CSCE 421 HW 4

Arya Rahmanian
Department of Computer Science
Texas A&M University
College Station
aryarahmanian@tamu.edu

March 30th, 2023

Problem 2

Starting from the two-class perceptron cost, I'll expand it to show how it equals the multi class perceptron cost

$$g(\mathbf{w}) = \frac{1}{P}\sum_{p=1}^{P} \max\left(0, -y_p \dot{\mathbf{x}}_p^T \mathbf{w}\right)$$

$$= \frac{1}{P}\sum_{p=1}^{P}\left[\left(\max_{j=0}^{1} y_p \dot{\mathbf{x}}_p^T \mathbf{w}_j\right) - y_p \dot{\mathbf{x}}_p^T \mathbf{w}_{y_p}\right] \qquad (where C = 2)$$

$$= \frac{1}{P}\sum_{p=1}^{P}\left[\left(\max_{j=0}^{1} \dot{\mathbf{x}}_p^T \mathbf{w}_j\right) - \dot{\mathbf{x}}_p^T \mathbf{w}_{y_p}\right]$$

$$= \frac{1}{P}\sum_{p=1}^{P}\left[\left(\max_{j=0}^{1} \dot{\mathbf{x}}_p^T \mathbf{w}_j\right) - \sum_{j=1}^{2} I(y_p = j)\dot{\mathbf{x}}_p^T \mathbf{w}_j\right]$$

$$= \frac{1}{P}\sum_{p=1}^{P}\left[\left(\max_{j=0}^{1} \dot{\mathbf{x}}_p^T \mathbf{w}_j\right) - \max_{k=0}^{1} \dot{\mathbf{x}}_p^T \mathbf{w}_k I(y_p = k)\right]$$

$$g\left(\mathbf{w}0, \ldots, \mathbf{w}\mathrm{C} - 1\right) = \frac{1}{P}\sum_{p=1}^{P}\left[\left(\max_{j=0}^{1} \dot{\mathbf{x}}_p^T \mathbf{w}_j\right) - \dot{\mathbf{x}}p^T \mathbf{w}y_p\right]$$

To summarize, we've shown that the two-class perceptron cost can be written as a special case of the multiclass perceptron cost, where C = 2 and $\mathbf{w}_0$ and $\mathbf{w}_1$

are used to represent the weights for the two classes. This allows us to use the same algorithm for both binary and multiclass classification tasks.

Problem 3-b Hyperparameter Tuning

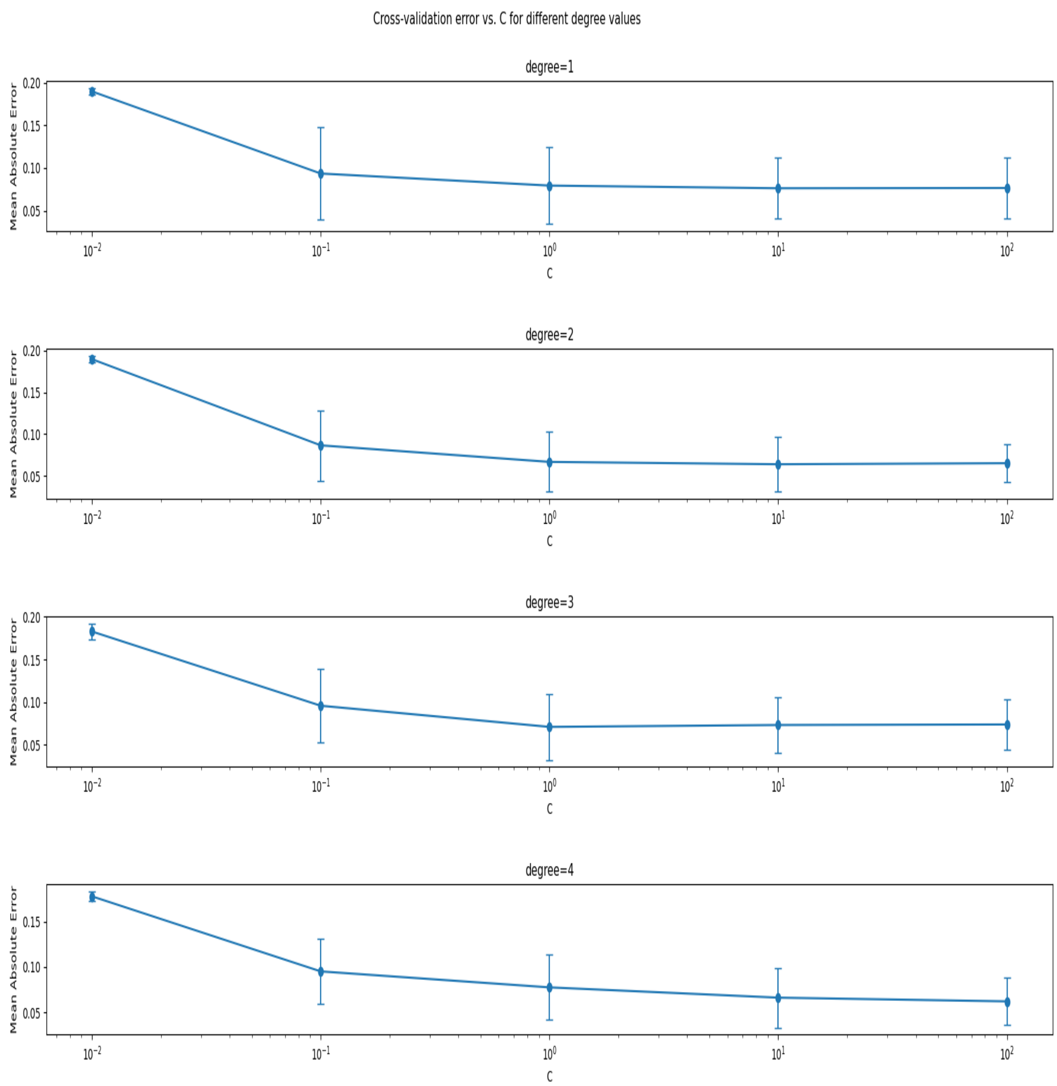

Figure 1: The figure above is the error vs C for the 4 different dimensions tested

Problem 3-c (Model Training and Testing)

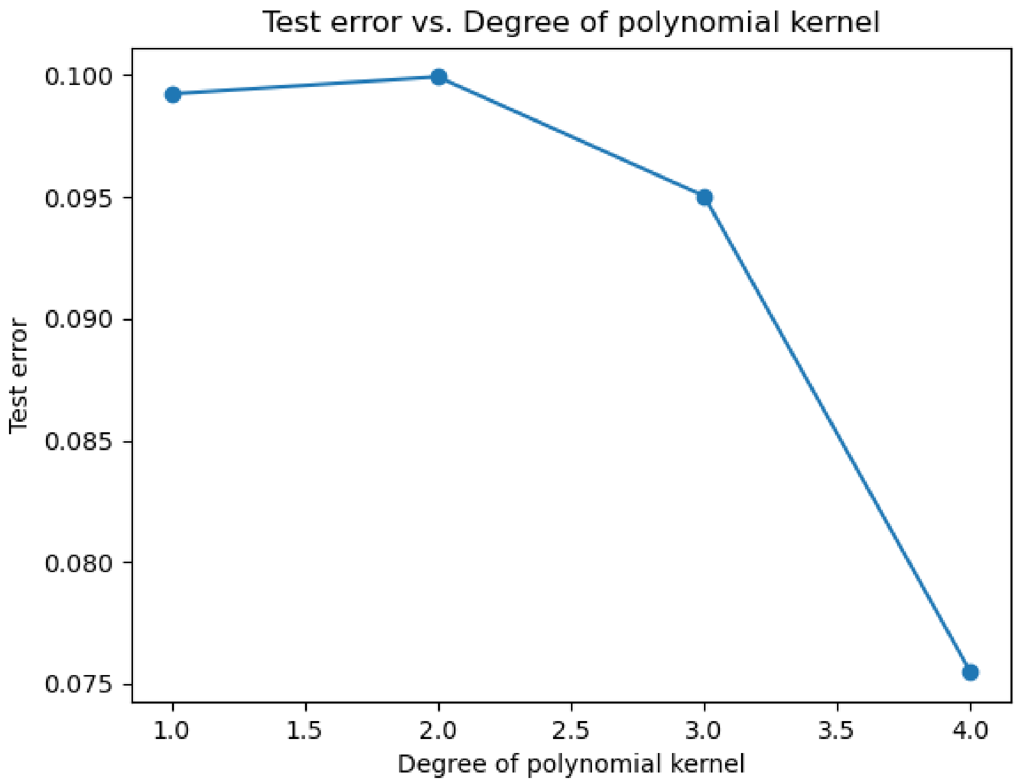

Figure 2: The figure above is the error vs degree of the kernel

Problem 3-d (Results Evaluation)

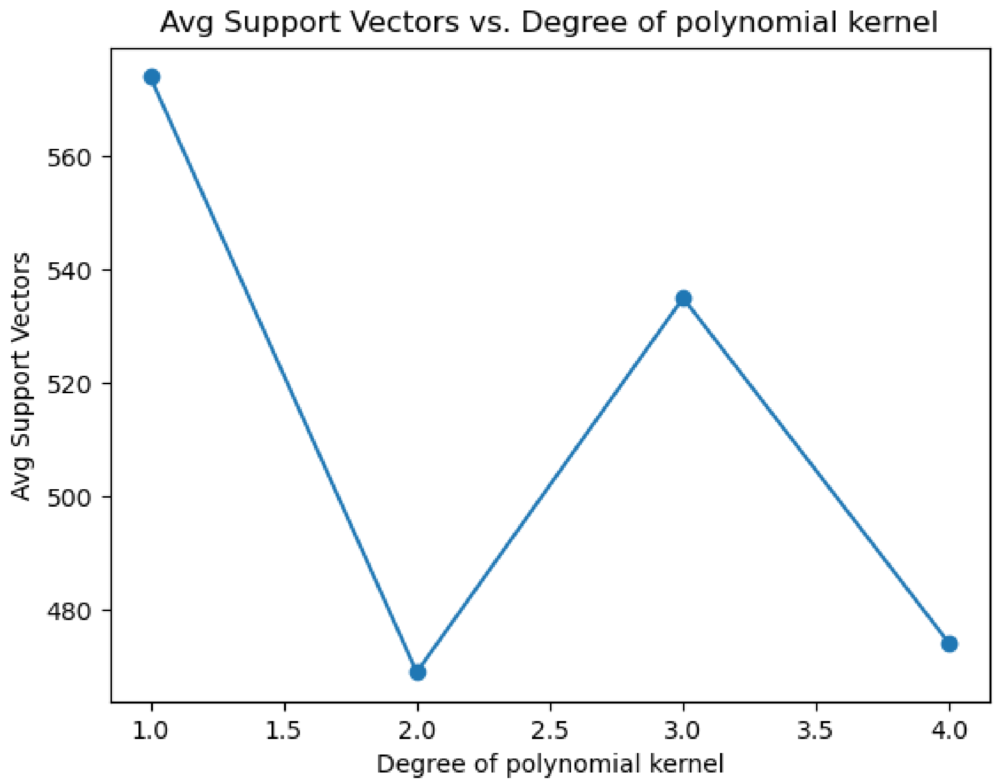

Figure 3: The figure above is the average number of support vectors in each degree of the kernel

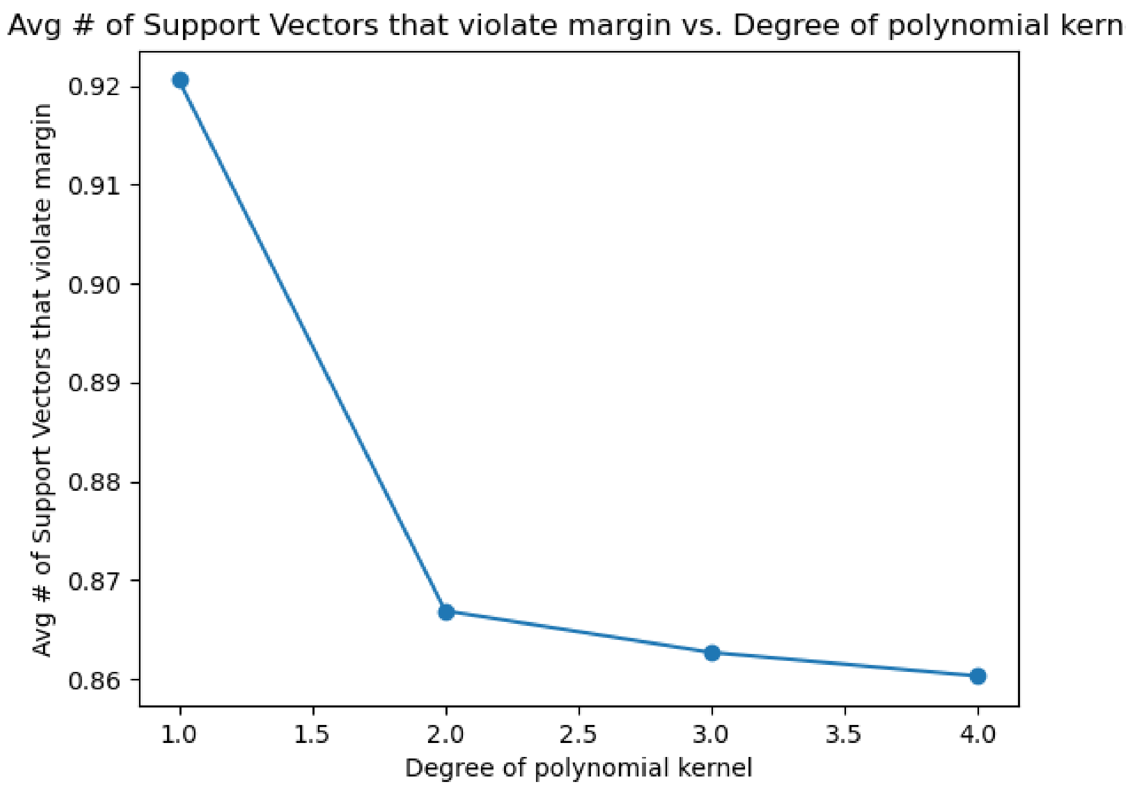

Figure 4: The figure above is the average number of support vectors that violate the margin in each degree of the kernel

To figure out how many support vectors lie on the margin, I'm going to assume since I have a low amount of support vectors that violate the margin > 1, that the number of support vectors on the margin is equal to what you see in figure 3.

Problem 3-e (Conceptual)

1

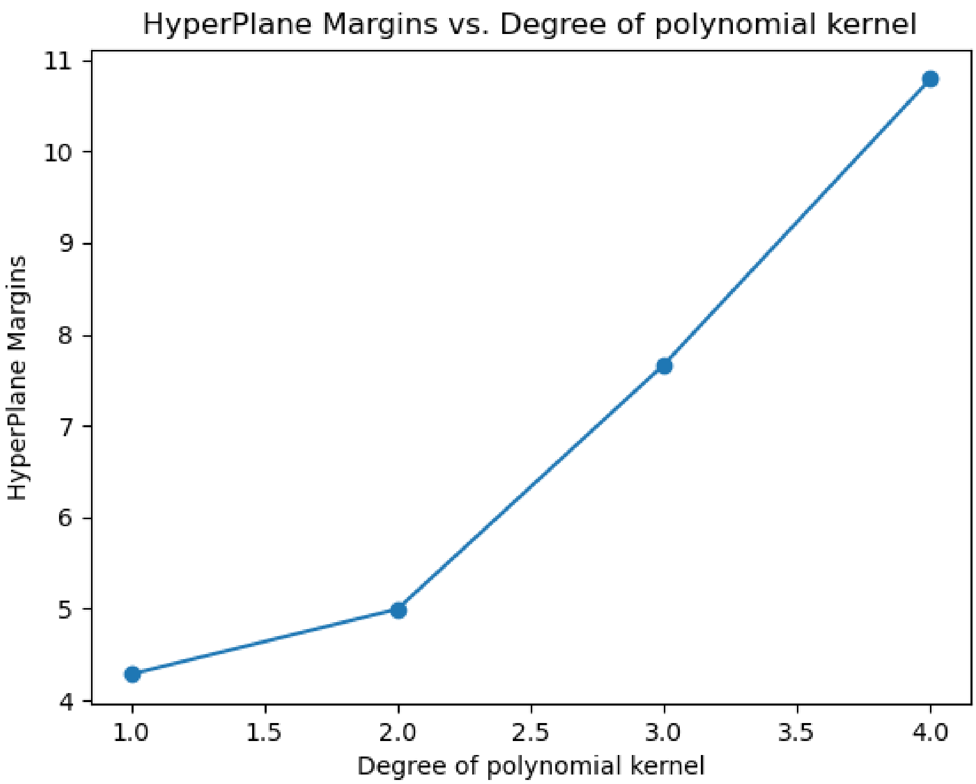

Figure 5: The figure above is the average hyperplane margin in each degree of the kernel

By observing my plots provided, a higher dimension led to less support vectors. This could be because increasing the degree of the polynomial kernel can lead to a more complex decision boundary that better fits the training data, which in turn may result in fewer support vectors being required to define that

boundary.
Additionally, increasing the degree led to larger margins, this is due to that the data may fit to a higher dimension (4th) better than 1, 2, or 3.

2

In an SVM model using an RBF kernel, the parameter that influences the model fit is the gamma parameter. Gamma controls the smoothness of the decision boundary, with smaller values leading to smoother boundaries and larger values leading to more complex and tighter boundaries.

A high gamma value tends to over-fit the training data and can result in poor generalization, while a low gamma value tends to under-fit the training data and can result in poor accuracy on the training data. The choice of gamma should be based on the specific problem and the available data.