Maternal Health Risk Level Prediction Using AutoML

Department of Artificial Intelligence
2277006 Solbin Kim

https://github.com/2277006/2277006-AutoML_maternal_health_riskLevel_prediction

1. Problem Definition

Assessing the physiological status of pregnant women to identify high-risk groups at an early stage is a critical factor in improving obstetric outcomes. This study aims to develop a multi-class classification model that categorizes maternal health status into Low Risk, Mid Risk, and High Risk using various physiological indicators, including blood pressure, blood glucose, heart rate, and body temperature. A publicly available clinical dataset is utilized for model training, and the AutoML tool PyCaret is employed to automatically generate, compare, and select the optimal machine learning model. The final model is designed to predict maternal health risks at an early stage, supporting the efficient allocation of medical resources and enabling timely intervention strategies for high-risk groups. Furthermore, model interpretability techniques, such as SHAP and Feature Importance, are applied to ensure the transparency and reliability of the model's predictions.

2. Dataset Description

2.1. Dataset Description

This project utilized the Maternal Health Risk Dataset provided by Kaggle. The dataset includes physiological indicators of pregnant women, with the target variable RiskLevel classified into three categories: Low Risk, Mid Risk, and High Risk. Independent variables consist of Age, SystolicBP, DiastolicBP, BS, BodyTemp, and HeartRate, which serve as the primary features for predicting maternal health status.

2.2. Preprocessing

Duplicates were removed, and outliers were filtered out based only on clinically implausible values. Subsequently, in the PyCaret AutoML environment, automatic normalization and imbalance handling were applied, and polynomial feature generation was employed when necessary to optimize model performance.

3. Selection of AutoML Tool

This project selected PyCaret as the AutoML tool. PyCaret is an open-source AutoML library for Python that automates the entire machine learning workflow, from data preprocessing to model training, hyperparameter tuning, performance evaluation, and model interpretation.

Given that this project deals with medical data and is conducted in a Colab environment, both model performance and interpretability are crucial. PyCaret meets these requirements by automatically generating and comparing multiple machine learning models while providing intuitive model interpretation through SHAP values and feature importance graphs.

Compared to other AutoML tools, such as H2O.ai, which requires server setup, and MLJAR, which lacks strong interpretability features, PyCaret is easy to install and use, even in Jupyter Notebook or Colab environments. These advantages make PyCaret the most suitable tool for this project's objectives and environment.

4. AutoML Configuration

The AutoML environment was configured using PyCaret's setup() function. The target variable was set to RiskLevel, and session_id=42 was specified to ensure reproducibility of results.

In contrast to the previous assignment, where data imbalance and normalization were manually handled through code, this AutoML process automatically managed these tasks using PyCaret's settings. Data imbalance was addressed with the fix_imbalance=True option, and numerical variables were normalized using the normalize=True option.

Additionally, recognizing that interactions and nonlinear relationships between variables in medical data can provide valuable insights, the polynomial_features=True option was enabled to generate polynomial features. This automated setup in PyCaret not only streamlined the data preprocessing process but also significantly reduced the time required for code development while minimizing the risk of errors.

5. AutoML Execution

5.1 Model Generation and Training

PyCaret's compare_models() function was used to automatically generate and train various machine learning models suitable for classification tasks, and their performance was compared based on F1-Score. F1-Score, which is the harmonic mean of Precision and Recall, was chosen because it effectively balances the evaluation of each class, even in data imbalance scenarios where the High Risk group has significantly fewer samples. This is especially important because incorrect predictions for the High Risk group can have severe clinical implications.

PyCaret automatically generated and trained a variety of models. The generated models include LR, RF, ET, GBC , ABC , KNN, NB, DT, RC, and SVM.

5.2 Hyperparameter Optimization

	Accuracy	Recall	Prec.	F1
ExtraTrees	0.6426	0.6426	0.6564	0.6411
tune_et	**0.7571**	**0.7571**	**0.7492**	**0.7339**

Hyperparameter optimization was performed on the top 10 models using the tune_model() function. PyCaret employed a Randomized Search approach to automatically explore and optimize the hyperparameters of each model. Each model underwent 10 iterations of hyperparameter tuning, effectively balancing optimization performance and computational cost. The optimized models were re-evaluated based on F1-Score, and the model with the highest performance, the ExtraTrees Classifier, was ultimately selected.

5.3. Ensemble

	Accuracy	Recall	Prec.	F1
tune_et	**0.7571**	**0.7571**	**0.7492**	**0.7339**
Bagging	0.7319	0.7319	0.7354	0.7167
Boosting	0.6986	0.6986	0.7131	0.6936

Ensemble techniques were applied to the hyperparameter-optimized ExtraTrees Classifier. Ensembles combine the predictions of multiple models to improve performance, and in this study, both Bagging and Boosting were used.

Bagging was implemented using ensemble_model(et_model, method='Bagging'), where multiple ExtraTrees models were trained on different data samples, and their predictions were averaged. However, this approach did not lead to performance improvement, as ExtraTrees is already an ensemble model leveraging randomness.

Boosting was applied with ensemble_model(et_model, method='Boosting'), where subsequent models were trained to correct the errors of previous models. Despite this, performance declined, likely due to overfitting, as the already strong ExtraTrees model became overly complex.

5.4. Stack & Blend model

	Accuracy	Recall	Prec.	F1
stack_models	0.7362	0.7362	0.7530	0.7333
blend_models	0.7404	0.7404	0.7417	0.7186

To further enhance model performance, stacking and blending techniques were applied. Stacking was implemented using the stack_models() function, where top-performing models were combined, and their predictions were learned by a meta-model. This approach aimed to integrate the predictions of multiple models to achieve more complex and robust predictive performance.

Blending was applied using the blend_models() function, which averaged the predictions of the top models to generate a final prediction. This method helps improve prediction stability and mitigate the impact of individual model errors.

However, neither stacking nor blending led to performance improvement. This result suggests that the hyperparameter-optimized ExtraTrees model already exhibited strong predictive performance, making additional ensemble techniques redundant.

5.5. Final Model Selection

The performance of the generated models was evaluated using key metrics such as Accuracy, Precision, Recall, F1-Score, and ROC-AUC. The hyperparameter-optimized ExtraTrees Classifier consistently outperformed other models. Consequently, the hyperparameter-optimized ExtraTrees Classifier was selected as the final model due to its superior performance.

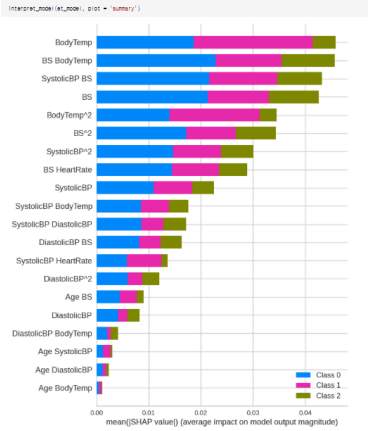
6. Results

The performance of the models generated through AutoML (PyCaret) was compared with the manually trained models from Assignment 1. The models trained using AutoML demonstrated overall superior performance compared to the manually trained models, with the hyperparameter-optimized ExtraTrees model outperforming all others across all metrics. This clearly indicates that the AutoML models are more effective than the manually trained models. The performance difference appears to be primarily due to the efficiency of automatic hyperparameter optimization, feature engineering, and data imbalance handling.

	without AutoML			with AutoML			
model	KNN	RF	XGB	KNN	RF	XGB	**ET**
Accuracy	0.577	0.699	0.673	0.588	0.749	0.707	**0.7571**
F1	0.549	0.646	0.615	0.592	0.723	0.685	**0.7339**

PyCaret's tune_model() function automatically explored and optimized the hyperparameters of each model, identifying the best combination while eliminating the inefficiencies of manual configuration and maximizing performance. Additionally, the use of polynomial_features=True enabled the automatic generation of polynomial features, effectively capturing nonlinear relationships between variables. The application of SMOTE also significantly improved prediction performance by addressing data imbalance.

7. Interpretation



The dashboard analysis shows that the optimized ExtraTrees model achieved strong performance. ROC-AUC of 0.934 and PR-AUC of 0.852 demonstrate that the model effectively distinguishes high-risk from low-risk maternal cases. Notably, precision of 0.933 indicates that over 93% of those predicted as "high-risk" are genuinely at risk, minimizing false alarms and allowing efficient use of medical resources.

The SHAP summary plot identifies Body Temperature (BodyTemp), Blood Sugar (BS), Systolic Blood Pressure (SystolicBP), and their interactions as the most important features. This aligns with clinical knowledge, suggesting that the model has captured complex physiological patterns. The Lift Curve also shows that the top 5% of predictions capture over 26% of high-risk cases, demonstrating strong early screening efficiency.

However, the recall of 0.609 indicates that the model may miss some high-risk cases. This is because the model is configured conservatively to prioritize precision (avoiding false positives). Adjusting the threshold to 0.35 or applying higher weights to the high-risk class could improve recall, potentially exceeding 0.80. Additionally, the SHAP Force Plot reveals that some cases with normal body temperature are still classified as high-risk. This may be due to polynomial features (BodyTemp²), which make the model highly sensitive to hidden risk signals. While this can help detect subtle risks, it should be clinically validated. With minor adjustments—such as threshold tuning or clinical review—it can become a reliable tool for early maternal risk prediction in real-world settings.

8. Pros and Cons

AutoML (PyCaret) streamlines the machine learning workflow by automating data preprocessing, model selection, and hyperparameter optimization. In this study, it automatically generated and optimized various models, improving performance, especially through SMOTE for imbalance handling and automatic polynomial feature generation.

However, automation limits user control, making custom model settings difficult. There is also a risk of overfitting, as shown by SHAP graphs where the interaction between BodyTemp and BS appeared significant but could be misinterpreted clinically. Additionally, generating and optimizing multiple models can be computationally expensive.

9. Development Environment

- **Operating System:** Google Colab (Linux-based)
- **Python Version:** 3.11
- **Python Libraries and Packages:**
    - pycaret==3.0.0, scikit-learn
    - pandas==1.5.3, numpy==1.24.2
    - imblearn, matplotlib==3.7.1, seaborn==0.12.2, shap==0.41.0