8-6-2009

Classification of Carpiodes Using Fourier Descriptors: A Content Based Image Retrieval Approach

Patrick Trahan
University of New Orleans

Classification of *Carpiodes* Using Fourier Descriptors: A Content Based Image Retrieval Approach

A Thesis

Submitted to the Graduate Faculty of the
University of New Orleans
in partial fulfillment of the
requirements for the degree of

Master of Science
in
Computer Science
Artificial Intelligence

by

Patrick J.Trahan

B.A. Nicholls State University, 1995
M.S. Nicholls State University, 1998

August, 2009

Acknowledgements

I would like to thank my advisor Dr. Dongxiao Zhu. His help was integral to the completion of this thesis. He taught me how to apply all of the knowledge and theories I learned in class to the application in research. I have learned much about proper research techniques and writing scientific reports. I appreciate all his time and efforts that he dedicated to helping me complete this project. He helped me to explore the depth of my own knowledge and abilities. Without all his valuable help, I would not have had the confidence in myself to accomplish this manuscript. He is a true mentor and friend. I feel I am a better Computer Scientist and Mathematician because of his help.

I would like to thank my previous advisor Dr. Yixin Chen. Thank you for believing in my ability to perform research. I really enjoyed your classes and I have obtained so much knowledge from them. I have also thoroughly enjoyed working with you. You never turned me down for help when I needed it. You were invaluable to my success.

I would like to thank my whole family for their support throughout this process. Specifically, I would like to thank my wife, Angie, my daughters Alexis Grace, and Brittany Nicole for giving me their time, inspiration, and, most of all, their patience, love, and support. You all taught me that working on research, although tough on the researcher, is equally as tough on family. Thank you all for the sacrifices you made for me. I love you all.

Finally, I would like to thank my parents, the late Wiley and Hilda Trahan. Although I lost you both at a very young age, you instilled many values in me that I carry with me to this day. One closest to my heart is the value of education. My education is one of my deepest treasures. There is no amount of thanks I could give to repay you. I dedicate this manuscript in memory of both of you. I love and miss you both.

Table of Contents

List of Figures

List of Tables

List of Illustrations

List of Images

Abstract

Taxonomic classification has always been important to the study of any biological system. Many biological species will go unclassified and become lost forever at the current rate of classification. The current state of computer technology makes image storage and retrieval possible on a global level. As a result, computer-aided taxonomy is now possible.

Content based image retrieval techniques utilize visual features of the image for classification. By utilizing image content and computer technology, the gap between taxonomic classification and species destruction is shrinking. This content based study utilizes the Fourier Descriptors of fifteen known landmark features on three *Carpiodes* species: *C.carpio*, *C.velifer*, and *C.cyprinus*.

Classification analysis involves both unsupervised and supervised machine learning algorithms. Fourier Descriptors of the fifteen known landmarks provide for strong classification power on image data. Feature reduction analysis indicates feature reduction is possible. This proves useful for increasing generalization power of classification.

Keywords: Content-Based Image Retrieval, Alpha Taxonomy, Beta Taxonomy, Gamma Taxonomy, Principal Component Analysis, K-Means Clustering, Hierarchical Clustering, K-Nearest Neighbor, Support Vector Machine, Random Forest, Quadratic Discriminant Analysis, Feature Reduction, Variable Importance

Chapter 1 Introduction

Taxonomic classification has always been important to the study of any biological system. However useful, it does have inherent problems. Classification is a slow, tedious process that follows very strict rules laid out in the International Code of Zoological Nomenclature [1]. Taxonomy can proceed along three lines: alpha, beta, and gamma taxonomy. Alpha taxonomy focuses on a low level of classification along the lines of the species. It deals with finding, describing and naming organisms. Beta taxonomy focuses on grouping organisms into higher taxa, beyond the species level. Gamma taxonomy examines evolutionary relationships, phylogenetics. With the discovery of characteristics on newly found organisms taxonomic revision can be justified. As a result previously recognized species can then be split into a new species.

Alpha Taxonomies used to diagnose new species can proceed among many lines. They can be morphometric and shape characteristics, color [2], appendages, and even DNA data [3]. Furthermore, these diagnostic tools are used to compare many sample specimens throughout its geographical range. As a result, taxonomic classification and revision can be very laborious and time consuming.

The slow process of alpha taxonomy creates a taxonomic crisis. Currently, the rate of species destruction is about 100 times the rate of classification [4]. Due to habitat destruction and human activities, this number will continue to increase [4]. With only about 1.4 million species known and described compared to about 30 times this number that is unknown, many species will become extinct long before they can even be described and known [2]. Current manual taxonomic methods may be useful, but, they are not efficient enough to keep the pace with species destruction. Computer technology can help combat the current biodiversity crisis.

Improvements in computer and database technologies and storage capacities make it feasible to store and document images electronically. As a result, recent years have seen a rapid increase in the size of digital image collections [5]. The internet bridges regional gaps that make image databases available in geographically different regions that may not have access to such tools otherwise [2, 6, 7]. Now, taxonomists around the world are linked to common databases and literature making classification possible at a faster, more accurate pace [2].

With the positive influence of technology, there is a downside. The world wide web has created an information revolution with respect to the taxonomic data. There are more data available than ever before. However, the more information on a subject, the harder it is to organize and retrieve relevant information [7]. We no longer have to be concerned with a lack of documented data. Instead, we are concerned with how one can find and process relevant data. Image databases must be organized in a way to allow access to the information in a useful, way to facilitate searching and retrieving relevant data [5]. Much research on image retrieval can be categorized as either text based or content based. Text based techniques draw heavily from database technologies. However, Computer Vision and Pattern Recognition tend to be the major source fields for content based techniques.

Text based image retrieval (TBIR) has been an active area of research for about 30 years [5]. Text based techniques rely on image annotations and database management tools. According to [5], text based techniques, although promising, suffered from some difficulties. The first difficulty is that text based techniques require annotations by a human observer: the subjective perception of the annotator. Different annotators can perceive the images differently, thus, influencing the annotations. The second difficulty involves the labor required to annotate images in databases. Small databases are not very problematic. However, larger databases obviously require much more work that may not be possible or efficient.

Content based image retrieval (CBIR) techniques have been studied since the early 1990's [5]. Content based image techniques do not rely on the image annotator's descriptions. Instead, CBIR extracts visual features inherent in the image. Because extraction is automatic and can depend on content features, the two major difficulties of TBIR can be greatly reduced with CBIR techniques. By using inherent image features, annotations may be eliminated. As will be shown later, this is not done in the strictest sense. Text based annotations are not completely obsolete in most systems. However, reduction of dependency on TBIR techniques reduces the subjectivity of the annotator. Furthermore, the automated process allows the use of larger image collections without much human labor.

CBIR techniques can vary on the features used for analysis. Some techniques utilize shape based features while some utilize color based features. These techniques can be combined to allow even more methods for analysis. Whatever method used, the goal is to extract features from the image itself for analysis based on statistical learning algorithms. Despite the abundance of features and techniques used for analysis, feature selection should not be redundant and should remain focused for analysis. The use of too many features can results in model – over-fitting. Therefore, poor generalization can result on test sets. This study attempts to utilize specific shape based features. Furthermore, it is designed to utilize the same shape based features used by taxonomists for species classification

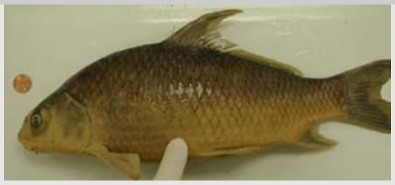
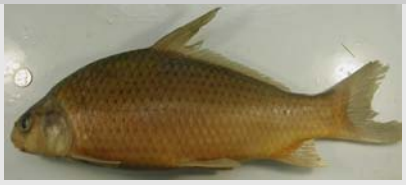
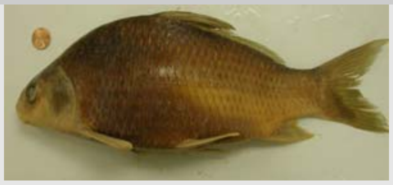
Chapter 2 Data Background

2.1 Data set

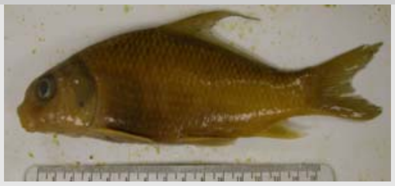
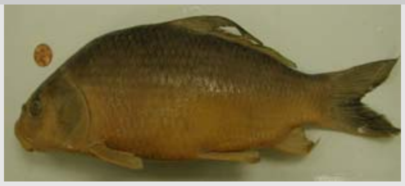
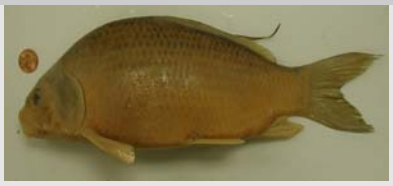
The dataset used in this research is an image database consisting of 651 specimens of the fish genus *Carpiodes*. *Carpiodes* fall into three different major species: *Carpiodes carpio* (*C.carpio*), the river carp-sucker, *Carpiodes cyprinus* (*C.cyprinus*), the quillback, and *Carpiodes velifer* (*C.velifer*), the highfin carp-sucker. Below is a representation of the images in the database (Images 2.1 -2.9).

Sample Images of the *Carpiode* Dataset

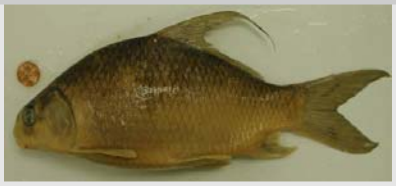
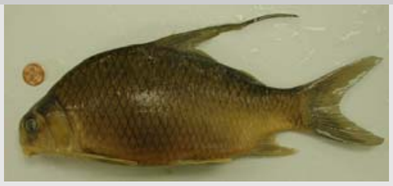
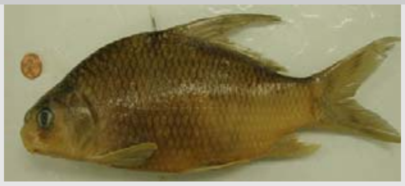
Images 2.1 – 2.3 *C.carpio* Fish Specimen

Images 2.4 – 2.6 *C.cyprinus* Fish Specimen

Images 2.7 – 2.9 *C.velifer* Fish Specimen

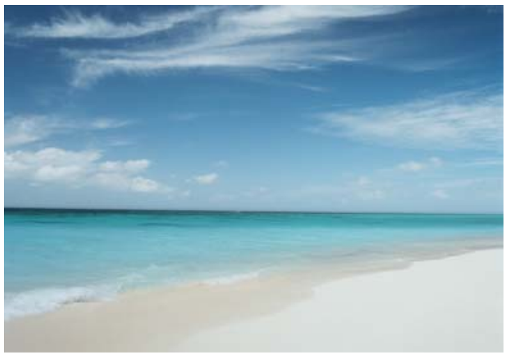




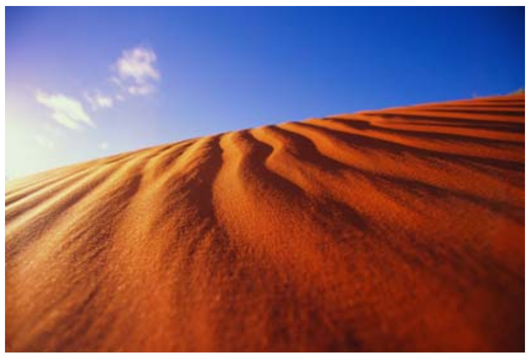
Around 40 years ago, traditional morphometrics used multivariate statistical tools and shape data to describe patterns of shape variations [8]. Linear distance measures, along with statistical tools such as Principle Component Analysis (PCA), Factor Analysis, Canonical Variate Analysis (CVA), and discriminant functional analysis, were combined with morphometric measurements for quantification and patterns of variation within and among samples [8]. This research will analyze more modern machine learning algorithms for analysis in combination with some classical techniques.

2.2. Semantic Classification

There is a problem with image query methods. A common technique, query by example, would be to find an image in a database that most closely matches that of the query image. Classification of images as similar depends on multiple features. Problems arise however when images are classified as similar based on color, shape, or object similarity. Images 2.10 and 2.11 below both can be classified as similar based on large regions of blue and white sky or large regions of sandy land. They do, in fact, represent different scenes.

Images 2.10 & 2.11 Beach Scene & Desert Scene





Some images share similar objects but represent two different scenes. The images below both contain sitting furniture. However, Image 2.12 represents an indoor living room. Image 2.13 represents an outdoor patio. These two images can be classified as similar based on objects in the scene. However, they represent two different scenes altogether. Sitting furniture is not the only object type that can be used to classify these images as similar. They both have plants as well as walls. Correct classification of these scenes would require detailed knowledge of the objects in each image.

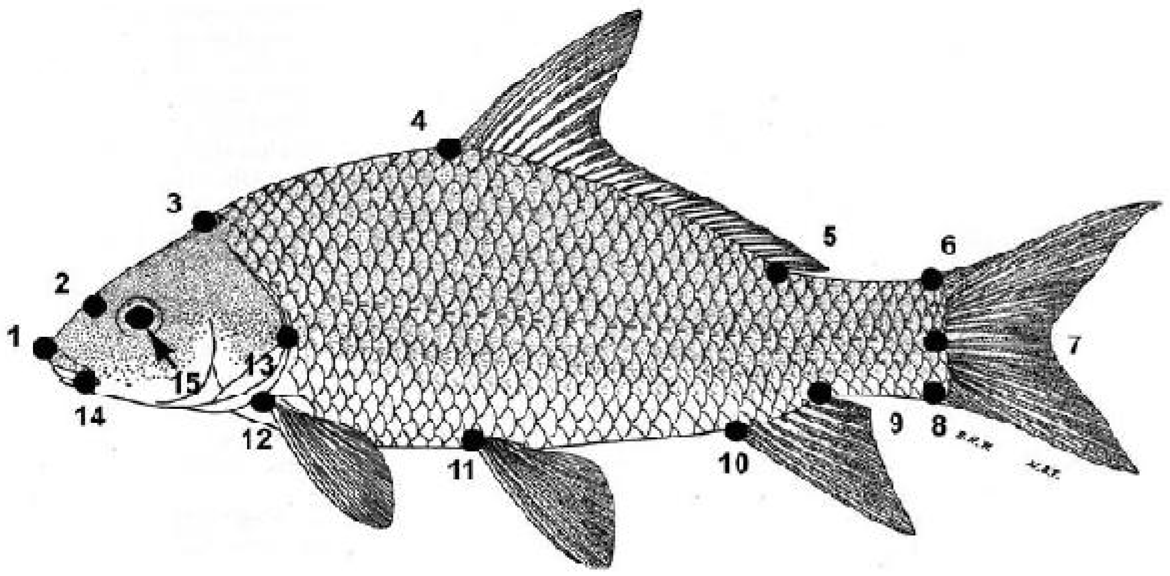
Image 2.12 & 2.13 Living Room Scene & Patio Scene

These areas of misclassification represent the semantic gap that can plague correct image classification.

Correct classification would then require some method of bridging this semantic gap. Images should include some feature that is unique to the queried scene. The fish images were analyzed based on fifteen homologous landmarks. Non-shape variation data should be removed before analysis. The center of the fish image should be the mean value of all the (x, y) points.

Given an image, there are 15 homologous landmarks on the fish image (Illustration 2.1). The images can be thought of as being on a Cartesian Coordinate System with the landmark data representing both x and y coordinates. Therefore, each landmark becomes an ordered (x, y) pair.

Illustration 2.1 Fish with 15 Known Landmarks



Each landmark can then be used to create features for each species.

Most images have text based annotations. They are considered an integral part of the CBIR system. Text annotations, however, may not be standardized and they may be handwritten. As a result, consistency among annotated images can be very difficult to obtain. Therefore, due to the limitations inherent in text based techniques, a more content based approach should be implemented.

With the content based approach, the use of a feature vector is involved. An infinite number of features can be employed. However, a large feature vector can suffer from dimensionality problems that can result in a number of problems from model over-fitting to an increase in computational time for what could be useless data. This study will utilize the fifteen known landmarks on the fish images for classification.

Chapter 3 Feature Representation

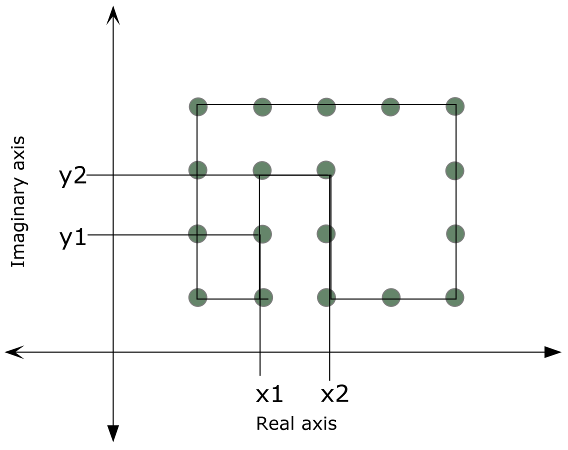
3.1 Shape Representation of an Image

Images can be decomposed into image connected components. In general, our concern is for the connected components. Connected components can trace out the boundary of the image. However, they can be internal values as well. In particular, the focus of study is on the view that connected components are the landmark data. Fourier descriptors are used to abstract the connected components of the landmark data. However, Fourier descriptors describe data in the frequency domain and they require values in the complex space. The points in the spatial plane need to be transformed into the complex plane. To understand how to abstract the data, consider illustration 3.1 depicting the connected components of a boundary object.

Illustration 3.1 Discrete Contour Points

Connected Components of a Shape Object

x-axis=real axis, y axis = imaginary axis



This is a two dimensional object in (x, y) coordinates. Let $d_t = a + bi$ be a complex number, where a represents the real number part and b represents the imaginary part.

Then, if we let x represent the real axis and y represent the imaginary axis, the two dimensional figure can be written in the form $d_t = x + yi$, which is a one dimensional complex number. Therefore, the two dimensional object in the real plane can be represented as a one dimensional object in the complex plane.

The 15 homologous landmarks are represented as two dimensional coordinates in the (x, y) plane. Let the x coordinate represent the real part and y represent the imaginary part of a complex number. Then, for $j = 1...15$, each landmark can be written as $LM_j = x_j + y_j i$. As a result, each landmark, LM_j, is transformed into a one dimensional complex number. LM_j is used to find the Fourier Descriptors.

3.2 Shape Representation using Fourier Descriptors

As shown in illustration 3.1, the shape of the image can be laid out on the complex plane in one dimension. If the image objects are periodic, they can be represented by a Fourier Descriptor [15]. From illustration 2.1, one can deduce that the landmark data can be viewed as discrete and periodic. Therefore, using the discrete Fourier Transform,

$$D_f = \frac{1}{N} \sum_{t=0}^{N-1} d_t e^{-\frac{j2\pi tf}{N}}$$

$f = 0 \ldots N-1, = \sqrt{-1}$, and $\frac{j2\pi tf}{N}$ represents the phase shift of the descriptor, N is the number of boundary points. The one dimensional complex number, d_t, can be applied to convert it to the frequency domain using Fourier Descriptors. The values $D_0, D_1, \ldots.$, are called the Fourier Descriptors. Furthermore, the Fourier Descriptors describe the contour of the object in the frequency domain.

The Fourier Descriptors in the frequency domain can be transformed back to their inverse in the spatial domain by

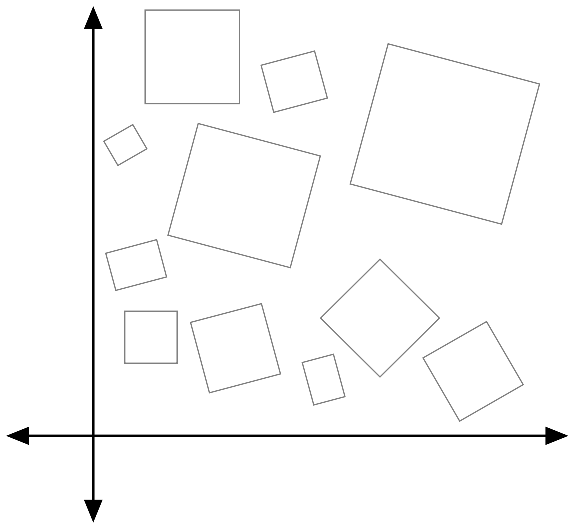
$$d_t = \frac{1}{N}\sum_{f=0}^{N-1} D_f e^{-\frac{j2\pi tf}{N}}$$

$t = 0 \dots N-1$, $j = \sqrt{-1}$, and $\frac{j2\pi tf}{N}$ represents the phase shift of the descriptor.

Fourier Descriptors carry general shape description information. The only descriptor that carries non-shape related data, or shape location data is D_0 , which represents the shape mean value. The remaining descriptors contain general shape data. The lower descriptors, lower frequency data, carry contour information. That is, the general shape of the object. The upper descriptors, higher frequency data, carry detail information about the object.

The images are subject to three types of variations that should be corrected due to image capture issues. Each image can have size variation, centroid, and location, variation, and rotation variation. Similar images can be classified as different based on different center values. Illustration 3.2 explains the three types of variation corrections needed.

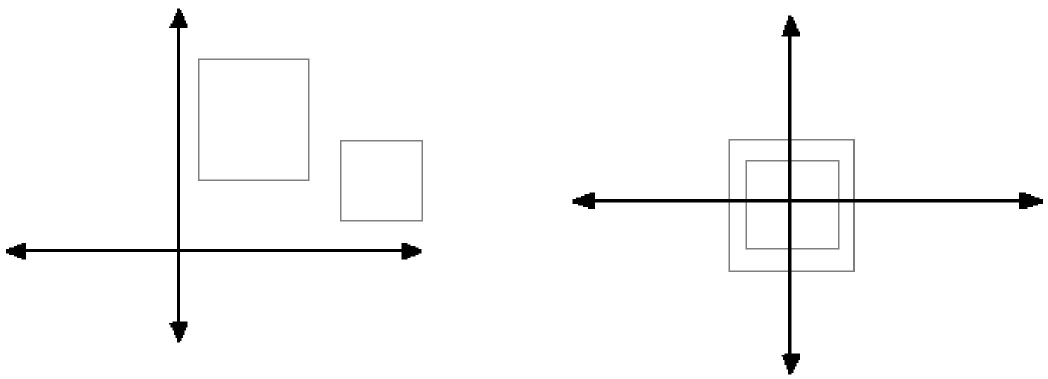
Illustration 3.2 Shape Variations Affecting Analysis: Centrod, Size, & Rotation



Using the landmark data for analysis will have problems with location, rotation, and scale. Each fish image, although seen as being superimposed onto a coordinate system, are not placed at the same center point. The images can be taken with some standard landmark method for object placement. However, there will usually be some minor rotation of the object creating non shaped based variation. Finally, fish size can vary with age. A comparison of fish based on size alone can encourage misclassification by grouping larger and smaller fish together without respect for species. Therefore, there will be variations in the locations of the landmark data that doesn't represent inherent differences in the fish species, but, differences in the coordinate position. As a result, shape variation data can be corrupted by non shape variation data.

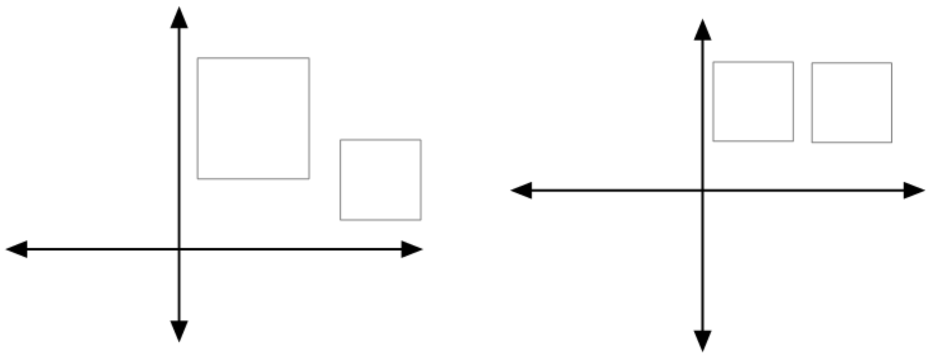
The only descriptor to carry location information is D_0. The location variation can be corrected by setting $D_0 = 0$. All images are now centered about zero (Illustration 3.3). Furthermore, since all images are centered about zero, D_0 does not contain any interesting information that could be used for discriminating species. Therefore, it can be ignored altogether in classification.

Illustration 3.3 Centroid Variation and Correction



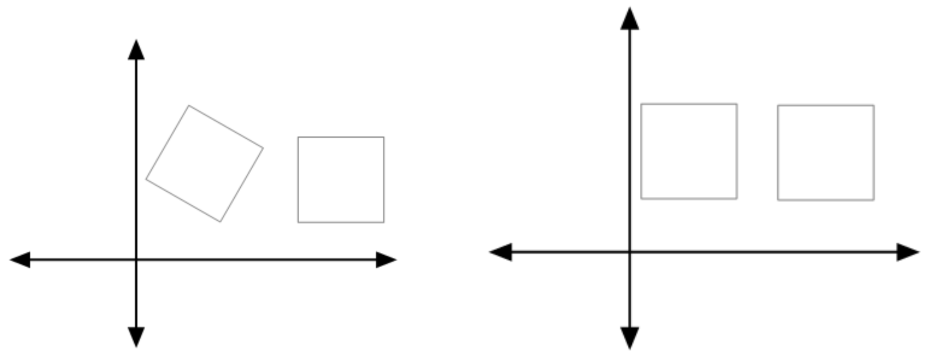
The second variation correction needed is size variation. Each image can vary with respect to size. Normalization by the sum of all the features will remove size variation by utilizing the unit vector property. For each image, k, the fifteen Fourier Descriptors are normalized as $\frac{D_{ki}}{\|D_{ki}\|}\ for\ i = 1 \dots 15$ (Illustration 3.4).

Illustration 3.4 Size Variation and Correction



Finally, rotation variation can be corrected by taking the amplitude of each descriptor, $|D_i|\ for\ i = 1 \dots 15$. Images can be classified as similar, or dissimilar, when, in fact, they are not based on rotation of the object in the image. If distance based classification techniques are used, similar objects with equal centroids and equal sizes can be misclassified because rotation implies the images are dissimilar (Illustration 3.5).

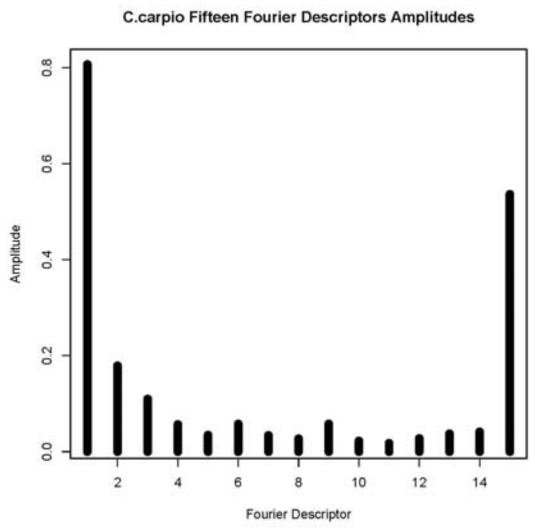
Illustration 3.5 Rotation Variation and Correction



All the images are now aligned by location, size, and rotation. Figure 3.1 describes the adjusted data in terms of the amplitude of the Fourier descriptors corrected for centroid, size, and rotation variations.

Figure 3.1 Fourier Descriptors Amplitude for a

***C.carpio* Fish Image**



Chapter 4 Unsupervised Learning Techniques

4.1 Introduction

Learning algorithms fall into two general classification types: unsupervised and supervised. The major difference lies herein with the use of classification labels. Supervised learning trains a model based on prior knowledge of the correct class of the record. However, the unsupervised techniques do not use such prior knowledge for clustering. In this study, I utilized three techniques for unsupervised learning: Principal Component Analysis, K-Means Clustering, and Hierarchical Clustering.

4.2 Principal Component Analysis

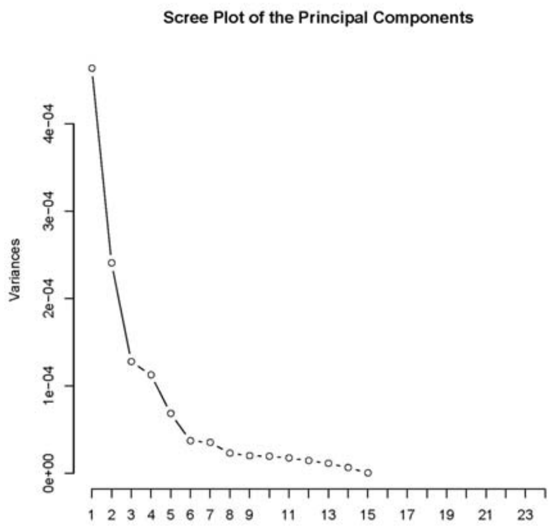
Principal Component Analysis is a method of finding the best features for representing the data. The result is a component set of the original features each rotated in the direction of decreasing variability. Therefore, the new dimensions are optimal for reducing the sum-squared error. Data dimensions tend to be large for some datasets. Furthermore, dimensions can be redundant and provide no more useful information to a classification study than a reduced subspace. Finally, classification techniques may not perform well due to high dimensionality. This is also known as the curse of dimensionality. Therefore, Principal Component Analysis reduces the dataset and views the data in terms of best features in a rotated subspace for analysis.

Given an $m \times n$ data matrix, D, the $m \times n$ covariance matrix is defined as $S = Cov(D)$. The feature mean vectors are given by $\vec{\mu}_j$. Each entry of S, s_{ij}, is the covariance of the i^{th} and j^{th} attributes. The eigenvalues and eigenvectors of S indicate the direction, by eigenvector, and magnitude, by eigenvalue, of the variance in the data. Because S is a positive semi-definite matrix, eigenvalues of S can be calculated and orderd as $\lambda_1 \geq \lambda_2 \geq \lambda_3 \geq \ldots \geq \lambda_{m-1} \geq \lambda_m$ with eigenvectors $U = [\vec{u}_1, \vec{u}_2, \vec{u}_3, \dots \vec{u}_{m-1}, \vec{u}_m]$. These eigenvectors are orthogonal to each other. As a result, they form basis vectors in a new rotated space. The eigenvalues and eigenvectors are ordered such that the pair with the largest eigenvalue represents the component that accounts for the most variance in the data and the corresponding eigenvector indicates the direction of variation in the data. This is known as the first principal component. The second is known as the second principal component. The pattern continues for all eigenvalue and eigenvector pairs. The new data values can be represented as

$$\vec{x}' = U^t(\vec{x} - \vec{\mu}).$$

Dimensionality reduction is performed by retaining only those eigenvectors that account for most of the variation in the attributes. There are methods designed to aid in this decision process. One method is to discard any attribute in which the variance is less than 1. Another method is to use a scree plot. Scree refers to debris falling from a mountain down to its base. The scree plot for Principal Component Analysis appears as an action sequence of this process (Figure 4.1).

Figure 4.1 Scree Plot of the Variance against the Principal Components



Therefore, analysis should stop at the point where the variance levels off. In terms of the eigenvalues and eigenvectors, scree refers to the leveling off of the variance. Those components that contain only a small amount of variation, force the scree plot to level off. Hence, they can be discarded resulting in dimensionality reduction. Of course, the limitation is that discarding dimensions also discards information about the dataset. It is possible that misclassification can increase. By retaining the components that account for the most variation, the error increase will be minimal.

4.3 K-Means Clustering

K-Means Clustering is an iterative algorithm designed to cluster m quantitative objects into k partitions or clusters. The objects are quantified by the euclidean distance measure

$$d(\vec{x}_i, \vec{\mu}_j) = \|\vec{x}_i - \vec{\mu}_j\|^2,$$

where x_i is a quantitative vector and $\vec{\mu}_j$ is a mean or centroid value.

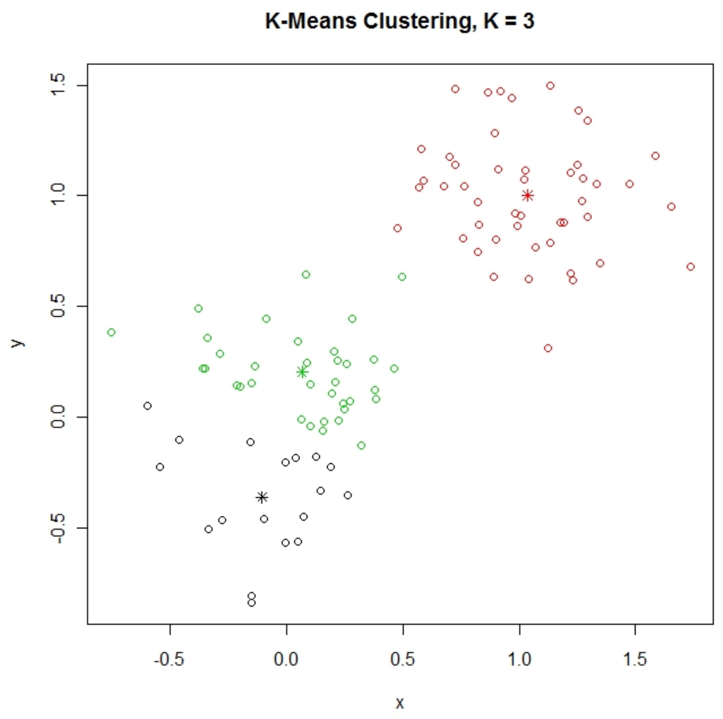
That is, K means are compared with each object, $\vec{x}_i$. $\vec{x}_i$ is then assigned to the cluster that produced the minimum distance to its mean. As a result with-in cluster variance is minimized by assigning $\vec{x}_i$ to minimum $d(\vec{x}_i, \vec{\mu}_j)$.

```
Algorithm: K-Means
Input: Q = {$\vec{x}_i | \vec{x}_i \in \mathbb{R}^m$}
        $Number\ of Clusters\ K \in \mathbb{Z}^+$
Initialize: $K$ initial means, $\mu_j \in Q$
While cluster assignment does not change
  Calculate
        $d(\vec{x}_i, \vec{\mu}_j) = \|\vec{x}_i - \vec{\mu}_j\|^2, for\ j = 1 \ldots K$
```

```
    Assign $\vec{x}_i$ to cluster $C_j$ by
        $$\underset{1 \le j \le K}{\operatorname{argmin}} \left\| \vec{x}_i - \vec{\mu}_j \right\|^2$$
    Recalculate $\vec{\mu}_j$ using $C_j$
End While
Output: Clustered data set
End
```

The results of the algorithm can be described in Figure 4.2, where each group is represented by different colors red, green, and black. The final calculated means are represented by asterisks.

Figure 4.2 Graph of K-Means Clustering of 2 Dimensional Data Set, K = 3

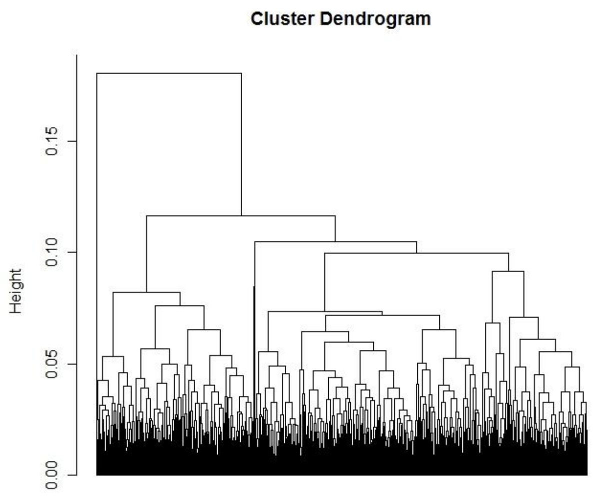


The algorithm is sensitive to the initial selection of K and the number of K classes. Therefore, the above graph can vary and produce different results. To produce acceptable results, selecting K should remain random. This method will produce good, although very likely sub-optimal, results.

4.4 Hierarchical Clustering

Hierarchical Clustering creates a cluster hierarchy by either of two methods: agglomerative (builds clusters) and divisive (break down clusters). The customary graphical depiction of the clustering algorithm is a rooted binary tree known as a dendrogram (Figure 4.3). The grouping points are called nodes and the terminal nodes are called leaves.

Figure 4.3 Dendrogram of Hierarchical Clustering Algorithm



While the K-Means algorithm is sensitive to the choice of K and number of classes, hierarchical methods do not require that type of specification. Instead, a distance based measure of pair-wise dissimilarity between groups of observations is used instead.

Agglomerative clustering, used in this study, builds cluster hierarchies by starting with the singleton elements, known as singleton clusters, building clusters from a bottom up method. The algorithm recursively builds larger clusters by merging pair-wise smaller clusters at each step. Each level of the hierarchy represents the level of dissimilarity between connected clusters.

Clustering requires a measurement of dissimilarity, $d(H, G)$, between two clusters , H and G, on which to base partitioning the data into larger clusters. Depending on the clustering requirements, this measure varies. There are three types of agglomerative clustering to consider. Type 1 is Single Linkage (SL) clustering. In this case, the inter-group dissimilarity measure is taken to be the closest pair in clusters H and G. Then dissimilarity becomes

$$d(G, H) = \min_{\substack{i \in G \\ j \in H}} d_{ij}$$

Type 2 is Complete Linkage (CL) clustering. The inter-group dissimilarity measure is taken to be the farthest pair in clusters H and G. Then dissimilarity becomes

$$d(G, H) = \underset{\substack{i \in G \\ j \in H}}{max}\, d_{ij}$$

Type 3 is Average Linkage (AL) clustering. The inter-group dissimilarity measure is taken to be the average dissimilarity between H and G.

$$d(G, H) = \frac{1}{N_G N_H} \sum_{i \in G} \sum_{j \in H} d_{ij}$$

N_G and N_H are the number of objects in each cluster.

Single Linkage Clustering seems to be the natural progression through classifying objects as similar because we tend to think of similar as closeness. However, due to chaining, when observations are linked by a series of close intermediate observations, the compactness property is violated. Average Linkage is sensitive to monotone transformations. Therefore, in this study, Complete Linkage is used to produce a compact tree due to the size of the fish data set. It can however violate the closeness property. Regardless, it can still represent an image of grouping of the data.

Algorithm

```
Input: Q = {$\vec{x}_i | \vec{x}_i \in \mathbb{R}^m$}
Initialize: $Number\ of\ Clusters = |Q|$, the cardinality of $Q$
Compute the pair-wise distances $d(x_i, x_j)$ for all $Q$. Each $x_i$ is considered a singleton cluster.
While $Number\ of\ Clusters > 1$
    Cluster pair-wise clusters $G$ and $H$ by
        $d(G,H) = \max_{\substack{i \in G \\ j \in H}} d_{ij}$
    The new cluster is GH.
    Recalculate the new distances $d(GH, K) = d_{ij}, i \in GH, and\ j \in K$
    $\text{Number of Clusters} = \text{Number of Clusters} - 1$
End While
Output: Clustered data set
```

Chapter 5 Kernel Based Classifiers

5.1 K-Nearest Neighbor

The K-Nearest Neighbor Algorithm is a memory-based, model free, nonparametric classification technique. Its uses are far reaching in Machine Learning. K-Nearest Neighbor algorithms have been used in character analysis [11,12] and medical ekg analysis [10] successfully. The algorithm is simple and flexible, yet powerful.

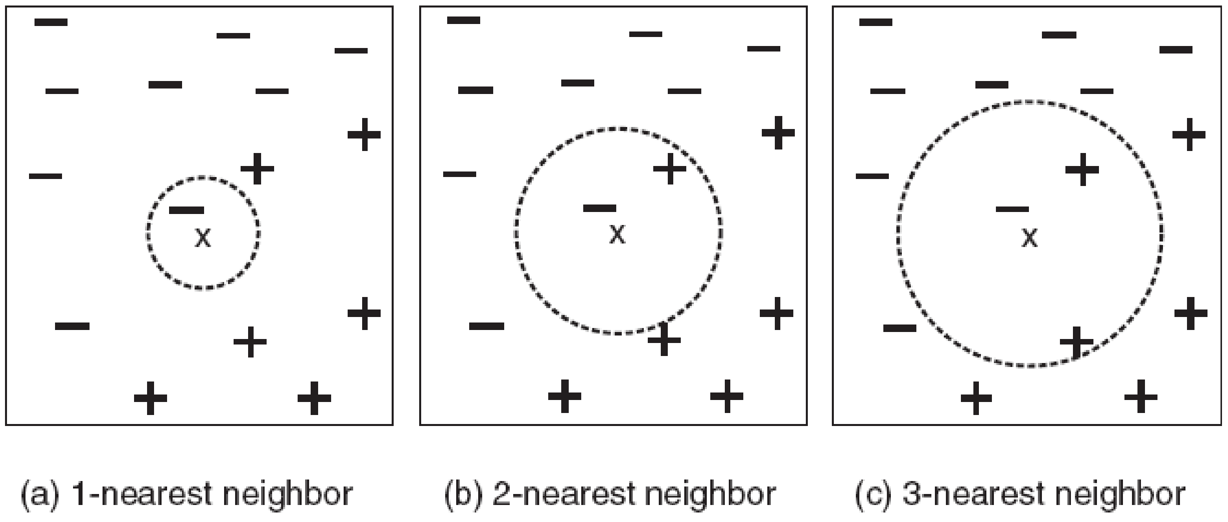
Algorithm

```
Given: Training set $T = \{(x_{i,}y_i) | x_i \in \mathbb{R}^n, y_i \in \mathbb{Z}\}$
Input: Query point set $Q$ such that $T = \{(x_i', y_i') | x_i' \in \mathbb{R}^n, y_i' \in \mathbb{Z}\}$ and the number of nearest neighbors $K$.
for all points in Q
    Compute $d_i = \|x_i' - x_i\|$.
    Select $T_K \subseteq T$ to be the $K$closest points to the query point in $Q$
    $y' = argmax \sum_{(x_i, y_{i)\in T_K}} I(y_i)$
End for
Output: y'
```

Given a query point (x_i', y_i') in $Q = \{(x_i', y_i') | x_i' \in \mathbb{R}^n, y_i' \in \mathbb{Z}\}$ and training points from a training set $T = \{(x_{i,}y_i) | x_i \in \mathbb{R}^n, y_i \in \mathbb{Z}\}$ classification of (x_i', y_i') is based on modal results of proximity measures between $x_i'$and x_i, for $i = 1 \ldots . K$. Mode induced classification will work with any Kvalue. However, it works best if K is an odd integer for binary classification. If K is even, then multimodal ties must be broken at random.

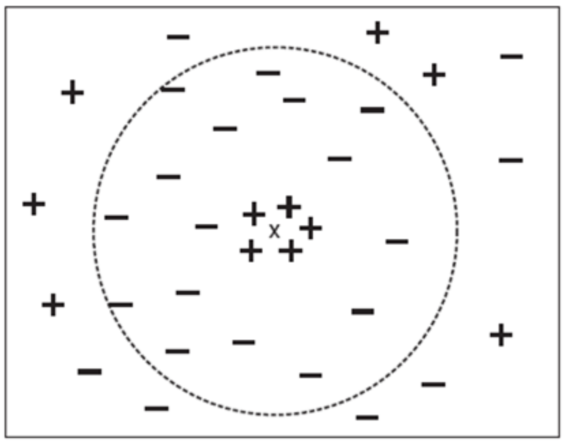
Therefore, each query point is compared against a K size neighborhood of points in T. The neighborhood of points can vary (Illustration 5.1). However, K must be fine tuned to minimize classification error.

Illustration 5.1 K-Nearest Neighbors [9]



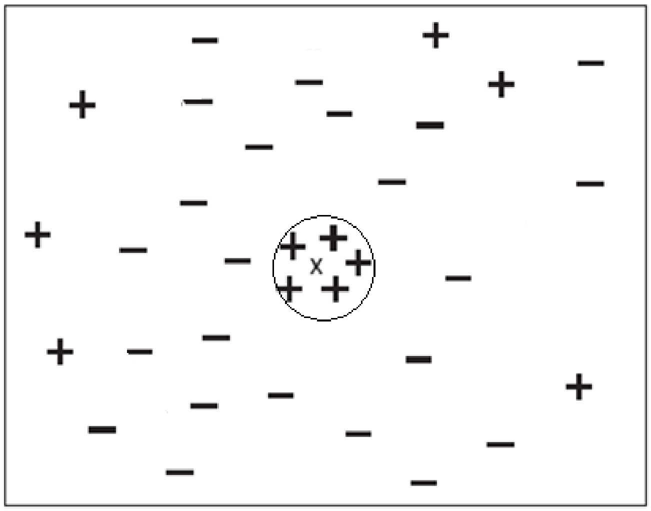
During the tuning phase for K, there are many considerations. KNN is sensitive to training set size for each class. It works best when there are a number of representative samples from each given class. However, if there are unequal training samples for each class, the algorithm will produce skewed classification in favor of the largest training class if K is sufficiently large enough (Illustration 5.2).

Illustration 5.2 KNN Sufficiently Large Neighborhood [9]



The same effect can occur if K is sufficiently small. KNN is a local optimizing algorithm. If the query point is locally optimized by training samples from a different class, it reduces the chances of obtaining a representative sample of the true neighborhood of points (Illustration 5.3). As a result misclassification can occur.

Illustration 5.3 KNN Sufficiently Small Neighborhood [9]



Furthermore, by choosing K sufficiently small enough, the classifier becomes sensitive to noisy or irrelevant points (Illustration 5.3). Therefore, classification error due to mislabeled training points will be magnified during the test phase of classification.

By fine tuning K, misclassification error due to dominant class samples, unrepresentative neighborhoods, and noisy data can be controlled. K is tuned using five-fold cross validation to vary K and compare the predicted generalization error against the training error.

Proximity measures indicate the need for a metric. Although they can vary, this approach utilizes the kernel based Euclidean distance measure: $d_i = \|x_i' - x_i\|$. Therefore, after choosing K all d_i are compared. Then, $N - K + 1$ training points are removed to form the neighborhood around the query point. Then, classification is assigned to the modal class of the neighborhood.

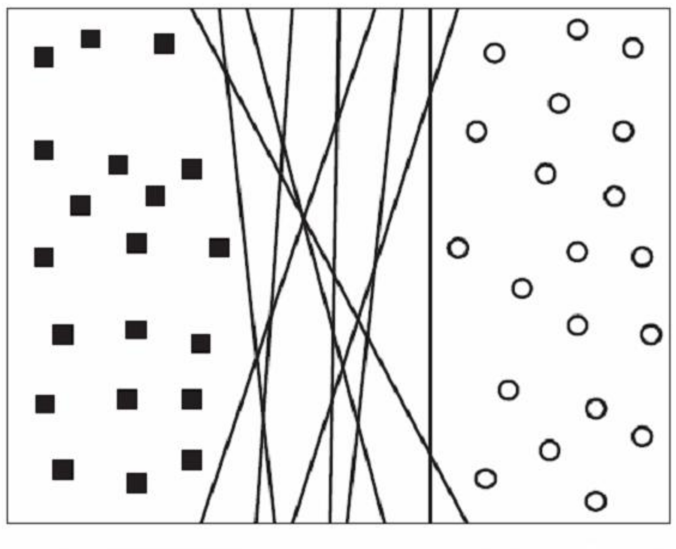
5.2 Support Vector Machine

5.2.1 Linearly Separable Support Vector Machine

The concept of the Support Vector Machine belongs to the family of linear classification techniques that utilize supervised learning [17]. Generally, the techniques are used for binary, linearly separable classification. But, there are some techniques that can classify more classes. Furthermore, there are methods that allow non-linear separable classification. The concept of the Support Vector Machine is relatively new to machine learning techniques. However, it has shown very promising results with respect to classification in both low and high dimensional datasets.

The Support Vector Machine approach to machine learning is based on the concept of the maximum margin hyperplane. Consider linearly separable classes represented by squares and circles that can be classified by infinitely many different decision boundary hyperplanes (Illustration 5.4.)

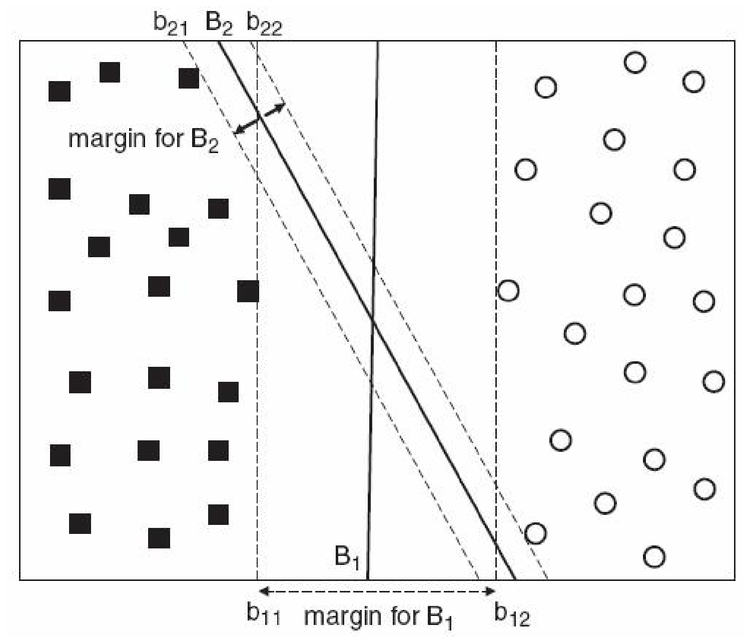
Illustration 5.4 SVM Linearly Separable Binary Classes [9]



In each of these cases, classification is perfect for each class. Unseen samples may not perform as well as the training examples. A good classifier should find a hyperplane that minimizes generalization errors.

Consider the two hyperplanes, B_1 and B_2, that classify the training examples (Illustration 5.5). Each hyperplane has a margin associated with it. B_1 has a margin defined by b_{11} and b_{12}. B_2 has a margin defined by b_{21} and b_{22}.

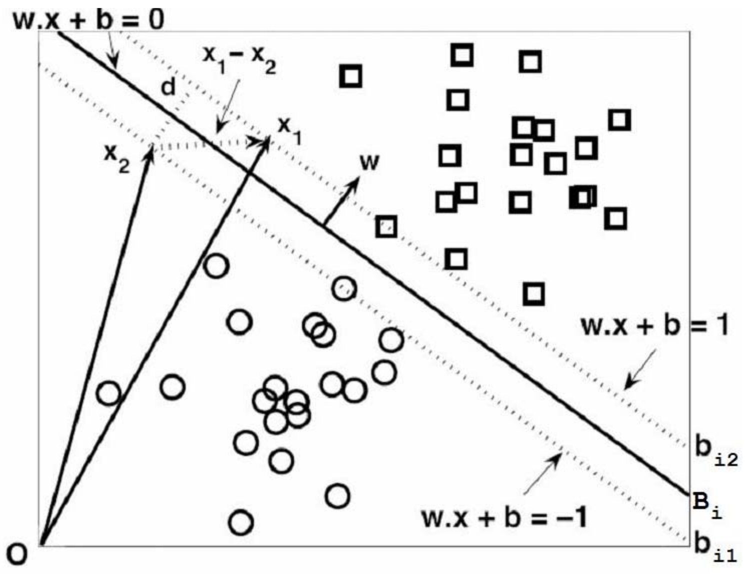
Illustration 5.5 SVM Two Margin Separating Hyperplanes [9]



Each hyperplane classifies the training examples with no errors. However, B_1 has a larger margin than B_2. Clearly, some of the data is closer to the hyperplane defined by B_2 than the hyperplane defined by B_1. As a result, there are some data that will more likely be misclassified due to slight perturbations in the hyperplane defined by B_2 than those classified by B_1. Therefore, by choosing a classifier that maximizes the distance to each training example from the hyperplane, it is less likely to cause classification errors due to small perturbations in the hyperplane.

To find the maximal margin separating hyperplane, consider a binary classifier (Illustration 5.6). Then, each set of data consists of the $(\vec{x}_i, y_i)$ pair for each i=1,2,…,N, where $\vec{x}_i = (x_{i1}, x_{i2}, ..., x_{id})^T$ is the i^{th} example attribute set and $y_i \in (-1,1)$. A hyperplane passing through the data points, $\vec{x}_i$, can be defined by the equation $\vec{w} * \vec{x} + b = 0$, where $\vec{w}$ and b are the parameters such that $\vec{w}$ is normal to the hyperplane and b is the offset of the hyperplane from the origin along $\vec{w}$. Both $\vec{w}$ and b should be chosen to separate the data into distinct classes and maximize the margin separating the data.

Illustration 5.6 SVM Maximum Margin Separating Hyperplane



Consider the two hyperplanes, b_{i1} and $b_{i2,}$ parallel to the hyperplane B_i. Then b_{i1} and b_{i2} can be described by the following equations.

$$b_{i2}: \vec{w} * \vec{x} + b = 1 \quad (1)$$

$$b_{i1}: \vec{w} * \vec{x} + b = -1 \quad (2)$$

Given two data points $\vec{x}_a$ and $\vec{x}_b$ on the hyperplanes, b_{i2} and b_{i1}, respectively, then, substituting the points into equations (1) and (2), yields

$$\vec{w} * \vec{x}_a + b = 1$$

$$\vec{w} * \vec{x}_b + b = -1$$

Then, subtracting both equations, we get

$$\vec{w} * \vec{x}_a - \vec{w} * \vec{x}_b = 2$$

$$\vec{w} * \vec{x}_a - 1 + d * \|\vec{w}\| + b = 2$$

$$\vec{w} * \vec{x}_a + b - 1 + d * \|\vec{w}\| = 2$$

$$1 - 1 + d\|\vec{w}\| = 2$$

$$d * \|\vec{w}\| = 2$$

$$d = \frac{2}{\|\vec{w}\|}$$

Any data example x_i located above the decision boundary can be written as

$$\vec{w} * \vec{x}_i + b = k, \text{ where } k > 0.$$

Any data example x_i located below the decision boundary can be written as

$$\vec{w} * \vec{x}_i + b = k' \text{where } k < 0.$$

By classifying all the data examples above the decision boundary as +1 and all the data examples below it as -1, then the class label y can be determined as

$$y = \begin{cases} 1\ if\ \vec{\mathrm{w}} * \vec{\mathrm{x}}_{\mathrm{i}} + \mathrm{b} > 0 \\ -1\ if\ \vec{w} * \vec{x}_i + b < 0 \end{cases}$$

5.2.2 Training the Support Vector Machine

Given a set of training data, we can estimate $\vec{w}$ and b of the decision boundary. The parameters are to be chosen such that

$$\vec{w} * \vec{x}_i + b \geq 1, \text{ if } y_i = 1$$

$$\vec{w} * \vec{x}_i + b \leq 1, \text{ if } y_i = -1$$

By choosing the parameters $\vec{w}$ and b based on the above conditions, we can summarize the conditions with $y_i(\vec{w} * \vec{x}_i + b) \geq 1,\ i = 1,2, \dots N$ If x_i is chosen on or above $\vec{w} * \vec{x} + b = 1$, the training instance belongs to class $y = 1$. If x_i is chosen on or below $\vec{w} * \vec{x} + b = -1$, the training instance belongs to class $y = -1$

Support vector machines impose the requirement that the margin, d , must be maximized. The maximization can be rewritten into the equivalent minimization problem defined by the objective function: $f(\vec{w}) = \frac{\|\vec{w}\|^2}{2}$.The minimization problem can be solved with the trivial solution such that $\vec{w} = 0$. Therefore, by constraining the optimization problem to include $y_i(\vec{w} * \vec{x}_i + b) \geq 1\ i = 1,2, \dots N$.Then, the linear separable case becomes

$$\min_{\vec{w}} \frac{\|\vec{w}\|^2}{2}$$

Subject to $y_i(\vec{w} * \vec{x}_i + b) \geq 1\ i = 1,2, \dots N$.

This problem belongs to the family of Quadratic Programming Problems. The optimization problem is quadratic or convex while the constraints are linear in form.

Using the Lagrange Multiplier Method for solution, rewrite the objective function with the constraint. The new objective function called the Lagrangian Optimization Problem becomes

$$L_P = \frac{1}{2}\|\vec{w}\|^2 - \sum_{i=1}^{N} \lambda_i \left(y_i(\vec{w} * \vec{x}_i + b) - 1\right)$$

where λ_i's are the Lagrangian multipliers. Rewriting using the Lagrangian multipliers ensures that feasible solutions for $\vec{w}$ and b can be found. Infeasible solutions will increase λ_i only. Because λ_i is unknown, $\vec{w}$ and b cannot be minimized and solved as

$$\frac{\partial L_P}{\partial \vec{w}} = 0$$

$$\vec{w} = \sum_{i=1}^{N} \lambda_i y_i \vec{x}_i$$

$$\frac{\partial L_P}{\partial b} = 0$$

$$\sum_{i=1}^{N} \lambda_i y_i = 0$$

By applying the Karush-Kuhn-Tucker (KKT) conditions

$$\lambda_i \geq 0$$

$$\lambda_i[y_i(\vec{w} * \vec{x}_i + b) - 1] = 0$$

solving for $\vec{w}$ and b become possible. All $\lambda_i = 0$ such that $y_i(\vec{w} * \vec{x}_i) + b = 1$. Where $\lambda_i > 0$ lies along the hyperplanes b_{i1} and b_{i2}, thus providing the support vectors.

Despite application of the KKT conditions, the above optimization problem can be very difficult to solve due to the large number of parameters. The problem can be simplified. By applying $\frac{\partial L_P}{\partial \vec{w}}$ and $\frac{\partial L_P}{\partial b}$ to L_P, we get

$$L_D = \frac{1}{2}\left\|\sum_{i=1}^{N} \lambda_i y_i \vec{x}_i\right\| - \sum_{i=1}^{N} \lambda_i y_i \vec{x}_i * \sum_{i=1}^{N} \lambda_i y_i \vec{x}_i + b\sum_{i=1}^{N} \lambda_i y_i + \sum_{i=1}^{N} \lambda_i.$$

This reduces to

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2}\sum_{i,j} \lambda_i \lambda_j y_i y_j \vec{x}_i * \vec{x}_j$$

Upon finding all $\lambda_i's$, $\vec{w}$ and b can be found by

$$\vec{w} = \sum_i \lambda_i y_i \vec{x}_i$$

$$\lambda_i \left[y_i \left(\sum_j \lambda_j y_j \vec{x}_j * \vec{x}_i + b \right) - 1 \right] = 0$$

Once support vector parameters have been found, query points can be classified by

$$f(\vec{z}) = sign(\vec{w} * \vec{z} + b) = sign\left(\sum_{i=1}^{N} \lambda_i y_i \vec{x}_i * \vec{z} + b \right).$$

If $f(\vec{z}) = -1$, then $\vec{z}$ is in the negative class. Otherwise, it is classified as positive.

5.2.3 Kernel Methods for the Non-Linearly Separable Support Vector Machine

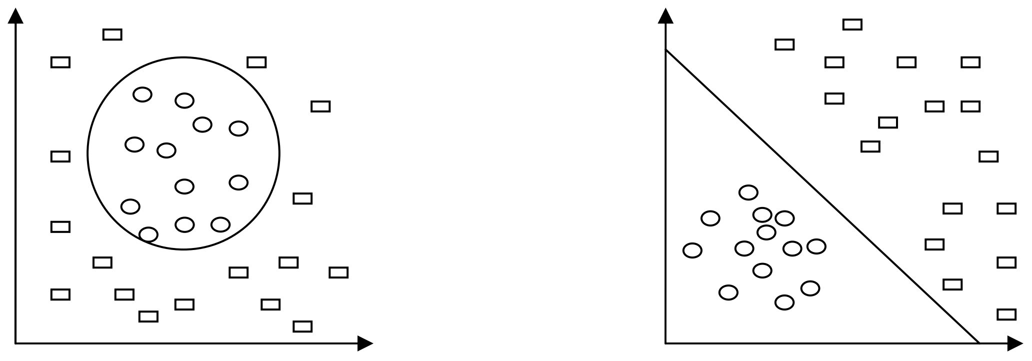
Vapnick's implementation of Support Vector Machines handled only linearly separable data. However, through the use of kernel methods, an implementation to classify non-linear problems became possible. The QP problem becomes

$$\min_{\vec{w}} \frac{\|\vec{w}\|^2}{2}$$

Subject to $y_i(\vec{w} * \phi(\vec{x}_i) + b) \geq 1,\ i = 1,2, \ldots N$

Where $\phi(\vec{x}_i)$ is a transformation from a non-linear space to a linear space (Illustration 5.7). The transformation can be a higher dimensional space than the linear space.

Illustration 5.7 Non-Linear Space & Transformation to a Linear Space



By the same process above, the Lagrangian becomes

$$L_D = \sum_{i=1}^{N} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \phi(\vec{x}_i) * \phi(\vec{x}_j).$$

Upon finding all λ_i's, $\vec{w}$ and b can be found by

$$\vec{w} = \sum_i \lambda_i y_i \phi(\vec{x}_i)$$

$$\lambda_i \left[y_i \left(\sum_j \lambda_j y_j \phi(\vec{x}_j) * \phi(\vec{x}_i) + b \right) - 1 \right] = 0.$$

Similarly, the classification of a query point follows from

$$f(\vec{z}) = sign(\vec{w} * \vec{z} + b) = sign\left(\sum_{i=1}^{N} \lambda_i y_i \phi(\vec{x}_i) * \phi(\vec{z}) + b \right).$$

Solving in the transformed space can prove to be problematic due to mapping into a higher dimensional transformation space. Mercer's Theorem guarantees any positive, semi-definite, continuous, and symmetric kernel function, $K(X, Z)$, can always be represented as the dot product $\langle \phi(\vec{w}), \phi(\vec{z}) \rangle$ in a high-dimensional transform space, without specific knowledge of the form of the transformation. That is, if K, a positive definite function, $K(\vec{w}, \vec{z}) = \langle \phi(\vec{w}), \phi(\vec{z}) \rangle$.

There are two advantages of interest to calculating in the original space over the transform space. First, without learning the specific form of the transformation, computational time is saved. Second, there is no curse of dimensionality problem.

There are many different kernel methods to use. The three most popular are the

Polynomial: $(\vec{x} * \vec{y} + 1)^P$

Gaussian: $e^{-\frac{\|\vec{x}-\vec{y}\|^2}{2\sigma^2}}$

Sigmoid: $\tanh\,(k\vec{x} * \vec{y} - \sigma)$

This research will utilize the Gaussian kernel since no polynomial assumption is made about the relationship of the features. The Gaussian kernel has one parameter, δ, that must be tuned.

Chapter 6 Discriminant Function for Normal Density

Given the discriminant function $g_i(\vec{x}) = \ln p(x|\omega_i) + p(\omega_i)$, it can be evaluated if $p(\vec{x}|\omega_i) \sim N(\mu_i, \sum_i)$. The general multivariate normal density in d dimensions is given by $p(\vec{x}) = \frac{1}{(2\pi)^{\frac{d}{2}}|\sum|^{\frac{1}{2}}} exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma^{-1}(\vec{x}-\vec{\mu})\right]$, x is a d component column vector, μ is a d component mean vector, and Σ and Σ^{-1} is a dxdcovariance matrix and inverse, respectively. Finally, $|\Sigma|$ is the determinant.

Then,

$$g_i(\vec{x}) = \ln p(x|\omega_i) + p(\omega_i)$$

$$\ln\left[\frac{1}{(2\pi)^{\frac{d}{2}}|\sum|^{\frac{1}{2}}} exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma^{-1}(\vec{x}-\vec{\mu})\right]\right] + \ln p(\omega_i)$$

$$= \ln\frac{1}{(2\pi)^{\frac{d}{2}}|\sum|^{\frac{1}{2}}} + \ln exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma^{-1}(\vec{x}-\vec{\mu})\right] + \ln p(\omega_i)$$

$$= \ln 1 - \ln(2\pi)^{\frac{d}{2}}|\Sigma_\mathrm{i}|^{\frac{1}{2}} - \frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma_\mathrm{i}^{-1}(\vec{x}-\vec{\mu}) + \ln p(\omega_i)$$

$$= -\ln(2\pi)^{\frac{d}{2}} - \ln|\Sigma_\mathrm{i}| - \frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma_\mathrm{i}^{-1}(\vec{x}-\vec{\mu}) + \ln p(\omega_i)$$

$$= -\frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_\mathrm{i}| - \frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma_\mathrm{i}^{-1}(\vec{x}-\vec{\mu}) + \ln p(\omega_i)$$

$$= -\frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma_\mathrm{i}^{-1}(\vec{x}-\vec{\mu}) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln p(\omega_i)$$

If the covariance matrices are different for all classes, then $\frac{d}{2}\ln(2\pi)$ is the only term that is independent of i. Therefore, it can be eliminated as it provides unimportant additive constants.

Then, the equation becomes

$$= -\frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma_\mathrm{i}^{-1}(\vec{x}-\vec{\mu}) - \frac{d}{2}\ln(2\pi) - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln p(\omega_i)$$

$$= \frac{1}{2}(\vec{x}-\vec{\mu})^t\Sigma_\mathrm{i}^{-1}(\vec{x}-\vec{\mu}) - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln p(\omega_i)$$

$$= -\frac{1}{2}[\vec{x}^t\Sigma_\mathrm{i}^{-1}\vec{x} - \vec{x}^t\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}} - \vec{\mathrm{\mu}}^\mathrm{t}\Sigma_\mathrm{i}^{-1}\vec{x} + \vec{\mathrm{\mu}}^\mathrm{t}\Sigma_\mathrm{i}^{-1}\vec{\mu}] - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln p(\omega_i)$$

$$= -\frac{1}{2}[\vec{x}^t\Sigma_\mathrm{i}^{-1}\vec{x} - (\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}})^t\vec{x} - (\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}})^t\vec{x} + \vec{\mu}^t\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}}] - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln p(\omega_i)$$

$$= -\frac{1}{2}[\vec{x}^t\Sigma_\mathrm{i}^{-1}\vec{x} - 2(\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}})^t\vec{x} + \vec{\mu}^t\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}}] - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln p(\omega_i)$$

$$= -\frac{1}{2}\vec{x}^t\Sigma_\mathrm{i}^{-1}\vec{x} + (\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}})^t\vec{x} - \frac{1}{2}\vec{\mu}^t\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}} - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln \mathrm{p}(\omega_\mathrm{i})$$

$$g_i(\vec{x}) = \vec{x}^t W_i \vec{x} + \vec{w}^t\vec{x} + w_{i0}$$

$$W_i = -\frac{1}{2}\Sigma_\mathrm{i}^{-1}$$

$$\vec{w_i} = \Sigma_\mathrm{i}^{-1}\vec{\mu}_i$$

$$w_{i0} = -\frac{1}{2}\vec{\mu}^t\Sigma_\mathrm{i}^{-1}\vec{\mathrm{\mu}} - \frac{1}{2}\ln|\Sigma_\mathrm{i}| + \ln \mathrm{p}(\omega_\mathrm{i})$$

Therefore, the discriminant function is quadratic in form.

In this paper, it is assumed the data do not have equal covariance matrices for each class. Therefore, I consider unequal Σ_i for each class is not relaxed. Furthermore, the dataset has more than two classes. Therefore, the data seems more suited for a non-linear discriminant analysis classifier.

Chapter 7 Random Forest

Random Forest (RF) is a classification algorithm that employs the use of decision tree classification [13]. This research uses supervised learning for classification although the algorithm can perform unsupervised learning. RF builds a forest of decision trees utilizing the CART algorithm for decision tree induction. Though the classification is produced from a forest of most likely sub-optimal decision trees, the classification algorithm is highly accurate. There are other benefits to the Random Forest. It does not suffer from the curse of dimensionality, it can estimate variable importance, it produces an unbiased estimate of the generalization error similar to cross validation without the need for a separate test set, it provides a method for detecting variable interactions, and finally learning a classifier is relatively fast.

Random Forest proceeds through tree building using the CART algorithm. Given a set of training points $T = \{(\vec{x}_i, y_i) | x_i \in \mathbb{R}^m, y_i \in \mathbb{Z}\}$, where $\vec{x}_i$ is the feature space and y_i is the response, the classification tree portion of CART will perform a recursive binary partitioning of the entire dataset until some stopping criteria is met. Partitioning into regions $R_1, R_2, \dots, R_m$ is based on the feature space of T where the locally optimal splitting variable and splitting points for the variable are found. Tree topology, though likely suboptimal due to local optimization, is controlled by the split variables and split points.

Each region R_m classifies the points by $f(x) = \text{argmax}_k \{p_{mk} = \sum_{x_i \in R_m} I(y_i = k)\}$, where p_{mk} is the proportion of points in R_m belonging to class k. Best splits are measured based on a node impurity measure. CART uses the Gini Index,

$$Gini(m) = \sum_{i=1}^{k} p(i|m)$$

Smaller Gini values indicate a purer data split than larger ones. Performance of the data splits are measured by comparing the degree impurity of the parent node with the degree of impurity of the child nodes. This is accomplished as

$$I(Parent) - \sum_{j=1}^{k} \frac{N(v_j)}{N} I(v_j)$$

where $I(v_j)$ is the Gini for node j, N is the total number of data in the parent node, and $N(v_j)$ is the number of data associated with child node j.

CART proceeds through recursive binary splits to create $R_1, R_2, \ldots, R_m$ regions. Although multi-way splits are possible, CART utilizes only binary splits. By creating multi-way splits, data can become fragmented producing insufficient data for training latter splits. Furthermore, since performing binary splits in series simulates multi-way splits while maintaining data integrity, it is the preferred method for tree building. Finally, CART trees require stopping criteria to halt tree building and a method for tree pruning. CART utilizes Cost Complexity Analysis with cross validation to prune decision trees. Since Random Forest utilizes fully grown trees, these will not be discussed further here.

The Random Forest Algorithm is simple in design.

Input: Training Set $T = \{(\vec{x}_i, y_i) | \, \vec{x}_i \in \mathbb{R}^m, y_i \in \mathbb{Z}\}$

Testing Set Q= $\{(\vec{x}_i', y_i) | \, \vec{x}_i' \in \mathbb{R}^n, y_i \in \mathbb{Z}\}$

K= number of trees to grow, m split variables

Draw $n_i, i = 1 \ldots K$, bootstrap samples with replacement.

For each n_i sample, grow a fully grown, un-pruned, tree using CART. But modify the best split of all features to be best split of $m < M$ randomly selected features , where m remains constant for each split.

Output: Predicted class for each point in Q based on the modal class of the n_i trees.

The algorithm provides a means of estimating the prediction error rate.
The predicted generalization error is determined by withholding about one-third of the training samples, referred to as out-of-bag (oob) cases for each tree. Each tree is grown with a different bootstrap sample, hence, a different set of oob cases. Prediction is performed by running the k^{th} oob cases down k^{th} tree to obtain a classification. At the end of the run, the proportion of times the j^{th} case was oob and not equal to the true class averaged over all cases is the out-of-bag error estimate.

Feature subset selection estimates are taken from variable importance. It gives a good measure of the variables that contributed the most to the classification of each case. To calculate, perform two runs of the k^{th} oob cases down the k^{th} tree. The oob cases which are classified correctly during the first run are compared against an m variable permuted case which are classified correctly on the second run. These cases are subtracted then averaged over all trees to get a measure of variable importance. The standard error is calculated as a z-score and significance is assigned based on the assumption of normality.

Chapter 8 Experimental Results

8.1 Classification Assessment

The classification system used consists of three species of *Carpiodes*: *C.carpio*, *C.velifer*, and *C.cyprinus*. Since the dataset is relatively small with 651 samples, there will only be a training set, 75% of the dataset, and a test set, 25% of the dataset. The data are distributed as follows in Table 8.1.

Table 8.1

***Carpiode*s Data Set**

Testing & Training Split

25% Testing 75% Training

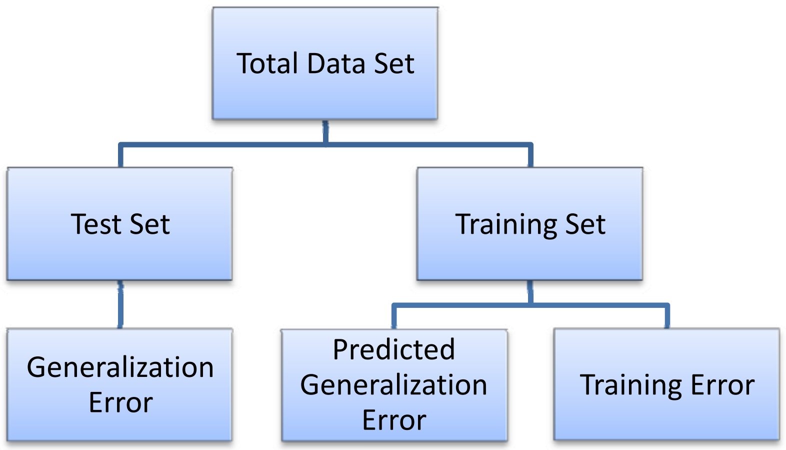
Species	***C.carpio***	***C.velifer***	***C.cyprinus***
Total	181	172	298
	Sets		
Training		**Testing**	
488		163	

Misclassification is equally important with respect to all classes. Therefore, all misclassification will be treated the same and grouped as misclassification error. The classification models are assessed based on the generalization error on a test set, prediction of the generalization error, henceforth called prediction error, from a training set either by five-fold cross validation or in the case of Random Forest, prediction error is based on the out of bag error estimate, and training error.

Prediction error, although left out of model building at the time of testing, will be used at some point in the design of the model. Therefore, it will be used for a prediction, or estimate, of the generalization error.

Generalization error is the misclassification error resulting from test data not used in the model building process. The test data are previously unseen by the classification model. High generalization error indicates the model cannot accurately predict on unseen data. Training error is the re-substitution of the training data into the model in which it was used for model building. Low training error can be an indication of model over-fitting if the test or predictive errors are very high. Assessment and selection of the appropriate models will require low generalization, predictive, and testing error (Figure 8.1).

Figure 8.1 Types of Classification Error

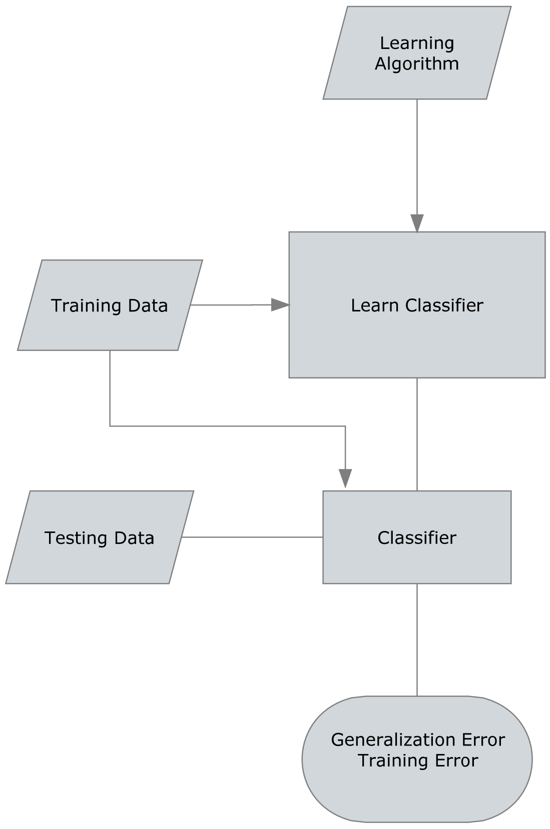


Model building and assessment utilizes both training and testing error.

Analysis proceeds by two methods: classification by multi-class data splits and recursive binary splits. To achieve binary splits, two species are combined as one set while the third species is held separate. In multi-class data splits, all three data sets are held separately.

A validation set will not be used. Instead training and testing will proceed along a five-fold cross validation method for K-Nearest Neighbors, Support Vector Machines, and Quadratic Discriminant Analysis. The Random Forest has methods built into the algorithm to generate predictive error. Again, this dataset is not very large. Therefore, ten-fold cross validation will not provide enough training and testing cases for the models. But, the use of leave one out cross validation is not practical as the dataset is large enough to provide sufficient testing and training sets. Figure 8.2 below indicates how the process proceeds.

Figure 8.2 Process of Classification

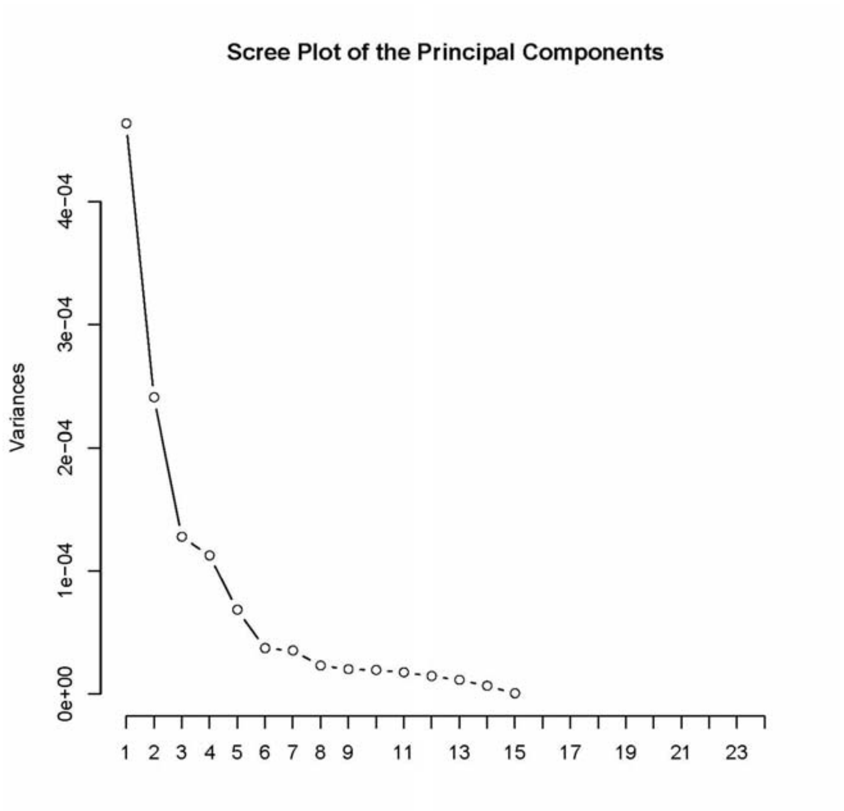


8.2 Unsupervised Learning Methods

8.2.1 Principal Component Analysis

The training set was analyzed by Principal Component Analysis to determine if the transformation can produce a new set of features that is reduced from the original fifteen but still produce good separation of the three species. By scree plot analysis in Figure 8.3, it appears that the feature set can be reduced.

Figure 8.3 Scree Plot of the Principal Components



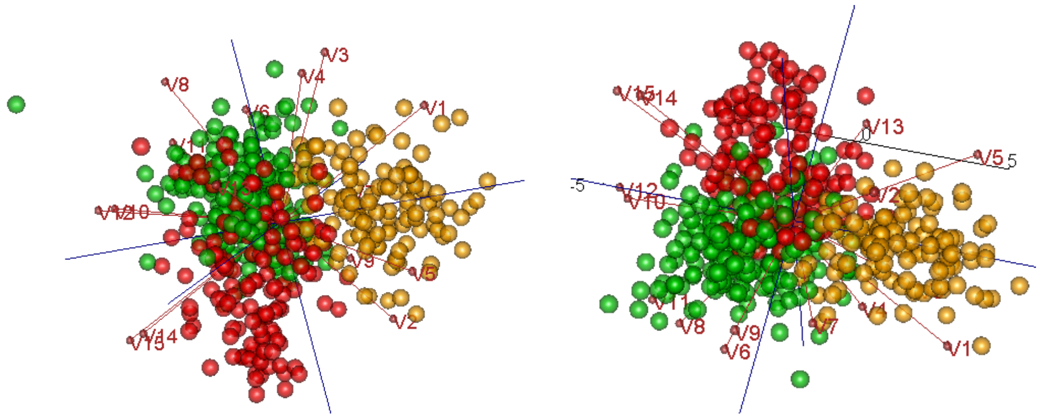
The first six principal components seem to accounts for most of the variance, about 87%. Therefore, the feature set can be significantly reduced.

The graph in Figure 8.4 illustrates the first three principal components. They do indicate that there is some data separation between the three classes.

Furthermore, there is good data separation between one cluster of data versus the remaining clusters. However, there is more overlap between two clusters. Classification of these non-linearly separated groups will produce misclassification error more often.

Figure 8.4

Graph of the First 3 Principal Components



8.2.2 Hierarchical and K-Means Clustering

The data were analyzed through unsupervised clustering to determine the natural clustering of the data. Without some natural grouping of the data, it is highly likely that the supervised learning algorithms would fail as well.

Hierarchical clustering with three clusters was used to analyze the grouping of the data. The graph in Figure 8.5 indicates a natural clustering of the data using three classes. Furthermore, K-Means Clustering, Table 8.2, indicates the natural grouping of the data. However, two groups clustered almost evenly. But, one group dominated with about one hundred more group members. This could be the result of an unbalanced sample size. Hence, there may be no statistical significance to the unbalanced grouping.

Figure 8.5 Hierarchical Clustering

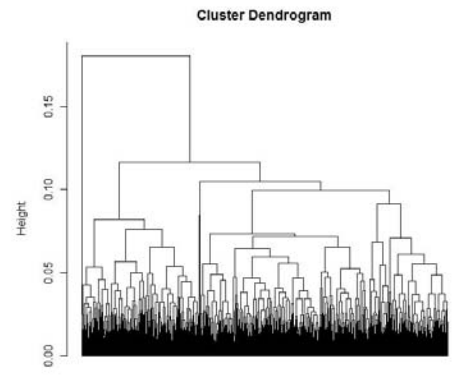


Table 8.2 K-Means Clustering

K-Means Clustering 3 Groups		
Group 1	**Group 2**	**Group 3**
127	222	139

8.3 Multi-class Assessment Methods

Multi-class assessment was performed on each classification method. K-nearest neighbors required tuning K to find the optimal nearest neighbor set. By ranging K from 1 to 25, the 9 nearest neighbors are optimal for separating all three classes. Although Figure 8.6 below indicates similar nearest neighbor classification error as does K = 9, I utilized the smallest nearest neighbor set for training. Table 8.3 contains all the results of the multi-class methods.

Figure 8.6 Tuning Phase of the K-Nearest Neighbor Algorithm

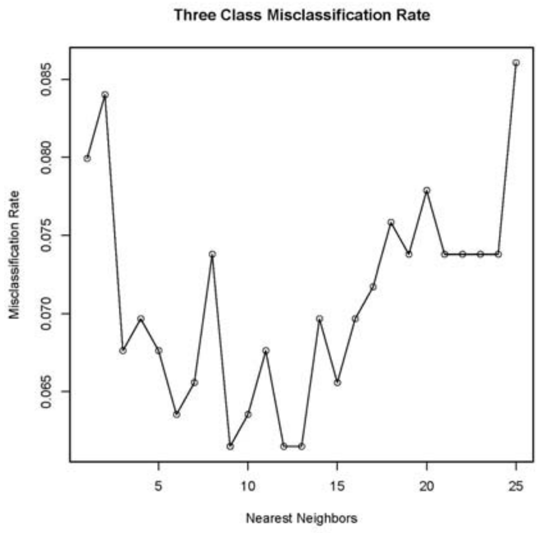


Table 8.3

Classification Errors on Four Classification Algorithms

Multi-Class Split

Classification Algorithm	**Predicted Error**	**Training Error**	**Generalization Error**
KNN **K = 9**	6.1%	4.5%	7.4%
SVM $\boldsymbol{\sigma = 0.01}$ **Support Vectors = 59**	1.4%	1.0%	4.9%
Random Forest **3 Variable Splits** **500 Trees**	5.9%	0%	6.7%
QDA	3.9%	1.8%	6.1%

8.4. Binary Split Assessment Methods

Binary split assessment proceeds by holding one class separate while combining the remaining two classes. Possible combinations are *C.carpio* vs *C.velifer* & *C.cyprinus*, *C.velifer* vs *C.carpio* & *C.cyprinus*, or *C.cyprinus* vs *C.carpio* & *C.velifer*. After classification through this level 1 binary split, further classification of the combined class is obtained by a level 2 recursive binary split. Tables 8.4 – 8.5 below show the results of the recursive binary split for each of the four algorithms used.

The K-Nearest Neighbor algorithm tuning indicates the best neighborhood, K value, for level 1 was $K = 7$ and level 2 was $K = 2$ (Figure 8.7).

Figure 8.7 Tuning Phases of the K-Nearest Neighbor Algorithm

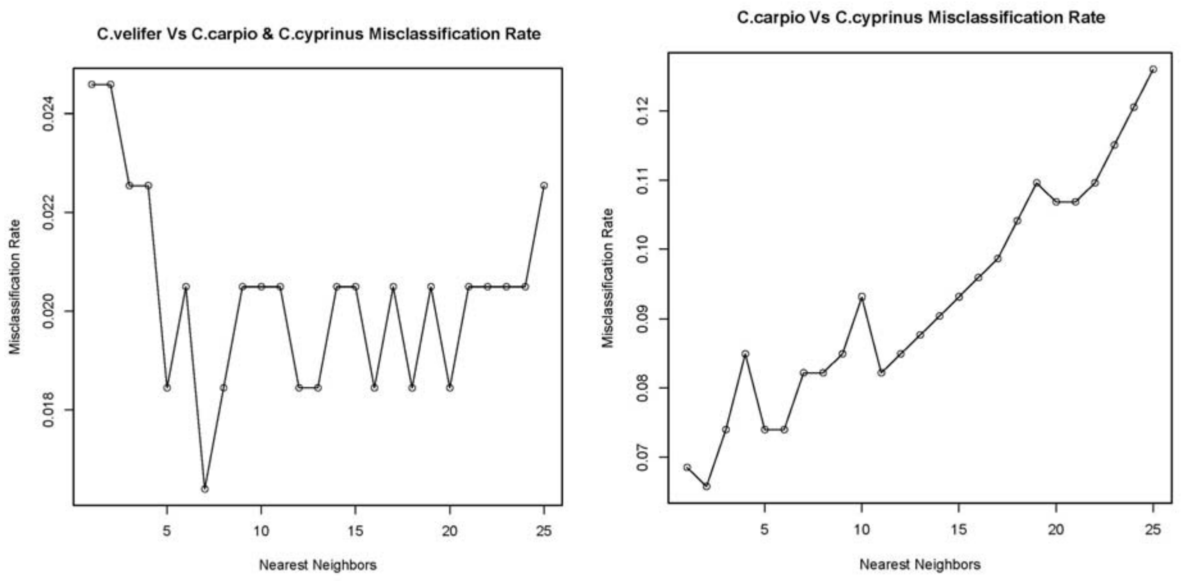


Table 8.4

Recursive Binary Split

Level 1 & Level 2

All Four Algorithms

Algorihm	Predicted		Train		Test	
	Level 1	**Level 2**	**Level 1**	**Level 2**	**Level 1**	**Level 2**
KNN	1.6%	6.6%	1.6%	3.3%	1.8%	7.9%
SVM $\boldsymbol{\sigma = 0.01}$ SV = 97 Level 1 SV = 127 Level 2	1.4%	5.3%	1.4%	5.3%	3.1%	8.1%
RF	2.5%	6.5%	0%	0%	3.1%	10.4%
QDA	1.2%	2.3%	0.6%	1.4%	3.1%	3.8%
SV = Support Vectors						

Table 8.5

Recursive Binary Split

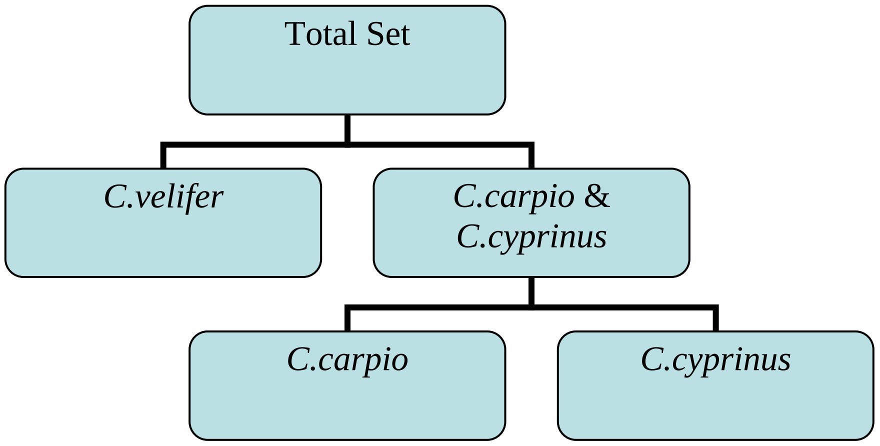
Total Error

All Four Algorithms

Algorithm	**Predicted**	**Train**	**Test**
KNN	8.2%	4.9%	9.7%
SVM	6.7%	6.7%	11.2%
RF	9.0%	0%	13.5%
QDA	3.5%	2.0%	6.9%

Each algorithm consistently chose the level 1 split to be *C.velifer* vs. *C.carpio*/*C.cyprinus* then recursively *C.carpio* vs. *C.cyprinus* for level 2 split (Figure 8.8).

Figure 8.8 Recursive Binary Split for Classification Algorithms

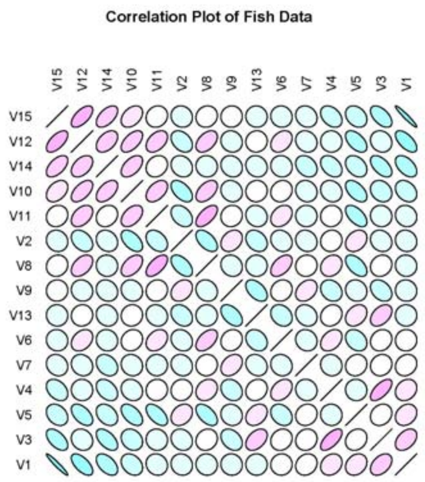


8.5 Feature Reduction

Feature reduction is a very important component to this classification study. Features can affect model over-fitting. Therefore, the generalization error will increase while decreasing the strength of the model. As a result, feature reduction can benefit model complexity and performance. Fifteen Fourier Descriptors were originally used for classification. Although, in many respects, fifteen features may be a sufficiently small set of features for classification, reduction was analyzed to determine if a smaller subset of the fifteen features could still sufficiently produce accurate classification.

Preliminary analysis of the correlation graph matrix in Figure 8.9 indicates some variable redundancy. Therefore, feature reduction should be implemented to find a subset of features that provides low misclassification error while reducing variable redundancy. Furthermore, Principal Component Analysis in Section 8.2.1 indicates that grouping data by fewer components is possible.

Figure 8.9 Covariance Plot of the Training Set

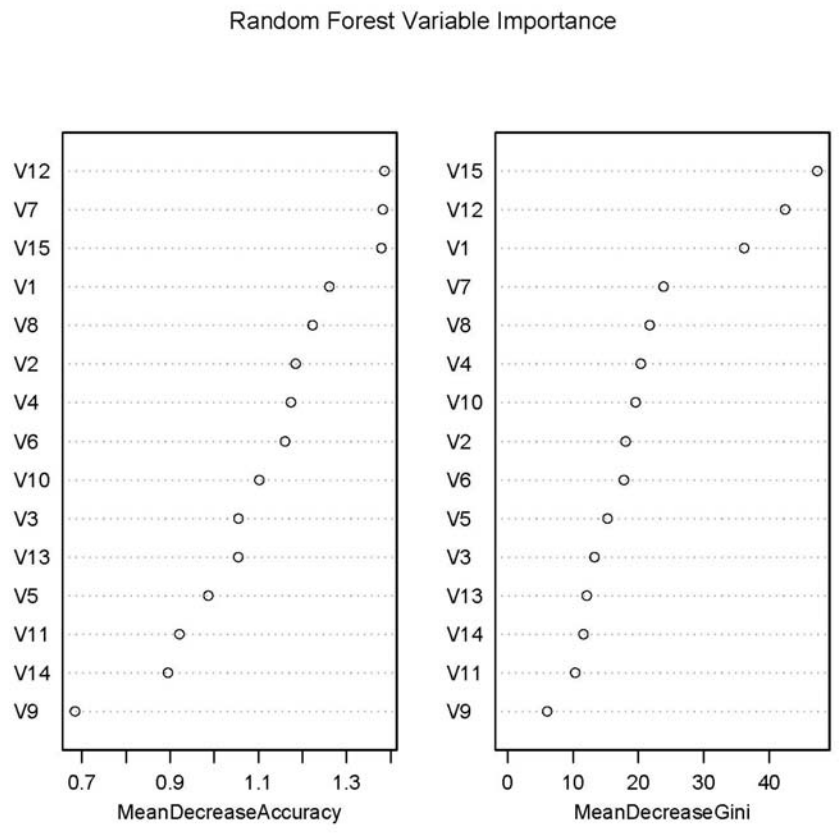


Feature reduction techniques fall into two general classes. The filter approach achieves feature reduction prior to algorithm implementation. The alternative, the wrapper approach, wraps feature reduction into the classification algorithm itself [14]. Techniques for feature reduction used in this analysis are Principal Component Analysis, based on the filter approach, and variable importance in the Random Forest algorithm, based on the wrapper approach.

8.5.1 Variable Importance Using the Random Forest

Examining the variable importance feature of the Random Forest algorithm indicates that feature reduction is possible. Following along the lines of Principal Component Analysis, only the six most important variables were retained (Figure 8.10).

Figure 8.10 Variable Importance Graphs Using Random Forest Algorithm



The graphs in Figure 8.10 are sorted by decreasing order of variable importance. There was some minor disagreement with respect to Mean Decrease Accuracy and Mean Decrease in Gini. However, by using Mean Decrease in Accuracy, the most important variables appear to be FD1, FD2, FD7, FD8, FD12, & FD15.The Random Forest algorithm was then again run on the six most important variables. The out of bag error estimate for generalization error did increase from the original full feature set. Furthermore, generalization error remained higher than predicted error and training error. However, test classification was still good as indicated by Table 8.8.

Table 8.6

Feature Reduction
Random Forest Algorithm
Using the 6 Most Important Descriptors
3 Variable Training Splits
FD1, FD2, FD7, FD8, FD12, & FD15

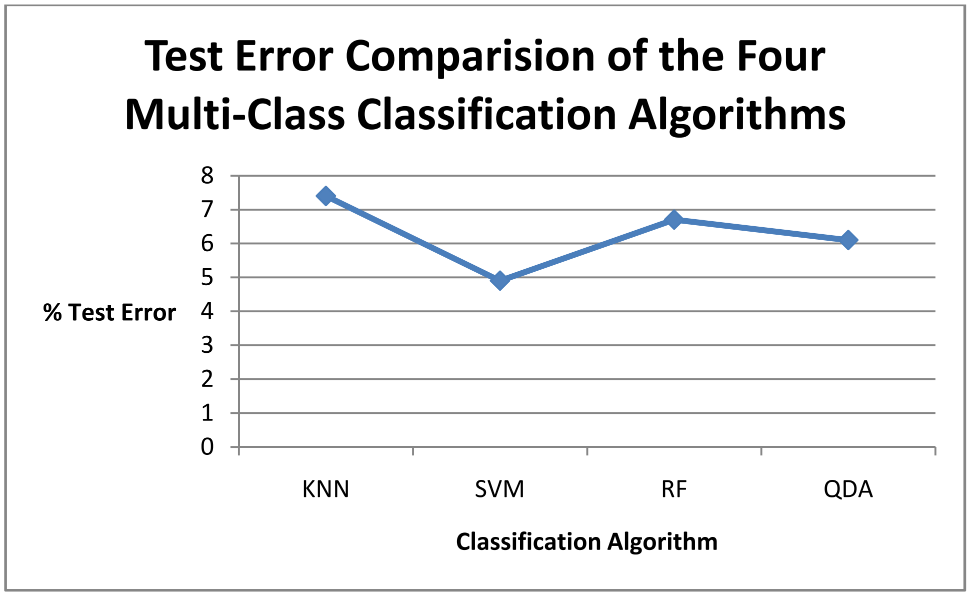
Misclassification Error	**Misclassification**	
	Reduced	**Original**
Predicted	7.0%	5.9%
Test	9.2%	6.7%
Train	0%	0%

Chapter 9 Discussion

Unsupervised learning methods each showed that there is a natural clustering to the data. This is important to the study since failure at this level will propagate into supervised methods. Principal Component Analysis using the first three principal components indicates the data groups into three separate classes. Hierarchical and K-Means clustering further support three class separations in the data. Therefore, the successful use of supervised learning algorithms is possible.

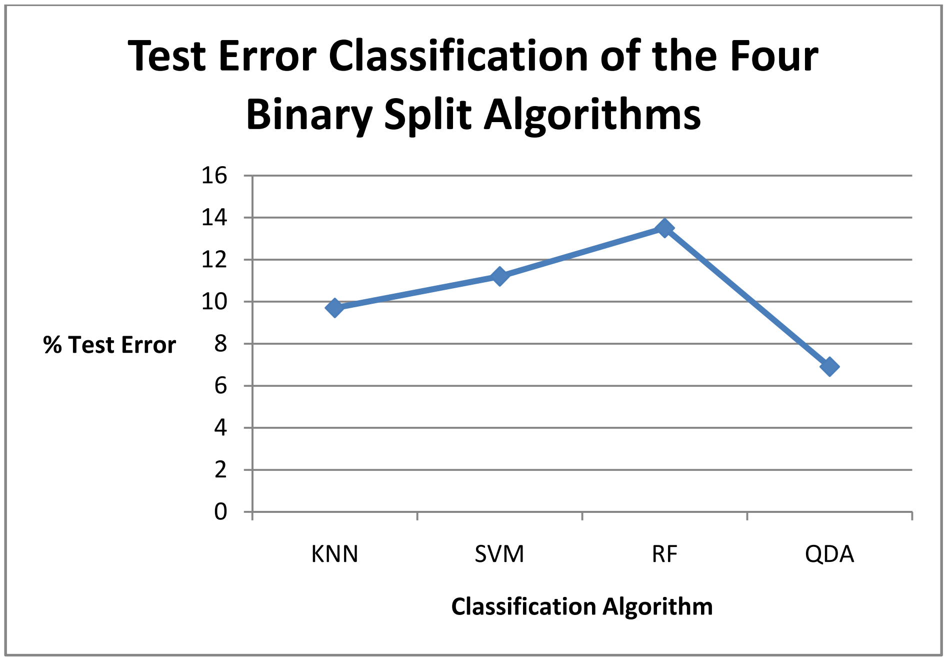
The multi-class classification methods did produce varying results (Figure 9.1). The Support Vector Machine algorithm produced the best results of all the methods while the K-Nearest Neighbor algorithm produced the worst, although the results were still very good. Whatever method used, misclassification error remained low and very similar to one another.

Figure 9.1 Test Error of Four Multi-Class Classification Algorithms



The binary split classification methods also produced varying, but acceptable, results (Figure 9.2). The Quadratic Descriminant Analysis algorithm produced the best results with the lowest misclassification error rates while the Random Forest produced the worst error rates.

Figure 9.2 Test Error of Four Recurisve Binary Split Classification Algorithms



Both multi-class and recursive binary splits produced very good classification results. Feature reduction showed promising results for the data set. By examination of the covariance plots, it can be seen that there are redundant features in the feature set. Analysis proceeded along two lines with each indicating that reducing the number of features is possible. By retaining the first six principal components that account for the most variance in the data and the six most important variables found by variable importance using the Random Forest algorithm, feature reduction can be achieved and it produces good classification results. Reusing the data through the Random Forest algorithm, the six most important variables produced good results for classification. Generalization error did increase and was greater than training error [16]. But, it was not at an unacceptable level.

Chapter 10 Conclusion

Four algorithms were used to analyze and classify the *Carpiodes* fish data set using fifteen Fourier Descriptors of known landmark data. Although there were some variations in the results, they were expected. The algorithms still produced very good classification results based on multi-class and recursive binary split classification.

Indications such as these can be used to develop a full featured classification system. That is, a query point can be submitted to a classification system that contains a sufficiently large database of known species and their landmark data. Then, through the techniques submitted here and possible future results, confidence in a correct classification can be obtained. As a result, species classifications can proceed at a much faster rate than they do by manual techniques that are still in use today.

Future study can proceed in many different ways. To begin with, text based features can be incorporated into the system to utilize many notes left on images by researchers. However, there must be more standardization with respect to text notations. One can utilize more descriptors than utilized in this study. However, I would not recommend using descriptors along the edge of the image as this was a study on Fourier Descriptors of known landmarks. Instead, utilizing the known landmarks, compound features can be generated and the Fourier Descriptor utilized. In all, there are many different possibilities for future study. This thesis does lay a foundation upon which to build.

References

[1] International Code of Zoological Nomenclature, 4th Edition. International Trust for Zooligical Nomenclature, c/o Natural History Museum, 1999.

[2] Chen, Bart, and Teng, “A Content-Based Image Retrieval System for Fish Taxonomy,” International Media Conference, 2005.

[3] Suttkus Royal D. and Bart, Jr., Henry L, “A Preliminary Analysis of the River Carpsucker, *Carpiodes Carpoi*, in the Southern Portion of its Range,” 2002.

[4] Primm, Stuart L. and Lawton, John H., “Ecology-Planning for Biodiversity,” Science, 2068-2068, 1998.

[5] Rui, Yong, Huang, Thomas S., and Chang, Shih-Fu, “Image Retrieval: Current Techniques, Promising Directions, and Open Issues,” 1999.

[6] Wheeler, Q.D, Raven, P.H., and Wilson, E.O., “Taxonomy: Impediment or Expediment?” Science, 303:285, 2004.

[7] Kherfi, M.L., Ziou, D., and Bernardi, A., “Image Retrieval from the World Wide Web: Issues, Techniques, and Systems,” 2004.

[8] Adams, Dean C., Rohlf, F. James, and Slice, Dennise E., “Geometric Morphometrics: Ten Years of Progress Following the ‘Revolution’,” Ital. J. Zool, vol. 71, pages 5-16, 2004.

[9] Tan, Pang-Ning, Steinbach, Michael, and Kumar, Vipin, Introduction to Data Mining, Boston, Pearson Education, Inc., pages 222, 225, 257-259, 2006.

[10] Zuo, W. M., Lu, W. G., Wang, K.Q., and Zhang, H., “Diagnosis of Cardiac Arrhythmia Using Kernel Difference Weighted KNN Classifier,” Computers in Cardiology, 35; pages 253-256, 2008.

[11] Lee, Yuchun, “Handwritten Digit Recognition Using K-Nearest Neighbor Radial-Basis Function and Back Propagation Neural Networks”, Neural Computation, Vol 3, Issue 3, Fall 1991, Pages 440 – 449, 1991.

[12] Baoli, Li, Shiven, You, and Qin, Lu, “An Improved K-Nearest Neighbor Algorithm for Text Characterization”.

[13] Breiman, Leo and Cutler, Adele, “Random Forests”, Machine Learning, 45, pages 5-32, Kluwer Academic Publishers, 2001.

[14] Guyon, Isabelle and Elisseeff, Andre, “Introduction to Variable and Feature Selection”, Journal of Machine Learning Research vol. 3, pages 1157–1182, 2003.

[15] Granlund, G.H., “Fourier Descriptors for Plane Closed Curve,” IEEE Trans. Computers, vol. C-21, pages 195-201, February 1972.

[16] Hastie, Trevor, Tibshirani, Robert, and Friedman, Jerome, The Elements of Statistical Learning Data Mining, Inference, and Prediction, Springer Series in Statistics, page 194, 2001.

[17] Vapnik, V., The Nature of Statistical Learning Theory, Springer-Verlag, New York, 1995.

Vita

Patrick J. Trahan was born in Chauvin, Louisiana, a small town 60 miles southwest of New Orleans, Louisiana. He graduated from Nicholls State University in Thibodaux, Louisiana in 1995 with a Bachelor’s Degree in Psychology. Opting for more education, he returned to Nicholls State University graduating in 1998 with a Master of Science Degree in Applied Mathematics. He began teaching at Nicholls State University in 1998 as a Lecturer of Mathematics. In 1999, he moved on to the field of Information Technology, working for Louisiana State University Health Sciences Center in New Orleans, where he remains to this date. Also, he is currently an adjunct instructor teaching classes in Mathematics for Nicholls State University and Computer Technology classes for L.E. Fletcher Technical Community College in Houma, Louisiana. He currently resides in Houma, Louisiana with his wife, Angie, and two children, Alexis Grace, and Brittany Nicole.