Uczenie maszynowe w bioinformatyce

Wykład 8: klasyfikacja

Tymon Rubel

Zakład Elektroniki Jądrowej i Medycznej
Instytut Radioelektroniki i Technik Multimedialnych PW

Etapy przetwarzania i analizy danych

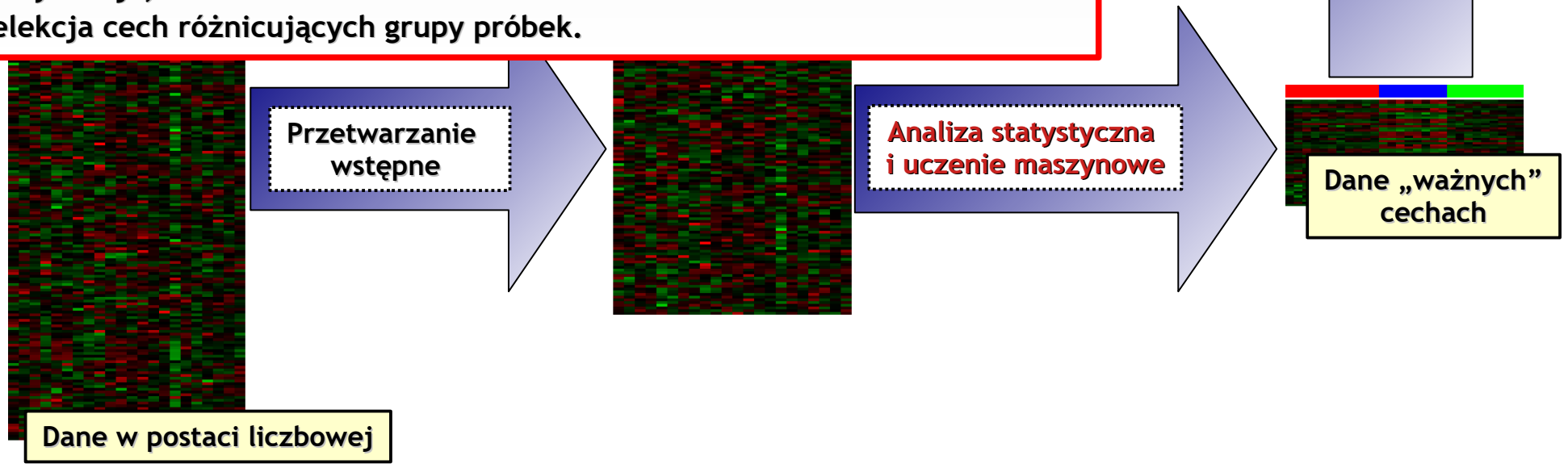
Wybór **metod statystycznych i uczenia się maszyn** używanych na etapie analizy danych jest ściśle uzależniony od celu badań. Stosowane tutaj techniki można w ogólności podzielić na **działające z nadzorem** lub **nienadzorowane**.

Metody działające bez nadzoru stosuje się do wykrywania zależności pomiędzy próbkami i/lub cechami jedynie w oparciu o dane, bez używania dodatkowych informacji (w tym też nie uwzględniając przynależności badanych obiektów do zadanych z góry klas). Do metod nienadzorowanych zaliczają się m. in.:

- ✓ klasteryzacja;
- ✓ techniki redukcji wymiarowości.

Metody nadzorowane korzystają z danych i dodatkowej wiedzy, dotyczącej np. oczekiwanych wartości na wyjściu lub sposobu przypisania próbek do znanych klas. Przykładami zagadnień, w których używa się takich metod są:

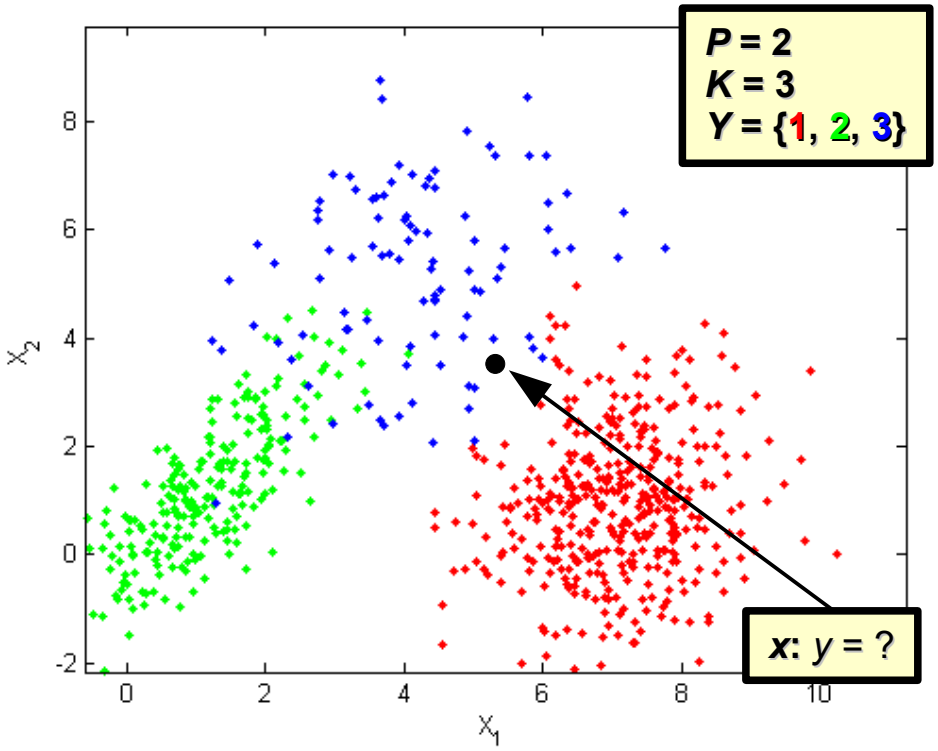
- klasyfikacja;
- selekcja cech różnicujących grupy próbek.



Klasyfikacja: przedstawienie problemu

W P-wymiarowej przestrzeni cech χ (dla danych z mikromacierzy $\chi \subseteq \Re^P$) występują wektory reprezentujące obiekty należące do K rozłącznych klas (grup, populacji) ω_k. Klasom ω_k przypisane są etykiety liczbowe ze zbioru $Y = \{1, 2, ..., K\}$.

Każdy z obiektów opisywany jest przez P-wymiarowy wektory cech $x \in \chi$ oraz zmienną $y \in Y$, reprezentującą przynależność obiektu do jednej z klas.



Zadanie klasyfikacji polega na przypisaniu nowo zaobserwowanego obiektu do jednej z góry ustalonych klas. Przypisanie to obywa się w oparciu o wartości wektora cech opisującego klasyfikowany obiekt. Innymi słowy, klasyfikację można przedstawić jako predykcję etykiety klasy na podstawie wektora cech.

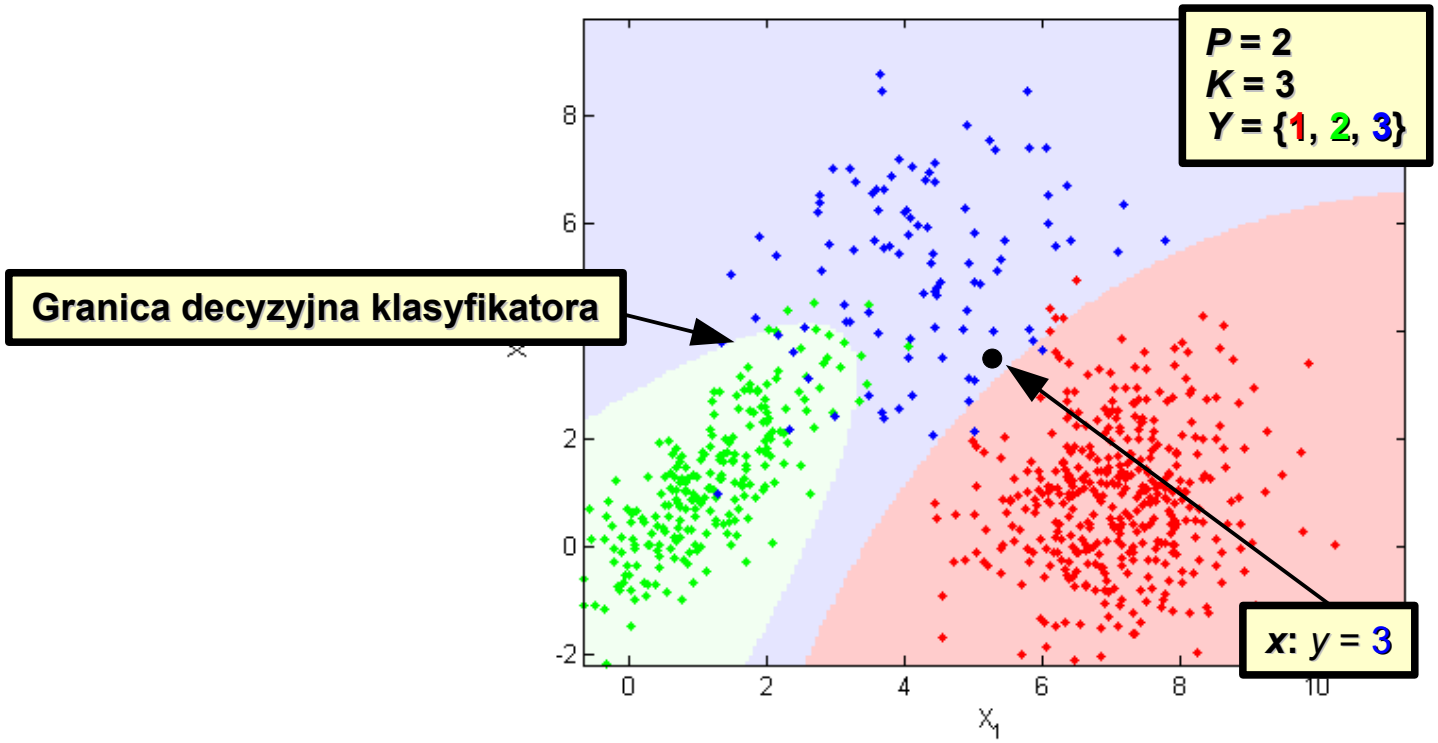
Klasyfikacja: przedstawienie problemu

Klasyfikator (reguła klasyfikacyjna) jest to funkcja $d(x)$, określona na przestrzeni cech χ, o wartościach w zbiorze etykiet klas Y:

$$d(x): \quad \chi \rightarrow Y = \{1, 2, \ldots, K\}$$

która nowo zaobserwowanemu wektorowi cech x przypisuje etykietę y:

$$y = d(x)$$



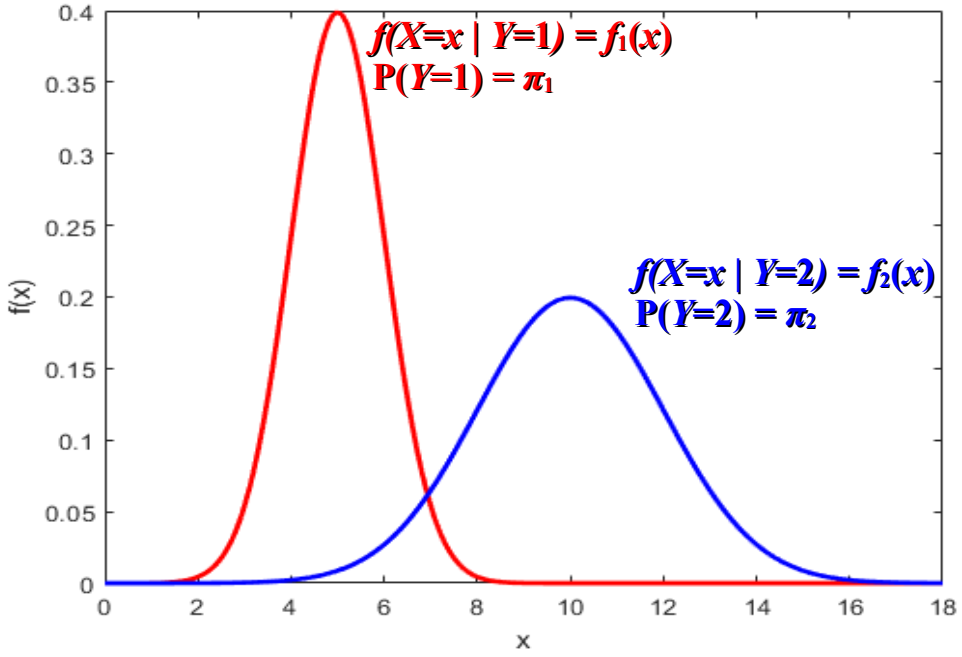
Oczekuje się przy tym, że dokonana przez klasyfikator predykcja będzie możliwie dokładna. Poszukuje się więc funkcji $d(x)$, która minimalizuje prawdopodobieństwo błędu $e(d)$, polegającego na przypisaniu etykiety $y = k$, podczas gdy w rzeczywistości wektor cech x nie należy do k-tej klasy.

Klasyfikacja: optymalny klasyfikator Bayesa

Klasyfikacja może być przedstawiona jako zagadnienie z zakresu statystycznej teorii decyzji. W takim podejściu każda z K klas opisywana jest przez:

- **prawdopodobieństwo klasy** $P(Y=k)=\pi_k$, określające prawdopodobieństwo obserwacji obiektów pochodzących z k-tej klasy $(\pi_1+\pi_2+\ldots+\pi_K=1)$;

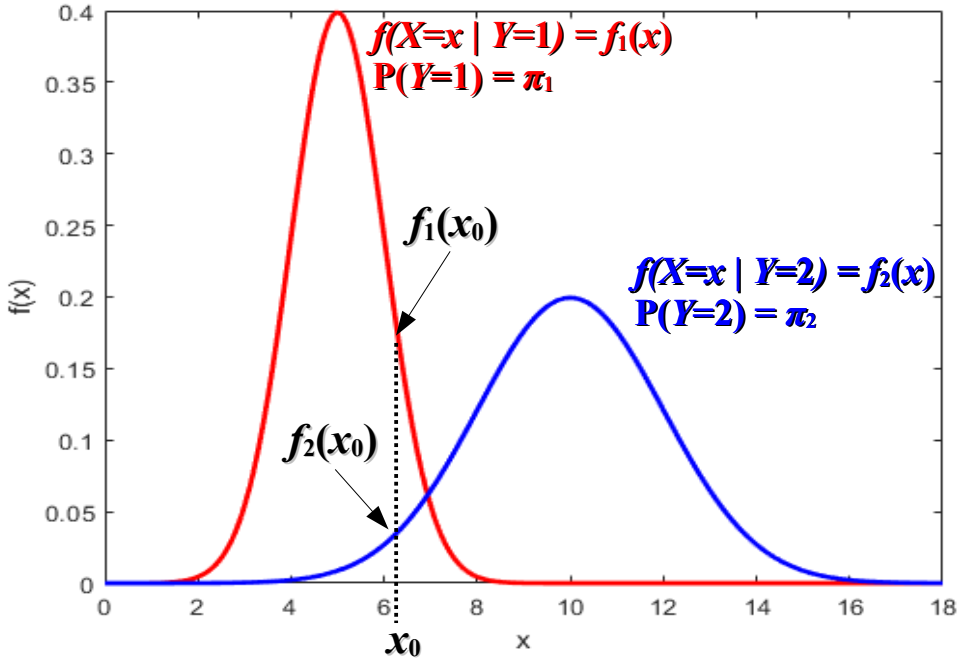
- **rozkład prawdopodobieństwa wektorów cech klasy** $f(X=x \mid Y=k)=f_k(x)$: P-wymiarowa funkcja gęstości prawdopodobieństwa wektorów cech obiektów z klasy k.



Wielkości te określa się jako ***a priori***, gdyż są danymi z góry właściwościami klas. Tym samym, są one niezależne od aktualnie klasyfikowanego wektora x (są takie same przed i po obserwacji tego wektora).

Klasyfikacja: optymalny klasyfikator Bayesa

Klasyfikacja odbywa w oparciu o prawdopodobieństwo, że wektor cech x należy do k-tej klasy $\mathrm{P}(Y=k|X=x)=\mathrm{P}(k|x)$. Nazywamy je *a posteriori*, gdyż jest zależne od wektora poddawanego klasyfikacji i może być określone dopiero po obserwacji tego wektora.



Prawdopodobieństwo *a posteriori* powiązane jest z prawdopodobieństwami *a priori* poprzez regułę Bayesa:

$$\mathrm{P}(Y=k|X=\boldsymbol{x})=\mathrm{P}(k|\boldsymbol{x})=\frac{\mathrm{P}(Y=k)\,f(X=\boldsymbol{x}|Y=k)}{\sum_{i=1}^{K}\mathrm{P}(Y=i)\,f(X=\boldsymbol{x}|Y=i)}=\frac{\pi_k f_k(\boldsymbol{x})}{\sum_{i=1}^{K}\pi_i f_i(\boldsymbol{x})}$$

Klasyfikacja: optymalny klasyfikator Bayesa

Gdy dla każdej z K klas znane są prawdopodobieństwa *a priori* $\pi_k = \mathrm{P}(Y=k)$ oraz warunkowe rozkłady *a priori* $f_k(x) = f(x|Y=k)$ optymalnym rozwiązaniem problemu klasyfikacji jest przypisanie wektora cech x do klasy, dla której obserwujemy największa wartość prawdopodobieństwa *a posteriori* $P(Y=k|x)$, wyznaczonego bezpośrednio z reguły Bayesa.

Klasyfikatorem Bayesa nazywamy klasyfikator $d_B(x)$ o regule decyzyjnej w postaci:

$$d_B(\boldsymbol{x}) = \underset{k}{\operatorname{argmax}}\, \mathrm{P}(k|\boldsymbol{x}) = \underset{k}{\operatorname{argmax}}\, \frac{\pi_k f_k(\boldsymbol{x})}{\sum_{i=1}^{K} \pi_i f_i(\boldsymbol{x})}$$

gdzie $\underset{k}{\operatorname{argmax}}(A)$ oznacza taką wartość k, które maksymalizuje wyrażenie A.

Występująca w mianowniku reguły Bayesa suma ma jednakową wartość dla każdej klasy, stąd też reguła decyzyjna klasyfikatora Bayesa sprowadza się do wyrażenia:

$$d_B(\boldsymbol{x}) = \underset{k}{\operatorname{argmax}}\, \pi_k f_k(\boldsymbol{x})$$

Klasyfikacja: klasyfikatory rzeczywiste i uczenie się pod nadzorem

Klasyfikator Bayesa d_B jest optymalny: jego błąd $e(d_B)$ jest zawsze nie większy od błędu $e(d)$ jakiegokolwiek innego klasyfikatora d.

W rzeczywistości jednak, niezwykle rzadko możliwe jest skonstruowanie idealnego klasyfikatora bayesowskiego, co ma dwie podstawowe przyczyny:

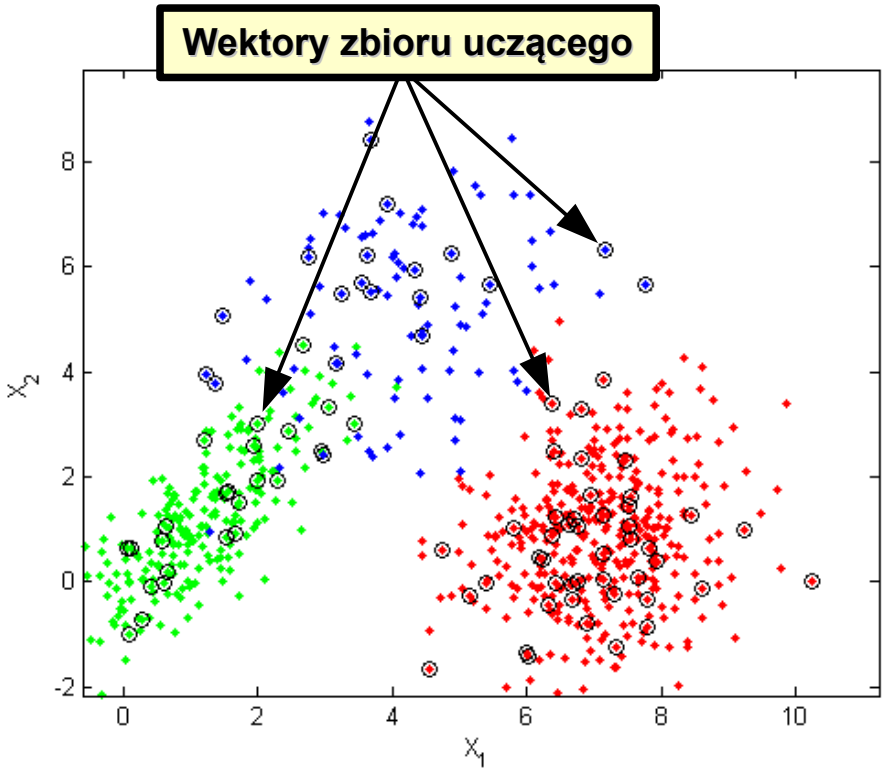
- **zwykle nie dysponujemy pełna wiedzą o wszystkich obiektach wchodzących w skład badanych K klasach, a jedynie zbiorem K prób losowych o liczebnościach $N_1, ..., N_K$ pobranych z poszczególnych klas;**
- **najczęściej nie znamy rozkładów prawdopodobieństwa wektorów cech w klasach $f_k(\boldsymbol{x}) = f(\boldsymbol{x} \mid Y=k)$ oraz rzeczywistych prawdopodobieństw występowania klas $\pi_k = \mathrm{P}(Y=k)$.**

Klasyfikacja: klasyfikatory rzeczywiste i uczenie się pod nadzorem

W praktyce zwykle konieczne jest przedstawienie klasyfikacji jako problemu uczenia się pod nadzorem (na przykładach, z nauczycielem, *supervised learnig*), którego celem jest skonstruowanie klasyfikatora w oparciu o nabytą wcześniej wiedzę o badanych populacjach.

Wiedza o badanych klasach ma postać zbioru uczącego (próby uczącej), złożonego z N wektorów cech o znanej przynależności do klas. Elementami zbioru uczącego L_N są więc przykłady, będące parami losowymi (x_{ki}, y_k), w których x_{ki} jest i-tym wektorem cech z k-tej klasy, a y_k jest etykietą k-tej klasy:

$$L_N = \{(x_{11}, y_1), (x_{12}, y_1), \ldots, (x_{1N_1}, y_1), \ldots, (x_{K1}, y_K), (x_{K2}, y_K), \ldots, (x_{K N_K}, y_K)\} \qquad N = N_1 + N_2 + \ldots + N_K$$

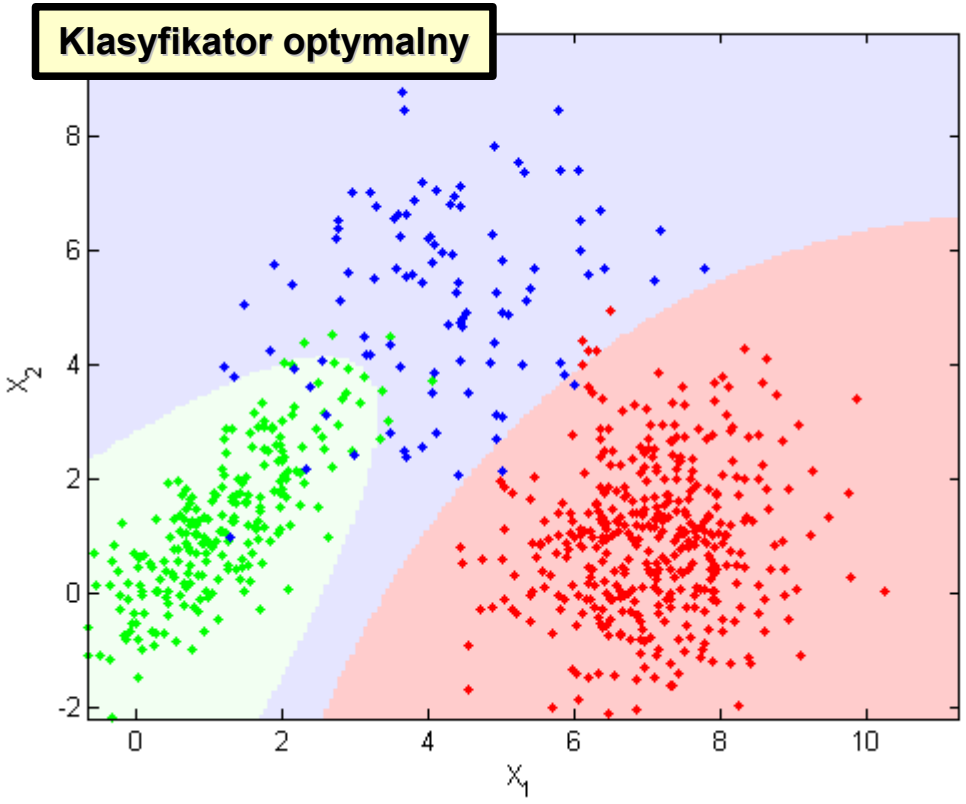


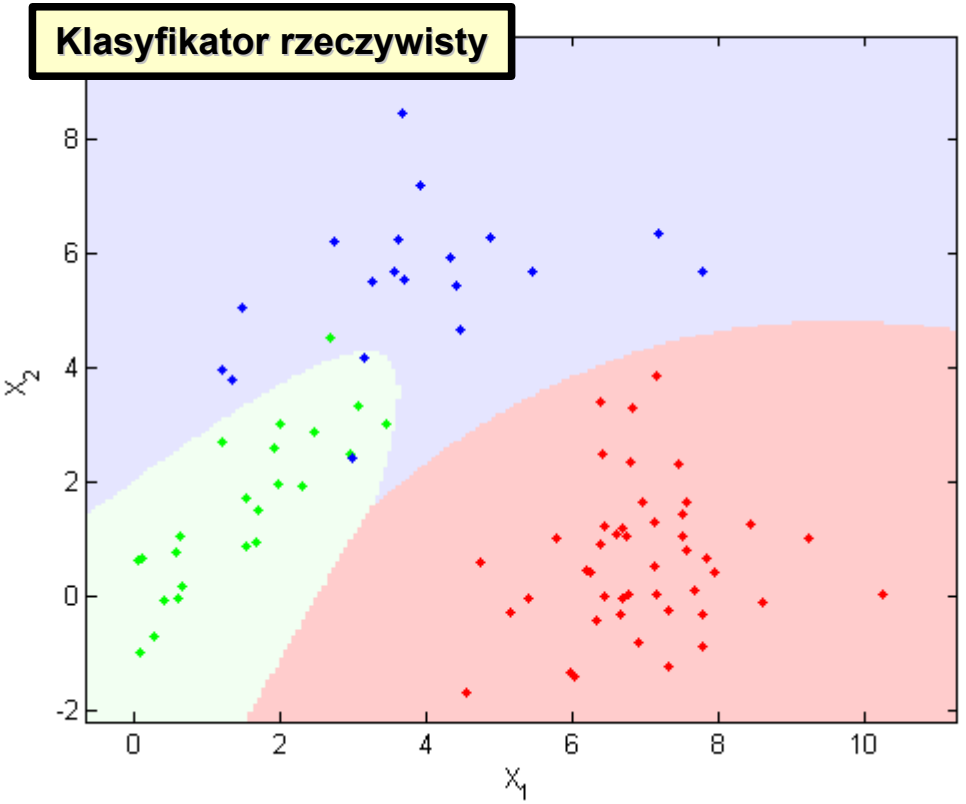
Klasyfikacja: klasyfikatory rzeczywiste i uczenie się pod nadzorem

Przy założeniu, że zbiór uczący jest reprezentatywny dla populacji (tzn. rozkłady występujących w nim wektorów cech są w przybliżeniu takie, jak rozkłady w całej populacji) możliwe jest zbudowanie na jego podstawie rzeczywistego klasyfikatora:

$$\hat{d}(\boldsymbol{x}) = \hat{d}(\boldsymbol{x}|L_N): \quad \chi \rightarrow Y = \{1, 2, \ldots, K\}$$

którego wynik $\hat{y} = \hat{d}(x|L_N)$ będzie zmienną losową zależną od losowej próby uczącej, o wartości będącej przybliżeniem predykcji jakiej dokonałby dla wektora cech x klasyfikator idealny. Celem jest więc zbudowanie klasyfikatora rzeczywistego o błędzie możliwie bliskim poziomowi błędu $e(d_B)$ optymalnego klasyfikatora Bayesa.

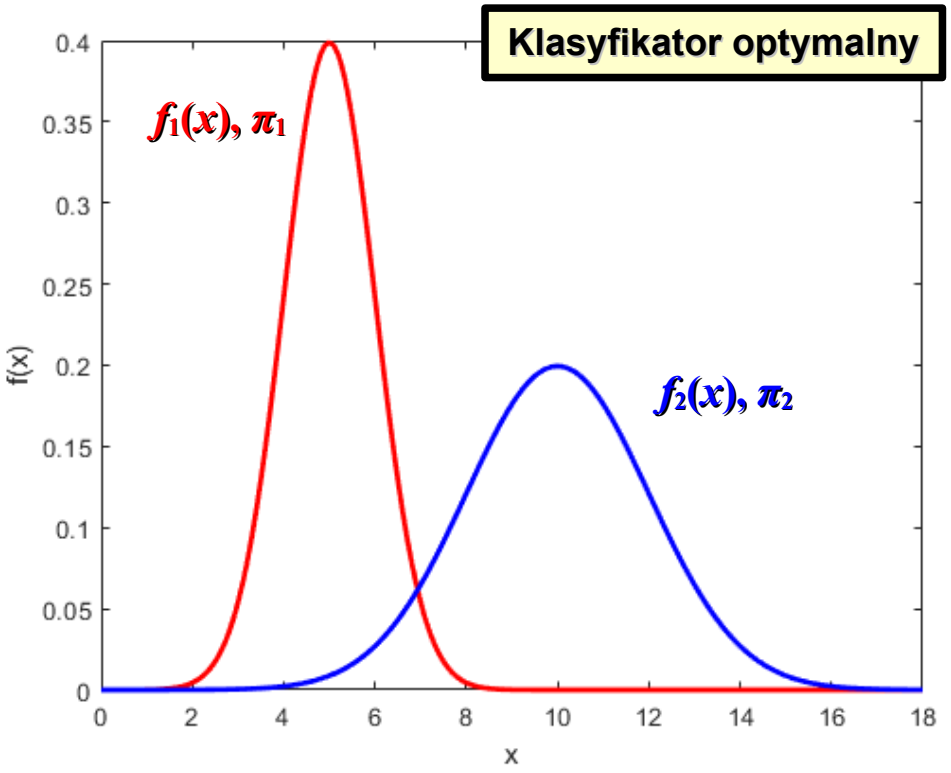


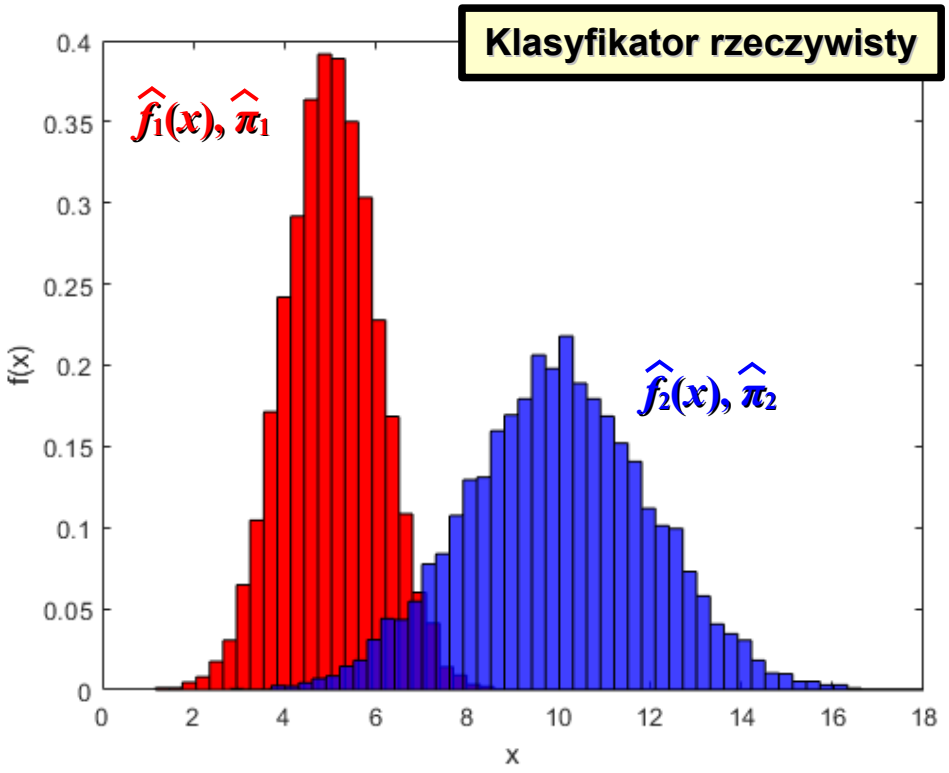


Klasyfikacja: klasyfikatory rzeczywiste i uczenie się pod nadzorem

Klasyfikatory rzeczywiste – tak jak idealny klasyfikator Bayesa – maksymalizują prawdopodobieństwo $P(k|x)$, które jednak musi być estymowane na podstawie próby uczącej. Reguła decyzyjna klasyfikatora rzeczywistego ma więc postać:

$$\hat{d}(\boldsymbol{x}) = \underset{k}{\operatorname{argmax}} \; \hat{P}(k|\boldsymbol{x}) = \underset{k}{\operatorname{argmax}} \; \hat{\pi}_k \hat{f}_k(\boldsymbol{x})$$





Klasyfikacja: klasyfikatory rzeczywiste i uczenie się pod nadzorem

Można wyróżnić dwa podstawowe podejścia do problemu budowania klasyfikatorów rzeczywistych zgodnych z powyższą regułą:

- **Pośrednia estymacja $\hat{\mathrm{P}}(k|x)$ poprzez estymację z próby uczącej rozkładów $\hat{f}_k(x)$ (najczęściej przez założenie z góry pewnej ich postaci) i prawdopodobieństw $\hat{\pi}_k$ (jako częstości występowania w próbie uczącej obserwacji z poszczególnych klas). Przy danych $\hat{f}_k(x)$ i $\hat{\pi}_k$ estymata $\hat{\mathrm{P}}(k|x)$ obliczana jest z reguły Bayesa.**

  Przykładami tego podejścia są gaussowski klasyfikator Bayesa i jego uproszczone wersje: liniowa analiza dyskryminacyjna, naiwny gaussowski klasyfikator Bayesa oraz klasyfikatory minimalnej odległości.

- **Bezpośrednia estymacja $\hat{\mathrm{P}}(k|x)$ w oparciu o próbę uczącą, bez zakładania postaci rozkładów prawdopodobieństwa cech w klasach.**

  Tutaj przykładami mogą być: metoda K-najbliższych sąsiadów oraz klasyfikatory wykorzystujące drzewa decyzyjne, sieci neuronowe i metodę wektorów nośnych.

Uczenie maszynowe w bioinformatyce

Wykład 8: gaussowskie klasyfikatory Bayesa

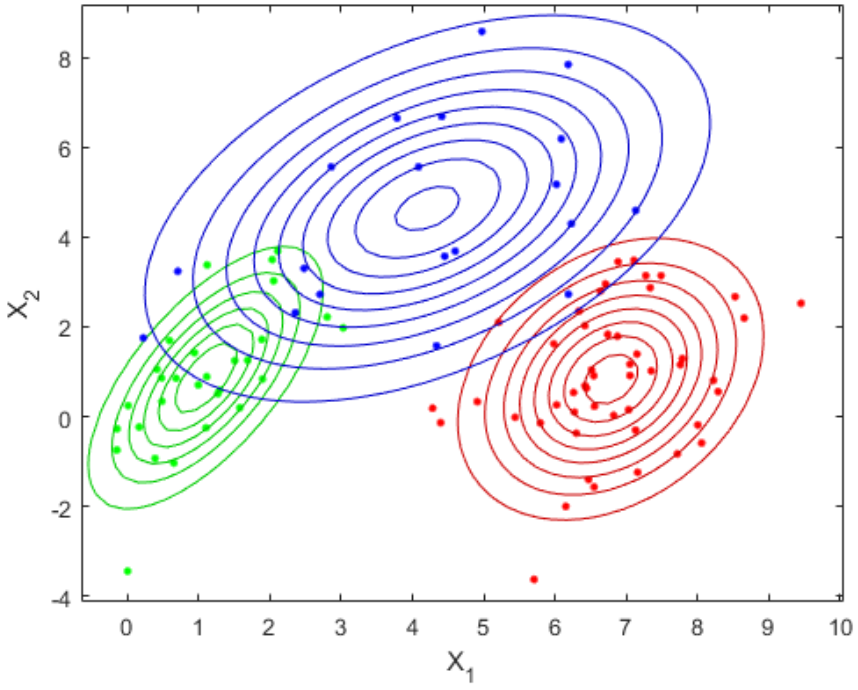
Klasyfikacja: gaussowskie klasyfikatory Bayesa

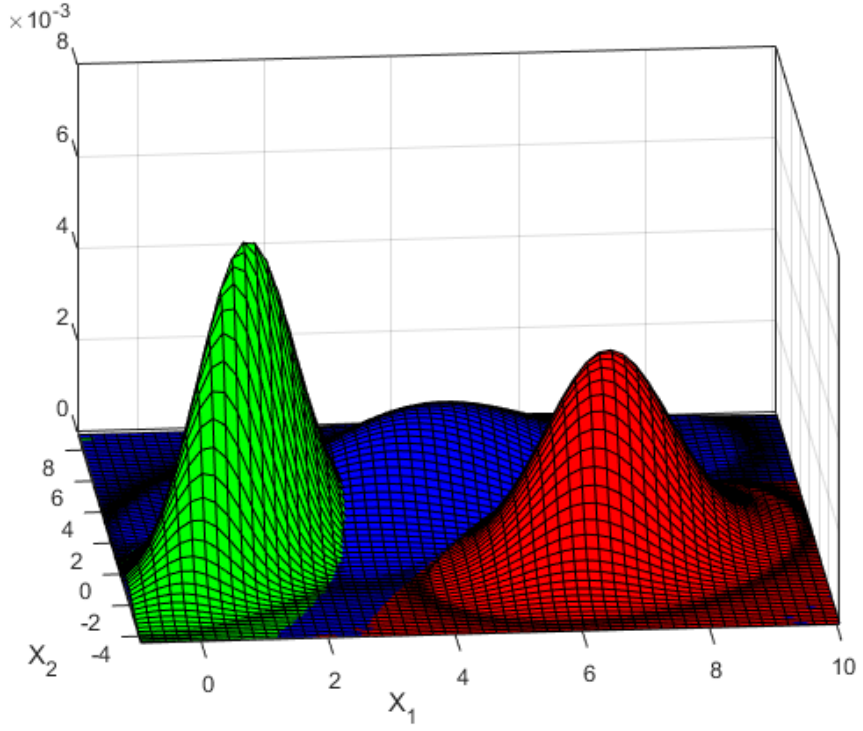
Najprostszym sposobem estymacji warunkowego rozkładu *a priori* $\hat{f}_k(x)$ k-tej klasy jest przyjęcie założeń co od postaci jego funkcji gęstości prawdopodobieństwa.

Przykładowo, można przyjąć, że w każdej z klas $\hat{f}_k(x)$ jest zgodny z funkcją gęstości prawdopodobieństwa P-wymiarowego rozkładu normalnego $N(\mu_k, \Sigma_k)$:

$$\hat{f}_k(\boldsymbol{x}; \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) = \frac{1}{(2\pi)^{P/2} |\boldsymbol{\Sigma}_k|^{1/2}} \exp\left(-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_k) \right)$$

gdzie μ_k jest wektorem wartości średnich w k-tej klasie, Σ_k jest macierzą kowariancji cech w k-tej klasie, a $|\Sigma_k|$ oznacza wyznacznik tej macierzy.





Klasyfikacja: gaussowskie klasyfikatory Bayesa

Nieznane parametry μ_k i Σ_k rozkładów normalnych w klasach zastępowane są poprzez ich estymatory wyznaczane z próby uczącej, przy czym dla k-tej klasy uwzględniane są jedynie te wektory uczące, które do niej należą:

$$\hat{\boldsymbol{\mu}}_k = \bar{\boldsymbol{x}}_k = \frac{1}{N_k}\sum_{j=1}^{N_k} \boldsymbol{x}_{kj} = \begin{bmatrix} \hat{\mu}_{k1} \\ \vdots \\ \hat{\mu}_{kP} \end{bmatrix} = \begin{bmatrix} \frac{1}{N_k}\sum_{j=1}^{N_k} x_{k1j} \\ \vdots \\ \frac{1}{N_k}\sum_{j=1}^{N_k} x_{kPj} \end{bmatrix}$$

$$\hat{\boldsymbol{\Sigma}}_k = \boldsymbol{S}_k = \frac{1}{N_k - 1}\sum_{j=1}^{N_k} (\boldsymbol{x}_{kj} - \bar{\boldsymbol{x}}_k)(\boldsymbol{x}_{kj} - \bar{\boldsymbol{x}}_k)^T = \frac{1}{N_k - 1}(\boldsymbol{X}_k - \bar{\boldsymbol{x}}_k)(\boldsymbol{X}_k - \bar{\boldsymbol{x}}_k)^T$$

Nieznane prawdopodobieństwo *a priori* k-tej klasy $\hat{\pi}_k$ może być estymowane jako częstość występowania reprezentujących ją wektorów w próbie uczącej:

$$\hat{\pi}_k = \frac{N_k}{N}$$

Klasyfikacja: gaussowskie klasyfikatory Bayesa

Bayesowska reguła klasyfikacyjna pozostaje bez zmian: poszukujmy klasy, dla której iloczyn $\hat{f}_k(x)\hat{\pi}_k$ ma maksymalną wartość. Jednak ze względu na postać funkcyjną rozkładu normalnego korzystnie jest ten iloczyn zlogarytmować:

$$\hat{d}(\boldsymbol{x}) = \underset{k}{\operatorname{argmax}} \ \ln(\hat{\pi}_k \hat{f}_k(\boldsymbol{x})) = \underset{k}{\operatorname{argmax}} \ \delta_k(\boldsymbol{x})$$

Wówczas klasyfikacja sprowadza się do policzenia dla każdej klasy wartości funkcji dyskryminacyjnej $\delta_K(x)$. Dla k-tej klasy wyznaczana jest ona ze wzoru:

$$\delta_k(\boldsymbol{x}) = \ln(\hat{\pi}_k \hat{f}_k(\boldsymbol{x})) = -\frac{1}{2}(\boldsymbol{x}-\bar{\boldsymbol{x}}_k)^T \boldsymbol{S}_k^{-1}(\boldsymbol{x}-\bar{\boldsymbol{x}}_k) - \frac{1}{2}\ln|\boldsymbol{S}_k| + \ln\frac{N_k}{N}$$

$$\hat{f}_k(\boldsymbol{x}) = \frac{1}{(2\pi)^{P/2}|\boldsymbol{S}_k|^{1/2}} \exp\left(-\frac{1}{2}(\boldsymbol{x}-\bar{\boldsymbol{x}}_k)^T \boldsymbol{S}_k^{-1}(\boldsymbol{x}-\bar{\boldsymbol{x}}_k)\right) \qquad \hat{\pi}_k = \frac{N_k}{N}$$

$$\delta_k(\boldsymbol{x}) = \ln(\hat{\pi}_k \hat{f}_k(\boldsymbol{x})) = \ln(\hat{\pi}_k) + \ln(\hat{f}_k(\boldsymbol{x})) =$$

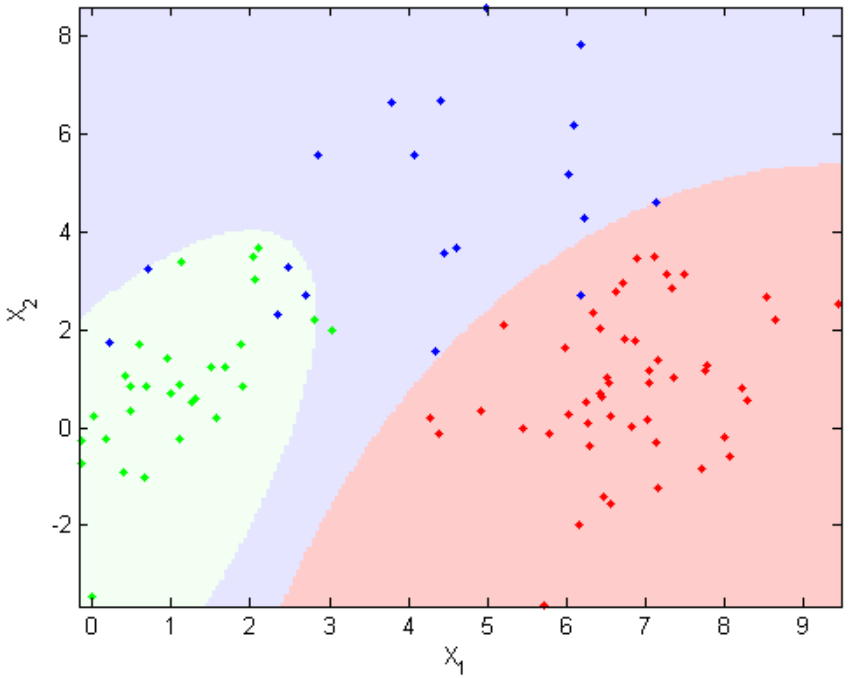
$$= -\frac{P}{2}\ln(2\pi) - \frac{1}{2}\ln|\boldsymbol{S}_k| - \frac{1}{2}(\boldsymbol{x}-\bar{\boldsymbol{x}}_k)^T \boldsymbol{S}_k^{-1}(\boldsymbol{x}-\bar{\boldsymbol{x}}_k) + \ln\frac{N_k}{N}$$

Klasyfikacja: gaussowskie klasyfikatory Bayesa (QDA)

W najbardziej ogólnym przypadku każda z klas opisywana jest osobną macierzą kowariancji ($\Sigma_1 \neq \Sigma_2 \neq \ldots \neq \Sigma_K$), a ich prawdopodobieństwa *a priori* mają różne wartości ($\pi_1 \neq \pi_2 \neq \ldots \neq \pi_K$). Wówczas funkcje dyskryminacyjne dane są zależnością:

$$\delta_k(x) = -\frac{1}{2}(x-\bar{x}_k)^T S_k^{-1}(x-\bar{x}_k) - \frac{1}{2}\ln|S_k| + \ln\frac{N_k}{N}$$

Dla powyższej postaci funkcji dyskryminacyjnych powierzchnie rozdzielająca klasy są drugiego stopnia (dla $P=2$ są to elipsy, parabole lub hiperbole). Dlatego też procedura klasyfikacji bazująca na ich wykorzystaniu nazywana jest kwadratową analizą dyskryminacyjną (QDA – *Quadratic Discriminant Analysis*).



	MATLAB	R
QDA:	`fitcdiscr`	`qda`

Klasyfikacja: gaussowskie klasyfikatory Bayesa (LDA)

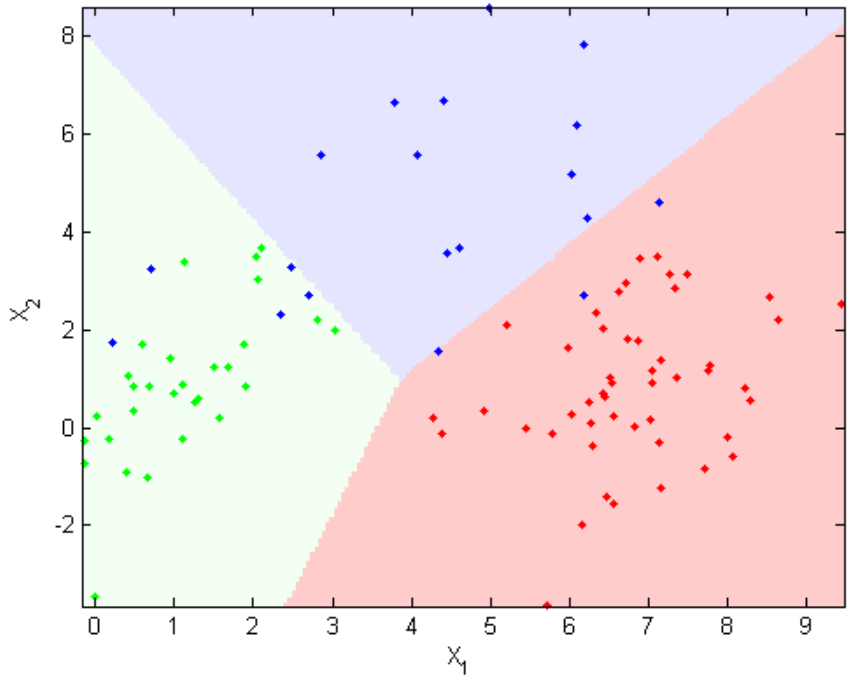
Jeżeli jednak założymy, że macierze kowariancji wszystkich klas są takie same ($\Sigma_1 = \Sigma_2 = \ldots = \Sigma_K = \Sigma$), to funkcje dyskryminacyjne upraszczają się do postaci:

$$\delta_k(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x} - \bar{\boldsymbol{x}}_k)^T \boldsymbol{S}^{-1}(\boldsymbol{x} - \bar{\boldsymbol{x}}_k) + \ln \frac{N_k}{N}$$

gdzie S jest estymatorem wspólnej macierzy kowariancji Σ klas:

$$\hat{\boldsymbol{\Sigma}} = \boldsymbol{S} = \frac{1}{N-K}\sum_{k=1}^{K}(N_k - 1)\boldsymbol{S}_k = \frac{1}{N-K}\sum_{k=1}^{K}\sum_{j=1}^{N_k}(\boldsymbol{x}_{kj} - \bar{\boldsymbol{x}}_k)(\boldsymbol{x}_{kj} - \bar{\boldsymbol{x}}_k)^T$$

W tym przypadku klasy rozdzielane są hiperpłaszczyznami (prostymi dla $P = 2$), stąd nazwa liniowa analiza dyskryminacyjna (LDA – *Linear Discriminant Analysis*).



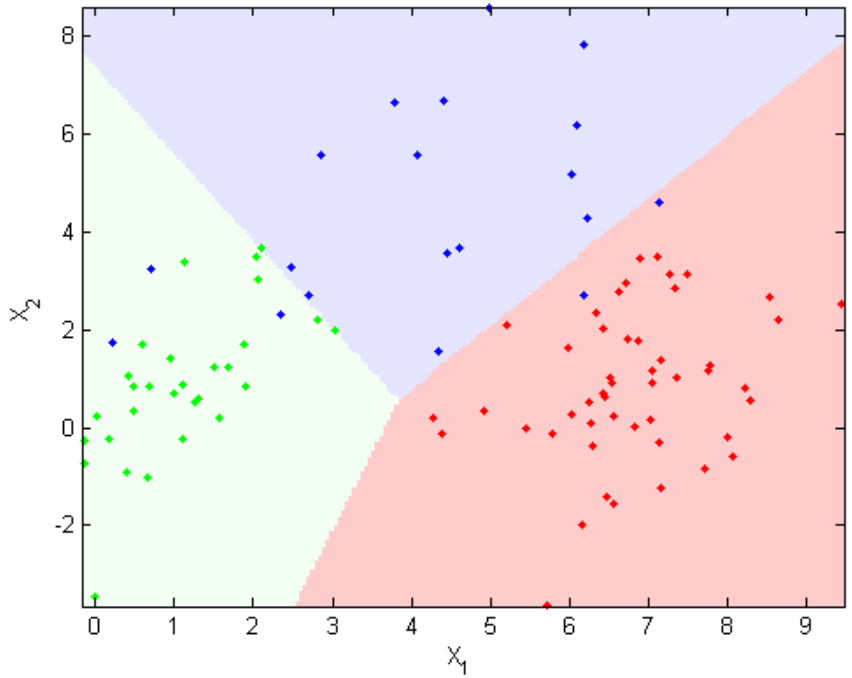
	MATLAB	R
LDA:	`fitcdiscr`	`lda`

Klasyfikacja: gaussowskie klasyfikatory Bayesa (min. odl. Mahalanobisa)

Dodatkowe założenie, mówiące o równych prawdopodobieństwach wszystkich klas ($\pi_1 = \pi_2 = \ldots = \pi_K = 1/K$) prowadzi do dalszego uproszczenia funkcji dyskryminacyjnych:

$$\delta_k(\boldsymbol{x}) = -\frac{1}{2}(\boldsymbol{x} - \bar{\boldsymbol{x}}_k)^T \boldsymbol{S}^{-1}(\boldsymbol{x} - \bar{\boldsymbol{x}}_k)$$

Klasyfikator taki nazywamy minimalnej odległości Mahalanobisa, gdyż przypisuje on wektor x na podstawie jego odległości Mahalanobisa od wektorów wartości średnich poszczególnych klas.



Klasyfikacja: gaussowskie klasyfikatory Bayesa

Niekorzystną cechą klasyfikatorów gaussowskich jest spadek skuteczności wraz ze wzrostem wymiaru P przestrzeni cech w porównaniu do liczebności próby uczącej N.

Problemem wynika z konieczności odwracania macierzy kowariancji (osobnych klas dla QDA lub łącznej dla LDA) podczas wyznaczania funkcji dyskryminacyjnych. Dla zbiorów uczących o małej liczbie próbek w porównaniu z wymiarem przestrzeni cech odwrócenie tych macierzy będzie niemożliwe ze względu na niepełny rząd. Sytuacja taka ma miejsce gdy:

- **dla klasyfikatora QDA liczba wektorów N_K z najmniejszej klasy w próbie uczącej jest nie większa od liczby cech P;**
- **dla klasyfikatora LDA liczność całej próby N jest nie większa od liczby cech P.**

Fakt ten ogranicza możliwość bezpośredniego użycia klasyfikatorów w takiej postaci do danych z wielkoskalowych technik pomiarowych, w przypadku których niemal zawsze prawdziwa jest zależność: $P >> N$.

Klasyfikacja: gaussowskie klasyfikatory Bayesa

Przykładowymi sposobami ominięcia powyższych ograniczeń mogą być:

- **zmniejszenie wymiarowości** przez **selekcję cech różnicujących** (co i tak zwykle jest robione, aby poprawić skuteczność klasyfikatora);

- **przeniesienie danych do przestrzeni składowych głównych**, gdzie między cechami nie występują korelacje (macierz kowariancji jest diagonalna);

- zastąpienie odwrotnej macierzy kowariancji z próby S^{-1} **macierzą pseudoodwrotną** S^{+}, wyznaczoną za pomocą rozkładu na wartości szczególne (SVD):

$$S = U \Sigma V^T$$

$$S^{+} = V \Sigma^{+} U^T$$

gdzie Σ^{+} jest macierzą diagonalną, zawierającą na głównej przekątnej odwrotności niezerowych wartości szczególnych macierzy S (elementów przekątnej macierzy Σ);

- **przyjęcie założenia o braku korelacji cech**, co oznacza, że macierz kowariancji staje się diagonalna. Podejście to określane jest jako **naiwna klasyfikacja bayesowska**.

Klasyfikacja: gaussowskie klasyfikatory Bayesa

Przykładowymi sposobami ominięcia powyższych ograniczeń mogą być:

- **zmniejszenie wymiarowości** przez **selekcję cech różnicujących** (co i tak zwykle jest robione, aby poprawić skuteczność klasyfikatora);

- **przeniesienie danych do przestrzeni składowych głównych**, gdzie między cechami nie występują korelacje (macierz kowariancji jest diagonalna);

- zastąpienie odwrotnej macierzy kowariancji z próby S^{-1} **macierzą pseudoodwrotną** S^{+}, wyznaczoną za pomocą rozkładu na wartości szczególne (SVD):

$$S = U \Sigma V^T$$

$$S^{+} = V \Sigma^{+} U^T$$

gdzie Σ^{+} jest macierzą diagonalną, zawierającą na głównej przekątnej odwrotności cierzy S (elementów przekątnej macierzy Σ);

ch, co oznacza, że macierz kowariancji staje est jako **naiwna klasyfikacja bayesowska**.

Dla przypadku *P* > *N*:

$$\Sigma = \begin{bmatrix} \sigma_1 & 0 & \cdots & 0 \\ 0 & \sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \sigma_N \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix} \quad \Sigma^{+} = \begin{bmatrix} 1/\sigma_1 & 0 & \cdots & 0 \\ 0 & 1/\sigma_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1/\sigma_N \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 \end{bmatrix}$$

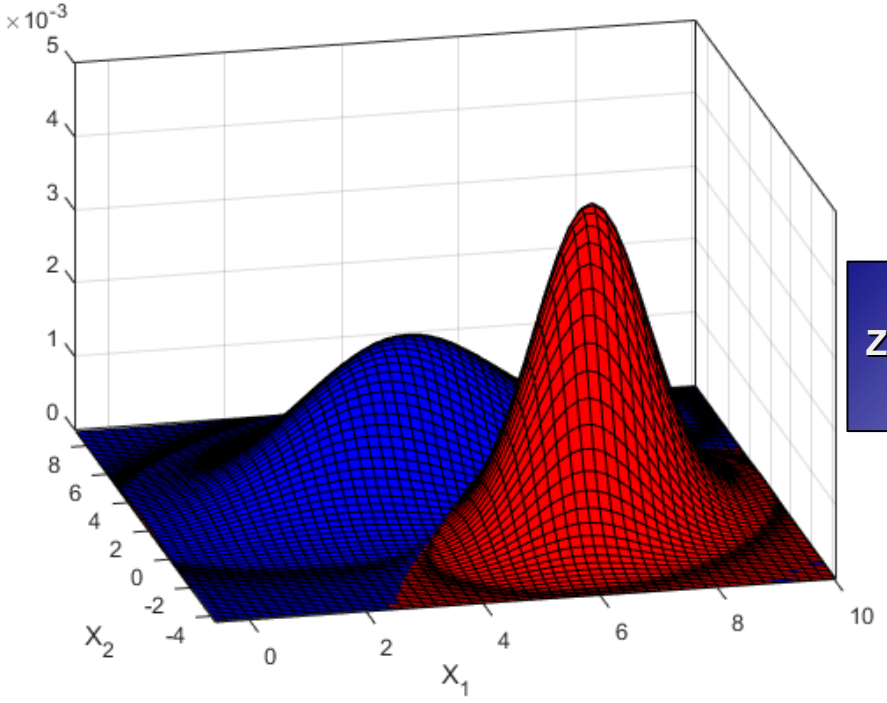
	MATLAB	R
Pseudoinwersja:	`pinv`	`pseudoinverse`

Uczenie maszynowe w bioinformatyce

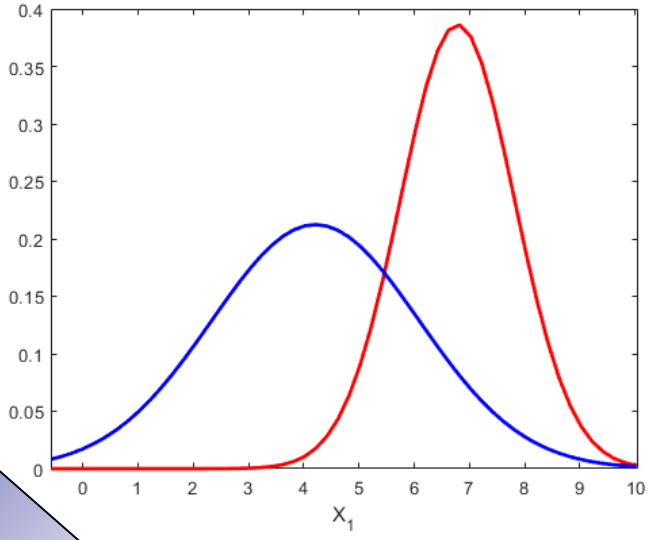
Wykład 8: naiwne klasyfikatory Bayesa

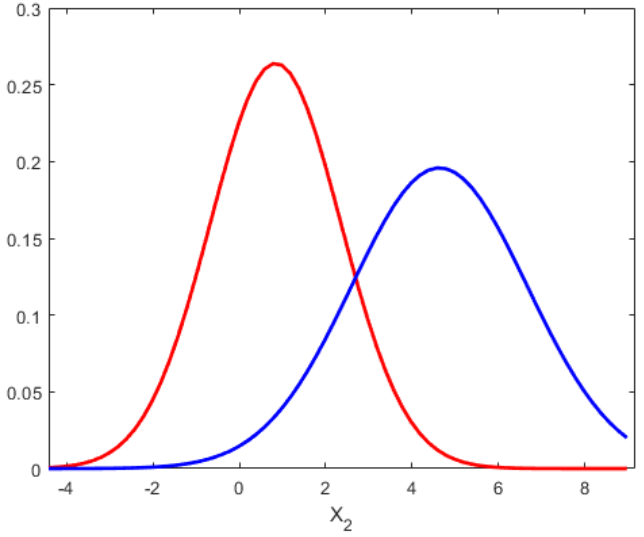
Klasyfikacja: naiwny klasyfikator Bayesa

W przypadku naiwnego klasyfikatora Bayesa (*naive Bayes classifier*) zakłada się niezależność wszystkich cech. Tym samym możliwe jest zredukowanie problemu estymacji P-wymiarowego rozkładu $\hat{f}_k(x)$ k-tej klasy do (zwykle znacznie prostszego) problemu estymacji P jednowymiarowych rozkładów poszczególnych cech.



Założenie o niezależności cech



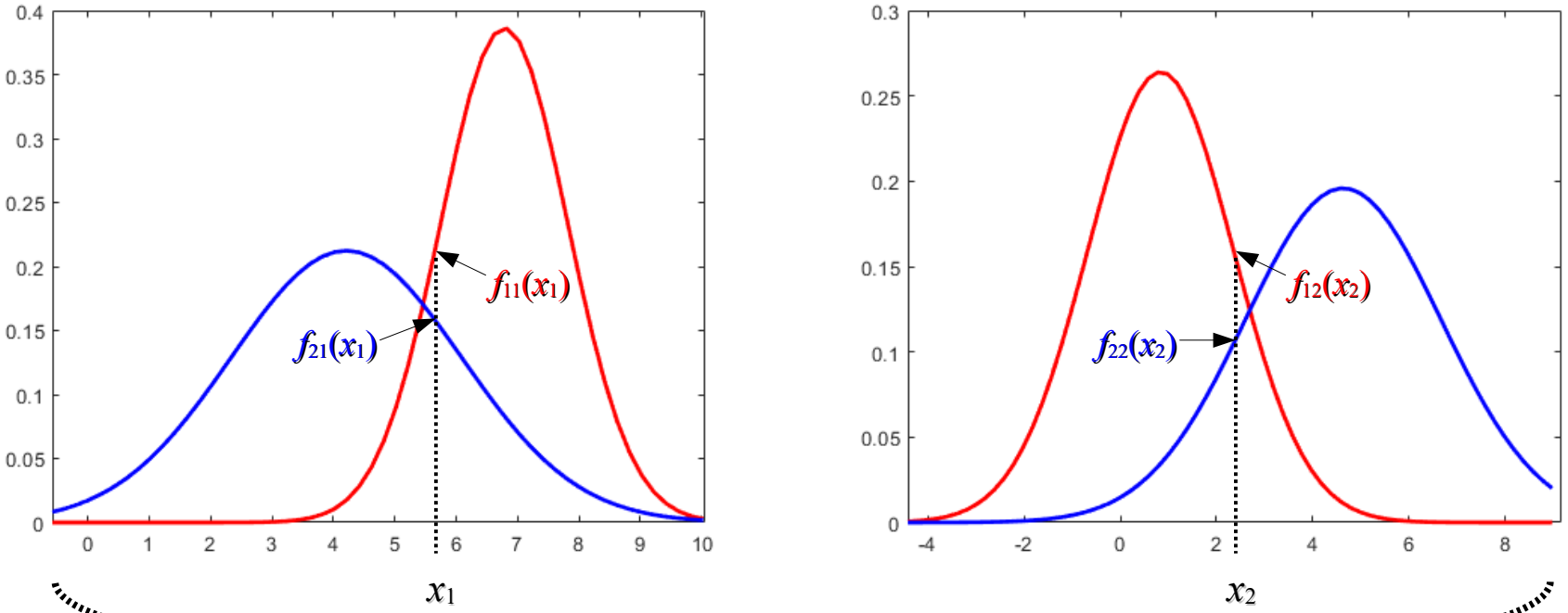


Klasyfikacja: naiwny klasyfikator Bayesa

Ze względu na niezależność cech, rozkład $\hat{f}_k(x)$ może być wyznaczony jako:

$$\hat{f}_k(\boldsymbol{x}) = \prod_{i=1}^{P} \hat{f}_{ki}(x_i)$$

gdzie $\hat{f}_{ki}(x_i)$ to estymowany jednowymiarowy rozkład i-tej cech w k-tej klasie.



$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

$$\hat{f}_1(\boldsymbol{x}) = \hat{f}_{11}(x_1) \cdot \hat{f}_{12}(x_2)$$

$$\hat{f}_2(\boldsymbol{x}) = \hat{f}_{21}(x_1) \cdot \hat{f}_{22}(x_2)$$

Jednowymiarowe rozkłady cech mogą być estymowane w sposób nieparametryczny (np. histogramem) w lub też po przyjęciu założeń co ich postaci (np. gaussowskiej).

Uczenie maszynowe w bioinformatyce

Wykład 8: gaussowskie naiwne klasyfikatory Bayesa

Klasyfikacja: gaussowskie naiwne klasyfikatory Bayesa

W gaussowskich naiwnych klasyfikatorach Bayesa poszczególne cechy w klasach są opisywane przez jednowymiarowe rozkłady normalne $N(\mu_{ki}, \sigma_{ki})$:

$$\hat{f}_{ki}(x_i) = \frac{1}{\sqrt{2\pi\sigma_{ki}^{\,2}}} \exp\left(\frac{-(x_i - \mu_{ki})^2}{2\sigma_{ki}^{\,2}}\right)$$

gdzie nieznane parametry μ_{ki} i σ_{ki} zastępowane są poprzez estymatory wyznaczane na podstawie wartości i-tej cechy w k-tej klasie:

$$\hat{\mu}_{ki} = \bar{x}_{ki} = \frac{1}{N_k}\sum_{j=1}^{N_k} x_{kij}$$

$$\hat{\sigma}_{ki}^{\,2} = s_{ki}^{\,2} = \frac{1}{N_k - 1}\sum_{j=1}^{N_k} (x_{kij} - \bar{x}_{ki})^2$$

Podobnie jak dla „zwykłego” klasyfikatora Bayesa, prawdopodobieństwa *a priori* klas są estymowane jako częstości występowania ich wektorów w próbie uczącej:

$$\hat{\pi}_k = \frac{N_k}{N}$$

Klasyfikacja: gaussowskie naiwne klasyfikatory Bayesa

Klasyfikacja odbywa się na podstawie zlogarytmowanych iloczynów $\hat{f}_k(\boldsymbol{x})\hat{\pi}_k$:

$$\hat{d}(\boldsymbol{x}) = \underset{k}{\operatorname{argmax}} \ \ln(\hat{\pi}_k \hat{f}_k(\boldsymbol{x})) = \underset{k}{\operatorname{argmax}} \ \delta_k(\boldsymbol{x})$$

Wartość funkcji dyskryminacyjnej $\delta_K(\boldsymbol{x})$ dla k-tej klasy dana jest wzorem:

$$\delta_k(\boldsymbol{x}) = \ln\left(\hat{\pi}_k \hat{f}_k(\boldsymbol{x})\right) = \sum_{i=1}^{P}\left(\ln\left(\frac{1}{\sqrt{2\pi s_{ki}^{\ 2}}}\right) - \frac{(x_i - \bar{x}_{ki})^2}{2 s_{ki}^{\ 2}}\right) + \ln\frac{N_k}{N}$$

$$\hat{f}_k(\boldsymbol{x}) = \prod_{i=1}^{P} \hat{f}_{ki}(x_i) \qquad \hat{f}_{ki}(x_i) = \frac{1}{\sqrt{2\pi s_{ki}^{\ 2}}} \exp\left(\frac{-(x_i - \bar{x}_{ki})^2}{2 s_{ki}^{\ 2}}\right) \qquad \hat{\pi}_k = \frac{N_k}{N}$$

$$\delta_k(\boldsymbol{x}) = \ln\left(\hat{\pi}_k \hat{f}_k(\boldsymbol{x})\right) = \ln(\hat{\pi}_k) + \ln\left(\prod_{i=1}^{P} \hat{f}_{ki}(x_i)\right) = \ln(\hat{\pi}_k) + \sum_{i=1}^{P} \ln\left(\hat{f}_{ki}(x_i)\right) =$$

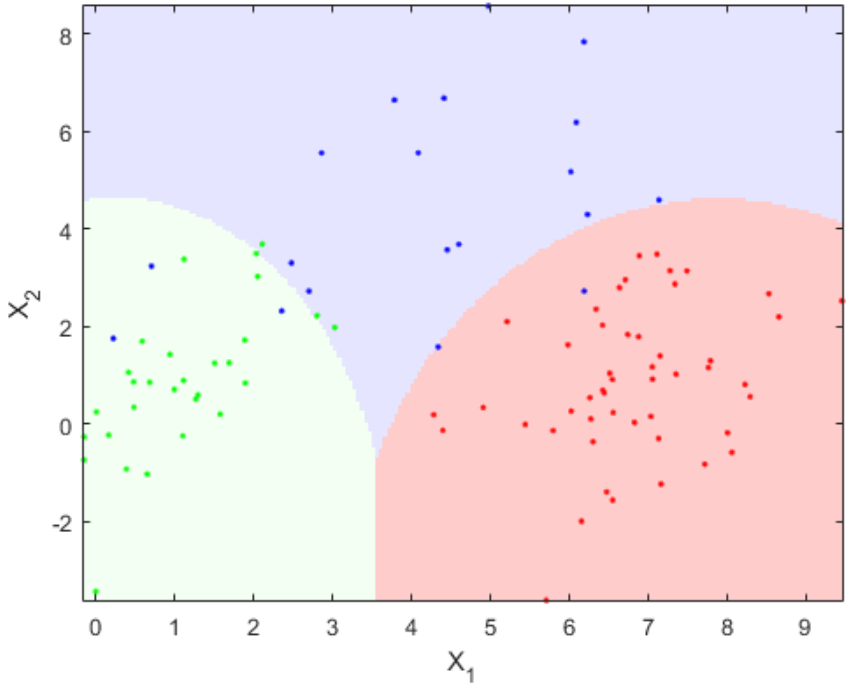
$$= \sum_{i=1}^{P}\left(\ln\left(\frac{1}{\sqrt{2\pi s_{ki}^{\ 2}}}\right) - \frac{(x_i - \bar{x}_{ki})^2}{2 s_{ki}^{\ 2}}\right) + \ln\frac{N_k}{N}$$

Klasyfikacja: gaussowskie naiwne klasyfikatory Bayesa (DQDA)

W najbardziej ogólnym przypadku dla i-tej cechy klasy opisywane są rozkładami $N(\mu_{ki}, \sigma_{ki})$ o różnych wariancjach ($\sigma_{1i} \neq \sigma_{2i} \neq \ldots \neq \sigma_{Ki}$), a prawdopodobieństwa *a priori* klas mają różne wartości ($\pi_1 \neq \pi_2 \neq \ldots \neq \pi_K$). Wówczas funkcje dyskryminacyjne dane są jako:

$$\delta_k(\boldsymbol{x}) = \sum_{i=1}^{P} \left(\ln \left(\frac{1}{\sqrt{2\pi\, s_{ki}^{\,2}}} \right) - \frac{(x_i - \bar{x}_{ki})^2}{2\, s_{ki}^{\,2}} \right) + \ln \frac{N_k}{N}$$

Dla powyższej postaci funkcji powierzchnie rozdzielająca klasy są drugiego stopnia, dlatego metoda ta nazywana jest diagonalną kwadratowa analizą dyskryminacyjną (DQDA – *Diagonal Quadratic Discriminant Analysis*).



	MATLAB	R
DQDA:	`fitcdiscr`	`dqda`

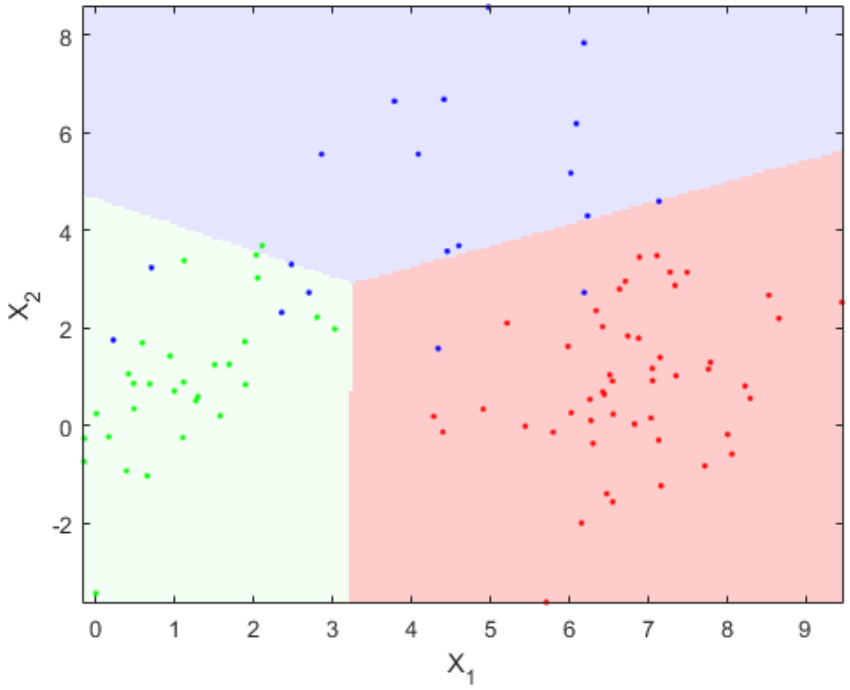
Klasyfikacja: gaussowskie naiwne klasyfikatory Bayesa (DLDA)

Założenie, mówiące że cechy w klasach mają rozkłady o jednakowych wariancjach ($\sigma_{1i} = \sigma_{2i} = \ldots = \sigma_{Ki} = \sigma_i$) prowadzi do diagonalnej liniowej analizy dyskryminacyjnej (DLDA – *Diagonal Linear Discriminant Analysis*). Funkcje dyskryminacyjne mają tu postać:

$$\delta_k(\boldsymbol{x}) = -\sum_{i=1}^{P} \frac{\left(x_i - \bar{x}_{ki}\right)^2}{2 s_i^2} + \ln \frac{N_k}{N}$$

gdzie s_i^2 to wariancja i-tej cechy estymowana dla całego zbioru danych.

Otrzymany w taki sposób klasyfikator jest liniowy, co oznacza, że klasy rozdzielane są hiperpłaszczyznami (prostymi dla $P = 2$).



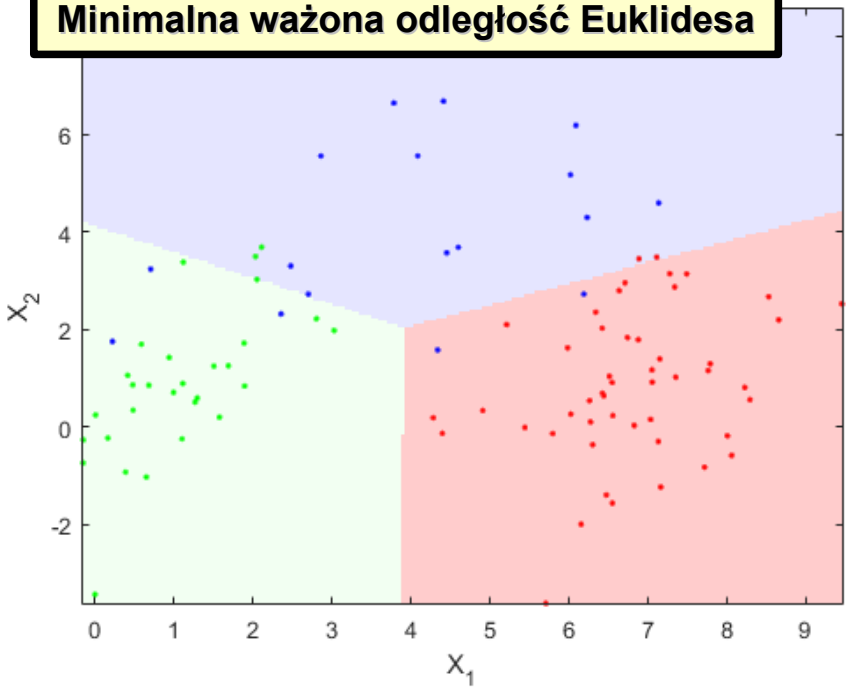
	MATLAB	R
DLDA:	`fitcdiscr`	`dlda`

Klasyfikacja: gaussowskie naiwne klasyfikatory Bayesa (min. odległości)

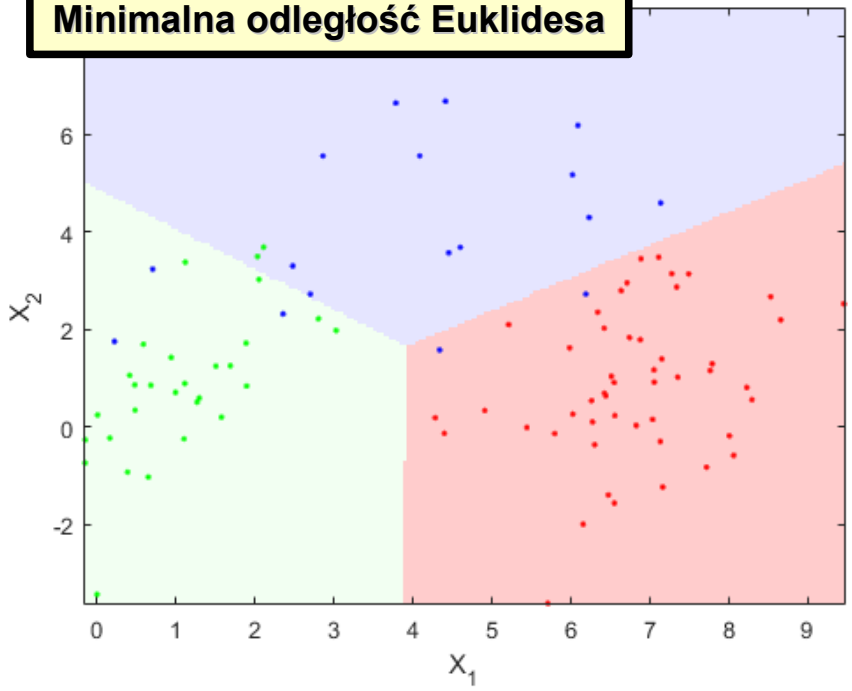
Przy dalszym założeniu, mówiącym o równości prawdopodobieństw π_k reguła ta sprowadza się do klasyfikacji na podstawie minimalnej ważonej odległości Euklidesa pomiędzy wektorem x a wektorami wartości średnich klas.

Jeżeli dodatkowo założymy, że cechy mają jednakową wariancję (np. równą 1, bo wykonano standaryzację zbioru danych), to naiwny klasyfikator Bayesa sprowadza się do klasyfikatora minimalnej odległości Euklidesa.

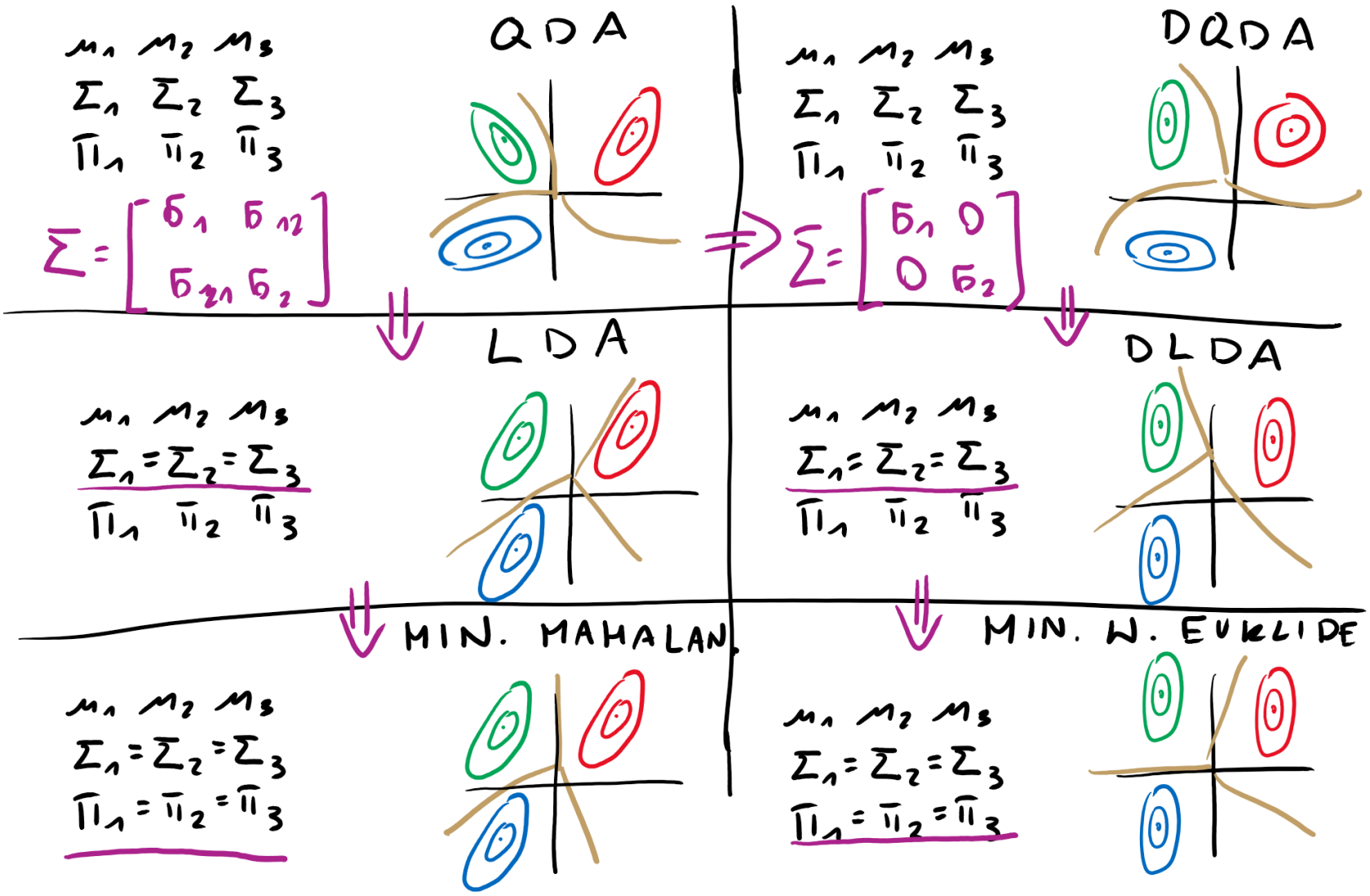
Minimalna ważona odległość Euklidesa



Minimalna odległość Euklidesa



Klasyfikacja: gaussowskie naiwne klasyfikatory Bayesa (min. odległości)



Uczenie maszynowe w bioinformatyce

Wykład 8: klasyfikator *J* najbliższych sąsiadów (*J*-NN)

Klasyfikacja: metoda *J* najbliższych sąsiadów

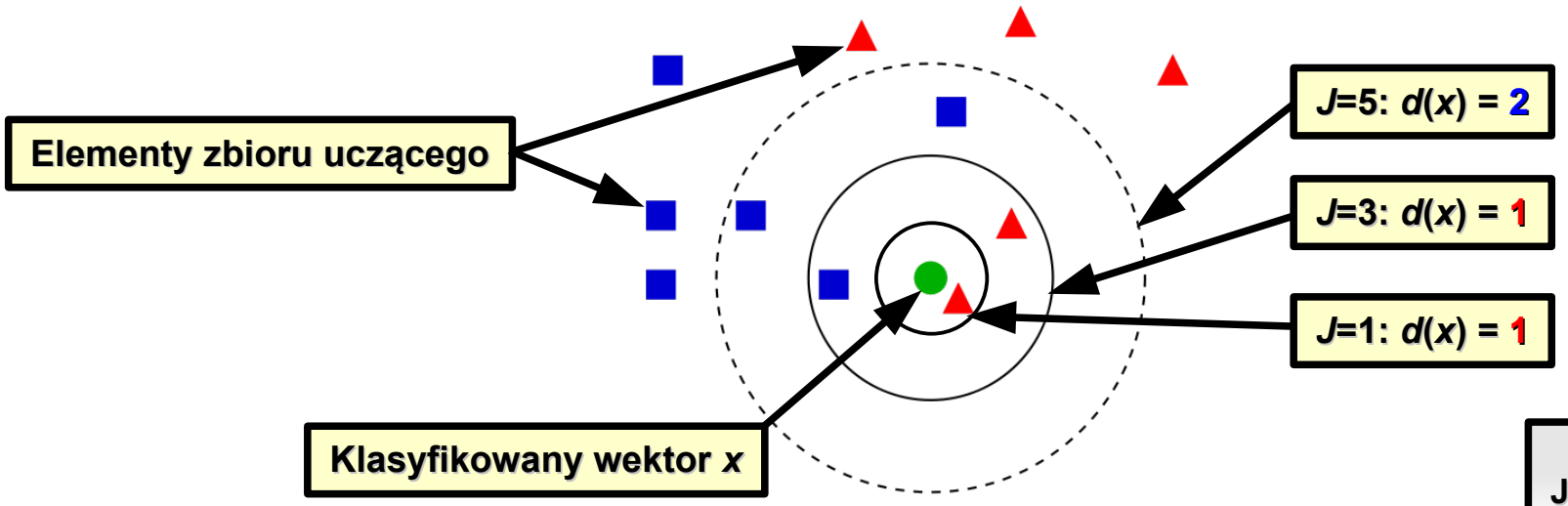
Popularnym podejściem do klasyfikacji, nie wymagającym założeń dotyczących funkcyjnej postaci rozkładów prawdopodobieństwa wektorów cech w klasach, jest metoda *J* najbliższych sąsiadów (*J*NN – *J Nearest Neighbor*).

W metodzie tej prawdopodobieństwo *a posteriori* $P(k|\boldsymbol{x})$ estymowane jest bezpośrednio na podstawie zbioru uczącego, jako:

$$\hat{\mathrm{P}}(k|\boldsymbol{x}) = \frac{1}{J}\sum_{i=1}^{N} I\left(\rho(\boldsymbol{x},\boldsymbol{x}_i) \leqslant \rho\left(\boldsymbol{x},\boldsymbol{x}^{(J)}\right)\right) I\left(y_i = k\right)$$

gdzie $I(A)$ to funkcja indykatorową (o wartości 1 gdy spełniony jest warunek A oraz 0 w przeciwnym wypadku), $\boldsymbol{x}^{(J)}$ to J-ty co do odległości od $\boldsymbol{x}$ wektor zbioru uczącego ($J \geq 1$ jest zadaną liczbą naturalną), a ρ jest w miarą niepodobieństwa wektorów cech.

Oznacza to, że klasyfikator $d(\boldsymbol{x})$ przypisuje rozpatrywany wektor cech do klasy, do której należy większość spośród J jego najbliższych sąsiadów ze zbioru uczącego.



JNN:	MATLAB	R
	`fitcknn`	`knn`

Klasyfikacja: metoda J najbliższych sąsiadów

Algorytm klasyfikacji metodą J najbliższych sąsiadów:

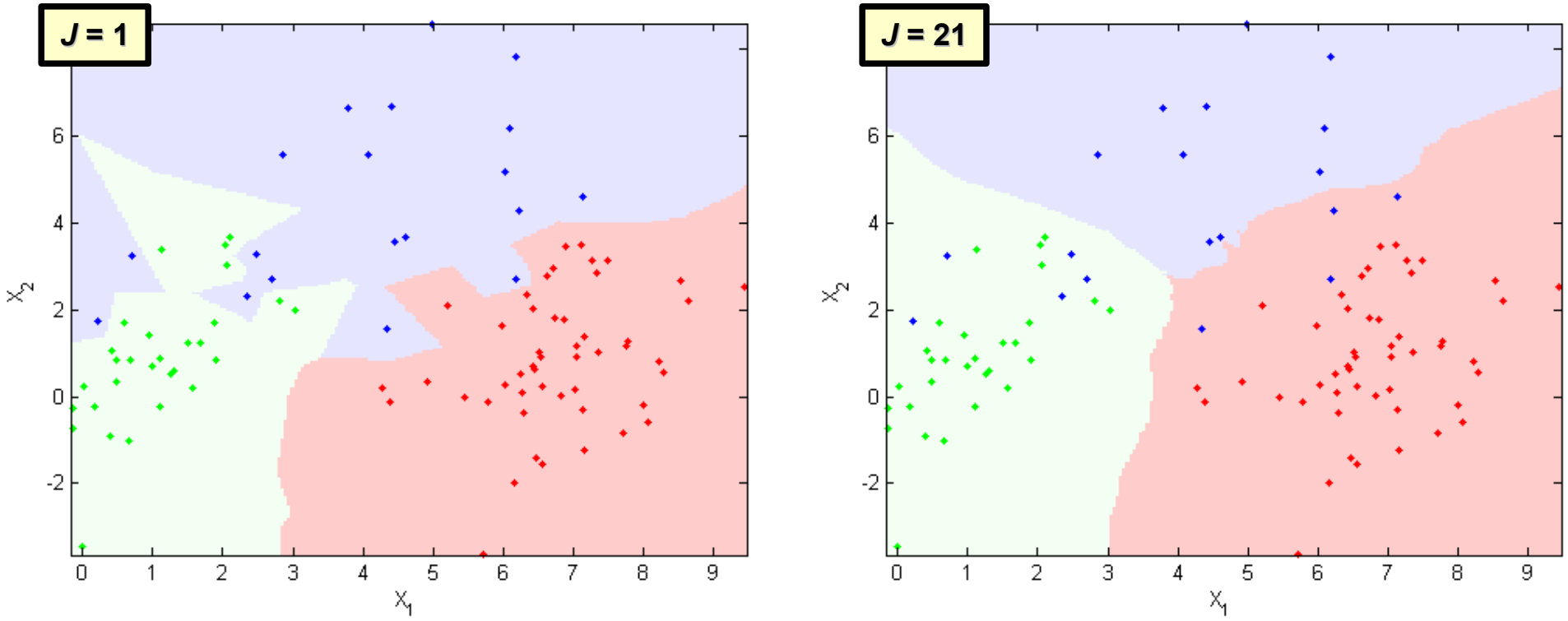
1. Wyznaczenie wartości miary niepodobieństwa (np. odl. Euklidesa) pomiędzy klasyfikowanym wektorem x a wszystkimi wektorami cech ze zbioru uczącego.
2. Posortowanie wektorów ze zbioru uczącego zgodnie z miarą niepodobieństwa do x i wybór sąsiedztwa złożonego z J wektorów o najmniejszej wartości tej miary.
3. Przypisanie x do klasy, do której należy większość wektorów z sąsiedztwa.

Dodatkowym **elementem algorytmu jest metoda rozwiązywania sytuacji remisowych**, w których otoczenie zawiera równą liczbę sąsiadów z dwóch lub większej liczby klas. Najczęściej spotykane są trzy podejścia do tego problemu:

- wybór wielkości otoczenia uniemożliwiającej zajście sytuacji remisowej (dla dwóch klas warunek ten spełnia dowolne nieparzyste J);
- zwiększanie otoczenia o jeden, aż do momentu rozwiązania sytuacji remisowej;
- wybranie klasy, do której należy najbliższy sąsiad (czyli sprowadzenie klasyfikatora do przypadku $J = 1$).

Klasyfikacja: metoda *J* najbliższych sąsiadów (znaczenie parametru *J*)

Wielkość sąsiedztwa jest parametrem mającym decydujące znaczenie dla działania metody: zbyt małe J może skutkować nadmiernym dopasowaniem klasyfikatora do zbioru uczącego (przeuczeniem, które ogranicza zdolność do generalizacji podczas klasyfikacji wektorów nie używanych do treningu), natomiast duże J może prowadzić nadmiernego „uliniowienia" granic decyzyjnych.



Nie istnieje wielkość otoczenia, która byłaby optymalna w każdym przypadku, gdyż jest ona zależna od charakterystyki danych. Dlatego trzeba ją ustalać empirycznie (np. przez walidację krzyżową), jako wartość J, która minimalizuje błąd klasyfikacji szacowany na danym zbiorze uczącym.

Uczenie maszynowe w bioinformatyce

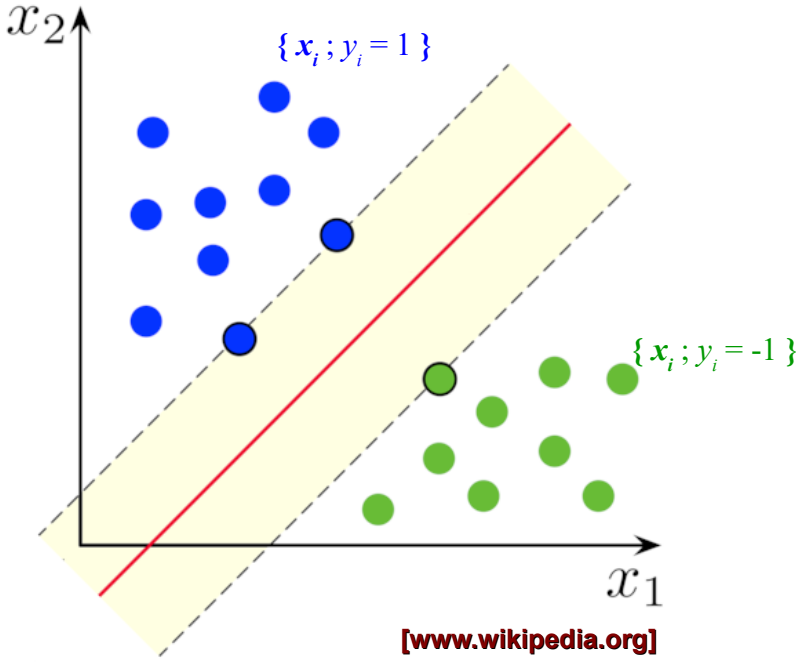
Wykład 8: klasyfikator SVM

Klasyfikacja: metoda wektorów nośnych

Klasyfikator korzystający z metody wektorów nośnych lub podpierających (*Support Vector Machine*, SVM) jest w stanie rozdzielić wektory cech należące do dwóch klas, typowo oznaczanych etykietami $y=1$ oraz $y=-1$.

Granica decyzyjna ma postać hiperpłaszczyzny (linii dla $P=2$, płaszczyzny dla $P=3$), która jest wyznaczana w taki sposób, aby była możliwie odległa od elementów zbioru uczącego. Tym samym maksymalizowany jest margines separacji między wektorami cech z różnych klas.

O przebiegu granicy rozdzielającej klasy decydują wektory nośne, które znajdują się na granicach marginesu separacji, czyli na tzw. hiperpłaszczyznach kanonicznych.



[www.wikipedia.org]

	MATLAB	R
SVM:	`fitcsvm`	`svm`

Klasyfikacja: metoda wektorów nośnych

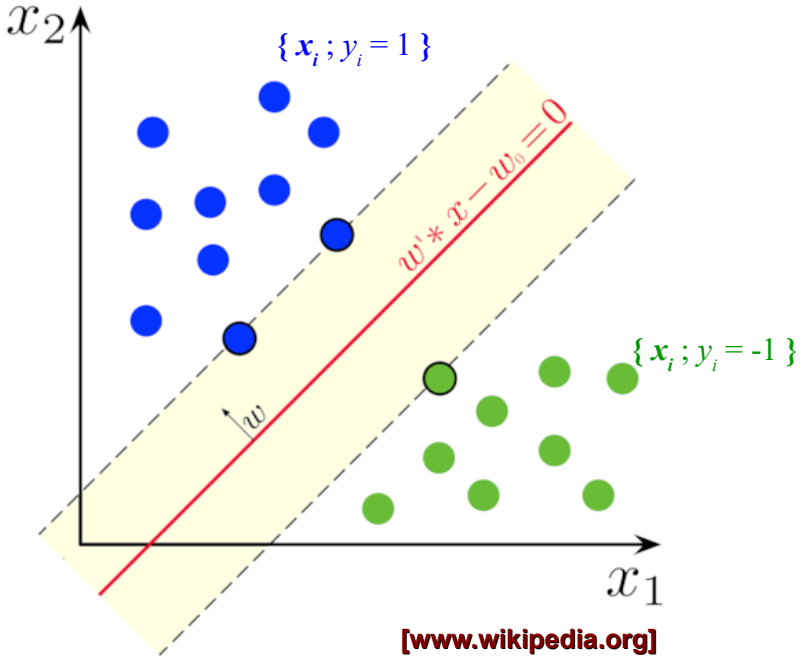
Postać funkcji decyzyjnej klasyfikatora SVM jest bezpośrednio powiązana ze wzorem hiperpłaszczyzny zdefiniowanej poprzez wektor normalny w i wyraz wolny w_0:

$$\delta(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \boldsymbol{x} + w_0 = \sum_{i=1}^{P} w_i \cdot x_i + w_0 = w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_P \cdot x_P + w_0$$

Reguła decyzyjna sprowadza się do określenia, po której stronie hiperpłaszczyzny $\delta(\boldsymbol{x}) = 0$ znajduje się klasyfikowany wektor $\boldsymbol{x}$:

$$\hat{d}(\boldsymbol{x}) = \begin{cases} +1 & gdy \quad \delta(\boldsymbol{x}) > 0 \\ -1 & gdy \quad \delta(\boldsymbol{x}) \leq 0 \end{cases}$$

Oznacza to, że prawidłowo zaklasyfikowane wektory x_i spełniają warunek $y_i\, \delta(x_i) \geq 0$.



[www.wikipedia.org]

Klasyfikacja: metoda wektorów nośnych

Przy odpowiednim przeskalowaniu wektora w, szerokość marginesu separacji jest równa $2/\|w\|$, gdzie $\|w\|$ to norma (długość) tego wektora:

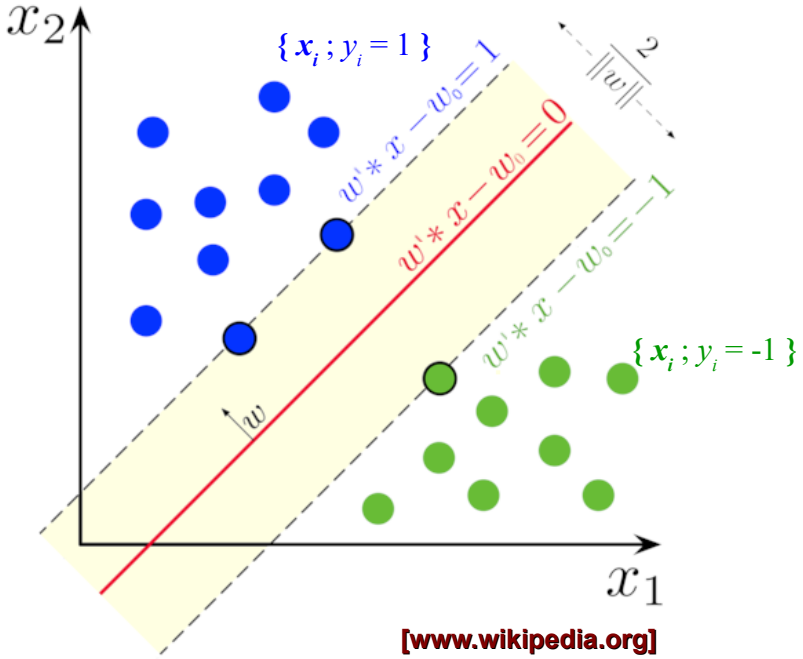
$$\|\boldsymbol{w}\| = \sqrt{w_1^2 + w_2^2 + \ldots + w_P^2} = \sqrt{\boldsymbol{w}^T \boldsymbol{w}}$$

Oznacza to, że maksymalizacja szerokości marginesu polega na znalezieniu takiego wektora w o minimalnej normie, aby wszystkie wektory x_i znalazły się po właściwych stronach obu granic marginesu. Minimalizowane jest więc wyrażenie:

$$Q(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 = \frac{1}{2}\boldsymbol{w}^T \boldsymbol{w}$$

przy ograniczeniach:

$$y_i \cdot \left(\boldsymbol{w}^T \cdot \boldsymbol{x_i} + w_0\right) \geq 1 \quad (i = 1, \ldots, N)$$



[www.wikipedia.org]

Klasyfikacja: metoda wektorów nośnych

W praktyce zwykle nie jest możliwe znalezienie płaszczyzny, która idealnie oddziela dane pochodzące z obu klas. Dlatego konieczne jest związanie z każdym wektorem x_i zmiennej $\xi_i \geq 0$, o wartości zależnej od odległości tego wektora od granicy marginesu (dla wektorów leżących po „właściwej" stronie granicy $\xi_i = 0$).

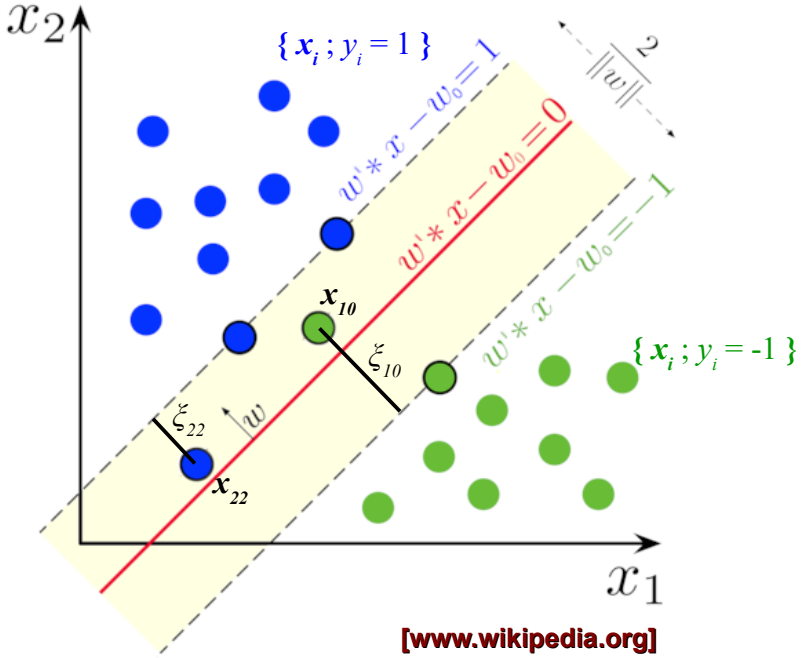
W takim przypadku minimalizowane jest wyrażenie:

$$Q(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{N}\xi_i$$

przy ograniczeniach:

$$y_i \cdot \left(\boldsymbol{w}^T \cdot \boldsymbol{x_i} + w_0\right) + \xi_i \geq 1 \quad (i = 1, \ldots, N)$$

Stała $C > 0$ we wzorze na $Q(w)$ jest parametrem, wyrażającym kompromis pomiędzy szerokością marginesu, a dopuszczaniem do błędów.



[www.wikipedia.org]

Klasyfikacja: metoda wektorów nośnych

Przedstawiony problem optymalizacyjny sprowadza się do zadania programowania kwadratowego, które najczęściej rozwiązywane jest metodą mnożników Lagrange'a. W wyniku otrzymuje się wektor w dany zależnością:

$$\boldsymbol{w} = \sum_{x_i \in SV} \alpha_i \, y_i \, \boldsymbol{x_i}$$

gdzie α_i to mnożniki Lagrange'a, zaś SV to zbiór wektorów x_i, dla których $\alpha_i \geq 0$. Mogą one być typowymi wektorami nośnymi, leżącymi na granicach marginesu separacji (wówczas $0 < \alpha_i < C$), ale mogą też znajdować po niewłaściwych stronach granic ($\alpha_i = C$).

Przy znanym w możliwe jest obliczenie wartości wyrazu wolnego w_0 na podstawie dowolnego wektora nośnego leżącego na granicy marginesu. Odbywa się to przez odpowiednie przekształcenie równania hiperpłaszczyzny kanonicznej:

$$w_0 = y_i - \boldsymbol{w}^T \cdot \boldsymbol{x_i}$$

Klasyfikacja: metoda wektorów nośnych

Ostatecznie funkcja decyzyjna klasyfikatora SVM uzyskuje postać:

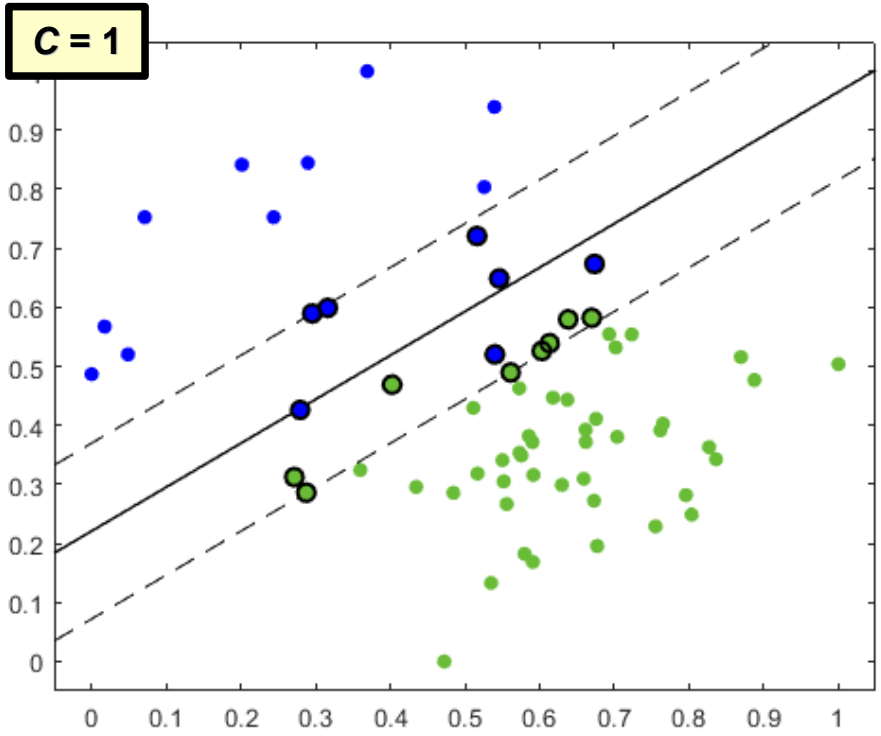
$$\delta(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \boldsymbol{x} + w_0 = \sum_{x_i \in SV} \alpha_i \, y_i \, \boldsymbol{x_i}^T \cdot \boldsymbol{x} + w_0$$

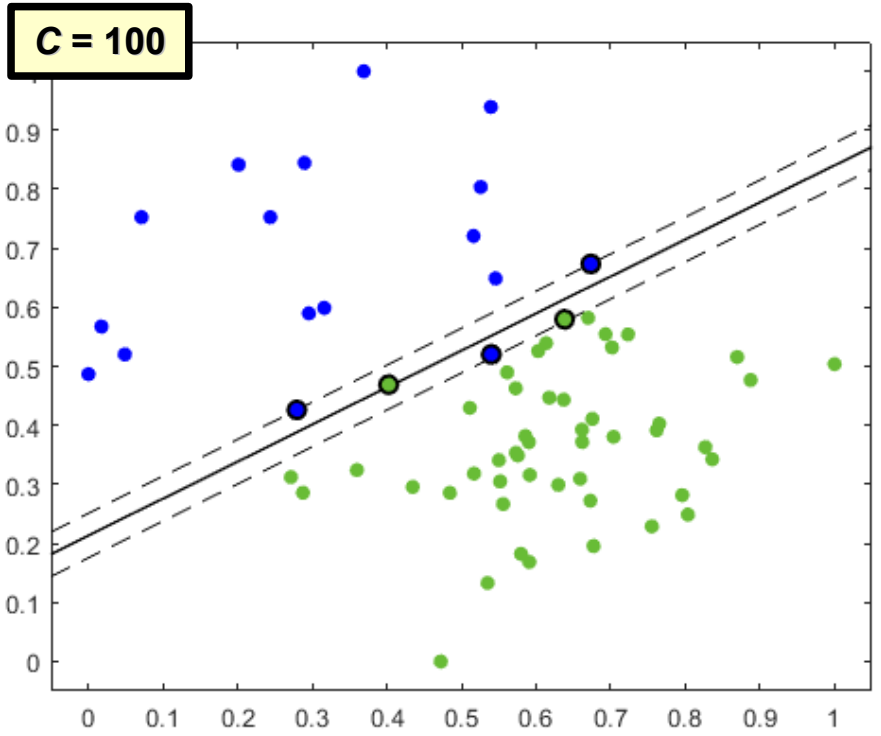
Klasyfikacja następuje na podstawie wartości zmiennej decyzyjnej $\delta(x)$ zgodnie z już wcześniej podaną regułą decyzyjną:

$$\hat{d}(\boldsymbol{x}) = \begin{cases} +1 & gdy \quad \delta(\boldsymbol{x}) > 0 \\ -1 & gdy \quad \delta(\boldsymbol{x}) \leq 0 \end{cases}$$

Klasyfikacja: metoda wektorów nośnych (znaczenie parametru C)

Duże znaczenie dla skuteczności klasyfikatora SVM ma wartość stałej C. Im jest ona większa, tym mniejsza jest szerokość marginesu separacji (bo podczas minimalizacji $Q(w)$ wzrasta wpływ członu związanego z kosztem błędnej klasyfikacji). W efekcie mniej wektorów uczących znajdzie się po niewłaściwych stronach granic marginesu, ale niesie to za sobą ryzyko spadku zdolności klasyfikatora do generalizacji.





Wartość stałej C trzeba ustalać empirycznie osobno dla każdego zbioru danych, jako taką, która zapewnia możliwie mały błąd klasyfikacji. Najczęściej robi się to poprzez walidację krzyżową na zbiorze uczącym.

Klasyfikacja: metoda wektorów nośnych (prawd. a posteriori)

Wartość zmiennej decyzyjnej $\delta(x)$ można zamienić na estymatę prawdopodobieństwa a posteriori $P(k \mid x)$ przy użyciu skalowania Platta:

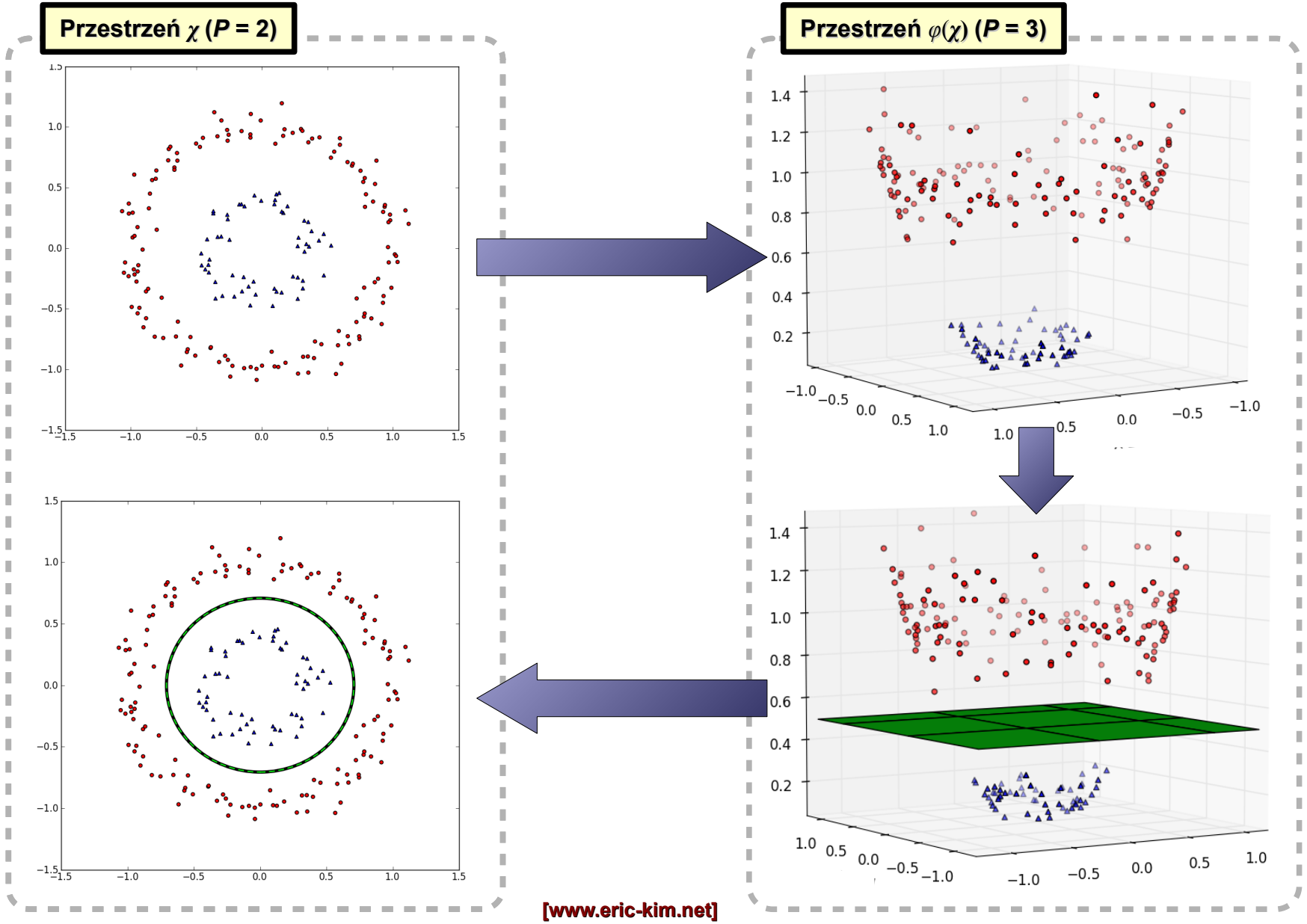
$$\hat{\mathrm{P}}(k|\boldsymbol{x}) = \frac{1}{1+\exp(A\,\delta(\boldsymbol{x})+B)}$$

gdzie A i B to wartości skalarne wyznaczane poprzez dopasowanie modelu regresji liniowej do wartości decyzyjnych uzyskanych dla wszystkich przykładów uczących ze zbioru danych.

Uwaga: typowe implementacje klasyfikatora SVM potrafią same obliczyć prawdopodobieństwo a posteriori poprzez skalowanie Platta, ale zwykle wymaga to podania odpowiedniego parametru podczas ich wywoływania (domyślnie zwracają jedynie wartości zmiennej decyzyjnej).

Klasyfikacja: metoda wektorów nośnych (przypadek nieliniowy)

SVM pozwala też uzyskać nieliniowe granice decyzyjne dzięki przeniesieniu danych z pierwotnej przestrzeni cech $\chi \subseteq \Re^P$ do przestrzeni $\varphi(\chi) \subseteq \Re^V$ o wyższej wymiarowości ($V > P$) i dokonania w niej klasyfikacji za pomocą modelu liniowego.



[www.eric-kim.net]

Klasyfikacja: metoda wektorów nośnych (przypadek nieliniowy)

Podniesienie wymiarowości wykorzystuje nieliniowe przekształcenie $\varphi(x)$: $\Re^P \rightarrow \Re^V$. Wykonanie go na wektorach cech prowadzi do funkcja decyzyjnej w postaci:

$$\delta(\boldsymbol{x}) = \boldsymbol{w}^T \cdot \phi(\boldsymbol{x}) + w_0 = \sum_{x_i \in SV} \alpha_i y_i \phi^T(\boldsymbol{x_i}) \cdot \phi(\boldsymbol{x}) + w_0$$

W praktyce jednak, zamiast bezpośrednio korzystać z przekształcenia φ, stosuje się funkcję jądra $K(x_i, x_j) = \varphi^T(x_i) \cdot \varphi(x_j)$, która pobiera na wejście dwa wektory z przestrzeni cech χ, a zwraca wartość ich iloczynu skalarnego w przestrzeni $\varphi(\chi)$.

Przy użyciu jądra $K(x_i, x_j)$, funkcja decyzyjna klasyfikatora definiowana jest jako:

$$\delta(\boldsymbol{x}) = \sum_{x_i \in SV} \alpha_i y_i K(\boldsymbol{x_i}, \boldsymbol{x}) + w_0$$

Klasyfikacja: metoda wektorów nośnych (przypadek nieliniowy)

Przykładowe funkcje jądra często używane w klasyfikatorze SVM:

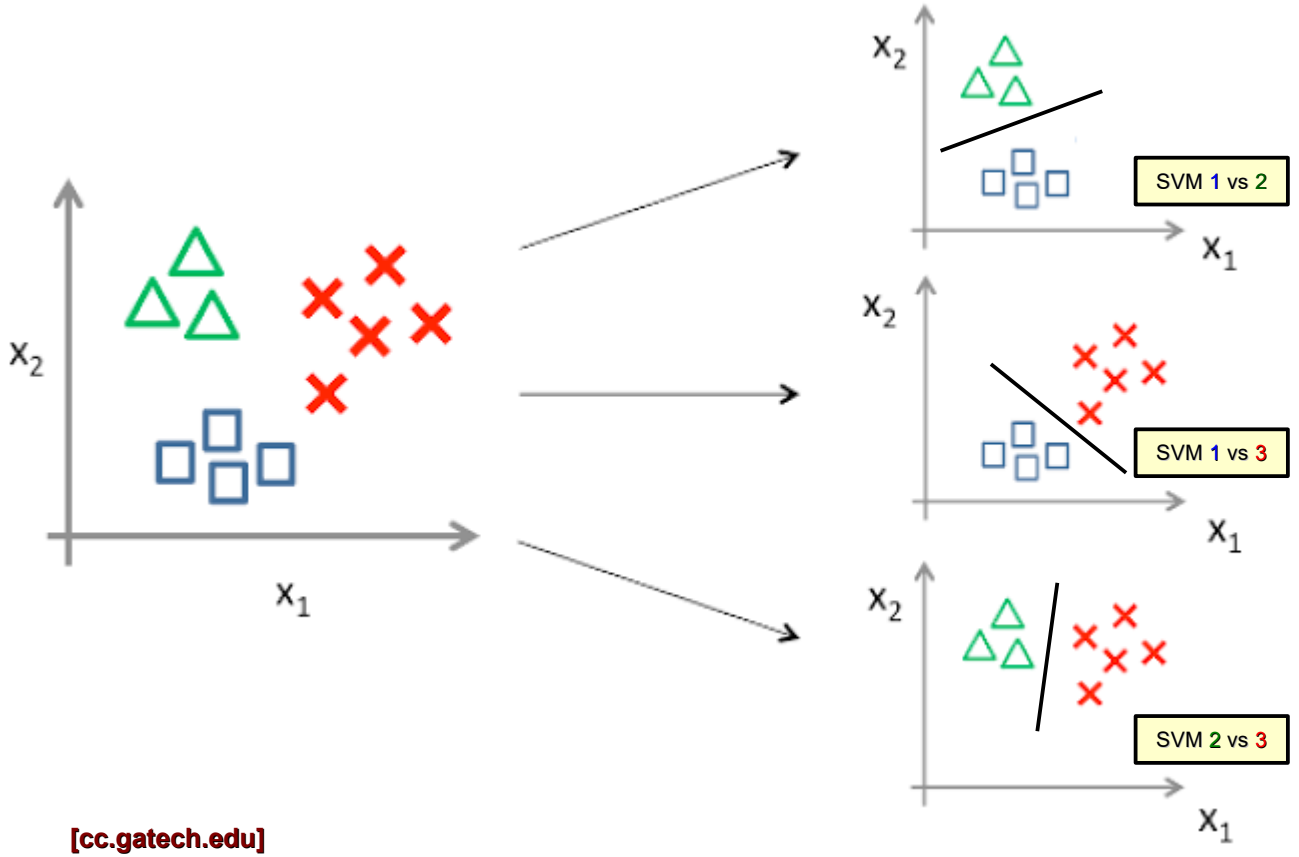
- **gaussowskie (RBF)** $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{1}{2\sigma}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2\right)$
- **wielomianowe** $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \left(1 + \boldsymbol{x}_i^T \boldsymbol{x}_j\right)^q$
- **sigmoidalne** $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = tgh\left(\kappa\, \boldsymbol{x}_i^T \boldsymbol{x}_j - \delta\right)$

Podobnie jak w przypadku stałej C, wartości parametrów jądra (np. σ dla RBF) należy ustalać dla konkretnego zbioru danych, w taki sposób, aby minimalizować błąd.

Klasyfikacja: metoda wektorów nośnych (przypadek wielu klas)

SVM jest klasyfikatorem binarnym – użycie go dla danych zawierających więcej niż dwie klasy wymaga budowania wielu modeli klasyfikacyjnych. Przykładami dwóch podejść do tego problemu są:

- ***One vs One (OvO)*** **– w czasie treningu tworzonych jest $K(K\text{-}1)/2$ klasyfikatorów binarnych, porównujących poszczególne pary klas. Rozpatrywany wektor podawany jest na wejście wszystkich klasyfikatorów, a o jego przypisaniu decyduje głosowanie, tzn. wybierana jest najczęściej wskazywana klasa.**

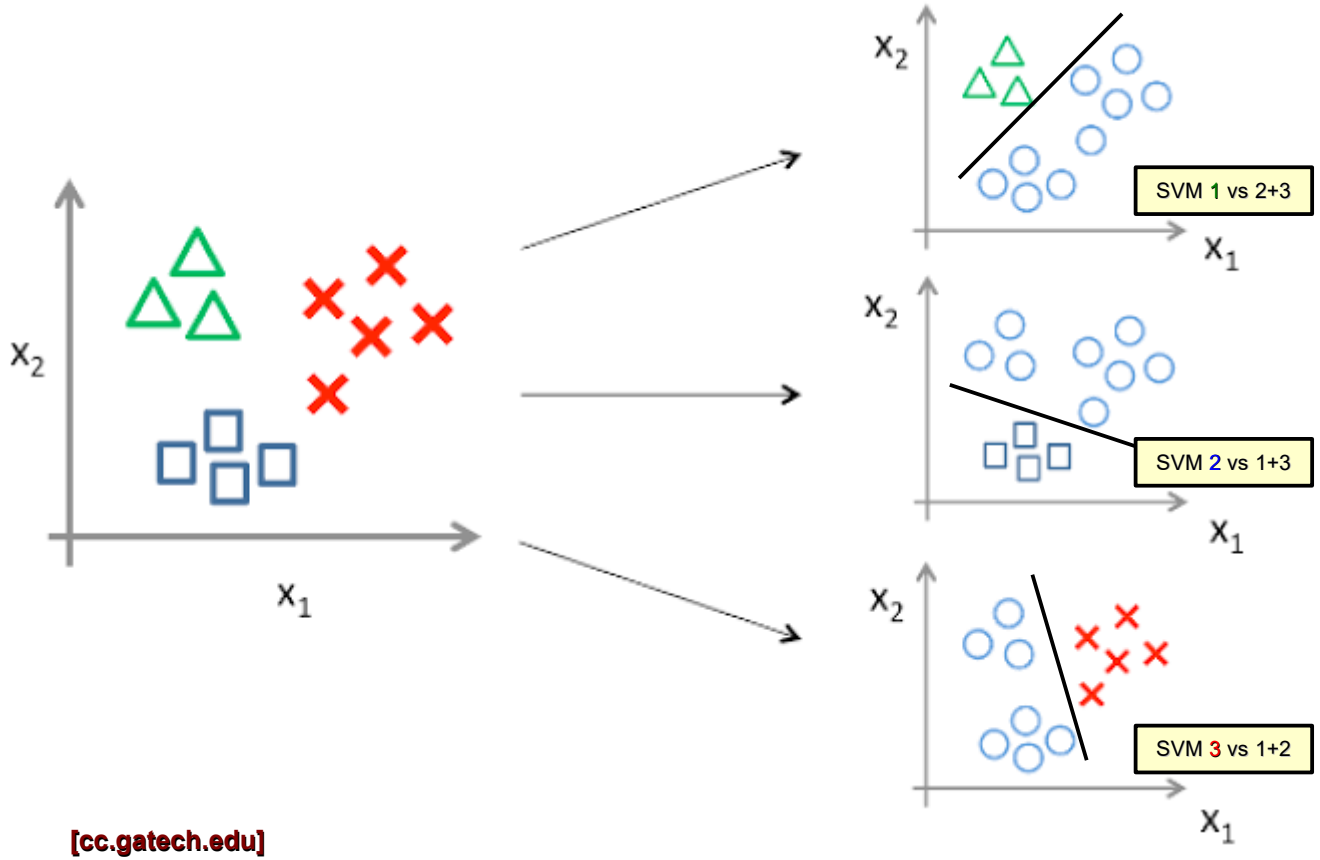


[cc.gatech.edu]

Klasyfikacja: metoda wektorów nośnych (przypadek wielu klas)

SVM jest klasyfikatorem binarnym – użycie go dla danych zawierających więcej niż dwie klasy wymaga budowania wielu modeli klasyfikacyjnych. Przykładami dwóch podejść do tego problemu są:

- ***One vs Rest (OvR)*, *One vs All (OvA)* – podczas treningu powstaje K klasyfikatorów binarnych, budowanych w oparciu o wektory z jednej klasy wobec połączonych przykładów z reszty klas. Klasyfikowany wektor podawany jest na wejście wszystkich modeli i przypisywany do klasy, dla której uzyskuje się największą wartość prawdopodobieństwa.**

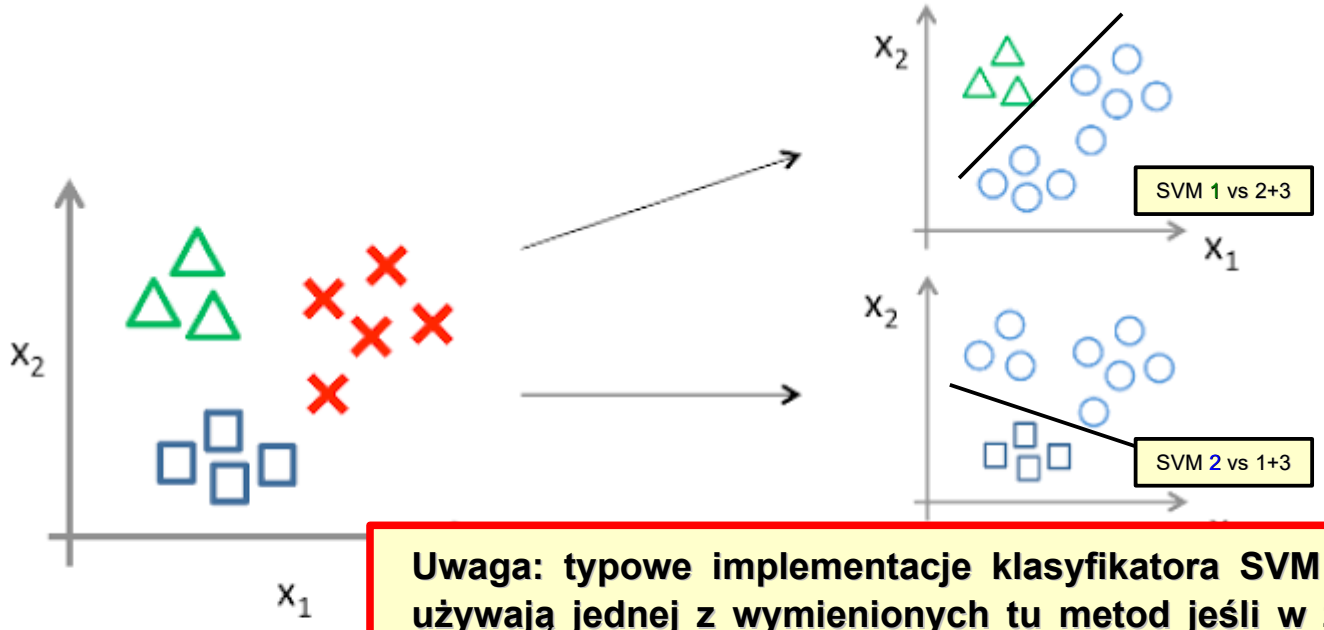


[cc.gatech.edu]

Klasyfikacja: metoda wektorów nośnych (przypadek wielu klas)

SVM jest klasyfikatorem binarnym – użycie go dla danych zawierających więcej niż dwie klasy wymaga budowania wielu modeli klasyfikacyjnych. Przykładami dwóch podejść do tego problemu są:

- ***One vs Rest (OvR)*, *One vs All (OvA)* – podczas treningu powstaje K klasyfikatorów binarnych, budowanych w oparciu o wektory z jednej klasy wobec połączonych przykładów z reszty klas. Klasyfikowany wektor podawany jest na wejście wszystkich modeli i przypisywany do klasy, dla której uzyskuje się największą wartość prawdopodobieństwa.**

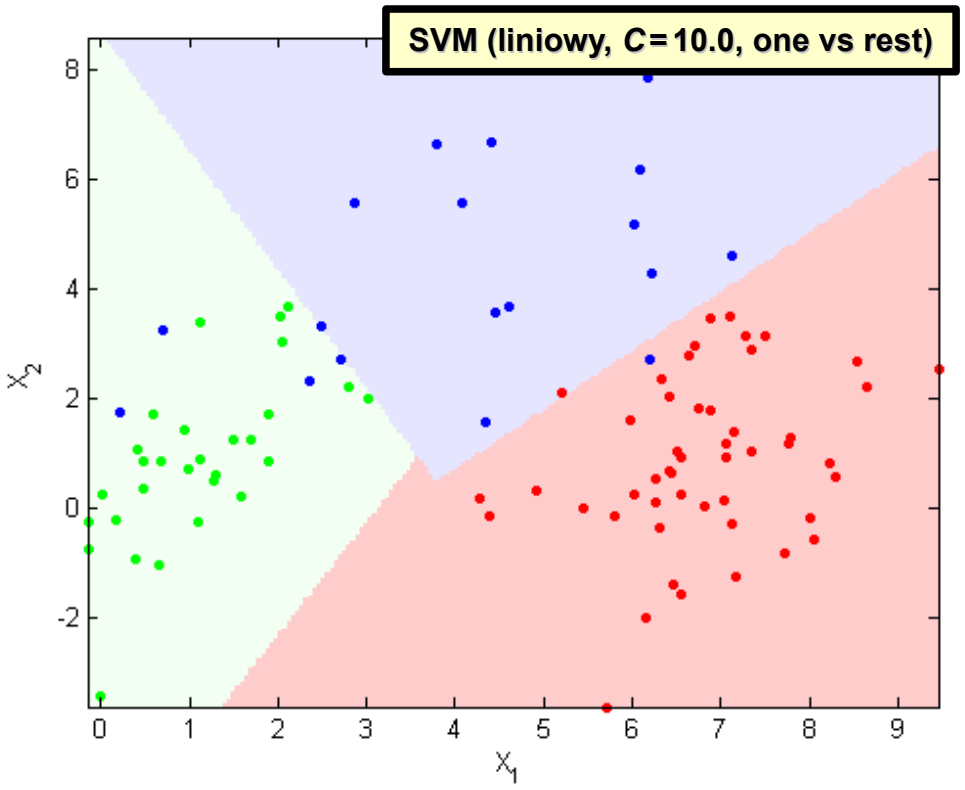


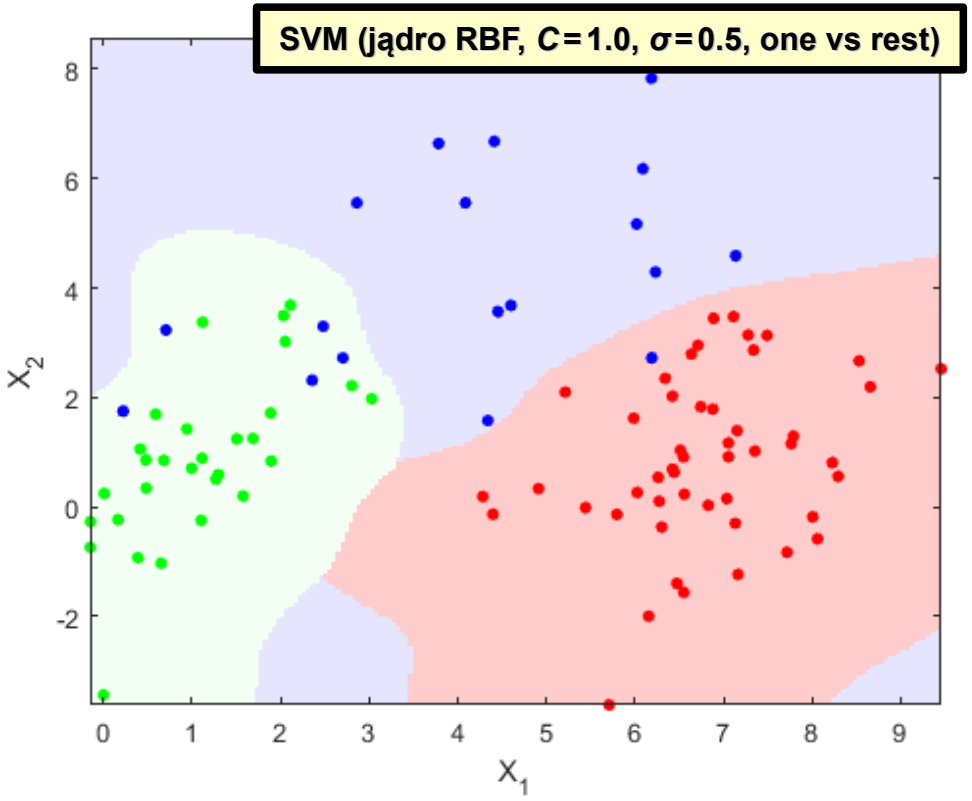
Uwaga: typowe implementacje klasyfikatora SVM automatycznie używają jednej z wymienionych tu metod jeśli w zbiorze danych występują więcej niż dwie klasy.

Warto też zaznaczyć, że z podejść OvO i OvR można użyć nie tylko dla SVM, ale również dla dowolnego klasyfikatora binarnego.

[cc.gatech.edu]

Klasyfikacja: metoda wektorów nośnych (podsumowanie)



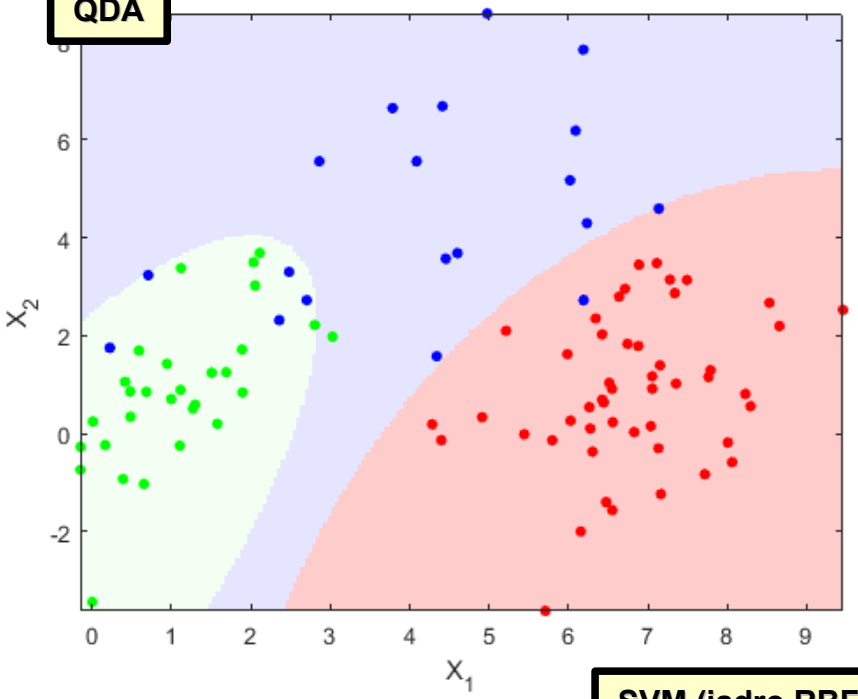


Uczenie maszynowe w bioinformatyce

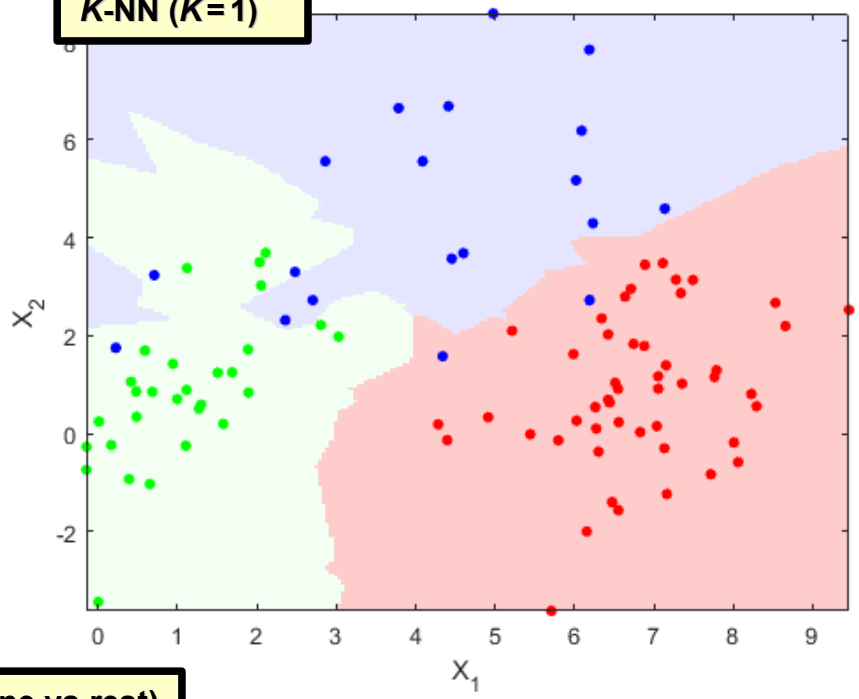
Wykład 8: ocena jakości klasyfikatora

Klasyfikacja: ocena klasyfikatorów

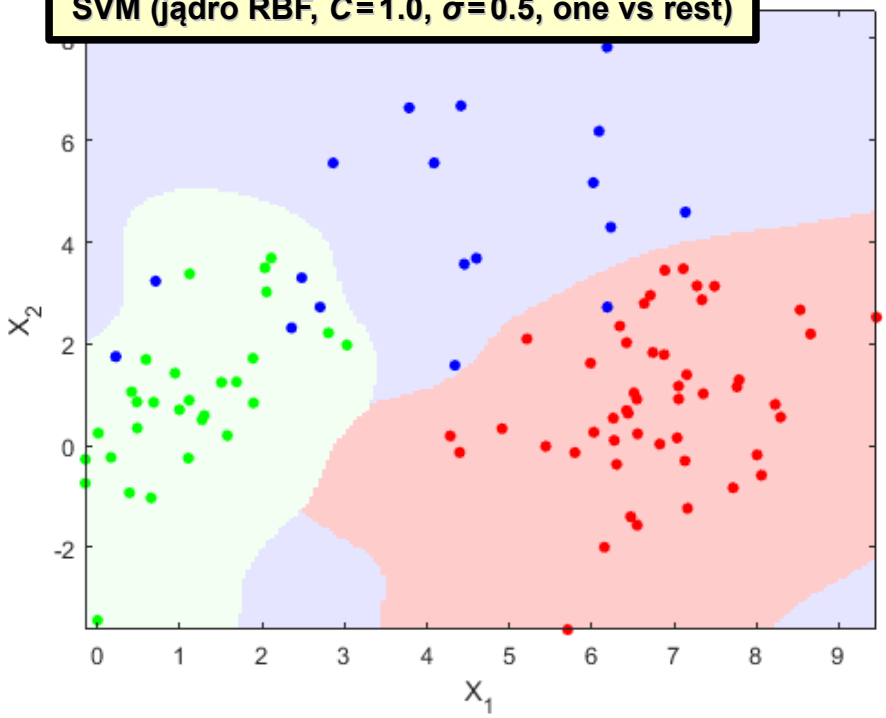
QDA



***K*-NN (*K*=1)**



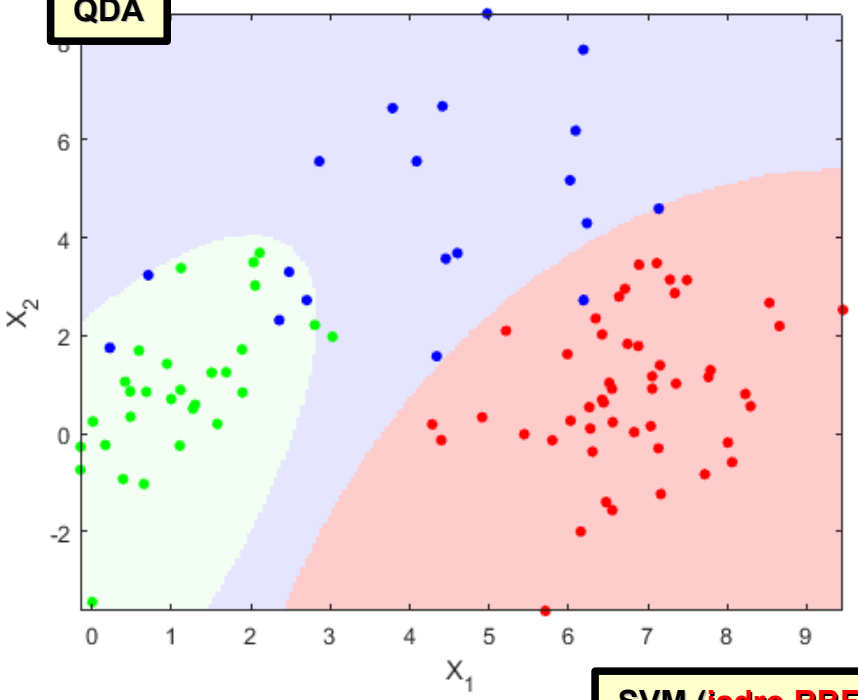
SVM (jądro RBF, *C*=1.0, σ=0.5, one vs rest)



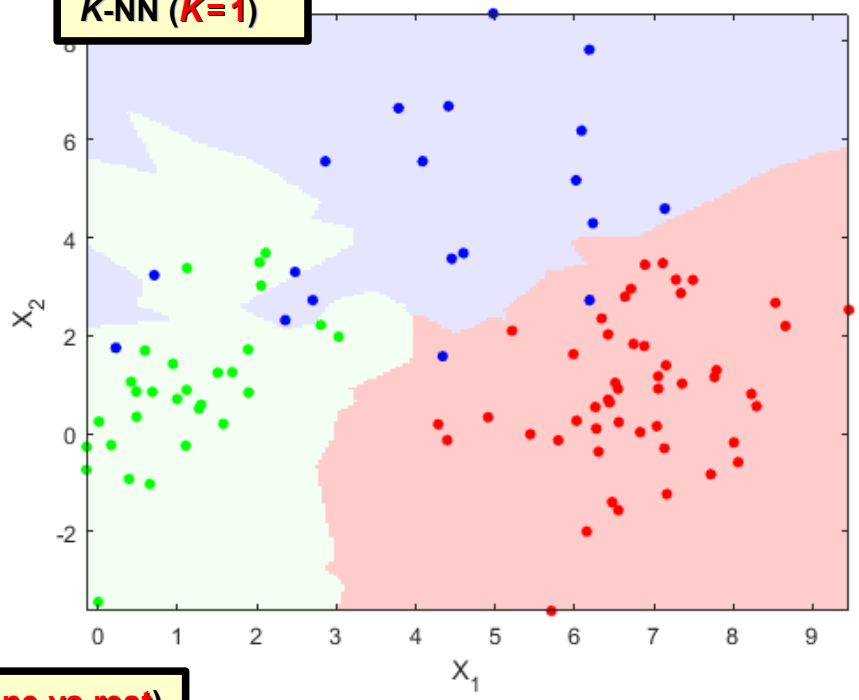
Który klasyfikator jest lepszy?

Klasyfikacja: ocena klasyfikatorów

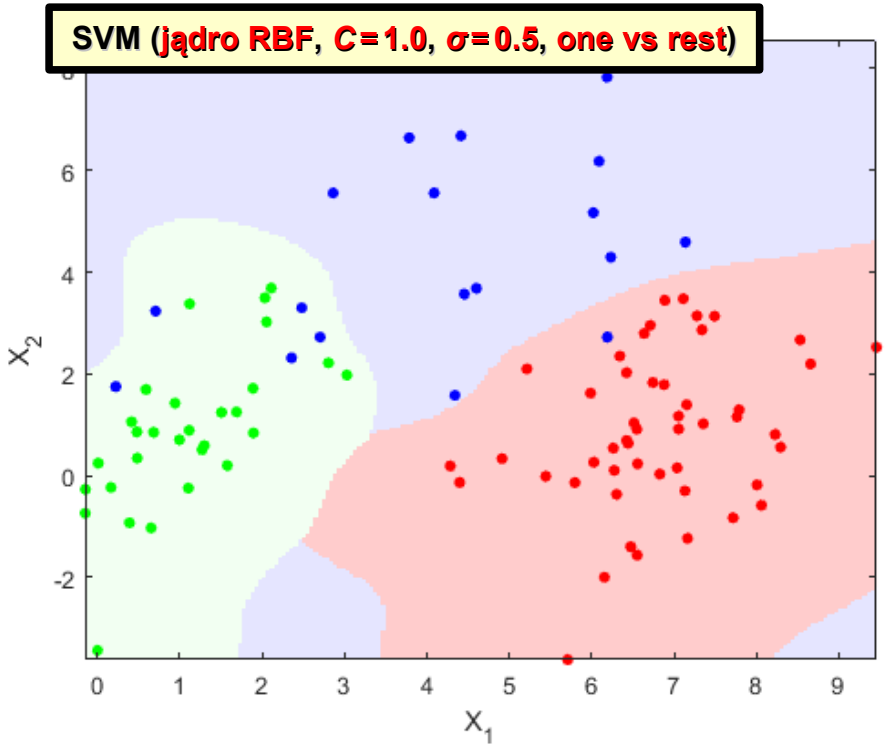
QDA



K-NN (*K*=1)



SVM (jądro RBF, $C=1.0$, $\sigma=0.5$, one vs rest)



Jak dobrać parametry?

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Miarą jakości rzeczywistego klasyfikatora $\hat{d}(x)$, zbudowanego na podstawie próby uczącej L_N jest błąd:

$$\hat{e} = e\left(\hat{d}\right) = \mathrm{P}\left(\hat{d}(\boldsymbol{X}) \neq Y \,|\, L_N\right)$$

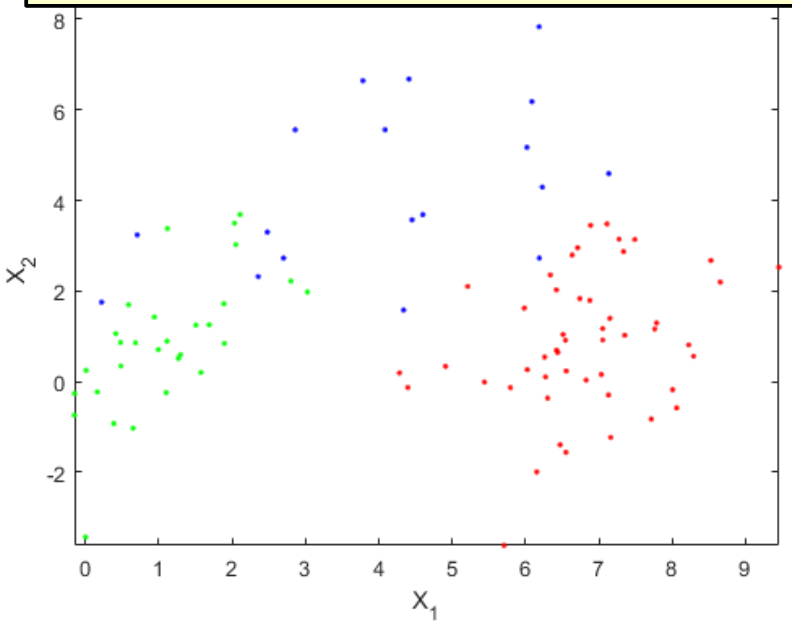
Jest on zmienną losową, zależną od próby uczącej. Dlatego, inaczej niż w przypadku klasyfikatora idealnego, poziom błędu klasyfikatora rzeczywistego nie jest dany bezpośrednio i musi być estymowany.

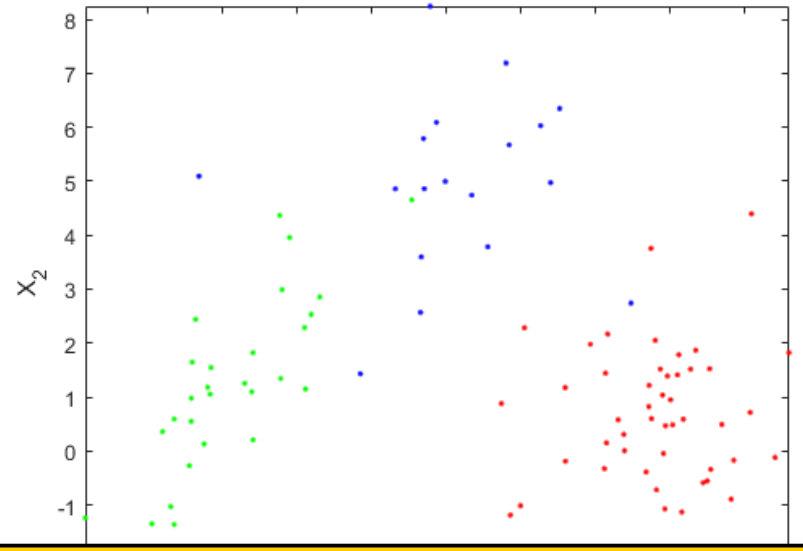
W idealnym przypadku estymacja błędu powinna odbywać się na zbiorze testowym złożonym z M wektorów cech pobranych w sposób niezależny od zbioru uczącego.

Zbiór uczący

$$L_N = \{(x_{11}, y_1), (x_{12}, y_1), \ldots, (x_{1N_1}, y_1), \ldots, (x_{K1}, y_K), (x_{K2}, y_K), \ldots, (x_{KN_K}, y_K)\}$$

$$N = N_1 + N_2 + \ldots + N_K$$





Zbiór testowy

$$T_M = \{(x^t_{11}, y_1), (x^t_{12}, y_1), \ldots, (x^t_{1M_1}, y_1), \ldots, (x^t_{K1}, y_K), (x^t_{K2}, y_K), \ldots, (x^t_{KM_K}, y_K)\}$$

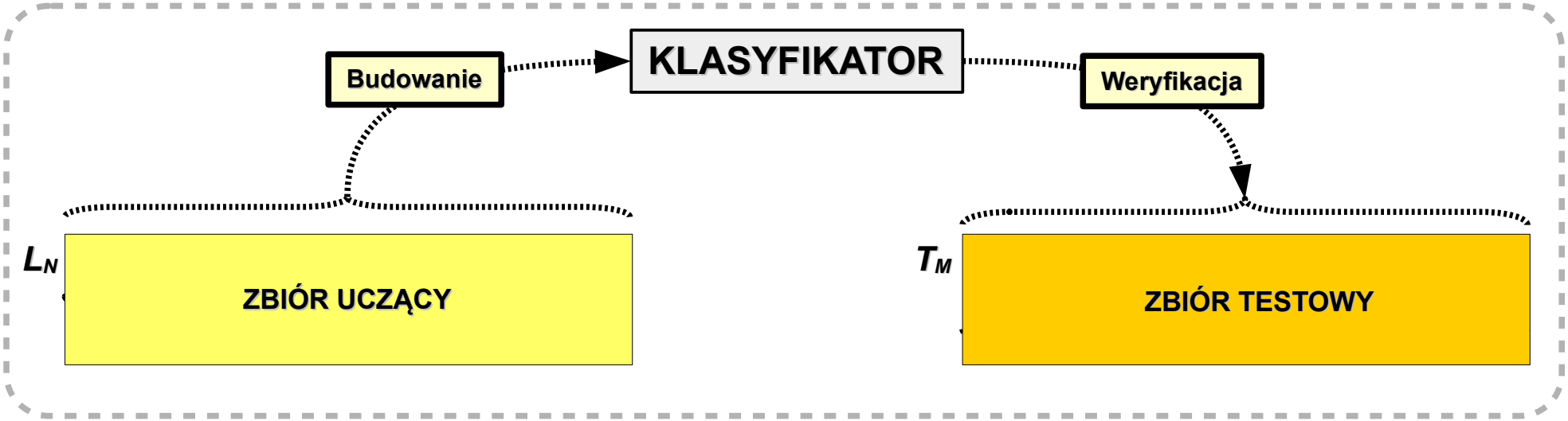
$$M = M_1 + M_2 + \ldots + M_K$$

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Dysponując niezależnym zbiorem testowym poziom błędu klasyfikatora $\hat{d}(x)$ można estymować jako liczbę nieprawidłowo zaklasyfikowanych wektorów z tego zbioru, odniesioną do jego liczebności:

$$\hat{e_{T_M}} = \frac{1}{M} \sum_{k=1}^{K} \sum_{j=1}^{M_k} I\left(\hat{d}\left(x_{kj}^{t}\right) \neq y_k\right)$$

gdzie $I(A)$ jest funkcją indykatorową, przyjmująca wartość 1 gdy spełniony jest warunek A oraz wartość 0 w przeciwnym wypadku.

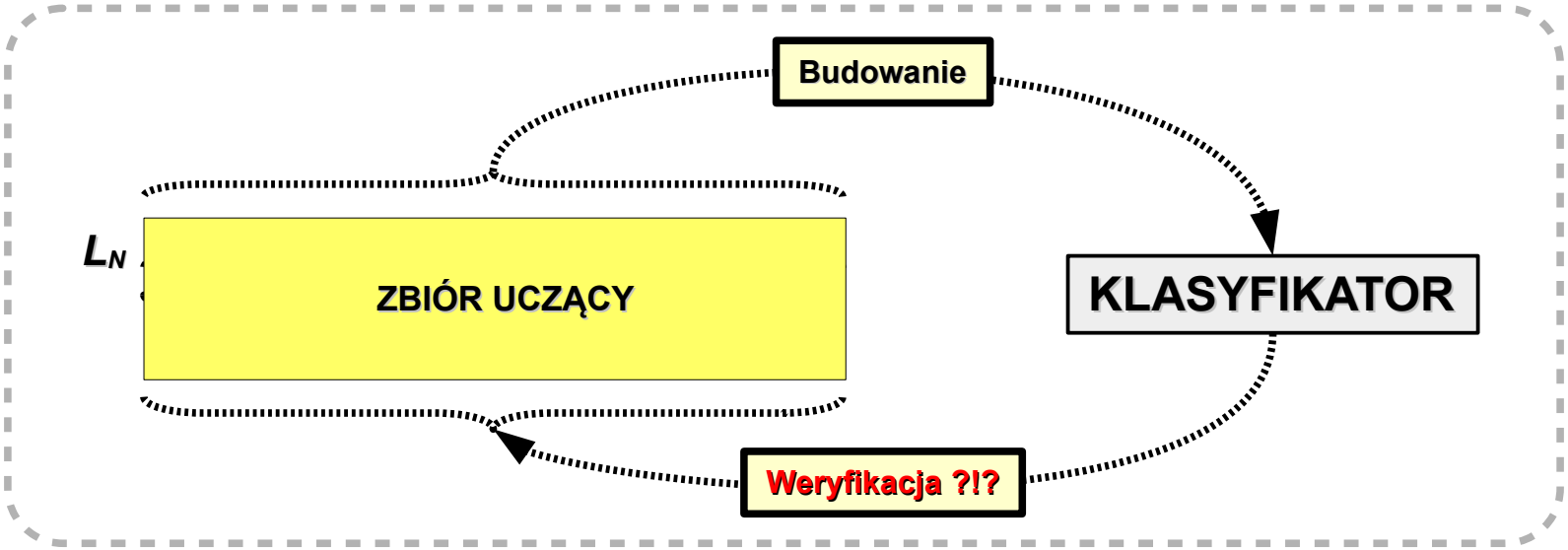


Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

W przypadku braku niezależnego zbioru testowego (co niestety jest bardzo często spotykaną sytuacją) możliwe zastosowanie jednej z metod estymacji poziomu błędu rzeczywistego klasyfikatora w oparciu o zbiór uczący.

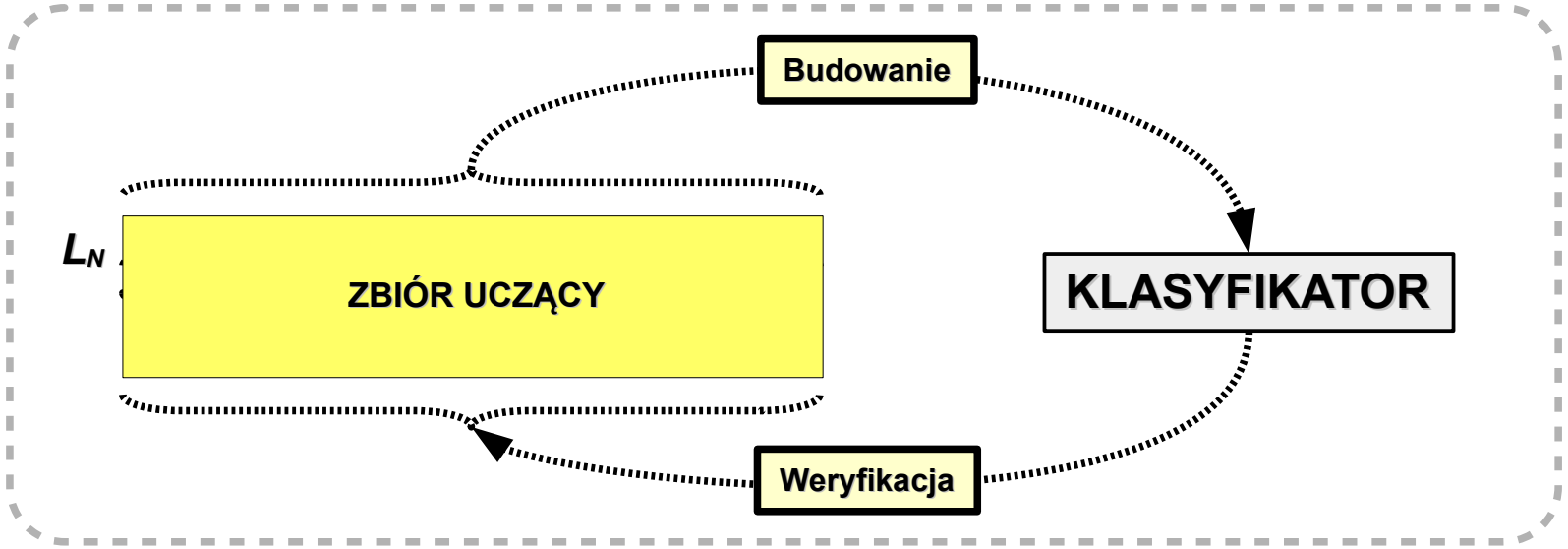
Przykładami tego typu metod są:

- **resubstytucja;**
- **jednokrotny losowy podział na część uczącą i testową (metoda *hold-out*);**
- **różne schematy walidacji krzyżowej, w tym *V*-krotna i *leave-one-out*.**



Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Najprostszym (i zdecydowanie niepolecanym) sposobem estymacji błędu w oparciu o zbiór uczący jest resubstytucja, czyli wykonanie klasyfikacji wszystkich wektorów, które posłużyły do budowy klasyfikatora.



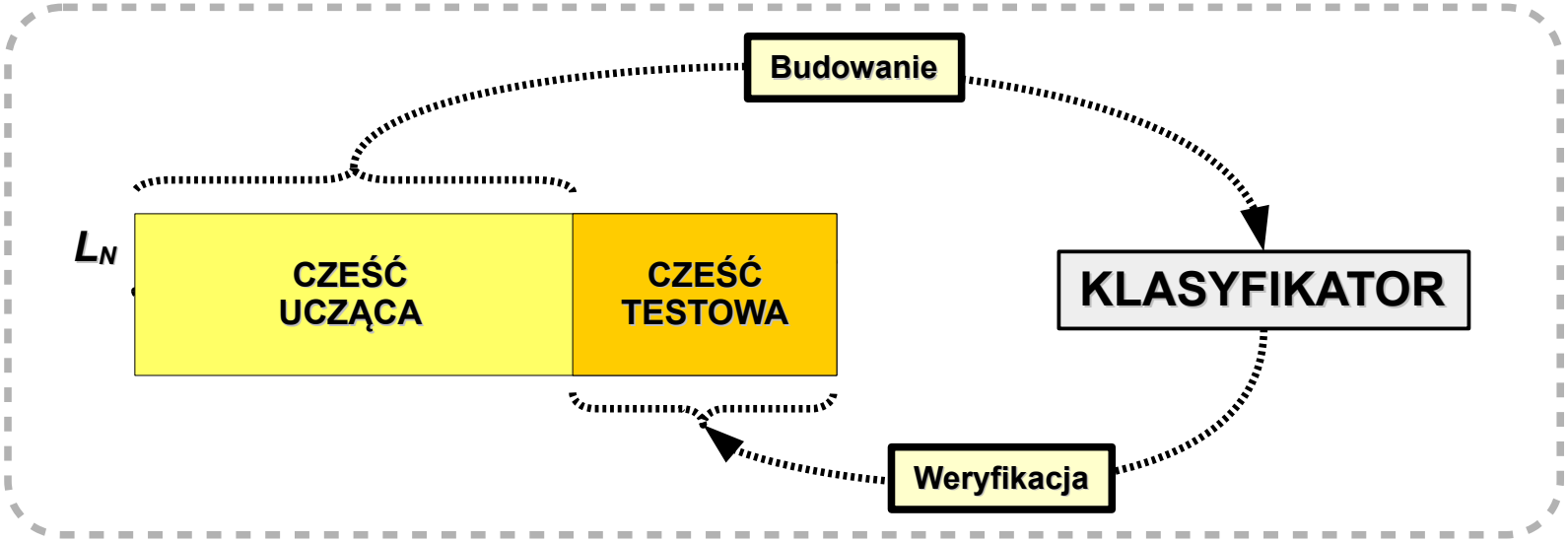
Wartość błędu wyznaczana jest wówczas za pomocą wyrażenia analogicznego jak w przypadku gdy dysponujemy niezależnym zbiorem testowym:

$$\hat{e}_{L_N} = \frac{1}{N}\sum_{k=1}^{K}\sum_{j=1}^{N_k} I\left(\hat{d}\left(x_{kj}\right) \neq y_k\right)$$

W tej metodzie klasyfikator jest weryfikowany na tych samych danych, na których przebiegało jego uczenie. Jest to przyczyną znacznego obciążenia estymatora błędu, który ma tendencje do zaniżania wartości tego ostatniego.

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Lepszym – choć wciąż niedoskonałym – podejściem do szacowania poziomu błędu jest metoda *hold-out*, czyli jednokrotny losowy podział zbioru na dwie części: uczącą i testową. Na podstawie pierwszej z nich budowany jest klasyfikator, a druga służy do jego weryfikacji.

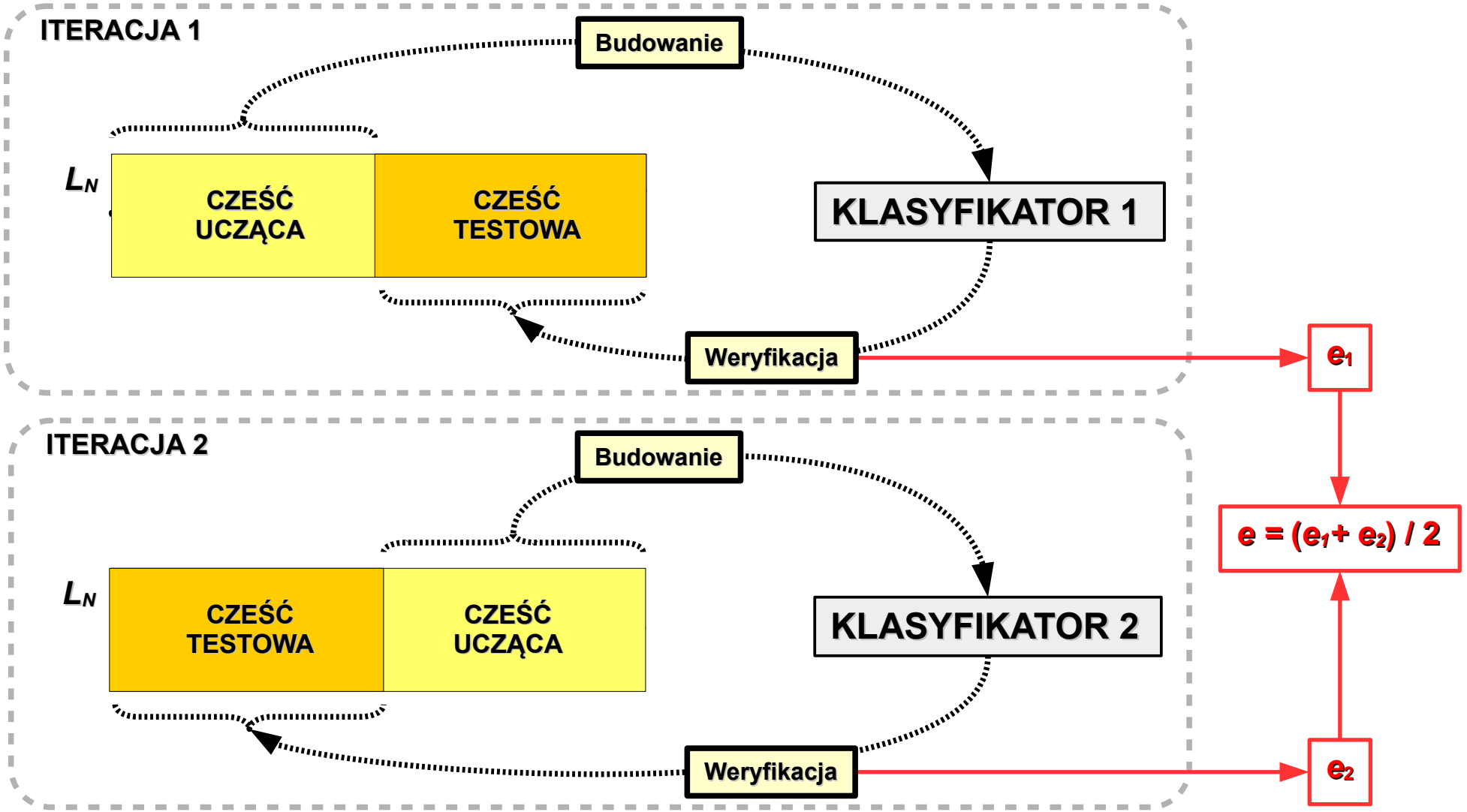


Błąd liczony jest tak samo jak poprzednio, z tym że jedynie na podstawie wektorów z testowej części zbioru danych.

	MATLAB	R
Podział danych:	`crossvalind`	`pakiet CARET`

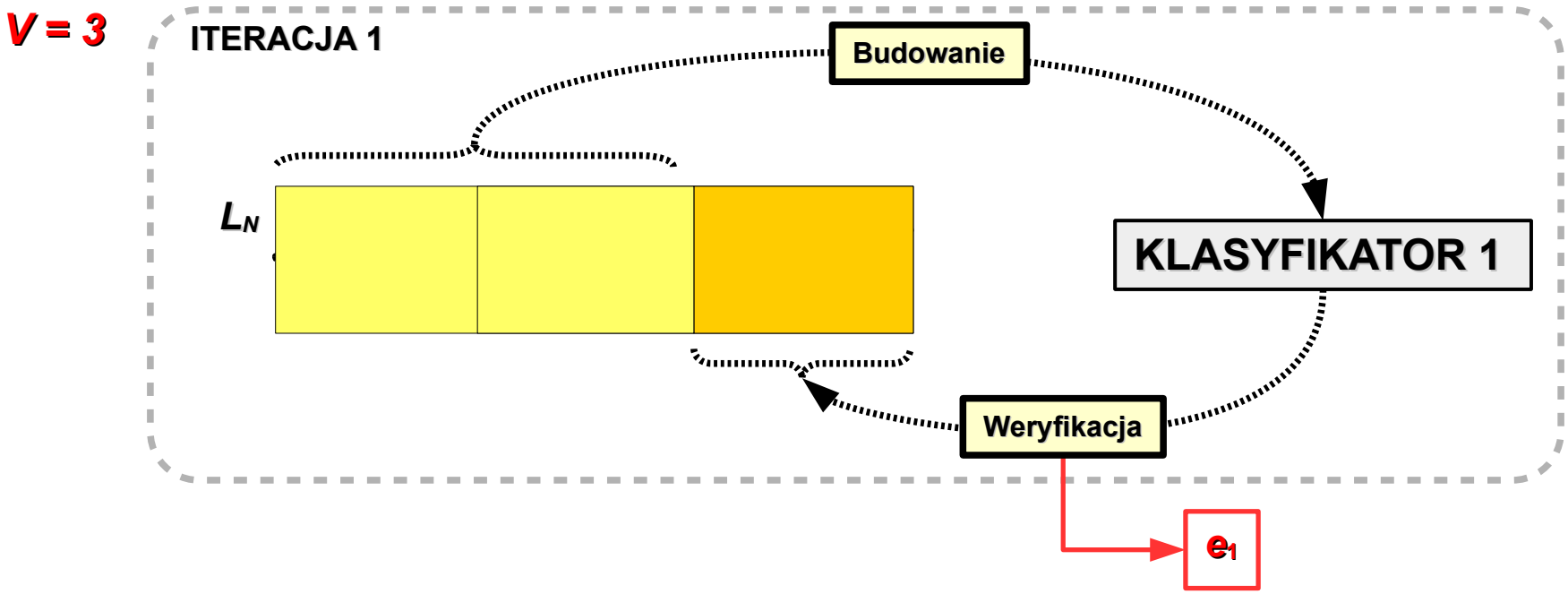
Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Alternatywnym podejściem jest użycie jednej z odmian walidacji krzyżowej (CV, *Cross-Validation*). W metodzie tej zbiór uczący jest iteracyjnie, w systematyczny sposób dzielony na dwie części, z których jedna służy do skonstruowania klasyfikatora, a druga do weryfikacji jego skuteczności. Ostateczną miara błędu staje się średnia błędów z poszczególnych podziałów zbioru uczącego.



Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

W przypadku *V*-krotnej walidacji krzyżowej (*V-fold cross-validation*) zbiór uczący jest dzielony na V równolicznych (lub w przybliżeniu równolicznych) części. W kolejnych V powtórzeniach jeden z tych podzbiorów staje się próba testową, podczas gdy V-1 służy do zbudowania klasyfikatora.

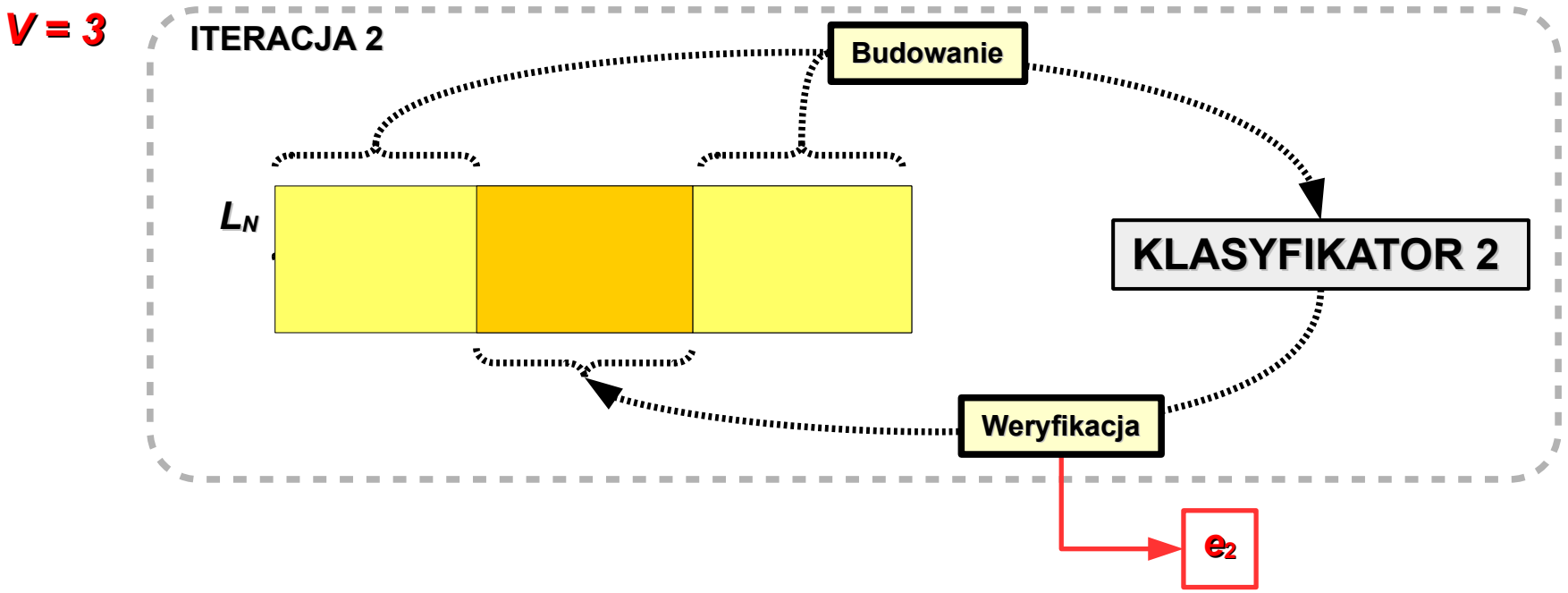


Wartością estymatora poziomu błędu staje się średnia ważona błędów w kolejnych iteracjach, przy czym wagami są liczebności używanych w nich podzbiorów (ważenie potrzebne jest w przypadku gdy liczba wektorów w zbiorze uczącym nie jest wielokrotnością V, co oznacza, że podzbiory nie są równoliczne).

	MATLAB	R
Podział danych:	`crossvalind`	`pakiet CARET`

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

W przypadku *V*-krotnej walidacji krzyżowej (*V-fold cross-validation*) zbiór uczący jest dzielony na V równolicznych (lub w przybliżeniu równolicznych) części. W kolejnych V powtórzeniach jeden z tych podzbiorów staje się próba testową, podczas gdy V-1 służy do zbudowania klasyfikatora.

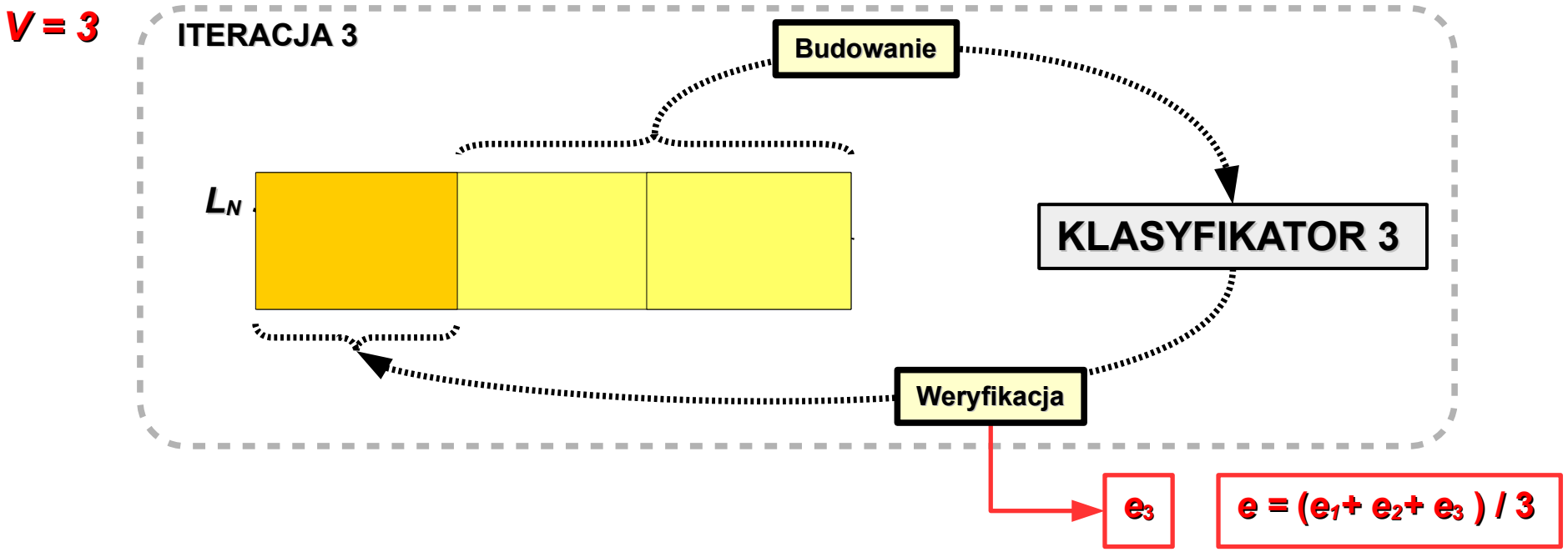


Wartością estymatora poziomu błędu staje się średnia ważona błędów w kolejnych iteracjach, przy czym wagami są liczebności używanych w nich podzbiorów (ważenie potrzebne jest w przypadku gdy liczba wektorów w zbiorze uczącym nie jest wielokrotnością V, co oznacza, że podzbiory nie są równoliczne).

	MATLAB	R
Podział danych:	`crossvalind`	`pakiet CARET`

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

W przypadku *V*-krotnej walidacji krzyżowej (*V-fold cross-validation*) zbiór uczący jest dzielony na V równolicznych (lub w przybliżeniu równolicznych) części. W kolejnych V powtórzeniach jeden z tych podzbiorów staje się próba testową, podczas gdy V-1 służy do zbudowania klasyfikatora.



Wartością estymatora poziomu błędu staje się średnia ważona błędów w kolejnych iteracjach, przy czym wagami są liczebności używanych w nich podzbiorów (ważenie potrzebne jest w przypadku gdy liczba wektorów w zbiorze uczącym nie jest wielokrotnością V, co oznacza, że podzbiory nie są równoliczne).

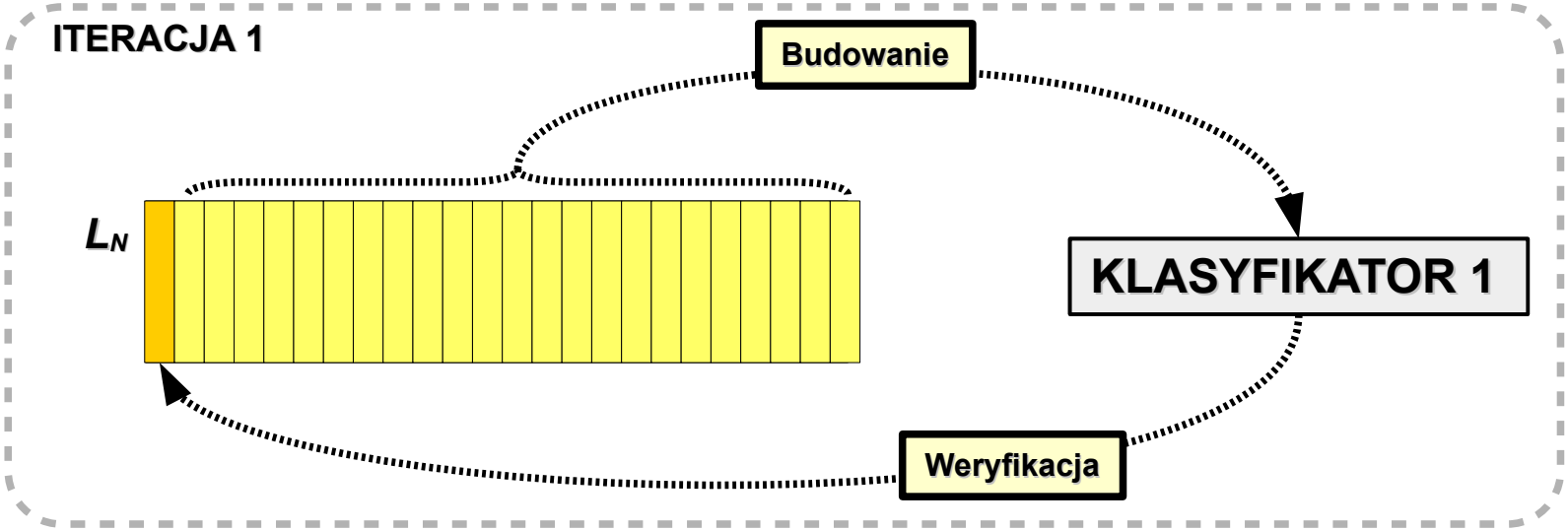
	MATLAB	R
Podział danych:	`crossvalind`	`pakiet CARET`

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Walidacja krzyżowa *leave-one-out* przy użyciu N-elementowego zbioru uczącego polega na N-krotnym wykonaniu następującej procedury:

- usunięcie ze zbioru uczącego wektora x_i (w każdym powtórzeniu innego, tak aby po procesie walidacji każdy z wektorów został jednokrotnie wykluczony ze zbioru);
- zbudowanie klasyfikatora na pomniejszonym zbiorze uczącym;
- wykonanie klasyfikacji wektora x_i i zapamiętanie, czy dała ona prawidłowy wynik.

Oznacza to, że w każdej iteracji pojedynczy wektor staje się jednoelementowym zbiorem testowym, niezależnym od części uczącej o liczebności N-1. Estymatą błędu staje się liczba nieprawidłowych klasyfikacji, odniesiona do N.



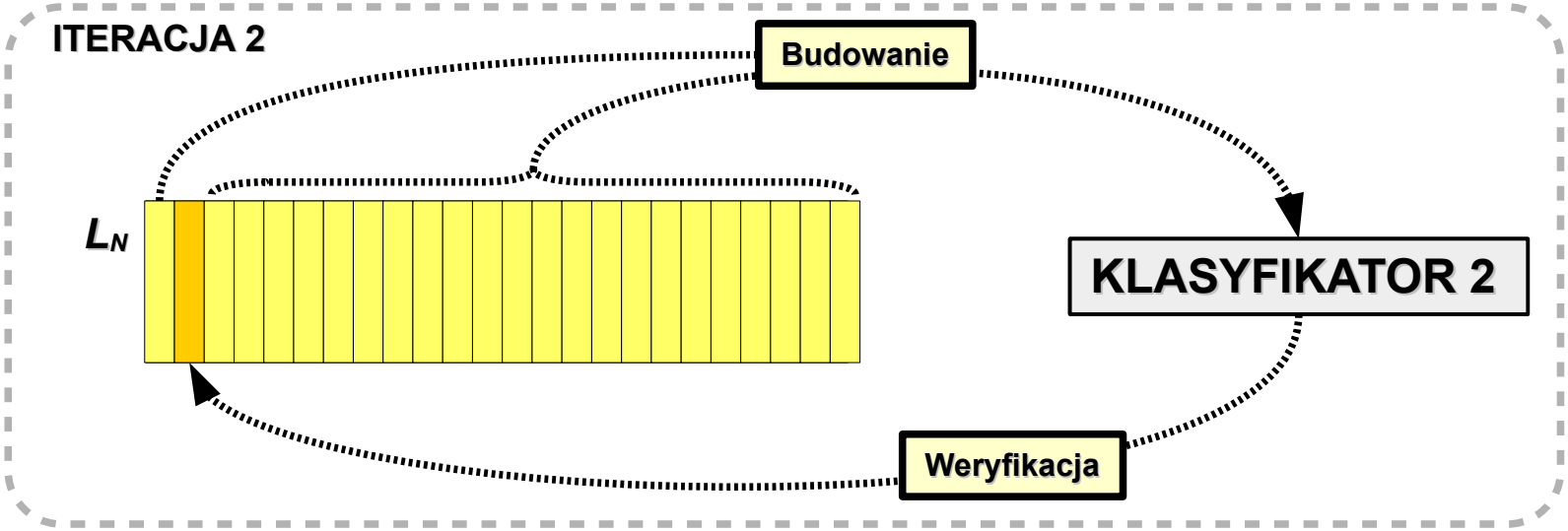
	MATLAB	R
Podział danych:	`crossvalind`	pakiet `CARET`

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Walidacja krzyżowa *leave-one-out* przy użyciu N-elementowego zbioru uczącego polega na N-krotnym wykonaniu następującej procedury:

- usunięcie ze zbioru uczącego wektora x_i (w każdym powtórzeniu innego, tak aby po procesie walidacji każdy z wektorów został jednokrotnie wykluczony ze zbioru);
- zbudowanie klasyfikatora na pomniejszonym zbiorze uczącym;
- wykonanie klasyfikacji wektora x_i i zapamiętanie, czy dała ona prawidłowy wynik.

Oznacza to, że w każdej iteracji pojedynczy wektor staje się jednoelementowym zbiorem testowym, niezależnym od części uczącej o liczebności N-1. Estymatą błędu staje się liczba nieprawidłowych klasyfikacji, odniesiona do N.



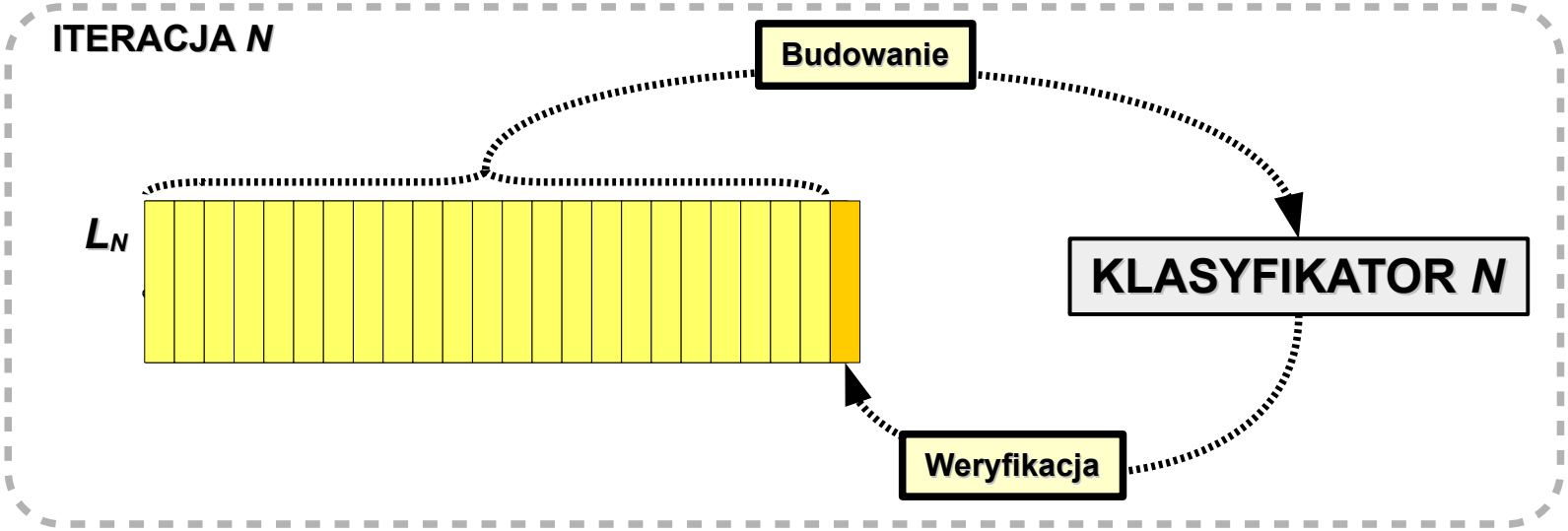
	MATLAB	R
Podział danych:	`crossvalind`	pakiet `CARET`

Klasyfikacja: poziom błędu klasyfikatora rzeczywistego

Walidacja krzyżowa *leave-one-out* przy użyciu N-elementowego zbioru uczącego polega na N-krotnym wykonaniu następującej procedury:

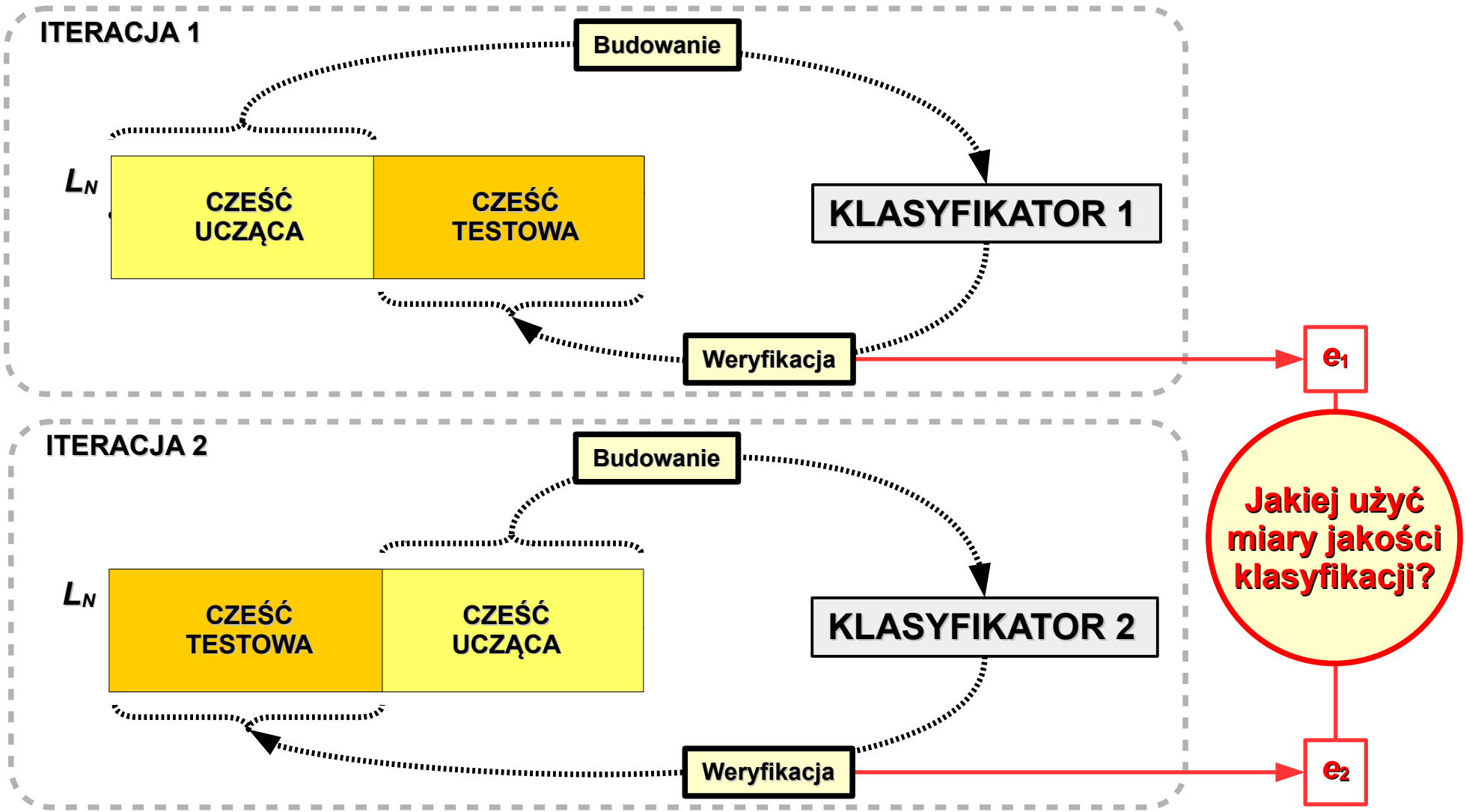
- usunięcie ze zbioru uczącego wektora x_i (w każdym powtórzeniu innego, tak aby po procesie walidacji każdy z wektorów został jednokrotnie wykluczony ze zbioru);
- zbudowanie klasyfikatora na pomniejszonym zbiorze uczącym;
- wykonanie klasyfikacji wektora x_i i zapamiętanie, czy dała ona prawidłowy wynik.

Oznacza to, że w każdej iteracji pojedynczy wektor staje się jednoelementowym zbiorem testowym, niezależnym od części uczącej o liczebności N-1. Estymatą błędu staje się liczba nieprawidłowych klasyfikacji, odniesiona do N.



	MATLAB	R
Podział danych:	`crossvalind`	pakiet `CARET`

Klasyfikacja: problem wyboru miary jakości klasyfikatora



Uczenie maszynowe w bioinformatyce

Wykład 8: miary jakości klasyfikatora

Klasyfikacja: miary jakości klasyfikatora (macierz pomyłek)

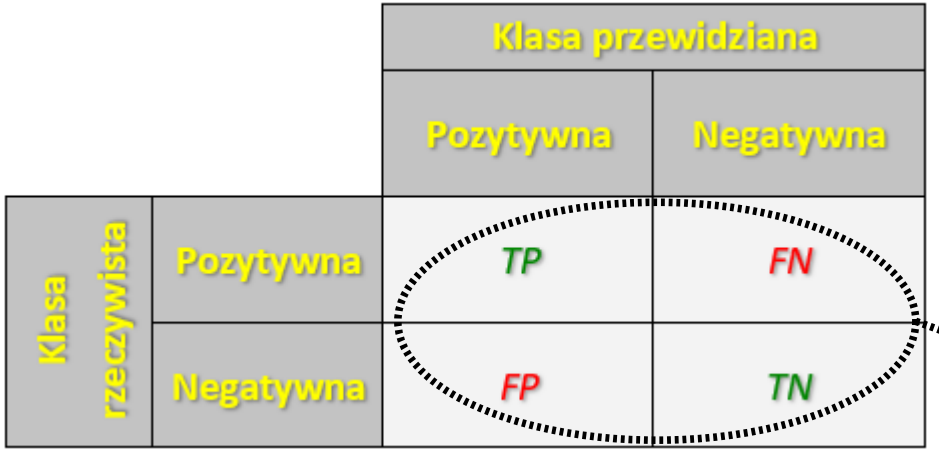
Na podstawie wyników klasyfikacji zbioru danych można utworzyć macierz pomyłek (*confusion matrix*), której element m_{ij} ma wartość równą liczbie wektorów z i-tej klasy przypisanych do klasy j.

W przypadku klasyfikacji binarnej (np. $Y = \{0, 1\}$ lub $Y = \{-1, 1\}$), gdy jedną z klas nazwie się „pozytywną”, a drugą „negatywną”, macierz ta ma widoczną poniżej postać.

		Klasa przewidziana	
		Pozytywna ($PN_P = TP + FP$)	Negatywna ($PN_N = TN + FN$)
Klasa Rzeczywista	Pozytywna ($N_P = TP + FN$)	Prawdziwie pozytywne (TP)	Fałszywie negatywne (FN)
	Negatywna ($N_N = TN + FP$)	Fałszywie pozytywne (FP)	Prawdziwie negatywne (TN)

Klasyfikacja: miary jakości klasyfikatora

Podstawowe miary jakości klasyfikatora obliczane na podstawie macierzy pomyłek.



***Error* (błąd klasyfikacji)**

$$e = 1 - ACC = \frac{FP + FN}{TP + TN + FP + FN}$$

***Accuracy* (dokładność klasyfikacji)**

$$ACC = 1 - e = \frac{TP + TN}{TP + TN + FP + FN}$$

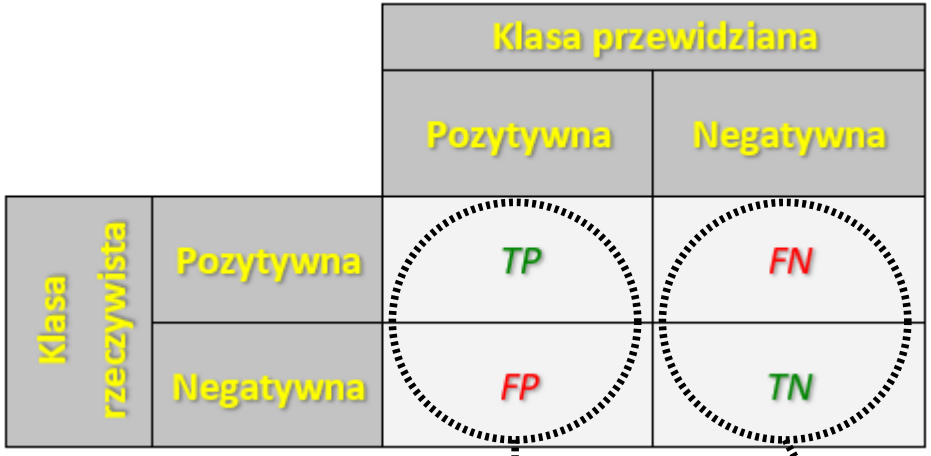
Matthews correlation coefficient

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP) \cdot (TP + FN) \cdot (TN + FP) \cdot (TN + FN)}}$$

	MATLAB	R
Miary jakości:	`classperf`	`confusionMatrix`

Klasyfikacja: miary jakości klasyfikatora

Podstawowe miary jakości klasyfikatora obliczane na podstawie macierzy pomyłek.



Negative predictive value

$$NPV = \frac{TN}{TN + FN}$$

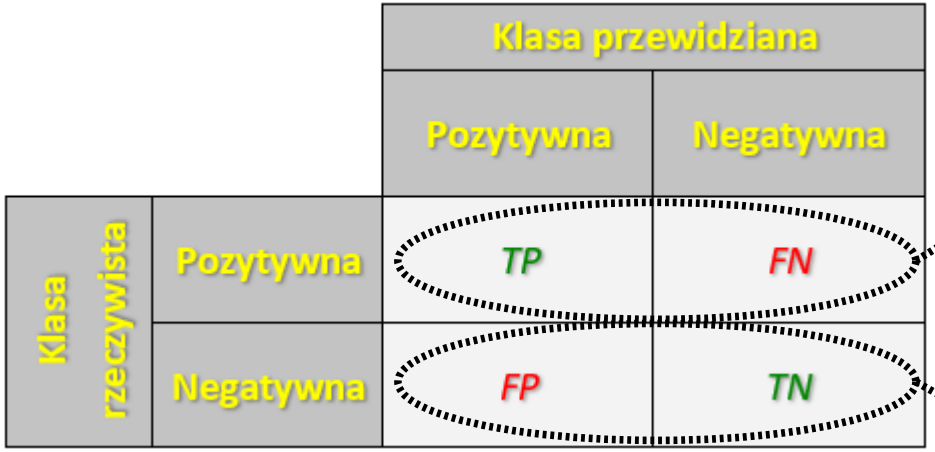
***Positive predictive value, precision* (precyzja)**

$$PPV = \frac{TP}{TP + FP}$$

	MATLAB	R
Miary jakości:	`classperf`	`confusionMatrix`

Klasyfikacja: miary jakości klasyfikatora

Podstawowe miary jakości klasyfikatora obliczane na podstawie macierzy pomyłek.



***True positive rate*, *recall*, *sensitivity* (czułość)**

$$TPR = SE = \frac{TP}{TP + FN}$$

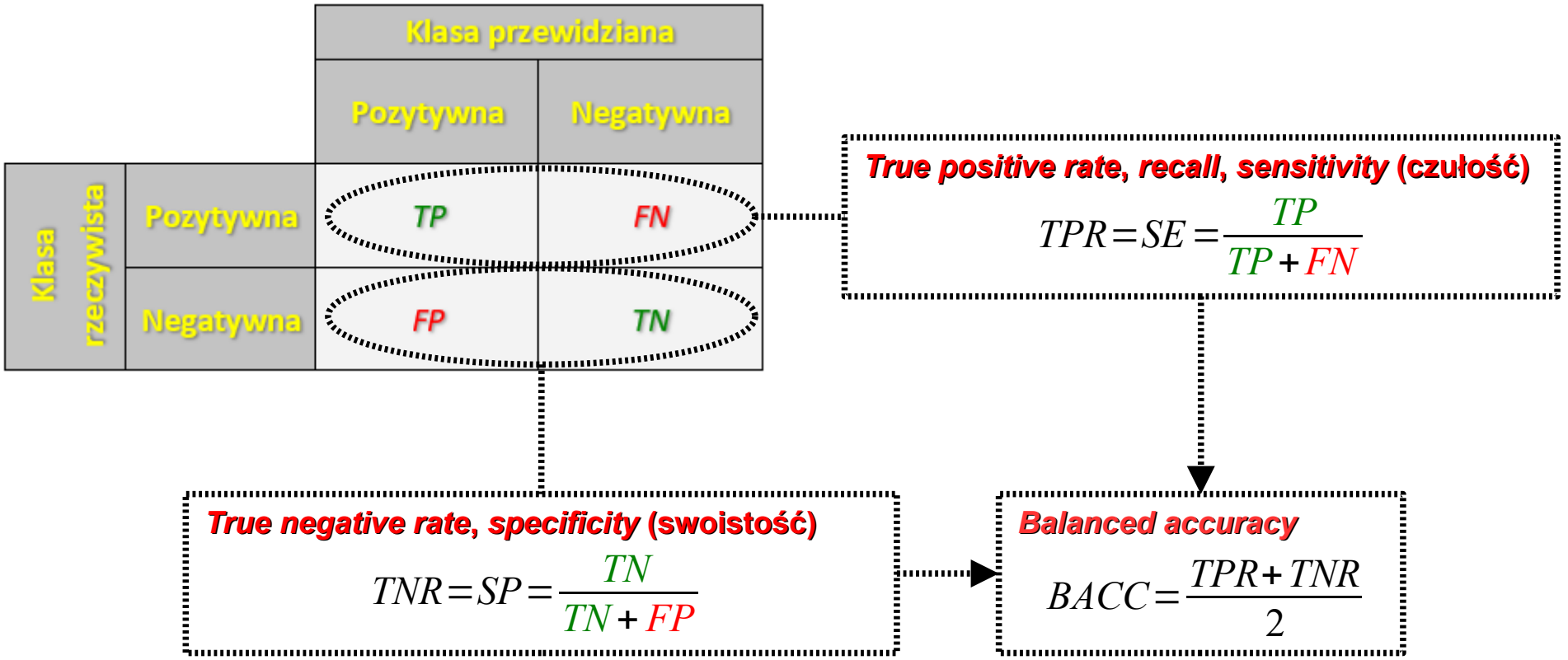
***True negative rate*, *specificity* (swoistość)**

$$TNR = SP = \frac{TN}{TN + FP}$$

	MATLAB	R
Miary jakości:	`classperf`	`confusionMatrix`

Klasyfikacja: miary jakości klasyfikatora

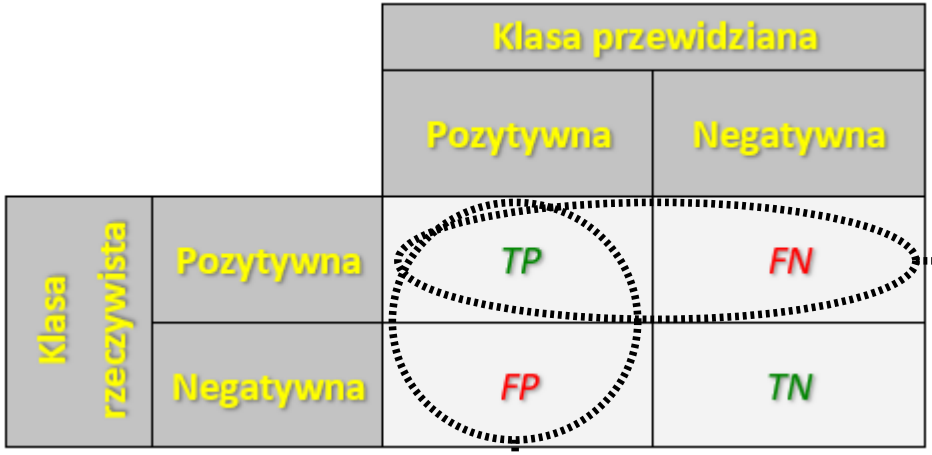
Podstawowe miary jakości klasyfikatora obliczane na podstawie macierzy pomyłek.



	MATLAB	R
Miary jakości:	`classperf`	`confusionMatrix`

Klasyfikacja: miary jakości klasyfikatora

Podstawowe miary jakości klasyfikatora obliczane na podstawie macierzy pomyłek.



***True positive rate*, *recall*, *sensitivity* (czułość)**

$$TPR = SE = \frac{TP}{TP + FN}$$

***Positive predictive value*, *precision* (precyzja)**

$$PPV = \frac{TP}{TP + FP}$$

***F*$_1$ *score* (miara *F*$_1$)**

$$F_1 = \frac{2}{PPV^{-1} + TPR^{-1}} = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR}$$

	MATLAB	R
Miary jakości:	`classperf`	`confusionMatrix`

Klasyfikacja: miary jakości klasyfikatora

Podstawowe miary jakości klasyfikatora obliczane na podstawie macierzy pomyłek.

		Klasa przewidziana: Pozytywna	Klasa przewidziana: Negatywna
Klasa rzeczywista	Pozytywna	TP	FN
Klasa rzeczywista	Negatywna	FP	TN

***True positive rate, recall, sensitivity* (czułość)**

$$TPR = SE = \frac{TP}{TP + FN}$$

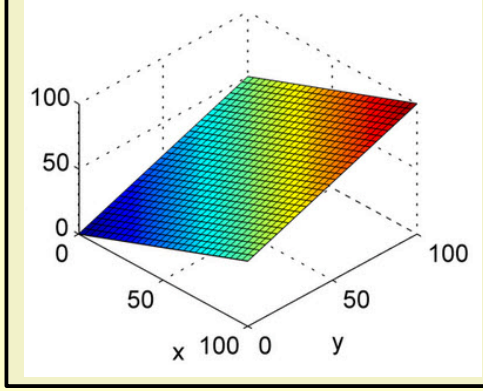
***Positive predictive value, precision* (precyzja)**

$$PPV = \frac{TP}{TP + FP}$$

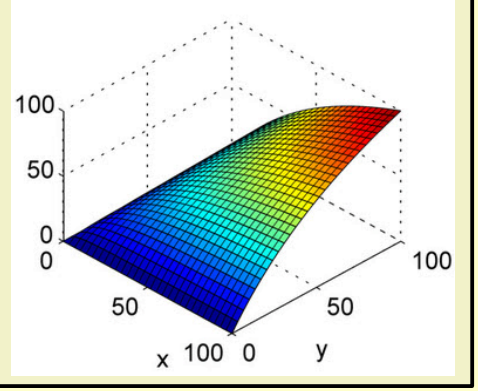
***F*$_1$ *score* (miara *F*$_1$)**

$$F_1 = \frac{2}{PPV^{-1} + TPR^{-1}} = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR}$$

Średnia arytmetyczna



Średnia harmoniczna

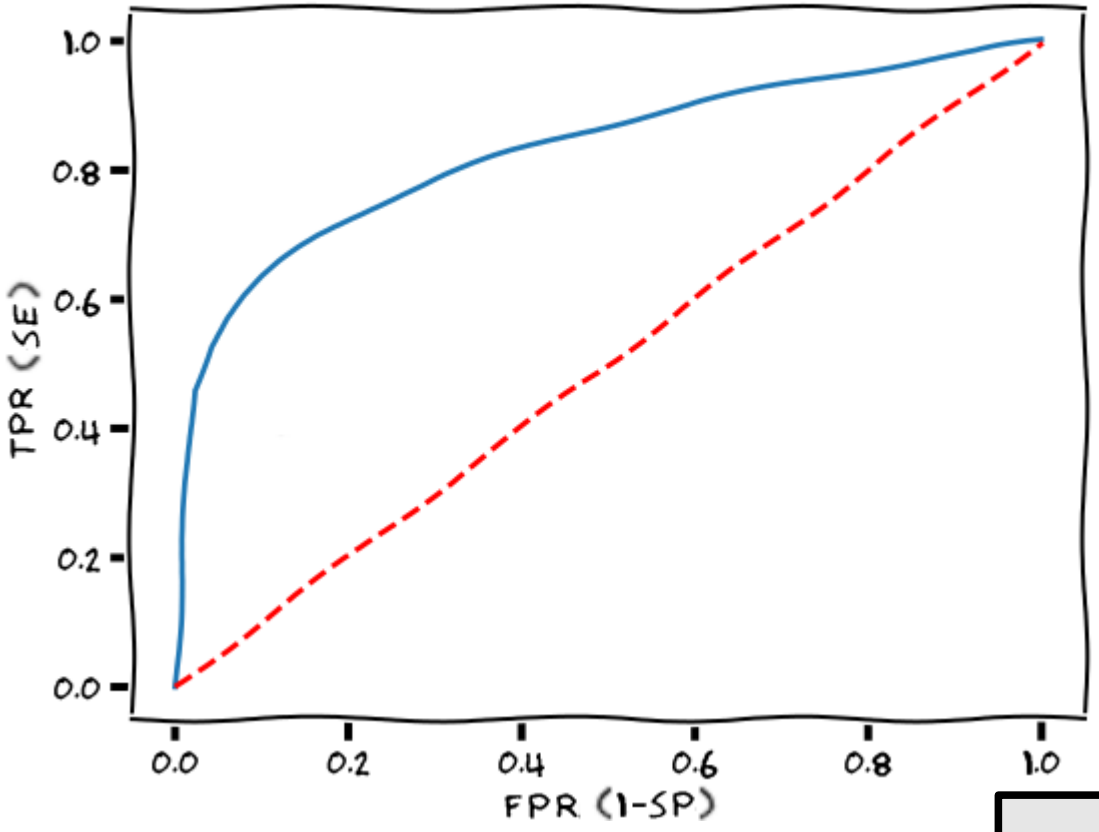


Miary jakości:	MATLAB	R
	`classperf`	`confusionMatrix`

Klasyfikacja: miary jakości klasyfikatora (krzywe ROC)

Krzywa ROC (*Receiver Operating Characteristic*) jest jedną z metod graficznej oceny klasyfikatora binarnego. Jest ona wykresem wskaźnika TPR (czułości, SE) wobec FPR (dopełnienia do jedynki swoistości, $1 - SP$) przy rosnącym progu wartości decyzyjnej, powyżej którego wektory ze zbioru danych są przypisywane do klasy pozytywnej.

Na podstawie wykresu można wyznaczyć AUC, czyli pole pod krzywą (*Area Under the Curve*), które jest często używaną liczbową miarą jakości klasyfikatora.



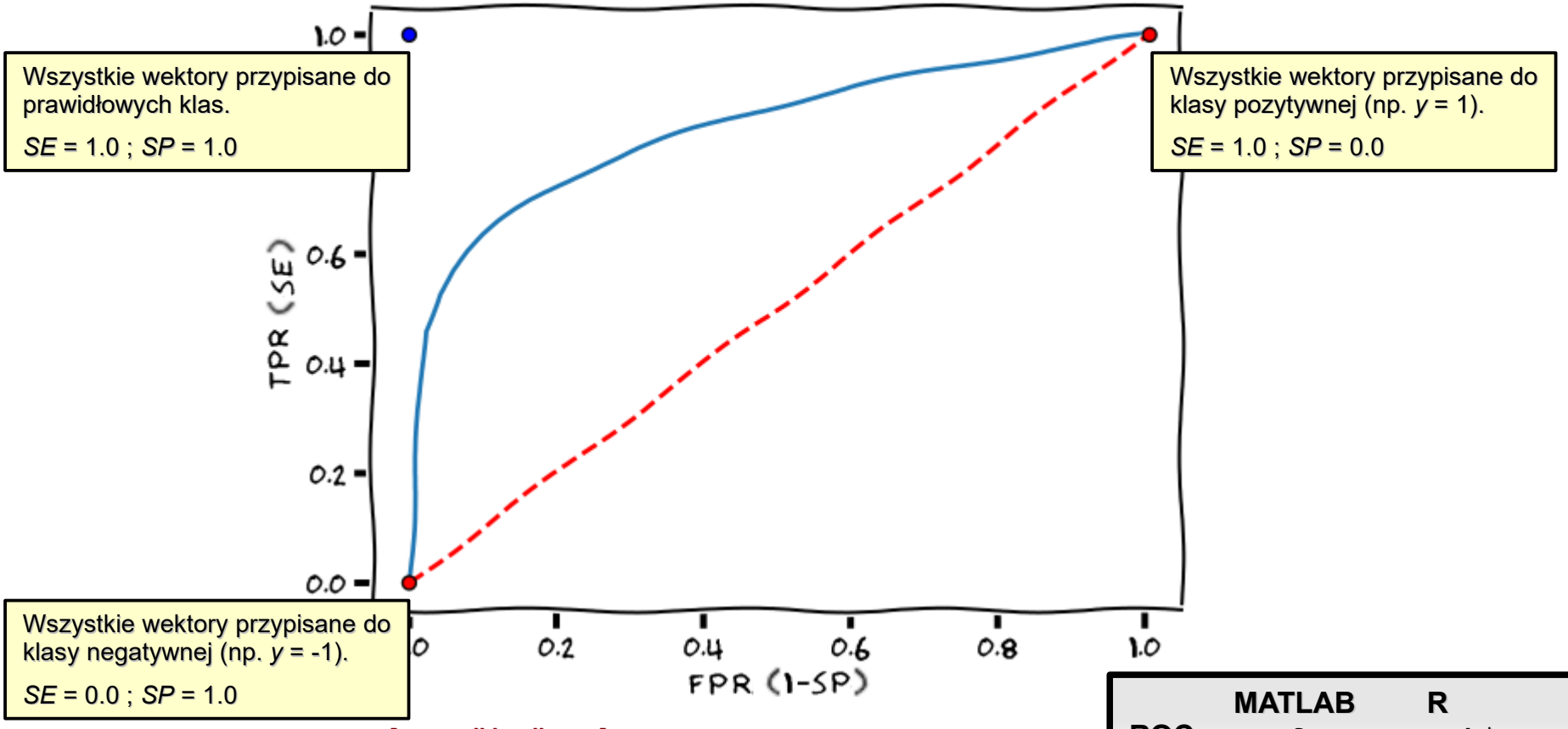
[www.wikipedia.org]

	MATLAB	R
ROC:	`perfcurve`	pakiet `ROCR`

Klasyfikacja: miary jakości klasyfikatora (krzywe ROC)

Krzywa ROC (*Receiver Operating Characteristic*) jest jedną z metod graficznej oceny klasyfikatora binarnego. Jest ona wykresem wskaźnika TPR (czułości, SE) wobec FPR (dopełnienia do jedynki swoistości, $1 - SP$) przy rosnącym progu wartości decyzyjnej, powyżej którego wektory ze zbioru danych są przypisywane do klasy pozytywnej.

Na podstawie wykresu można wyznaczyć AUC, czyli pole pod krzywą (*Area Under the Curve*), które jest często używaną liczbową miarą jakości klasyfikatora.



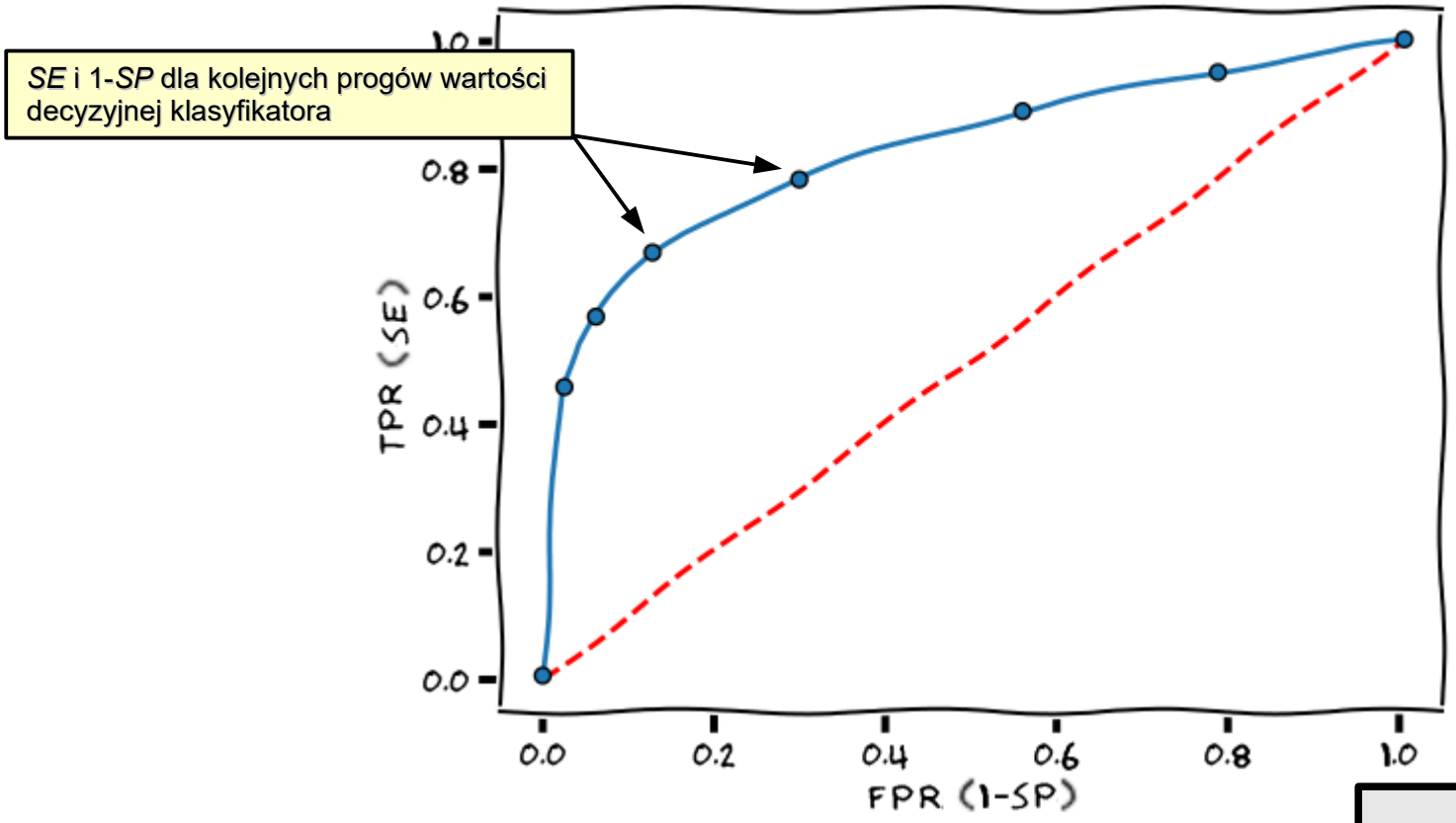
[www.wikipedia.org]

	MATLAB	R
ROC:	`perfcurve`	pakiet `ROCR`

Klasyfikacja: miary jakości klasyfikatora (krzywe ROC)

Krzywa ROC (*Receiver Operating Characteristic*) jest jedną z metod graficznej oceny klasyfikatora binarnego. Jest ona wykresem wskaźnika TPR (czułości, SE) wobec FPR (dopełnienia do jedynki swoistości, $1 - SP$) przy rosnącym progu wartości decyzyjnej, powyżej którego wektory ze zbioru danych są przypisywane do klasy pozytywnej.

Na podstawie wykresu można wyznaczyć AUC, czyli pole pod krzywą (*Area Under the Curve*), które jest często używaną liczbową miarą jakości klasyfikatora.



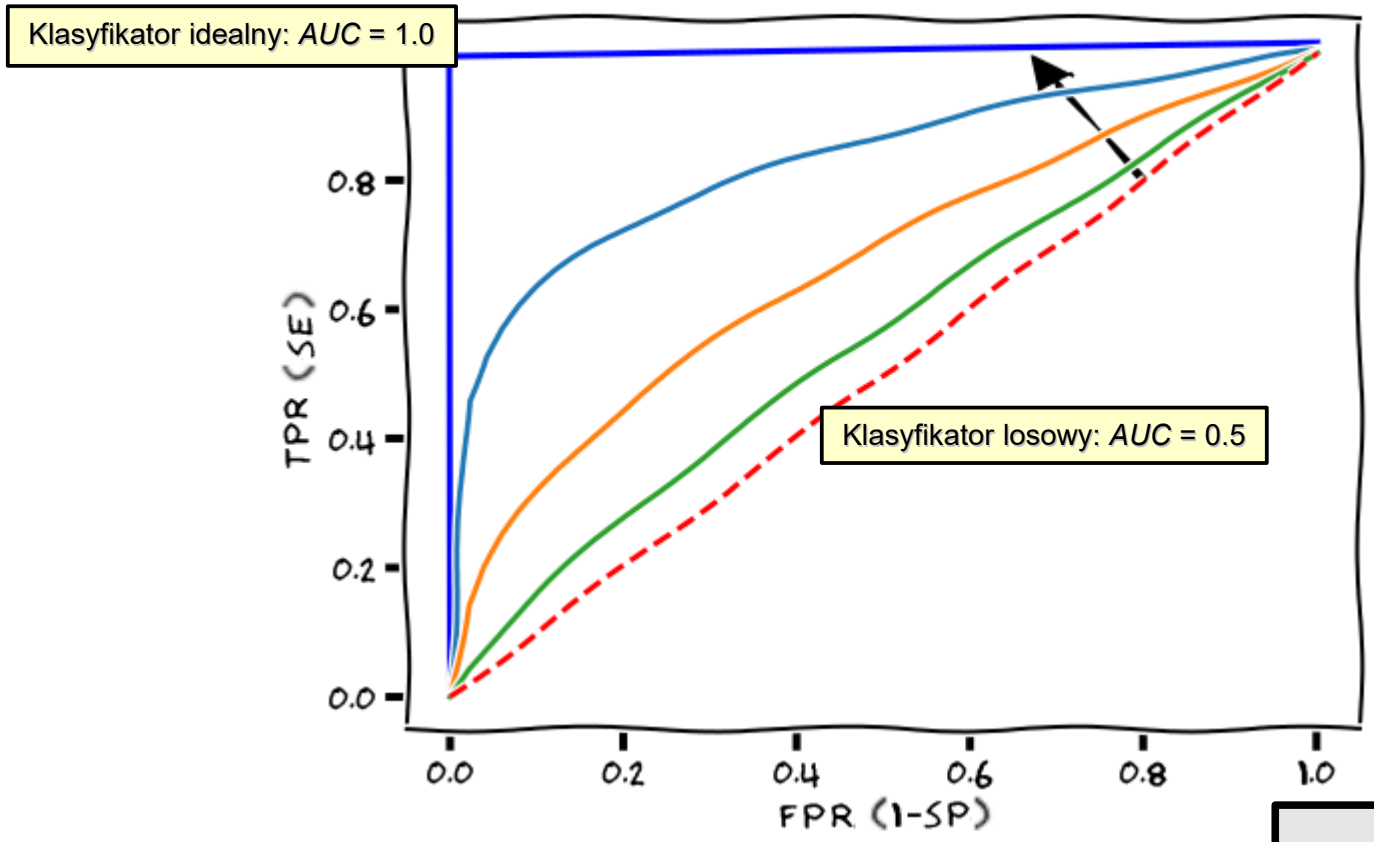
[www.wikipedia.org]

	MATLAB	R
ROC:	`perfcurve`	pakiet `ROCR`

Klasyfikacja: miary jakości klasyfikatora (krzywe ROC)

Krzywa ROC (*Receiver Operating Characteristic*) jest jedną z metod graficznej oceny klasyfikatora binarnego. Jest ona wykresem wskaźnika TPR (czułości, SE) wobec FPR (dopełnienia do jedynki swoistości, $1 - SP$) przy rosnącym progu wartości decyzyjnej, powyżej którego wektory ze zbioru danych są przypisywane do klasy pozytywnej.

Na podstawie wykresu można wyznaczyć AUC, czyli pole pod krzywą (*Area Under the Curve*), które jest często używaną liczbową miarą jakości klasyfikatora.



[www.wikipedia.org]

	MATLAB	R
ROC:	`perfcurve`	pakiet `ROCR`