Implementing an SVM
A shot in the dark

Daniel Hanson, Sam Kreter, Brendan Marsh, and Christina R.S. Mosnick

***Abstract*—The abstract goes here.**

I. Introduction

OUR introduction paragraph goes here. This is just some sample text to fill up the space.

A. Subsection #1

Subsection text here. Sample text and stuff goes here.

II. Implementation

Implementation paragraph text goes here.

III. Experiments

After the SVMs were all trained using the bootstrapping method, we used a committee-waterfall approach to determine the best class for each test point. In order to do this, the SVMs are grouped by classifier, with 7 independently trained SVMs per each of the 8 classifiers. Each test point is run through each of the 7*8=56 SVMs. When committee results are gathered, if the point has less than 4 committee votes for each classifier, it is unclassified. If the point has 4 or more votes from just one classifier group, it is classified to that group. If the point has 4 or more votes from multiple classification committees, it is classified to the committee with the most votes, or in the event of a tie, to a random choice between the tie.

IV. Conclusion

Conclusion paragraph text goes here.

Acknowledgment

Christina would like to thank her mom for her support.

References

[1] Tulloch, Andrew. *A Basic Soft-Margin Kernel SVM Implementation In Python* Tullo.ch. N.p., 2013. Web. 24 Mar. 2016.
[2] *Quadratic Programming With Python And CVXOPT. N.p., 2016. Web. 24 Mar. 2016.*
[3] *How To Calculate A Gaussian Kernel Effectively In Numpy.* Stats.stackexchange.com. N.p., 2016. Web. 24 Mar. 2016.
[4] *Scipy.Spatial.Distance.Pdist Scipy V0.17.0 Reference Guide.* Docs.scipy.org. N.p., 2016. Web. 24 Mar. 2016.