Group#4

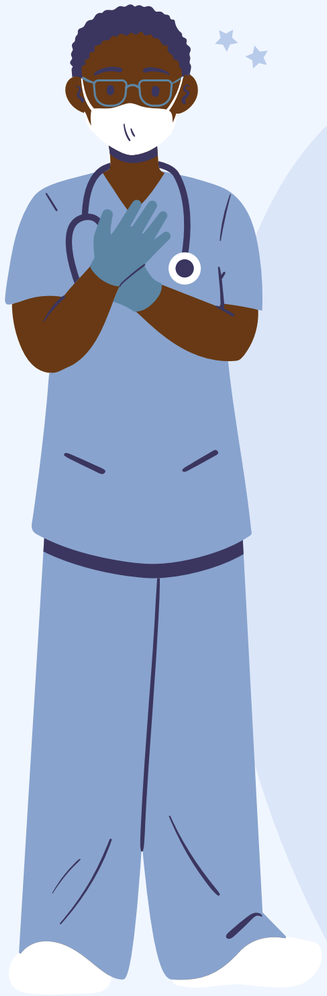
Salama

- Shahad Alsabaie - ID: 444200061
- Sarah Alomran - ID: 444200911
- Reem Al Mutlaq - ID: 444200533
- Shadn Alsaif - ID: 443201150
- Mashael Albugami - ID: 44202218

Outline

- Introduction
- Methodology & Data
- Phase 1 key findings and steps
- Phase 2 key findings and steps
- Phase 3 key findings and steps
- Phase 4 key findings and steps
- Conclusion & Recommendations
- Lessons

Introduction



Problem Statement

Develop an intelligent medical advice system that can predict diseases based on patient-reported symptoms and provide actionable healthcare recommendations.

Business Value:

- Early disease detection
- Reducing diagnostic time
- Improving patient outcomes
- Optimizing healthcare resource allocation

Methodology & Data

Data Overview

The full dataset can be viewed here.

Dataset.

Dimensions:
- Number of patients (rows): 246945
- Number of symptoms (columns): 378

symptoms represented as binary values (O=absent, 1=present)

Process/Approach

Data Cleaning → Handle Missing Values
Remove Duplicates

Supervised Technique → Random Forest
SVM

Unsupervised Technique → K-Means Clustering
Hierarchical Clustering
DBSCAN

GAI → OpenAI GPT

Tools Used

Software & Platform

jupyter notebook, GitHub

Languages

Python

Libraries

Data Handling & Manipulation	→	pandas, numpy, joblib
Visualization	→	matplotlib.pyplot, seaborn
Data Handling & Manipulation	→	pandas, numpy, joblib
Model Evaluation	→	accuracy, precision, recall, f1_score

Phase 1: Steps & Key Findings

Steps

- Finding a proper sized dataset that suits our purpose, dataset source: Kaggle Diseases and Symptoms Dataset
- Specifying the target column in the dataset (Diseases) with the rest 377 columns representing symptoms
- analyzing the diseases and symptoms distributions, missing values, duplicates, and the data types
- Preprocessing of the data by removing duplicate rows, filling out missing symptoms with O, filtering out disease with cases less than 10,and removing symptoms with O occurrences for all diseases
- Representation and analysis of the Imbalance in the dataset

Phase 1: Steps & Key Findings

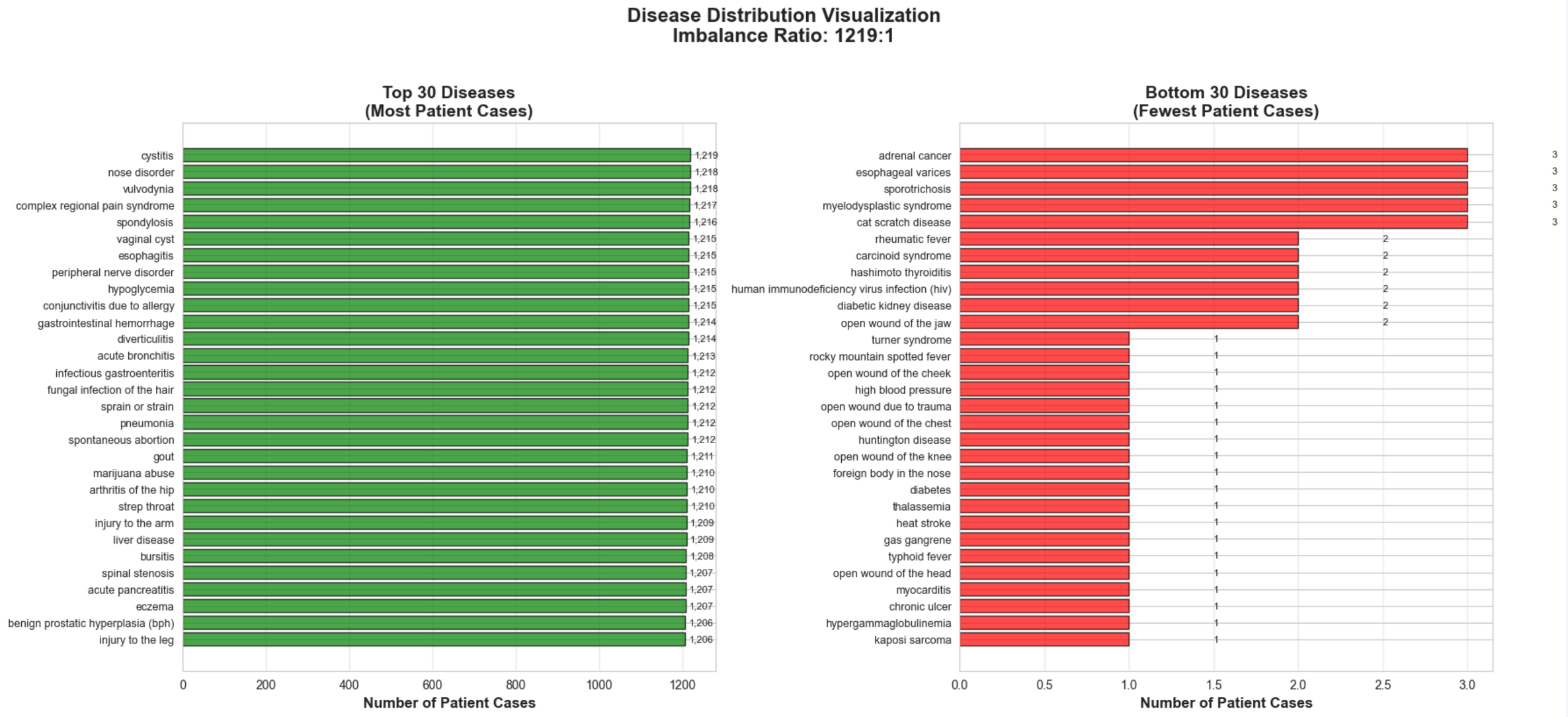
Key findings

- The dataset has 246,945 patient disease cases represented with their symptoms in binary form, after pre processing the dataset the cases have become 188,920
- The dataset shows a significant imbalance in terms of diseases distribution and symptoms distribution

- The imbalance in diseases has been solved by suspending diseases that has cases less than 10 (cases should be >=10)
- The imbalance in symptoms was considered beneficial since rare occurring symptoms lead to specific diseases and reflects medical cases in real-world but symptoms with 0 occurrence has been suspended since they provide no benefit

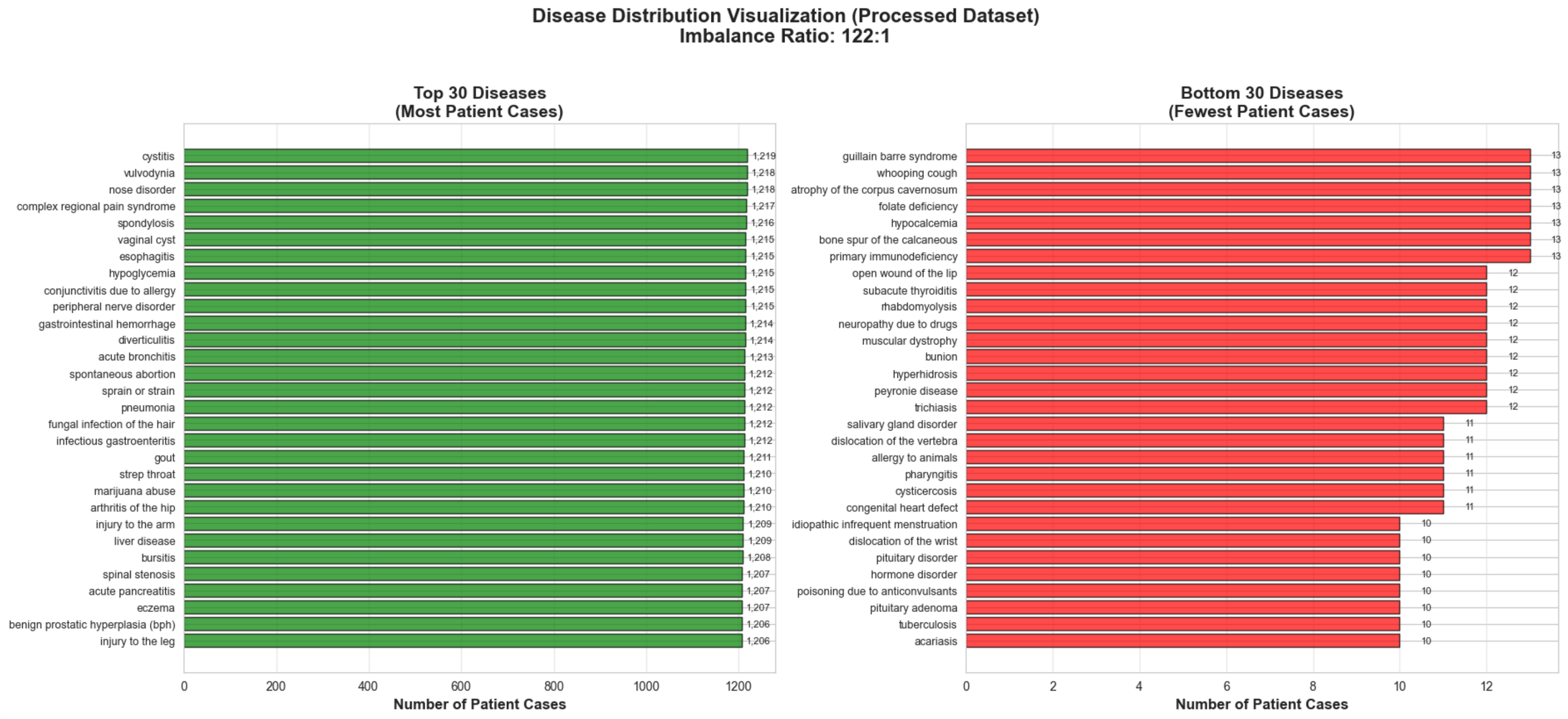
Phase 1

Diseases visualization - Before preprocessing



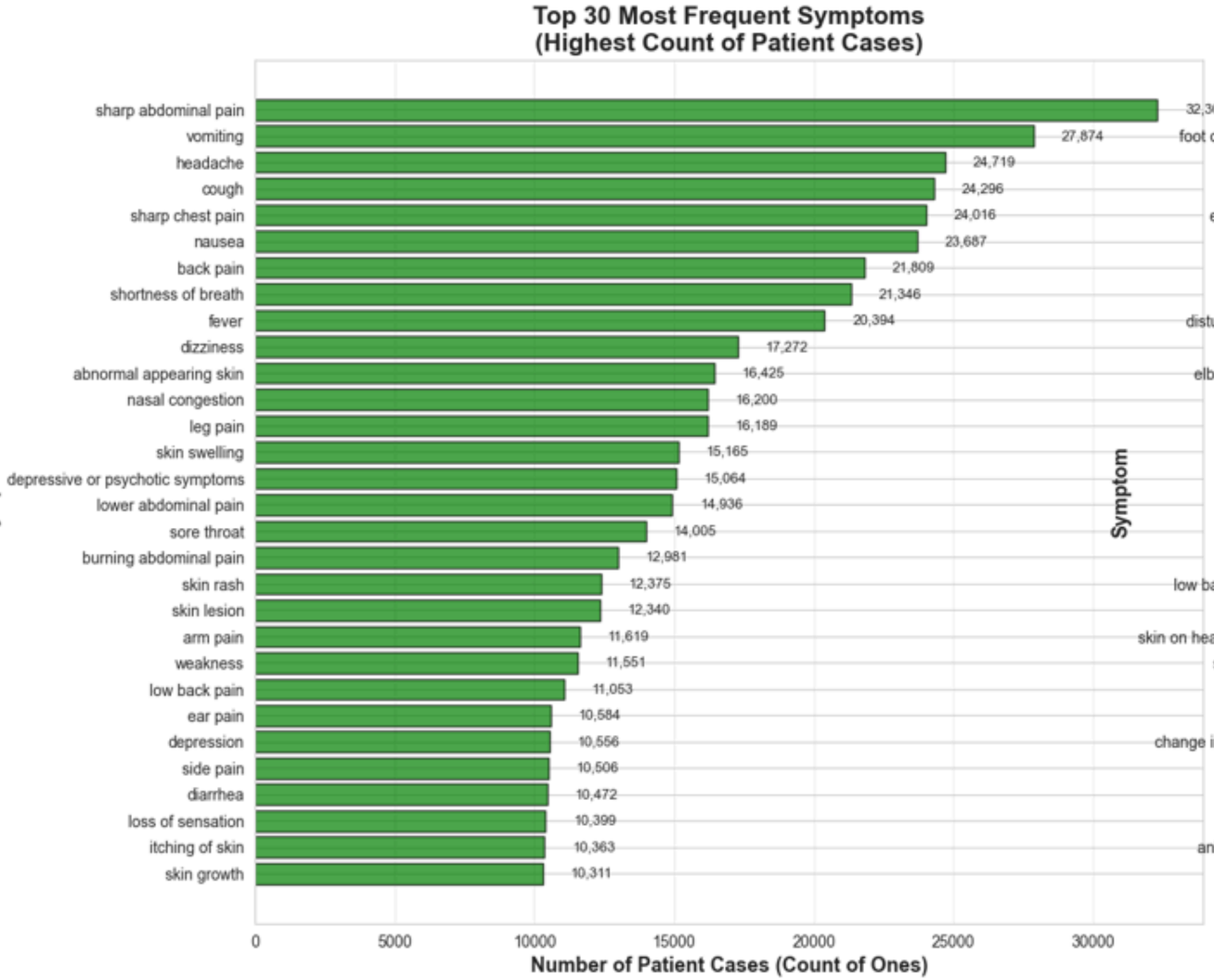
Phase 1

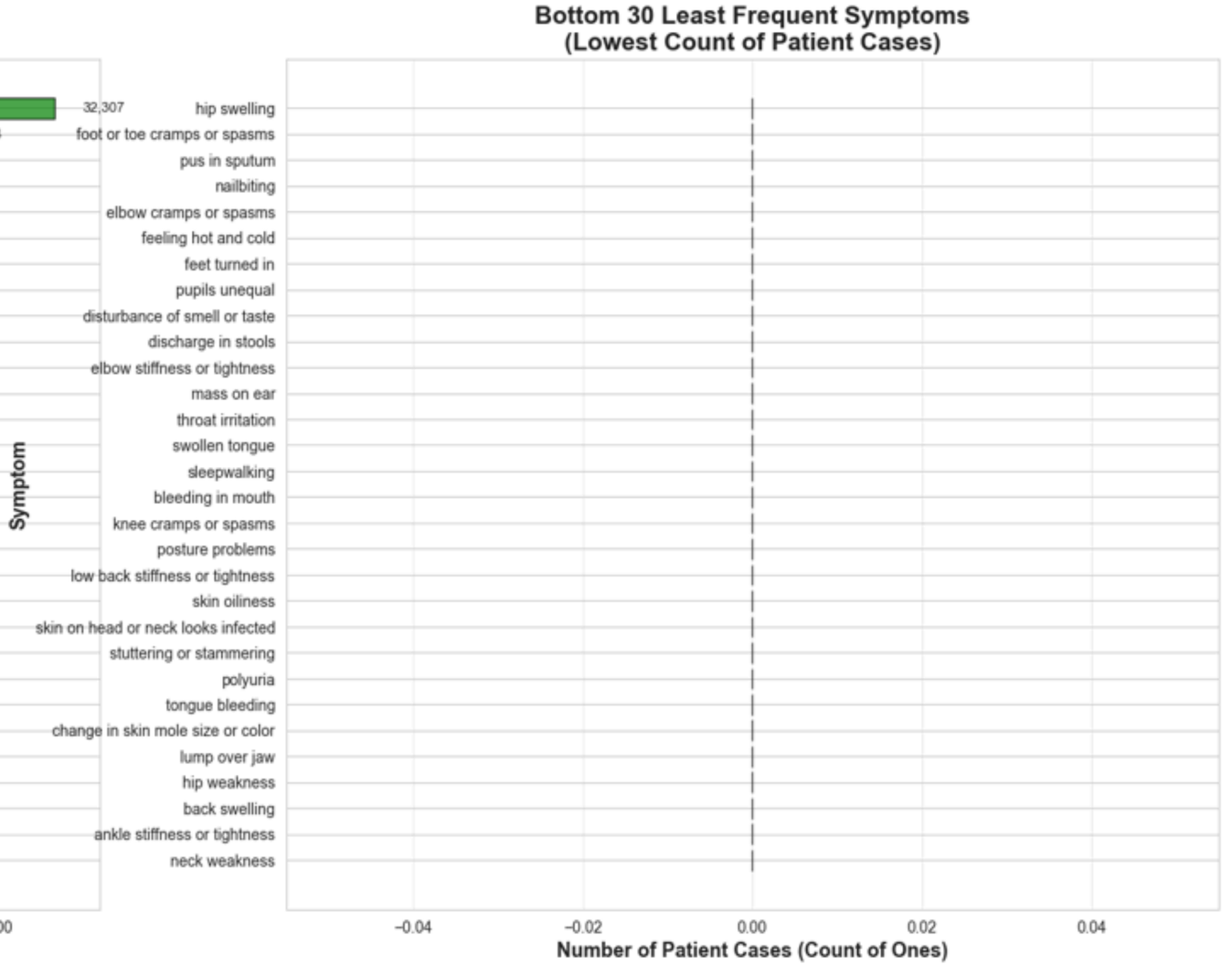
Diseases visualization - After preprocessing



Phase 1

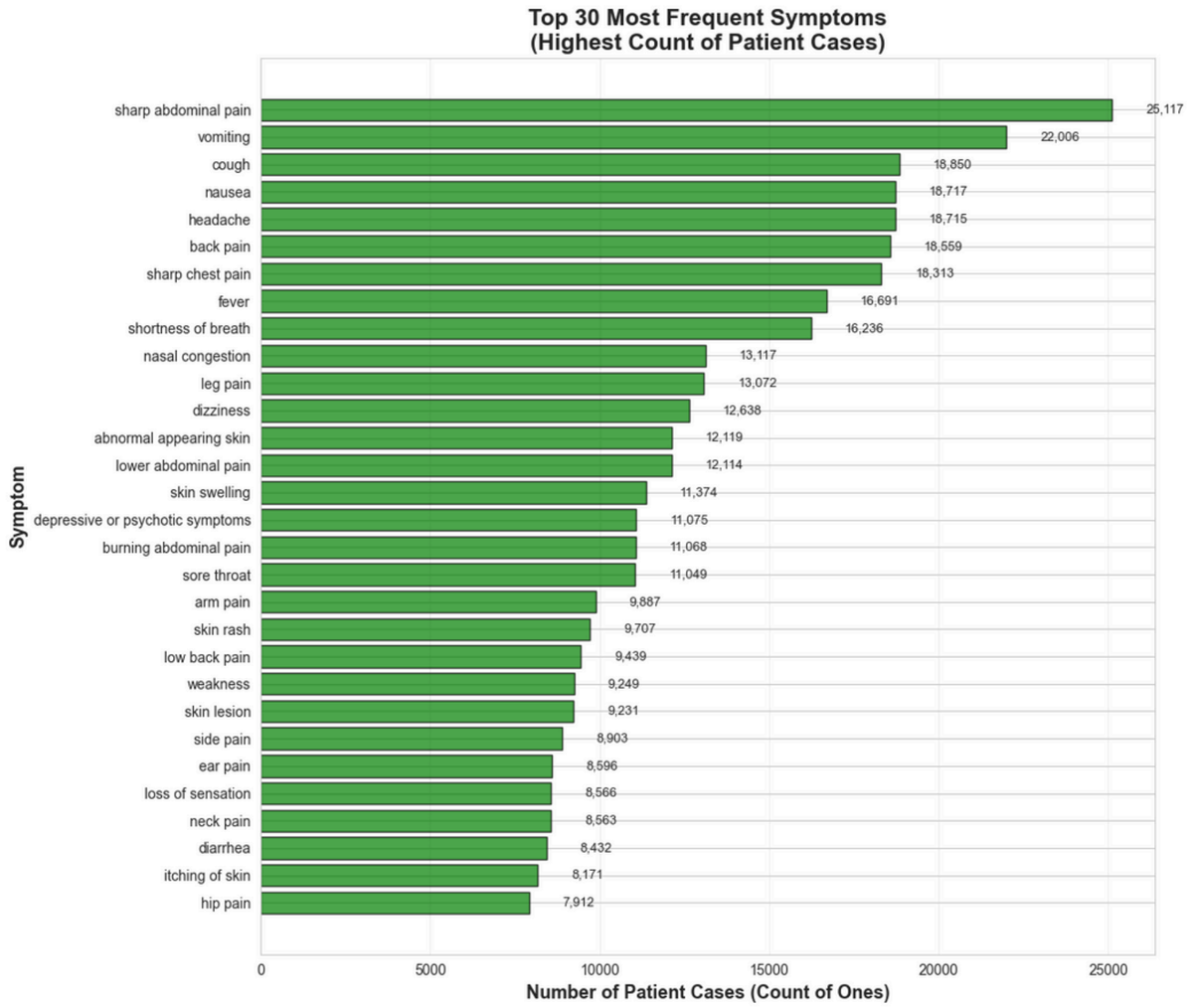
Symptoms visualization - Before preprocessing

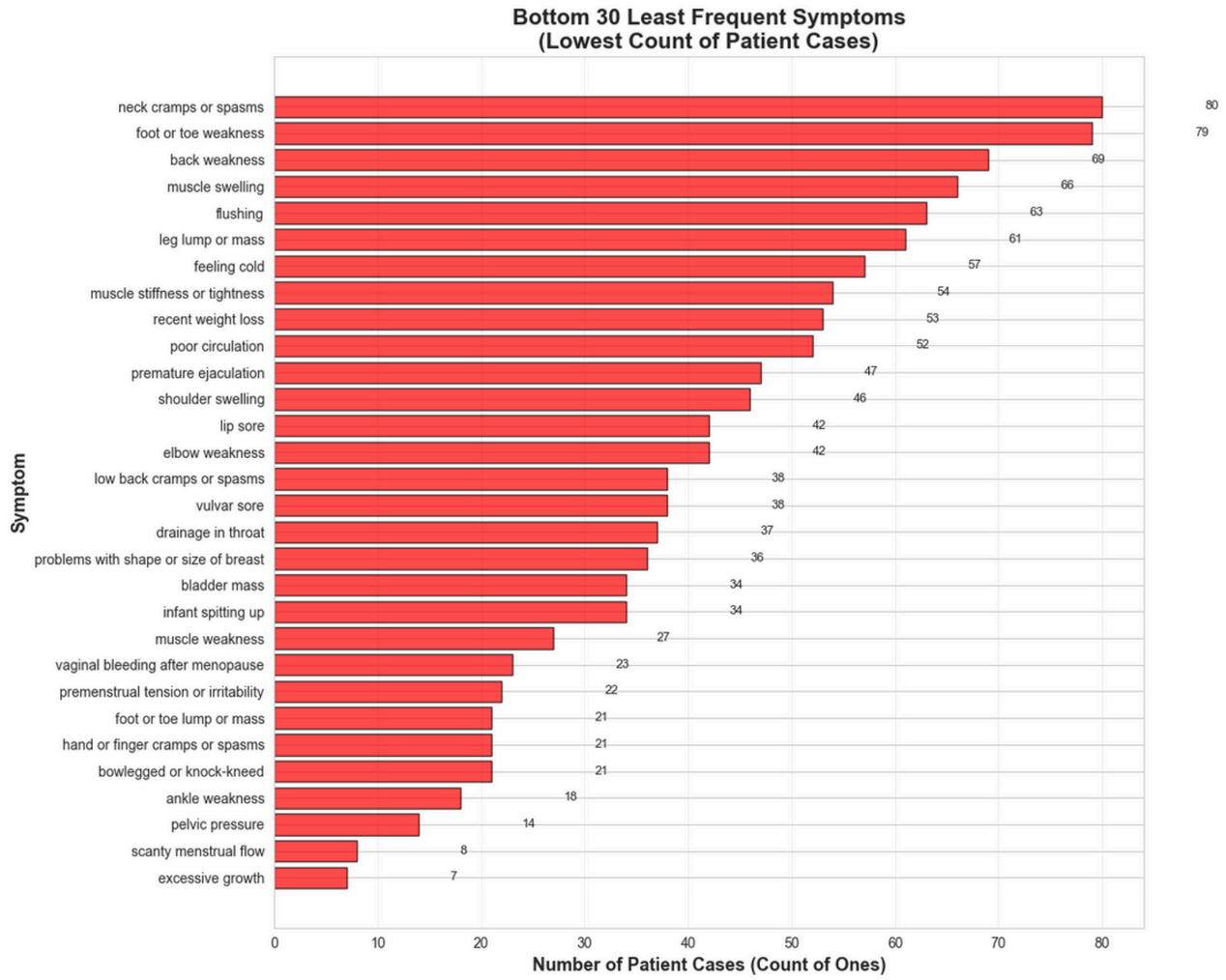




Phase 1

Symptoms visualization - After preprocessing





Phase 2

Algorithm Selection & Justification

- Why Random Forest?

1. Ensemble method combining multiple decision trees
2. Robust to outliers and imbalanced data

- Why Support Vector Machine (SVM)?

1. Linear Kernel: Efficient for large datasets with high-dimensional data (320 symptoms) where linear separation is effective
2. Good Generalization: Maximizes margin between classes, leading to better generalization
3. Excels with high-dimensional data (320 symptoms)

Why These Two Together?

- Complementary Strengths: Random Forest handles imbalanced data better, while SVM can capture complex boundaries
- Different Approaches: Tree-based vs. geometric approach provides diverse perspectives
- Baseline Comparison: Comparing these established algorithms helps validate our dataset quality

Random Forest

4.2 Model Evaluation

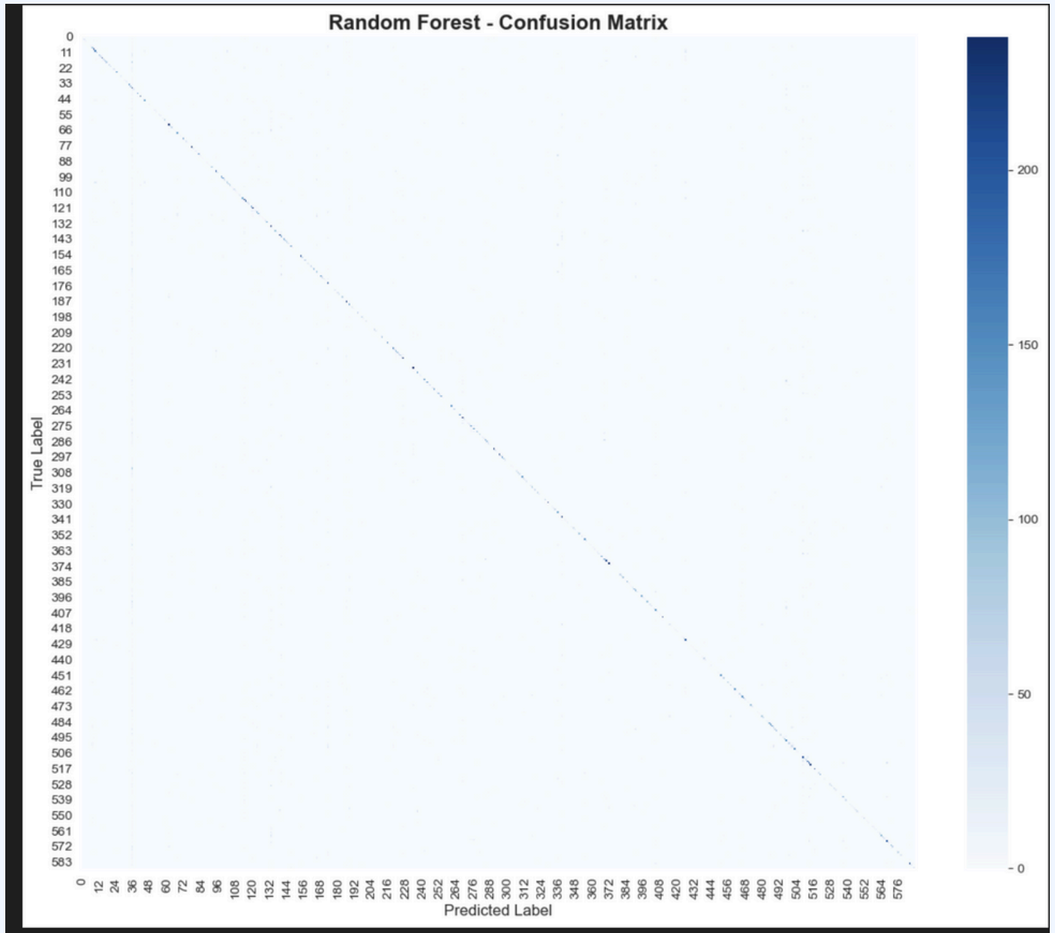
```python
# Calculate metrics
rf_accuracy = accuracy_score(y_test, y_pred_rf)
rf_precision = precision_score(y_test, y_pred_rf, average='weighted', zero_divi
rf_recall = recall_score(y_test, y_pred_rf, average='weighted', zero_division=0
rf_f1 = f1_score(y_test, y_pred_rf, average='weighted', zero_division=0)

print("\n" + "="*70)
print("RANDOM FOREST - PERFORMANCE METRICS")
print("="*70)
print(f"\nAccuracy:  {rf_accuracy:.4f} ({rf_accuracy*100:.2f}%)")
print(f"Precision: {rf_precision:.4f} ({rf_precision*100:.2f}%)")
print(f"Recall:    {rf_recall:.4f} ({rf_recall*100:.2f}%)")
print(f"F1-Score:  {rf_f1:.4f} ({rf_f1*100:.2f}%)")
```

✓ 1.5s Python

```
======================================================================
RANDOM FOREST - PERFORMANCE METRICS
======================================================================

Accuracy:  0.6747 (67.47%)
Precision: 0.7622 (76.22%)
Recall:    0.6747 (67.47%)
F1-Score:  0.6749 (67.49%)
```



✓ Confusion matrix saved as 'rf_confusion_matrix.png'

SVM

5.2 Model Evaluation

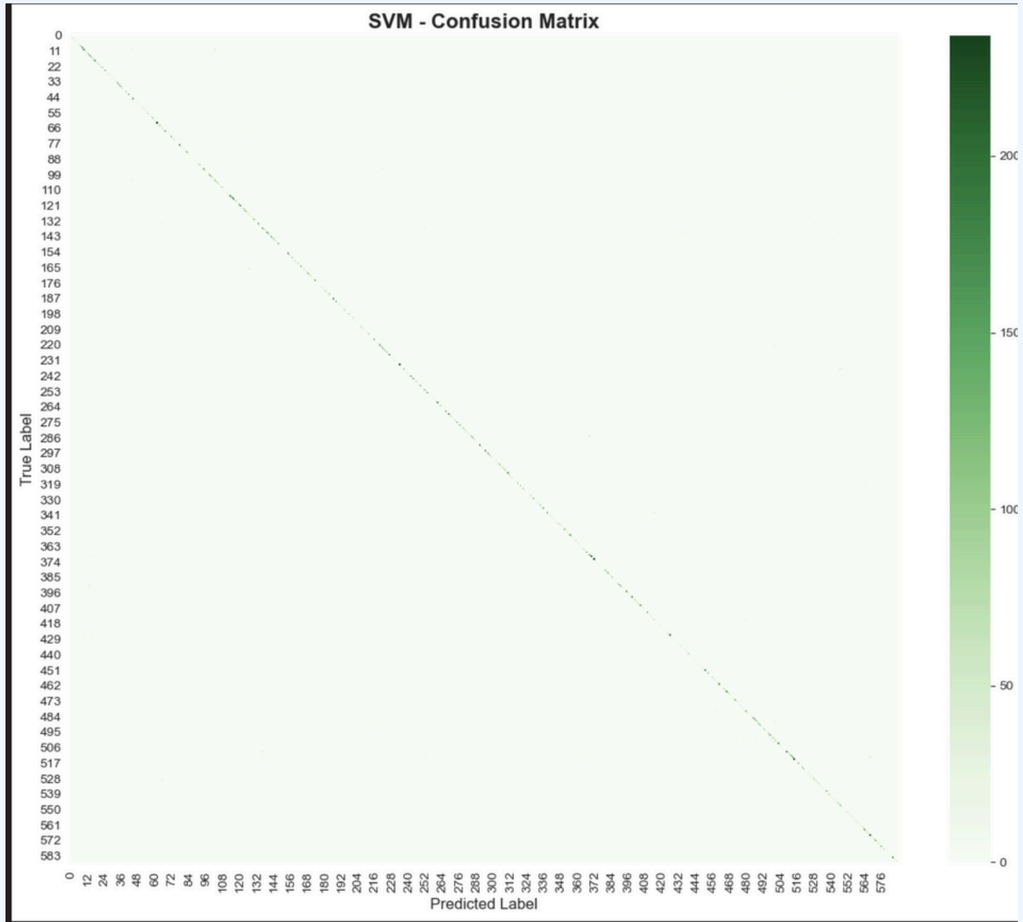
```python
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_s

svm_accuracy = accuracy_score(y_test, y_pred_svm)
svm_precision = precision_score(y_test, y_pred_svm, average='weighted', zero_di
svm_recall = recall_score(y_test, y_pred_svm, average='weighted', zero_division
svm_f1 = f1_score(y_test, y_pred_svm, average='weighted', zero_division=0)

print("\n" + "="*70)
print("SVM - PERFORMANCE METRICS")
print("="*70)
print(f"Accuracy:  {svm_accuracy:.4f} ({svm_accuracy*100:.2f}%)")
print(f"Precision: {svm_precision:.4f} ({svm_precision*100:.2f}%)")
print(f"Recall:    {svm_recall:.4f} ({svm_recall*100:.2f}%)")
print(f"F1-Score:  {svm_f1:.4f} ({svm_f1*100:.2f}%)")
```

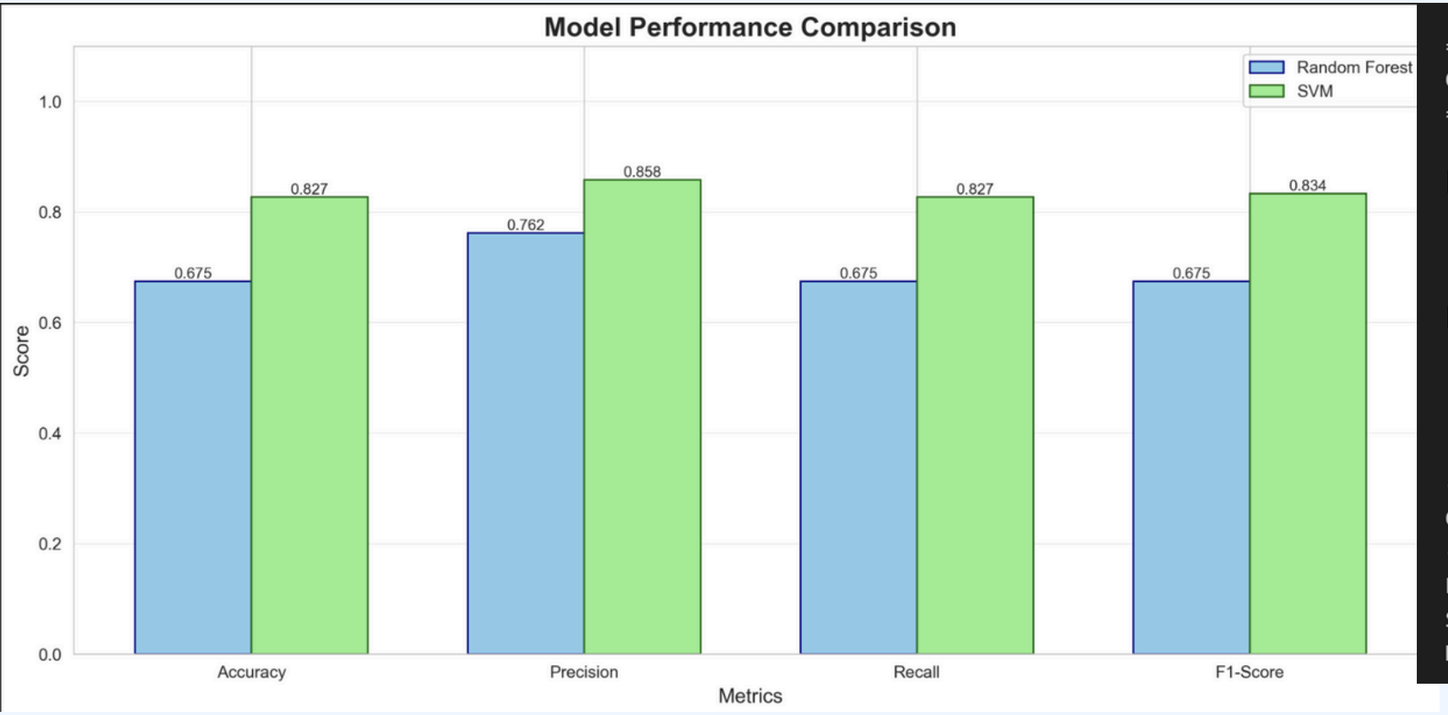
✓ 1.4s Python

```
======================================================================
SVM - PERFORMANCE METRICS
======================================================================
Accuracy:  0.8266 (82.66%)
Precision: 0.8584 (85.84%)
Recall:    0.8266 (82.66%)
F1-Score:  0.8338 (83.38%)
```




```
✓ Confusion matrix saved as 'svm_confusion_matrix.png'
```

Model performance comparison




```
================================================================================
CROSS-VALIDATION ANALYSIS (5-Fold)
================================================================================

[1] Random Forest Cross-Validation...
    Cross-Validation Scores: [0.68121427 0.67525937 0.67697967 0.66956913 0.655727
    Mean CV Score: 0.6717 (+/- 0.0177)

[2] SVM Cross-Validation...
    (This may take several minutes...)
    Cross-Validation Scores: [0.83874127 0.83839721 0.83228351 0.82450243 0.810422
    Mean CV Score: 0.8289 (+/- 0.0212)

--------------------------------------------------------------------------------
Cross-Validation Summary:
--------------------------------------------------------------------------------
Random Forest Mean: 0.6717
SVM Mean:           0.8289
Difference:         0.1571
```

Performance Gap: 15.19% (SVM superior)

Phase 2: Steps & Key Findings

Steps:

Step 1: Algorithm Selection

- Selected Random Forest & SVM based on data characteristics
- Random Forest: handles imbalance, provides interpretability
- SVM: excels with high-dimensional data

Step 2: Data Preparation

- Split: 80% training (151,136 records), 20% testing (37,784)
- Features: 320 symptoms (binary/categorical)
- Target: 587 disease classes

Step 3: Model Training

- Random Forest: 100 trees, optimized depth
- SVM with Linear kernel (LinearSVC)
- Training completed in <10 minutes

Step 4: Systematic Evaluation

- Metrics: Accuracy, Precision, Recall, F1-Score
- Validation: 5-fold cross-validation
- Comparison: Side-by-side performance analysis

Key Findings:

Finding 1: SVM achieves 82.66% accuracy (15.19% better than RF)
Finding 2: High-dimensional data (320 features) requires SVM
Finding 3: Cross-validation confirms consistent performance
Finding 4: Balanced metrics across all 587 disease classes

Phase 3

- Why K-means Clustering?

1. Scalability: Works efficiently with large datasets (246,945 samples)
2. interpretability: Produces clear cluster centroids representing typical symptom patterns
3. Binary Feature Compatibility: Works well with binary symptom features
4. Baseline Algorithm: Industry-standard for comparison

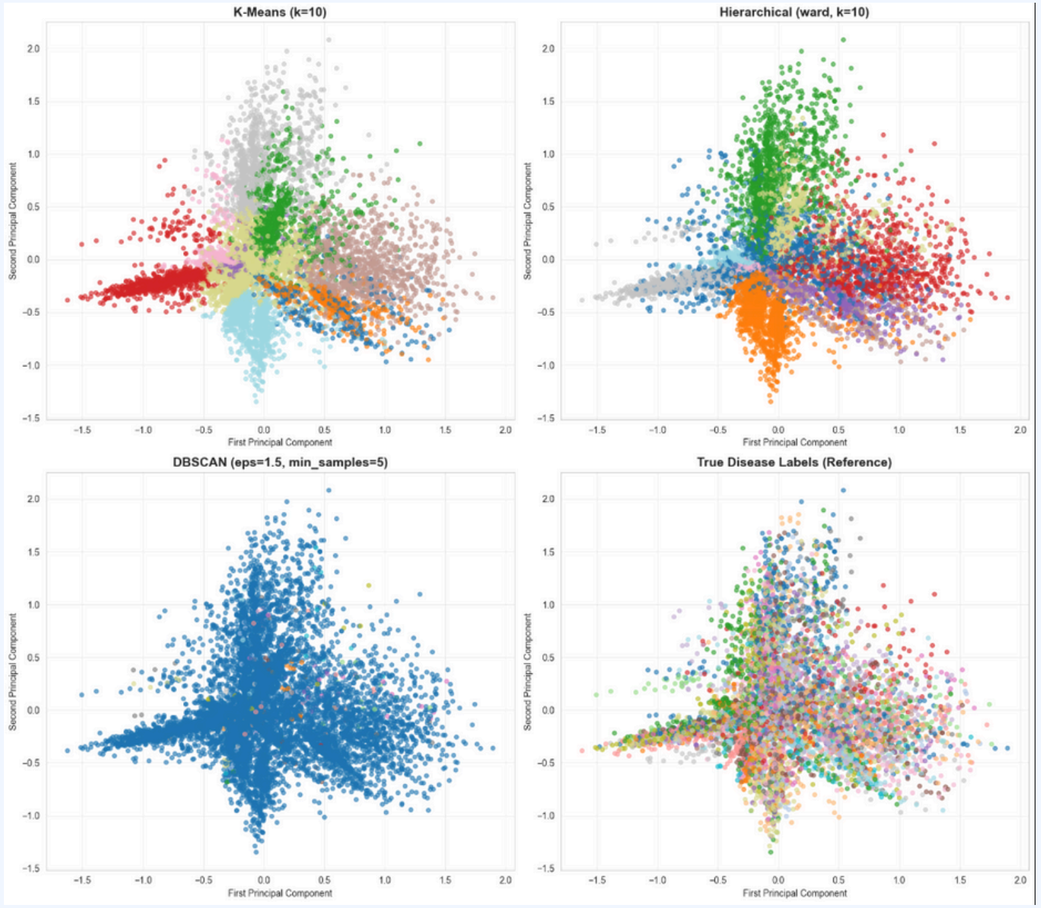
- Why Hierarchical Clustering?

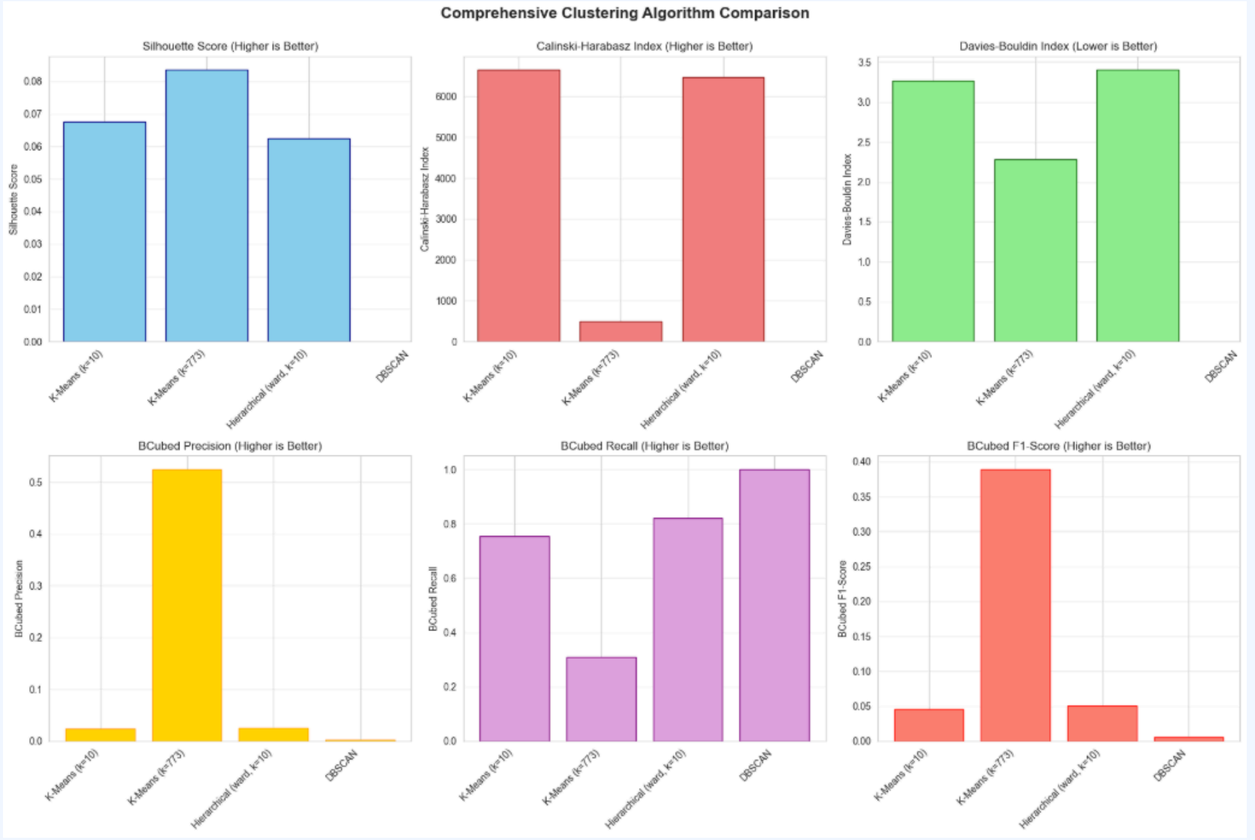
1. Dendrogram Visualization: Shows hierarchical relationships between diseases
2. No Need to Pre-specify K: Can determine optimal clusters from dendrogram
3. Medical Insight: Hierarchical structure aligns with medical taxonomy
4. Multiple Linkage Methods: Can compare different approaches (ward, average, complete)

- Why DBSCAN?

1. Noise Detection: Can identify outlier diseases or unusual symptom patterns
2. Arbitrary Cluster Shapes: Not limited to spherical clusters like K-Means
3. No Pre-specified K: Automatically determines number of clusters
4. Robustness: Less sensitive to outliers than K-Means

comparison


K-Means (k=10)
Hierarchical (ward, k=10)
DBSCAN (eps=1.5, min_samples=5)
True Disease Labels (Reference)
First Principal Component
Second Principal Component


Comprehensive Clustering Algorithm Comparison
Silhouette Score (Higher is Better)
Calinski-Harabasz Index (Higher is Better)
Davies-Bouldin Index (Lower is Better)
BCubed Precision (Higher is Better)
BCubed Recall (Higher is Better)
BCubed F1-Score (Higher is Better)
K-Means (k=10)
K-Means (k=773)
Hierarchical (ward, k=10)
DBSCAN

Phase 3: Key Steps

STEP 1: Data Preparation

- Removed class label ('diseases') to enable unsupervised learning.
- Applied PCA: Reduced 377 symptoms down to 193 key patterns (keeps 95% of information)
- Created 2D visualization showing data complexity.

Findings:

- Need 193 PCA components to retain 95% variance, showing high data complexity.

STEP 2: Determining Optimal Number of Clusters

- Tested K-Means with k=2 to k=20.
- Used Elbow Method (WCSS) and Silhouette Analysis.

Findings:

- k=10 is optimal with highest Silhouette Score of 0.0659.

STEP 3: Applying Three Clustering Algorithms

- K-Means: Applied with k=10 and k=773.
- Hierarchical: Applied with Ward linkage, created dendrograms.
- DBSCAN: Parameter tuning and application.

Findings:

- K-Means (k=773) achieved best Silhouette Score of 0.0836.
- DBSCAN failed, finding only 1 cluster in the 193 dimensions.

Phase 3: Key Findings

STEP 4: Comprehensive Evaluation Using Multiple Metrics

- Internal Metrics: Silhouette Score, Calinski-Harabasz, Davies-Bouldin, WCSS
- External Metrics: BCubed Precision, Recall, F1-Score.
- Findings:
- K-Means consistently outperformed other algorithms across all metrics.

Findings:

- k=10 has high recall (75.5%), k=773 has higher precision (52.4%).

STEP 5: Cluster Analysis & Interpretation

- Analysed characteristics of all 10 clusters.
- Found which symptoms are common in each cluster.

Findings:

- Natural disease families emerged (Cluster 0=Urinary, Cluster 1=Mental Health, etc).

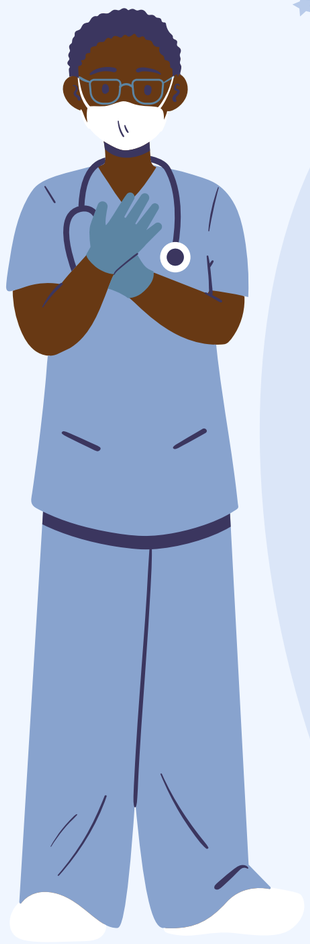
STEP 6: Integration with Supervised Learning

- Added cluster information as a new piece of data
- Trained baseline Random Forest model WITHOUT cluster info: 60.18% accurate.
- Trained enhanced Random Forest model WITH cluster info: 65.05% accurate

Findings:

- Adding cluster feature improved accuracy from 60.18% to 65.05% (+4.87%).

Phase 4



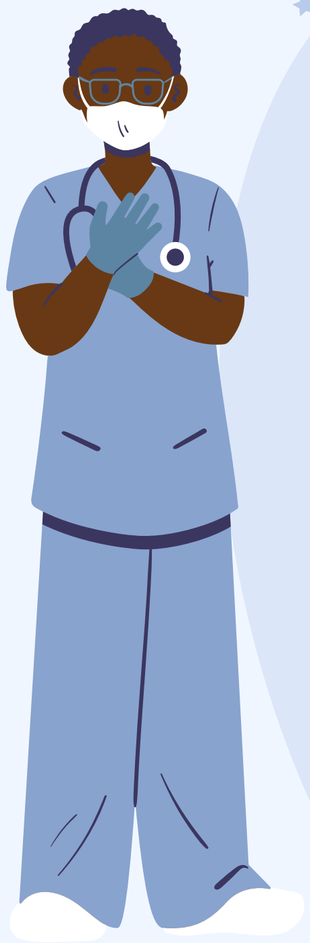
Purpose:

Enhance the system's explanations by using Generative AI (GPT) to produce clear and personalized medical advice based on user symptoms.

Why this matters:

Users need understandable explanations, not just predictions.

Phase 4- Implementation Overview

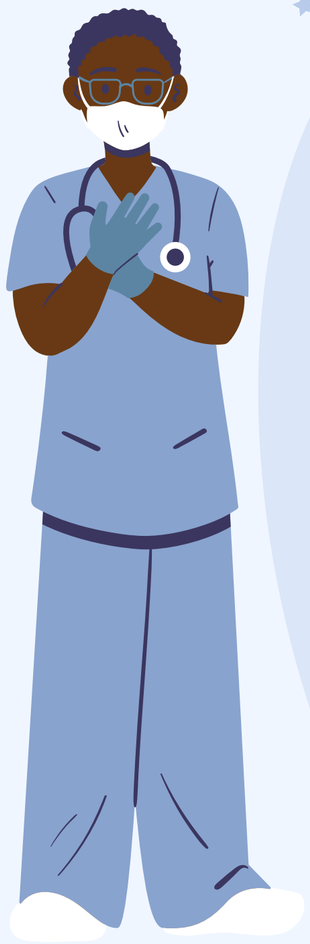


Implementation Overview:

- Connected GPT through the OpenAI API
- Designed two prompt templates with different explanation styles
- Tested both templates on multiple patient scenarios
- Compared outputs to choose the best template
- Integrated the chosen template into the system workflow
-

Note:
The Generative AI explanations build on the disease predictions produced in Phase 2 (SVM model).

Phase 4-The Two Prompt Templates



Template 1 – Detailed Medical Explanation

- Structured
- Rich medical terminology
- Action-oriented advice

Template 2 – Simple User-Friendly Explanation

- Conversational
- Easy to read
- Less clinical detail

By comparing both templates, we identify the one that better supports users' medical understanding.

Phase 4-AI Outputs

```
================================================================================
TEMPLATE 1: DETAILED MEDICAL PROFESSIONAL ADVICE
================================================================================
1. DISEASE OVERVIEW:
   - Atrial fibrillation (AF) is a cardiovascular condition characterized by an irregular and often rapid heart rate that can lead to blood clots, stroke, heart failure, a
   - Pathophysiologically, AF is caused by disorganized electrical activity in the atria, the upper chambers of the heart, leading to ineffective contraction and irregular
   - In terms of epidemiology, AF is the most common sustained cardiac arrhythmia, affecting approximately 2-3% of the population in developed countries. Prevalence increa

2. RISK FACTORS:
   - Specific risk factors based on the patient's profile include advanced age and being female. Women with AF are more likely to experience symptoms and are at a higher r
   - Non-modifiable risk factors include aging and genetic predisposition. Modifiable risk factors include hypertension, obesity, diabetes, heavy alcohol use, and other he
```

Figure 1. Output generated using the Detailed Medical Template, providing structured clinical insights and professional recommendations.

```
================================================================================
TEMPLATE 2: SIMPLE PATIENT-FRIENDLY ADVICE
================================================================================
1. WHAT IS HAPPENING:
Atrial fibrillation is like your heart's natural pacemaker is acting up. Instead of the steady, regular beats, it's causing your heart to flutter or beat too fast. This car

2. WHAT TO DO RIGHT NOW:
- Try to relax. Stress can make your symptoms feel worse. Take deep, slow breaths and sit or lie down if you need to.
- Avoid caffeine and alcohol. These can trigger irregular heartbeats.
- Keep a record of your symptoms. Note when they occur and what you were doing at the time. This can help your doctor understand your condition better.
```

Figure 2. Output generated using the Simple Patient-Friendly Template, offering easy-to-read advice and practical lifestyle tips.

Phase 4-Template Comparison & Final Decision

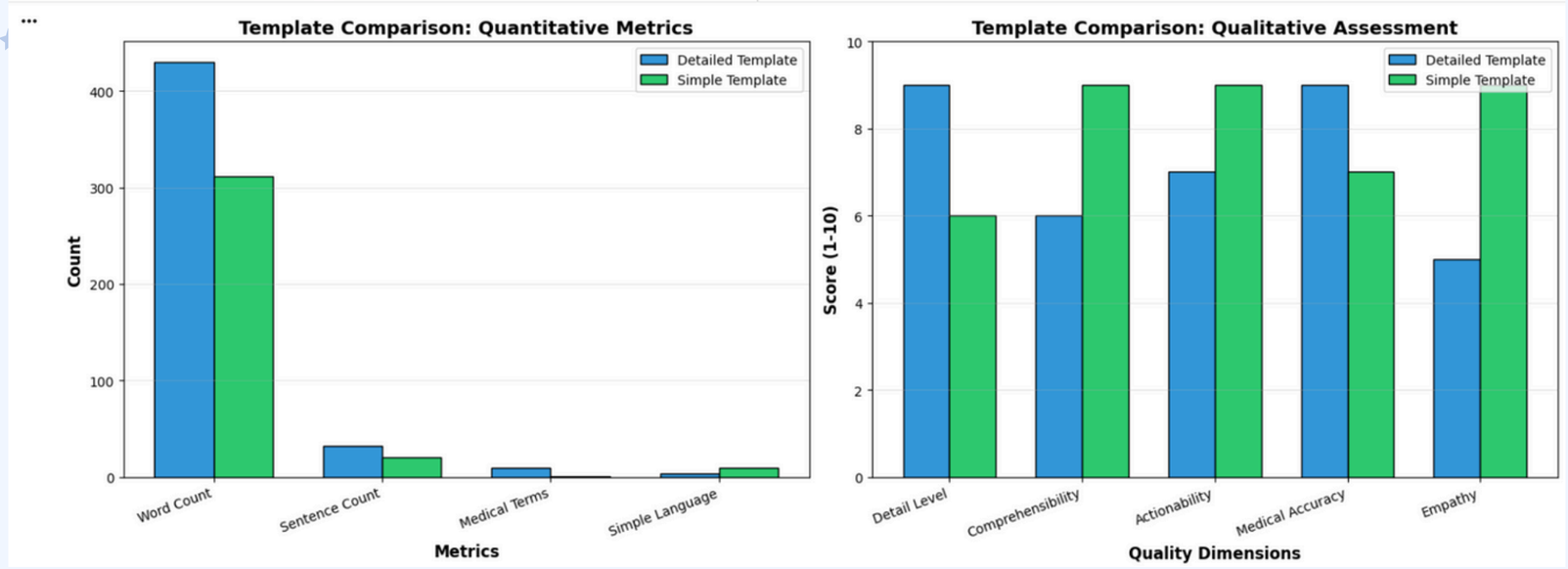


Figure 3. Quantitative and qualitative comparison between the detailed and simple templates, showing differences in word count, medical terminology, readability, and overall quality.

Template Comparison – Key Insights

- The detailed template uses more words and clinical terminology, which may overwhelm non-medical users.
- The simple template scores higher in readability, empathy, and patient comprehension.
- The simple template provides clearer, more accessible action steps for general patients.

Final Decision

We selected Template 2 because it offers higher comprehension, clearer action steps, an empathetic tone, and better supports general patients who may not have medical background.

Summary

Insight 01

High-Performance Prediction: The project successfully built and evaluated classification models (Random Forest, SVM).

Insight 02

Optimal GAI Prompting: Rigorous testing of GAI prompt templates proved essential.

Insight 03

Effective Patient Profiling: Unsupervised learning successfully created interpretable patient clusters

Conclusion & Recommendations

Recommendations

- Integrate the clustering module to auto-assign a patient to a profile.
- Deploy the GAI integration to provide immediate, personalized advisory notes to patients upon diagnosis.

Future Work

- Explore deep learning architectures for potentially higher predictive accuracy.
- Conduct further analysis on metrics to improve the predictions.

Lessons

Template Design

prompt engineering and template design are the key determinants of the GenAI's output style, detail, and tone.

Metrics

performing a comparison between the selected algorithms and using secondary metrics are crucial for selecting a reliable model in an imbalanced dataset.

Visualization

Visualisation provides the "why" behind the numerical metrics, making it indispensable for troubleshooting and decision-making.

Thank you