# Bootcamp Data Science

Przemysław Spurek

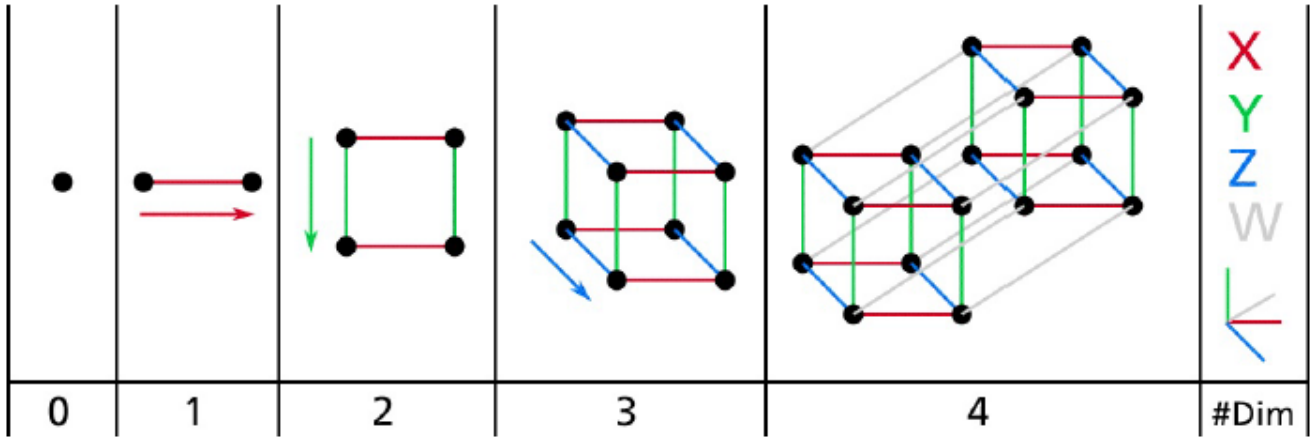
- Wiele problemów pojawiających się w uczeniem maszynowym dotycz danych zawierających tysiące nawet miliony współrzędnych.
- To nie tylko sprawia, że trening jest bardzo wolny, ale może również znacznie utrudnić znalezienie dobrego rozwiązania.
- Ten problem jest często nazywany **przekleństwem wymiarowości** (curse of dimensionality).

- Na szczęście w rzeczywistych problemach często można znacznie zmniejszyć liczbę współrzędnych bez utraty znaczącej informacji.
- Oprócz przyspieszania treningu, redukcja wymiarowości jest również niezwykle przydatna do wizualizacji danych.
- Zmniejszenie liczby wymiarów do dwóch (lub trzech) pozwala na narysowanie wysoko-wymiarowego zbioru danych.

# Curse of Dimensionality

- Jesteśmy tak przyzwyczajeni do życia w trzech wymiarach, że nasza intuicja zawodzi nas, gdy próbujemy wyobrazić sobie wielowymiarową przestrzeń.
- Nawet podstawowy hipersześcian 4D jest niesamowicie trudny do zobrazowania w naszym umyśle, nie mówiąc już o 200-wymiarowej elipsoidzie zanurzonej w 1000-wymiarowej przestrzeni.

# Curse of Dimensionality




Rysunek:

# Curse of Dimensionality

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z01_curse_of_dimensionality.ipynb

# Curse of Dimensionality

- Okazuje się, że wiele rzeczy zachowuje się bardzo różnie w przestrzeni wielowymiarowej. Na przykład, jeśli wybierzesz losowy punkt w kwadracie jednostkowym (kwadratu $1 \times 1$), będzie on miał tylko około 0,4% szansy na to, że znajdzie się on w odległości mniejszej niż 0,001 od granicy.
- Ale w 10 000-wymiarowym jednostkowym hiperześcianie (kostka $1 \times 1 \times \ldots \times 1$) prawdopodobieństwo to jest większe niż 99,999999%.
- Większość punktów wysoko-wymiarowego hiperześcianu znajduje się bardzo blisko granicy.

# Operacje na macierzach - intuicja

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z02_matrix.ipynb

# Operacje na macierzach - intuicja

Rozkład SVD to rozkład na macierze $U, S, V$. Powstałe macierz pozwalają na rekonstrukcję oryginalnej macierzy w następujący sposób:

$$A = USV^T$$

gdzie $S$ to macierz diagonalna z wyrazami $s$ na przekątnej. Wymiar $S$ jest taki jak wymiar macierzy $A$.
Zobacz: `https://en.wikipedia.org/wiki/Singular-value_decompositiondlailustracji.`

# Co robi SVD

Wyrazy $s$ to wartości singularne i są one powiązane z wartościami własnymi macierzy kowaiancji, a kolumny $V$ to wektory własne.
Żeby zrozumieć co robi SVD, trzeba zdać sobie sprawę z tego że macierz to odwzorowanie liniowe. Każde odwzorowanie liniowe można przedstawić w postaci złożenia 3 odwzorowań:

- obrotu,
- skalowania,
- obrotu

to jest właśnie rozkład SVD.
`https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z03_SVD.ipynb`

# Redukcja wymiarowości

- Mówiliśmy, że oryginalna macierz może zostać odtworzona z rozkładu SVD.
- Ciekawsze z naszego punktu widzenia jest to, że dane można zapisać używając mniejszej ilości komponentów, czyli skompresować, a później odtworzyć w sposób stratny.

Dokładniej bierzemy dane wymiaru $D$ i chcemy zredukować do wymiary $d < D$. Rozkład SVD daje nam przepis jak wrócić z wymiaru $d$ do wymiaru $D$ (ale stratnie).
Mianowicie trzeba wziąć:

- $d$ wierszy $V^T$
- podmacierz $S$ wymiaru $d$ na $d$
- $d$ kolumn $U$

i pomnożyć ze sobą.
`https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z04_SVD_dimensional_reduction.ipynb`

# Redukcja wymiarowości

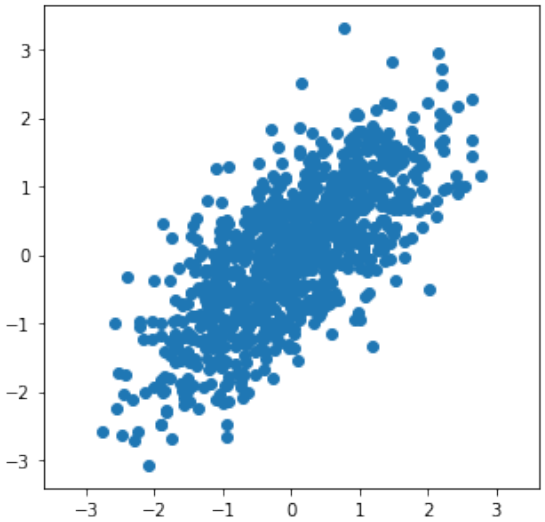
https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z05_noise_reduction.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z06_compression.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z07_NLP.ipynb

# PCA

Pytamy się, które współrzędne są najważniejsze - opisują najwięcej informacji o naszych danych.




Pierwszym krokiem jest normalizacja danych. Dokonujemy tego w dwóch krokach:

- przesuwamy dane do środka układu współrzędnych
- normalizujemy dane (dzielimy każdą współrzędną przez średnią długość wszyskich punktów)

# PCA

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z08_PCA_introduction.ipynb

# PCA

Aby wyznaczyć kierunki decydujące o kształcie naszych danych, należy policzyć wektory i wartości wałasne z macierzy kowariancji.

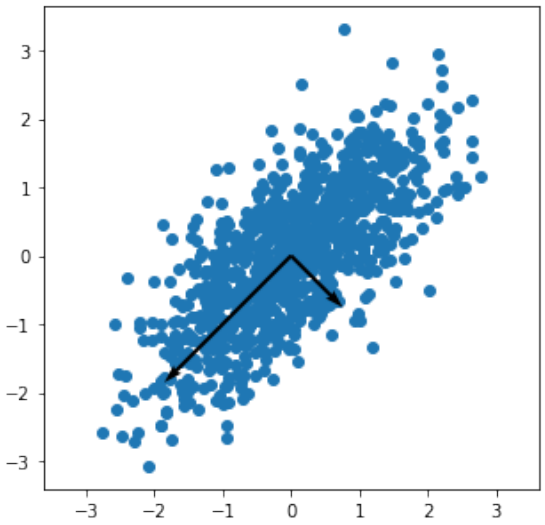
## Definition

Dla macierzy kwadratowej $A$, wektor własny $v$ i wartość własna $\lambda$ spełnia:

$$Av = \lambda v$$

# PCA

Zilustrujemy powyższe wielkości na przykładzie.




Przedstawiliśmy macierz kowariancji $\Sigma$ w postaci iloczynu:

$$\Sigma = VSV^T$$

gdzie $V$ to macierz zawierająca na kolumnach wektory własne, a $S$ to macierz diagonalna, która na przekątnej ma wartości własne.

# PCA

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z09_PCA_visualization.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z10_PCA_visualization.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z11_PCA_visualization.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z12_visualization.ipynb

# PCA

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z14_PCA_linear_regression.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z15.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z16_LogisticRegression.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z17_LogisticRegression.ipynb

# PCA

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z18.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z19_class.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z19_clustering.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z20_clustering.ipynb

# PCA

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z21_incremental_PCA.ipynb

https://github.com/przem85/bootcamp/blob/master/dimensional_reduction/Z22_randomized_PCA.ipynb