Introduction to Machine Learning, Fall 2015 - Exercise session VI

Rodion “rodde” Efremov, student ID 013593012

December 4, 2015

Problem 3 (3 points)

Given a set of N points $y_1, \dots, y_N$, with each $y_i \in \mathbb{R}$, show that

(a) the value y^* which minimizes the sum of *squared* errors, i.e.

$$y^* = \arg\min_{\hat{y}} \sum_{i=1}^{N} (y_i - \hat{y})^2$$

is given by the *mean* of the y_i, i.e. $y^* = \sum_i y_i / N$.

(b) the value y^* which minimizes the sum of *absolute* errors, i.e.

$$y^* = \arg\min_{\hat{y}} \sum_{(} i = 1)^N |y_i - \hat{y}|$$

is given by the *median* of the y_i.

Solution to (a)

Let us first rewrite the sum:

$$f(x) = \sum_{i=1}^{N} (y_i - x)^2 = \sum_{i=1}^{N} (y_i^2 - 2y_i x + x^2).$$

We want to find the value of x which will minimize $f(x)$. Derivating $f(x)$ with respect to x:

$$\frac{\mathrm{d}}{\mathrm{d}x} f(x) = \sum_{i=1}^{N} (-2y_i + 2x),$$

since derivation is a linear operator. We will also need the second derivative of f:

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2} f(x) = \sum_{i=1}^{N} 2 = 2N.$$

Since $N > 0$, we have that the second derivative is always positive. Also, a root of the first derivative of f is given by equation

$$\sum_{i=1}^{N}(2x - 2y_i) = 0,$$

which leads to

$$\begin{aligned} \sum_{i=1}^{N} 2x &= \sum_{i=1} 2y_i, \\ 2Nx &= \sum_{i=1}^{N} 2y_i, \\ Nx &= \sum_{i=1}^{N} y_i, \\ x &= \frac{1}{N}\sum_{i=1}^{N} y_i, \end{aligned}$$

which is a requested average of $y_1, \dots, y_N$. Note since the second derivative is always positive, f attains its **minimum** at x.

Solution to (b)

Suppose that the values $y_1, \dots, y_N$ are **sorted** ($y_1 \le y_2 \le \cdots \le y_N$). Now consider the smallest and the largest values: y_1 and y_N, respectively. Denote by x the value that minimizes the sum of absolute errors. As long $x \in [y_1, y_N]$, the sum $|x - y_1| + |y_N - x|$ is minimized (if you take x outside of $[y_1, y_N]$, you will increase the value of $|x-y_1|+|y_N-x|$ beyond its minimal possible value). Now, in order to minimize the sum of absolutes errors, we **must** keep x within the range $[y_1, y_N]$. Now consider the second smallest and the second largest values: y_2 and y_{N-1}, respectively. Now since $y_2 \ge y_1$ and $y_{N-1} \le y_N$, keeping x within the range $[y_1, y_N]$ is not enough as x might be outside of the range $[y_2, y_{N-1}]$ which will increase the sum of absolute errors above its minimum. Using the same reasoning, we continue this until reaching a median (or two of them).

If N is even, the strongest bound for x is $[y_{N/2}, y_{N/2+1}]$. We can choose

$$x = \frac{y_{N/2} + y_{N/2+1}}{2},$$

which is, in fact, the median of $(y_i)_{i=1}^{N}$. However, if N is odd, we must satisfy $x \in [y_{(N+1)/2}, y_{(N+1)/2}]$, which is satisfied only if $x = y_{(N+1)/2}$, which is a median once again.

Problem 3 (15 Points)

The confusion matrix for one-versus-all Perceptron:

	0	1	2	3	4	5	6	7	8	9
0	2796	0	9	27	7	51	23	33	21	20
1	0	3196	52	17	24	35	12	56	106	25
2	21	15	2643	209	14	32	45	160	151	24
3	2	29	38	2340	3	104	1	44	58	29
4	16	1	52	14	2592	31	119	91	33	207
5	47	27	17	275	8	2205	31	14	192	35
6	32	5	80	20	30	56	2681	0	7	2
7	1	5	13	9	3	3	0	2443	1	28
8	46	36	91	71	29	169	30	35	2348	88
9	1	5	15	76	206	26	1	282	59	2488

The confusion matrix for all-versus-all Perceptron:

	0	1	2	3	4	5	6	7	8	9
0	2906	0	5	6	8	9	11	2	10	5
1	1	3235	23	4	6	16	4	4	19	7
2	42	37	2773	23	35	11	30	30	17	12
3	24	18	110	2695	4	97	9	12	72	17
4	11	11	12	1	2726	13	17	5	5	115
5	68	38	38	198	35	2110	46	1	168	10
6	52	9	67	1	49	29	2723	0	13	0
7	19	27	57	47	56	24	0	2791	5	132
8	36	80	108	67	53	71	19	3	2461	78
9	18	19	4	51	167	28	0	51	16	2592

The one-vs-all got 25732 (86%) correct classifications and all-vs-all got 27012 (90%) corect classifications. As far as recall the accuracy is comparable to prototype based classifier if not better. No, I do not see anything fancy in the confusion matrices except that some mismatches have larger frequency due to the same digit geometry.