Amanda Solis

Abstract

The purpose of this paper was to explore and compare how Support Vector Machines, K Nearest Neighbors Classifiers, and Artificial Neural Networks perform classification on three different data sets.

Introduction

Supervised learning algorithms try to learn from labeled training data to predict or classify unlabeled test data. This paper explores three well-known supervised learning classifiers and their ability to correctly classify new samples from three very different data sets. The classifiers used in this paper were K-Nearest Neighbors (KNN), Support Vector Machines (SVM), and Artificial Neural Networks (ANN). KNN is a non-parametric algorithm that classifies an incoming sample by assigning it to the class most common among its k nearest neighbors. SVMs are binary classifiers that represent the examples as points in space in such a way that the two classes are divided by as wide a margin as possible. When the data is not linearly separable, SVMs can use kernels to map the inputs into high-dimensional feature spaces and then fit the maximum-margin hyperplane in this transformed feature space. Finally, ANNs are nonlinear classifiers that learn internal representations of the input by in order to classify new samples. They are systems of units and connections that are modified during training in order to minimize the difference between the desired and actual output.

Data

I initially tried working with the data sets used in Caruana 2006 and 2008 (like ADULT, COVTYPE, MNIST), however, training on such large data sets was not feasible given my computing power. The data sets I use in this paper are comparable in feature length to those used in Caruana but have fewer samples. Choosing to work with these other data sets gave me more opportunities to tune classifier parameters.

The Diabetic Retinopathy (DIAB) data set is a binary classification problem where 19 features are computed from images of patient's eyes who are either healthy or suffering from diabetic retinopathy. The original data set consists of 1151 samples, of which 921 (80%) are used for training models and 230 (20%) for testing. The Isolet (ISOLET) is a multiclass classification problem that contains processed recordings of subjects saying any one letter of the alphabet, with each sample containing 617 features and corresponding to 1 of the 26 letters. The training set contained 4990 (80%) samples, while the testing set contained 1248 (20%) samples. The final data set used was an EEG data set where subjects were recorded performing left-hand imagery, right-hand imagery, or speaking a random word. Each subject's data was classified separately, with each file containing about 3500 samples and 96 features corresponding to power spectral density estimates of 8 EEG electrodes in 12 different subbands between 8-30 Hz. Each subject's data was split into about 2800 (80%) training samples and 700 (20%) testing samples.

Data Set Information

	Isolet	Diab	EEG Subject 1	EEG Subject 2	EEG Subject 3
Dimensions	6238x617	1151X19	~3500x96	~3500x96	~3500x96
Classes	26	2	3	3	3

Methods

Each data set was normalized so that values for all features were in the range of 0 and 1. Each data set was randomly shuffled, and split into two sets: 80% for training and 20% for testing. 10-Fold Cross Validation was used for training all classifiers and for finding the best hyperparameters.

For classifying with SVM, I decided used the following four kernels: linear, polynomial (degrees 2 and 3), and radial basis function. I searched for the best cost value (cost=0.5, 1, 2, 4, 8, 16) for linear kernels, and performed a grid search for the best gamma (gamma=0.125, 0.25, 0.5, 1, 2, 4, 8) and cost values for the remaining kernels. I used these kernels because they are the most commonly used and consistently produce good results (Caruana 2006).

$$\text{linear: } K(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i^T \mathbf{x}_j.$$

$$\text{polynomial: } K(\mathbf{x}_i, \mathbf{x}_j) = (\gamma {\mathbf{x}_i}^T \mathbf{x}_j + r)^d, \gamma > 0.$$

$$\text{radial basis function (RBF): } K(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|^2), \gamma > 0.$$

For classifying with KNN, I searched for the best value of K between the values of 1 and 20 using an inverse weighting method (1/distance). I used the inverse weighting scheme because intuitively it seemed like it would provide more robust results as each neighbor has less influence the further it is from the training sample. I also experimented with different distance metrics to see how each performed, the distance metrics used were: euclidean, chebychev, and mahalanobis:

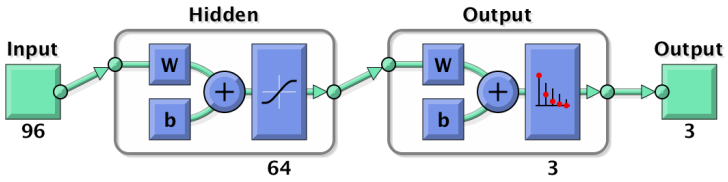
Euclidean $$d\,(x\,y) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

Chebychev $$d\,(x\,y) = \max(|x_1 - x_2|, |y_1 - y_2|)$$

Mahalanobis $$D_M(\underline{x}) = \sqrt{(\underline{x} - \underline{\mu})^T S^{-1} (\underline{x} - \underline{\mu})}.$$

Finally, for ANN, I trained the networks using gradient descent with varying values of momentum and hidden units. For momentum I searched for the best value of momentum=0.1 0.3 0.6 0.9 and for the number hidden units I searched for the best value of hidden units=1, 2, 4, 8, 16, 32, 64, 128, 256, 512 and trained the networks to minimize the cross entropy error. I used the TanSig activation for the hidden layer which squashes the values between -1 and 1 and a softmax activation on the output function

which outputs the probabilities of the training vector being in each class and summing up to 1. A view of my network structure can be seen below:



Results

	Isolet	Diab	EEG Subject 1	EEG Subject 2	EEG Subject 3
SVM Linear	**100%** **Best: C= 1.0**	69.1304% Best: C= 16.0	**87.4525%** **Best: C= 0.5**	**86.5385%** **Best: C= 2.0**	78.2101% Best: C= 1.0
SVM Poly D2	96.7147% Best: C= 0.5, G=0.125	**76.9565%** **Best: C= 16.0, G=2.0**	79.4677% Best: C= 16.0, G=0.125	78.0769% Best: C= 16.0, G=8.0	64.5914% Best: C= 16.0, G=0.25
SVM Poly D3	96.7147% Best: C= 0.5, G=0.125	76.087% Best: C= 8.0, G=2.0	81.7490% Best: C= 4.0, G=0.25	78.8462% Best: C= 8.0, G=0.25	65.7588% Best: C= 8.0, G=0.25
SVM RBF	90.8654% Best: C= 2.0, G=8.0	74.7826% Best: C= 8.0, G=0.5	83.6502% Best: C= 16.0, G=1.0	79.2308% Best: C= 16.0, G=1.0	69.6498% Best: C= 16.0, G=2.0
KNN-Euclidean	90.62% # Neighbors: 13	72.61% # Neighbors: 18	84.03% # Neighbors: 1	81.54% # Neighbors: 1	83.27% # Neighbors: 1
KNN-Chebychev	73.56% # Neighbors: 4	68.70% # Neighbors: 12	69.96% # Neighbors: 7	56.92% # Neighbors: 2	52.92% # Neighbors: 2
KNN-Mahalanobis	39.26% # Neighbors: 1	63.91% # Neighbors: 6	78.71% # Neighbors: 2	82.31% # Neighbors: 2	**83.66%** **# Neighbors: 2**
ANN	88.402% Best: Momentum=0.9 # Hidden Units: 32	73.3615% Best: Momentum=0.9 # Hidden Units: 64	52.1082 % Best: Momentum=0.9 # Hidden Units: 64	84.8434% Best: Momentum=0.9 # Hidden Units: 256	81.4287% Best: Momentum=0.9 # Hidden Units: 256

Test Classification Accuracy and Best Parameters

Conclusions

From the results table above we can see that the SVM with a Linear kernel was the best classifier for three of the five data sets. Though not listed here, one interesting thing I noted while working with the SVMs was that a larger number of support vectors per class led to a lower test classification accuracy. Like most of the literature suggests, we see that the best hyperparameter value in SVMs tended to be small for gamma and a high value for momentum in ANNs (LIBSVM, Stackoverflow). All of the classifiers did fairly well except for the KNN using Chebychev distance which consistently had the lowest test classification accuracy. There was also a large difference in accuracy for some data sets using the KNN with Mahalanobis distance. This classifier performed particularly low on the Isolet database and took significantly longer to train. Interestingly, with the KNN classifier, the data sets tended to favor a lower number of neighbors, with the most commonly occurring best neighbor value=2.

References

Caruana, Rich, Alexandru Niculescu-Mizil. "An Empirical Comparison of Supervised Learning Algorithms." Proceedings of the international conference on Machine learning. ACM, 2006.

Caruana, Rich, Nikos Karampatziakis, and Ainur Yessenalina. "An empirical evaluation of supervised learning in high dimensions." Proceedings of the 25th international conference on Machine learning. ACM, 2008.

Cogs 118a, all lecture slides

MATLAB, **https://www.mathworks.com/**

LIBSVM, **https://www.csie.ntu.edu.tw/~cjlin/libsvm/**

Diabetic Retinopathy Data Set, **http://archive.ics.uci.edu/ml/datasets/Diabetic+Retinopathy+Debrecen+Data+Set#**

Balint Antal, Andras Hajdu: An ensemble-based system for automatic screening of diabetic retinopathy, Knowledge-Based Systems 60 (April 2014), 20-27.

The data set is based on features extracted from the Messidor image data set.

Isolet Data Set, **https://archive.ics.uci.edu/ml/datasets/ISOLET**

EEG Data Set, **http://www.bbci.de/competition/iii/**, Data Set V, Millán, J. del R.. On the need for on-line learning in brain-computer interfaces Proc. Int. Joint Conf. on Neural Networks., 2004.