ANALIZA DANYCH

Wykład 4: Uczenie nadzorowane



dr inż. Bartosz Czech

Katedra Genetyki · Uniwersytet Przyrodniczy we Wrocławiu

bartosz.czech@upwr.edu.pl

Plan i cel

Plan wykładu

1. Uczenie nadzorowane (ang. *supervised learning*): klasyfikacja i regresja
2. Dane do modelu i punkt odniesienia
3. Granica decyzyjna
4. Pięć prostych modeli: **k-NN**, **regresja logistyczna**, **naiwny Bayes**, **LDA**, **QDA**
5. Walidacja krzyżowa (ang. *cross-validation*): strojenie i porównanie modeli
6. Regresja wartości ciągłej

Cel

Nauczyć model i **uczciwie** ocenić go na danych, których nie widział.

Model jako uproszczenie rzeczywistości

„All models are wrong, but some are useful."

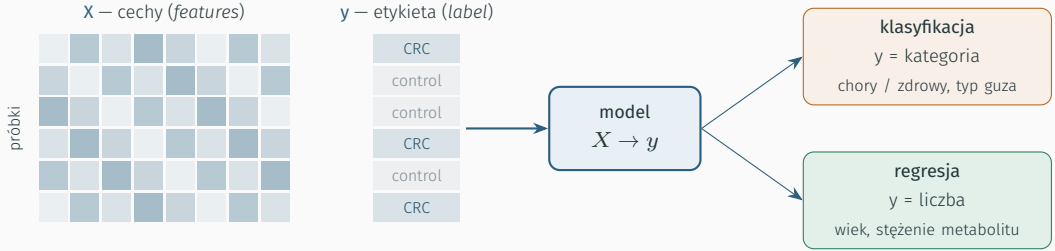
„Wszystkie modele są błędne, ale niektóre są użyteczne." — George E. P. Box, statystyk (Box i Draper, 1987)

Model (ten wykład)	Upraszczające założenie
k-NN	podobne próbki mają podobną etykietę
regresja logistyczna	logarytm szans to suma ważona cech
naiwny Bayes	cechy są niezależne w obrębie klasy
LDA / QDA	cechy w klasie mają rozkład normalny (wspólny / własny rozrzut)

Pytanie nie brzmi „który model jest prawdziwy?", tylko „który jest użyteczny?" — rozstrzyga to ocena na danych, których model nie widział.

Uczenie nadzorowane — przypomnienie (W1)

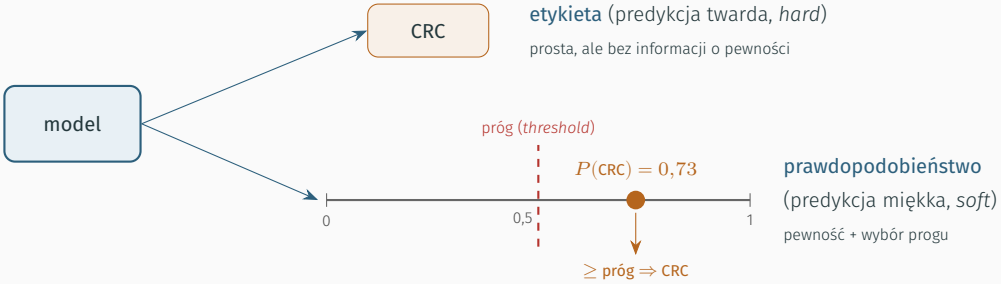
Model uczy się na próbkach, dla których znamy **cechy** X i **etykietę** y.



Cel: trafne predykcje dla **nowych** próbek, spoza treningu.

Dlatego zbiór testowy (ang. *test set*) odkładamy na samym początku.

Dwa wyjścia klasyfikatora: etykieta i prawdopodobieństwo



Z prawdopodobieństwa zawsze uzyskamy etykietę (przez próg) — odwrotnie nie. Próg można dobrać do kosztu błędu, np. obniżyć w badaniu przesiewowym.

Proste modele jako punkt wyjścia

Wykład zaczyna od modeli prostych z trzech powodów:

powód	co to znaczy	przykład z tego wykładu
interpretowalne	widać, które cechy i jak mocno wpływają na decyzję	iloraz szans taksonu w regresji logistycznej
punkt odniesienia	model złożony (np. las losowy) ma sens, jeśli jest od nich wyraźnie lepszy	pięć modeli porównanych w tych samych częściach walidacji
uczą pojęć wspólnych dla wszystkich modeli	granica decyzyjna, próg, przeuczenie, hiperparametr	granica k-NN i regresji, próg 0,5, liczba sąsiadów k

Najpierw prosty model i uczciwa ocena — potem model złożony.

Dane do modelu

Dane do modeli: podział, filtr, CLR i PCA

```
source("~/analiza_danych_bioinf/wyklady/load_cached.R")
d <- load_demo()                                # ZellerG_2014: control i CRC
X <- d$X; y <- d$y                              # X: udziały taksonów (114 próbek), y: control / CRC
set.seed(42)                                    # podział warstwowy 75/25 (jak na W2)
idx_test <- unlist(lapply(split(seq_along(y), y),
  function(i) sample(i, round(0.25 * length(i)))))
tr <- setdiff(seq_along(y), idx_test)
keep <- colMeans(X[tr, ] > 0) >= 0.10           # takson obecny w >= 10% próbek treningu (272)
Xtr <- clr(X[tr, keep]); Xte <- clr(X[idx_test, keep])   # CLR osobno
pca <- prcomp(Xtr)                              # PCA uczone na treningu
Ztr <- pca$x[, 1:10]; Zte <- predict(pca, Xte)[, 1:10]   # test rzutowany
```

Zasada: najpierw **podział**, potem filtr częstości występowania (zostają taksony obecne w co najmniej 10% próbek treningu) i PCA, uczone **tylko na treningu**; test tylko przekształcamy. Po co PCA? 272 taksony to więcej cech niż próbek (86) — model dopasowałby trening idealnie; liczbę 10 składowych przyjęliśmy z góry (można ją też stroić). `clr()` pochodzi z `load_cached.R`.

> Na co dzień: `createDataPartition()` (`caret`), `train_test_split(stratify = y)` (scikit-learn).

Punkt odniesienia: klasyfikator bazowy

Klasyfikator bazowy (*baseline*)

„Model", który nie patrzy na cechy i każdej próbce daje tę samą etykietę — klasę najczęstszą w treningu (*klasę większościową*; u nas 46 kontroli wobec 40 chorych, więc zawsze „control").



zbiór testowy: 28 próbek, każdej przypisano „control"

Dokładność (*accuracy*) = odsetek próbek z poprawną etykietą; dla reguły „zawsze control": $15/28 = 0{,}536$.

```
round(mean(y[idx_test] == "control"), 3)   # odsetek trafień reguły „zawsze control"
```

```
[1] 0.536
```

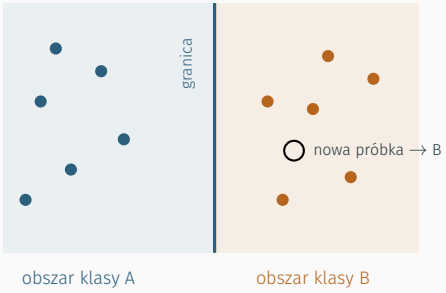
Model z dokładnością ok. 0,54 nie nauczył się niczego; każdy model musi przekroczyć 0,536.

Granica decyzyjna

Granica decyzyjna — przypomnienie (W1)

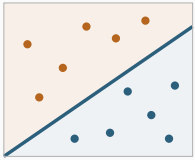
Granica decyzyjna (*decision boundary*)

Linia lub powierzchnia rozdzielająca **obszary** przestrzeni cech (ang. *feature space*) przypisane do klas.



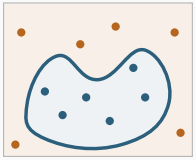
Modele różnią się przede wszystkim **kształtem** granicy.

Model liniowy



liniowa: prosta

regresja logistyczna, LDA



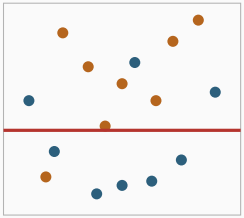
nieliniowa: krzywa

k-NN, naiwny Bayes, QDA

- Model liniowy liczy **sumę ważoną** cech i porównuje ją z progiem — granica to prosta (w wielu wymiarach: płaszczyzna).
- W modelu nieliniowym granica może być krzywą dowolnego kształtu, dopasowaną do układu próbek.

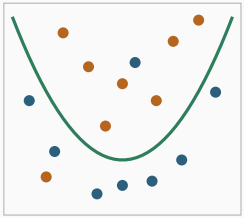
> Liniowy nie znaczy „gorszy”: prosta granica rzadziej dopasowuje się do szumu, więc przy małej liczbie próbek bywa **bezpieczniejsza** — pod warunkiem, że cech nie jest za dużo (u nas 10 składowych PCA zamiast 272 taksonów).

Złożoność granicy a przeuczenie — przypomnienie (W1)



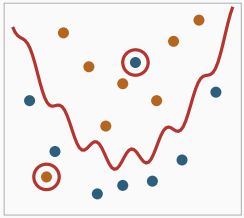
za prosta

niedouczenie (*underfitting*)



odpowiednia

uchwycony wzorzec



zbyt złożona

przeuczenie (*overfitting*)

złożoność

najmniejszy błąd testowy

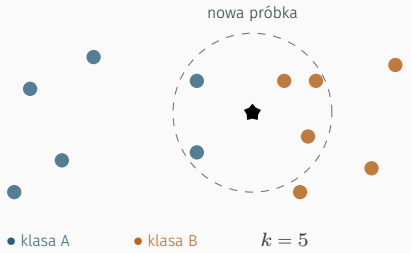
- Za prosta: nie oddaje kształtu danych, myli się nawet na treningu.
- Zbyt złożona: dopasowuje się do każdego punktu treningu, także do szumu — na nowych danych myli się częściej.
- Szukamy złożoności pośredniej; wskaże ją walidacja.

k najbliższych sąsiadów

Metoda k najbliższych sąsiadów

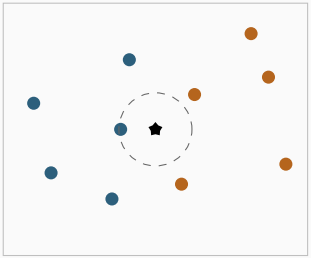
Metoda k najbliższych sąsiadów (*k-nearest neighbours*, k-NN)

Nowa próbka dostaje klasę, która przeważa wśród jej k najbliższych próbek uczących.



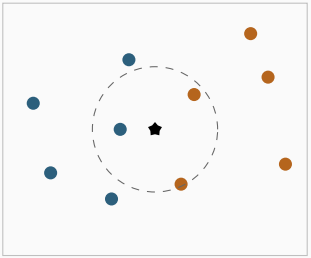
1. Odległość do **każdej** próbki uczącej
2. Wybór k najbliższych (tu 5) — liczymy **sąsiadów**, nie promień
3. Głosowanie większościowe (*majority vote*): **3 × B**, 2 × A $\longrightarrow$ **B**

Wpływ k na klasę nowej próbki



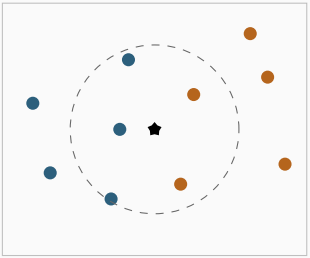
$k = 1$: 1 × A

$\rightarrow$ klasa A



$k = 3$: 2 × B, 1 × A

$\rightarrow$ klasa B



$k = 5$: 3 × A, 2 × B

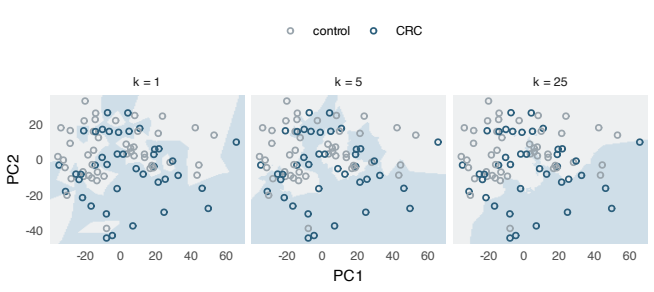
$\rightarrow$ klasa A

Ustalamy **liczbę sąsiadów** k; promień to tylko odległość do k-tego sąsiada.

Ta sama próbka może dostać inną klasę przy innym k.

Wpływ k na granicę decyzyjną

86 próbek treningowych na PC1–PC2; tło = klasa, którą k-NN przypisałby nowej próbce w tym miejscu.

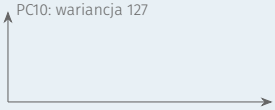


Dokładność na treningu: k = 1: 1,000 (drobne wyspy, przeuczenie) · k = 5: 0,744 · k = 25: 0,581 (dwa duże obszary, niedouczenie)

Jak dobrać k? To **hiperparametr**: sprawdzamy kilka wartości (np. 1, 3, ..., 25) **walidacją krzyżową** i wybieramy najlepszą. Przy dwóch klasach k nieparzyste — bez remisów.

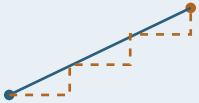
Miara odległości w k-NN

skala osi



oś o dużym rozrzucie
waży więcej

miara odległości



euklidesowa: w linii prostej
Manhattan: suma różnic po osiach

liczba wymiarów

272 → 10

setki cech: wszystkie próbki
„równie daleko”
⇒ składowe PCA

k-NN — zalety i wady

Zalety

- Dowolny kształt granicy
- Łatwy do wyjaśnienia
- Jeden hiperparametr (k)

Wady

- Wrażliwy na skalę i liczbę cech
- Wolny przy dużych zbiorach
- Nie wskazuje ważnych cech

Rola: **punkt odniesienia** — czy w danych jest jakikolwiek sygnał?

k-NN w R — trening i predykcja

```r
library(class)              # Ztr, Zte: składowe z treningu / rzut testu
set.seed(1)                 # remisy w głosowaniu knn rozstrzyga losowo
pred <- knn(train = Ztr, test = Zte, cl = y[tr], k = 5)
length(tr); length(idx_test)              # rozmiary: trening, test
rbind(trening = table(y[tr]), test = table(y[idx_test]))
```

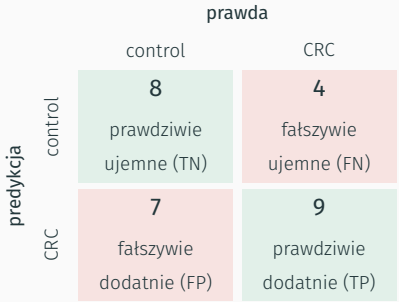
```
[1] 86
[1] 28
         control CRC
trening       46  40
test          15  13
```

Podział **warstwowy** (*stratified*): te same proporcje klas.

Macierz pomyłek na teście

```r
cm <- table(predykcja = pred,
            prawda = y[idx_test])
cm
sum(diag(cm)) / sum(cm)   # dokładność
```

```
         prawda
predykcja control CRC
  control       8   4
  CRC           7   9
[1] 0.6071429
```



Przekątna (zielona) = trafienia.

FN: przeoczeni chorzy · FP: fałszywe alarmy

$$\text{dokładność} = \frac{8+9}{28} = 0{,}61$$

Dokładność, czułość i swoistość

Dokładność

	control	CRC
	TN = 8	FN = 4
	FP = 7	TP = 9

$\frac{\text{TP+TN}}{\text{wszystkie}} = \frac{17}{28} = 0{,}61$

Czułość

	control	CRC
	TN = 8	FN = 4
	FP = 7	TP = 9

$\frac{\text{TP}}{\text{TP+FN}} = \frac{9}{13} = 0{,}69$

Swoistość

	control	CRC
	TN = 8	FN = 4
	FP = 7	TP = 9

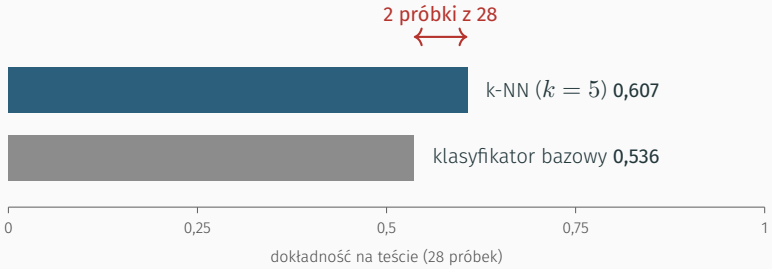
$\frac{\text{TN}}{\text{TN+FP}} = \frac{8}{15} = 0{,}53$

k-NN ($k = 5$), test; kolumny: prawda, wiersze: predykcja; zielone pola — licznik, obramowane — mianownik.

	co mierzy	kiedy najważniejsza	pułapka
dokładność	jaką część **wszystkich** próbek model sklasyfikował dobrze (u nas 17 z 28)	gdy obu klas jest podobnie dużo	gdy chorych jest 5%, model mówiący każdemu „zdrowy” ma dokładność 0,95, choć nie wykrywa nikogo
czułość	jaką część **chorych** model rozpoznał jako chorych (9 z 13)	badania przesiewowe: nie przeoczyć chorego	model mówiący każdemu „chory” ma czułość 1, ale każdego zdrowego też uznaje za chorego
swoistość	jaką część **zdrowych** model rozpoznał jako zdrowych (8 z 15)	przed inwazyjnym leczeniem: nie leczyć zdrowego	model mówiący każdemu „zdrowy” ma swoistość 1, ale nie wykrywa żadnego chorego

Dlatego czułość i swoistość podaje się zawsze razem, a dokładność porównuje z klasyfikatorem bazowym.

Wynik k-NN na tle klasyfikatora bazowego



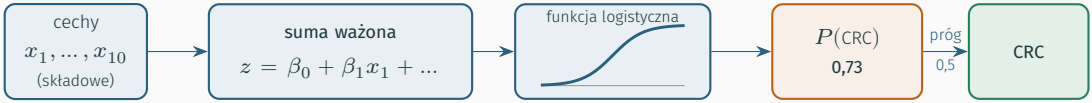
- Lepiej niż punkt odniesienia — **ale niewiele**
- Jeden mały test = wynik niepewny ⟶ walidacja krzyżowa

Regresja logistyczna

Regresja logistyczna jako klasyfikator

Regresja logistyczna (*logistic regression*)

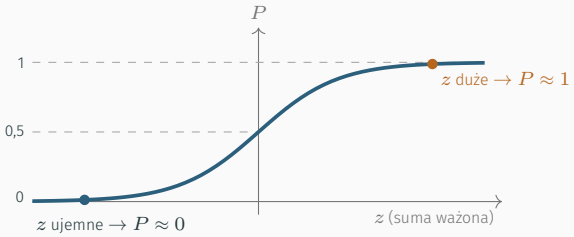
Model zwracający prawdopodobieństwo klasy jako funkcję ważonej sumy cech. Mimo nazwy — klasyfikator.



1. **Suma ważona**: każdą cechę mnoży się przez jej współczynnik (wagę) β_j uczony z danych; dodatnia waga przemawia za CRC, ujemna za kontrolą; β_0 to wyraz wolny.
2. **Funkcja logistyczna** zamienia dowolną liczbę z na prawdopodobieństwo między 0 a 1.
3. **Próg** zamienia prawdopodobieństwo na etykietę.

Przykład: $z = 1{,}0 \Rightarrow P = \frac{1}{1+e^{-1}} = 0{,}73 \geq 0{,}5 \Rightarrow$ CRC. Granica decyzyjna: $z = 0$, czyli $P = 0{,}5$ – **prosta** (płaszczyzna) w przestrzeni cech.

Funkcja logistyczna



$$P = \frac{1}{1 + e^{-z}}$$

- Dowolne $z \rightarrow P$ między 0 a 1
- $z = 0 \rightarrow P = 0{,}5$ (granica)
- Wagi β: metoda **największej wiarygodności** (*maximum likelihood*)

Szansa i logarytm szans

Szansa (*odds*)

Stosunek prawdopodobieństwa zdarzenia do prawdopodobieństwa jego braku:

$$\text{szansa} = \frac{P}{1-P}$$

P	0,2	0,5	0,8
szansa	0,25	1	4
czytamy	1 do 4	1 do 1	4 do 1

Logarytm szans (*logit*)

Logarytm naturalny szansy. Przyjmuje dowolną wartość (od $-\infty$ do $+\infty$), więc można go zapisać sumą ważoną cech:

$$\ln \frac{P}{1-P} = \beta_0 + \beta_1 x_1 + \cdots = z$$

- z to ta sama suma ważona co na poprzednich slajdach.
- $P = 0{,}5 \to$ szansa 1 $\to$ logit 0 (granica).

Iloraz szans

Iloraz szans (*odds ratio*, OR)

Stosunek dwóch szans: szansy przy cesze x_j większej o 1 do szansy przy wyjściowej wartości, przy niezmienionych pozostałych cechach.

$$\text{OR}_j = \frac{\text{szansa}(x_j + 1)}{\text{szansa}(x_j)} = e^{\beta_j}$$

- Wzrost x_j o 1 dodaje β_j do logitu, czyli mnoży szansę przez e^{β_j}.
- Przykład: $\beta_j = 0{,}215 \Rightarrow \text{OR} = e^{0{,}215} = 1{,}24 -$ szansa rośnie 1,24 raza.
- $\text{OR} = 1$: brak związku; > 1: większa szansa; < 1: mniejsza.

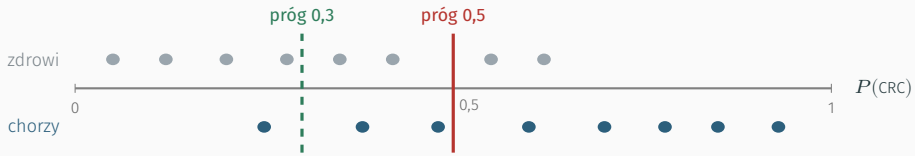
Iloraz szans mnoży szansę, a nie prawdopodobieństwo.

Przykład dla OR = 1,24:

	szansa	P
przed	1	0,50
po ($\times\, 1{,}24$)	1,24	0,55

P rośnie o 5 punktów procentowych, a nie o 24%.

Próg decyzyjny



	wykryci chorzy	fałszywe alarmy
próg 0,5	5 z 8	2 z 8
próg 0,3	7 z 8	4 z 8

- Niższy próg: czułość rośnie (7 z 8 chorych zamiast 5 z 8), swoistość maleje (4 z 8 zdrowych rozpoznanych zamiast 6 z 8).
- Ten sam model, bez ponownego uczenia — zmienia się tylko próg (schemat).

Regresja logistyczna w R — trening

```
dtr <- data.frame(y = y[tr], Ztr)              # Ztr: 10 składowych z treningu
m <- glm(y ~ ., data = dtr, family = binomial)
p <- predict(m, newdata = data.frame(Zte), type = "response")   # P(CRC)
pred <- factor(ifelse(p >= 0.5, levels(y)[2], levels(y)[1]),   # P >= 0,5 -> CRC
               levels = levels(y))
cm <- table(predykcja = pred, prawda = y[idx_test])
cm
sum(diag(cm)) / sum(cm)
```

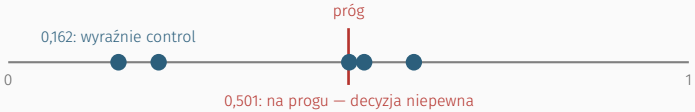
```
         prawda
predykcja control CRC
  control      11   6
  CRC           4   7
[1] 0.6428571
```

Dokładność **0,643** — o jedną próbkę więcej niż k-NN (0,607).

Regresja logistyczna w R — prawdopodobieństwa

```
round(head(p, 5), 3)       # P(CRC) pierwszych 5 próbek testu
levels(y)[2]               # klasa modelowana jako "sukces"
```

```
CCIS80834637ST-4-0 CCIS50003399ST-4-0 CCIS00146684ST-4-0 CCIS27927933ST-4-0
             0.523              0.162              0.221              0.596
CCIS07648107ST-4-0
             0.501
[1] "CRC"
```



Ilorazy szans w modelu na trzech taksonach

Model na **trzech** taksonach o największej różnicy średnich między klasami (tylko trening); `exp(współczynnik)` to iloraz szans. `Xtr`, `y`, `tr`: jak na slajdzie „Dane do modeli”.

```
effect <- apply(Xtr, 2, function(x)        # różnica średnich klas / odch. stand. taksonu
  abs(mean(x[y[tr] == "CRC"]) - mean(x[y[tr] == "control"])) / sd(x))
top3 <- names(sort(effect, decreasing = TRUE))[1:3]   # 3 taksony o największej różnicy
dtr3 <- data.frame(y = y[tr], Xtr[, top3])            # dane do modelu: etykieta + 3 taksony
names(dtr3) <- c("y", "t1", "t2", "t3")
m3 <- glm(y ~ t1 + t2 + t3, data = dtr3, family = binomial)
round(exp(coef(m3)), 2)                               # iloraz szans
```

```
(Intercept)          t1          t2          t3
       9.22        1.24        1.32        1.25
```

1: brak związku

< 1: mniejsza szansa na CRC | > 1: większa szansa na CRC

t1 *P. stomatis* 1,24 · t2 *F. nucleatum* 1,32 · t3 *D. pneumosintes* 1,25 · `(Intercept)`: wyraz wolny, bez interpretacji biologicznej

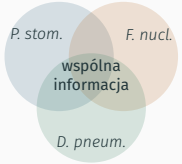
Odczyt ilorazu szans i współliniowość

Odczyt

P. stomatis, OR = 1,24: gdy CLR tego taksonu wzrośnie o 1, a dwa pozostałe taksony się nie zmienią, szansa na CRC wzrośnie 1,24 raza.

Co znaczy CLR o 1 wyżej: $\text{CLR} = \ln \frac{\text{udział taksonu}}{\text{średnia geometryczna udziałów próbki}}$, więc $+1$ mnoży ten stosunek przez $e \approx 2{,}72$. Np. stosunek 2 ($\text{CLR} = \ln 2 = 0{,}69$) $\rightarrow$ CLR $1{,}69$ $\rightarrow$ stosunek $e^{1{,}69} = 5{,}4 = 2 \times 2{,}72$.

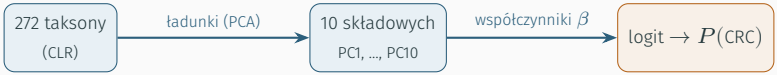
Takson	osobno	razem
P. stomatis	1,36	1,24
F. nucleatum	1,85	1,32
D. pneumosintes	1,66	1,25



współliniowość (*collinearity*) korelacje 0,61–0,73

- „Osobno”: model z jednym taksonem; „razem”: model z trzema — ilorazy **maleją** (spadek $\neq$ takson mniej ważny).
- **Współliniowość**: silnie skorelowane cechy niosą tę samą informację (u nas wspólne zera u kontroli), więc model dzieli między nie wpływ.

Współczynniki modelu na składowych PCA



model widzi tylko składowe, nie taksony

```
round(exp(coef(m)[c("PC2", "PC3")]), 3)
```

```
  PC2   PC3
0.953 1.075
```

- Iloraz szans dotyczy **składowej**: PC3 o 1 wyżej $\rightarrow$ szansa na CRC $\times\, 1{,}075$.
- 1 jednostka to mało (odch. stand. PC3: 15,9). O jedno odchylenie: PC3 $\times\, 3{,}17$, PC2 $\times\, 0{,}45$.
- Składowa to **mieszanka** 272 taksonów; na PC3 największe ładunki mają m.in. *Intestinimonas* i *Butyricimonas*.
- Znak składowej jest umowny: PCA może odwrócić oś, wtedy β zmienia znak.

Od składowych z powrotem do taksonów

Współczynnik zastępczy taksonu

Ładunek taksonu na składowej razy jej współczynnik, zsumowane po 10 składowych: $b_j = \sum_k \text{ładunek}_{jk}\, \beta_k$.

```
b_tax <- drop(pca$rotation[, 1:10] %*% coef(m)[-1])   # ładunki × współczynniki
names(b_tax) <- sub("species:", "", names(b_tax))
round(head(b_tax[order(-abs(b_tax))], 6), 3)          # 6 największych |b|
```

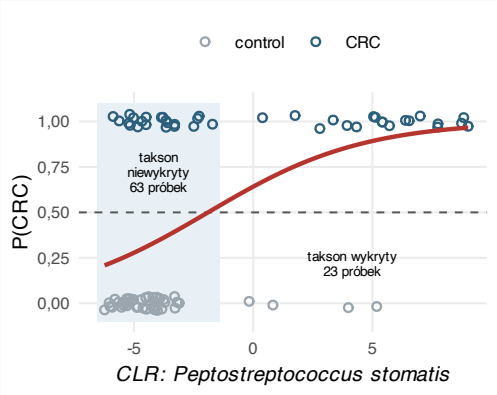
```
Peptostreptococcus stomatis      Dialister pneumosintes
                      0.027                       0.022
          Parvimonas micra          Bacteroides nordii
                      0.022                       0.021
          Escherichia coli     Fusobacterium nucleatum
                      0.019                       0.019
```

Na czele bakterie jamy ustnej; trzy taksony z modelu z ilorazami szans są w pierwszej szóstce.

Ograniczenia. To nie efekt pojedynczego taksonu: model zna tylko 10 kierunków (38% wariancji). Składowe wybrano bez etykiet (istotne są PC2 i PC3, nie PC1). Brak p-wartości dla taksonów — tylko ranking.

Wniosek. PCA służy predykcji przy $p \gg n$; do interpretacji lepszy jest model na wybranych taksonach, jak ten z ilorazami szans.

Regresja logistyczna z jednym taksonem



Regresja logistyczna z **jednym** taksonem, *P. stomatis*.

- Kółko = próbka treningowa (u góry chorzy, u dołu kontrole); w poziomie — CLR taksonu.
- **Niebieski pas**: 63 z 86 próbek, w których taksonu **nie wykryto** (42 z 46 kontroli, 21 z 40 chorych). Ich CLR jest niski (od $-6{,}2$ do $-1{,}7$).
- Na prawo od pasa: 23 próbki z wykrytym taksonem, w tym 19 chorych.
- Czerwona krzywa: $P(\text{CRC})$ od **0,21** do **0,97**; próg 0,5 przy CLR ok. $-1{,}9$, na skraju pasa.

Model działa prawie jak reguła „takson wykryty $\rightarrow$ CRC”; dla połowy chorych cecha nic nie wnosi. Kształt krzywej poza pasem to **założenie** modelu, słabo sprawdzone przez dane.

Interpretacja biologiczna i jej granice

Model podpowiada, że

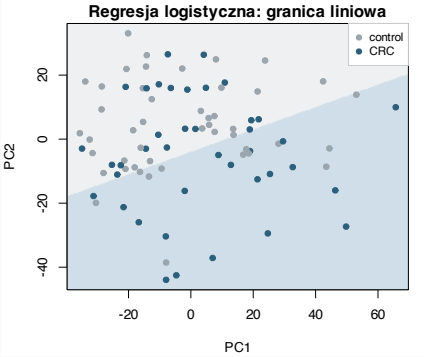
- *F. nucleatum* i *P. stomatis* mają wyższy udział u chorych
- to zgadza się z literaturą: bakterie jamy ustnej (Zeller i in., 2014)
- warto to sprawdzić w kolejnych badaniach (hipoteza)

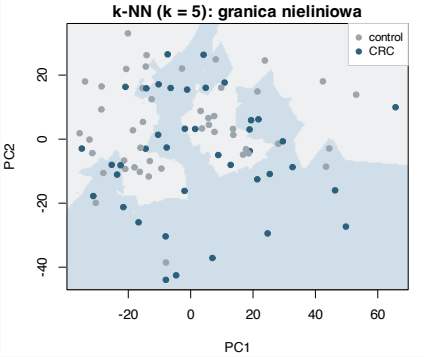
Model nie dowodzi, że

- te bakterie wywołują raka
- jeden takson wystarczy do rozpoznania choroby
- to bakteria wpływa na guz, a nie guz na skład mikrobiomu

Dieta i leki to możliwe **czynniki zakłócające** (ang. *confounders*).

Granica liniowa i nieliniowa na danych przykładowych





86 próbek treningowych na PC1–PC2; tło = przypisana klasa.

Regresja logistyczna: **jedna prosta**. k-NN: granica **nieregularna**, lokalna. Klasy silnie się nakładają.

Regresja logistyczna a k-NN

	regresja logistyczna	k-NN
granica	liniowa	✓ elastyczna
które cechy decydują	✓ ilorazy szans	×
prawdopodobieństwo	✓ ciągłe	× tylko $k+1$ wartości (odsetek głosów)
wiele cech ($p \gg n$, W1)	PCA lub regularyzacja (W1)	PCA

Nie ma modelu „najlepszego”. Potrzebne wyjaśnienie → regresja; wyraźnie nieliniowa granica → k-NN.

Naiwny klasyfikator Bayesa

Prawdopodobieństwo warunkowe

$P(A \mid B)$ — prawdopodobieństwo A, gdy wiemy, że zaszło B: patrzymy **tylko na przypadki z B**. Przykład: czy w próbce **wykryto takson** *P. stomatis* (udział > 0), 86 próbek treningowych.

	control	CRC	razem
takson wykryty	4	**19**	23
niewykryty	42	21	63
razem	46	40	86

$P(\text{wykryty} \mid \text{CRC})$: tylko kolumna CRC, $19/40 = \mathbf{0{,}48}$

	control	CRC	razem
takson wykryty	4	**19**	23
niewykryty	42	21	63
razem	46	40	86

$P(\text{CRC} \mid \text{wykryty})$: tylko wiersz „takson wykryty”, $19/23 = \mathbf{0{,}83}$

Wzór

$$P(A \mid B) = \frac{P(A \text{ i } B)}{P(B)}, \quad \text{np. } P(\text{CRC} \mid \text{wykryty}) = \frac{P(\text{CRC i wykryty})}{P(\text{wykryty})} = \frac{19/86}{23/86} = \frac{19}{23} = 0{,}83$$

Kierunek ma znaczenie: $P(A \mid B) \neq P(B \mid A)$ — te same liczby, inne pytanie, inny wynik.

Twierdzenie Bayesa

$$\underbrace{P(\text{klasa} \mid x)}_{\text{a posteriori}} = \frac{\overbrace{P(x \mid \text{klasa})}^{\text{wiarygodność}} \cdot \overbrace{P(\text{klasa})}^{\text{a priori}}}{\underbrace{P(x)}_{\text{prawdopodobieństwo całkowite}}}$$

$$P(x) = \underbrace{P(x \mid \text{klasa 1}) \cdot P(\text{klasa 1})}_{\text{klasa 1}} + \underbrace{P(x \mid \text{klasa 2}) \cdot P(\text{klasa 2})}_{\text{klasa 2}} + \cdots$$

Przykład, x = takson *P. stomatis* wykryty:

$$P(\text{CRC} \mid \text{wykryty}) = \frac{(19/40) \cdot (40/86)}{(19/40) \cdot (40/86) \;+\; (4/46) \cdot (46/86)} = \frac{19/86}{19/86 + 4/86} = \frac{19}{23} = 0{,}83$$

- A priori $P(\text{CRC}) = 40/86 = 0{,}47$: zanim spojrzymy na cechę. A posteriori $P(\text{CRC} \mid \text{wykryty}) = 0{,}83$: po zobaczeniu cechy.
- Wiarygodność $P(\text{wykryty} \mid \text{CRC}) = 19/40$: jak często cecha występuje w klasie.

Zasada działania naiwnego Bayesa

Naiwny klasyfikator Bayesa (*naive Bayes*)

Dla każdej klasy liczy **wynik**: jak częsta jest klasa i jak często w tej klasie występują cechy nowej próbki.
Próbka dostaje etykietę klasy z **większym** wynikiem.

$$\underbrace{P(\text{klasa} \mid x)}_{\text{a posteriori}} \propto \underbrace{P(\text{klasa})}_{\text{a priori}} \times \underbrace{P(x_1 \mid \text{klasa})}_{\text{cecha 1 w klasie}} \times \underbrace{P(x_2 \mid \text{klasa})}_{\text{cecha 2 w klasie}} \times \cdots \times P(x_{10} \mid \text{klasa})$$



- Wszystkie częstości liczymy **raz, na treningu**; każdą próbkę testową tylko podstawiamy.
- Twierdzenie Bayesa dla wielu cech, bez mianownika ($\propto$); na końcu dzielimy przez sumę wyników.
- **Naiwne** założenie: cechy w klasie są niezależne, więc częstości wolno po prostu **mnożyć**.

Naiwny Bayes — przykład

Dwie cechy tak/nie: czy wykryto takson *P. stomatis* i czy wykryto *P. micra*. Częstości z **treningu** (86 próbek):

	control (46)	CRC (40)
częstość klasy (a priori)	46/86 = 0,535	40/86 = 0,465
P. stomatis wykryty	4/46 = 0,087	19/40 = 0,475
P. stomatis niewykryty	42/46 = 0,913	21/40 = 0,525
P. micra wykryty	20/46 = 0,435	22/40 = 0,550
P. micra niewykryty	26/46 = 0,565	18/40 = 0,450

Próbka A: oba taksony wykryte

control: $46/86 \times 4/46 \times 20/46 = 0{,}020$

CRC: $40/86 \times 19/40 \times 22/40 = \mathbf{0{,}122}$

$P(\text{CRC}) = 0{,}122/(0{,}122 + 0{,}020) = 0{,}86$

większy wynik: CRC $\Rightarrow$ etykieta CRC

Próbka B: żaden nie wykryty

control: $46/86 \times 42/46 \times 26/46 = \mathbf{0{,}276}$

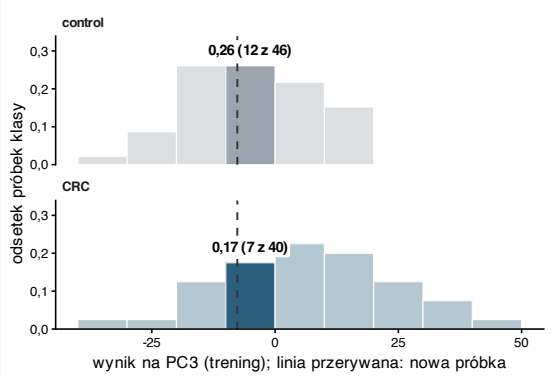
CRC: $40/86 \times 21/40 \times 18/40 = 0{,}110$

$P(\text{CRC}) = 0{,}110/(0{,}110 + 0{,}276) = 0{,}28$

większy wynik: control $\Rightarrow$ etykieta control

W każdej klasie „wykryty” + „niewykryty” = 1. Iloczyny liczone z ułamków, wynik zaokrąglony do 3 miejsc.

Naiwny Bayes — cechy ciągłe



- Składowe PCA nie są tak/nie. Liczymy więc, jaki **odsetek próbek klasy** ma podobną wartość: słupek histogramu w miejscu nowej próbki.
- Pierwsza próbka testowa: PC3 $= -7{,}65$. W tym przedziale kontrole **0,26**, chorzy **0,17** $\Rightarrow$ PC3 przemawia za kontrolą.
- Model robi tak dla każdej z 10 składowych i mnoży odczyty, jak w przykładzie z taksonami.
- `naiveBayes()` zamiast słupków używa gładkiej krzywej (średnia i odchylenie z treningu); idea ta sama.

Naiwny Bayes w R

```r
library(e1071)
m <- naiveBayes(x = Ztr, y = y[tr])        # trening: częstości klas + opis każdej składowej w klasie
round(head(predict(m, Zte, type = "raw"), 4), 3)   # test: P(klasa) dla 4 pierwszych próbek
pred <- predict(m, Zte)                    # etykieta = klasa z większym wynikiem
cm <- table(predykcja = pred, prawda = y[idx_test])
cm
sum(diag(cm)) / sum(cm)
```

```
     control   CRC
[1,]   0.676 0.324
[2,]   0.499 0.501
[3,]   0.805 0.195
[4,]   0.643 0.357
          prawda
predykcja control CRC
  control      12   6
  CRC           3   7
[1] 0.6785714
```

Założenie niezależności na danych przykładowych

	10 składowych	272 taksony (CLR)
korelacje cech w klasie	słabe	silniejsze
skrajne $P(\text{CRC})$ ($< 0{,}01$ lub $> 0{,}99$)	0 z 28	26 z 28
dokładność na teście	0,679	0,714

Skorelowane taksony liczone wiele razy $\longrightarrow$ model „pewny” prawie zawsze, także gdy się myli.

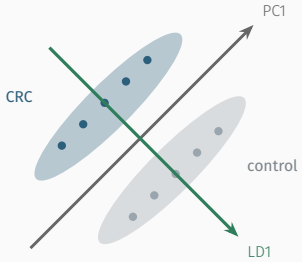
Etykiety bywają trafne — prawdopodobieństwa nie nadają się do oceny ryzyka (słaba *kalibracja*: $P = 0{,}99$ nie znaczy, że 99% takich osób choruje).

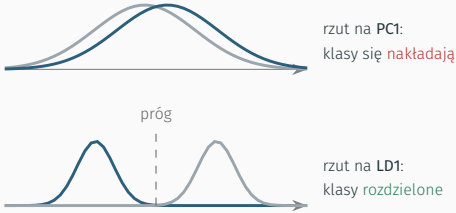
Analiza dyskryminacyjna

Liniowa analiza dyskryminacyjna (LDA)

Liniowa analiza dyskryminacyjna (*linear discriminant analysis*, LDA)

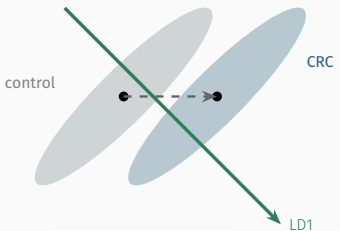
Szuka osi — sumy ważonej cech — na której średnie klas leżą daleko od siebie, a próbki jednej klasy blisko siebie. To nadzorowany odpowiednik PCA (W3), które etykiet nie zna.





PC1 biegnie wzdłuż największej zmienności, która tu nie ma związku z klasą. LD1 korzysta z etykiet i wybiera kierunek, w którym klasy się rozchodzą (schemat).

Klasyfikacja nowej próbki w LDA



Chmury są wydłużone: wzdłuż strzałki średnich nakładają się, wzdłuż LD1 — nie.

Trening (liczony raz, bez iteracji)

1. Średnie cech w każdej klasie $\longrightarrow$ strzałka między średnimi.
2. Rozrzut wewnątrz klas (kształt chmury, wspólny dla obu klas).
3. LD1 = strzałka średnich **poprawiona o rozrzut**: kierunki, w których klasy są szerokie, dostają mniejszą wagę. Tak powstaje oś z największym J (ramka).

Predykcja (każda próbka testowa)

4. Wynik na LD1 = suma ważona cech próbki.
5. Klasa według strony progu (w połowie między średnimi klas).

Kryterium Fishera — co oś LD1 czyni największym

$$J = \frac{(\bar{z}_{\text{CRC}} - \bar{z}_{\text{control}})^2}{s^2_{\text{CRC}} + s^2_{\text{control}}} \qquad \frac{\text{odległość średnich klas na osi}}{\text{rozrzut wewnątrz klas na osi}}$$

z — wynik próbki na osi; im większe J, tym lepiej oś rozdziela klasy.

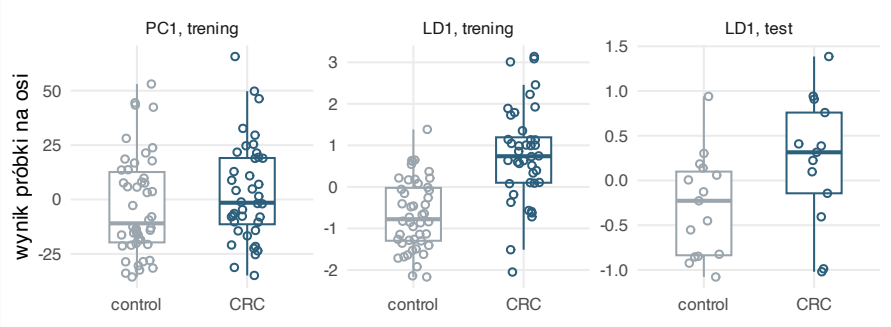
LDA w R na danych przykładowych

```r
library(MASS)
m <- lda(y ~ ., data = dtr)              # dtr: y i 10 składowych (trening)
round(m$scaling[, 1], 3)                 # współczynniki osi LD1
pl <- predict(m, data.frame(Zte))        # $class, $posterior, $x
table(predykcja = pl$class, prawda = y[idx_test])
```

```
   PC1    PC2    PC3    PC4    PC5    PC6    PC7    PC8    PC9   PC10
 0.012 -0.034  0.046  0.017 -0.029  0.025 -0.011 -0.034  0.022 -0.008
          prawda
predykcja control CRC
  control      12   5
  CRC           3   8
```

- **Pierwszy wydruk**: wagi osi LD1, czyli przepis na wynik próbki: $0{,}012 \cdot \text{PC1} - 0{,}034 \cdot \text{PC2} + 0{,}046 \cdot \text{PC3} + \cdots$
- **Drugi wydruk**: macierz pomyłek na teście: 12 + 8 trafnych, dokładność **0,714** (20 z 28).

Oś dyskryminacyjna na danych przykładowych



- Wyniki próbek na osi; pudełka: kontrole i chorzy. PC1 (bez etykiet) prawie nie rozdziela klas: różnica średnich **0,27** odchylenia wewnątrz klas.
- LD1 na treningu: **1,44**; na teście tylko **0,78**. Oś dopasowano do treningu, więc tam rozdziela zbyt dobrze — liczy się test.

LDA a regresja logistyczna

	LDA	regresja logistyczna
założenie o rozkładzie cech	× normalny, wspólny rozrzut	✓ brak
skąd prawdopodobieństwo	reguła Bayesa	wprost
granica	liniowa	liniowa
dodatkowo	✓ oś do rysowania	✓ ilorazy szans

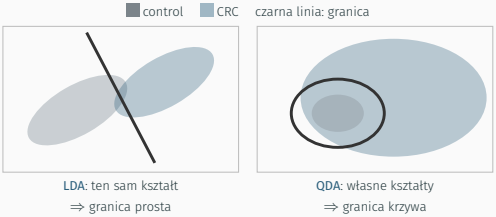
Etykiety na teście (28 próbek):

	regresja logistyczna	
LDA	control	CRC
control	**16**	1
CRC	1	**10**

To nie są trafienia, tylko zgodność dwóch modeli: ta sama etykieta u 16 + 10 = 26 z 28 próbek. Różnią się 2 próbkami; w obu rację ma LDA — stąd 0,714 wobec 0,643.

Kwadratowa analiza dyskryminacyjna (*quadratic discriminant analysis*, QDA)

Model jak LDA (rozkład normalny w klasach + reguła Bayesa), ale każda klasa ma **własny kształt chmury** (własny rozrzut). Dzięki temu granica może być **krzywa**.



LDA: ten sam kształt
⇒ granica prosta

QDA: własne kształty
⇒ granica krzywa

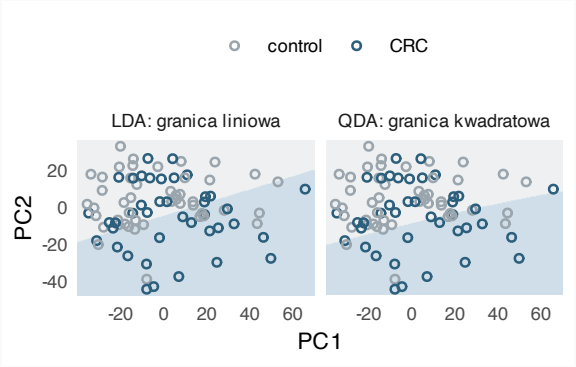
Trening: średnia i rozrzut **osobno w każdej klasie**.

Predykcja: jak typowa jest próbka dla chmury każdej klasy (odległość od środka w rozrzucie tej klasy) × a priori; wygrywa większy wynik.

To nie regresja. „Kwadratowa", bo granica jest krzywą (np. elipsą). Na jednej cesze to **dwa progi**: CRC, gdy PC3 $> 6{,}9$ albo PC3 $< -42{,}2$. LDA ma jeden próg: PC3 $> 3{,}2$.

Naiwny Bayes, LDA i QDA działają tak samo: a priori × „jak typowa jest próbka dla klasy". Różnią się opisem chmury: **NB** — każda cecha osobno; **LDA** — jeden kształt dla obu klas; **QDA** — własny kształt każdej klasy (potrzeba więcej próbek).

QDA na danych przykładowych



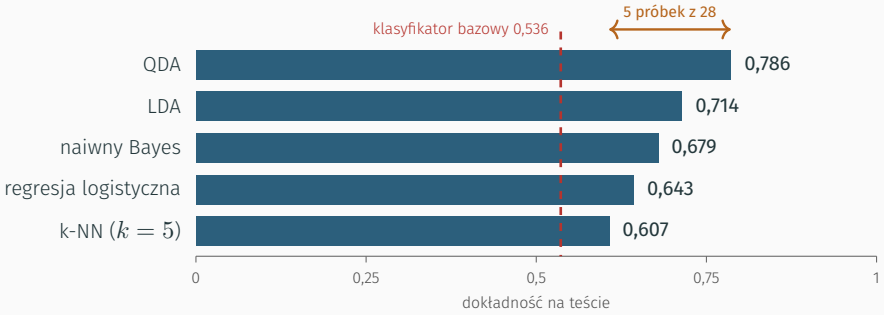
- Na PC1–PC2 podobne rozrzuty ⟶ granica QDA **prawie prosta**
- 10 składowych, dokładność na teście: **0,786**
- Na treningu: QDA 0,884, LDA 0,744 ⟶ ryzyko przeuczenia

Przewaga QDA na teście to tylko **2 próbki** — za mało, by uznać QDA za lepszy model.

Walidacja krzyżowa

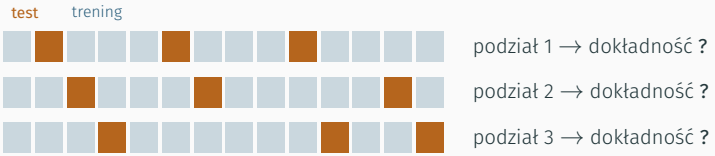
Pięć modeli na zbiorze testowym

Ten sam podział i te same 10 składowych; słupki: **dokładność na teście** (28 próbek).



Który model wybrać? **Na tej podstawie — żaden.**

Zmienność wyniku jednego podziału

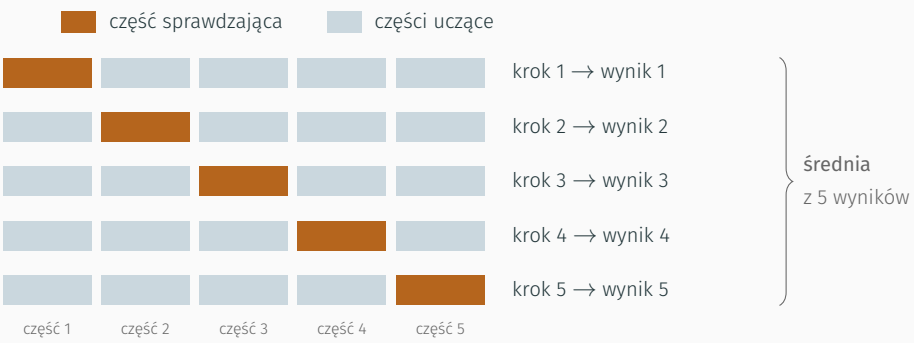


Ta sama regresja logistyczna, **300** losowych podziałów: dokładność od 0,50 do 0,93.

Walidacja krzyżowa (*cross-validation*, CV)

Dane dzielimy na kilka części (*folds*); każda po kolei służy do sprawdzenia, pozostałe do uczenia; wyniki uśredniamy.

k-krotna walidacja krzyżowa — przebieg



Każda próbka **raz** trafia do części sprawdzającej; w tym kroku model się na niej nie uczy.

Uwaga: k w „k-krotnej” to **liczba części**, a nie k z k-NN.

Walidacja krzyżowa — warianty

k-krotna (*k-fold*)



$k = 5$ lub 10
standard

warstwowa (*stratified*)



w każdej części
te same proporcje klas

powtarzana (*repeated*)

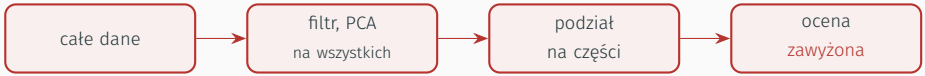
powt. 1	1,6	2,7	3,8	4,9	5,10
powt. 2	3,9	1,10	5,6	2,8	4,7
powt. 3	2,5	4,8	1,9	6,10	3,7

cała walidacja od nowa,
inny przydział próbek

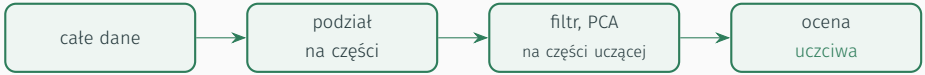
U nas: **5 części, warstwowo, 10 powtórzeń**. Całą walidację 5-krotną wykonuje się 10 razy, za każdym razem z innym losowym przydziałem próbek do części (liczby w kratkach: numery próbek) — razem $10 \times 5 = 50$ ocen; wynik to ich średnia.

Przeciek w walidacji krzyżowej

× źle



✓ dobrze



Filtr i PCA **uczą się** z danych. Policzone na całości „widzą" część sprawdzającą, więc wynik walidacji jest zawyżony.

> Przygotowanie danych należy do **modelu**: w każdej części od nowa.

Walidacja krzyżowa w R i Pythonie

```r
library(caret); set.seed(7)             # Xtr: taksony po filtrze i CLR (trening)
ctrl <- trainControl(method = "repeatedcv", number = 5,   # 5 części,
                     repeats = 10,                        # 10 powtórzeń,
                     preProcOptions = list(pcaComp = 10)) # 10 składowych
fit <- train(x = Xtr, y = y[tr], method = "knn",
             preProcess = "pca",                          # PCA w każdej części
             tuneGrid = data.frame(k = seq(1, 25, by = 2)), trControl = ctrl)
fit$bestTune                            # wybrane k
```

```python
# Python (scikit-learn): ten sam pomysł
model = make_pipeline(PCA(n_components=10), KNeighborsClassifier())
cv = RepeatedStratifiedKFold(n_splits=5, n_repeats=10, random_state=7)
grid = GridSearchCV(model, {"kneighborsclassifier__n_neighbors": range(1, 26, 2)}, cv=cv)
grid.fit(X_tr, y_tr)              # X_tr, y_tr: te same dane co Xtr, y[tr]; best_params_: k
```

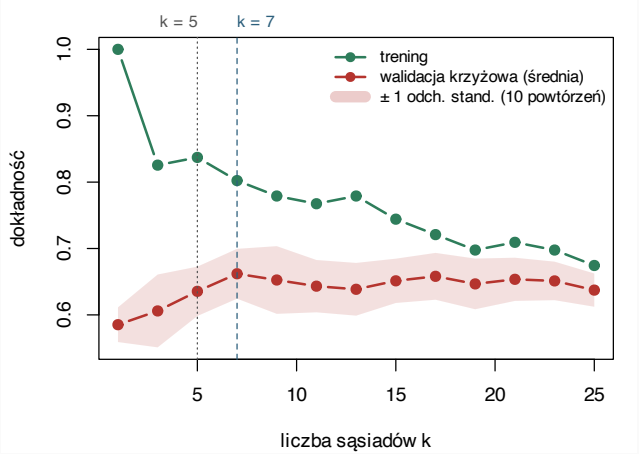
> Jedno wywołanie: części, PCA w każdej części, siatka k. `caret` wybiera $k = 5$.

Wybór hiperparametru k

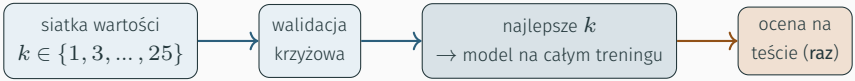
Czerwona krzywa: średnia dokładność w walidacji (10 × 5); zielona: trening.

k	dokladnosc_CV
1	0.585
3	0.606
5	0.635
7	0.662
11	0.643
15	0.651
25	0.637

- $k = 1, 3$: przeuczenie
- Od $k = 5$: **płasko**, różnice $<$ odch. stand. (0,04)
- Maksimum przy $k = 7$; `caret` (ze standaryzacją) wybrał $k = 5$ – różnica mniejsza niż rozrzut



Strojenie hiperparametrów

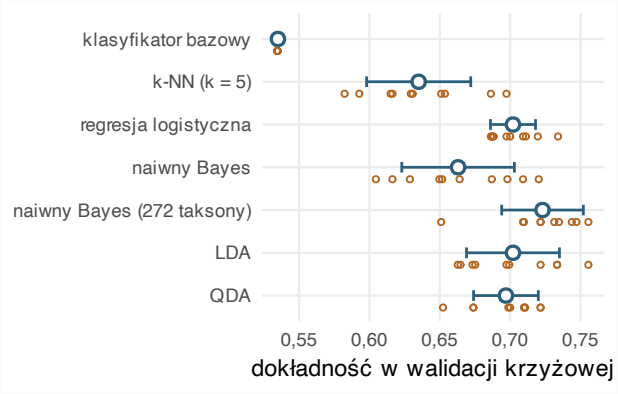




Hiperparametr: ustawienie wybierane przed treningiem (np. k); parametry (wagi) model uczy sam.

Wybór k na teście $\rightarrow$ wynik zawyżony, nie powtórzy się na nowych danych.

Porównanie modeli w walidacji krzyżowej



- Te same części dla wszystkich (10 × 5)
- Regresja logistyczna, LDA, QDA: **ok. 0,70**
- k-NN: 0,635 — słabszy w każdym powtórzeniu
- Ranking **inny** niż na teście

Różnice 0,70–0,72 są mniejsze niż rozrzut: wniosek „k-NN słabszy”, nie „X najlepszy”.

Porównanie modeli w caret

```r
set.seed(7)
folds <- createMultiFolds(y[tr], k = 5, times = 10)        # te same części dla wszystkich
ctrl <- trainControl(method = "repeatedcv", index = folds,
                     preProcOptions = list(pcaComp = 10))
models <- list(
  knn = train(x = Xtr, y = y[tr], method = "knn", preProcess = "pca",
              trControl = ctrl, tuneGrid = data.frame(k = 5)),
  glm = train(x = Xtr, y = y[tr], method = "glm",         preProcess = "pca", trControl = ctrl),
  nb  = train(x = Xtr, y = y[tr], method = "naive_bayes", preProcess = "pca", trControl = ctrl),
  lda = train(x = Xtr, y = y[tr], method = "lda",         preProcess = "pca", trControl = ctrl),
  qda = train(x = Xtr, y = y[tr], method = "qda",         preProcess = "pca", trControl = ctrl))
acc <- summary(resamples(models))$statistics$Accuracy       # 50 ocen na model
round(acc[, c("Min.", "Median", "Mean", "Max.")], 3)         # bwplot() rysuje porównanie
```

```
     Min. Median  Mean  Max.
knn 0.471  0.657 0.681 0.889
glm 0.412  0.706 0.704 0.882
nb  0.471  0.706 0.692 0.824
lda 0.529  0.706 0.722 0.882
qda 0.471  0.706 0.705 0.889
```

Model zmienia się jednym słowem (`method`). Średnio: k-NN 0,68, pozostałe 0,69–0,72; rozrzut (0,41–0,89) jest dużo większy niż różnice. Liczby nieco inne niż na poprzednim slajdzie (`caret` standaryzuje cechy przed PCA).

Porównanie modeli w scikit-learn

```
cv = RepeatedStratifiedKFold(n_splits=5, n_repeats=10, random_state=7)   # te same części
models = {"knn": KNeighborsClassifier(n_neighbors=5),
          "glm": LogisticRegression(penalty=None, max_iter=5000),        # bez kary, jak glm
          "nb":  GaussianNB(),
          "lda": LinearDiscriminantAnalysis(),
          "qda": QuadraticDiscriminantAnalysis()}
for name, model in models.items():
    acc = cross_val_score(make_pipeline(PCA(n_components=10), model), X_tr, y_tr, cv=cv)
    print(f"{name}: {acc.mean():.3f}")
```

```
knn: 0.623
glm: 0.688
nb: 0.660
lda: 0.701
qda: 0.691
```

Jeden potok (PCA + model), te same części; każdy model ma `fit()` i `predict()`. Wniosek jak w R: k-NN najsłabszy, pozostałe 0,66–0,70 (tu PCA bez standaryzacji, stąd inne liczby).

Regresja wartości ciągłej

Regresja: przewidywanie wartości ciągłej

Przykład: przewidujemy **wiek** pacjenta z profilu mikrobiomu (te same 10 składowych).

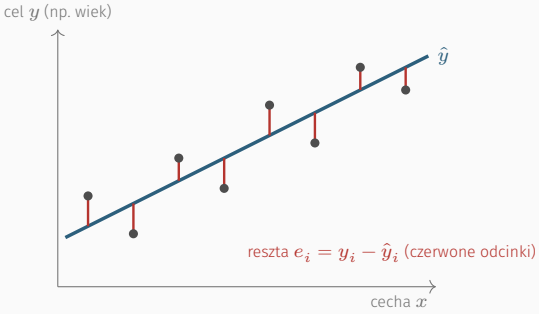
$$\hat{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_{10} x_{10}$$

Regresja liniowa (znana z biostatystyki), współczynniki metodą najmniejszych kwadratów.

Nowe jest tylko to, jak ją oceniamy: błędem predykcji na danych spoza treningu.

	klasyfikacja	regresja
cel y	kategoria (CRC / control)	liczba (wiek)
modele	k-NN, regresja logistyczna, …	**regresja liniowa**, k-NN
miara	dokładność, czułość, swoistość	**błąd** (MAE, RMSE), R^2
punkt odniesienia	klasyfikator bazowy	**średnia**

Regresja liniowa — metoda najmniejszych kwadratów



Metoda najmniejszych kwadratów (*least squares*)

Współczynniki β dobiera się tak, by suma kwadratów reszt — pionowych odległości punktów od prostej — była najmniejsza:

$$\min_{\beta} \sum_{i} (y_i - \hat{y}_i)^2$$

- Kwadrat: reszty dodatnie i ujemne się nie znoszą, a duże pomyłki ważą więcej.
- Przy 10 cechach to samo, tylko zamiast prostej płaszczyzna w 10 wymiarach.

Miary błędu regresji

MAE (*mean absolute error*)

Średni błąd bezwzględny:

$$\text{MAE} = \frac{1}{n}\sum_i |y_i - \hat{y}_i|$$

w jednostkach celu.

RMSE (*root mean squared error*)

Pierwiastek ze średniego kwadratu błędu:

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

mocniej karze duże pomyłki.

R^2 (współczynnik determinacji)

Jaka część błędu średniej znika dzięki modelowi:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$$

Przykład: reszty 2, -4 i 6 lat $\Rightarrow$ MAE $= \frac{2+4+6}{3} = 4$ lata; RMSE $= \sqrt{\frac{4+16+36}{3}} = 4{,}32$ roku — więcej niż MAE, bo pomyłka o 6 lat waży najwięcej.

> R^2: 1 — model idealny; 0 — tak dobry jak zawsze średnia z treningu; poniżej 0 — gorszy od średniej. Punkt odniesienia regresji to średnia, tak jak w klasyfikacji klasyfikator bazowy.

```r
age <- d$cd$age                          # wiek z metadanych pacjentów
dtr <- data.frame(age = age[tr],         Ztr[, 1:10])
dte <- data.frame(age = age[idx_test],   Zte[, 1:10])

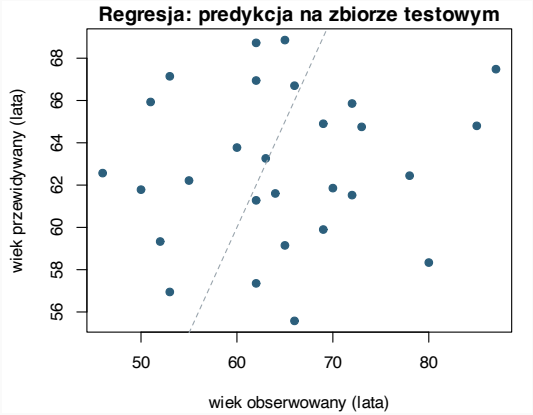
fit <- lm(age ~ ., data = dtr)           # trening na składowych
pr  <- predict(fit, dte)                 # predykcja na teście

sqrt(mean((dte$age - pr)^2))             # RMSE — błąd w latach
mean(abs(dte$age - pr))                  # MAE — błąd w latach
```

```
[1] 10.53796
[1] 8.691536
```

RMSE **10,5** roku, MAE **8,7** roku. RMSE jest większe, bo zdarzają się duże pomyłki (do 22 lat).

Ocena regresji



linia przerywana: predykcja idealna (= wiek obserwowany)

	RMSE	MAE
regresja liniowa	10,5	8,7
„zawsze średnia”	10,3	8,0

błędy w latach; średnia z treningu

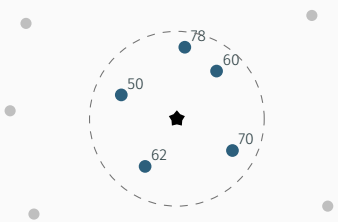
$$R^2 = 1 - \frac{\text{błąd modelu}}{\text{błąd średniej}} = -0{,}04$$

błąd = suma kwadratów różnic na teście

R^2: 1 idealnie · 0 jak średnia · < 0 gorzej niż średnia

Model gorszy od średniej — nie wykryliśmy sygnału wieku.

k-NN dla wartości ciągłej



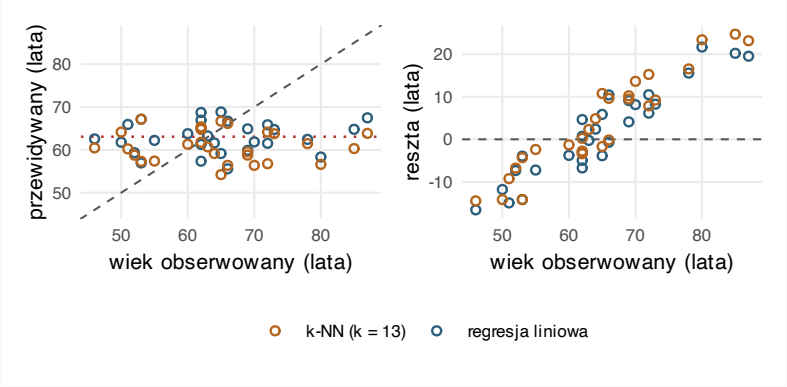
$$\hat{y} = \frac{50+60+62+70+78}{5} = 64 \text{ lata}$$

- Zamiast głosowania — **średnia** celu sąsiadów
- k z walidacji krzyżowej: $k = 13$
- Predykcja nie wychodzi poza zakres wartości z treningu

```
pk <- FNN::knn.reg(train = Ztr, test = Zte, y = age[tr], k = 13)$pred
sqrt(mean((age[idx_test] - pk)^2))      # RMSE na teście (lata)
```

```
[1] 11.58751
```

Regresja liniowa a k-NN na danych przykładowych



Lewy panel: linia przerywana — predykcja idealna, kropkowana — średnia z treningu. Prawy: reszta = obserwowany — przewidywany.

Oba modele przewidują prawie zawsze to samo: wiek bliski średniej z treningu (63 lata).

Regresja wieku i BMI — wyniki

Cel	Model	RMSE test	R^2 test	RMSE CV
wiek (lata)	średnia z treningu	10,31	0	12,02
	regresja liniowa	10,54	−0,044	11,99
	k-NN ($k = 13$)	11,59	−0,263	12,14
BMI	średnia z treningu	3,88	0	4,38
	regresja liniowa	3,95	−0,038	4,50
	k-NN ($k = 15$)	4,28	−0,220	4,30

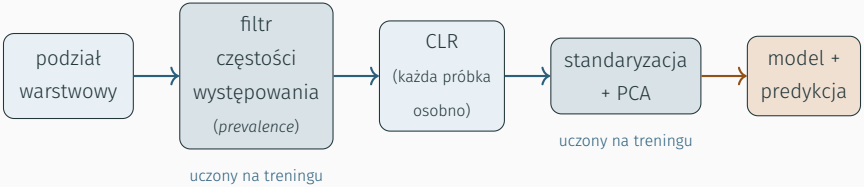
- RMSE CV: błąd w walidacji krzyżowej na treningu (10 × 5)
- Żaden model nie jest lepszy od **średniej**
- R^2 na treningu: 0,17 (wiek) — dopasowanie do szumu

> Wynik negatywny raportujemy tak samo starannie jak pozytywny.

Pełny potok na mikrobiomie

Potok klasyfikatora bez przecieku

Potok (*pipeline*): cała droga od surowych danych (udziałów taksonów) do predykcji. Traktujemy go jak **jeden model** — uczony na treningu, stosowany do testu.



Kroki uczące się z danych (**filtr, standaryzacja, PCA**) — tylko na treningu, test tylko **przekształcamy**.

Zmiana modelu (`knn`, `glm`, `naiveBayes`, `lda`) nie rusza reszty potoku.

Kroki potoku: cel i uczenie się z danych

Krok	Po co	Uczy się z danych?
podział warstwowy	odłożyć test; te same proporcje klas w obu zbiorach	—
filtr częstości występowania	usunąć taksony obecne u mniej niż 10% próbek (prawie same zera)	tak → tylko trening
CLR	udziały sumują się do 1, więc wzrost jednego taksonu obniża pozostałe; CLR zamienia udział na logarytm stosunku do średniej próbki: liczą się proporcje między taksonami, a skala pasuje do PCA i modeli liniowych	nie (każda próbka osobno)
standaryzacja + PCA	272 cechy → 10 składowych (więcej cech niż próbek)	tak → tylko trening
model	uczy się na składowych	tak
ocena	walidacja krzyżowa na treningu; test raz, na końcu	—

CLR (*centred log-ratio*, W2)

$\mathrm{clr}(x)_j = \ln \dfrac{x_j}{g(x)}, \quad g(x)$ — średnia geometryczna udziałów próbki; zera zastępuje się małą stałą.

Reguła: krok, który uczy się z danych, liczymy na treningu (a w walidacji — w każdej części od nowa).

Potok w R (caret)

```
# X, y, tr, idx_test: jak na slajdzie „Dane do modeli”
keep <- colMeans(X[tr, ] > 0) >= 0.10                  # 1. filtr: tylko trening
Xtr  <- clr(X[tr, keep]); Xte <- clr(X[idx_test, keep]) # 2. CLR: każda próbka osobno
ctrl <- trainControl(method = "repeatedcv", number = 5, repeats = 10,
                     preProcOptions = list(pcaComp = 10))
set.seed(7)
fit  <- train(x = Xtr, y = y[tr], method = "glm",       # 3. PCA + model,
              preProcess = "pca", trControl = ctrl)     #    PCA w każdej części
round(fit$results$Accuracy, 3)                          # walidacja krzyżowa
pred <- predict(fit, Xte)                               # 4. test: raz, na końcu
round(mean(pred == y[idx_test]), 3)
```

```
[1] 0.698
[1] 0.75
```

Walidacja krzyżowa **0,70**; test **0,75** (21 z 28). Inny model: zmiana `method`.

Potok w Pythonie (scikit-learn)

```
def clr(M):                                          # CLR: każda próbka osobno
    L = np.log(M + 1e-6)
    return L - L.mean(axis=1, keepdims=True)

# X, y: dane jak w R; tr, te: te same indeksy treningu i testu
keep = (X[tr] > 0).mean(axis=0) >= 0.10              # 1. filtr: tylko trening
X_tr, X_te = clr(X[tr][:, keep]), clr(X[te][:, keep])   # 2. CLR
model = make_pipeline(StandardScaler(), PCA(n_components=10),   # 3. potok:
                      LogisticRegression(penalty=None, max_iter=5000))  # skala, PCA, model
cv = RepeatedStratifiedKFold(n_splits=5, n_repeats=10, random_state=7)
print(round(cross_val_score(model, X_tr, y[tr], cv=cv).mean(), 3))   # walidacja krzyżowa
model.fit(X_tr, y[tr])                               # 4. model końcowy na całym treningu
print(round(model.score(X_te, y[te]), 3))            #    test: raz, na końcu
```

```
0.702
0.75
```

Te same kroki i ten sam wynik co w R. `Pipeline` pilnuje, by skala i PCA uczyły się **tylko** na części uczącej.

Podsumowanie

Typowe pułapki uczenia nadzorowanego

× pułapka	✓ zamiast tego
ocena na treningu	ocena na **teście**
strojenie na teście	strojenie **walidacją krzyżową**
filtr / PCA przed podziałem	przekształcenia uczone **tylko na treningu**
dokładność bez punktu odniesienia	porównanie z **klasyfikatorem bazowym**
jeden podział	**walidacja krzyżowa**

Błąd to zwykle nie zły model, lecz **nieuczciwa ocena**.

Porównanie modeli wykładu

	granica	interpretacja	prawdopodobieństwa	hiperparametr	dokładność CV
k-NN	nieregularna	brak	tylko $k+1$ wartości	k	0,635
regresja logistyczna	liniowa	ilorazy szans	ciągłe	brak	**0,702**
naiwny Bayes	zwykle krzywa	rozkłady w klasach	skrajne	brak	0,663
LDA	liniowa	oś LD1	ciągłe	brak	**0,702**
QDA	kwadratowa	słaba	ciągłe	brak	0,697

skrajne = bliskie 0 lub 1, gdy cechy są skorelowane; ilorazy szans wprost tylko na taksonach (przy PCA — na składowych); dokładność CV: 10 powtórzeń × 5 części

Czołówka (regresja logistyczna, LDA, QDA): ok. 0,70, różnice mniejsze niż rozrzut; k-NN słabszy. Wybór zależy od potrzeby: wyjaśnienie, prawdopodobieństwa czy elastyczna granica.

Wybór modelu

Gdy zależy nam na…	…sięgamy po
wyjaśnieniu, które cechy decydują	regresję logistyczną (ilorazy szans)
prawdopodobieństwach i progu	regresję logistyczną
elastycznej, nieliniowej granicy	k-NN (lub modele złożone)
szybkim punkcie odniesienia przy $p \gg n$	naiwny Bayes
jednej osi pokazującej rozdzielenie klas	LDA (oś LD1, oceniana na teście)
klasach o wyraźnie różnym rozrzucie i dużej próbie	QDA

Najpierw model prosty i interpretowalny; złożony — gdy walidacja pokazuje realny zysk.

Do zapamiętania

- Model uczy się zależności $X \to y$, oceniamy go na **nowych** danych — klasyfikacja i regresja tak samo
- Modele różni **kształt granicy**: liniowa (regresja logistyczna, LDA) albo elastyczna (k-NN, naiwny Bayes, QDA)
- k-NN: liczba sąsiadów k steruje złożonością granicy
- Regresja logistyczna: **prawdopodobieństwa** i ilorazy szans
- Hiperparametry stroimy **walidacją krzyżową** — nigdy na teście
- Zawsze **punkt odniesienia**: klasyfikator bazowy (zawsze klasa większościowa), w regresji średnia

Jedna myśl

Każdy model upraszcza rzeczywistość; który jest **użyteczny**, rozstrzyga tylko uczciwa ocena na danych, których nie widział.

Literatura

- James, Witten, Hastie, Tibshirani. *An Introduction to Statistical Learning* — rozdz. 2 (dokładność, obciążenie–wariancja), 3 (regresja liniowa a k-NN), 4 (regresja logistyczna, LDA, QDA, naiwny Bayes), 5 (walidacja krzyżowa). Wydania R i Python (ISLP): statlearning.com.
- Boehmke, Greenwell. *Hands-On Machine Learning with R* — k-NN, próbkowanie i walidacja: bradleyboehmke.github.io/HOML.
- Kuhn, Johnson. *Feature Engineering and Selection* (bezpłatnie): feat.engineering — o przecieku i potokach.
- Fisher (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 7:179–188 — liniowa analiza dyskryminacyjna.
- Efron (1975). The efficiency of logistic regression compared to normal discriminant analysis. *JASA* 70:892–898.
- Domingos, Pazzani (1997). On the optimality of the simple Bayesian classifier under zero-one loss. *Machine Learning* 29:103–130.
- Dokumentacja pakietu `MASS` (`lda`, `qda`): cran.r-project.org/package=MASS.

Na następny wykład

Wykład 5: Ocena modelu

- Pozostałe miary z macierzy pomyłek: precyzja, NPV, F1
- Miary uwzględniające obie klasy: dokładność zbalansowana (ang. *balanced accuracy*), MCC, kappa
- Krzywe ROC i PR; niezrównoważenie klas; kalibracja prawdopodobieństw
- Ocena wieloklasowa i miary regresji; rzetelność oceny

Dziękuję za uwagę

dr inż. Bartosz Czech · bartosz.czech@upwr.edu.pl