Artificial Intelligence: Assignment 2 - Applied ML

Rareș - Matei Dumitrescu, 331CA

May 26, 2024

Abstract

This document details the implementation and evaluation of various machine learning models applied to two distinct datasets for binary classification tasks. The report covers data exploration, preprocessing steps, and the use of logistic regression and multi-layer perceptron models for both salary classification and stroke prediction tasks.

Contents

1 Part I: Salary Classification

1.1 Introduction

In this assignment, we explore common tasks in the field of artificial intelligence and machine learning, particularly focusing on data visualization, feature extraction, and model evaluation for the salary classification dataset.

1.2 Dataset Description

The dataset contains personal, educational, and professional information about employees. The objective is to classify employees into two categories: those earning below $50K per year and those earning above $50K per year.

Attribute Name	Data Type	Description
fnl	Numeric	Socio-economic characteristic of the individual's population
hpw	Numeric	Number of working hours per week
relation	Categorical	Type of relationship the individual is involved in
gain	Numeric	Capital gain
country	Categorical	Country of origin
job	Categorical	Job title
edu_int	Numeric	Number of years of education
years	Numeric	Age of the individual
loss	Numeric	Capital loss
work_type	Categorical	Type of job
partner	Categorical	Type of partner
edu	Categorical	Type of education
gender	Categorical	Gender of the individual
race	Categorical	Race of the individual
prod	Numeric	Capital production
gtype	Categorical	Type of employment contract

Table 1: Attributes of the Salary Classification Dataset

1.3 Exploratory Data Analysis (EDA)

1.3.1 Attribute Analysis

Numeric Attributes For numeric attributes, we calculate and present the number of non-missing values, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values. Additionally, we provide boxplots to visualize the range of values.

The dataset contains numerical values with the following characteristics:

`fnl` (Social-economic characteristic of the population)

- **Count**: 9999

- **Mean**: 190,352.9
- **Standard Deviation**: 106,070.9
- **Min**: 19,214
- **25th Percentile**: 118,282.5
- **Median (50th Percentile)**: 178,472
- **75th Percentile**: 237,311
- **Max**: 1,455,435
- **Interpretation**: The `fnl` attribute has a wide range, with values varying from 19,214 to 1,455,435. The mean and median are relatively close, suggesting a somewhat symmetric distribution, but the high standard deviation indicates significant variability.

`hpw` **(Number of working hours per week)**

- **Count**: 9199
- **Mean**: 40.42
- **Standard Deviation**: 12.52
- **Min**: 1
- **25th Percentile**: 40
- **Median (50th Percentile)**: 40
- **75th Percentile**: 45
- **Max**: 99
- **Interpretation**: The `hpw` attribute shows that the majority of individuals work around 40 hours per week, with the median and 25th percentile at 40 hours. The standard deviation is 12.52, indicating some variation, and there are outliers working as few as 1 hour or as many as 99 hours per week.

`gain` **(Capital gain)**

- **Count**: 9999
- **Mean**: 979.85
- **Standard Deviation**: 7003.80
- **Min**: 0
- **25th Percentile**: 0
- **Median (50th Percentile)**: 0

- **75th Percentile**: 0
- **Max**: 99,999
- **Interpretation**: The `gain` attribute has a highly skewed distribution, with most individuals having no capital gain (as indicated by the 25th, 50th, and 75th percentiles all being 0). However, the maximum value is 99,999, and the high standard deviation suggests that a few individuals have very high capital gains.

`edu_int` **(Number of years of education)**

- **Count**: 9999
- **Mean**: 14.26
- **Standard Deviation**: 24.77
- **Min**: 1
- **25th Percentile**: 9
- **Median (50th Percentile)**: 10
- **75th Percentile**: 13
- **Max**: 206
- **Interpretation**: The `edu_int` attribute has an unusual distribution with some extremely high values (up to 206 years of education), likely indicating data entry errors or outliers. The mean is 14.26 years, but the median is lower at 10 years, suggesting a right-skewed distribution.

`years` **(Age of the individual)**

- **Count**: 9999
- **Mean**: 38.65
- **Standard Deviation**: 13.75
- **Min**: 17
- **25th Percentile**: 28
- **Median (50th Percentile)**: 37
- **75th Percentile**: 48
- **Max**: 90
- **Interpretation**: The `years` attribute represents ages ranging from 17 to 90, with a mean age of 38.65. The distribution appears fairly balanced, with the median close to the mean.

`loss` **(Capital loss)**

- **Count**: 9999
- **Mean**: 84.11
- **Standard Deviation**: 394.04
- **Min**: 0
- **25th Percentile**: 0
- **Median (50th Percentile)**: 0
- **75th Percentile**: 0
- **Max**: 3,770
- **Interpretation**: Similar to capital gain, the `loss` attribute is highly skewed, with most individuals having no capital loss. The high maximum value and standard deviation indicate that a small number of individuals have significant capital losses.

`prod` **(Capital production)**

- **Count**: 9999
- **Mean**: 2014.93
- **Standard Deviation**: 14,007.60
- **Min**: -28
- **25th Percentile**: 42
- **Median (50th Percentile)**: 57
- **75th Percentile**: 77
- **Max**: 200,125
- **Interpretation**: The `prod` attribute has a very wide range, with values from -28 to 200,125, indicating possible errors or outliers. The mean is 2014.93, which is much higher than the median of 57, suggesting a heavily right-skewed distribution with significant outliers.

This analysis helps in understanding the characteristics of the numeric features, which is essential for making informed decisions during the data preprocessing and modeling stages.

Categorical Attributes For categorical attributes, we calculate and present the number of non-missing values and the number of unique values. Histograms are used to visualize the distribution of values for each categorical attribute.

The dataset contains several categorical features with the following characteristics:

`relation`

- Unique values: 6
- Most common value: H (4097 occurrences)

`country`

- Unique values: 41
- Most common value: United-States (8978 occurrences)

`job`

- Unique values: 14
- Most common value: Craft-repair (1277 occurrences)

`work_type`

- Unique values: 9
- Most common value: Priv (6940 occurrences)

`partner`

- Unique values: 7
- Most common value: MCS (4667 occurrences)

`edu`

- Unique values: 16
- Most common value: HSG (3178 occurrences)

`gender`

- Unique values: 2
- Most common value: M (6179 occurrences)

`race`

- Unique values: 5
- Most common value: White (8588 occurrences)

`gtype`

- Unique values: 2
- Most common value: AC (6711 occurrences)

`money (target variable)`

- Unique values: 2
- Most common value: `<=50K` (7591 occurrences)

1.3.2 Class Balance Analysis

We create bar plots to show the frequency of each class in the training and test datasets. This helps to understand if there is any class imbalance that could affect model performance.

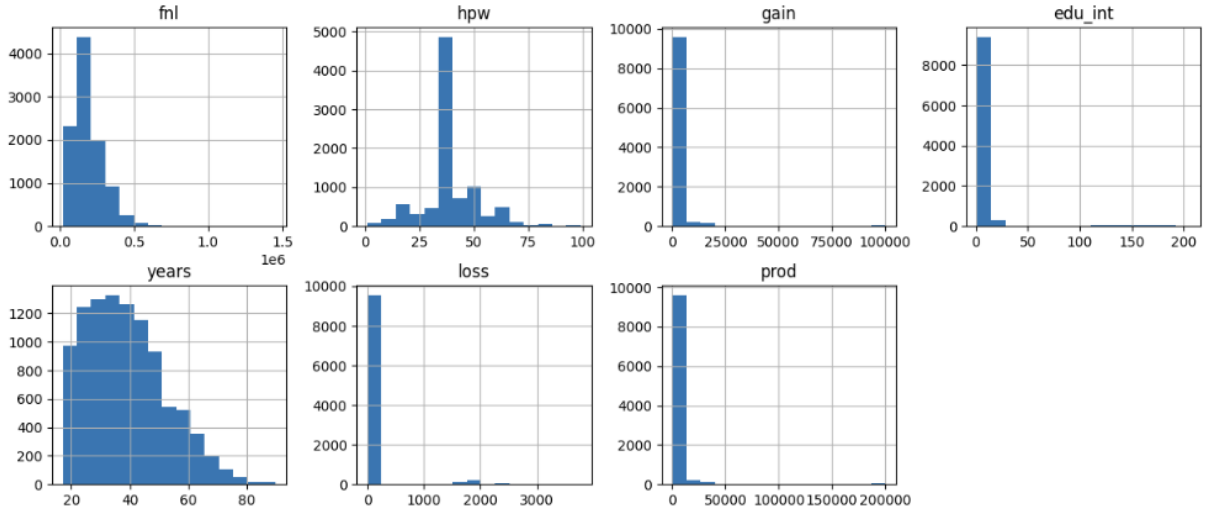

Figure 1: Numerical Features

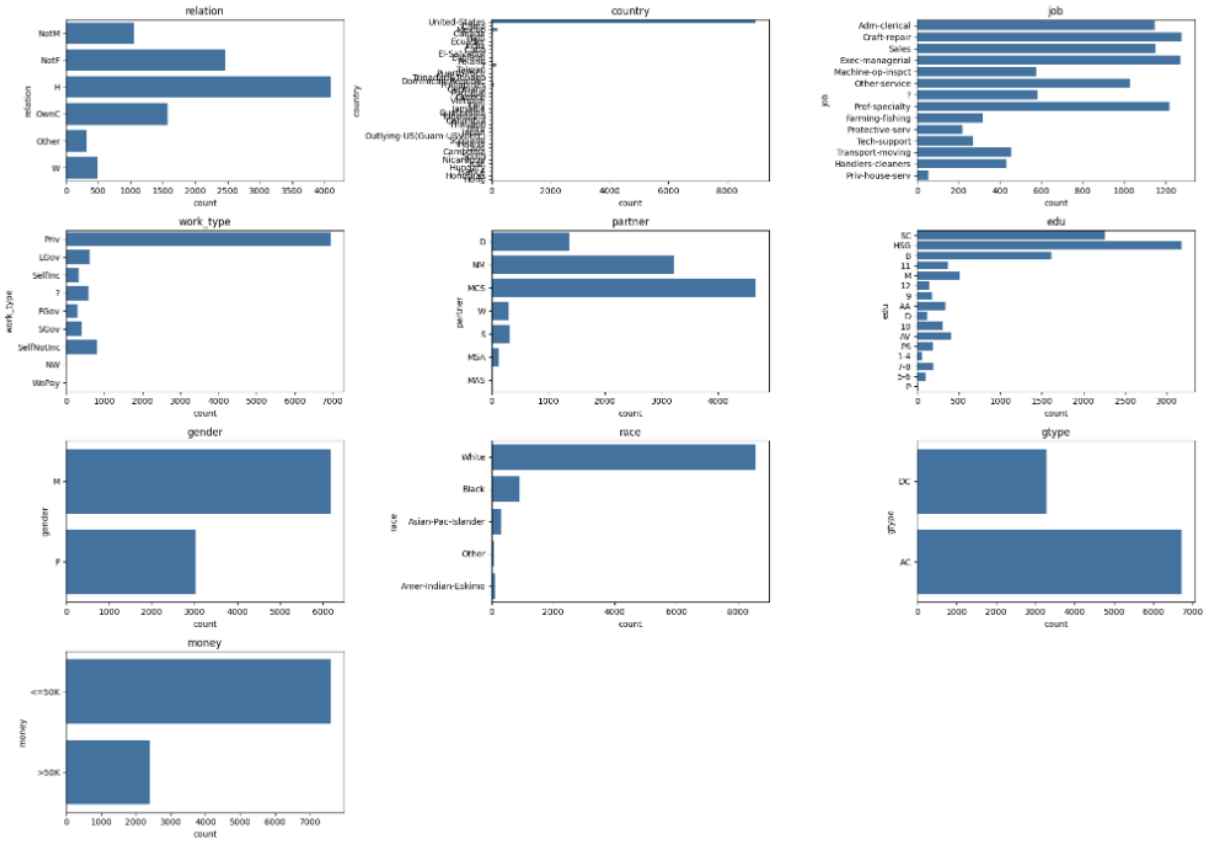

Figure 2: Categorical Features

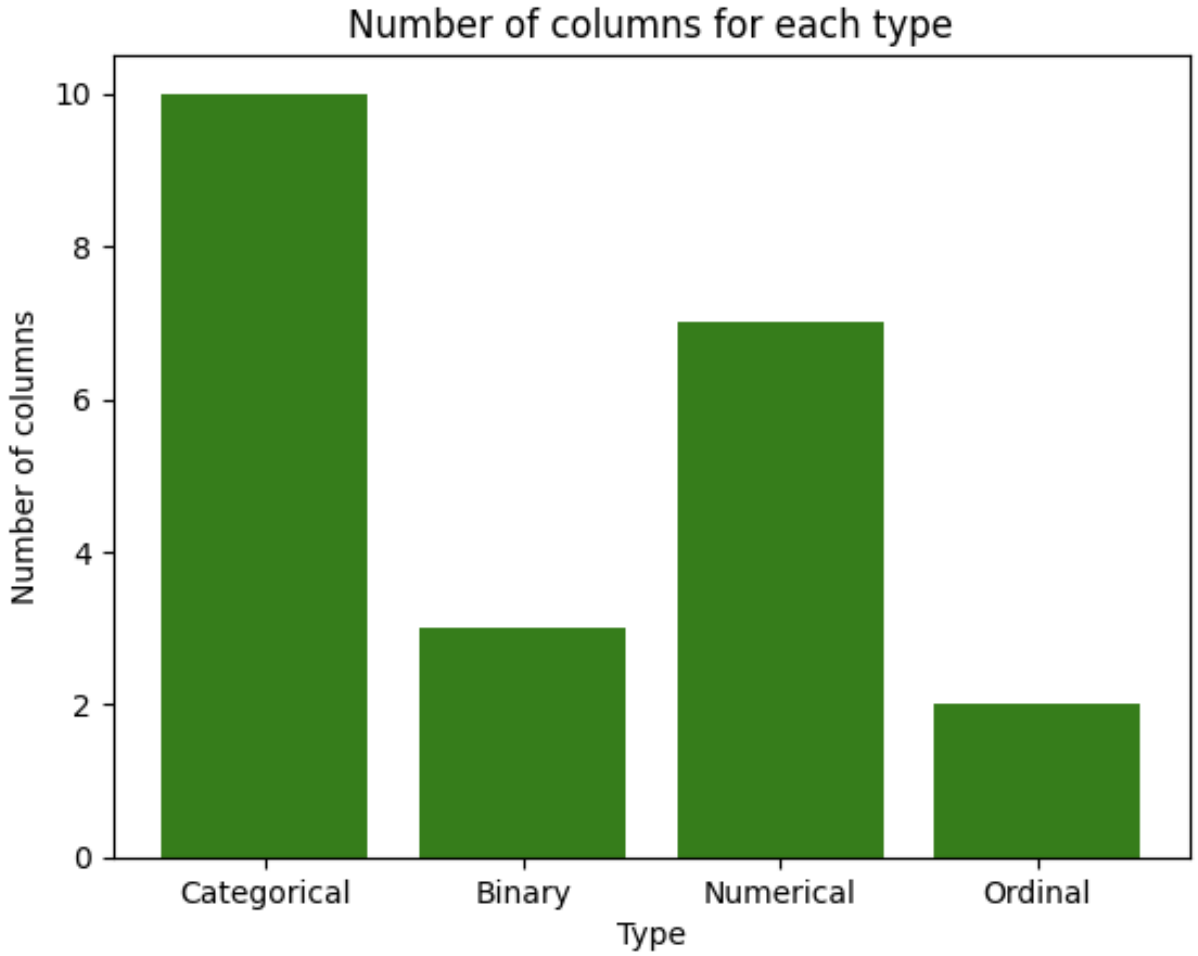

Figure 3: Types of Features in Dataset

1.3.3 Correlation Analysis

We perform correlation analysis between numeric attributes to identify redundant attributes. For categorical attributes, we use the Chi-Squared test to check for independence.

Correlation Matrix for Numerical Values

Finally, for a much better understanding of this dataset, we will use a correlation matrix for the numerical values in order to potentially find redundancy.

What we observed: *Low correlations:*

- **fnl** and **hpw** have a correlation of -0.026, indicating a very weak negative correlation.
- **fnl** and **gain** have a correlation of 0.004, indicating almost no linear relationship.

High correlations:

- **gain** and **prod** with a correlation of 1, indicating a perfect positive correlation. This suggests that these two features likely represent similar or identical information, making one of them potentially redundant.

- **hpw** and **gain** with a correlation of 0.097, indicating a weak positive correlation.

Negative correlations:

- **years** and **fnl** have a correlation of -0.07, indicating a very weak negative correlation.
- **loss** and **years** have a correlation of 0.046, which is also a weak positive correlation.

Summary of correlation matrix: The perfect correlation between **gain** and **prod** suggests that one of these features might be redundant. This redundancy should be considered during feature selection or model training to avoid multicollinearity.

Most features have weak linear relationships with each other, implying that they might contribute unique information to the model.

The weak correlations suggest that the features are largely independent of each other, which can be beneficial for certain machine learning algorithms that assume feature independence. For example, Naive Bayes assumes that the features are independent of each other. This assumption is called the "naive" assumption, and it simplifies the computation and the model. This can be advantageous for algorithms that assume or benefit from independent features, leading to simpler, more interpretable models and potentially better performance in certain cases.

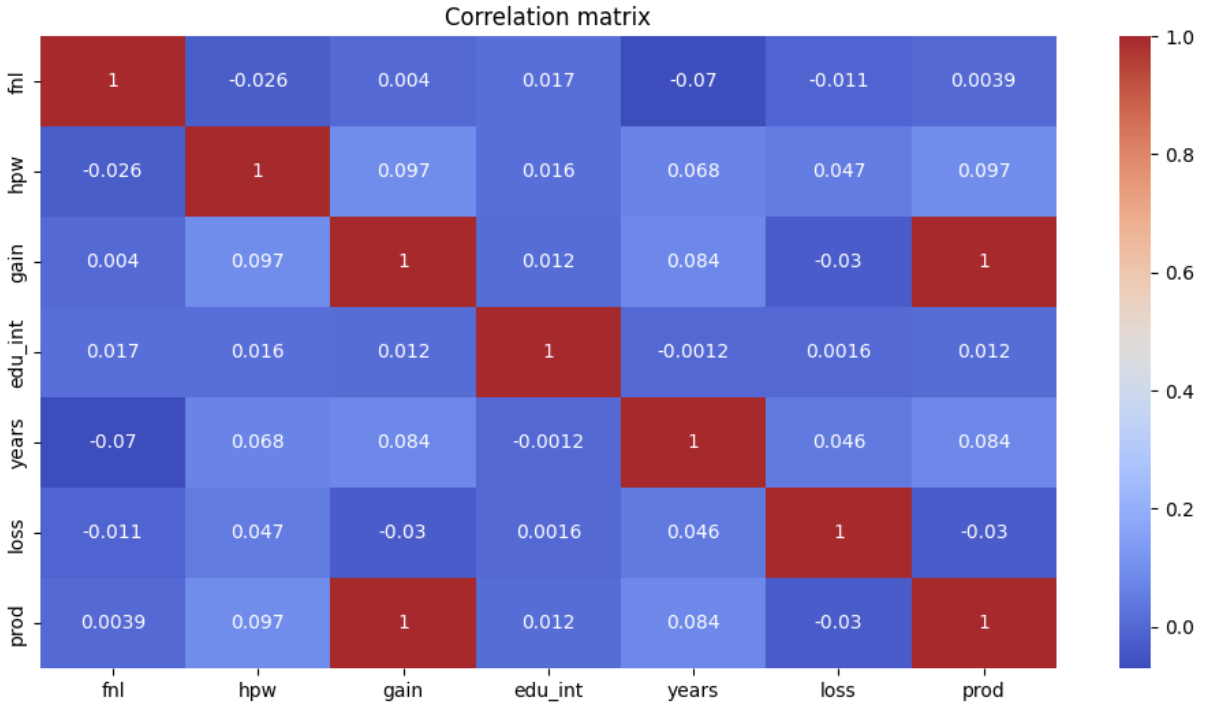

Figure 4: Numerical Features Correlation Matrix

Interpretation of Correlation Analysis for Categorical Variables

We analyzed the correlations between the categorical variables in the dataset using the Chi-Squared test and visualized the results in heatmaps.

P-Value Matrix

- **Significant Associations**:
  - Most pairs of variables have p-values close to 0, indicating statistically significant associations.
- **Non-Significant Associations**:
  - `country` and `work_type` have a p-value of 0.17, suggesting a non-significant relationship.
  - `country` and `gender` have a p-value of 0.0076, which is significant but higher compared to other pairs.

Summary

1. **Strong Associations**: Strong associations exist between `job` and `work_type`, and `relation` and `partner`.
2. **Statistical Significance**: Most associations are statistically significant.
3. **Potential Non-Significant Associations**: Some pairs, like `country` and `work_type`, do not have a significant association.

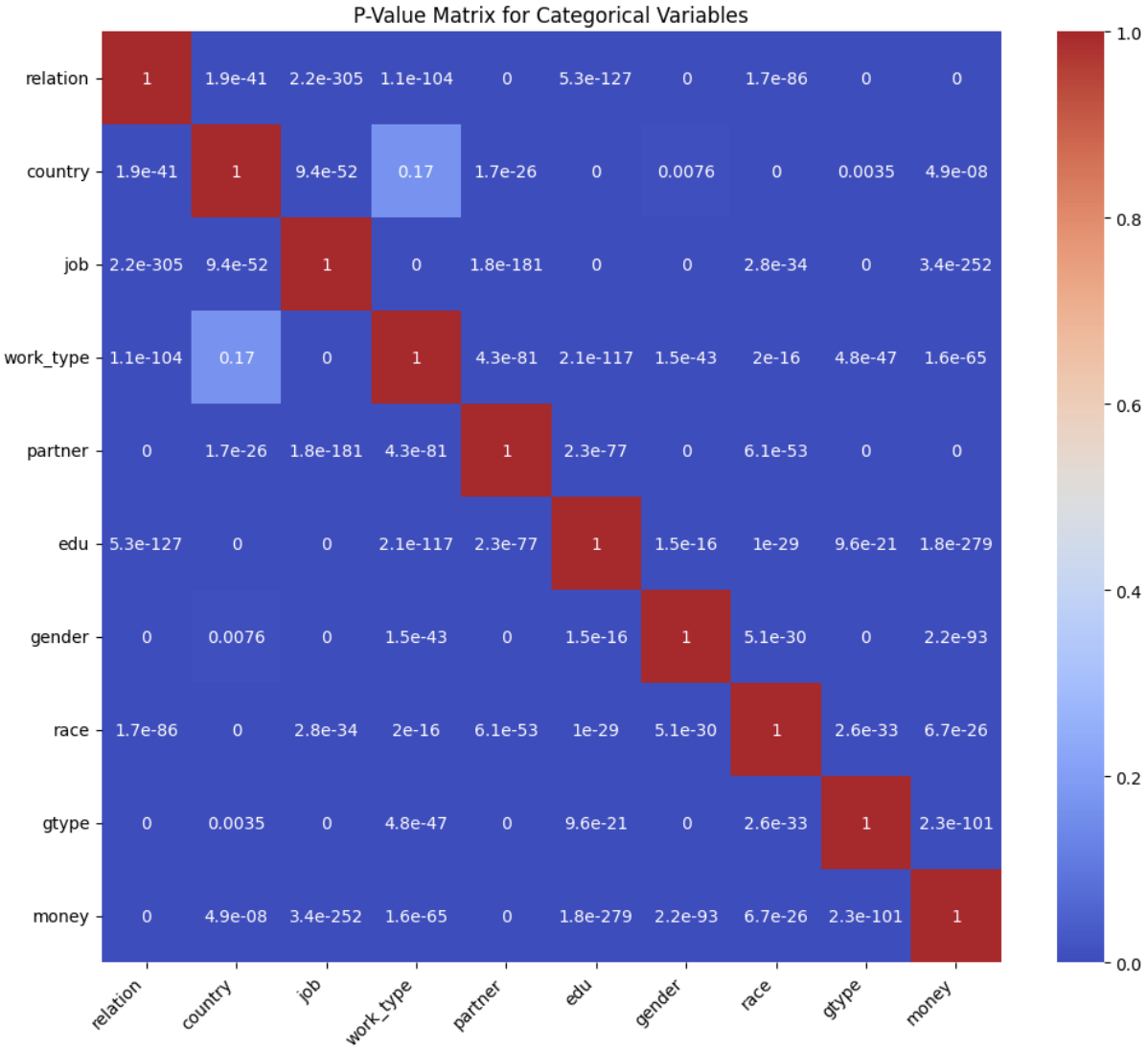

Figure 5: P-Value Correlation Matrix for Categorical Features

1.4 Data Preprocessing

In this step, we will focus on preprocessing our data:

- Handle missing values
- Encode categorical variables
- Scale numeric features
- Handle outliers
- Feature selection
- Split the data into features and target

This process will focus on all datasets (train, test, full). We will start by dropping some columns that are too correlated with others.

1.4.1 Handling Missing Values

We identify attributes with missing values and use appropriate imputation methods, such as mean, median, or mode for univariate imputation, and regression methods for multivariate imputation.

1.4.2 Handling Outliers

We detect outliers using the interquartile range (IQR) method and replace them using imputation techniques.

1.4.3 Standardization

Numeric attributes are standardized to ensure they have similar scales, which is important for algorithms like logistic regression.

1.4.4 Encoding Categorical Variables

Categorical variables are encoded using techniques such as one-hot encoding to convert them into a format suitable for machine learning algorithms.

1.4.5 Feature Selection

We analyze the correlation matrix to identify and drop columns that are too correlated with others to avoid multicollinearity. This helps in selecting the most relevant features for model training.

1.4.6 Data Splitting

We split the data into features (X) and target (y) for both training and testing sets.

Observation of Plots Before preprocessing, we will observe and plot the distribution of the features to understand their initial state. After preprocessing, we will re-plot these distributions to observe the changes and ensure that the preprocessing steps have been effective.

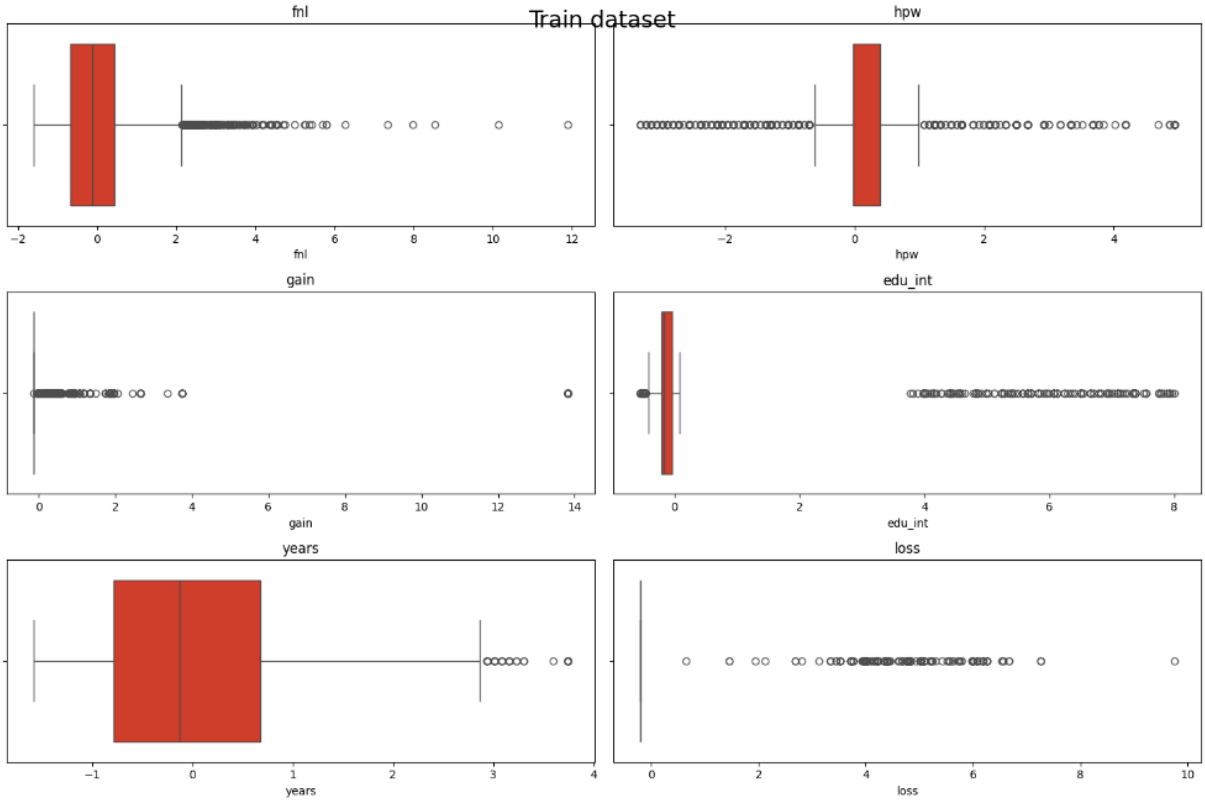

Figure 6: Train Dataset Before Preprocessing

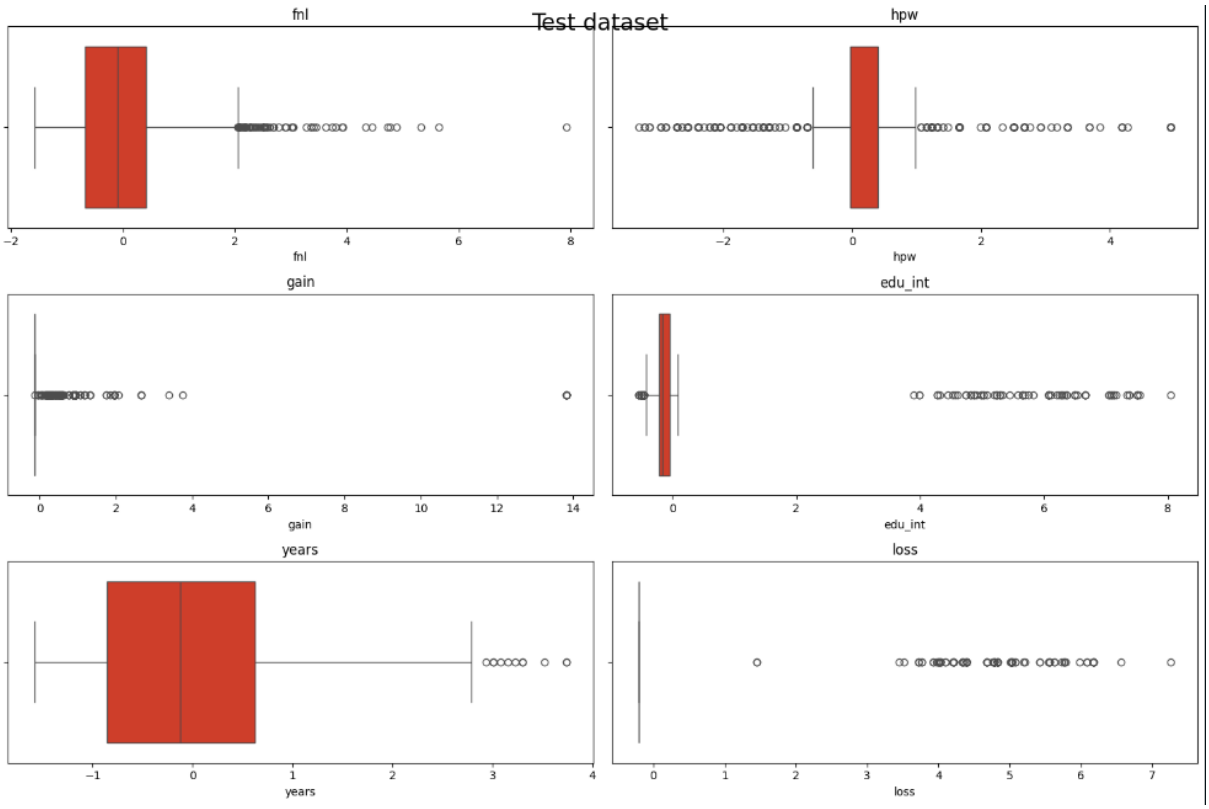

Figure 7: Test Dataset Before Preprocessing

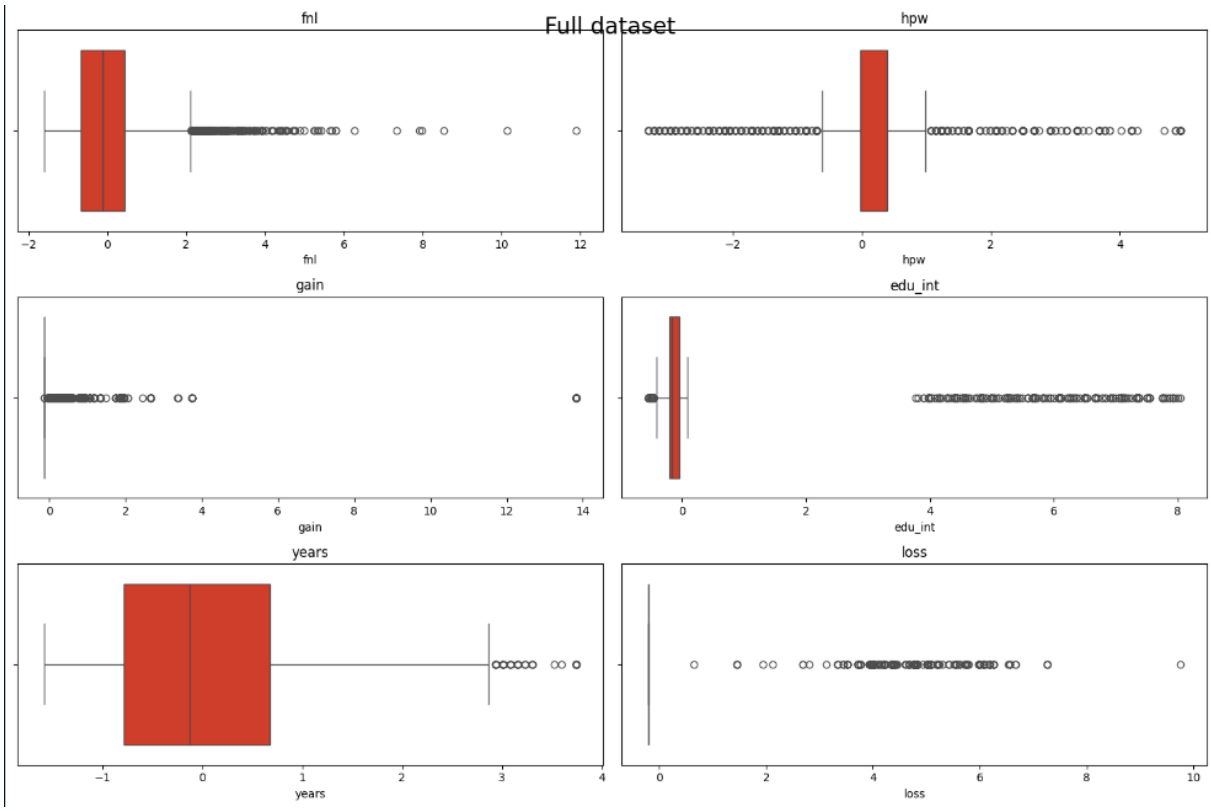

Figure 8: Full Dataset Before Preprocessing

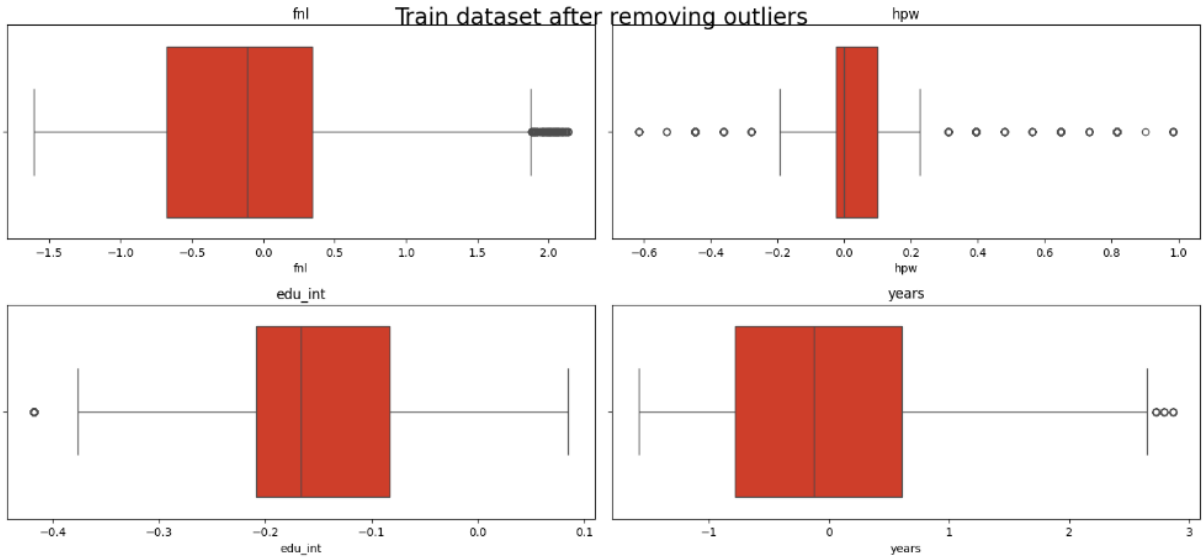

Figure 9: Train Dataset After Preprocessing

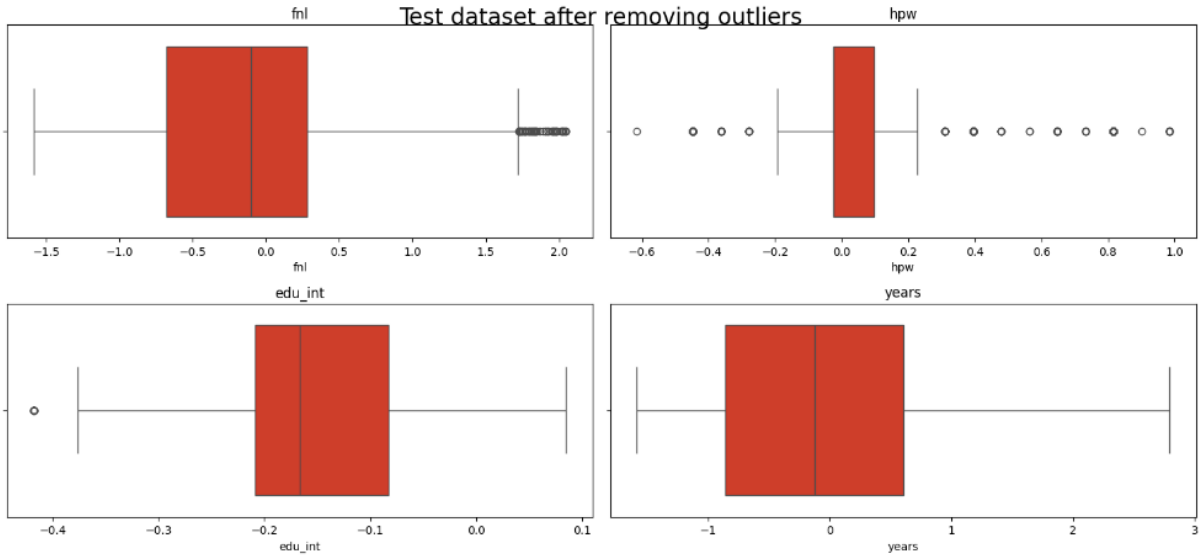

Figure 10: Test Dataset After Preprocessing

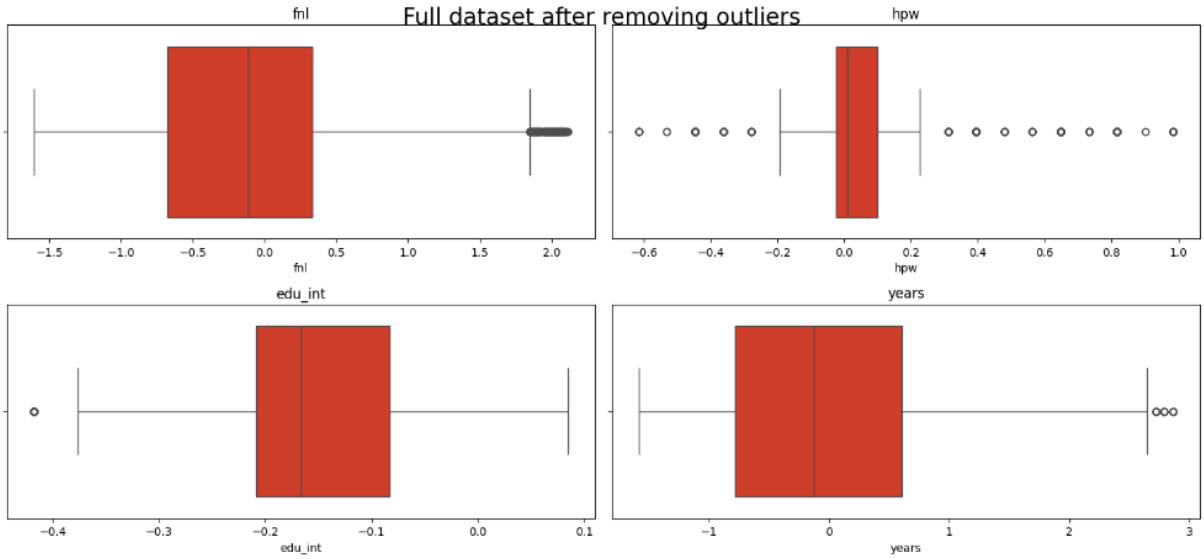

Figure 11: Full Dataset After Preprocessing

1.5 Model Implementation

1.5.1 Logistic Regression

Manual Implementation We manually implement logistic regression using gradient descent optimization.

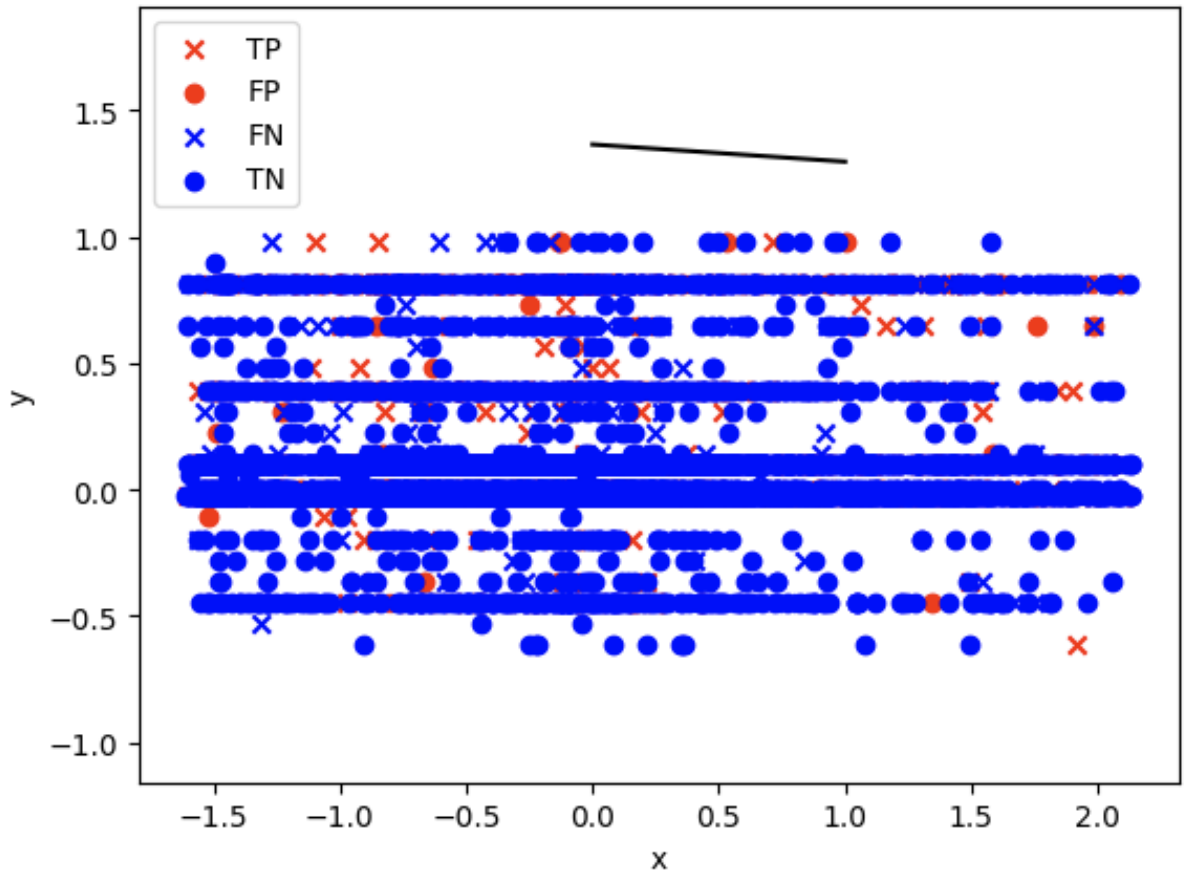

Figure 12: Predictions Hits For Manual LR

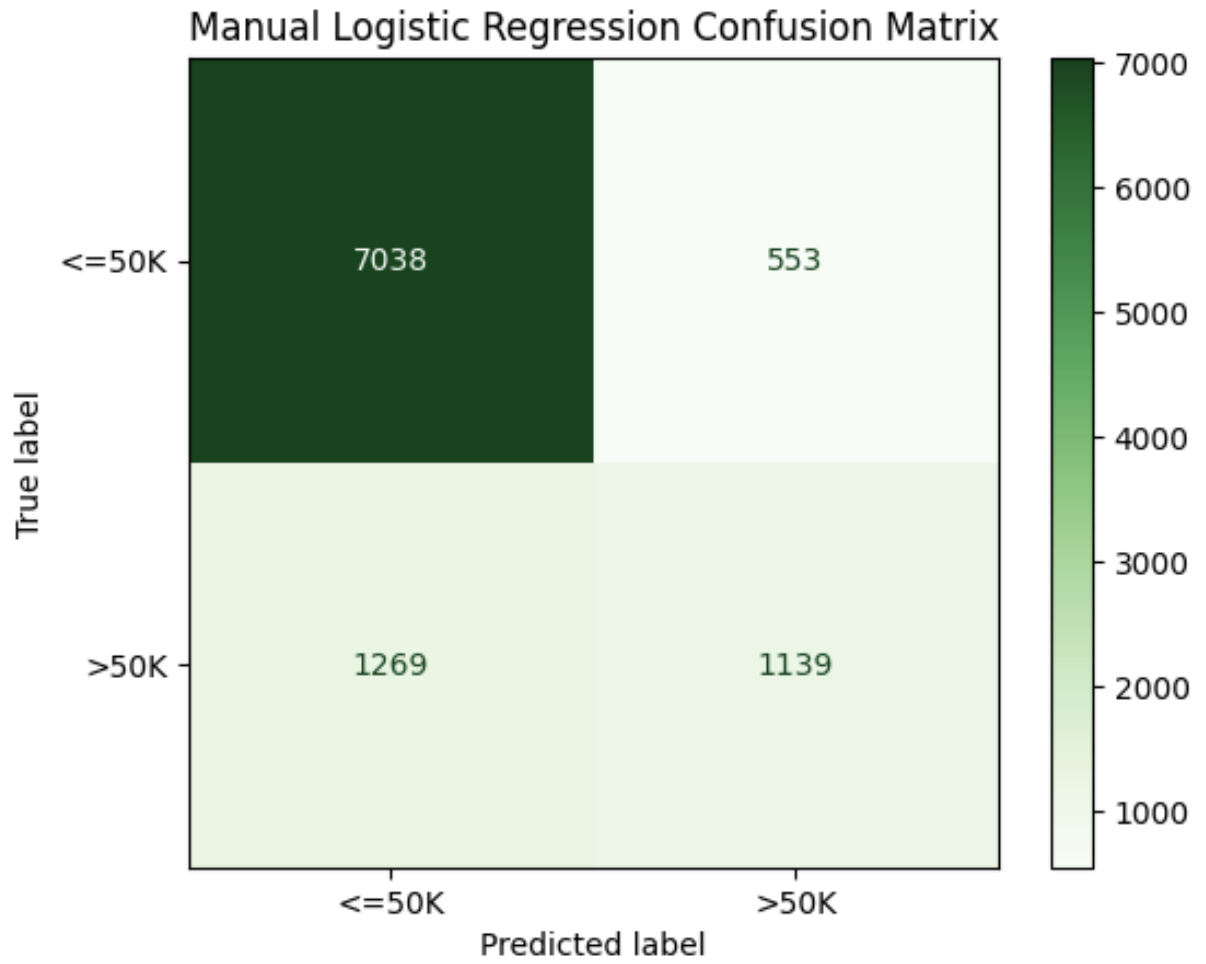

Figure 13: Manual Logistic Regression Confusion Matrix

Using scikit-learn We use scikit-learn to train and evaluate the logistic regression model.

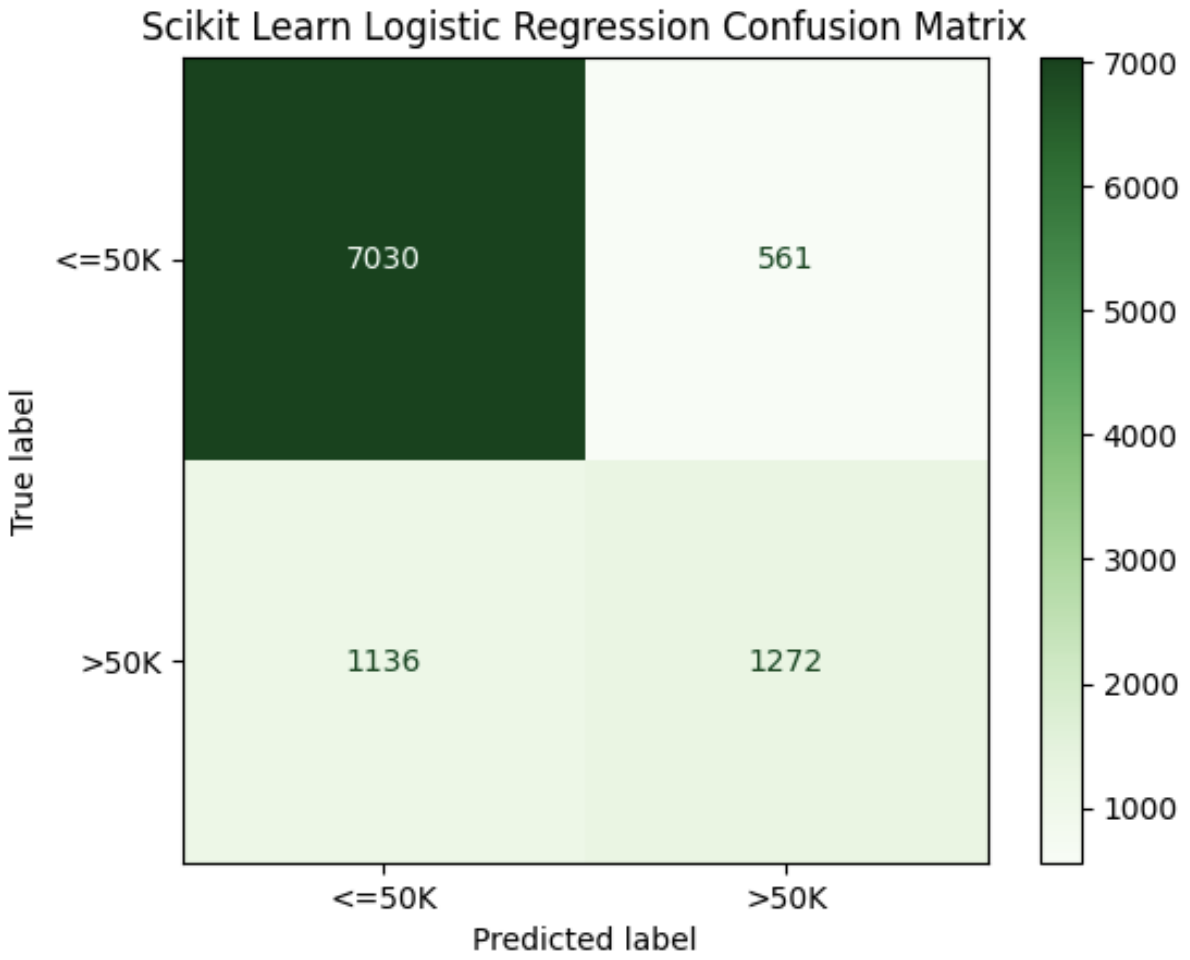

Figure 14: SK Logistic Regression Confusion Matrix

1.5.2 Multi-Layer Perceptron (MLP)

Manual Implementation We manually implement an MLP model using standard backpropagation.

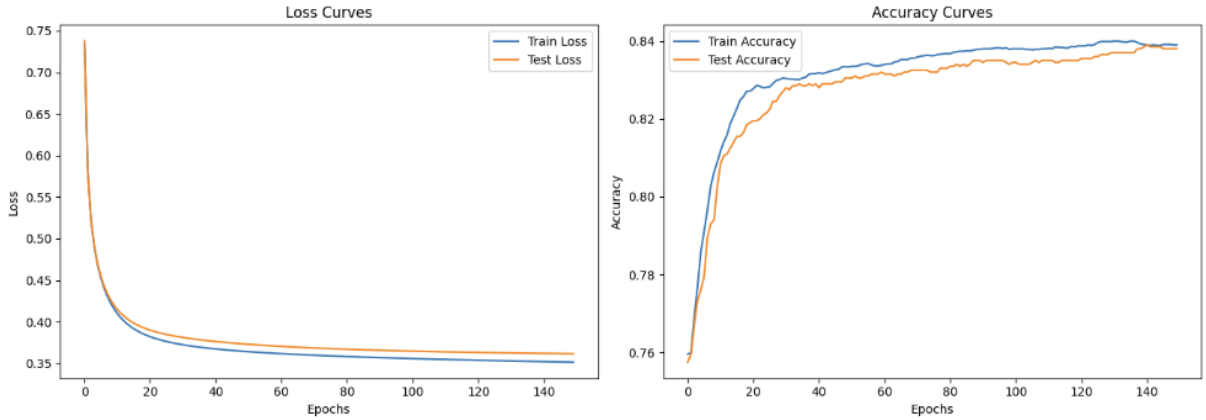

Figure 15: Errors and Accuracy Curves Manual MLP

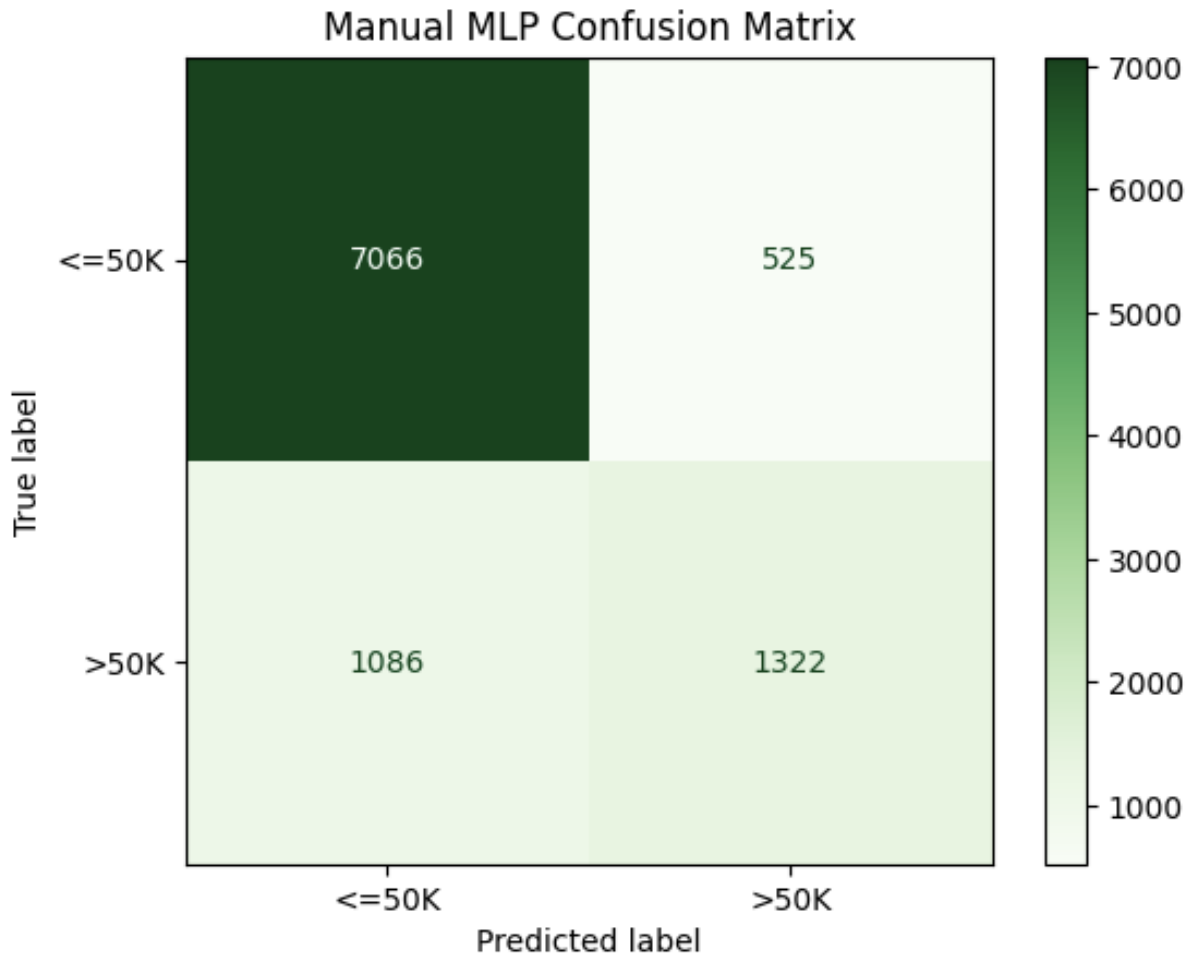

Figure 16: Manual MLP Confusion Matrix

Using scikit-learn We use scikit-learn to train and evaluate the MLP classifier.

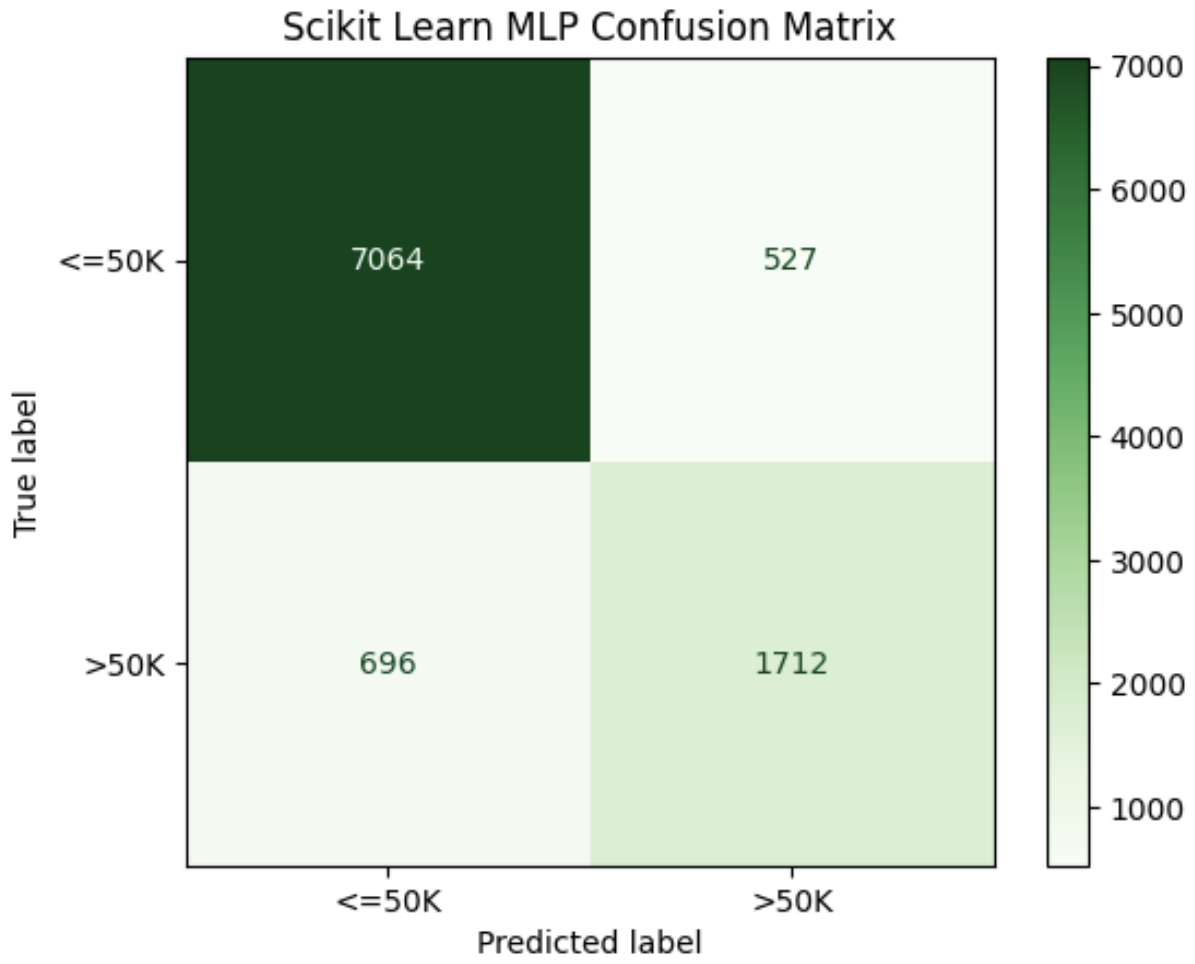

Figure 17: SK MLP Confusion Matrix

1.6 Evaluation

1.6.1 Hyperparameter Tuning

We document the final set of hyperparameters used for each model.

- **Manual Logistic Regression**:
  - Learning Rate (lr): 0.15
  - Number of Epochs (epochs_no): 1500
- **Scikit-Learn Logistic Regression**:
  - Max Iterations (max_iter): 1000
  - Solver: 'lbfgs'
- **Manual MLP**:
  - Number of Epochs: 1000
  - Learning Rate: 0.01
- **Scikit-Learn MLPClassifier**:

- Hidden Layer Sizes: (300, 10)
- Max Iterations (max_iter): 1200
- Random State: 3
- Learning Rate (learning_rate): 'adaptive'
- Learning Rate Initialization (learning_rate_init): 0.02
- Solver: 'sgd'

1.7 Analysis

Train Accuracy:

- The Scikit-Learn MLP model achieves the highest train accuracy (0.894237), indicating that it fits the training data very well.
- The Manual Logistic Regression model has the lowest train accuracy (0.818352), suggesting it might be underfitting compared to the other models.

Test Accuracy:

- The Manual MLP model shows the highest test accuracy (0.838000), suggesting it generalizes well to unseen data.
- The Manual Logistic Regression and Scikit-Learn MLP models both have a test accuracy of 0.815500, which is slightly lower.

Precision:

- Precision is highest for the Scikit-Learn MLP model (0.764627), indicating it has the least false positives compared to the other models.
- The Manual Logistic Regression model has the lowest precision (0.673168).

Recall:

- Recall is also highest for the Scikit-Learn MLP model (0.710963), suggesting it successfully identifies a higher number of true positives.
- The Manual Logistic Regression model has the lowest recall (0.473007), indicating it misses more actual positives.

F1 Score:

- The Scikit-Learn MLP model achieves the highest F1 Score (0.736819), which is a balance between precision and recall.
- The Manual Logistic Regression model has the lowest F1 Score (0.555610), reflecting its lower precision and recall.

Overall, the Scikit-Learn MLP model demonstrates the best performance across most metrics, including train accuracy, precision, recall, and F1 score, indicating it is a robust model for this dataset. The Manual MLP also performs well, particularly in terms of test accuracy. The logistic regression models (both manual and Scikit-Learn) show reasonable performance but are outperformed by the MLP models, especially the Scikit-Learn MLP. This comparison highlights the effectiveness of more complex models like MLPs in capturing intricate patterns in the data, provided they are properly tuned and regularized.

	Manual Logistic Regression	Scikit Learn Logistic Regression	Manual MLP	Scikit Learn MLP
Train accuracy	0.818352	0.830479	0.838980	0.894237
Test accuracy	0.815500	0.829500	0.838000	0.811500
Precision	0.673168	0.693944	0.715755	0.764627
Recall	0.473007	0.528239	0.549003	0.710963
F1 score	0.555610	0.599859	0.621387	0.736819

Figure 18: Comparison For Salary Dataset

2 Part II: Stroke Prediction

2.1 Introduction

In this assignment, we explore common tasks in the field of artificial intelligence and machine learning, particularly focusing on data visualization, feature extraction, and model evaluation for the stroke prediction dataset.

2.2 Dataset Description

The dataset contains medical and lifestyle information about 5110 individuals. The goal is to predict whether a person is likely to have a stroke.

Attribute Name	Data Type	Description
mean_blood_sugar_level	Numeric	Average blood sugar level
cardiovascular_issues	Categorical	Presence of cardiovascular issues (0, 1)
job_category	Categorical	Employment category (e.g., child, entrepreneurial, etc.)
body_mass_indicator	Numeric	Body mass index
sex	Categorical	Gender (F, M)
tobacco_usage	Categorical	Tobacco usage (ex-smoker, smoker, non-smoker)
high_blood_pressure	Categorical	High blood pressure (0, 1)
married	Categorical	Marital status (Y, N)
living_area	Categorical	Type of living area (City, Countryside)
years_old	Numeric	Age in years
chaotic_sleep	Categorical	Irregular sleep pattern (0, 1)
analysis_results	Numeric	Medical analysis results
biological_age_index	Numeric	Biological age index
cerebrovascular_accident	Categorical	Stroke indicator (0, 1)

Table 2: Attributes of the Stroke Prediction Dataset

2.3 Exploratory Data Analysis (EDA)

2.3.1 Attribute Analysis

Numeric Attributes For numeric attributes, we calculate and present the number of non-missing values, mean, standard deviation, minimum, 25th percentile, median, 75th percentile, and maximum values. Additionally, we provide boxplots to visualize the range of values. The dataset contains numerical values with the following characteristics:

`mean_blood_sugar_level`

- **Count**: 5110
- **Mean**: 106.15
- **Standard Deviation**: 45.28
- **Min**: 55.12

- **25th Percentile**: 77.25
- **Median (50th Percentile)**: 91.89
- **75th Percentile**: 114.09
- **Max**: 271.74
- **Interpretation**: The mean blood sugar level is 106.15, with values ranging from 55.12 to 271.74. The distribution appears to be right-skewed with a higher standard deviation.

`cardiovascular_issues`

- **Count**: 5110
- **Mean**: 0.05
- **Standard Deviation**: 0.23
- **Min**: 0
- **25th Percentile**: 0
- **Median (50th Percentile)**: 0
- **75th Percentile**: 0
- **Max**: 1
- **Interpretation**: Cardiovascular issues are rare in this dataset, with a mean of 0.05 and most values being 0 (no issues).

`body_mass_indicator`

- **Count**: 4909
- **Mean**: 28.89
- **Standard Deviation**: 7.85
- **Min**: 10.3
- **25th Percentile**: 23.5
- **Median (50th Percentile)**: 28.1
- **75th Percentile**: 33.1
- **Max**: 97.6
- **Interpretation**: The body mass indicator shows a mean of 28.89, indicating an average BMI in the overweight range. There is a wide range with some outliers.

high_blood_pressure

- **Count**: 5110
- **Mean**: 0.10
- **Standard Deviation**: 0.30
- **Min**: 0
- **25th Percentile**: 0
- **Median (50th Percentile)**: 0
- **75th Percentile**: 0
- **Max**: 1
- **Interpretation**: High blood pressure is uncommon in this dataset, with a mean of 0.10 and most individuals not having high blood pressure.

years_old

- **Count**: 5110
- **Mean**: 46.57
- **Standard Deviation**: 26.59
- **Min**: 0.08
- **25th Percentile**: 26
- **Median (50th Percentile)**: 47
- **75th Percentile**: 63.75
- **Max**: 134
- **Interpretation**: The age distribution is broad, with ages ranging from 0.08 to 134 years. The mean age is 46.57, suggesting a wide age range in the dataset.

chaotic_sleep

- **Count**: 5110
- **Mean**: 0.05
- **Standard Deviation**: 0.23
- **Min**: 0
- **25th Percentile**: 0
- **Median (50th Percentile)**: 0
- **75th Percentile**: 0

- **Max**: 1
- **Interpretation**: Chaotic sleep patterns are infrequent, with a mean of 0.05 and most individuals reporting no issues.

`analysis_results`

- **Count**: 4599
- **Mean**: 323.52
- **Standard Deviation**: 101.58
- **Min**: 104.83
- **25th Percentile**: 254.65
- **Median (50th Percentile)**: 301.03
- **75th Percentile**: 362.82
- **Max**: 756.81
- **Interpretation**: The analysis results show a mean value of 323.52, with values ranging widely, indicating diverse health measurement outcomes.

`biological_age_index`

- **Count**: 5110
- **Mean**: 134.78
- **Standard Deviation**: 50.40
- **Min**: -15.11
- **25th Percentile**: 96.71
- **Median (50th Percentile)**: 136.37
- **75th Percentile**: 172.51
- **Max**: 266.99
- **Interpretation**: The biological age index has a mean of 134.78, with a broad range of values indicating variability in biological aging among individuals.

`cerebrovascular_accident (target variable)`

- **Count**: 5110
- **Mean**: 0.05
- **Standard Deviation**: 0.22
- **Min**: 0
- **25th Percentile**: 0
- **Median (50th Percentile)**: 0
- **75th Percentile**: 0
- **Max**: 1
- **Interpretation**: Cerebrovascular accidents are rare, with a mean of 0.05 and most individuals not having experienced a stroke.

Categorical Attributes For categorical attributes, we calculate and present the number of non-missing values and the number of unique values. Histograms are used to visualize the distribution of values for each categorical attribute.

job_category

- **Count**: 5110
- **Unique**: 5
- **Top**: private_sector
- **Frequency**: 2925
- **Interpretation**: The most common job category is 'private_sector', with 2925 occurrences out of 5110.

sex

- **Count**: 5110
- **Unique**: 2
- **Top**: F
- **Frequency**: 2994
- **Interpretation**: The dataset has more females (F) with 2994 occurrences out of 5110.

tobacco_usage

- **Count**: 5110
- **Unique**: 4
- **Top**: non-smoker
- **Frequency**: 1892
- **Interpretation**: The most common tobacco usage status is 'non-smoker', with 1892 occurrences out of 5110.

married

- **Count**: 4599
- **Unique**: 2
- **Top**: Y
- **Frequency**: 3014
- **Interpretation**: The majority of the individuals are married (Y), with 3014 occurrences out of 4599.

living_area

- **Count**: 5110
- **Unique**: 2
- **Top**: City
- **Frequency**: 2596
- **Interpretation**: The most common living area is 'City', with 2596 occurrences out of 5110.

2.3.2 Class Balance Analysis

We create bar plots to show the frequency of each class in the training and test datasets. This helps to understand if there is any class imbalance that could affect model performance.

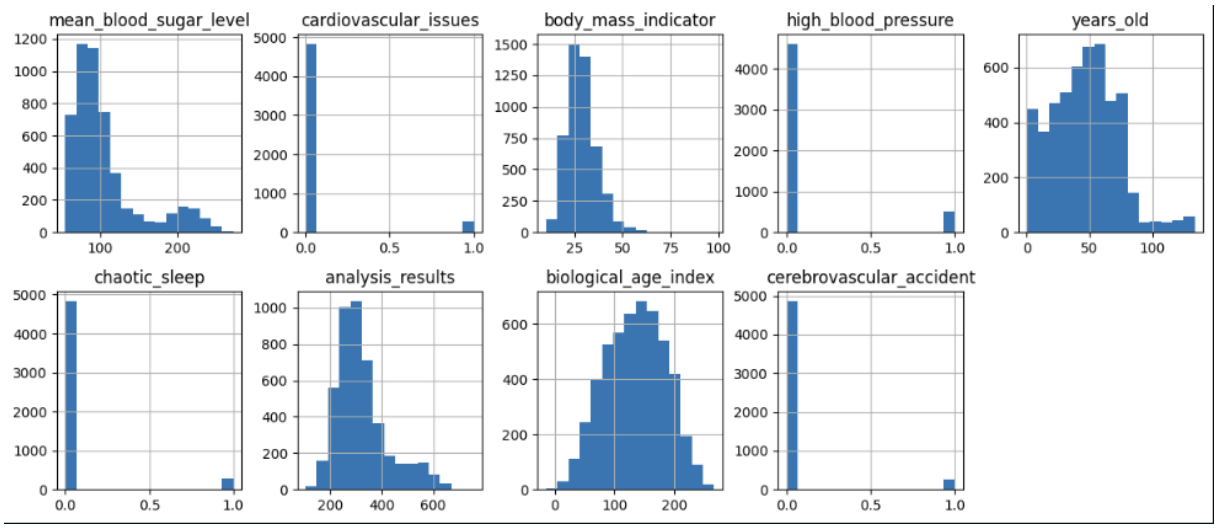

Figure 19: Numerical Features

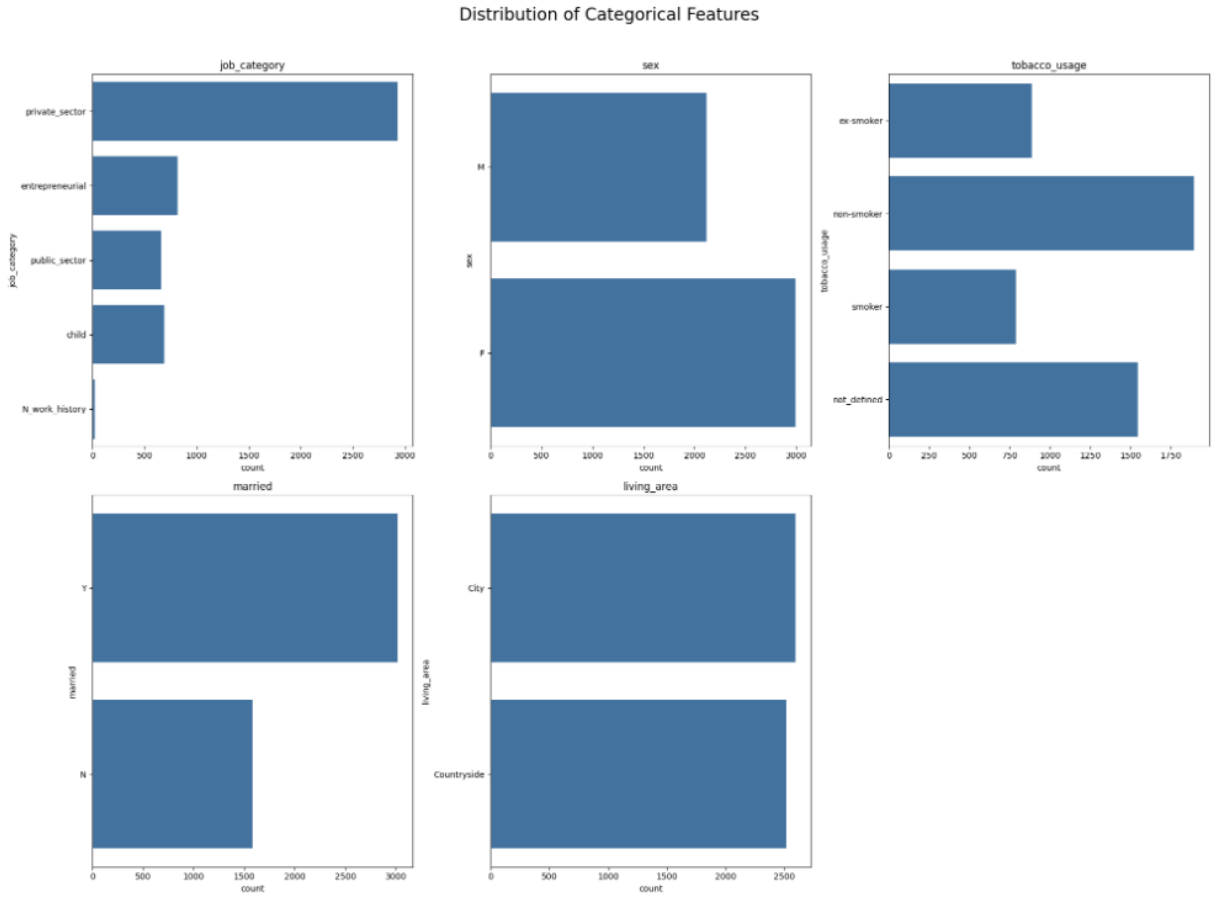

Figure 20: Categorical Features

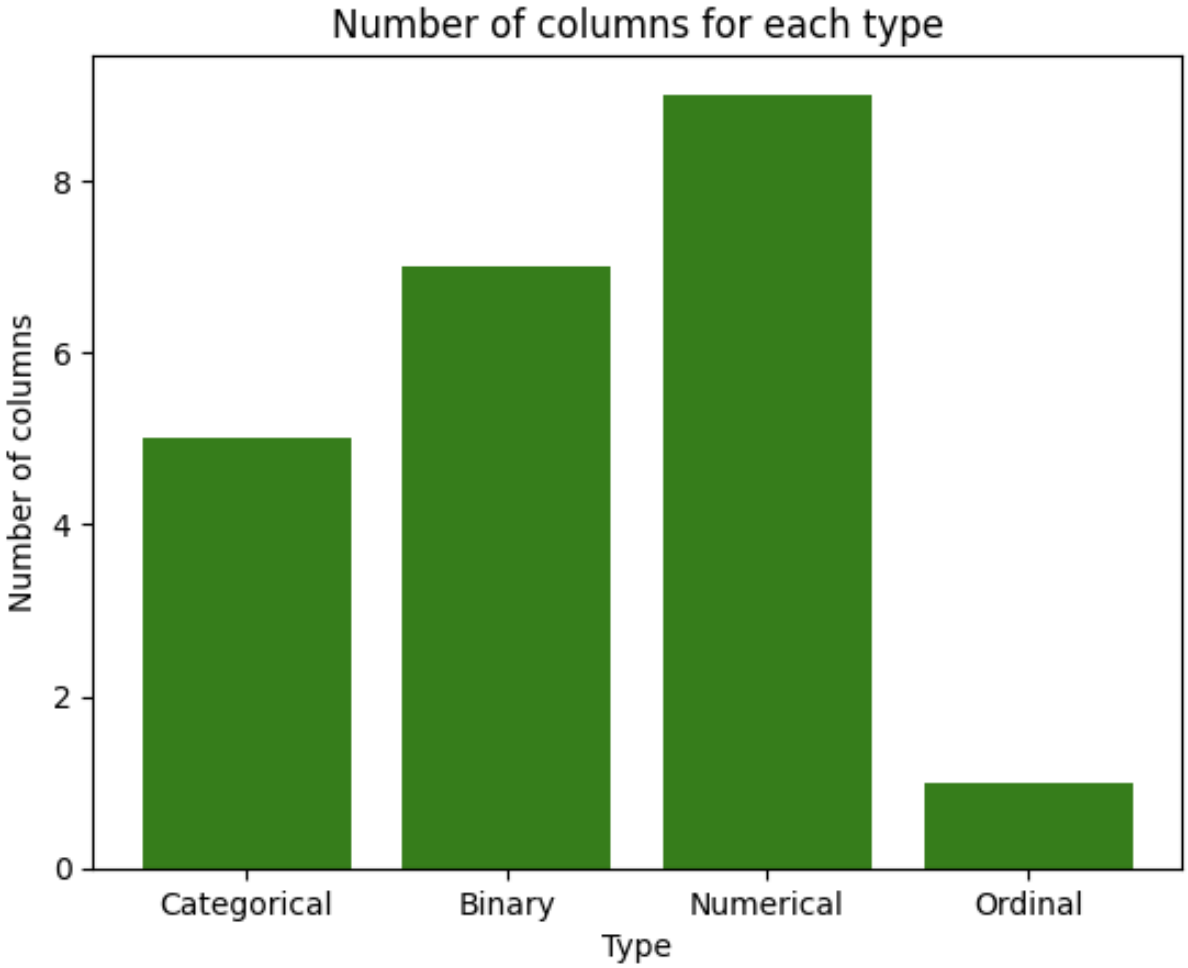

Figure 21: Types of Data in Dataset

2.3.3 Correlation Analysis

We perform correlation analysis between numeric attributes to identify redundant attributes. For categorical attributes, we use the Chi-Squared test to check for independence.

Finally, for a much better understanding of this dataset, we will use a correlation matrix for the numerical values in order to potentially find redundancy. What we observed:

Low correlations

- **mean_blood_sugar_level** and **cardiovascular_issues** have a correlation of 0.16, indicating a weak positive correlation.
- **body_mass_indicator** and **cardiovascular_issues** have a correlation of 0.041, indicating almost no linear relationship.

High correlations

- **mean_blood_sugar_level** and **analysis_results** with a correlation of 0.89, indicating a very strong positive correlation. This suggests that these two features likely represent similar information, making one of them potentially redundant.

- **years_old** and **biological_age_index** with a correlation of 0.71, indicating a strong positive correlation.

Negative correlations

- **cardiovascular_issues** and **chaotic_sleep** have a correlation of -1, meaning they perform the same on the dataset.
- **cerebrovascular_accident** and **body_mass_indicator** have a correlation of 0.042, which is a very weak positive correlation, almost negligible.
- **cerebrovascular_accident** and **mean_blood_sugar_level** have a correlation of 0.13, indicating a weak positive correlation.

Summary of correlation matrix

The strong correlation between mean_blood_sugar_level and analysis_results suggests that one of these features might be redundant. This redundancy should be considered during feature selection or model training to avoid multicollinearity.

Most features have weak linear relationships with each other, implying that they might contribute unique information to the model.

The weak correlations suggest that the features are largely independent of each other, which can be beneficial for certain machine learning algorithms that assume feature independence. For example, Naive Bayes assumes that the features are independent of each other. This assumption is called the "naive" assumption, and it simplifies the computation and the model. This can be advantageous for algorithms that assume or benefit from independent features, leading to simpler, more interpretable models and potentially better performance in certain cases.

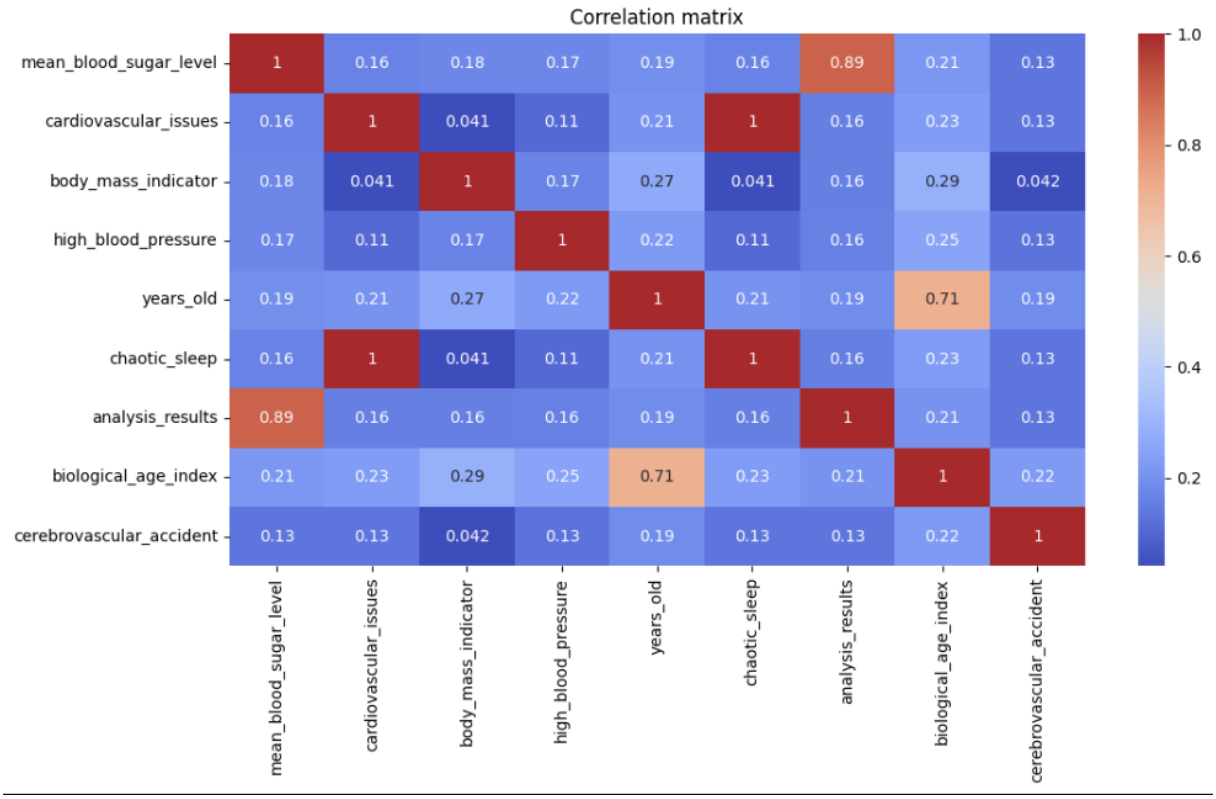

Figure 22: Correlation Matrix for Numerical Values

We analyzed the correlations between the categorical variables in the dataset using the Chi-Squared test and visualized the results in heatmaps.

P-Value Matrix

Significant Associations

- **job_category** and **sex** have a p-value of 1.5×10^{-8}, indicating a statistically significant association.
- **job_category** and **tobacco_usage** have a p-value of 3.1×10^{-290}, indicating a very strong significant association.
- **sex** and **tobacco_usage** have a p-value of 2×10^{-11}, indicating a statistically significant association.
- **tobacco_usage** and **married** have a p-value of 9.4×10^{-116}, indicating a very strong significant association.
- **married** and **living_area** have a p-value of 0.99, indicating a very strong significant association.

Non-Significant Associations

- **job_category** and **living_area** have a p-value of 0.32, suggesting a non-significant relationship.
- **sex** and **living_area** have a p-value of 0.67, suggesting a non-significant relationship.

- **tobacco_usage** and **living_area** have a p-value of 0.14, suggesting a non-significant relationship.
- **married** and **sex** have a p-value of 0.089, suggesting a non-significant relationship.

Summary

1. **Strong Associations**: There are strong associations between **job_category** and **tobacco_usage**, **sex** and **tobacco_usage**, and **tobacco_usage** and **married**.
2. **Statistical Significance**: Most associations are statistically significant.
3. **Potential Non-Significant Associations**: Some pairs, such as **job_category** and **living_area**, and **sex** and **living_area**, do not have significant associations.

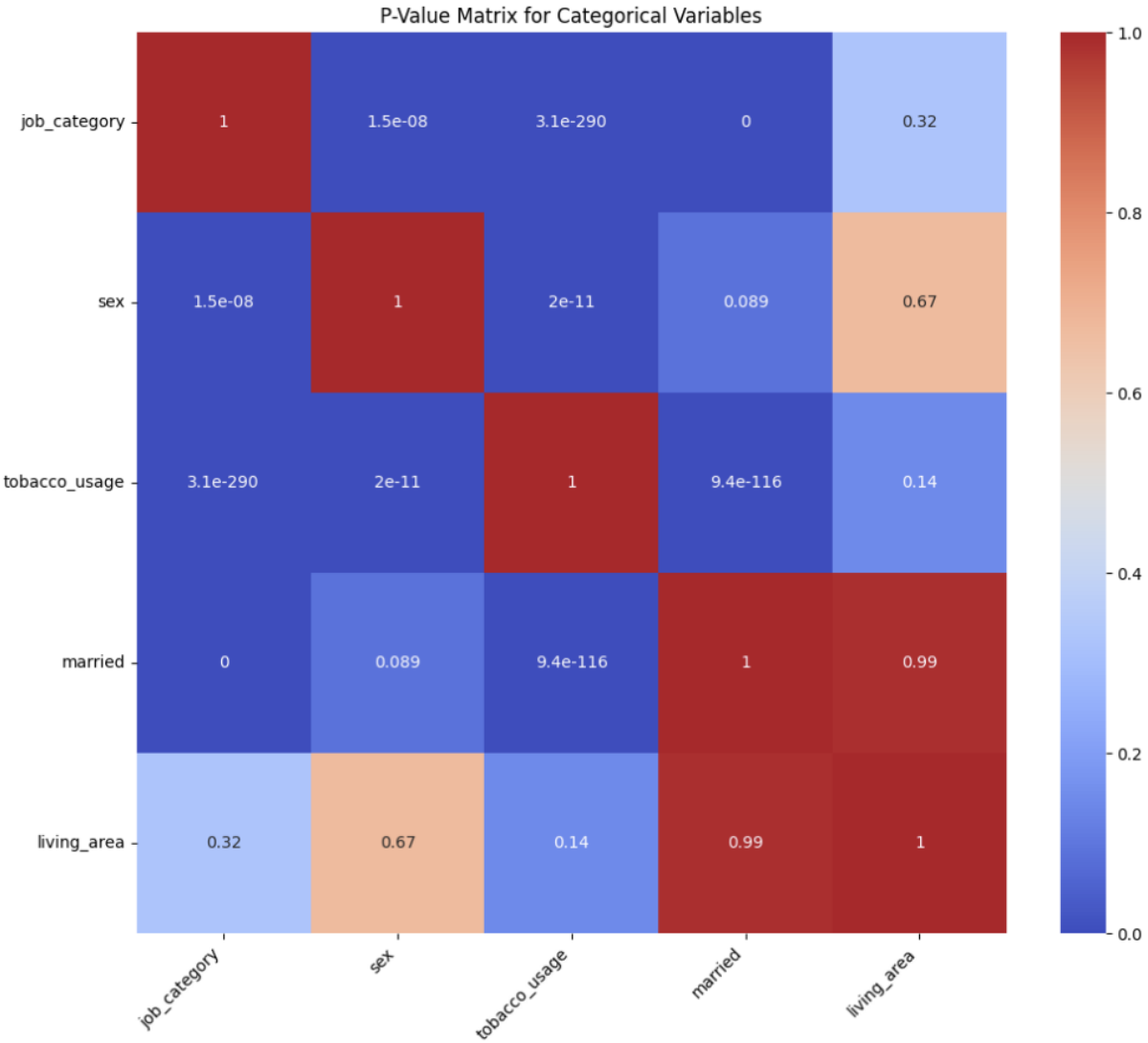

Figure 23: P-Value Matrix for Categorical Values

2.4 Data Preprocessing

In this step, we will focus on preprocessing our data:

- Handle missing values
- Encode categorical variables
- Scale numeric features
- Handle outliers
- Feature selection
- Split the data into features and target

This process will focus on all datasets (train, test, full). We will start by dropping some columns that are too correlated with others.

2.4.1 Handling Missing Values

We identify attributes with missing values and use appropriate imputation methods, such as mean, median, or mode for univariate imputation, and regression methods for multivariate imputation.

2.4.2 Handling Outliers

We detect outliers using the interquartile range (IQR) method and replace them using imputation techniques.

2.4.3 Standardization

Numeric attributes are standardized to ensure they have similar scales, which is important for algorithms like logistic regression.

2.4.4 Encoding Categorical Variables

Categorical variables are encoded using techniques such as one-hot encoding to convert them into a format suitable for machine learning algorithms.

2.4.5 Feature Selection

We analyze the correlation matrix to identify and drop columns that are too correlated with others to avoid multicollinearity. This helps in selecting the most relevant features for model training.

2.4.6 Data Splitting

We split the data into features (X) and target (y) for both training and testing sets.

Observation of Plots Before preprocessing, we will observe and plot the distribution of the features to understand their initial state. After preprocessing, we will re-plot these distributions to observe the changes and ensure that the preprocessing steps have been effective.

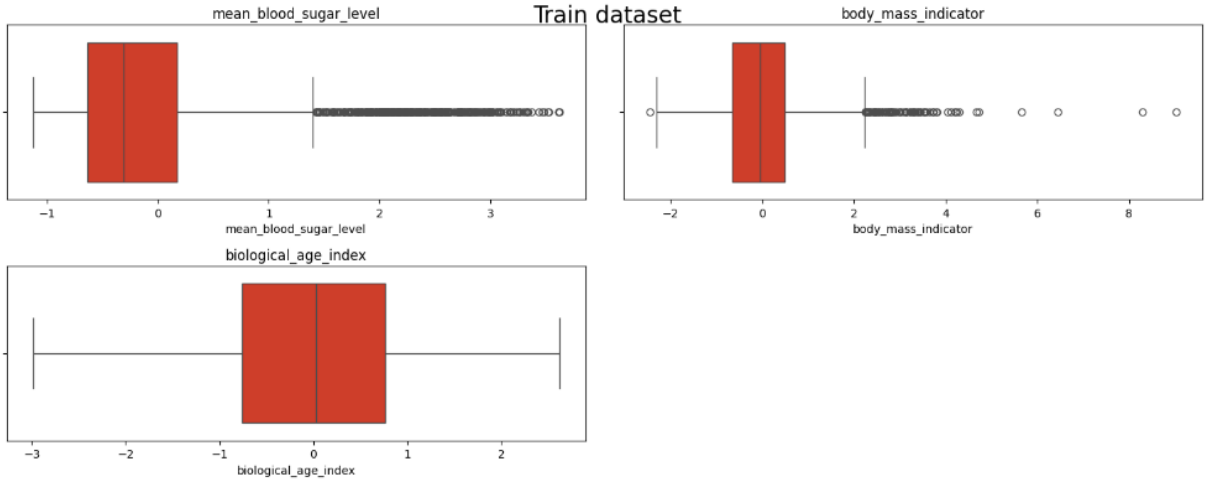

Figure 24: Train Dataset Before Preprocessing

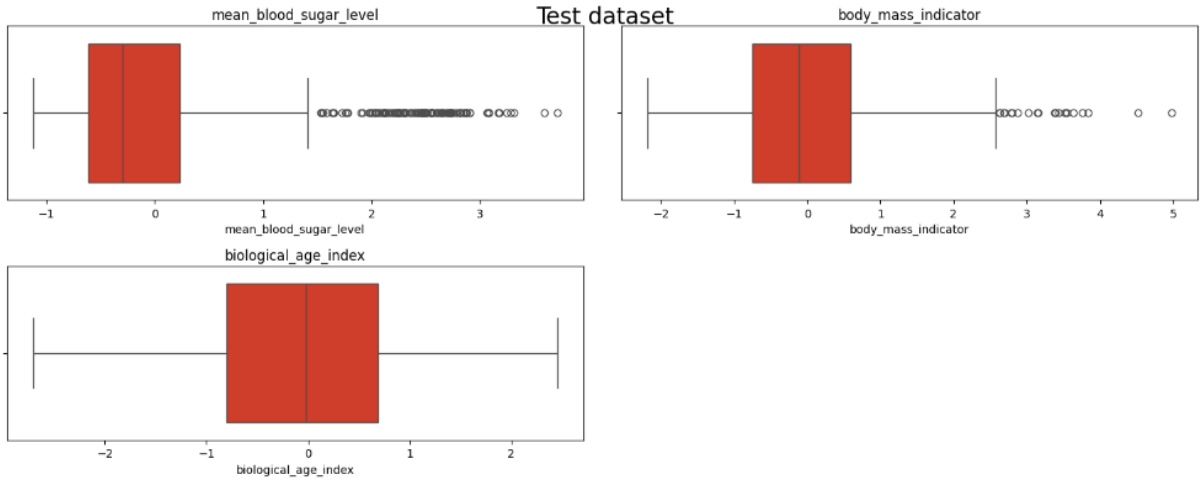

Figure 25: Test Dataset Before Preprocessing

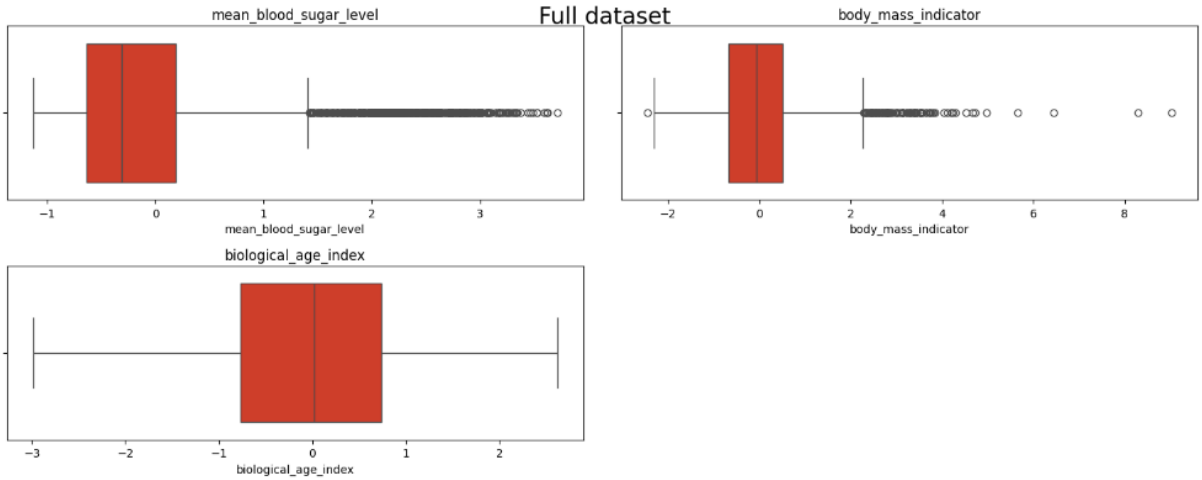

Figure 26: Full Dataset Before Preprocessing

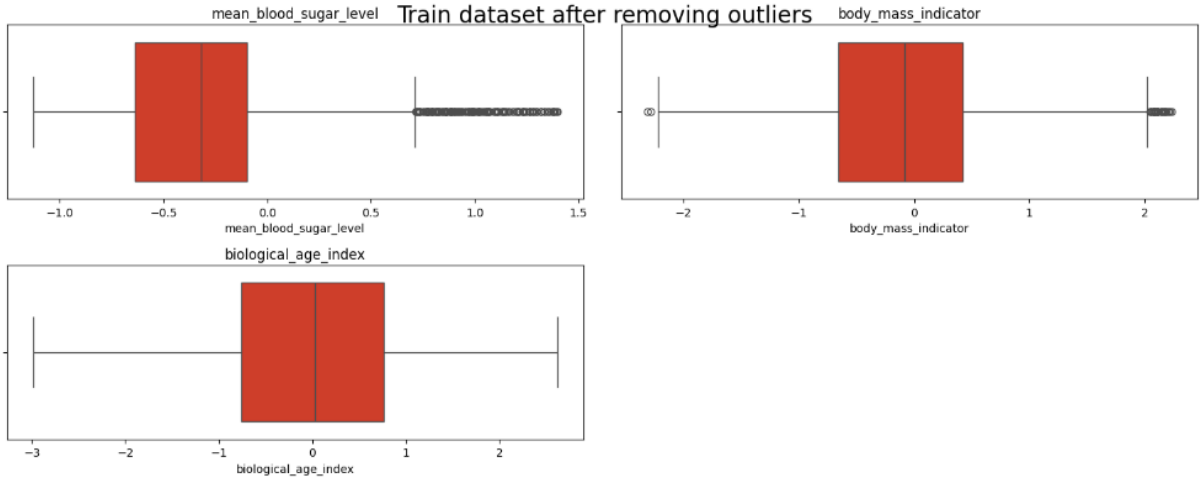

Figure 27: Train Dataset After Preprocessing

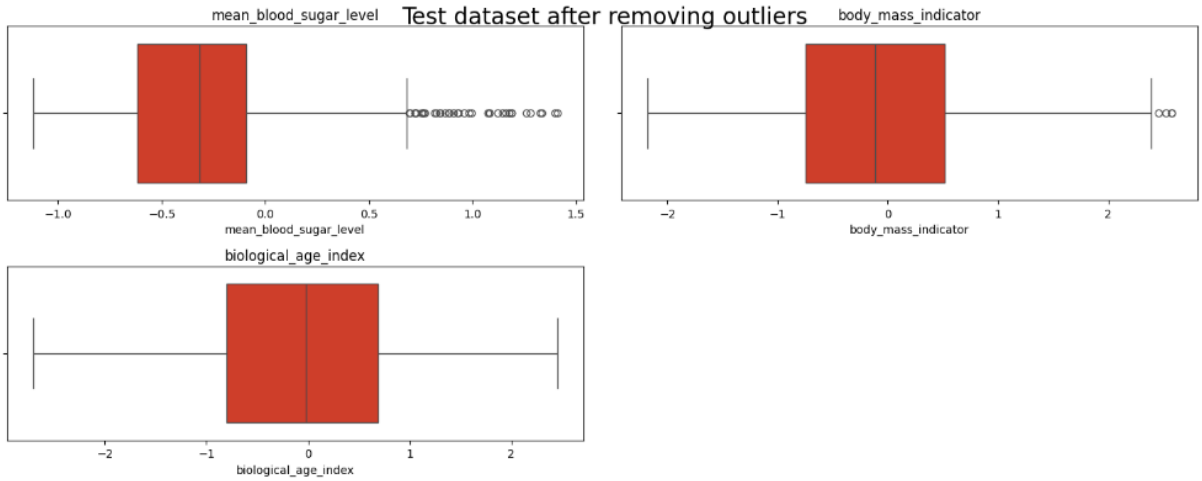

Figure 28: Test Dataset After Preprocessing

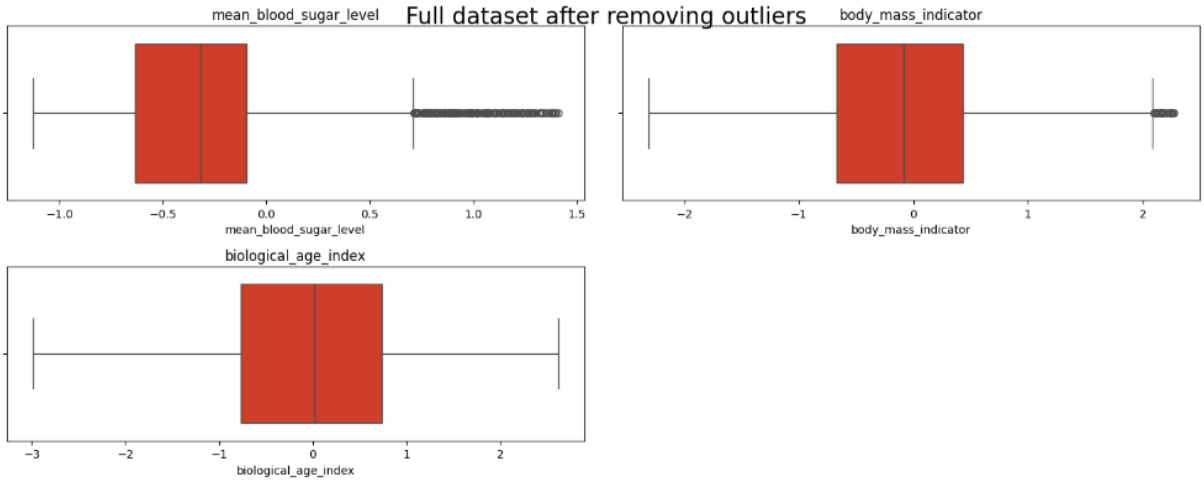

Figure 29: Full Dataset After Preprocessing

2.5 Model Implementation

2.5.1 Logistic Regression

Manual Implementation We manually implement logistic regression using gradient descent optimization.

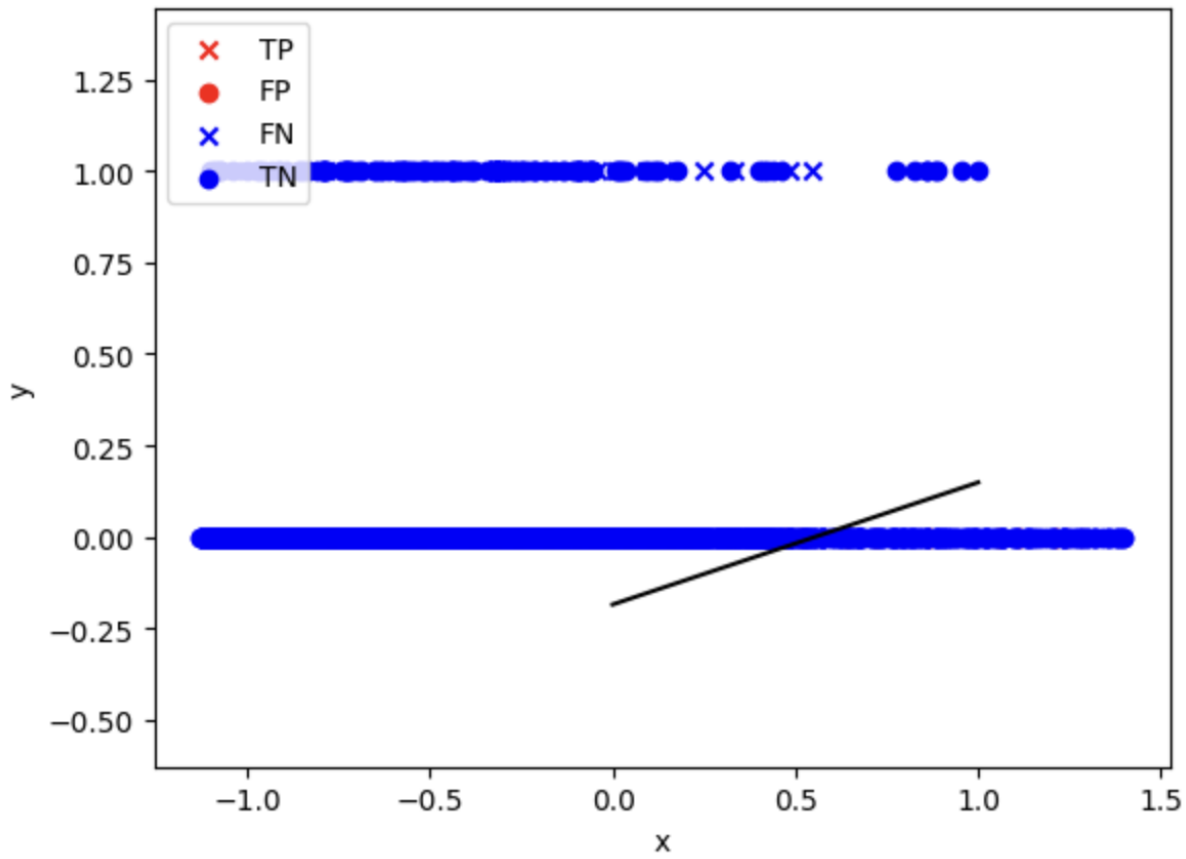

Figure 30: Predictions Hits For Manual LR

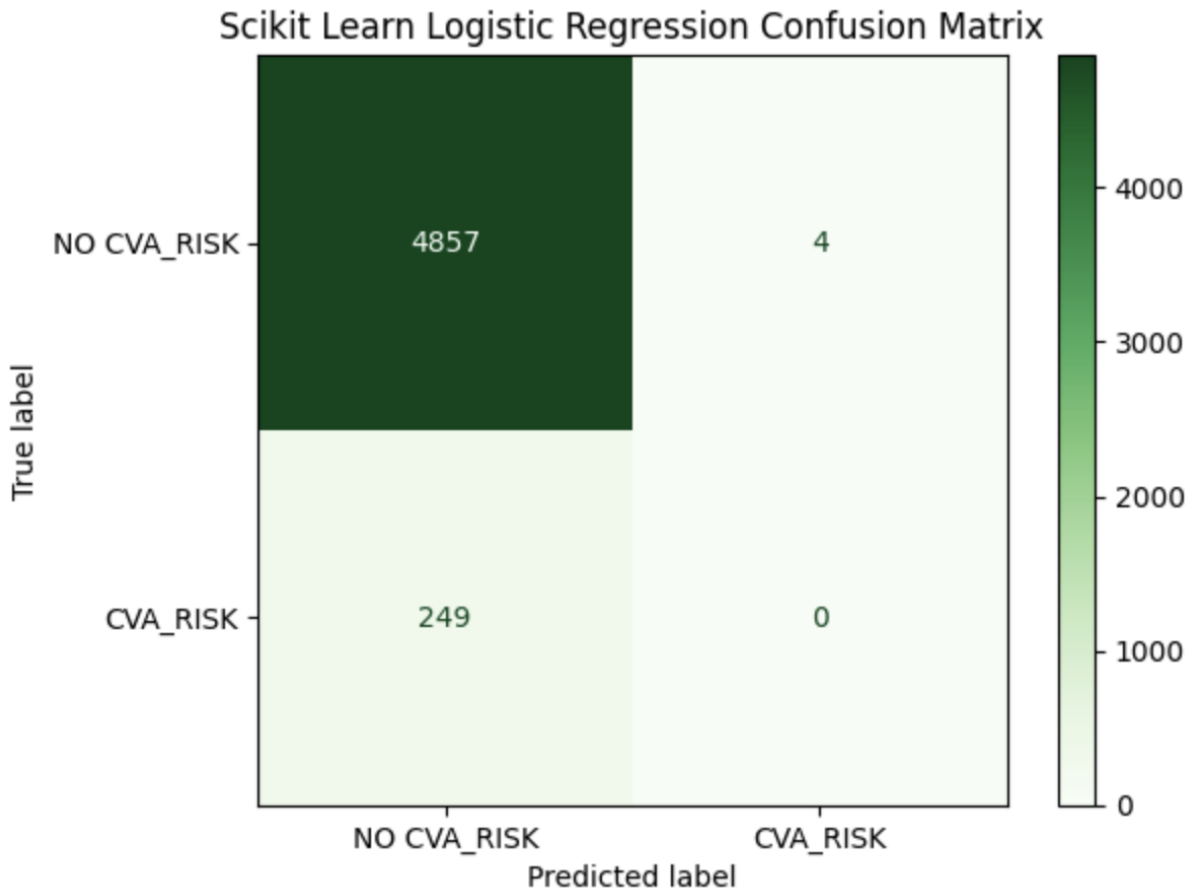

Figure 31: Manual Logistic Regression

Using scikit-learn We use scikit-learn to train and evaluate the logistic regression model.

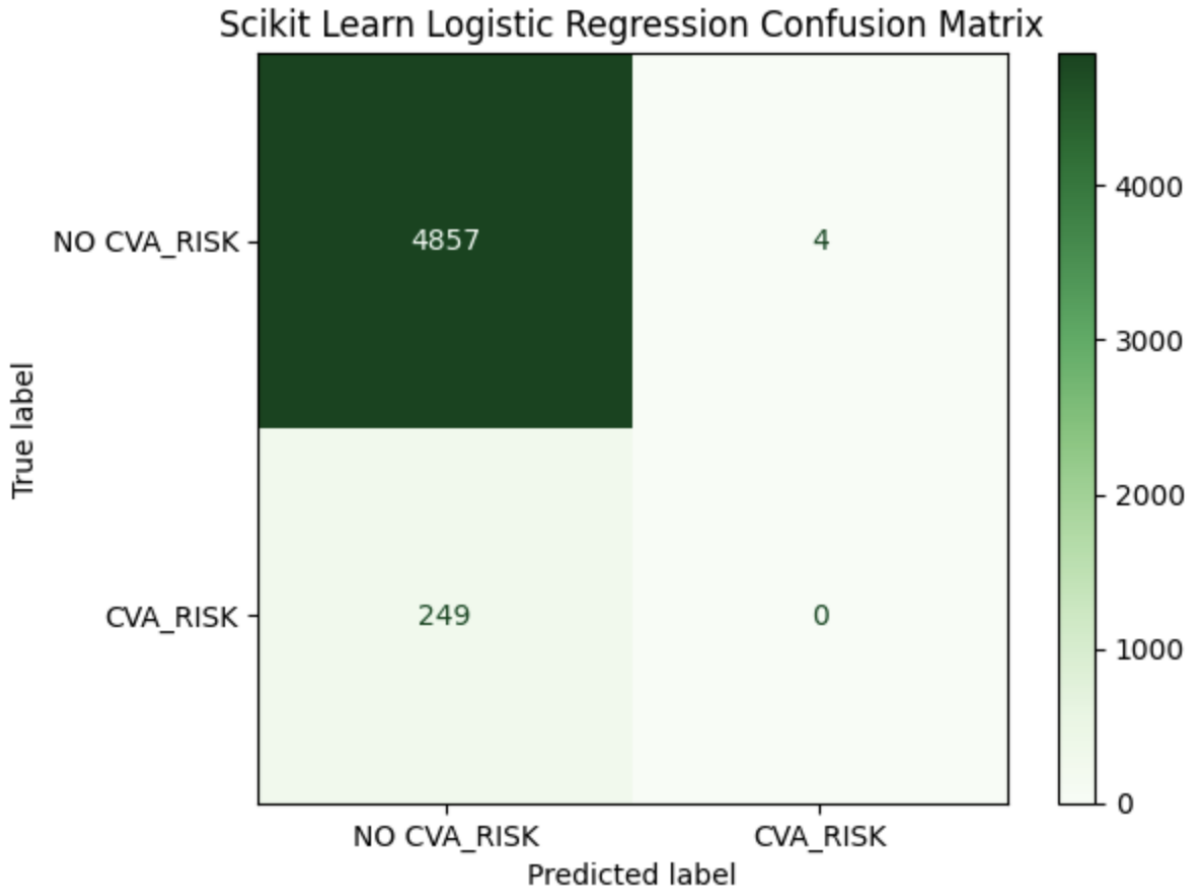

Figure 32: SK Logistic Regression Confusion Matrix

2.5.2 Multi-Layer Perceptron (MLP)

Manual Implementation We manually implement an MLP model using standard backpropagation.

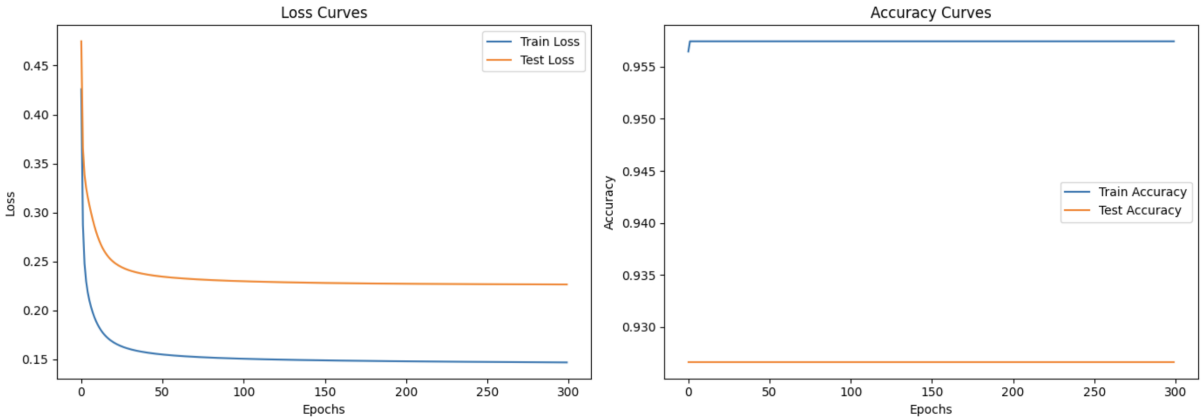

Figure 33: Error and Accuracy Curves Manual MLP

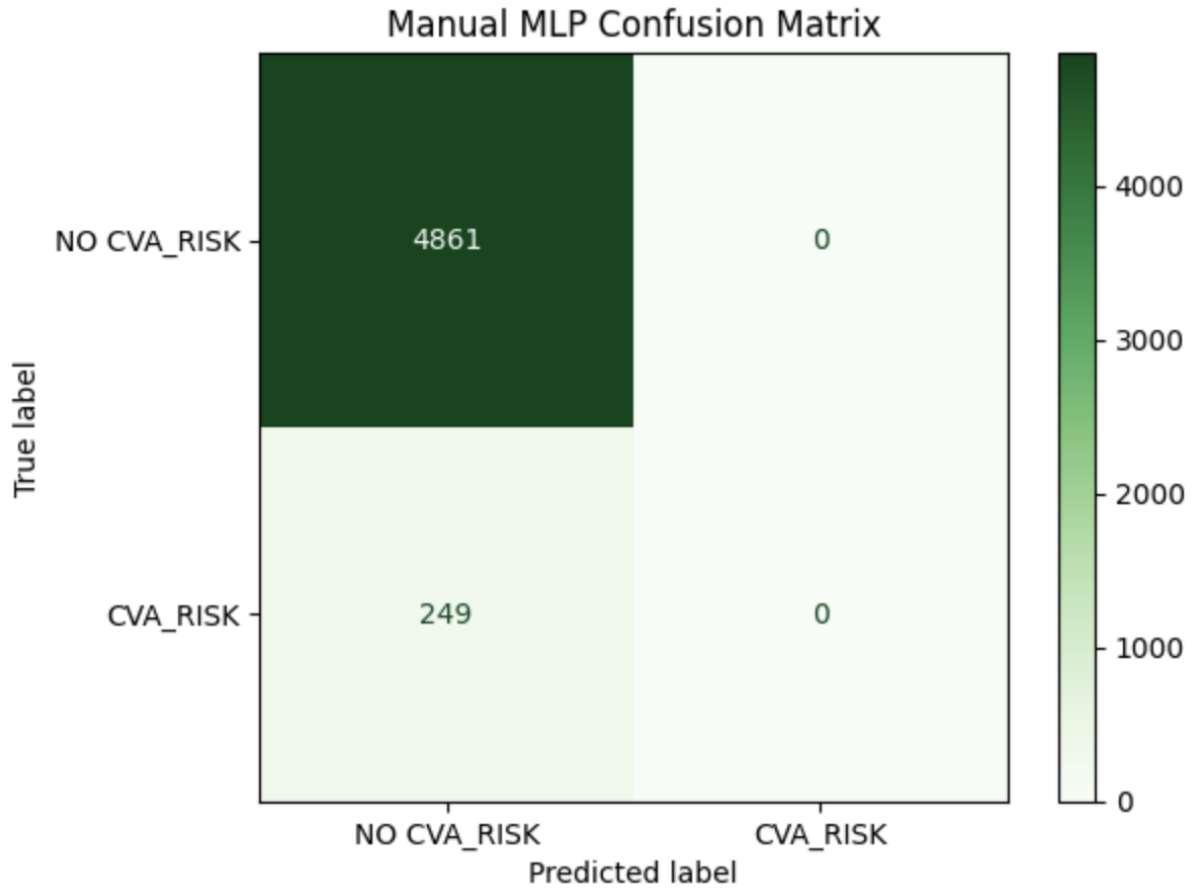

Figure 34: Manual MLP Confusion Matrix

Using scikit-learn We use scikit-learn to train and evaluate the MLP classifier.

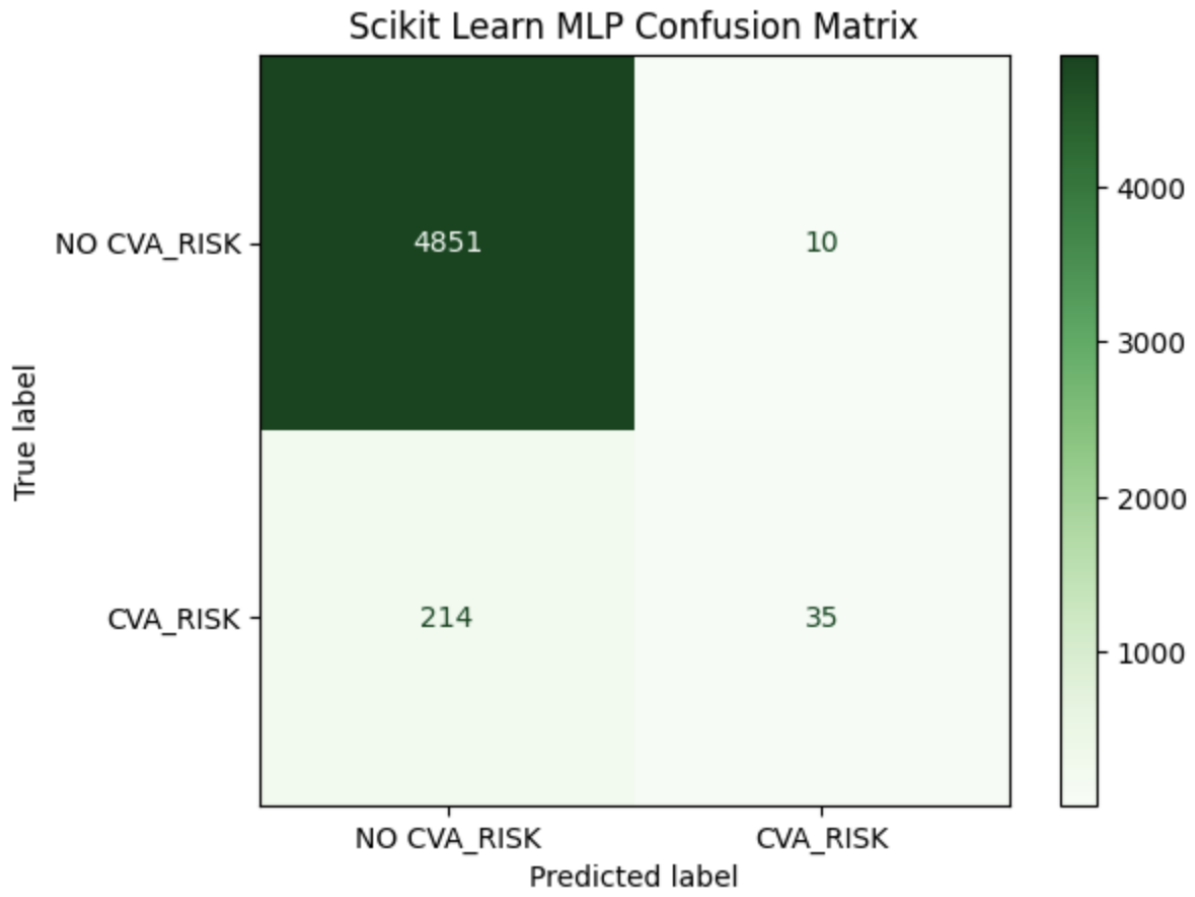

Figure 35: SK MLP Confusion Matrix

2.6 Evaluation

2.6.1 Hyperparameter Tuning

We document the final set of hyperparameters used for each model.

Manual Logistic Regression

- **Learning Rate (lr)**: 0.15
- **Number of Epochs (epochs_no)**: 1500

Scikit-Learn Logistic Regression

- **Max Iterations (max_iter)**: 1000
- **Solver**: 'lbfgs'

Manual MLP

- **Number of Epochs**: 1000
- **Learning Rate**: 0.01

Scikit-Learn MLPClassifier

- **Hidden Layer Sizes**: (300, 10)
- **Max Iterations (max_iter)**: 1200
- **Random State**: 3
- **Learning Rate (learning_rate)**: 'adaptive'
- **Learning Rate Initialization (learning_rate_init)**: 0.02
- **Solver**: 'sgd'

2.6.2 Analysis

Train Accuracy

- The Scikit-Learn MLP model achieves the highest train accuracy (0.964286), indicating that it fits the training data very well.
- The Manual Logistic Regression model has the lowest train accuracy (0.955235), but the difference is quite small.

Test Accuracy

- The Scikit-Learn MLP model shows the highest test accuracy (0.932679), suggesting it generalizes well to unseen data.
- The Manual Logistic Regression and Manual MLP models both have a test accuracy of 0.926614, which is slightly lower but still comparable.

Precision

- Precision is highest for the Scikit-Learn MLP model (0.777778), indicating it has the least false positives compared to the other models.
- The Scikit-Learn Logistic Regression model has the lowest precision (0.000000), indicating it fails to correctly identify positive instances.

Recall

- Recall is also highest for the Scikit-Learn MLP model (0.140562), suggesting it successfully identifies a higher number of true positives.
- Both the Manual Logistic Regression and Manual MLP models have very low recall, indicating they miss most actual positives.

F1 Score

- The Scikit-Learn MLP model achieves the highest F1 Score (0.238095), which is a balance between precision and recall.
- The Scikit-Learn Logistic Regression model has the lowest F1 Score (0.000000), reflecting its poor performance in both precision and recall.

Conclusion

Overall, the Scikit-Learn MLP model demonstrates the best performance across most metrics, including train accuracy, test accuracy, precision, recall, and F1 score, indicating it is the most robust model for this dataset. The Manual MLP also performs well in terms of test accuracy but has issues with precision and recall, possibly due to improper handling of positive instances. The logistic regression models (both manual and Scikit-Learn) show reasonable accuracy but suffer from low precision and recall.

These results highlight the effectiveness of more complex models like MLPs in capturing intricate patterns in the data, provided they are properly tuned and regularized. The Scikit-Learn MLP, in particular, stands out as the most reliable model for this classification task.

IMPORTANT: The model overfitting to 'NO AVC RISK' is due to the low number of 'AVC RISK' (1 value) in the dataset. The model fits to value 0. Thus, 0 and nan values for precision, recall, and F1 are due to overfitting of the model.

	Manual Logistic Regression	Scikit Learn Logistic Regression	Manual MLP	Scikit Learn MLP
Train accuracy	0.947162	0.956703	0.957436	0.964286
Test accuracy	0.918787	0.925636	0.926614	0.923679
Precision	0.068966	0.000000	nan	0.777778
Recall	0.016064	0.000000	0.000000	0.140562
F1 score	0.026059	nan	nan	0.238095

Figure 36: Comparison for AVC Dataset