Don't Overfit II Competition

Zheng Shouwen
21097982d
COMP 4432

Keywords— Machine Learning, Logistic regression, Ensemble learning

I. INTRODUCTION

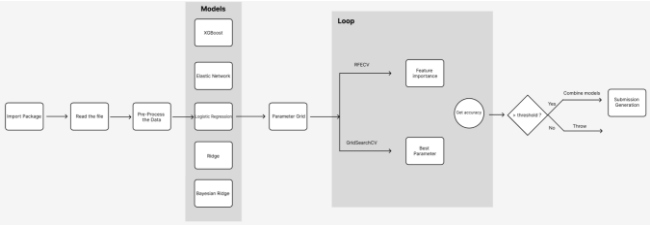
This report details my personal experience with Kaggle competition - Don't Overfit II. I focus on reducing the risk of overfitting through a series of methods, including logistic regression models, boosting models, etc. To obtain a better model, a series of operations such as parameter search, iterative optimization, and result optimization were also carried out.

Team Name: Zheng Peter
Final Rank: Around 900
Final Score: Private Score: 0.810; Public score: 0.831.
URL：
https://www.kaggle.com/code/zhengpeter/21097982d-don-t-overfit-2

The whole process is divided into 6 stages:

- Python library import
- Read Dataset
- Preprocess Dataset
- Model Building
- Model Training
- Result Processing & File Export

Here is the Process Flow of the whole Project:



(Figure1. Process Flow)

II. ALGORITHM FLOW

A. Python library import

```
import pandas as pd
import numpy as np

from sklearn.linear_model import ElasticNet
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import BayesianRidge
from sklearn.linear_model import Ridge
import xgboost as xgb

from sklearn.feature_selection import RFECV
from sklearn.preprocessing import RobustScaler
from sklearn.model_selection import StratifiedShuffleSplit, GridSearchCV
from sklearn.metrics import roc_auc_score, r2_score, make_scorer
```

(Figure2. library import)

These libraries(NumPy, Pandas, sklearn, etc.) are used for data manipulation and the establishment of some models. as well as for evaluating model performance.

B. Read Dataset

```
train = pd.read_csv('train.csv')
test = pd.read_csv('test.csv')

X_train = train.drop(['id','target'],axis=1).values
Y_train = train['target']
test = test.drop(['id'], axis=1).values
```

(Figure3. Read Dataset)

Extract the data from the training set (train.cvs) and test set (target.cvs) and convert them into feature matrices and target matrices.

Convert these extracted matrix values into numpy arrays for subsequent operations.

C. Preprocess Dataset

```
#Full feature scaling processing to increase the robustness of the original data

Total_Data = np.concatenate([X_train , test], axis=0)
Scaled_Data = RobustScaler().fit_transform(Total_Data)
X_train = Scaled_Data[:250]
test = Scaled_Data[250:]

r2_threshold = 0.16
random_seed = 963
noise = 0.01
np.random.seed(random_seed)

# add a bit of noise to train_X to reduce overfitting
X_train += np.random.normal(0, noise, X_train.shape)
```

(Figure4. Preprocess Dataset)

This step formats the features of the training set and test set to improve the robustness of the data. At the same time, small amounts of random noise are added to enhance stability and reduce the risk of data overfitting

At the same time, define random seeds, noise values, and regularization thresholds for subsequent model construction.

D. Model Building

```
## Model building
#Elastic Network
model = ElasticNet(alpha=0.031, tol=0.01,l1_ratio = 0.7, random_state=random_seed, selection='random')

#LogisticRegression
#model = LogisticRegression(random_state=42)

#BayesianRidge
#model = BayesianRidge()

#Ridge
#model = Ridge()

#XGB - Boosting
#model= xgb.XGBRegressor(objective='reg:squarederror', n_estimators=100, learning_rate=0.1, max_depth=3)



## Parament Searching
#Elastic Network

param_grid = {
    'alpha' : [0.019, 0.022, 0.021, 0.02, 0.023, 0.024, 0.025, 0.026],
    'tol'   : [0.0012, 0.0013, 0.0014, 0.001, 0.0015, 0.0016],
    'l1_ratio': [0.8,1],
}

#define the Scoler if there is one data then set it to 0.5
def scoring_roc_auc(y, y_pred):
    try:
        return roc_auc_score(y, y_pred)
    except:
        return 0.5


robust_roc_auc = make_scorer(scoring_roc_auc)

# define recursive elimination feature selector
feature_selector = RFECV(model, min_features_to_select=12, scoring=robust_roc_auc, step=10, verbose=0, cv=30, n_jobs=-1)

predictions = pd.DataFrame()
counter = 0

def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn
```

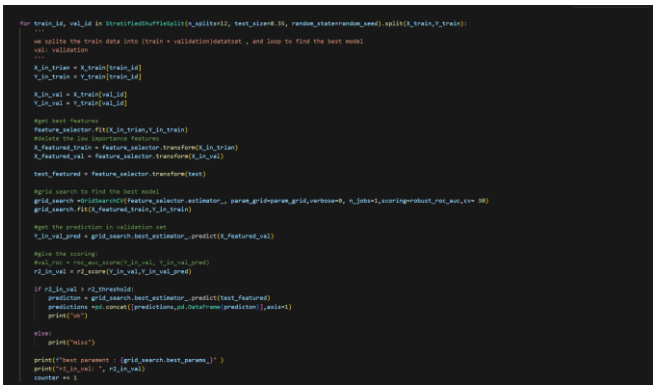
(Figure5. Model Building)

At this step, the main task is to establish different models and corresponding Grid search hyperparameter networks. There are 4 models in total: elastic network, Bayesian regression, ridge regression, logistic regression and XGBoost.

Through continuous testing, the best one is finally found Model: elastic network.

At the same time, the function of the score model (ROC) and the construction of the feature selection model (RFECV) are also defined. It should be emphasized that in RFECV, features with greater weight are screened through a large number of cross-validation.

E. Model Training



(Figure6. Model Training)

First, a specified number of data sets are randomly selected and divided into proportional training sets and validation sets for iterative model construction. In each iteration, optimized features are first selected and put into the model construction. At the same time, the network is Use grid search to obtain the optimal parameters. Use these selected features and the optimal parameter model to predict the validation set.

We use different model evaluation functions for comparison. When the accuracy and Robustness of the model reach the threshold, we accept the model of this iteration, predict the test set, and store the predicted value in the Dataframe. When this iteration When the model is not satisfied, the model will be discarded to prevent contamination of the prediction model.

F. Result Processing & File Export

(Figure7. Result Processing & File Export)

We calculated a weighted average of the predictions from various qualified models to obtain a reasonable prediction value (a score of 0-1), and predicted by comparing the bias of 0-1: greater than 0.5 is 1, less than 0. Finally, the predicted value Write to submission.cvs.

III. EXPERIMENTAL PROCESS

A. First Attempt

Use ensemble learning and random forest for testing. Through the test, it was found that the score was less than 0.5. So, it was discarded. Through analysis, it was found that appropriate regularization and data preprocessing were not performed during the training. At the same time, during the prediction process, it was consumed. It took a very long time, so improvements were made.

B. Second Attempt

After reviewing the information and discussions, we made a second attempt to integrate the model, adopting the Boosting method and using the XBG model for prediction. It is important to note that L1, L2 regularization and model preprocessing were added.

Find the most important features by fitting the model in advance and 5-fold cross-validation, and then conduct Grid search through these features to find the best parameters. Finally, predict the corresponding test set through the best hyperparameters and selected features, get the final prediction result.

In this attempt, Public Score: 0.648 Private score: 0.644 was obtained. This is a good improvement.

(Figure8. First Score)

After further reflection, I found that integrated models generally have a drawback: the model is too complex, and in the data, environment given by the competition (higher features and lower training sets), the performance may not be better than that of a simple model in resisting overfitting. Especially for Boosting, many hyperparameters need to be adjusted to obtain the best results. For Grid search, this is a huge consumption of computer resources and takes a lot of time, which is unrealistic.

C. Third Attempt

Through online discussions, I gradually gained an inspiration. If a simple classification model is used, will the anti-overfitting effect be much better? So in the third attempt, I re-wrote a logistic regression model, and the same After data preprocessing, feature selection, and hyperparameter search. The difference is that there is no built-in multiple verification function in the simple model, so I randomly divided the test set into multiple sub-data sets, trained the models separately, and finally passed the evaluation Compare the value thresholds, accept high-scoring models, and conduct test set verification. Finally, the final value is obtained through the weighted voting model.

In this attempt, different regression models were verified: Logistic Regression; Bayesian Ridge; Ridge. The best solution was found to be simple logistic regression (Public Score: 0.773 Private score: 0.789).

(Figure9. Second Score)

D. Final Attempt

Finally, the optimal solution is Elastic Network. In the entire data set, I add noise consistent with data fluctuations to prevent the model from learning other than the underlying data distribution. This helps prevent the model from overfitting.

In this attempt, I used manual-automated search methods to adjust the hyperparameters, and finally obtained Public Score: 0.831 Private score: 0.810

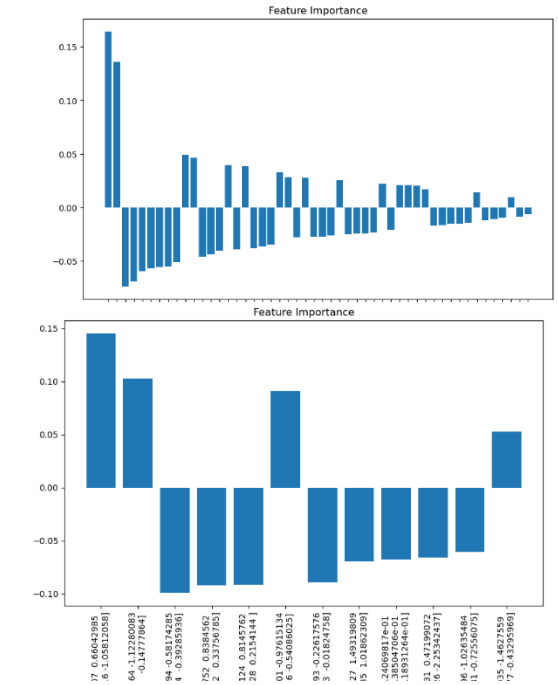
(Figure10. Third Score)

IV. Results Evaluation

In this competition, the final version obtained Public Score: 0.831 Private score: 0.810. Personally, I think it is a good version. The scores of other versions were not very ideal due to the lack of certain anti-overfitting steps. We passed MSE, MAE and ROC are used to evaluate the model and measure the balance between Bias and Variance of the model. Finally, we choose ROC to measure the accuracy and robustness of the model.

At the same time, in order to better display the model changes and accuracy, we conducted visual analysis, but it should be noted that there is no corresponding visual code in the execution code (partly because an overly bloated structure may destroy the beauty of the code). If you want to check it out, please watch the notes at the end.

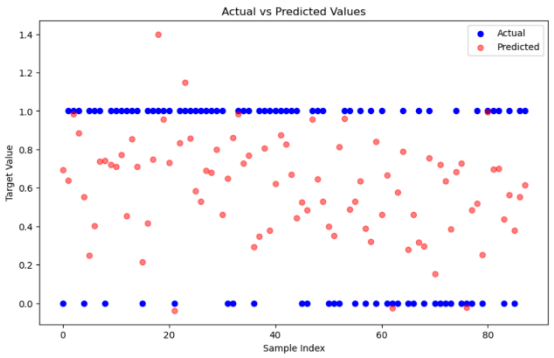
In each iteration of training, the model will first perform feature selection. The following y is the feature importance map of the accepted model. Those with larger weights will be included in feature selection first. Irrelevant weights will be discarded.



(Figure11. Feature Importance)

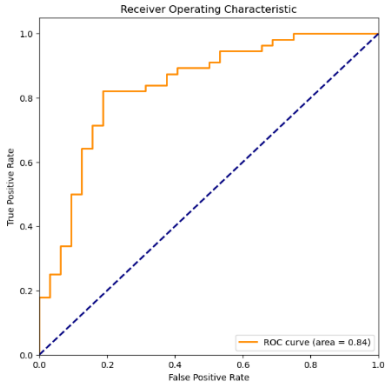
Similarly, when performing model verification, the predicted value (red) and the actual value (blue) are compared to calculate the accuracy of the model. We use 0.5 as the median point. If it is greater than 0.5, we set this value to 1, otherwise, it is 0.

Note that for predicted continuous data sets, there is a "switching function" that turns continuous values into binary (predicted values 1&0). In the subsequent confusion matrix visualization, this function is used to convert continuous data to binary data.



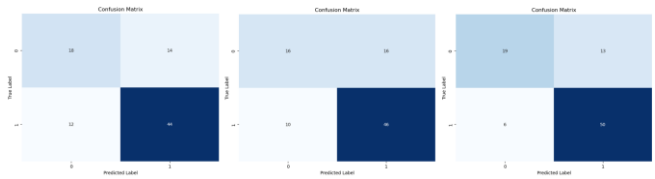
(Figure12. Comparison of Predicted and Actual Values)

It can be seen that the area under the curve is very close to 1, indicating that the accuracy of the model results in true positive results is high.



(Figure13. ROC Curve)

Through the confusion matrix, we can further visualize the absolute value of the data for which the prediction was successful. The following three are selected from the acceptor submodels that exceeded the threshold in the iterative optimization model.



(Figure14. Confusion Matrix)

V. REFLECTIONS

Through this project, I gained the deeper understanding of the importance of overfitting in small data sets and multi-feature data. Sometimes, simpler models may perform better than complex models. At the same time, , Data preprocessing, regularization, random deviation addition and other anti-overfitting steps are also very helpful in building a stable model.

Through many attempts, discussions and reflections, I gained rich machine learning experience: including model construction, hyperparameter search,

Data processing, cross-validation, model evaluation, and hyperparameter optimization using heuristic annealing algorithms (although it will be deprecated in subsequent versions due to resource consumption and poor performance)

In the process of training the model, I compared the differences between the traditional regression model and the integrated model, and found that the simple model was no worse than the integrated model after multiple iterations of integration, while the integrated model, although the effect was very good, failed because of its parameters. Rich and complex models make it difficult to find the optimal model solution.

With the continuous development of AI, the underlying theory of machine learning has received sufficient attention. As a COMP student, it is very beneficial to understand these theoretical knowledge. At the same time, I have a full understanding of my career development. Data analysis, NLP , computer vision and other directions are inseparable from machine learning. I believe there will be more subject innovations in the field of machine learning in the future