Developing Machine Learning Methods to Identify High Risk Pregnancies

Amelia Young and Alyssa Grogorenz-de Oliveira
University of Exeter

Abstract – **Cardiotocography monitors foetal health during the third trimester of pregnancy. Improving cardiotocography data interpretation using machine learning will improve detection of abnormal foetal health states, enabling early intervention and preventing fatalities. This report utilised XGBoost and Decision Tree classifiers, which have previously performed well on this cardiotocography dataset, to advance existing literature and establish a robust workflow with valid methods for unbalanced data. Findings showed that random under-sampling led to more generalisable classifiers compared with the synthetic minority over-sampling technique (SMOTE). Feature selection also improved classifier performance. However, there was a substantial trade-off between Recall and Precision. Future research should investigate hybrid resampling approaches and aim to improve Precision whilst maintaining high Recall.**

Index Terms – Cardiotocography, Foetal Health, Machine Learning, Classification

Introduction

In 2022, 47% of global mortalities in children under 5 were newborns; leading causes included birth complications such as trauma and asphyxia [1]. The United Nations' Sustainable Development Goal (SDG) 3.2 aims to end preventable deaths of newborns and children under 5 years by 2030 [2]. Achieving this goal requires improvements in foetal health monitoring methods that enable timely interventions and prevent foetal death.

Cardiotocography (CTG) is commonly used to assess foetal health by measuring foetal heart rate (FHR) and uterine contractions. However, CTG interpretation by specialists is labour-intensive and subjective [3]. Both time-efficient and consistent, machine learning can combat intra- and inter-observer variability, provide aid in interpreting uncertain situations, and ensure detection of subtle abnormalities in foetal health. Additionally, machine learning can act as a replacement where no skilled professionals are available, such as in developing countries where child mortality rates are higher [1].

This report advances existing machine learning methods for foetal health classification using a standard CTG dataset [4]. Normal (N), suspect (S), and pathological (P) cases make up 77.9%, 13.8%, and 8.3% of the dataset, respectively, reflecting severe class imbalance. Whilst classification models trained on imbalanced datasets perform well for majority classes, the minority classes have poorer outcomes [5]. Addressing class imbalance aids detection of minority classes and is therefore vital for detecting foetal health abnormalities.

Many previous studies using this dataset fail to address class imbalance effectively, including some which ignore it completely [6], [7]. One paper stratifies the train-test split but fails to balance the training set [3], whilst another resamples the data before the train-test split [8]. The former approach remains subject to the disadvantages of imbalanced data, whilst the latter results in data leakage.

Over-sampling and under-sampling are two methods used for balancing data. Under-sampling involves selecting subsets of the majority class(es), whilst over-sampling generates more instances of the minority class(es); both methods lead to the same number of samples in each class [5]. Under-sampling is rarely used for this dataset. One study uses a derived dataset, only including 300 instances of the normal class [9]. However, information is lost from the excluded normal cases.

A study by Hoodbhoy *et al*. [10] uses the synthetic minority over-sampling technique (SMOTE) to address class imbalance effectively. Whilst this method strengthens class boundaries, it is prone to overfitting as it replicates the minority class [11]. Their two best-performing classifiers are Decision Tree and XGBoost. Expanding these results, this report investigates these two models to compare whether repeated random under-sampling (RUS), which avoids replicating the minority class, can outperform their SMOTE technique.

Additionally, feature selection (FS) can improve the performance of classification algorithms by removing irrelevant features [12], [13]. Therefore, this report also compares models that have been trained with all features to models trained with selected features.

Overall, this report provides a workflow which is suitable for training classifiers on imbalanced data, providing valid and robust results. This method is recommended for future research involving this dataset.

Methods

I. Data

The CTG dataset obtained from the University of California Irvine Machine Learning Repository [4] was generated using the SisPorto 2.0 software [14] and contained data for 2,126 women. There were 21 features and one target variable classifying each foetal health state as N, S, or P (Table I) according to the consensus of three specialists [3]. There was no missing data, and 13 duplicate entries were removed.

TABLE I.
SYMBOLS AND DESCRIPTIONS OF FEATURES IN THE CARDIOTOCOGRAPHY DATASET. SUBSET OF SELECTED FEATURES ARE IN BOLD.

Feature Symbol	Feature Description
BV	Baseline value of foetal heart rate (FHR) in beats per minute
AC	**Number of accelerations per second**
FM	Number of foetal movements per second
UC	Number of uterine contractions per second
LD	Number of light decelerations per second
SD	Number of severe decelerations per second
PD	**Number of prolongued decelerations per second**
ASTV	**Percentage of time with abnormal short-term variability**
MSTV	**Mean value of short-term variability**
ALTV	**Percentage of time with abnormal long-term variability**
MLTV	**Mean value of long-term variability**
HW	FHR histogram width
HMin	FHR histogram minimum
HMax	FHR histogram maximum
NP	Number of FHR histogram peaks
NZ	Number of FHR histogram zeroes
HMo	FHR histogram mode
HMe	**FHR histogram mean**
HMed	FHR histogram median
HV	FHR histogram variance
HT	FHR histogram tendency
NSP	Foetal state code (N = normal, S = suspect, P = pathological)

II. Feature Selection

Similarly to previous studies [3], [8], [9], features were selected using a correlation-based approach. The linear relationship between features was calculated using Pearson's correlation coefficient r [15]. Features strongly correlated with the target variable NSP ($|r| \geq 0.4$) were retained (Figure I). These features were PD, ASTV, and ALTV. HV was excluded due to strong correlation with PD.

The mutual information (MI) score assessed the importance of remaining features in predicting foetal health (1) [16]:

$$MI(U,V) = \sum_{i=1}^{|X|} \sum_{j=1}^{|Y|} \frac{|X_i \cap Y_j|}{N} ln \frac{N|X_i \cap Y_j|}{|X_i||Y_j|} \qquad (1)$$

where $|X|$ and $|Y|$ are the number of samples of features X and Y, X_i and Y_j the i-th and j-th observation, respectively, and N the total number of samples between both features.

A higher MI score indicates a stronger relationship between two features. Unlike r, the MI score also captures non-linear relationships [16]. The MI score of each feature was split by its relationship with N, S, or P cases. Features with a low MI score (< 0.02) were removed. MI scores for S and P cases were prioritised, ensuring that models could reliably detect these foetal health instances. Features removed in this step were UC, SD, LD, NZ, NP, and HT.

Among remaining features, strongly correlated ($|r| \geq 0.4$) pairs were identified, and the feature with a higher overall MI score retained. The final selected features were AC, PD, ASTV, MSTV, ALTV, MLTV, and HMe.

III. Resampling

SMOTE creates synthetic samples by randomly generating minority class instances between a minority sample and one of its k-nearest neighbours [17]. S and P classes were resampled using $k = 5$. RUS selects a random subset of the majority classes N and S without replacement. Each class was resampled independently. For both methods, cross validation enabled repetition of resampling to ensure robustness.

IV. Machine Learning Algorithms

A Decision Tree classifier consists of nodes and branches which form a set of decisions about the class of an observation based on its features. Each split is a Boolean statement about a specific feature, defining how an observation is handled within the tree and ultimately determining its class. The quality of each split is assessed by its impurity (or loss function). Minimising impurity ensures a lower likelihood of misclassification. Hyperparameters determine the depth, complexity, and loss function of a decision tree, which is important to prevent overfitting [18].

XGBoost (extreme gradient boosting) is a computationally efficient tree boosting algorithm. As an ensemble classifier of Decision Trees, it calculates the sum of predictions from each tree, which are learned by minimising a loss function. XGBoost uses a weighted quantile sketch algorithm to find the best aggregate statistic for the split of each consecutive branch. The regularisation parameter within the loss function helps to avoid overfitting, as does feature subsampling, and shrinkage, which reduces the weight of newly added trees [19].

For both algorithms, hyperparameter tuning using scikit-learn's GridSearchCV function [20] exhaustively generated classifiers from a provided hyperparameter

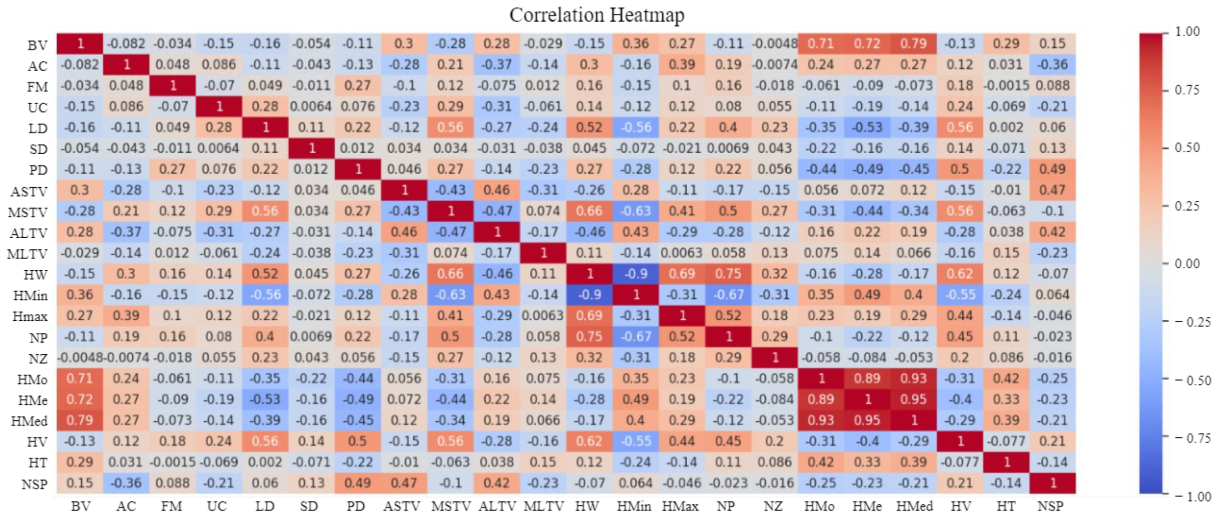

FIGURE I.

CORRELATION HEATMAP OF FEATURES, USING PEARSON'S CORRELATION COEFFICIENT.

grid [21]. The combination of hyperparameters with the highest Weighted Recall (2) was chosen as the best classifier.

Repeated stratified k-fold cross validation with $k = 10$ folds helped to avoid overfitting [22]. For each fold, a different part of the training data was held out for internal evaluation [23]. The algorithm was rerun $n = 100$ times, each with a different split of folds. Stratified sampling reflected the relative class imbalance in each fold.

V. Performance Metrics

The number of True Positives, False Negatives, and False Positives were calculated by considering each class as a binary problem (e.g. "pathological" and "not pathological"). TP_c, FN_c, and FP_c are the number of True Positives, False Negatives, and False Positives of each class c, respectively, and n_c is the number of instances in each class.

The Weighted Recall (2) is the overall Recall weighted by the relative size of each class [24].

$$Weighted\ Recall = \frac{\sum_{c=1}^{3} n_c \cdot TP_c}{\sum_{c=1}^{3} n_c \cdot (TP_c + FN_c)} \qquad (2)$$

Precision measures how well a model correctly predicts the class of a given instance, whereas Recall measures how many relevant instances of a given class the model identifies [25]. $Precision_c$ (3) and $Recall_c$ (4) represent the Precision and Recall of a given class c.

$$Precision_c = \frac{TP_c}{TP_c + FP_c} \qquad (3)$$

$$Recall_c = \frac{TP_c}{TP_c + FN_c} \qquad (4)$$

The Fβ-score is a summary statistic of Precision and Recall (5). The value of β determines the relative weight of Recall over Precision [26]. Since the cost of misidentifying an unhealthy foetal state as healthy (False Negative) is greater than that of misidentifying a healthy foetal state as unhealthy (False Positive) [25], valuing Recall over Precision ensures that more lives are saved. Therefore, $\beta = 2$ was chosen to value Recall twice as much as Precision.

$$F_{\beta_c} = \frac{(1 + \beta^2) \cdot TP_c}{(1 + \beta^2) \cdot TP_c + FP_c + \beta^2 \cdot FN_c} \qquad (5)$$

The Balanced Accuracy $BAcc$ accounts for class imbalance, providing an unbiased accuracy metric of a model (6) [27]:

$$BAcc = \frac{Recall_N + Recall_S + Recall_P}{3} \qquad (6)$$

where $Recall_N$, $Recall_S$, and $Recall_P$ were the Recall scores of N, S, and P classes, respectively.

VI. Workflow

The dataset was analysed in Python, using scikit-learn [20] and imbalanced-learn [28] packages. Data was split into stratified train and test sets with 80% and 20% of the data, respectively. Test data was held out completely and only used for model evaluation. Classification, with and without feature selection, was carried out on the train set (Figure II). The imbalanced-learn pipeline incorporated data standardisation, resampling, hyperparameter tuning, and cross validation into the classification process, ensuring no data leakage into the test set.

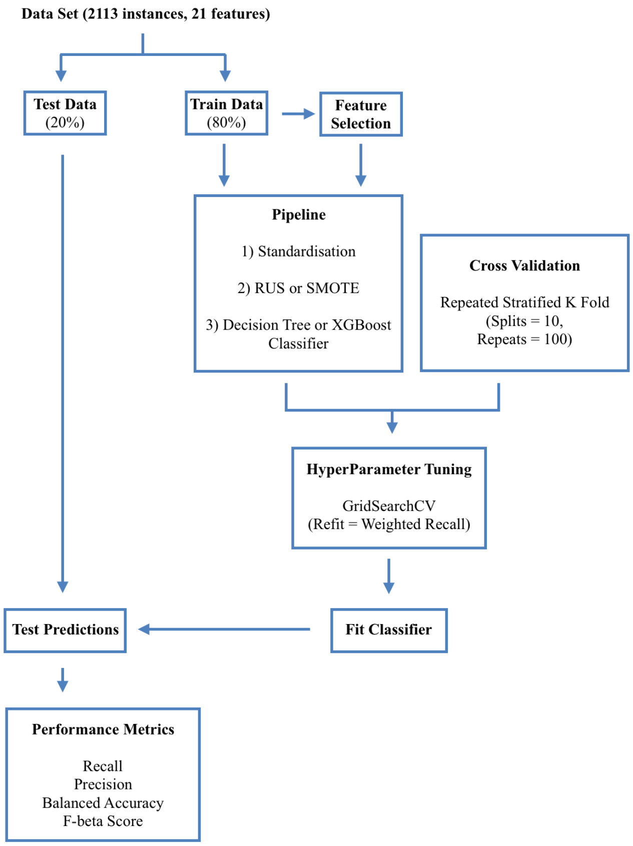

FIGURE II.
MACHINE LEARNING PIPELINE.
RUS= RANDOM UNDER-SAMPLING, SMOTE = SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE.

RESULTS

The performance of eight machine learning classifiers for predicting foetal health were compared. XGBoost and Decision Tree models were trained with the following combinations of data resampling and FS: SMOTE, SMOTE + FS, RUS, RUS + FS.

Overall, XGBoost models had higher Balanced Accuracy than Decision Tree models, and classifiers using RUS had higher Balanced Accuracy for test data than classifiers using SMOTE (Figure IIIa). Conversely, Balanced Accuracy of the train data was higher for classifiers rebalanced with SMOTE (Figure IIIb). Whilst FS had little effect on train data, it mostly increased Balanced Accuracy for test data. The models with the highest Balanced Accuracies for the test data were XGBoost + RUS, XGBoost + RUS + FS, and XGBoost + SMOTE + FS.

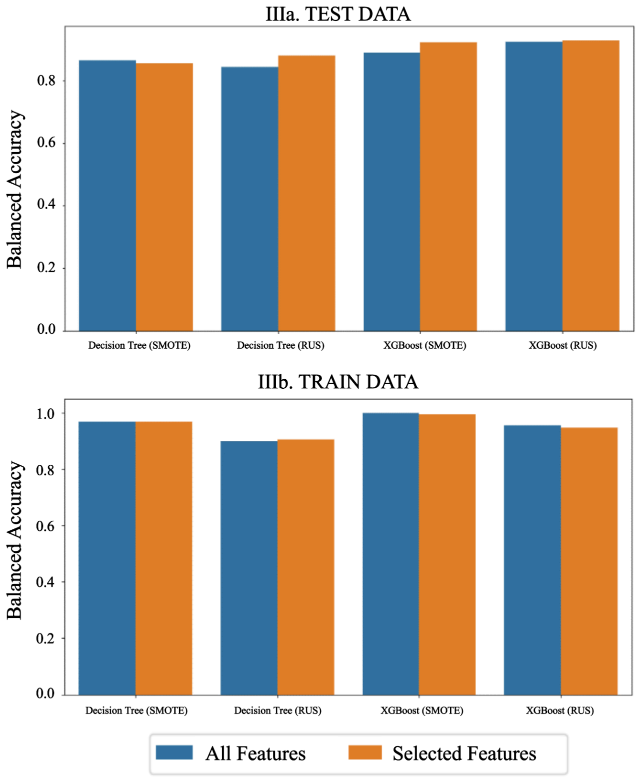

FIGURE III.
BALANCED ACCURACY SCORES FOR TEST (A) AND TRAIN (B) DATA.
RUS = RANDOM UNDER-SAMPLING, SMOTE = SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE.

The Recall of all XGBoost models was equal to or greater than the Recall of their Decision Tree counterparts. Compared to SMOTE, classifiers using RUS had higher $Recall_S$ and $Recall_P$, but lower $Recall_N$. The classifiers with highest Balanced Accuracies also had the joint highest $Recall_P$ $(= 0.94)$. XGBoost + RUS + FS had the highest $Recall_S$ $(= 0.97)$ (Table II).

TABLE II.
COMPARISON OF RECALL FOR EACH CLASS AND CLASSIFIER.
FS = FEATURE SELECTION, SMOTE = SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE, RUS = RANDOM UNDER-SAMPLING.
N = NORMAL, S = SUSPECT, P = PATHOLOGICAL.

Model	Resampling + FS	Recall		
		N	S	P
Decision Tree	SMOTE	0.95	0.72	0.91
	SMOTE + FS	0.96	0.69	0.91
	RUS	0.83	0.84	0.86
	RUS + FS	0.85	0.84	0.94
XGBoost	SMOTE	0.98	0.78	0.91
	SMOTE + FS	0.96	0.86	0.94
	RUS	0.88	0.95	0.94
	RUS + FS	0.87	0.97	0.94

Notably, classifiers with the highest $Recall_P$and $Recall_S$ had low Precision, and vice versa. Classifiers using RUS had lower $Precision_P$ and $Precision_S$ and higher $Recall_N$, compared to classifiers using SMOTE. Nevertheless, XGBoost models still had equivalent or higher Precision than their Decision Tree counterparts (Table III).

TABLE III.
COMPARISON OF PRECISION FOR EACH CLASS AND CLASSIFIER.
FS = FEATURE SELECTION, SMOTE = SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE, RUS = RANDOM UNDER-SAMPLING.
N = NORMAL, S = SUSPECT, P = PATHOLOGICAL.

Model	Resampling + FS	Precision		
		N	S	P
Decision Tree	SMOTE	0.95	0.75	0.91
	SMOTE + FS	0.95	0.75	0.91
	RUS	0.96	0.58	0.58
	RUS + FS	0.97	0.53	0.80
XGBoost	SMOTE	0.96	0.87	0.91
	SMOTE + FS	0.98	0.78	0.97
	RUS	1.00	0.61	0.80
	RUS + FS	1.00	0.62	0.77

The Fβ-score ($\beta = 2$) helped gain an overview of classifier performance. XGBoost + SMOTE + FS had the highest ${F_\beta}_P (= 0.95)$, whilst XGBoost + RUS + FS had the highest ${F_\beta}_S$ (= 0.87) (Table IV). However, both classifiers still had high Fβ-scores for each class. XGBoost + SMOTE + FS had higher ${F_\beta}_N$ compared to XGBoost + RUS + FS.

TABLE IV.
COMPARISON OF Fβ FOR EACH CLASS AND CLASSIFIER.
FS = FEATURE SELECTION, SMOTE = SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE, RUS = RANDOM UNDER-SAMPLING.
N = NORMAL, S = SUSPECT, P = PATHOLOGICAL.

Model	Resampling + FS	Fβ (β = 2)		
		N	S	P
Decision Tree	SMOTE	0.95	0.73	0.91
	SMOTE + FS	0.96	0.70	0.91
	RUS	0.85	0.77	0.78
	RUS + FS	0.87	0.76	0.91
XGBoost	SMOTE	0.98	0.79	0.91
	SMOTE + FS	0.97	0.84	0.95
	RUS	0.90	0.85	0.91
	RUS + FS	0.89	0.87	0.90

DISCUSSION

I. Discussion of Results

This report compared the performance of XGBoost and Decision Tree classifiers, trained with SMOTE- or RUS-resampled data, and with or without FS, to predict foetal health using the CTG dataset.

Overall, XGBoost classifiers performed consistently better than Decision Tree classifiers, probably because the former is an ensemble of Decision Trees. Additionally, SMOTE classifiers had lower Balanced Accuracy than RUS classifiers for test data, but higher Balanced Accuracy than RUS classifiers for train data, suggesting overfitting of classifiers trained using SMOTE-resampled data. This indicates that RUS leads to more generalisable classifiers. Similarly, FS had little effect on the Balanced Accuracy for train data but mostly improved Balanced Accuracy for test data. FS may also allow classifiers to be more generalisable since fewer features reduce the likelihood of overfitting [13].

Classifiers trained using SMOTE (but not FS) were used as a benchmark with which to compare other classifiers, since the rationale for this report was based on findings by Hoodbhoy *et al.* [10]. Notably, the Balanced Accuracy scores for Decision Tree + SMOTE and XGBoost + SMOTE in this report were lower than the Accuracy scores reported by Hoodbhoy *et al.* Since they did not use Balanced Accuracy, this discrepancy is likely caused by class imbalance inflating their reported Accuracy. Precision and Recall for the Decision Tree + SMOTE and XGBoost + SMOTE classifiers were largely similar to their results. Both FS and RUS improved $Recall_P$ for XGBoost classifiers.

This report suggests that XGBoost + SMOTE + FS, XGBoost + RUS, and XGBoost + RUS + FS are the best classifiers based on their high Balanced Accuracy and $Recall_P$. However, there are noticeable trade-offs between Recall and Precision, and between scores for N versus S and P. Compared to SMOTE classifiers, RUS classifiers have higher $Recall_S$ and lower $Precision_S$ and $Precision_P$, but lower $Recall_N$ and higher $Precision_N$. This reflects the different methods of resampling.

Whilst Recall was prioritised, low precision may still be problematic since misidentification of N leads to unnecessary and costly medical intervention. Therefore, the Fβ-score provided a more useful indication of model performance. In terms of the Fβ-score, no classifier outperformed all others; XGBoost + SMOTE + FS or XGBoost + RUS + FS are suggested as the best choices of classifier. The former had the highest ${F_\beta}_P$, but was less generalisable for unseen data, and training time was longer. The latter had the highest Balanced Accuracy score, but ${F_\beta}_N$ and ${F_\beta}_P$ were lower compared to other XGBoost models. Additionally, it had high ${F_\beta}_S$, RUS was more generalisable, and training time was shorter, suggesting that it is the optimal classifier. Both models incorporate FS, suggesting that FS improves classifier performance. FS also reduced classifier training time.

The five most important features reported by Hoodbhoy *et al.* [10] were ASTV, ALTV, AC, MSTV, and UC. Features selected in this report included all of these except UC and were therefore mostly consistent with their results.

II. Comparison to Wider Literature

Sahin and Subasi [7] removed the S class, allowing for binary classification of N and P. Direct comparison with this paper was not possible due to differences in methodology and the authors neglecting class balance; additionally, they did not investigate Decision Tree or XGBoost models. However, their classifier with highest Recall was a Random Forest model, consistent with the finding that an ensemble classifier performs best.

Salini *et al.* [3] trained several classifiers on the CTG dataset, including a Decision Tree model. Their methods of over-sampling and feature selection were unclear. Additionally, data standardisation prior to the train-test split led to data leakage. The Decision Tree + RUS + FS model in this report achieved lower Precision scores than their Decision Tree model, but higher $Recall_S$ and $Recall_P$, and comparable $Recall_N$. Again, this indicates that under-sampling outperforms an over-sampling approach.

Chinnasamy *et al.* [6] trained an artificial neural network, which achieved higher $Precision_N$, $Precision_P$, $Recall_N$, and $Recall_P$ than the models presented here. However, $Precision_S$ and $Recall_S$ were low. During trials, their neural network occasionally failed to identify any S cases at all, potentially since class imbalance was neglected. Therefore, the ability to classify S more accurately is a strength of this report.

Nandipati and XinYing [9] utilised correlation-based feature selection. Their final set of features were BV, AC, FM, UC, LD, SD, PD, ASTV, ALTV, MLTV, HMin, NZ, and HT, which includes five of the seven features selected in this report. Their classifiers performed better on a smaller number of features, consistent with the findings here. However, direct comparisons cannot be made since they calculated overall Precision and Recall scores for each classifier, without specifying the averaging method for this multiclass problem.

Mehbodniya *et al.* [29] used filter-based feature selection. Their final set of features were AC, PD, ASTV, ALTV, HMo, HMe, HMed, and HV, containing four of the seven features used here. None of their classifiers were Decision Tree or XGBoost classifiers. However, their ensemble method Random Forest outperformed their other classifiers, consistent with the better performance of XGBoost in this report. Again, direct comparisons are not possible because the averaging method for overall Precision and Recall was not specified.

Rahmayanti *et al.* [8] considered removing strongly correlated features using Variable Inflation Factors, but ultimately achieved their best results with the full dataset. Among other classifiers, they used XGBoost, optimising the hyperparameters via grid search cross validation like in this report. Their Precision for all classes and $Recall_S$ and $Recall_P$, were higher than results for XGBoost + SMOTE in this report. However, the authors scaled and over-sampled the data prior to the train-test split, leading to data leakage which inflated their results.

Overall, this report improves many methods and results in the literature for the CTG dataset. Given the success of Random Forest in previous studies, investigating the performance of a Random Forest in combination with FS and RUS may yield a better performing classifier than those presented here. Additionally, the performance of the XGBoost classifier may be further improved by using a combination of over- and under-sampling. Such a hybrid rebalancing method can help reduce noise in the dataset, allowing for an improvement of classifier performance [30], [31].

Conclusion

Classification of foetal health states using cardiotocography data can be enhanced through utilisation of machine learning algorithms. XGBoost is a particularly promising classifier. This report provides a robust approach for classifying imbalanced datasets. Repeated random under-sampling and feature selection both lead to high accuracy of the classifier, help avoid the risk of overfitting, and reduce training time. Overall, this report expands the previous literature and presents a model capable of high-quality classification of foetal health states. Further improvements to algorithms using the robust approach in this report will aid reduction of global foetal mortality rates.

Reproducibility

Code has been shared along with relevant files via a OneDrive folder named “MTHM 602 Code (Milly and Alyssa)”. It can be accessed via the following URL: https://bit.ly/44mwjyY.

References

[1] World Health Organization, ‘Newborn Mortality’, Newborn Mortality. Accessed: Mar. 23, 2024. [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/newborn-mortality

[2] World Health Organization, ‘SDG Target 3.2: End preventable deaths of newborns and children under 5 years of age’, The Global Health Observatory. Accessed: Mar. 23, 2024. [Online]. Available: https://www.who.int/data/gho/data/themes/topics/sdg-target-3_2-newborn-and-child-mortality

[3] Y. Salini, S. N. Mohanty, J. V. N. Ramesh, M. Yang, and M. M. V. Chalapathi, ‘Cardiotocography Data Analysis for Fetal Health

Classification Using Machine Learning Models', *IEEE Access*, vol. 12, pp. 26005–26022, 2024, doi: 10.1109/ACCESS.2024.3364755.

[4] D. Ayres-de-Campos and J. Bernardes, 'Cardiotocography'. UC Irvine Machine Learning Repository, 2010. doi: 10.24432/C51S4N.

[5] M. S. Shelke, P. R. Deshmukh, and V. K. Shandilya, 'A Review on Imbalanced Data Handling Using Undersampling and Oversampling Technique', *IJRTER*, vol. 3, no. 4, pp. 444–449, May 2017, doi: 10.23883/IJRTER.2017.3168.0UWXM.

[6] S. Chinnasamy, M. Chitradevi, and G. Geetharamani, 'Classification of Cardiotocogram Data using Neural Network based Machine Learning Technique', *International Journal of Computer Applications (0975 – 888)*, vol. 47, no. 14, Jun. 2012.

[7] H. Sahin and A. Subasi, 'Classification of the cardiotocogram data for anticipation of fetal risks using machine learning techniques', *Applied Soft Computing*, vol. 33, pp. 231–238, Aug. 2015, doi: 10.1016/j.asoc.2015.04.038.

[8] N. Rahmayanti, H. Pradani, M. Pahlawan, and R. Vinarti, 'Comparison of machine learning algorithms to classify fetal health using cardiotocogram data', *Procedia Computer Science*, vol. 197, pp. 162–171, 2022, doi: 10.1016/j.procs.2021.12.130.

[9] S. C. R. Nandipati and C. XinYing, 'Classification and Feature Selection Approaches for Cardiotocography by Machine Learning Techniques', *JTEC*, vol. 12, no. 1, pp. 7–14, Mar. 2020.

[10] Z. Hoodbhoy, M. Noman, A. Shafique, A. Nasim, D. Chowdhury, and B. Hasan, 'Use of machine learning algorithms for prediction of fetal risk using cardiotocographic data', *Int J App Basic Med Res*, vol. 9, no. 4, p. 226, 2019, doi: 10.4103/ijabmr.IJABMR_370_18.

[11] T. Wongvorachan, S. He, and O. Bulut, 'A Comparison of Undersampling, Oversampling, and SMOTE Methods for Dealing with Imbalanced Classification in Educational Data Mining', *Information*, vol. 14, no. 1, p. 54, Jan. 2023, doi: 10.3390/info14010054.

[12] M. Dash and P. W. Koot, 'Feature Selection for Clustering', in *Encyclopedia of Database Systems*, L. Liu and M. T. Özsu, Eds., Boston, MA: Springer US, 2009, pp. 1119–1125. doi: 10.1007/978-0-387-39940-9_613.

[13] X. Ying, 'An Overview of Overfitting and its Solutions', *J. Phys.: Conf. Ser.*, vol. 1168, p. 022022, Feb. 2019, doi: 10.1088/1742-6596/1168/2/022022.

[14] D. Ayres-de-Campos, J. Bernardes, A. Garrido, J. Marques-de-Sá, and L. Pereira-Leite, 'Sisporto 2.0: A program for automated analysis of cardiotocograms', *J. Matern. Fetal Med.*, vol. 9, no. 5, pp. 311–318, Sep. 2000, doi: 10.1002/1520-6661(200009/10)9:5<311::AID-MFM12>3.0.CO;2-9.

[15] A. Bravais, *Analyse mathématique sur les probabilités des erreurs de situation d'un point.* Impr. Royale, 1844.

[16] scikit-learn developers, 'Mutual Information Score', scikit-learn. Accessed: Apr. 23, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mutual_info_score.html

[17] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, 'SMOTE: Synthetic Minority Over-sampling Technique', *Journal of Artifical Intelligence Research*, vol. 16, pp. 321–357, Jun. 2002.

[18] scikit-learn developers, 'Decision Trees', Scikit-learn. Accessed: Apr. 26, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/tree.html

[19] T. Chen and C. Guestrin, 'XGBoost: A Scalable Tree Boosting System', in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Francisco California USA: ACM, Aug. 2016, pp. 785–794. doi: 10.1145/2939672.2939785.

[20] F. Pedregosa *et al.*, 'Scikit-learn: Machine Learning in Python', *Journal of Machine Learning Research*, vol. 12, no. 85, pp. 2825–2830, Oct. 2011.

[21] scikit-learn developers, 'Tuning the hyper-parameters of an estimator', scikit-learn. Accessed: Apr. 24, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/grid_search.html

[22] scikit-learn developers, 'Cross Validation', Scikit-learn. Accessed: Apr. 24, 2024. [Online]. Available: https://scikit-learn.org/stable/modules/cross_validation.html

[23] R. Kohavi, 'A study of cross-validation and bootstrap for accuracy estimation and model selection', *IJCAI*, vol. 14, no. 2, pp. 1137–1145, 1995.

[24] scikit-learn developers, 'Metrics and scoring: quantifying the quality of predictions', Scikit-learn. Accessed: Apr. 24, 2023. [Online]. Available: https://scikit-learn.org/stable/modules/model_evaluation.html

[25] D. M. W. Powers, 'Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation'. arXiv, Oct. 10, 2020. Accessed: Apr. 24, 2024. [Online]. Available: http://arxiv.org/abs/2010.16061

[26] C. J. Van Rijsbergen, *Information Retrieval*, 2nd ed. Butterworth-Heinemann, 1979.

[27] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, 'The Balanced Accuracy and Its Posterior Distribution', in *2010 20th International Conference on Pattern Recognition*, Istanbul, Turkey: IEEE, Aug. 2010, pp. 3121–3124. doi: 10.1109/ICPR.2010.764.

[28] G. Lemaître, F. Nogueira, and C. K. Aridas, 'Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning', *JMLR*, vol. 18, no. 17, pp. 1–5, Jan. 2017.

[29] A. Mehbodniya *et al.*, 'Fetal health classification from cardiotocographic data using machine learning', *Expert Systems*, vol. 39, no. 6, p. e12899, Jul. 2022, doi: 10.1111/exsy.12899.

[30] N. Junsomboon and T. Phienthrakul, 'Combining Over-Sampling and Under-Sampling Techniques for Imbalance Dataset', in *Proceedings of the 9th International Conference on Machine Learning and Computing*, Singapore Singapore: ACM, Feb. 2017, pp. 243–247. doi: 10.1145/3055635.3056643.

[31] C. Lin, C.-F. Tsai, and W.-C. Lin, 'Towards hybrid over- and under-sampling combination methods for class imbalanced datasets: an experimental study', *Artif Intell Rev*, vol. 56, no. 2, pp. 845–863, Feb. 2023, doi: 10.1007/s10462-022-10186-5.

AUTHOR CONTRIBUTIONS

Alyssa Grogorenz-de Oliveira. Joint writing of the Introduction, Results, Discussion, and Conclusion. Training of XGBoost models and conduction of feature selection. Writing of the following Methods sections: Data, Feature Selection, Machine Learning Algorithms (XGBoost), and Performance Metrics.

Amelia Young. Joint writing of the Introduction, Results, Discussion, and Conclusion. Training of Decision Tree models. Writing of Abstract and the following Methods sections: Resampling, Machine Learning Algorithms (Decision Tree), and Workflow.