Machine Learning (ML)

- Думать как человек (Cognitive Science , Computational Neurobiology)
- Думать рационально (мы не рациональны)
- Действовать как человек (Turing Test "Imitation game" (1950)

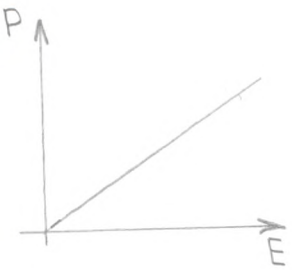
- Действовать рационально - оптимальное

Machine Learning

E (experience)

T (tasks)

P (proficiency)



Stastical approaches : estimation - learning

- Supervised Learning : Regression ($y(\vec{x}) \to R^c$), Classification ($y(\vec{x}) \to \vec{P}_{class}$)
- Unsupervised Learning : Clusterization
- Reinforcement Learning : Agent $\rightleftharpoons$ Environment

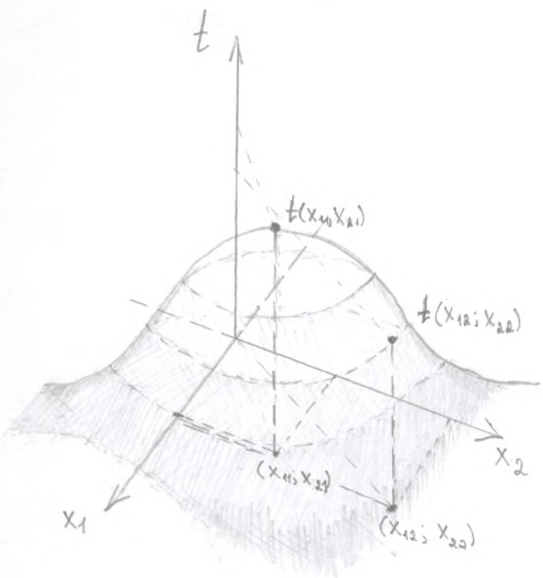
Supervised Learning

- Regression ($\bar{X} \to R$)
- Classification ($\bar{X} \to Class$)

Data

Object features - $\bar{x} \in R^D$

Training sample	Target values
$\bar{X}_1$	$\bar{t}_1$
$\bar{X}_2$	$\bar{t}_2$
$\bar{X}_3$	$\bar{t}_3$
...	...
$\bar{X}_N$	$\bar{t}_N$

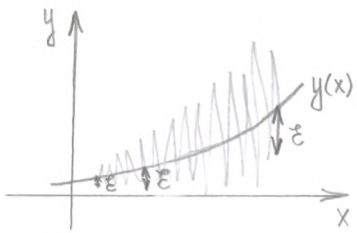


Notation
- y - target
- $\hat{y}$ - predicted

Linear Regression

$$t(\bar{x}) = y(\bar{x}, \bar{w}) + \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

$\sigma \neq P(\bar{x}) \Rightarrow$ const

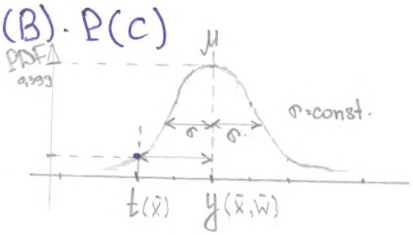


$$y(\bar{x}, \bar{w}) = w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_D x_D = w_0 + \sum_{j=1}^{D} w_j x_j = \bar{w}^T \bar{x}, \quad x_0 = 1$$

Likelihood

Общая условная плотность вероятностей $\bar{t}$ при $X, \vec{w}, \sigma^2$

Events are Independent: $P(A \cap B \cap C) = P(A) \cdot P(B) \cdot P(C)$



$$p(\vec{t} \mid X, \vec{w}, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid y(\bar{x}_n, \bar{w}), \sigma^2) \to \max_x$$

$$p(\vec{t} \mid X, \vec{w}, \sigma^2) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{t_n - y(\bar{x}_n, \bar{w})}{\sigma}\right)^2} \to \max_x$$

$$\ln p(\vec{t} \mid X, \vec{w}, \sigma^2) = \ln \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{t_n - y(\bar{x}_n, \bar{w})}{\sigma}\right)^2} \to \max_x$$

$f(x) \to \max$

$\arg\max_x f(x) = \arg\max_x \log(f(x))$

$$= \sum_{n=1}^{N} \ln \frac{1}{\sigma\sqrt{2\pi}} + \ln \sigma$$

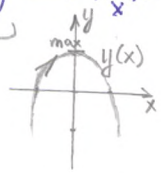
$$\ln p(\vec{t} \mid X, \vec{w}, \sigma^2) = \sum_{n=1}^{N} \ln \frac{1}{\sigma\sqrt{2\pi}} + \sum_{n=1}^{N} \ln e^{-\frac{1}{2}\left(\frac{t_n - y(\bar{x}_n, \bar{w})}{\sigma}\right)^2} =$$

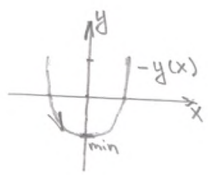
$\log(xy) = \log(x) + \log(y)$

$$= \sum_{n=1}^{N} \ln \frac{1}{\sigma\sqrt{2\pi}} + \sum_{n=1}^{N} -\frac{1}{2}\left(\frac{t_n - y(\bar{x}_n, \bar{w})}{\sigma}\right)^2 =$$

$$= \sum_{n=1}^{N} \ln \frac{1}{\sigma\sqrt{2\pi}} + \frac{1}{\sigma^2} \cdot \left(-\sum_{n=1}^{N} \frac{1}{2}\left(t_n - y(\bar{x}_n, \bar{w})\right)^2\right) \to \max_x$$

Loss Function





Loss Function

$$E(\vec{w}) = \sum_{n=1}^{N} \frac{1}{2}\left(t_n - y(\bar{x}_n, \bar{w})\right)^2 \to \min_x$$ — Mean Squared Error (MSE)

Optimization problem

Adding Non-linearity to Linear Regression

$$t_i(\bar{x}_i) = y(\bar{x}_i, \bar{w}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2)$$

$$y(\bar{x}_i, \bar{w}) = \vec{w}^T \vec{x} = \sum_j^D x_j w_j = w_0 + x_1 w_1 + x_2 w_2 + \dots + x_D w_D$$

$\bar{x}_i \in R^D$ – sample

$X^T \in R^{D \times N}$

$$\bar{y}(X, \bar{w}) = X^{N \times D} \cdot \bar{w}^{D \times 1} = \begin{bmatrix} \vdots \end{bmatrix}^{N \times 1}$$

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1D} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2D} \\ \dots & & & & \\ x_{N1} & x_{N2} & x_{N3} & \dots & x_{ND} \end{bmatrix}^{N \times D}$$

rows – samples
cols – features

Basis Function

$$\vec{\varphi}(\vec{x}_i) = [\vec{\varphi}_0(\bar{x}_i) \ \vec{\varphi}_1(\bar{x}_i) \ \vec{\varphi}_2(\bar{x}_i) \dots \vec{\varphi}_{M-1}(\bar{x}_i)]^T$$

$\vec{\varphi} \in R^M$

$$y(\vec{x}_i, \vec{w}) = \vec{w}^T \cdot \vec{\varphi}(\vec{x}_i) = w_0 + \sum_{j=1}^{M-1} w_j \cdot \varphi_j(\vec{x}_i), \quad \varphi_0(\vec{x}) = 1$$

Polynomial Basis Functions:

$$[\underbrace{x_1 \ x_2 \dots x_D}_{\vec{\varphi}_1} \ \underbrace{x_1^2 \ x_2^2 \dots x_D^2}_{\vec{\varphi}_2} \ \underbrace{x_1^3 \ x_2^3 \dots x_D^3}_{\vec{\varphi}_3}]^T$$

Design Matrix

$$\Phi(X) = [\bar{\varphi}^T(\bar{x}_1) \ \bar{\varphi}^T(\bar{x}_2) \dots \bar{\varphi}^T(\bar{x}_N)]^T$$

$$\Phi(X) = \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_1(\bar{x}_1) & \dots & \varphi_{M-1}(\bar{x}_1) \\ \varphi_0(\bar{x}_2) & \varphi_1(\bar{x}_2) & \dots & \varphi_{M-1}(\bar{x}_2) \\ \dots & & & \\ \varphi_0(\bar{x}_N) & \varphi_1(\bar{x}_N) & \dots & \varphi_{M-1}(\bar{x}_N) \end{bmatrix}^{N \times M}$$

$$\bar{y}(X, \bar{w}) = \Phi(X)^{N \times M} \cdot \bar{w}^{M \times 1} = \begin{bmatrix} \vdots \end{bmatrix}^{N \times 1}$$

$\bar{y}$ нелинейна относительно X.

Linear Regression Optimization.
Normal Equation. Analytical Solution.

$t(\bar{x}) = y(\bar{x}, \bar{w}) + \varepsilon$; $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ $\quad \sigma \neq f(\bar{x})$

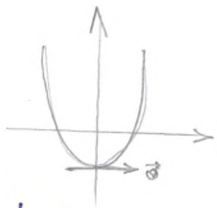
$\bar{x}_i \in R^D$, $\quad X \in R^{N \times D}$, $\quad \bar{\varphi}(\bar{x}_i) \in R^M$, $\quad \Phi(X) \in R^{N \times M}$

$$y(\bar{x}, \bar{w}) = \bar{w}^T \bar{\varphi}(\bar{x}) = \sum_{i=0}^{M-1} w_i \varphi_i(\bar{x})$$

$$p(\vec{t} \mid X, \vec{w}, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid \underbrace{y(\bar{x}_n, \bar{w})}_{\text{mean}}, \underbrace{\sigma^2}_{\text{std}}) \to \max_w$$

Loss Function

$$E(\bar{w}) = \sum_{n=1}^{N} \frac{1}{2} (t_n - y(\bar{x}_n, \bar{w}))^2 \to \min_w$$



In local minima gradient is equal to zero vector:

$$\nabla E(\bar{w}) = \vec{0}$$

Gradient is vector of partial derivatives : $\nabla y(\bar{x}) = \frac{dy}{dx_1}\vec{i} + \frac{dy}{dx_2}\vec{j} + \frac{dy}{dx_3}\vec{k}$

$$\frac{dE(\bar{w})}{d w_i} = \left(\sum_{n=1}^{N} \frac{1}{2} (t_n - y(\bar{x}_n, \bar{w}))^2 \right)' = \sum_{n=1}^{N} 2 \cdot \frac{1}{2} (t_n - y(\bar{x}_n, \bar{w})) \cdot (t_n - y(\bar{x}_n, \bar{w}))'$$

$(f(x)+g(x))' = f'(x) + g'(x)$
$(f(g(x)))' = f'(g(x)) \cdot g'(x)$
$(x^n)' = n x^{n-1}$
$(C)' = 0$
$(Cx)' = C$

$$y(\bar{x}_n, \bar{w}) = \sum_{j=0}^{M-1} w_j \cdot \varphi_j(\bar{x}_n) \; ; \quad y'_{w_i}(\bar{x}_n, \bar{w}) = 0 + \varphi_i(\bar{x}_n)$$

$$\frac{dE(\bar{w})}{d w_i} = -\sum_{n=1}^{N} (t_n - y(\bar{x}_n, \bar{w})) \cdot \varphi_i(\bar{x}_n) \text{ - scalar}$$

$$\nabla E(\bar{w}) = -\sum_{n}^{N} (t_n - y(\bar{x}_n, w)) \cdot \bar{\varphi}^T(\bar{x}_n) = \left[\frac{dE(\bar{w})}{dw_0} \; \frac{dE(\bar{w})}{dw_1} \; \frac{dE(\bar{w})}{dw_2} \; \dots \; \frac{dE(\bar{w})}{dw_{M-1}} \right]^T$$

vector of partial derivatives

$$\vec{0}^T = -\sum_{n=1}^{N} (t_n - y(\bar{x}_n, \bar{w})) \cdot \bar{\varphi}^T(\bar{x}_n) =$$

$$= -\sum_{n=1}^{N} t_n \bar{\varphi}^T(\bar{x}_n) + \sum_{n=1}^{N} y(\bar{x}_n, w) \bar{\varphi}^T(\bar{x}_n) = -\sum_{n=1}^{N} t_n \bar{\varphi}^T(\bar{x}_n) + \sum_{n=1}^{N} \underbrace{\bar{w}^T \bar{\varphi}(\bar{x}_n)}_{1\times M \cdot M\times 1 \cdot 1\times M} \bar{\varphi}^T(\bar{x}_n) =$$

$$= \underbrace{-\sum_{n=1}^{N} t_n \bar{\varphi}^T(\bar{x}_n)}_{1} + \bar{w}^T \underbrace{\sum_{n=1}^{N} \bar{\varphi}(\bar{x}_n) \bar{\varphi}^T(\bar{x}_n)}_{2} =$$

$$= -\bar{t}^T \Phi + \bar{w}^T \Phi^T \Phi = \vec{0}^T$$

$$(AB)^T = B^T A^T$$
$$(A^{-1})^T = (A^T)^{-1}$$

$$\bar{w}^T \Phi^T \Phi = \bar{t}^T \Phi$$

$$\bar{w}^T = \bar{t}^T \Phi (\Phi^T \Phi)^{-1}$$

$$\bar{w} = (\vec{t}^T \Phi (\Phi^T \Phi)^{-1})^T = ((\Phi^T \Phi)^{-1})^T \cdot (\vec{t}^T \Phi)^T = ((\Phi^T \Phi)^T)^{-1} \cdot (\Phi^T \bar{t})$$

$$\vec{w} = \underbrace{(\Phi^T \Phi)^{-1} \Phi^T}_{\text{Moore-Penrose Pseudoinverse}} \vec{t}$$

- $\Phi^{T\,M\times N} \cdot \Phi^{N\times M} = A^{M\times M}$ - depends on number of features of $\vec{x}$
- $A^{-1} = ?$
- $\Rightarrow$ Gradient Descent

(Linear Regression Optimization)

$$\vec{t} = \begin{bmatrix} t_1 \\ t_2 \\ \dots \\ t_n \end{bmatrix}$$

$$\bar{\varphi}(\bar{x}_i) = \begin{bmatrix} \varphi_0(\bar{x}_i) \\ \varphi_1(\bar{x}_i) \\ \varphi_2(\bar{x}_i) \\ \dots \\ \varphi_{M-1}(\bar{x}_i) \end{bmatrix} \qquad \Phi(X) = \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_1(\bar{x}_1) \dots \varphi_{M-1}(\bar{x}_1) \\ \varphi_0(\bar{x}_2) & \varphi_1(\bar{x}_2) \dots \varphi_{M-1}(\bar{x}_2) \\ \dots & \\ \varphi_0(\bar{x}_N) & \varphi_1(\bar{x}_N) \dots \varphi_{M-1}(\bar{x}_N) \end{bmatrix}^{N \times M}$$

$$\sum_{n=1}^{N} t_n \bar{\varphi}^T(\bar{x}_n) = \begin{matrix} [t_n\varphi_0(x_n) & t_n\varphi_1(x_n) & \dots] \\ + & + & \\ [t_{n+1}\varphi_0(x_{n+1}) & t_{n+1}\varphi_1(x_{n+1}) & \dots] \\ \dots & & \end{matrix} = [t_n\varphi_0(x_n) + t_{n+1}\varphi_0(x_{n+1}) \qquad t_n\varphi_1(x_n) + t_{n+1}(\varphi_1(x_{n+1}))]$$

1) $$\vec{t}^T \cdot \Phi^*(X) = \underbrace{[t_1 \; t_2 \dots t_n]}_{1 \times N} \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_1(\bar{x}_1) \\ \varphi_0(\bar{x}_2) & \varphi_1(\bar{x}_2) \\ \varphi_0(\bar{x}_N) & \varphi_1(\bar{x}_N) \dots \end{bmatrix}^{N \times M} = \left[\sum_{n=1}^{N} t_n \varphi_0(\bar{x}_n) \quad \sum_{n=1}^{N} t_n \varphi_1(\bar{x}_n) \; \dots \right]$$

2) $$\sum_{n=1}^{N} \underbrace{\bar{\varphi}(\bar{x}_n)}_{M \times 1} \underbrace{\bar{\varphi}^T(\bar{x}_n)}_{1 \times M} = \sum_{n=1}^{N} \begin{bmatrix} \varphi_0(\bar{x}_n) \\ \varphi_1(\bar{x}_n) \\ \varphi_{M-1}(\bar{x}_n) \end{bmatrix} [\varphi_0(\bar{x}_n) \; \varphi_1(\bar{x}_n) \; \varphi_{M-1}(\bar{x}_n)] = \sum_{n=1}^{N} \begin{bmatrix} \varphi_0\varphi_0 & \varphi_0\varphi_1 \dots \varphi_0\varphi_{M-1} \\ \varphi_1\varphi_0 & \varphi_1\varphi_1 \dots \varphi_1\varphi_{M-1} \\ \varphi_{M-1}\varphi_0 & \varphi_{M-1}\varphi_1 \dots \varphi_{M-1}\varphi_{M-1} \end{bmatrix}_n =$$

$$\begin{bmatrix} \varphi_0\varphi_0 & \varphi_0\varphi_1 \dots \varphi_0\varphi_{M-1} \\ \varphi_1\varphi_0 & \varphi_1\varphi_1 \dots \varphi_1\varphi_{M-1} \\ \dots & \\ \varphi_{M-1}\varphi_0 & \varphi_{M-1}\varphi_1 \dots \varphi_{M-1}\varphi_{M-1} \end{bmatrix}_n$$

$$= \Phi^T \Phi$$

$M \times N \cdot N \times M = M \times M$

$$\Phi^T\Phi = \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_0(\bar{x}_2) \dots \varphi_0(\bar{x}_N) \\ \varphi_1(\bar{x}_1) & \varphi_1(\bar{x}_2) \dots \varphi_1(\bar{x}_N) \\ \dots & \\ \varphi_{M-1}(\bar{x}_1) & \varphi_{M-1}(\bar{x}_2) \dots \varphi_{M-1}(\bar{x}_N) \end{bmatrix} \cdot \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_1(\bar{x}_1) \dots \varphi_{M-1}(\bar{x}_1) \\ \varphi_0(\bar{x}_2) & \varphi_1(\bar{x}_2) \dots \varphi_{M-1}(\bar{x}_2) \\ \dots & \\ \varphi_0(\bar{x}_N) & \varphi_1(\bar{x}_N) \dots \varphi_{M-1}(\bar{x}_N) \end{bmatrix}$$

Weight Decay Regularization

$t(\bar{x}) = y(\bar{x}, \bar{w}) + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, $\sigma^2 \neq f(\bar{x})$

$p(\bar{t} \mid X, \bar{w}, \sigma^2) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid y(\bar{x}_n, \bar{w}), \sigma^2) \to \max_w$ — общая условная плотность (product) вероятностей $\bar{t}$ при X, $\bar{w}$ и σ^2.

$y(\bar{x}_n, \bar{w}) = \bar{w}^T \cdot \bar{\varphi}(\bar{x}_n)$

$$E(\bar{w}) = \frac{1}{2} \sum_{n=1}^{N} (t_n - \bar{w}^T \bar{\varphi}(\bar{x}_n))^2 + \frac{\lambda}{2} \sum_{j=1}^{M-1} |w_j|^q \longrightarrow \min_w$$

- w_0 - bias, его нежелательно включать в параметр регуляризации.
- standard values for q are 1 and 2.

Регуляризация нужна для более менее мягкого распределения влияния параметров на модель.

$$E(\bar{w}) = \frac{1}{2} \sum_{n=1}^{N} (t_n - \bar{w}^T \bar{\varphi}(\bar{x}))^2 + \frac{\lambda}{2} \bar{w}^T \bar{w} \to \min$$

$$\frac{d\left(\frac{\lambda}{2} \sum_{i=1}^{M-1} w_i^2\right)}{d(w_j)} = 2 \cdot \frac{\lambda}{2} \cdot (0 + \dots + w_j) = \lambda w_j \Rightarrow \nabla = \begin{bmatrix} \lambda w_1 \\ \lambda w_2 \\ \dots \\ \lambda w_{M-1} \end{bmatrix} = \lambda \bar{w}$$

$$\nabla E(\bar{w}) = -\sum_{n=1}^{N} (t_n - \bar{w}^T \bar{\varphi}(\bar{x}_n)) \bar{\varphi}(\bar{x}_n) + \lambda \bar{w} \to \min$$

$$\bar{0}^T = \nabla E(\bar{w})^T = -t^T \Phi + \bar{w}^T \Phi^T \Phi + \underbrace{\lambda \bar{w}^T}_{1 \times M \quad M \times M} = -\bar{t}^T \Phi + \bar{w}^T (\Phi^T \Phi + \lambda I)$$

$$\bar{w}^T = \vec{t}^T \Phi (\Phi^T \Phi + \lambda I)^{-1}$$

$$\bar{w} = (\vec{t}^T \Phi (\Phi^T \Phi + \lambda I)^{-1})^T = ((\Phi^T \Phi + \lambda I)^{-1})^T \cdot (\vec{t}^T \Phi)^T = \underbrace{(\Phi^T \Phi + \lambda I)^{-1} \cdot (\Phi^T \vec{t})}_{\text{Normal Equation}}$$

$$\nabla E(\bar{w}) = -\sum_{n=1}^{N} \left(t_n - \bar{w}^T \bar{\varphi}(\bar{x}_n)\right) \bar{\varphi}(\bar{x}_n) + \lambda \bar{w}$$

$$\bar{0} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ \dots \end{bmatrix}$$

$n \times 1 \quad n \times 1$

$$\bar{0} = \nabla E(\bar{w}) = -\sum_{n=1}^{N} \left(t_n \vec{\varphi}(\vec{x}_n) - \underbrace{\left(\bar{w}^T \bar{\varphi}(\bar{x}_n)\right) \bar{\varphi}(\bar{x}_n)}_{S} \right) + \lambda \bar{w} =$$

$$= -\varphi^T I +$$

Therefore we should transpose

$$\begin{bmatrix} \varphi_0(x_1) & \varphi_0(x_2) \\ \varphi_1(x_1) & \varphi_1(x_2) \\ \varphi_2(x_1) & \varphi_2(x_2) \\ \varphi_3(x_1) & \varphi_3(x_2) \end{bmatrix} \begin{bmatrix} \varphi_0(x_1) & \varphi_1(x_1) & \varphi \end{bmatrix}$$

$$\sum_{n=1}^{N} \bar{\varphi}(\bar{x}_n)\, \varphi^T(\bar{x}_n) = \left\{ \sum_{n=1}^{N} \begin{bmatrix} \varphi_0(x_n) \\ \varphi_1(x_n) \\ \varphi_2(x_n) \end{bmatrix}^{3\times 1} \cdot \begin{bmatrix} \varphi_0(x_n) & \varphi_1(x_n) & \varphi_2(x_n) \end{bmatrix}^{1\times 3} = \begin{bmatrix} \dots \end{bmatrix}^{3\times 3} \right\} =$$

$$= \sum_{n=1}^{N} \begin{bmatrix} \varphi_0\varphi_0 & \varphi_0\varphi_1 & \varphi_0\varphi_2 \\ \varphi_1\varphi_0 & \varphi_1\varphi_1 & \varphi_1\varphi_2 \\ \varphi_2\varphi_0 & \varphi_2\varphi_1 & \varphi_2\varphi_2 \end{bmatrix}$$

$$\underline{\varphi}(\underline{X}) = \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_1(\bar{x}_1) & \varphi_2(\bar{x}_1) \\ \varphi_0(\bar{x}_2) & \varphi_1(\bar{x}_2) & \varphi_2(\bar{x}_2) \\ \varphi_0(\bar{x}_3) & \varphi_1(\bar{x}_3) & \varphi_2(\bar{x}_3) \end{bmatrix}^{N\times M}$$

$$\varphi^T(\underline{X}) = \begin{bmatrix} \varphi_0(\bar{x}_1) & \varphi_0(\bar{x}_2) & \varphi_0(\bar{x}_3) \\ \varphi_1(\bar{x}_1) & \varphi_1(\bar{x}_2) & \varphi_1(\bar{x}_3) \\ \varphi_2(\bar{x}_1) & \varphi_2(\bar{x}_2) & \varphi_2(\bar{x}_3) \\ \dots & \dots & \varphi_{M-1}(\bar{x}_N) \end{bmatrix}^{M\times N}$$

$$\underline{\varphi}^T \underline{\varphi} = \begin{bmatrix} \sum_{n=1}^{N} \bar{\varphi}_0^T(\bar{x}_n)\bar{\varphi}_0(\bar{x}_n) & \sum_{n=1}^{N} \bar{\varphi}_0^T(\bar{x}_n)\bar{\varphi}_1(\bar{x}_n) & \dots \\ \sum_{n=1}^{N} \bar{\varphi}_1^T(\bar{x}_n)\bar{\varphi}_0(\bar{x}_n) & \sum_{n=1}^{N} \bar{\varphi}_1^T(\bar{x}_n)\bar{\varphi}_1^T(\bar{x}_n) & \dots \end{bmatrix}^{M\times M}$$

$$\underbrace{w^T}_{1\times M} \underbrace{\varphi^T \varphi}_{M\times M} = \begin{bmatrix} w_1\Sigma + w_2\Sigma + w_3\Sigma & w_1\Sigma + w_2\Sigma + w_3\Sigma & \dots \end{bmatrix}$$

$$w^T \varphi^T \varphi + \lambda w^T = \begin{bmatrix} w_1\Sigma + w_2\Sigma + w_3\Sigma + w_i & \dots \end{bmatrix}$$

Методология решения задачи регрессии

process → Data → Analyze → Estimate (Learning)

estimate unknown param-s of process generating data.

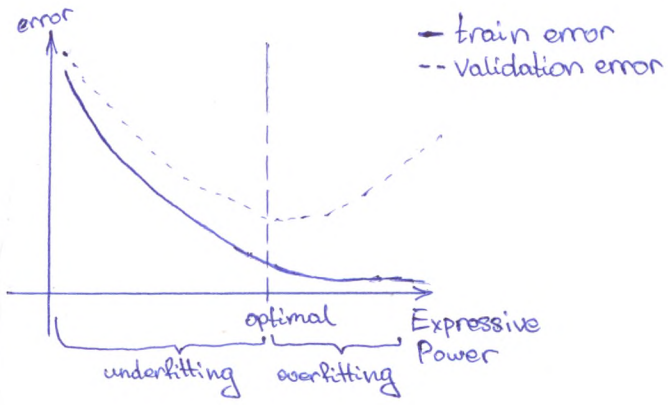
– Мы не знаем закономерности.

- Обычное изучение данных:
  - Какие данные у нас есть;
  - За какой период времени;
  - Сколько данных;
  - Среднее (μ);
  - Разброс (σ^2);

Обучение:

Train	Validation	Test
60%	20%	20%

1) Try Linear Regression $\bar{w}^T \bar{x} = \hat{y}$
2) Try other basis functions $\bar{w}^T \bar{\varphi}(\bar{x})$
   Test on Validation dataset
3) Underfitting vs. Overfitting
4) Regularization
5) Перебираем гиперпараметры: φ_i, λ
6) Модель с наименьшей ошибкой на валидационной выборке отбираем.
7) Test on test dataset. – честно.



- Пример почему train, validation, test

$10^{1000...}$ моделей

модели рандомные генераторы, среди них можно найти новые комбинации, и можно подобрать 100% accuracy на validation, но на тестовой выборке будет ~50% acc.

Classification

$Class = \{0, 1, 2, \dots, K\}$

u - object (cat, patient ...)

$\bar{x} = \mu(u)$ - features of object (image; temperature, pain...)

$\bar{x} \in R^D$, $\bar{x} = [x_1, x_2, \dots, x_D]^T$

- **Binary Classification ($K = 2$)**

$y(\bar{x}, \bar{w}) \to Class$ or $P(Class)$ — t

- **Multiclass Classification ($K > 2$)**

$\bar{y}(\bar{x}, \bar{w}) = [p_1 \; p_2 \cdots p_K]^T$ — $\vec{t}$ - one-hot encoding

Probability vector

$\sum_{k=1}^{K} p_k = 1$, $p_k \geq 0$, $k = 1, 2, \dots K$

p_k - *вероятность принадлежности объекта к классу k.*

Confusion Matrix

$C = \{c_{ij}\}_{K \times K}$

$c_{ij} = |\{u \in C_i \mid y(\bar{x}(u), \bar{w}) = j\}|$

t \ y	0	1
0	TN	FP
1	FN	TP

В многоклассовом случае может дать понять что с чем классификатор путает.

Binary Classification Error

$$y(\bar{x}, \bar{w}) = \{0, 1\}$$

✓ True Positive $= |\{u_i \in C_1 \mid y(\bar{x}, \bar{w}) = 1\}|$

✓ True Negative $= |\{u_i \in C_0 \mid y(\bar{x}, \bar{w}) = 0\}|$

✗ False Positive $= |\{u_i \in C_0 \mid y(\bar{x}, \bar{w}) = 1\}|$

✗ False Negative $= |\{u_i \in C_1 \mid y(\bar{x}, \bar{w}) = 0\}|$

$$\text{Accuracy} = \frac{TP + TN}{N}$$

$TP + TN + FP + FN = \#$
$TP + FN = \#P$
$TN + FP = \#N$

Problem of accuracy is in imbalanced dataset.

Здоровых людей больше, чем больных:
(99%) (1%)

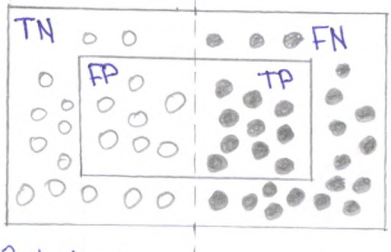
if $y(\bar{x}, \bar{w}) = 0$, then accuracy is 99%, but it is useless.

Ошибки первого и второго рода

$$\alpha_N = \frac{FP}{FP + TN} \quad (FPR)$$
false positive rate

$$\beta_P = \frac{FN}{FN + TP} \quad (FNR)$$
false negative rate

$\text{Recall} + \beta_P = 1$



○ - Negative
● - Positive

Best when:

If model → 1
then $\text{Precision} = \frac{>0}{>0+>0} > 0$
$\text{Recall} = \frac{>0}{>0+0} = 1$
$\text{Fscore} = \frac{2 \cdot 1 \cdot >0}{1 + >0} = \frac{0{,}00\ldots 1}{1{,}00\ldots 1} = >0$ $\quad \frac{2 \cdot 1 \cdot 0.5}{1 + 0.5} = \frac{1}{1.5} = 66\%$

TN можем быть неопределен

Precision & Recall

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN} \quad (TPR)$$
true positive rate

$$F_1\text{-Score} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Fscore vs. average

Sensitivity & Specificity

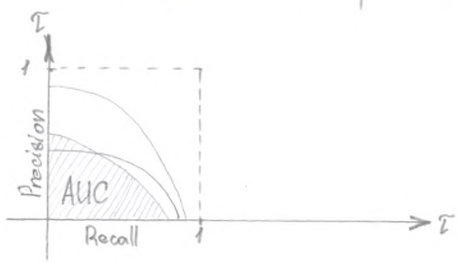
$$\text{Sensitivity (TPR, Recall)} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{TN + FP} \quad (TNR)$$
true negative rate

Precision - Recall Curve

Precision & Recall - значения для определённого порогового значения (τ, max).

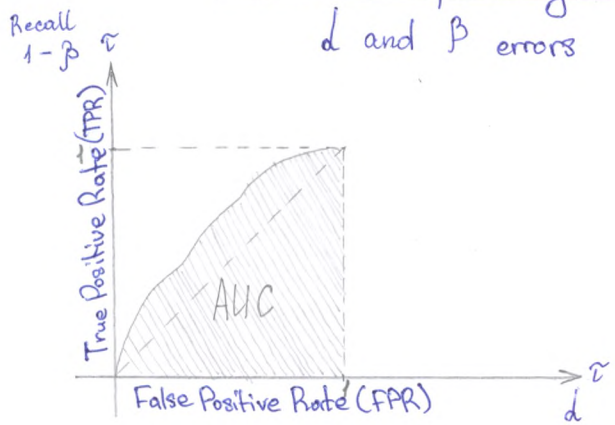
t	p_0	p_1	$\tau = 0{,}5$	$\tau = 0{,}6$	$\tau = 0{,}75$
0	0,3	0,7	1	1	0
0	0,4	0,6	1	1	0
1	0,45	0,55	1	0	0
1	0,2	0,8	1	1	1
		↑	Acc. = 0,5	Acc = 0,25	Acc. = 0,75



AUC - Area Under the Curve

Receiver Operating Characteristic (ROC)

α and β errors



Accuracy:

$$\text{Precision} = \frac{TP}{TP + FP}$$

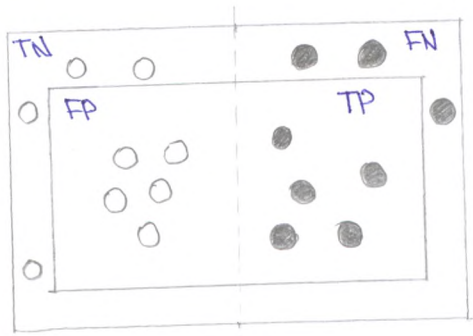
$$\text{Recall} = \frac{TP}{TP + FN} \quad (TPR)$$

Error:

$$\alpha_N = \frac{FP}{FP + TN} \quad (FPR)$$

$$\beta_P = \frac{FN}{FN + TP} \quad (FNR)$$

● - positive
○ - negative



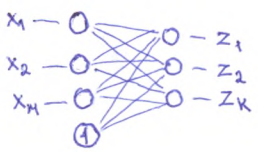
Logistic Regression (Classification)

Data

$\vec{x}_i$	Class Name	t_i	One-hot encoding vector
$\vec{x}_1$	Cat	1	[1 0 0]
$\vec{x}_2$	Dog	2	[0 1 0]
$\vec{x}_3$	Horse	3	[0 0 1]
$\vec{x}_n$	"String"	$t_n \in [1;K]$	$\vec{t}_n = [t_n^{(1)}\ t_n^{(2)}\ t_n^{(K)}]^T$ $t_n^{(t_n)} = 1$, otherwise $t_n^{(k)} = 0$

Много маленьких линейных регрессий

$$\begin{cases} z_1 = w_{10} + w_{11}x_1 + w_{12}x_2 + \dots + w_{1M}x_M = \bar{w}_1^T\bar{x} + b_1 \\ z_2 = w_{20} + w_{21}x_1 + w_{22}x_2 + \dots + w_{2M}x_M = \bar{w}_2^T\bar{x} + b_2 \\ \dots \\ z_K = w_{K0} + w_{K1}x_1 + w_{K2}x_2 + \dots + w_{KM}x_M = \bar{w}_K^T\bar{x} + b_K \end{cases}$$



$\bar{x} \in R^M$

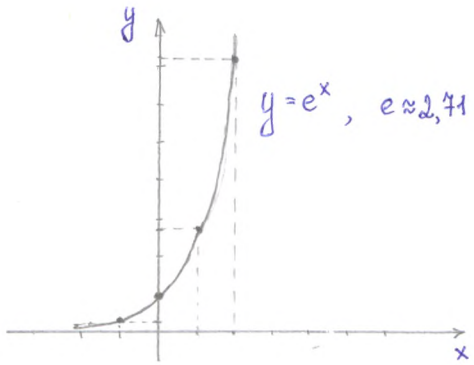
$\bar{y} \in R^K$

$$W^{K\times M} = \begin{bmatrix} \bar{w}_1^T \\ \bar{w}_2^T \\ \dots \\ \bar{w}_K^T \end{bmatrix}, \quad \mathbb{W}^{K\times(M+1)} = (W\ \bar{b})$$

$$\mathbb{W}^{K\times(M+1)} \cdot \vec{X}^{(M+1)\times 1} = \vec{Z}^{K\times 1}$$

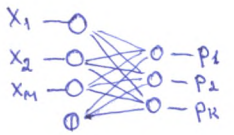
($\vec{X}$: # of samples)

- $\vec{y}_i(\vec{x}_i, \mathbb{W}) = \text{Softmax}(W\bar{x} + \vec{b}) =$
- $= \text{Softmax}(\mathbb{W}\vec{x})$, $\vec{x} = [1\ \bar{x}^T]^T$
- $\text{Softmax}(\vec{z}) = \left(\frac{e^{z_i}}{\sum_i e^{z_i}}\right)$
  - 1) $\sum_i \frac{e^{z_i}}{\sum_i e^{z_i}} = 1$
  - 2) Softmax is differentiable function



Logistic Regression. Loss Function.

$$\vec{y}_i(\vec{x}_i, W) = \text{Softmax}(W\vec{x}_i + \vec{b}), \quad \vec{y} \in R^{K\times 1}$$



p_k - Softmax probability, not PDF of Normal Distribution.

Вероятность правильного класса i на 1-ом sample-е и класса j на 2-ом sample-е.

$$P(y_i, y_j) = y_i(\vec{x}_1, W) \cdot y_j(\vec{x}_2, W)$$ - Independent Events: $P(A \text{ and } B) = P(A) \cdot P(B)$

Maximum Likelihood

Оценка общей вероятности для training samples $X^{M\times N}$, которые есть в обучающей выборке, наблюдать правильные ответы $T^{K\times N}$

$$P(T \mid X, W) = \underbrace{y_{t_1}^{(t_1)}}_{y_k \text{ on right class}}(\vec{x}_1, W) \cdot \ldots \cdot y_{t_n}^{(t_n)}(\vec{x}_n, W) \to \max_W$$

$$P(T \mid X, W) = \prod_{n=1}^{N} \underbrace{\prod_{k=1}^{K} y_k^{(k)}(\vec{x}_n, W)^{t_n^{(k)}}}_{y_n^{(t_n)}(\vec{x}_n, W) = \prod_{k=1}^{K} y_n^{(k)}(\vec{x}_n, W)^{t_n^{(k)}}} \to \max_W$$

$t_n^{(k)} \in \{0, 1\}$

$\begin{cases} t_n^{(k)} = 1 & \text{if } k = t_n \\ t_n^{(k)} = 0 & \text{if } k \neq t_n \end{cases}$

$\vec{t}_n = [0\,0\,0 \ldots 1 \ldots 0\,0\,0]^T$ - one-hot encoding

t_n - # of class that's correct.

$$-\ln(P(T \mid X, W)) = -\sum_{n=1}^{N}\sum_{k=1}^{K} \ln\left(y_n^{(k)}(\vec{x}_n, W)^{t_n^{(k)}}\right) \to \min_W$$

Loss Function (Целевая функция)

$$E(W) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_n^{(k)} \ln\left(y_n^{(k)}(\vec{x}_n, W)\right) \to \min_W$$

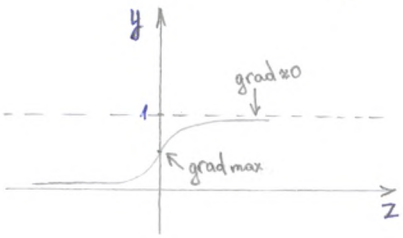
Weights initialization (Glorot / Xavier)

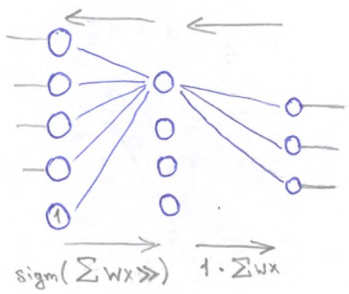
Vanishing / Exploding Gradients Problem ?

$y(\bar{X}, W) = sigm(\underbrace{\sum_i w_i x_i + b}_{z})$, $X \sim \mathcal{N}(0, \sigma^2 = 1)$

sigm → 1) $X \in [0; 1]$
$W \in [0; \frac{1}{n}]$
$\sum_i^n x \cdot w = [0; 1]$

tanh → 2) $X \in [-1; 1]$ or $\mathcal{N}(0, 1)$
$W \in [-\frac{1}{n}; \frac{1}{n}]$ or $\mathcal{N}(0, \frac{1}{n})$
$\sum_i^n xw = [-1; 1]$
↑ we need this





$$w_i \sim \mathcal{N}(\mu = 0, \sigma^2 = \left(\frac{2}{n_{in} + n_{out}}\right)^2)$$

average

↑ sigms ≈ 1 ↑ exploding gradient

Balance between :
1) $\sum wx \gg$
2) Exploding gradient

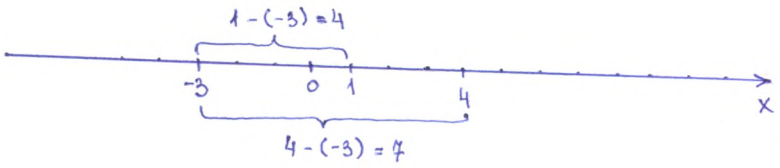
Data Normalization

$$\vec{X} = [\, X^{(1)} \quad X^{(2)} \quad \dots \quad X^{(M)} \,]^T$$

$$\tilde{X}^{(j)} = 2\,\frac{X^{(j)} - \min_i X_i^{(j)}}{\max_i X_i^{(j)} - \min_i X_i^{(j)}} - 1$$

$$\tilde{X}^{(j)} \in [-1\,;\,1]$$

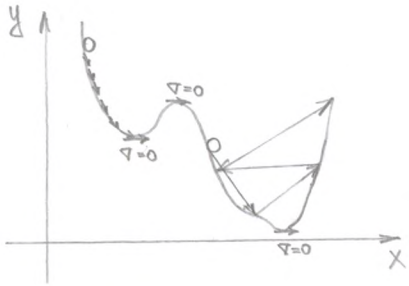
$X^{(j)}$ нормализуем именно по j значениям векторов $\vec{X}_i$ из $X \in R^{M \times N}$



$$\tilde{X}^{(j)} = \frac{X^{(j)} - \mu^{(j)}}{\sigma^{(j)}}, \quad \sigma \gg 0$$

$$\tilde{X}^{(j)} \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$$

Momentum SGD



1. ∇_1 ← velocity
2. $\boxed{\nabla_2 + \mu \cdot \nabla_1}$
3. $\nabla_3 + \mu\nabla_2 + \mu^2\nabla_1 = \nabla_3 + \mu(\nabla_2 + \mu\nabla_1)$

$V_2 = V_1 + \mu V_0$ но если он не может сразу остановиться.

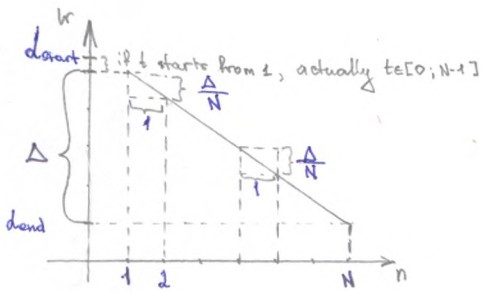
velocity = 0
iter
  velocity * = momentum
  velocity += ∇E(w)

  w -= lr * velocity

Regularized SGD

$w -= lr \cdot \nabla E(w) + \lambda w$

Learning Rate Decay



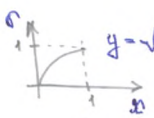
1. Linear

$d_t = d_{start} - (d_{start} - d_{end}) \cdot \frac{t}{N}$

2. Exponential

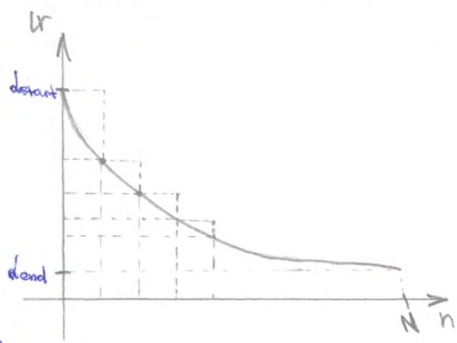
$d_t = d_{start} \sigma^t$

$\sigma = \left(\frac{d_{end}}{d_{start}}\right)^{\frac{1}{N-1}}$



$y = \sqrt{x}$, $x \in [0; 1]$

$d_t = d_{start} \cdot \left(\frac{d_{end}}{d_{start}}\right)^{\frac{1}{N-1}}$



$\left(\left(d_{start} \cdot \underbrace{\sqrt[3]{\sigma}}_{\text{const}}\right) \cdot \sqrt[3]{\sigma}\right) \cdot \sqrt[3]{\sigma} = d_{start} \cdot \frac{d_{end}}{d_{start}} = d_{end}$

(0, 1, 2, 3)

Dropout

- training 0,2 → 0 , p
- inference input · 0,8 , we mimic Magnitude · (1-p)

Logistic Regression. Model parameters estimation.

Gradient Descent

Loss Function

$$E(\mathbb{W}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_n^{(k)} \cdot \ln\left(y^{(k)}(\bar{x}_n, \mathbb{W})\right) \longrightarrow \min_{\mathbb{W}}, \quad \mathbb{W} = (W, \vec{b})$$

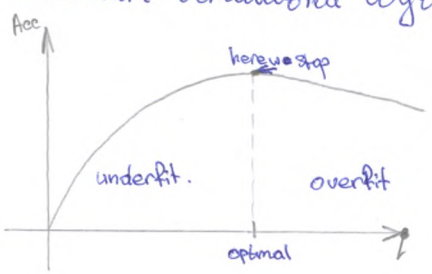
1. Случайная инициализация весов.

$$w_i \sim \mathcal{N}\left(\mu = 0, \sigma^2 = \frac{1}{M}\right)$$

2. Итеративный процесс:

$$\mathbb{W}_k = \mathbb{W}_{k-1} - \gamma \cdot \nabla_{\mathbb{W}} E(\mathbb{W}_{k-1}), \quad \gamma \text{ - learning rate, hyperparameter.}$$

3. Метод прекращает работу, когда точность на валидационной выборке перестаёт расти.
Момент остановки обучения - это тоже Hyperparameter.



Logistic Regression. Model parameters estimation.

Loss Function

$$E(\mathbb{W}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_n^{(k)} \ln y^{(k)}(\bar{x}_n, \mathbb{W}) \to \min_{\mathbb{W}}, \quad \mathbb{W} = (\bar{W}, \vec{b})$$

Для 1-го элемента выборки

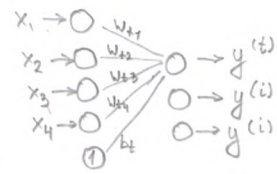
$$E(\mathbb{W}) = -\sum_{k=1}^{K} t^{(k)} \ln y^{(k)}(\bar{x}, \mathbb{W}) = -\ln y^{(t)}(\bar{x}, \mathbb{W}) \to \min_{\mathbb{W}}$$

$$\bar{y}(\bar{x}, \mathbb{W}) = Softmax(\bar{z}(\bar{x}, \mathbb{W})) = Softmax(W\bar{x} + \bar{b})$$

$$E(\mathbb{W}) = -\ln y^{(t)}(\bar{x}, \mathbb{W}) = -\ln\left(\frac{e^{z_t}}{\sum_{k=1}^{K} e^{z_k}}\right) = -\ln e^{z_t} + \ln\left(\sum_{k=1}^{K} e^{z_k}\right) = -z_t + \ln\left(\sum_{k=1}^{K} e^{z_k}\right) \to \min_{\mathbb{W}}$$

$$\begin{cases} z_i = \sum_j w_j x_j = \bar{w}_i^T \bar{x}_i + b_i \\ \vec{z} = W\bar{x} + \vec{b} \end{cases}$$

$$\frac{dE(\mathbb{W})}{dw_{tl}} = -(w_{tl} x_l)' + \underbrace{\frac{1}{\sum_{k=1}^{K} e^{z_k}} \cdot e^{z_t}}_{y^{(t)}} \cdot (w_{tl} \cdot x_l)' = -x_l + y_t^{(t)} \cdot x_l = (y_t^{(t)} - 1) \cdot x_l$$



$$\frac{dE(\mathbb{W})}{dw_{il}} = -0 + y_t^{(i)} \cdot (w_{il} \cdot x_l)' = y^{(i)} \cdot x_l, \quad i \neq t$$

$$\nabla_{\mathbb{W}} E = \begin{bmatrix} \frac{dE}{dw_{11}} & \frac{dE}{dw_{12}} & \frac{dE}{dw_{13}} & \cdots & \frac{dE}{dw_{1M}} \\ \cdots & & & & \\ \frac{dE}{dw_{K1}} & \frac{dE}{dw_{K2}} & \frac{dE}{dw_{K3}} & \cdots & \frac{dE}{dw_{KM}} \end{bmatrix} = \begin{bmatrix} y_1 x_1 & y_1 x_2 & \cdots & y_1 x_M \\ (y-t) x_1 & (y-t) x_2 & \cdots & (y-t) x_M \\ \cdots & & & \\ y_K x_M & y_K x_2 & \cdots & y_K x_M \end{bmatrix}$$

$$\nabla_{\mathbb{W}} E = (\bar{y} - \bar{t}) \cdot \bar{x}^T = \begin{bmatrix} \cdot \\ \cdot \\ \cdot \end{bmatrix}^{K \times 1} \cdot \begin{bmatrix} \cdot & \cdot & \cdot & \cdot \end{bmatrix}^{1 \times M} = \begin{bmatrix} \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}^{K \times M}$$

$$\frac{dE}{db_t} = -1 + y_t = y_t - 1$$

$$\frac{dE}{db_i} = -0 + y_i = y_i$$

$$\Rightarrow \nabla_{\bar{b}} E = (\bar{y} - \bar{t})$$

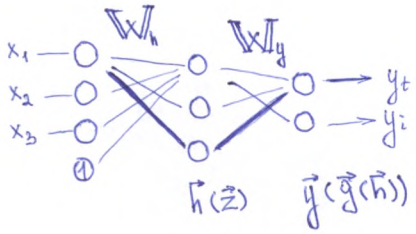
Regularization

$$\nabla_{W} E = (\bar{y} - \bar{t}) \cdot \bar{x}^T + \lambda W$$

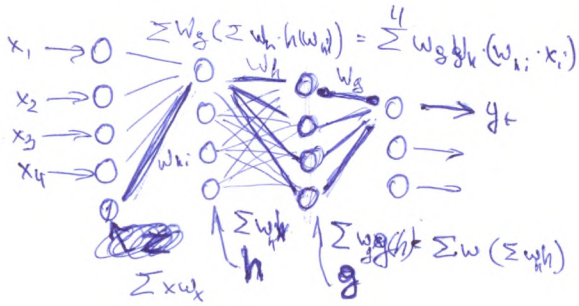
$$\nabla_{\bar{b}} E = (\bar{y} - \bar{t})$$

- Batch gradient descent - entire dataset
- Mini-batch - sample from dataset
- Stochastic - 1 element from dataset

Multi layer perceptron



$\text{Sum} \xrightarrow{\vec{z}} \text{Sigm} \xrightarrow{\vec{h}} \text{Sum} \xrightarrow{\vec{z}} \text{Sigmoid} \xrightarrow{\vec{h}} \text{Sum} \xrightarrow{\vec{g}} \text{Softmax} \xrightarrow{\vec{y}}$



$$\vec{y}(\vec{g}(\vec{h}(\vec{z}(\vec{x})))) = \text{Softmax}(\vec{g}(\vec{h})) = \left(\frac{e^{g_i}}{\sum_{j=1}^{K} e^{g_j}}\right)$$

$$P(T|X, W_h, W_y) = \prod_{n=1}^{N}\prod_{k=1}^{K} y^{(k)}(\bar{x}_n, W_h, W_y)^{t_n^{(k)}} \to \max_{W_h}$$

$$-\ln(P(T|X, W_h, W_y)) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_n^{(k)} \ln y^{(k)}(\bar{x}_n, W_h, W_y) \to \min_{W_h}$$

$$-\sum_{k=1}^{K} \ln y^{(k)}(\bar{x}_n, W_h, W_y)^{t^{(k)}} = -\ln y^{(t)}(\bar{x}, W_h, W_y)^1 \to \min_{W_h, W_y}$$

$$E(W_h) = -\ln y^{(t)}(\bar{x}, W_h, W_y) = -\ln\left(\frac{e^{g_t(\vec{h})}}{\sum_{k=1}^{K} e^{g_k(\vec{h})}}\right) = -\ln\left(e^{g_t(\vec{h})}\right) + \ln\left(\sum_{k=1}^{K} e^{g_k(\vec{h})}\right) \to \min_{W_h}$$

$$E(W_h) = -g_t(\vec{h}) + \ln\left(\sum_{k=1}^{K} e^{g_k(\vec{h})}\right) \to \min_{W_h}$$

$$g_t(\vec{h}(\vec{z})) = \sum_{i=1}^{H} w_{ti} \cdot h_i(z_i)$$

$$h_i(z_i) = \frac{1}{1+e^{-z_i}}$$

$$z_i(\bar{w}_{hi}) = \sum_{i=1}^{I} w_{hi} \cdot x_i$$

$$h(x) = \frac{1}{1+e^{-z_x}}$$

$$X(w) = \sum w_i x_i$$

?

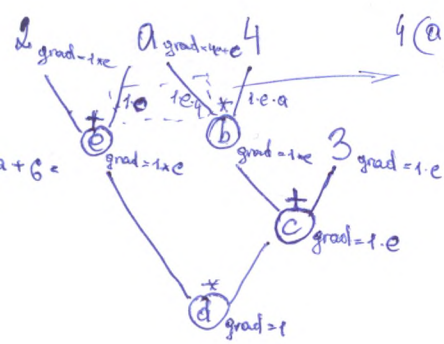
!

AutoGrad

DL from scratch p.176

```
a = NumberWithGrad(3)
b = a + 4
c = b + 3
d = c + (a+2)
```

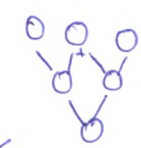
$d(a) = (b+3)(a+2) = (4a+3)(a+2) = 4a^2 + 8a + 3a + 6 = 8a + 11$



$4(\underbrace{a+2}_{e}) + (\underbrace{b+3}_{c}) = 4a + 8 + 4a + 3 = 8a + 11$

$$\frac{df}{dg}\frac{dg}{dx} + \frac{df}{dh}\frac{dh}{dx} \; ?$$

пока $F(g(x), h(x))$?



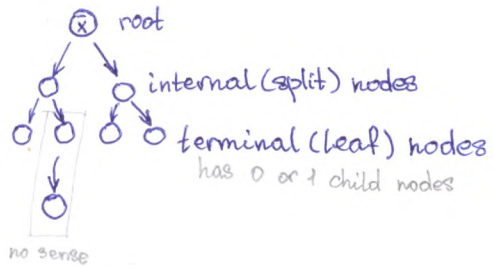
Почему мы суммируем градиенты тензоров участвующих/его в нескольких операциях?

Это можно и совпадает с нашим решением

Decision Tree

Decision Tree ML Algorithm – набор слабых классификаторов, организованных в иерархическую структуру – дерево.

Дерево – связный ориентированный граф без циклов, в котором каждая вершина имеет не более 1-го ребра.

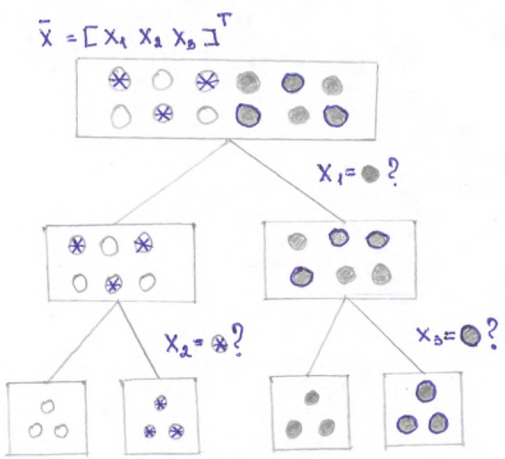


Слабый классификатор – это underfit-ящий классификатор, который имеет мало параметров, высокую ошибку, опирается на небольшое кол-во features, быстро вычисляется.

Главное преимущество Decision Tree в том, что для него не нужен весь вектор характеристик ($\vec{x}$), в отличии от регрессий, то есть когда есть стоимость вычисления каждой характеристики, можно минимизировать издержки $\longleftrightarrow$ максимизировать прибыль.

Например, в случае логистической регрессии для диагностирования заболевания нужно как будто сдать все возможные мед. тесты. Но, если мы сдали 1 и всё ok, то возможно остальные 500 сдавать не нужно.

Decision Tree легко интерпретировать.



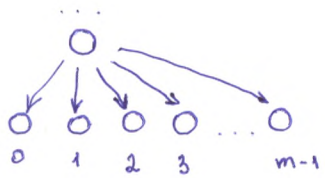
Используем 2 параметра из 3-х.
(feature)

Decision Tree. Internal (split) nodes.

Split Function - разделяющая функция.

$h(\bar{x}, \bar{\theta}_j) : \mathbb{R}^D \times \mathcal{T} \to \{0, m-1\}$

args Domain Range

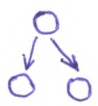


$\bar{\theta}_j \in \mathcal{T}$ - параметры разделяющей функции

theta

Split function - функция, определяющая следующий узел из m потомков j-го узла.

If $m = 2$ → Binary Decision Tree



$h(\bar{x}, \theta_j) : \mathbb{R}^D \times \mathcal{T} \to \{0, 1\}$

Feature Selection Function

$\bar{\psi}(\bar{x}) : \mathbb{R}^D \to \mathbb{R}^{D'}, \quad D' \leq D$

(psi)

$\bar{x} = [x_1\ x_2\ x_3\ x_4 \dots x_D]^T$

$\bar{\psi}(\bar{x}) = [x_2\ x_3]^T, \quad D' = 2$

$\bar{\psi}(\bar{x})$ - ~~гипер~~параметр,
своя отдельная для каждого внутреннего узла,
благодаря чему не используем весь feature vector $\bar{x}$.

$\psi(\bar{x}) = x_i$ - частый случай.

Количество используемых характеристик - гиперпараметр, потому что иначе лучший результат будет давать выбор всех, и мы будем выбирать все.

Какие именно характеристики - это параметр.

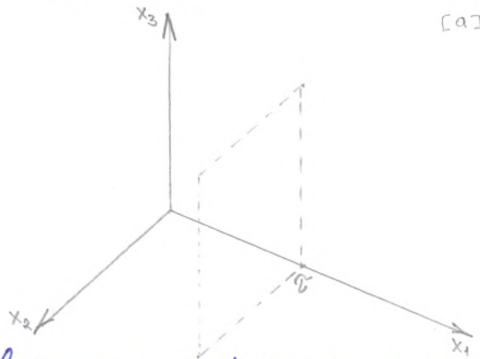
Decision Tree. 3 types of split function.

$[a] = \begin{cases} 1, a = \text{True} \\ 0, a = \text{False} \end{cases}$

1) Default

$h(\bar{x}, \theta_j) = [\underset{\infty}{\tau_1} > \psi(\bar{x}) > \underset{-\infty}{\tau_2}]$

$\theta_j = (\psi, \tau_1, \tau_2)$, $\psi(\bar{x}) = x_i$



Гиперплоскость параллельная всем осям координат, кроме одной.

i.e. $x_i < \tau$ (brute force)

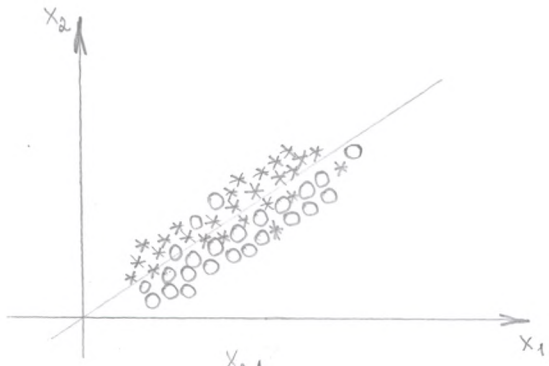
2) Linear

$h(\bar{x}, \theta_j) = [\tau_1 > \bar{\psi}(\bar{x})^T \cdot \bar{w} > \tau_2]$

$\theta_j = (\bar{\psi}, \bar{w}, \tau_1, \tau_2)$

i.e. $x_1 w_1 + x_2 w_2 < \tau$

$x_1 w_1 + \tau < x_2$ (regression)



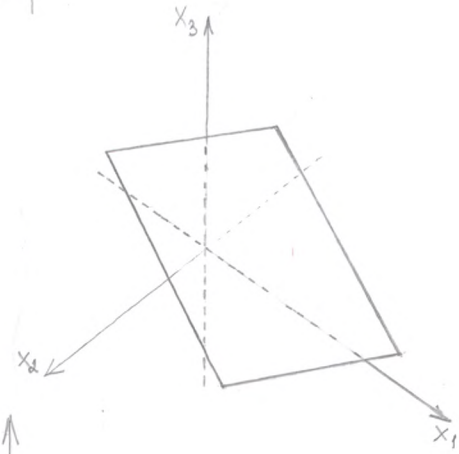
– 2D

3) Non - Linear

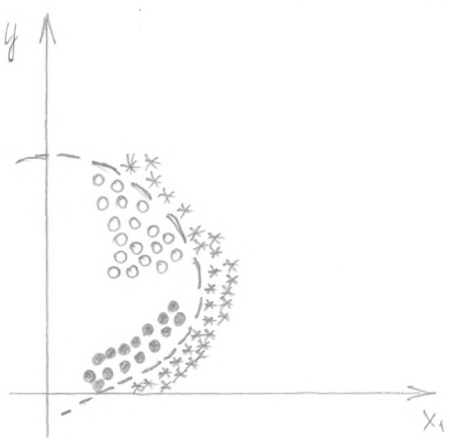
$h(\bar{x}, \theta_j) = [\tau_1 > \bar{\psi}(\bar{\psi}(\bar{x}))^T \bar{w} > \tau_2]$

$\theta_j = (\bar{\psi}, \bar{w}, \tau_1, \tau_2)$

ψ - basis functions.



– 3D



Задействуется только один τ (max).

Decision Tree Training.

Notation

Classification classes $C_1, C_2 \dots C_K$

S_i - множество элементов дошедших до узла i (часть dataset-а)

$S_i^{(k)}$ - множество элементов класса k, дошедших до узла i.

$N_i = |S_i|$ - количество элементов множества S_i

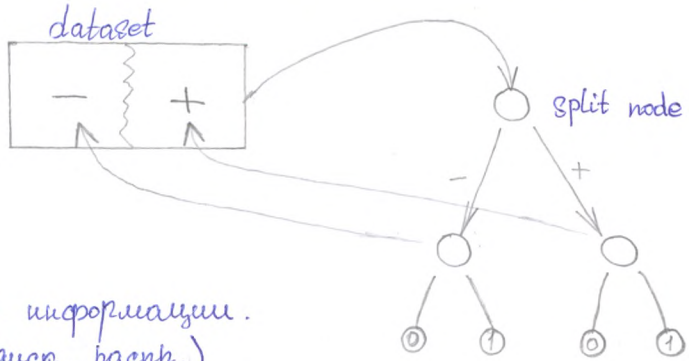
$N_i^{(k)} = |S_i^{(k)}|$

S_{ij} - множество элементов потомков i дошедших до узла j.

$S_{ij}^{(k)}$

$N_{ij} = |S_{ij}|$

$N_{ij}^{(k)} = |S_{ij}^{(k)}|$

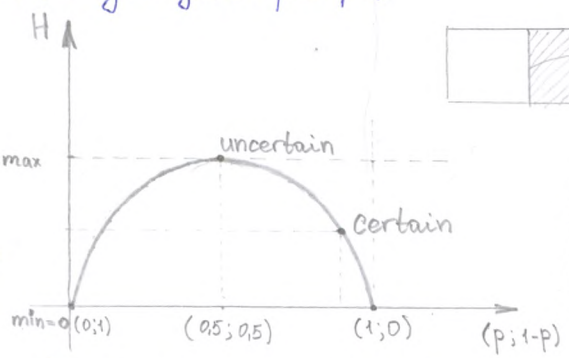


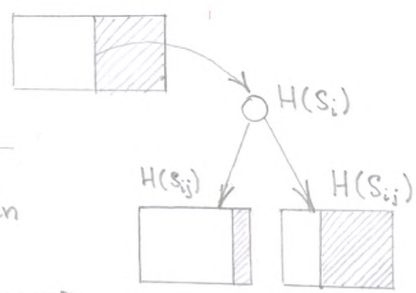
Entropy - ожидание количества информации.
(Теория информации для дискр. распр.)

$$H = -\sum_i p_i \log p_i$$

$$\frac{N_i^{(k)}}{N_i} = p^{(k)}$$

$$H(S_i) = -\sum_{k=1}^{K} \frac{N_i^{(k)}}{N_i} \cdot \log \frac{N_i^{(k)}}{N_i}$$





Энтропия низкая при детерминистическом распределении количества классов, и наоборот высокая при uniform распределении.

Information Gain
(Loss Function)

$$IG = H(S_i) - \sum_j^m \frac{N_{ij}}{N_i} \cdot H(S_{ij}) \to \max_\theta$$

$$E(\theta) = \sum_j^m \frac{N_{ij}}{N_i} \cdot H(S_{ij}) \to \min_\theta \quad \in [0; H(S_i)]$$

Находим $\theta = (\psi, \tau)$, которые минимизирует целевую функцию.
Минимизируем uncertainty на child узлах.

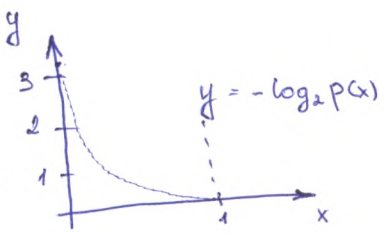
Information theory

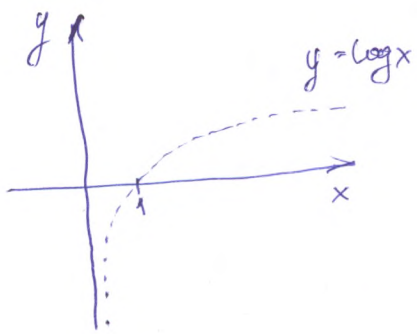
Information

$h(x) = -\log_2(p(x))$

Shannon, e-mats

rare events have more information.
В data compression используется.



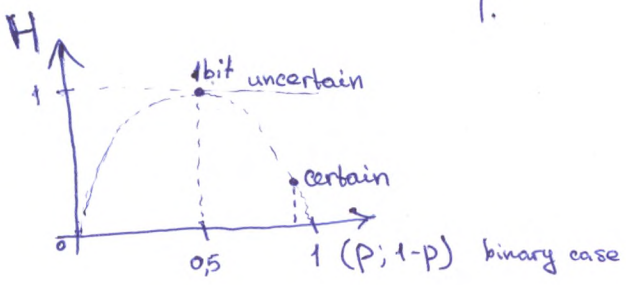


Entropy

$$H(x) = -\sum_i p_i(x) \cdot \log_2(p_i(x))$$

Shannon Entropy

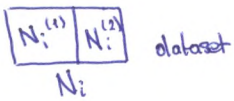
$$\frac{1}{2} \cdot \log_2 \frac{1}{2} = \frac{1}{2} \cdot (\log_2 1 - \log_2 2) = \frac{1}{2}(0 - 1) = -\frac{1}{2}$$



Entropy, или ожидание информации, больше всего когда, распределение uniform и каждое событие может произойти с одинаковой вероятностью, то есть они uncertain. Deterministic distr. [90% 10%] - low entropy. certain

$$\frac{N_i^{(k)}}{N_i} = p^{(k)}$$

$$H(S_i) = \sum_{k=1}^{K} \frac{N_i^{(k)}}{N_i} \cdot \log_2 \frac{N_i^{(k)}}{N_i}$$

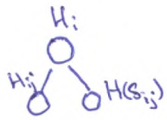


Ожидание информации,
uncertain - много информации.

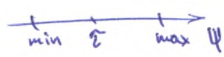
Information Gain

$$IG = \underbrace{H(S_i)}_{\text{не влияет на split found}} - \sum_j \frac{N_{ij}}{N_i} \cdot H(S_{ij}) \rightarrow \max_\Theta$$

$E(\Theta)$



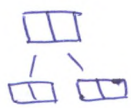
$$E(\Theta) = \sum_j \frac{N_{ij}}{N_i} \cdot H(S_{ij}) \rightarrow \min_\Theta$$



минимизируем ожидание uncertainty энтропии на child узлах, то есть сделать certain (90% одного класса)

1) $H_i(\{0,5; 0,5\}) = 1$

$IG = 1 - \sum_i 0,5 \cdot 1 = 0$



useless

2) $IG = 1 - (0,1 \cdot 0 + 0,9 \cdot H_{ij}(\{0,3; 0,7\}) = 1 - (0,9 \cdot (0,3 \log_2 0,3 + 0,7(\log_2 0,7)) = 1 - 0,\boxed{0,5}$

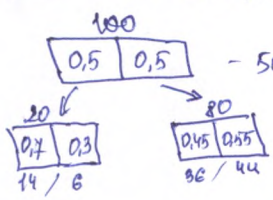
~~$IG \in [0; 1]$~~ $IG \in [0; H(S_i)]$

отщипнули чистую выборку

0,375 0,625

actually percents should be different

3)



- 50/50

$H(S_i) = -\sum p(x_i) \log_2 p(x_i) = -(\frac{1}{2} \cdot \log_2 \frac{1}{2}) = 1$

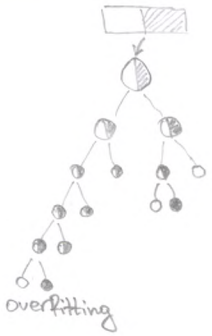
$IG = 1 - (0,2 \cdot H(S_i) + 0,8\, H(S_j)) = 1 - 0,9936 = \boxed{0,03}$ ⇒ лучше отщипнуть чистую выборку

$- 0,2\,(0,7 \log_2 0,7 + 0,3 \log_2 0,3) = 0,2 \cdot 0,88$

$- 0,8\,(0,45 \log_2 0,45 + 0,55 \log_2 0,55) = 0,8 \cdot 0,9928$

Decision Tree. Terminal Node.

Stopping Criteria (to create terminal node)



- Достигнута заданная глубина d
- $N_i < n$
- $H(S_i) < \varepsilon$

If $N_i = N_i^{(k)}$ - были бы очень глубокие переобученные деревья, с малым кол-вом элементов на терминальных узлах. Мало элементов на внутренних узлах → overfitting.

Random Forest: overfit + overfit = ok

Prediction

1) $y = \underset{k}{\operatorname{argmax}}\, N_i^{(k)} \to \{0, 1, \dots, K\}$

2) $\vec{y} = \left(\frac{N_i^{(k)}}{N_i} \quad \frac{N_i^{(2)}}{N_i} \quad \frac{N_i^{(3)}}{N_i} \dots \frac{N_i^{(K)}}{N_i} \right)^T$

Большая уверенность похожа на overfitting.

Random Forest

Random Forest - это ансамбль деревьев решений.

Ensemble classifier - это объединение нескольких классификаторов.

Если мы объединяем в один алгоритм несколько разных алгоритмов, то это будет называться fusion classifier.

С помощью слабых линейных классификаторов RF может описывать нелинейные данные.

M деревьев: $T_1, T_2, \dots, T_M$

$$RF(\bar{x}) = RF(T_1(\bar{x}), T_2(\bar{x}), \dots, T_M(\bar{x})) = \frac{1}{M}\sum_{j=1}^{M} T_j(\bar{x})$$

$$T(\bar{x}) = \left(\frac{N_i^{(1)}}{N_i} \quad \frac{N_i^{(2)}}{N_i} \cdots \frac{N_i^{(K)}}{N_i}\right), \quad \bar{x} \in S_i \text{ при обучении}$$

Проблема: детерминированный процесс обучения DecisionTree на одинаковой обучающей выборке даёт одинаковые деревья.

$$T_i = T_j, \ \forall i,j \Rightarrow RF(\bar{x}) = T_1(\bar{x})$$

Bagging (Bootstrap Aggregation)

Метод во взятии случайного подмножества обучающей выборки.

Random Sampling with replacement, то есть можно брать один и тот же элемент несколько раз.

$\{1,2,3,4,5,6,7\} \rightarrow \{3,2,1,5,5,2\}$

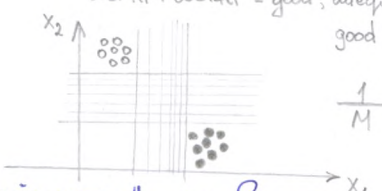
Random Node Optimization

Выбор параметров split function из случайного подмножества параметров разделения.

$$IG \rightarrow \max_{\theta_i \in T_i}, \quad T_i \subset T$$

overfit + underfit = good, adequate confidence

good confidence map



$$\frac{1}{M}\sum_{j=1}^{M} T_j(\bar{x})$$

$\rho = |T_i|$ - "количество случайности", комбинация параметров θ_i.

$\theta_1 = (x_1, \tau_1)$
$\theta_2 = (x_2, \tau_2)$
$\theta_3 = (x_1, \tau_3)$
$\theta_4 = (x_1, \tau_4)$
$\theta_5 = (x_2, \tau_5)$

$\rightarrow \arg\max_{\theta} IG$ - выбираем комбинацию, дающую максимальное значение IG.

- $\rho = 1$, оптимизации нет, так себе деревья, но разнообразные
- $\rho = n$, если n большое, деревья будут ближе к оптимальному
- $\rho = |T|$, если переберем все значения, то деревья одинаковые

Чем больше кол-во деревьев, M, тем лучше, но после какого-то момента не будет давать улучшений, а только нагружать.

$\frac{(n+k-1)!}{k!(n-1)!}$ unique sets

ABC
ABCA $\binom{3+4-1}{4}$
permutations $\binom{2+3-1}{2} = \frac{4!}{2!\cdot 2!}$
4!

○○○○

A
B
C
○ - A|B|C

AdaBoost. Adaptive Boosting

Недостаток Random Forest в том, что он прислушивался к каждому дереву. RF не имеет возможности обратить внимание на то, что есть особенно сложные примеры, чтобы закрыть проблему.

Взвешенная обучающая выборка

Training samples	Sample weight	Class labels
X_i	w_i	t_i
X_N	w_N	t_N
$\vec{X}$	$\vec{w}$	$\vec{t}$

Целевая функция

Регрессия:

$$E(\theta) = \frac{1}{2}\sum_{i=1}^{N} w_i (t_i - y(\bar{x}_i; \theta))^2$$

На тех элементах, на которых большие веса, мы хотим быть правы.

Классификация:

$$E(\theta) = -\sum_{i=1}^{N} w_i \sum_{k=1}^{K} t_i^{(k)} \ln y^{(k)}(\bar{x}_i, \theta)$$

Дерево решений:

$$IG_w = H_w(S_i) - \sum_j \frac{\tilde{N}_{ij}}{\tilde{N}_i} H_w(S_{ij}) \to \max_\theta$$

$$H_w(S_i) = -\sum_k \frac{\tilde{N}_i^{(k)}}{\tilde{N}_i} \log \frac{\tilde{N}_i^{(k)}}{\tilde{N}_i}$$

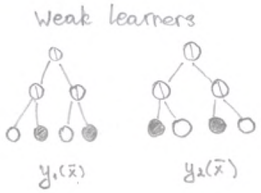
$\tilde{N}_i = \sum_{x_i \in S_i} w_i$ — не количество элементов, а сумма весов элементов. например, если вес нулевой, он никак не влияет на целевую функцию.

$$\vec{t}_i = \left(\frac{\tilde{N}_i^{(1)}}{\tilde{N}_i} \quad \frac{\tilde{N}_i^{(2)}}{\tilde{N}_i} \quad \frac{\tilde{N}_i^{(3)}}{\tilde{N}_i} \quad \dots \quad \frac{\tilde{N}_i^{(K)}}{\tilde{N}_i}\right)^T$$

AdaBoost. Постановка задачи.

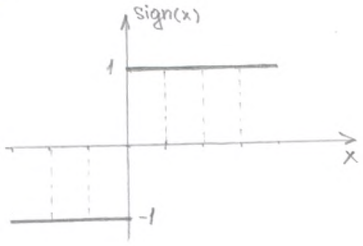
Бинарная задача классификации:

C_{-1}, C_1 ; $t_i \in \{-1, 1\}$, $i = 1, 2, \dots, N$



Слабые классификаторы:

$y_j(\bar{x}) \Rightarrow \{-1, 1\}$, $j = 1, 2 \dots, M$



Итоговый классификатор:

$$Y(\bar{x}) = \text{sign}\left(\sum_{j=1}^{M} d_j y_j(\bar{x})\right)$$

params

d_j — weight of weak classificator

Если d_j отрицательно, то он результат инвертирует

Инициализация весов обучающей выборки

На шаге j

$w_i^{(j)} \geq 0$, $\sum_{i=1}^{N} w_i^{(j)} = 1$

$w_i^{(1)} = \frac{1}{N}$, $i = 1, 2, \dots, N$

for balanced dataset.

or $w_i^{(1)} = \frac{1}{2N_k}$, for imbalanced dataset

For imbalanced dataset

1) downsampling
2) upsampling
3) $N = N_{-1} + N_1$

$w_i^{(1)} = \frac{1}{2N_1}$, $t_i = 1$

$w_i^{(1)} = \frac{1}{2N_{-1}}$, $t_i = -1$

Минимизируемая целевая функция

Основная идея, можно использовать другие.

Indicator function

$$I(a) = \begin{cases} 1, & \text{if } a = \text{True} \\ 0, & \text{if } a = \text{False} \end{cases}$$

$$J_j = \sum_{i=1}^{N} w_i^{(j)} \underbrace{I(y_j(\bar{x}_i) \neq t_i)}_{\theta_j}$$, если $y \neq t$, то вес суммируется, если нет, то не влияет

Это не эвристические методы, а выводятся из exponential loss.

Ошибка слабого классификатора

$$\varepsilon_j = \sum_{i=1}^{N} w_i^{(j)} \cdot I(y_j(\bar{x}_i) \neq t_i)$$, если $\varepsilon_j > \frac{1}{2}$, то он ошибается чаще, и можно инвертировать результаты тем угадывает

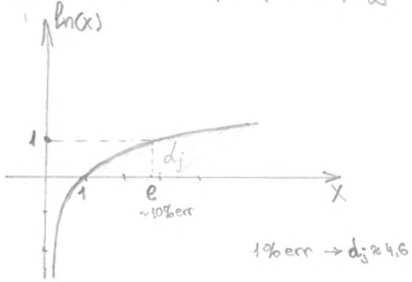
Вычисление веса слабого классификатора

$$d_j = \ln \frac{1 - \varepsilon_j}{\varepsilon_j}$$

best 1) $\ln \frac{1-0}{0} = \ln \infty = \infty$

random 2) $\ln \frac{1-\frac{1}{2}}{\frac{1}{2}} = \ln 1 = 0$

worst 3) $\ln \frac{1-1}{1} = \ln 0 = -\infty$



1% err → $d_j \approx 4.6$

Обновление весов обучающей выборки

$w_i^{(j)}$, $i = 1, 2, \dots, N$

$$w_i^{(j+1)} = \frac{w_i^{(j)} \cdot e^{d_j \cdot I(y_j(\bar{x}_i) \neq t_i)}}{Z_j} = \begin{cases} w_i^{(j)} \\ w_i^{(j)} \cdot (>1) \end{cases}$$ вес всегда растет.

$i = 1, 2, \dots, N$

$$Z_j = \sum_{i=1}^{N} w_i^{(j)} e^{d_j I(y_j(\bar{x}_i) \neq t_i)}$$ — for normalization.

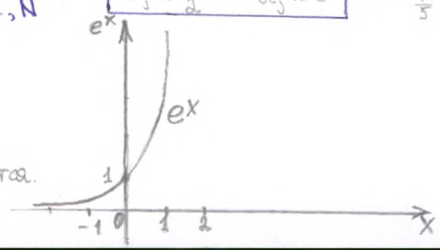
сумма увеличивается всегда,
и те элементы на которых мы правое веса уменьшаются.

if $\varepsilon = \frac{1}{2}$ → stop learning, nothing gonna change

$\varepsilon_j \leq \frac{1}{2} \rightarrow d_j \geq 0$

$\ln \frac{\frac{1}{2}}{\frac{1}{2}} = \ln \frac{1}{2} < 0$

$\ln \frac{\frac{4}{5}}{\frac{1}{5}} = \ln 4 > 0$



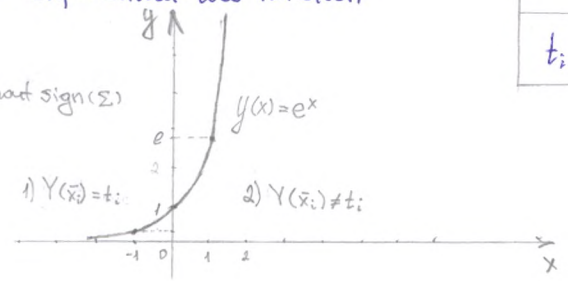
(Более наглядная анимация dataset-а: high w of AdaBoost, самые увеличенные синим NN etc)

Математические основы AdaBoost.

Целевая функция

$$E_0 = \sum_{i=1}^{N} \underbrace{e^{-\frac{1}{2} t_i Y_M^*(\bar{x}_i)}}$$ – exponential loss function

$$Y_M^*(\bar{x}_i) = \sum_{j=1}^{M} d_j y_j(\bar{x}_i)$$ – without sign(Σ)



weak learners $y_j(\bar{x}) \to \{-1; 1\}$, $j = 1, \dots, M$

$t_i \to \{-1; 1\}$

$e^{-\frac{1}{2} t_i Y_M^*(\bar{x}_i)}$; $t_i \to \{-1; 1\}$

1) $Y_M^*(\bar{x}_i) \to \{-c; c\}$
$\forall\, y_j(\bar{x}_i) = t_i$
$e^{-\frac{1}{2}|Y_M^*(\bar{x}_i)|}$

2) $\forall\, y_j(\bar{x}_i) \neq t_i$
$e^{+\frac{1}{2}|Y_M^*(\bar{x}_i)|}$

$y_1, y_2, \dots, y_{m-1}$ – already given weak learners

$y_m - ?$ – AdaBoost is greedy algorithm, so it cares only about currently last classifier m.

$$E_0 = \sum_{i=1}^{N} e^{-\frac{1}{2} t_i Y_m^*(\bar{x}_i)} = \sum_{i=1}^{N} e^{-\frac{1}{2} t_i (Y_{m-1}^*(\bar{x}_i) + d_m y_m(\bar{x}_i))} = \sum_{i=1}^{N} e^{-\frac{1}{2} t_i Y_{m-1}^*(\bar{x}_i) - \frac{1}{2} t_i d_m y_m(\bar{x}_i)} =$$

$$= \sum_{i=1}^{N} \underbrace{e^{-\frac{1}{2} t_i Y_{m-1}^*(\bar{x}_i)}}_{\tilde{w}_i^{(m)}} \cdot e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)} = \sum_{i=1}^{N} \tilde{w}_i^{(m)} \cdot e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)} = \underbrace{\tilde{Z}_m}_{\text{norm}} \underbrace{\sum_{i=1}^{N} w_i^{(m)} e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)}}_{E} \to \min_{d_m y_m}$$

- $w_i^{(m)} = \frac{\tilde{w}_i^{(m)}}{\tilde{Z}_m}$; $w_i^{(m)} > 0$; $\sum_{i=1}^{N} w_i^{(m)} = 1$
- $\tilde{Z}_m = \sum_{i=1}^{N} \tilde{w}_i^{(m)} \Rightarrow \tilde{Z}_m \cdot w_i^{(m)} = \tilde{Z}_m \cdot \frac{\tilde{w}_i^{(m)}}{\tilde{Z}_m} = \tilde{w}_i^{(m)}$

$$E = \sum_{i=1}^{N} w_i^{(m)} \cdot e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)} = \underbrace{\sum_{i=1}^{N} w_i^{(m)} e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)}}^{t_i \cdot y_m(\bar{x}_i) = -1} \cdot I(y_m(\bar{x}_i) \neq t_i) + \underbrace{\sum_{i=1}^{N} w_i^{(m)} e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)}}^{t_i y_m(\bar{x}_i) = 1} \cdot I(y_m(\bar{x}_i) = t_i) =$$

Incorrectly classified — Correctly classified

$$= \sum_{i=1}^{N} w_i^{(m)} \underbrace{e^{+\frac{1}{2} d_m}}_{\text{const}} \cdot I(y_m(\bar{x}_i) \neq t_i) + \sum_{i=1}^{N} w_i^{(m)} \underbrace{e^{-\frac{1}{2} d_m}}_{\text{const}} \cdot I(y_m(\bar{x}_i) = t_i) \pm \underbrace{e^{-\frac{1}{2} d_m} \sum_{i=1}^{N} w_i^{(m)} I(y_m(\bar{x}_i) \neq t_i)}_{0} =$$

$$= \left(e^{\frac{1}{2} d_m} - e^{-\frac{1}{2} d_m}\right) \sum_{i=1}^{N} w_i^{(m)} I(y_m(\bar{x}_i) \neq t_i) + e^{-\frac{1}{2} d_m} \underbrace{\sum_{i=1}^{N} w_i^{(m)}}_{\text{by def is 1}} \underbrace{\left(I(y_m(\bar{x}_i) = t_i) + I(y_m(\bar{x}_i) \neq t_i)\right)}_{1} =$$

$$= \underbrace{\left(e^{\frac{d_m}{2}} - e^{-\frac{d_m}{2}}\right)}_{+\text{const}} \underbrace{\sum_{i=1}^{N} w_i^{(m)} I(y_m(\bar{x}_i) \neq t_i)}_{J} + \underbrace{e^{-\frac{d_m}{2}}}_{\text{const}} \to \min_{d_m y_m}$$

($d_m \geq 0$)

$$J = \sum_{i=1}^{N} w_i^{(m)} \cdot I(y_m(\bar{x}_i) \neq t_i) = \varepsilon_m \Rightarrow E_{d_m} = \left(e^{\frac{d_m}{2}} - e^{-\frac{d_m}{2}}\right) \varepsilon_m + e^{-\frac{d_m}{2}} \to \min_{d_m}$$

$$E'_{d_m} = \left(\frac{1}{2} e^{\frac{d_m}{2}} + \frac{1}{2} e^{-\frac{d_m}{2}}\right) \varepsilon_m + \left(-\frac{1}{2} e^{-\frac{d_m}{2}}\right) = 0 \; ; \; \left(e^{\frac{d_m}{2}} + e^{-\frac{d_m}{2}}\right) \varepsilon_m = e^{-\frac{d_m}{2}} \;|\cdot e^{d_m/2} \; ; \; \left(e^{d_m} + 1\right) \varepsilon_m = 1 \; ; \; e^{d_m} + 1 = \frac{1}{\varepsilon_m} \; ; \; e^{d_m} = \frac{1}{\varepsilon_m} - \frac{\varepsilon_m}{\varepsilon_m}$$

$$d_m = \ln \frac{1 - \varepsilon_m}{\varepsilon_m}$$

Математические основы AdaBoost. (2)

$$E_0 = \sum_{i=1}^{N} e^{-\frac{1}{2} t_i Y_M^*(\bar{x}_i)}$$

$$Y_M^*(\bar{x}_i) = \sum_{j=1}^{M} d_j y_j(\bar{x}_i)$$

$y_1, y_2, \dots y_{m-1}$

$y_m - ?$

$$E_0 = \sum_{i=1}^{N} e^{-\frac{1}{2} t_i Y_M^*(\bar{x}_i)} = \sum_{i=1}^{N} e^{-\frac{1}{2} t_i (Y_{m-1}^*(\bar{x}_i) + d_m y_m(\bar{x}_i))} = \sum_{i=1}^{N} \underbrace{e^{-\frac{1}{2} t_i Y_{m-1}^*(\bar{x}_i)}}_{\tilde{w}_i^{(m)}} \cdot e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)} =$$

$$= \tilde{N} \sum_{i=1}^{N} w_i^{(m)} e^{-\frac{1}{2} t_i d_m y_m(x_i)}$$

when $m=1 \to \tilde{w}_i^{(m)} = e^0 = 1 \Rightarrow w_i^{(m)} = \frac{1}{\sum_{i=1}^{N} 1} = \frac{1}{N}$

$$w_i^{(m)} = \frac{\tilde{w}_i^{(m)}}{\tilde{N}_m} \Rightarrow w_i > 0, \quad \sum_{i=1}^{N} w_i = 1$$

$$\tilde{N}_m = \sum_{i=1}^{N} \tilde{w}_i^{(m)}$$

- $\tilde{w}_i^{(m)} = e^{-\frac{1}{2} t_i Y_{m-1}^*(\bar{x}_i)}$
- $\tilde{w}_i^{(m+1)} = e^{-\frac{1}{2} t_i Y_m^*(\bar{x}_i)} = e^{-\frac{1}{2} t_i (Y_{m-1}^*(\bar{x}_i) + d_m y_m(\bar{x}_i))} = e^{-\frac{1}{2} t_i Y_{m-1}^*(\bar{x}_i) - \frac{1}{2} t_i d_m y_m(\bar{x}_i)} =$

$$= \underbrace{e^{-\frac{1}{2} t_i Y_{m-1}^*(x_i)}}_{\tilde{w}_i} \cdot e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)} = \tilde{w}_i^{(m)} \cdot e^{-\frac{1}{2} t_i d_m y_m(\bar{x}_i)} = \dots$$

$$t_i y_m(x_i) = 1 - 2 I(y_m(\bar{x}_i) \neq t_i)$$

$$\begin{cases} t_i \cdot y_m(\bar{x}_i) = 1, & \text{if } t_i = y_m(\bar{x}_i) \\ t_i y_m(\bar{x}_i) = -1, & \text{if } t_i \neq y_m(\bar{x}_i) \end{cases}$$

$$\dots = \tilde{w}_i^{(m+1)} = \tilde{w}_i^{(m)} \cdot e^{-\frac{1}{2} d_m (1 - 2 I(y_m(\bar{x}_i) \neq t_i))} = \tilde{w}_i^{(m)} \cdot e^{-\frac{1}{2} d_m + I(y_m(\bar{x}_i) \neq t_i)} = \tilde{w}_i^{(m)} \cdot e^{-\frac{1}{2} d_m} \cdot e^{d_m I(y_m(\bar{x}_i) \neq t_i)}$$

Norm: $w_i^{(m+1)} = \frac{\tilde{w}_i^{(m+1)}}{\sum_{l=1}^{N} \tilde{w}_l^{(m+1)}} = \frac{\tilde{w}_i^{(m)} \cdot \left(e^{-\frac{1}{2} d_m}\right) \cdot e^{d_m I(y_m(x_i) \neq t_i)}}{\sum_{l=1}^{N} \tilde{w}_l^{(m)} \cdot \left(e^{-\frac{1}{2} d_m}\right) \cdot e^{d_m I(y_m(x_l) \neq t_l)}} \cdot \frac{e^{\frac{1}{2} d_m}}{e^{\frac{1}{2} d_m}} = \frac{\tilde{w}_i^{(m)} \cdot e^{d_m I(y_m(\bar{x}_i) \neq t_i)} \cdot \frac{1}{\tilde{N}_m}}{\sum_{l=1}^{N} \tilde{w}_l^{(m)} \cdot e^{d_m I(y_m(x_l) \neq t_l)} \cdot \frac{1}{\tilde{N}_m}} =$

$$= \frac{\frac{\tilde{w}_i^{(m)}}{\tilde{N}_m} \cdot e^{d_m I(y_m(\bar{x}_i) \neq t_i)}}{\sum_{l=1}^{N} \underbrace{\frac{\tilde{w}_l^{(m)}}{\tilde{N}_m}}_{w_l^{(m)}} e^{d_m I(y_m(\bar{x}_i) \neq t_l)}} \qquad \underbrace{\qquad\qquad}_{Z_m}$$

$$w_i^{(m+1)} = \frac{w_i^{(m)} \cdot e^{d_m I(y_m(\bar{x}_i) \neq t_i)}}{Z_m}$$

1. $J = \sum_{i=1}^{N} w_i^{(m)} \cdot I(y_m(\bar{x}_i) \neq t_i) = \varepsilon_m \to \min_{y_m}$
2. $d_m = \ln \frac{1 - \varepsilon_m}{\varepsilon_m}$
3. $w_i^{(m+1)} = \frac{w_i^{(m)} \cdot e^{d_m I(y_m(\bar{x}_i) \neq t_i)}}{Z_m}$

$w_i^{(1)} = \frac{1}{N}$ or $\frac{1}{2N_k}$ for imbalanced dataset.

K-Means Clustering

Задача кластеризации – отнесение элементов выборки к одному из нескольких кластеров так, чтобы элементы из одного кластера были похожи, а из разных непохожи.

Данные: $\bar{x}_1, \bar{x}_2, \dots, \bar{x}_N$; $\bar{x}_i \in R^D$

Кластеры: $C_1, C_2, \dots, C_K$

Центроиды: $\bar{\mu}_i \in R^D$, $i = 1,2,\dots,K$, то есть для каждого кластера свой центроид.

Целевая функция

$$E = \sum_{k=1}^{K} \sum_{\bar{x} \in C_k} \| \bar{\mu}_k - \bar{x} \|^2$$ — Euclidean Distance (d) $= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

Инициализация центроидов: Random, min-max range, dataset elements

Назначение элементов кластерам:

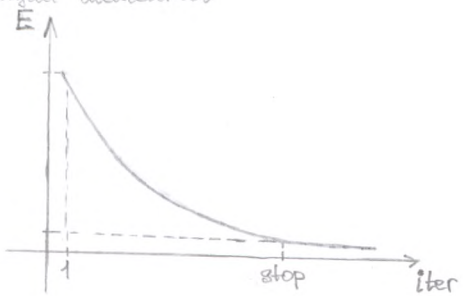
$$C_j = \{ \bar{x}_i \mid d(\bar{x}_i, \bar{\mu}_j) = \underbrace{\min_k d(\bar{x}_i, \bar{\mu}_k)}_{\text{min distance по всем}}, \; i = 1,2,\dots,N \}, \; j = 1,2,\dots,N$$

Обновление центроида

$$\bar{\mu}_j = \underbrace{\frac{1}{|C_j|}}_{\text{кол-во}} \sum_{\bar{x}_i \in C_j} \bar{x}_i$$ – среднее значение всех входящих элементов

Критерии остановки

- обновление центроидов прекратилось
- падение целевой функции почти прекратилось
- достигнуто заранее заданное кол-во итераций



$$E = \sum_{k=1}^{K} \sum_{\bar{x} \in C_k} \| \bar{\mu}_k - \bar{x} \|^2 \to \min_{\mu}$$

1. $\bar{x} \to \mu$
2. k, C_k решение по центроиду кластера принимается независимо

$$E_k = \sum_{\bar{x} \in C_k} \| \bar{\mu}_k - \bar{x} \|^2 = \sum_{\bar{x} \in C_k} \sum_{i=1}^{D} (\mu_k^{(i)} - x^{(i)})^2 \to \min_{\bar{\mu}}$$

$$\frac{dE_k}{\mu_k^{(i)}} = \sum_{\bar{x} \in C_k} 2 (\mu_k^{(i)} - x^{(i)}) = 0 .$$

$$\sum_{\bar{x} \in C_k} (\mu_k^{(i)} - x^{(i)}) = \sum_{\bar{x} \in C_k} \mu_k^{(i)} - \sum_{\bar{x} \in C_k} x^{(i)} = 0 \; ; \; \sum_{\bar{x} \in C_k} \mu_k^{(i)} = |C_k| \mu_k^{(i)} = \sum_{\bar{x} \in C_k} x^{(i)} \Rightarrow$$

$$\Rightarrow \mu_k^{(i)} = \frac{1}{|C_k|} \cdot \sum_{x \in C_k} x^{(i)} \Rightarrow \mu_j = \frac{1}{|C_j|} \cdot \sum_{\bar{x} \in C_j} \bar{x}$$

Agglomerative Clustering

$d(\bar{x}_i, \bar{x}_j) = R$ (Не всегда можно посчитать расстояние : муз. произведение).

Постепенное объединение кластеров в кластеры большего размера.

1. Single link

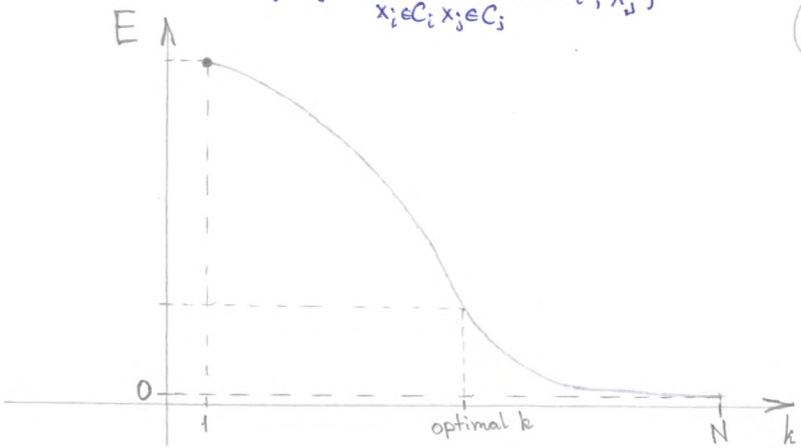
$$d(C_i, C_j) = \min_{x_i \in C_i,\ x_j \in C_j} d(x_i, x_j)$$

2. Average link

$$d(C_i, C_j) = \frac{1}{|C_i||C_j|} \sum_{x_i \in C_i} \sum_{x_j \in C_j} d(x_i, x_j)$$

3. Complete link

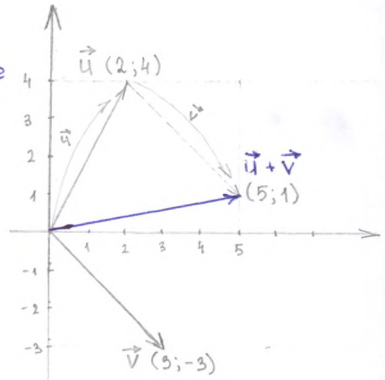
$$d(C_i, C_j) = \max_{x_i \in C_i,\ x_j \in C_j} d(x_i, x_j)$$



Basic Linear Algebra

$\vec{u} + \vec{v}$ - elementwise addition of vectors, it's like stepping in space

$c \cdot \vec{v}$ - scaling of vector



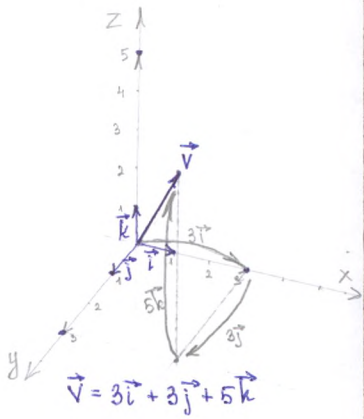
The span of $\vec{u}$ and $\vec{v}$ is a set of all linear combinations:

$$c_1\vec{u} + c_2\vec{v}$$

Linearly dependent vectors: $\vec{u} = c_1\vec{v} + c_2\vec{w}$

Basis vectors of a vector space is a set of linearly independent vectors $\{\vec{i}, \vec{j}, \vec{k}\}$ that span the full space:

$$\vec{v} = c_1\vec{i} + c_2\vec{j}$$



Linear Transformations

grid lines remain parallel and evenly spaced

$L(\vec{v})$ → transformed $\vec{v}$

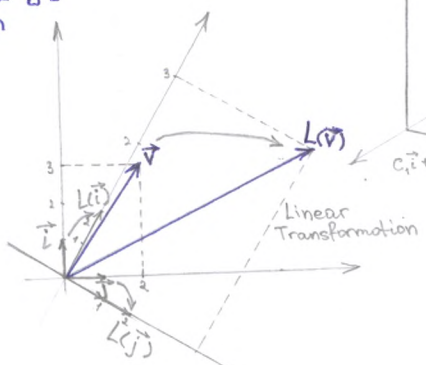
We can describe any linear transformation for all vectors using the transformed basis vectors:

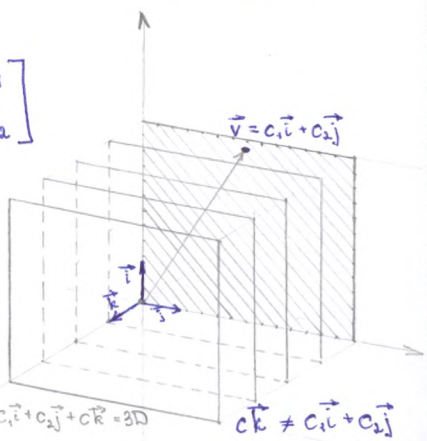
$$L(\vec{v}) = c_1 L(\vec{i}) + c_2 L(\vec{j}) = \begin{bmatrix} L(\vec{i})_x & L(\vec{j})_x \\ L(\vec{i})_y & L(\vec{j})_y \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}$$

$$L(\vec{v}) = \underbrace{\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}}_{\text{shear}} \underbrace{\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}}_{\text{rotation}} \begin{bmatrix} x \\ y \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & -1 \\ 1 & 0 \end{bmatrix}}_{\text{composition}} \begin{bmatrix} x \\ y \end{bmatrix}$$

$ABC = A(BC) = (AB)C$

$AB \neq BA$



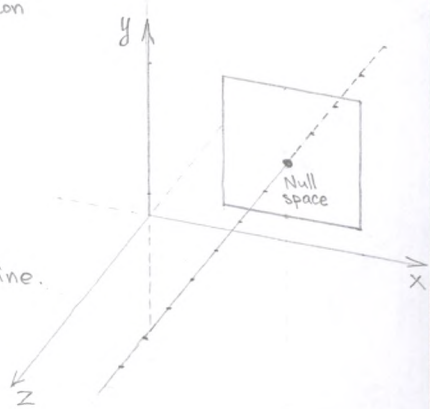


$$\begin{bmatrix} i_x & j_x \\ i_y & j_y \\ i_z & j_z \end{bmatrix} \quad 2D \to 3D \quad \text{Det} = \infty$$

Full rank 2

$$\begin{bmatrix} i_x & j_x & k_x \\ i_y & j_y & k_y \end{bmatrix} \quad 3D \to 2D \quad \text{Det} = 0$$

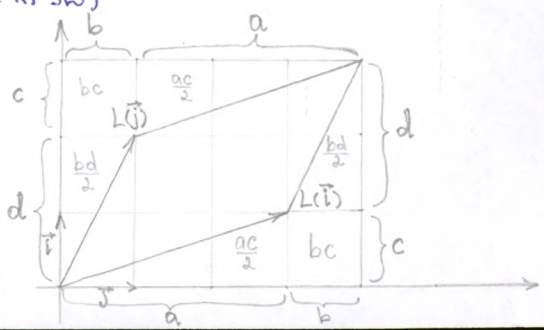
Rank = 2
Column space = 2
Null space = 1, 1 point can be re-mapped to line.
(kernel)



Determinant

determinant - scale of area of linear transformation (or volume in 3D)

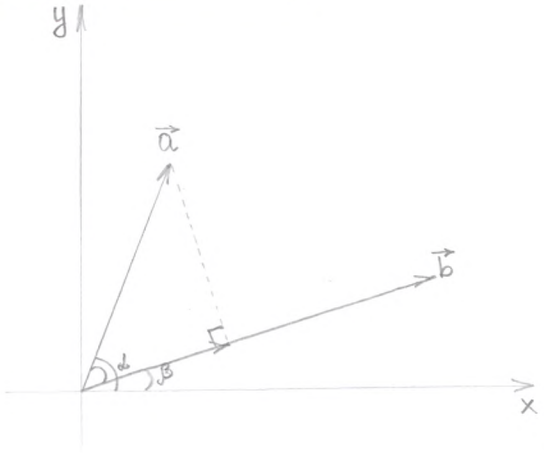
$$\det\left(\begin{bmatrix} a & b \\ c & d \end{bmatrix}\right) = \overbrace{(a+b)(c+d) - bc - bc - ac - bd}^{ac+ad+bc+bd} = ad - bc$$



Dot product of vectors

$$\vec{a} \cdot \vec{b} = |\vec{a}| \cdot |\vec{b}| \cdot \cos(\hat{\vec{a}, \vec{b}})$$

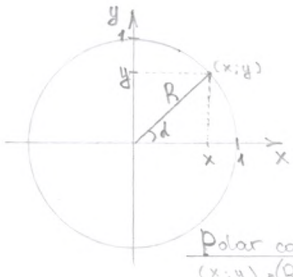
$$\vec{a} \cdot \vec{b} = \vec{a}^T \vec{b} = \vec{b}^T \vec{a} = \begin{bmatrix} i & j \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = xi + yj$$



$$\vec{a} = |\vec{a}| \cdot (\cos\alpha ; \sin\alpha)^T$$
$$\vec{b} = |\vec{b}| \cdot (\cos\beta ; \sin\beta)^T$$

$$\vec{a}^T \vec{b} = |\vec{a}||\vec{b}| \cos\alpha \cos\beta + |\vec{a}||\vec{b}| \sin\alpha \sin\beta =$$
$$= |\vec{a}||\vec{b}| \cos(\alpha - \beta)$$

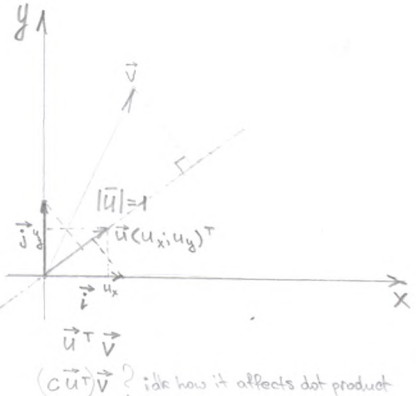
$$\boxed{\cos(\alpha - \beta) = \cos\alpha \cos\beta + \sin\alpha \sin\beta}$$



$$\cos\alpha = \frac{x}{R}$$
$$\sin\alpha = \frac{y}{R}$$
$$\tan\alpha = \frac{\sin\alpha}{\cos\alpha} = \frac{y}{x}$$
$$ctg = \frac{x}{y}$$

Polar coordinate

$(x;y) = (\underbrace{R}_{\text{length of a vector}} \cdot \cos\alpha ; R \cdot \sin\alpha)$



$(c\vec{u}^T)\vec{v}$? idk how it affects dot product

Taylor Series

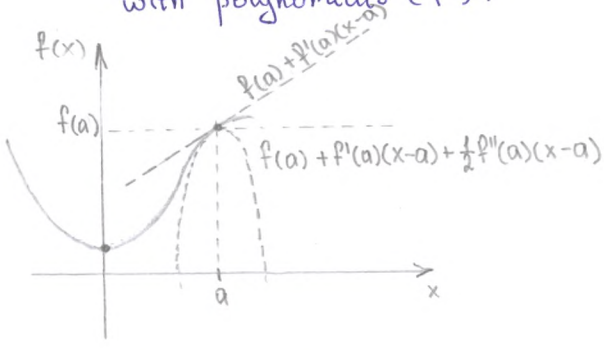
$$f(x) = P(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n$$

Approximation of non-polynomial functions (f) with polynomials (P).

$$P^{(n)}(a) = f^{(n)}(a)$$

Значения функций в точке a и производные должны совпадать.

(изменения изменений роста функции)



$$P(x) = c_0 + c_1(x-a) + c_2(x-a)^2 + c_3(x-a)^3 + c_4(x-a)^4 \dots$$

0) $P(a) = c_0 + c_1\underbrace{(a-a)}_{0} = c_0 = f(a)$

1) $P^{(1)}(a) = (0 + c_1(1) + 2c_2(a-a) + \dots) = c_1 = f'(a)$

2) $P^{(2)}(a) = 0 + 0 + 2c_2 + 3\cdot 2\cdot c_3\underbrace{(a-a)}_{0} + 4\cdot 3\cdot c_4\underbrace{(a-a)^2}_{0} + \dots = 2c_2 = f^{(2)}(a)$

$\Rightarrow c_2 = \frac{f^{(2)}(a)}{2}$

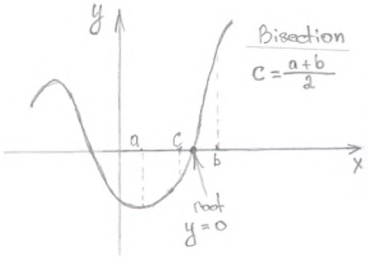
3) $P^{(3)}(a) = 0 + 0 + 0 + 3\cdot 2\cdot 1\, c_3 + 4\cdot 3\cdot 2\cdot c_4\underbrace{(a-a)}_{0} + \dots = 3\cdot 2\cdot 1\cdot c_3 = f^{(3)}(a)$

$\Rightarrow c_3 = \frac{f^{(3)}(a)}{3!}$

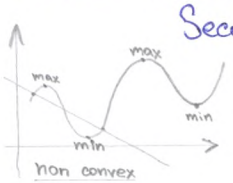
$$P(x) = f(a) + f'(a)(x-a) + \frac{1}{2}f^{(2)}(a)(x-a)^2 + \frac{1}{6}f^{(3)}(a)(x-a)^3 + \frac{1}{24}f^{(4)}(a)(x-a)^4 + \dots$$

Gradient
Jacobian
Hessian

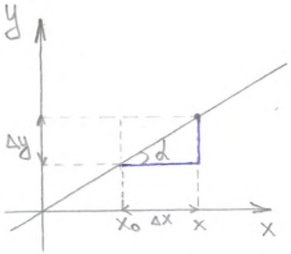
Newton's Method



1. Root-finding method. Other: Bisection (Interval halving), Secant, Brent's methods
2. Optimization method



Derivative



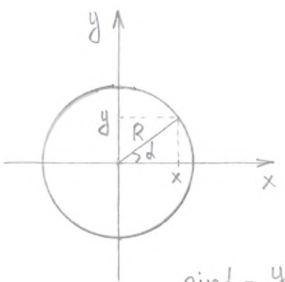
$$f'(x) = \frac{\Delta y}{\Delta x} = tg\alpha$$

$$y = f(x)$$

$$f'(x) = \frac{df}{dx} = \lim_{\Delta x \to 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

$$tg\alpha = \frac{\Delta y}{\Delta x} = v$$

$$f'(x_0) = \lim_{\Delta x \to 0} \frac{f(x_0+\Delta x) - f(x_0)}{\Delta x}$$



Tanget line

$$f'(x_0) = \lim_{\Delta x \to 0} \frac{f(x_0+\Delta x) - f(x_0)}{\Delta x}$$

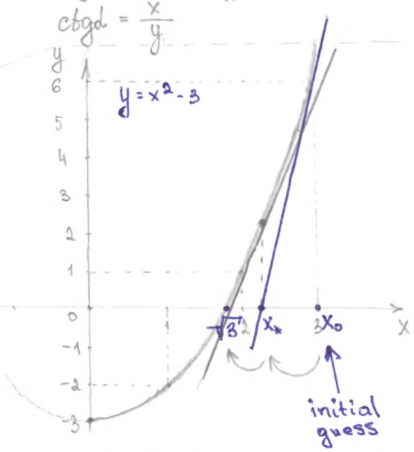
$$\Delta x = x - x_0$$

$$f'(x_0) = \frac{f(x_0+(x-x_0)) - f(x_0)}{x - x_0} \Rightarrow f'(x_0)(x-x_0) = f(x) - f(x_0)$$

$$f(x) = f(x_0) + f'(x_0)(x-x_0)$$

$S_0 + v \cdot \Delta t$

$$\sin\alpha = \frac{y}{R}$$
$$\cos\alpha = \frac{x}{R}$$
$$tg\alpha = \frac{\sin\alpha}{\cos\alpha} = \frac{y}{x}$$
$$ctg\alpha = \frac{x}{y}$$



Newton's root finding method (iterative)

$$f'(x_0) = \frac{f(x) - f(x_0)}{x - x_0} \Rightarrow x - x_0 = \frac{f(x) - f(x_0)}{f'(x_0)}, \quad f'(x) \neq 0$$

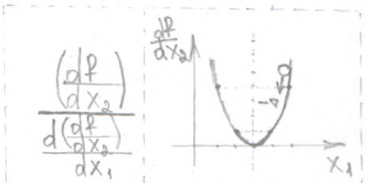
$f(x) = 0$ - root

$$\boxed{x_* = x_0 - \frac{f(x_0)}{f'(x_0)}}, \text{ until } |x_{n+1} - x_n| < \varepsilon;$$

Newton's optimization method

$y'(x) = 0$ - critical points (minima/maxima), so we cantry to find roots of $y'(x)$, i.e. $y'(x) = 0$.

$$\boxed{x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)}}$$



Multidimentional case:

$$\vec{x}_1 = \vec{x}_0 - \underbrace{H_{x_0}^{-1}}_{\text{Hessian Matrix}} \nabla f(\vec{x}_0)$$

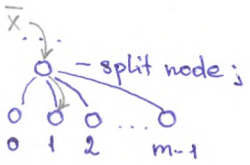
$$\begin{bmatrix} \frac{d^2f}{dx_1^2} & \frac{d^2f}{dx_1dx_2} \cdots \frac{d^2f}{dx_1dx_D} \\ \frac{d^2f}{dx_2dx_1} & \frac{d^2f}{dx_2^2} \cdots \frac{d^2f}{dx_2dx_D} \\ \cdots \\ \frac{d^2f}{dx_Ddx_1} & \frac{d^2f}{dx_Ddx_2} \cdots \frac{d^2f}{dx_D^2} \end{bmatrix}^{-1} \begin{bmatrix} \frac{df}{dx_1} \\ \frac{df}{dx_2} \\ \cdots \\ \frac{df}{dx_D} \end{bmatrix} = \begin{bmatrix} \frac{df}{dx_1}\frac{d^2f}{dx_1^2} + \frac{df}{dx_2}\frac{d^2f}{dx_1dx_2} + \cdots + \frac{df}{dx_D}\frac{d^2f}{dx_1dx_D} \\ \cdots \\ \frac{df}{dx_1}\frac{d^2f}{dx_Ddx_1} = \frac{df}{dx_1}\frac{d(\frac{df}{dx_1})}{dx_D} + \cdots \end{bmatrix}$$

H^{-1} $\quad \nabla f$

f twice-differentiable.

Decision Tree for Regression Task

- target variables : $t_1, t_2, \dots, t_N \in \mathbb{R}$ $\quad t(\bar{x}) = y(\bar{x}) + \varepsilon$
- sample : $\bar{x}_i \in \mathbb{R}^D$
- split function : $h(\bar{x}, \bar{\theta}_j) : \mathbb{R}^D \times T \to \{0, 1, \dots, m-1\}$ (args, Domain, Range)
  - $\bar{\theta}_j \in T$ - split function parameters
  - usually $m = 2$ i.e. Binary Decision Tree



- feature selection function : $\bar{\psi}(\bar{x}) : \mathbb{R}^D \to \mathbb{R}^{D'}$, $D' \le D$
  - usually $D' = 1$ i.e. $\psi(\bar{x}) = x_i$

$\bar{x} = [x_1\ x_2\ \dots\ x_D]^T$
$\bar{\psi}(\bar{x}) = [x_2, x_5]^T$, $D' = 2$

- Split function types remain the same, for example, default hyperplane :

$$h(\bar{x}, \theta_j) = [\tau_1 > \psi(\bar{x}) > \tau_2]$$
$$\theta_j = (\psi, \tau_1, \tau_2)$$
$$\psi(\bar{x}) = x_i$$

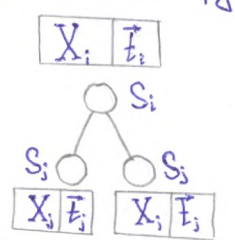
Indicator function
$$[a] = \begin{cases} 1, \text{ if } a = \text{true} \\ 0, \text{ if } a = \text{false} \end{cases}$$

Loss function

$$L = V(S_i) - \underbrace{\sum_j \frac{N_{ij}}{N_i} V(S_j)}_{\min} \to \max_\theta$$

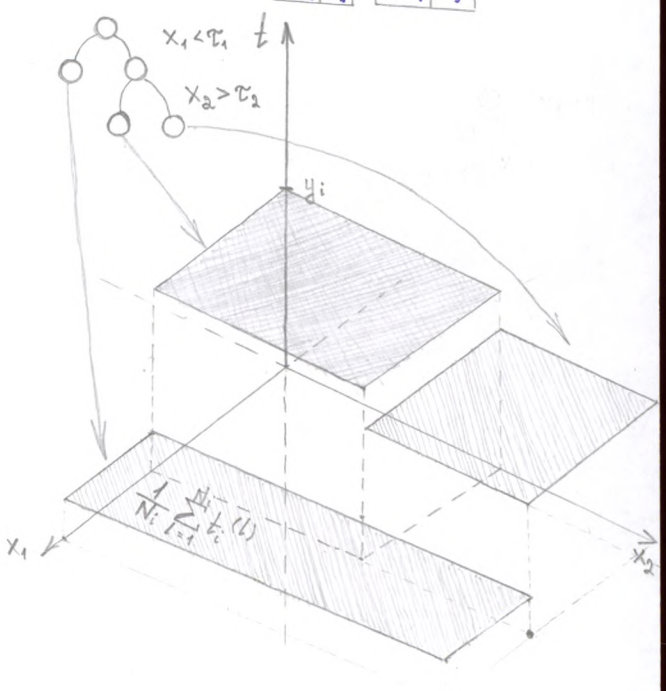
(we use variance instead of entropy)

$$V(S_i) = \frac{1}{N_i} \sum_{l=1}^{N_i} (t_i^{(l)} - \underbrace{m_i}_{\text{computed only once}})^2, \quad \underbrace{m_i}_{\text{prediction}} = \frac{1}{N_i} \sum_{l=1}^{N_i} t_i^{(l)}$$



Stopping criteria to create terminal node

- depth d has been reached
- $N_i < n$
- $V(S_i) < \varepsilon$



Terminal node (prediction)

$$y_i = \frac{1}{N_i} \sum_{l=1}^{N_i} t_i^{(l)} \quad \text{- average}$$

additionally, we could return variance.

Gradient Boosting : Regression

Объединение слабых регрессоров в сильный регрессор.

Weak regressor – Decision Tree for Regression Task.
(weak learner in general)

Итоговый регрессор

- $F(\bar{x}) = \sum_{j=0}^{M} h_j f_j(\bar{x}) = f_0 + \sum_{j=1}^{M} h f_j(\bar{x})$ – сумма слабых регрессоров, которые исправляют ошибки (residuals).

  $h_0 = 1$ $\quad f_0$ – const

  $h_j = h \in [0;1]$ as learning rate

Loss Function (any differentiable loss function)

L2 or L1

$L(F_m(\bar{x}), t) = 0 \iff F_m(\bar{x}_i) = t_i, \forall i$

if $\exists i : F_m(\bar{x}_i) \neq t_i \Rightarrow L > 0$

but $\bar{x}_i = \bar{x}_j$ and $t_i \neq t_j \Rightarrow f(\bar{x}_i) = \frac{t_i + t_j}{2}$

- $L_2 = \frac{1}{2} \sum_{i=1}^{N} (F(\bar{x}_i) - t_i)^2$

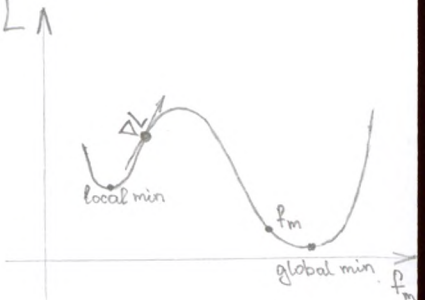
- $L \to \min_{f_0}$; $\left(\frac{1}{2} \sum_{i=1}^{N} (f_0 - t_i)^2 \right)'_{f_0} = \sum_{i=1}^{N} (f_0 - t_i) = N f_0 - \sum_{i=1}^{N} t_i = 0 \Rightarrow \boxed{f_0 = \frac{1}{N} \sum_{i=1}^{N} t_i}$

- $L \to \min_{f_m}$; $L = \frac{1}{2} \sum_{i=1}^{N} (F_m(\bar{x}_i) - t_i)^2 = \frac{1}{2} \sum_{i=1}^{N} (F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i) - t_i)^2 =$

  $= \frac{1}{2} \sum_{i=1}^{N} (f_m(\bar{x}_i) - \underbrace{(t_i - F_{m-1}(\bar{x}_i))}_{\tilde{t}_i^{(m)} \text{ – residuals}})^2 = \boxed{\frac{1}{2} \sum_{i=1}^{N} (f_m(\bar{x}_i) - \tilde{t}_i^{(m)})^2}$

У каждого слабого регрессора f_m свой набор целевых переменных $\tilde{t}^{(m)}$ – residuals, соответственно каждый регрессор будет разный.

Gradient Boosting напоминает Gradient Descent ($f_m \sim (-\nabla L)$, $h \sim lr$), и стоит отметить, что f_m указывает global minima.



Потенциально f_m может overfit-нуться, тут идея похожа на ансамбли деревьев в Random Forest.

Gradient Boosting: Classification

Итоговый классификатор

$$y(\bar{x}) = \underbrace{\sigma}_{\text{sigmoid}}(F(\bar{x})) = \sigma(\underbrace{f_0}_{\text{const}} + \alpha \sum_{j=1}^{M} f_j(\bar{x})), \quad f_j \text{ - regression tree}$$

Sigmoid (σ)

$$\text{Softmax}(\bar{x}) = \left(\frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}\right), \quad K = 2$$

$$\text{Softmax}(\bar{x}) = \left(\frac{e^{x_0}}{e^{x_0}+e^{x_1}}, \frac{e^{x_1}}{e^{x_0}+e^{x_1}}\right) = \left(\frac{1}{1+e^{x_1-x_0}}, \frac{1}{e^{x_0-x_1}+1}\right) = \left| x_1 - x_0 = t \right| =$$

$$= \left(\underbrace{\frac{1}{1+e^{t}}}_{1-\sigma}, \underbrace{\frac{1}{1+e^{-t}}}_{\sigma}\right)$$

$$\frac{1}{1+e^{t}} = \frac{1+e^{t}-e^{t}}{1+e^{t}} = 1 - \frac{e^{t}}{1+e^{t}} = 1 - \frac{1}{e^{-t}+1} = 1 - \sigma(t) = \sigma(-t)$$

Loss function

$\vec{y} = \begin{bmatrix} 30\% \\ 70\% \end{bmatrix}$ - probability vector

$$P(\vec{t} \mid X) = \prod_{n=1}^{N} \prod_{k=1}^{K} y^{(k)}(\bar{x}_n)^{t_n^{(k)}} \to \max$$

$$L = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_n^{(k)} \ln y^{(k)}(\bar{x}_n) \to \min, \quad K = 2$$

$$L = -\sum_{n=1}^{N} \sum_{k=1}^{2} t_n^{(k)} \ln y^{(k)}(\bar{x}_n) = -\sum_{n=1}^{N} \left(t_n^{(0)} \ln y^{(0)}(\bar{x}_n) + t_n^{(1)} \ln y^{(1)}(\bar{x}_n) \right), \quad \sigma(x) \in [0;1] \text{ - scalar}, \quad t_n \in \{0;1\}$$

$$L = -\sum_{n=1}^{N} \left((1-t_n) \ln(1 - y(\bar{x}_n)) + t_n \ln y(\bar{x}_n) \right) \to \min$$

$t_n = c_0$ $t_n = c_1$ Один из них обнуляется

$$y(\bar{x}) = \sigma(f_0)$$

$$L = -\sum_{n=1}^{N} \left((1-t_n) \ln(1 - \sigma(f_0)) + t_n \ln \sigma(f_0) \right) \to \min_{f_0}$$

$$L' = -\sum_{n=1}^{N} \left((1-t_n) \frac{1}{1-\sigma(f_0)} \cdot (-\sigma(f_0))' + t_n \frac{1}{\sigma(f_0)} \cdot (\sigma(f_0))' \right) =$$

$$= -\sum_{n=1}^{N} \left((1-t_n) \frac{1}{1-\sigma(f_0)} \cdot (-\sigma(f_0)(1-\sigma(f_0))) + t_n \frac{1}{\sigma(f_0)} \cdot \sigma(f_0)(1-\sigma(f_0)) \right) =$$

$$= -\sum_{n=1}^{N} \left((1-t_n) \cdot (-\sigma(f_0)) + t_n (1-\sigma(f_0)) \right) = -\sum_{n=1}^{N} \left(-\sigma(f_0) + t_n \sigma(f_0) + t_n - t_n \sigma(f_0) \right) =$$

$$= -\sum_{n=1}^{N} \left(-\sigma(f_0) + t_n \right) = 0 \quad \text{- is it minimum? proof} \to \text{if } f'' = +C$$

$$\left(\sum_{n=1}^{N} \sigma(f_0) - t_n \right)' = \sum_{n=1}^{N} \underbrace{\sigma(f_0)}_{\geq 0} \underbrace{(1-\sigma(f_0))}_{\geq 0} = +C$$

$$\sum_{n=1}^{N} \sigma(f_0) = \sum_{n=1}^{N} t_n = N \sigma(f_0)$$

$$\sigma(f_0) = \frac{1}{N} \sum_{n=1}^{N} t_n = \frac{1}{1+e^{f_0}} = p_1 \Rightarrow$$

$$1 + e^{f_0} = \frac{1}{p_1}$$

$$e^{f_0} = \frac{1}{p_1} - 1 = \frac{1-p_1}{p_1} = \frac{p_0}{p_1}$$

$$\boxed{f_0 = \ln \frac{p_0}{p_1}}$$

bias, dataset balance

0. $(a^x)' = a^x \ln a \cdot (x)'$

1. $(\log_a f(x))' = \frac{1}{f(x) \ln a} \cdot f'(x)$

2. $\sigma'(x) = (1 \cdot (1+e^{-x})^{-1})' = 0 + (-1(1+e^{-x})^{-2} \cdot (e^{-x})') =$

$$= -\frac{1}{(1+e^{-x})^2} \cdot (e^{-x}) \cdot (-x)' =$$

$$= \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1 \cdot (e^{-x})}{(1+e^{-x})(1+e^{-x})} = \sigma(x) \cdot \frac{e^{-x}+1-1}{1+e^{-x}} =$$

$$= \sigma(x) \cdot \frac{1+e^{-x}-1}{1+e^{-x}} = \boxed{\sigma(x)(1-\sigma(x))}$$

Gradient Boosting: Classification (2)

Итоговый классификатор

$$y(\bar{x}) = \sigma(F(\bar{x})) = \sigma\left(f_0 + \alpha \sum_{j=1}^{M} f_j(\bar{x})\right), \quad f_j \text{ - regression tree}$$

$$\sigma(f_0) = \frac{1}{N}\sum_{n=1}^{N} t_n \longleftrightarrow f_0 = \ln\frac{p_0}{p_1}$$

Loss function

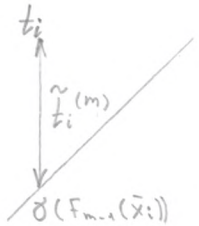
$$L = -\sum_{n=1}^{N}\left((1-t_n)\left(\ln(1-y(\bar{x}_n)) + t_n \ln y(\bar{x}_n)\right)\right) \to \min$$

Given: $f_0, f_1, \dots, f_{m-1}$

f_m - ?

$$\begin{cases} F_m(\bar{x}_i) = F_{m-1}(\bar{x}_i) + \alpha f_m(\bar{x}_i) \\ F_m(\bar{x}_i) = F_{m-1}(\bar{x}_i) - \gamma L'\left(\sigma(F_{m-1}(\bar{x}_i))\right)_{F_{m-1}} \end{cases}$$

$f_m = -L'$



$\tilde{t}_i^{(m)}$ - (residuals)

$t_i \in \{0,1\}$

$\sigma(F(\bar{x})) \in [0;1]$

$\Rightarrow \tilde{t}_i^{(m)} \in [-1;1]$

$L'\left(\sigma(F_{m-1}(\bar{x}_i))\right)_{F_{m-1}}$ - скорость роста целевой функции L в точке $F_{m-1}(\bar{x}_i)$, и чтобы уменьшить L нужно смещать $F_{m-1}(\bar{x}_i)$ в обратном направлении.

$$-L'\left(\sigma(F_{m-1}(\bar{x}_i))\right)_{F_{m-1}} = \left((1-t_i)\ln(1-y(\bar{x}_i)) + t_i \ln y(\bar{x}_i)\right)'_{F_{m-1}} =$$

$\left(F_{m-1}(\bar{x}_i)\right)'_{F_{m-1}} = 1$, as $x' = 1$

$$= \left((1-t_i)\ln(1-\sigma(F_{m-1}(\bar{x}_i))) + t_i \ln \sigma(F_{m-1}(\bar{x}_i))\right)' =$$

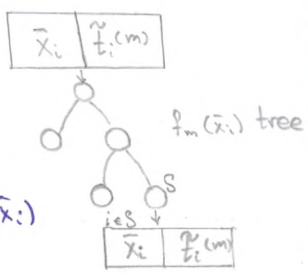
$$= (1-t_i)\frac{1}{(1-\sigma(F_{m-1}(\bar{x}_i)))}\cdot\left(-\sigma(F_{m-1}(\bar{x}_i))\right)\left(1-\sigma(F_{m-1}(\bar{x}_i))\right) + t_i \frac{1}{\sigma(F_{m-1}(\bar{x}_i))}\cdot\sigma(F_{m-1}(\bar{x}_i))\left(1-\sigma(F_{m-1}(\bar{x}_i))\right) =$$

$$= (1-t_i)\left(-\sigma(F_{m-1}(\bar{x}_i))\right) + t_i\left(1-\sigma(F_{m-1}(\bar{x}_i))\right) =$$

$$= -\sigma(F_{m-1}(\bar{x}_i)) + t_i\,\sigma(F_{m-1}(\bar{x}_i)) + t_i - t_i\,\sigma(F_{m-1}(\bar{x}_i)) =$$

$$= t_i - \sigma(F_{m-1}(\bar{x}_i)) = \tilde{t}_i^{(m)} \text{ - label for } f_m(\bar{x}_i) \text{ tree}$$

$$\boxed{\tilde{t}_i^{(m)} = t_i - \sigma(F_{m-1}(\bar{x}_i))}$$



we try to approximate $\tilde{t}_i^{(m)}$ by the underfitting weak regressor tree $f_m(\bar{x}_i)$ but even if it will approximate 100% i.e. $f_m(\bar{x}_i) = \tilde{t}_i^{(m)}$, $\forall i$

$\boxed{\sigma(F_{m-1}(\bar{x}_i) + \tilde{t}_i^{(m)}) \neq t_i}$, it does not guarantee convergence $t_i \neq \sigma(F(\bar{x}_i)), \forall i$

$\tilde{t}_i^{(m)}$ - это направление. Пока что мы пытаемся сделать так, чтобы в дереве наиболее похожие residuals были в одном terminal узле.

$L'\left(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))\right)_{f_m} = 0$ we can't just return average of $\tilde{t}_i^{(m)}$, too slow convergence

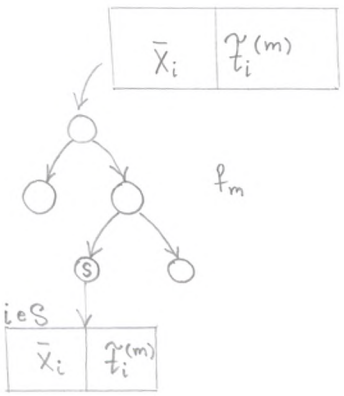
Gradient Boosting: Classification (3)

$y(\bar{x}) = \sigma(F(\bar{x})) = \sigma(f_0 + \alpha \sum_{j=1}^{M} f_j(\bar{x}))$, f_j - regression tree

$\sigma(f_0) = \frac{1}{N}\sum_{n=1}^{N} t_n \longleftrightarrow f_0 = \ln\frac{p_0}{p_1}$

Given : $f_0, f_1, \dots, f_{m-1}$

f_m - ?



$$\begin{cases} F_m(\bar{x}_i) = F_{m-1}(\bar{x}_i) + \alpha f_m(\bar{x}_i) \\ F_m(\bar{x}_i) = F_{m-1}(\bar{x}_i) - \gamma L'(\sigma(F_{m-1}(\bar{x}_i)))_{F_{m-1}} \end{cases}$$

$f_m(\bar{x}_i) = -L'(\sigma(F_{m-1}(\bar{x}_i)))_{F_{m-1}}$

$-L'(\sigma(F_{m-1}(\bar{x}_i)))_{F_{m-1}} = t_i - \sigma(F_{m-1}(\bar{x}_i)) = \tilde{t}_i^{(m)}$ residuals, we try to minimize Variance at the nodes of the f_m regression tree at these values

$\tilde{t}_i^{(m)}$ - direction for minimising L

$\sigma(F_{m-1}(\bar{x}_i) + \tilde{t}_i^{(m)}) \neq t_i$, so we can't just return average on terminal node of f_m tree.

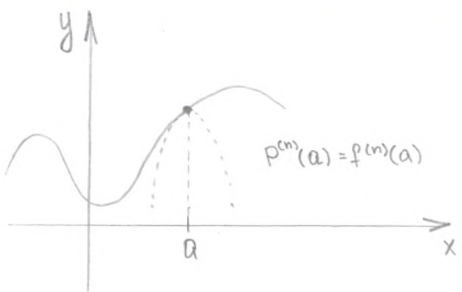
$$L(\sigma(\underbrace{F_{m-1}(\bar{x})}_{\text{const}} + \underbrace{f_m(\bar{x})}_{\text{var}})) = -\sum_{n=1}^{N}\left((1-t_n)\ln(1-\sigma(F_{m-1}(\bar{x}_n)+f_m(\bar{x}_n))) + t_n \ln\sigma(F_{m-1}(\bar{x}_n)+f_m(\bar{x}_n))\right) \to \min_{f_m}$$

we can't take the derivative of f_m, because we would need f_m in σ but we don't know it.

Taylor Series

$f(x) = P(x) = \sum_{n=0}^{N} \frac{f^{(n)}(a)}{n!}(x-a)^n$

$P(\underbrace{x}_{0}+a) = \sum_{n=0}^{N} \frac{f^{(n)}(a)}{n!}((x+a)-a)^n$



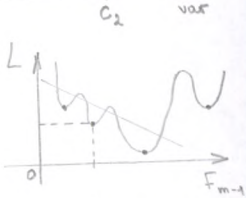
$f = L$, $a = F_{m-1}$, $x = f_m$

$$L(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))) \approx \sum_{n=0}^{2} \frac{L^{(n)}(F_{m-1})}{n!} f_m^n = \underbrace{L(\sigma(F_{m-1}(\bar{x}_i)))}_{C_0} + \underbrace{L^{(1)}(\sigma(F_{m-1}(\bar{x}_i)))}_{C_1} \underbrace{f_m}_{\text{var}} + \underbrace{\frac{1}{2}L^{(2)}(\sigma(F_{m-1}(\bar{x}_i)))}_{C_2} \underbrace{f_m^2}_{\text{var}}$$

$$L'_{f_m}(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))) \approx L^{(1)}(\sigma(F_{m-1}(\bar{x}_i))) + L^{(2)}(\sigma(F_{m-1}(\bar{x}_i)))\, f_m = 0$$

on terminal node

$$f_m = \frac{-L^{(1)}(\sigma(F_{m-1}(\bar{x}_i)))_{F_{m-1}}}{L^{(2)}(\sigma(F_{m-1}(\bar{x}_i)))_{F_{m-1}}}$$



- Looks like Newton's Optimization Method and it is

$L' = 0$ - minima, we try to find roots of L'

but it's not guaranteed to be global minima, because L is not convex function

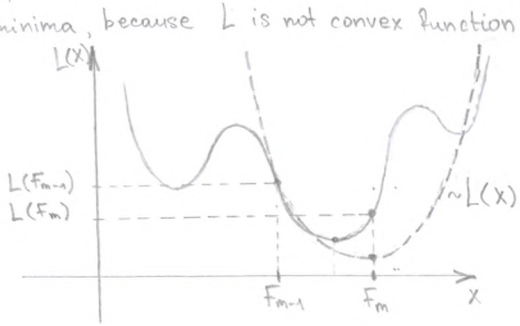
$F_m(\bar{x}_i) = F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i)$

$\sim L'(F_m(\bar{x}_i)) = 0$ local minimum with approximation error

$\sim L(F_m(\bar{x}_i)) > 0$ but minima

$L'(F_m(\bar{x}_i)) = \emptyset$

$L(F_m(\bar{x}_i)) > 0$ close to minima



Gradient Boosting: Classification (4)

$$y(\bar{x}) = \sigma(F(\bar{x})) = \sigma\left(f_0 + \alpha \sum_{j=1}^{M} f_j(\bar{x})\right)$$

$$\sigma(f_0) = \frac{1}{N}\sum_{n=1}^{N} t_n \longleftrightarrow f_0 = \ln\frac{p_0}{p_1}$$

$$\tilde{t}_i^{(m)} = t_i - \sigma(F_{m-1}(\bar{x}_i))$$

$$f_m(\bar{x}) = -\frac{L'(\sigma(F_{m-1}(\bar{x}_i)))}{L''(\sigma(F_{m-1}(\bar{x}_i)))}, \quad \forall i \in S$$

$$L(\sigma(F_{m-1}(\bar{x}))) = -\sum_{n=1}^{N}\left((1-t_n)\ln(1-\sigma(F_{m-1}(\bar{x}_n))) + t_n \ln\sigma(F_{m-1}(\bar{x}_n))\right)$$

$$L'_{F_{m-1}}(\sigma(F_{m-1}(\bar{x}))) = -\sum_{\substack{n=1 \\ n\in S}}^{N}\left((1-t_n)\frac{1}{1-\sigma(F_{m-1}(\bar{x}_n))}\cdot(-\sigma(F_{m-1}(\bar{x}_n))(1-\sigma(F_{m-1}(\bar{x}_n)))) + t_n\frac{1}{\sigma(F_{m-1}(\bar{x}_n))}\cdot\sigma(F_{m-1}(\bar{x}_n))(1-\sigma(F_{m-1}(\bar{x}_n)))\right) =$$

$$= -\sum_{n=1}^{N}\left((1-t_n)(-\sigma(F_{m-1}(\bar{x}_n))) + t_n(1-\sigma(F_{m-1}(\bar{x}_n)))\right) =$$

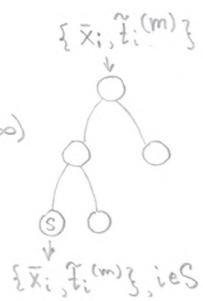
$$= -\sum_{n=1}^{N}\left(-\sigma(F_{m-1}(\bar{x}_n)) + t_n\sigma(F_{m-1}(\bar{x}_n)) + t_n - t_n\sigma(F_{m-1}(\bar{x}_n))\right) =$$

$$= \sum_{n=1}^{N}\left(\sigma(F_{m-1}(\bar{x}_n)) - t_n\right)$$

$$L''_{F_{m-1}}(\sigma(F_{m-1}(\bar{x}))) = \sum_{\substack{n=1 \\ n\in S}}^{N}\sigma(F_{m-1}(\bar{x}_n))(1-\sigma(F_{m-1}(\bar{x}_n)))$$

$$f_m(\bar{x}) = \frac{\sum_{n\in S}(t_n - \sigma(F_{m-1}(\bar{x}_n)))}{\sum_{n\in S}\sigma(F_{m-1}(\bar{x}_n))(1-\sigma(F_{m-1}(\bar{x}_n)))} = \frac{\sum_{n\in S}\tilde{t}_i^{(m)}}{\sum_{n\in S}\sigma(F_{m-1}(\bar{x}_n))(1-\sigma(F_{m-1}(\bar{x}_n)))}$$

$\in [0;\infty)$; $\in (0;0{,}25]$



$\sim L'(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))) = 0$

$\sim L(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))) > 0$ but minima of $\sim L$

$L'(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))) = \emptyset$

$L(\sigma(F_{m-1}(\bar{x}_i) + f_m(\bar{x}_i))) > 0$ somewhere close to minima according to approximated $\sim L$

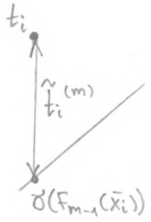
Gradient Boosting: Classification. Summary.

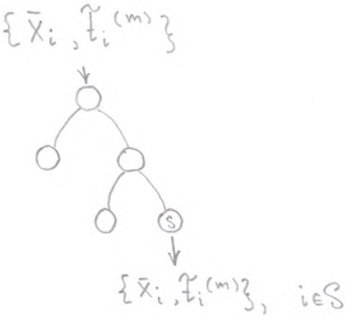
$$y(\bar{x}) = \sigma(F(\bar{x})) = \sigma\left(f_0 + \alpha \sum_{j=1}^{M} f_j(\bar{x})\right)$$

$$f_0 = \frac{p_0}{p_1} \longleftrightarrow \sigma(f_0) = \frac{1}{N}\sum_{n=1}^{N} t_n$$

$$\tilde{t}_i^{(m)} = t_i - \sigma(F_{m-1}(\bar{x}_i)) \quad \in [-1; 1]$$

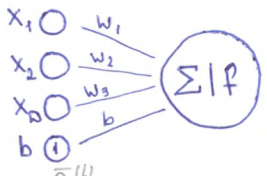
$$f_m(\bar{x}) = \frac{\sum_{i \in S} (t_i - \sigma(F_{m-1}(\bar{x}_i))}{\sum_{i \in S} \sigma(F_{m-1}(\bar{x}_i))(1 - \sigma(F_{m-1}(\bar{x}_i)))} \quad \in [0; \infty)$$





Multilayer Perceptron (MLP)

Neuron



$$\longrightarrow f\left(b + \sum_{i=1}^{D} x_i w_i\right)$$

$\bar{a}_i^{(l)}$

$$z_i^{(l)} = b_i^{(l)} + \sum_{i=1}^{M_l} x_i w_i = b_i^{(l)} + \vec{w}_i^T \cdot \vec{x}$$

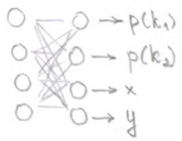
f - activation function:

- Sigmoid: $f(a) = \frac{1}{1+e^{-a}}$
- Tanh: $f(a) = \frac{e^{2a}-1}{e^{2a}+1}$
- ReLu: $f(a) = \max(0; a)$
- Leaky ReLU: $f(a) = \max(\lambda a; a)$

Layer

$$\begin{bmatrix} \vec{w}_1^T \\ \vec{w}_2^T \\ \vec{w}_3^T \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \vec{w}_1^T \vec{x} \\ \vec{w}_2^T \vec{x} \\ \vec{w}_3^T \vec{x} \end{bmatrix}$$

$W \quad \vec{x} \quad \vec{a}$



M_l - number of neurons

$$\vec{z}^{(l)} = \vec{f}(\vec{a}^{(l)}) = \vec{f}(\vec{a}^{(l)}) = \vec{f}(W^{(l)} \cdot \vec{z}^{(l-1)} + \vec{b}^{(l)}) =$$

$$= \vec{f}(\vec{b}^{(l)} + W^{(l)} \cdot \vec{f}(\vec{b}^{(l-1)} + W^{(l-1)} \cdot \vec{f}(\ldots, \vec{x})))$$

$$Y(\vec{x}, \mathbb{W}) = \text{Softmax}(\vec{z}^{(L)}(\vec{x}, \mathbb{W}))$$

$z_i > z_j \rightarrow y_i > y_j$

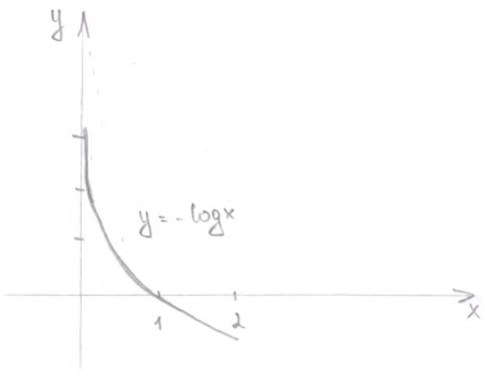
Loss function

$$L(\mathbb{W}) = -\sum_{i=1}^{N}\sum_{k=1}^{K} t_i^{(k)} \ln Y_k(\vec{x}_i, \mathbb{W})$$

$$t_i^{(k)} = \begin{cases} 1, & \text{if } k = t \\ 0, & \text{otherwise} \end{cases}$$

$\vec{t} = (0, 0, 1, 0, 0..)^T$

$$L(\mathbb{W}, X_{tr}, T_{tr}) \rightarrow \min_{\mathbb{W}}$$



1. Weights initialization

$$w_i \sim \mathcal{N}\left(\mu = 0, \sigma^2 = \frac{2}{n_i}\right)$$

2. $w_n = w_{n-1} - \gamma \nabla L(w_{n-1})$

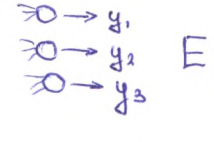
3. Stop criteria

MLP Backprop (Jacobian)

$E(\bar{y}(\bar{x}))$ $\nabla_E E = 1$

$\bar{y}(\bar{x})$, $\nabla_y E = \left[\frac{dE}{dy_1} \; \frac{dE}{dy_2} \; \dots \; \frac{dE}{dy_M}\right]^T$

$$\bar{y}(\bar{x}) = \begin{bmatrix} y_1(\bar{x}) = b_1 + \sum_{j=1}^{M} w_{j1} x_j \\ y_2(\bar{x}) = b_2 + \sum_{j=1}^{M} w_{j2} x_j \\ y_3(\bar{x}) = b_3 + \sum_{j=1}^{M} w_{j3} x_j \end{bmatrix}$$



$\nabla_x E$ - ?

$$\nabla_x E = \left[\underbrace{\sum_{i=1}^{M} \frac{dE}{dy_i}\frac{dy_i}{dx_1}}_{\text{Why sum?}} \quad \sum_{i=1}^{M} \frac{dE}{dy_i}\frac{dy_i}{dx_2} \; \dots\right]^T$$

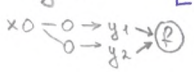
$$\frac{dE}{dx_1} = \underbrace{\frac{dE}{dy_1}\cdot\frac{dy_1}{dx_1}} + \underbrace{\frac{dE}{dy_2}\cdot\frac{dy_2}{dx_1}} + \underbrace{\frac{dE}{dy_3}\cdot\frac{dy_3}{dx_1}}$$

$$\text{Jacobian} = \frac{d\bar{y}}{d\bar{x}} = \left[\left[\frac{d\bar{y}}{dx_1}\right] \left[\frac{d\bar{y}}{dx_2}\right] \dots\right] = \begin{bmatrix} \frac{dy_1}{dx_1} & \frac{dy_1}{dx_2} & \dots & \frac{dy_1}{dx_D} \\ \frac{dy_2}{dx_1} & \frac{dy_2}{dx_2} & \dots & \frac{dy_2}{dx_D} \\ \dots & \dots & & \dots \\ \frac{dy_M}{dx_1} & \frac{dy_M}{dx_2} & & \frac{dy_M}{dx_D} \end{bmatrix}$$

Матрица частных производных

$$\frac{df}{dx} = \frac{dy}{dx}\cdot\frac{df}{dy} \rightarrow \frac{df}{dx} = \frac{df}{dy_1}\frac{dy_1}{dx} + \frac{df}{dy_2}\frac{dy_2}{dx}$$

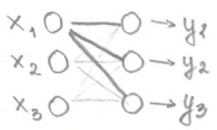
$$\nabla_x E = \text{Jacobian}^T \cdot \nabla_y E$$



Why sum?

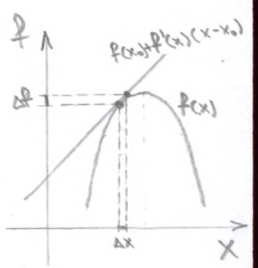
$$\frac{dE}{dx_1} = \sum_{i=1}^{M} \underbrace{\frac{dE}{dy_i}\frac{dy_i}{dx_1}}_{\text{chain rule}}$$

↑ Сумма вкладов $y_i(x_1)$ описывает общее изменение.



$$E = \sum_{j=1}^{M} \left(t^{(j)} - y^{(j)}(x_1, x_2, x_3)\right)^2$$

$$f'(x) = \lim_{\Delta x \to 0} \frac{\Delta f}{\Delta x} = \lim_{\Delta x \to 0} \frac{f(x_0+\Delta x) - f(x_0)}{\Delta x} = \frac{df}{dx}$$



$$f'(x) = \frac{f(x_0+\Delta x) - f(x_0)}{\Delta x} = \frac{f(x_0+x-x_0) - f(x_0)}{x - x_0} = \frac{f(x) - f(x_0)}{x - x_0} \Rightarrow f(x) = f(x_0) + f'(x_0)(x - x_0)$$

$$f'(x) = \frac{\Delta f}{\Delta x} \Rightarrow \Delta f = f'(x)\Delta x \rightarrow df = f'(x)\cdot dx$$

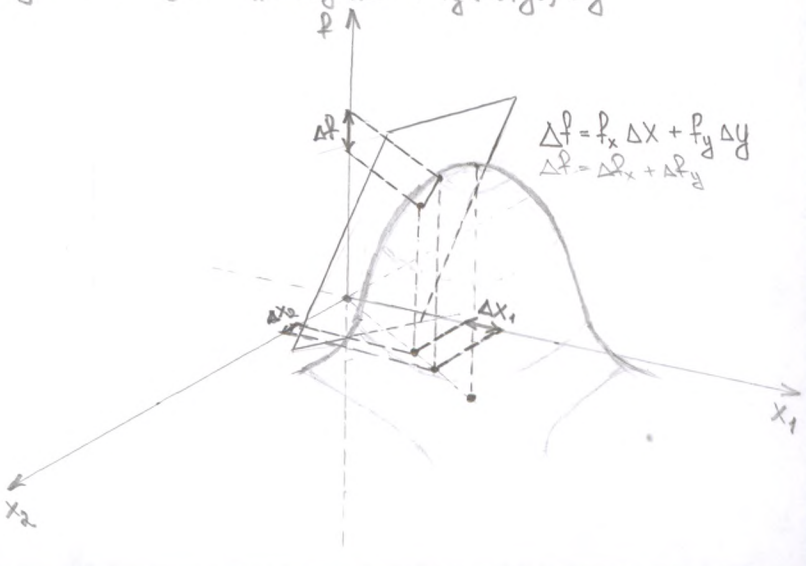
$\Delta S = v \cdot \Delta t$

$f(x,y)$

$$\Delta f = \left[f(x_0,y_0) + f_x(x_0,y_0)\Delta x + f_y(x_0,y_0)\cdot\Delta y\right] - f(x_0,y_0) = f_x(x_0,y_0)\Delta x + f_y(x_0,y_0)\Delta y$$

$$df = \frac{df}{dx}\cdot dx + \frac{df}{dy}\cdot dy$$

$E(y_1(x_1), y_2(x_1) \dots)$



Classification MLP Backprop[1] Loss Function

$\vec{y} = Softmax(\vec{z}^{(L)})$, $\vec{t} = [0\ 0\ 1\ 0 \dots 0]$ one-hot encoding.

$$P(T \mid X, W) = \prod_{n=1}^{N} \prod_{k=1}^{K} y^{(k)}(\bar{x}_n, W)^{t_n^{(k)}} \to max$$

$$\ln p(T \mid X, W) = \sum_{n=1}^{N} \sum_{k=1}^{K} t_n^{(k)} \ln y^{(k)}(\bar{x}_n, W) \to max$$

$$E = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_n^{(k)} \ln y^{(k)}(\bar{x}_n, W) \to min$$

Forward pass

$\bar{y} = Softmax(\bar{z}^{(L)}) = Softmax(\vec{f}(\bar{a}^{(L)})) = Softmax(\vec{f}(\bar{b}^{(L)} + W^{(L)} \bar{z}^{(L-1)}))$

$\bar{z}^{(0)} = \bar{x}$

$\bar{z}^{(0)} \to \bar{a}^{(1)} \to \bar{z}^{(1)} \to \dots \to \bar{a}^{(L)} \to \bar{z}^{(L)} \to \bar{y} \to E$

Backward pass

$\nabla_E E \to \nabla_y E \to \nabla_{z^{(L)}} E \to \nabla_{a^{(L)}} E \to \dots \to \nabla_{a^{(1)}} E$

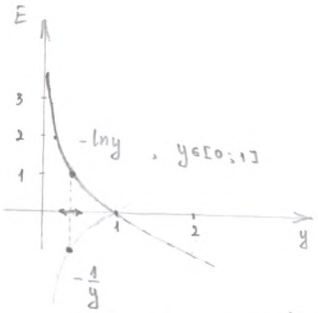
$\nabla_E E$: 1 — $\nabla_y E$: $\ln(y)$ — $\nabla_{z^{(L)}} E$: $Softmax(z^{(L)})$ — $\nabla_{a^{(L)}} E$: $\nabla_{W^{(L)}} E$, $\nabla_{b^{(L)}} E$ — $\nabla_{a^{(1)}} E$: $\nabla_{W^{(1)}} E$, $\nabla_{b^{(1)}} E$

$$E = -\sum_{k=1}^{K} t^{(k)} \ln y^{(k)}(\bar{x}, W) \to min$$

$$\begin{cases} \frac{dE}{dy^{(i)}} = \left(-\ln y^{(i)}(\bar{x}, W)\right)'_{y^{(i)}} = -\frac{1}{y^{(i)}(\bar{x}, W)}, & i = t_k \text{ - ground truth} \\ \frac{dE}{dy^{(i)}} = 0, & i \neq t_k \end{cases}$$

$\vec{t} = [\overset{0}{0}\ \overset{1}{0}\ \dots\ \overset{t_k}{1} \dots \overset{K-1}{0}\ \overset{K}{0}]^T$

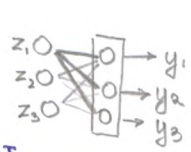
$$\nabla_y E = [0\ 0 \dots -\frac{1}{y^{(t_k)}(\bar{x}, W)} \dots 0\ 0]^T$$



отрицательный рост, значит нужно увеличивать y, чтобы уменьшить E

$\bar{y} = Softmax(\bar{z}^{(L)})$

$$\frac{dE}{dz_i^{(L)}} = \frac{dE}{dy_j} \cdot \frac{dy_j}{dz_i^{(L)}}$$



$$\Rightarrow \frac{dE}{dz_i^{(L)}} = \sum_{j=1}^{M} \frac{dE}{dy_j} \cdot \frac{dy_j}{dz_i}$$

$$\nabla_{z^{(L)}} E = \left[\sum_{i=1}^{M} \frac{dE}{dy_i} \frac{dy_i}{dz_1^{(L)}}, \sum_{i=1}^{M} \frac{dE}{dy_i} \frac{dy_i}{dz_2^{(L)}} \dots\right]^T = \underbrace{\left[\left[\frac{d\bar{y}}{dz_1^{(L)}}\right]\left[\frac{d\bar{y}}{dz_2^{(L)}}\right]\right]^T}_{Jacobian^T} \cdot \nabla_y E$$

я не знаю как интерпретировать Jacobian $-y^{(i)}y^{(i)}$

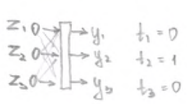
$$\nabla_{z^{(L)}} E = \left(\frac{d\bar{y}}{dz^{(L)}}\right)^T \cdot \nabla_y E$$

$$\nabla_{z^{(L)}} E = \vec{y} - \vec{t}$$

$$\nabla_z E = \begin{bmatrix} 0{,}1 \\ 0{,}5 \\ 0{,}4 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0{,}1 \\ -0{,}5 \\ 0{,}4 \end{bmatrix}$$

Чтобы уменьшить E, нужно двигаться в обратном направлении $\nabla_z E$

$$-\nabla_z E = \begin{bmatrix} -0{,}1 \\ 0{,}5 \\ -0{,}4 \end{bmatrix}$$

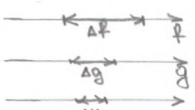


Уменьшаем соответственно ошибки

Chain Rule:

$$f'(g(x))_x = f'(g)_g \cdot g'(x)_x$$

$$\frac{df}{dx} = \frac{df}{dg} \cdot \frac{dg}{dx}$$



Classification MLP Backprop (2) (Jacobian)

$\nabla_y E$

$$\bar{y} = \text{Softmax}(\bar{z}^{(L)}) = \Big(\underbrace{\frac{e^{z_1^{(L)}}}{\sum_{i=1}^{M} e^{z_i^{(L)}}}}_{y^{(1)},\ i=j} \quad \underbrace{\frac{e^{z_2^{(L)}}}{\sum_{i=1}^{M} e^{z_i^{(L)}}}}_{y^{(2)},\ i\neq j} \quad \underbrace{\frac{e^{z_3^{(L)}}}{\sum_{i=1}^{M} e^{z_i^{(L)}}}}_{y^{(3)},\ i\neq j} \Big)$$

For example $z_1^{(L)}$ on $y^{(1)}$

$i=j$ $\quad \frac{dy^{(j)}}{dz_i^{(L)}} = \Big(\frac{e^{z_j^{(L)}}}{\sum_{u=1}^{M} e^{z_u^{(L)}}} \Big)'_{z_i^{(L)}} = \underbrace{(e^{z_j^{(L)}})'_{z_j^{(L)}}}_{(e^{z_1^{(L)}})' = e^{z_1^{(L)}},\ z_1} \Big(\frac{1}{\sum_{u=1}^{M} e^{z_u^{(L)}}} \Big) + \underbrace{(e^{z_j^{(L)}})}_{e^{z_1^{(L)}}} \Big(\frac{-1}{(\sum_{u=1}^{M} e^{z_u^{(L)}})^2} \cdot \underbrace{e^{z_i^{(L)}}}_{e^{z_1^{(L)}}} \Big) =$

$$= y^{(j)} + (-y^{(j)} \cdot y^{(i)}) = \boxed{y^{(j)} - (y^{(j)})^2}$$

$\left(\frac{u}{v}\right)' = (u \cdot v^{-1})' = u'v^{-1} + u \cdot (-1 v^{-2})$

$(f(g(x)))' = f'(g) \cdot g'(x)$

$z_1^{(L)}$ on $y^{(2)}$

$i \neq j$ $\quad \frac{dy^{(j)}}{dz_i^{(L)}} = \Big(\frac{e^{z_j^{(L)}}}{\sum_{u=1}^{M} e^{z_u^{(L)}}} \Big) = \underbrace{(e^{z_j^{(L)}})'_{z_i^{(L)}}}_{0} \Big(\frac{1}{\sum_{u=1}^{M} e^{z_u^{(L)}}} \Big) + \underbrace{e^{z_j^{(L)}}}_{e^{z_2^{(L)}}} \cdot \Big(\frac{-1}{(\sum_{u=1}^{M} e^{z_u^{(L)}})^2} \cdot \underbrace{e^{z_i^{(L)}}}_{e^{z_1^{(L)}}} \Big) =$

$$= 0 - y_j \cdot y_i = \boxed{-y^{(j)} y^{(i)}}$$

$$\text{Jacobian} = \frac{d\bar{y}}{d\bar{z}^{(L)}} = \Big[\Big[\frac{d\bar{y}}{dz_1^{(L)}} \Big] \Big[\frac{d\bar{y}}{dz_2^{(L)}} \Big] \dots \Big] = \Big[\underbrace{\begin{matrix} \frac{dy_1}{dz_1^{(L)}} \\ \frac{dy_2}{dz_1^{(L)}} \\ \dots \end{matrix}}_{dz_1^{(L)}} \quad \begin{matrix} \frac{dy_1}{dz_2^{(L)}} \\ \frac{dy_2}{dz_2^{(L)}} \\ \dots \end{matrix} \quad \dots \Big]$$

$$\frac{d\bar{y}}{d\bar{z}^{(L)}} = \Big[\underbrace{\begin{matrix} y^{(1)} - (y^{(1)})^2 \\ -y^{(2)} y^{(1)} \\ -y^{(3)} y^{(1)} \\ \dots \\ -y^{(M)} y^{(1)} \end{matrix}}_{dz_1^{(L)}} \quad \begin{matrix} -y^{(1)} y^{(2)} \\ y^{(2)} - (y^{(2)})^2 \\ -y^{(3)} y^{(2)} \\ \dots \\ -y^{(M)} y^{(1)} \end{matrix} \quad \begin{matrix} -y^{(1)} y^{(3)} \\ -y^{(2)} y^{(3)} \\ y^{(3)} - (y^{(3)})^2 \\ \dots \\ -y^{(M)} y^{(3)} \end{matrix} \quad \dots \Big]$$

$$\nabla_{z^{(L)}} E = \Big(\frac{d\bar{y}}{d\bar{z}^{(L)}} \Big)^T \nabla_y E \quad , \quad \nabla_y E = [0\ 0\ \dots\ -\frac{1}{y^{(t_k)}}\ \dots\ 0\ 0]^T$$

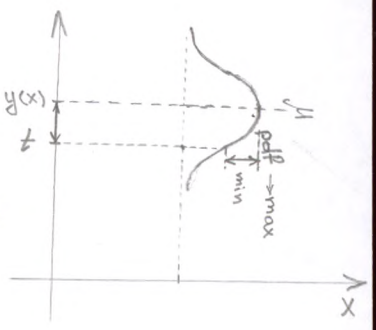
$$\nabla_{z^{(L)}} E = \begin{bmatrix} 0 + (-y^{(1)} y^{(t_k)}) \cdot (-\frac{1}{y^{(t_k)}}) = y^{(1)} \\ 0 + (-y^{(2)} y^{(t_k)}) \cdot (-\frac{1}{y^{(t_k)}}) = y^{(2)} \\ \dots \\ 0 + (y^{(t_k)} - (y^{(t_k)})^2)(-\frac{1}{y^{(t_k)}}) = -1 + y^{(t_k)} = y^{(t_k)} - 1 \\ \dots \\ y^{(M)} \end{bmatrix} = \bar{y} - \bar{t}$$

t_k-th row

1) $-y^{(1)} y^{(t_k)}$ когда вычисляем $z_i^{(L)}$, $i \neq t_k$
2) $y^{(t_k)} - y^{(t_k)} y^{(t_k)}$, когда вычисляем $z_i^{(L)}$, $i = t_k$

other rows $\times 0$

Regression MLP Backprop



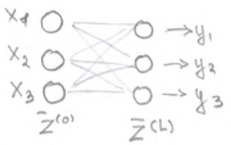
$t(\bar{x}) = y(\bar{x}, \bar{w}) + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \sigma^2)$

$$p(T \mid X, W) = \prod_{n=1}^{N} \mathcal{N}(t_n \mid y(\bar{x}_n, W), \sigma^2) = \prod_{n=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}\left(\frac{t_n - y(\bar{x}_n, W)}{\sigma}\right)^2} \to \max$$

$$\ln p(T \mid X, W) = \sum_{n=1}^{N} \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right) + \sum_{n=1}^{N}\left(-\frac{1}{2}\left(\frac{t_n - y(\bar{x}_n, W)}{\sigma}\right)^2 \cdot \ln e\right) =$$

$$= \underbrace{N \ln\left(\frac{1}{\sigma\sqrt{2\pi}}\right)}_{\text{const}} - \underbrace{\frac{1}{\sigma^2}}_{\text{const}} \cdot \underbrace{\frac{1}{2}\sum_{n=1}^{N}(t_n - y(\bar{x}_n, W))^2}_{\min} \to \max$$

$$E = \frac{1}{2}\sum_{n=1}^{N}(t_n - y(\bar{x}_n, W))^2 \to \min$$



$\bar{y} = [R\ R\ R]^T$

For multiple outputs: $\bar{y} \in \mathbb{R}^M$

$$E = \frac{1}{2}\sum_{n=1}^{N}\sum_{j=1}^{M}(t_n^{(j)} - y^{(j)}(\bar{x}_n, W))^2 \to \min$$

Linear activation function: $f(a) = a$

Forward pass

$\bar{x} = \bar{z}^{(0)}$

$$\bar{z}^{(0)} \to \bar{a}^{(1)} \to \bar{z}^{(1)} \to \bar{a}^{(2)} \to \bar{z}^{(2)} \to \dots \to \bar{z}^{(L)} \to y \to E$$

$$\bar{y} = \bar{z}^{(L)} = f(\bar{a}^{(L)}) = f(\bar{b}^{(L)} + W^{(L)} \bar{z}^{(L-1)})$$

Backward pass

$$\underbrace{\nabla_E E}_{1} \to \underbrace{\nabla_y E}_{\bar{y} = \bar{z}^{(L)}} \to \nabla_{z^{(L)}} E \to \nabla_{a^{(L)}} E \to \nabla_{z^{(L-1)}} E \to \nabla_{a^{(L-1)}} E \to \dots \to \nabla_{a^{(1)}} E$$

$\nabla_{a^{(L)}} E \to \nabla_{W^{(L)}} E,\ \nabla_{b^{(L)}} E$

$\nabla_{a^{(L-1)}} E \to \nabla_{W^{(L-1)}} E,\ \nabla_{b^{(L-1)}} E$

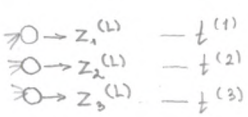
$\nabla_{a^{(1)}} E \to \nabla_{W^{(1)}} E,\ \nabla_{b^{(1)}} E$

$$E = \frac{1}{2}\sum_{n=1}^{N}\sum_{j=1}^{M}(t_n^{(j)} - z_j^{(L)})^2 \to \min$$

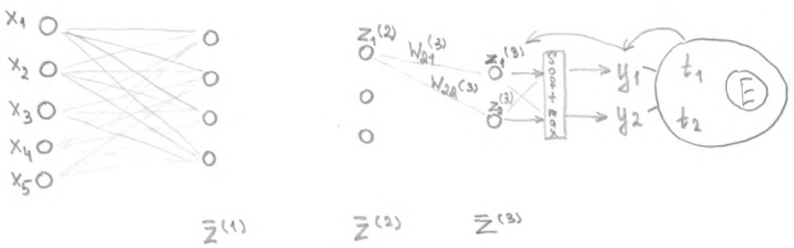
$$\frac{dE}{dz_k^{(L)}} = \frac{1}{2}\sum_{n=1}^{N}\sum_{j=1}^{M} 2 \cdot (t_n^{(j)} - z_j^{(L)}) \cdot (t_n^{(j)} - z_j^{(L)})'_{z_k^{(L)}} = \sum_{n=1}^{N}\sum_{j=1}^{M}(z_j^{(L)} - t_n^{(j)})\, I(j=k) = \sum_{n=1}^{N}(z_k^{(L)} - t_n^{(k)})$$

$$\begin{cases} -1, \text{ if } j=k \\ 0, \text{ otherwise} \end{cases}$$

$$\boxed{\nabla_{z^{(L)}} E = \bar{z}^{(L)} - \bar{t}}$$



Backprop (1)



$$\bar{z}^{(l)} = \bar{f}(\bar{a}^{(l)}) = \bar{f}(\bar{b}^{(l)} + W^{(l)}\bar{z}^{(l-1)})$$

$$\frac{dE}{da} = \frac{dE}{dy}\frac{dy}{dz}\frac{dz}{df}\frac{df}{da} = \frac{dE}{dz}\underbrace{\frac{dz}{df}}_{f=z,\ 1}\frac{df}{da} = \frac{dE}{dz}\cdot\frac{df}{da}$$

$\nabla_{z^{(l)}} E$, $\bar{z}^{(l)} = \bar{f}(\bar{a}^{(l)})$

$z_1^{(l)} = \mathrm{ReLU}(a_1^{(l)})$

$$\frac{dz}{da} = \frac{dz}{df_1}\frac{df_1}{da} + \frac{dz}{df_2}\frac{df_2}{da} \dots \Rightarrow \nabla_{a^{(l)}} E = \left[\sum_{i=1}^{M}\frac{dz}{df_i}\frac{df_i}{da_1} \quad \sum_{i=1}^{M}\frac{dz}{df_i}\frac{df_i}{da_2} \dots\right]^T$$, but $f_2(a_2) \neq f_2(a_1)$ $\Rightarrow f_2'(a_1) = 0$

$$\nabla_{a^{(l)}} E = \left(\frac{d\bar{f}}{d\bar{a}}\right)^T \cdot \nabla_{z^{(l)}} E \qquad \nabla_{a^{(l)}} E = \left[\frac{dz}{df_1}\frac{df_1}{da_1} \quad \frac{dz}{df_2}\frac{df_2}{da_2} \dots\right]^T$$

$$\frac{d\bar{f}}{d\bar{a}} = \begin{bmatrix} \frac{df_1}{da_1} & \frac{df_1}{da_2} & \dots & \frac{df_1}{da_M} \\ \frac{df_2}{da_1} & \frac{df_2}{da_2} & \dots & \\ \dots & \dots & \dots & \frac{df_M}{da_M} \end{bmatrix} = \begin{bmatrix} \frac{df_1}{da_1} & 0 & \dots & 0 \\ 0 & \frac{df_2}{da_2} & \dots & 0 \\ 0 & 0 & \dots & 0 \\ \dots & \dots & \dots & \frac{df_M}{da_M} \end{bmatrix}$$

$$\left(\frac{d\bar{f}}{d\bar{a}}\right)^T \cdot \nabla_{z^{(l)}} E = \left[\frac{df_1}{da_1^{(l)}}\frac{dE}{dz_1^{(l)}} \quad \frac{df_2}{da_2^{(l)}}\frac{dE}{dz_2^{(l)}} \dots\right]$$

$$\nabla_{a^{(l)}} E = \underbrace{f'(\bar{a}^{(l)}) \circ \nabla_{z^{(l)}} E}_{\text{elementwise}} = \left(\frac{d\bar{f}}{d\bar{a}}\right)^T \nabla_{z^{(l)}} E = \nabla_{a^{(l)}} E$$ — activation functions

$$\frac{dE}{db_i^{(l)}} = \frac{dE}{dz_i^{(l)}}\cdot\frac{dz_i^{(l)}}{da_i^{(l)}}\cdot\frac{da_i^{(l)}}{db_i^{(l)}} = \frac{dE}{da_i^{(l)}}\cdot\frac{da_i^{(l)}}{db_i^{(l)}} = \frac{dE}{da_i^{(l)}} \qquad a(b) = b + Wz^{(l-1)}$$
$$a'(b) = (b)_b' = 1$$

$$\nabla_{b^{(l)}} E = \left[\frac{dE}{da_1^{(l)}} \quad \frac{dE}{da_2^{(l)}} \quad \frac{dE}{da_3^{(l)}} \dots\right]^T = \nabla_{a^{(l)}} E$$

$$\nabla_{b^{(l)}} E = \nabla_{a^{(l)}} E$$

$\nabla_{z^{(l)}} E$ - loss function

$\nabla_{a^{(l)}} E$ - activation function

$$\bar{z} = \bar{f}(\bar{a}) = \bar{f}(\bar{b} + Wz^{(l-1)})$$

$$\frac{dE}{da} = \frac{dE}{dz}\cdot\underbrace{\frac{dz}{da}}_{f'(a)} \rightarrow \frac{dE}{db} = \frac{dE}{dz}\frac{dz}{da}\cdot\frac{da}{db} = \boxed{\frac{dE}{dz}\cdot f'(a)\cdot a'(b)}$$

Backprop (2)

$$\bar{z}^{(l)} = \bar{f}(\bar{a}^{(l)}) = \bar{f}(\bar{b}^{(l)} + W^{(l)}\bar{z}^{(l-1)}) \qquad (M_l \times M_{l-1}) \times (M_{l-1} \times 1)$$

$$W^{(l)} = \begin{bmatrix} w_{11} & w_{21} & w_{31} & w_{41} \\ w_{12} & w_{22} & w_{32} & w_{42} \\ \dots & & & \\ w_{1M} & w_{2M} & \dots & w_{M-1M} \end{bmatrix}$$

$\nabla_{z^{(l)}} E$, $\nabla_{a^{(l)}} E$, $\nabla_{b^{(l)}} E$

$\nabla_{W^{(l)}} E$ - ?

$$\bar{z}^{(l-1)} = \begin{bmatrix} z_1 \\ z_2 \\ \dots \\ z_{M-1} \end{bmatrix}$$

$$a_1^{(l)} = b_1^{(l)} + \sum_{i=1}^{M-1} w_{i1} \cdot z_i^{(l-1)}$$

$$a_j(w_{ij}) = b_j + \sum_{m=1}^{M-1} w_{mj} \cdot z_m^{(l-1)} \quad , \quad a'(w_{ij}) = 0 + (0 + (w_{ij} \cdot z_i^{(l-1)})'_{w_{ij}}) = z_i^{(l-1)}$$

$$\frac{dE}{dw_{ij}} = \frac{dE}{da_j^{(l)}} \cdot \frac{da_j^{(l)}}{dw_{ij}} = \frac{dE}{da_j^{(l)}} \cdot z_i^{(l-1)}$$

$$\nabla_{W^{(l)}} E = \begin{bmatrix} \frac{dE}{dw_{11}} & \frac{dE}{dw_{21}} & \dots & \frac{dE}{dw_{M-11}} \\ \frac{dE}{dw_{12}} & \frac{dE}{dw_{22}} & \dots & \frac{dE}{dw_{M-12}} \\ \dots & \dots & \ddots & \\ \frac{dE}{dw_{1M}} & \frac{dE}{dw_{2M}} & & \frac{dE}{dw_{M-1M}} \end{bmatrix} = \begin{bmatrix} \frac{dE}{da_1^{(l)}} \cdot \frac{da_1^{(l)}}{dw_{11}} & \frac{dE}{da_1^{(l)}} \frac{da_1^{(l)}}{dw_{21}} \\ \dots & \end{bmatrix} = \begin{bmatrix} \frac{dE}{da_1^{(l)}} \cdot z_1^{(l-1)} & \frac{dE}{da_1^{(l)}} \cdot z_2^{(l-1)} \\ \frac{dE}{da_2^{(l)}} z_1^{(l-1)} & \frac{dE}{da_2^{(l)}} \cdot z_2^{(l-1)} \end{bmatrix}$$

$$\nabla_{W^{(l)}} E = \nabla_{a^{(l)}} E \cdot (\bar{z}^{(l)})^T = \begin{bmatrix} \frac{dE}{da_1^{(l)}} \\ \frac{dE}{da_2^{(l)}} \end{bmatrix} \begin{bmatrix} z_1^{(l)} & z_2^{(l)} \end{bmatrix}$$

$\nabla_{z^{(l-1)}} E$ - ?

$$\frac{dE}{dz_1^{(l-1)}} = \frac{dE}{dz_1^{(l)}} \frac{da}{dz_1^{(l-1)}} + \frac{dE}{dz_2^{(l)}} \frac{dz_2^{(l)}}{da_2^{(l)}} \frac{da_2^{(l)}}{dz_1^{(l-1)}} = \dots$$

$$= w_{11} + w_{12} + w_{13} \dots$$

$$\nabla_{z^{(l-1)}} E = W_l^T \cdot \nabla_{a^{(l)}} E$$

Forward

$$\bar{z}^{(0)} \to \bar{a}^{(1)} \to \bar{z}^{(1)} \to \dots \to \bar{z}^{(l-1)} \to \bar{a}^{(l)} \to \bar{z}^{(l)} \to y \to E$$

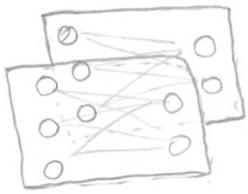
Backward

$$\nabla_E E \to \nabla_y E \to \nabla_{\bar{z}^{(l)}} E \to \nabla_{a^{(l)}} E \to \nabla_{z^{(l-1)}} E \to \dots \to \nabla_{z^{(1)}} E \to \nabla_{a^{(1)}} E$$

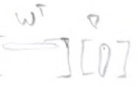
$\nabla_{a^{(l)}} E \to \nabla_{W^{(l)}} E$, $\nabla_{b^{(l)}} E$

$\nabla_{a^{(1)}} E \to \nabla_{W^{(1)}} E$, $\nabla_{b^{(1)}} E$

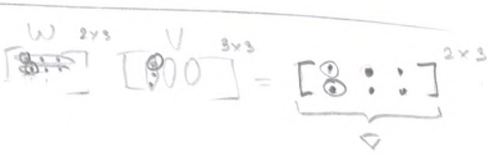
$W \cdot \vec{v}$



$(W \cdot \vec{v})'_{\vec{v}} = W^T \cdot \nabla_{\vec{y}}$

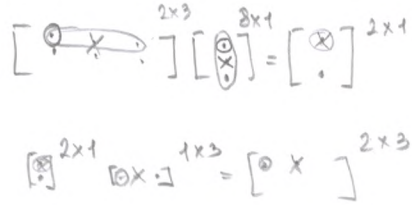


$W \cdot V = [000]$ (m×k, k)

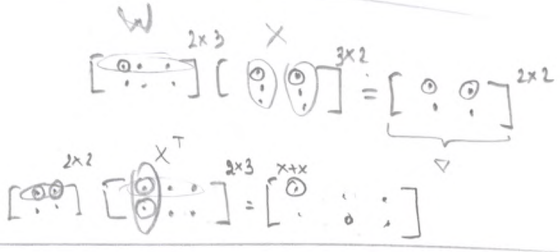


$(W \cdot V)'_V = W^T \nabla$, либо $\nabla^{2\times 3}$

$(Wx)'_W = \nabla \cdot x^T$



$(WX)'_W = \nabla X^T$



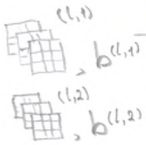
$W \cdot V' = W^T \cdot \nabla$

$W' \cdot V = \nabla \cdot V^T$

CNN

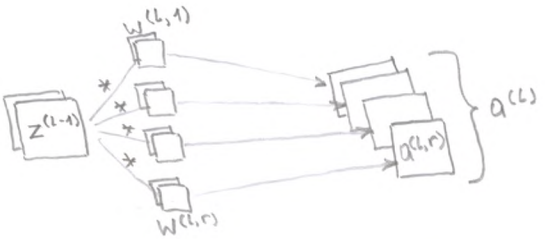
Input Tensor

$z^{(l-1)} \in R^{m_{l-1} \times n_{l-1} \times r_{l-1}}$



Convolution filters

$w^{(l,r)} \in R^{\tilde{m}_l \times \tilde{n}_l \times r_{l-1}}$, $r = 1, 2, \dots, r_l$



$z^{(l-1)} * w^{(l)} = a^{(l)}$

Activation maps

$a^{(l,r)} \in R^{m_l \times n_l}$, $r = 1, 2, \dots, r_l$

$a^{(l)} \in R^{m_l \times n_l \times r_l}$

Feature maps ($z^{(l,r)} = f(a^{(l,r)})$)

$z^{(l,r)} \in R^{m_l \times n_l}$

$z^{(l)} \in R^{m_l \times n_l \times r_l}$ – output tensor

Stride – шаг

dilation – внутреннее пространство

Convolution actually Cross-Correlation (no flip)

Image $(I)^{H \times W}$

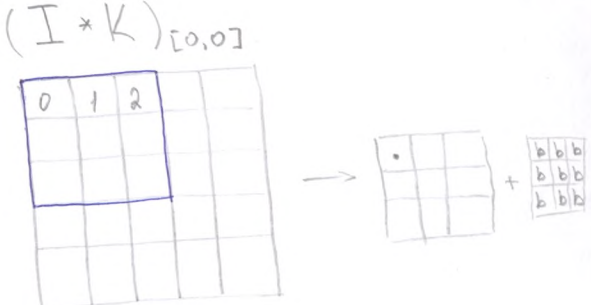
Kernel $(K)^{M \times N}$ – convolution filter

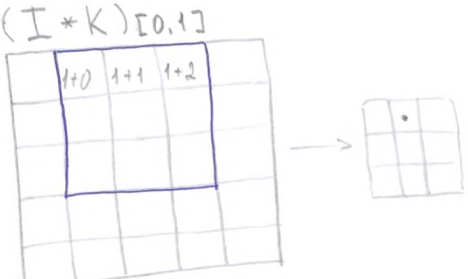
$I_{[i,j]}$ – element

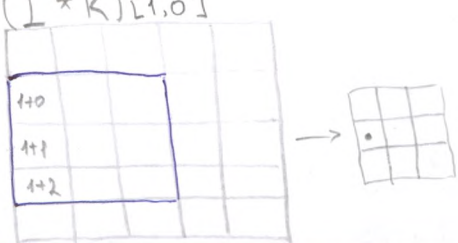
$K_{[m,n]}$ – element

$$(I * K)_{[i,j]} = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} I_{[i+m, j+n]} \cdot K_{[m,n]} + b$$

elementwise





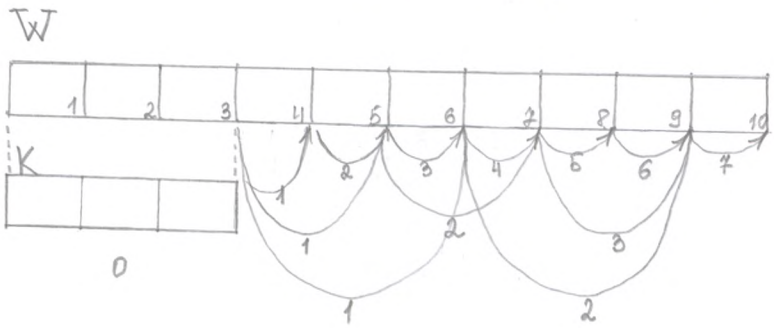


Generally

- $a^{(l,r)} = z^{(l-1)} * w^{(l,r)} + b^{(l,r)} \cdot J$, $J = \{1\}_{m_l \times n_l}$, $r = 1, 2, \dots, r_l$
- $a^{(l)} = z^{(l-1)} * w^{(l)} + b^{(l)}$

$z^{(l)} = f(a^{(l)})$, f – ReLU

CNN. Output shape



$W = 10,\ K = 3$

dilation = 1

- stride = 1 : $W_{out} = 8 = (10-3)+1$
- stride = 2 : $W_{out} = 4 = \lfloor (\frac{10-3}{2}+1) \rfloor = \lfloor 3.5+1 \rfloor$
- stride = 3 : $W_{out} = 3 = \lfloor (\frac{7}{3}+1) \rfloor = \lfloor 2.3+1 \rfloor$

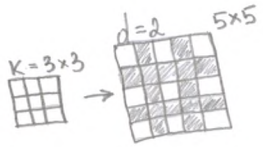
$$W_{out} = \left\lfloor \frac{(W+2p) - (d(K-1)+1)}{S} + 1 \right\rfloor$$

(W+2p): beginning shape; d(K-1)+1: as kernel; S: stride

2p - padding

d - dilation, default = 1

Dilation, как расширение ядра

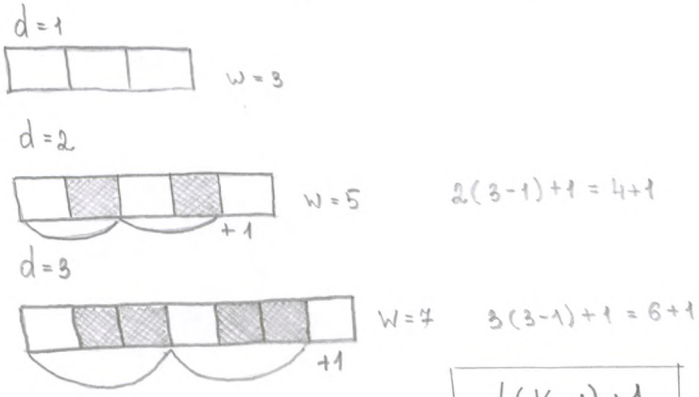


K = 3:

d = 1 — W = 3

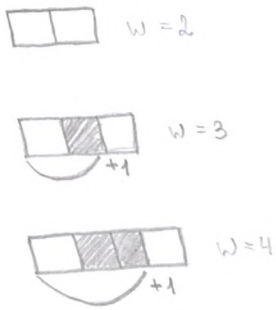
d = 2 — W = 5 — $2(3-1)+1 = 4+1$

d = 3 — W = 7 — $3(3-1)+1 = 6+1$

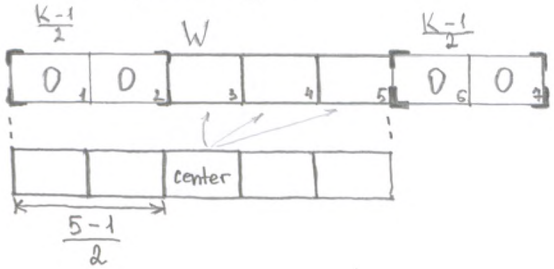


$$d(K-1)+1$$

K = 2:



Zero padding

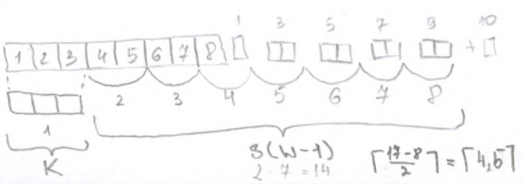


$K = 2n+1 \in \{3, 5, 7, 9 \dots\}$

Чтобы получить $W_{out} = W$ нужно запанить $\frac{K-1}{2}$ нулями по краям.

$$p = \frac{K-1}{2}$$

$$p = \left\lceil \frac{S(W-1) - W + K}{2} \right\rceil$$

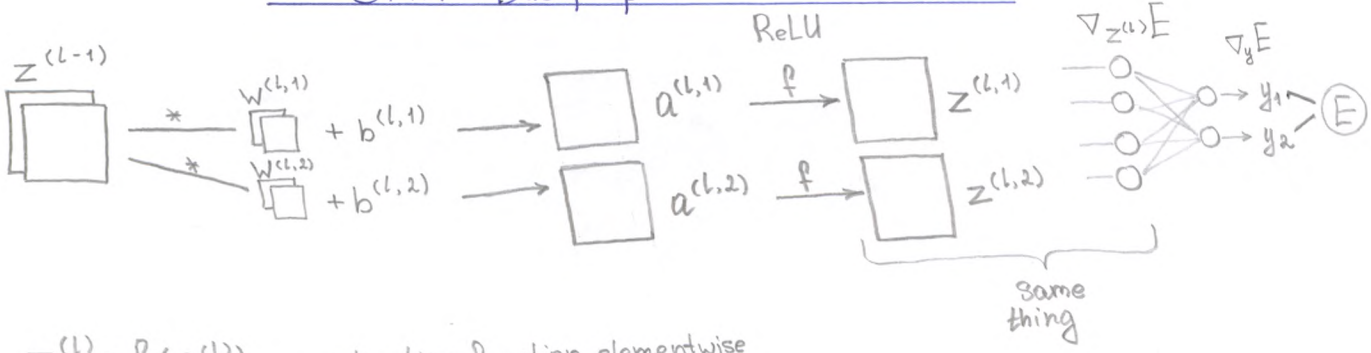


В общем должно быть $K + S(W-1)$

1 2 3 4 5 6 7 — 7 - 4 — $\lceil \frac{7-4}{2} \rceil = \lceil 1.5 \rceil = 2$

1 2 3 4

CNN Backprop.



$z^{(l)} = f(a^{(l)})$ - activation function elementwise

$\nabla_{a^{(l)}} E = \left(\frac{d\bar{f}}{d\bar{a}^{(l)}}\right)^T \cdot \nabla_{z^{(l)}} E$ $\qquad \frac{dE}{da} = \frac{dE}{dz} \cdot \underbrace{\frac{dz}{df}}_{1} \frac{df}{da}$

$$\begin{bmatrix} \frac{df_1}{da_1} & \frac{df_2}{da_1} & \dots \\ \frac{df_1}{da_2} & \frac{df_2}{da_2} & \\ \dots & & \ddots \end{bmatrix} = \begin{bmatrix} [\bar{f}'(a^{(l,1)})] & 0 & 0 \\ 0 & [\bar{f}'(a^{(l,2)})] & \dots \\ \dots & \dots & \end{bmatrix}$$

представим, что мы вытягиваем $a^{(l)}$ в вектор типа

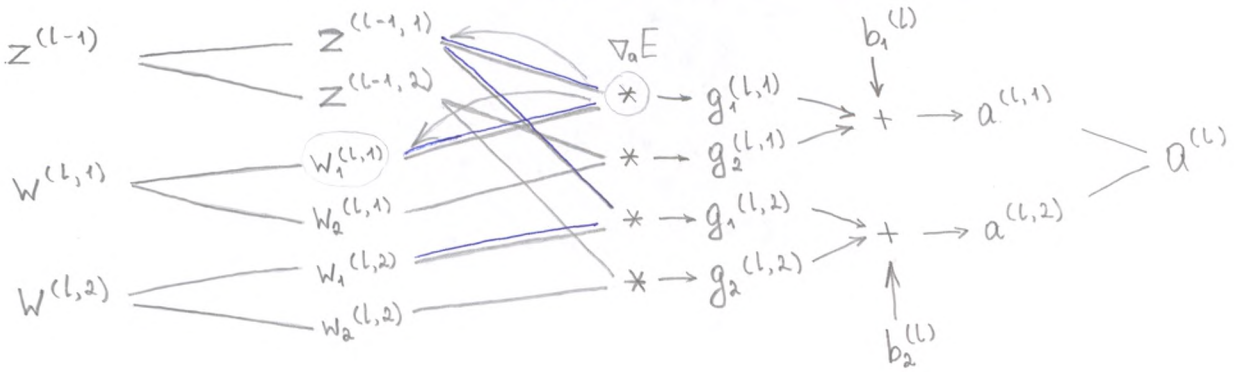
$\nabla_{a^{(l)}} E = f'(a^{(l)}) \circ \nabla_{z^{(l)}} E$

elementwise

$$\left(\begin{bmatrix} f(a_{11}^{(l,1)}) & f(a_{12}^{(l,1)}) & \dots \\ f(a_{21}^{(l,1)}) & f(a_{22}^{(l,1)}) & \dots \\ \dots & \dots & \ddots \end{bmatrix}_a \right)' =$$

$$= \begin{bmatrix} \frac{dE}{dz^{(l,1)}} \cdot \frac{df}{da_{11}^{(l,1)}} & \dots \\ \frac{dE}{dz^{(l,1)}} \cdot \frac{df}{da_{21}^{(l,1)}} & \end{bmatrix}$$

Backprop through convolutions $(Z * W)'_W$

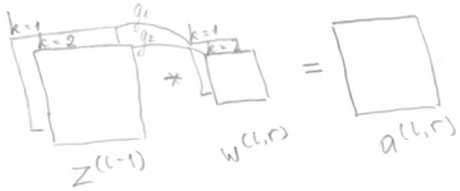


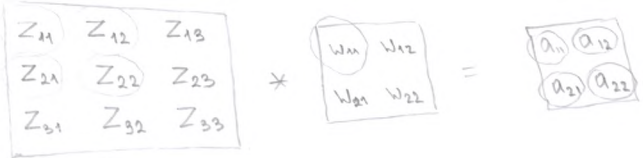
$$g_k^{(l,r)} = z^{(l-1,k)} * w_k^{(l,r)}$$

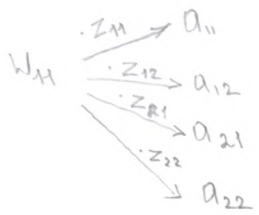
$$a^{(l,r)} = \sum_k g_k^{(l,r)} + b_r^{(l)}$$

$$a = \sum_k g_k = \Sigma(z * w) \Rightarrow \frac{dE}{dw} = \frac{dE}{da} \cdot \frac{da}{dw}$$

$\nabla_{W_1^{(l,r)}} E$ - ?







$$\frac{dE}{dw_{11}} = \frac{dE}{a_{11}} \cdot z_{11} + \frac{dE}{da_{12}} z_{12} + \frac{dE}{da_{21}} \cdot z_{21} + \frac{dE}{da_{22}} \cdot z_{22}$$

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix}^{3\times3} * \begin{bmatrix} \frac{dE}{da_{11}} & \frac{dE}{da_{12}} \\ \frac{dE}{da_{21}} & \frac{dE}{da_{22}} \end{bmatrix}^{2\times2} = \begin{bmatrix} \frac{dE}{dw_{11}} & \frac{dE}{dw_{12}} \\ \frac{dE}{dw_{21}} & \frac{dE}{dw_{22}} \end{bmatrix}^{2\times2}$$

$$z^{(l-1)} * w^{(l,r)}$$

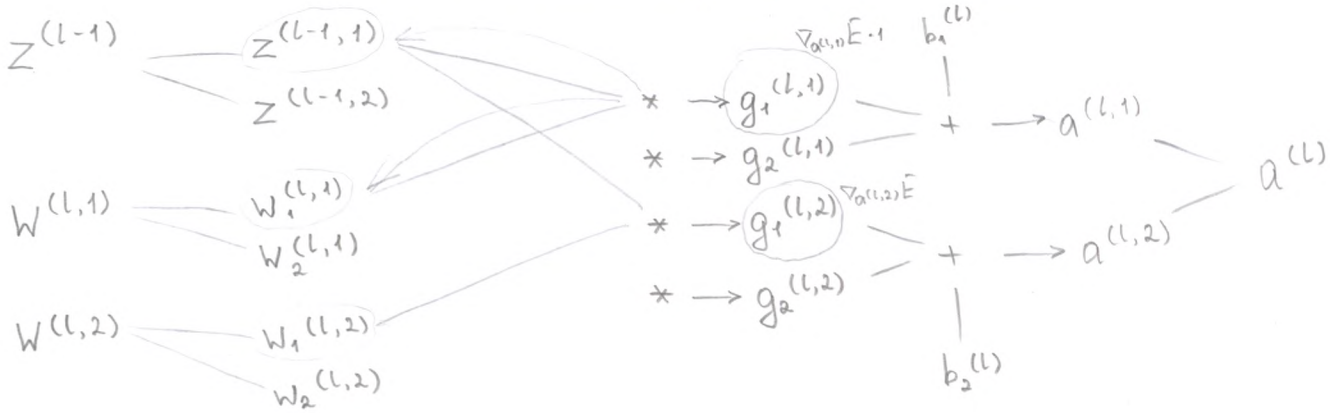
$$z^{(l-1,k)} * \nabla_{a^{(l,r)}} E = \nabla_{w_k^{(l,r)}} E$$

$$\underbrace{\nabla_{w_k^{(l,r)}} E}_{\text{by channel}} = z^{(l-1,k)} * \nabla_{a^{(l,r)}} E$$

, дальше нужно сложить в тензор

$$\nabla_{W^{(l,r)}} E = z^{(l-1)} * \nabla_{a^{(l,r)}} E$$

Backprop through Convolution $(Z * W)'_Z$



$$Z^{(l-1)} * W^{(l)} + b^{(l)} \quad f, \; f = \{1\}_{m_l \times n_l}$$

$$\begin{cases} Z^{(l-1,1)} * W_1^{(l,1)} = g_1^{(l,1)} \\ Z^{(l-1,2)} * W_1^{(l,2)} = g_1^{(l,2)} \end{cases}$$

$a' = g'$

$\nabla_{Z^{(l-1,1)}} E$ - ?

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix} * \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \quad g_k^{(l,r)}$$

$\tilde{W}_k^{(l,r)} = W_k^{(l,r)}$. flip [: :]

1 channel

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \nabla a_{11} & \nabla a_{12} & 0 \\ 0 & \nabla a_{21} & \nabla a_{22} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} w_{22} & w_{21} \\ w_{12} & w_{11} \end{bmatrix} \quad \tilde{W}_k^{(l,r)}$$

$k \to$, $P = \lfloor \frac{\tilde{W}}{2} \rfloor$

z_{22} : $\cdot w_{22} \to a_{11}$, $\cdot w_{21} \to a_{12}$, $\cdot w_{12} \to a_{21}$, $\cdot w_{11} \to a_{22}$

$$\Rightarrow \frac{dE}{dz_{22}} = \frac{dE}{da_{11}} \cdot w_{22} + \frac{dE}{da_{12}} \cdot w_{21} + \frac{dE}{da_{21}} \cdot w_{12} + \frac{dE}{da_{22}} \cdot w_{11}$$

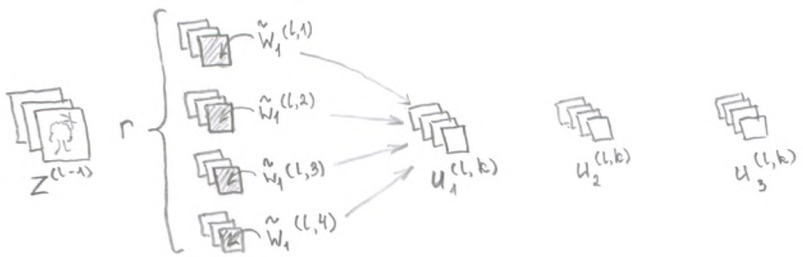
$\nabla_a E \cdot \tilde{W}$ like

$$\sim \nabla_{Z^{(l-1,k)}} E = \text{ZeroPad}\left(\nabla_{a^{(l,r)}} E\right) * \tilde{W}_k^{(l,r)}$$

Derivative $\underbrace{\square * \square}_{z^{(l-1,1)} * W_1^{(l,1)}}$, но это не полный, потому что он еще зависит от второй свертки

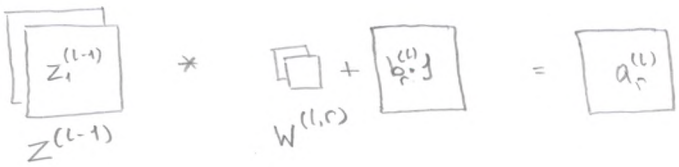
$$\boxed{\nabla_{Z^{(l-1,k)}} E = \sum_r \text{ZeroPad}\left(\nabla_{a^{(l,r)}} E\right) * \tilde{W}_k^{(l,r)}} = \text{ZeroPad}\left(\nabla_{a^{(l)}} E\right) \cdot \underbrace{U^{(l,k)}}_{[W_1^{(l,1)} \; W_1^{(l,2)} \; \dots \; W_1^{(l,r)}]}$$

$U_r^{(l,k)} = \tilde{W}_k^{(l,r)}$

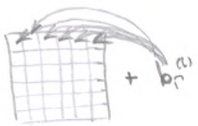


Это интересно в плане нотации, но на практике мы будем использовать AutoGrad и нам важно только $\text{ZeroPad}(\nabla_a E) * \tilde{W} = (Z * W)'_Z$

Backprop through convolution $(Z * W + b \cdot 1)'_b$



$$a^{(l,r)} = z^{(l-1)} * w^{(l,r)} + b_r^{(l)} \cdot 1$$

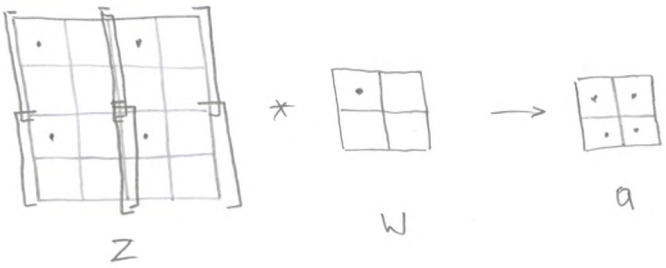


$$\frac{dE}{db^{(l,r)}} = \frac{dE}{da_{11}^{(l,r)}} \cdot 1 + \frac{dE}{da_{12}} \cdot 1 + \dots$$

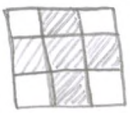
$$\Rightarrow \quad \nabla_{b^{(l,r)}} E = \underset{\text{elements of arg.}}{\text{Sum}} \left(\nabla_{a^{(l,r)}} E \right), \quad r = 1, \dots, n^l$$

Backprop through convolution with stride

1)



Dilate(a) =



$$\nabla_W E = z^{(l-1)} * \text{Dilate}(\nabla_a E)$$

2)

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} & z_{14} & z_{15} \\ z_{21} & z_{22} & z_{23} & z_{24} & z_{25} \\ z_{31} & z_{32} & z_{33} & z_{34} & z_{35} \\ z_{41} & z_{42} & z_{43} & z_{44} & z_{45} \\ z_{51} & z_{52} & z_{53} & z_{54} & z_{55} \end{bmatrix} * \begin{bmatrix} w_{11}^{4} & w_{12} & w_{13}^{3} \\ w_{21} & w_{22} & w_{23} \\ w_{31}^{2} & w_{32} & w_{33}^{1} \end{bmatrix} = \begin{bmatrix} a_{11}^{1} & a_{12}^{2} \\ a_{21}^{3} & a_{22}^{4} \end{bmatrix}$$

z_{33} · w_{33} a_{11}

· w_{31} a_{12}

· w_{13} a_{21}

· w_{11} a_{22}

$$\frac{dE}{dz_{33}} = \underbrace{\frac{dE}{da_{11}} \cdot w_{33}} + \underbrace{\frac{dE}{da_{12}} \cdot w_{31}} + \underbrace{\frac{dE}{da_{21}} \cdot w_{13}} + \underbrace{\frac{dE}{da_{22}} \cdot w_{11}}$$

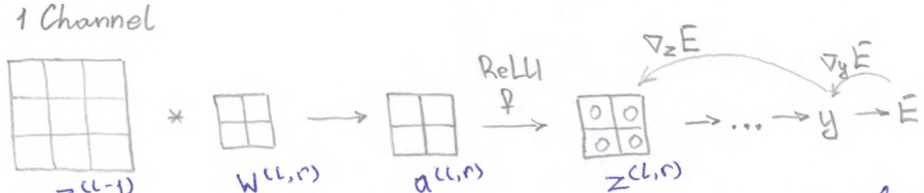
$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{dE}{da_{11}} & 0 & \frac{dE}{da_{12}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{dE}{da_{21}} & 0 & \frac{dE}{da_{22}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} w_{33} & w_{32} & w_{31} \\ w_{23} & w_{22} & w_{21} \\ w_{13} & w_{12} & w_{11} \end{bmatrix} = \nabla_z E$$

$\tilde{w}$

$$\text{ZeroPad}\left(\text{Dilate}_{s-1}(\nabla_{a^{(l)}} E)\right)$$

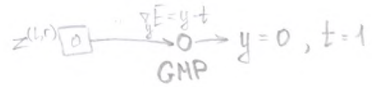
CNN Learnability

1 Channel



Свёрточные фильтры обучаются только на активированных feature map.

$\nabla_{z^{(l,r)}} E$ not zero



$$\nabla_{z^{(l,r)}} E = \nabla_{y^{(l,r)}} E$$

$\nabla_{a^{(l,r)}} E$

$$ReLU'(a^{(l,r)}) = \begin{cases} 1, & \text{if } a > 0 \\ 0, & \text{if } a \le 0 \end{cases}$$

$$z^{(l-1)} * w^{(l,r)} = a^{(l,r)}$$

$\nabla_{w^{(l,r)}} E$

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix} * \begin{bmatrix} w_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

Все веса влияют на целую карту

w_{11}: $z_{11} \to a_{11}$, $z_{12} \to a_{12}$, $z_{21} \to a_{21}$, $z_{22} \to a_{22}$

$$\Rightarrow \frac{dE}{dw_{11}} = \frac{dE}{dz_{11}} \frac{dz_{11}}{da_{11}} + \frac{dE}{dz_{12}} \frac{dz_{12}}{da_{12}} + \dots$$

$$\nabla_w E = z^{(l-1)} * \nabla_a E$$

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} \\ z_{21} & z_{22} & z_{23} \\ z_{31} & z_{32} & z_{33} \end{bmatrix} * \begin{bmatrix} \frac{dE}{da_{11}} & \frac{dE}{da_{12}} \\ \frac{dE}{da_{21}} & \frac{dE}{da_{22}} \end{bmatrix} = \begin{bmatrix} \frac{dE}{dw_{11}} & \frac{dE}{dw_{12}} \\ \frac{dE}{dw_{21}} & \frac{dE}{dw_{22}} \end{bmatrix}$$

ReLU'

Если $\nabla_a E = 0$, то свертка не обучается.

Если хоть одно значение активировалось $\to \frac{dE}{da_{ij}} > 0$ то свертка обучается.

$\nabla_{z^{(l-1)}} E$

$$= ZeroPad(\nabla_{a^{(l)}} E) * \tilde{w}^{(l,r)}$$

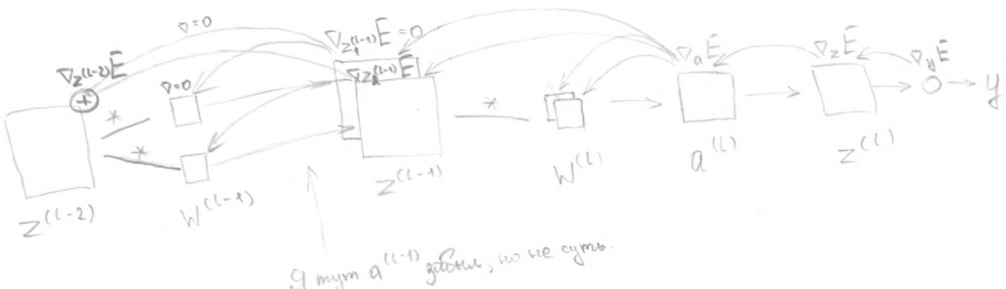
$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & \frac{dE}{da_{11}} & \frac{dE}{da_{12}} & 0 \\ 0 & \frac{dE}{da_{21}} & \frac{dE}{da_{22}} & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} w_{22} & w_{21} \\ w_{12} & w_{11} \end{bmatrix} = \begin{bmatrix} \frac{dE}{dz_{11}} & \frac{dE}{dz_{12}} & \frac{dE}{dz_{13}} \\ \frac{dE}{dz_{21}} & \frac{dE}{dz_{22}} & \frac{dE}{dz_{23}} \\ \frac{dE}{dz_{31}} & \frac{dE}{dz_{32}} & \frac{dE}{dz_{33}} \end{bmatrix}$$

Здесь снова если хоть одно значение активировалось, то потенциально предыдущие слои будут обучаться. (?)

∇ градиент может затухнуть по каналу, но он там будет суммироваться с градиентами других каналов и обучение пройдёт.

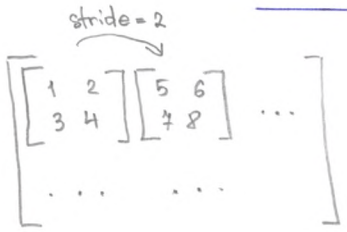
$$(a^{(l-1,r)})' = \frac{dE}{dz^{(l-1)}} \cdot ReLU'(a^{(l-1,r)})$$

$\frac{dE}{dz^{(l-1)}} = \nabla_{z^{(l-1)}} E$



А тут $a^{(l-1)}$ добыли, но не суть.

Pooling



Max Pooling $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \rightarrow 4$

Average Pooling $\begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \rightarrow \frac{1+2+3+4}{4} = 2,5$

stride = kernel size

Global Pooling

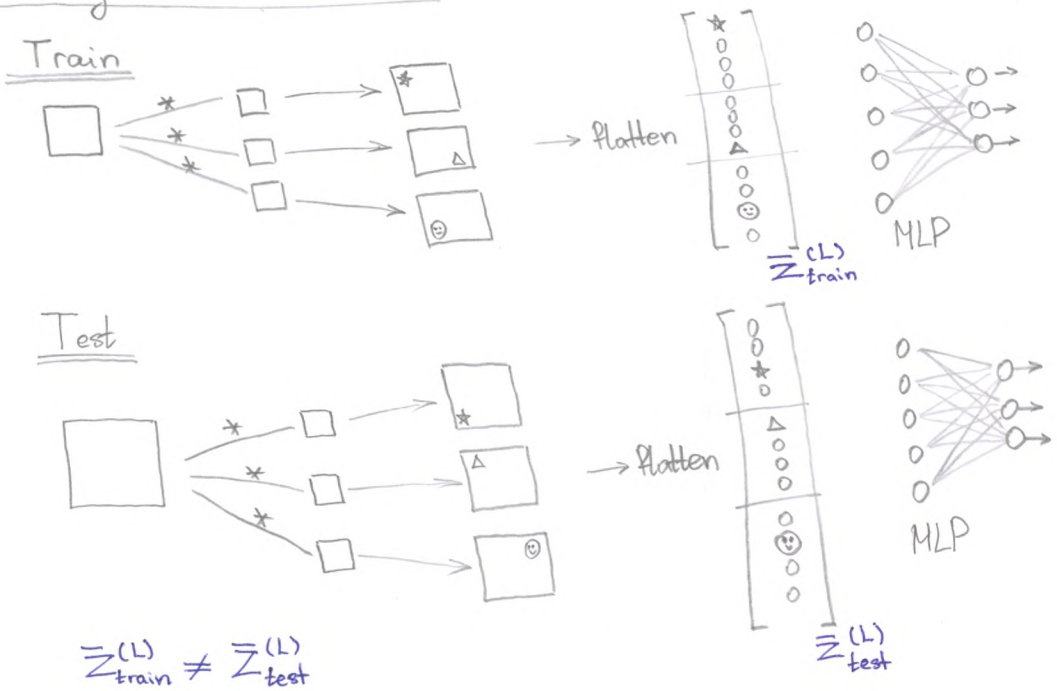
kernel size = input tensor size

$\begin{bmatrix} \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \\ 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix} \end{bmatrix}$

Global Max Pooling$^{(GMP)} \rightarrow 8$

Global Average Pooling$^{(GAP)} \rightarrow \frac{2 \sum_{i=1}^{8} i}{16}$

Why to use GMP



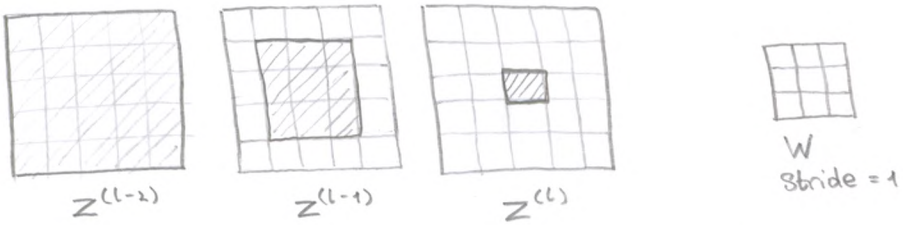
$z^{(L)}_{train} \neq z^{(L)}_{test}$

CNN эквивариантна, может выдавать хорошие feature maps, но если делать flatten, то там такая же проблема, как Image → MLP.

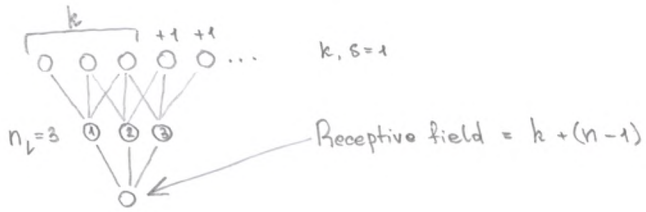
→ Fully Convolutional Neural Network

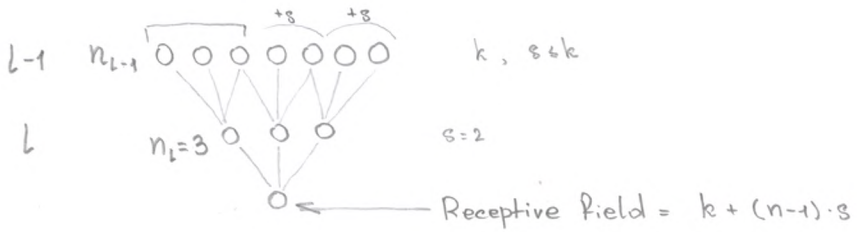
Receptive Field

$1 \to 3 \to 5$



Receptive field - это множество пикселей изображения от которых зависит активация определённого нейрона.





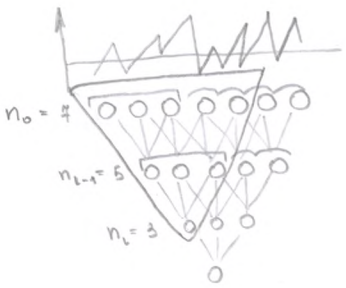
if k, s > k

k·n

Но это не реальное кол-во нейронов

$$n_{l-1} = k + (n_l - 1) \cdot s$$

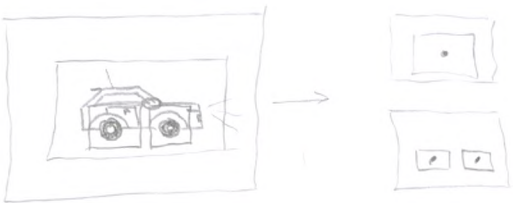
n_l - сколько нейронов l-го слоя видит нейрон



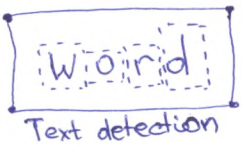
Dilated convolution увеличивает Receptive field но...

зная receptive field нейрона, можем ли мы определить rf следующего слоя за O(1)? Кажется, нет

В зависимости от Receptive field нейрон может распознавать паттерн определенного размера, или относительное расположение паттерна.



CTC for OCR



1. Classify 1M words. Problems: 1) if not in Domain, we can't classify, 2) A lot of images per each word
2. Slice into letters. Problems: 1) Hard task, 2) Annotation for each letter
3. Seq-to-seq. Problem: frame-wise alignment

End-to-end solution – Connectionist Temporal Classification
Изначально для Speech Recognition, Audio → Text
и RNN

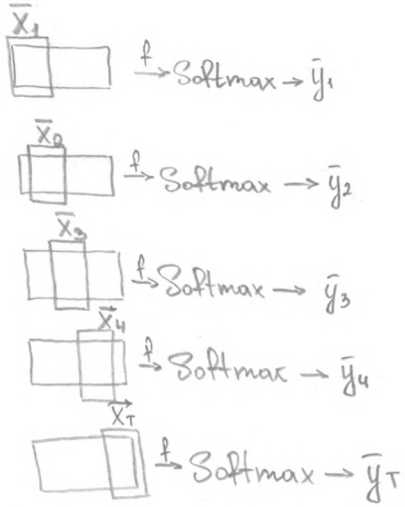
Мы используем Fully CNN

- L – конечный алфавит (ABC, . ; : ? etc.)
- $\mathcal{Z} = L^*$ – множество последовательностей символов L
  $\{A\}, \{B\}, \{C\}, \{AB\}, \{BA\}, \{ABC\}, \{ACB\}, \{BAC\}, \{BCA\}, \{CAB\}, \{CBA\}$
  То есть можно рассматривать, как слова.
- $\mathcal{X} = (R^m)^*$ – множество последовательностей m-мерных векторов.
  $\{[\,]^m, [\,]^m, [\,]^m, [\,]^m \dots\}$
  То есть это изображение $\in R^{m \times n}$, высотой m и произвольной шириной.
- S – множество обучающих примеров из распределения $D_{\mathcal{X} \times \mathcal{Z}}$

$f(\mathcal{X}) \to \mathcal{Z}$

$f((R^m)^*) \to L^*$

$f(\text{Image}) \to L^*$



$X = (\bar{x}_1, \bar{x}_2, \bar{x}_3, \dots, \bar{x}_T)$ – Input sequence

$Z = (z_1, z_2, z_3, \dots, z_U)$ – Target sequence

$U \leq T$

$(X, Z) \in S, \quad X \in \mathcal{X}, \quad Z \in \mathcal{Z}$

$f(X) \to Z$

T – кол-во шагов

– WW – – OOO – R – DDDD –

Z = WORD

Training a Classifier

L – alphabet

$\mathcal{Z} = L^*$ – sequence of letters

$\mathcal{X} = (\mathbb{R}^m)^*$ – sequence of image parts

S – множество $D_{\mathcal{X} \times \mathcal{Z}}$

$X = (\bar{x}_1, \dots, \bar{x}_T)$ – input sequence (image)

$Z = (z_1, \dots, z_u)$ – target sequence (word)

$u \leq T$

$(X, Z) \in S$, $X \in \mathcal{X}$, $Z \in \mathcal{Z}$

Train a classifier on S dataset

$$h : \underset{\text{images}}{\mathcal{X}} \to \underset{\text{words}}{\mathcal{Z}}$$

Label Error Rate (LER)

как оценивать ошибки

S' – validation dataset

$$LER(h, S') = \frac{1}{|S'|} \sum_{(X,Z) \in S'} ED(h(X), Z) \to \min$$

$ED(a, b)$ – минимальное кол-во операций вставки, удаления и замены, необходимых для получения последовательности b из a.

$ED(a = \text{bankt}, b = \text{bank}) = 1$

ED – Euclidean Distance?
Levenstein Distance?

CTC

$L' = L \cup \{blank\}$, blank (-) - пустой символ, межбуквенное пространство

L'^* - множество последовательностей.

i.g. $\bar{l} = (l_1, l_2, l_3 \dots, l_u)$ - target последовательность, $\bar{l} \in L^*$

$\bar{l}' = (-, l_1, -, l_2, -, l_3, -, \dots, -, l_u, -)$, $\bar{l}' \in L'^*$

$|\bar{l}'| = 2 \cdot |\bar{l}| + 1$

CNN → - C - N - N -

$\bar{x} = (\bar{x}_1, \bar{x}_2, \bar{x}_3, \dots, \bar{x}_T)$ - input sequence

$h(\bar{x}_t) \to \underset{softmax}{R^{|L'|}}$, $t \le T$ - i.e. предсказание на каждую букву алфавита + blank

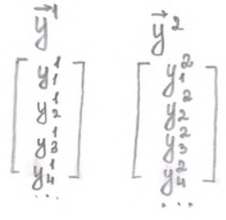
$y^t_{\pi_t}$ π_t - symbol from L'

Для каждого из t Softmax-ов у нас нет информации о label-ах, у нас нет разметки по буквам.

Assumption (strong)

$$p(\pi | x) = \prod_{t=1}^{T} y^t_{\pi_t} \quad , \forall \pi \in L'^T$$

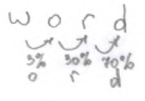
(π - path; $y^t_{\pi_t}$ - вероятность символа π_t на шаге t)



$\bar{\pi} = [\pi_1 \ \pi_2 \ \pi_3 \dots]^T$ (π_1 - symbol number)

- Как будту у $y^t_{\pi_t}$ стоит поменять верхний и нижний индекс местами, $\bar{y}^t$ - вектор на шаге t, $y^t_{\pi_t}$ - конкретное значение вектора

- Правдоподобие последовательности π - это произведение уверенности модели в символах идущих подряд, которые она предсказывает. Как Independent Events, но это кажется Dependent



Но, таким образом мы можем легко распознавать слова с ошибками или выдуманные

CTC

$L' = L \cup \{blank\}$ - алфавит $\quad \bar{Z} = L'^*$

L'^* - последовательность букв + {blank}

$\bar{I} \in L^*$, For example $\bar{I} = hello$

$\bar{I}' \in L'^*$ $\quad \bar{I}' = -h-e-l-l-o-$, $|\bar{I}'| = 2 \cdot |\bar{I}| + 1$

$X = (\bar{x}_1, \bar{x}_2, \bar{x}_3 \dots \bar{x}_T)$, T - width $\quad X \in \mathcal{X}$

$h(\bar{x}_t) \to \mathbb{R}^{|L'|}$ выдаём вероятность на каждую букву $\in L'$

$h(\bar{x}_t) = \bar{y}^t$

path

$\bar{\pi} = [\pi_1 \ \pi_2 \ \pi_3 \dots \pi_T]^T$, π_t - symbol

$p(\bar{\pi}|X) = \prod_{t=1}^{T} y^t_{\pi_t}$, $\forall \pi \in L'^T$

$L'^T \in L'^*$

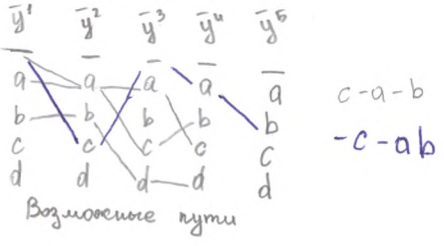
↑ послед. длины T

Decoding

$B: \bar{Z} \to Z$

1. Удаление повторов: AAA → A
2. Удаление {blank}: -AA-A → AAA

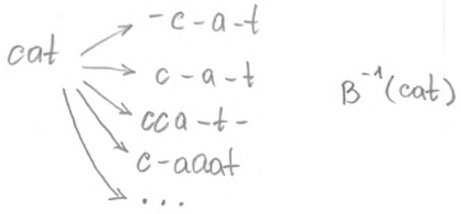
$B(-h-ee-ll-l-o) \to hello$



Возможные пути

$$p(\bar{I}|X) = \sum_{\bar{\pi} \in B^{-1}(I)} p(\bar{\pi}|X) \quad \text{- Likelihood}$$

Вероятность последовательности $\bar{I}$ при x - это сумма вероятностей всех предсказанных последовательностей, которые декодируются в $\bar{I}$, а не в одном конкретном пути.



Безумное количество возможных путей.

$$h(x) = \underset{\bar{I} \in L^{\leq T}}{\operatorname{argmax}} \, p(\bar{I}|X)$$ - наиболее вероятная последовательность, очевидно.

classifier

Forward - Backward Algorithm

$$\alpha_t(s) \overset{def}{=} \sum_{\vec{\pi} \in N^T:\ B(\pi_{1:t}) = B(l'_{1:s})} \underbrace{\prod_{t'=1}^{t} y^{t'}_{\pi_{t'}}}_{p(\vec{\pi} \mid x)}$$

$B(l'_{1:s}) \in L$

$N^T = L'^T, \quad L'^T \in L'^*$

Сумма правдоподобий всех путей декодирующихся в $l'_{1:s}$

l = hello

l' = -h-e-l-l-o-

$l'_{1:3}$ = -h-

π = h-ee-ll-l-oo-

$\alpha_t(s)$ - правдоподобность того, что первые t символов будут декодироваться в первые $\underset{s}{\downarrow}$ символов l'.

$B(\bar{y}^1 \ \bar{y}^2 \ \bar{y}^3 \ \dots \ \bar{y}^T)$

$B(- \ l_1 - l_2 - l_3 - \dots - l_u)$

$\bar{y}^1$ - softmax i.e. $[P(\text{blank}), P(A), P(B), P(C), \dots, P(z)]^T$

$\alpha_1(1) = y^1_b$

$\alpha_1(2) = y^1_{l_1}$

$\bar{y}^1 \to - \ l_1 -$ нам норм если мы в первой попытке сразу предскажем l_1

$\alpha_1(s) = 0, \quad \forall s > 2$

Одним предсказанием мы не можем предсказать 2 и более символов

Recursion

Forward - Backward Algorithm

$$\bar{\alpha}_t(s) \stackrel{\text{def}}{=} \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$$

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s)\, y^t_{l'_s}, & \text{if } l'_s = b, \text{ or } l'_{s-2} = l'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2))\, y^t_{l'_s}, & \text{otherwise} \end{cases}$$

– мы запрещаем пропускать blank.

– дополнительный случай.

1) $\alpha_t(s) = \bar{\alpha}_t(s)\, y^t_{l'_s}$, if $l'_s = b$ -H-E-L-L-O-

$\bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$

$\alpha_t(s)$ - вероятность, что первые t отсчетов будут декодироваться в первые s символов l'.

$\bar{\alpha}_t(s)$ - вероятность того, что $t-1$ отсчетов будут декодироваться в s, либо $s-1$ символов l'.

- $\alpha_{t-1}(s)$

$\pi_{1:t-1} =$ – – HH – – EE – } – t

$l'_{1:s} =$ -H-E- (s) nothing changes

- $\alpha_{t-1}(s-1)$

$\pi_{1:t-1} =$ – – HH – – EE } – t

$l'_{1:s-1}$ -H-E ($s-1$) } - s

2) $\alpha_t(s) = \bar{\alpha}_t(s)\, y^t_{l'_s}$, if $l'_s = l'_{s-2}$ -E-M-M-A

$\bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$

- $\alpha_{t-1}(s)$

$\pi_{1:t-1} =$ -EE-MM--M } M t

$l'_{1:s} =$ -E-M-M (s) nothing changes

- $\alpha_{t-1}(s-1)$

$\pi_{1:t-1} =$ -EE-MM-- } M t

$l'_{1:s-1} =$ -E-M- ($s-1$) } M s

Forward - Backward Algorithm

$$\bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$$

$$\alpha_t(s) = \sum_{\substack{\bar{\pi} \in N^T: \\ B(\pi_{1:t}) = B(l'_{1:s})}} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}}$$

$$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) \cdot y^t_{l'_s}, & \text{if } l'_s = b \text{ or } l'_{s-2} = l'_s \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) \cdot y^t_{l'_s}, & \text{otherwise} \end{cases}$$

3) General case

$$\alpha_t(s) = (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) \cdot y^t_{l'_s} = (\underline{\alpha_{t-1}(s)} + \underline{\alpha_{t-1}(s-1)} + \underline{\alpha_{t-1}(s-2)}) \cdot y^t_{l'_s}$$

· $\alpha_{t-1}(s)$

$\pi_{1:t-1} = -HH--EE--L\}L$ (t)

$l'_{1:s} = -H-E-L$ ($s = 6$)

$-H-E-L$

· $\alpha_{t-1}(s-1)$

$\pi_{1:t-1} = -HH--EE--\}L$ (t)

$l'_{1:s-1} = -H-E-$ ($s-1 = 5$)

$-H-E-L$

· $\alpha_{t-1}(s-2)$

$\pi_{1:t-1} = --HH--EE\}L$ (t)

$l_{1:s-2} = -H-E$ ($s-2 = 4$)

$-H-EL$

Дополнительный случай, когда мы можем разрешить пропустить blank.

В трех случаях $B(\pi_{1:t}) = HEL$, то есть последовательности декодируются в $l'_{1:s}$

$$\alpha_t(s) = \sum_{\substack{\bar{\pi} \in N^T: \\ B(\pi_{1:t}) = B(l'_{1:s})}} y^t_{\pi_t} \cdot \prod_{t'=1}^{t-1} y^{t'}_{\pi_{t'}} = |\pi_t = l'_s| = y^t_{l'_s} \cdot (\underline{\alpha_{t-1}(s)} + \underline{\alpha_{t-1}(s-1)} + \underline{\alpha_{t-1}(s-2)})$$

Forward - Backward Algorithm

Likelihood of particular path

$$p(\bar{\pi} \mid x) = \prod_{t=1}^{T} y_{\pi_t}^t$$

Likelihood of target sequence $\bar{l}$

$$p(\bar{l} \mid x) = \sum_{\pi \in B^{-1}(\bar{l})} p(\bar{\pi} \mid x) = \sum_{\pi \in B^{-1}(\bar{l})} \prod_t y_{\pi_t}^t = \sum_{\substack{\bar{\pi} \in N^T: \\ B(\pi_{1:T}) = B(l'_{1:|l'|})}} \prod_{t=1}^{T} y_{\pi_t}^t$$

сумма (~~всех~~) правдоподобий всех путей, декодирующихся в $\bar{l}$.

Likelihood of subsequence $\bar{l}'_{1:s}$

$$\alpha_t(s) = \sum_{\substack{\bar{\pi} \in N^T: \\ B(\pi_{1:t}) = B(l'_{1:s})}} \prod_{t'=1}^{t} y_{\pi_{t'}}^{t'} = \sum_{\substack{\bar{\pi} \in N^T: \\ B(\pi_{1:t}) = B(l'_{s:s})}} y_{\pi_t}^t \cdot \prod_{t'=1}^{t-1} y_{\pi_{t'}}^{t'}$$

$$\alpha_1(1) = y_b^1$$

$$\alpha_1(2) = y_{l_1}^1$$

$$\alpha_1(s) = 0, \quad \forall s > 0$$

$$\bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$$

$$\alpha_t(s) = \begin{cases} y_{l'_s}^t \cdot \bar{\alpha}_t(s), & \text{if } l'_s = b \text{ or } l'_s = l'_{s-2} \\ y_{l'_s}^t \cdot (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)), & \text{otherwise} \end{cases}$$

$-H-\{-$ $\quad -L-L\{L$

$-H\{-$ $\quad -L-\{L$

t , t

$-H-E-L\{L$

$-H-E-\{L$

$-H-E\{L$

t

Likelihood of target sequence $\bar{l}'$

$$p(\bar{l} \mid x) = \alpha_T(|\bar{l}'|) + \alpha_T(|\bar{l}'| - 1)$$

$-H-I-$ $\quad -H-I$

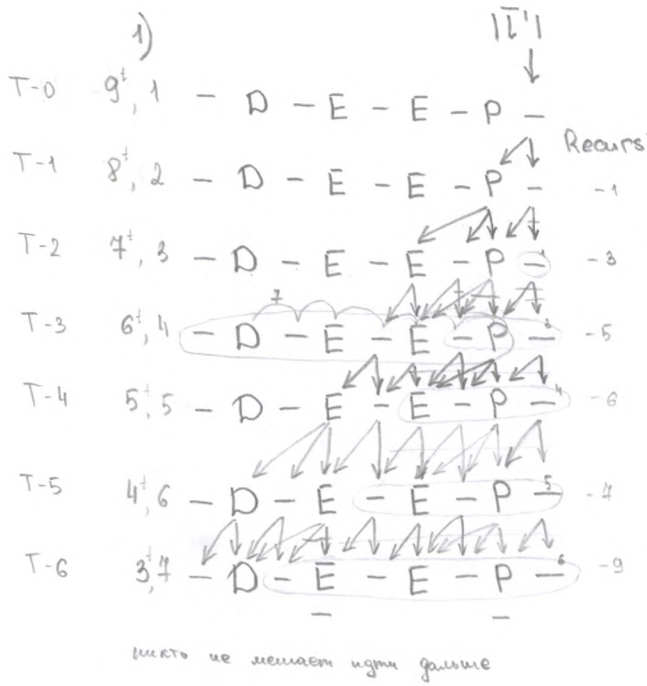
we can skip last blank

Image $m \times n$, $n > m$

Forward - Backward Algorithm

$$p(\bar{l} \mid x) = \alpha_T^{1)}(|\bar{l}'|) + \alpha_T^{2)}(|\bar{l}'|-1)$$

$$\alpha_t(s) = \begin{cases} y^t_{l'_s} \cdot (\alpha_{t-1}(s) + \alpha_{t-1}(s-1)) & \text{, if } l'_s = b \text{ or } l'_s = l'_{s-2} \\ y^t_{l'_s} \cdot (\alpha_{t-1}(s) + \alpha_{t-1}(s-1) + \alpha_{t-1}(s-2)) & \text{, otherwise} \end{cases}$$

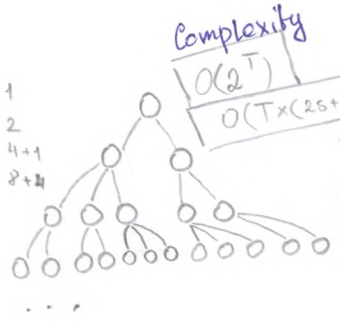


никто не мешает идти дальше

$$\alpha_T(|\bar{l}'|) = y^T_b \cdot (\alpha^{1)}_{T-1}(|\bar{l}'|) + \alpha^{2)}_{T-1}(|\bar{l}'|-1))$$

1) $\alpha_{T-1}(|\bar{l}'|) = y^{T-1}_{l'_{|l'|}}(\alpha_{T-2}(|l'|) + \underbrace{\alpha_{T-2}(|\bar{l}'|-1)}_{\text{dyn. prog.}})$

2) $\alpha_{T-1}(|\bar{l}'|-1) = y^{T-1}_{l'_{|l'|-1}}(\underbrace{\alpha_{T-2}(|l'|-1)}_{\text{dyn. prog.}} + \alpha_{T-2}(|l'|-2) + \alpha_{T-2}(|\bar{l}'|-3)$



$2 \cdot O(n^2)$ in best case

$2 \cdot O(n^3)$ in worst case

$O(\frac{n^3+n^2}{2})$ - average?

$-l_1-l_2-$
$\wedge\wedge\wedge\wedge\wedge$
$n^2\ n^3\ n^2\ n^3\ n^2$

or average just $O(n^3)$

но это не проход нейронкой

$T \times T - (S-1-\text{@})^2$

С дин прог опять посчитать всего 1 раз $\alpha_t(s)$ и всё

Если мы будем выбирать blank постоянно, например, то после какого-то момента мы не успеем уже слово распознать, потащу, кажется, можно отбросить некоторые ветки.

$T \leq |l'|$

1) if $\alpha_t(s)$, $t < s-1$ то 100% rr

$t: 1\ 2\ 3\ 4$
$s: 1\ 2\ 3\ 4\ 5$, but $t: 1\ 2\ 3\ 4$ $s: 1\ 2\ 3\ 4\ 5\ 6$

2) In example $\alpha_{T-2}(|l'|) = \alpha_7(9)$ не выполняется (это неправильно)

$\alpha_7(8)$ выполняется, но из-за того что $l'_8 \neq b$ and $l'_8 \neq l'_6$, иначе тоже бы не использовалось

$\alpha_7(9)$ тоже выполнялось бы, если бы не E-E

То есть не последовательные буквы позволяют нам считать 2 сразу, а не по одному

$t + (\text{кол-во не пост букв} - 1) < s-1$

$t < s-1-(2)$

$t < s-3$

$7 < 9-3 = 6$
$7 < 11-3 = 8$

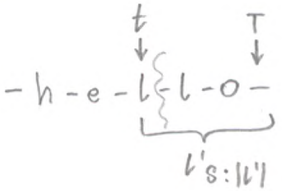
Pruning
if in $\alpha_t(s)$
$\Rightarrow$ $t < S-1$ – кол-во не пост букв, кроме первой
завершаем ветку (0)
не s, а остаток пути, это тоже самое

Мы всегда бинарные, кроме того когда буквы $l_t \neq l_{t-2}$ and $l_t \neq b$ (можем 1 или 2 делать)

Minimalno 1 буква = m pixels [a] [a|b] [ab]
1 2 ... m
необяз
может быть так

Forward-Backward Algorithm

$$\beta_t(s) = \sum_{\substack{\pi \in N^T : \\ B(\pi_{t:T}) = B(l'_{s:|l'|})}} \prod_{t'=t}^{T} y^{t'}_{\pi_{t'}}$$

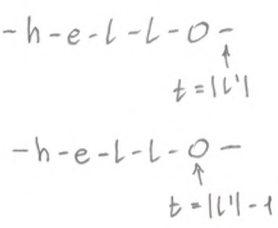


$$\beta_T(|l'|) = y^T_b$$

$$\beta_T(|l'|-1) = y^T_{l_{|l|}}$$

$$\beta_T(s) = 0 \;, \quad \forall s < |l'| - 1$$

Не успеем отгадать.



$\beta_t(s)$ - вероятность отгадать конечные s символов, начиная от t-го

$$\bar{\beta}_t(s) = \beta_{t+1}(s) + \beta_{t+1}(s+1)$$

$$\beta_t(s) = \begin{cases} \bar{\beta}_t(s) \cdot y^t_{l'_s} \;, & \text{if } l'_s = b \text{ or } l'_{s+2} = l'_s \\ (\bar{\beta}_t(s) + \beta_{t+1}(s+2)) \, y^t_{l'_s} \;, & \text{otherwise} \end{cases}$$

$\beta_{t+1}(s)$ $\quad -\{-l-l-o-$ ($l'_{s:|l'|}$)

$\beta_{t+1}(s+1)$ $\quad -\{l-l-o-$ ($l'_{s+1:|l'|}$)

if $l'_s = b$ or $l'_{s+2} = l'_s$

мы запрещаем пропускать blank

$\beta_{t+1}(s+2)$ $\quad e\{l-l-o-$ ($l'_{s+2:|l'|}$)

$e\{-l-l-o-$ ($l'_{s+1:|l'|}$)

$e\{e-l-l-o-$ ($l'_{s:|l'|}$)

при обычном распорядке,

мы разрешаем пропускать blank

Forward - Backward Algorithm

$\prod_x x = 0$, $x < 1$ – underflow

$C_t = \sum_s \alpha_t(s)$ $\qquad \hat{\alpha}_t(s) = \frac{\alpha_t(s)}{C_t}$

$D_t = \sum_s \beta_t(s)$ $\qquad \hat{\beta}_t(s) = \frac{\beta_t(s)}{D_t}$ - Normalized

нормирующий коэффициент

$\hat{\alpha}_t(s)$, $\hat{\beta}_t(s)$ – вероятность того, что на t отсчёте выдавать ~~(s-й символ)~~

$\hat{\alpha}_t(s)$ – выдавать первые s символов l'

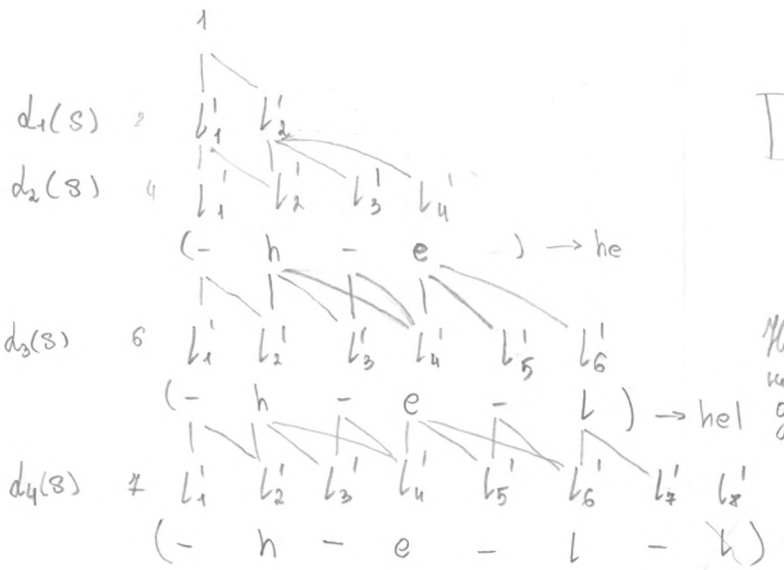
$\hat{\beta}_t(s)$ – выдавать от s символа до конца l'

Получается, нам в любом случае вычитывать все $\alpha_t(s)$ и $\beta_t(s)$. Сложность $O(T \cdot S)$.

Но, там $\alpha_1(2), \alpha_1(3) \dots \alpha_1(S) = 0$ и не только $t=1$.

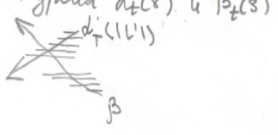
Последние $t < S-1-n$ $\alpha_t(S) = 0$, мы уже вышли, не успеем всё слово распознать

И первые? Обычно растет на 2, но если $\underbrace{l'_s = b \text{ or } l'_s = l'_{s-2}}_{L_t =}$, то растет на 1 всего



$2t$ - (кол-во $l_i = l_{i+1}$ до t)

Но, тут на сколько растет я думал, на самом деле рекурсия $\alpha_t(s)$ и $\beta_t(s)$ другая.



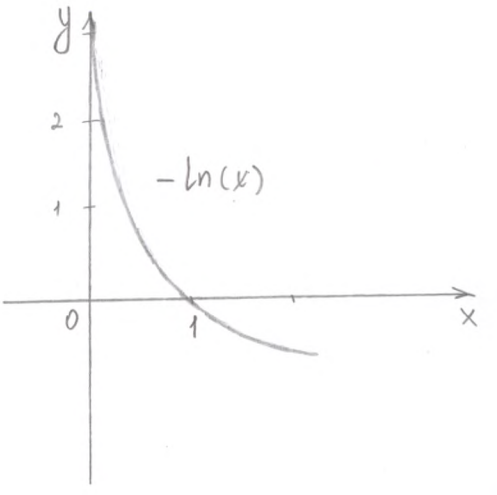
Forward - Backward Algorithm

Loss Function

$$P(S, \mathcal{N}_w) = \prod_{(x,z)\in S} p(z|x) \rightarrow \max$$

$$O^{ML}(S, \mathcal{N}_w) = -\sum_{(x,z)\in S} \ln(p(z|x))$$

$$-\ln(P(S, \mathcal{N}_w)) = -\sum_{(x,z)\in S} \ln(p(z|x)) \rightarrow \min$$



O - objective function

ML - maximum likelihood

S - dataset

$\mathcal{N}_w$ - model

x - input sequence.

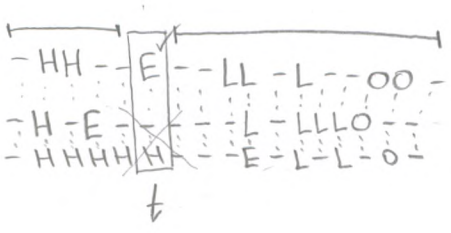
z - target sequence.

$$\frac{dO^{ML}(\{(x,z)\}, \mathcal{N}_w)}{dy_k^t} = \frac{-d\ln(p(z|x))}{dy_k^t}$$

partial derivative by some y_k^t

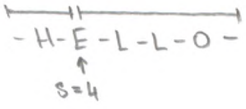
Forward - Backward Algorithm

$$\alpha_t(s) \cdot \beta_t(s) = \sum_{\pi \in B^{-1}(l):\ \pi_t = l'_s} y^t_{l'_s} \cdot \prod_{t'=1}^{T} y^{t'}_{\pi_{t'}}$$



$t = 6$

$l'_s = "E"$



$\alpha_t(s)$ - вероятность за t угадать до s, то есть $l'_{1:s}$

$\beta_t(s)$ - вероятность за t угадать от s до конца, то есть $l'_{s:|l'|}$

То есть $\pi_t = l'_s$, так как в и в $\alpha_t(s)$ и в $\beta_t(s)$ мы пытаемся вынуть $l's$ и рекурсивно проходим по всем возможным предыдущим путям, или последующим в случае $\beta_t(s)$.

$$\alpha_t(s) = \sum_{\bar{\pi} \in N^T:\ B(\bar{\pi}_{1:t}) = B(l'_{1:s})} \prod_{t'=1}^{t} y^{t'}_{\bar{\pi}_{t'}}$$

$$\beta_t(s) = \sum_{\bar{\pi} \in N^T:\ B(\pi_{t:T}) = B(l'_{s:|l'|})} \prod_{t'=t}^{T} y^{t'}_{\bar{\pi}_{t'}}$$

$$\alpha_t(s) = \sum_{\bar{\pi} \in N^T:\ B(\pi_{1:t}) = B(l'_{1:s})} y^t_{\pi_t} \prod_{t'=1}^{t-1} y^{t'}_{\pi_{t'}}$$

$$\beta_t(s) = \sum_{\bar{\pi} \in N^T:\ B(\pi_{t:T}) = B(l'_{s:|l'|})} y^t_{\pi_t} \prod_{t'=t+1}^{T} y^{t'}_{\pi_{t'}}$$

$$\alpha_t(s) \cdot \beta_t(s) = (abc + fgh) \cdot (cde + hij) = abc \cdot cde + fgh \cdot hij + \underbrace{abc \cdot hij + fghcde}_{\text{несовместимые последовательности}} =$$

$$= c\underbrace{(abcde)}_{\substack{\text{full}\\ \text{word}\\ \text{path}}} + h\underbrace{(fghij)}_{\substack{\text{full}\\ \text{word}\\ \text{path}}} + 0$$

$$\alpha_t(s)\cdot\beta_t(s) = \sum_{\substack{\pi\in B^{-1}(l):\\ \pi_t = l'_s}} \underbrace{y^t_{l'_s}}_{\text{const}} \cdot \underbrace{\prod_{t'=1}^{T} y^{t'}_{\pi_{t'}}}_{p(\pi|X)}$$

t = const
s = const

$\alpha_1(3) = 0$
$\alpha_2(5) = 0$
...

$$p(\pi|X) = \prod_{t=1}^{T} y^t_{\pi_t}, \quad \forall \pi \in L'^T, \quad L'^T \in L'^*$$

$$\alpha_t(s)\cdot\beta_t(s)\cdot\frac{1}{y^t_{l'_s}} = \sum_{\substack{\pi\in B^{-1}(l):\\ \pi_t = l'_s}} p(\pi|X)$$

1) $p(l|X) = \sum_{\pi\in B^{-1}(l)} p(\pi|X) =$, но тут t нет и как будто по всем t

2) $\boxed{p(l|X)_t = \sum_{s=1}^{|l'|} \frac{1}{y^t_{l'_s}}\cdot\alpha_t(s)\cdot\beta_t(s)}$, но тут t = const, так что 1) ≠ 2)

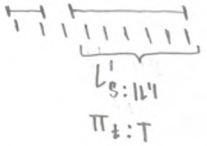
частный случай $p(l|X)$

$$\sum_{s=1}^{|l'|}\sum_{\substack{\pi\in B^{-1}(l):\\ \pi_t = l'_s}} p(\pi|X), \quad t$$

$$E = O^{ML}(S, \mathcal{X}_\omega) = -\sum_{(x,z)\in S} \ln(p(l|x))$$

Мы отбрасываем те t с которыми мы не сможем посчитать всё слово, там либо $\alpha_t(s) = 0$, либо $\beta_t(s) = 0$, соответственно $p(l|X)_t = 0$ ...?

Чем $\alpha_1(3) = 0$ $\beta_1(3) > 0$



Для любого t, $p(l|X)_t > 0$, если в целом $t \geq s-1-n$ для α и $\geq s$ для β

π_t

$T - t \geq (|l'|-s) - 1 - n$?

шаги = остаток -1(- скипы?)
скипы после s

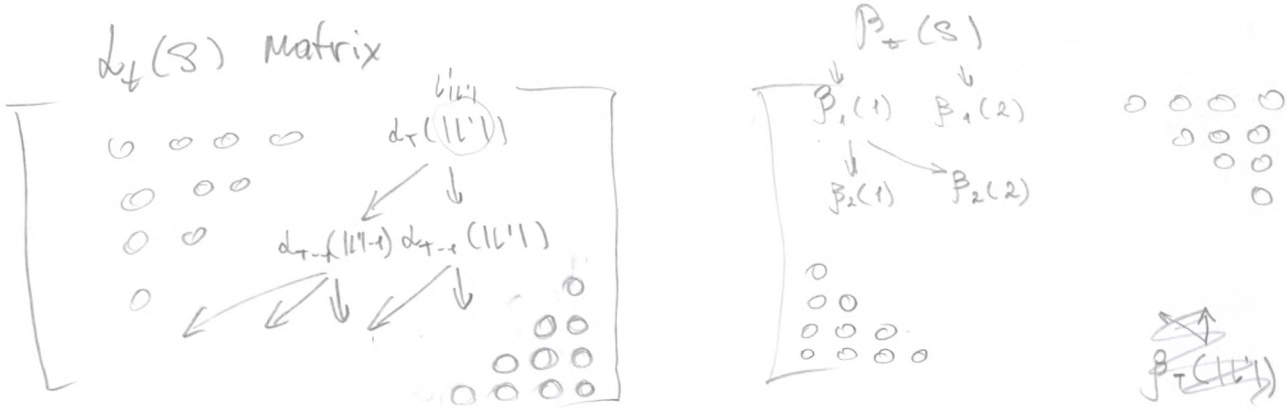
$s-3=6$ – чтобы до посл. blanka дойти
$6-1=5$ - blank
$5-1=4$ - skip

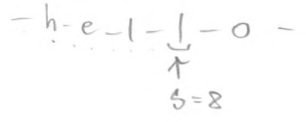
$$\frac{dO^{ML}}{dy^t_k} = \frac{d(-\sum \ln(p(l|x)))}{dy^t_k}$$

$$\nabla E_{y^t} = \begin{bmatrix} y^{t'}_k \\ y^{t'}_k \\ y^{t'}_k \\ y^{t'}_k \\ \dots \end{bmatrix}$$

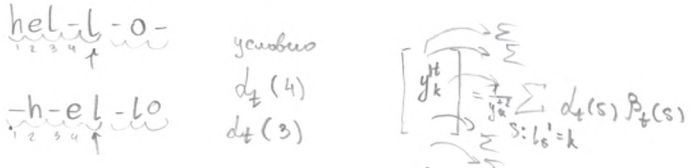
for t of 1 word

$$\nabla E_y = \sum_t \nabla E_{y^t}$$

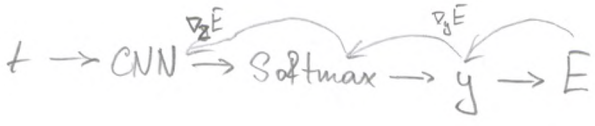




Рекурсивно вычисляем матрицы $\alpha_t(s)$ и $\beta_t(s)$.



Как быстро отсеять $s: l'_s = k$?
Видимо никак? только циклом?
Там такая сумма для каждой буквы алфавита нужна.



- Почему нужно и α и β?
- Включать ли пробелы? 80% за нет, иначе нужны RNN.
- В конце нужен Levenstein? или наиболее правдоподобное другое слово типа.
- Потенциально 'ь', 'ъ' и редкие буквы будут хуже распознаваться?

CTC Loss. Forward-Backward Algorithm

$p(\bar{\pi}|X) = \prod_{t=1}^{T} y^t_{\pi_t}$ - Likelihood of particular path $\bar{\pi}$

$\forall \pi \in L'^T$

$p(l|X) = \sum_{\bar{\pi} \in B^{-1}(l)} p(\bar{\pi}|X)$

$\mathcal{Z} = L^*$

$L' = \{\text{blank}\} \cup L$, $\bar{\mathcal{Z}} = L'^*$

$B : \bar{\mathcal{Z}} \to \mathcal{Z}$ - Decoding

$h(X = (\mathbb{R}^m)^*) = \mathcal{Z}$

$h(\bar{x}_t) = \bar{y}^t$, $\bar{y}^t \in \mathbb{R}^{|L'|}$

$\bar{y}^t$ - softmax i.e. $[P(\text{blank}), P(L_1) \dots]^T$

$h(X) = \underset{l \in L^{\le T}}{\operatorname{argmax}} p(l|X)$ - Classifier (Greedy)

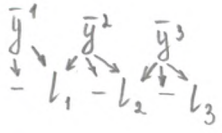
$\alpha_t(s) = \sum_{\bar{\pi} \in N^T : B(\pi_{1:t}) = B(l'_{1:s})} \prod_{t'=1}^{t} y^{t'}_{\pi_{t'}}$

Сумма правдоподобие того, что первые t символов будут декодироваться в первые s символов l'.

$\alpha_1(1) = y^1_b$

$\alpha_1(2) = y^1_{l_1}$

$\alpha_1(s) = 0$, $\forall s > 2$



Одним предсказанием не можем предсказать 2 символа

$\bar{\alpha}_t(s) = \alpha_{t-1}(s) + \alpha_{t-1}(s-1)$

$\alpha_t(s) = \begin{cases} \bar{\alpha}_t(s) \cdot y^t_{l'_s}, & \text{if } l'_s = b \text{ or } l'_s = l'_{s-2} \\ (\bar{\alpha}_t(s) + \alpha_{t-1}(s-2)) \, y^t_{l'_s}, & \text{otherwise} \end{cases}$

$\beta_t(s) = \sum_{\bar{\pi} \in N^T : B(\pi_{t:T}) = B(l'_{s:|l'|})} \prod_{t'=t}^{T} y^{t'}_{\pi_{t'}}$

$\beta_T(|l'|) = y^T_b$

$\beta_T(|l'|-1) = y^T_{l_{|l|}}$

$\beta_T(s) = 0$, $\forall s < |l'|-1$

$\bar{\beta}_t(s) = \beta_{t+1}(s) + \beta_{t+1}(s+1)$

$\beta_t(s) = \begin{cases} y^t_{l'_s} \bar{\beta}_t(s), & \text{if } l'_s = b \text{ or } l'_s = l'_{s+2} \\ y^t_{l'_s} (\bar{\beta}_t(s) + \beta_{t+1}(s+2)), & \text{otherwise} \end{cases}$

$p(\bar{l}|X) = \alpha_T(|l'|) + \alpha_T(|l'|-1)$

$- l_1 - \{ -$
$- l_1 \{ -$

$- l_1 - l_2 \{ l_2$
$- l_1 - \{ l_2$

$- l_1 - l_2 \{ l_2$
$- l_1 - \{ l_2$
$- l_1 \{ l_2$

$- l_1 - l_2 -$
$- \{ - l - l - 0 -$
$- \{ l - l - 0 -$

$O^{ML}(S, \mathcal{N}_w) = -\sum_{(x,z) \in S} \ln(p(z|x))$

$\alpha_t(s) \cdot \beta_t(s) = \sum_{\pi \in B^{-1}(l) : \pi_t = l'_s} y^t_{l'_s} \underbrace{\prod_{t'=1}^{T} y^{t'}_{\pi_{t'}}}_{p(\pi|x)}$

- - H - EE E - LL - L - 0 -

$p(l|X) = \sum_{s=1}^{|l'|} \sum_{\pi \in B^{-1}(l) : \pi_t = l'_s} p(\pi|X) = \sum_{s=1}^{|l'|} \frac{1}{y^t_{l'_s}} \cdot \alpha_t(s) \cdot \beta_t(s)$

$p'(l|x) = \sum \frac{1}{y^t_{l'_s}} \alpha_t(s) \beta_t(s)$

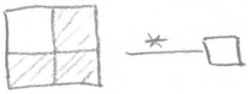
$\frac{dp(l|x)}{d y^t_k} = \frac{1}{(y^t_k)^2} \sum_{s \in lab(l,k)} \alpha_t(s) \beta_t(s)$ $\Rightarrow$ $\frac{dO^{ML}(S, \mathcal{N}_w)}{d y^t_k} = \frac{d(-\ln p(l|x))}{d y^t_k} = -\frac{1}{p(l|x)} \cdot \frac{dp(l|x)}{dy^t_k}$

$lab(l,k) = \{s : l'_s = k\}$ — set s where — must Symbols

For each word we'll get matrix of derivatives $T \times |L'| \to [\frac{\partial}{\partial}], [\frac{\partial}{\partial}] \dots$

В слове не всегда есть все буквы, они будут разделяющий 0-ми

Сложность $O(T \times S)$



w/2 h/2

Вычислительная сложность осталась такой же

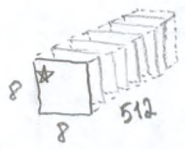
$3 + 2^{1} + 2^{2} + 2^{3} + 2^{4}$

$3 \quad 5 \quad 7 \quad 9$

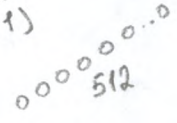
1) 3, 5 → 10
2) 12, 14 → 28
3) 30, 32 → 64
4) 66, 68 → 126

Receptive Field

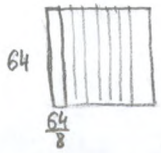
3	3×64×64
5	64×64×64
10	
12	64×32×32
14	128×32×32
28	
30	128×16×16
32	256×16×16
64	
66	256×8×8
68	512×8×8



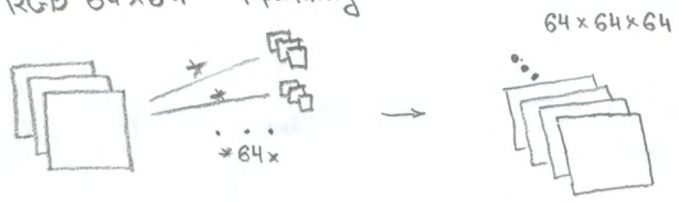
1) GMP (H, W)

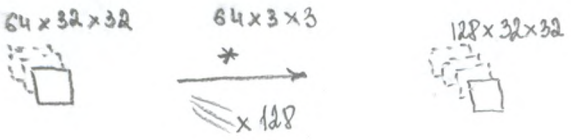


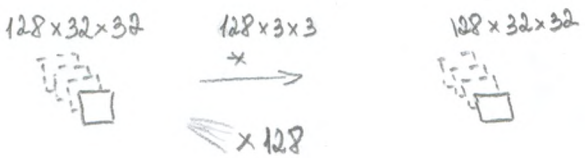
2) GMP (1, *)

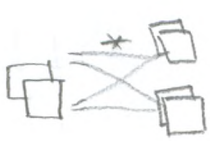


VGG









Computational Complexity remains the same

Машиналық оқыту бұл адамның араласуынсыз
өздігінен автоматты түрде талдайтын және
шешім қабылдайтын алгоритмдер мен деректерді
қамтитын жасанды интеллект қосымшасы.