CPT_S 570

Machine Learning, Fall 2020

Homework # 1

Tayyab Munir

11716089

Question 1 – Analytical Part

1. Answer the following questions with a yes or no along with proper justification.

   a. Is the decision boundary of voted perceptron linear?

   Answer: The decision boundary of voted perceptron is nonlinear. The reason is that it relies on survival times for voted selection. One hyperplane does not form the decision boundary. A very simple example would be say w1=(2,0) and w1=(0,2) are the weights with survival times 1. Then both get equal voting rights and the decision boundary would be such that only examples in first quadrant are classified as positive. This boundary is non-linear.

   b. Is the decision boundary of averaged perceptron linear?

   Answer: The decision boundary for average perceptron is linear and it is the hyperplane perpendicular to the averaged weight vector. Since boundary will always be a D-1 dimensional hyperplane in D-dimensional space the decision boundary is linear. For instance say w1=(2,0) and w1=(0,2) are the weights with survival times 1, then Wavg=(1,1) and decision boundary would be a straight line perpendicular to vector (1,1).

2. In the class, we saw the Passive-Aggressive (PA) update that tries to achieve a margin equal to *one* after each update. Derive the PA weight update for achieving margin *M*.

$$w_{t+1} = w_t + \tau\, y_t x_t \text{ ----(1)}$$

$$Margin = y_t w_t + 1\, x_t \text{ ----(2)}$$

$$M = y_t(w_t + \tau y_t x_t)x_t \text{ ----(3)}$$

$$\tau = \frac{M - y_t(w_t x_t)}{||x_t||^2} \text{ ----(4)}$$

3. Consider the following setting. You are provided with n training examples: $(x1; y1; h1); (x2; y2; h2); \cdots; (xn; yn; hn)$, where *xi* is the input example, *yi* is the class label (+1 or -1), and $hi > 0$ is the importance weight of the example. The teacher gave you some additional information by specifying the importance of each training example.

a. How will you modify the perceptron algorithm to be able to leverage this extra information? Please justify your answer.

Answer: We can modify the perceptron algorithm by doing aggressive update when there is a mistake on high importance training example and less aggressive update when mistake is on low importance training example. The update in such a formulation would be proportional to the importance of training example.
Modified Update Step: $w = w + h_i(y_i * x_i)$

b. How can you solve this learning problem using the standard perceptron algorithm? Please justify your answer. I am looking for a reduction-based solution.

Answer: In this case we do not want to modify the perceptron algorithm. The solution could be sorting the examples in order of their importance. The perceptron algorithm inherently gives more importance to examples that it encounters first during the training. This would result in automatic consideration of the importance of training examples.

4. Consider the following setting. You are provided with n training examples: $(x1; y1); (x2; y2); \cdots ; (xn; yn)$, where xi is the input example, and yi is the class label (+1 or -1). However, the training data is highly imbalanced (say 90% of the examples are negative and 10% of the examples are positive) and we care more about the accuracy of positive examples.

a. How will you modify the perceptron algorithm to solve this learning problem? Please justify your answer.

Answer: We can modify the perceptron algorithm for unbalanced data set by assigning more importance (in terms of weights) to positive examples and less importance (In terms of weights) to negative examples. Let hp be the importance for each positive example, and hn be the importance of negative example, then;
Modified Positive Update Step: $w = w + h_p(y_i * x_i)$
Modified Negative Update Step: $w = w + h_n(y_i * x_i)$

b. How can you solve this learning problem using the standard perceptron algorithm? Please justify your answer. I'm looking for a reduction-based solution.

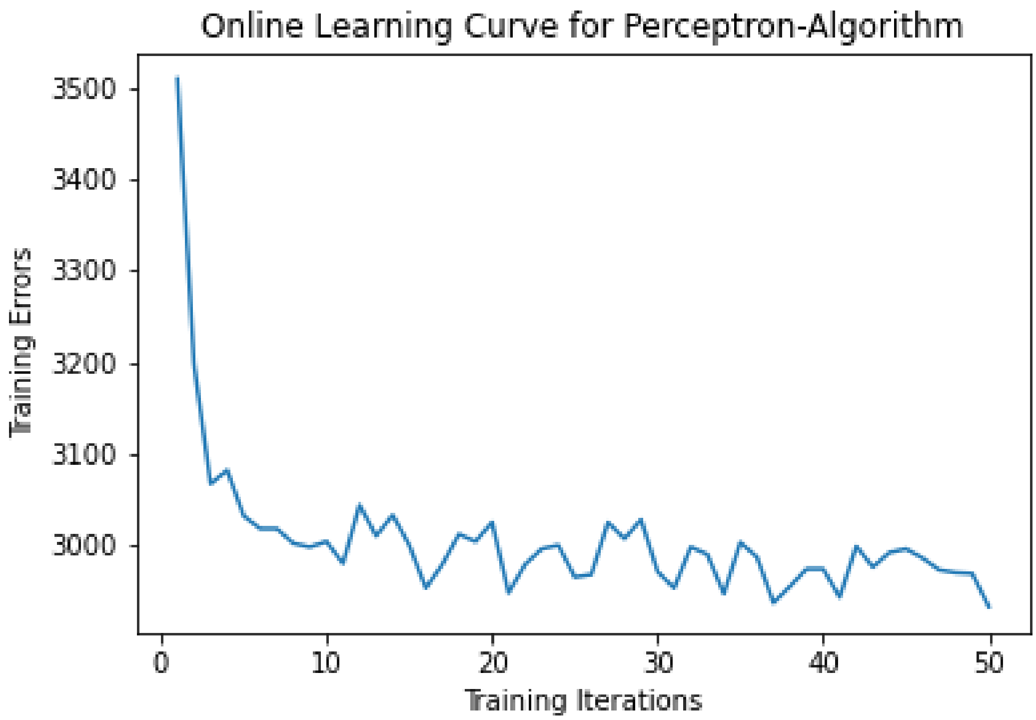
Answer: If we do not want to modify the algorithm, best approach would be resampling to get balanced data. One can make a balanced set by choosing 10% negative examples and all positive examples. Weights can be trained by including replacing negative examples with 10% new negative training examples and keeping positive examples intact. This can be done until all negative data has been considered.

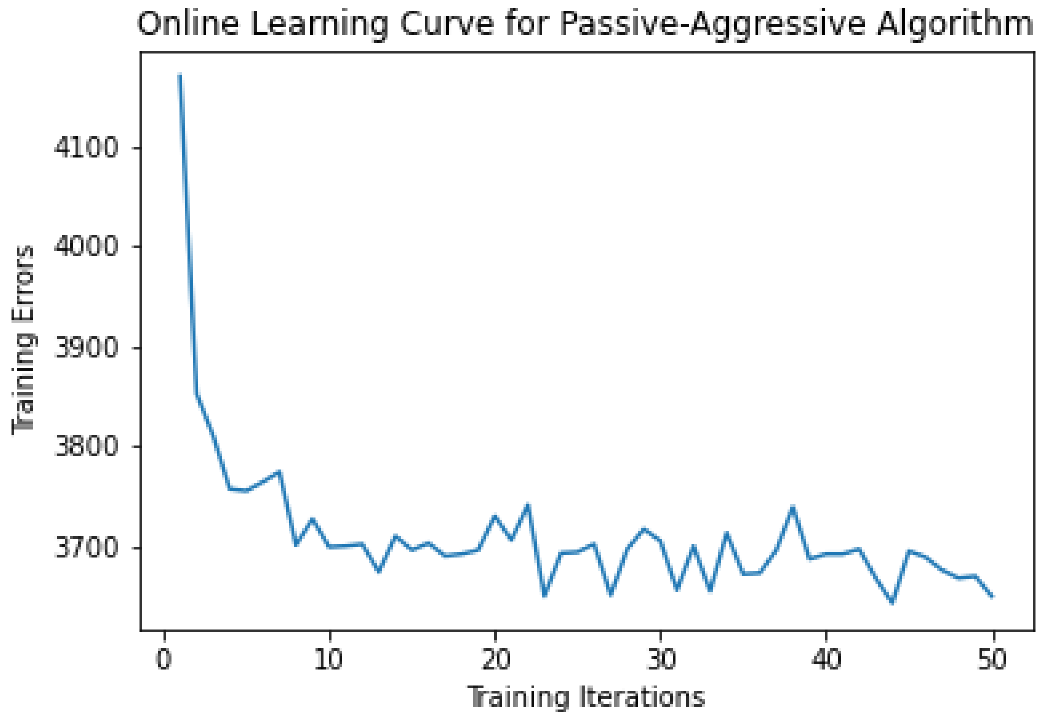
Question 2 – Programming and Empirical Analysis Part

```
Out[18]:
array([3510., 3198., 3067., 3082., 3032., 3018., 3018., 3002., 2998.,
       3004., 2980., 3044., 3010., 3033., 3000., 2953., 2979., 3012.,
       3004., 3025., 2948., 2979., 2996., 3000., 2965., 2968., 3025.,
       3007., 3028., 2971., 2953., 2998., 2990., 2947., 3003., 2987.,
       2937., 2955., 2974., 2974., 2943., 2999., 2976., 2992., 2996.,
       2986., 2973., 2970., 2969., 2933.])
```

5.1 Binary Classification Learn a binary classifier to classify *even labels* (0, 2, 4, 6, 8) and *odd labels* (1, 3, 5, 7, 9).

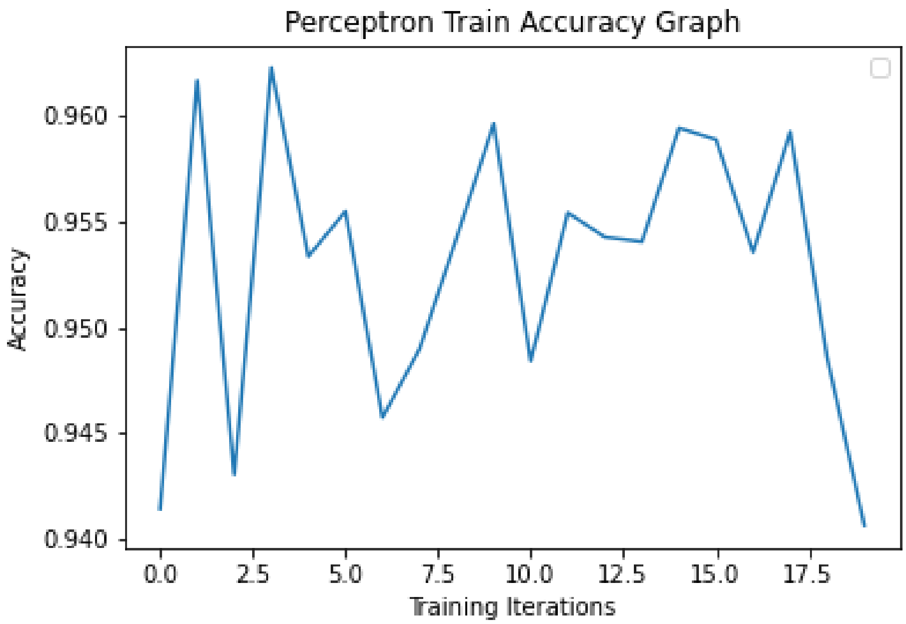
a. Compute the online learning curve for both Perceptron and PA algorithm by plotting the number of training iterations (1 to 50) on the x-axis and the number of mistakes on the y-axis. Compare the two curves and list your observations.

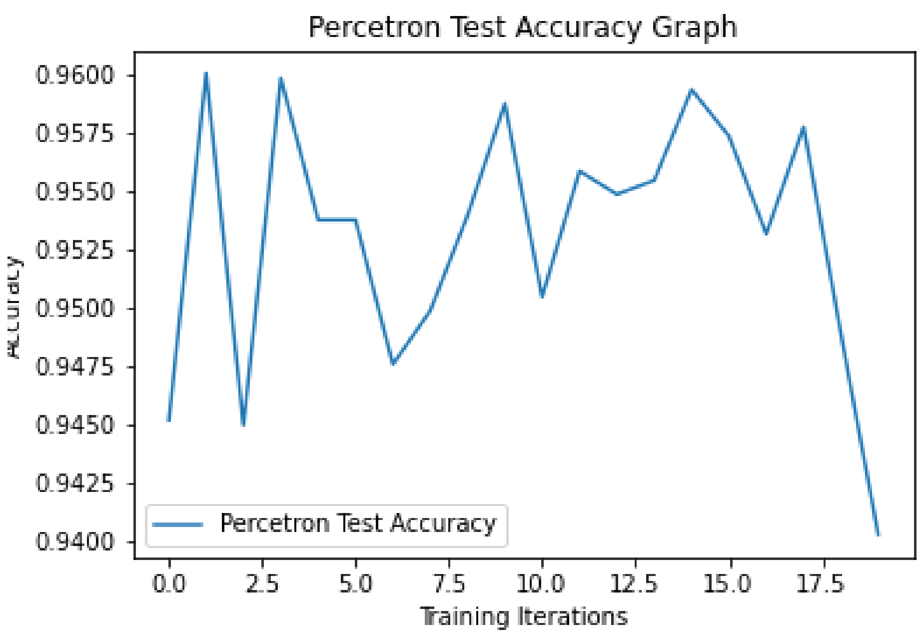


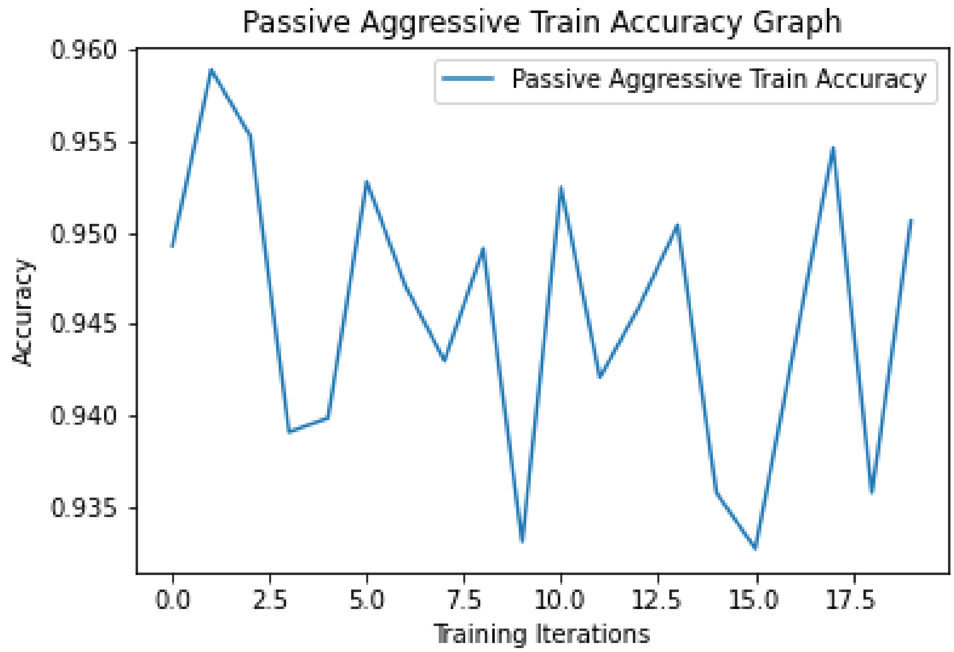


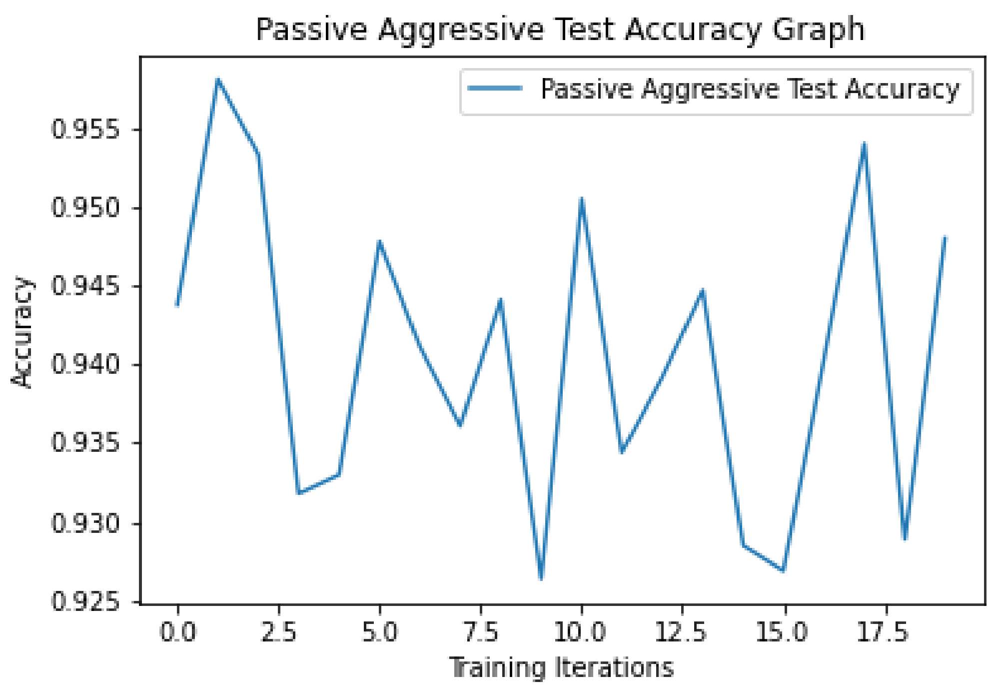
It can be deduced from the learning curves that Perceptron and Passive Aggressive Algorithms Learning performance increases with respect to the number of training passes and converges in the end. Moreover, learning performance of Perceptron is better than the Passive Aggressive Algorithm in terms of training data's error.

b. Compute the accuracy of both Perceptron and PA algorithm on the training data and testing data for 20 training iterations. So you will have two accuracy curves for Perceptron and another two accuracy curves for PA algorithm. Compare the four curves and list your observations.



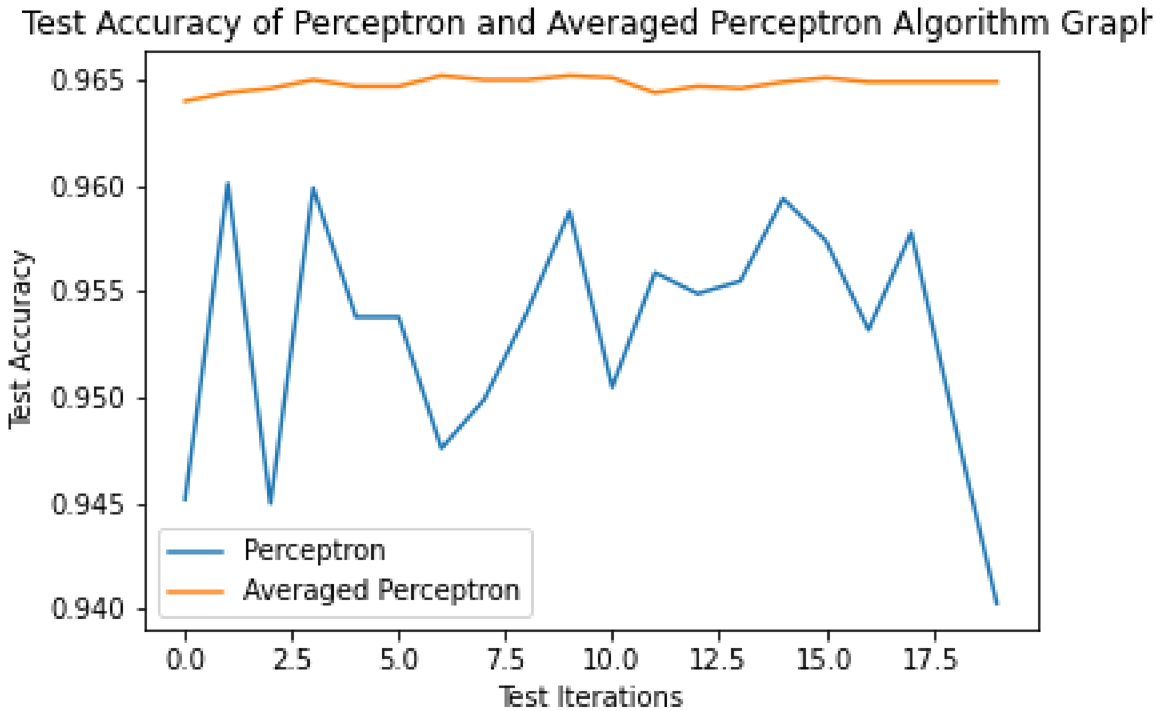






It can be observed from the 4 graphs that are obtained above that Perceptron and Passive Aggressive Algorithm's accuracy in terms of performance are similar on Tests and training data sets.

c. Repeat experiment (b) with averaged perceptron. Compare the test accuracies of plain perceptron and averaged perceptron. What did you observe?



It can be observed from the graph above that Accuracy of Averaged Perceptron is better and more smooth than the Plain Perceptron.

d. Compute the general learning curve (vary the number of training examples starting from 100 in the increments of 100) for 20 training iterations. Plot the number of training examples on x-axis and the testing accuracy on the y-axis. List your observations from this curve.

For Range: r = range(3000,63000,3000)

3000

6000

9000

12000

15000

18000

21000

24000

27000

30000

33000

36000

39000

42000

45000

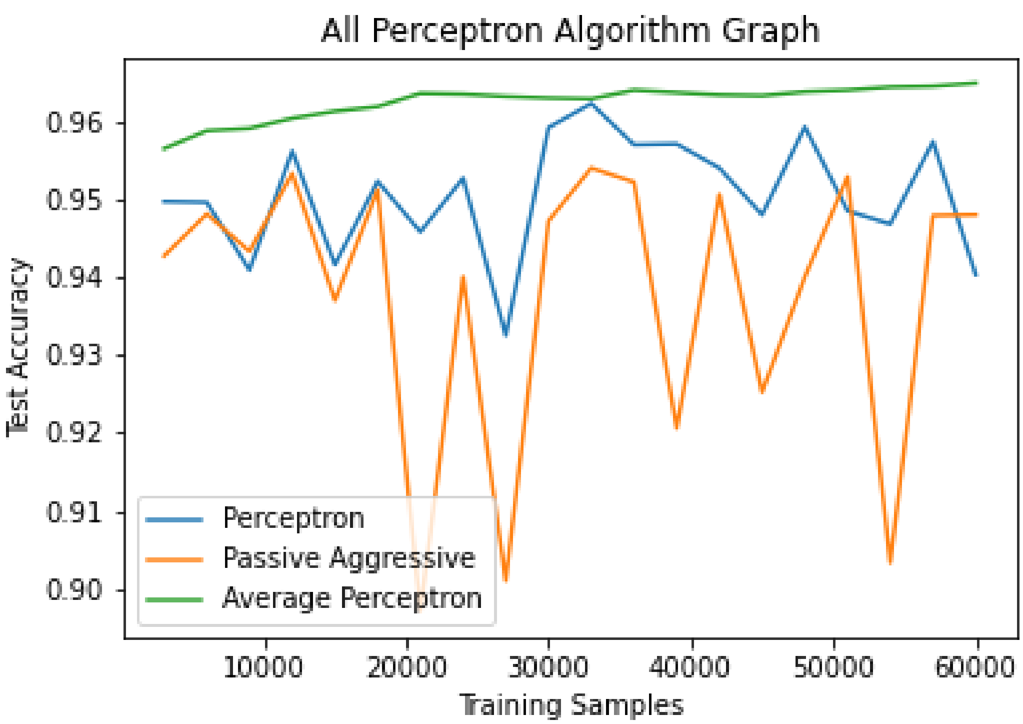
48000

51000

54000

57000

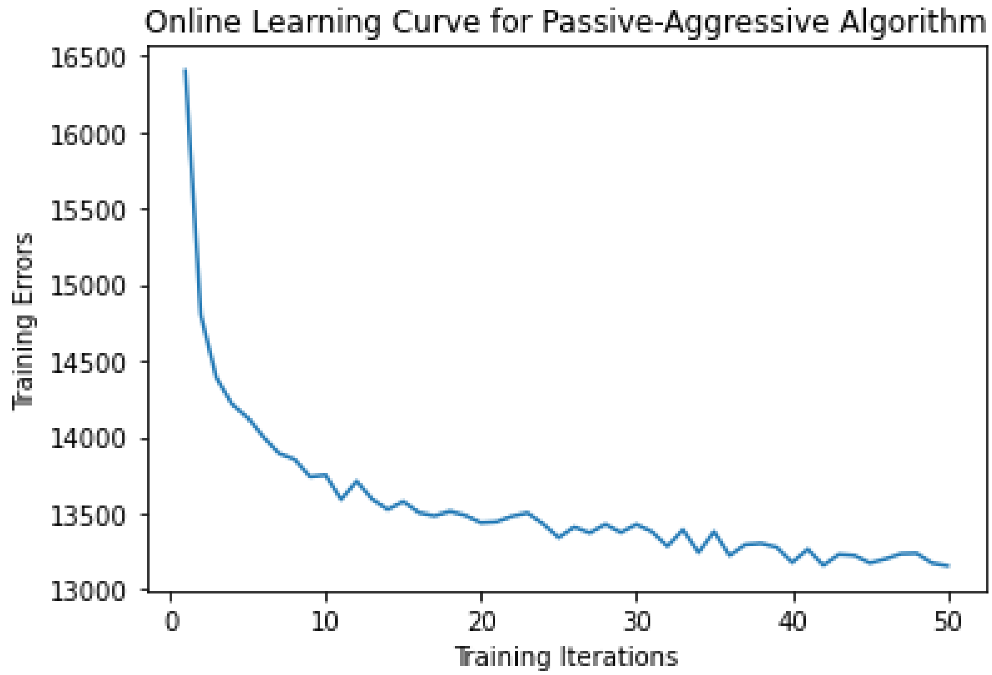
60000

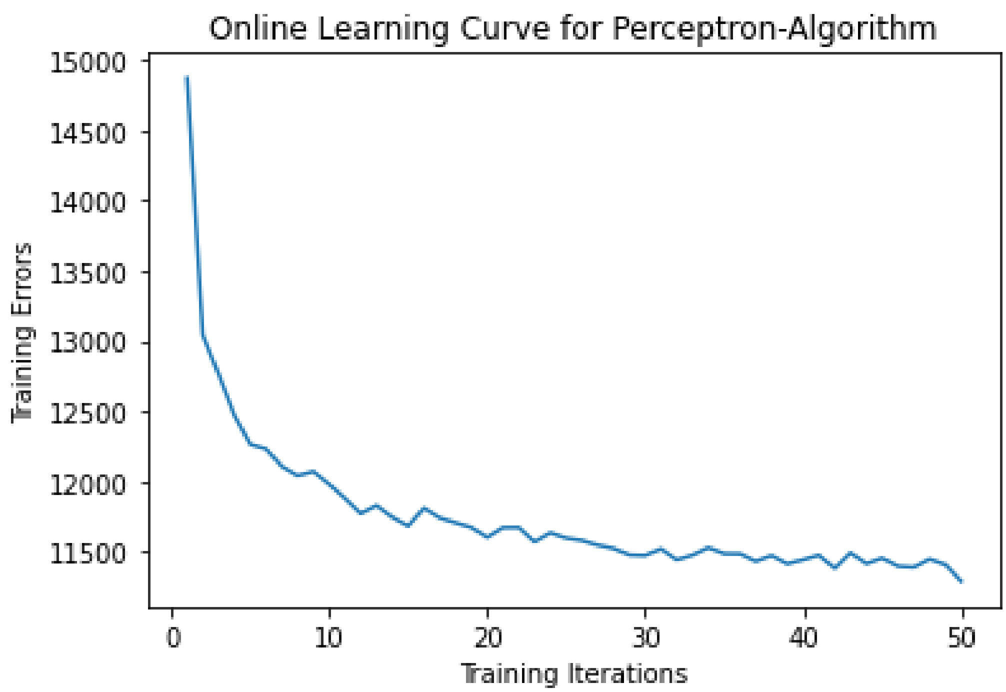


It can be deduced from the above graph that Averaged Perceptron Algorithm's Accuracy performance is way better than Perceptron and Passive Aggressive Algorithms in terms of generalization and with increasing numbers of training examples.

5.2 Multi-Class Classification Learn a multi-class classifier to map images to the corresponding fashion label.

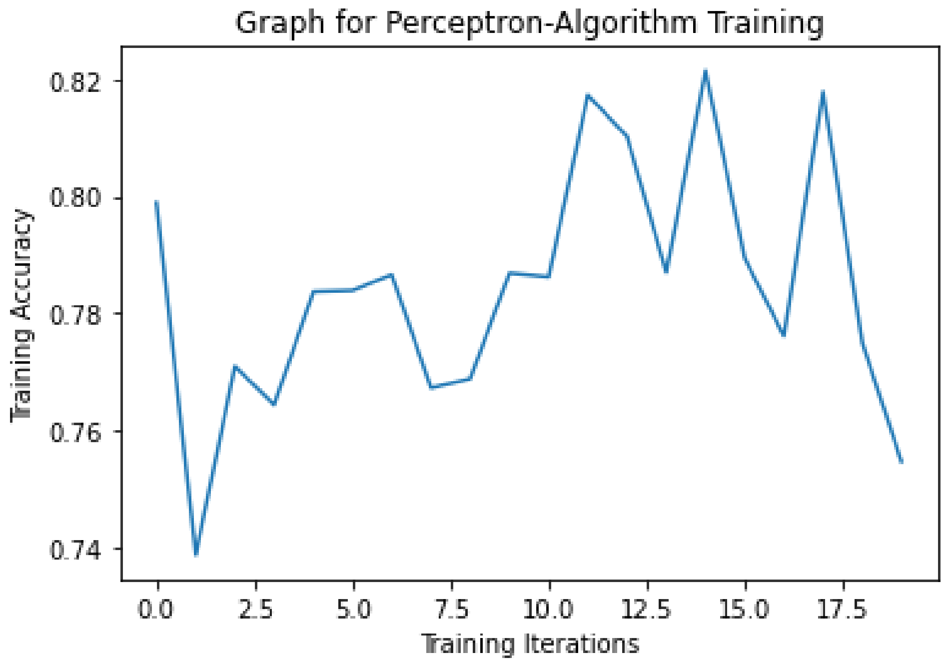
a. Compute the online learning curve for both Perceptron and PA algorithm by plotting the number of training iterations (1 to 50) on the x-axis and the number of mistakes on the y-axis. Compare the two curves and list your observations.

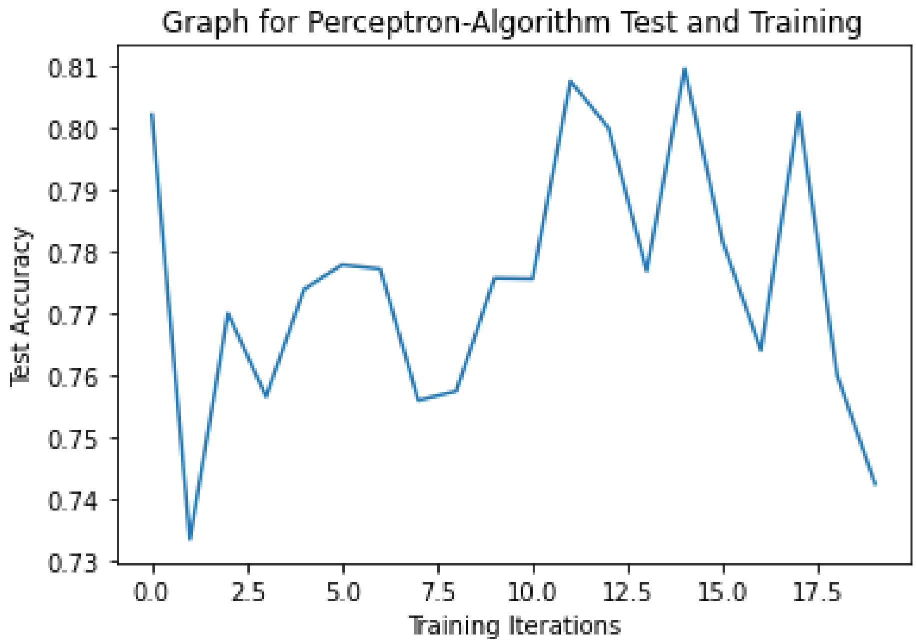


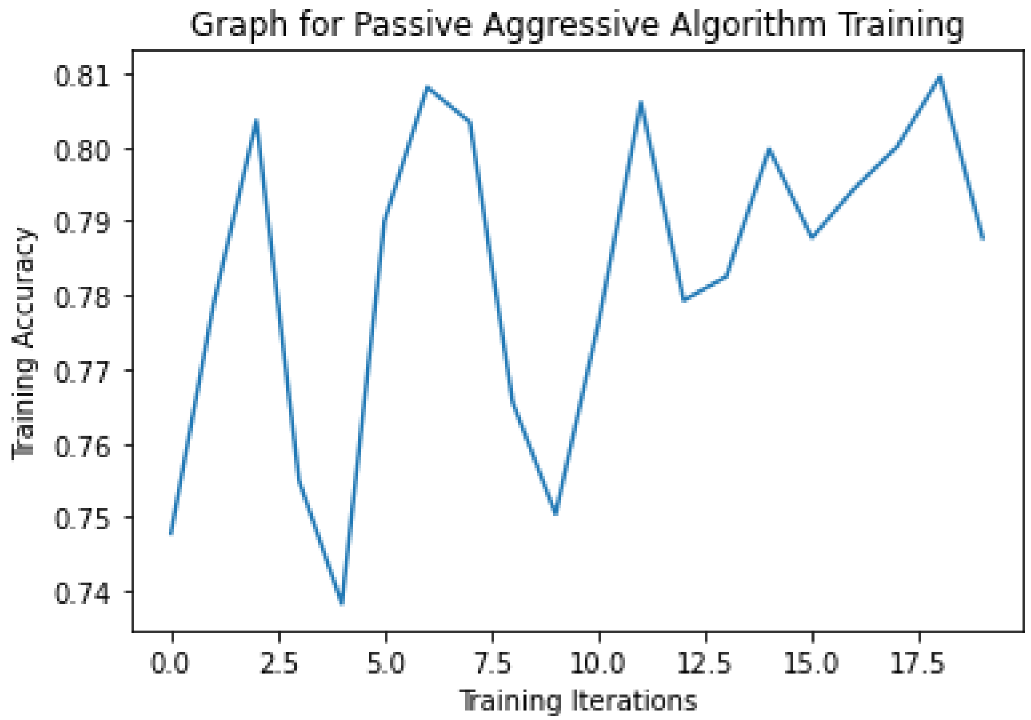


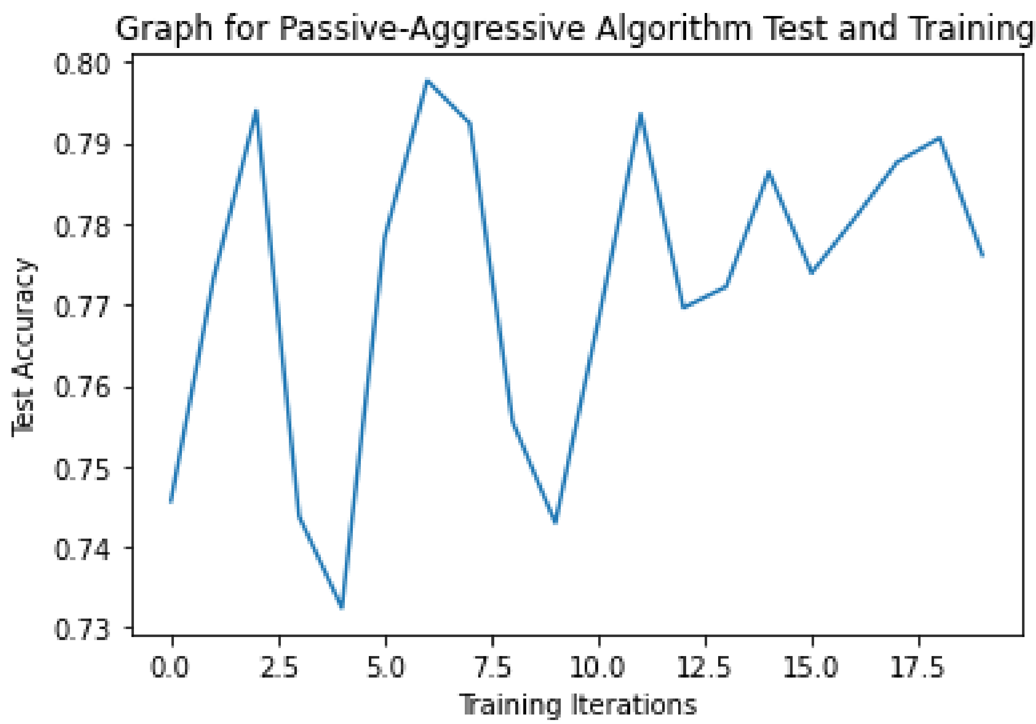
It can be deduced from the graphs above that Online learning curves for Perceptron and Passive Algorithms are more likely similar, but if we look closer at the graphs it can be said that Perceptron algorithm is performing better than the Passive Aggressive Algorithm in terms of testing data.

b. Compute the accuracy of both Perceptron and PA algorithm on the training data and testing data for 20 training iterations. So you will have two accuracy curves for Perceptron and another two accuracy curves for PA algorithm. Compare the four curves and list your observations.



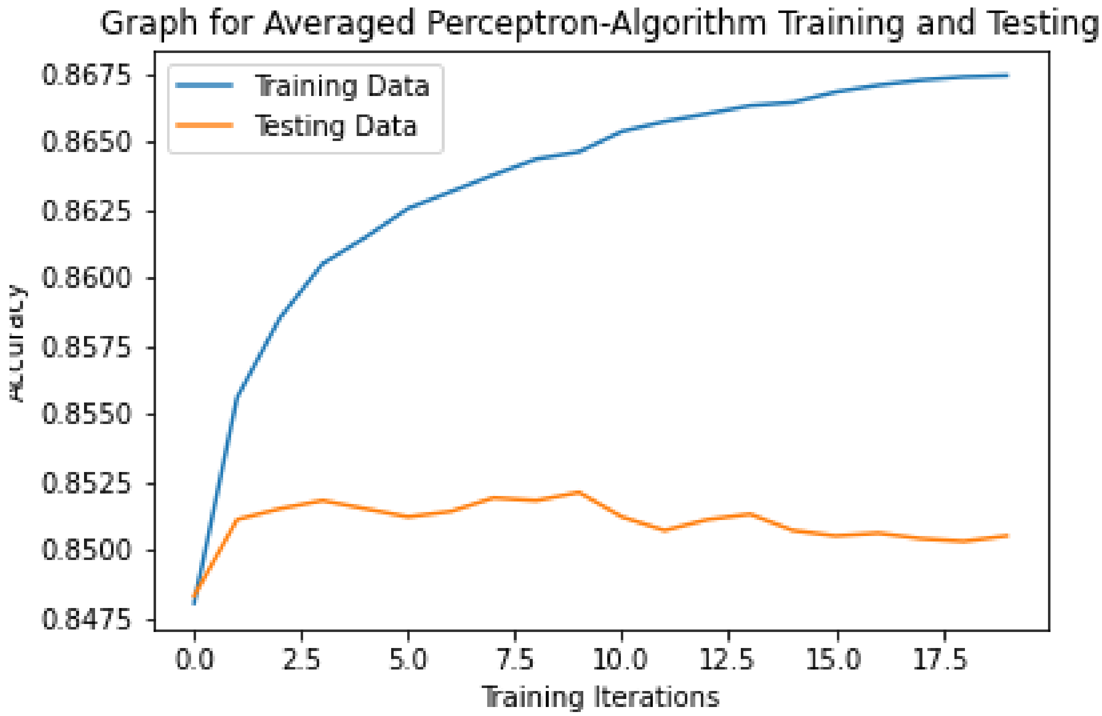






It can be deduced by looking at the above four graphs that performance of weights in terms of training are much more similar and better for both, Perceptron and Passive Aggressive Algorithms.

c. Repeat experiment (b) with averaged perceptron. Compare the test accuracies of plain perceptron and averaged perceptron. What did you observe?



[0.84803333 0.85558333 0.8585 0.8605 0.86146667 0.86253333 0.86315
0.86376667 0.86435 0.86461667 0.86536667 0.86573333 0.86601667 0.86631667
0.86643333 0.86681667 0.86706667 0.86725 0.86736667 0.86741667]

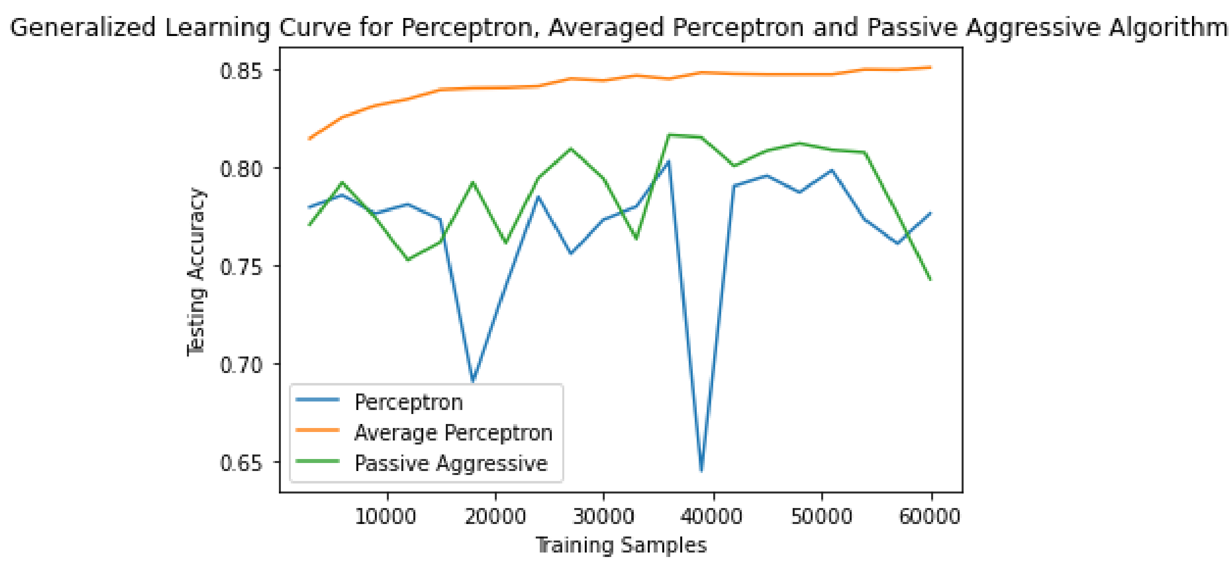
It can be observed from the above graph that training data set performed better than testing data with this algorithm.

d. Compute the general learning curve (vary the number of training examples starting from 100 in the increments of 100) for 20 training iterations. Plot the number of training examples on x-axis and the testing accuracy on the y-axis. List your observations from this curve.

3000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
6000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
9000

Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
12000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
15000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
18000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
21000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
24000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
27000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
30000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
33000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
36000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
39000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron

42000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
45000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
48000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
51000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
54000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
57000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron
60000
Complete_Train_Perceptron
Complete_Train_Passive_Aggressive
Train_Averaged_Perceptron



It can be observed from the graph above that Averaged Perceptron performed better in terms of generalization and pushing of more new data and with further processing its accuracy improves as we train with more data. Furthermore, Perceptron Algorithms also performed better as more data is used and can be said that it generalizes better with new data. Passive Aggressive algorithm's performance is not as better as the other two algorithms. Overall, we can say that Averaged perceptron performed better.