PROJECT 4: DEFEAT LEARNERS

DUE DATE

09/27/2020 11:59PM Anywhere on Earth time

REVISIONS

This assignment is subject to change up until 3 weeks prior to the due date. We do not anticipate changes; any changes will be logged in this section.

OVERVIEW

This assigment counts towards 5% of your overall grade.

For this homework you will generate data that you believe will work better for one learner than another. This will test your understanding of the strengths and weaknesses of various learners. The two learners you should aim your datasets at are:

- A decision tree learner with leaf_size = 1 (DTLearner). Note that for testing purposes we will use our implementation of DTLearner
- The LinRegLearner provided as part of the repo.

Your data generation should use a random number generator as part of its data generation process. We will pass your generators a random number seed. Whenever the seed is the same you should return exactly the same data set. Different seeds should result in different data sets.

The provided grading script may be similar to, but will not include all of the instructor tests.

TEMPLATE

Instructions:

- Download the appropriate zip file File:Defeat_Learners_2020Fall.zip
- You should see the following files and directory
  - `defeat_learners/` the assignment directory
  - `defeat_learners/gen_data.py` An implementation of the code you are supposed to provide: It includes two functions that return a data set, and a third function that returns a user ID. Note that the data sets those functions return DO NOT satisfy the requirements for the homework. But they do show you how you can generate a data set.
  - `defeat_learners/LinRegLearner.py` Our friendly, working, correct, linear regression learner. It is used by the grading script. Do not rely on local changes you make to this file, as you may only submit `gen_data.py`.
  - `defeat_learners/DTLearner.py` A working, but INCORRECT, Decision Tree learner. Replace it with your working, correct DTLearner.
  - `defeat_learners/testbest4.py` Code that calls the two data set generating functions and tests them against the two learners. Useful for debugging.
  - `defeat_learners/grade_best4.py` The grading script; for more details see here: ML4T_Software_Setup

TASKS

Implement Dataset Functions

Create a Python program called gen_data.py that implements two functions. The two functions should be named as follows, and support the following API:

```
X1, Y1 = best_4_lin_reg(seed = 5)
X2, Y2 = best_4_dt(seed = 5)
```

- **seed** Your data generation should use a random number generator as part of its data generation process. We will pass your generators a random number seed. Whenever the seed is the same you should return exactly the same data set. Different seeds should result in different data sets.

Linear Regression Dataset

best4LinReg() should return data that performs significantly better (see rubric) with LinRegLearner than DTLearner.

Each data set should include from 2 to 10 columns in X, and one column in Y. The data should contain from 10 (minimum) to 1000 (maximum) rows.

Decision Tree Dataset

best4DT() should return data that performs significantly better with DTLearner than LinRegLearner.

Each data set should include from 2 to 10 columns in X, and one column in Y. The data should contain from 10 (minimum) to 1000 (maximum) rows.

Implement author() (Up to 10 point penalty)

You should implement a function called `author()` that returns your Georgia Tech user ID as a string. This is the ID you use to log into Canvas. It is not your 9 digit student number. Here is an example of how you might implement author():

```
def author():
        return 'tb34' # replace tb34 with your Georgia Tech username.
```

Implementing this method correctly does not provide any points, but there will be a penalty for not implementing it.

WHAT TO TURN IN

Be sure to follow these instructions diligently!

Gradescope:

- (SUBMISSION) Project 4: Defeat Learners
  - Your code as `gen_data.py`

We WILL NOT use your DTLearner, or LinRegLearner, so do not submit them.

Do not submit any other files.

You are only allowed 3 submissions to **(SUBMISSION) Project 4: Defeat Learners** but unlimited resubmissions are allowed on **(TESTING) Project 4: Defeat Learners.**

Note that Gradescope does **not** grade your assignment live; instead, it pre-validates that it will run against our batch autograder that we will run after the deadline. There will be **no** credit given for coding assignments that do not pass this pre-validation.

Refer to the Gradescope Instructions for more information.

RUBRIC

Report

No report

Code

See Auto-Grader

Auto-Grader [100 points]

Deductions:

- Does either dataset returned contain fewer or more than the allowed number of samples? (-20 points each)
- Does either dataset returned contain fewer or more than the allowed number of dimensions in X? (-20 points each)
- When the seed is the same does the best4LinReg dataset generator return the same data? (-20 points otherwise)
- When the seed is the same does the best4DT dataset generator return the same data? (-20 points otherwise)
- When the seed is different does the best4LinReg dataset generator return different data? (-20 points otherwise)
- When the seed is different does the best4DT dataset generator return different data? (-20 points otherwise)
- Is the author() method implemented? (-10 points if not)
- Does the code attempt to import a learner? (-10 points if so)

For best4LinReg (1 test case):

- We will call best4LinReg 15 times, and select the 10 best datasets. For each successful test +5 points (total of 50 points)
- For each test case we will randomly select 60% of the data for training and 40% for testing.
- Success for each case is defined as: RMSE LinReg < RMSE DT * 0.9

For best4DT (1 test case):

- We will call best4DT 15 times, and select the 10 best datasets. For each successful test +5 points (total of 50 points)
- For each test case we will randomly select 60% of the data for training and 40% for testing.
- Success for each case is defined as: RMSE DT < RMSE LinReg * 0.9

REQUIRED, ALLOWED & PROHIBITED

Required:

- No reading of data from files.
- Your project must be coded in Python 3.6.x.
- Your code must be submitted to Gradescope in the appropriate Gradescope assignment.
- Your code must run in less than 5 seconds per test case.
- The code you submit should NOT include any data reading routines. You should generate all of your data within your functions.
- The code you submit should NOT generate any output: No prints, no charts, etc.
- Reference any code used in the "Allowed" section in your code. At minimum it should have the link/filename/video name of where it came from.

Allowed:

- You can develop your code on your personal machine, but it must also run successfully on Gradescope.
- Your code may use standard Python libraries.
- You may use the NumPy, SciPy, matplotlib and Pandas libraries. Be sure you are using the correct versions.
- Code provided by the instructor, or allowed by the instructor to be shared.
- Cheese.

Prohibited:

- Any reading of data files.
- Any libraries not listed in the "allowed" section above.
- Any code you did not write yourself.
- Any Classes (other than Random) that create their own instance variables for later use (e.g., learners like kdtree).

- Code that includes any data reading routines. The provided testlearner.py code reads data for you.
- Code that generates any output when verbose = False: No prints, no charts, etc.
- Absolute import statements of the **current** project folder such as `from defeat_learners import XXXX` or `import defeat_learners.XXXX`
- Extra directories (manually or code created)
- Extra files not listed in “WHAT TO TURN IN”
- Ducks and wood.