# 1 Wnioskowanie Bayesowskie

Typowym sposobem połączenia wyników matematycznej teorii prawdopodobieństwa z eksperymentami jest interpretacja częstościowa, która stwierdza iż prawdopodobieństwo zdarzenia $A$ to częstość realizacji $A$ w dużej liczbie prób. Dla wielu zdarzeń pojęcie ***dużej liczby prób*** nie jest jednak dobrze zdefiniowane. Wówczas interpretacja Bayesowska prawdopodobieństwa jako miary niepewności co do rezultatu określonego zdarzenia jest bardziej naturalna. Wnioskowanie Bayesowskie opiera się na twierdzeniu Bayesa

$$\underbrace{p(\boldsymbol{x} \mid \boldsymbol{y})}_{\text{posterior}} = \frac{\overbrace{p(\boldsymbol{y} \mid \boldsymbol{x})}^{\text{likelihood}}\overbrace{p(\boldsymbol{x})}^{\text{prior}}}{\underbrace{\int p(\boldsymbol{y} \mid \boldsymbol{x})p(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x}}_{\text{evidence}}}$$

oraz dwóch regułach znanych z rachunku prawdopodobieństwa: ***sum rule*** i ***product rule***.

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}, \boldsymbol{y})\,\mathrm{d}\boldsymbol{y}\ , \quad p(\boldsymbol{x}, \boldsymbol{y}) = p(\boldsymbol{x} \mid \boldsymbol{y})p(\boldsymbol{y})\,.$$

Typowe zastosowanie wnioskowania Bayesowskie przebiega następująco: mamy pewien zbiór obserwacji $\mathcal{X}$, o których zakładamy, iż pochodzą z pewnego rozkładu prawdopodobieństwa z parametrami $\boldsymbol{\theta}$, tj. znamy wiarygodność $p(\mathcal{X} \mid \boldsymbol{\theta})$. Zakładamy dodatkowo pewien prior nad parametrami $p(\boldsymbol{\theta})$. Następnie korzystając z twierdzenia Bayesa wyznaczamy rozkład a posteriori nad parametrami modelu

$$p(\boldsymbol{\theta} \mid \mathcal{X}) = \frac{p(\mathcal{X} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})}{\int p(\mathcal{X} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}}.$$

Rozkład a posteriori podsumowuje całą naszą wiedzę o estymowanym parametrze (z perspektywy wnioskowania Bayesowskiego). Korzystając z tego rozkładu możemy:

- Wyznaczyć estymatę punktową (tzw. estymatę ***maximum a posteriori*** (MAP)) parametru $\boldsymbol{\theta}$ jako
$$\boxed{\boldsymbol{\theta}_{\mathrm{MAP}} = \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta} \mid \mathcal{X})} \quad .$$
- Wyznaczyć ***przedział wiarygodności*** (ang. ***credible interval***, nie mylić z przedziałem ufności) parametru $\boldsymbol{\theta}$ jako
$$C_\alpha(\boldsymbol{\theta}; \mathcal{X}) = (\underline{\boldsymbol{\theta}}, \overline{\boldsymbol{\theta}})\,,$$
gdzie
$$\int_{\underline{\boldsymbol{\theta}}}^{\overline{\boldsymbol{\theta}}} p(\boldsymbol{\theta} \mid \mathcal{X})\,\mathrm{d}\boldsymbol{\theta} = 1 - \alpha\,.$$
- Skonstruować ***rozkład predyktywny*** (ang. ***posteriori predictive distribution***) dla nowych obserwacji
$$\boxed{p(\boldsymbol{x} \mid \mathcal{X}) = \int p(\boldsymbol{x} \mid \boldsymbol{\theta})p(\boldsymbol{\theta} \mid \mathcal{X})\,\mathrm{d}\boldsymbol{\theta}} \quad .$$
- Możemy wyznaczyć wartość parametru $\boldsymbol{\theta}$, która minimalizuje oczekiwaną stratę dla pewnej funkcji kosztu
$$\begin{aligned}\boldsymbol{\theta}^* &= \arg\min_{\boldsymbol{\theta}'} \mathbb{E}_{\boldsymbol{\theta} \sim p(\boldsymbol{\theta} \mid \mathcal{X})}\left[L(\boldsymbol{\theta}', \boldsymbol{\theta})\right] \\ &= \arg\min_{\boldsymbol{\theta}'} \int L(\boldsymbol{\theta}, \boldsymbol{\theta}')p(\boldsymbol{\theta} \mid \mathcal{X})\,\mathrm{d}\boldsymbol{\theta}\ .\end{aligned}$$

## 1.1 Rozkład normalny

Wielowymiarowy rozkład normalny o średniej $\boldsymbol{\mu} \in \mathbb{R}^n$ i symetrycznej, nieujemnie określonej macierzy kowariancji $\boldsymbol{\Sigma} \in \mathbb{R}^{n \times n}$ to rozkład o gęstości danej przez

$$p(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu})^\intercal \boldsymbol{\Sigma}^{-1}(\boldsymbol{x} - \boldsymbol{\mu})\right].$$

Macierz $\boldsymbol{\Lambda} = \boldsymbol{\Sigma}^{-1}$ nazywamy macierzą precyzji. Będziemy często korzystać z następującej własności rozkładów normalnych: jeśli zmienne $\boldsymbol{x} \in \mathbb{R}^n$ i $\boldsymbol{y} \in \mathbb{R}^m$ mają łącznie wielowymiarowy rozkład normalny

$$\begin{bmatrix}\boldsymbol{x} \\ \boldsymbol{y}\end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_x \\ \boldsymbol{\mu}_y\end{bmatrix}, \begin{bmatrix}\boldsymbol{\Sigma}_{xx} & \boldsymbol{\Sigma}_{xy} \\ \boldsymbol{\Sigma}_{yx} & \boldsymbol{\Sigma}_{yy}\end{bmatrix}\right),$$

to rozkłady brzegowe są rozkładami normalnymi

$$\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_{xx})\,, \quad \boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_{yy})$$

oraz rozkłady warunkowe są rozkładami normalnymi

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

gdzie

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

## 1.2 Liniowe modele Gaussowskie

Jednym z najprostszych modeli, do którego można zastosować wnioskowanie Bayesowskie jest tzw. liniowy model Gaussowski. Zakładamy, iż mamy obserwację $\boldsymbol{y}$, która pochodzi z rozkładu normalnego

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

gdzie $\boldsymbol{x}$ jest pewną zmienną ukrytą (ang. ***hidden variable***), natomiast $\boldsymbol{A}$, $\boldsymbol{b}$ i $\boldsymbol{\Sigma}_y$ są znane dokładnie.

Dodatkowo przyjmijmy ***prior sprzężony do wiarygodności*** dla parametru $\boldsymbol{x}$ będący rozkładem normalnym o znanych parametrach $\boldsymbol{\mu}_x$, $\boldsymbol{\Sigma}_x$

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

Wówczas rozkład a posteriori również będzie rozkładem normalnym o parametrach

$$\begin{aligned}\boldsymbol{\mu}_{x|y} &= \boldsymbol{\Sigma}_{x|y}\left(\boldsymbol{A}^\intercal\boldsymbol{\Sigma}_y^{-1}(\boldsymbol{y}-\boldsymbol{b}) + \boldsymbol{\Sigma}_x^{-1}\boldsymbol{\mu}_x\right)\\ \boldsymbol{\Sigma}_{x|y} &= \left(\boldsymbol{\Sigma}_x^{-1} + \boldsymbol{A}^\intercal\boldsymbol{\Sigma}_y^{-1}\boldsymbol{A}\right)^{-1}\end{aligned}\,.$$

# 2 Regresja liniowa

Załóżmy, że modelujemy obserwacje postaci $(y, \boldsymbol{x})$, gdzie $y$ to skalar zwany ***zmienną objaśnianą***, natomiast $\boldsymbol{x}$ to wektor cech lub inaczej ***zmiennych objaśniających***. Zakładamy, że zmienna objaśniana zależy liniowo od zmiennych objaśniających oraz, iż zależność ta jest obarczona pewnym błędem losowym. Model ten ma więc postać

$$\boxed{y(\boldsymbol{x}) = \phi(\boldsymbol{x}; \boldsymbol{w}, b) + \epsilon = \boldsymbol{w}^\intercal\boldsymbol{x} + b + \epsilon\,,}$$

gdzie $y(\boldsymbol{x})$ jest wartością zmiennej losowej obserwowaną dla znanego dokładnie wektora cech $\boldsymbol{x}$ (wektor cech ***nie*** jest zmienną losową), natomiast $\boldsymbol{w}$ i $b$ to parametry modelu $\phi$, o których chcemy wnioskować. Model ten jest typowym przykładem ***regresji liniowej***. Parametryzację regresji liniowej można uprościć przyjmując, że do wektora cech dokładamy stałą wartość 1, a parametr $b$ włączamy do $\boldsymbol{w}$

$$\boldsymbol{x}^\intercal \leftarrow \begin{bmatrix}\boldsymbol{x}^\intercal & 1\end{bmatrix}, \quad \boldsymbol{w}^\intercal \leftarrow \begin{bmatrix}\boldsymbol{w}^\intercal & b\end{bmatrix}.$$

W zależności od przyjętego rozkładu zmiennej opisującej błąd losowy $\epsilon$ otrzymujemy różne modele, np.:

- regresja liniowa (ang. ***linear regression***), $\epsilon \sim \mathcal{N}(0, \sigma^2)$ dla znanego $\sigma^2$,
- regresja odporna (ang. ***robust regression***), $\epsilon \sim \text{Laplace}(0, \lambda)$ dla znanego $\lambda$

$$\text{Laplace}(x; \mu, \lambda) = \frac{1}{2b}\exp\left[-\frac{|x-\mu|}{b}\right] \quad ,$$

- regresja kwantylowa (ang. ***quantile regression***), $\epsilon \sim \text{ALD}(0, \lambda, p)$ dla znanego $\lambda$ i percentyla $p$

$$\text{ALD}(x; \mu, \lambda, p) = \frac{p(1-p)}{\lambda}\cdot\begin{cases}\mathrm{e}^{-\frac{(p-1)(x-\mu)}{\lambda}} & , x \le \mu\\ \mathrm{e}^{-\frac{p(x-\mu)}{\lambda}} & , x > \mu\end{cases}.$$

Zauważmy tutaj, że założenie iż znamy skalę (tj. parametry $\sigma$, $\lambda$) błędu jest dosyć mocne – z reguły nie wiemy jak dokładnie możemy przybliżyć zmienną objaśnianą. Dokładne wnioskowanie Bayesowskie przeprowadzimy tylko dla pierwszego z powyższych modeli, gdzie błędy mają rozkład normalny. To założenie, choć sensowne, ma również swoje konsekwencja – nasz model będzie dosyć wrażliwy na obserwacje odstające (ang. ***outliers***). Zamiast rozkładu normalnego można przyjąć rozkład, w którym gęstość prawdopodobieństwa ma tzw. ciężkie ogony (ang. ***heavy tails***) – np. rozkład Laplace'a. Otrzymamy wówczas model zwany odporną regresją liniową (ang. ***robust linear regression***). Oba powyższe modele równo rozkładają masę prawdopodobieństwa błędów. Jeśli interesuje nas natomiast estymacja kwantyli warunkowych (tj. prostej która najlepiej dzieli obserwacje tak aby odpowiedni ułamek z nich znalazł się pod nią), to możemy wykorzystać asymetryczny rozkład Laplace'a. Otrzymany model nazywamy wówczas kwantylową regresją liniową (ang. ***quantile linear regression***).

Przyjmijmy obecnie, że dysponujemy zbiorem obserwacji $\mathcal{X} = \{(y_i, \boldsymbol{x}_i)\}_{i=1}^n$. Zakładamy, że obserwacje te są i.i.d. – niezależne i pochodzące z tego samego rozkładu $y_i \sim \mathcal{N}(\phi(\boldsymbol{x}_i; \boldsymbol{w}), \sigma^2)$. Jawne wnioskowanie Bayesowskie możliwe jest również dla regresji liniowej. Bardzo często jednak stosuje się prostsze podejście adekwatne również do pozostałych dwóch modeli. Otóż poszukujemy jedynie estymaty punktowe parametrów $\boldsymbol{w}$, ignorując niepewność ich oszacowania. Taką estymację punktową można wyznaczyć ***metodą największej wiarygodności*** (ang. ***maximum likelihood estimation*** (MLE))

$$\boxed{\boldsymbol{w}_{\text{MLE}} = \arg\max_{\boldsymbol{w}} p(\mathcal{X} \mid \boldsymbol{w})}\,.$$

Korzystamy wówczas jedynie z wiarygodności $p(\mathcal{X} \mid \boldsymbol{w})$. Z założenia, iż obserwacje są i.i.d. możemy łatwo wyznaczyć wiarygodność

$$\begin{aligned}p(\mathcal{X} \mid \boldsymbol{w}) &= \prod_{i=1}^n \mathcal{N}(y_i; \phi(\boldsymbol{x}_i; \boldsymbol{w}), \sigma^2)\\ &\propto \prod_{i=1}^n \exp\left(-\frac{(y_i - \phi(\boldsymbol{x}_i; \boldsymbol{w}))^2}{2\sigma^2}\right),\end{aligned}$$

gdzie pominęliśmy współczynnik, gdyż nie ma on wpływu na zagadnienie optymalizacji. W praktyce często łatwiej jest minimalizować sumę pewnych wyrażeń, niż ich iloczyn, dlatego wprowadzamy ***zanegowaną logarytmiczną funkcję wiarygodności*** (ang. ***negated log-likelihood function*** (NLL))

$$\boxed{L(\mathcal{X}, \boldsymbol{w}) = -\log p(\mathcal{X} \mid \boldsymbol{w})}\,,$$

gdzie korzystamy oczywiście z faktu, iż logarytm (naturalny) jest funkcją ściśle rosnącą. Estymatę MLE możemy wówczas znaleźć jako

$$\boxed{\boldsymbol{w}_{\mathrm{MLE}} = \arg\min_{\boldsymbol{w}} L(\mathcal{X}, \boldsymbol{w})}\,.$$

W przypadku regresji liniowej otrzymujemy

$$\boldsymbol{w}_{\mathrm{MLE}} = \arg\min_{\boldsymbol{w}} \frac{1}{2}\sum_{i=1}^{n}(y_i - \phi(\boldsymbol{x}_i; \boldsymbol{w}))^2\,.$$

Z powyższego problemu optymalizacyjnego wynika popularna nazwa tej formy regresji liniowej – ***metoda najmniejszych kwadratów*** (ang. ***Ordinary Least Squares*** (OLS)), ponieważ minimalizujemy sumę kwadratów rezyduów. W tym szczególnym przypadku postać $\boldsymbol{w}_{\mathrm{MLE}}$ można znaleźć analitycznie. Istotnie wprowadźmy

$$\boldsymbol{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}, \quad \boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1^\intercal \\ \vdots \\ \boldsymbol{x}_n^\intercal \end{bmatrix}, \quad \boldsymbol{\phi} = \boldsymbol{X}\boldsymbol{w}\,.$$

Wówczas mamy

$$\frac{\partial L}{\partial \phi_\beta} = (\phi_\beta - y_\beta)$$

oraz dalej

$$\frac{\partial L}{\partial w_\alpha} = \sum_{\beta=1}^{n} \frac{\partial L}{\partial \phi_\beta}\frac{\partial \phi_\beta}{\partial w_\alpha} = \sum_{\beta=1}^{n}(\phi_\beta - y_\beta)X_{\beta\alpha}\,.$$

Z warunków koniecznych dla minimum lokalnego mamy $\forall \alpha : \frac{\partial L}{\partial w_\alpha} = 0$, co możemy zapisać jako

$$\boldsymbol{X}^\intercal(\boldsymbol{\phi} - \boldsymbol{y}) = \boldsymbol{X}^\intercal(\boldsymbol{X}\boldsymbol{w}_{\mathrm{MLE}} - \boldsymbol{y}) = \boldsymbol{0}\,,$$

skąd

$$\boldsymbol{w}_{\mathrm{MLE}} = (\boldsymbol{X}^\intercal\boldsymbol{X})^{-1}\boldsymbol{X}^\intercal\boldsymbol{y}\,.$$

Wielkość $(\boldsymbol{X}^\intercal\boldsymbol{X})^{-1}\boldsymbol{X}^\intercal$ nazywa się pseudo odwrotnością Moore'a–Penrose macierzy $\boldsymbol{X}$. Warto tutaj zaznaczyć, iż w praktyce rzadko wyznaczamy $\boldsymbol{w}_{\mathrm{MLE}}$ korzystając wprost z powyższego wzoru, ze względu na dość słabe własności numeryczne. Typowo jeśli macierz $\boldsymbol{X}$ jest niewielka, to pseudo odwrotność obliczamy korzystając z rozkładu SVD macierzy $\boldsymbol{X}$. Natomiast dla dużych macierzy nie korzystamy ze wzoru analitycznego tylko z metody iteracyjnej stochastycznego spadku wzdłuż gradientu

$$w_\alpha^{(t+1)} = w_\alpha^{(t)} - \eta\frac{\partial L}{\partial w_\alpha}\,,$$

która w tym przypadku zbiega do minimum globalnego $\boldsymbol{w}_{\mathrm{MLE}}$.

## 2.1 Regresja Bayesowska

Pokażemy teraz, jak przeprowadzić pełne wnioskowanie Bayesowskie dla modelu regresji liniowej. Założymy rozkład a priori nad parametrami $\boldsymbol{w}$ w postaci rozkładu normalnego

$$\boldsymbol{w} \sim \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)\,.$$

Wiarygodność, jak pokazaliśmy powyżej ma natomiast postać

$$\mathcal{X} \mid \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{X}\boldsymbol{w}, \sigma^2\boldsymbol{I})\,.$$

Zauważmy, że jest to instancja modelu liniowego, więc korzystając z wyprowadzonych wzorów wiemy, iż

$$\boldsymbol{w} \mid \mathcal{X} \sim \mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)\,,$$

gdzie

$$\begin{aligned} \boldsymbol{\mu}_n &= \boldsymbol{\Sigma}_n\left(\frac{1}{\sigma^2}\boldsymbol{X}^\intercal\boldsymbol{y} + \boldsymbol{\Sigma}_0^{-1}\boldsymbol{\mu}_0\right) \\ \boldsymbol{\Sigma}_n &= \left(\boldsymbol{\Sigma}_0^{-1} + \frac{1}{\sigma^2}\boldsymbol{X}^\intercal\boldsymbol{X}\right)^{-1} \end{aligned}\,.$$

W powyższych wzorach nazwy parametrów nie są przypadkowe. Przed zaobserwowaniem danych mamy rozkład a priori $\boldsymbol{w} \mid \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$. Po zaobserwowaniu $n$ przykładów aktualizujemy nasze przekonania w postaci rozkładu a posteriori $\boldsymbol{w} \mid \mathcal{X} \sim \mathcal{N}(\boldsymbol{\mu}_n, \boldsymbol{\Sigma}_n)$.

Znając rozkład a posteriori nad $\boldsymbol{w}$ możemy również wyznaczyć rozkład predykcyjny nad zmienną objaśnianą $y$ dla ustalonego wektora cech $\boldsymbol{x}$. Formalnie interesuje nas rozkład $p(y \mid \mathcal{X})$. Rozkład ten możemy opisać w następujący sposób

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

gdzie interesuje nas rozkład

$$p(y \mid \mathcal{X}) = \int p(y \mid \boldsymbol{w})p(\boldsymbol{w} \mid \mathcal{X})\,\mathrm{d}\boldsymbol{w}\,.$$

Można pokazać, iż rozkład ten jest rozkładem normalnym

$$y \mid \mathcal{X} \sim \mathcal{N}(\mu_y, \sigma_y^2)$$

o parametrach

$$\mu_y = \boldsymbol{\mu}_n^\intercal\boldsymbol{x}\,, \quad \sigma_y^2 = \sigma^2 + \boldsymbol{x}^\intercal\boldsymbol{\Sigma}_n\boldsymbol{x}\,.$$

## 2.2 Regresja grzbietowa

Rozważmy Bayesowską regresję liniową, w której prior ma postać $\boldsymbol{w} \sim \mathcal{N}(0, \tau^2\boldsymbol{I})$, czyli elementy wektora parametrów są a priori wzajemnie niezależne a

ich amplitudy są rzędu $\tau$. Znajdźmy estymatę MAP parametrów $\boldsymbol{w}$. Zauważmy, że

$$-\log p(\boldsymbol{w} \mid \mathcal{X}) \propto \left[ \frac{1}{2\sigma^2} \sum_{i=1}^{n} (y_i - \phi(\boldsymbol{x}_i; \boldsymbol{w}))^2 + \frac{1}{2\tau^2} \|\boldsymbol{w}\|_2^2 \right].$$

Wyrażenie w nawiasach możemy przemnożyć przez $\sigma^2$ – nie wpłynie to na położenie minimum, bo $\sigma^2$ to dodatnia stała. Estymata MAP w tym modelu to zatem

$$\boldsymbol{w}_{\text{MAP}} = \arg\min_{\boldsymbol{w}} \left[ \frac{1}{2} \sum_{i=1}^{n} (y_i - \phi(\boldsymbol{x}_i; \boldsymbol{w}))^2 + \frac{\gamma}{2} \|\boldsymbol{w}\|_2^2 \right]$$

gdzie $\gamma = \sigma^2/\tau^2$ to ***stała regularyzująca***. Pierwszy z członów to znana z OLS suma kwadratów rezyduów, natomiast drugi człon to ***człon regularyzujący*** rozwiązanie, który ogranicza amplitudy parametrów (wag). Jest to tzw. model ***regresji grzbietowej*** (ang. ***ridge regression***). Jest to jednocześnie oczywiście moda rozkładu a posteriori, czyli w przypadku rozkładu normalnego – wyznaczony wcześniej parametr $\boldsymbol{\mu_n}$

$$\boldsymbol{w}_{\text{MAP}} = \left( \frac{\sigma^2}{\tau^2} \boldsymbol{I} + \boldsymbol{X}^\intercal \boldsymbol{X} \right)^{-1} \boldsymbol{X}^\intercal \boldsymbol{y}.$$

## 2.3 Regresja w ujęciu częstościowym

Podstawowym pytaniem, na które sama estymacja punktowa nie daje odpowiedzi w przypadku modelu liniowego jest: czy dany podzbiór współczynników jest istotny statystycznie? Oczywiście w ujęciu Bayesowskim odpowiedzi na to pytanie udziela rozkład a posteriori nad parametrami modelu. Chcemy mieć jednak również klasyczną, częstościową argumentację. Niech $\hat{y}_i$ będzie odpowiedzą modelu dla cech $\boldsymbol{x}_i$, a $y_i$ prawdziwą odpowiedzią. Wielkość

$$\text{RSS} := \sum_{i=1}^{N} (y_i - \hat{y}_i)^2$$

nazywamy ***rezydualną sumą kwadratów*** (ang. ***residual sum of squares***). Wówczas jeśli potrzebujemy testu sprawdzającego istotność podzbioru współczynników tzn. testujemy hipotezę

$$H_0 : w_{i_1} = 0, \ldots, w_{i_q} = 0$$

to odpowiednią statystyką testową jest

$$\frac{\text{RSS}_0 - \text{RSS}}{\text{RSS}} \frac{n-p-1}{q} \sim F(q, n-p-1),$$

gdzie $\text{RSS}_0$ oznacza rezydualną sumę kwadratów dla modelu spełniającego $H_0$, $F$ oznacza rozkład F Snedecora, natomiast $p$ to liczba wszystkich parametrów modelu.

Korzystając z metod częstościowych możemy również podać przedział ufności na poziomie $1-\alpha$ przy przewidywaniu odpowiedzi dla nieznanej wartości $\boldsymbol{x}_0$ jako

$$\hat{\boldsymbol{w}}^\intercal \boldsymbol{x}_0 \pm t_{1-\frac{\alpha}{2}, n-p-1} \hat{\sigma}^2 \sqrt{\boldsymbol{x}_0^\intercal \left( \boldsymbol{X}^\intercal \boldsymbol{X} \right)^{-1} \boldsymbol{x}_0},$$

gdzie

$$\hat{\sigma}^2 = \frac{\text{RSS}}{n-p-1}$$

natomiast $t_{1-\frac{\alpha}{2}, n-p-1}$ jest kwantylem rzędu $1-\frac{\alpha}{2}$ rozkładu t Studenta z parametrem $n-p-1$.

## 2.4 Predyktory jakościowe

Zmienne jakościowe, które mają być użyte jako predyktory w regresji liniowej wymagają reprezentacji numerycznej. Typowo stosowanym kodowaniem jest kodowanie za pomocą tzw. ***dummy variables*** w taki sposób, że zmienna jakościowa o $k$ poziomach jest kodowana jako wektor binarny o $k-1$ składowych (aby uniknąć tzw. ***dummy variable trap*** – naiwny one-hot-encoding powoduje redundancję w postaci liniowej zależności składowych wektora kodującego $x_1 + \ldots + x_k = 1$). Przykładowo dla zmiennej jakościowej $\mathsf{ShelveLoc} \in \{\mathsf{Bad}, \mathsf{Medium}, \mathsf{Good}\}$ tworzymy wektor binarny o dwóch składowych: $\mathsf{ShelveLocBad} \in \{0,1\}$, $\mathsf{ShelveLocGood} \in \{0,1\}$. Wektor $(0,0)$ reprezentuje wówczas wartość $\mathsf{Medium}$ zmiennej $\mathsf{ShelveLoc}$.

## 2.5 Ocena jakości dopasowania

Najpopularniejszymi miarami dopasowania modelu są:

- RSE (ang. ***residual standard error***) zdefiniowany jako $\sqrt{\frac{\text{RSS}}{n-p-1}}$,
- Ułamek wariancji wyjaśnionej przez model

$$R^2 := 1 - \frac{\text{RSS}}{\text{TSS}}, \quad \text{TSS} = \sum_{i=1}^{n} (y_i - \overline{y})^2,$$

  który powinien być jak najbliższy 1.

Należy pamiętać, iż powyższe miary dopasowania ***nie*** są dobrymi miarami przy porównywaniu modeli o ***różnej*** liczbie predyktorów (jeśli liczymy te miary na zbiorze treningowym). Aby poradzić sobie z tym problemem mamy dwie opcje: metody bezpośrednie, w których estymujemy błąd testowy za pomocą walidacji krzyżowej lub metody holdout lub metody pośrednie, w których wprowadzamy poprawki do błędu treningowego. W szczególności miarami pośrednimi są:

- adjusted $R^2$ obliczany jako $1 - \mathrm{RSS}(n-1)/\mathrm{TSS}(n-p-1)$
- kryterium Mallowa $C_p = \frac{1}{n}(\mathrm{RSS} + 2p\hat{\sigma}^2)$

# 3 Uogólnione modele liniowe

Prosty model regresji liniowej może posłużyć nam jako wzór do tworzenia bardziej skomplikowanych modeli statystycznych. Formalnie dowolny model statystyczny jest uogólnionym modelem liniowym (ang. ***Generalized Linear Model, GLM***) jeśli zachodzi

1. Parametry $\boldsymbol{\beta}$ modelu są związane z wektorem $\boldsymbol{x}$ jedynie poprzez tzw. ***składnik systematyczny*** $\boldsymbol{\beta}^\intercal \boldsymbol{x}$ (jest to zatem model liniowy).
2. Rozkład $y \mid \boldsymbol{x}, \boldsymbol{\beta}$ pochodzi z ustalonej wykładniczej rodziny rozkładów $P(\theta, \tau)$ z parametrami $\theta, \tau$ o gęstości danej przez
$$p(z; \theta, \tau) = \exp\left[\frac{\theta z - f(\theta)}{d(\tau)} + k(z, \tau)\right],$$
dla pewnych funkcji $f, d, k$.
3. Parametr $\theta$ jest związany ze składnikiem systematycznym równością
$$g(f'(\theta)) = \boldsymbol{\beta}^\intercal \boldsymbol{x}$$
dla pewnej funkcji wiążącej $g$.

## 3.1 Regresja logistyczna

Rozpatrzymy teraz problem klasyfikacji binarnej z perspektywy estymacji MLE. Niech $\mathcal{X} = \{t_i(\boldsymbol{x}_i)\}_{i=1}^n$ będzie zbiorem obserwacji i.i.d., gdzie zakładamy, iż $t \in \{0, 1\}$. Jako model statystyczny przyjmiemy, iż klasa $t(\boldsymbol{x})$ pochodzi z rozkładu Bernoulliego z parametrem $\pi(\boldsymbol{x}; \boldsymbol{w})$ (prawdopodobieństwem klasy pozytywnej, gdzie $\pi$ jest tzw. funkcją wiążącą, ang. ***link function***) zależnym od estymowanych parametrów $\boldsymbol{w}$

$$t \mid \boldsymbol{x}, \boldsymbol{w} \sim \mathrm{Ber}(\pi(\boldsymbol{x}; \boldsymbol{w}))\,.$$

W przypadku regresji logistycznej jako funkcję $\pi(\boldsymbol{x}; \boldsymbol{w})$ przyjmiemy $\sigma(\phi(\boldsymbol{x}; \boldsymbol{w}))$, gdzie

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

to tzw. ***funkcja logistyczna*** (będąca szczególnym przypadkiem funkcji sigmoidalnej), natomiast $\phi(\boldsymbol{x}; \boldsymbol{w})$ może być dowolną funkcją wykorzystywaną do zagadnienia regresji. W przypadku regresji logistycznej przyjmujemy prosty model liniowy

$$\phi(\boldsymbol{x}; \boldsymbol{w}) = \boldsymbol{w}^\intercal \boldsymbol{x}\,.$$

Zauważmy, iż taka postać uwzględnia człon stały poprzez podstawienie $\boldsymbol{x}^\intercal \leftarrow \begin{bmatrix}\boldsymbol{x}^\intercal & 1\end{bmatrix}$. Wiarygodność dla powyższego modelu statystycznego ma postać

$$p(\mathcal{X} \mid \boldsymbol{w}) = \prod_{i=1}^n \pi(\boldsymbol{x}_i; \boldsymbol{w})^{t_i}(1 - \pi(\boldsymbol{x}_i; \boldsymbol{w}))^{1-t_i}\,,$$

skąd funkcja NLL

$$\boxed{L(\mathcal{X}; \boldsymbol{w}) = -\sum_{i=1}^n \left[t_i \log \pi_i + (1 - t_i)\log(1 - \pi_i)\right]\,,}$$

gdzie $\pi_i = \pi(\boldsymbol{x}_i; \boldsymbol{w})$. Taką funkcję kosztu nazywamy ***binarną entropią krzyżową*** (ang. ***binary cross-entropy function***). Niestety dla takiej postaci funkcji kosztu nie można znaleźć minimum w postaci analitycznej (jak zrobiliśmy w przypadku regresji liniowej), dlatego musimy wykorzystywać algorytmy optymalizacji numerycznej, które najczęściej wykorzystują pierwsze pochodne funkcji kosztu po parametrach. Typowo wykorzystywanym do tego celu algorytmem jest ***spadek wzdłuż gradientu*** (ang. ***gradient descent***), w którym iteracyjnie wykonujemy

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \eta \frac{\partial L}{\partial \boldsymbol{w}}\bigg|_{\boldsymbol{w}_t}\,.$$

Wyznaczymy więc pochodne funkcji $L$ po parametrach $\boldsymbol{w}$. Wprowadźmy macierz $X_{\alpha\beta} = x_\beta^{(\alpha)}$ oraz wektory $\phi_\alpha = \phi(\boldsymbol{x}_\alpha; \boldsymbol{w})$, $\sigma_\alpha = \sigma(\phi_\alpha)$. Wówczas

$$L(\mathcal{X}; \boldsymbol{w}) = -\sum_{\alpha=1}^n \left[t_\alpha \log \sigma_\alpha + (1 - t_\alpha)\log(1 - \sigma_\alpha)\right]\,.$$

W takim razie mamy

$$\frac{\partial L}{\partial \phi_\alpha} = \sum_{\beta=1}^n \frac{\partial L}{\partial \sigma_\beta}\frac{\partial \sigma_\beta}{\partial \phi_\alpha}\,,$$

gdzie

$$\frac{\partial L}{\partial \sigma_\beta} = \frac{1 - t_\beta}{1 - \sigma_\beta} - \frac{t_\beta}{\sigma_\beta}$$

oraz

$$\frac{\partial \sigma_\beta}{\partial \phi_\alpha} = \sigma'(\phi_\beta)\delta_{\alpha\beta}\,,$$

gdzie

$$\sigma'(z) = \frac{\mathrm{e}^{-z}}{(1 + \mathrm{e}^{-z})^2} = \sigma(z)(1 - \sigma(z))\,,$$

zatem

$$\frac{\partial L}{\partial \phi_\alpha} = \left(\frac{1 - t_\alpha}{1 - \sigma_\alpha} - \frac{t_\alpha}{\sigma_\alpha}\right)\sigma_\alpha(1 - \sigma_\alpha) = \sigma_\alpha - t_\alpha\,.$$

W takim razie, z powyższego mamy

$$\frac{\partial L}{\partial w_\alpha} = \sum_{\beta=1}^{n} \frac{\partial L}{\partial \phi_\beta} \frac{\partial \phi_\beta}{\partial w_\alpha} = \sum_{\beta=1}^{n} (\sigma_\beta - t_\beta) X_{\beta\alpha} \,,$$

co możemy zapisać w zwartej postaci macierzowej

$$\frac{\partial L}{\partial \boldsymbol{w}} = \boldsymbol{X}^\intercal (\boldsymbol{\sigma} - \boldsymbol{t}) \,.$$

## 3.2 Wieloklasowa regresja logistyczna

Regresję logistyczną możemy w prosty sposób uogólnić na przypadek klasyfikacji jednej z $k$ klas. Zakładamy teraz, iż zbiór obserwacji i.i.d. ma postać $\mathcal{X} = \{t_i(\boldsymbol{x}_i)\}_{i=1}^n$ dla $t \in \{1, \ldots, k\}$. Nasz model statystyczny uogólnimy do postaci rozkładu kategorialnego (ang. ***categorical distribution*** lub ***multinomial distribution***) z parametrem $\boldsymbol{\pi}(\boldsymbol{x}; \boldsymbol{W})$ (wektorem prawdopodobieństw każdej klasy) zależnym od estymowanych parametrów $\boldsymbol{W}$

$$t \mid \boldsymbol{x}, \boldsymbol{W} \sim \mathrm{Cat}(\boldsymbol{\pi}(\boldsymbol{x}; \boldsymbol{W})) \,.$$

W przypadku regresji softmax jako funkcję $\boldsymbol{\pi} : \mathbb{R}^m \mapsto [0; 1]^k$ przyjmiemy funkcję $\boldsymbol{\sigma}(\boldsymbol{\phi}(\boldsymbol{x}; \boldsymbol{W}))$, gdzie $\boldsymbol{\sigma} : \mathbb{R}^k \mapsto [0; 1]^k$

$$\sigma_\alpha(\boldsymbol{z}) = \frac{\exp(z_\alpha)}{\sum_{\beta=1}^k \exp(z_\beta)}$$

to tzw. ***funkcja softmax***, natomiast $\boldsymbol{\phi} : \mathbb{R}^m \mapsto \mathbb{R}^k$, to dowolna funkcja. W przypadku regresji softmax przyjmiemy prosty model liniowy

$$\boldsymbol{\phi}(\boldsymbol{x}; \boldsymbol{W}) = \boldsymbol{W}\boldsymbol{x} \,.$$

Wiarygodność dla powyższego modelu ma postać

$$\prod_{\alpha=1}^{n} \prod_{\beta=1}^{k} \pi_{\alpha\beta}^{t_{\alpha\beta}} \,,$$

skąd funkcja NLL ma postać

$$\boxed{L(\mathcal{X}; \boldsymbol{W}) = -\sum_{\alpha=1}^{n} \sum_{\beta=1}^{k} t_{\alpha\beta} \log \pi_{\alpha\beta} \,,}$$

gdzie

$$\pi_{\alpha\beta} = \sigma_\beta(\boldsymbol{\phi}(\boldsymbol{x}_\alpha; \boldsymbol{W})) = \frac{\exp \phi_{\alpha\beta}}{\sum_{\gamma=1}^{k} \exp \phi_{\alpha\gamma}} \,.$$

$$\phi_{\alpha\beta} = \phi_\beta(\boldsymbol{x}_\alpha; \boldsymbol{W})$$

$$t_{\alpha\beta} = [t_\alpha = \beta]$$

Wiersze macierzy $\boldsymbol{t}$ są binarnymi wektorami kodującymi w sposób one-hot klasę danego przykładu. Niestety dla takiej postaci funkcji kosztu nie można znaleźć minimum w postaci analitycznej, dlatego musimy wykorzystywać algorytmy optymalizacji numerycznej. Wyznaczymy więc jeszcze pierwsze pochodne funkcji $L$ po parametrach $\boldsymbol{W}$

$$\frac{\partial L}{\partial \pi_{\beta\beta'}} = -\frac{t_{\beta\beta'}}{\pi_{\beta\beta'}} \,.$$

Jednocześnie mamy

$$\frac{\partial L}{\partial \phi_{\alpha\alpha'}} = \sum_{\beta=1}^{n} \sum_{\beta'=1}^{k} \frac{\partial L}{\partial \pi_{\beta\beta'}} \frac{\partial \pi_{\beta\beta'}}{\partial \phi_{\alpha\alpha'}} \,,$$

gdzie

$$\begin{aligned} \frac{\partial \pi_{\beta\beta'}}{\partial \phi_{\alpha\alpha'}} &= \frac{\exp \phi_{\beta\beta'} \delta_{\alpha\beta} \delta_{\alpha'\beta'}}{\sum_{\gamma=1}^{k} \exp \phi_{\beta\gamma}} - \frac{\exp \phi_{\beta\beta'} \exp \phi_{\beta\alpha'} \delta_{\alpha\beta}}{\left[\sum_{\gamma=1}^{k} \exp \phi_{\beta\gamma}\right]^2} \\ &= \pi_{\beta\beta'} \delta_{\alpha\beta} \delta_{\alpha'\beta'} - \pi_{\beta\beta'} \pi_{\beta\alpha'} \delta_{\alpha\beta} \,. \end{aligned}$$

Z powyższego mamy zatem

$$\frac{\partial L}{\partial \phi_{\alpha\alpha'}} = \pi_{\alpha\alpha'} \sum_{\beta'=1}^{k} t_{\alpha\beta'} - t_{\alpha\alpha'} = \pi_{\alpha\alpha'} - t_{\alpha\alpha'} \,.$$

Ostatecznie zatem

$$\frac{\partial L}{\partial W_{\beta\beta'}} = \sum_{\alpha=1}^{n} \sum_{\alpha'=1}^{k} \frac{\partial L}{\partial \phi_{\alpha\alpha'}} \frac{\partial \phi_{\alpha\alpha'}}{\partial W_{\beta\beta'}} \,,$$

gdzie

$$\frac{\partial \phi_{\alpha\alpha'}}{\partial W_{\beta\beta'}} = X_{\alpha\beta'} \delta_{\alpha'\beta} \,,$$

zatem

$$\frac{\partial L}{\partial W_{\beta\beta'}} = \sum_{\alpha=1}^{n} (\pi_{\alpha\beta} - t_{\alpha\beta}) X_{\alpha\beta'} \,,$$

co możemy zapisać w zwartej postaci macierzowej

$$\frac{\partial L}{\partial \boldsymbol{W}} = (\boldsymbol{\pi} - \boldsymbol{t})^\intercal \boldsymbol{X} \,.$$

## 3.3 Regresja probitowa

Regresja probitowa to model analogiczny do regresji logistycznej, w którym jako funkcję wiążącą przyjmujemy

$$\pi(\boldsymbol{x}; \boldsymbol{w}) = \Phi(\boldsymbol{w}^\intercal \boldsymbol{x}) \,,$$

gdzie $\Phi$ jest dystrybuantą standardowego, jednowymiarowego rozkładu normalnego. Taki model możemy

traktować jako model ze ***zmienną ukrytą*** (ang. ***latent variable***)

$$h \mid \boldsymbol{x}, \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{w}^\intercal \boldsymbol{x}, 1)\,,$$

która wyznacza obserwowalną zmienną odpowiedzi jako

$$t = \begin{cases} 1\,, & \text{jeśli } h > 0 \\ 0\,, & \text{w.p.p.} \end{cases}$$

### 3.4 Regresja Poissonowska

Regresja Poissonowksa to podstawowy model dla danych zliczeniowych, w których zmienna objaśniana $t$ przyjmuje dowolne wartości naturalne $t \in \mathbb{N}$. Jest to zasadniczo model postaci

$$t \mid \boldsymbol{x}, \boldsymbol{w} \sim \text{Pois}\left(\mathrm{e}^{\boldsymbol{w}^\intercal \boldsymbol{x}}\right)\,.$$

Zanegowana logarytmiczna funkcja wiarygodności ma dla tego modelu postać

$$L(\mathcal{X}; \boldsymbol{w}) = -\sum_{i=1}^{n} \left(t_i \boldsymbol{w}^\intercal \boldsymbol{x}_i - \mathrm{e}^{\boldsymbol{w}^\intercal \boldsymbol{x}_i}\right)\,,$$

pamiętając, iż funkcja masy prawdopodobieństwa dla rozkładu Poissona to

$$P(Y = k; \lambda) = \frac{\lambda^k}{k!}\mathrm{e}^{-\lambda}\,.$$

### 3.5 Regresja porządkowa

Regresja porządkowa to podstawowy model w przypadku, w którym zmienna objaśniana $t$ przyjmuje wartości z ***liniowo uporządkowanego zbioru***, o którym bez straty ogólności założymy, iż ma postać $\{1, 2, \ldots, k\}$. Regresja porządkowa może być rozumiana jako model klasyfikacyjny, ale ze względu na nacisk na zachowanie porządku etykiet przypomina też regresję całkowitoliczbową (Poissona). Jest to zasadniczo model postaci

$$t \mid \boldsymbol{x}, \boldsymbol{w}, \theta_1, \ldots, \theta_{k-1} \sim \text{Cat}(\pi_1(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{\theta}), \ldots, \pi_k(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{\theta}))\,,$$

gdzie

$$\pi_\alpha(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{\theta}) := \Phi(\theta_\alpha - \boldsymbol{w}^\intercal \boldsymbol{x}) - \Phi(\theta_{\alpha-1} - \boldsymbol{w}^\intercal \boldsymbol{x})$$

dla $\theta_0 = -\infty < \theta_1 < \cdots < \theta_{k-1} < \theta_k = +\infty$ i $\Phi$ będącego dystrybuantą standardowego, jedynowymiarowego rozkładu normalnego. Zauważmy, że przy takim wyborze funkcjie $\pi_\alpha$ zachodzi wymagany warunek

$$\begin{aligned}\sum_{\alpha=1}^{k} \pi_\alpha(\boldsymbol{x}; \boldsymbol{w}, \boldsymbol{\theta}) &= \Phi(\theta_k - \boldsymbol{w}^\intercal \boldsymbol{x}) - \Phi(\theta_0 - \boldsymbol{w}^\intercal \boldsymbol{x}) \\ &= \Phi(+\infty) - \Phi(-\infty) = 1\,.\end{aligned}$$

Parametry modelu $\boldsymbol{w}$ i $\boldsymbol{\theta}$ są estymowane metodą największej wiarygodności. Zauważmy, że możemy ten model interpretować jako model ze ***zmienną ukrytą*** (ang. ***latent variable***)

$$h \mid \boldsymbol{x}, \boldsymbol{w} \sim \mathcal{N}(\boldsymbol{w}^\intercal \boldsymbol{x}, 1)\,,$$

która wyznacza obserwowaną odpowiedź modelu $t$ zgodnie z

$$t = \begin{cases} 1\,, & \text{jeśli } \theta_0 < h \leq \theta_1 \\ 2\,, & \text{jeśli } \theta_1 < h \leq \theta_2 \\ \vdots & \\ k\,, & \text{jeśli } \theta_{k-1} < h \leq \theta_k \end{cases}$$

## 4 Wprowadzenie do procesów Gaussowskich

Macierz kowariancji $n$–wymiarowej zmiennej losowej $\boldsymbol{x}$ o wartości oczekiwanej $\boldsymbol{\mu}$ jest zdefiniowana jako

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

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

$$\boldsymbol{K} = \boldsymbol{L}\boldsymbol{L}^\intercal\,,$$

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

$$\mathbb{E}\left[(\boldsymbol{L}\boldsymbol{z})(\boldsymbol{L}\boldsymbol{z})^\intercal\right] = \boldsymbol{L}\mathbb{E}[\boldsymbol{z}\boldsymbol{z}^\intercal]\boldsymbol{L}^\intercal = \boldsymbol{L}\boldsymbol{1}\boldsymbol{L}^\intercal = \boldsymbol{K}\,.$$

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

Funkcję $k : \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}$ taką, że $\forall m \in \mathbb{N} : \forall X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\} \subset \mathbb{R}^n$ macierz

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

jest dodatnio określoną macierzą symetryczną nazywamy funkcją kowariancji, jądrem dodatnio określonym (ang. ***positive definite kernel***) lub ***jądrem Mercera***.

Dla dwóch zbiorów punktów $X = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\} \subset \mathbb{R}^n$ i $Y = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_s\} \subset \mathbb{R}^n$ i funkcji kowariancji $k$ wprowadzimy oznaczenie

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

Poniżej podajemy kilka przykładów funkcji kowariancji

- ***Gaussian kernel*** dla normy $\|\cdot\|$ i hiperparametrów $a, l$ (amplituda i skala długości)

$$k(\boldsymbol{x},\boldsymbol{y}) = a^2 \exp\left\{-\frac{1}{2l^2}\|\boldsymbol{x}-\boldsymbol{y}\|^2\right\}$$

- ***Periodic kernel*** dla normy $\|\cdot\|$ i hiperparametrów $a, l, p$ (amplituda, skala długości, okres zmienności)

$$k(\boldsymbol{x},\boldsymbol{y}) = a^2 \exp\left\{-\frac{2}{l^2}\sin^2\left(\frac{\pi}{p}\|\boldsymbol{x}-\boldsymbol{y}\|\right)\right\}$$

- ***White noise kernel*** dla hiper-parametru $\sigma$

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

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

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

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

Suma lub iloczyn dwóch funkcji kowariancji oraz złożenie funkcji kowariancji z wielomianem o nieujemnych współczynnikach jest również funkcją kowariancji

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

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

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

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

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 $\boldsymbol{\mu}_X = \boldsymbol{0}$, natomiast przepisem na macierz kowariancji może być zdefiniowana wyżej funkcja kowariancji $k(X,X)$ tj.

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

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 dokładnie znanych przez nas punktach $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$ 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

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

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

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

Zauważmy, iż z twierdzenia o własnościach niezdegenerowanego rozkładu normalnego wnioskujemy, iż warunkowy $\boldsymbol{f}_Y \mid \boldsymbol{f}_X$ jest również rozkładem normalnym o parametrach

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

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

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

gdzie $\sigma$ 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

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

# 5 Liczby losowe w komputerze

Opiszemy teraz pokrótce metody generowania liczb pseudolosowych z dowolnych rozkładów prawdopodobieństwa w sposób algorytmiczny. Podstawowym narzędziem, którego będziemy potrzebować do generowania próbek z bardziej skomplikowanych rozkładów będzie prosty generator liczb z rozkładu jednostajnego $\mathcal{U}(0,1)$. Moglibyśmy oczywiście wykorzystać jakieś fizyczne urządzenie lub proces, który generuje liczby prawdziwie losowe (np. detektor Geigera-Mullera, szum lamp elektronowych, ruletka), ale błędem byłaby rezygnacja z odtwarzalności. Poszukujemy zatem deterministycznej metody, która generuje sekwencje liczb, które są w przybliżeniu losowe. Podstawową metodą do algorytmicznego generowania liczb pseudolosowych jest tzw. ***liniowy generator kongruentny*** (ang. ***Linear Congruential Generator, LCG***), który jest opisany zależnością rekurencyjną

$$I_{j+1} = (aI_j + c) \mod m\,,$$

gdzie $a, c, m$ to pewne ustalone dodatnie liczby całkowite, a $I_0$ to tzw. ziarno (ang. ***seed***). LCG generuje liczby całkowite, więc w dużym uproszczeniu liczby zmiennoprzecinkowe z rozkładu $\mathcal{U}(0,1)$ otrzymujemy jako $I_j/m$ (trzeba tutaj jednak uwzględnić problemy wynikające z arytmetyki zmiennoprzecinkowej).

## 5.1 Metoda odwrotnej dystrybuanty

Mając już generator liczb z rozkładu jednostajnego $\mathcal{U}(0,1)$ i znając jawny wzór na dystrybuantę $F(x)$ innego rozkładu jednowymiarowego możemy generować liczby z tego rozkładu korzystając z tzw. ***metody odwrotnej dystrybuanty***. Istotnie, jeśli $U \sim \mathcal{U}(0,1)$, to $F^{-1}(U) \sim F$. Istotnie

$$\Pr\big(F^{-1}(U) \le x\big) = \Pr(U \le F(x)) = F(x)\,.$$

Metoda ta ma jedną zasadniczą wadę – musimy znać jawny wzór na dystrybuantę $F$. W przypadku np. tak ważnych rozkładów, jak rozkład normalny dystrybuanta nie jest funkcją elementarną i metoda ta nie jest najlepsza.

## 5.2 Metoda Boxa–Mullera

W przypadku rozkładu normalnego znacznie lepszą metodą jest tzw. ***metoda Boxa–Mullera***. Weźmy rozkład łączny dwóch niezależnych zmiennych losowych $X, Y$ pochodzących ze standardowego rozkładu normalnego

$$p(x,y) = \frac{1}{2\pi}\exp\left(\frac{1}{2}(x^2+y^2)\right).$$

Skorzystamy ze wzoru na transformację zmiennych losowych. Istotnie niech

$$X = \sqrt{Z}\cos\Phi\,,\quad Y = \sqrt{Z}\sin\Phi\,,$$

dla $0 < Z$ oraz $0 \le \Phi < 2\pi$, wówczas

$$q(z,\phi) = \frac{1}{4\pi}\exp\frac{z}{2}\,.$$

Zauważmy, że otrzymaliśmy sferycznie symetryczny rozkład wykładniczy. Możemy zatem wylosować kąt $\phi$ z rozkładu jednostajnego oraz wartość $z$ z rozkładu wykładniczego korzystając z metody odwrotnej dystrybuanty. Wówczas wartości $x = \sqrt{z}\cos\phi$, $y = \sqrt{z}\sin\phi$ będą pochodzić ze standardowego rozkładu normalnego. Aby wygenerować próbki z ogólnego wielowymiarowego rozkładu normalnego, korzystamy z definicji, tj. najpierw generujemy $n$ próbek ze standardowego rozkładu normalnego, a następnie korzystamy z przekształcenia afinicznego $\boldsymbol{x} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{\mu}$, gdzie $\boldsymbol{\Sigma} = \boldsymbol{A}\boldsymbol{A}^\intercal$.

## 5.3 Monte Carlo

Umiemy już generować próbki z wielowymiarowego rozkładu normalnego. Chcemy teraz poznać metodę, która umożliwi generowanie próbek ze skomplikowanych, wielowymiarowych rozkładów prawdopodobieństwa, których gęstość znamy jedynie z dokładnością do stałej normalizującej, tj. znamy jedynie $\tilde{p}(\boldsymbol{x}) = Zp(\boldsymbol{x})$. Ograniczenie to wynika z chęci próbkowania z posteriora $p(\boldsymbol{x} \mid \boldsymbol{y})$ w sytuacji, gdy znamy jedynie rozkład łączny $p(\boldsymbol{x},\boldsymbol{y}) = p(\boldsymbol{y} \mid \boldsymbol{x})p_{\boldsymbol{X}}(\boldsymbol{x})$. Okazuje się, iż znajomość rozkładu jedynie z dokładnością do stałej normalizującej jest wystarczająca do generowania próbek z tego rozkładu. Generowanie próbek z kolei wystarcza natomiast, na mocy silnego prawa wielkich liczb, do szacowania wartości średnich dowolnych funkcji zmiennej $\boldsymbol{x}$. Przypomnijmy, iż na mocy silnego prawa wielkich liczb ciąg średnich częściowych $(\overline{\boldsymbol{X}}_n)$ ciągu zmiennych losowych $(\boldsymbol{X}_n)$ i.i.d. z rozkładu $\boldsymbol{X} \sim \mathcal{D}$ jest zbieżny z prawdopodobieństwem 1 do wartości oczekiwanej $\mathbb{E}[\boldsymbol{X}]$ tj.

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

Wartość oczekiwaną $\mathbb{E}[\boldsymbol{X}]$ możemy zatem przybliżyć średnią $\overline{\boldsymbol{X}}_n$ z dużej ilości próbek. Poniżej przedstawimy dwa algorytmy próbkowania: algorytm IS oraz Metropolisa–Hastingsa będący szczególną realizacją całej rodziny algorytmów próbkowania zwanych Markov Chain Monte Carlo (MCMC).

### 5.3.1 Importance sampling

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

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

tj. szukamy

$$\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[f(\boldsymbol{x})] = \int f(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\,.$$

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

$$\begin{aligned}\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[f(\boldsymbol{x})] &= \int f(\boldsymbol{x})p(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\\ &= \int f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}q(\boldsymbol{x})\mathrm{d}\boldsymbol{x}\\ &= \mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]\\ &= \frac{Z_q}{Z_p}\mathbb{E}_q\left[f(\boldsymbol{x})\frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})}\right]\,.\end{aligned}$$

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

$$Z_p = \int \tilde{p}(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = Z_q\int\frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})}q(\boldsymbol{x})\,\mathrm{d}\boldsymbol{x} = Z_q\mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}\left[\frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})}\right]\,,$$

skąd ostatecznie

$$\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[f(\boldsymbol{x})] = \frac{\mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}\left[f(\boldsymbol{x})\frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})}\right]}{\mathbb{E}_{\boldsymbol{x}\sim q(\boldsymbol{x})}\left[\frac{\tilde{p}(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})}\right]}\,.$$

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

$$\boxed{\mathbb{E}_{\boldsymbol{x}\sim p(\boldsymbol{x})}[f(\boldsymbol{x})] \approx \frac{\sum_{i=1}^m f(\boldsymbol{x}_i)\frac{\tilde{p}(\boldsymbol{x}_i)}{\tilde{q}(\boldsymbol{x}_i)}}{\sum_{j=1}^m \frac{\tilde{p}(\boldsymbol{x}_j)}{\tilde{q}(\boldsymbol{x}_j)}} = \sum_{i=1}^m \lambda_i f(\boldsymbol{x}_i)\,,}$$

gdzie

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

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

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

dla $\boldsymbol{x}_i \sim q$ mamy

$$\begin{aligned}\mathbb{V}[\overline{f}_m] &= \frac{1}{m}\mathbb{V}_q\left[f(\boldsymbol{x})\frac{p(\boldsymbol{x})}{q(\boldsymbol{x})}\right]\\ &= \frac{1}{m}\int\frac{(f(\boldsymbol{x})p(\boldsymbol{x}) - \mu_f q(\boldsymbol{x}))^2}{q(\boldsymbol{x})}\,\mathrm{d}\boldsymbol{x}\,.\end{aligned}$$

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 $f(\boldsymbol{x})p(\boldsymbol{x})$, co może być trudne do praktycznego zrealizowania.

### 5.3.2 Algorytm Metropolisa–Hastingsa

Cała klasa algorytmów próbkowania MCMC opiera się na idei wyrażenia generowania próbek jako ewolucji pewnego łańcucha Markowa.

Łańcuchem Markowa nazwiemy ciąg zmiennych losowych $(\boldsymbol{X}_t)$ o wartościach w $\mathbb{R}^n$ taki, że spełnione jest ***kryterium Markowa***

$$\begin{aligned}\forall A \subset \mathbb{R}^n : \Pr(\boldsymbol{X}_t \in A \mid \boldsymbol{X}_{t-1} = \boldsymbol{x}_{t-1},\ldots,\boldsymbol{X}_0 = \boldsymbol{x}_0)\\ = \Pr(\boldsymbol{X}_t \in A \mid \boldsymbol{X}_{t-1} = \boldsymbol{x}_{t-1})\,.\end{aligned}$$

Elementy ciągu nazywamy stanami łańcucha.

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

$$\int \pi(\boldsymbol{y} \mid \boldsymbol{x})\,\mathrm{d}\boldsymbol{y} = 1\,,$$

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

daną funkcją przejścia $\pi$ nazwiemy rozkładem stacjonarnym tego łańcucha jeśli

$$p(\boldsymbol{y}) = \int \pi(\boldsymbol{y} \mid \boldsymbol{x}) p(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \; .$$

Rozkład stacjonarny danego łańcucha oznaczymy przez $p^*(\boldsymbol{x})$. Zauważmy, iż jeśli stan początkowy łańcucha $\boldsymbol{X}_0$ pochodzi z rozkładu stacjonarnego $p^*$ to każdy kolejny stan $\boldsymbol{X}_t$ 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

$$p_t(\boldsymbol{y}) = \int \pi(\boldsymbol{y} \mid \boldsymbol{x}) p_{t-1}(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x} \; , \quad \text{dla } t > 1.$$

Rozkładem granicznym łańcucha Markowa nazwiemy granicę w sensie zbieżności punktowej

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

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 $p(\boldsymbol{x})$ był rozkładem stacjonarnym łańcucha Markowa jest

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

co wynika z scałkowania powyższego równania. Kryterium to nazywamy ***kryterium lokalnego balansu*** (ang. ***detailed balance condition***).

Podstawowa idea wykorzystania łańcuchów Markowa do generowania próbek ze skomplikowanego rozkładu $p$ jest więc następująca: tworzymy łańcuch Markowa, dla którego $p$ jest rozkładem stacjonarnym, wówczas rozpoczynając w dowolnym dopuszczalnym stanie początkowym $\boldsymbol{X}_0$ po wykonaniu dużej liczby kroków (etap ten nazywamy okresem przejściowym ang. ***burn-in period***) stan $\boldsymbol{X}_t$ (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 $\pi(\boldsymbol{y} \mid \boldsymbol{x})$ dla danego rozkładu granicznego $p(\boldsymbol{x})$. Podstawową konstrukcję podaje algorytm Metropolisa–Hastingsa.

Funkcja przejścia ma postać

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

**Algorithm 1:** Metropolis-Hastings Algorithm

**Input:** Target distribution $p(\boldsymbol{x})$, proposal distribution $q(\boldsymbol{y} \mid \boldsymbol{x})$, number of iterations `max_iter`

**Output:** Samples from the target distribution

Initialize state $\boldsymbol{x}$ with any admissible value;
**for** $i = 1$ **to** ***max_iter*** **do**
  Sample candidate $\boldsymbol{y} \sim q(\boldsymbol{y} \mid \boldsymbol{x})$;
  Compute acceptance ratio:

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

  Accept $\boldsymbol{y}$ with probability $r(\boldsymbol{y} \mid \boldsymbol{x})$;
  **if** ***candidate y is accepted*** **then**
    Set $\boldsymbol{x} \leftarrow \boldsymbol{y}$;
  **else**
    Keep current state $\boldsymbol{x}$;
  **end**
**end**

gdzie

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

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

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

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

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

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***).

# 6 Drzewa decyzyjne, bagging, boosting

Drzewa decyzyjne (ang. ***Classification and Regression Trees, CART***) to metoda uczenia maszynowego, której zasadniczą ideą jest segmentacja przestrzeni predyktorów na skończoną liczbę obszarów o

prostej geometrii, typowo kostek. W każdym z obszarów wartość przewidywana jest obliczana jako średnia (w przypadku regresji) lub moda (w przypadku klasyfikacji) wartości odpowiedzi dla elementów ze zbioru uczącego należących do tego obszaru.

Reguły podziału przestrzeni predyktorów są naturalnie reprezentowane w postaci drzewa binarnego – stąd nazwa metody. Poniżej przedstawiono standardowy algorytm uczenia drzewa CART.

**Algorithm 2:** CART

**Input:** Zbiór danych $\mathcal{X} = \{(\boldsymbol{x}_i, y_i)\}$ w $R$
**Output:** Drzewo decyzyjne (podział przestrzeni cech)

Zainicjalizuj region $R = \{\boldsymbol{x}_i \mid (\boldsymbol{x}_i, y_i) \in \mathcal{X}\}$ jako całą przestrzeń cech;
**if** ***spełniony warunek stopu w*** $R$ **then**
  Zwróć liść z wartością $\overline{y}_R$;
**else**
  Znajdź cechę $k^* \in \{1, \ldots, p\}$ i próg $s^*$ minimalizujące:

$$\sum_{i:\boldsymbol{x}_i \in R_-(k,s)} \left(y_i - \overline{y}_{R_-(k,s)}\right)^2 + \sum_{i:\boldsymbol{x}_i \in R_+(k,s)} \left(y_i - \overline{y}_{R_+(k,s)}\right)^2$$

  gdzie:;
  $R_-(k,s) = \{\boldsymbol{x} \in R \mid x_k < s\}$;
  $R_+(k,s) = \{\boldsymbol{x} \in R \mid x_k \geq s\}$;
  Rekurencyjnie twórz poddrzewa dla podregionów $R_-(k^*, s^*)$ i $R_+(k^*, s^*)$;

Powyższy algorytm został przedstawiony dla zagadnienia regresji zmiennej ciągłej. W przypadku zadania klasyfikacji przewidywana wartość w każdym regionie to klasa najczęściej występująca. Natomiast przy dokonywaniu podziałów zamiast RSS używa się innych miar – najczęście indeksu Giniego

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

lub entropii krzyżowej

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \,,$$

gdzie $\hat{p}_{mk}$ jest proporcją obserwacji w regionie $R_m$, które należą do klasy $k$.

## 6.1 Przycinanie drzewa

Opisany proces prowadzi do modelu z małym błędem treningowym, ale silnie zagrożonego przeuczeniem. Jedną z możliwości poradzenia sobie z tym problemem jest zatrzymywanie podziału przy niezbyt dużym spadku RSS. Lepszą alternatywą jest wyhodowanie dużego drzewa, a następnie jego przycięcie.

## 6.2 Bagging

Bagging opiera się na prostej obserwacji: jeśli $Z_1, \ldots, Z_n$ jest prostą próbą losową z rozkładu o wariancji $\sigma^2$, to $\mathrm{Var}(\overline{Z}) = \sigma^2/n$. Idea jest zatem następująca: z oryginalnego zbioru uczącego $\mathcal{X}$ tworzymy $B$ zbiorów uczących typu bootstrap (tj. zbiorów zawierających tyle samo obserwacji co $\mathcal{X}$, tworzonych poprzez losowanie ***ze zwracaniem*** z $\mathcal{X}$). Na każdym z tych $B$ zbiorów trenujemy drzewo decyzyjne ***bez przycinania*** – jest to więc drzewo o dużej wariancji, ale małym obciążeniu. Finalny model jest tworzony jako średnia lub moda z odpowiedzi modeli z zespołu, tj. w przypadku regresji

$$\hat{f}_{\mathrm{bag}}(\boldsymbol{x}) = \frac{1}{B} \sum_{b=1}^{B} \hat{f}_b(\boldsymbol{x}) \,.$$

Można pokazać, iż zbiory typu bootstrap obejmują średnio 2/3 dostępnych obserwacji. Niewykorzystane obserwacje (tzw, ***out-of-bab observations, OOB***) mogą być użyte do estymacji błędu testowego poprzez uśrednienie błędów OOB każdego z drzew. Bagging zwiększa dokładność predykcji kosztem intepretowalności. W przypadku metody bagging można jednak wprowadzić tzw. ważności cech (ang. ***feature importances***) jako sumaryczne redukcje RSS (lub indeksu Giniego, entropii) uzyskane przez podział względem tego predyktora, uśrednione po $B$ drzewach.

## 6.3 Lasy losowe

Lasy losowe są odmianą metody bagging, w której w trakcie budowania drzew, przy każdym podziale, rozważa się tylko $m < p$ losowo wybranych predyktorów (typowo dla zagadnień regresji $m \approx \sqrt{p}$, a dla klasyfikacji $m \approx p/3$). Poprawka ta redukuje korelacje między drzewami będącą efektem dominacji najważniejszych predyktorów.

Ani w metodzie bagging, ani w lasach losowych zwiększenie liczby drzew $B$ ***nie powoduje przeuczenia***.

## 6.4 Boosting gradientowy

Boosting jest techniką ogólną stosowaną do różnych metod uczenia statystycznego w celu istotnego zwiek-

szenia dokładności. Pomysł uczenia zespołowego jest podobny jak w baggingu i lasach losoweych, jednak łączenie wyników odbywa się w zupełnie inny sposób.

**Algorithm 3:** Proste stopniowe modelowanie addytywne

**Input:** Zbiór danych $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, funkcja straty $L(y, f(\boldsymbol{x}))$, liczba iteracji $M$

Zainicjalizuj model: $f_0(\boldsymbol{x}) \leftarrow 0$;

**for** $m = 1$ **to** $M$ **do**

  Znajdź parametry $(\beta_m, \gamma_m)$ minimalizujące:

$$\sum_{i=1}^{n} L\left(y_i,\ f_{m-1}(\boldsymbol{x}_i) + \beta \cdot b(\boldsymbol{x}_i; \gamma)\right)$$

  Zaktualizuj model:

$$f_m(\boldsymbol{x}) \leftarrow f_{m-1}(\boldsymbol{x}) + \beta_m \cdot b(\boldsymbol{x}; \gamma_m)$$

**return** $f_M(\boldsymbol{x})$

W każdej iteracji szukamy optymalnej (dla danej funkcji kosztu $L$) funkcji bazowej (z pewnej rodziny parametrycznej) i współczynnika, które tworzą wyraz dodawany do aktualnego rozwinięcia funkcji $\hat{f}$

W przeciwieństwie do baggingu nadmierne zwiększanie liczbny drzew $M$ prowadzi do przeuczenia. Brak kontroli nad wielkością drzew $J_m$ może istotnie zdegradować jakość predykcji. Aby uniknąć problemu silnie rozbudowanych drzew często przyjmuje się wspólną, niewielką wartość $J_m = J$ wielkości drzew. Zamiast $J$ można też użyć parametru zwanego głębokością interakcji $d = J - 1$ oznaczającego liczbę podziałów w drzewie (dla $d = 1$ finalny model jest addytywny). Inne metody stosowane w celu uniknięcia przeuczenia to w szczególności: early-stopping; użycie współczynnika spowalniającego $\nu \in (0; 1]$, przez który mnożony jest dodawany człon; subsampling – hodowanie drzew w każdej iteracji na wylosowanym fragmencie zbioru uczącego (tzw. stochastyczny boosting).

**Algorithm 4:** Boosting gradientowy dla drzew

**Input:** Zbiór danych $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$, funkcja straty $L(y, f(\boldsymbol{x}))$, liczba iteracji $M$

Inicjalizuj model:

$$f_0(\boldsymbol{x}) = \arg\min_{\gamma} \sum_{i=1}^{n} L(y_i, \gamma)$$

**for** $m = 1$ **to** $M$ **do**

  Oblicz pseudorezydua;

  **for** $i = 1$ **to** $n$ **do**

    $r_{im} = -\nabla_f L(y_i, f_{m-1}(\boldsymbol{x}_i))$

  Dopasuj regresyjne drzewo decyzyjne do $\{(\boldsymbol{x}_i, r_{im})\}$, uzyskując regiony terminalne $R_{jm}$ dla $j = 1, \ldots, J_m$;

  Oblicz wagi;

  **for** $j = 1$ **to** $J_m$ **do**

    $\gamma_{jm} = \arg\min_{\gamma} \sum_{i:\boldsymbol{x}_i \in R_{jm}} L(y_i, f_{m-1}(\boldsymbol{x}_i) + \gamma)$;

  Aktualizuj model

$$f_m(\boldsymbol{x}) = f_{m-1}(\boldsymbol{x}) + \sum_{j=1}^{J_m} \gamma_{jm} \cdot \mathbf{1}_{R_{jm}}(\boldsymbol{x})$$

**return** $f_M(\boldsymbol{x})$

# 7 Inne metody uczenia maszynowego

## 7.1 KDE

W poprzednim rozdziale opisaliśmy w jaki sposób mając (nieznormalizowaną) funkcję gęstości prawdopodobieństwa $\tilde{p}(\boldsymbol{x})$ generować algorytmicznie próbki z opisanego przez nią rozkładu. Teraz zajmiemy się problemem odwrotnym tj. mając realizację prostej próby losowej z pewnego rozkładu chcemy znaleźć funkcję $\hat{p}(\boldsymbol{x})$, która estymuje gęstość rozkładu prawdopodobieństwa, z którego pochodzą próbki. Opiszemy tutaj jedną z najprostszych metod zwaną estymatorem jądrowym (ang. ***Kernel Density Estimator, KDE***). Estymatorem jądrowym gęstości funkcji $p$ nazywamy funkcję

$$\hat{p}(\boldsymbol{x}) := \frac{1}{n} \sum_{i=1}^{n} K_{\boldsymbol{H}}(\boldsymbol{x} - \boldsymbol{x}_i)\,,$$

gdzie $\boldsymbol{H}$ jest symetryczną i dodatnio określoną macierzą zwaną *bandwidth matrix*, funkcja $K_{\boldsymbol{H}}$ ma postać

$$K_{\boldsymbol{H}}(\boldsymbol{x}) = (\det \boldsymbol{H})^{-1/2} K(\boldsymbol{H}^{-1/2}\boldsymbol{x})\,,$$

gdzie funkcja $K$ zwana jądrem (z ang. *kernel*) jest gęstością prawdopodobieństwa pewnego sferycznie symetrycznego rozkładu wielowymiarowego. Wybór funkcji $K$ nie jest kluczowy ze statystycznego punktu widzenia, więc możemy bez problemu założyć, iż jest to gęstość wielowymiarowego rozkładu normalnego, tj.

$$K_{\boldsymbol{H}}(\boldsymbol{x}) = \frac{1}{\sqrt{(2\pi)^m \det \boldsymbol{H}}} \exp\left(-\frac{1}{2}\boldsymbol{x}^T \boldsymbol{H}^{-1} \boldsymbol{x}\right)\,,$$

gdzie zakładamy $\boldsymbol{x} \in \mathbb{R}^m$. Większym problemem w przypadku KDE jest wybór odpowiedniego parametru $\boldsymbol{H}$. Jednym z prostszych wyborów w przypadku estymatora jest tzw. ***reguła Silvermana***, która podaje następujący przepis na macierz $\boldsymbol{H}$

$$H_{\alpha\beta} = 0\,, \quad \sqrt{H_{\alpha\alpha}} = \left(\frac{4}{4+m}\right)^{\frac{1}{m+4}} n^{\frac{-1}{m+4}} \sigma_\alpha\,,$$

gdzie $\sigma_\alpha$ jest estymatorem wariancji $\alpha$-tej współrzędnej zmiennej $\boldsymbol{X}$. Inną możliwością jest tzw. ***reguła Scotta***

$$H_{\alpha\beta} = 0\,, \quad \sqrt{H_{\alpha\alpha}} = n^{\frac{-1}{m+4}} \sigma_\alpha\,.$$

KDE w praktycznych zastosowaniach często przyspiesza się za pomocą odpowiednich struktur danych do wyszukiwania najbliższych sąsiadów w przestrzeni $\mathbb{R}^m$, tj. zamiast sumować przyczynki od wszystkich punktów $\boldsymbol{x}_i$ dla danego $\boldsymbol{x}$, znajdujemy jego $k$ najbliższych sąsiadów $\boldsymbol{x}$ ze zbioru $\{\boldsymbol{x}_i\}_{i=1}^n$ (np. stosując ***approximate nearest neighbors***) i obliczamy przyczynki do $\hat{p}(\boldsymbol{x})$ tylko od nich.

## 7.2 KNN

Metody oparte o sąsiedztwo to zbiór metod, które wykorzystują najbliższych (w sensie pewnej metryki lub półmetryki) sąsiadów (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 $\mathcal{X} = \{t_i(\boldsymbol{x}_i)\}_{i=1}^n$, 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 $\{\boldsymbol{x}_i\}_{i=1}^n$ 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

$$\hat{t}(\boldsymbol{x}) = \operatorname*{arg\,max}_{c \in \{1,\ldots,C\}} \frac{1}{k} \sum_{i:\boldsymbol{x}_i \in N_k(\boldsymbol{x})} [t_i = c]\,.$$

W przypadku regresji zamiast zwracać klasę występującą najczęściej, zwracamy średnią arytmetyczną wartości $t$ dla $k$ najbliższych sąsiadów (użycie średniej powoduje, iż KNN w przypadku regresji potrafi tylko interpolować wartości)

$$\hat{t}(\boldsymbol{x}) = \frac{1}{k} \sum_{i:\boldsymbol{x}_i \in N_k(\boldsymbol{x})} t_i\,.$$

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 wariancja modelu. KNN jest modelem nieliniowym i potrafi nauczyć się nietrywialnych granic decyzyjnych. Jest również bardziej elastyczny w porównaniu do modeli liniowych (np. regresji logistycznej). Standardowo wykorzystywane metryki to w szczególności metryka euklidesowa, metryka manhattan, podobieństwo cosinusowe.

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 przybliżone (z ang. *Approximate Nearest Neighbors*). W algorytmach przybliżonych zamiast znajdować dokładnych $k$ najbliższych sąsiadów znajdujemy $k$ sąsiadów, którzy niekoniecznie muszą być rzeczywiście najbliższymi, potrafimy to jednak zrobić znacznie szybciej.

## 7.3 NBC

Mając zbiór uczący dla zafania klasyfikacji postaci $\mathcal{X} = \{t_i(\boldsymbol{x}_i)\}_{i=1}^n$ możemy próbować estymować rozkłady $p(\boldsymbol{x} \mid k)$ i $p(k)$ dla klasy $k \in \{1, \ldots, K\}$, a

następnie korzystając z twierdzenia Bayesa stworzyć klasyfikator zgodny z regułą MAP

$$\hat{t}(\boldsymbol{x}) = \operatorname*{arg\,max}_{k\in\{1,\ldots,K\}} p(k \mid \boldsymbol{x}) = \operatorname*{arg\,max}_{k\in\{1,\ldots,K\}} p(\boldsymbol{x} \mid k)p(k)\,.$$

W przypadku drugiego członu sprawa jest prosta – estymatorem MLE jest

$$p(k) = \frac{1}{n}\sum_{i=1}^{n}[t_i = k]\,.$$

Problemem jest zwykle estymowanie rozkładu warunkowego $p(\boldsymbol{x} \mid k)$, zwłaszcza dla wektora cech o wielu predyktorach $p \gg 1$ (klątwa wymiarowości). Naiwny klasyfikator Bayesowski (ang. ***Naive Bayesian Classifier, NBC***) czyni (naiwne) założenie o warunkowej niezależności predyktorów względem klasy, tj.

$$p(x_1, \ldots, x_p \mid k) = \prod_{i=1}^{p} p(x_i \mid k)\,.$$

Wówczas rozkładu warunkowe dla poszczególnych predyktorów $p(x_i \mid k)$ estymujemy niezależni, albo zgodnie z przyjętym modelem zdarzeń (np. przyjmując jakiś rozkład parametryczny i estymując jego parametry), albo używając metod nieparametrycznych (np. jednowymiarowych KDE).

Zauważmy, że NBC zasadniczo różni się od modeli rozważanych wcześniej (regresja liniowa, logistyczna, itp.), w których bezpośrednio modelowaliśmy rozkład warunkowy $p(y \mid \boldsymbol{x})$. Tutaj zamiast rozkładu warunkowego estymujemy rozkład łączny cech i odpowiedzi $p(y, \boldsymbol{x})$. Model taki nazywa się ***modelem generatywnym***. Natomiast model, w którym wprost modelujemy rozkład warunkowy nazywa się ***modelem dyskryminatywnym***.

## 7.4 LDA i QDA

Modele LDA (ang. ***Linear Discriminant Analysis***) i QDA (ang. ***Quadratic Discriminant Analysis***) są modelami generatywnymi, podobnymi do NBC. Zasadniczą różnicą względem NBC jest to, że nie czynią założenia warunkowej niezależności. Zamiast tego zakładają, że rozkład predyktorów w ramach danej klasy jest wielowymiarowym rozkładem normalnym. W przypadku LDA mamy

$$\boldsymbol{x} \mid k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$$

nastomiast w przypadku QDA

$$\boldsymbol{x} \mid k \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\,.$$

Modele LDA i QDA różnią się między sobą założeniem o równości macierzy kowariancji między klasami. W przypadku LDA mamy tylko jedną macierz kowariancji niezależną od klasy, natomiast w przypadku QDA mamy osobne macierze dla każdej z klas. W obu przypadkach rozkład a priori $p(k)$ estymujemy analogicznie jak w przypadku NBC, tj.

$$p(k) = \frac{1}{n}\sum_{i=1}^{n}[t_i = k]\,.$$

Właściwy klasyfikator jest również budowany zgodnie z bayesowską regułą MAP

$$\begin{aligned}\hat{t}(\boldsymbol{x}) &= \arg\max_k \log p(k \mid \boldsymbol{x}) \\ &= \arg\max_k \left(\log p(\boldsymbol{x} \mid k) + \log p(k)\right)\,.\end{aligned}$$

Rozwijając wyrażenie pod operatorem argmax otrzymujemy odpowiednio dla LDA (pomijając człony stałe)

$$\delta_k(\boldsymbol{x}) = \boldsymbol{x}^\intercal \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k - \frac{1}{2}\boldsymbol{\mu}_k^\intercal \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_k + \log p(k)$$

i QDA

$$\begin{aligned}\delta_k(\boldsymbol{x}) = &-\frac{1}{2}\log\det\boldsymbol{\Sigma}_k - \frac{1}{2}\boldsymbol{x}^\intercal \boldsymbol{\Sigma}_k^{-1}\boldsymbol{x} + \boldsymbol{x}^\intercal \boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k \\ &- \frac{1}{2}\boldsymbol{\mu}_k^\intercal \boldsymbol{\Sigma}_k^{-1}\boldsymbol{\mu}_k + \log p(k)\end{aligned}$$

Funkcję $\delta_k$ nazywamy ***funkcją dyskryminatora***. Widzimy również, skąd biorą się nazwy modeli – w przypadku LDA funkcja dyskryminatora jest funkcją liniową względem predyktorów, natomiast w przypadku QDA ma ona postać formy kwadratowej względem predyktorów. Uczenie modeli LDA i QDA polega na estymacji parametrów rozkładów warunkowych: $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}$ w przypadku LDA i $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ w przypadku QDA. Można pokazać, iż odpowiednimi estymatorami są

$$\boldsymbol{\mu}_k = \frac{1}{n_k}\sum_{i:t_i=k}\boldsymbol{x}_i$$

$$\boldsymbol{\Sigma} = \frac{1}{n-K}\sum_{k=1}^{K}\sum_{i:t_i=k}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\intercal$$

$$\boldsymbol{\Sigma}_k = \frac{1}{n_k - 1}\sum_{i:t_i=k}(\boldsymbol{x}_i - \boldsymbol{\mu}_k)(\boldsymbol{x}_i - \boldsymbol{\mu}_k)^\intercal$$

## 7.5 PCA

Analiza składowych głównych (ang. ***Principle Component Analysis***) to liniowa metoda redukcji wymiarowości, która może zostać wykorzystana zarówno w celach wizualizacji wysokowymiarowych danych, jak i do zmniejszenia liczby wymiarów celem ułatwionego ich późniejszego przetwarzania przez inne modele uczenia maszynowego. Zasadnicza idea

jest następująca, mamy macierz $\boldsymbol{X}$ wymiaru $n \times d$, gdzie $n$ to liczba przykładów, a $d$ to liczba cech. Wiemy, że dla każdej macierzy $\boldsymbol{X}$ wymiaru $n \times d$ ($n \geq d$) istnieje jej rozkład SVD

$$\boldsymbol{X} = \sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^\intercal + \ldots + \sigma_d \boldsymbol{u}_r \boldsymbol{v}_r^\intercal \,,$$

gdzie $r \leq \min\{n, d\}$ to rząd macierzy $\boldsymbol{X}$, $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r \geq 0$ – to wartości osobliwe, natomiast $\boldsymbol{u}_i$ to ortonormalne wektory kolumnowe wymiaru $n$, a $\boldsymbol{v}_i$ to ortonormalne wektory kolumnowe wymiaru $d$. Możemy w takim razie przybliżyć macierz $\boldsymbol{X}$ za pomocą kilku pierwszych wyrażeń, w szczególności dwóch

$$\boldsymbol{X} \approx \sigma_1 \boldsymbol{u}_1 \boldsymbol{v}_1^\intercal + \sigma_2 \boldsymbol{u}_2 \boldsymbol{v}_2^\intercal \,.$$

Zauważmy jednak teraz, iż wektory $\boldsymbol{u}_1$ i $\boldsymbol{u}_2$ możemy potraktować jako kolejne współrzędne 2-D naszego zbioru punktów.

## 7.6 PDP

Wykresy PDP (ang. Partial Dependence Plots) to prosta i intuicyjna metoda do wizualizacji wpływu danych predyktorów modelu uczenia maszynowego na jego odpowiedź. Jest to wykres funkcji częściowej zależności

$$f_S(\boldsymbol{x}_S) = \mathbb{E}_{X_{\{1,\ldots,p\}\setminus S}}[f(\boldsymbol{x}_S, X_{\{1,\ldots,p\}\setminus S})] \,,$$

gdzie $S$ jest badanym podzbiorem predyktorów, natomiast $f$ to funkcja modelu. Praktycznie stosowaną estymatę oblicza się uśredniając po zbiorze danych

$$\hat{f}_S(\boldsymbol{x}_S) = \frac{1}{n} \sum_{i=1}^{n} f(\boldsymbol{x}_S, \boldsymbol{x}^{(i)}_{\{1,\ldots,p\}\setminus S}) \,,$$

przy czym $\boldsymbol{x}^{(i)}_{\{1,\ldots,p\}\setminus S}$ są wartościami nieinteresujących cech obiektów ze zbioru danych. Zaletami wykresów częściowej zależności są w szczególności: intuicyjność, łatwość implementacj, przejrzystość interpretacji (o ile predyktory nie są istotnie skorelowane) oraz możliwą interpretację przyczynową. Ograniczenia PDP to m.in.

- Nie da się zwizualizować PDP dla więcej niż 2 cech jednocześnie.
- Jeśli predyktory są skorelowane, może występować uśrednianie po nierealistycznych danych.
- PDP mogą ukrywać efekty niejednorodne (np. uśrednienie powoduje wzajemne zniesienie efektów pochodzących od różnych części zbioru danych).

## 7.7 Klasteryzacja

Klasteryzacja to metoda uczenia nienadzorowanego, której celem jest grupowanie obiektów, aby odkryć, czy nasze dane mają jakąś strukturę. Typowo w danych występują klastry (ang. clusters), czyli gęściej zbite chmury punktów, które odpowiadają one obiektom o podobnych własnościach. Zastosowania metod klasteryzacji to m.in. analiza zbioru danych jako część eksploracyjnej analizy danych (EDA).

### 7.7.1 k-means

Ideą algorytmu klasteryzacji k-means jest istnienie reprezentantów (prototypów) danych group wokół których dane klastry są wycentrowane. Zakładamy, że znamy pożądaną liczbę klastrów $k$ i szukamy tylu centroidów, czyli średnich z punktów z danego klastra. Algorytm jest następujący

**Algorithm 5:** Algorytm k-średnich (Lloyda)

**Input:** Zbiór punktów $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$, liczba klastrów $k$
**Output:** Centroidy $C = \{c_1, \ldots, c_k\}$ oraz przypisanie punktów do klastrów

Wybierz losowo $k$ punktów z $X$ jako początkowe centroidy $C = \{c_1, \ldots, c_k\}$;
**repeat**
  **foreach** $x_i \in X$ **do**
    Przypisz $x_i$ do najbliższego centroidu:

$$\text{cluster}(x_i) = \underset{j \in \{1,\ldots,k\}}{\arg\min} \|x_i - c_j\|^2$$

  **foreach** $c_j \in C$ **do**
    Oblicz nową pozycję centroidu jako średnią punktów przypisanych do klastra $j$:

$$c_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$$

**until** ***iteration limit***;

Formalne założenia: klastry mają rozkład normalny, wycentrowany na centroidach; co więcej, rozkłady te są sferyczne, tzn. mają zerową kowariancję (są izotoniczne). Taki klaster może być jednoznacznie zdefiniowany przez centroid, czyli wartość średnią. k-means wymaga normalizacji danych, bo liczymy odległości. Występuje klątwa wymiarowości, typowo zwalczana z pomocą PCA. Jest czułe na outliery, główna wada algorytmu

### 7.7.2 Klastrowanie hierarchiczne

Idea: klastry tworzą punkty blisko siebie; zbiór może mieć wiele klastrów, o różnych rozmiarach, ale wszystkie powinny być gęste; duże klastry mogą się składać z mniejszych. Metody hierarchiczne tworzą hierarchię klastrów, łącząc mniejsze klastry w coraz większe. Trzeba wybrać, ile finalnie klastrów chcemy.

**Algorithm 6:** Hierarchiczna klasteryzacja aglomeracyjna (bottom-up)

**Input:** Zbiór punktów $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$
**Output:** Drzewo dendrogramowe opisujące proces łączenia klastrów

**Inicjalizacja:** Utwórz $n$ jednoelementowych klastrów $\mathcal{C} = \{\{x_1\}, \{x_2\}, \ldots, \{x_n\}\}$;
**while** ***liczba klastrów w*** $\mathcal{C} > 1$ **do**
  Oblicz macierz odległości $D(C_i, C_j)$ dla wszystkich $C_i, C_j \in \mathcal{C}$, $i \neq j$;
  Znajdź parę klastrów $(C_p, C_q)$ taką, że

$$(C_p, C_q) = \arg \min_{C_i \neq C_j} D(C_i, C_j)$$

  Połącz $C_p$ i $C_q$ w nowy klaster $C_{new} = C_p \cup C_q$;
  Zaktualizuj $\mathcal{C} := \mathcal{C} \setminus \{C_p, C_q\} \cup \{C_{new}\}$;
  Dodaj $(C_p, C_q, D(C_p, C_q))$ do struktury dendrogramu;

# 8 Ewaluacja modeli

## 8.1 Przygotowanie danych

Kluczem do uzyskania dobrych wyników przy korzystaniu z algorytmów uczenia maszynowego jest odpowiednie przygotowanie danych (ang. ***preprocessing***). Typowo preprocessing składa się z:

- eksploracji danych oraz wstępnego czyszczenia, w szczególności usunięcia jawnych wartości odstających (ang. ***outliers***) oraz cech posiadających zbyt dużo wartości brakujących;
- analizy rozkładu zmiennej docelowej oraz ewentualnej transformacji logarytmicznej, która poprawia stabilność numeryczną, gdy przewidywane wartości są dużymi dodatnimi liczbami rzeczywistymi, zmienia dziedzinę zmiennej objaśnianej z $\mathbb{R}_+$ na $\mathbb{R}$ oraz dodatkowo jest przykładem transformacji stabilizującej wariancję;
- ***podziału zbioru na część treningową oraz testową***;
- dokonania skalowania i imputacji brakujących wartości cech (metody `.fit()` wywołujemy jedynie dla zbioru treningowego);
- usunięcia silnie skorelowanych cech;
- zakodowania wartości kategorycznych za pomocą tzw. ***one–hot encoding*** pamiętając o ***dummy variable trap*** – jedną z $k$ kategorii kodujemy za pomocą wektora ***one–hot*** długości $n - 1$, aby uniknąć zależności liniowej między cechami (opcja `drop="first"` w `OneHotEncoder` w scikit-learn);
- wykonania feature engineering – dodania wielomianów cech do naszych danych lub skonstruowania innych cech (np. cech określających miesiąc, dzień itp.).

Podział zbioru na część treningową i testową jest najważniejszym etapem preprocessingu. Zbiór testowy wydzielamy, aby po wytrenowaniu modelu sprawdzić, jak poradzi on sobie na nowych, niewidzianych wcześniej danych. Powinniśmy go traktować jako dane, które będziemy w przyszłości dostawać po wdrożeniu modelu do realnego systemu. Takie dane również będziemy musieli przeskalować, zakodować itp., ale parametry potrzebne do wykonania tych transformacji możemy wziąć jedynie z dostępnego wcześniej zbioru treningowego. Wykorzystanie danych testowych w procesie treningu to ***błąd wycieku danych*** (ang. ***data leakage***). Skutkuje on niepoprawnym, nadmiernie optymistycznym oszacowaniem jakości modelu.

## 8.2 Metryki do oceny regresji i klasyfikacji

Zasadniczo, aby ocenić predykcje modelu używamy odpowiednich metryk, których wartości określają jak dobry jest model.

W przypadku regresji najczęściej używanymi metrykami są RMSE (ang. ***Root Mean Squared Error***) oraz MAE (ang. ***Mean Absolute Error***) zdefiniowane odpowiednio jako

$$\text{RMSE} := \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

$$\text{MAE} := \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

Metryki te mają jednakową jednostkę jak predykcje. Jeśli chcielibyśmy mieć liczbę względną określającą jakość modelu to mamy do dyspozycji metryki MAPE

(ang. ***Mean Absolute Percentage Error***) oraz SMAPE (ang. ***Symmetric Mean Absolute Percentage Error***) zdefiniowane odpowiednio jako

$$\text{MAPE} := \frac{1}{n}\sum_{i=1}^{n}\left|\frac{y_i - \hat{y}_i}{y_i}\right|$$
$$\text{SMAPE} := \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{|y_i| + |\hat{y}_i|}$$

Obie te metryki mają zakres od 0 do 1, przy czym niższa wartość oznacza lepszy model. Metryki te mają jednak szereg problemów, z których najpoważniejsze to: problemy, gdy wartości są bliskie 0, asymetryczne traktowanie predykcji za dużych oraz za małych. Z tych powodów znacznie lepszą względną metryką jest MASE (ang. ***Mean Absolute Scaled Error***)

$$\text{MASE} := \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\sum_{i=1}^{n}|y_i - \overline{y}|},$$

gdzie $\overline{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$. Metryka MASE jest zatem względnym błędem MAE jaki popełnia nasz model w stosunku do modelu naiwnego, który przewiduje zawsze wartość średnią.

W przypadku zadania klasyfikacji binarnej naszym celem dla danego wektora cech jest zwrócenie jednej z dwóch klas, które będziemy nazywać klasą pozytywną i negatywną. O ile w przypadku regresji pomiar jakości modelu był całkiem prosty, o tyle w przypadku klasyfikacji sytuacja jest nieco bardziej skomplikowana. Zauważmy bowiem, iż mamy 4 możliwości odpowiedzi klasyfikatora

- ***True Positive (TP)*** – poprawnie zaklasyfikowaliśmy klasę pozytywną jako pozytywną
- ***True Negative (TN)*** – poprawnie zaklasyfikowaliśmy klasę negatywną jako negatywną
- ***False Positive (FP)*** – niepoprawnie zaklasyfikowaliśmy klasę negatywną jako pozytywną
- ***False Negative (FN)*** – niepoprawnie zaklasyfikowaliśmy klasę pozytywną jako negatywną

Na podstawie ilości TP, TN, FP i FN w zbiorze testowym możemy wykreślić tzw. ***macierz pomyłek*** (ang. ***confusion matrix***) pokazującą ilość każdej z możliwości. Następnie możemy obliczyć różne stosunki tych wartości, aby uzyskać różne metryki. Najbardziej standardowymi są ***accuracy***, ***precision*** oraz ***recall*** (lub inaczej sensitivity) zdefiniowane jako

$$\text{Accuracy} := \frac{\text{TP} + \text{TN}}{n}$$
$$\text{Precision} := \frac{\text{TP}}{\text{TP} + \text{FP}}$$
$$\text{Recall} := \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Wartość accuracy mówi po prostu jaki stosunek przykładów został poprawnie zaklasyfikowany (zauważmy tutaj, że $\text{TP} + \text{TN} + \text{FP} + \text{FN} = n$). Nie jest to jednak dobra miara jakości, gdy nasz zbiór jest niezbalansowany, tj. zawiera więcej przykładów określonej klasy.

Wartość precision określa jak pewny jest klasyfikator przy wykrywaniu klasy pozytywnej, natomiast recall mówi o tym jak dobrze klasyfikator wyławia przykłady pozytywne. Zauważmy jednak, iż nie możemy stosować żadnej z tych metryk w odosobnieniu. Istotnie klasyfikator, który zwraca zawsze klasę pozytywną ma maksymalny recall, a klasyfikator, który zwraca zawsze klasę negatywną ma nieokreślony precision (i jest oczywiście beznadziejnym klasyfikatorem). Musimy więc zawsze ewaluować model na obu tych metrykach i jedynie dobry wynik obu z nich mówi o jakości klasyfikatora. Oczywiście czasami chcielibyśmy określić jakość modelu za pomocą jednej liczby, a niekoniecznie sprawdzać zawsze macierz pomyłek (choć jest to bardzo użyteczne) lub podawać wartości dwóch metryk. Metryką, która łączy precision i recall jest $F_\beta$–***score*** zdefiniowany jako

$$F_\beta := (1 + \beta^2)\frac{\text{Precision} \cdot \text{Recall}}{\beta^2 \cdot \text{Precision} + \text{Recall}},$$

gdzie $\beta$ określa ile razy bardziej ważny jest recall od precision. Typowo używa się $F_1$–score

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

Wiele klasyfikatorów oprócz twardych predykcji zwraca również rozkład prawdopodobieństwa nad klasami. W przypadku klasyfikacji binarnej jest to oczywiście rozkład zero-jedynkowy z parametrem $p$ określającym prawdopodobieństwo klasy pozytywnej dla danego wektora cech. Standardowo oczywiście twardą predykcją jest ta z klas, która ma większe prawdopodobieństwo, czyli (co równoważne) predykcją jest klasa pozytywna jeśli $p > 0.5$. W niektórych problemach chcemy jednak zmienić ten próg i dokonać tzw. ***threshold tuning***. Wykresem, który pozwala na dokonanie tuningu progu jest ***krzywa ROC*** (ang. ***Receiver Operatic Characteristic curve***), która jest krzywą parametryczną wyznaczoną przez punkty

(FPR(threshold), TPR(threshold)) dla progów z zakresu $[0; 1]$, gdzie

$$\mathrm{TPR} := \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}, \quad \mathrm{FPR} := \frac{\mathrm{FP}}{\mathrm{FP} + \mathrm{TN}}.$$

Metryką niezależną od wybranego progu jest tzw. ***AUROC*** (ang. ***Area under ROC curve***) zdefiniowany jako pole powierzchni pod krzywą ROC dla danego klasyfikatora. Zauważmy, że klasyfikator losowy, który zwraca zawsze klasę pozytywną z prawdopodobieństwem równym wartości progu ma wartość AUROC równą 0.5, natomiast idealny klasyfikator, który niezależnie od wartości progu klasyfikuje wszystkie przykłady poprawnie ma AUROC równy 1.

Inną analogiczną metryką jest ***AUPRC***, gdzie zamiast krzywej ROC stosujemy krzywą ***PRC*** (z ang. ***Precision–Recall Curve***), w której zamiast TPR i FPR używamy odpowiednio Precision i Recall. Metryka AUPRC jest często wykorzystywana w przypadku klasyfikacji ekstremalnie niezbalansowanej, w której mamy bardzo mało ($< 1\%$) klasy pozytywnej.

W przypadku klasyfikacji wieloklasowej używamy zasadniczo takich samych metryk jak w klasyfikacji binarnej, ale wprowadzamy mikro i makro uśrednianie (ang. ***micro/macro-averaging***). Przez $\mathrm{TP}_k$ będziemy rozumieć liczbę prawidłowo zaklasyfikowanych przykładów z klasy $k$, $\mathrm{FP}_k$ to liczba przykładów z innych klas, które zaklasyfikowaliśmy nieprawidłowo jako $k$–tą klasę, $\mathrm{FN}_k$ to liczba przykładów z klasy $k$, które zaklasyfikowaliśmy jako inną klasę. Wówczas odpowiednie metryki mają postać

$$\mathrm{MicroPrecision} := \frac{\sum_k \mathrm{TP}_k}{\sum_k \mathrm{TP}_k + \sum_k \mathrm{FP}_k},$$

$$\mathrm{MacroPrecision} := \frac{1}{K} \sum_{k=1}^{K} \frac{\mathrm{TP}_k}{\mathrm{TP}_k + \mathrm{FP}_k}$$

oraz

$$\mathrm{MicroRecall} := \frac{\sum_k \mathrm{TP}_k}{\sum_k \mathrm{TP}_k + \sum_k \mathrm{FN}_k},$$

$$\mathrm{MacroRecall} := \frac{1}{K} \sum_{k=1}^{K} \frac{\mathrm{TP}_k}{\mathrm{TP}_k + \mathrm{FN}_k}.$$

W przypadku klasyfikacji wieloklasowej macierz pomyłek jest macierzą wymiaru $K \times K$, gdzie $K$ jest liczbą klas.

## 8.3 Tuning hiperparametrów i walidacja skrośna

Praktycznie wszystkie modele uczenia maszynowego mają hiperparametry, często liczne, które w zauważalny sposób wpływają na wyniki, a szczególnie na underfitting i overfitting. Ich wartości trzeba dobrać zatem dość dokładnie. Proces doboru hiperparametrów nazywa się tuningiem hiperparametrów (ang. ***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 (ang. ***grid search***) – definiujemy możliwe wartości dla różnych hiperparametrów, a metoda sprawdza ich wszystkie możliwe kombinacje (czyli siatkę),
- losowe przeszukiwanie (ang. ***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 wyciek danych, 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 (ang. ***validation set***). Zbiór taki wycina się ze zbioru treningowego.

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 pesymistyczne, 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 (ang. ***cross validation***, CV). Polega na tym, że dzielimy zbiór treningowy 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 ewentualnie obliczyć odchylenie standardowe). Takie wyniki są znacznie bardziej wiarygodne.

Walidacja krzyżowa bez jednego (ang. ***Leave-One-***

***Out Cross-Validation***, LOOCV) jest metodą iteracyjną, której kroki wyglądają następująco:

- wybieramy kolejny element zbioru obserwacji $(\boldsymbol{x}_i, y_i)$;
- zbiorem walidacyjnym jest $\{(\boldsymbol{x}_i, y_i)\}$, a zbiorem uczącym reszta obserwacji;
- metodą zbioru walidacyjnego wyliczamy błąd walidacyjny $E_i$.

Na koniec obliczamy estymatę błędu testowego

$$CV_{(n)} = \frac{1}{n}\sum_{i=1}^{n} E_i\,.$$

Korzyściami z zastosowania LOOCV jest redukcja obciążenia – uczenie modelu korzysta w każdym kroku z $n-1$ obserwacji – w konsekwencji ewentualne przeszacowanie błędu treningowego jest nieznaczne. Dodatkowo metoda jest ***deterministyczna*** – wielokrotne uruchomienie daje ten sam wynik.

***Bootstrap*** to metoda statystyczna stosowana do oszacowania niepewności związanej z danym estymatorem lub metodą uczenia statystycznego. Łatwo stosuje się w sytuacjach, w których nie ma innych – lepiej uzasadnionych teoretycznie – metod oszacowania zmienności. Mamy do dyspozycji zbiór danych $\mathcal{X}$ o wielkości $n$, przy pomocy którego wyznaczamy estymatę $\hat{\alpha}$ pewnej statystyki $\alpha$. Zadanie polega na estymowaniu błędu standardowego $\hat{\alpha}$, tzn. $\sigma(\hat{\alpha})$ przy braku możliwości uzyskania nowych danych. Konstruujemy zbiory danych bootstrap $\mathcal{X}_1, \ldots, \mathcal{X}_B$ o wielkości $n$ przez ***losowanie ze zwracaniem*** z $\mathcal{X}$ dla pewnego dużego $B$ i na każdym z $\mathcal{X}_b$ obliczamy estymatę $\hat{\alpha}_b$. Ostatecznie błąd standardowy (tzw. ***błąd typu bootstrap***) ma postać

$$\begin{aligned} m(\hat{\alpha}) &= \frac{1}{B}\sum_{b=1}^{B} \hat{\alpha}_b \\ \sigma(\hat{\alpha}) &= \sqrt{\frac{1}{B-1}\sum_{b=1}^{B} \left(\hat{\alpha}_b - m(\hat{\alpha})\right)^2}\,. \end{aligned}$$