CS 412 Project Report

Cardiovascular Disease Prediction

Venkata Pratheek Reddy Chavva

Uday Nair

Pranav Veldurthi

Mounika Boggada

Your personal views about the fields and its applications, a brief description of what you have learned in this course, and what you want to study more;

The journey to machine learning through CS 412 has been a deeply enriching experience that has broadened our understanding of this dynamic field and its myriad applications. Initially, machine learning appeared as a concept defined merely by unknown algorithms serving input-output processes. However, as the course progressed, we gained a profound appreciation for the transformative potential of machine learning across diverse domains. One of the most striking aspects of machine learning is its vast array of applications, spanning from predicting consumer behavior to diagnosing medical conditions. This realization sparked our curiosity to explore the multifaceted landscape of machine learning and discover its implications for society and industry alike.

Throughout the course, we delved into a comprehensive array of topics and techniques, each contributing to our understanding of machine learning. From the foundational principles of linear algebra to the concepts of deep learning architectures, each domain presented unique insights and approaches. Notably, supervised learning, with its emphasis on linear regression and logistic regression, provided a solid foundation for understanding predictive modeling. However, it was the realm of unsupervised learning that truly captured our fascination. The exploration of unsupervised learning, encompassing techniques such as clustering, principal component analysis (PCA), and Gaussian mixture models (GMM), unveiled the intricate complexities of data analysis. The prospect of extracting meaningful patterns and structures from unlabeled data presented both a technical challenge and an intellectual pursuit. This aspect of machine learning, where algorithms can autonomously uncover inherent structures of the data underscores the potential of unsupervised learning in extracting insights from vast and unstructured datasets.

Looking ahead, we are eager to delve deeper into the technical nuances of unsupervised learning, exploring advanced methodologies such as deep generative models and reinforcement learning. Additionally, we’re keen to explore the practical applications of machine learning in specialized domains such as natural language processing (NLP), computer vision, and healthcare informatics. By combining theoretical knowledge with hands-on experience, we aim to contribute to the development of innovative solutions that address complex societal challenges and drive meaningful advancements in technology. Moving forward, we’re excited about using machine learning to make real-world differences and spur innovation.

GOAL

Cardiovascular diseases are one of the major causes of death, affecting millions worldwide.However, addressing these diseases comes with its challenges. Limited access to facilities and resource constraints often hinder early detection and effective treatment.

This is where predictive analytics comes in. We can identify individuals who are at high risk of getting CVDs by combining data and advanced techniques. This project leverages predictive analytics, using advanced machine learning techniques such as Gradient Boosting and Random Forest, to develop a model that identifies individuals at high risk of CVDs.

It focuses on developing a predictive model to identify individuals at high risk of cardiovascular diseases (CVDs) using machine learning techniques. The primary objective is to improve accuracy in identifying high-risk individuals, facilitate early detection, and enable personalized interventions.

PROBLEM STATEMENT

The project addresses the need for accurate prediction tools to better manage and prevent cardiovascular diseases. The problem statement of the project revolves around the need for advanced tools in predicting cardiovascular diseases.

The challenge lies in effectively managing and utilizing this large dataset to extract meaningful insights and patterns that can predict disease risk.

The data is gathered via phone based on the health conditions and behavior. The dataset has various features that give insights into the factors contributing to cardiovascular diseases. By evaluating this data, we can uncover trends and correlations that will help us develop our predictive model, resulting in more effective interventions and better health outcomes. While the goal is to achieve high predictive accuracy, it is equally important to develop a model that is efficient in terms of computation and memory usage, making it scalable and practical for real-world applications.

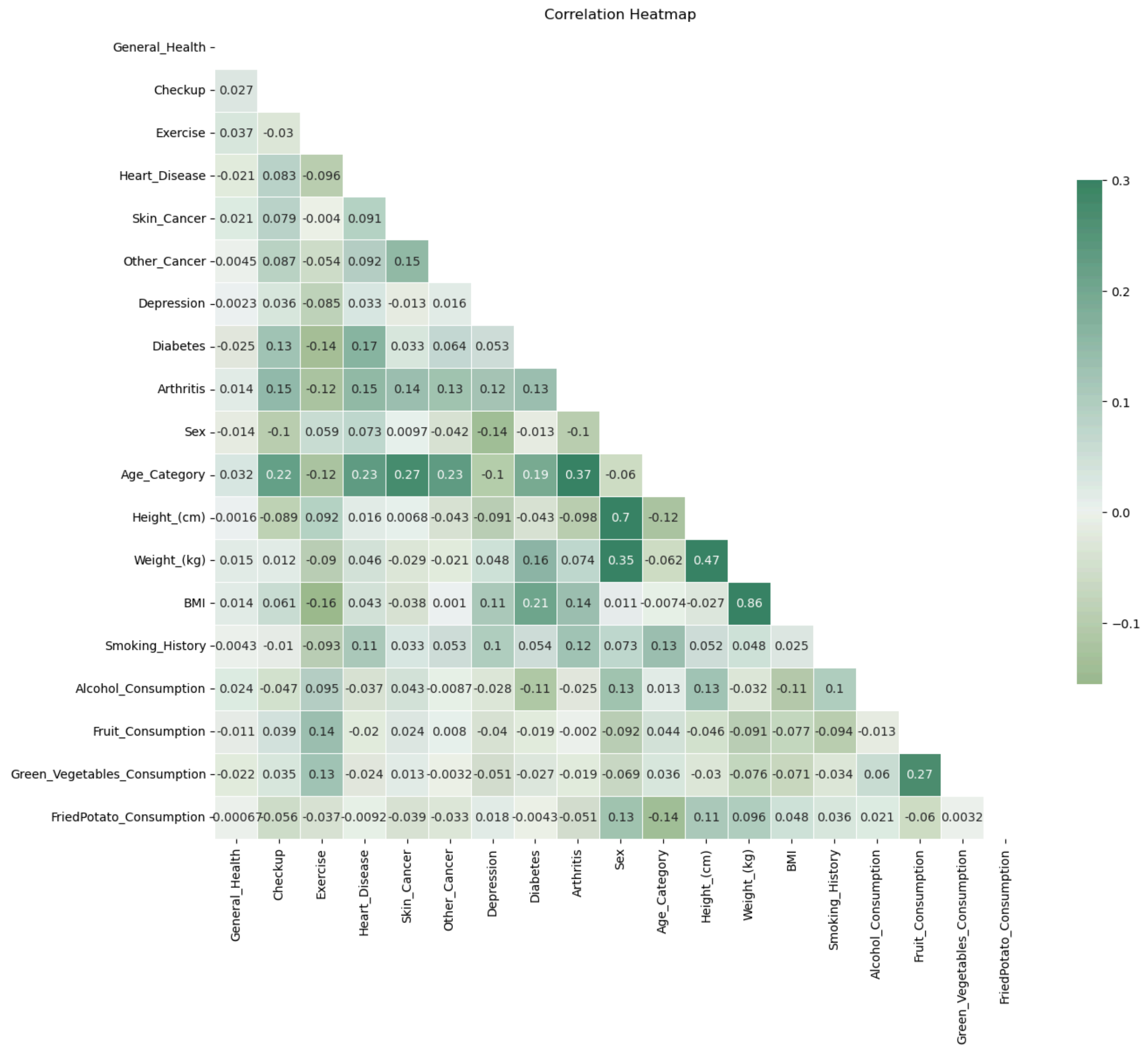
DESIGN

In our final project on Cardiovascular Disease Prediction, we undertook a comprehensive evaluation to gauge the performance of three distinct machine learning models: K-Nearest Neighbors (KNN), Random Forest Classifier, and Gradient Boosting. Our primary objective was to develop a practical predictive tool capable of identifying individuals at risk of cardiovascular diseases. Employing a pragmatic, data-driven approach, we utilized Exploratory Data Analysis (EDA) to unveil underlying patterns within the dataset. Through a systematic comparative study, we assessed the efficacy of each model, considering their respective strengths and limitations. Our findings aim to provide a comprehensive understanding of the predictive capabilities of these models, offering valuable insights for potential healthcare applications and future research endeavors.

DATASET

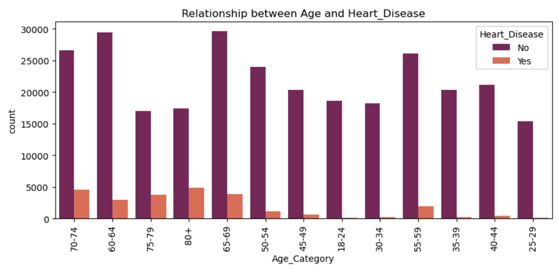
The Behavioral Risk Factor Surveillance System (BRFSS) is a cornerstone of US health surveillance, gathering data via phone surveys on residents' behaviors, conditions, and service utilization. Its dataset includes general health, checkup, general health - checkup, exercise, heart disease, skin cancer, other cancer, depression, diabetes, arthritis, sex, age category, height, weight, BMI, smoking history, alcohol consumption, fruit consumption, green vegetables consumption, fried potato consumption, and height. This comprehensive data informs policymakers, researchers, and public health officials, facilitating targeted interventions and resource allocation to address health disparities and promote well-being nationwide. By providing insights into lifestyle patterns and disease prevalence, the BRFSS plays a crucial role in shaping public health initiatives and guiding efforts to improve health outcomes across diverse populations. Its adaptive approach ensures that it remains a vital tool in understanding and addressing the evolving health challenges facing communities throughout the United States.

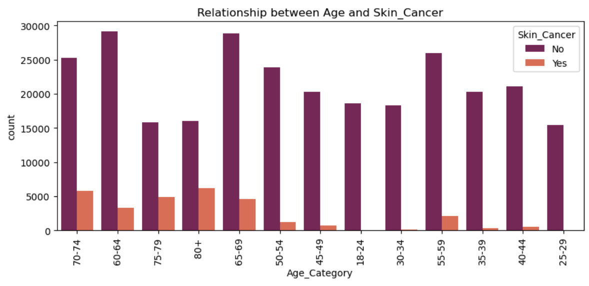
EDA

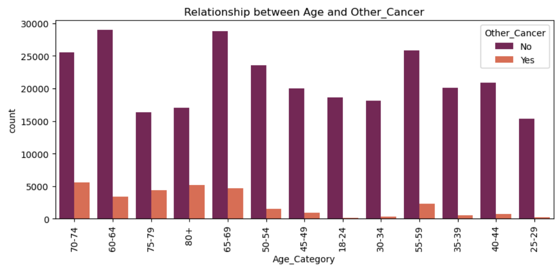


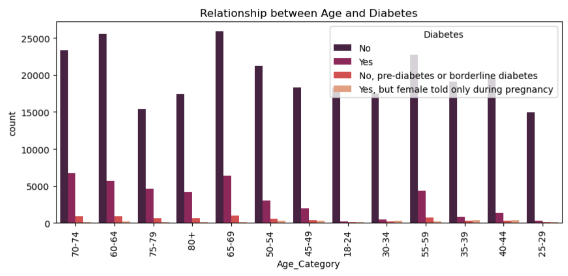
The image presents a correlation heatmap for all the columns in the dataset. The heatmap is a matrix of squares, where the color intensity indicates the strength and direction of the correlation between each pair of variables. This map can be used to identify which factors are most closely related to general health, and to identify potential areas for intervention or further study. For example, the strong positive correlation between age category and diseases suggests that age may be an important factor for risk of the diseases. Negative correlation between self-rated General_Health and diseases underscores the impact of patient perception; those rating health poorly are more likely to have chronic conditions, possibly due to symptom management. Correlations highlight gender-specific disease patterns; for instance, heart disease is more common in males, while skin cancer prevalence is higher in females. It's important to remember correlation doesn't imply causation; further studies are necessary to establish causal relationships between variables.

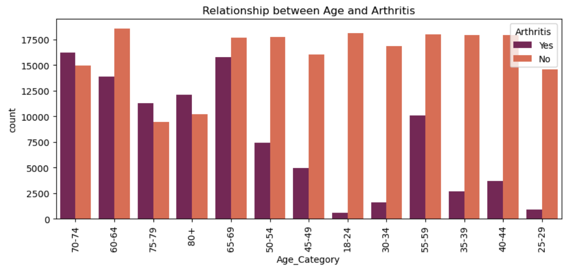
So, we have further plotted age vs each disease plot.

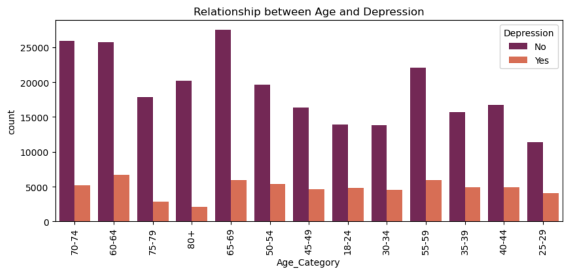












The positive correlation between Age_Category and diseases aligns with medical knowledge, reflecting increased risk due to factors like cumulative exposure and physiological changes.

Data Preprocessing

We have implemented several data preprocessing steps to ensure the quality and usability of your dataset. Here's an explanation of each step:

1. Removal of Null Values, Duplicates, and Outliers:

- Null values: Any missing data points in your dataset can skew the analysis or model training. Removing them ensured the integrity of the data.
- Duplicates: Duplicate entries can lead to biased results or over-representation of certain observations. Removing duplicates maintained the uniqueness of your dataset.
- Outliers: Outlying data points can significantly affect statistical analyses or machine learning models. Detecting and removing outliers helped in obtaining more reliable and accurate results.

2. Categorical Encoding:

- BMI Categorization: BMI (Body Mass Index) is categorized into groups like underweight, normal weight, overweight, and obese for easier analysis and interpretation.
- Checkup Frequency Mapping: We mapped different frequencies of health checkups (e.g., annually, biannually, sporadically) to numerical values for analysis.
- One-hot Encoding for 'Sex' Column: Converted categorical data (e.g., male, female) into binary format (0s and 1s) to facilitate machine learning algorithms' understanding.

- Binary Encoding for Disease Columns: Encoded disease columns (e.g., heart disease, diabetes) into binary format to represent their presence or absence, simplifying analysis.
- Ordinal Encoding for 'General Health', 'Age Category, and 'BMI Category': Assigned numerical values to categorical data based on their order or rank, enabling numerical analysis and comparison.

Implementing these preprocessing steps ensured that the dataset is clean, standardized, and suitable for further analysis or modeling tasks in your project.

MACHINE LEARNING MODELS IMPLEMENTATION:

K-Nearest Neighbours (KNN)

KNN is a simple machine learning algorithm which can be used for classification and regression. KNN is a non-parametric algorithm which makes predictions based on the similarity of the input data to training data. It refers to the K nearest neighbors to make predictions. Different distance metrics such as Euclidean, Manhattan, and Minkowski distance can be used in a KNN implementation. The target variables of the K-Nearest Neighbors based on the selected distance are used to predict the target variable of the input data, mode of the target variable of the K-Nearest Neighbors for classification and mean of the target variable of the K-Nearest Neighbors for regression.

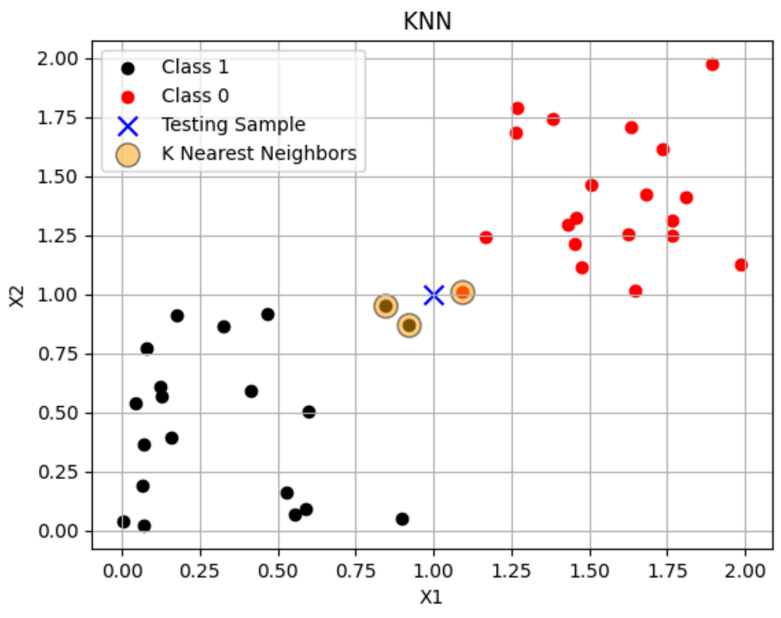


Fig.KNN Implementation

KNN stores the training data points, when a new, unlabeled data point is given, KNN finds the K nearest training points to that point based on the distance metric. For classification, KNN assigns the majority class label among the K nearest neighbors to the new data point. For regression, KNN predicts the average or weighted average of the target values of the K nearest neighbors. KNN is used when the data has a complex decision boundary that is difficult to model with parametric methods. It is suitable for multi-class classification problems. KNN is useful when the input data is not linearly separable. It is also useful when the training data is small, as KNN doesn't require a separate training phase. We implemented KNN to perform classification on our heart disease dataset. We started with encoding the sex column from label to a binary (0,1), and then performed normalization using scipy's standard scaler to normalize the effect of all columns while calculating the distance. We have run a version of KNN on GPU to minimize the high inference time of KNN. The dataset is split into training and test sets using train_test_split() from scikit-learn, with 20% of the data reserved for testing. The training and test data are converted to

PyTorch tensors and moved to the GPU using .cuda(). The KNN model is defined as a PyTorch module. It takes the number of nearest neighbors (k) as a parameter. The forward() method computes the squared Euclidean distances between each test example and all training examples, finds the indices of the k nearest neighbors, and performs majority voting to predict the class label for each test example. The train() method stores the training data and labels. The model is trained by calling the train() method with the training data and labels. Predictions are made on the test set by iterating over the test data loader, passing each batch through the model, and storing the predictions. The classification report and accuracy score are computed using the classification_report() and accuracy_score() functions from scikit-learn. We performed 5 fold cross-validation on this dataset for K-NN with K values from 1-20. We finalized on k=11 as it had the highest average accuracy.

Random Forest

In approaching the problem of predicting heart disease using machine learning techniques, our strategy revolved around leveraging the power of Random Forest, a versatile ensemble learning method known for its robust performance across various domains. For our modeling phase, we opted for a Random Forest Classifier due to its ability to handle complex datasets and mitigate overfitting through the use of multiple decision trees. We implemented the Random Forest algorithm using the scikit-learn library in Python, employing default hyperparameters initially. This choice was informed by the algorithm's effectiveness in handling both categorical and numerical data, making it suitable for our heterogeneous dataset.

To address class imbalance and enhance model performance, we explored the application of Synthetic Minority Over-sampling Technique (SMOTE) to generate synthetic samples of the minority class. This technique aimed to alleviate the skewness in the dataset and improve the model's ability to generalize to unseen data. We visualized the effectiveness of SMOTE by comparing the distribution of classes before and after oversampling.

We conducted hyperparameter tuning using RandomizedSearchCV to identify the optimal combination of hyperparameters for our Random Forest model. By defining a grid of hyperparameters and conducting a randomized search over a specified number of iterations, we aimed to find the configuration that maximized the model's performance metrics, such as accuracy, precision, and recall. Our implementation utilized various tools and libraries, including pandas for data manipulation, scikit-learn for model building and evaluation, and matplotlib for visualizing decision trees. We also employed memory_profiler to monitor memory usage during model training and prediction, providing insights into resource consumption.

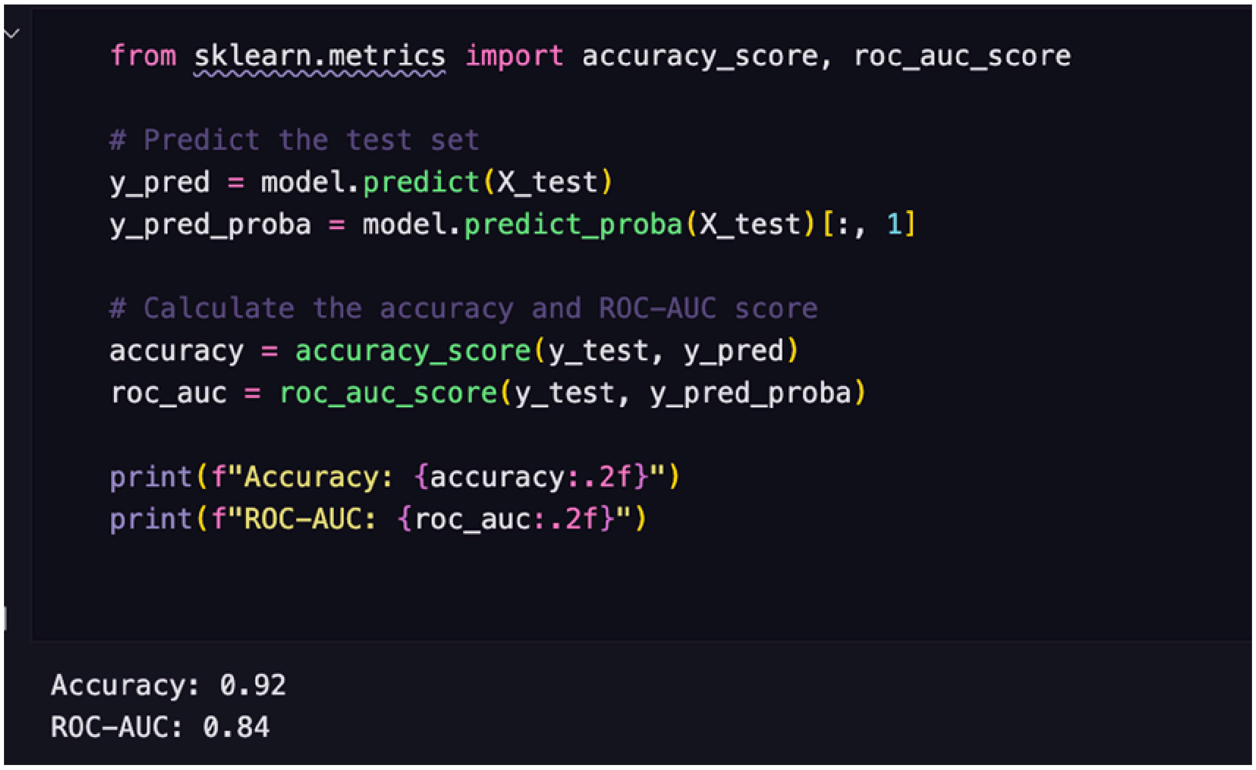
Throughout our experimentation, we visualized decision trees generated by the Random Forest classifier to gain interpretability into the model's decision-making process. Additionally, we analyzed feature importance to identify the most influential variables driving predictions, facilitating domain understanding and feature selection. In summary, our approach encompassed data preprocessing, model selection, hyperparameter tuning, and performance evaluation, with a focus on enhancing model robustness and interpretability. By leveraging Random Forest and employing techniques such as SMOTE and hyperparameter optimization, we aimed to develop a predictive model capable of accurately identifying instances of heart disease.

Gradient Boosting Classifier

We implemented a gradient boosting algorithm which is one of the most effective machine learning strategies that can handle classification and regression jobs with ease.

Model Development with XGBoost:

We use a tool called XGBoost, a fast and powerful way to implement gradient boosting. We began by loading the preprocessed data and splitting it into features and target variables (X and Y). Then, it is divided into training (X_train and Y_train) and testing (X_test and Y_test) sets to facilitate model training and evaluation. Then we start with a basic setup to see how well it does and use specific measures, like accuracy and ROC-AUC, to see if it's good at identifying both the sick and healthy correctly.



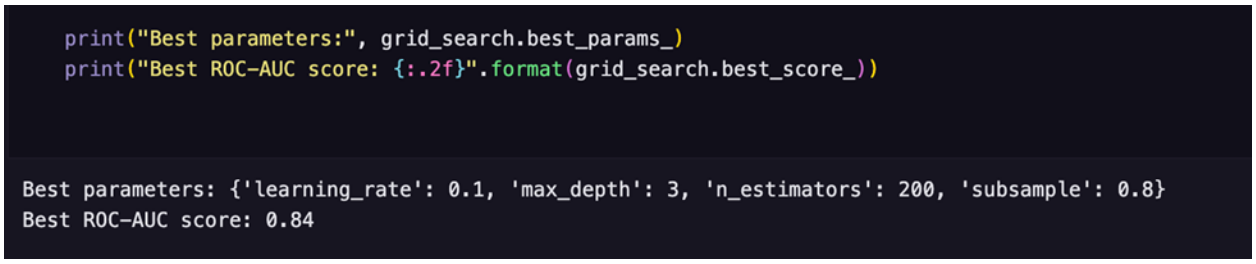
```python
from sklearn.metrics import accuracy_score, roc_auc_score

# Predict the test set
y_pred = model.predict(X_test)
y_pred_proba = model.predict_proba(X_test)[:, 1]

# Calculate the accuracy and ROC-AUC score
accuracy = accuracy_score(y_test, y_pred)
roc_auc = roc_auc_score(y_test, y_pred_proba)

print(f"Accuracy: {accuracy:.2f}")
print(f"ROC-AUC: {roc_auc:.2f}")
```

```
Accuracy: 0.92
ROC-AUC: 0.84
```

We assess the model's performance using two key metrics: accuracy, representing the proportion of correctly classified instances, and ROC-AUC score, measuring the model's ability to distinguish between positive and negative cases. The results indicate a high accuracy (92%) and a decent ROC-AUC score (84%). This suggests that the model is well-tuned and performs robustly on the test data.

Model Tuning and Optimization

To optimize the model's performance, we conduct hyperparameter tuning using GridSearchCV, exploring variations in parameters such as the number of trees (n_estimators), maximum depth of trees (max_depth), learning rate, and subsampling ratio (subsample). This method is exhaustive and considers all parameter combinations, which ensures finding the most optimal settings for the model. This grid search is conducted with cross-validation to ensure that the tuning process is rigorous and generalizable. The output from GridSearchCV shows the best parameters and the associated ROC-AUC score.



```
print("Best parameters:", grid_search.best_params_)
print("Best ROC-AUC score: {:.2f}".format(grid_search.best_score_))
```

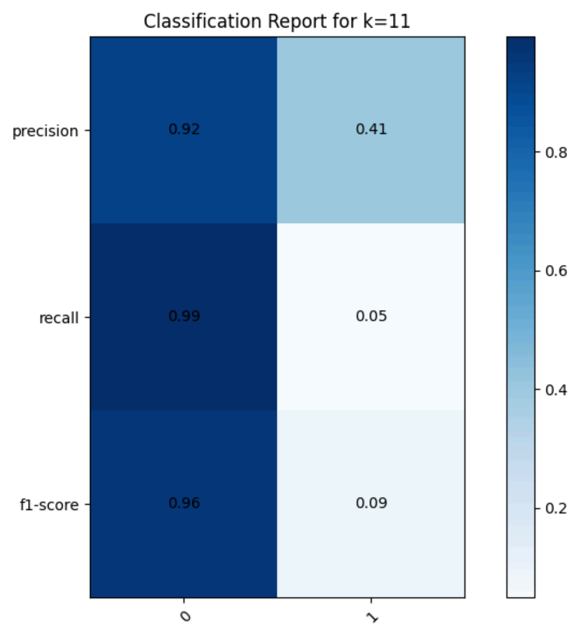
```
Best parameters: {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200, 'subsample': 0.8}
Best ROC-AUC score: 0.84
```

Based on the model's output, we observe a high overall accuracy of 92%, which indicates that the model successfully predicts the correct outcome in 92 out of every 100 cases, suggesting effective general performance. The ROC-AUC score, at 0.84, further supports this, showing a strong ability to distinguish between patients with and without heart disease.

Overall, the results indicate that the model is effective. And the stability in its predictions is a good sign of its robustness.

RESULTS & ANALYSIS

These are results of different models represented in the corresponding classification reports. The classification report is a commonly used evaluation metric in machine learning, particularly for classification problems. It provides a comprehensive overview of the performance of a classification model by presenting various metrics for each class in the dataset. The classification report typically includes the following aspects: Precision: Precision measures the accuracy of positive predictions made by the model. It is calculated as the ratio of true positive predictions to the total number of positive predictions. A high precision indicates that when the model predicts a particular class, it is highly likely to be correct. Precision is useful when the cost of false positives is high, and we want to minimize them. Recall measures the model's ability to correctly identify positive instances. It is calculated as the ratio of true positive predictions to the total number of actual positive instances. A high recall indicates that the model is able to correctly identify a large portion of the positive instances. Recall is important when the cost of false negatives is high, and we want to minimize them. F1-score: The F1-score is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall. The F1-score is calculated as: 2 * (precision * recall) / (precision + recall). It is useful when we want to find a balance between precision and recall and when the class distribution is uneven.



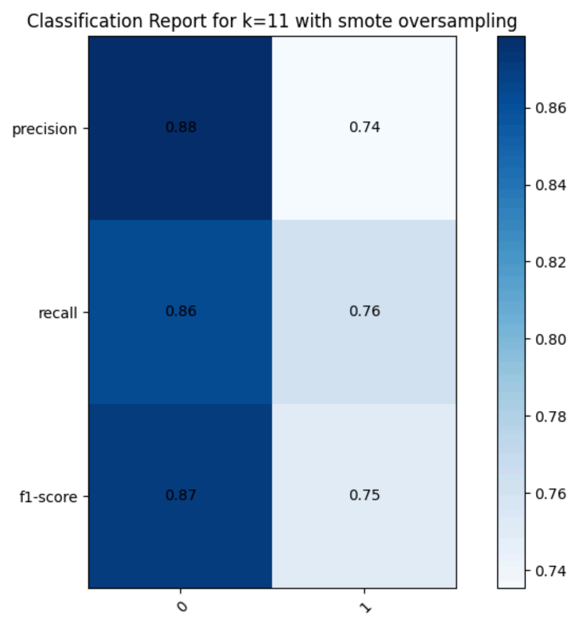


Fig. Classification reports for KNN with K=11

The accuracy of KNN with k=11 is 80%. Although a decent accuracy, the model fails to predict the class of heart disease with a poor accuracy of 41%. This is mainly due to the skewed nature of the dataset with a high number of data points without any heart disease. We implemented a SMOTE oversampling on the dataset, which generates random samples of the minority class. The results after oversampling are comparatively better with an overall accuracy of 85 % and the heart disease class accuracy is also comparatively a lot better.

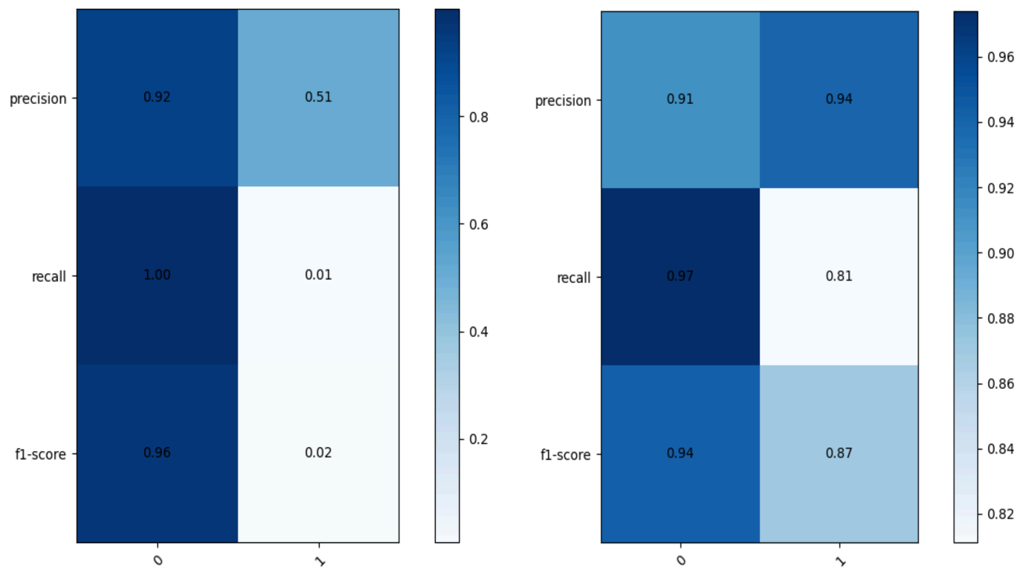


Fig. Classification reports for Random forest

The accuracy of random forest is 86%. Although a decent accuracy, the model fails to predict the class of heart disease with a poor accuracy of 51%. The results after oversampling are comparatively better with an overall accuracy of 92 % and the heart disease class accuracy is also comparatively a lot better. This model performs better compared to the KNN for all cases.

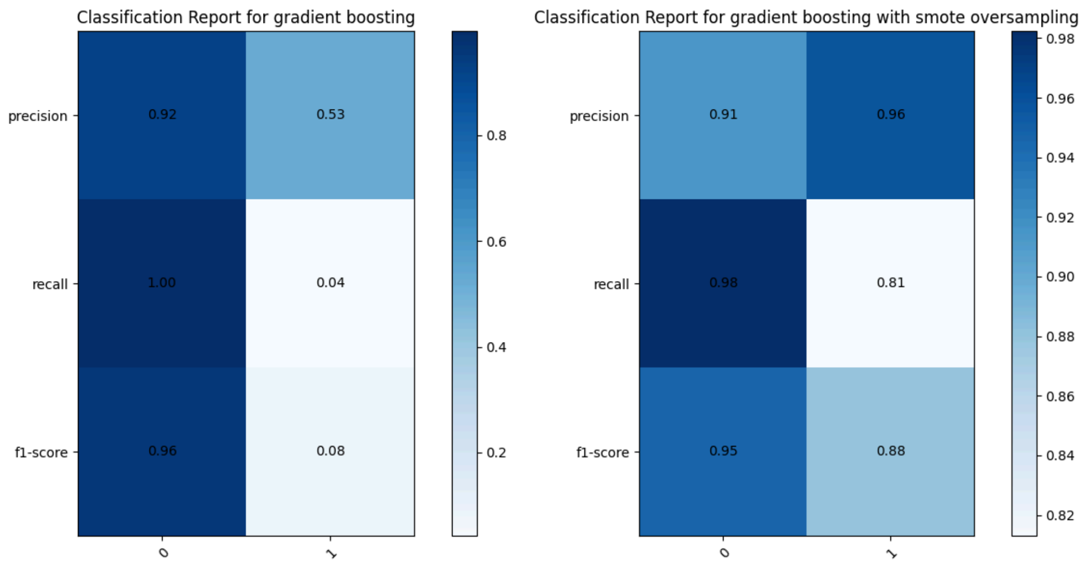
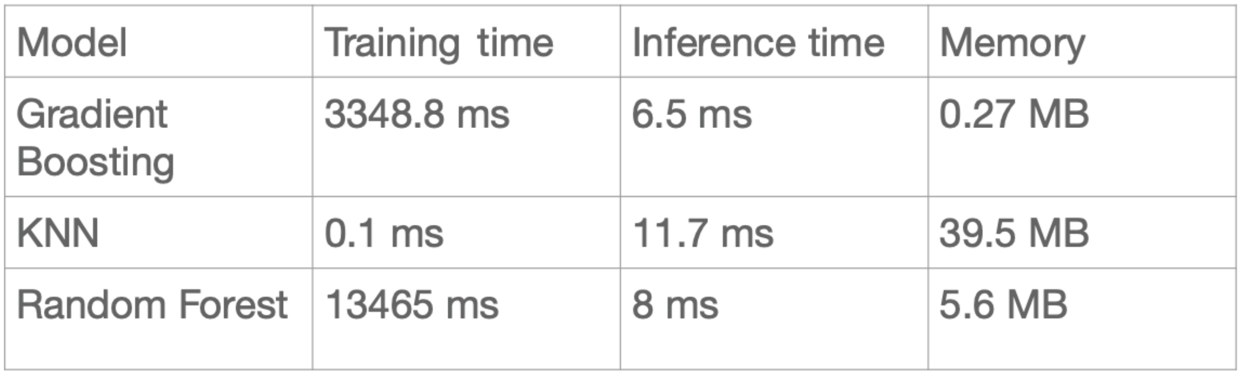


Fig. Classification reports for gradient boosting

The accuracy of gradient boosting is 88%. Although a decent accuracy, the model fails to predict the class of heart disease with a poor accuracy of 53%. The results after oversampling are comparatively better with an overall accuracy of 95 % and the heart disease class accuracy is also comparatively a lot better. This is the best performing model of all the three models.

CONCLUSION & FUTURE WORK



Model	Training time	Inference time	Memory
Gradient Boosting	3348.8 ms	6.5 ms	0.27 MB
KNN	0.1 ms	11.7 ms	39.5 MB
Random Forest	13465 ms	8 ms	5.6 MB

KNN has almost no training time as it is a non-parametric model and it can be validated with the above table of observations. KNN also has a high inference and high memory as it stores all the training data points and calculates the distance between the input and all training points during inference. Gradient Boosting is the best model as it achieves high accuracy comparable to Random Forest while having significantly lower training time, and memory usage. For future work, we would like to focus on methods to improve the bias in the dataset , and how to further optimize the efficiency of all the algorithms. We should learn more about oversampling, optimizing different hyperparameters of the model and how each of them affects model performance and efficiency.

A brief summary of each team member’s contributions to this project.

Venkata Pratheek Reddy Chavva:

Pratheek focused on exploratory data analysis (EDA) and data preprocessing stages of the project. His contributions included conducting thorough data exploration, visualizing relationships between variables, robust treatment of missing values, duplicates, and outliers, as well as encoding categorical features. Pratheek's work ensured that the dataset was properly prepared for accurate predictive modeling, laying the foundation for accurate predictions.

Uday Nair:

Uday was responsible for building the Random Forest Classifier and implementing SMOTE (Synthetic Minority Over-sampling Technique) to handle dataset imbalance. This involved training the Random Forest model, tuning hyperparameters using Randomized Search CV, and ensuring that the model effectively addressed the class imbalance issue inherent in the datasets. Uday's contributions also extended to optimizing the Random Forest model's performance.

Mounika Boggada:

Mounika led the development of the Gradient Boosting model for cardiovascular disease prediction. This involved implementing the Gradient Boosting algorithm, tuning hyperparameters using techniques like GridSearchCV, and evaluating the model's performance. Mounika also contributed to the overall project design and provided insights into how the Gradient Boosting model performed in the predictive modeling pipeline with and without the SMOTE sampling on the dataset.

Pranav Veldurthi:

Pranav led the development of the K-Nearest Neighbors (KNN) model and conducted comparative analysis of the different models. His contributions likely included implementing the KNN algorithm, determining the optimal number of neighbors (k), and evaluating the model's performance. Pranav also played a key role in assessing the strengths and weaknesses of each model, providing valuable insights into their relative effectiveness for the prediction.

Overall, each team member made significant contributions to different stages of the project, ranging from data preprocessing and exploratory analysis to model development, tuning, and evaluation. The collective efforts resulted in the successful creation of a predictive model for forecasting cardiovascular disease.