In this project you will be training various machine learning algorithms on a few different regression and classification data sets and building pipelines to manage data and explore hyperparameters. The primary objectives are to:

- gain experience with data pre-processing, model selection, and hyperparameter tuning,
- get familiar with Python libraries for data science and machine learning, and
- practice planning experiments and visualizing/writing up the results.

Data Sets

In your starting-point repository, you have data files for two data sets from the UCI machine learning repository. You are also expected to select at least *one additional data set* from the UCI repository (please look around for something you find interesting, and don't just choose the first option in the list).

- `ENB2012_data.xlsx` is a regression data set about building energy efficiency. The original source of the data, with more information about the features can be found here: `https://archive-beta.ics.uci.edu/ml/datasets/energy+efficiency`.
- `wdbc.data` and `wdbc.names` represent a classification data set about breast cancer diagnosis. The original source of the data with more information can be found here: `https://archive-beta.ics.uci.edu/ml/datasets/breast+cancer+wisconsin+diagnostic`.

To get started with these data sets (and the additional one you chose), import each into a Jupyter notebook as a Pandas data frame. You should then take a look at the Pandas documentation to see what sorts of things you can do with these data frames.

```
import pandas
energy_data = pandas.read_excel("ENB2012_data.xlsx")
cancer_data = pandas.read_csv("wdbc.data")
```

If these commands (or others you try later) fail because a library isn't installed, you can add it using a command similar to the following at the terminal:

```
pip3 install --user library
```

Data Processing

Your first task is to get the data into a format where you can use it to train a machine learning algorithm. Pandas has some tools for this, but you will also want to make significant use of the Scikit-Learn pre-processing libraries. But before calling any of these functions you should look over the documentation for your data sets and think about which features of the data should be inputs, which should be outputs, and which should be ignored.

In addition to basic translation into valid ML inputs, you are required to explore a number of pre-processing options. At a bare minimum, you should try:

- normalizing the data to zero mean and unit variance,
- augmenting the input dimensions with polynomial features, and
- at least one other pre-processing step.

You *must* also perform a train/validate/test split, reserving at least 10% of each data set for testing, and either holding out a further 10% for validation, or using cross-validation on the training set.

Model Selection and Tuning

For each regression task, you should try using linear regression and at least one other regression algorithm. For each classification task, you should try using logistic regression and at least one other classification algorithm. Your other algorithms *must not* use neural networks. For each algorithm, you should:

- enumerate all potentially-relevant hyperparameters,
- find plausible ranges for each hyperparameter, and
- jointly optimize the hyperparameters and the pre-processing.

Your goal is to find the very best version of each model for each task and then present a comparison between algorithms using the best-variants you identified. You are encouraged to make use of cross validation and randomized parameter optimization where appropriate. You are strongly advised to visualize your results with plots as often as possible.

Documenting Your Work

In this project, you will be doing a very large amount of trial-and-error, and it will be important to take notes as you go! Some parts of that trial-and-error will be about figuring out how to use the available tools, while other parts will be about experimenting with your data sets and the available options for your machine learning pipelines. To keep track of all of this, you should be using several Jupyter notebooks, and you should also extract functionality you find important and want to re-use into stand-alone python scripts. In the starting-point code, you've been provided the following files, which are *not* meant to be exhaustive!

- `scratchwork.ipynb`: for trying things out and figuring out how to call library functions and manage your data,
- `experiments.ipynb`: for performing deliberate experiments to optimize your models,
- `presentation.ipynb`: your cleaned-up demonstration of what you learned, and
- `my_pipelines.py`: a script where you can write functions that aren't interactive, are needed by multiple notebooks, or otherwise don't need to live in the Jupyter notebooks. Note that while you are welcome to use Scikit-Learn pipelines, the name of this file is not meant to indicate that as a requirement.

It's very likely that you will want to create even more files to separate out functionality, take notes, and otherwise keep things organized! In general, you should be making a lot more use in this project of writing outputs to files, plotting or other types of visualization, and commentary using `markdown` cells.

Presenting Your Results

Your deliverable this week (for both the code review and the submission) is a Jupyter notebook that explains what problems you were solving, what things you tried in order to solve them, and what approaches you found to work best. This notebook should not include large amounts of code. Instead, it should focus on explanatory markdown text and plots that help to visualize your results. Try to pull as much code as possible out into other files, and then in the presentation notebook either call external functions or read in already-processed data.