CS589: Machine Learning - Fall 2018

Homework 1: Regression

Assigned: September 12th, 2018   Due: September 27, 2018

Getting Started: In this assignment, you will train and evaluate different regression models on two datasets. Please install Python 3.6 via Anaconda on your personal machine. For this homework you will only be using numpy, scipy, sklearn and matplotlib packages. Download the homework file HW01.zip via Piazza. Unzipping this folder will create the directory structure shown below:

```
HW01
--- HW01.pdf
--- Data
    |--PowerOutput
    |--IndoorLocalization
--- Submission
    |--Code
    |--Figures
    |--Predictions
       |--PowerOutput
       |--IndoorLocalization
```

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:

- **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 3.6 (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 an '@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. 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. If you used Python to generate report figures, place them in Submission/Figures. Finally, create a zip file of your submission directory, Submission.zip (NO rar, tar or other formats). 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. Finally, 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. The submission time for your assignment is considered to be the later of the submission timestamps of your code, report and Kaggle submissions.

Academic Honesty Statement: Copying solutions from external sources (books, internet, etc.) or other students is considered cheating. Sharing your solutions with other students is also considered cheating. Posting your code to public repositories like GitHub, stackoverflow is also considered cheating. Any detected cheating will result in a grade of -100% on the assignment for all students involved, and potentially a grade of F in the course.

Task:

Regression: The regression task consists on finding a model that, for every input, outputs a value. While for classification tasks this output is one of several possible classes, for regression problems it is a real number. An example is shown in Fig. 1[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).

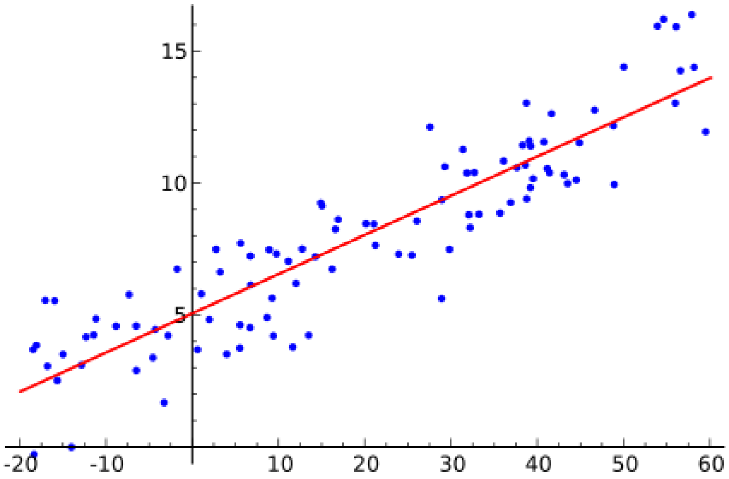

Figure 1: Example of linear regression

Model selection: For each trained model you will try different parameters. Which set of parameters provides the best performance? Is the one that provides a lower training error? Not necessarily, lower training error

[1] Image extracted from https://en.wikipedia.org/wiki/Regression_analysis

does not mean lower testing error (or better generalization). For this homework we will be using mean absolute error (MAE). One way to estimate the best model is via model selection. In other words, after training a model one would like to know how well will that model generalize, *i.e.* how well will the model perform on new unseen data. This cannot be known, but can be estimated via cross-validation. This consists of splitting the training data into K pieces, training using a subset of $K-1$ pieces, and evaluating the final performance on the unused piece. This is repeated K times, leaving out for testing one different piece each time. The average of the accuracy obtained in this K simulations can be used as an estimation for the out-of-sample error.

Note: One of the goals of this assignment is for you to write your own cross-validation implementation. You are allowed to sklearn.model_selection.Kfold, but you *may not* use sklearn.model_selection.GridSearchCV. (If necessary, will deduct points for violating this rule.)

Models: You will train decision trees with different maximum depths, nearest neighbors with different number of neighbors, and linear models with different regularization parameters.

Data: You will work with two datasets:

- Estimation of the power output of a power plant (7176 samples for training, 2392 samples for testing).

  Attributes: Measurements of the four variables: temperature, pressure, humidity and exhaust vacuum

  Output: Electrical energy output

- Indoor localization system (19937 samples for training, 1111 labels for testing).

  Attributes: Measurements of 400 sensors inside a building (each sensor measures proximity, and outputs a specific value if it does not sense anything)

  Output: Position in a grid of the user with respect to a fixed coordinate system (two values, x and y)

The target outputs for the test sets are not provided. You have to predict them and upload the results of the best model to Kaggle (best model chosen using cross-validation). To help you get started we have provided you with sample code in Code/run_me.py file. This file has functions to read in data files and to compute MAE given predicted and original outputs.

Questions:

1. (*24 points*) Decision trees:

(4) a. 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.

(10) b. For the power plant dataset train 5 different decision trees using the following maximum depths $\{3, 6, 9, 12, 15\}$. Using 5-fold cross-validation, estimate the output of sample error for each model, and report them using a table. Measure the time (in milliseconds) that it takes to perform cross-validation with each model and report the results using a graph (make sure to label both axis; points will be deducted for missing labels). Choose the model with lowest estimated out of sample error, train it with the full training set and predict the target outputs for the samples in the test set. Following this Kaggleize your predictions (i.e. write them out in Kaggle CSV file), upload your submission to Kaggle to `https://inclass.kaggle.com/c/589-hw1-regression-power-plant` and report the MAE on the public leaderboard. Is the predicted out of sample error close to the test error? Make sure that your report clearly states which model was chosen and what was the predicted out of sample error for it.

(10) c. Repeat the previous question (1.b) with the indoor localization dataset using the following maximum depths $\{20, 25, 30, 35, 40\}$.

Use Kaggle url `https://www.kaggle.com/c/589-hw1-indoor-localization`.

2. (*16 points*) **Nearest neighbors:**

(8) a. For the power plant dataset 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.

(8) b. Repeat the previous question with the indoor localization dataset.

3. (*20 points*) **L1 versus L2 regularization:**

(8) a. L1 penalties, unlike L2 penalties, cause *sparsity* in the learned weight parameters for a model. We'll explore this in a simplified setting. Consider a one-parameter, "bias-term-only" model, where the task is to make a constant prediction w to predict two different datapoints, $y_1 = 2$ and $y_2 = 4$. The L2 regularized objective G to be minimized is:

$$G(w, \lambda) = (w - 2)^2 + (w - 4)^2 + \frac{1}{2}\lambda w^2$$

Derive an analytic form for the optimal w solution that minimizes G in terms of λ.

(8) b. Now we consider an L1 penalty instead, and call the overall L1-regularized objective H:

$$H(w, \lambda) = (w - 2)^2 + (w - 4)^2 + \lambda|w|$$

Simple calculus doesn't quite work to solve this, since the absolute value function is not differentiable everywhere. But it's possible to work it out by separating the domains $w \geq 0$ versus $w < 0$:

$$H(w, \lambda) = \begin{cases} (w - 2)^2 + (w - 4)^2 + \lambda w & \text{if } w \geq 0 \\ (w - 2)^2 + (w - 4)^2 - \lambda w & \text{if } w < 0 \end{cases}$$

For our objective, there is no optimal solution when $w < 0$ (note that λ is always ≥ 0). Derive the optimal w that minimizes H, in terms of λ when $w \geq 0$.

(4) c. Based on these derivations, explain how large values of λ impact w differently when L1 versus L2 penalties are applied.

4. (*24 points*) Linear model:

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

(10) b. Train a Ridge and a Lasso linear model using the power plant dataset 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.

(10) c. Repeat the previous question with the indoor localization dataset for $\alpha = \{10^{-4}, 10^{-2}, 1, 10\}$ (8 models only).

5. (*12 points*) Kaggle Competition:

(6) a. Train a regression model of your choice from either decision trees, nearest neighbors or linear models on the power plant 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). 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 to:

`https://www.kaggle.com/c/589-hw1-regression-power-plant`

(6) b. Repeat the previous question with the indoor localization dataset. Kaggle submission should be made to:

`https://www.kaggle.com/c/589-hw1-indoor-localization`.

6. (*4 points*) Code Quality:

(4) Your code should be sufficiently documented and commented that someone else (in particular, the TAs and graders) can easily understand what each method is doing. Adherence to a particular Python style guide is not required, but if you need a refresher on what well-structured Python should look like, see the Google Python Style Guide: `https://google.github.io/styleguide/pyguide.html`. You will be scored on how well documented and structured your code is.