Yiding Fan, Yue Ma

CMPUT 466 Mini Project

1 December 2017

Machine Learning: Semeion Handwritten Digit

Instruction:

We demonstrate Machine Learning on an essential question: The Distinguishment over the Handwritten Digits. Three different algorithms (Naive Bayes, Logistic Classification and Neural Network) are used as the methodologies. The results of different algorithms are compared. As the sources for Naive Bayes and Neural Network, the scikit learning package provides a convenient API. Logistic Regression is implemented based on the previous assignment.

Description of the dataset:

1593 handwritten digits from around 80 persons were scanned, stretched in a rectangular pixel box 16x16 in a gray scale of 256 values. Then each pixel of each image was scaled into a boolean (1/0) value using a fixed threshold. Each person wrote on a paper all the digits from 0 to 9, twice. The commitment was to write the digit the first time in the normal way and the second time in a fast way. For a single record, there are 256 input features (0/1) and 10 output values. For the output value, it uses 10 boolean values to represent 10 digits (0-9). If one boolean value shows true, the rest will show false (one vs all). For example:Y = 0000000001 is one digit and Y = 0000000010 is another digit. The 256 features are independent. And the data set is a balance data set which contains all the digits equally.

The importance of the problem:

In this project, the recognition on handwritten digits(0-9) is the problem to be solved. This problem is a sub-question of the Image Identification. When we want to convert the analog world

into a digital world in which the information can be easily gained and transferred, the analog input such as human hand-writing will be a good aspect. And the handwritten digits (0-9) seems to be a suitable start.

Description of the three algorithms:

Stochastic Gradient Descent:

Though the SGD is not one of the three algorithms in this project, the SGD method is used in both logistic regression classification and the neural network classification.

SGD is an incremental gradient descent method, is a stochastic approximation of the gradient descent optimization and iterative method for minimizing an objective function that is written as a sum of differential functions.

For each time calculation: w = w - stepsize*g, where g is the gradient of the cost function.

Through the SGD approaches, the proper weights can be found.

Logistic Regression Classification:

Logistic regression is a generalized linear model, where the distribution over Y given x is a Bernoulli distribution, and the transfer function is the sigmoid function. The input of one record is 256 features which are independent. The output is 10 boolean values which are discrete value, so it is suitable to use the logistic regression. For our problem, the output should be ten one-vs-all probability problems. When all probabilities over 10 outputs according to one input are calculated, the output which has the highest probability to be 1 will be set to 1 and the other nine outputs will be set to 0. The cost function used is the negative log-likelihood function. The details are shown as below:

For learning on the training set:

SGD method is used to renew the weights: $w_{t+1} = w_t - \text{stepsize}*(\text{sigmoid}(x_i^T w_t)-y_i)x_{ij}$

For prediction on a new data record (X):

Y=[0,0,0,0,0,0,0,0,0,0] # initialization

$p(y_k=1|X)= \text{argmax}([p(y_0=1|X), p(y_1=1|X),....., p(y_9=1|X)]$

Y[k]=1

(For each $p(y_n=1|X)$: $p(y_n=1|X) = \text{sigmoid}(w_n^T x)$)

Therefore, our problem will be 10 MLP learning w_k using the SGD method.

Naive Bayes Classification:

Naive Bayes classification is a generative approach to prediction based on applying Bayes' theorem with strong independence assumptions between the features. Since the 256 features of the dataset are independent, the Naive Bayes algorithm can be tested. Since the problem is a one vs all probability problem, the decision rule for labeling:

$$\text{argmax } p(y=c|X)$$

Where $p(y = c|x) \propto p(X|y = c)p(y = c)$.

The $p(X|y=c)$ will have different distribution according to assumptions (Bernoulli, Multinomial, Gaussian).

Neural Network:

The Neural Network algorithm comes from the bionics of the Biological Neural Networks. Since humans using Neural for thinking and recognizing the digits, the Neural Network will be a reasonable solution to this problem. Using the neural network, different features can be linked together, which may lead to a higher accuracy. The MLPClassifier in sklearn.neural_network package can be used as the function. The structure of this Neural Network looks like:

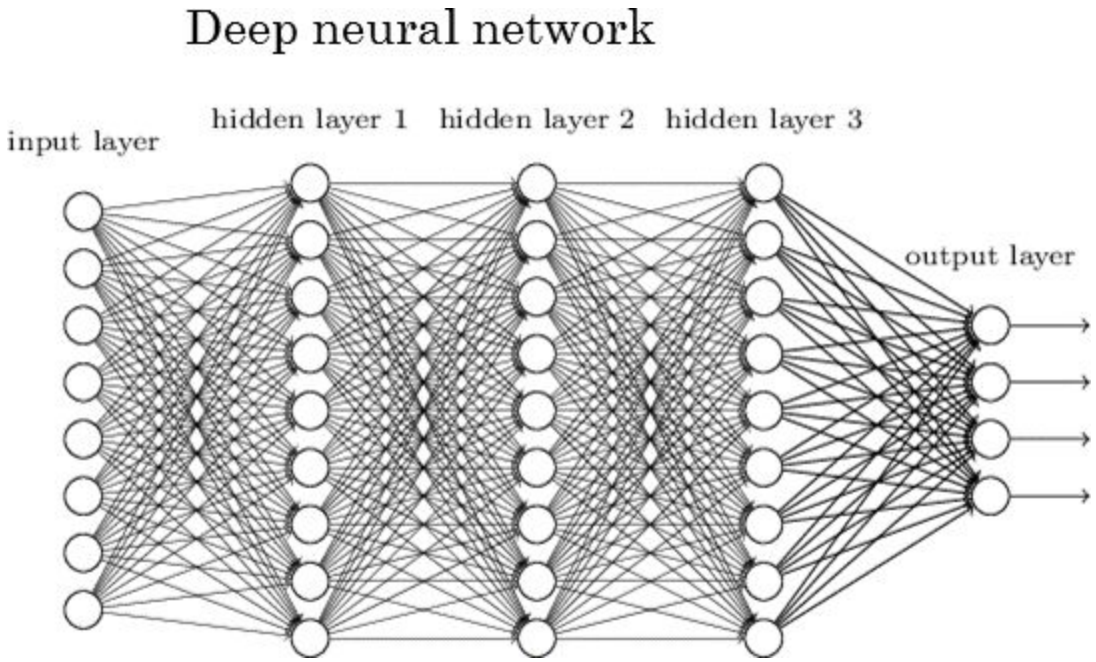

Figure 1. Deep neural network

The API of this function MLPClassifier (solver='sgd', alpha=0, max_iter=1500, activation='relu', hidden_layer_sizes=(num_nodes, num_layers),tol=1e-6). The solver is the gradient descent solver, in this problem we choose the SGD as an alignment with the logistics regression. Alpha is for the l2 regularizer, we don't use it for now. The activation function we use is Relu: f(x)=max(0,x). The hidden_layer_sizes is for the shape of the hidden layers. The tol and max_iter is for the train stops, any of two reached will stop the training. At this point we transform the Y into (0~9) digits. For example:Y = 0000000001 is '9' and Y = 0000000010 is '8'. It is a multi-classification problem. The loss function used by the API is the cross-entropy.

Design of experiments:

There are 1593 records in this dataset. In each time experiment, we randomly choose 1000 records as the training set and 500 records as the test set. For each algorithm, we run 10 times and get the average error so that we can compare each algorithm through the average error and the standard error. Since the results are a set of digits and the set is balanced, the hamming distance will be the difference of two sets which means how many different digits appear at the same position in the set. The hamming distance is the error function for our method.

$$Err(Xtest, Y\,test, method) = hamming(Y\,test, Y\,predict)/len(Y\,test)$$

The error is reported in percentage.

Logistic Regression Classification:

There are two parameters used in this algorithm: 'epoch' and 'stepsize'. The cross-validation method is chosen to find the best epoch and the best stepsize.

First, the epoch is set up to 1000 and the stepsize changes between 0.01, 0.02, ...,0.1, the cross-validation is applied to find the best stepsize. After we randomly pick 1000 records as the training dataset, the training data is evenly splitted into four groups. For each parameter, we do four times cross-validation learning over these four groups. For each time, the three groups of training data is used to learn the model, and the left one group is used to test the model. Then an average error of these four times cross-validation learning is calculated. After all 10 different

stepsizes are tested using the cross-validation method, we chose the best stepsize (0.03) with the least average error. Also, if the stepsize is 0.01 and 0.02, the average error is also very small, but it takes more time to get converge than 0.03. If the stepsize is too large, it will miss the global minimum point.

Second, the stepsize is fixed at 0.03 and the epoch changes between 300, 600, 900,...,1500, 1800. We use the same cross-validation method to find the best epoch which is 1500. The epoch being 1500 is reasonable, since if the epoch is too small, the error is not converged, and if the epoch is too large, the model will have an overfitting problem (see figure 2).

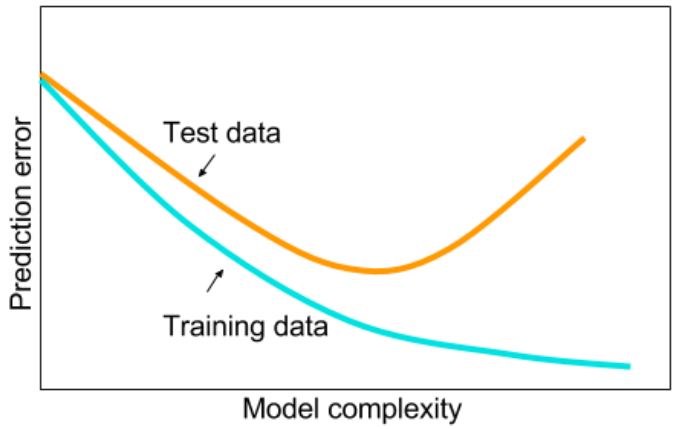

Figure 2: Model complexity VS error

Finally, we run 10 times using the best parameters (epoch = 1500 and stepsize = 0.03) to get the result. The average error is 9.58+-0.2505 , which means about 47 out of 500 testing data has a wrong prediction.

Naive Bayes Classification:

We use the naive bayes function from 'scikit-learn' to learn the model. The p(X|y=c) can be one of the three different distributions: Bernoulli, Multinomial, Gaussian. We use the cross-validation method to find the best distribution assumption. The cross-validation approach is the same as the one in the Logistic regression. The 1000 training data is splitted into four groups. For each distribution assumption, we do four times cross-validation training and get the average error. Finally, we compare the average error and find the best distribution is 'Bernoulli'

distribution. It means this dataset should fit a Bernoulli distribution. We run 10 times using the 'Bernoulli' parameter to get a final result: average error = 14.52+-0.3624.

Neural Network Classification

The neural network function in scikit-learn is used. First, we consider about cross-validation on this method, it turns out to be too difficult to achieve. Because we have two parameters for SGD (max_iter,tol), one for activation ('Relu', 'tanh', 'sigmoid'...) and two for hidden layers (number of nodes and number of layers). We run some tests on different activation, the 'Relu' works best and stable. Then we try to find the best layers and nodes. For the selection of the nodes, we use idea of binary search, we choose from (256-2000) at fixed layers (10) . Finally, when nodes equal to 500, the method perform the best. And for layers, we select from (2-20). The 10 layers structure works better than the others. With the experience of using the Logistic Regression, we choose the max_iter =1500 and tol=1e-6. The average error is 8.94+-0.3340.

Analysis of the winner algorithm:

The three algorithms all can work well. The Naive Bayes method runs the fastest, but the accuracy is low. There is a The Logistic Regression method can reach 90.42% accuracy and the standard error is 0.2505. The neural network has 91.06% accuracy, but the standard error is a little bit higher than the one in Logistic Regression.

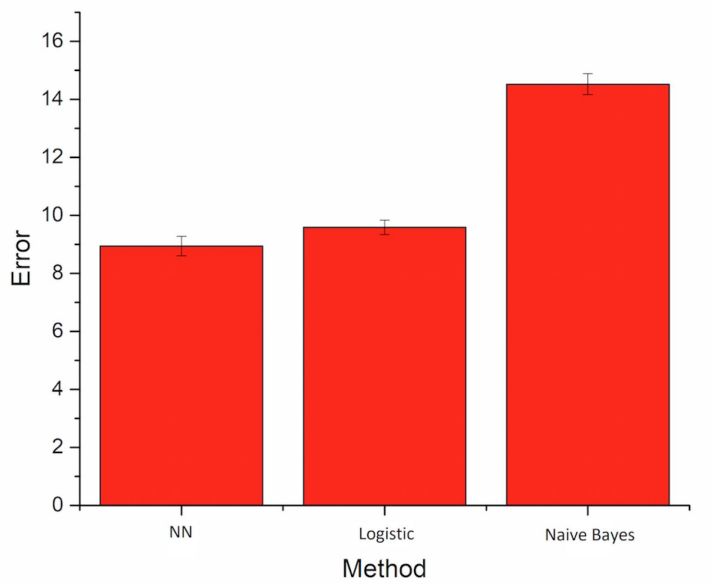

Figure 3. Results of the three methods

There is a trade-off between the speed and the accuracy. Since this dataset is not large, the speed is not the first concern. According to the results from these three algorithms, the best method is neural network since it has the lowest average error with a reasonable standard error: 8.94+-0.3340. The reason of the Neural Network works better is that this dataset is transformed from a 16*16 pixel images and extend into a 1d array. The Neural Network we used is a fully-connected Neural Network, it can contain the information of one pixel and the pixels around that one. And the dataset may contain non-linear relations, the Neural Network can handle those relations. With the more data input, the Neural Network can learn more about the data. But the logistic regression will converge at some point. Overall, the Neural Network is the best algorithm for this problem.

Conclusion

As the solution to recognition of the handwritten digits, the Neural Network can achieve 91% accuracy. The Naive Bayes is not a good method for this complex task. The Logistic Regression can achieve 90% accuracy. Both Logistic Regression and Neural Network are acceptable; However, as the input data increases, the Neural Network can be even better. Since our ‘big’

question is Image Identification, so a convolutional neural network can also improve the performance in the further study.

Reference:

CMPUT 466 notes

CMPUT 466 Assignment1

CMPUT 466 Assignment2

https://archive.ics.uci.edu/ml/machine-learning-databases/semeion/semeion.names

https://en.wikipedia.org/wiki/Stochastic_gradient_descent

http://gluon.mxnet.io/chapter02_supervised-learning/regularization-scratch.html

NN:http://www.rsipvision.com/wp-content/uploads/2015/04/Slide5.png

http://scikit-learn.org/stable/modules/neural_networks_supervised.html