Machine Learning, 48, 321–351, 2002

Decision Region Connectivity Analysis: A Method for Analyzing High-Dimensional Classifiers

OFER MELNIK melnik@cs.brandeis.edu
DIMACS Center, Rutgers University, Piscataway, NJ 08854

Editors: Yoshua Bengio and Dale Schuurmans

Abstract. In this paper we present a method to extract qualitative information from any classification model that uses decision regions to generalize (e.g., feed-forward neural nets, SVMs, etc). The method's complexity is independent of the dimensionality of the input data or model, making it computationally feasible for the analysis of even very high-dimensional models. The qualitative information extracted by the method can be directly used to analyze the classification strategies employed by a model, and also to compare strategies across different model types.

Keywords: classification, decision region, decision boundary, convex, concave, DRCA, PCA, rule extraction, high-dimensional visualization, minimum volume ellipsoid

1. Introduction

It is typically difficult to understand what a high-dimensional classifier is doing. The most common form of analysis usually consists of examining raw performance scores. However, as simple one-dimensional measures, they do not lend much insight as to what a model's advantages and shortcomings may be. This problem is exacerbated when we want to compare across different methods that solve the same problem, for instance, across a bank of different neural networks, different graphical model's, or different SVMs, etc.

A model is usually trained or constructed by being given sample input/output pairings that demonstrate its desired function, from which the model is expected to generalize to the rest of the input space. Thus, the way that a model can form sets in the input space (with an infinite number of points) from a finite training sample is intrinsically tied in to how it can generalize. Many of the models used today for classification such as Feed-Forward Neural Networks, Support Vector Machines, Nearest Neighbor classifiers, Decision Trees and many Bayesian Networks generate classification sets that are mostly manifolds or manifolds with boundaries. Sets of this sort exhibit strong locality properties. That is, most of the points in the set have a neighborhood surrounding them such that all points in the neighborhood are also part of the set. Thornton (1997) demonstrated that many of the datasets in the UCI machine learning repository (Blake & Merz, 1998) contain data points that exhibit neighborhood properties, and as such are amenable to generalization by manifold type classifiers.

Given this common generalization method of classifiers, what differentiates between different classifiers is how they individually partition the training points into decision regions. A classifier might only use separate convex decision regions to classify (e.g. a linear

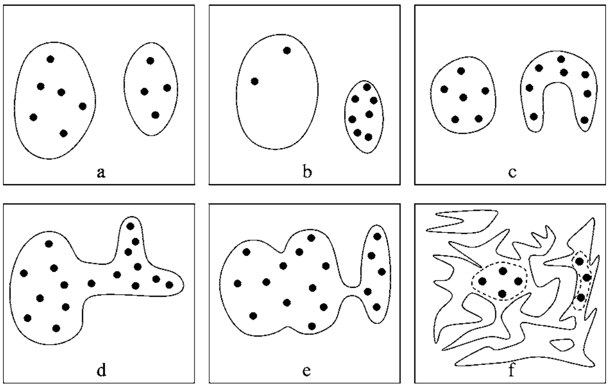


Figure 1. Examples of some of the variations possible in decision region structure.

discriminant classifier). Thus, sample points are separated explicitly by completely segregating them from each other in separate decision regions. However, most interesting classifiers use more complex decision regions to organize the sample points. The points are organized into decision regions with concavity, thus creating a partitioning of the points without explicitly placing them in disconnected decision regions. In a sense this partitioning allows a finer grain of differentiation since points may be closely associated by being in a convex subcomponent of the decision region or be distantly associated through a "network" of other convex subcomponents. Figure 1 illustrates some of the questions that we may want to ask about the decision region structure of a classifier, and some of the interpretations of this information:

1. How many separate decision regions are used (figure 1(a)): On the one hand, too many decision regions might imply that the classifier had a hard time fitting the data to decision regions, and would lead to bad generalization. But some separate decision regions may indicate that the data points themselves come from multiple subclasses.
2. How many and which sample points were used to decide the structure of each decision region (figure 1(b)): Generally, more points throughout a decision region raises our confidence that it actually encapsulates the data. But sometimes having one big decision region may also imply that the data used to construct the classifier is too sparse.
3. The geometry and topology of each decision region (figure 1(c)): A decision region may be convex (the left one), such that if we take any group of points inside the decision region, the area between them is also inside the decision region. Such a decision region describes a kind of uniformity, an extended neighborhood in the input space where all points inside belong to one class. In contrast, a decision region might be concave (the right one), where locations between points from the decision regions may not belong to it, implying some substructure between the components of the decision region.
4. The geometry of convex subcomponents (figure 1(d)): A concave decision region can be decomposed into convex subcomponents, like the one in the figure which can be decomposed into at least three convex subcomponents. The purpose of this is twofold:

First, the decomposition allows us a glimpse into the structure of the decision region, how it separates its different constituent portions and their interrelationships. Second, being convex, the subcomponents can be analyzed using common tools to understand their geometry. Thus, decomposition is an important step in the analysis of the larger concave decision region.

5. Connection strength of convex subcomponents (figure 1(e)): The degree that the convex subcomponents are attached together in a concave decision region is indicative of how separated they are. Concavity might be almost negligible between two strongly attached convex subcomponents (left and middle), or the concavity might act almost like a complete wall between two very weakly connected subcomponents (middle and right).

Our aim is to analyze the decision regions of classifiers. We would like a means to extract a classifier's different decision regions and be able to decompose them individually. To do this in an exact manner would seem to be ideal. Unfortunately, in doing so we run into two related problems, the problem of dimensionality and the problem of complexity. Initially, dimensionality appears to be our bane in the form of visualization difficulties. We can not directly visualize a decision region of more than three dimensions. However, assuming we can overcome that hurdle, dimensionality also influences the complexity of the models. For example, a neural network can have a number of decision regions that is exponential in the input dimension, where the complexity of the individual decision regions is also exponential with respect to the input dimension (Melnik & Pollack, 2002).

In this context, it seems like an almost futile endeavor. However, there is an intrinsic discrepancy between the potential complexity of the model, the complexity of the data and the relevant complexity of a trained model. In figure 1(f) we see an example of this. The model could be representing some highly complex decision regions. However, the actual data points only reside in a simple part of the of decision regions. And with respect to these data points, the model is basically enclosing them in two pseudo-decision regions, one convex and one slightly concave. Not only is the additional complexity of the model an artifact, but if a model is successful at generalizing it must have found some underlying redundancy in the data, therefore in some respect its relevant complexity is even less than that of the data.

Our analysis method tries to extract this relevant complexity, by elucidating the properties of the decision regions in the vicinity of the data points. This is not done by examining the decision regions directly, but rather by examining the effects that the decision regions have on the relationships between the data points, and encapsulating this information in a mathematical graph, a dimensionless form that allows us to make out the properties of the decision regions. This examination of the relationships between the points instead of the general decision regions not only allows us to extract only the relevant complexity, but also makes the analysis method practically independent of the model type and dimensionality of the input space.

The rest of the paper is organized as follows: We first introduce the core analysis method, a method that extracts the structure of decision regions by representing the relationships of the internal points using graphs. Following, we give a relatively simple example of its application to a neural network that classifies points in a three-dimensional space. The fact that it is three-dimensional allows us to visually compare the structure described by the graph with the actual network decision regions. In the section after, we refine the graph analysis method, and explain the method by which we decompose the graph into the subgraphs which correspond to decision region subcomponents. We then continue with an example where we analyze two different types of classifiers, both applied to a high-dimensional letter recognition problem. Using the graph analysis method allows us to clearly show differences in the classification strategies of the two classifiers, and show where one of them will generalize incorrectly. The section after that contains a discussion about the analysis of convex subcomponents, leading to an in depth example of this on an SVM model applied to a dataset from the Statlog project (King, Feng, & Shutherland, 1995). We conclude with a detailed discussion of the method and some of its caveats and then discuss how the method can be extended, suggesting possible avenues of new research.

2. Low level analysis

The fundamental way that manifold type classifiers create decision regions is by enclosing points together in common neighborhoods, which is what our analysis method tries to detect. As input we are given two things, the classifier we wish to analyze and relevant labeled sample points, possibly the training data. It is important that the sample points embody the part of the input space that is of interest, otherwise we would be analyzing the classifier's artifacts and not the relevant regions. From now on, when we refer to decision regions we will mean only the relevant portions of the decision regions (Note that this relevant area can be extended almost arbitrarily if needed).

Figure 2(a) graphically illustrates how the analysis method works. We take all pairs of points with the same classification label (in this case points A–C). Between each pair we

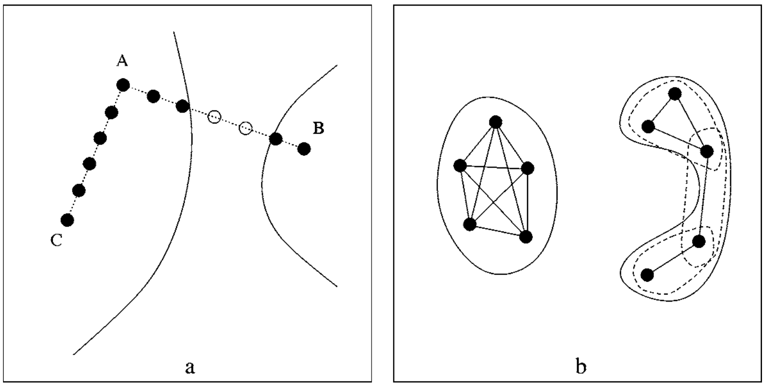


Figure 2. a) The connectivity graph is generated by sampling between the sample points. In this case we see how sampling between points A and B detects a boundary, but points A and C share a neighborhood. b) A connectivity graph for two decision regions, one convex and one concave superimposed over the actual decision regions.

extend a line segment in the input space. We then sample along this line using the classifier. In other words, we find a series of points in the input space along the line and apply the classifier to them. What we look for is a break in the connectivity, a change in the classification label in one or more of the points. Such a change implies that between the two points there is a decision region boundary, and the two points do not share a common neighborhood. Algorithm 1 explicitly describes this operation. Note that if we take a constant number of samples on the line between each pair of points then this algorithm's complexity is $O(n^2)$, where n is the number of points.

With this connectivity information we construct a graph in the mathematical sense. In this graph each sample point is assigned a vertex, and the edges are the actual connectivity information. That is, if two points are connected in the actual input space with respect to the classifier then their vertices are connected in the graph.

This *connectivity graph* can tell us three basic pieces of information: What points reside in separate decision regions, if points are colocated in a convex decision region, or if points reside in a concave decision region. Moreover, in the latter case we can decompose this concave decision region and find what points reside in its different convex subcomponents.

Figure 2(b) illustrates how the graph relates these three pieces of information:

Algorithm 1 The Connectivity Algorithm generates the *connectivity graph* by examining the connectivity between points in the decision regions.

```
$X$ is the set of sample points.
$G(V,E)$ is the undirected connectivity graph, where $|V| = |X|$.
$c(x)$ is the classifier function, returns a 1 if $x$ belongs to the class and other values
otherwise.
$v(x)$ is a function that returns the vertex $v \in V$ corresponding to the sample point $x$.

$Y \leftarrow X$
for all $x \in X$ do
  $Y \leftarrow Y \setminus x$
  for all $y \in Y$ do
    $delta \leftarrow \frac{x-y}{NUMSAMPLES+1}$
    set $(v(x), v(y))$ in $E$
    for $i = 1 \ldots NUMSAMPLES$ do
      if $c(x + i \cdot delta) \neq 1$ then
        clear $(v(x), v(y))$ in $E$
        break
      end if
    end for
  end for
end for
```

1. If decision regions are disconnected then the graphs of the points they enclose are also disconnected. In the figure we see this with respect to two decision regions, whose internal points form two disconnected graphs in the connectivity graph.

2. When points are in a convex decision region then by definition they are fully connected and as such form a clique in the graph. We see this convexity property in the left decision region- it is convex and hence its graph is fully connected.
3. Cliques within the graphs of concave decision regions correspond to its convex subcomponents. The right decision region is not convex and so its graph is not fully connected. However cliques within its graph represent convex subregions of this concave decision region. In this example decision region there are three cliques, representing a decomposition into three convex subcomponents (shown with dashed lines).

Note that in the figure the graph is superimposed over the actual two-dimensional decision regions. Thus the vertices of the graph correspond to the actual points in the input space. This is done for visual convenience, essentially, we could have drawn these vertices anywhere.

3. Analyzing a three-dimensional neural network

A 15 hidden-unit threshold neural network was trained to predict whether a thrown ball will hit a target. As input, it received the throwing angle, initial velocity and the target distance (figure 3), Having only three inputs makes it possible to visualize its decision regions. After iterations of back-propagation and hill-climbing it achieved an 87% success rate on the training data. This system can be easily solved analytically, and the analytic decision region is shown in figure 4 contrasted with the neural network decision region which was extracted using the DIBA algorithm (Melnik & Pollack, 2002).

Using the first 78 of the positive training points, a connectivity graph was generated for the hit class, as seen in figure 5(b). The graph was drawn using a spring-gravity type algorithm (Bruss & Frick, 1995), where the edges are modeled as springs in a physical model. This drawing algorithm has the property of making highly interconnected vertices cluster together, allowing us to recognize cliques.

In the graph we can discern four different clusters that practically form cliques. Assigning a label to the vertices based on which cluster they belong to (if they belong to any cluster), we can plot the position of the actual points in the decision region corresponding to the

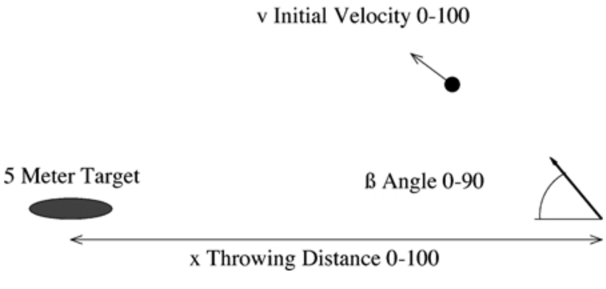


Figure 3. The classification task of the ball throwing network is to predict whether a ball thrown at a certain velocity and angle will hit a target at a given distance.

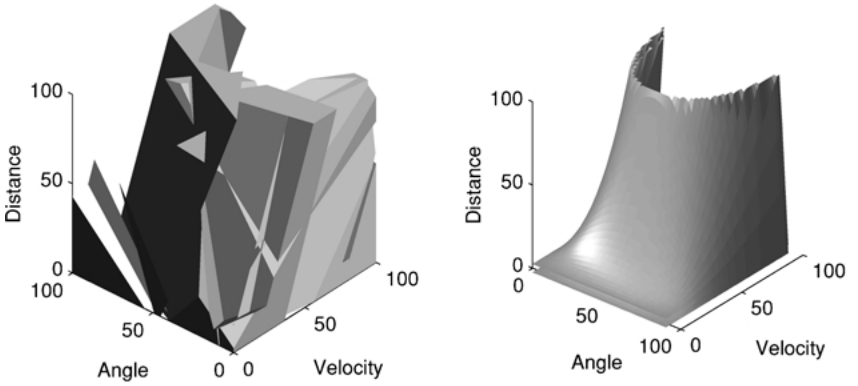


Figure 4. The decision region of the ball throwing network contrasted with the analytic decision region.

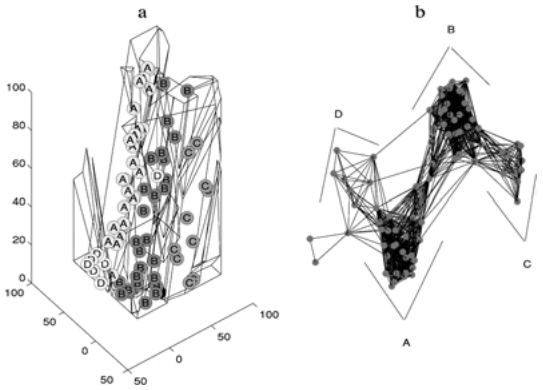


Figure 5. a) The points extracted from the labeling in the connectivity graph superimposed in their correct position within the decision region. b) The connectivity graph of the decision region in figure 4 with respect to 78 internal points. The vertices are labeled by association to four different clique like clusters.

labeled vertices (figure 5(a), compare with figure 4). In this figure we can literally see that the points that make up each of these clusters correspond to four different "slabs" or conspicuous convex subregions that make up the actual neural network concave decision region. In addition, notice how the connections between the clusters in the graph correspond to the relationships between the subregions. That is, how the slab containing the C points touches the slab containing the B points which in turn touches the A slab which touches the D slab, all properties evident in the connectivity graph.

Since we have separated the points into convex subregions we can also analyze their geometric properties. For example, by performing principal component analysis (Bishop, 1995) (PCA) on each of these clusters of points, we can discern their dimensionality and also their orientation. Figure 6 shows the three eigenvalues for each of the clusters as well as for all the points combined. The eigenvalues of the clusters all have a practically negligible

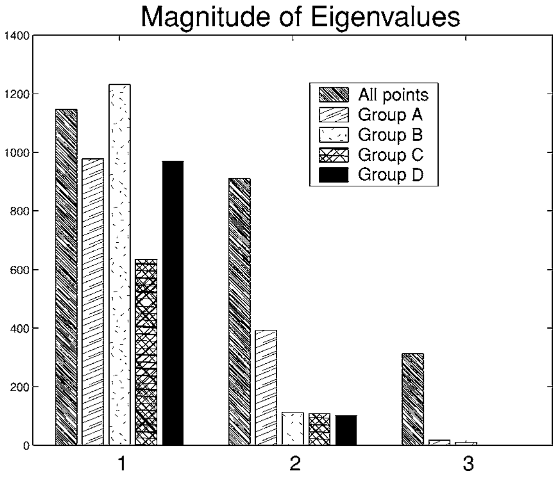


Figure 6. The eigenvalues of the PCA analysis for each of the groups contrasted with the eigenvalues of all the points taken together. The decomposition into groups allows us to realize that the points in the decision region form a two-dimensional embedding in the space, which would not have been discernible by just performing a PCA on all the points together.

third eigenvalue. This indicates that they all form part of a decision region which takes up little volume in the input space, rather it is almost a two-dimensional embedding in a three-dimensional space. In contrast, this is not a property we could have discerned by just performing a PCA of all the points, since the eigenvalues of all the points together show a sizeable magnitude in all three dimensions (as seen in the figure).

4. Higher level graph analysis

The connectivity graph contains the topological information about how the sample points are partitioned into decision regions. With relatively small graphs it may be possible to visually discern the different components of the graph. But with larger or more complex graphs we need a method to decompose the graph into convex subcomponents.

There is no single way to decompose a concave shape into convex subregions. Figure 7 shows some examples of how a concave region can be decomposed. Example *a* shows a decomposition that starts with the largest possible convex subregion, and then incrementally finds the largest subregions in the remaining uncovered areas. From the perspective of the graph (assuming some relationship between number of points and the size of the region) this may be approximated by incrementally locating the largest clique in the uncovered areas of the graph (max clique problem (Bomze et al., 1999)). Example *b* demonstrates a decomposition into the smallest number of convex subregions. In terms of the graph this corresponds to a minimal clique partitioning (Paz & Moran, 1981).

While both of these decomposition approaches satisfy different optimality constraints with respect to the form or quantity of convex subregions that they partition the original

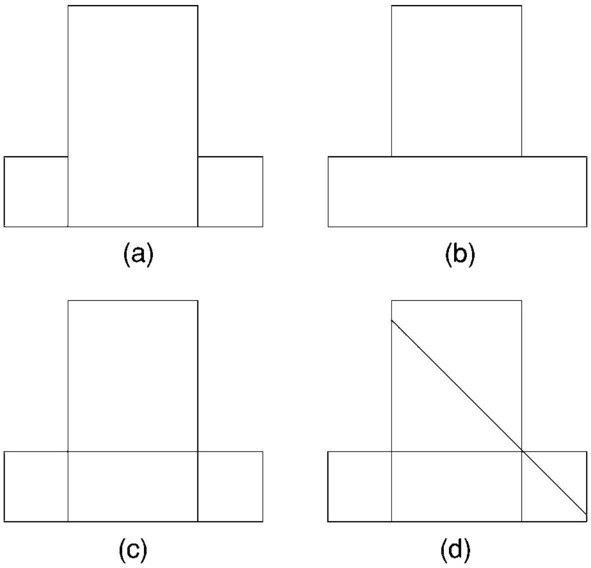


Figure 7. The same concave region can be partitioned into convex subregions in different ways: a) maximum subregion size b) minimum number of subregions. c + d) partitionings that reflect on the original region by allowing its reconstruction by combining subregions.

region into, they do not address our primary goal, understanding the structure of the original region. That is, running these methods may confer the knowledge that the region is decomposable into a particular group of subregions, but these partitionings do not lead to an understanding of how these subregions combine together to form the original region.

Contrast this with the decomposition in examples *c* and *d*. These decompositions have the property that if two subregions are touching then they (and possibly their neighbors) can be combined to form a larger convex subregion. The effect of this property is that it engenders a form of blueprint for the original region from the adjacency relationships of the convex subregions. With the adjacency information we can deduce how the subregions go together to form the original region, illustrating the structure of the concavity and showing how the original region is the product of different overlapping convex subregions.

Figure 8 shows how this kind of decomposition can be generated. We start with one location in the region and find the largest convex subregion that contains that location. We do this repeatedly, at least until the original region is covered by these convex subregions. Then we superimpose the subregions, using their linear borders to further partition the subregions. The previously stated property is maintained by only partitioning already convex subregions, guaranteeing that the new adjacent subregions can be combined to form larger convex regions.

The partitioning in figure 8 was built from the convex regions A–C, forming the smaller convex subregions D through H. One way to think of the partitioning is as a set operation, the intersections and subtractions of the larger convex regions (sets). Thus, we can

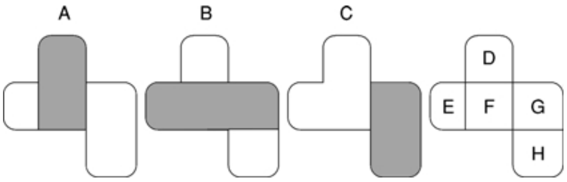


Figure 8. A decomposition that allows reconstruction of the original region can be performed by locating maximal convex subregions from different parts of the region and then superimposing them to form the partition boundaries.

visualize the partitioning as a kind of Venn diagram, showing all the distinctive subsets that can be formed by these set operations. In this particular example we can write $D = A\backslash B$, $E = (B\backslash A)\backslash C$, $F = A \cap B$, $G = B \cap C$, and $H = C\backslash B$. This set theory analogy of the decomposition leads to an interesting observation, each subregion belongs to a unique combination of larger convex regions. Since each subregion is the result of a different application of set operators, no two subregions belong to the same group of larger convex subregions, as evidenced by the following relations.

$$\begin{array}{lll}
D \subset A & D \subset \bar{B} & D \subset \bar{C} \\
E \subset \bar{A} & E \subset B & E \subset \bar{C} \\
F \subset A & F \subset B & F \subset \bar{C} \\
G \subset \bar{A} & G \subset B & G \subset C \\
H \subset \bar{A} & H \subset \bar{B} & H \subset C
\end{array}$$

As just stated, the decomposition is described as a top-down process, start with the complete shape, do an initial partitioning, and gradually refine that partition to a desired level. However, we may also look at the partitioning from a bottom-up perspective. Noting the set properties, it is possible to think of this type of decomposition as a grouping of points from the original region that belong to a unique combination of larger convex regions. In this way, all the points with the same signature combination are grouped together to form one of the subregions. The advantage of this perspective is that it acts to define how we should go about analyzing the connectivity graph, by telling us how to group together vertices that belong to the same subregions.

Vertices belong to the same subregion if they belong to the same signature combination of larger convex regions. In terms of the graph, belonging to a convex region implies that the vertex is connected to all the other vertices in that region. Thus, if two vertices belong to the same convex regions then by definition they should be connected to the same vertices. And, by the same token, if those same two vertices do not belong to the same convex regions then by definition they should be disconnected from the same vertices. As such, the vertices that make up a subregion all have the same (or similar) *connectivity signature* pattern. The higher-level analysis approach uses the connectivity signature of vertices to decompose the connectivity graph in a way that allows us to understand the underlying structure of the concave regions represented by the connectivity graph.

This higher-level analysis method extracts two pieces of information from the connectivity graph: First, which points are colocated in the same convex subregions. Second, using the adjacency information, how do the convex subregions combine to form the original decision region. The basic assumption of the method is that multiple points inhabit the convex subregions of the decision region. This is a fair assumption if we expect the classifier to have generalization properties, since a manifold type classifier can only generalize by recognizing neighborhoods of points to enclose together.

In the first stage of the method we seek to group sample points with similar connectivity signatures, to group vertices that are mostly connected to the same vertices and disconnected from the same vertices. This is done as follows: Consider the connectivity matrix associated with the graph, where each row enumerates the edges of a vertex in the form of a binary vector. The basic grouping operation is simply: If the hamming distance between two such vectors is sufficiently small then we group their respective vertices together. Allowing the distance to fall within a threshold, rather than mandating an exact match, serves to both give a robustness to the algorithm, allowing imperfections in the convex subregions, and acts as a granularity control for subregion generation. These elements of the algorithm are further discussed in Section 7.

To see how this grouping operation works, consider the concave decision region in figure 9(a) and its respective connectivity graph in figure 9(b). Notice how the points in the bottom left part of the decision region are all connected to each other by belonging to the same convex subcomponent, in addition notice that they are also all connected to the points in the top portion of the decision region, but not to any of the points in the bottom right portion of the decision region. Thus, all these points exhibit a distinctive connectivity signature which is distinct from the other convex subcomponents of this decision region. In contrast, for example, the points in the top portion of the decision region are connected to all the other points, whereas the points in the bottom right portion are connected to all but the points in the bottom left. Thus, if we label the vertices based on similar connectivity, then we end up with three groups of points as illustrated. Each group represents a convex subregion, as each group's vertices forms a clique. However, these groups are differentiated

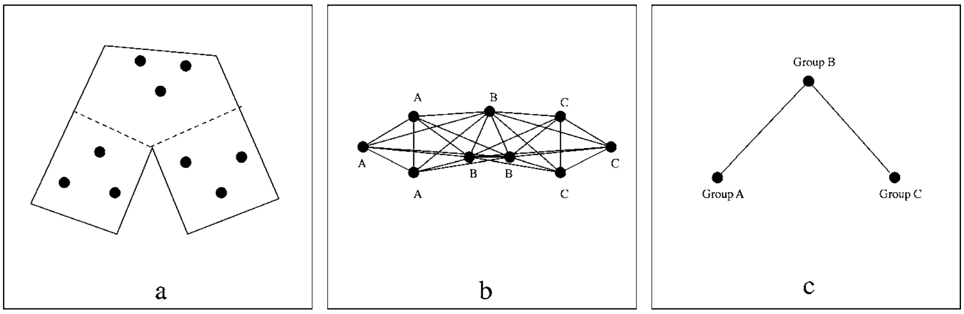


Figure 9. a) A concave decision region housing nine points in different convex subcomponents. b) The connectivity graph for the points in the decision region. c) The group graph associated with the labeling of the connectivity graph.

with respect to their position in the decision region, which is divined by their intergroup connectivity.

Algorithm 2 shows the grouping procedure. If we discount the computational cost of calculating the Hamming distance (which on some platforms is almost an atomic operation) then the complexity of this algorithm ranges from $O(n^2)$ in the worst case to $O(n)$ depending on the density of the groups.

Algorithm 2 The Grouping Algorithm groups together vertices with a similar *connectivity signature*.

```
$G(V,E)$ is the connectivity graph.
$\alpha$ is the percentage Hamming similarity.
$\beta$ is the minimum group size.

$M \leftarrow V$
while $|M| > 0$ do
  arbitrarily pick $i \in M$
  assign new group label to $i$
  $M \leftarrow M \setminus i$
  for all unlabeled $j \in V$ do
    if $\frac{|V|-HammingDistance(i,j)}{|V|} > \alpha$ then
      assign group label of $i$ to $j$
      $M \leftarrow M \setminus j$
    end if
  end for
  if group size $< \beta$ then
    remove label for all group members
    $M \leftarrow M \cup \{group \setminus i\}$
  end if
end while

function HammingDistance (vertex $i$, vertex $j$)
$distance \leftarrow 0$
for all $k \in V$ do
  if $((i,k) \in E \wedge (j,k) \notin E) \vee ((i,k) \notin E \wedge (j,k) \in E)$ then
    $distance \leftarrow distance + 1$
  end if
end for
return $distance$
```

In the second stage of the analysis we wish to simplify the original connectivity graph, in order to represent the adjacency relationships between the convex subregions. This is done as follows: For each group, take all its vertices and merge them, thus transforming the group into one labeled vertex with the same intergroup connectivity as the group originally had. Doing so, we are left with a much smaller and sparser *group graph* that relates the

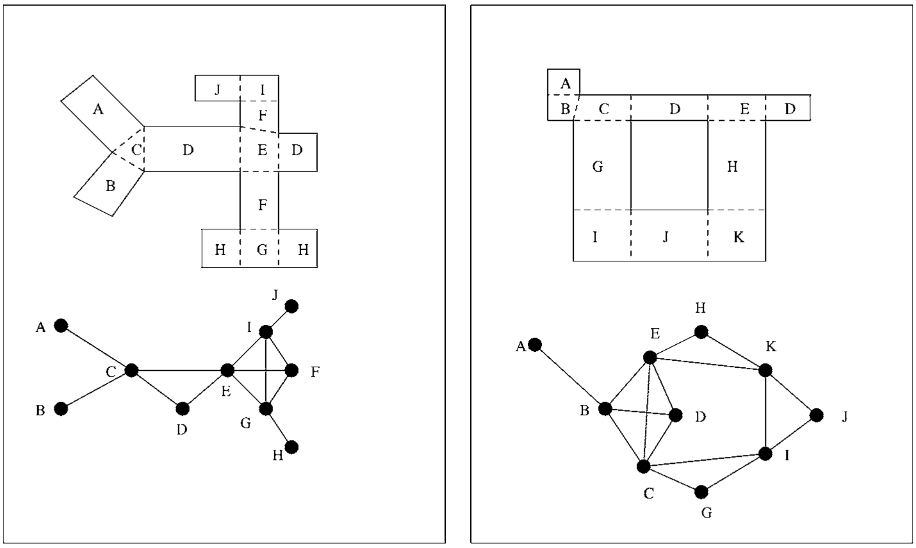


Figure 10. An example of two different concave decision regions and their respective group graph.

relationships between the groups—their intergroup connectivity. In figure 9(c) we see this operation performed on the connectivity graph in figure 9(b). This new graph shows us that with respect to the sample points, the original decision region partitioned the space into three groups such that one of the groups (group B) is connected to both of the other groups, but that the other two groups are not directly connected. Notice how the graph is similar in structure to the actual decision region.

Figure 10 illustrates the relationship between the group graph and two different more complex concave decision regions. Cliques in the group graph represent groups that may be combined to form larger convex subcomponents, possibly for a geometrical or statistical analysis of the points in the subregion. In the trivial case, every edge in the graph is a clique and represents a potential combined convex region. Another group graph characteristic is loops of cliques. Loops of cliques in the group graph represent the existence of holes in the decision region, as the convex subregions that go around them must be connected together. The fundamental characteristic of interest in the group graph is the branching structure. It relates the actual partitioning between convex subregions, which subregions are directly connected, which are distantly connected and their connection paths.

The next example applies the higher-level analysis method to compare between two different types of high dimensional classifiers on the same task. By understanding the different ways that these two classifiers partition the training data, we can learn how they generalize differently.

5. Comparing two classifiers: A high-dimensional example

The UCI repository (Blake & Merz, 1998) contains a dataset contributed by Alpaydin and Kaynak (1998) of handwritten digits. The original dataset contains 32 by 32 pixelated images, normalized for scale and position. There is also a preprocessed version of the dataset, where the 32 by 32 images are shrunk to 8 by 8 by counting the number of pixels in each 4 by 4 region of the original image. This training set contains 3823 samples from 30 people.

Using the preprocessed dataset the following classification task was constructed. The data corresponding to the numerals 3 and 4 were assigned to one class, while the remaining numerals were assigned to a second class. Thus the task consisted of classifying a 64-dimensional input into two classes.

Two classifiers were used, a sigmoidal feed-forward neural network with one hidden layer of 7 units and a K-nearest neighbor classifier with K set to 9 (Bishop, 1995). The network was trained using conjugate gradient (Bishop, 1995) until it reached perfect classification on the test data.

In order to make the connectivity graph more presentable, only the first 63 cases of the 3-4 class were used to draw it. In the additional levels of analysis 300 exemplars were used.

Figure 11 shows the connectivity graph for the neural network. Since the graph is connected it consists of one decision region. However, it is apparent that this graph is illustrating a concave decision region because the graph consists of two highly connected regions with only very sparse connectivity between them. The points were labeled using the labeling method described before at a 90% Hamming similarity, which labeled the points as expected into two classes corresponding to these (practically) clique subregions. In the connectivity graph the clusters are almost completely disconnected, therefore we do not need to draw

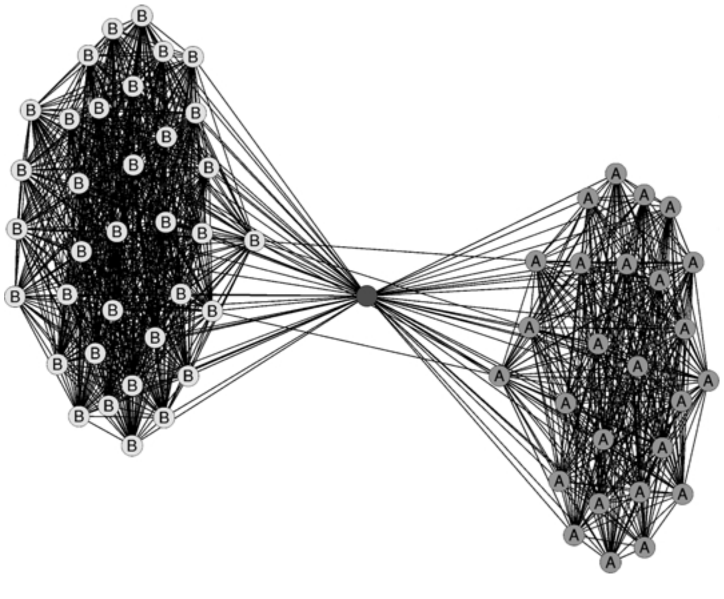


Figure 11. The connectivity graph of a 64-input neural network trained to classify the numerals 3 and 4 as one class and the other numerals as another class. The graph clearly illustrates that for the 3-4 class the network constructed a decision region with two separate subclasses.

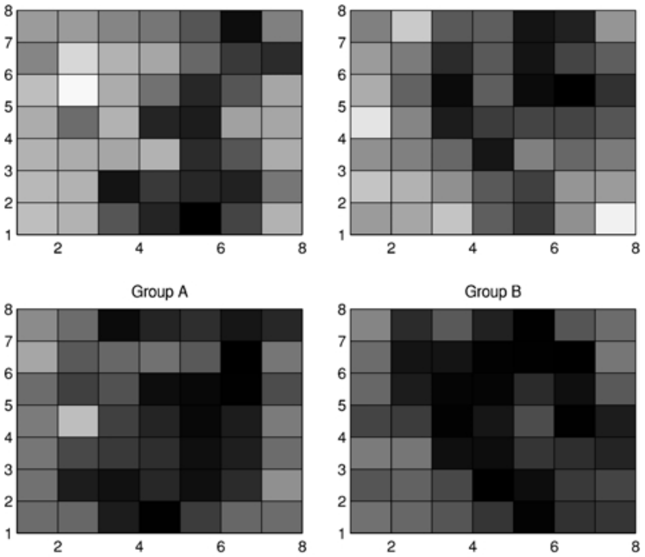


Figure 12. For the two labeled sets in the connectivity graph in figure 11 a PCA analysis was done to estimate the extent of the convex subregions, and two extreme values are shown. As seen, group A contains threes, and group B contains fours.

a group graph, since a group graph is only constructed when the groups make up parts of larger convex subregions.

When we examine the actual numeral associated with the labeled points we realize that the points associated with the first label all correspond to the threes, and all the points with the second label correspond to the fours. What this means is that the network discovered that the 3-4 class really consists of two subclasses and divided its decision region to clearly separate between them. Suppose that we did not know that the class was decomposable and wanted to know the composition of the subregions that the neural network generated. As before, since the subregions are convex we can analyze them using PCA. In figure 12, for each group, we took the mean of the points, in the top half we added the first 20 eigenvectors of the PCA normalized by their standard deviation, and in the bottom half we subtracted the same values. This gives a coarse approximation of the decision region's scope, the part of the input space that in encapsulated by the region. As can be seen, the left images correspond to threes and the right to fours, so we can literally see that the two subregions correspond to two logically separate subclasses, without looking up the original classes of the points.

Figure 13 shows the connectivity graph for the K-Nearest Neighbor classifier. Again, it is a connected graph and hence has one decision region. This graph doesn't lend itself to a simple visual analysis, since it is more dense. However, when we apply the labeling method at 80% Hamming similarity we get three labeled classes as illustrated. The group graph analysis of the three labeled sets shows that the vertices in group C are connected to both group A and B, but that there are hardly any connections between groups A and B directly. Therefore the group graph is of the form we saw in the example in figure 9. In figure 14

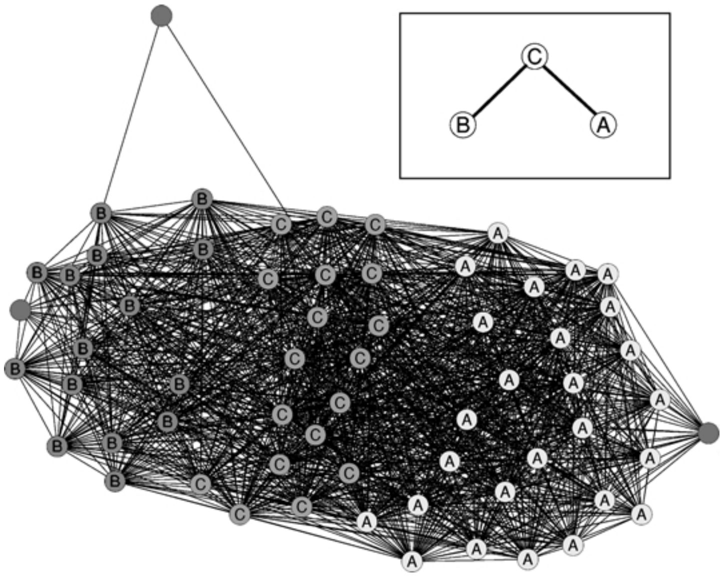


Figure 13. The connectivity graph of a K-nearest neighbor classifier set to classify the numerals 3 and 4 as one class and the other numerals as another class. The labeling of the graph suggests a weaker concavity than the neural network's decision region.

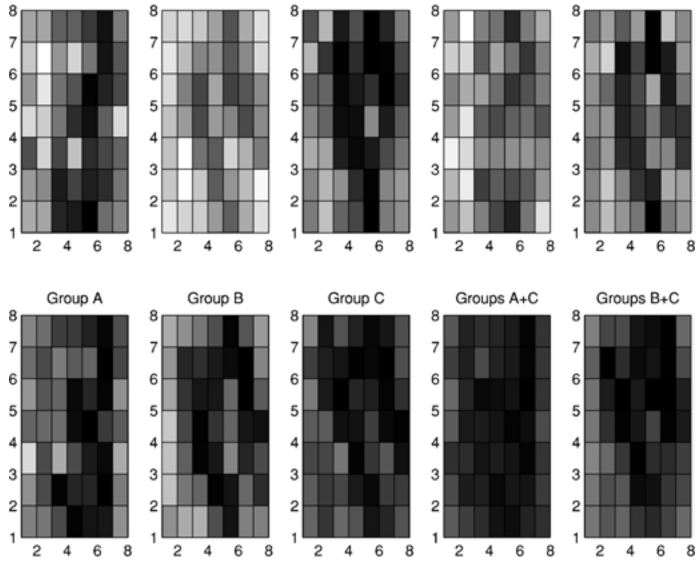


Figure 14. A PCA analysis was done to estimate the extent of the convex subregions of the KNN connectivity graph. Two extreme values are shown for each region analyzed. The subregion containing groups A and C is suspicious.

we present the results of applying PCA analysis (as before) on the three different groups as well as on the two cliques of the group graph. The figure shows that group A corresponds to threes, and groups B and C correspond to fours. Since B + C and A + C form cliques in the group graph, they form larger convex regions. The PCA analysis of the composition of B and C corresponds (as expected) to fours, but the images of the composition of groups A and C are not interpretable. This lack of interpretability goes with what we know about how the data is structured, a convex subregion, such as this one, consisting of both threes and fours would have to contain spurious data, data which is neither a three or four, and thus lead to a malformed classification set. When we examine the actual numerals associated with the labeled points, we see that the A-labeled points do correspond to threes, and the B and C labeled points do correspond to fours.

Both of the classifiers realized that the points making up the 3-4 class are not homogeneous. This is demonstrated by the fact that both classifiers used a concave decision region to house the points of the class. However the discrepancy between them lies in how clearly they realized what the two subclasses are. The neural network made a very clean distinction, clearly dividing the space between the threes and fours. Whereas the K-Nearest neighbor classifier divided some of the threes completely from some of the fours (groups A and B). However, it did not differentiate between the threes in group A and the fours in group C, hence we would expect potential misclassification in that region of the input space.

6. Learning from lower dimensions: A high-dimensional example

6.1. PCA on ellipsoids

Due to specific properties of the previous two examples (low dimensionality or visual interpretation of the input space) we could perform a visual interpretation for the PCA results of the convex subcomponents. For many applications a visual interpretation is not possible. However, PCA results can still be numerically interpreted to gain an understanding of the strategies employed in different parts of the input space by a classifier. The next example is one which requires such a numerical interpretation, as its variables are not readily visually interpreted. Using PCA we will see how the classifier treats the variables differently across the subcomponents of its decision region. But before we proceed we need to reexamine how we perform PCA on the points of the convex subcomponents.

In the previous two examples we performed a PCA on the points that make up cliques in the standard way, using the covariance matrix. PCA is typically performed on the covariance or correlation matrix of the data points making the assumption that the points were sampled from an underlying distribution. The assumption of a distribution attributes a significance to the density or measure of the points. Thus, if the sample contained more points in a particular part of the space the impact on the covariance matrix would be greater than points coming from a sparser part of the space.

In our application there is no real meaning to the density of the points. Rather than representing a sample from a distribution, the points define the shape and extent of a convex part of a decision region. We know that all the points are inside the convex region and that all the space between the points is also inside the convex region. Therefore, looking at the density of the sample points would be deceiving since the complete interior of the convex hull of these points is uniformly inside the decision region. Instead, our interest lies with the geometric properties of the convex hull defined by the points.

It is difficult to study the convex hull directly. As such, we aspire to approximate the hull with a more regular shape, specifically an ellipsoid. Using an ellipsoid to approximate data is a common approach both in statistics (Rousseeuw, 1997) and in other domains (Sabharwal & Potter, 1995). This is in part due to the relationship proved by Lowner and John (Lovasz, 1986) between the minimum volume enclosing ellipsoid of the hull and the maximum volume enclosed ellipsoid. They proved that these two ellipsoids are concentric and the same except for a constant shrink factor. Thus implying that the *minimum volume ellipsoid* (MVE) would make a reasonable approximation to a convex region by somehow capturing and bounding its geometry.

The advantage of using an ellipsoid to approximate the convex region is that it allows us to continue using PCA to understand the shape of the region. Instead of performing the PCA on the covariance matrix we apply it to the scatter matrix of the ellipsoid, using the geometric interpretation of PCA (Jolliffe, 1986). Given an ellipsoid described by the equation:

$$(x - \mu)'\Sigma^{-1}(x - \mu) = c$$

Where μ is the center of the ellipsoid, Σ^{-1} is the scatter matrix and c is a constant equal to the dimension of the space, then the principal components of Σ correspond to the directions of the principal axes of the ellipsoid, and the eigenvalues can be used to calculate the half-lengths or radii of the axes.

In the next example we demonstrate how to use this method of performing PCA on the MVE's scatter matrix. There, we calculate the MVE using the algorithm proposed by Titterington (1978), a relatively fast iterative algorithm. Note that this is an area of active research and there are other algorithms to compute the MVE (Shor & Berezovski, 1992).

In this example we will also show another advantage of using ellipsoids. By enclosing points in ellipsoids in the analysis we indirectly build an alternative classifier with the same group structure as the one we are studying. On the one hand, this ellipsoid classifier allows us to verify our analysis. But, as we will see, it also allows us to transfer decision region structure across different input dimensions.

6.2. *The task and classifier*

The vehicle database (Siebert, 1987) used in the Statlog project (King, Feng, & Shutherland, 1995) describes the silhouettes of four different types of vehicles: an Opel, a Saab, a van and a truck. Each entry in the database contains 18 different geometrical and statistical measures of the respective silhouette. There are 846 entries in the database, which after random shuffling, were divided into a 550 entry training set and 296 entry test set. As stated before, the previous two examples had a visual component to their PCA/convex analysis. In this dataset we do not have that luxury, which is one reason it was chosen, to demonstrate how the analysis can be done on raw numerical data.

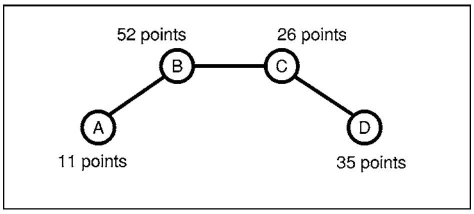


Figure 15. The group graph of the six input SVM model classifying the van class from the vehicle dataset.

Using the SVM-Light package (Joachims, 1999) a degree 3 polynomial kernel SVM (61 support vectors) was trained to recognize the van vehicle using only the first 6 of the 18 measures (normalized to the range $[-1, 1]$). The reason we used a smaller number of inputs was primarily to simplify the analysis, having fewer variables implies less interdependencies between them. Using the first six inputs was an arbitrary choice. Any subset which allowed the classifier to achieve sufficient accuracy on the task could have been used. This particular classifier achieved 98.1% on the training data and 93.9% on the test set.

Using the training data a connectivity analysis was conducted rendering the group graph in figure 15. As can be seen, the classifier consists of one concave decision region with 4 connected convex subregions, A–D. The connectivity between these regions means that A and B form a larger convex subregion, so do B and C, as do C and D. So, for the sake of analysis, we can decompose the decision region into 3 overlapping convex subregions.

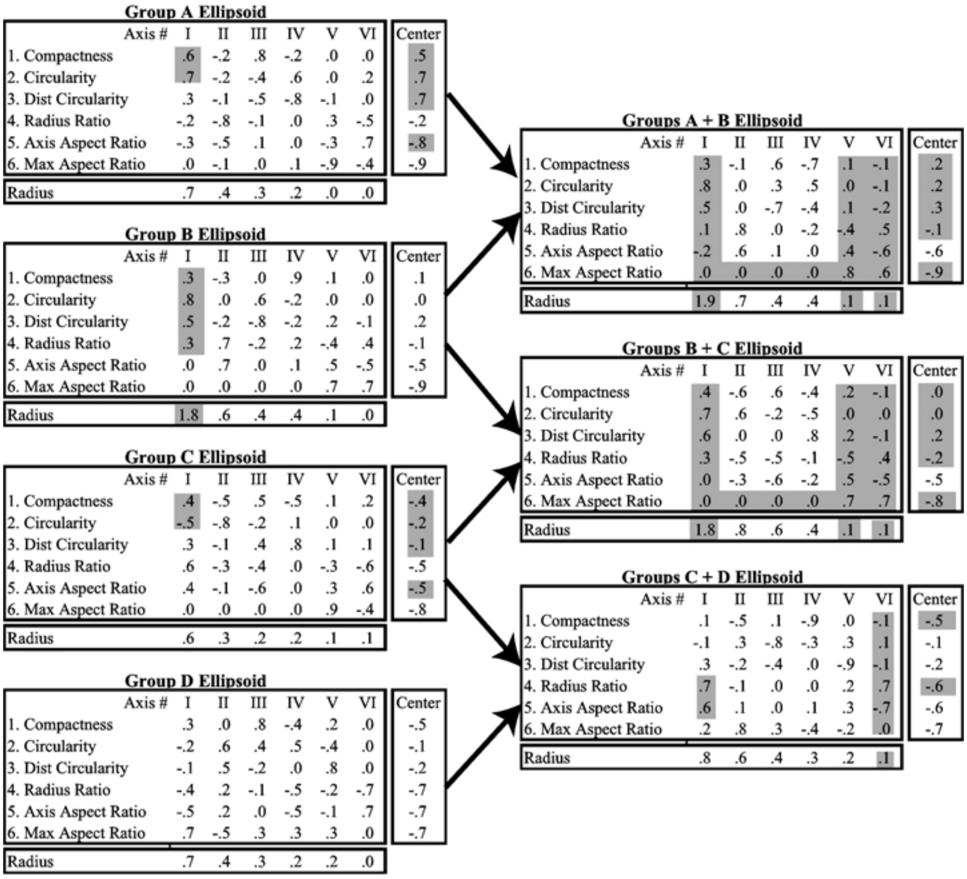
Figure 16 contains the results of the ellipsoid PCA method applied to each of the convex subregions. Note that the values in the tables are rounded. This is common practice as PCA is robust to rounding with respect to interpretability (Jolliffe, 1986). Next, we will interpret these tables in order to understand what geometric properties define membership in the van class, for each of these different convex subregions of the input space.

6.3. PCA interpretation

Interpretation of PCA data is an art unto its own (see Jolliffe, 1986, for example, for a deeper treatment). To generalize for the sake of simplifying the process, the principal components are roughly divided into two sets: those with a relatively high eigenvalue or radius, and those with a small eigenvalue or radius.

Those components with a large radius describe correlations in the data. That is, the variables with a large magnitude in a principal component vector with a large radius, are variables that move together: Either they move in the same direction together (positive correlation) if they have the same sign, or they move in opposite directions together (negative correlation) if they have opposite signs.

On the other hand, components with a small radius act as constraints, describing relationships between the variables in the component vector which must be maintained in order to stay within the decision region. Constraints are probably more important to understanding



Group A Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.6	-.2	.8	-.2	.0	.0	.5
2. Circularity	.7	-.2	-.4	.6	.0	.2	.7
3. Dist Circularity	.3	-.1	-.5	-.8	-.1	.0	.7
4. Radius Ratio	-.2	-.8	-.1	.0	.3	-.5	-.2
5. Axis Aspect Ratio	-.3	-.5	.1	.0	-.3	.7	-.8
6. Max Aspect Ratio	.0	-.1	.0	.1	-.9	-.4	-.9
Radius	.7	.4	.3	.2	.0	.0	

Group B Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.3	-.3	.0	.9	.1	.0	.1
2. Circularity	.8	.0	.6	-.2	.0	.0	.0
3. Dist Circularity	.5	-.2	-.8	-.2	.2	-.1	.2
4. Radius Ratio	.3	.7	-.2	.2	-.4	.4	-.1
5. Axis Aspect Ratio	.0	.7	.0	.1	.5	-.5	-.5
6. Max Aspect Ratio	.0	.0	.0	.0	.7	.7	-.9
Radius	1.8	.6	.4	.4	.1	.0	

Group C Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.4	-.5	.5	-.5	.1	.2	-.4
2. Circularity	-.5	-.8	-.2	.1	.0	.0	-.2
3. Dist Circularity	.3	-.1	.4	.8	.1	.1	-.1
4. Radius Ratio	.6	-.3	-.4	.0	-.3	-.6	-.5
5. Axis Aspect Ratio	.4	-.1	-.6	.0	.3	.6	-.5
6. Max Aspect Ratio	.0	.0	.0	.0	.9	-.4	-.8
Radius	.6	.3	.2	.2	.1	.1	

Group D Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.3	.0	.8	-.4	.2	.0	-.5
2. Circularity	-.2	.6	.4	.5	-.4	.0	-.1
3. Dist Circularity	-.1	.5	-.2	.0	.8	.0	-.2
4. Radius Ratio	-.4	.2	-.1	-.5	-.2	-.7	-.7
5. Axis Aspect Ratio	-.5	.2	.0	-.5	-.1	.7	-.7
6. Max Aspect Ratio	.7	-.5	.3	.3	.3	.0	-.7
Radius	.7	.4	.3	.2	.2	.0	

Groups A + B Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.3	-.1	.6	-.7	.1	-.1	.2
2. Circularity	.8	.0	.3	.5	.0	-.1	.2
3. Dist Circularity	.5	.0	-.7	-.4	.1	-.2	.3
4. Radius Ratio	.1	.8	.0	-.2	-.4	.5	-.1
5. Axis Aspect Ratio	-.2	.6	.1	.0	.4	-.6	-.6
6. Max Aspect Ratio	.0	.0	.0	.0	.8	.6	-.9
Radius	1.9	.7	.4	.4	.1	.1	

Groups B + C Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.4	-.6	.6	-.4	.2	-.1	.0
2. Circularity	.7	.6	-.2	-.5	.0	.0	.0
3. Dist Circularity	.6	.0	.0	.8	.2	-.1	.2
4. Radius Ratio	.3	-.5	-.5	-.1	-.5	.4	-.2
5. Axis Aspect Ratio	.0	-.3	-.6	-.2	.5	-.5	-.5
6. Max Aspect Ratio	.0	.0	.0	.0	.7	.7	-.8
Radius	1.8	.8	.6	.4	.1	.1	

Groups C + D Ellipsoid

Axis #	I	II	III	IV	V	VI	Center
1. Compactness	.1	-.5	.1	-.9	.0	-.1	-.5
2. Circularity	-.1	.3	-.8	-.3	.3	.1	-.1
3. Dist Circularity	.3	-.2	-.4	.0	-.9	-.1	-.2
4. Radius Ratio	.7	-.1	.0	.0	.2	.7	-.6
5. Axis Aspect Ratio	.6	.1	.0	.1	.3	-.7	-.6
6. Max Aspect Ratio	.2	.8	.3	-.4	-.2	.0	-.7
Radius	.8	.6	.4	.3	.2	.1	

Figure 16. This figure shows the PCA analysis of each of the ellipsoids used to approximate the convex regions of the SVM classifier. The hi-lighted values are discussed in the text.

a classifier's decision regions than are correlations, as the constraints describe the sharp boundaries between belonging or not belonging to the class.

How does a small radius principal component express a constraint? Since the component represents a small axis of the underlying ellipsoid then it implies that moving from the center in the direction of the axis vector will quickly take us out of the ellipsoid. Hence the component points in the direction of the decision region border. A complementary algebraic perspective is to consider that any point $\bar{x}$ in the ellipsoid (taken from the ellipsoid's center) can be expressed as $\bar{z}$ using the coordinate system given by the PCA vectors from the center of the ellipsoid. The component z_i of the coordinate corresponding to the i-th principal component $\overline{p_i}$ is calculated by the inner product $\bar{x} \cdot \overline{p_i}$, due to the orthonormality of the principal components. To be inside the ellipsoid z_i has to be smaller than the radius of $\overline{p_i}$. But since the radius of $\overline{p_i}$ is very small or almost negligible we get the approximate equation $\bar{x} \cdot \overline{p_i} = 0$, a direct linear constraint on the values of $\bar{x}$.

6.4. The analysis

We would like to analyze the larger convex subregions, AB, BC and CD. In our analysis we are interested in answering two questions: 1) What part of the input space does the ellipsoid address? 2) What unique identifier of belonging to the class, or constraints on the data are imposed by the ellipsoid?

First consider the group consisting of C and D (figure 16). Starting with what part of the input space is covered by the ellipsoid, we contrast the center of this ellipsoid with the other two ellipsoids to find that it differs mostly from the other two by having a low Compactness (COM) and a lower Radius Ratio (RR). Looking at the COM values in the principal components factored by their radii, we find that this low COM value is true for the full ellipsoid as it is limited in this ellipsoid to stay below -0.2. We next try to find the properties of the points in the ellipsoid. Examining the PCA, what is particularly interesting is the last PC, one which expresses a constraint. In this PC all but the 4th and 5th elements, RR and Primary Axis Aspect Ratio (PAAR), have an almost negligible value. Since the value of these two significant variables is almost equal but of opposite sign, then we arrive at the approximate constraint: $RR - PAAR = 0 \Rightarrow RR = PAAR$, a direct strong limitation on class membership. The first PC, the one with the largest radius, also demonstrates that this constraint is the source of greatest variance for the set. In the first PC, the RR and PAAR elements have the most significant magnitude of any other variable. Their magnitude is also very similar between them, implying that they are correlated and move together across the full stretch of the large radius. Thus the group CD limits membership in the van class to points with a relatively low compactness, which are constrained to have an almost equal RR and PAAR.

The groups AB and BC share many common properties. First, their centers are very similar, hovering near zero for the first 4 variables and having a low Maximum Length Aspect Ratio (MLAR). In fact this low MLAR is a constraint for both of these groups. Looking, for example, at the group BC (the same is true for AB), whose last two PCs act as constraints, we can see that other than the MLAR the variables in the two constraints are mirrors of each other, their magnitudes are close but their signs are reversed. Thus, adding the two constraints together, the other variables cancel out and we conclude that the MLAR is constrained to be zero (relative to the center of the ellipsoid). Note that this is also borne out in the larger radius PCs, where the magnitude of the MLAR is negligible. Another point of similarity between these two groups is their first PC. This PC has a very large radius, allowing for the Circularity (CIR) and Distance Circularity (DC) to move together through almost their full value range, between -1 and 1. There is also some correlation with COM across this major axis of the ellipsoid. This major axis is obviously a contribution from the points in the B group which also has a very similar first PC.

Given these 3 large similarities between AB and BC, what are the differences that put them into two separate convex regions? While these differences are hidden in the respective PCs of these groups, it is easier to examine their two unconnected subcomponents, A and C. The centers of A and C convey that these two groups are from largely different regions of the input space. Where A is located to enclose points with a high COM, CIR, DC and relatively low PAAR, C encloses points with a low COM, relatively low CIR and DC, and a relatively higher PAAR. Other than that important difference, the constraints of these two groups are similar–both constraining the MLAR to zero, with a few small variations. The fact that their kernel is similar (the subspace defined by their constraints) implies that these ellipsoids have a similar orientation in the input space. Nevertheless, we can still discern an important difference between them by looking at their first PC. Where as group A's PC shows a positive correlation between COM and CIR, group C's PC shows a negative correlation. Thus, in the direction of maximal change, these groups show an opposing relationship between these variables (as well as others).

In summary, using the PCA analysis on the scatter matrix of the ellipsoids we saw that the CD ellipsoid primarily addresses data points with a low COM and varying MLAR, by constraining membership in the class to points having similar RR and PAAR values. We then saw that the AB and BC ellipsoids both allow their CIR, DC and to some degree their COM to vary together a great deal under the constraint that their MLAR stays constant. Their main differences stem from the contributions of the A and C groups which tackle points in the opposite extremes of COM, CIR and DC by imposing variations on the relationships of these variables. Thus, we conclude from the analysis that the classifier's construction of a concave decision region facilitates imposing a different classification strategy on the different parts of the input space.

6.5. The ellipsoids as a model

In our previous analysis we constructed ellipsoids to enclose the points belonging to each of the composite groups in order to analyze them using PCA. In doing so we have indirectly constructed an alternative classifier, the model which consists of these ellipsoids. That is, we can take an unclassified point in the input space and check whether it is on the inside of any of the ellipsoids, if so classify it as belonging to the van class.

We now compare this model with the original SVM model. Where the SVM model achieved 93.9% on the test set, the ellipsoid model achieved 91.2%. Of the 296 entries in the test set the output of the two models agreed on 254 entries, 85.8%. The SVMs average positive output response was 5.64 and −10.8 for negative outputs. For the points where the models disagreed the SVMs average positive output was 2.84 and the average negative output was −2.59. The fact that these responses are closer to zero implies that these points of contention between the models are points which are close to the SVM's boundary. We would not expect the ellipsoids to be an exact match to the SVM model, differences in the underlying forms of the decision boundaries, and limited information about the exact nature of the SVM decision boundary would preclude that. However, as an approximation the ellipsoid model gives a reasonable match to the SVM, capturing a large part of the essence of its classification strategy.

Another SVM was trained using the same kernel on the full 18 input classification problem (56 support vectors). It achieved 97.6% accuracy on the test set. The connectivity analysis of this classifier showed that its decision strategy with respect to the training set consists of one large convex region. Thus, in the process of adding input variables, some of the concave structure present in the lower dimensional model was removed. There could be different reasons for this, but it is a fair assumption that the "Curse of Dimensionality"

(Bishop, 1995), the fact that as we increase the input dimensions the problem becomes exponentially less specified, is involved. This allows for a structurally simpler model (one convex decision region as opposed to a concave one composed of 4 parts) to fit the data, as the added dimensionality loosens the restrictions on the shape of the decision regions.

Fitting a minimum volume ellipsoid to the data gave a classifier with 87.84% accuracy on the test data. However, this model does not take into account any margin information (where to put the boundary between the van and other classes). We took a simple, relatively naive approach to this, just expanding the model by a factor. As such, the ellipsoid would remain with the same center and relative axes proportions, but we would expand or shrink it appropriately. Rather than using an absolute factor we calculated the factor that it would take to bring the ellipsoid to just touch the nearest member of the other class outside the ellipsoid, and normalized the factors to that value. Thus, a factor of 1 corresponds to just touching the first member of the other class. Using the test set for validation the ellipsoid was expanded by a factor of 2.4, giving an accuracy of 96.96% on the test set, misclassifying only two examples more than the SVM.

Even though the 18 input SVM model did not display the same structure as its lower dimensional counterpart, that structure can still be applied to the 18 input problem. Consider the connectivity graph as a way to organize the data points. In essence it defines which points go together in convex decision regions. Thus, we can build the same ellipsoid model used in the 6 input case, in terms of which points to place in which ellipsoid, but use the full 18 dimensions of the input for the MVE construction. Doing so renders a model with 80.4% accuracy on the test set. Using a factor of 1 to adjust the margins of all three ellipsoids gives a model with 97.97% accuracy, and validating with respect to the test set (factor = 1.15) gives a model with 98.99% accuracy on the test set. That is an improvement from 2.4% error for the SVM to 1% error. Even though it is hard to draw strong conclusions from this result it does speak to an important issue in decision region based classifiers, over-generalization. The model with one convex decision region can perform well on the test data. However, ultimately the task of the classifier is to define the class set, what it means to be a van. By using one convex region to enclose all the points, the model allowed the class to be, at the very least, any point between the vans represented in the training data. Thus, rather than finding what makes a van, it found what makes a van not an Opel, Saab or truck. In a world with more than four types of cars the model would most probably misclassify other inputs which fell into its decision region. In the lower dimensional case, the training data was denser (relative to the dimensionality of the input space) and forced greater constraints on the shape of the decision regions. By using this additional structure in the higher dimensional case we may be getting closer to what it means to be a van.

7. Analysis method details and discussion

7.1. Low-level analysis

In the low-level analysis stage the connectivity graph is constructed by sampling between the sample points. There is only one parameter that may be varied at this stage, the sampling rate on the line. Unlike DSP applications where assumptions can be made about the data

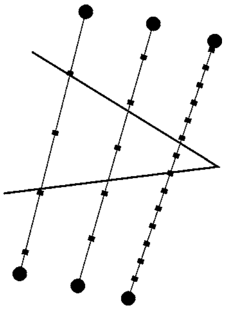


Figure 17. There is no absolute sampling frequency which will guarantee the detection of all decision boundaries. In this figure we see that the closer we come to the corner the higher the sampling rate must be to detect the boundary.

source allowing use of the Nyquist frequency to sample, in our particular case there is no one correct frequency, as illustrated in figure 17. In the figure we see that as our two sample points approach the decision boundary at the corner, we will need a continuously increasing sampling rate to detect the output transition. The consequence of losing that transition is that the two sample points may seem connected when there is in fact a region between them with a different output. Thus, we can never lose the fact that two points are connected, only that there may be a hole between them. Increasing the sampling frequency increases our sensitivity to smaller and smaller holes. From an applied perspective, ultimately it comes to the question of how small a transition is important to detect. Typically, the heuristic used is to try a few sampling frequencies until an increase in the sampling rate does not significantly change the connectivity graph.

Another decision to be made is what points to use. In this article we have used the points of the training set. However, there is no reason to be limited to only those points. One can also use the points of the test set, validation set, and unlabeled data. In addition it is possible to generate additional points to explore the classifier's response in under represented parts of the input space. For example, if we were interested in the shape that the decision region has between two classes (at the margin), we might generate new points by sampling between the two classes to find points near the boundary to be used for the connectivity analysis.

7.2. Higher-level analysis

At the higher-level analysis stage we want to partition the unconnected subgraphs of the connectivity graph into convex (or almost convex) subregions in order to understand the concavity in the represented regions, as well as to extract the interior points of extended convex subregions for further analysis. To this purpose the algorithm described in Section 4 allows us to partition the graph into a network of interconnected subregions, based on the connectivity signature of points as it reflects their membership in larger convex regions.

The main parameter influencing partitioning characteristics is α, which acts as the cutoff value for the Hamming distance. Two vertices will be grouped together if the percentage of their connectivity signatures that is identical is greater than α. On the one hand this parameter

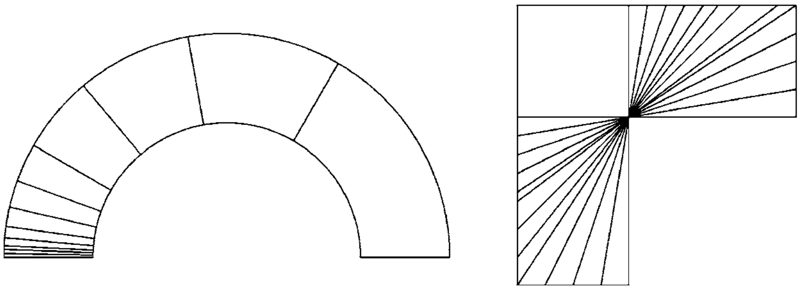


Figure 18. A example of two regions where the granularity of their partitioning influences how well their concavity is conveyed or the number of larger convex regions detected.

confers a certain robustness, allowing vertices with similar but not identical connectivity signatures to be grouped together, and as such, to gloss over small noise or irregularities in the decision region structure. But more importantly the α parameter guides the *granularity* of the partitioning, how finely it captures the curvature of the concavity or how sensitive it is to possible larger convex regions.

To understand the issues that granularity encompasses consider the concave regions in figure 18. The left region is an example of a region that can not be partitioned into a finite number of convex subregions, as any subregion that includes a non-point part of the lower border would be concave. As seen in the figure, in approximating a partitioning we effectively choose what degree of concavity is acceptable for a subregion, thereby controlling the number of subregions used to describe the region. In contrast, the right region in figure 18 can be partitioned using a finite number of convex subregions. However, the lines inside the figure illustrate that an infinite number of convex subregions are bounded within it. In partitioning this figure we must choose our sensitivity to all these potential larger convex regions. That is, we need to effectively decide how different two larger convex regions need to be for us to distinguish between them in the partitioning.

By acting as a limit on how different two points can be and still belong to the same group, the α parameter allows us to address these two granularity concerns. Figure 19 shows the effect of varying α on the quality of the partitioning for these two types of regions. These figures were made by first generating 200 random points in the interior of the region, calculating the connectivity graph on the region's raster image, and then running the partitioning algorithm at different values of α. For each group identified in the partitioning, the convex hull of the points was calculated and plotted to illustrate the location of the identified subregion. It is apparent that increasing the value of α increases the granularity of the partitioning, generating more groups. Notice how, for the more refined partitioning, the groups can be combined to form larger convex regions, easily supporting the construction of group graphs. In addition, note how the lines of the subregions support the shape of the larger regions. On the downside, with higher α values we run the risk of generating an overly complex group graph and also may lose more points that do not fit into any group directly. For intermediate values of α the region is partitioned into a smaller number of groups, but the groups themselves and their combination may exceed (hopefully

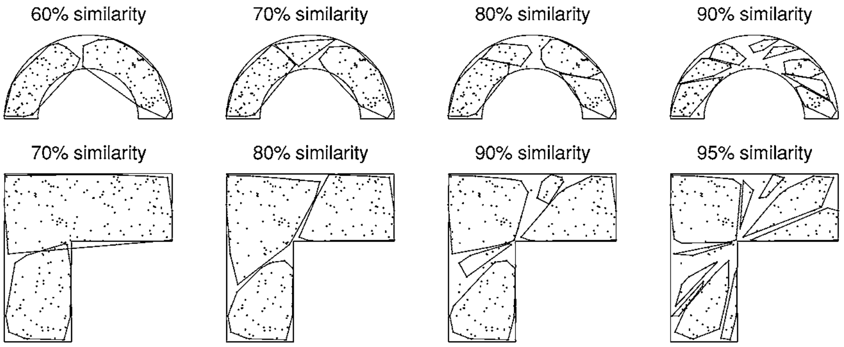


Figure 19. Examples of how different settings of the α parameter (Hamming similarity) influence the granularity of the partitioning.

slightly) the concave boundaries of the original region. With low α values, while almost all the points are grouped into a small number of distinct groups, the partitionings may not be refined enough to actually allow the construction of a group graph as the intergroup connectivity may be ambiguous.

The algorithm does not explicitly state how to pick the initial point from which to construct the group. In the previous examples this point was randomly selected from the pool of unassigned points. A random selection approach is obviously not deterministic, as multiple runs will generate different partitionings. When the Hamming similarity is high, the effects of this are only marginal as the groups are more distinctly defined. It is possible to employ a deterministic policy, for example to always select the point with the smallest hamming distance to the existing groups, or to pick points with a small or large number of connections. However, beyond providing a deterministic outcome, there is no clear advantage to these approaches, as the quality of a partitioning may improve or degrade with each of these depending on the context. In that regard, it is advantageous to compare multiple partitionings at the same α level, using the tools described next. In addition, the β parameter helps eliminate hapless partitionings by imposing a minimum on the number of elements in a group, if this is set high enough it avoids the formation of small spurious groups.

As discussed, partitioning involves a human element to experiment with different decomposition parameters and verify the qualities and integrity of the partitioning. In this process, feedback to the user about the partitioning is provided by group membership (e.g., number of points in a group) and the intergroup connectivity matrix G. Let i, j be the labels of two groups, let S_i be the set of vertices with label i, and define $n(l, S_i)$ to be the number of connections that vertex l has with group S_i. Then the matrix G is constructed as (where the equalities are due to symmetry of the connectivity matrix):

$$G_{ji} = G_{ij} = \frac{\sum_{l \in S_i} n(l, S_j)}{|S_i||S_j|} = \frac{\sum_{l \in S_j} n(l, S_i)}{|S_i||S_j|}$$

What G describes is how connected any two groups are. Each entry is a value between 0 and 1, where 1 implies that the two groups are 100% connected—all points in i are connected to all points in j and vice-versa. Typically most values are not 0 or 1, but somewhere in between. Usually we apply a cutoff, for example, groups with over 85% connectivity are considered connected while groups with less than 15% connectivity are considered disconnected.

The intergroup connectivity matrix can be used to evaluate a partitioning. It relates which groups are connected and can be combined to form larger convex regions (to form the group graph), which groups are not connected and which groups have ambiguous connectivity. That is, their intergroup connectivity value is somewhere between the cutoff values, not allowing us to explicitly state whether they are connected or not. As seen before the reason for this is that during the grouping procedure we may purposely seek large robust subregions by setting a relatively low value for the α parameter. However, as a side effect it may allow groups to form which consist of points that, at a coarse level, have a similar connectivity signature, but with respect to a few groups may have different connectivity.

A solution to this is to perform a refined grouping. After identifying two groups with ambiguous connectivity, we perform a second round of partitioning on the members of one of the groups. However, when we check the Hamming distance we do so only with respect to the members of the other group. What this does is to split the original group with respect to how the vertices are connected to the ambiguous second group. Typically, we may end up splitting the group up into two groups based on whether or not they are connected to the ambiguous group. The procedure may be repeated until we are left with an interpretable group connectivity matrix G, allowing us to draw a group graph of the relationships between the groups.

For example, the region in figure 20(a) contains a partitioning with a 70% Hamming similarity. Following is the rounded G matrix for this partitioning:

	A	*B*	*C*
A	1.0	0.1	0.5
B	0.1	1.0	0.5
C	0.5	0.5	1.0

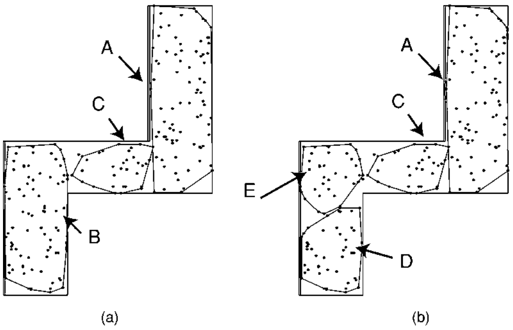


Figure 20. The partitioning on the left was conducted with a 70% Hamming similarity. The partitioning on the right is a refinement of group B into two subgroups that have a clearer connectivity.

From the matrix we see that while groups A and B are effectively not connected, as seen from the figure, there is ambiguity with respect to how group C is connected to A and B. We can additionally partition the points in group B with respect to how they connect to group C. By applying a 50% Hamming similarity as a cutoff, we split B into two groups, D and E, as seen in figure 20(b). Following is the new *G*, showing that the ambiguity was eliminated, but the connections between A, C and E need to be further refined.

	A	*D*	*C*	*E*
A	1.0	0.0	0.5	0.3
D	0.0	1.0	0.1	1.0
C	0.5	0.1	1.0	1.0
E	0.3	1.0	1.0	1.0

As described, the high-level analysis phase is inherently flexible. Depending on the context and the user's judgment, it allows for partitionings at different granularities and levels of refinement. It may be enough to generate a decomposition that captures the basic distinctive concavity of a region, or we may desire a decomposition that identifies small differences between the convex subregions. Using these tools, the user can make the choices most appropriate for their application.

7.3. *Convex region analysis*

In the convex analysis stage, we wish to locate which points go together in convex regions, and to apply a convex analysis methodology to those points. During the labeling of the various groups, some points that are part of convex regions may end up not being labeled. We call these *corner points*. As figure 21 illustrates these corner points are points that, due to their position in the decision region, may not have a connectivity signature that is shared by other points. Thus, before we start the convex analysis, we can go through each unlabeled point and check which groups it is highly connected to. Then, when we do the

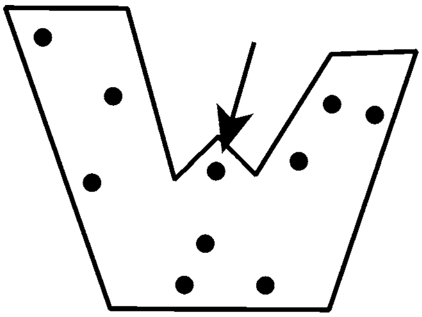


Figure 21. The point at which the arrow is pointing has a unique connectivity signature which would preclude including it in any group. However, it is part of the convex region that includes the bottom 3 points.

actual convex analysis we include those unlabeled points that are also highly connected to the group being analyzed.

In this paper we have used PCA on the covariance matrix of the data and PCA on the scatter matrix of the MVE. There are other methodologies that may be applied to analyze convex regions, for example convex regions may be approximated by axis-parallel boxes (Bespamyatnikh & Segal, 1997; Zhu, 1997), allowing a "rule like" representation of the data. Unfortunately, a detailed discussion of these and other methods are beyond the scope of this manuscript.

8. Extending and generalizing the method

The Decision Region Connectivity Analysis method (DRCA) as described here is applied to decision regions in the input space. At times we may want to interpret the classifier as first applying a coordinate transformation to the data (feature extraction, hidden unit space, kernel space, etc) and then enclosing the transformed data in decision regions. In such a case we can apply the DRCA method in this transformed space instead of the input space. For example, for a neural network we may apply the method to the data points only as they are represented in the first layer of hidden units, thus sampling along lines in the hidden-unit space instead of the input space.

It is also interesting to note that the only location in the basic algorithm that we make an assumption about the metric of the input space is in how we define what it means to sample "on the line". Thus, another possible modification to the algorithm would be to adjust this definition to reflect prior knowledge about how either the input space is organized or how the classifier interprets it. However, modifications of this sort reflect changes in how neighborhoods are defined, and as such would preclude the typical analysis of convex regions using PCA.

In general though, how do we extend this method to other types of classifiers? It may seem that we want to mathematically abstract the cornerstones of the method, introducing abstract notions of convexity and concavity. Singer (1997), describes this procedure as "One selects a certain property that usual convex sets in R^n have, but many other objects in possibly other settings also have, and one uses that property to define a 'generalized' sort of 'convexity"'. Even though this sounds vague it hints at the real question, why are we interested in convexity to begin with?

When a model is given training data, it is given a sampling of a category set, and asked to deduce the actual complete set. So if a model has generalized from training data then the class set it has constructed approximates the actual underlying category set. Thus the model builds a larger (probably infinite) set from a finite sample. Whether the model will generalize successfully depends on whether these larger sets are the right form of extrapolation from the sample. Our goal is to understand the model's class set, we want to describe these infinite sets that are derived from the sample. In a decision region type classifier, points are enclosed in contiguous volume filling areas of the input space. By examining where points are fully enclosed in convex portions of decision regions, we can infer how the model generalizes. These convex portions describe the unique, contiguous, volume filling parts of the input space where all the points belong to the class. Thus, convex regions are the essential mechanism by which these models construct infinite sets from finite samples. In order to adapt this method to a different type of classifier we would need to understand how it constructs infinite class sets from samples and find a method to recognize and analyze the structure of these sets.

9. Conclusion

Many classifiers, such as feed-forward networks, nearest-neighbor classifiers, support vector machines, decision trees and their ensembles, operate by constructing complex decision regions in the input space. These decision regions can be few or many, convex or concave, have large or small volumes, etc. By focusing on the sample points enclosed in these regions we have demonstrated a method with low computational complexity, DRCA, to extract these properties which is independent of the classifier type or the dimensionality of the input space. It thus allows us not only to analyze individual high-dimensional classifiers but to compare completely different classifier models on the same problems. We demonstrated this method on a number of examples: Analyzing a 3-dimensional neural network, allowing a comparison of the method with its actual decision region; Comparing a neural network and KNN classifier on a handwritten digit classification problem, and demonstrating fundamental differences in their generalization strategy; Analyzing a high-dimensional SVM model, and demonstrating how it partitioned its decision region to apply different classification strategies to different parts of the input space. In addition we have described how the approach may be used to transfer constraints across dimensions, as well as discussing points of potential expansion for the approach.

This method offers a significant opportunity in helping to unite a field with many models and approaches by giving an analysis tool which addresses their greatest common denominator, their method of generalization. Thus it allows the qualitative analysis of present and future high-dimensional classifiers, providing greater insight into these models and the problems they are applied to.

Acknowledgments

Thanks to Jordan Pollack and the members of the DEMO lab for all their support and assistance.

References

Alpaydin, E., & Kaynak, C. (1998). Cascading classifiers. *Kybernetika*, *34:4*, 369–374.

Bespamyatnikh, S., & Segal, M. (1997). Covering a set of points by two axis-parallel boxes. In *Proc. 9th. Canad. Conf. Comput. Geom.* (pp. 33–38).

Bishop, C. M. (1995). *Neural networks for pattern recognition.* Oxford: Oxford University Press.

Blake, C. L., & Merz, C. J. (1998). UCI repository of machine learning databases.

Bomze, I. M., Budinich, M., Pardalos, P. M., & Pelillo, M. (1999). The maximum clique problem. *Handbook of combinatorial optimization (Supplement Volume A)*, Boston, MA: Kluwer Academic Publishers.

Bruss, I., & Frick, A. (1995). Fast interactive 3-d graph visualization. In Brandeburg (Ed.), *Proceedings Workshop on Graph Drawing '95*, Lecture Notes in Computer Science 1027 (pp. 99–110). Berlin: Springer-Verlag.

Joachims, T. (1999). Making large-scale svm learning practical. In B. Schelkopf, C. Burges, & A. Smola (Eds.), *Advances in kernel methods—Support vector learning*. Cambridge, MA: MIT Press.

Jolliffe, I. T. (1986). *Principal component analysis*. Berlin: Springer-Verlag.

King, R. D., Feng, C., & Shutherland, A. (1995). Statlolg: Comparison of classification algorithms on large real-world problems. *Applied Artificial Intelligence*, *9:3*, 259–287.

Lovasz, L. (1986). *An algorithmic theory of numbers, graphs and convexity*. CBMS-NSF Regional Conference Series in Applied Mathematics Vol. 50. SIAM, Philadelphia.

Melnik, O., & Pollack, J. (2002). Theory and Scope of Exact Representation Extraction from Feed-Forward Networks. *Cognitive Systems Research, 3:2*.

Paz, A., & Moran, S. (1981). Non deterministic polynomial optimization problems and their approximations. *Theoretical Computer Science, 15*, 251–277.

Rousseeuw, P. J. (1997). Introduction to positive-breakdown methods. *Handbook of statistics* (Vol. 15) (pp. 101–121). Amsterdam: Elsevier.

Sabharwal, A., & Potter, L. C. (1995). Set estimation via ellipsoidal approximation. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*.

Shor N. Z., Berezovski (1992). New algorithms for constructing optimal circumscribed and inscribed ellipsoids. *Optimization Methods and Software, 1*, 283–299.

Siebert, J. P. (1987). Vehicle recognition using rule based methods. Technical Report, Turing Institute.

Singer, I. (1997). *Abstract convex analysis*. New York: Wiley-Interscience.

Thornton, C. (1997). Separability is a learner's best friend. In J. A. Bullinaria, D. W. Glasspool, & G. Houghton (Eds.), *Proceedings of the Fourth Neural Computation and Psychology Workshop: Connectionist Representations* (pp. 40–47). Berlin: Springer-Verlag.

Titterington, D. M. (1978). Estimation of correlation coefficients by ellipsoidal timming. *Appl. Statist., 27:3*, 227–234.

Zhu, B. (1997). Approximating the convex hull polyhedra with axis-parallel boxes. *International J. Comput. Geom. Appl.,* 7, 253–267.

Received August 11, 2000
Revised January 12, 2001
Accepted January 12, 2001
Final manuscript January 26, 2001