Cardiovascular Disease Prediction using different Machine Learning Models

Group - Algo Thinkers

Team Member	**Models**
Saira Kiran	Decision Tree
Hetavi Gheewala	K-Nearest neighbors
Gia Calip	Neural Network
Cassandra Tolton	Logistic Regression
Jathelin Gambiza	

Table of Contents

List of Tables

List of Figures

1. Introduction

Heart disease, also called cardiovascular disease (CVD), is currently the most significant concern in the healthcare field. It is a leading cause of death globally, being both highly lethal and chronic. According to the World Health Organization (WHO), approximately 20.5 million people die each year from cardiovascular disease, accounting for roughly 31.5% of all global deaths. This number is expected to increase to 24.2 million by 2030 [1]. Early prediction of cardiovascular disease (CVD) plays an important role in preventing severe health outcomes. By identifying individuals at risk, timely interventions can be implemented, allowing people to adopt healthier lifestyles and seek proper medical treatment before the disease progresses. Early detection can involve monitoring risk factors such as blood pressure, cholesterol levels, and lifestyle habits. However, early detection or prediction of heart disease is a challenging task in the medical field, as it requires accurate identification of potential risks from a wide range of complex factors. Machine learning (ML) has shown promise in assisting with disease prediction by analyzing large datasets and recognizing patterns.

In our project, we applied various machine learning models to predict heart disease risk based on specific criteria (clinical measurements). Our main focus was to train different models, evaluate their performance, and identify which one provided the most accurate and reliable predictions. By comparing the results of these models, we aimed to determine the best approach for predicting heart disease risk and offer insights into improving early detection methods.

2. Dataset Selection

We selected Cardiovascular dataset, available online at the Kaggle [2]. The data medical information related to individuals' cardiovascular health. The dataset contains several attributes, including both objective and subjective features, and a binary target variable indicating whether an individual has cardiovascular disease. The dataset contains 13 attributes, including the target variable, with a total of 7000 instances. The attributes and first few entries of dataset are detailed in the Table1 and Figure 1.

Table 1: Dataset Attributes

No.	Attributes
1	ID
2	Age
3	Gender
4	Height
5	Weight
6	Systolic Blood Pressure (ap_hi)
7	Diastolic Blood Pressure (ap_lo)
8	Cholesterol
9	Glucose
10	Smoke
11	Alcohol
12	Active
13	Cardio

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	id	age	gender	height	weight	ap_hi	ap_lo	cholesterol	gluc	smoke	alco	active	cardio
2	0	18393	2	168	62	110	80	1	1	0	0	1	0
3	1	20228	1	156	85	140	90	3	1	0	0	1	1
4	2	18857	1	165	64	130	70	3	1	0	0	0	1
5	3	17623	2	169	82	150	100	1	1	0	0	1	1
6	4	17474	1	156	56	100	60	1	1	0	0	0	0
7	8	21914	1	151	67	120	80	2	2	0	0	0	0
8	9	22113	1	157	93	130	80	3	1	0	0	1	0
9	12	22584	2	178	95	130	90	3	3	0	0	1	1
10	13	17668	1	158	71	110	70	1	1	0	0	1	0
11	14	19834	1	164	68	110	60	1	1	0	0	0	0
12	15	22530	1	169	80	120	80	1	1	0	0	1	0
13	16	18815	2	173	60	120	80	1	1	0	0	1	0
14	18	14791	2	165	60	120	80	1	1	0	0	0	0
15	21	19809	1	158	78	110	70	1	1	0	0	1	0
16	23	14532	2	181	95	130	90	1	1	1	1	1	0
17	24	16782	2	172	112	120	80	1	1	0	0	0	1
18	25	21296	1	170	75	130	70	1	1	0	0	0	0
19	27	16747	1	158	52	110	70	1	3	0	0	1	0
20	28	17482	1	154	68	100	70	1	1	0	0	0	0
21	29	21755	2	162	56	120	70	1	1	1	0	1	0

Figure 1: Sample records from the Cardiovascular Disease Dataset

2.1 From Binary to Multi-Class Classification

The dataset was originally designed for binary classification, where the target variable indicated the presence or absence of cardiovascular disease (CVD). However, we redesigned the target variable by classifying people according to their cardiovascular risk categories in order to increase the prediction accuracy. We applied risk criteria to classify individuals into four categories**: low risk, moderate risk, high risk, and no risk.** Table 2 provides details of risk criteria. Using this new classification system, we performed a multi-class classification task, where the goal was to predict the risk category for each individual. This allowed for a more detailed risk assessment and improved the prediction model's ability to categorize individuals based on varying levels of cardiovascular risk.

Table 2: Risk Map

Category	Criteria
No Risk	• cardio = 0 • Cholesterol = 1 (normal) • Glucose = 1 (normal) • Systolic Blood Pressure (ap_hi) < 120 • Diastolic Blood Pressure (ap_lo) <= 80 • Alcohol = 0 • Smoke = 0
Low Risk	• cardio = 0 • Cholesterol <= 2 (slightly elevated or normal) • Glucose <= 2 (slightly elevated or normal) • Systolic Blood Pressure (ap_hi) < 130 • Diastolic Blood Pressure (ap_lo) <= 85 • Alcohol = 1 • Smoke = 1
Moderate Risk	• cardio = 1 • Cholesterol <= 3 (elevated) • Glucose <= 3 (elevated) • Systolic Blood Pressure (ap_hi) between 130 and 139 • Diastolic Blood Pressure (ap_lo) between 85 and 89 • Alcohol = 1 • Smoke = 1

High Risk	• cardio = 1 • Cholesterol = 3 (high) • Glucose = 3 (high) • Systolic Blood Pressure (ap_hi) >= 140 • Diastolic Blood Pressure (ap_lo) >= 90 • Alcohol = 1 • Smoke = 1

3. Models Selection

We applied five different machine learning models on the dataset to predict the cardiovascular risk. The models used in this project are:

1. K-Nearest Neighbors
2. Neural Network
3. Logistic Regression
4. Decision Tree

Each model was trained on the dataset using the risk categories: low risk, moderate risk, high risk, and no risk as the target variable. We then evaluated the performance of these models using evaluation metrics such as accuracy, precision, recall, and F1-score to determine which model provided the best results for predicting cardiovascular risk.

4. K-Nearest Neighbors

K-Nearest Neighbors was applied to the dataset for cardiovascular risk prediction assist effective for multiclassification and can handle complex relationship between features.

4.1 Data Preprocessing - KNN

Following steps were performed to make the dataset ready for model training.

- **Data Filtering**: First, the data was filtered to remove the outliers based on the blood pressure (ap_lo and ap_hi). Specifically, rows where ap_hi was below 50 or above 200, and ap_lo exceeded 120, were excluded to ensure valid and consistent data.

- **Risk Level Assignment**: A custom risk function was applied to categorize individuals into four risk levels (No Risk, Low Risk, Moderate Risk, High Risk) based on various health indicators. This function assigns a risk score to each individual based on these defined parameters. Invalid or inconsistent cases were removed by filtering out rows where the risk level was set to -1.
- **Feature Selection**: The dataset was then reduced to relevant input features for the model training: age, gender, height, weight, blood pressure, cholesterol, glucose levels, smoking, alcohol consumption, and physical activity. The target variable (risk_level) was also defined as the output feature.
- **Data Split**: The dataset was split into training and testing sets, with 80% for training and 20% for testing.
- **Normalization**: To enhance model performance, input features were normalized using StandardScaler, ensuring that all features had a mean of 0 and standard deviation of 1, making them easier for the model to handle.
- **Dimensionality Reduction**: Principal Component Analysis (PCA) was applied to reduce the feature space while retaining 95% of the variance in the dataset. This helped improve computational efficiency and model performance by eliminating redundant features.

4.2 Model Training - KNN

Model was trained using KNN classifier.

- **Distance Metrics and K Selection**: Model was experimented with three different distance metrics (manhattan, euclidean, and minkowski) to evaluate which one would work best for the classification of heart risk. For each metric, the model's performance was tested with different values of K ranging from 1 to 20 and then the K value that achieved the highest accuracy was selected for each metric.
- **Best Model Configuration**: After evaluating multiple combinations of distance metrics and K values, the optimal configuration was selected. The KNN classifier was then trained on the training data using the best K value and distance metric, which achieved the highest accuracy.

- **Model Evaluation**: The trained KNN model was evaluated on the test dataset. The accuracy, precision, recall, F1 score, confusion matrix, and classification report of each class were used to assess the model's performance.

4.3 Results -KNN

The overall performance of the K-Nearest Neighbors (KNN) classifier for heart risk prediction are shown in Table 3. The final model achieved an accuracy of approximately 80% using the best K value (K=16) and the Manhattan distance metric.

Table 4 represents the classification report. The K-Nearest Neighbors (KNN) model shows mixed performance across different risk categories. For High-Risk class, the model performs good with a precision of 0.89, F1-score of 0.83 and a recall of 0.77. For Low Risk, the precision is 0.79, but recall drops to 0.66, indicating that the model misses some low-risk cases. The model fails to classify Moderate Risk individuals, as it achieves a precision, recall, and F1-score of 0.00, meaning it does not identify moderate-risk cases correctly. However, the model performs reasonably well for No Risk class, with a high recall of 0.92. Overall, the KNN model struggles with detecting moderate risk but is effective in identifying high-risk and no-risk individuals, though it could benefit from further optimization to improve its performance across all categories. Figure 2 shows the implementation results of the model.

Table 3: Model Performance Metrices - KNN

KNN			
Accuracy	**Precision**	**Recall**	**F1- Score**
0.80	0.81	0.80	0.80

Table 4: Classification Report - KNN

KNN			
Category	**Precision**	**Recall**	**F1-Score**
High Risk	0.89	0.77	0.83
Low Risk	0.79	0.66	0.68
Moderate Risk	0.00	0.00	0.00
No Risk	0.70	0.92	0.79

```
Results by Distance Metric:
Metric: manhattan, Optimal K: 20, Accuracy: 79.84%
Metric: euclidean, Optimal K: 20, Accuracy: 79.84%
Metric: minkowski, Optimal K: 20, Accuracy: 79.84%

Optimal Metric: manhattan, Optimal K: 20, Accuracy: 79.84%
Model Accuracy with Best K (20): 79.84%

Confusion Matrix:
                  P-No Risk  P-Low Risk  P-Moderate Risk  P-High Risk
A-No Risk              7306           0                0          613
A-Low Risk              279        2080                0          785
A-Moderate Risk           0           0                0           91
A-High Risk            2886         898                0        12608

Classification Report:
                precision    recall  f1-score   support

      No Risk       0.70      0.92      0.79      7919
     Low Risk       0.70      0.66      0.68      3144
Moderate Risk       0.00      0.00      0.00        91
    High Risk       0.89      0.77      0.83     16392

     accuracy                           0.80     27546
    macro avg       0.57      0.59      0.58     27546
 weighted avg       0.81      0.80      0.80     27546
```

Figure 2: Model Implementation Results - KNN

5. Neural Network

Neural network was also applied on the dataset for risk classification as its ability to capture complex relationship between features. Model implementation process is outlined below.

4.1 Data Preprocessing – Neural network

Following steps were performed to make the dataset ready for neural network model training:

- **Data Cleaning and Column Renaming**: The dataset was first loaded and cleaned to ensure consistent column names. This helped avoid any issues when accessing columns for processing.
- **Risk Level Assignment**: A custom function was applied to categorize individuals into four risk levels: **No Risk**, **Low Risk**, **Moderate Risk**, and **High Risk**. This function

evaluated factors such as cholesterol levels, blood pressure, smoking status, and alcohol consumption.

- **Feature Selection**: To improve model performance, only relevant features were selected for training the model. This included age, gender, height, weight, cholesterol levels, blood pressure, glucose levels, smoking, and alcohol consumption. The target variable, risk_category, was defined as the output feature, which represents the risk level of the individual.
- **Correlation Analysis**: A correlation matrix was computed to evaluate the relationships between features and the target variable. Features with a correlation greater than 0.2 to the target were selected for inclusion in the final dataset. This step helped reduce the dimensionality of the data by keeping only the most relevant variables for the model.
- **Data Encoding**: The target variable risk_category was encoded into numerical values using Label Encoding.
- **Data Normalization**: The features were normalized using **StandardScaler**, which ensured all features had a mean of 0 and standard deviation of 1. This was done to prevent features with larger ranges from dominating the training process and to speed up model convergence.
- **Data Split**: The dataset was split into **training** and **testing** sets, with 80% of the data used for training the model and 20% for testing.

4.2 Model Training – Neural Network

The model was trained using a Neural Network classifier built with tensorflow. Training steps are mentioned below.

- **Model Architecture**: The neural network was designed with an input layer containing 64 neurons and a **ReLU activation function**. A **Dropout layer** was included with a rate of 0.3 to reduce overfitting. The hidden layer consisted of 32 neurons, also using **ReLU activation**. The output layer had a number of neurons equal to the number of classes (four risk categories), with a **softmax activation function** to produce a probability distribution over the classes.

- **K-Fold Cross-Validation**: To ensure the model's robustness, **K-Fold Cross-Validation** (with 5 folds) was applied. The model was trained on 4 folds and tested on the remaining fold in each iteration. This process was repeated for each fold to ensure that all data points were used for both training and evaluation. The model's accuracy was averaged across all folds to provide a more reliable performance estimate.
- **Model Compilation**: The model was compiled using the **Adam optimizer**, which is well-suited for training neural networks. The **categorical cross-entropy** loss function was used, as it is appropriate for multi-class classification tasks. Accuracy was used as the evaluation metric to monitor the model's performance.
- **Model Training**: The neural network was trained for **20 epochs**, with a **batch size of 32**.
- **Model Evaluation**: After training, the model was evaluated on the test dataset. Various performance metrics, accuracy, precision, recall, F1- score, confusion Metrix, and classification report were used to assess the performance of the model.

4.3 Results – Neural Network

Table 5 shows the performance of the neural network classifier. The classification report of each class is shown in Table 6. The classification report shows varied performance across different risk categories. For High Risk, the model demonstrates good precision (0.85), meaning it correctly identifies high-risk individuals 85% of the time, but its recall (0.71) is slightly lower, indicating it misses some high-risk individuals. The F1-Score for high-risk is 0.77, balancing precision and recall. In the Low-Risk category, the model has a precision of 0.70 and a recall of 0.67, leading to an F1-Score of 0.68, indicating moderate performance with room for improvement. For Moderate Risk, the model achieves perfect performance, with both precision and recall at 1.00 and an F1-Score of 1.00, meaning it identifies all moderate-risk individuals correctly without error. Lastly, for No Risk, the model performs well with a precision of 0.74 and a high recall of 0.93, resulting in a strong F1-Score of 0.83, reflecting its ability to accurately predict individuals with no risk.

Based on the confusion matrix values, the high performing prediction was "no risk." "No risk" predicted 3718 within the 3959 predicted value. "Moderate risk" follows after, with minimal misclassification. Lastly, "high risk" needs notable improvement as it misclassified 611 as "low risk" and 1287 as being "no risk." The overall performance of the program harbored an accuracy

of 82% and an F1 score of 81%, suggesting that further improvements need to focus on reducing the misclassification for “high risk. The model implementation results are shown in Figure 3.

Table 5: Model Performance Metrices - Neural Network

Neural Network			
Accuracy	Precision	Recall	F1- Score
0.82	0.84	0.82	0.82

Table 6: Classification Report - Neural Network

Neural Network			
Category	Precision	Recall	F1-Score
High Risk	0.89	0.67	0.77
Low Risk	0.68	0.83	0.75
Moderate Risk	0.99	1.00	1.00
No Risk	0.74	0.94	0.83

```
Detailed Results for all k:
Confusion Matrix (Text Output):
[[3986  611   10 1287]
 [ 255 1311    6    1]
 [   1    0 2346    0]
 [ 241    0    0 3718]]

Formatted Confusion Matrix (with labels):
              High Risk  Low Risk  Moderate Risk  No Risk
High Risk          3986       611             10     1287
Low Risk            255      1311              6        1
Moderate Risk         1         0           2346        0
No Risk             241         0              0     3718
Classification Report:
               precision    recall  f1-score   support

    High Risk       0.89      0.68      0.77      5894
     Low Risk       0.68      0.83      0.75      1573
Moderate Risk       0.99      1.00      1.00      2347
      No Risk       0.74      0.94      0.83      3959

     accuracy                           0.82     13773
    macro avg       0.83      0.86      0.84     13773
 weighted avg       0.84      0.82      0.82     13773
```

Figure 3: Model Implementation Results - Neural Network

5. Logistic Regression

Logistic regression was another model applied to the dataset for risk classification due to its effectiveness in predicting multiclass and its ability to model the relationship between input features and the probability classes. The model implementation is described below.

6.1 Data Preprocessing – Logistic Regression

The following preprocessing steps were performed to prepare the data for training the Logistic Regression model:

- **Data Loading and Risk Level Assignment**: The dataset was loaded and a custom function (rsk_lvl) was applied to categorize individuals into four risk levels: No Risk (0), Low Risk (1), Moderate Risk (2), and High Risk (3).
- **Feature Selection**: The dataset was split into input features (X) and output labels (Y). The input features include health parameters, while the output labels represent the risk levels assigned in the previous step.
- **Data Split**: The dataset was divided into training and testing sets using an 80/20 split (80% for training, 20% for testing).
- **Normalization**: To ensure that all features had a comparable scale and improve the model's performance, the data was normalized using StandardScaler. This process standardizes the data by transforming it to have a mean of 0 and a standard deviation of 1, which is especially important for distance-based models like Logistic Regression.
- **Dimensionality Reduction (PCA)**: Principal Component Analysis (PCA) was applied to reduce the feature space. Three principal components were selected to retain as much variance as possible, reducing the dimensionality while maintaining important information.

6.2 Model Training – Logistic Regression

The Logistic Regression model was trained using the preprocessed data:

- **Model Initialization and Training**: A Logistic Regression classifier was used for training. The model was trained on the normalized training data, with the corresponding risk level labels.
- **Prediction**: After training, the model was used to make predictions on the test data. The predict() method was applied to generate the predicted risk levels for the test set.

- **Model Evaluation**: Several metrics were used to evaluate the performance of the model including accuracy, precision, recall, F1 score, confusion metric, classification report of each category and ROC Curve.

6.3 Results – Logistic Regression

The logistic regression model achieved an accuracy of 96.7%. Table 7 shows the overall performance of the model. Classification report for each class is shown in Table 8. The classification report indicates that the Logistic Regression model performs well for High Risk and No Risk categories, with high precision, recall, and F1-scores of 0.98 and 0.97, respectively. This means the model is accurately identifying most instances of these classes. However, the model fails to identify any instances of the Moderate risk, resulting in zero precision, recall, and F1-score. The model implementation results are shown in Figure 4.

Table 7: Model Performance metrices - Logistic Regression

Logistic Regression			
Accuracy	**Precision**	**Recall**	**F1- Score**
0.96	0.96	0.97	0.96

Table 8: Classification Report - Logistic Regression

Logistic Regression			
Category	Precision	Recall	F1-Score
High Risk	0.98	0.98	0.98
Low Risk	0.89	0.83	0.86
Moderate Risk	0.00	0.00	0.00
No Risk	0.97	1.00	0.98

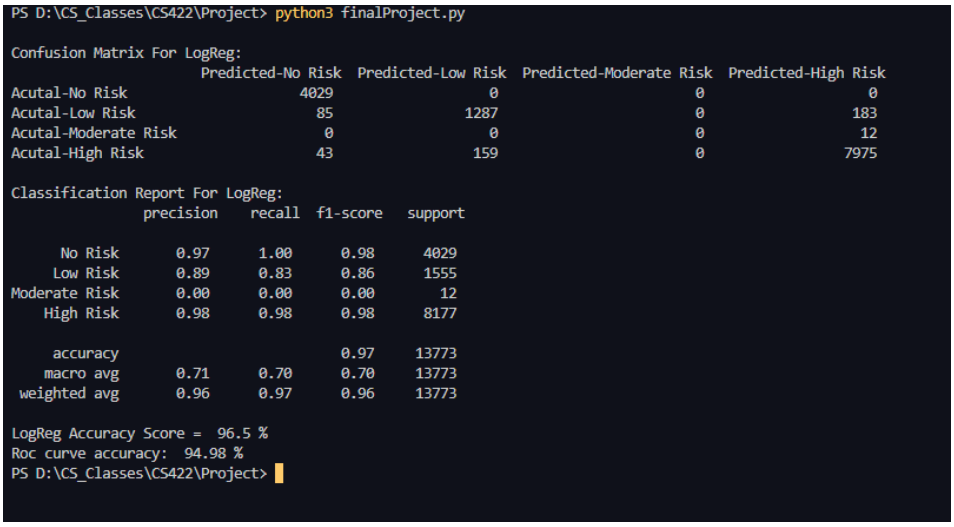


Figure 4: Model Implementation Results - Logistic Regression

6. Decision Tree

Decision tree classifier was applied to the classification of heart risk. It's a powerful model, particular useful for large dataset. It can capture complex pattern without requiring extensive feature engineering.it provides clear decision boundaries. The detail information of model implementation is described below.

7.1 Data Preprocessing - Decision Tree

The following preprocessing steps were performed to prepare the data for training the Decision Tree model:

- **Risk Level and Missing Values:** A customize risk function was applied to assign the risk level (low, high, moderate and no risk) to each instance in the dataset based on their parameters after medical assessment. After that total instance count for each risk category was checked along with missing values. There were no missing values in the dataset. But data was highly imbalance after assigning the risk category to each instances.
- **Feature and Label Separation**: The dataset was separated into input features and the target label. The input features included various health parameters, while the target labels corresponded to the risk levels assigned in the previous step.
- **Data Split**: The dataset was split into training and testing sets, with 80% of the data used for training and 20% for testing.

- **Handling Class Imbalance with SMOTE**: After assigning the risk to each instance, there was class imbalance between the 4 categories. To address class imbalance, the Synthetic Minority Over-sampling Technique (SMOTE) was applied. This technique generated synthetic samples for the minority classes, ensuring that the training data had a balanced distribution of the different risk levels.
- **Normalization**: The input features were normalized using StandardScaler to standardize the data. This step helped improve the performance of the machine learning model.
- **Feature Selection using RFE**: Recursive Feature Elimination (RFE) was used to select the important features for model training. This feature selection helped reduce dimensionality and focus on the most relevant features.

7.2 Model Training - Decision Tree

The Decision Tree model was trained and evaluated using the following steps:

- **Model Training with Cross-Validation**: To evaluate the model's generalization ability, 10-fold cross-validation was performed using the training data. Cross-validation helps assess how the model performs on unseen data by splitting the training set into 10 subsets and evaluating the model's performance on each fold.
- **Model Training**: After evaluating the model's performance using cross-validation, the Decision Tree classifier was trained on the entire resampled training data, which was processed using SMOTE and RFE. The decision tree learned to classify the individuals into one of the four risk levels based on the selected features.
- **Prediction**: After training, the model was used to predict the risk levels for the test data, which had also been processed similarly to the training data.
- **Model Evaluation**: Evaluation metrics; accuracy, precision, recall, F1-score, classification report, and confusion matrix curve were used to assess the performance of the trained Decision Tree model.

7.3 Results – Decision Tree

Decision tree achieved a good accuracy of 97.86%. With 95% of precision recall and f1 score respectively. The mean cross validation accuracy of the model was 91.76%. Table 9 shows the performance of the model. The classification report for the Decision Tree is shown in Table 10.

Decision tree performed excellent across all categories, particularly High Risk, No Risk, and Moderate Risk, where it achieves perfect precision, recall, and F1-scores of 1.00.it means that the model can accurately identify instances of these categories with no false positives or false negatives. Figures 5 and 6 show the confusion matrix and model implementation results.

Table 9: Model Performance Metrices - Decision Tree

Decision Tree			
Accuracy	**Precision**	**Recall**	**F1- Score**
0.97	0.97	0.96	0.97

Table 10: Classification Report -Decision Tree

Decision Tree			
Category	**Precision**	**Recall**	**F1-Score**
High Risk	1.00	1.00	1.00
Low Risk	0.81	0.95	0.87
Moderate Risk	1.00	0.94	0.97
No Risk	1.00	1.00	1.00

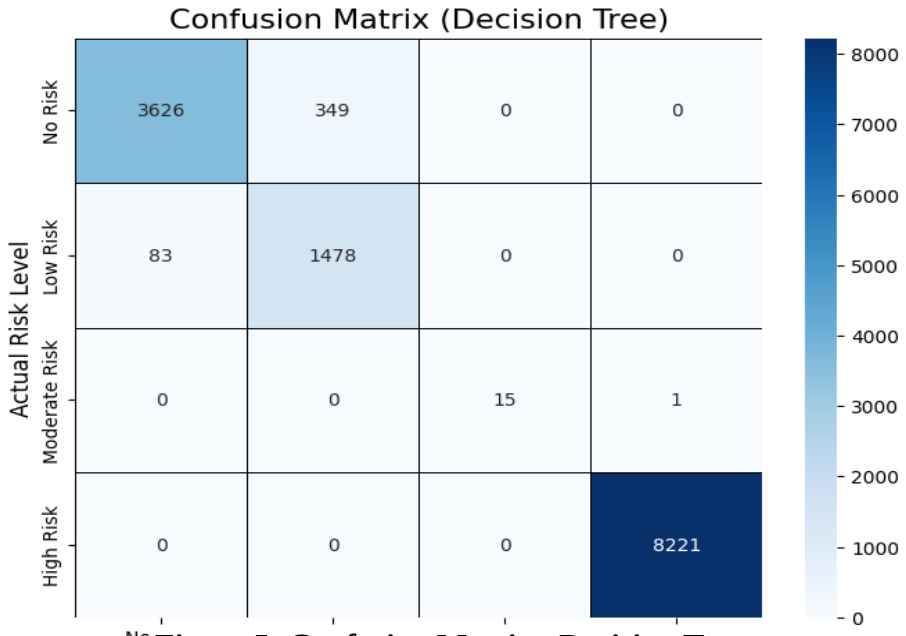
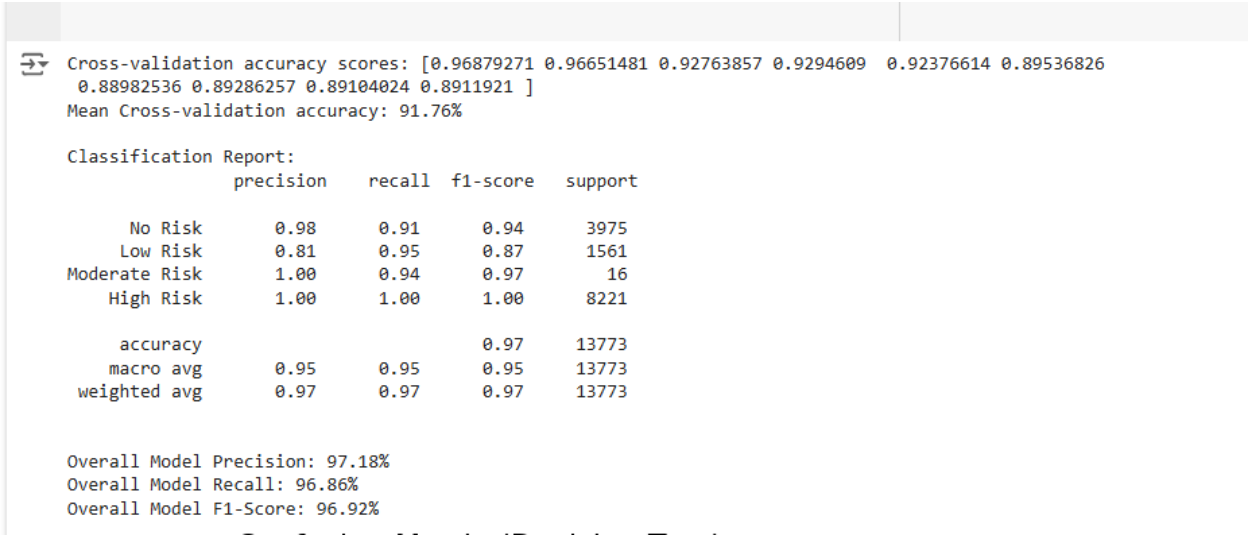


Figure 5: Confusion Matrix - Decision Tree



```
Cross-validation accuracy scores: [0.96879271 0.96651481 0.92763857 0.9294609  0.92376614 0.89536826
 0.88982536 0.89286257 0.89104024 0.8911921 ]
Mean Cross-validation accuracy: 91.76%

Classification Report:
               precision    recall  f1-score   support

      No Risk       0.98      0.91      0.94      3975
     Low Risk       0.81      0.95      0.87      1561
Moderate Risk       1.00      0.94      0.97        16
    High Risk       1.00      1.00      1.00      8221

     accuracy                           0.97     13773
    macro avg       0.95      0.95      0.95     13773
 weighted avg       0.97      0.97      0.97     13773


Overall Model Precision: 97.18%
Overall Model Recall: 96.86%
Overall Model F1-Score: 96.92%
```

Figure 6: Model Implementation Results - Decision Tree

9. Performance Comparison of Models and Discussion

Table shows the accuracy, precision, recall and f1 score comparison of all the models. Out of all the model's decision tree achieved the highest accuracy of 97%. Logistic regression classifier achieved the second highest accuracy of 96%. Table shows classification report comparison of all the models. Decision tree performed well for all the categories with good precision recall and f1 score.

Table 11: Performance Comparison of all Models

Model	Accuracy	Precision	Recall	F1-Score
Decision Tree	97%	97%	96%	97%
Logistic Regression	96%	96%	97%	96%
Neural network	82%	84%	82%	82%
K-Nearest Neighbor	80%	81%	80%	80%

Table 12: Classification Report Comparison of all Models

Models	Precision	Recall	F1-Score
Decision Tree			
High Risk	1.00	1.00	1.00
Low Risk	0.81	0.95	0.87
Moderate Risk	1.00	0.94	0.97
No Risk	1.00	1.00	1.00
Logistic Regression			
High Risk	0.98	0.98	0.98
Low Risk	0.89	0.83	0.86
Moderate Risk	0.00	0.00	0.00
No Risk	0.97	1.00	0.98
Neural Network			
High Risk	0.89	0.67	0.77
Low Rik	0.68	0.83	0.75
Moderate Risk	0.99	1.00	1.00
No Risk	0.74	0.94	0.83
K-Nearest Neighbors			
Low Risk	0.89	0.77	0.83
High Rik	0.79	0.66	0.68
Moderate Risk	0.00	0.00	0.00
No Risk	0.70	0.92	0.79

Conclusion

In this project, we applied different machine learning models for the classification of cardiovascular disease. Across all models, we achieved commendable accuracy, with the highest accuracy reaching 97%. These results demonstrate that machine learning techniques can be highly effective in predicting cardiovascular risk levels. Given the promising accuracy achieved, these models have the potential to be beneficial in clinical settings for early detection and risk stratification of cardiovascular diseases, ultimately contributing to improved patient outcomes and preventive healthcare measures.

Challenges and Future Direction

One of the biggest challenges faced during this project was handling the large dataset size and the class imbalance present in the target classes. However, various techniques, such as resampling and feature selection, were utilized to handle this issue and improve model performance.

In the future, incorporating a larger and more balanced dataset, particularly for underrepresented classes, could lead to better model accuracy.

Team Member Contributions

Team Member	**Contribution**
Saira Kiran	Report writing
Hetavi Gheewala, Gia Calip	Created the Presentation slides
Jathelin Gambiza, Cassandra Tolton	Presented the slides

References

[1] WHO. [Online]. Available: https://www.who.int/health-topics/cardiovascular-diseases. [Accessed 2024].

[2] Cardiovascular dataset. [Online]. Available: https://www.kaggle.com/datasets/sulianova/cardiovascular-disease-dataset.

.