DETECTION OF CLUSTERED OUTLIERS IN MEDICAL DATA

Matthew Greffin, Theresa Lucena, & Cayte Lyonnais
Slippery Rock University
mig1003@sru.edu, txl1025@sru.edu, & cdl1009@sru.edu

ABSTRACT
Many studies exist pertaining to medical data, but the detection of outliers in medical data through machine learning is a new field of research. As such, there is no documentation currently available on the accuracy of diagnosis using outlier detection methods. Determining which method is most effective in detection of outliers of medical diagnoses would allow for an increase in patient survival rates. In the current study, the accuracy of four outlier detection algorithms (Isolation Forest, Split-Selection Forest Criteria, Local Outlier Factor, and Cluster Based Local Outlier Factor) is compared using the Area Under the Receiving Operating Characteristics Curve. Findings show that the predictive capacity of the Isolation Forest and Split-Selection Forest Criteria models was 100%, or equivalent once model predictions were inverted; both Local Outlier Factor variants were found to be less effective. This demonstrates that the Isolation Forest and Split-Selection Forest Criteria algorithms are capable of effectively detecting outliers within medical data. We speculate that this approach might be applied to sets of patient chart data sorted by diagnosis code, for the purpose of comparing estimated outlier rate to estimated rates of misdiagnosis by diagnosis code. Limitations related to technological constraints and further study recommendations are discussed.

KEY WORDS
Medical Outliers; Isolation Forest (iForest); Split-Selection Forest Criteria (SCiForest); Local Outlier Factor (LOF); Cluster Based Outlier Factor (CBLOF); Area Under Receiver Operating Characteristics Curve (AUC-ROC Curve)

1. Introduction

Physicians strive to provide their patients with the best possible medical care. This desire to help those in need has pushed the scientific community to continually research better treatments and medications. However, this focus on the advancement of technology and pharmacology, while necessary, ignores the more basic -- yet equally important -- issue of patient information management. For over a century, medical science has provided us with innovative technologies to identify illnesses and insights into the pathogens behind disease, but to effectively defend against maladies, it is important to correctly diagnose them. Patient outcomes depend upon doctors accurately diagnosing ailments and providing the indicated treatment, making the accuracy of this process critical.

Rates of diagnostic accuracy might be identified by applying outlier detection methods to patient chart data. Outliers are data objects which differ significantly in some respect from comparable data objects [1]. Identifying these outliers within a dataset of medical information may allow for retrospective detection of possible diagnostic errors, which may lead to improvements in accuracy for diagnostic processes. Current estimates of the overall rate of misdiagnosis in clinical practice range from 11% [2] to as high as 38% [3]. Such diagnostic errors result in significant disability and mortality among patients [2]. These estimates suggest that many diagnostic errors are related to a small subset of nosological entities. We suppose this might suggest clusters of outliers, and note that such clustering leads to increased difficulty in detection [4], [5].

Although diagnostic accuracy is vital to patient survival, there are no extant publications documenting the accuracy of diagnoses using outlier detection. Furthermore, using outlier detection in medical data is a new development within this field, as seen in the work of Samariya and colleagues, published as recently as April of 2023 [6]. Samariya et al. compared the Isolation Forest and Local Outlier Factor algorithms. Performance was measured by the Area Under the Receiver Operating Characteristics Curve (AUC-ROC Curve) and determined that the Isolation Forest algorithm was the favorable method.

With this paper, we intend to expand upon this previous work by comparing these outlier detection algorithms to variants thereof devised specifically to address the phenomenon of clustered outliers. We will use the AUC-ROC Curve measure to determine which of the following algorithms functions most effectively in identifying outliers: Isolation Forest (iForest), Split-Selection Criteria Forest (SciForest), Local Outlier Factor (LOF), or Cluster-Based Local Outlier Factor (CBLOF).

2. Methods

To measure the effectiveness of the four algorithms, their performance will be compared through the application of two sets of medical data available from the Stony Brook University Outlier Detection Datasets (ODDS) repository: Lymphography (Lympho) and Thyroid Disease (Annthyroid). Each dataset contains known outliers and has been used in previous studies involving outlier detection.

2.1 Datasets

Lympho: We used the Lympho dataset [7] found in the ODDS Repository. This data was obtained from the University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. The set consists of 148 observations for each of 19 variables describing the presence and condition of tumors in patients' lymph nodes. The classification variable takes four possible values, of which there are 81, 61, 4, and 2 instances in the dataset. The latter two classes are taken as outliers, for a total of six outlier instances.

Annthyroid: We used the Annthyroid dataset [8] found in the ODDS Repository. This data was obtained from Garavan Institute in Sydney, Australia. The set consists of 7,200 observations for each of 21 attributes (15 binary, 6 continuous) among 3 classes which were meant to classify thyroid abnormalities in patients. The 534 instances which classify as either hypo- or hyper-functional are taken as outliers.

2.2 Outlier Detection Algorithms

The Local Outlier Factor and Cluster-Based Outlier Factor methods were computed using Python 3.12.0 [9]. The Isolation Forest and Split-Selection Criteria Forest were computed using the IsoTree package in R Statistical Software (v4.1.2) [10].

Isolation Forest [11], [12], [13]: Isolation Forest (iForest) is an outlier detection procedure introduced by Liu and colleagues in 2008 [11]. In contrast to earlier outlier detection methods, which first build a profile of ordinary instances within a dataset and then flag those instances which deviate from that profile, the Isolation Forest starts by isolating the anomalous instances from the rest of the dataset. The procedure is divided into two stages: Training, in which a forest of Isolation Trees (iTrees) is constructed; and Evaluation, in which the iTrees are compared to determine which instances are anomalous.

The Training stage consists of building an iForest, a set of iTrees. An iTree is built by sampling ψ instances from the dataset, and then repeatedly partitioning the dataset. At each partitioning step, an attribute of the dataset is randomly selected, and a split-point is randomly selected from the selected attribute's range. The dataset is then split into two subsets; the left subset, containing all instances for which the value of the selected attribute is less than the split point, and the right subset, containing all instances for which the value of the selected attribute is greater than or equal to the selected attribute. This partitioning step is applied repeatedly, until either all sampled instances are isolated – partitioned into a subset containing no other instances – or a preset tree height limit (by default, $log_2(\psi)$) [11] is reached.

In the Evaluation stage, the path length $h(x)$ – the number of partitions required to isolate an instance – is calculated for every instance x, using the partitions defined for each iTree in the forest. For those iTrees in which the instance reaches a terminal node without becoming isolated, the remainder of the path length is estimated with (1) [11], where the argument n is the number of instances contained in the terminal node, and $H(i)$ is the harmonic number function, estimated by $ln(i) + 0.5772156649$.

$$c(n) = 2H(n - 1) – (2(n – 1)/n) \quad (1)$$

The expected path length $E(h(x))$ for each instance x is then determined by calculating the average path length $h(x)$ across every iTree in the forest. The instance x is then given an anomaly score, according to (2) [11], where argument x is the instance being scored, and argument n is the total count of instances in the dataset being evaluated. The anomaly score is interpreted such that instances with scores very near 1 are very likely to be anomalous and instances with scores significantly less than 0.5 are very likely to not be anomalous [11]. Should it turn out that every instance in the dataset has an anomaly score near 0.5, then there are very likely no anomalous instances in the dataset [11].

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}} \quad (2)$$

Split-Selection Criteria Forest [4]: The Split-Selection Criteria Forest (SCiForest) is a variant of the iForest method introduced by Liu and colleagues in 2010 [4]. It has been altered to better detect clusters of anomalies, outlier data points which are very close to one another – hence, difficult to isolate – yet far from the majority of points in the dataset, and hence still outliers. This is accomplished by accounting for multiple attributes during each partitioning step, while optimizing both the split-point and the attributes factored for anomalousness. In a manner of speaking, the algorithm looks at each data point from many angles and chooses the angle from which it appears strangest. Following is a detailed explanation.

SCiForest represents two major changes from iForest. The first is that, rather than randomly selecting a single attribute at each partitioning step, q such attributes are randomly selected, from which several hyperplanes f of the dataset are constructed according to (3) [4], wherein x is an instance in the dataset X', Q is the set of q randomly selected attributes, $\sigma()$ is the usual standard deviation function, c_j is a coefficient randomly selected from [-1, 1],

p is the optimal split-point of the hyperplane f for maximizing the hyperplane's gain in standard deviation, calculated per (4), and j subscripts indicate reference to the *jth* attribute of the dataset. The selection of an optimal split point p, rather than a random selection of split point, is the second major change in SCiForest. Additionally, the hyperplane itself is also optimized for gain in standard deviation, by generating τ hyperplanes at each partitioning stage and selecting the best.

$$f(x) = \sum_{j \in Q} c_j \frac{x_j}{\sigma(X_j')} - p \quad (3)$$

The aforementioned gain in standard deviation is defined in (4), where $Y = f(x)$ is a hyperplane of the dataset, l and r superscripts are used to denote the left and right partitions of Y, $\sigma()$ is the usual standard deviation function, and $avg()$ is the usual arithmetic mean function.

$$sd_{gain}(Y) = \frac{\sigma(Y) - avg(\sigma(Y^l), \sigma(Y^r))}{\sigma(Y)} \quad (4)$$

Local Outlier Factor: The Local Outlier Factor (LOF) algorithm is a data mining and machine learning technique designed for outlier detection. It was first introduced by Breunig and colleagues in 2000. It assesses the local density deviation of data points with respect to their neighbors to identify anomalies.

LOF calculates a score for each data point where a higher score indicates a higher likelihood of being an outlier. First, the density of data points is measured. LOF calculates the density with respect to its k-nearest neighbor. k-distance, the distance from a data point (p) and its k-nearest neighbor o, is calculated per (5) [13].

$$d(p, o) = \sqrt{\sum_{t=1}^{n} (p_t - o_t)^2} \quad (5)$$

In this context, the k-Nearest Neighbors (kNN) of data point p encompass any data point q within a distance not exceeding the k-distance from p. These k-Nearest Neighbors of p collectively compose the k-distance neighborhood of p [14] as outlined in (6).

$$N_{k-distance(p)}(p) = \{q \in D/\{p\} | d(p, q) \leq k - distance(p)\} \quad (6)$$

Next, LOF will consider reachability distance. This says the reachability of point p with respect to its neighbor o is the maximum of the distance between p and o and the local density of o as seen in (7) [14].

$$reach - dist_k(p, o) = \max\{k - distance(o), d(p, o)\} \quad (7)$$

When the average reachability distance is greater (indicating distant neighbors from the point), there's a lower density of points around that specific point, referred to as the Local Reachability Distance (LRD). This metric provides insight into the distance between p point and its nearest cluster of points. Lower LRD values suggest that the closest cluster is distant from the point itself [14].

$$Lrd_{MinPts}(p) = \frac{|MinPts(p)|}{\sum_{o \in N_{MinPts(p)}} reach-dist_{MinPts}(p,o)} \quad (8)$$

Cluster-Based Local Outlier Factor [15]: The Cluster-Based Local Outlier Factor (CBLOF) algorithm is similar to the Local Outlier Factor algorithm. It is designed to enhance outlier detection in datasets where there are clusters of information. CBLOF combines the ideas of LOF with the idea of cluster-based mining. By doing this the algorithm can provide a better way of picking out outliers from the dense and sparse regions.

CBLOF first will create clusters in the dataset by using k-means. Then it estimates the density of each cluster. Then it will use the reachability distance for each cluster. The algorithm determines outliers by figuring out the distance between the center of the cluster to each data point. CBLOF can decipher local outliers within each cluster and global outliers which are outliers that deviate from the entire dataset. Points that are outliers outside of a small dataset may have their outlier factor score increased, where points outside a large cluster may have their score decreased. Cluster-Based Outlier Factor is beneficial to use when dealing with large datasets that may have many different size clusters. When comparing it to LOF, LOF may misclassify points within dense clusters whereas CBLOF considers clusters and their structures.

2.3 AUC-ROC Curve Measure

Each of these outlier detection algorithms is considered a classification method. Within machine learning, the performance of classification methods is commonly analyzed using the Area Under the Curve for Receiver Operating Characteristics (AUC-ROC) measure.

In order to discuss the AUC-ROC Curve, it is important to first understand the confusion matrix. The confusion matrix is created as a result of four possible outcomes from a binary prediction.

1) True negative (TN): where the algorithm correctly predicts a negative class (0) as a negative class (0).
2) False negative (FN): where the algorithm incorrectly predicts a positive class (1) as a negative class (0).
3) True positive (TP): where the algorithm correctly predicts a positive class (1) as a positive class (1).
4) False positive (FP): where the algorithm incorrectly predicts negative class (0) as a positive class (1).

These outcomes are used to identify the true positive rate (TPR) and false positive rate (FPR). The true positive rate is the proportion of positive data points correctly identified as positive in relation to the total number of data points. When the TPR is higher, fewer positive data points are misidentified. This value is calculated as follows in (10).

$$TPR = TP/(TP+FN) \quad (10)$$

The false positive rate is the proportion of negative data points that are misidentified as positive in relation to the total number of data points. When the FPR is higher, the

more negative data points are misclassified. This value is calculated as follows in (11).

$$FPR=FP/(FP+TN) \quad (11)$$

The values for the true positive rate and false positive rate are computed across various thresholds and plotted on a single graph in which the false positive rate delineates the horizontal axis, and the true positive rate marks the vertical axis. The resulting curve is the Receiver Operating Curve (ROC). Once the curve is graphed, the area below the curve is shaded. This shaded area beneath the ROC constitutes the AUC-ROC measure [16].

The AUC-ROC is bounded between zero and one (inclusive) and is used to determine how well an algorithm (model) distinguishes between classes. The closer the AUC value is to 1, the better the model is at correctly predicting classes. Note that an AUC =1 marks a perfect classifier. Alternatively, if the AUC value is close to 0, the model consistently identifies positive classes as negative classes and vice versa. An AUC value close to zero remains useful as the decisions are consistently switched, allowing for simply inverting the output of the algorithm in order to obtain a good model. The only undesirable outcome is when AUC = 0.5 as this would mean the model has no capacity to distinguish between positive and negative classes.

3. Results and Discussion

3.1 Lympho Results

Our results obtained by analyzing the Lympho dataset with the four selected outlier detection methods are reported in Table I and Figs. 1 through 4. Our findings indicate that the Isolation Forest and Split-Selection Criteria Forest algorithms reliably classified outliers. The "perfect" AUC scores should be understood as the result of these models converging on a single (accurate) threshold value. We also find that the Local Outlier Factor method over-estimates the number of outliers. Curiously, the Cluster-Based Local Outlier Factor method would seem to have a tremendous false positive rate, judging by its AUC score, yet it predicts the correct quantity of outliers.

3.2 Annthyroid Results

Our results obtained by analyzing the Annthyroid dataset with the four selected outlier detection methods are reported in Table II and Figs. 5 through 8. Our findings again indicate that the Isolation Forest and Split-Selection Criteria Forest algorithms reliably classified outliers, though again, we note that the "perfect" AUC scores should be understood as the result of these models converging on a single (accurate) threshold value. We find that once again the Local Outlier Factor method overestimates outlier count. We again observe the Local Outlier Factor method resulting in a significantly greater AUC score than the Cluster-Based Local Outlier Factor, which again appears to a problem of false positives, per the AUC score. Despite this, we note that the CBLOF method significantly underpredicted outlier count for this dataset.

3.3 Tables and Graphs

Table 1
Lympho Results

Detection Method	**Results**			
	Outliers Count	***Inliers Count***	***Outliers %***	***AUC Score***
LOF	15	133	10.14%	0.9871
CBLOF	6	142	4.05%	0.0319
Iso Forest	6	142	4.05%	1.0000
SCiForest	6	142	4.05%	1.0000

Figure 1
ROC Curve for Local Outlier Factor, Lympho Dataset

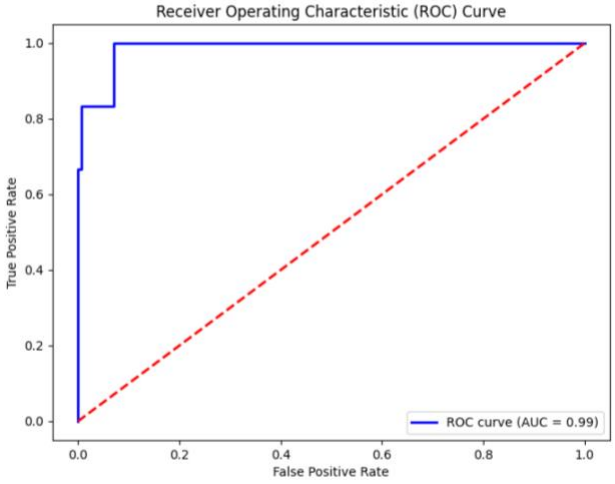


Figure 2
ROC Curve for Cluster-Based Local Outlier Factor, Lympho Dataset

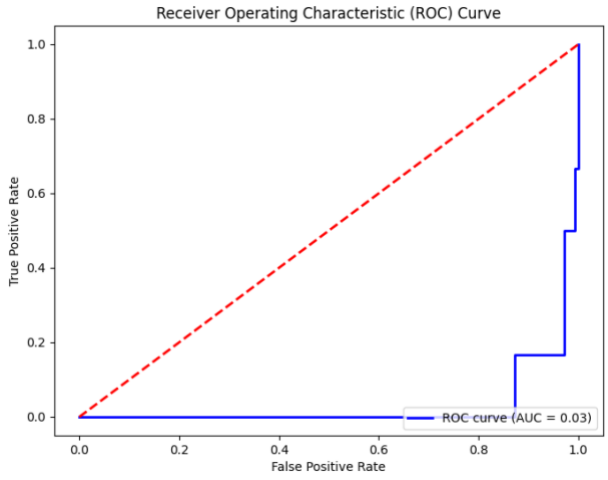


Figure 3
ROC Curve for Isolation Forest, Lympho Dataset

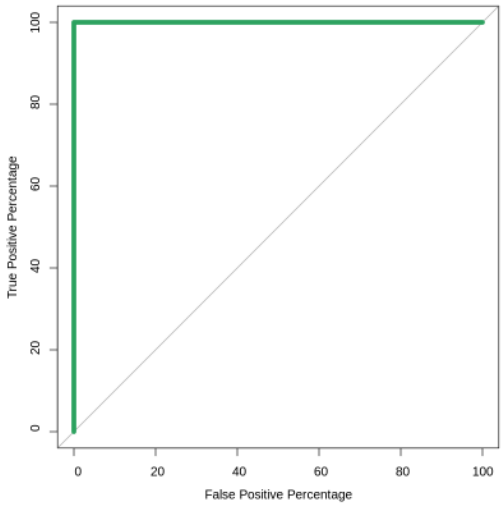


Figure 4
ROC Curve for Split-Selection Criteria Forest, Lympho Dataset

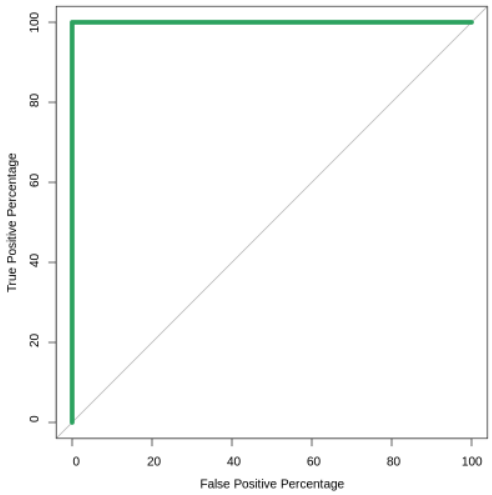


Table 2
Annthyroid Results

Detection Method	**Results**			
	Outliers Count	***Inliers Count***	***Outliers %***	***AUC Score***
LOF	720	6480	10.00%	0.7373
CBLOF	359	6841	4.99%	0.3540
Iso Forest	534	6666	7.42%	1.0000
SCiForest	534	6666	7.42%	1.0000

Figure 5
ROC Curve for Local Outlier Factor, Annthyroid Dataset

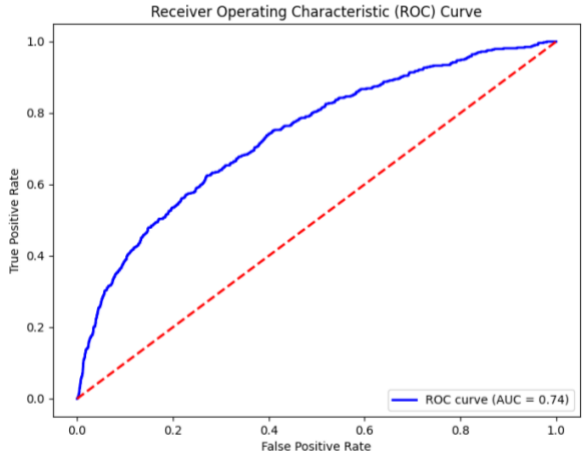


Figure 6
ROC Curve for Cluster-Based Local Outlier Factor, Annthyroid Dataset

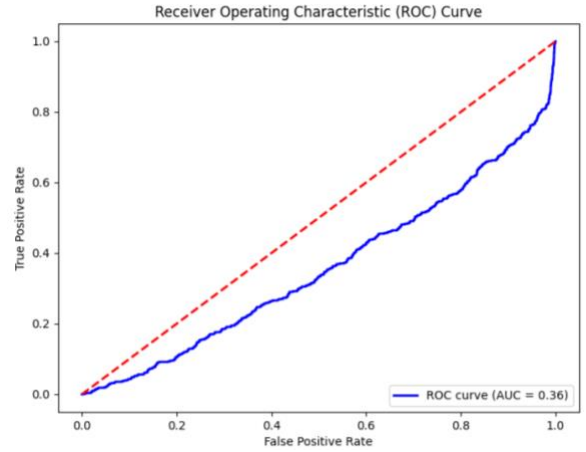


Figure 7
ROC Curve for Isolation Forest, Annthyroid Dataset

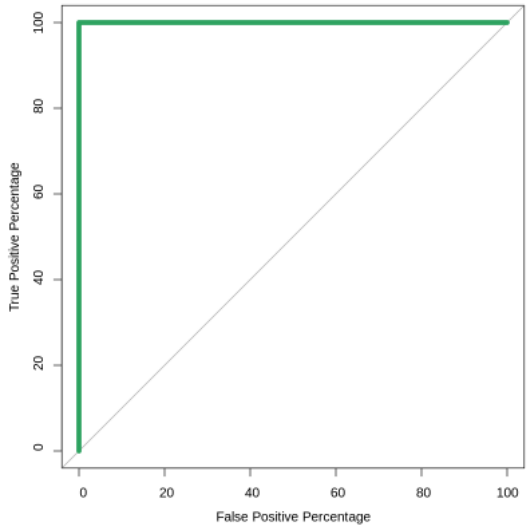
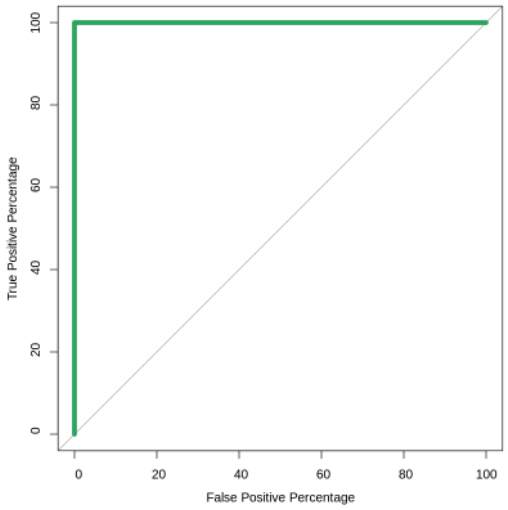


Figure 8
ROC Curve for Split-Selection Criteria Forest, Annthyroid Dataset



3. Conclusion

We find that the Isolation Forest method, as well as its variant, the Split-Selection Criteria Forest method, reliably detect anomalous instances across two sets of medical data. We find that the Local Outlier Factor and Cluster-Based Local Outlier Factor methods perform less admirably. Having confirmed the efficacy of the isolation tree-based methods, we anticipate a future study employing these same methods to analyze large sets of patient chart data, such as the MIMIC databases at Massachusetts Institute of Technology's Laboratory for Computational Physiology, to test the theory that when sorted into subsets by diagnosis code, those subsets will have outliers at rates proportional to previously-published per-condition rates of misdiagnosis.

References:

[1] R. Suri, M Narasimha Murty, G Athithan, and Springerlink, Outlier Detection: Techniques and Applications: A Data Mining Perspective. Cham: Springer International Publishing, 2019.

[2] D. E. Newman-Toker et al., "Rate of diagnostic errors and serious misdiagnosis-related harms for major vascular events, infections, and cancers: Toward a national incidence estimate using the 'Big Three,'" Diagnosis, vol. 8, no. 1, pp. 67–84, 2020. doi:10.1515/dx-2019-0104.

[3] K. Sadegh-Zadeh, "The logic of diagnosis," Philosophy of Medicine, pp. 357–424, 2011. doi:10.1016/b978-0-444-51787-6.50012-x.

[4] F. T. Liu, K. M. Ting, and Z.H. Zhou, "On detecting clustered anomalies using SCiForest," Machine Learning and Knowledge Discovery in Databases, pp. 274–290, 2010. doi:10.1007/978-3-642-15883-4_18.

[5] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," Pattern Recognition Letters, vol. 24, no. 9–10, pp. 1641–1650, 2003. doi:10.1016/s0167-8655(03)00003-5.

[6] D. Samariya, J. Ma, S. Aryal, and X. Zhao, "Detection and explanation of anomalies in healthcare data," Health Information Science and Systems, vol. 11, no. 1, 2023. doi:10.1007/s13755-023-00221-2.

[7] M. Zwitter, M. Soklic, "Lymphography," UCI Machine Learning Repository. doi:10.24432/C54598, Stony Brook Outlier Detection Datasets Repository, https://odds.cs.stonybrook.edu/lympho/.

[8] R. Quinlan, "Thyroid Disease," UCI Machine Learning Repository. doi: 10.24432/C5D010, Stony Brook Outlier Detection Datasets Repository, https://odds.cs.stonybrook.edu/thyroid-disease-dataset/.

[9] "The Python Tutorial — Python 3.7.4 documentation," Python.org, 2019. https://docs.python.org/3/tutorial/index.html.

[10] R: A language and environment for statistical computing, R Core Team (2021). R Foundation for Statistical Computing, Vienna, Austria. https://www.R-project.org/.

[11] F. T. Liu, K. M. Ting, and Z.H. Zhou, "Isolation Forest," IEEE International Conference on Data Mining 2008, doi:10.1109/ICDM.2008.17.

[12] Y. Chabchoub, M.U. Togbe, A. Boly, and R. Chiky, "An in-depth study and improvement of Isolation Forest," IEEE Access, pp.10219 - 10237, 2022. doi: 10.1109/ACCESS.2022.3144425.

[13] D. Cortes, "Revisiting randomized choices in isolation forests," Dec. 2021. doi:2110.13402.

[14] O. Alghushairy, R. Alsini, T. Soule, and X. Ma, "A Review of Local Outlier Factor Algorithms for Outlier Detection in Big Data Streams," Big Data and Cognitive Computing, vol. 5, no. 1, p. 1, Dec. 2020, doi: 10.3390/bdcc5010001.

[15] Z. He, X. Xu, and S. Deng, "Discovering cluster-based local outliers," Pattern Recognition Letters, vol. 24, no. 9–10, pp. 1641–1650, Jun. 2003, doi: https://doi.org/10.1016/s0167-8655(03)00003-5.

[16] S. Narkhede, "Understanding AUC - ROC Curve," Medium, Jun. 26, 2018. https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5.