Part III

Part III

Chapter 9

A Technique for Semi-Supervised Learning using Non-Linear Dimensionality Reduction

This chapter describes the process of classification through Manifold or Topological learning. Manifold learning is a dimensionality reduction process of estimating a low-dimensional structure which underlies a collection of high-dimensional data. In this chapter, we present a semi-supervised learning technique that uses non-linear manifold learning methods for dimensionality reduction and then applying various classification techniques for prediction. We adopt an spatial autocorrelation derived approach for selecting the best value for K during the neighborhood graph creation phase. We show the advantages of this technique in terms of classification accuracy, reduced dimensions and complexity based on comparison with other classification techniques. Furthermore, we use extreme learning machines for predicting unseen data in order to improve complexity.

9.1 Dimensionality Reduction and Manifold Learning

The mining and exploration of data is necessary in order to find interesting and deeper relationships between variables. This can be done by visual methods because of the power of the human eye to detect structures. However, most problems of interest in data min-

ing involve data with a large number of dimensions thus, it becomes necessary to reduce the number of dimensions of the dataset by applying some dimensionality reduction techniques. Dimensionality reduction is a process of finding a small set of features to describe a large set of observed dimensions. Besides the obvious fact that reducing the number of dimensions makes it easier to visualize the data (we cannot easily view data in more than 3 dimensions), dimensionality reduction is also useful in discovering a compact representation, thus decreasing the computational processing time (since a small data set is much faster to process than a large one). In addition, it may serve to separate the important features from the less important ones and it can also be a preprocessing for supervised learning i.e simplify, reduce, and clean the data for subsequent supervised training. Furthermore, the generated simpler models due to this type of pre-processing makes them much more robust than comparatively more complex models.

Among the core existing dimensionality reduction methods, X.Huo et al [132] categorize them into five categories that include classical methods like PCA [53], semi-classical methods like multidimensional scaling (MDS) [52], manifold learning methods like ISOMAP [57], Methods rooted in continuum spectral theory, including the Laplacian eigenmaps [84] and advanced manifold methods that include charting [83]. However, broadly we can catogorize these methods into linear and non-linear dimensionality reduction techniques.

9.1.1 Linear Dimensionality Reduction Techniques

These methods find lower dimensional linear representation of the data with their main disadvantage is there limited effectiveness by their global linearity.

1. Principal component analysis (PCA) [53] provides a sequence of best linear approximations to a given high-dimensional observation. The key idea of PCA is to find the low-dimensional linear subspace which captures the maximum proportion of the variation within the data.

2. Multidimensional scaling (MDS) [52] is closely related to PCA where the key idea is to find a mapping from a high-dimensional space to a low-dimensional space, choosing the best pairwise distances between the observed points. An intuitive example is to

recover the relative positions of cities from the inter-city distances [132]. Imagine that the exact locations (coordinates) of N cities are lost. However, we have the driving distances between pairs of them. These distances form an N $\times N$ matrix. Based on this matrix, MDS can recover a 2-D coordinate system that includes the locations of theses cities, subject to a rigid motion (a combination of rotation, shifting, and reflection), such that the distances among the points on this 2-D plane are close to the driving distances among those cities.

9.1.2 Non-Linear Dimensionality Reduction Techniques

These techniques resolve the problem of dimensionality reduction in nonlinear cases

1. Manifold learning techniques such as local linear embedding (LLE) [109], generative topographic mapping (GTM) [13] and ISOMAP [57]. Generative topological mapping (GTM) does not contain the same sophisticated numerical approaches. But its formulation highlights some key components in modern dimension reduction. LLE finds the optimal local convex combinations of the k-nearest neighbors to represent each original vector. ISOMAP is another nonlinear dimension reduction method. It can be viewed as an extension of metric MDS, by replacing the Euclidean distance with another type of distance. This technique is discussed in a bit more detail later.

2. Methods like Laplacian eigenmaps [84] which are based on elegant theory in spectral analysis, and then discretize the results in the continuum to generate numerical approaches.

3. Methods like [83] are based on the key insight in these methods is the realization that the global alignment can be achieved via an eigenvalue computation.

Other methods include Hessian LLE [35], Local Tangent Space Alignment (LTSA) [144], Diffusion Maps, Stochastic Neighbor Embedding [42], Manifold Charting etc.

9.1.3 Learning through Dimensionality Reduction

In this chapter, we also show that reduction of dimensionality can lead new possibilities in designing efficient and possibly more effective learning schemes. Among these learning

techniques, classification is a key step for many tasks in data mining, whose aim is to discover unknown relationships and/or patterns from large set of data. A variety of methods have been proposed to address the problem that may be supervised like decision trees [75] and unsupervised such as the K-nearest neighbor method (KNN) [43]. The KNN finds the K-nearest neighbors of the query point x_0 in the dataset, and then predicts the class label of x_0 as the most frequent one occurring in the K neighbors. However, when applied on large datasets in high dimensions, the time required to compute the neighborhoods (i.e., the distances of the query from the points in the dataset) becomes prohibitive, making answers intractable and in the case of decision trees produces a very large and complex tree structure making comprehensibility very difficult. Moreover, the curse-of-dimensionality, that affects any problem in high dimensions, causes highly biased estimates, thereby reducing the accuracy of predictions.

Thus, here we try to use manifold learning learning technique for classification purpose and see whether it improves the classifier performance or not. Our technique uses semi-supervised manifold learning which differs from the above mentioned methodologies as it makes use of both labeled and unlabeled data for training - typically a small amount of labeled data with a large amount of unlabeled data. The concept of semi-supervised learning is explained below.

9.1.4 Semi-Supervised Learning

Semi-supervised learning, as the name suggests, is halfway between supervised and unsupervised learning. In addition to unlabeled data, the algorithm is provided with some labeled data, but not necessarily for all examples. Often, this information will be the targets associated with some of the examples. In this case, the data set $X = (x_i, ..., x_n)$ can be divided into two parts: the points $X_m = (x_1, ..., x_m)$, for which labels $Y_m = (y_1, ..., y_m)$ are provided, and the points $X_l = (x_{l+1}, ..., x_{l+l})$, the labels of which are not known.

Other forms of partial supervision are possible [139]. The different settings of this supervision corresponds to a different views of semi-supervised learning. In Some approaches is is seen as unsupervised learning guided by constraints While in contrast, most other approaches see semi-supervised learning as supervised learning with additional information on

the distribution of the examples, which seem to be more popular and in line with applications with the goal is the same as supervised learning. But the problem arises if the classes are not known in advance but have to be inferred from the data. In this case, the first approach seems much feasible.

A problem related to SSL was introduced by Vapnik which is the so-called transductive learning. In this type of learning, we have a labeled training set and an unlabeled test set. The idea of transduction is to perform predictions only for the test points. This is in contrast to inductive learning, where the goal is to output a prediction function which is defined on the entire space X.

Semi-supervised learning depends heavily on the distribution of examples, which the unlabeled data will help to be relevant for the classification problem. One could say that the knowledge on $p(x)$ that one gains through the unlabeled data has to carry information that is useful in the inference of $p(y|x)$ [109]. If this is not the case, semi-supervised learning will not yield an improvement over supervised learning. It might even happen that using the unlabeled data degrades the prediction accuracy by misguiding the inference.

However, for semi-supervised technique to be effective certain assumptions have to be made e.g [139].

- The smoothness assumption of supervised learning where if two points x1, x2 are close, then so should be the corresponding outputs y1, y2.
- The Cluster Assumption where if points are in the same cluster, they are likely to be of the same class.
- The decision boundary should lie in a low-density region.
- Manifold assumption where the (high-dimensional) data lie (roughly) on a low-dimensional manifold.

9.2 Our Approach

9.2.1 Framework

In mathematical terms, the problem we investigate can be stated as: given the p-dimensional random variable $x = (x_1, ..., x_p)^T$ find, a lower dimensional representation of it, $x = (s_1, ..., s_k)^T$ with $k \leq p$, that captures the content in the original data, according to some criterion. Whereas, in a classification problem, we are given J classes and N training observations. The training observations consist of p feature measurements $x = (x_1, ..., x_p) \in R^p$ and the known class labels, y, $y = 1, ..., J$. The goal is to predict the class label of a given query.

9.2.2 Manifold

Non linear dimensionality reduction is generally achieved via manifold learning techniques which aim at recovering the real dimensionality of a dataset. Roughly speaking, a manifold denotes that it locally lies in one of the metric space R^d. The manifold is a metric space M with the following property: $if x \in M$ then there is some neighborhood $Uofx$ and some integer $d \geq 0$ such that U is homeomorphic to R^d. Figure 1 (a) shows an example of a one dimensional manifold embedded in a 2 dimensional space.

9.2.2.1 Manifold Learning using ISOMAP

We explain ISOMAP technique as it is use in our method and works as follows. Consider N points, $X_i, i = 1, 2, .., N$, in the data space. First of all, for each data point X_i, consider its neighbors. There are two possibilities:

- k-nearest neighbors of each point X_i.
- An ϵ-neighborhood, which includes all the points that are no more than ϵ-distance away from X_i.

Let N_i denote the index set of the points that are the neighbors of X_i. We construct a graph, in which each i is a vertex, and two vertices are connected if and only if $i \in N_j$ or $j \in N_i$. Define the distance between two points, X_i and X_j, to be the sum of the arc

lengths of the shortest chain connecting X_i and X_j. The shortest chain can be computed via dynamic programming (e.g., Dijkstra, 1959). The above is called a graphical distance. The **geodesic distance** between two points on a manifold is the length of the shortest curve that is on the manifold and connects the two points. The authors [57] show that the graphical distance is in some sense a good substitute for the geodesic distance. Note that a graphical distance is computable from data, while the geodesic distance is not computable. A low dimensional projection is then generated by calling a metric MDS.

9.2.3 Our Semi-Supervised Manifold Learning Technique: (SSM-Learn)

Whatever the manifold learning technique used, it starts by searching for each instance x_i its K nearest neighbors before applying the more sophisticated tasks in order to recover the real dimensionality (structure) of data. The choice of K is of primary importance and can lead to completely different results as illustrated in figure 9.2, 9.3 and 9.4.

Though some theoretical results have recently emerged on a possible strategy to choose an optimal value for this parameter [80], to the best of our knowledge, there is no clear methodology for choosing K in an optimal way. Thus, we adopt a novel approach in order to choose a priori the value of K which is explained in the following sections. Once the value of K have been chosen, we apply a manifold learning technique such as ISOMAP [57] on the whole dataset (labeled and unlabeled examples) in order to derive new features which will represent the new representation space of our semi-supervised learning problem. For completely unseen examples we use neural network based extreme learning machines [41] in order to learn the mapping from X to $\acute{X}$ where X is the input space and $\acute{X}$ is the space composed of the new features generated by MDS on geodesic distance matrix (explained later). This altogether avoids the need to consider these examples as new unseen instances and computing all the manifold process once again. At the end, we apply classification methods such as decision trees on the reduced dimensions for predicting the class.

We start by searching for each instance x_i its K nearest neighbors before applying the more sophisticated tasks in order to recover the real dimensionality (structure) of data. The choice of K is of primary significance and we use a technique which is explained later to approximate the best K. Thus, for the choice of this parameter two configurations are

possible namely supervised learning scheme and semi supervised scheme.

9.2.3.1 Supervised Learning Scheme

In a supervised learning scheme, two main strategies are employed:

1. Distances between points are modified before the learning phase. Typically, a transformation is applied on distances between each pair of points such that the farthest points having the same label are nearer than the nearest points having distinct labels. For example, in [130], the authors introduce the following distance measure between points:

   $D(x_i, x_j) = \sqrt{1 - e^{\frac{-d^2(x_i,x_j)}{\beta}}}$, if $y_i = y_j$ or $D(x_i, x_j) = \sqrt{e^{\frac{d^2(x_i,x_j)}{\beta}}} - \alpha$, if $y_i \neq y_j$

   Another simpler method is an a posteriori modification of the distances which works as follows :

   $\Delta^1 = \Delta + \alpha max(\Delta)\Lambda$, where $\alpha \in [0, 1]$, $max(\Delta)$ is the maximum distance of Δ and $\Lambda_{ij} = 1$ if $y_i = y_j$, otherwise 0.

2. The other strategy consists of using an overall evaluation process (e.g cross-validation) and to keep the value of K which gives the best classification accuracy. This approach can give good results, however, manifold learning algorithms are really time consuming, so this approach is clearly not reliable when we are faced with a large amount of data.

9.2.4 Semi-Supervised Learning Scheme

In a semi-supervised learning context, we have to deal with both supervised learning and unsupervised techniques. Since we are in possession of unlabeled data, we cannot use the modification techniques of distances introduced above. However, we have partial information concerning the label of some points, and this information can be used a priori in order to search for a reasonable value of K. As mentioned earlier that the strategy of trying for several values of K and keeping the one which gives the best classification accuracy seems

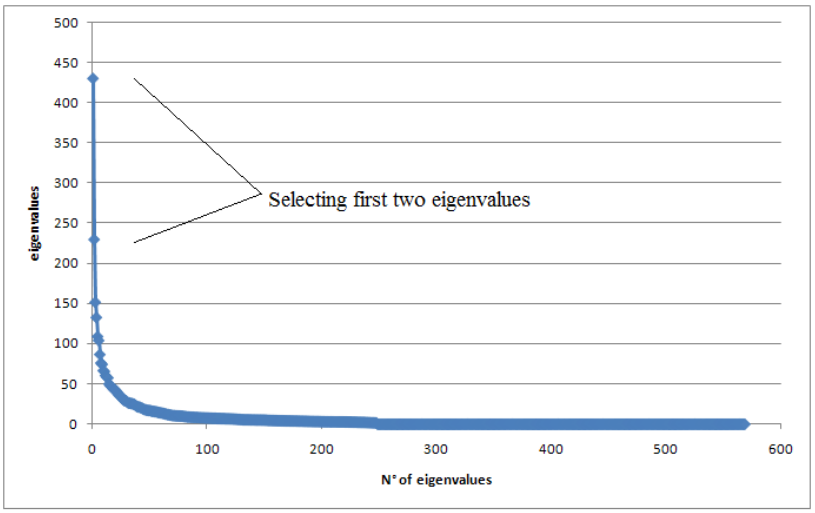

Figure 9.1: Distribution of eigenvalues and selecting the resulting number of dimensions.

to be intractable in this case. Thus, we use an approach using a similarity measure η in order to choose a priori the value of K. The principle of this approach is the following:

1. We construct the KNN graph with default value of k, where we consider the geodesic distance measure between each instances (points). This distance matrix is calculated by taking into account the shortest distance between two points in the graph. This step is easily calculated using Dijkstra algorithm.

2. We then transform these distances into similarity η and convert to a similarity matrix by the following formula:

$$\eta(x_i, x_j) = \frac{1}{1+D(x_i,x_j)}$$

3. Now we calculate the following statistic which has to be maximized in order to achieve the best value for K.

$$\frac{(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\eta(x_i,x_j):y_i=y_j)-(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\eta(x_i,x_j):y_i\neq y_j)}{(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}\eta(x_i,x_j))}$$

At the end of this process, we keep the value of K which maximizes the value of the above statistic.

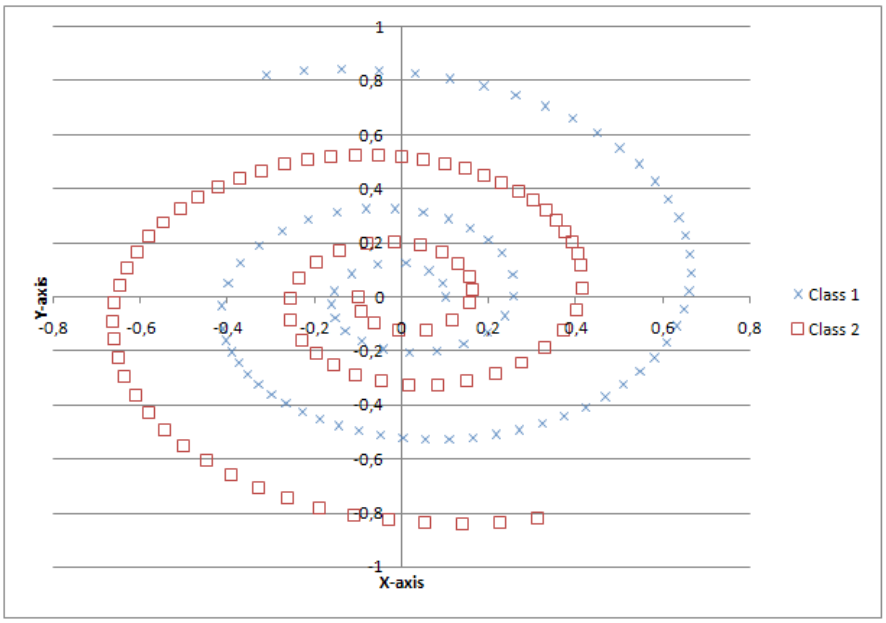

Figure 9.2: Example spiral dataset with two classes.

Once the value of K have been chosen, we apply ISOMAP [57] on the whole dataset (labeled and unlabeled examples) in order to derive new features which will represent the new representation space of our semi-supervised learning problem. The goal of ISOMAP is to replace original distances (euclidean distances) between observations by the geodesic distances along the graph, i.e, distance between x_i and x_j becomes the length of the shortest path between x_i and x_j along the graph. Once the new distance matrix is built, we apply a classical MDS [52] which will derive the new features of the semi supervised learning problem. We can note that other techniques of manifold learning such as Laplacian Eigenmaps can be used in order to build new features. However, since there are no major differences in the results given by those methods, we have decided to apply the most intuitive of them. The choice of the number of new features that have to be taken into consideration is a hard problem in the semi-supervised learning framework. Indeed, the most informative features, i.e, features that have the highest eigenvalues are not necessarily the most discriminant. However, it is our experience that keeping the most informative set of features (i.e, the features with the highest eigenvalues) leads to consistent results. We apply an identical

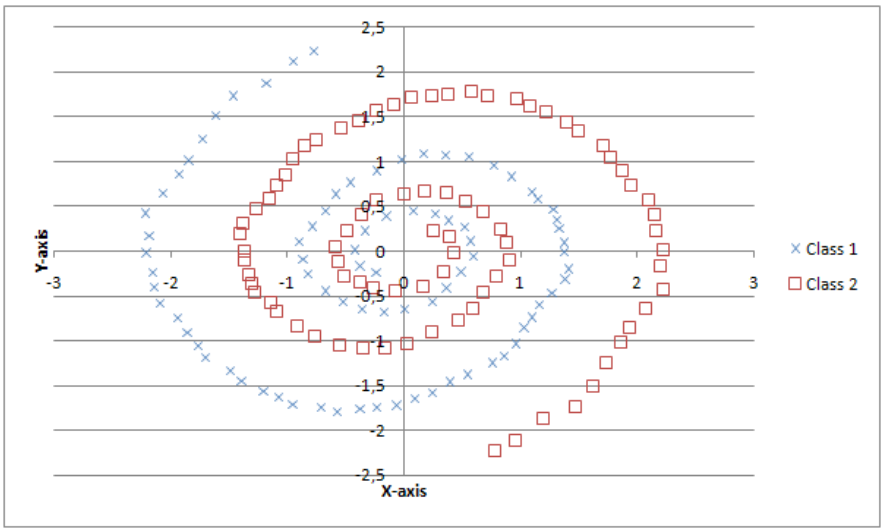

Figure 9.3: Resulting manifold after selecting a bad K value.

procedure that is used in the selection of principal components in PCA [53]. Consequently, we plot on x-axis the number of eigenvalues and on y-axis their values. We keep all eigenvectors (new features) corresponding to eigenvalues, until we observe a sort of stabilization in the decreasing of eigenvalues as shown in fig 8.1. Once we select the resulting features, we classify them using a classification method such as decision trees.

Our algorithm can be used in several ways for predicting labels for unseen instances. If we consider unseen instances as a part of the semi-supervised learning set, we are in the transductive framework and our approach directly calculates predicted labels. However, it is useful to classify completely unseen data (i.e, data that does not belong to the semi-supervised learning set). In this case, the easiest way of computing predictions is to consider these examples as new unseen instances and computing all the process once again. This methodology would be optimal in terms of prediction power (good accuracy), however, it is completely untractable on real datasets containing large amount of data. Thus, in order to deal with this phenomenon, we propose to use a neural network in order to learn the mapping from X to $\acute{X}$ as explained earlier. The choice of neural network is based on the fact that their great flexibility allow to learn any kind of function. Recently, a new class of robust and fast single layer neural networks have been introduced known as Extreme

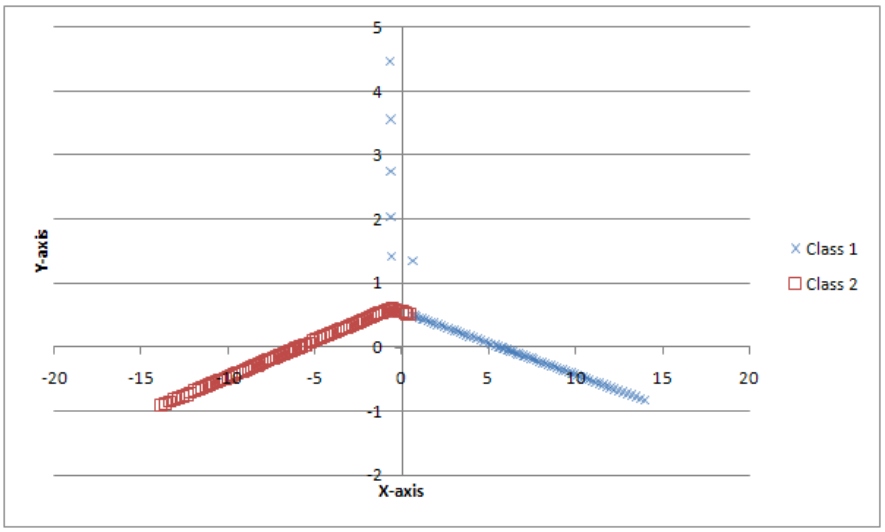

Figure 9.4: Resulting manifold showing distribution of classes.

Dataset	Attributes	Size	Class
Breast	10	699	2
Ionosphere	35	351	2
Iris	4	150	3
Pima	8	768	2
Waveform	21	5000	3

Figure 9.5: Distribution of eigenvalues and selecting the resulting number of dimensions.

Learning Machine (ELM) [41]. Several works have proved their robustness and high-level performances on real and synthetic datasets [41]. We have implemented in our approach these neural networks. As input, we give them the set of initial features and we consider the ISOMAP set of features as the endogenous features. We test several configuration of ELM (several numbers of neurons) and we keep the one which gives the best R^2 (percentage of explained variance) as our space learner.

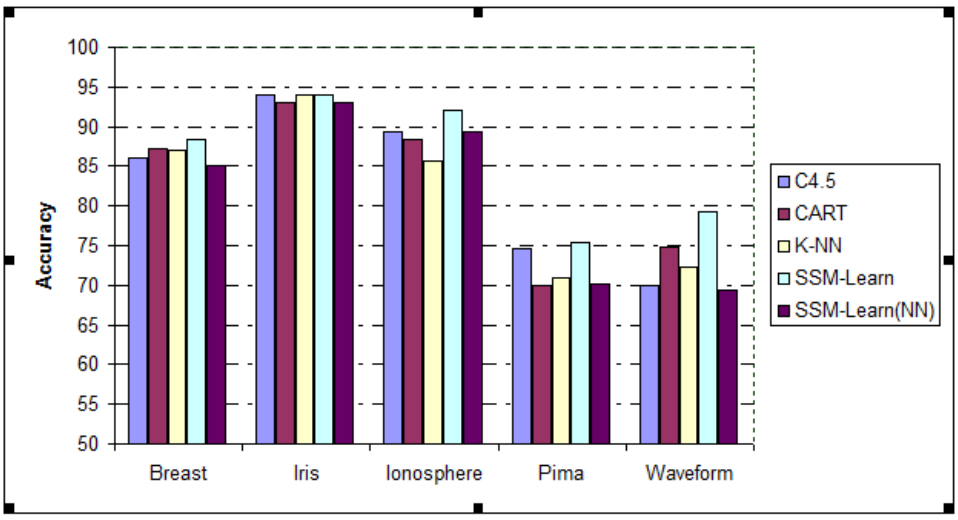

Figure 9.6: Comparison of classification rates of all the methods.

9.3 Results and Analysis

9.3.1 Illustration through an Example Dataset

Here, we analyze the results by experimenting with an example data which comprises of a 2D spiral having two classes as shown in figure 8.2. Selecting a too high value for K leads to the creation of a "'short circuit"' i.e. we connect points that are far apart on the manifold but close in the original euclidean space as shown in figure 8.3. This makes it very difficult to recover the manifold structure. After choosing a suitable K as explained in the previous section, we apply ISOMAP and the resulting class distribution in k dimensions as shown in figure 8.4 which is a well separated manifold structure in terms of class values, making is easy for a classification algorithm such as decision trees to classify. In contrast, applying decision trees directly to this spiral structure will result in a complex and intractable tree with many nodes required to separate the classes.

9.3.2 Analysis by Experimentation with Classical Datasets

In this section we perform experiments using five classical datasets from the U.C.Irvine data repository [15] shown in figure 8.5. We implement our technique SSM-Learn by using C4.5

and then by OC1 [106] decision tree for classification of the dimensionally reduced data. As for SSM-Learn(NN) is concerned, we only use C4.5 with it. We perform a 10 fold cross validation on for each data set and the result shown are the product of 10 trials. In the case of SSM-Learn(NN) we use the 9 folds of data in the following manner. Out of these 9 folds for training we only use two third of this data to perform SSM-Learn and the remaining one third of the data is learned through extreme machine learning as explained previously. Then it is tested on the 10th fold and so on. Thus, we perform the manifold learning algorithm only on two thirds of the training data instead of all the training data, thus consuming lesser time. The other three classification methods used in order to compare are CART, C4.5 and K-NN [43].

Although all kinds of supervised techniques can be used for comparison. But we choose decision tree techniques because of two main reasons:

- The decision tree classifiers are easy to implement and easily interpretable.
- We believe that transversal and eventually oblique cuts are well suited for discovering boundaries between classes once the manifold has been unfolded.

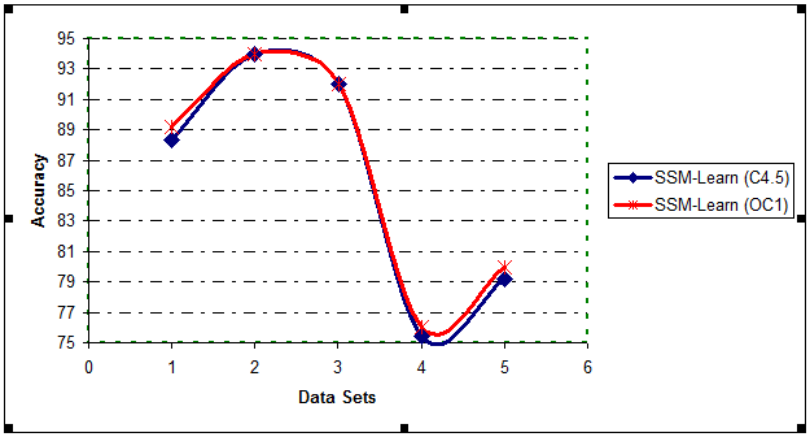

Figure 9.7: Bi-criteria analysis of classification rate and number of dimensions.

9.3.2.1 Accuracy Comparison

Figure 8.6 shows the classification rates of all the methods illustrating the dominance of SSM-Learn approach over the other methods in terms of classification rates. As far as SSM-Learn(NN) is concerned it is inferior to SSM-Learn, although its performance is comparable to the other classification algorithms for Iris, Breast and Ionosphere databases, it suffers in Waveform and Pima databases.

Next, we compare the usage of crisp and oblique cuts in decision trees for classification using the SSM-Learn algorithm. Here, we argue that oblique cuts are better suited for discovery boundaries in the manifolds. Figure 8.7 shows a comparison between the two approaches and shows a slightly better accuracy of the oblique method (OC1) [106] in comparison to the crisp C4.5.

9.3.2.2 Bi-Criteria Analysis

In figure 8.8, we perform a bi-criteria analysis between the classification rate and number of resulting dimensions. Here, we group accuracy with the average number of dimensions which we consider as the two most important criteria and thus, we demonstrate the dominance of SSM-Learn over the other techniques in terms of both these criteria. When C4.5 was used with SSM-Learn, we noted a 30-40 percent decrease in the size of the tree as compared to applying C4.5 directly to the data. This illustrates lesser complexity in terms of readability (an important feature of decision trees) and time complexity.

9.3.3 Model Complexity

The reduction of dimensionality and thus, complex topologies certainly, also reduce the complexity of the decision tree i.e. the number of test nodes. We observed an almost 40 percent decrease in the number of test nodes over all data sets and thus, a significant decrease in the complexity of the decision tree model.

9.3.4 Time Complexity

Although, the overall time complexity of SSM-Learn is a lot higher than that of the above mentioned classification methods, but improvements can be made by using L-ISOMAP as the

manifold learning method which brings the complexity from $O(N^3)$ down to $O(dlnlog(n) + l^3)$. SSM-Learn(NN) used with extreme learning machine to predict unseen data improves the time complexity to an extent but at the same time suffers in its classification accuracy. We argue that there is a trade-off between three important criteria i.e. classification, complexity and dimensionality reduction.

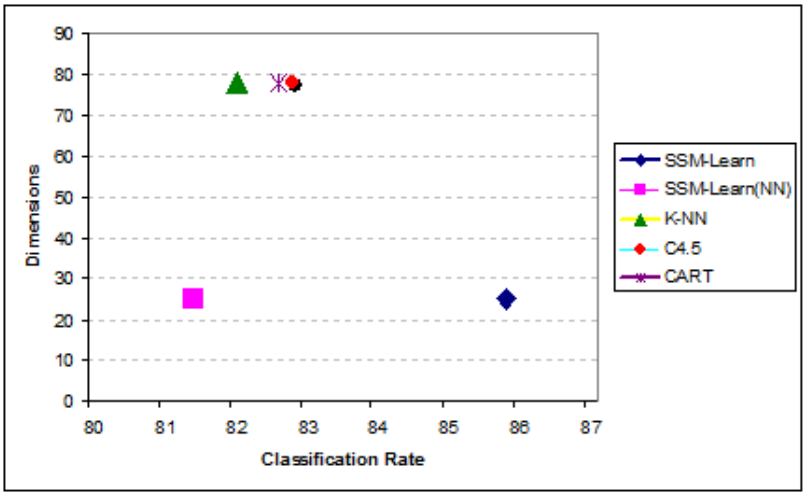

Figure 9.8: Bi-criteria analysis of classification rate and number of dimensions.

9.4 Conclusion for this Chapter

We derive a non-linear dimensionality reduction technique using manifold learning for the purpose of classification. We observe whether using dimensionally reduced manifolds can have a positive effect on the classification performance. During the manifold process we use a technique based on spatial autocorrelation statistics to estimate the value of K and stress its importance in building the manifold structure. Then we apply ISOMAP and convert the original distances to geodesic distances on the graph. Next, we choose the resulting number of features and apply a decision tree based classification method for learning. For unseen examples either we do the entire process again or use the help of a neural network based extreme learning machine to convert the unseen data into labeled form, which improves in time

complexity but suffers in classification accuracy. On comparison with classical classification methods our technique performed better on classification accuracy and comprehensibility due to dimension reduction but suffers from the complexity criteria due to additional computations. As future work, we shall try to improve on issues related to time complexity and benchmarking with more data sets.