Wprowadzenie do uczenia maszynowego część I

Franciszek Górski 2021

GRADIENT
KOŁO NAUKOWE

Czym jest uczenie maszynowe?

- Uczenie maszynowe - proces poprawy wyników algorytmu wraz ze zdobywanym doświadczeniem, które to doświadczenie pozyskiwane jest z danych.
- Proces pozyskiwania doświadczenia nazywany jest uczeniem się.
- W czasie uczenia tworzony jest model na podstawie danych uczących.

GRADIENT
KOŁO NAUKOWE

Dane uczące

Jeżeli model ma zdobywać doświadczenie z danych, to dane muszą zawierać informację. Dlatego w Informatyce dane przedstawiane są w postaci zbioru parametrów np. - dane opisujące różne komputery

Pamięć RAM	Pamięć VRAM GPU	Liczba rdzeni CPU	Taktowanie CPU
8	0	4	2,4
16	6	8	2,5
8	2	8	2,3

GRADIENT
KOŁO NAUKOWE

Etykiety danych

Przedstawione dane mają z kolei przypisane etykiety. Załóżmy, że prezentowany przez nas zbiór ma służyć opisaniu cen komputerów w zależności od ich parametrów:

Pamięć RAM	Pamięć VRAM GPU	Liczba rdzeni CPU	Taktowanie CPU	Cena
8	0	4	2.4	1500
16	6	8	2.5	3000
8	2	8	2.3	1900

GRADIENT
KOŁO NAUKOWE

Dane uczące

Pamięć RAM	Pamięć VRAM GPU	Liczba rdzeni CPU	Taktowanie CPU
8	0	4	2.4
16	6	8	2.5
8	2	8	2.3

= X

Cena
1500
3000
1900

= y

X - macierz liczb rzeczywistych o wymiarach 3x4 (3 przykłady danych, każdy opisany 4 parametrami)

y - wektor liczb rzeczywistych 3x1

GRADIENT
KOŁO NAUKOWE

Dane uczące - nasze uproszczenie

Pamięć RAM
8
16
8

$= X$

Cena
1500
3000
1900

$= y$

X - macierz (wektor) liczb rzeczywistych o wymiarach 3x1 na nasze potrzeby załóżmy 1 parametr opisujący dane

y - wektor liczb rzeczywistych 3x1

Hipoteza h(x)

Przykładowa hipoteza dla naszego przypadku - cena laptopów na rynku:

$$h(x) = a*x1 + b$$

Nasze dane:

X[0] = [8]

X[1] = [16]

X[2] = [8]

h(X[0]) = 8a + b

h(X[1]) = 16a + b

h(X[2]) = 8a + b

GRADIENT
KOŁO NAUKOWE

Inicjalizacja wag w hipotezie h(x)

Nasza hipoteza: *h(x) = a*x1 + b*

Nasze wyniki:

h(X[0]) = 8a + b

h(X[1]) = 16a + b

h(X[2]) = 8a + b

Etykiety danych:

y[0] = 1500

y[1] = 3000

y[2] = 1900

Inicjalizacja wag - wektor parametrów θ (theta)

θ = [a, b]

θ = [100, 150]

Wyniki predykcji:

y_pred[0] = 800 + 150 = 950

y_pred[1] = 1600 + 150 = 1750

y_pred[2] = 800 + 150 = 950

GRADIENT
KOŁO NAUKOWE

Jak zmierzyć błąd algorytmu? - funkcja kosztu L(y, y_pred)

Nasze wyniki predykcji:

y_pred[0] = 800 + 150 = 950

y_pred[1] = 1600 + 150 = 1750

y_pred[2] = 800 + 150 = 950

Posiadane etykiety danych:

y[0] = 1500

y[1] = 3000

y[2] = 1900

Przykładowa funkcja kosztu l(y[i], y_pred[i]):

l(y[i], y_pred[i]) = |y[i] - y_pred[i]|

lub

l(y[i], y_pred[i]) = (y[i] - y_pred[i])^2

MSE (mean square error):

L(y, y_pred) **=** 1/n * ∑i {(y[i] - y_pred[i])^2}

Jak zmierzyć błąd algorytmu? - funkcja kosztu L(y, y_pred) c.d.

Nasze wyniki predykcji:

y_pred[0] = 800 + 150 = 950

y_pred[1] = 1600 + 150 = 1750

y_pred[2] = 800 + 150 = 950

Posiadane etykiety danych:

y[0] = 1500

y[1] = 3000

y[2] = 1900

Nasza funkcja kosztu L(y, y_pred)

MSE (mean square error):

L(y, y_pred) = 1/n * ∑i [(y[i] - y_pred[i])^2]

Funkcja kosztu L(y, y_pred)

L(y, y_pred) = ⅓ * [(1500 - 950)^2 + (3000 - 1750)^2 + (1900 - 950)^2] = ⅓ * [302500 + 1562500 + 902500] = ⅓ * 2767500 ~= **922500**

GRADIENT
KOŁO NAUKOWE

Funkcja kosztu L(y, y_pred)

Funkcja kosztu L(y, y_pred) = 922500 - to nie jest dobry wynik :)

Co jest naszym celem? - minimalizacja funkcji kosztu L(y, y_pred) czyli **min L(y, y_pred)**

Co jest naszym celem? - minimalizacja funkcji kosztu L(y, y_pred) czyli **min L(y, y_pred)**

Jak to zrobić? - zmieniając parametry ***θ* =** [a = 100, b = 150] przy pomocy metody gradientu prostego - ***Gradient Descent***

GRADIENT
KOŁO NAUKOWE

Metoda gradientu prostego (Kocioł pod Polskim Grzebieniem, Tatry Słowackie)

GRADIENT
KOŁO NAUKOWE

Metoda gradientu prostego

Jest wiele metod optymalizujących funkcję, my skupimy się na jednej z najpopularniejszych z nich - metodzie gradientu prostego.

Wyobraźmy sobie dolinę wśród gór w której chcemy znaleźć najniżej położony punkt, który określimy jako minimum globalne.

W tym celu zastosujemy właśnie metodę gradientu prostego, która licząc pochodne cząstkowe parametrów optymalizowanej funkcji wskazuje kierunek **wzrostu** funkcji!

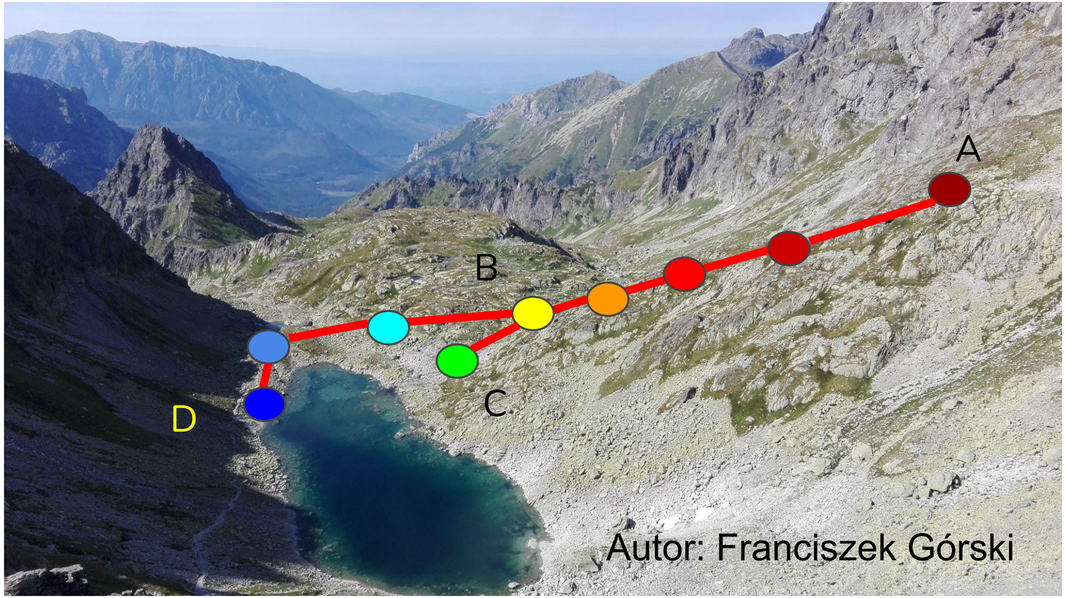
Metoda gradientu prostego

Wzrostu? Ale my chcemy minimalizować funkcję! - Dlatego będziemy wykorzystywali **zanegowaną** wartość gradientu - **negacja wzrostu == spadkowi**

Startujemy w punkcie A i korzystając z zanegowanej wartości gradientu kierujemy się w dół zbocza doliny.

Aż do punktu D, który jest najniżej położonym punktem w dolinie.

Uwaga na lokalne minima np. w punkcie C!



Metoda gradientu prostego - obliczenia

$\boldsymbol{L(y, y\_pred)} = 1/n * \sum i \{(y[i] - y\_pred[i])^2\}$

$\boldsymbol{h(x)} = a*x1 + b, h(x[i]) = y\_pred[i]$

$\boldsymbol{\theta} = [a, b]$

$\boldsymbol{L(y[i], x[i], \theta)} = 1/n * \sum i \{(y[i] - a*x[i] - b)^2\}$

$\boldsymbol{\nabla L(a, b)} = \{\partial a, \partial b\} =$

$\partial a = - 2/n * \sum i \{x[i]*(y[i] - a*x[i] + b)\}$

$\partial b = - 2/n * \sum i \{(y[i] - a*x[i] + b)\}$

$\theta 0 = [a0 = 100, b0 = 150]$

$a1 := a0 - \eta * \partial a = a0 - \eta * (- 2/n * \sum i \{x[i]*(y[i] - a0*x[i] + b0)\}) = a0 - \eta * (- 2/n * \sum i \{x[i]*(y[i] - y\_pred[i])\})$

$b1 := b0 - \eta * \partial b = b0 - \eta * (- 2/n * \sum i \{(y[i] - a0*x[i] + b0)\}) = b0 - \eta * (- 2/n * \sum i \{(y[i] - y\_pred[i])\})$

η (eta)- współczynnik uczenia (ang. *learning rate*)

GRADIENT
KOŁO NAUKOWE

Metoda gradientu prostego - obliczenia

θ0 = [a0 = 100, b0 = 150]

y = [1500, 3000, 1900]

y_pred = [950, 1750, 950]

η = 0.001

Nasze nowe parametry **θ1:**

θ1 = [***a1*** = - 78,66, ***b1*** = -175]

a1 := 100 - 0.001 * ⅔ * [8 * (1500 - 950) + 16 * (3000 - 1750) + 8 * (1900 - 950)] = 100 - 0.001 * ⅔ * [8 * 550 + 16 * 1250 + 8 * 950] = 100 - 0.001 * ⅔ * (4400 + 20000 + 7600) = 100 - 0.001 * ⅔ * 32000 = 100 - ⅔ * 32 ~= 100 - 21,33 = **- 78,66**

b1 := 150 - 0.001 * ⅔ * [150 * (1500 - 950) + 150 * (3000 - 1750) + 150 * (1900 - 950)] = 150 - 0.001 * ⅔ * [150 * 550 + 150 * 1250 + 150 * 950] = 150 - 0.001 * ⅔ * (82500 + 187500 + 142500) = 100 - 0.001 * ⅔ * 412500 = 100 - ⅔ * 412,5 ~= 100 - 275 = **- 175**

GRADIENT
KOŁO NAUKOWE

Regresja liniowa jednej zmiennej

Pamięć RAM
8
16
8

$= X$

Cena
1500
3000
1900

$= y$

Omawiany przez nas przykład określany jest jako **regresja liniowa jednej zmiennej**.

Regresja oznacza, że **zbiorem wartości** funkcji są **liczby rzeczywiste.**

W ramach uproszczenia przedstawiliśmy regresję dla hipotezy z jedną zmienną niezależną x.

GRADIENT
KOŁO NAUKOWE

Regresja liniowa wielu zmiennych

Pamięć RAM	Pamięć VRAM GPU	Liczba rdzeni CPU	Taktowanie CPU
8	0	4	2.4
16	6	8	2.5
8	2	8	2.3

$= X$

Cena
1500
3000
1900

$= y$

W prawdziwych problemach spotkacie się jednak z danymi zawierającymi wiele zmiennych niezależnych x, czyli wieloma parametrami danych.

GRADIENT
KOŁO NAUKOWE

Regresja liniowa wielu zmiennych

$$h(x) = a*x4 + b*x3 + c*x2 + d*x1 + e$$

I wtedy taka hipoteza zostanie wykorzystana do treningu modelu, reszta kroków pozostaje niezmienna …

Jednak w przypadku wielu zmiennych model liniowy może okazać się niewystarczający, ale o tym kiedy indziej …

GRADIENT
KOŁO NAUKOWE

Klasyfikacja

- Zbiorem wartości zamiast liczb rzeczywistych są dyskretne (z góry określone) wartości np. liczby [1, 2, 3]
- Klasyfikacja polega na przypisania danych do konkretnych klas
- Tak samo jak w regresji tutaj też jest hipoteza, jej parametry, funkcja kosztu i optymalizacja parametrów np. metodą gradientu prostego
- Inne są jednak hipotezy i funkcje kosztu

GRADIENT
KOŁO NAUKOWE

Klasyfikacja - dane

Pamięć RAM	Pamięć VRAM GPU	Liczba rdzeni CPU	Taktowanie CPU
8	0	4	2.4
16	6	8	2.5
8	2	8	2.3

= X

Klasa jakości (1 -3)
1
3
2

= y

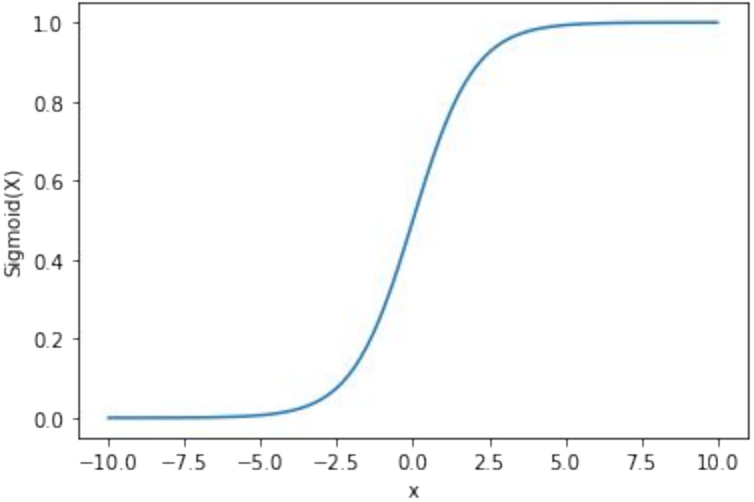
Jak widać teraz te same dane zostały przypisane do jednej z 3 klas określających poziom “możliwości” poszczególnych laptopów

Klasyfikacja - hipoteza

Jako hipotezy w klasyfikacji wykorzystuje się funkcje logistyczne takie jak sigmoid, tangens hiperboliczny czy ReLu.

sigmoid(x) = 1 / (1 + e^-x)

Zbiór wartości : <0, 1> albo <-1, 1>



GRADIENT
KOŁO NAUKOWE

Klasyfikacja - funkcja kosztu

Funkcja kosztu ma postać:

L(y, y_pred) = 1/n * ∑i [y[i] * (- log(y_pred[i])) + (1 - y[i])*(- log(1 - y_pred[i]))]

gdzie ***y_pred[i]*** = ***sigmoid***(x[i]) = 1 / (1 + e^-x[i])

Pytania?

Jeśli nie to …

pora na krótką prezentację w Colabie

GRADIENT
KOŁO NAUKOWE