

# Wykorzystanie bibliotek Pythona do analizy i przetwarzania danych

Sebastian Słomian



# Data Science

Interdyscyplinarna dziedzina, na którą składają się: matematyka, statystyka, sztuczna inteligencja oraz inżynieria komputerowa. Służąca do wydobywania wartościowych informacji dla biznesu.

Elementy Data Science:

- zbieranie danych
- czyszczenie danych
- analiza danych
- modelowanie
- wizualizacja

# Biblioteki



# NumPy

<https://numpy.org/>

Podstawowe narzędzie do przeprowadzania obliczeń numerycznych w Pythonie.  
Głównym elementem biblioteki są wielowymiarowe tablice **ndarray**.

- tablice NumPy są szybsze i wydajniejsze od struktur danych Pythona
- zawiera funkcje do wykonywania operacji na elementach tablicy oraz przeprowadzania operacji matematycznych
- narzędzia do odczytu i zapisu zbiorów danych opartych na macierzach
- funkcje operacji algebry liniowej, transformacja Fouriera i generatory liczb losowych
- wiele narzędzi stosuje tablice NumPy jako główną strukturę danych lub są w pełni z nimi kompatybilne

# NumPy

Import biblioteki NumPy

```
import numpy as np
```

Tworzenie tablicy NumPy

```
arr = np.array([1, 2, 3])
arr2d = np.array([[1, 2, 3], [4, 5, 6]])
```

Wymiar tablicy

```
arr.ndim
```

Kształt tablicy

```
arr.shape
```

Typ danych w tablicy

```
arr.dtype
```

# NumPy

```
a = np.array([1, 2, 3])
b = np.arr([3, 4, 5])
```

## Działania matematyczne na tablicach

```
# tablica <działanie> tablica
a + b
a - b
a * b
a / c
# tablica <działanie> wartość
a + 2
a - 2
a * 2
a / 2
```

## Dostęp do elementu tablicy

```
# a[index_elementu]
a[1]
# wycinek
a[1:2]
```

# pandas

<https://pandas.pydata.org/>

Biblioteka wykorzystywana do pracy ze zbiorami danych.

Zawiera wysokopoziomowe funkcje służące do analizy, oczyszczania, eksploracji i przetwarzania danych.

Podstawowe właściwości:

- zawiera szybkie i wydajne struktury DataFrame do przekształcania danych
- funkcje ułatwiające obsługę brakujących danych
- Inteligentne wycinki, indeksowanie specjalne oraz podzbiory oparte na etykietach
- łatwe dodawanie i usuwanie kolumn danych
- agregacja i transformowanie danych oparte na grupowaniu
- wysoko wydajne metody łączenia danych

Import biblioteki pandas

```
import pandas as pd
```

## pandas - Series

Jednowymiarowe obiekty przypominające pojedyńczą kolumnę tablicy. Składa się z sekwencji wartości oraz związanych etykiet pełniących funkcję indeksów.

### Tworzenie obiektu Series

```
s = pd.Series([1, 2, 3])
```

### Series z etykietą

```
s = pd.Series([1, 2, 3], index=['a', 'b', 'c'])
```

### Dostęp do elementu

```
# index elementu
s[0]
# etykieta elementu
s['a']
```

# pandas - DataFrame

Jest to tabela na dane. Zawiera uporządkowany zbiór kolumn, a w każdej z kolumn może znaleźć się wartość innego typu. Pomimo że może wydawać się, że jest to tabela dwu wymiarowa, DataFrame może posiadać wiele wymiarów. Pojedyńcza kolumna jest obiektem Series.

## Tworzenie obiektu DataFrame

```
df = pd.DataFrame({'A': [1, 2, 3]})
```

## DataFrame z etykietą

```
df = pd.DataFrame({'A': [1, 2, 3]}, index=['w1', 'w2', 'w3'])
```

## Dostęp do kolumny

```
# nazwa kolumny  
df['A']
```

## Dostęp do elementu

```
df['A', 'w1']
```

# Matplotlib

<https://matplotlib.org/>

Najpopularniejsza biblioteka Pythona do wizualizacji danych. Świecznie integruje się z innymi narzędziami. Na bazie Matplotlib powstały inne ciekawe narzędzia do wizualizacji danych.

- wspiera wiele typów wykresów
- zaawansowane opcje dostosowania wyglądu
- początkowo może wydawać się trudny do wykorzystania
- bardziej zaawansowane opcje wymagają znacznej ilości kodu

Import biblioteki Matplotlib

```
import matplotlib.pyplot as plt
```

Podstawowy wykres liniowy

```
plt.plot([0, 1], [0, 1])
```

## seaborn

<https://seaborn.pydata.org/>

Biblioteka do wizualizacji danych zbudowana na bazie biblioteki Matplotlib. Zapewnia wysokopoziomowy interfejs do tworzenia wizualizacji danych.

- łatwy i przyjazny interfejs do tworzenia wizualizacji
- wspiera wiele typów wykresów
- wiele palet kolorów do wykorzystania w wykresach
- mniejsza ilość opcji dostosowania wyglądu
- brak interaktywnych wizualizacji

Import biblioteki Matplotlib

```
import seaborn as sns
```

# Uczenie Maszynowe

Machine Learning

## Uczenie maszynowe

Jest to dziedzina nauki programowania komputerów w sposób umożliwiający im uczenie się z danych.

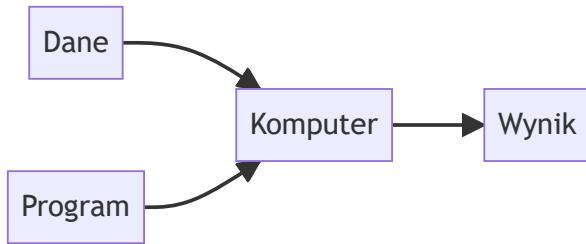
Uczenie maszynowe to dziedzina nauki dająca komputerom możliwość uczenia się bez konieczności ich jawnego programowania.

– Arthur Samuel

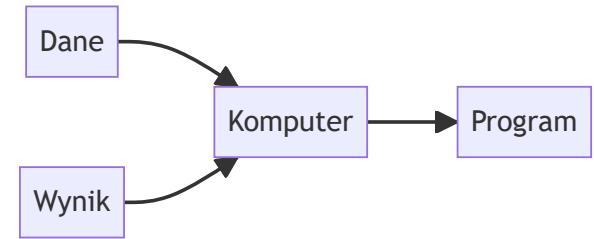


# Programowanie / Uczenie maszynowe

Tradycyjny sposób programowania



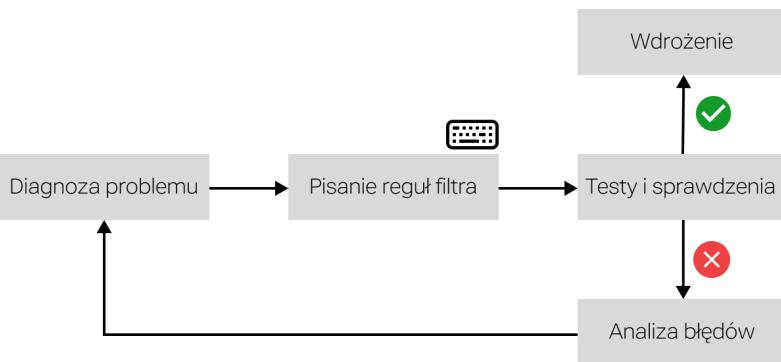
Uczenie maszynowe



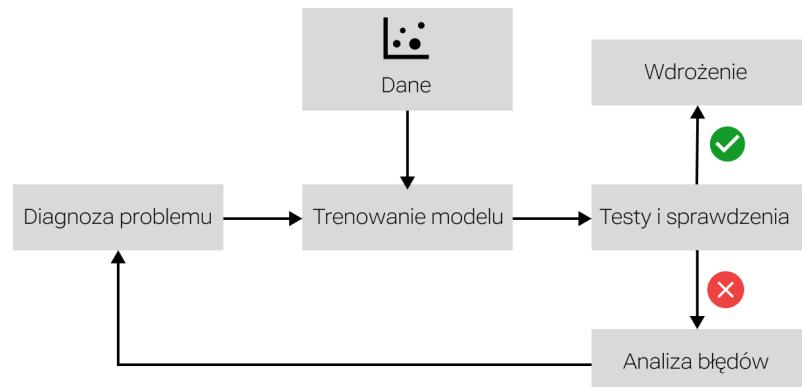
# Programowanie / Uczenie maszynowe

## Filtr spamu

### Tradycyjny sposób programowania



### Uczenie maszynowe



# Uczenie maszynowe - etapy procesu

## 1. Diagnoza problemu

*Zdefiniowanie celu biznesowego oraz zrozumienie jak uczenie maszynowe może pomóc*

*Wybór odpowiedniego typu uczenia maszynowego*

## 2. Zbieranie danych

*Gromadzenie danych z odpowiednich źródeł*

*Zapewnienie wystarczającej ilości i jakości danych*

## 3. Przygotowanie danych

*Czyszczenie danych: usuwanie wartości odstających, brakujących danych oraz błędnych*

*Analiza danych: zrozumienie struktury i relacji danych*

*Transformacja cech: skalowanie, kodowanie kategoryalne*

## 4. Wybór modelu

*Na podstawie typu problemu oraz danych wybór odpowiedniego algorytmu uczenia maszynowego*

## 5. Trenowanie modelu

*Podział na zbiory treningowe i testowe*

*Nauka modelu na zbiorze treningowym*

*Optymalizacja parametrów modelu*

# Uczenie maszynowe - etapy procesu

## 6. Ewaluacja modelu

*Sprawdzenie i ocena modelu na wcześniej niewidzianych danych - zbiór testowy*

## 7. Optymalizacja i strojenie

*W razie potrzeby dostosowanie cech, parametrów lub zmiana modelu*

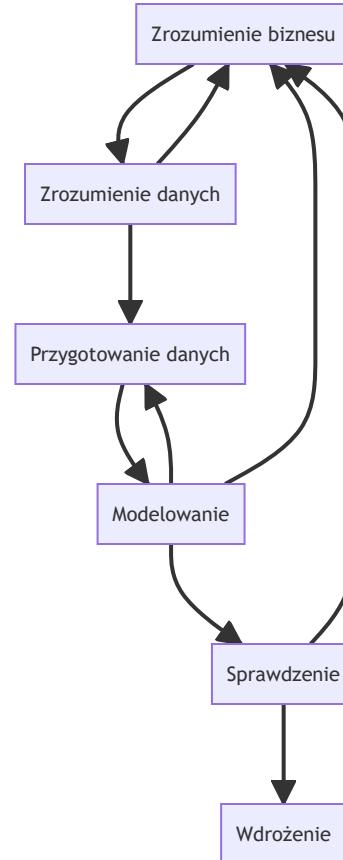
## 8. Wdrożenie

*Produkcyjne wdrożenie modelu*

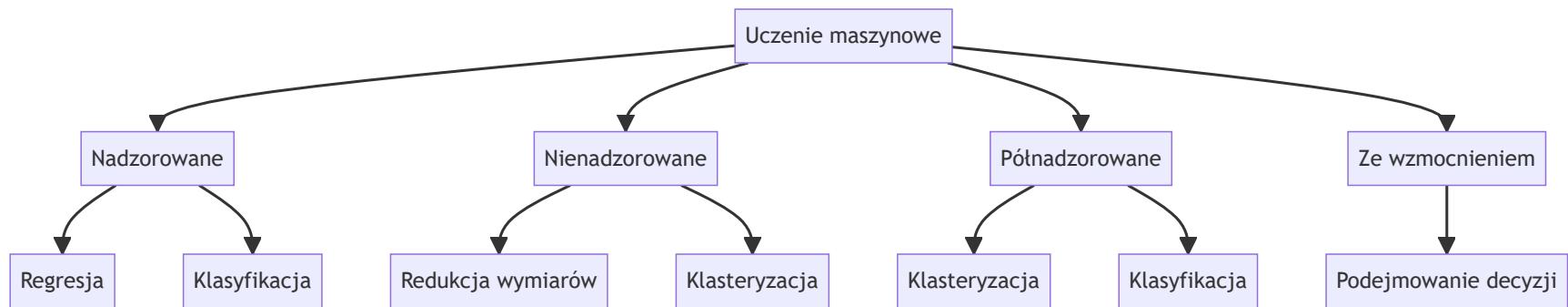
# CRISP-DM

Cross-Industry Standard Process for Data Mining umożliwia eksplorację danych zgodnie ze sprawdzonymi procedurami branżowymi.

- metodologia udostępniająca opisy typowych faz projektu i zadań realizowanych w każdej fazie
- model procesu zawierający przegląd cyklu życia procesu eksploracji danych

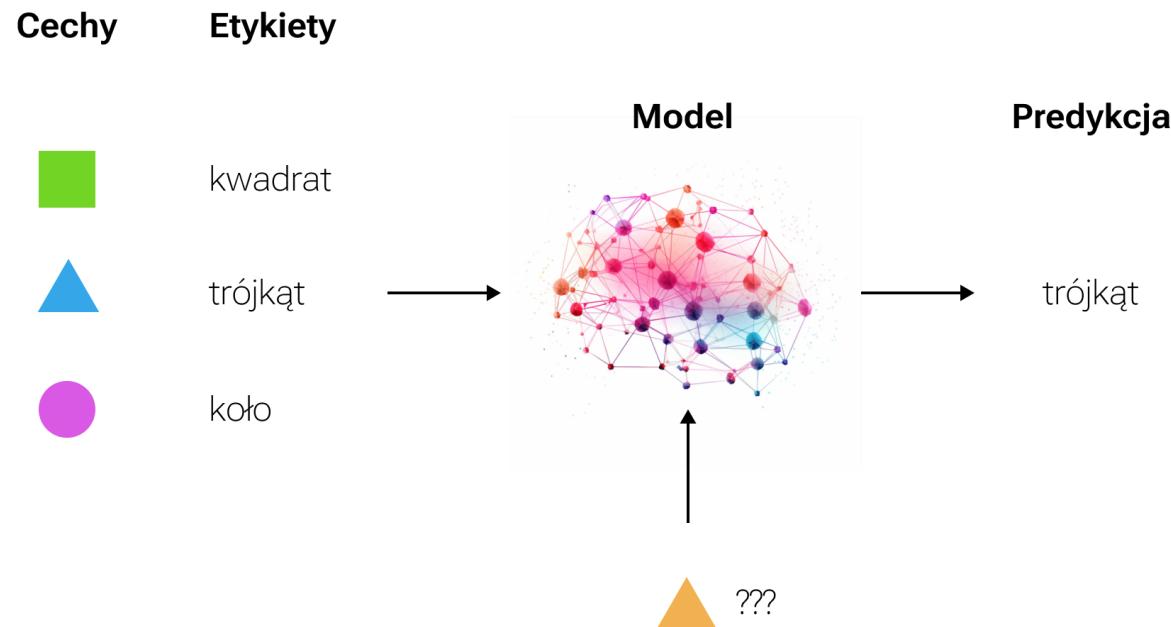


## Uczenie maszynowe - podział



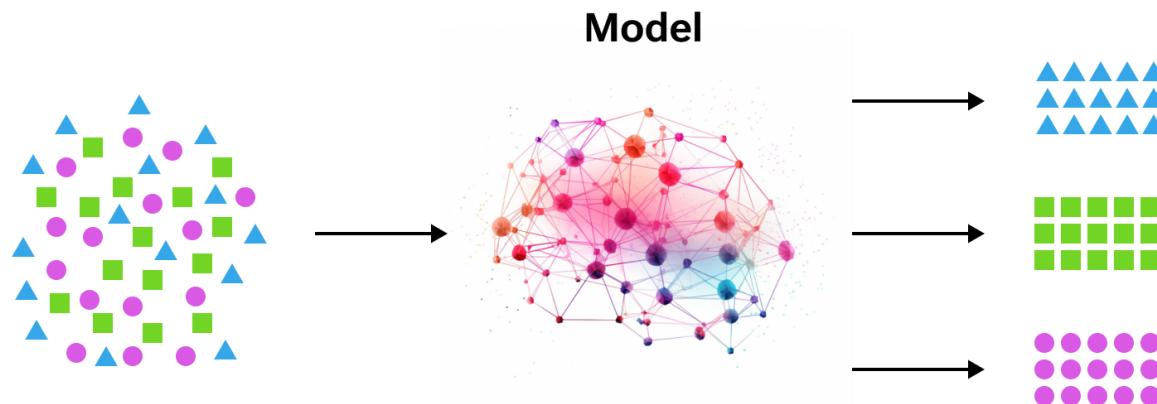
## Uczenie nadzorowane

Dane uczące przekazywane algorytmowi zawierają dołączone rozwiązanie problemu tzw. etykiety. Celem jest nauka mapowania między cechami a odpowiedziami w taki sposób, żeby model był w stanie przewidywać odpowiedzi dla nowych, nieznanych danych.



# Uczenie nienadzorowane

Dane uczące są nieoznakowane. Celem jest znalezienie struktury lub wzorców w samych danych. Typowym zadanie uczenia nienadzorowanego jest klasteryzacja (grupowanie podobnych danych razem).

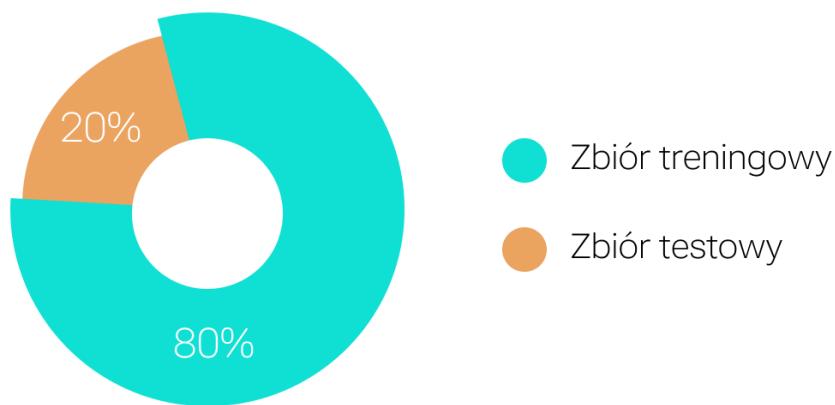


## Zbiór treningowy i testowy

Zbiór treningowy - podzbiór dostępnych danych, które są używane do trenowania modelu uczenia maszynowego.

Zbiór testowy - podzbiór dostępnych danych, które NIE są używane do trenowania modelu, natomiast są wykorzystywane do oceny wydajności modelu, na wcześniej nieznanych danych.

Stosunek zbioru treningowego do testowego to 80% do 20%, lecz zależy od rozmiaru zestawu danych.



## Cechy i cele/etykiety

Cechy (data) - dane wejściowe przekazywane do modelu uczenia maszynowego w celu przewidywania lub klasyfikacji.

Cele (target) - dane wyjściowe, które model stara się przewidzieć. W **zbiorze treningowym** cele są znymi wartościami, na podstawie których model stara się nauczyć przewidywać. Natomiast w **zbiorze testowym** model jest oceniany na podstawie tego, jak dokładnie może przewidywać te cele z rzeczywistymi wartościami.

Cechy

Przejechane kilometry	Pojemność silnika	Klimatyzacja
1500	1.8	TAK
2500	1.4	NIE

Cel

Zużyte paliwo
50
80

## scikit-learn

<https://scikit-learn.org/>

Biblioteka języka Python zawierająca wiele algorytmów uczenia maszynowego. Ze względu na przystępna strukturę oraz interfejs doskonale nadaje się dla osób rozpoczynających przygodę z uczeniem maszynowym.

- proste i efektywne narzędzie do predykcji rezultatów
- zbudowana na NumPy, matplotlib
- przejrzysta i rozbudowana dokumentacja
- nie najlepszy wybór do bardziej zaawansowanych metod

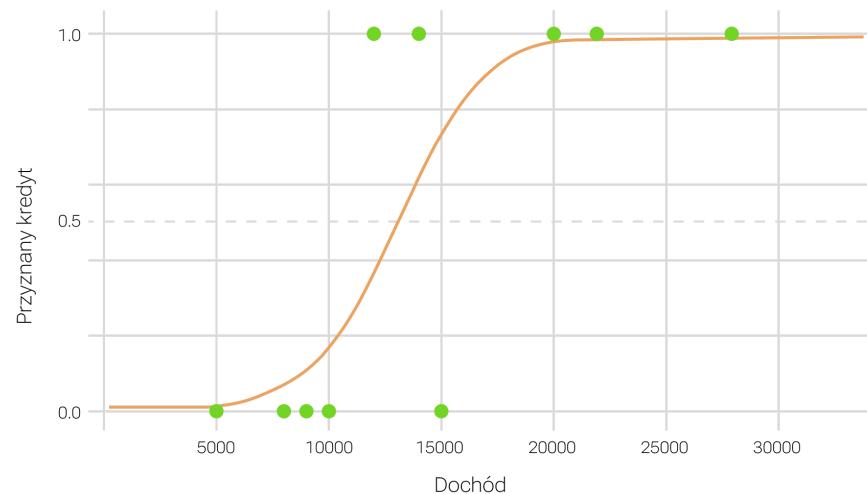
Przykładowy import algorytmu regresji logistycznej

```
from sklearn.linear_model import LogisticRegression
```

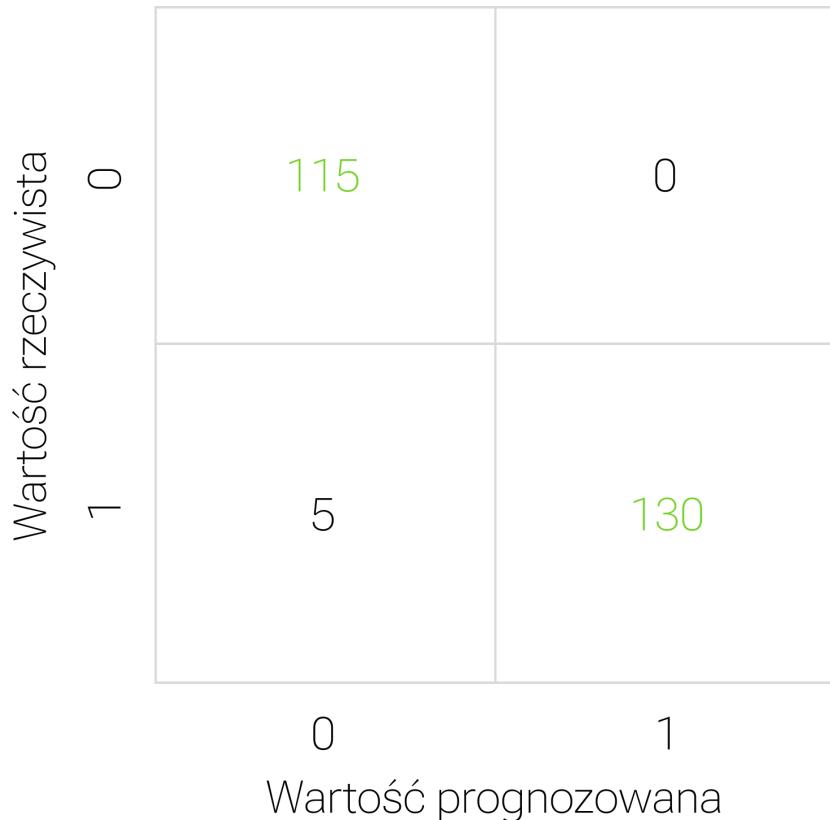
## Klasyfikacja - Regresja Logistyczna

Klasyfikacja jest problemem polegającym na określeniu, do której z zestawu kategorii należy nowo obserwowana próbka.

Służy do szacowania prawdopodobieństwa przynależności do danej klasy. Jeżeli oszacowane prawdopodobieństwo przekracza określoną wartość progową (zazwyczaj 50%), to model prognozuje, że próbka należy do tej właśnie klasy, w innym przypadku orzeka, że próbka nie stanowi tej klasy.



## Macierz pomyłek



Dokładność (accuracy)

*stosunek prawidłowo oznaczonych próbek do przykładów nieprawidłowo sklasyfikowanych*

Precyza (precision)

*odsetek pozytywnych przykładów, które są faktycznie pozytywne*

Czułość (recall)

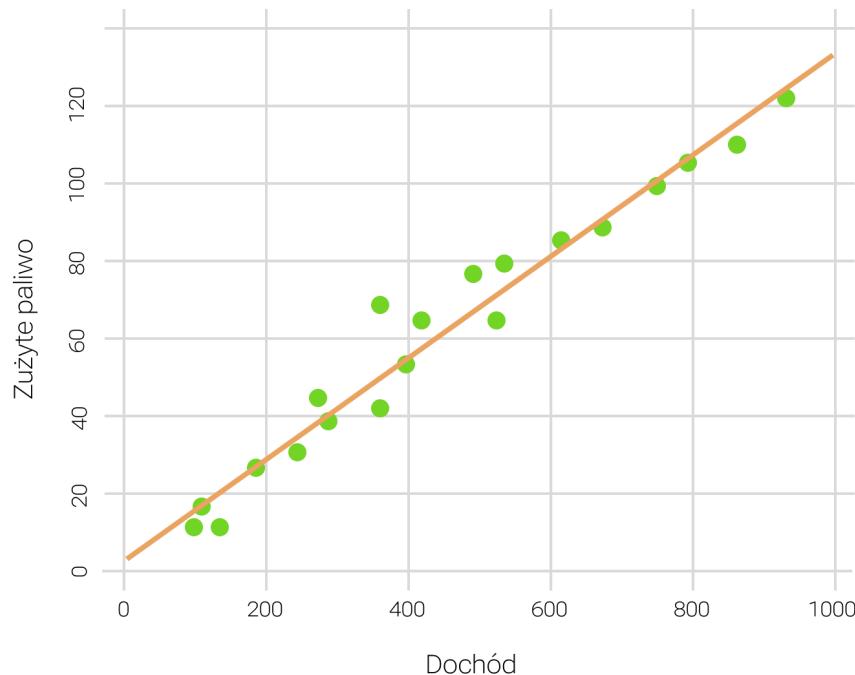
*odsetek pozytywnych przykładów, które zostały prawidłowo rozpoznane przez klasyfikator*

F1

*średnia harmoniczna precyzji i czułości*

## Regresja - Regresja Liniowa

Regresja wykorzystywana jest do przewidywania i prognozowania zachowań. Dzięki regresji możliwe jest wychwytywanie trendów spowodowanych zmianami pewnych cech w populacji.



## Ocena modelu

MAE (Mean Absolute Error) średni błąd bezwzględny

*ile średnio mylimy się w przewidywaniach*

MSE (Mean Squared Error) średni błąd kwadratowy

*średnia kwadratów różnic wartości rzeczywistej z przewidywaną, bardziej karze duże błędy w porównaniu do MAE*

RMSE (Root Mean Square Error) pierwiastek błędu średniokwadratowego

*w jakim stopniu model myli się w przewidywaniach, wynik w identycznych jednostkach co oryginalne dane, pierwiastek z MSE*

R2 współczynnik determinacji

*jak dobrze zmienne niezależne (cechy) przewidują wartość zmiennej zależnej (cel)*

# Problemy uczenia maszynowego

- niedobór danych uczących

*brak wystarczającej ilości danych do skutecznego trenowania modelu*

- niereprezentatywne dane uczące

*dane treningowe nie odzwierciedlają rzeczywistego środowiska, w którym model będzie stosowany*

- dane słabej jakości

*dane zawierają błędy, braki i szумy*

- nieistotne cechy

*dane nie zawierają cech, na podstawie których można prawidłowo przewidzieć wynik*

- przetrenowanie modelu

*model zbyt dokładnie dopasowany do danych treningowych*

- niedotrenowanie modelu

*model zbyt prosty, nie uchwycił istotnych wzorców z danych uczących*

# Lektury

- Wes McKinney

Python w analizie danych. Przetwarzanie danych za pomocą pakietów pandas i NumPy oraz środowiska Jupyter. Wydanie III  
<https://wesmckinney.com/book/>

- Aurélien Géron

Uczenie maszynowe z użyciem Scikit-Learn, Keras i TensorFlow. Wydanie III

- Thomas Nield

Podstawy matematyki w data science. Algebra liniowa, rachunek prawdopodobieństwa i statystyka