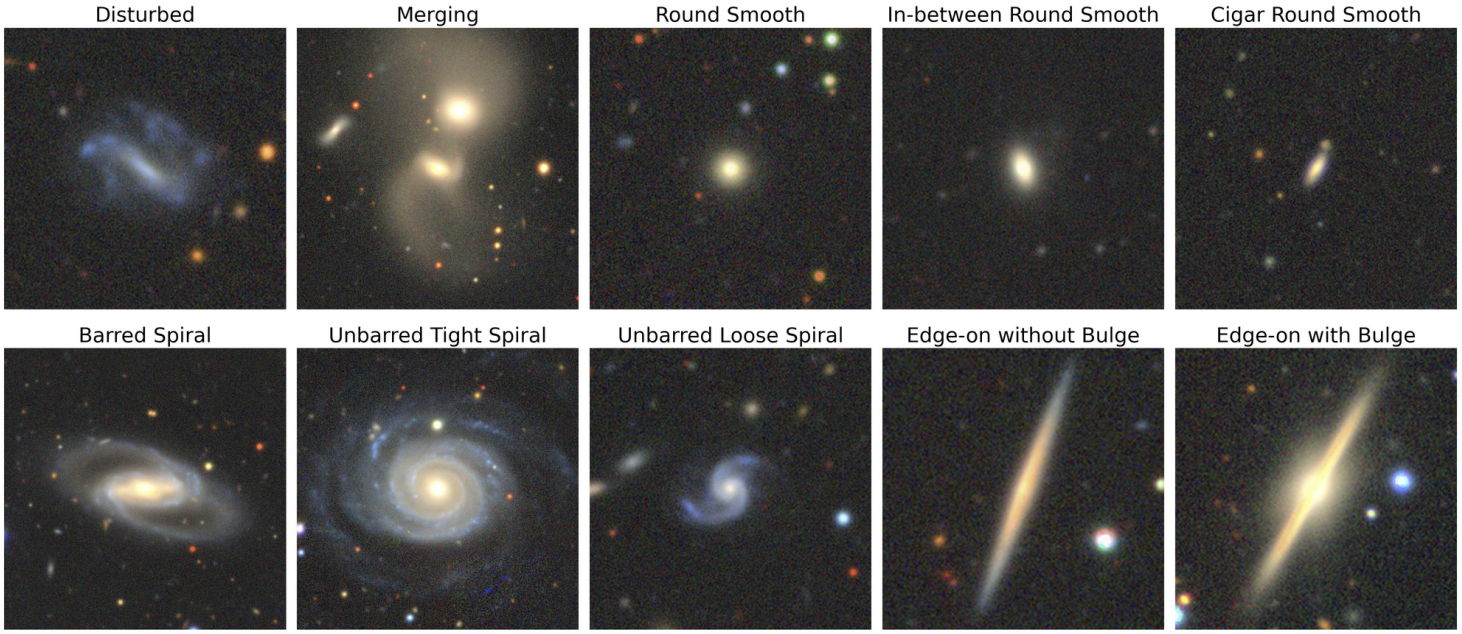
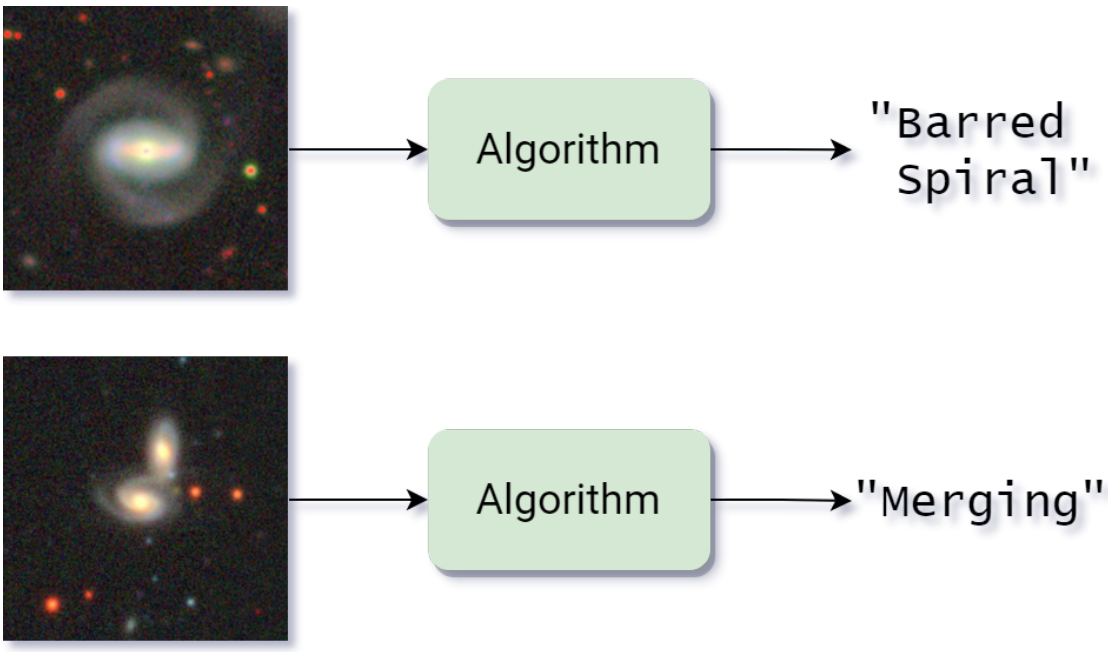
Introduction to Machine Learning Project

1) **Rules:** the project is mandatory. Students will carry out the project and discuss it during the oral exam. The project is individual, and each student is required to develop an original solution which should be properly justified and sustained. The student is required to produce a report of 1/2 pages (figures are excluded from the page count) which will include an explanation of the design choices, the issues and performances encountered during the development of the project. In addition, the student is required to provide the source code. The report and the code might be sent in a zip file. Alternatively, students can produce a Colab notebook with runnable code, similar to the one you have seen during the laboratory session. In case you opt for the latter solution, the submission consists in sharing the link to your colab notebook besides the report (sent by email).



2) **Objective:** the main objective of this project is to create an algorithm that is able to identify, given an RGB image of a galaxy as input, the morphological class it belongs to. The morphological classes that we target are 10, and they are depicted in the figure below.

   In order to start to understand the task, here is an example:



In this example we have two observations of two different galaxies (actually three). The first one is a “Barred Spiral” galaxy, you can recognize this by looking at its arms, which are folded in a spiral-like fashion, and at the bright central bar-shaped structure composed of stars. Interestingly, The Milky Way Galaxy, where the Solar System is located, is classified as a barred spiral galaxy, and exhibits a similar appearance to the one in the picture. On the other hand, the bottom picture shows two galaxies that are colliding. Such images are labeled as “Merging” galaxies.
Given these images as input the algorithm should be able to assign them the correct class label. By looking at these two examples, this looks like a very easy task but differences in shapes can sometimes be subtle, making the task harder. Despite this, in most cases there exist meaningful features that the algorithm can exploit to correctly classify the images.

3) **Data:** we have a dataset composed of ~17k images, each one has a 256x256 pixels resolution. As usually done when developing such an algorithm, we need to make some considerations:

   A. **Subdivision of the data into splits:** as said before, we have ~17k images at our disposal. Despite this fact, we cannot use them all just to develop our algorithm, but we need to reserve a part of them in order to assess the performance. For this reason, we will divide the dataset into two splits: *train* and *test*. The *train* split will be given to the students with its corresponding ground truth labels, meaning that for each image of a galaxy we will also know its morphological class. On the other hand, for the *test* split, only the images will be given. The *train* split will be used to develop and improve the algorithm, while the *test* split will be used in order to verify the performance of the algorithm. The *test* set must not be used for training.

In addition to this, the students are encouraged to further divide the *train* split to obtain a *validation* split. The *validation* split provides an unbiased evaluation of the model performance, and therefore should be used to perform model selection (e.g. testing design choices, tuning hyperparameters, etc...)

This table shows statistics about the three splits:

Class / Split	Train	Test	Total
Barred Spiral	1430	613	2043
Cigar Shaped Smooth	234	100	334
Disturbed	757	324	1081
Edge-on with Bulge	1311	562	1873
Edge-on without Bulge	996	427	1423
In-between Round Smooth	1419	608	2027
Merging	1297	556	1853
Round Smooth	1851	794	2645
Unbarred Loose Spiral	1840	788	2628
Unbarred Tight Spiral	1280	549	1829
TOTAL	**12415**	**5321**	**17736**

Given the ~17k images, we consider ~70% for *train*, ~30% for *test.* For *validation*, a suitable amount of data can be 20%.

B. **Number of images for each class:** as you can see from the table above, the number of images for each class is fairly different. In particular, the number of Cigar Shaped Smooth Galaxies is very limited. This is a very common issue which is usually found in many datasets, also known as *class imbalance*. In order to obtain the best results, it will be very important to take this class imbalance into account.

4) **Where to find the data and template for the report**: the data and the template for writing the report is available through this google drive folder.
In the “report_template” folder you will find the Latex report template. Please open the “egpaper_final.tex” file with any Latex editor you like (e.g. the open source TexMaker, or overleaf) and modify it. The document that needs to be submitted as a report is the PDF version of that document. An example of a good report is also present in the shared folder.

In the “dataset.tar.gz” file you will find 2 sub-folders, one for each split. In the *train* folder, the data will be divided into 10 sub-folders, one for each class. In the test folder you will only find the images and features. In order to load the data into Colab follow this guide https://colab.research.google.com/notebooks/io.ipynb.

5) **Methods to use:** you can use any of the methods which have been taught during this course. We encourage you not to limit the solution to just one method, but instead to try out different ones in order to better understand their strengths and limitations and report results obtained by each method. We also suggest trying both deep and shallow methods. For shallow methods, you should first extract some features from the image. This can be done in many ways. We suggest to use one of the following methods:
   a) (Recommended) Using a pretrained convolutional neural network. You can find pretrained network weights in torchvision.models
   b) Using the color histogram

6) **Fairness:** Copying and cheating are not permitted. Students exhibiting unfair behavior will be expelled from the competition and will fail the exam. In particular, the following behaviors are not permitted:
   a) Hand-labeling the test set is forbidden. This not only makes you lose a lot of time and fail the exam but also does not teach you anything (except perhaps galactic morphology)
   b) Copying your friend’s assignment is an act of plagiarism. Do not do it!

7) **How to measure the performance:** the objective of the project is to obtain the highest accuracy on the *test* set. However, you will not have access to the ground-truth for the test data, but can estimate the generalization performance on the available validation data. Two metrics will be used for evaluation:
   a) **sample-wise accuracy**: The metric that simply measures the number of correct predictions with respect to the total number of samples in the test set. In formulas:

$$A_{sample-wise} = \frac{1}{N} \sum_{n=0}^{N} 1(y_n == p_n)$$

   where $1()$ is an indicator operator, y_n is the ground truth labels and p_n is the label predicted by the algorithm.
   b) **class-wise accuracy**: The metric that we will use takes into account the accuracy of each category and, in order to obtain good performances, the algorithms need to be accurate on all of them. In particular, the overall accuracy will be called $\mathbf{A_{tot}}$ and will be computed as the average of the accuracies on each category. Since we have ten categories, it will be

$$A_{class-wise} = \frac{1}{|C|} \sum_{c \in C} A_c$$

where **C** is the set of our ten categories, $|C|$is the cardinality of the set (10 in this case) and $\mathbf{A_c}$ is the accuracy on each category, computed as

$$A_c = \frac{TP_c}{TP_c + FN_c}$$

where $\mathbf{TP_c}$ is the number of images which have been *correctly* assigned to category "**c**", while $\mathbf{FN_c}$ is the number of images which have been *incorrectly* assigned to category "**c**".

8) **How to measure the performance on the test set:** since you don't know the labels of the test set, you will need to send us the predicted class for each image included into the test set. Your test set accuracy result will be sent back to you. In order to motivate you we will also keep a leaderboard with all the scores. We can keep it anonymous if you don't want to show your name. We will limit the number of test submissions to 5. The reason we do this is that you should try to optimize your algorithms relying only on *train* and *validation* as much as you can, and only at the very end try to evaluate the performance on the *test* set. In order to have a standardized way of sending your test set results, please provide your prediction as a single csv file, with a line for every image. Each line will contain two items, the image_id and class_name. Just to be clear, produce and send a file that looks like this:

0,Edge-on with Bulge
1,Barred Spiral
2,Round Smooth
3,Merging
4,Unbarred Loose Spiral
5,Merging
6,Round Smooth
7,Barred Spiral
8,Barred Spiral
9,Disturbed
10,In-between Round Smooth
…

The total number of lines in this file has to be the number of images in the test set which is 5321, the image_id (that is the filename of the image without ".png") should go from 0 to 5320, the set of possible class_names is {*Barred Spiral, Cigar Shaped Smooth, Disturbed, Edge-on with Bulge ,Edge-on without Bulge, In-between Round Smooth, Merging, Round Smooth, Unbarred Loose Spiral, Unbarred Tight Spiral*}.

To return the path of the image from ImageFolder you can do the following:

```
from torchvision import datasets

class ImageFolderWithPath(datasets.ImageFolder):
    def __getitem__(self, index):
        return super().__getitem__(index), self.imgs[index][0]
```

9) **Contacts:** if you have any question or you want to submit your test results, send an email to enrico.fini@unitn.it
10) **Deadline:** 7 days before the discussion of the project in the oral exam. It will be allowed to discuss the project and carry out the exam about theoretical contents of the course in two separate exam sessions.
11) **Loading the dataset** in Google Colab (example):
    a) copy the dataset in your Google Drive
    b) mount your Google Drive in Google Colab:

```
from google.colab import drive
drive.mount('/content/drive')
```

    c) extract the dataset:

```
! tar -xf drive/path/to/dataset.tar.gz
```

    d) now you can use torchvision.datasets.ImageFolder to parse the dataset
    e) and torch.utils.data.DataLoader to load the images