Analysis of Naive Bayes, SVM, and MLP on Fashion-MNIST and Adult Income data sets

Qiang Ye, Jinfang Luo, Ying Xiao, Yan Ai, Lifeng Wan
20139927, 20111308, 20111402, 20027063, 20108546

Abstract

Different machine learning algorithms perform differently on different data sets. We experimented the performance of Naive Bayes, support vector machine, and multi-layer perceptron with different hyper-parameter combinations on Fashion-MNIST and Adult Income data sets to explore such differences. By using sklearn module, we found that: these three algorithms with hyper-parameters well tuned perform very closely on Adult Income test data set with the average accuracies of 79.63%, 74.12%, and 79.48% respectively; while on Fashion-Mnist test data set, support vector machine performs best with average accuracy of 89.16%, and the number of naive Bayes and multi-layer perceptron are 71.62% and 85.75% accordingly. We concluded that there's no one algorithm which can always do best on every data set, and it is essential to try different algorithms with different hyper-parameter's combination on a data set to find a best model on a certain problem.

1 Introduction

Naive Bayes(NB)[1], support vector machine(SVM)[2], and multi-layer perceptron(MLP)[3] are three basic machine learning algorithms with each has certain properties. Naive Bayes, for example, assumes that features are independent with each other which is often not true in real world, performs very well in solving some problems; SVM with Gaussian kernel was considered the best algorithms for almost all kind of problems before the thriving of deep learning; MLP could fit all kinds of non-linear mapping when given sufficient hidden layers and hidden neurons. For answering the question of which algorithm should be used for solving a specific problem, many different data sets have been built. In this paper, we use two popular data sets: Fashion-MNIST[4] and Adult Income[5] to explore the properties of the three algorithms and compare their differences of the performance on the two data sets.

2 Methods

2.1 Data pre-processing

We use sklearn modules, which is a very popular machine learning algorithm library with the interface of most algorithms unified, as our platform to explore and compare the three algorithms: , SVM, and MLP. The data sets we chose, Fashion-MNIST and Adult Income, are also very popular. Fashion-MNIST is a data set consisting of a training set of 60,000 examples and a test set of 10,000 examples; each example is a 28x28 gray-scale image, associated with a label from 10 classes; Adult Income data set is a data set with more than 30 thousand population examples; each uses fourteen characteristics to describe the properties of a person, and gives a label indicating whether the person has an income

Project of IFT6390 Machine Learning, Montréal, Canada.

over or less than 50K.The examples in Adult Income data set are unbalanced as there are 24720 examples in one class(income <=50k) yet only 7841 examples in another(income >50k).

Fashion-MNIST is a data set of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 gray scale image, associated with a label from 10 classes: 0 : 'T-shirt/top', 1 : 'Trouser', 2 : 'Pullover', 3 : 'Dress', 4 : 'Coat', 5 : 'Sandal', 6 : 'Shirt', 7 : 'Sneaker', 8 : 'Bag', 9 : 'Ankle boot'. The entire data set is well clean and organized. There is no data pre-processing for the data set.

Adult Income data set was extracted by Barry Becker from the 1994 Census database. A set of reasonably clean records was extracted using the following conditions. This is a binary classification task, we represent 'Target' by -1 and 1. -1 means income <=50K, 1 means income >50K. We found that both features 'Education' and 'Education-Num' represents the same thing. We would like to keep the 'Education-Num' and drop another one. There are six features with continuous data. We choose three of them to map category intervals. These are "Age", "fnlwgt" and "Hours per week". As for "Education-Num", "Capital Gain" and "Capital Loss", we keep them as original form. For the missing data, they are represented by '?' in some features with object type; we keep them as original form due to process encoded, they could be represented by category number. Method LabelEncoder() is used to encode the object type of each feature with value between 0 and n_classes-1. For example, feature 'sex' with two different value: male and female, with processing, they are represented by 0 and 1. Feature selection[6] is a process of selecting a subset of relevant features for use in model construction, and will be used in Adult Income data set.

The two training data sets are first splitted into training and validation subsets with the ratio 0.7. For each algorithm and each data set, models with different hyper-parameter combinations are trained to fit the training subsets, and then validated on validation subsets. As the examples in Adult Income data set are not well balanced, we choose accuracy and F score[7] to evaluate the performance of models on this data set, For Fashion-MNIST data set, we only use the average accuracy. Finally, we observe the performance of the best models on two test data sets.

2.2 Selection of hyper-parameters

Table 1: hyper-parameters options for the algorithms on two data sets

Algorithm	Hyper-parameter	Fashion-MNIST	Adult Income
naive Bayes	density estimator	(Gaussian, Multinomial, Bernoulli)	
SVM	C	(0.001, 0.01, 0.1, 1, 10, 100, 1000)	
	gamma	(1e-6)	
	kernel	('rbf', 'poly')	
MLP	hidden layer settings	((16), (32), (64), (128), (256), (512), (1024), (1024, 1024), (1024, 512), (1024, 256), (1024, 128), (1024, 64), (1024, 32), (1024, 16), (512, 512), (512, 256), (512, 128), (512, 64), (512, 32), (512, 16), (256, 256), (256, 128), (256, 64), (256, 32), (256, 16), (128, 128), (128, 64), (128, 32), (128, 16), (64, 64), (64, 32), (64, 16), (32, 32), (32, 16), (16, 16))	((4), (8), (16), (32), (32, 32), (32, 16), (32, 8), (32, 4), (16, 16), (16, 8), (16, 4), (8, 8), (8, 4), (4, 4), (32, 32, 32), (16, 16, 16), (8, 8, 8), (4, 4, 4), (32, 16, 8), (32, 16, 8, 4))

As there are several hyper-parameters for each algorithm, we need first to find the best hyper-parameter set for each algorithm on training data sets, then to compare the performance of these algorithms on the test data sets. It is impossible to experiment on all possible hyper-parameter's combinations; therefore, we focus on some hyper-parameters and for each hyper-parameter we set a list of the values in which we are interested. Table1 lists the values of the hyper-parameters we chose. For the learning rate and max iterate times, we carefully set them so as they don't have a significant affection on the parameters. For the hyper-parameters not mentioned, we use their default values by sklearn library.

3 Results

3.1 Feature selection

We implemented two kinds of methods for trying to avoid redundant or even noise features. SelectKBest API of sklearn is used to get best K features before training models, showing the least two important features are "Relationship" and "Hours-per-week". When K = 12, we obtained a relatively good result(Naive Bayes: 80.23, SVM: 76.38, MLP: 75.27) for all three models. We also got a score of each feature with respect to the importance on income predict dataset with xgboost library. The result shows that "Education" and "Race" are least important. We compared horizontally the final accuracies of three models between xgboost and SelectBest by choosing best 12 features. Except same accuracy result 76.38 of SVM, results of SelectBest (Naive Bayes: 80.23, MLP: 75.27) are slightly better than xgboost's (Naive Bayes: 76.30, MLP: 29.16).

3.2 Performance of different models on both data sets

For each algorithm, the performance of models with different hyper-parameter set on Fashion-MNIST and Adult Income data sets are shown in table2.

On Fashion-MNIST data set: naive Bayes with Bernoulli distribution as estimator performs best among all three distribution; support vector machine with C=1000, gamma = 1e-6, and kernel = 'rbf' performs best among all possible combinations of c, gamma, and kernel; multi-layer perceptron with 1024 features of one hidden layer is the best. On Adult Income data set, Gaussian naive Bayes, support vector machine with C=100, gamma = 1e-6, and kernel = 'rbf', and MLP with (16, 16, 16) architecture performs best among all other different architectures in their model family.

We selected the best models of each algorithm, observed their performance on test data sets. As Figure1 illustrates, SVM with Gaussian kernel achieved best average accuracy on Fashion-MNIST data set. For Adult Income data set, naive Bayes achieved best average accuracy on Adult Income data set, but with a relatively low average F score than MLP. MLP obtained close average accuracy. We finally considered MLP performs best on this data set. Table3 and 4 shows the best models from each algorithm and their performances on both test data sets.

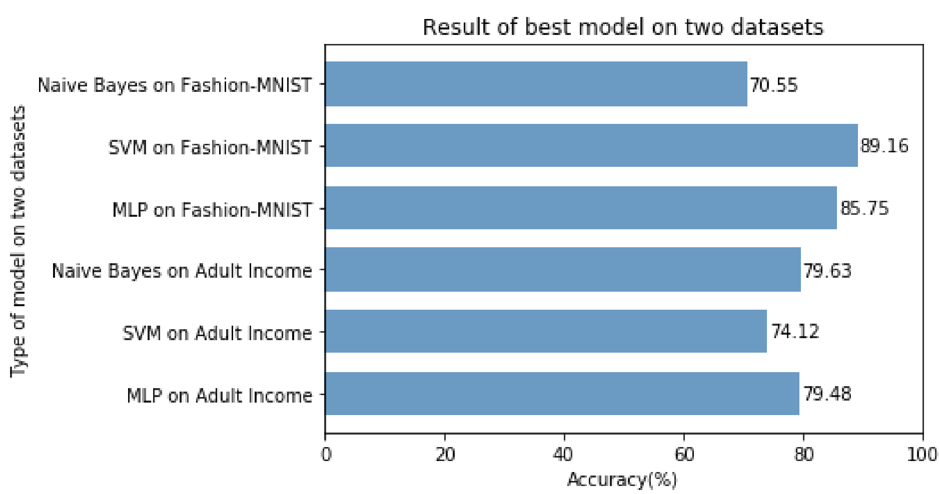

Figure 1: performance of best models from three algorithms on test data sets

Table 2: performance of different models on Fashion-MNIST and Adult Income data sets

algorithm	hyper-parameters	average accuracy (%)					average F score	
		Fashion-MNIST			Adult Income		Adult Income	
		train	val		train	val	train	val
naive Bayes	Gaussian	58.49	58.09		79.80	78.96 *	0.77	0.77
	Multinomial	66.51	67.22		78.29	78.20	0.74	0.74
	Bernoulli	71.62	71.89 *		72.56	72.84	0.74	0.75
support vector machine	C gamma kernel							
	0.001	85.85	83.98		76.15	75.39	0.66	0.65
	0.01	93.9	87.51		76.42	75.71	0.66	0.65
	0.1	98.95	88.26		76.02	75.69	0.66	0.65
	1 1e-6 poly	99.91	87.88		76.08	75.36	0.66	0.65
	10	100	87.47		76.15	75.39	0.66	0.65
	100	100	87.08		76.15	75.39	0.66	0.65
	1000	100	87.08		75.87	76.03	0.66	0.65
	0.001	10.20	9.79		76.02	75.69	0.66	0.65
	0.01	64.85	64.32		75.82	74.99	0.66	0.65
	0.1	84.41	81.93		76.15	75.39	0.66	0.65
	1 1e-6 rbf	97.81	89.53		76.15	75.39	0.65	0.66
	10	100	89.31		75.87	76.03	0.67	0.67
	100	100	89.31		76.69	76.43 *	0.67	0.65
	1000	100	89.57 *		72.56	71.48	0.66	0.65
multi-layer perceptron	hidden layer setting			hidden layer setting				
	(16)	53.28	52.65	(2)	23.84	24.60	0.66	0.65
	(32)	79.65	77.96	(4)	23.85	24.61	0.75	0.75
	(64)	81.70	79.66	(8)	79.03	79.03	0.73	0.73
	(128)	77.80	76.79	(16)	78.97	78.79	0.76	0.75
	(256)	88.80	85.11	(2, 2)	76.15	75.39	0.66	0.65
	(512)	89.49	85.93	(4, 2)	77.52	76.87	0.66	0.65
	(1024)	90.99	87.13 *	(8, 2)	76.62	75.94	0.67	0.66
	(1024, 16)	9.95	10.13	(16, 2)	76.15	75.39	0.66	0.65
	(1024, 32)	10.06	9.90	(4, 4)	78.12	77.54	0.74	0.74
	(1024, 64)	72.41	71.42	(8, 4)	79.02	78.86	0.74	0.74
	(1024, 128)	85.83	84.30	(16, 4)	79.23	79.06	0.74	0.73
	(1024, 256)	90.00	86.43	(8, 8)	78.09	77.57	0.70	0.69
	(1024, 512)	88.08	85.32	(16, 8)	79.46	79.20	0.74	0.74
	(1024, 1024)	89.35	85.26	(16, 16)	79.47	79.08	0.75	0.75
	(512, 16)	33.65	32.83	(2, 2, 2)	76.54	75.83	0.66	0.65
	(512, 32)	66.02	65.22	(4, 4, 4)	76.14	75.39	0.66	0.65
	(512, 64)	85.34	84.06	(8, 8, 8)	79.46	79.12	0.74	0.74
	(512, 128)	81.37	79.94	(16, 16, 16)	79.49	79.23 *	0.76	0.76
	(512, 256)	89.14	85.32					
	(512, 512)	88.75	85.47					
	(256, 16)	87.11	84.74					
	(256, 32)	87.10	84.63					
	(256, 64)	88.23	86.21					
	(256, 128)	88.18	84.64					
	(256, 256)	88.17	84.17					
	(128, 16)	87.99	85.85					
	(128, 32)	81.16	79.61					
	(128, 64)	88.93	85.67					
	(128, 128)	85.07	82.52					
	(64, 16)	85.94	83.64					
	(64, 32)	87.69	85.62					
	(64, 64)	85.00	83.11					
	(32, 16)	85.77	83.92					
	(32, 32)	86.10	83.71					
	(16, 16)	36.13	35.72					

* represents the best model of that algorithm on a data set

Table 3: performance of best models of each algorithm on test Fashion-MNIST data set

algorithm	hyper-parameters			test data set average accuracy(%)
naive Bayes	Bernoulli density estimator			70.55
SVM	C 1000	gamma 1e-6	kernel rbf	89.16
MLP	hidden layer and feature numbers (1024)			85.75

Table 4: performance of best models of each algorithm on test Adult Income data set

algorithm	hyper-parameters			average accuracy(%)	average F Score
naive Bayes	Gaussian density estimator			79.63	0.77
SVM	C 100	gamma 1e-6	kernel rbf	74.12	0.67
MLP	hidden layer setting (16, 16, 16)			79.48	0.84

4 Discussion

The results indicate that there is no model consistently better than others. Naive Bayes has better performance on Adult Income data set, yet failed on Fashion-MNIST. Support vector machine and multi-layer perceptron play closely on both data sets. None of the models provided satisfied accuracy on Adult Income data set, which may hint the features of the data set do not have strong relationship with the labels(predicted income class). In real world, we would recommend that more features are to be considered to make similar prediciton.

For multi-layer perceptron, adding more hidden layers didn't always increase the accuracy of the model, nor did increasing the number of features in hidden layers. For example, on Fashion-MNIST data set, the model with one hidden layer with 1024 features defeated others, including some two-hidden-layer models with the first layer has the same number of features. The number hidden layers and the number of features in each hidden layer affect the capacity of a of a MLP, which is partially supported by our results. Of all the MLP models with two hidden layers, the capacity decreases as the number of the features in the last hidden layer decreases. One interesting finding is that, if the number of features in the first hidden layer of the model is larger, its capacity drops more. As we can see from Figure2: on Fashion-MNIST data set, when the number of features in second hidden layer decreases to 16, the average accuracy of the model with first hidden layer has 1024 features drops quickly to 10.13%, while the model which has 512 features in its first hidden layer still has 32.83% average accuracy, and this number increases to 85.85% if the the first hidden layer has 128 features. On Adult Income data set, similar phenomenon occurs when the last hidden layer has 2 features. This finding may indicate that the capacity does not always increase as the number of parameters in a model increases.

MLP and naive Bayes do not have sufficient capacity on either data set. While SVM with Gaussian kernels fits well on Fashion-MNIST data set, which surprised the team. It can 100% fit the training Fashion-MNIST data set when hyper-parameter C is larger than 10, and gamma is 1e-6. Their performances on validation and test data sets also beat other models using naive Bayes and multi-layer perceptron. As sklearn library doesn't open the access to control the leanring procedure, we

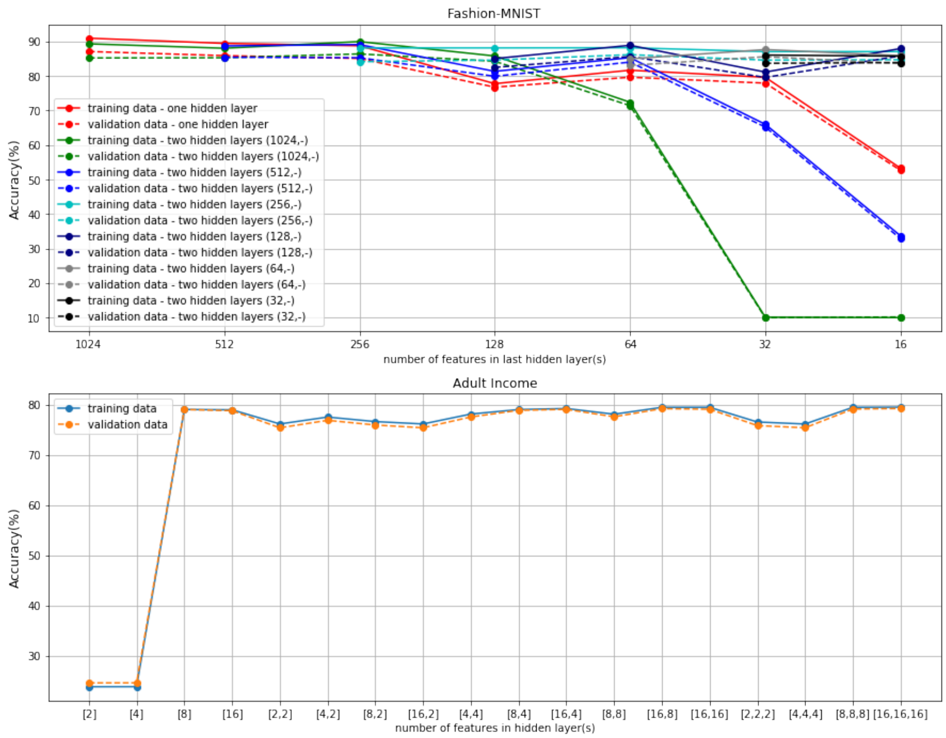

Figure 2: Learning curves of multi-layer perceptron with different hidden layer settings

don't know if the SVMs are over-fitted the training data set. In further research, we will monitor the learning proecedure to check if it is really over-fitted the data set.

As we expected, MLP doesn't have strong capacity on Fashion-MNIST data set, which is suitable for a convolutional network[8], nor does it on Adult Income data set. Nevertheless, it achieves the best performance on Adult Income data set, and its performance on Fashion-MNIST data set is very close to the best which is from SVM with Gaussian kernel.

The relationship between a MLP's capacity with the number of hidden layer and number of features in each hidden layer is very interesting. As we revealed in this paper, there is no direct linear relationship between them. In our experiments, although we have tried many hidden layers settings, we still believe it's not enough, and we should try more hyper-parameter setting before we can get insight of this issue.

Average accuracy is used to evaluate the performance of models. It is not a good criteria to evaluate the the performance of a model where examples are unbalanced. Adult Income data set is such the case. it has 24720 examples in one class(<=50k) yet only 7841 examples in another(>50k). Accuracy is not good enough to compare the models on this data set; therefore, F score is also considered as a metrics.

Acknowledgments

Contribution:

Jing Fang, Luo performed data-preprocessing, ran the experiments codes, and wrote related section of the paper;

Lifeng Wan performed data-preprocessing, ran the experiment codes, and wrote related section of the paper;

Qiang Ye planned the experiment, ran the experiment codes, and wrote other sections of the paper;

Yan Ai planned the experiment and collected the data;

Ying Xiao wrote and ran the experiment codes, and produced the Figures in the paper.

We thank Google for providing the Colab computational platform.

References

1. https://scikit-learn.org/stable/modules/naive_bayes.html
2. https://scikit-learn.org/stable/modules/svm.html
3. https://scikit-learn.org/stable/modules/neural_networks_supervised.html
4. https://elitedatascience.com/data-cleaning
5. https://www.valentinmihov.com/2015/04/17/adult-income-data-set/
6. https://en.wikipedia.org/wiki/Feature_selection
7. https://en.wikipedia.org/wiki/F1_score
8. CNN https://www.deeplearningbook.org/contents/convnets.html