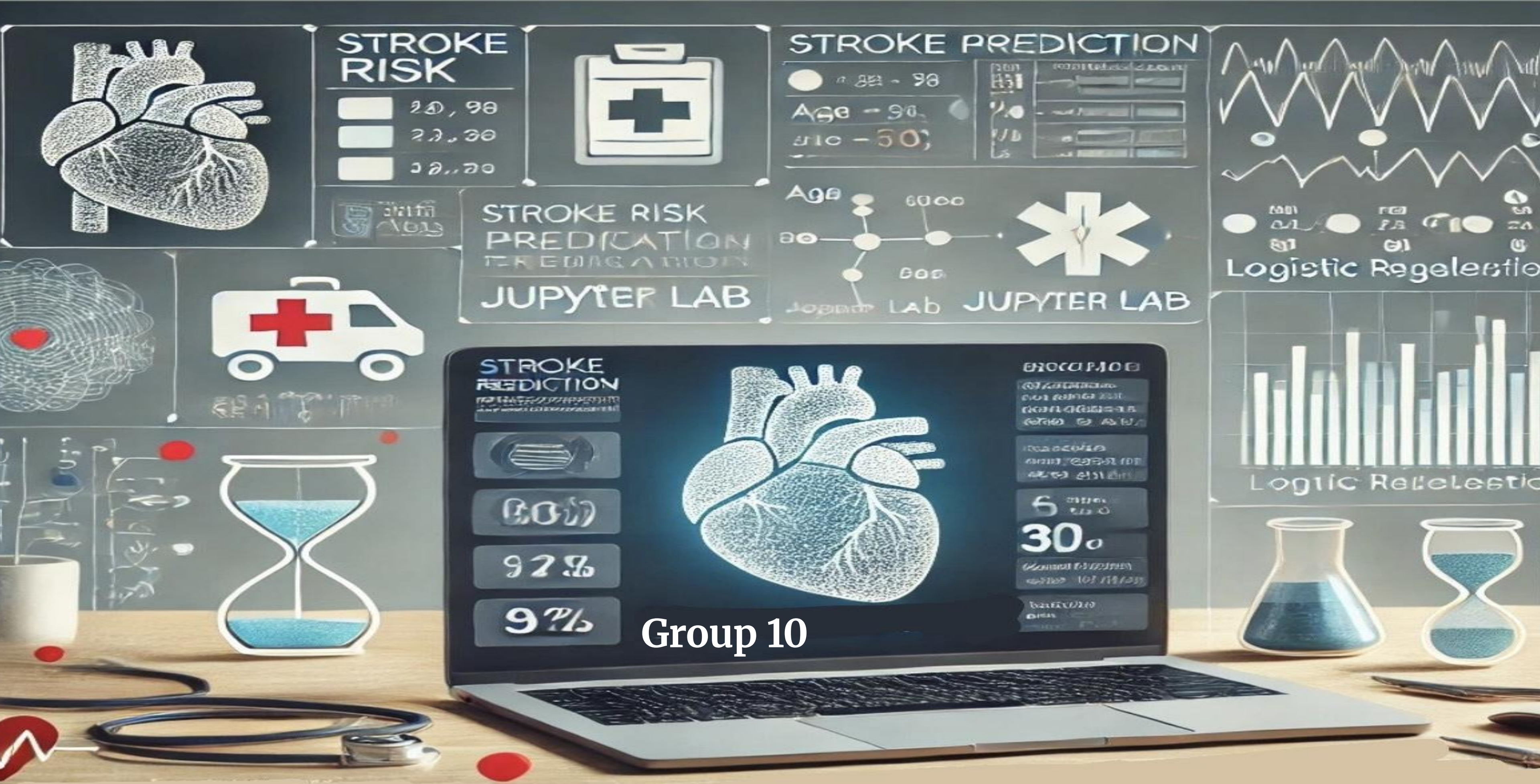


Group 10

Gayatri R., Janet R., Somayyeh K., Chris S., Ekene O.

INTRODUCTION

Overview of Stroke Risk Prediction:

Stroke is a leading cause of disability and death, often occurring suddenly, with severe consequences for individuals and healthcare systems.

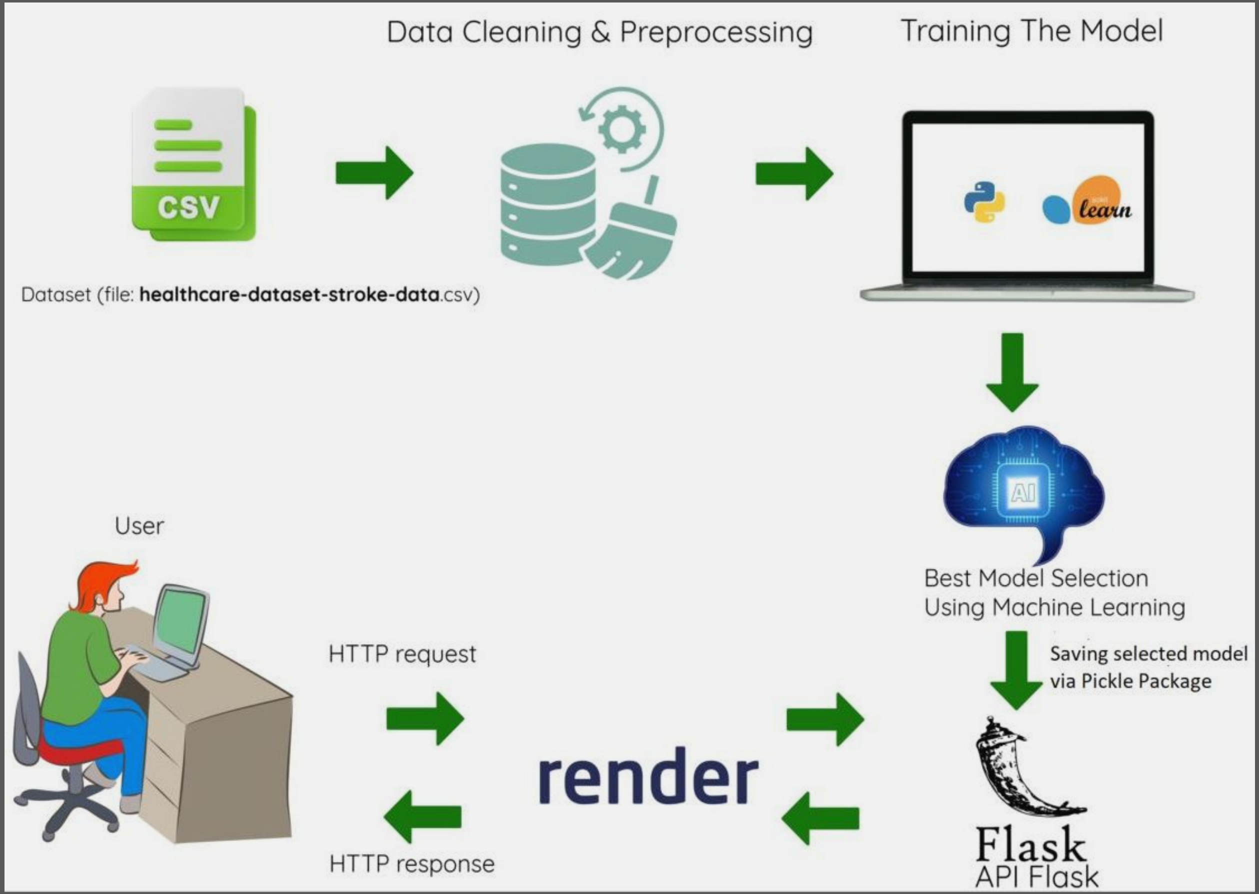
Importance of early prediction using ML:

Early stroke risk prediction using machine learning enables timely interventions, potentially preventing strokes and improving outcomes.

Goal of the Project:

The primary goal of this project is to develop a predictive model that estimates stroke risk based on health and lifestyle data. By identifying high-risk individuals early, this model aims to support healthcare providers in making informed decisions, enabling preventive measures and personalized health recommendations.

OVERVIEW OF THE PROJECT:



DATA CLEANING & PREPROCESSING

Dataset Source:
https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset/data
The dataset consists of 5,110 records with 12 features, focusing on demographic, health, and lifestyle factors that influence stroke risk.

Handling Missing Values:
The null values in the bmi column were removed. The smoking_status column included an "unknown" category, which was retained due to a high number of entries with this value

Encoding categorical variables:
Categorical data in columns such as ever_married, Residence_type, smoking_status, work_type, and gender were encoded using the get_dummies method.

Splitting data into train & test sets:
The dataset was split into training and testing sets using an 80-20 ratio. The train_test_split function was used with a fixed random state to ensure reproducibility. SMOTE applied to handle class imbalance.

Standardization:
Numerical features such as age, BMI, and glucose levels were standardized for consistent scaling across the model.

MODEL TRAINING & SELECTION:

Models tested:

Random Forest
SVM
Logistic Regression
Voting Classifier (Logistic Regression + Random Forest)
Voting Classifier (Logistic Regression + Random Forest + SVM)
XGBoost
XGBoost with Random Forest, Logistic Regression & SVM
Neural Network with MLP Classifier

Performance metrics used:

Accuracy, Confusion Matrix, Classification Report

Comparison of Results:

XGBoost outperformed the other models with an accuracy of 96.8%

Final Model Selection & Reasons

The model achieved 96.81% accuracy, correctly classifying most cases with a small number of misclassifications (24 false positives, 35 false negatives). It maintained a high precision (96%-97%) and recall (96%-97%) for both stroke and non-stroke cases, resulting in a strong F1-score of 0.97. Overall, the model demonstrates excellent performance with balanced precision and recall, making it effective for stroke prediction.

FLASK API DEVELOPMENT:

Overview of Flask:

Flask is a lightweight and flexible web framework for Python, ideal for building APIs and web applications. It provides built-in support for routing, request handling, and template rendering, making it easy to develop and deploy machine learning models as web services.

Creating API Endpoints:

API endpoints are defined using Flask's `@app.route()` decorator, allowing users to send requests and receive responses.

The `/predict` endpoint processes input data, applies transformations, and returns stroke prediction results based on the trained model.

Loading Trained Model in Flask:

The trained model and scaler are loaded using `joblib.load()`, ensuring they are available for making predictions.

DEPLOYMENT ON RENDER:

Why Render for deployment:

Easy deployment & scalability

Free tier & Managed Services

Steps to Deploy Flask API on Render:

Prepare Your Flask App for Deployment

- Ensure all dependencies are listed in a `requirements.txt` file (`pip freeze > requirements.txt`).
- Create a `Procfile` (if deploying as a web service) with:

```
web: gunicorn app:app
```

Push Your Code to GitHub

Deploy on Render

- Go to Render.com and sign up/log in.
- Click on New Web Service and connect your GitHub repository.
- Set the runtime as Python and specify the start command as:

```
gunicorn app:app
```

- Choose a free or paid tier based on project needs.
- Click Deploy, and Render will automatically build and host your Flask API.

Deployed @ https://project4-machinelearning.onrender.com/

CSS & Bootstrap Styling:

Some issues faced:

- Bootstrap's .bg-light/dark can interfere with the mode selector and color selection which makes the color and styling difficult.
    - Causes conflicts with .btn, .container, and entire form backgrounds.
- Container restrictions have to be forced onto the CSS and manually entered into the code.

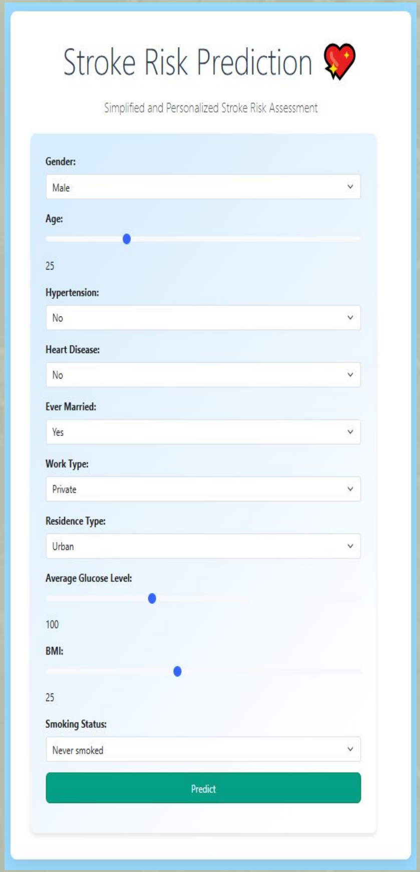
Fixes applied:

- Custom CSS after Bootstrap.
- Used !important tags to override styles applied by Bootstrap.
- Debugged with DevTools using inspect to see what exactly was going wrong in each mode.

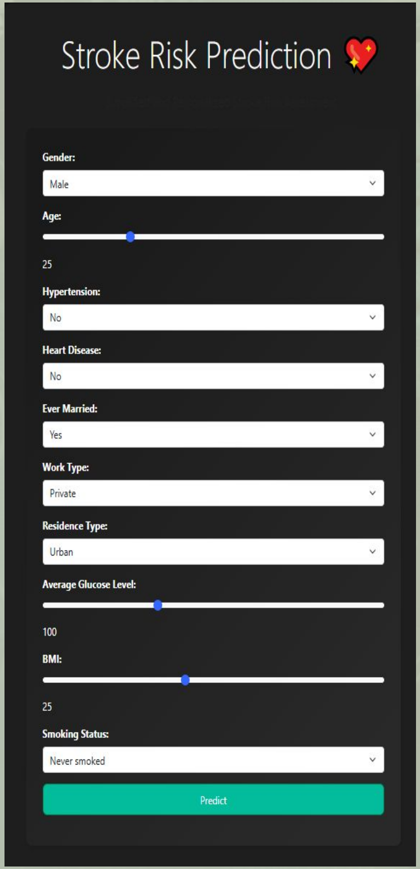
Outcome:

- XGBoost integrated with Flask API for real-time predictions.
- Website styling is fully functional with Bootstrap.
- No more conflicts with CSS that affect it's visibility or ***use***.

Light Mode



Dark Mode



Stroke Risk Prediction Tool- Frontend Interface

The app's personalized recommendations can help users reduce their stroke risk by suggesting actionable lifestyle and medical interventions based on their risk prediction

The person is at a risk of stroke:

- Consult a healthcare provider immediately.
- Quit smoking if applicable.
- Medication: Consider blood pressure or cholesterol management.
- Lifestyle Changes: Reduce salt and alcohol intake, increase physical activity

The person is not at a risk of stroke):

- Maintain current lifestyle: Continue regular exercise and a balanced diet.
- Annual Check-ups: Recommended for ongoing monitoring.
- Avoid Smoking: Continue avoiding smoking for long-term benefits.

CONCLUSION & KEY FINDINGS:

Model Performance:

The machine learning model developed for stroke prediction performed reasonably well based on accuracy and other metrics like precision, recall, and F1-score.

Potential Improvements:

The dataset used for training could benefit from additional samples to improve generalizability. Increasing the dataset size may help reduce overfitting and improve model accuracy.

Exploring Neural Networks: Further experimentation with neural networks, including fine-tuning hyperparameters, could potentially enhance model performance and provide deeper insights into stroke prediction.

Next Steps:

Continuous Monitoring: After deployment, it’s important to monitor the model's performance and retrain it periodically with updated data to maintain its accuracy over time.

Integration with Healthcare Systems: Explore opportunities for integrating the model into healthcare applications or systems to aid in early detection and prevention of strokes.

Collaboration with Experts: Collaborating with medical professionals to validate the predictions and model performance in real-world scenarios is crucial for improving reliability.