Project of Machine learning for classification

You Zuo, Kexin SHAO

November 3, 2020

1 Data preprocessing

1.1 Introduction of dataset

In our experiments, we pick up only the digits with two labels ’6’ and ’8’ in the data set. There are totally 181 digit six and 174 digit eight.

. To visualize some of the samples:

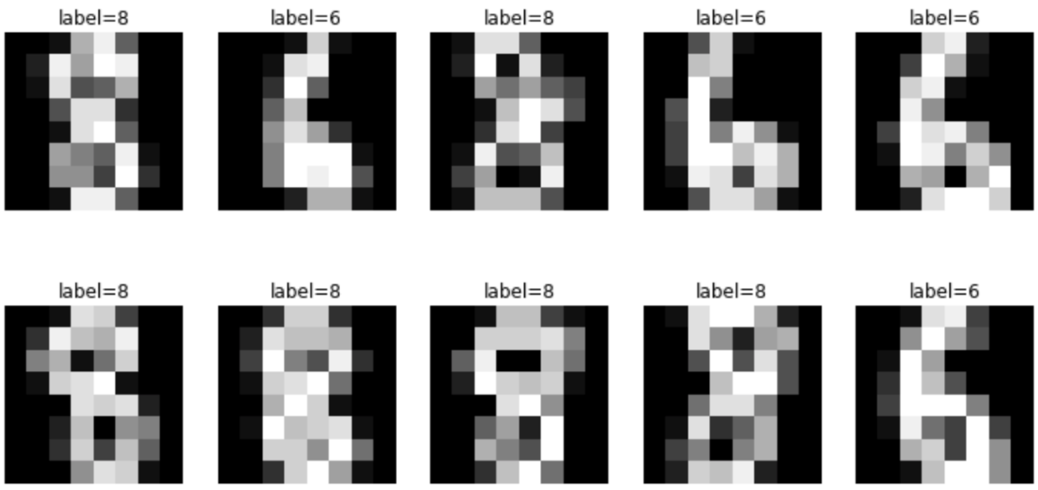

Figure 1: samples picked up randomly from our dataset

1.2 Data visualisation

We want to visualize the distribution of our samples in 2-D. In order to achieve this, we apply a PCA with two components:

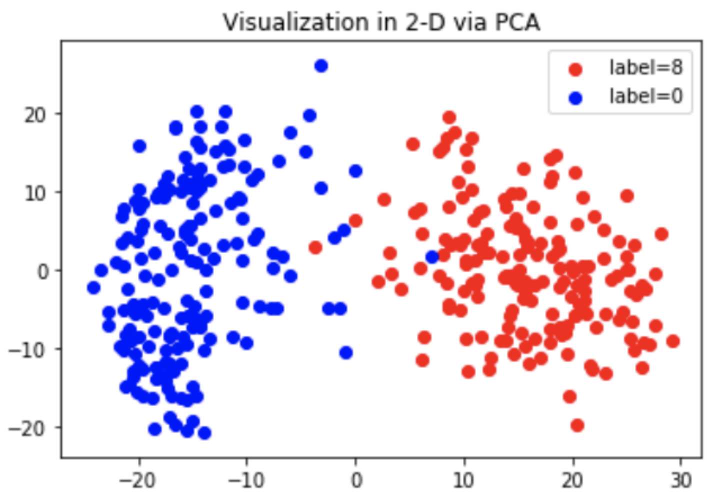

Figure 2: Visualization in 2-D via PCA

2 Construction of Models

2.1 Simple Classifiers

We would like to implement all possible simple binary classifiers first, and then choose the best one by evaluating their performances. We will train our model using a 3-fold cross-validation, and return the evaluations on training and testing sets separately.

2.1.1 Naive Bayes classifier

We want to evaluate the model performance by computing their accuracy, precision, recall, f1-score and auc. To avoid over-fitting, we implemented a 3-fold cross validation for each model.

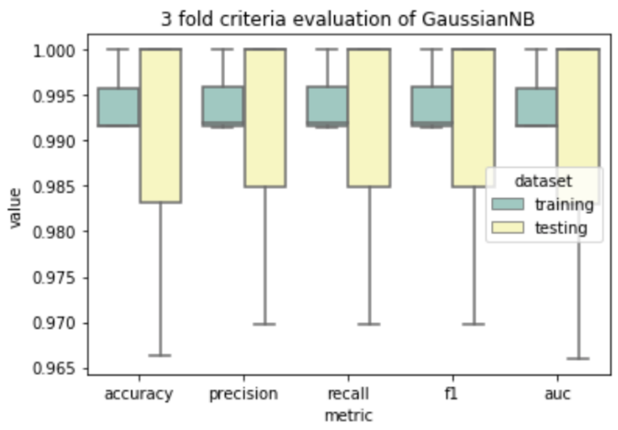

Figure 3: 3 fold criteria evaluation of GuassianNB

From the boxplots above, we can see that the Naive Bayes Classifier has very good and stable results. The values of metrics are about 0.992 for both training and testing datasets.

2.1.2 LDA and QDA

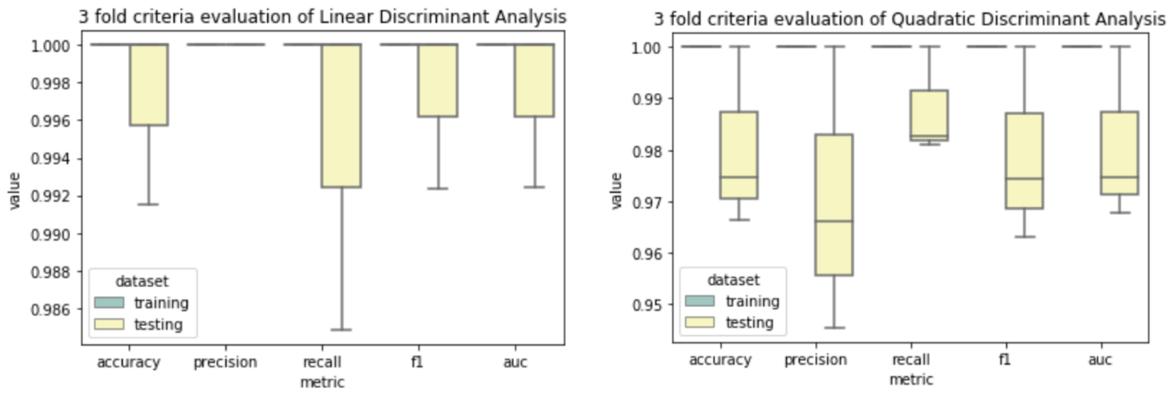

Figure 4: LDA/ QDA performance

From the results of LDA and QDA, we can notice that LDA actually has a better performance than QDA do. The main difference between them is that, with LDA, the standard deviation is the same for all the classes, while each class has its own standard deviation with QDA. To better understand our results, we visualize the decision boundary learned by LDA and QDA:

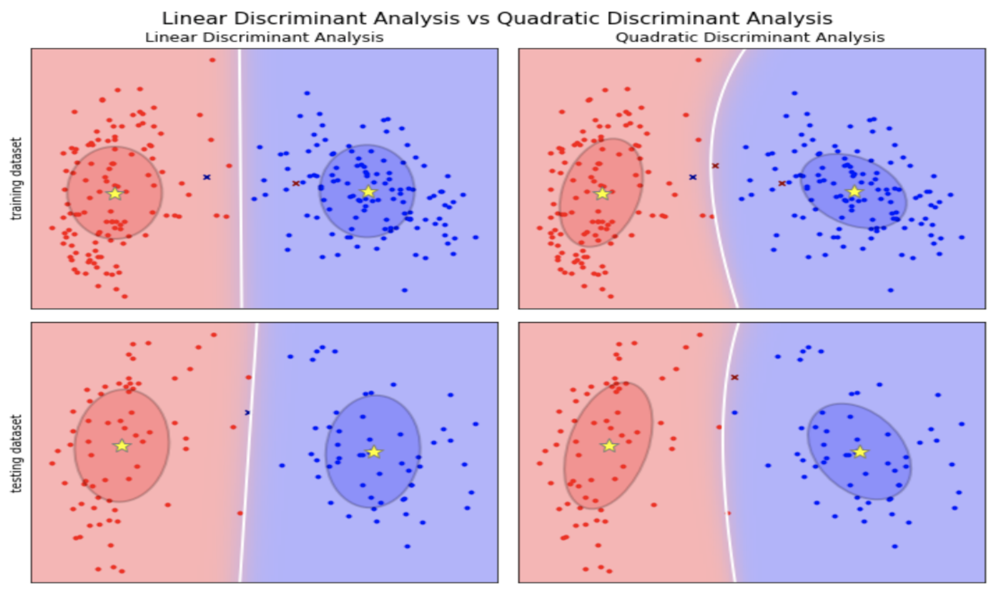

Figure 5: distribution of training and testing datasets of LDA/QDA

In the figure above, the ellipses represent the covariance and the stars represent the center. The lines refers to the decision boundary that the two classifiers have made. We can see clearly that in our case, LDA fits better. However, the curve for QDA causes some kind of inaccuracy.

2.1.3 Logistic Regression

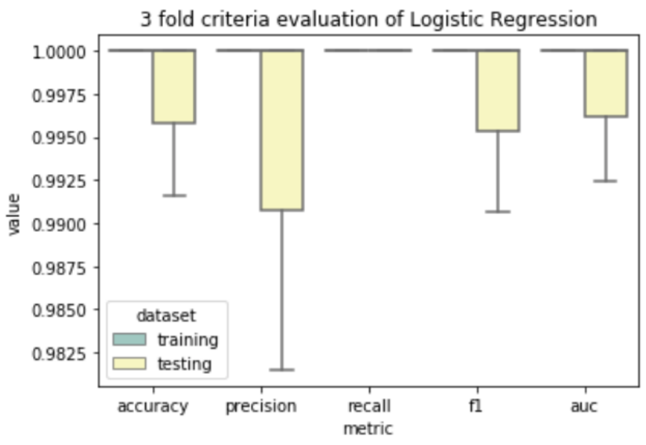

Figure 6: 3 fold criteria evaluation of logistic regression

2.1.4 K Nearest Neighbors

For knn, we have k which a hyper-parameter. We tried more possible values to pick the k which maximized the classification accuracy. Finally we chose $k = 3$, because when $k = 3$ we have the highest value of classification accuracy and a relatively very low fitting time:

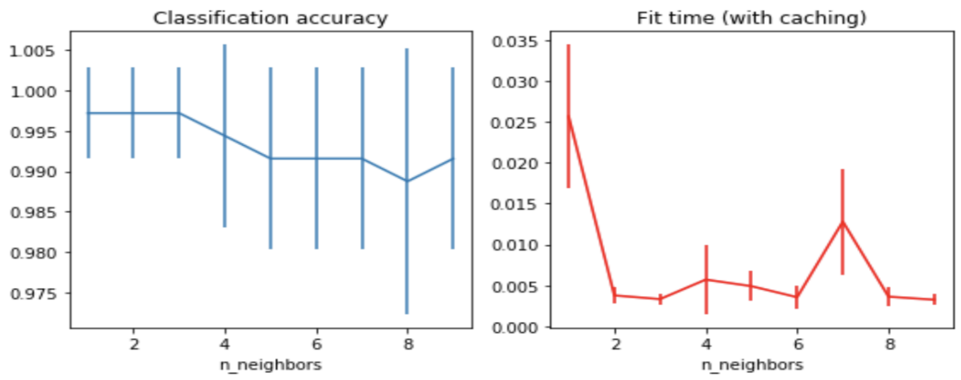

Figure 7: selection of k for knn

2.1.5 Decision Tree

In our dataset, we have 64 features (8×8 pixel matrix) for each data, but we have relatively fewer samples (less than 400). So it is necessary for us to tune some meta-parameters in order to avoid over-fitting. Here we want to tune the parameter max_leaf_nodes, and actually we can achieve this using cross validation with the help of **GridSearchCV()** function.

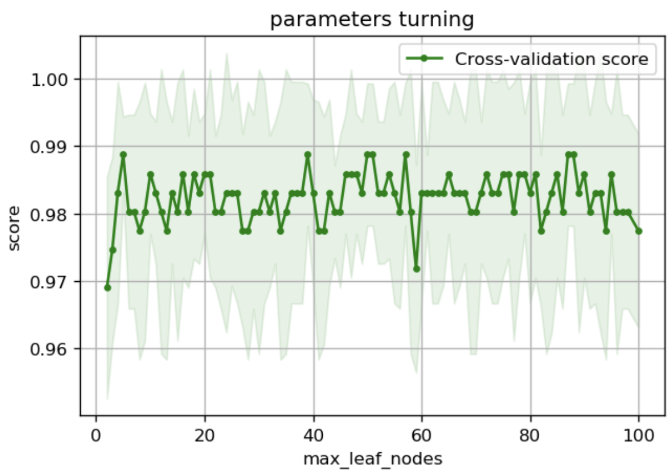

Figure 8: tuning of max_leaf_node

According to the process above, we can see that when $max_leaf_nodes = 5$, we have the best score for Decision Tree Classifier.

2.1.6 Selection of classifier

To choose a best simple classifier, we plot the ROC curves:

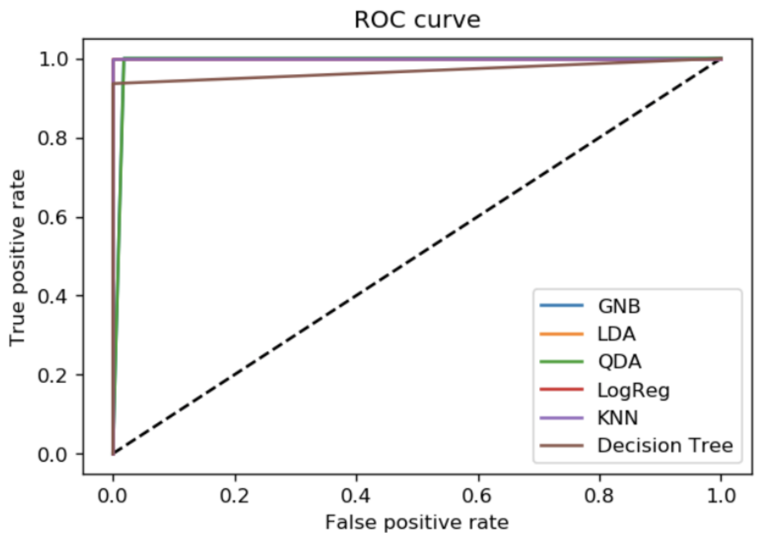

Figure 9: ROC curve

The best classifier is the one with the highest AUC score. Here is the KNN with k = 3. Later we will compete it with the other ensemble methods.

2.2 Ensemble classifiers

To do the ensemble classifiers, we choose 5 methods including bagging, RandomForest, Adaboost and Gradient Boosting. And we also have the ROC curve plot below:

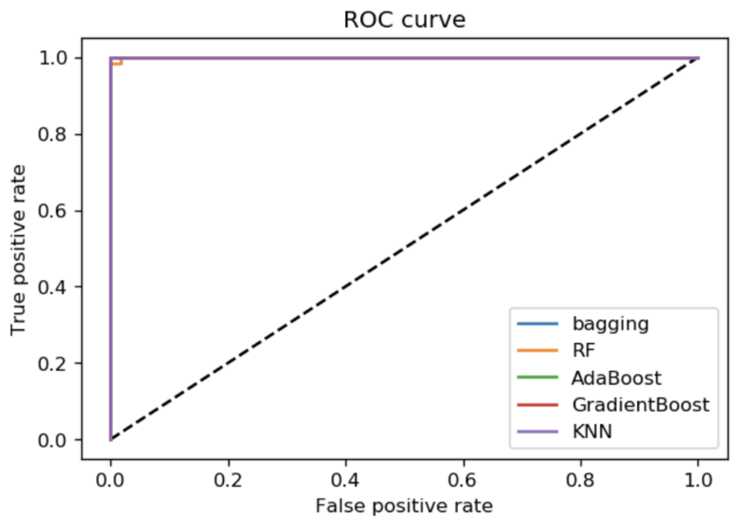

Figure 10: ROC curve

Since all of these ensemble models have pretty nice forms of roc curves, it is relatively hard to figure out which one is better than the others. After we printed their classification accuracy for comparison:

a. bagging: 0.9748

b. Random Forest: 0.9916

c. AdaBoost: 1.0

d. Gradient Boost: 0.9664

e. KNN: 0.9916

We can see that among all the ensemble models, AdaBoost has the best performance on testing dataset, then comes the Random Forest and our best simple classifier KNN. See more details in our Jupiter Notebook :)