CA1

Zhimin Hou

March 2, 2020

1 Introduction

Two kinds of feature processing method are implemented and four classification methods are implemented. Binarized features is applied for Beta-binomial Naive Bayes(BNB), and log-transform features is applied for the other three algorithms(Gaussian Naive Bayes(GNB), Logistic regression(LR), K-Nearest Neighbor(KNN)). All the code is written by python(See CA1.py and readme.txt).

2 Algorithms

2.1 Beta-binomial Naive Bayes(BNB)

First, the class label is calculated using ML as $\lambda = [0.6045, 0.3955]$, calculate the feature distribution based on the Beta(α, α) given $\alpha = \{0, 0.5, 1., 1.5, \cdots, 100\}$.

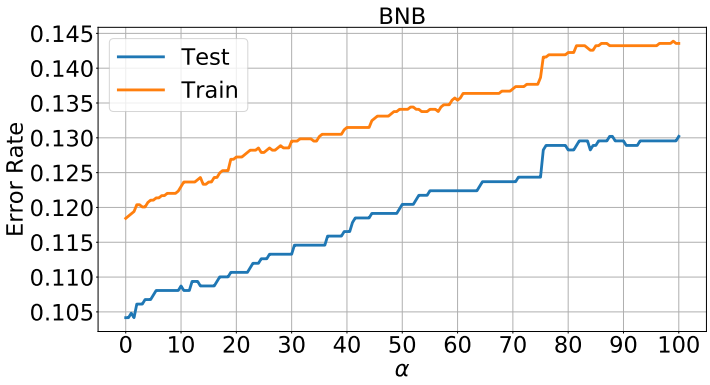

Figure 1: Training and test error rate versus α

As shown in Fig. 1, the training error rate and test error rate will become bigger as λ increase.

For $\alpha = 1, 10, 100$: **Training error rates:** 11.9413%, 12.3002%, 14.3556%; **Test error rates:** 10.4167%, 10.8724%, 13.0208%.

2.2 Gaussian Naive Bayes(GNB)

First, the class label is calculated using as $\lambda = [0.6045, 0.3955]$, the mean and var of each class $i \in [0, 1]$ as μ_i and σ_i^2.

Training error rate: 16.5742%

Test error rate: 17.1223%

2.3 Logistic regression(LR)

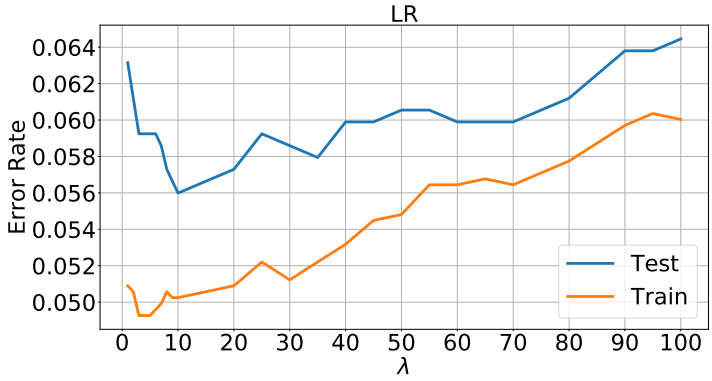

Figure 2: Training and test error rate versus λ

From Fig. 2, as the λ increase, the training error and test error will become smaller and then become bigger. Small λ may results in over-fitting easily, so the training error rate is good but test error rate is not as good as training error rate; however, the bigger λ will limit the estimation ability of the classification model.

For $\lambda = \{1, 10, 100\}$: **Training error rate:** 5.0571%, 5.6770%, 6.0033% **Test error rate**:6.120%, 5.9896%, 6.4453%

2.4 K-Nearest Neighbor(KNN)

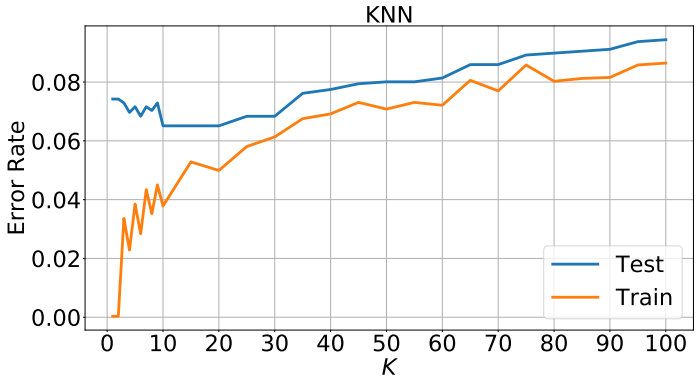

Figure 3: Training and test error rate versus K

As shown in Fig. 3, given a small K, it may results in over-fitting, the training error rate is quite low, but the test error rate is not good enough due to the noise cannot be recognized well. However, given a bigger small K, the test error rate and training error rate also will be higher due to too much useless information has been taken into account. Therefore, the K can be selected through trying different parameters. The 'optimal' parameter depends on the complexity of model to train.

For $K = \{1, 10, 100\}$: **Training error rate:** 0.0326%, 5.2855%, 8.6460%; **Test error rate**: 7.4218%, 6.5104%, 9.4401%

3 Survey

I have tried all the algorithms and compare it with the implementation in scikit-learn. Most of them can achieve the equal performance. Additionally, the normalization of features in Gaussian Naive Bayes(GNB) can increase performance.

References