COMP 9517 Computer Vision
T3, 2020 Project
Individual Component

Jianlong Sun
Z5186809

INTRODUCTION AND BACKGROUND

In this project, the topic to be realized is a classification problem. Because the data set already has corresponding labels, supervised learning is the best choice. Because only two labels need to be classified, this project will choose MultinomialNB and SVM as the classifiers. This project will use SIFT as a feature extraction method. After feature extraction, the feature set will be put into the trainer for training. The SIFT algorithm completely combines the steps of spot detection, feature vector generation, and feature matching search for optimization. It has reached a speed close to real-time computing. I also use some pre-processing methods such as normalization, and noise reduction.

METHOD

- **MultinomialNB and SVM**: The reason for choosing these two methods in this project is that they support classification learning and can learn very quickly. Compared with DT, they can complete the learning task faster in the same time. [1]
- **SIFT**: That is, the image is blurred by Gaussian blur in different degrees. The fine areas or points will definitely not change much, while the edges, points, corners and other areas with complex textures must change greatly, so that the points that change greatly are feature points. Using this classifier can quickly and accurately find the corresponding features. [2]
- **Normalize**: First of all, the normalization is for the convenience of subsequent data processing, and the second is to ensure that the convergence speeds up when the program is running.
- **Preprocess:** Gaussian Blur and Smoothing Filters are used in the preprocessing function to reduce noise.
- **Resize:** In order to shorten the training time, I used this function to scale the size of the picture, which can speed up the overall training speed.
- **Label Encoder**: The encoder is added to this project to speed up the training, because the space required after encoding is greatly reduced, and the processing time will also be reduced.

EXPERIMENT

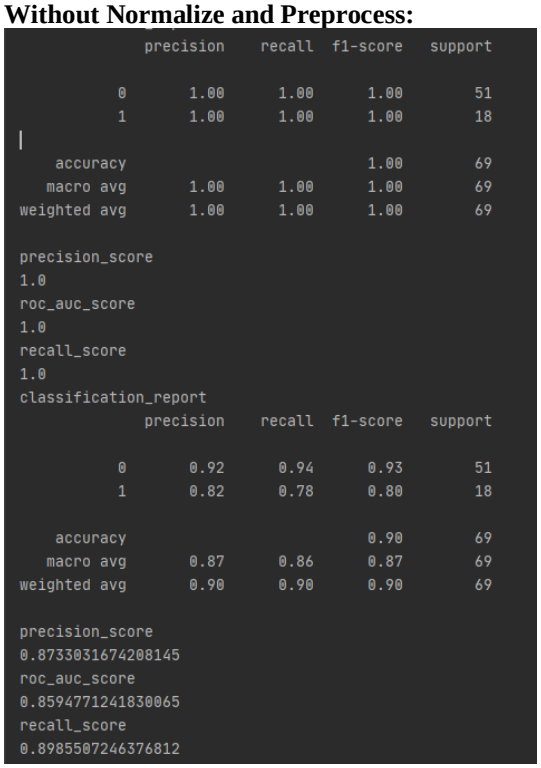
In this experiment, I will use accuracy and AUC scores to measure the quality of the model. And set the training result of MultinomialNB without any processing as the baseline, and finally compare the effects.
In this experiment, the training results of SVM are better than MultinomialNB without preprocessing and standardization. After adding preprocessing, in the two models, the training accuracy and AUC score are better than the baseline.
In terms of processing time, although some adjustments will greatly improve the accuracy, it will also greatly increase the time required. In this project, there will also be a corresponding time test.

In the overall process, this project is designed like this.

- The first step is to read the file name and store the file name and corresponding label in the dictionary.
- The second step is to read the file in the dictionary and perform corresponding preprocessing and feature extraction.
- The third step is to put the extracted features and labels in the classifier for classification.
- The last step is to obtain precision, recall and AUC based on the classification results.

RESULTS AND DISCUSSION

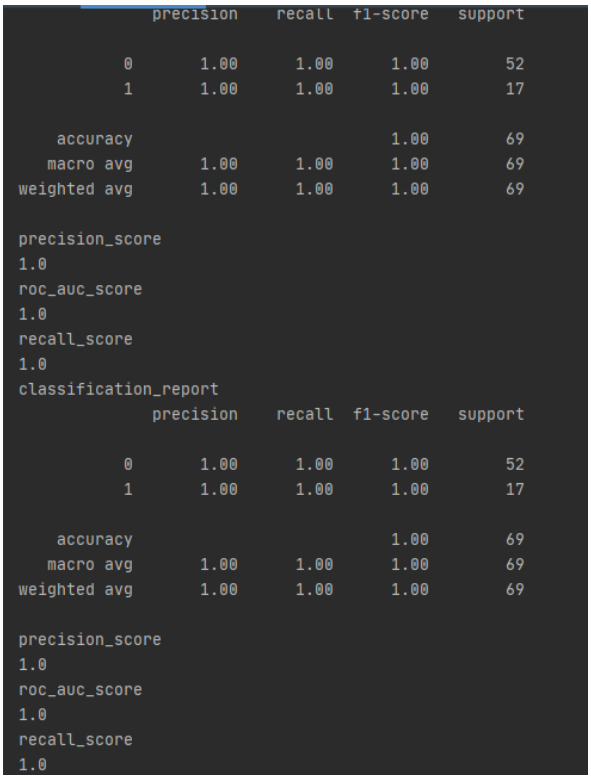
Without Normalize and Preprocess:



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        51
           1       1.00      1.00      1.00        18

    accuracy                           1.00        69
   macro avg       1.00      1.00      1.00        69
weighted avg       1.00      1.00      1.00        69

precision_score
1.0
roc_auc_score
1.0
recall_score
1.0
classification_report
              precision    recall  f1-score   support

           0       0.92      0.94      0.93        51
           1       0.82      0.78      0.80        18

    accuracy                           0.90        69
   macro avg       0.87      0.86      0.87        69
weighted avg       0.90      0.90      0.90        69

precision_score
0.8733031674208145
roc_auc_score
0.8594771241830065
recall_score
0.8985507246376812
```

In the above figure, the upper result is the result after SVM training, and the lower one is the result after MNB training. We can see that without any preprocessing, SVM performs better than MNB.
In this part, I tested the running time of the MNB model as 7.250597238540649.

With Normalize and Preprocess:



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        52
           1       1.00      1.00      1.00        17

    accuracy                           1.00        69
   macro avg       1.00      1.00      1.00        69
weighted avg       1.00      1.00      1.00        69

precision_score
1.0
roc_auc_score
1.0
recall_score
1.0
classification_report
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        52
           1       1.00      1.00      1.00        17

    accuracy                           1.00        69
   macro avg       1.00      1.00      1.00        69
weighted avg       1.00      1.00      1.00        69

precision_score
1.0
roc_auc_score
1.0
recall_score
1.0
```

As we can see in the above figure, after preprocessing, the performance of the MNB model has been greatly improved.
In this part, I tested the running time of the MNB model as 107.25265526771545. It can be seen from the results that while the performance of the B model is much higher than the baseline, it also takes much time than that.

REFERENCES

[1] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

[2] https://en.wikipedia.org/wiki/Scale-invariant_feature_transform.