Anomaly Detection in Hypothyroidism Dataset

Alessio De Luca
ID: 919790
a.deluca63@campus.unimib.it

Camilla Tomasoni
ID: 915297
c.tomasoni5@campus.unimib.it

Università degli Studi di Milano-Bicocca
MSc Artificial Intelligence for Science and Technology
July 2024

Link to Colab Notebook:
`https://drive.google.com/file/d/1-LZYJeoU-TX8ARTcR4NOQUNFH6qMKCaP/view?usp=sharing`

Abstract **- The aim of this report is to present our methodology for anomaly detection within the hypothyroidism dataset. We discuss the steps involved in data preprocessing and anomaly detection through unsupervised learning techniques, followed by a comparison of the different results and final conversion to anomaly probabilities with logistic regression.**

Index terms **- anomaly detection, unsupervised learning**.

1 Introduction

Anomaly detection refers to the identification of rare items, events, or observations that deviate significantly from the majority of the data. These anomalies can indicate critical incidents, such as technical faults, fraud, or defects, making their timely detection crucial in various fields like finance, healthcare, and cybersecurity. In our case we are dealing in an unsupervised scenario, which means we don't have the true labels associated with the observations.

2 Dataset Description

The dataset consists of 7,200 objects, each characterized by 21 variables. Specifically, 15 columns describe categorical features whose value is either True or False, while 6 columns represent continuous features with values in the range [0,1].

Before proceeding with the examination of the dataset, we identified 71 duplicate rows. However we decided not to delete them, as there is the possibility that they are not errors.
Next, we examined the univariate distributions of the features. For the continuous ones, Figures 1 and 2 show that the values mostly range from 0.0 to 0.5, with the only exception being Dimensions 0 and which takes on more distributed values.

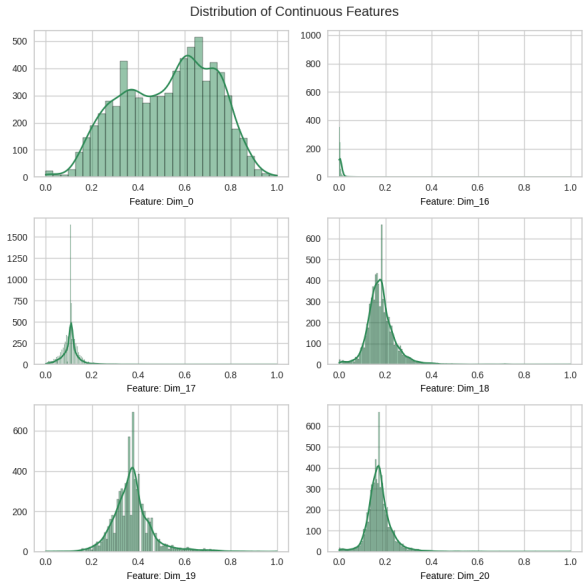

Figure 1: Univariate distributions of the continuous features.

Then, to better understand the relationships between the variables, we calculated the Pearson correlation coefficient for each pair. This is one of the most commonly used measures of linear correlation, and ranges from -1 (perfect negative correlation) to 1 (perfect positive correlation).

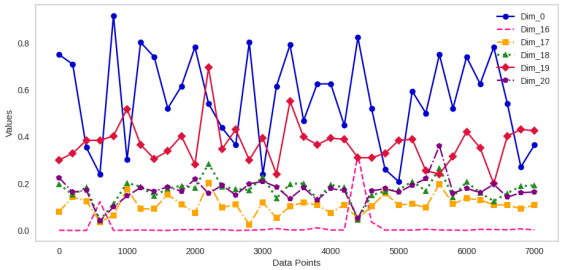

Figure 2: Visualization of continuous features sampled every 200 data points.

As shown in the correlation heatmap in Figure 3, dimensions 18 and 20 have a much higher correlation than the others, with a coefficient of 0.788. This suggests that we could eventually remove one of these variables without a significant information loss, but we chose to keep them both since we don't know the variables' real interpretation.

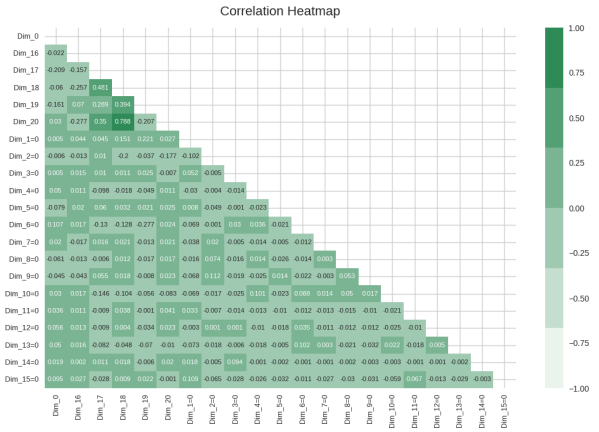

Figure 3: Correlation matrix based on Pearson correlation coefficient.

2.1 Dissimilarity Measure

In anomaly detection tasks, one of the first step consists in selecting an appropriate metric for measuring the distance between two data points. Since our dataset is a mix of categorical and quantitative variables, we have opted for Gower's distance, which is defined for handling different data types. For quantitative variables, Gower's distance utilizes range-normalized Manhattan distance. For categorical variables, it employs the Dice coefficient, which evaluates the similarity between two sets based on their overlap.

In Figure 4 we can see that the values of Gower's distance vary in the range [0,0.4].

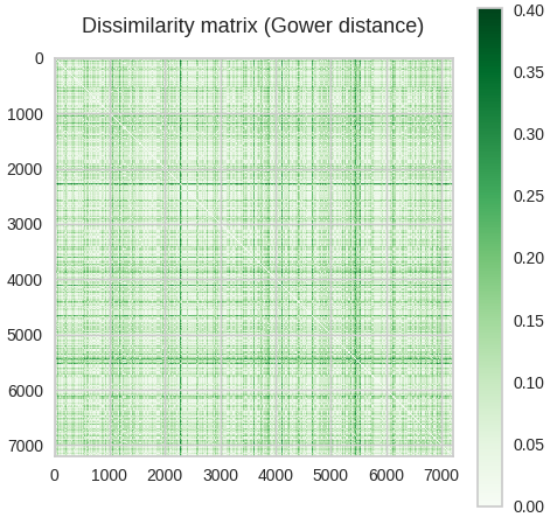

Figure 4: Gower distance matrix.

3 Anomaly Detection Algorithms

The primary types of anomaly detection algorithms can be categorized into proximity-based, cluster-based, statistical approaches, and reconstruction-based methods.

Proximity-based methods identify anomalies by examining the distances between data points, with anomalies being those that are far from their neighbors. Cluster-based methods detect anomalies by grouping data into clusters and identifying points that do not fit well into any cluster. Statistical approaches rely on statistical distributions to model the data, identifying points that deviate significantly from the expected distribution as anomalies. Reconstruction-based methods involve the projection of the objects into a lower-dimensional space and a reconstruction in the initial space. The distance between the original and reconstructed data is used to identify anomalies.

Generally, each method, except for DBSCAN, produces either a distance value or an anomaly score for each data point. After sorting these scores from lowest to highest and plotting them for all 7200 data points, we employed two methods to identify anomalies:

- *Knee Locator*: The Knee of a curve is defined as the point of maximum curvature, and the outliers are the objects with an anomaly score greater than the value determined by the Knee Locator.
- *Top n%*: the n points with the greatest anomaly score are labeled as anomalies. We chose to use the top 5%, meaning that each method identified as anomalies the 360 objects with the highest anomaly score.

Each type of plot in the following report has been generated, when possible, for each method. However, to avoid overwhelming the report with images, we have chosen to include only one representative plot for each type.

3.1 Connectivity-Based Outlier Factor

COF stands for Connectivity-Based Outlier Factor, a density based approach for anomaly detection. It assigns an outlier score to each data object, considering both its distance from nearby points and the geometric characteristics of its local neighborhood. Formally, an outlier is identified as a data point whose average chaining distance is larger than the average chaining distance of its k nearest neighbors. This definition determines anomalies as points whose neighborhood is sparser than that of their neighbors.
The average chaining distance is defined as:

$$ac - dist_{kNN(p)} = \sum_{j=1}^{r} \frac{2(r-1)}{r(r-1)} dist(e_i)$$

To implement COF, we defined a function to find the k-nearest neighbors for each point, where a rule of thumbs suggests $k = \sqrt{7200} \approx 84$.
For each point, the function calculates the average chaining distance using the formula above. Then, it computes the COF score, which is defined as:

$$COF(p) = \frac{average\ chaining\ distance\ of\ p}{average\ ac\text{-}dist\ at\ p\text{'s}\ neighborhood}$$

We plotted the sorted COF scores and identified the outliers as the points with the highest COF score. In particular, we obtained 210 anomalies with the Knee Locator criterion, and 360 as the top 5% anomalies, which are shown in Figure 5.

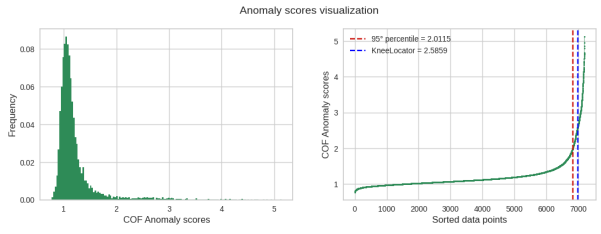

Figure 5: Visualization of the anomaly scores for COF. The left graph shows the frequency of the anomaly scores. The right graph shows the sorted scores and the criteria to identify the anomalies.

3.2 Principal Component Analysis

In anomaly detection tasks, Principal Component Analysis (PCA) is used as a reconstruction-based approach. This method reduces the dimensionality of a dataset to create a smaller set of features. Each data point is first represented in this lower-dimensional space and then projected back to the original space. Anomalies are identified as the data points with the largest reconstruction error, measured by the distance between the original and reconstructed points.

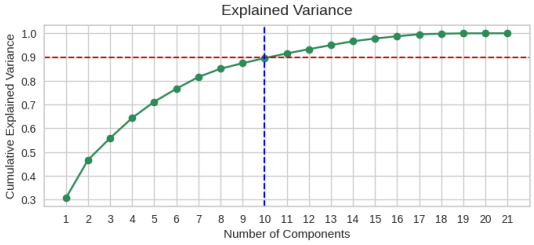

Figure 6: Cumulative explained variance of the dataset.

$$ReconstructionError = \|x - \hat{x}\|$$

where x is the original object and $\hat{x}$ is the reconstructed object.
In our approach, we initially computed the explained variance, which is a measure of how much the total variance in the dataset is explained by each component. A rule of thumb for generating the lower-dimensional space is to select a number of variables that maintain at least 80% of the variance. However, in our analysis we chose to retain 90% of the variance, corresponding to 10 principal components, since the curve of the cumulative explained variance starts to flatten between 0.9 and 1.0, as depicted in Figure 6.
Then, we computed the PCA by representing the objects in the new 10-dimensional space, and reconstructing them in the original space. This allowed us to compute the reconstruction errors, which were then sorted and plotted. Anomalies were identified as data points with the highest reconstruction error. We obtained 360 anomalies according to the top 5% criterion and 127 anomalies using the Knee Locator (Figure 7).

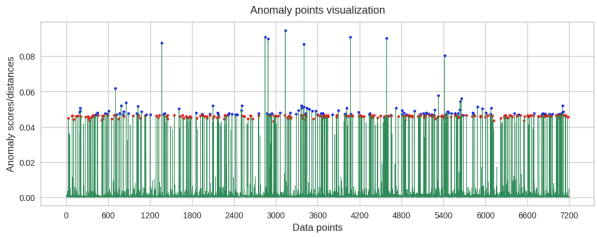

Figure 7: Visualization of the anomalies of PCA for not sorted data points. Blue points are the anomalies identified by the Knee Locator. Red points together with the blue ones are the top 5% anomalies.

A final consideration concerns the characteristics of normal data and anomalies identified by PCA.

In particular, we generated radial plots that display the mean values for all six continuous features of normal points and anomalies. This analysis helps us to highlight the dimensions where anomalies differ significantly from normal points. In this case, we can observe that outliers tend to assume higher values in the Dimension 19, while they have the same mean value of normal objects in Dimension 20.
We chose to report the radial plots for PCA since it is the method in which the overlap percentage is the lowest (84.25%).

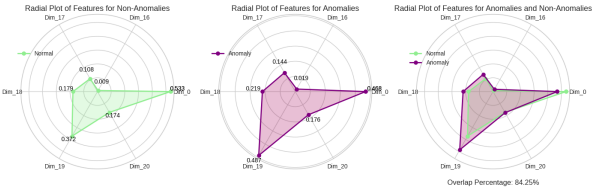

Figure 8: Radial Plots of normal data and anomalies for PCA.

3.3 K-Nearest Neighbors

KNN is a proximity-based approach where outliers are identified by their distance from other points. Initially, distances to the k nearest neighbors are computed for each object, and either the k^th distance or the average between the k distances is considered for each point. The outliers are identified as the points having the largest distance. A common rule of thumb is to choose k as the square root of the number of objects. For our dataset, this yields $k = \sqrt{7200} \approx 84$.
We computed the KNN and determined the average distance to the neighbors for each point, then we sorted these distances and found 153 anomalies with the Knee Locator, 360 with the percentage criterion (Figure 9).

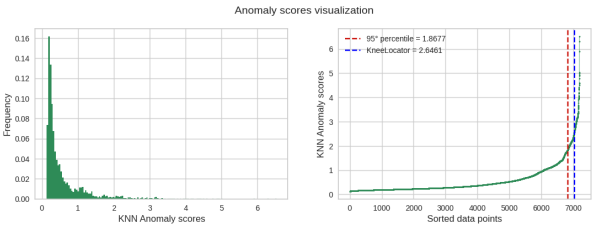

Figure 9: Visualization of the anomaly scores for KNN.

To visualize the dataset distribution and the identified anomalies, we created two distinct 2D plots, as illustrated in Figure 10. The plot on the left displays two continuous dimensions, Dim 16 and Dim 19, whereas the plot on the right shows the first and second dimensions derived from the t-SNE reconstruction.
t-Distributed Stochastic Neighbor Embedding (t-SNE) is a machine learning algorithm used for dimensionality reduction, particularly well-suited for the visualization of high-dimensional data. It works by converting the similarities between data points into joint probabilities and then minimizes the divergence between these joint probabilities in the high-dimensional and low-dimensional spaces.
t-SNE effectively captures the local structure of the data while also revealing global patterns, making it useful for visualizing clusters and anomalies. The algorithm aims to minimize the Kullback-Leibler (KL) divergence between the probability distributions, which in this case is 1.96.

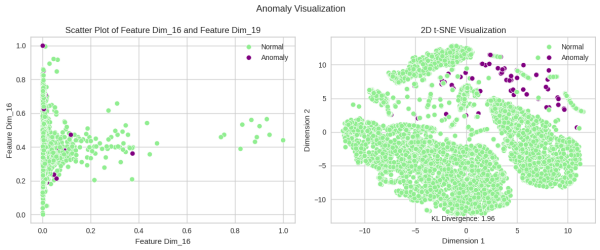

Figure 10: 2D scatter plots of the dataset. On the left using two dimensions in the dataset while on the right using t-SNE reduction.

3.4 Density-Based Spatial Clustering of Applications with Noise

DBSCAN is a density-based clustering technique that groups data into cluster based on point density, and it is sensible to two parameters: ϵ and $minPoints$.
The model creates a circle of radius ϵ around each point and classifies data points as follows:

- Core points: points within a cluster.
- Border points: points in the neighborhood of a core point.
- Noise points: points that do not belong to any cluster.

The parameter $minPoints$ specifies the minimum number of points required within a circle for it to be considered a cluster. A common rule is that its value must be at least greater than the number of dimensions.
Since we have a large dataset we chose:

$$\text{minPts} = 2 \times \text{number of dimensions} = 42$$

Instead, ϵ is determined using the K-distance graph. Specifically, we first computed the k-nearest neighbors (KNN) with $k = minPoints$, plotted the sorted distances, and identified the optimal value of ϵ at the point of maximum curvature on the graph, that is $\epsilon = 1.7$, as it is shown in Figure 11.
After selecting the parameters we applied DBSCAN, which assigned a label to each object, where a label of -1 indicates anomalies. The analysis identified a total of 322 anomalies.

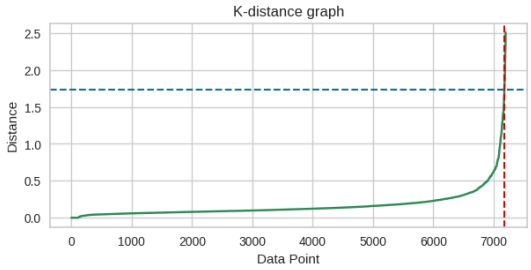

Figure 11: K-distance graph for DBSCAN. The value of ϵ is given by the blue line.

4 Comparison

Since we worked in a totally unsupervised scenario, without true labels and without the possibility to conduct a supervised evaluation of the models, we just compared the anomalies identified by various algorithms.
To quantify the similarity between the lists of anomaly indices obtained with different methods, we employed the Intersection over Union (IoU) metric. The IoU metric is calculated as follows:

$$IoU = \frac{\mid Intersection\ of\ the\ two\ lists \mid}{\mid Union\ of\ the\ two\ lists \mid}$$

An IoU value of 1.0 indicates that the intersection of the two lists is identical to their union, meaning the two lists are identical. Instead, a value of 0.0 implies that the lists have no elements in common, indicating that the anomalies identified by one method are completely different from those identified by another method.
As Figure 12 shows, the methods whose output is most similar are the DBSCAN and the KNN according to the 5% criterion, with a IoU value of 0.894.

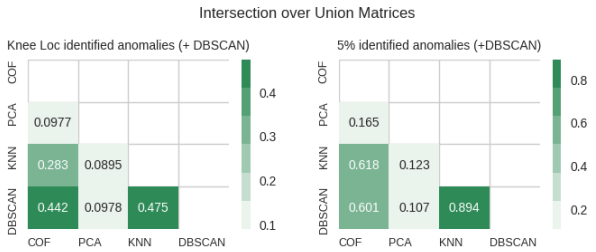

Figure 12: Intersection over Union values.

5 Conversion to Anomaly Probability

After having explored four anomaly detection techniques, although there were uncertainties, we ultimately decided to use the anomaly scores derived from PCA obtained with the Knee Locator. This method had the least overlap in the Radial Graph (Overlap Percentage: 84.25%), suggesting clearer anomaly distinction compared to the other methods.
To finalize our analysis, we created a copy of the original dataset and labeled the anomalies. We then trained a logistic regression model on the dataset to calculate the probability of each point being an anomaly. The resulting probabilities were added to the dataset, giving us a perspective on the likelihood of each data point being an anomaly.
We identified 127 anomalies with corresponding probability percentages, recognizing that this approach made the most sense to us despite the inherent uncertainties.

6 Conclusion

Figure 13 summarizes in a nutshell our project: given the unsupervised nature of the task and the limited information available about the dataset, our results could be somewhat accurate in an optimistic scenario but also potentially misleading. Nevertheless, we are confident in our analysis, understanding that without statistics, we wouldn't have been able to attempt to uncover the truth at all.

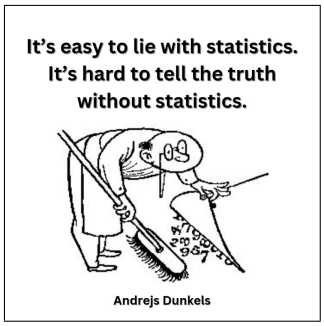

Figure 13: Take home message for the project.

The authors declare that this work has been done independently and does not contain any form of plagiarism.

References

[1] Scikit-learn: Machine Learning in Python. "Unsupervised learning," Available at: `https://scikit-learn.org/stable/unsupervised_learning.html`, 2024

[2] University of Milano-Bicocca E-Learning Platform. "Course: Machine Learning," Available at: `https://elearning.unimib.it/course/view.php?id=51018`, 2024

[3] SciPy Documentation. Available at: `https://docs.scipy.org/doc/`, 2024