MINIST Dataset Classification using different machine learning methods

Haotian Shi (hshi84@wisc.edu)

1. Introduction

The classification problem is an important application area of machine learning. Classification is to predict the dependent variable category based on one or more independent variables, including spam detection, customer churn prediction, and sentiment analysis, for example. MINIST dataset that includes 10 handwritten numbers (0-9) is a widely used dataset for classification. This project is to apply, compare, and analyze four algorithms (KNN, Logistic Regression, SVM, Neuron Networks) to classify the handwritten numbers in the MINSIT dataset.

2. Datasets

For the MINIST dataset, the dimension of the training data 'X' is 60000 x 784, where each sample corresponds to a label from numbers 0-9. The dimension for the testing data' X' is 10000 x 784. Due to the huge amount of data, which leads to the demanding computation, the first 2000 samples of the original dataset are used as the training set in this project, and the first 400 samples of the test set are used as the testing set in this project. The dataset was normalized after the loading process to improve the training performance and classifying performance. The normalizing process is to standardize features, which removes the mean and scales to the unit variance.

3. Investigated Algorithms

Four algorithms were applied for the classification experiment: KNN algorithm, Logistic algorithm, SVM algorithm, and Neuron Networks.

3.1 k-Nearest Neighbor (KNN)

The KNN algorithm is a theoretically mature classification method. This method's idea is: in the feature space if most of the k nearest (the nearest neighbors in the feature space) samples near a sample belong to a certain category, the sample also belongs to this category.

3.2 Logistic Regression

Logistic regression is one of the most commonly used machine learning algorithms in binary classification tasks. Logistic regression measures the relationship between the dependent variable (the label to be predicted) and one or more independent variables (features) by using its inherent logistic function (sigmoid function) to estimate the probability.

3.3 (Support Vector Machine) SVM

SVM is a discriminative classifier defined by a classification hyperplane. Given a set of labeled training samples, the algorithm will output an optimal hyperplane to classify new samples (test samples). The SVM algorithm is to find a hyperplane, and its distance to the nearest training sample should be the largest. That is, the optimal segmentation hyperplane maximizes the training sample boundary.

3.4 Neuron Networks

A neural network is a supervised machine learning model that includes neuron nodes, layers, weights, bias, activation function, and other hyper parameters. It is usually trained by applying gradient descent through a backpropagation algorithm. The advantage of the neural network is that almost arbitrarily complex classification boundaries can be realized, and the classification on the training set can be realized without error.

4. Results and discussion

4.1 KNN algorithm

For the KNN algorithm, three experiments (basic case, parameter tuning, and cross-validation) were implemented. A basic case directly applies the KNN algorithm with ten nearest neighbors for classification and evaluation, which reaches 85.25% for the testing accuracy. Then, the number of the nearest neighbor K was analyzed and optimized, with results shown in Figure 1. When K = 10, the testing accuracy reaches a maximum of 85.25%.

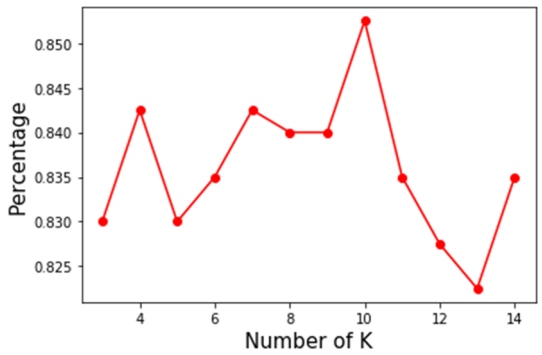

Figure 1 parameter tuning for the number of the nearest neighbor K

Thirdly, cross-validation was applied in the KNN algorithm based on the ten nearest neighbors. The training dataset was split into eight folds. The seven folds of data were used for training, and the rest dataset was used for validation. After eight iterative training processes, the model that shows the highest validation accuracy was used for the testing process. Figure 2 shows the training confusion matrix and validation confusion matrix for the best KNN model. Although the highest validation accuracy has been improved to 0.896, the test dataset's testing accuracy remains 0.8575.

```
Training confusion matrix:                    Validation confusion matrix:
 [[165   1   0   0   0   1   0   0   1   0]  [[20  0  0  0  0  2  1  0  0  0]
 [  0 190   0   1   0   0   0   0   0   1]   [ 0 28  0  0  0  0  0  0  0  0]
 [  4  18 129   3   6   1   2   8   3   1]   [ 0  0 21  0  1  0  0  1  0  0]
 [  1   3   2 147   0   3   0   1   0   3]   [ 0  0  1 25  0  3  0  0  1  1]
 [  1  11   1   1 166   1   1   1   0  12]   [ 0  1  0  0 16  2  0  0  0  0]
 [  3   2   1   8   1 126   1   0   6   4]   [ 0  1  0  0  1 26  0  0  0  0]
 [  6   3   0   0   2   1 170   0   0   0]   [ 0  0  0  0  0  0 18  0  0  0]
 [  0   9   0   1   5   0   0 166   0  17]   [ 0  1  0  0  1  0  0 23  0  1]
 [  1   7   0   2   0   7   1   0 123   4]   [ 0  0  0  0  0  3  0  0 22  2]
 [  2   1   2   2   8   0   0   0   3 165]]  [ 0  0  0  0  0  0  0  2  0 25]]
Training accuracy: 0.884                      Validation accuracy: 0.896
```

Figure 2 Training confusion matrix and validation confusion matrix for the KNN algorithm

4.2 Logistic Regression

For the logistic regression, the classification task is decomposed into ten binary classification tasks with ten classifiers. Each classifier predicts a confidence level for the target number. For each sample, the category with the highest confidence is the predicted category. The logistic algorithm experiments include a basic case and the parameter tuning (learning rate and the number of iterations) experiments.

The basic case experiment directly applies the logistic algorithm with a 0.5 learning rate and 200 iterations for the gradient descent. During the training process, each classifier's training accuracy and testing accuracy were printed, demonstrating the logistic regression algorithm's performance on classifying different numbers. The 1th classifier (which classifies number 1) shows the highest testing accuracy, 98.75%. This is because the features of number 1 are

easiest to distinguish from other numbers. The entire dataset's training accuracy is 84.8%, while the testing accuracy is only 70.5%.

The parameter tuning includes the learning rate and the number of iterations for the gradient descent, with results shown in Figure 3. The learning rate ranges from 0.00001 to 0.8, and it shows that the best learning rate is 0.00001, which leads to the highest testing accuracy, 90%. The number of iterations ranges from 10 to 2000 based on the optimal learning rate. The results show that when the number of iterations is 200, the testing accuracy reaches the highest 0.9. Specifically, the training accuracy increases as the number of iterations increases. However, as the number of iterations increases, the testing accuracy reaches the peak and then decreases. This is because too many iterations can cause an overfitting problem, which reduces the model's generalization capability.

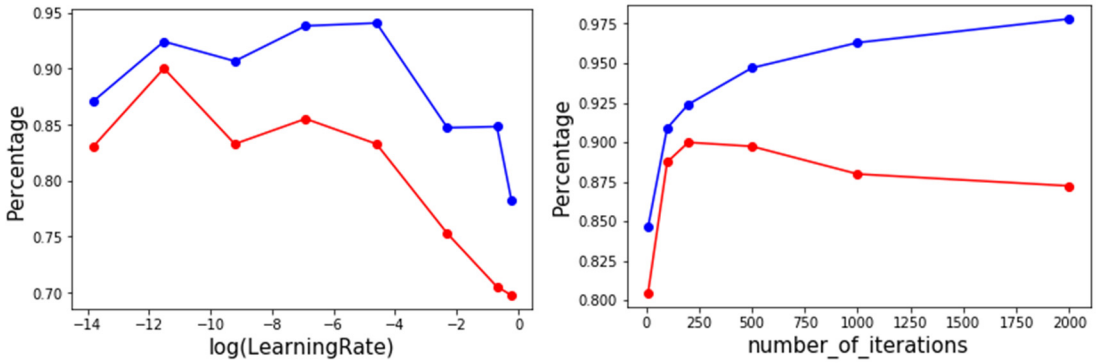

Figure 3 parameter tuning for the learning rate and number of iterations

4.3 SVM algorithm

For the SVM algorithm, a series of experiments (basic case, parameter tuning, and cross-validation) were implemented. A basic case directly uses the default settings for the experiment. The SVM's accuracy for the training dataset is 97.75%, while the SVM accuracy for the testing dataset is 90%. Then, two parameters: kernel (linear, poly, rbf), and gamma (defines how far the influence of a single training example reaches), were analyzed and optimized. The results show that the best kernel for this experiment is the "rbf" kernel, which leads to the highest testing accuracy 90%. The gamma will not impact the performance of the "rbf" SVM classifier in this case. Based on the optimal parameters, I did the cross-validation for the SVM classifier. Figure 4 shows the training confusion matrix and validation confusion matrix for the best SVM model. The validation accuracy improves to 99.6%, and the testing accuracy for the best model reaches 90.0%.

```
Training confusion matrix:
 [[167   0   0   0   0   0   1   0   0   0]
 [  0 189   1   1   0   0   0   1   0   0]
 [  0   0 172   0   1   0   0   0   1   1]
 [  0   0   0 155   0   2   0   1   1   1]
 [  0   0   0   0 192   0   1   0   0   2]
 [  0   1   1   1   1 146   2   0   0   0]
 [  0   0   0   0   0   2 180   0   0   0]
 [  0   1   0   0   1   0   0 192   0   4]
 [  0   1   1   0   0   2   1   0 140   0]
 [  3   0   0   2   1   2   0   2   0 173]]
Training accuracy: 0.9748571428571429
```

```
Validation confusion matrix:
 [[23  0  0  0  0  0  0  0  0  0]
 [ 0 28  0  0  0  0  0  0  0  0]
 [ 0  0 23  0  0  0  0  0  0  0]
 [ 0  0  0 31  0  0  0  0  0  0]
 [ 0  1  0  0 18  0  0  0  0  0]
 [ 0  0  0  0  0 28  0  0  0  0]
 [ 0  0  0  0  0  0 18  0  0  0]
 [ 0  0  0  0  0  0  0 26  0  0]
 [ 0  0  0  0  0  0  0  0 27  0]
 [ 0  0  0  0  0  0  0  0  0 27]]
Validation accuracy: 0.996
```

Figure 4 Training confusion matrix and validation confusion matrix for the SVM algorithm

4.4 neuron networks

Similar to the three previous methods, a series of experiments (basic case, parameter tuning, cross-validation) were implemented for the neuron networks. A basic case experiment applies three hidden layers with ten neurons for each layer, and the solver is 'lbfgs'. The neuron networks' accuracy for the training dataset is 99.95%, while for the testing dataset is only 0.83.

Then, the parameters: solver (sgd, adam, lbfgs), activation function (logistic, tanh, identity), learning rate, hidden size, and maximum iterations are tuned and optimized. The results of the hidden size and number of iterations are shown in figure 5. The results show that the best solver is the 'lbfgs', which leads to the highest testing accuracy, 82.75%. The gamma will not influence the performance of the "lbfgs" neuron networks classifier. For the activation function, the 'identity' activation function achieves the best performance based on the 'lbfgs' solver. The neuron networks' accuracy using the "identity" activation function for the testing dataset improves to 84.5%. The learning rate has little impact on performance. For the neuron size of the hidden layer, when the number of neurons for each hidden layer reaches 100, the accuracy achieves 87.25%, which is the highest. The number of iterations shows that the maximum iterations should be larger than the appropriate threshold and will not influence the performance when it reaches the convergence state.

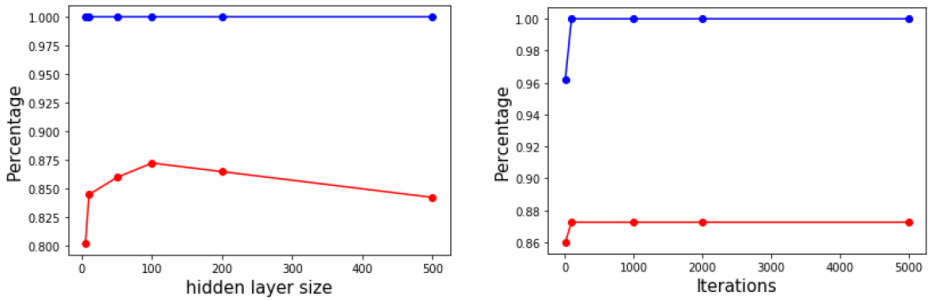

Figure 5 parameter tuning for the hidden layer size and number of iterations

Like other algorithms and experimental settings, I did cross-validation for neuron networks based on the optimal parameters, with the results shown in figure 6. Although the highest validation accuracy reaches 100%, the testing dataset's accuracy remains 87.25%.

```
Training confusion matrix:
 [[168   0   0   0   0   0   0   0   0   0]
 [  0 192   0   0   0   0   0   0   0   0]
 [  0   0 175   0   0   0   0   0   0   0]
 [  0   0   0 160   0   0   0   0   0   0]
 [  0   0   0   0 195   0   0   0   0   0]
 [  0   0   0   0   0 152   0   0   0   0]
 [  0   0   0   0   0   0 182   0   0   0]
 [  0   0   0   0   0   0   0 198   0   0]
 [  0   0   0   0   0   0   0   0 145   0]
 [  0   0   0   0   0   0   0   0   0 183]]
Training accuracy: 1.0
```

```
Validation confusion matrix:
 [[23  0  0  0  0  0  0  0  0  0]
 [ 0 28  0  0  0  0  0  0  0  0]
 [ 0  0 23  0  0  0  0  0  0  0]
 [ 0  0  0 31  0  0  0  0  0  0]
 [ 0  0  0  0 19  0  0  0  0  0]
 [ 0  0  0  0  0 28  0  0  0  0]
 [ 0  0  0  0  0  0 18  0  0  0]
 [ 0  0  0  0  0  0  0 26  0  0]
 [ 0  0  0  0  0  0  0  0 27  0]
 [ 0  0  0  0  0  0  0  0  0 27]]
Validation accuracy: 1.0
```

Figure 6 Training confusion matrix and validation confusion matrix for the neuron networks

4.5 Comparison of several algorithms

After assessing the different algorithms and optimizing the parameters, the four algorithms' performance is compared in terms of the training accuracy and testing accuracy, which is shown in table 1 below.

Models	Training Accuracy (%)	Testing Accuracy (%)
KNN	88.4%	85.8%
Logistic Regression	92.5%	90.0%
SVM	97.5%	90.0%
Neuron Networks	100%	87.25%

Table 1 Comparison results of the four algorithms

The results demonstrate that all methods show a relatively good prediction result, the accuracy of all of them is above 85%. The logistic and SVM reach the highest testing accuracy, which reaches 90%, and the training accuracy of the neuron networks is the highest with no training errors.

The advantages and limitations of these models can be demonstrated to some extent. The performance of the KNN algorithm was not great in this project. This is because the sample size is not large enough, which leads to the sample size imbalance. When the sample is unbalanced, the prediction deviation for the KNN algorithm is relatively large. However, KNN is an online technology, which means new data can be directly added to the data set without retraining, so it is simple to implement. The logistic regression is convenient for observing the sample probability score and is easy to implement, which reaches a relatively great result in this project. However, the classification problems must be linearly separable. For nonlinear features, the conversion is required. SVM can solve high-dimensional problems and is good at solving machine learning problems under small samples. Thus, it also achieves a relatively great result comparing other algorithms. However, when there are many observation samples, the efficiency is not very high, and sometimes it is difficult to find a suitable kernel function for nonlinear problems. The neuron networks achieve the highest accuracy in terms of the training samples. However, the generalization capability is not so great due to the small training samples. In addition, this is a black-box process, so the learning process between observations cannot be observed, and the output results are difficult to interpret.

5. Conclusion

The classification problem is an important application area of machine learning. This project applied and analyzed four algorithms (KNN, Logistic Regression, SVM, Neuron Networks) to classify the handwritten numbers in the MINSIT dataset. Each algorithm has its own advantages and disadvantages in terms of complexity, interpretability, accuracy, and generalization. In general, the SVM algorithm has achieved the best performance. It is worth noting that there are still some areas for improvement in this project. The dataset samples are not large enough, considering the time cost, so the accuracy can be further improved using larger training samples. Besides, other algorithms can be implemented and compared, such as the naïve Bayes classifier and K-means classifier.

6. Reference

[1] MNIST Database: http://yann.lecun.com/exdb/mnist/

[2] Zhang, Min-Ling, and Zhi-Hua Zhou. "ML-KNN: A lazy learning approach to multi-label learning." Pattern recognition 40.7 (2007): 2038-2048.

[3] Wright, Raymond E. "Logistic regression." (1995).

[4] Joachims, Thorsten. Making large-scale SVM learning practical. No. 1998, 28. Technical Report, 1998.

7. Github link

https://github.com/HaotianShi/CS532