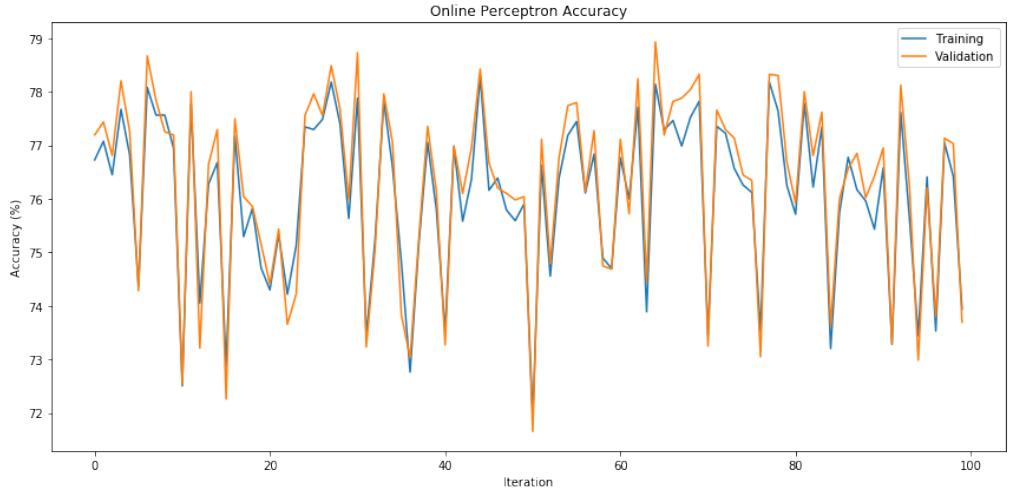
Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

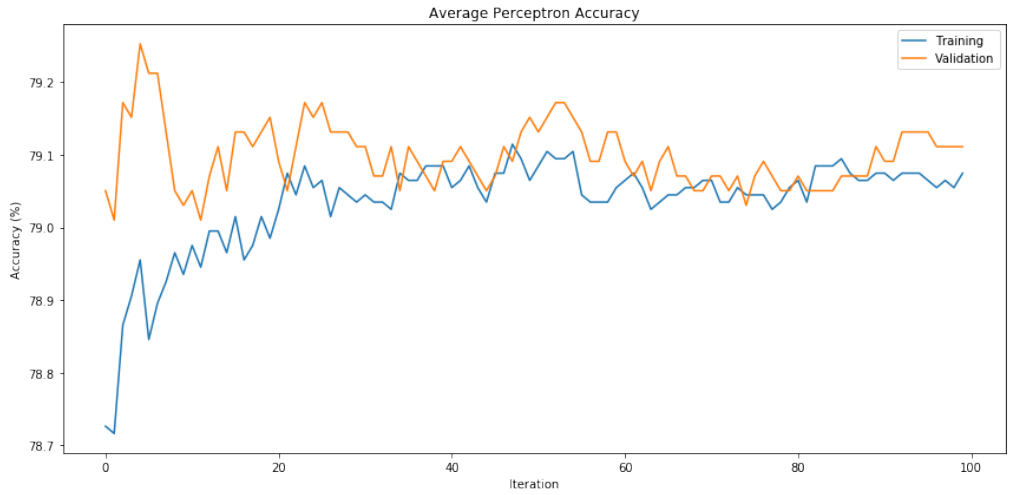
Introduction

For this assignment, we implemented various perceptron algorithms (online perceptron, average perceptron, and kernelized perceptron), which learned from a provided dataset and examined the accuracies of each algorithm and were tasked to find the number of iterations that achieved the best validation accuracy. For the kernelized perceptron, 5 different experiments were ran to compare the accuracies for each polynomial value ($p \in \{1, 2, 3, 4, 5\}$) used for the kernel. We were also given a task to optimize the algorithm to the best of our ability and plot the runtime as a function of iterations. The results and explanations for each section are written below:

Part 1 (35 pts) : Average Perceptron.

(a) Apply your implemented algorithm to learn from the training data with maxiter = 100. Plot the train and validation accuracy of $\boldsymbol{w}$ (online perceptron) and $\overline{\boldsymbol{w}}$ (average perceptron) at the end of each training iteration.





Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

(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?

- It can be observed from the plots that the online perceptron is much noisier compared to the average perceptron and seems to jump between 72% and 79% of accuracy. Unlike the online perceptron, the average perceptron seems to have a smoother convergence at 79%
- This makes sense since an online perceptron evaluates each data point independently causing the change in weights to be large resulting it to be noisy. Averaging the weights smooths out this noise and also indirectly keeps a history of the weights which keeps it more stable and achieve a better accuracy.

(c) Focusing on average perceptron, use the validation accuracy to decide the best number of iterations to stop.

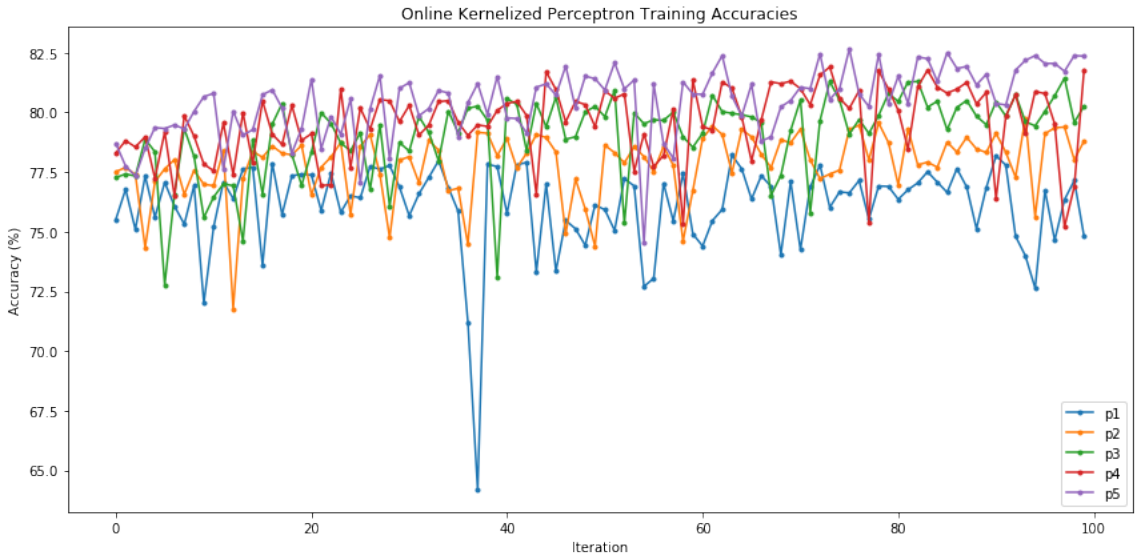
- Based on the average perceptron accuracy plot, the best number of iterations would be around 55 iterations. Although the validation accuracy was able to achieve its highest on the 4^{th} iteration, this may be overfitting the validation data since the training data have not yet converged so choosing a stopping point when the training data have converged maybe a safer choice.
- After 55 iterations, it can be observed that the accuracy has started declining, so training more than this would yield to no benefits.

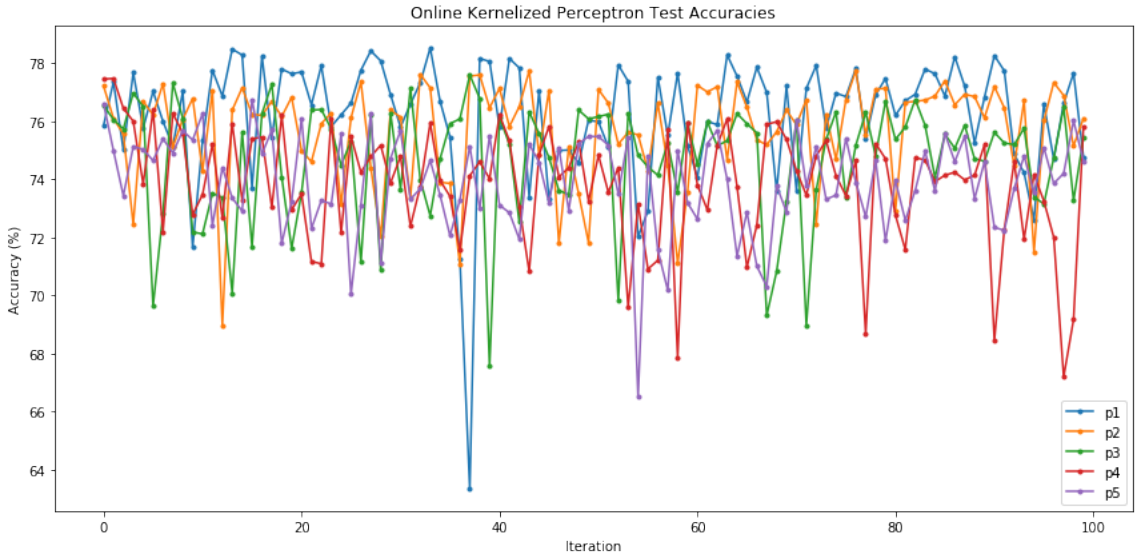
Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

Part 2 (55 pts). Perceptron with Polynomial Kernel.

Part 2a. (40 pts) With your implementation of Algorithm 2, perform the following experiments:

(a) For each p value, at the end of each training iteration use the current model (aka the current set of α's) to make prediction for both the training and validation set. Record and plot the train and validation accuracy as a function of training iterations.

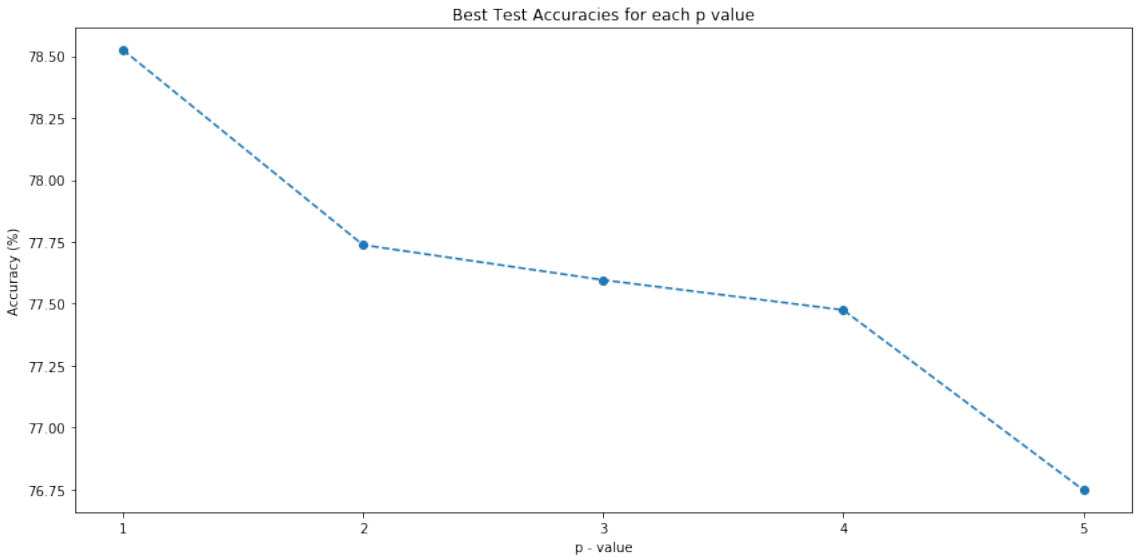




Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

(b) Record the best validation accuracy achieved for each p value over all iterations. Plot the recorded best validation accuracy versus p. How do you think p is affecting the train and validation accuracy and what is your explanation for the observation?

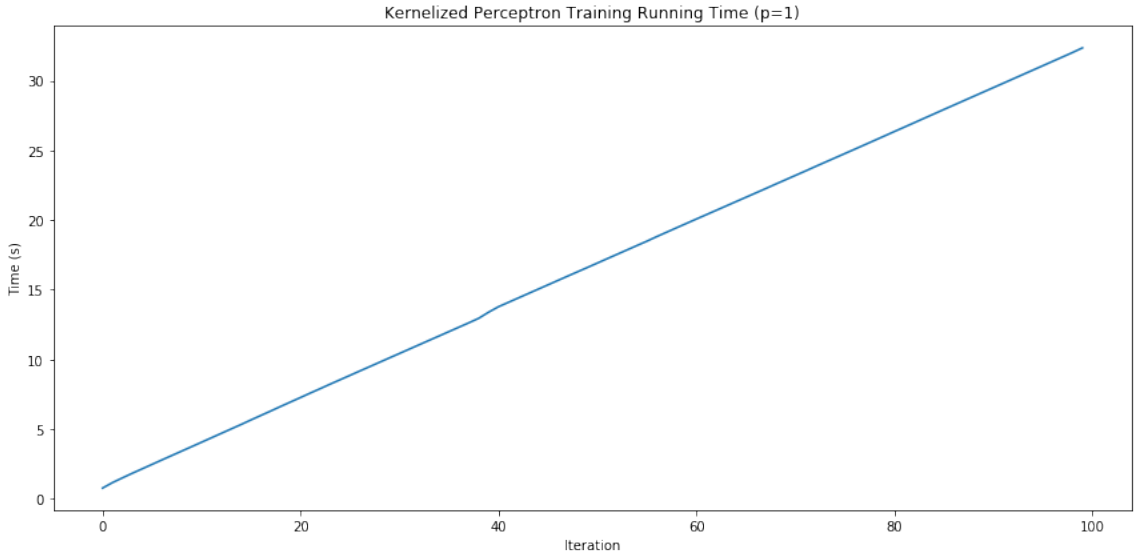
- The p-value in this experiment maps the data to a higher dimensional space that can introduce linear separability and as observed from the plots, increasing the p-value decreases the performance in the validation, however, the opposite trend is observed in the training accuracy.
- This is a sign that as the p-value increases, it overfits the training data more and more, resulting to a poor validation performance. It seems that mapping the data to a higher dimensional space is prone to overfitting the data.



Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

(c) What is the asymptotic runtime of your algorithm in terms of the number of training examples n? Try to make your implementation as efficient as possible. For the p value chosen above, plot the empirical runtime of your algorithm as a function of the iterations.

- The asymptotic runtime of the algorithm in terms of examples n would be **O(n * maxiter)**
- After optimizing the algorithm as efficient as possible, each iteration took an average of 37 ms and the algorithm took a total runtime of 34 s for 100 iterations



Part 2b. (15 pts) Algorithm 2 implements the online kernelized perceptron. Now please modify it to implement batch kernelized perceptron with tunable learning rate and perform the following experiments:

(a) Apply your batch kernel perceptron with the best p value selected in part 2a. You should tune your learning rate as well as the number of iterations to maximize the validation accuracy. Report the configuration of learning rate and iteration number with the best validation accuracy you achieve.

- Any learning rate can be used since it doesn't make an impact
- The algorithm was able to achieve the highest validation accuracy on the 73rd iteration

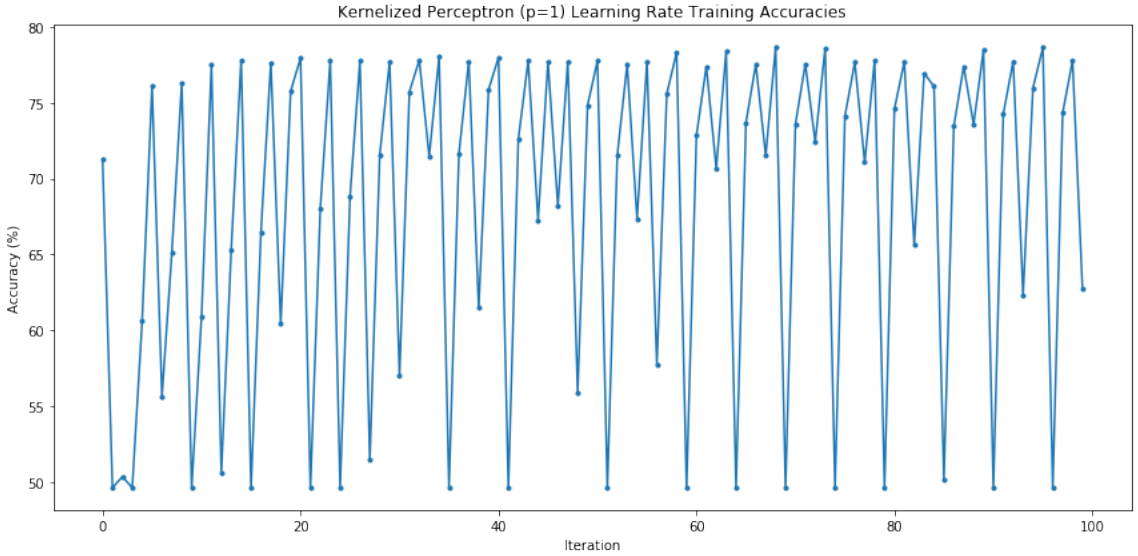
(b) Explain why changing the learning rate does not impact the learning trajectory and the learned decision boundary?

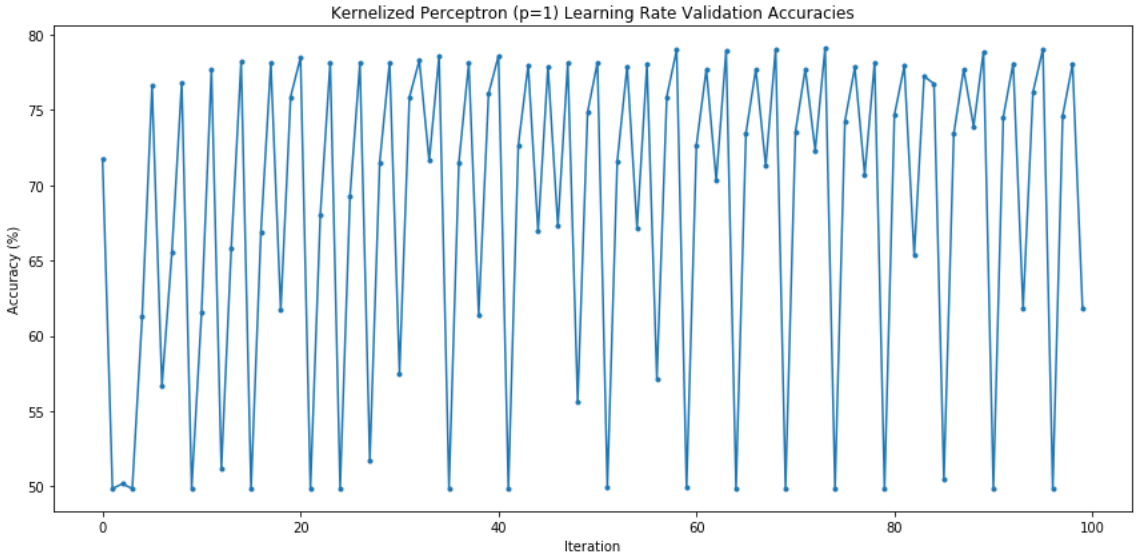
- Learning rate doesn't impact the learning trajectory because the learning rate is being added to the alphas as a constant number. Adding a fixed constant number, however large it may be, will give the same impact for each alpha that it is added to.

Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

(c) Record and plot the training accuracy and validation accuracy as a function of the iterations. Comparing these curves with the ones acquired with the same p value in part 2a, what do you observe? What are your explanations for the observation?

- Compared to the online algorithm, the batch algorithm is very unstable and its performance for both training and validation are significantly noisy.
- Because batch algorithm computes the predictions before updating the alphas, it results to creating predictions with an old model which likely explains its noisy performance.





Francis L. Bermillo
bermillf@oregonstate.edu
Implementation Assignment 3: Perceptron and Kernels
CS534 Machine Learning

(d) What is the asymptotic runtime of your algorithm in terms of the number of training examples n? Try to make your implementation as efficient as possible. Plot the empirical runtime of your algorithm as a function of the iterations. How does it compare to the runtime of the online algorithm?

- The asymptotic runtime of the algorithm in terms of examples n would be **O(maxiter)**
- After optimizing the batch algorithm, each iteration took an average of 7 ms and took a total runtime of 6 s for 100 iterations
- Compared to the online algorithm, batch algorithm ran significantly faster. This makes sense since the batch algorithm computes all the predictions at once before calculating the alphas, eliminating the for loop which slows down the algorithm.