Classification with Dimensionality Reduction and Performance Evaluations

PROJECT 2

Joshua Chavarria | ECE471 | March 12, 2017
jchavarr@vols.utk.edu

Abstract:

This report covers the development and implementation of various methods to perform pattern classification. Methods of normalization, dimensionality reduction, and discriminant functions are all utilized in this project to determine the classification of the data which ultimately indicates whether a population of women of Prima Indian heritage have diabetes or not. The files provided for this experiment include a training and testing set that are each initially seven dimensional and are made up of relevant values for bmi, age, number of pregnancies, and more. The main objective of Project 2 is to familiarize oneself with the use and implementation of the previously mentioned methods to evaluate the performance of the training set. This project also demonstrated the implementations of dimensionality reductions and performance evaluations can be used to uncover more information and patterns from given data. From this experiment, a decent prediction function was found for the preprocessed data, transformed data, FLD data, and PCA data. The maximum accuracy from discriminant function case 3 from the first classifier averaged around 75%, while the other discriminant functions were around averaging around 80%. Overall, these numbers were indicative of a decent decision rule.

Introduction:

Pattern Recognition is known primarily as a very important field in computer science, and one that is likely to lead to implementation of more advanced technology in the future. Pattern recognition is a broad field that encompasses very many studies. Its main concern, as its title suggests, is recognizing patterns, especially those used in innovating sound and visual patter recognition.

In a broad sense, pattern recognition is a branch of machine learning that analyzes regularities in data. **[1]** "Pattern Recognition is a subject researching object description and classification method, it is also a collection of mathematical, statistical, heuristic and inductive techniques of fundamental role in executing the tasks like human being on computers". In many cases, pattern recognition systems are trained using one set of data referred to as the training set, and are tested with another set of data typically labeled as the testing set for accuracy. The use of this labeled training data is known as "supervised learning" in which the system is tasked with using this data to make inferences and comparing its results with that of the desired output values. Modern day applications of pattern recognition are numerous and cover a broad scope of fields and activities, as previously mentioned. For example, pattern recognition over the years has been heavily worked on in order to innovate in the creation of speech recognition software. The voice assistant integrated in most modern phones would not be possible without the ability to convert sound patterns to text. As one might guess, pattern recognition systems drive the world around us in ways that we take for granted.

Objective:

Given a training and testing set of values, a decision rule must be generated to predict an ideal testing set for maximum accuracy. Each of these data sets are classified as either 0, representing that a patient does not have diabetes, and 1, representing a patient does have diabetes. Ultimately calculations will be made to find the most efficient testing set.

Background:

Dimensionality reduction is an essential tool for classifying data sets of high dimension and converting them into a space of fewer dimensions. Two methods of dimensional reduction were used in this experiment and the first of those technique is known as principal component analysis, which is thought to be the main linear technique for reducing data dimensions. This technique performs a linear mapping of data into a lower dimensional space by maximizing variance and finding components that are useful for representing the data. To do this, first the covariance matrix of the data must be found along with the eigen vectors of the data set. These two results are then used to reconstruct a fraction of the variance of the original data which in turn reduces the original data space. In short, principal component analysis (PCA) performs reduction while preserving as much of the variance of the higher dimensional space as possible. Without loss of generality, the eigen vectors are sorted in terms of their eigenvalues. This is used to find the principal component of the data.

The second method used to carry out the experiment is known as Fisher's Linear Discriminant (FLD). This method is supervised and compared to PCA, performs dimensionality reduction while preserving as much of the class discriminatory information as possible. These two methods are usually performed prior to applying K-Nearest Neighbors algorithm (kNN) to avoid the effects of what is known as the "curse of dimensionality" which simply refers to a notorious phenomenon that arises when analyzing and organizing high dimensional spaces that don't occur in low dimensional settings. The benefit of both PCA and FLD have is that they carry out both feature extraction and dimension reduction. The results of these methods will be shown further on in this report.

Covariance Matrix:

$$\Sigma = \begin{bmatrix} \sigma_{11} & \cdots & \sigma_{1d} \\ \vdots & \ddots & \vdots \\ \sigma_{d1} & \cdots & \sigma_{dd} \end{bmatrix} = \begin{bmatrix} \sigma_{1^2} & \cdots & \sigma_{1d} \\ \vdots & \ddots & \vdots \\ \sigma_{d1} & \cdots & \sigma_{d^2} \end{bmatrix}$$

$J(w) = (w^T * S_B * w)/(w^T * S_W * w)$ where S_B is the between classes scatter matrix and S_W is the "within classes scatter matrix. Because scatter matrices are proportional to the covariance matrices we could have defined J using covariance matrices the

proportionality constant would have no effect on the solution. The definitions of the scatter matrices are:

$$S_B = Xc\ (\mu c - x)(\mu c - x)^T$$

$$S_W = \sum c\ \ \sum i \in c\ (x_i - \mu_c\ )(x_i - \mu_c\ )^T$$

The test set is classified using the discriminant functions and its three cases as well as kNN. As recap, the discriminant has three main cases which are as follows:

- For the first case ($\sum_I$ =): The features are independent from each other which means that covariance and expected value will be equal regardless of feature.
- For the second case ($\sum_I$ =$\sum$): Features are equally distributed regardless of class but not independent of each other.
- For the third case ($\sum_I$ = $\sum$ is arbitrary): This means that neither features or classes are independent of each other and therefore leads to different covariance matrices for each.
- kNN decision rule tells us to look in a neighborhood of the unknown feature vector for k samples. If within that neighborhood more samples lie, we assign the unknown as belonging to class i:

$$P(\omega_m \mid x) = \frac{p(x \mid \omega_m) P(\omega_m)}{p(x)} = \frac{\frac{k_m}{n_m V}\frac{n_m}{n}}{\frac{k}{nV}} = \frac{k_m}{k}$$

Technical Approach:

Much like the previous project assignment the previously mentioned methods had to be implemented via C++ code. For the most part, I gathered what I had turned in for the last one and iterated on it to satisfy the requirements for this project. As always, before starting the implementation a few preprocessing steps had to be carried out including the normalization of the data set. This was done easily by finding the mean and sigma values for the training set first, and using those values later to normalize the testing set. The next tasks included the implementation of PCA on the dataset as well as the FLD dimensional reduction methods. These steps included calculations to find the covariance matrix, eigen values/vectors, and projection vector for the data plots. Lastly, I used the discriminant function as well as kNN to classify the test set. All four classifiers were compared using the prior probability base on the training set.

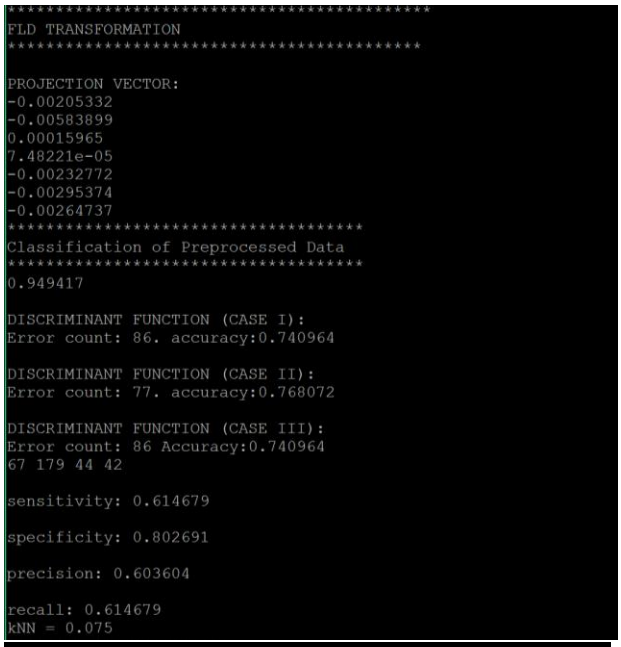
Experiments and Results:

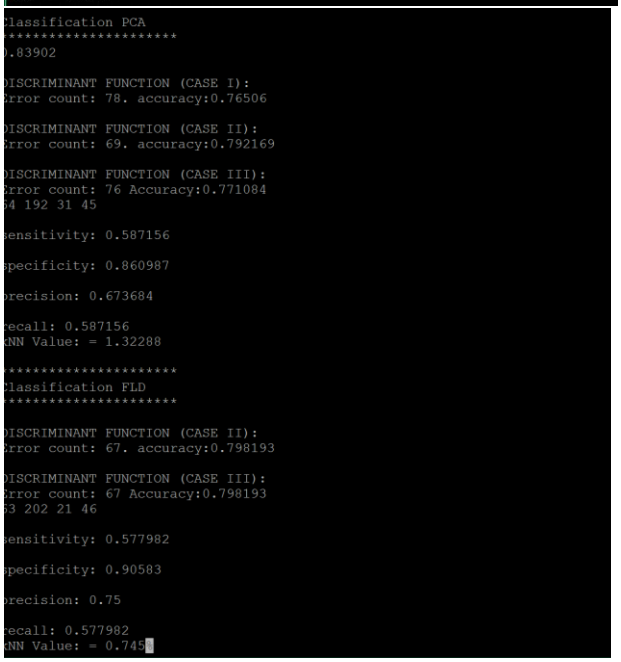
As mentioned above, the program output is pretty straightforward, and taking a look at the functions will show exactly how each variable made up the equations used to find the required values. Below is the output one gets when running my code. I implemented a **makefile** in order to efficiently compile the code and its related files. In order to run the code from the command line, simply type the following arguments: **>/.Project2 pima.tr pima.te**

This opens both the training and test file in order to begin retrieving data from it. Below is a snapshot of all the outputs generated by my code for convenience.

Output:

```
MEAN VALUES: 3.57 123.97 71.26 29.215 32.31 0.460765 32.11

SIGMA VALUES: 3.36627 31.6672 11.4796 11.7246 6.13021 0.307225 10.9754

*************************************
PCA TRANSFORMATION
*************************************


COVARIANCE MATRIX:
1 0.170525 0.252061 0.109049 0.0583361 -0.119473 0.598922
0.170525 1 0.269381 0.217597 0.21679 0.0607101 0.343407
0.252061 0.269381 1 0.264963 0.238821 -0.0473996 0.391073
0.109049 0.217597 0.264963 1 0.659036 0.0954027 0.251926
0.0583361 0.21679 0.238821 0.659036 1 0.190551 0.13192
-0.119473 0.0607101 -0.0473996 0.0954027 0.190551 1 -0.0714096
0.598922 0.343407 0.391073 0.251926 0.13192 -0.0714096 1

EIGEN VALUES:
0.302499 0.389819 0.690135 0.799971 0.911868 1.49645 2.40926


EIGENVECTORS:
-0.306227 -0.532254 -0.132269 0.500593 -0.0855989 -0.462526 0.365493
-0.107167 -0.137762 -0.444888 -0.680105 -0.41827 -0.012743 0.366002
-0.0847379 -0.108268 0.823943 -0.334982 0.0899823 -0.108116 0.412573
-0.584248 0.381459 -0.154327 0.133054 0.336837 0.414656 0.431537
0.575116 -0.413853 -0.103888 0.113808 0.241103 0.512878 0.391213
-0.0792639 0.0416248 0.259385 0.324426 -0.785018 0.449834 0.0291347
0.457353 0.606208 -0.0617026 0.196835 -0.147593 -0.36998 0.471296

BASIC VECTORS:
-0.132269 0.500593 -0.0855989 -0.462526 0.365493
-0.444888 -0.680105 -0.41827 -0.012743 0.366002
0.823943 -0.334982 0.0899823 -0.108116 0.412573
-0.154327 0.133054 0.336837 0.414656 0.431537
-0.103888 0.113808 0.241103 0.512878 0.391213
0.259385 0.324426 -0.785018 0.449834 0.0291347
-0.0617026 0.196835 -0.147593 -0.36998 0.471296
```





Discussion:

Overall the results from these experiments seemed accurate enough to what was provided and what was previously hinted at by Dr. Qi. Most of the approaches were specified in this report and shows what steps were taken in conducting these calculations. This project served to further my knowledge of the applications of these dimension reduction and performance evaluation methods. I learned a lot

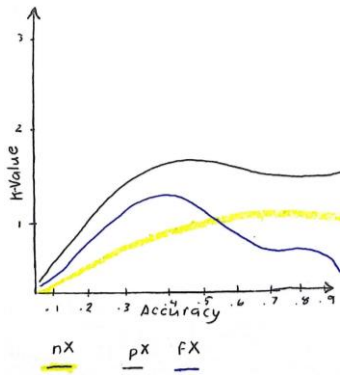
about the applications of these algorithms and how complex they can be within the field of pattern recognition.

References

[1] American Journal Of Intelligent Systems 201, Vinita Dutt, Vikas Chaudhry, and Imran Khan. "Pattern Recognition: An Overview." *American Journal of Intelligent Systems 2012* (2012): n. pag. Bhagwant University. Web.

Graphs:

Performance curve:



ROC Curve: