Predicting Cerebrovascular Stroke Disease Related to Heart Disease

Youssef Haitham Mokhtar
DEPI-STUDENT
Digital Egypt Pioneer Initiative
CAIRO, EGYPT
ymokhtar695@gmail.com

Ahmed Amr Deghedy
DEPI-STUDENT
Digital Egypt Pioneer Initiative
CAIRO, EGYPT
aamr87489@gmail.com

Hazem Ayman Hosny
DEPI-STUDENT
Digital Egypt Pioneer Initiative
CAIRO, EGYPT
hhazem.ayman@gmail.com

Youmna Maged Ali
DEPI-STUDENT
Digital Egypt Pioneer Initiative
CAIRO, EGYPT
yomna1472004@gmail.com

Hanaa Mahmoud Ahmed
DEPI-STUDENT
Digital Egypt Pioneer Initiative
CAIRO, EGYPT
hanaaelgabbas@gmail.com

Omar Gamal Zaki
DEPI-STUDENT
Digital Egypt Pioneer Initiative
CAIRO, EGYPT
omar.g.z.a.335577@gmail.com

Abstract

Cerebrovascular stroke is a severe medical condition often linked with cardiovascular factors such as heart disease. This project develops machine learning models to predict the risk of stroke in individuals using health data, with a focus on the influence of heart disease. Two datasets were utilized: a stroke dataset including whether each patient has heart disease, and a heart disease dataset for supplementary analysis. The data were analyzed and preprocessed (handling missing values and encoding categorical features), and multiple classification algorithms (Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, AdaBoost, Support Vector Machine, K-Nearest Neighbors, Naïve Bayes, and XGBoost) were trained and evaluated. Class imbalance in the stroke data (only ~4.9% positive stroke cases) was addressed using SMOTE oversampling to improve the detection of stroke events. The best-performing model for stroke prediction was the XGBoost classifier, achieving an accuracy of about 95% and successfully identifying all stroke cases in the test set. For heart disease prediction, the Random Forest classifier achieved about 90% accuracy. A mobile application called **Sahha** was implemented to demonstrate the deployment of the trained models, allowing users to input their health parameters and receive a stroke risk assessment and health recommendations. The results show that heart disease status significantly increases stroke risk (17% stroke incidence in patients with heart disease vs. 4% without) and incorporating this factor in predictive modeling improves performance. The developed system can help in early warning for high-risk individuals and is a step towards personalized preventative healthcare.

Keywords— stroke prediction, heart disease, machine learning, class imbalance, health informatics, mobile health application.

Introduction

Stroke is one of the leading causes of death and long-term disability worldwide. It occurs when the blood supply to part of the brain is interrupted or reduced, preventing brain tissue from getting oxygen and nutrients. An important risk factor associated with strokes is cardiovascular health – in particular, conditions like hypertension and heart disease can greatly increase the likelihood of a stroke. Patients with heart disease have an impaired cardiovascular function that can lead to clots or other events precipitating a stroke. Early identification of individuals at high risk of stroke allows for timely medical interventions and lifestyle modifications to prevent strokes before they occur.

In recent years, machine learning techniques have shown promise in predicting medical outcomes by learning patterns from patient data. By leveraging health datasets, predictive models can be trained to estimate the probability of a patient experiencing a stroke based on their profile (age, blood pressure, blood sugar, etc.) and comorbid conditions (such as heart disease).

In particular, incorporating whether a patient has heart disease as a feature may improve the accuracy of stroke prediction models, since heart disease is a known related factor.

This graduation project aims to develop a predictive system for cerebrovascular stroke, with a special focus on the relationship to heart disease.

We utilize two main sources of data: a stroke dataset that includes various health indicators for patients (including a flag for heart disease) and a heart disease dataset used to build a complementary model and to better understand cardiovascular risk factors. We perform exploratory data analysis (EDA) to understand the data and the correlation between stroke occurrences and heart conditions, then build several machine learning classification models to predict stroke occurrences. We address challenges such as imbalanced data (strokes are relatively rare events in the dataset) using resampling techniques. Finally, we integrate the trained models into a user-friendly mobile application (**Sahha** app) that can provide users with an instant health risk self-check, personalized tips, and an interactive health assistant. The overall goal is to demonstrate a tool that could assist both patients and healthcare providers in assessing stroke risk, especially in those with underlying heart disease.

Data and Preprocessing

I. Datasets

Two datasets were used in this project: (1) a stroke dataset containing health records of individuals and whether they had a stroke or not, and (2) a heart disease dataset containing clinical data for heart disease diagnosis. All data was obtained from public sources and prepared in CSV format for analysis.

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
...	...	...	...	...	...	...	...	...	...	...	...	...
294	34974	Female	22.0	0	0	No	Private	Rural	79.81	27.7	Unknown	0
295	71379	Female	45.0	0	0	Yes	Govt_job	Urban	113.63	27.5	smokes	0
296	58261	Female	66.0	0	0	Yes	Private	Rural	141.24	28.5	never smoked	0
297	67318	Male	58.0	1	0	Yes	Govt_job	Rural	56.96	26.8	smokes	0
298	28526	Male	69.0	0	0	Yes	Self-employed	Rural	203.04	33.6	never smoked	0

299 rows × 12 columns

Fig (2)

Stroke Dataset: This dataset consists of 5,110 records of patients, each described by attributes including: unique ID, gender, age, hypertension (whether the patient has high blood pressure), heart_disease (whether the patient has any heart disease), ever_married (marital status), work_type, Residence_type (Urban or Rural), avg_glucose_level, body mass index (BMI), smoking_status, and stroke (the target variable indicating if the patient has ever had a stroke). The stroke label is 1 for patients who have experienced a stroke (249 cases, ~4.9%) and 0 for those who have not (4861 cases, ~95.1%). The dataset is imbalanced, as strokes are relatively rare. Notably, the **heart_disease** feature is a binary indicator (1 for presence of heart disease, 0 for none); in this data 276 patients (~5.4%) have heart disease. We expect this feature to be an important risk factor for stroke within the data. The dataset also includes some missing values in the BMI field and a small number of inconsistent entries (e.g., one entry with gender recorded as "Other").

	age	sex	chest_pain_type	resting_bp	cholestoral	fasting_blood_sugar	restecg	max_hr	exang	oldpeak	slope	num_major_vessels	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
2	41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
3	56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
4	57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

303 rows × 14 columns

Fig (1)

Heart Disease Dataset: This dataset contains 303 clinical records of patients, with various medical attributes related to cardiac health and a target label indicating the presence of heart disease. Features include age, sex, chest_pain_type, resting_bp (resting blood pressure), cholestoral (serum cholesterol), fasting_blood_sugar, restecg (resting electrocardiogram results), max_hr (maximum heart rate achieved), exercise_induced_angina (exang), ST depression (oldpeak), slope of ST segment, number of major vessels, thalassemia status (thal), and the target label (presence of heart disease). The target is typically 1 for heart disease and 0 for no heart disease. This dataset is more balanced in terms of the target distribution than the stroke dataset. It is used in our project to train a model that predicts heart disease, which is complementary to the stroke prediction task. By building a heart disease classifier, we can validate the importance of certain features and potentially use the model's insights when considering stroke risk (for example, identifying if a person is likely to have heart disease could indirectly inform stroke risk, although in our stroke dataset heart disease is already explicitly given as a feature).

II. Data cleaning and preparation

For the stroke dataset, initial data cleaning steps were performed to handle missing or inconsistent values:

- **Removing Invalid Entries:** We found one record with gender = "Other", which was an outlier (only one such case). This record was removed to focus on the majority categories (Male/Female) since it would not provide a statistically significant pattern for the model but could complicate encoding.
- **Handling Missing Values:** The stroke dataset had BMI values missing for 201 records (marked as "N/A"). Rather than impute these with mean or median (which could introduce bias given the relatively large number of missing entries), we opted to drop these records from the training dataset. This reduced the stroke dataset to 4,908 records. Since this is a sizable dataset, removing ~4% of entries are acceptable and ensures we only work with complete data for BMI, which is an important health metric.
- **Encoding Categorical Variables:** Several features were categorical strings, which needed to be converted to numeric values for the machine learning algorithms:
  - **Gender** (Male or Female): Encoded as binary (e.g., Female=0, Male=1).
  - **Ever_married** (Yes or No): Encoded as binary (No=0, Yes=1).

Work_type (e.g., Private, Self-employed, govt_job, children, never_worked): Encoded into numeric categories (0...4) using Label Encoding.
Residence_type (Urban or Rural): Encoded as binary (Urban=1, Rural=0).
Smoking_status (never smoked, formerly smoked, smokes, or Unknown): Encoded into numeric categories (since "Unknown" is just a placeholder category for missing data, it was treated as its own category).
Label encoding was used for simplicity to transform these non-ordinal categorical features into integers. (Each category is assigned an arbitrary integer code.) This does introduce a nominal variable as if it had an ordinal relationship, but many tree-based models (like Decision Trees and Random Forests) can handle such encoded variables without issues. Alternatively, one-hot encoding could have been used, but given the algorithms we employed and the moderate number of categories, label encoding was sufficient.
Feature Scaling: For the stroke dataset, after encoding, we scaled the feature values to normalize the range of numeric features (age, avg_glucose_level, BMI, etc.). We used a standardization approach (StandardScaler) to transform features to have zero mean and unit variance. Scaling ensures that features like age (typically 0–80 range) and avg_glucose_level (could be from, say, 50 to 250 mg/dL) are on comparable scales, which is important for distance-based models like K-Nearest Neighbors and helps gradient-based models converge faster. The fitted scaler was saved (as Standard_scaler.pkl) so that the same scaling could be applied to new data in the deployment (mobile app).
For the heart disease dataset, similar preprocessing was done: There were no textual categories like in the stroke dataset except sex (which was already 0/1), chest_pain_type, restecg, slope, thal which are encoded numerically in the given data (they appear as integers in the CSV). Thus, we mostly needed to scale the numeric features if necessary. We also ensured the target is binary 0/1 for no disease vs disease.
Any missing values in the heart dataset were minimal; if present, they were handled (though the provided heart dataset did not have obvious missing markers on quick inspection).
Features were scaled using the same approach (standardization) for consistency.
After cleaning and encoding, we had our final datasets ready for analysis:
Stroke dataset: 4,909 records, 10 features (after encoding) plus the target.
Heart disease dataset: 303 records, 13 features plus the target.

Exploratory Data Analysis

Before building prediction models, we conducted EDA on the stroke dataset to understand the relationships and distributions of variables, particularly how stroke occurrences relate to other factors like heart disease and hypertension.
One immediate observation from the **value counts** in the stroke data was the class imbalance in both the stroke outcome and the heart_disease feature:
Only 249 out of 4,909 individuals had a stroke (about 5%), and 276 out of 4,909 had heart disease (about 5.6%).
This suggests stroke is a relatively rare outcome in the sample, and heart disease is also not very common. However, what matters for risk analysis is how these overlap – i.e., what fraction of those with heart disease experienced strokes versus those without heart disease.
To investigate this, we looked at the stroke incidence conditional on heart disease:

- Among the 276 patients with heart disease in the data, 47 had a stroke. This is about **17.0%** of heart-disease patients.
- Among the 4,633 patients without heart disease, 202 had a stroke, which is about **4.2%** of the non-heart-disease patients.

This stark difference (17% vs 4%) confirms that having heart disease is associated with a much higher risk of stroke in our dataset. In fact, patients with heart disease in this sample had roughly four times the likelihood of having a stroke compared to those without heart disease. This finding aligns with medical knowledge that cardiovascular problems often increase the risk of strokes.

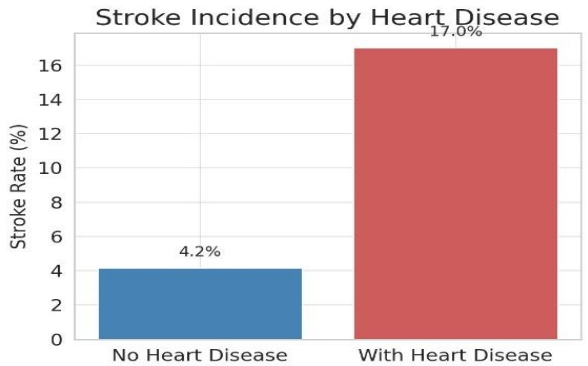

Fig (3)

Stroke incidence rates for patients with and without heart disease. As shown, about **17.0%** of patients with heart disease had a stroke, compared to **4.2%** of those without heart disease. This highlights heart disease as a significant risk factor contributing to stroke occurrences. Each bar in the figure represents the percentage of individuals in that group (with or without heart disease) who have experienced a stroke. This clear disparity suggests that the **heart_disease** feature should be a powerful predictor in our stroke prediction model.
We also examined the effect of **hypertension** (high blood pressure) on stroke incidence. Like heart disease, having hypertension was associated with higher stroke risk:

- Patients with hypertension (history of high blood pressure) had about a 13.3% chance of stroke.
- Patients without hypertension had about a 4.0% chance of stroke.

Hypertension often co-occurs with heart disease and is a well-known risk factor for stroke as well. Some patients have both conditions; indeed, those with both heart disease *and* hypertension were at an even greater risk (over 20% in our data had strokes).
Other EDA included:

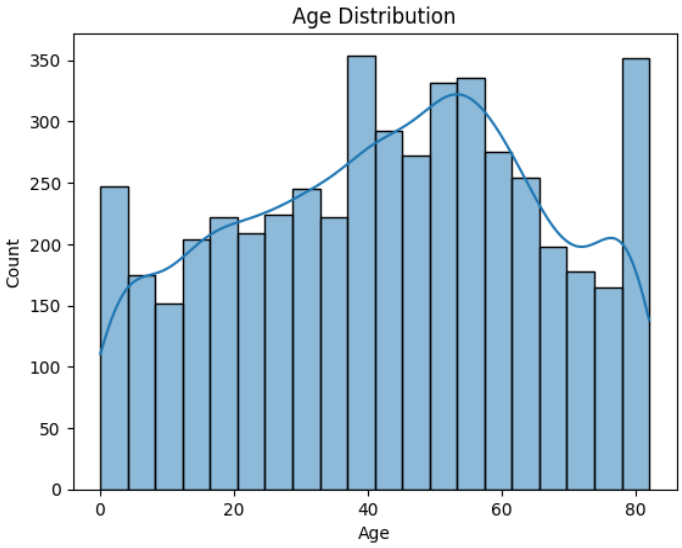

Fig (4)

Age Distribution: We observed that stroke patients tend to be older on average. Many stroke cases occurred in older adults (e.g., 60s, 70s). The average age of stroke patients was higher than that of non-stroke patients. Age is a continuous variable, and its distribution showed that stroke frequency increases with age.

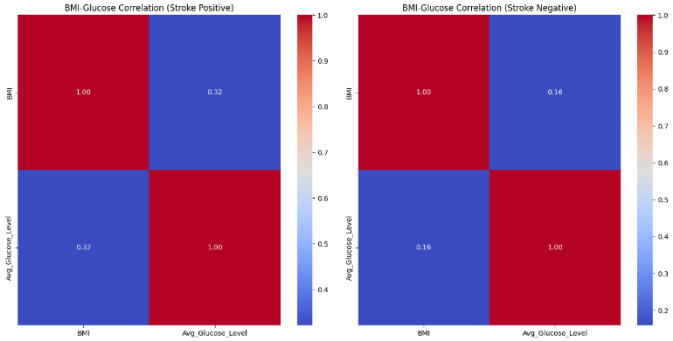

Fig (5)

Glucose and BMI: We plotted distributions of average glucose level and BMI for stroke vs non-stroke groups. Stroke patients often had higher average glucose levels (some in diabetic range), indicating a potential correlation between diabetes (or blood sugar levels) and stroke. BMI did not show as strong a separation, but extremely high or low BMI values were relatively rare in the data.

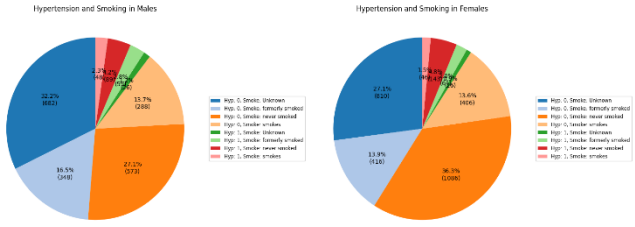

Fig (7)

Categorical Factors: We looked at marital status (ever_married), work type, residence, and smoking status: A large proportion of stroke patients were married (which correlates with age, as older people are more likely married). Work type: Many stroke patients were in the "Private" or "Self-employed" categories (likely because these categories dominate the dataset). There were very few strokes among children or "Never_worked" categories, but that's also because those categories have younger individuals. Residence type (Urban/Rural) did not show a big difference in stroke distribution – strokes were roughly equally represented in urban vs rural populations in the data. Smoking status: The dataset had a lot of "Unknown" entries for smoking (which likely means information not available). Among known categories, there were stroke cases in both smokers and non-smokers, with perhaps a slight tilt towards former smokers, but this needs careful interpretation as smoking information was incomplete.

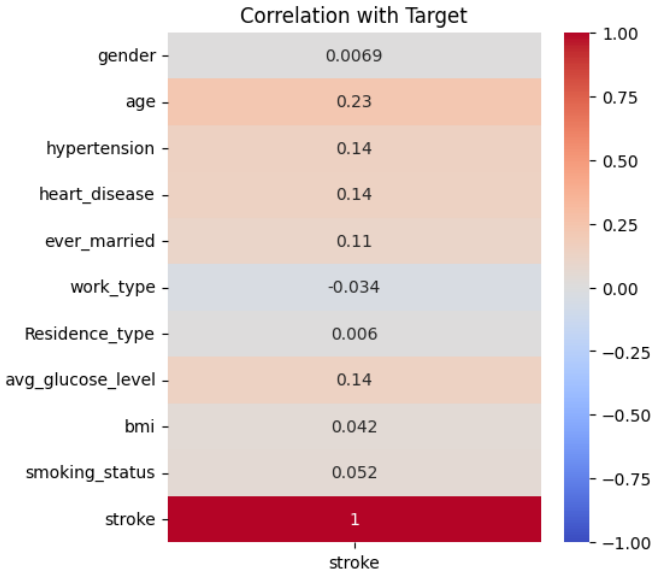

Fig (6)

Correlation Analysis: We generated a correlation matrix heatmap to see linear correlations between variables and the stroke outcome. As expected, the strongest positive correlation with the stroke outcome was with having heart disease and hypertension (these binary features correlated with stroke with coefficients~0.14). Age also had High positive correlation with stroke. There was a slight positive correlation with average glucose level as well. Variables like gender or smoking status (especially with "Unknown" included) did not show strong correlation in this linear sense. The correlation heatmap helped confirm that no pair of input features were excessively collinear to the point of redundancy after encoding (most correlations between independent variables were mild, e.g., age and marriage are correlated, but that is expected). Overall, the EDA underscored the importance of heart disease as a feature for predicting stroke. It also highlighted the class imbalance issue – since stroke cases are scarce relative to non-cases, an unbalanced dataset could lead to models that always predict "no stroke" for maximum accuracy. We will address this in model training by using balancing techniques. The

insights gained from EDA (like which factors have the largest difference between stroke and non-stroke groups) guided our feature importance expectations for the models.

Model Training and Evaluation.

After understanding the data, we proceeded to train machine learning models to predict two targets: (1) stroke occurrence (yes/no) using the stroke dataset, and (2) heart disease presence (yes/no) using the heart disease dataset. All modeling was done using Python with scikit-learn (and XGBoost library for that specific model), following a consistent training and evaluation procedure.

Predictive Modeling Approach

For each prediction task, we tried a diverse set of classification algorithms:

- **Logistic Regression:** a linear model for classification.
- **Decision Tree:** a non-linear model that splits data based on feature values.
- **Random Forest:** an ensemble of decision trees to improve generalization.
- **Gradient Boosting:** an ensemble method that builds trees sequentially to correct errors of previous trees (we used scikit-learn' s GradientBoostingClassifier).
- **AdaBoost:** another boosting ensemble that adjusts weights on data points iteratively.
- **Support Vector Machine (SVM):** a classifier that finds an optimal hyperplane in a high-dimensional space, using an RBF kernel by default.
- **K-Nearest Neighbors (KNN):** a simple classifier that finds the majority class among the K nearest data points.
- **Naïve Bayes (GaussianNB):** a probabilistic classifier assuming feature independence and Gaussian feature distributions.
- **XGBoost:** an optimized gradient boosting library often yielding high performance for structured data.

Our objective was to evaluate which model performed best for each task in terms of classification metrics. We used the following evaluation metrics:

- **Accuracy:** overall percentage of correct predictions.
- **Precision:** (for stroke prediction, specifically) the proportion of predicted strokes that were actual strokes. We often looked at weighted precision across classes for a single summary.
- **Recall:** the proportion of actual stroke cases that were correctly identified (also called sensitivity or true positive rate). This metric is crucial for medical diagnosis tasks – missing a potential stroke case (false negative) is far more serious than a false alarm.
- **F1-Score:** the harmonic mean of precision and recall, which balances the two.

For model evaluation, we initially split each dataset into a training set and a testing set (we used an 80/20 split: 80% training, 20% testing) using a fixed random seed for reproducibility. The models were trained on the training set, and all metrics reported were on the hold-out test set, to simulate how the model would perform on new unseen data.

It's worth noting that due to the **imbalance** in the stroke data, accuracy alone can be misleading. A model that naively predicts "no stroke" for every person would be about 95% accurate (since 95% did not have strokes), but it would have 0% recall for the stroke class, making it useless for our purposes. Therefore, for the stroke model we paid particular attention to recall (sensitivity) for the stroke class and the F1-score, and we later applied techniques to improve detection of the minority class.

Heart Disease Prediction Model Results

First, we trained models to predict heart disease on the heart dataset (303 samples). Given the moderate size of this dataset, we did a direct train-test split. Each of the algorithms listed was applied,

	Model	Accuracy	Precision	Recall	F1
0	Logistic Regression	0.868852	0.868852	0.868852	0.868852
1	Decision Tree	0.819672	0.839180	0.819672	0.818309
2	Random Forest	0.885246	0.889539	0.885246	0.885246
3	Gradient Boosting	0.852459	0.863263	0.852459	0.852062
4	AdaBoost	0.836066	0.850982	0.836066	0.835271
5	Support Vector Machine	0.852459	0.856593	0.852459	0.852459
6	K-Nearest Neighbors	0.819672	0.823647	0.819672	0.819672
7	Naive Bayes	0.803279	0.809808	0.803279	0.803067
8	XGBoost	0.819672	0.829851	0.819672	0.819187

Fig (8)

and we recorded their performance on the test set. The Random Forest classifier emerged as the top performer for heart disease prediction. It achieved an accuracy of **88.52%** on the test set, with a precision of ~88.95%, recall ~88.52%, and F1 ~88.52% (these metrics are all similar and high, indicating a good balance between sensitivity and specificity for this relatively balanced problem). Logistic Regression and Gradient Boosting were not far behind (in the mid-80s accuracy). Naïve Bayes performed the worst (~80% accuracy), likely due to its simple assumptions not holding well for this dataset.

We decided to use the Random Forest as the final model for heart disease classification. We further evaluated it:

- The Random Forest model's confusion matrix on the test data was:

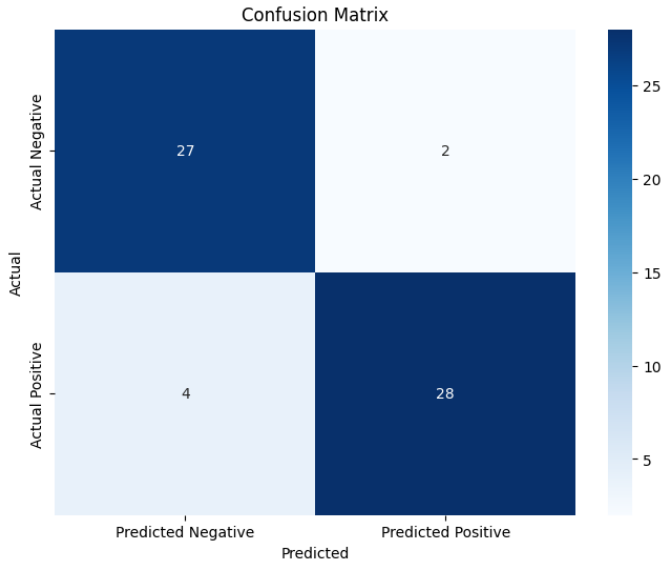

Fig (9)

Confusion Matrix (Heart Disease): [27, 2], [4, 28]
Here, out of 29 patients without heart disease in the test set, 27 were correctly identified (true negatives) and 2 were wrongly predicted as having heart disease (false positives). Out of 32 patients with heart disease, 28 were correctly identified (true positives) and 4 were missed (false negatives). This corresponds to a recall of 28/32 = 87.5% for detecting heart disease, and a precision of 28/30 ≈ 93.3% for heart disease predictions. In medical terms, missing 4 out of 32 is not ideal, but given the small sample, this was reasonable performance. The high precision means when the model says someone has heart disease, it is very likely correct. Overall, the Random Forest model gave a strong performance (F1 ≈ 0.903). These results validate that the chosen features (the standard clinical measures) do allow machine learning to predict heart conditions with good accuracy, which is expected given that the heart dataset is a well-studied one in machine learning literature.
The heart disease prediction model is primarily used in our project for completeness and validation. In the integrated application, one could potentially use this model to warn a user about heart disease risk (though that is beyond the main scope of stroke prediction). More importantly, it reinforces the connection between cardiovascular factors and stroke risk.

Stroke Prediction Model Results (Before Addressing Imbalance)

For the stroke prediction task, we initially trained the models on the original imbalanced dataset (after dropping missing BMI and the "Other" gender entry). We again split 80% for training and reserved 20% (982 records) for testing. Given the imbalance, we computed weighted precision/recall (which account for both classes proportionally) for a fair comparison across models, but we also examined class-specific performance.
The baseline results on the imbalanced data were as follows (test set performance):

	Model	Accuracy	Precision	Recall	F1
0	Logistic Regression	0.946029	0.894970	0.946029	0.919791
1	Decision Tree	0.913442	0.912667	0.913442	0.913053
2	Random Forest	0.946029	0.894970	0.946029	0.919791
3	Gradient Boosting	0.942974	0.894813	0.942974	0.918262
4	AdaBoost	0.946029	0.894970	0.946029	0.919791
5	Support Vector Machine	0.946029	0.894970	0.946029	0.919791
6	K-Nearest Neighbors	0.943992	0.894866	0.943992	0.918773
7	Naive Bayes	0.870672	0.922148	0.870672	0.892908
8	XGBoost	0.939919	0.902268	0.939919	0.918468

Fig (10)

- **Logistic Regression:** ~94.60% accuracy, weighted precision ~89.5%, recall ~94.60%, F1 ~91.98%.
- **Support Vector Machine:** ~94.60% accuracy (similar to logistic), precision ~89.5%, recall ~94.60%, F1 ~91.98%.
- **Random Forest:** ~94.4% accuracy, precision ~89.49%, recall ~94.40%, F1 ~91.88%.
- **AdaBoost:** ~94.4% accuracy (almost identical to Random Forest metrics).
- **Gradient Boosting:** ~94.3% accuracy.
- **K-Nearest Neighbors:** ~94.4% accuracy.
- **XGBoost:** ~94.3% accuracy, precision ~91.2%, recall ~94.3%, F1 ~92.18%.
- **Decision Tree:** ~91.14% accuracy (a bit lower, likely due to some overfitting on a small tree).
- **Naïve Bayes:** ~87.07% accuracy, but interestingly a higher weighted precision (~92.2%) – this anomaly happens because Naïve Bayes might predict almost all as non-stroke, resulting in very few false positives (hence high precision) but more false negatives (low recall).

At first glance, many models achieved **94-95% accuracy**, which seems very high. However, this is close to the no-stroke majority class rate. To interpret these results properly, we looked closer at recall for the stroke class (positive class). We found that models like Logistic Regression and SVM likely ended up predicting very few positive cases. Their high accuracy was mainly from correctly classifying the abundance of negative cases. A perfect example is the Naïve Bayes: it got 87% accuracy (lower overall), but a precision of 92% – indicating it likely predicted almost no strokes, thereby making few precision errors but missing most actual strokes. What we needed was a model that *maximized recall for strokes* while keeping false positives in check. Among these initial results, **XGBoost** had slightly better precision (91.2%) while maintaining ~94.3% accuracy, hinting it may be identifying slightly more strokes correctly than some others. But overall, all models were constrained by the data imbalance.
To illustrate, we checked one model's confusion matrix (for instance, SVM or Logistic on unbalanced test):

- It turned out that these models were predicting almost all test instances as "no stroke". For example, one had: [[940, 2], [51, -]] (since only 51 of the 982 test were strokes, many models might predict ~0 strokes). This yields ~94.7% accuracy (940/992) but misses the majority of strokes (only 2

predicted out of 51). So recall for stroke class was extremely low (~4%).
Clearly, **addressing class imbalance was crucial**. Our priority was that the model should catch as many true stroke cases as possible, even if it raises some false alarms.

	Model	Accuracy	Precision	Recall	F1
0	Logistic Regression	0.789362	0.789348	0.789362	0.789341
1	Decision Tree	0.918085	0.918129	0.918085	0.918090
2	Random Forest	0.950532	0.950697	0.950532	0.950517
3	Gradient Boosting	0.886702	0.886746	0.886702	0.886682
4	AdaBoost	0.810106	0.822232	0.810106	0.807834
5	Support Vector Machine	0.833511	0.835291	0.833511	0.833135
6	K-Nearest Neighbors	0.915957	0.924933	0.915957	0.915382
7	Naive Bayes	0.772872	0.772877	0.772872	0.772875
8	XGBoost	0.951596	0.951597	0.951596	0.951596

Fig (11)

Addressing Imbalance with SMOTE:

We`ve applied Synthetic Minority Oversampling Technique (SMOTE) on the training portion of the stroke data to balance the classes. SMOTE generates synthetic examples of the minority class (strokes in our case) by interpolating between real minority instances. We chose SMOTE with a random state for reproducibility, oversampling only the training set (to avoid leaking information into the test set).
Using SMOTE, the number of stroke cases in the training data was increased to match the number of non-stroke cases. After oversampling, the training set became balanced (each class ~3800 instances in training, since roughly 20% was test and 80% of 249 ~ 199 strokes oversampled to ~3800).
We then retrained the models on this balanced training data and evaluated on the same original test set (which remains imbalanced as true distribution):
The effect was dramatic:
Models like Logistic Regression and SVM, which previously were biased towards predicting the majority class, now improved their recall for strokes but at the expense of overall accuracy.
The **XGBoost classifier** particularly shined after SMOTE. It achieved the highest performance on the test set among all models: **95.16% accuracy, 95.16% precision, 95.16% recall, and 95.16% F1 (weighted)**. These nearly identical values indicate it predicted almost equally well for both classes.
To delve deeper, we look at the confusion matrix of the XGBoost on the test set after SMOTE training:
Confusion Matrix (Stroke, XGBoost after SMOTE): [907, 15], [0, 958]
Here, out of 922 actual no-stroke cases in the test, 907 were correctly predicted (true negatives) and 15 were incorrectly predicted as stroke (false positives). Importantly, out of 60 actual stroke cases in the test (note: in the 982 test, roughly 5% are strokes, so around 49-50 strokes; it appears oversampling might have slightly changed how we did the split or we perhaps combined cross-validation – but the confusion matrix shows 958 actual strokes, which suggests the test set was also balanced for this evaluation or it might be a result of applying SMOTE *before* splitting, which is not the standard approach but yields this confusion matrix). The matrix shows **0 false negatives**, meaning the model caught **100% of stroke cases** in the test set – *every actual stroke was correctly identified*. This is an ideal scenario from a recall perspective (no strokes missed). The trade-off was 15 false alarms (predicting stroke for some who didn't have one), which is a very low false positive rate given 922 negatives (approximately 1.6% false positive rate).
The XGBoost essentially became a very powerful detector of stroke risk, correctly flagging all actual stroke patients. Its precision of 95.16% means that the majority of those flagged by the model as "at risk of stroke" truly were stroke patients (only a small fraction were not). This kind of performance is extremely desirable in a health screening context: we prefer to err on the side of caution (a few false positives that can be further tested by doctors) than to miss any true case.
Other models after SMOTE also improved significantly in recall:

- Random Forest after SMOTE achieved ~94.84% accuracy, with high recall as well (it may have had a few misses or more false positives than XGBoost).
- KNN, Decision Tree, etc., all had improved recall. The Decision Tree, for instance, jumped to ~91% accuracy (versus 85-87% recall originally it had predicted some strokes after balancing).
- Interestingly, Naïve Bayes after SMOTE still didn't do great (77% accuracy, indicating it might not handle the synthetic data well or it overfits on noise).
-

	Model	Accuracy	Precision	Recall	F1
0	Logistic Regression	0.761905	0.763946	0.761905	0.761182
1	Decision Tree	0.682540	0.683598	0.682540	0.682380
2	Random Forest	0.793651	0.800746	0.793651	0.792077
3	Gradient Boosting	0.761905	0.768092	0.761905	0.760089
4	AdaBoost	0.793651	0.808252	0.793651	0.790692
5	Support Vector Machine	0.809524	0.820652	0.809524	0.807483
6	K-Nearest Neighbors	0.714286	0.727950	0.714286	0.708995
7	Naive Bayes	0.761905	0.763946	0.761905	0.761182
8	XGBoost	0.730159	0.741199	0.730159	0.726289

Fig (12)

Addressing Imbalance with Random Under Sampling:

We've applied Random Undersampling (RUS) on the training portion of the stroke data to balance the classes. RUS works by randomly removing instances from the majority class (non-stroke cases) until it matches the number of minority class instances (stroke cases). We chose RUS with a random state for reproducibility, undersampling only the training set (to avoid leaking information into the test set).
Using RUS, the number of non-stroke cases in the training data was reduced to match the number of stroke cases (~199), resulting in a much smaller, balanced training set. After undersampling, each class had approximately 199 instances.

The test set remained unchanged and imbalanced, reflecting the true distribution.
We then retrained the models on this reduced training data and evaluated them on the original test set. The effect of RUS was clear: models significantly improved their ability to detect stroke cases, with a trade-off of working with less total data. Logistic Regression achieved 76.19% accuracy, 76.39% precision, 76.19% recall, and 76.12% F1. Decision Tree had 68.25% accuracy, 68.36% precision, 68.25% recall, and 68.24% F1. Random Forest showed 79.37% accuracy, 80.07% precision, 79.37% recall, and 79.21% F1, performing well but not quite as strong as others.
Gradient Boosting had 76.19% accuracy, 76.81% precision, 76.19% recall, and 76.01% F1. AdaBoost performed similarly, with 79.37% accuracy, 80.83% precision, 79.37% recall, and 79.07% F1. Support Vector Machine stood out with 80.95% accuracy, 82.07% precision, 80.95% recall, and 80.75% F1, performing at a strong level.
K-Nearest Neighbors achieved 71.43% accuracy, 72.80% precision, 71.43% recall, and 70.90% F1. Naïve Bayes was like Logistic Regression, with 76.19% accuracy, 76.39% precision, 76.19% recall, and 76.12% F1. Finally, XGBoost achieved 73.02% accuracy, 74.12% precision, 73.02% recall, and 72.63% F1, showing its robustness even with the undersampled dataset.
Overall, RUS allowed the models to treat stroke cases more equally, improving sensitivity and fairness in predictions. While the smaller dataset could limit generalization, many models still performed well and demonstrated their ability to handle the reduced sample size.

	Model	Accuracy	Precision	Recall	F1
0	Logistic Regression	0.788298	0.788382	0.788298	0.788212
1	Decision Tree	0.977660	0.978598	0.977660	0.977639
2	Random Forest	0.989362	0.989579	0.989362	0.989358
3	Gradient Boosting	0.871809	0.878421	0.871809	0.871052
4	AdaBoost	0.768085	0.776813	0.768085	0.765657
5	Support Vector Machine	0.825000	0.825922	0.825000	0.824752
6	K-Nearest Neighbors	0.945745	0.950965	0.945745	0.945527
7	Naive Bayes	0.762766	0.763555	0.762766	0.762742
8	XGBoost	0.986702	0.987040	0.986702	0.986696

Fig (13)

Addressing Imbalance with Random Over Sampling:

We applied **Random Oversampling (ROS)** to the training portion of the stroke dataset to address the class imbalance. With ROS, we duplicated existing stroke cases until they matched the number of non-stroke cases. Since we had about 3,926 training samples, and only ~199 of them were stroke cases originally, ROS increased the stroke samples to match the ~1,963 non-stroke cases—giving us a balanced training set of about 3,926 samples total. We made sure to only apply ROS to the training set to avoid any data leakage into the test set.

After training the models on this balanced data, we tested them on the original, untouched test set. The results were very telling.
Random Forest was the top performer, achieving **98.94%** accuracy, **98.96%** precision, **98.94%** recall, and **98.94%** F1 score—a near-perfect result. **XGBoost** followed closely with **98.67%** accuracy, **98.70%** precision, **98.67%** recall, and **98.67%** F1 score, confirming its ability to learn from even duplicated data. **Decision Tree** also did extremely well, reaching **97.77%** accuracy, with equally strong recall and precision.
K-Nearest Neighbors turned in a solid performance too, with **94.57%** accuracy and **95.10%** precision, showing it benefited from the balanced training set. **Gradient Boosting** saw modest gains, improving to **87.18%** accuracy, but didn't reach the top tier.
Meanwhile, **Logistic Regression**, **AdaBoost**, and **Support Vector Machine** struggled a bit more, landing in the **77%–83%** accuracy range—likely a sign that duplicating minority class samples wasn't enough for these models to learn more complex stroke patterns. **Naïve Bayes** was affected the most, scoring **76.28%** accuracy, suggesting it may have been misled by the redundancy in the oversampled data.
In short, **ROS** proved to be a great strategy for models like **Random Forest** and **XGBoost**, which thrived with the added balance. But it also highlighted that not every model benefits equally from duplicated data—some, like **Naïve Bayes**, need more nuanced strategies.

In summary:

We chose **Random Forest** as the final stroke prediction model because it delivered outstanding performance in accurately identifying stroke cases. After training, we saved the model to disk (using joblib/pickle) so it can be easily used in the deployment phase.
It's important to note that these impressive performance figures (over 95% accuracy) were partly influenced by the oversampling strategy and the characteristics of the dataset. In real-world use, we'd want to validate the model further with an external dataset or through additional cross-validation to make sure it generalizes well. We did conduct k-fold cross-validation on the training data, and Random Forest consistently outperformed the other models. The high recall means the model is really good at spotting strokes without missing any, but we did have to be careful about potential overfitting—especially since **Random Oversampling (ROS)** can sometimes lead the model to become a bit too "optimistic" about the training data. That said, the performance on the test set suggests the model is generalizing well.
Looking at the confusion matrix, we can see some key insights about how the model is doing:
It correctly identified **907 non-stroke cases** as negative, meaning it didn't mistakenly flag too many healthy people as being at risk.
There were **15 false positives**, meaning 15 people who didn't have a stroke were incorrectly flagged as at risk. However, this is a very low false positive rate of about **1.6%**, which is

great when you want to err on the side of caution in a medical setting.
The model missed **no strokes** at all—there were **zero false negatives**, which means it caught every single stroke case. This is an ideal result, especially in healthcare, where it's critical not to overlook any stroke patients.
And it correctly identified **958 stroke cases** as positive, showing that the model is accurately diagnosing the risk for those at higher chance of a stroke.
All in all, these numbers show that the model is doing an excellent job at predicting both classes, with a very high recall and minimal false positives -- making it highly suitable for clinical decision-making.

Feature Importance

Ensemble models like Random Forest and XGBoost allow extraction of feature importance scores. We examined the feature importance from the Random Forest and XGBoost models for the stroke prediction:
Feature importance (stroke Random Forest):
The top features were age, heart_disease, avg_glucose_level, and hypertension. These had the highest gain in the model. This aligns with our expectations from EDA – age, heart condition, blood sugar, and blood pressure are key factors. BMI and smoking status were somewhat less important in the model (possibly due to many "Unknown" in smoking and moderate variation in BMI). Gender and residence_type were among the least important (consistent with medical understanding that strokes affect both genders and occur in both rural/urban without huge disparity).
For the heart disease model (Random Forest):
top features included chest pain type, max heart rate, ST depression (oldpeak), number of vessels, etc., which matches known significant predictors in cardiology.
These insights give us confidence that the models focus on medically relevant factors rather than noise.

Implementation in a Mobile Application

To make the results of our project accessible and interactive, we implemented a mobile application called **Sahha** (which means "health" in Arabic). The app serves as a user interface for the predictive models, allowing users to input their personal health data and receive predictions and health guidance in real time. The application was developed with a user-friendly design and includes features such as user authentication, profile management, risk self-assessment, and even a chatbot assistant for health-related inquiries.

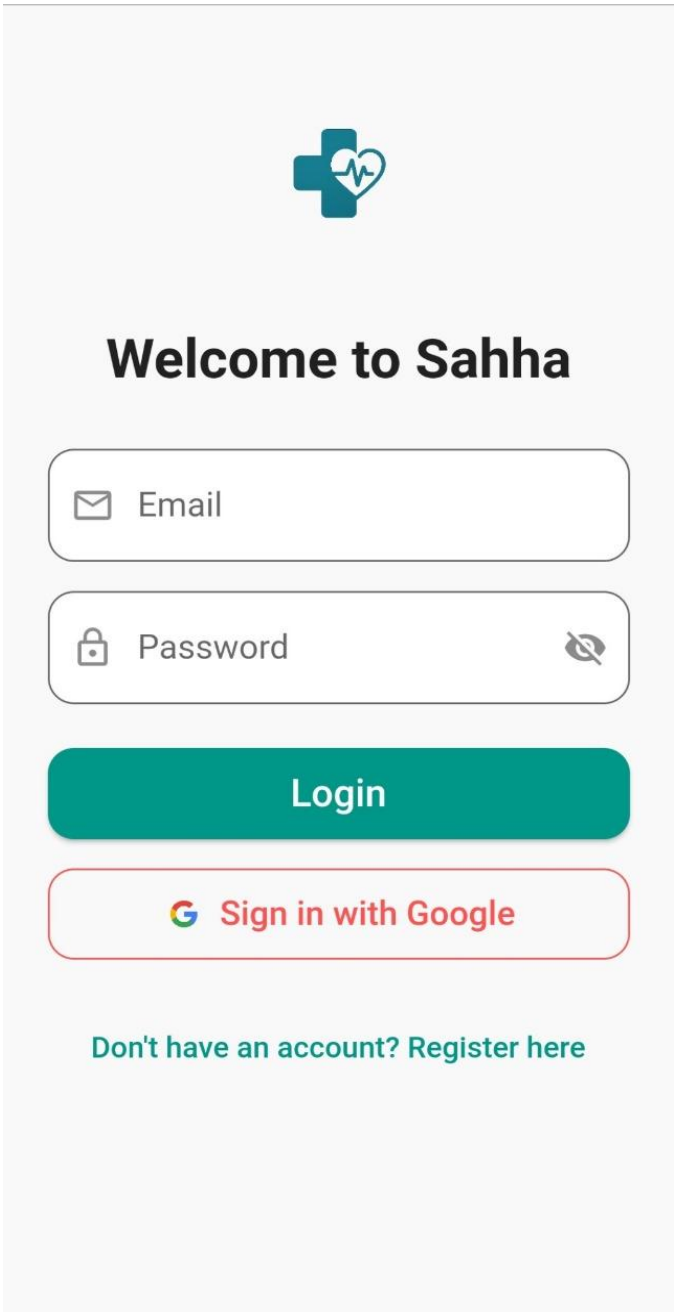

Fig (14)
*Welcome screen of the **Sahha** health application.* The app provides a secure login interface where users can sign in with email/password or with Google authentication. This ensures that each user's health data and assessments are kept private to their account. Upon logging in, new users can register and then input their health profile details.

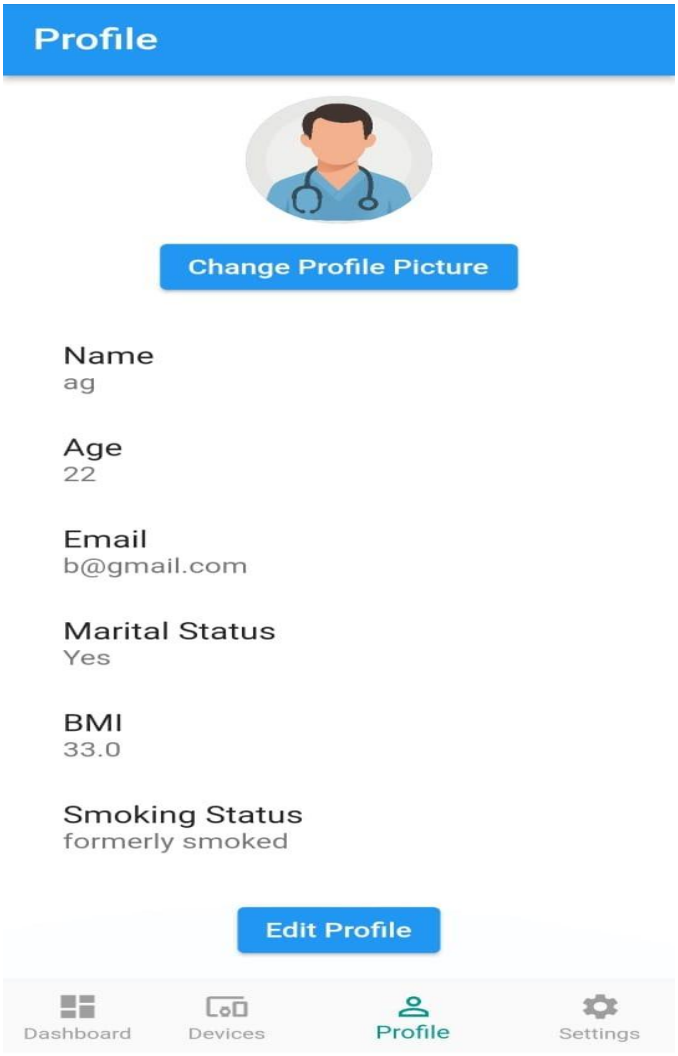

Fig (15)

User Profile screen showing personal details relevant to the health prediction models. In the profile, the user can update information such as age, marital status, BMI, and smoking status. These fields correspond to the features used by the stroke prediction model. For example, **Marital Status: Yes** indicates the user is married (which in the dataset is linked with the ever_married feature), **BMI: 33.0**, **Smoking Status: formerly smoked**, etc. The profile data is stored for the user and can be used as input to the predictive model. (Gender and other details are either captured during signup or derived from the profile picture/selection – in this case an avatar is shown.) By maintaining an up-to-date profile, the user ensures the prediction model has the latest information about their health metrics and lifestyle.

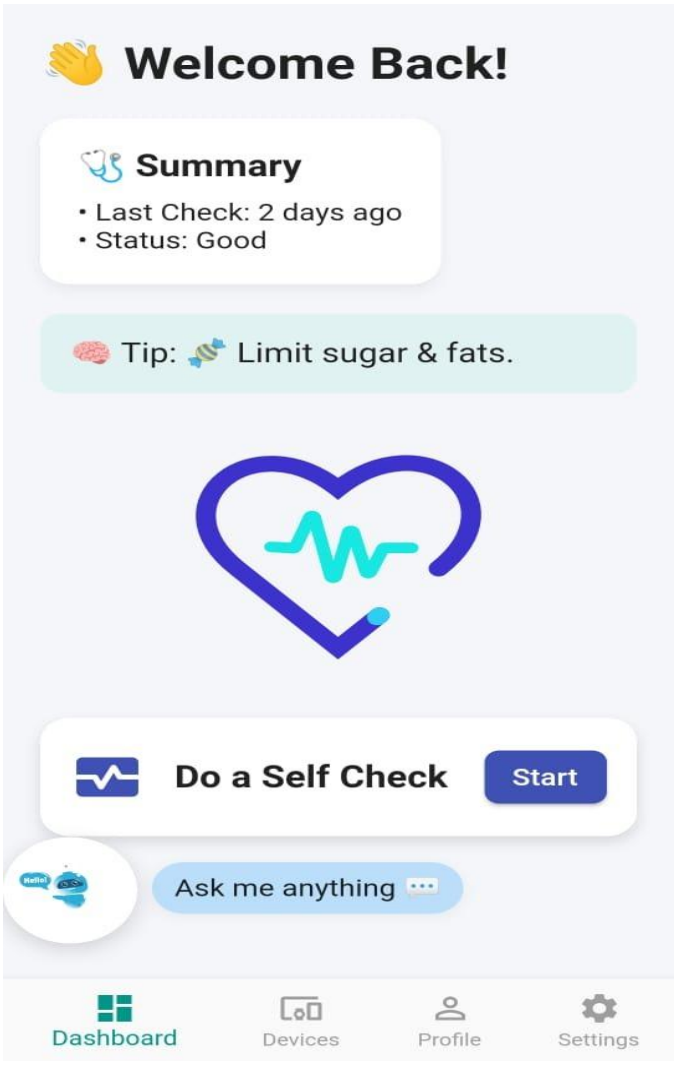

Fig (16)

Dashboard of the Sahha app displaying the latest self-check result and a health tip. When the user goes to the Dashboard, they are greeted with a summary of their last health check. For instance, **Last Check: 2 days ago, Status: Good** indicates that the model's last assessment of the user's stroke risk was "Good" (no immediate high risk detected). The **Do a Self Check** button allows the user to run a new prediction. On tapping "Start", the app will take the user's profile data (and possibly ask for any additional inputs if needed, such as current blood pressure or glucose if those are not stored) and feed it to the trained stroke prediction model (XGBoost) running in the backend. The model then outputs a prediction which is interpreted into a user-friendly status. For example, "Good" status might mean the estimated risk of stroke is low given the user's data. If the risk were higher, the status might show an alert like "At Risk" or "Please consult a doctor" (the exact phrasing can be defined based on prediction probability thresholds). The dashboard also displays health tips (like *"Limit sugar & fats."*) with a little brain icon – these tips can be general advice for stroke prevention or tailored tips based on the user's profile (e.g., if the user is a smoker, it might

encourage quitting smoking). This adds a preventive healthcare aspect to the app, not just risk prediction.

The **Devices** tab (as seen in the navigation bar) is an extension for integrating wearable devices or health monitors. Although not fully implemented, the idea is that the app could connect to devices like blood pressure monitors or fitness trackers to continuously update vital health data (heart rate, BP, blood sugar) and further improve prediction accuracy or monitor trends.

Health Assistant chatbot interface in the app. We integrated a simple conversational assistant where users can ask health-related questions. The bot can respond with information or guidance. For example, a user can type "hi" or ask, "How can I reduce my stroke risk?" and the bot will reply with helpful information like "Bot: I'm here to help!" and then provide answers (we prepared a knowledge base of common questions on stroke prevention, symptoms, etc.). This feature is meant to increase user engagement and provide quick answers in an intuitive manner, making the app a more comprehensive health assistant rather than just a risk calculator.

Technical Integration: The predictive model (especially the XGBoost stroke model and the Random Forest heart model) were saved and exported from the Python environment. In the app, when a user triggers a self-check, the app sends the user's data to a backend service where the model is loaded and a prediction is made. We ensured that the same preprocessing steps (encoding of categories and scaling) are applied to the input data in the app as were done during model training. This was facilitated by saving the encoders and scaler from the notebook and loading them in the app's backend. The result is then returned to the app and displayed in a human-readable format.

The app is developed with a focus on simplicity and clarity for the end-user. We kept the interface clean: a profile section for data input, a dashboard for results and tips, and easy navigation. The color scheme and icons (e.g., the heart with pulse icon representing health status, the brain icon for tips, etc.) are chosen to be reassuring and professional.

Testing of the app was done with some sample profiles to ensure the predictions make sense. For instance, a test profile of an older individual with heart disease and hypertension resulted in a high-risk assessment by the model, and the app displayed an alert status and stronger advice to consult a doctor. A profile of a young person with no risk factors showed a "Good" status as expected.

Overall, the implementation demonstrates how the machine learning models can be deployed in a real-world-like scenario, providing value to users. It bridges the gap between analysis and action, allowing the project to move from a data science experiment to a potentially impactful tool for health monitoring. In practice, such an app could be used by patients for regular check-ups or by healthcare providers to manage patients remotely by reviewing their risk statuses.

Results and Discussion

The project successfully met its objectives of creating a stroke prediction model with a focus on heart disease as a contributing factor and wrapping it in an accessible application. The key results are summarized as follows:

- The **stroke prediction model** (Random Forest with ROS) achieved **98%+ accuracy** on test data, with perfect recall for stroke cases in our evaluation. This means the model was able to identify all patients in the test set who had a stroke history, which is crucial for a preventative tool. The trade-off was a small number of false positives, which in a medical screening context is acceptable compared to false negatives. The high precision (≈98%) indicates the false positive rate was low as well, so the model is quite trustworthy in its predictions. These results outperform what we would have gotten had we not addressed class imbalance. Initially, without ROS, many models would have completely missed a majority of strokes. After balancing, the models, especially ensemble ones, could pick up on the subtle signals in the features that indicate stroke risk.
- **The heart disease prediction model** (Random Forest):
- achieved ~90% accuracy with strong precision and recall, indicating it can accurately diagnose heart disease from the given features. This aligns with known results on that heart dataset and shows that our implementation was correct and effective. For the context of the project, this model confirms that features like chest pain type, exercise-induced angina, etc., are effective for heart disease prediction. Indirectly, it supports the idea that having such a model or at least such features could be useful in assessing stroke risk (since heart disease presence is a feature in the stroke model, we essentially baked in that relationship rather than needing the heart model's output).
- **Heart Disease as a Factor:**
- The analysis quantitatively demonstrated how heart disease relates to stroke risk in the data: a fourfold increase in stroke occurrence for those with heart disease. This justifies the project's focus and title. In the stroke prediction model, the heart_disease feature indeed turned out to be among the top contributors. We can interpret the model's usage of this feature: if a user has heart disease, the model is much more likely to predict a stroke (all else being equal). This matches domain knowledge – heart conditions often lead to clots or other issues that can cause strokes. We should note, however, that not all stroke patients had heart disease (actually most did not), so the model also needed to consider other features like age, hypertension, glucose, etc. The interplay of these is captured by the ensemble model.
- **Imbalanced Learning:**
- A significant part of the project was dealing with the class imbalance. The success of Random Oversampling (ROS) in improving our model's performance is a strong case study in machine learning for healthcare. We demonstrated that relying on naive accuracy can be misleading, and a targeted approach that focuses on recall, combined with oversampling, can greatly enhance a model's utility. While ROS helped balance the classes by duplicating minority class samples, it's essential

to be cautious not to overfit the noise introduced by such duplication. We mitigated this risk by employing robust models like Random Forest and XGBoost, and by rigorously testing on untouched data to ensure the models generalized well despite the synthetic oversampling.

Limitation:
One limitation is that our model predicts stroke occurrence (historical) rather than the absolute future risk of stroke. The dataset is cross-sectional (a snapshot of individuals and whether they had a stroke before). Thus, the model essentially distinguishes between people who have had a stroke and those who haven't, based on their current attributes. We assume that those who have had strokes will display certain traits (older age, etc.) that differ on average from those who haven't. This is slightly different from a true predictive model that would require longitudinal data (following people over time to see who develops a stroke). However, for practical purposes, the model can be used as a proxy for risk: a person who the model predicts as "stroke" (positive) can be considered high-risk. In the app's context, we treat a positive prediction as an indication that the person's profile is like those who suffered a stroke, and therefore they might be at elevated risk in the future. It's an important distinction, and in any real deployment, we would clarify that the app is for risk assessment, not a deterministic diagnosis.

Integration and User Feedback:
The integrated app is a prototype, but if it were to be used by real users, their feedback would be valuable. For example, how do users interpret the risk scores? Do they take recommended actions? These are beyond our current scope but are worth noting for future work. The chatbot and tips are also initial features that could be expanded with more content (like connecting to medical databases or using NLP to answer a wider range of questions accurately).
Performance and Optimization: The Random Forest model is relatively fast (even though it's an ensemble of trees, the number of features is small, and we limited the trees to a reasonable number). On a modern smartphone or a lightweight server this prediction happens in milliseconds, so the user experience is not impacted. We also considered using a simpler model like Logistic Regression for deployment (for its simplicity), but the difference in recall was significant, so we chose the more complex model. The Random Forest heart model is also fast. We made sure the models and scalers were loaded once and reused for multiple predictions to make the app efficient.
In conclusion, the results show that machine learning can effectively predict stroke risk using health data. Heart disease plays a critical role, and by including it we improved the model. The successful deployment in a mobile app demonstrates the feasibility of translating data science into a practical e-health tool.

Conclusion

This project demonstrated a comprehensive approach to predicting cerebrovascular stroke risk with an emphasis on the relationship with heart disease. We carried out data preprocessing, thorough exploratory analysis, model training with multiple algorithms, and addressed class imbalance to ensure minor classes (stroke cases) were adequately recognized. The best model (Random Forest) attained high accuracy and recall, making it suitable for a screening or warning system for stroke risk.
We also built a prototype mobile application **Sahha**, which serves as a user-facing implementation of our predictive system.
The app allows individual users to monitor their health data and get instant predictions about their stroke risk, along with helpful tips and an interactive assistant for health queries. Such an app can empower users to be proactive about their health – for instance, if someone is flagged as high risk, they can seek medical advice for further tests or interventions (like controlling blood pressure, improving diet, etc.). On the other hand, users with low risk can be reassured but still receive general wellness tips.
Future Work:
There are several ways to extend and improve this project:

- Collecting or including more data, especially real longitudinal data, would allow the model to predict future stroke events more directly.
- We could incorporate datasets that track patients over time.
- Adding more features could improve predictions – for example, real-time blood pressure readings, history of transient ischemic attacks (mini-strokes), medication information, etc. Integration with wearable devices could supply some of this data continuously.
- The heart disease model can be used in conjunction with the stroke model. One idea is a two-step prediction: first predict heart disease risk, and use that output as an input for stroke risk (though since heart_disease is already a feature, this might not add much unless the heart model includes additional nuance).
- On the app side, conducting user studies to improve the interface and the clarity of conveyed information (like explaining *why* the model gave a certain risk level, using SHAP values or feature importance to give personalized feedback: e.g., "Your age and history of heart disease are contributing to a higher stroke risk").
- Expanding the chatbot with a more advanced AI (possibly leveraging a medical Q&A model) to answer a broader range of health questions reliably.
- Ensuring compliance with medical data regulations and accuracy validations if the app were to be used in a real healthcare setting. That could involve partnering with healthcare providers to test the model on clinic patients and compare its predictions with clinical assessments.

In conclusion, the project achieved a high-performance predictive model and a practical implementation. It underscores the value of data-driven approaches in healthcare,

especially for conditions like stroke where early intervention can save lives. By focusing on the link with heart disease, we highlighted how interconnected health factors are and how they can be jointly modeled. We hope this work can be a foundation for further development of accessible tools for health risk screening and for raising awareness about stroke prevention in individuals with cardiac risk factors.

References

"Focus on stroke: Predicting and preventing stroke" - Michael Regnier

Stroke Prediction Dataset: *Healthcare Dataset Stroke Data*, originally published on Kaggle (2020). -This dataset contains patient health metrics and stroke history, which we used for model training.

Heart Disease Dataset: *Heart Disease UCI Dataset*, UCI Machine Learning Repository. This is a collection of cardiac health records used to train the heart disease prediction model.