TP "Python for AI":
a crash course focused on linear regression models

Name: Andrei Zinovyev

E-mail: **Andrei.Zinovyev.U900@gmail.com**

Page:

Python language

- *Franca lingua* for data science community today
- General purpose programming language (not specialized for a field like R)
- Rich set of standard libraries for data manipulation
- Rich set of standard libraries for numerical computations
- Rich set of standard libraries for machine learning
- Rich set of standard libraries for deep learning

- We will use Python 3!

Jupyter Python notebook

- To learn Python
- To write simple Python programs
- To develop complex Python program prototypes
- To exploit powerful computers remotely
- To do cloud computing
- To do data analysis and machine learning!
- To do data analysis without installing Python!

Colab: using Google computers and their GPUs, for your own projects

- Only a gmail account is needed

- Based on Jupyter notebooks

- Integrated with Google Drive

- Has Google machine learning products installed (such as TensorFlow)

- Free Cloud service with free GPU

Kaggle: open data scientist environment

- An online Community (3 million users) of Data Scientists and Machine Learners
- Enter competitions to solve data science challenges
- Data science online education
- Allows finding and publishing data sets

Other available services for online machine learning using Jupyter notebooks

binder

Microsoft Azure

COCALC

Datalore

nanoHUB

amazon web services

Few Python libraries we will use in this TP

- pandas – for manipulating data frames ('tables with headers')
- numpy – for manipulating 1d and 2d arrays
- scipy – for statistical calculations
- matplotlib – for standard data plotting
- seaborn – for easy data graphics with pandas
- scikit-learn – for machine learning
- tensorflow – for computing regression as a neural network

- lifelines – for survival analysis

scikit-learn: the most popular library machine learning

- Almost any machine learning method, using the same code pattern

```
model = SomeModel(parameters)
model.fit(X,Y)
Y_predicted = model.predict(X)
```

- Excellent documentation: https://scikit-learn.org/
- Started and implemented in France by INRIA in 2010

tensorflow: Google library oriented to deep learning

- a free and open-source software library for machine learning
- Developed by Google Brain in 2017
- Based on dataflow and computational graph
- Based on differentiable programming

Linear regression

- Connects many independent and one dependent variable via linear function
- Linear regression: the father of all supervised machine learning methods
- Linear regression: the most used machine learning method today
- Linear regression: the first machine learning method to apply, and see what it gives
- Linear regression: common problems and ways to deal with them (regularization)
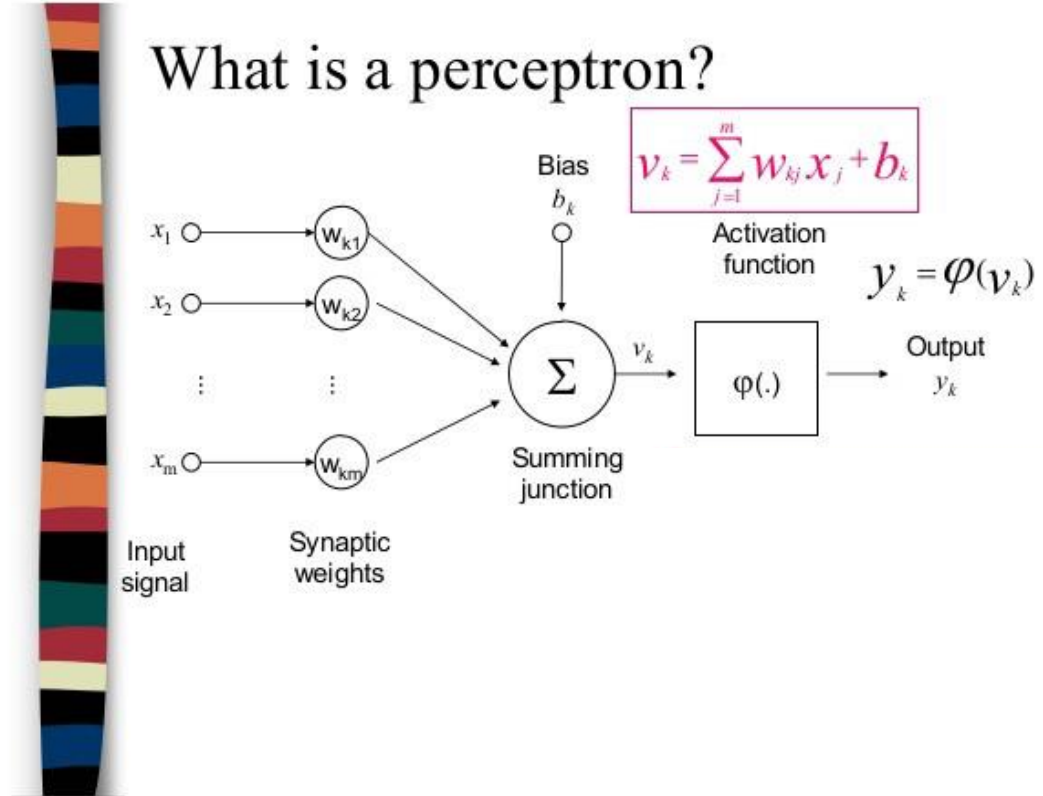- Linear regression can be used to produce non-linear data models
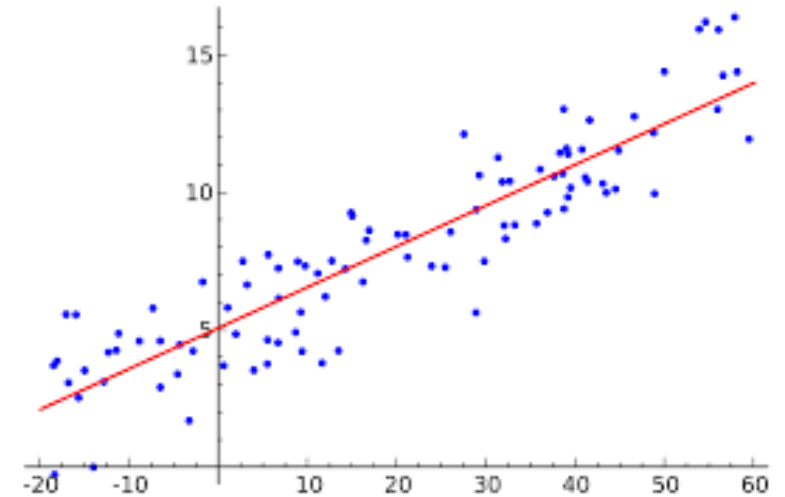
Three major flavors of linear regression in healthcare-related data science

- Simple Ordinary Least Square : when the target variable is continuous

- Logistic linear regression (logit): when the target variable is discrete (for example, binary)

- Survival Cox linear regression : when the target variable is a pair (follow up time + event)
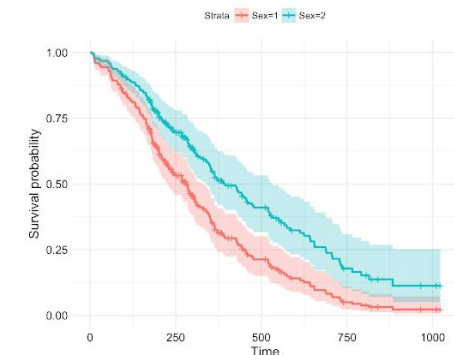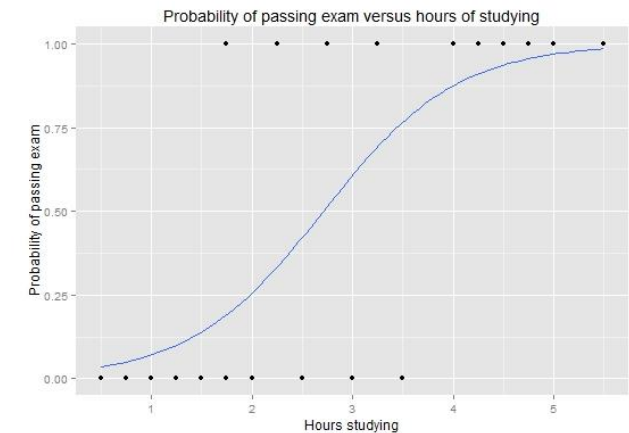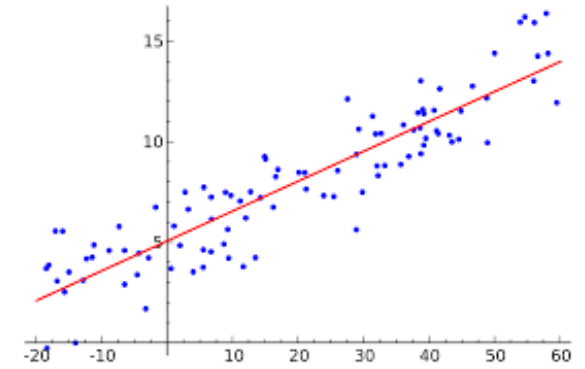
Linear regression as simple perceptron (formal neuron)

- Invented by Frank Rosenblatt in 1950s
- Elementary unit of any complex and deep neural network today

What is a perceptron?

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k$$

Bias b_k

Activation function

$$y_k = \varphi(v_k)$$

x_1 — w_{k1}

x_2 — w_{k2}

x_m — w_{km}

Σ v_k $\varphi(.)$ Output y_k

Summing junction

Input signal

Synaptic weights

If $\varphi(x) = x$, then it is simple linear regression model

If $\varphi(x)$ is a step-wise or sigmoidal function then it is a binary classifier

Objective of the TP: learn how to build simple regression models in Python

- Univariate and multi-variate regressions
- Validating the regression models in Python
- Building regression models using scikit-learn
- Building regression models using scipy
- Building regression models using tensorflow (as simple neural networks)

Plan of the TP

- **<u>Lesson I (~2.5 hours)</u>**
  - Connecting to Colab
  - Creating first Jupyter notebook
  - Working with a toy dataset
  - Visualizing simplest linear regression
  - Linear regression: using scikit-learn
  - Linear regression: using scipy statmodels
  - Loading a real-life dataset and exploring it
  - Simplifying the regression model

Plan of the TP

- ## <u>Lesson II (~2.5 hours)</u>
  - Testing and validating the linear regression

  Implementing linear regression in tensorflow
  - Basic things in tensorflow
  - First computational graph
  - Linear regression: using tensorflow and gradient descent

Ways to work on TP: all depends on your Python level/motivation!

- Some initial level of Python is assumed:
  - If not then a good start is at:
    https://wiki.python.org/moin/BeginnersGuide/NonProgrammers
    https://wiki.python.org/moin/BeginnersGuide/Programmers
    https://www.w3resource.com/python-exercises/python-basic-exercises.php
- We will start with a simple warming up exercise
- We will have a basic code example on a tiny imaginary example
- Five real-life datasets have been prepared (preprocessed) for this TP
- In some tasks you will need to insert missing code indicated by '…'
- Do not hesitate to copy-paste (from anywhere, e.g. Stack Overflow)
- Do not hesitate to read Wikipedia