

## 1 Walidacja krzyżowa

Algorytm wykorzystywany podczas nauki modelu ma za zadanie znalezienie takich parametrów, które sprawiają, że model odwzorowuje dane wykorzystane do nauki w sposób jak najlepszy z możliwych. Jeśli do walidacji modelu wykorzystamy inną, niezależną próbkę danych pochodzącą z tego samego zbioru co podzbiór uczący, zazwyczaj okaże się, że model nie działa aż tak dobrze jak przy użyciu zbioru uczącego. Rozmiar tej różnicy zwiększa się, szczególnie wtedy gdy wielkość zbioru treningowego jest niewielka, lub gdy liczba parametrów modelu jest bardzo duża. Walidacja krzyżowa to metoda statystyczna, która ma za zadanie zminimalizować tę różnicę przez co pomaga ocenić i zwiększyć trafność przewidywań modelu predykcyjnego.

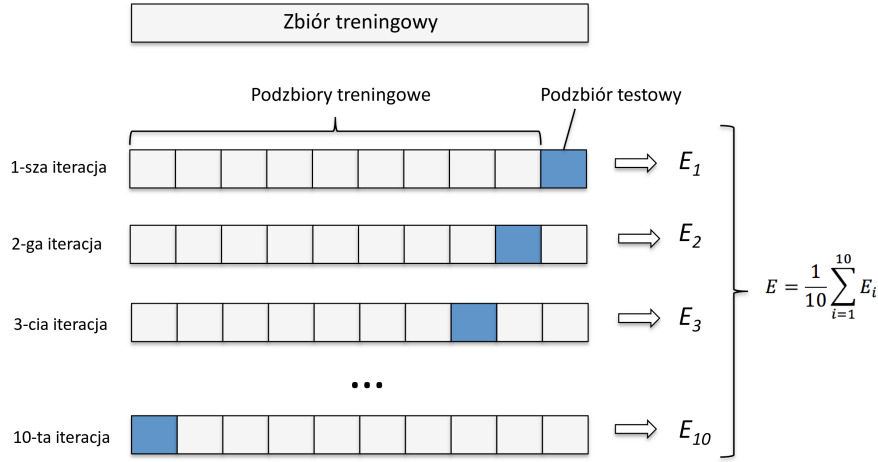
W najprostszym przykładzie walidacji krzyżowej zbiór danych dzieli się na dwa podzbiory: uczący i walidacyjny. Podczas gdy zbiór uczący służy do nauki modelu, zbiór walidacyjny wykorzystuje się aby zmierzyć błąd modelu na nieznanym zbiorze danych.

W algorytmie  $k$ -krotnej walidacji krzyżowej zbiór danych jest losowo dzielony na  $k$  równych wielkością podzbiorów. Jeden z  $k$  podzbiorów jest przeznaczony na zbiór walidacyjny, pozostałe  $k - 1$  podzbiorów służą jako dane treningowe. Powyżej opisana procedura jest powtarzana  $k$  razy, a każdy  $k$  podzbiorów dokładnie raz zostaje wykorzystany jako zbiór testowy. Następnie  $k$  wyników modelu jest uśrednianych dając w rezultacie jeden wynik. Rysunek 1 przedstawia sposób działania 10-krotnej walidacji krzyżowej.

Cytując [Page 184, An Introduction to Statistical Learning, 2013.], "(...) istnieje pewien kompromis między obciążeniem a wariancją, związany z wyborem parametru  $k$  w  $k$ -krotnej walidacji krzyżowej. Zazwyczaj stosuje się wartości z przedziału od 5 do 10, ponieważ pokazano empirycznie, że w takim wypadku otrzymujemy przewidywania, które nie cierpią nadmiernie ani z powodu dużego obciążenia ani dużej wariancji." Podczas treningu modelu wybrano  $k = 6$ , wykorzystując fakt, że liczba próbek w zbiorze danych jest całkowicie podzielna przez tę liczbę co zapewnia równy rozmiar wszystkich zbiorów treningowych i walidacyjnych.

## 2 Wczesne zatrzymanie

Algorytmy uczenia maszynowego dopasowują parametry modelu na podstawie danych treningowych o skończonym rozmiarze. Podczas procesu szkolenia model jest oceniany na podstawie tego, jak dobrze przewiduje obserwacje zawarte w tym zbiorze. Jednak celem uczenia maszynowego jest stworzenie modelu, który



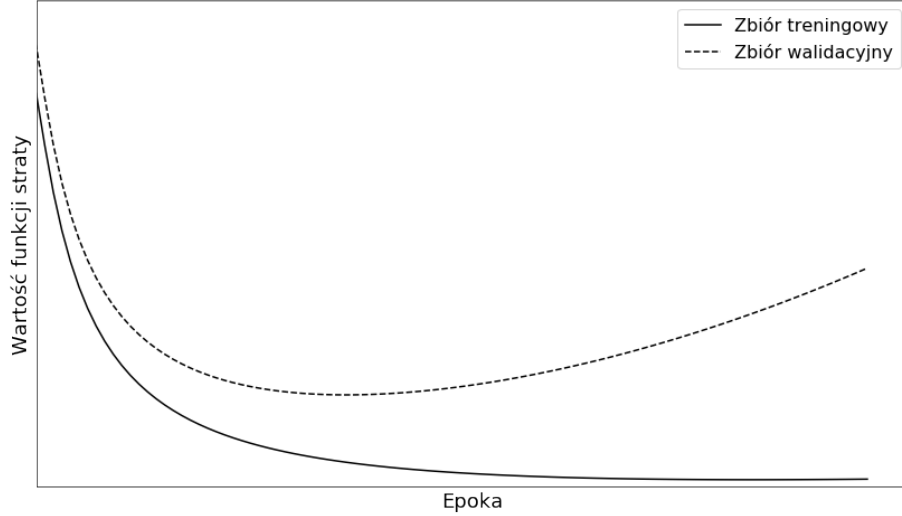
Rysunek 1: Na podstawie [Python Machine Learning Book by Sebastian Raschka]

ma zdolność do przewidywania uprzednio niewidzianych obserwacji. Nadmierne dopasowanie to zjawisko pojawiające się wtedy gdy model za bardzo dopasowuje się do danych w zbiorze uczącym co powoduje zmniejszenie wartości błędu na tym zbiorze lecz równocześnie jest przyczyną wzrostu błędu na zbiorze testowym. Nadmierne dopasowanie modelu to problem, który może się pojawiać gdy model zawiera więcej parametrów niż wymagałaby tego natura modelowanego zjawiska. Sieć neuronowa to struktura skłonna do przeuczenia. Podczas gdy obserwowany błąd obliczany w oparciu o dane treningowe spada, w pewnym momencie wartość błędu dla zbioru walidacyjnego zaczyna wzrastać. Rysunek 2 przedstawia często zamieszczane w literaturze, wyidealizowane krzywe zmiany wartości funkcji straty w czasie, dla zbiorów treningowego i walidacyjnego. Najlepszy model predykcyjny miałby parametry, które odpowiadają momentowi globalnego minimum dla zbioru walidacyjnego.

W dziedzinie uczenia maszynowego, metoda wczesnego zatrzymania to forma regularyzacji, która pozwala uniknąć problemu przeuczenia, zatrzymując naukę modelu gdy wartość funkcji straty na zbiorze walidacyjnym zaczyna wzrastać. Rzeczywisty przebieg wartości funkcji straty ma wiele lokalnych minimów, dlatego na podstawie obserwacji krzywych uczenia dokonano wyboru kryteriów zatrzymania nauki modelu. Niech  $\Theta_{wa}(t)$  to wartość funkcji straty na zbiorze walidacyjnym po  $t$  epokach,  $\Theta_{min}(t)$  to dotychczasowe minimum funkcji straty na zbiorze walidacyjnym po  $t$  epokach, definiowane jako:

$$\Theta_{min}(t) \equiv \min_{t' < t} \Theta_{wa}(t')$$

Niech  $\Theta_{sr}(t)$  będzie średnią wartością funkcji straty dla zbioru walidacyjnego z ostatnich 10 epok.



Rysunek 2

$$\Theta_{sr}(t) \equiv \frac{1}{10} \sum_{i=0}^{10} \Theta_{wa}(t-i)$$

Oraz zdefiniujemy pomocniczy parametr  $GL(t)$

$$GL(t) \equiv \frac{\Theta_{sr}(t)}{\Theta_{min}} - 1$$

Podczas nauki przedstawionego modelu, do wczesnego zatrzymania wystarczyło spełnienie jednego z dwóch obowiązujących warunków:

- $\Theta_{min}(t) = \Theta_{min}(t+200)$  dla wszystkich  $t \in [t, t+200]$ , brak zmniejszenia minimalnej wartości funkcji straty dla zbioru walidacyjnego przez 200 epok
- $GL(t) > 2$ , względny wzrost średniej wartości funkcji straty przez ostatnie 10 epok względem osiągniętego minimum jest większy niż 200%

Po skończeniu nauki, wybierany jest model, który ma najmniejszą wartość funkcji straty na zbiorze testowym.

### 3 Ilość neuronów

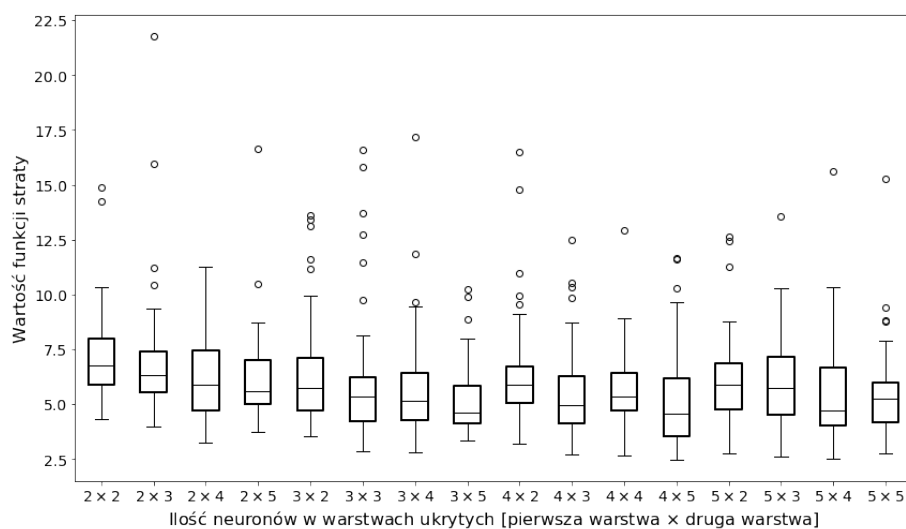
Architektura sieci neuronowej, tzn. ilość warstw ukrytych oraz ilość neuronów w warstwach ukrytych jest zdeterminowana przez wymiar danych wejściowych, rodzaj rozwiązywanego problemu (klasyfikacja czy regresja) oraz relację między zmiennymi objaśniającymi i zmienną objaśnianą.

Uogólniony model liniowy przydatny w szerokim zakresie zastosowań, nie potrzebując żadnej warstwy ukrytej. Bywa szczególnie przydatny gdy zbiór zawiera mało danych lub są one obciążone dużą niedokładnością. Nawet w przypadku gdy relacja między zmiennymi jest lekko nieliniowa, użycie prostego modelu liniowego może skutkować lepszym uogólnieniem problemu niż skomplikowany model będący wrażliwy na każdy szum znajdujący się w danych. Zgodnie z uniwersalnym twierdzeniem aproksymacyjnym jedna warstwa ukryta z wystarczająco dużą liczbą neuronów wystarcza aby z dowolną dokładnością dowolną ciągłą funkcję [cybenko]. Jeśli zmienna objaśniająca jest jednowymiarowa, wydaje się, że nie odniesiemy żadnej korzyści z skonstruowania sieci neuronowej o więcej niż jednej warstwie ukrytej. Sprawy komplikują się jednak gdy zmienna wejściowa jest dwu lub więcej wymiarowa. Dwuwarstwowa sieć neuronowa zachowuje właściwości jednowarstwowej sieci neuronowej oraz osiąga zdolność nauki każdego problemu klasyfikacyjnego [1995 Bishop 123], ponadto wielowarstwowa sieć neuronowa z dwoma warstwami może skutkować dokładniejszymi wynikami wykorzystując mniejszą ilość parametrów niż jednowarstwowa sieć [Chester (1990)]. Na tej podstawie, do rozwiązania problemu regresji gdzie wejściem jest para liczb  $(\varepsilon, Q^2)$  postanowiłem wybrać sieć neuronowa z dwoma warstwami ukrytymi.

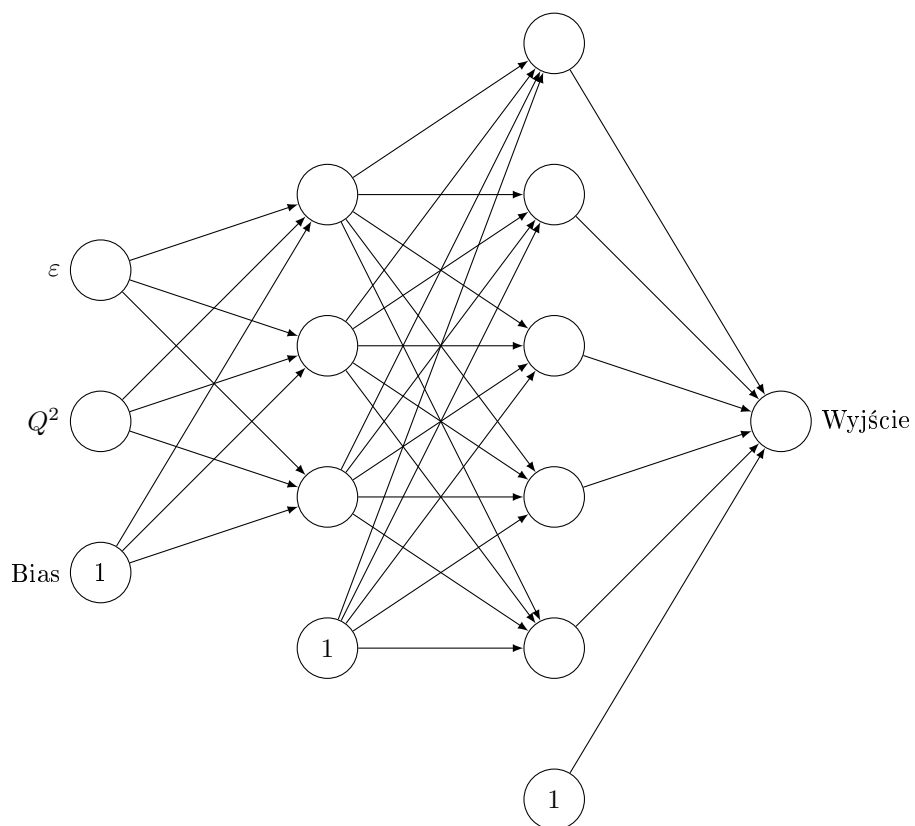
Aby znaleźć odpowiednią liczbę neuronów w dwóch warstwach ukrytych, stworzyłem siatkę  $[2, 3, 4, 5] \times [2, 3, 4, 5]$  neuronów i sprawdziłem, która konfiguracja daje najmniejszy błąd zbioru walidacyjnego. Dane zostały podzielone na zbiór treningowy i testowy w stosunku 2:1. Dla każdej konfiguracji wytrenowano 50 sieci i sprawdzono jak wygląda statystyka błędu. Tabela 1 zawiera porównanie liczby parametrów sieci neuronowej w zależności od liczby neuronów w warstwach ukrytych. Do eksperymentów wybrano konfiguracje charakteryzujące się rozsądną w porównaniu do rozmiaru danych wejściowych liczbą parametrów. Rysunek 3 przedstawia rozkłady minimalnej wartości funkcji straty uzyskanej na danych walidacyjnych uzyskanej z 50 treningów sieci dla każdej konfiguracji ilości neuronów. Wykres pudełkowy to forma graficznej prezentacji rozkładu, która pozwala w łatwy sposób ukazać położenie, rozproszenie oraz kształt empirycznego rozkładu badanej cechy statystycznej. Konfiguracja  $3 \times 5$  charakteryzuje się najniższą medianą wartości funkcji straty oraz małą liczbą wartości odstających. Ta obserwacja pozwoliła zdecydować, że liczby neuronów będą wynosiły 3 i 5 w odpowiednio pierwszej i drugiej warstwie ukrytej, co za tym idzie sieć będzie miała 36 parametrów.

Tabela 1: Liczba parametrów sieci neuronowej z dwoma warstwami ukrytymi w zależności od liczby neuronów w warstwach

I warstwa \ II warstwa	2	3	4	5
	2	3	4	5
2	14	18	22	26
3	19	24	29	34
4	24	30	36	42
5	29	36	43	50



Rysunek 3: Wykresy pudełkowe przedstawiające rozkład wartości funkcji straty w zależności od ilości neuronów w pierwszej i drugiej warstwie ukrytej



Rysunek 4: Schemat zastosowanej sieci neuronowej, która składa się z: i) warstwy wejściowej z dwoma neuronami, ii) dwóch warstw ukrytych z odpowiednio trzema i pięcioma neuronami, iii) warstwy wyjściowej z jednym neuronem. Linie zakończone strzałką oznaczają wagę odpowiadającą każdej z par neuronów.

- 4 Algorytm uczący - SGD, parametry, funkcja aktywacji
- 5 Dane wejściowe
- 6 Funkcja straty
- 7 Replikacja danych