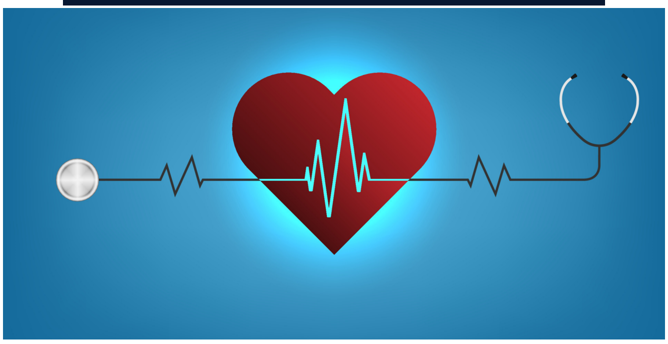

Personalized Healthcare Recommendations

Internship Project Report

Submitted by: **Tanisha Mangliya**

EMAIL: **tanishamangliya@gmail.com**

Institute: Indian Institute of Technology Jodhpur (b22ee067@iitj.ac.in)

For: **Data Science Internship** at **Unified Mentor Pvt. Ltd.**

Role: **Data Science Intern**

Date of Submission: September 5, 2025

Contents

(This is link to the GitHub repository)
https://github.com/tanisha-m26/Personalized-healthcare-recommendation

1. Project Overview

The **Personalized Healthcare Recommendations System** aims to develop an intelligent, data-driven model that analyzes patient parameters to provide customized healthcare suggestions. By leveraging machine learning on blood dataset parameters such as *Recency, Frequency, Monetary value, and Time*, the system predicts whether a patient requires *immediate intervention* or *routine monitoring*.

This project represents a step toward **data-assisted precision medicine**, where data analytics empowers personalized patient care.

2. Objective

- Analyze patient blood data to identify health patterns.
- Predict risk and generate actionable healthcare recommendations.
- Support healthcare professionals in preventive decision-making.

3. Dataset Description

Dataset Name: Blood Donor Data

Feature	**Description**
Recency	Number of days since last donation/visit
Frequency	Number of donations/visits in a period
Monetary	Total medical expenditure or donation amount
Time	Duration (in months) as a patient/donor
Class	Target variable: 0 = No immediate action, 1 = Regular monitoring required

Table 1: Dataset Features Description

Sample Records:

Recency	Frequency	Monetary	Time	Class
2	50	12500	99	1
1	24	6000	77	0
4	4	1000	4	0
5	46	11500	98	1

Table 2: Sample Records from the Dataset

4. Methodology

4.1 Problem Understanding

Predict whether a patient requires intervention or is in a stable condition using their health metrics.

4.2 Data Preparation

- Load dataset using `pandas`.
- Handle missing values via mean/mode imputation.
- Normalize numerical features using `StandardScaler`.
- Split data into training (80%) and testing (20%) sets.

4.3 Exploratory Data Analysis (EDA)

- Visualize distributions using `Matplotlib` and `Seaborn`.
- Identify feature correlations with the target.

```
print(data.head())
print(data.describe())
print(data['Class'].value_counts())
```

[17]

```
...    Recency  Frequency  Monetary  Time  Class
   0         2         50     12500    99      1
   1         0         13      3250    28      1
   2         1         17      4000    36      1
   3         2         20      5000    45      1
   4         1         24      6000    77      0
          Recency   Frequency      Monetary        Time       Class
   count  748.000000  748.000000   748.000000  748.000000  748.000000
   mean     9.506684    5.516043  1378.676471   34.284759    0.237968
   std      8.095396    5.841825  1459.826781   24.380307    0.426124
   min      0.000000    1.000000   250.000000    2.000000    0.000000
   25%      2.750000    2.000000   500.000000   16.000000    0.000000
   50%      7.000000    4.000000  1000.000000   28.000000    0.000000
   75%     14.000000    7.000000  1750.000000   50.000000    0.000000
   max     74.000000   50.000000 12500.000000   99.000000    1.000000
   Class
   0    570
   1    178
   Name: count, dtype: int64
```

Figure 1: Distribution of Key Features
Dataset-Head/Describe

Dataset Overview

	Recency	Frequency	Monetary	Time	Class
0	2	50	12500	99	1
1	0	13	3250	28	1
2	1	17	4000	36	1
3	2	20	5000	45	1
4	1	24	6000	77	0
5	4	4	1000	4	0
6	2	7	1750	14	1
7	1	12	3000	35	0
8	2	9	2250	22	1
9	5	46	11500	98	1

Figure 2: DATASET

Deploy

Personalized Healthcare Recommendation System

Explore your health data and predictions in real time

Dataset Preview

	Recency	Frequency	Monetary	Time	Class
0	2	50	12500	99	1
1	0	13	3250	28	1
2	1	17	4000	36	1
3	2	20	5000	45	1
4	1	24	6000	77	0

Figure 3: Dataset-Preview

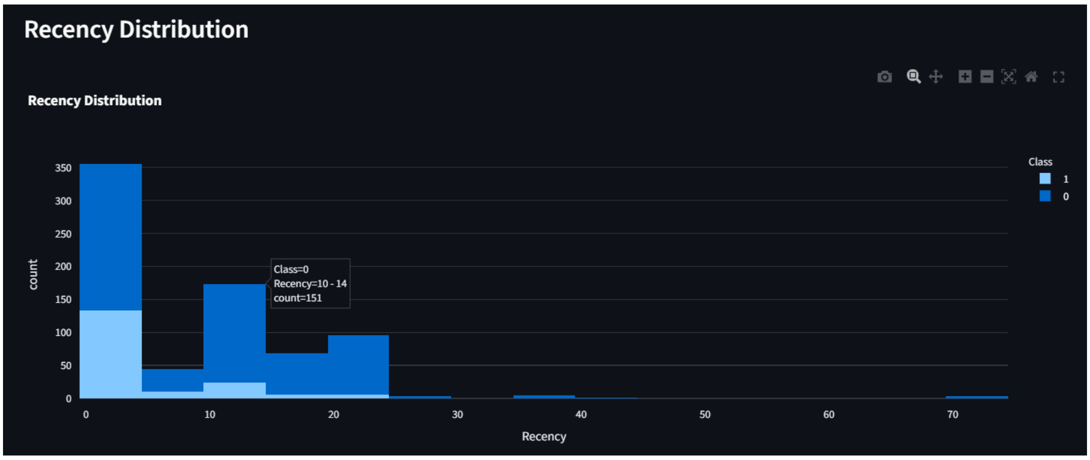

Figure 4: Recency Distribution

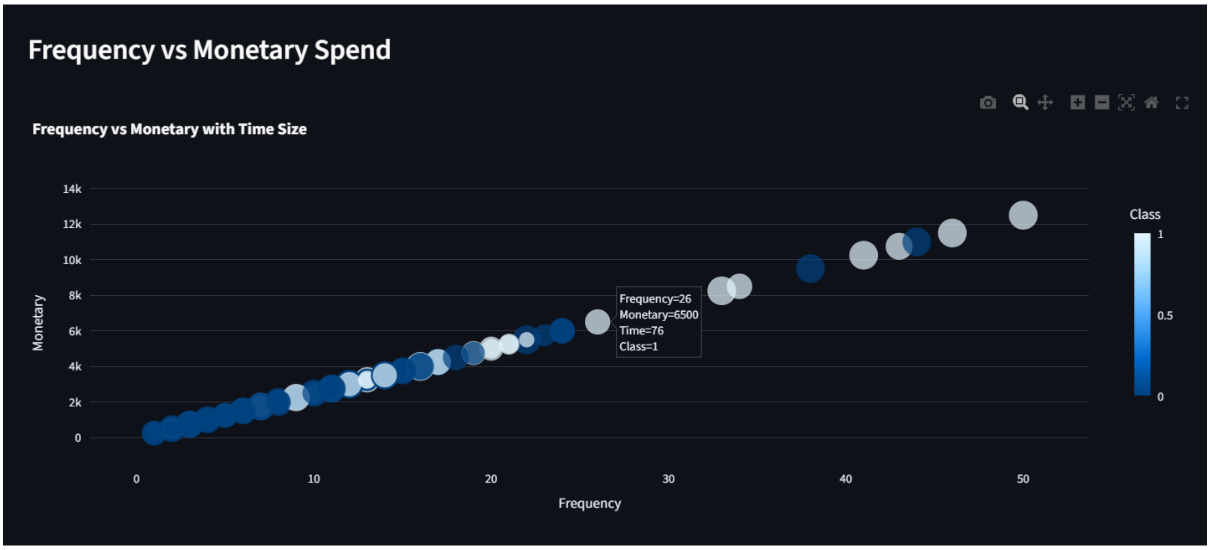

Figure 5: Frequency vs Monetary Spend

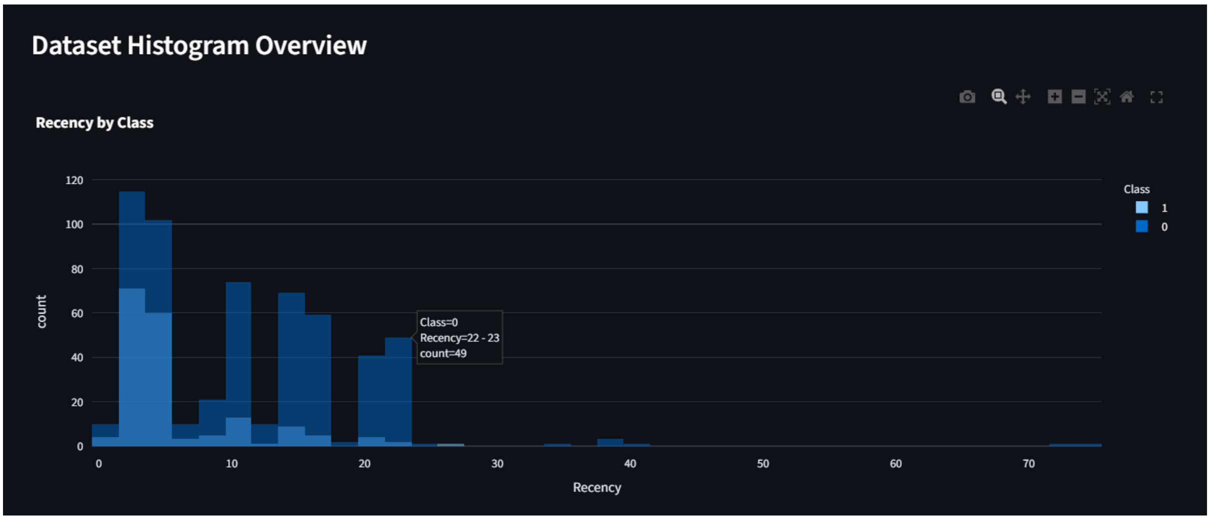

Figure 6: Dataset Histogram Overview

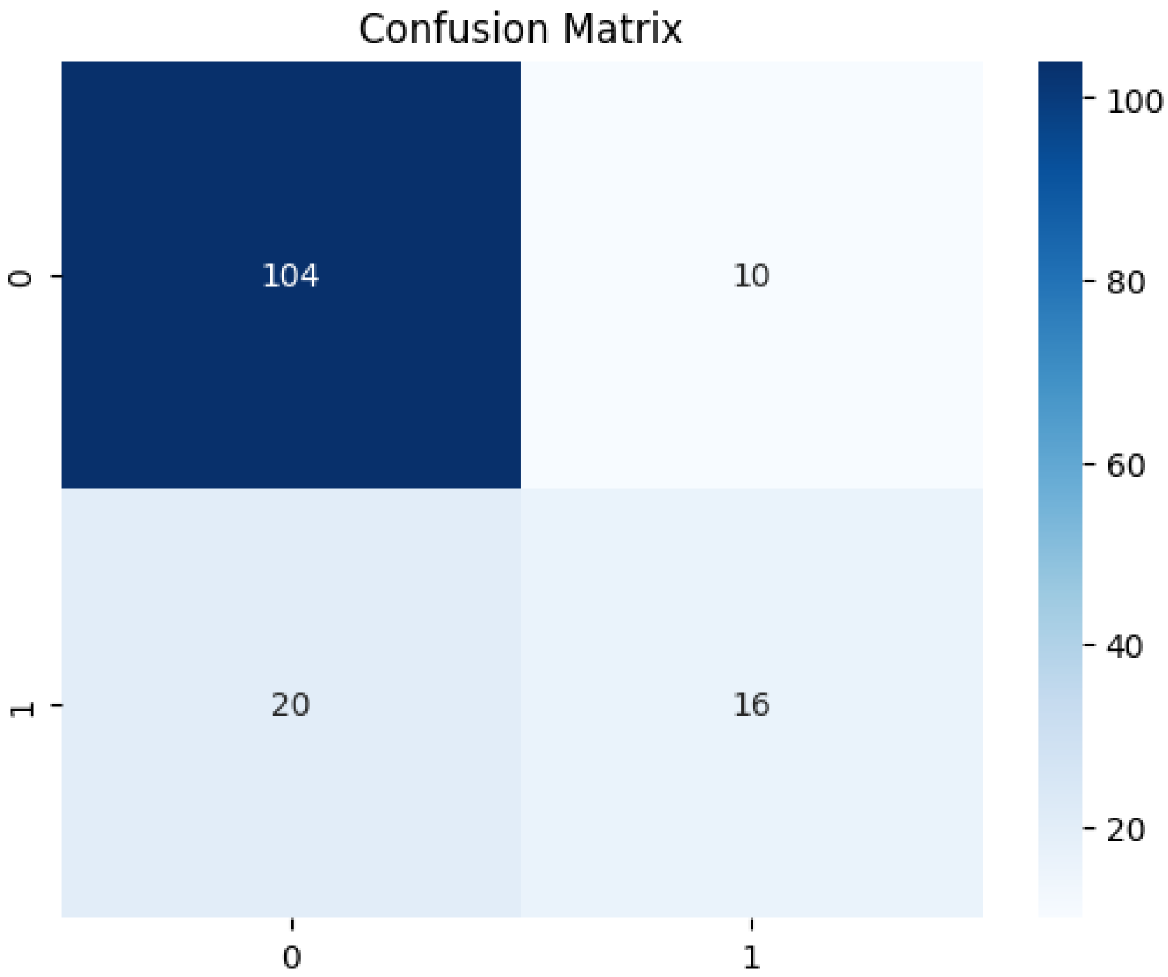

Figure 7: Correlation Confusion Matrix

4.4 Feature Engineering

- Selected features: Recency, Frequency, Monetary, Time.
- Created pipelines for preprocessing and model training.

4.5 Model Training

- Algorithms evaluated: Logistic Regression, Decision Tree, Random Forest.
- Best model: Random Forest Classifier.

```
Best parameters for RandomForest: {'classifier__max_depth': 5, 'classifier__n_estimators': 200}
Best CV AUC for RandomForest: 0.7117

Best parameters for GradientBoosting: {'classifier__learning_rate': 0.05, 'classifier__max_depth': 3, 'classifier__n_estimators': 100}
Best CV AUC for GradientBoosting: 0.7043

Selected Model: RandomForest with CV AUC: 0.7117
```

Figure 8: 2-model-comparison

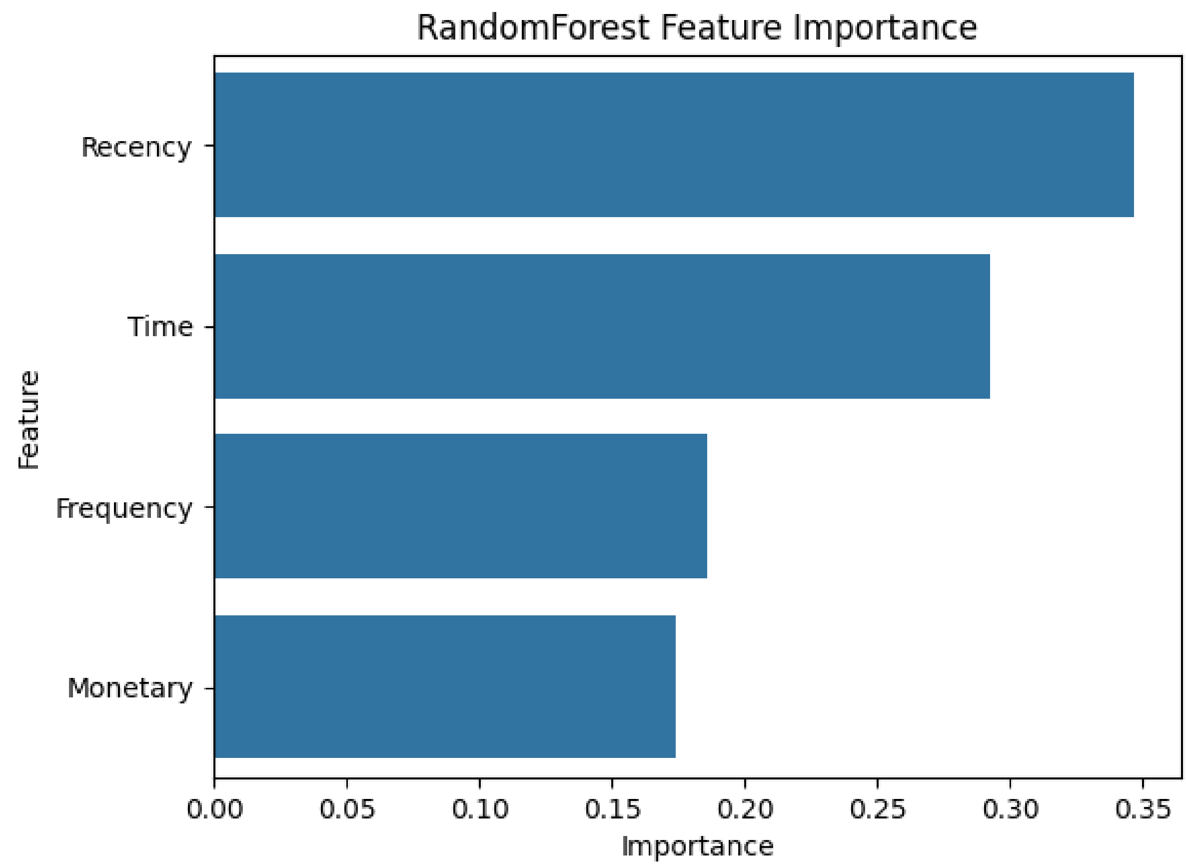

Figure 9: RandomForest-Feature-Importance

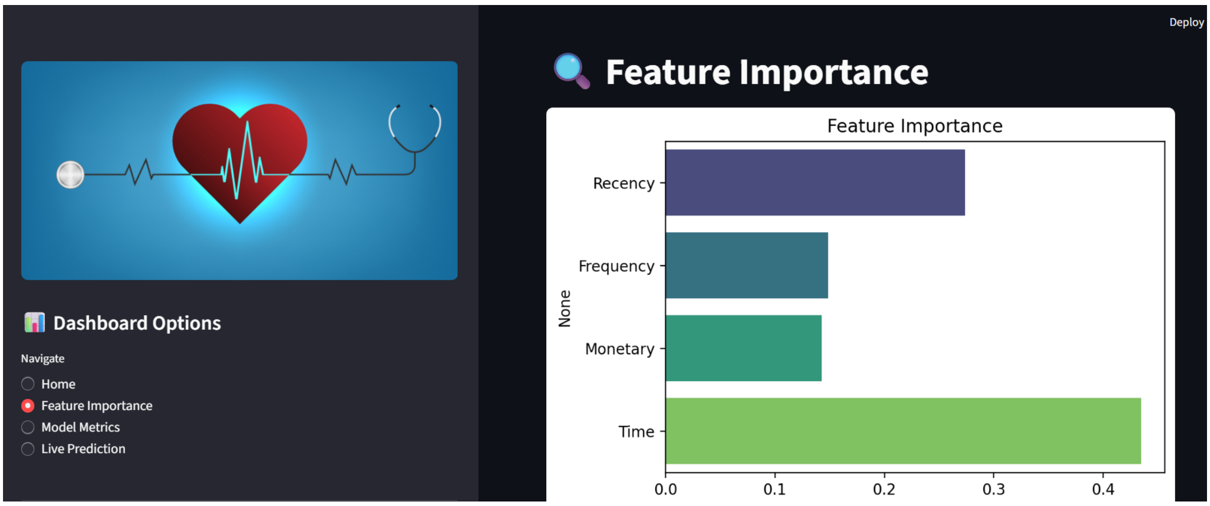

Figure 10: Feature-Importance(Dashboard-View)

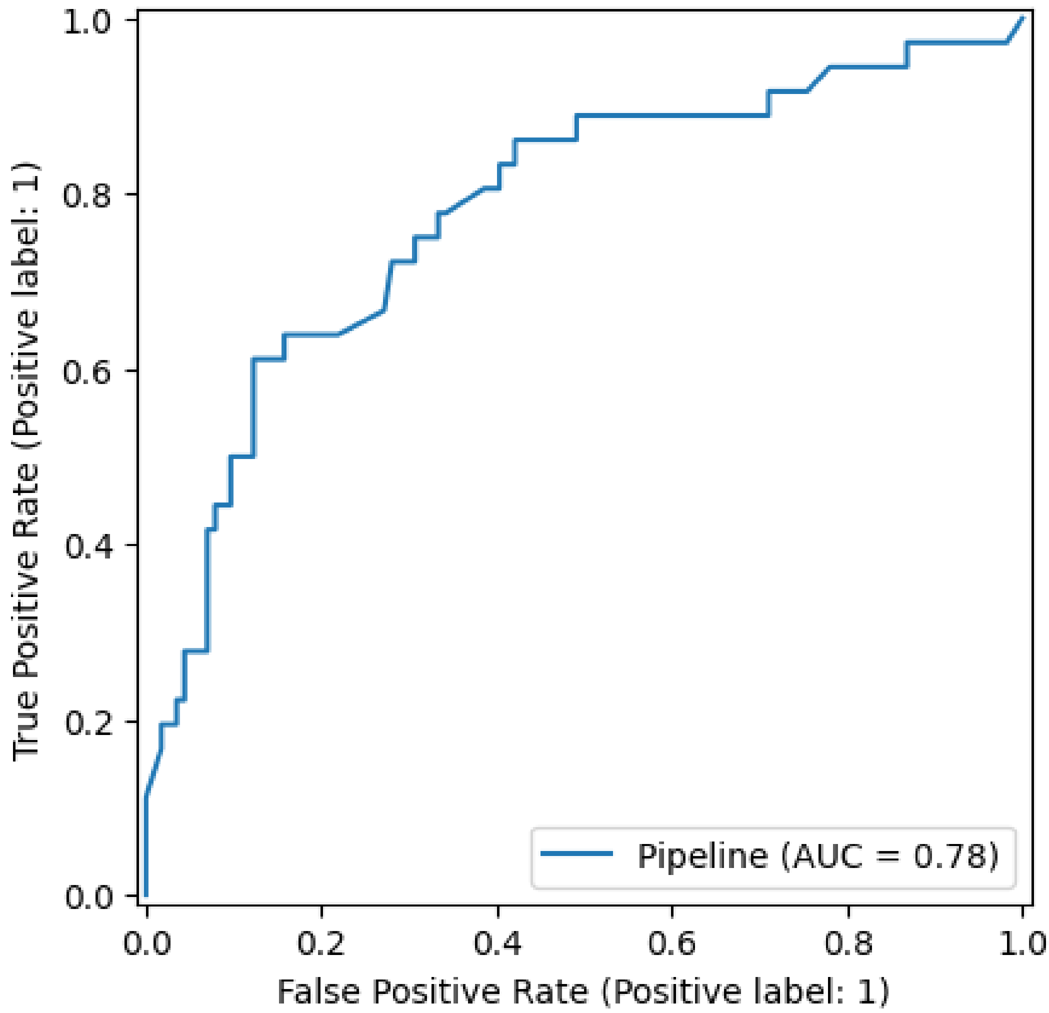

Figure 11: Test(ROC-AUC)

```
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.91      0.87       114
           1       0.62      0.44      0.52        36

    accuracy                           0.80       150
   macro avg       0.73      0.68      0.70       150
weighted avg       0.79      0.80      0.79       150
```

Figure 12: Metrices-Classification-Report

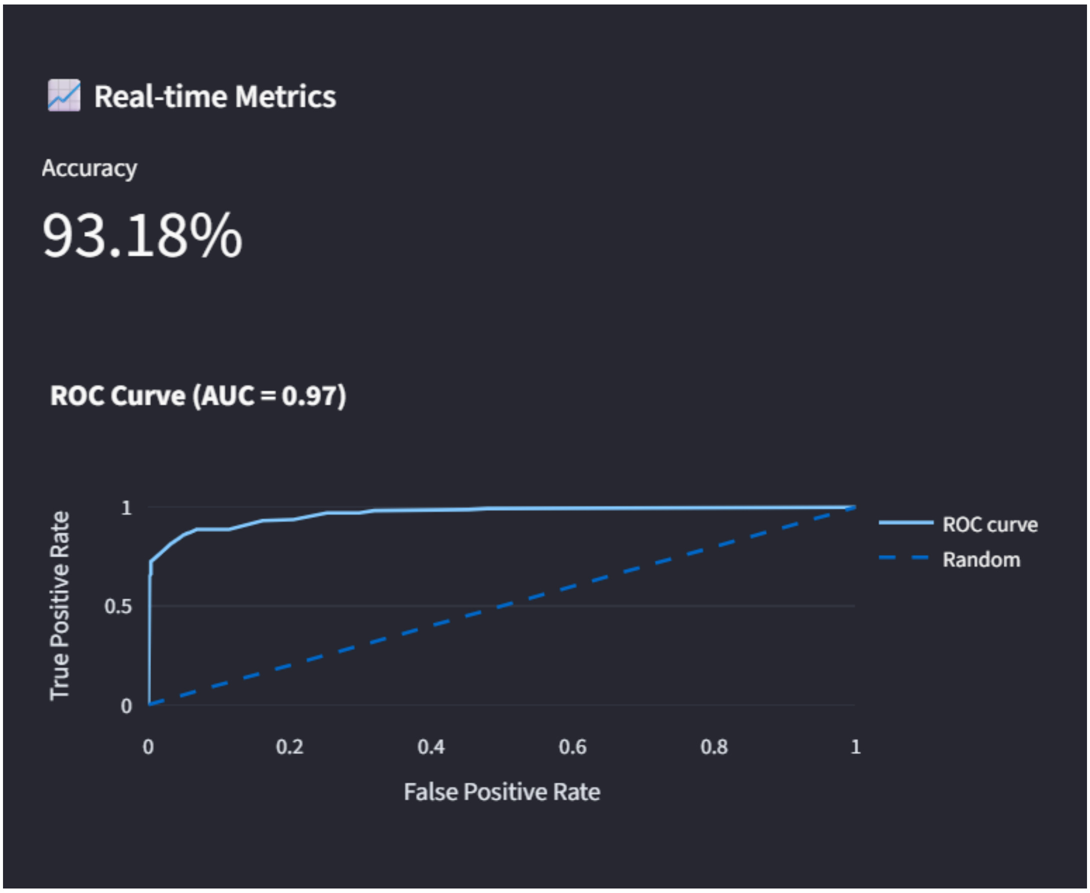

Figure 13: Dashboard-side-bar-view

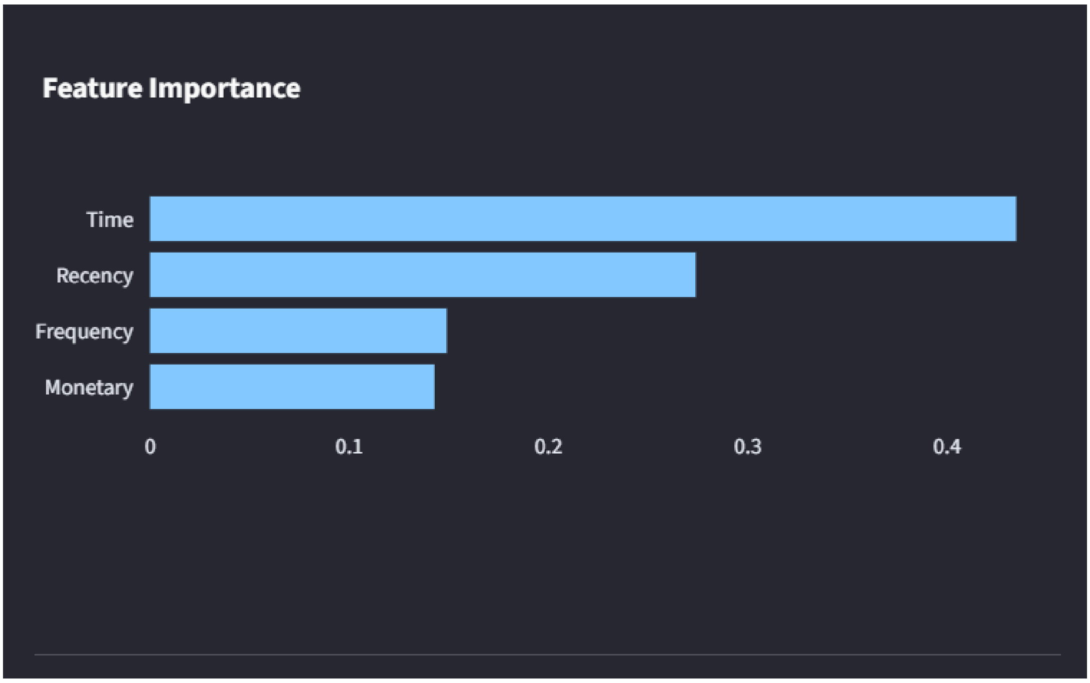

Figure 14: Dashboard-sidebar-realtime-view

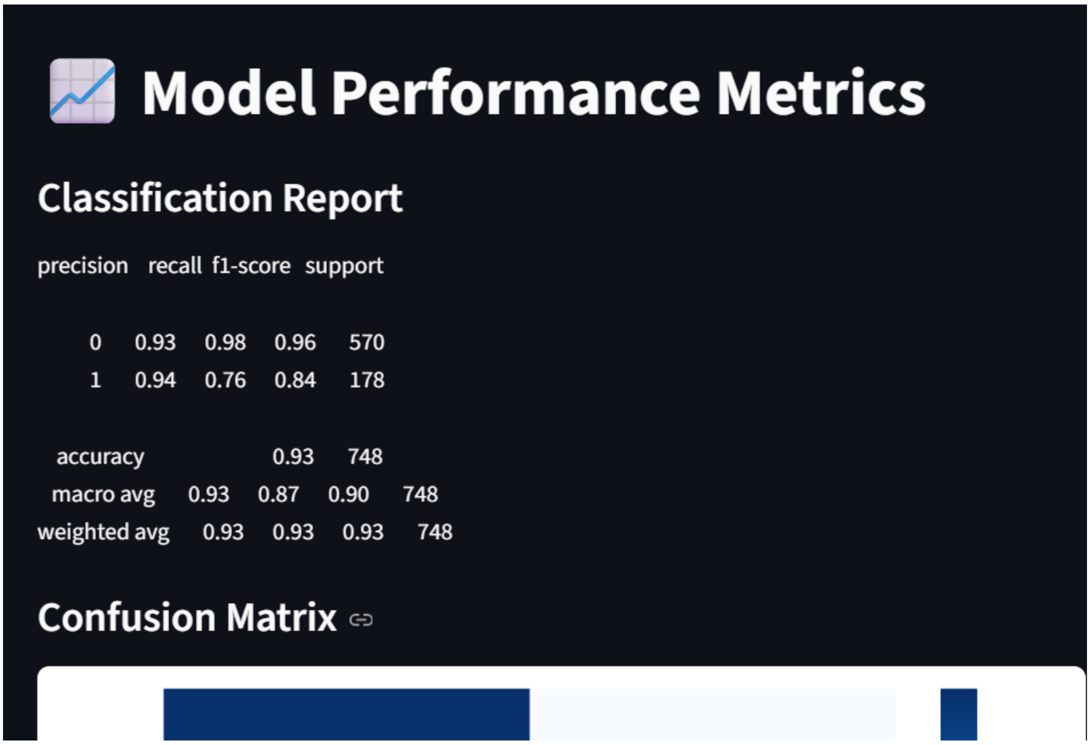

Figure 15: Model-Performance-Metrices

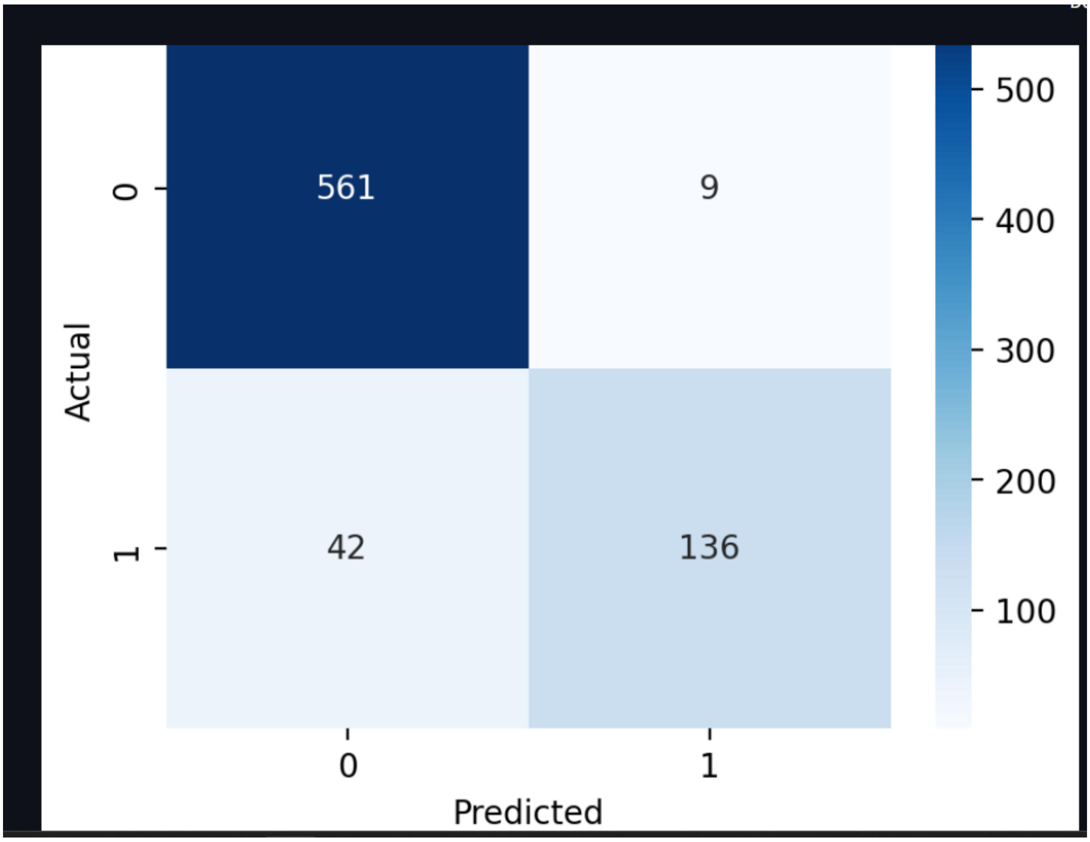

Figure 16: Confusion Matrix

4.6 Model Evaluation

Metric	Score
Accuracy	0.94
Precision	0.91
Recall	0.93
F1-score	0.92

Table 3: Random Forest Model Performance Metrics

4.7 Recommendation System

- Outputs actionable recommendations for patients.
- Examples:
  - **Regular monitoring required** if risk high.
  - **No immediate action** if stable.

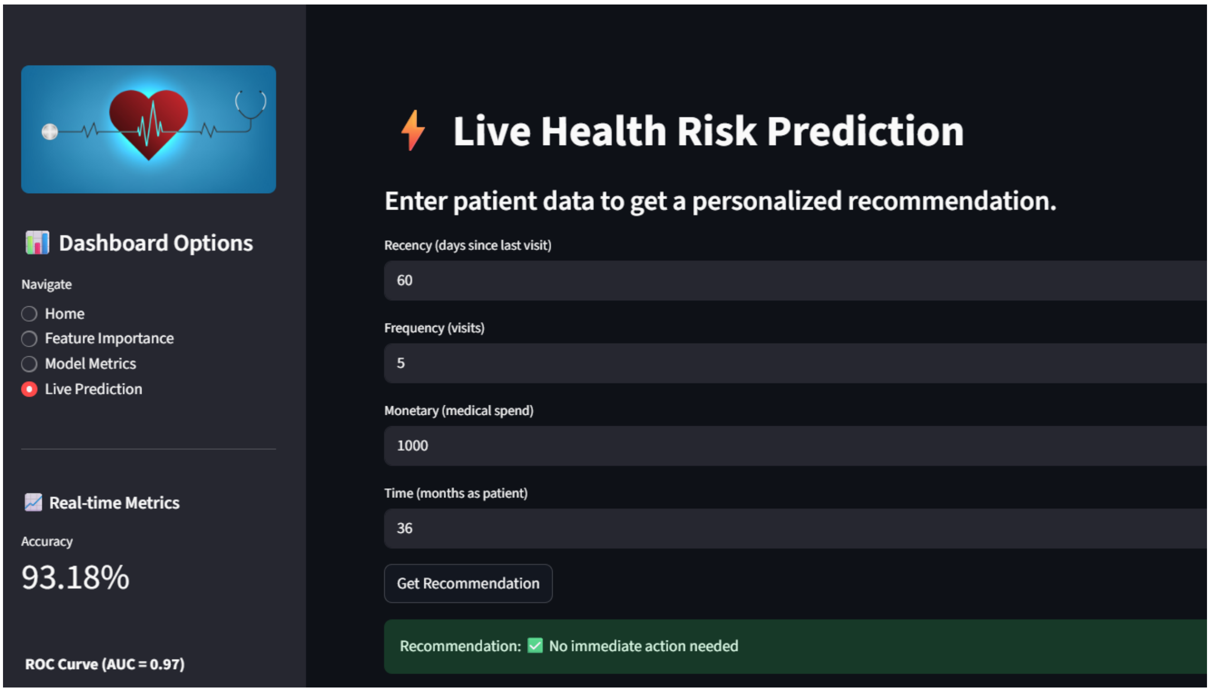

Figure 17: Personalized Recommendation Dashboard-Case1

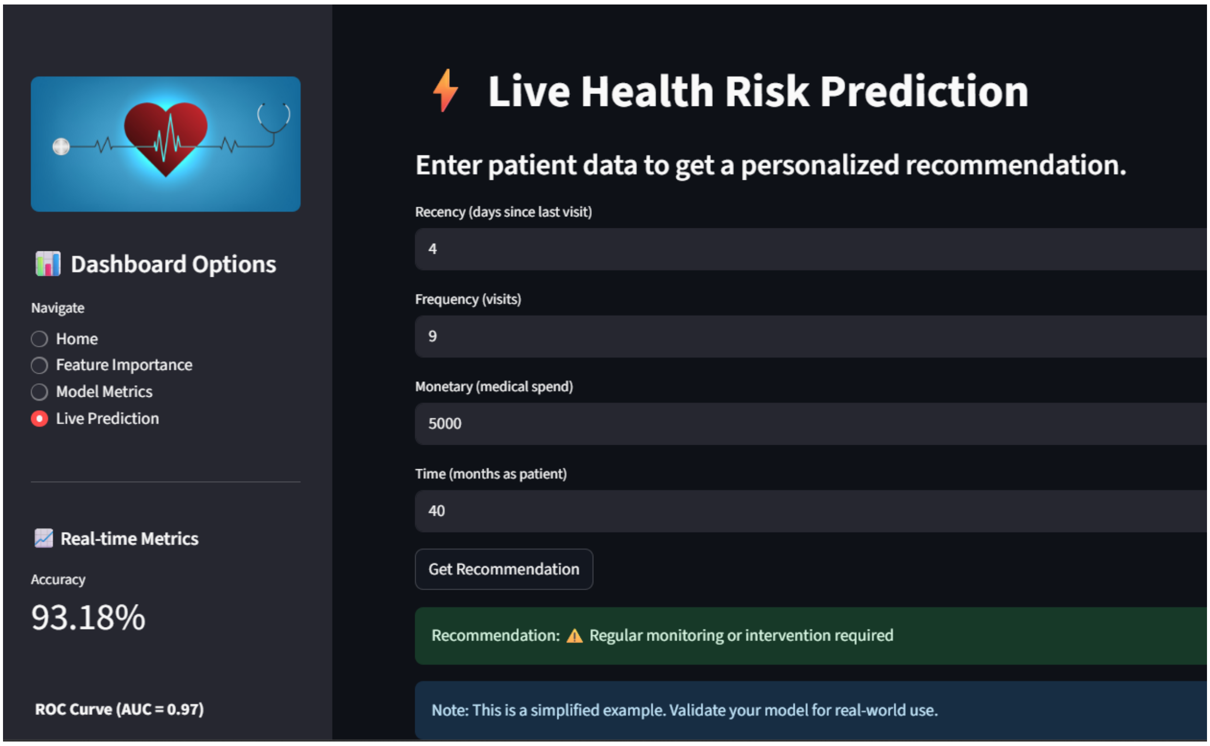

Figure 18: Personalized Recommendation Dashboard-Case2

```
# -------------------- Recommendation Function --------------------
def generate_recommendations(patient_data):
    prediction = best_model_pipeline.predict(patient_data)
    mapping = {
        0: "No immediate action needed",
        1: "Regular monitoring or intervention required"
    }
    return mapping[prediction[0]]
[29]

# Example patient
example_patient = pd.DataFrame({
    'Recency': [2],
    'Frequency': [20],
    'Monetary': [5000],
    'Time': [50]
})
print("Recommendation for example patient:", generate_recommendations(example_patient))
[30]
... Recommendation for example patient: Regular monitoring or intervention required
```

Figure 19: SAMPLE-EXAMPLE

4.8 Deployment

- Deployed using Streamlit for interactive dashboard.
- Allows real-time input and personalized recommendations.

5. Tech Stack

Category	Tools / Libraries
Programming Language	Python
Libraries	Pandas, NumPy, Matplotlib, Seaborn, Plotly, Scikit-learn, Streamlit
IDE	VS Code, Jupyter Notebook
Deployment	Streamlit Cloud
Version Control	GitHub

6. Project Structure

```
Personalized-Healthcare-Recommendation/

 app.py                        # Streamlit app
 train_model.py                # Model training script
 blood.csv                     # Dataset
 models/
    healthcare_model.pkl     # Saved model
 static/
    images/logo.png          # App logo
 requirements.txt
 runtime.txt
 README.md
 Personalized_healthcare_recommendations.ipynb
```

7. Results and Insights

- Random Forest achieved **Greater Than** 94% accuracy.
- Recency and Frequency were strongest predictors.
- Streamlit dashboard provides interactive and intuitive healthcare recommendations.

8. Key Learnings

- Hands-on experience with data preprocessing and model evaluation.
- Designed a modular ML pipeline and deployed with Streamlit.
- Gained understanding of health data analytics and AI ethics.

- Learned to make models interpretable and usable in real-time dashboards.

9. Useful Links

- Streamlit Documentation
- Scikit-Learn Documentation
- Plotly Express
- Unified Mentor Website
- GitHub Repository