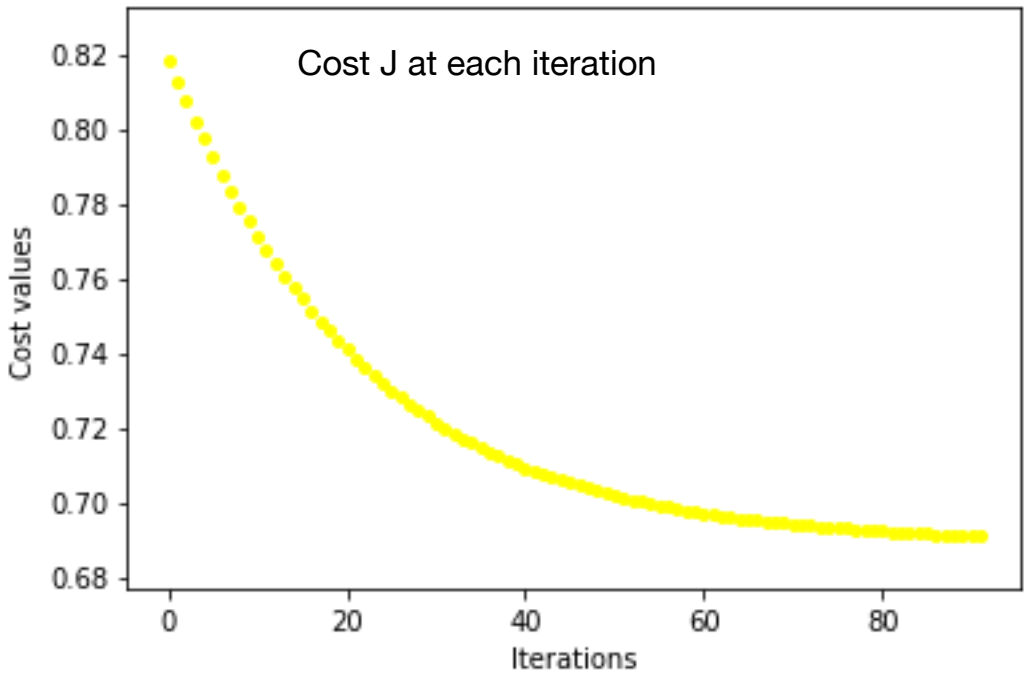
Aaron Yerke, HW 2 for ML 2019
1. (50 points) Implement gradient descent-based logistic regression in Python. Use
ΔJ = 0.00001 as the stopping criterion.
2. (50 points total distributed as below) Apply your code from question 2 to the iris virginica and virsicolor flowers. Specifically, randomly select 99 of these flowers for training your logistic model and use the remaining one flower for testing. You only need to do training once and testing once with your specific choice of the training flowers and testing flowers. That is to say, you don't need to do the leave-one-out cross validation 100 times.
(a) (15 points) After your training, plot the total cost J vs iterations for your 99 training flowers for four scenarios.
(b) (20 points) Predict the flower type of your testing flower for each of the four scenarios.
(c) (15 points) Apply sklearn.linear model.LogisticRegression to your specific choice of training flowers. With the intercept and coefficients produced by sklearn, calculate the total final cost J for your 99 flowers.

#Notes from John:
#Python code for both running logistic regression with gradient descent and analyzing the iris data.
#Some document (docx, pdf, etc.) for part 2 (a,b,c) with minimal descriptions of results.
#
#There's no reason this can't be in one script, but for readability calling the class in a separate analysis script would be easier.

See attached code, or go to https://github.com/palomnyk/machineLearningFall2019 to view code online.
The f"" function that is use was released in python 3.67, so older versions of python may have issues with this.

A) In general, my code takes around 50-100 iterations to reach the stop location (see figure below for example output).



B) My understanding from reading part 1, is that we didn't need to do the leave-one-out. This is a little confusing, so please allow be to correct this if I misunderstood.

C) My cost function indicates that the cost is around negative 1 for the sklearn logistic regression.