Zajęcie 2. Metoda gradientu prostego. Stosowanie do algorytmu wstecznej propagacji błędu

Abstract

Celem jest nabycie podstawowej znajomości obliczenia gradientów oraz stosowanie gradientu w celu optymalizacji funkcji wielu zmiennych i wstecznej propagacji błędu

1. Zadanie 1

Zadanie dotyczy odnalezienia wartości minimalnej funkcji dwóch zmiennych f oraz zmiennych x i y metodą gradienta wraz z wizualizacją w 3D odpowiednio do określonego zadania. Można skorzystać z dowolnych bibliotek Python.

1. $f(x,y) = (x+3y)^3 + 2*x, \quad x \in [1;100], \quad y \in [1;100]$
2. $f(x,y) = \cos(x+3y)^{\sin(x)}, \quad x \in [0;2\pi], \quad y \in [0;2\pi]$
3. $f(x,y) = e^{\sqrt{(x+3y)}}, \quad x \in [1;16], \quad y \in [1;16]$
4. $f(x,y) = \log(x+3y)^2, \quad x \in [1;10], \quad y \in [1;10]$
5. $f(x,y) = (x+3y)^2 + e^{-x}, \quad x \in [1;10], \quad y \in [1;10]$
6. $f(x,y) = \cos((x+3y)^2), x \in [1,3], \quad y \in [1,3]$
7. $f(x,y) = (x+3y)^3 - 2*x, \quad x \in [1;100], \quad y \in [1;100]$
8. $f(x,y) = \sin(x+3y)^{\cos(x)}, \quad x \in [0;2\pi]$
9. $f(x,y) = \sin((x+3y)^2), x \in [1,3], y \in [1,3]$

10. $f(x,y) = x^2 - e^{-(x+3y)}, \quad x \in [1;10], \quad y \in [1;10]$

11. $f(x,y) = (x+3y)^{-1/4}, \quad x \in [1;10], \quad y \in [1;10]$

12. $f(x,y) = \frac{1}{\sqrt{(x+3y)}}, \quad x \in [1;10], \quad y \in [1;10]$

2. Zadanie 2

Zadanie dotyczy obliczenia gradientów sieci neuronowej z pomocą biblioteki numpy zadanej z pomocy architektury

1. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "relu"},
       {"input_dim": 2, "output_dim": 1, "activation": "relu"}
   ]
   ```
2. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "sigmoid"},
       {"input_dim": 2, "output_dim": 1, "activation": "relu"}
   ]
   ```
3. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "sigmoid"},
       {"input_dim": 2, "output_dim": 1, "activation": "sigmoid"}
   ]
   ```
4. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "tanh"},
       {"input_dim": 2, "output_dim": 1, "activation": "tanh"}
   ]
   ```
5. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "tanh"},
       {"input_dim": 2, "output_dim": 1, "activation": "relu"}
   ]
   ```
6. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "relu"},
       {"input_dim": 2, "output_dim": 1, "activation": "tanh"}
   ]
   ```
7. ```
   nn_architecture = [
       {"input_dim": 2, "output_dim": 2, "activation": "sigmoid"},
   ```
   ```

```
        {"input_dim": 2, "output_dim": 1, "activation": "tanh"}
    ]
```

8.
```
nn_architecture = [
        {"input_dim": 2, "output_dim": 2, "activation": "tanh"},
        {"input_dim": 2, "output_dim": 1, "activation": "sigmoid"}
    ]
```

9.
```
nn_architecture = [
        {"input_dim": 2, "output_dim": 2, "activation": "elu"},
        {"input_dim": 2, "output_dim": 1, "activation": "tanh"}
    ]
```

10.
```
nn_architecture = [
        {"input_dim": 2, "output_dim": 2, "activation": "elu"},
        {"input_dim": 2, "output_dim": 1, "activation": "elu"}
    ]
```

11.
```
nn_architecture = [
        {"input_dim": 2, "output_dim": 2, "activation": "elu"},
        {"input_dim": 2, "output_dim": 1, "activation": "sigmoid"}
    ]
```

12.
```
nn_architecture = [
        {"input_dim": 2, "output_dim": 2, "activation": "sigmoid"},
        {"input_dim": 2, "output_dim": 1, "activation": "elu"}
    ]
```

Sprawozdania w postaci:

1. Sprawozdanie (plik .pdf)
2. plik .ipynb
3. pdf-eksport pliku .pynb

zachować w zdalnym repozytorium (np Github) link na który umieściś w sprawozdaniu. Sprawozdanie należy wysłać na e-uczelnię zgodnie z ustalonym terminem.

References

References

[pandasUG] Pandas User's Guide `https://pandas.pydata.org/pandas-docs/stable/user_guide/index.html`

[DA2016] Data Analysis with Python and pandas using Jupyter Notebook
https://dev.socrata.com/blog/2016/02/01/pandas-and-jupyter-notebook.html