

Project 2 Final Presentation


Hung Yee Wong a1815836

Topic Introduction (Data Science)


What is Data Science ?

According to IBM , Data science is a multidisciplinary approach to extracting actionable insights from the large and ever-increasing volumes of data collected and created by today's organizations.

Why is Data Science important and relevant ?



175 Zettabytes By 2025



Tom Coughlin Contributor ©
Enterprise Tech

Follow

How large Is a Zettabyte ?

| WHAT'S A ZETTABYTE? | |
|---------------------|---|
| 1 kilobyte | 1,000,000,000,000,000,000 |
| 1 megabyte | 1,000,000,000,000,000,000,000 |
| 1 gigabyte | 1,000,000,000,000,000,000,000,000 |
| 1 terabyte | 1,000,000,000,000,000,000,000,000,000 |
| 1 petabyte | 1,000,000,000,000,000,000,000,000,000,000 |
| 1 exabyte | 1,000,000,000,000,000,000,000,000,000,000,000 |
| 1 zettabyte | 1,000,000,000,000,000,000,000,000,000,000,000,000 |

Most, if not all of these data can be analysed by using Data Science methods to produce interesting trends and insights !

These insights help businesses answer the important questions such as, would this product sell ?

The Challenge (Outlier Detection)

What is an outlier ?

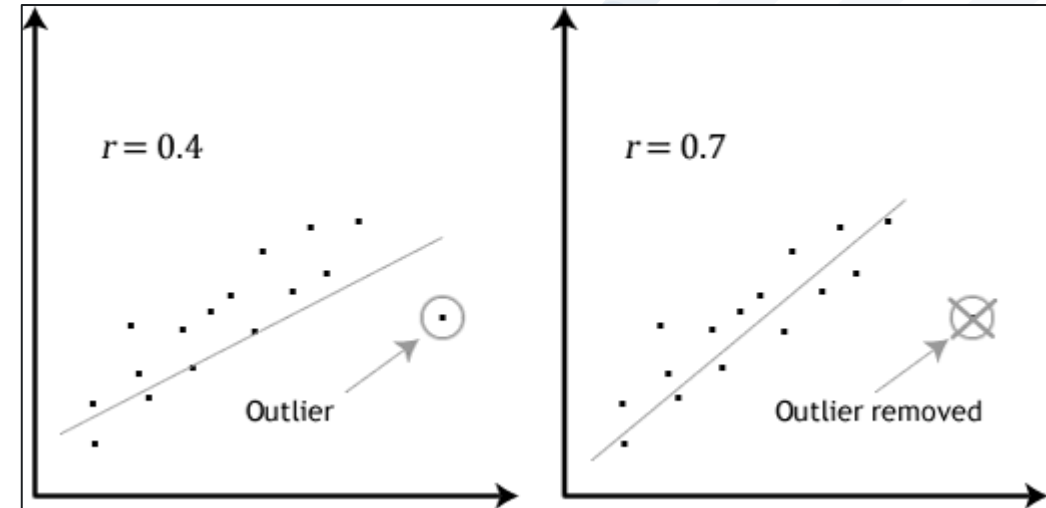
An outlier is an unusually large or small observation.

Why are outliers a bad thing ?

Outliers can have disproportionate effects on statistical results, such as the mean, which can cause misleading interpretations that may bring losses to businesses and individuals

What is the challenge ?

To devise an accurate outlier detection method



Here's an example of how an outlier affects the interpretation of data

Thus, we aim to accurately remove the outliers within our data

Multi-Rule Outlier (MIRO) Algorithm

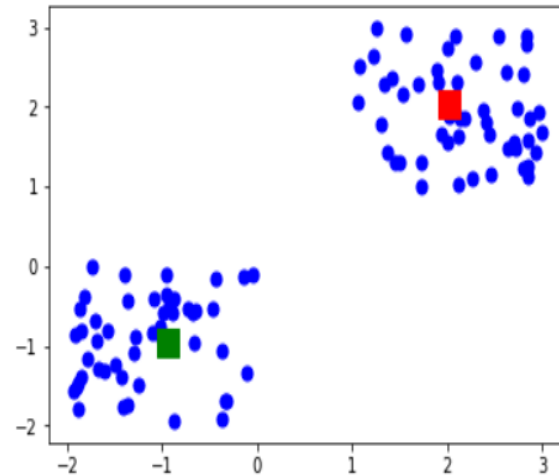
The first solution is the MIRO algorithm which is a pruning distance based algorithm [1]

Pruning in this context means removal

3.1 Cluster Based Pruning

In this phase, we first cluster the dataset DS (using Algorithm 1) and subsequently identify upper and lower bounds of the outlier score for each resultant cluster (using Algorithm 2). Algorithm 1 is in fact based on the clustering algo-

Here is the output:



Example output of K-Means clustering, green and red points are the optimized centroids

First, clustering is done by a K-means algorithm

For the predetermined number of clusters, M , and the number of points per cluster n_c , the algorithm will run and reiterate to optimize positions of centroids

MIRO Continued

Then, for each point in the resulting clusters, the bounds of the outlier score of the point is calculated by taking the sum of distances between the point and the k-nearest data points.

Then, the bounds for each cluster, are defined as the max upper and the minimum lower bound of any point within the cluster

Clusters are then ordered in descending order of lower bound value and are picked until n data points are reached, we let the lower bound value of the last cluster be C.

Any cluster that has a upper bound lower than C is removed (pruned) and not considered.

Finally, all the remaining points are checked. If the outlier score of a point p is smaller than C or if the inequality $C > kD(p, q) + outlierscore(q)$ returns true for p, where q is a point that is being compared with p, then p is not an outlier.

Let C be the set of clusters obtained as a result of applying Algorithm 1 on DS with predetermined values of M and it . For each cluster $C_i \in C$, let $|C_i|$ denote its cardinality (or the number of data points allocated to C_i), o_{C_i} its centroid, and r_{C_i} its radius. l_{C_i} , u_{C_i} are the estimated lower and upper bounds of the outlier scores of all data points in C_i respectively. These bounds are only estimations since the true bounds can only be known when the true scores of member data points are identified. A data point p by itself is also a cluster C_i with $o_{C_i} = p$, $r_{C_i} = 0$, $l_{C_i} = u_{C_i} = F_{out}(p)$.

Definition 2. [DISTANCE BETWEEN CLUSTERS]

The minimum distance between clusters C_i and C_j is

$$minDis(C_i, C_j) = \max\{D(o_{C_i}, o_{C_j}) - r_{C_i} - r_{C_j}, 0\},$$

and maximum distance between clusters C_i and C_j is

$$maxDis(C_i, C_j) = D(o_{C_i}, o_{C_j}) + r_{C_i} + r_{C_j}.$$

Consider a data point $p \in C_i$. To compute the lower bound of its outlier score, we have to find the *closest* clusters to p in terms of $minDis()$. In order to do this we consider all clusters closest to C_i as well as other data points in C_i (as clusters). So we choose $Min_p = Min_{C_i} \cup C_i \setminus p$. In order to estimate the cumulative distance from p to its k nearest neighbors, we order Min_p and choose the top z clusters $M_1 \dots M_z$ s.t. $\sum_{i=1}^{z-1} |M_i| < k \leq \sum_{i=1}^z |M_i|$. Now the lower bound of the outlier score of p can be computed as $l_p = \sum_{i=1}^{z-1} |M_i| \cdot minDis(p, M_i) + (k - \sum_{i=1}^{z-1} |M_i|) \cdot minDis(p, M_z)$.

Similarly we can compute the upper bound of p 's outlier score, $u_p = \sum_{i=1}^{z-1} |M_i| \cdot maxDis(p, M_i) + (k - \sum_{i=1}^{z-1} |M_i|) \cdot maxDis(p, M_z)$, where $\{M_1 \dots M_z\}$ are the top z clusters in Max_p defined as $Max_{C_i} \cup C_i \setminus p$.

Definition 3. [BOUNDS OF A CLUSTER'S OUTLIER SCORE]. The upper and lower bounds of a cluster's outlier score in terms of its contained points are given as: $u_{C_i} = \max\{u_p, p \in C_i\}$ and $l_{C_i} = \min\{l_p, p \in C_i\}$, respectively.

Drawbacks of MIRO

First, we notice that there are a lot of user defined parameters in this algorithm, therefore we can conclude that a good amount of knowledge of the dataset is required to produce accurate results.

From a survey paper, they state that :

It is not extended to large
and high-dimensional data
sets.

Thus, this solution also faces the curse of dimensionality

Recap of Local Outlier Factor (LOF)

Before reviewing the next paper, let's do a quick recap of the LOF algorithm [2] as the next solution uses elements from this solution.

First, we define the k -distance(0) :

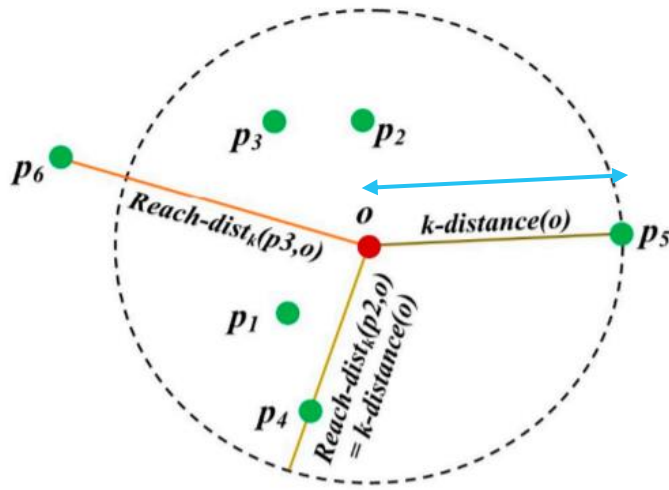


Figure 4. The reachability distance for different data points p with regard to o , when k equals 5.

The k -distance(0) is the maximum distance between the point 0 itself and any of the k -closest points to 0, where k is a user defined parameter

The reachability distance of 0:

Definition 5: (reachability distance of an object p w.r.t. object o)

Let k be a natural number. The *reachability distance* of object p with respect to object o is defined as

$$\text{reach-dist}_k(p, o) = \max \{ k\text{-distance}(o), d(p, o) \}.$$

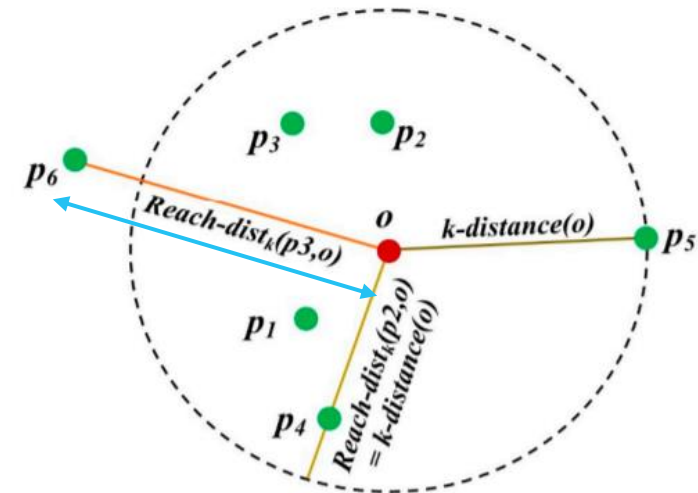


Figure 4. The reachability distance for different data points p with regard to o , when k equals 5.

If P is within the neighbourhood of O , the reachability distance is the k -distance(O), else the reachability distance is the distance between P and O itself

LOF Continued

- Then using this formula, the reachability density of the point, p , is calculated :

O is one of the K -nearest neighbours of P

Definition 6: (local reachability density of an object p)

The local reachability density of p is defined as

$$lrd_{MinPts}(p) = \frac{1}{\frac{\sum_{o \in N_{MinPts}(p)} reach-dist_{MinPts}(p, o)}{|N_{MinPts}(p)|}}$$

Reach-dist of P w.r.t O

Number of K -nearest neighbours

- Finally, the local outlier factor, LOF for the point is calculated :

Definition 7: ((local) outlier factor of an object p)

The (local) outlier factor of p is defined as

$$LOF_{MinPts}(p) = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|}$$

- Where a threshold is set by user to filter outliers, which is generally close to 1, if the LOF of a point is larger than the threshold, it is considered as outlier

General drawbacks :

Fails to deal with the multi-granularity problem and is sensitive to the choice of the $MinPts$.

Requires the computation for all objects in the dataset, which is expensive and might miss possible outliers whose local neighborhood density is very close to that of its neighbors.

- Sensitive to the parameter k ($MinPts$)
 - User defined parameter
- Computation is expensive
 - Average runtime of $O(n^2)$

Robust Kernel Outlier Factor (RKOF)

RKOF employs a Kernel Density Estimation (KDE) approach [3]. A kernel can be said to be a function which takes in parameters and returns a probability value.

Similar to the LOF method, the K-distance of a point p , $K\text{-distance}(p)$ is defined as:

Definition 1. Given a data set D , an object p , and any positive integer k , the $k\text{-distance}(p)$ is defined as the distance $d(p, o)$ between p and an object $o \in D$, such that:

- for at least k objects $o' \in D \setminus \{p\}$, it holds that $d(p, o') \leq d(p, o)$.
- for at most $k - 1$ objects $o' \in D \setminus \{p\}$, it holds that $d(p, o') < d(p, o)$.

Definition 2. Given a data set D , an object p , and any positive integer k , the $k\text{-distance}$ neighborhood of p , named $N_k(p)$, contains every object whose distance from p is not greater than the $k\text{-distance}(p)$, i.e., $N_k(p) = \{q \in D \setminus \{p\} | d(p, q) \leq k\text{-distance}(p)\}$, where any such object q is called a $k\text{-distance}$ neighbor of p . $|N_k(p)|$ is the number of the $k\text{-distance}$ neighbors of p .

The method first calculates the local kernel density estimate of a point:

Definition 3. (Local kernel density estimate of object p)
The local kernel density estimate of p is defined as

$$kde(p) = \frac{\sum_{o \in N_k(p)} \{h^{-\gamma} \lambda_o^{-\gamma} K(h^{-1} \lambda_o^{-1}(p - o))\}}{|N_k(p)|}$$

$$\lambda_o = \{f(o)/g\}^{-\alpha} \quad \log g = |D|^{-1} \sum_{q \in D} \log(f(q))$$

where h is the smoothing parameter, γ is the sensitivity parameter, $K(x)$ is the multivariate kernel function and λ_o is the local bandwidth factor. $f(x)$ is a pilot density estimate that satisfies $f(x) > 0$ for all the objects, α is the sensitivity parameter that satisfies $0 \leq \alpha \leq 1$, and g is the geometric mean of $f(x)$.

In this paper, we compute the pilot density function $f(x)$ by the approximate nearest neighbor density estimate according to Equation 1.

$$f(o) = \frac{1}{k\text{-distance}(o)} \quad (2)$$

Gamma is given a default value 2, and alpha is given the value 1

RKOF Continued

Then, the estimation of the neighbourhood density which contains the point is calculated.

Definition 4. (Weighted density estimate of object p 's neighborhood)
The weighted density estimate of p 's neighborhood is defined as

$$wde(p) = \frac{\sum_{o \in N_k(p)} \omega_o \cdot kde(o)}{\sum_{o \in N_k(p)} \omega_o} \quad \omega_o = \exp \left\{ - \frac{\left(\frac{k\text{-distance}(o)}{\min_k} - 1 \right)^2}{2\sigma^2} \right\}$$

where ω_o is the weight of object o in the k -distance neighborhood of object p , σ is the variance with the default value 1, and $\min_k = \min_{o \in N_k(p)} (k\text{-distance}(o))$.

Definition 5. (Robust kernel-based outlier factor of object p) *The robust kernel-based outlier factor of p is defined as*

$$RKOF(p) = \frac{wde(p)}{kde(p)}$$

where $wde(p)$ is the density estimate of the k -distance neighborhood of p , and $kde(p)$ is the local density estimate of p .

With the values of the local density estimate and neighbourhood density estimate, the RKOF is calculated

With the values of the local density estimate and neighbourhood density estimate, the RKOF is calculated. The larger this value is, the higher probability that p is an outlier. This translates to the neighbourhood which p is in being very dense but p 's surroundings are not dense, thus it makes sense that p is isolated from others

Drawbacks of RKOF

As we can tell from the explanations and formula shown, when we compare this method to the last method, this method will have to traverse through the whole dataset, increasing computational cost.

Another drawback is that it only carries out comparison to 3 other methods, it is difficult to determine the true improvement brought by the method :

A survey also notes that this method is very high in complexity :

The complexity of the method.

| <div>Methods \ Data</div> | KDD | Mammography | Ann-thyroid | Shuttle (average) |
|---------------------------|------------------|----------------|----------------|-------------------|
| RKOF ^a | 0.962 (1918.1s) | 0.871 (15.8s) | 0.970 (4.9s) | 0.990 (36.4s) |
| RKOF ^b | 0.961 (2095.2s) | 0.870 (19.8s) | 0.970 (5.2s) | 0.990 (36.9s) |
| RKOF ^c | 0.944 (2363.7s) | 0.855 (48.2s) | 0.965 (13.2s) | 0.993 (36.7s) |
| LOF | 0.610 (2160.1s) | 0.640 (28.8s) | 0.869 (5.9s) | 0.852 (42.0s) |
| LDF | 0.941 (2214.9s) | 0.824 (36.4s) | 0.943 (7.2s) | 0.962 (37.1s) |
| LPF | 0.98 (>>2363.7s) | 0.87 (>>48.2s) | 0.97 (>>13.2s) | 0.992 (>>42.0s) |
| Bagging | 0.61(±0.25) | 0.74(±0.07) | 0.98(±0.01) | 0.985(±0.031) |
| Boosting | 0.51(±0.004) | 0.56(±0.02) | 0.64 | 0.784(±0.13) |
| Feature Bagging | 0.74(±0.1) | 0.80(±0.1) | 0.869 | 0.839 |
| Active Learning | 0.94(±0.04) | 0.81(±0.03) | 0.97(±0.01) | 0.999(±0.0006) |

a. Using Volcano kernel b. Using Gaussian kernel c. Using Epanechnikov kernel

Solution Motivation

My solution aims to solve some of the challenges faced by the current solutions, especially density based algorithms, we take a look at what a survey says :

b: DISADVANTAGES, CHALLENGES, AND GAPS

Even though some density-based methods are shown to have improved performance, they are more complicated and **① computationally expensive** when compared especially to statistical methods in most cases [96]. They are **sensitive to parameter settings** such as in **determining the size of the neighbors**. They need to cautiously take into consideration several factors, which consequently results in expensive computations. For varying density regions, it becomes more complicating and results in poor performance. Density-based methods, due to their inherent complexity and the lack of update of their outlierness measures, some of these algorithms, such as INFLO and MDEF, cannot resourcefully handle data streams. In addition, they can be a poor choice for outlier detection in data stream scenarios. **It is also challenging for high dimensional data when the outlier scores are closely related to each other.** **②**

③

In the first cut, we discussed the problems faced by current solutions

- Computationally expensive
- Sensitive to parameter settings
- Curse of dimensionality

And what can we do to solve them

- Reduce the number of iterations needed
- Change the type of distance metric
- Carry out dimension reduction
- Cluster the data

Proposed solution

We propose a solution which is a combination of multiple techniques which may reduce the drawbacks of past solutions :

We know high dimensionality causes problems, first we use a method called SC-MDS (Split and Combine Multidimensional Scaling) [4], which is a fast MDS algorithm with $O(n)$ complexity to reduce dimensions

Now with the reduced dimension data, we need to carry out clustering using a clustering algorithm such as the k-means method

The selection of the parameter k for k-means can be done using an existing algorithm [5]

After obtaining the clustered data, we run a density based method locally on each cluster, thus the outlier scoring only occurs within each cluster

As we know, density methods also require the k parameter for each cluster, this will be done by applying the k value algorithm from NELOF [6]. Time complexity is $O(n)$

Lastly, a density based algorithm such as RDOS[7] or RKOF[3] is applied to each cluster

Conclusion: Our new method reduces the computational complexity from $O(N^3)$ to $O(N)$ when the dimension of the feature space is far less than the number of genes N , and it successfully reconstructs the low dimensional representation as does the classical MDS. Its performance depends on the grouping method and the minimal number of the intersection points between groups. Feasible methods for grouping methods are suggested; each group must contain both neighboring and far apart data points. Our method can represent high dimensional large data set in a low dimensional space not only efficiently but also effectively.

Time complexity of SC-MDS

Algorithm 5 Nearest Neighborhood Variance-Balancing Algorithm

Input: data point to be detected: y , threshold: $thres_k$, set of nearest neighborhoods: $N = \{area_1, area_2, \dots, area_n\}$
Output: K

BEGIN

$K = 2$

$n = \text{length}(N)$

for $i = 1$ to n do

{

for each point p in $area_i$ do

{

$euDist(p, y) = \text{EuclideanDistance}(p, y)$

$euDist_ordered(p, y) = \text{sort}(euDist(p, y))$

}

}

for $K = 2$ to $thres_k$ do

{

for each $area_i$ in N do

{

$k_nearest < k, eu_dist \geq$ the first K values of $euDist_ordered$

$k_variance < k, variance \geq$ variance of the $k_nearest$

}

}

for each $area_i$ in N do{

$Ki[] = K$ corresponding to the local minimum of $k_variance[K]$

$K =$ first equivalence element of $\{K1[], K2[], \dots, Kn[]\}$

}

}

END

Based on the time complexity analysis of algorithm 1, we can state the following theorem.

Theorem 1: Time complexity of the proposed ensemble KNN classifier can be approximated to linear function $O(n)$.

NELOF k-parameter estimation algorithm

References

- [1] N. H. Vu and V. Gopalkrishnan, "Efficient pruning schemes for distancebased outlier detection," in Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases, 2009, pp. 160–175
- [2] M. Breunig, H. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," ACM SIGMOD Rec., vol. 29, no. 2, pp. 93–104, 2000.
- [3] J. Gao, W. Hu, Z. Zhang, X. Zhang, and O. Wu, "RKOF: Robust kernel based local outlier detection," in Proc. Pacific-Asia Conf. Knowl. Discovery Data Mining, 2011, pp. 270–283.
- [4] Tzeng, J., Lu, H.HS. & Li, WH. Multidimensional scaling for large genomic data sets. BMC Bioinformatics 9, 179 (2008).
<https://doi.org/10.1186/1471-2105-9-179>
- [5] Pham, D. & Dimov, Stefan & Nguyen, Cuong. (2005). Selection of K in K -means clustering. Proceedings of The Institution of Mechanical Engineers Part C-journal of Mechanical Engineering Science - PROC INST MECH ENG C-J MECH E. 219. 103-119. 10.1243/095440605X8298.
- [6] P. Yang, D. Wang, Z. Wei, X. Du and T. Li, "An Outlier Detection Approach Based on Improved Self-Organizing Feature Map Clustering Algorithm," in IEEE Access, vol. 7, pp. 115914-115925, 2019, doi: 10.1109/ACCESS.2019.2922004.
- [7] Jin Ning, Leiting Chen, and Junwei Chen. 2018. Relative Density-Based Outlier Detection Algorithm. In Proceedings of the 2018 2nd International Conference on Computer Science and Artificial Intelligence (CSAI '18). Association for Computing Machinery, New York, NY, USA, 227–231. DOI:<https://doi.org/10.1145/3297156.3297236>