gridsearchcv method was used and 5-fold cross validation was used to train each parameter combination and get the score;
(4) The best parameter combination was obtained according to the score ranking;
(5) Using the model constructed by the optimal parameter combination, the training set and the test set were respectively predicted and the classification error was obtained.

3. Results of hyperparameter training

3.1 Xgboost

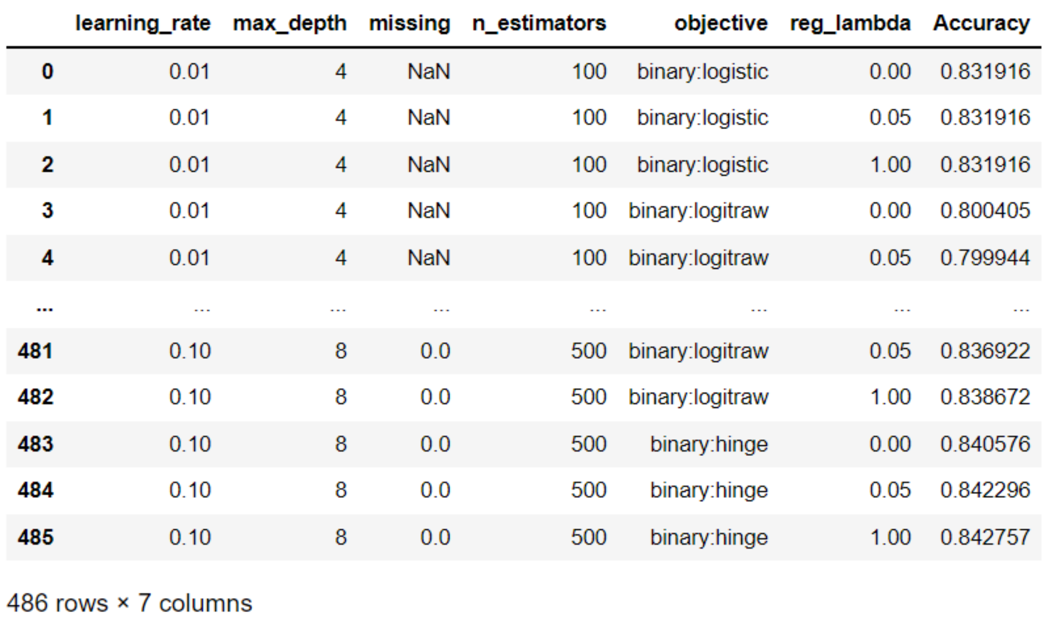
3.1.1 Parameter combination

Parms_grid = {' max_depth: [4, 6, 8], 'learning_rate: [0.01, 0.05, 0.1].
'missing' : [None, 0], 'n_estimators: [100, 300500].
'reg_lambda: [0.0, 0.05, 1.0],' objective ': [' binary: logistic', 'binary: logitraw', 'binary: hinge']}

We selected the following parameters: max_depth (default: 3), learning_rate (default: 0.1), missing (default: None), n_estimators (default: 100), reg_lambda (default: 1) objective (Default: binary:logistic
)

3.1.2 Training results

Average cross-validation score for each parameter combination:



	learning_rate	max_depth	missing	n_estimators	objective	reg_lambda	Accuracy
0	0.01	4	NaN	100	binary:logistic	0.00	0.831916
1	0.01	4	NaN	100	binary:logistic	0.05	0.831916
2	0.01	4	NaN	100	binary:logistic	1.00	0.831916
3	0.01	4	NaN	100	binary:logitraw	0.00	0.800405
4	0.01	4	NaN	100	binary:logitraw	0.05	0.799944
...	...	...	...	...	...	...	...
481	0.10	8	0.0	500	binary:logitraw	0.05	0.836922
482	0.10	8	0.0	500	binary:logitraw	1.00	0.838672
483	0.10	8	0.0	500	binary:hinge	0.00	0.840576
484	0.10	8	0.0	500	binary:hinge	0.05	0.842296
485	0.10	8	0.0	500	binary:hinge	1.00	0.842757

486 rows × 7 columns

Fig1 Training results of model xgboost

The figure above shows the average score of different parameter combinations under the five-fold cross verification. It can be seen from the figure that there are 486 different parameter combinations.

3.1.3 Optimal parameter combination

After 5-flod cross-validation, with accuracy as the scoring standard, the optimal parameter combination is finally obtained:

{'learning_rate': 0.05, 'max_depth': 4, 'missing': None, 'n_estimators': 500, 'objective': 'binary:logistic', 'reg_lambda': 0.0}

3.1.4 Best model verification

The 5-fold cross validation error and prediction error of the optimal model on the training set are:

Cross_val_error: 0.14984177364416884

Training error: 0.13804858573139644

The prediction error of the optimal model on the test set is:

Accuracy test_error: 0.14691972237577544

3.2 Random forest

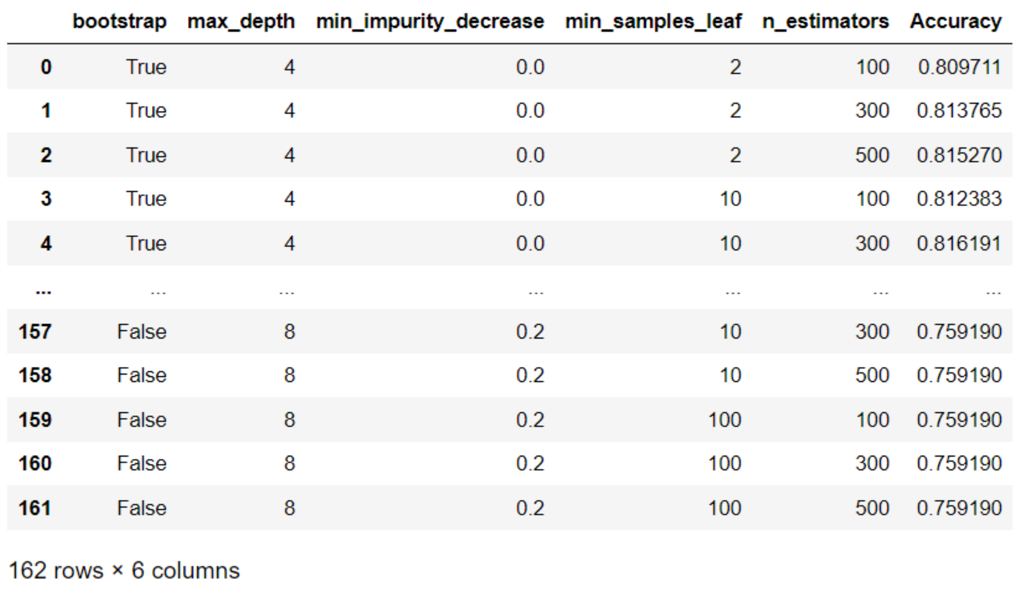
3.2.1 Parameter combination

parms_grid={'max_depth':[4, 6, 8],'bootstrap':[True, False],'n_estimators':[100,300,500],
'min_impurity_decrease':[0.0, 0.1,0.2],'min_samples_leaf':[2, 10, 100]}

We selected the following parameters: max_depth (default: None), bootstrap (default: True), n_estimators (default: 100), min_impurity_decrease (default: Decrease) 0.0), min_samples_leaf (default: 1)

3.2.2 Training results:

Average cross-validation score for each parameter combination:



	bootstrap	max_depth	min_impurity_decrease	min_samples_leaf	n_estimators	Accuracy
0	True	4	0.0	2	100	0.809711
1	True	4	0.0	2	300	0.813765
2	True	4	0.0	2	500	0.815270
3	True	4	0.0	10	100	0.812383
4	True	4	0.0	10	300	0.816191
...	...	...	...	...	...	...
157	False	8	0.2	10	300	0.759190
158	False	8	0.2	10	500	0.759190
159	False	8	0.2	100	100	0.759190
160	False	8	0.2	100	300	0.759190
161	False	8	0.2	100	500	0.759190

162 rows × 6 columns

Fig2 Training results of model Random forest

The figure above shows the average score of different parameter combinations under the five-fold cross verification. It can be seen from the figure that there are 162 different parameter combinations.

3.2.3 Optimal parameter combination

After 5-flod cross-validation, with accuracy as the scoring standard, the optimal parameter combination is finally obtained:

```
{'bootstrap': True, 'max_depth': 8, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 10,
'n_estimators': 100}
```

3.2.4 Best model verification

The 5-flod cross validation error and prediction error of the optimal model on the training set are:

```
Cross_val_error: 0.16418392660907632

Training error:    0.16258714412948005
```

The prediction error of the optimal model on the test set is:

```
Accuracy test_error: 0.16325778514833245
```

3.3 SVM

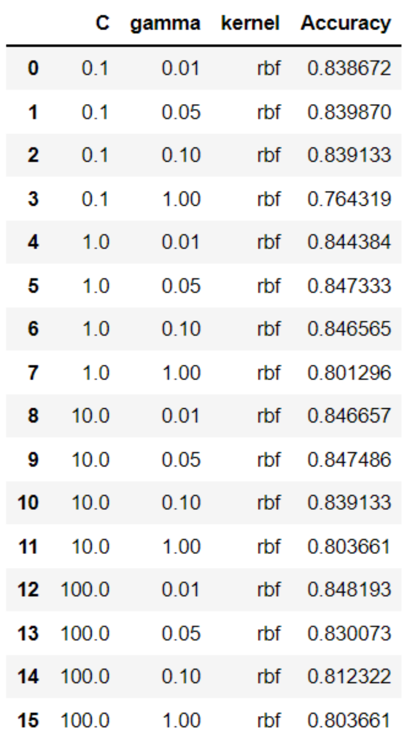
3.3.1 Parameter combination

```
parms_grid={'kernel':['rbf'],'gamma':[0.01, 0.05, 0.1,1],
                  'C':[0.1,1,10,100]}
```

The parameters selected are: kernel (default: rbf), gamma (default: scale), C (default: 1.0)

3.3.2 Training results:

Average cross-validation score for each parameter combination:



	C	gamma	kernel	Accuracy
0	0.1	0.01	rbf	0.838672
1	0.1	0.05	rbf	0.839870
2	0.1	0.10	rbf	0.839133
3	0.1	1.00	rbf	0.764319
4	1.0	0.01	rbf	0.844384
5	1.0	0.05	rbf	0.847333
6	1.0	0.10	rbf	0.846565
7	1.0	1.00	rbf	0.801296
8	10.0	0.01	rbf	0.846657
9	10.0	0.05	rbf	0.847486
10	10.0	0.10	rbf	0.839133
11	10.0	1.00	rbf	0.803661
12	100.0	0.01	rbf	0.848193
13	100.0	0.05	rbf	0.830073
14	100.0	0.10	rbf	0.812322
15	100.0	1.00	rbf	0.803661

Fig3 Training results of model SVM

The figure above shows the average score of different parameter combinations under the five-fold cross verification. It can be seen from the figure that there are 16 different parameter combinations.

3.3.3 Optimal parameter combination

After 5-fold cross-validation, with accuracy as the scoring standard, the optimal parameter combination is finally obtained:

{'bootstrap': True, 'max_depth': 8, 'min_impurity_decrease': 0.0, 'min_samples_leaf': 10, 'n_estimators': 100}

3.3.4 Best model verification

The 5-fold cross validation error and prediction error of the optimal model on the training set are:

Cross_val_error: 0.1518072247114164

Training error: 0.13718866128190166

The prediction error of the optimal model on the test set is:
Accuracy test_error: 0.15109637000184262

Clustering

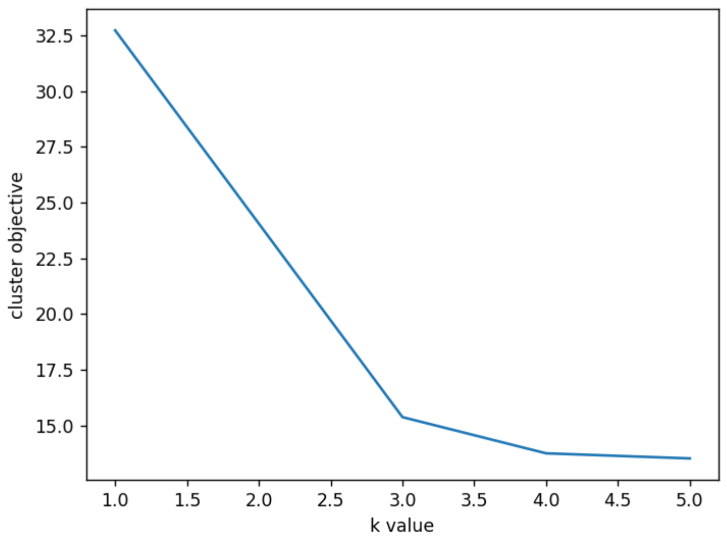


Fig4 Objectives value vs k values

As shown in the figure above, we draw a graph of target loss changing with the value of k. It can be seen from the graph that as the number of clusters increases, the cluster loss also decreases. However, we could not select the value of k corresponding to the lowest target loss, because it would easily cause overfitting. Therefore, we adopted the elbow method and selected the inflection point of the curve, namely k=3, as the optimal clustering number.

When k=3, the variation curve of target loss with the number of iterations is shown as follows:

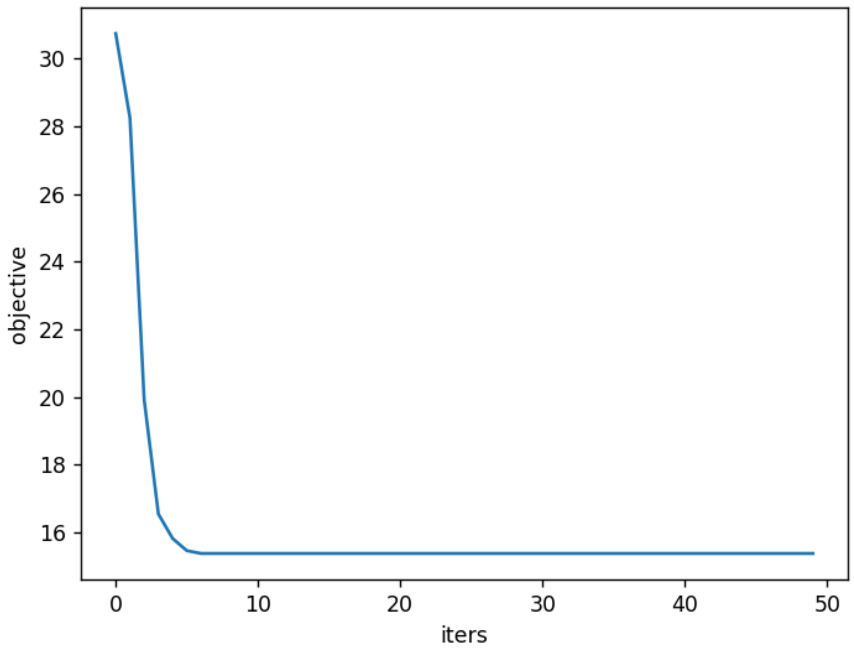


Fig5 Training results with k=3

The clustering results are as follows:

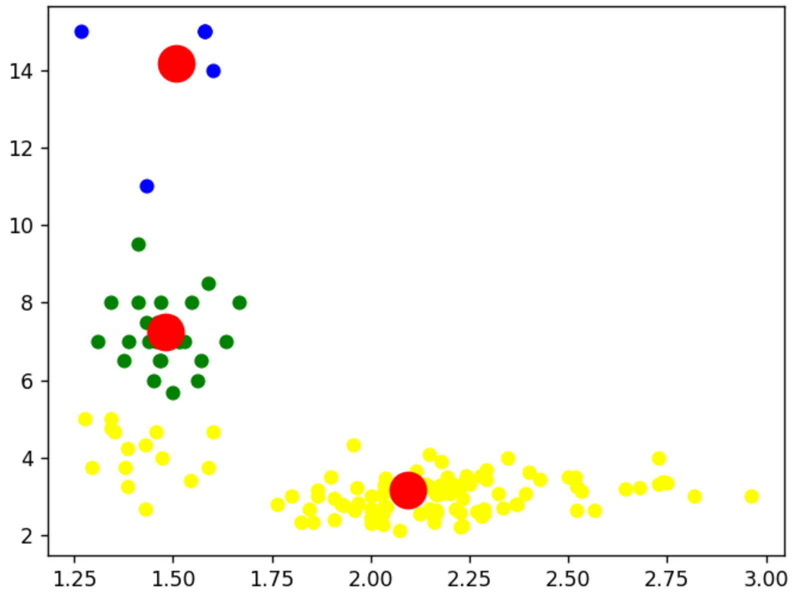


Fig6 Clustering results with k=3