Machine learning operations Project

Master in Data Science and Advanced Analytics

NOVA Information Management School
Universidade Nova de Lisboa

Diabetes Classification Problem

Francisco Pontes, 20211583

Isabella Costa, 20240685

Spring Semester 2024-2025

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1. Problem Overview

Diabetes is a chronic condition that affects the ability of the human body to process blood sugar. If left untreated, it can cause long-term complications such as cardiovascular disease, kidney failure, vision problems and nerve damage. Given its growing presence worldwide, early detection and effective treatment are essential.
For this project, we utilize the *Diabetes Health Indicators* Dataset from Kaggle, which has many health and lifestyle metrics associated with diabetes risk. The main objective of this project is to develor a robust end-to-end MLOps pipeline that can predict if an individual has diabetes, is a pre-diabetic, or has no concerns about this topic. In more technical terms, we aim to ensure strong data governance and a scalable model. This approach ensures that the pipeline is accurate, but also interpretable and easy to maintain.

1.2. Success Metrics

The main goal, performance-wise, of our project is to, given the class imbalance and the health-related context, be able to distinguish efficiently the different classes (i.e: avoid classifying a pre-diabetic as a non-diabetic, or vice-versa).

To evaluate the performance of our classification models, a range of metrics was considered. Given the already mentioned imbalance present in the target variable, the Macro F-1 Score was chosen as the primary metric, as it provides a balanced assessment of performance across all classes, regardless of their frequency.

From a system performance standpoint, we aim to achieve a reproductible pipeline, that can detect if there is drift when comparing different batches over time.

Beyond these core performance metrics, we also consider the reliability and overall quality of the solution. This includes ensuring model reproducibility through modular and version-controlled pipelines built with Kedro, implementing regular data drift detection mechanisms, enhancing model transparency and interpretability using SHAP values, and validating data integrity with Great Expectations to ensure that input data consistently adheres to predefined quality standards.

2. PROJECT PLANNING AND METHODOLOGY

2.1. Agile Sprint Breakdown

For the planning methodology we adopted an Agile inspired plan to organize, breaking down into small chunks of work to allow us to deliver work incrementally. The project initially was structured into six sprints. The blocks established were:

- **Sprint 1: Understanding the data and EDA** - exploratory data analysis (EDA) to understand variable distributions, correlations, and class imbalances.
- **Sprint 2: Ingestion and Pipeline Development** – build a ingestion layer using Kedro and integrating Great Expectations and Feature Store with Hopswork for early-stage validation of

raw data. A parallel dummy pipeline development to support isolated testing and speed up experimentation due to the long running times of the offline feature store.

- **Sprint 3: Preprocessing and feature engineering:** preprocessing steps such as missing value handling, outlier removal, attempt of SMOTE resampling. With direct dependency, later engineer features to improve model signal based on domain insights and EDA findings.
- **Sprint 4: Feature Selection & Initial Modeling:** feature selection with RFE and Random Classifier for modeling
- **Sprint 5 – Model Evaluation:** evaluation metrics applied to trained model on test and validation. Implement SHAP for model explainability. Log pipeline steps with MLFlow
- **Sprint 6 – Data Drift**: implement pipeline for data drift detecting with Evidently.

While the initial plan provided a good starting structure and organized plan, later the lack of some steps was noticed and implemented afterwards, specifically model selecting, small hyperparameter tuning for model trained and sampling of data drift dataset.

3. DATA EXPLORATION AND PREPROCESSING

The exploratory data analysis conducted on the dataset revealed several important insights that guided the design of the preprocessing pipeline. Our original dataset contains 253680 rows and 22 columns. The target variable, *Diabetes_012*, exhibited a clear class imbalance [Figure 1– Target Variable Distribution], with most samples falling into the "Not Diabetic" category, followed by "Diabetic" and a small proportion of "Pre Diabetic" cases. This imbalance suggests that particular attention may be required in model selection or evaluation metrics to ensure fair performance across all classes.

Histograms of non-binary variables such as *BMI*, *General Health*, *Mental Health*, *Physical Health*, *Age*, *Education*, and *Income* revealed diverse distributions Figure 2 – Non-binary Features Distributions. Several features, especially *MentHlth* and *PhysHlth*, were highly right skewed, and the presence of extreme values was confirmed using boxplots. BMI showed a limited number of extreme outliers Figure 3 - Non-binary Features Boxplots, which were removed during preprocessing by discarding instances with a value above 80, but only from the training set to avoid information leakage.

Binary features, including *HighBP*, *CholCheck*, *Smoker*, *PhysActivity*, and *DiffWalk*, were mostly balanced, though some variables such as *CholCheck* displayed minimal variance Figure 4 - Binary Features Distributions. Correlation analysis indicated moderate relationships among health indicators, notably between general and physical health, but no problematic multicollinearity was identified.

While missing values were nonexistent in the original dataset, for reproducibility purposes, our preprocessing pipeline explicitly flags missing data by imputing all missing values with the sentinel value -9999. This choice allows us to flag any observations with missing data.

The presence of this flag can be tracked using a custom utility function to verify its propagation through the pipeline.

Standardization was applied to selected continuous features—*BMI*, *MentHlth*, *PhysHlth*, and *Age*—using a StandardScaler, which was fitted on the training set only. This ensured that validation and test data were transformed using the same scaling parameters, thus preserving the integrity of the evaluation process. All binary features were already numerically encoded, so no additional encoding step was necessary.

A correlation analysis was conducted amongst the set of variables, including the target Figure 5 - All features correlation matrix. The correlation matrix for the numerical features showed moderate positive relationships between variables such as General Health and Physical Health, and between Income and Education. These patterns are consistent with the domain context, where poorer general health is often associated with physical limitations, and education tends to correlate with economic status. Importantly, no strong correlations (> 0.8) were identified that would suggest multicollinearity issues, and therefore no dimensionality reduction was applied at this stage.

Overall, our preprocessing strategy combines minimal intervention with a cautious treatment of missingness and outliers, aligning with the goals of having a reproducible model development under MLOps principles.

4. FEATURE ENGINEERING, SELECTION, AND MODELING

4.1. Feature Engineering

We have decided to create more features, based on the base columns. The columns created were:

- HealthIndex
- LifestyleScore
- Mental_Physical_Gap
- AccessIssues
- DiabetesRiskComposite

These features were, as well as the original columns, put into the feature store, which is a centralized repository designed to manage, version, and serve features consistently across training and inference. This ensures reproducibility and supports efficient collaboration between teams. By integrating the new features into the feature store, we facilitate their reuse in future experiments and help maintain a streamlined and standardized pipeline.

Table 1 - Feature Engineering Information gives more details about how these were created, and why.

4.2. Feature Selection

After assessing that there is no multicollinearity issues, feature selection was performed using Recursive Feature Elimination (RFE), a wrapper-based approach that recursively removes less important features based on model performance. A previously trained classification model (stored as a pickle file) was used when available; otherwise, a Random Forest Classifier was initialized using baseline parameters. The method iteratively ranked and eliminated features, retaining only those that contributed most significantly to the predictive performance on the training set. After applying RFE, a

refined subset of features was retained and exported for reproducibility. This step not only reduced dimensionality but also helped mitigate overfitting and improve the generalization capability of the model pipeline.

4.3. Modeling

During the modelling phase, an iterative approach was adopted with a strong focus on reproducibility and traceability, ensured through integration with MLflow.
The choise of model was done using a model selection pipeline that trained and evaluated 4 models in the same conditions. The models were: Random Forest, Gradient Boosting, Logistic Regression and XGBoost. Most of the models are present in the SKlearn library, except for XBoost that requires a package installation.
Training was conducted with or without feature selection based on the configuration provided, and performance metrics such as training and validation accuracy were computed. All runs were logged in MLflow, including model parameters, metrics, artifacts, and model signatures.

Additionally, interpretability was addressed using SHAP (Shapley Additive Explanations) to better understand the contribution of individual features to the model predictions. SHAP summary plots were generated per class when applicable, offering visual insights into feature importance across the model. These plots were saved and logged to MLflow as artifacts. This approach provides a solid foundation for future model iterations and comparisons.

5. QUALITY TESTING AND MONITORING

5.1. Data Unit Tests (Great Expectations)

To ensure the reliability and consistency of the dataset used in this project, a data validation pipeline was developed using Great Expectations. This pipeline constructs an ExpectationSuite tailored to specific feature groups — numerical features, binary indicators, and the target variable. For each column, appropriate constraints were defined, including value ranges for categorical-like variables (e.g., age, income), type enforcement for numerical features (float64), and domain restrictions for binary fields (e.g., values in {0.0, 1.0}).

5.2. Pytest

For continuous integration, we implemented pipeline tests under Kedro test folder, that mainly focused on asserted expected behavior from the nodes functions. The test ensures that upon further changes, the pipelines that were working properly still do so. In total the project has a 64% test coverage.

5.3. Data Drift Detection

Data drift analysis was implemented to assess whether the statistical distribution of the dataset changed between training and inference periods. To simulate this interaction, we sampled the original dataset generating two samples of twenty thousand rows each. The first one was used as reference and the second was our current data. Both datasets were in the same condition, meaning they went through the pipeline up until the same alteration (Feature Selection).

To evaluate the drift, we used the library Evidently and generated an interactive HTML report using the DataDrift Preset metric, saved under data (08_reporting). The report compares the features distributions in reference and current datasets using a combination of statistical test.
The latest run of this report presented features with stable distributions as no drift detected above the threshold (0.5) in a total of 9 features. On the other hand, 3 features presented. This shows that the current implementation of the data drift pipeline could be used to implement trigger to retrain or adjust bound for features as needed.

6. FINAL PIPELINE

The complete flow of the sequence used from ingestion to model evaluation can be seen at Figure 6 - Kedro viz

The final pipeline is structured to follow a modular and reproducible Machine Learning workflow. It begins with the ingestion, going with data validation through unit tests and only after the preprocessing. Feature engineering and feature selection are subsequently done.

For modeling have the model related pipelines, model selection, model train responsible for at the end generating the SHAP plots. Last but not least we have data drift with two sample of data for comparison

The final implementation follows a modular and reproducible Machine Learning workflow, designed to support transparency and scalability. The pipeline begins with data ingestion, followed by data validation using unit tests that ensure schema and quality expectations are met before proceeding to downstream tasks.

7. MODEL RESULTS

As per the latest full pipeline run, Gradient Boost awas the winning model, substituting the champion model at the time and advancing for training and performance assessment. The results were heavily influenced by the unbalanced structure of the data obtaining an F1 Score of 0.37 as we can see on generated inside the report.

In the SHAP summary for classs zero we can see that features such as HealthIndex, HighBP, and DiabetesRiskComposite are the most influential in predicting the "No Diabetes" class Figure 88 - SHAP class 0. For class 2, high values of features such as HealthIndex, HighBP, and DiabetesRiskComposite strongly increase the model’s prediction probability for this class, as indicated by red points predominantly on the right side of the plot. Conversely, lower values of these features (in blue) push the prediction away from class 2 Figure 1010 - SHAP calass 2. This suggests that poor general health and elevated health risks are major drivers of predictions for the high-risk.

The SHAP plot for class 1 indicates that the model has very limited and weak feature influence when predicting this class. The SHAP values are clustered tightly around zero, showing that no feature consistently pushes predictions strongly toward or away from this class Figure 9 9 - SHAP class 1. This behavior is due to this class being severely underrepresented in the data.

The full list of packages used during the conception of this project can be found at the following link.

8. NEXT STEPS

As we reflect on the work done for this project, there are some issues that can still be addressed, and could be useful to look at in the future. We believe that our pipeline is robust and reproducible. We have many different modules of code that make it easy to isolate tasks, and recreate them in the future, given other contexts. However, when looking at the metrics and results, we are not 100% satisfied.

Our Macro F-1 Score was very low when compared with the overall accuracy of our models. This leads us to believe that, for minority classes, the model does not predict well. To tackle this issue, we will prioritize Oversampling as one of the next steps, using techniques like SMOTE to generate random entries of the lower represented classes. We have tried this approach, but it compromised the whole pipeline, and so we left it as pendent work.

Another valuable strategy for improving model performance could be fine-tuning hyperparameters. This approach can significantly enhance results by optimizing the model's behavior, ultimately leading to more accurate and competitive outcomes when comparing different models.

Scability of this solution can be applied on upload data to Hopswork as of right now we only apply that for features upen Great Exepectation validation. A production pipeline would require a separation of the predicted labels from the train model pipeline to an individual one.

9. CONCLUSION

To conclude our report, we will reflect one last time on the work done: A modular, reproducible, and explainable machine learning pipeline was created for diabetes classification. Although the final F1-score did not meet expectations, the project successfully implemented key components such as robust data validation, drift detection, and model explainability. These elements ensured transparency, reliability, and a solid foundation for further improvements. Our pipeline is designed with scalability and adaptability in mind, making it ready for real-world deployment and future enhancements, including hyperparameter tuning and the integration of more advanced models.

APPENDIX

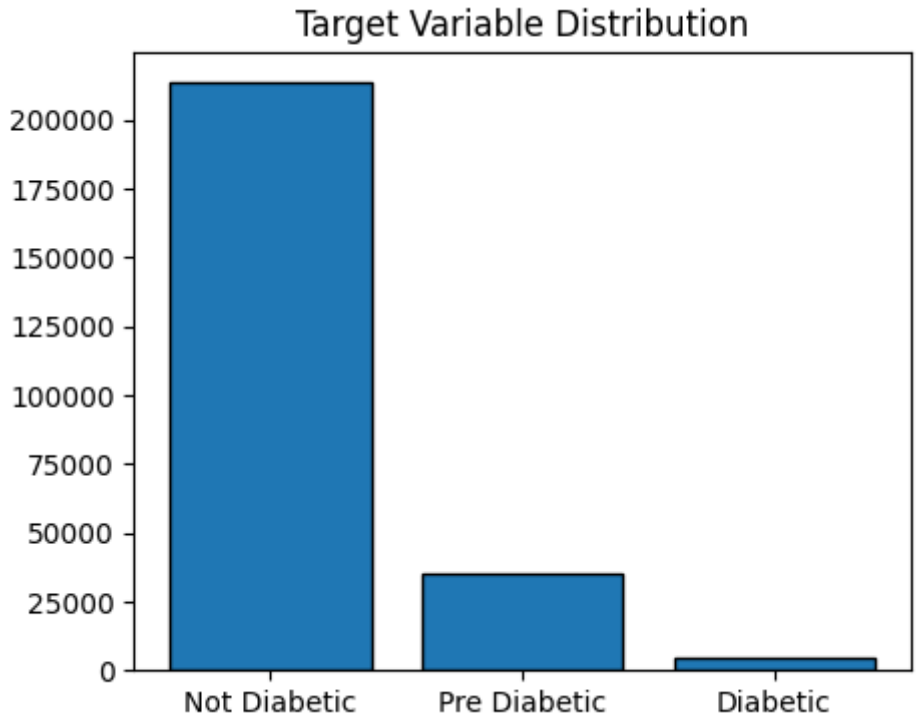

Figure 1– Target Variable Distribution

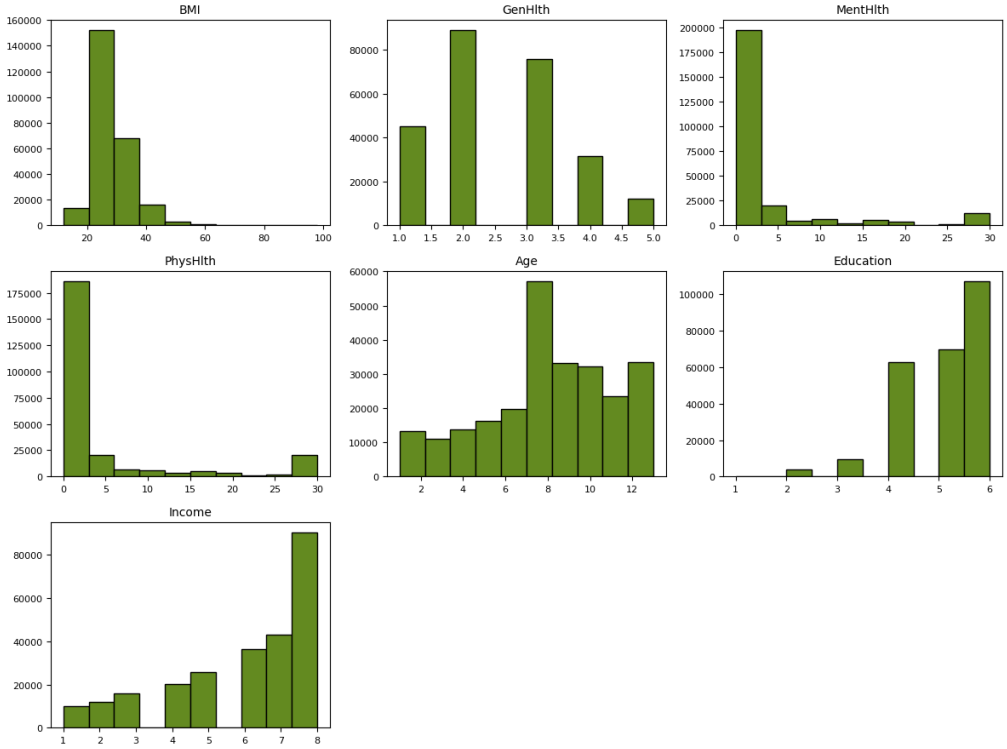

Figure 2 – Non-binary Features Distributions

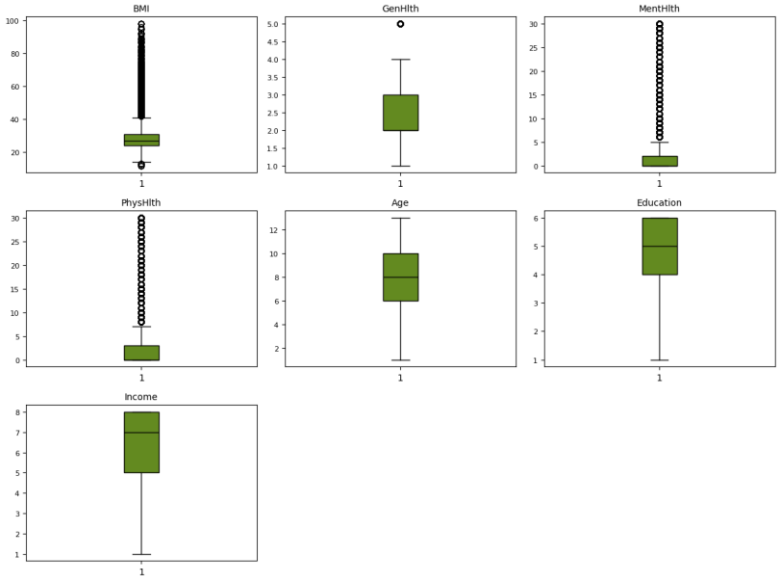

Figure 3 - Non-binary Features Boxplots

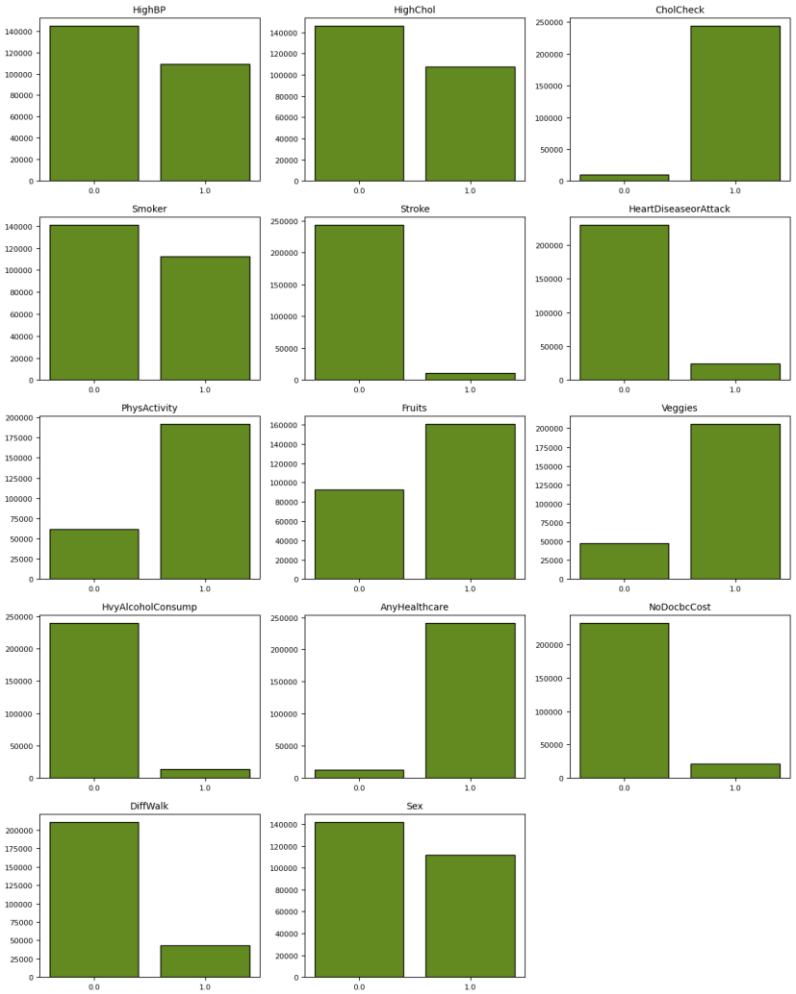

Figure 4 - Binary Features Distributions

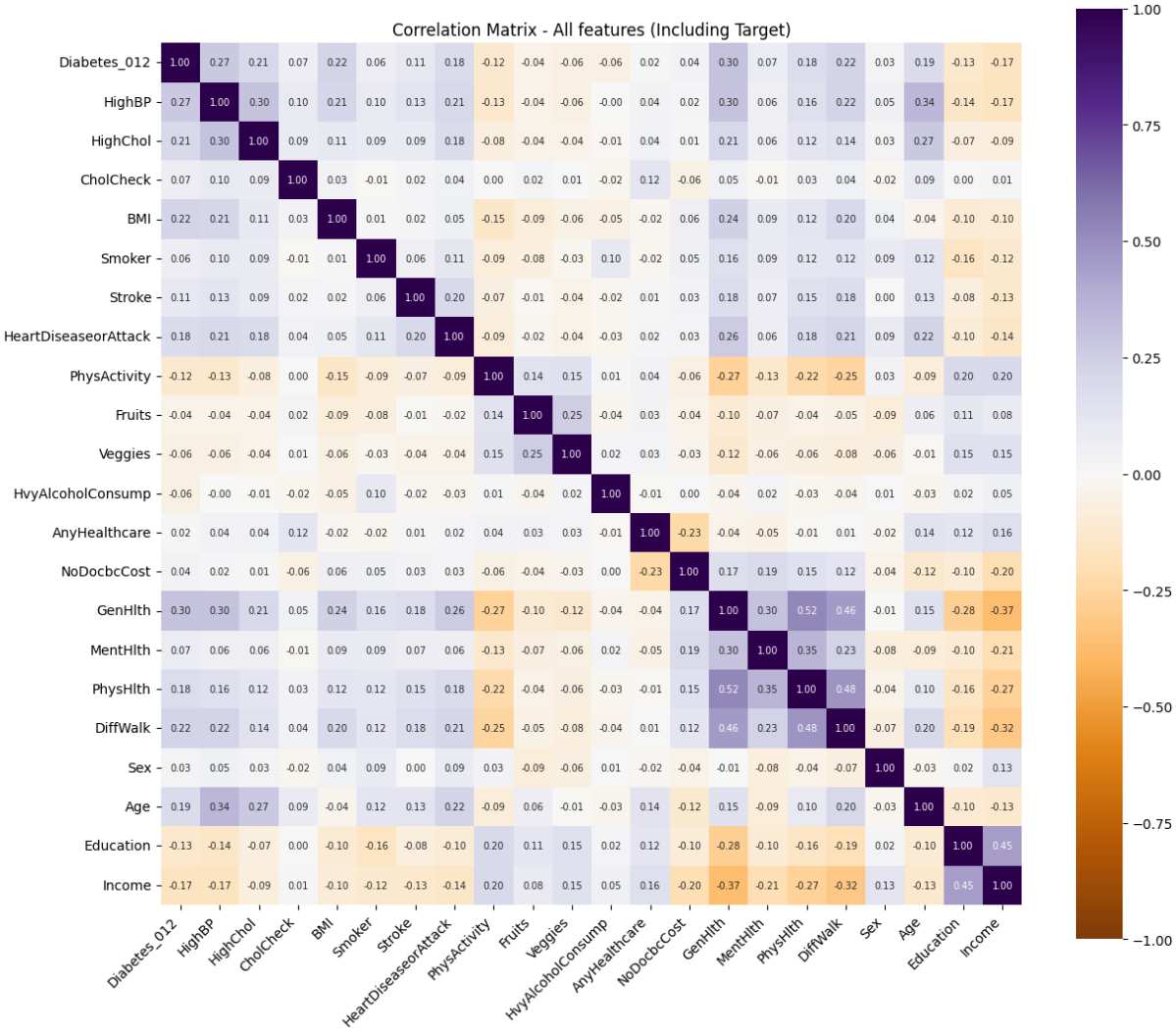

Figure 5 - All features correlation matrix

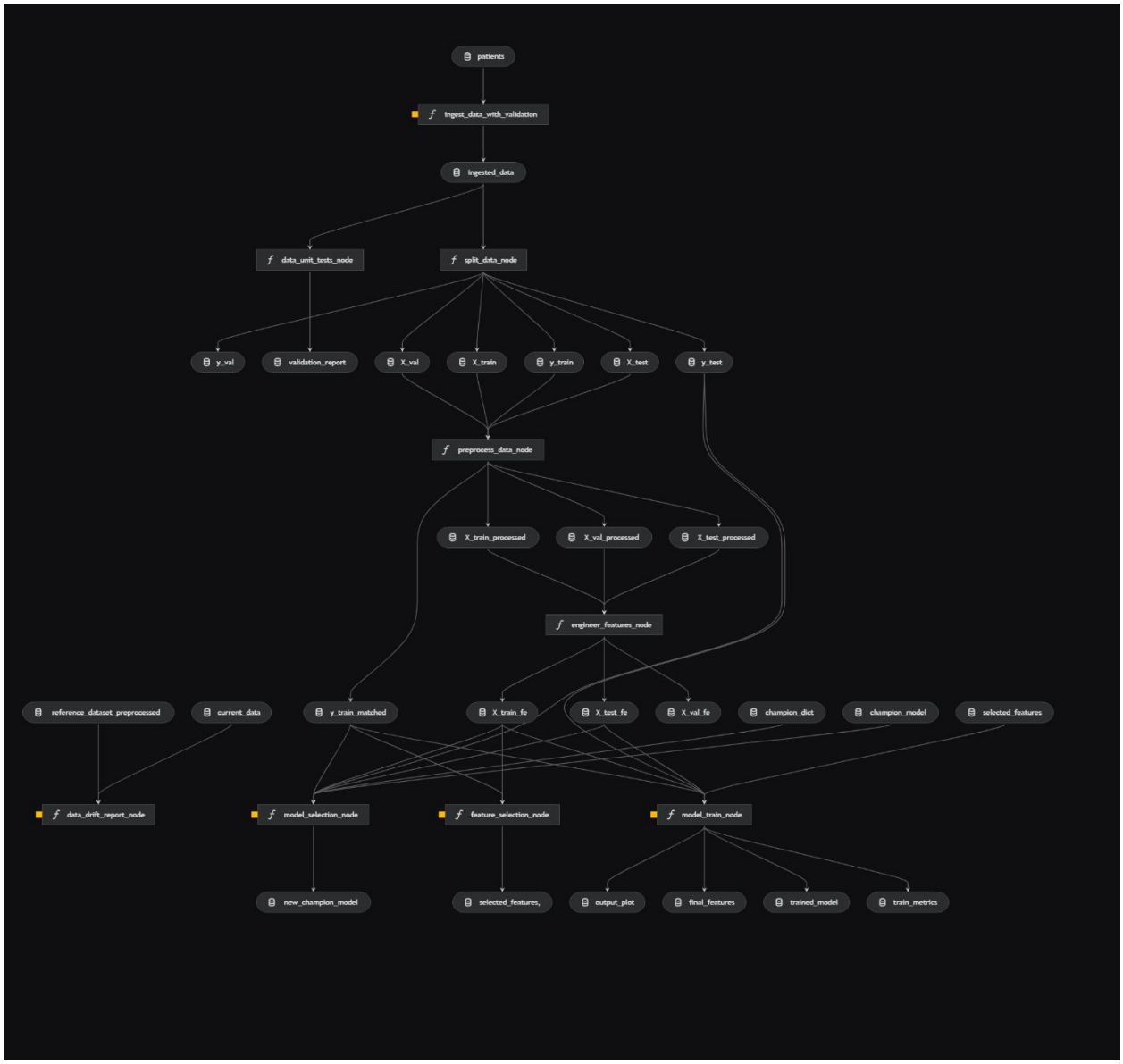

Figure 6 - Kedro viz

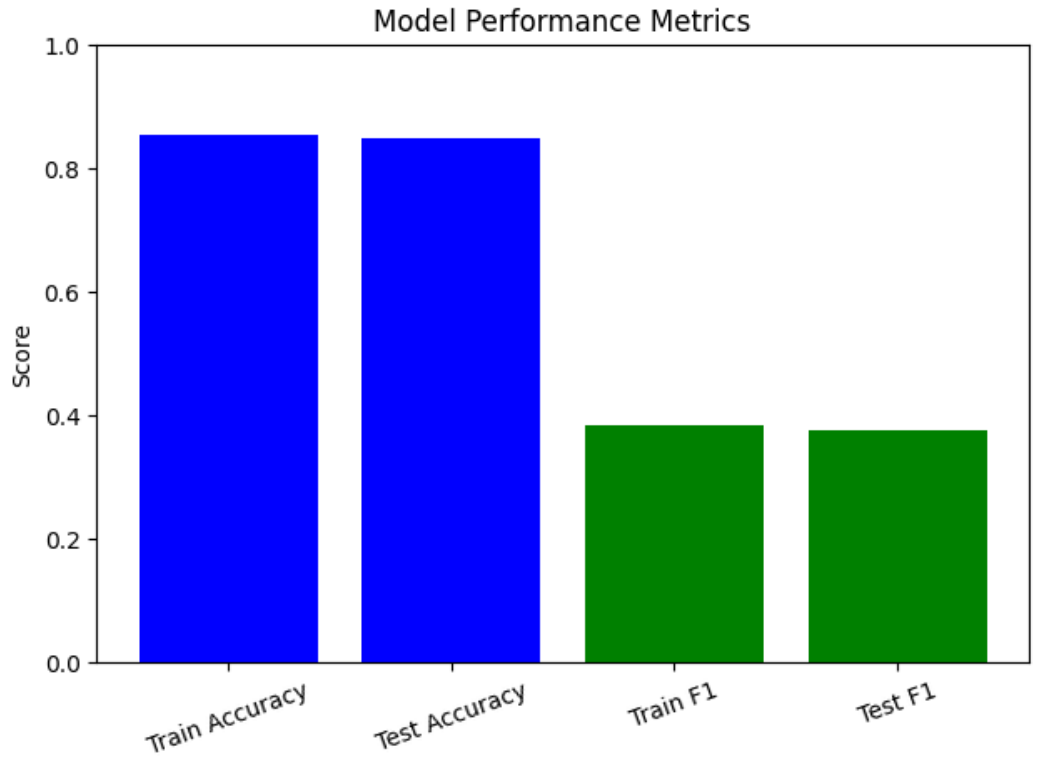

Figure 77 – Final model performance

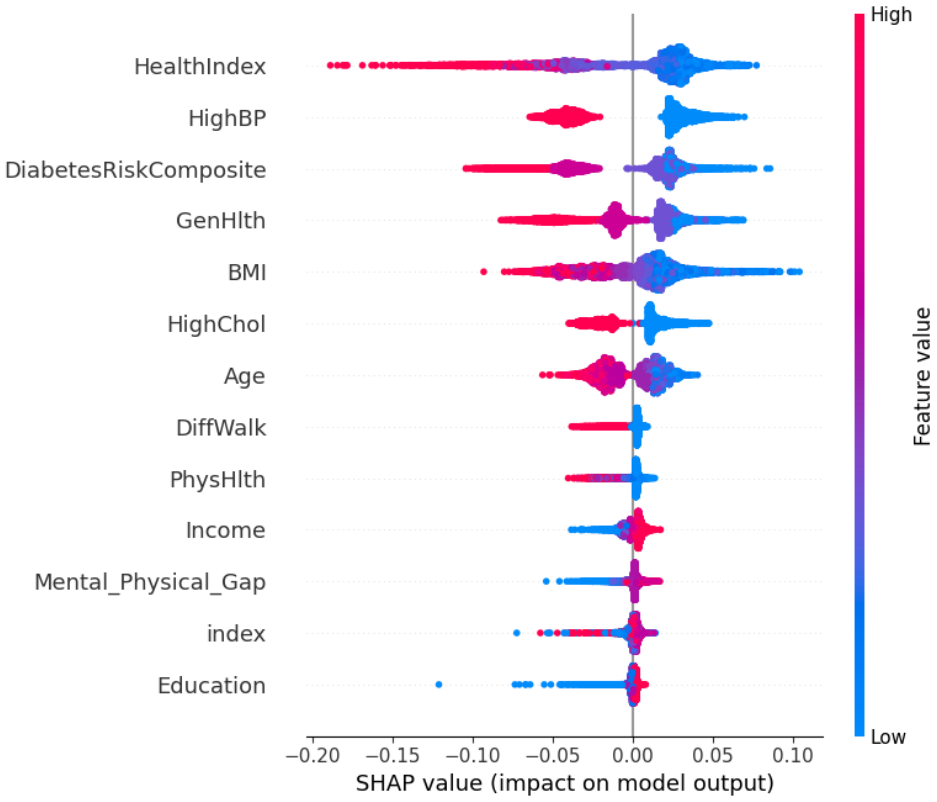

Figure 88 - SHAP class 0

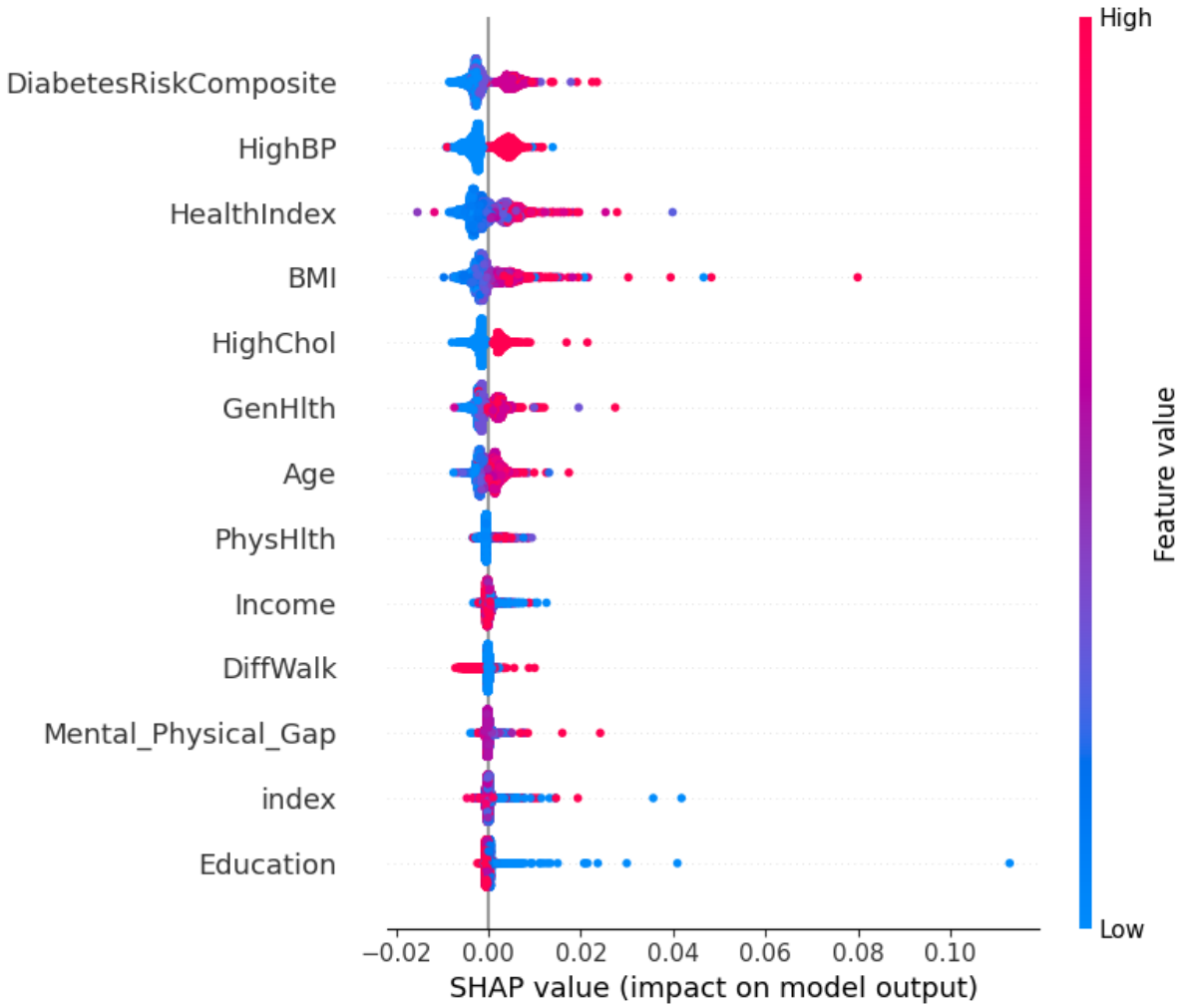

Figure 9 9 - SHAP class 1

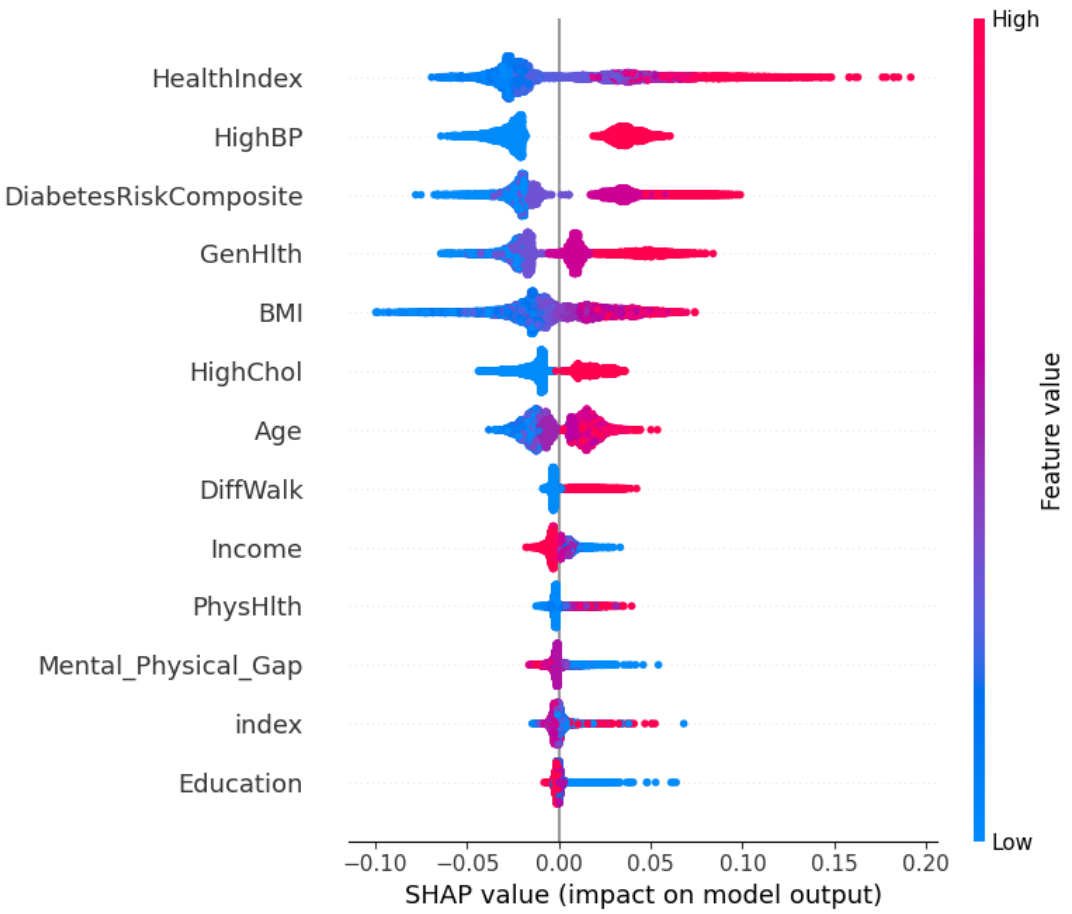

Figure 1010 - SHAP calass 2

Feature Name	Calculation	Description
HealthIndex	(BMI / max(BMI)) + (GenHlth / 5) + (MentHlth / 30) + (PhysHlth / 30) + DiffWalk	Composite health indicator combining general, physical, and mental health, adjusted for BMI and walking difficulty.
LifestyleScore	PhysActivity + Fruits + Veggies - HvyAlcoholConsump - Smoker	Score reflecting lifestyle quality based on physical activity, diet, and harmful habits.
Mental_Physical_Gap	MentHlth - PhysHlth	Difference between reported mental and physical health days, used to assess discrepancy.
AccessIssues	(1 - AnyHealthcare) + NoDocbcCost	Indicator of access to healthcare, capturing both lack of coverage and cost-related barriers.
DiabetesRiskComposite	HighBP + HighChol + (BMI > 30) + (PhysActivity == 0) + (Age >= 8)	Binary risk score for diabetes based on hypertension, cholesterol, obesity, inactivity, and age.

Table 1 - Feature Engineering Information