Everardo Villasenor

CS 7641

Problem Set 1

Problem Set 1

1. Derive the proper error function for finding the ML hypothesis using Bayes Rule

$d_i = \{1, 0\}$

Given $\{ \langle x_i, d_i \rangle \}$ $\qquad h_{ML} = \arg\max P(D \mid h)$

Assume i.i.d. $\qquad \cancel{= \arg\max \prod_i P(d_i \mid h)}$

1) $$= \underset{h \in H}{\arg\max} \prod_i P(x_i, d_i \mid h)$$

2) $$= \underset{h \in H}{\arg\max} \prod_i P(d_i \mid h, x_i)\, P(x_i)$$

hypothesis is non-deterministic

$$P(d_i \mid h, x_i) = \begin{cases} \cancel{1 \text{ if } d_i = 1} & h(x_i) \text{ if } d_i = 1 \\ \cancel{0 \text{ if } d_i = 0} & 1 - h(x_i) \text{ if } d_i = 0 \end{cases}$$

3) $$= \underset{h \in H}{\arg\max} \prod_i P(x_i)\, h(x_i)^{d_i} (1 - h(x_i))^{1-d_i}$$

~~[illegible]~~

not in h

4) $$= \underset{h \in H}{\arg\max} \sum_i \cancel{\ln(P(x_i))} + d_i \ln(h(x_i)) + (1-d_i) \cdot \ln(1 - h(x_i))$$

$$= \underset{h \in H}{\arg\max} \sum_i d_i \ln(h(x_i)) + (1-d_i) \cdot \ln(1 - h(x_i))$$

2. Difference to the deterministic rule:
The deterministic function tries to minimize error by minimizing the sum of squares function $(\sum_i (d_i - h(x_i))^2$. The non-deterministic function seeks to maximize the equation above in order to reduce error.

2c. Neural Network using sum of squared errors:

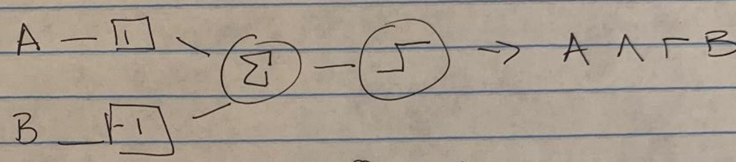
You would modify the error function (gradient descent) with the one derived and have it search for the argmax.

Y was an estimate of the probability instead of 0s and 1s:

This means we need an expression for the probability of d_i given that $h(x_i)$ is correct. Then it looks like the derivation using Gaussian noise, but it depends on what distribution of noise one chooses to add.

2. Design a two input perceptron for $A \wedge \neg B$

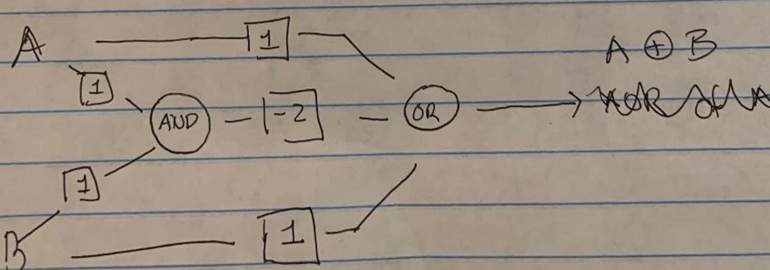
A	B	$A \wedge \neg B$
0	0	0
0	1	0
1	0	1
1	1	0



$\theta = 0.5$

Design a two-layer of $A \oplus B$:

A	B	$A \oplus B$ = OR - AND		
0	0	0	0	0
0	1	1	1	0
1	0	1	1	0
1	1	0	1	1



3. Derive the perceptron training rule and gradient descent training rule for a single unit with output o

$O = \omega_0 + \omega_1 x_1 + \omega_1 x_1^2 + \omega_n x_n + \omega_n x_n^2$

Perceptron Rule:

$O = \omega_0 + \omega_1 (x_1 + x_1^2) + \omega_n (x_n + x_n^2)$

where in our output function $O \Rightarrow \Sigma_i w_i x_i$

while error

$w_i = w_i + \Delta w_i$

$\Delta w_i = \eta (y - \hat{y}) x_i$

$\hat{y} = (\sum_i w_i x_i \geq \emptyset)$

while error:

* Thresholded first *

$\cancel{\hat{y} = o \geq \emptyset}$ $\quad w_i = w_i + \Delta w_i$

$\Delta w_i = \eta (y - \omega_0 + \omega_1 (x_1 + x_1^2) + \omega_n (x_n + x_n^2)) x_i$

$\hat{y} = (o \geq \emptyset)$

Gradient Descent:

$O = \omega_0 + \omega_1 (x_1 + x_1^2) + \omega_n (x_n + x_n^2)$

* Not thresholded

$E(\omega) = \frac{1}{2} \sum_{(x,y) \in D} (y - O_x)^2$

$$\frac{\partial E}{\partial w_i} = \frac{\partial}{\partial w_i} \frac{1}{2} \sum_{(x,y) \in D} (y - O_x)^2$$

$$= \sum_{(x,y) \in D} (y - O_x) \underbrace{\frac{\partial}{\partial w_i} (y - O_x)}$$

$$= 0 - \frac{\partial}{\partial w_i} \left[\cancel{\omega_0} + \cancel{\omega_1} (x_1 + x_1^2) + \cancel{\omega_n} (x_n + x_n^2) \right]$$

$$= \sum_{(x,y) \in D} (y - O_x)(-x_n - x_n^2)$$

$$\Delta w_i = \eta \sum_{(x,y) \in D} (y - O_x)(x_n + x_n^2)$$

The perceptron rule applies to problems that are linearly seperable, otherwise it will not converge. The gradient descent method can be applied to linear and not linear seperable problems. This is good because problems of higher dimensions are tough to know beforehand if they are linearly seperable. Therefore using calculus gradient descent is a more robust function.