Programming assignment 2

SEIFEDDINE MEJRI (33%)

WAJID RAMEEZ (33%)

ADHIKARI AASHISH (33%)

Introduction:

This Implementation Assignment wants to test our understanding of different types of PERCEPTRON ALGORITHMS. Online Perceptron is the simplest kind where we update the weights after training for each example at each iteration. Average perceptron is an improved version where we use an averaged weight vector to make predictions. Kernelized perceptron, on the other hand, learns alpha values to reconstruct the weight vectors.

PART 1:

a)

0.94742226
0.95765139
0.94987725
0.96481178
0.95683306
0.95765139
0.95887889
0.95662848
0.95908347
0.96215221
0.96215221
0.95908347
0.9646072
0.96624386
0.9564239

Table -1: accuracies for training data for 15 iterations

Following are the accuracies for the validation data for 15 iterations:

0.93370166
0.94536525
0.9349294
0.94843462
0.93677103
0.94352363

0.93799877
0.93861265
0.94782075
0.93922652
0.94045427
0.94290976
0.94229589
0.95273174 → Highest Accuracy for validation data
0.94597913

Table 2 the accuracies for the validation data for 15 iterations

Following is a plot illustrating the training and validation accuracies versus the iteration numbers from 0 to 14 (a total of 15).

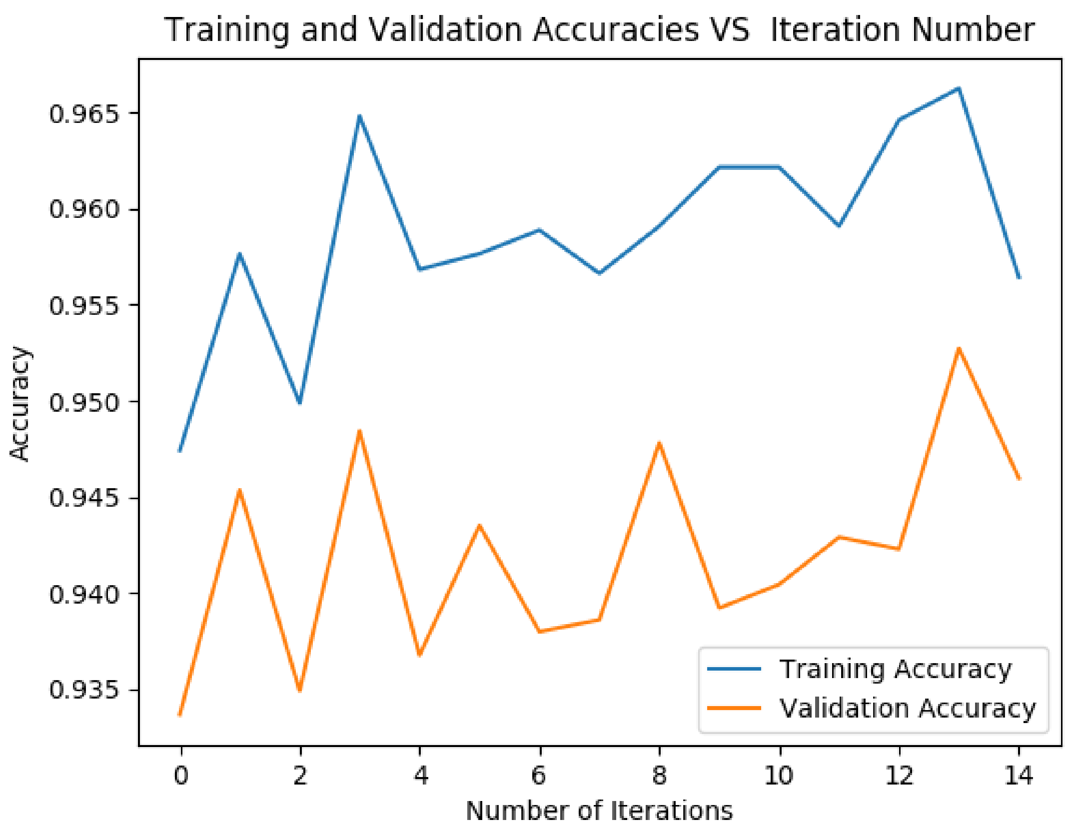

Figure 1: Training and validation accuracy vs iteration

b)

No, the training accuracy does not reach 100% as seen in the figure.

It is because as we can see on the plot, the accuracy decreases at certain iterations and this happens when the model updates its weights and makes incorrect predictions on certain training examples that it predicted correctly in the earlier iteration.. Even if we increase the number of iterations from 15 to a large number, the model might not get an accuracy of 100% if the data itself is not linearly separable. In such cases, we map features into some higher dimension and make the data separable by a hyperplane. This allows a linear model like perceptron to classify data that was not linearly separable in its original dimension.

c)

After plotting the validation accuracy, we see that the best number of iterations is the second last iteration, iteration number 14 (labelled as iteration 13 after starting the loop from 0 instead of 1).

The resulting model was used to generate a prediction file for the test data.

The openlabel.csv file is included in the current folder.

PART 2:

(a) Please see the README.md file on how to run **part2.py**
(b) Running the average perceptron under part2.py with iteration=15 and using matplotlib to plot the validation and train accuracy:

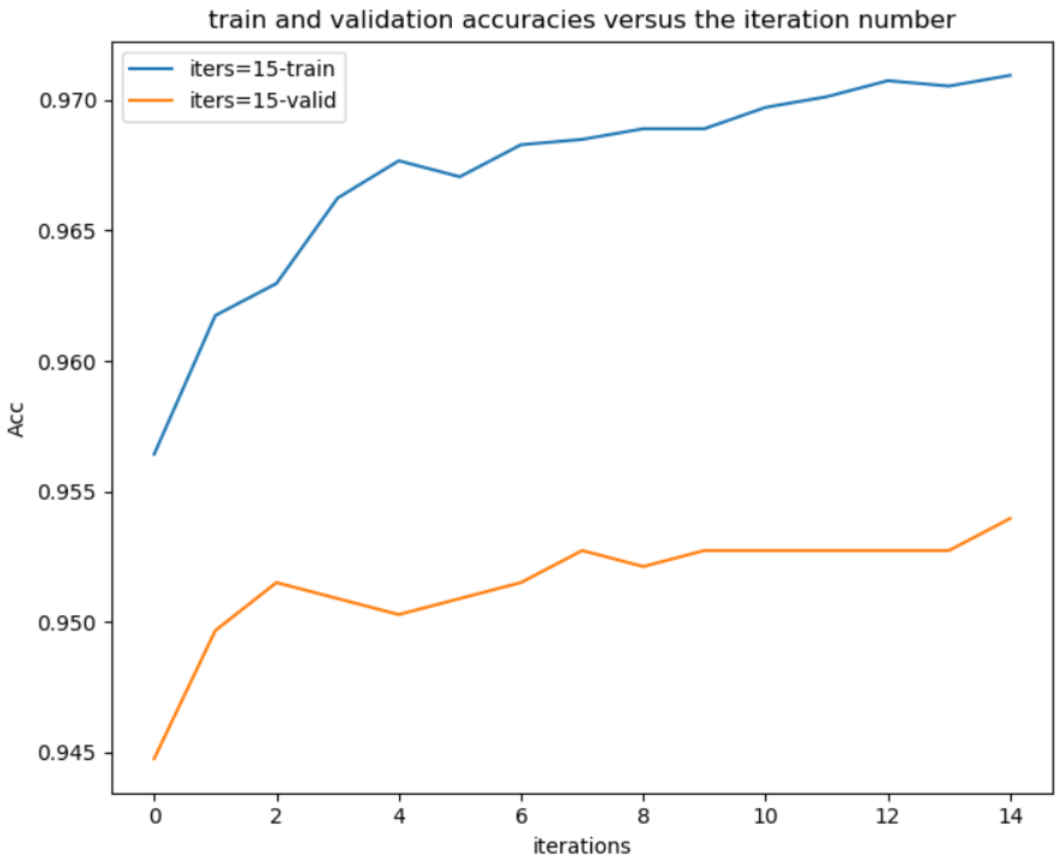

Figure 2 Training and validation accuracy vs iterations

The validation accuracy is less than the training accuracy (observation).

We have obtained for iteration number equal to 15:

Training accuracy	97.09% (at 15^{th} iteration)
Validation accuracy	95.39%(at 15^{th} iteration)

Training accuracy function of number of iterations:

Iter	1^{st}	2^{nd}	3	4	5	6	7	8	9
Iter	10	11	12	13	14	15			

[0.9564239 , 0.96174304 , 0.96297054 , 0.96624386 , 0.96767594 , 0.96706219

0.96828969, 0.96849427 , 0.96890344 , 0.96890344 , 0.96972177 , 0.97013093

0.97074468 , 0.9705401 , 0.97094926]

Validation accuracy function of number of iterations:

[0.94475138 , 0.94966237 , 0.95150399 , 0.95089012 , 0.95027624 , 0.95089012

0.95150399 , 0.95273174 , 0.95211786 , 0.95273174 , 0.95273174 , 0.95273174

0.95273174 , 0.95273174 , 0.95395948]

(c) The average model has better accuracy compared to the online model. In term of validation accuracy, the average perceptron has improved accuracy compared with the online perceptron.
95.39% (Average perceptron) and 95.27% (online perceptron)

Online Perceptron	Average Perceptron
95.27%	95.39%

Table -3 comparison of accuracy

The accuracy of average perceptron has improved comparing to the online perceptron by almost 0.12%.

This is not too much of improvement, but we should expect higher validation accuracy improvement for the average perceptron for much larger dataset that contains larger number of example (In our case, we only have 1629).

PART 3: Kernel perceptron

(a)(b)

For the Kernelized version of the perceptron, a polynomial kernel was used as mentioned in Algorithm 3. A Gram matrix was created using the training samples.

(c)

The Algorithm was implemented for degree (p-values) of [1, 2, 3, 7, 15] . The prediction in the respective cases was computed using samples in the training set with non-zero alpha (also called support vectors). For the validation set we use all samples x_j in the train set for which alpha[j] is non-zero.

The training and validation accuracies for each p-value vs iterations:

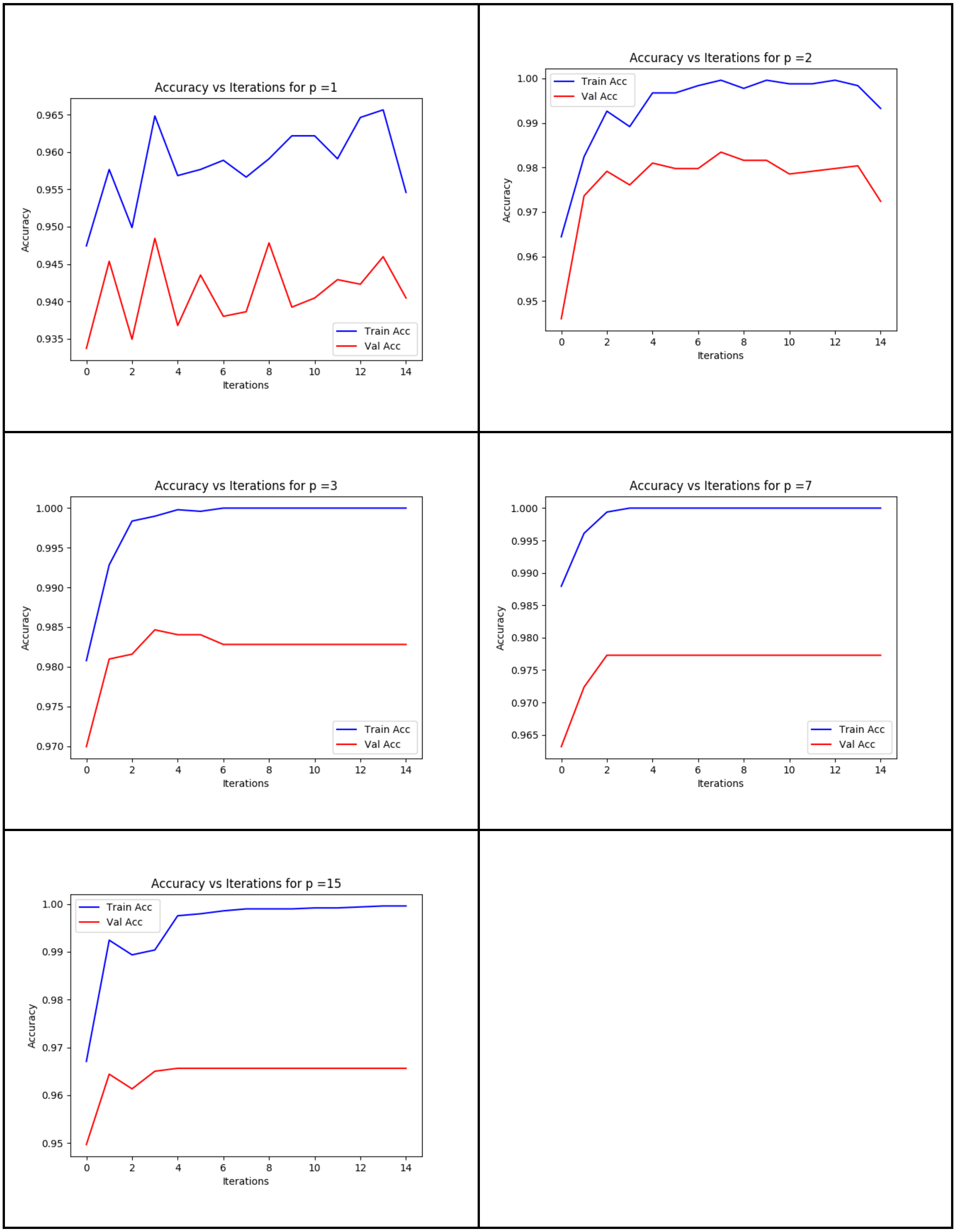

Figure 3: The training and validation accuracies for each p-value vs iterations

c.4)

The maximum validation accuracy for each p over all iterations are shown in the following table:

Polynomial Degree	1	2	3	7	15
Max Validation Accuracy	94.8%	98.3%	98.5%	97.7%	96.6%

Table 4: maximum validation accuracy for each p over all iterations

(d)

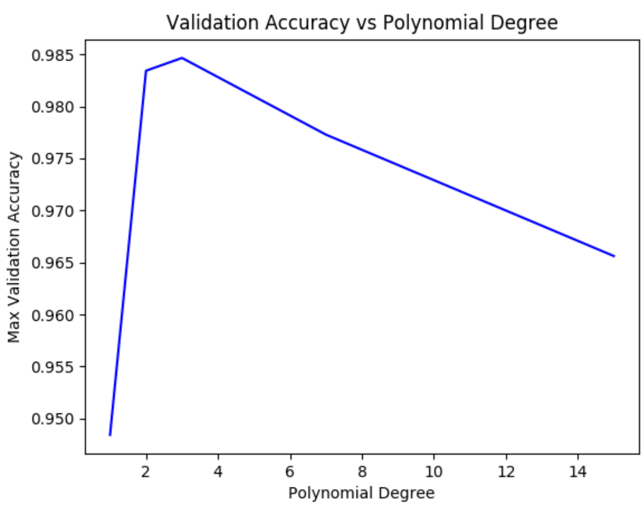

Figure 4: Validation accuracy vs polynomial degree

It can be seen from above results that degree of polynomial effects validation accuracy. At p=1 we have a linear class boundary. As p starts increasing the boundary becomes a curve and hence more flexible. This leads to higher accuracies. However, as p becomes large , the boundary of higher degree tends to overfit over training data. Hence, the validation accuracy starts decreasing after p=3.

(e)

The prediction for the test data was computed using our best results; i.e. the best validation accuracy encountered during training. The optimum results were obtained at *p* =3, and the *4th* iteration. The best validation accuracy achieved was *98.5%* .

A test prediction file '*kplabel.csv*' was generated.

Conclusion:

In this assignment, we studied the behavior of three variations of perceptron. The first variation is Online, then average and finally the kernelized version of perceptron.