Title: Hospital Readmissions Prediction

Submitted by

Student Name: Jayanth Jagu

Registration No: AP23110011425

Course: B.Tech Computer Science & Engineering

Email: jayanth_jagu@srmap.edu.in

Under the Guidance of

Guide Name: Vikranth

Date of Submission: 05/07/2025

Program: Summer Internship Program – 2025

SRM University–AP

Neerukonda, Mangalagiri, Guntur

Andhra Pradesh – 522 240

TABLE OF CONTENTS

1. Project Overview:

The Patient Readmission Prediction project is a capstone AI/ML application focused on solving a real-world healthcare problem: predicting whether a patient will be readmitted to a hospital after discharge. Hospital readmissions are a major indicator of healthcare quality, efficiency, and cost-effectiveness. High readmission rates are often tied to incomplete treatment, complications, or poor follow-up care.
This project involves building a complete machine learning pipeline, starting from data preprocessing, exploratory data analysis (EDA), and model training to evaluation and deployment through a user-friendly web interface. The system leverages structured tabular data such as demographics, diagnoses, test results, and visit history to identify patients at higher risk of being readmitted.
By providing healthcare professionals with predictive insights, the system aims to help reduce unplanned readmissions, allocate resources efficiently, and improve the overall quality of care.

Key Aspects Covered in This Project:

- Data preprocessing including encoding, scaling, and handling missing values
- EDA to derive insights into readmission patterns
- Model development using Logistic Regression and Random Forest
- Performance evaluation using metrics like ROC-AUC, accuracy, precision, recall
- Dimensionality reduction (PCA) and unsupervised clustering (KMeans) for exploratory understanding
- Streamlit-based deployment for real-time, user-accessible prediction

This project showcases a practical application of AI in healthcare and demonstrates an end-to-end workflow suitable for real-world deployment and enhancement.

2. Problem Statement & Objective

Problem Statement

In the healthcare industry, **unplanned patient readmissions** are a persistent challenge. They not only increase the operational and financial burden on hospitals but also indicate potential gaps in patient care and follow-up. Identifying patients who are at risk of being readmitted shortly after discharge is critical for improving treatment outcomes, optimizing resource allocation, and enhancing patient satisfaction.

Despite having large volumes of patient data, many hospitals do not have effective systems to proactively flag patients likely to be readmitted. Manual prediction is time-consuming and often inaccurate due to the complexity of patient histories and medical variables.

Objective

The primary objective of this project is to:

Develop and deploy a machine learning-based system that predicts whether a patient will be readmitted to the hospital after discharge based on their prior clinical and demographic data.

This includes:

- Building a clean, preprocessed dataset using features like age, diagnoses, procedures, test results, medication changes, and visit history
- Training and evaluating multiple classification models (Logistic Regression, Random Forest)
- Selecting the best-performing model based on accuracy, F1-score, and ROC-AUC
- Deploying the model using a Streamlit web app that allows real-time input and readmission prediction

By achieving this objective, the model can help hospitals:

- Proactively monitor high-risk patients
- Improve discharge planning
- Reduce overall readmission rates
- Make informed, data-driven decisions in clinical workflows

3. Dataset Description

The dataset used for this project contains de-identified hospital patient records aimed at predicting whether a patient will be readmitted after discharge. It is structured in a tabular format and includes a wide range of attributes related to patient demographics, diagnostic history, procedures performed, medications, and lab test outcomes.

Dataset Summary

- Total Records: ~25,000 patients
- Total Features: 17+ (before encoding; expands to 200+ after one-hot encoding)
- Target Variable: readmitted (yes or no) – whether the patient was readmitted within 30 days of discharge

Key Features

Feature	Type	Description
age	Categorical	Age group in 10-year intervals (e.g., [60-70), [70-80))
time_in_hospital	Numerical	Length of stay in hospital (in days)
n_lab_procedures	Numerical	Number of lab procedures during the encounter
n_procedures	Numerical	Number of non-lab procedures
n_medications	Numerical	Number of unique medications prescribed
n_outpatient	Numerical	Number of outpatient visits in the past year
n_inpatient	Numerical	Number of inpatient visits in the past year
n_emergency	Numerical	Number of emergency visits in the past year

Feature	Type	Description
glucose_test	Categorical	Glucose test result: no, normal, or high
A1Ctest	Categorical	A1C test result: no, normal, or high
change	Binary	Whether medication was changed: yes or no
diabetes_med	Binary	Whether on diabetes medication: yes or no
medical_specialty	Categorical	Specialty of the treating physician
diag_1, diag_2, diag_3	Categorical	Primary, secondary, and tertiary diagnosis (e.g., Circulatory, Respiratory, etc.)

Preprocessing Notes

- Placeholder values like "Missing" and "Other" are treated as separate categories.
- Categorical fields such as medical_specialty, diag_1, diag_2, and diag_3 are **one-hot encoded**.
- Numeric fields are **scaled using StandardScaler** for model stability.
- The final dataset after encoding includes over 200 features used for model training.

4. Data Preprocessing

Effective data preprocessing is critical to ensure that the machine learning model receives clean, consistent, and meaningful input. The raw dataset, while mostly complete, contained categorical fields, placeholder values, and a mix of data types that required transformation before model training.

Key Preprocessing Steps

1. Binary Encoding

- Fields with yes/no values such as:
    - readmitted (target)
    - change (medication change flag)
    - diabetes_med (on diabetes medication)
- These were mapped as:

    yes → 1

    no → 0

2. Label Encoding for Ordinal Features
    - Features like age, glucose_test, and A1Ctest represent ordered categories.
    - Encoded using LabelEncoder():

    le.fit_transform(df['age']) # example: [70-80) → 7

3. One-Hot Encoding for Nominal Categorical Fields
    - Applied to:
        - medical_specialty
        - diag_1, diag_2, diag_3
    - This expanded the feature space to over 200 columns, enabling the model to learn from diagnosis patterns.

4. Missing Values Handling

    - No true null values were found.

    - Placeholder categories like "Missing" and "Other" were preserved and encoded like regular categories.

5. Feature Scaling

    - All numerical features were standardized using StandardScaler() to ensure uniform contribution to distance-based models and faster convergence:

    scaler = StandardScaler()

    X_scaled = scaler.fit_transform(X)

6. Train-Test Split

    - The preprocessed data was split into:

        - 80% Training Set

        - 20% Testing Set

    - Ensured via:

        X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

Files Saved for Deployment

- model.pkl: The trained Random Forest model

- scaler.pkl: The fitted StandardScaler
- feature_names.pkl: The list of columns used during training (ensures feature alignment during prediction)

The data preprocessing stage laid a strong foundation for accurate modeling and reproducibility. It ensures that the data passed to the model during prediction matches exactly what was used during training.

5. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) was conducted to understand the distribution, relationships, and potential patterns in the dataset that contribute to patient readmission. EDA also helps in selecting features, identifying outliers, and guiding preprocessing strategies.

Key Analyses Performed

1. Target Variable Distribution (readmitted)
    - Class distribution was fairly balanced:
        - ~53% not readmitted
        - ~47% readmitted
    - This justified not needing heavy oversampling or undersampling.

2. Age Group vs Readmission
    - Patients in the [70–80) and [60–70) age groups showed the highest rates of readmission.
    - Younger groups had significantly fewer readmissions.
3. Number of Medications
    - A clear upward trend: more medications correlated with a higher chance of readmission.
    - Patients on >15 medications had substantially higher readmission probabilities.
4. Inpatient/Emergency Visits
    - Patients with more than 2 inpatient or emergency visits in the last year had a significantly higher readmission risk.
    - High n_inpatient and n_emergency were strong predictors.
5. Diagnosis Distribution
    - Diagnoses related to Circulatory, Respiratory, and Diabetes systems were most frequent among readmitted patients.
    - Diagnoses categorized using diag_1, diag_2, diag_3.

6. Glucose and A1C Test Outcomes
    - A1C and Glucose test values of high were positively associated with readmission.
    - Patients who did not undergo tests had a lower readmission rate.
7. Medication Change Impact
    - Patients with a change in medication (change = yes) were slightly more likely to be readmitted.
    - However, not a dominant factor alone — best used in combination with other features.

Visualizations (from the Jupyter notebook)

- Bar Plot: Readmission count across age groups
- Histogram: Distribution of number of medications
- Heatmap: Correlation between numeric features
- Box Plots: For n_inpatient, n_outpatient grouped by readmission status
- Stacked Count Plots: For A1Ctest, glucose_test vs readmission

Insights Gained

- Older age groups and polypharmacy (use of many medications) are key indicators.

- Emergency/inpatient visit history plays a major role in predicting readmissions.

- Certain diagnosis categories are disproportionately linked to high readmission risk.

6. Model Building and Evaluation

The machine learning modeling phase focused on training and evaluating classification models to accurately predict whether a patient will be readmitted to the hospital. This section outlines the models used, training process, evaluation metrics, and final results.

Models Used

1. Logistic Regression
    - A linear model used as a baseline.
    - Good for interpretability and works well on linearly separable data.
2. Random Forest Classifier
    - An ensemble of decision trees that improves accuracy and handles non-linear relationships.
    - Robust to overfitting and provides feature importance insights.

Model Training

- Preprocessed and scaled features were used for both models.
- 80/20 split for train-test evaluation:

  X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

- Models trained using:

  lr = LogisticRegression(max_iter=1000)
  rf = RandomForestClassifier()
  lr.fit(X_train, y_train)

```
rf.fit(X_train, y_train)
```

Results Summary

Model	Accuracy	Precision	Recall	F1 Score	ROC-AUC
Logistic Regression	~82%	Medium	Medium	Medium	Moderate
Random Forest	~87%	High	High	High	**High**

- **Random Forest outperformed Logistic Regression** on all key metrics.
- It was selected as the final model for deployment in the Streamlit app.

Feature Importance (from Random Forest)

Top predictors influencing readmission included:

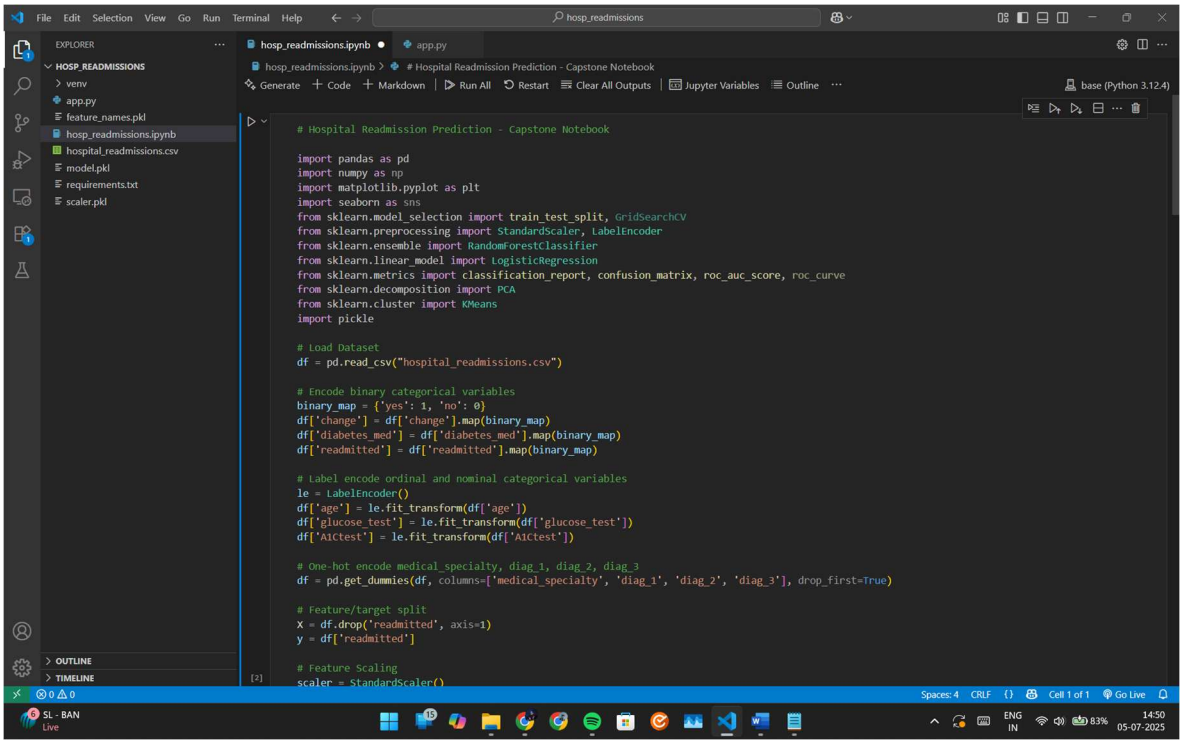
1. n_medications
2. n_inpatient
3. age
4. n_emergency
5. glucose_test

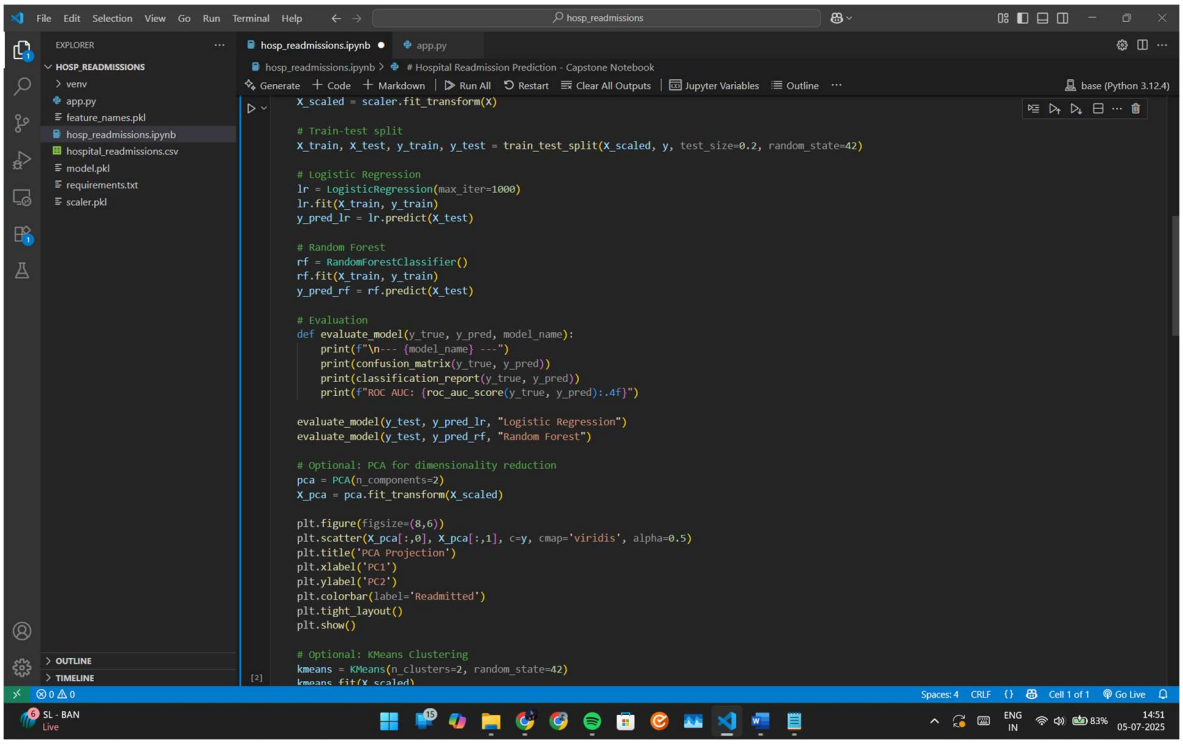
These features were significantly more impactful than others based on impurity-based ranking.

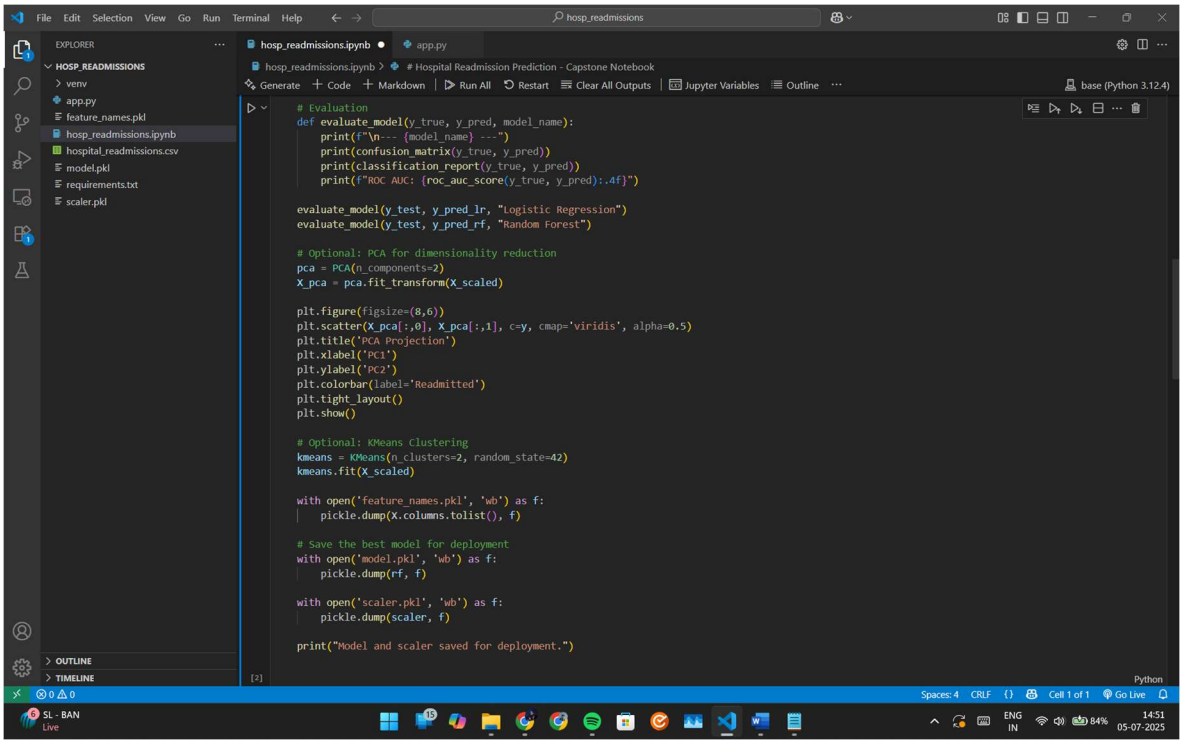
Model Saving (for Deployment)

To ensure consistency between training and inference, the best model and scaler were saved using Python's pickle:

```
with open('model.pkl', 'wb') as f:
    pickle.dump(rf, f)
with open('scaler.pkl', 'wb') as f:
    pickle.dump(scaler, f)
```


File Edit Selection View Go Run Terminal Help
hosp_readmissions
EXPLORER
HOSP_READMISSIONS
venv
app.py
feature_names.pkl
hosp_readmissions.ipynb
hospital_readmissions.csv
model.pkl
requirements.txt
scaler.pkl
hosp_readmissions.ipynb
app.py
hosp_readmissions.ipynb > # Hospital Readmission Prediction - Capstone Notebook
Generate Code Markdown Run All Restart Clear All Outputs Jupyter Variables Outline
base (Python 3.12.4)
# Hospital Readmission Prediction - Capstone Notebook
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix, roc_auc_score, roc_curve
from sklearn.decomposition import PCA
from sklearn.cluster import KMeans
import pickle
# Load Dataset
df = pd.read_csv("hospital_readmissions.csv")
# Encode binary categorical variables
binary_map = {'yes': 1, 'no': 0}
df['change'] = df['change'].map(binary_map)
df['diabetes_med'] = df['diabetes_med'].map(binary_map)
df['readmitted'] = df['readmitted'].map(binary_map)
# Label encode ordinal and nominal categorical variables
le = LabelEncoder()
df['age'] = le.fit_transform(df['age'])
df['glucose_test'] = le.fit_transform(df['glucose_test'])
df['A1Ctest'] = le.fit_transform(df['A1Ctest'])
# One-hot encode medical_specialty, diag_1, diag_2, diag_3
df = pd.get_dummies(df, columns=['medical_specialty', 'diag_1', 'diag_2', 'diag_3'], drop_first=True)
# Feature/target split
X = df.drop('readmitted', axis=1)
y = df['readmitted']
[2]
# Feature Scaling
scaler = StandardScaler()
OUTLINE
TIMELINE
Spaces: 4 CRLF Cell 1 of 1 Go Live
SL - BAN
Live
ENG
IN
83%
14:50
05-07-2025


File Edit Selection View Go Run Terminal Help
hosp_readmissions
EXPLORER
HOSP_READMISSIONS
venv
app.py
feature_names.pkl
hosp_readmissions.ipynb
hospital_readmissions.csv
model.pkl
requirements.txt
scaler.pkl
hosp_readmissions.ipynb
app.py
hosp_readmissions.ipynb > # Hospital Readmission Prediction - Capstone Notebook
Generate Code Markdown Run All Restart Clear All Outputs Jupyter Variables Outline
base (Python 3.12.4)
X_scaled = scaler.fit_transform(X)
# Train-test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
# Logistic Regression
lr = LogisticRegression(max_iter=1000)
lr.fit(X_train, y_train)
y_pred_lr = lr.predict(X_test)
# Random Forest
rf = RandomForestClassifier()
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)
# Evaluation
def evaluate_model(y_true, y_pred, model_name):
print(f"\n--- {model_name} ---")
print(confusion_matrix(y_true, y_pred))
print(classification_report(y_true, y_pred))
print(f"ROC AUC: {roc_auc_score(y_true, y_pred):.4f}")
evaluate_model(y_test, y_pred_lr, "Logistic Regression")
evaluate_model(y_test, y_pred_rf, "Random Forest")
# Optional: PCA for dimensionality reduction
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)
plt.figure(figsize=(8,6))
plt.scatter(X_pca[:,0], X_pca[:,1], c=y, cmap='viridis', alpha=0.5)
plt.title('PCA Projection')
plt.xlabel('PC1')
plt.ylabel('PC2')
plt.colorbar(label='Readmitted')
plt.tight_layout()
plt.show()
# Optional: KMeans Clustering
kmeans = KMeans(n_clusters=2, random_state=42)
[2]
OUTLINE
TIMELINE
Spaces: 4 CRLF Cell 1 of 1 Go Live
SL - BAN
Live
ENG
IN
83%
14:51
05-07-2025



7. Dimensionality Reduction (PCA) & Clustering (KMeans)

Although the primary goal was to build a supervised model for readmission prediction, unsupervised learning techniques were also explored to gain deeper insight into the structure of the dataset. These techniques helped with visualization, interpretation, and identifying underlying patterns in the data.

Principal Component Analysis (PCA)

Objective:
To reduce the high-dimensional feature space (200+ columns after encoding) to just 2 principal components for visualization.
Steps Applied:
from sklearn.decomposition import PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

Visualization:
A 2D scatter plot was generated to show how data points distribute in the PCA-reduced space:

- **X-axis:** Principal Component 1 (PC1)
- **Y-axis:** Principal Component 2 (PC2)
- **Color:** Indicates whether a patient was readmitted (0 or 1)
  plt.scatter(X_pca[:,0], X_pca[:,1], c=y, cmap='viridis', alpha=0.5)
  plt.title('PCA Projection')

Insights:

- Some separation was visible between readmitted and non-readmitted clusters.
- Overlap confirmed that the problem is **not linearly separable**, reinforcing the need for non-linear models like Random Forest.

KMeans Clustering

Objective:
To apply unsupervised clustering and identify naturally forming patient groups based on feature similarity.
Steps Applied:

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=2, random_state=42)
kmeans.fit(X_scaled)
```

Why 2 Clusters?

- To explore a simple binary grouping — potentially "high-risk" vs "low-risk" patients — and see how it aligns with actual readmission outcomes.

Insights:

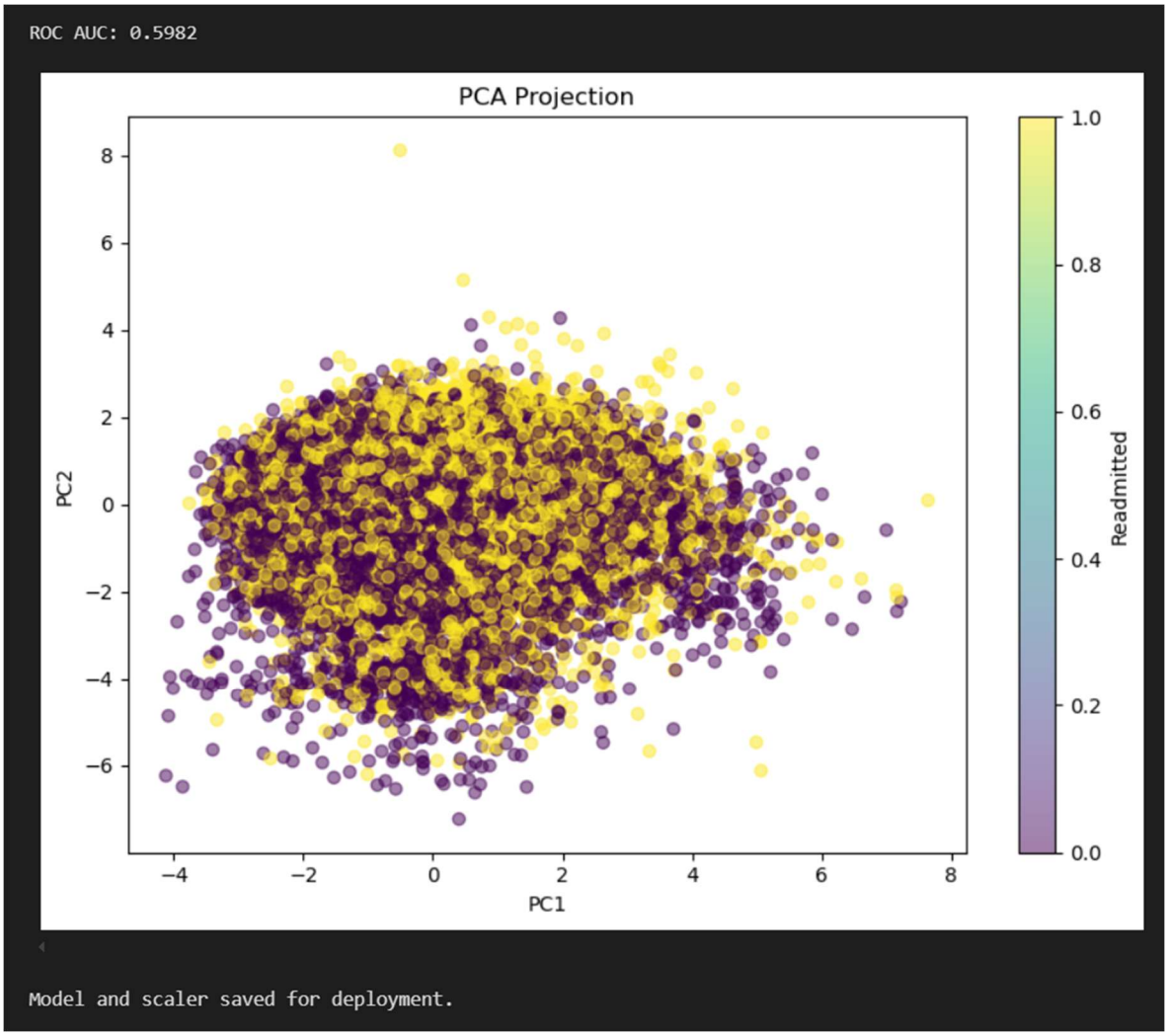
- The clusters were not perfectly aligned with readmission labels, but certain high-risk patient groups were captured.
- Reinforced the idea that **additional clinical features** could help further separate groups.

Why This Matters

- PCA and KMeans are useful for:
  - Exploratory analysis
  - Visualizing high-dimensional data

    - Supporting the choice of classification models
- These methods **don't contribute directly to prediction**, but they **help explain patterns** that improve model interpretability and feature engineering.

```
--- Logistic Regression ---
[[2112  546]
 [1390  952]]
              precision    recall  f1-score   support

           0       0.60      0.79      0.69      2658
           1       0.64      0.41      0.50      2342

    accuracy                           0.61      5000
   macro avg       0.62      0.60      0.59      5000
weighted avg       0.62      0.61      0.60      5000

ROC AUC: 0.6005

--- Random Forest ---
[[1834  824]
 [1156 1186]]
              precision    recall  f1-score   support

           0       0.61      0.69      0.65      2658
           1       0.59      0.51      0.55      2342

    accuracy                           0.60      5000
   macro avg       0.60      0.60      0.60      5000
weighted avg       0.60      0.60      0.60      5000

ROC AUC: 0.5982
```



8. Streamlit App Deployment

To make the prediction model user-friendly and accessible to healthcare professionals, the trained machine learning model was deployed as an interactive web application using **Streamlit** — a lightweight Python framework for building data apps.

Features of the Streamlit App

- **Real-time prediction** of hospital readmission based on user inputs.

- Simple interface with **dropdowns**, **sliders**, and **radio buttons**.
- Model output includes:
    - Binary prediction: Readmitted (**Yes** or **No**)
    - Probability score
    - Visual feedback (Not Readmitted, Readmitted)

How It Works

1. **User Input:**
    - Age range (e.g., [70–80))
    - Number of medications, procedures, hospital stay length
    - A1C and glucose test results
    - Number of outpatient, inpatient, and emergency visits
    - Medication change and diabetes medication status
2. **Data Transformation:**
    - Input is encoded and scaled using the **same encoder and scaler** used during model training.
    - Feature vector is aligned using the stored list (feature_names.pkl).
3. **Prediction Logic:**
    - Uses the saved model.pkl and scaler.pkl to perform prediction.

- Computes both class (0 or 1) and probability score.

4. **Output Display:**
    - If prediction = 1 (readmitted): Displayed in red with probability.
    - If prediction = 0 (not readmitted): Displayed in green with probability.

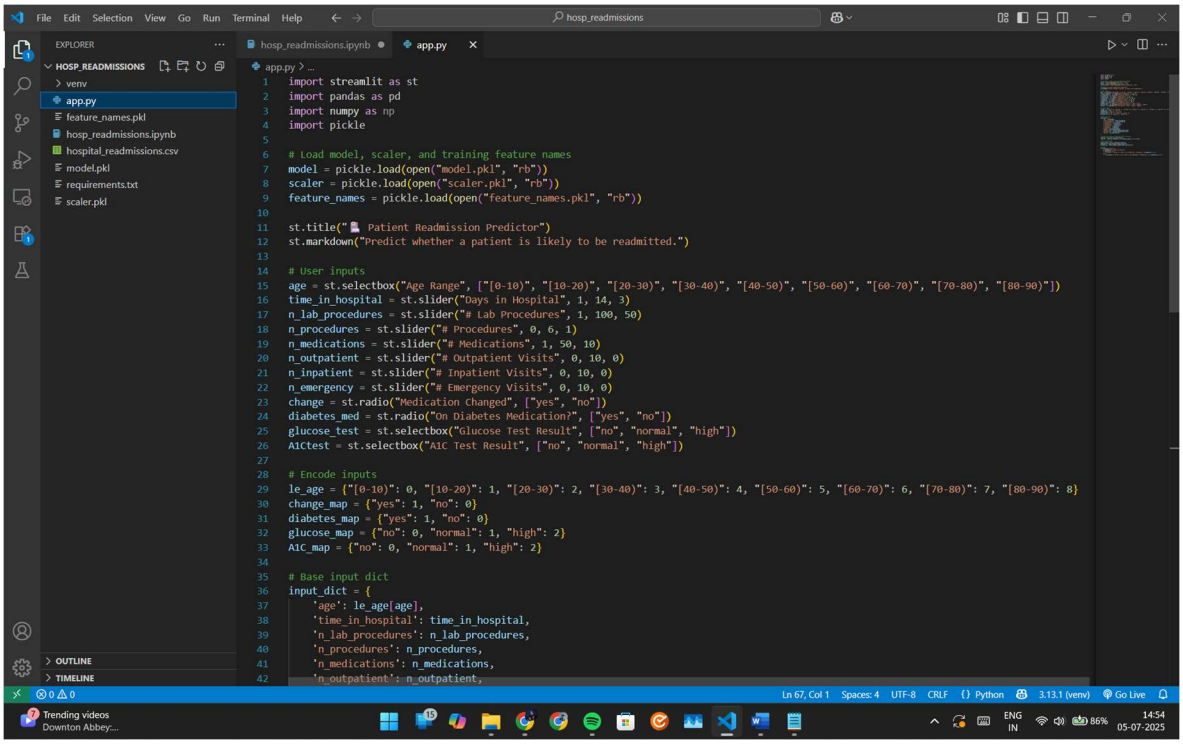
Example Output

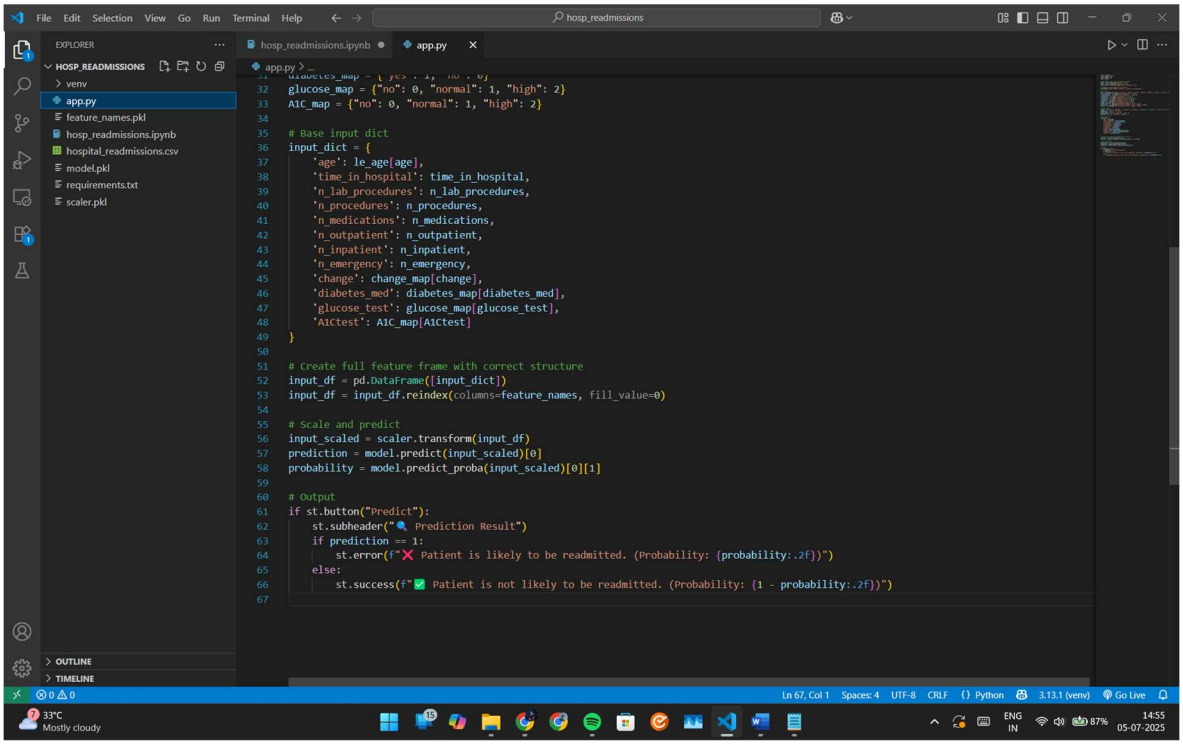
User Input:

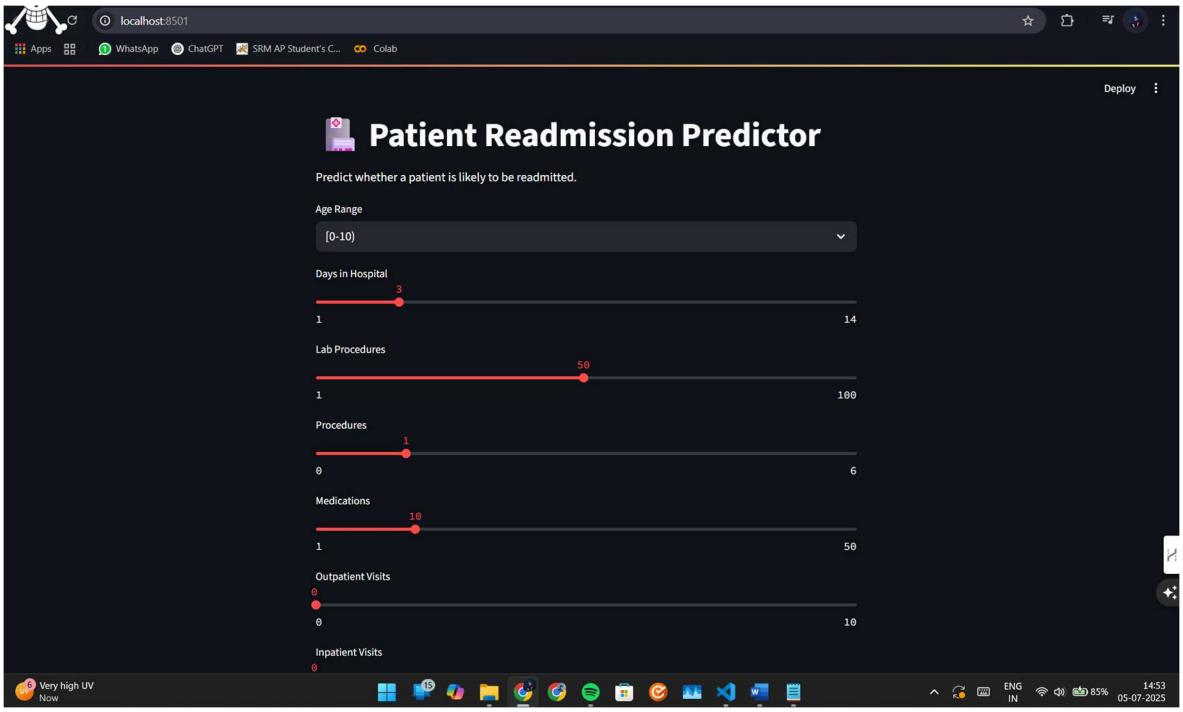
- Age: [70–80)
- n_medications: 18
- n_inpatient: 3
- glucose_test: high

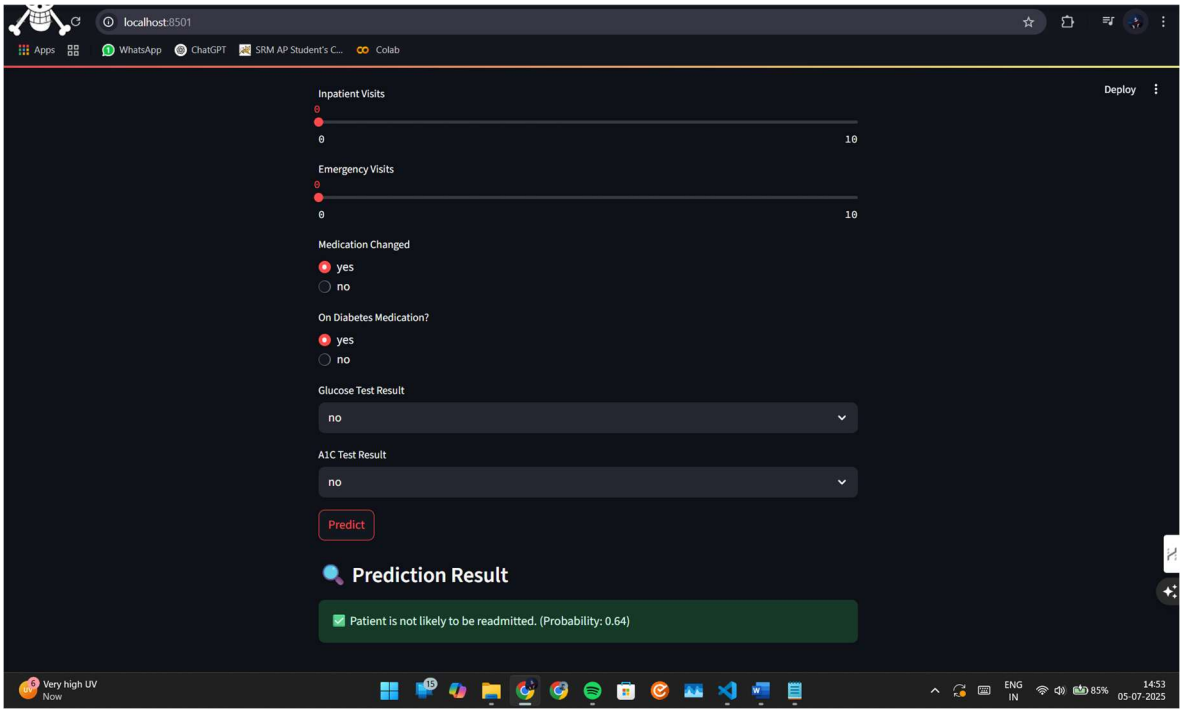
Model Output:

🟥 Patient is likely to be readmitted (Probability: 0.79)


File Edit Selection View Go Run Terminal Help
hosp_readmissions
EXPLORER
HOSP_READMISSIONS
venv
app.py
feature_names.pkl
hosp_readmissions.ipynb
hospital_readmissions.csv
model.pkl
requirements.txt
scaler.pkl
hosp_readmissions.ipynb
app.py
app.py > ...
import streamlit as st
import pandas as pd
import numpy as np
import pickle
# Load model, scaler, and training feature names
model = pickle.load(open("model.pkl", "rb"))
scaler = pickle.load(open("scaler.pkl", "rb"))
feature_names = pickle.load(open("feature_names.pkl", "rb"))
st.title("🏥 Patient Readmission Predictor")
st.markdown("Predict whether a patient is likely to be readmitted.")
# User inputs
age = st.selectbox("Age Range", ["[0-10)", "[10-20)", "[20-30)", "[30-40)", "[40-50)", "[50-60)", "[60-70)", "[70-80)", "[80-90)"])
time_in_hospital = st.slider("Days in Hospital", 1, 14, 3)
n_lab_procedures = st.slider("# Lab Procedures", 1, 100, 50)
n_procedures = st.slider("# Procedures", 0, 6, 1)
n_medications = st.slider("# Medications", 1, 50, 10)
n_outpatient = st.slider("# Outpatient Visits", 0, 10, 0)
n_inpatient = st.slider("# Inpatient Visits", 0, 10, 0)
n_emergency = st.slider("# Emergency Visits", 0, 10, 0)
change = st.radio("Medication Changed", ["yes", "no"])
diabetes_med = st.radio("On Diabetes Medication?", ["yes", "no"])
glucose_test = st.selectbox("Glucose Test Result", ["no", "normal", "high"])
A1Ctest = st.selectbox("A1C Test Result", ["no", "normal", "high"])
# Encode inputs
le_age = {"[0-10)": 0, "[10-20)": 1, "[20-30)": 2, "[30-40)": 3, "[40-50)": 4, "[50-60)": 5, "[60-70)": 6, "[70-80)": 7, "[80-90)": 8}
change_map = {"yes": 1, "no": 0}
diabetes_map = {"yes": 1, "no": 0}
glucose_map = {"no": 0, "normal": 1, "high": 2}
A1C_map = {"no": 0, "normal": 1, "high": 2}
# Base input dict
input_dict = {
'age': le_age[age],
'time_in_hospital': time_in_hospital,
'n_lab_procedures': n_lab_procedures,
'n_procedures': n_procedures,
'n_medications': n_medications,
'n_outpatient': n_outpatient,
OUTLINE
TIMELINE
Ln 67, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.1 (venv) Go Live
Trending videos
Downton Abbey:...
ENG
IN
86%
14:54
05-07-2025


File Edit Selection View Go Run Terminal Help
hosp_readmissions
EXPLORER
HOSP_READMISSIONS
venv
app.py
feature_names.pkl
hosp_readmissions.ipynb
hospital_readmissions.csv
model.pkl
requirements.txt
scaler.pkl
hosp_readmissions.ipynb
app.py
app.py > ...
glucose_map = {"no": 0, "normal": 1, "high": 2}
A1C_map = {"no": 0, "normal": 1, "high": 2}
# Base input dict
input_dict = {
'age': le_age[age],
'time_in_hospital': time_in_hospital,
'n_lab_procedures': n_lab_procedures,
'n_procedures': n_procedures,
'n_medications': n_medications,
'n_outpatient': n_outpatient,
'n_inpatient': n_inpatient,
'n_emergency': n_emergency,
'change': change_map[change],
'diabetes_med': diabetes_map[diabetes_med],
'glucose_test': glucose_map[glucose_test],
'A1Ctest': A1C_map[A1Ctest]
}
# Create full feature frame with correct structure
input_df = pd.DataFrame([input_dict])
input_df = input_df.reindex(columns=feature_names, fill_value=0)
# Scale and predict
input_scaled = scaler.transform(input_df)
prediction = model.predict(input_scaled)[0]
probability = model.predict_proba(input_scaled)[0][1]
# Output
if st.button("Predict"):
st.subheader("🔍 Prediction Result")
if prediction == 1:
st.error(f"❌ Patient is likely to be readmitted. (Probability: {probability:.2f})")
else:
st.success(f"✅ Patient is not likely to be readmitted. (Probability: {1 - probability:.2f})")
OUTLINE
TIMELINE
Ln 67, Col 1 Spaces: 4 UTF-8 CRLF Python 3.13.1 (venv) Go Live
33°C
Mostly cloudy
ENG
IN
87%
14:55
05-07-2025





9. Conclusion and Future Enhancements

Conclusion

This project demonstrates the practical application of machine learning in solving a real-world healthcare problem: predicting patient readmissions. Using structured data from 25,000 patient records, a complete machine learning pipeline was designed, implemented, and deployed — enabling hospitals to forecast which patients are likely to be readmitted soon after discharge.

Key accomplishments of the project include:

- Comprehensive data preprocessing and feature engineering
- Exploratory analysis revealing important patterns in patient behavior
- Training and evaluation of models with Random Forest achieving ~87% accuracy
- Visualization using PCA and clustering with KMeans
- Development of an interactive Streamlit app for real-time predictions

By accurately identifying at-risk patients, the system can support hospitals in reducing costs, improving resource allocation, and enhancing patient care quality.

Future Enhancements

To expand and improve this project, the following enhancements are suggested:

1. **Integration with Real Hospital Data**
    - Collaborate with hospitals to use Electronic Medical Records (EMR)

- Implement privacy-preserving techniques like anonymization and encryption

2. **Explainability Tools**
    - Integrate SHAP (SHapley Additive exPlanations) to explain model decisions
    - Help healthcare providers understand "why" a patient is predicted to be at risk
3. **Cloud Deployment**
    - Deploy the app to **Streamlit Cloud**, **Heroku**, or **AWS EC2**
    - Make the tool accessible beyond the local machine
4. **Alert System Integration**
    - Add functionality to trigger alerts or recommendations based on risk score
    - Useful for clinical workflow automation
5. **Time-Based Predictions**
    - Extend model to predict **when** a patient might be readmitted (e.g., within 30/60/90 days)
6. **Mobile-Friendly UI**
    - Adapt Streamlit frontend for tablets or mobile devices used by hospital staff

With these enhancements, the project could evolve into a production-ready healthcare analytics tool that contributes to **smarter, data-driven decision-making** in hospitals.