Artificial Intelligence Spring 2019 Homework 3: Machine Learning

PROGRAMMING

We ask you to implement three small and separate machine learning models. We provide sample input/output file pairs for your reference. The file `visualize.py` contains two functions for plotting data which you may choose to use, modify, reference and steal ideas from at will.

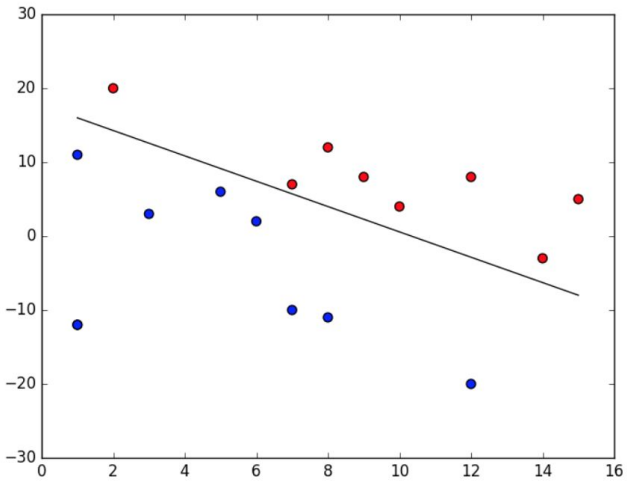
I. Perceptron
II. Linear Regression
III. Clustering

Note: The Python Pandas library helps simplify a lot of the intermediate steps we ask from you below. Key functions include reading and writing to CSVs, matrix operations, and visualizing data. Examples are provided in `visualize.py`.

I. Perceptron

Implement the perceptron learning algorithm (PLA) for a linearly separable dataset. Your starter code includes `input1.csv`, where each line contains `feature_1,feature_2,label`. All values are numeric with labels 1 or -1. Take a look at the data input file. We suggest using matplotlib or pandas.DataFrame.plot (in `visualize.py`) to view data, and pandas.DataFrame.describe to see stats.

Write your PLA in `problem1.py` in python 3. Your program takes a csv of input data and a location to write the output csv with the weights from each iteration of your PLA, written as `weight_1,weight_2,b`. The weights in the last line of your output csv defines the **decision boundary** computed for the given dataset. Formatting is shown in `output1.csv`. NOTE: `output1.csv` values are purely filler and not representative of the values you should get. Feel free to visualize and include a screenshot of your final decision boundary in your README, like the one below:



We will execute your code as follows: `$ python problem1.py input1.csv output1.csv`

II. Linear Regression

Use gradient descent to build a linear regression model for predicting height (m) using age (yr) and weight (kg), using data derived from CDC growth charts data.

Data Preparation and Normalization. Load and understand the data from `input2.csv` [age(years), weight(kg), height(m)] remembering to add a vector column for the intercept at the front of your matrix. You'll notice the features are not on the same scale. What is the mean and standard deviation of each feature? Scale each feature (i.e. age and weight) by its standard deviation, so each scaled feature has a mean of zero. You do not need to scale the intercept. For each feature column, x, use the following formula:

$$x_{\text{scaled}} = \frac{x - \mu(x)}{stdev(x)}$$

Gradient Descent. Implement gradient descent to find a regression model in `problem2.py`. Initialize your β's to zero. Recall the empirical risk and gradient descent rule as follows:

$$R(\beta) = \frac{1}{2n}\sum_{i=0}^{n}(f(x_i) - y_i)^2$$

$$\forall_j \quad \beta_j := \beta_j - \alpha\frac{1}{n}\sum_{i=0}^{n}(f(x_i) - y_i)x_{ij}$$

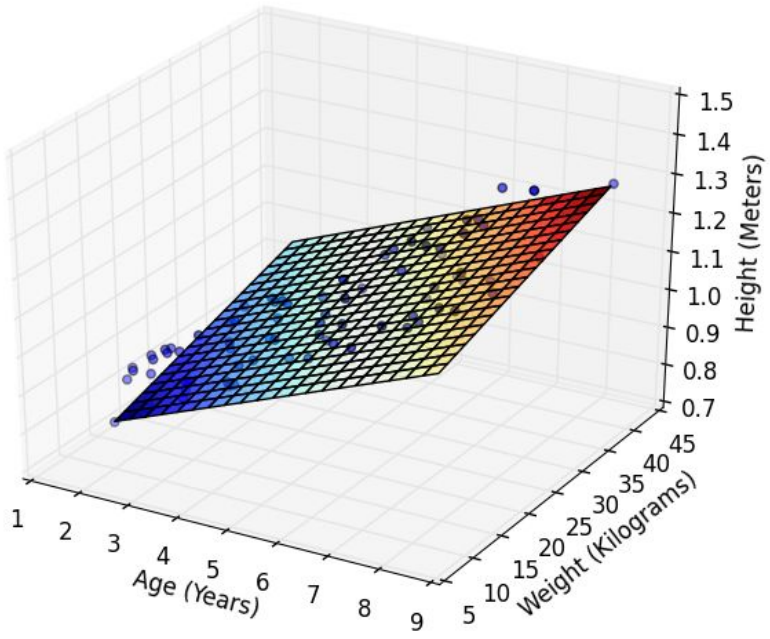
You will run gradient descent with these nine learning rates: α ∈ {0.001, 0.005, 0.01, 0.05, 0.1, 0.5, 1, 5, 10}, **exactly 100 iterations per α-value**, plus a tenth rate and number of iterations of your choice. To pick the tenth rate and loop count, observe how α affects the rate of convergence for the nine rates listed, then pick a rate and loop count you believe will perform well. Briefly explain your choice in your README.

The program should generate an output file containing ten lines, one for each (α, num_iters) hyperparameter pair. Each line contains: `α,num_iters,bias,b_age,b_weight`, expressed to your choice of decimal places (see example `output2.csv` in your starter code). Each line of this file defines the regression model your gradient descent method computed on the given data and hyperparameters.

We will execute your code as follows:

```
$ python problem2.py input2.csv output2.csv
```

Optional. Visualize the result of each linear regression model in three-dimensional space. You can plot each feature on the xy-plane, and plot the regression equation as a plane in xyz-space. Ex:



III. K-Means Clustering

We will segment and group the pixels of trees.png according to their RGB values with k-means. You can read about image segmentation here: https://en.wikipedia.org/wiki/Image_segmentation.

Experiment with the k-means algorithm. Choose 3 representative k values and use sklearn.cluster.KMeans to process the image file. Thoroughly record (screenshots, short annotations) and explain your results in your README.

Save your work as problem3.py.

IV. Before You Submit

- **Make sure** your code executes. In particular, make sure you name your files correctly.
- **Make sure** your program does not print anything extraneous to the screen.
- **Make sure** your written portion is in .pdf (ideal) or .txt format, and your README is .txt or .pdf.