January 2022

[1] $R(q) = \sum_x \sum_k p(x,k)\, w(k, q(x))$

$$q^*(x) = \operatorname*{argmin}_d R(x,d) = \operatorname*{argmin}_d \sum_k p(x,k)\, w(k,d) =$$

(a) $$p(T|1) = \frac{p(1|T)\,p(T)}{p(1)} = \frac{p(1|T)\,p(T)}{\sum_k p(1,k)} =$$

$$\frac{p(1|T)\,p(T)}{\sum_k p(1|k)\,p(k)} = \frac{0,2\cdot 0,01}{0,2\cdot 0,01 + 0,1\cdot 0,99} \approx 0,02$$

(b)

$p(x,k)$	0	1	2
E	0,8811	0,099	0,0099
T	0,0001	0,002	0,0079

$w(k,d)$	Y	N
E	0	1
T	1000	0

$$q^*(0) = \operatorname{argmin} \{ \overset{Y}{0,8811\cdot 0 + 0,0001\cdot 1000}; \overset{N}{0,8811\cdot 1 + 0,0001\cdot 0} \} = Y$$

$$q^*(1) = \operatorname{argmin} \{ 0,099\cdot 0 + 0,002\cdot 1000; 0,099\cdot 1 + 0,002\cdot 0 \} = N$$

$$q^*(2) = \operatorname{argmin} \{ 0,0099\cdot 0 + 0,0079\cdot 1000; 0,0099\cdot 1 + 0,0079\cdot 0 \} = N$$

(c) We want to minimize Bayesian risk

$$R(q) = \sum_x \sum_k p(x,k)\, w(k, q(x))$$

to find optimal strategy $q^* = \operatorname*{argmin}_q R(q)$

This can be achieved by minimizing partial risk

$$R(x,d) = \sum_{k} p(x,k)\, w(k,d) \quad \forall x$$

Then

$$q^*(x) = \underset{d}{\operatorname{argmin}}\ R(x,d)$$

[2] (a) Assuming random variables are i.i.d

$$p(n) = p(\overline{six})^{n-1} p(six) = (1-p(six))^{n-1} p(six) =$$

$$= (1-q)^{n-1} q$$

(b) $\max\limits_{q} \ (1-q)^{k-1} q$

Log-likelihood: $l = (k-1)\ln(1-q) + \ln q$

$$\frac{\partial l}{\partial q} = \frac{k-1}{1-q}\cdot(-1) + \frac{1}{q} = 0$$

$$\frac{k-1}{1-q} = \frac{1}{q} \longrightarrow 1-\cancel{q} = qk-\cancel{q}$$

$$\hat{q} = \frac{1}{k}$$

(c) $\max\limits_{q} \ (1-q)^{n_{max}} \longrightarrow \hat{q} = 0$

[3] (a) Vanishing gradient problem might occur during back propagation especially when using activation functions like sigmoid, tanh or with very deep models

To avoid it one might use ReLU or skip connections.

(b) It is a function that returns maximum value of some neighbourhoods. In some architectures used to reduce input dimensionality.

(c) It is a function that takes a vector and returns a vector with values in range $[0, 1]$ which may be interpreted as probabilities

(d) SGD is a method for minimizing/maximizing functions computed only on a subset of available training data

[4]

(1) Initialize cluster centers S_k

(2) Assign each data point to its nearest cluster

$$r_k = \{ x \mid \|x - S_k\|^2 \leq \|x - S_j\|^2 \; \forall j \}$$

(3) Update cluster centers

$$S_k = \begin{cases} \operatorname{argmin}_S \sum_{x \in r_k} \|S - x\|^2 & \text{if } r_k \neq \emptyset \\ \text{reinitialize} & \text{if } r_k = \emptyset \end{cases}$$

(4) Terminate if $r_k^{t+1} = r_k^t \; \forall k$, otherwise goto 2

The difference of k-means++ from k-means is that initializes cluster centers in a smart way.

Initialization step:

(1) Choose s_1 uniformly at random from T.
Set $S = \{s_1\}$

(2) For each data point compute its distance from the nearest cluster center

$$d_i = \min_{s_j \in S} \| x_i - s_j \| \quad \forall i$$

(3) Construct probability distribution

$$p(i) = \frac{d_i^2}{\sum_j d_j^2}$$

And sample from it to get the index i

(4) $S \longleftarrow S \cup \{x_i\}$

Reason to use k-means++ over k-means:
Usually finds better local minima as the solution.

January 2021

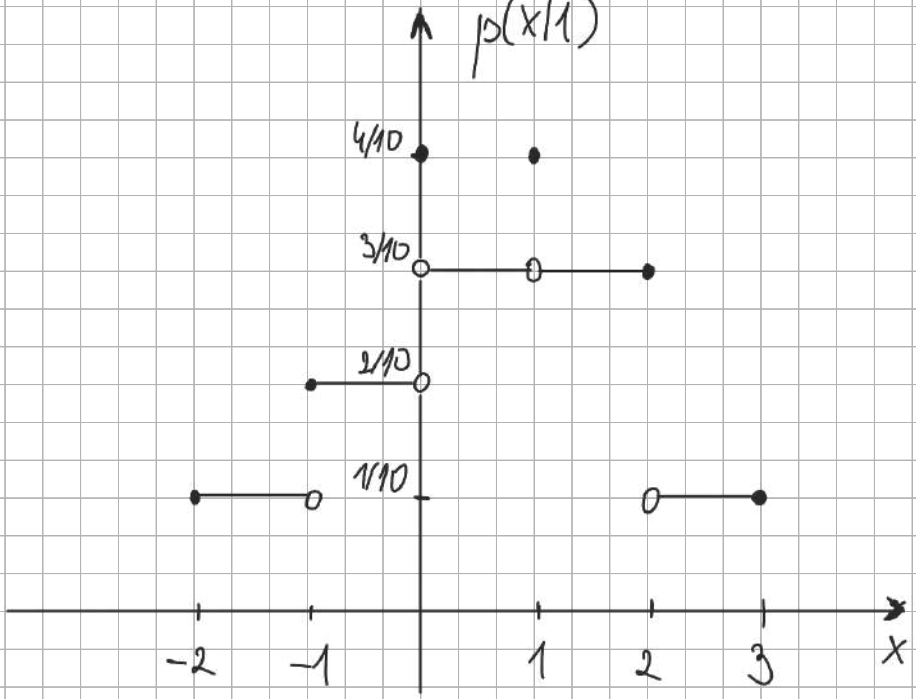
[1] (a) $p(x) = \frac{1}{n} \sum_{i=1}^{n} k(x - x_i)$

where $k(x) = \begin{cases} \frac{1}{h} & -\frac{h}{2} \leq x \leq \frac{h}{2} \\ 0 & \text{elsewhere} \end{cases}$

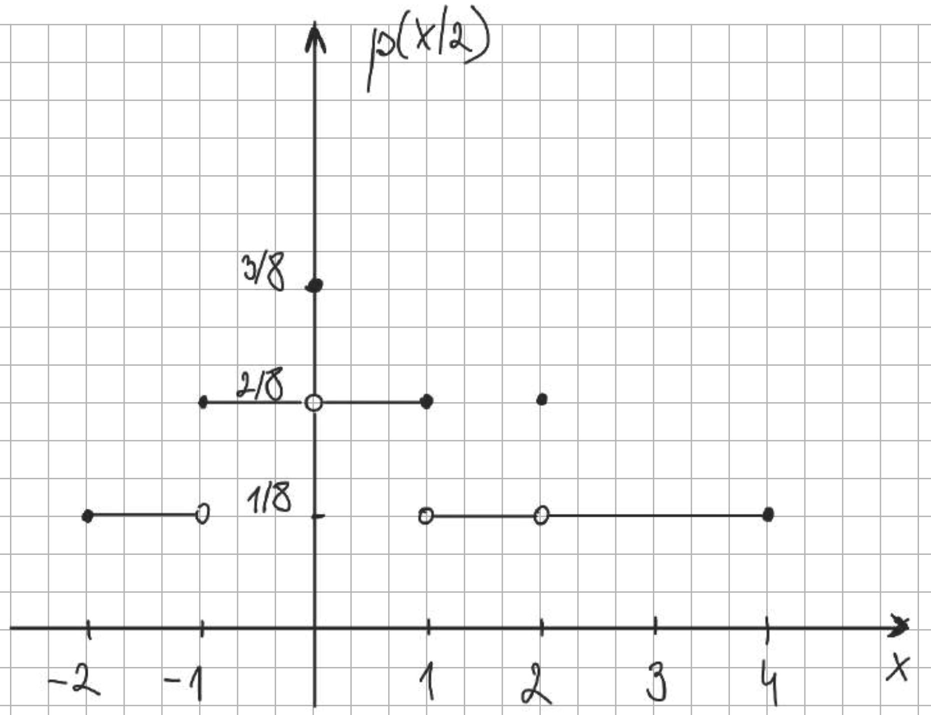
Choose $h=2$, then

$$k(x) = \begin{cases} \frac{1}{2} & -1 \leq x \leq 1 \\ 0 & \text{elsewhere} \end{cases}$$

$X_1 = \{1, -1, 1, 2, 0\}$



$X_2 = \{-1, 0, 1, 3\}$



$$p(X=0{,}5\,|\,1) = 0{,}3$$

$$p(x=0{,}5\,|\,2) = 0{,}25$$

(b) $\varepsilon_k = \int\limits_{x|q(x)\neq k} p(x|k)\,dx$ classification error for class k

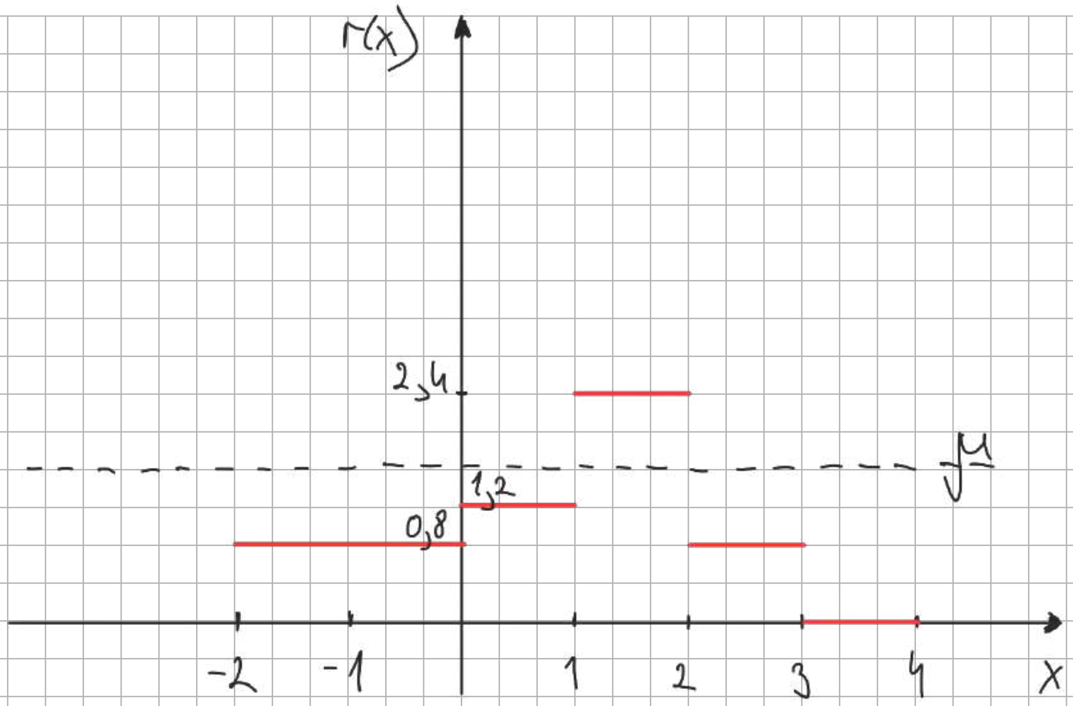
$\min\limits_{q} \; \varepsilon_1$

subject to $\varepsilon_2 < 0{,}4$

Compute likelihood ratio $r(x) = \dfrac{p(x|1)}{p(x|2)}$

Find threshold μ, then

$$q^*(x) = \begin{cases} 1 & \text{if } \dfrac{p(x|1)}{p(x|2)} \geq \mu \\ 2 & \text{otherwise} \end{cases}$$



Jump discontinuties don't matter so much. They don't impact overall classification error Because integral "ignores" them.

(i) $\mu = 0$ then $\varepsilon_1 = 0$ $\varepsilon_2 = 1$

(ii) $\mu \in (0; 0{,}8]$ then $\varepsilon_1 = \int_3^4 p(x|1)\,dx = 0$

$$\varepsilon_2 = \int_{-2}^{3} p(x|2)\,dx = \frac{7}{8}$$

(iii) $\mu \in (0{,}8; 1{,}2]$ then $\varepsilon_1 = 1 - \int_0^2 p(x|1)\,dx = 0{,}4$

$$\varepsilon_2 = \int_0^2 p(x|2)\,dx = \frac{3}{8} = 0{,}375$$

(iv) $\mu \in (1{,}2 \,;\, 2{,}4]$ then $\varepsilon_1 = 1 - \int_1^2 p(x|1)dx = 0{,}7$

$$\varepsilon_2 = \int_1^2 p(x|2)dx = \frac{1}{8}$$

(v) $\mu \in (2{,}4 \,;\, +\infty)$ then $\varepsilon_1 = 1$ $\quad \varepsilon_2 = 0$

$\mu^* \in (0{,}8 \,;\, 1{,}2]$ $\quad \varepsilon_1 = 0{,}4$ $\quad \varepsilon_2 = 0{,}375$

[2] (a) Given training data $\mathcal{T} = \{(x_i, y_i)\}$ and a set of weak classifiers $\mathcal{H} = \{h_j\}$

1. Initialize data weights $D_i = \frac{1}{|\mathcal{T}|}$ for each data point

2. Select weak classifier with lowest error

$$h^t = \operatorname*{argmin}_{h \in \mathcal{H}} \sum_i D_i [\![h(x_i) \neq y_i]\!]$$

Denote the error $\quad \varepsilon = \sum_i D_i [\![h^t(x_i) \neq y_i]\!]$

3. Terminate if $\varepsilon \geqslant 0{,}5$

4. Compute classifier weight

$$\alpha^t = \frac{1}{2} \ln \frac{1-\varepsilon}{\varepsilon}$$

5. Update data weights

$$D_i \longleftarrow \frac{D_i}{2(1-\varepsilon)} \qquad \text{if } h^t(x_i) = y_i$$

$$D_i \longleftarrow \frac{D_i}{2\varepsilon} \quad \text{if } h^t(x_i) \neq y_i$$

6. Repeat steps 2-5 for specified number of iterations

Resulting classifier $H(x) = \text{sign}\left(\sum_t \alpha^t h^t(x)\right)$

(b) $O(nT)$
where n is number of iterations
T is time to find best weak classifier

(c) $O(nT)$
where n is number of weak classifiers
T is inference time for a weak classifier

(d) Time to find a weak classifier may depend on number of data points, maybe use only a subset of them for finding the best weak classifier (idea like SGD).

Time to classify depends on number of classifiers, maybe use only subset of such found weak classifiers during inference.

(e) Adaboost may combine any other classifiers (that is they are perceived as weak classifiers) in such a way to minimize overall classification error with a guaranteed upper bound.
Data to classify doesn't have to be real numbers for the Adaboost to work unlike in other methods.

This is achieved thanks to a specific choice of weak classifiers.

[3] $\max\limits_{\mu} \prod\limits_i p(x_i)$

$$\max_{\mu} \prod_i \cancel{\underbrace{(2\pi)^{-\frac{d}{2}}}_{>0}} \cancel{\underbrace{\det(C)^{-\frac{1}{2}}}_{>0}} \exp\left(-\frac{1}{2}(x_i-\mu)^T C^{-1}(x_i-\mu)\right)$$

which is equivalent to

$$\max_{\mu} \prod_i \exp\left(-\frac{1}{2}(x_i-\mu)^T C^{-1}(x_i-\mu)\right)$$

Log likelihood

$$l = \sum_i -\frac{1}{2}(x_i-\mu)^T C^{-1}(x_i-\mu) = -\frac{1}{2}\sum_i (x_i-\mu)^T C^{-1}(x_i-\mu)$$

$$\frac{\partial l}{\partial \mu} = -\frac{1}{2}\sum_i 2(x_i-\mu)^T C^{-1}\cdot(-1) = \sum_i (x_i-\mu)^T C^{-1} =$$

$$= \sum_i x_i^T C^{-1} - \sum_i \mu^T C^{-1} = \sum_i x_i^T C^{-1} - N\mu^T C^{-1} = 0$$

$$N\mu^T C^{-1} = \sum_i x_i^T C^{-1} \quad | \; C \text{ from the right}$$

$$N\mu^T = \sum_i x_i^T \longrightarrow \mu = \frac{1}{N}\sum_i x_i$$

[4] Use k-fold cross-validation to find k with the lowest mean classification on the validation fold. Initial range of k to consider might be $[3, \sqrt{N}]$ N number of data points and choose only odd numbers to avoid ties.

January 2020

[1]

(1) Initialize cluster centers $\mathcal{S}_k$

(2) Assign each data point to its nearest cluster

$$\Upsilon_k = \{ x \mid \|x - \mathcal{S}_k\|^2 \leq \|x - \mathcal{S}_j\|^2 \ \forall j \}$$

(3) Update cluster centers

$$\mathcal{S}_k = \begin{cases} \frac{1}{|\Upsilon_k|} \sum_{x \in \Upsilon_k} x & \text{if } \Upsilon_k \neq \emptyset \\ \text{reinitialize} & \text{if } \Upsilon_k = \emptyset \end{cases}$$

(4) Terminate if $\Upsilon_k^{t+1} = \Upsilon_k^t \ \forall k$, otherwise goto 2

Global objective $J = \sum_k \sum_{x \in \Upsilon_k} \|x - \mathcal{S}_k\|^2$

Step 2: each data point's contribution to J either stays the same or becomes lower

Step 3: for a fixed assignment Υ_k, the mean of the cluster is the optimal solution under the least squares objective J

The difference of k-means++ from k-means is that initializes cluster centers in a smart way.

Initialization step:

(1) Choose S_1 uniformly at random from T.
Set $S = \{S_1\}$

(2) For each data point compute its distance from the nearest cluster center

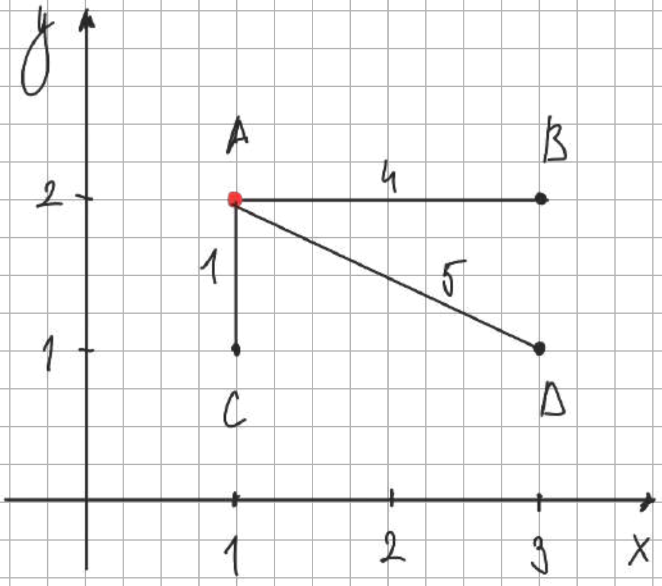
$$d_i = \min_{S_j \in S} \|x_i - S_j\| \quad \forall i$$

(3) Construct probability distribution

$$p(i) = \frac{d_i^2}{\sum_j d_j^2}$$

And sample from it to get the index i

(4) $S \longleftarrow S \cup \{x_i\}$



K=2, assume A is chosen as the first center
If randomly initialized, probability of suboptimal

solution is $\frac{1}{3}$ (if C is chosen as next cluster center)

However, during k-means such probability is lower and is equal to 0,1

Therefore, on average in this example k-means++ leads to a better local minimum.

[2] Assume log-odds can be modeled as a linear function

$$\ln \frac{p(1|x)}{p(-1|x)} = w^T x + b = \begin{bmatrix} w \\ b \end{bmatrix}^T \begin{bmatrix} x \\ 1 \end{bmatrix} := w^T x$$

Then build a classifier $g(x) = \text{sign}(w^T x + b)$
How to find parameters w, b?
Use maximum likelihood estimation.

$$\frac{p(1|x)}{p(-1|x)} = \exp(w^T x)$$

$$\frac{p(1|x)}{1-p(1|x)} = \exp(w^T x)$$

$$\exp(w^T x) - p(1|x)\exp(w^T x) = p(1|x)$$

$$p(1|x)\left[1 + \exp(w^T x)\right] = \exp(w^T x)$$

$$p(1|x) = \frac{\exp(w^T x)}{1+\exp(w^T x)} = \frac{1}{1+\exp(-w^T x)} = \sigma(w^T x)$$

$$p(-1|x) = 1 - p(1|x) = 1 - \frac{\exp(w^T x)}{1+\exp(w^T x)} = \frac{1}{1+\exp(w^T x)} =$$

$$= \sigma(-w^T x)$$

$$p(k|x) = \sigma(k w^T x) \quad k \in \{1, -1\}$$

$$\underset{w}{\operatorname{argmax}} \prod_i p(x_i, k_i) = \underset{w}{\operatorname{argmax}} \prod_i p(k_i|x_i)\, p(x_i) =$$

$$= \underset{w}{\operatorname{argmax}} \sum_i \ln p(k_i|x_i) + \underbrace{\sum_i p(x_i)}_{\text{constant}} =$$

$$= \underset{w}{\operatorname{argmax}} \sum_i \ln \sigma(k_i w^T x_i) =$$

$$= \underset{w}{\operatorname{argmax}} \sum_i \ln \frac{1}{1+\exp(-k_i w^T x_i)} =$$

$$= \underset{w}{\operatorname{argmax}} -\sum_i \ln(1+\exp(-k_i w^T x_i)) =$$

$$= \underset{w}{\operatorname{argmin}} \sum_i \ln(1+\exp(-k_i w^T x_i))$$

$$\frac{\partial l}{\partial w} = \sum_i \frac{1}{1+\exp(-k_i w^T x_i)} \cdot \exp(-k_i w^T x_i) \cdot (-k_i x_i^T) =$$

$$= \sum_i \frac{-k_i}{1+\exp(k_i w^T x_i)} x_i^T$$

$$w \longleftarrow w - \alpha \sum_i \frac{-k_i}{1+\exp(k_i w^T x_i)} x_i =$$

$$= w + \alpha \sum_i \frac{k_i}{1+\exp(k_i w^T x_i)} x_i$$

gradient descent

$$T = \{(-1,-1), (-2,-1), (0,1), (2,1)\}$$

$$w \longleftarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \alpha = 1$$

$$w \longleftarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix} + 1\left(-\frac{1}{2}\begin{bmatrix} -1 \\ 1 \end{bmatrix} - \frac{1}{2}\begin{bmatrix} -2 \\ 1 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 0 \\ 1 \end{bmatrix} + \frac{1}{2}\begin{bmatrix} 2 \\ 1 \end{bmatrix} \right) =$$

bias

$$= \begin{bmatrix} 1/2 \\ -1/2 \end{bmatrix} + \begin{bmatrix} 1 \\ -1/2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1/2 \end{bmatrix} + \begin{bmatrix} 1 \\ 1/2 \end{bmatrix} = \begin{bmatrix} 5/2 \\ 0 \end{bmatrix}$$

After one iteration:

$$w = \begin{bmatrix} 5/2 \\ 0 \end{bmatrix}$$

[3] $\underset{\theta}{\operatorname{argmax}} \prod_i p(t_i) = \underset{\theta}{\operatorname{argmax}} \sum_i \ln p(t_i) =$

$$= \underset{\theta}{\operatorname{argmax}} \sum_i \ln \frac{1}{\theta\left(\frac{t_i}{\theta}+1\right)^2} =$$

$$= \underset{\theta}{\operatorname{argmin}} \sum_i \ln \theta\left(\frac{t_i}{\theta}+1\right)^2 =$$

$$= \underset{\theta}{\operatorname{argmin}} \sum_i \ln \theta + \sum_i \ln\left(\frac{t_i}{\theta}+1\right)^2 =$$

$$= \underset{\theta}{\operatorname{argmin}} N \ln \theta + \sum_i \ln\left(\frac{t_i}{\theta}+1\right)^2$$

$$\frac{\partial \ell}{\partial \theta} = \frac{N}{\theta} + \sum_i \frac{1}{\left(\frac{t_i}{\theta}+1\right)^2} 2\left(\frac{t_i}{\theta}+1\right) \cdot \frac{-t_i}{\theta^2} =$$

$$= \frac{N}{\theta} - \frac{2}{\theta^2}\sum_i \frac{t_i}{\frac{t_i}{\theta}+1} = 0 \quad \Big| \cdot\theta>0$$

$$N - \frac{2}{\theta}\sum_i \frac{t_i}{\frac{t_i}{\theta}+1} = 0$$

For $t_1=1$, $t_2=4$ simplifies to:

$$2 - \frac{2}{\theta}\left(\frac{1}{\frac{1}{\theta}+1} + \frac{4}{\frac{4}{\theta}+1}\right)=0$$

$$2 - 2\left(\frac{1}{1+\theta}+\frac{4}{4+\theta}\right)=0$$

$$\overset{4+\theta}{\frac{1}{1+\theta}} + \overset{1+\theta}{\frac{4}{4+\theta}} = 1$$

$$\frac{4+\theta+4+4\theta}{\theta^2+5\theta+4}=1$$

$$\theta^2+5\cancel{\theta}+4 = 5\cancel{\theta}+8 \longrightarrow \theta^2=4, \ \hat{\theta}=2$$

Bulb is okay at time $T=10$:

$$1-p(10) = 1-\left[1-\frac{1}{\frac{10}{2}+1}\right] = 1-1+\frac{1}{6}=\frac{1}{6}$$

[4] $IG(A) = H(P) - \sum_i \frac{|P_i|}{|P|}H(P_i)$

$\uparrow$ information gain

$$H(P) = -\sum_j \frac{|P^{(j)}|}{|P|}\log_2\frac{|P^{(j)}|}{|P|}$$

where $P^{(j)}$ is number of class j in P

Top-down algorithm

1. Find the "best" attribute in terms of information gain
2. Partition examples based on the attribute's values
3. Apply recursively to each partition

Advantages

- fast inference time
- intuitive, interpretable
- no restrictions on feature space
- to classify not all features are needed
- robust to outliers
- different scales are not a problem
- can save statistics at tree nodes for futher analysis

Disadvantages

- not well justified learning algoritm
- prone to overfitting

January 2019

[1] (a) $\arg\max_{\theta} \prod_i p(t_i) = \arg\max_{\theta} \prod_i \frac{1}{\theta} e^{-\frac{t_i}{\theta}} =$

$$= \arg\max_{\theta} \sum_i \ln \frac{1}{\theta} \exp\left(-\frac{t_i}{\theta}\right) = \arg\max_{\theta} \sum_i \left(\ln\left(\frac{1}{\theta}\right) - \frac{t_i}{\theta}\right) =$$

$$= \arg\max_{\theta} \sum_i - \ln \theta - \sum_i \frac{t_i}{\theta} = \arg\max_{i} -N \ln \theta - \frac{1}{\theta} \sum_i t_i =$$

$$= \arg\min_{\theta} N \ln \theta + \frac{1}{\theta} \sum_i t_i$$

$$\frac{\partial l}{\partial \theta} = \frac{N}{\theta} + \sum_i t_i \cdot \left(\frac{-1}{\theta^2}\right) = 0 \quad \Big| \cdot \theta^2 \neq 0$$

$$N\theta - \sum_i t_i = 0$$

$$N\theta = \sum_i t_i \longrightarrow \hat{\theta} = \frac{1}{N} \sum_i t_i$$

$$\hat{\theta} = \frac{1}{6}\left(0 + 1 + 2{,}3 + 2{,}7 + 6{,}0 + 12{,}0\right) = 4$$

(b) Probability of a bulb failing after $t_{end} = 4$:

$$\int_{t_{end}}^{+\infty} p(t)\,dt = \int_{4}^{+\infty} \frac{1}{\theta} \exp\left(-\frac{t}{\theta}\right) dt = \frac{1}{\theta} \int_{4}^{+\infty} \exp\left(-\frac{t}{\theta}\right) dt =$$

$$= \frac{1}{\theta}\left[-\theta \exp\left(-\frac{t}{\theta}\right)\right]_4^{+\infty} = -\left[\exp\left(-\frac{t}{\theta}\right)\right]_4^{+\infty} =$$

$$= -\left(0 - \exp\left(-\frac{4}{\theta}\right)\right) = \exp\left(-\frac{4}{\theta}\right)$$

$$\underset{\theta}{\arg\max} \prod_i p(t_i) \cdot \exp\left(-\frac{4}{\theta}\right)^2 =$$

$$= \underset{\theta}{\arg\max} \prod_i \frac{1}{\theta} \exp\left(-\frac{t_i}{\theta}\right) \cdot \exp\left(-\frac{4}{\theta}\right)^2 =$$

$$= \underset{\theta}{\arg\max} \; -N\ln\theta - \frac{1}{\theta}\sum_i t_i + 2\left(-\frac{4}{\theta}\right) =$$

$$= \underset{\theta}{\arg\min} \; N\ln\theta + \frac{1}{\theta}\sum_i t_i + \frac{8}{\theta}$$

$$\frac{\partial l}{\partial \theta} = \frac{N}{\theta} + \sum_i t_i\left(-\frac{1}{\theta^2}\right) - \frac{8}{\theta^2} = 0 \;\Big|\; \theta^2 \neq 0$$

$$N\theta - \sum_i t_i - 8 = 0$$

$$\theta = \frac{1}{N}\left(\sum_i t_i + 8\right)$$

$$\hat{\theta} = \frac{1}{4}\left(0 + 1 + 2{,}3 + 2{,}7 + 8\right) = \frac{\cancel{14}^{7}}{\cancel{4}_{2}} = 3{,}5$$

(c) Failing before $t_{end} = 4$:

$$1 - \exp\left(-\frac{4}{\theta}\right)$$

$$\underset{\theta}{\arg\max} \left(1 - \exp\left(-\frac{4}{\theta}\right)\right)^4 \exp\left(-\frac{4}{\theta}\right)^2 =$$

$$= \underset{\theta}{\arg\max} \; 4\ln\left(1 - \exp\left(-\frac{4}{\theta}\right)\right) - \frac{8}{\theta}$$

$$\frac{\partial l}{\partial \theta} = \frac{4}{1-\exp(-\frac{4}{\theta})} \cdot -\exp(-\frac{4}{\theta}) \cdot \left(+\frac{4}{\theta^2}\right) + \frac{8}{\theta^2} = 0 \ \Big|\cdot \theta^2 \neq 0$$

$$\frac{-16}{1-\exp(-\frac{4}{\theta})} \exp(-\frac{4}{\theta}) + 8 = 0$$

$$\frac{-16}{\exp(\frac{4}{\theta})-1} = -8 \ \Big| \cdot \left(-\frac{1}{8}\right)$$

$$\frac{2}{\exp(\frac{4}{\theta})-1} = 1 \ , \ \exp(\frac{4}{\theta}) - 1 = 2$$

$$\exp(\frac{4}{\theta}) = 3 \ \Big| \ln$$

$$\frac{4}{\theta} = \ln 3 \ , \ \hat{\theta} = \frac{4}{\ln 3} \approx 3{,}64$$

EM algorithm:

1. Use initial estimate $\hat{\theta}$ (i.e. $\hat{\theta} = 3{,}5$)
2. Estimate failure times of bulbs failing after $t_{end} = 4$

$$E(T) = \int_0^{+\infty} t\, p(t)\, dt = \int_0^{+\infty} t \frac{1}{\theta} \exp(-\frac{t}{\theta})\, dt =$$

$$= \frac{1}{\theta} \int_0^{+\infty} t \exp(-\frac{t}{\theta})\, dt = \left| \begin{array}{ll} u = t & du = 1 \\ dv = \exp(-\frac{t}{\theta}) & v = -\theta \exp(-\frac{t}{\theta}) \end{array} \right| =$$

pes partes

$$= \frac{1}{\theta}\left[-\theta t \exp\left(-\frac{t}{\theta}\right)\right]_0^{+\infty} - \frac{1}{\theta}\int_0^{+\infty} -\theta \exp\left(-\frac{t}{\theta}\right) dt =$$

$$= -\left[t \exp\left(-\frac{t}{\theta}\right)\right]_0^{+\infty} + \int_0^{+\infty} \exp\left(-\frac{t}{\theta}\right) dt =$$

$$= \underbrace{-\lim_{t\to+\infty} t \exp\left(-\frac{t}{\theta}\right)}_{0} + \underbrace{0\cdot\exp(0)}_{0} + \int_0^{+\infty} \exp\left(-\frac{t}{\theta}\right) dt =$$

$$= \left[-\theta \exp\left(-\frac{t}{\theta}\right)\right]_0^{+\infty} = 0 + \theta\exp(0) = \theta$$

Memoryless property of the exponential distribution:

$$P(T > t_{end} + t \mid T > t_{end}) = P(T > t) \quad \forall t, t_{end} \geq 0$$

Then

$$E(T \mid T > t_{end}) = E(T) + t_{end} = \theta + t_{end}$$

Times of bulbs failining after $t_{end} = 4$:

$$3{,}5 + 4 = 7{,}5$$

3. Update parameter estimate

$$\hat{\theta} = \frac{1}{6}\left(0 + 1 + 2{,}3 + 2{,}7 + 7{,}5 + 7{,}5\right) = 3{,}5$$

4. Repeat steps 2,3 until convergence

In our case, EM algorithm finished after 1 iteration.

[2]

1. Initialize data weights D_i
2. Choose the "best" weak classifier

$$h^t = \arg\min_h \sum_i D_i [\![h(x_i) \neq y_i]\!]$$

$$\varepsilon = \sum_i D_i [\![h^t(x_i) \neq y_i]\!]$$

3. Terminate if $\varepsilon \geq 0{,}5$
4. Compute classifier weight

$$\alpha^t = 2 \ln \frac{1-\varepsilon}{\varepsilon}$$

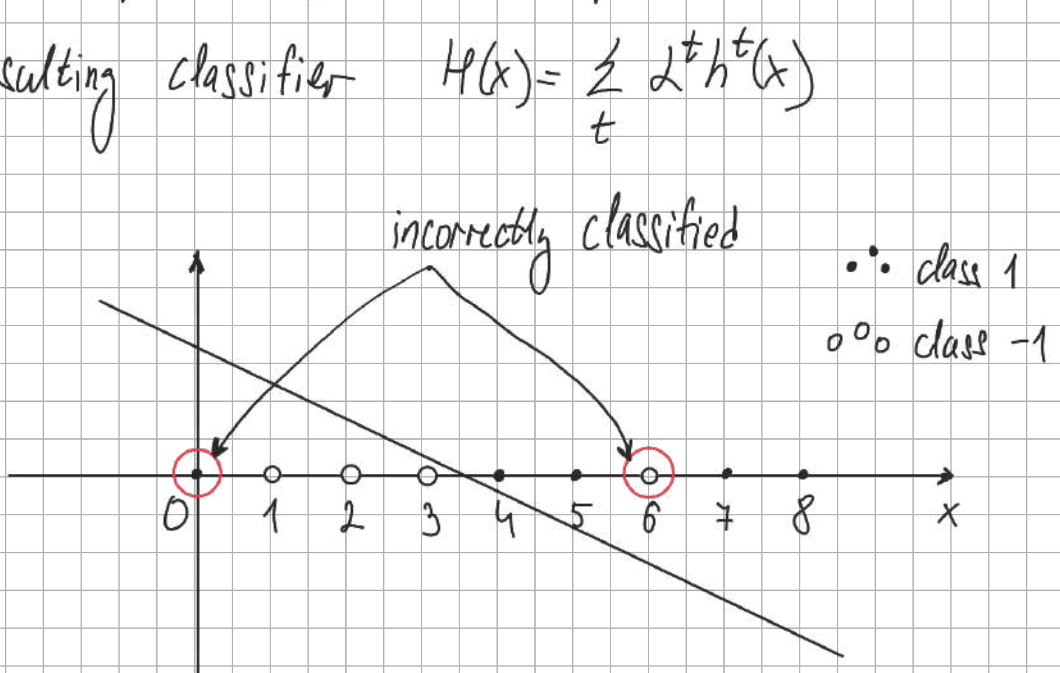
5. Update data weight

$$D_i \longleftarrow \frac{D_i}{2(1-\varepsilon)} \quad \text{if } h^t(x_i) = y_i$$

$$D_i \longleftarrow \frac{D_i}{2\varepsilon} \quad \text{if } h^t(x_i) \neq y_i$$

6. Repeat (2–5) for the specified number of iterations

Resulting classifier $H(x) = \sum_t \alpha^t h^t(x)$



Data weights $D_i = \frac{1}{9}$

One of the best classifiers is

$$h^1(x) = \text{sign}(x-3{,}5)$$

$$\varepsilon = \frac{2}{9}$$

$$\alpha^1 = 2\ln\frac{\frac{7}{9}}{\frac{2}{9}} = 2\ln\frac{7}{2} \approx 2{,}51$$

$$D_i = \frac{1}{2}\cdot\frac{1}{9}\cdot\frac{1}{1-\frac{2}{9}} = \frac{1}{{}^{2}\cancel{18}\cdot\frac{7}{\cancel{9}_1}} = \frac{1}{14}$$

for $i \in \{1,2,3,4,5,7,8\}$ correctly classified

$$D_i = \frac{1}{{}^{2}\cancel{18}\cdot\frac{2}{\cancel{9}_1}} = \frac{1}{4}$$ $i \in \{0,6\}$ incorrectly classified

[3] Save training data $T = \{(x_i, y_i)\}$
To classify new data point, choose K nearest neighbours in terms of distance and predicted label is the most frequent one among chosen neighbours.

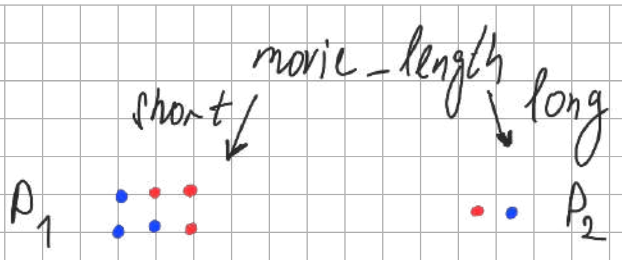
Pros	Cons
• Easy to implement	• Features have to be real numbers
• Provides a baseline	• Hard to choose suitable metric (problem of scales in different axes)
• High memory requirements	

[4] $$IG(A) = H(A) - \sum_i \frac{|A_i|}{|A|} H(A_i)$$

$$H(A) = -\sum_j \frac{|A_j|}{|A|}\log_2\frac{|A_j|}{|A|}$$

$|A_j|$ is a number of j class points in A

Go? true / false

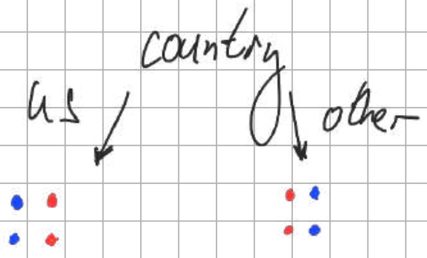


$$H(P) = -2 \cdot \frac{4}{8} \log_2 \frac{4}{8} = -\log_2 \frac{1}{2} = 1$$

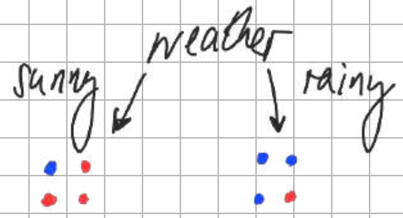
$$H(P_1) = -2 \cdot \frac{3}{6} \log_2 \frac{3}{6} = -\log_2 \frac{1}{2} = 1$$

$$H(P_2) = -2 \cdot \frac{1}{2} \log_2 \frac{1}{2} = 1$$

$$IG(movie\_length) = 1 - \frac{6}{8} \cdot 1 - \frac{2}{8} \cdot 1 = 0$$



$$IG(country) = 0$$



$$H(P_1) = H(P_2) = -\frac{3}{4} \log_2 \frac{3}{4} - \frac{1}{4} \log_2 \frac{1}{4} =$$

$$= -\frac{3}{4}\left(\log_2 3 - \log_2 4\right) - \frac{1}{4} \cdot (-2) =$$

$$= -\frac{3}{4}\left(\log_2 3 - 2\right) + \frac{1}{2} = -\frac{3}{4} \log_2 3 + \frac{3}{2} + \frac{1}{2} = 2 - \frac{3}{4} \log_2 3 \approx$$

$$\approx 0{,}811$$

$$IG(weather) \approx 0{,}189$$

January 2018

[1] (a) $p(x|k)$ are known
$p(k)$ are unknown

$$\varepsilon_k = \sum_{x | q(x) \neq k} p(x|k)$$

$k = \{1, 2\}$ where 2 is "dangerous" class

$$\min_q \varepsilon_1$$

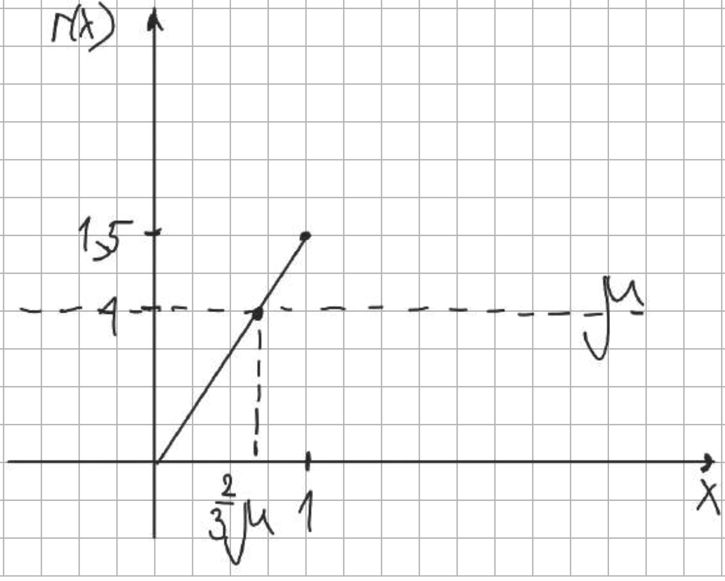
subject to $\varepsilon_2 \leq \bar{\varepsilon}_2$ where $\bar{\varepsilon}_2$ is given

(b) Construct likelihood ratio

$$r(x) = \frac{p(x|2)}{p(x|1)} = \frac{3x^2}{2x} = 1{,}5x \quad x \in [0, 1]$$

Find (somehow) threshold μ, then

$$q^*(x) = \begin{cases} 2 & \text{if } r(x) \geq \mu \\ 1 & \text{otherwise} \end{cases}$$



Solve $\min\limits_{q} \varepsilon_2$

subject to $\varepsilon_1 \leq 0{,}1$

Minimization of ε_2 is achieved at $\varepsilon_1 = 0{,}1$

$$\varepsilon_1 = \int_{\frac{2}{3}\mu}^{1} p(x|1)\,dx = \int_{\frac{2}{3}\mu}^{1} 2x\,dx = \left[x^2\right]_{\frac{2}{3}\mu}^{1} =$$

$$= 1 - \frac{4}{9}\mu^2 = 0{,}1$$

Then $\frac{4}{9}\mu^2 = \frac{9}{10}$, $\mu^2 = \frac{9}{10} \cdot \frac{9}{4} = \frac{81}{40}$

$$\mu = \sqrt{\frac{81}{40}} = \frac{9}{2\sqrt{10}}$$

$$q^*(x) = \begin{cases} 2 & \text{if } 1{,}5x \geq \frac{9}{2\sqrt{10}} \\ 1 & \text{otherwise} \end{cases}$$

$$q^*(x) = \begin{cases} 2 & \text{if } x \geq \frac{3}{\sqrt{10}} \\ 1 & \text{otherwise} \end{cases}$$

[2] Already discussed multiple times

[3] (a) Already done

(b) Using EM algorithm

$\hat{\theta} = 4$

$E(T) = \theta$

$E(T \mid T > t_{end}) = E(T) + t_{end}$ - memoryless property of the exponential distribution

Estimate time of failed bulbs as

$$E(T \mid T > t_{end}) = E(T) + t_{end} = \hat{\theta} + 4 = 8$$

New estimate of the parameter θ is

$$\hat{\theta} = \frac{1}{6}(0 + 1 + 2{,}3 + 2{,}7 + 8 + 8) \approx 3{,}7$$

New estimate of time of failed bulbs is

$$\hat{\theta} + 4 = 7{,}7$$

New estimate of the parameter

$$\hat{\theta} = \frac{1}{6}(0 + 1 + 2{,}3 + 2{,}7 + 7{,}7 + 7{,}7) \approx 3{,}57$$

Iterate all over again

$$\hat{\theta} + 4 = 7{,}57$$

$$\hat{\theta} \approx 3{,}52$$

$$\vdots$$

Eventually should converge to

$$\hat{\theta} = 3{,}5$$

[4] Already done

January 2017

[1] Already discussed

[2] Given labeled data $T = \{(x_i, y_i)\}$, try to find such plane, that maximizes margin between data points (while classifying them correctly).

$$x := \begin{bmatrix} x \\ 1 \end{bmatrix}$$

$$w := \begin{bmatrix} w \\ b \end{bmatrix}$$ (includes bias)

$$\max_{w} \frac{2\varepsilon}{\|w\|} \longleftarrow \text{margin}$$

subject to $y_i w^T x_i \geqslant \varepsilon \quad \forall i, \; \varepsilon > 0$

correct classification

Pick $\varepsilon = 1$, then

$$\max_{w} \frac{2}{\|w\|}$$

subject to $y_i w^T x_i \geqslant 1 \quad \forall i$

Which is equivalent to solving

$$\min_{w} \|w\|^2$$

subject $y_i w^T x_i \geqslant 1 \quad \forall i$

This can be solved by solving the dual problem using quadratic programming.

Pros	Cons
Provides optimal solution (if it exists)	Features have to be real numbers
Fast inference time (thanks to support vectors)	
With kernel trick may transform given data to high-dimensional space	
Good generalization	

Here may be comparison to other classification methods someday

[3] (a) 1. Initialize cluster centers c_K

2. Assign each data point to its nearest cluster

$$\Gamma_K = \{ x \in \Gamma \mid \|x - c_K\| \le \|x - c_j\| \quad \forall j \}$$

generic metric

3. Update cluster centers

$$c_K = \underset{c}{\arg\min} \sum_{x \in \Gamma_K} \|x - c\|$$

generic metric

4. Terminate if $\Gamma_K^t = \Gamma_K^{t+1} \quad \forall K$

(b) $$c_K = \underset{c}{\arg\min} \sum_{x \in \Gamma_K} \|x - c\|_1$$

The solution with L_1 norm is the median of Γ_K

(c) $c_k = \underset{c}{\operatorname{argmin}} \sum_{x \in \Gamma_k} \|x - c\|_2$

The solution with L_2 norm is the geometric median of Γ_k, which has no closed-form solution.

Weiszfeld algorithm

$$\underset{m}{\operatorname{argmin}} \sum_i \|x_i - m\|_2$$

1. Initialize m
2. Compute weights w_i

$$w_i = \frac{1}{\|x_i - m\|}$$

3. Obtain new estimate as weighted sum of points

$$m \longleftarrow \frac{\sum_i w_i x_i}{\sum_i w_i}$$

4. Finish if termination criteria is met.

(d) Each iteration requires $O(Nk)$ distance computations, where N is the number of points, k is the number of clusters.

Some generalizations might be useful in certain real-world applications.

(e) Already discussed many times

[4] (a) $\underset{\theta}{\operatorname{argmax}}\ p(T|\theta) = \underset{\theta}{\operatorname{argmax}} \prod_i p(t_i) =$

$= \underset{\theta}{\operatorname{argmax}} \prod_i \theta \exp(-\theta t_i) = \underset{\theta}{\operatorname{argmax}} \sum_i \ln\theta - \theta \sum_i t_i =$

$= \underset{\theta}{\operatorname{argmax}}\ N \ln\theta - \theta \sum_i t_i$

$$\frac{\partial l}{\partial \theta} = \frac{N}{\theta} - \sum_i t_i = 0 \longrightarrow \hat{\theta} = \frac{N}{\sum_i t_i}$$

Then $\hat{\theta} = \dfrac{3}{56+120+424} = 0{,}005$

(b) Probability of a hard drive failing after $t=300$ is

$$\int_{300}^{+\infty} p(t)\,dt = \int_{300}^{+\infty} \theta e^{-\theta t}\,dt = \theta\left[e^{-\theta t}\cdot\left(-\frac{1}{\theta}\right)\right]_{300}^{+\infty} =$$

$$= -\left[e^{-\theta t}\right]_{300}^{+\infty} = -\left(0 - e^{-300\theta}\right) = \exp(-300\theta)$$

$\underset{\theta}{\operatorname{argmax}} \prod_i \theta \exp(-\theta t_i) \cdot \exp(-300\theta) =$

$= \underset{\theta}{\operatorname{argmax}} \sum_i \ln\theta - \theta \sum_i t_i - 300\theta =$

$= \underset{\theta}{\operatorname{argmax}}\ N \ln\theta - \theta \sum_i t_i - 300\theta$

$$\frac{\partial l}{\partial \theta} = \frac{N}{\theta} - \sum_i t_i - 300 = 0$$

$$\hat{\theta} = \frac{N}{\sum_i t_i + 300} \qquad \hat{\theta} = \frac{2}{(56+120)+300} \approx 0{,}004$$