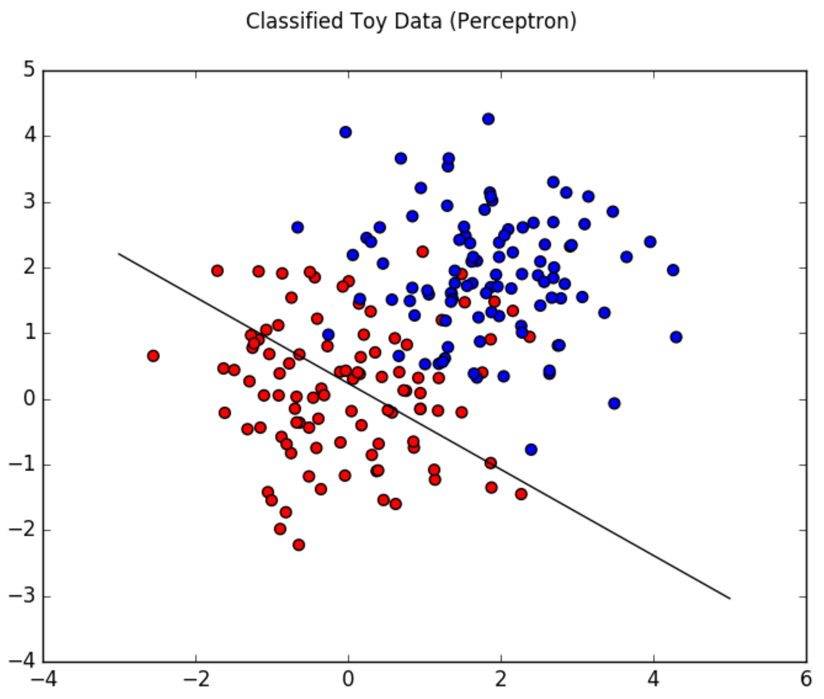
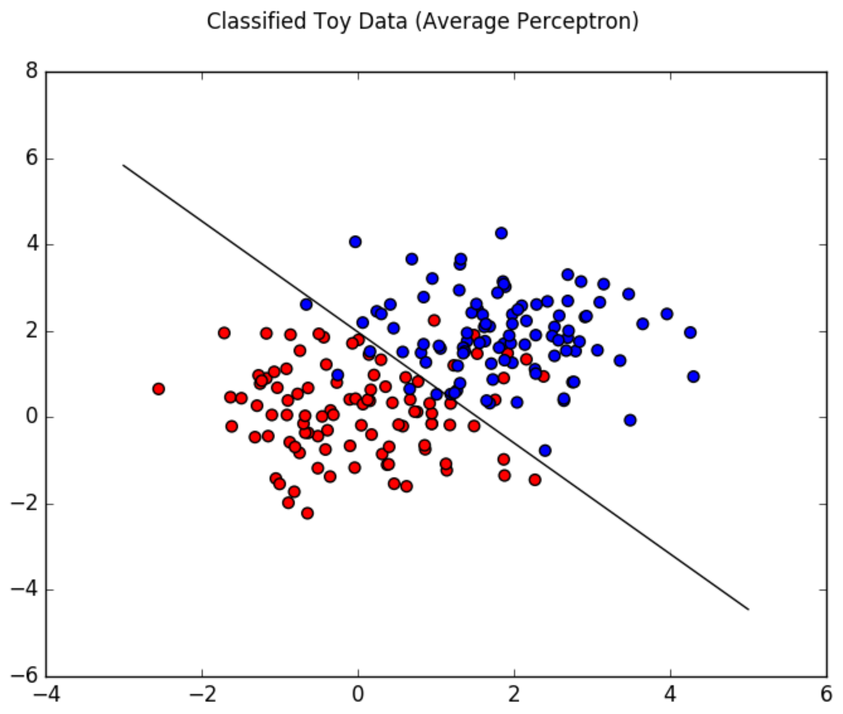
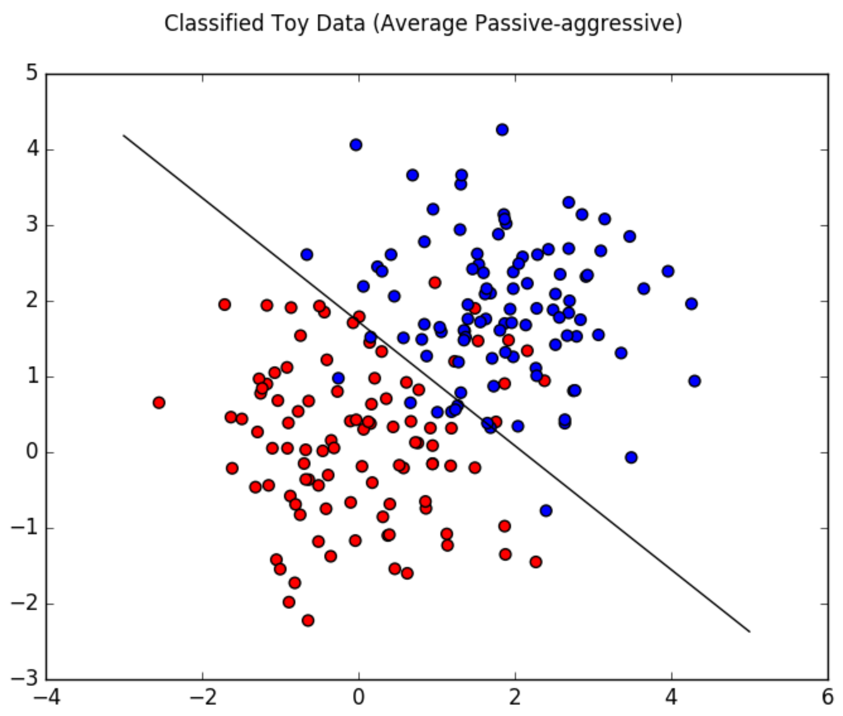
1. Implementing Classifiers

Part 7







The perceptron misclassifies more points than the other two algorithms do. This is because it uses the zero-one loss function. It doesn't take into account how big the error might be while updating θ; it treats all errors equally. This is why the decision boundary doesn't enter the area where red and blue points are mixed and therefore it doesn't correctly classify as many points as the other algorithms do.

The average perceptron performs better than the regular perceptron. This is because in the regular perceptron algorithm it is possible for θ to update in the wrong direction. Average perceptron prevents this by taking an average of all the previously obtained θs. This does in fact result in better overall performance as seen in the plot.

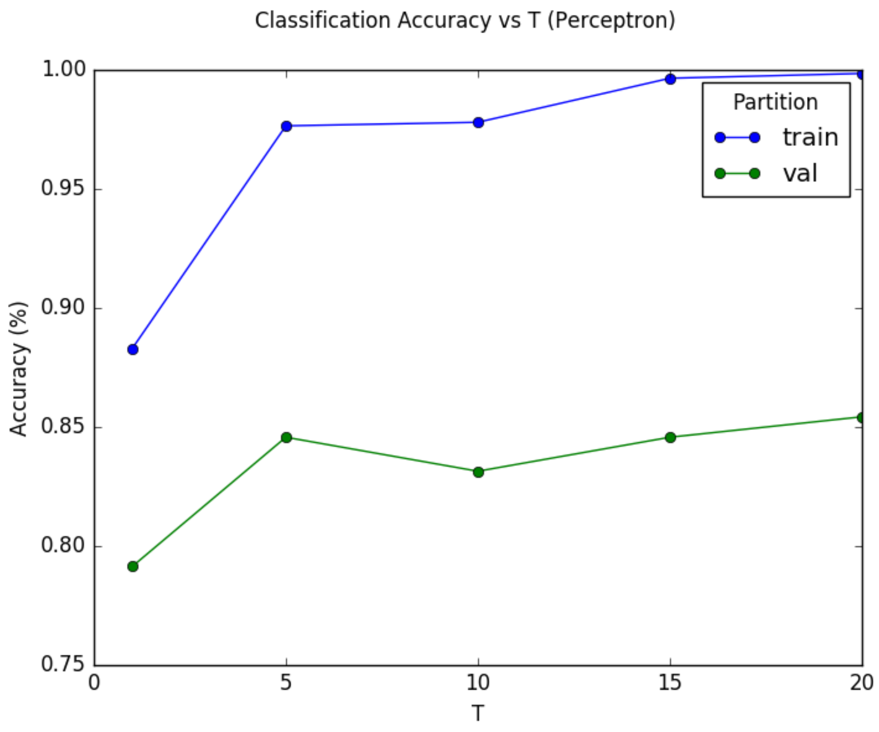
The passive-aggressive performs the best amongst all three algorithms. Though its performance isn't significantly better than that of average perceptron. The reason passive-aggressive algorithm performs the best is because it find the optimal decision boundary by using the hinge loss function instead of zero-one loss. What this algorithm does is minimize the hinge loss and keep the parameters close to what they were before. The parameter λ determines how passive or aggressive the algorithm is. Smaller values of λ permit larger deviations so as to minimize the loss. The improvement in performance can be seen from the plot.

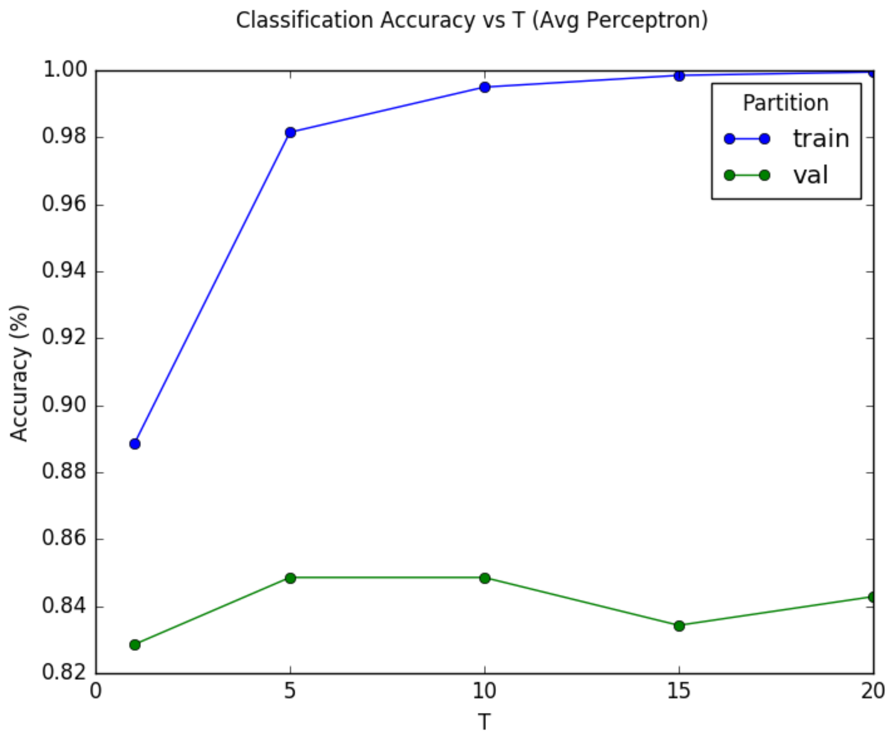
Part 9

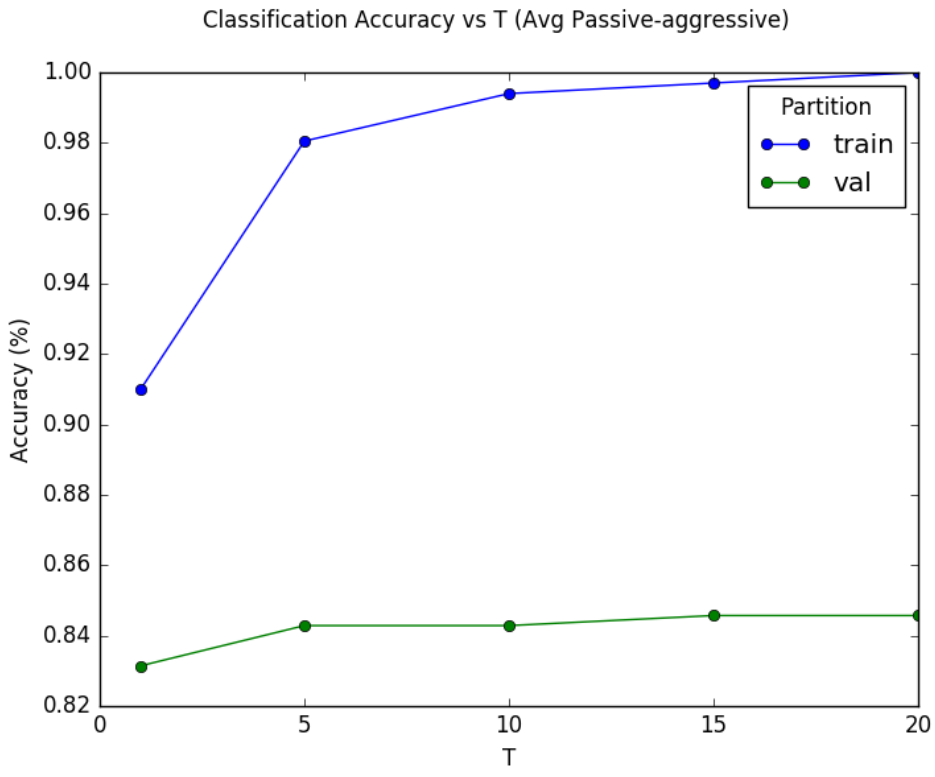
	Perceptron	Avg Perceptron	Avg PA
Training Accuracy	0.9765	0.9815	0.9805
Validation Accuracy	0.8457	0.8486	0.8429

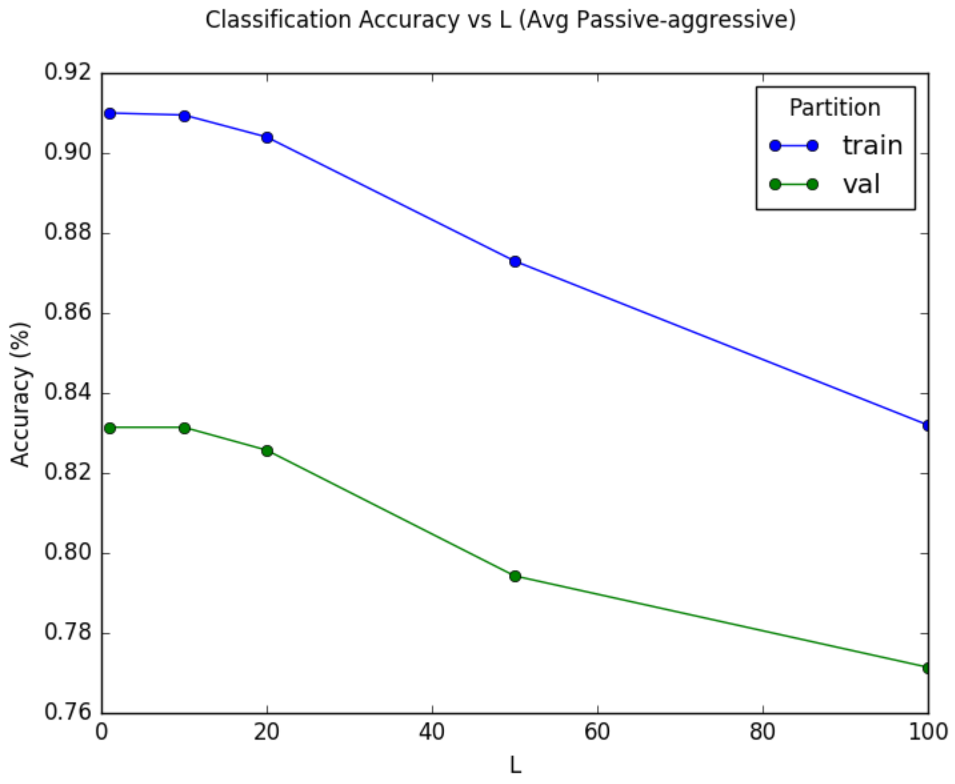
2. The Automatic Review Analyzer

Part 10






Classification Accuracy vs T (Avg Passive-aggressive)
Accuracy (%)
T
Partition
train
val
1.00
0.98
0.96
0.94
0.92
0.90
0.88
0.86
0.84
0.82
0
5
10
15
20


Classification Accuracy vs L (Avg Passive-aggressive)
Accuracy (%)
L
Partition
train
val
0.92
0.90
0.88
0.86
0.84
0.82
0.80
0.78
0.76
0
20
40
60
80
100

a. Yes, they do behave similarly. Because both show the performance of the same classifier, just on different data sets. The classifier was originally trained on the training data set and then for validation of its performance, it gets applied to a different data set. So, it is expected to behave similarly.

b. Avg passive aggressive algorithm performed the best.

c. Optimal values: L = 1, T=20

Part 11

a. Train accuracy: 1.0, Test accuracy: 0.82857

b. Top 10 most explanatory word features: ‘!’, ‘you’, ‘great’, ‘favorite’, ‘delicious’, ‘can’, ‘and’, ‘have’, ‘excellent’, ‘very’

3. New features and the Challenge

Part 12

I tried one additional improvement: Length of the text.

I added a new feature (length of review) by adding a new column to the feature matrix. I thought it would be worth checking if there is a correlation between the value of a review (as positive or negative) and the length of it. For example, maybe positive reviews tend to be longer because people who have liked a product have a lot to say about it, whereas negative reviews tend to be shorter because people who have disliked a product would say things like “Don’t buy it”, “Very disappointing”, “Too small. Came in wrong color” etc. So I thought maybe there is actually a pattern that negative reviews tend to be shorter than positive reviews. I used the code below:

```
def extract_additional_features(reviews):
    new_column = np.zeros(len(reviews), 1)
    for i in range(len(reviews)):
        new_column[i] = len(reviews[i])
    return new_column
```

After constructing the final feature representation I ran the code in *Section 12* of *main.py* to evaluate the accuracy of the algorithm and compare it with the previous accuracy I got without the additional feature. The resulting accuracy was around the same. No improvement was observed. Therefore, I concluded that it wasn’t a useful feature addition.