Diabetes Diagnosis
Predicting Diabetes Through Binary and Ternary Classification Approaches
Final Project

Pedro Silva, Ramyad Raadi
uc2023235452@student.uc.pt, uc2023205634@student.uc.pt

University of Coimbra, Faculty of Science and Technologies
Bachelor of Data Science and Engineering, ML Course

May 2025

1 Introduction

Diabetes is a major chronic disease affecting hundreds of millions worldwide, and early detection is crucial to prevent complications. According to the International Diabetes Federation (IDF), roughly 537 million adults had diabetes in 2021, a number projected to rise to 783 million by 2045, according to several studies[1][2]. The type 2 diabetes, in particular, is associated with serious outcomes such as cardiovascular disease, kidney failure, and neuropathy, and even milder stages (prediabetes) carry elevated health risks. However, early and accurate diagnosis is challenging for health techniques, especially at the population level, as the study [1] reports. Predictive models using routine health surveys and clinical data can support public health efforts by identifying high-risk individuals before complications arise. Indeed, many recent studies have applied machine learning (ML) to automate diabetes risk prediction, showing promise in augmenting traditional screening methods. Therefore, the main goal of this project is to investigate early detection of diabetes or prediabetes using two datasets - binary and ternary classification - and using Support Vector Machines (SVM), Neural Networks (NN) and Random Forest (RF) as predictive models instruments. This study will follow the workflow shown below for both datasets:

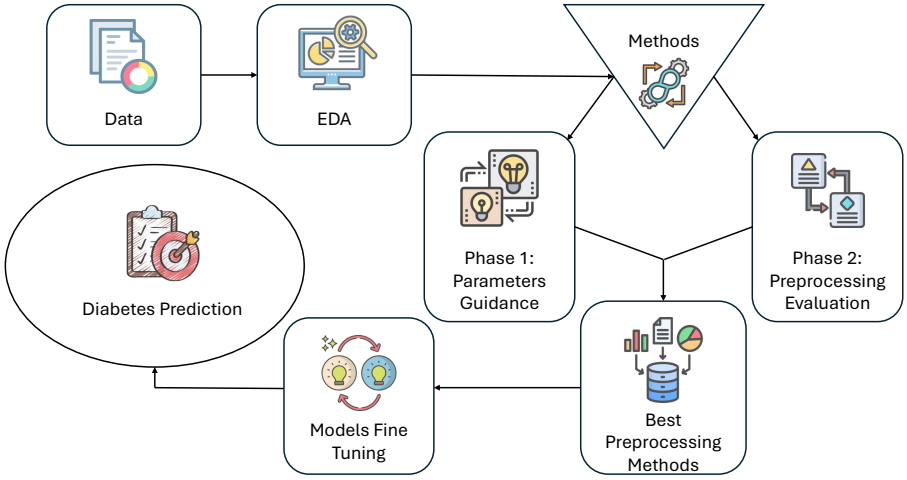


Figure 1: Graphical Abstract of This Study Workflow

2 Related Work

Predicting diabetes and its risk factors using machine learning has become an important research area in public health. Several studies made utilizing machine learning models to classify diabetes or predict risk factors from health-related data.

A study by Ahmed et al. (2020) [3] applied various classification algorithms, including RF and SVM, to predict diabetes using clinical data. Their results showed that SVM achieved high accuracy, but they emphasized the importance of careful data preprocessing, such as handling missing values and feature scaling, to improve model performance. Similarly, Chaurasia et al. (2021) [4] used RF and SVM models to predict diabetes onset in a population dataset. They found that Random Forest outperformed other algorithms in terms of robustness, especially in the presence of imbalanced classes, achieving 89.0% accuracy.

Neural Networks (NN) have also been applied to diabetes prediction, particularly in cases with more complex, high-dimensional data. For instance, Khanam and Foo (2024) [5] demonstrated the effectiveness of deep learning models in predicting diabetes risk, achieving 99% accuracy with a neural network incorporating attention mechanisms. However, they noted

that deep learning models require large, clean datasets and more computational resources, which can be a limitation in some healthcare settings.

Another notable study by Sahoo et al. (2022) [6] compared multiple machine learning techniques, including SVM, RF, and NN, for diabetes prediction using health survey data. The authors found that although all models performed well, Random Forest and SVM had a slight edge over neural networks, especially when feature selection was applied. This study reinforced the idea that ensemble methods like Random Forest tend to be more stable in real-world applications, where datasets can be noisy or incomplete.

In summary, the application of machine learning models such as SVM, RF, and NN to predict diabetes has been widely studied, with each model showing strengths in different aspects of the prediction task. SVM and RF tend to perform well with smaller or more imbalanced datasets, while NN models excel in handling complex relationships in larger datasets. These findings underscore the importance of carefully choosing the right algorithm based on dataset characteristics and the need for robust preprocessing techniques.

3 Materials

3.1 Data Description

In this project, we used 2 preprocessed datasets made available by Alex Teboul [1], but the original data is from the CDC's Behavioral Risk Factor Surveillance System (BRFSS) 2015 survey, as provided on Kaggle [2] and contains responses from 441 455 individuals and has 330 features.

The datasets used contains responses from 253 680 U.S. adults, with 21 features spanning demographics, lifestyle and health indicators. The target labels classify each respondent as healthy or diabetic (binary dataset) or healthy, prediabetic or diabetic (ternary dataset), being these the only difference between both datasets to be used.

The features, and it's description, are in the figure below and you can find more information about them in the footnotes.

[1] **Project Datasets:** `https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset`

[2] **Original Dataset:** `https://www.kaggle.com/datasets/cdc/behavioral-risk-factor-surveillance-system`

Column	Description
Diabetes_binary [Binary Dataset]	No diabetes or diabetes (0,1)
Diabetes_012 [Ternary Dataset]	No diabetes, prediabetes or diabetes (0,1,2)
HighBP	Adults with high blood pressure - measured by a health professional (0,1)
HighChol	High blood cholesterol - measured by a health professional (0,1)
CholCheck	Cholesterol check within past five years (0,1)
BMI	Body Mass Index (BMI)
Smoker	Smoked at least 100 cigarettes in entire life (0,1)
Stroke	No stroke or stroke (0,1)
HeartDiseaseorAttack	Coronary heart disease (CHD) or myocardial infarction (MI) (0,1)
PhysActivity	Adults who reported doing physical activity during the past 30 days other than their regular job (0,1)
Fruits	Consume fruit 1 or more times per day (0,1)
Veggies	Consume vegetables 1 or more times per day (0,1)
HvyAlcoholConsump	Heavy drinkers: men >14 drinks/week, women >7 drinks/week (0,1)
AnyHealthcare	Health care coverage (0,1)
NoDocbcCost	Needed to see a doctor in past year but could not because of cost (0,1)
GenHlth	General health rating (1–5)
MentHlth	Days in past 30 days mental health was not good (0–30)
PhysHlth	Days in past 30 days physical health was not good (0–30)
DiffWalk	Serious difficulty walking or climbing stairs (0,1)
Sex	Sex of respondent (0 = Female, 1 = Male)
Age	Fourteen-level age category (1–14)
Education	Highest grade or year of school completed (1–6)
Income	Annual household income level (1–8; refusal coded separately)

Table 1: Dataset Column Descriptions. Adapted from: `https://www.kaggle.com/code/rahul713/diabetes-data-analysis#Data-Analysis`

3.2 Tools and Packages

The project was implemented using the following tools and packages:

- **Scikit-Learn, tensorflow, optuna:** Utilized for constructing, training, optimize and evaluating the machine learning models.
- **Pandas, Numpy, Itertools, Collections, Imblearn:** for data manipulation, preprocessing and feature engineering tasks.
- **Copy:** used for copying lists without conflicts.
- **Scipy (stats):** Utilized for statistics in exploratory data analysis (EDA).
- **Matplotlib, Seaborn:** Used for visualizing the data and understanding trends in EDA.

4 Exploratory Data Analysis

In this section, we'll explore in depth the datasets, considering that the major difference between them is the target feature. To do this task, we'll answer some questions about the data with graphics, tables and statistics.

The following subsections divide the main aspects that are more relevant according to the problem context.

4.1 Data Overview

In data overview, we studied:

1. **Inspection of data types:** Here, we discovered that all variables are floats and the majority of them are binary. The ones that are not binary - continuous or categorical - are:

   - `BMI:` Continuous - varying from 12 to 98.
   - `GenHlth:` Categorical - varying from 1 to 5.
   - `MentHlth:` Continuous - varying from 0 to 30.
   - `PhysHlth:` Continuous - varying from 0 to 30.
   - `Age:` Categorical - varying from 1 to 13.
   - `Education:` Categorical - varying from 1 to 6.
   - `Income:` Categorical - varying from 1 to 8.

2. **Missing values and Duplicates:** We detected no missing values, but on the other hand, we detected 24206 duplicate cases. This duplicates will be dropped in the Preprocessing.

4.2 Target Variable Exploration

In Target Variable Exploration, we studied:

1. **Class Distribution:** We explore in detail the balance between categories in both datasets. We discovered that the binary dataset is unbalanced, with `No Diabetes` class having 86.07% and the `Diabetes` class having 13.93%. The ternary dataset, by comparison, is also unbalanced. The `No Diabetes` class has 84.24%, the `Prediabetes` class 1.83% and the `Diabetes` class has the remaining percentage: 13.93%. With these insights, we concluded that the `Diabetes` is equal in both

datasets, which means that probably, in the datasets containg 3 categories on the target, the models will encounter difficulties in distinguish `Prediabetes` and `No Diabetes`. Concluding, it is mandatory to make use of oversampling or undersampling techniques. See Figures 2 and 3.

4.3 Univariate Analysis

In Univariate Analysis, we studied:

1. **Continuous Features and Age Distribution:** Continuous features such as `BMI`, `PhysHlth`, and `MentHlth`, along with the categorical feature `Age`, were analyzed, since they appeared relevant to our study, using histograms, bar charts and skewness values. Most features exhibited strong negative skewness — particularly `PhysHlth` and `MentHlth`, indicating that the majority of individuals had very few unhealthy days, with fewer reporting high day values.

2. **Binary Features Distribution:** Similarly to `PhysHlth` and `MentHlth`, the investigation of the balance of binary variables revealed that the majority of variables are not balanced and showed an imbalance in some behaviors: a lower proportion of heavy alcohol consumption and fewer individuals reporting physical activity. This discoverers leads to Bivariate Analysis.

Check the Figure 5 and it's subfigures.

4.4 Bivariate Analysis

In Bivariate Analysis, we studied:

1. **Correlation Heatmap Analysis:**

   The correlation heatmap of continuous features (Figures 6 and 7) revealed limited strong linear relationships. There were no features that showed strong direct correlation with the binary diabetes outcome as well as the ternary diabetes. However, `HighBP` is the feature with most correlation for both diabetes variables, with $r = -0.25$ and $r = -0.25$, respectively for the binary and ternary ones.

   We can note other linear associations between features in Figure 8 and 9:

- `PhysHlth` and `DiffWalk`: strong positive correlation ($r = 0.48$).
- `MentHlth` and `PhysHlth`: moderate positive correlation ($r = 0.35$).
- `HighChol` and `HighBP`: moderate positive correlation ($r = 0.30$).
- `DiffWalk` and `PhysActivity`: moderate negative correlation ($r = -0.25$).

2. **Cramér's V Correlation Analysis:**

   A Cramér's V correlation matrix (Figures 8 and 9) was calculated to explore associations between categorical features. The results indicate that none of the categorical variables are strongly related to each other, suggesting that the categorical features are relatively independent in these datasets, including the diabetes variables.

3. **Potential Relationships Between Continuous Features and Diabetes:**

   Based on the analysis of Figure 10 we draw the following conclusions:

   (a) **`BMI` vs `Diabetes`:** Individuals with diabetes tend to have higher BMI compared to those without the condition. Additionally, there appears to be a progressive increase in BMI from non-diabetic to prediabetic to diabetic individuals, as shown in Figure 10a).

   (b) **`MentHlth` and `PhysHlth` vs `Diabetes`:** Healthy individuals report fewer days of poor mental and physical health compared to those with diabetes. However, the distinction between prediabetic and diabetic groups is less pronounced for these variables.

4. **Lifestyle Factors and Diabetes Prevalence:**

   This subsection examines how lifestyle behaviors, such as smoking, heavy alcohol consumption, and physical activity, relate to diabetes. From the analysis of Figure 11 we conclude:

   (a) Smoking is associated with a higher prevalence of diabetes.

   (b) Although the features `PhysActivity` and `HvyAlcoholConsump` are imbalanced (see Table 11), current results suggest that individuals who practice physical activity or consume alcohol heavily may show a lower prevalence of diabetes. However, due to the imbalance, these findings should be interpreted with caution, especially the relationship between alcohol and diabetes.

5. **Smoking and Alcohol Consumption:**

   A chi-square test showed a highly significant association between smoking and heavy alcohol consumption ($p < 0.0001$). This indicates that smokers are more likely to consume alcohol heavily compared to non-smokers. We can visualize this on Figures

6. **Age and High Cholesterol:**

   A Mann-Whitney U Test confirmed a significant difference in age between individuals with and without high cholesterol ($p < 0.0001$). This suggests that older individuals are more likely to have high cholesterol. Also, this affirmation can be evaluated in Figure 12.

7. **Physical and Mental Health Relationship:**

   Both Spearman ($\rho = 0.31$) and Pearson ($r = 0.35$) correlations reveal a moderate positive relationship between physically and mentally unhealthy days. This suggests that individuals reporting poor physical health are also more likely to experience poor mental health.

8. **Health Care Access by Sex:**

   A chi-square test showed a statistically significant association between `AnyHealthcare` coverage and `Sex` ($p < 0.05$), indicating a potential difference in health care access between genders. Check Figure 13

9. **BMI and Dietary Habits:**

   It was made statistical tests between BMI and consumption of fruits and vegetables (Figure 14):

   - Mann-Whitney U Test for BMI vs `Fruits`: $p < 0.0001$
   - Mann-Whitney U Test for BMI vs `Veggies`: $p < 0.0001$

   These results suggest a significant difference in BMI between people who consume fruits or vegetables regularly and those who do not.

10. **Income Level and Education:**

    A statistically significant and moderate association was found between education and income (Cramér's V = 0.22, $p < 0.001$), suggesting that individuals with higher education levels are somewhat more likely to report higher income levels. We can check that on Figure 15 .

11. **BMI and Sex:**

    Analyzing the box-plot in Figure 16 we notice that there does not seem to be any relevant difference between `BMI` by gender.

4.5 Multivariate Analysis

While univariate and bivariate analysis help uncover direct relationships, multivariate analysis allows us to explore more complex interactions between health factors. This can reveal clusters of individuals with similar health profiles that can contribute to diagnosis diabetes.

1. **BMI Average:**

   The average BMI increases progressively across diabetes categories — both in the binary and ternary classifications - as we can see on Figure 17. This reinforces the well-established association between elevated BMI and diabetes risk, as seen previously, suggesting a potential gradient of risk as BMI rises.

2. **Impact of Difficulty Walking on Diabetes by Sex:**

   There is a noticeable difference in diabetes prevalence between sexes with respect to walking difficulty. Men generally showed higher diabetes prevalence and a higher proportion of reported walking difficulty. Women show lower prevalence rates, regardless of difficulty status. This suggests a possible gender-related difference in physical limitations and how they relate to diabetes outcomes. Check Figure 18.

3. **BMI, Physical Health, and Mental Health Averages Across Diabetes Categories:**

   BMI and physically unhealthy days increase across the diabetes spectrum, from non-diabetics to prediabetics and diabetics, as we can verify on Figure 19. This pattern underscores the physical health burden associated with diabetes. Mental health, while slightly worse among diabetics, shows less variation between groups - particularly between prediabetics and diabetics—indicating, a potentially weaker or less direct relationship.

4.6 Note on Statistical Weights:

This analysis did not apply statistical weights. As a result, all observations were treated equally, regardless of group size. This can lead to biased

interpretations if the features are not balanced. Without weights, group imbalances may exaggerate or underestimate the true effect size.

4.7 Conclusion of Exploratory Data Analysis

This exploratory data analysis revealed trends that align with established findings from the medical literature on diabetes and prediabetes. As observed in the dataset, individuals with higher `BMI` and older age are more likely to present with diabetes or prediabetes, consistent with the CDC's recognition of these factors as primary risks for type 2 diabetes [7]. Similarly, lifestyle factors such as low physical activity and poor diet were more prevalent among diabetic individuals, supporting CDC recommendations for preventive health behaviors [9]. Furthermore, a moderate relationship between poor physical and mental health was noted, since diabetic individuals reported higher rates of both, reinforcing CDC findings on the mental health burden of diabetes [8].

5 Methods

After exploring the data, we're going to work on the main goal of this project: diabetes for both datasets: the binary and the ternary. To achieve this, it is essential to preprocessing the data. In order to use a good preprocessing, we design two setups of experiments for each dataset:

1. **Phase 1:** The first phase consists into plan of our prediction of what could be the best preprocessing workflow, with several experiments to check the general performance, using recall, precision and f-score metrics, of tree models - Support Vector Machines, Neural Networks and Random Forests - to predict what could be the best hyperparameters for methods and models.

2. **Phase 2:** The second phase aims to assess how different preprocessing methods affect model performance when using the baseline classifiers. The key goal is to determine the most effective preprocessing strategy before fine-tuning the models, using a design of experiments.

The following sections explain our workflow to complete both phases for each dataset:

5.1 Binary Dataset Methods

5.1.1 Phase 1: Feature Selection

The goal of Feature Selection is to provide better conditions for the models. Feature selection aims to remove redundant or irrelevant variables, reducing dimensionality and computational complexity, retaining essential information. The dropped columns, and the explanation of why they were dropped, are below:

1. `"PhysHlth"`, `"Veggies"` and `"NoDocbcCost"`, because we detected correlation between variables in the correlation heatmap.

5.1.2 Phase 1: Remove duplicates

1. As we discovered in EDA, there was some duplicates. These don't add anything relevant to the model's, so we removed them.

5.1.3 Phase 1: Data Preparation

1. **Principal Components Analysis:** Principal Components Analysis, or PCA, is a dimensionality reduction technique that transforms high-dimensional data into a smaller set of uncorrelated variables called principal components (PC's). It captures the most important information by maximizing variance, helping to reduce noise and improve model efficiency. To set the best number of PC's, we created the Scree Plot of PCA (Figure 20) and we concluded that 17 PC's might be a good trade off between the explained variance and reducing dimensionality.

2. **Data Standardization:** It is crucial to standardize data for the models, specially for the models that the weights of each features are decisive.

3. **Balancing and Undersampling the Dataset:** To improve model performance and reduce computational complexity, it is crucial to address the class imbalance in the dataset (Healthy: 86%; Diabetes: 14%). To better understand the structure of the data, we first applied **UMAP** (Uniform Manifold Approximation and Projection), a nonlinear dimensionality reduction technique that preserves the global and local structure of high-dimensional data in lower dimensions. This allowed us to visualize and assess the separation between the two classes. See it on Figures 23 and 24 (ternary dataset).

Based on this analysis, we determined that a better class balance could be achieved through **random undersampling**, where we reduced the majority class (class 0) to 60,000 instances using `sampling_strategy={0.0: 60000}`. This method helps prevent the model from being biased toward the majority class while maintaining enough data for generalization.

We also experimented with **SMOTEENN**, a hybrid technique combining SMOTE (Synthetic Minority Over-sampling Technique), which generates synthetic examples of the minority class, and ENN (Edited Nearest Neighbors), which cleans the data by removing ambiguous points near class boundaries. We used `sampling_strategy=0.7`. Additionally, we tested **TomekLinks**, a method that removes overlapping instances between classes to improve boundary clarity and reduce noise.

Other variations were tested, including SMOTE alone, TomekLinks alone, and various sampling strategies combined with different numbers of principal components from PCA (Principal Component Analysis), as illustrated in Figure 22.

Ultimately, the final balanced dataset consisted of approximately 53.52% for class 0 and 46.48% for class 1, providing a significantly more even distribution that helped improve model learning and evaluation.

4. **Splitting Data into train and test:** We split the data in way to get 80% of them into training and 20% for the test part.

5.1.4 Phase 1: Training Models

As discussed previously, the three models used were:

1. **Support Vector Machines (SVM):** A discriminative classifier that aims to find the optimal hyperplane that maximizes the margin between classes. It is effective in high-dimensional spaces but can be sensitive to noise and may overfit without appropriate regularization. For this phase, the hyperparameter `C` was set to 10 and `dual` to false. Additionally, `class_weight` was set to `balanced` to address class imbalance.

2. **Random Forests (RF):** An ensemble learning method that builds multiple decision trees and combines their predictions. Each tree is trained on a random subset of the data, which helps reduce overfitting.

For this phase, the hyperparameters `n_estimators` were set to 300, `criterion` to `gini`, and `max_depth` to 10, with `class_weight` set to `balanced` to handle class imbalance.

3. **Neural Networks (NN):** A model inspired by biological neural networks, consisting of layers of neurons that learn to classify data through backpropagation. It is flexible and capable of capturing complex patterns but can be prone to overfitting without regularization. For this phase, the model was defined with two hidden layers (`64` and `32` units) and dropout regularization (`0.5`) to mitigate overfitting. The optimizer used was `Adam` with a learning rate of `0.001`, and the loss function was `binary_crossentropy`.

5.1.5 Phase 1: Conclusion

After testing with the full balanced dataset and the with minor modifications on parameters on balancing techniques and some other hyperparameters as discussed before, we observed that the results were quite similar - only 1/2 hundredths changed (see table below with final results of phase 1). Despite the application of various balancing techniques, the performance metrics remained consistent across all models.

We specifically experimented, in NN, adjusting the number of hidden layers (denoted as n). When n was set below 0.5 (i.e., with more regularization), we found that the metrics for class 1 decreased while those for class 0 improved. This behavior indicated a greater degree of overfitting. On the other hand, increasing n to values greater than 0.5, specifically setting it to 0.65, led to slight improvements in the metrics for both classes, indicating a more balanced model.

In conclusion, the model performance did not significantly vary with the balancing techniques, and the hyperparameter tuning for NN showed that an n value slightly greater than 0.5 provided the best trade-off between class 0 and class 1 performance. We spectate that with hyperparameters tuning we achieve better results.

Model	Precision	Recall	F-score
SVM	0.3	0.77	0.43
RF	0.31	0.77	0.44
NN	0.36	0.60	0.45

Table 2: Initial Performance of models without fine-tuning

5.1.6 Phase 1: Notes

After the firsts results, we aimed to improve the precision of class 1 (the minority class). Several experiments were conducted, including undersampling the majority class (class 0) before splitting the data, inspired by recent studies on imbalanced learning state-of-the-art [21], [22] and [23]. Specifically, we reduced class 0 to 60,000 instances while leaving class 1 untouched. This approach helped reduce the dominance of the majority class without distorting the overall distribution.

We ensured a proper `train_test_split` using `stratify=y`, preserving class proportions across the splits. Importantly, we created 2 test set: one completely untouched by any resampling technique and other just afected by undersampling before splitting. After the data splitiing, all balancing techniques were applied strictly to the training data using cross-validation. This ensured that any synthetic data was generated only within training folds, avoiding data leakage.

The final evaluations were performed on the test set after undersampling and on clean, original test set to reflect real-world conditions. The results, on the test set after undersampling, showed promising results: good recall for class 1 (0.81) but lower precision (0.65), while class 0 achieved high precision (0.86) with lower recall (0.68). F1-scores were approximately 0.70 for both classes, which is solid given the original imbalance. But the test set of the real-world conditions showed basically the same results as previous results (balancing techniques after the data splitting).

We were expecting that with this the model could learn more, but unfortunately it led to the same results as before: not good, since the F1-score and Precision of class 1 maintained (0.45 and 0.31, respectively). Also, it proves that the good results obtain existed because the model overfitted. probably it occur data leakage between train/test sets

For now on, we will maintain the same approach as in the beginning: balancing techniques only after splitting data into train/test sets.

5.1.7 Phase 2: Data Preparation

After we get some good insights on some parameters and data behaviors, we'll try to improve the preprocessing methodology, so this subsection is similar to the first phase, but has the intuition of improving the preprocessing. For that, for each experiment, we ran the process five times, changing the `random_state` in each run. This helped reduce the impact of randomness in the data split and performance results. Above you have the design of experiments and the table with each results.

Experiment	**Model**	**Precision**	**Recall**	**F-score**
1-Default	SVM	0.30	0.77	0.43
	RF	0.31	0.77	0.44
	NN	0.36	0.60	0.45
2-SMT	SVM	0.31	0.74	0.44
	RF	0.30	0.77	0.43
	NN	0.48	0.26	0.34
3-N	SVM	0.28	0.74	0.41
	RF	0.30	0.74	0.43
	NN	0.34	0.57	0.43
4-NS	SVM	0.30	0.75	0.42
	RF	0.31	0.72	0.43
	NN	0.39	0.50	0.43
5-FS	SVM	0.30	0.76	0.42
	RF	0.30	0.77	0.43
	NN	0.35	0.61	0.45
6-NPCA	SVM	0.15	1.00	0.26
	RF	0.15	1.00	0.26
	NN	0.17	0.97	0.30

Table 3: Performance of Different Preprocessing Methods Across Models

Default	**SMOTE (SMT)**
• Removing Duplicates • Feature Selection • Standardize data • Undersampling ({0.0: 60000}) • SMOTEENN (0.7) • TomekLinks • PCA (pc = 5)	• Removing Duplicates • Feature Selection • Standardize data • Undersampling ({0.0: 60000}) • SMOTE ("auto") • TomekLinks • PCA (pc = 5)
Normalization (N)	**No Standardization (NS)**
• Removing Duplicates • Feature Selection • Normalize data • Undersampling ({0.0: 60000}) • SMOTEENN (0.7) • TomekLinks • PCA (pc = 5)	• Removing Duplicates • Feature Selection • Undersampling ({0.0: 60000}) • SMOTEENN ("auto") • TomekLinks • PCA (pc = 5)
Feat Selection (FS)	**No PCA (NPCA)**
• Removing Duplicates • Standardize data • Undersampling ({0.0: 60000}) • SMOTEENN (0.7) • TomekLinks • PCA (pc = 5)	• Removing Duplicates • Feature Selection • Undersampling ({0.0: 60000}) • Standardize data • SMOTEENN (0.7) • TomekLinks

Table 4: Comparison of preprocessing configurations

5.1.8 Phase 2: Conclusion

In summary, our evaluation of multiple preprocessing pipelines demonstrated that the combination of standard preprocessing techniques with minimal intervention consistently outperforms more aggressive alternatives. Among all tested configurations, the **Default** preprocessing method delivered the best results in terms of F-score, particularly when using the Neural Network model.

Based on these findings, we conclude that the best preprocessing approach involves using the **Default** configuration, as it effectively balances performance and complexity. It highlights the importance of combining classical techniques like PCA and Tomek Links, while avoiding excessive resampling strategies that may introduce noise or redundancy. Lastly, our results show that alternative undersampling methods do not significantly impact performance, reinforcing the robustness of the Default pipeline.

5.2 Fine-Tuning and Final Results

In this section we explore a comprehensive approach of fine-tuning, by testing a range of parameters to identify the best model configurations.

To achieve this, we use the `Optuna` package, an automatic hyperparameter optimization framework designed for machine learning. It uses state-of-the-art algorithms such as Tree-structured Parzen Estimators (TPE) to efficiently explore the hyperparameter search space and identify optimal combinations based on a specified objective function.

The following hyperparameters were tested for each model:

- **Support Vector Machines (SVM)**
  - `params = {`
    `"C": trial.suggest_float("C", 0.1, 10.0, log=True),`
    `"kernel": "rbf"`
    `"gamma": trial.suggest_float("gamma", 1e-4, 1e-1, log=True),`
    `"class_weight": "balanced"`
    `}`
- **Random Forests (RF)**
  - `params = {`
    `"n_estimators": trial.suggest_int("n_estimators", 100, 600),`
    `"max_depth": trial.suggest_int("max_depth", 5, 60),`

```
"min_samples_split":  trial.suggest_int("min_samples_split",
2, 10),
"min_samples_leaf":  trial.suggest_int("min_samples_leaf",
1, 10),
"class_weight":  "balanced",
}
```

- **Neural Networks (NN)**

  - ```
    units1 = trial.suggest_int("units1", 64, 256)
    dropout1 = trial.suggest_float("dropout1", 0.2, 0.5)
    n_layers = 2
    units_i = trial.suggest_int("units_i", 32, 128)
    dropout_i = trial.suggest_float("dropout_i", 0.2, 0.5)
    lr = trial.suggest_float("lr", 1e-4, 1e-2, log=True)
    batch_size = trial.suggest_categorical("batch_size", [32,
    64, 128])
    ```

To ensure robust evaluation, we applied on `Optuna Optimizer` with:

- **5-fold cross-validation** for reliable performance estimation.
- **30 independent trials** to ensure result stability.
- **accuracy, precision and recall multi-optimization**, to try to maximize all metrics.

After running the hyperparameter tuning process, we identified the best-performing configurations for each model. The table below summarizes the best parameters for each model, along with their corresponding accuracy, precision, recall and F-score metrics.
    ```

Model	Best Parameters	Accuracy	Precision	Recall	F-score
SVM	C: 0.0088 gamma: 0.0044 n_components: 606	0.70	0.31	0.72	0.43
RF	n_estimators: 371 max_depth: 30 min_samples_split: 2 min_samples_leaf: 3	0.81	0.35	0.55	0.42
NN	**units1: 246** **dropout1: 0.38** **units2: 116** **dropout2: 0.30** **lr: 0.0003** **batch_size: 64**	**0.67**	**0.31**	**0.73**	**0.43**

Table 5: Best Hyperparameters and Performance Metrics for Each Model

In conclusion, the best model was the Neural Networks with metrics in table 5.

5.3 Ternary Dataset Methods

5.3.1 Phase 1: Data preparation

After all the process for the binary dataset, we stick with very good knowledge of this dataset , since they're equal with exception of the target feature. With all this insights, we decided that the feature selection and the duplicates remotion would be the same.

The remaining data preparation for this dataset is very similar as before, as shown below:

1. **Principal Components Analysis:** To set the best number of PC's, we created the Scree Plot of PCA (Figure 21) and we concluded that 17 PC's might be a good trade off between the explained variance and reducing dimensionality.

2. **Data Standardization:** It is crucial to standardize data for the models, specially for the models that the weights of each features are decisive.

3. **Balancing and Under Sampling the Dataset:** To get better results whether in model predictions, whether in reducing the computational complexity, its crucial to balance all classes of diabetes (Healthy: 84%; prediabetes: 2%; diabetes: 14%).

   To better understand the structure of the data, we first applied **UMAP**. This allowed us to visualize and assess the separation between the two classes. See it on Figures 23 and 24 (ternary dataset).

   Based on this analysis, we determined that a better class balance could be achieved through **random undersampling**, where we reduced the majority class (class 0) to 35,000 instances using `sampling_strategy={0.0: 35000}`. This method helps prevent the model from being biased toward the majority class while maintaining enough data for generalization.

   We also experimented with **SMOTEENN**. We used `sampling_strategy="auto"`. Additionally, we tested **TomekLinks** to improve boundary clarity and reduce noise.

   Other variations were tested, including SMOTE alone, TomekLinks alone, and various sampling strategies combined with different numbers of principal components from PCA (Principal Component Analysis), as illustrated in Figure 23.

   Ultimately, the final balanced dataset consisted of approximately 33,78% for class 0, 34.98% for class 1 and 31,24% for class 2, providing a significantly more even distribution that helped improve model learning and evaluation.

4. **Splitting Data into train and test:** We split the data in way to get 80% of them into training and 20% for the test part.

5.3.2 Phase 1: Training Models

The same models were used: Support Vector Machines (SVM), Random Forests (RF) and Neural Networks (NN);

1. **Support Vector Machines (SVM):** For this phase, the hyperparameter `C` was set to 10 and `kernel` was set to `linear`. Additionally, `class_weight` was set to `balanced` to address class imbalance.

2. **Random Forests (RF):** For this phase, the hyperparameters `n_estimators` were set to 200, `criterion` to `gini`, and `max_depth` to 10, with `class_weight` set to `balanced` to handle class imbalance.

3. **Neural Networks (NN):** For this phase, the model was defined with two hidden layers (`64` and `32` units) and dropout regularization (`0.5`) to mitigate overfitting. The optimizer used was `Adam` with a learning rate of `0.001`, and the loss function was `categorical_crossentropy`.

5.3.3 Phase 1: Conclusion

After testing with the full balanced dataset and the with minor modifications on parameters on balancing techniques and some other hyperparameters as discussed before, we observed that the results were quite similar - only 1/2 hundredths changed (see table below with final results of phase 1) like in the binary dataset. Despite the application of various balancing techniques, the performance metrics remained consistent across all models. Also,the model performance did not significantly vary with the balancing techniques.

Model	Precision	Recall	F-score
SVM	0.46	0.48	0.47
RF	0.45	0.49	0.47
NN	0.41	0.44	0.42

Table 6: Initial Performance of models without fine-tuning. Note that these results are the macro average of each class metric.

5.3.4 Phase 2: Data Preparation

After we get some good insights on some parameters and data behaviors, we'll try to improve the preprocessing methodology, so this subsection is similar to the first phase, but has the intuition of improving the preprocessing. For that, for each experiment, we ran the process five times, changing the `random_state` in each run. This helped reduce the impact of randomness in the data split and performance results. Above you have the design of experiments and the table with each results.

Experiment	Model	Precision	Recall	F-score
1-Default	SVM	0.30	0.77	0.43
	RF	0.45	0.49	0.47
	NN	0.41	0.44	0.42
2-SMT	SVM	0.43	0.48	0.45
	RF	0.43	0.49	0.47
	NN	0.43	0.48	0.45
3-N	SVM	0.33	0.42	0.36
	RF	0.44	0.48	0.46
	NN	0.33	0.42	0.36
4-NS	SVM	0.30	0.41	0.35
	RF	0.38	0.47	0.43
	NN	0.31	0.37	0.33
5-FS	SVM	0.47	0.48	0.47
	RF	0.45	0.49	0.47
	NN	0.32	0.42	0.37
6-NPCA	SVM	0.06	0.33	0.11
	RF	0.01	0.33	0.01
	NN	0.14	0.36	0.22

Table 7: Performance of Different Preprocessing Methods Across Models.Note that these results are the macro average of each class metric.

Default	**SMOTE (SMT)**
• Removing Duplicates • Feature Selection • Standardize data • Undersampling ({0.0: 35000}) • SMOTEENN (”auto”) • TomekLinks • PCA (pc = 5)	• Removing Duplicates • Feature Selection • Standardize data • Undersampling ({0.0: 35000}) • SMOTE (”auto”) • TomekLinks • PCA (pc = 7)
Normalization (N)	**No Standardization (NS)**
• Removing Duplicates • Feature Selection • Normalize data • Undersampling ({0.0: 35000}) • SMOTEENN (”auto”) • TomekLinks • PCA (pc = 7)	• Removing Duplicates • Feature Selection • Undersampling ({0.0: 35000}) • SMOTEENN (”auto”) • TomekLinks • PCA (pc = 7)
Feat Selection (FS)	**No PCA (NPCA)**
• Removing Duplicates • Standardize data • Undersampling ({0.0: 35000}) • SMOTEENN (”auto”) • TomekLinks • PCA (pc = 7)	• Removing Duplicates • Feature Selection • Undersampling ({0.0: 35000}) • Standardize data • SMOTEENN (”auto”) • TomekLinks

Table 8: Comparison of preprocessing configurations

5.3.5 Phase 2: Conclusion

In summary, our evaluation of multiple preprocessing pipelines demonstrated that the combination of standard preprocessing techniques with minimal intervention consistently outperforms more aggressive alternatives. Among all tested configurations, the **SMT** preprocessing method delivered the best results in terms of F-score, particularly when using the Random Forest model.

Based on these findings, we conclude that the best preprocessing approach involves using the **SMT** configuration, as it effectively balances performance and complexity. It highlights the importance of combining classical techniques like PCA and Tomek Links, while avoiding excessive resampling strategies that may introduce noise or redundancy. Lastly, our results show that alternative undersampling methods do not significantly impact performance, reinforcing the robustness of the Default/SMT pipelines (since the difference between them is the use of SMOTEENN/SMOTE, respectively).

5.4 Fine-Tuning and Final Results

We had already performed some initial fine-tuning of model hyperparameters. However, in this section, we explore a more comprehensive approach by testing a broader range of parameters to identify the best model configurations.

To achieve this, we use the `Optuna` package, an automatic hyperparameter optimization framework designed for machine learning. It uses state-of-the-art algorithms such as Tree-structured Parzen Estimators (TPE) to efficiently explore the hyperparameter search space and identify optimal combinations based on a specified objective function.

The following hyperparameters were tested for each model:

- **Support Vector Machines (SVM)**
  - ```
    params = {
    "C":  trial.suggest_float("C", 0.1, 10.0, log=True),
    "kernel":  "rbf",
    "gamma":  trial.suggest_float("gamma", 1e-4, 1e-1, log=True),
    "class_weight":  "balanced"
    }
    ```
- **Random Forests (RF)**
  - ```
    params = {
    "n_estimators":  trial.suggest_int("n_estimators", 100,
    ```

```
600),
"max_depth":  trial.suggest_int("max_depth", 5, 60),
"min_samples_split":  trial.suggest_int("min_samples_split",
2, 10),
"min_samples_leaf":  trial.suggest_int("min_samples_leaf",
1, 10),
"class_weight":  "balanced",
}
```

- **Neural Networks (NN)**

  - ```
    units1 = trial.suggest_int("units1", 64, 256)
    dropout1 = trial.suggest_float("dropout1", 0.2, 0.5)
    n_layers = 2
    units_i = trial.suggest_int("units_i", 32, 128)
    dropout_i = trial.suggest_float("dropout_i", 0.2, 0.5)
    lr = trial.suggest_float("lr", 1e-4, 1e-2, log=True)
    batch_size = trial.suggest_categorical("batch_size", [32,
    64, 128])
    ```

To ensure robust evaluation, we apply on `Optuna Optimizer` with:

- **5-fold cross-validation** for reliable performance estimation.
- **30 independent trials** to ensure result stability.
- **accuracy, precision and recall multi-optimization**, to try to maximize all metrics.

After running the hyperparameter tuning process, we identified the best-performing configurations for each model. The table below summarizes the best parameters for each model, along with their corresponding accuracy, precision, recall and F-score metrics.

In conclusion, the best model was Support Vector machines , with metrics above described.
    ```

Model	Best Parameters	Accuracy	Precision	Recall	F-score
SVM	**C: 0.026** **gamma: 0.07** **n_components: 446**	**0.80**	**0.45**	**0.44**	**0.43**
RF	n_estimators: 344 max_depth: 54 min_samples_split: 6 min_samples_leaf: 3	0.79	0.42	0.42	0.42
NN	units1: 218 dropout1: 0.39 units2: 98 dropout2: 0.34 lr: 0.00034 batch_size: 64	0.58	0.42	0.48	0.39

Table 9: Best Hyperparameters and Performance Metrics for Each Model

6 Results and Discussion

The experimental results for both binary and ternary classification tasks are presented in Figures **??** (binary) and **??** (ternary). The confusion matrices reveal critical insights into model performance, particularly in the context of traffic congestion severity prediction and medical diagnosis (diabetes).

6.1 Binary Classification

The NN model achieved a balanced accuracy of 67%, which is reasonable given the complexity of diabetes pathology involving numerous variables. The confusion matrix (Figure **??**) shows:

- **True Positives (Diabetic)**: 5116 correctly predicted cases (actual diabetic classified as diabetic).
- **False Positives**: 11595 cases where non-diabetic was misclassified as diabetic (could lead to unnecessary medical interventions).
- **False Negatives**: 1903 cases where diabetic was misclassified as non-diabetic (missed diagnoses, posing serious health risks).
- **True Negatives (Non-diabetic)**: 27281 correctly predicted cases.

Despite these misclassifications, preprocessing techniques like PCA and Tomek Links improved metrics by addressing class imbalance and reducing dimensionality.

6.2 Ternary Classification

The ternary confusion matrix (Figure **??**) highlights further challenges:

- **Class 0 (Non-Diabetic)**: 21,862 correct predictions but significant misclassifications as Class 2.
- **Class 1 (prediabetes)**: Only 204 correct predictions, with most misclassified as Class 2 (472) or Class 1 (250).
- **Class 2 (diabetes)**: 4440 correct predictions, but 1401 cases were misclassified as Class 1.

The model struggles with intermediate classes, suggesting the need for more nuanced feature engineering or alternative architectures.

6.3 Medical Context (Diabetes)

In medical terms, precision and recall are critical:

- **Precision (Positive Predictive Value)**: Measures the proportion of correctly identified diabetic cases. Low precision increases false alarms, wasting resources.
- **Recall (Sensitivity)**: Indicates the model's ability to capture actual diabetic cases. Low recall risks missing diagnoses, with severe health implications.

The current models exhibit overfitting, as repeated preprocessing and tuning yielded minimal improvements. This underscores the complexity of the tasks and the limitations of the tested approaches.

7 Conclusion

The evaluation of the neural network model for binary classification tasks and the support vector machines for ternary classification tasks in the medical domain, specifically for diabetes prediction, reveals several important findings. For binary classification, the model achieved a balanced accuracy of 67%, which, while modest, reflects the inherent complexity of predicting chronic diseases based on diverse clinical variables. The confusion matrix shows a concerning rate of false positives and false negatives, indicating risks of both overdiagnosis and underdiagnosis.

In the ternary classification setting, performance deteriorates significantly, particularly in the accurate identification of the prediabetic class. The model consistently confuses intermediate cases, which is medically critical, as early intervention opportunities may be lost. These issues underscore the need for more refined classification strategies.

Preprocessing methods such as PCA, SMOTE and Tomek Links contributed to moderate improvements by addressing class imbalance and reducing feature dimensionality. However, the persistent challenges—especially in recall and class separability—highlight limitations in the current model architecture. Moreover, signs of overfitting despite extensive tuning suggest that more robust approaches are necessary.

8 Future Work

To address the observed limitations, the following directions are proposed:

- **Alternative Models**: Exploring ensemble methods (e.g.Gradient Boosting) or deep learning architectures to capture non-linear patterns in traffic and medical data.
- **Advanced Preprocessing**: Investigate synthetic data generation (e.g., others than SMOTE) or cost-sensitive learning to improve minority-class recall.
- **Cross-Domain Validation**: Test models on diverse datasets to ensure generalizability beyond the current benchmarks.

The persistent overfitting indicates a need for fundamentally different approaches, such as semi-supervised learning or domain adaptation, to enhance robustness and performance.

9 Acknowledgments

This project was a key learning experience in understanding Machine Learning. From data preprocessing to model tuning, every step helped us handle the challenges of building and optimizing predictive models. It was a hands-on journey that made us appreciate the importance of good data, the right features, and careful fine-tuning to achieve good results.

Also, we would like to mention that AI [ChatGPT] was used to:

1. Correct some syntax errors and make the text more cohesive and coherent, especially in the Introduction and Related Work sections.
2. searching medical references.
3. Get some insights into what we could explore in EDA.
4. In creating tables in LateX.
5. Building guide lines between sections.
6. Exploring Optuna optimizations

10 References

1. `https://pmc.ncbi.nlm.nih.gov/articles/PMC10107388/#:~:text=diabetes%20can%20cause%20kidney%20failure,27`

2. https://pmc.ncbi.nlm.nih.gov/articles/PMC11218995/#:~:text=disease%2C%20kidney%20damage%2C%20neuropathy%2C%20and,3%5D.%20The%20escalating

3. Ahmed, S., et al. (2020). Diabetes Prediction Using Machine Learning Algorithms. International Conference on Machine Learning and Applications (ICMLA). https://www.researchgate.net/publication/339543101_Diabetes_Prediction_using_Machine_Learning_Algorithms

4. Chaurasia, V., et al. (2021). Prediction of Diabetes Using Random Forest and Support Vector Machine. Journal of Intelligent & Fuzzy Systems. https://www.scitepress.org/PublishedPapers/2021/105638/pdf/index.html

5. Khanam, M., & Foo, E. (2024). Deep Learning for Diabetes Prediction: A Novel Approach. IEEE Access. https://www.researchgate.net/publication/381929559_Comparison_of_Machine_Learning_Models_for_Diabetes_Prediction

6. Sahoo, S., et al. (2022). Comparing Machine Learning Models for Diabetes Prediction Using Health Data. International Conference on Signal Processing and Communication Technology (ICST). https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0311222

7. Centers for Disease Control and Prevention, 2020, *Body Weight and Diabetes*, https://www.cdc.gov/diabetes/library/features/truth-about-weight.html

8. Centers for Disease Control and Prevention, 2023, *Depression and Diabetes*, https://www.cdc.gov/diabetes/managing/mental-health.html

9. Centers for Disease Control and Prevention, 2022, *National Diabetes Statistics Report*, https://www.cdc.gov/diabetes/data/statistics-report/index.html

10. https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset

11. https://www.kaggle.com/code/rahul713/diabetes-data-analysis

12. https://www.kaggle.com/code/mohamedelsayedaffan/diabetes-classification-and-deployment

13. https://www.kaggle.com/code/afshintaraghijoo/diabetes-health-prediction

14. https://www.kaggle.com/code/mohamedelsayedaffan/diabetes-classification-and-deployment

15. https://www.kaggle.com/code/alaasweed/diabetes-exploration

16. https://www.kaggle.com/code/josephcurtis/at-risk-diabetes-classifier

17. https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

18. https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html#sklearn.svm.LinearSVC

19. https://scikit-learn.org/stable/modules/generated/sklearn.kernel_approximation.Nystroem.html#sklearn.kernel_approximation.Nystroem

20. https://scikit-learn.org/stable/modules/kernel_approximation.html#kernel-approximation

21. An Undersampling Method Approaching the Ideal Classification Boundary for Imbalance Problems https://www.mdpi.com/2076-3417/14/13/5421

22. Resampling Imbalanced Data for Network Intrusion Detection Datasets https://journalofbigdata.springeropen.com/articles/10.1186/s40537-020-00390-x

23. Machine Learning-Based Network Intrusion Detection for Big and Imbalanced Data Using Oversampling, Stacking Feature Embedding, and Feature Extractionhttps://journalofbigdata.springeropen.com/articles/10.1186/s40537-024-00886-w

11 Figures and Tables

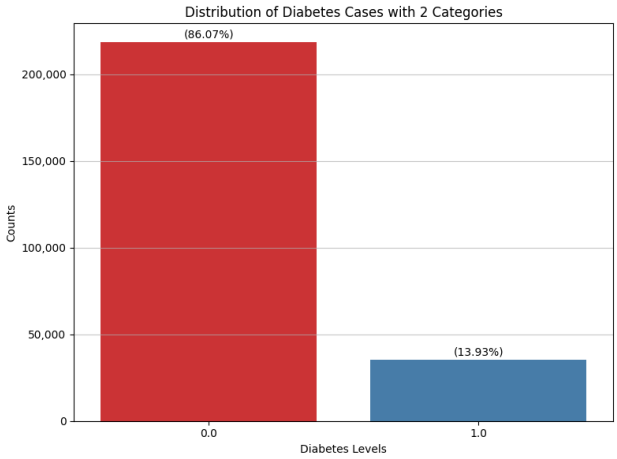


Figure 2: Distribution of Diabetes Cases - Binary target

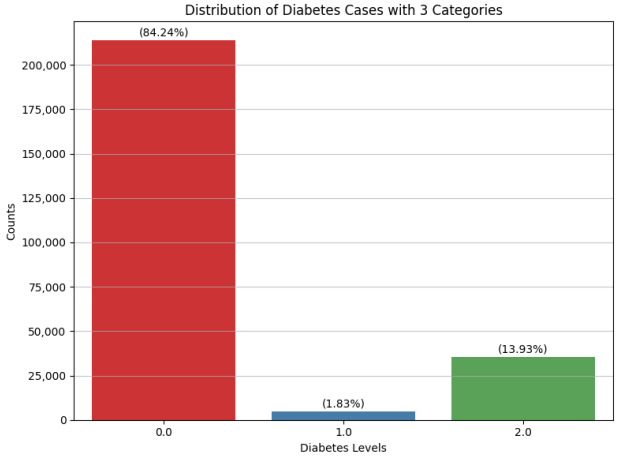
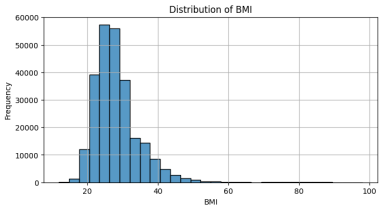
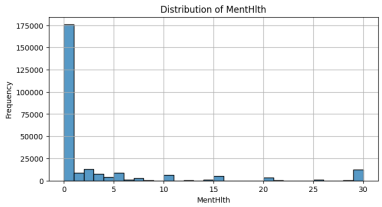


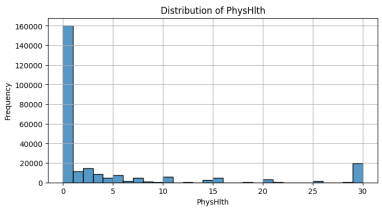
Figure 3: Distribution of Diabetes Cases - Ternary target



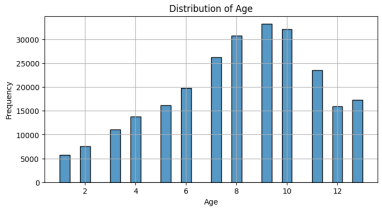
(a) BMI distribution



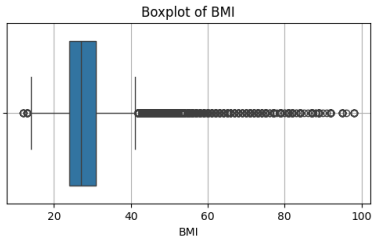
(b) MentHlth Distribution



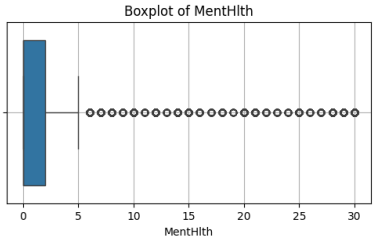
(c) PhysHlth Distribution



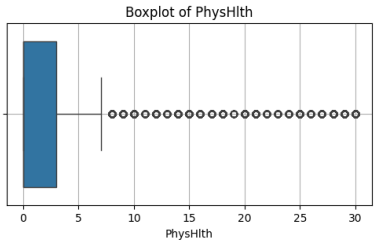
(d) Age distribution



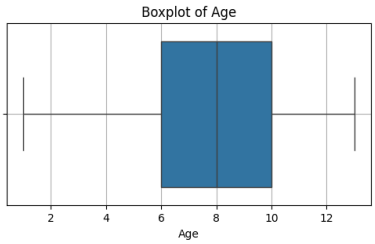
(a) BMI distribution



(b) MentHlth Distribution



(c) PhysHlth Distribution



(d) Age distribution

Figure 5: Continuous features distributions, including Age

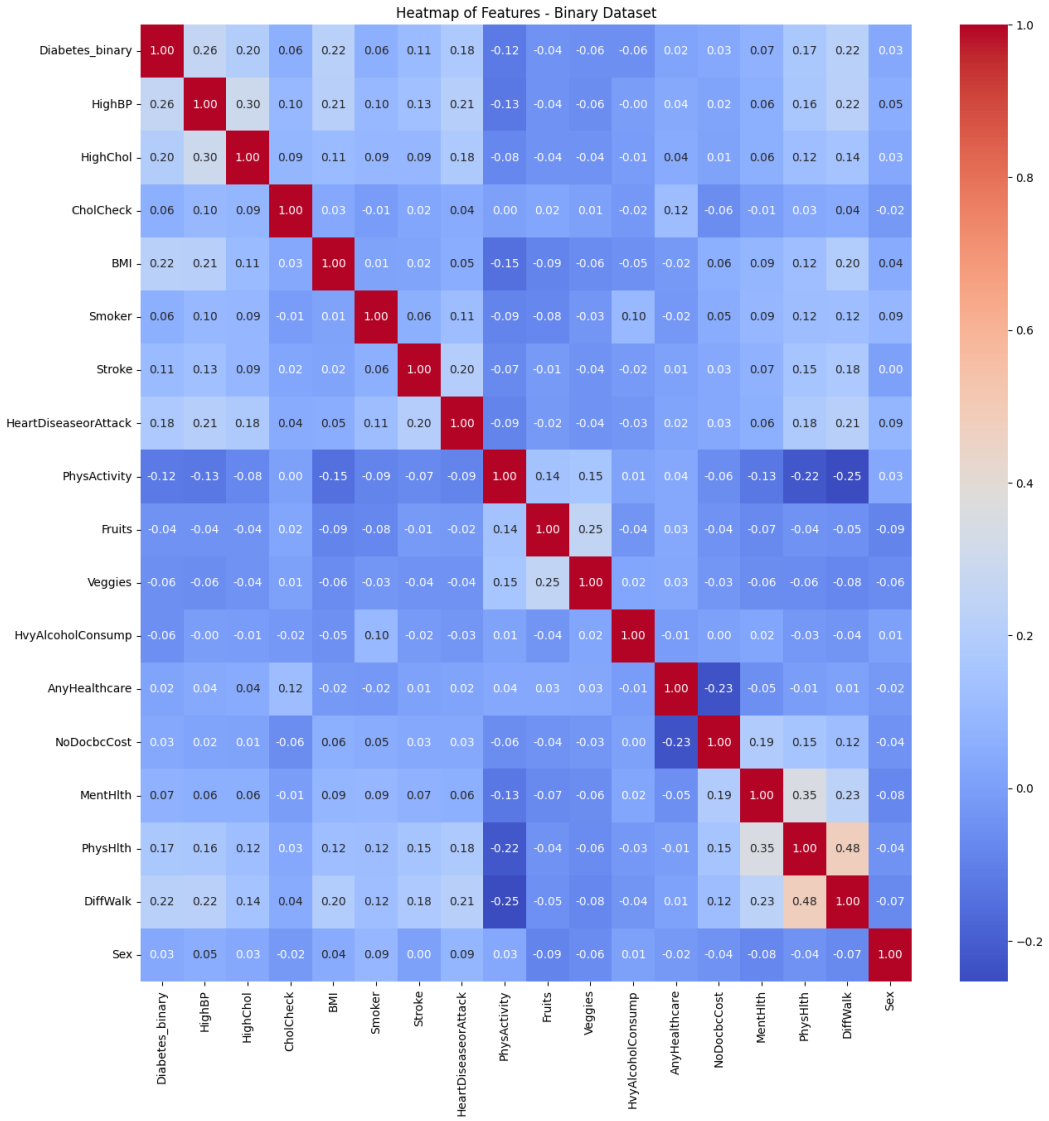


Figure 6: Heatmap Correlation of Continuous Features - Binary Target

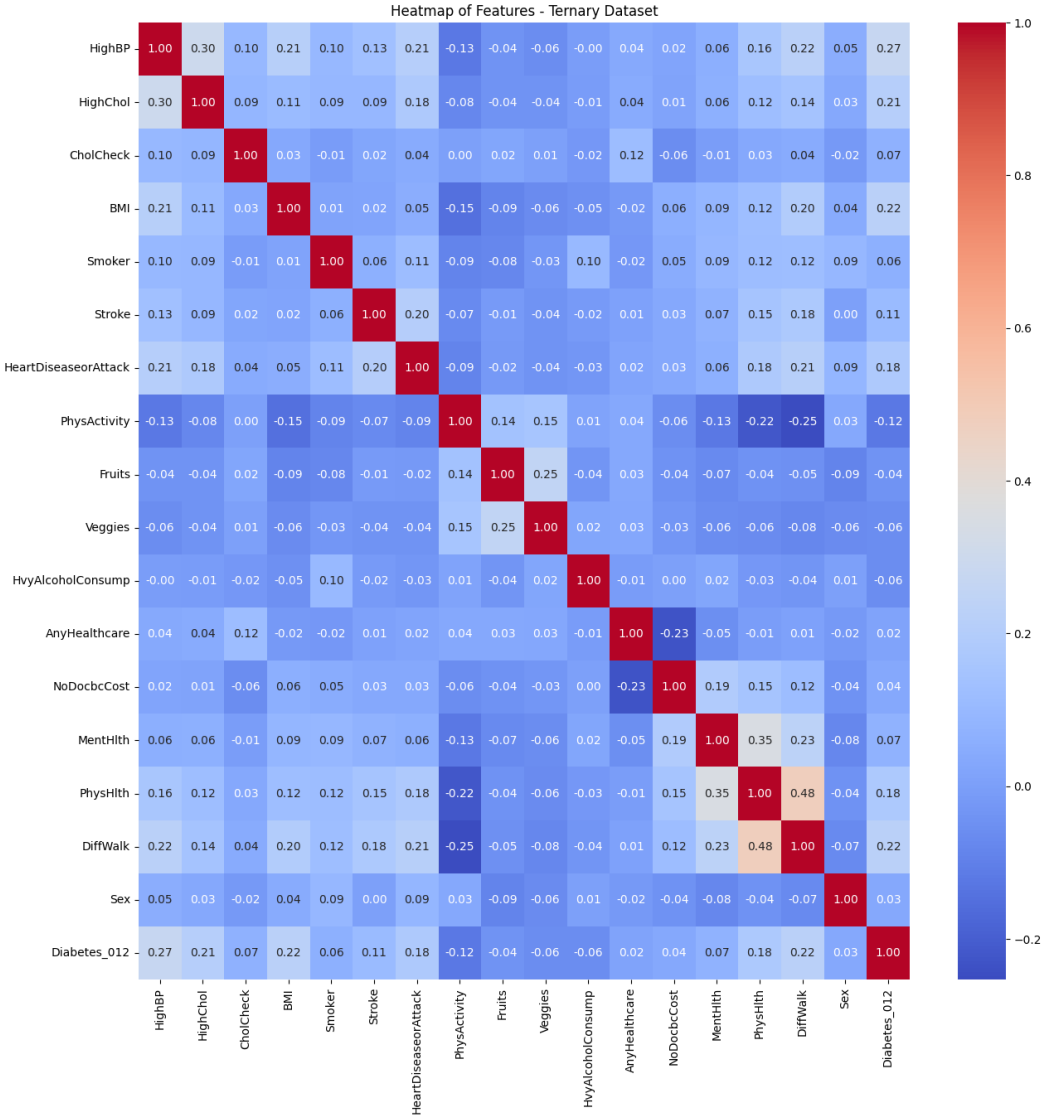


Figure 7: Heatmap Correlation of Continuous Features - Ternary Target

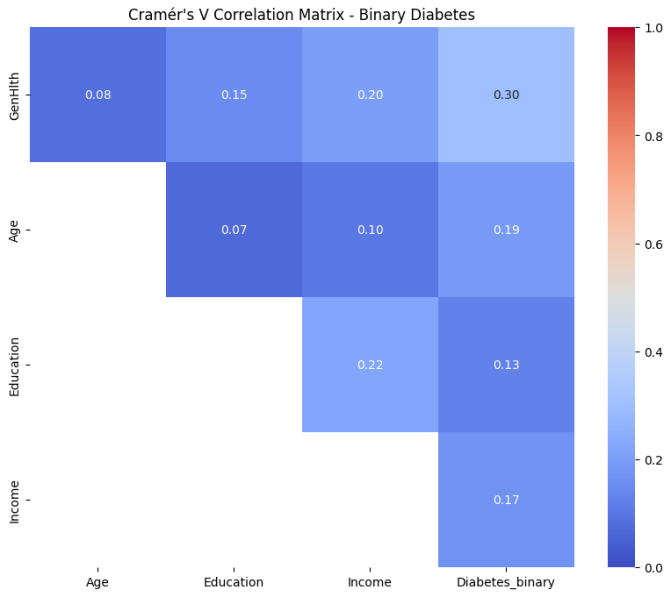


Figure 8: Heatmap Crámer's V Correlation - Binary Target

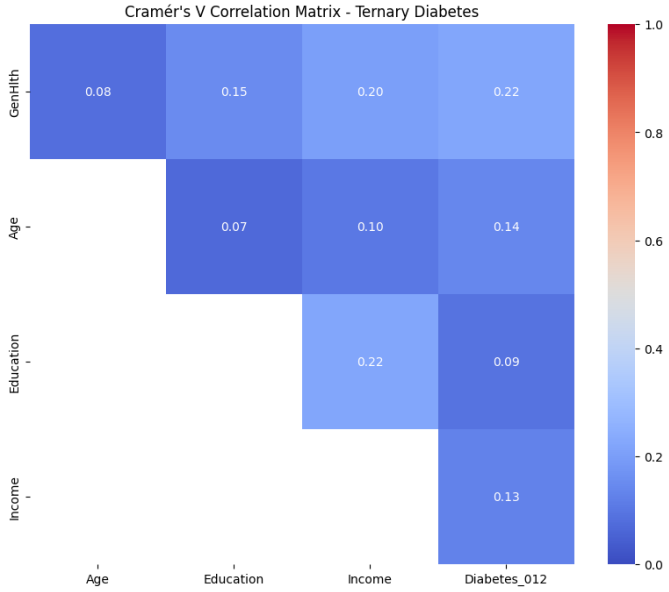
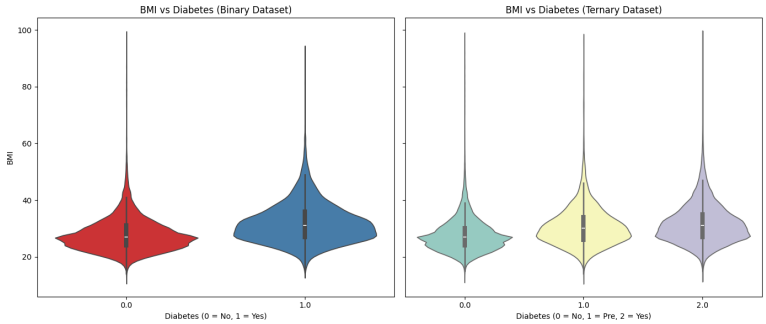
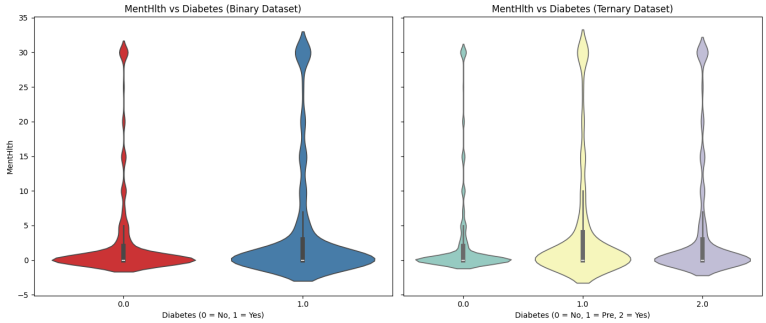


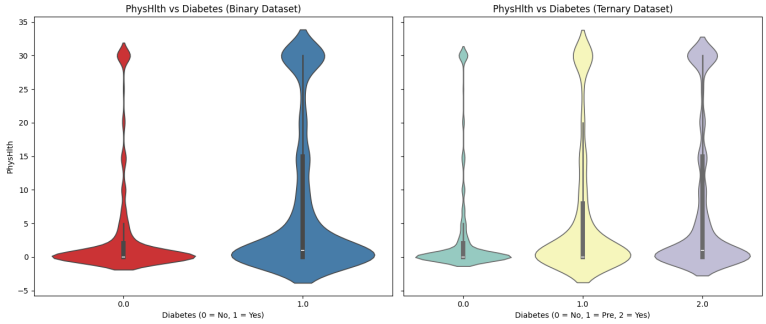
Figure 9: Heatmap Crámer's V Correlation - Ternary Target



(a) BMI and Diabetes Violin Plot

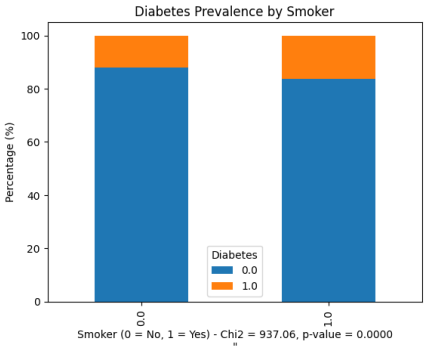


(b) MentHlth and Diabetes Violin Plot

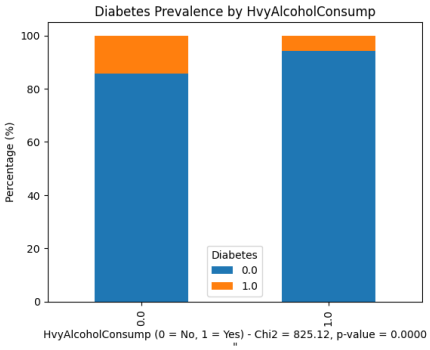


(c) PhysHlth and Diabetes Violin Plot

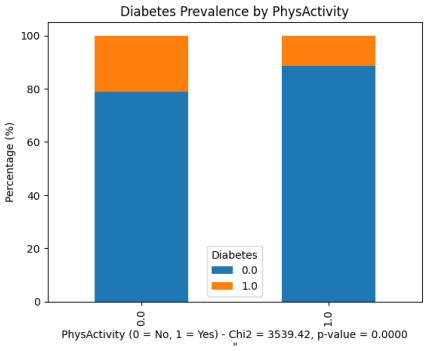
Figure 10: Violin plots showing distribution of BMI, MentHlth, and PhysHlth across diabetes categories.



(a) Diabetes Prevalence by Smoker



(b) Diabetes Prevalence by Alcohol



(c) Diabetes Prevalence by PhysActivity

Figure 11: Bar Plots of Diabetes Prevalence between Life Factors.

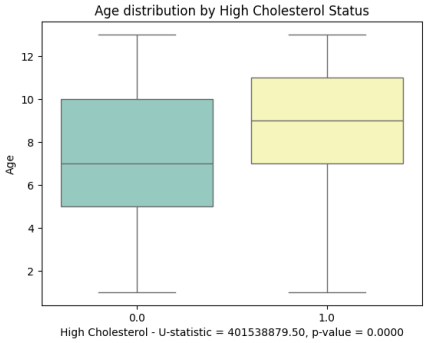


Figure 12: Age Distribution by High Cholesterol

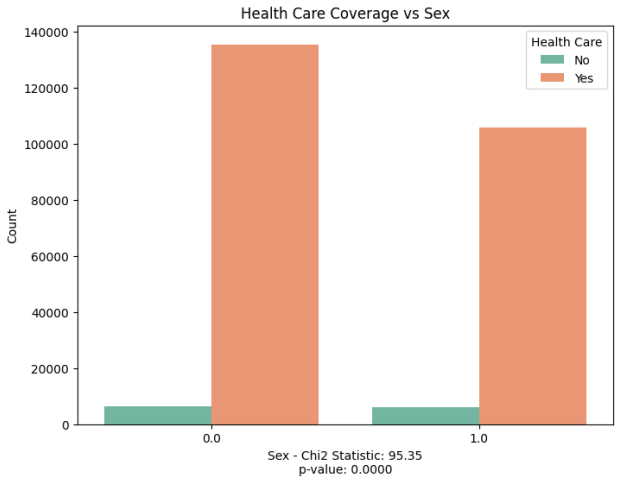


Figure 13: Health Care Access vs Sex

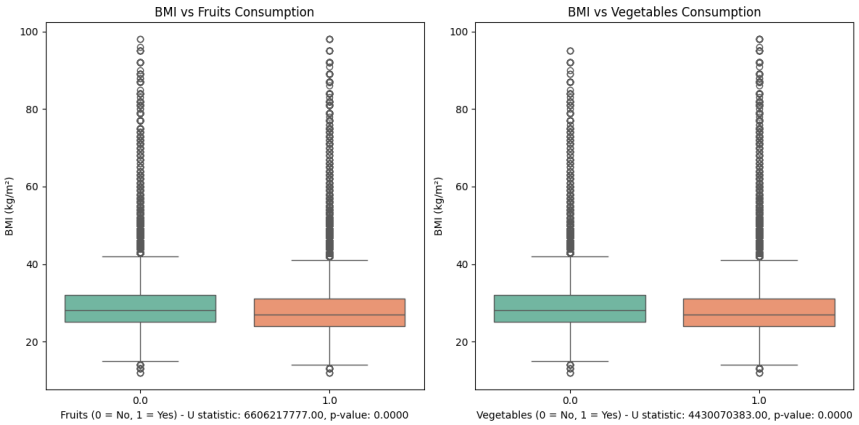


Figure 14: BMI vs Fruit or Vegetables Consumption

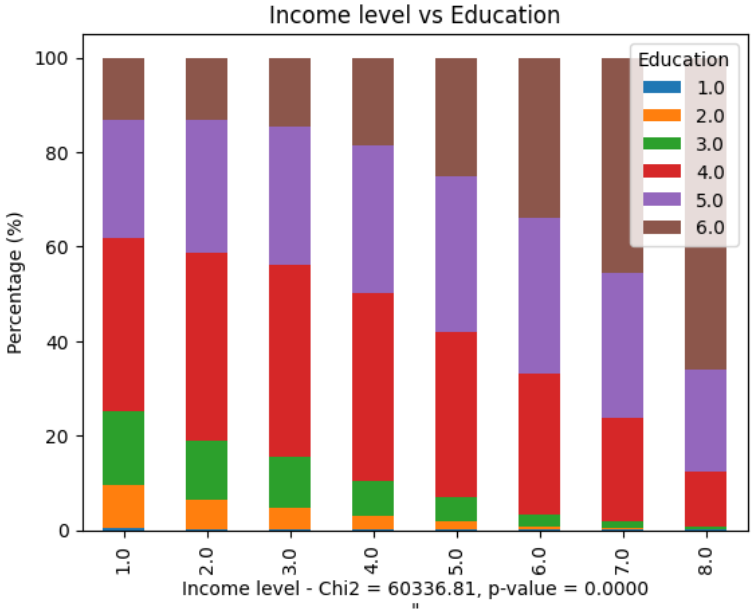


Figure 15: Income level vs Education

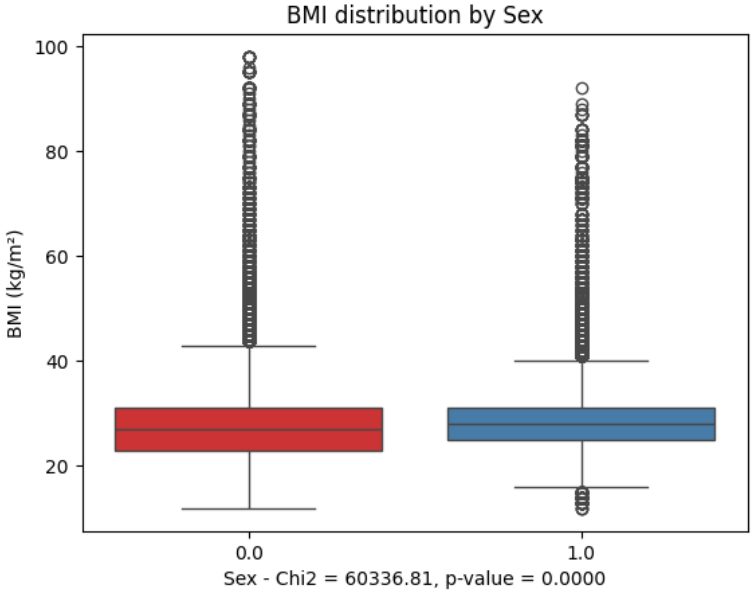


Figure 16: BMI Distribution by Sex

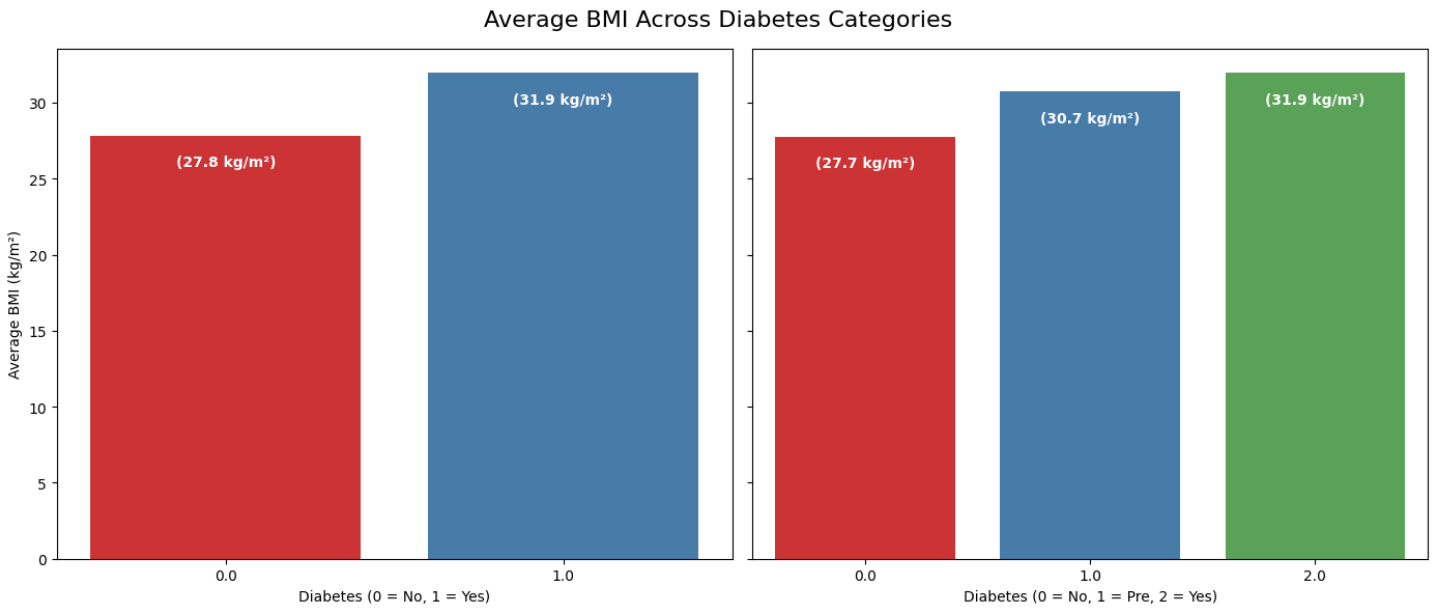


Figure 17: Average BMI Across Diabetes Categories

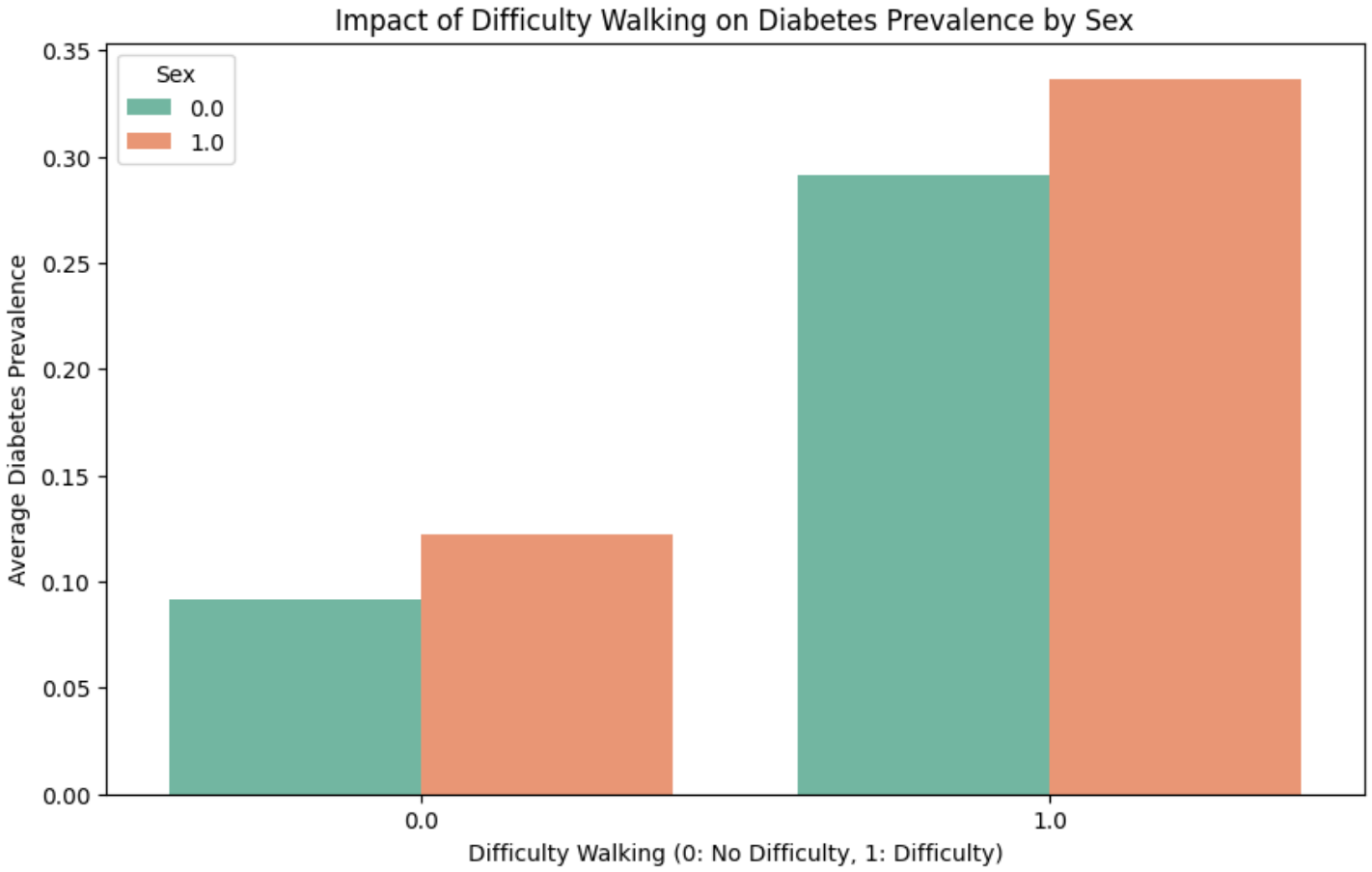


Figure 18: (BMI and Diabetes Violin Plot

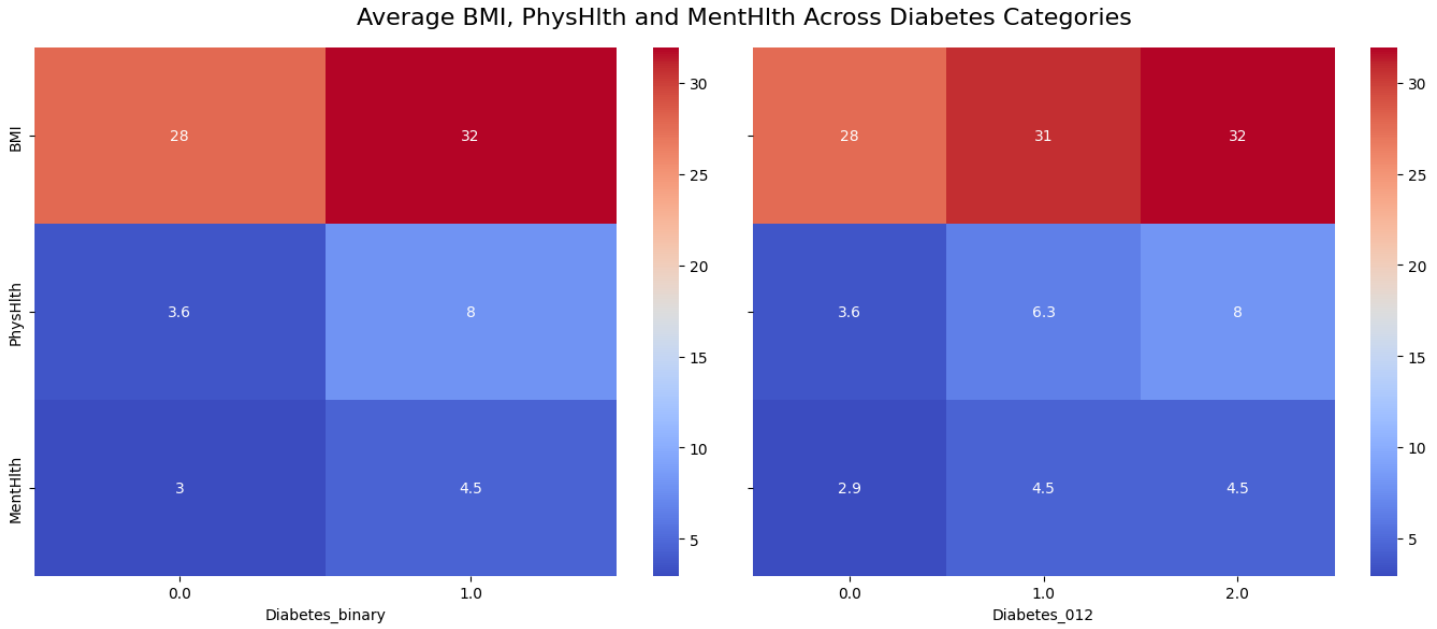


Figure 19: (BMI and Diabetes Violin Plot

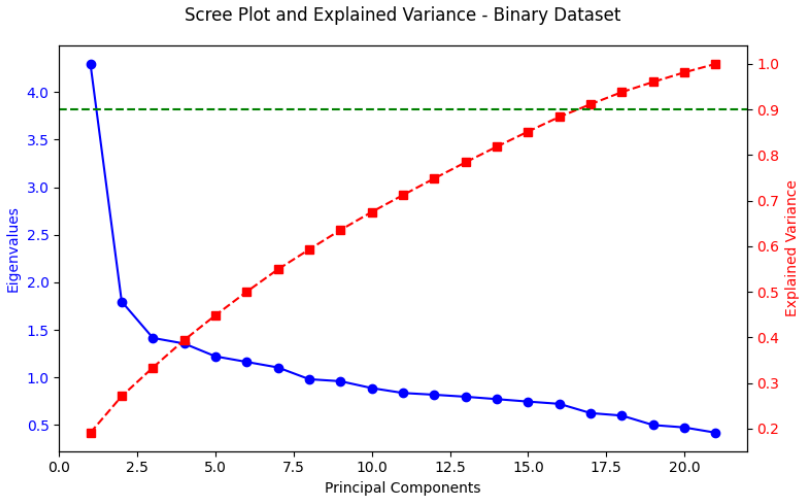


Figure 20: Scree Plot and Explained Variance - Binary Dataset

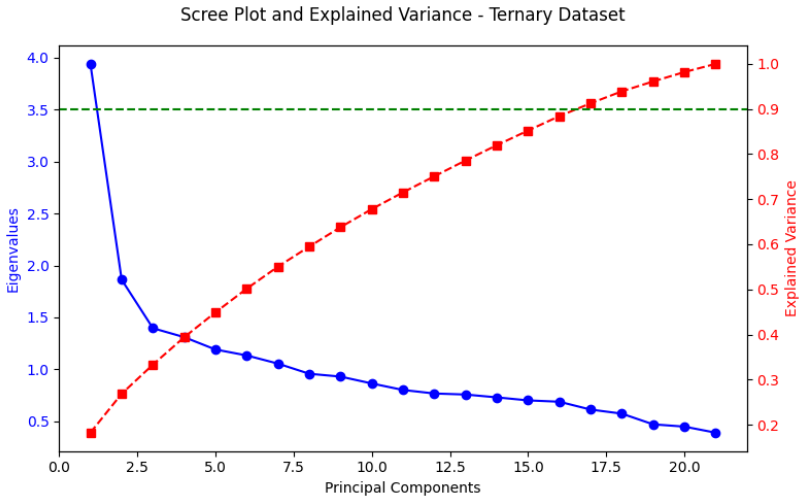
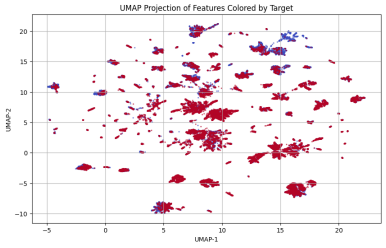
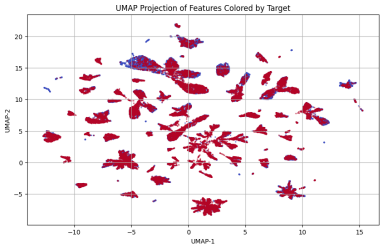


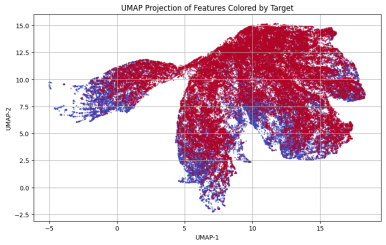
Figure 21: Scree Plot and Explained Variance - Ternary Dataset



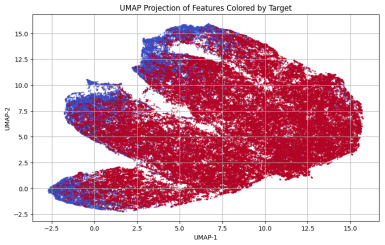
(a) PC = 17, class 0: aprox. 65% ; Class 1: aprox. 35%



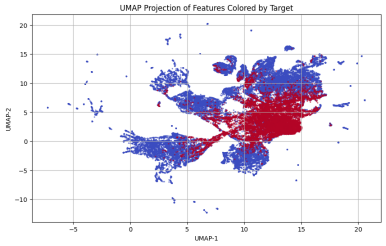
(b) PC = 13; class 0: aprox. 65% ; Class 1: aprox. 35%



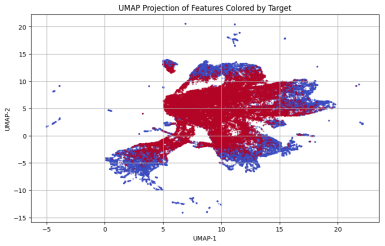
(c) PC = 7; class 0: aprox. 65% ; Class 1: aprox. 35%



(d) PC = 9; class 0: aprox. 53% ; Class 1: aprox. 47%

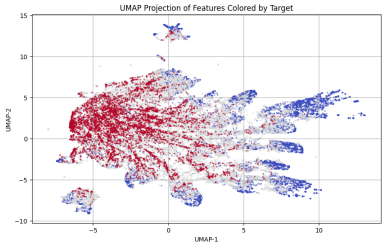


(e) PC's: 5; class 0: aprox. 53% ; Class 1: aprox. 47%

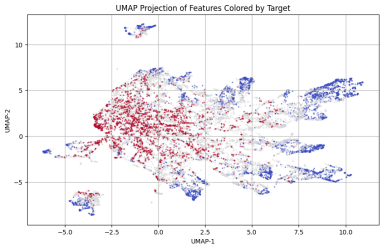


(f) PC's: 5; class 0: aprox. 72.5% ; Class 1: aprox. 27.5%

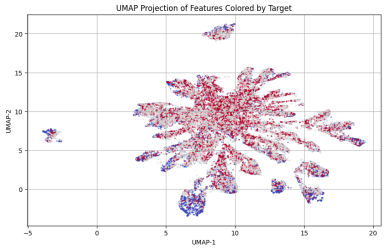
Figure 22: UMAPs of Binary dataset and it's parameters



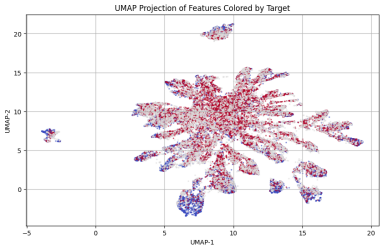
(a) PC = 17



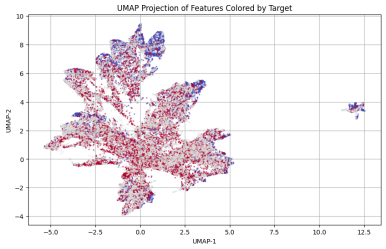
(b) PC = 13



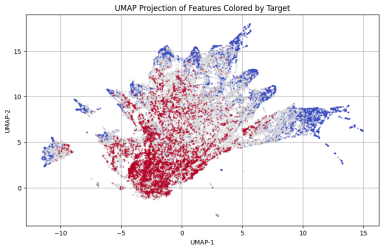
(c) PC = 7; class 0: aprox. 65% ; Class 1: aprox 35%



(d) PC = 5; class 0: aprox. 65% ; Class 1: aprox. 35%



(e) PC's: 5; class 0: aprox. 65% ; Class 1: aprox. 35%



(f) PC's: 5; class 0: aprox. 65% ; Class 1: aprox. 35%

Figure 23: UMAPs of Ternary dataset and it's parameters

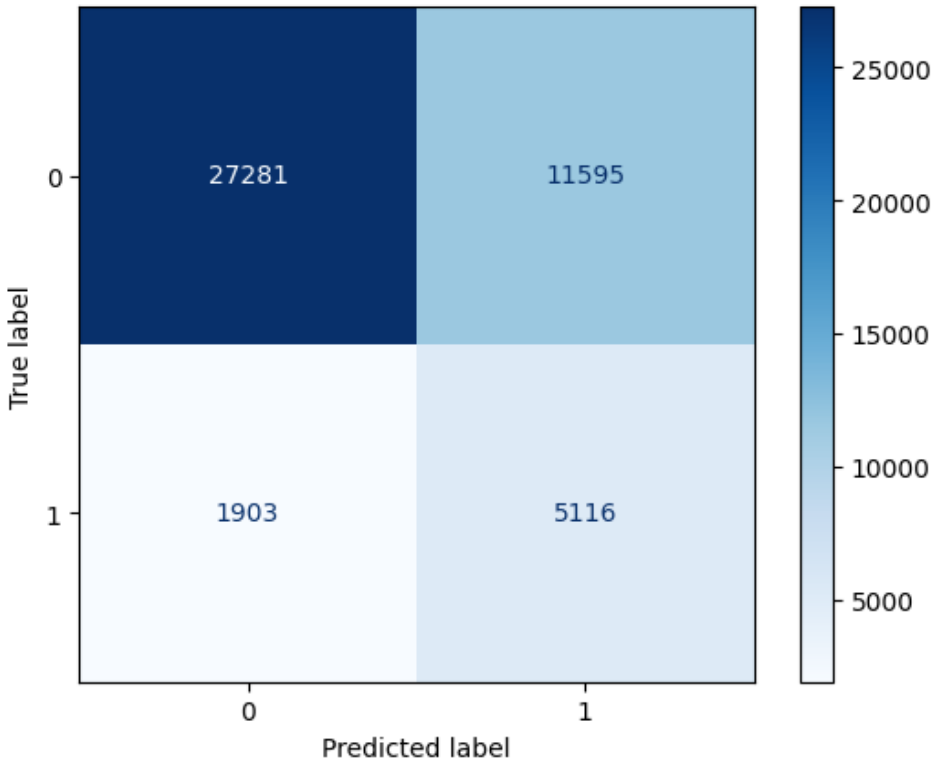


Figure 24: Confusion Matrix of Neural Network model - Binary Dataset

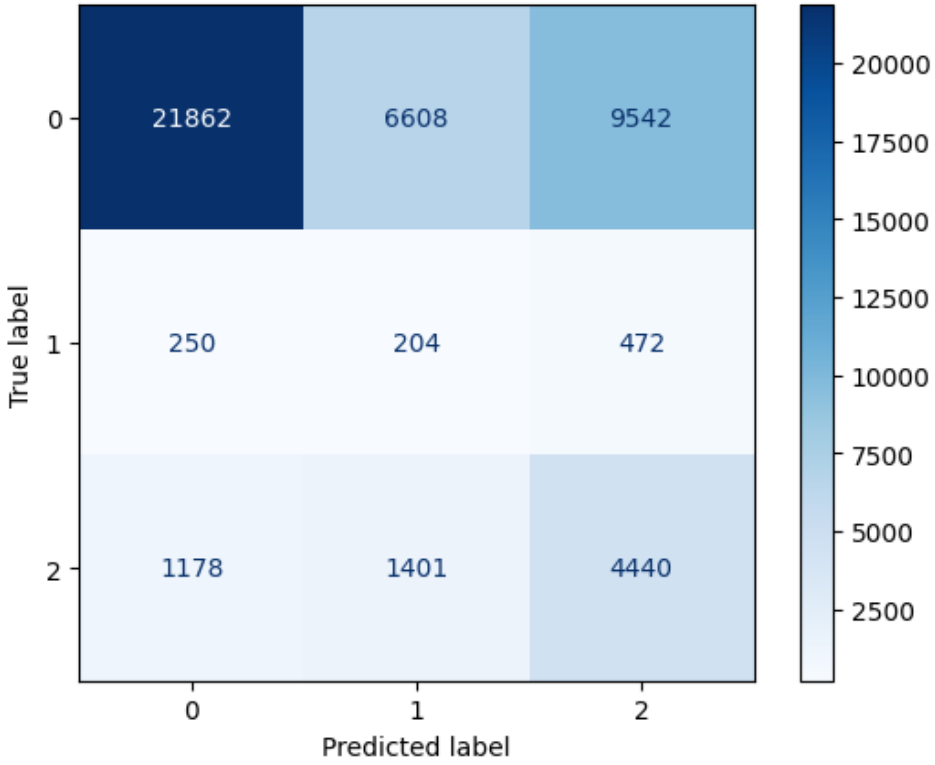


Figure 25: Confusion Matrix of Support Vector Machine model - Ternary Dataset

Variable	**Skew**	**Variable**	**Skew**	**Variable**	**Skew**
Income	**-0.89**	HighBP	0.29	HighChol	0.31
CholCheck	**-4.88**	BMI	**2.12**	Smoker	0.23
Stroke	**4.66**	HeartDiseaseorAttack	**2.78**	PhysActivity	**-1.20**
Fruits	-0.56	Veggies	**-1.59**	HvyAlcoholConsump	**3.85**
AnyHealthcare	**-4.18**	NoDocbcCost	**3.00**	GenHlth	0.42
MentHlth	**2.72**	PhysHlth	**2.21**	DiffWalk	**1.77**
Sex	**0.24**	Age	-0.36	Education	**-0.78**

Table 10: Skewness values for selected health-related variables.

Variable	**Class 0 (%)**	**Class 1 (%)**	**Variable**	**Class 0 (%)**	**Class 1 (%)**
HighBP	57.10	42.90	HighChol	57.59	42.41
CholCheck	**3.73**	**96.27**	Smoker	55.68	44.32
Stroke	**95.94**	**4.06**	HeartDiseaseorAttack	**90.58**	**9.42**
PhysActivity	**24.35**	**75.65**	Fruits	63.43	36.57
Veggies	**18.86**	**81.14**	HvyAlcoholConsump	**94.38**	**5.62**
AnyHealthcare	**4.89**	**95.11**	NoDocbcCost	**91.58**	**8.42**
DiffWalk	**83.18**	**16.82**	Sex	55.97	44.03

Table 11: Class proportions for each binary variable.