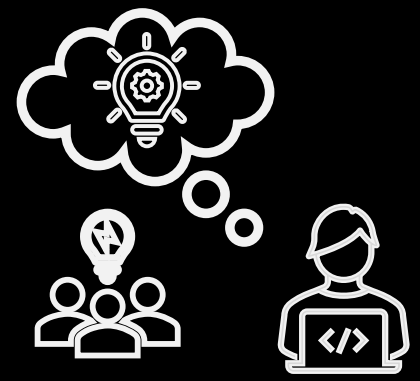
Unsupervised Learning:
Lab activities & Warm Up

Giulia Cisotto

Department of Informatics, Systems and Communication

University of Milan-Bicocca

giulia.cisotto@unimib.it



Academic year 2023/2024 – II semester

LAB ACTIVITIES – ORGANIZATION AND CONTACTS (1/3)

Instructor: Dr. Giulia Cisotto, Ph.D.

Contact: giulia.cisotto@unimib.it

*PLEASE: use the **Forum «Lab»** on E-Learning to share doubts, ask questions, and comments on the lab activities.*

Lab schedule (12 lab sessions):

March: 12th, 19th, 26th

April: 9th, 16th, 23th, 24th (extra, *lab TBD*), 30th

May: 7th, 14th, 21st, 28th

Lab room: LAB719 (U7)

LAB ACTIVITIES – ORGANIZATION AND CONTACTS (2/3)

Each lab session consists of the following «phases»:

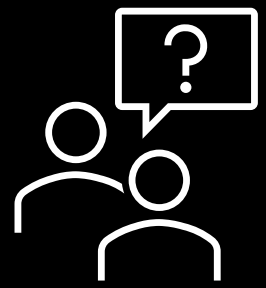
1. «pre-lab», where material (but not data) are provided (the day before)
2. «lab tasks», where students work in pairs (or individually, if needed) to solve some guided tasks.
3. «post-lab», the day after the Python code solution will be provided for you to check your solving method.

No mark for the lab activity this year!
Anyway, lab sessions are very useful to understand concepts explained by Prof. Stella and to have some methods for your final project.

LAB ACTIVITIES – ORGANIZATION AND CONTACTS (3/3)

Resources/Tools:

- Python 3.8 @Google Colab
- Forum «Lab» @eLearning
- Material «pre-lab» @eLearning



Let's start!

Motivation

Steps:

1. Introduction to Python, Colab, and tasks for today
2. A first example: loading and manipulating a .csv file
    a. Load the .csv file and extract part of the dataset **[TASK 1]**
    b. Manipulate the dataset **[TASK 2]**
    c. Explore and visualize the data **[TASK 3]**

3. *Let's collect some data..*
4. Repeat tasks on a new dataset **[Task 4]**

Motivation

This lab aims to warm you up and get familiar with several tools that you will be using across the future labs. You can also refer to Chapter 1 of the *Introduction to Data Mining* book, the first three related notebooks ("Getting Started", "Introduction to Python" and "Introduction to Numpy and Pandas") and to Prof. Stella's first classes.

Today we will introduce *Python* using *Google Colab*, load and explore some record data.

Useful **packages**: numpy, pandas, matplotlib, seaborn

Useful Python **data structures**: DataFrame, Series

Motivation

Steps:

1. Introduction to Python, Colab, and tasks for today
2. A first example: loading and manipulating a .csv file
    a. Load the .csv file and extract part of the dataset **[TASK 1]**
    b. Manipulate the dataset **[TASK 2]**
    c. Explore and visualize the data **[TASK 3]**

3. *Let's collect some data..*
4. Repeat tasks on a new dataset **[Task 4]**

PYTHON

What is Python? Executive Summary

Python is an ***interpreted, object-oriented, high-level programming language with dynamic semantics***. Its high-level built in **data structures**, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. **Python supports modules and packages**, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be **freely distributed**.

Useful **packages**: numpy (for math), pandas (for data management), matplotlib (for plots), seaborn (for better plots)

Useful Python **data structures**:

- built-in data structures: list, dictionaries, arrays,
- Pandas (additional package) offers also: **DataFrame**, Series

COLAB

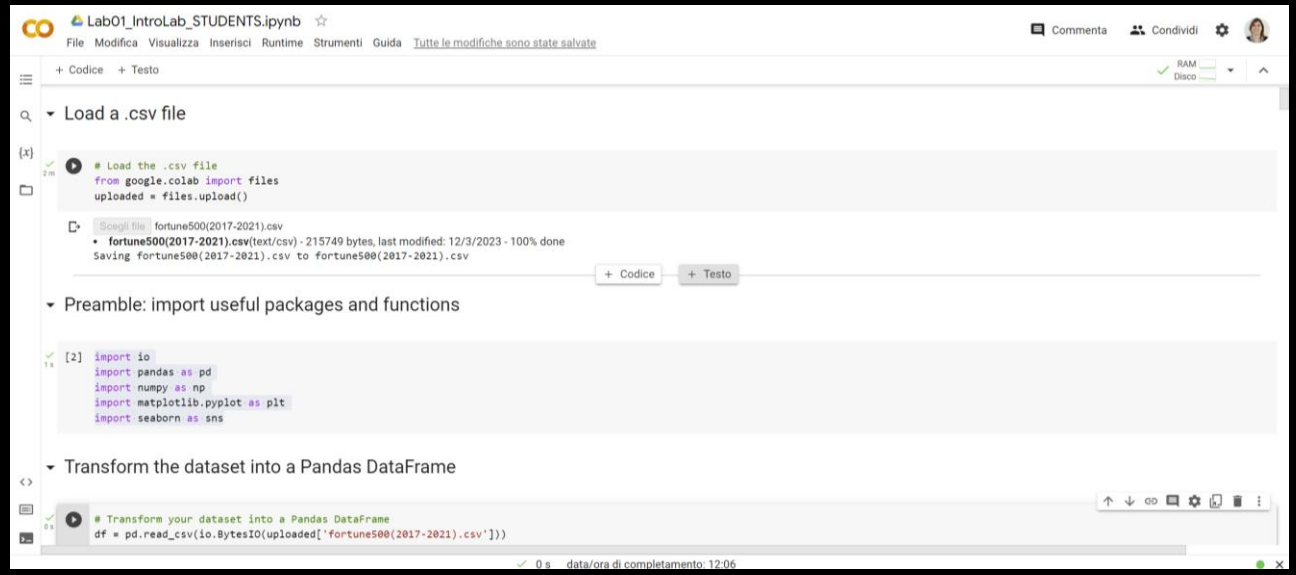
Colab (short for Collaboratory) is Google's free platform which enables users **to code in Python**. It is a **Jupyter Notebook-based cloud service**, provided by Google. This platform allows us to train Machine Learning models (i.e., including unsupervised learning models) directly in the cloud and all for free. Google Colab does whatever your Jupyter Notebook does and a bit more, i.e. **you can use GPU** and TPU for free. Some of Google Colab's advantages include quick installation and real-time sharing of Notebooks between users.

Code + text

Run a cell using CTRL+ENTER

Automatic saving (checkpoint)

Sharing and use of Colab during labs



Tasks to learn today

How to read from a .csv file in Colab (Towards Data Science article)

- Colab (short for Colaboratory) is a free platform from Google that allows users to code in Python. Colab is essentially the Google Suite version of a Jupyter Notebook.

How to manipulate the data

- **Pandas dataframe/Series**: structure, selection of a part of the dataframe, basic operations on it
- **Numpy array**: structures, selection of part of the data, basic operations on it
- **Preprocessing**: scaling and normalization (e.g., min-max normalization or z-score)

How to visualize the data

- matplotlib
- seaborn

Motivation

Steps:

1. Introduction to Python, Colab, and tasks for today
2. A first example: loading and manipulating a .csv file
    a. Load the .csv file and extract part of the dataset **[TASK 1]**
    b. Manipulate the dataset **[TASK 2]**
    c. Explore and visualize the data **[TASK 3]**

3. *Let's collect some data..*
4. Repeat tasks on a new dataset **[Task 4]**

LOAD A .CSV FILE

A .csv file is a spreadsheet where each row contains one person's answers, and each column contains all the answers to a particular question, typically. Let's take a look to a snippet of our data set.

About Dataset: fortune500(2017-2021).csv [211kb unzipped]
It was formed after scraping the Fortune 500 companies and their corresponding information from the https://fortune.com/ website between the years 2017 and 2021.

1. Go to this link (Kaggle repo), download the «**Fortune500(2017-2021)» dataset,** save it in a local folder, and extract from the .zip
2. Unzip the folder and read the «**Fortune500(2017-2021).csv»** in Excel, following this procedure:
   i. Go to «Data»
   ii. Import from «Text/CSV»
   iii. Find your .csv file
   iv. Load it

Column1	Rank	Name	Revenues($M)	Revenue Percent Change	Profits ($M)	Profits Percent Change	Assets ($M)	Market Value as of march 31st of that year	Change in Rank (Full 1000)	Employees
0	1	Walmart	5591510	7	135100	-9	2524960	3826428		23000000
1	2	Amazon	3860640	38	213310	84	3211950	15580696		12980000
2	3	Apple	2745150	6	574110	4	3238880	20506659	10	1470000
3	4	CVS Health	2687060	5	71790	8	2307150	986532	10	2565000
4	5	UnitedHealth Group	2571410	6	154030	11	1972890	3517250	20	3300000
5	6	Berkshire Hathaway	2455100	-4	425210	-48	8737290	5878230		3600000

GETTING STARTED WITH YOUR JUPYTER NOTEBOOK

Jupyter Notebooks are a popular tool for data analysis because they are quick to set up and very convenient to use. Further details can be found at this in-depth Jupyter Notebooks tutorial.

Here is the link to **the Notebook @Colab for today**

Preamble: import packages
We'll import Pandas to work with our data, Matplotlib to plot charts, and Seaborn to make our charts prettier. It's also common to import NumPy but in this case, pandas imports it for us.

Preamble: load the .csv file (from local)

The dataset is transformed into a Pandas(*) DataFrame:
(*) Pandas (PANel Data Analysis) is a popular library when it comes to data analysis and machine learning. Pandas library is built on top of Numpy.

```
df = pd.read_csv(io.BytesIO(uploaded['fortune500(2017-2021).csv']))
```

To get on-the-fly help (e.g.):

```
list?
```

EXPLORE THE DATASET [TASK 1]

Pandas DataFrame

pandas.DataFrame documentation

Mutable, (here, a good description)

Your tasks here are to describe your dataset:

1. Print out the first and the last 10 elements (hint: **.head** and **.tail**)
2. Find out how many objects and attributes are in the dataset (hint: **.shape**)
3. What data type are included for each attribute (hint: **.dtypes**)
4. Extract one single column (e.g., the profits)
5. Extract only rows of a specific company (e.g., ABM Industries)
6. Count the number of objects you have per each “year” (absolute value and normalized). Hint: **.value_counts**

MANIPULATION OF THE DATASET [TASK 2]

1. Rename the columns in a simpler way:

```
['id', 'rank', 'company', 'revenues', 'revenue_perc', 'profit', 'profit_perc',
 'assets', 'market_value', 'change1000', 'employees', 'change500', 'year']
```

2. Check if any «NaN» are in the dataset (hint: use **.isnull**)
3. Count the number of «NaN» in the «profit» column (hint: **.sum**)
4. Compute the mean, min, max, and standard deviation profit of company = «ABM Industries». Hint: **.mean, .min, .max, .std**
5. Decide how to scaleone column and report the first 5 objects before and after manipulation. By the way, to notice the difference between different scaling methods, you can plot the scaled values after applying different scalings.

Scaling options

MinMaxScaler: to bring values in the interval [0,1]. Alternatively, you can force the range to be in [A,B]. Formulas:

$$z = (x - min) / (max - min)$$
$$z\_alternative = z * (B - A) + A$$

StandardScaler: to bring values centered in zero and having standard deviation equal to 1. Formula:

$$z = (x - u) / s$$

Typically scaling is done by standard scaling. However, *outliers* can often influence the sample mean / variance in a negative way. In such cases, using the median and the interquartile range often give better results.

RobustScaler: similar to the StandardScaler, but it remove the median and divide by the IQR (the range between the 25th quartile and the 75th quartile).

Normalizer: it normalizes each feature so that its **L1** (sum of values) **or L2** (default: sqrt of squared values) **norm** is equal to 1.

Although using the normalize() function results in values between 0 and 1, it's not the same as simply scaling the values to fall between 0 and 1.

VISUALIZATION THE DATASET [TASK 3]

```
%matplotlib inline
```

That first line isn't a Python command, but it uses something called a line magic to instruct Jupyter to capture Matplotlib plots and render them in the cell output.

It might help to convert from Series to numpy array:

```
x = np.array( Series_NAME )
```

Your tasks here are to describe your dataset:

1. Using Matplotlib, plot the revenues
2. Using Matplotlib, plot the revenues of a specific company in the years
3. Using Matplotlib, make a scatterplot to see if any trend exists between profits and employees
4. Using Seaborn, repeat the same as in 3.

Next Lab on
Proximity measures

March, 26th – 2.30 p.m.

@LAB**719**, U7

Meanwhile, if you have any questions…

use the Lab Forum @eLearning!