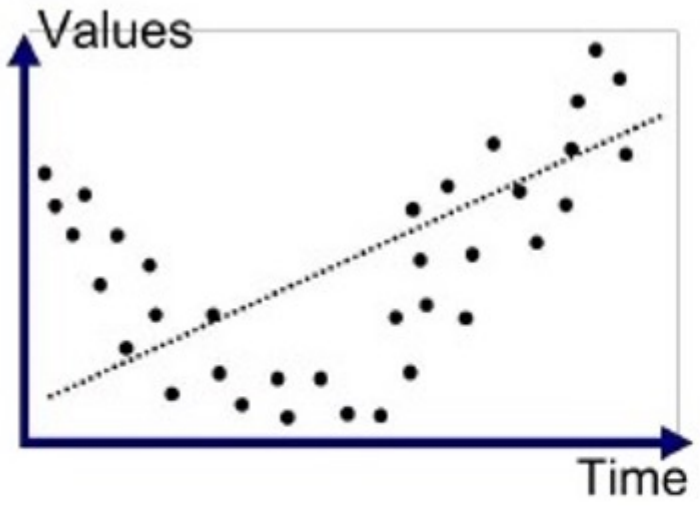
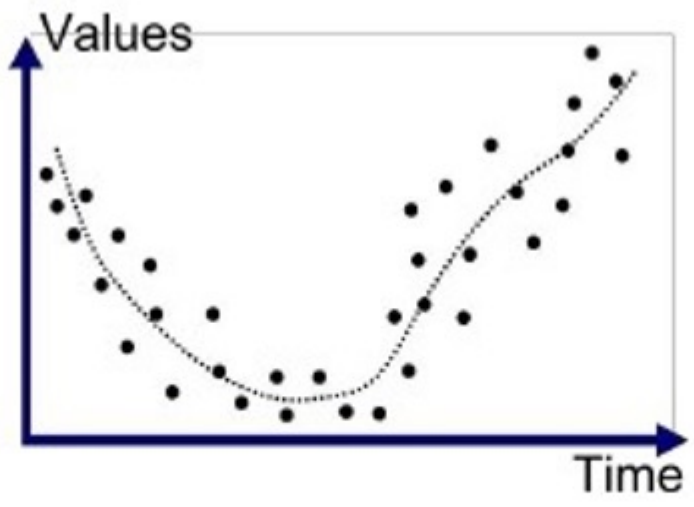
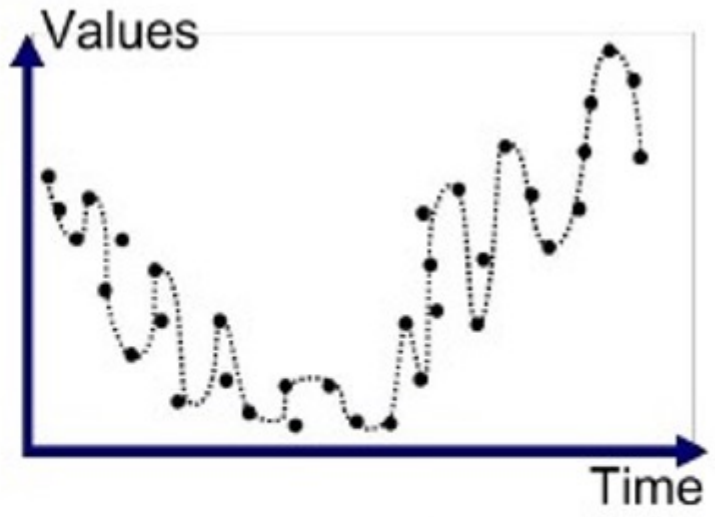
Który model lepszy?







Walidacja krzyżowa

(Cross-validation)

- Do oszacowania skuteczności modelu wyodrębniane są dwa podzbiory - zbiór treningowy i testowy

- Problem jednak pojawia się, gdy chcemy poddac model optymalizacji, czyli dokonać doboru modelu, który polega na optymalizacji jego hiperparametrów oraz porównaniu wpływu różnych konfiguracji na skuteczność jego działania

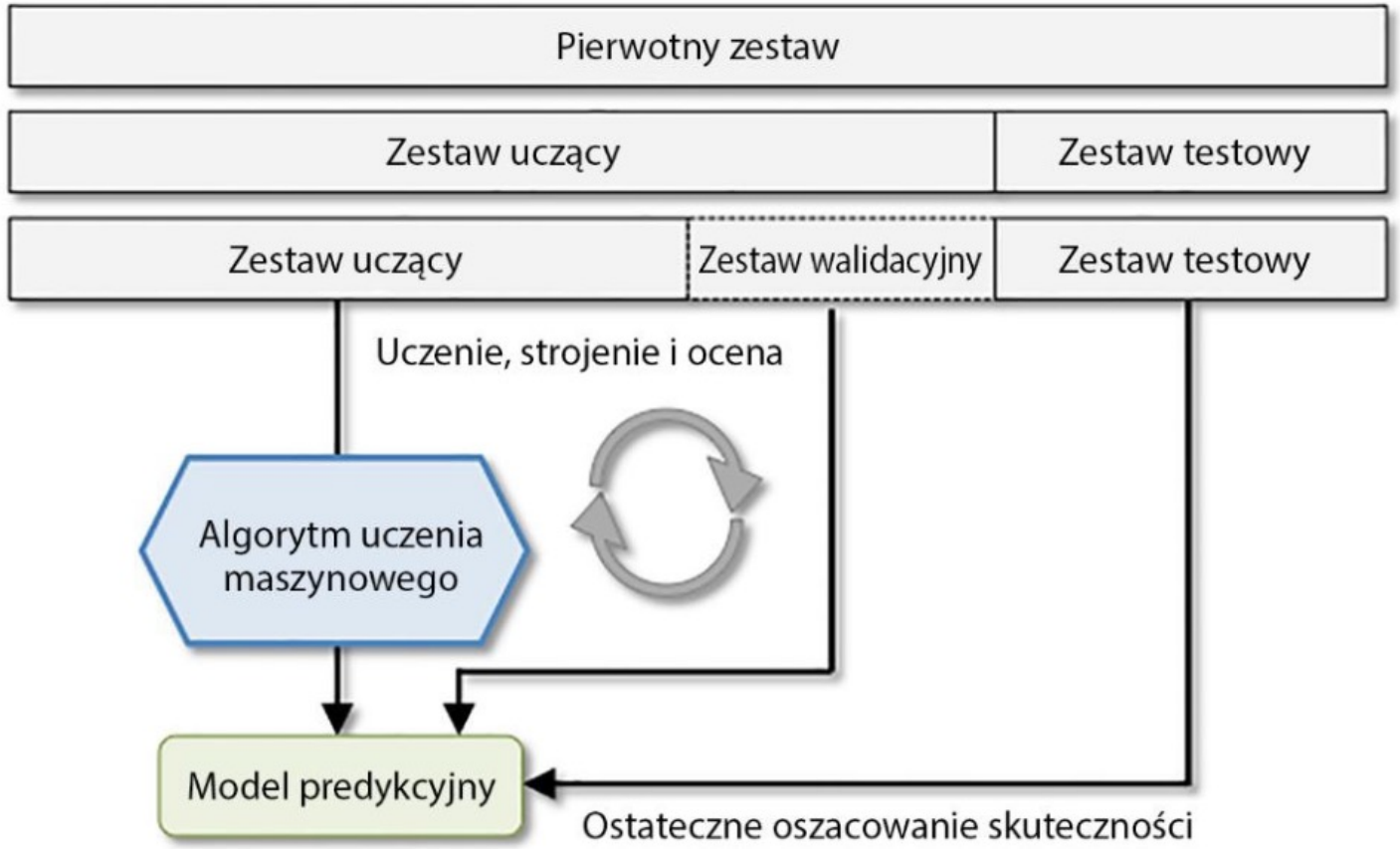
- Mając tylko dwa podzbiory, przy każdej iteracji (nowej konfiguracji hiperparametrów) wykorzystujemy do oceny działania ten sam zbiór testowy, co prowadzi do tego, że staje się on częścią zbioru uczącego – może to skutkować niestabilnością wyników

Walidacja krzyżowa

(Cross-validation)

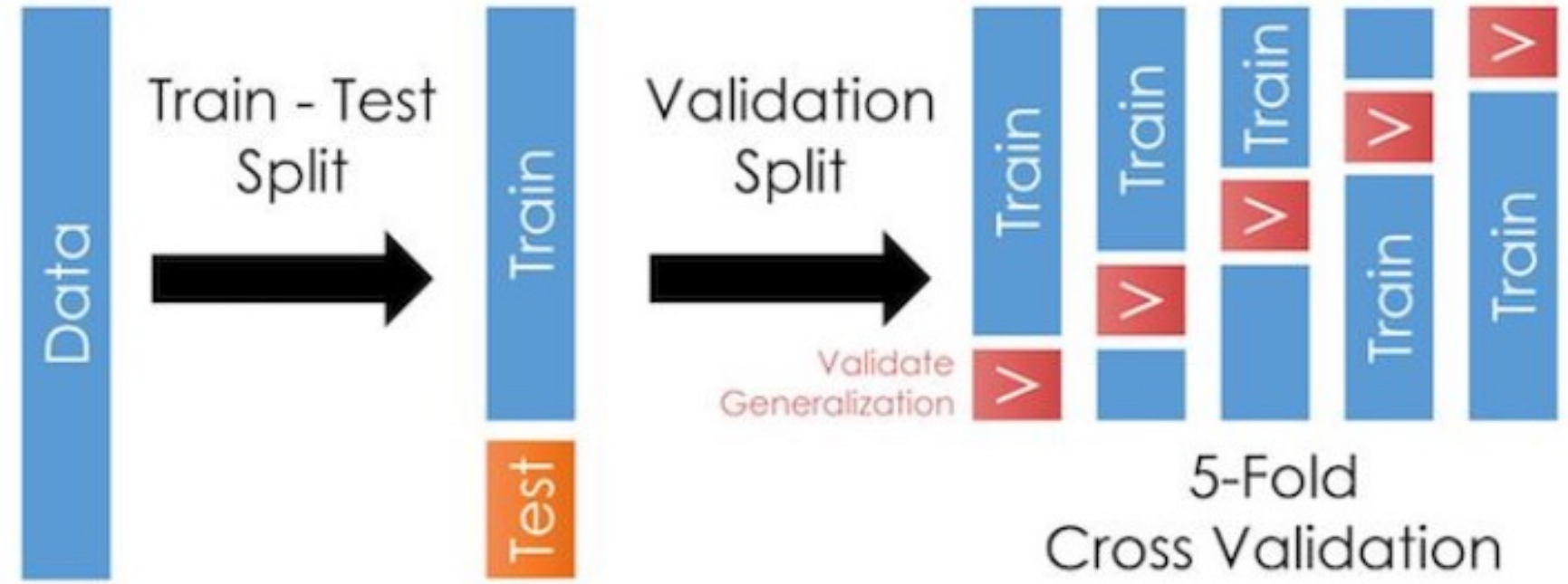
- W odpowiedzi na ten problem powstała ulepszona metoda wydzielania, która wyodrębnia trzy podzbiory: treningowy, walidacyjny i testowy, gdzie:
  - zbiór treningowy służy do uczenia modelu
  - zbiór walidacyjny do weryfikacji skuteczności
  - zbiór testowy stosuję się do oszacowania skuteczności finalnego modelu

- Jednak to rozwiązanie również ma swoje wady – jedna z nich to duża wrażliwość oszacowania na sposób podziału danych, co może prowadzić do istotnych różnic w wynikach w zależności od przyjętych wielkości poszczególnych podzbiorów



K-krotna walidacja krzyżowa

(k-fold cross-validation)

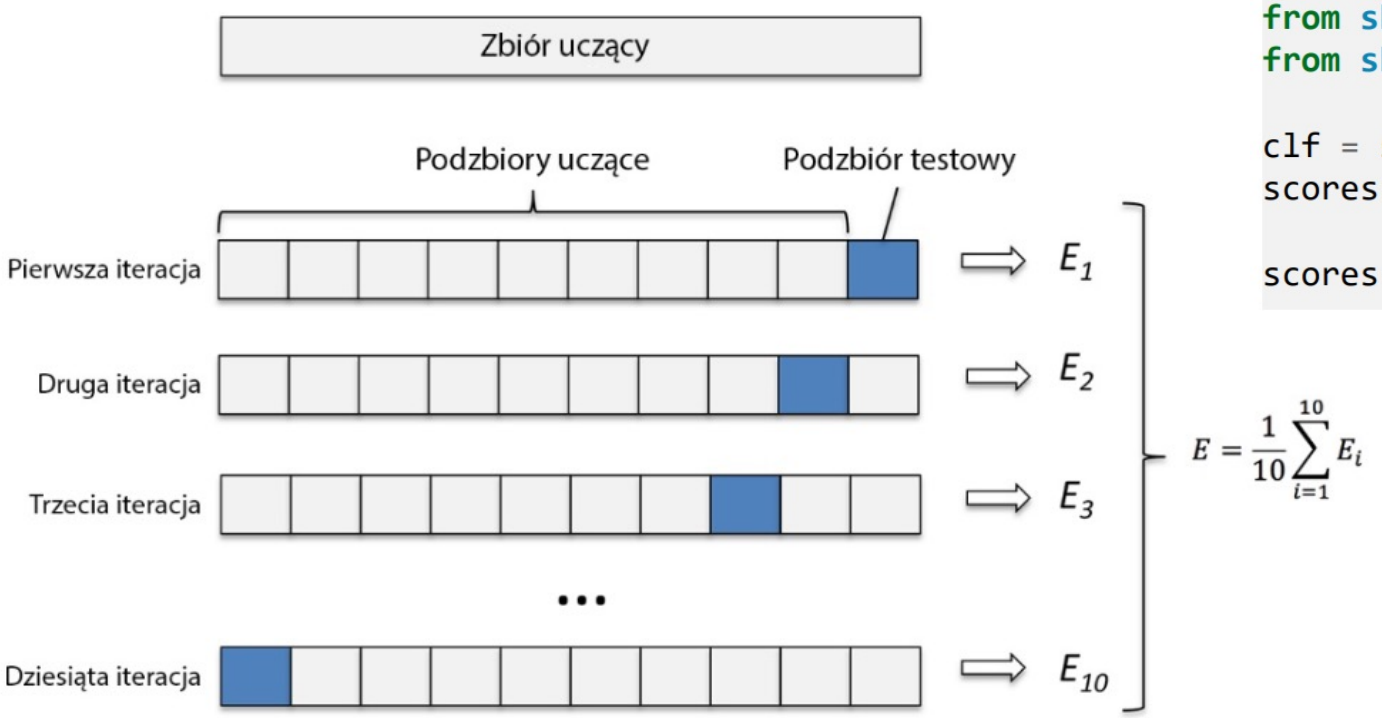


K-krotna walidacja krzyżowa

(k-fold cross-validation)

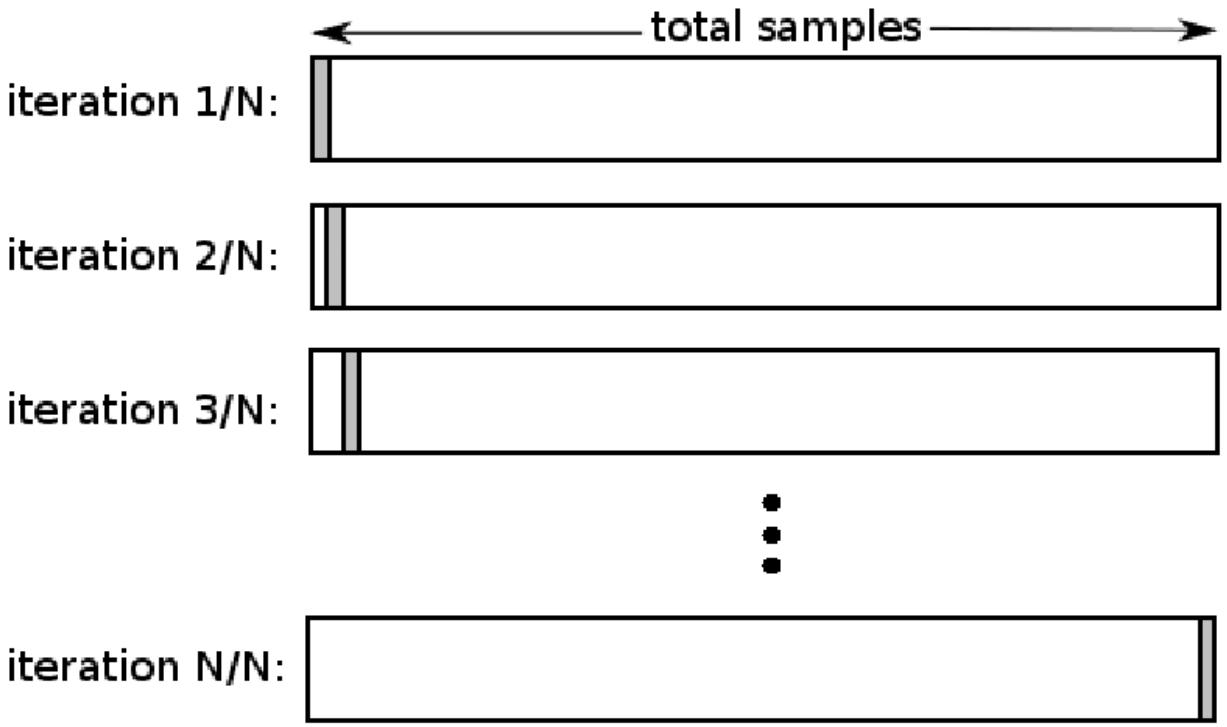
- Największą zaletą k-krotnej kroswalidacji jest to, że każdy przykład w danych treningowych jest użyty dokładnie raz do trenowania i raz do walidacji – dokonywane jest próbkowanie bez zwracania

- Model jest trenowany k razy, następnie finalna ocena skuteczności jest uzyskiwany poprzed uśrednienie wyników ze wszystkich k modeli (k iteracji)



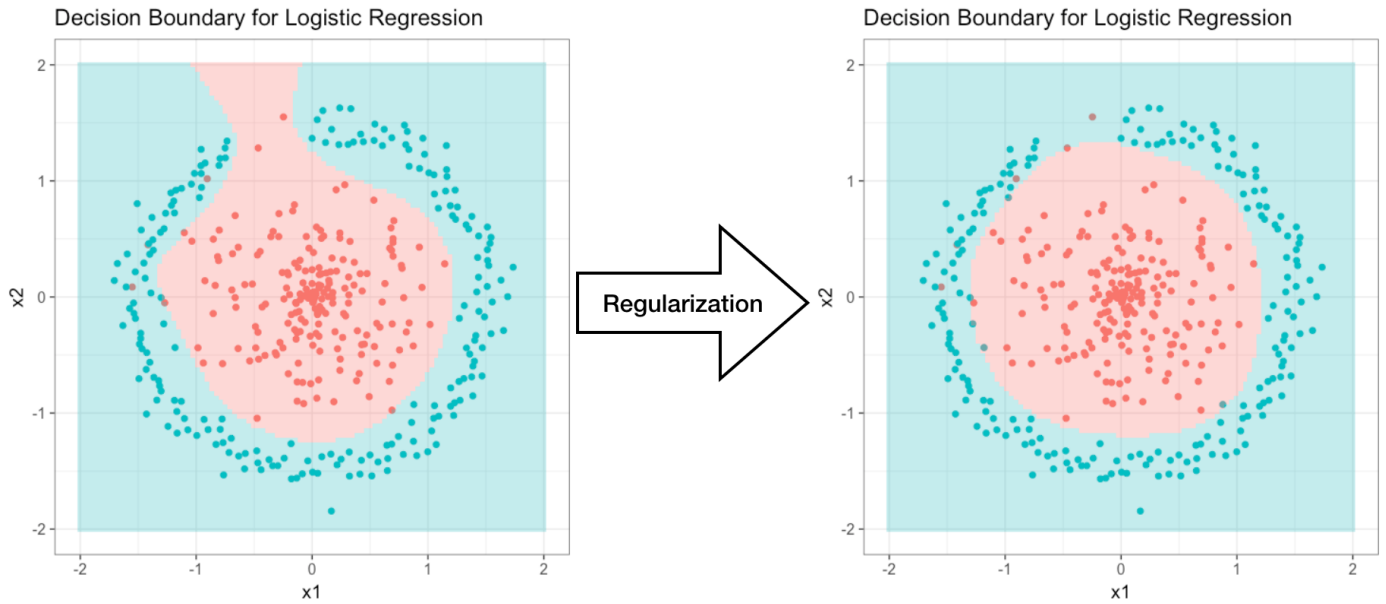

```
from sklearn.model_selection import cross_val_score
from sklearn import metrics

clf = svm.SVC(kernel='linear', C=1)
scores = cross_val_score(clf, iris.data, iris.target,
         scoring= f1_macro', cv=5)
scores array([0.96..., 1. ..., 0.96..., 0.96..., 1. ])
```

Metoda Leave-one-out cross-validation



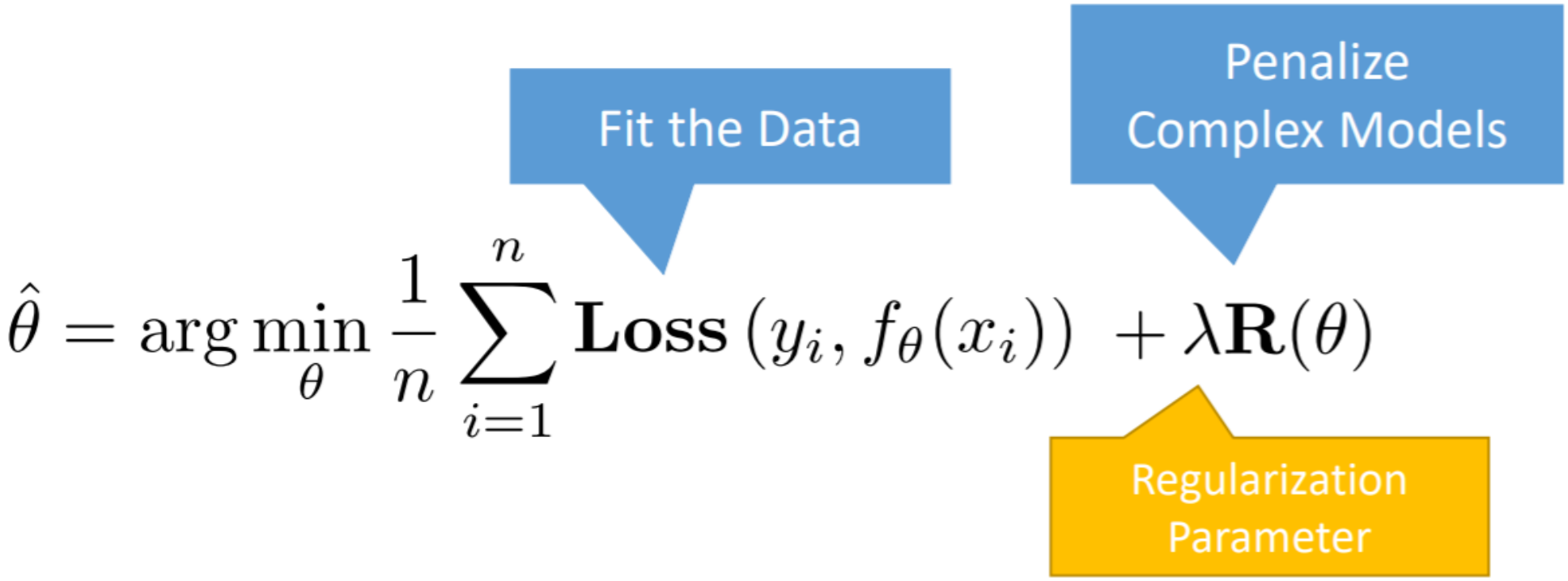
Regularyzacja

- Model przeuczony bardzo kiepsko sprawdzi się na nieobserwowanych dotąd danych.
- Potrzeba mechanizmu, który potrafi zapobiegać przeuczeniu, ’sterować’ tym na ile model ma generalizować dane trenujące.
- Do funkcji kosztu (na etapie trenowania modelu) dodajemy komponent regularyzacyjny.



Regularyzacja

- Jak określić R(θ)?
- Jak wybrać odpowiednią wartość dla λ?



Regularyzacja L2 – Ridge Regression

- W przypadku zwykłej regresji szukane są wagi minimalizujące funkcję kosztu postaci:

Linear Regression

$$J(w) = \sum_{i=1}^{m} (y_i - w^T x_i)^2$$

Logistic Regression

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} (y_i \log(h_\theta(x_i)) - (1 - y_i)\log(1 - h_\theta(x_i)))$$

- Dla regresji grzbietowej - Ridge regression lub Thikonov - dodana jest *funkcja kary*, czyli składnik ograniczający wartości współczynników:

Linear Regression

$$J(w) = \sum_{i=1}^{m} (y_i - w^T x_i)^2 + \lambda\|w\|^2$$

Logistic Regression

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m} \big(y_i \log\big(h_\theta(x_i)\big) - (1 - y_i)\log\big(1 - h_\theta(x_i)\big)\big) + \lambda\|\theta\|^2$$

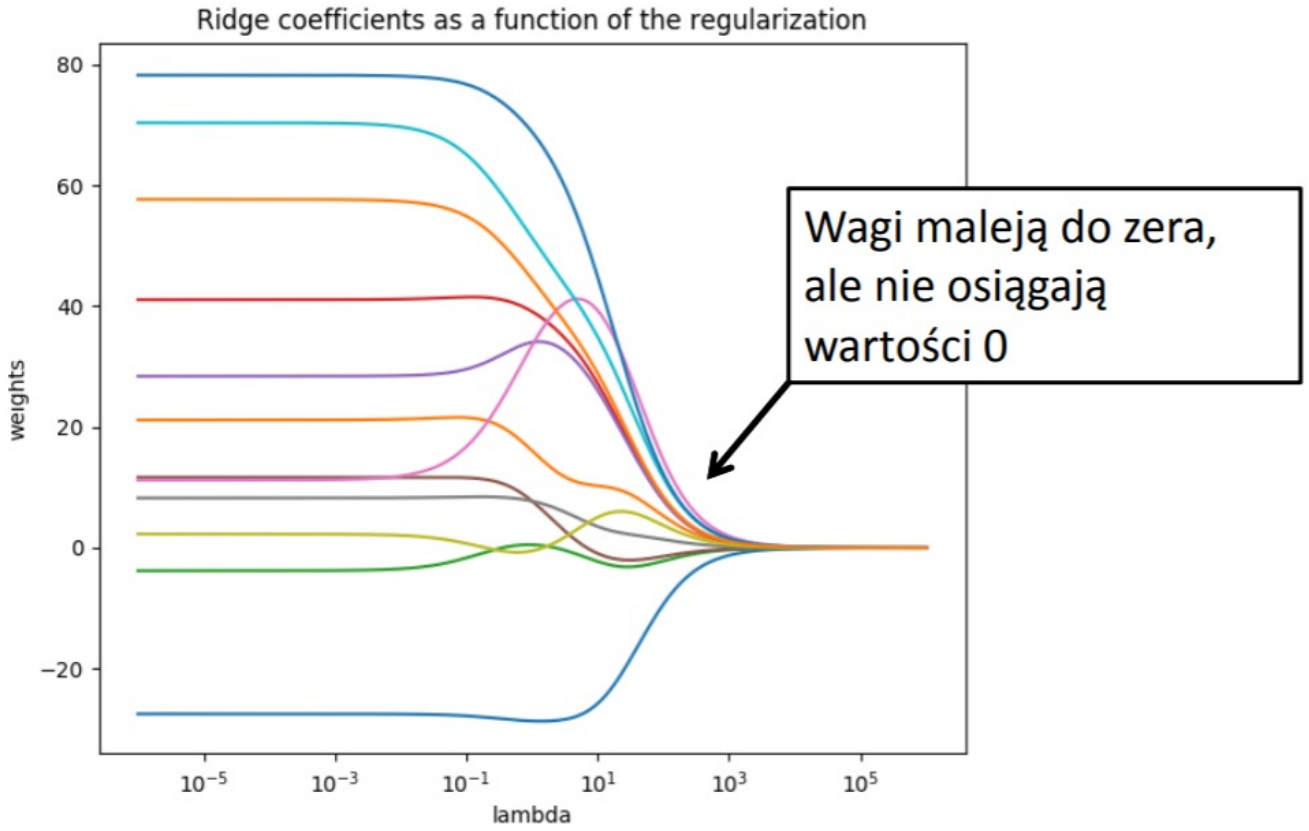
- Czynnik $\|w\|^2$ to tzw. norma L^2, czyli norma euklidesowa:

$$w^T w = \sum_{i=1}^{n} w_i^2$$

- Learning rate, czyli λ określa udział kary w funkcji celu (niekiedy learning rate oznaczany jest jako α)

Regularyzacja L2 – Lambda

- Jeżeli $\lambda = 0$, funkcja celu jest taka sama, jak dla zwykłej regresji
- Dla małych wartości λ wpływ czynnika regularyzującego będzie mniejszy – współczynniki będą się powiększać
- Dla dużych wartości λ współczynniki będą bliskie zeru (i większości przypadków błąd RSS będzie duży)



https://scikit-learn.org/stable/auto_examples/linear_model/plot_ridge_coeffs.html?highlight=regularisation

Regularyzacja L1 – Lasso

- W przypadku regularyzacji L1 składnikiem regularyzującym jest norma L1 (czyli suma wartości bezwzględnych wag)

Linear Regression

$$J(w) = \sum_{i=1}^{m} (y_i - w^T x_i)^2 + \lambda \sum_{i=1}^{n} |w_i|$$

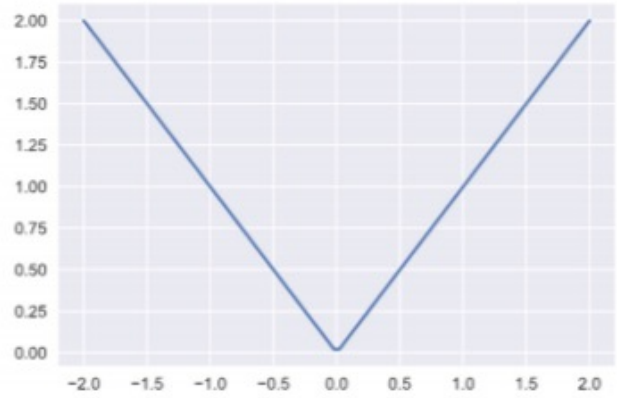
Logistic Regression

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \big(y_i \log\big(h_\theta(x_i)\big) - (1 - y_i) \log\big(1 - h_\theta(x_i)\big)\big) + \lambda \sum_{i=1}^{n} |w_i|$$

- Lasso może sprowadzić wartości wag do 0, przez co ten typ regulazycacji działa jak mechanizm wyboru atrybutów (feature selection) - stopniowo odrzuca współliniowe atrybuty, pozostawia zbiór najbardziej istotnych (tych, które najlepiej „objaśniają” zmienność wartości wyjściowych).

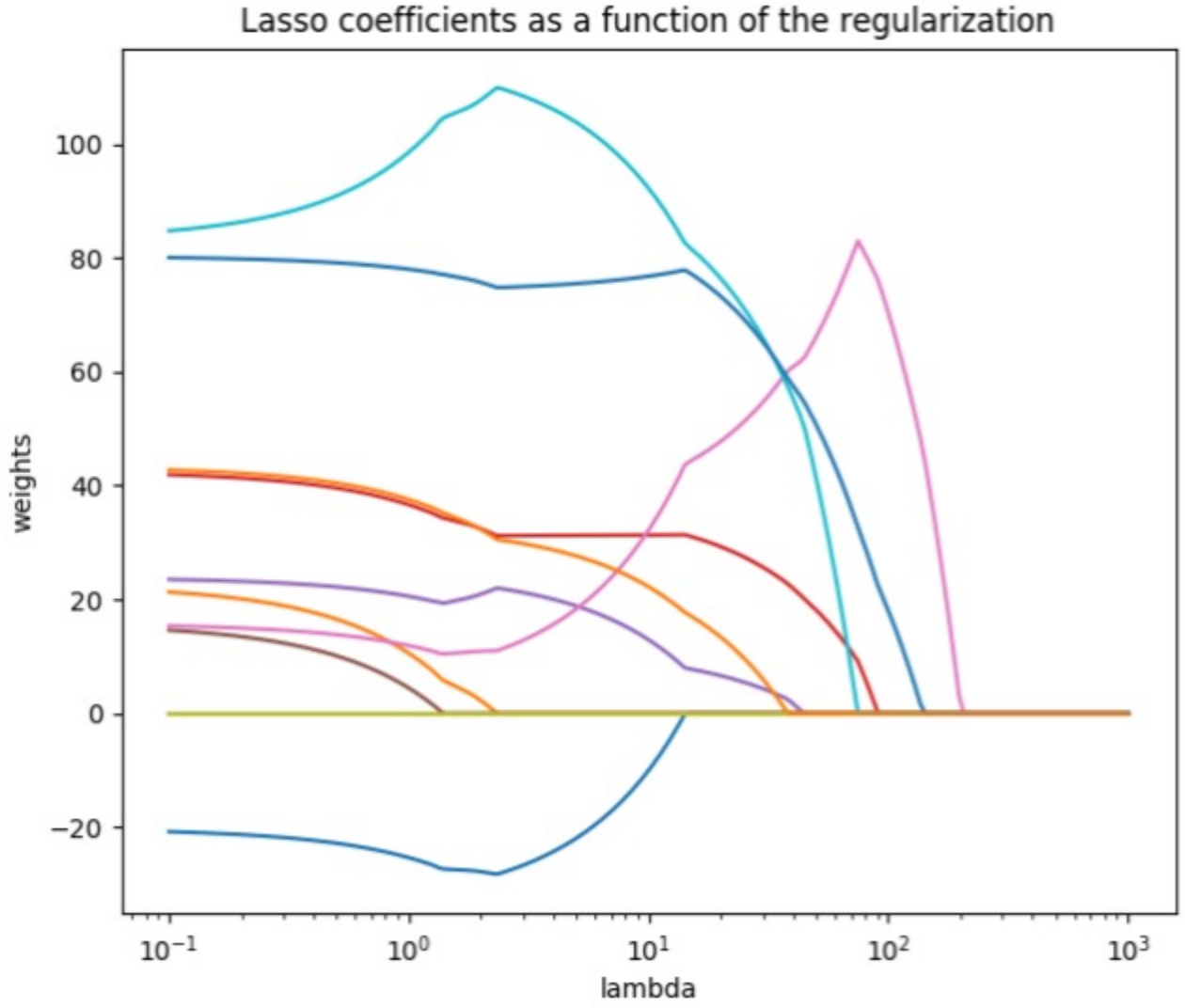
LASSO (L1-Reg)

$$R_{\text{Lasso}}(\theta) = \sum_{i=1}^{d} |\theta_i|$$



Regularyzacja L1 – Lambda

- Dla L1 wraz ze wzrostem lambda kolejne współczynniki będą znikać (przyjmować wartość 0)
- Dla L2 wagi będą stawały się dowolnie małe, ale nie zanikały zupełnie



Regularyzacja Elastic Net

(Elastic Net regression)

- Łączy regresję grzbietową z Lasso

$$J(w) = \frac{\sum_{i=1}^{m}(y_i - w^T x_i)^2}{2m} + \lambda\left(\frac{1-\alpha}{2}\sum_{i=1}^{n} w_i^2 + \alpha\sum_{i=1}^{n}|w_i|\right)$$

Lambda – jak wybrać parametr

- Stosując metodę walidacji krzyżowej: dla siatki różnych wartości λ obliczamy błąd walidacji krzyżowej dla tego wyboru λ.
- Wybieramy tę wartość λ, dla której ten błąd jest najmniejszy
- Trenujemy model już na całych danych, z wybranym λ.