

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

A. Dual Representation (from Eqn. 7.2 [1])

$$\mathbf{L} = \sum_N a_n - \frac{1}{2} \sum_n \sum_m a_n a_m t_n t_m \mathbf{K} \quad (1)$$

B. Quadratic Programming Problem [3]

$$\min_{\mathbf{x}} \frac{1}{2} \mathbf{x}^T \mathbf{P} \mathbf{x} - \mathbf{Q}^T \mathbf{x} \quad (2)$$

C. Parameters for Quadratic Programming

$$\mathbf{P} = \sum_n \sum_m t_n t_m \mathbf{K} \quad (3)$$

$$\mathbf{Q} = \begin{pmatrix} -1 & -1 & -1 \\ -1 & -1 & -1 \\ -1 & -1 & -1 \end{pmatrix} \quad (4)$$

- Where \mathbf{Q} 's dimensions are determined by number of samples

D. Constraints

$$\mathbf{G} \mathbf{x} \preceq \mathbf{h} \quad (5)$$

$$a_n \geq 0 \rightarrow -a_n \leq 0 \quad (6)$$

$$a_n \leq \mathbf{C}$$

$$\text{std} \mathbf{G} \begin{pmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{pmatrix} \quad (7)$$

$$\text{std} \mathbf{H} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (9)$$

$$\text{slack} \mathbf{G} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad (10)$$

$$\text{slack} \mathbf{H} \begin{pmatrix} \mathbf{C} & \mathbf{C} & \mathbf{C} \\ \mathbf{C} & \mathbf{C} & \mathbf{C} \\ \mathbf{C} & \mathbf{C} & \mathbf{C} \end{pmatrix} \quad (11)$$

$$\mathbf{A} \mathbf{x} \leq \mathbf{b} \quad (12)$$

$$\sum_{n=1}^N a_n t_n = 0 \quad (13)$$

$$\mathbf{A} = \mathbf{y} \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix} \quad (14)$$

$$\mathbf{B} = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \quad (15)$$

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] Bishop, Christopher M. *Pattern Recognition And Machine Learning*. New York: Springer, 2006. Print.
- [2] Tulloch, Andrew. *A Basic Soft-Margin Kernel SVM Implementation In Python*. Tullo.ch. N.p., 2013. Web. 24 Mar. 2016.
- [3] *Quadratic Programming With Python And CVXOPT*. N.p., 2016. Web. 24 Mar. 2016.
- [4] *How To Calculate A Gaussian Kernel Effectively In Numpy*. Stats.stackexchange.com. N.p., 2016. Web. 24 Mar. 2016.
- [5] *Scipy.Spatial.Distance.Pdist Scipy V0.17.0 Reference Guide*. Docs.scipy.org. N.p., 2016. Web. 24 Mar. 2016.