Hand-written Digits Classification and Letter Recognition

Group 10
Sixiang Ma, Yuan Xiao, Xiaokuan Zhang

12/09/2016

Abstract

say something

1 Introduction

1.1 Problem Description

Our group wants to solve the problem of classifying hand-written digits and black-and-white rectangular pixel-displayed letters. We have two datasets from UCI machine learning repository [11]. One is hand-written digits dataset [1]. Another is letter recognition dataset [2]. We will introduce them in more detail in Section 4.

1.2 Classifying Algorithms

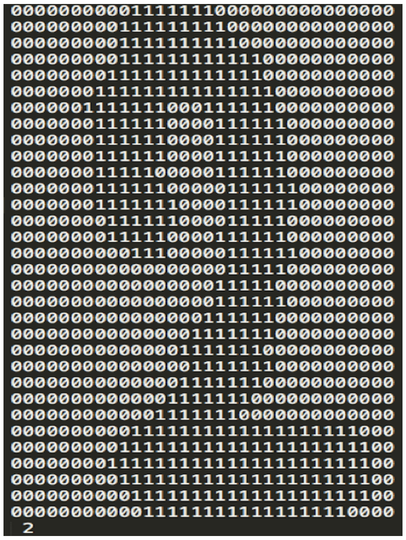
In this report, we test three machine learning algorithms, i.e., Support Vector Machine (SVM), Neural Networks, and Naive Bayes. We will present the detail of algorithms in Section 3.

1.3 Result Summary

For the first dataset, after tuning the parameters, the three classifiers can achieve , , , accuracy.

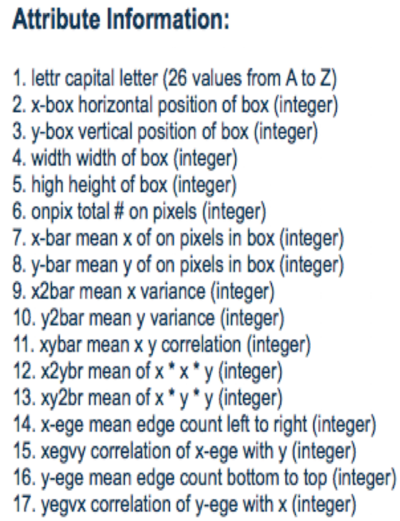
2 Background

The hand-written digit dataset was first used in [10]. In the paper they combined different classifiers to obtain a better performance. The letter recognition dataset was first used in [7]. They generated classification rules to distinguish different letters.



(a) A Digit Example: 2

(b) Letter Examples



(c) Features of Letters

Figure 1: Examples and Features

3 Methodology

3.1 Support Vector Machine

Support vector machines (SVMs) [5] are supervised learning models that analyze data for classification or regression. Provided with a set of training examples, which are marked with their belonging categories, a SVM algorithm performs to build a model so as to recognize and assign testing examples to the predicted categories.

Besides linear classification, SVMs are also able to efficiently perform non-linear classification when kernel tricks are applied[4] , which will map original input into high-dimensional attribute space.

3.2 Neural Networks

Neural networks, or artificial neural networks [8], simulate the functions of nerve cells of human brain and serve as an important computational approach in machine learning. They typically form a structure of multiple layers of basic perceptrons and support both supervised and unsupervised learning.

Neural networks have a long history, dating back to the 1940s [12]. However, the idea of artificial neural networks was not popular at early days due to its limitation in solving logical calculations [13]. Modern neural networks revived in the past decade, along with the rise of deep learning [3, 14].

3.3 Naive Bayes

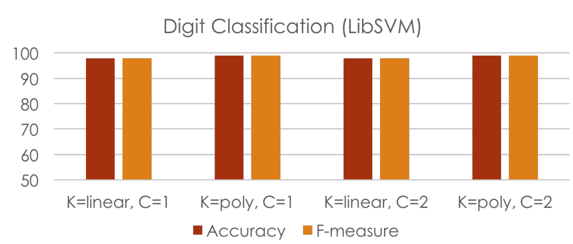
Naive Bayes classifier makes use of the Bayes Theorem. It is basically a conditional probability model. It is one of the simplest machine learning algorithms. Compared to Bayesian Networks, Naive Bayes is technically a special case by assuming that all features are conditionally independent from each other given the class label. One of the earliest papers that described this algorithm was from 1970s [6].

4 Experiment

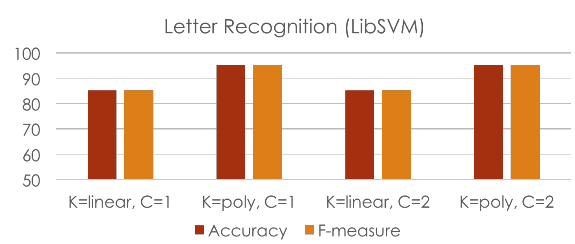
In this project, we mainly used `Weka` [9] to test different machine learning algorithms. For each algorithm, we use different parameter settings, which will be covered in this section.

4.1 Datasets

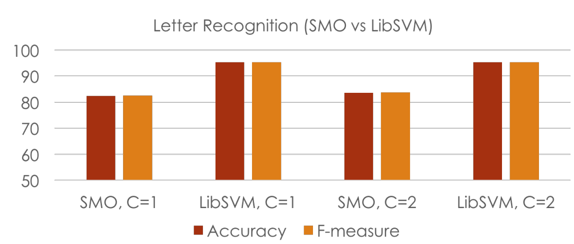
We have two datasets, the hand-written digits dataset [1] and the letter recognition dataset [2]. The hand-written digits dataset has 5620 instances. For each instance, there are 1024 attributes (32x32 matrix), whose values are either 0 or 1. An example is shown in Figure 1a, which is clearly a '2'. After grouping every 4x4 blocks, the dimension is reduced to 64 (8x8 matrix), and each attribute ranges from 0 to 16. Another is letter recognition dataset [2]. The character images were based on 20 different fonts. It has 20000 instances. For each instance, there are 16 attributes, whose values range from 0 to 15. The attributes are sophisticated, as shown in Figure 1c. Some examples of the fonts are shown in Figure 1b. More details of the datasets and features can be found by visiting the links.



(a) Digit Classification



(b) Letter Recognition



(c) Comparison between SMO and LibSVM

Figure 2: Performance of Support Vector Machine

4.2 Hand-written Digits Classification

4.2.1 Support Vector Machine

As shown in Figure 2a, SVM classifiers provide a good performance on the Digit dataset. In the experiment, two key parameters are tuned. One is the kernel type and the other is the complexity parameter. For kernels, the linear kernel and the non-linear polynomial kernel are tested. When the polynomial kernel is applied, 98.97% accuracy can be achieved, and 97.92% accuracy is achieved in cases of the linear kernel.

From the figure, two results can be observed. Firstly, polynomial kernel has better performance than the linear one, regardless of values of the complexity parameter. Secondly, the complexity parameter has little effect on the SVM model in regard with the Digit dataset.

4.2.2 Neural Networks

In order to understand how the many parameters influence the performance of multilayer perceptron in classification, we decide to change only one parameter at a time. The default setting is "-L 0.3 -M 0.2 -N 500 -V 0 -S 0 -E 20 -H a". And the parameters that we are interested in are learning rate, training time and validation threshold.

Figure 3a shows the average performance for different parameter combinations. For default settings, we have the average accuracy of 92.73%. And the root relative squared error is as low as 38.53%. Changing validation threshold to either more or less does not affect the performance at all. And if we let training time be less, the overall accuracy lowers a little, which matches our expectation.

It can be easily noticed that changing learning rate to 0.6 affects the performance heavily. Thus we tested with the same parameter combination again. In Figure 3b, when we ran the test again, the accuracy grew to 98.36%. Such inconsistency in the test result showed that the performance of multipayer perceptron is influenced by stochastic. If we increase the stochastic by modifying the parameters, the fluctuation grows.

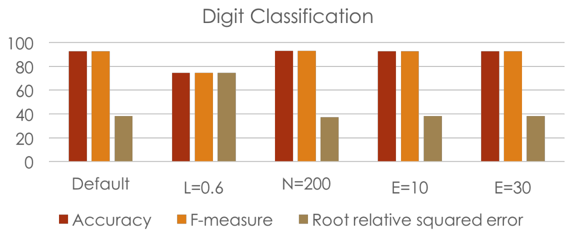
4.2.3 Naive Bayes

The Naive Bayes works well on the Hand-written Digit dataset. In Figure 4, K means whether to use kernel estimator, and D means whether to use supervised discretization. In Figure 4a, when K=F, D=F, which is the worst case scenario, the accuracy and F-measure are still higher than 90%. In the best case (K=T, D=F), they both exceeds 92%.

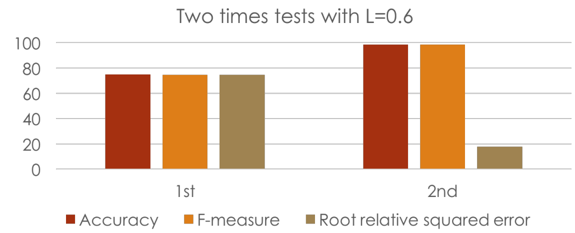
4.3 Letter Recognition

4.3.1 Support Vector Machine

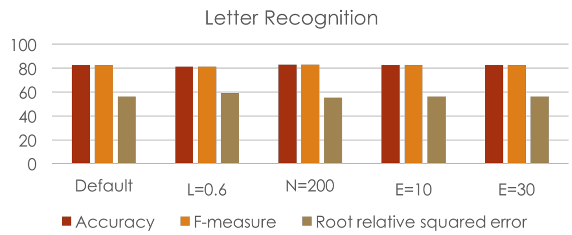
Compared with the Digit dataset, the polynomial kernel still has a good performance on the Letter dataset as more than 95% instances are correctly recognized. However, the accuracy from the linear kernel is much lower no matter which values of the complexity parameter are configured. From Figure 2b, we can see that



(a) Digit Classification



(b) L=0.6 in Digit Classification



(c) Letter Recognition

Figure 3: Performance of Multilayer Perceptron

Parameters	Class (Letter)	Accuracy
K = linear, C = 1	S	68%
	H	69.8%
	R	73.8%
K = poly, C = 1	H	91%
	R	91%
	S	64%
K = linear, C = 2	S	67.5%
	H	69.7%
	B	92.4%
K = poly, C = 2	R	90.8%
	H	91.1%
	F	92.5%

Table 1: Worst Cases: Support Vector Machine

only 85% accuracy and F-measure are achieved when the linear kernel is used, whereas those are more than 97% for the Digit dataset.

To explore an insight of which letters have large contribution to inaccurate recognition, a detailed inaccuracy analysis by class is given. Although the accuracy from the polynomial kernel is much higher than the linear one, as shown in Table 1, both of them have least accuracy on recognizing letters 'H', 'S', and 'R'.

4.3.2 Neural Networks

The overall performance of multilayer perceptron is quite stable. The accuracy is around 82% and the root relative squared error is around 56%. However, there are some classes with relatively low accuracy making the average performance not so good as digit recognition. Table 2 shows the least correct classes: ‘G’, ‘H’ and ‘S’.

Neural networks do a better job than the other two algorithms in recognizing ‘O’ and ‘Q’ and it can correctly tell ‘X’ apart from ‘S’. Changing learning rate higher helps the performance with ‘H’.

Parameters	Class (Letters)	Accuracy
Default	G	69%
	H	64.4%
	S	65%
L = 0.6	G	68.2%
	H	69.1%
	S	64%
N = 200	G	69.3%
	H	65.1%
	S	64.4%

Table 2: Worst Cases: Multilayer Perceptron

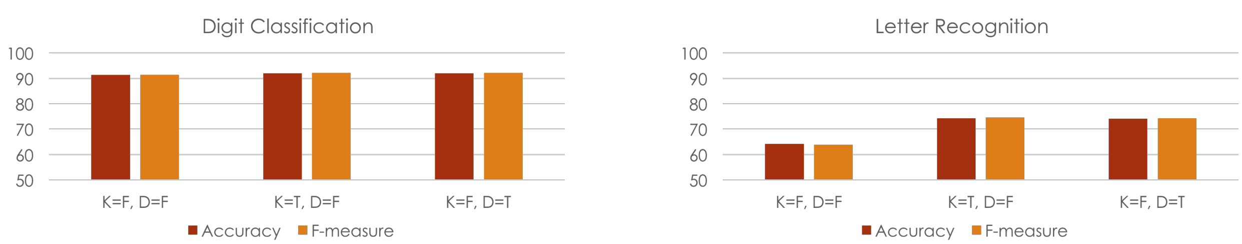
Parameters	Class (Letters)	Accuracy
K = F, D = F	S	29.4%
	H	30.5%
	Y	33.1%
K = T, D = F	H	57.4%
	S	64.2%
	X	64.3%
K = F, D = T	H	57.5%
	E	60.7%
	X	64.4%

Table 3: Worst Cases: Naive Bayes

4.3.3 Naive Bayes

Naive Bayes does not work well on the Letter Recognition dataset. From Figure 4b, we can see that the highest accuracy is lower than 75%. Generally speaking, Naive Bayes is not suitable for classifying letters.

In this dataset, we are also interested in which letters Naive Bayes performs worst. Table 3 shows the top 3 worst cases when using Naive Bayes with different parameters. In all three parameter settings, ‘H’ is always one of the top 3, which means that ‘H’ is quite hard to classify for Naive Bayes. Also, the same applies to ‘S’ and ‘X’, as they appear in two of the three cases.



(a) Digit Classification

(b) Letter Recognition

Figure 4: Performance of Naive Bayes

5 Discussion

5.1 Evaluation Matrix

In our presentation, we used `root relative squared error` to evaluate the performance of our algorithms and falsely claimed that SVM was not suitable for our datasets. However, it makes little sense to evaluate this feature on non-binary datasets. So in the report, we changed it to F-measure.

5.2 Different Libraries of SVM

In our presentation, we only discussed the performance of the polynomial kernel in SMO. To have a comprehensive understanding on the performance of SVMs, in the later work, experiments based LibSVM are performed.

Figure 2c compares the accuracy and F-measure between SMO and LibSVM when the polynomial kernel is applied. Surprisedly, although both the complexity and kernel parameters are configured as the same, the accuracy from LibSVM is higher than that from SVM by 12%. This difference may come from distinct algorithms and implementations behind them.

5.3 Parameters of Multilayer Perceptron

Multilayer Perceptron has a number of parameters. In the project, we would like to look into the following specific parameters: learning rate, training time and validation threshold. Other parameters include momentum, seed, nominalToBinaryFilter, hiddenLayers etc.

Learning rate (-L) stands for the amount the weights are updated. Training time (-N) is the number of epochs to train through. If the validation set is non-zero then it can terminate the network early. Validation threshold (-E) is used to terminate validation testing. The value here dictates how many times in a row the validation set error can get worse before training is terminated.

In our tests, the validation threshold does not affect the performance.

6 Conclusion

In general, the correctness of handwritten letter/digit recognition depends greatly on the preprocess of data. Our experiments show that using the original pixels as the characters has a better performance than extracting the features from the data. Also, all the algorithms we test meet with difficulties of telling similar letters/digits apart.

In particular, Nave Bayes performs badly in letter recognition when K=f, D=f. With other parameter settings, the correctness is around 75% with small deviation among different letters. In general, it has a better performance in doing digit classification than in letter recognition. SVM is generally suitable for both letter recognition and digit classification. However, due to the preprocess difference, the letter recognition performance is a little worse than digit classification. The classes that cannot be correctly classified are: H, S and R. Artificial Neural Network is heavily influenced by the parameters that control the stochastics. But generally speaking, it has a relatively stable performance. And it has the highest accuracy for most of the cases in our experiments compared to the other two algorithms. It is also weak in classifying less letters, which are G, H and S.

References

[1] Optical recognition of handwritten digits data set. `http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits`.

[2] Optical recognition of handwritten digits data set. `https://archive.ics.uci.edu/ml/datasets/Letter+Recognition`.

[3] Yoshua Bengio. Learning deep architectures for ai. *Foundations and trends® in Machine Learning*, 2(1):1–127, 2009.

[4] Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152. ACM, 1992.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[6] Richard O Duda, Peter E Hart, et al. *Pattern classification and scene analysis*, volume 3. Wiley New York, 1973.

[7] Peter W Frey and David J Slate. Letter recognition using holland-style adaptive classifiers. *Machine Learning*, 6:161, 1991.

[8] Martin T Hagan, Howard B Demuth, Mark H Beale, and Orlando De Jesús. *Neural network design*, volume 20. PWS publishing company Boston, 1996.

[9] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18, 2009.

[10] C Kaynak. Methods of combining multiple classifiers and their applications to handwritten digit recognition. Master's thesis, Institute of Graduate Studies in Science and Engineering, Bogazici University, 1995.

[11] M. Lichman. UCI machine learning repository, 2013.

[12] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[13] Marvin Minsky and Seymour Papert. Perceptrons: An introduction to computational geometry (expanded edn), 1988.

[14] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015.