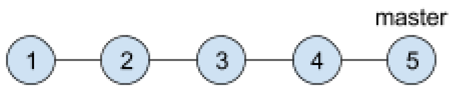
Homework 1

All assignments need to be submitted via github classroom, assigment
https://classroom.github.com/assignment-invitations/98f1f96fb1bce3413b9d1a3315e1f6ba

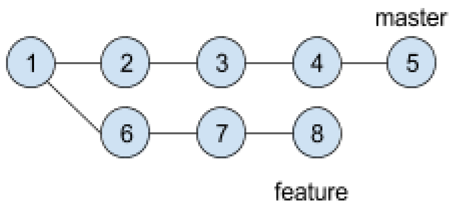
Each assignment needs to be a separate subfolder in the repository called `task1`, `task2`, `task3` and `task4`. Please add a Readme.md to your repository stating your UNI so that we can identify you. Please also add links to the output of the travis run for task2 and the generated website for task4.

Task 1: Git

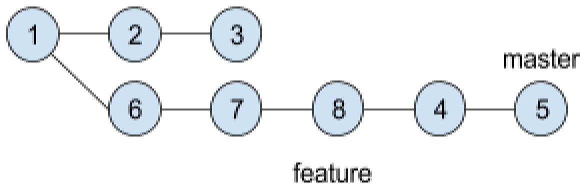
Write a shell script that creates a new folder, a repository in that new folder and a series of commits as described in the following. Each commit should create a new empty file with the number of the commit as file name (1 to 10), using `touch`.
First, create a chain of five commits on the branch `master`:



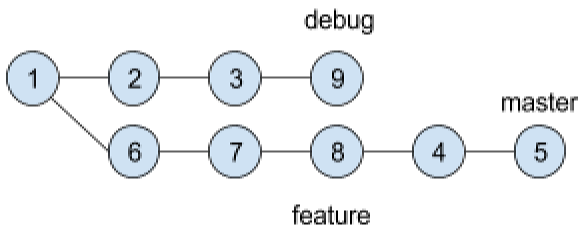
Then, create three new commits starting from the first commit, on a branch called `feature`:



Then, move the commits 4 and 5 to the `feature` branch:



Create a new commit attached to commit 3 in a branch called `debug`



Finally, amend the content of commit “9” to include file “7”.

Task 2: Continuous integration

Ensure that all tests that are written as part of this task are run by continuous integration and pass.

2.1 Set up travisci for your repository, running for Python2.7, Python3.4 and Python3.5. Travis is already enabled for your github classroom repository, you can check the status at https://travis-ci.com/AppliedMachineLearning/<your_repo_name>

2.2 Create a test using py.test that ensures that two divided by eight is 0.25, once using built-in functions, once using numpy arrays of shape `(1,)`.

2.3 Write a test that reads `input.txt` and ensures that it has the expected number of characters. The io module might be helpful.
The input.txt file should be in your repository.
2.4 Write a test to ensure that `KNeighborsClassifier` achieves at least 70% mean cross-validation accuracy on the iris dataset, when using 5-fold cross-validation.
See `sklearn.model_selection.cross_val_score`, `sklearn.neighbors.KNeighborsClassifier`, `sklearn.datasets.load_iris`.

Task 3: Documentation

3.1 Write a function that takes two inputs: a dataset represented as a numpy-array, and an “axis” argument. The function computes mean and standard deviation of a dataset along the specified “axis”, which can be 0, 1, or None (in which case it computes the mean of all entries), and returns them. The axis argument is optional with a default of 0.

3.2 Document the function using NumPy doc style.

3.3 Use Sphinx to generate a html documentation of the function.

3.4 Write a readme in restructured text that explains what the function does using LaTeX equations and links to the function documentation generated by sphinx.

3.5 Use readthedocs to render the documentation and readme using sphinx.

Task 4: Data Visualization and Analysis

All figures that are part of this task need to be generated by stand-alone scripts. Use sphinx and githubs gh-pages to publish the script together with the figure on a website. The figure needs to be autogenerated by the code to ensure the code works.

To generate the figure, you can use the plot directive as explained here:
http://stackoverflow.com/questions/16047271/plot-directive-in-restructured-text
and here:
http://matplotlib.org/sampledoc/extensions.html#inserting-matplotlib-plots

Alternatively you can also chose to use sphinxgallery:
https://sphinx-gallery.readthedocs.io/en/latest/

To publish the website, push the generated html content to a branch called gh-pages in your repository.

4.1 Create a pair-plot of the iris dataset similar to Figure 1-3 in IMLP using only numpy and matplotlib. Ensure all axes are labeled. The diagonals need to contain histograms, the different species need to be distinguished by color or glyph, and there needs to be a legend for the species.