On detection of outliers and their effect in supervised classification

Edgar Acuña and Caroline Rodriguez
edgar@cs.uprm.edu, caroline@math.uprm.edu
Department of Mathematics
University of Puerto Rico at Mayaguez
Mayaguez, Puerto Rico 00680

Abstract

An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism (Hawkins, 1980). Outlier detection has many applications, such as data cleaning, fraud detection and network intrusion. The existence of outliers can indicate individuals or groups that have behavior very different from the most of the individuals of the dataset. Frequently, outliers are removed to improve accuracy of the estimators. But sometimes the presence of an outlier has a certain meaning, which explanation can be lost if the outlier is deleted.

In this paper we compare detection outlier techniques based on statistical measures, clustering methods and data mining methods. In particular we compare detection of outliers using robust estimators of the center and the covariance matrix for the Mahalanobis distance, detection of outliers using partitioning around medoids (PAM), and two data mining techniques to detect outliers: Bay's algorithm for distance-based outliers (Bay and Schwabacher, 2003) and the LOF a density-based local outlier algorithm (Breuning et al., 2000). The effect of the presence of outliers on the performance of three well-known classifiers is discussed.

1 Introduction.

According to Hawkins (1980), "An outlier is an observation that deviates so much from other observations as to arouse suspicion that it was generated by a different mechanism". Almost all the studies that consider outlier identification as their primary objective are in the field of statistics. A comprehensive treatment of outliers appears in Barnet and Lewis (1994). They provide a list of about 100 discordancy tests for detecting outliers in data following well known distributions. The choice of an appropriate discordancy test depends on: a) the distribution, b) the knowledge of the distribution parameters, c) the number of expected outliers, and d) the type of expected outliers. These methods have two main drawbacks: First, almost all of them are for univariate data making them unsuitable for multidimensional datasets. Second, all of them are distribution-based, and most of the time the data distribution is unknown. Real-world data are commonly multivariate with unknown distribution.

Detecting outliers, instances in a database with unusual properties, is an important data mining task. People in the data mining community got interested in outliers after Knorr and Ng (1998) proposed a non-parametric approach to outlier detection based on the distance of an instance to its nearest neighbors. Outlier detection has many applications among them: Fraud detection and network intrusion, and data cleaning. Frequently, outliers are removed to improve accuracy of the estimators. However, this practice is not recommendable because sometimes outliers can have very useful information. The presence of outliers can indicate individuals or groups that have behavior very different from a standard situation.

Section 2 of this paper includes a brief discussion of the treatment of outliers for univariate data. The section 3 focuses on

methods for detection of multivariate outliers. Four methods of outlier detection are considered: a method based on robust estimation of the Mahalanobis distance, a method based on the PAM algorithm for clustering, a distance-based method and a density-based method. The section 4 of this paper covers the effect and treatment of outliers in supervised classification. The experimental results appear in section 5, and the conclusions of our work are presented in section 6.

2. Univariate Outliers

Given a data set of n observations of a variable x, let $\bar{x}$ be the mean and let s be standard deviation of the data distribution. One observation is declared as an outlier if lies outside of the interval

$$(\bar{x} - ks, \bar{x} + ks). \qquad (1)$$

where the value of k is usually taken as 2 or 3. The justification of these values relies on the fact that assuming normal distribution one expects to have a 95% (99%, respectively) percent of the data on the interval centered in the mean with a semi-length equal to two (three, respectively) standard deviation. Also, one expects to have the whole data inside an interval centered at the mean and three standard deviations as semi-length. From equation (1), the observation x is considered an outlier if

$$\frac{|x - \bar{x}|}{s} > k. \qquad (2)$$

The problem with the above criteria is that it assumes normal distribution of the data something that frequently does not occur. Furthermore, the mean and standard deviation are highly sensitive to outliers.

John Tukey (1977) introduced several methods for exploratory data analysis, one of them was the *Boxplot*. The *Boxplot* is a graphical display where the outliers appear tagged. Two types of outliers are distinguished: *mild outliers* and *extreme outliers*. An observation x is declared an *extreme outlier* if it lies outside of the interval (Q_1-3×IQR, Q_3+3×IQR). Notice that the center of the interval is (Q_1+Q_3)/2 and its radius is 3.5×IQR, where IQR=Q_3-Q_1 is called the *Interquartile Range* and can be considered a robust estimator of variability which can replace s in equation (2). On the other hand, (Q_1+Q_3)/2 is a robust estimator of the center that can be used instead of $\bar{x}$ in equation (1). An observation x is declared *a mild outlier* if it lies outside of the interval (Q_1-1.5×IQR, Q_3+1.5×IQR). The numbers 1.5 and 3 are chosen by comparison with a normal distribution. All major statistical software applications include boxplots among their graphical displays. Figure 1 shows the outliers detected through their boxplots of the features in the class 1 (*setosa*) of the very well known *Iris* dataset, which has 150 instances and three classes.

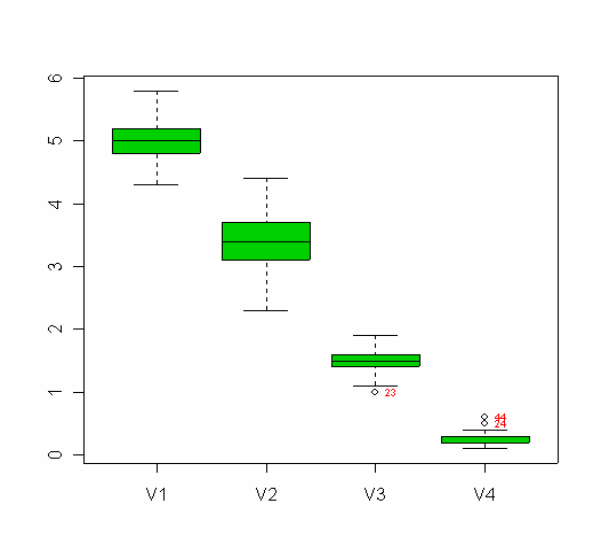

Figure 1. Outliers of the features in class 1 of the *Iris* data set

Using the same graphical display we detect as outliers the instance number 99 in the second class which has an abnormal value in the third feature and the instance 107 that has an abnormal value in the first feature of class 3.

3. Multivariate Outliers

Let us consider a dataset D with p features and n instances. In a supervised classification context, we must also know the classes to which each of the instances belongs. It is very common to include the classes as the last column of the data matrix. The objective is to detect all the instances that seem to be unusual, these will be the multivariate outliers. One might think that multivariate outliers can be detected based on the univariate outliers in each feature, but as shown in the figure 2 this is not true. The instance appearing in the upper right corner is a multivariate outlier but it is not an outlier in each feature. On the other hand, an instance can have values that are outliers in several features but the whole instance might not be a multivariate outlier.

There are several methods to detect multivariate outlier. The methods discussed in this paper are: statistical-based outlier detection, outlier detection by clustering, distance-based outlier detection and density-based local outlier detection. The before mentioned methods are discussed in the next sections.

3.1. Statistical based outlier detection.

Let **x** be an observation of a multivariate data set consisting of n observations and p features. Let $\bar{\mathbf{x}}$ be the centroid of the dataset, which is a p-dimensional vector with the means of each feature as components. Let **X** be the matrix of the original dataset with columns centered by their means. Then the p×p matrix **S**=1/(n-1) **X'X** represents the covariance matrix of the p features.The multivariate version of equation (2) is

$$D^2(\mathbf{x},\bar{\mathbf{x}}) = (\mathbf{x}-\bar{\mathbf{x}})\mathbf{S}^{-1}(\mathbf{x}-\bar{\mathbf{x}}) > k\,. \qquad (3)$$

where D^2 is called the Mahalanobis square distance from **x** to the centroid of the dataset. An observation with a large Mahalanobis distance can be considered as an outlier.

Assuming that the data follows a multivariate normal distribution it can be proved that the distribution of the Mahalanobis distance behaves as a Chi-Square distribution for a large number of instances. Therefore the proposed cutoff point in (3) is given by $k=\chi^2_{(p,1-\alpha)}$, where χ^2 stands for the Chi-Square distribution and α is a signification level usually taken as .05.

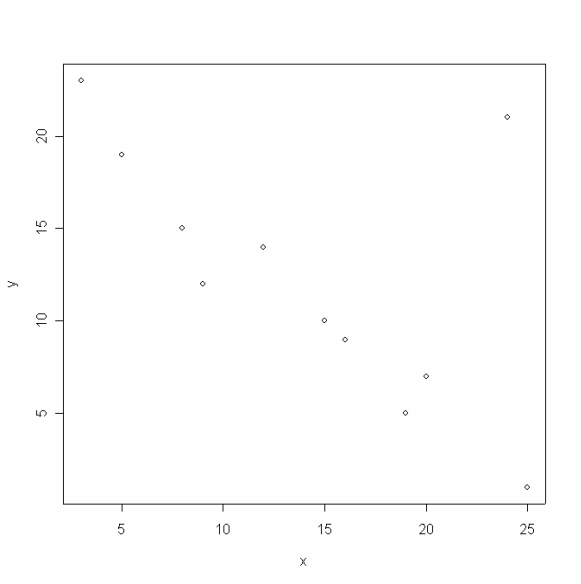

Figure 2. Example of a bidimensional outlier that is not an outlier in either of its projections.

A basic method for detecting multivariate outliers is to observe the outliers that appear in the *boxplot* of the distribution of the Mahalanobis distance of the all instances.

Looking at figure 3 we notice that only two outliers (instances 119 and 132) are detected in class 3 of the *Iris* dataset. People in the data mining community prefer to rank the instances using an outlyingness measures rather than to classify the instances in two types: outliers and non-outliers. Rocke and Woodruff (1996) stated that the Mahalanobis distance works well identifying scattered outliers. However, in data with clustered outliers the Mahalanobis distance measure does not perform well detecting outliers. Data sets with multiple outliers or clusters of outliers are subject to the *masking* and *swamping* effects.

Masking effect. It is said that an outlier masks a second one that is close by if the latter can be considered an outlier by itself, but not if it is considered along with the first one.

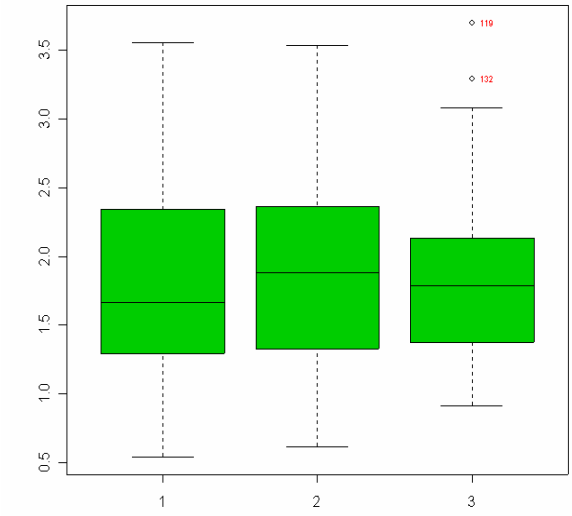

Figure 3. Boxplots of the Mahalanobis distances for each instance in the three classes of the *Iris* dataset

Equivalently after the deletion of one outlier, the other instance may emerge as an outlier. Masking occurs when a group of outlying points skews the mean and covariance estimates toward it, and the resulting distance of the outlying point from the mean is small.

Swamping effect. It is said that an outlier swamps another instance if the latter can be considered outlier only under the presence of the first one. In other words after the deletion of one outlier, the other outlier may become a "good" instance. Swamping occurs when a group of outlying instances skews the mean and covariance estimates toward it and away from other "good" instances, and the resulting distance from these "good" points to the mean is large, making them look like outliers.

For instance, consider the data set due to Hawkins, Bradu, and Kass (Rousseeuw and Leroy, 1987) consisting of 75 instances and 3 features, where the first fourteen instances had been contaminated to be outliers. Using the Mahalanobis distance only observation 14 is detected as an outlier as shown in Figure 4. The remaining 13 outliers appear to be masked.

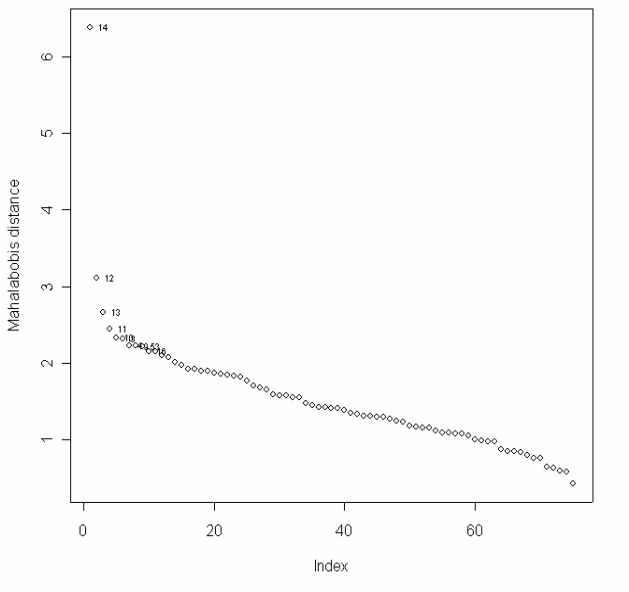

Figure 4: The Masking effect of multivariate outliers in the Hawkins data set

The masking and swamping problem can be solved by using robust estimates of the centroid (location) and the covariance matrix (shape), which by definition, are affected less by outliers.

Outlying points are less likely to enter into the calculation of the robust statistics, so they will not be able to influence the parameter estimates used in the Mahalanobis distance. Among the robust estimators of the centroid and the covariance matrix are the minimum covariance determinant (MCD) and the minimum volume ellipsoid (MVE) both of them introduced by Rousseeuw (1985).

The *Minimum Volume Elipsoid (MVE) estimator* is the center and the covariance of a subsample size h ($h \leq n$) that minimizes the volume of the covariance matrix associated to the subsample. Formally,

$$\text{MVE}=(\bar{\mathbf{x}}_J^*, S_J^*), \quad (4)$$

where

J={set of h instances: $Vol(S_J^*) \leq Vol(S_K^*)$ for all K s. t. #(K)= h}. The value of h can be thought of as the minimum number of instances which must not be outlying and usually h=[(n+p+1)/2], where [.] is the greatest integer function.

On the other hand the *Minimun Covariance Determinant (MCD) estimator* is the center and the covariance of a subsample of size h ($h \leq n$) that minimizes the determinant of the covariance matrix associate to the subsample. Formally,

$$\text{MCD}=(\bar{\mathbf{x}}_J^*, S_J^*), \quad (5)$$

where

J={set of h instances: $|S_J^*| \leq |S_K^*|$ for all K s. t. #(K)= h}. As before, it is common to take h=[(n+p+1)/2].

The MCD estimator underestimates the scale of the covariance matrix, so the robust distances are slightly too large, and too many instances tend to be nominated as outliers. A scale-correction has been implemented, and it seems to work well. The algorithms to compute the MVE and MCD estimators are based on combinatorial arguments (for more details see Rousseeuw and Leroy, 1987). Taking in account their statistical and computational efficiency, the MCD is preferred over the MVE.

In this paper both estimators, MVE and MCD, have been computed using the function **cov.rob** available in the package **lqs** of **R**. This function uses the best algorithms available so far to compute both estimators.

Replacing the classical estimators of the center and the covariance in the usual Mahalanobis distance, equation (3), by either the MVE or MCD estimator, outlying instances will not skew the estimates and can be identified as outliers by large values of the Mahalanobis distance. The most common cutoff point k is again the one based in a Chi-Square distribution, although Hardin and Rocke (2004) propose a cutoff point based on the F distribution that they claim to be a better one.

In this paper, two strategies to detect outliers using robust estimators of the Mahalanobis distances have been used: First, choose a given number of instances appearing at the top of a ranking based on their robust Mahalanobis measure. Second, choose as a multivariate outlier the instances that are tagged as upper outliers in the *Boxplot* of the distribution of these robust Mahalanobis distances.

In order to identify the multivariate outliers in each of the classes of the *Iris* dataset through boxplots of the distribution of the robust version of the Mahalanobis distance, we consider 10 repetitions of each algorithm obtaining the results shown in tables 1 and 2.

Table 1. Top outliers per class in the Iris dataset by frequency and the outlyingness measure using the MVE estimator

Instance	Class	Frequency	Outlyingness
44	1	8	5.771107
42	1	8	5.703519
69	2	9	5.789996
119	3	8	5.246318
132	3	6	4.646023

Notice that both methods detect two outliers in the first class, but the MVE method detects the instance 42 as a second outlier whereas the MCD method detects the instance 24. All the remaining outliers detected by both methods are the same. Three more outliers are detected in comparison with the use of the Mahalanobis distance.

The figure 5 shows a plot of the ranking of the instances in class 3 of the *Iris* dataset by their robust Mahalanobis distance using the MVE estimator.

Table 2. Top outliers in class 1 by frequency and the outlyingness measure using the MCD estimator

Instance	Class	Frequency	Outlyingness
44	1	10	6.557470
24	1	10	5.960466
69	2	10	6.224652
119	3	10	5.390844
132	3	7	4.393585

The Figure 6 shows a plot of the ranking of the instances in class 3 of *Iris* by their robust Mahalanobis distance using the MCD estimator.

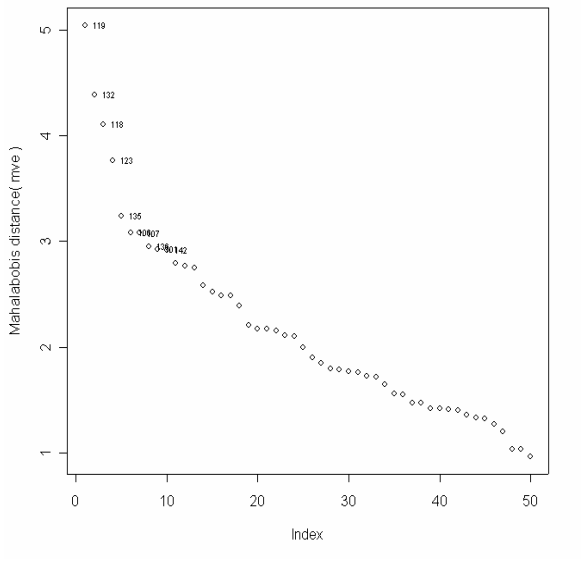

Figure 5. Plot of the instances ranked by their Mahalanobis distance using MVE estimator

According to Rocke and Woodruff (2002) robust methods work well detecting scattered outliers but fail to detect clustered outliers. For this type of outliers it is better to use a clustering algorithm as will be discussed in the next section.

3.2. Detection of outliers using clustering

A clustering technique can be used to detect outliers. Scattered outliers will form a cluster of size 1 and clusters of small size can be considered as clustered outliers. There are a large number of clustering techniques. In this paper we only considered the Partitioning around Medoids (PAM) method. It was introduced by Kaufman and Rousseeuw (1990) uses k-clustering on medoids to identify clusters. PAM works efficiently on small data sets, but it is extremely costly for larger ones. This led to the development of CLARA (Clustering Large Applications) (Kauffman and Rousseuw, 1990) where multiple samples of the data set are generated, and then PAM is applied to the samples.

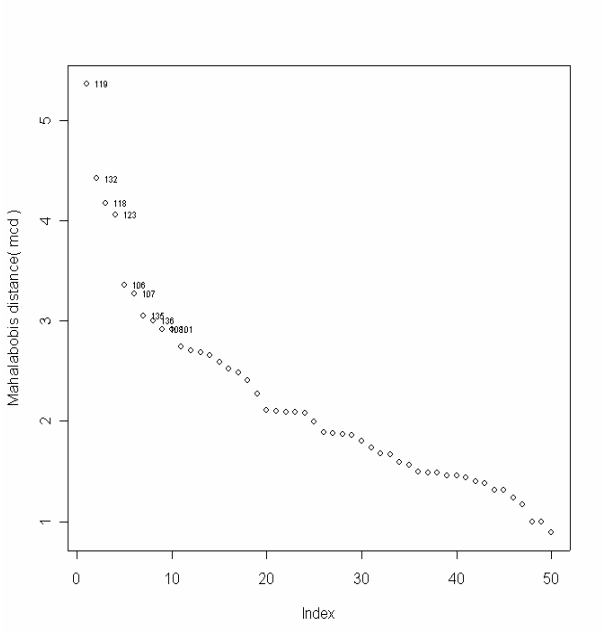

Figure 6. Plot of the instances ranked by their Mahalanobis distance using MCD estimator

Given k, the number of partitions to construct, the PAM method creates an initial partitioning. It then uses an iterative relocation technique that attempts to improve the partitioning by moving instances from one group to another. The

general criterion of good partitioning is that instances in the same cluster are "close" or related to each other, whereas instances of different clusters are '"far apart" or very different.

In order to find k clusters, PAM's approach is to determine a representative instance for each cluster. This representative instance called *medoid*, is meant to be the most centrally located instance within the cluster. More specifically, a *medoid* can be defined as that instance of a cluster, whose average dissimilarity to all the objects in the cluster is minimal. After finding the set of *medoids*, each object of the data set is assigned to the nearest *medoid*.

The PAM algorithm's complexity is $O(k(n-k)^2)$. Hence it becomes too costly for large values of *n* and *k*. PAM is very robust to the presence of outliers and does not depend on the order in which instances are examined. After the allocation of the instances to the k clusters, one must determine the *separation* between them. The *separation* of the cluster *C* is defined as the smallest dissimilarity between two objects; one of which belong to Cluster *C* and the other which does not. If the separation of an outlier is large enough then all the instances that belong to the cluster are considered outliers. In order to detect the clustered outliers one must vary the number k of clusters until clusters of small size are obtained that have a large separation from others clusters. The PAM algorithm can be evaluated using the function **pam** available in the library **cluster** in R.

Looking at the separation measures of ten clusters generated by the PAM algorithm in each of the classes of the *Iris* dataset, we detected the outliers shown in table 3. Notice that in the third class, PAM detects the instance number 107 as an outlier but does not detect the instance 119.

3.3. Distance based outlier detection

Given a distance measure on a feature space, two different definitions of distance-based outliers are the following:

Table 3. Outliers in the Iris dataset according to the PAM algorithm

Instance	Class	Separation
42	1	0.6244998
58	2	0.6480741
61	2	0.6480741
94	2	0.6480741
99	2	0.6480741
107	3	0.9110434
118	3	0.8185353
132	3	0.8185353

1. An instance **x** in a dataset D is an outlier with parameters p and λ if at least a fraction p of the objects are a distance greater than λ from **x**. (Knorr and Ng, 1997, 1998, Knorr et al. 2000). This definition has certain difficulties such as the determination of λ and the lack of a ranking for the outliers. Thus an instance with very few neighbors within a distance λ can be regarded as strong outlier as an instance with more neighbors within a distance λ. Furthermore, the time complexity of the algorithm is $O(kn^2)$, where k is the number of features and n is the number of instances. Hence it is not an adequate definition to use with datasets having a large number of instances.

2. Given the integer numbers k and n ($k<n$), outliers are the top n instances with the largest distance to their k-th nearest neighbor. (Ramaswamy et al., 2000). One shortcoming of this definition is that it only considers the distance to the k-th neighbor and ignores information about closer points. An alternative is to use the greatest average distance to the k nearest neighbors. The drawback of this alternative is that it takes longer to be calculated.

In this paper a variant of one recently developed algorithm (Bay and Schwabacher, 2003) for distance-based outlier detection has been used.

The Bay's Algorithm.

Bay and Schwabacher (2003) proposed a simple nested loop algorithm that tries to

reconcile definitions 1 and 2. It gives near linear time performance when the data are in random order because a simple pruning rule is used. The performance of the algorithm in the worst case is of quadratic order. The algorithm is described in Figure 7.

Input: k: number of nearest neighbors; n: number of outliers to return; D: dataset randomly ordered, BS: size of blocks in which D is divided.
1. Let distance(x,y) return the Euclidean distance between x and y. Another type of distance such as the Manhattan can be used instead of the Euclidean distance.
2. Let maxdist(x,Y) return the maximum distance between the instance x and the set of instances Y.
3. Let Closest(x,Y,k) return the k closest instances in Y to x.
4. Let score(x) return median distance to the k neighbors
5. Begin
c← 0 Set the cutoff for pruning to 0.
O←ϕ Initializate the set of outliers as the empty set.
NB←ceiling(# instances in D/BS)
While nb<NB {
Neighbors(b)←ϕ for all b in B(nb)
For each d in D {
For each b in B, b≠d{
If |Neigbors(b)|<k or distance(b,d)<maxdist(b,Neighbors(b))){
Neighbors(b) ←Closest(b,Neighbors(b)Ud,k)
If(score(Neighbors(b),b)<c{
Remove b from B(nb)
}}}}
O←Top(B(nb) ∪ O, n) ; Keep only the top n outliers
c←min(score(o)) for all in O ; The cutoff is the score of the weakest outlier
}
end

Output: O, a set of outliers

Figure 7. Bay's Algorithm for finding distance-based outliers

The main idea in the algorithm is that for each instance in D one keeps track of the closest neighbors found so far. When an instance's closest neighbors achieve a score lower than a cutoff then the instance is removed because it can no longer be an outlier. In this paper the score function used has been the median distance to the k neighbors. Bay used the average distance to the k neighbors, but the median is more robust than the mean. As more instances are processed the algorithm finds more extreme outliers and the cutoff increases along with pruning efficiency.

Bay and Schwabacher showed experimentally that the Bay's algorithm is linear with respect to the number of neighbors and that is almost linear with respect to the number of instances. Using six large datasets they found a complexity of order $O(n^{\alpha})$ where α varied from 1.13 to 1.32. Working with three datasets: *Ionosphere*, *Vehicle* and *Diabetes* we have obtained an α value near to 1.5.

The top 20 outliers in the third class of the *Iris* dataset according to the Bay's algorithm are shown in Figure 8. Clearly the instance 107 is detected as an outlier. There is a second group that includes 119, 120, 132, 123 and 118.

3.4. Density-based local outliers.

In this type of outliers the density of the neighbors of a given instance plays a key role. Furthermore an instance is not explicitly classified as either outlier or non-outlier; instead for each instance a local outlier factor (LOF) is computed which will give an indication of how strongly an instance can be an outlier.

The figure 9 taken from Breuning et al.(2000) shows the weakness of the distance-based method which identifies as outlier the instance o_1, but does not consider o_2 as an outlier.

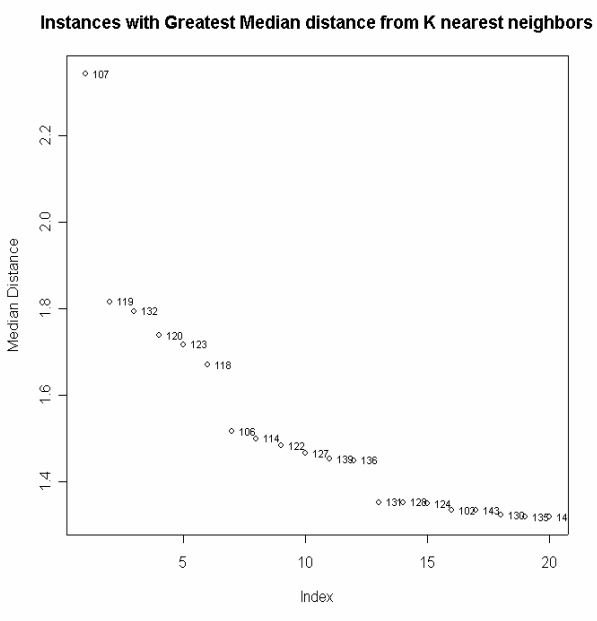

Figure 8. Instances of class 3 of the Iris dataset ranked by the Bay's algorithm outlyingness measure

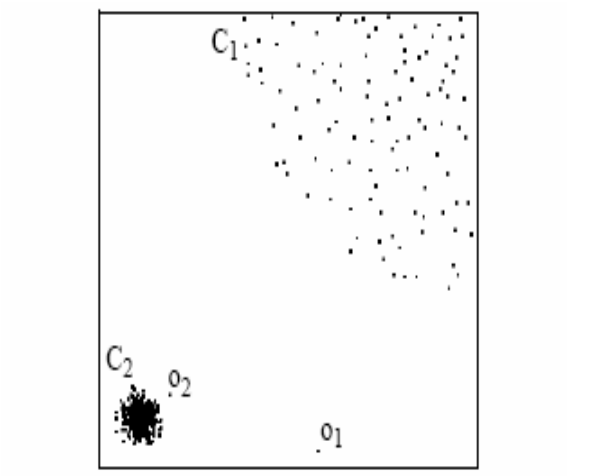

Figure 9. Example to show the weakness of the distance-based method to detect outliers

The following two definitions are needed in order to formalize the LOF algorithm

Definition 1. *k-distance of an instance x*

For any positive integer k, the k-distance of an instance x, denoted by k-distance(x), is defined as the distance d(x,y) between x an instance y ε D such that:

(i) for at least k instances y' ε D-{x} it holds that $d(x,y') \leq d(x,y)$

(ii) for at most k-1 instances y' ε D-{x} it holds that $d(x,y') < d(x,y)$.

Definition 2. *Reachability distance of an instance x w.r.t. instance y*

Let k be a positive integer number. The reachability distance of the instance x with respect to the instance y is defined as

$$\text{reach-dist}_k(x,y)=\max\{\text{k-distance}(y),d(x,y)\} \quad (6)$$

The *Local outlier factor (LOF) of an instance x is defined by*

$$LOF_{MinPts}(x)=[\frac{\sum_{y\in N_{MinPts}(x)}\frac{lrd_{MinPts}(y)}{lrd_{MinPts}(x)}}{|N_{MinPts}(x)|}]^{-1} \quad (7)$$

where *lrd(.)* represents the *Local reachability density of an instance.* Given an instance x, its *lrd* is defined as the inverse of the average *reachability distance* based on the *MinPts*-nearest neighbor of the instance x.

The density-based local algorithm to detect outliers requires only one parameter, *MinPts*, which is the number of nearest neighbors used in defining the local neighborhood of the instance.

The LOF measures the degree to which an instance x can be considered as an outlier. Breunig et al. show that for instances deep inside a cluster their LOF's are close to 1 and should not be labeled as a local outlier. Since LOF is not monotonic, they recommended to find the LOF for each instance of the datasets using MinPts-nearest neighbor, where MinPts assumes a range of values from MinPtsLB to MinPtsUB. They also suggested MINPtsLB=10 and MinPtsUB=20. In this paper, LOF were found in a range of MinPtsLB=10 and MinPtsUB=20 or

MinPtsLB=20 and MinPtsUB=30, depending on which range produced a more monotonic graph. Having determined MInPtsLB and MinPtsUB, the LOF of each instance is computed within this range. Finally all the instances are ranked with respect to the maximum LOF value within the specified range. That is, the ranking of an instance x is based on:

$$\text{Max}\{LOF_{MinPts}(x) \quad \text{s.t.} \quad MinPtsLB \leq MinPts \leq MinPtsUB\} \quad (8)$$

The LOF algorithm is shown in figure 10.

Input: Dataset D, MinptsLB, MinptsUB
Let Maxlofevct=ϕ
For each i in the interval [MinPtsLB, MinPtsUB]
{
1. Find the i nearest neighbors and their distance from each observation in D
2. Calculate the local reachability density for each observation in D
3. Compute the lof of each observation in D
4. Maxlofvect=max(maxlofvect, lof)
}
end
Output: Maxlofvect

Figure 10. The LOF Algorithm

Breunig et al. state that the time complexity of the maxLOF algorithm depends on the dimensionality of the data and it can be analyzed by studying independently the time complexity of the two main steps required to produce the LOF factor for each instance of the dataset. The first step, finding the k-distance neighborhood, has a runtime complexity of O(*n**time for a k-nn query). Therefore, the actual time complexity of this step is determined by the method used to perform the k-nn queries. For low dimensionality (no more than 5 features), if a grid based approach is used the query can be performed in constant time leading to a complexity of O(*n*) to complete the entire step. For medium dimensionality (between 5 and 10 features), an index can be used that would provide an average complexity of O(log *n*) for the k-nn queries, leading to a total complexity of O(*n*log*n*). Finally, for high dimensional data, a sequential scan may be used with a complexity of O(*n*) that would lead to a total complexity of $O(n^2)$. Finding the maximum outlier factors of all observations in the dataset can be done in linear time. Using the *Ionosphere* dataset, which has 32 features and 351 instances, the time complexity estimated by us was $O(n^{1.95})$.

In figure 11 we show the top 10 outliers of the third class in the *Iris* dataset using the LOF algorithm. Clearly the instance 107 is detected as an outlier. There is a second group that includes 119, 118, 132 and 123. After this group, instance 106 is chosen.

4. Effect of the treatment of outliers in supervised classification.

In the literature, it is frequently mentioned that the presence of outliers affects the performance of a classifier, but there are few studies verifying such claim. This is not the case in a regression context where there are a large number of studies showing the effect of outliers in regression problems. Two main aspects to consider in supervised classification are feature selection and the estimation of the misclassification error rate. In this paper, an evaluation of the effect of outliers in those aspects is considered. We have considered three classifiers: Linear discriminant analysis (LDA), k-nearest neighbor classifier and a classifier based on decision trees named Rpart. We use 10-fold cross validation as the estimation method for the misclassification error rate. There are plenty of feature selection methods and we choose two that have given us good results in our previous work (Acuna et al. 2003). One is the sequential forward selection (SFS) method and the second one is the Relief. The first one is a wrapper method that requires a classifier and the second one is a filter method that does not require a classifier and usually selects more features than a wrapper. After the detection of outliers we want to see how they affect a) the estimation of the misclassification error rate, b)

the feature selection process and, c) the misclassification error rate after feature selection.

5. Experimental results

Two well-known Machine Learning datasets: *Iris and Bupa* are used to show the effect of outliers on feature selection and on the estimation of the misclassification error rate. These datasets are available on the Machine Learning database repository at the University California, Irvine (Blake and Mertz, 1998).
Using all the criteria described in section 3 and the visual help provided by the parallel coordinate plot (Wegman, 1990) to decide about the doubtful outliers, the following outliers have been detected in the *Iris* dataset.

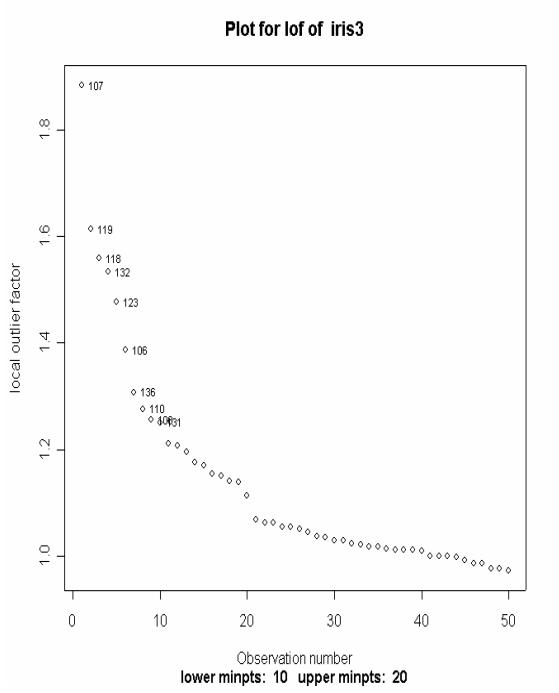

Figure 11. Instances of the class 3 in the *Iris* dataset ranked by the LOF's algorithm outlyingness measure

Outliers in class 1: (7)
16, 15 ,34, 42, 44, 24, 23

Outliers in class 2 (7)
71, 63 ,58 ,61, 94, 99, 69

Outliers in class 3 (5)
107, 119, 132, 118 ,120

In Table 4, the misclassification error of the three classifiers has been computed using three type of samples: a) the original sample, b) deleting the outliers from the original sample and, c) deleting a random subsample from the original sample.

Notice that all three classifiers are affected when outliers are removed, whereas there is only a minimum change on the misclassification error when a random subsample of instances is removed.

Table 4. The misclassification error rate for the LDA, knn and rpart classifiers in *Iris* using three different types of samples

	Original Simple	Deleting outliers	Deleting a random subsample
LDA	2.02	1.54	2.30
Knn(k=1)	4.05	2.35	4.10
Rpart	6.69	2.90	7.32

Table 5 shows the feature selected using the three type of samples described before. The feature selected methods used here are the sequential forward selection (SFS) with the three classifiers used in table 4 and the Relief

Table 5. Features selected in *Iris* using SFS and Relief for the three type of samples

	Original Sample	Deleting outliers	Deleting a random subsample
SFS(lda)	4,2	4,2	4,3
SFS(knn)	4,3	4,3	4,3
SFS(rpart)	4	4	4
Relief	2,3,4	4,3	4,3

There are few differences between the subset of features selected by the four methods.

Finally table 6 shows the misclassification error rates of the three classifiers after feature selection and for the three type of samples. Once again the performance of the three classifiers are affected by the deletion of outliers.

Let us consider now the *Bupa* dataset, which have 345 instances, 6 features and 2 classes. Using all the criteria described in section 3 we have detected the following outliers in each class of *Bupa*.

Outliers in class 1: (22)
168, 175, 182, 190, 205, 316, 317, 335, 345, 148, 183, 261, 311, 25, 167, 189, 312, 326, 343, 313, 20, 22

Table 6. Misclassification error rate after feature selection in *Iris* for the three type of samples

	Original Sample	Deleting outliers	Deleting a random subsample
LDA	3.70	2.33	5.31
knn(k=1)	4.01	1.87	4.80
Rpart	5.29	2.29	5.25

Outliers in class 2 (26)
36, 77, 85, 115, 134, 179, 233, 300, 323, 331, 342, 111, 139, 252, 294, 307, 123, 186, 286, 2, 133, 157, 187, 224, 278, 337

In Table 7 the misclassification error of three classifiers: LDA, knn and rpart had been computed based on the three type of samples considered before.

Table 7. The misclassification error rate for the LDA, knn and rpart classifiers in *Bupa* using three different type of samples

	Original Simple	Deleting outliers	Deleting a random subsample
LDA	31.82	26.23	31.17
Knn(k=7)	31.55	27.65	32.26
Rpart	31.86	33.24	35.07

Notice that the classifiers LDA and Knn are the most affected and the least affected has been the Rpart. The latter makes sense since it is well known that Rpart is a classifier that is robust to outliers.

Table 8 shows the feature selected using the three type of samples described before. The feature selected methods used here are the sequential forward selection (SFS) with the three classifiers used in table 4 and the Relief

Table 8. Features selected in *Bupa* using SFS and Relief for the three type of samples

	Original Sample	Deleting outliers	Deleting a random subsample
SFS(lda)	5,4,3,6	5,3,4	5,4,3,6
SFS(knn)	5,3,1	5,3,1,6	5,3,4,1
SFS(rpart)	5,3,6,2	5,3,2	5,3,2
Relief	6,3,4	4,2,5,3	2,4,3

There are differences between the subset of features selected by the four methods. Finally in table 9 the misclassification error rates of the three classifiers after feature selection and for the three type of samples.

Table 9. Misclassification error rate after feature selection in *Bupa* for the three types of samples

	Original Sample	Deleting outliers	Deleting a random subsample
LDA	34.94	26.72	35.62
knn(k=7)	36.53	30.65	40.99
Rpart	37.47	32.48	39.78

Notice that the lower misclassification errors are obtained for a sample where the feature selection is performed after deleting outliers.

An alternative to deleting outliers is treating them as missing values. Some people prefer the latter because it avoids the loss of sample size but other people do not like it too much because

it can create bias in the estimation. In this paper we have not experimented with this option.

6. Conclusions

In this paper we have used several methods for outlier detection. We can not recommend a unique method to detect outliers because some methods are efficient for detecting certain types of outliers but fail to detect others. On the other hand, our experimental results evidence the effect of the deletion of outliers on the performance of classifiers. The LDA and K-nn classifiers seem to be more affected by the presence of outliers than the Rpart classifier. On the other hand, the presence of outliers shows some effect on feature selection but this effect does not seem to be as evident as is in the estimation of the misclassification error rate.
In a future work we are planning to include more outlier detection methods and to consider larger datasets.

7. Acknowledgment

Edgar Acuna's work was supported by grant N00014-00-1-0360 from ONR and Caroline Rodriguez held a graduate fellowship through the grant EIA 99-77071 from NSF when this work was carried out.

8. References

1. Acuña, E., Coaquira, F. and Gonzalez, M. (2003). A comparison of feature selection procedures for classifiers based on kernel density estimation. Proc. of the Int. Conf. on Computer, Communication and Control technologies, CCCT'03. Vol I.p. 468-472. Orlando, Florida.

2. Atkinson, A. (1994). Fast very robust methods for the detection of multiple outliers. Journal of the American Statistical Association, 89:1329-1339.

3. Atkinson, A.C, Riani, M., and Cerioli, A. (2004). *Exploring Multivariate Data with th Forward Search*. Springer-Verlag. New York.

4. Barnett, V. and Lewis, T. (1994). *Outliers in Statistical Data.* John Wiley. New York.

5. Bay, S.D., and Schwabacher (2003). Mining distance-based outliers in near linear time with randomization and a simple pruning rule. Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.

6. Blake, C.L. and Mertz, C.J. (1998). UCI Repository of machine learning databases [http://www.ics.uci.edu/mlearn/MLRepository.html]. Irvine, CA: University of California, Department of Information and Computer Science.

7. Breuning, M., Kriegel, H., Ng, R.T, and Sander. J. (2000). LOF: Identifying density-based local outliers. In Proceedings of the ACM SIGMOD International Conference on Management of Data.

8. Fraley, C., and Raftery, A.E. (2002). Model-based clustering, discriminant analysis, and density estimation. Journal of the American Statistical Association, 97, 611-631.

9. Hadi, A. (1992). Identifying multiple outliers in multivariate data. Journal of the Royal Statistical Society B, 54:761-771.

10. Hardin, J. and Rocke, D. M. (2004). Outlier Detection in the Multiple Cluster Setting using the Minimum Covariance Determinant Estimator. *Computational Statistics and Data Analysis*, **44**,625–638.

11. Hawkins, D. (1980). *Identification of Outliers*. Chapman and Hall. London.

12. Kaufman, L. and Rousseeuw, P.J. (1990). *Finding Groups in Data: An Introduction to Cluster Analysis.* Wiley, New York.

13. Knorr, E. and Ng, R. (1997). A unified approach for mining outliers. In Proc. KDD, pp. 219–222.

14. Knorr, E., and Ng. R. (1998). Algorithms for mining distance-based outliers in large datasets. In Proc. 24th Int. Conf. Very Large Data Bases, VLDB, pages 392–403, 24–27.

15. Knorr., E., R. Ng, and V. Tucakov. (2000). Distance-based outliers: Algorithms and applications.VLDB Journal: Very Large Data Bases, 8(3–4):237–253.

16. Ng, R.T. and Han, J. (1994). Efficient and effective clustering methods for spatial data mining. Proc. 20th Int. Conf. on Very Large Data bases. Morgan and Kaufmann Publishers, San Francisco, 144-155.

17. Ramaswamy, S., Rastogi, R., and Shim, K. (2000). Efficient algorithms for mining outliers from large data sets. In Proceedings of the ACM SIGMOD International Conference on Management of Data.

18. Rocke, D. and Woodruff, D. (2002). Computational Connections Between Robust Multivariate Analysis and Clustering," in *COMPSTAT 2002 Proc. in Computational Statistics*, Wolfgang Härdle and Bernd Rönz eds., 255–260, Heidelberg: Physica-Verlag.

19. Rocke, D. and Woodruff, D. (1996). Identification of outliers in multivariate data. Journal of the American Statistical Association, 91:1047-1061.

20. Rodriguez, C.K. (2004). A computational environment for data preprocessing in supervised classification. Master Thesis University of Puerto Rico at Mayagüez.

21. Rousseeuw, P (1985). Multivariate estimation with high breakdown point. Mathematical statistics and applications.

22. Rousseeuw, P. and Leroy, A. (1987). *Robust Regression and Outlier Detection*. John Wiley.

23. Rousseeuw, P. and Van Zomeren, B. (1990). Unmasking multivariate outliers and leverage points. Journal of the American Statistical Association, 85:633-639.

24. Rousseeuw, P. J. & Van Driessen, K. (1999). A Fast Algorithm for the Minimum Covariance Determinant Estimator. Technometrics, 41, 212-223.

25. Tukey, J.W. (1977). *Exploratory Data Analysis.* Addison-Wesley.

26. Wegman, E. (1990). Hyperdimensional Data Analysis Using Parallel Coordinate Plot. Journal of the American Statistical Association, 85, 664-675.