Handwritten Digits Recognition Using KNN, SVM, RF And NN

Yang, J.

1 Introduction

THE handwritten digits recognition is the ability which allows computer or some other devices to recognize digits. To recognize digits, computer need to read input from scanned documents, then use classification algorithm to classify the digits shown on these documents. In this report, I will use four classifiers which are KNN, SVM, Random Forest and Neural Network to do classification job, and compare their performance.

2 Preparation

2.1 Dataset

In this classification job, I will use MNIST handwritten digit dataset, which is commonly used for traning handwritten digits recongnition, and the images form this dataset were normalized to the size of 28*28 pixel.

2.2 Construct training, testing and validation split

75% of the data are used for training, and the rest of 25% data is used for testing, furthermore, take 10% of the training data as validation set to find the value of k which can lead to the largest accuracy.

3 Model Selection

3.1 KNN

K-nearest neighbors(KNN) is a supervised algorithm which is used for solving classification tasks. Supervised learning means given a bunch of labels, for each incoming unlabeled data, they need to output their corresponding label based on the learning process from the labeled data. Hence, for every data point from test data, it will look for k nearest points from other points(training data), then assign the class in majority. Furthermore, if choose k = 1, that means the test data point is belonging to the class which its nearest point has, and conversely if choose k = n, that means the test data point is belonging to the class which has the highest occuring probability within the whole dataset.

3.2 Support Vector Machine

SVM is a supevised learning algorithm as well, it looks for the maximum margin between the two nearest points belonging the two different classes, then it draws an optimal hyperplane which classifies the region into two different categories by using Langrangian multipliers to solve this dual problem.
SVM has linear kernel and non-linear kernal, in this task, I will use LinearSVC to perform the classification of MNIST dataset.

3.3 Random Forest

Random Forest is a baggin method that uses number of trees (classifiers) to do prediction. The bagging method is used in Random Forest, the training stage of the method consists in building multiple trees, each on trained on a bootstrap sample of the original training data, meanwhile, to make the process more random, it uses an algorithm which is randomly selecting features from all features.
The speacific procedure is:
1. For N instances in the traning set, sample N cases at random with replacement
2. For M features, a subset of K features is drawn at random for each node(tree)
3. Each tree is grown to its maximum size and unpruned

3.4 Multi-Layer Neural Network

Neural Networks are modeled as collections of neurons that are connected, that means the outputs of some neurons can become inputs to other neurons, it normally consists of three layers, one is input layer, one is hidden layer(may have many), one is output layer which is used to output the class that objects belong to. Between these layers, the activation function defines the output of the node given an input or set of inputs, the common used activation functions includee Sigmoid, ReLu and Maxout.

4 Results and Discussion

For **KNN**, the accuracies are same when k = 1 to k=15, then it will go down when k > 15.

For **SVM**, I test several gamma values, and the accuracy is highest when gamma = 0.001.

For **Random Forests**, I try several groups of n_estimators and max_features, and get the highest accuracy when n_estimators=50, 100 or higher, max_features=5,7,9,11, but the accuracy will decrease if the max_features increases.

For **MLP**, I do parameter tuning on 3 parameters, hidden_layer_size, alpha and max_iter. Usually the alpha should not be too small, otherwise the speed is low and the value of max_iter should be larger.

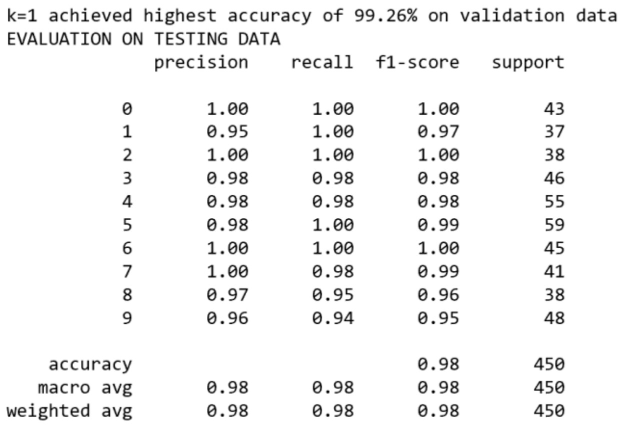
I compare models' accuracy by obesering the precision, recall, f1-score and accuracy.

Precison is the ratio of correctly predicted positive observations of the total predicted positive observations.

Recall is the ratio that TP/(TP + FN).

F1-score is 2 *(Recall * Precisio) / (Recall + Precision).
Accuracy is (TP + TN) / total.

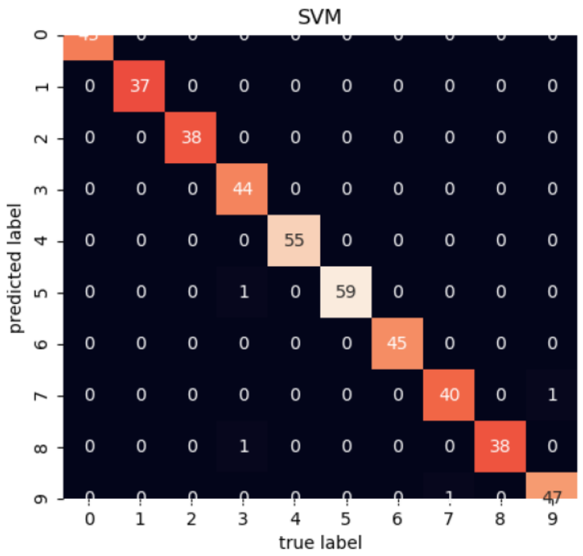
4.1 KNN



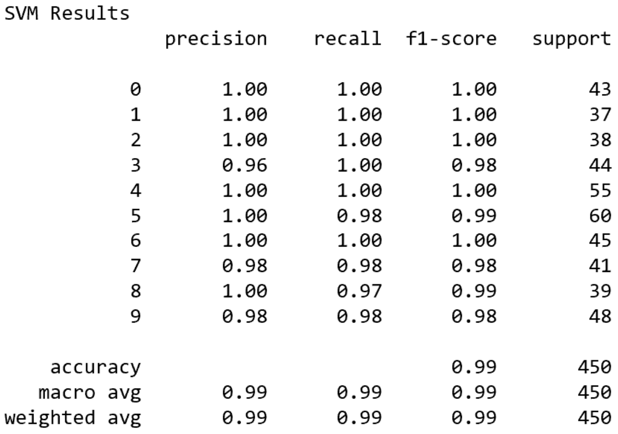
```
k=1 achieved highest accuracy of 99.26% on validation data
EVALUATION ON TESTING DATA
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        43
           1       0.95      1.00      0.97        37
           2       1.00      1.00      1.00        38
           3       0.98      0.98      0.98        46
           4       0.98      0.98      0.98        55
           5       0.98      1.00      0.99        59
           6       1.00      1.00      1.00        45
           7       1.00      0.98      0.99        41
           8       0.97      0.95      0.96        38
           9       0.96      0.94      0.95        48

    accuracy                           0.98       450
   macro avg       0.98      0.98      0.98       450
weighted avg       0.98      0.98      0.98       450
```



4.2 SVM

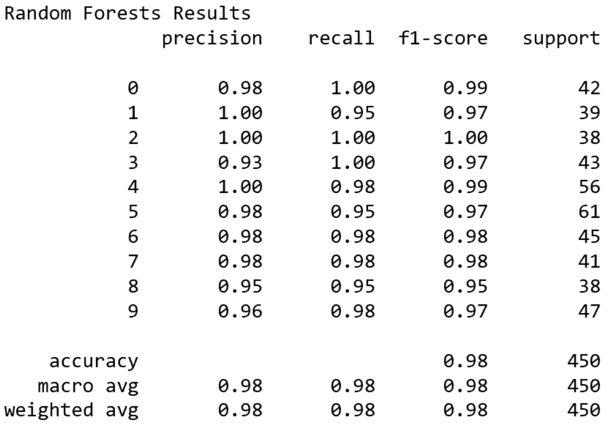


```
SVM Results
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        43
           1       1.00      1.00      1.00        37
           2       1.00      1.00      1.00        38
           3       0.96      1.00      0.98        44
           4       1.00      1.00      1.00        55
           5       1.00      0.98      0.99        60
           6       1.00      1.00      1.00        45
           7       0.98      0.98      0.98        41
           8       1.00      0.97      0.99        39
           9       0.98      0.98      0.98        48

    accuracy                           0.99       450
   macro avg       0.99      0.99      0.99       450
weighted avg       0.99      0.99      0.99       450
```

4.3 Random Forest

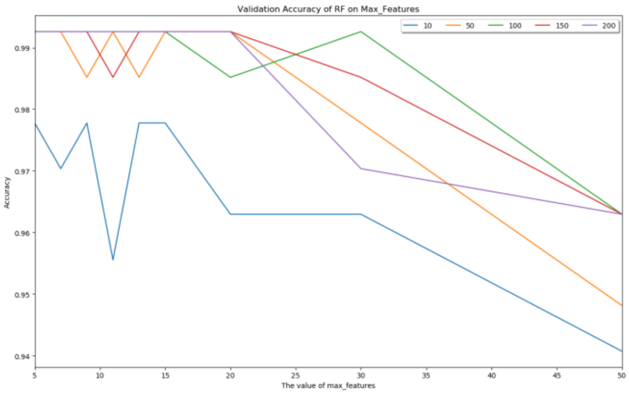


```
Random Forests Results
              precision    recall  f1-score   support

           0       0.98      1.00      0.99        42
           1       1.00      0.95      0.97        39
           2       1.00      1.00      1.00        38
           3       0.93      1.00      0.97        43
           4       1.00      0.98      0.99        56
           5       0.98      0.95      0.97        61
           6       0.98      0.98      0.98        45
           7       0.98      0.98      0.98        41
           8       0.95      0.95      0.95        38
           9       0.96      0.98      0.97        47

    accuracy                           0.98       450
   macro avg       0.98      0.98      0.98       450
weighted avg       0.98      0.98      0.98       450
```

Validation Accuracy of **RF** on Max_Features



With the increasing value of max_features, the accuracy decreases no matter the number of trees.

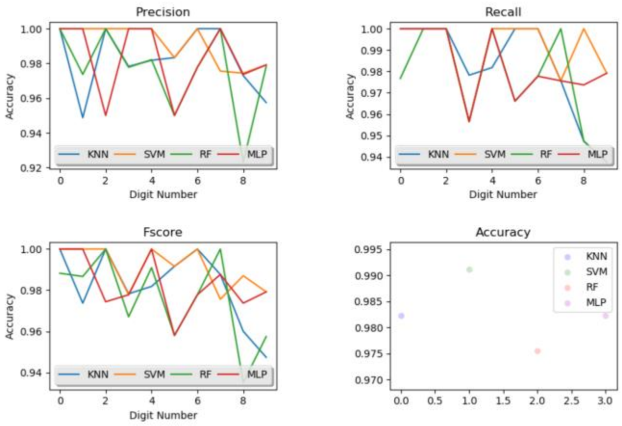
4.4 MLP

```
Multi-Layer Perceptron
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        43
           1       1.00      1.00      1.00        37
           2       1.00      0.97      0.99        39
           3       0.96      1.00      0.98        44
           4       1.00      1.00      1.00        55
           5       0.98      0.97      0.97        60
           6       0.98      0.98      0.98        45
           7       0.98      0.98      0.98        41
           8       1.00      1.00      1.00        38
           9       0.98      0.98      0.98        48

    accuracy                           0.99       450
   macro avg       0.99      0.99      0.99       450
weighted avg       0.99      0.99      0.99       450
```

4.5 Overall Comparison



5 Conclusion

From the Precision and Recall chart, **SVM** performs best on most of digits (only excepte the digit 7).

Besides, from the Accuracy chart, we can see **SVM** has the highest accuracy as well within the four models.

Hence **SVM** works well on this classification problem, then **KNN** and **MLP** rank second, and **RF** perfoms not better than the other three models.