

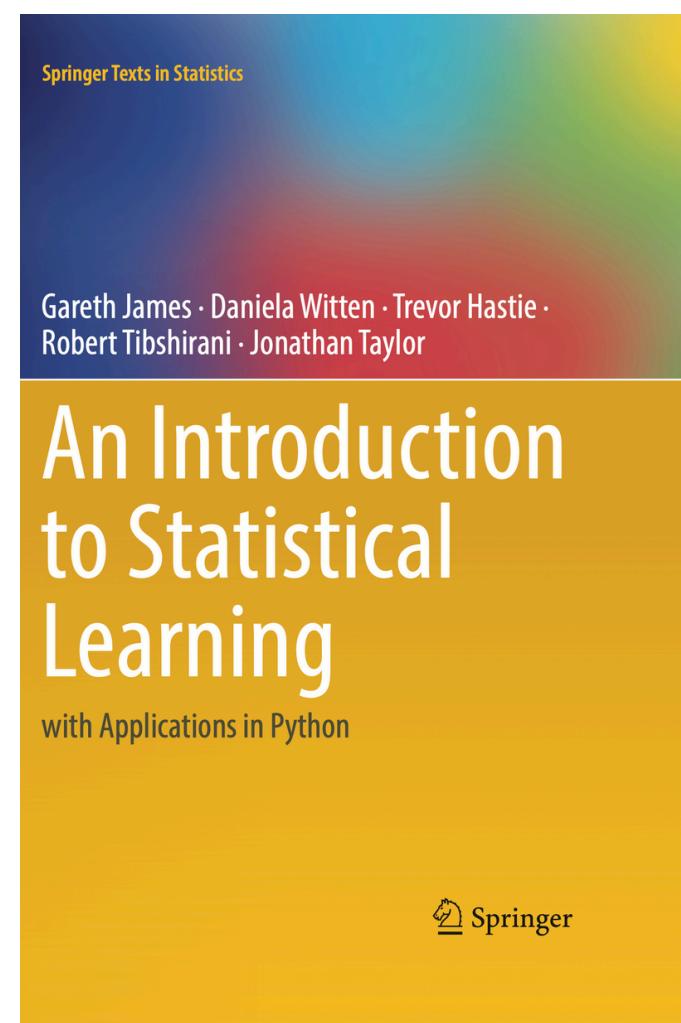
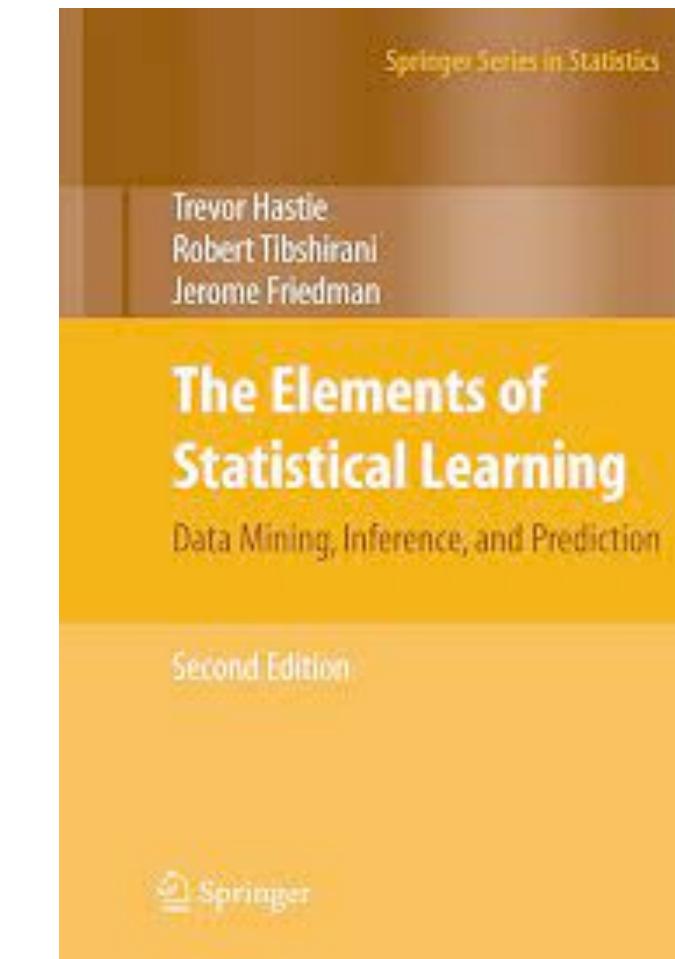
Podstawy uczenia maszynowego

Część 1

Dr Mateusz Radzimski

Materiały pomocnicze dla chętnych

- „Linear regression with Andrew Ng” - seria nagrani na YouTube prowadzona przez Andrew Ng
- Książki:
 - The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (wersja elektroniczna dostępna za darmo: <https://hastie.su.domains/ElemStatLearn/>)
 - An Introduction to Statistical Learning (with Applications in Python) — nowa książka z 2023 roku, również do pobrania za darmo: <https://www.statlearning.com/>
 - Effective Pandas: Patterns for Data Manipulation



Czego się nauczyszmy?

- Kilku podstawowych algorytmów ML
- Sposobów przygotowania danych, trenowania, ewaluacji uczenia maszynowego
- Poznamy trochę teorii i intuicji. Niektóre detale techniczne pominiemy (ale zainteresowanych zachęcam do zadawania pytań!).
- Poznamy jak zaimplementować takie modele w Pythonie przy użyciu bibliotek:
 - Pandas, numpy, sci-kit learn, matplotlib
 - Poznamy również kilka „klasycznych” zbiorów danych używanych w uczeniu maszynowym

Uczenie maszynowe (ang. Machine Learning, ML)

- Dziedzina Sztucznej Inteligencji (AI)
- Wykorzystuje metody statystyczne, algorytmy, sieci neuronowe itd.
- Zdolność uczenia się, ale bez konieczności posiadania bezpośrednich instrukcji
- Uczenie się odbywa na podstawie danych wejściowych: poprzez odnajdywanie wzorców lub generalizację przypadków

ML i inne *buzzwordy*

Sztuczna Inteligencja (AI) - cecha maszyn polegająca na postrzeganiu, syntezie, wnioskowaniu (np. rozpoznawanie obrazów, rozpoznawanie i rozumienie języka mówionego, synteza mowy, wykonywanie czynności tradycyjnie itd.)

Machine Learning - uczenie maszynowe (patrz poprzedni slajd)

Deep Learning - uczenie maszynowe wykorzystujące tzw. głębokie sieci neuronowe (architektury sieci neuronowych wykorzystujące więcej niż jedną warstwę ukrytą, np. RNN, CNN, GAN, Transformer, GNN, itd.)

Machine Learning Engineer (rola) - osoba potrafiąca wykorzystać model uczenia maszynowego (stworzonego przez Data Scientistą) w procesie biznesowym. Inżynier posiadający doświadczenie zarówno w dziedzinie inżynierii oprogramowania jak również w dziedzinie uczenia maszynowego

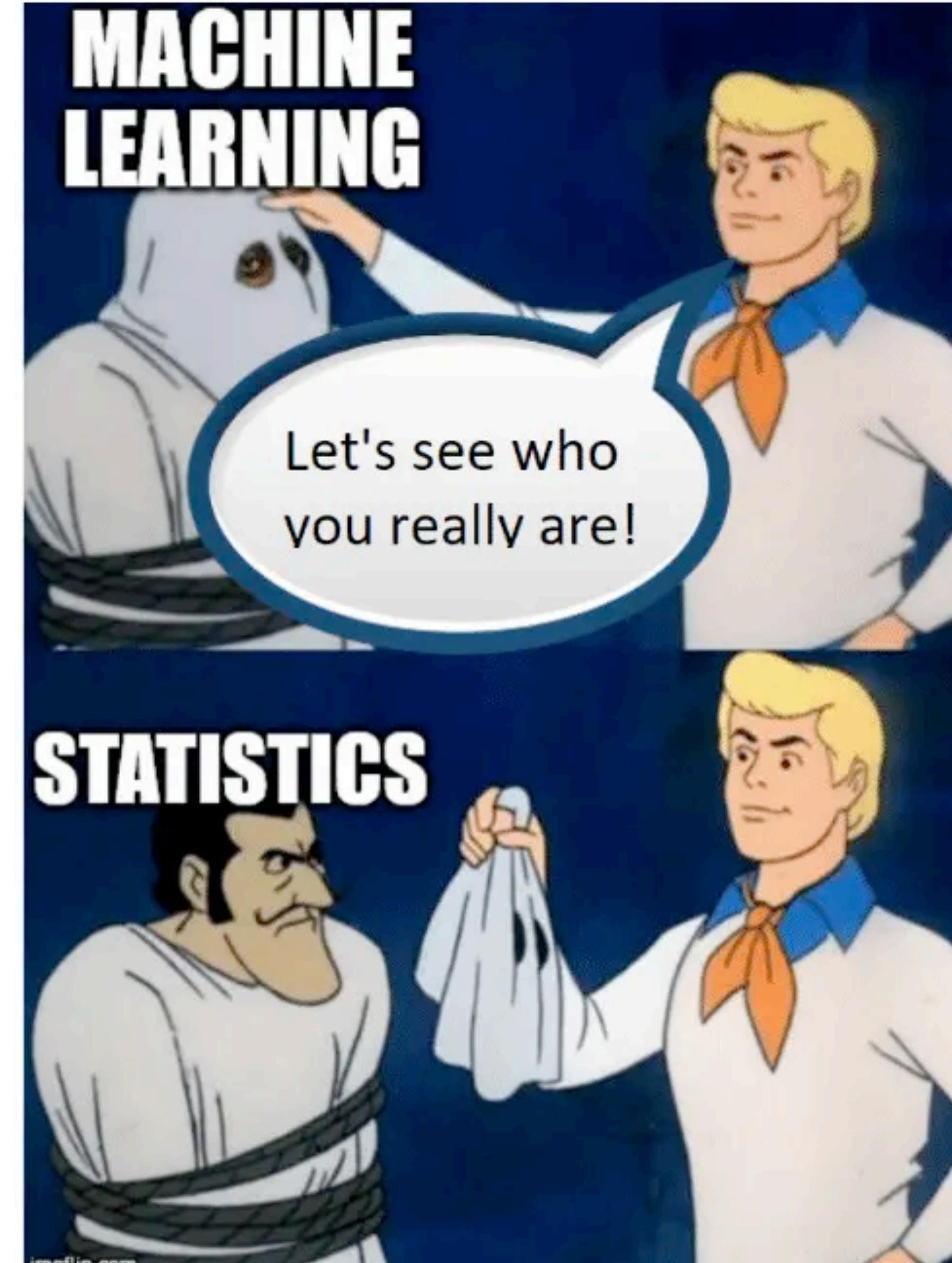
Data Science (dziedzina) / Data Scientist (rola) - dziedzina wykorzystująca metody statystyczne, uczenie maszynowe, eksplorację i analizę danych, techniki wizualizacji w celu zrozumienia danych dla potrzeb biznesowych. Data Scientist to osoba wykonująca pracę w tej dziedzinie, tworząca modele ML.

MLOps - zbiór praktyk i technologii związanych z wdrażaniem i utrzymaniem modeli uczenia maszynowego w środowiskach produkcyjnych

LLM - Large Language Model - Duży model językowy - model sztucznej inteligencji wyspecjalizowany w realizacji zadań związanych z przetwarzaniem języka naturalnego, np. generowanie tekstu, konwersacja, odpowiadanie na pytania itd. Modele te trenowane są na olbrzymich korpusach danych w sposób „samonadzorowany” (self-supervised learning).

GPT - Generative Pre-trained Transformers - przełomowy i bardzo popularny rodzaj modeli LLM opartych o architekturę sieci neuronowych zwaną „transformatorami” (ang. transformer). Zaproponowany przez OpenAI już w 2018, ale największą popularność zdobył w 2022 jako ChatGPT (modele GPT-3.5).

ML i inne *buzzwordy*



Dlaczego warto?

W ostatnich latach jesteśmy świadkami wielu przełomów z wykorzystaniem technologii uczenia maszynowego:

- Lepsze systemy rozpoznawania mowy (głębokie sieci neuronowe) —> rozwój wirtualnych asystentów, lepsze komendy głosowe w smartfonach
- Rozpoznawanie obrazów w czasie rzeczywistym —> Prototypy autonomicznych samochodów
- Algorytmy generatywne, „AI art”: DALL·E, Midjourney, Stable Diffusion
- LLM (duże modele językowe, ang. Large Language Models) —> ChatGPT, Llama, Copilot, Claude, Deepseek, PLLuM, itd...

Technologie te mają coraz większe znaczenie komercyjne i często umożliwiają zdobycie przewagi konkurencyjnej.

Klasyfikacja uczenia maszynowego

Uczenie maszynowe - rodzaje

- Uczenie nadzorowane (supervised learning)
- Uczenie nienadzorowane (unsupervised learning)
- Uczenie przez wzmacnianie (reinforcement learning)

W praktyce, rodzaje te stosujemy w zależności od problemu.

Istnieje jeszcze kilka innych rodzajów, które są pochodnymi powyższych, np. semi-supervised learning, self-supervised learning itd.

Uczenie nadzorowane (supervised)

Cechy:

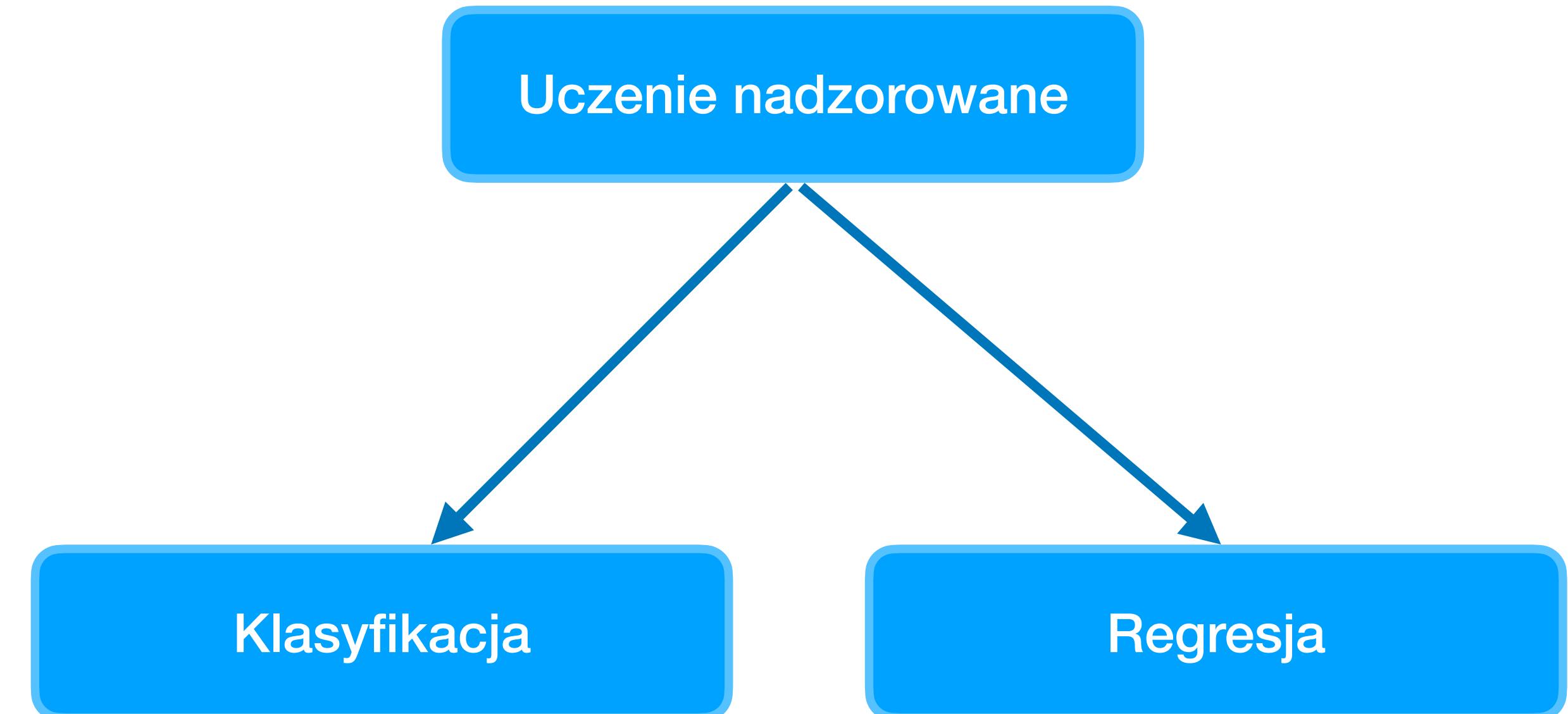
- Posiadamy zbiór uczący (training data), który zawiera dane wejściowe wraz z pożądaną odpowiedzią / etykietą (labels; labeled data).
- Celem jest przewidywanie odpowiedzi dla nowych danych wejściowych.

Rezultatem uczenia jest „model”, który dokonuje predykcji (prediction, inference).

Uczenie nadzorowane (supervised)

Predykcją może być np.:

- Klasyfikacja: zaklasyfikowanie nowej obserwacji do jednej z kategorii dyskretnych (classification task)
- Regresja: uzyskanie wartości liczbowej (ciągłej) (regression)

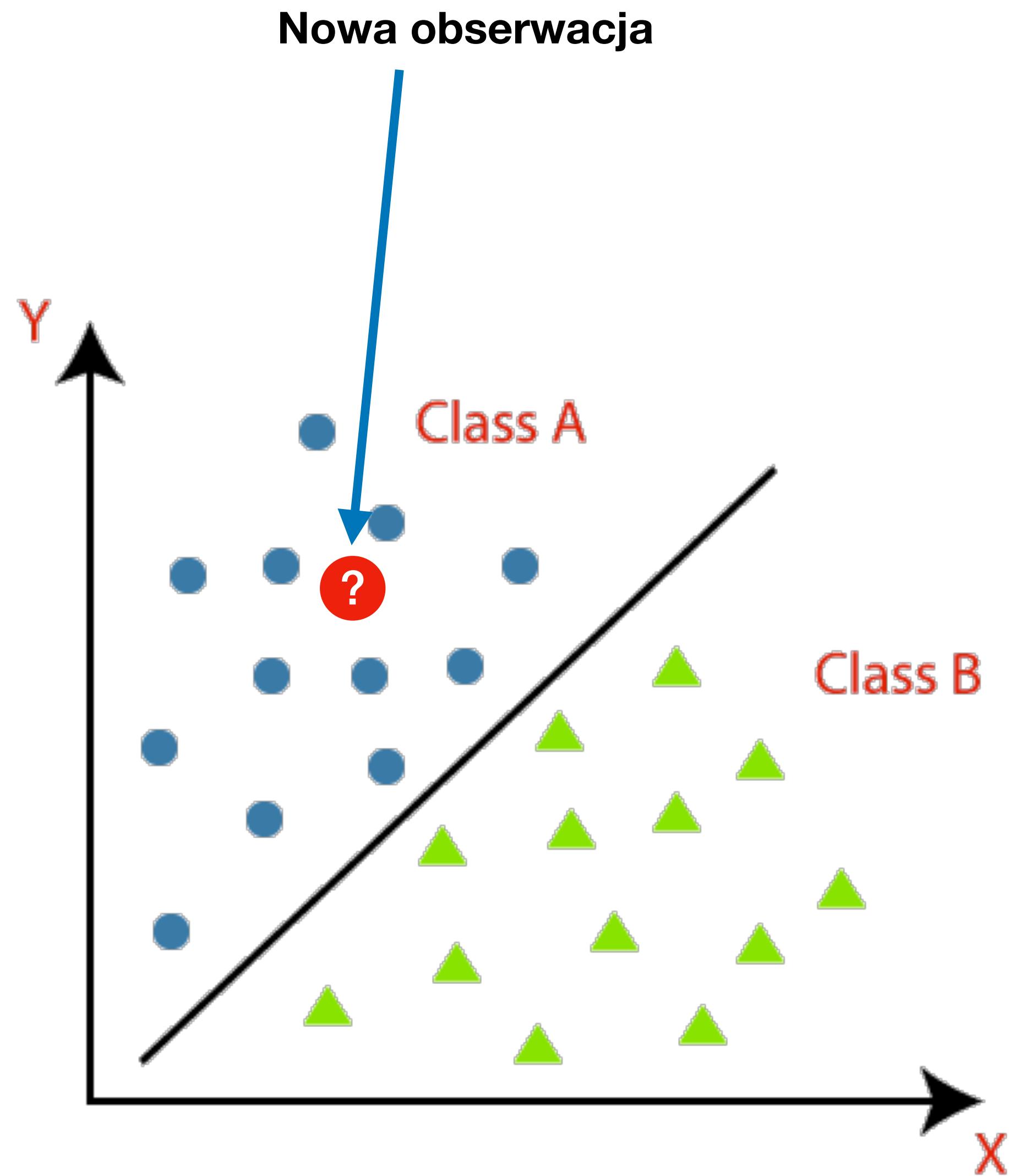


Uczenie nadzorowane - przykłady klasyfikacji (classification)

- Klasyfikacja sentymenu - określenie czy dany tekst ma wydźwięk pozytywny lub negatywny
- Filtr spamu - klasyfikacja email do jednej z dwóch grup: spam, nie-spam
- Rozpoznawanie cyfr: klasyfikacja do jednej z 10 grup (0...10)



Uczenie nadzorowane - ilustracja klasyfikacji (classification)

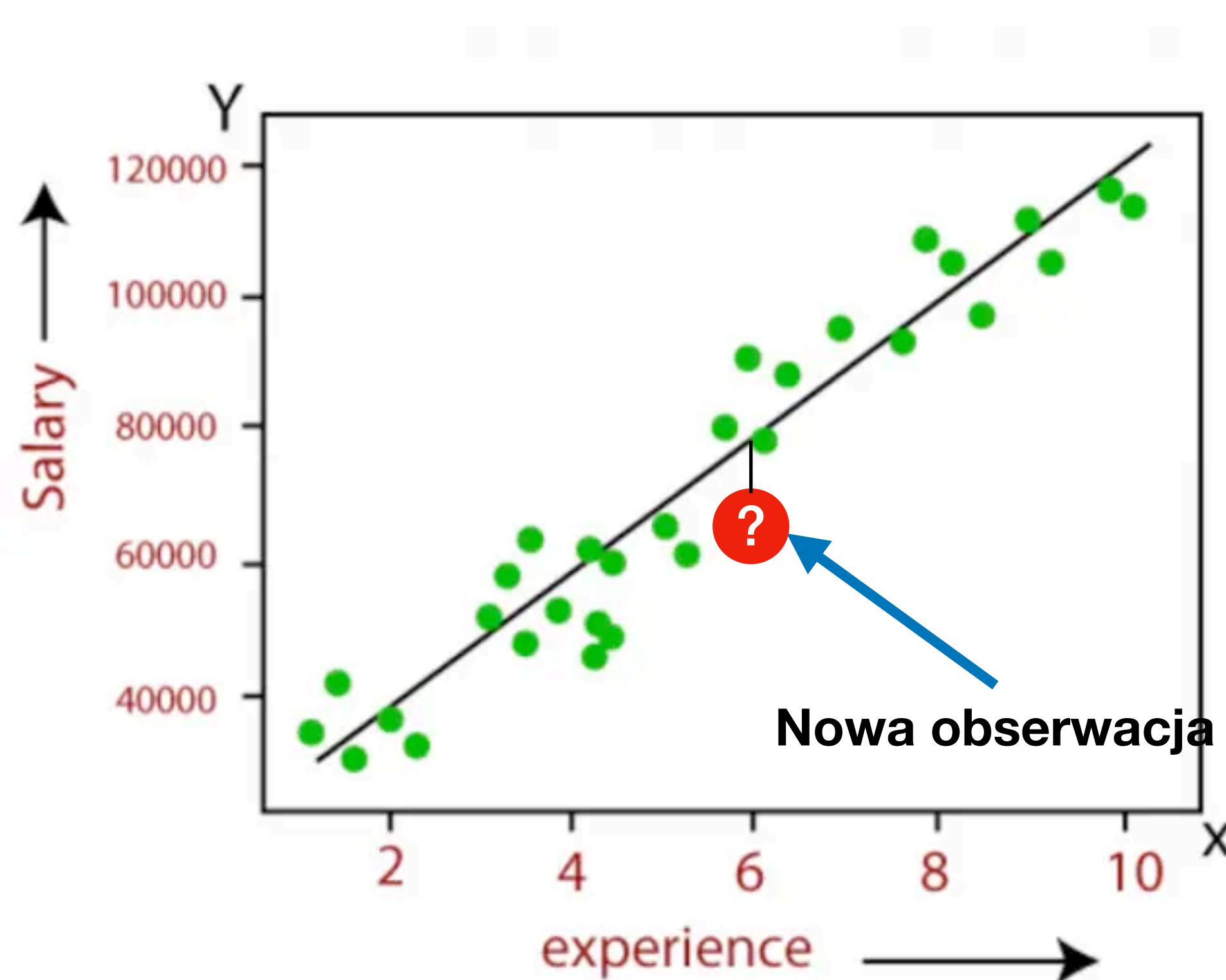


- Zbiór uczący zawierający 25 elementów
- 2 klasy: niebieskie kropki i zielone trójkąty
- Zbiór jest 2-wymiarowy (każdy element zbioru jest opisany przez 2 zmienne)
- Nasz algorytm uczenia maszynowego nauczył się oddzielać kropki od trójkątów
- Modelem jest płaszczyzna oddzielająca dwie klasy od siebie
- Płaszczyzna ta jest w stanie oddzielić (=zaklasyfikować) nową obserwację, której dotychczas nie widzieliśmy

Uczenie nadzorowane - przykłady regresji (regression)

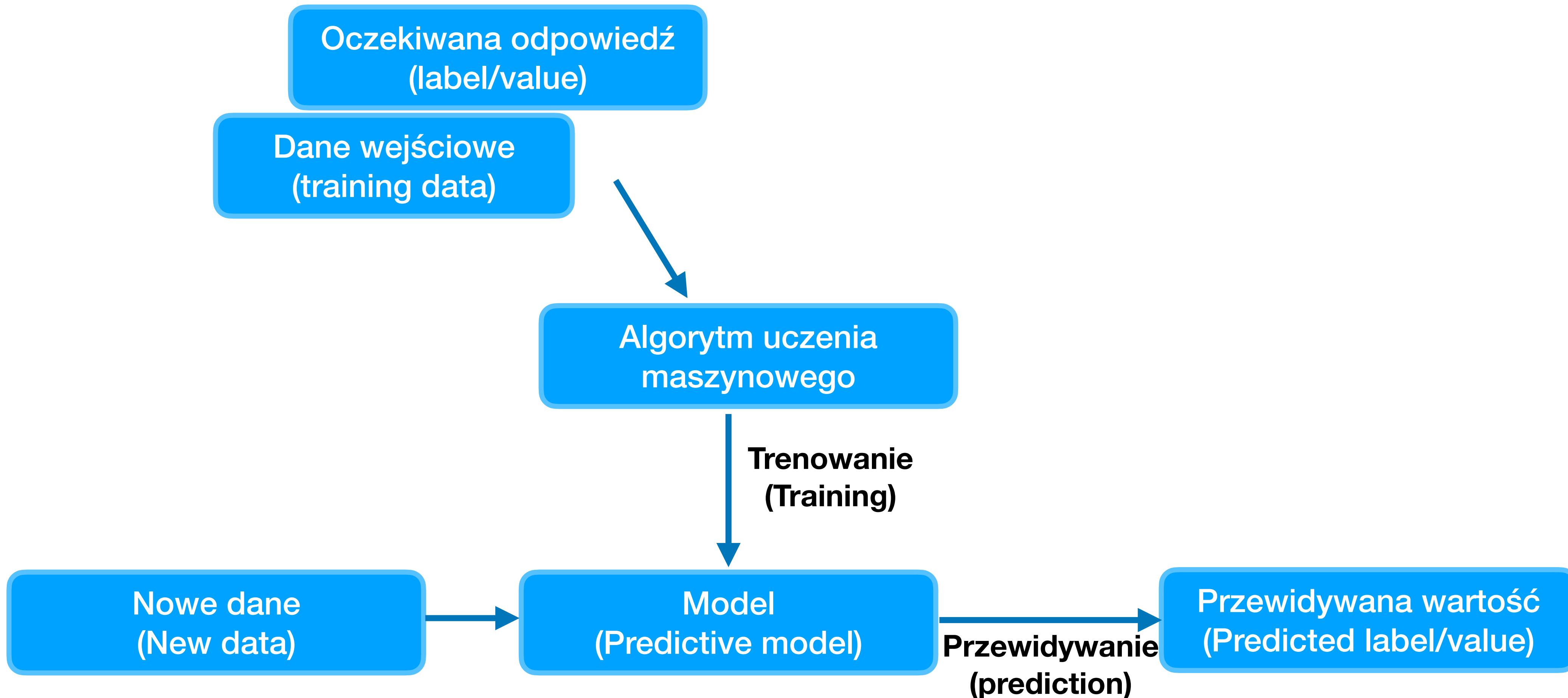
- Przewidywanie cen nieruchomości na podstawie danych takich, jak: ilość pokoi, metraż, lokalizacja, piętro, odległość od przystanku autobusowego, itd.
- Przewidywanie wielkości sprzedaży w oparciu o dane historyczne
- Przewidywanie zapotrzebowania na energię elektryczną w zależności od pogody, godziny, dnia tygodnia
- Przewidywanie wpływu sytuacji makroekonomicznej na wyniki finansowe firmy
- Przewidywanie serii czasowych

Uczenie nadzorowane - przykłady regresji (regression)



- Zielone kropki to obserwacje: punkty danych
- Oś X to tzw. zmienna niezależna (feature variable or explanatory variable)
- Oś Y to tzw. zmienna zależna (target variable or response variable)
- Modelem jest prosta minimalizującą odległość między każdym z tych punków i tą prostą
- Najczęściej odległość ta jest wyrażona jako suma kwadratów odległości (o tym później)

Uczenie nadzorowane - schemat uczenia



Model

- Generalizacja rzeczywistości przydatna do rozwiązania jakiegoś konkretnego problemu.
- Uchwycenie istotnych aspektów rzeczywistości przy jednoczesnym redukowaniu nadmiernych szczegółów.
- Model (w uczeniu maszynowym) pozwala nam na podjęcie decyzji w oparciu o nowe dane, które nie były znane wcześniej.
- Najczęściej jest to model matematyczny (np. funkcja), program, zbiór reguł itd.
- Model jest budowany na podstawie danych „trenujących” (training data) w trakcie tzw. „uczenia” (model training), jest zoptymalizowany dla konkretnego problemu, np. klasyfikacji.

Ale po co tworzyć modele?

- Przewidywanie (prediction) — użycie modelu do przewidywania pewnej wartości Y (która jest trudna do uzyskania) na podstawie innych zmiennych, które są łatwo dostępne (X_1, X_2, \dots, X_n)
- Wnioskowanie, inferencja (inferencje) — poznanie w jaki sposób zmienne niezależne X_1, X_2, \dots, X_n wpływają na interesującą nas wartość Y . Jakie z nich mają większy albo mniejszy wpływ? Jak Y zmienia się pod ich wpływem? Jaka to jest zależność?

Uczenie nienadzorowane (unsupervised learning)

Cechy:

- Nasz zbiór danych nie posiada etykiet lub oczekiwanej wartości
- Ingerencja człowieka jest minimalna; nie wymagamy przygotowania danych lub „nadzoru” nad uczeniem
- Nie znamy dobrze struktury danych; naszym celem jest odkrycie pewnych cech tej struktury

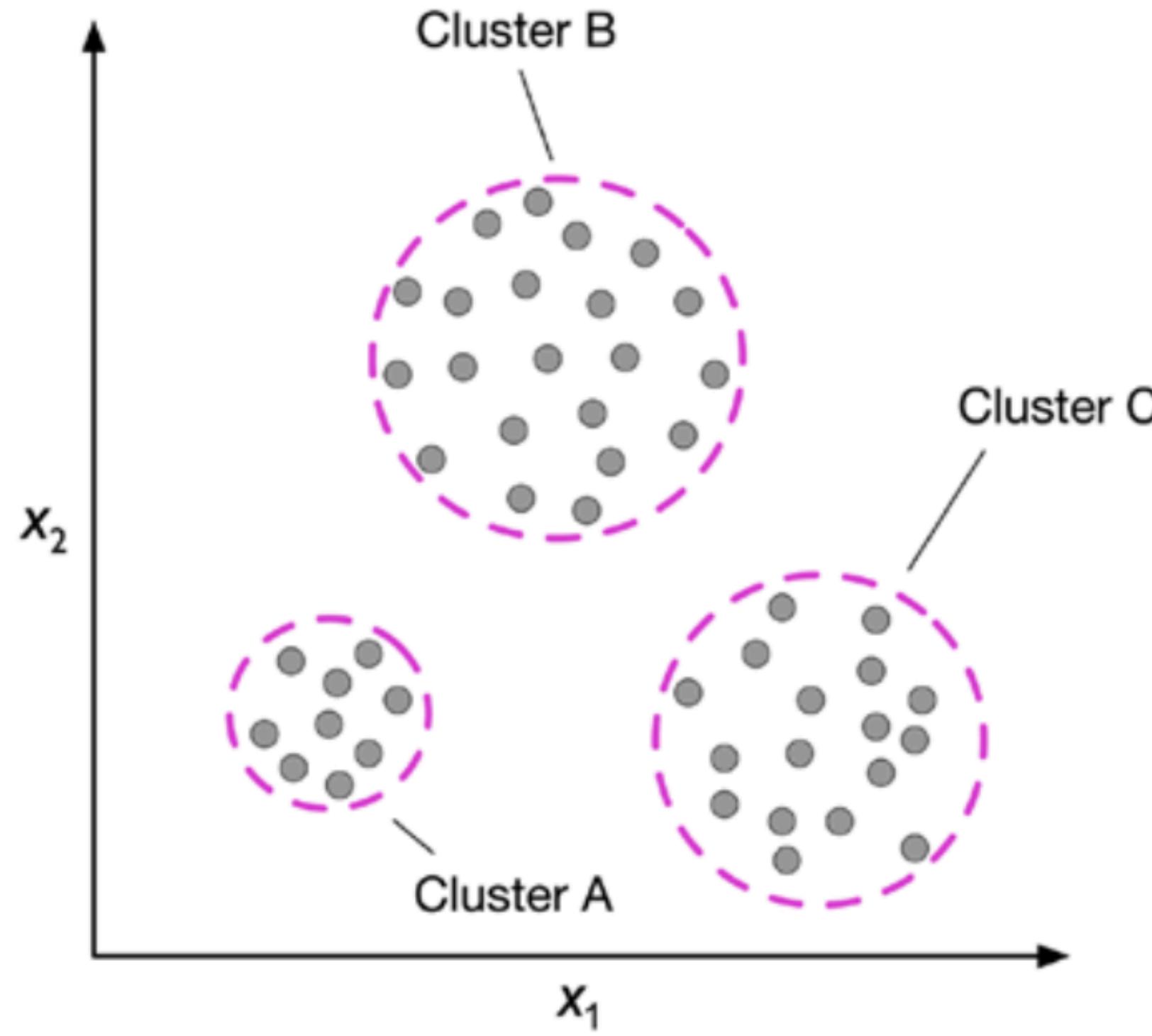
Uczenie nienadzorowane (unsupervised learning)

Rodzaje:

- Analiza skupień (clustering)
- Analiza składowych głównych (PCA - Principal Component Analysis)
- Metody redukcji wymiarowości (dimensionality reduction) (t-SNE, UMAP)
- Reguły asocjacyjne (association rule learning)

Uczenie nienadzorowane

- analiza skupień



Organizacja informacji i odkrywanie grup (clusters) danych bez wcześniejszej wiedzy o ich istnieniu. Np. Identyfikacja grup klientów pod kątem marketingowym.

Przykłady algorytmów:

- k-Means
- DBSCAN

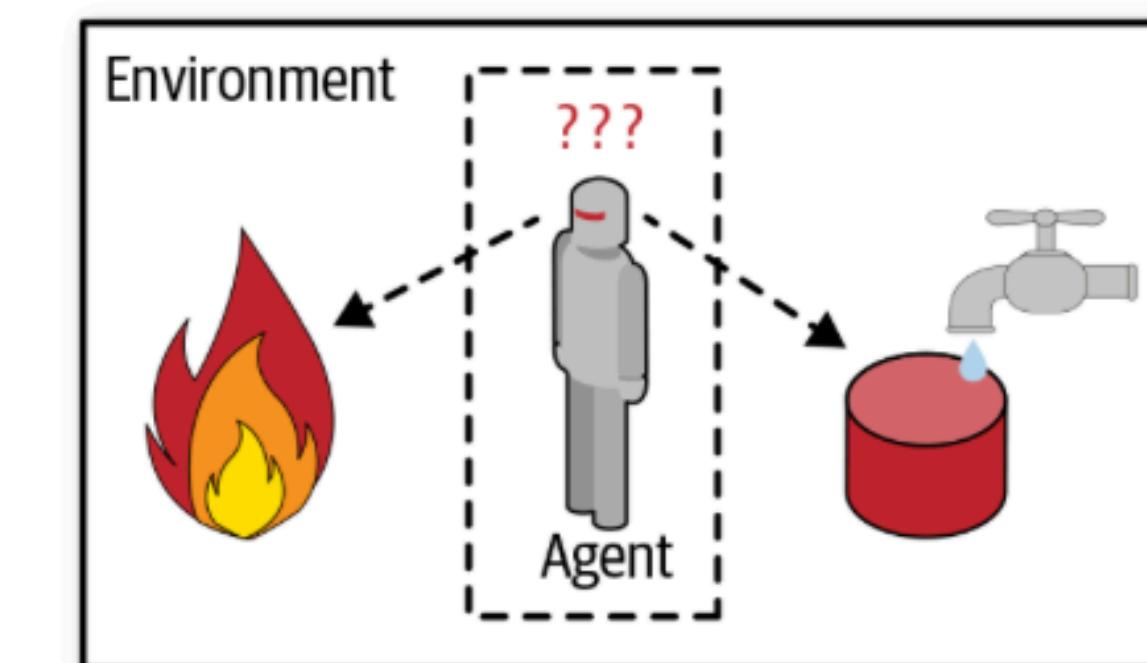
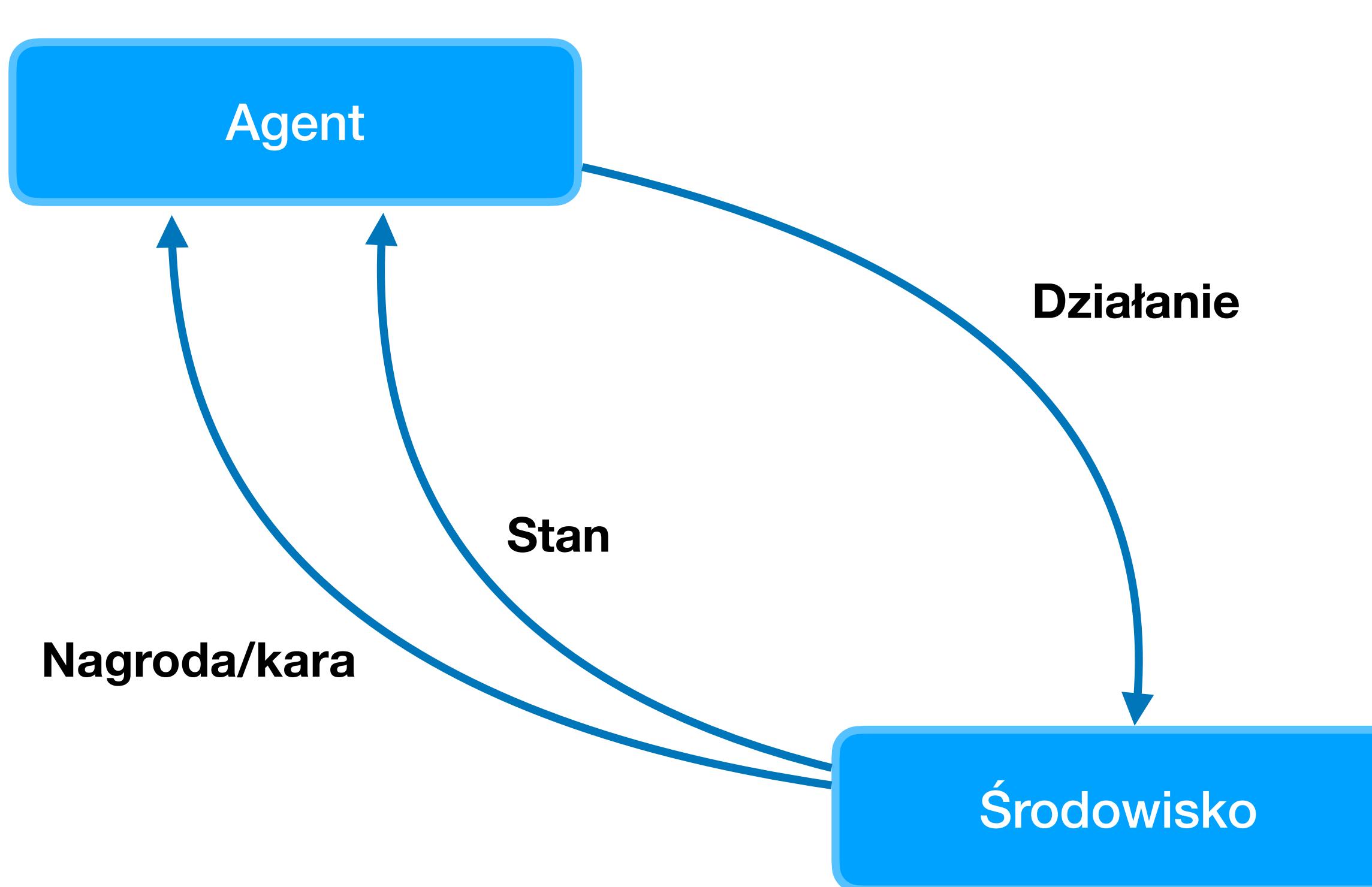
Analiza skupień może być również użyta w celu eliminowania danych nietypowych/odstających (outliers)

Uczenie przez wzmacnianie (reinforcement learning)

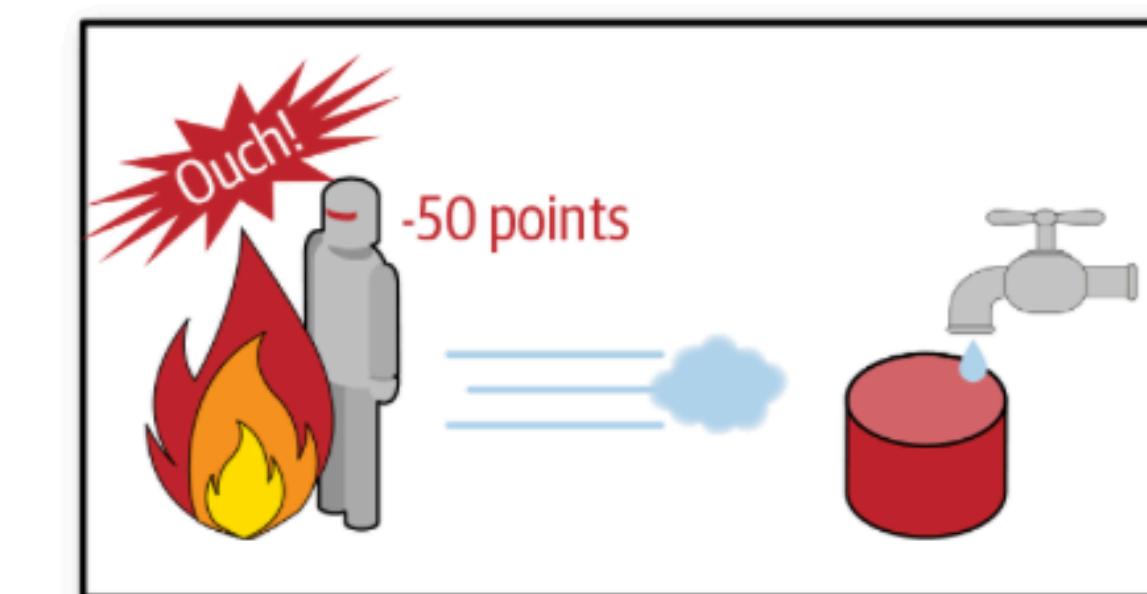
Istnieje cała grupa metod uczenia przez wzmacnianie. Ich głównym założeniem jest:

- Uczenie poprzez interakcje ze środowiskiem.
- Agent za swoje działania otrzymuje nagrody (positive rewards) lub kary (negative rewards). Nagrodą może być osiągnięcie celu lub poprawne wykonanie pewnego kroku.
- Agent dąży do maksymalizacji nagród.

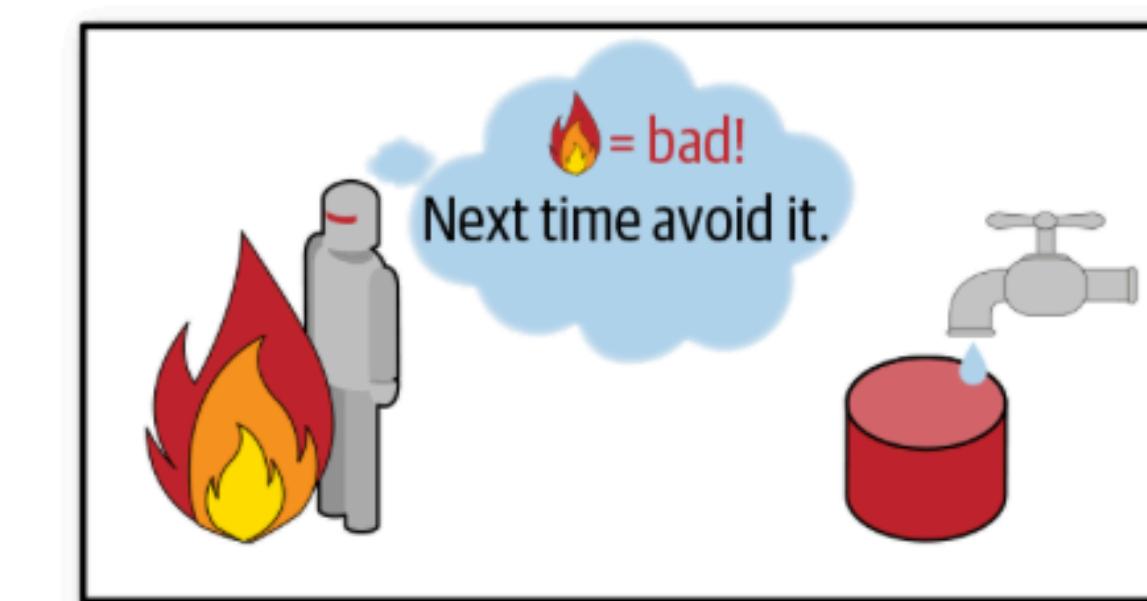
Uczenie przez wzmacnianie (reinforcement learning)



- 1 Observe
- 2 Select action using policy



- 3 Action!
- 4 Get reward or penalty



- 5 Update policy (learning step)
- 6 Iterate until an optimal policy is found

Regresja liniowa

Zanim zaczniemy

- Zbiór danych (dataset) „Advertising”
 - Zawiera dane sprzedaży pewnego produktu
 - na 200 różnych rynkach...
 - wraz z budżetami przeznaczonymi na...
 - 3 różne kanały marketingowe:
 - TV, radio, prasa

Przyjrzymy się relacji miedzy budżetem na TV i wielkością sprzedaży

Notebook: 01 - Loading Data with Pandas

Pandas - popularna biblioteka języka Python do analizy i przetwarzania danych.

Pandas Data Frames (ramki danych)

EDA - Exploratory Data Analysis

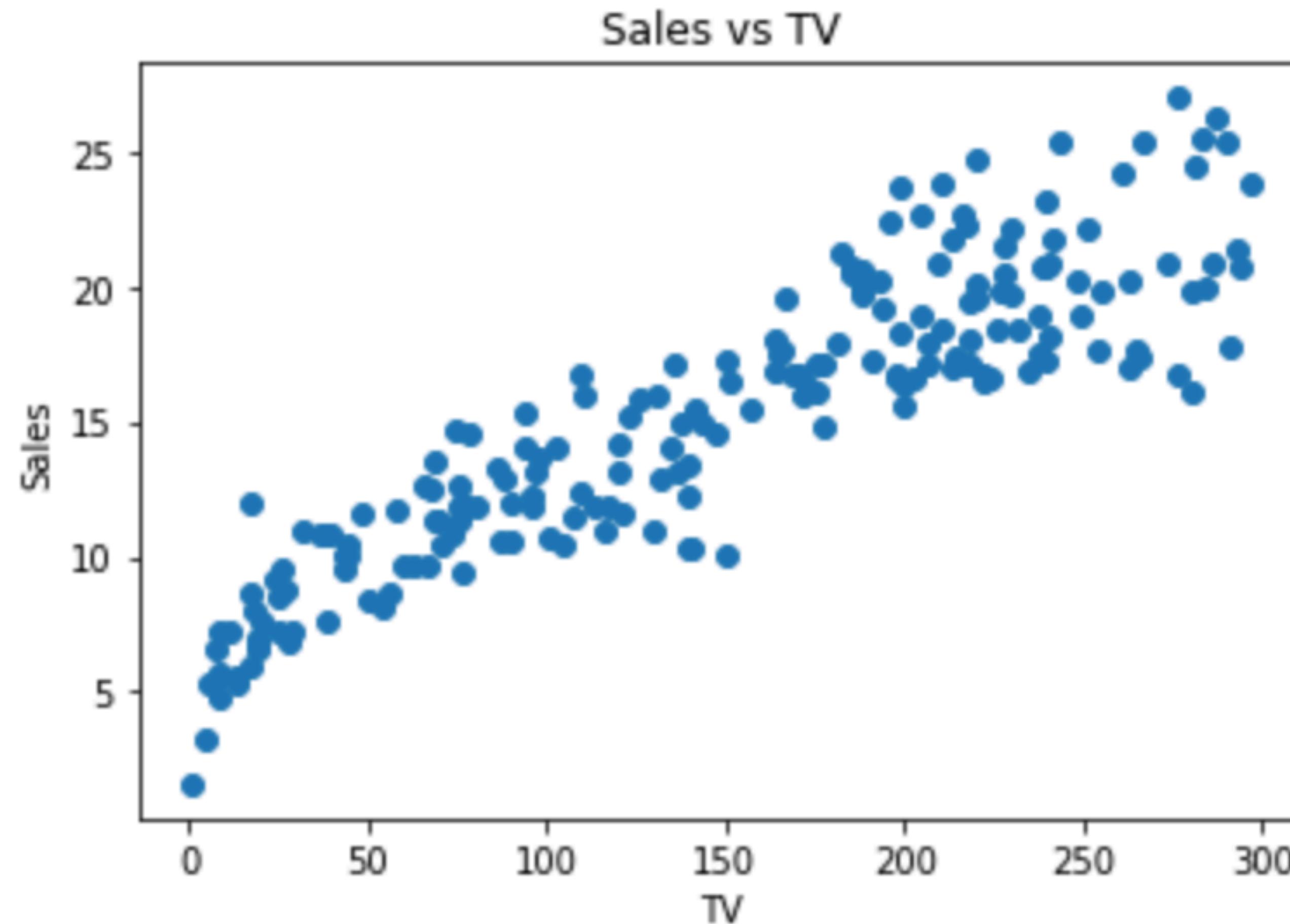
```
df = pd.read_csv('advertising.csv')  
#just print the data frame  
df
```

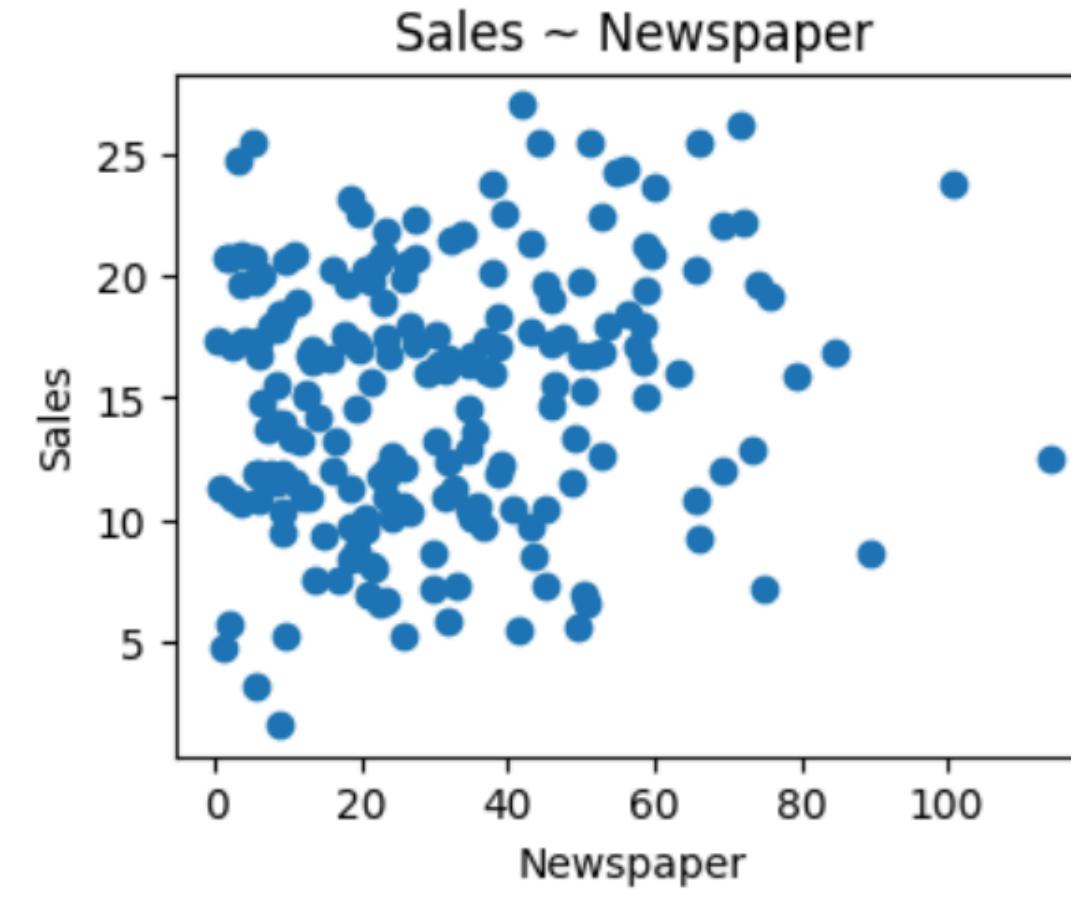
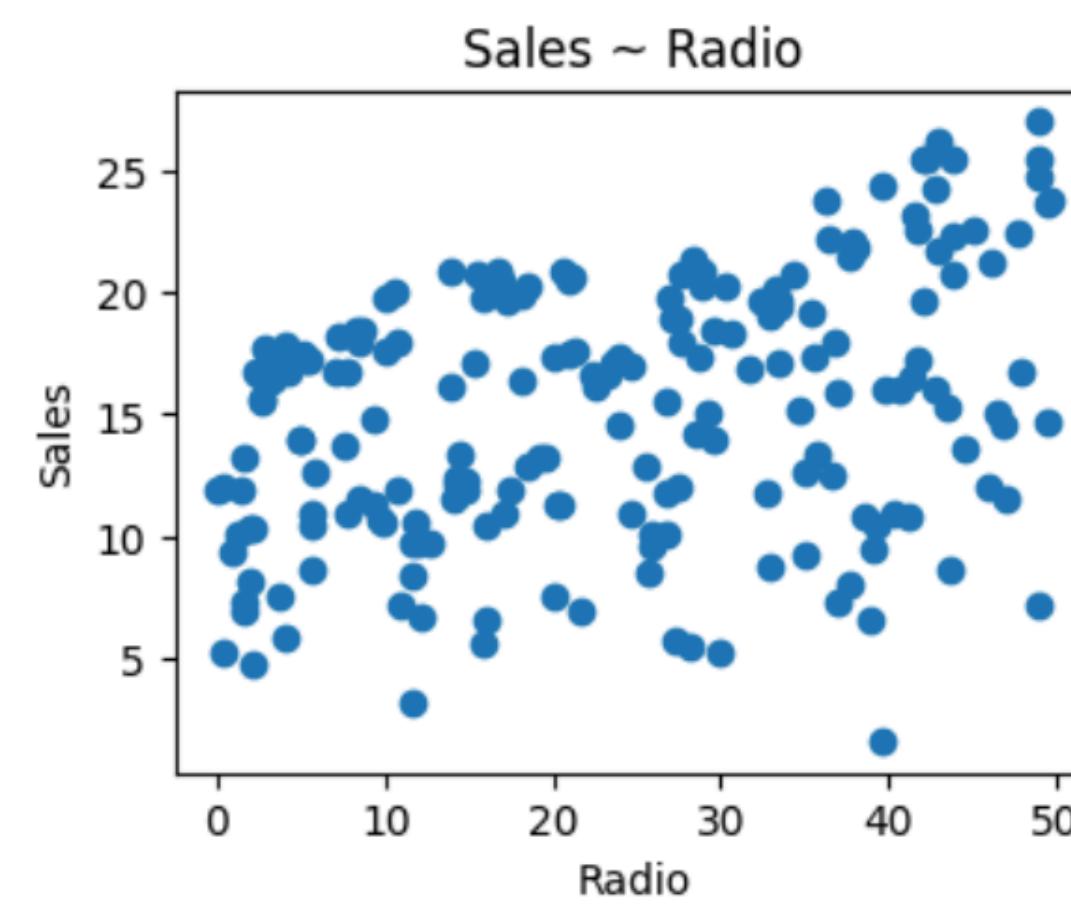
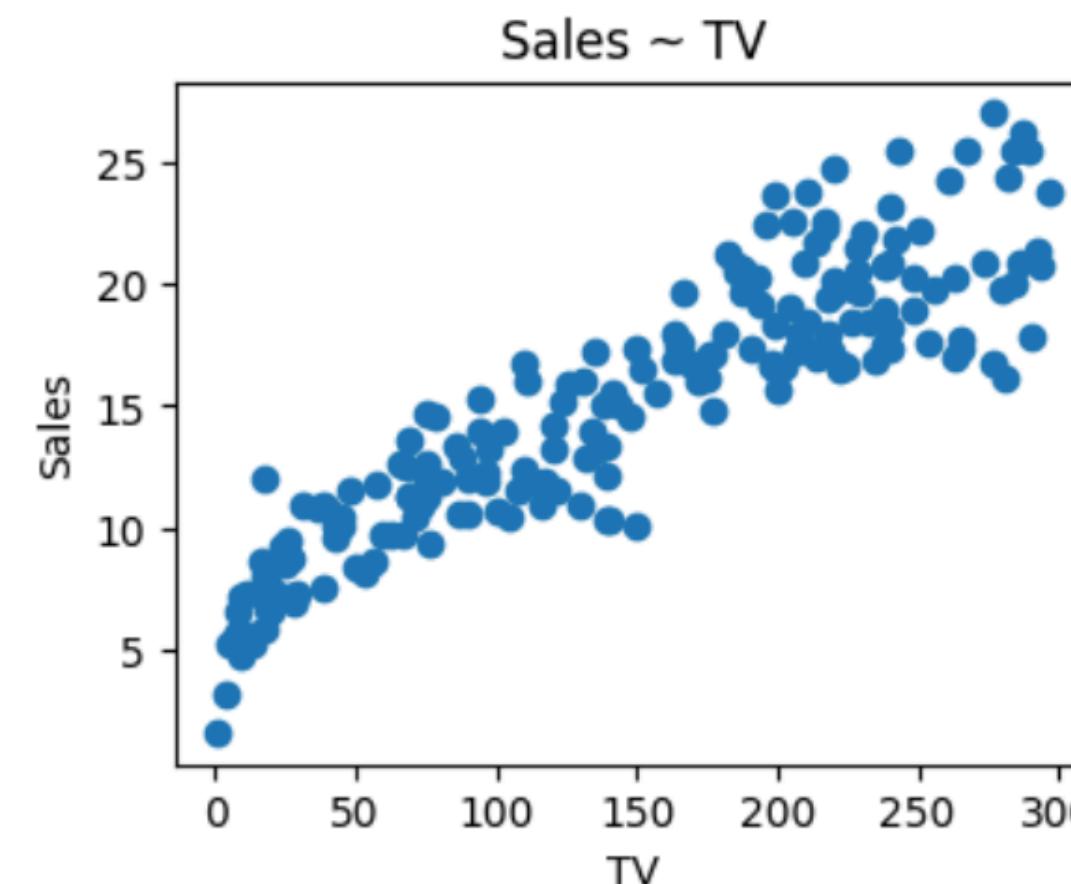
Obserwacje

	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

200 rows × 4 columns

```
# wykres punktowy (scatter plots)
plot.scatter(df[ "TV" ], df[ "Sales" ])
plot.xlabel( "TV" )
plot.ylabel( "Sales" )
plot.title( "Sales vs TV" )
```





	TV	Radio	Newspaper	Sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	12.0
3	151.5	41.3	58.5	16.5
4	180.8	10.8	58.4	17.9
...
195	38.2	3.7	13.8	7.6
196	94.2	4.9	8.1	14.0
197	177.0	9.3	6.4	14.8
198	283.6	42.0	66.2	25.5
199	232.1	8.6	8.7	18.4

200 rows × 4 columns

Regresja liniowa - założenia

Prosta regresja liniowa (univariate or simple linear regression) polega na przewidywaniu odpowiedzi Y (**zmiennej zależnej**) na podstawie pojedynczej **zmiennej niezależnej** X. Regresja liniowa zakłada, że pomiędzy X i Y istnieje zależność liniowa, tzn taka którą da się opisać funkcją liniową:

$$Y \approx \beta_0 + \beta_1 X$$

$\hat{\beta}_0, \hat{\beta}_1$ nazywane są parametrami modelu. Trenowanie modelu polega na estymacji tych parametrów, a następnie na postawie wartości x przewidzenie \hat{y}

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$$

$\hat{y}, \hat{\beta}_0, \hat{\beta}_1$ - daszek nad tymi symbolami oznacza estymowaną wartość (nieznaną)

Regresja liniowa - estymacja parametrów

Parametry β_0, β_1 są nieznane, więc aby dokonać predykcji należy najpierw je wyznaczyć.

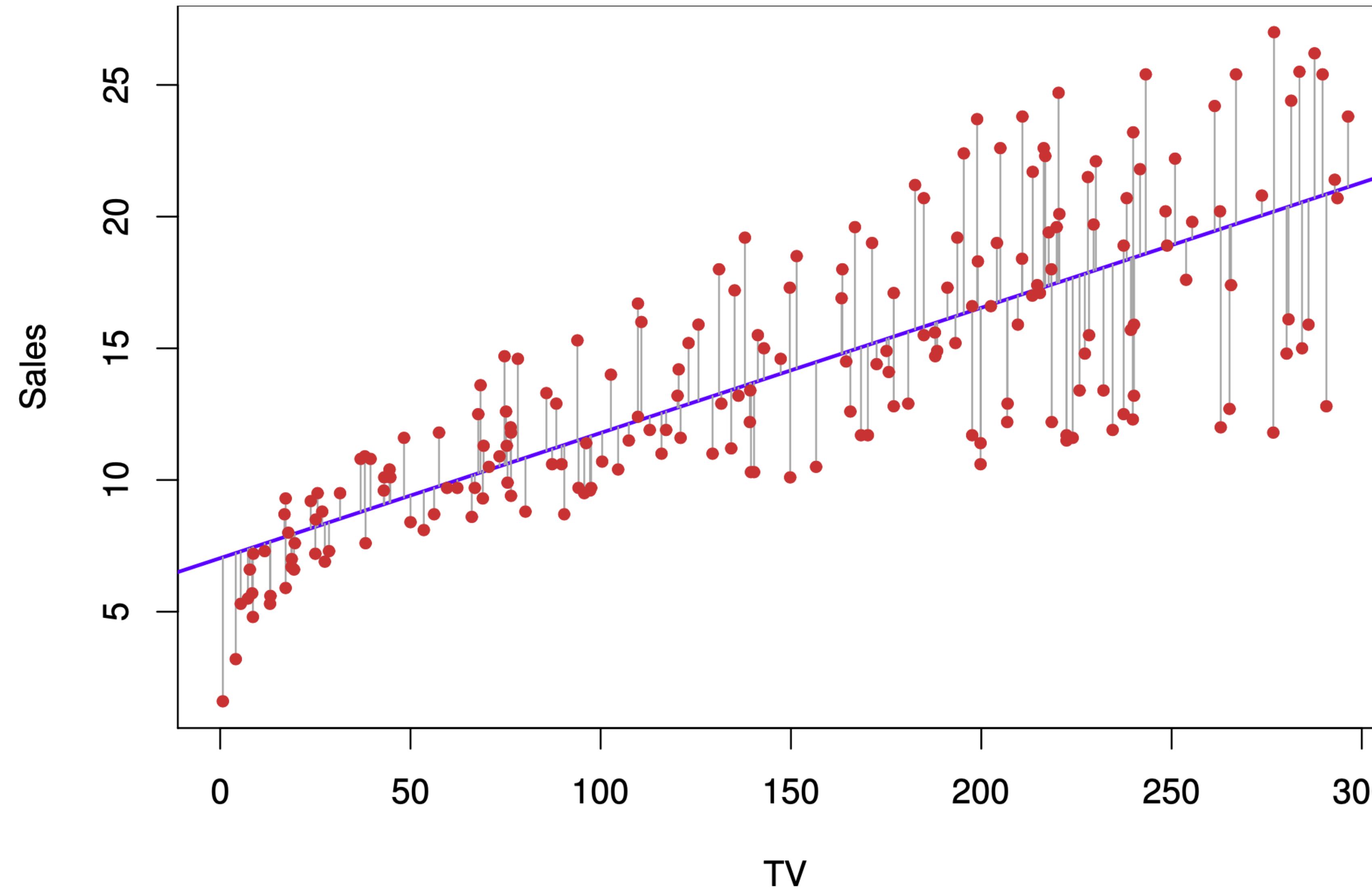
Używamy do tego naszego zbioru n obserwacji (punktów):

$$(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)$$

Celem jest takie wyznaczenie parametrów β_0, β_1 aby równanie było spełnione dla wszystkich obserwacji:

$$y_i \approx \hat{\beta}_0 + \hat{\beta}_1 x_i \text{ dla } i = 1 \dots n$$

Estymacja parametrów – intuicja



Chcemy aby prosta
 $y_i \approx \hat{\beta}_0 + \hat{\beta}_1 x_i$ była jak
najbliżej wszystkich
punktów.

Co to znaczy „jak
najbliżej”?

Regresja liniowa - funkcja straty (loss function)

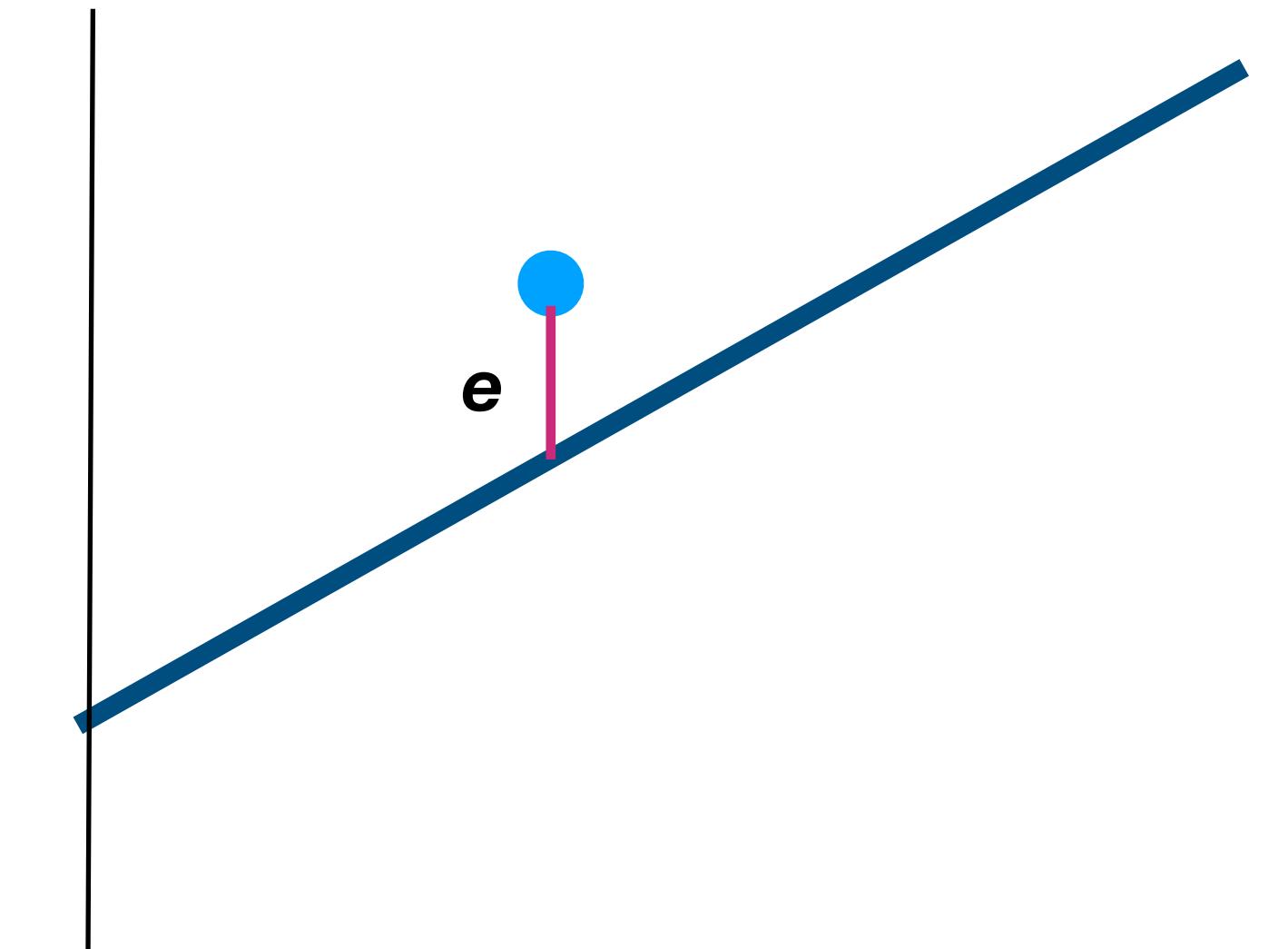
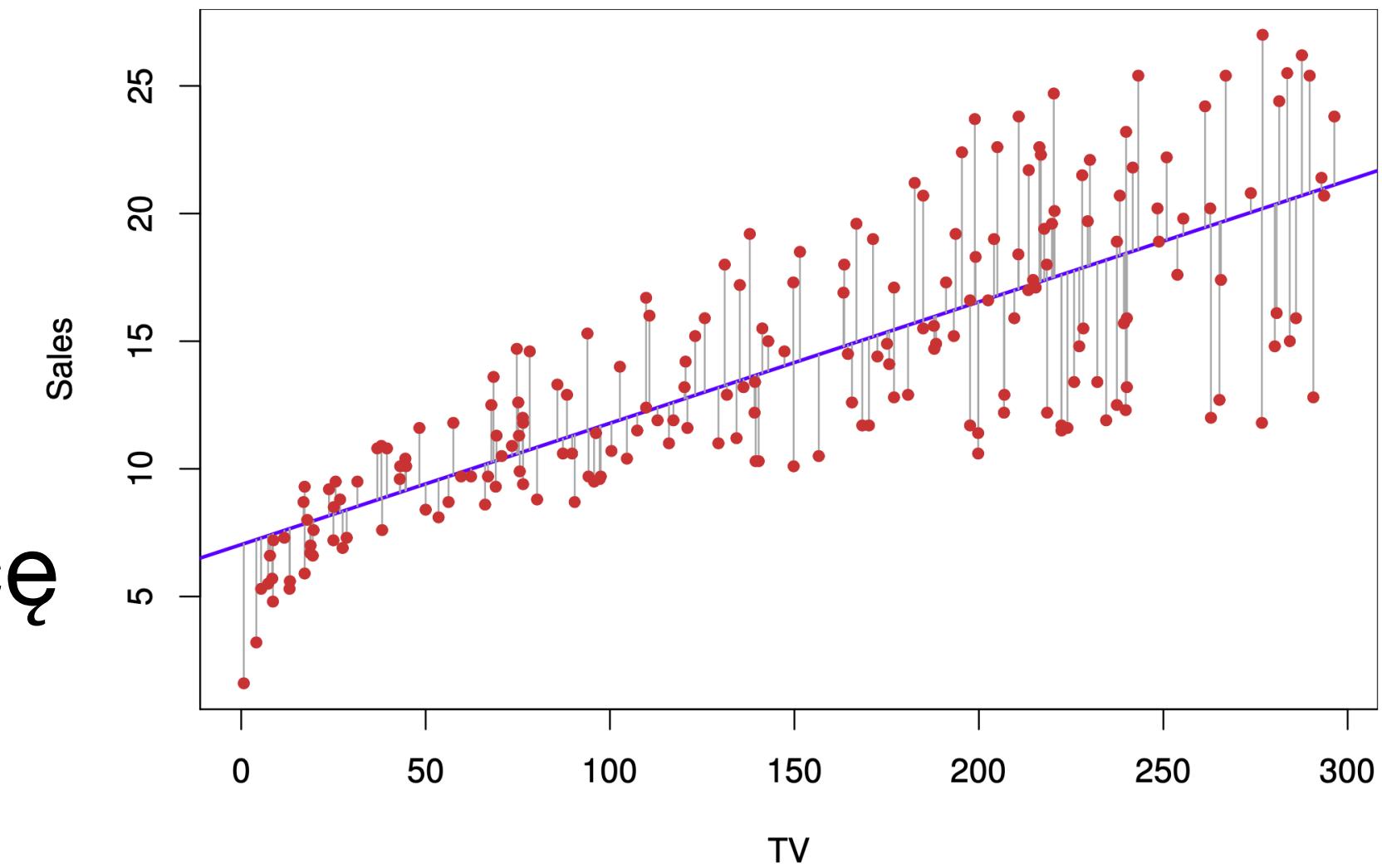
Jeśli $\hat{y}_i \approx \hat{\beta}_0 + \hat{\beta}_1 x_i$ opisuje przewidywana wartość Y w oparciu o i-tą wartość X, to zdefiniujmy:

$e_i = y_i - \hat{y}_i$ jako wartość resztową (residual), czyli różnicę między obserwacją (y_i) a wartością przewidzianą (\hat{y}_i)

Sumą kwadratów wszystkich wartości resztowych będzie opisana wzorem:

$$RSS = e_1^2 + e_2^2 + \dots + e_n^2$$

(RSS - Residual sum of squares)



Chwila, co?

- To jest moment w którym definiujemy naszą „funkcję celu” (objective function).
- Funkcja celu pomaga nam ocenić jak dobra jest nasza regresja: tj. im suma błędów jest większą tym regresja jest gorsza (prosta jest „daleko od punktów”).
- Naszym celem jest zatem optymalizacja funkcji celu aby suma była jako najmniejsza -> będziemy dążyli aby funkcja dążyła do minimum.
- Dlaczego podnosimy do kwadratu? Gauss tak wymyślił!
 - Na serio: funkcja kwadratowa ma ciekawe własności: większe liczby (błąd) podniesione do kwadratu dają jeszcze większą wartość (i na odwrót)
 - Penalizujemy w ten sposób duże odchylenia a faworyzujemy małe
 - Także: kwadrat wartości ujemnej (odchylenie w drugą stronę) jest wartością dodatnią



Regresja liniowa - RSS

Find $\min_{\hat{\beta}_0, \hat{\beta}_1} Q(\hat{\beta}_0, \hat{\beta}_1) = \min RSS$

$$RSS = e_1^2 + e_2^2 + \dots + e_n^2 = (y_1 - \hat{\beta}_0 - \hat{\beta}_1 x_1)^2 + (y_2 - \hat{\beta}_0 - \hat{\beta}_1 x_2)^2 + \dots + (y_n - \hat{\beta}_0 - \hat{\beta}_1 x_n)^2$$

Dla regresji prostej łatwo można policzyć tzw. rozwiązanie analityczne. Rozwiążując to równanie dla parametrów $\hat{\beta}_0, \hat{\beta}_1$ otrzymamy:

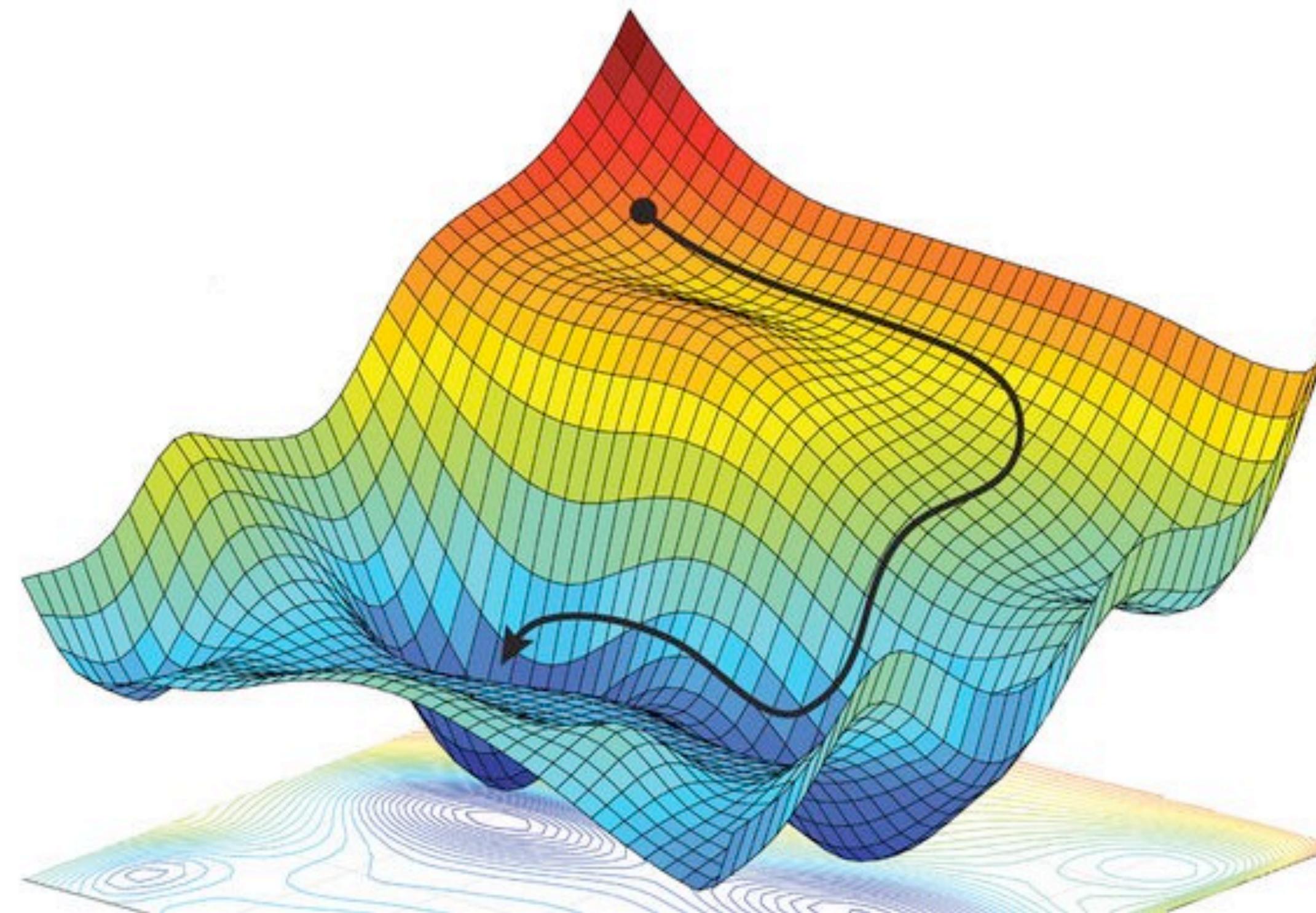
$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}$$

Gdzie: $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ oraz $\bar{x} = \frac{1}{n} \sum_{i=1}^n x_i$ (po prostu są to średnie z X i Y)

Regresja liniowa - poszukiwanie minimum

Rozwiązanie analityczne jest OK dla regresji prostej (z jedną zmienną). W praktyce do poszukiwania minimum funkcji celu używamy metody gradientu prostego.



Regresja liniowa - poszukiwanie minimum

Funkcja celu (Q) ma najczęściej postać MSE (ang. Mean Squared Error), tj. $MSE = \frac{RSS}{n}$

Zapisując inaczej, poszukujemy parametrów $\hat{\beta}_0, \hat{\beta}_1$ takich, aby błąd średnio kwadratowy był najmniejszy:

$$\min_{\hat{\beta}_0, \hat{\beta}_1} Q(\hat{\beta}_0, \hat{\beta}_1) = \min MSE$$

$$MSE = \frac{e_1^2 + e_2^2 + \dots + e_n^2}{n} = \frac{1}{n} \sum_{i=0}^n (y_i - \hat{y}_i)^2 = \frac{1}{n} \sum_{i=0}^n (y_i - (\hat{\beta}_1 x_i + \hat{\beta}_0))^2$$

Obliczamy pochodne cząstkowe (partial derivative) dla $\hat{\beta}_0, \hat{\beta}_1$:

$$D_{\hat{\beta}_1} = \frac{1}{n} \sum_{i=0}^n 2(y_i - (\hat{\beta}_1 x_i + \hat{\beta}_0))(-x_i) = \frac{-2}{n} \sum_{i=0}^n x_i(y_i - \hat{y}_i)$$

$$\text{Tak samo dla } \hat{\beta}_0: D_{\hat{\beta}_0} = \frac{-2}{n} \sum_{i=0}^n (y_i - \hat{y}_i)$$

Regresja liniowa - poszukiwanie minimum

Jak znaleźć $\hat{\beta}_0, \hat{\beta}_1$?

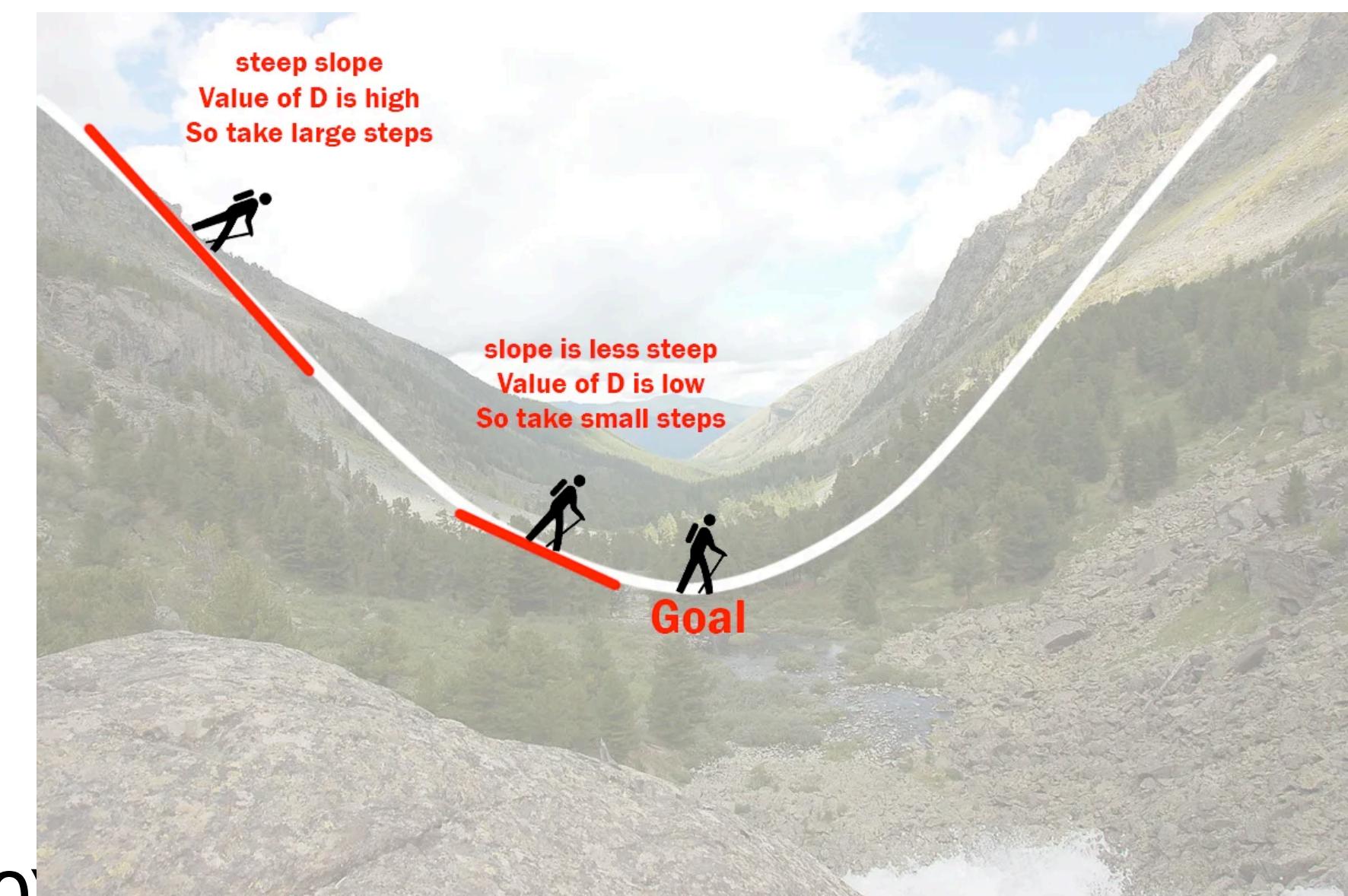
Zaczynamy od jakiejś dowolnej wartości $\hat{\beta}_0, \hat{\beta}_1$ (0, 0). Ustalamy współczynnik α (learning rate) - to będzie „długość naszego kroku”, czyli jak szybko będziemy zmieniać wartości $\hat{\beta}_0, \hat{\beta}_1$

Uaktualniamy wartości $\hat{\beta}_0, \hat{\beta}_1$:

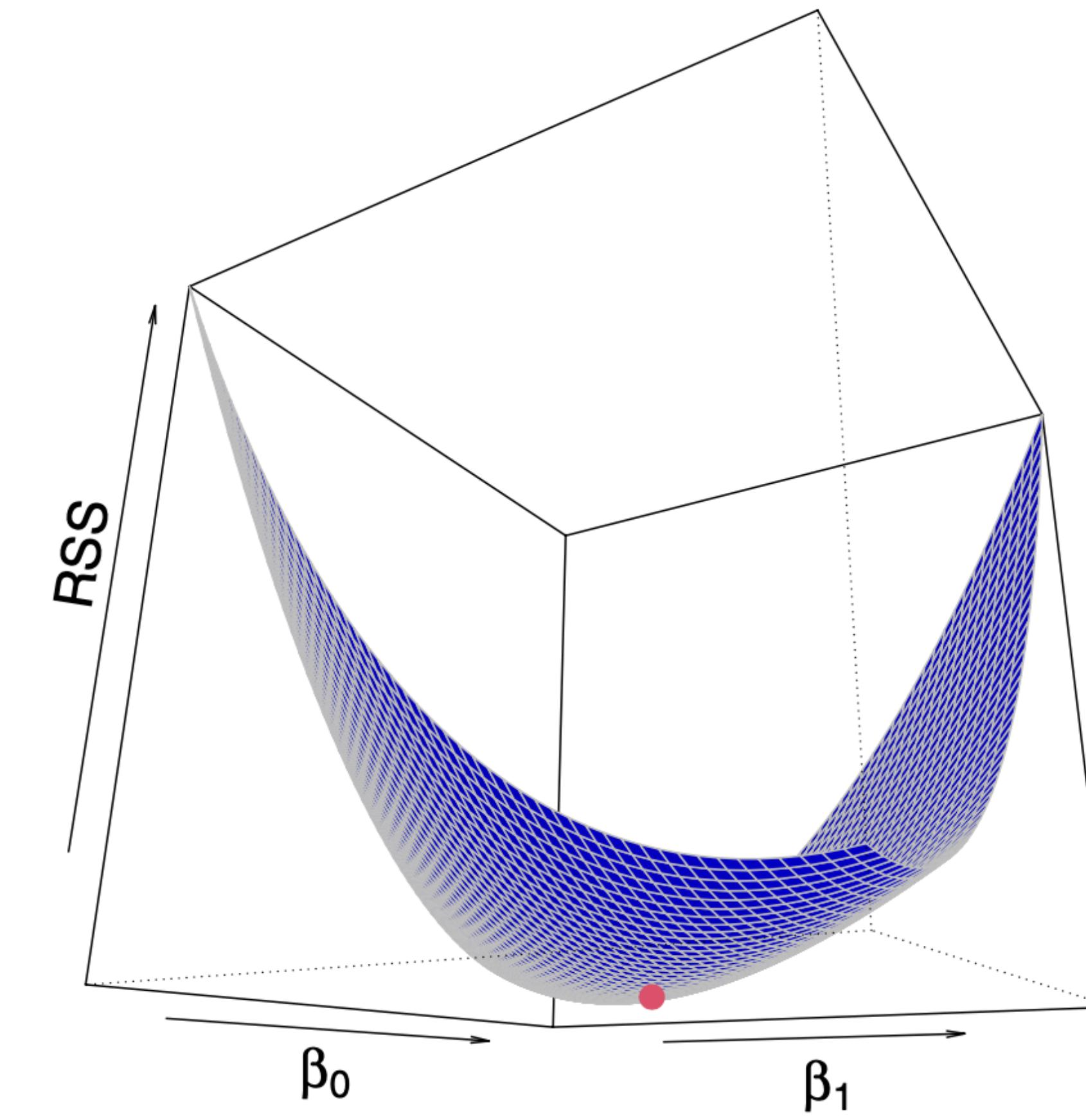
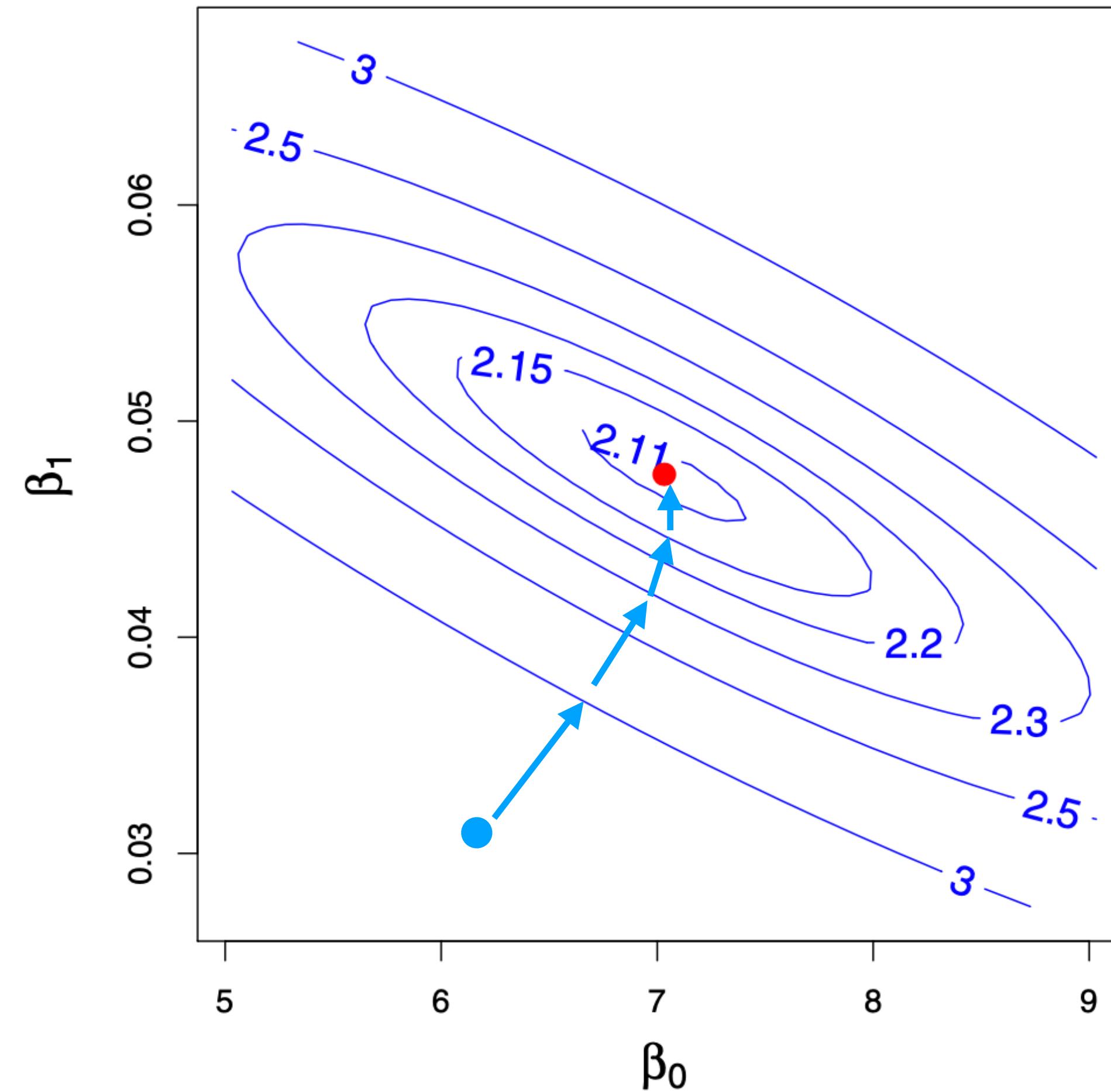
$$\hat{\beta}_1 = \hat{\beta}_1 - \alpha D_{\hat{\beta}_1}$$

$$\hat{\beta}_0 = \hat{\beta}_0 - \alpha D_{\hat{\beta}_0}$$

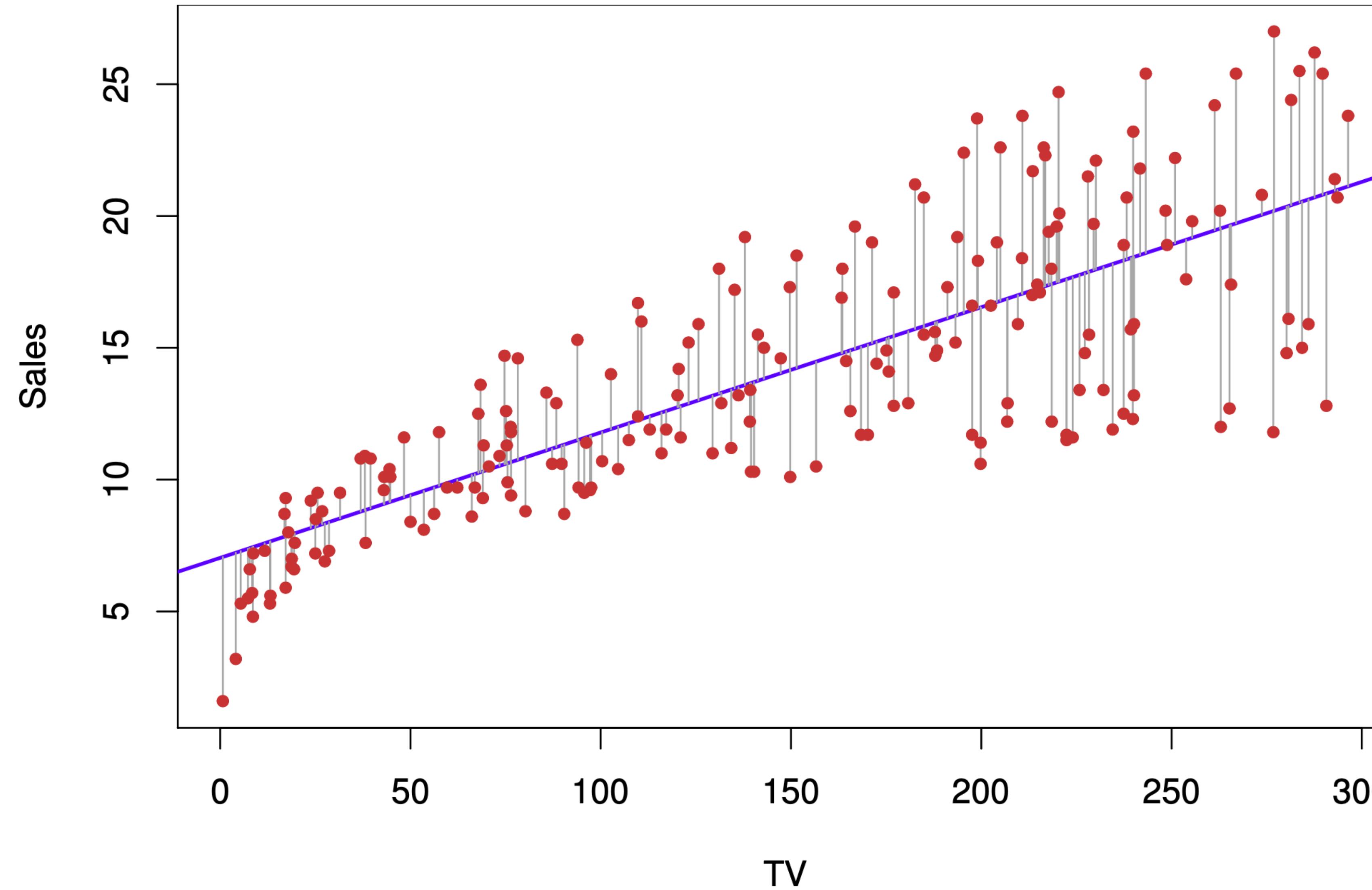
Kontynuujemy aż nasza funkcja straty jest bardzo mała (bliska 0)



Regresja liniowa - minimalizacja RSS metoda gradientu prostego



Regresja liniowa - RSS



Wizualizacja prostej
regresji liniowej (linear
regression fit)

$$\hat{\beta}_0 = 6,97$$

$$\hat{\beta}_1 = 0,055$$

Czyli: każde kolejne
1000\$ wydane na
reklamę w TV to
 $1000 \cdot 0,055 = 55$
dodatkowych jednostek
sprzedaży

Ewaluacja parametrów

- Założenie regresji liniowej jest takie, że istnieje zależność pomiędzy X i Y. Jak to sprawdzić?
 - Hipoteza zerowa: $H_0: \beta_1 = 0$
 - Hipoteza alternatywna: $H_1 : \beta_1 \neq 0$
- β_1 powinno być dalekie od zera ale jak bardzo?
 - 2x odchylenia standardowe (SE, Standard Error)
 - P-value: niska wartość wskazuje, że jest to bardzo mało prawdopodobne, że zależność Y od X jest przypadkowa zakładając, że Y i X w rzeczywistości od siebie nie zależą. Przyjęto się, że wartości 0.05 albo 0.01 wystarczają, żeby odrzucić hipotezę zerową.

Ewaluacja modelu

- Kiedy udowodniliśmy, że istnieje zależność Y od X, warto jeszcze sprawdzić jak dobry jest nasz model. Popularne metryki:
 - RSE (Residual Standard Error) - odchylenie standardowe składnika resztowego
$$RSE = \sqrt{\frac{1}{n - 2} RSS}$$
 - $MSE = \frac{1}{N} \sum_{i=1}^n (y_i - \hat{y})^2$ & $RMSE = \sqrt{MSE}$
 - R^2 - współczynnik determinacji - liczba z przedziału $<0,1>$. Określa jaka część zmienności/wariancji Y jest wyjaśniona przez korelacje z X. R^2 bliskie 1 oznacza, że większość wariancji wyjaśniona jest zależnością liniową.

Ewaluacja parametrów w praktyce

Odchylenie standardowe

Parametry:
const to β_0

TV to β_1

```
import statsmodels.api as sm
from scipy import stats

X_train = df["TV"]
y_train = df["Sales"]

X2 = sm.add_constant(X_train)
est = sm.OLS(y_train, X2)
regression_model = est.fit()
print(regression_model.summary())
```

OLS Regression Results						
Dep. Variable:	Sales	R-squared:	0.812			
Model:	OLS	Adj. R-squared:	0.811			
Method:	Least Squares	F-statistic:	856.2			
Date:	Fri, 12 May 2023	Prob (F-statistic):	7.93e-74			
Time:	21:32:42	Log-Likelihood:	-448.99			
No. Observations:	200	AIC:	902.0			
Df Residuals:	198	BIC:	908.6			
Df Model:	1					
Covariance Type:	nonrobust					
	coef	std err	t	P> t	[0.025	0.975]
const	6.9748	0.323	21.624	0.000	6.339	7.611
TV	0.0555	0.002	29.260	0.000	0.052	0.059
Omnibus:	0.013	Durbin-Watson:	2.029			
Prob(Omnibus):	0.993	Jarque-Bera (JB):	0.043			
Skew:	-0.018	Prob(JB):	0.979			
Kurtosis:	2.938	Cond. No.	338.			

Notes:

[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

R^2

P-value

Notatnik:

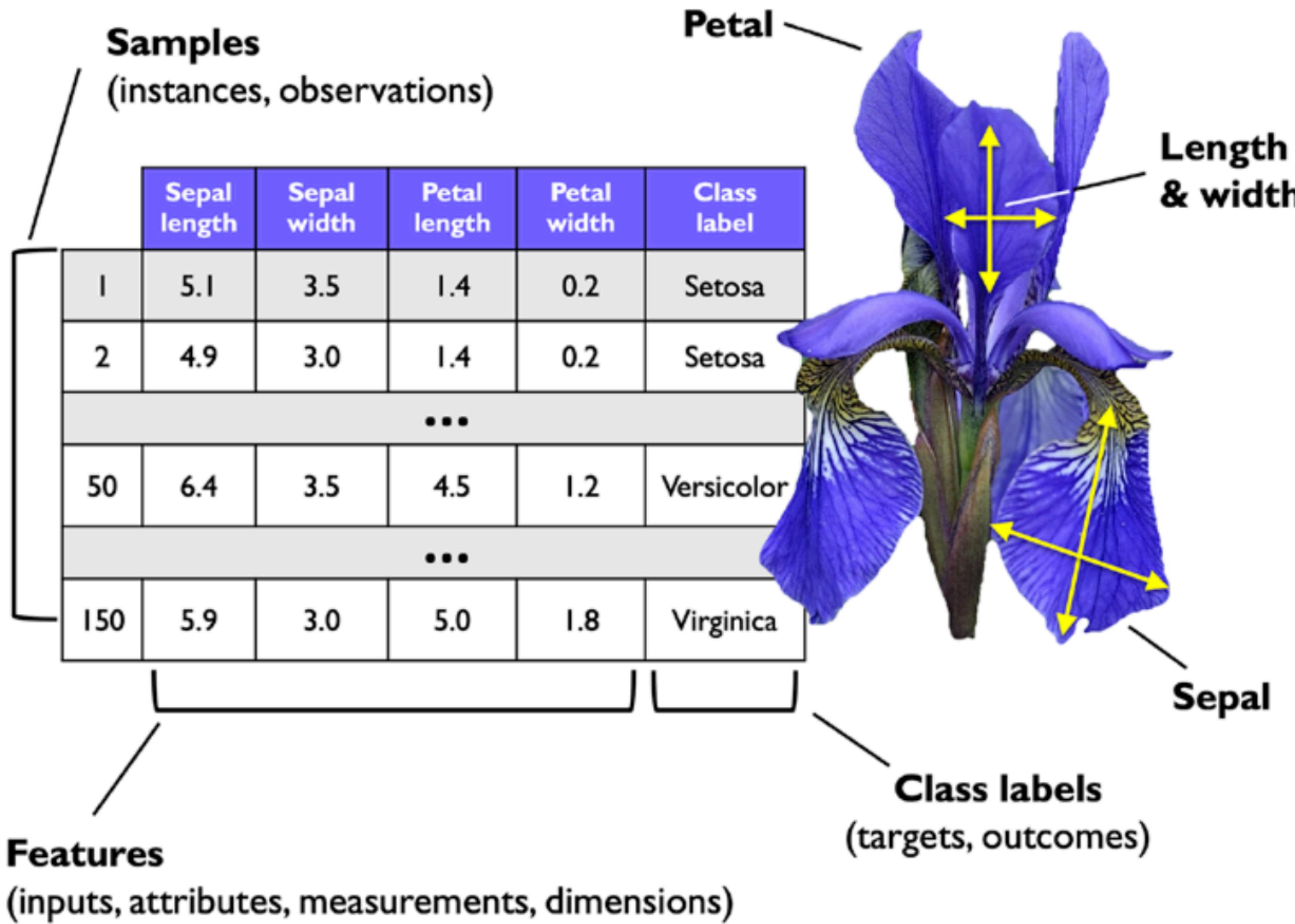
02 - Simple Linear Regression

Regresja logistyczna

To jest algorytm zaliczany do
algorytmów klasyfikacji
(a nie regresji)



Iris dataset

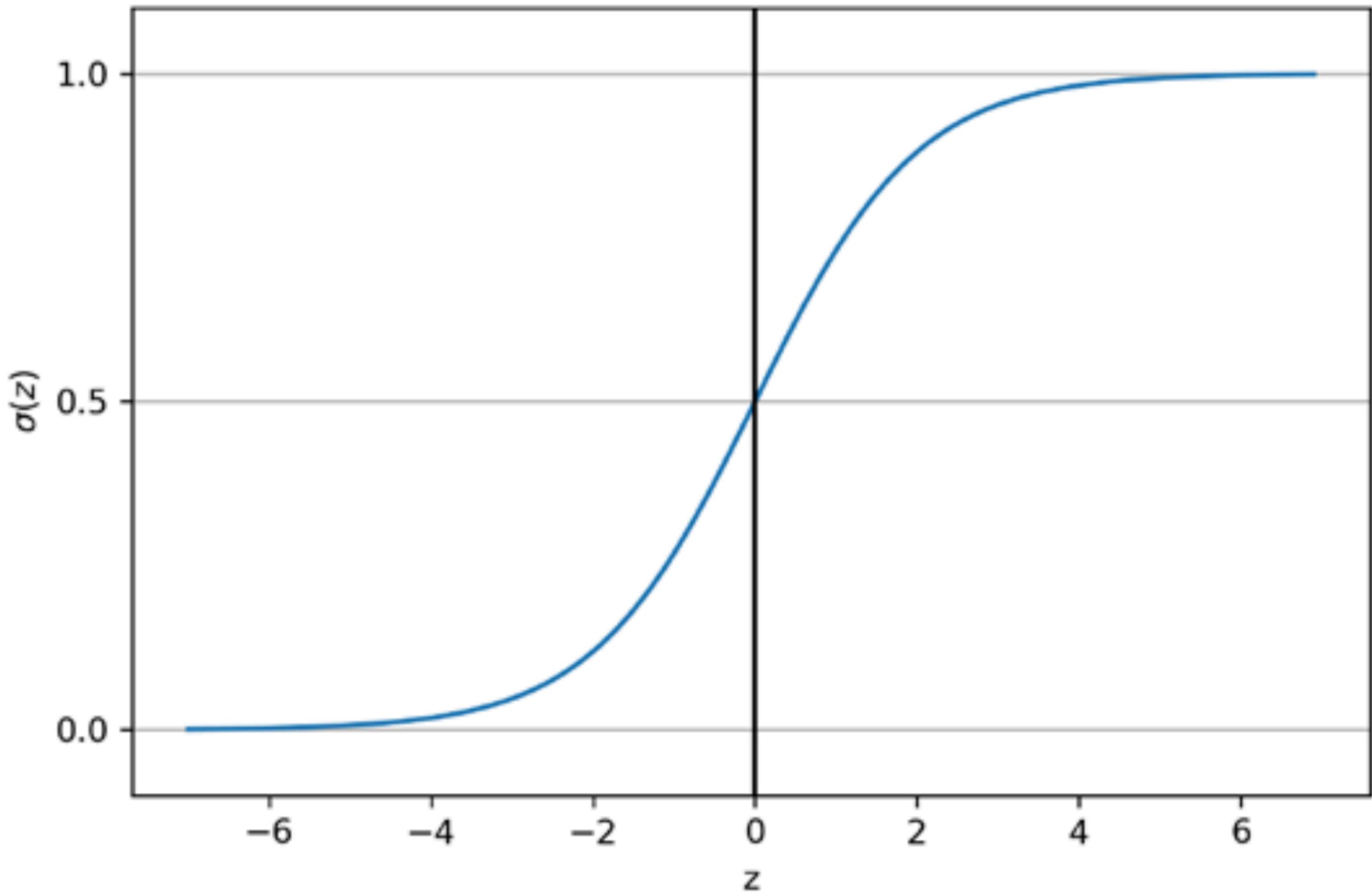


Iris dataset

- Iris setosa - kosaciec szczecinowy
- Iris virginica
- Iris versicolor - kosaciec różnobarwny



Funkcja logistyczna



Szansa zdarzenia (logit function):

$$\frac{p}{(1 - p)}$$

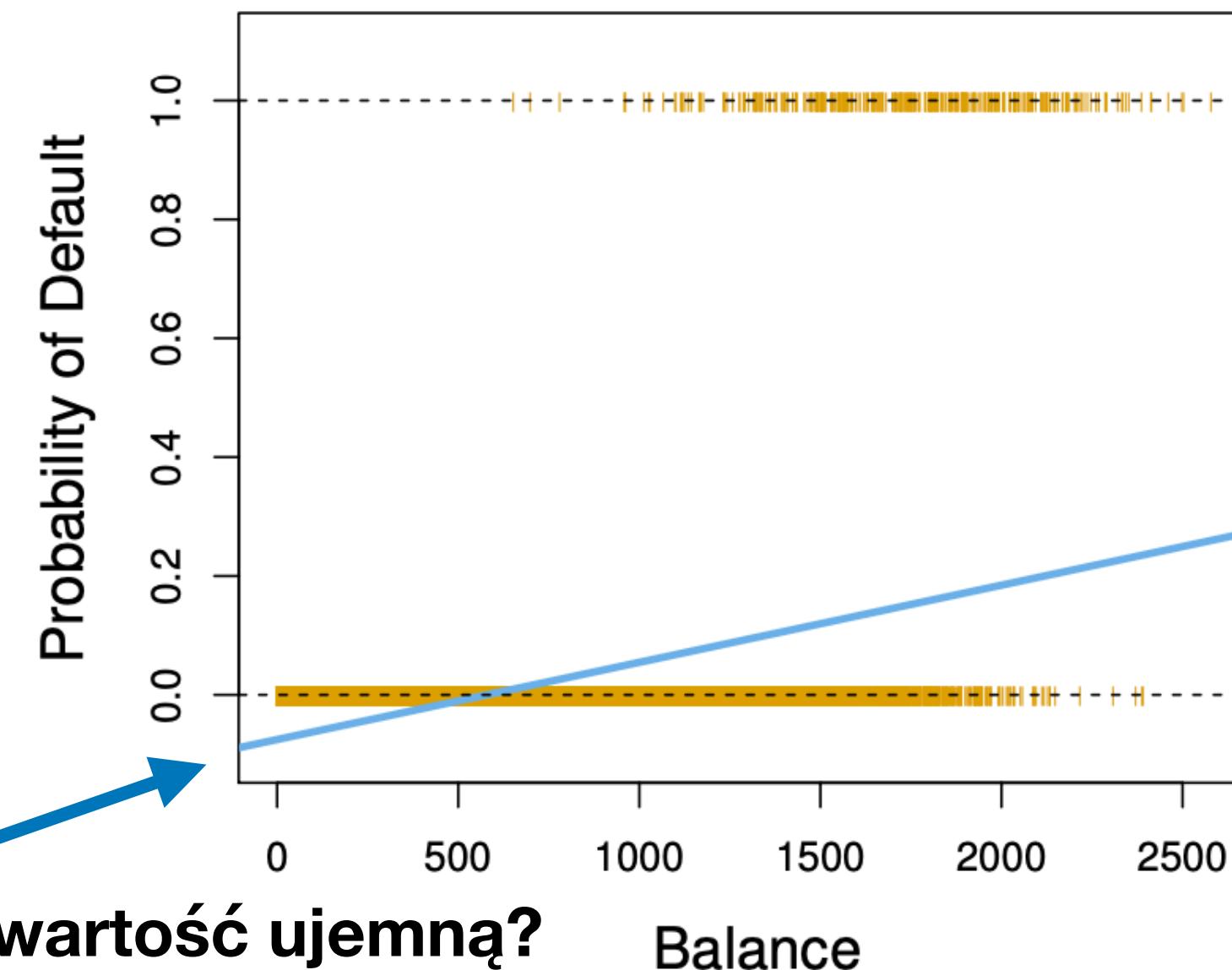
Funkcja logistyczna jest jej odwrotnością:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

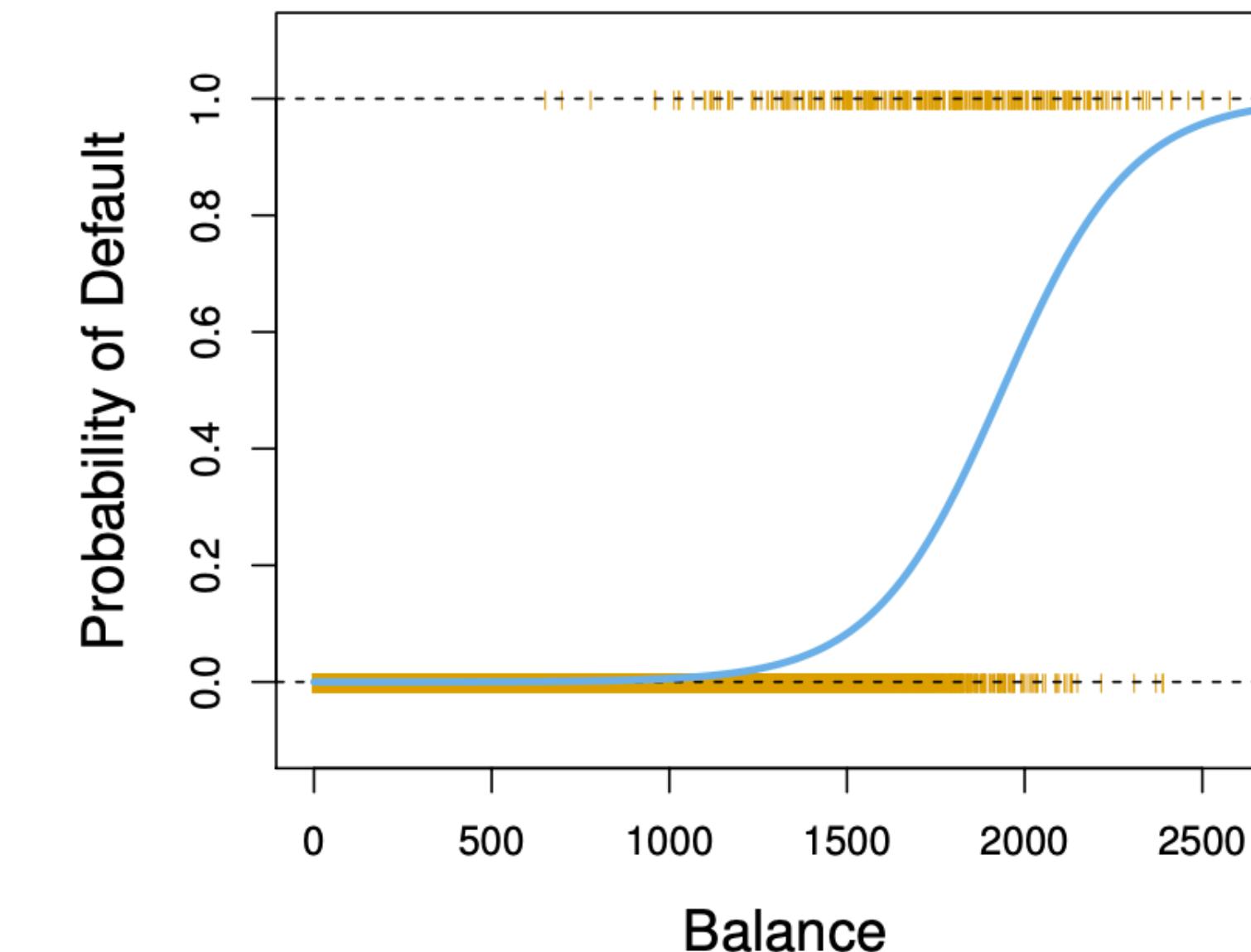
Przyjmuje wartości z przedziału (0,1)

Dlaczego regresja logistyczna?

Regresja liniowa nie ma „sensu” w zadaniach klasyfikacyjnych, gdzie chcemy najlepiej „odseparować” klasy



Jak zinterpretować wartość ujemną?



Klasyfikacja - dane wejściowe

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	Target	
0		5.1	3.5	1.4	0.2	0
1		4.9	3.0	1.4	0.2	0
2		4.7	3.2	1.3	0.2	0
3		4.6	3.1	1.5	0.2	0
4		5.0	3.6	1.4	0.2	0
5		5.4	3.9	1.7	0.4	0
6		4.6	3.4	1.4	0.3	0
7		5.0	3.4	1.5	0.2	0
8		4.4	2.9	1.4	0.2	0
9		4.9	3.1	1.5	0.1	0
10		5.4	3.7	1.5	0.2	0
11		4.8	3.4	1.6	0.2	0
12		4.8	3.0	1.4	0.1	0
13		4.3	3.0	1.1	0.1	0

Notatnik:

03 - Logistic regression

Zbiór uczący i zbiór testowy

Aby w ogóle móc dokonać ewaluacji algorytmów klasyfikujących, musimy podzielić nasz zbiór danych uczących na 2 części:

- Zbiór używany do uczenia algorytmu (train set)
- Zbiór używany do testowania/walidacji (test set)

Np. 80% train set i 20% test set. Trenowanie wykonujemy na zbiorze trenującym a następnie dokonujemy walidacji modelu na zbiorze testowym w celu porównania wyników.

Zbiór uczący i zbiór testowy *

W bardziej zaawansowanych przypadkach trenowania modeli uczenia maszynowego praktykuje się podział zbioru obserwacji na zbiory:

- Uczący
- Walidacyjny
- Testowy

Ma to miejsce w sytuacjach, kiedy dokonujemy ewaluacji modelu w trakcie trenowania. Na przykład dobierając parametry modelu (ang hyper-parameter optimisation), lub decydując kiedy model jest wystarczająco wytrenowany. Ponieważ ostateczna ewaluacja modelu nastąpi z użyciem zbioru testowanego, ważne jest aby nie był on wcześniej „widziany” przez model, *nawet* przy optymalizacji parametrów.

W naszych notatkach będziemy używać tylko zbioru uczącego i testowego.

Zbiór uczący i zbiór testowy

Dlaczego rozdzielamy zbiory?

- Unikamy w ten sposób sytuacji, w której model „widział” już daną obserwację i najprawdopodobniej zaklasyfikuje ją właściwie („nauczy się ich na pamięć”)

Jak wybrać zbiór testowy?

- Najczęściej można wybrać po prostu losowe obserwacje. Czasami jednak chcemy się upewnić, że klasy są jednakowo reprezentowane, wtedy warto o to zadbać zachowując takie same proporcje klas w zbiorze testowym i treningowym (stratified validation)

Sprawdzian krzyżowy - Cross-validation

4-fold validation ($k=4$)



Nie musimy ograniczać się do walidacji tylko jednokrotnej.

ε_1 Dzięki walidacji krzyżowej możemy sprawdzić nasz algorytm na wszystkich przypadkach.

ε_2 Wyniki uzyskane w każdej walidacji uśredniamy.

Ewaluacja



actual predicted



0	2	2
---	---	---

1	0	0
---	---	---

2	1	1
---	---	---

3	0	0
---	---	---

4	0	0
---	---	---

5	0	0
---	---	---

6	2	2
---	---	---

7	2	2
---	---	---

8	2	2
---	---	---

9	1	1
---	---	---

Polega na porównaniu wyniku rzeczywistej obserwacji z wynikiem przewidzianym przez model.

Jak porównywać takie wyniki?

Macierz błędów (confusion matrix)

		Predicted condition	
		Positive (PP)	Negative (PN)
Total population $= P + N$	Positive (P)	True positive (TP)	False negative (FN)
Actual condition	Negative (N)	False positive (FP)	True negative (TN)

precyzja (*ang. precision, positive predictive value*)

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

czułość (*ang. recall, sensitivity, true positive rate*)

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

dokładność (*ang. accuracy*)

$$ACC = \frac{TP+TN}{P+N}$$

F1-score

$$F_1 - score = \frac{2TP}{2TP+FP+FN}$$

Precyzja (precision) - PPV

		Predicted condition	
		Positive (PP)	Negative (PN)
Total population	= P + N	Positive (P)	Negative (N)
Positive (P)	True positive (TP)	False negative (FN)	
Negative (N)	False positive (FP)	True negative (TN)	

$$\text{Precyzja} = \frac{TP}{TP + FP}$$

Intuicja: Ile jest prawdziwych jabłek spośród wszystkich owoców, które zaklasyfikowaliśmy jako jabłka?

Idealna wartość to 100%

Problemy:

A co z prawdziwymi jabłkami które odrzuciliśmy (zaklasyfikowaliśmy jako gruszki?). Algorytm może mieć 100% precyzję ale znajdować bardzo mało jabłek.

Czułość (recall) - TPR

		Predicted condition	
		Positive (PP)	Negative (PN)
Total population = P + N	Positive (P)	True positive (TP)	False negative (FN)
Actual condition	Negative (N)	False positive (FP)	True negative (TN)

$$\text{Czułość} = \frac{TP}{P} = \frac{TP}{TP + FN}$$

Intuicja: Ile znaleźliśmy prawdziwych jabłek spośród wszystkich jabłek?

Idealna wartość to 100%

Problem:

Algorytm może mieć 100% czułość, ale klasyfikować błędnie gruszki jako jabłka. Szczególnie istotny problem gdy ilość gruszek jest bardzo duża w porównaniu do jabłek.

Dokładność (accuracy) - ACC

		Predicted condition	
		Positive (PP)	Negative (PN)
Total population = P + N	Positive (P)	True positive (TP)	False negative (FN)
Actual condition	Negative (N)	False positive (FP)	True negative (TN)

$$\text{Dokładność} = \frac{TP + TN}{P + N}$$

Intuicja: Ile jest dobrze sklasyfikowanych owoców spośród wszystkich owoców. 100% oznacza, że wszystko jest idealnie (nie ma błędów)

Problem:
Wśród 100 owoców jest 5 jabłek, które chcemy odnaleźć. Algorytm może nie odnaleźć żadnego jabłka i wciąż mieć wysoką dokładność (95%).

Inne metryki

- Swoistość (specificity, true negative rate)

$TNR = \frac{TN}{N}$ - to jak czułość ale dla klasy negatywnej. Ważna metryka, gdy zależy nam na pewności, że nasz model dobrze rozpoznaje przypadki negatywne. Ma to duże znaczenie np. w diagnostyce medycznej

- Wskaźnik fałszywych odkryć (false discovery rate)

$FDR = \frac{FP}{FP + TP}$ - odwrotność precyzji, tj $FDR = 1 - PPV$

Macierz błędów (confusion matrix)

		Przewidziane
		Jabłko
		Gruszka
Prawdziwe	Jabłko	12
	Gruszka	15
		10 ("Poprawnie")
		5 ("Fałszywy alarm", błąd pierwszego rodzaju)
		2 ("Nierozpoznany przypadek", błąd drugiego rodzaju)
		10 ("Poprawne odrzucenie")

Przykład: system wykrywania jabłek spośród jabłek i gruszek.

$$\text{Precyzja} = 10 / 15 = 0,66$$

$$\text{Czułość} = 10 / 12 = 0,83$$

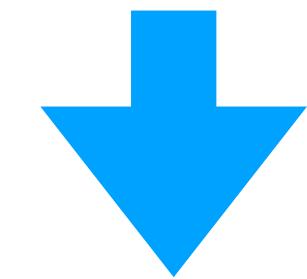
$$\text{Dokładność} = 20 / 27 = 0,74$$

Macierz błędów (confusion matrix)

Przykład z notebooka „04 - Logistic regression”

```
▶ from sklearn.metrics import confusion_matrix  
pd.DataFrame(confusion_matrix(y_test, predictions),  
              index=iris.target_names,  
              columns=iris.target_names)
```

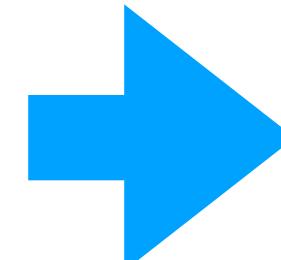
Przewidziane klasy
(Predicted labels)



	setosa	versicolor	virginica
setosa	10	0	0
versicolor	0	10	0
virginica	0	1	9



Rzeczywiste klasy
(true labels)



Ewaluacja modeli wieloklasowych

Przewidziane

		Przewidziane		
		Setosa	Versicolor	Virginica
Prawdziwe	Setosa	10	0	0
	Versicolor	0	10	0
	Virginica	0	1	9

Ewaluacja modeli wieloklasowych

		Przewidziane		
		Setosa	Versicolor	Virginica
Prawdziwe	Setosa	TN 10	FP 0	TN 0
	Versicolor	FN 0	TP 10	FN 0
	Virginica	TN 0	FP 1	TN 9

Dla klasy Versicolor:

$$\text{Precyzja} = \frac{TP}{TP + FP} = \frac{10}{10 + 1} = 0,91$$

$$\text{Czułość} = \frac{TP}{P} = \frac{TP}{TP + FN} = \frac{10}{10} = 1$$

$$\text{Dokładność} = \frac{TP + TN}{P + N} = \frac{29}{30} = 0,97$$

Notatnik:

03 - Logistic regression