CS 532 Project Update 2

Name: Haotian Shi

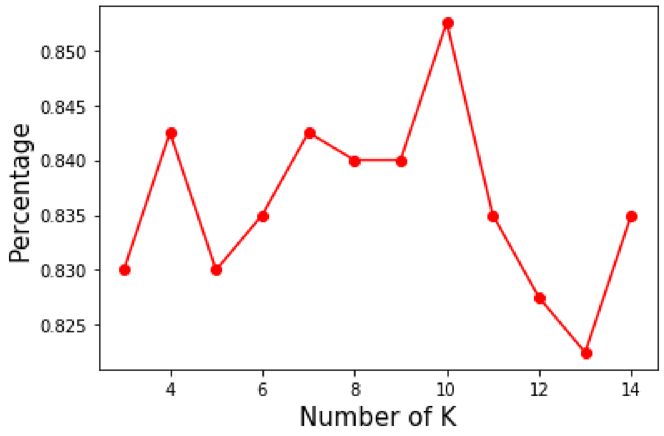
1. Current Progress

For the original dataset (MINIST), 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 test data' X' is 10000 x 784. Due to the huge amount of data, which leads to 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.

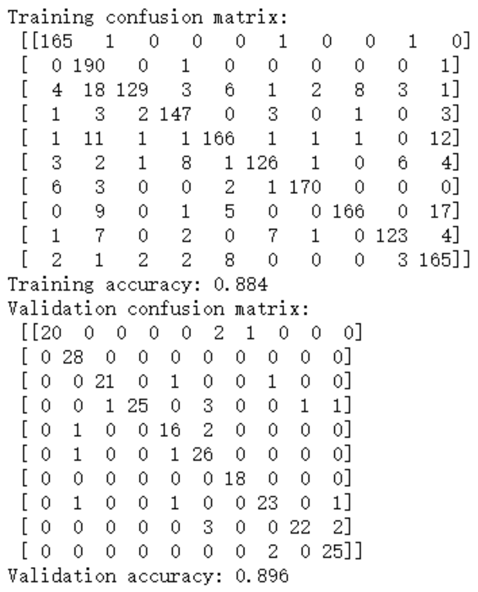
Currently, I have implemented four algorithms for classifying the dataset: KNN algorithm, Logistic algorithm, SVM algorithm, and neuron networks for my experiment. For the KNN algorithm, three works have been done.

KNN algorithm

Firstly, I did a basic case experiment, which directly uses the KNN algorithm with ten nearest neighbors for classification and use the model to evaluate the testing dataset. The testing accuracy is 85.25%. Then, I experimented with the number of the nearest neighbor K. I chose the K ranging from 3 to 14. The results show that when K = 10, the testing accuracy reaches a maximum of 85.25%, which shows below.



Thirdly, I did cross-validation in the KNN algorithm based on 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. I also printed the training confusion matrix and validation confusion matrix for the best KNN model, which is shown below:



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

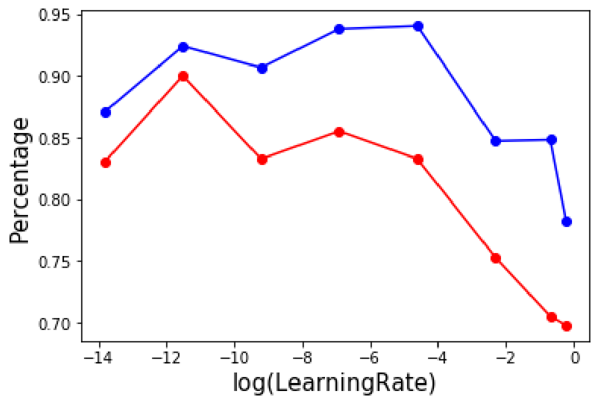
The accuracy of the best knn model after cross-validation is 0.8575. Although the highest validation accuracy has been improved to 0.896. The accuracy for the test dataset still remains 0.8575.

logistic algorithm

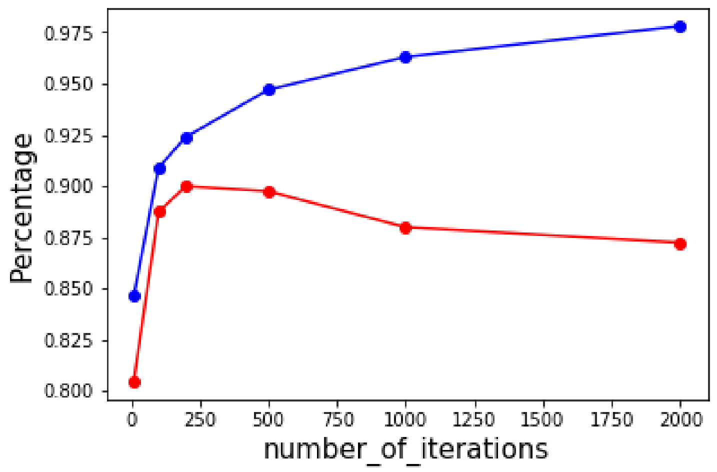
For the logistic algorithm, three works have been done. The task is decomposed into ten binary classification tasks with ten classifiers. Each classifier predicts a confidence level for the target number. For each sample, the category with the highest confidence is the category that the sample belongs to.

Firstly, I did a basic case experiment, which directly uses the logistic algorithm with a 0.5 learning rate and 200 iterations for gradient descent. During the training process, each classifier shows the training accuracy and testing accuracy, demonstrating the performance of the logistic regression algorithm on classifying different numbers. The 1th classifier shows highest test accuracy, which reaches 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%.

Then, I did an experiment that tunes the learning rate. The learning rate ranges from 0.00001 to 0.8. The results show that the best learning rate is 0.00001, which leads to the highest testing accuracy 90%. The figure below shows the results.



Thirdly, I experimented with the number of iterations for gradient descent based on the optimal learning rate. The number of iterations ranges from 10 to 2000. 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 at first and decreases. This is because too many iterations can cause an overfitting problem, which reduces the model's generalization capability. The figure below demonstrates the result.

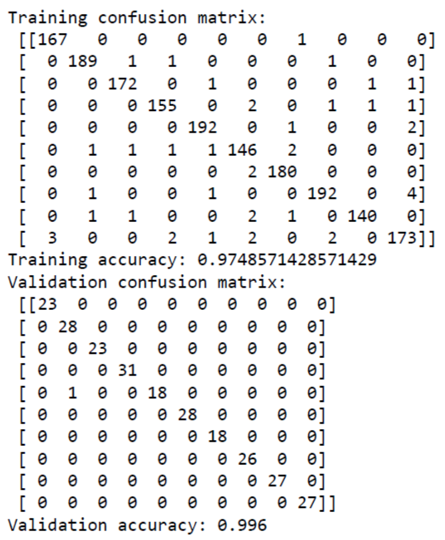


SVM algorithm

For the SVM algorithm, three works have been done. Firstly, I did a basic case experiment, which 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, I did experiments which tune the parameters: kernel and gamma. 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.

Based on the optimal parameters, I did the cross-validation for the SVM classifier. The confusion matrix for the best model is shown below:



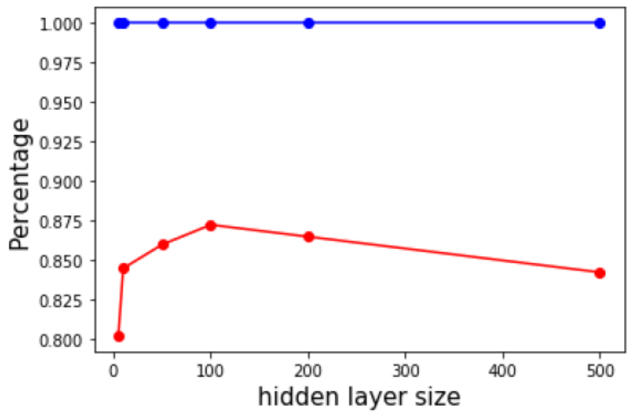
```
Training confusion matrix:
 [[167   0   0   0   0   0   1   0   0   0]
 [  0 189   1   1   0   0   0   1   0   0]
 [  0   0 172   0   1   0   0   0   1   1]
 [  0   0   0 155   0   2   0   1   1   1]
 [  0   0   0   0 192   0   1   0   0   2]
 [  0   1   1   1   1 146   2   0   0   0]
 [  0   0   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
```

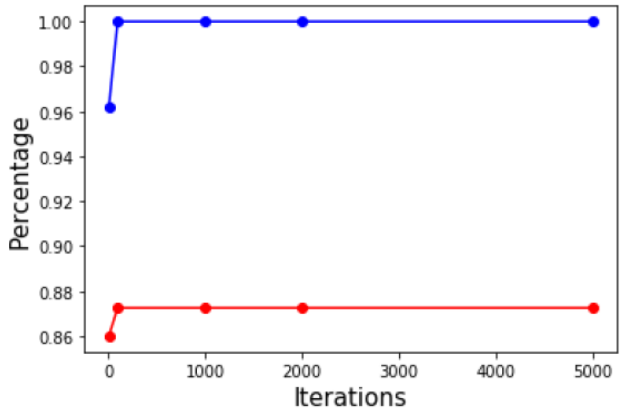
Although the highest validation accuracy has improved to 0.996, the test dataset's accuracy remains 0.9.

neuron networks

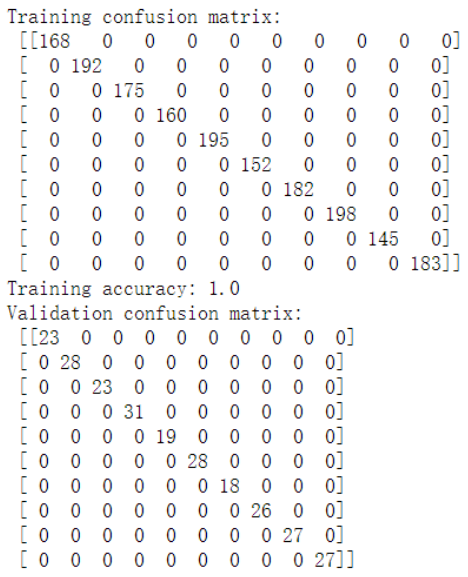
For the neuron networks, three works have been done. Firstly, I did a basic case experiment, which uses three hidden layers with ten neurons for each layer. The solver is 'lbfgs'. The neuron networks' accuracy for the training dataset is 99.95%, while for the testing dataset is only 0.83.

Then, I did a series of experiments that tune the parameters: solver (sgd, adam, lbfgs), activation function (logistic, tanh, identity), the learning rate, hidden size, and maximum iterations. The results show that the best solver is the 'lbfgs', which leads to the highest testing accuracy 82.75%. The gamma will not impact the performance of the "rbf" SVM classifier. For the activation function, the identity activation function achieves the best performance based on 'lbfgs' solver. The neuron networks' accuracy using the "identity" activation function for the testing dataset is 84.5%. The learning rate has little impact on training. 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 some threshold and will not impact the performance when it ensures the convergence. The results are shown in the below figures.





Similar to other algorithms, I did cross-validation for neuron networks. Although the highest validation accuracy is 100%, the test 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]]
```

2. Plan for the next step

Next, I will finish the final report. I am generally on track based on the timeline.

The github link below shows the details for the project.

Github link: https://github.com/HaotianShi/CS532

Reference: Project Proposal

Project Datasets

The data processed in this project is MNIST dataset, which is a widely used dataset for classification algorithm from National Institute of Standards and Technology (NIST). The source link is http://yann.lecun.com/exdb/mnist/. The training set consists of handwritten numbers from 250 different people, of which 50% are high school students and 50% are from the Census Bureau staff. The test set is also handwritten digital data in the same proportion. The training data set contains 60,000 samples, and the test data set contains 10,000 samples. Each picture in the MNIST data set consists of 28 x 28 pixels, and each pixel is represented by a gray value. The 28 x 28 pixels are expanded into a one-dimensional row vector with 784 values. These rows are the feature rows in the first array of an image. The second array (labels) contains the corresponding target variable, which is the class label of the handwritten number (integer 0-9).

Thus, the dimension of the training data' X' is 60000 x 784, where each sample corresponds to a label from numbers 0-9. Similarly, the dimension for the test data' X' is 10000 x 784. The classification problem is defined as: which number from 0 to 9 is most likely to be given a feature vector with 784 features.

Investigated Algorithms

The following three algorithms will be applied on the dataset, and corresponding parameters will be analyzed.

1. k-nearest neighbor
   parameters for experiment: number of neighbors K; weights (all points in each neighborhood are weighted equally or weight points by the inverse of their distance); algorithm (which algorithm is used to search for the nearest k points when building the KNN model), etc.

2. neural networks
   parameters for experiment: number of hidden layers, learning rate, momentum, activation function; optimization algorithm, regularization, etc.

3. SVM
   parameters for experiment: kernel, degree, gamma, decision function shape, etc.

Validation task

The dataset will be divided into multiple subsets for cross-validation, and the validation results are averaged over the rounds to better estimate the model's predictive performance.

Confusion matrix, precision and recall will be used to evaluate the model performance. The performance of each algorithm will be compared and the basic principle behind the algorithm will be analyzed to discuss the evaluation results.

Schedule

Week	**Tasks**
10/19-10/25	Understanding the basic principles of the three algorithms
10/26-11/01	Dataset processing, developing training environment
11/02-11/08	Learning algorithm investigation for k-nearest neighbor and neural networks
11/09-11/15	Learning algorithm design for SVM
11/16-11/22	Evaluation and validation, results discussion
11/23-11/29	Write final report
11/30-12/08	Revise final report

Github link: https://github.com/HaotianShi/CS532