UNIVERSITY OF
PATRAS
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

BIG DATA MANAGEMENT

3rd Academic Year Assignment 2023-2024, due 31/01/2024

Collaborators
Georgakis Giorgos, 1052995
Benekos Konstantinos, 1067754
Servos Dimitrios, 1067563
Petroulis Veliassarios, 1074579

Contents

1 Task 1

OLS and Gradient decent are different methods for determining optimal coefficients in linear regression models. OLS uses a direct formula for the solution of the problem, while Gradient decent computes them by starting with a set of random coefficients and adapts them with the purpose of minimizing a cost function. The cost function in gradient decent, as in OLS, is measured by the mean of squared errors. Differences in the results between the two methods arise due to the nature of each of the methods. Gradient decent keeps trying to improve its solution bit by bit, it can be affected by how it starts, since the initial parameters(thetas) are usually set at random, this might lead to it getting stuck in not-so-great (local minima) solutions, especially in complex situations. In comparison, OLS is quicker for smaller to medium-sized sets of data. Yet on the other hand, Gradient Descent is better for handling bigger sets of data and lots of features.

In our case when looking at the gradient decent cost function plotted against the number of iterations, we see that it showcases a smooth decent, meaning that in the first iterations the cost function decreases rapidly, since the algorithm makes significant improvements to reduce the error. As the iterations progress, the rate of decrease in the cost function remains relatively constant. This indicates a well-balanced learning rate that allows the algorithm to converge efficiently. The overall shape of the plot resembles a smooth curve without erratic jumps or oscillations. This indicates that the learning rate is not too large, causing overshooting, or too small, causing slow convergence. Towards the end of the plot, as the algorithm approaches convergence, the rate of improvement in the cost function starts slowing down and the cost function starts to flatten indicating that the algorithm has found, or at least has come close to, finding the minimum.

2 Task 2

The differences between estimated coefficients and cost functions obtained with batch gradient descent and stochastic gradient descent can be attributed to the inherent characteristics of these two optimization algorithms. BGD computes the gradient of the cost function with respect to the parameters using the entire dataset in each iteration. It aims to find the global minimum by considering the average gradient over the entire dataset. SGD updates the parameters based on the gradient of the cost function for a single data point chosen randomly in each iteration. It introduces more randomness and noise in parameter updates, making it less likely to converge to the global minimum but often more adaptable to variations in the data, yet it comes at a smaller computational cost. The nature of these optimizations impacts the convergence behavior, with BGD producing smoother updates and SGD exhibiting more oscillations, something which can be observed when plotting the cost function against the number of iterations. Learning rate plays a crucial role, with BGD applying it to the average gradient and SGD applying it to individual data points, influencing the stability and variability of updates. While BGD is computationally expensive, especially for large datasets, SGD offers greater efficiency by processing one data point at a time.

Overall, in BGD, the cost function is generally smoother and exhibits a more regular decrease over iterations. The updates to the parameters are based on the average gradient computed from the entire dataset, leading to a more stable convergence. As a result, the cost function graph tends to show a gradual and continuous decrease, indicating a systematic approach towards the global minimum. Whilst the cost function plot for SGD is characterized by more irregular fluctuations and oscillations. This behavior arises from the fact that SGD updates parameters based on the gradient of individual, randomly selected data points. While this introduces more variability and noise in the updates, it can lead to faster convergence in terms of iterations.

Both algorithms are sensitive to the choice of learning rate, and the cost function plots can be affected accordingly. If the learning rate is too large, both BGD and SGD might exhibit overshooting or divergence, while too small of a learning rate can slow down convergence.

3 Task 3

3.1 I

The provided code begins by generating an empty data frame called my data with 7 columns all (y, x1. . . .x6) all numeric, it then precedes to generate 4 rows, using a loop. During each iteration, generates a row of random values between 1 and 10 using the "runif" function. Each row corresponds to a different observation. (in the code: 7 being the number of random values to generate 1 and 10 being the minimum and maximum values possible). It then fits a linear model where "y", the first column, is the dependent variable and all other columns are used as independent variables (in the code: this is indicated using the dot "."). Finally, the codes print's the coefficients of the linear model.

3.2 II

The value of the coefficient associated with a variable gives us an indication of how that independent variable is related to the dependent variable. The direction (positive or negative) of the coefficient indicates the direction of the relationship between the independent and dependent variables. The magnitude of the coefficients gives us an idea of the importance of each variable in the prediction. It is important to note that the scale of the independent variables can affect the scale of the coefficients in a linear regression model. When the scale of the variable's changes, the scale of the coefficients can change proportionally. This is because the coefficients represent the change in the dependent variable for a one-unit change in the corresponding independent variable. In our case all independent variables as well as the dependent variable take values in a range of 1-10 hence the coefficients, will always be within the 1-10 range, consistent with the scale of your variables. Finally, we notice that no matter how many times we run the model the coefficients for x4, x5 and x6 always come back as NA. The consistent occurrence of NA values for x4, x5, and x6 in the linear regression model is attributed to perfect multicollinearity within the dataset. This multicollinearity leads to a singular or rank-deficient model matrix, rendering the coefficients for these variables undefined. The correlation matrix highlights a notable correlation among x4, x5, and x6, and the condition number of the design matrix indicates near singularity. This effect could persist, even if the numbers are generated at random each time we run the code, due to the small sample size which leads to unstable estimations, we could also explore regularization techniques such as ridge regression to mitigate the impact of multicollinearity or examine the data for patterns in the values of x4, x5 and x6.

3.3 III

Based on the results of the executed program and their interpretation, we can conclude that the method of least squares is sensitive when perfect multicollinearity exists among variables. Multicollinearity leads to an almost singular estimation matrix, resulting in the inability to determine certain coefficients. This could lead to issues such as the inability to estimate specific coefficients and an increased risk of overfitting.

Mathematically, the matrix X^TX represents the result of multiplying the transpose of the matrix X, which contains the independent variables of a linear regression model, by X itself. In OLS, this is done to compute the variance-covariance matrix of coefficient estimates, detecting multicollinearity through the condition number, and, most importantly, to obtain estimated coefficients through $(X^TX)^{-1}X^Ty$. Here, X is the matrix of independent variables, y is the vector of dependent variable values, X^T is the transpose of X, and $(X^TX)^{-1}$ is the inverse of the matrix product of X^TX. Overall, the above expression arises from solving the normal equations in the method of least squares. The objective is to minimize the sum of squared residuals, representing the squared differences between observed and predicted values. To find the coefficients minimizing this sum, the calculus of variations is used. By setting the derivative of the sum of squared residuals with respect to each coefficient to zero, a system of linear equations known as the normal equations is obtained. Solving these equations involves the matrix X^TX and its inverse $(X^TX)^{-1}$. The multiplication of $(X^TX)^{-1}X^T$ by the vector y yields the coefficients that minimize the sum of squared differences, providing an optimal linear model.

Hence, it becomes clear that in the presence of multicollinearity, $X^T X$ may not be invertible. This renders the calculation of the coefficient estimates impossible, and without a unique estimation of coefficients, the solution is not uniquely determined.

4 Task 4

In situations where the training error is very small (indicating a good fit to the training data) but the generalization error is very large (indicating poor performance on new, unseen data), we describe this phenomenon as overfitting. This occurs when the model fits the training data too closely, making it less effective for predicting outcomes on unfamiliar data.

5 Task 5

In the first scenario, using the entire dataset for prediction results in a Root Mean Squared Error (RMSE) of 42. In the second scenario, where we only consider observations with a dependent variable ('area') less than 3.2 hectares (i.e., $area < 3.2$), the RMSE reduces significantly to 0.79. This indicates that our model excels at predicting small fires in this specific situation. The improvement can be attributed to the narrower range of values in the dependent variable, leading to smaller coefficients and a considerable enhancement in prediction accuracy. It also makes the coefficients less sensitive to large values in the dataset. Moreover, our method of calculating errors makes the model responsive to extreme values, further influencing the precision of our predictions.

6 Task 6

The Gradient Boosted Regression Trees (GBRT) method belongs to the category of machine learning algorithms and is primarily used for regression and classification problems. It goal is to create a robust model by combining weak models, typically decision trees. GBRT combines the power of decision trees with the concept of continuous improvement through the minimization of the error gradient. The operation of GBRT begins with the construction of a weak model, usually a decision tree, which makes an initial prediction. Then, the residual error is calculated, i.e., the difference between the actual value and the prediction of the first model. Subsequently, a new weak model is constructed, focusing on predicting this residual error. The predictions of the weak models are combined in some way, usually by adding them, to create an improved overall prediction. The cycle is repeated, with new models focusing on predicting the remaining error. The process is iterated until the desired performance is achieved or until a predefined number of iterations is completed.

Can be used on various types of data, as it is a powerful machine learning technique that can tackle diverse challenges. Among the types of data that can be used are:1)Numeric data: This can be useful, for example, in problems involving the prediction of values, such as predicting the stock price.2)Categorical data: GBRT can handle categorical data by adapting the structure of their trees to deal with categorical variables.3)Data with missing values: GBRT are relatively robust to missing values and can handle these gaps during

training. While they are powerful and flexible, there are some cases where they may face challenges or may not be the preferred choice. Such as Large datasets: in very large datasets with a high number of features, training a GBRT can be time-consuming and resource-intensive.Also Imbalanced Data: in cases where the data is unbalanced in terms of classes, GBRT may have problems predicting the least representative class.And finally Overly simple problems: In very simple problems where other algorithms may be more efficient and require fewer resources.

Next, I will examine three publications that analyze the Gradient Boosted Regression Trees method. The text titled 'Forecasting of Real GDP Growth Using Machine Learning Models: Gradient Boosting and Random Forest Approach' by Jaehyun Yoon presents a method for creating machine learning models, specifically a gradient boosting model and a random forest model, to predict the real GDP growth of Japan. The study focuses on the actual GDP growth of Japan and generates predictions for the years from 2001 to 2018. The forecasts of the International Monetary Fund and the Bank of Japan are used as reference points. The study uses cross-validation to select the optimal hyperparameters, aiming to improve out-of-sample prediction. The accuracy of predictions is measured using the Mean Absolute Percentage Error (MAPE) and the Root Mean Squared Error (RMSE). The results of the study indicate that, for the period 2001-2018, the predictions from the gradient boosting model and the random forest model are more accurate than the reference points' predictions. Additionally, among the two models, the gradient boosting model is more accurate. The text emphasizes the role of machine learning models in forecasting macroeconomic variables, supporting the need for increased use in the field of macroeconomic forecasting. As for the Gradient Boosted Regression Trees method, it is a technique that combines multiple weak models (in this case, decision trees) to improve the accuracy of the final model. The method starts with a model that predicts the variable in a certain way and gradually adds new models, adjusting them to reduce the error of the previous model. This process continues until further improvement is not possible or until a predetermined number of models is reached. Overall, the study provides a significant contribution to the literature, illuminating the performance of machine learning models in predicting GDP growth in Japan.

In the second text titled "A gradient boosting decision tree approach for insider trading identification: An empirical model evaluation of China stock market," the study aims to identify cases of insider trading in the China stock market using a combination of methods, such as Gradient Boosting Decision Tree (GBDT) and Differential Evolution (DE). The study collects data on insider trading cases and non-insider trading cases from 2007 to 2017. It trains the GBDT model, optimizes its initial parameters using DE, and evaluates the model's performance on new data.GBDT is a machine learning algorithm used for classification and prediction problems. It constructs a series of decision trees, where each tree corrects the errors of the previous one. DE is an optimization algorithm used to find the optimal parameters of the GBDT model.The study concludes that the proposed GBDT–DE method outperforms other methods in terms of performance. The optimized parameters from DE improved the performance of the GBDT model. It was also observed that a 90-day time window provided the best identification accuracy, and four specific indicators were significant for insider trading identification.Overall, the study aims to contribute to the detection of insider trading in the China stock market using machine learning techniques.

The GBDT–DE method is suggested as more effective compared to other methods, and the specific indicators highlighted by the model provide interesting information for identifying insider trading activity.

The third text titled "Credit Risk Assessment based on Gradient Boosting Decision Tree" examines the importance of the credit risk assessment system, focusing on the Gradient Boosting Decision Tree method for determining the creditworthiness of individuals or businesses. The text discusses an experiment where this method is compared with other models such as SVM, Decision Tree, and MLP, concluding that the Gradient Boosting Decision Tree method is one of the best, providing high accuracy. The question examined is the risk of creditworthiness due to subprime loans, which can lead to liquidity crises. The need for effective financial risk management systems is emphasized, with a focus on regulation and the importance of creditworthiness.The concept of Gradient Boosting Decision Tree is introduced as an efficient machine learning method for data categorization and generalization. Subsequently, a detailed description of the experiment is provided, including data processing, feature selection, the use of the SMOTE algorithm for data balancing, and the adjustment of parameters for the Gradient Boosting Decision Tree model. Additionally, parameters tuned for model optimization, such as the number of estimators, learning rate, and minimum split reduction, are analyzed.Finally, the significance of creditworthiness for businesses and financial institutions is highlighted, along with the need for reliable prediction models. Possible improvements to the methodology are suggested, including data processing and sample collection, as well as the exploration of other advanced models. The text concludes by indicating that prospects for future research include further refining the methodology and examining different models depending on the nature of the problem.

7 Task 7

The paper discusses cluster analysis algorithms, particularly in the context of economic data. Cluster analysis aims at identifying groups of similar objects based on selected variables, overall cluster analysis is a powerful tool in multivariate data exploration. The fundamental concept in cluster analysis is similarity. Similarity is frequently assessed by calculating the distance between two objects. When dealing with quantitative variables, a widely used measure is the Euclidean distance. The Euclidean distance formula is expressed as:

$$d(x_i, x_j) = \sqrt{\sum_{l=1}^{m} (x_{i,l} - x_{j,l})^2}$$

Here, $d(x_i, x_j)$ represents the Euclidean distance between objects i and j, m denotes the number of variables (such as economic indicators), and $x_{i,l}$, $x_{j,l}$ are the values for the l-th variable of objects i and j respectively. The formula calculates the square root of the sum of squared differences between corresponding variable values of the two objects, providing a quantitative measure of their dissimilarity/similarity.

Recent developments in cluster analysis focus on challenges like clustering large datasets, handling categorical variables, and incorporating fuzzy clustering. Issues such as outlier detection and cluster number determination remain relevant. The resurgence of interest,

especially in connection with evolving data mining techniques, has led to the proposal of new algorithms. The article explores the evolution of different clustering methods.

The author then goes on to present different clustering algorithms starting with the classical ones, namely Hierarchical cluster analysis and K-clustering. The first algorithm discussed, and arguably the most widely used in the context of economic data, is agglomerative hierarchical cluster analysis, this method utilizes a proximity matrix, which includes the similarity evaluation for all pairs of objects, accommodating various measures for different variable types. It offers advantages such as graphical output in the form of a dendrogram, especially useful for smaller datasets. The article demonstrates the application of hierarchical cluster analysis on EU countries based on economic activity rates, providing insights into country groupings. Additionally, the paper explores clustering nominal variables. The basic measure when dealing with nominal variables is simple matching coefficients or in other word the overlap measure. The similarity between vectors xi and xj, denoted as sij, is calculated by comparing values in the ith and jth rows for all variables. The relationship for the lth variable is expressed as slij. If xil = xjl, then slij = 1; otherwise, slij = 0. The overall similarity, sij, is computed as the arithmetic mean.

$$s_{lij} = \frac{\sum_{l=1}^{m} s_{lij}}{m}$$

The article also motions that additionally hierarchical cluster analysis can directly utilize a proximity matrix to assess relationships among all pairs of variables, facilitating the identification of similarity in variables and groups of variables within a dendrogram.

Additionally, the paper explores the challenges and solutions related to clustering nominal variables, introducing measures like Eskin, OF, IOF, and Lin to address mismatch weights and varying category numbers, since the overlap measure does not consider different numbers of categories for individual variables.

The next algorithm discussed is k-clustering, here objects are divided into a specified number (k) of clusters, and various approaches can be classified from different perspectives. The primary classifications include hard and fuzzy clustering. Hard clustering assigns each object exclusively to one cluster, generating a membership matrix with ones (object assigned to cluster) and zeroes (object not assigned). In fuzzy clustering, membership degrees are calculated for all cluster-object pairs, introducing uncertainty expressions in cluster analysis.

Another classification distinguishes k-centroid and k-medoids clustering. In k-centroid, the cluster center is represented by a vector of variable characteristics, while k-medoids represents the center by a selected object from the input matrix. One widely used k-centroid technique is the k-means algorithm, which minimizes an objective function involving Euclidean distances.

$$J_{HCM} = \sum_{h=1}^{k} \sum_{i=1}^{n} u_{\text{hard},ih} d_{ih}^2$$

The variables $u_{\text{hard},ih}$, taking values in the set $\{0, 1\}$, represent the assignment of object vectors to clusters, where 1 signifies the assignment. The term d_{ih}^2 corresponds to the Euclidean distance between the i-th object and the center (a vector of means) of the h-th cluster. The following conditions must be satisfied:

$$\sum_{h=1}^{k} u_{\text{hard},ih} = 1 \quad \text{for } i = 1, 2, \ldots, n$$

$$\sum_{i=1}^{n} u_{\text{hard},ih} > 0 \quad \text{for } h = 1, 2, \ldots, k$$

Overall, the objective function J_{HCM} represents the Hard C-Means (HCM) algorithm's goal to minimize the sum of squared Euclidean distances in the process of clustering. In the equation, k signifies the number of clusters, n denotes the total number of objects, $u_{\text{hard},ih}$ is a binary variable indicating whether object i belongs to cluster h, and d_{ih}^2 represents the squared Euclidean distance between the i-th object and the center of the h-th cluster. The objective is to find optimal assignments and cluster centers that minimize the total within-cluster sum of squares. The conditions ensure that each object is precisely assigned to one cluster and that each cluster has at least one assigned object, essential for the coherence of the clustering results.

The text notes that K-centroid clustering is advantageous for large datasets due to its scalability but is prone to instability, yielding different results for various orders of object vectors. The outcome is influenced by the initialization method for determining initial centroids, and while these methods seek optimal solutions, the achieved optimality may be local rather than global. Despite these challenges, k-clustering methods play a crucial role in exploratory data analysis.

On the other hand, k-medoids, while more stable, might be less suitable for large datasets. The hard k-medoids algorithm, also known as PAM (Partitioning Around Medoids), minimizes an objective function to determine cluster assignments, where mh represents the medoid of the hth cluster.

$$f_{KM} = \sum_{h=1}^{k} \sum_{i=1}^{n} u_{\text{hard},ih} \|x_i - m_i\|$$

The expression f_{KM} represents the objective function for the k-means algorithm, where k denotes the number of clusters, h and i are indices iterating over clusters and data points, respectively. The term $u_{\text{hard},ih}$, taking values in $\{0, 1\}$, signifies the assignment of data points to clusters, with 1 indicating the assignment. The expression $\|x_i - m_i\|$ corresponds to the Euclidean distance between the i-th data point and the center (mean) of the h-th cluster. The objective is to minimize this function, determining optimal cluster assignments by iteratively updating cluster centroids based on the data points' distances. In contrast to the k-medoids objective function, which uses the actual data point as the center (medoid) of the cluster, the k-means objective function utilizes the mean, making it sensitive to outliers and noise but allowing for efficient application to large datasets.

Next the article discusses fuzzy cluster analysis, fuzzy cluster analysis is implemented to address the inherent uncertainty and imprecision present in real-world data. Unlike traditional hard clustering methods that assign objects strictly to one cluster, fuzzy clustering allows for a more nuanced representation of membership by assigning degrees of belongingness to multiple clusters. Among various algorithms for fuzzy clustering, the fuzzy k-means (FCM) algorithm, also known as fuzzy c-means, stands out. FCM minimizes the objective function, involving membership degrees (uih) with values between 0 and 1, and a fuzzifier

parameter (q) to control the degree of fuzziness.

$$J_{FCM} = \sum_{h=1}^{k} \sum_{i=1}^{n} u_{ih}^{q} d_{ih}^{2}$$

In FCM the fuzziness parameter q controls the sensitivity of these membership degrees to changes in distances between objects and cluster centroids, q is a weighting exponent, and its value is typically set greater than 1. Commonly, q is set to 2 in FCM, known as the Euclidean norm case. When q is 2, the algorithm tends to be more sensitive to the Euclidean distances between objects and cluster centroids. Higher values of q (greater than 2) lead to increased fuzziness, allowing objects to have more evenly distributed membership degrees across multiple clusters. Lower values of q (less than 2) result in sharper membership assignments, making objects more distinctly belong to one cluster. Overall, the parameter q controls the trade-off between soft and hard assignments in fuzzy clustering.

The author mentions encounter challenges with sensitivity to noise and outliers, as exemplified when objects are equidistant from cluster centroids, resulting in equal membership degrees. To address this, alternative algorithms like the possibilistic k-means (PCM) are introduced, minimizing an objective function that incorporates membership degrees a fuzzifier (q), and specific conditions. Furthermore, rough set theory offers an alternative to hard clustering, with the rough k-means (RCM) algorithm defining clusters based on lower and upper approximations. The RCM algorithm introduces two values characterizing the membership of an object, hence providing more nuanced assignments. Combining rough and fuzzy concepts yields the rough-fuzzy k-means (RFCM) algorithm. Additionally, modifications such as the shadowed k-means (SCM) incorporate user-defined thresholds for dynamic cluster evaluation based on original data. These diverse approaches highlight the ongoing efforts to enhance clustering techniques by addressing their inherent limitations and tailoring them to various data characteristics.

Finally, the article delves into other approaches, than hierarchical clustering and K-clustering, which have been proposed to address specific challenges in clustering, such as handling large data sets and incorporating categorical variables. The two-step cluster analysis, exemplified in IBM SPSS Statistics, utilizes the BIRCH algorithm to cluster large data sets with both quantitative and qualitative variables. This method first organizes data into cluster features (CFs) and then applies hierarchical clustering to these CFs. However, it is sensitive to the order of objects. Two-step cluster analysis allows users to employ either Euclidean distance for quantitative data or log-likelihood distance for a combination of quantitative and qualitative variables. This method has been applied to cluster households based on material deprivation indicators, illustrating its effectiveness in handling large data sets with mixed variable types. Alternative techniques, such as CLARA for large applications and BCLUST for high-dimensional data in R, offer additional solutions to clustering challenges. Approaches for clustering categorical data are summarized, and when dealing with mixed-type variables, cluster ensembles, like CLUE for R, provide a viable solution by combining individual cluster solutions for different variable types.

8 Task 8

8.1 I

The apriori algorithm applied to the fertility dataset has generated a total of 496 rules. These rules represent associations between various antecedent conditions, which are factors related to a man's sperm quality, and the consequent diagnosis of the sperm being altered or normal (O or N). Each rule provides insights into the relationships and patterns observed in the dataset, indicating under which conditions a man's sperm is more likely to be diagnosed as altered. The antecedent conditions specified in the rules capture the factors influencing fertility, and the consequent reflects the ultimate diagnosis based on those conditions.

8.2 II

Firstly, a minimum support threshold of 0.02 is chosen. This means that only associations or item-sets appearing in at least 2% of the transactions will be considered significant, indicating their frequency in the dataset. Next, a confidence level of 1 is selected. A confidence of 1 implies that only rules with 100% confidence will be included in the results. This signifies a strong relationship between the antecedent conditions and the consequent outcome. In other words, the presence of the antecedent conditions ensures the presence of the specified consequent, which, in this case, is a diagnosis of "altered." Additionally, a constraint is applied to the right-hand side of the rules, specifying that only those with "Diagnosis=altered" should be considered. This focuses the analysis on rules that directly lead to an altered diagnosis, providing targeted insights into factors associated with this particular outcome in male fertility. After applying these conditions, the algorithm returns 26 rules. These 26 rules reflect the strongest associations between specific antecedent conditions and the consequent diagnosis of "altered." Each rule encapsulates a pattern or relationship observed in the dataset, shedding light on potential factors influencing the alteration in sperm quality.

8.3 III

Next, we eliminate based on criteria of being a super rule. A rule is considered a super rule if there exists another rule with a larger or equal lift and identical antecedent and consequent conditions. In simpler terms, if one rule entirely encompasses another in terms of conditions and has an equal or greater lift value, the latter is deemed a super rule and can be excluded from the final set of rules. Lift quantifies the strength of a rule by calculating the ratio of the observed support of the rule to the expected support if the antecedent and consequent were independent. A lift value greater than 1 indicates that the rule has a positive influence, suggesting that the occurrence of the antecedent conditions increases the likelihood of the consequent. Conversely, a lift less than 1 implies a negative or no association. By sorting the rules based on lift, we prioritize those with higher lift values, emphasizing stronger and more meaningful associations. After the elimination process, the code identifies and retains five rules that stand out as significant patterns within the dataset. These remaining rules are characterized by strong and non-redundant associations within the dataset, and their higher lift values, highlighting robust associations between specific antecedent conditions and the consequent diagnosis of "altered" in male fertility.