Hypothyroidism: An Unsupervised Approach to Anomaly Detection

Ahmed ElAlfy
Matr. Num. 910411
MSc AI4ST
a.elalfy@campus.unimib.it

Hero Rfaat Mohammed
Matr. Num. 908424
MSc AI4ST
h.mohammed@campus.unimib.it

Mohammed Faidi
Matr. Num. 910044
MSc AI4ST
m.faidi@campus.unimib.it

***Abstract*— How can anomalies be detected without any prior knowledge of the data? How challenging it would be to deal with mixed data types? Which steps should be taken before detecting anomalies from such data? In this report, we investigate the possibility of detecting outliers from such datasets using different outlier detection methods and techniques. We will compare their results and conclude the most effective method.**

1. Introduction

Today anomalies in datasets produced by the current tools can happen to faults, glitches, or cyber-attacks which require anomaly detection techniques [1]. This activity to the task of searching for patterns in data that do not fit into the usual behavior of the data [2]. Anomalies can be found in different domains and fields such as in the healthcare field, anomalies present in the medical records, or the manufacturing and industrial fields, quality issues or anomalies must be detected before certain defects appear, or in the environment field where detecting anomalies is important for weather prediction, and the list goes on. Research on identifying anomalies or outliers has been extensively studied in the statistics society since the 19th century. As time goes by, various anomaly detection techniques have been created and a considerable lot of these techniques have been particularly created to suit certain application areas while others are more to nonspecific domains [3]. Anomaly detection methods can be divided into three types:

- **Supervised**: It requires pre-labeled data classified as abnormal and normal to detect anomalies [4]
- **Semi-Supervised**: This category of technique assumes that the training data has labeled instances of the normal class while labels for the anomaly are not required.
- **Unsupervised (our task)**: This is our focus to detect anomalies in the dataset without prior knowledge.

2. Dataset Description

The dataset consists of 21 attributes. It contains 15 binary and 6 continuous attributes with a total of 7200 objects. No further information was given about the significance of those attributes or how they contribute in the case of Hypothyroidism.

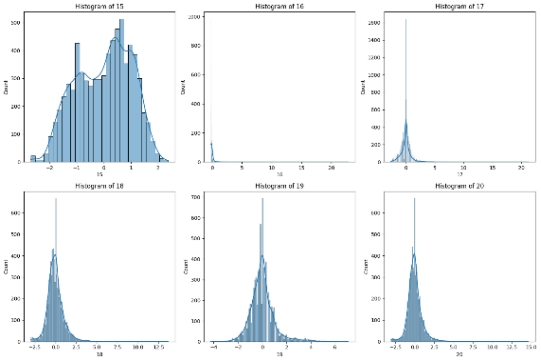

Figure 1. Grid of histograms for continuous attributes

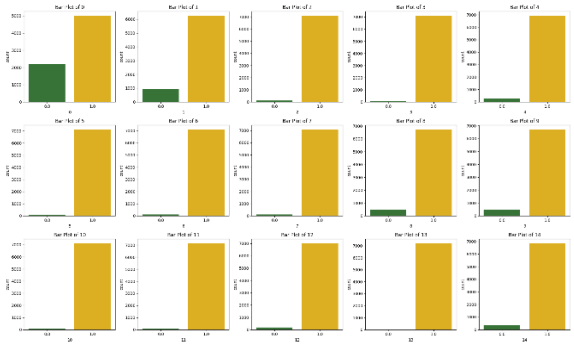

Figure 2. Grid of bar plots for binary attributes

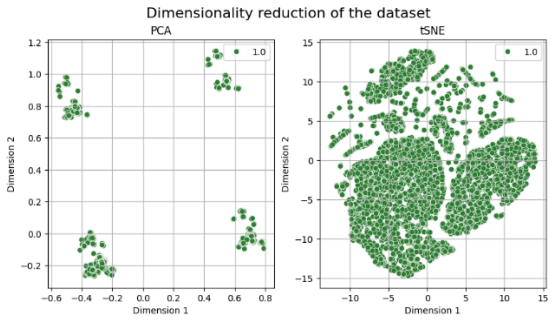

Figure 3. PCA and tSNE visualization for the whole dataset

3. Pre-Processing

Preprocessing the data is the first step towards successful anomaly detection. First, we had to clean our data by removing the headers in the CSV file and splitting the data by semi-column. Also, we replaced the comma with a dot in the continuous values to be processed as proper floats. One of the most important steps to do in the preprocessing of data is normalization as it is very crucial in adjusting the values of columns in a dataset without distorting differences in the ranges of values or losing information. In our project, we proceeded with the standard normalization or the Z-score standardization which is setting the data to have a mean = 0, and a standard deviation = 1 [5]. This normalization technique was applied to the continuous attributes so that the data became in the range of the binary values (0,1).

$$Z = \frac{x-\mu}{\sigma}$$

where:

- Z is the standardized value,
- x is the original value,
- μ is the mean of the original data,
- σ is the standard deviation of the original data.

4. Proximity Measure

The Proximity measure plays a vital role in the field of anomaly detection as the more suitable it is, the more effective the outlier detection algorithm becomes. Many types of proximity matrices can be calculated; however, it depends on which type of attributes we have. We have used two types of proximity distances: the Gower Distance and the Mahalanobis Distance. The Mahalanobis was used directly as an outlier detection method so we will discuss it later in the paper. Our choice to use the Gower Distance came from its ability to deal with different kinds of attributes. The resulting Gower distance values lie between 0 and 1, where 0 indicates identical objects and 1 indicates maximal dissimilarity. For two observations x and y with n attributes, the Gower distance $d(x, y)$ is calculated as follows:

- Numerical:

$$d_i(x,y) = \frac{|x_i - y_i|}{R_i}$$

- Categorical:

$$d_i(x,y) = \{0 \; if \; x_i = y_i \; 1 \; if \; x_i \neq y_i$$

- Binary:

  Similar to categorical but can handle symmetric and asymmetric binary attributes.

- Ordinal:

$$d_i(x,y) = \frac{|rank(x_i) - rank(y_i)|}{R_i}$$

5. Anomaly Detection

This is the main stage in this project in which we try to detect anomalies. Anomalies in a dataset can be detected in various ways and algorithms whether it is a proximity-based approach such as the Nearest Neighbor approach (NN), Local Outlier Factor (LOF), or Connectivity Outlier Factor (COF), which is either distance or density based. There are also Cluster, Statistical, and Reconstruction-Based Approaches such as Likelihood Approach, DBscan, and Principal Component Analysis (PCA). Given the nature of our data which is a mixed dataset of binary and continuous attributes, it was crucial to go through various algorithms and techniques that deal with this kind of data as many algorithms fail to do so. Our work was divided into applying different algorithms to the binary attributes alone, applying them to the continuous ones alone, and finally applying them to the whole data. Algorithms were used included:

- **Binary Attributes**:
  - Mahalanobis distance
  - Isolation forest

- **Continuous Attributes**:
  - Principle Component Analysis (PCA)
  - Isolation Forest
  - Local Outlier Factor (LOF)
- **Mixed Attributes**:
  - Hierarchical Clustering
  - Isolation Forest
  - Variational Autoencoder (VAE)

5.1 Binary Attributes

5.1.1 Mahalanobis Distance

Mahalanobis distance is the measure of the distance between a point and its distribution [6]. It differs from Euclidean distance by taking into account the correlations of the dataset. One of its advantages also is that not being affected by the scale of the measurements because it normalizes the differences using the covariance matrix. Being able to evaluate how each observation relates to the multivariate distribution formed by all variables together, made this approach perfect for capturing outliers in the binary data as it identifies points that are unusual considering the joint distribution, rather than just individual variables. The steps to calculate the Mahalanobis Distance are as follows:

1. Compute the Mean Vector
2. Compute the Covariance Matrix
3. Compute the Inverse of the Covariance Matrix

Compute the distance using the formula below:

$$d(x, \mu) = \sqrt{(x - \mu)^T \Sigma^{-1} (x - \mu)}$$

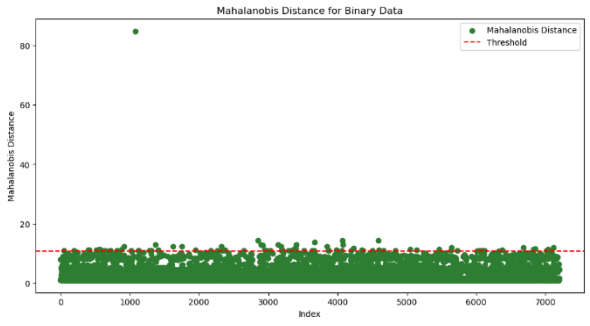

Figure 4. Scatterplot of Mahalanobis Distance for binary data

Total number of outliers detected is 90.

5.1.2 Isolation Forest

Isolation Forest or iForest builds an ensemble of trees for a given data set, then anomalies are those instances that have short average path lengths on the trees [7]. This algorithm uses a tree structure in which each of the tree nodes represents a decision based on randomly selected features and a value is split between the minimum and maximum values of these selected features. Once the forest of trees is built, each data point is passed through the trees, and the average path length across all trees is computed. The anomaly score is derived from the path length, with shorter path lengths indicating higher anomaly scores. The anomaly score for a data point x is given by:

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

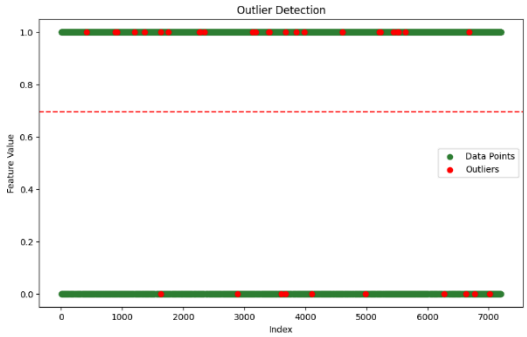

Figure 5. Scatterplot of Isolation Forest for binary data

Although the algorithm was used only on binary data, it wasn't effective in detecting outliers which were 36. After detecting the anomalies using the Mahalanobis Distance and Isolation Forest, we took the intersection set between them to be more accurate about the chosen anomalies which were in total 14 outliers in the binary attributes.

5.2 Continuous Attributes

5.2.1. Principal Component Analysis (PCA)

PCA is a well-known statistical technique that is widely used in dimensionality reduction. It transforms data into a lower-dimensional space by converting the data into a new set of variables called principal components. Before proceeding with applying the PCA to any dataset, it must be

standardized first by having a mean equal to 0 and a standard deviation equal to 1. The covariance matrix is calculated to understand the relationship between the variables, which will then be used to find the eigenvectors and eigenvalues to find the principal components. Finally, the original data is projected onto the new dimensional space that is defined by the principal components. This algorithm is very effective in outlier detection as the principal components capture the most variance, which makes it easy to detect outliers by filtering the noise. PCA workflow is as follows:

- **Transform the data into a lower dimensional space.**
- **Compute the reconstruction error:** which is the error between the original data points and their reconstruction. In our case, we used the Mean-Squared Error (MSE) as our reconstruction error.
- **Setting the threshold:** points with reconstruction errors higher than the threshold are considered outliers.

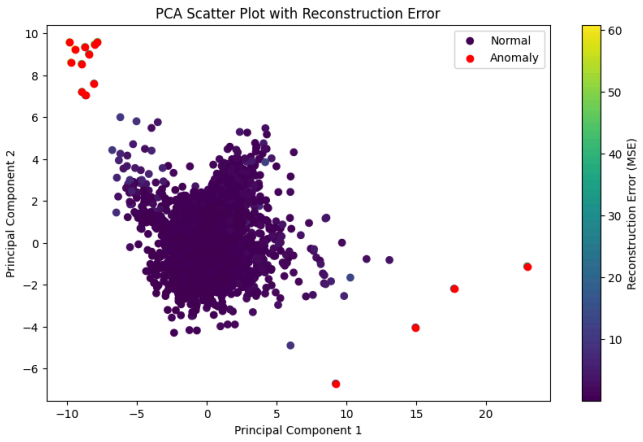

Figure 6. Outlier Detection using PCA

The total number of outliers detected was 15.

5.2.2. Isolation Forest

As we stated before Isolation Forest is very effective in outlier detection, so we applied it again separately on the continuous data.

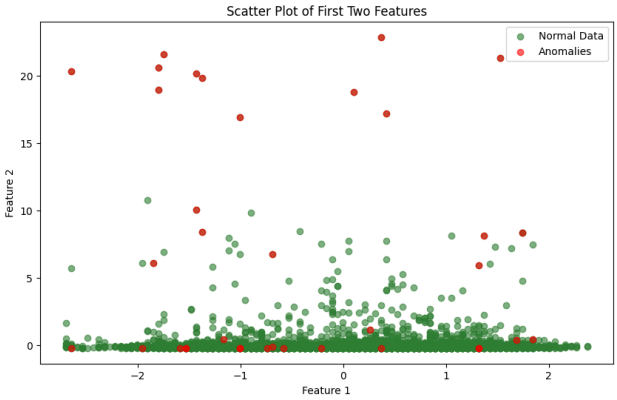

Figure 7. Scatterplot of outlier detection on continuous data using Isolation Forest

As seen from the visualization the Isolation Forest failed in detecting many of the outliers. The total number of outliers detected was 36.

5.2.3. Local Outlier Factor (LOF)

Local Outlier Factor is one of the density-based outlier detections that overcomes some drawbacks of the other distance approaches. LOF measures the density of a point and compares this density to the densities of its neighbors which makes this algorithm able to detect the outliers in a density diverse dataset. LOF algorithm goes as follows:

K-Neighbors: calculate the distance of the k-nearest neighbor for each point

Reachability Density: For each point p and each of its k-neighbours o.

$$LRD_K(P) = \frac{|k-neighbors(p)|}{\sum o \subset k-neighbors(p)^{reachability-distance(k)(p,o)}}$$

Compute the average ratio of the local reachability density of p and those of its k-Neighbors:

$$LOF_K(p) = \frac{\sum o \subset k-neighbors(p)^{\frac{LRD_K(o)}{LRD_K(p)}}}{|k-neighbors(p)|}$$

A point is considered to be an outlier if it has a significantly greater amount than 1.

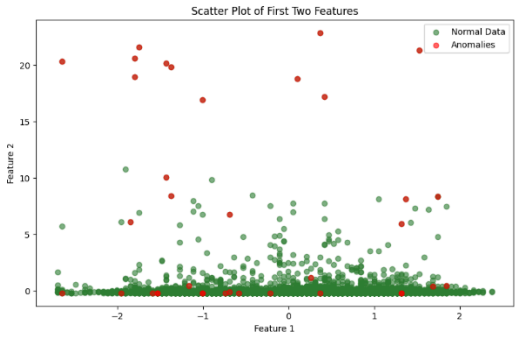

Figure 8. Scatterplot of outlier detection on continuous data using Isolation Forest

Again, the LOF failed to capture many of the outliers present in the dataset. The total number of outliers detected was again 36. By finding the intersection set of the outliers detected by the previous methods, we had great results with several outliers equal to 13.

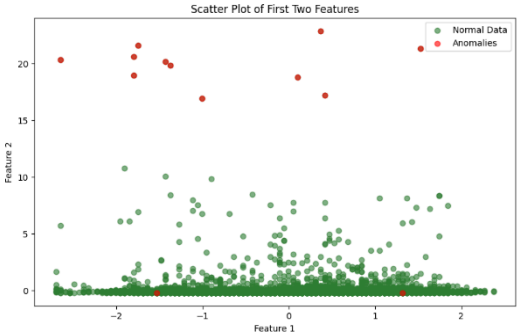

Figure 9. Scatterplot of the final outliers detected

5.3. Mixed Attributes

5.3.1 Hierarchical Clustering

According to the clustering process, existing algorithms are classified into two major types: partitioning clustering and hierarchical clustering [8]. Partitioning clustering is based on dividing the data into non-overlapping clusters like the famous k-means or DBscan. They have an objective function that must be optimized. Hierarchical clustering is based on nesting clusters inside of each other so that it is organized like a hierarchical tree. The strength of this algorithm comes from being able to cluster without assuming several clusters. Any number of clusters can be achieved by cutting the dendrogram tree at a certain distance. Hierarchical clustering itself is divided into two types: Agglomerative and Divisive clustering. In Divisive clustering, we keep splitting until each cluster contains a single point or we reach a certain number of clusters. On the other hand, Agglomerative clustering, which we have used in our paper, starts with each point as a cluster and keeps merging the closest pair of clusters until there is one single cluster or a certain number of clusters left. One of the most important things to consider while performing agglomerative clustering is the linkage method, as the results differ completely depending on the linkage method used whether it is **single**, **complete, average, centroid, ward, or median.** The differences between those linkage types are as follows:

- **Single or minimum**: Measures the distance between the closest pair of points between two clusters.
- **Complete or maximum**: Measures the greatest distance between a pair of points between two clusters.
- **Average**: Measures the average distance between all pairs of points, one in each cluster.
- **Centroid**: Measures the distance between the centroids of the clusters.
- **Ward**: Measures the increase in the total within-cluster variance when two clusters are merged.
- **Median**: Measures the distance between the median points of clusters.

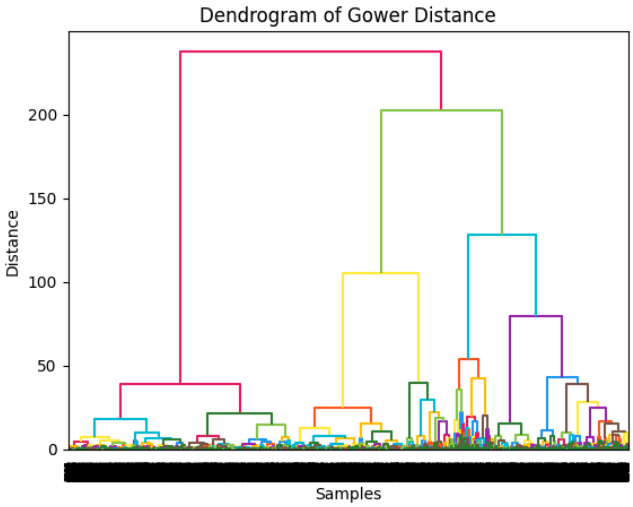

Figure 10. Dendrogram

After comparing different types of linkage, we chose the Ward as it has the best results in clustering such diverse data. We cut the forest at a distance = 39 so we could get a

decent number of clusters after analyzing the dendrogram which was 11 clusters. However, this method fails to capture all the outliers accurately despite achieving a very good silhouette score which was 0.438.

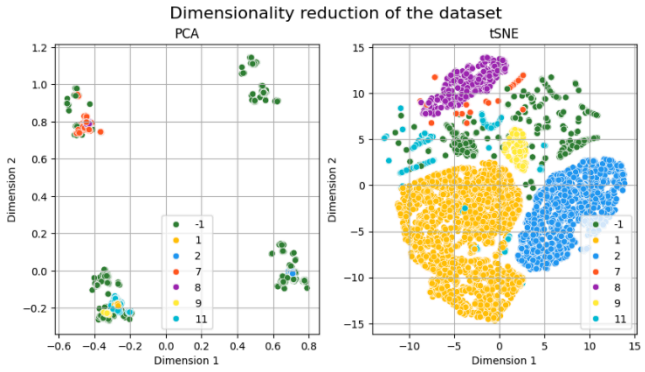

Figure 11. Outlier detection using Hierarchical Clustering

5.3.2 Isolation Forest

Isolation Forest was applied for the first time but now on all the data. However, it failed again in achieving good results.

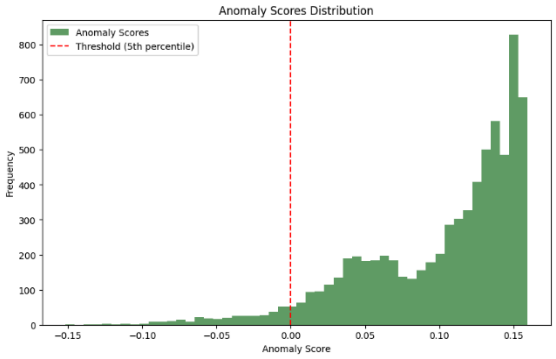

Figure 12. Anomalies score distribution

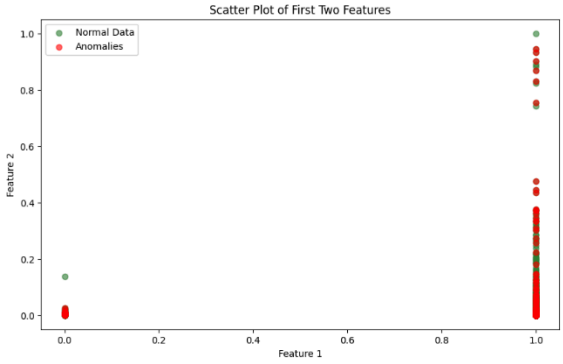

Figure 13. Scatter plot of anomalies detected by Isolation Forest

5.3.3 Variational Autoencoder (VAE)

A Variational Autoencoder (VAE) belongs to the family of autoencoders, which are neural networks designed to learn efficient representations of data, typically for dimensionality reduction and feature learning. The VAE works as follows:

- **Encoder**: in the normal auto-encoders, the encoder takes the input x and transforms it into a latent space z. In the case of VAE, the encoder outputs a probability parameter that describes the distribution z such as mean μ or variance σ^2.
- **Latent space** z **:** a lower dimensional space that captures the relevant features.
- **Decoder**: outputs the points in the latent space z back to their original form as much as possible.
- **Objective Function**: During the training VAEs try to measure how accurately the re-construction is performed by measuring the reconstruction loss.

After training our model on 50 epochs with 100 batch size while using the MSE (Mean-Squared Error) as the reconstruction error and setting the threshold to be the 99th percentile of the MSE loss, we were able to achieve our most accurate outlier detection algorithm that succeeded in detecting 72 outliers.

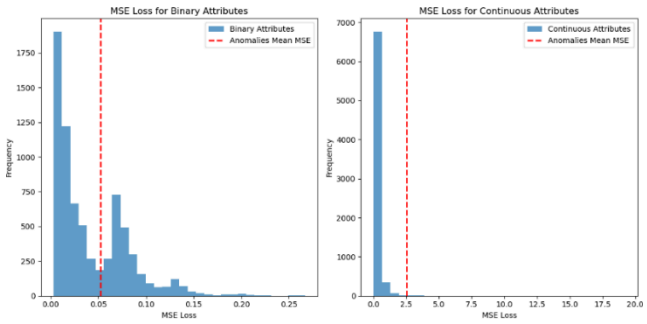

Figure 14. Histogram of MSE for binary and continuous attributes

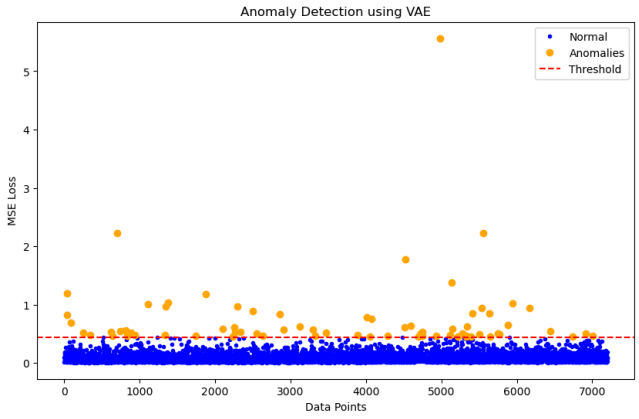

Figure 15. Scatterplot of outlier detection using VAE

Probabilities of each data point being an outlier or not were recorded using the MSE.

P	Q	R	S	T	U	V	W	X
feature_16	feature_17	feature_18	feature_19	feature_20	feature_21	MSE Loss	Anomaly	Anomaly Probability
1.106869082	-0.1854602397	-0.6601680232	0.2938776821	-0.8299398571	0.9083027793	0.08998824941	FALSE	0.01494347804
-1.482215372	-0.2006836979	1.329572821	0.9333771336	1.842566631	-0.1444726734	0.02020914792	FALSE	0.00239159764
-0.2669282445	-0.1288797204	0.5336764831	-0.2065885831	1.73776534	-0.9756111886	0.1246892635	FALSE	0.02118550423
0.6313232683	-0.1724049711	-0.394869244	-0.9016880078	-0.410723375	-0.7816700029	0.03155071209	FALSE	0.00443172074
-1.535050414	-0.2006836979	0.7989752623	0.8221605606	-0.410723375	1.102243965	0.1846937952	FALSE	0.03197913282

Figure 16. A snippet of the CSV file recording the probability of being an outlier for each data point.

6. Discussion

It is necessary to acknowledge the limitations of this work, mainly three limitations. The first limitation encountered was the presence of mixed data types of attributes, both continuous and binary types. These different data types require different methods for preprocessing and anomaly detection. Additionally, it is challenging to understand the correlations of mixed data types. The second limitation we encountered was a lack of context and specific descriptions of the dataset and the attributes which limited us in applying domain-specific knowledge that could have improved the data analysis and anomaly detection process. Lastly, the lack of context made it difficult to establish a clear ground truth which resulted in needing a concrete way to evaluate the results.

Future work should focus on developing advanced feature engineering techniques and specialized algorithms tailored to mixed-type datasets, incorporating contextual information to improve model performance as well as specific methods to evaluate unsupervised anomaly detection models.

7. Conclusion

In conclusion, the problem requires detecting anomalies in a Hypothyroidism dataset without giving any prior knowledge about the nature of its attributes and their significance. We had to go through the pre-processing of the dataset first and clean it before starting to apply anomaly detection algorithms. After proper preparation of the data, we came to the decision of which algorithms were suitable to apply for this kind of data which is a mix of continuous and binary data. The number of binary attributes wasn't that small to neglect in our calculations, so we had to include them. However, dealing with this kind of data that has a lot of binary attributes is challenging as they lead to high dimensionality and sparsity. This sparsity can complicate the detection of patterns and anomalies.
To deal with this sparsity we had to normalize our continuous attributes using a standard scaler normalization to be in the range of the binary values. Second, we decided to split our workflow into three major steps which were: detecting outliers from binary data, detecting outliers from continuous data, and last detecting the outliers from the whole data. Keep in mind that before proceeding with the detection of outliers from the whole data, we had to come up with a proper proximity measure that deals with this kind of data which was the Gower distance.
Our results were as follows:

- **Detecting outliers from binary data:**
  - **Mahalanobis Distance:** The number of outliers detected was 90.
  - **Isolation Forest:** The number of outliers detected was 36, however, it wasn't that effective.
  - **We took the common outliers between them which were 14.**

- **Detecting outliers from continuous data:**
  - **Principal Component Analysis (PCA):** A decent result was achieved, with several outliers detected equal to 15.
  - **Isolation Forest:** Failed to detect many outliers with several outliers detected equal to 36.
  - **Local Outlier Factor (LOF):** Also, failed to detect many of the outliers with several outliers detected equal to 36.
  - **We took the common outliers between them which were 13.**

- **Detecting outliers from the whole (mixed) data:**
  - **Hierarchical Clustering:** after using the Gower distance as a distance metric, it failed to detect many of the outliers.
  - **Isolation Forest:** as usual it failed to detect many outliers.
  - **Variational Autoencoder (VAE):** This was our final decision after being very successful in detecting outliers using the MSE as a

  reconstruction error. The number of outliers detected was 72.

8. Disclosure Statement

The authors declare that this report is entirely original and does not contain any plagiarism. The research explained in this report was conducted by the authors themselves, and all the sources have been cited in the Reference Section. None of the content was generated using automated language models.

9. References

[1] L. Zhang and L. Zhang Big Big Data Analytics for Fault Detection and its Application in Maintenance Operation and Maintenance Engineering.

[2] D. (Daphne) Yao, X. Shu, L. Cheng, S. J. Stolfo, E. Bertino and R. Sandhu, "Anomaly Detection as a Service: Challenges Advances and Opportunities", Synth. Lect. Inf. Secur. Privacy Trust, vol. 9, no. 3, pp. 1-173, Oct. 2017.

[3] N. Schwenzfeier and V. Gruhn, "Towards a practical process model for anomaly detection systems", Proceedings of the 1st International Workshop on Software Engineering for Cognitive Services - SE4COG '18, pp. 41-44, 2018.

[4] N. Görnitz NICOGOERNITZ, K. Rieck KONRADRIECK and U. Brefeld, Toward Supervised Anomaly Detection Marius Kloft, 2013.

[5] Muhammad Ali, P., & Faraj, R. (1990). Data normalization and standardization: A technical report. https://doi.org/10.13140/RG.2.2.28948.04489

[6] Groenewald, E., & Vuuren, G. (2024). Visualisation of Mahalanobis distances for trivariate joint distributions. International Journal of Economics and Financial Issues, 14, 203-206. https://doi.org/10.32479/ijefi.15663

[7] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in *Proceedings of the 2008 Eighth IEEE International Conference on Data Mining*, Pisa, Italy, 2008, pp. 413-422, doi: 10.1109/ICDM.2008.17.

[8] Nielsen, F. (2016). Hierarchical Clustering. In: Introduction to HPC with MPI for Data Science. Undergraduate Topics in Computer Science. Springer, Cham. https://doi.org/10.1007/978-3-319-21903-5_8