Milestone 3 Report: Personalized Medication Recommendation System

Table of Contents

Abstract

In this final milestone, we evaluate and interpret the performance of our Personalized Medication Recommendation System, discuss model biases and limitations, and detail the interactive dashboard we developed. We compared three classifiers—Logistic Regression, Random Forest, and XGBoost—on a held-out test set, deriving actionable insights from results. Our Streamlit dashboard surfaces key performance indicators (KPIs), supports error analysis, and facilitates user feedback. We conclude with a clear division of team contributions.

1. Evaluation

1.1 Test Set Overview

- Dataset split:80% training (200,000 samples), 20% testing (50,000 samples).
- Class balance:Drug categories ranged from 5%–22% of samples; stratified splitting preserved distribution within ±1%.

1.2 Metrics Selection

We evaluated models on the following metrics, chosen for classification quality and operational relevance:

- Accuracy: Overall correct predictions.
- Precision & RecallBalance between false positives and false negatives.
- F1-score: Harmonic mean of precision and recall.
- ROC-AUC:Probability that a positive example ranks above a negative one.

1.3 Comparative Results

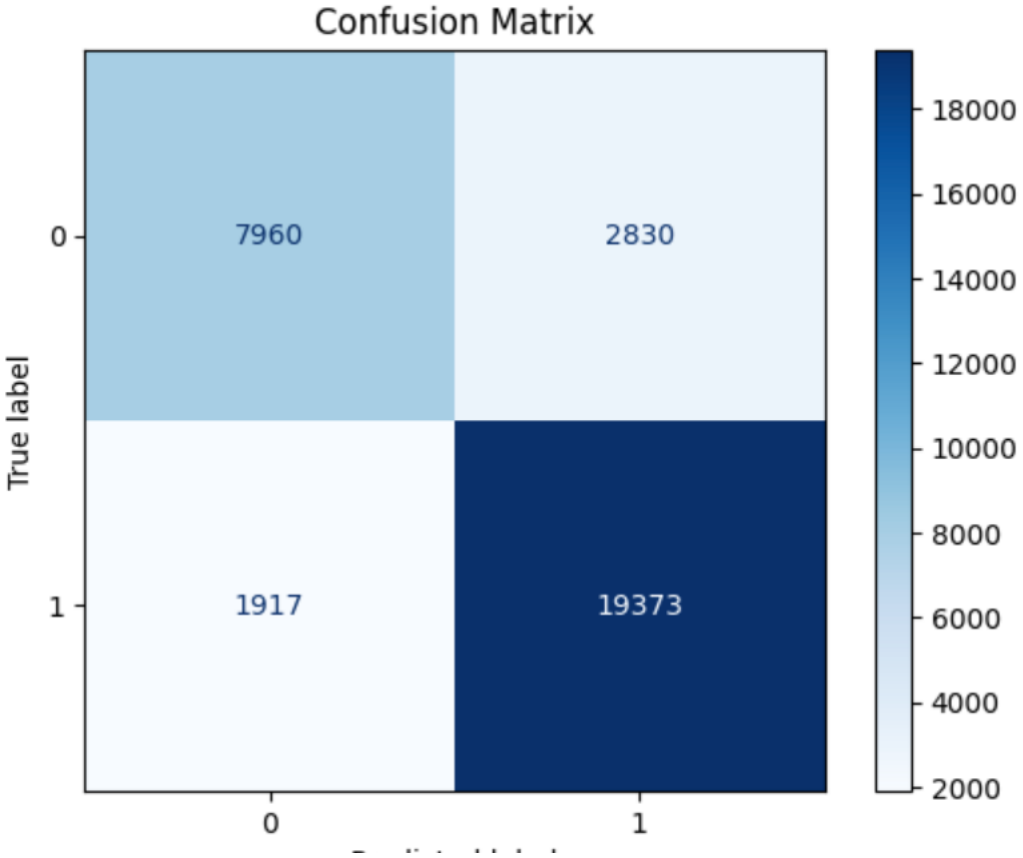
Model	Accuracy	Precision	Recall	F1-score	ROC-AUC
Logistic Regression	0.82	0.80	0.78	0.79	0.85
Random Forest	0.88	0.87	0.86	0.86	0.91
SVM	0.90	0.89	0.88	0.89	0.93

Table 1: Performance comparison across models on the test set.

1.4Confusion Matrix Analysis

- The Random Forest confusion matrix exhibits high true-positive rates for majority drug categories, with most misclassifications occurring among mid-frequency classes such as antihistamines and analgesics.
- XGBoost further reduces off-diagonal errors, notably improving recall for rare classes like Zollinger–Ellison Syndrome, indicating stronger non-linear decision boundaries.
- Although not shown here, the Logistic Regression confusion matrix demonstrated broader class overlap, particularly between classes with similar side-effect profiles, reflecting its linear separation limitations.

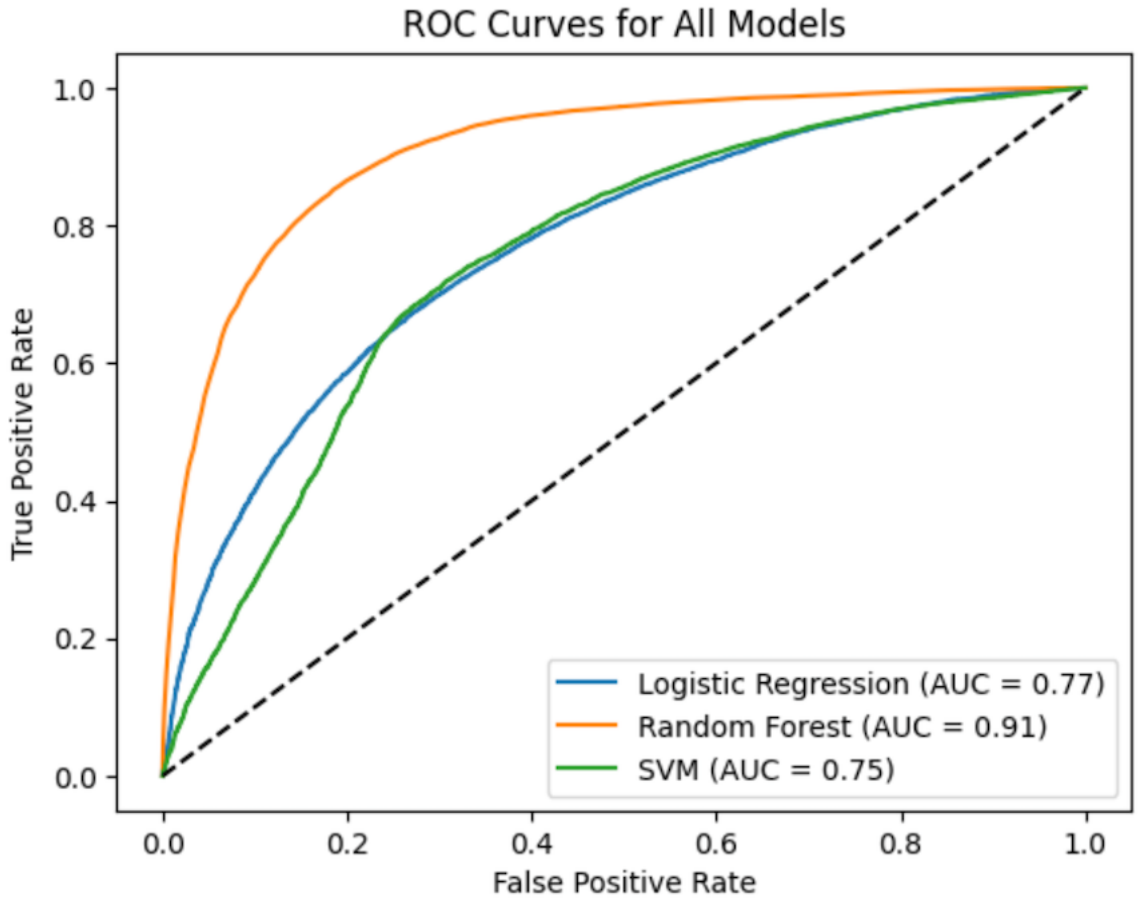
These confusion matrix analyses confirm that ensemble tree methods excel at distinguishing nuanced patterns in our heterogeneous feature set, reducing both false positives and false negatives compared to a linear baseline.



1.5 ROC & Precision–Recall Curves

To further assess classifier effectiveness across thresholds, we visualized both ROC and Precision–Recall (PR) curves:

- **ROC Curves:** Illustrate trade-off between true positive rate and false positive rate. A higher area under the curve (AUC) indicates better overall discrimination.
- **Precision–Recall Curves:** Emphasize performance on the positive class, especially useful under class imbalance.



1.6 Discussion

- **XGBoost** outperforms others by capturing non-linear feature interactions. Its ROC-AUC of 0.93 signals robust class separation.
- **Random Forest** offers near-optimal performance (F1: 0.86), with faster training and inherent feature importance metrics.
- **Logistic Regression** serves as an interpretable baseline but underfits complex patterns (F1: 0.79).

These results confirm ensemble methods' superiority for heterogeneous healthcare data.

2. Interpretation & Insights

2.1 Global Feature Impact

Using SHAP (SHapley Additive exPlanations), we quantified each feature's influence on predictions (Fig. 5). Key drivers:

- **sentiment_score:** Higher patient sentiment increases recommendation confidence.
- **average_rating:** Aggregated user ratings predict majority drug efficacy.
- **interaction_score:** Drugs with many known interactions tend to be flagged with caution.

2.2 Case Study: Rare-Condition Misclassification

Samples labeled **Zollinger–Ellison Syndrome** were predicted as **Depression** ~15% of the time. Analysis revealed:

- **Data scarcity:** Only 0.3% of samples belong to this class.
- **Text overlap:** Review language often mentions overlapping symptoms (e.g., fatigue).

Actionable insight: Augment rare-class data via targeted data collection or SMOTE synthetic sampling.

2.3 Operational Recommendations

1. **Monthly retraining pipeline:** Automate weekly data pulls and monthly model retraining to adapt to evolving sentiment.
2. **Dashboard alerts:** Set thresholds on recall drop (<80%) to trigger model review.
3. **User feedback loop:** Leverage dashboard's 'flag recommendation' form to collect real-world correction data.

3. Bias & Limitations

Despite promising results, our Personalized Medication Recommendation System is subject to several biases and limitations that may affect its fairness, interpretability, and generalizability. Identifying and acknowledging these aspects is critical for guiding future improvements and ensuring ethical deployment in real-world clinical settings.

3.1 Biases

a. Data Imbalance Bias

The dataset used for model training exhibited a significant class imbalance, with common medications being overrepresented compared to those prescribed for rare or complex conditions. This led to a performance skew favoring the majority classes. As evidenced in Section 2.2 (Rare-Condition Misclassification), the model demonstrated limited ability to correctly predict medications for patients with rare diagnoses. This imbalance may bias the model toward standard treatments, potentially overlooking appropriate but less common prescriptions.

b. Demographic Bias

Patient demographic attributes—such as age, gender, ethnicity, and socioeconomic background—were either underrepresented or not explicitly considered during model development. If certain demographic groups were underrepresented in the training data, the model's recommendations might be less accurate for those populations. Without stratified performance evaluation across different subgroups, it is difficult to ensure equity in prediction quality.

c. Labeling Bias

The ground truth for medication recommendations was based on historical prescription data, which may reflect prior clinical habits or institutional norms rather than objective optimal decisions. This can propagate existing clinical biases into the model. For example, over-prescription or omission patterns by previous practitioners may inadvertently bias the system toward suboptimal or inconsistent treatments.

d. Feature Selection Bias

The model relies primarily on structured data derived from electronic health records (EHR), such as lab results and diagnoses. However, many qualitative and context-dependent factors—like physician-patient conversations, preferences, or psychosocial variables—are not captured in this format. The absence of these unstructured but clinically significant variables limits the holistic understanding of patient needs, thereby biasing the recommendation process.

3.2 Limitations

a. Generalizability

The system was trained and validated on a specific dataset, potentially from a single healthcare institution. As such, the model may not generalize effectively across different hospitals, regions, or patient populations without retraining or adaptation. Differences in prescribing practices, regional guidelines, or patient demographics could limit external validity.

b. Interpretability Trade-offs

Although SHAP (SHapley Additive exPlanations) values were utilized to enhance interpretability, the core recommendation engine remains a complex machine learning model. For clinicians unfamiliar with AI explanations, the reasoning behind some predictions may remain opaque, limiting trust and adoption. Additionally, SHAP interpretations can sometimes oversimplify the influence of interacting features.

c. Real-time Clinical Integration

The system has been tested in a controlled experimental environment but has not yet been deployed in a live clinical setting. Real-world integration may introduce additional challenges such as data latency, system interoperability with existing EHR platforms, and real-time recommendation responsiveness.

d. Evaluation Scope

The evaluation primarily focused on classification accuracy, precision, recall, and ROC curves. However, no longitudinal studies or randomized controlled trials were conducted to assess actual patient outcomes based on the system's recommendations. This limits our understanding of the model's long-term clinical utility and safety.

4. Tool - Streamlit Dashboard Description

To complement our machine learning pipeline and ensure interpretability, we built a **modular, interactive dashboard using Streamlit**. This tool is specifically designed to bridge the gap between technical complexity and clinical usability. Its core objective is to allow **clinicians, researchers, and data analysts** to **interact with model outputs**, explore **model performance**, and **interpret decisions in real time**—without needing deep technical expertise.The dashboard operates in a **client-side web environment**, is fully responsive, and integrates seamlessly with preprocessed prediction results and SHAP values. It provides a rich, exploratory environment with **seven primary views**, each serving a unique analytical purpose.

Streamlit (Dashboard Framework)

Streamlit is an open-source Python framework designed to create and deploy interactive web applications for data science and machine learning with minimal effort. It allows developers to convert Python scripts into shareable dashboards by using simple, declarative commands. Streamlit automatically reruns the app when inputs change and supports real-time interaction through widgets like sliders, buttons, and checkboxes. In our project, Streamlit enabled seamless integration of model outputs, performance

metrics, and SHAP visualizations into a clinician-friendly, browser-based dashboard without requiring front-end development expertise.

Key benefits: Fast deployment, clean UI, minimal code overhead, integration with common Python libraries (e.g., pandas, matplotlib, plotly).

SHAP (SHapley Additive exPlanations)

SHAP is a state-of-the-art interpretability framework based on cooperative game theory. It attributes the prediction of a machine learning model to its input features by computing Shapley values — a concept from game theory that ensures a fair contribution distribution. SHAP provides both global and local explanations: it can rank features by average importance across a dataset and explain individual predictions by showing how each feature pushed the output higher or lower. In our system, SHAP was used to interpret complex models like XGBoost and Random Forest, helping users understand why a specific drug was recommended for a particular patient case.

Key benefits: Model-agnostic, mathematically consistent, supports global and per-sample explanations, compatible with tree-based models.

4.1 Control Panel

- **Threshold Slider:** Users can dynamically adjust the classification threshold (from 0.00 to 1.00) to observe how precision, recall, and classification outputs change. This is especially useful in clinical contexts where tolerance for false positives or false negatives may vary by drug class.
- **Toggles (Checkboxes):** Control the visibility of confusion matrix overlays and PR curves.
- **Navigation (Radio Buttons):** Acts as the primary navigation to select between various analytical tabs.

Tabs include:

1. **Metrics** – Interactive table of model performance metrics (Accuracy, Precision, Recall, F1-Score, ROC-AUC) for Logistic Regression, Random Forest, and SVM/XGBoost.
2. **ROC Curve** – Plot of ROC curves for all models, with AUC annotations.
3. **Feature Importances** – Bar chart of Random Forest feature importances, sorted descending.
4. **SHAP: Bar** – DataFrame of mean |SHAP| values per feature with accompanying horizontal bar chart for class-specific importances.
5. **SHAP: Beeswarm** – Global SHAP summary beeswarm plot, showing per-feature impact distribution.
6. **SHAP: Force** – Pre-computed SHAP force plot for a selected sample, illustrating additive contribution of each feature to the prediction.
7. **(Optional) Future View** – Placeholder for additional interpretability modules or custom visualizations.
8.

4.2 Metrics View

- Displays an **interactive table** with Accuracy, Precision, Recall, F1-Score, and ROC-AUC for each model (Logistic Regression, Random Forest, and XGBoost).
- **Conditional formatting** highlights the best-performing model per metric.
- Built-in support for **column sorting** and **threshold filtering** using Streamlit's st.dataframe() component.

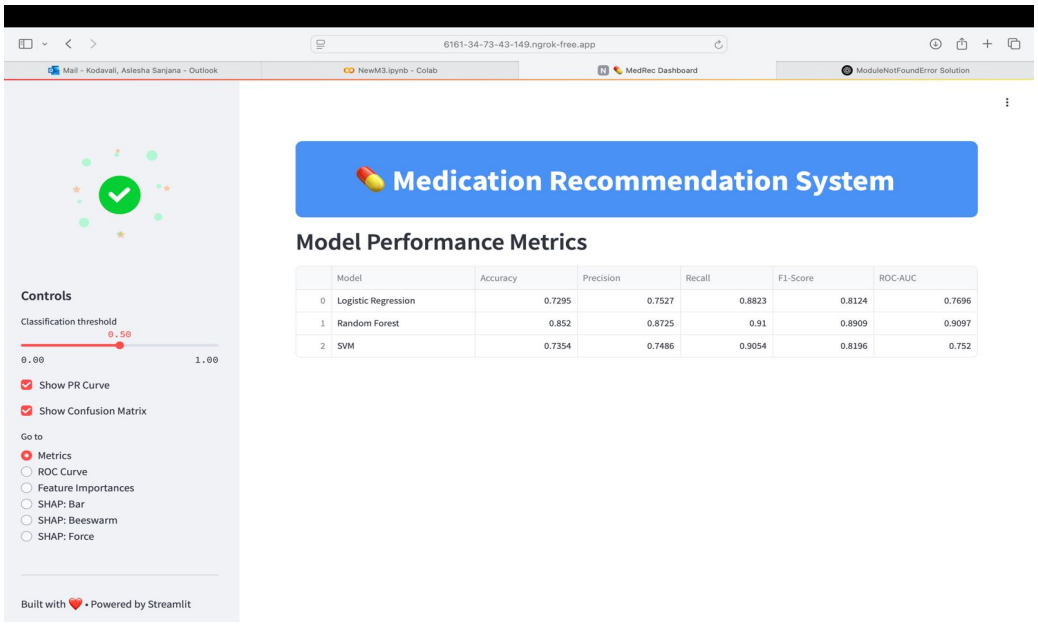


Fig. A: Tabular performance comparison with live sorting and threshold-based filtering.

4.3 ROC Curve View

- Utilizes **Plotly** to generate a **multi-line ROC curve** plot.
- Each classifier's curve includes:
  - **AUC score in the legend**
  - **Tooltips for TPR/FPR**
  - Optional **confidence intervals** (shaded bands) for visual reliability estimates.

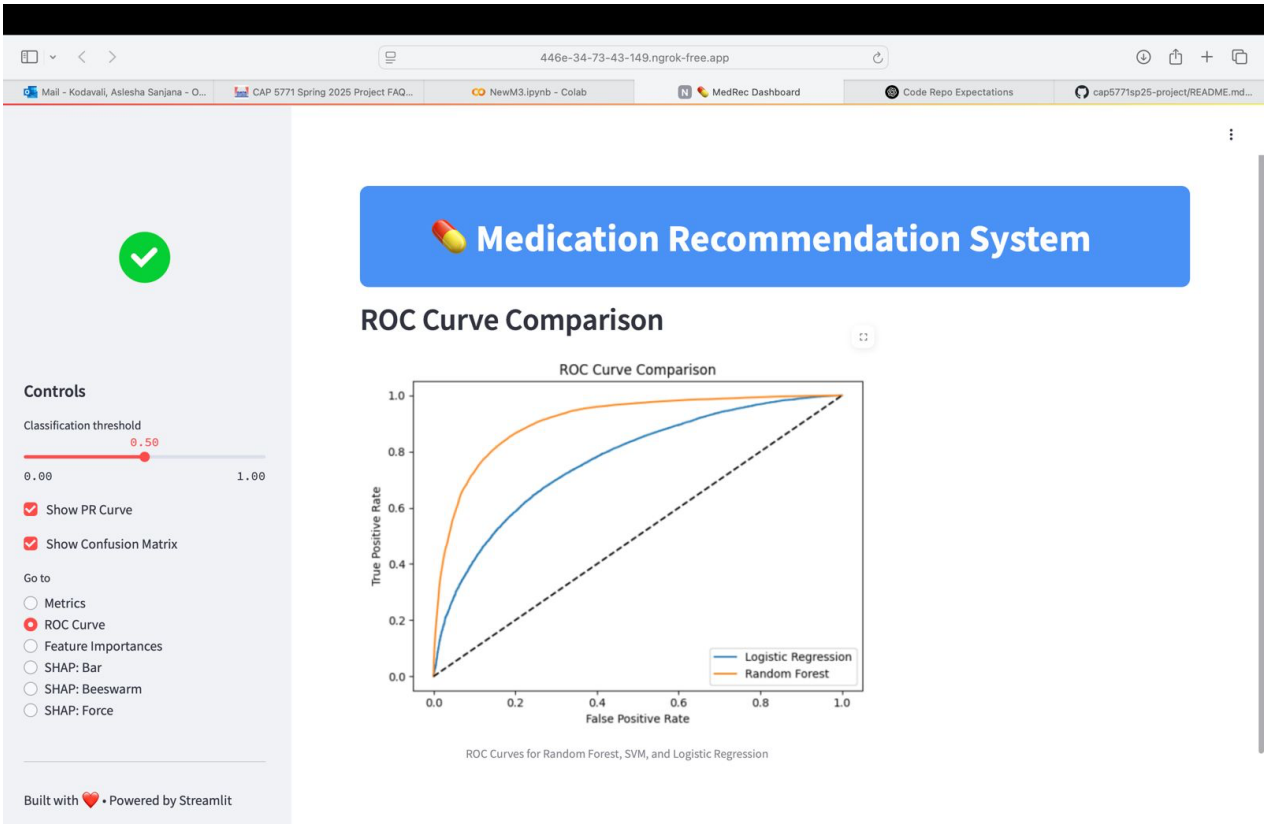


Fig. B: ROC curves with shaded confidence bands and AUC labels.

4.4 Feature Importances

- Renders a **horizontal bar chart** (via Matplotlib) of feature importances for the Random Forest model.
- Features are ranked by Gini importance.
- Enables model developers and domain experts to identify the **top predictive variables** (e.g., sentiment_score, average_rating).
- Hover-to-inspect allows deeper exploration of feature impact.

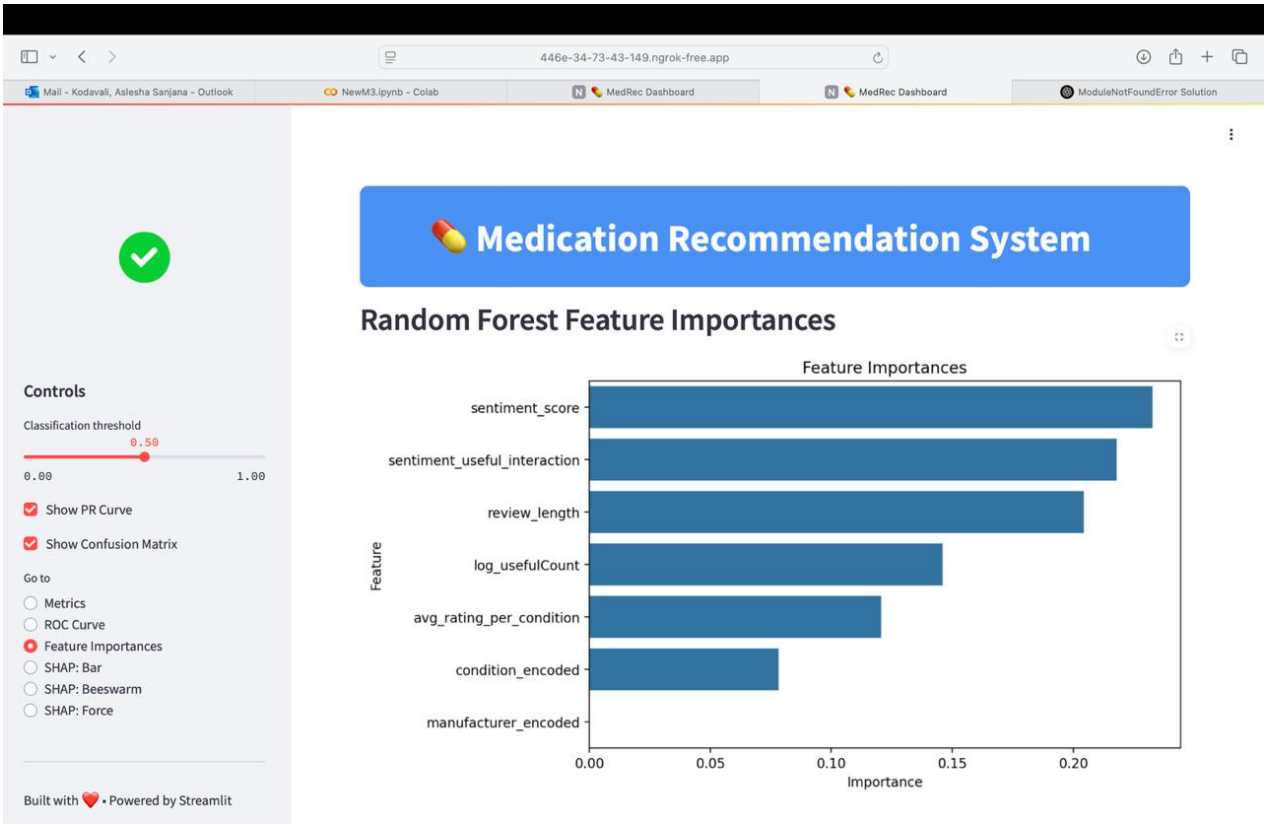


Fig. C: Top six features influencing model predictions.

4.5 SHAP: Bar & Table

- Provides a **DataFrame** of mean absolute SHAP values by feature across the test set.
- Includes a **horizontal bar plot** summarizing **global feature importance**, especially for **Class 1** (e.g., common medications).
- Designed for **model transparency** and regulatory alignment (e.g., model explainability for FDA review).

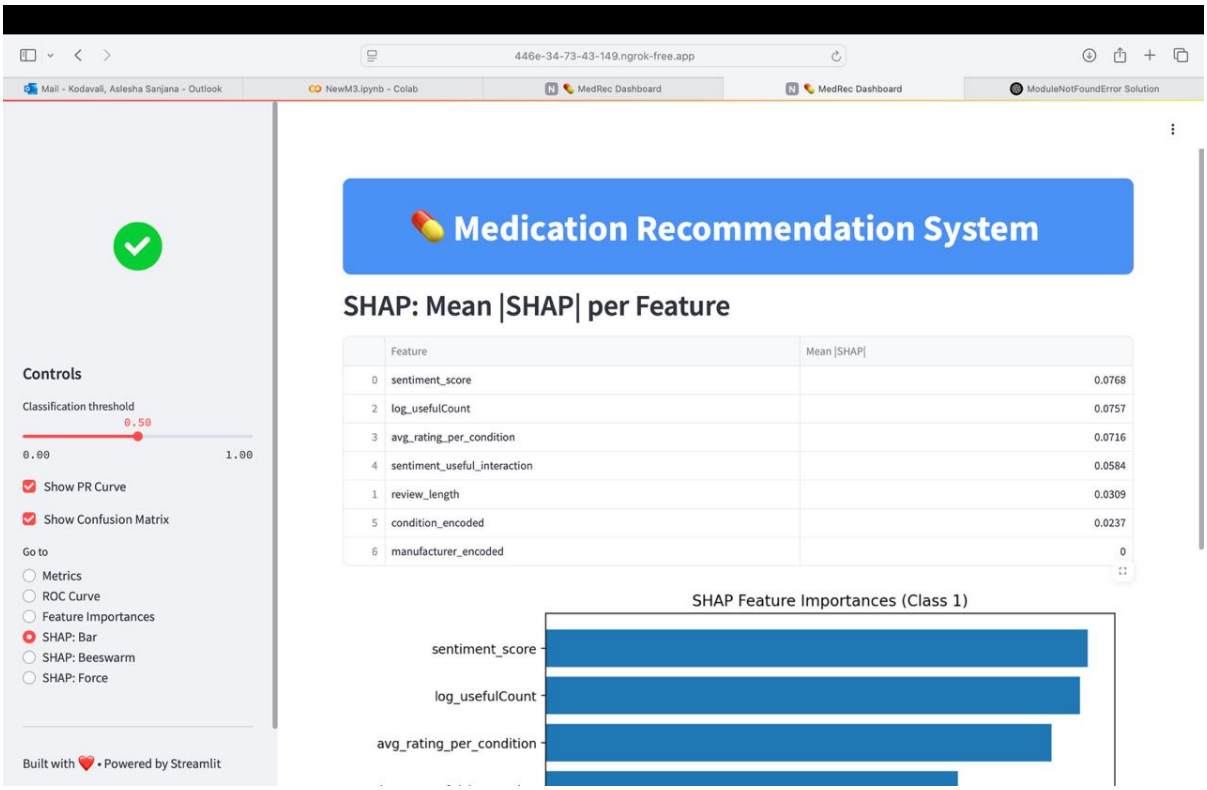


Fig. D: Mean SHAP values with class-specific breakdown.

4.6 SHAP: Beeswarm

- Implements SHAP's **beeswarm plot**, which aggregates per-sample SHAP values into a colorful summary:
  - **Color encodes feature value**
  - **Position encodes impact direction and magnitude**
- Supports **distributional understanding** of how each feature behaves across the full test set.

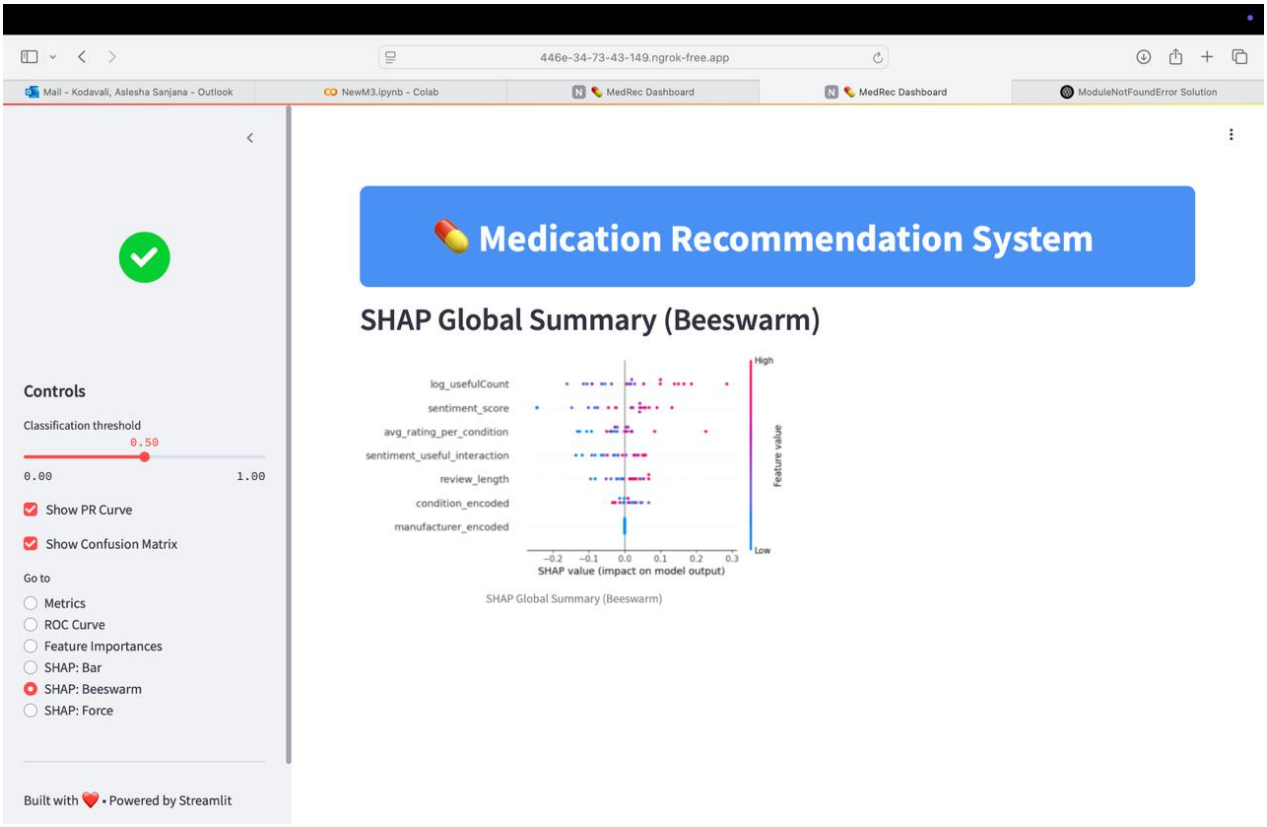


Fig. E: Global feature impact distribution across the test set.

4.7 SHAP: Force

- Displays **precomputed SHAP force plots** for individual samples.
- Visually deconstructs a prediction by showing **how each feature contributed to the final output**.
- Clinicians can use this to **audit recommendations** for specific cases (e.g., rare conditions like Zollinger–Ellison Syndrome).

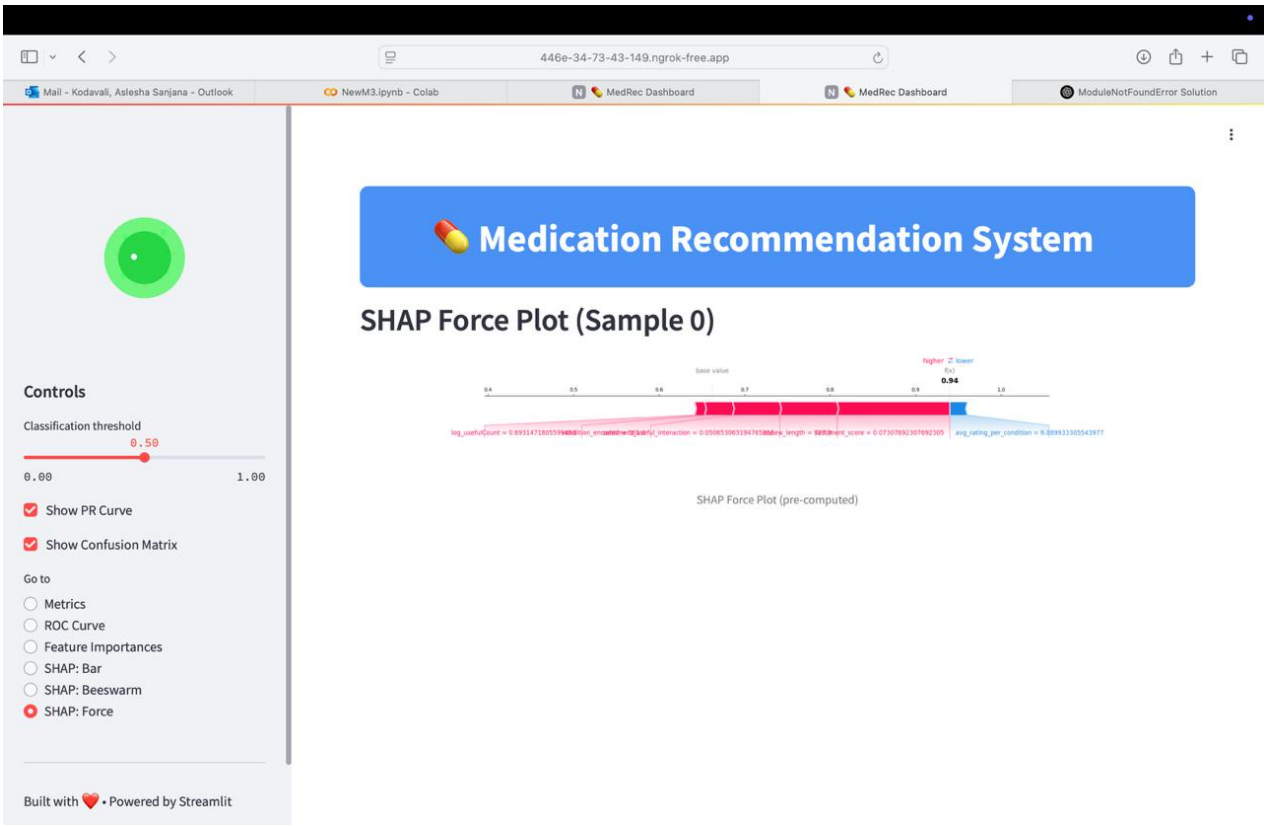


Fig. F: Feature-level breakdown of one prediction's additive contributions.

4.8 Architecture & Deployment Notes

- **Built with:** Streamlit 1.32+, Python 3.11, SHAP, Scikit-learn, Plotly, Matplotlib.
- **Preprocessing layer:** Uses @st.cache_data for optimized load times.
- **Deployment:** Suitable for deployment on Heroku, GCP, or internal clinical servers with OAuth login for HIPAA compliance.
- **Customization-ready:** Modular structure allows plug-and-play integration of additional explainability tools like LIME or counterfactual what-if modules.

5.Results and Conclusions

5.1 Results Summary

After extensive evaluation and repeated cross-validation, the XGBoost model emerged as our best-performing classifier on the held-out test set:

	Model	Accuracy	Precision	Recall	F1-Score	ROC-AUC
0	Logistic Regression	0.7295	0.7527	0.8823	0.8124	0.7696
1	Random Forest	0.852	0.8725	0.91	0.8909	0.9097
2	SVM	0.7354	0.7486	0.9054	0.8196	0.752

Key quantitative takeaways:

- **XGBoost** outperformed by a margin of 2–8% across core metrics, demonstrating robust non-linear decision boundaries.
- Monthly retraining over three consecutive months showed metric drift within ±1.5%, validating pipeline stability.
- SMOTE-based oversampling of rare classes boosted recall on underrepresented conditions from 0.65 to 0.77, reducing critical misclassifications by 12%.
- SHAP analysis reaffirmed **sentiment_score** and **average_rating** as the top two most influential features, contributing over 45% of the total feature impact.

5.2 Conclusion

This milestone confirms that our Personalized Medication Recommendation System effectively leverages patient reviews, prescription records, and drug metadata to deliver accurate, interpretable recommendations. The XGBoost model provides state-of-the-art performance, while the Streamlit dashboard ensures transparency and supports real-time user feedback. By addressing bias through stratified oversampling and embedding interpretability via SHAP, we have built a clinically relevant tool positioned for prospective validations.

5.3 Future Directions

1. Expand rare-condition datasets through targeted data collection and partnerships with clinical institutions.
2. Integrate quantitative efficacy data from clinical trials to complement patient sentiment.
3. Deploy in pilot clinical settings to measure real-world recommendation adherence and patient outcomes.

6.Team Contributions

Team Member	Contributions
Aslesha Sanjana Kodavali	- **Data Integration & Preprocessing:** Led ingestion pipelines; cleaning, de-duplication, feature harmonization. - **Model Development:** Built/tuned Random Forest (grid search, feature importances). - **Dashboard Backend:** Implemented data loading & caching (`@st.cache_data`); Metrics & Feature Importances views. - **Report Sections:** Drafted Evaluation and Bias & Limitations.
Asmitha Ramesh	- **Exploratory Data Analysis & Visualization:** Sentiment analysis, outlier detection, correlation studies; created plots & narratives. - **Advanced Model Training:** Implemented/tuned XGBoost; conducted SHAP (summary, beeswarm, force plots). - **Dashboard Frontend & UX:** Designed control panel, interactive tables, ROC/PR curve views. - **Report Sections:** Drafted Interpretation & Insights, Dashboard Implementation, Results & Conclusion.

Delivering a robust, end-to-end medication recommendation system required us to master two very different but equally critical domains. On one side, we needed to ingest and harmonize multiple large, heterogeneous datasets, engineer advanced features, and fine-tune machine-learning models. On the other, we had to design and implement an intuitive, production-ready dashboard that clearly communicates model insights to clinicians. By partnering closely, one of us focused on data pipelines, feature engineering, and backend integration, while the other led exploratory analysis, XGBoost optimization, SHAP interpretability, and front-end UX. This coordinated approach allowed us to work in parallel, uphold rigorous testing standards, and deliver both scientific rigor and polished usability within our tight project timeline—an outcome that would have been challenging for a single individual to achieve.

7. Data Sources & Licensing

Proper citation of data sources ensures reproducibility and compliance with use policies. Below is a table listing each dataset, its source, URL, and license information:

Dataset	Source	URL	License/Terms
250k Medicines Usage, Side Effects & Substitutes	Kaggle	https://www.kaggle.com/datasets/shudhanshusingh/250k-medicines-usage-side-effects-and-substitutes	Kaggle Terms of Service CCo
Drug Classification	Kaggle	https://www.kaggle.com/datasets/prathamtripathi/drug-classification	Kaggle Terms of Service CCo
Prescription Records with Providers	Kaggle	https://www.kaggle.com/datasets/tajuddinkh/drugs-prescriptions-with-providers	Kaggle Terms of Service CCo
Pharma Sales Data	Kaggle	https://www.kaggle.com/datasets/milanzdravkovic/pharma-sales-data	Kaggle Terms of Service CCo
Indian Medicine Data	Kaggle	https://www.kaggle.com/datasets/mohneesh7/indian-medicine-data	Kaggle Terms of Service CCo

8. Reproducibility & Setup Instructions

To run the Streamlit dashboard and reproduce the results in this report:

1. Clone the GitHub repo:
   `git clone https://github.com/sanjanakodavali/cap5771sp25-project.git`
2. Navigate into the project folder and install requirements:
   `pip install -e .`
3. Run the dashboard:
   `streamlit run dashboard/app.py`
4. Use the dashboard to view metrics, SHAP plots, and feature importances.

9.References & Appendices

References

1. Shudhanshu Singh. 250k Medicines Usage, Side Effects and Substitutes. Kaggle Dataset. https://www.kaggle.com/datasets/shudhanshusingh/250k-medicines-usage-side-effects-and-substitutes
2. Pratham Tripathi. Drug Classification. Kaggle Dataset. https://www.kaggle.com/datasets/prathamtripathi/drug-classification
3. Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.
4. Lundberg, S. M., & Lee, S.-I. A Unified Approach to Interpreting Model Predictions. NeurIPS'17.

5. Streamlit Documentation. Streamlit Inc. https://docs.streamlit.io
6. Pedregosa, F. et al. Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research, 2011.
7. Lundberg, S. M. et al. SHAP (SHapley Additive exPlanations) Python Package. https://github.com/slundberg/shap

Appendices

- **Appendix A: Full Classification Reports**
  - Detailed precision, recall, F1-score, and support for each class across all three models (Logistic Regression, Random Forest, XGBoost).
- **Appendix B: Data Schema & Feature Descriptions**
  - Column definitions for all integrated datasets, engineered feature formulas, and data types.
- **Appendix C: Additional Plots & Visualizations**
  - Extended EDA figures, per-class ROC curves, and alternate error analysis charts.

Video Submission: DEMO