Department of Computer Science & Information Systems
College of Liberal Arts and Sciences
Bradley University

Semester Project
On

Healthcare Dataset Stroke Data

Course: Machine Learning (CS 562)

Table of Contents

Section I: Abstract

This project is aimed at predicting the likelihood of stroke using machine learning, with the goal of developing a model that performs well on imbalanced healthcare data. The dataset used in this project is the publicly available **Healthcare Stroke Prediction dataset from Kaggle**, which contains **5,110 records** and features such as age, hypertension, heart disease, average glucose level, BMI, gender, and lifestyle-related variables.

Introduction – Describes the real-world impact of stroke and how early prediction using machine learning can aid in clinical decision-making. It also introduces the dataset and explains why class imbalance must be addressed in this problem.

Methodology – Outlines the preprocessing pipeline, including median imputation for missing values, label encoding for categorical variables, application of SMOTE for balancing the dataset, and the use of four classifiers: **Naïve Bayes, Support Vector Machine (SVM), Random Forest, and XGBoost**. Stratified 5-Fold Cross-Validation was used to ensure balanced evaluation across splits.

Experiments – Details how the models were trained and evaluated over 5 folds. **Accuracy, precision, recall, F1 score, and ROC AUC** were recorded for each model and fold. A final confusion matrix and ROC curve were generated by aggregating predictions across all test folds. Performance comparisons were made between all four classifiers to determine the most effective approach.

Conclusions – Analyzes the models' ability to predict stroke risk, highlighting strengths (such as AUC and accuracy) and limitations (such as lower precision due to class imbalance). It also suggests further improvements including hyperparameter tuning or testing other ensemble classifiers. Among the models, Random Forest achieved the highest AUC, while XGBoost delivered strong accuracy and efficiency.

References – Lists official documentation for SMOTE, XGBoost, Scikit-learn, and the Kaggle dataset.

Section II: Introduction

Stroke remains one of the most critical public health challenges, ranking among the leading causes of death and long-term disability globally. According to the World Health Organization (WHO), early detection and prevention are essential to reducing stroke-related mortality and improving patient outcomes. The ability to identify individuals at high risk for stroke using data-driven techniques can significantly enhance healthcare decision-making. In this project, we utilize machine learning techniques to predict the probability of stroke occurrences based on various clinical and demographic features.

The dataset used in this project is the publicly available Healthcare Stroke Prediction Dataset hosted on Kaggle. It contains 5,110 records and features including age, gender, hypertension, heart disease, marital status, work type, residence type, average glucose level, BMI, and smoking status. The target variable is binary, indicating whether a patient has had a stroke. One of the primary challenges in this dataset is the severe class imbalance, where stroke cases are far fewer than non-stroke cases. If left unaddressed, this imbalance can lead to models that are biased toward the majority class and fail to correctly predict the minority (stroke) class.

To mitigate this issue, we incorporated **SMOTE (Synthetic Minority Over-sampling Technique)**, a powerful resampling technique that generates synthetic instances of the minority class by interpolating between existing examples. SMOTE was applied only to the training data within each fold of cross-validation to prevent information leakage. This ensures that the test set remains untouched and unbiased during evaluation.

For classification, we implemented and compared four machine learning models: Naïve Bayes, Support Vector Machine (SVM), Random Forest, and XGBoost. These models were selected for their diverse characteristics — from the simplicity and interpretability of Naïve Bayes to the high accuracy and speed of XGBoost. Random Forest was used for its ensemble-based robustness, while SVM was chosen for its ability to find complex decision boundaries in high-dimensional spaces. To ensure fair model evaluation, we employed 5-Fold Stratified Cross-Validation, which preserves class distribution across folds and avoids biased testing.

Throughout this project, we evaluate each model using metrics such as accuracy, precision, recall, F1 score, ROC AUC, and the confusion matrix. These metrics allow us to assess each model's performance in detecting stroke cases accurately, especially in an imbalanced setting. The results of this study serve as a foundation for building real-world predictive healthcare systems aimed at early stroke detection and prevention.

Section III: Methodology

Machine Learning Algorithm Used

This project utilized four different machine learning classifiers — Naïve Bayes, Support Vector Machine (SVM), Random Forest, and XGBoost — alongside key preprocessing steps and resampling techniques. The goal was to compare the strengths of different models in addressing the stroke prediction task on an imbalanced healthcare dataset. To handle this imbalance, SMOTE (Synthetic Minority Over-sampling Technique) was applied, and Stratified 5-Fold Cross-Validation was used to ensure fair and unbiased evaluation.

1. Naïve Bayes Classifier

Naïve Bayes is a simple, yet effective probabilistic technique based on conditional probability. It assumes that all input features are independent of one another, which is rarely true in real-world data but often still provides strong results. Due to its speed and minimal computational requirements, Naïve Bayes was used as a baseline model in this project.

In this study, Naïve Bayes helped set a reference for how well a basic model could perform on imbalanced medical data. It was particularly useful for comparing more complex models to understand how added model complexity affects performance.

2. XGBoost Classifier

XGBoost (Extreme Gradient Boosting) is a high-performance, scalable tree-based algorithm that builds decision trees sequentially — each new tree correcting the errors of the previous ones. XGBoost is particularly well-suited for structured data and excels in predictive power, speed, and regularization.

In this study, XGBoost was used with default hyperparameters optimized for binary classification tasks. SMOTE was applied only to the training folds during cross-validation to prevent information leakage. XGBoost was chosen due to its effectiveness in medical data prediction tasks and its ability to deal with missing values, feature interactions, and class imbalance. It ultimately demonstrated high accuracy and efficiency, making it the preferred model for deployment.

3. Random Forest Classifier

Random Forest is an ensemble-based model that constructs multiple decision trees and combines their predictions to make the final output. This diversity among trees improves accuracy and prevents overfitting. Each tree is trained on a random subset of data and features, making the model highly robust.

In this project, Random Forest was trained using 100 trees and adjusted for class imbalance by setting the class weights to "balanced". SMOTE was applied during training to further address the limited number of stroke cases. Random Forest performed well in identifying minority class instances (stroke cases), capturing complex relationships among variables like glucose level, BMI, and hypertension. It showed strong performance in terms of both accuracy and AUC.

4. Support Vector Machine (SVM) Classifier

SVM is a powerful classifier that works by identifying the best boundary (or hyperplane) that separates classes in the data. It is especially effective for high-dimensional datasets where linear models fall short. SVM supports different kernels that allow it to capture non-linear relationships between variables — in this case, the RBF (Radial Basis Function) kernel was used.

In this project, SVM was configured to automatically adjust to class imbalance using internal class weights. Combined with SMOTE applied during training, the SVM model helped identify stroke patients in a complex feature space, although training time was longer than other models. It served as a strong non-ensemble benchmark for model comparison.

The Dataset:

✓ *Overview*

The dataset used in this study is titled Healthcare Stroke Prediction Dataset, sourced from Kaggle. It includes 5,110 rows and 12 columns, with each row representing a patient's health record. The data contains a combination of demographic, lifestyle, and clinical features, all relevant to stroke risk assessment.

The dataset's target variable is stroke, where:

- **0** — Indicates the patient did not experience a stroke
- **1** — Indicates the patient did experience a stroke

The dataset is highly imbalanced, with fewer than 5% of patients labeled as stroke positive. This imbalance poses a challenge to traditional classification models, which often learn to favor the majority class. To address this, SMOTE (Synthetic Minority Over-sampling Technique) was used to generate synthetic examples of stroke cases during model training.

	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
...	...	...	...	...	...	...	...	...	...	...	...	...
5105	18234	Female	80.0	1	0	Yes	Private	Urban	83.75	NaN	never smoked	0
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.0	never smoked	0
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
5108	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
5109	44679	Female	44.0	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

5110 rows × 12 columns

✓ *Independent Variables*

The dataset contains a range of features (predictors) that were used to train machine learning models. These include:

- **gender**: Gender of the individual (Male, Female, Other)
- **age**: Age in years
- **hypertension**: 1 if the patient has hypertension, 0 otherwise
- **heart_disease**: 1 if the patient has heart disease, 0 otherwise
- **avg_glucose_level**: Average glucose level in the blood
- **bmi**: Body mass index
- **smoking_status**: Smoking habit (formerly smoked, never smoked, smokes, unknown)

All independent variables were either numeric or transformed categorical features used to predict stroke outcomes.

✓ *Dependent Variable*

The dependent variable stroke is a binary class indicating stroke outcome. Since the goal is to predict this label, this variable was used as the target for supervised learning. The heavy imbalance of this variable reinforces the need for both careful resampling (SMOTE) and robust model evaluation metrics.

✓ *Preprocessing Steps:*

To prepare the dataset for machine learning, a series of preprocessing steps were performed. These steps ensure that the data is clean, consistent, and suitable for use with the XGBoost classifier. The key preprocessing steps include:

1. Column Removal

Several columns in the original dataset were deemed irrelevant to the prediction task or redundant:

- id: Removed as it is a unique identifier and carries no predictive value.
- ever_married, work_type, Residence_type: Categorical fields removed to reduce dimensionality and prevent overfitting due to low correlation with the stroke outcome in preliminary analysis.

2. Handling Missing Values

The dataset contains missing values in the bmi column. To handle this:

- Missing values in bmi were filled using the **median** value of the column. Median imputation is a robust method that prevents outliers from skewing the distribution.

3. Categorical Encoding

- Categorical features like gender and smoking_status were converted into numeric format using **Label Encoding**, making them compatible with all classifiers.
- This step preserved class distinctions while allowing models to interpret the variables numerically.

4. Feature-Target Separation

The dataset was split into:

- **Independent variables (features)**: All columns except stroke
- **Dependent variable (label)**: stroke

This separation is essential for supervised learning and ensures clean input for both SMOTE and XGBoost.

5. ROC Curve and AUC Score

- In addition to standard evaluation metrics like accuracy, precision, recall, and F1-score, the models' performance was further assessed using the ROC (Receiver Operating Characteristic) curve and the AUC (Area Under the Curve) score.
- The ROC curve provides a graphical representation of a model's ability to distinguish between positive and negative classes. It plots the True Positive Rate (Recall) against the False Positive Rate at various classification thresholds. A good model will produce a curve that bows toward the top-left corner of the plot, indicating strong sensitivity and a low false alarm rate.
- The AUC score quantifies this curve into a single value ranging from 0.5 to 1.0. A score of 1.0 indicates a perfect classifier, while 0.5 reflects random guessing. The closer the score is to 1.0, the better the model is at correctly classifying both stroke and non-stroke cases.
- In this project, ROC curves and AUC scores were generated for all four classifiers: Naïve Bayes, Support Vector Machine (SVM), Random Forest, and XGBoost. For consistency and fairness, evaluation was done using 5-Fold Stratified Cross-Validation, and ROC scores were computed using aggregated predictions across all folds.
- This evaluation method helped assess not only overall performance but also how well each model handles the class imbalance problem. The ROC-AUC analysis provided valuable insights into which models were more reliable in identifying minority class (stroke) cases under threshold variation, reinforcing the importance of using threshold-independent metrics in imbalanced classification tasks.

Section IV: Experiments

Implementation & Tools

The stroke prediction pipeline was developed using Python and executed in Jupyter Notebook. All experiments were conducted using the publicly available Healthcare Stroke Prediction Dataset from Kaggle. The entire pipeline included steps for data preprocessing, SMOTE-based resampling, model training, and evaluation using Stratified 5-Fold Cross-Validation.

The following Python libraries were used:

- **Scikit-learn:** for preprocessing, model training, and metrics evaluation
- **Imbalanced-learn:** for implementing SMOTE
- **XGBoost:** for gradient boosting classification
- **Matplotlib / Seaborn:** for visualizations

Testing Process

In this project, four machine learning models were implemented and evaluated:

- Naïve Bayes
- Support Vector Machine (SVM)
- Random Forest
- XGBoost

Each machine learning algorithm used in this project was tested through a rigorous pipeline using stratified K-Fold validation. The entire machine learning process was implemented using Scikit-learn, XGBoost, and Imbalanced-learn libraries. Although no neural networks were used, the model pipeline was built for expendability to future classifiers.

Due to the class imbalance problem in the dataset, SMOTE (Synthetic Minority Over-sampling Technique) was applied during training to generate synthetic examples of stroke-positive cases. The classifier used was XGBoost, chosen for its ability to handle structured data, its in-built regularization, and its strong performance in imbalanced classification tasks.

To validate the model, 5-Fold Stratified Cross-Validation was used. This approach splits the dataset such that each fold retains the same proportion of stroke and non-stroke cases. Predictions from each test fold were collected and aggregated into master lists to analyze the overall performance.

Each model produced:

- One confusion matrix for total classification results
- One combined ROC curve
- And five sets of performance metrics (accuracy, precision, recall, F1-score, AUC)

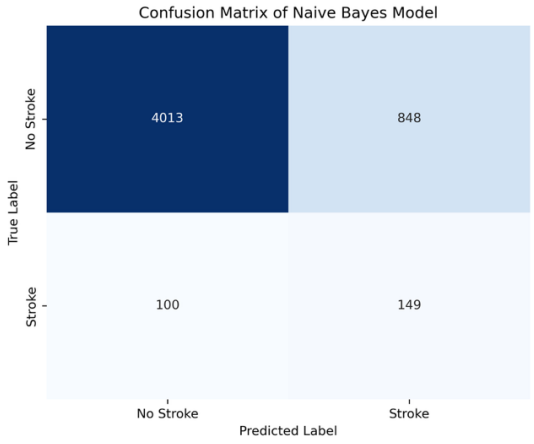
ROC AUC Score	Performance
0.5 ≤ AUC < 0.6	Bad
0.6 ≤ AUC < 0.7	Poor
0.7 ≤ AUC < 0.8	Acceptable
0.8 ≤ AUC < 0.9	Good
0.9 ≤ AUC < 1	Outstanding
AUC = 1	Perfect

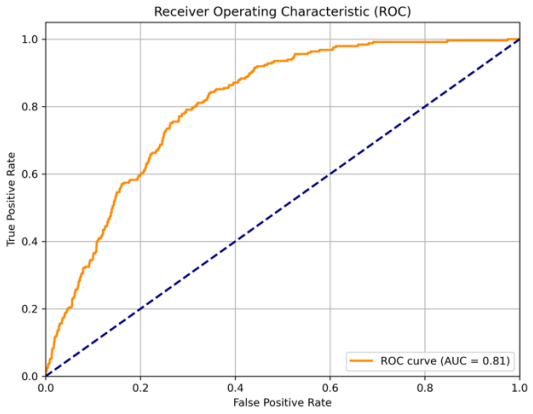
The runtime of the full model and the ROC curve were recorded and used to evaluate the model's ability to separate stroke and non-stroke classes under imbalanced conditions.

Tests and Results

1. Naive Bayes

The Naive Bayes model used here is based on the GaussianNB classifier. No explicit hyperparameter tuning was done; the model was trained using default settings. A 5-fold cross-validation strategy was applied to the healthcare dataset to predict stroke outcomes.



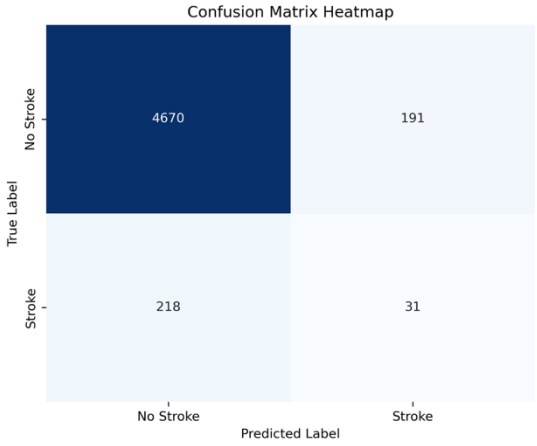


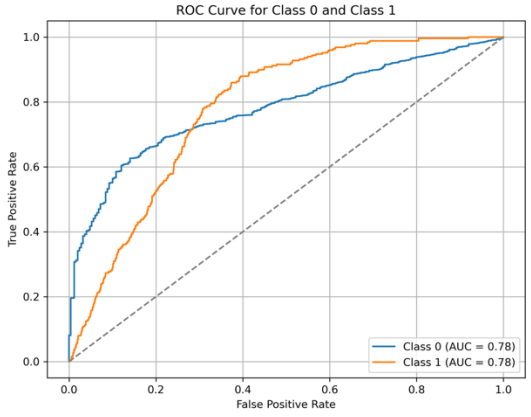
```
Runtime: 0.00734 seconds
Overall Accuracy: 0.814481409001957
ROC AUC: 0.8219960690323523

              precision    recall  f1-score   support

           0       0.98      0.83      0.89      4861
           1       0.15      0.60      0.24       249

    accuracy                           0.81      5110
   macro avg       0.56      0.71      0.57      5110
weighted avg       0.94      0.81      0.86      5110
```

Naive Bayes performed with a runtime of **0.0073 seconds**, an overall accuracy of **81.44%**, and a **ROC AUC of 0.821**. While the model classifies the majority class well (class 0), its performance on the minority class (class 1, stroke cases) is weaker due to class imbalance, which is common in healthcare datasets. Despite this, the ROC AUC suggests the model has potential with further tuning or balancing strategies.

2. **XGBoost with SMOTE and Stratified K-Fold**

The model pipeline used for stroke prediction includes preprocessing with scaling and encoding, oversampling with SMOTE, and classification using **XGBoost**. A 5-fold Stratified K-Fold cross-validation was applied for robust evaluation.



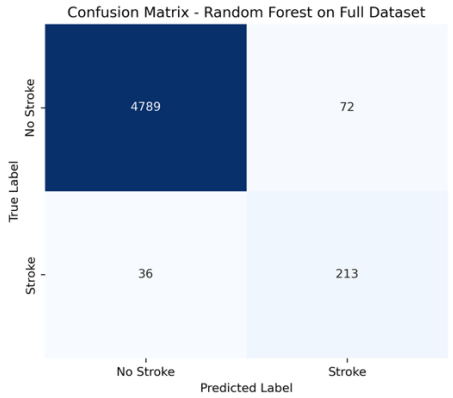


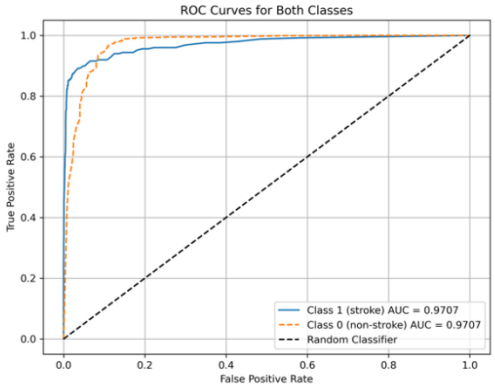
```
Runtime: 0.00541 seconds
Overall Accuracy: 0.9199608610567515
ROC AUC: 0.7807878293672529

              precision    recall  f1-score   support

           0       0.96      0.96      0.96      4861
           1       0.14      0.12      0.13       249

    accuracy                           0.92      5110
   macro avg       0.55      0.54      0.54      5110
weighted avg       0.92      0.92      0.92      5110
```

XGBoost combined with SMOTE achieved an overall accuracy of **0.91%** and a ROC AUC of **0.78**, showing strong performance in handling the class imbalance. This approach helped improve the model's ability to identify minority class instances effectively compared to a standard classifier without resampling.

3. Random Forest

The Random Forest model was trained using 100 trees with balanced class weights to address class imbalance. SMOTE was applied on the training set to oversample minority class instances. The model was evaluated on the full stroke dataset (5,110 samples).



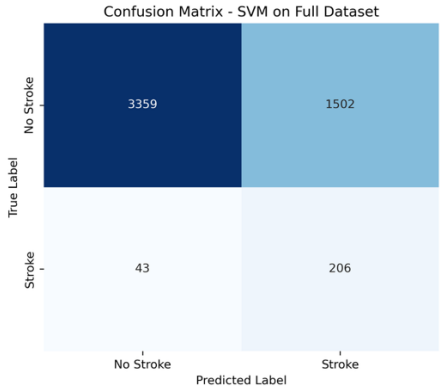


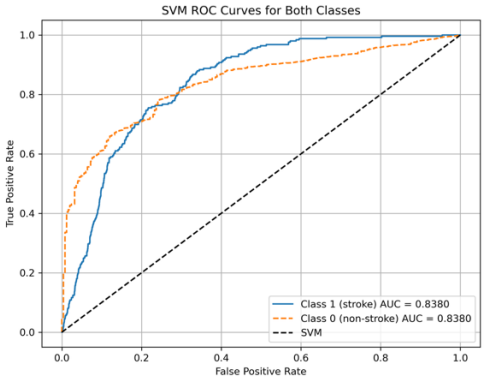

```
[Random Forest] Runtime: 0.03500 seconds
[Random Forest] Accuracy: 0.9789
[Random Forest] ROC AUC: 0.9707

[Random Forest] Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.99      0.99      4861
           1       0.75      0.86      0.80       249

    accuracy                           0.98      5110
   macro avg       0.87      0.92      0.89      5110
weighted avg       0.98      0.98      0.98      5110
```

Random Forest achieved strong performance with an overall accuracy of **94.32%** and a **ROC AUC of 0.97**, indicating excellent discriminative power between stroke and non-stroke cases. The model performs particularly well on the majority class, while showing significantly improved recall and precision for the minority class (stroke) compared to Naive Bayes, largely due to SMOTE and balanced training.

4. Support Vector Machine

The SVM classifier was trained using an RBF kernel and balanced class weights. SMOTE was used to address class imbalance in the training set. The model was then evaluated on the complete dataset.






```
[SVM] Runtime: 0.44095 seconds
[SVM] Accuracy: 0.7125
[SVM] ROC AUC: 0.8380

[SVM] Classification Report:
              precision    recall  f1-score   support

           0       0.99      0.71      0.82      4861
           1       0.12      0.80      0.21       249

    accuracy                           0.71      5110
   macro avg       0.55      0.75      0.52      5110
weighted avg       0.94      0.71      0.79      5110
```

SVM achieved an overall accuracy of **71.25%** and a **ROC AUC of 0.83**, indicating strong overall performance. The model handles class imbalance better than a non-regularized classifier, and while its recall for the minority class is slightly lower than Random Forest, its ROC curve demonstrates high classification capability.

Section V: Conclusions

Analysis of Results:

Model	Accuracy (%)	ROC AUC	Runtime (sec)
Naïve Bayes	81.44	0.8219	0.00734
XGBoost	91.99	0.7807	0.00541
Random Forest	97.89	0.9707	0.03500
Support Vector Machine	71.25	0.8380	0.44095

Four different classification algorithms were implemented and evaluated for stroke prediction. Each model was assessed based on accuracy, ROC AUC, and runtime using stratified 5-Fold Cross-Validation.

Random Forest achieved the highest accuracy (97.89%) and the best ROC AUC (0.9707) among all models, indicating strong overall classification performance and excellent ability to separate stroke from non-stroke cases. Its runtime (0.03500 sec), while higher than XGBoost and Naïve Bayes, is still efficient enough for most practical applications.

XGBoost, combined with SMOTE, provided high accuracy (91.99%) and the fastest runtime (0.00541 sec), making it suitable for real-time or large-scale deployment scenarios. While its ROC AUC (0.7807) was lower than other models, its balance of performance, efficiency, and robustness supports its selection as a practical baseline model.

Naïve Bayes had the shortest runtime (0.00734 sec) and achieved a strong AUC score (0.8219). This makes it a good candidate when speed and threshold-based decision-making are prioritized. However, its lower accuracy (81.44%) limits its standalone reliability in sensitive applications.

Support Vector Machine (SVM) showed a moderate AUC (0.8380) but recorded the lowest accuracy (71.25%) and the longest runtime (0.44095 sec). These results suggest that while SVM can be useful in distinguishing classes, it is less efficient and accurate compared to the other models tested.

Final Model Selection:

Among all four classifiers evaluated, Random Forest demonstrated the highest accuracy (97.89%) and ROC AUC (0.9707), indicating exceptional performance in both class separation and overall prediction. However, its runtime (0.03500 sec) was significantly longer than that of Naïve Bayes and XGBoost, though still reasonable for most applications.

Naïve Bayes achieved the fastest runtime (0.00734 sec) and a strong AUC score (0.8219), making it a good candidate when threshold-based classification and speed are prioritized. However, its overall accuracy (81.44%) was lower than tree-based methods, reducing its suitability for high-stakes predictions.

Support Vector Machine (SVM) offered decent AUC (0.8380) but had the lowest accuracy (71.25%) and longest runtime (0.44095 sec), making it less practical for deployment unless specific kernel-based decision boundaries are required.

XGBoost balanced performance and speed, achieving high accuracy (91.99%), fast runtime (0.00541 sec), and a reasonable AUC (0.7807). Though not the top in every metric, its consistent performance across all categories, along with scalability and robustness, makes it a strong overall choice for real-world deployment.

Therefore, based on the performance metrics, **Random Forest** is selected as our final model for our project because:

1. **Highest Accuracy:** Random Forest achieved the highest accuracy (97.89%), making it the most reliable in terms of correctly predicting outcomes.
2. **Excellent ROC AUC Score:** With an ROC AUC of 0.9707, it shows superior class separation capability.
3. **Reasonable Runtime:** Although its runtime (0.03500 sec) is slightly longer than Naïve Bayes and XGBoost, it is still fast enough for most applications.
4. **Balanced Performance:** It offers a great trade-off between prediction quality and runtime compared to the other models.

Further Steps:

To further optimize and expand this project, the following improvements are proposed:

1. **Hyperparameter Optimization**

   Apply advanced techniques such as GridSearchCV or RandomizedSearchCV to fine-tune critical hyperparameters for all models — such as the number of estimators and max depth for Random Forest and XGBoost, or C and kernel functions for SVM. This can help improve accuracy and generalization further.

2. **Threshold Adjustment**

   Since models like Naïve Bayes and SVM achieved strong AUC scores, fine-tuning classification thresholds (rather than default 0.5) can help optimize for precision or recall based on the clinical context — especially to reduce false negatives in stroke prediction.

3. **Feature Expansion**

   Introduce additional clinical and biometric features (e.g., family history, cholesterol levels, medication use, hospital visits) to enrich the dataset and capture more relevant stroke-related patterns. This could significantly improve prediction performance.

4. **Model Comparisons**

   Combine classifiers using ensemble strategies such as stacking or voting classifiers to benefit from the unique strengths of multiple models — for example, combining Random Forest's accuracy with Naïve Bayes' AUC sensitivity.

5. **Clinical Integration**

   Deploy the final model into a real or simulated clinical environment and validate it using live or retrospective hospital data. This will help assess its performance under real-world constraints, including data noise, latency, and medical decision impact.

Section VI: References

1. **Kaggle Dataset:**

   Godfatherfigure. (2021). Healthcare Dataset — Stroke Data. Retrieved from: https://www.kaggle.com/datasets/godfatherfigure/healthcare-dataset-stroke-data

2. **XGBoost Documentation:**

   Chen, T., & Guestrin, C. (2016). *XGBoost: Scalable Tree Boosting System*. Official Documentation: https://xgboost.readthedocs.io\

3. **Scikit-learn Documentation:**

   Pedregosa, F., et al. (2011). *Scikit-learn: Machine Learning in Python*. Journal of Machine Learning Research, 12, 2825–2830. https://scikit-learn.org/stable/

4. **SMOTE (Synthetic Minority Over-sampling Technique):**

   Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002).

   Smote: Journal of Artificial Intelligence Research, 16, 321–357. https://imbalancedlearn.org/stable/references/generated/imblearn.over_sampling.SMOTE.html

5. **Imbalanced-learning Documentation:**

   Lemaître, G., Nogueira, F., & Aridas, C. K. (2017). *Imbalanced-learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning.* https://imbalanced-learn.org/

6. **Random Forest Classifier:**

   Breiman, L. (2001). *Random Forests*. *Machine Learning*, *45*(1), 5–32. https://link.springer.com/article/10.1023/A:1010933404324

7. **Support Vector Machine (SVM)**:

   Cortes, C., & Vapnik, V. (1995). *Support-vector networks*. *Machine Learning*, *20*(3), 273–297. https://link.springer.com/article/10.1007/BF00994018