Machine Learning Homework 2

Logistic Regression

Due Date: 16/02/2020

Dataset. In this programming homework, you will use student exams data set. It is a collection of exam scores and a binary value indicating whether student has passed the course (exams.csv).

Your homework is divided into:

10% - loading data and visualization

10% - sigmoid function

10% - cost function

30% - gradient descent implementation from scratch

20% - testing

20% - Solve same problem by Logistic Regression with regularization using library and test.

You are free to use any programming language, although Python is highly recommended. Also, consider using Jupyter Notebook for your own convenience.

Description.

Loading data. For this task you need to read exams.csv file. It contains three columns. First two columns (exam_1, exam_2) are exam scores of a student and the third value indicates whether the student has passed the course or not (1 or 0). (Using pandas library is recommended)

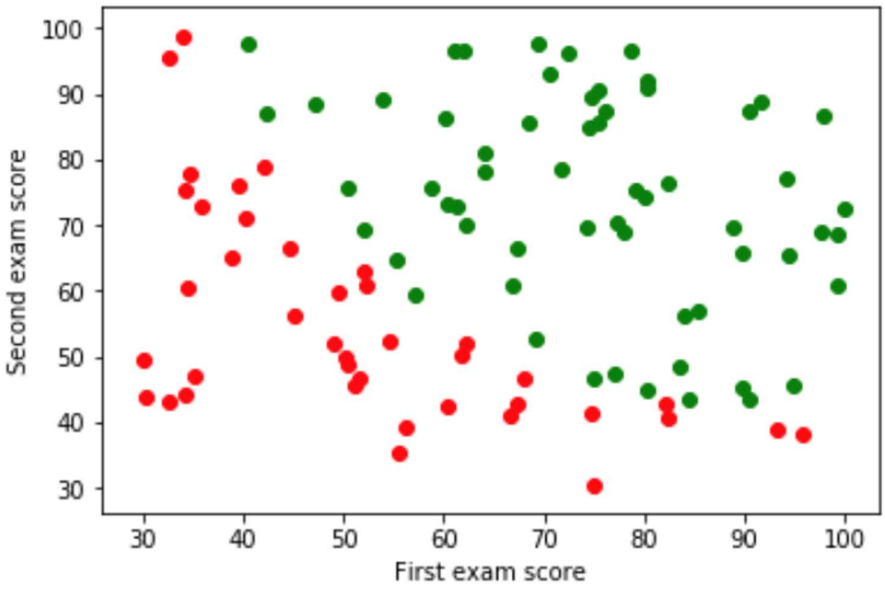
It is recommended to normalize the data at this stage. You will see that gradient descent algorithm performs much better when data is normalized. Use min max normalization.

(**https://en.wikipedia.org/wiki/Feature_scaling#Rescaling_(min-max_normalization)**

(**https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html**)

Visualization part 1.

Plot a graph of first exam score vs second exam score. Admitted student points should be green and failed student points should be red. It should look similar to this one.



Sigmoid function. Implement a function which returns sigmoid of a value given the value.

(https://en.wikipedia.org/wiki/Sigmoid_function)

Cost function. Implement the following cost function.

$$J(w) = -\frac{1}{n}\left[\sum_{i=1}^{n} y^{(i)} log h_w\left(x^{(i)}\right) + \left(1 - y^{(i)}\right) \log\left(1 - h_w\left(x^{(i)}\right)\right)\right]$$

Gradient descent implementation from scratch. Implement a code which finds best fit parameters for logistic regression using gradient

descent from scratch. You should be able to change training step and number of iterations through the variables (or input to the function). In addition to that, you should save cost at each iteration and plot it in the next sub-task.

$$w_j := w_j - \alpha \frac{1}{n} \sum_{i=1}^{n} \left(h_w\left(x^{(i)}\right) - y^{(i)} \right) x_j^{(i)}$$

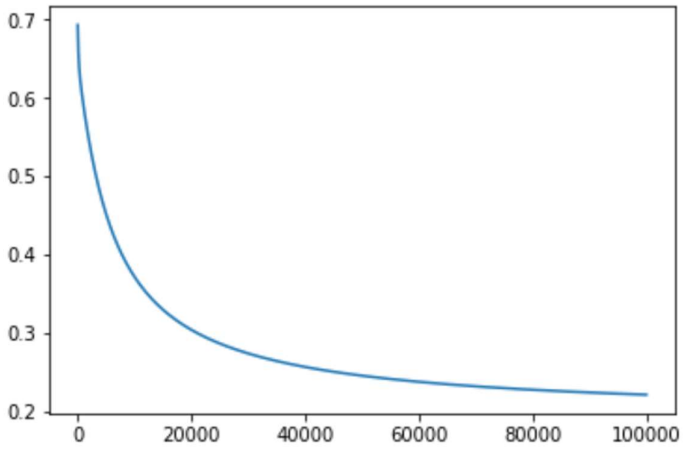
simultaneously update all w_j

$$h_w(x) = \frac{1}{1 + e^{-w^T x}}$$

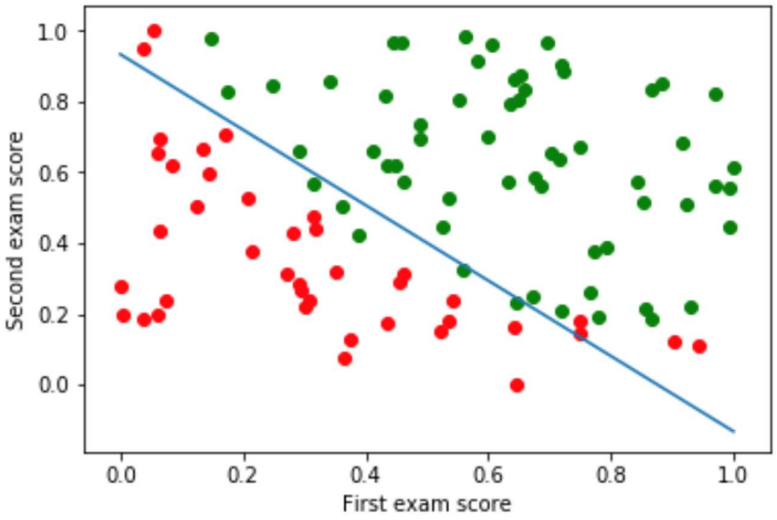
Visualization part 2.

You should plot:

1. Array of costs at each iteration (Cost vs Iteration). It should look similar to this: (cost decreases with each iteration)



2. Plot points of first exam score vs second exam score. Admitted student points should be green and failed student points should be red. (same as Visualization part 1). And plot the decision boundary using the parameters found by gradient descent on the same graph. It should look similar to this:



Test.

Make predictions of the training data using your trained model, compare predicted labels with actuals label and print the score indicating how well your model performs. You can use **accuracy_score** function from **scikit-learn** library.

After doing that, check if your model makes correct predictions for these data: {55, 70, 1} and {40, 60, 0}.

Please follow the instruction below when you submit your homework assignment to the Blackboard System.

- Submit homework solution in 2 files: 1) python file (.py or ipynb) 2) word file containing your codes and achieved results.
- Write your full name and number of homework assignment in the FILE NAME.
- Don't submit files in Zip file format.

Note. Please be informed that your assignment on coding will be checked through Safe Assign in Blackboard. In case of two same (similar) assignment submissions, both students will be scored as a zero. Furthermore, any student whose solution may arise a question or, will be asked for some explanations as well.