

Wprowadzenie do uczenia maszynowego

Igor Wojnicki

February 23, 2022

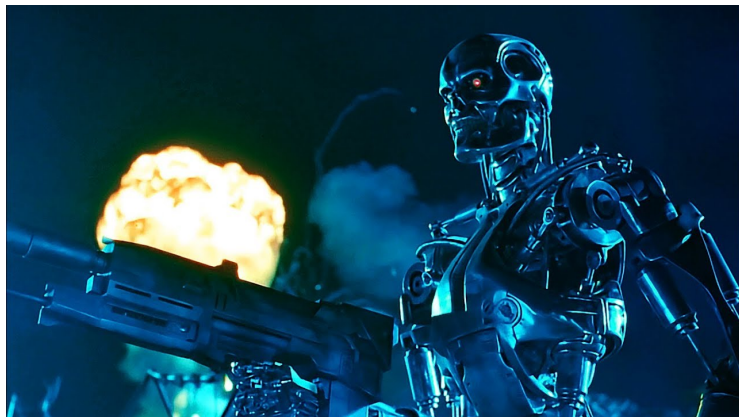
Plan prezentacji

Czym jest Uczenie Maszynowe?

Jak działa Uczenie Maszynowe

Wczesne zastosowania

1



- ▶ OCR
- ▶ spam filter
- ▶ czołgi?

¹Terminator 2: Judgment Day

Referencje

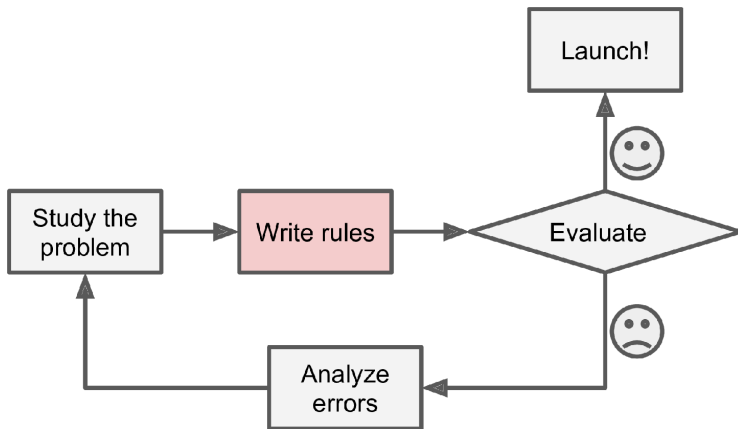
- ▶ Książki
 - ▶ Aurelien Geron, *Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*, O'Reilly
- ▶ Narzędzia <https://www.python.org/>
<https://scikit-learn.org> <https://keras.io/>
<https://www.tensorflow.org/>

Co to?

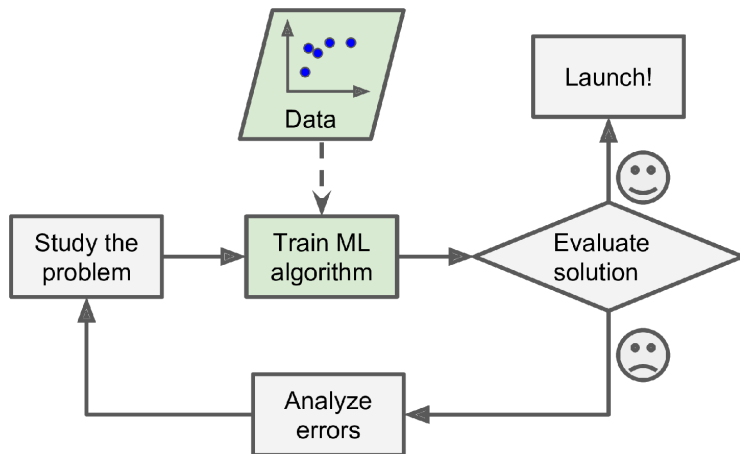
- ▶ ML is the science and art of programming computers so they *learn from data*.
- ▶ ...a field of study that gives computers the ability to learn without being explicitly programmed.
 - Arthur Samuel, 1959
- ▶ A computer program is said to learn from experience E with respect to some task T and some performance measure P , if its performance on T , as measured by P , improves with experience E .
 - Tom Mitchell, 1997

Dlaczego – podejście tradycyjne.

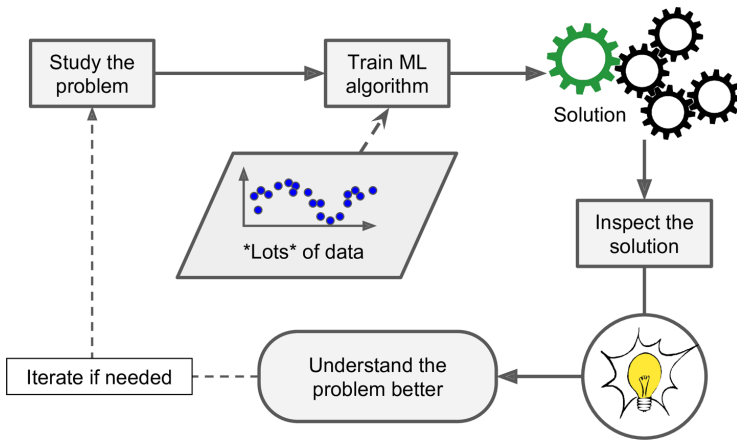
2



Dlaczego – Uczenie Maszynowe



Zalety – odkrywanie wiedzy



Przykłady

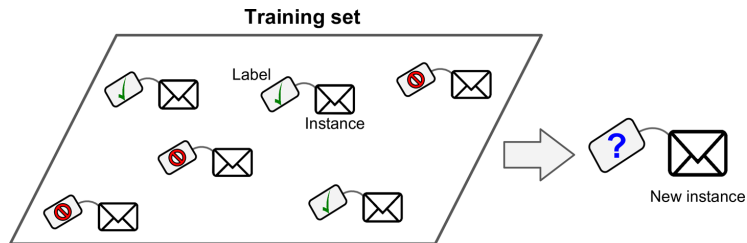
- ▶ klasyfikacja produktów na taśmie produkcyjnej, sieci neuronowe CNN,
- ▶ wykrywanie schorzeń, sieci neuronowe CNN,
- ▶ klasyfikacja tekstów, sieci neuronowe CNN, RNN,
- ▶ streszczenia tekstów, sieci neuronowe CNN, RNN,
- ▶ dialog (asystent głosowy) i rozpoznawanie mowy,
- ▶ przewidywanie np. przychodu w firmie, regresja,
- ▶ wykrywanie oszustw finansowych,
- ▶ segmentacja klientów, klasteryzacja,
- ▶ reprezentacja danych wielowymiarowych w zrozumiały sposób, redukcja wymiarów,
- ▶ rekomendacje produktów, filmów itp.

Rodzaje – ze względu na nadzór

- ▶ nadzorowane – trzeba etykietować zbiór uczący,
- ▶ nienadzorowane – nie trzeba etykietować zbioru uczącego (uczenie bez nauczyciela),
- ▶ częściowo nadzorowane,
- ▶ uczenie przez wzmacnianie (reinforcement).

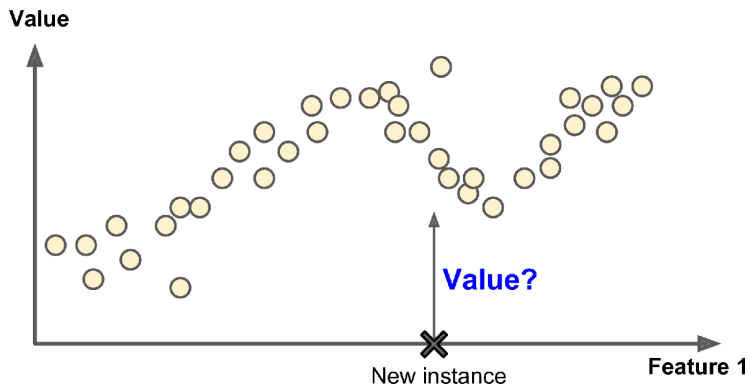
Nadzorowane – klasyfikacja

Czy wiadomość jest spamem czy nie?

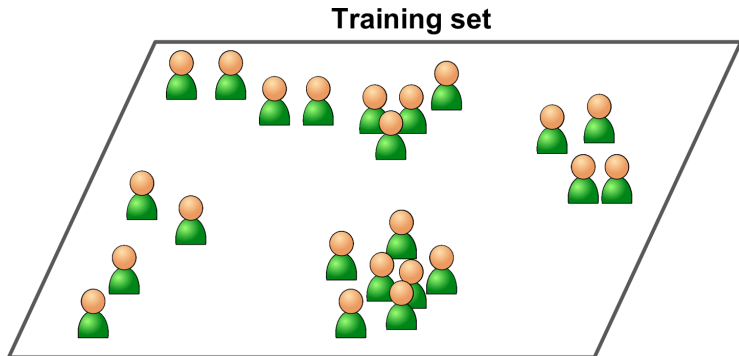


Nadzorowane – regresja

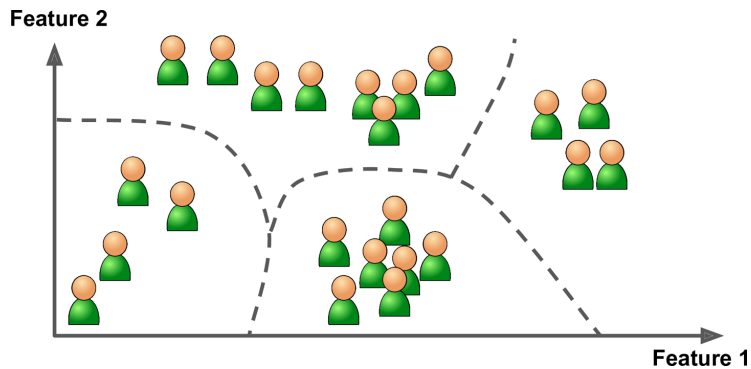
Jaka jest cena samochodu biorąc pod uwagę jego przebieg?



Nienadzorowane – klasteryzacja, przed



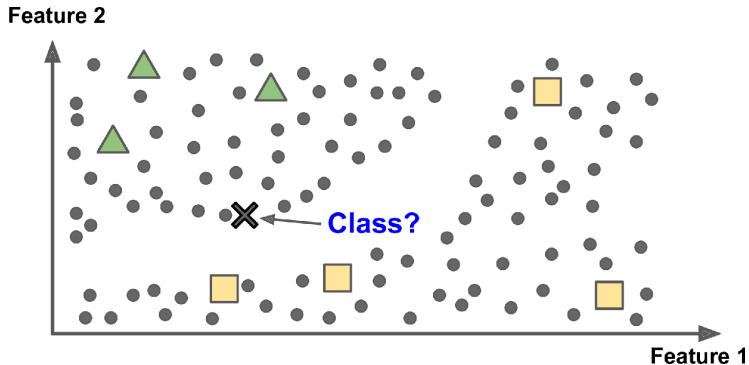
Nienadzorowane – klasteryzacja, po



Nienadzorowane – detekcja anomalii



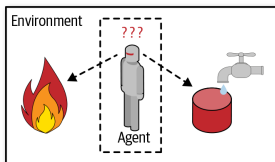
Częściowo nadzorowane



Zwykle połączenie metod nienadzorowanych i nadzorowanych.

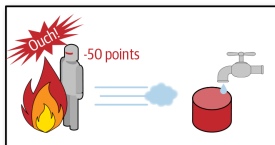
Uczenie przez wzmacnianie

Obserwator jest karany lub nagradzany.



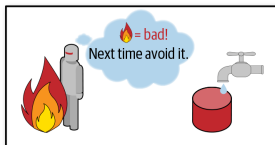
1 Observe

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

Rodzaje ze względu na dostęp do danych

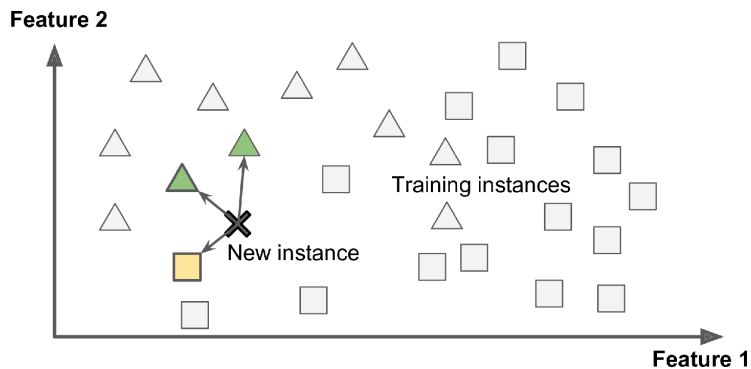
- ▶ batch,
 - ▶ uczenie na podstawie kompletnego zbioru danych,
- ▶ online (mini-batches),
 - ▶ uczenie przyrostowe,
 - ▶ umożliwia out-of-core learning – uczenie gdy dane treningowe nie mieszczą się w pamięci.

Rodzaje ze względu na sposób uogólnienia

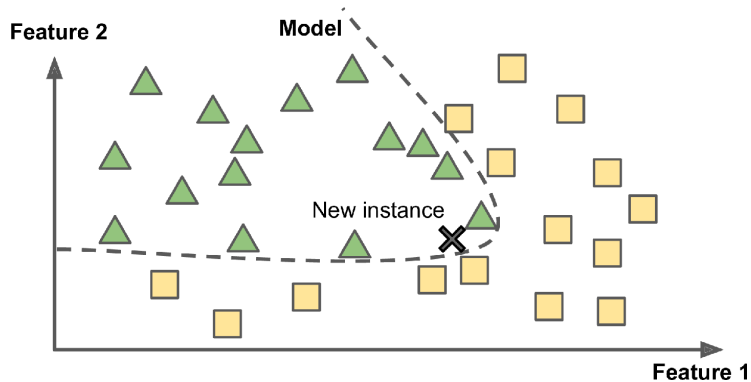
oparte o:

- ▶ instancje,
- ▶ model.

Sposób uogólnienia – instancja



Sposób uogólnienia – model



Wyzwania Uczenia Mszynowego

1. wybrać odpowiedni uczący,
 - ▶ overfitting (nadmierne dopasowanie / przeuczenie),
 - ▶ underfitting (niedouczenie: nieodpowiedni model dla danych, np. zastosowanie modelu liniowego do nieliniowych danych),
2. wybrać odpowiednie dane do uczenia,
 - ▶ za mało danych,
 - ▶ niereprezentatywne dane,
 - ▶ dane niskiej jakości,
 - ▶ nieistotne cechy/dane,

Testowanie

- ▶ Dane uczące (training set),
- ▶ Dane testujące (test set).
- ▶ Dane uczące -> oszacowanie *błędu uczenia*.
- ▶ Dane testujące -> otrzymamy oszacowanie *błędu generalizacji* (generalization/out-of-sample error).

Błąd uczenia niski, błąd generalizacji wysoki -> overfitting.

Częste proporcje zbiór uczący – zbiór testujący: 80%–20%

Overfitting

- ▶ Model działa dobrze na danych uczących, ale niezbyt dobrze generalizuje (marnie działa na danych testujących).
- ▶ Model zbyt złożony w kontekście danych uczących i ewentualnego szumu.
- ▶ Rozwiązania:
 - ▶ uprościć model (wybierz taki, który ma mniej parametrów np. liniowy zamiast wielomianowy),
 - ▶ zmniejszenie liczby atrybutów,
 - ▶ ograniczenie modelu (o ile algorytm na to pozwala, zwykle jest to tzw. hiperparametr tzn. parametr algorytmu a nie modelu),
 - ▶ zebrać więcej danych,
 - ▶ zmniejszyć szum w danych uczących (naprawić błędy, usunąć *outliers*).

Który z dwóch modeli/algorytmów jest lepszy?

- ▶ Trzeba nauczyć oba.
- ▶ Porównać wyniki.

No Free Lunch Theorem

– David Wolpert, 1996.

- ▶ Nie znając założeń odnośnie danych, nie można powiedzieć, który model/algorytm będzie działał najlepiej.
- ▶ Więc trzeba sprawdzić jak działa każdy z nich.
- ▶ W praktyce należy poczynić założenia odnośnie danych i na tej podstawie wybrać zbiór modeli/algorytmów do sprawdzenia.

Plan prezentacji

Czym jest Uczenie Maszynowe?

Jak działa Uczenie Maszynowe

Etapy projektu ML

1. Zastanów się co masz zrobić.
2. Zdobądź dane.
3. Oglądnij dane; wizualizacja.
4. Przygotuj dane.
5. Wybierz model i go wytrenuj.
6. Popraw parametry modelu (fine tune).
7. Wdrożenie, monitoring, utrzymanie.

Dane

bash

```
zcat data/housing.csv.gz | head -4
```

```
longitude,latitude,housing_median_age,total_rooms,total_bedrooms,
-122.23,37.88,41.0,880.0,129.0,322.0,126.0,8.3252,452600.0,1
-122.22,37.86,21.0,7099.0,1106.0,2401.0,1138.0,8.3014,358500
-122.24,37.85,52.0,1467.0,190.0,496.0,177.0,7.2574,352100.0
```

Kolumny:

longitude, latitude, housing_median_age, total_rooms,
total_bedrooms, population, households, median_income,
median_house_value, ocean_proximity

Dane

```
1 import csv
2 import pandas as pd
3
4 df = pd.read_csv('data/housing.csv.gz')
5 print(df.head())
```

	longitude	latitude	housing_median_age	total_rooms	tot
0	-122.23	37.88	41.0	880.0	
1	-122.22	37.86	21.0	7099.0	
2	-122.24	37.85	52.0	1467.0	
3	-122.25	37.85	52.0	1274.0	
4	-122.25	37.85	52.0	1627.0	

Dane

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 20640 entries, 0 to 20639
```

```
Data columns (total 10 columns):
```

#	Column	Non-Null Count	Dtype
---	-----	-----	-----
0	longitude	20640 non-null	float64
1	latitude	20640 non-null	float64
2	housing_median_age	20640 non-null	float64
3	total_rooms	20640 non-null	float64
4	total_bedrooms	20433 non-null	float64
5	population	20640 non-null	float64
6	households	20640 non-null	float64
7	median_income	20640 non-null	float64
8	median_house_value	20640 non-null	float64
9	ocean_proximity	20640 non-null	object

```
dtypes: float64(9), object(1)
```

```
memory usage: 1.6+ MB
```


Dane

```
print(df["ocean_proximity"])
```

```
0      NEAR BAY
```

```
1      NEAR BAY
```

```
2      NEAR BAY
```

```
3      NEAR BAY
```

```
4      NEAR BAY
```

```
...
```

```
20635    INLAND
```

```
20636    INLAND
```

```
20637    INLAND
```

```
20638    INLAND
```

```
20639    INLAND
```

```
Name: ocean_proximity, Length: 20640, dtype: object
```

Dane

```
print(df["ocean_proximity"].value_counts())
```

```
<1H OCEAN      9136
```

```
INLAND         6551
```

```
NEAR OCEAN     2658
```

```
NEAR BAY       2290
```

```
ISLAND          5
```

```
Name: ocean_proximity, dtype: int64
```

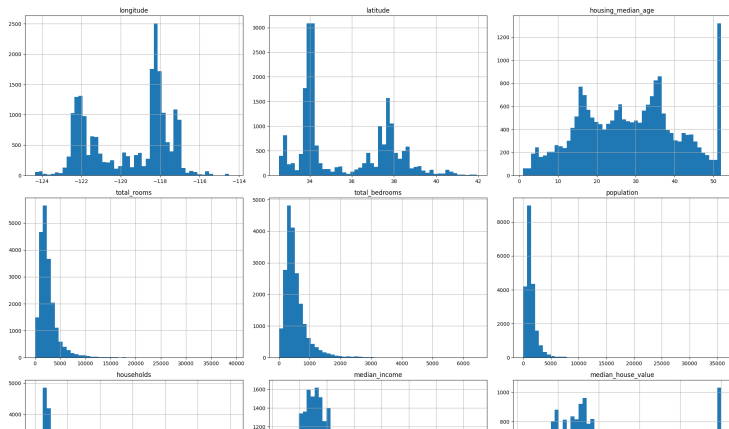
Dane

```
print(df.describe())
```

	longitude	latitude	housing_median_age	total_rooms
count	20640.000000	20640.000000	20640.000000	20640.000000
mean	-119.569704	35.631861	28.639486	263.958939
std	2.003532	2.135952	12.585558	218.961472
min	-124.350000	32.540000	1.000000	2.000000
25%	-121.800000	33.930000	18.000000	144.000000
50%	-118.490000	34.260000	29.000000	212.000000
75%	-118.010000	37.710000	37.000000	314.000000
max	-114.310000	41.950000	52.000000	3932.000000

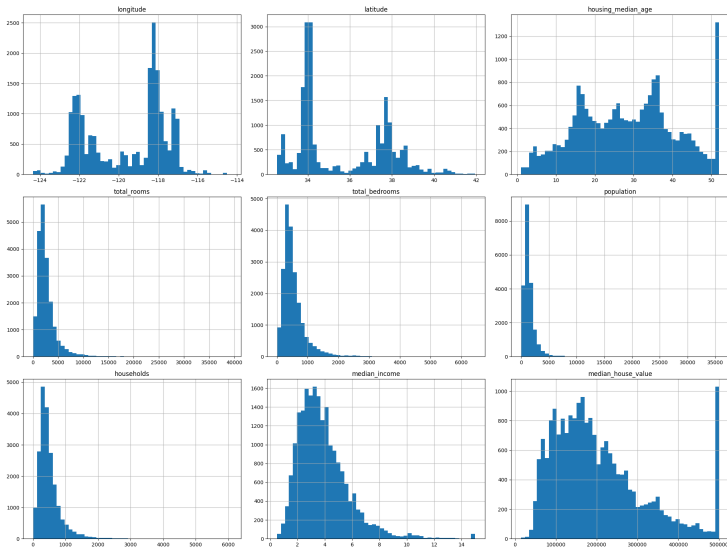
Dane

```
1 import matplotlib.pyplot as plt
2 df.hist(bins=50, figsize=(20,15))
3 f = "housing_hist.png"
4 plt.tight_layout()
5 plt.savefig(f)
6 print(f)
```



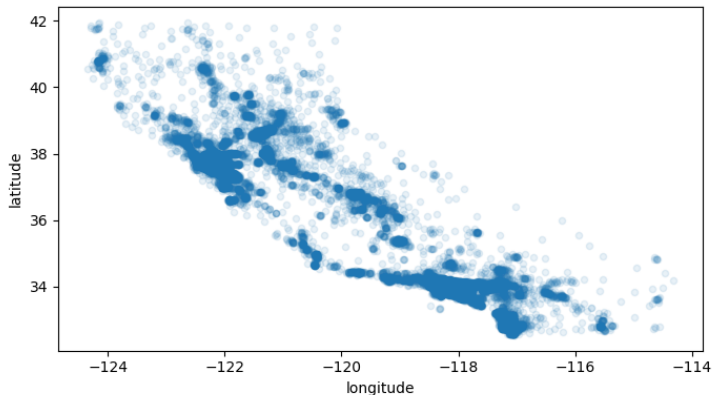
Dane, histogram

median_income – jednostka?, housing_median_age, median_housing_value – odcięte od góry, ogólnie – nie są dzwonowe



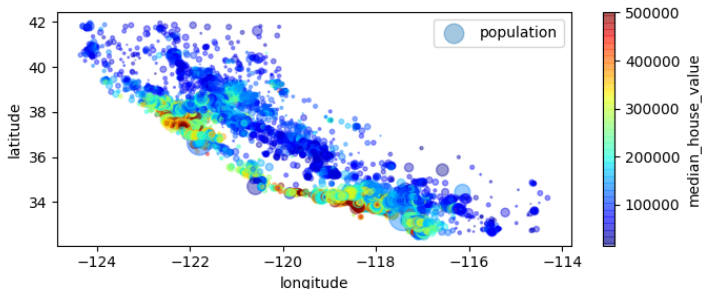
Dane, położenie geograficzne

```
1 df.plot(kind="scatter", x="longitude", y="latitude",  
2         alpha=0.1, figsize=(7,4))  
3 f = "housing_lon_lan.png"  
4 plt.tight_layout()  
5 plt.savefig(f)  
6 print(f)
```



Dane, ceny

```
1 df.plot(kind="scatter", x="longitude", y="latitude",  
2         alpha=0.4, figsize=(7,3), colorbar=True,  
3         s=df["population"]/100, label="population",  
4         c="median_house_value", cmap=plt.get_cmap("jet"))  
5 f = "housing_price.png"  
6 plt.tight_layout()  
7 plt.savefig(f)  
8 print(f)
```



Dane, macierz korelacji

Uwaga: pokazuje jedynie zależności liniowe!

```
print(df.corr()["median_house_value"] .  
      sort_values(ascending=False))
```

```
median_house_value    1.000000  
median_income         0.688075  
total_rooms          0.134153  
housing_median_age    0.105623  
households            0.065843  
total_bedrooms        0.049686  
population            -0.024650  
longitude             -0.045967  
latitude              -0.144160  
Name: median_house_value, dtype: float64
```


Zbiór uczący i testujący

```
1 from sklearn.model_selection import train_test_split
2 train_set, test_set = train_test_split(df,
3                                     test_size=0.2,
4                                     random_state=42)
5 print(len(train_set), len(test_set))
```

16512 4128