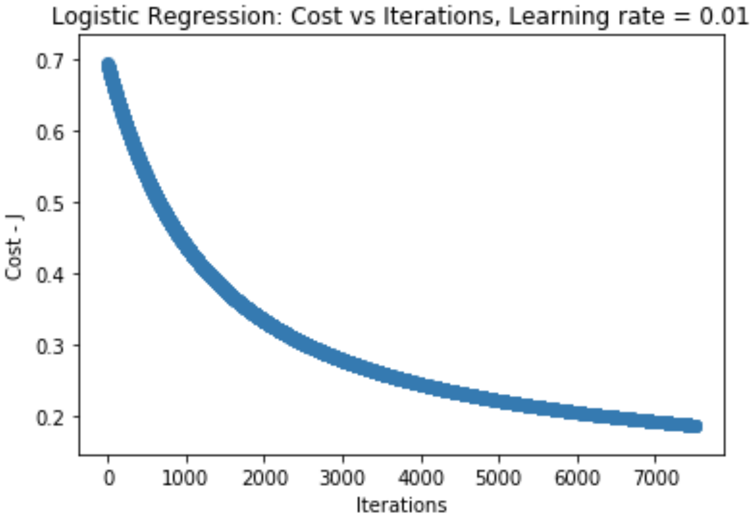
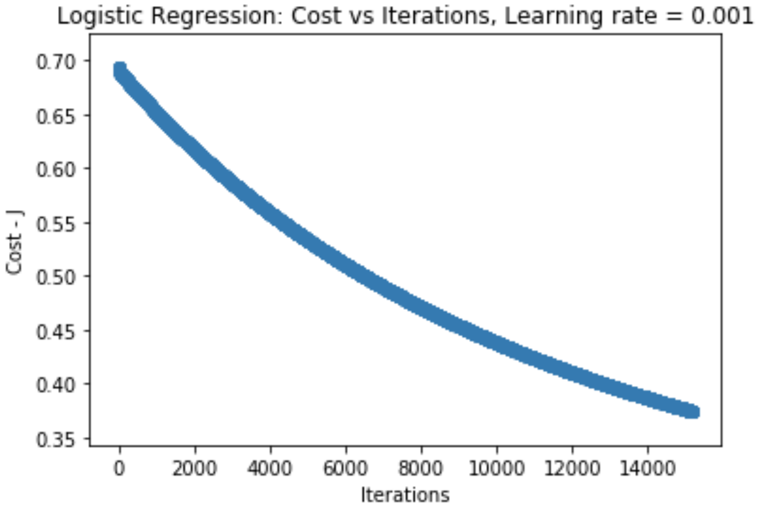
Homework 2 –Logistic regression
Brendan Kearney
November 21, 2019

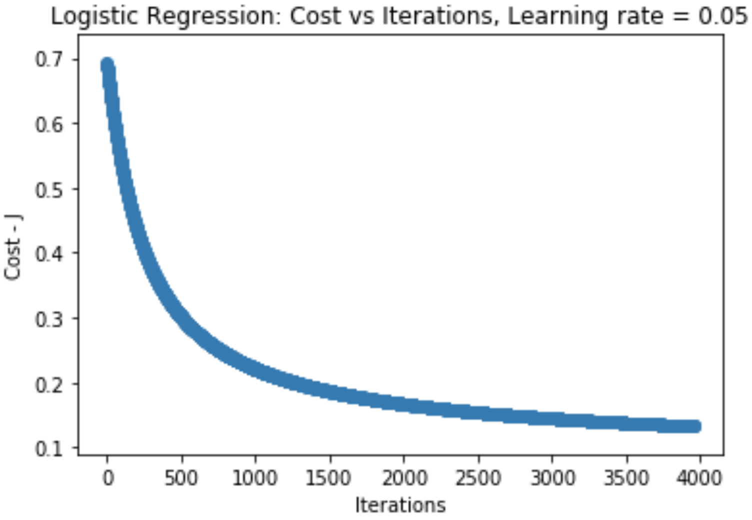
Results and Analysis

my_logistic_reg.py – Logistic regression class
logistic_iris.py – application of class on iris data and plotting/output

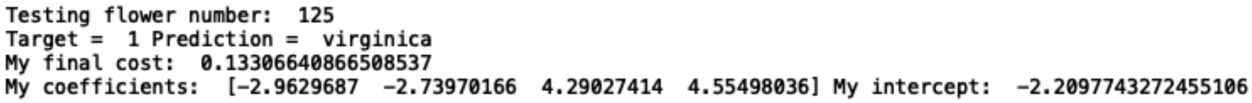
(2a) **Plots of total cost J vs the number of iterations, with different learning rates**. As expected, increasing the learning rate decreases the number of iterations needed to reach our stopping criteria, and therefore lessens the time spent training. However, there is also an upper limit to how high the learning rate can get.







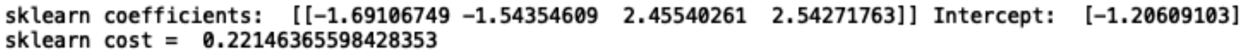
(2b) **Predict the flower type of your testing flower**



```
Testing flower number:  125
Target =  1 Prediction =  virginica
My final cost:  0.13306640866508537
My coefficients:  [-2.9629687  -2.73970166  4.29027414  4.55498036] My intercept:  -2.2097743272455106
```

Flower #125 was randomly selected to be the testing flower. From there, the other 99 samples were trained using the logistic regression class so that the predict() method could return a prediction between 0 and 1. A deciding threshold (in this case 0.5) was used to sort the prediction into either versicolor or virginica.

(2c) **sklearn**



```
sklearn coefficients:  [[-1.69106749 -1.54354609  2.45540261  2.54271763]] Intercept:  [-1.20609103]
sklearn cost =  0.22146365598428353
```

The coefficients and intercept were retrieved from sklearn’s logistic regression on the same training set as in (2b). After that, I used a cost function identical to the one used in the LogReg class to calculate the total final cost J. Surprisingly, the final cost is consistently lower in my gradient descent when compared to sklearn.