# Automation of Biological Research: 02-450/02-750

# Carnegie Mellon University

## Homework 1

*Version: 1.0; updated 1/27/2021*

### Due: February 25 by 11:59pm

**Hand-in:** A single PDF to Gradescope that contains the following items:

1. A cover page that lists your name and Andrew id
   - If you worked on a team, the name(s) and Andrew id(s) of your teammate(s).
     - Teams cannot be larger than 3 people.
     - Note: Each team should only hand in one pdf. It does not matter who does the actual upload.
     - Indicate your partners in Gradescope
   - Points will be deducted if you do not have a cover page or fail to indicate your partners
2. A PDF export of the Jupyter notebook for questions 1 and 2

You can combine all the PDFs into one using Adobe Acrobat, or similar tool.

### Overview

This homework consists of 2 questions, 50 points each. The purpose of the homework is to become familiar with basic Active Learning algorithms. In question 1 you will implement different sampling algorithms such as Random Sampling, Uncertainty Sampling, Density-based Sampling and Query-by-Committee using python library for Active Learning called modAL and analyze your results. In question 2 you will implement Importance Weighted Active Learning algorithm (IWAL) from scratch, perform some experiments on synthetic data, and compare the results with conventional machine learning when no Active Learning is used. You will be using Python and Jupyter notebook to complete this assignment. Jupyter notebook provides a convenient interface for running experiments, visualizing results and adding comments.

### Installation instructions

First, you need to install Python with all the required packages. We recommend that you use the Anaconda package manager. Bellow are step-by-step installation instructions:

1. If you are using Windows, we recommend installing GitBash. Download Git for Windows from https://gitforwindows.org/ and follow the installation instructions.

2. Install Miniconda for Python 3.7 from https://docs.conda.io/en/latest/miniconda.html. Follow instructions depending on your operating system. Don't forget to add Anaconda to my PATH environment.

3. Open Terminal (or GitBash for Windows users).

4. Create new Anaconda environment for Python 3.7:

```
conda create -n modAL python=3.7
```

5. Active the created environment:

```
conda activate modAL6.
```

6. Install required packages by running the commands:

```
pip install modAL
```

```
conda install numpy scikit-learn pandas matplotlib jupyter
```

7. Start a Jupyter notebook server:

```
jupyter notebook
```

8. You are all set!

Below are the links for additional resources on the libraries and tools that you will need for this homework. Feel free to look at the documentation and examples or just google your problem (may StackOverflow be with you!):

- Python: https://docs.python.org/3.7/
- Anaconda: https://docs.conda.io/projects/conda/en/latest/user-guide/index.html
- Jupyter: https://jupyter.readthedocs.io/en/latest/
- modAL: https://github.com/modAL-python/modAL

- NumPy: https://docs.scipy.org/doc/numpy/reference/
- Scikit-learn: https://scikit-learn.org/stable/
- Pandas: https://pandas.pydata.org/pandas-docs/stable/

## Question 1, Intro to Active Learning with modAL (50 points)

This first question will serve as an introduction to working with modAL in order to implement some of the basic active learning concepts we have covered thus far in the lectures. This includes pool-based sampling, as well as query by committee. Conveniently, modAL includes the ActiveLearner and Committee classes, which helps make both of these steps simpler. You will be required to instantiate the appropriate objects using these classes, utilize modAL supported query strategies, as well as implement one of your own. In doing so, you will perform some of your first active learning experiments, and report the results as asked below. You are provided with an imaging data set “Data.csv”. This data comprises 500 samples each with 26 features. Each sample is labeled with one of ten possible subcellular locations. You are also given the Jupyter Notebook template for completing this assignment. Notice there is a cell that will load and partition the data for you.

**Complete the code under ###TODO in each cell and produce the required plots.** Feel free to define any helper functions as you see fit. You may import and use anymodules in scikit-learn and NumPy to help with your implementations (modAL is built specifically to support and mimic scikit-learn functionality)

## Part 1.1 Random Sampling (5 pts. total)

Create a random_query() function that provides a method for randomly selecting a sample to query from the oracle, and must be consistent with the query methods expected by modAL when called. **Do not change the random seed.** Perform random sampling for 100 calls to the oracle with each ActiveLearner. After each call to the oracle, measure the accuracy of each learner on predictions across the test sets (X_test and y_test). Report the final accuracy and make a plot with query number as the x-axis, and accuracy as the y-axis (Some code is provided that you may use if you want).

Do this for using both logistic regression and random forest classifier as an estimator for the ActiveLearner (2.5 points each)

**Hint***:* Check the modAL documentation for how to call an ActiveLearner. Within the documentation, check the Extending modAL section for a tutorial on implementing a custom query strategy.

## Part 1.2: Uncertainty Sampling (15 pts. total)

Create an ActiveLearner with Uncertainty sampling as the sampling strategy for both logistic regression and random forest classifier as an estimator (7.5 points each). Report the final accuracy and plot the classification accuracy with respect to the number of labels queried for both the uncertainty and random sampling for each estimator.

## Part 1.3: Density-based sampling (15 pts. total)

Create an ActiveLearner with information density for both logistic regression and random
forest classifier as an estimator (7.5 points each). Use the Euclidean metric as a measure
of similarity and uncertainty for utility function. Set $\beta = 1$. Report the final accuracy and
plot the classification accuracy with respect to the number of labels queried for both the
uncertainty and random sampling for each estimator.

## Part 1.4: Query-By-Committee with max disagreement sampling (10 pts. total)

Finally, we will now use modAL to implement query by committee. We will form a committee with max disagreement sampling. Within modAL, forming a committee is as easy as creating a list of ActiveLearners, and passing this along with the query method to Committee(). The Committee object can then be used exactly as the ActiveLearner objects we used in the previous parts (eg. Use *Committee.query(X,y)* or *Committee.teach(X,y)*, etc.). Construct the committee described above, with logistic regression classifier as the first member and as random forest as the second. As in the previous section, perform 100 queries to the oracle, and plot number of queries against prediction accuracy across the test sets and compare with both the random forest and logistic regression classifier with a random sampling strategy.

### Part 1.5 (5 pts.)

As you've seen in this example, the incremental improvement of accuracy by querying new labels vary across different sampling strategies and estimator. Given you only have enough labeling budget of 20 samples, which query strategy/estimator would you use. If the goal was to achieve an 85%

accuracy, which strategy/estimator would you use.

**Question 2, Implement IWAL algorithm (50 points)**
The purpose of this question is to implement *Importance Weighted Active Learning* (IWAL)[1] algorithm. For this question, you will not use modAL, but instead will implement IWAL routine from scratch using scikit-learn, NumPy and native Python. In this question, we will use a simple synthetic dataset for a binary classification problem. Each data point has only 2 features. The dataset is provided in 2 files -- “data_iwal.npy”, which contains features and “labels_iwal.npy”, which contains labels. For simplicity, you will implement bootstrapping rejection sampling subroutine with logistic regression and hinge loss.

[1]Beygelzimer, A., Dasgupta, S., & Langford, J. (2009, June). Importance weighted active learning. In *Proceedings of the 26th annual international conference on machine learning* (pp. 49-56).

**Complete the code under ###TODO in each cell and produce the required plots.**
Feel free to define any helper functions as you see fit. You may import and use any
modules in scikit-learn and NumPy to help with your implementations.

**Part 2.1 (5 points)** Read IWAL paper and answer the following questions:

1. What is the idea behind IWAL algorithm?
2. What are the assumption behind IWAL algorithm?
3. What are the pros and cons of IWAL algorithm?

**Part 2.2 (15 points)** In this part you will implement a function that performs a single query of Algorithm 1 IWAL (subroutine rejection-sampling) from the paper. A more detailed description of the function is provided inside Jupyter notebook with the assignment.

**Part 2.3 (15 points)** Implement bootstrapping rejection sampling subroutine described in section 7.2 in the paper. A more detailed description of the function is provided inside Jupyter notebook with the assignment.

**Part 2.4 (10 points)** In this part you need to organize all implemented functions into a single pipeline. The pipeline should include initialization (dictionary for history, initial set of hypothesis H), training initial hypothesis on bootstrapped data, performing queries and recording loss.

**Part 2.5 (5 points)** Compare results of the model trained with IWAL algorithm with the ones of the model trained with no Active Learning. Comment on your observations.