# CS534 - Machine Learning

# Implementation Assignment 2

# Fall 2019

Praveen Ilango (933597560) - 33.3%
Fransiskus Derian (933296975) - 33.3%
You-Jen Lin (933663373) - 33.3%

Oregon State University
Oct 31, 2019

**Introduction**

The primary purpose of this report is to perform binary classification prediction based on input datasets that contain various handwritten digits features that is suitable for Optical Character Recognition. This report captures different behavior of various Perceptron algorithms such as Online, Average and Kernel Perceptron. The description of each part is described per section in the report below.

**Part 1 (20 pts) : Online Perceptron.** In the online perceptron algorithm we train a linear classifier with parameter $\boldsymbol{w}$ to predict the label of a sample with equation:

$$\hat{y} = \text{sign}(\boldsymbol{w}^T \boldsymbol{x}) \tag{1}$$

Where $\hat{y} \in \{-1, 1\}$. Algorithm 1 describes the online perceptron.

**Algorithm 1** Online Perceptron

**procedure** OnlinePerceptron
 $\boldsymbol{w} \leftarrow \boldsymbol{0}$
 while iter $<$ iters:
  for all sample $\boldsymbol{x_t}$ in train set: // no shuffling
   if $y_t \boldsymbol{w}^T \boldsymbol{x_t} \leq 0$:
    $\boldsymbol{w} \leftarrow \boldsymbol{w} + y_t \boldsymbol{x_t}$
 return $\boldsymbol{w}$

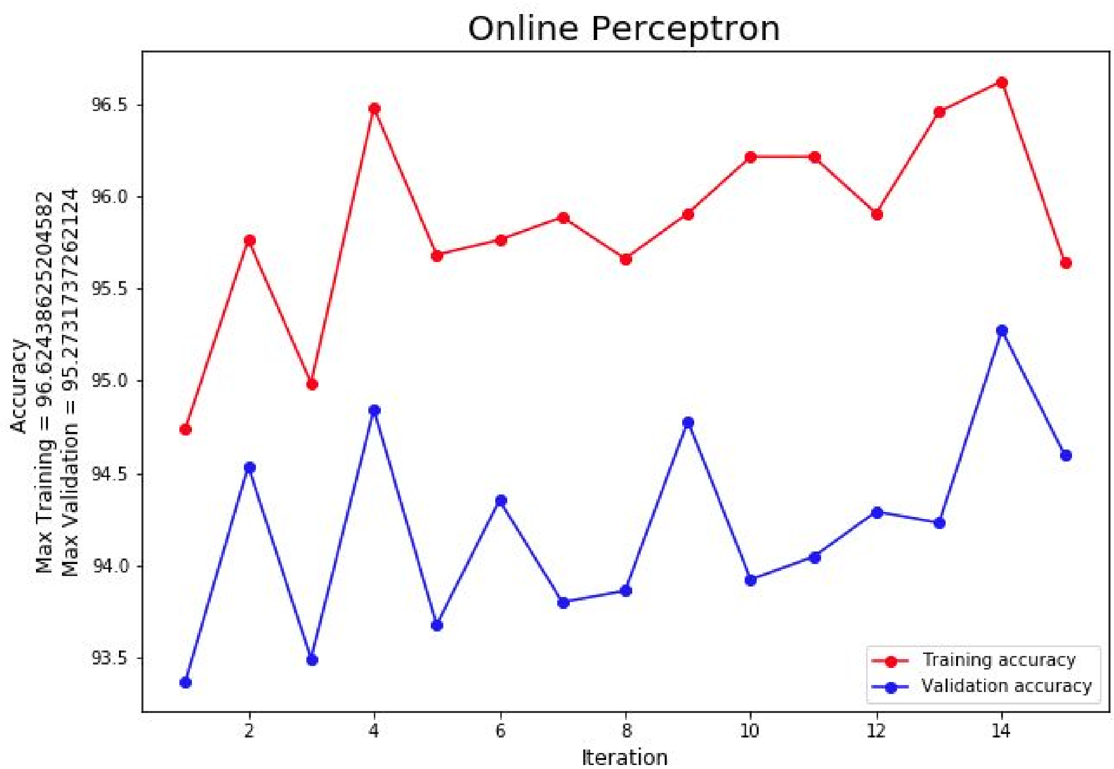
In this part we are interested in the following experiments for the online perceptron algorithm:

(a) Implement online perceptron as described in Algorithm 1. Set the iters = 15. During the training, at the end of each iteration use the current $\mathbf{w}$ 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. Does the train accuracy reach to 100%? Why?

**ANSWER:**

| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 94.7422259 | 93.3701657 |
| 2 | 95.7651391 | 94.5365255 |
| 3 | 94.987725 | 93.4929405 |
| 4 | 96.4811784 | 94.8434622 |
| 5 | 95.6833061 | 93.6771025 |
| 6 | 95.7651391 | 94.3523634 |
| 7 | 95.8878887 | 93.7998772 |

| | | |
|---|---|---|
| 8 | 95.6628478 | 93.8612646 |
| 9 | 95.908347 | 94.7820749 |
| 10 | 96.2152209 | 93.9226519 |
| 11 | 96.2152209 | 94.0454266 |
| 12 | 95.908347 | 94.2909761 |
| 13 | 96.4607201 | 94.2295887 |
| 14 | 96.6243863 | 95.2731737 |
| 15 | 95.6423895 | 94.5979128 |




No, the train accuracy does not reach 100%. This could because we stopped after 15 iterations and the algorithms might need more iterations to converge. Another possibility is that the data is not linearly separable.

(b) Use the validation accuracy to decide the best value for *iters*. Apply the corresponding learned 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.

Output is generated as “oplabel.csv”

**Part 2 (20 pts) : Average Perceptron.** In this part you will implement and experiment with average perceptron, which is described in Algorithm 2.

**Algorithm 2** Average Perceptron

**procedure** AVERAGEPERCEPTRON
    $\boldsymbol{w} \leftarrow \boldsymbol{0}$ //initialize weight vector
    $\bar{\boldsymbol{w}} \leftarrow \boldsymbol{0}$ // initialize average weight vector
    $s \leftarrow 1$ //initialize the example counter to 1
    while iter < iters:
        for all sample $\boldsymbol{x_t}$ in the train set: // no shuffling
            if $y_t \boldsymbol{w}^T \boldsymbol{x_t} \leq 0$:
                $\boldsymbol{w} \leftarrow \boldsymbol{w} + y_t \boldsymbol{x_t}$
            $\bar{\boldsymbol{w}} \leftarrow \frac{s\bar{\boldsymbol{w}} + \boldsymbol{w}}{s+1}$ // update $\bar{\boldsymbol{w}}$ regardless of $\boldsymbol{w}$ udpate
            $s \leftarrow s + 1$
    return $\bar{\boldsymbol{w}}$

Note that in Algorithm 2, the running average $\bar{\boldsymbol{w}}$ always gets updated every time an example is process, regardless whether the current $\boldsymbol{w}$ correctly classify it or not [1].

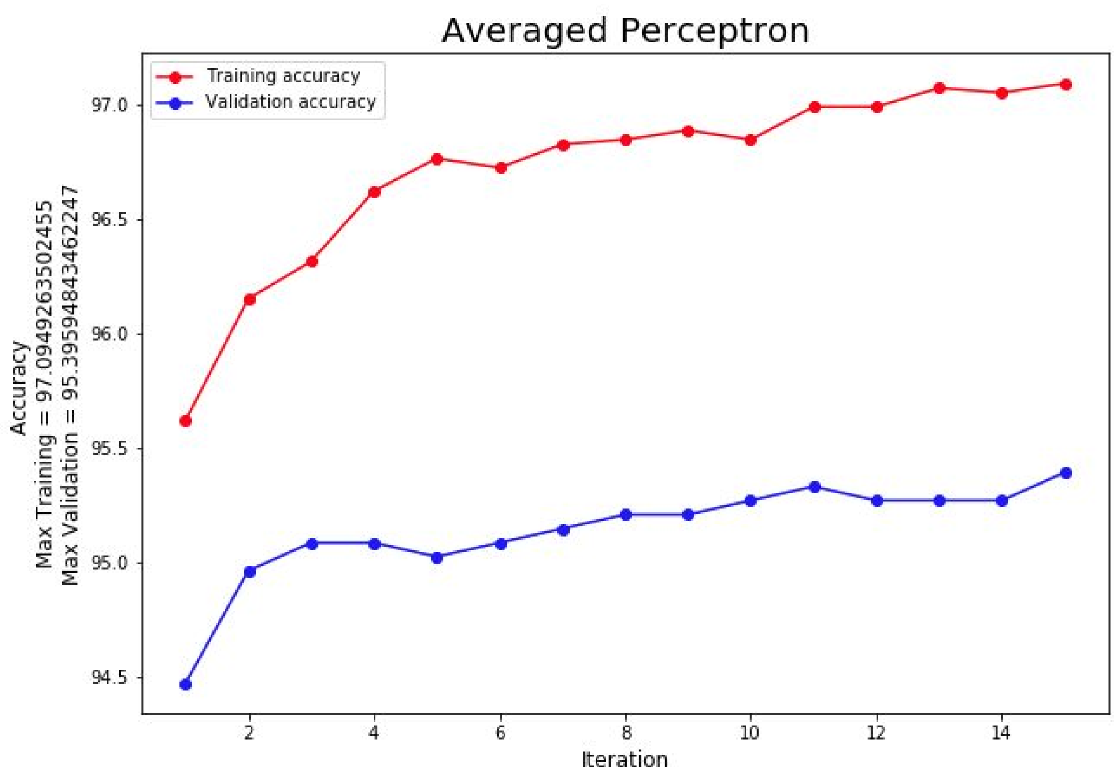
Perform the following experiments:

(a) Apply your implemeted average perceptron to learn from the training data. Plot the train and validation accuracies versus the iteration number for iters = 1, ..., 15.

**ANSWER:**

| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 95.6219313 | 94.4751381 |
| 2 | 96.1538462 | 94.966237 |
| 3 | 96.3175123 | 95.0890117 |
| 4 | 96.6243863 | 95.0890117 |
| 5 | 96.7675941 | 95.0276243 |
| 6 | 96.7266776 | 95.0890117 |
| 7 | 96.8289689 | 95.150399 |
| 8 | 96.8494272 | 95.2117864 |

| | | |
|---|---|---|
| 9 | 96.8903437 | 95.2117864 |
| 10 | 96.8494272 | 95.2731737 |
| 11 | 96.992635 | 95.3345611 |
| 12 | 96.992635 | 95.2731737 |
| 13 | 97.0744681 | 95.2731737 |
| 14 | 97.0540098 | 95.2731737 |
| 15 | 97.0949264 | 95.3959484 |




(b) What are your observations when comparing the training accuracy and validation accuracy curves of the average perceptron with those of the online perceptron? What are your explanation for the observations?

**ANSWER:**

We observe that the accuracy curve of average perceptron is smoother with less oscillations in comparison to the online perceptron. The maximum validation accuracy achieved for average perceptron(95.3959484) is slightly greater than that of online perceptron(95.2731737).
The graph is smoother for average perceptron as we take all possible weight vectors encountered during training into consideration and make predictions using the average of all the encountered weights.

(c) Use the validation accuracy to decide the best value for *iter* and apply the corresponding learned model to make predictions for the test data. Please name the predicted file as aplabel.csv.

Output is generated as “aplabel.csv”

**Part 3 (40 pts). Polynomial Kernel Perceptron.** The online/average perceptron in Algorithm( 1 and 2) are linear models. In this part we will consider kernelized perceptron, as described in Algorithm 3, with a polynomial kernel $k_p$ of degree $p$:

$$k_p(\boldsymbol{x}_1, \boldsymbol{x}_2) = (1 + \boldsymbol{x}_1^T \boldsymbol{x}_2)^p \tag{2}$$

**Algorithm 3** Kernel (polynomial) Perceptron

**procedure** KernelPerceptron
    $\alpha_i \leftarrow 0$ for $i = 1, ..., N$
    compute the gram matrix $K(i, j) = k_p(\boldsymbol{x}_i, \boldsymbol{x}_j)$
    while iter $<$ iters:
        for all sample $\boldsymbol{x}_t$ in training set: // no shuffling
            $u = \sum_i \alpha_i K(i, t) y_i$
            if $y_t u \leq 0$:
                $\alpha_t \leftarrow \alpha_t + 1$

Implement the kernelized perceptron with polynomial kernel and perform the following experiments:

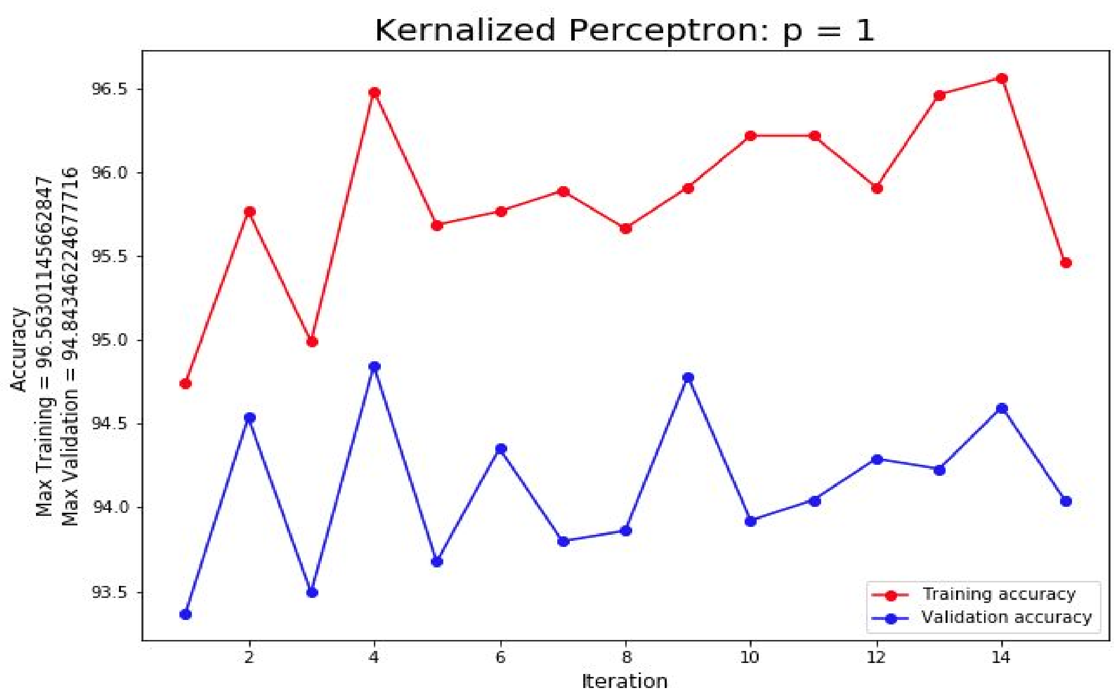
(a) Apply the kernelized perceptron with different $p$ values in [1, 2, 3, 4, 5]:
1) For each $p$ value, run your algorithm with $iter = 15$. At the end of each iteration use the current model (aka $\alpha$'s) to make prediction for the validation set. Record the train and validation accuracy for each iteration and plot the train and validation accuracies versus the iteration number.
2) Record the best validation accuracy achieved for each $p$ over all iterations. Plot the recorded best validation accuracies versus $p$. How do you think $p$ is affecting the train and validation performance?

**ANSWER:**

P = 1

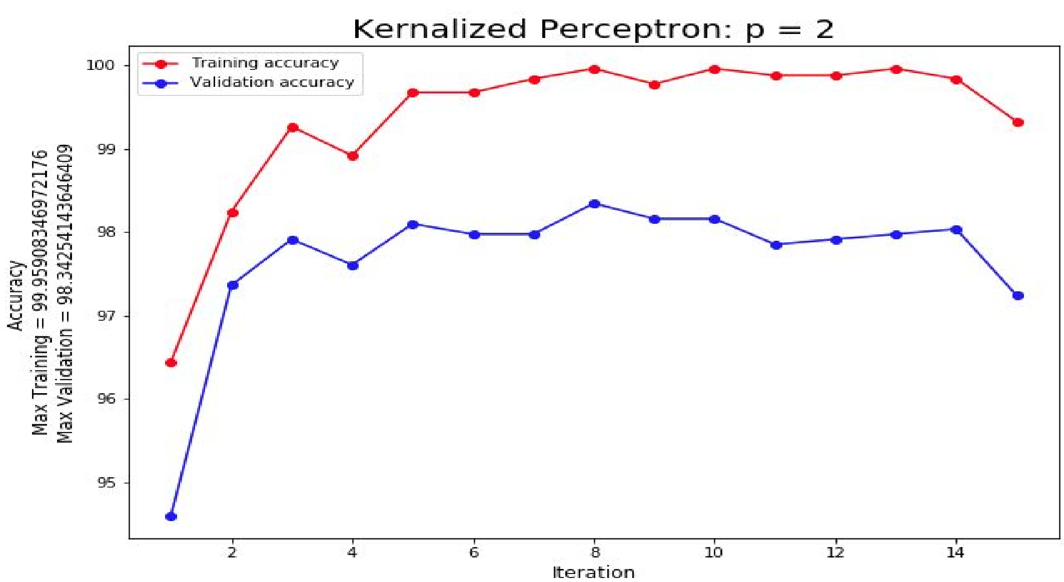
| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 94.7422259 | 93.3701657 |
| 2 | 95.7651391 | 94.5365255 |
| 3 | 94.987725 | 93.4929405 |

| | | |
|---|---|---|
| 4 | 96.4811784 | 94.8434622 |
| 5 | 95.6833061 | 93.6771025 |
| 6 | 95.7651391 | 94.3523634 |
| 7 | 95.8878887 | 93.7998772 |
| 8 | 95.6628478 | 93.8612646 |
| 9 | 95.908347 | 94.7820749 |
| 10 | 96.2152209 | 93.9226519 |
| 11 | 96.2152209 | 94.0454266 |
| 12 | 95.908347 | 94.2909761 |
| 13 | 96.4607201 | 94.2295887 |
| 14 | 96.5630115 | 94.5979128 |
| 15 | 95.4582651 | 94.0454266 |



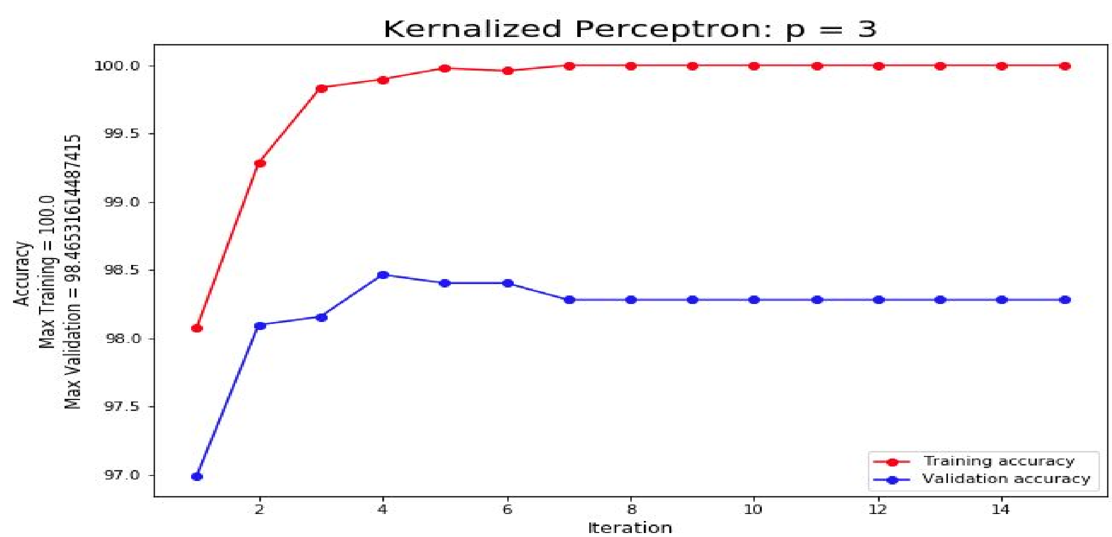
P = 2

| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 96.4402619 | 94.5979128 |
| 2 | 98.2405892 | 97.3603438 |
| 3 | 99.2635025 | 97.91283 |
| 4 | 98.9157119 | 97.6058932 |
| 5 | 99.6726678 | 98.096992 |
| 6 | 99.6726678 | 97.9742173 |
| 7 | 99.8363339 | 97.9742173 |
| 8 | 99.9590835 | 98.3425414 |
| 9 | 99.7749591 | 98.1583794 |
| 10 | 99.9590835 | 98.1583794 |
| 11 | 99.8772504 | 97.8514426 |
| 12 | 99.8772504 | 97.91283 |
| 13 | 99.9590835 | 97.9742173 |
| 14 | 99.8363339 | 98.0356047 |
| 15 | 99.3248773 | 97.2375691 |



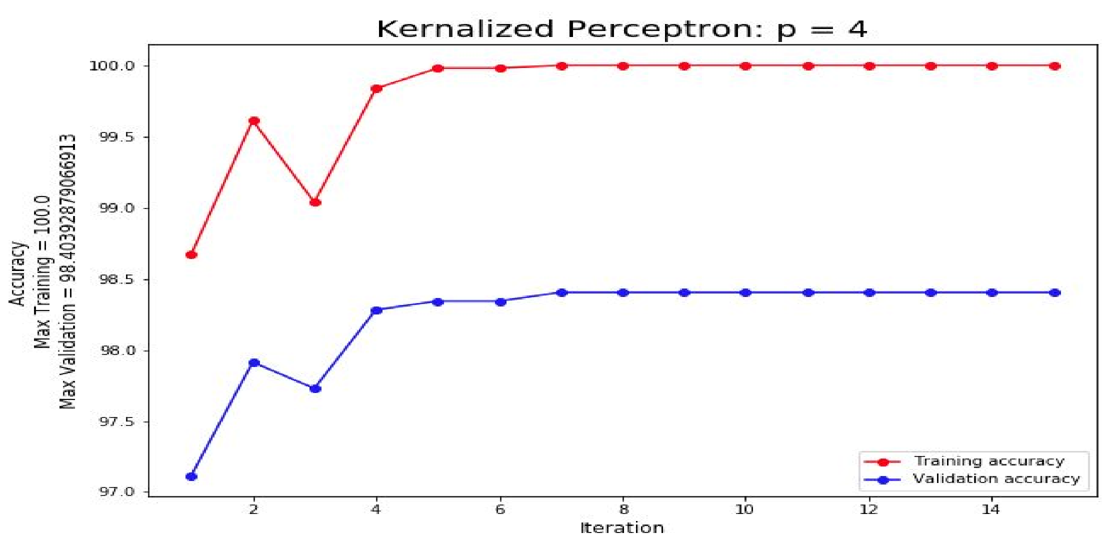
P = 3

| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 98.0769231 | 96.9920196 |
| 2 | 99.2839607 | 98.096992 |
| 3 | 99.8363339 | 98.1583794 |
| 4 | 99.8977087 | 98.4653161 |
| 5 | 99.9795417 | 98.4039288 |
| 6 | 99.9590835 | 98.4039288 |
| 7 | 100 | 98.2811541 |
| 8 | 100 | 98.2811541 |
| 9 | 100 | 98.2811541 |
| 10 | 100 | 98.2811541 |
| 11 | 100 | 98.2811541 |
| 12 | 100 | 98.2811541 |
| 13 | 100 | 98.2811541 |
| 14 | 100 | 98.2811541 |
| 15 | 100 | 98.2811541 |



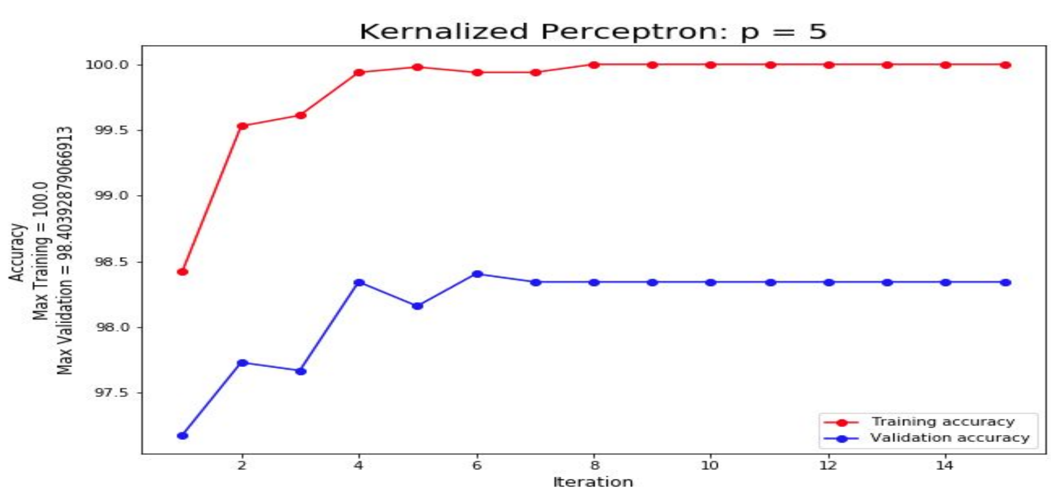
P = 4

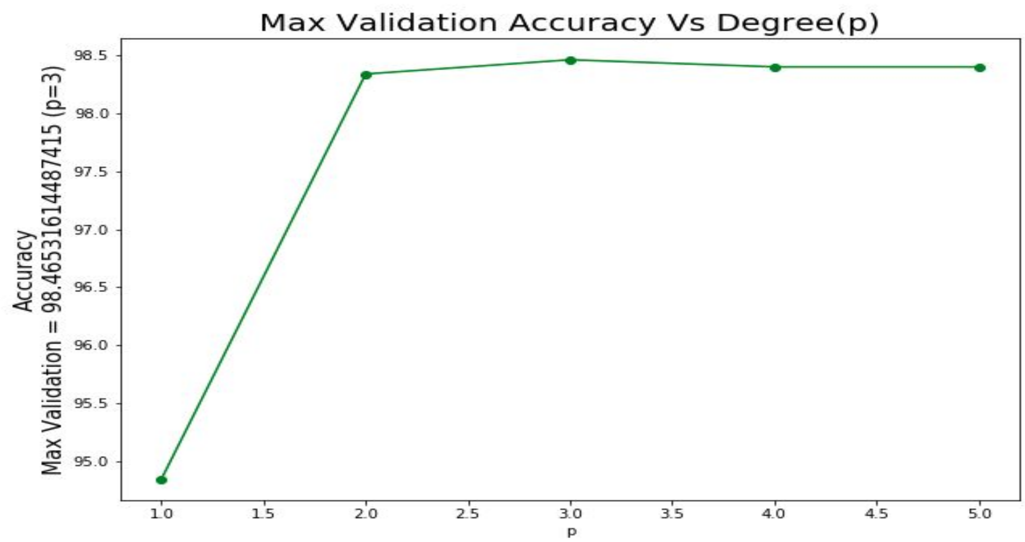
| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 98.6702128 | 97.1147944 |
| 2 | 99.611293 | 97.91283 |
| 3 | 99.0384615 | 97.7286679 |
| 4 | 99.8363339 | 98.2811541 |
| 5 | 99.9795417 | 98.3425414 |
| 6 | 99.9795417 | 98.3425414 |
| 7 | 100 | 98.4039288 |
| 8 | 100 | 98.4039288 |
| 9 | 100 | 98.4039288 |
| 10 | 100 | 98.4039288 |
| 11 | 100 | 98.4039288 |
| 12 | 100 | 98.4039288 |
| 13 | 100 | 98.4039288 |
| 14 | 100 | 98.4039288 |
| 15 | 100 | 98.4039288 |



P = 5

| Iteration | Training Accuracy | Validation Accuracy |
|---|---|---|
| 1 | 98.4247136 | 97.1761817 |
| 2 | 99.5294599 | 97.7286679 |
| 3 | 99.611293 | 97.6672805 |
| 4 | 99.9386252 | 98.3425414 |
| 5 | 99.9795417 | 98.1583794 |
| 6 | 99.9386252 | 98.4039288 |
| 7 | 99.9386252 | 98.3425414 |
| 8 | 100 | 98.3425414 |
| 9 | 100 | 98.3425414 |
| 10 | 100 | 98.3425414 |
| 11 | 100 | 98.3425414 |
| 12 | 100 | 98.3425414 |
| 13 | 100 | 98.3425414 |
| 14 | 100 | 98.3425414 |
| 15 | 100 | 98.3425414 |






| p | Accuracy |
|---|---|
| 1 | 94.8434622 |
| 2 | 98.3425414 |
| 3 | 98.4653161 |
| 4 | 98.4039288 |
| 5 | 98.4039288 |

As p increases from 1 to 2, the validation accuracy increases significantly. But further increase in p causes the validation accuracy to diminish. This could be because the model is overfitting to the training data.

(b) Use your best model (the best one you found over all $p$ values and all iterations above) to make prediction for the test set. Please name the predicted file as kplabel.csv.

Output is generated as “kplabel.csv”