Classification of the German Credit Dataset

CS534 Final Project

Miao Yang, Chunxiao Wang

1 Introduction

In this project we are interested in the behavior of different machine learning classifiers, like logistic regression, random forest, support vector machine, neural network. We will apply all these approaches to German credit dataset. The misclassification rate will be used as the evaluation criterion.

We will consider 4 different scenarios:

- Classify the original data
- Classify the data after feature selection
- Classify the data after label filtering
- Classify the data considering label imbalance

2 Data description

There are $1,000$ observations in this dataset, which represent $1,000$ loan applicants. Each applicant has been classified to one of the two classes for the credit worthiness: good (700) or bad (300), which is our **response** here. There are 20 explanatory variables (**features**) in the dataset, including checking account status, duration, credit history, purpose of the loan, amount of the loan, savings accounts or bonds, employment duration, installment rate in percentage of disposable income, personal information, other debtors/guarantors, residence duration, property, age, other installment plans, housing, number of existing credits, job information, number of people being liable to provide maintenance for, telephone, and foreign worker status, which are used to evaluate applicant's current credit status.

Our **target** here is to provide a bank manager guidance for making a decision whether to approve a loan to a prospective applicant based on his/her profiles.

For different method under different scenarios, we will randomly select 80% observations as training data and treat the rest 20% as test set. In order to eliminate the random initialization, we will run each case 10 times, and calculate the mean misclassification error.

3 Algorithms

3.1 Logistic regression

For logistic regression, we will code the creditability of each applicant as 1 if his/her credit worthiness is good, and 0 otherwise, which will give us a binary response. For the 20 features (predictors), even though they are all numbers in the original data set, most of them are categorical variables, for example sex marital status, type of apartments, occupation, purpose, etc. Therefore we will code such features as factors, and in our logistic regression, we will create an indicator variable for each level of such features. We fit a logistic regression model, and get the mean misclassification rate.

Since there are many features (20 of them), we would like to use LASSO regularization method to get sparse solutions, which is the same as maximizing the following problem:

$$l(\beta) = -\sum_{i=1}^{n} \left\{ (1-y_i)\,\beta^\top \mathbf{x}_i + \ln\left(1+\exp\left(-\beta^\top \mathbf{x}_i\right)\right)\right\} - \lambda \sum_{k=1}^{p} |\beta_k|$$

We need first select the regularization coefficient λ, then use the selected λ to get a parsimonious logistic model. Use the new model for prediction and calculate the misclassification rate.

3.2 Random Forest

In each random forest, we will grow L decision trees and randomly select half of the features for "feature bagging" process in each candidate split, and majority voting from these trees will determine the predicted label for each observation in the testing set. For each decision tree, we will use information gain as the criterion for node determination.

3.3 Support vector machine

For SVM (support vector machine), we are applying kernel SVM, and we will use 3 different kernel functions:

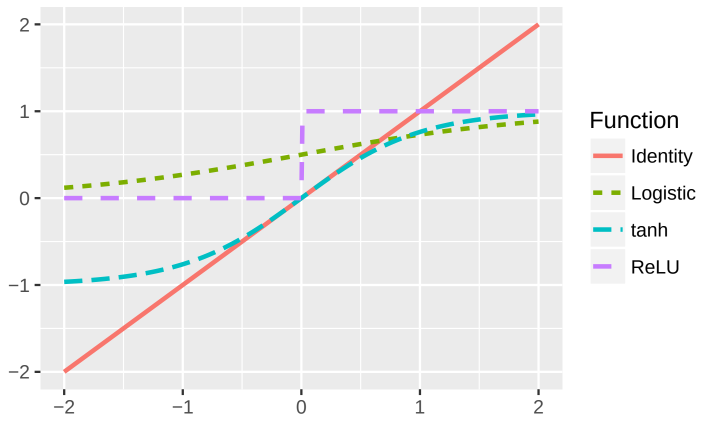
- Linear: $\kappa(a,b) = (a \cdot b)$
- Radial-basis-function (RBF): $\kappa(a,b) = \exp\left(-\frac{(a^2+b^2)}{2\sigma^2}\right)$
- Sigmoid (hyperbolic tangent function): $\kappa(a,b) = \tanh(a \cdot b)$

Another popular kernel function we don't list here is the "polynomial kernel". Within our exploration, the computation time is prohibitive. We leave this as further exploration.

3.4 Neural network

For neural network, we will use 3 hidden layers, each hidden layer has 3 nodes. Apply 4 different activation function:

- Identity function: $f(x) = x$;
- Logistic function: $f(x) = \frac{1}{1+e^{-x}}$;
- Hyperbolic tangent function: $f(x) = \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$;
- Rectified linear unit function (ReLU): $f(x) = \max(0,x)$.

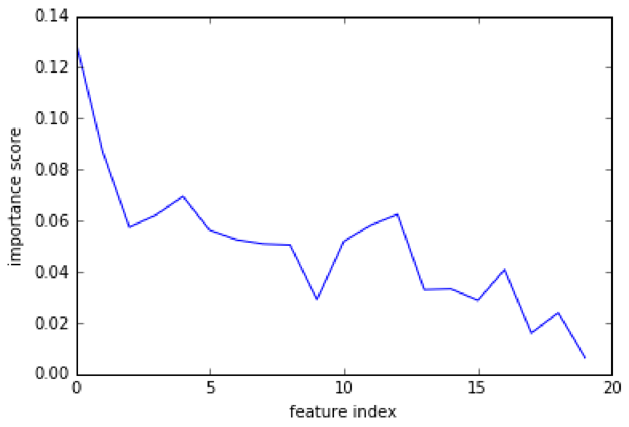


Use the stochastic gradient descent to fit neural networks.

3.5 Feature selection

There are 20 features in the data, and not all of them contribute to the accuracy of the predictive model from a pre-evaluation. We apply a feature selection procedure here and see how different approaches behave afterwords. This is done through random forest Gini importance scoring procedure. Among these random forests, we will add up the gini decreases for each features over all trees. Then an important score is computed based on the Gini decrease for each feature, and a higher important score will correspond to a larger gini decrease. We will pick the ones above the mean score. The features with higher importance scores will contribute more to the accuracy of the predictive model, otherwise it will contribute less or even decrease the accuracy. Therefore we conduct **feature selection** procedure based on each feature's importance score.

The following figure is an example we have run for this data. The x-axis is the feature index (starting from 0), and the y-axis is the score for each feature.



3.6 Label filtering

Definition: Given two examples e_i and e_j belonging to different classes, with $d(e_i, e_j)$ the distance between e_i and e_j. A (e_i, e_j) pair is called a **Tomek link** if there is no example e_l , so that $d(e_i, e_l) < d(e_i, e_j)$ or $d(e_j, e_l) < d(e_i, e_j)$. If two examples form a Tomek link, then either one of these examples is noise or both examples are borderline.

Tomek link is a popular tool to undersampling data. In our case, we think there might be some overlapping of the good credit candidates and bad credit candidates in the feature space. But we have to point out here it is hard to make plot to illustrate this concern because of the feature space dimension. The perspective is to shrink the overlapping area to reduce confusion for the classification. For implementation, we pick a random order of all the training points. For each of them, if the point together with another one forms a Tomek link, we will delete this point.

For our case, this approach has pros and cons. On the one hand, we are able to remove "noisy" samples, which will improve our classification accuracy. But on the other hand, if there is no clear boundary between two classes, many observations will be removed, which can decrease the power of classification tools.

3.7 Imbalance

For the proportion of good credit candidates is 70% over 30% bad credit candidates, a possible issue for this data set is its imbalance, meaning the two classes may not be approximately equally represented. The prediction could be less powerful when the data is imbalanced in the sense that the False Negative rate is always over estimated which should be prohibited when the cost for False Negative is large.

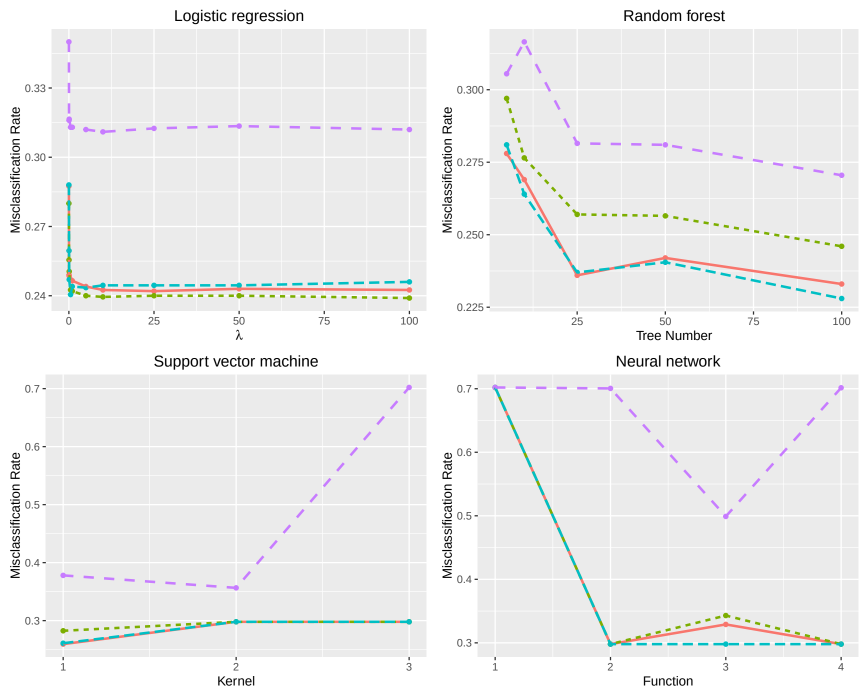
In this project we will use an oversampling method, called **SMOTE** (Synthetic Minority Oversampling TEchnique). We will manually increase sample size by creating "synthetic" examples rather than by over-sampling with replacement. We generate synthetic examples in a less application-specific manner, by operating in "feature space" rather than "data space". The method is applied to the minority class. The basic idea is to pick up a sample from minority, then choose the k nearest neighbors around the sample and compute the k vectors from the sample to these neighbors. Then the synthetic sample is chosen from the direction of the combination of the k vectors. In implementation, we use the "auto" option in the python built-in function which aims to balance the ratio of two category observations to 1. There are also undersampling techniques to pursue for this particular data set. But with the scope to retain as much information as possible, we choose the oversampling technique.

4 Result and discussion

We use the above algorithms, and summarize our result in Figure 1.

In each figure, there are four lines, all of which denote the misclassification rate for different parameters. The solid red line stands for the raw data classification; the dotted green line is classification after label filtering; the dashed blue line is what we get after feature selection; the longdashed purple line is the result after SMOTE.

Figure 1: Classification results



In the logistic regression panel, the x-axis is the regularization parameter λ in LASSO. In random forest panel, the x-axis is the number of trees in a forest. In SVM, x-axis denotes different Kernel functions: the numbers $1,2,3$ correspond to Linear, RBF and Sigmoid Kernels, respectively. In neural network, the numbers $1,2,3,4$ represent identity, logistic, ReLU, tanh activation functions, respectively.

From the figure, we can see that

- Overall, **logistic regression** and **random forest** work well in terms of misclassification rates. We will focus on the discussion of these two algorithm in what follows.
- After feature selection, the error rate increases for logistic and decreases for random forest. The reason for that is logistic regression with LASSO penalty itself is another feature selection process; we are picking the procedures twice, which is too much and will lead to an underfitted model.
- For label filtering, the logistic regression becomes better and random forest gets worse. Our label filtering approach is deleting those confusing points, which will make the linear boundary more clear, so a better logistic model will be fitted. However, random forest is a nonlinear process, label filtering will make the training sets less representative of the population, leading to a "underfitted" trees, therefore worse behaved random forest.
- For SMOTE, the error rate after SMOTE increases dramatically, meaning this is an unsuccessful trial. The reason for that, we think, is because we are "creating" fake data points based on the training set. On the one hand, this will make the two classes more balanced. On the other hand, there are several drawbacks for this synthetic examples:
  - This will make the observations in the minority class much more correlated, which may lead to overfitting.
  - We may add more confusing points at the boundary, making the classification problem more difficult.
  - Another point against SMOTE would be: the population distribution of applicants credit might be biased in nature, i.e., there are actually more people with good credit status, i.e., the data space for minority class is comparatively smaller. Generating more points will not help but adding more noise to the data.
  - Moreover, the motivation for SMOTE is to balance FP (false positive) and FN (false negative) in the confusion matrix,

5 Conclusion

In this project, we explored logistic regression, random forest, support vector machine and neural network on a specific classification problem: German credit data. The conclusion is:

- Logistic regression and random forest work pretty well, while SVM and neural network are worse-behaved.
- Label filtering helps improve logistic regression classifier, while feature selection makes random forest better.
- SMOTE doesn't provide a satisfactory solution for imbalanced problem in logistic regression nor random forest. We should look for other methods to handle the imbalance in this problem. Possible approaches include: under-sampling techniques, cost sensitive learning, etc.

Therefore if we are employees for the bank, we will recommend the random forest method with feature selection to make prediction for future candidates.

6 Reference

1. CS534 notes.
2. Friedman, J., Hastie, T., & Tibshirani, R. (2001). The elements of statistical learning (Vol. 1). Springer, Berlin: Springer series in statistics.
3. Kuhn, M., Wing, J., Weston, S., Williams, A., Keefer, C., Engelhardt, A., ... & Lescarbeau, R. (2014). caret: classification and regression training. R package version 6.0-24.