Dual Formulation of a Nonlinear MMSE Classifier

Keith H. Bova

***Abstract*—This paper presents the kernel technique as a method of non-linear classification. Similarities and differences between the kernel technique and the Volterra approach are discussed. Source code to reproduce the experiments is given at the end of the paper.**

***Index Terms*—Kernel technique, regression, vector classifier**

I. Introduction

THE kernel method of non-linear classification uses higher-order polynomials to make a linear combinations of data. The kernel method helps to alleviate the curse of dimensionality we face with volterra expansion.

A. Ploynomial Expansion

We represent the kernel as:

$$\underline{\underline{\mathbf{K}}} = (\underline{\underline{\mathbf{X}}}_i^T \underline{\underline{\mathbf{X}}}_j + 1)^3 \tag{1}$$

We represent the weights as:

$$\underline{\underline{\alpha}} = \underline{y}(\underline{\underline{\mathbf{K}}} + \gamma \underline{\underline{\mathbf{I}}})^{-1} \tag{2}$$

We represent the testing data as a vector:

$$\underline{y}_{test} = \underline{\underline{\mathbf{K}}}_{test}^T \underline{\underline{\alpha}} \tag{3}$$

Where the kernel testing matrix is represented as:

$$\underline{\underline{\mathbf{K}}}_{test}^T = (\underline{\underline{\mathbf{X}}}_{test}^T \underline{\underline{\mathbf{X}}}_{train} + 1)^3 \tag{4}$$

II. Experiments

A. Classifying With Low Values of α

Similar to the previous two experiments using a Volterra series, in the following two experiments, we generate data with the same methods. First, we start with a distribution that follows a low convolutional spread, where:

$$\alpha = 0.2, N = 100 \tag{5}$$

A plot of the data is shown below.

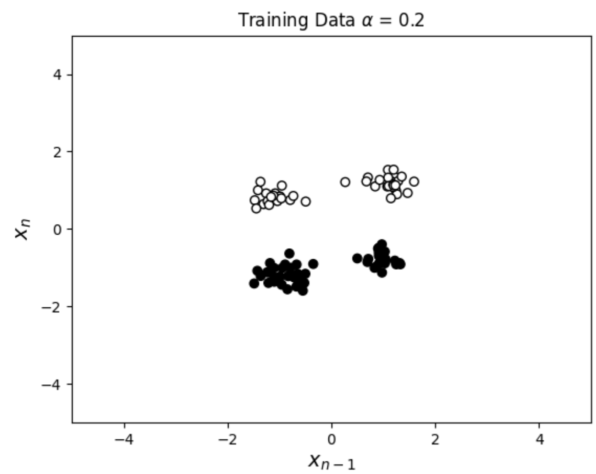

Fig. 1. Data clusters in $\mathbb{R}^2$ using $\alpha = 0.2, N = 100$

Again, we see that this distribution is linearly separable. After passing the data through the kernel classifier, we are met with the following results.

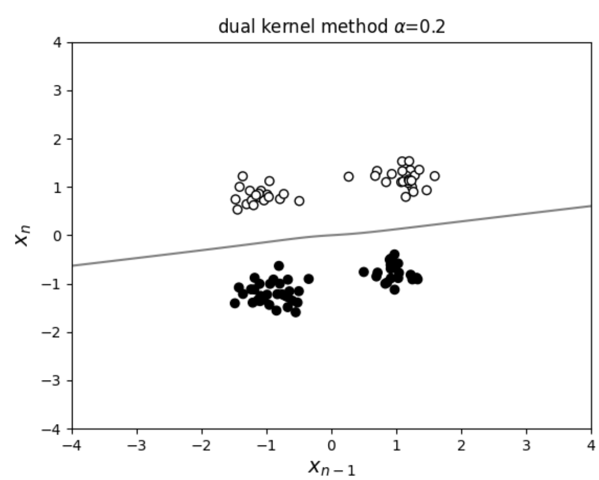

Fig. 2. Example in $\mathbb{R}^2$ classifying linearly separable data using the kernel method

We see that the contour is *almost* linear–but not quite.

B. Classifying With High Values of α

For this experiment, we used the same function to generate our data; however, as with the Volterra series, we vary the parameters such that:

$$\alpha = 1.5, N = 100 \quad (6)$$

Once again, we are met with a distribution that is no longer linearly separable with zero errors.

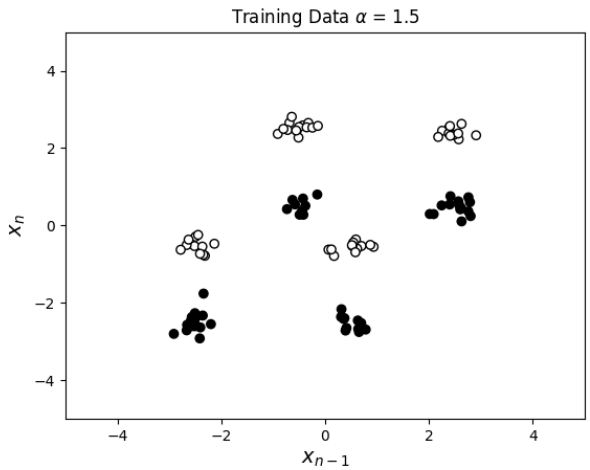

Fig. 3. Data clusters in $\mathbb{R}^2$ using $\alpha = 1.5, N = 100$

After passing this distribution into the third order polynomial kernel, we see a contour that is almost identical to the one presented with the Volterra series.

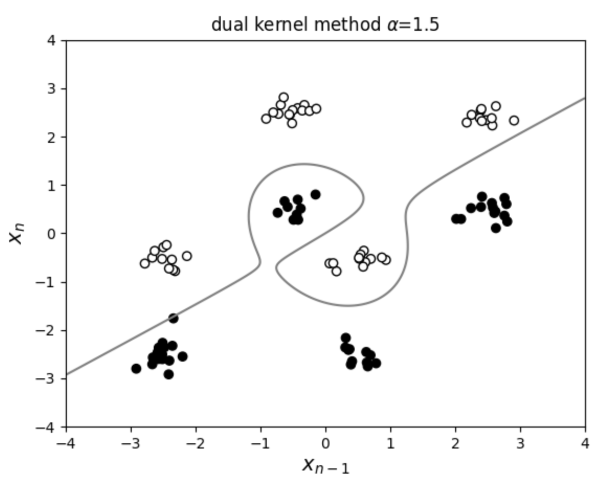

Fig. 4. Applying the kernel technique to our data $\alpha = 1.5, N = 100$

III. Conclusion

The third order polynomial kernel shows a very similar contour to the volterra expansion in $\mathbb{R}^{10}$. Both the volterra expansion and the polynomial kernel underfit; however, the polynomial kernel features a lower computational complexity and doesn't suffer from the curse of dimensionality in the same way that we see in the volterra series.

IV. Source Code

https://github.com/keithhbova/support_vector_machines/