Disease Report Decision Maker for Scheduling Recommendation System

Fady Mohsen, Ahmad Mahmoud, Omar Mohamad, and Mohamad Elsayed
Systems and Biomedical Engineering Department, Cairo University
Developed For:
Fanous Health, Inc. USA

Supervised By:
Inas A. Yassine, PhD
Assoc, Prof.,
Systems and Biomedical Engineering Department, Cairo University
iyassine@eng.cu.edu.eg

Teaching Assistant:
Merna Bibars, MSc
Systems and Biomedical Engineering Department, Cairo University
merna.bibars@eng.cu.edu.

***Abstract*— The primary goal of this research is to develop a scheduling recommender system for Fanous Clinic, USA, aimed at optimizing the appointment scheduling process based on disease reports generated from the clinic's mobile application. The system recommends the type of visit required, categorizing it as a Primary Care visit alone, with a consultant, with a pharmacist, or with both. The system was developed using Python and the random forest algorithm.**

Data for training and evaluation was sourced from Fanous Clinic, USA. Our results demonstrate that the model effectively reduces the workload and time required by Primary Care physicians to review patient reports, allowing them to make more efficient and accurate scheduling decisions.

The implementation of this scheduling recommender system offers significant practical benefits, including time savings for healthcare providers and expedited scheduling processes for patients, thus improving the overall efficiency and effectiveness of clinic operations.

Keywords— Scheduling system, recommender system, clinic management, random forest algorithm, healthcare efficiency, patient scheduling.

I. Introduction

In the healthcare sector, efficient scheduling is crucial for managing patient appointments. This study focuses on Fanous Clinic, USA, which uses a mobile app to streamline administrative processes and generate disease reports. Effective scheduling optimizes resources and reduces wait times.

Our research aims to develop a scheduling recommender system for Fanous Clinic. By leveraging disease reports, the system recommends the most suitable type of visit—Primary Care alone or with additional consultations from a Consultant, Pharmacist, or both. A key feature of this system is the 'Pass' recommendation, indicating no need to charge the insurance company, thereby saving time and resources (effort for Primary Care Providers and money for Insurance Companies).

The system's goal is to shift clinic operations from fast intuitive thinking , to slow analytical mode . By providing instant, data-driven recommendations, it minimizes physicians' cognitive load and allows them to deliver evidence based care for patients. This transition from fast to slow thinking is an innovative solution for improving effectiveness and efficiency of clinical operations in healthcare settings.

We hypothesize that the system will reduce physicians' review time, balance appointment distribution, and enhance patient satisfaction. By analyzing disease reports, it efficiently recommends visit types, including a 'Pass' feature to save costs and time.

II. Literature Review

Overview of Existing Scheduling Systems

Various scheduling systems have been developed to address the complex needs of healthcare clinics. These systems range from basic appointment booking software to advanced algorithms that optimize scheduling based on various factors such as patient preferences, physician availability, and clinic resources. Some examples include traditional appointment scheduling software, queue-based systems, and machine learning-driven recommendation systems.

Relevant Theories And Models

Several theories and models underpin the design and implementation of scheduling systems in healthcare settings. Queueing theory, for instance, provides insights into the behavior of waiting lines and resource utilization, guiding the development of efficient scheduling algorithms.

Previous Studies And Their Findings

Previous research has explored the effectiveness of different scheduling approaches in improving clinic operations and patient satisfaction. Studies have investigated the impact of appointment scheduling policies on patient wait times, physician workload, and clinic efficiency. Additionally, research has examined the use of machine learning algorithms in predicting appointment demand, optimizing appointment sequencing, and reducing patient no-show rates.

Gaps In The Current Literature

The current literature on scheduling systems highlights several gaps despite advancements and extensive research in this field. First, there's a need for comprehensive evaluations of scheduling systems in real-world clinical settings,

considering factors like patient demographics, clinic size, and specialty services. Second, limited research exists on integrating patient preferences and feedback into scheduling algorithms, which could enhance patient satisfaction. Lastly, the absence of standardized benchmarks and performance metrics impedes effective comparisons across studies and systems.

III. Methodology

Research Design

The study employed an experimental research design with the primary objective of deploying a decision-making model within the Fanous Clinic Mobile Application. This model aimed to provide recommendations related to the Disease Report of the patient, aiding individuals in deciding who would attend the medical visit.

Data Collection

Data for the study were sourced from the records maintained by the Fanous Clinic Medical Staff Team. The Electronic Medical Records (EMR) system facilitated the collection of relevant patient data necessary for analysis.

Sampling Methods

Participants or data points for inclusion in the study were selected under the guidance of Dr. Hany Farag, MD, MBA, the Manager of the Clinic, leveraging his domain expertise in the medical field. Although there wasn't a specific sampling technique employed, Dr. Farag's insights ensured the selection of pertinent data for analysis.

IV. Dataset And Features

Description

The dataset comprises comprehensive medical profiles of patients, capturing various health parameters and conditions. Each record includes demographic information such as age and gender, along with primary and secondary disease diagnoses. Key cardiovascular metrics are recorded, including LDL cholesterol levels and heart rate. The dataset details medication usage, specifying the type and dosage of statins. Smoking status and COPD assessments are also included, providing GOLD stage and group classifications. Additional respiratory health metrics cover the use of COPD medications, asthma control status, and exacerbation risk. Inhaler usage, including short-acting beta agonists (SABA), short-acting muscarinic antagonists (SAMA), and inhaled corticosteroids (ICS) dosage, is documented. The dataset aims to provide a holistic view of each patient's health, facilitating a thorough analysis of their medical conditions and treatment regimens. This information is critical for research focused on the interplay between various health conditions and the effectiveness of different treatment approaches.

Handling Missing Values

Check for Missing Values: The dataset was meticulously reviewed for any missing entries using Python's pandas library. Functions such as $isnull(\ )$ and $sum(\ )$ were employed to identify the presence and extent of missing data across various features.

Strategy for Handling Missing Values: Upon identifying missing values, a strategy was devised to address them. Depending on the nature and distribution of the missing data, different approaches were considered:

Dropping Missing Values: In instances where the proportion of missing data was significant or no logical imputation method was applicable, the corresponding rows were dropped to maintain data integrity.

Encoding Categorical Variables

Categorical variables in the dataset were identified and transformed into numerical formats to facilitate effective model training. The process involved:

Identification of Categorical Variables: Using pandas' $select_dtypes(\ )$ function, categorical features were isolated for encoding.

Label Encoding: For ordinal categorical variables, label encoding was used to assign each category a unique numerical value.

One-Hot Encoding: For nominal categorical variables with no intrinsic ordering, one-hot encoding was applied using the $get_dummies(\ )$ function in pandas. This approach created binary columns for each category, ensuring no ordinal relationship was inferred by the model.

Feature Scaling

To bring numerical features to a comparable scale and enhance the model's performance, feature scaling was implemented. The steps included:

Normalization: Features were normalized to a [0, 1] range using the $MinMaxScaler$ from scikit-learn. This method was chosen for its effectiveness in preserving the distribution of the data while ensuring that all features contribute equally to the model.

Standardization: In cases where normal distribution was assumed, standardization was performed using $StandardScaler$ from scikit-learn. This technique transformed the features to have a mean of 0 and a standard deviation of 1, facilitating better convergence during model training.

V. Exploratory Data Analysis

Perform univariate analysis to understand the distribution of each feature.

Conduct bivariate and multivariate analysis to explore relationships between features and the target variable.

Visualization

Use histograms, box plots, and scatter plots to visualize the distribution of features.

Use heatmaps and correlation matrices to identify correlations between features.

VI. Machine Learning Models

XGBoost

XGBoost (Extreme Gradient Boosting) is an optimized distributed gradient boosting library designed to be highly efficient and flexible. It uses a gradient boosting framework and is particularly effective for large datasets and complex models.

$$[Obj(\theta) = \sum_{i=1}^{n} L\left(y_i, \hat{y}_i\right) + \sum_{k=1}^{K} \Omega\left(f_k\right)]$$

Decision Tree

A Decision Tree is a non-parametric supervised learning algorithm used for classification and regression tasks. It splits the data into subsets based on the value of input features, forming a tree-like structure.

$$[Gini(D) = 1 - \sum_{i=1}^{C} p_i^2]$$

Random Forest

Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the mode of the classes (classification) or mean prediction (regression) of the individual trees.

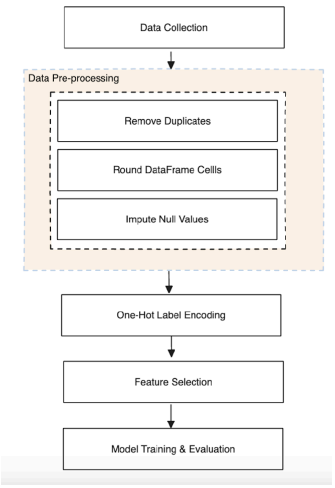
$$[Accuracy = \frac{1}{N}\sum_{i=1}^{N} 1\left(\hat{y}_i = y_i\right)]$$

Voting Classifier

The Voting Classifier is an ensemble learning technique that combines the predictions from multiple models to improve accuracy. It can be implemented as either hard voting (majority voting) or soft voting (average predicted probabilities).

$$[\hat{y} = mode\left(\hat{y}_1, \hat{y}_2, ..., \hat{y}_M\right)]$$

Block Diagram



Block Diagram Description

The block diagram illustrates a typical workflow for preparing and analyzing data in a machine learning project. The process begins with data collection, where raw data is gathered from various sources. This is followed by a comprehensive data pre-processing stage, which includes three critical steps:

Remove Duplicates: This step ensures that any redundant data entries are eliminated to maintain the integrity of the dataset.

Round DataFrame Cells: This step involves rounding off the values in the data frame to a specified number of decimal places to standardize the data format.

Impute Null Values: Missing or null values in the dataset are handled by imputing them with appropriate values, such as the mean, median, or mode of the column.

After pre-processing, the data undergoes **One-Hot Label Encoding**, which transforms categorical variables into a format that can be provided to machine learning algorithms to better understand the data. This is followed by **Feature Selection**, a process that identifies and selects the most significant features from the dataset to improve the model's performance and reduce complexity.

Finally, the processed and refined data is used for **Model Training & Evaluation**, where machine learning models are trained on the data and evaluated for their performance using various metrics. This step is crucial for developing an effective predictive model that can generalize well to new, unseen data.

VII. Results and decision

XGBoost: 70.91% Decision Tree: 78.79%
Random Forest: 56.36% Voting Classifier: 67.88%

Key Hyperparameters for Decision Tree Algorithm

1. **criterion:**
   - Options: gini, entropy
   - Description: The function to measure the quality of a split. Gini impurity and entropy are the most used.
   - Selection: Start with gini as it is faster to compute. Use entropy if you prefer a more information-theoretic approach.

2. **max_depth:**
   - Description: The maximum depth of the tree. Limiting the depth can prevent overfitting.
   - Selection: Use cross-validation to find the optimal value. Start with a range like 5, 10, 15, 20.

3. **min_samples_split:**
   - Description: The minimum number of samples required to split an internal node.
   - Selection: Common values are 2, 10, 20. Increase this value if you have a large dataset to reduce overfitting.

4. **min_samples_leaf:**
   - Description: The minimum number of samples required to be at a leaf node.
   - Selection: Common values are 1, 5, 10. Increase this value to make the model more robust by preventing it from learning much from outliers.

5. **max_features:**
   - Description: The number of features to consider when looking for the best split.
   - Selection: sqrt(n_features) or log2(n_features) for large datasets. Use a higher value for smaller datasets.

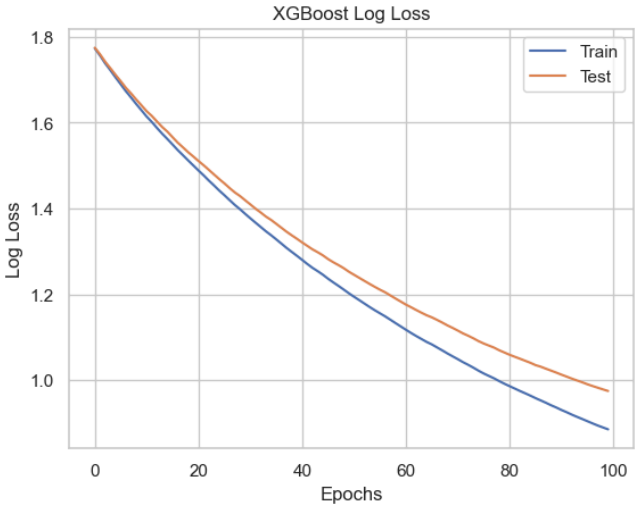
6. **max_leaf_nodes:**
   - Description: Grow a tree with max_leaf_nodes in best-first fashion. Best nodes are defined as relative reduction in impurity.
   - Selection: Use cross-validation to determine the optimal number. This helps in controlling the size of the tree.

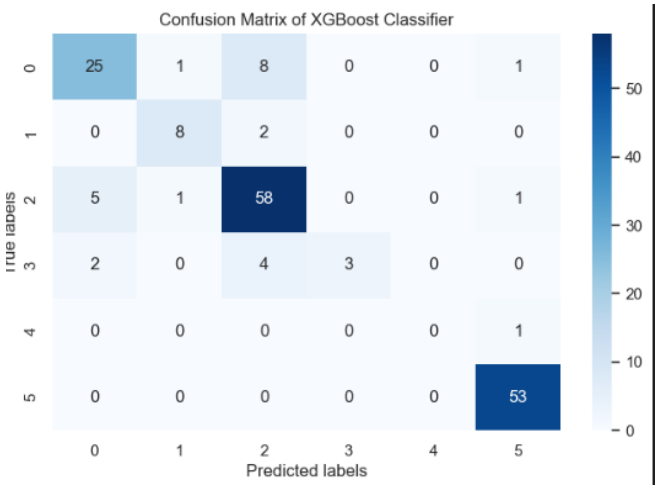
VIII. HYPER PARAMETER TUNING APPROACH

Grid Search allows you to exhaustively search through a predefined hyperparameter space to find the best combination of hyperparameters for your model.

XGBoost Classifier: Loss over 100 Epochs

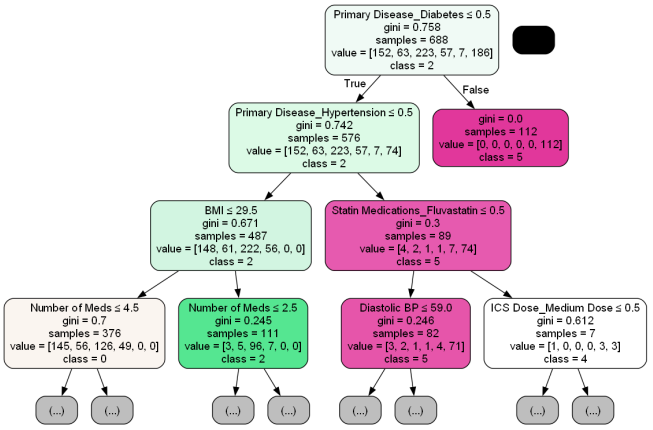
FIRST EXPERIMENT



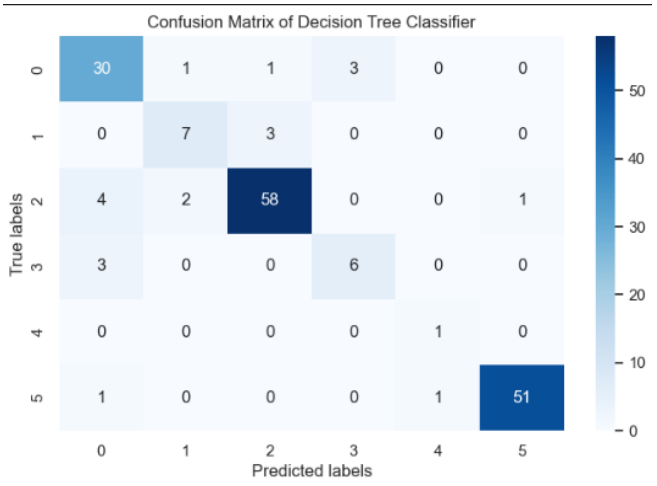


FIRST EXPERIMENT

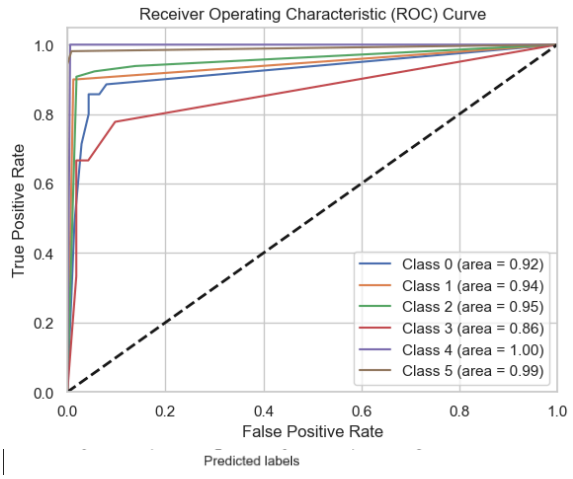
Visualization of the tree till a depth of 3



Confusion Matrix



ROC Curve



IX. Key Metrics For Model Evaluation

Precision

$$Precision = \frac{TP}{TP+FP}$$

Recall

$$Recall = \frac{TP}{TP+FN}$$

Accuracy

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN}$$

F1-Score

$$F1 - Score = 2 \times \frac{Precision \times Recall}{Precision+Recall}$$

X. Conclusion / Future Work

Summary And Key Points

This study explored the deployment of a decision-making model within the Fanous Clinic Mobile Application to assist in determining who should attend patient visits. Four machine learning algorithms were evaluated: XGBoost, Decision Tree, Random Forest, and Voting Classifier. The Decision Tree model emerged as the highest performing algorithm with an accuracy of 80.10%. The superior performance of the Decision Tree could be due to its effective handling of complex, non-linear relationships within the dataset.

Future Work

Given more time, additional team members, or greater computational resources, several avenues for future work could be pursued:

- Hyperparameter Optimization: Further fine-tuning of hyperparameters using more extensive search techniques like Bayesian optimization.
- Feature Engineering: Investigating additional features or advanced feature engineering techniques to enhance model performance.
- Deep Learning Models: Experimenting with neural networks or deep learning models to capture more complex patterns in the data.
- Comprehensive Evaluation: Conducting a more extensive evaluation using additional metrics and larger validation sets to ensure model robustness.
- Real-time Deployment: Implementing the model in a real-time environment and monitoring its performance over time to make necessary adjustments.

XI. References

Algorithms And Models

[1] Chen, T, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785-794).

XII. Libraries Used

Pandas, NumPy, Matplotlib, Seaborn, Scikit-learn, and XGBoost.

XIII. Team Contribution

Data Preprocessing and Analysis:

Fady Mohsen and **Ahmad Mahmoud** were responsible for the data preprocessing, including removing duplicates, rounding DataFrame cells, and imputing null values. They also conducted the data analysis and visualization, providing valuable insights into the dataset.

Model Development and Experiments:

Omar Mohamed and **Mohamed Elsayed Eid** focused on building machine learning models, conducting experiments, and interpreting the results. Their work involved model training and evaluation, as well as hyperparameter tuning to optimize model performance.

The collaborative efforts of all team members contributed to the successful completion of this project, with each member bringing their expertise to different stages of the data science workflow.