University of Tennessee

k-Nearest-Neighbors, Dimensionality Reduction, m-Fold Cross Validation, and Computational Optimization of kNN for Classification: An Analysis

Kirolos Shahat

kshahat@vols.utk.edu

ECE 471: Pattern Recognition

Dr. Hairong Qi

03/20/18

Abstract

We studied different combinations of classification using preprocessing such as normalization, Fisher's linear discriminant, and Principal Component Analysis with Case 1, Case 2, and Case 3 multivariate Gaussian discriminant function as well as using kNN. The result was significantly in favor of using FLD because the accuracy was skewed in the favor of these classifier. The highest accuracy came out to be approximately 80% and this was from case 2 and case 3 of using FLD data. When using m-fold cross validation on the fglass data, the result was fairly unanimous in voting for the k based on the best performance and seemed to be around k = 1 and Minkowski distance of 1 as well, also known as Manhattan Distance. After implementing Naive Bayes Classifier fusion on the prima data set the result was strictly FLD focused but did not actually improve score. Thus fusion was unneeded for this data.

Introduction

Classifying new data under a previously existing category is a fundamental problem that arises daily. This is a trivial problem to accomplish for the human brain because it takes little thought to tell the differences between an apple or an orange and classify easily. The issue with the human brain is that it can only handle a small amount of data while maintaining accuracy and thus the problem of automating classification by other means becomes a necessity. The benefits of automation is that it is always accurate in calculations because there is no physical exhaustion that the computation suffers from performing the same task repeatedly but the problem arises when decisions need to be made. This is a problem because an algorithm is unable to recall from memory in the same manner as humans and thus a way to simulate this is using classifiers based on a training set.

These theories were tested on two different training and testing sets. The first was a dataset of classifying whether a patient has diabetes or not and the second was classifying a certain type of glass based on some features of the glass for a crime scene. For the glass data set a testing set was not provided but instead a 10-fold cross validation split was provided and so this was used to validate the parameters that were chosen.

The objective of this experiment was to compare different classifiers and objective functions and analyze their strengths and benefits as well as their weaknesses and costs. The classifiers we used were: Case 1 discriminant multivariate Gaussian, Case 2 discriminant multivariate Gaussian, Case 3 discriminant multivariate Gaussian, and k-Nearest-Neighbors. These classifiers were implemented with normalization and run with and without dimensionality reduction and compared. The dimensionality reduction methods used were Principal Component Analysis and Fisher’s Linear Discriminant. All of the classifier data went through a common normalization step for preprocessing to make all of the features on the same scale. The achievement was just as one had hoped: physical values for scores and, for the case of kNN, optimization concepts and approaches for optimization in Computational time.

Technical Approach

The first step performed on all of the training datasets was normalization in the Bayesian sense. This accomplishes the goal of transforming the data into a Gaussian which has a mean of zero and standard deviation of one across the whole training set. Doing this allows the data features to be equivalently weighed and thus samples to also be equivalently weighed. The next step was an optional one which was dropping dimensions. The two dimensionality reduction methods used were Principal Component Analysis(PCA) and Fisher's Linear Discriminant(FLD). PCA works by dropping Eigenvectors based on an accepted error rate. This, it is formally defined as:

$$\sum_{i=m+1}^{d} \lambda_i / \sum_{i=1}^{d} \lambda_i$$

where m are dimensions kept and d are total dimensions. FLD works by finding the best transformation which discriminates the data as a linear function. The transformation vector is defined as:

$$\omega = S_{\omega}^{-1}(\mu_1 - \mu_2)$$

and this results in the linear discriminating transformation.

There were four total classifiers used and three of them assumed that the data obeyed a multivariate Gaussian distribution, those three being the case one discriminant function, case two discriminant function, and case three discriminant functions. Case one assumed that all of the covariance matrices were equivalent and equal to:

$$\Sigma_i = \Sigma = \sigma^2 I$$

while case two assumed only that the covariance matrices were all equivalent. Case three has no assumptions above the already assumed Gaussian distribution which the data obeys. kNN has no assumption of a distribution that the data obeys but rather uses the locality of neighboring k points to vote on the type of that class using a distance metric in a simple majority rules setting. All of these are optimal in the Bayesian sense and thus are classified as maximum likelihood metrics. The distance which was used in this experiment was the Minkowski distance of varying degrees between 1 and 100 inclusive and k values between 1 and $\sqrt{n}$ where n is the number of training set samples. All of these transformations were used on the prima and fglass data sets and the only classifier used on the fglass data set was kNN while all classifiers were used on the prima set.

The final approaches used were methods of m-fold cross validation on the fglass data set. We were given a splitting file which represented the rows used in each fold and we were to split the data set and try all combinations of training sets and test sets. This provides a way to report confidence on classification for different values of k and different distances in order to give one the best k value for the set. And lastly a Naive Bayes classifier fusion method which attempts to find the best fusion of two different classifiers for the prima data. This works by using the confusion matrices of the two classifiers and converting them into probability matrices and then multiplying the probability matrices together and taking the maximum class chosen as the fused result.

Results

Using the discriminant function as the classifier and using the prior probabilities from the training set as:

$P(\omega_0)$	0.66
$P(\omega_1)$	0.34

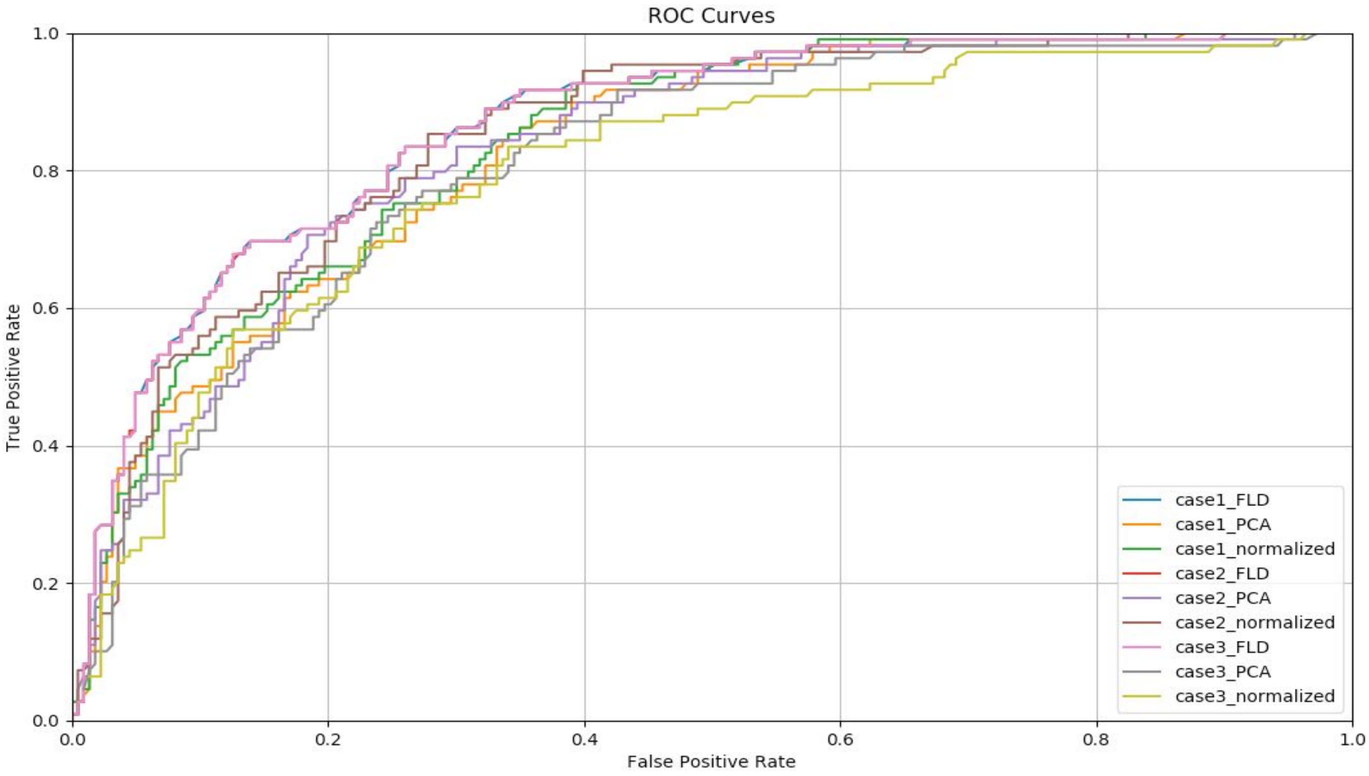
These prior probabilities allowed us to get accuracy scores for the discriminant functions using the normalized data and transformed and non-transformed data. The following results were achieved from the use of these priors and for the case of kNN having k fixed at three and using the Euclidean distance metric:

Prima Data Set	Normalized	PCA	FLD
Case 1	0.768072	0.762048	0.671687
Case 2	0.780120	0.756024	0.804217
Case 3	0.771084	0.734940	0.804217
kNN	0.740964	0.734940	0.771084
Sequential kNN time-taken(in seconds)	0.072595	0.051887	0.035164
Parallel kNN time-taken(in seconds)	0.141532	0.112741	0.085325

And for the fglass data set kNN using cross validation, the following were the best scores for each fold:

Fglass Data Set	k	Distance	Accuracy
Fold 1	9	5	0.733333
Fold 2	1	3	0.761905
Fold 3	1	1	0.821429
Fold 4	1	1	0.705882
Fold 5	1	2	0.588235
Fold 6	1	1	0.846154
Fold 7	3	1	0.681818
Fold 8	3	1	0.791667
Fold 9	3	1	0.807692
Fold 10	3	4	0.894737

When the priors were varied, ROC curves were generated for the cases and the following graph was the result



And when the k and distance were varied for the prima data set using the different transformed datasets the following were the four best results:

Prima dataset transformed with FLD	k	Distance	Accuracy
	3	100	0.771084
	11	96	0.771084
	11	98	0.771084
	11	99	0.771084

Prima dataset	k	Distance	Accuracy

transformed with PCA			
	11	10	0.771084
	11	9	0.771084
	11	8	0.768072
	11	7	0.765060

Prima dataset normalized only	k	Distance	Accuracy
	11	20	0.774096
	11	19	0.774096
	11	17	0.774096
	11	16	0.774096

And finally the best fused classes was:

Fused classes	FLD for Case 1
FLD for Case 2	Accuracy = 0.804217

All of these results are discussed in the discussion section.

Discussion

The first thing that is noticed is how well the FLD transformed data performs compared to the PCA or normalized data for cases 2 and 3 as well as the kNN implementation. Presumably, this is because of the goal of FLD is to best discriminate and as such it fits better for the case of classification. PCA tends to be the worst performing classifier likely because of the lack of the classification as a goal. Thus, when searching for the best fusion the result was case 2 and case 3 FLD because of their high accuracy with respect to the other classifiers. Even when k was varied for kNN the result still did not match the performance of FLD which shows how powerful the goal of discrimination can be. The normalized data set was the middle ground performer. It tended to do average compared to the other classifiers but did not perform as well as one would have hoped.

When m-fold cross validation was performed the best k was difficult to find but the chosen k value became 1 because of the number of occurrences and constantly performing well in the folds. The distance that maximized the validation was almost unanimously chosen to be 1, also known as the Manhattan distance. These reported an average score of about 73% which was decent.

The kNN algorithm was fairly intensive in testing but had no training time at all, thus the more classification needed the more intensive the algorithm becomes. The memory used became O(k) because the only memory needed was a list of the k neighbors but the computation became O(n * |xte| * k) because for each testing sample one must compare all the training sample distances and find the max. Thus, with memory being O(k) the algorithm becomes more expensive with time. Thus a way to reduce that was to parallelize each testing sample classification to run on a different thread of execution. This would theoretically classify all of the testing samples in the time it takes to classify one of them if run on a computer that had cores that matched the number of training samples but on a personal computer, the time needed to spawn a thread becomes more expensive than just doing the comparisons sequentially. Thus, this is application dependant and it showed that running on a macbook pro with intel i5 processor, multithreading took almost double the time for all cases and thus did not help.

Conclusion

We compared multiple classifiers with different degrees of preprocessing and compared their performance. For the prima data set the best performing classifier turned out to be case 2 or 3 based on normalization and FLD preprocessing steps. This was likely due to the FLD's goal being discrimination and through the transformation found a good representation of the data in the form of a line. After implementing 10-fold cross validation on the fglass set, the result was not strictly unanimous but it seemed to be consistent that the best k and distance values were both one and seemed to average to about 73%. The goal moving forward with these would be to attempt to try different methods of dimensionality reduction and different classifiers that could classify these sets in a better manner.