Exploring the SVM-KNN

Joon Hyuck Choi (jchoi100, jchoi100@jhu.edu), Joo Chang Lee (jlee381, jlee381@jhu.edu)

November 4, 2016

1 Abstract

The k-Nearest Neighbor (KNN) algorithm is a non-parametric method used for classification and regression. In a previous homework assignment, we implemented the standard KNN classifier and one of its variants, the distance weighted KNN. We felt the need to take another step from this assignment and explore another type of the KNN algorithm, which is a bit more invovled.

The KNN variant that we explore in this work is the SVM-KNN, a KNN algorithm that makes use of a kernel multiclass SVM as a subprocedure. The algorithm first attempts multiclass classification using the standard KNN with a unanimous voting scheme. When at least one vote is different, it turns to the kernel mutliclass SVM for prediction.

We use SVM in hopes of addressing issues with majority voting where a dominant label over the samples can prevent prediction of the labels with less frequency[2]. Using a SVM can be effective in the neighborhood of a small number of examples and a small number of classes[1].

2 Methods

Solve the dual problem for optimization in SVM. In order to be able to extend our algorithm to support non-linear data, we are planning on implementing a kernel SVM.

Basic Algorithm[1]

For input instance,

(1) Compute distances of the instance to all training instances and pick K nearest neighbors.

(2) If the K neighbors have the same labels, the instance is labeled and terminate.
Else, compute the pairwise distances between the K neighbors.

(3) Convert the distance matrix to a kernel matrix and apply multiclass SVM.

(4) Use the resulting classifier to label the input instance.

The KNN algorithm often suffers from the problem of high variance in the case of limited sampling. Therefore, SVMs might help alleviate the high variance problem although it might consume more computational time. However, given a small neighborhood of instances and a small number of classes, SVMs may give better performance than other types of classifiers.

Loss Function

$\frac{1}{n}\Sigma_{i=1}^{n}(1 - y_i[w \cdot x_i]^+) + \lambda\frac{1}{2}||w||^2$ (hinge loss)

Objective Function

$\max \Sigma_{i=1}^{n}\alpha_i - \frac{1}{2}\Sigma_{i=1}^{n}\Sigma_{j=1}^{n}y_i y_j \alpha_i \alpha_j (x_i x_j^T)$

s.t. $\alpha_i \geq 0$ and $\Sigma_{i=1}^{n}\alpha_i y_i = 0$ (dual form)

3 Resources

(1) **External Libraries:** SVM algorithm (e.g. scikitlearn[3])

(2) **Dataset:** Use the dataset given for homework assignments. (e.g. vision, nlp, etc.)

(3) **Programming Language:** Python 2.7

4 Milestones

4.1 Must achieve

(1) Implement standard KNN.

(2) Incorporate SVM into KNN from Step 1 by using external library for SVM.

(3) Analysis and comparison of SVM-KNN with standard KNN.

4.2 Expected to achieve

(1) Implement linear SVM on our own.

(2) Extend linear SVM from above to kernel SVM.

4.3 Would like to achieve

(1) Extend kernal SVM from 4.2 to support multiclass classification.

(2) Explore other simpler variants of KNN. (e.g. ϵ-ball KNN)

5 Final Writeup

(a) Motivation for exploring SVM-KNN

(b) Methods (dataset, algorithm)

(c) Accuracy / Running time comparison (vs. KNN algorithms implemented in HW3)

(d) Final Discussion

6 Bibliography

(1) H. Zhang, A. Berg, M. Maire, and J. Malik. ″SVM-KNN: Discriminative Nearest Neighbor Classification for Visual Category Recognition″, CVPR, 2006.

(2) D. Coomans, D. L. Massart. ″Alternative k-nearest neighbour rules in supervised pattern recognition: Part 1. k-Nearest neighbour classification by using alternative voting rules″, *Analytica Chimica Acta*. 136: 15-27.

(3) http://scikit-learn.org/stable/modules/svm.html