Implementation 2

Machine Learning - CS 534

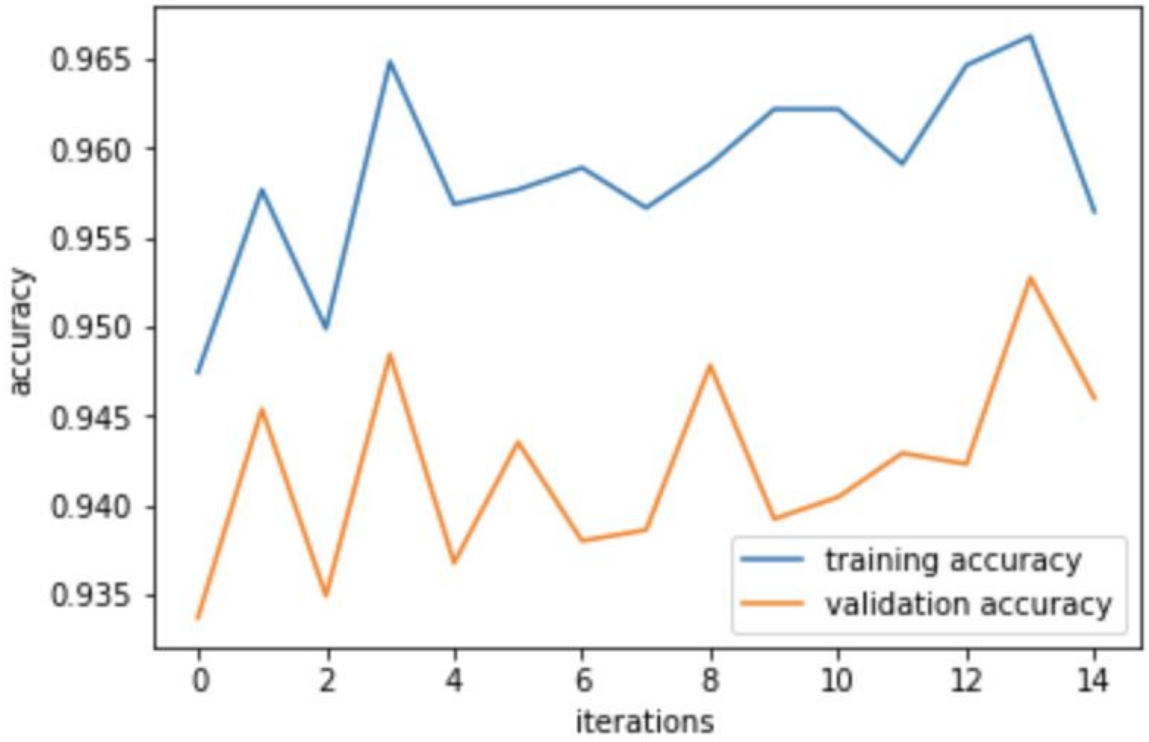
Uday Phalak 933232441.
Parijat Bhatt 933619125.
Akshay Shinde 933589936.

Contribution: Equal contribution for implementation by all members.

Online Perceptron.

(a) Implement the online perceptron model with algorithm described in Algorithm 1. Set the iters = 15. During the training, at the end of each iteration use the current w to make prediction on the validation samples. Record the accuracies for the train and validation at the end of each iteration. Plot the recorded train and validation accuracies versus the iteration number.

>



(b) Does the train accuracy reach to 100%? Why?

> *The train accuracy does not reach 100% because the data may not be linearly separable. Some 3's may look like 5 or some 5's may look like 3.*

(c) Use the validation accuracy to decide the test number for iters. Apply the resulting model to make predictions for the samples in the test set. Generate the prediction file oplabel.csv. Please note that your file should only contain +1 (for 3) and -1 (for 5) and the number of rows should be the same as pa2 test.csv.

> *The validation accuracy peaks on the 14th iteration with accuracy 95.27%. We have used the weight vector from this iteration for prediction. The output file is generated once the program runs.*

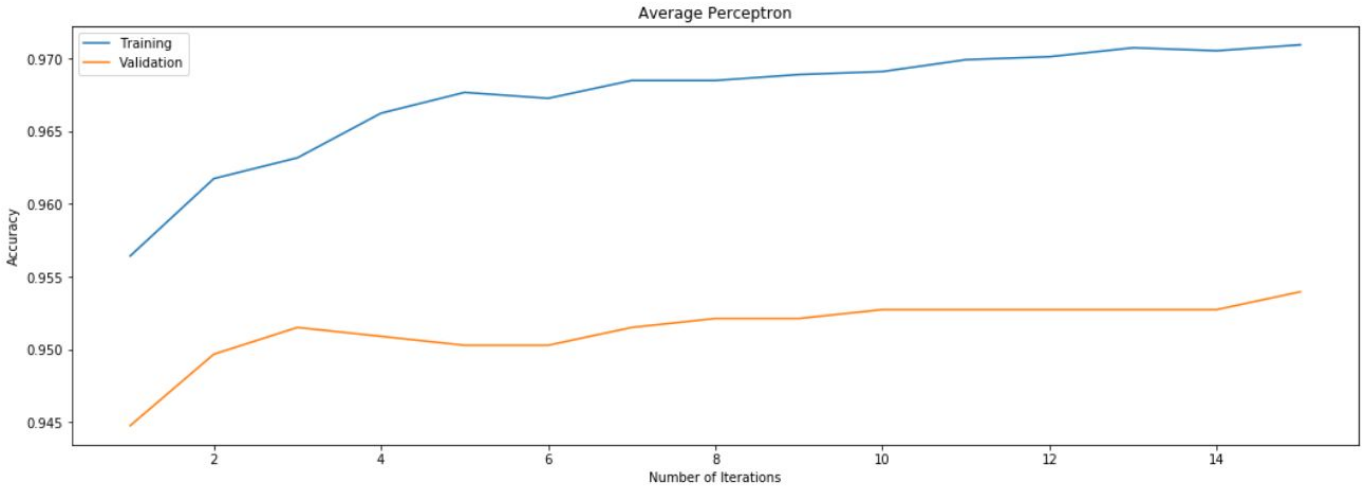
Average Perceptron.

(a) Please implement the average perceptron described in Algorithm 2.

> average_perceptron( )

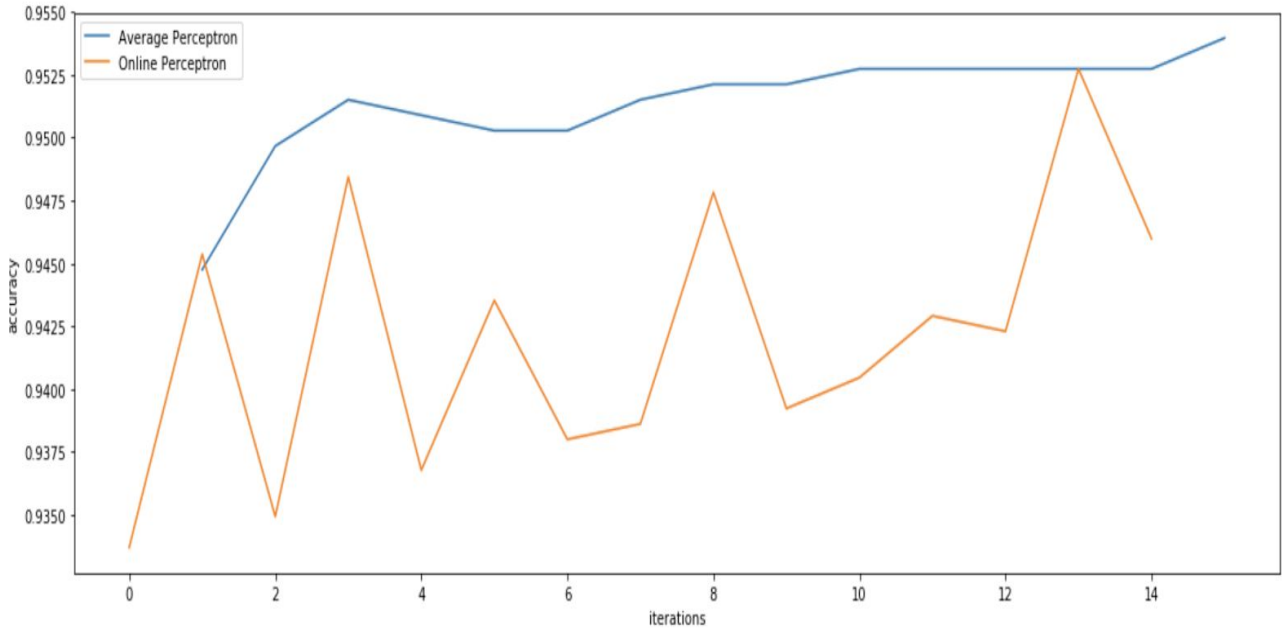
(b) Plot the train and validation accuracies versus the iteration number for iters = 1,..., 15.

>



(c) How the average model has affected the validation accuracy comparing to the online perceptron?

> We are getting smooth curve as compare to online perception which has a zigzag curve. As we are taking weighted average it gets uniformly distributed.



Polynomial Kernel Perceptron.

(a) Implement the polynomial kernel function kp in Algorithm 3. This function takes two vectors x 1 and x 2 and an integer p for the polynomial degree and returns a real value.

> implemented in main.py

(b) Define a Gram matrix K with size N × N where N is the number of training samples. Fill matrix K(i, j) = k p (x i , x j) for all of the pairs in the training set.

> implemented in main.py

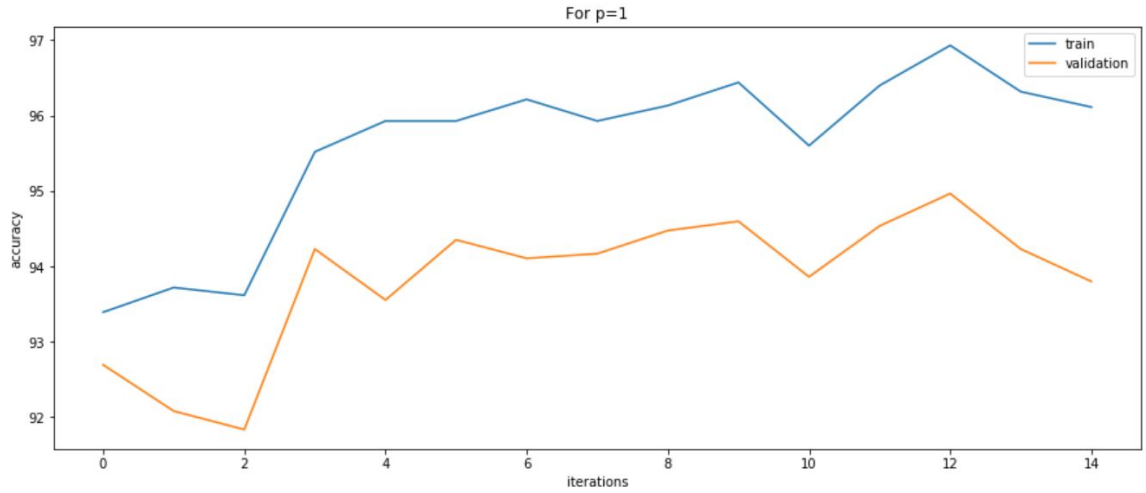
(c) Implement the rest of the kernel perceptron in Algorithm 3. For each p in [1, 2, 3, 7, 15]:

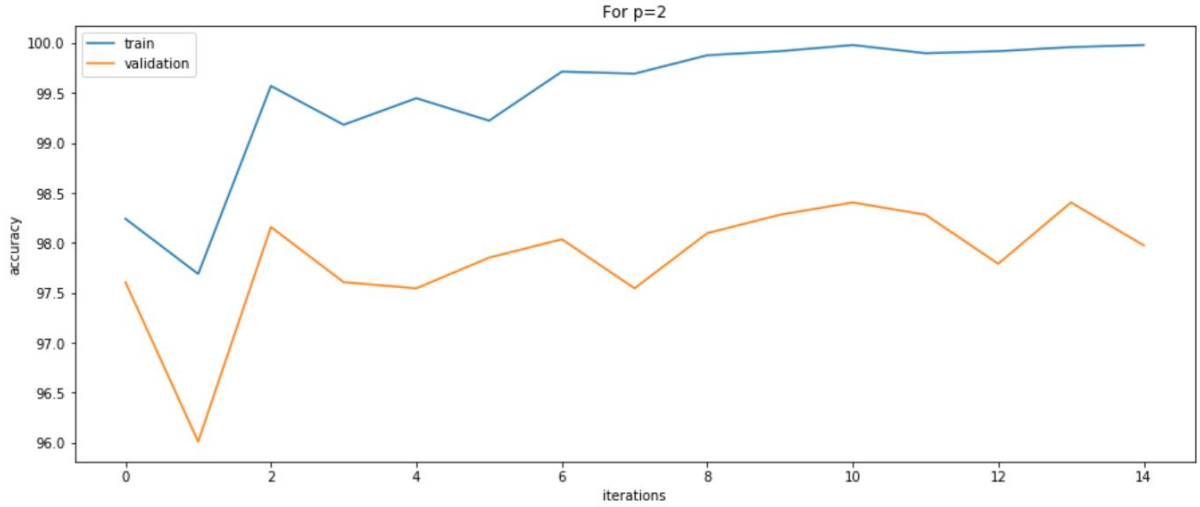
1) Run the algorithm to compute α.

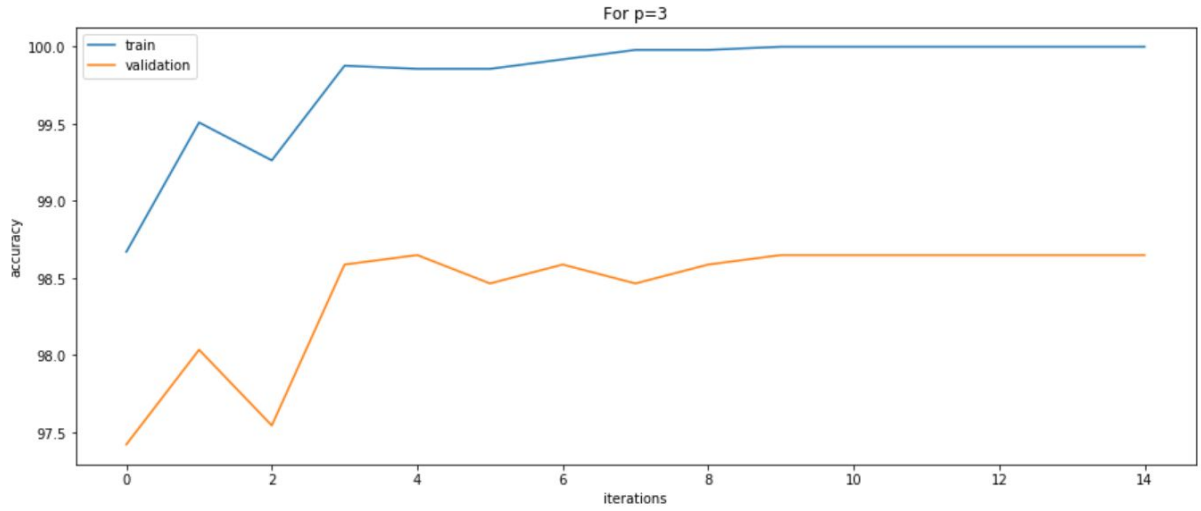
2) At the end of each iteration use the current α to predict the validation set.

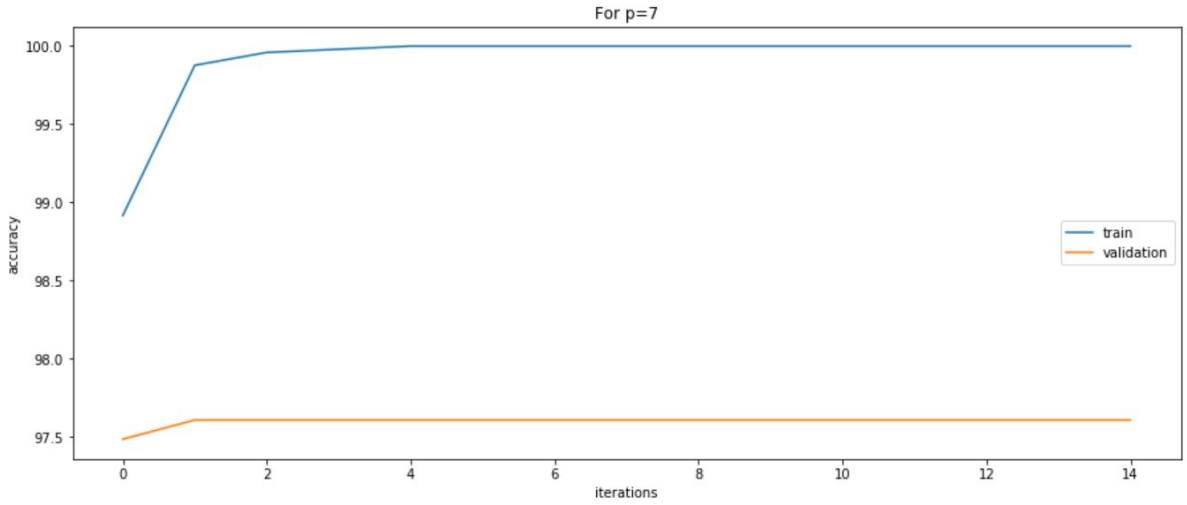
3) Record the train and validation accuracy for each iteration and plot the train and validation accuracies versus the iteration number.

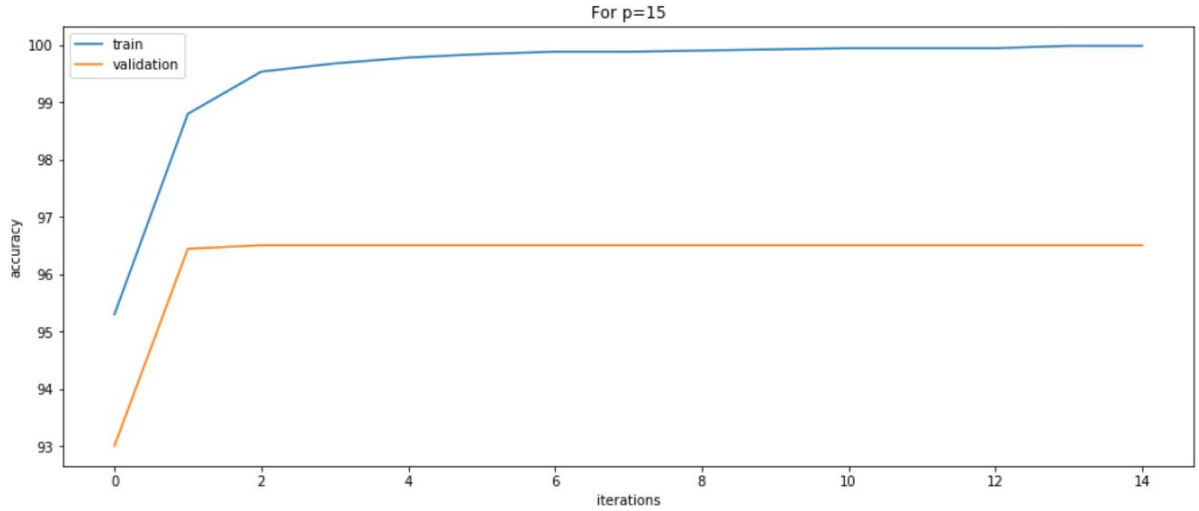
4) Record the best validation accuracy achieved for each p over all iterations.







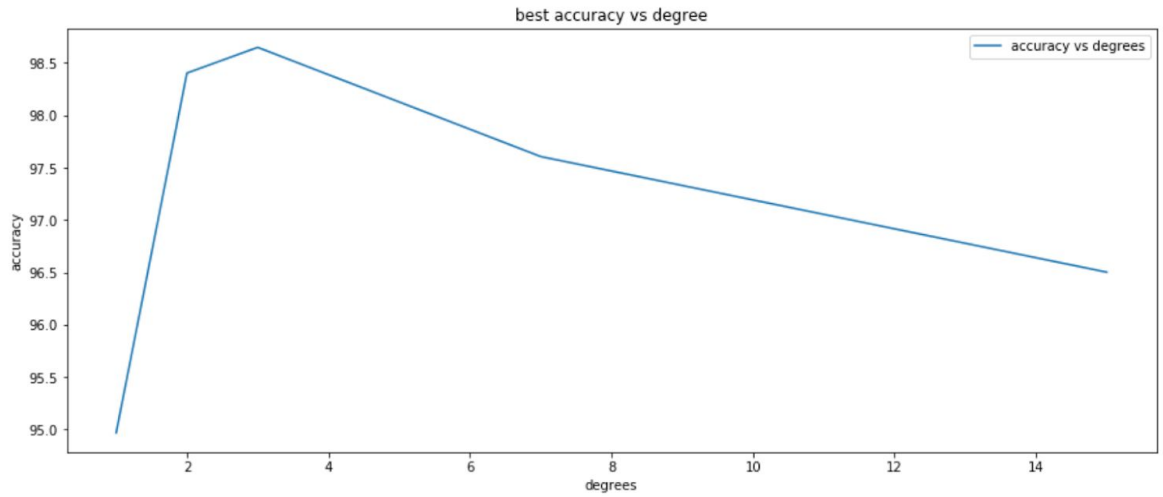




Best validation accuracy achieved for p=3 for iteration number 4

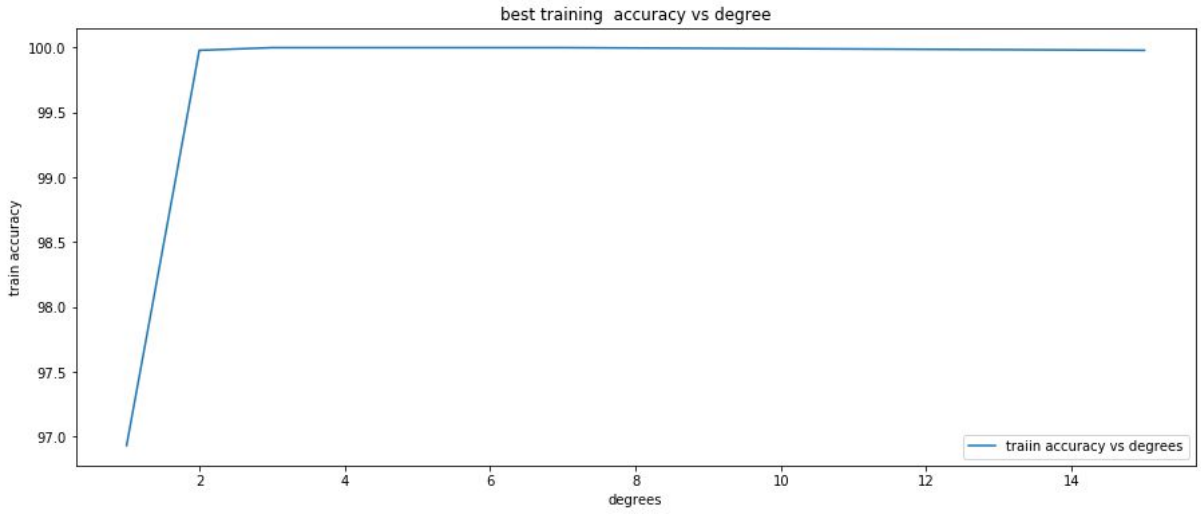
(d) Plot the recorded best validation accuracies versus degrees. Please explain how p is affecting the train and validation performance.

>



Best Validation Accuracies VS Degree

- Performance increase as we move from linear to cubic kernel but further increase in p will overfit the model. If we compare validation and training accuracies for all p's then cubic kernel gives the best performance.



(e) Use your best α (the best you found overall d and iterations above) to predict the test data-set. Please name the predicted file as kplabel.csv.

- For cubic kernel and 4th iteration, we got best alpha.