Illustrative Emphasis of Outliers

Kedar Bartake
Computer Science Department
Stony Brook University
kbartake@cs.stonybrook.edu

Sakshi Gupta
Computer Science Department
Stony Brook University
sakgupta@cs.stonybrook.edu

Klaus Mueller
Professor
Computer Science Department
Stony Brook University
mueller@cs.stonybrook.edu

ABSTRACT

In today's data-driven world, it is an important step to isolate the abnormalities or 'outliers' in the data during the knowledge discovery step. Although there are many methods as to how to identify these outliers and compute their 'outlierness' as a metric, there are no proper methods to visualize outliers in a data-set so that they can be perceived easily by humans. This paper aims to present a 'caricature-like distortion' approach in which gradients of the density fields of the points will be used to perform a transformation which will highlight the outliers effectively.

1 INTRODUCTION

An outlier is defined as an observation point that is distant from other observations [1]. Mendenhall et al. [2] applies the term "outliers" to values "that lie very far from the middle of the distribution in either direction" [3]. Outliers could occur due to two basic reasons [4]:

- as an error while sampling, data entry, measurement etc
- not as errors but extreme and exceptional values

It is important to identify outliers in the data-set for various reasons such as improving the data quality, avoiding bias in simple statistical methods,prevent producing incorrect correlation coefficients in regression models and performing correct data analysis. It is due to these reasons that there are now numerous methods to spot outliers.

2 METHODS FOR OUTLIER DETECTION

2.1 Using Data Visualization

One of the most rudimentary steps is to visualize the data using box-plots, scatter-plots or histograms to get an idea of the distribution of the data. There are many libraries available in Python to perform this task such as matplotlib, scikit-learn and seaborn.

2.2 Using Statistical Methods

This approach involves using methods such as [1]:

- Numeric Outlier: uses the Inter Quartile Range (IQR) metric and finds outliers as being outside this range.
- Standard Deviation / Z-score: assumes the data has Gaussian distribution and identifies outliers as being very far from the mean i.e. near the tails.

2.3 Using Distance Based Methods

The basic idea here is that Normal data objects have a dense neighborhood and Outliers are far apart from their neighbors, i.e., have a less dense neighborhood.
Various clustering algorithms such as K-Means clustering or Hierarchical clustering can be used here but the mostly widely used is DBSCAN clustering algorithm. Here, all data points are defined either as Core Points, Border Points or Noise Points. Core Points are data points that have at least 'MinPts' neighboring data points within a distance 'Epsilon'. Border Points are neighbors of a Core Point within the distance 'Epsilon' but with less than 'MinPts' neighbors within the distance 'Epsilon'. All other data points are Noise Points, also identified as outliers. Outlier detection thus depends on the required number of neighbors MinPts, the distance 'Epsilon' and the selected distance measure, like Euclidean or Manhattan [5].

2.4 Using Isolation Forests

This method uses a different idea in the sense that it does not calculate normal regions and then identify outliers as points lying outside this region. Instead, what it does is assign scores to data points while exploiting the fact that outliers are minority data points having attribute values that are different from the regular data points [1].

2.5 Using Density Based Methods

The intuition here is to compare the density around a point with the density around its local neighbors. The relative density of a point compared to its neighbors is computed as an outlier score. This approach is based on the assumption that the density around a normal data object is similar to the density around its neighbors whereas the density around an outlier is considerably different to the density around its neighbors.
This method overcomes the problem of dealing with different densities which is something the distance based methods suffer from. In order to spot an outlier, an Local Outlier Factor (LOF) Score is computed [6]. The LOF of a point tells the density of this point compared to the density of its neighbors. If LOF = 1, the point belongs to a cluster and hence is not an outlier else if LOF $>>$ 1, the point is an outlier. There are various variants of LOF such as Connectivity-based outlier factor (COF), Influenced Outlierness (INFLO) and Local outlier correlation integral (LOCI).

3 IMPLEMENTATION

The data set sources for our experiment are UCI ML Repository and ELKI Toolkit. Our initial code is implemented using one of the Ames Housing Data sets on Kaggle. We have focused on Density Based methods for detecting the outliers. The initial step was to pre-process the data in order to normalize it and using graphs such as scatter-plots to visualize our data. For simplicity, we picked out only 2 features for our initial experiments.
The idea here is that we compute a displacement vector for each data point and the magnitude of the vector can either turn out to be positive (towards the density peak) - mostly in case of inliers, or negative (away from the density peak) in case of the outliers. This would in turn mean that inliers will be pushed towards the density peaks to create a highly dense data point cluster and the outliers will be pushed far away from the cluster which will make them more visible.
This method has the following steps after the tasks mentioned above:

1. Calculate the LOF Score for each data point using the scikit-learn python library. Data points having LOF $>>$ 1.5 are marked with orange in order to visualize them as outliers.

2. Calculate the density field around each data point using Kernel Density Estimation (KDE), also available in scikit-learn.

3. Compute a threshold value, which is the top 10 percent of the KDE density points

4. Calculate all data points having density value greater than the threshold value as consider them as density peaks for all the other data points. The density peaks are marked with yellow color.

5. Train a Nearest Neighbour model in order to assign each data point to a density peak to which it will belong to.

6. Calculate a translation vector for the point which will have a direction towards the nearest density peak assigned to it and a magnitude which will be calculated based on KDE density of the point, LOF score of the point and Distance of the point from the assigned density peak.

7. Perform the transformation for each data point based on each vector.

We used two mathematical formulae to calculate the scaling factor for the translation vectors:

$$Formula1 : scale = A - B * 10 - C$$

where A = KDE value, B = LOF Score and C = Distance from the nearest density peak.

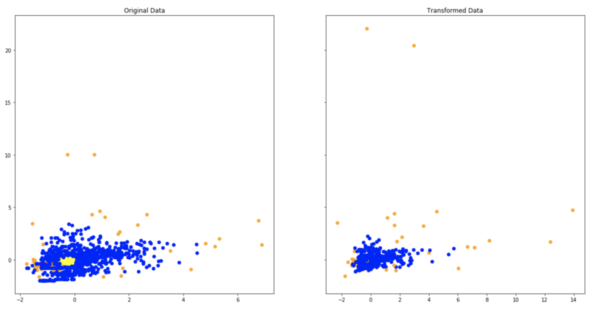

Figure 1: Transformation using Formula 1

$$Formula2 : scale = ((A - B) * 10 * C])/D$$

where A = LOF Score, B = Subtraction Factor, C = Distance from the nearest density peak and D = KDE value. The subtraction factor here can be experimented with in order to adjust the amount by which the data points are moved.

Our main challenge here was to come up with an evaluation metric to test our algorithm, find the optimal subtraction factor for Formula 2 and compare transformations with and without inward inlier movement.

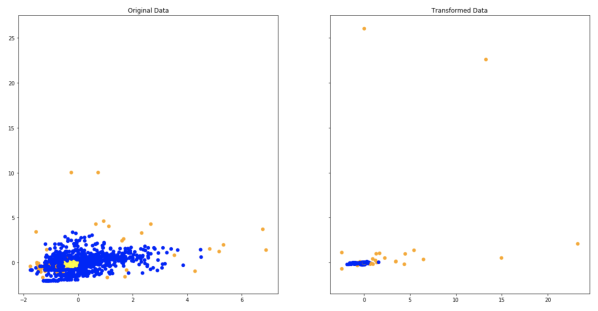

Figure 2: Transformation using Formula 2

3.1 Modifications

One of the problems with this approach is that we take into account just the first nearest neighbor due to which we run into a problem when there are multiple clusters in data. So an outlier, based on its first nearest neighbor, could get pushed away from one cluster but towards another cluster, distorting the original data.
One solution that we came up with was before choosing the points with highest density as our density peaks, we worked with clustering algorithms that do not take the number of clusters as its input. Once we get the clusters, we further iterate over all the clusters and find the density peaks within those clusters (local density fields).
First, we tried out DBSCAN clustering algorithm, but it did not turn out to be successful as the number of resultant clusters is dependent on the value of epsilon that we pass to the model which is not what we want. So, next we tried to find the optimal number of clusters using the K-elbow method and then we performed K-Means clustering in order to get the cluster labels, thus eliminating the need to manually choose the number of clusters.
The rest of the algorithm remains the same and we proceed by performing step 2 to step 4 to get a list of density peaks for the entire data set and then perform the rest of the steps as mentioned before.
Code: Github Link (Click Here)

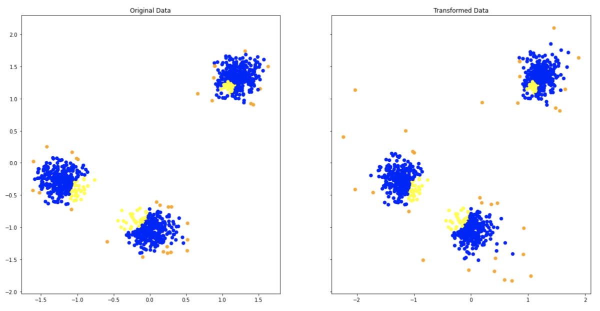

Figure 3: Transformation after modification

4 RESULTS

4.1 User Study

We designed a dashboard using JavaScript D3 Library which displays a scatter-plot of the transformed data and provides a slider for the users. There is also a drop down which gives an option to choose from amongst multiple data-sets to understand how different data are transformed differently. Using the slider, the users can select from amongst different values of the subtraction factor (B in the formula) from a range of 0 to 1.75. Based on the value chosen, the scaling factor (Formula 2mentioned in Section III) gets changed as the value of the subtraction factor changes and the outliers are pushed more outwards. We have also provided two radio buttons to provide a comparison between transformation in which inliers are shifted inwards and one where inliers are not shifted inwards. The users were asked to experiment with different data-sets provided and decide a particular value of the subtraction factor at which the outliers look prominently pushed out.

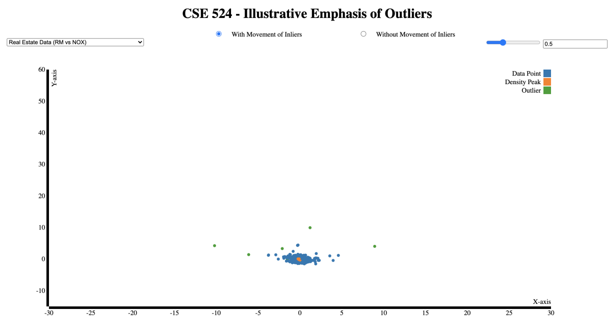
CSE 524 - Illustrative Emphasis of Outliers
Real Estate Data (RM vs NOX)
● With Movement of Inliers
○ Without Movement of Inliers
0.5
Y-axis
X-axis
Data Point 🟦
Density Peak 🟧
Outlier 🟩
60
50
40
30
20
10
0
-10
-30
-25
-20
-15
-10
-5
0
5
10
15
20
25
30

Figure 4: Dashboard for User Study

4.2 Evaluation Metric

In order to measure the effectiveness of our datapoint manipulation scheme, we need to numerically quantify the displacement that our scheme produces on the outliers. Our aim is to visually emphasize the outliers in the scatterplot. Therefore, visually, the outliers should have more 'outlierness' than they had before. Our assumption is that since, in the ideal case, all the outliers have more 'outlierness' than before, their Local Outlier Factor (LOF) score should increase. Therefore, we define our evaluation metric as the mean of the sum of differences of LOF score of previously identified outlier points before and after our scheme translates those points.

If O1, O2, O3,...,On are n outlier points identified before the transformation is applied, and LOF(O1), LOF(O2), LOF(O3),....,LOF(On) are the LOF scores before transformation and LOFT(O1), LOFT(O2), LOFT(O3),....,LOFT(On) are the LOF scores after transformation, then we calculate the metric as follows:

$$E = ((LOF_T(O_1) - LOF(O_1)) + ... + (LOF_T(O_n) - LOF(O_n)))/n$$

We propose to use this metric to measure the performance of our outlier transformation method.

4.3 User Study Results

We asked a small group of 10 people to use our dashboard and tell us a value from the slider at which they are best able to distinguish the outliers from the inliers. Here are the statistics:

Table 1: Table showing votes of users on Subtraction Factor

Subtraction Factor	Votes	Percentage
0	0	0
0.25	0	0
0.5	0	0
0.75	0	0
1	1	10
1.25	3	30
1.5	5	50
1.75	1	10

From this data, we concluded that 1.5 seems to be the optimal subtraction factor based on the dashboard.

4.4 Experimental Results

The main goal of performing these experiments was to decide on an optimal subtraction factor and choose which transformations work best visually.

4.4.1 Experiment 1

In this experiment, we compared the score values between transformation with Inlier shift and without Inlier shift. The resultant data is shown in the table. From this we could deduce that for majority of the cases the score values are high for transformation without any inlier shift.

Table 2: Data Table for Experiment 1

Dataset	Column X	Column Y	Score (Inlier Shift)	Score (No Inlier Shift)
Housing	SalePrice	LotFrontage	0.441	1.565
Housing	SalePrice	LotArea	2.795	3.34
Housing	TotalBsmtSF	LotArea	5.047	5.084
UCI air quality	T	C6H6_GT	1.439	1.882
UCI air quality	T	RH	0.627	2.994
UCI air quality	RH	AH	0.244	0.998
Real Estate Data	CRIM	NOX	0.927	1.963
Real Estate Data	RM	NOX	6.544	1.958
Real Estate Data	INDUS	DIS	0.878	0.941
Cereal Nutrition	Calories	Sodium	6.4	1.847
Cereal Nutrition	Calories	Fiber	2.327	2.327
Cereal Nutrition	Potass	Sodium	7.75	1.012
Forest Fires	FFMC	DMC	0.712	1.344
Forest Fires	RH	DC	0.438	4.08
Forest Fires	wind	temp	2.26	0.283

4.4.2 Experiment 2

In this experiment we wanted to understand which subtraction factor would work best for the scaling formula and hence would push out the outliers the most. Along with this, as in Experiment 1, we also compared the score values of the two different transformation techniques.

Table 3: Data Table for Experiment 2

Subtraction Factor	Score (Inlier Shift)	Score (No Inlier Shift)
0	0.42	1.437
0.25	0.393	1.484
0.5	0.38	1.49
0.75	0.348	1.488
1	0.304	1.57
1.25	0.395	1.581
1.5	0.604	1.599
1.75	0.38	1.468

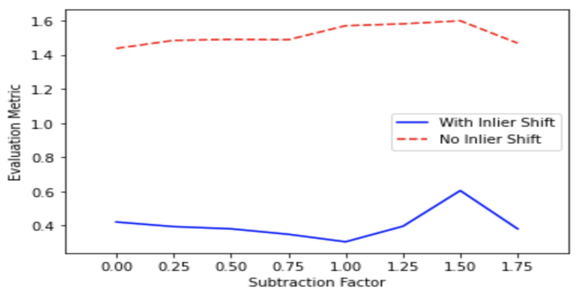

Figure 5: Comparison of Evaluation Metric Based on Inlier Shift

From both the table and the line graph, we deduced that the value 1.5 works as the optimal subtraction factor and our deduction of experiment 1 is again confirmed here i.e. transformation without inlier shift has greater score values.

5 CONCLUSION

Availability of data is at it's boom in today's time and it is highly important to understand the data one is working with. Knowing and understanding the presence of outliers is an important step in the process of Data Analysis. Our paper proposes an algorithm which produces a caricature-like transformation on the original data which helps in emphasizing the presence of outliers in the data-set. In addition, we have devised an evaluation metric to compare different methods of producing this transformation and have also developed a simple dashboard for user study purposes. The experimental data helped us to find an optimal value for our transformation and compare various transformations and decide the best one amongst them.

6 FUTURE SCOPE

There is still some scope left to further improve our objective to identify outliers. Here are some of our ideas that we worked on:

- We could further explore the concept of Local Gradient Field in order to provide better differentiation between data points and outliers. But this approach had two major problems. One of them was that the grid size for sampling the density on the scatter-plot as it is difficult to find a value that is feasible for all types of data. Another problem was choosing an optimal bandwidth for the Kernel Density Function (Later on we chose to average the densities over a range of bandwidth inputs).

- In order to solve the grid size problem, Adaptive Kernel Density Estimation (AKDE) could be used which doesn't need the grid size to be fixed and can be used for all types of data. This size can be used to fit a Gaussian and compute the gradients and move the outlier points according to the gradients.

- This method could therefore work as another method to identify outliers and we could use our evaluation metric to compare and make further conclusions.

ACKNOWLEDGMENTS

We wish to thank our advisor, Professor Klaus Mueller for his active guidance and expertise through the project.

REFERENCES

[1] https://towardsdatascience.com/ways-to-detect-and-remove-the-outliers-404d16608dba

[2] Mendenhall, W., Reinmuth, J.E., and Beaver, R.J. (1993). Statistics for Management and Economics. Belmont, CA: Duxbury Press.

[3] https://www.ise.bgu.ac.il/faculty/mlast/papers/outliers2.pdf

[4] https://www.theanalysisfactor.com/outliers-and-their-origins/

[5] https://www.kdnuggets.com/2018/12/four-techniques-outlier-detection.html

[6] https://archive.siam.org/meetings/sdm10/tutorial3.pdf

[7] M. Young, The Technical Writer's Handbook. Mill Valley, CA: University Science, 1989.