Implementing LVQ2 and SVM Networks

Bilal Ahmad

November 5, 2019

1 Problem

1.1 Applying SVM and LVQ2 Networks to a Dataset

For this exercise we are using support vector machines (SVM) and learning vector quantization (LVQ) to perform classifications on a dataset. These models work by mapping the input dataset onto a feature space and using the geometry of the distribution to determine which class an input vector corresponds to. These algorithms are widely used in industrial applications and often outperform other algorithms on computer vision tasks.

1.2 Dataset

For this exercise I am using a database of divorce information [1]. The database has 54 input parameters and two classifications. The input parameters describe information about a couple (how well spouses know each other, do conversation ever become condescending, etc) and the classifications are whether the couple divorced or not. The data originally contained enumerations for some questions but I converted and normalized those data inputs. I used the first 75% of the data to train the networks and the remaining 25% to test their performance.

1.3 Implementing The SVM and LVQ2

For the SVM and LVQ2 implementation, I used built-in libraries available in neupy and scikit [2] [3]. For the back propagation network, I implemented the one used in previous exercises [4]. The network used sigmoid activation functions and uses a constant learning rate of .15. For the full implementation of all the algorithms, see the Appendix to this document.

2 Results

After training the data, I ran each network with the test data and determined how many correct classifications were made for each classification type. The results of the trials are seen in Table 1. From Table 1 it is evident that the SVM and LVQ2 implementations were more accurate on the test data - the LVQ2 implementation got every single classification correct. I adjusted different parameters for the LVQ2 and SVM but still ended up with correct classifications rate near 1.0. I think this is due to this specific dataset being highly separable between classes. I tried running the same tests with only half the input parameters and still received similar results for the SVM and LVQ2 while the back propagation implementation suffered in performance.

	Back Propagation	**SVM**	**LVQ2**
Class 1	.92	1.0	1.0
Class 2	.94	.98	1.0

Table 1: Correct Classification Ratio for Each Network

3 Conclusion

The SVM and LVQ2 algorithms performed better than the back propagation network. For further study I would used a dataset that is less separable and see how the algorithms handle data inputs existing in overlapping vector spaces. I think I would also spend some time implementing these networks from scratch. Because the LVQ2 and SVM implementations use different libraries, it is difficult to tell if there may be some lurking factors that make their comparison less valid.

4 References

[1] Yöntem, M , Adem, K , İlhan, T , Kılıçarslan, S. (2019). DIVORCE PREDICTION USING CORRELATION BASED FEATURE SELECTION AND ARTIFICIAL NEURAL NETWORKS. Nevşehir Hacı Bektaş Veli University SBE Dergisi, 9 (1), 259-273.

[2] Malik, Usman - Implementing SVM and Kernel SVM with Python's Sckikit-Learn https://stackabuse.com/implementing-svm-and-kernel-svm-with-pythons-scikit-learn/

[3] Shevchuk - Yurii, Neural Networks in Python http://neupy.com/apidocs/neupy.algorithms.competitive.lvq.html

[4] Brownlee, Jason - How to Code a Neural Network with Backpropagation In Python
https://machinelearningmastery.com/implement-backpropagation-algorithm-scratch-python/