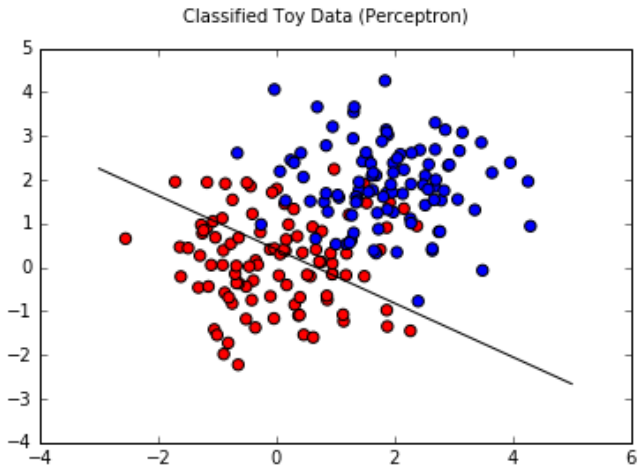
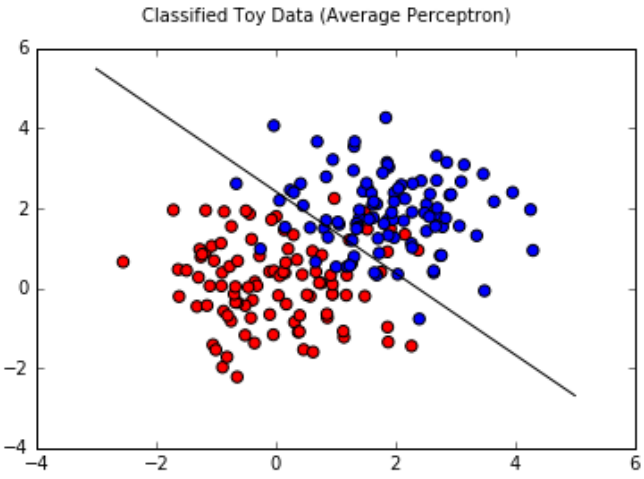
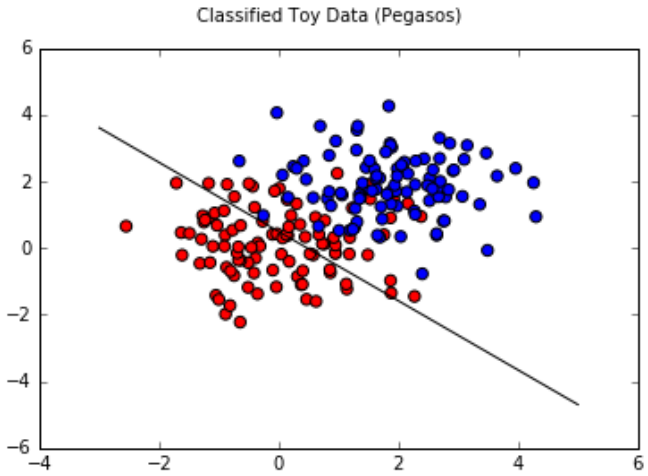
Oluwafemi Oladipupo
6.036 Project 1

7) Why do these algorithms provide different decision boundaries?
The algorithms converge to different boundaries because they calculate their decision parameters via different method. While the average perceptron algorithm makes use of the same per step update as the normal perceptron algorithm, the final parameters are calculated from the average of the parameters after each update. This means that earlier parameters have more weight in the result than with a normal perceptron algorithm. The pegasos algorithm has a completely different per step function, and the change per update scales inversely with the number of updates already performed. This causes the decision boundary difference between the perceptron and pegasos algorithms.







9) Training and validation accuracies of each algorithm with T = 10 and λ = 0.01
Training accuracy for perceptron: 0.9593
Validation accuracy for perceptron: 0.8240
Training accuracy for average perceptron: 0.9760
Validation accuracy for average perceptron: 0.8420
Training accuracy for Pegasos: 0.9227
Validation accuracy for Pegasos: 0.8080

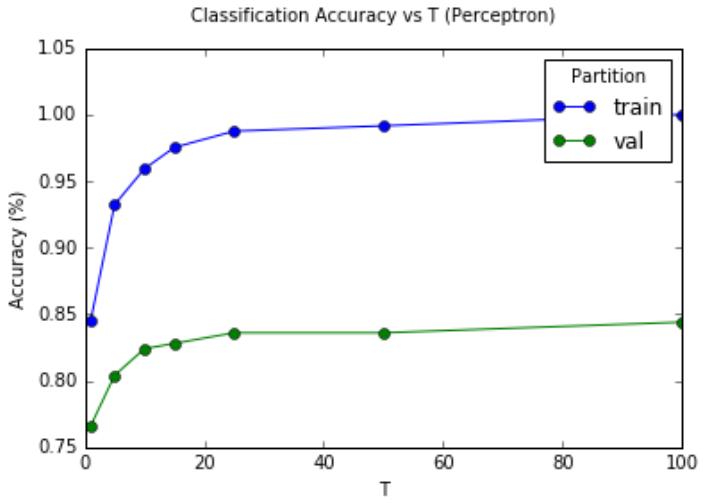
10a) Do the training and validation accuracies behave similarly as a function of λ and T? Why?
Training and validation accuracies do behave similarly. This is because a training set is ideally representative of the greater data population. With more samples the similarity should increase.

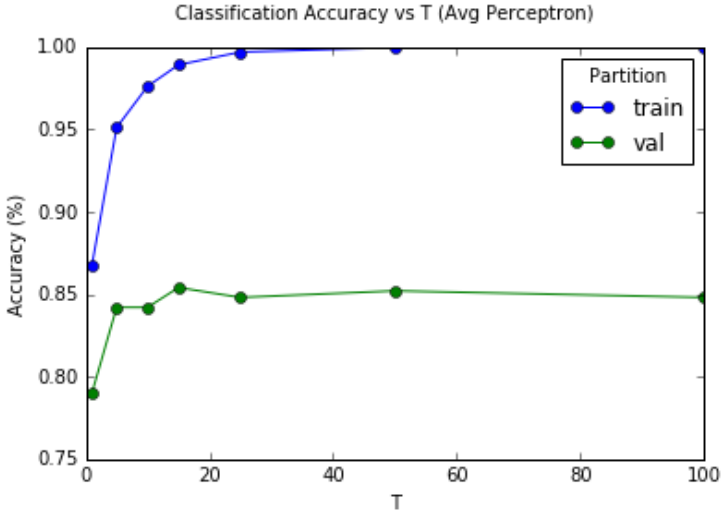
10b) Which algorithm performed the best of the three?
Average Perceptron performed the best over the tested points. Perceptron algorithms in general performed better on these data sets.

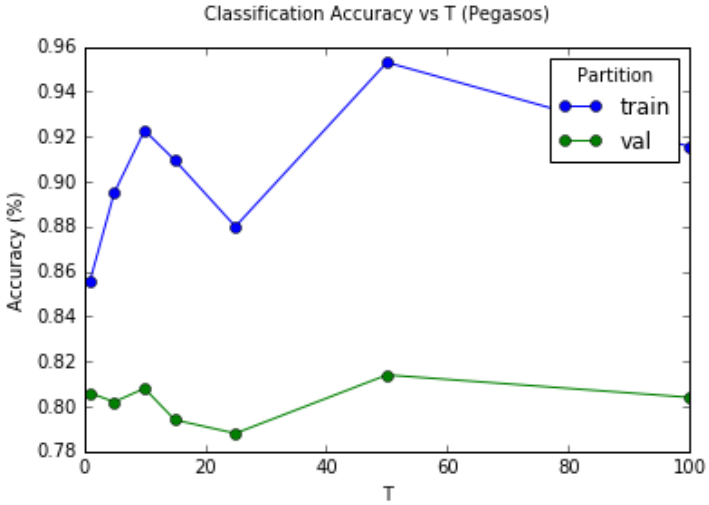
10c) What are the optimal values of T for your perceptron? For average perceptron? What are the optimal values of T and λ for your Pegasos algorithm?
Perceptron:
T = 100

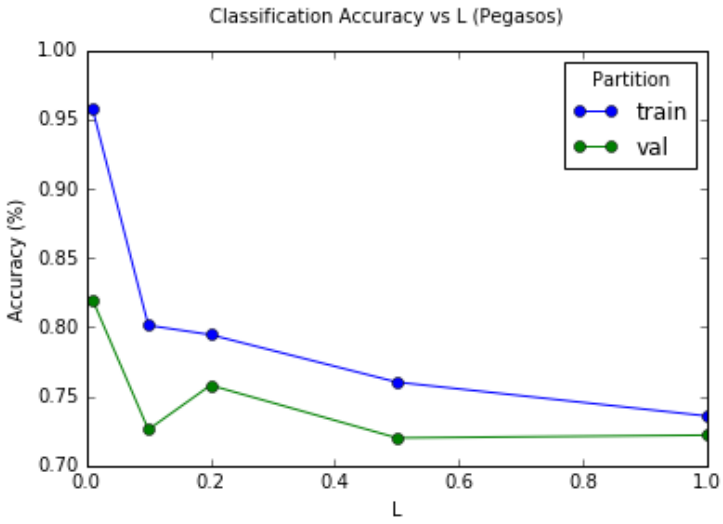
Avg Perceptron:
T = 15

Pegasos:
T=95
L = 0.01


Classification Accuracy vs T (Perceptron)
Partition
train
val
Accuracy (%)
T
1.05
1.00
0.95
0.90
0.85
0.80
0.75
0
20
40
60
80
100


Classification Accuracy vs T (Avg Perceptron)
Partition
train
val
Accuracy (%)
T
1.00
0.95
0.90
0.85
0.80
0.75
0
20
40
60
80
100


Classification Accuracy vs T (Pegasos)
Partition
train
val
Accuracy (%)
T
0.96
0.94
0.92
0.90
0.88
0.86
0.84
0.82
0.80
0.78
0
20
40
60
80
100



11a) Applying Avg Perceptron to Data with T=15
Training accuracy for average perceptron: 0.9890
Validation accuracy for average perceptron: 0.8540

11b) Applying Avg Perceptron to Data with T=15
['delicious', 'amazing', 'wonderful', 'glad', 'pleased', 'plan', 'canned', 'helped', 'excellent', 'run']

12) Improving Avg Perceptron @ T=15
No improvement values:

Training accuracy: 0.9890
Validation accuracy: 0.8540
Test accuracy: 0.7160

The first improvement was to the dictionary, cutting the stop words as outlined in the instructions. The expectation is that by eliminating words that do not have a positive or negative meaning, the accuracy of the analysis will be improved. This resulted in a decrease in validation accuracy, from 0.8540 to 0.8300, but an increase in accuracy when tested against the test data. This is an expected improvement.
Training accuracy: 0.9942
Validation accuracy: 0.8300
Test accuracy: 0.7200

The second improvement was to the also to the dictionary. The idea was to cut words that did not strongly indicate a positive or negative review. This was done by generating an initial decision parameter, then using this parameter to extract the top 25% of dictionary words by significance, measured by the theta dimension magnitude. This resulted in a decrease in validation accuracy and testing accuracy, which was against expectations.

Training accuracy: 0.9645
Validation accuracy: 0.7560
Test accuracy: 0.6480

Additionally, the second improvement was ran with a larger and smaller threshold of extracted parameters, at 70% and 20%.
70% Training accuracy: 1.0
70% Validation accuracy: 0.8180
70% Test accuracy: 0.7080

20% Training accuracy: 0.9437
20% Validation accuracy: 0.7350
20% Test accuracy: 0.632