

CAP5638 Project 2

Classification Using Linear Discriminant Functions and Boosting Algorithms

Suhib Sam Kiswani

December 2, 2015

The algorithms were implemented in *Python 3.5*, with a dependence on the *scipy* [1] library.

1 Basic Two-Class Classification Using Perceptron Algorithms

Abstractly, the problem is as follows: Given n labeled training samples, $D = \{(x_1, L_1), (x_2, L_2), \dots, (x_n, L_n)\}$, where $L_i = \pm 1$, implement Algorithm 4 (Fixed-Increment Single-Sample Perceptron Algorithm) and Algorithm 8 (Batch Relaxation with Margin) of Chapter 5 in the textbook.

The algorithms used for this method are:

Algorithm 5.4 (Fixed-Increment Single-Sample Perceptron)

```
1: initialize  $a, k = 0$ 
2: do  $k \leftarrow (k + 1) \bmod n$ 
3:   if  $\mathbf{y}_k$  is misclassified by  $\mathbf{a}$  then  $\mathbf{a} \leftarrow \mathbf{a} + \mathbf{y}_k$ 
4: until all patterns properly classified
5: return  $a$ 
```

Algorithm 5.8 (Batch Relaxation with Margin)

```
1: initialize  $a, \eta(\cdot), b, k \leftarrow 0$ 
2: do  $k \leftarrow (k + 1) \bmod n$ 
3:    $\mathcal{Y}_k = \{\}$ 
4:    $j = 0$ 
5:   do  $j \leftarrow j + 1$ 
6:     if  $\mathbf{a}^t \mathbf{y}^j \leq b$  then Append  $\mathbf{y}^j$  to  $\mathcal{Y}_k$ 
7:   until  $j = n$ 
8:    $\mathbf{a} \leftarrow \mathbf{a} + \eta(k) \sum_{\mathbf{y} \in \mathcal{Y}_k} \frac{b - \mathbf{a}^t \mathbf{y}}{\|\mathbf{y}\|^2} \mathbf{y}$ 
9: until  $\mathcal{Y}_k = \{\}$ 
10: return  $a$ 
```

1.1 Results

TODO For each dataset, train a classifier to classify class 1 against the rest, class 2 against the rest, and class 3 against the rest by applying the two algorithms on the training set and then use the trained classifier on the test set (with labeling consistent with the training labeling). Document classification accuracy and iterations in training, and compare the two different algorithms. Note that USPS digit dataset may not be linearly separable and you then need to stop the algorithms in some way.

UCI Wine Data Set

Algorithm 5.4 (Fixed-Increment Single-Sample Perceptron)

| Training Statistics | | |
|---------------------|------------|-------------|
| Class | Iterations | Runtime (s) |
| ω_1 | 22864 | 27.498 |
| ω_2 | 52918 | 62.130 |
| ω_3 | 35600 | 127.227 |
| Total | | |

The weights for each class after training were:

$$\begin{aligned}
 \mathbf{a}_1 = & \begin{bmatrix} 61035.869 \\ -29616.504 \\ 21752.882 \\ 1483.633 \\ -25062.26 \\ 157.921 \\ 11769.882 \\ 39925.061 \\ -8012.231 \\ 32849.055 \\ 18856.758 \\ -8917.463 \\ 31289.131 \\ 568.08 \end{bmatrix} & \mathbf{a}_2 = & \begin{bmatrix} 137199.142 \\ 37238.975 \\ -18303.661 \\ 6240.703 \\ 5099.608 \\ -3584.203 \\ 28903.859 \\ -3043.506 \\ 21158.897 \\ -25565.128 \\ -60256.757 \\ 34175.689 \\ 20013.206 \\ -55.379 \end{bmatrix} & \mathbf{a}_3 = & \begin{bmatrix} 90871.97 \\ 2481.881 \\ -12760.566 \\ -7305.273 \\ 6178.2 \\ 3980.622 \\ -31851.585 \\ -51828.543 \\ -3346.438 \\ -17933.398 \\ 39059.302 \\ -20064.289 \\ -62224.546 \\ -595.573 \end{bmatrix}
 \end{aligned}$$

This resulted in 78 correct classifications out of 89 (87.6% accuracy).

Algorithm 5.8 (Batch Relaxation with Margin)

| Training Statistics | | |
|---------------------|------------|-------------|
| Class | Iterations | Runtime (s) |
| ω_1 | 100000 | 104.886 |
| ω_2 | 100000 | 146.686 |
| ω_3 | 100000 | 147.255 |
| Total | 300000 | 398.827 |

After training the perceptron using Algorithm 5.8, the weights were:

$$\mathbf{a}_1 = \begin{bmatrix} 0.582 \\ -0.213 \\ 0.487 \\ 0.728 \\ -0.381 \\ -0.069 \\ 0.881 \\ 0.103 \\ 0.246 \\ 0.188 \\ 0.494 \\ 0.443 \\ 0.534 \\ 0.008 \end{bmatrix} \quad \mathbf{a}_2 = \begin{bmatrix} 0.539 \\ 0.306 \\ 0.402 \\ 0.633 \\ -0.193 \\ -0.021 \\ 0.284 \\ 0.145 \\ 0.69 \\ 0.973 \\ -0.288 \\ 0.054 \\ 0.657 \\ -0.006 \end{bmatrix} \quad \mathbf{a}_3 = \begin{bmatrix} 0.371 \\ 0.141 \\ 0.511 \\ 0.841 \\ -0.176 \\ -0.042 \\ 0.062 \\ 0.114 \\ 0.545 \\ 0.337 \\ 0.601 \\ 0.142 \\ -0.14 \\ -0.003 \end{bmatrix}$$

This resulted in 83 correct classifications out of 89 (93.26% accuracy)

USPS Handwritten Digit Data Set

TODO

2 Multi-Class Classification

Use the basic two-class perceptron algorithms to solve multi-class classification problems by using the one-against-the-rest and one-against-the-other methods. Note that you need to handle ambiguous cases properly.

2.1 Results

TODO For each dataset, now train a classifier to classify all the classes using the one-against-the-rest and the one-against-the-other methods based on the two two-class algorithms, resulting in four different classifiers on each dataset and then classify the test set. Document classification accuracy, iterations in training, and classification time for test, and compare the one-against-the-rest and the one-against-the-other methods.

1. UCI Wine Data Set TODO
2. USPS Handwritten Digit Data Set TODO

3 Adaboost to Create Strong Classifiers

Implement Algorithm 1 (AdaBoost) in Chapter 9 of the textbook to create a strong classifier using the above linear discriminant functions.

Algorithm 9.1 (AdaBoost)

```
1: initialize  $\mathcal{D} = \{\mathbf{x}^1, y_1, \dots, \mathbf{x}^n, y_n, k_{max}, W_1(i) = 1/n, i = 1 \dots n\}$ 
2:  $k = 0$ 
3: do  $k \leftarrow k + 1$ 
4:   train weak learner  $C_k$  using  $\mathcal{D}$  sampled according to  $W_k(i)$ 
5:    $E_k \leftarrow$  training error of  $C_k$  measured on  $\mathcal{D}$  using  $W_k(i)$ 
6:    $\alpha_k \leftarrow 0.5 \ln [(1 - E_k)/E_k]$ 
7:    $W_{k+1}(i) = \frac{W_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k} & \text{if } h_k(\mathbf{x}^i) = y_i \text{ (correct classification)} \\ e^{\alpha_k} & \text{if } h_k(\mathbf{x}^i) \neq y_i \text{ (incorrect classification)} \end{cases}$ 
8: until  $k = k_{max}$ 
9: return  $C_k$  and  $\alpha_k$  for  $k = 1$  to  $k_{max}$  (ensemble of classifiers with weights)
```

3.1 Results

Boost Algorithm 8 to create a strong classifier for class 1 vs. class 2, class 1 vs. class 3, and class 2 vs. class 3 on the two datasets. Then classify the corresponding test samples from the relevant classes in test sets (in other words, for example, for the class 1 vs. class 2 classifier, you only need to classify test samples from classes 1 and 2); then document classification accuracy and show and analyze the improvement.

1. UCI Wine Data Set TODO
2. USPS Handwritten Digit Data Set TODO

4 Extra Credit

4.1 Support vector machines

By using an available quadratic programming optimizer or an SVM library, implement a training and classification algorithm for support vector machines. Then use your algorithm on the USPS dataset. Document the classification accuracy and compare the results with that from the two basic algorithms.

4.1.1 Results

4.2 Kernel method for linear discriminant functions

Given a kernel function, derive the kernel-version of Algorithm 4 and implement the algorithm, and then apply it on the given wine and USPS datasets. Document the classification accuracy and compare the results with that from the two basic algorithms without kernels. Use the polynomial function of degree three as the kernel function; optionally, you can use other commonly used kernel functions.

4.2.1 Results

TODO

1. UCI Wine Data Set TODO
2. USPS Handwritten Digit Data Set TODO

4.3 Multiple-class linear machines and multiple-class boosting

Use the Keslers construction to train a linear machine for multi-class classification and then use the SAMME algorithm to boost its performance on the training set. Apply the algorithm on both datasets and classify the corresponding test samples in the test sets. Document the classification accuracy and compare the results with that from the one-against-the-rest and one-against-the- other algorithms.

4.3.1 Results

TODO

1. UCI Wine Data Set TODO
2. USPS Handwritten Digit Data Set TODO

References

- [1] Jones E, Oliphant E, Peterson P, *et al.* **SciPy: Open Source Scientific Tools for Python**, 2001-, <http://www.scipy.org/> [Online; accessed 2015-10-24].