CSE 158, Winter 2017: Homework 1

Instructions

Please submit your solution **by the beginning of the week 3 lecture (Jan 23).** Submissions should be made on **gradescope**. Please complete homework **individually**.

You will need the following files:

50,000 beer reviews : `http://jmcauley.ucsd.edu/cse158/data/beer/beer_50000.json`

UCI Wine Quality Dataset :
`http://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-white.csv`

Code examples : `http://jmcauley.ucsd.edu/cse158/code/week1.py` (regression) and `http://jmcauley.ucsd.edu/cse158/code/week2.py` (classification)

Executing the code requires a working install of Python 2.7 or Python 3 with the scipy packages installed. **Please include the code of (the important parts of) your solutions.**

Tasks — Regression (week 1):

In the first three questions, we'll see how ratings vary across different years in our dataset of 50,000 beer reviews. These questions should be completed on the *entire dataset.*

1. First, let's extract a few simple features from the dataset. What are the years in which reviews were written (see 'review/timeStruct'/'year')? How many reviews were written in each year? What is the average rating ('review/overall') in each year? (1 mark)

2. Next, let's train a predictor that uses the year to predict the overall rating, i.e.,

$$\text{review/overall} \simeq \theta_0 + \theta_1 \times \text{year}.$$

   You may use Python libraries to do so, so long as you include the code of your solutions. What are the fitted values of θ_0 and θ_1? (1 mark)

3. A simple regressor like the one above may not be very realistic—it assumes that ratings get linearly better or linearly worse over time. Can you come up with a better representation of the year variable?
   (a) Describe your representation and write down an equation for it in terms of θ (like the equation from Q2 above) (1 mark).
   (b) Compare the the new representation to the representation from Question 2 in terms of the Mean Squared Error (i.e., report the MSE for both representations) (1 mark).

Next, we'll use the *UCI Wine Quality Dataset* to train a regressor with a few more features. This data is in *CSV* format, and can be processed using the Python CSV library (`https://docs.python.org/3.6/library/csv.html`). See `https://archive.ics.uci.edu/ml/datasets/Wine+Quality` for a few more details about this dataset.

4. Start by training a regressor that uses the first 11 features to predict the last feature ('quality'), i.e.,

$$\text{quality} = \theta_0 + \theta_1 \times \text{`fixed acidity'} + \theta_2 \times \text{`volatile acidity'} + \theta_2 \times \text{`citric acid'} + \ldots + \theta_{11} \times \text{`alcohol'}.$$

   Write down the fitted coefficients and the MSE (1 mark).

5. Of course, what we really want is a method that works well on *unseen* data. Let's split our dataset into 'train' and 'test' portions by taking the first half of the rows for training data and the remaining rows as test data. What are the *train* and *test* MSE when using these splits? (1 mark)

Classification (week 2):

Finally, we'll treat the *Wine Quality* task from above as a *classification* task. Again, split the data so that the first half is used for training and the second half is used for testing as we did for Q5.

To turn this into a classification problem, split the data so that 'negative' examples are those where quality ≤ 5 and 'positive' examples are those where quality > 5.

6. Again using the first 11 features, run an SVM classifier on the data (see the code provided in class) – remember to train on the first half and test on the second half. What is the accuracy (percentage of correct classifications) of the predictor on the train and test data? (1 mark)

7. Finally we'll try and get better performance by using a validation set. Split your data into training, validation, and test sets as follows:

   ```
   X_train = X[:len(X)/2]
   X_valid = X[len(X)/2:3*len(X)/4]
   X_test = X[3*len(X)/4:]
   ```

   The paramater C used to train the SVM controls the regularization level. Run the SVM for all values of $C \in \{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$, and report their training, validation, and test errors. Which of these models do you expect to work best on unseen data? (1 mark)