CSE 158/258, Fall 2022: Homework 1

Instructions

Please submit your solution by Monday Oct 10. Submissions should be made on gradescope. Please complete homework individually.

You should submit two files:

answers_h1.txt should contain a python dictionary containing your answers to each question. Its format should be like the following:

```python
{ "Q1": 1.5, "Q2": [3,5,17,8], "Q2": "b", (etc.) }
```

The provided code stub demonstrates how to prepare your answers and includes an answer template for each question.

homework1.py A python file containing working code for your solutions. The autograder will not execute your code; this file is required so that we can assign partial grades in the event of incorrect solutions, check for plagiarism, etc. Your solution should clearly document which sections correspond to each question and answer. We may occasionally run code to confirm that your outputs match submitted answers, so please ensure that your code generates the submitted answers.

You will need the following files:

Homework 1 stub: https://cseweb.ucsd.edu/classes/fa22/cse258-a/stubs/

GoodReads Young Adult Reviews: https://cseweb.ucsd.edu/classes/fa22/cse258-a/data/young_adult_10000.json.gz

Beer Reviews: https://cseweb.ucsd.edu/classes/fa22/cse258-a/data/beer_50000.json

The above are json formatted datasets. Code to read them is included in the stub.

Further code examples for regression and classification are available on the class and textbook webpages. Executing the code requires a working install of Python 2.7 or Python 3 with the scipy packages installed.

Each question is worth one mark unless otherwise specified.

Tasks — Regression:

First, using the (GoodReads) book review data, let’s see whether ratings can be predicted as a function of a few simple features from a review.

1. Train a simple predictor that estimates a rating from the number of times the exclamation mark (!) symbol is used in the review (i.e., `review.count('!')`):

   star rating \( \simeq \theta_0 + \theta_1 \times \text{[number of ‘!’ characters]} \)

   Report the values \( \theta_0 \) and \( \theta_1 \), and the Mean Squared Error of your predictor (on the entire dataset).

2. Re-train your predictor so as to include a second feature based on the length (in characters), i.e.,

   star rating \( \simeq \theta_0 + \theta_1 \times \text{[length]} + \theta_2 \times \text{[number of ‘!’ characters]} \)

   Report the values of \( \theta_0 \), \( \theta_1 \), and \( \theta_2 \), along with the MSE of the new model.

3. Train a model that fits a polynomial function to estimate ratings based on our ‘!’ feature. I.e.,

   star rating \( \simeq \theta_0 + \theta_1 \times \text{[number of ‘!’]} + \theta_2 \times \text{[number of ‘!’]}^2 + \theta_3 \times \text{[number of ‘!’]}^3 \)

   Fit polynomials from degree one to five (i.e., including up to \( \text{[number of ‘!’]}^5 \)) and report the MSE of each.
4. Repeat the above question, but this time split the data into a training and test set. You should split the data into 50%/50% train/test fractions. The *first half* of the data should be in the training set and the *second half* in the test set. Report the MSE of each model on the *test* set.

5. Given a trivial predictor, i.e., \( y = \theta_0 \), what is the best possible predictor (i.e., value of \( \theta_0 \)) in terms of the Mean Absolute Error (MAE), i.e., \( \frac{1}{N} \sum_i |y_i - \theta_0| \)? For your answer report the MAE of your predictor on the test set from the previous question.

**Tasks — Classification:**

In this question, using the *beer review* data, we’ll try to predict user gender based on users’ beer reviews. Load the 50,000 beer review dataset, discarding any entries that don’t include a specified gender.

6. Fit a logistic regressor that estimates gender from the number of ‘!’ characters, i.e.,

\[
p(\text{gender is female}) \simeq \sigma(\theta_0 + \theta_1 \times [\text{number of !}])
\]

Report the number of True Positives, True Negatives, False Positives, False Negatives, and the Balanced Error Rate of the predictor (your answer should be a list of length 5). You may use a logistic regression library with default parameters, e.g., `linear_model.LogisticRegression()` from `sklearn`.

7. Retrain the regressor using the `class_weight='balanced'` option, and report the same error metrics as above.

8. Report the precision@K of your balanced classifier for \( K \in [1, 10, 100, 1000, 10000] \) (your answer should be a list of five precision values).

\(^1\)Although this isn’t best practice (compared to random splitting), it is easier for autograding!