RUTGERS UNIVERSITY

Image Classification

Project

Hanxiong Wang, Xiaobing Zhang

Contents

May 2017

1 Algorithm Implementation

In this project, we are supposed to classify two sets of data: digits and faces. For digits data, we should classify them into 0,1,2,...,9 correctly. For faces data, we should tell whether it is face or not. Three algorithms are implemented: Perceptron, Naive Bayes and large-margin (MIRA). Our code is implemented based on the library provided by the Pacman project of CS 188 at Berkeley (http://ai.berkeley.edu/classification.html).

1.1 Perceptron

For Perceptron method, it uses a weight vector of each label to make its decisions. For a given image, we can extract the corresponding feature lists vector.And for each label, we can give a score to them based on below equation:

$$score(f, y) = \sum_i f_i w_i^y$$

Both weight vector and feature list are counters in our codes, the score could be calculated by a simple dot product. And the guess label can be get based on a argument max equation:

$$y' = argmaxscore(f, y'')$$

If y' matches the training data label y, we do nothing. In other cases, we modify the weight vectors for y' and y accordingly:

$$w^y = w^y + f$$
$$w^{y'} = w^{y'} - f$$

In this way, for next iteration, we could possibly score y higher and y' lower, thus increasing the accuracy.

1.2 Naive Bayes Classifier

Naive Bayes classifier make the decisions based on probabilities. To get the full joint probability, we need to calculate prior distribution and conditional probabilities firstly. These two probabilities could be calculated and trained from the training data. With the trained prior distribution and conditional probabilities, for every image feature list, we can find the most probable label. To prevent underflow, log probabilities is used:

$$argmaxlogP(y|f_1, f_2, ..., f_m) = argmax\{logP(y) + \sum_{i=1} logP(f_i|y)\}$$

To make the estimate smoother, we add one term k to every possible observation value. And we can tune k to the best based on the validation accuracy.

$$P(F_i = f_i|Y = y) = \frac{c(f_i, y) + k}{\sum_{f_i' in\{0,1\}}(c(f_i', y) + k)}$$

1.3 An algorithm of your choice: MIRA

Very similar to Perceptron method but with different weight vector modification step size:

$$w^y = w^y + \tau f$$

$$w^{y'} = w^{y'} - \tau f$$

As to the value of τ , it is decided based on below equation:

$$\tau = min[C, \frac{(w^{y'} - w^y)f + 1}{2||f||_2^2}]$$

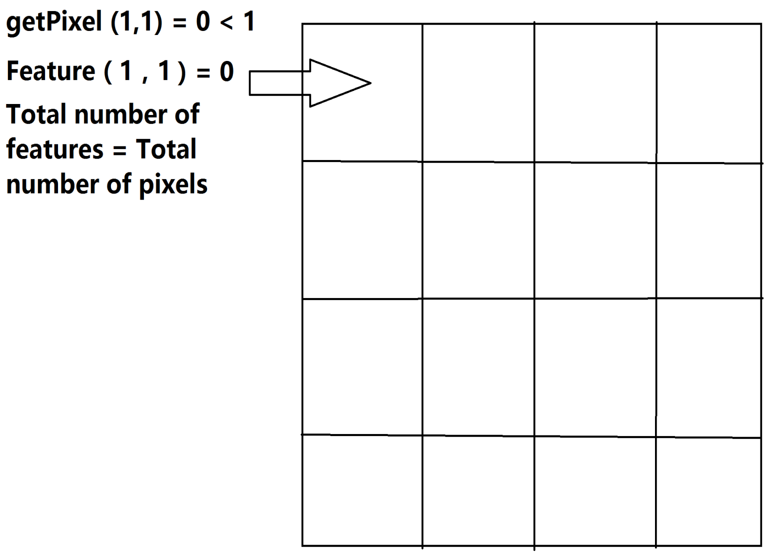
Value C is chosen to provide a limit to the value of τ.

2 Features Design

We have implemented one basic feature and one enhanced features.

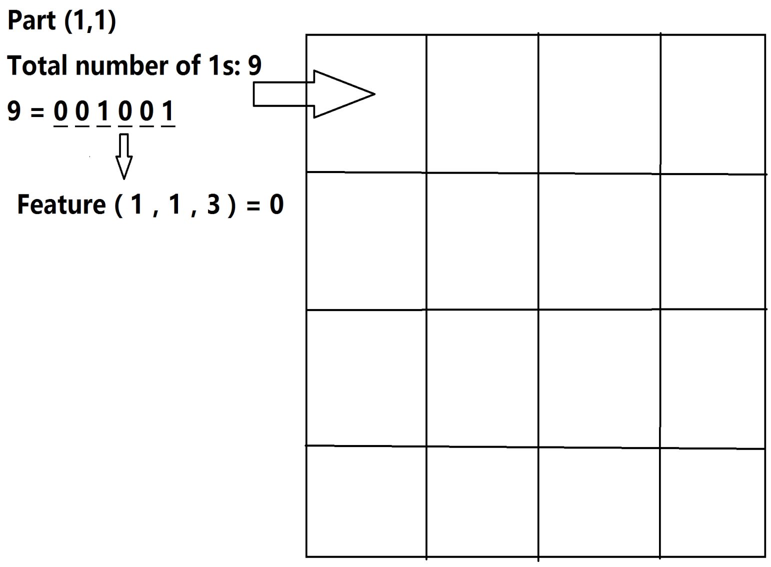
2.1 Basic feature

The basic feature takes each pixel as one element and it takes values 0 and 1. So for the digit data, the feature vector is a 28*28 vector. For the faces data, the feature vector is a 60*70 vector.



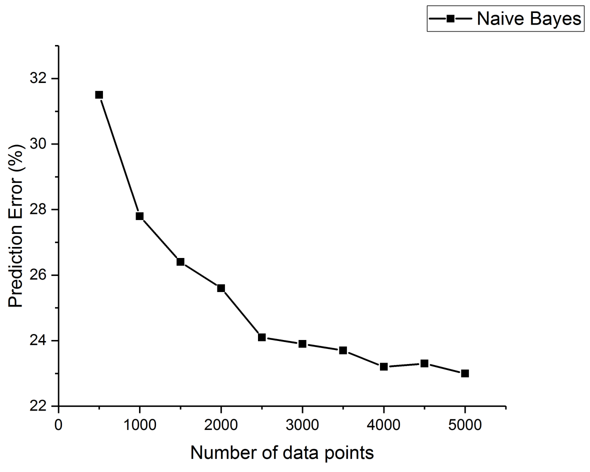
2.2 Enhanced feature

The enhanced feature divide the whole image into different parts (7*7 for digits and 15*17.5 for faces). For each part, we count the number of 1s N. N is converted into its binary format. And each binary digits is one feature element. Our enhanced feature does not increase the prediction accuracy, however, it reduces the time cost by a lot.

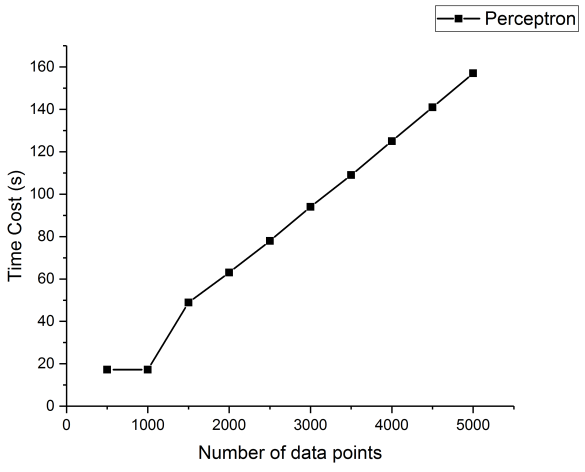


3 Performance

For the training data set, there are in total 5000 items for the digits data and 450 items for the faces. We could see from the error and time cost figures that as the number of data points increases, the prediction error reduces but the time costs increases.



(a) One example Error function.



(b) One example Time costs function.

Figure 1: Errors and time costs as a function of data points number (See tables below for detailed data).

3.1 For Basic feature

Table. 1 Basic Feature for digit data.

Digits	Perceptron		Naive Bayes (k=0.5)		MIRA	
No. of data points	Time cost	Prediction error	Time cost	Prediction error	Time cost	Prediction error
500 (10 %)	17.202	25%	50.08	31.5%	19	29%
1000 (20 %)	17.202	25%	52.05	27.8%	36	31%
1500 (30 %)	48.881000	25%	50.34	26.4%	54	19%
2000 (40 %)	63	18%	52.16	25.6%	70	16%
2500 (50 %)	78	15%	52.67	24.1%	85	15%
3000 (60 %)	94	15%	53.30	23.9%	82	19%
3500 (70 %)	109	17%	53.28	23.7%	118	17%
4000 (80 %)	125	18%	54.10	23.2%	135	13%
4500 (90 %)	141	15%	54.49	23.3%	152	19%
5000 (100 %)	157	19 %	55.36	23%	169	24%

Table. 2 Basic Feature for faces data.

Faces	Perceptron		Naive Bayes (k=0.5)		MIRA	
No. of data points	Time cost	Prediction error	Time cost	Prediction error	Time cost	Prediction error
45 (10 %)	4	38%	7.58	29%	4.8	28%
90 (20 %)	5.7	24%	7.93	23%	7	38%
135 (30 %)	7.6	28%	8.58	18%	9	16%
180 (40 %)	9.3	29%	9.27	12%	11	17%
225 (50 %)	10.7	21%	9.98	15%	13	17%
270 (60 %)	12.5	13%	10.69	14%	15	15%
315 (70 %)	14.3	20%	11.75	13%	17	14%
360 (80 %)	15.7	20%	12.25	12%	19	14%
405 (90 %)	17.7	18%	13.09	12%	21	16%
450 (100 %)	19	15%	13.90	12%	22.8	13%

3.2 For enhanced feature

Table. 3 Enhanced Feature for digit data.

Digits	Perceptron		Naive Bayes (k=0.5)		MIRA	
No. of data points	Time cost	Prediction error	Time cost	Prediction error	Time cost	Prediction error
500 (10 %)	8	38%	20.79	37.9%	9	40%
1000 (20 %)	15	36%	21.17	33.7%	16	32%
1500 (30 %)	22	23%	21.55	33%	25	25%
2000 (40 %)	29	26%	22.38	31.6%	33	29%
2500 (50 %)	36	25%	21.92	31.3%	40	28%
3000 (60 %)	42	26%	22.07	31.5%	48	31%
3500 (70 %)	50	25%	22.27	30.7%	55	31%
4000 (80 %)	57	22%	22.83	29.6%	62	26%
4500 (90 %)	65	27%	22.85	29.6%	70	25%
5000 (100 %)	69	23%	23.16	29.5%	77	25%

Table. 4 Enhanced Feature for faces data.

Faces	Perceptron		Naive Bayes (k=0.5)		MIRA	
No. of data points	Time cost	Prediction error	Time cost	Prediction error	Time cost	Prediction error
45 (10 %)	0.7	46%	0.43	45.4%	0.7	44%
90 (20 %)	0.6	44%	0.44	36%	1.2	43%
135 (30 %)	1.1	45%	0.47	26%	1.1	47%
180 (40 %)	0.9	30%	0.50	30%	1.3	38%
225 (50 %)	1.4	39%	0.53	26%	1.6	38%
270 (60 %)	1.5	39%	0.54	24%	1.9	27%
315 (70 %)	1.3	35%	0.57	22%	2.2	30%
360 (80 %)	1.6	29%	0.60	22%	2.2	31%
405 (90 %)	2	28%	0.62	19%	2.5	26%
450 (100 %)	2	29%	0.63	20%	2.8	35%

4 Conclusion

(1) From this project, we learned the basic theory of training and data classification, which include Perceptron, Naive Bayes and MIRA methods.
(2) Different features have been designed and performed. From the results, we could see that a good feature plays an very important role in the accuracy. For our case, even though the enhanced feature does not increase the accuracy. It reduces the time cost by a lot. So it is a good feature when the time cost is important.
(3) In general, as the number of data points increases, the prediction error reduces but the time cost increases. So a good balance should be chosen.
(4) More iterations will lead to a better accuracy but with more time cost.
(5)The prediction error for classification can be reduced to as low as 13% for digits data and 12% for faces data.

5 Reference

(1) UC Berkeley CS188 Intro to AI Course Materials (http://ai.berkeley.edu/classification.html)