**Project 4: Support Vector Machines (SVM) and Bayesian Classifiers for Gender Classification**

**CS479: Pattern Recognition**

**Terence Henriod**

**May 14, 2014**

**Introduction**

In this project, Support Vector Machines (SVMs) were used to demonstrate high accuracy gender classification on grayscale face images.

The data used was a set of 48x60 grayscale images provided by the instructor. There were 400 images in total, 200 male and 200 female. This set of images was subdivided to produce subsets of 134 training and 266 testing images, 3 times over (the proportions of male to female images in each subset were approximately equal, but not the same proportions each time).

To further improve classification, the PCA approach was applied (described below). Note: this transformation was applied prior to reception of the data set. Finally, the first 30 values of the image vectors projected onto the eigen-space were the features selected for the classification experiments.

SVM experiments were carried out using the code from the LIBSVM open-source SVM libarary. In order to use this, data had to be properly formatted and then scaled using the included LIBSVM tools.

A Bayesian classification approach was also used in order to benchmark the SVM approach’s performance. Then mean and covariance of the eigen-features were used for training the Bayes classifiers.

**Technical Discussion**

*Principal Component Analysis*

Principal Component Analysis (PCA) seeks to reduce the dimensionality of the data by finding a way to represent the data in a space that still preserves most of the information, yet requires far fewer dimensions to do this. The data may also be better represented since the new space

PCA accomplishes this by finding the unit-length eigenvectors of the covariance matrix of a data set, and then keeping a subset of the eigenvectors corresponding to a subset of the largest eigenvalues. When data vectors are projected onto the eigen-space, this linear combination of the subset of eigenvectors will preserve a large amount of the data, usually with far fewer features due to the fact that the eigenvectors are orthogonal to one another and are oriented along the directions in which the data varies the most.

*Support Vector Machines*

Support Vector Machines (SVM) are a non-statistical, discriminant method for classifying data. SVMs aim to represent data in a high enough dimensional space so that the classes are separable, and then construct a discriminant function (which represents a hyperplane) that has the maximum margin of separation between the data.

SVMs aim to minimize the empirical error (or training error) and minimize the VC dimension (or model complexity) through maximizing the width of the aforementioned margin. The width of this margin is determined by the support vectors the technique is named for. The support vectors represent the distances between the optimal hyperplane, represented by the discriminant function, and the “worst classified” data. The support vectors will represent the most difficult to classify patterns, but these patterns are the most important for the classification problem.

As mentioned previously, SVMs (typically) use a very high dimensionality space to achieve separability of the data. This would be troublesome for computations were it not for the “kernel trick.” The kernel trick allows SVM computations to be done using a kernel function that exists in the original space the data exists in, but to get the same result of the computation done in the high-dimensional space required for data separability (The computation being referred to is specifically a dot product computation). Commonly used kernels use a variety of classification functions such as polynomial, radial basis, and Gaussian functions.

*Bayesian Classification*

Bayesian classification relies on the assumption that features independently contribute to the overall probability that an object is in a certain class. It is common practice to assume that the data is Normally distributed when using Bayesian classification, although other distributions may be assumed/used. In our case, since the Gaussian data distributions were assumed, the discriminant functions used by the classifiers were of the form:

|  |  |
| --- | --- |
|  | [1] |

and the class with the least resulting discriminant value for an input vector was decided to be the most likely class for the data to belong to.

**Results**

*Experiment 1: SVMs*

The SVMs performed reasonably well, giving relatively high classification accuracies, although not as high as might be achieved using different software packages (I observed a graduate student achieve 97% accuracy using the WEKA package, anecdotally). However, given that the goal of the assignment was to successfully apply SVMs to the problem, and not to find the optimum parameter values for optimum SVM performance, slightly sub-optimal performance is acceptable. This being said, various parameter values were tested to find what might be considered “locally” optimal results, if not “globally” optimal results.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Polynomial Kernel SVM Performance  with C = , γ = | | | |  |
|  | Correctly Classified Images | Number of Tested Images | Accuracy (%) | Parameter Values C, γ Used |
| Data Fold 1 | 247 | 265 | 93.208 | 15, 0.8 |
| Data Fold 2 | 243 | 266 | 91.353 | 2, 0.2 |
| Data Fold 3 | 243 | 266 | 91.353 | 3, 0.2 |
| Average Accuracy | 733 | 798 | 91.855 |  |

Table 1: Accuracy results achieved using an SVM that utilized a polynomial kernel function

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| RBF Kernel SVM Performance  with C = 15 , γ = 0.2 | | | |  |
|  | Correctly Classified Images | Number of Tested Images | Accuracy (%) | Parameter Values C, γ Used |
| Data Fold 1 | 248 | 265 | 93.585 | 15, 0.2 |
| Data Fold 2 | 250 | 266 | 93.985 | 5, 0.1 |
| Data Fold 3 | 245 | 266 | 92.105 | 2, 0.2 |
| Average Accuracy | 743 | 798 | 93.108 |  |

Table 2: Accuracy results achieved using an SVM that utilized a Radial Basis Function (RBF) kernel

*Experiment 2: Bayesian Classifiers*

The Bayes classifier approach performed very well, better than might be expected considering my previous experiences with Bayes classifiers. The performance of the Bayes classifier was likely enhanced by the fact that the features used resulted from a PCA transformation of the data, so the features used were “independent” and very descriptive of the data.

|  |  |  |  |
| --- | --- | --- | --- |
| Bayesian Classifier Performance  (Assumed Gaussian Distribution, Equal Likelihood of Both Classes) | | | |
|  | Correctly Classified Images | Number of Tested Images | Accuracy (%) |
| Data Fold 1 | 245 | 265 | 92.453 |
| Data Fold 2 | 239 | 266 | 89.850 |
| Data Fold 3 | 238 | 266 | 89.474 |
| Average Accuracy | 722 | 798 | 90.476 |

Table 3: Accuracy Results using a two-class Bayes classification approach

*SVM vs. Bayes Classification*

Figure 1: A graphical comparison of the performance of classification methods used in the experiment.

The SVMs consistently outperformed the Bayesian classification, as expected. It was hoped that the SVMs would perform better than they did (as discussed previously), allowing the Bayesian method to perform more closely to the SVMs than expected.

Given the consistent outperformance of Bayesian classifiers, lack of formal parameter optimization, and the anecdotal observations of extremely high performance of SVMs using other software (namely WEKA), it is clear that the SVM method is superior to the Bayesian classification method for the purposes of gender classification with grayscale images, and likely other applications as well. However, the Bayesian classifier method was less technical and possibly less computationally intensive, so Bayesian classification may be desirable in applications accuracy needs to be sacrificed for performance.

**Appendix**

*Program Listings*

The LIBSVM library was central to the project.

Important citation information:

Chih-Chung Chang and Chih-Jen Lin, LIBSVM : a library for support vector machines.

ACM Transactions on Intelligent Systems and Technology, 2:27:1--27:27, 2011.

Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>

The student would like to thank the makers of the Eigen linear algebra C++ library for their open-source contribution to the world. Hours were saved in coding and debugging thanks to their contribution. Eigen can be found at <http://eigen.tuxfamily.org/>.

The project was coded in C++. This included the use of the C++ STL, particularly the vector template container.

The use of Linux bash scripts greatly streamlined the workflow of the project.

bayes\_classifier.cpp: The implementation of a naïve Gaussian Bayes Classifier with arbitrary dimension.

bayes\_classifier.h: The specification of a naïve Gaussian Bayes Classifier with arbitrary dimension.

data\_preparation\_script.sh: A script that used various terminal commands and program calls in order to take the data from its raw format separate from the data labels, to an appropriate format with data files neatly separated by fold set and with validation and test data merged to make larger test sets.

merge-4-libsvm.cpp: A program to take a raw data file and a label file and merge them into one file in LIBSVM readable format.

project\_4\_bayes\_classification\_driver.cpp: A driver program that conducted the training and testing of a two-class Bayesian classifier experiment.

project\_4\_script.sh: Orchestrated the entire experiment, including calling the data preparation script and conducting the 3 types of classification experiments with the 3 data sets and different parameter combinations for the SVM experiments.

project\_4\_svm\_classification\_driver.cpp: A program that conducted the training and testing of an SVM classifier experiment.

*Complete program listings are attached to this document in the above order.*