

WB-XIC, Lab2:

Wstęp do sieci neuronowych i PyTorch

Hubert Baniecki

hbaniecki@gmail.com | <http://hbaniecki.com>

Cel:

1. Zrozumienie **istoty** działania sieci neuronowej
 - ~~2. Pierwsze kroki w **Google Colab**~~
 3. Zaznajomienie się z pakietem **torch** w Python
 4. Zaimplementowanie algorytmu regresji liniowej
- + omówienie pracy domowej na 16 marca (8pkt, 2 tygodnie)

Komu udało się obejrzeć

Neural networks by 3Blue1Brown?

Classical Machine Learning

Task Driven

Supervised Learning

(Pre Categorized Data)

Classification

(Divide the socks by Color)

Eg. Identity
Fraud Detection

Regression

(Divide the Ties by Length)

Eg. Market
Forecasting

Clustering

(Divide by Similarity)

Eg. Targeted
Marketing

Data Driven

Unsupervised Learning

(Unlabelled Data)

Association

(Identify Sequences)

Eg. Customer
Recommendation

Dimensionality Reduction

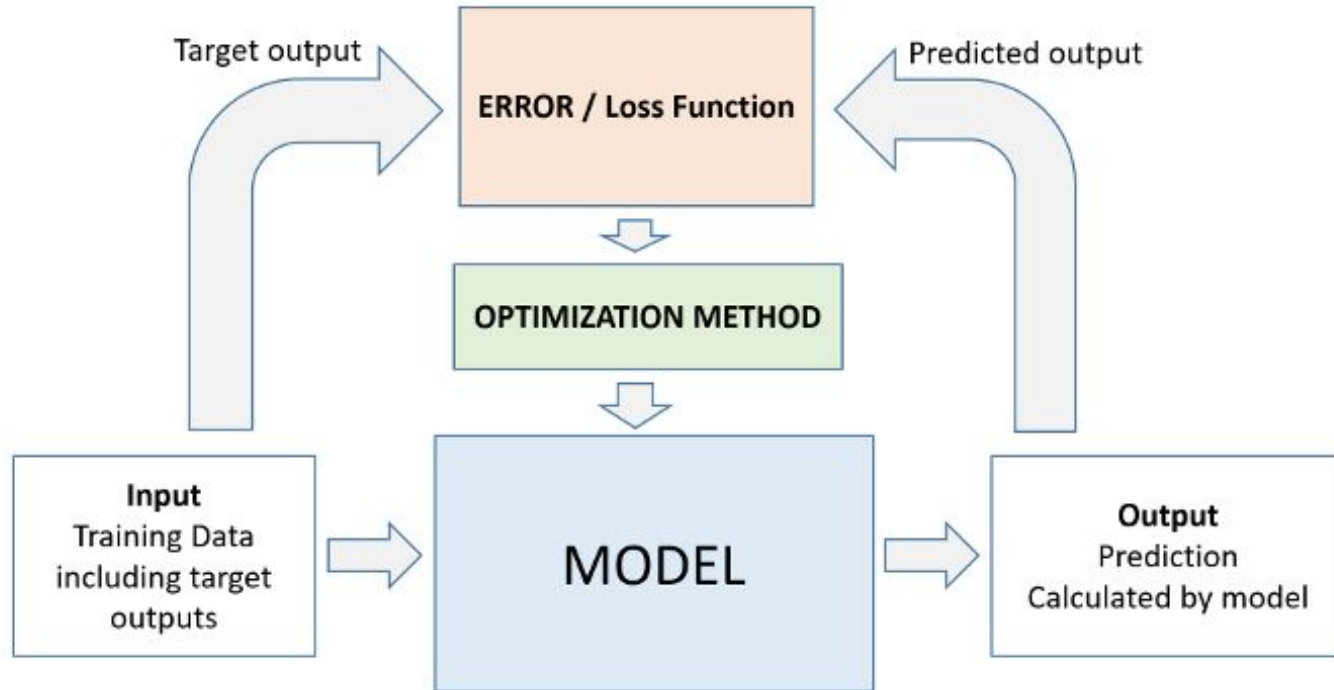
(Wider Dependencies)

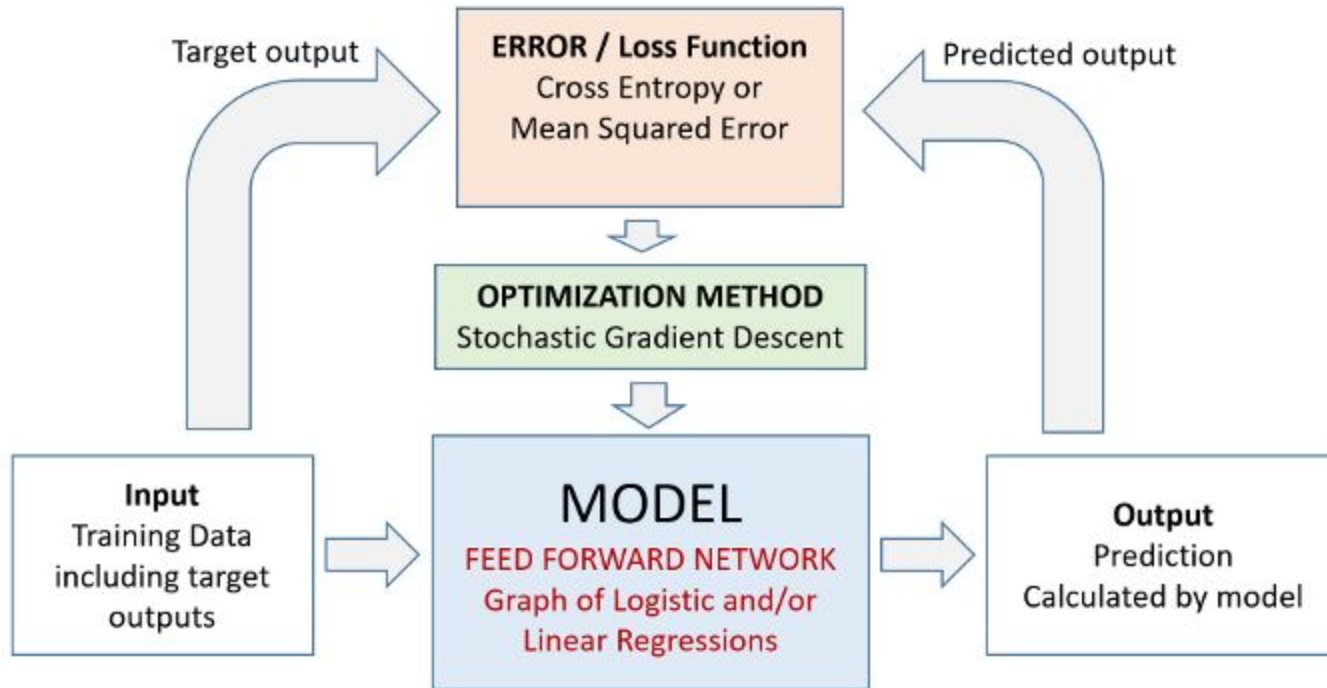
Eg. Big Data
Visualization

Obj: Predications & Predictive Models

Pattern/ Structure Recognition

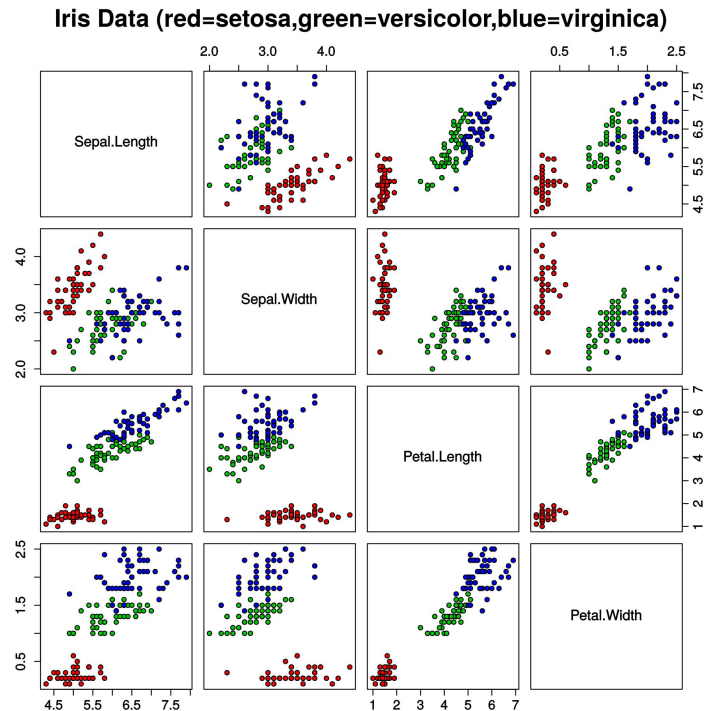
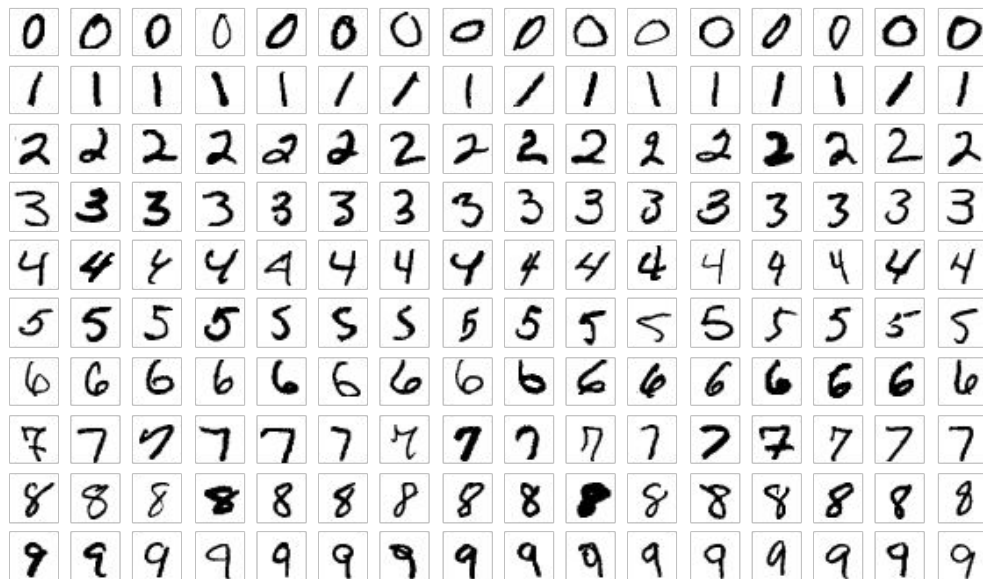






We need

1. Data for classification: X, Y



We need

1. Data for classification: X, Y
2. Loss function: $L(Y, Y^{\wedge})$

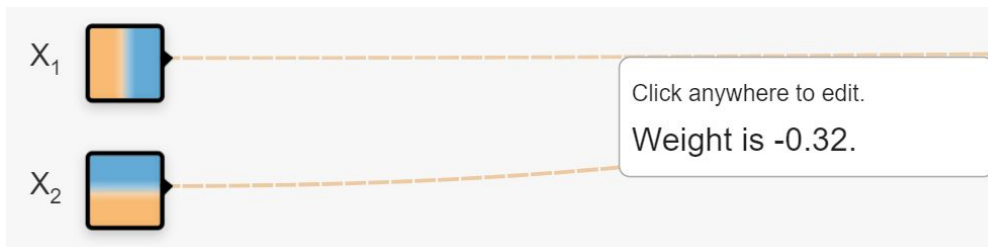
$$\text{MSE} = \overbrace{\frac{1}{n} \sum_{i=1}^n}^{\text{Mean}} \underbrace{(Y_i - \hat{Y}_i)}_{\text{Error}} \underbrace{)^2}_{\text{Squared}}$$

$$\underbrace{- \sum_{j=1}^M y_j \log(p(y_j))}_{\substack{\text{Sum over trials} \\ \updownarrow \text{Sum over classes}}} = \underbrace{- \sum_{i=1}^N y_i \log(p(y_i)) + (1 - y_i) \log(1 - p(y_i))}_{\substack{\text{Label} \quad \text{Prob of positive class} \quad \text{Label} \quad \text{Prob of positive class}}}$$

Diagram illustrating the relationship between the two loss functions. The top expression is the cross-entropy loss for a single trial, where y_j is the indicator variable and $p(y_j)$ is the probability of class j . The bottom expression is the cross-entropy loss averaged over N trials, where y_i is the label and $p(y_i)$ is the probability of the positive class. A blue double-headed arrow labeled "Sum over classes" connects the two expressions, indicating that the top expression is summed over classes to get the bottom expression. A pink arrow labeled "Sum over trials" points to the bottom expression, indicating that the top expression is summed over trials to get the bottom expression.

We need

1. Data for classification: X, Y
2. Loss function: $L(Y, Y^{\wedge})$
3. Statistical model: $f(X) = Y^{\wedge}$

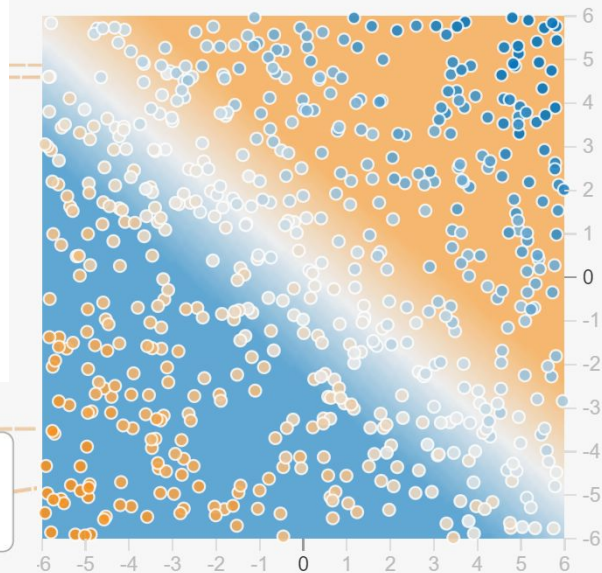


1. [A Neural Network Playground \(tensorflow.org\)](https://www.tensorflow.org/playground)
2. [A Neural Network Playground \(tensorflow.org\)](https://www.tensorflow.org/playground)
3. [A Neural Network Playground \(tensorflow.org\)](https://www.tensorflow.org/playground)

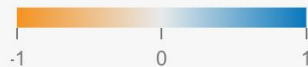
OUTPUT

Test loss 2.330

Training loss 2.429



Colors shows
data, neuron and
weight values.





Epoch
001,490

Learning rate

0.03

Activation

Tanh

Regularization

None

Regularization rate

0

Problem type

Classification

DATA

Which dataset do you want to use?



Ratio of training to test data: 50%

Noise: 0

Batch size: 10

REGENERATE

FEATURES

Which properties do you want to feed in?

X_1

X_2

X_1^2

X_2^2

X_1X_2

+ - 2 HIDDEN LAYERS

+ -

4 neurons

+ -

2 neurons

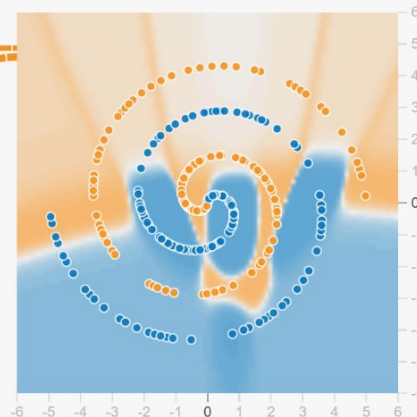
The outputs are mixed with varying weights, shown by the thickness of the lines.

This is the output from one neuron. Hover to see it larger.

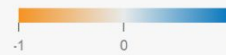
OUTPUT

Test loss 0.303

Training loss 0.208



Colors shows data, neuron and weight values.



☐ Show test data

☐ Discretize output

Zadanie.

Minimalizujemy funkcję straty na zbiorze treningowym Spirali.

Domyślne parametry, manipulujemy architekturą sieci.

<https://playground.tensorflow.org>

We need

1. Data for classification
2. Loss function
3. Statistical model
4. Optimization method

Stochastic gradient descent a.k.a. Analiza Matematyczna 1

- Choose an initial vector of parameters w and learning rate η .
- Repeat until an approximate minimum is obtained:
 - Randomly shuffle samples in the training set.
 - For $i = 1, 2, \dots, n$, do:
 - $w := w - \eta \nabla Q_i(w)$.

Iterate data in a loop through:

1. Feedforward

multiply input by parameters to obtain a prediction

2. Backpropagation

update parameters with gradient descent

[An Introduction to Statistical Learning, Chapter 10](#)

From neural networks
to **deep learning**



[Published: 27 May 2015](#)

Deep learning

[Yann LeCun](#) , [Yoshua Bengio](#) & [Geoffrey Hinton](#)

[Nature](#) **521**, 436–444 (2015) | [Cite this article](#)

689k Accesses | **29263** Citations | **1123** Altmetric

www.youtube.com/watch?v=HzilDlhWhrE



Wojciech Zaremba

OpenAI, GPT-3, Codex

<https://youtu.be/CV856uXQXnU>



Ian Goodfellow

Generative Adversarial Networks (GANs),
Deep Learning book (MIT press)

<https://youtu.be/Z6rxFNMGdn0?t=1039>



Ilya Sutskever

OpenAI, AlexNet won the
ImageNet Challenge 2012

<https://youtu.be/13CZPWmke6A?t=2480>

Neural networks 1989-2021

Convolutional Network Demo by Yann LeCun

https://www.youtube.com/watch?v=FwFduRA_L6Q

Tesla Full Self Driving explained by Andrej Karpathy

https://youtu.be/3SypMvnQT_s?t=480

https://youtu.be/3SypMvnQT_s?t=1445

<https://people.idsia.ch/~juergen/scientific-integrity-turing-award-deep-learning.html>

 **Jürgen Schmidhuber** @SchmidhuberAI · Sep 24, 2021

Critique of 2021 Turing Lecture, 2018 Turing Award: three Europeans went to North America, where they republished methods and concepts first published by other Europeans whom they did not cite - not even in later surveys.



people.idsia.ch

Scientific Integrity, Turing Lecture, Turing Award for Deep Learning

ACM and the awardees credit the awardees for work that did not cite the inventors of the used methods. But science is self-correcting.

12 72 268

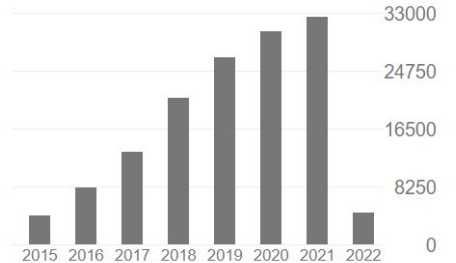
[neural networks](#) [physics](#)

[FOLLOW](#)

CITED BY	YEAR
61763	1997
15119	2015
5292 *	2012
5252	2000

Cited by [VIEW ALL](#)

	All	Since 2017
Citations	156501	128600
h-index	108	84
i10-index	391	243



Public access [VIEW ALL](#)

3 articles 144 articles

Jupyter Notebook

Cel:

1. Zrozumienie **istoty** działania sieci neuronowej
 - ~~2. Pierwsze kroki w **Google Colab**~~
 3. Zaznajomienie się z pakietem **torch** w Python
 4. Zaimplementowanie algorytmu regresji liniowej
- + omówienie pracy domowej na 16 marca (8pkt, 2 tygodnie)