Assignment 2, Machine Learning Fall 2018

Cristian Mitroi, dmn470

4 December 2018

Contents

1 Illustration of Hoeffding Inequality

1.1 Plot

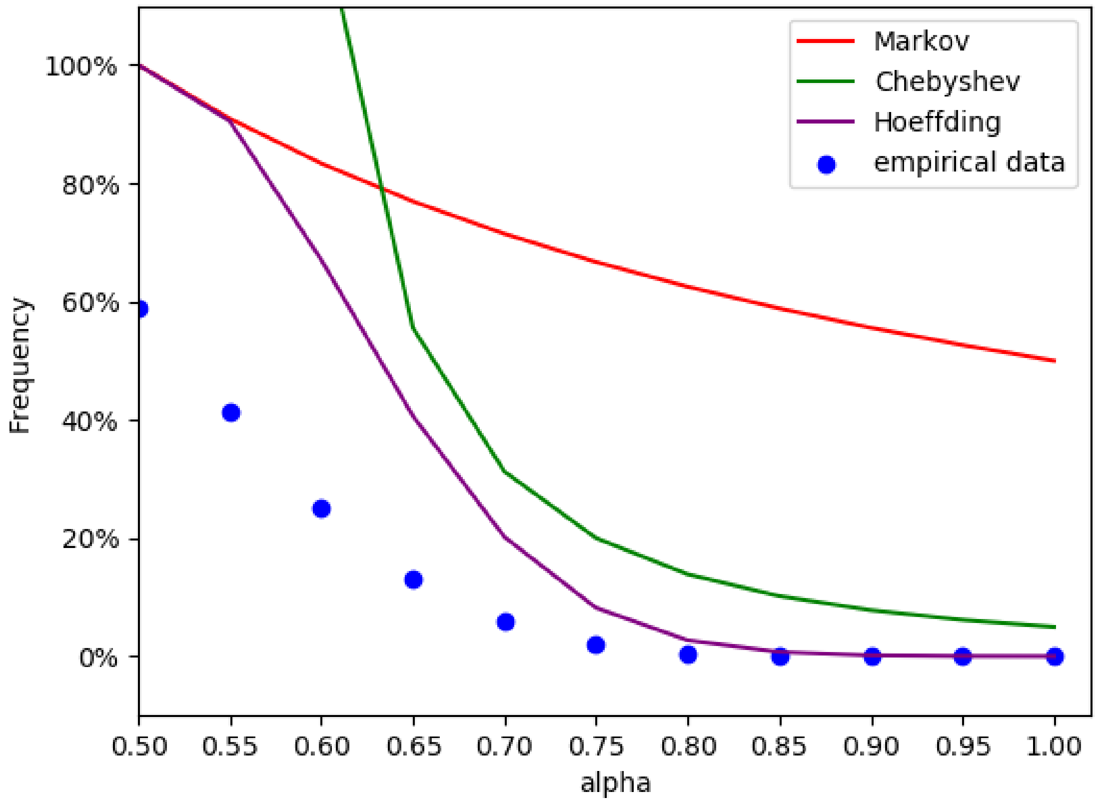

Figure 1: Plot

1.2 Comparison

We can see in fig. 1 that Hoeffding is tighter than both Markov and Chebyshev.

1.3 Probabilities for 0.95 and 1

Hoeffding values for 0.95 and 1: $3.03539138e - 04$ and $4.53999298e - 05$

Calculated probabilities were: $(\alpha = 1) = 9.536743164062511e - 27$ and $(\alpha = 0.95) = 1.9073486328125e - 05$.

These were computed using the Binomial random variable:

$\binom{20}{20}0.5^{20}0.5^0 = 0.5^{20}$

$\binom{20}{19}0.5^{19}0.5^1 = 20 \cdot 0.5^{20} = 1.9073486328125e - 05$

These are below the Hoeffding bounds.

2

$$P(\frac{\sum_{i=1}^n Z_i}{n} - \mu \geq \epsilon)$$

$$= P(\sum_{i=1}^n Z_i - n\mu \geq n\epsilon)$$

For any $\lambda > 0$, using Chernoff's technique:

$$= P(e^{\lambda(\sum_{i=1}^n Z_i - n\mu)} \geq e^{\lambda n\epsilon})$$

$$\leq \frac{E[e^{\lambda \sum_{i=1}^n (Z_i - \mu)}]}{e^{\lambda n\epsilon}}$$

$$= \frac{\prod_{i=1}^n e^{\lambda(Z_i - \mu)}}{e^{\lambda n\epsilon}}$$

$$\leq \frac{(e^{\frac{\lambda^2}{8}})^n}{e^{\lambda n\epsilon}}$$

$$= e^{n\frac{\lambda}{8} - \lambda n\epsilon} \leq e^{-2n\epsilon^2}$$

q.e.d.

3 Practical question

3.1.

$P(Z \leq z) \leq 0.05$

$P(100 - Z \leq 100 - z) \leq 0.05$

$P(Q \leq 100 - z) \leq 0.05$

$P(-Q \geq z - 100) \leq 0.05 = \frac{E[-Q]}{z-100}$

We have that $p = E[X] = 50$ and:

$E[-Q] = -1 \cdot E[Q] = -1 \cdot E[100 - Z] = -1 \cdot (100 - 50) = -50$

$z - 100 = \frac{-50}{0.05}$

$z = -900$

Which is a vacuous answer for z.

3.2

Let $Z = \hat{Z}$ for convenience

$$P(Z \leq z) \leq 0.05$$

$$P(Z - E[Z] \leq z - E[X]) \leq 0.05$$

$$P(50 - Z \geq 50 - z) \leq 0.05 = \frac{Var[Z]}{(50 - z)^2} \leq \frac{Var[Y]}{(50 - z)^2}$$

$$Var[Y] = Var(0, 100) = 2500$$

$$2500/(50 - z)^2 \geq 0.05$$

$$|50 - z| \geq \sqrt{50000}$$

$$50 - z \geq 223$$

$$-z \geq 173$$

$$z \leq -173$$

(which is a vacuous answer)

$$50 - z \geq -223$$

$$-z \geq -273$$

$$z \leq 273$$

So maximal value of z is 273.

3.3

$$P(\mu - \sum X_i \geq \epsilon) \leq e^{-2n\epsilon^2}$$

$$P(50 - \sum Z \geq \epsilon) \leq e^{-14\epsilon^2}$$

$$P(-\sum Z \leq 50 - \epsilon) \leq e^{-14\epsilon^2}$$

$$z = 50 - \epsilon$$

Thus

$$P(-\sum Z \leq 50 - \epsilon) \leq e^{-14(50-z)^2} = 0.05$$

We calculate:

$$e^{-14(50-z)^2} = 0.05$$

$$-14z^2 = -3, z = \{-3/14, 3/14\}$$

For $z = -3/14$ we have a vacuous value.

We have $z = 3/14$

4 Airline questions

4.1

$P(Y = 100) = \binom{100}{100} p^{100}(1-p)^0$

$= \binom{100}{100} p^{100}$

$= \frac{100!}{100! \cdot 0!} \cdot p^{100}$

$= p^{100} = 0.05^{100}$

4.2

We have two events:

$A =$ observed turn-out of 9500 out of 10000 $= P(\sum^{10000} X_i = 9500)$

In order to bound this using Hoeffding, we transform A by dividing by 10000 and then subtracting p:

$A = P(\frac{\sum^{10000} X_i}{10000} - p = 0.95 - p)$

We can then bound this by:

$A \leq P(\frac{\sum^{10000} X_i}{10000} - p \geq 0.95 - p)$

Which can then by bound using Hoeffding:

$A \leq P(\frac{\sum^{10000} X_i}{10000} - p \geq 0.95 - p) \leq e^{-2n\epsilon^2} = e^{-20000(0.95-p)^2}$

and the other event:

$B =$ probability that the flight is overbooked $=$ out of 100 tickets sold, all people show up $= P(\sum^{100} X_i = 100) = \binom{100}{100} p^{100} p^0 = p^{100}$

In order to calculate the probability that they both occur simultaneously we do:

$P(A)P(B) = e^{-20000(0.95-p)^2} \cdot p^{100}$

In order to obtain p we find value of p in range $[0, 1]$ that maximizes the above expression:

(See code in `ex4.py`)

$p = 0.9526$

Thus the probability of $B = 0.9526^{100} = 0.0078$

5 Logistic Regression

5.1

5.1 a)

Thus we have max likelihood from previous page:

$$\frac{1}{N} \sum_{n=1}^{N} \ln(\frac{1}{P(y_n|x_n)}) \tag{1}$$

$= \frac{1}{N} \sum_{n=1}^{N} (y \cdot \ln(\frac{1}{P(y=1|x)}) \cdot (1-y) \ln(\frac{1}{P(y=-1|x)}))$

We have:

$P(y|x) = h(x)$,if $y = +1$

$P(y|x) = 1 - h(x)$, if $y = -1$

Thus eq. 1 is:

$= \frac{1}{N}\sum_{n=1}^{N}(y \cdot \ln(\frac{1}{h(x)}) \cdot (1-y)\ln(\frac{1}{1-h(x)}))$

q.e.d.

5.1 b)

We replace $h(x_n)$ with $\theta(w^t x)$ in the equation in a):

$$E_{in}(w) = \sum_{n=1}^{N}[y=+1]\ln\frac{1}{\theta(w^t x)}[y=-1]\ln\frac{1}{1-\theta(w^t x)} \tag{2}$$

We apply *sigmoid* function:

$\frac{1}{\theta(w^t x)} = \frac{1}{\frac{e^{w^t x}}{1+e^{w^t x}}}$

$= 1 \cdot \frac{1+e^{w^t x}}{e^{w^t x}} = \frac{1+e^{w^t x}}{e^{w^t x}}$

$= e^{w^t x}(1+\frac{1}{e^{w^t x}})/e^{w^t x} = 1 + \frac{1}{e^{w^t x}}$

and:

$\frac{1}{1-\theta(w^t x)} = \frac{1}{1-\frac{e^{w^t x}}{1+e^{w^t x}}}$

$= 1 + e^{w^t x}$

Thus eq. 2 is:

$E_{in}(w) = \sum_{n=1}^{N}[y=+1]\ln(1+\frac{1}{e^{w^t x}})[y=-1]\ln(1+e^{w^t x})$

Which is equal to 3.9 when we replace for $x \in \{\pm 1\}$:

$E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N}[y=+1]\ln(1+e^{-w^t x_n})[y=-1]\ln(1+e^{w^t x_n})$

$= \frac{1}{N}\sum_{n=1}^{N}[y=+1]\ln(1+(1/e)^{w^t x_n})[y=-1]\ln(1+e^{w^t x_n})$

$= \frac{1}{N}\sum_{n=1}^{N}[y=+1]\ln(1+\frac{1}{e^{w^t x_n}})[y=-1]\ln(1+e^{w^t x_n})$

q.e.d.

5.2

5.2. a

We use the following shorthand:

$$y = y_n$$

$$x = x_n$$

$$w = w^T$$

We apply partial derivative wrt w of 3.9:

$$\frac{\partial E}{\partial w} = \frac{\partial E}{\partial w} \frac{1}{N} \sum_{n=1}^{N} (1 + e^{-ywx})$$

We ignore sum and division in calculating the derivative.

We take

$$t = 1 + e^{-ywx} \tag{3}$$

The derivative of $log(t) = \frac{1}{t}$

We apply the chain rule:

Multiply eq. 3 by

$$\frac{\partial}{\partial w}(1 + e^{-ywx}) \tag{4}$$

In order to do this:

First we calculate the derivative of $1 + e^{-ywx}$:

1. Derivative of 1 is 0
2. Let

$$g = -ywx$$

3. Then

$$\frac{\partial}{\partial g} e^g = e^g ln(e)$$

4. Chain rule again. Multiply by

$$\frac{\partial}{\partial w}(-ywx) = -yx$$

Result is =

$$-e^{-ywx}xylog(e) = -e^{-ywx}xy$$

The result of first chain rule of eqns. 3, 4 is thus:

$$= \frac{-e^{-ywx}xy}{1 + e^{-ywx}}$$

Now we simplify:

$$= -\frac{e^{-ywx}xy}{1 + e^{-ywx}}$$

$$= -\frac{e^{-ywx}(xy)}{e^{-ywx}(\frac{1}{e^{ywx}} + 1)}$$

$$= -\frac{xy}{(\frac{1}{e})^{ywx} + 1}$$

$$= -\frac{xy}{e^{ywx} + 1}$$

q.e.d.

5.2. b

In order to show that a misclassified point has a greater contribution to the gradient we compare the gradient for two sample points, one that is misclassified and one that is correctly classified:

Correct classification point: $y_n = +1$, $w^T x_n = +1$

Then we have (from the second form of E_{in}):

$E_{in}(w) = -x_n\theta(-1) = -x_n \frac{e^{-1}}{1+e^{-1}}$

Incorrect classification point: $y_n = -1$, $w^T x_n = +1$

Then we have:

$E_{in}(w) = x_n\theta(1) = x_n \frac{e}{1+e}$

We can see that the incorrect classification point yields a larger gradient.

5.3 Logistic Regression implementation

Firstly, my implementation relies on the intercept being already added to the data. I do this when applying the model to the Iris dataset. This can be changed to be added by the algorithm. I chose to separate that to the data preprocessing steps (in `load_data`).

Firstly, I decide on a **number of iterations** (or *epochs*, the number of times we will compute the gradient and adjust the weights) and a **learning rate** (the size of the step taken by gradient descent down the slope of the function). I choose 100000 and 0.04, respectively. These are crucial for the algorithm to be able to find the global minimum (or even a good local minimum).

We initialize the weights to zeros:

```
self.W = np.zeros(x.shape[1])
```

The algorithm starts as follows:

```
for _ in range(num_steps):
```

- For each of the epochs, we do the following:

```
h = self.sigmoid(np.matmul(x, W))
```

- compute the sigmoid of the matrix multiplication of the data and the weights

```
grad = np.matmul(x.T, (h - y)) / len(y)
```

- we then take the gradient as being the division between the matrix multiplication of the data and the difference between `h` (see above) and the labels, and the number of labels (or data points)

```
W = W - lr * grad
```

- finally, we adjust the weights by subtracting the gradient times the learning rate

For computing predictions we perform:

```
y_pred = self.sigmoid(np.matmul(x, self.W))
y_pred = (y_pred > .5).astype(int)
```

We perform matrix mult. between the data and the weights. We then round the numbers up and down.

5.4 Iris dataset

As mentioned in the previous section, the learning rate and nr. of iterations are crucial in this step. I have chosen as mentioned, and get these numbers:

- 0-1 loss on training data: 0.0161
- loss on test data: 0.0385
- weights: 5.6145; -31.1992; -20.1178; (the last one being the intercept)