

1)

$$y(x, w, b) = x^T w + b$$

L^2 -regularizace: „menší model“ ~ „lepší model“, takže
dráhá polehkově za hodně vyšších vah

$$MSE: \frac{1}{2} \sum_{i=1}^N (y(x_i, w) - t_i)^2 + \frac{\lambda}{2} \|w\|^2$$

Explicit solution of LR: -derivace podle j -té váhy

$$\frac{\partial}{\partial w_j} \frac{1}{2} \sum_{i=1}^N (x_i^T w - t_i)^2 = \frac{1}{2} \sum_{i=1}^N 2(x_i^T w - t_i) x_{ij} = \sum_{i=1}^N x_{ij} (x_i^T w - t_i)$$

$$\text{takže chci } \forall_j \sum_{i=1}^N x_{ij} (x_i^T w - t_i) = X_{*,j}^T (Xw - t) = 0 \quad \sim \text{pro všechny } j$$

$$X^T (Xw - t) = 0$$

$$X^T X w = X^T t$$

pokud $X^T X$ reg., existuje inverze

$$w = (X^T X)^{-1} X^T t$$

2.1)

SGD:

Snažíme se najít nejlepší váhy inkrementálním způsobem.

- chci minimalizovat error fci $\arg\min_w E(w)$, $\rightarrow$ learning-rate

pak gradient descent vypadá jako: $w = w - \alpha \nabla_w E(w)$

Mějme $X \in \mathbb{R}^{N \times D}$, $t \in \mathbb{R}^N$ trén data, $\hat{p}(x, t) = \frac{|\{i: (x_i, t) = (x, t)\}|}{N}$,

předpokládáme $\nabla_w E(w) = \mathbb{E}_{(x,t) \sim \hat{p}(x,t)} \nabla_w L(y(x, w), t)$

a) Standard GD: využitím všech trénovacích dat na výpočet $\nabla_w E(w)$

b) Stochastic GD: odhadem $\nabla_w E(w)$ pomocí náhodného vektoru. Unbiased, noisy.

$\rightarrow \nabla_w E(w) \approx \nabla_w L(y(x, w), t)$ pro náhodné $(x, t) \sim \hat{p}$

c) Mini-batch GD: trochu, $\nabla_w E(w)$ odhadem ze sample B hodnot z trén.

$\rightarrow \nabla_w E(w) \approx \frac{1}{|B|} \sum_{i \in B} \nabla_w L(y(x_i, w), t_i)$ pro náhodné $(x_i, t_i) \sim \hat{p}$

2.2)

L^2 -reg. minibatch SGD, $X \in \mathbb{R}^{N \times D}$, $t \in \mathbb{R}^N$, learning rate $\alpha \in \mathbb{R}^+$, L^2 -sila $\lambda \in \mathbb{R}$

$w = 0$ nebo random

until convergence:

- sample minibatch

$$-w = w - \alpha \cdot \sum_i \frac{1}{|b|} ((x_i^T w - t_i) x_i - \alpha \lambda w$$

2.3)

SGD, always returns the best?

Přijme learning rates α_i , asymptotický odhad $J(w)$ reálného $\nabla_w E(w)$:

$$w_{i+1} = w_i - \alpha_i J(w_i)$$

Pokud je ztr. fce L konvexní a spojitá, SGD konverguje do unitárního optimum skoro jistě, pokud learning rates splňují:

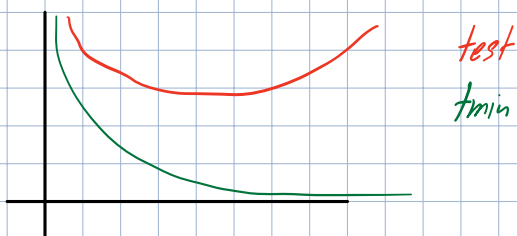
$$\forall_i \alpha_i > 0; \sum_i \alpha_i = \infty; \sum_i \alpha_i^2 < \infty$$

- 3. podmínka implikuje $\alpha_i \rightarrow 0$

- pro nekonzexní L lze garantovat pouze lokální optimum.

2.4) SGD: after-finish

Skončil jsem s: nízkou trénovací chybou, vysokou testovou chybou. Co dělat?



-> Buď začnu regularizovat nebo zastavím trénink dříve v „test údolí“

2.5) Fixed train and test, report model perform.

Jeden z hyperparametrů by byla $L2$ regularizace.

Pokud by šlo o LR, další by byl maximální stupňový polynom,

pokud o GD, velikost minibatche, max. počet iterací

2.6) Feature engineering:

1) MinMax scaling: $S(x) = \frac{x - \min}{\max - \min}$

2) Standard scaling: $S(x) = \frac{x - \mu}{\sigma}$

Features v rozdílných rozsazích by potřebovaly jiné learning rates

3.1) Binning classification:

Binární klasifikace: rozdělení vstupních dat do dvou tříd.

- lze dělat pomocí LR na základě thresholdu.

Perceptron alg: $(X \in \mathbb{R}^{N \times D}, t \in \{-1, +1\}^N)$, linear-separable

Výstup: $w \in \mathbb{R}^D$, s.t. $t_i x_i^T w > 0 \forall i$

alg:

$$w = 0$$

until all items correctly classified:

$$y = x_i^T w \quad \text{— prediction}$$

if $t_i y \leq 0$: — wrong prediction

$$w = w + t_i x_i$$

3.2) Entropy

Množství překvapení v celé distribuci.

$I(x)$... lidit informace v tom je

$$H(P) = \mathbb{E}_{x \sim P} [I(x)] = -\mathbb{E}_{x \sim P} [\log P(x)]$$

Pro diskrétní P :

$$H(P) = -\sum_x P(x) \cdot \log P(x)$$

Cross-Entropy:

$$H(P, Q) = -\mathbb{E}_{x \sim P} [\log Q(x)]$$

Gibbs - Inequality:

$$H(P, Q) \geq H(P)$$

$$-\mathbb{E}_{x \sim P} [\log Q(x)] = H(P, Q)$$

$$-\mathbb{E}_{x \sim P} [\log P(x)] = H(P)$$

$$H(P) = H(P, Q) \Leftrightarrow P = Q$$

KL-divergence:

$$D_{KL}(P \parallel Q) = H(P, Q) - H(P) = \mathbb{E}_{x \sim P} [\log P(x) - \log Q(x)]$$

Dk. Gibbs

$$\log x \leq (x-1), \quad \log x = x \text{ pouze pro } x=1$$

$$\text{Mějme: } H(P) - H(P, Q) = \sum_x P(x) \log \frac{Q(x)}{P(x)}$$

součet přes všechna prvky musí být 1.

$$\sum_x P(x) \log \frac{Q(x)}{P(x)} \leq \sum_x P(x) \cdot \left(\frac{Q(x)}{P(x)} - 1 \right) = \sum_x Q(x) - \sum_x P(x) = 0$$

Aby rovnost platila, musí být $\frac{Q(x)}{P(x)} = 1 \forall x$, tedy $P=Q$

Gibbs $\Rightarrow$ KL:

$$D_{KL}(P \parallel Q) \geq 0, \quad D_{KL}(P \parallel Q) = 0 \Leftrightarrow P=Q$$

3.3) Likelihood in MLE:

Jde o funkci, kterou mi řekne, s jakou pravděpodobností vidím daná data, mám-li daný parametrický model. Tedy $p(x_i; w)$ je pravděpodobnost, že vidím data point x_i , pokud má model parametry w . Najde o distribuci, prostě se snažíme $L(w)$ maximalizovat.

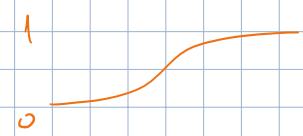
$$L(w) = p(X, w) = \prod_{i=1}^N p(x_i, w) \sim \text{likelihood, není to distr.}$$

3.4) MLE jako min NLL, CE, KL-div, určete vztahy

MLE w : „predict $\dagger$ when x given”

$$\begin{aligned} w_{MLE} &= \arg\max_w p(\dagger | X, w) = \arg\max_w \prod_{i=1}^N p(\dagger | x_i, w) \\ &= \arg\min_w \sum_i -\log p(\dagger | x_i, w) \quad \text{NLL} \\ &= \arg\min_w \mathbb{E}_{x \sim \hat{p}} [-\log p(\dagger | x, w)] \\ &= \arg\min_w H(\hat{p}(x), p(\dagger | x, w)) \quad \text{CE} \\ &= \arg\min_w D_{KL}(\hat{p}(x) \parallel p(\dagger | x, w)) \quad \text{KL-div} \end{aligned}$$

3.5) Logistic regression:

sigmoid: $\sigma(x) = \frac{1}{1+e^{-x}} \in (0,1) \forall x$: 

predictions: $p(C_1|x) = \sigma(x^T w + b)$

$p(C_0|x) = 1 - p(C_1|x)$

$\bar{y}(x, w) = x^T w \Rightarrow y(x, w) = \sigma(\bar{y}(x, w)) = \sigma(x^T w)$

→ jde o nashinovanou lin. predikci

logit: $\bar{y}(x, w) = \log\left(\frac{p(C_1, w)}{p(C_0, w)}\right)$

3.6) L^2 mini-batch SGD

alg:

$w = 0$

until convergence: process a batch B

$g = \frac{1}{|B|} \sum_{i \in B} \nabla_w (-\log(p(C_{t,i} | x_{i,w}))) + \lambda w$

$w = w - \alpha g$

loss function:

$\frac{1}{N} \sum_i -\log(p(C_{t,i} | x_{i,w})) + \frac{\lambda}{2} \|w\|^2$

gradient:

$(y(x) - t) x$

4.1) MSE jak MLE:

MSE: $\operatorname{argmin}_w \frac{1}{N} \sum_i (y(x_{i,w}) - t_i)^2$

Vezmeme distr. s nejvyšší entropií: normální rozdělení $\mathcal{N}(t_i; y(x_{i,w}), \sigma^2)$

s použitím MLE:

$\operatorname{argmax}_w p(t | X, w) = \operatorname{argmin}_w \sum_i^N -\log p(t_i | x_{i,w})$

$= \operatorname{argmin}_w - \sum_i^N \log\left(\sqrt{\frac{1}{2\pi\sigma^2}} \cdot e^{-\frac{(t_i - y(x_{i,w}))^2}{2\sigma^2}}\right)$

$= \operatorname{argmin}_w -N(\log(2\pi\sigma^2))^{-\frac{1}{2}} - \sum_i^N -\frac{(t_i - y(x_{i,w}))^2}{2\sigma^2}$

$= \operatorname{argmin}_w \sum_i^N \frac{(t_i - y(x_{i,w}))^2}{2\sigma^2} = \operatorname{argmin}_w \frac{1}{N} \sum_i^N (y(x_{i,w}) - t_i)^2$

4.2) Extenze logist. regrese:

váhy pro každou třídu: $W \in \mathbb{R}^{D \times K}$

softmax:

obecný sigmoid: $\text{softmax}(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$

alg: opět je predikce lin. část prohnání softmaxem.

$$p(C_i | x_i, W) = y(x_i, W) = \text{softmax}(\bar{y}(x_i, W))_i = \text{softmax}(x_i^T W)_i$$

$$\bar{y}(x_i, W) = \log(p(C_i | x_i, W)) + c$$

4.3) Sigmoid vs. Softmax

Sigmoid je speciální případ softmaxu:

$$\sigma(x) = \text{softmax}([x, 0])_0 = \frac{e^x}{e^x + e^0} = \frac{1}{1 + e^{-x}}$$

4.4) L^2 minibatch SGD pro k -label

alg: $X \in \mathbb{R}^{N \times D}$, $t \in \{0, \dots, k-1\}^N$, learning rate $\alpha \in \mathbb{R}^+$
 $w := W$

$w = 0$

until convergence: process minibatch B

$$g = \frac{1}{|B|} \sum_{i \in B} \nabla_w (-\log(p(C_{t,i} | x_i, w)))$$

$$w = w - \alpha g$$

gradient:

$$(y(x) - 1_+) x^T$$

loss function:

$$\frac{1}{N} \sum_i -\log(p(C_{t,i} | x_i, w))$$

4.5) Multilabel, convex decision regions

Rozhodovací regiony jsou konvexní. Uvažme dva body stejné kategorie x_a a x_b . Libovolný bod ležící na přímce $x = \lambda x_a + (1-\lambda) x_b$ je jejich konvexní kombinací díky linearnosti.

$$\bar{y}(x) = x^T W \Rightarrow \bar{y}(x) = \lambda \bar{y}(x_a) + (1-\lambda) \bar{y}(x_b)$$

Tím, že $\bar{y}(x_a)$ i $\bar{y}(x_b)$ mají největší k -tou složku, bude ji mít největší i $\bar{y}(x)$.

4.6) MLP

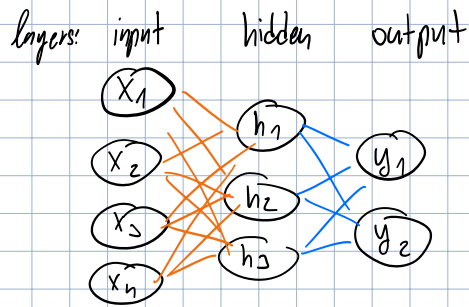
$$h_i = f\left(\sum_j x_j w_{ji}^{(h)} + b_i^{(h)}\right)$$

$$y_i = \alpha\left(\sum_j h_j w_{ji}^{(y)} + b_i^{(y)}\right)$$

což je maticově:

$$h = f(x^T W^{(h)} + b^{(h)})$$

$$y = \alpha(h^T W^{(y)} + b^{(y)}) \rightarrow \text{to jsou záměrné predice}$$



4.7) MLP output activations:

Binary: $\sigma(x)$ - Bernoulliho distribuce : $\frac{1}{1+e^{-x}}$

K-class: $\text{softmax}(x)$ - kategorie distr.: $\text{softmax}(x) \propto e^x$, $\text{softmax}(x)_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$

Hidden-layer:

$$\sigma(x)$$

$$\tanh(x) = 2\sigma(2x) - 1 \quad - \text{dělá } \sigma \text{ symetrické a vezme v nulu}$$

$$\text{ReLU}(x) = \max(0, x) \quad - \text{nejčastější nelineární}$$

5.1) Single-layer MLP, ReLU hidden, Softmax output

1) $\frac{\partial L}{\partial y}$ 2) $\frac{\partial y}{\partial y^{(in)}}$ $\rightarrow$ inputs to the output layer $y = a(y^{(in)})$

3) $\frac{\partial y^{(in)}}{\partial W^{(y)}}$ a $\frac{\partial y^{(in)}}{\partial b^{(y)}}$ $\rightarrow$ 2 čísel získaáme: $\frac{\partial L}{\partial W^{(y)}} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial y^{(in)}} \cdot \frac{\partial y^{(in)}}{\partial W^{(y)}}$
a stejné $\frac{\partial L}{\partial b^{(y)}}$

4) $\frac{\partial y^{(in)}}{\partial h}$ a $\frac{\partial h}{\partial h^{(in)}}$

5) $\frac{\partial h^{(in)}}{\partial W^{(h)}}$ a $\frac{\partial h^{(in)}}{\partial b^{(h)}}$ $\rightarrow$ 2 čísel získaáme: $\frac{\partial L}{\partial W^{(h)}} = \frac{\partial L}{\partial y} \cdot \frac{\partial y}{\partial y^{(in)}} \cdot \frac{\partial y^{(in)}}{\partial h} \cdot \frac{\partial h}{\partial h^{(in)}} \cdot \frac{\partial h^{(in)}}{\partial W^{(h)}}$
a stejné $\frac{\partial L}{\partial b^{(h)}}$

5.2) Universal approx. theorem

Mějme $\varphi(x): \mathbb{R} \rightarrow \mathbb{R}$ nekonzstantní, omezenou a neklesající spojitou funkci.

Pro libovolné $\varepsilon > 0$ a libovolnou spojitou $f: [0,1]^D \rightarrow \mathbb{R}$, existuje $H \in \mathbb{N}$, $v \in \mathbb{R}^H$, $b \in \mathbb{R}^H$ a $W \in \mathbb{R}^{D \times H}$ takové že pokud

$$F(x) = v^T \varphi(x^T W + b) = \sum_{i=1}^H v_i \varphi(x^T W_{*,i} + b_i)$$

kde φ je použita „elementwise“,

$$\text{pak } \forall x \in [0,1]^D: |F(x) - f(x)| < \varepsilon$$

5.3) Minimum search in $f(x): \mathbb{R}^D \rightarrow \mathbb{R}$

Mějme $f(x): \mathbb{R}^D \rightarrow \mathbb{R}$ s minimem v x s podmínkami

$g_1(x) = 0, \dots, g_n(x) = 0$. $f, g_1, \dots, g_n$ necht' mají spoj. parc. der.

a necht' $\nabla_x g_1(x), \dots, \nabla_x g_n(x)$ jsou lin. nez.

Pak existují $\lambda_1, \dots, \lambda_m \in \mathbb{R}$ t.č.

$$\mathcal{L}(x, \lambda) = f(x) - \sum_{i=1}^m \lambda_i g_i(x) \text{ má nulový gradient v } x \text{ i } \lambda.$$

5.4) Categorical distr. and maximum entropy proof:

Máme $p = (p_1, \dots, p_n)$, u které chceme maximalizovat entropii:

Tedy chceme minimalizovat $-H(p)$ za podmínek:

$$\forall i: p_i \geq 0 \rightarrow \text{tu budeme ignorovat}$$

$$\sum_i p_i = 1$$

$$L = \left(\sum_i p_i \log p_i \right) - \lambda \cdot \left(\sum_i p_i - 1 \right), \text{ z čehož lze odvodit:}$$

$$0 = \frac{\partial L}{\partial p_i} = 1 \cdot \log p_i + p_i \cdot \frac{1}{p_i} - \lambda = \log p_i + 1 - \lambda$$

$$\text{tedy } p_i = e^{\lambda-1} \text{ bude stejné pro všechna } i \text{ a } \sum_i p_i = 1 \Rightarrow p_i = \frac{1}{n}$$

5.5)

Softmax derivation using max. entropy principle

Chceme minimalizovat $-\sum_i^N H(\pi(x_i))$ za podmínek:

$$1) \text{ pro } 1 \leq i \leq N, 1 \leq k \leq K: \pi(x_i)_k \geq 0$$

$$2) \text{ pro } 1 \leq i \leq N: \sum_{k=1}^K \pi(x_i)_k = 1$$

$$3) \text{ pro } 1 \leq j \leq D, 1 \leq k \leq K: \sum_{i=1}^N \pi(x_i)_k x_{ij} = \sum_{i=1}^N [\mathbb{1}_{\{t_i=k\}}] x_{ij}$$

Máme tedy:

$$\begin{aligned} L = & \sum_i^N \sum_k^K \pi(x_i)_k \log(\pi(x_i)_k) \\ & - \sum_j^D \sum_k^K \lambda_{jk} \left(\sum_i^N \pi(x_i)_k x_{ij} - [\mathbb{1}_{\{t_i=k\}}] x_{ij} \right) \\ & - \sum_i^N \beta_i \left(\sum_k^K \pi(x_i)_k - 1 \right) \end{aligned}$$

5.6) Precision, Recall, F_1 a F_β score

$$\text{Precision} = TP / (TP + FP), \text{ Recall} = TP / (TP + FN), \text{ Accuracy} = (T^*) / (T^* + F^*)$$

$$F_1 = 2 \cdot \text{precision} \cdot \text{recall} / (\text{precision} + \text{recall}), F_\beta = (1 + \beta^2) \cdot \text{precision} \cdot \text{recall} / (\beta^2 \cdot \text{precision} + \text{recall})$$

5.7) micro F_1 vs macro F_1

Micro F_1 : Nejprve sečtn všechny TP, FP, FN všech samostatných binárních klasifikací a vypočítám jedno F_1 -score.

Macro F_1 : Nejprve vypočítám všechnu jednotlivou F_1 -score a pak udělám jejich průměr.

5.8) Why accuracy is not always good

Accuracy není vhodná, pokud se ve vzorku přirozeně vyskytuje hodně správných bodů. Když mám velmi ojedinelou nemoc, test může jen fixně říkat „negativní“ a bude mít velmi vysokou přesnost.

6.1) TF, IDF

$$TF(t, d) = \frac{\# \text{ výskytů } t \text{ v } d}{\# \text{ termů v } d}, \quad IDF(t) = \log \left(\frac{\# \text{ dokumentů } d}{\# \text{ dokumentů s } t} \right)$$

TF-IDF = TF · IDF → jak moc je důležitý term pro dokument v korpusu

6.2) Conditional entropy:

vars $x, y : x \sim X, y \sim Y$ → v diskrétním případě

Conditional entropy: $H(Y|X) = E_{x,y} [I(y|x)] = \sum_{x,y} P(x,y) \cdot \log P(y|x)$

Mutual info: $I(X,Y) = E_{x,y} \left[\log \frac{P(x,y)}{P(x) \cdot P(y)} \right]$

$$I(X,Y) = I(Y,X) = H(Y) - H(Y|X) = H(X) - H(X|Y)$$

$$I(X,Y) = D_{KL}(P(X,Y) \parallel P(X) \cdot P(Y))$$

$$\Rightarrow I(X,Y) \geq 0$$

$$I(X,Y) = 0 \Leftrightarrow P(X,Y) = P(X) \cdot P(Y), \text{ tedy jsou nezávislé.}$$

6.3) TF-IDF as mutual info

Mějme D dokumentů a T termů. Kdyžkoliv vytáhujeme dokument, děláme to uniformně.

$$P(d) = 1/|D| \text{ a } I(d) = H(D) = \log |D|$$

$$P(d|t \in d) = 1/|\{d \in D : t \in d\}|$$

$$I(d|t \in d) = H(D|t) = \log |\{d \in D : t \in d\}| \quad 0 \cdot \log 0 = 0$$

$$I(d) - I(d|t \in d) = H(D) - H(D|t) = \log \frac{|D|}{|\{d \in D, t \in d\}|} = \text{IDF}(t)$$

Tedy $I(D, T)$:

$$= \sum_{d, t \in d} P(d) \cdot P(t|d) \cdot (I(d) - I(d|t)) = \frac{1}{|D|} \sum_{d, t \in d} \text{TF}(t, d) \cdot \text{IDF}(t)$$

6.4) Word embedding:

Místo abych slova reprezentovali sparse one-hot vektory, reprezentují je reálný vekt. čísel, které ve vektorovém prostoru svou vekt. hodnotou reprezentují podobnost / spříznost mezi daty.

V MLP learningu máme vektor slova jako vstup pro trénování. Vstup učíme pracovat s contextem slov, což činí MLP lépe pochopit význam. Vstup to dovoluje analyzovat komplexní syntaktické a sémantické konstrukce.

6.5) SkipGram: Negative sampling

Rozdělíme output matici W na output embedding v .

Pro vstupní slovo w s emb. e_w a kontextovým slovem c s v_c provedeme logistickou regresi a zavedeme loss fci

$$-\log \sigma(e_w^T v_c)$$

Pak nasamplujeme U negativních samplů c_i které nejsou v kontextovém okně a negativně je přidáme do loss fce:

$$-\sum_i \log \sigma(-e_w^T v_{c_i})$$

Distribuce, ze které samplujeme je heuristicky upravený kat. distr. založený na frekv. slov.

6.6) Pre-trained word-embeddings

Použil bych sliding okno embeddingů a klasifikoval bych podle prostředního.

7.1) k -nearest neighbors

Regression: k nejbližších sousedů má hodnotu t_i a váhy w_i :

$$\hat{t} = \sum_i \frac{w_i}{\sum_j w_j} \cdot t_i$$

Classification: nejčastější třída je mojí předikcí

$$t_i \in \mathbb{R}^k: \hat{t} = \sum_i \frac{w_i}{\sum_j w_j} t_i, \text{ predicted je: } \underset{h}{\operatorname{argmax}} \sum_i w_i t_i$$

L_p norm: $x, y \in \mathbb{R}^D$, vzdálenost je $\|x - y\|_p$, kde

$$\|x\|_p = \left(\sum_i |x_i|^p \right)^{1/p}$$

Váhy: - uniformní

- inverzní $\rightarrow$ proporcionalitní k $1/\text{vzdálenost}$

- softmax $\rightarrow$ proporcionalitní k softmax(-distance)

7.2) L^2 jako Bayesian inference:

$$w_{\text{MAP}} = \underset{w}{\operatorname{argmax}} p(X|w) \cdot p(w)$$

$$= \underset{w}{\operatorname{argmax}} \prod_{i=1}^N p(x_i|w) p(w)$$

$$= \underset{w}{\operatorname{argmin}} \sum_i -\log p(x_i|w) - \log p(w)$$

po substituci Gaussian prioru

$$w_{\text{MAP}} = \underset{w}{\operatorname{argmin}} \sum_i \left(-\log p(x_i|w) + \frac{D}{2} \log(2\pi\sigma^2) + \frac{\|w\|^2}{2\sigma^2} \right)$$

$\uparrow$ to je L^2 regularizace

7.3) $p(C_w|x)$ under Naive Bayes

Modelováni distribuce $p(x|C_w)$ je složité, protože x může být vysokodimenzionální.

Předpoklad: Všechny x_d jsou nezávislé se zbytkem C_w .

Pak lze připsat: $p(x|C_w) = p(x_1|C_w) \cdot p(x_2|C_w, x_1) \dots$

Předpokládáme: $\underset{h}{\operatorname{argmax}} p(x|C_w) \cdot p(C_w)$

$$\text{ne: } p(x|C_w) = \prod_{d=1}^D p(x_d|C_w).$$

8.1) Independent discrete vars are uncorrelated

$$\text{Nehorelování} := \text{Cov}(X, Y) = 0$$

$$\text{Cov}(X, Y) = E[(X - E(X)) \cdot (Y - E(Y))]$$

$$= \sum_{x, y} P(x, y) (X - E(X)) \cdot (Y - E(Y))$$

protože nezávislé

$$= \sum_x P(x) \cdot (x - E(x)) \cdot \sum_y P(y) (y - E(y))$$

$$= E_x [X - E[X]] E_y [Y - E[Y]] = \underline{\underline{0}}$$

8.2) Variance a Pearsonův koef.

Variance: $\text{Cov}(X, Y) = E[(X - E(X)) \cdot (Y - E(Y))]$

Pearson:

$$\rho = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)} \cdot \sqrt{\text{Var}(Y)}} \in \langle -1, 1 \rangle$$

8.3) Spearman korelace a Kendallův muk

Spearman ρ je počítán je měřen na rangích, kde muk elementu je jeho index ve vzestupně seřazené posloupnosti.

Kendall T měří objem koncordantních pářů ($y \uparrow / \downarrow$ když to stejné dělá x)

$$= \frac{\sum_{i < j} \text{sign}(x_j - x_i) \cdot \text{sign}(y_j - y_i)}{\binom{n}{2}} \in \langle -1, 1 \rangle$$

8.4) Correlation as evaluation metric

Vhodné využít na:

- vyhodnocení metriků řešení
- měření kvalit algoritmy
- měření kvalit automatických metrik v porovnání s lidskou metrikou.

8.5) Assess validity of evaluation metric

Mám-li větší hodnocené data, chci měřit metrikou, která bude nejvíce „borelout“ s většími hodnoceními. Tam kde to rozhodně použiju.

8.6) Cohen's K

Cohen's K vyjadřuje **inter-annotator agreement (IAA)**

$$k = \frac{P_o - P_e}{1 - P_e}$$

, kde P_o je observed agreement,
 P_e je expected agreement

V machine-learningu se používá pro filtrování matoucích dat a chybných anotací.
dříve ne všichni outliers jsou šum...

8.7) Ensembling MSE:

Mějme $\varepsilon_i(x) :=$ chybu i -tého modelu od skutečnosti:

$$\text{MSE ensemble} = E\left[\left(\frac{1}{M} \sum \varepsilon_i(x)\right)^2\right]$$

Jelikož mají ε_i průměr 0 a jsou nezávislé, musí platit:

$$E[\varepsilon_i(x) \varepsilon_j(x)] = 0 : i \neq j$$

a tedy

$$E\left[\left(\frac{1}{M} \sum \varepsilon_i(x)\right)^2\right] = E\left[\frac{1}{M^2} \sum_{i,j} \varepsilon_i(x) \varepsilon_j(x)\right] = \frac{1}{M} E\left[\frac{1}{M} \sum \varepsilon_i^2(x)\right]$$

takže chyba ensemble je $\frac{1}{M}$ krát menší.

8.8) Knowledge distillation

Velký ensemble může být moc velký nebo pomalý. Pro to pak potřebuji model, co mimikuje ten velký.

1) Nalézáme velký $P_{\text{teacher}}(y|x,w)$

2) Nalézáme malý model s $H(P_{\text{student}}(y|x,w), P_{\text{teacher}}(y|x,w))$ jako trénovacím cílem.

9.1) Decision - tree:

Ve vnitřních uzlech jsou držené podmínky, které určí další dělení.

V listech jsou reprezentanti výsledných tříd. Σ - jako průměr prvků, co tam spadly.

Criterion popisuje, jak moc vhodné jsou data ve vrcholu.

$$C_{SE} := \sum_{i \in I_T} (t_i - \bar{t}_T)^2, \text{ kde } \bar{t}_T = \frac{1}{|I_T|} \sum_{i \in I_T} t_i.$$

Splituj tak, abych každým uzlem reprezentoval nejmenší
solution marking, které na vstupní mám. Snadno se minimalizovat
 $C_R + C_L - C_T$

9.2)

9.3) Multiclass:

Ve vnitřních vrcholech držím podmínku, podle které splituju.

V listech držím listy, co mi tam spadly.

Předpokládám v listu takovou třídu, kterou je v něm nejvícejší.

$$P_T(k) := \text{prob. třídy } k \text{ v regionu } T.$$

Opět splituju podle $C_R + C_L - C_T$.

Gini index: jak často by byl náhodně zvolený element
náhodně oblíbený podle $P_T(k)$ špatně oblíbený.

$$C_{\text{gini}}(T) = |I_T| \sum_k P_T(k)(1 - P_T(k))$$

Entropy criterion

$$C_{\text{entropy}}(T) = |I_T| \cdot H(P_T) = -|I_T| \sum_{P_T(k) \neq 0} P_T(k) \cdot \log P_T(k)$$

9.4) Binary Cross

$$n_T(0) = \# \text{ elements se annotation } 0, \quad n_T(1) = \dots$$

$$p_T = \frac{1}{|I_T|} \sum_{i \in I_T} t_i = \frac{n_T(1)}{n_T(0) + n_T(1)}$$

$$\text{Sum of squares: } L(p) = \sum_{i \in I_T} (p - t_i)^2$$

$$\Leftrightarrow p, \text{ které minimalizuje loss, je } p = p_T \quad p_T^2 - 2p_T + 1$$

$$L(p_T) = \sum_{i \in I_T} (p_T - t_i)^2 = n_T(0) \cdot (p_T - 0)^2 + n_T(1) \cdot (p_T - 1)^2$$

$$= \frac{n_T(0) \cdot n_T(1)^2}{(n_T(0) + n_T(1))^2} + \frac{n_T(1) \cdot n_T(0)^2}{(n_T(0) + n_T(1))^2} = \frac{(n_T(1) + n_T(0)) \cdot n_T(0) \cdot n_T(1)}{(n_T(0) + n_T(1)) \cdot (n_T(0) + n_T(1))} =$$

$$= (n_T(0) + n_T(1)) \cdot (1 - p_T) \cdot p_T = |I_T| \cdot p_T \cdot (1 - p_T)$$

9.5) Entropy criterion derivation:

$$n_T(k) := \# \text{ elements s targetem } k \vee T.$$

$$p_T(k) = \frac{1}{|I_T|} \sum_{i \in I_T} [t_i = k] = \frac{n_T(k)}{|I_T|}$$

Uvažme distr. p na $\mathcal{U}$ a NLL loss $L(p) = \sum_{i \in I_T} -\log p_{t_i}$.
Minimalizujme $p - p_T(k)$.

$$L(p_T) = \sum_{i \in I_T} -\log p_{t_i} = - \sum_{k: p_T(k) \neq 0} n_T(k) \log p_T(k)$$

$$= -|I_T| \sum_{k: p_T(k) \neq 0} p_T(k) \log p_T(k) = |I_T| \cdot H(p_T)$$

9.6) Stromy se trénují odděleně, jejich trénovací data nejsou stejné.

a) Bagging

- vytvoříme si ze vstupní množiny více množin tím, že do nich data uniformně náhodně vybíráme s opakováním.

b) Random subset

- v každém splitu náhodně zahrádíme některá data, když vyhodnotíme split.

Volíme pak softwar pro regresi, hardwar pro klasifikaci.

10.1)

$$E^{(t)}(w_+, w_{1..t-1}) = \sum_i \left[\ell(t_i, y^{(t-1)}(x_i, w_{1..t-1}) + y_+(x_i, w_+)) \right] + \frac{\lambda}{2} \|w_+\|^2$$

$$g_i = \frac{\partial \ell(t_i, y^{(t-1)}(x_i))}{\partial y^{(t-1)}(x_i)}, \quad h_i = \frac{\partial^2 \ell(t_i, y^{(t-1)}(x_i))}{\partial y^{(t-1)}(x_i)^2}$$

$$w_T^* = - \frac{\sum_{i \in I_T} g_i}{\lambda + \sum_{i \in I_T} h_i}$$

$$\text{Criterion: } -\frac{1}{2} \sum_T \frac{\left(\sum_{i \in I_T} g_i \right)^2}{\lambda + \sum_{i \in I_T} h_i} + \text{const}$$

10.2)

Pro každý timestep trénují u stromů $w_{t,u}$, kde každý představuje single lin. část.

Predikce je pak prováděna jako:

$$\text{softmax}(y(x_i)) = \text{softmax}\left(\sum_{t=1}^T y_{t,1}(x_i, w_{t,1}), \dots, \sum_{t=1}^T y_{t,u}(x_i, w_{t,u})\right)$$

a loss je tedy per-example:

$$\ell(t_i, y(x_i)) = -\log(\text{softmax}(y(x_i))_{t_i})$$



10.3)

Vhodné pro low-dimensional data, kde každá feature nese informace o celé výzvě.

- protože čím dříve rozdělíme data podle jejich feature, tím dříve feature nese informaci

11.1)

Libovolnou matici X velikosti $m \times n$, hodností r , lze faktorizovat jako:

$$X = U \Sigma V^T, \text{ kde}$$

U je ortonormální $m \times m$

Σ je diagonální $m \times n$ s nezápornými hodnotami, v sestupném pořadí

V je ortonormální $n \times n$

U jsou komponenty řádků, V jsou komponenty sloupců.

$$X = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \dots + \sigma_r u_r v_r^T \quad (X \text{ má hodnost } r)$$

Redukční verze SVD: vyhodíme všechna $\sigma_k : k > r$ a použijeme menší U, V .

11.2)

$X \in \mathbb{R}^{n \times m}$, $X_k = \sigma_1 u_1 v_1^T + \dots + \sigma_k u_k v_k^T$ je aproximace pomocí SVD.

$\forall B \in \mathbb{R}^{n \times m}$ s $\text{rank} = k$:

$$\|X - X_k\|_F \leq \|X - B\|_F$$

11.3)

Nechť $S = \frac{1}{N} (X - \bar{X})^T (X - \bar{X})$, pak PCA X -ln jsou vlastní vektory S ,

mean-centered

tedy V matice SVD dekompozice $X - \bar{X}$, pouze přesklopení $\frac{1}{N}$

11.4)

- spočítat mean μ řádku X
- spočítat kovarianční matici: $S = \frac{1}{N} (X - \mu)^T (X - \mu)$
- for i in $\{1, \dots, M\}$
 - $v_i = \text{náhodné}$
 - repeat until convergence
 - $v_i = S v_i$
 - $\lambda_i = \|v_i\|$
 - $v_i = v_i / \lambda_i$
 - $S = S - \lambda_i v_i v_i^T$
- return XV , kde sloupce V jsou $v_1, v_2, \dots, v_m$

11.5)

- initialize $\mu_1, \dots, \mu_k$ randomly
- repeat until convergence:
 - compute best possible $z_{i,k} = \begin{cases} 1 & \text{if } k = \arg\min_j \|x_i - \mu_j\|^2 \\ 0 & \text{otherwise} \end{cases}$

→ pokud k -tý bod je nejbližší
 - compute best possible $\mu_k = \arg\min_{\mu} \sum_i z_{i,k} \|x_i - \mu\|^2$
 - což jako derivaci podle μ dává: $\mu_k = \frac{\sum_i z_{i,k} x_i}{\sum_i z_{i,k}}$

→ per-iteration avg.
- k-means++ $\rightarrow \mu_1 - \mu_k$ jsou inicializovány tak, že μ_1 náhodně, ostatní proporcionalně k 2. mocnině vzdálenosti od středu nejbližšího clusteru

12.1)

	H_0	H_1
H_0	true negative	false negative type II
$\neg H_0$	false positive type I	true positive

significance level = type I error rate

12.2)

Test-statistics: je to jako „shrnutí“ viděných dat, dovolí rozlišovat nulovou a alt. hypotézu.

p-values: past. řídkost test-stat. hodnoty alespoň tak extrémní jako té aktuálně viděné, předpokládající platit nulovou hypotézu.

- low p-value := velmi malá past. při nulové hypotéze

12.3)

- 1) Formulace H_0 , alt. i H_1
- 2) zvolení test. statistiky
- 3) Spočítání hodnoty test-statistiky
- 4) Spočítání p-value
- 5) rejecting H_0 pokud p-value je menší než zvolený α (at most 5%)

12.4)

One-sample samplejeme z jedné distribuce,

Two-sample samplejeme ze dvou distribucí,

Paired-test samplejeme ze dvou distribucí, ale jejich hodnoty párujeme

12.5)

$$\text{FWER} = P\left(\bigcup_i (p_i \leq \alpha)\right)$$

Bonferroni correction: rejects H_0 pokud v rodině velikosti m $p_i \leq \frac{\alpha}{m}$

jelikož $P\left(\bigcup_i A_i\right) \leq \sum_i P(A_i)$

$$\text{FWER} = P\left(\bigcup_i p_i \leq \frac{\alpha}{m}\right) \leq \sum_i P\left(p_i \leq \frac{\alpha}{m}\right) = m \cdot \frac{\alpha}{m} = \alpha$$

12.6)

perf = [] $\rightarrow$ získáme jako result 2,5-97,5 percentily

repeat R times:

- sample N test sets with replacement, together with predictions
- measure performance using metric E and append to perf

12.7)

diffs = []

repeat R times:

- sample N test sets with replacement, together with predictions
- measure performance of the models y and z on the sampled data using metrics E , append their diff to diffs

return the ratio of the diffs that are ≤ 0

12.8) Myslenka je:

Jsou-li modely stejně dobré, je jedno jestli vezmeme predikci z prvního nebo druhého.

Pak uvažujeme-li všechny možné výběry združených predikcí, máme distr. perform. za předpokladu, že jsou modely stejně dobré.

Potom, p -value je quantile perform. dotazovaného modelu.