Spis treści

1 Rachunek prawdopodobieństwa

1.1 Definicja przestrzeni probabilistycznej

Rozkładem prawdopodobieństwa P w pewnym zbiorze zdarzeń elementarnych $\Omega \neq \emptyset$ nazywamy odwzorowanie

$$P : \Sigma \mapsto [0; 1] \,,$$

gdzie Σ jest rodziną podzbiorów Ω (inaczej rodziną zdarzeń) taką, że

$$\Omega \in \Sigma \,, \quad A \in \Sigma \implies A' \in \Sigma \,, \quad \forall A_1, A_2, \ldots \in \Sigma : \bigcup_i A_i \in \Sigma \,,$$

które spełnia: $P(\Omega) = 1$ oraz dla dowolnych parami rozłącznych zdarzeń $A_1, A_2, \ldots$ należących do Σ zachodzi

$$P\left(\bigcup_i A_i\right) = \sum_i P(A_i) \,.$$

Trójkę (Ω, Σ, P) nazywamy przestrzenią probabilistyczną. Z powyższej definicji wynikają znane własności prawdopodobieństwa tj. $P(A') = 1 - P(A)$ oraz $P(A \cup B) = P(A) + P(B) - P(A, B)$.

1.2 Prawdopodobieństwo warunkowe i niezależność zdarzeń

Definiujemy prawdopodobieństwo warunkowe zdarzenia A pod warunkiem zdarzenia B o dodatnim prawdopodobieństwie jako

$$P(A \mid B) := \frac{P(A, B)}{P(B)} \,.$$

Zdarzenia $A_1, .., A_n$ nazwiemy niezależnymi iff

$$P(A_1, .., A_n) = P(A_1) \cdot P(A_2) \cdot ... \cdot P(A_n) \,.$$

Zdarzenia $A_1, .., A_n$ nazwiemy *warunkowo niezależnymi* względem zdarzenia B iff

$$P(A_1, .., A_n \mid B) = P(A_1 \mid B) \cdot ... \cdot P(A_n \mid B) \,.$$

1.3 Prawdopodobieństwo całkowite

Jeśli zdarzenia $A_1, A_2, \ldots \in \Sigma$ są parami rozłączne tj. $\forall i \neq j : A_i \cap A_j = \emptyset$ i zachodzi $\bigcup_i A_i = \Omega$ to zbiór zdarzeń $\{A_i\}$ nazywamy *układem zupełnym zdarzeń.*

Jeśli $\{A_i\}$ jest układem zupełnym zdarzeń to prawdopodobieństwo całkowite dowolnego zdarzenia $B \in \Sigma$ jest dane przez

$$P(B) = \sum_i P(B \mid A_i)P(A_i)\,.$$

1.4 Twierdzenie Bayesa

Jeśli $\{A_i\}$ jest układem zupełnym zdarzeń to z definicji prawdopodobieństwa warunkowego zachodzi

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} = \frac{P(B \mid A)P(A)}{\sum_i P(B \mid A_i)P(A_i)}$$

1.5 Reguła wielokrotnego warunkowania

Dla dowolnych zdarzeń $A_1, .., A_n \in \Sigma$ zachodzi

$$\begin{aligned}
&P(A_1, .., A_n) = P(A_n \mid A_{n-1}, .., A_2, A_1) \cdot P(A_{n-1}, .., A_2, A_1) \\
&= P(A_n \mid A_{n-1}, .., A_2, A_1) \cdot P(A_{n-1} \mid A_{n-2}, .., A_2, A_1) \cdot P(A_{n-2}, .., A_2, A_1) \\
&= ... \\
&= P(A_n \mid A_{n-1}, .., A_2, A_1) \cdot P(A_{n-1} \mid A_{n-2}, .., A_2, A_1) \cdot ... \cdot P(A_2 \mid A_1) \cdot P(A_1)
\end{aligned}$$

1.6 Zmienne losowe

W uczeniu maszynowym będą interesować nas zmienne o wartościach w $\mathbb{R}^n$. Zmienne takie nazywamy zmiennymi losowymi wielowymiarowymi i definiujemy jako odwzorowania

$$X : \Omega \mapsto \mathbb{R}^n$$

takie, że dla każdego $A \subseteq \mathbb{R}^n$ zbiór $\{\omega \in \Omega \mid X(\omega) \in A\}$ należy do rodziny zdarzeń Σ. Przy takiej definicji prawdopodobieństwo, iż zmienna X ma wartość należącą do pewnego przedziału A wynosi

$$P(X \in A) = P(\{\omega \in \Omega \mid X(\omega) \in A\})\,.$$

Dowolny rozkład prawdopodobieństwa zmiennej losowej n–wymiarowej $X = (X_1, X_2, \ldots, X_n)$ jest wyznaczony jednoznacznie przez zadanie funkcji $F(\mathbf{x}) : \mathbb{R}^n \mapsto [0; 1]$ zwanej dystrybuantą zdefiniowanej jako

$$F(\mathbf{x}) = F(x_1, \ldots, x_n) := P(X_1 \leq x_1, \ldots, X_n \leq x_n)\,.$$

Zasadniczo będą nas interesować jednak dwa przypadki rozkładów prawdopodobieństwa zmiennych losowych: rozkłady dyskretne i rozkłady ciągłe. W przypadku rozkładu dyskretnego istnieje pewien przeliczalny zbiór $S \subset \mathbb{R}^n$ taki, że $P(X \in S) = 1$. Rozkład ten jest zadany jednoznacznie przez podanie $|S|$ liczb $p_i > 0$ określających prawdopodobieństwa $p_i = P(X = \mathbf{x}_i)$ dla wszystkich $\mathbf{x}_i \in S$. W przypadku rozkładu ciągłego istnieje z kolei funkcja $p(\mathbf{x}) : \mathbb{R}^n \mapsto [0; \infty)$ taka, że

$$P(X_1 \in [a_1; b_1], \ldots, X_n \in [a_n; b_n]) = \int_{a_1}^{b_1} \cdots \int_{a_n}^{b_n} p(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} \, .$$

Funkcje $p(\mathbf{x})$ nazywamy gęstością prawdopodobieństwa. W obu przypadkach musi być spełniony warunek unormowania postaci odpowiednio

$$\sum_i p_i = 1 \, , \quad \int_{\mathbb{R}^n} p(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} = 1 \, .$$

Będziemy często wykorzystywać wartość oczekiwaną pewnej funkcji $f(\mathbf{x})$ zmiennej losowej X zdefiniowaną odpowiednio dla rozkładu p – dyskretnego lub ciągłego jako

$$\mathbb{E}[f(\mathbf{x})] := \sum_{\mathbf{x}_i \in S} f(\mathbf{x}_i) p_i \cong \int_{\mathbb{R}^n} f(\mathbf{x}) p(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} \, .$$

Zauważmy przy tym, iż funkcja $f(\mathbf{x})$ może być zupełnie dowolna, np. dla funkcji charakterystycznej (indykatorowej) zbioru $A \subset \mathbb{R}^n$ $f(\mathbf{x}) = \mathcal{I}_A$ mamy $\mathbb{E}[\mathcal{I}_A(\mathbf{x})] = P(X \in A)$ lub dla iloczynu funkcji Heaviside'a $f(\mathbf{x}) = \theta(t_1 - x_1) \cdots \theta(t_n - x_n)$ mamy $\mathbb{E}[f(\mathbf{x})] = F(t_1, \ldots, t_n)$.

1.7 Rozkłady brzegowe

Niech $X = (X_1, \ldots, X_n)$ będzie n–wymiarową zmienną losową o dystrybuancie $F(\mathbf{x})$. Rozkład brzegowy względem k zmiennych $X_{\sigma(1)}, \ldots, X_{\sigma(k)}$ definiujemy jako rozkład wyznaczony przez dystrybuantę

$$F_{X_{\sigma(1)}, \ldots, X_{\sigma(k)}}(x_{\sigma(1)}, \ldots, x_{\sigma(k)}) := \lim_{x_{\sigma(k+1)} \to \infty, \ldots, x_{\sigma(n)} \to \infty} F(x_1, \ldots, x_n) \, .$$

1.8 Zmienne losowe niezależne

Niech $X = (X_1, \ldots, X_k)$ będzie n–wymiarową zmienną losową o rozkładzie wyznaczonym przez dystrybuantę $F(\mathbf{x})$. Powiemy, iż zmienne losowe $n_1, \ldots, n_k$ – wymiarowych $(n_1 + \ldots + n_k = n)$ $X_1, \ldots, X_k$ są niezależne iff dla dowolnych $\mathbf{x}_1 \in \mathbb{R}^{n_1}, \ldots, \mathbf{x}_k \in \mathbb{R}^{n_k}$ zachodzi

$$F(\mathbf{x}_1, \ldots, \mathbf{x}_k) = F_{X_1}(\mathbf{x}_1) \cdot \ldots \cdot F_{X_k}(\mathbf{x}_k) \, .$$

1.9 Funkcja tworząca momenty

Funkcją tworzącą momenty (z ang. *moment generating function*) nazywamy funkcję określoną wzorem

$$M_X(t) := \mathbb{E}\left[\mathrm{e}^{tX}\right]$$

dla zmiennej losowej rzeczywistej $X : \Sigma \mapsto \mathbb{R}$. Powyższa wielkość jest określona zawsze tylko dla $t = 0$. Dla innych t, $M_X(t)$ może w ogólności nie istnieć. MGF używamy, aby łatwiej obliczać momenty różnych rzędów. Istotnie jeśli $M_X(t)$ istnieje w pewnym otoczeniu $t = 0$ to

$$\mathbb{E}[X^k] = \frac{\mathrm{d}^k M_X}{\mathrm{d}t^k}\Big|_{t=0}.$$

Jednocześnie łatwo pokazać, że zachodzi $M_X(at) = M_{aX}(t)$ oraz $M_{X+b}(t) = M_X(t)\mathrm{e}^{tb}$.

1.10 Funkcja charakterystyczna

Funkcję charakterystyczną rozkładu o gęstości $p(x)$ definiujemy jako

$$\varphi_X(x) = \mathbb{E}\left[\mathrm{e}^{\mathrm{i}tx}\right] = \int_{-\infty}^{+\infty} p(x)\mathrm{e}^{\mathrm{i}tx}\,\mathrm{d}x\ .$$

Widzimy, iż jest to zatem transformata Fouriera funkcji gęstości. Funkcja charakterystyczna koduje pełną informację o rozkładzie i możemy wyciągną z niej funkcję gęstości przez zastosowanie odwrotnej transformacji Fouriera.

1.11 Rozkłady warunkowe

W ogólnym przypadku zmiennej losowej n – wymiarowej $Z = (Z_1, \ldots, Z_n)$ o ciągłym rozkładzie $p(\mathbf{z})$ jeśli wydzielimy zmienne k i $n-k$ – wymiarowe $X = (Z_{\sigma(1)}, \ldots, Z_{\sigma(k)})$, $Y = (Z_{\sigma(k+1)}, \ldots, Z_{\sigma(n)})$ to rozkład warunkowy zmiennej $X \mid Y$ definiujemy jako rozkład zadany przez gęstość prawdopodobieństwa

$$p(\mathbf{x} \mid \mathbf{y}) := \frac{p(\mathbf{z})}{p_Y(\mathbf{y})} = \frac{p(\mathbf{x}, \mathbf{y})}{p_Y(\mathbf{y})}\,.$$

1.12 Transformacja zmiennych wielowymiarowych

Niech $X = (X_1, \ldots, X_n)$ będzie zmienną losową wielowymiarową o rozkładzie ciągłym o gęstości $p_X(\mathbf{x})$. Rozważmy bijekcję $(X_1, \ldots, X_n) \mapsto (Y_1, \ldots, Y_n)$. Chcemy

znaleźć wyrażenie na gęstość $p_Y(\mathbf{y})$ w nowych zmiennych. Ponieważ infinitezymalne prawdopodobieństwo jest niezmiennicze względem zmiany współrzędnych więc zachodzi

$$p_X(x_1, \ldots, x_n)\, \mathrm{d}x_1 \ldots \mathrm{d}x_n = p_Y(y_1, \ldots, y_n)\, \mathrm{d}y_1 \ldots \mathrm{d}y_n \ ,$$

skąd

$$p_Y(y_1, \ldots, y_n) = \left| \frac{\partial(x_1, \ldots, x_n)}{\partial(y_1, \ldots, y_n)} \right| p_X(x_1(\mathbf{y}), \ldots, x_n(\mathbf{y}))\,.$$

1.13 Macierz kowariancji

Macierz kowariancji funkcji $f(\mathbf{x})$ zmiennej losowej X definiujemy jako

$$\mathbf{\Sigma}[f(\mathbf{x})] := \mathbb{E}\left[(f(\mathbf{x}) - \boldsymbol{\mu}_f)(f(\mathbf{x}) - \boldsymbol{\mu}_f)^T\right] \ ,$$

gdzie $\boldsymbol{\mu}_f = \mathbb{E}[f(\mathbf{x})]$. Elementy diagonalne Σ_{ii} tej macierzy nazywamy wariancjami zmiennych X_i, natomiast elementy pozadiagonalne Σ_{ij} nazywamy kowariancjami zmiennych X_i i X_j. Oczywiście $\mathbf{\Sigma}$ jest macierzą symetryczną. Nadto jeśli f jest funkcją identycznościową tj. $f(\mathbf{x}) = \mathbf{x}$ to $\mathbf{\Sigma}$ jest macierzą nieujemnie określoną, gdyż dla dowolnego $\mathbf{v} \in \mathbb{R}^n$ mamy

$$\mathbf{v}^T \mathbf{\Sigma} \mathbf{v} = \mathbb{E}[\mathbf{v}^T(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T\mathbf{v}] = \mathbb{E}[z^2] \geq 0\,,$$

gdzie $z = \mathbf{v}^T(\mathbf{x} - \boldsymbol{\mu}) \in \mathbb{R}$. Jeśli $X_1, \ldots, X_n$ są niezależne i f jest funkcją identycznościową to $\mathbf{\Sigma}$ jest macierzą diagonalną.

1.14 Wielowymiarowy rozkład normalny

Jeśli zmienna wielowymiarowa $X = (X_1, \ldots, X_n)$ ma wielowymiarowy rozkład normalny (z ang. *Multivariate Normal distribution – MVN*) z wartością oczekiwaną $\boldsymbol{\mu}$ i macierzą kowariancji $\mathbf{\Sigma}$, co oznaczamy jako $X \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$, to gęstość prawdopodobieństwa jest dana

$$\phi(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^n \det \mathbf{\Sigma}}} \exp\left\{ -\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu}) \right\}$$

Macierz $\mathbf{\Lambda} = \mathbf{\Sigma}^{-1}$ nazywamy macierzą precyzji. Jeśli $\mathbf{v}_i$ są unormowanymi wektorami własnymi macierzy $\mathbf{\Sigma}$, a λ_i odpowiadającymi im wartościami własnymi i zakładając, iż widmo $\{\lambda_i\}$ jest niezdegenerowane mamy z twierdzenia spektralnego

$$\mathbf{\Lambda} = \sum_{i=1}^{n} \frac{1}{\lambda_i} \mathbf{v}_i \mathbf{v}_i^T$$

oraz wiemy, iż wektory $\{\mathbf{v}_i\}$ tworzą bazę ortonormalną przestrzeni $\mathbb{R}^n$. Z powyższego możemy zatem wyrazić wektor $\mathbf{x}-\boldsymbol{\mu}$ jako kombinację liniową wektorów $\{\mathbf{v}_i\}$ tj.

$$\mathbf{x}-\boldsymbol{\mu}=\sum_{i=1}^{n} t_i\mathbf{v}_i\,,$$

co pozwala zapisać gęstość prawdopodobieństwa jako

$$\phi(t_1,\ldots,t_2)\cong\exp\left\{-\frac{1}{2}\sum_{i=1}^{n}\frac{t_i^2}{\lambda_i}\right\}\,.$$

Z powyższego wzoru widać, iż poziomice gęstości są wielowymiarowymi elipsoidami, których półosie są skierowane wzdłuż wektorów własnych $\boldsymbol{\Sigma}$ i mają długości proporcjonalne do $\sqrt{\lambda_i}$.

Powiemy, iż wielowymiarowa zmienna losowa $X\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ ma standardowy wielowymiarowy rozkład normalny jeśli $\boldsymbol{\mu}=\mathbf{0}$ i $\boldsymbol{\Sigma}=\mathbf{1}$. Wówczas

$$\phi(\mathbf{x})=\frac{1}{\sqrt{(2\pi)^n}}\exp\left\{-\frac{1}{2}\mathbf{x}^T\mathbf{x}\right\}\,.$$

Można wykazać, iż jeśli $X\sim\mathcal{N}(\boldsymbol{\mu},\boldsymbol{\Sigma})$ dla $\boldsymbol{\Sigma}$ o niezdegenerowanym widmie to wszystkie rozkłady brzegowe i warunkowe X są rozkładami normalnymi.

1.15 Zbieżność w rachunku prawdopodobieństwa

W rachunku prawdopodobieństwa definiujemy trzy zasadnicze rodzaje zbieżności ciągu zmiennych losowych (X_n).

- Ciąg (X_n) jest zbieżny do X stochastycznie iff

$$\forall\epsilon>0:\lim_{n\to\infty}P(|X_n-X|<\epsilon)=1\,.$$

- Ciąg (X_n) jest zbieżny do X z prawdopodobieństwem 1 iff

$$P\left(\lim_{n\to\infty}X_n=X\right)=1\,.$$

- Ciąg (X_n) n–wymiarowych zmiennych losowych jest zbieżny do X według dystrybuant iff

$$\forall\mathbf{x}\in\mathbb{R}^n, F_X(\mathbf{x})\text{ – ciągła w }\mathbf{x}:\lim_{n\to\infty}F_{X_n}(\mathbf{x})=F_X(\mathbf{x})$$

Pomiędzy tak zdefiniowanymi rodzajami zbieżności zachodzą następujące implikacje:

1. $X_n \to X$ z prawdopodobieństwem 1 $\implies$ $X_n \to X$ stochastycznie
2. $X_n \to X$ stochastycznie $\implies$ $X_n \to X$ według dystrybuant
3. $X_n \to X$ stochastycznie $\implies$ istnieje podciąg (X_{n_k}) zbieżny do X z prawdopodobieństwem 1

1.16 Rozkłady prawdopodobieństwa

1.16.1 Rozkład Bernoulliego

Jeśli zmienna losowa ma wartości w zbiorze $\{x_1, x_2\}$ oraz $P(X = x_1) = p$ i $P(X = x_2) = 1 - p$ (dla $0 \leq p \leq 1$) to mówimy, że $X \sim \text{Ber}(p)$ (zmienna X ma rozkład Bernoulliego z parametrem p).

1.16.2 Rozkład dwumianowy

Próbą Bernoulliego nazywamy doświadczenie losowe, którego wynik $X \sim \text{Ber}(p)$. Schematem dwumianowym nazywamy n–krotne powtórzenie próby Bernoulliego przy założeniu, iż poszczególne próby są niezależne. Jeśli S będzie zmienną losową o wartościach w $\mathbb{N} \cup \{0\}$, która opisuje liczbę sukcesów w schemacie dwumianowym długości n. Wówczas

$$P(S = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

i mówimy, że $S \sim \text{Bin}(n, p)$ (zmienna S ma rozkład dwumianowy z parametrami n, p).

1.16.3 Rozkład geometryczny

Rozważamy schemat Bernoulliego o nieskończonej długości. Jeśli T będzie zmienną losową o wartościach w $\mathbb{N} \cup \{0\}$, która opisuje liczbę prób Bernoulliego z parametrem p do momentu uzyskania pierwszego sukcesu to

$$P(T = k) = (1-p)^{k-1} p$$

i mówimy, że $T \sim \text{Geo}(p)$ (zmienna T ma rozkład geometryczny z parametrem p).

1.16.4 Rozkład Poissona

Załóżmy, iż mamy dany skończony przedział czasowy $[0;\tau]$ dla pewnego τ. Niech zmienna losowa N o wartościach w $\mathbb{N} \cup \{0\}$ opisuje liczbę wystąpień pewnego zdarzenia w tym przedziale, przy czym

- zdarzenia występują niezależnie od siebie
- intensywność wystąpień jest stała

to

$$P(N = k) = \frac{\mathrm{e}^{-\lambda}\lambda^k}{k!}$$

i mówimy, że $N \sim \mathrm{Pois}(\lambda)$ (zmienna N ma rozkład Poissona z parametrem λ). Zachodzi ponadto **twierdzenie Poissona**: niech (S_n) będzie ciągiem takim, że $S_n \sim \mathrm{Bin}(n, p_n)$, gdzie ciąg (p_n) jest taki, iż $\lim_{n\to\infty} np_n = \lambda$, wówczas

$$\lim_{n\to\infty} P(S_n = k) = \frac{\mathrm{e}^{\lambda}\lambda^k}{k!}\,.$$

1.16.5 Rozkład jednostajny

Jeśli zmienna losowa X o wartościach rzeczywistych ma gęstość prawdopodobieństwa daną przez

$$p(x) = \begin{cases} \frac{1}{b-a}\,, & x \in [a;b] \\ 0\,, & x \notin [a;b] \end{cases}$$

to mówimy $X \sim \mathcal{U}(a,b)$ (zmienna X ma rozkład jednostajny na odcinku $[a;b]$).

1.16.6 Rozkład wykładniczy

Jeśli zmienna losowa T o wartościach rzeczywistych opisuje prawdopodobieństwo uzyskania pierwszego zdarzenia po czasie x modelowanego przez rozkład $\mathrm{Pois}(\lambda)$ to

$$p(x) = \begin{cases} \lambda\mathrm{e}^{-\lambda x}\,, & x \geq 0 \\ 0\,, & x < 0 \end{cases}$$

i mówimy $T \sim \mathrm{Exp}(\lambda)$ (zmienna T ma rozkład wykładniczy z parametrem λ).

1.16.7 Rozkład gamma

Mówimy, że zmienna X o wartościach rzeczywistych ma rozkład gamma z parametrami $p, a > 0$ tj. $X \sim \Gamma(p,a)$ iff gęstość prawdopodobieństwa ma postać

$$p(x) = \begin{cases} \frac{a^p}{\Gamma(p)}x^{p-1}\mathrm{e}^{-ax}\,, & x > 0 \\ 0\,, & x \leq 0 \end{cases}\,,$$

gdzie Γ to funkcja gamma Eulera. Parametr p nazywamy parametrem kształtu, a a – parametrem intensywności. Szczególnym przypadkiem rozkładu gamma jest rozkład χ^2 zdefiniowany jako rozkład $\chi^2(n) := \Gamma(n/2, 1/2)$.

1.17 Elementarz teorii informacji

1.17.1 Definicja i własności entropii

Mając dany skończony zbiór zdarzeń elementarnych $\{A_1, \ldots, A_n\}$ taki, że wynikiem eksperymentu losowego może być dokładnie jedno z nich oraz prawdopodobieństwa $p_1, \ldots, p_n$, $\sum_i p_i = 1$ każdego z nich powiemy, iż

$$A := \begin{pmatrix} A_1 & A_2 & \cdots & A_n \\ p_1 & p_2 & \cdots & p_n \end{pmatrix}$$

jest *schematem skończonym* (z ang. *finite scheme*). Przykładowo rzut sprawiedliwą, sześcienną kostką do gry jest opisany przez schemat

$$\begin{pmatrix} A_1 & A_2 & A_3 & A_4 & A_5 & A_6 \\ 1/6 & 1/6 & 1/6 & 1/6 & 1/6 & 1/6 \end{pmatrix}.$$

Zauważmy, że każdy schemat skończony opisuje pewną *niepewność* dotyczącą doświadczenia losowego. Przykładowo jest oczywiste, iż dla schematów

$$\begin{pmatrix} A_1 & A_2 \\ 0.99 & 0.01 \end{pmatrix}, \quad \begin{pmatrix} A_1 & A_2 \\ 0.5 & 0.5 \end{pmatrix}$$

pierwszy z nich opisuje znacznie mniejszą niepewność od drugiego, gdyż prawie z pewnością wynikiem eksperymentu losowego będzie A_1. Wprowadzimy teraz wielkość, która w sensowny sposób mierzy ilość niepewności w danym schemacie skończonym. Wielkością taką jest *entropia Shannona* zdefiniowana dla schematu

$$A = \begin{pmatrix} A_1 & A_2 & \cdots & A_n \\ p_1 & p_2 & \cdots & p_n \end{pmatrix}$$

jako

$$H(A) = H(p_1, p_2, \ldots, p_n) := -\sum_{i=1}^{n} p_i \lg p_i \,,$$

gdzie możemy wybrać dowolną ustaloną podstawę logarytmu oraz stwierdzamy, iż jeśli $p_k = 0$ to $p_k \lg p_k = 0$. Jeśli jako podstawę wybierzemy liczbę 2 to entropię mierzymy w *bitach* tj. 1 bit jest to ilość niepewności zawarta w schemacie skończonym o dwóch jednakowo prawdopodobnych wynikach

$$H = -\log_2 \frac{1}{2} = 1 \,.$$

Przekonamy się teraz, iż tak zdefiniowana miara niepewności ma szereg własności, których spodziewalibyśmy się dla sensownej miary niepewności. Zauważmy wpierw, iż $H(p_1, \ldots, p_n) = 0$ iff dokładnie jedno zdarzenie $A_k \in A$ jest pewne, a pozostałe niemożliwe. Zauważmy dodatkowo, iż z nierówności Jensena mamy dla funkcji wypukłej $\phi(x) = x \lg x$

$$\phi\left(\sum_{i=1}^{n} \lambda_i x_i\right) \leq \sum_{i=1}^{n} \lambda_i \phi(x_i)\,,$$

dla dowolnych $x_1, \ldots, x_n \in \mathbb{R}$ i $\lambda_1, \ldots, \lambda_n \in [0;1]$, $\sum_i \lambda_i = 1$, skąd

$$\frac{1}{n} \lg \frac{1}{n} \leq \frac{1}{n} \sum_{i=1}^{n} p_i \lg p_i = -\frac{1}{n} H(p_1, \ldots, p_n)\,,$$

czyli

$$H(p_1, \ldots, p_n) \leq -\lg \frac{1}{n} = H(1/n, 1/n, \ldots, 1/n)\,,$$

czyli niepewność zawarta w danym schemacie skończonym jest mniejsza lub równa od niepewności zawartej w analogicznym schemacie, w którym wszystkie wyniki są jednakowo prawdopodobne.

Załóżmy teraz, że mamy dwa niezależne schematy skończone

$$A = \begin{pmatrix} A_1 & A_2 & \cdots & A_n \\ p_1 & p_2 & \cdots & p_n \end{pmatrix}, \quad B = \begin{pmatrix} B_1 & B_2 & \cdots & B_m \\ q_1 & q_2 & \cdots & q_m \end{pmatrix}$$

takie, że dla każdej pary zdarzeń A_i, B_j prawdopodobieństwo wystąpienia zdarzenia $A_i B_j$ wynosi $p_i q_j$. Zbiór zdarzeń $A_i B_j$ z prawdopodobieństwami $r_{ij} = p_i q_j$ reprezentuje nowy schemat skończony AB. Wówczas

$$\begin{aligned} -H(AB) &= \sum_{i=1}^{n} \sum_{j=1}^{m} r_{ij} \lg r_{ij} = \sum_{i=1}^{n} \sum_{j=1}^{m} p_i q_j \left(\lg p_i + \lg q_j\right) \\ &= \sum_{i=1}^{n} p_i \lg p_i + \sum_{j=1}^{m} q_j \lg q_j = -H(A) - H(B)\,, \end{aligned}$$

skąd

$$H(AB) = H(A) + H(B)\,.$$

Rozważmy teraz przypadek gdy schematy A, B są zależne. Przez q_{ij} oznaczmy prawdopodobieństwo zajścia zdarzenia B_j pod warunkiem zdarzenia A_i tj. $q_{ij} = p(B_j \mid A_i)$. Schemat AB jest teraz opisany prawdopodobieństwami $r_{ij} = p_i q_{ij}$ zatem

$$-H(AB) = \sum_{i=1}^{n} \sum_{j=1}^{m} p_i q_{ij} \left(\lg p_i + \lg q_{ij}\right) = -H(A) + \sum_{i=1}^{n} p_i \sum_{j=1}^{m} q_{ij} \lg q_{ij}$$

gdyż $\sum_j q_{ij} = 1$ (prawdopodobieństwo zajścia dowolnego zdarzenia z B pod warunkiem wystąpienia zdarzenia A_i wynosi 1), natomiast wielkość $-\sum_{j=1}^{m} q_{ij} \lg q_{ij}$ jest warunkową entropią schematu B pod warunkiem zajścia zdarzenia A_i, co oznaczymy jako $H(B \mid A = A_i)$

$$H(AB) = H(A) + \sum_{i=1}^{n} p_i H(B \mid A = A_i)\,.$$

Ostatni człon jest w takim razie wartością oczekiwaną wielkości $H(B)$ w schemacie A, co oznaczymy jako $H(B \mid A)$. Mamy w takim razie

$$H(AB) = H(A) + H(B \mid A)\,.$$

Z nierówności Jensena można dodatkowo pokazać, że zachodzi $H(B \mid A) \leq H(B)$.

1.17.2 Entropia względna

Dla dwóch ciągłych rozkładów prawdopodobieństwa $p(\mathbf{x})$, $q(\mathbf{x})$ definiujemy ich entropię względną (nazywaną również *Kullback-Leibler (KL) divergence*) jako

$$\mathbb{D}_{\mathrm{KL}}(p, q) = \int\limits_{\mathbb{R}^n} p(\mathbf{x}) \log \frac{p(\mathbf{x})}{q(\mathbf{x})}\, \mathrm{d}^n\mathbf{x}\ ,$$

która określa podobieństwo między dwoma rozkładami prawdopodobieństwa tj. dla ustalonego rozkładu p dla wszystkich q zachodzi $\mathbb{D}_{\mathrm{KL}}(p, q) \geq 0$, przy czym równość zachodzi iff $p = q$ (ponownie nierówność Jensena).

Rozważmy teraz rozkład łączny $p(\mathbf{x}, \mathbf{y})$. Jeśli zmienne losowe $\mathbf{x}, \mathbf{y}$ są niezależne to $p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$. Jeśli zmienne nie są niezależne to możemy określić stopień ich zależności właśnie poprzez entropię względną między rozkładem łącznym $p(\mathbf{x}, \mathbf{y})$, a rozkładem faktoryzowanym $p(\mathbf{x})p(\mathbf{y})$. Wielkość taką nazywamy informacją wzajemną (z ang/ *mutual information*)

$$\mathbb{I}(\mathbf{x}, \mathbf{y}) = \mathbb{D}_{\mathrm{KL}}(p(\mathbf{x}, \mathbf{y}), p(\mathbf{x})p(\mathbf{y})) = \int\limits_{\mathbb{R}^n \times \mathbb{R}^m} p(\mathbf{x}, \mathbf{y}) \log \left\{ \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})p(\mathbf{y})} \right\} \mathrm{d}^n\mathbf{x}\, \mathrm{d}^m\mathbf{y}\ .$$

1.18 Wnioskowanie statystyczne

Modelem statystycznym nazwiemy parę $(\chi, \mathcal{P})$, gdzie $\mathcal{P}$ jest rodziną rozkładów prawdopodobieństwa w zbiorze χ, przy czym będziemy zakładać $\chi = \mathbb{R}^n$

$$\mathcal{P} := \{p(\mathbf{x} \mid \theta) \mid \theta \in \Theta\}\ ,$$

gdzie Θ jest zbiorem parametrów modelu $\mathcal{P}$. Prostą próbą losową w modelu $\mathcal{P}$ nazwiemy ciąg niezależnych zmiennych losowych $X_1, \ldots, X_n$ o wartościach w $\mathbb{R}^n$ i pochodzących z tego samego rozkładu $p(\mathbf{x} \mid \theta) \in \mathcal{P}$ (w angielskiej terminologii taki ciąg zmiennych losowych nazwiemy *i.i.d.* tj. *independent and identically distributed*). Statystyką z kolei nazwiemy zmienną losową T będącą funkcją prostej próby losowej tj. $T = T(X_1, \ldots, X_n)$. Być może najważniejszym przykładem statystyki jest średnia oznaczana jako $\overline{X}$

$$\overline{X}(X_1, \ldots, X_n) := \frac{X_1 + \ldots + X_n}{n}\,.$$

Wartość oczekiwana statystyki średniej $\overline{X}(X_1, \ldots, X_n)$ dla X_i z rozkładu $X \sim \mathcal{D}$ o gęstości p wynosi

$$\mathbb{E}[\overline{X}] = \int \cdots \int \frac{1}{n} \left(\sum_{i=1}^{n} X_i \right) p(X_1) \cdots p(X_n)\, \mathrm{d}X_1 \ldots \mathrm{d}X_n = \mathbb{E}[X]\,.$$

Wariancja statystyki średniej wynosi z kolei

$$\begin{aligned}
&\mathrm{Var}[\overline{X}] = \mathbb{E}[\overline{X}^2] - \mathbb{E}[\overline{X}]^2 \\
&= \int \cdots \int \frac{1}{n^2} \left(\sum_{i=1}^{n} X_i^2 + \underbrace{\sum_{i \neq j} X_i X_j}_{n(n-1)} \right) p(X_1) \cdots p(X_n)\, \mathrm{d}X_1 \ldots \mathrm{d}X_n - \mathbb{E}[X]^2 \\
&= \frac{1}{n}\mathbb{E}[X^2] + \frac{n(n-1)}{n^2}\mathbb{E}[X]^2 - \mathbb{E}[X]^2 = \frac{1}{n}\left[\mathbb{E}[X^2] - \mathbb{E}[X]^2\right] = \frac{1}{n}\mathrm{Var}[X]\,.
\end{aligned}$$

1.19 Twierdzenie Gliwenki–Cantelliego

Dystrybuantą empiryczną nazywa się funkcję

$$\hat{F}(x) = \frac{1}{N} \# \left\{ i \in \{1, \ldots, N\} \mid x_i \leq x \right\}\,,$$

gdzie $\{x_1, \ldots, x_N\}$ jest realizacją prostej próby losowej. Twierdzenie Gliwenki–Cantelliego stwierdza, iż jeśli $F(x)$ jest dystrybuantą pewnego rozkładu prawdopodobieństwa to

$$\sup_{x \in \mathbb{R}} |\hat{F}_n(x) - F(x)| \to 0\,, \text{przy } n \to \infty.$$

1.20 Silne prawo wielkich liczb

Niech (X_n) będzie ciągiem zmiennych losowych i.i.d. z pewnego rozkładu $X \sim \mathcal{D}$. Przez $(\overline{X}_n)$ oznaczmy ciąg średnich częściowych tj.

$$\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i \,.$$

Wówczas zachodzi silne prawo wielkich liczb

$$P\left(\lim_{n \to \infty} \overline{X}_n = \mathbb{E}[X] \right) = 1 \,,$$

czyli średnia próbek zbiega do wartości oczekiwanej z prawdopodobieństwem 1.

Silne prawo wielkich liczb daje nam potężne narzędzie do szacowania wartości oczekiwanych, gdyż możemy je przybliżać średnią z dużej liczby próbek losowych, a dokładność tego przybliżenia zależy jedynie od liczby próbek i wariancji X. Jeśli X jest zmienną wielowymiarową to dokładność przybliżenia nie zależy wprost od liczby wymiarów i unikamy tzw. *curse of dimensionality*.

1.21 Centralne Twierdzenie Graniczne

Niech (X_n) będzie ciągiem k–wymiarowych zmiennych losowych i.i.d. z dowolnego rozkładu $X \sim \mathcal{D}$ o wartości oczekiwanej $\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}]$ i odwracalnej macierzy kowariancji $\boldsymbol{\Sigma}$. Oznaczając przez $(\overline{X}_n)$ ciąg średnich częściowych ciągu (X_n) zachodzi

$$\sqrt{n}\left(\overline{X}_n - \boldsymbol{\mu}\right) \to Z \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}) \,.$$

Oznacza to, iż dla ciągu $X_1, \ldots, X_n$ zmiennych losowych i.i.d. z praktycznie dowolnego rozkładu $X \sim \mathcal{D}$ dla odpowiednio dużych n średnią z próbek możemy traktować jako zmienną losową o rozkładzie normalnym $\mathcal{N}(\boldsymbol{\mu}, n^{-1/2}\boldsymbol{\Sigma})$.

1.22 Estymatory punktowe MLE i MAP

Rozważamy model statystyczny $\mathcal{P} = \{p(\mathbf{x} \mid \theta) \mid \theta \in \Theta\}$. Estymatorem parametru θ nazwiemy statystykę $\hat{\theta}(X_1, \ldots, X_n)$ służącą do oszacowania wartości tego parametru. Wartość tej statystki dla konkretnej realizacji prostej próby losowej $\hat{\theta}(\mathbf{x}_1, \ldots, \mathbf{x}_n)$ nazwiemy estymatą parametru θ. Dodatkowo definiujemy obciążenie (z ang. *bias*) estymatora jako wielkość

$$\mathbb{B}[\hat{\theta}] := \mathbb{E}[\hat{\theta}] - \theta \,.$$

Zasadniczo będą nas interesować dwa rodzaje estymat: MLE i MAP. W przypadku estymaty MLE (z ang. *Maximum Likelihood Estimate*) definiujemy funkcję

wiarygodności (*likelihood*) dla modelu $\mathcal{P} = \{p(\mathbf{x} \mid \theta) \mid \theta \in \Theta\}$ i realizacji prostej próby losowej (którą nazwiemy również danymi lub obserwacjami) $D = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$ jako

$$p(D \mid \theta) = \prod_{i=1}^{n} p(\mathbf{x}_i \mid \theta) \,.$$

Estymatą MLE nazywamy taką wartość parametru $\theta_{\mathrm{MLE}} \in \Theta$, że

$$p(D \mid \theta_{\mathrm{MLE}}) = \max_{\theta \in \Theta} p(D \mid \theta) \,.$$

Ponieważ znajdywanie maksimum funkcji będącej iloczynem nie jest zadaniem przyjemnym (chociażby obliczanie pochodnych iloczynu funkcji jest trudniejsze od sumy), więc wprowadzamy zanegowaną logarytmiczną funkcję wiarygodności

$$\ell(D \mid \theta) = -\log p(D \mid \theta) = -\sum_{i=1}^{n} \log p(\mathbf{x}_i \mid \theta) \,,$$

wówczas ze względu na fakt, iż funkcja $\log x$ jest ściśle rosnąca estymatę MLE możemy równoważnie wyznaczyć jako

$$\ell(D \mid \theta_{\mathrm{MLE}}) = \min_{\theta \in \Theta} \ell(D \mid \theta) \,.$$

Funkcję ℓ będziemy również nazywać funkcją kosztu.

W przypadku estymaty MAP (z ang. *Maximum a posteriori estimate*) wprowadzamy gęstość rozkładu a posteriori jako

$$p(\theta \mid D) = \frac{1}{Z} p(D \mid \theta) \pi(\theta) \,,$$

gdzie Z jest stałą wynikającą z warunku unormowania, a $\pi(\theta)$ to gęstość prawdopodobieństwa opisująca rozkład a priori parametru θ. Estymatą MAP nazywamy taką wartość parametru $\theta_{\mathrm{MAP}} \in \Theta$, że

$$p(\theta_{\mathrm{MAP}} \mid D) = \max_{\theta \in \Theta} p(\theta \mid D) \,.$$

Zauważmy przy tym iż liczba Z nie jest nam potrzebna, gdyż wystarczy zmaksymalizować licznik tj.

$$\theta_{\mathrm{MAP}} = \arg\max_{\theta \in \Theta} p(D \mid \theta) \pi(\theta) \,.$$

2 Probabilistyczne uczenie maszynowe

2.1 Wnioskowanie Bayesowskie

Zajmiemy się teraz wnioskowaniem opartym na twierdzeniu Bayesa. Rozpatrujemy model statystyczny $\mathcal{P} = \{p(\mathbf{x} \mid \theta) \mid \theta \in \Theta\}$. Załóżmy, iż mamy obserwacje $D = (\mathbf{x}_1, \ldots, \mathbf{x}_n)$, wówczas twierdzenie Bayesa możemy zapisać jako

$$p(\theta \mid D) = \frac{p(D \mid \theta)\pi(\theta)}{p_D(D)} = \frac{p(D \mid \theta)\pi(\theta)}{\int\limits_{\Theta} p(D \mid \theta)\pi(\theta)\,\mathrm{d}\theta}\,,$$

gdzie $p(\theta \mid D)$ nazywamy rozkładem a posteriori (posteriorem), $p(D \mid \theta)$ – wiarygodnością (likelihood), a $\pi(\theta)$ – rozkładem a priori (priorem).

Całe wnioskowanie Bayesowskie opiera się na wyznaczeniu rozkładu a posteriori, który wyraża całą naszą wiedzę o estymowanym parametrze θ. Na podstawie tego rozkładu możemy wyznaczyć estymatę punktową MAP maksymalizującą gęstość prawdopodobieństwa a posteriori, jak również niepewność związaną z wyznaczeniem tej estymaty np. poprzez wyznaczenie przedziału wiarygodności $C_{1-\alpha}(\theta \mid D) = [\theta_l; \theta_u]$ takiego, że

$$P(\theta \in [\theta_l; \theta_u] \mid D) = 1 - \alpha\,,$$

dla ustalonego $0 < \alpha < 1$. Możemy również skonstruować rozkład predykcyjny (z ang. *posterior predictive distribution*) określający prawdopodobieństwo zaobserwowania nowej obserwacji $\mathbf{x}$

$$p(\mathbf{x} \mid D) = \int\limits_{\Theta} p(\mathbf{x} \mid \theta)p(\theta \mid D)\,\mathrm{d}\theta\ .$$

Znając rozkład a posteriori estymowanego parametru θ możemy nie tylko wyznaczyć estymaty punktowe, wartości oczekiwane i przedziały wiarygodności, ale również znaleźć estymator Bayesa (z ang. *Bayes estimator*), który minimalizuje wartość oczekiwaną pewnej funkcji kosztu (z ang. *loss/cost function*) $L(\theta, \hat{\theta})$ po wszystkich estymatorach $\hat{\theta}$

$$\theta_{\text{Bayes}} = \arg\min_{\hat{\theta}} \int\limits_{\Theta} L(\theta, \hat{\theta})p(\theta \mid D)\,\mathrm{d}\theta\ .$$

Całkę w powyższym wzorze nazywa się również funkcją ryzyka (z ang. *risk function*) $R(\hat{\theta})$, która określa oczekiwaną stratę spowodowaną wykorzystaniem danego estymatora parametru θ. W przypadku gdy funkcja kosztu ma postać błędu kwadratowego (L2)

$$L(\theta, \hat{\theta}) = (\theta - \hat{\theta})^2$$

funkcję ryzyka możemy zapisać jako

$$\begin{aligned} R(\hat{\theta}) &= \int_{\Theta} \theta^2 p(\theta \mid D)\,\mathrm{d}\theta - 2\hat{\theta} \int_{\Theta} \theta p(\theta \mid D)\,\mathrm{d}\theta + \hat{\theta}^2 \\ &= \mathrm{Var}[\theta \mid D] + \mathbb{E}[\theta \mid D]^2 - 2\hat{\theta}\mathbb{E}[\theta \mid D] + \hat{\theta}^2 \\ &= \mathrm{Var}[\theta \mid D] + \left(\mathbb{E}[\theta \mid D] - \hat{\theta}\right)^2 . \end{aligned}$$

2.2 Bayesowski wybór modeli

Załóżmy, iż mamy rodzinę $\mathcal{M}$ modeli statystycznych (może to być zbiór dyskretny lub zbiór modeli indeksowanych ciągłym, wielowymiarowym parametrem $\boldsymbol{\lambda}$). Naszym zadaniem jest wybór najbardziej prawdopodobnego modelu dla danych D. Możemy na to zadanie patrzeć jako zadanie z teorii decyzji: dla danej funkcji kosztu $L(M, M^*)$ i rozkładu a posteriori nad modelami $p(M \mid D)$ chcemy wybrać model, który minimalizuje ryzyko $\mathbb{E}[L(M, M^*)]$. Jeśli jako koszt wybierzemy tzw. *0–1 loss* tj.

$$L(M, M^*) = \begin{cases} 0 & \text{, jeśli } M = M^* \\ 1 & \text{, w.p.p.} \end{cases}$$

to

$$\mathbb{E}[L(M, M^*)] = 1 - p(M^* \mid D)$$

i wybieramy model M o największym prawdopodobieństwie (estymata MAP). Pozostaje tylko wyznaczenie $p(M \mid D)$

$$p(M \mid D) = \frac{p(D \mid M)\pi(M)}{\sum_{M \in \mathcal{M}} p(D \mid M)\pi(M)} .$$

Jeśli jako prior przyjmiemy rozkład jednostajny $\pi(M) = |\mathcal{M}|^{-1}$ to estymata MAP sprowadza się do MLE czyli szukamy modelu

$$M^* = \arg\max_{M \in \mathcal{M}} p(D \mid M) .$$

Jeśli przez θ_M oznaczymy parametry modelu M to

$$p(D \mid M) = \int_{\Theta_M} p(D \mid \theta_M)\pi(\theta_M)\,\mathrm{d}\theta_M \ .$$

Powyższą wielkość nazywamy wiarygodnością brzegową (z ang. *marginal likelihood*) lub *model evidence*.

2.3 Estymator jądrowy gęstości (KDE)

Załóżmy, że mamy zbiór obserwacji iid $\{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ taki, że $\mathbf{x}_i \sim \mathcal{D}$ dla pewnego n–wymiarowego ciągłego rozkładu prawdopodobieństwa $\mathcal{D}$ z nieznaną gęstością prawdopodobieństwa $p(\mathbf{x})$. Chcemy znaleźć estymator $\hat{p}(\mathbf{x})$ tej funkcji. Estymatorem jądrowym gęstości funkcji p (z ang. *kernel density estimator*) nazywamy funkcję

$$\hat{p}(\mathbf{x}) := \frac{1}{mh^n} \sum_{i=1}^{m} K\left(\frac{\mathbf{x} - \mathbf{x}_i}{h}\right),$$

gdzie $h \in \mathbb{R}$ jest pewnym hiperparametrem zwanym *bandwidth*, a $K : \mathbb{R}^n \mapsto [0; \infty)$ to tzw. funkcja jądrowa będąca parzystą funkcją posiadającą w 0 maksimum globalne oraz spełniającą warunek unormowania

$$\int_{\mathbb{R}^n} K(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} = 1 \,.$$

Ze statystycznego punktu widzenia, postać jądra nie ma istotnego znaczenia i wybór funkcji K może być arbitralny, uwzględniający przede wszystkim pożądane własności otrzymanego estymatora, na przykład klasę jego regularności (ciągłość, różniczkowalność itp.). W przypadku jednowymiarowym jako funkcję K przyjmuje się klasyczne postacie gęstości rozkładów probabilistycznych, na przykład gęstość rozkładu normalnego. W przypadku wielowymiarowym stosuje się tzw. jądro radialne tj. dla jądra jednowymiarowego K wielowymiarowe jądro radialne definiujemy jako

$$K(\mathbf{x}) = K(\|\mathbf{x}\|)$$

dla pewnej normy (typowo normy euklidesowej) $\|\cdot\|$.

2.4 Modele Gaussowskie

Jak już wspomnieliśmy w przypadku gdy zmienna losowa ma wielowymiarowy rozkład normalny $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ wszystkie rozkłady brzegowe i warunkowe są również rozkładami normalnymi. W szczególnym przypadku gdy zmienne k i $n-k$ –wymiarowe $\mathbf{x}$ i $\mathbf{y}$ mają łącznie rozkład normalny

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{y} \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \,,$$

gdzie

$$\boldsymbol{\mu} = \begin{bmatrix} \boldsymbol{\mu}_\mathbf{x} \\ \boldsymbol{\mu}_\mathbf{y} \end{bmatrix}, \quad \boldsymbol{\Sigma} = \begin{bmatrix} \boldsymbol{\Sigma}_{\mathbf{xx}} & \boldsymbol{\Sigma}_{\mathbf{xy}} \\ \boldsymbol{\Sigma}_{\mathbf{yx}} & \boldsymbol{\Sigma}_{\mathbf{yy}} \end{bmatrix}$$

można pokazać iż

$$\mathbf{x} \mid \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}}), \quad \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{yy}}),$$

gdzie

$$\begin{aligned} \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} &= \boldsymbol{\mu}_{\mathbf{x}} + \boldsymbol{\Sigma}_{\mathbf{xy}}\boldsymbol{\Sigma}_{\mathbf{yy}}^{-1}(\mathbf{y} - \boldsymbol{\mu}_{\mathbf{y}}) \\ \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}} &= \boldsymbol{\Sigma}_{\mathbf{xx}} - \boldsymbol{\Sigma}_{\mathbf{xy}}\boldsymbol{\Sigma}_{\mathbf{yy}}^{-1}\boldsymbol{\Sigma}_{\mathbf{yx}} \end{aligned} .$$

2.5 Liniowe modele Gaussowskie

Powyższe własności rozkładów łącznych pozwalają jawnie wnioskować w tzw. liniowych modelach Gaussowskich (z ang. *Linear Gaussian Models*). Załóżmy, iż nasze obserwacje są modelowane przez n–wymiarową zmienną losową $\mathbf{y}$ o rozkładzie normalnym z estymowanym parametrem $\mathbf{x}$ i znanymi parametrami $\mathbf{A}, \mathbf{b}, \boldsymbol{\Sigma}_{\mathbf{y}}$ tak, że wiarygodność ma postać

$$\mathbf{y} \mid \mathbf{x} \sim \mathcal{N}(\mathbf{A}\mathbf{x} + \mathbf{b}, \boldsymbol{\Sigma}_{\mathbf{y}}),$$

gdzie $\mathbf{A}$ jest macierzą wymiaru $n \times k$. Jako prior na parametr $\mathbf{x}$ przyjmiemy również rozkład normalny o pewnych zadanych parametrach $\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}}$ (taki wybór rozkładu a priori nazywamy rozkładem sprzężonym do wiarygodności)

$$\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}}, \boldsymbol{\Sigma}_{\mathbf{x}}).$$

Wówczas łatwo pokazać, iż rozkład a posteriori jest rozkładem normalnym

$$\mathbf{x} \mid \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}})$$

z parametrami

$$\begin{aligned} \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}} &= \left[\boldsymbol{\Sigma}_{\mathbf{x}}^{-1} + \mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}\mathbf{A}\right]^{-1} \\ \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} &= \boldsymbol{\Sigma}_{\mathbf{x}|\mathbf{y}}\left[\mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{y} - \mathbf{b}) + \boldsymbol{\Sigma}_{\mathbf{x}}^{-1}\boldsymbol{\mu}_{\mathbf{x}}\right] \end{aligned} .$$

Załóżmy teraz, iż mamy ciąg obserwacji $(\mathbf{y}_1, \ldots, \mathbf{y}_m)$. Wnioskowanie Bayesowskie możemy wówczas stosować iteracyjnie tzn. na początku dla 0 obserwacji rozkład estymowanego parametru jest opisany przez prior $\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$. Po zaobserwowaniu jednego $\mathbf{y}_1$ aktualizujemy nasze przekonania co do parametru $\mathbf{x}$ zgodnie z powyższym wzorem i otrzymujemy rozkład normalny o parametrach

$$\begin{aligned} \boldsymbol{\Sigma}_1 &= \left[\boldsymbol{\Sigma}_0^{-1} + \mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}\mathbf{A}\right]^{-1} \\ \boldsymbol{\mu}_1 &= \boldsymbol{\Sigma}_1\left[\mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{y}_1 - \mathbf{b}) + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right] \end{aligned} .$$

Po zaobserwowaniu kolejnego $\mathbf{y}_2$ ponownie wykorzystujemy powyższe wzory ale jako prior wykorzystując rozkład w poprzedniej iteracji. W ogólności możemy zapisać wzór rekurencyjny na $m+1$ rozkład jako

$$\begin{aligned}\boldsymbol{\Sigma}_{m+1} &= \left[\boldsymbol{\Sigma}_m^{-1} + \mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}\mathbf{A}\right]^{-1} \\ \boldsymbol{\mu}_{m+1} &= \boldsymbol{\Sigma}_{m+1}\left[\mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{y}_{m+1}-\mathbf{b}) + \boldsymbol{\Sigma}_m^{-1}\boldsymbol{\mu}_m\right]\end{aligned},$$

skąd możemy od razu podać wzór na parametry m–tego rozkładu

$$\begin{aligned}\boldsymbol{\Sigma}_m &= \left[\boldsymbol{\Sigma}_0^{-1} + m\mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}\mathbf{A}\right]^{-1} \\ \boldsymbol{\mu}_m &= \boldsymbol{\Sigma}_m\left[\mathbf{A}^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}\left(\sum_{i=1}^m \mathbf{y}_i - m\mathbf{b}\right) + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right]\end{aligned}.$$

Taki sam wynik można by uzyskać rozpatrując łączny rozkład a posteriori dla obserwacji $D = (\mathbf{y}_1, \ldots, \mathbf{y}_m)$ tj.

$$p(\mathbf{x} \mid D) \cong \pi(\mathbf{x})\prod_{i=1}^m p(\mathbf{y}_i \mid \mathbf{x}) \cong$$
$$\exp\left\{-\frac{1}{2}\left[(\mathbf{x}-\boldsymbol{\mu}_0)^T\boldsymbol{\Sigma}_0^{-1}(\mathbf{x}-\boldsymbol{\mu}_0) + \sum_{i=1}^m(\mathbf{y}_i - \mathbf{A}\mathbf{x} - \mathbf{b})^T\boldsymbol{\Sigma}_{\mathbf{y}}^{-1}(\mathbf{y}_i - \mathbf{A}\mathbf{x} - \mathbf{b})\right]\right\}.$$

2.6 Regresja liniowa

Załóżmy, iż modelujemy obserwacje postaci $(y, \mathbf{x})$ gdzie y to skalar zwany zmienną objaśnianą, którego wartość obserwujemy, a $\mathbf{x}$ to wektor zmiennych objaśniających, który kontrolujemy tj. zakładamy, iż wektor $\mathbf{x}$ dla danego pomiaru y znamy dokładnie. Dodatkowo zakładamy, iż y zależy liniowo od $\mathbf{x}$ tj.

$$y = \mathbf{w}^T\mathbf{x} + \epsilon\,,$$

gdzie $\epsilon \sim \mathcal{N}(0, \sigma^2)$ dla znanego σ jest tzw. błędem losowym, a $\mathbf{w}$ jest estymowanym przez nas parametrem. Możemy zatem zapisać

$$y \mid \mathbf{w} \sim \mathcal{N}(\mathbf{w}^T\mathbf{x}, \sigma^2)\,.$$

Powiedzmy, iż zaobserwowaliśmy ciąg obserwacji $D = (y_1, \ldots, y_m)$ dla zadanych (lub dokładnie znanych) przez nas $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Wiarygodność ma zatem postać

$$p(D \mid \mathbf{w}) \cong \prod_{i=1}^m \exp\left\{-\frac{1}{2\sigma^2}\left(y_i - \mathbf{w}^T\mathbf{x}_i\right)^2\right\}.$$

W przypadku regresji liniowej zamiast pełnego wnioskowania Bayesowskiego o parametrze $\mathbf{w}$ często stosuje się prostsze podejście polegające na znalezieniu estymaty punktowej MLE. Zanegowana logarytmiczna funkcja wiarygodności ma postać

$$\ell(D \mid \mathbf{w}) = \frac{1}{2\sigma^2} \sum_{i=1}^{m} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \text{const.}$$

Człon stały możemy oczywiście pominąć i zapisać

$$\ell(D \mid \mathbf{w}) \cong \sum_{i=1}^{m} (y_i - \mathbf{w}^T \mathbf{x}_i)^2 = (\mathbf{y} - \mathbf{X}\mathbf{w})^T (\mathbf{y} - \mathbf{X}\mathbf{w}) \ ,$$

gdzie

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}, \quad \mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_m^T \end{bmatrix}.$$

Ponieważ otrzymana funkcja ℓ ma postać formy kwadratowej, więc problem optymalizacyjny polegający na znalezieniu minimum ℓ nazywa się metodą najmniejszych kwadratów (z ang. *OLS – Ordinary Least Squares*). Aby wyznaczyć estymatę $\mathbf{w}_{\text{MLE}}$ musimy rozwiązać równanie

$$\frac{\partial \ell}{\partial \mathbf{w}} = \frac{\partial}{\partial \mathbf{w}} \left[\mathbf{y}^T \mathbf{y} + \mathbf{w}^T \mathbf{X}^T \mathbf{X} \mathbf{w} - 2\mathbf{y}^T \mathbf{X} \mathbf{w} \right] = \mathbf{0} \, ,$$

skąd

$$2\mathbf{X}^T \mathbf{X} \mathbf{w} - 2\mathbf{X}^T \mathbf{y} = \mathbf{0} \, ,$$

zatem

$$\mathbf{w}_{\text{MLE}} = (\mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y} = \mathbf{X}^+ \mathbf{y} \, ,$$

gdzie $\mathbf{X}^+$ oznacza tzw. *pseudoodwrotność Moore'a–Penrose'a*, którą można efektywnie obliczyć korzystając z rozkładu SVD macierzy $\mathbf{X}$.

Pełniejszą informację o parametrze $\mathbf{w}$ możemy uzyskać rozpatrując rozkład a posteriori $p(\mathbf{w} \mid D)$. Jeśli jako prior przyjmiemy rozkład normalny z pewnymi parametrami $\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0$ to zauważmy, iż otrzymujemy instancję liniowego modelu Gaussowskiego

$$\begin{aligned} \mathbf{y} \mid \mathbf{w} &\sim \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2 \mathbf{1}) \\ \mathbf{w} &\sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) \end{aligned} \ ,$$

skąd rozkład a posteriori jest rozkładem normalnym

$$\mathbf{w} \mid \mathbf{y} \sim \mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$$

o parametrach

$$\begin{aligned}\boldsymbol{\Sigma}_m &= \left[\boldsymbol{\Sigma}_0^{-1} + \sigma^{-2}\mathbf{X}^T\mathbf{X}\right]^{-1} \\ \boldsymbol{\mu}_m &= \boldsymbol{\Sigma}_m\left[\sigma^{-2}\mathbf{X}^T\mathbf{y} + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right]\end{aligned} .$$

W powyższych wzorach nazwy parametrów nie są przykładowe: po zaobserwowaniu 0 przykładów rozkład parametru $\mathbf{w}$ jest rozkładem a priori $\mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$; po zaobserwowaniu po jednej wartości y_i w m zadanych (znanych dokładnie) punktach $\mathbf{x}_i$ otrzymujemy rozkład a posteriori $\mathcal{N}(\boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m)$. Gdybyśmy w każdym z m punktów $\mathbf{x}_i$ dokonywali pomiaru y_i s–krotnie to wtedy wykorzystując wzory wyprowadzone przy iteracyjnym stosowaniu wnioskowania w liniowym modelu Gaussowskim otrzymujemy rozkład normalny o parametrach

$$\begin{aligned}\boldsymbol{\Sigma}_{m;s} &= \left[\boldsymbol{\Sigma}_0^{-1} + \frac{s}{\sigma^2}\mathbf{X}^T\mathbf{X}\right]^{-1} \\ \boldsymbol{\mu}_{m;s} &= \boldsymbol{\Sigma}_{m;s}\left[\sigma^{-2}\mathbf{X}^T\sum_{i=1}^{s}\mathbf{y}_i + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right]\end{aligned} .$$

Rozkład predykcyjny dla nowej obserwacji y poczynionej w punkcie $\mathbf{x}$ jest dany przez

$$p(y \mid \mathbf{y}) = \int\limits_{\mathbb{R}^n} p(y \mid \mathbf{w})p(\mathbf{w} \mid \mathbf{y})\mathrm{d}^n\mathbf{w}\,.$$

Nietrudno zauważyć, iż będzie to rozkład normalny o parametrach

$$\begin{aligned}\mu_{y|\mathbf{y}} = \mathbb{E}[y \mid \mathbf{y}] &= \int\limits_{\mathbb{R}} yp(y \mid \mathbf{y})\,\mathrm{d}y = \int\limits_{\mathbb{R}^n}\mathrm{d}^n\mathbf{w}\,p(\mathbf{w} \mid \mathbf{y})\int\limits_{\mathbb{R}}\mathrm{d}y\,yp(y \mid \mathbf{w}) \\ &= \int\limits_{\mathbb{R}^n}\mathrm{d}^n\mathbf{w}\,p(\mathbf{w} \mid \mathbf{y})\mathbf{x}^T\mathbf{w} = \mathbf{x}^T\boldsymbol{\mu}_m\,.\end{aligned}$$

oraz

$$\begin{aligned}\sigma^2_{y|\mathbf{y}} &= \mathbb{E}\left[(y-\mu_{y|\mathbf{y}})^2 \mid \mathbf{y}\right] = \int\limits_{\mathbb{R}^n}\mathrm{d}^n\mathbf{w}\,p(\mathbf{w} \mid \mathbf{y})\int\limits_{\mathbb{R}}\mathrm{d}y\,(y-\mu_{y|\mathbf{y}})^2p(y \mid \mathbf{w}) \\ &= \int\limits_{\mathbb{R}^n}\mathrm{d}^n\mathbf{w}\,p(\mathbf{w} \mid \mathbf{y})\int\limits_{\mathbb{R}}\mathrm{d}y\left(y^2+\mu^2_{y|\mathbf{y}}-2\mu_{y|\mathbf{y}}y\right)p(y \mid \mathbf{w}) \\ &= \int\limits_{\mathbb{R}^n}\mathrm{d}^n\mathbf{w}\,p(\mathbf{w} \mid \mathbf{y})\left(\sigma^2+(\mathbf{x}^T\mathbf{w})^2+\mu^2_{y|\mathbf{y}}-2\mu_{y|\mathbf{y}}\mathbf{x}^T\mathbf{w}\right) \\ &= \sigma^2 + \int\limits_{\mathbb{R}^n}\mathrm{d}^n\mathbf{w}\,p(\mathbf{w} \mid \mathbf{y})\left(\mathbf{x}^T\mathbf{w}-\mathbf{x}^T\boldsymbol{\mu}_m\right)^2 \\ &= \sigma^2 + \mathbf{x}^T\mathbb{E}[(\mathbf{w}-\boldsymbol{\mu}_m)(\mathbf{w}-\boldsymbol{\mu}_m)^T \mid \mathbf{y}]\mathbf{x} = \sigma^2 + \mathbf{x}^T\boldsymbol{\Sigma}_m\mathbf{x}\,.\end{aligned}$$

Powyżej skorzystaliśmy ze znanego faktu, iż dla jednowymiarowej zmiennej losowej zachodzi $\sigma^2 = \mathbb{E}[(X - \mu_X)^2] = \mathbb{E}[X^2] - \mu_X^2$, skąd $\mathbb{E}[X^2] = \sigma^2 + \mu_X^2$. Podsumowując rozkład predykcyjny ma postać

$$y \mid \mathbf{y} \sim \mathcal{N}(\mathbf{x}^T \boldsymbol{\mu}_m, \sigma^2 + \mathbf{x}^T \boldsymbol{\Sigma}_m \mathbf{x}) \,.$$

2.7 Regularyzacja

Regularyzacją nazywamy proces polegający na wprowadzeniu ad hoc do zagadnienia optymalizacji dodatkowych członów tak, aby rozwiązanie było regularne (prostsze, nieosobliwe, jednoznaczne ...). W przypadku funkcji kosztu ℓ najczęściej dodajemy człon penalizujący rozwiązania o dużej normie estymowanego parametru postaci

$$\gamma \|\theta\|$$

dla pewnej normy $\|\cdot\|$ i hiper-parametru γ określającego siłę regularyzacji. W kontekście Bayesowskim regularyzację można również rozumieć jako pewną niechęć ("tłumienie", zachowawczość) modelu do zmiany rozkładu a priori estymowanego parametru po pojawieniu się kolejnych obserwacji.

Przykładowo jeśli w zagadnieniu Bayesowskiej regresji liniowej jako prior przyjmiemy rozkład normalny

$$\mathbf{w} \sim \mathcal{N}(\mathbf{0}, \tau^2 \mathbf{1})$$

to rozkład a posteriori jest rozkładem normalnym o parametrach

$$\begin{aligned} \boldsymbol{\Sigma}_m &= \sigma^2 \left[\gamma \mathbf{1} + \mathbf{X}^T \mathbf{X}\right]^{-1} \\ \boldsymbol{\mu}_m &= \left[\gamma \mathbf{1} + \mathbf{X}^T \mathbf{X}\right]^{-1} \mathbf{X}^T \mathbf{y} \end{aligned} \quad ,$$

gdzie $\gamma = \sigma^2/\tau^2$ jest hiper-parametrem określającym siłę regularyzacji. Zauważmy, że im większa jest wartość γ (mniejsza niepewność związana z rozkładem a priori) tym drugi człon w nawiasie staje się mniej istotny. Taki sam wynik możemy uzyskać metodą OLS jeśli do funkcji kosztu dodamy człon regularyzujący dla zwykłej normy euklidesowej (norma L^2) postaci $\gamma \sum_{i=1}^n w_i^2$. Zagadnienie minimalizacji funkcji kosztu będącej formą kwadratową z dodanym członem regularyzującym nazywamy również regresją grzbietową (z ang. *ridge regression*).

Innym przykładem regularyzacji jest przyjęcie jako priora ma parametry modelu rozkładu Laplace'a postaci

$$p(\mathbf{w}) \cong \prod_{i=1}^{n} \exp\left(-\frac{|w_i|}{\lambda}\right)$$

co powoduje dodanie do funkcji kosztu członu postaci $\gamma \sum_{i=1}^n |w_i|$ będącego normą L^1 wektora wag $\mathbf{w}$. Takie zagadnienie nazywamy regresją LASSO i w tym przypadku nie da się prosto analitycznie znaleźć estymaty punktowej MAP. Zwykle

do tego zagadnienia używamy algorytmu optymalizacji numerycznej CD (z ang. *coordinate descent*). Można również połączyć regularyzacje L^1 i L^2 tj. dodać do funkcji kosztu człon postaci $\gamma_1\|\mathbf{w}\|_1 + \gamma_2\|\mathbf{w}_2\|$ i takie zagadnienie nazywamy regresją ElasticNet i również znajdujemy estymatę MAP korzystając z CD.

2.8 Robust regression

Jednym z problemów przy modelowaniu zależności $y(\mathbf{x})$ przez rozkład normalny jest duża czułość takiego modelu na wartości odstające (tzw. *outliers*). Wynika to z tego, iż wynikający z takiego modelu błąd będący zanegowaną logarytmiczną funkcją wiarygodności jest proporcjonalny do kwadratu odległości punktu od prostej regresji. Tzw. metody *robust* to zbiór metod odpornych na outliery, które przypisują im mniejszą wagę niż standardowa regresja liniowa. Jednym z modeli robust regression jest tzw. regresja LAD (z ang. *Least Absolute Deviations*), która modeluje zależność poprzez rozkład Laplace'a, który posiada tzw. ciężkie ogony (z ang. *heavy tails*) i dzięki temu przypisuje większe prawdopodobieństwo wartościom odstającym niż rozkład normalny

$$y \mid \mathbf{w} \sim \text{Laplace}(\mathbf{w}^T\mathbf{x}, \lambda)\,,$$

gdzie λ jest parametrem skali. Funkcja gęstości prawdopodobieństwa dla rozkładu Laplace'a z parametrami μ, λ ma postać

$$p(y) = \frac{1}{2\lambda}\exp\left(-\frac{|y-\mu|}{\lambda}\right)\,,$$

co przekłada się na funkcję kosztu postaci

$$\ell(D \mid \mathbf{w}) \cong \sum_{i=1}^{m} |y_i - \mathbf{w}^T\mathbf{x}_i|$$

gdzie założono, iż parametr λ jest ustalony i stały. Bardziej ogólnie można modelować zależność $y(\mathbf{x})$ przez asymetryczny rozkład Laplace'a (z ang. *Asymmetric Laplace Distribution*)

$$y \mid \mathbf{w} \sim \text{ALD}(\mathbf{w}^T\mathbf{x}, \lambda, q)$$

o gęstości prawdopodobieństwa i dystrybuancie dla parametrów μ, λ, q, gdzie $0 < q < 1$

$$p(y) = \frac{q(1-q)}{\lambda}\begin{cases}\exp\left(-\frac{q-1}{\lambda}(y-\mu)\right) & ,y \le \mu \\ \exp\left(-\frac{q}{\lambda}(y-\mu)\right) & ,y > \mu\end{cases}$$

$$F(y) = \begin{cases}q\exp\left(\frac{1-q}{\lambda}(y-\mu)\right) & ,y \le \mu \\ 1-(1-q)\exp\left(-\frac{q}{\lambda}(y-\mu)\right) & ,y > \mu\end{cases}$$

Z powyższego widzimy, iż o ile estymata MLE parametrów $\mathbf{w}$ dla zwykłej regresji liniowej jest taka, iż dla dowolnego $\mathbf{x}$ wielkość $\mathbf{w}^T\mathbf{x}$ jest wartością oczekiwaną rozkłady warunkowego $y \mid \mathbf{w} \sim \mathcal{N}(\mathbf{w}^T\mathbf{x}, \sigma^2)$ to w przypadku asymetrycznego rozkładu Laplace'a estymata MLE zwraca $\mathbf{w}$ takie, że $\mathbf{w}^T\mathbf{x}$ jest kwantylem rzędu q założonego rozkładu ALD. W szczególności więc rozkład Laplace'a zwraca medianę. Powyższa postać gęstości prawdopodobieństwa prowadzi do funkcji kosztu postaci

$$\ell \cong \sum_{i=1}^{m} \left\{ (q-1)(y_i - \mathbf{w}^T\mathbf{x}_i)\theta(\mathbf{w}^T\mathbf{x}_i - y_i) + q(y_i - \mathbf{w}^T\mathbf{x}_i)\theta(y_i - \mathbf{w}^T\mathbf{x}_i) \right\} ,$$

gdzie θ oznacza funkcję Heaviside'a. Taki model nazywamy regresją kwantylową (z ang. *quantile regression*). Zarówno w przypadku regresji LAD oraz quantile do znalezienia estymat MLE korzystamy z programowania liniowego. Istotnie można łatwo sformułować problem minimalizacji nieograniczonej wyrażenia $\sum_{i=1}^{m} |x_i|$ jako problem programowania liniowego

$$\begin{array}{ll} \text{minimize} & \displaystyle\sum_{i=1}^{m} v_i \\ \text{subject to} & \forall i : v_i \geq +x_i \\ & \forall i : v_i \geq -x_i \end{array} .$$

W obu przypadkach do funkcji kosztu możemy dodać człon regularyzujący L^1 (członu L^2 dodać nie możemy bo wówczas nie można problemu minimalizacji przedstawić jako problemu programowania liniowego).

Ostatnim przykładem robust regression jaki podamy będzie tzw. regresja Hubera, w której heurystycznie wprowadzamy funkcję kosztu Hubera (tzw. *Huber loss*) postaci

$$\ell_{\text{Huber}}(y_i \mid \mathbf{w}) = \begin{cases} \frac{1}{2}(y_i - \mathbf{w}^T\mathbf{x}_i)^2 & \text{, jeśli } |y_i - \mathbf{w}^T\mathbf{x}_i| \leq \epsilon \\ -\frac{1}{2}\epsilon^2 + \epsilon|y_i - \mathbf{w}^T\mathbf{x}_i| & \text{, w.p.p.} \end{cases} .$$

Taka funkcja kosztu jest różniczkowalna oraz wypukła, co umożliwia jest minimalizację korzystając z różnych algorytmów optymalizacji numerycznej (np. L-BFGS).

2.9 Procesy Gaussowskie

Jak już wspomnieliśmy macierz kowariancji n–wymiarowej zmiennej losowej $\mathbf{x}$ o wartości oczekiwanej $\boldsymbol{\mu}$ jest zdefiniowana jako

$$\boldsymbol{\Sigma} = \mathbb{E}\left[(\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^T\right] .$$

Pokazaliśmy również, iż macierz ta jest nieujemnie określona. Dodatkowo pokażemy, iż dla każdej nieujemnie określonej macierzy symetrycznej $\mathbf{K}$ wymiaru $n \times n$ istnieje n–wymiarowa zmienna losowa o wielowymiarowym rozkładzie normalnym dla której $\mathbf{K}$ jest macierzą kowariancji. Istotnie dla każdej nieujemnie określonej macierzy symetrycznej istnieje macierz $\mathbf{L}$ taka, że

$$\mathbf{K} = \mathbf{L}\mathbf{L}^T,$$

jest to tzw. dekompozycja Choleskiego. Niech $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$, wówczas zmienna losowa $\mathbf{Lz}$ ma rozkład o zerowej wartości oczekiwanej i macierzy kowariancji

$$\mathbb{E}\left[(\mathbf{Lz})(\mathbf{Lz})^T\right] = \mathbb{E}\left[\mathbf{L}\mathbf{z}\mathbf{z}^T\mathbf{L}^T\right] = \mathbf{L}\mathbb{E}[\mathbf{z}\mathbf{z}^T]\mathbf{L}^T = \mathbf{L}\mathbf{1}\mathbf{L}^T = \mathbf{K}.$$

Powyższe własności wskazują, iż macierze kowariancji można w pewnym sensie utożsamiać z nieujemnie określonymi macierzami symetrycznymi.

Zdefiniujemy teraz funkcję kowariancji $k : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ taką, że $\forall m \in \mathbb{N}$: $\forall X = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \subset \mathbb{R}^n$ macierz

$$k(X, X) = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & k(\mathbf{x}_1, \mathbf{x}_2) & \cdots & k(\mathbf{x}_1, \mathbf{x}_m) \\ k(\mathbf{x}_2, \mathbf{x}_1) & k(\mathbf{x}_2, \mathbf{x}_2) & \cdots & k(\mathbf{x}_2, \mathbf{x}_m) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_m, \mathbf{x}_1) & k(\mathbf{x}_m, \mathbf{x}_2) & \cdots & k(\mathbf{x}_m, \mathbf{x}_m) \end{bmatrix}$$

jest dodatnio określoną macierzą symetryczną. Funkcję k nazywamy również jądrem dodatnio określonym (z ang. *positive definite kernel*) lub jądrem Mercera. Dla dwóch zbiorów punktów $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \subset \mathbb{R}^n$ i $Y = \{\mathbf{y}_1, \ldots, \mathbf{y}_s\} \subset \mathbb{R}^n$ i funkcji kowariancji k wprowadzimy oznaczenie

$$k(X, Y) := \begin{bmatrix} k(\mathbf{x}_1, \mathbf{y}_1) & k(\mathbf{x}_1, \mathbf{y}_2) & \cdots & k(\mathbf{x}_1, \mathbf{y}_s) \\ k(\mathbf{x}_2, \mathbf{y}_1) & k(\mathbf{x}_2, \mathbf{y}_2) & \cdots & k(\mathbf{x}_2, \mathbf{y}_s) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_m, \mathbf{y}_1) & k(\mathbf{x}_m, \mathbf{y}_2) & \cdots & k(\mathbf{x}_m, \mathbf{y}_s) \end{bmatrix}.$$

Poniżej podajemy kilka przykładów funkcji kowariancji

- *Gaussian kernel* dla normy $\|\cdot\|$ i hiper-parametrów a, l (amplituda i skala długości)
$$k(\mathbf{x}, \mathbf{y}) = a^2 \exp\left\{-\frac{1}{2l^2}\|\mathbf{x} - \mathbf{y}\|^2\right\}$$
- *Periodic kernel* dla normy $\|\cdot\|$ i hiper-parametrów a, l, p (amplituda, skala długości, okres zmienności)
$$k(\mathbf{x}, \mathbf{y}) = a^2 \exp\left\{-\frac{2}{l^2}\sin^2\left(\frac{\pi}{p}\|\mathbf{x} - \mathbf{y}\|\right)\right\}$$

- *White noise kernel* dla hiper-parametru σ

$$k(\mathbf{x}, \mathbf{y}) = \sigma^2 \delta_{\mathbf{x},\mathbf{y}}$$

- *Matérn kernel* dla normy $\|\cdot\|$ i hiper-parametrów a, l, ν (amplituda, skala długości, regularność)

$$k(\mathbf{x}, \mathbf{y}) = a^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{l} \|\mathbf{x} - \mathbf{y}\| \right)^{\nu} K_\nu \left(\frac{\sqrt{2\nu}}{l} \|\mathbf{x} - \mathbf{y}\| \right) ,$$

  gdzie $\Gamma(x)$ to funkcja gamma Eulera, a $K_\nu(x)$ to zmodyfikowana funkcja Bessela 2-go rodzaju rzędu ν.

Dodatkowo suma lub iloczyn dwóch funkcji kowariancji oraz złożenie funkcji kowariancji z wielomianem o nieujemnych współczynnikach jest również funkcją kowariancji.

Procesem Gaussowskim (z ang. *Gaussian Process*) nazywamy rodzinę skalarnych zmiennych losowych indeksowanych przez punkty $\mathbf{x} \in \mathbb{R}^n$

$$\mathcal{GP} = \{ f_{\mathbf{x}} \mid \mathbf{x} \in \mathbb{R}^n \}$$

taką że każdy skończony podzbiór $\mathcal{GP}$ ma łącznie wielowymiarowy rozkład normalny tj. dla dowolnego zbioru $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\} \subset \mathbb{R}^n$ zachodzi

$$\begin{bmatrix} f_{\mathbf{x}_1} \\ \vdots \\ f_{\mathbf{x}_m} \end{bmatrix} \sim \mathcal{N}(\boldsymbol{\mu}_X, \boldsymbol{\Sigma}_X) .$$

Zauważmy, iż proces Gaussowski możemy jednoznacznie zdefiniować podając przepisy na parametry $\boldsymbol{\mu}_X$ i $\boldsymbol{\Sigma}_X$ dla dowolnego zbioru X. W praktyce często przyjmujemy $\boldsymbol{\mu}_X = \mathbf{0}$, natomiast przepisem na macierz kowariancji może być zdefiniowana wyżej funkcja kowariancji $k(X, X)$ tj.

$$\begin{bmatrix} f_{\mathbf{x}_1} \\ \vdots \\ f_{\mathbf{x}_m} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, k(X, X)) .$$

Process Gaussowski daje nam w praktyce rozkład prawdopodobieństwa nad funkcjami $f : \mathbb{R}^n \mapsto \mathbb{R}$, których charakter jest określony przez jądro k (np. funkcja gładka dla jądra Gaussowskiego, okresowa dla jądra periodycznego, itp.). Zauważmy, że nie wnioskujemy tu o parametrach konkretnej rodziny funkcji (jak w przypadku regresji liniowej); interesuje nas jedynie rozkład predykcyjny. Załóżmy,

iż w zadanych (lub dokładnie znanych) przez nas punktach $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_m\}$ zaobserwowaliśmy wartości pewnej funkcji, o których zakładamy, iż pochodzą z procesu Gaussowskiego zadanego jądrem k, które wyraża nasze założenia a priori co do charakteru badanej funkcji

$$\mathbf{f}_X = \begin{bmatrix} f_{\mathbf{x}_1} \\ \vdots \\ f_{\mathbf{x}_m} \end{bmatrix} \sim \mathcal{N}(\mathbf{0}, k(X,X)) \,.$$

Powiedzmy, iż chcemy znać wartości $\mathbf{f}_Y$ tej funkcji w zadanych punktach $Y = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_s\}$. Ponieważ założyliśmy, iż wartości funkcji pochodzą z procesu Gaussowskiego, więc rozkład łączny $\mathbf{f}_X$ i $\mathbf{f}_Y$ jest rozkładem normalnym

$$\begin{bmatrix} \mathbf{f}_X \\ \mathbf{f}_Y \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} k(X,X) & k(X,Y) \\ k(Y,X) & k(Y,Y) \end{bmatrix}\right) .$$

Zauważmy, iż jest to instancja modelu Gaussowskiego, więc rozkład warunkowy $\mathbf{f}_Y \mid \mathbf{f}_X$ jest również rozkładem normalnym o parametrach

$$\begin{aligned} \boldsymbol{\mu} &= k(Y,X)k^{-1}(X,X)\mathbf{f}_X \\ \boldsymbol{\Sigma} &= k(Y,Y) - k(Y,X)k^{-1}(X,X)k(X,Y) \end{aligned} .$$

Dodatkową niepewność związaną z pomiarem wartości $\mathbf{f}_X$ możemy uchwycić zmieniając postać jądra

$$k(\mathbf{x},\mathbf{y}) \leftarrow k(\mathbf{x},\mathbf{y}) + \mathcal{I}_X(\mathbf{x})\sigma^2\delta_{\mathbf{x},\mathbf{y}} \,,$$

gdzie σ jest hiper-parametrem określającym precyzję pomiaru. Oczywiście k jest dalej funkcją kowariancji, gdyż takie podstawienie powoduje jedynie dodanie dodatnich członów do pewnych elementów diagonalnych macierzy kowariancji, więc macierz ta jest nadal symetryczna i dodatnio określona. Wówczas rozkład predykcyjny ma parametry

$$\begin{aligned} \boldsymbol{\mu} &= k(Y,X)\left[k(X,X) + \sigma^2\mathbf{1}\right]^{-1}\mathbf{f}_X \\ \boldsymbol{\Sigma} &= k(Y,Y) - k(Y,X)\left[k(X,X) + \sigma^2\mathbf{1}\right]^{-1}k(X,Y) \end{aligned} .$$

Pozostaje jeszcze kwestia początkowej estymacji parametrów funkcji jądrowych dla zbioru danych $D = (\mathbf{f}_X, X)$. W łatwy i wydajny sposób możemy znaleźć estymaty MLE tych parametrów. Istotnie oznaczmy zbiór parametrów jądra przez θ, wówczas

$$D \mid \theta \sim \mathcal{N}(\mathbf{0}, k(X,X;\theta))$$

czyli

$$\ell = -\log p(D \mid \theta) = \frac{1}{2}\log\left[\det\big(k(X,X;\theta) + \sigma^2\mathbf{1}\big)\right] + \frac{1}{2}\mathbf{f}_X^T\left[k(X,X;\theta) + \sigma^2\mathbf{1}\right]^{-1}\mathbf{f}_X \,.$$

Funkcję kosztu ℓ minimalizujemy numerycznie, aby znaleźć estymatę MLE parametrów θ funkcji jądrowej. Następnie parametry rozkładu predykcyjnego nad wartościami w nowych punktach znajdujemy korzystając ze wzorów dwie linijki wyżej.

2.10 Wieloklasowa regresja logistyczna

Załóżmy, iż modelujemy obserwacje postaci $(t, \mathbf{x})$, gdzie $t \in \{\tau_1, \tau_2, \ldots, \tau_s\}$ to etykieta określająca przynależność do jednej z s klas, a $\mathbf{x} \in \mathbb{R}^n$ jest znanym (lub zadanym) przez nas dokładnie wektorem cech obiektu dla których zaobserwowaną klasą jest t. Zakładamy ponadto, iż prawdopodobieństwo przynależności do klasy τ_j (jednej z s klas) dla wektora cech $\mathbf{x}$ ma postać tzw. funkcji softmax

$$\pi_j(\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \mathrm{e}^{\mathbf{w}_j^T \mathbf{x}},$$

gdzie $\mathbf{w}_j$ są estymowanymi przez nas parametrami. Ze względu na warunek unormowania musimy mieć

$$\sum_{j=1}^{s} \pi_j = 1,$$

skąd stała normalizacyjna $Z(\mathbf{x})$ ma postać

$$Z(\mathbf{x}) = \sum_{j=1}^{s} \mathrm{e}^{\mathbf{w}_j^T \mathbf{x}}.$$

Uwaga, powyższy wzór nie może być stosowany bezpośrednio do obliczeń numerycznych przez niestabilność numeryczną (wartość funkcji wykładniczej może być ogromna). W praktyce wartość funkcji softmax dla argumentu $\mathbf{z} = (z_1, \ldots, z_n)$ obliczamy jako

$$z_i' := z_i - \max_{j=1,..,n} z_j$$
$$\mathrm{softmax}(z_i) = \frac{\mathrm{e}^{z_i'}}{\sum_{j=1}^{n} \mathrm{e}^{z_j'}}.$$

Rozkład zmiennej losowej t jest w takim razie dyskretnym rozkładem wielopunktowym (z ang. *categorical distribution*) postaci

$$t \mid \mathbf{w}_1, \ldots, \mathbf{w}_s \sim \mathrm{Cat}(\pi_1(\mathbf{x}), \ldots, \pi_s(\mathbf{x})).$$

Zauważmy, iż prawdopodobieństwo wylosowania etykiety t dla parametrów $\mathbf{w}_j$ możemy zapisać jako

$$p(t \mid \mathbf{w}_1, \ldots, \mathbf{w}_s) = \prod_{j=1}^{s} \pi_j(\mathbf{x})^{\delta(t,\tau_j)}.$$

Powiedzmy, że mamy obserwacje $D = (t_1, \ldots, t_m)$ dla znanych (lub zadanych) przez nas dokładnie wektorów cech $(\mathbf{x}_1, \ldots, \mathbf{x}_m)$. Funkcja wiarygodności ma wówczas postać

$$p(D \mid \mathbf{w}_1, \ldots, \mathbf{w}_s) = \prod_{i=1}^{m} p(t_i \mid \mathbf{w}_1, \ldots, \mathbf{w}_s) = \prod_{i=1}^{m} \prod_{j=1}^{s} \pi_j(\mathbf{x}_i)^{\delta(t_i, \tau_j)} .$$

Jako prior dla parametrów $\mathbf{w}_j$ przyjmiemy rozkład normalny z pewnym hiper-parametrem γ

$$\forall j \in \{1, \ldots, s\} : \mathbf{w}_j \sim \mathcal{N}(\mathbf{0}, \gamma^{-1}\mathbf{1}) .$$

W przypadku regresji logistycznej ograniczymy się do znalezienia estymaty MAP parametrów $\mathbf{w}_j$ tak, aby w przyszłości do nowego wektora cech $\mathbf{x}$ przyporządkować klasę o największym prawdopodobieństwie $\pi_j(\mathbf{x})$. Znalezienie estymaty MAP sprowadza się do znalezienia minimum zregularyzowanej funkcji kosztu

$$\begin{aligned}
\ell^*(D \mid \mathbf{w}_1, \ldots, \mathbf{w}_s) &= -\log[p(D \mid \mathbf{w}_1, \ldots, \mathbf{w}_s)\pi(\mathbf{w}_1, \ldots, \mathbf{w}_s)] \\
&= -\log\left[\prod_{k=1}^{s} \mathrm{e}^{-\frac{\gamma}{2}\mathbf{w}_k^T \mathbf{w}_k} \prod_{i=1}^{m} \prod_{j=1}^{s} \pi_j(\mathbf{x}_i)^{\delta(t_i, \tau_j)}\right] \\
&= \frac{\gamma}{2} \sum_{j=1}^{s} \mathbf{w}_j^T \mathbf{w}_j - \sum_{i=1}^{m} \sum_{j=1}^{s} \delta(t_i, \tau_j) \log \pi_j(\mathbf{x}_i) .
\end{aligned}$$

Niestety dla tak zdefiniowanej funkcji kosztu nie można znaleźć wzoru na minimum w postaci analitycznej, dlatego wykorzystamy numeryczny algorytm optymalizacji zwany spadkiem wzdłuż gradientu.

Algorytm spadku wzdłuż gradientu

1. Wybierz parametry początkowe $\mathbf{x}_1^{(0)}, \ldots, \mathbf{x}_m^{(0)}$
2. Powtarzaj

$$\begin{aligned}
\mathbf{x}_1^{(t+1)} &= \mathbf{x}_1^{(t)} - \epsilon_1 \frac{\partial f}{\partial \mathbf{x}_1}\bigg|_{\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_m^{(t)}} \\
&\vdots \\
\mathbf{x}_m^{(t+1)} &= \mathbf{x}_m^{(t)} - \epsilon_m \frac{\partial f}{\partial \mathbf{x}_m}\bigg|_{\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_m^{(t)}}
\end{aligned}$$

gdzie $\epsilon_1, \ldots, \epsilon_m$ to hiper-parametry zwane stałymi uczącymi (z ang. *learning rate*).

Zakładając $\epsilon_1 = \ldots = \epsilon_m = \epsilon$ i wprowadzając

$$\mathbf{X} := \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_m^T \end{bmatrix}, \quad \frac{\partial f}{\partial \mathbf{X}} := \begin{bmatrix} \frac{\partial f}{\partial \mathbf{x}_1}^T \\ \vdots \\ \frac{\partial f}{\partial \mathbf{x}_m}^T \end{bmatrix}$$

możemy zapisać powyższe równania w kompaktowej formie

$$\mathbf{X}^{(t+1)} = \mathbf{X}^{(t)} - \epsilon \frac{\partial f}{\partial \mathbf{X}}\bigg|_{\mathbf{X}^{(t)}}.$$

Aby zminimalizować numerycznie funkcję kosztu ℓ^* stosując metodę spadku wzdłuż gradientu musimy obliczyć pochodne funkcji kosztu po parametrach $\mathbf{w}_j$

$$\frac{\partial \ell^*}{\partial \mathbf{w}_k} = \gamma \mathbf{w}_k - \sum_{i=1}^{m} \sum_{j=1}^{s} \delta(t_i, \tau_j) \frac{\partial}{\partial \mathbf{w}_k} \log \pi_j(\mathbf{x}_i)\,,$$

ale

$$\begin{aligned} \frac{\partial}{\partial \mathbf{w}_k} \log \pi_j(\mathbf{x}_i) &= \frac{1}{\pi_j(\mathbf{x}_i)} \frac{Z(\mathbf{x}_i) \frac{\partial \mathrm{e}^{\mathbf{x}_i^T \mathbf{w}_j}}{\partial \mathbf{w}_k} - \mathrm{e}^{\mathbf{x}_i^T \mathbf{w}_j} \frac{\partial Z(\mathbf{x}_i)}{\partial \mathbf{w}_k}}{Z^2(\mathbf{x}_i)} \\ &= \frac{Z(\mathbf{x}_i)}{\mathrm{e}^{\mathbf{x}_i^T \mathbf{w}_j}} \frac{Z(\mathbf{x}_i) \mathbf{x}_i \mathrm{e}^{\mathbf{x}_i^T \mathbf{w}_k} \delta_{jk} - \mathrm{e}^{\mathbf{x}_i^T \mathbf{w}_j} \mathrm{e}^{\mathbf{x}_i^T \mathbf{w}_k} \mathbf{x}_i}{Z^2(\mathbf{x}_i)} \\ &= \mathbf{x}_i \delta_{jk} - \mathbf{x}_i \pi_k(\mathbf{x}_i) \end{aligned}$$

zatem

$$\frac{\partial \ell^*}{\partial \mathbf{w}_k} = \gamma \mathbf{w}_k - \sum_{i=1}^{m} \mathbf{x}_i \sum_{j=1}^{s} \delta(t_i, \tau_j) \delta_{jk} + \sum_{i=1}^{m} \mathbf{x}_i \pi_k(\mathbf{x}_i) \sum_{j=1}^{s} \delta(t_i, \tau_j)\,.$$

Zauważmy jednak, iż

$$\sum_{j=1}^{s} \delta(t_i, \tau_j) = 1\,, \quad \sum_{j=1}^{s} \delta(t_i, \tau_j) \delta_{jk} = \delta(t_i, \tau_k)\,,$$

zatem ostatecznie

$$\frac{\partial \ell^*}{\partial \mathbf{w}_k} = \gamma \mathbf{w}_k + \sum_{i=1}^{m} \mathbf{x}_i \left[\pi_k(\mathbf{x}_i) - \delta(t_i, \tau_k)\right].$$

Wprowadzając macierze

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1^T \\ \vdots \\ \mathbf{x}_m^T \end{bmatrix}, \quad \mathbf{W} = \begin{bmatrix} \mathbf{w}_1^T \\ \vdots \\ \mathbf{w}_s^T \end{bmatrix},$$

$$\mathbf{S} = \begin{bmatrix} \pi_1(\mathbf{x}_1) & \cdots & \pi_s(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \pi_1(\mathbf{x}_m) & \cdots & \pi_s(\mathbf{x}_m) \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} \delta(t_1, \tau_1) & \cdots & \delta(t_1, \tau_s) \\ \vdots & \ddots & \vdots \\ \delta(t_m, \tau_1) & \cdots & \delta(t_m, \tau_s) \end{bmatrix}$$

możemy w takim razie zapisać zdefiniowaną wyżej macierz pochodnych wymaganych do algorytmu spadku wzdłuż gradient w kompaktowej formie jako

$$\frac{\partial \ell^*}{\partial \mathbf{W}} = (\mathbf{S} - \mathbf{T})^T \mathbf{X}\,.$$

Zauważmy, iż zregularyzowana funkcja kosztu rośnie wraz ze wzrostem liczby obserwacji m. Wynika z tego, iż stała ucząca musi być zależna od liczby przykładów. Możemy na przykład stwierdzić, iż $\epsilon \leftarrow m^{-1}\epsilon$ i wówczas minimalizujemy tak naprawdę średni koszt ℓ^*/m.

2.11 Naiwny klasyfikator Bayesowski

Ponownie rozpatrzmy problem klasyfikacji, w którym dla danych zmiennych objaśniających $\mathbf{x}$ mamy przyporządkować jedną z k klas. Interesuje nas rozkład warunkowy

$$p(\tau_i \mid \mathbf{x}) = p(\tau_i \mid x_1, .., x_d)\,,$$

gdzie $i = 1, .., k$. Zamiast konstruować parametryczny model statystyczny jak w przypadku regresji logistycznej skorzystamy teraz z twierdzenia Bayesa

$$p(\tau_i \mid x_1, .., x_d) = \frac{p(x_1, .., x_d \mid \tau_i) p(\tau_i)}{p(x_1, .., x_d)}$$

wraz z *naiwnym* założeniem, iż cechy $x_1, .., x_d$ są warunkowo niezależne względem τ_i tj.

$$p(\tau_i \mid x_1, .., x_d) = \frac{p(x_1 \mid \tau_i) ... p(x_d \mid \tau_i) p(\tau_i)}{p(x_1, .., x_d)}\,.$$

Dla danych cech $x_1, .., x_d$ wybierzemy klasę zgodnie z regułą MAP, czyli klasę, która maksymalizuje wypisane wyżej prawdopodobieństwo a posteriori (czyli efektywnie maksymalizuje licznik tego wyrażenia)

$$t(x_1, .., x_d) = \arg\max_{i=1,..,k} p(\tau_i \mid x_1, .., x_d) = \arg\max_{i=1,..,k} p(x_1 \mid \tau_i) ... p(x_d \mid \tau_i) p(\tau_i)\,.$$

2.12 Wnioskowanie metodami Monte Carlo

Całe wnioskowanie Bayesowskie opiera się na wyznaczaniu rozkładów a posteriori, które wyrażają naszą wiedzę o estymowanym parametrze. Do tej pory rozważaliśmy modele Bayesowskie dla których prior i wiarygodność były dane przez rozkłady normalne. Dzięki temu mogliśmy wyprowadzić analityczne wzory na parametry rozkładu a posteriori, który również był rozkładem normalnym. Dla wielu interesujących modeli nie jesteśmy jednak w stanie tego zrobić (np. w zagadnieniu regresji logistycznej ograniczyliśmy się jedynie do estymaty punktowej), gdyż obliczenie stałej normalizującej dla rozkładu $p(\theta \mid D)$ może wymagać obliczenia całki, której nie jesteśmy w stanie wyrazić w sposób jawny lub sumy po wykładniczo wielu elementach. Wnioskowanie Bayesowskie można jednak prowadzić w modelach, w których nie dysponujemy jawnym wzorem na gęstość prawdopodobieństwa rozkładu a posteriori. Okazuje się, iż do generowania próbek z rozkładu $p(\theta \mid D)$ wystarcza znajomość tego rozkładu z dokładnością do stałej normalizującej, a zatem wystarczy znać rozkład łączny $p(\theta, D) = p(D \mid \theta)\pi(\theta)$. Generowanie próbek z kolei wystarcza natomiast, na mocy silnego prawa wielkich liczb, do szacowania wartości średnich dowolnych funkcji estymowanego parametru θ. Przypomnijmy, iż na mocy silnego prawa wielkich liczb ciąg średnich częściowych $(\overline{X}_n)$ ciągu zmiennych losowych (X_n) i.i.d. z rozkładu $X \sim \mathcal{D}$ jest zbieżny z prawdopodobieństwem 1 do wartości oczekiwanej $\mathbb{E}[X]$ tj.

$$P\left(\lim_{n\to\infty} \overline{X}_n = \mathbb{E}[X]\right) = 1\,.$$

Wartość oczekiwaną $\mathbb{E}[X]$ możemy zatem przybliżyć średnią $\overline{X}_n$ z dużej ilości próbek.

Wnioskowanie Monte Carlo pozwala nam szacować różne wielkości w tzw. hierarchicznych modelach Bayesowskich (z ang. *Bayesian hierarchical modeling*). Rozważmy jeszcze raz przykład regresji liniowej w ujęciu Bayesowskim, ale rozważmy teraz model postaci

$$\begin{aligned} \sigma^2 &\sim \mathcal{D}(\lambda) \\ \mathbf{w} &\sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0) \\ y \mid \mathbf{w}, \sigma^2 &\sim \mathcal{N}(\mathbf{w}^T\mathbf{x}, \sigma^2) \end{aligned} \quad ,$$

gdzie $\lambda, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0$ są pewnymi hiper-parametrami. Dla takiego modelu nie możemy w ogólności znaleźć jawnej postaci rozkładu a posteriori. Jeśli jednak umiemy generować próbki z rozkładu łącznego

$$Z \cdot p(\mathbf{w}, \sigma^2 \mid D) = p(D, \mathbf{w}, \sigma^2) = p(D \mid \mathbf{w}, \sigma^2)\pi(\mathbf{w})\pi(\sigma^2)$$

to wszystkie interesujące wielkości możemy oszacować jako odpowiednie średnie. Pozostaje pytanie w jaki sposób generować próbki ze skomplikowanych rozkła-

dów prawdopodobieństwa, których gęstości znamy jedynie z dokładnością do stałej normalizującej. Poniżej przedstawimy dwa algorytmy próbkowania: algorytm IS oraz Metropolisa–Hastingsa będący szczególną realizacją całej rodziny algorytmów próbkowania zwanych Markov Chain Monte Carlo (MCMC).

2.12.1 Algorytm Importance Sampling (IS)

Załóżmy, iż chcemy obliczyć wartość oczekiwaną pewnej funkcji zmiennej losowej $\mathbf{x}$ względem skomplikowanego rozkładu prawdopodobieństwa $p(\mathbf{x})$, który znamy jedynie z dokładnością do stałej normalizującej

$$p(\mathbf{x}) = \frac{1}{Z_p}\tilde{p}(\mathbf{x})$$

tj. szukamy

$$\mathbb{E}_p[f(\mathbf{x})] = \int f(\mathbf{x})p(\mathbf{x})\mathrm{d}^n\mathbf{x}\,.$$

Jeśli umiemy generować próbki $\mathbf{x}$ z innego (prostszego) rozkładu $q(\mathbf{x})$, który nazywamy rozkładem proponującym kandydatów (z ang. *proposal distribution*) to możemy zapisać

$$\begin{aligned}\mathbb{E}_p[f(\mathbf{x})] &= \int\limits_{\mathbb{R}^n} f(\mathbf{x})p(\mathbf{x})\mathrm{d}^n\mathbf{x} = \int\limits_{\mathbb{R}^n} f(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}q(\mathbf{x})\mathrm{d}^n\mathbf{x} \\ &= \mathbb{E}_q\left[f(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})}\right] = \frac{Z_q}{Z_p}\mathbb{E}_q\left[f(\mathbf{x})\frac{\tilde{p}(\mathbf{x})}{\tilde{q}(\mathbf{x})}\right]\,.\end{aligned}$$

Zakładamy tutaj, iż nośnik rozkładu p zawiera się w nośniku q tj. $\operatorname{supp} p \subseteq \operatorname{supp} q$. Stosunek stałych Z_p/Z_q również możemy oszacować z próbek z q, gdyż mamy

$$Z_p = \int\limits_{\mathbb{R}^n} \tilde{p}(\mathbf{x})\,\mathrm{d}^n\mathbf{x} = Z_q \int\limits_{\mathbb{R}^n} \frac{\tilde{p}(\mathbf{x})}{\tilde{q}(\mathbf{x})}q(\mathbf{x})\,\mathrm{d}^n\mathbf{x} = Z_q\mathbb{E}_q\left[\frac{\tilde{p}(\mathbf{x})}{\tilde{q}(\mathbf{x})}\right]\,,$$

skąd ostatecznie

$$\mathbb{E}_p[f(\mathbf{x})] = \frac{\mathbb{E}_q\left[f(\mathbf{x})\frac{\tilde{p}(\mathbf{x})}{\tilde{q}(\mathbf{x})}\right]}{\mathbb{E}_q\left[\frac{\tilde{p}(\mathbf{x})}{\tilde{q}(\mathbf{x})}\right]}\,.$$

Jeśli z rozkładu q wygenerowaliśmy próbki $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$ to na mocy silnego prawa wielkich liczb mamy

$$\mathbb{E}_p[f(\mathbf{x})] \approx \frac{\sum_{i=1}^m f(\mathbf{x}_i)\frac{\tilde{p}(\mathbf{x}_i)}{\tilde{q}(\mathbf{x}_i)}}{\sum_{i=1}^m \frac{\tilde{p}(\mathbf{x}_i)}{\tilde{q}(\mathbf{x}_i)}} = \sum_{i=1}^m \lambda_i f(\mathbf{x}_i)\,,$$

gdzie

$$\lambda_i = \frac{\tilde{p}(\mathbf{x}_i)/\tilde{q}(\mathbf{x}_i)}{\sum_{j=1}^{m} \tilde{p}(\mathbf{x}_j)/\tilde{q}(\mathbf{x}_j)} \,.$$

Algorytm Importance Sampling jest prostym algorytmem Monte Carlo, który ma jeden zasadniczy problem. W jaki sposób mamy wybrać rozkład proponujący kandydatów q? Pewną odpowiedź na to pytanie sugeruje analiza wariancji statystyki

$$\overline{f}_m(\mathbf{x}_1, \ldots, \mathbf{x}_m) = \frac{1}{m} \sum_{i=1}^{m} \frac{f(\mathbf{x}_i)p(\mathbf{x}_i)}{q(\mathbf{x}_i)}$$

dla $\mathbf{x}_i \sim q$ i zakładając dla uproszczenia, iż f jest funkcją skalarną mamy

$$\mathrm{Var}[\overline{f}_m] = \frac{1}{m}\mathrm{Var}_q \left[f(\mathbf{x})\frac{p(\mathbf{x})}{q(\mathbf{x})} \right] = \frac{1}{m} \int\limits_{\mathbb{R}^n} \frac{(f(\mathbf{x})p(\mathbf{x}) - \mu_f q(\mathbf{x}))^2}{q(\mathbf{x})} \,\mathrm{d}^n\mathbf{x} \,.$$

Chcemy oczywiście, aby wariancja była jak najmniejsza, gdyż wówczas mała liczba próbek da dobre przybliżenie wartości oczekiwanej. Rozkład proponujący kandydatów powinien być zatem proporcjonalny do $f(\mathbf{x})p(\mathbf{x})$, co może być trudne do praktycznego zrealizowania.

2.12.2 Algorytm Metropolisa–Hastingsa

Cała klasa algorytmów próbkowania MCMC opiera się na idei wyrażenia generowania próbek jako ewolucji pewnego łańcucha Markowa. Łańcuchem Markowa nazywamy ciąg zmiennych losowych (X_t) o wartościach w $\mathbb{R}^n$ taki, że spełnione jest kryterium Markowa

$$\forall A \subset \mathbb{R}^n : P(X_t \in A \mid X_{t-1} = \mathbf{x}_{t-1}, \ldots, X_0 = \mathbf{x}_0) = P(X_t \in A \mid X_{t-1} = \mathbf{x}_{t-1}) \,.$$

Elementy ciągu nazywamy stanami łańcucha Markowa. Dany łańcuch jest zadany jednoznacznie przez podanie gęstości prawdopodobieństwa przejścia łańcucha ze stanu $\mathbf{x} \to \mathbf{y}$, którą będziemy oznaczać przez $\pi(\mathbf{y} \mid \mathbf{x})$ (zakładamy, iż prawdopodobieństwo przejścia jest niezależne od chwili t – łańcuch taki nazywamy jednorodnym). Funkcja π spełnia oczywiście warunek unormowania

$$\int\limits_{\mathbb{R}^n} \pi(\mathbf{y} \mid \mathbf{x}) \,\mathrm{d}^n\mathbf{y} \,,$$

istotnie prawdopodobieństwo przejścia gdziekolwiek ze stanu $\mathbf{x}$ jest równe 1. Będziemy zakładać dodatkowo, iż $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n : \pi(\mathbf{y} \mid \mathbf{x}) > 0$. Rozkład $p(\mathbf{x})$ łańcucha Markowa (tj. rozkład prawdopodobieństwa z którego losujemy stan łańcucha w

danej chwili t) z daną funkcją przejścia π nazwiemy rozkładem stacjonarnym tego łańcucha iff

$$p(\mathbf{y}) = \int\limits_{\mathbb{R}^n} \pi(\mathbf{y} \mid \mathbf{x}) p(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} \; .$$

Rozkład stacjonarny danego łańcucha oznaczymy przez $p^*(\mathbf{x})$. Zauważmy, iż jeśli stan początkowy łańcucha X_0 pochodzi z rozkładu stacjonarnego p^* to każdy kolejny stan X_t również pochodzi z rozkładu stacjonarnego. Jeśli z kolei stan początkowy pochodzi z jakiegoś innego rozkładu p_0 to rozkład łańcucha w chwili t jest dany przez relację rekurencyjną

$$p_t(\mathbf{y}) = \int\limits_{\mathbb{R}^n} \pi(\mathbf{y} \mid \mathbf{x}) p_{t-1}(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} \; , \quad \text{dla } t > 1.$$

Rozkładem granicznym łańcucha Markowa nazwiemy granicę w sensie zbieżności punktowej

$$\lim_{t\to\infty} p_t(\mathbf{x}) \, .$$

Przy podanych wyżej założeniach istnieje twierdzenie, które mówi iż taki łańcuch Markowa posiada jednoznaczny rozkład stacjonarny tożsamy z rozkładem granicznym. Ponadto warunkiem wystarczającym, aby dany rozkład $p(\mathbf{x})$ był rozkładem stacjonarnym łańcucha Markowa jest

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^n : \pi(\mathbf{y} \mid \mathbf{x}) p(\mathbf{x}) = \pi(\mathbf{x} \mid \mathbf{y}) p(\mathbf{y}) \, ,$$

co wynika z scałkowania powyższego równania

$$\int\limits_{\mathbb{R}^n} \pi(\mathbf{y} \mid \mathbf{x}) p(\mathbf{x}) \, \mathrm{d}^n\mathbf{x} = \int\limits_{\mathbb{R}^n} \pi(\mathbf{x} \mid \mathbf{y}) p(\mathbf{y}) \, \mathrm{d}^n\mathbf{x} = p(\mathbf{y}) \int\limits_{\mathbb{R}^n} \pi(\mathbf{x} \mid \mathbf{y}) \, \mathrm{d}^n\mathbf{x} = p(\mathbf{y}) \, .$$

Kryterium to nazywamy kryterium lokalnego balansu (z ang. *detailed balance condition*).

Podstawowa idea wykorzystania łańcuchów Markowa do generowania próbek ze skomplikowanego rozkładu p jest więc następująca: tworzymy łańcuch Markowa opisany powyżej, dla którego p jest rozkładem stacjonarnym, wówczas rozpoczynając w dowolnym dopuszczalnym stanie początkowym X_0 po wykonaniu dużej liczby kroków (etap ten nazywamy okresem przejściowym z ang. *burn-in period*) stan X_t (dla $t \gg 1$) tego łańcucha będzie w przybliżeniu pochodził z rozkładu granicznego p (nie jest jednak prosto stwierdzić po jak długim okresie przejściowym przybliżenie to jest wystarczająco dobre). Aby otrzymać z takiej procedury próbki prawdziwie i.i.d. każda z próbek musiałaby pochodzić z ponownego uruchomienia takiego łańcucha. Oczywiście jest to nieefektywne, więc w praktyce generujemy

próbki z jednego łańcucha po prostu odrzucając pewne z nich tak aby uniknąć znaczących korelacji.

Pozostaje pytanie jak skonstruować funkcję przejścia $\pi(\mathbf{y} \mid \mathbf{x})$ dla danego rozkładu granicznego $p(\mathbf{x})$. Podstawową konstrukcję podaje algorytm Metropolisa –Hastingsa:

1. Jako stan początkowy przyjmij dowolną dopuszczalną wartość $\mathbf{x} \leftarrow \mathbf{x}_0$.

2. Powtarzaj:

   (a) Będąc w aktualnym stanie $\mathbf{x}$ z prostego rozkładu proponującego kandydatów $q(\mathbf{y} \mid \mathbf{x})$ wylosuj kandydata $\mathbf{y}$ na wartość łańcucha w kolejnym stanie.

   (b) Z prawdopodobieństwem

   $$r(\mathbf{y} \mid \mathbf{x}) = \min\left\{1, \frac{p(\mathbf{y})q(\mathbf{x} \mid \mathbf{y})}{p(\mathbf{x})q(\mathbf{y} \mid \mathbf{x})}\right\}$$

   zaakceptuj kandydata jako nowy stan i przejdź do stanu $\mathbf{y}$. W przeciwnym razie pozostać w stanie $\mathbf{x}$

Funkcja przejścia ma zatem postać

$$\pi_{\mathrm{MH}}(\mathbf{y} \mid \mathbf{x}) = q(\mathbf{y} \mid \mathbf{x})r(\mathbf{y} \mid \mathbf{x})\,.$$

Pozostaje tylko wykazać, iż spełnione jest kryterium lokalnego balansu. Istotnie mamy

$$\begin{aligned}\pi_{\mathrm{MH}}(\mathbf{y} \mid \mathbf{x})p(\mathbf{x}) &= \min\{q(\mathbf{y} \mid \mathbf{x})p(\mathbf{x}), q(\mathbf{x} \mid \mathbf{y})p(\mathbf{y})\} \\ \pi_{\mathrm{MH}}(\mathbf{x} \mid \mathbf{y})p(\mathbf{y}) &= \min\{q(\mathbf{x} \mid \mathbf{y})p(\mathbf{y}), q(\mathbf{y} \mid \mathbf{x})p(\mathbf{x})\}\end{aligned},$$

skąd $\pi_{\mathrm{MH}}(\mathbf{y} \mid \mathbf{x})p(\mathbf{x}) = \pi_{\mathrm{MH}}(\mathbf{x} \mid \mathbf{y})p(\mathbf{y})$. Zauważmy, iż nie musimy znać $p(\mathbf{x})$ z dokładnością do stałej normalizującej, gdyż

$$\frac{p(\mathbf{y})}{p(\mathbf{x})} = \frac{\tilde{p}(\mathbf{y})/Z_p}{\tilde{p}(\mathbf{x})/Z_p} = \frac{\tilde{p}(\mathbf{y})}{\tilde{p}(\mathbf{x})}\,.$$

Poza algorytmem Metropolisa–Hastingsa jest wiele innych algorytmów z rodziny MCMC. Większość z nich implementuje konkretny sposób generowania (zostawiając resztę struktury) tak, aby zmniejszyć korelację po okresie przejściowym i przyspieszyć zbieżność. Standardowo wykorzystywanymi algorytmami z tej klasy są algorytmy HMC (*Hamiltonian Monte Carlo*) oraz NUTS (*No U-Turn Sampler*).

3 Podstawy uczenia maszynowego

3.1 Preprocessing danych

Kluczem do uzyskania dobrych wyników przy korzystaniu z algorytmów uczenia maszynowego jest odpowiednie przygotowanie danych i ich preprocessing. Typowo preprocessing składa się z:

- wczytywania danych (np. plików w formacie csv)
- czyszczenia i imputacji danych (usunięcie silnie skorelowanych cech, usunięcie oczywistych outlierów lub cech posiadających zbyt dużo brakujących wartości, uzupełnienie wartości cech)
- wstępnej analizy/eksploracji danych (wykresy, histogramy, correlation heatmap)
- podział zbioru danych na zbiór treningowy i testowy
- skalowanie zmiennych numerycznych (parametry potrzebne do skalowania są obliczane jedynie na zbiorze treningowym, aby uniknąć tzw. *data leakage*). Wydzieliliśmy dane testowe po to, żeby sprawdzać, jak model poradzi sobie z danymi, których do tej pory nigdy nie widział, bo to właśnie takie dane będzie on dostawać w praktyce, po wdrożeniu do realnego systemu. Ta ocena obejmuje też etap preprocessingu, w tym skalowania. Więc jeśli etap preprocessingu zobaczy dane testowe, to nie będziemy w stanie uczciwie estymować jego zachowania na nowych danych. Wykorzystanie danych testowych w procesie treningu to błąd wycieku danych. Skutkuje on niepoprawnym, nadmiernie optymistycznym oszacowaniem jakości modelu.
- one-hot encoding zmiennych kategorycznych (należy pamiętać, że kodując zmienną kategoryczną o N wartościach używamy $N-1$ *dummy variables*, aby nie wpaść w tzw. *dummy variable trap* – zależność liniową między cechami)
- transformacja logarytmiczna zmiennej objaśnianej – jeśli histogram wartości zmiennej objaśnianej nie przypomina rozkładu normalnego (jest asymetryczny, skośny) to przekształcamy wartości tej zmiennej $y \mapsto \log(y)$. Dodatkowa korzyść z takiej transformacji jest taka, że regresja liniowa przewiduje dowolne wartości rzeczywiste. Po przekształceniu logarytmicznym jest to całkowicie ok, natomiast w oryginalnej przestrzeni trzeba by wymusić przewidywanie tylko wartości pozytywnych.
- feature engineering – dodanie wielomianów cech do naszych danych lub skonstruowanie innych danych (np. dla przebiegów czasowych dodanie cech określających miesiąc, dzień itp.)

3.2 Tuning hiperparametrów i walidacja skrośna

Praktycznie wszystkie modele ML mają hiperparametry, często liczne, które w zauważalny sposób wpływają na wyniki, a szczególnie na underfitting i overfitting. Ich wartości trzeba dobrać zatem dość dokładnie. Jak to zrobić? Proces doboru hiperparametrów nazywa się tuningiem hiperparametrów (hyperparameter tuning).

Istnieje na to wiele sposobów. Większość z nich polega na tym, że trenuje się za każdym razem model z nowym zestawem hiperparametrów i wybiera się ten zestaw, który pozwala uzyskać najlepsze wyniki. Metody głównie różnią się między sobą sposobem doboru kandydujących zestawów hiperparametrów.

Najprostsze i najpopularniejsze to:

- pełne przeszukiwanie (grid search) - definiujemy możliwe wartości dla różnych hiperparametrów, a metoda sprawdza ich wszystkie możliwe kombinacje (czyli siatkę),
- losowe przeszukiwanie (randomized search) - definiujemy możliwe wartości jak w pełnym przeszukiwaniu, ale sprawdzamy tylko ograniczoną liczbę losowo wybranych kombinacji.

Jak ocenić, jak dobry jest jakiś zestaw hiperparametrów? Nie możemy sprawdzić tego na zbiorze treningowym - wyniki byłyby zbyt optymistyczne. Nie możemy wykorzystać zbioru testowego - mielibyśmy data leakage, bo wybieralibyśmy model explicite pod nasz zbiór testowy. Trzeba zatem osobnego zbioru, na którym będziemy na bieżąco sprawdzać jakość modeli dla różnych hiperparametrów. Jest to zbiór walidacyjny (validation set). Zbiór taki wycina się ze zbioru treningowego. Dzielimy zatem nasze dane nie na dwie, ale trzy części: treningową, walidacyjną i testową. Typowe proporcje to 60-20-20% lub 80-10-10%.

Jednorazowy podział zbioru na części nazywa się split validation lub holdout. Używamy go, gdy mamy sporo danych, i 10-20% zbioru jako dane walidacyjne czy testowe to dość dużo, żeby mieć przyzwoite oszacowanie. Zbyt mały zbiór walidacyjny czy testowy da nam mało wiarygodne wyniki - nie da się nawet powiedzieć, czy zbyt pesymityczne, czy optymistyczne! W praktyce niestety często mamy mało danych. Trzeba zatem jakiejś magicznej metody, która stworzy nam więcej zbiorów walidacyjnych z tej samej ilości danych. Taką metodą jest walidacja skrośna (cross-validation, CV). Polega na tym, że dzielimy zbiór na K równych podzbiorów, tzw. foldów. Każdy podzbiór po kolei staje się zbiorem walidacyjnym, a pozostałe łączymy w zbiór treningowy. Trenujemy zatem K modeli dla tego samego zestawu hiperparametrów i każdy testujemy na zbiorze walidacyjnym. Mamy K wyników dla zbiorów walidacyjnych, które możemy uśrednić (i ew. obliczyć odchylenie standardowe). Takie wyniki są znacznie bardziej wiarygodne.

Szczególnym przypadkiem jest Leave-One-Out Cross-Validation (LOOCV), w którym ilość podzbiorów (foldów) jest równa ilości rekordów. Czyli w danej chwili

tylko 1 przykład jest zbiorem walidacyjnym. Daje to możliwość prawie całkowitego wykorzystania naszych danych (w każdej iteracji musimy wydzielić tylko 1 przykład na zbiór walidacyjny, cała reszta jest naszym zbiorem treningowym), ale wprowadza ogromny koszt obliczeniowy. Jest to opłacalne tylko w szczególnych przypadkach.

3.3 Metryki do oceny klasyfikacji

W przypadku regresji ocena jakości modelu sprawdzała się do użycia jednej z metryk RMSE, MAE. W przypadku klasyfikacji, a zwłaszcza klasyfikacji silnie niezbalansowanej ocena jakości klasyfikatora jest zadaniem trudniejszym. Rozpatrujemy tutaj jedynie klasyfikatory binarne. Dla danego zestawu cech odpowiedź klasyfikatora może być jedną z czterech:

- *True Positive* – poprawnie zaklasyfikowaliśmy klasę pozytywną jako pozytywną
- *True Negative* – poprawnie zaklasyfikowaliśmy klasę negatywną jako negatywną
- *False Positive* – niepoprawnie zaklasyfikowaliśmy klasę negatywną jako pozytywną
- *False Negative* – niepoprawnie zaklasyfikowaliśmy klasę pozytywną jako negatywną

Na podstawie ilości TP, TN, FP i FN na zbiorze testowym możemy wprowadzić następujące podstawowe metryki

- *precision* – mówi nam o tym jak pewnie klasyfikator znajduje klasę pozytywną
$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$
- *recall* – mówi nam o tym jak dobrze wykrywamy klasę pozytywną
$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Zauważmy, że metryki te mają zasadniczy problem, jeśli chcielibyśmy używać tylko jednej z nich: klasyfikator, który zawsze zwraca klasę pozytywną ma maksymalny recall, a klasyfikator, który zawsze zwraca klasę negatywną ma nieskończony precision. Nie możemy zatem używać tych miar osobno, tj. dobry klasyfikator powinien

mieć wysokie zarówno precision, jak i recall. Miarą, która łączy precision i recall jest F1-score zdefiniowany jako średnia harmoniczna precision i recall

$$F_1 = \frac{2}{\text{precision}^{-1} + \text{recall}^{-1}} .$$

Metryka ta przyjmuje wartości od 0 do 1 i im jej wartość większa tym lepszy klasyfikator. Zauważmy bowiem, iż aby F1-score był wysoki zarówno precision, jak i recall muszą mieć wartość bliską 1.

Czasami bywa tak, iż każda z czterech możliwości klasyfikatora ma realny koszt (np. przy predykcji, czy klient przestanie używać naszych usług) i wtedy powinniśmy używać takiej empirycznej funkcji kosztu (patrz: 2.1).

W przypadku klasyfikacji niezbalansowanej często na wytrenowanym już klasyfikatorze dokonujemy tzw. *threshold tuning*. Normalnie w przypadku klasyfikatora, który zwraca prawdopodobieństwa klas wybiera klasę o największym prawdopodobieństwie (czyli w przypadku klasyfikacji binarnej o prawdopodobieństwie większym od 0.5), threshold tuning polega na obniżeniu tego progu. Pytanie jak wybrać ten próg. Wykorzystujemy do tego celu krzywą ROC (z ang. *Receiver Operating Characteristic curve*) czyli wykres parametryczny recall i FPR (*False Positive Rate*, czyli recall dla klasy negatywnej) dla różnych progów klasyfikacji i wybieramy próg, dla którego F1-score jest maksymalny. Krzywa ROC pozwala dodatkowo wprowadzić jeszcze jedną miarę oceny klasyfikatora, która jest niezależna od przyjętego progu, tzw. AUROC (z ang. *Area under ROC*), czyli pole powierzchni pod wykresem krzywej ROC. Miara ta przyjmuje wartości z zakresu [0;1] i im jest większa tym lepszy mamy klasyfikator. AUROC jest dobrą miarą jakości klasyfikatora (tj. dobrze różnicuje modele) jeśli mamy klasyfikację umiarkowanie niezbalansowaną. W przypadku klasyfikacji ekstremalnie niezbalansowanej używamy podobnej miary od AUROC zwanej AUPRC (z ang. *Area under Precision-Recall Curve*) zdefiniowanej jako pole powierzchni pod wykresem krzywej PRC, czyli krzywej parametrycznej precision i recall dla różnych wartości progu klasyfikacji.

3.4 Metody oparte o sąsiedztwo

Metody oparte o sąsiedztwo to zbiór metod, które wykorzystują najbliższych (w sensie pewnej metryki lub półmetryki) sąsiadów (z ang. *nearest neighbors*) dla danego nowego punktu. Zasadniczą ideą tych metod jest iż punkty leżące blisko siebie w pewnej przestrzeni powinny mieć podobne własności, czyli taka przestrzeń metryczna powinna być *semantycznie sensowna*.

Jednym z najprostszych algorytmów z tej dziedziny jest *k-nearest neighbors* (kNN) używany najczęściej do klasyfikacji, ale możliwe jest również użycie go do regresji. Załóżmy, iż mamy zbiór treningowy $\mathcal{X} = \{(\mathbf{x}_i, t_i)\}_{i=1}^n$, gdzie $\mathbf{x}$ to wektor

cech, a t to prawidłowa klasa. Chcąc przewidzieć klasę dla nowego wektora cech $\mathbf{x}$ znajdujemy k najbliższych sąsiadów $\mathbf{x}$ w zbiorze $\{\mathbf{x}_i\}_{i=1}^n$ względem metryki d, zliczamy klasy najbliższych sąsiadów i zwracamy klasę występującą najczęściej lub rozkład prawdopodobieństwa nad możliwymi klasami jako stosunek liczby punktów z daną klasą do liczby sąsiadów k. W przypadku regresji zamiast zwracać klasę występującą najczęściej, zwracamy średnią arytmetyczną wartości t dla k najbliższych sąsiadów (uwaga! użycie średniej powoduje, iż kNN w przypadku regresji potrafi tylko interpolować wartości).

Liczba sąsiadów k jest hiperparametrem modelu kNN, który silnie wpływa na jakość predykcji. Zasadniczo im większa wartość k tym większy jest bias modelu, natomiast im mniejsza tym większa jest variance modelu. kNN jest modelem nieliniowym i potrafi nauczyć się nietrywialnych granic decyzyjnych. Ma również dużą pojemność (z ang. *capacity*) w porównaniu do modeli liniowych (np. regresji logistycznej).

Standardowo wykorzystywane metryki w kNN to

- metryka euklidesowa (L2) $d(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(\mathbf{x}_i - \mathbf{x}_j)^T(\mathbf{x}_i - \mathbf{x}_j)}$;
- metryka manhattan (L1) $d(\mathbf{x}_i, \mathbf{x}_j) = \sum_{k=1} |x_i^k - x_j^k|$;
- podobieństwo cosinusowe $\text{sim}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i^T \mathbf{x}_j}{\|\mathbf{x}_i\| \|\mathbf{x}_j\|}$.

Często wykorzystywaną modyfikacją kNN jest tzw. ważenie sąsiadów. Opiera się ono na prostej obserwacji, iż nie wszyscy najbliżsi sąsiedzi danego punktu są jednakowo ważni, gdyż w rzadkich obszarach przestrzeni sąsiedzi ci mogą być dosyć daleko. Rozwiązaniem jest ważenie sąsiadów odwrotnością odległości do nich. Wówczas w przypadku klasyfikacji prawdopodobieństwa obliczamy jako stosunek sumy wag danej klasy do sumy wszystkich wag, a w przypadku regresji obliczamy średnią ważoną zamiast arytmetycznej.

W vanilla kNN etap treningu polega jedynie na zapamiętaniu zbioru danych i dopiero na etapie predykcji znajdowani są najbliżsi sąsiedzi. Takie podejście jest bardzo nieefektywne, złożoność obliczeniowa etapu predykcji to $O(kn)$. W praktyce podczas treningu budowana jest pomocnicza struktura danych, która pozwala efektywniej przeszukiwać punkty w celu znalezienia najbliższych sąsiadów. Przykładami takich struktur są np. k-d tree, ball tree, quad tree, octree. W przypadku k-d tree złożoność treningu (budowy struktury) wynosi $O(n \log n)$, natomiast wyszukania k sąsiadów $O(k \log n)$. W przypadku zastosowania kNN do wyszukiwania taka złożoność jest niestety nadal za duża w praktyce (chcemy wyszukiwać wśród setek milionów obiektów), dlatego powstały również metody aproksymacyjne (z ang. *Approximate Nearest Neighbors*). W algorytmach aproksymacyjnych zamiast znajdować dokładnych k najbliższych sąsiadów znajdujemy k sąsiadów, którzy nie-

koniecznie muszą być rzeczywiście najbliższymi, potrafimy to jednak zrobić znacznie szybciej.

3.5 Selekcja cech

Zwykle dane w zbiorach wykorzystywanych w uczeniu maszynowym mają dużą wymiarowość (wiele cech). Nas jednak najbardziej interesuje tzw. *intrinsic dimensionality*, czyli podprzestrzeń, która jest realnie ważna dla problemu klasyfikacji lub regresji. Celem selekcji cech jest właśnie wybór tych cech, które są istotne. Typowo obliczamy ważność cech (z ang. *feature importance*) i usuwamy te najmniej wartościowe. Takie podejście może pomóc nam usunąć szum z danych i poprawić wyniki modeli (jest to szczególne ważne dla modeli, które używają cech wprost np. kNN, Naive Bayes). Najczęstszymi podejściami do selekcji cech są tzw. metody *filter* i *embedded*.

- **Metody *filter***. Waga cechy jest obliczana na podstawie pewnej ogólnej miary jakości cech; są typowo oparte na pewnych statystykach i sprawdzają pojedynczą zmienną naraz (tzw. metody *univariate*).
  - ***Variance threshold***. Przeskalowujemy cechy korzystajac z MinMax, aby wszystkie miały ten sam zakres wartości. Obliczamy wariancję każdej cechy. Odrzucamy cechy o bardzo małej wariancji, oznacza to bowiem, iż są praktycznie stałe. Próg typowo jest ustalany na bardzo niski typu 0.01.
  - **Korelacja cech**. Obliczamy macierz korelacji i cech, po czym z każdej pary eliminujemy najmocniej skorelowane cechy, poprzez usunięcie tej o niższej wariancji lub tej, która ma średnio większe korelacje z innymi cechami.
  - **Korelacja ze zmienną zależną**. Obliczamy korelacje między cechami, a zmienną zależną i usuwamy te o wartości bezwzględnej bliskiej 0.
  - ***Mutual Information***. Obliczamy informację wzajemną (patrz: 1.17.2) i na jej podstawie usuwamy cechy.
- **Metody *embedded***. W metodach embedded trenujemy jakiś parametryczny model uczenia maszynowego na naszych danych i wagi tego modelu interpretujemy jako wagi cech w zbiorze. Metody te wymagają dobrych modeli, zauważmy bowiem, że wagi modelu mówią tak naprawdę jak ważna jest dana cecha dla predykcji tego modelu, a nie ogólnie dla zagadnienia. Jako modele można wybrać proste modele liniowe, w których mamy prostą interpretację wag.

3.6 Redukcja wymiarowości

Ideą redukcji wymiarowości jest rzutowanie cech (z ang. *feature projection*) na nową przestrzeń o mniejszej liczbie wymiarów. W przeciwieństwie do selekcji cech, zamiast usuwać cechy tworzymy ich kombinacje (liniowe lub nieliniowe) i w taki sposób otrzymujemy podprzestrzeń realnie ważną dla problemu. Zaletą takiego podejścia jest to, iż rozważa złożone przekształcenia całej przestrzeni cech, ale z tego względu jest mniej interpretowalne od metod selekcji cech.

Najbardziej typowe są liniowe metody redukcji wymiarowości w tym przede wszystkim SVD (z ang. *Singular Value Decomposition*) i PCA (z ang. *Principal Components Analysis*). W przypadku podejść nieliniowych najpopularniejsze metody to UMAP (z ang. *Uniform Manifold Approximation and Projection*) oraz t-SNE (z ang. *t-distributed Stochastic Neighbor Embedding*).

4 Sieci neuronowe

4.1 Tensory

W przypadku uczenia maszynowego *k-tensor* będziemy utożsamiali z odwzorowaniem

$$X : \mathbb{N}^k \supset I \mapsto \mathbb{K}\,,$$

gdzie $\mathbb{K}$ jest pewnym ciałem, a I jest pewnym spójnym zbiorem wielowskaźników tj. zakładamy, iż

$$\forall \alpha \in \{1,\ldots,k\} : i_\alpha > 0 \wedge (i_1,..,i_\alpha,..,i_k) \in I \implies (i_1,..,i_\alpha - 1,..,i_k) \in I\,.$$

Jest to zatem po prostu wielowymiarowa tablica, której elementy są skalarami. My będziemy zawsze przyjmować $\mathbb{K} = \mathbb{R}$. Miejsce α wielowskaźnika będziemy nazywali osią. Wartość elementu tensora X pod wielowskaźnikiem $(i_1,\ldots,i_k)$ będziemy oznaczać $X_{i_1,..,i_k}$. Czasami będziemy utożsamiać zapis $X_{i_1,..,i_k}$ z odwzorowaniem X, a nie z jego wartością dla argumentu $(i_1,\ldots,i_k)$, aby podkreślić jakie argumenty przyjmuje. Jest to tożsame z częstym w matematyce oznaczaniem funkcji jako $f(x)$ podczas, gdy funkcją jest f, a $f(x)$ to jej wartość dla argumentu x.
Dla tensora $X : \mathbb{N}^k \supset I \mapsto \mathbb{R}$ definiujemy tzw. operator kształtu (z ang. *shape operator*) ш

$$\text{ш}(X) := (\text{ш}_1(X),\ldots,\text{ш}_k(X))\,, \quad \text{ш}_\alpha(X) := \text{liczba elementów } X \text{ wzdłuż osi } \alpha.$$

Operator kształtu zwraca wektor liczb naturalnych zawierający uogólnienie liczb kolumn/wierszy dla macierzy. W szczególności dla tablicy jednowymiarowej (tj. 1-tensora) T zawierającej n elementów będziemy stosowali zapis $\text{ш}(T) = (n,)$.
Zbiór wszystkich tensorów danego kształtu $\text{ш} = (n_1,\ldots,n_k)$ o elementach rzeczywistych będziemy oznaczać przez $\mathbb{R}^{n_1\times..\times n_k} \equiv \mathbb{R}^{\text{ш}}$. Odwzorowanie F o argumentach i wartościach tensorowych zdefiniujemy jako $F : \mathbb{R}^{\text{ш}_1} \mapsto \mathbb{R}^{\text{ш}_2}$. Wówczas wartość tego odwzorowania dla argumentu X $(\text{ш}(X) = \text{ш}_1)$ tj. $F(X)$ jest tensorem $(\text{ш}(F(X)) = \text{ш}_2)$. Ponadto będziemy utożsamiać zapis $(F(X))_{i_1,..,i_k}$ z $F_{i_1,..,i_k}(X)$.
W wyrażeniach przypisania postaci

$$T_{x_1,..,x_s} = X_{i_1,..,i_p} Y_{j_1,..,j_q} Z_{k_1,..,k_r} \cdots$$

stosujemy (uproszczoną) konwencję sumacyjną Einsteina tj. dla każdego indeksu po prawej stronie, który nie występuje po lewej należy przed całe wyrażenie po prawej stronie postawić znak sumy po tym indeksie (tzw. *indeks sumacyjny*), przy czym dla zwiększenia przejrzystości zapisu nie piszemy znaków sumy explicite. Oznacza to w szczególności, iż możemy łatwo definiować nowe tensory jako odpowiednie iloczyny np. iloczyn macierzy $A \in \mathbb{R}^{n\times m}, B \in \mathbb{R}^{m\times l}$ możemy zapisać jako

$$C_{ik} = A_{ij} B_{jk}\,.$$

Wyrażenie to definiuje nowy tensor $C \in \mathbb{R}^{n \times l}$, którego elementy są dane poprzez sumę iloczynów odpowiednich elementów A, B. Możemy w ten sposób zapisać również sumę wszystkich elementów tensora jako $\alpha = X_{ijk..} = \sum_{i,j,k,..} X_{ijk..}$. Dodatkowo dla funkcji $f : \mathbb{R} \mapsto \mathbb{R}$ zapis postaci $Y_{ijk..} = f(X_{ijk..})$ dla dowolnego tensora X definiuje tensor Y, którego wartości są wartościami funkcji f dla odpowiednich wartości tensora X. Dodatkowo jeśli $f : \mathbb{R} \mapsto \mathbb{R}$, a $F : \mathbb{R}^{ш_1} \mapsto \mathbb{R}^{ш_2}$ to przez zapis $Y = (f \circ F)(X)$ będziemy rozumieć

$$Y_{ijk..} = f(F_{ijk..}(X)) \,.$$

Ostatnim elementem notacji, który będzie nam potrzebny jest tzw. operator zakresu (z ang. *range operator*). Przez zapis

$$X_{i_1:n_1:d_1, i_2:n_2:d_2, .., i_k:n_k:d_k}$$

będziemy rozumieć k-tensor będący podtensorem X powstałym przez wybranie z osi α tensora X n_α elementów rozpoczynając od i_α i biorąc co d_α-ty element. Wartości tego podtensora będziemy oznaczać jako $[X_{i_1:n_1:d_1, .., i_k:n_k:d_k}]_{j_1, .., j_k}$ Jeśli dla danej osi wybieramy cały zakres to będziemy pisali sam znak „:". Jeśli z kolei któryś z d_α będzie wynosił 1 to będziemy pomijać go w zapisie pisząc tylko $i : n$.

4.2 Neuron

Podstawowym elementem każdej sieci neuronowej jest pojedynczy neuron, który możemy traktować jako odwzorowanie $y : \mathbb{R}^n \mapsto \mathbb{R}$ będące złożeniem pewnego odwzorowania nieliniowego $\sigma : \mathbb{R} \mapsto \mathbb{R}$ z odwzorowaniem afinicznym

$$\begin{aligned} &y : \mathbb{R}^n \mapsto \mathbb{R} \\ &y = \sigma\left(W_i X_i + b\right) \end{aligned} \,.$$

W praktyce jako funkcji aktywacji σ (*activation function*) najczęściej używamy funkcji ReLU, GELU lub funkcji sigmoidalnych:

$$\mathrm{ReLU}(x) := \max(0, x) \,, \quad \mathrm{GELU}(x) := x\Phi(x) \,,$$

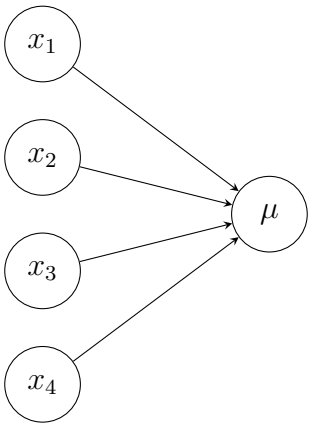
gdzie $\Phi(x)$ to dystrybuanta standardowego rozkładu normalnego. Pojedyncze neurony są następnie łączone w sieci w określony sposób tworząc daną architekturę sieci neuronowej.

4.3 Sieci MLP

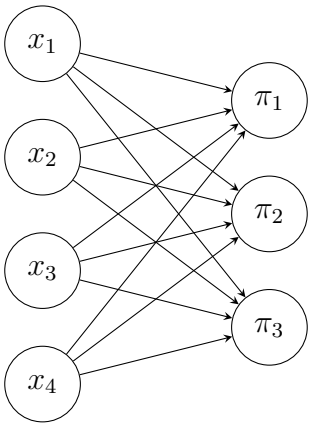
Opis sieci neuronowych zaczniemy od architektury MLP (z ang. *Multilayer Perceptron*). Sieć MLP składa się z równoległych warstw neuronów, przy czym połączenia

występują tylko między neuronami w sąsiednich warstwach i nie ma połączeń między neuronami w obrębie jednej warstwy. Pierwszą warstwę sieci nazywamy warstwą wejściową (z ang. *input layer*), ostatnią – warstwą wyjściową (z ang. *output layer*), a pozostałe nazywamy warstwami ukrytymi (z ang. *hidden layers*).

Zauważmy, iż opisane wcześniej modele regresji liniowej i wieloklasowej regresji logistycznej są przykładamy najprostszych sieci MLP bez żadnych warstw ukrytych. Ich graficzne reprezentacje jako sieci MLP zamieszczono na Rysunku 1a i 1b.



(a) Graficzna reprezentacja regresji liniowej jako najprostszej sieci MLP



(b) Graficzna reprezentacja wieloklasowej regresji logistycznej jako najprostszej sieci MLP

Zauważmy, iż wyjściem sieci są parametry docelowego rozkładu prawdopodobieństwa nad obserwacjami tj. odpowiednio wartość oczekiwana μ w przypadku regresji liniowej i prawdopodobieństwa π_i każdej z klas rozkładu kategorialnego w przypadku regresji logistycznej. W przypadku regresji liniowej funkcja aktywacji neuronu w warstwie wyjściowej to po prostu funkcja identycznościowa, natomiast w przypadku regresji logistycznej jest to funkcja soft-max.

Przejdźmy teraz do matematycznego opisu architektury MLP. Załóżmy, iż sieć składa się z $H+2$ warstw (H warstw ukrytych) o liczbach neuronów $N_{in} = N_0$ (warstwa wejściowa), $N_1, \ldots, N_H$ (warstwy ukryte), $N_{out} = N_{H+1}$ (warstwa wyjściowa). Jeśli $X \in \mathbb{R}^{N_{h-1}}$ jest wejściem h-tej warstwy ukrytej to jej wyjście $Y^h \in \mathbb{R}^{N_h}$ jest dane przez

$$Y_i^h = \sigma\left(W_{ij}^h X_j + B_i^h\right) \Longleftrightarrow Y = \left(\sigma \circ L^h\right)(X),$$

gdzie W_{ij}^h jest macierzą wag w h-tej warstwie ukrytej, a B_i^h jest wektorem obciążeń (z ang. *bias*) w tej warstwie (indeks górny nie jest potęgą tylko numeruje warstwy). Zdefiniowaliśmy tutaj odwzorowanie

$$\begin{aligned} &L^h : \mathbb{R}^{N_{h-1}} \mapsto \mathbb{R}^{N_h} \\ &L_i^h(X) = W_{ij}^h X_j + B_i^h \end{aligned} .$$

Bez straty ogólności zakładamy, iż wyjście warstwy wyjściowej jest po prostu obciążoną kombinacją liniową wejść bez żadnej funkcji nieliniowej. Wyjście z sieci MLP jest zatem dane przez następujące złożenie

$$\begin{aligned} &F : \mathbb{R}^{N_{in}} \mapsto \mathbb{R}^{N_{out}} \\ &F(X) = \left(L^{H+1} \circ \sigma \circ L^H \circ \cdots \circ \sigma \circ L^2 \circ \sigma \circ L^1\right)(X) \end{aligned} .$$

W takiej postaci sieć MLP przetwarza pojedynczo każdy wektor X_j należący do zbioru danych $\chi = \{(Y_i^k, X_j^k) \mid k \in \{1, \ldots, D\}\}$. Możemy jednak rozszerzyć odwzorowanie F w taki sposób, aby sieć przetwarzała od razu tensor X_{jk} zawierający wszystkie przykłady ze zbioru uczącego

$$\begin{aligned} &F : \mathbb{R}^{N_{in} \times D} \mapsto \mathbb{R}^{N_{out} \times D} \\ &F(X) = \left(L^{H+1} \circ \sigma \circ L^H \circ \cdots \circ \sigma \circ L^2 \circ \sigma \circ L^1\right)(X) \end{aligned}$$

$$.$$

$$\begin{aligned} &L^h : \mathbb{R}^{N_{h-1} \times D} \mapsto \mathbb{R}^{N_h \times D} \\ &L_{ik}^h(X) = W_{ij}^h X_{jk} + B_i^h I_k \end{aligned}$$

gdzie $\mathbb{1}_I = \{1\}$.

Wyjście z sieci neuronowej będziemy wykorzystywać przy tworzeniu modeli statystycznych w taki sam sposób jak w modelach płytkich (regresja liniowa, regresja softmax) używaliśmy kombinacji liniowej wektora obserwacji z wektorem wag $\mathbf{w}^T\mathbf{x}$. Uczenie parametrów W_{ij}^h, B_i^h będzie zatem polegało na znalezieniu ich estymaty punktowej MLE dla danego modelu statystycznego. Będziemy zatem minimalizować pewien funkcjonał J (funkcję kosztu) o postaci danej przez zanegowaną logarytmiczną funkcję wiarygodności. Przykładowo dla regresji użyjemy funkcjonału kwadratowego

$$J[F] = \frac{1}{2}\left(F_{ij}(X) - Y_{ij}\right)^2 ,$$

gdzie (Y_{ik}, X_{jk}) jest zbiorem danych treningowych. Natomiast w przypadku wieloklasowej regresji logistycznej jako funkcjonał przyjmiemy entropię krzyżową

$$J[F] = -T_{ik} \log \Pi_{ik} \,, \quad \Pi_{ik} = \frac{\exp\{F_{ik}(X)\}}{I_j \exp\{F_{jk}(X)\}} \,,$$

gdzie $T_{:,k}$ jest wektorem w postaci *one-hot encoding* prawidłowej klasy dla wektora zmiennych objaśniających $X_{:,k}$ ze zbioru danych treningowych.

4.4 Wsteczna propagacja błędu

W przypadku algorytmów minimalizacji takich jak spadek wzdłuż gradientu musimy znać wartości pochodnych funkcji kosztu po parametrach modelu. Wyznaczymy teraz te pochodne dla wag w sieci MLP. Niech $Y^h_{ab} = \sigma(L^h_{ab})$, wówczas $L^h_{ab} = W^h_{ac} Y^{h-1}_{cb} + B^h_a I_b$, przy czym zakładamy $Y^0 = X$. Mamy odpowiednio

$$\begin{aligned}
\frac{\partial L^h_{ab}}{\partial W^h_{ef}} &= Y^{h-1}_{cb} \delta_{ae} \delta_{cf} = Y^{h-1}_{fb} \delta_{ae} \\
\frac{\partial L^h_{ab}}{\partial Y^{h-1}_{ef}} &= W^h_{ac} \delta_{ce} \delta_{fb} = W^h_{ae} \delta_{fb} \qquad . \\
\frac{\partial Y^h_{ab}}{\partial L^h_{cd}} &= \sigma'(L^h_{ab}) \delta_{ac} \delta_{bd}
\end{aligned}$$

Jednocześnie zauważmy, że

$$\frac{\partial J}{\partial W^h_{ab}} = \left(\frac{\partial J}{\partial F_{cd}} \frac{\partial F_{cd}}{\partial L^{H+1}_{ef}} \right) \left(\frac{\partial L^{H+1}_{ef}}{\partial Y^H_{ij}} \frac{\partial Y^H_{ij}}{\partial L^H_{kl}} \right) \left(\frac{\partial L^H_{kl}}{\partial Y^{H-1}_{mn}} \frac{\partial Y^{H-1}_{mn}}{\partial L^{H-1}_{pq}} \right) \left(\cdots \right) \frac{\partial L^h_{xy}}{\partial W^h_{ab}} \,.$$

Pochodne możemy obliczać zatem iteracyjnie jako

$$\frac{\partial J}{\partial W^h_{ab}} = \Delta^h_{ij} \frac{\partial L^h_{ij}}{\partial W^h_{ab}} = \Delta^h_{ij} Y^{h-1}_{bj} \delta_{ai} = \boxed{\Delta^h_{aj} Y^{h-1}_{bj}} \,,$$

gdzie Δ^h_{ij} wyznaczamy iteracyjnie jako

$$\begin{aligned}
\Delta^{H+1}_{ij} &= \frac{\partial J}{\partial F_{kl}} \frac{\partial F_{kl}}{\partial L^{H+1}_{ij}} = \frac{\partial J}{\partial F_{kl}} \delta_{ik} \delta_{jl} = \boxed{\frac{\partial J}{\partial F_{ij}}} \\
\Delta^h_{ij} &= \Delta^{h+1}_{kl} \frac{\partial L^{h+1}_{kl}}{\partial Y^h_{mn}} \frac{\partial Y^h_{mn}}{\partial L^h_{ij}} = \Delta^{h+1}_{kl} W^{h+1}_{km} \delta_{ln} \sigma'(L^h_{mn}) \delta_{mi} \delta_{nj} = \boxed{\Delta^{h+1}_{kj} W^{h+1}_{ki} \sigma'(L^h_{ij})}
\end{aligned}$$

Zauważmy również, iż wzór na pochodną funkcji kosztu J po parametrach B_a^h możemy od razu zapisać jako

$$\frac{\partial J}{\partial B_a^h} = \Delta_{aj}^h I_j \,.$$

Możemy zatem zapisać algorytm obliczania pochodnych funkcji kosztu po parametrach sieci neuronowej zwany algorytmem wstecznej propagacji błędu (z ang. *error backpropagation*)

Algorytm wstecznej propagacji błędu

1. Dla zbioru danych (Y_{pq}, X_{rq}) dokonaj propagacji naprzód (z ang *forward propagation*) sieci MLP i zapamiętaj wartości Y^{h-1} oraz L^h.

2. Wyznacz rekurencyjnie i zapamiętaj wartości Δ_{ij}^h korzystając ze wzorów (etap *backpropagation*)

$$\boxed{\Delta_{ij}^{H+1} = \frac{\partial J}{\partial F_{ij}}\,, \quad \Delta_{ij}^h = \Delta_{kj}^{h+1} W_{ki}^{h+1} \sigma'(L_{ij}^h)}\,.$$

3. Wyznacz wartości pochodnych funkcji kosztu po parametrach sieci korzystając z

$$\boxed{\frac{\partial J}{\partial W_{ab}^h} = \Delta_{aj}^h Y_{bj}^{h-1}\,, \quad \frac{\partial J}{\partial B_a^h} = \Delta_{aj}^h I_j}\,,$$

pamiętając, iż $Y^0 = X$.

Obliczmy jeszcze pochodne $\partial J/\partial F_{ab}$ dla problemów regresji i klasyfikacji. W przypadku regresji sprawa jest prosta

$$J = \frac{1}{2}\left(F_{ij} - Y_{ij}\right)^2$$
$$\frac{\partial J}{\partial F_{ab}} = \left(F_{ij} - Y_{ij}\right)\delta_{ai}\delta_{bj} = F_{ab} - Y_{ab}\,.$$

W przypadku entropii krzyżowej rachunki są trochę bardziej zawiłe:

$$J = -T_{ik}\log \Pi_{ik}\,, \quad \Pi_{ik} = \frac{\exp F_{ik}}{I_j \exp F_{jk}}\,,$$
$$\frac{\partial J}{\partial F_{ab}} = \frac{\partial J}{\partial \Pi_{cd}}\frac{\partial \Pi_{cd}}{\partial F_{ab}}\,,$$

dalej

$$\frac{\partial J}{\partial \Pi_{cd}} = -\frac{T_{ik}}{\Pi_{ik}}\delta_{ic}\delta_{kd} = -\frac{T_{cd}}{\Pi_{cd}}$$

oraz

$$\begin{aligned}\frac{\partial \Pi_{cd}}{\partial F_{ab}} &= \frac{\exp(F_{cd})\delta_{ac}\delta_{bd} I_j \exp(F_{jd}) - \exp(F_{cd}) I_j \exp(F_{jd})\delta_{aj}\delta_{bd}}{[I_j \exp(F_{jd})]^2} \\ &= \frac{\exp(F_{cd})\delta_{ac}\delta_{bd}}{I_j \exp(F_{jd})} - \frac{\exp(F_{cd}) I_a \exp(F_{ad})\delta_{bd}}{[I_j \exp(F_{jd})]^2}\,,\end{aligned}$$

skąd

$$\begin{aligned}\frac{\partial J}{\partial F_{ab}} &= -\frac{T_{cd}}{\Pi_{cd}} \frac{\exp(F_{cd})\delta_{ac}\delta_{bd}}{I_j \exp(F_{jd})} + \frac{T_{cd}}{\Pi_{cd}} \frac{\exp(F_{cd}) I_a \exp(F_{ad})\delta_{bd}}{[I_j \exp(F_{jd})]^2} \\ &= -\frac{T_{ab}}{\Pi_{ab}} \frac{\exp F_{ab}}{I_j \exp F_{jb}} + \frac{T_{cb}}{\Pi_{cb}} \frac{I_a \exp(F_{cb}) \exp(F_{ab})}{[I_j \exp(F_{jb})]^2} \\ &= -\frac{T_{ab}}{\Pi_{ab}} \Pi_{ab} + \frac{T_{cb}}{\Pi_{cb}} I_a \Pi_{cb} \Pi_{ab} \\ &= -T_{ab} I_{ab} + T_{cb} I_{cb} I_a \Pi_{ab}\end{aligned}$$

ale z definicji T mamy, iż $\sum_c T_{cb} I_{cb} = I_b$, gdyż dla danego przykładu b dokładne jedna klasa c jest poprawna

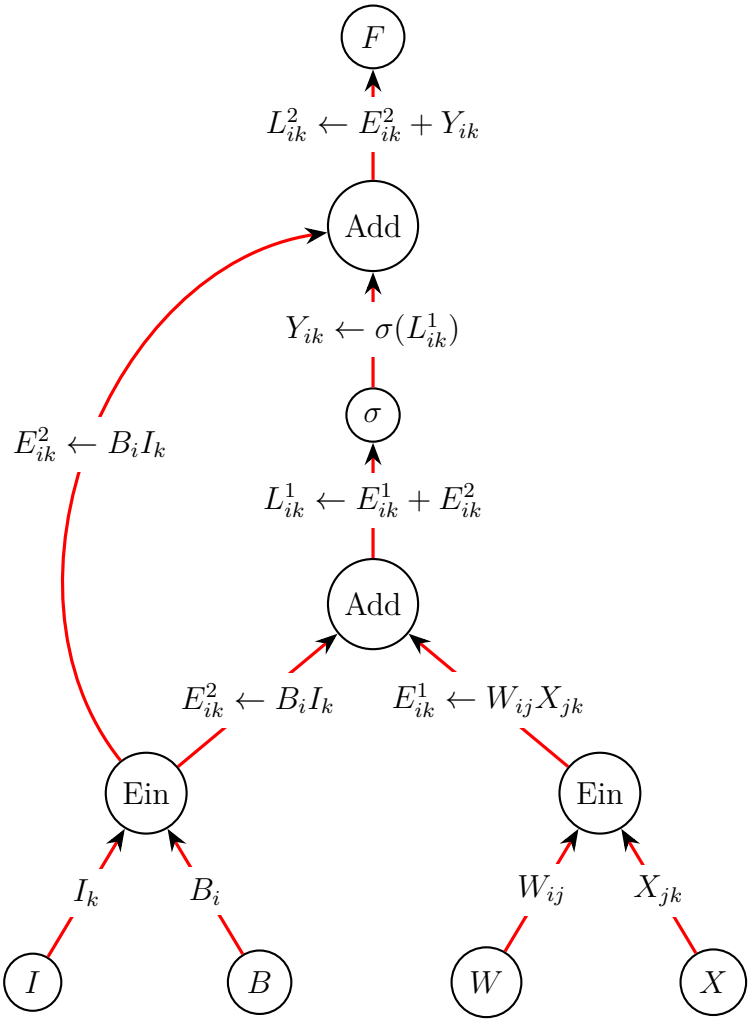
$$\frac{\partial J}{\partial F_{ab}} = -T_{ab} I_{ab} + I_a I_b \Pi_{ab} = \Pi_{ab} - T_{ab}\,.$$

4.5 Automatyczne różniczkowanie

Algorytm wstecznej propagacji błędu jest szczególnym przypadkiem automatycznego różniczkowania metodą „wstecz". Załóżmy, że mamy funkcję tensorową będącą złożeniem pewnych elementarnych funkcji tensorowych: dodawanie tensorów, mnożenie tensorów z konwencją sumacyjną (inaczej operacja redukcji), operacje element-wise postaci $Y_{ijk..} = f(X_{ijk..})$ dla $f : \mathbb{R} \mapsto \mathbb{R}$, dla których wiemy jak obliczać (symbolicznie) pochodne. Funkcję taką możemy reprezentować jako skierowany graf bez cykli (DAG) zwany grafem obliczeń (z ang. *computation graph*). Każdy liść takiego grafu reprezentuje niezależny parametr względem którego możemy zróżniczkować wyjściową funkcję. Różniczkowanie (metodą „wstecz" lub inaczej top-down) polega na przekazywaniu tzw. *ziarna* (z ang. *seed*), czyli pochodnej funkcji wyjściowej po danym wierzchołku grafu. Ziarno jest obliczane zgodnie z regułą łańcuchową i przekazywane do poprzedników danego wierzchołka. Przykładowo rozpatrzmy funkcję

$$F_{ik} = \sigma\left(W_{ij} X_{jk} + B_i I_k\right) + B_i I_k\,,$$

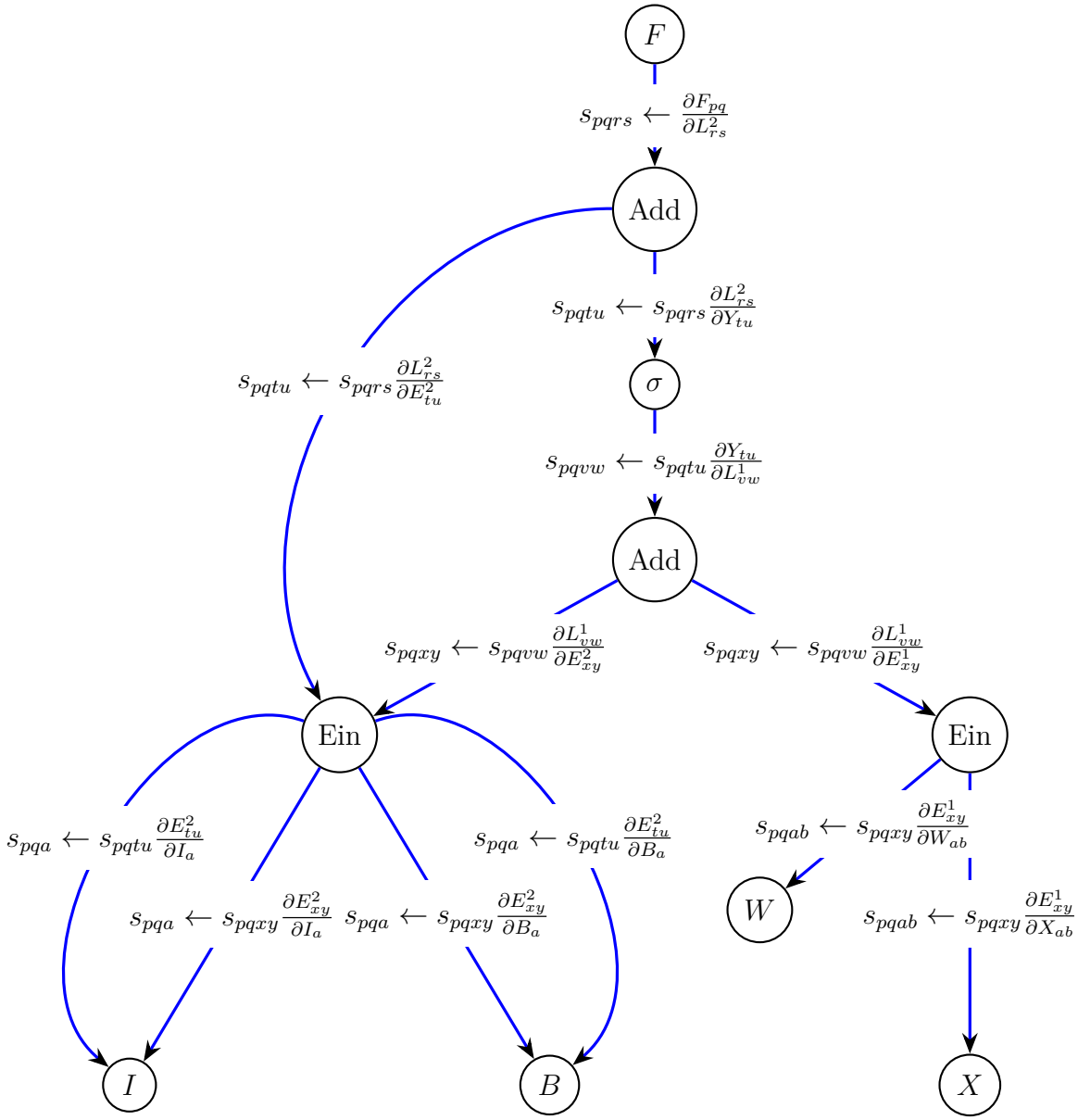
którą możemy reprezentować jako graf obliczeń.



Chcemy obliczyć pochodne postaci $\partial F_{pq}/\partial X_{ab}, \partial F_{pq}/\partial W_{ab}, \partial F_{pq}/\partial B_a, \partial F_{pq}/\partial I_a$, czyli pochodne po liściach naszego grafu obliczeń. W tym celu idąc top-down tj. zaczynając od wierzchołka F przekazujemy ziarno do każdej krawędzi wchodzącej do tego wierzchołka, przy czym ziarno obliczamy rekurencyjnie jako iloczyn (w sensie notacji sumacyjnej Einsteina) ziarna w danym wierzchołku i pochodnej wyrażenia w tym wierzchołku po wyrażeniu w wierzchołku, z którego wychodzi dana krawędź. Dochodząc do liścia inkrementujemy ziarno pamiętane w liściu o przychodzącą wartość. Otrzymana wartość ziarna w wierzchołku będzie równa pochodnej funkcji wyjściowej po tym parametrze. Rozpatrzmy przykład.

Wartość ziarna w wierzchołku przykładowo B wynosi

$$s_{pqa} = s^1_{pqa} + s^2_{pqa}\,,$$



gdzie

$$s^1_{pqa} = \frac{\partial F_{pq}}{\partial L^2_{rs}} \frac{\partial L^2_{rs}}{\partial E^2_{tu}} \frac{\partial E^2_{tu}}{\partial B_a}$$
$$s^2_{pqa} = \frac{\partial F_{pq}}{\partial L^2_{rs}} \frac{\partial L^2_{rs}}{\partial Y_{tu}} \frac{\partial Y_{tu}}{\partial L^1_{vw}} \frac{\partial L^1_{vw}}{\partial E^2_{xy}} \frac{\partial E^2_{xy}}{\partial B_a}$$

widzimy więc, że $s_{pqa} = \frac{\partial F_{pq}}{\partial B_a}$ i analogicznie dla ziaren w wierzchołkach I, X, W. Oczywiście wyrażenia na pochodne wyrażenia w wierzchołku po poprzednikach

musimy obliczyć symbolicznie i zaimplementować ręcznie dla wszystkich wykorzystywanych rodzajów wierzchołków

4.6 Metody optymalizacji numerycznej

Mając algorytm efektywnego obliczania pochodnych funkcji kosztu, funkcję minimalizujemy korzystając z jednego z algorytmów optymalizacji numerycznej pierwszego rzędu (tj. wykorzystującego jedynie pochodne 1-go rzędu). W przypadku sieci neuronowych funkcja kosztu nie jest funkcją ściśle wypukłą, więc algorytmy te nie znajdą w ogólności globalnego minimum; możemy liczyć jedynie na znalezienie minimum lokalnego.

Algorytm MBGD

1. Zainicjuj parametry W^h_{ab}, B^h_a niewielkimi wartościami losowymi.
2. Podziel dane treningowe na K rozłącznych mini-batchy $D_1, \ldots, D_K$ jednakowej wielkości.
3. Powtarzaj przez N epok:
   Dla każdego mini-batcha D:
   (a) Oblicz pochodne funkcji kosztu J po parametrach sieci korzystając z algorytmu wstecznej propagacji błędu.
   (b) Zaktualizuj wartości parametrów zgodnie z:

$$\boxed{\begin{aligned} W^h_{ab} &\leftarrow W^h_{ab} - \frac{\epsilon}{|D|}\frac{\partial J}{\partial W^h_{ab}} \\ B^h_a &\leftarrow B^h_a - \frac{\epsilon}{|D|}\frac{\partial J}{\partial B^h_a} \end{aligned}} \quad ,$$

   gdzie ϵ jest hiperparametrem zwanym stałą uczącą (z ang. *learning rate*).

4.7 Regularyzacja

Sieci neuronowe są bardzo *pojemnymi* modelami, przez co są bardzo podatne na overfitting i nie generalizują się dobrze poza zbiór treningowy. Bardzo ważnym elementem budowy sieci neuronowych są więc metody regularyzacji.

1. **Czynniki regularyzujące**

Analogicznie jak w przypadku prostych modeli liniowych jedną z możliwości regularyzacji jest dodanie do funkcji kosztu J czynnika regularyzującego zawierającego odpowiednią sumę norm poszczególnych parametrów sieci

$$J^* = J + \frac{\lambda}{2} \sum_{h=1}^{H+1} \sum_{i,j} \left(W_{ij}^h\right)^2 , \quad \text{regularyzacja } L^2$$

$$J^* = J + \frac{\lambda}{2} \sum_{h=1}^{H+1} \sum_{i,j} \left|W_{ij}^h\right| , \quad \text{regularyzacja } L^1$$

Różnica między regularyzacją L^2 i L^1 jest taka, że norma L^2 „zachęca" sieć do wybierania małych wag, ale nie dokładnie będących 0, natomiast L^1 zmniejsza wagi, ale w szczególności prowadzi do wag wynoszących dokładnie 0. W związku z tym waga L^1 przeprowadza selekcję cech - "wyłączy" te, które są mało ważne.

2. **Early stopping**
   Innym prostym, lecz efektywnym podejściem do regularyzacji sieci neuronowych jest procedura *early stopping*. Polega ona na zatrzymaniu procesu uczenia (optymalizacji funkcji kosztu) w momencie, w którym wartość funkcji kosztu na wydzielonym ze zbioru treningowego zbiorze walidacyjnym zaczyna rosnąć (lub nie spada przez odpowiednio długi czas). Unikamy w ten sposób przeuczenia modelu. Takie podejście wymaga jednak dość dużego zbioru danych, z których musimy wydzielić trzy zbiory: zbiór treningowy (na którym uczymy model), zbiór walidacyjny (na którym ewaluujemy model przy procedurze early stopingu) oraz zbiór testowy (na którym ewaluujemy model po treningu). Okres, przez który czekamy na uzyskanie lepszego wyniku, to cierpliwość (z ang. *patience*). Im mniejsze, tym mocniejszy jest ten rodzaj regularyzacji, ale trzeba z tym uważać, bo łatwo jest przesadzić i zbyt szybko przerywać trening. Niektóre implementacje uwzględniają tzw. *grace period*, czyli gwarantowaną minimalną liczbę epok, przez którą będziemy trenować sieć, niezależnie od wybranej cierpliwości.

3. **Dropout**
   Bardzo popularną i szeroko stosowaną metodą regularyzacji sieci neuronowych jest dropout polegający na tym, iż w procesie treningu ze z góry zadanym prawdopodobieństwem „wyłączamy" wyjście neuronu (tj. dla każdej aktualizacji wartości parametrów sieci, w algorytmie wstecznej propagacji błędu mnożymy wyjście Y_{ab}^h przez tensor P_{ab}^h, którego każdy element może być równy 0 z p-stwem p lub 1 z p-stwem $1-p$). Mechanizmu dropout używamy tylko w procesie treningu. W przypadku inferencji wyjście neuronu

jest z kolei mnożone przez następujący stosunek

$$1 - \frac{\#\text{wyłączeń neuronu}}{\#\text{liczba aktualizacji parametrów sieci}}.$$

Typowo usuwa się dość sporo neuronów w warstwach ukrytych, np. 50%. Dzięki dropoutowi można używać z nim dość dużych stałych uczących oraz współczynników pędu, a także trenować znacznie większe i głębsze sieci. Okazuje się, iż w praktyce takie duże, mocno trenowane sieci z regularyzacją uczą się lepiej niż mniejsze, które by tej regularyzacji nie potrzebowały.

4. **Połączenia rezydualne**
   Sporym problemem w procesie uczenia sieci neuronowych jest fakt, iż funkcja kosztu J nie jest wypukła, a dodatkowo często nie jest dość regularna tj. pojawiają się tzw. „poszarpania" (z ang. *shattered*), które powodują problemy z obliczaniem gradientu. Metodą, która pozawala do pewnego stopnia wyeliminować ten problem jest wprowadzenie w sieci połączeń rezydualnych tj. warstw, których wyjście jest dane przez

$$\begin{aligned} &Y^h : \mathbb{R}^{N\times D} \mapsto \mathbb{R}^{N\times D} \\ &Y^h_{ik}(X) = \sigma\left(W^h_{ij}X_{jk} + B^h_i I_k\right) + X_{ik} \end{aligned}.$$

   Oczywiście wprowadzenie takiej warstwy oznacza, iż liczba neuronów w następnie warstwie musi być taka sama.

5. **Data augmentation**
   Metoda data augmentation polega na dodaniu do zbioru treningowego replik przykładów z tego zbioru poddanych pewnym transformacjom względem których oczekujemy, iż odpowiedzi sieci będą niezmienione.

4.8 Sieci CNN

Sieci konwolucyjne (CNN) to architektura sieci neuronowych przystosowana do przetwarzania danych posiadających pewne zależności "przestrzenne", przez co rozumiemy, iż jeśli X_{ijk} jest tensorem wejściowym to zależności występują między wartościami sąsiednimi $X_{i,j,k}, X_{i+a,j+b,k+c}$ dla niewielkich a, b, c. Przykładem takich danych są oczywiście obrazy. Przetwarzanie takich danych przez sieć MLP byłoby niezwykle nieefektywne, gdyż połączenia każdy z każdym oznaczałyby iż przykładowo dla obrazu w skali szarości o rozdzielczości 512×512 mielibyśmy rzędu 10^9 parametrów dla jednej warstwy ukrytej. Dodatkowo sieć MLP szukałaby powiązań między odległymi pikselami, pomiędzy którymi takich zależności nie ma. Sieć CNN robi to znacznie efektywniej wykorzystując warstwy konwolucyjne. Jeśli X_{ijk}

jest tensorem wejściowym do warstwy konwolucyjnej (np. pojedynczym obrazem reprezentowanym przez 3-tensor o kształcie $(512, 512, 3)$) to wyjście warstwy C_{abc} jest opisane przez funkcję (**uwaga** zakładamy, iż indeksy są liczbami naturalnymi **wliczając** 0)

$$C : \mathbb{R}^{h_{in} \times w_{in} \times c_{in}} \mapsto \mathbb{R}^{h_{out} \times w_{out} \times c_{out}}$$
$$C_{abc}(X) = [X_{sa:h:d, sb:w:d, :}]_{ijk} W_{ijkc} + B_c I_{ab} \,,$$

gdzie $\text{ш}(W) = (h, w, c_{\text{in}}, c_{\text{out}})$ oraz $c_{\text{in}} = \text{ш}_3(X)$. Tensor W_{ijkc} dla ustalonego c nazywany *filtrem konwolucyjnym*, natomiast cały tensor W – *mapą konwolucyjną* (formalnie nie jest to operacja konwolucji tylko korelacja skrośna). Warstwę konwolucyjną charakteryzuje szereg parametrów:

- liczba kanałów wejściowych $c_{\text{in}} = \text{ш}_3(X)$ (oś 3. nazywamy również wymiarem głębi)
- liczba kanałów wyjściowych równa liczbie filtrów c_{out}
- wysokość i szerokość tzw. *pola odbiorczego* $h \times w$
- *stride* s, zwykle jednakowy dla wymiaru góra-dół i lewo-prawo chociaż w ogólności możemy przyjąć różne wartości s_h, s_w. Parametr ten określa o ile przesuwamy pole odbiorcze w każdym kroku.
- *dilation* d, ponownie zwykle jednakowy dla obu wymiarów chociaż w ogólności możemy przyjąć różne wartości d_h, d_w

Zależność między kształtami wejściowym $(h_{\text{in}}, w_{\text{in}}, c_{\text{in}})$ i wyjściowym $(h_{\text{out}}, w_{\text{out}}, c_{\text{out}})$ dla dowolnych parametrów $h, w, s, d, c_{\text{out}}$ można zapisać jako

$$h_{\text{out}} = \left\lfloor \frac{h_{\text{in}} - d_h(h-1) - 1}{s_h} + 1 \right\rfloor , \quad w_{\text{out}} = \left\lfloor \frac{w_{\text{in}} - d_w(w-1) - 1}{s_w} + 1 \right\rfloor .$$

Do algorytmu wstecznej propagacji błędu potrzebujemy znać pochodne wyjścia warstwy konwolucyjnej po parametrach sieci. Oczywiście pochodna po parametrach tej warstwy jest oczywista do obliczenia, podamy więc jedynie pochodną elementu podtensora X po pewnym elemencie tego tensora

$$\frac{\partial [X_{a_1:n_1:d_1, .., a_k:n_k:d_k}]_{i_1, .., i_k}}{\partial X_{x_1, .., x_k}} = \prod_{\alpha=1}^{k} \delta_{a_\alpha + i_\alpha d_\alpha, x_\alpha} \,.$$

Analogicznie jak w przypadku sieci MLP możemy rozszerzyć definicję funkcji C definiującej warstwę konwolucyjną w taki sposób, aby przetwarzała od razu cały

4-tensor zawierający pojedyncze przykłady będące 3-tensorami (np. kolorowe obrazy)

$$C : \mathbb{R}^{h_{in} \times w_{in} \times c_{in} \times D} \mapsto \mathbb{R}^{h_{out} \times w_{out} \times c_{out} \times D}$$
$$C_{abce}(X) = [X_{sa:h:d,sb:w:d,:,:}]_{ijke} W_{ijkc} + B_c I_{abe}$$

W powyższej definicji stosujemy tzw. zapis *batch-last*, tzn. oś (wymiar) związany z różnymi przykładami w zbiorze uczącym (*batch dimension*) jest ostatnim wymiarem tensora.

Zauważmy, że dany filtr konwolucyjny (ustalone c) ma te same wartości (po treningu) niezależnie od położenia pola odbiorczego w wolumenie wejściowym. Filtr taki w takim razie wykrywa konkretną (ale nie zakodowaną ręcznie tylko wytrenowaną w procesie uczenia sieci) cechę.

Drugim rodzajem warstw typowo wykorzystywanym w sieciach CNN są tzw. *warstwy poolingowe* (zbierające). Są to warstwy bezparametrowe, które z pola odbiorczego wyciągają pojedynczą liczbę. Najczęściej stosowane są warstwy typu max pooling wyciągające wartość największą lub average pooling, które biorą średnią z wartości w polu odbiorczym. Zwykle parametry d, s dobiera się w taki sposób, aby kolejne pola odbiorcze nie zachodziły na siebie, choć nie jest to żadna reguła. Warstwy poolingowe nie zmieniają wymiaru głębi, ale pozwalają zredukować wymiar przestrzenny.

4.9 Mechanizmy atencji

4.10 Sieci Transformer

4.11 Generatywne modele dyfuzyjne