School of Computer Science
Northwestern Polytechnical University

Experiment Report

——Classifier by Perceptron Algorithm

Jiajun Jiang

2014-5-18

一.Requirements

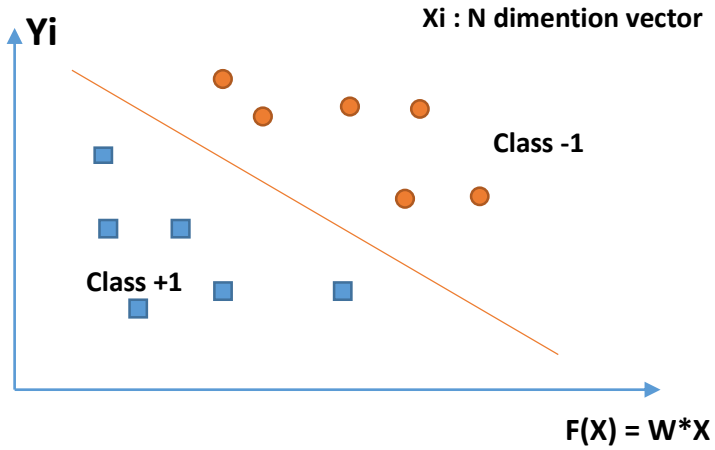
1. Use the datasets in data1 folder to implement a simple linear-classifier that can distinguish a specific input file from baseball and hockey. There are five groups' datasets having been classified for the five-fold Cross-validation.

2. Use the datasets in data2 folder to re-implement a simple linear-classifier that can distinguish a specific input file from positive and negative. Use the file s1 to s5 to Cross-validation.

3. Measure the precision, recall and f1 according to above.

二.Algorithm

The Perceptron Algorithm can be illustrated as bellow:



First we make an explanation for the variables we use next step:

Symbol	Instruction
D	Set of training files (or) All unique words in the training set of files
D[i]	Number of unique words in current processing file
y[i]	class(say -1 or 1)
x[i]	input variable
w	number of unique words in given

	set of files
alpha	learning rate

The Perceptron Algorithm can be described as:

1. **training procedure:**

```
while( !convergence){
        for(i = 0; i < D; i ++){            //processing the current file
            p = 0;
            for(j = 0; j < D[i]; j++){
                p += w[j] * x[j];
            }
        }
        if (y[i] * p <= 0 ){                //updating the new w
            for(j = 0; j < D[i]; j++){
                w[j] += alpha * y[i] * x[j];
            }
        }
}
```

2. **testing procedure:**

```
for(i = 0; i < D; i ++){
    p = 0;
    for(j = 0; j < D[i]; j++){      //processing the current file
        p += w[j] * x[j];
    }
    if(p <= 0){                 //predict the class according to the process
        predict->class1;
    }else{
        predict->class2;
     }
}
```

三.Implementation

1. For Data 1

   For this experiment, I chosen the java language to implement the algorithm. This experiment mainly consists of four parts:

   ->the first part is to divide the file into independent words;

   This part I used the NLPIR tool developed by Pro. Zhang Huaping from Beijing University to divide the file into independent words which called pre-processing, after which we can identify the attributes of a specific word, and then we can delete the useless words.

   ->the second part is the extraction of keywords;

   This part I referenced the TF-IDF algorithm. First I extracted the total words in the file with some simple processing, then selecting the most five words as a file's keywords and finally collecting all file keywords together as the final keywords' set.

   ->the third part is the training of the classifier;

   This part I used the perceptron algorithm above to train the classifier, and at the meantime, I have wrote the processed data into specific files. The keywords used was wrote into train1.txt in the folder keyword, and the training data was wrote into train1.txt in the folder weight. Also there may have a train2.txt file in each folder, which is created by another training after changing the training datasets.

   ->the forth part is the experimental test.

   This part is the simplest section. In this part I just using the classifier trained to testing the classification result, and then according to the result analysis the classifier performance, such as precision, recall, and f1.

See detailed implementation in code.

2. For Data 2

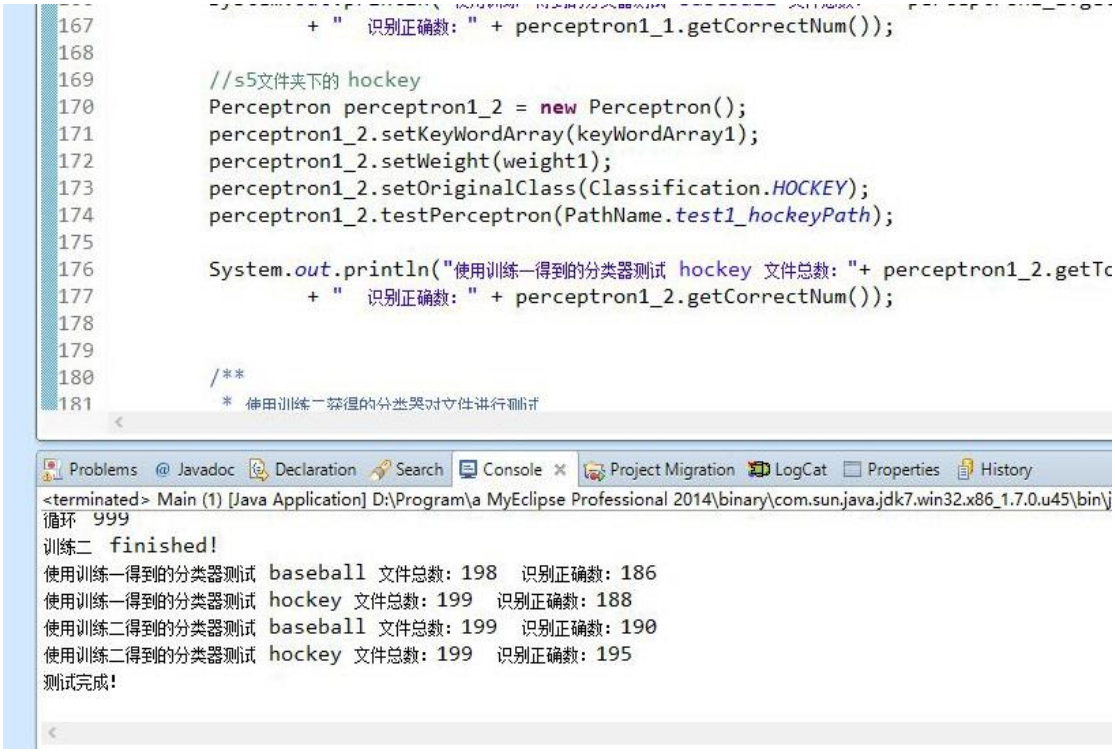
For this experiment, I chosen the language of C++ for coding. For it has the same algorithm as experiment one and just has some difference on the organization of data. So I will make a brief introduction.

First I also read the file and extracted the keywords. For the key words are integers and they may be discontinuous, I select the map in STL library of C++ to save the data. And then using the same way to train and test the classifier as the previous one.

四.Testing

1. For Data1

(1) Here is the running screenshots of experiment one:

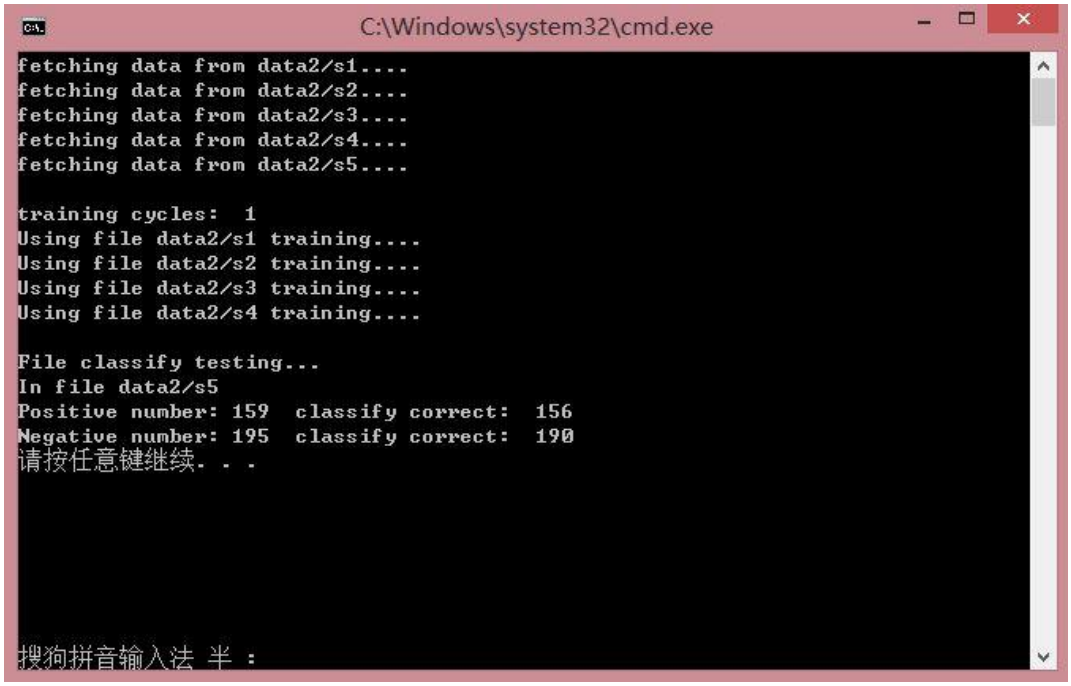


(2) Here is the process data screenshots (trained weight)：

```
2980 cycle:  993
2981 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
2982
2983 cycle:  994
2984 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
2985
2986 cycle:  995
2987 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
2988
2989 cycle:  996
2990 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
2991
2992 cycle:  997
2993 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
2994
2995 cycle:  998
2996 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
2997
2998 cycle:  999
2999 1.0  0.75  -1.0  4.25  3.5  4.5  -0.75  1.25  3.75  0.0  0.75  2.0  -1.0  -0.5  4.0  -0.75  0.0  0.0  2.5  1.25  -1.0  -0.25  1
3000
3001
```

2. For Data2

(1) Here is the running screenshots of experiment two:



五.Analysis

According to the above running screenshots we can get the following data tables:

1. For Data1

Train1 data:

	Actual class(expectation)	
Predicted class (observation)	tp: 186	fp: 12
	fn: 11	tn: 188

Percision $= tp/(tp + fp)$ = 186/(186+12) = 0.939

Recall $= tp/(tp + fn)$ = 186/(186+11) = 0.944

F1 $= 2 * \ （precision * recall）/（precision + recall）$

=2*(0.939*0.944)/(0.939+0.944) = 0.941

Train 2 data:

	Actual class(expectation)	
Predicted class (observation)	tp: 190	fp: 9
	fn: 4	tn: 195

Percision $= tp/(tp + fp)$ = 190/(190+9) = 0.955

Recall $= tp/(tp + fn)$ = 190/(190+4) = 0.979

F1 $= 2 * \ （precision * recall）/（precision + recall）$

=2*(0.955*0.979)/(0.955+0.979) = 0.967ssss

2. For Data2

	Actual class(expectation)	
Predicted class (observation)	tp: 156	fp: 3
	fn: 5	tn: 190

Percision $= tp/(tp + fp)$ = 156/(156+3) = 0.981

Recall $= tp/(tp + fn)$ = 156/(156+5) = 0.969

F1 $= 2 * \ （precision * recall）/（precision + recall）$

=2*(0.981*0.969)/(0.981+0.969) = 0.975

3. Through the above analysis we can know, the classifier is very accurate.

六.Summary

Through this experiment, I have a further understanding to the artificial intelligence and machine learning. Also I have learnt much knowledge that I couldn't learn from the classes. And this experiment

involves the theory of probability knowledge, which give me a good opportunity to apply the theory into practice. So it is a very meaningful experience for me. However, there still are many aspects that could be improved in my project, such as the running efficiency. I have just taken the simplest idea, therefore, its efficiency is not so good.