

1. Uniwersalny przepływ roboczy uczenia maszynowego

2. Nadmierne i zbyt słabe dopasowanie modeli

Michał Oziebło
ozieblo.michal@icloud.com
github.com/ozieblo/DL

Machine Learning Study Groups
05/11/2019

Opracowane na podst. podręcznika François Chollet w tłum. K. Matuk,
Deep Learning - Praca z językiem Python i biblioteką Keras, wyd. Helion
Tytuł oryginału: *Deep Learning with Python*

Uniwersalny przepływ roboczy uczenia maszynowego

- Definiowanie problemu i przygotowywanie zbioru danych
- Wybór miary sukcesu
- Określanie techniki oceny wydajności modelu
- Przygotowywanie danych
- Tworzenie modeli lepszych od linii bazowej
- Skalowanie w górę
- Regularyzacja modelu i dostrajanie jego hiperparametrów

Definiowanie problemu i stawianie hipotez

- Czym będą dane wejściowe i co chcesz przewidywać? Czy masz dostęp do potrzebnych informacji?
- Model może zauważać tylko prawidłowości, które widział wcześniej
- Problemy niestacjonarne



Wybór miary sukcesu

- Tabela dokładności
- AUC

```
model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=[ 'accuracy' ])
```

Tabela dokładności

	Rzeczywistość +	Rzeczywistość -
Wynik testu +	TP	FP
Wynik testu -	FN	TN

$$\text{czułość} = \frac{TP}{TP + FN}$$

$$\text{swoistość} = \frac{TN}{FP + TN}$$

$$\text{wartość predykcyjna dodatnia} = \frac{TP}{TP + FP}$$

$$\text{wartość predykcyjna ujemna} = \frac{TN}{FN + TN}$$

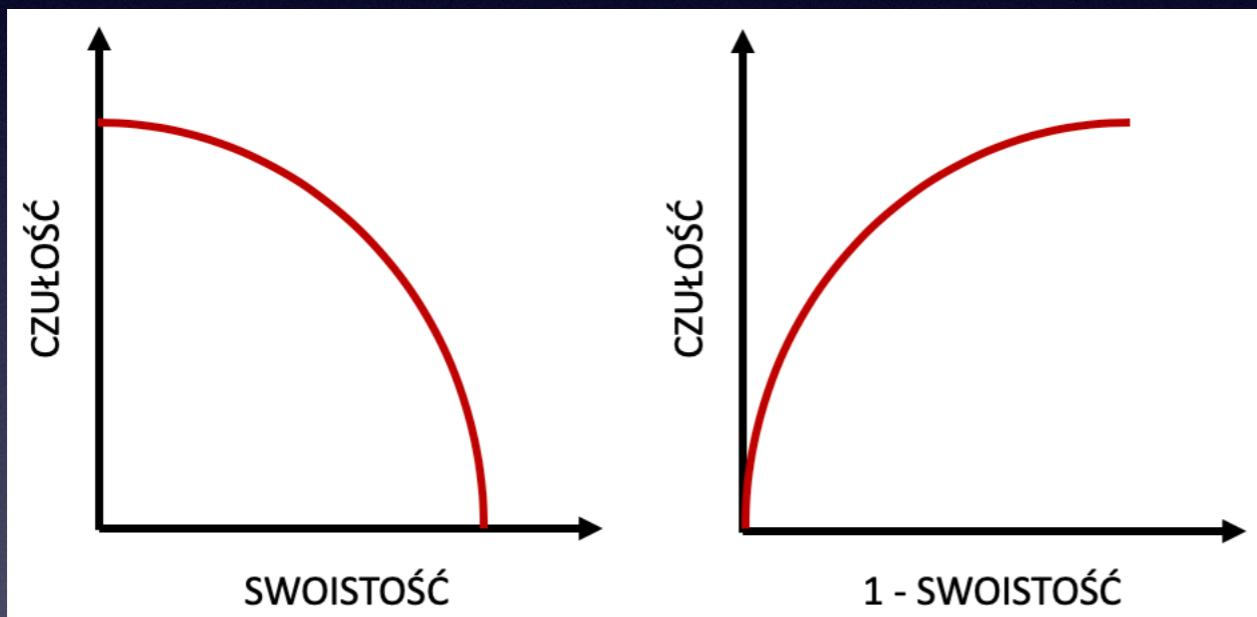
CZUŁOŚĆ

SWOISTOŚĆ

SWOISTOŚĆ

CZUŁOŚĆ

ROC



$$\text{czułość} = \frac{TP}{TP + FN}$$

$$\text{swoistość} = \frac{TN}{FP + TN}$$

$$\text{wartość predykcyjna dodatnia} = \frac{TP}{TP + FP}$$

$$\text{wartość predykcyjna ujemna} = \frac{TN}{FN + TN}$$

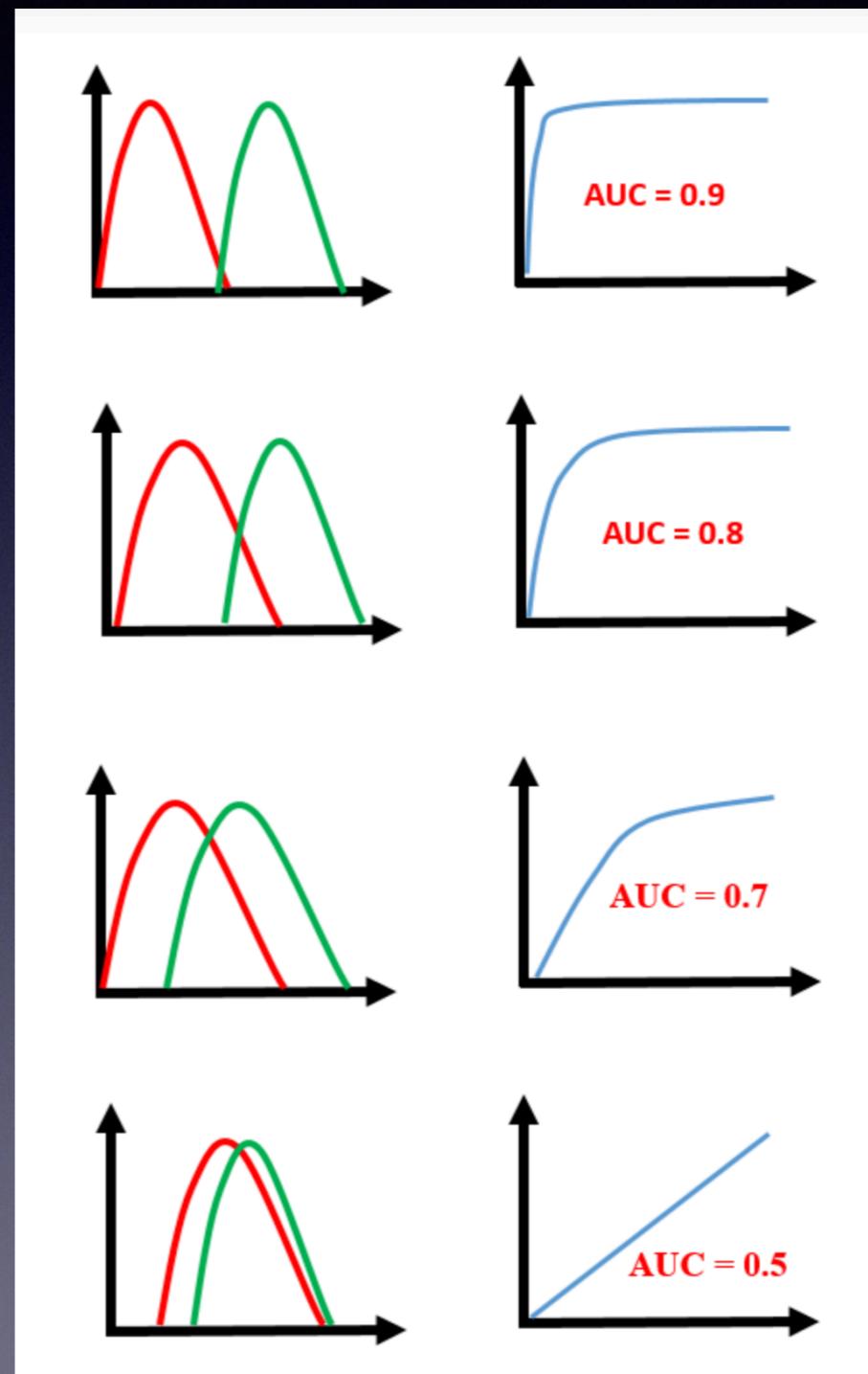
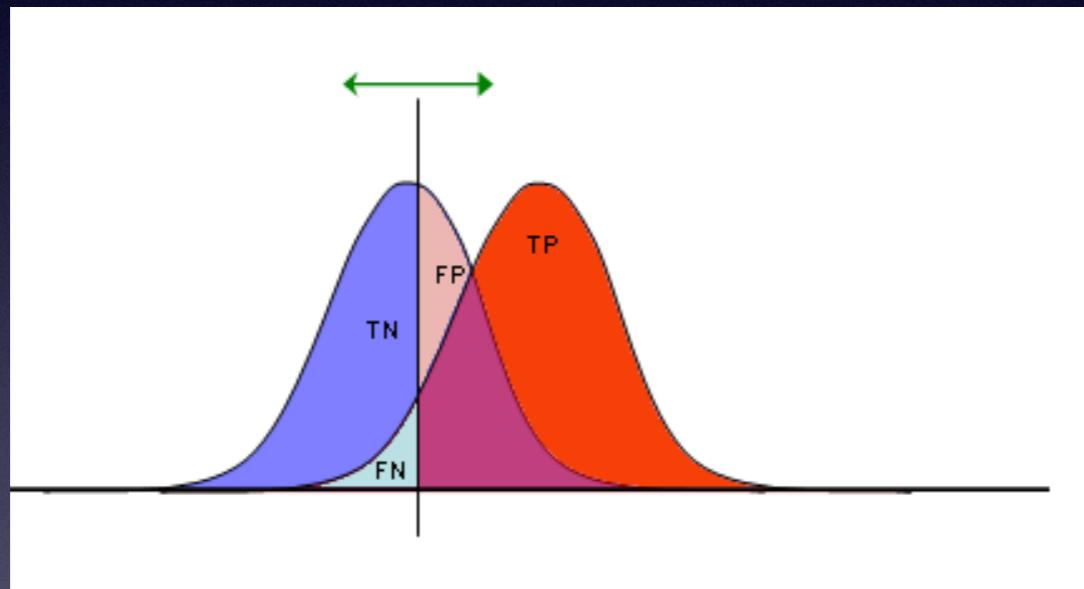
$$\text{Specificity} = \frac{TN}{TN + FP}$$

$$1 - \text{Specificity} = 1 - \frac{TN}{TN + FP}$$

$$1 - \text{Specificity} = \frac{TN + FP - TN}{TN + FP}$$

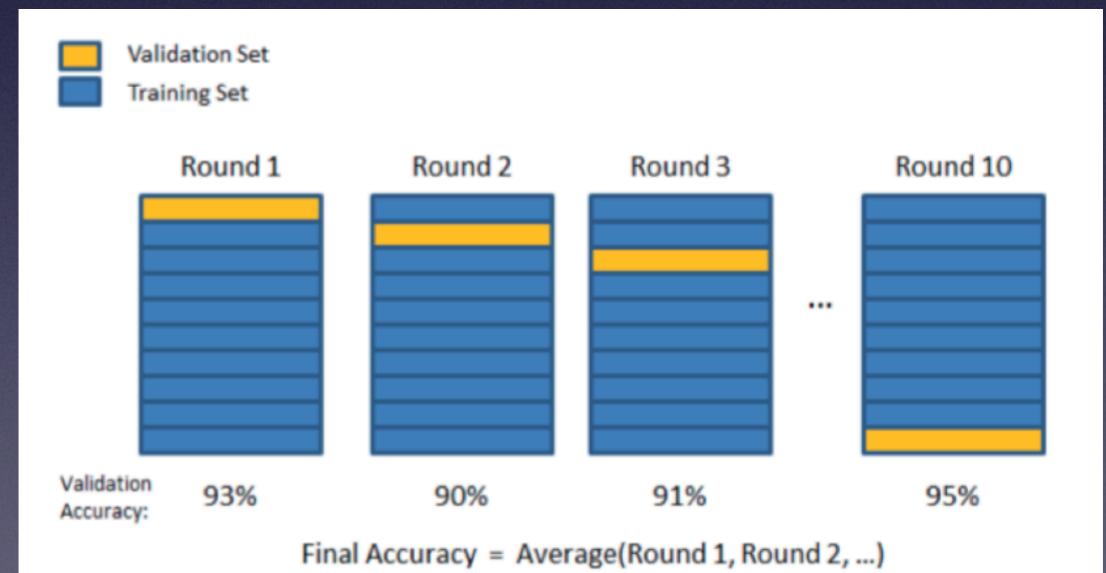
$$1 - \text{Specificity} = \frac{FP}{TN + FP}$$

AUC



Techniki walidacji

- Odkładanie danych w celu utworzenia zbioru walidacyjnego
- K-krotna walidacja krzyżowa
- Iteracyjna walidacja k-krotna



Przygotowanie danych

- Dane powinny mieć formę tensorów
- Wartości umieszczane w tensorach powinny zostać przeskalowane tak, aby przyjmowały małe wartości mieszczące się w zakresie od –1 do 1 lub w zakresie od 0 do 1
- Jeżeli różne cechy przyjmują wartości znajdujące się w różnych zakresach (dane są heterogeniczne), to należy poddać je normalizacji
- Czasami, zwłaszcza w przypadku problemów związanych z małą ilością danych, warto przeprowadzić inżynierię cech

Moc statystyczna modelu

- MNIST: 0.1
- IMDB: 0.5

Moc zależy od:

- Wielkości próby użytej w badaniu
- Rzeczywistej wielkości efektu na tle losowej zmienności w populacji
- Przyjętego poziomu istotności

Funkcja aktywacji **ostatniej** warstwy

- Funkcja aktywacji ostatniej warstwy — funkcja ta narzuca praktyczne ograniczenia na dane generowane przez sieć.
- W przypadku klasyfikacji zbioru danych IMDB w ostatniej warstwie użyliśmy funkcji *sigmoid*, a w przykładzie regresji nie stosowaliśmy żadnej funkcji aktywacji ostatniej warstwy sieci.

Rodzaj problemu	Funkcja aktywacji ostatniej warstwy
Klasyfikacja binarna	sigmoid
Wieloklasowa klasyfikacja jednoetykietowa	softmax
Wieloklasowa klasyfikacja wieloetykietowa	sigmoid
Regresja dowolnych wartości	Brak
Regresja wartości z zakresu od 0 do 1	sigmoid

```
[ ] from tensorflow.python.keras import models
from tensorflow.python.keras import layers
model = models.Sequential()
model.add(layers.Dense(16, activation='relu', input_shape=(5000,)))
model.add(layers.Dense(16, activation='relu'))
model.add(layers.Dense(1, activation='sigmoid'))
```

Funkcja straty

- Funkcja straty — funkcja ta powinna zostać dobrana do natury rozwiązywanego problemu. W przykładzie zbioru danych IMDB używaliśmy funkcji binarnej entropii krzyżowej *binary_crossentropy*, a w przykładzie regresji zastosowaliśmy funkcję *mse*.
- Funkcja straty zdefiniowana jest dla każdego używanego algorytmu i jest to główna miara oceny dokładności wyszkolonego modelu.

Tabela 4.1. Wybór odpowiedniej funkcji aktywacji ostatniej warstwy i funkcji straty modelu

Rodzaj problemu	Funkcja aktywacji ostatniej warstwy	Funkcja straty
Klasyfikacja binarna	sigmoid	<i>binary_crossentropy</i>
Wieloklasowa klasyfikacja jednoetykietowa	softmax	<i>categorical_crossentropy</i>
Wieloklasowa klasyfikacja wieloetykietowa	sigmoid	<i>binary_crossentropy</i>
Regresja dowolnych wartości	Brak	<i>mse</i>
Regresja wartości z zakresu od 0 do 1	sigmoid	<i>mse lub binary_crossentropy</i>

```
model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=[ 'accuracy' ])
```

Konfiguracja optymalizacji

- Jakiego optymalizatora będziesz korzystać?
- Jaki współczynnik uczenia wybierzesz
- Optymalizator - mechanizm dostrajania sieci na podstawie danych zwracanych przez funkcję straty
- W większości sytuacji bezpiecznym wyborem jest optymalizator *rmsprop* i domyślna wartość jego współczynnika uczenia

```
model.compile(optimizer='rmsprop', loss='binary_crossentropy', metrics=[ 'accuracy' ])
```

An optimizer is one of the two arguments required for compiling a Keras model:

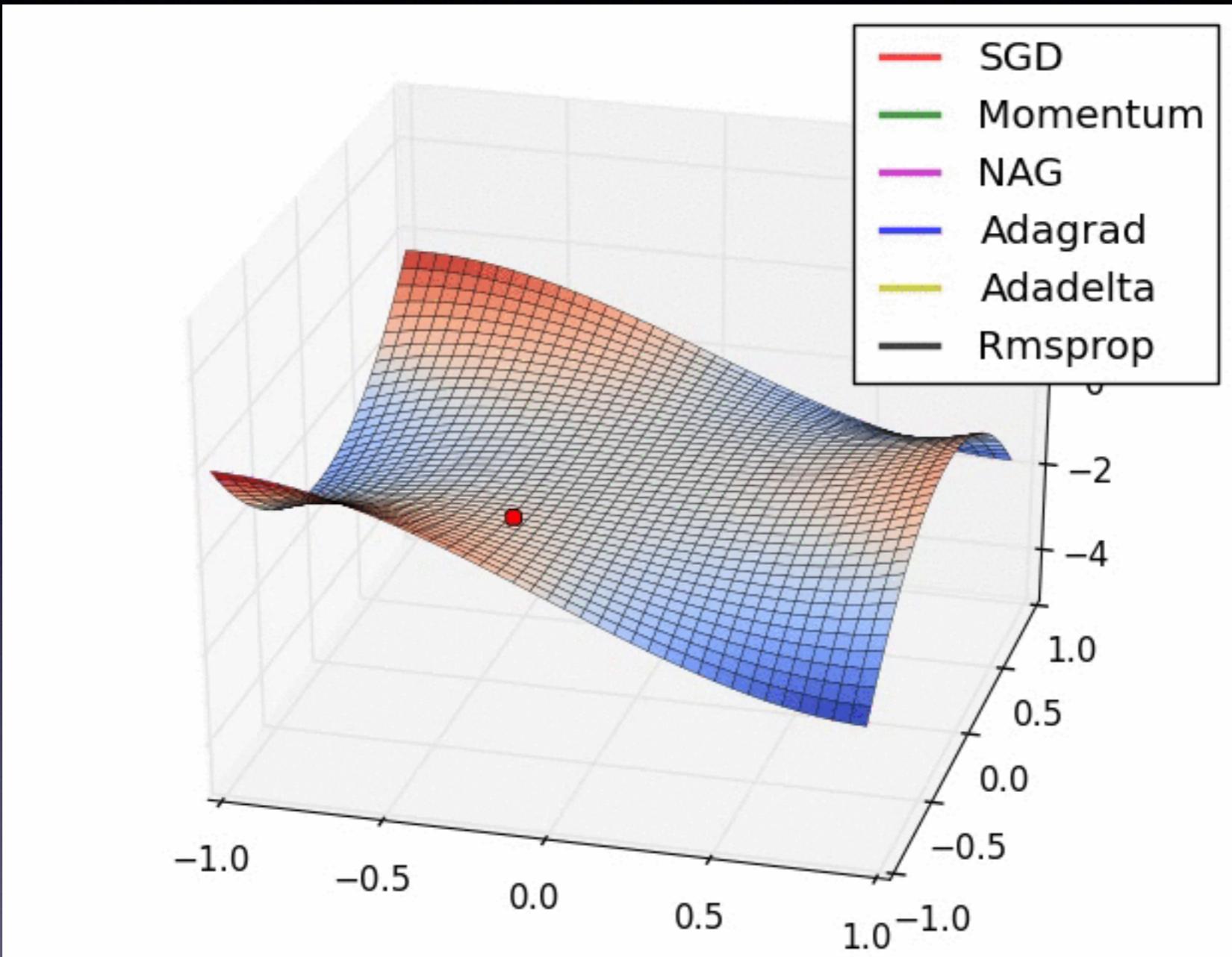
```
from keras import optimizers

model = Sequential()
model.add(Dense(64, kernel_initializer='uniform', input_shape=(10,)))
model.add(Activation('softmax'))

sgd = optimizers.SGD(lr=0.01, decay=1e-6, momentum=0.9, nesterov=True)
model.compile(loss='mean_squared_error', optimizer=sgd)
```

You can either instantiate an optimizer before passing it to `model.compile()`, as in the above example, or you can call it by its name. In the latter case, the default parameters for the optimizer will be used.

```
# pass optimizer by name: default parameters will be used
model.compile(loss='mean_squared_error', optimizer='sgd')
```



<https://imgur.com/a/Hqolp#NKsFHJb>

Skalowanie w górę

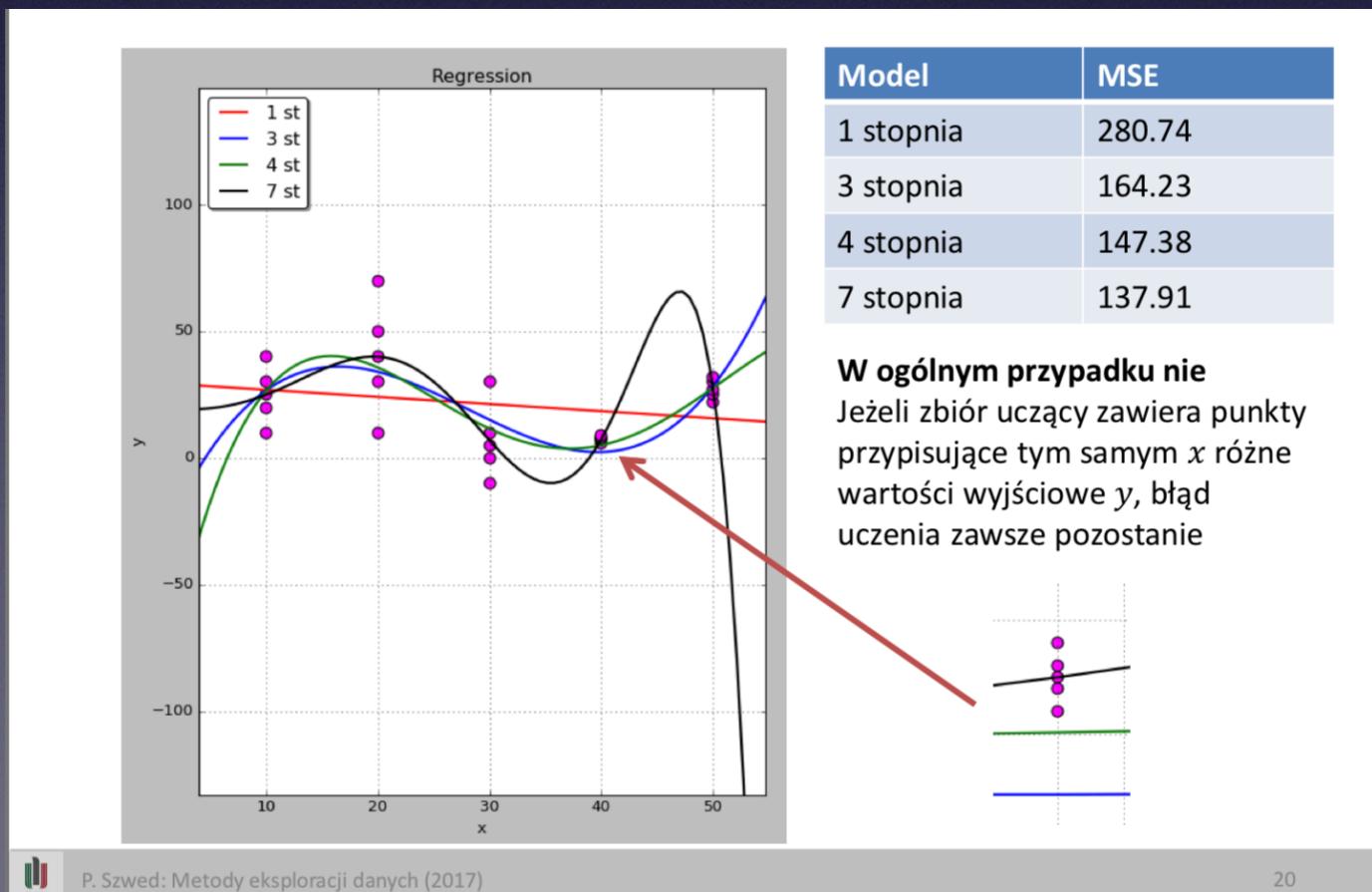
- więcej warstw
- większy rozmiar warstw
- więcej epok trenowania

*Podstawowym problemem uczenia
maszynowego jest kompromis pomiędzy
optymalizacją, a uogólnianiem*

Nadmierne i zbyt słabe dopasowanie modeli

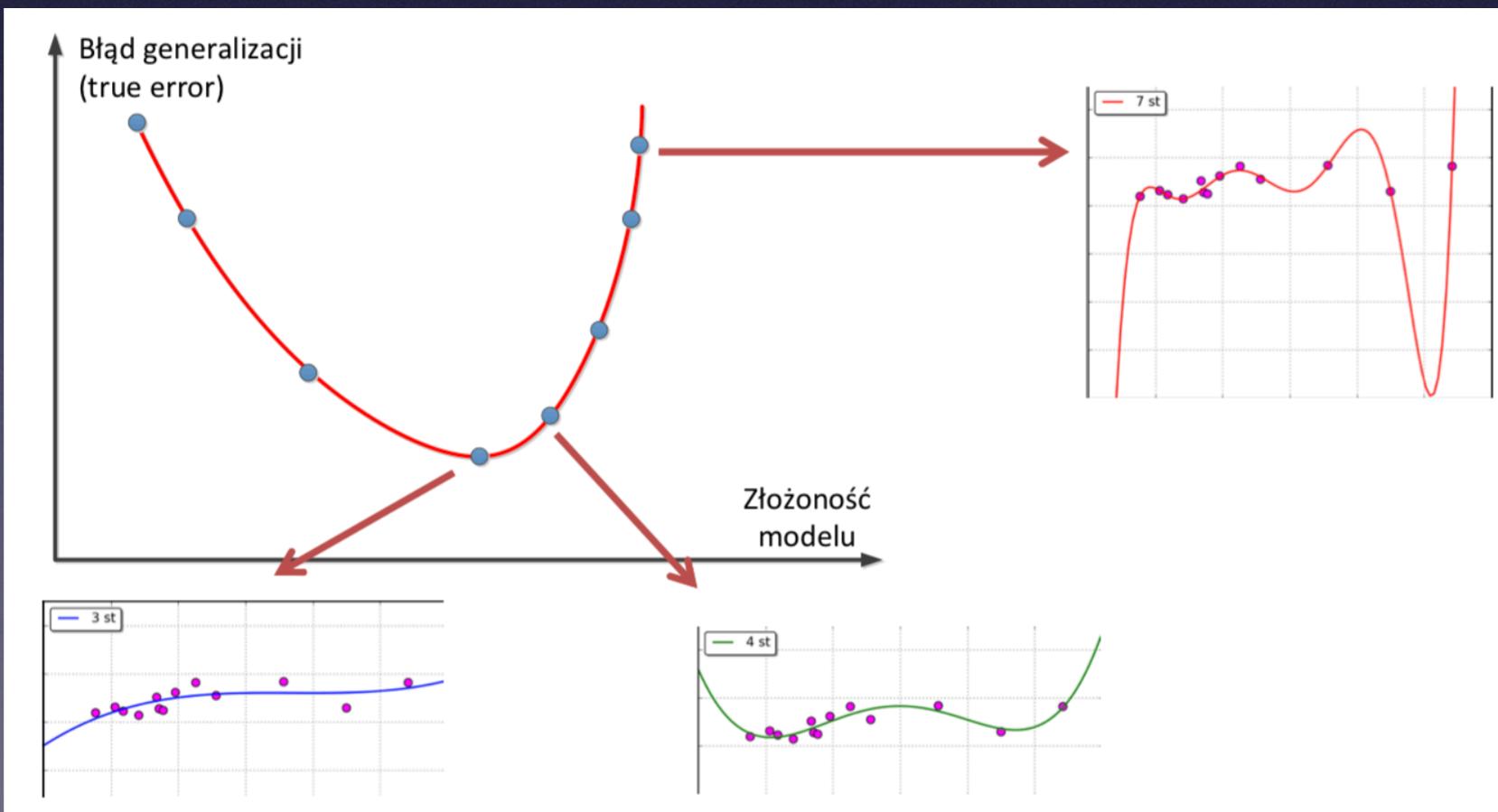
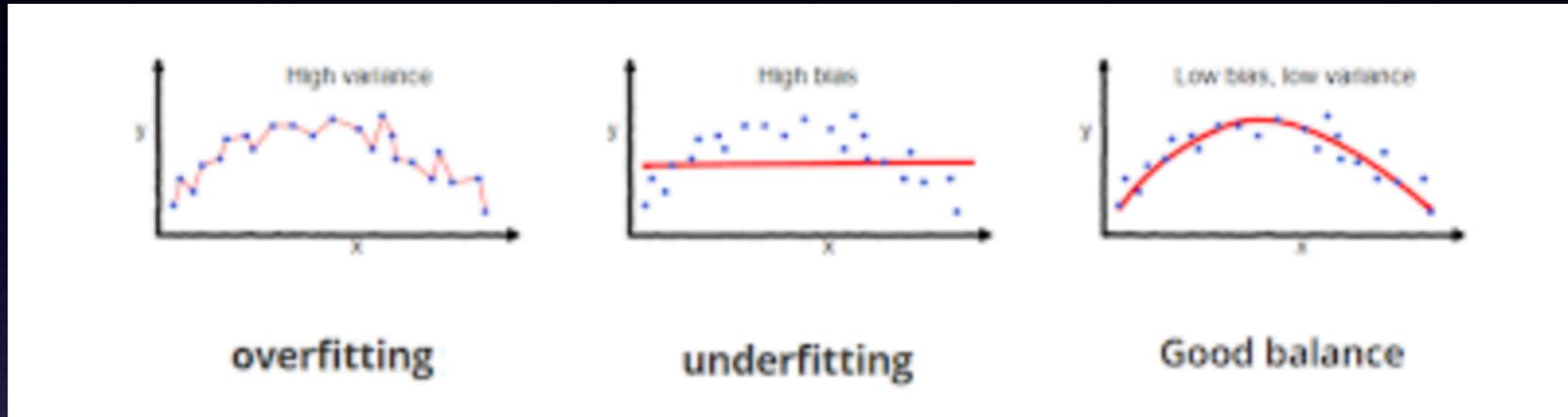
Czym jest optymalizacja?

- Proces dostrajania modelu w celu uzyskania najlepszej możliwej wydajności na danych **treningowych**
- Czy błąd uczenia można zredukować do zera?
- Czy błąd uczenia pozwala ocenić jakość modelu podczas predykcji?

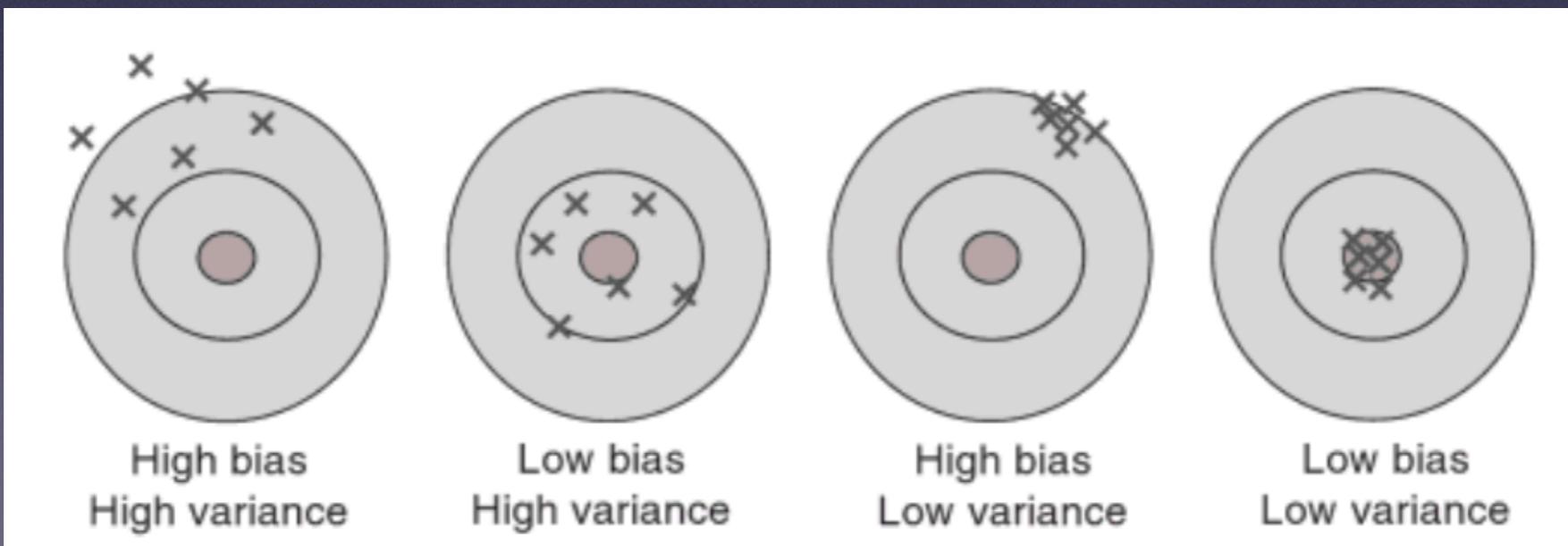
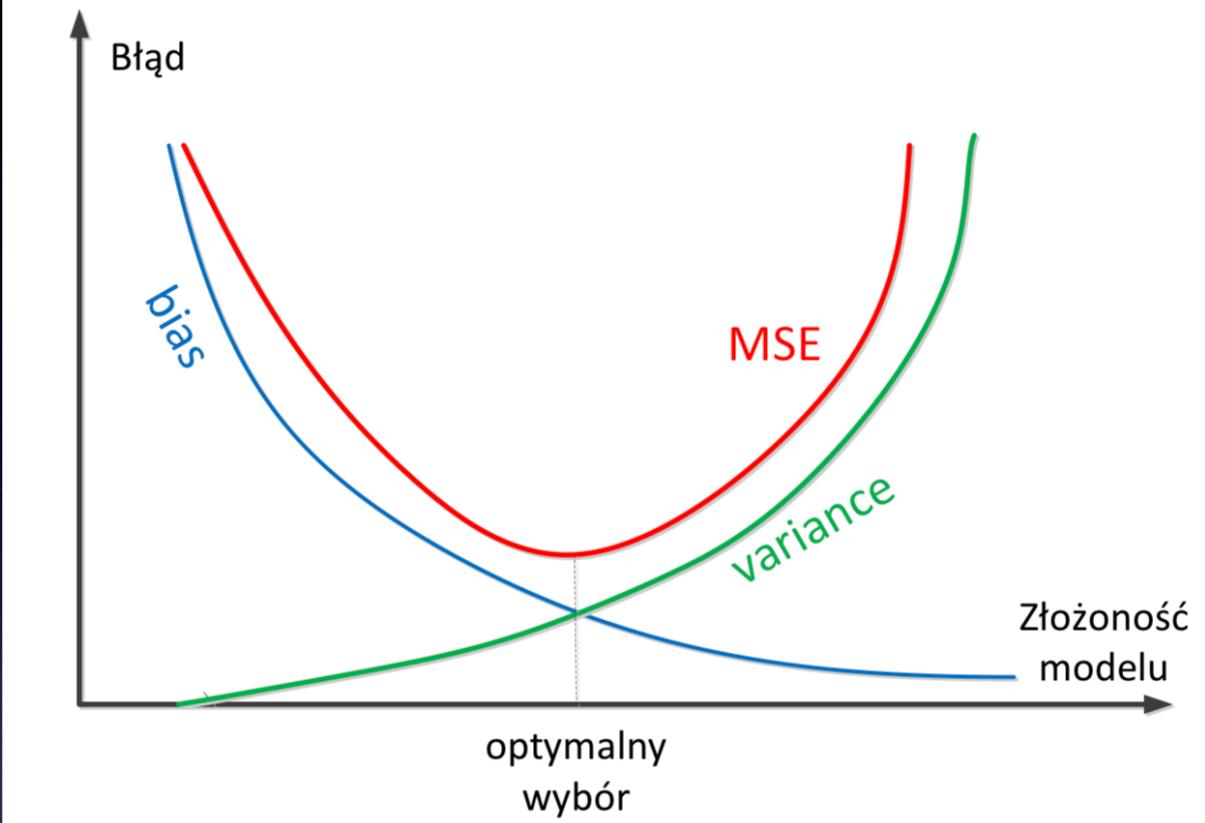


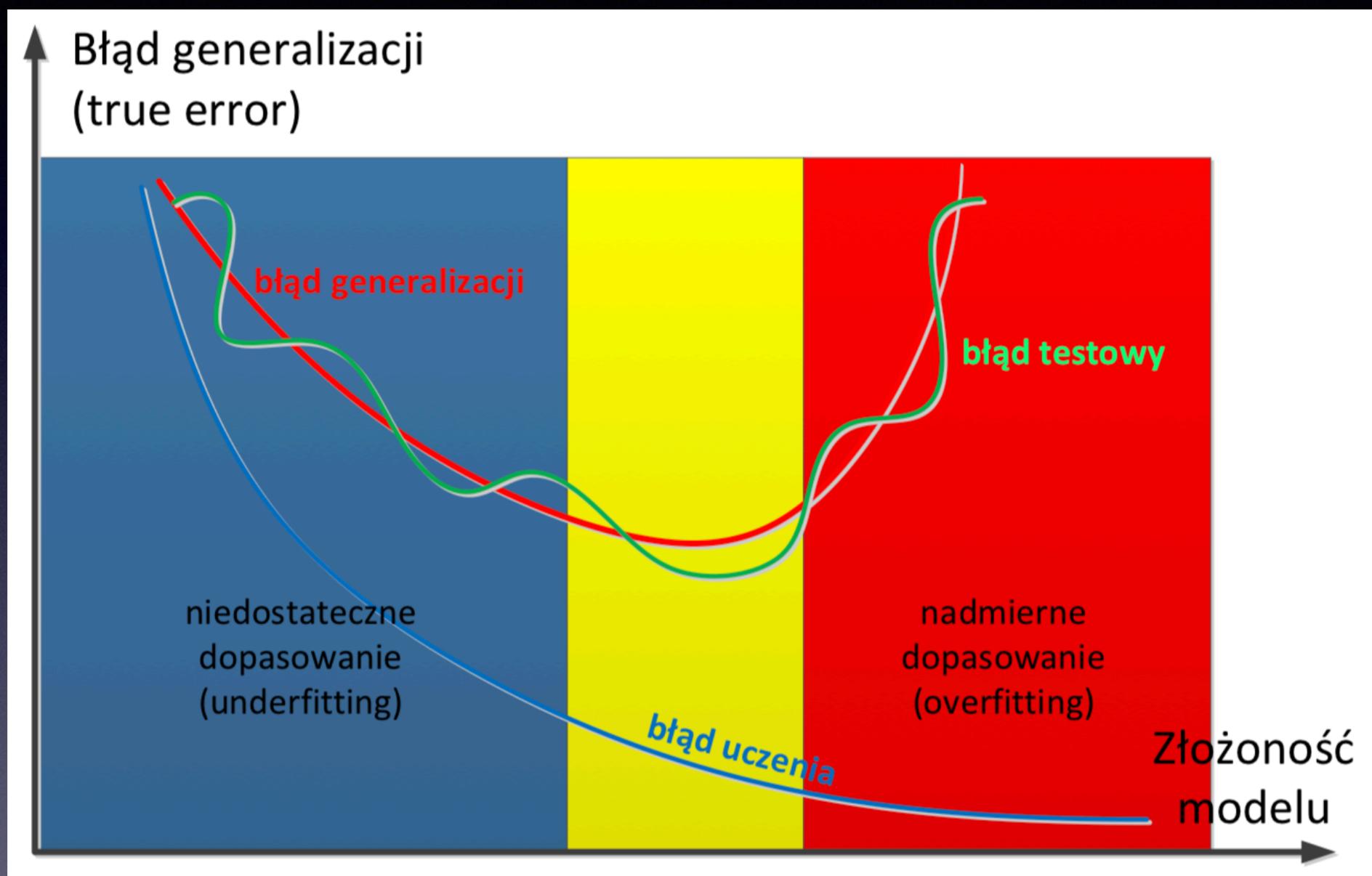
Nadmierne i zbyt słabe dopasowanie modeli

Kiedy model jest zbyt słabo lub nadmiernie dopasowany?



Kompromis pomiędzy bias i variance



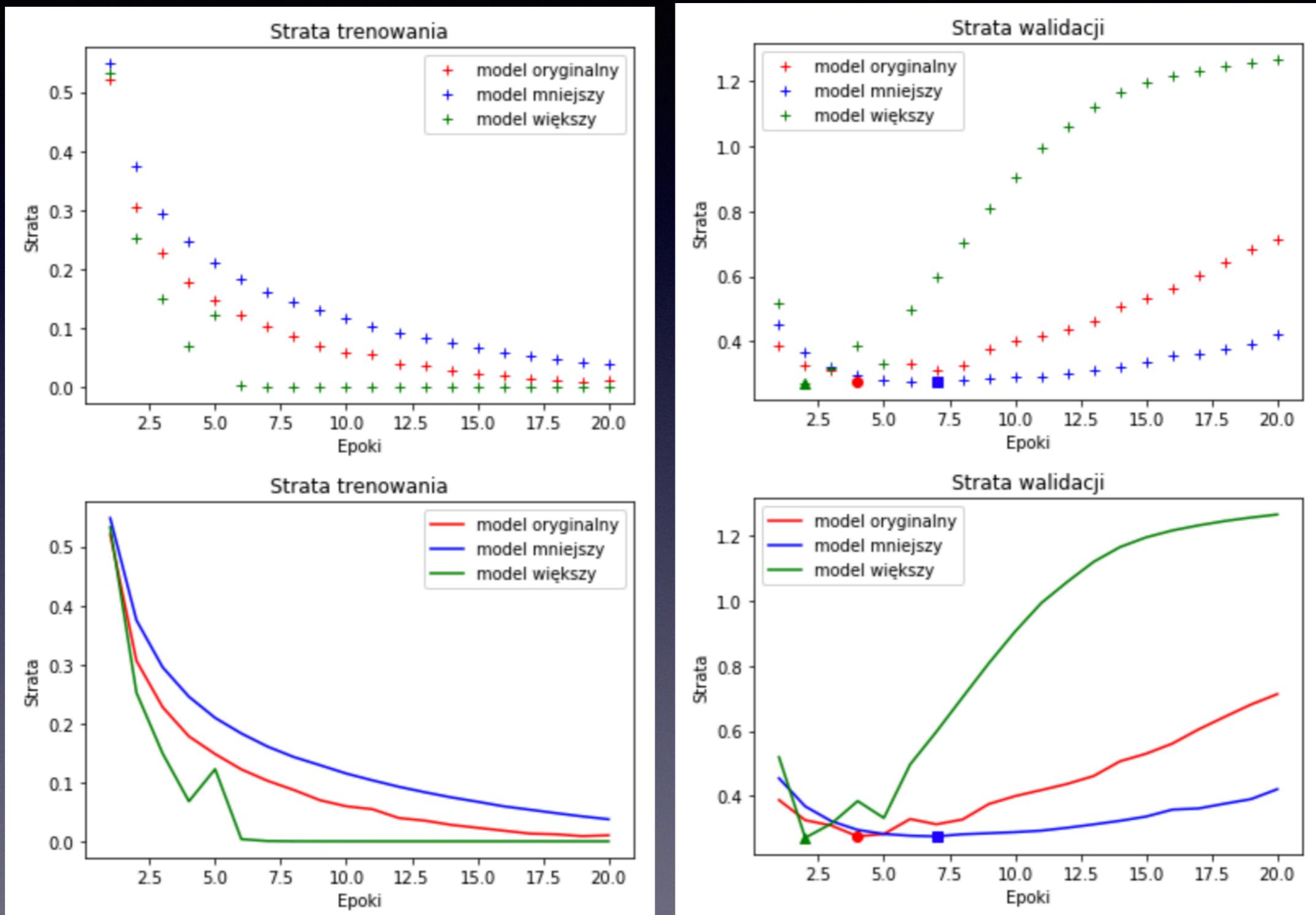


Essentially, all models are wrong, but some are useful.
George Box

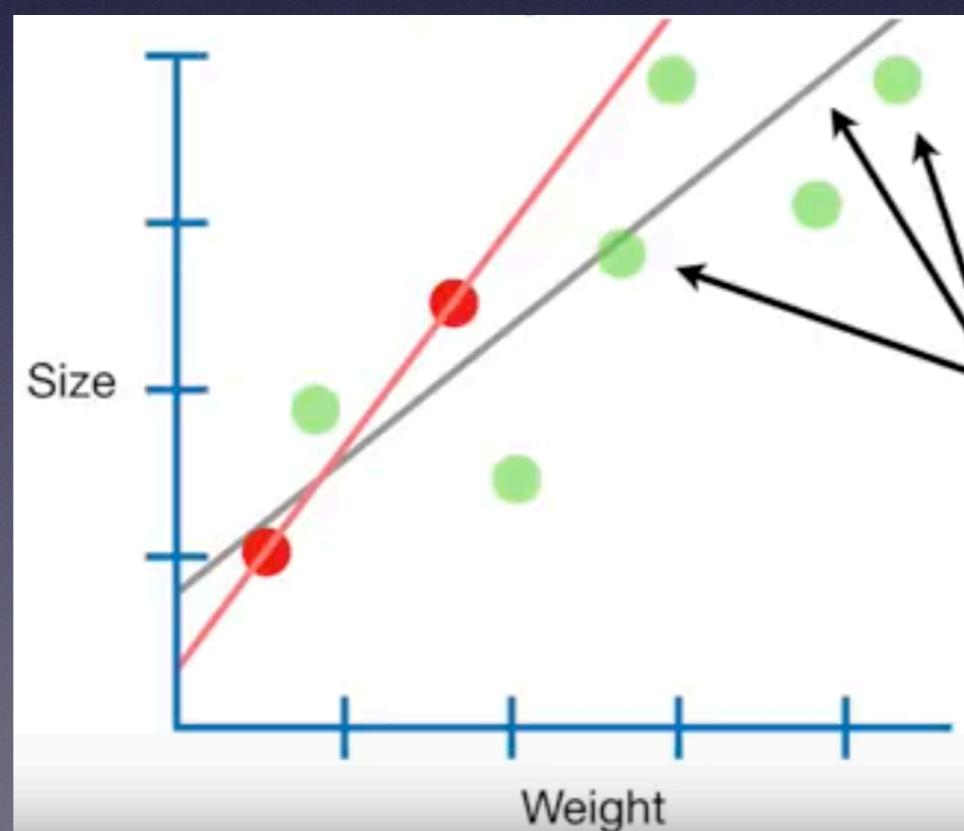
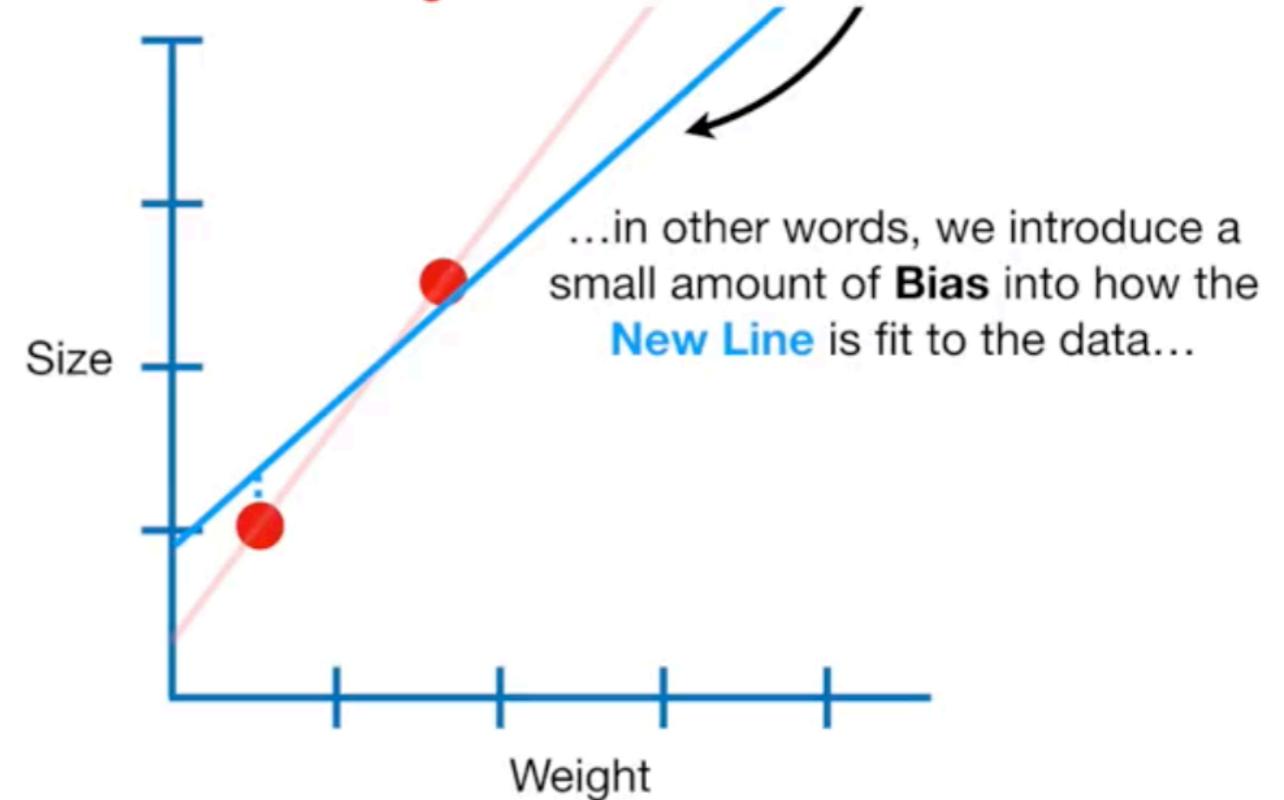
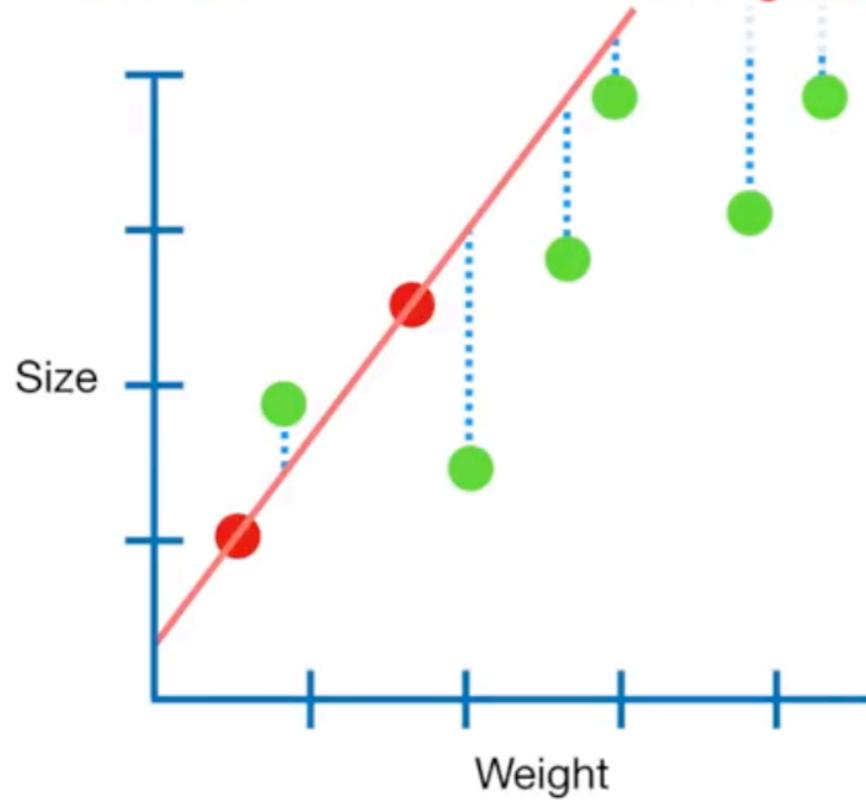
Jak zapobiec uczeniu się przez model błędnych lub zbędnych wzorców treningowego zbioru danych?

- Redukcja rozmiaru sieci
- Dodawanie regularyzacji wag
- Technika dropout

Redukcja rozmiaru sieci



Nadmierne i zbyt słabe dopasowanie modeli



Ridge

- regularyzacja L2 — koszt jest dodawany proporcjonalnie do kwadratu wartości współczynników wag (normy L2 wag)
- w kontekście sieci neuronowych regularyzacja L2 jest również określana mianem rozkładu wag
- pomimo innej nazwy jest to ten sam proces, który w matematyce określamy jako regularyzacja L2

Lasso

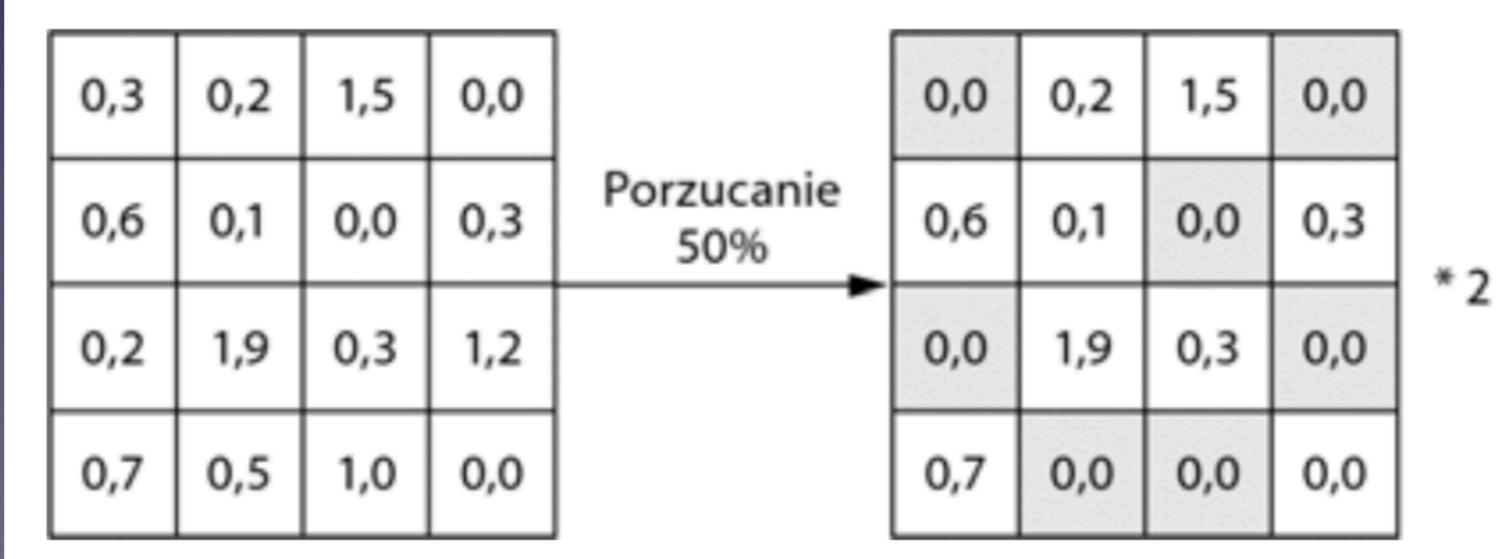
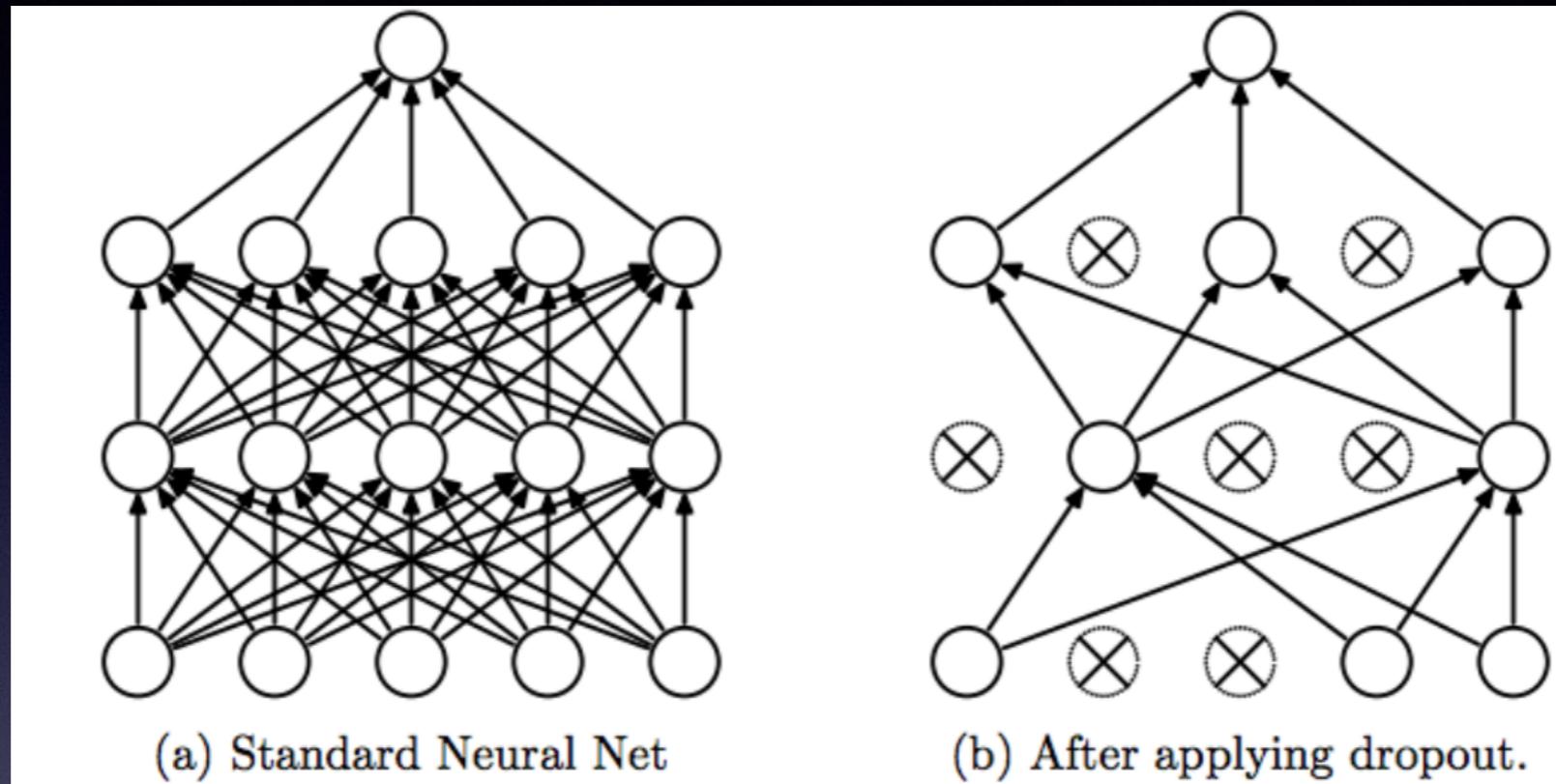
- *least absolute shrinkage and selection operator*
- regularyzacja L1 — koszt jest dodawany proporcjonalnie do bezwzględnej wartości współczynników wag (normy L1 wag)
- działa więc jak mechanizm wyboru cech (ang. *feature selection*)
- stopniowo odrzuca współliniowe atrybuty, pozostawia zbiór najbardziej istotnych

Sieci elastyczne

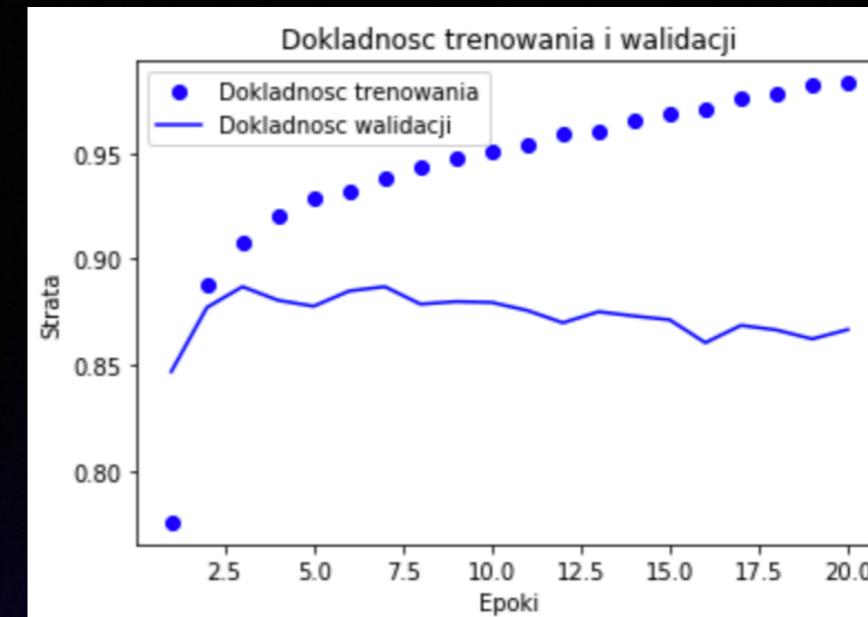
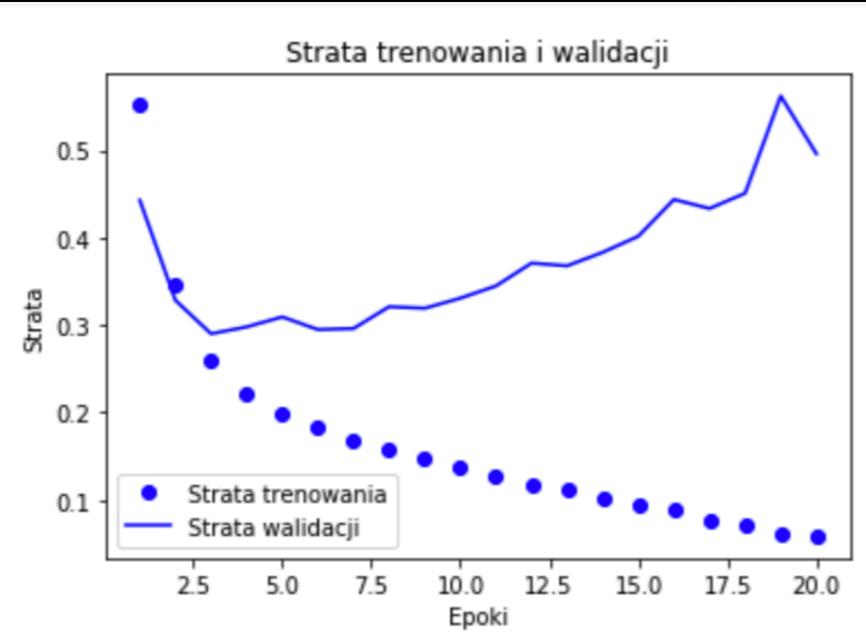
- Hybrydowa sieć elastyczna dobrze radzi sobie w sytuacji, kiedy występują korelacje między parametrami

```
regularizers.l1_l2(l1=0.001, l2=0.001) ← Jednoczesna regularyzacja L1 i L2.
```

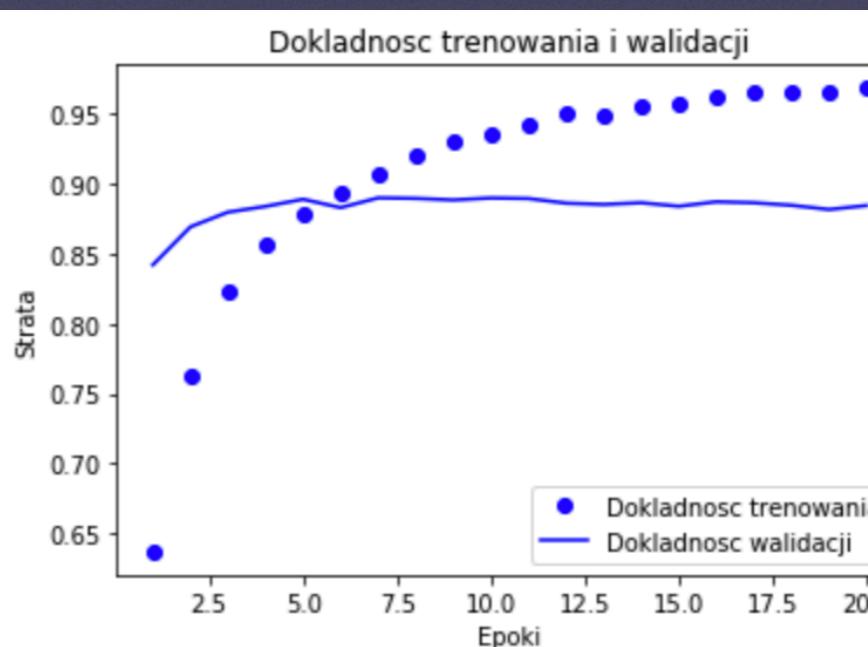
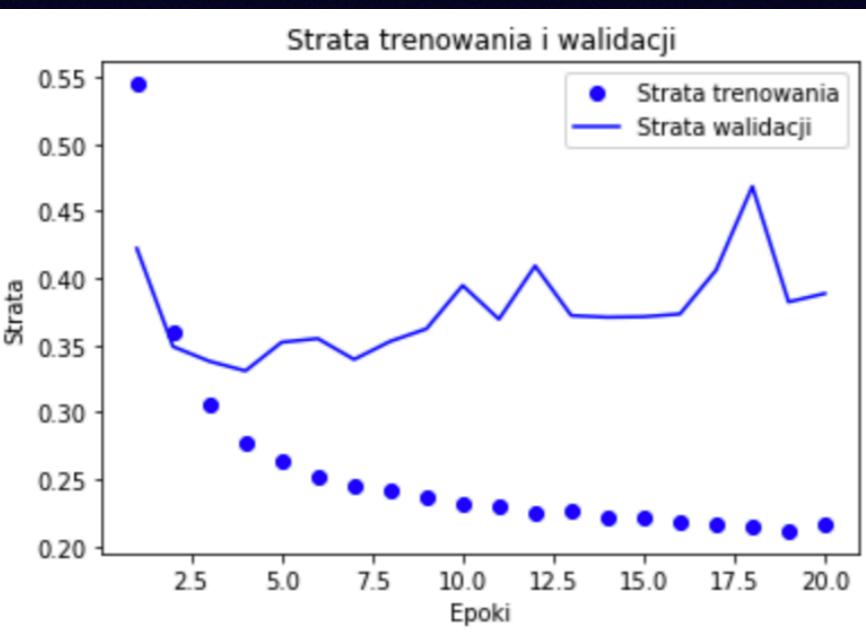
Technika dropout



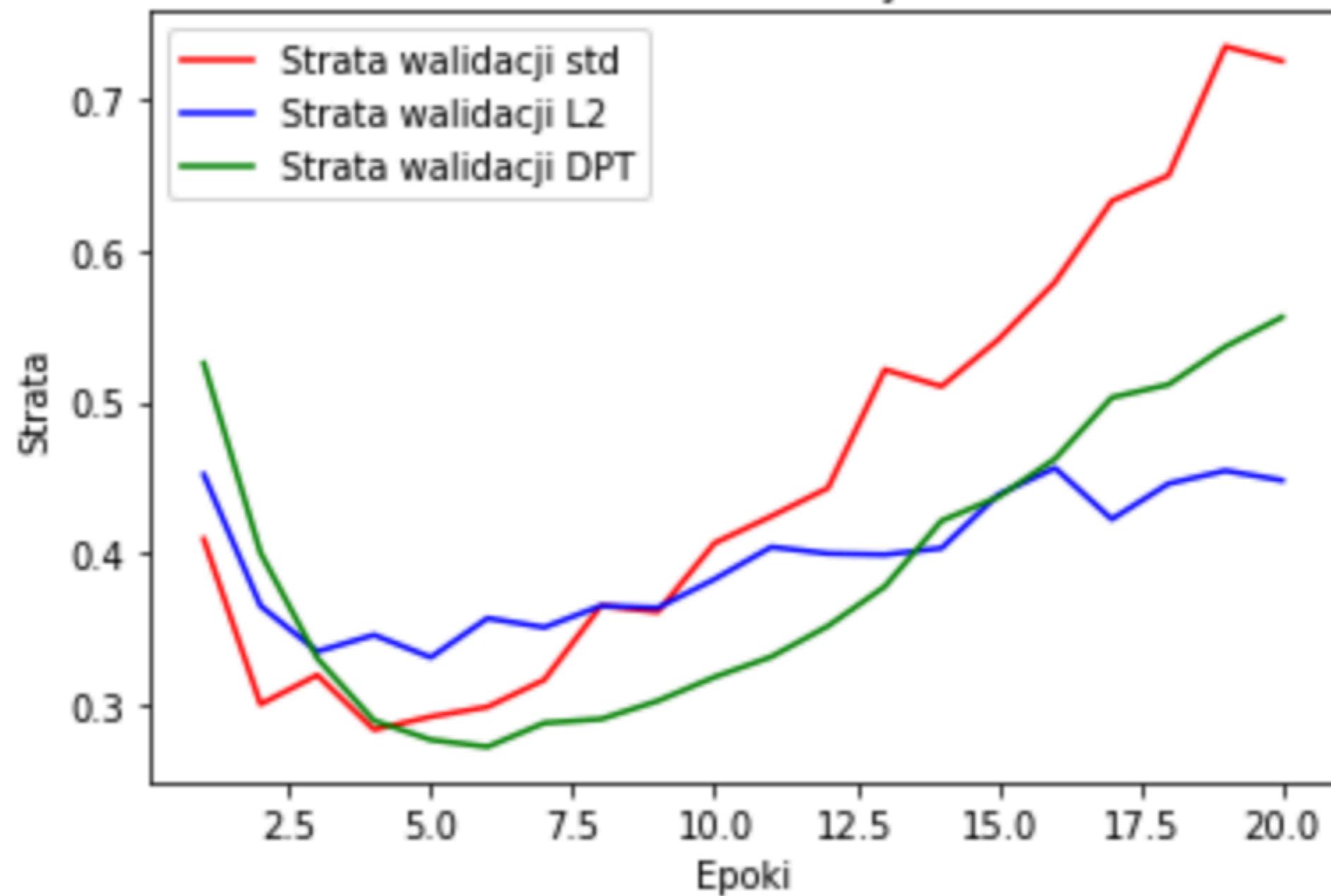
L2:



DPT:



Strata walidacji



Regularyzacja i dostrajanie modelu

- zastosowanie różnych architektur poprzez dodawanie lub usuwanie warstw
- definiowanie różnych hiperparametrów modelu, takich jak liczba jednostek w poszczególnych warstwach i współczynnik uczenia się optymalizatora
- dodanie regularyzacji L1 lub/i L2
- mechanizm porzucania
- przeprowadzenie dodatkowej inżynierii cech: dodanie nowych cech, usunięcie cech, które wydają się nie zawierać przydatnych informacji

Podsumowanie

Podsumowanie rozdziału

- Zdefiniuj problem, nad którym pracujesz, i dane, na których będziesz trenować swój model. Zbierz niezbędne dane i utwórz etykiety obserwacji (o ile są potrzebne).
- Określ sposób mierzenia sukcesu. Która metryka powinna być monitorowana w procesie walidacji?
- Wybierz technikę walidacji. Możesz korzystać ze zwykłej walidacji na odłożonym fragmencie zbioru danych treningowych. Czy też lepiej jest skorzystać z metody walidacji krzyżowej? Która część danych powinna wejść w skład walidacyjnego zbioru danych?
- Opracuj swój pierwszy model, który będzie sprawdzał się lepiej od linii bazowej (model mający moc statystyczną).
- Opracuj model nadmiernie dopasowujący się do danych.
- Przeprowadź regularyzację modelu i dostrajaj jego parametry na podstawie wydajności przetwarzania walidacyjnego zbioru danych. Wiele osób zajmujących się uczeniem maszynowym skupia się głównie na tym etapie pracy nad modelem, ale musisz pamiętać również o pozostałych etapach opracowywania modelu.

Źródła

- <https://towardsdatascience.com/batch-normalization-in-neural-networks-1ac91516821c>
- <http://adam.mchtr.pw.edu.pl/~sztyber/ino/w3.pdf>
- <http://pbiecek.github.io/Przewodnik/Predykcja/regularyzacja.html>
- <http://enroute.pl/regularyzacja-ridge-lasso-elastic-net/>
- <http://home.agh.edu.pl/~pszwed/wiki/lib/exe/fetch.php?media=med-w03.pdf>
- <https://statquest.org/2018/09/24/regularization-part-1-ridge-regression/>
- <https://www.youtube.com/watch?v=NGf0voTMIcs>
- <https://www.youtube.com/watch?v=1dKRdX9bflo>
- https://media.statsoft.pl/_old_dnn/downloads/krzywe_roc_czyli_ocena_jakosci.pdf
- <https://www.youtube.com/watch?v=Os6gFKY5RbY>
- <https://medium.com/greyatom/lets-learn-about-auc-roc-curve-4a94b4d88152>
- <https://dataminingalapolonaise.wordpress.com/2013/03/17/pare-uzytecznych-rzeczy-na-temat-eksploracji-danych/>
-