CogSci I: Exercise Perceptron

Barbara Plank, University of Copenhagen

September 8, 2014

Exercises

Today we have seen the perceptron classifier. Here are a couple of exercises related to the it. You can get the code and data from github: `https://github.com/bplank/cog-sci-1-2014` see folder `ex2-p`.

1. Given the following two vectors: $\mathbf{v_1} = \langle 5, -2, 3 \rangle$ and $\mathbf{v_2} = \langle -8, -7, -4 \rangle$. Compute the dot product between the two vectors.

2. Linear classifiers. Assume you have simple spam classifier with weight vector $\mathbf{w}$. The classifier is based on two features: number of times the token `viagra` appeared ($\phi_1(x)$) and whether the email mentioned your name ($\phi_2(x)$).
   $\mathbf{x_1}$: { `viagra: 2, name: 0`}
   $\mathbf{x_2}$: { `viagra: 4, name: 1`}
   $\mathbf{x_3}$: { `viagra: 1, name: 1`}
   $\mathbf{w}$: { `-1 2`}

   (a) Draw the three data points (feature vectors) and the weight vector in a 2-dimensional plot.

   (b) Which data points are classified as spam (+) or non-spam (-) using weight vector $\mathbf{w}$? Reminder: $\hat{\mathbf{y}} = \text{sign}(\sum_{j=1}^{m} \mathbf{w_j} \cdot \phi_\mathbf{j}(\mathbf{x}))$

   (c) Draw the line (hyperplane) associated with $\mathbf{w}$ that separates the positive and negative data points.

3. The perceptron's learning rule (also called the delta rule) is given by:

$$\mathbf{w} = \mathbf{w} + \Delta\mathbf{w}$$

$$\mathbf{w} = \mathbf{w} + \eta(\hat{\mathbf{y}} - \mathbf{y}) \cdot \phi(\mathbf{x})$$

   where η is the learning rate (assume it is 1.0), $\mathbf{y}$ is the correct label and $\hat{\mathbf{y}}$ is the current predicted label. Explain: What happens to the weight vector in the following cases: a) if $\hat{\mathbf{y}} = 1$ and $\mathbf{y} = 1$; b) if $\hat{\mathbf{y}} = 0$ and $\mathbf{y} = 1$.

4. Open the file `perceptron.py`:

(a) Explain every step in the python code to each other until the entire group feels it understands the entire process. What are the values of the weight vector after training?

(b) What is the predicted class for the new test data point (*testPoint*), what is its activation value (`score`)?

5. What are strengths and weaknesses of the perceptron algorithm?

6. Extend the code in `perceptron.py` to the averaged perceptron (cf. Section 2.5. in Collins, 2002: `http://www.aclweb.org/anthology/W02-1001`).