CS480 – Project Phase 3
Assigned on: Monday, 11/21/2015
Due: Sunday, 12/06/2015, 11:59pm
Please submit your solutions through black board assignment page.

Description (the same from phase 1)

You are introduced an electronic product that has a $*value* and a $*price*. Each product can be in an *Excellent* condition, which works flawlessly, or in a *Trash* condition, where, hmm, it is trash. If you buy a product and if it is *Excellent*, your wealth increases by $*value*-$*price*. If, however, the product turns out to be *Trash*, your wealth decreases by $*price.* You do not know whether a product is *Excellent* or *Trash* until you buy it; however, there will be clues for you to make an informed decision. Your objective is to increase your wealth as much as possible.

The project will consist of multiple phases. The number of phases is tentatively set to 4. Depending on time and interest, we might need to eliminate one of the existing phases or add a new phase.

According to the syllabus (http://www.cs.iit.edu/~mbilgic/classes/fall15/cs480/CS480-FALL15-SYLLABUS.pdf), the project is worth 20% of the grade.

Important: the project is different from homeworks and programming exercises. The homeworks and programming are based on what is already covered in the lectures. The project, however, is not based on past materials in class; instead, it requires you to do research and innovative thinking. Therefore, you need to start each phase of the project as soon as it is assigned.

Phase 3

In this phase, unlike phase 2

1. You'll be using scikit-learn's (http://scikit-learn.org/stable/) BernoulliNB, LogisticRegression, and SVC classifiers.
2. Instead of fitting one classifier, you'll implement an algorithm such that given a training dataset <X_train, y_train> and a validation set <X_val, y_val> fits the best classifier on <X_train, y_train> where best is defined as the best classifier that has the best performance measure when tested on the *validation* set.

You are provided two python files and several product files (csv files):

- agents.py
    - This file has the base agent Agent and a baseline agent called Agent_single_sklearn. Agent_single_sklearn simply fits one classifier and ignores the validation set.
- simulate_agents_phase3.py
    - This file simulates the baseline agents and your agent.

The product files are triplets of files named as dataseti_train.csv, dataseti_val.csv, and dataseti_test.csv where i ranges between 1 and 3. These files are used to create X_train, y_train, X_val, y_val, and X_test, and y_test, and the format of the files is the same as in phase 2. Your agent is provided the X_train, y_train, X_val, y_val during training and it is tested using X_test. During testing, y_test is not revealed to the agent.

Your task is to create a file called agent_hawkusername.py and an agent in it called Agent_hawkusername where hawkusername is your Hawk username. For example, for me, it would be agent_mbilgic.py file that has Agent_mbilgic. Your agent should inherit the base Agent and override one method:

1. choose_the_best_classifier(self, X_train, y_train, X_val, y_val).
   Task: Among three classifiers, your agent should choose the 'best' sklearn classifier, which when trained on <X_train, y_train>, performs the best on <X_val, y_val>. See the documentation in the agent.py file for details.

Rules:

1. Your agent is not allowed to read the train, validation, and test files directly. Your agent is not allowed to access y_test at any time.
2. Your agent should not hardcode the chosen classifiers based on the dataset names.
3. Your strategy should be general enough to be applicable to any train, validation, and test triplets. We will probably test your agent on different datasets.
4. If you like, your agent can override the default constructor as long as we can still call the constructor using your agent's name and everything works as expected.
5. If you need, you can create additional classes and methods in your agent_hawkusername.py file.

Submit a zip file that contains two (and only two) files:

1. agent_hawk_username.py file.
2. A report in .pdf format. The report should have:
    a. A detailed description of the choose_the_best_classifier method.
    b. The simulation results comparing your agent to the baseline agents. (This is the output from the simulate_agents_phase3.py file after you add
        i. Two more Agent_single_sklearn agents and
        ii. your agent to the agents list. See lines 66 through 74 in the simulate_agents_phase3.py file.

The simulation code evaluates the agents in three categories: agent wealth (the higher the better), log-loss (the smaller the better), and 0/1 error (the smaller the better). Your agent should have the same performance as the best performing baseline agent per each of the three datasets; that is, it should be able to choose the correct classifier for each of the datasets. Our primary performance measure is wealth. The wealth measure always trumps the logloss, 0/1 loss, and other performance measures.

Please remember to put your files (agent_hawkusername.py and report.pdf) in a folder, zip it, and submit it.