CS589: Machine Learning - Spring 2020

Homework 1: Regression

Assigned: Tuesday, Feb 11. Due: Friday, Feb 21 at 05:00pm

In this assignment you will create regression models that predict a real number for each possible input. An example is shown in Fig. 1 in which the blue dots are the data, and the red line is the learned model used to predict, given an input value (x axis) the corresponding output value (y axis).

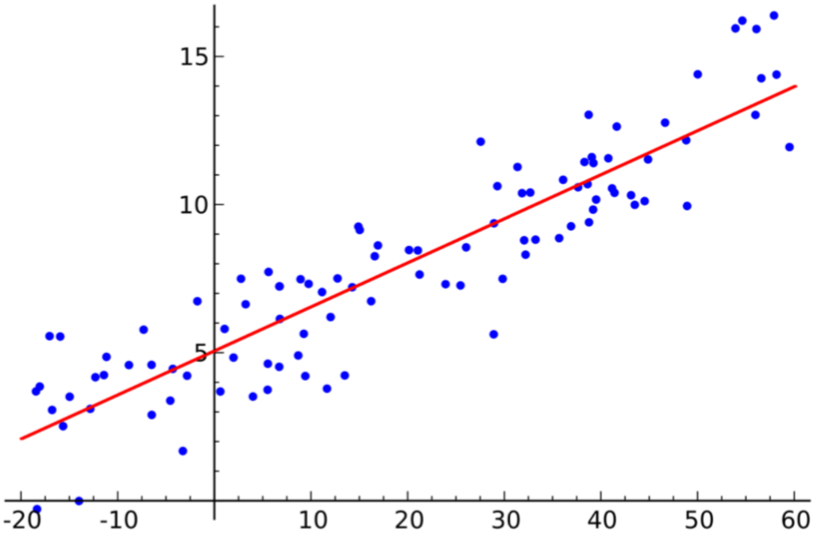

Figure 1: Example of linear regression.

When fitting regression models, we often face choices, such as a regularization constant, or what depth a decision tree might have. We sometimes call these choices *model selection*. We wish to choose to make the *generalization performance* as good as possible. In this assignment, you will explore the use of cross-validation to perform model selection.

Getting Started: In this assignment, you will train and evaluate different regression models on two datasets. Please setup `Python` **`3.7`** via `Anaconda` on your personal machine. Instructions for this has been provided seperately. Download the homework file `HW01.zip`. Unzipping this folder will create the directory structure shown below,

```
HW01
--- Data
     |--dataA
     |--dataB
```

```
--- Submission
    |--Code
    |--Predictions
       |--dataA
       |--dataB
    |--Report
```

The data files for each data set are in `Data` directory respectively. You will write your code under the `Submission/Code` directory. Make sure to put the deliverables (explained below) into the respective directories.

Deliverables: This assignment has three types of deliverables: a report, code files, and Kaggle submissions.

- **Report:** The solution report will give your answers to the homework questions (listed below). Try to keep the maximum length of the report to 5 pages in 11 point font, including all figures and tables. Reports longer than five pages will only be graded up until the first five pages. You can use any software to create your report, but your report must be submitted in PDF format.
- **Code:** The second deliverable is the code that you wrote to answer the questions, which will involve implementing a regression models. Your code must be `Python 3.7` (no iPython notebooks or other formats). You may create any additional source files to structure your code. However, you should aim to write your code so that it is possible to re-produce all of your experimental results exactly by running `python run_me.py` file from the `Submissions/Code` directory.
- **Kaggle Submissions:** We will use `Kaggle`, a machine learning competition service, to evaluate the performance of your regression models. You will need to **register** on `Kaggle` using a `umass.edu` email address to submit to Kaggle (you can use any user name you like). You will generate test prediction files, save them in Kaggle format (helper code provided called `Code/kaggle.py`) and upload them to Kaggle for scoring. Your scores will be shown on the Kaggle leaderboard, and **12%** of your assignment grade will be based on how well you do in these competitions. The Kaggle links for each data set are given under respective questions.

Submitting Deliverables: When you complete the assignment, you will upload your report and your code using the `Gradescope.com` service. Here are the steps:

1. Place your final code in `Submission/Code`, and the Kaggle prediction files for your best-performing submission only for each data set in `Submission/Predictions/<Data Set>/best.csv`
2. Create a **zip** file of your submission directory, `Submission.zip` (No rar, tar or other formats).
3. Upload this single zip file on Gradescope as your solution to the `HW01-Regression-Programming` assignment. Gradescope will run checks to determine if your submission contains the required files in the correct locations.
4. Upload your pdf report to the `HW01-Regression-Report` assignment. When you upload your report please make sure to select the correct pages for each question respectively. Failure to select the correct pages will result in point deductions.
5. The submission time for your assignment is considered to be the latest of the submission timestamps of your code, report and Kaggle submissions.

IPython/Jupyter Notebooks: Some students have requested to be able to submit an IPython notebook to save time. It is allowed to use IPython to prepare the report, but not to write your code. Thus using an IPython notebook doesn't change any of the instructions above. You still must export the notebook to a .pdf and upload that .pdf to gradescope. The same rules apply as if you prepared the report using any other writing system! Your code should not be in the notebook itself (other than perhaps a tiny command like plot_answer_2b() before each plot).

Academic Honesty Policy: You are required to list the names of anyone you discuss problems with on the first page of your solutions. This includes teaching assistants or instructors. Copying any solution materials from external sources (books, web pages, etc.) or other students is considered cheating. To emphasize: no detectable copying is acceptable, even, e.g., copying a single sentence from an outside source. Sharing your code or solutions with other students is also considered cheating. Any detected cheating will result in a grade of -100% on the assignment for all students involved (negative credit), and potentially a grade of F in the course.

Note: You cannot use any of sklearn's inbuilt cross-validation functions. One of the goals of this assignment is for you to write one yourself. You are allowed to use `Kfold`, but no other existing model selection functions. We will perform checks and points will be deducted accordingly.

Data Sets: In this assignment, you will experiment with two different datasets: dataA and dataB. The basic properties of these data sets are shown below.

Dataset	Training Cases	Test Cases	Dimensionality
dataA	1342	617	6
dataB	5482	2513	7

Each data set has been split into a training set and a test set and stored in NumPy binary format. The provided `Submission/Code/run_me.py` file provides example code for reading in all data sets. The following are the required Kaggle links to the respective competitions:

- dataA → `https://www.kaggle.com/t/5402162443f04f4a89d0fec1d7742046`
- dataB → `https://www.kaggle.com/t/41ff7f7569c94983b34a8824ca46d71f`

Questions:

1. (*0 points*) **Collaboration statement**:
Please list the names of anyone you discussed the assignment with, including TAs or instructors.

2. (*26 points*) **Decision trees**:

a. (*6 pts*) What is the criteria used to select a variable for a node when training a decision tree? Is it optimal? If yes, explain why it is optimal. If no, explain why is the optimal ordering not used.

b. (*10 pts*) The metric we would use to evaluate models for this question is mean absolute error or MAE. For $i = 1..n$ observations y_i, and corresponding predictions $\hat{y}_i$, the MAE is defined as

$$MAE(y,\hat{y}) = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}_i|$$

where || is the modulus or absolute function
For this question you would be using cross-validation to estimate the performance of different configurations of decision trees on dataset dataA.

- Train 5 different decision trees with the maximum depths set to {3, 6, 9, 12, 15}
- Evaluate the out of sample error rate of each model using 5-step cross validation
- Measure the time (in milliseconds) that each model takes to do cross-validation
- Report the obtained out of sample errors in a table
- Present the run time of each model using a graph
- Choose the configuration/model with the lowest out of sample error. Using that configuration train a new model on the entire training data.
- Use the model learnt in the previous step to predict the target values for the provided test set
- kaggleize your prediction (to make them kaggle-friendly). you can use kaggle.py provided in the zip file
- Upload predictions to kaggle and report the MAE obtained in the report
- Was the out of sample error prediction close to the test error? Provide clear answer in the report.

Make sure that the report states which model was chosen and what was its predicted out-of-sample error. Also make sure your graphs are marked and labelled (with units where appropriate).

c. (*10 pts*) Repeat the previous question (1.b) with the dataset dataB using the following maximum depths {20, 25, 30, 35, 40}.

3. (*15 points*) Cross-validation:

a. (*9 pts*) Suppose that training a model on a dataset with K samples takes K units of time, and that you are given a dataset with N samples. In order to perform cross-validation, you split this dataset into chunks of M samples (that is, N/M subsets). Ignoring the time that it takes to evaluate the model, what is the time complexity of performing cross-validation with this partition? Give the answer using *big-O* notation with respect to N and M. What happens if $M = 5$? And when $M = N/2$?

b. (*6 pts*) Can you mention one advantage of making M small?

4. (*16 points*) Nearest neighbors:

a. (*8 pts*) For the dataset dataA train 5 different nearest neighbors regressors using the following number of neighbors {3, 5, 10, 20, 25}. Using 5-fold cross-validation, estimate the out of sample error for each model, and report them using a table. Choose the model with lowest estimated out of sample error, train it with the full training set, predict the outputs for the samples in the test set and report the MAE (follow the steps as question 1 to report MAE). Is the predicted out of sample error close to the real one? Make sure that your report clearly states which model was chosen and what was the predicted out of sample error for it.

b. (*8 pts*) Repeat the previous exercise with the dataset dataB.

5. (*26 points*) **Linear model:**

a. (*6 pts*) What is the purpose of penalties (regularization) used in Ridge and Lasso regression?

b. (*10 pts*) Train a Ridge and a Lasso linear model using the dataset dataA with the following regularization constants $\alpha = \{10^{-6}, 10^{-4}, 10^{-2}, 1, 10\}$ for each. Using 5-fold cross-validation, estimate the out of sample error for each model, and report them using a table. Choose the model with lowest estimated out of sample error (out of the 10 trained models), train it with the full training set, predict the target outputs for the samples in the test set and report the MAE (follow the steps as question 1 to report MAE). Make sure that your report clearly states which model was chosen and what was the predicted out of sample error for it.

c. (*10 pts*) Repeat the previous exercise with the dataset dataB for $\alpha = \{10^{-4}, 10^{-2}, 1, 10\}$ (2x4 = 8 models only).

6. (*12 points*) **Kaggle Competition:**

a. (*6 pts*) Train a regression model of your choice from either decision trees, nearest neighbors or linear models on the dataA dataset. Pick ranges of hyperparameters that you would like to experiment with (depth for decision trees, number of neighbors for nearest neighbors and regularization constants for linear models). Also pick k in k-fold cross-validation used to tune hyperparameters. Your task is to make predictions on the test set, kagglize your output and submit to kaggle public leadership score (limited to ten submissions per day). Make sure to list your choice of regression model, hyperparameter range, k in k-folds, your final hyperparameter values from cross-validation and best MAE. Save the predictions associated to the best MAE under `Submissions/Predictions/<Data set>/best.csv`. Kaggle submission should be made at the aforementioned link for dataset dataA .

b. (*6 pts*) Repeat the previous question with other dataset dataB . Kaggle submission should be done at the aforementioned link for dataset dataB .

7. (*5 points*) **Code Quality:**

a. (*5 pts*) Your code should be sufficiently documented and commented that someone else (in particular the graders) can understand what each method is doing. Adherence to a particular Python style is not required, but if you need a refresher see the Google Style Guide [1]. You will be scored on the quality of your code.

[1] https://google.github.io/styleguide/pyguide.html