- Patient Flow Model for Hospital Admission Analysis -

Prepared by:

Miss Ikmee U-daidee

Supervisor:

Dr. Nurul Izrin Binti Md Saleh

Problem Background

- Hospital overcrowding is a pervasive issue that affects healthcare systems worldwide, leading to long waiting times and delays in admissions to critical care units. This problem arises when hospitals operate beyond their capacity due to a combination of factors, including an influx of patients seeking treatment, a shortage of medical staff, and insufficient inpatient beds.

- Inefficient management of patient flow is a crucial factor in this equation. Delays in patient movement can result in longer hospital stays and increased readmission rates, significantly impacting the quality of patient care.

Solution

- This project I will use the k-Nearest Neighbor (KNN) model to analyze hospital admission data, and emphasize the role of data visualization in understanding and addressing the issues related to patient flow and overcrowding.
- This approach aims to enhance resource allocation and improve overall operational efficiency and patient care.

Problem statement

- **Insufficient Understanding of Diagnosis-Specific Flow Patterns:**

  The absence of data visualized of how specific diagnoses contribute to patient flow dynamics creates challenges in identifying which medical conditions are most closely associated with overcrowding at different times.

- **Difficulty in Integrating Historical Data for Predictions:**

  Hospitals face challenges in integrating historical patient diagnosis data to create accurate prediction models, limiting their ability to anticipate and mitigate future overcrowding effectively.

- **Lack of Explainability in Prediction Models:**

  Stakeholders struggle with interpreting and understanding the predictive models used for anticipating overcrowding. The absence of explainability makes it difficult for healthcare professionals to trust and act upon the predictions, which limits the effectiveness of these models in decision-making and patient flow management.

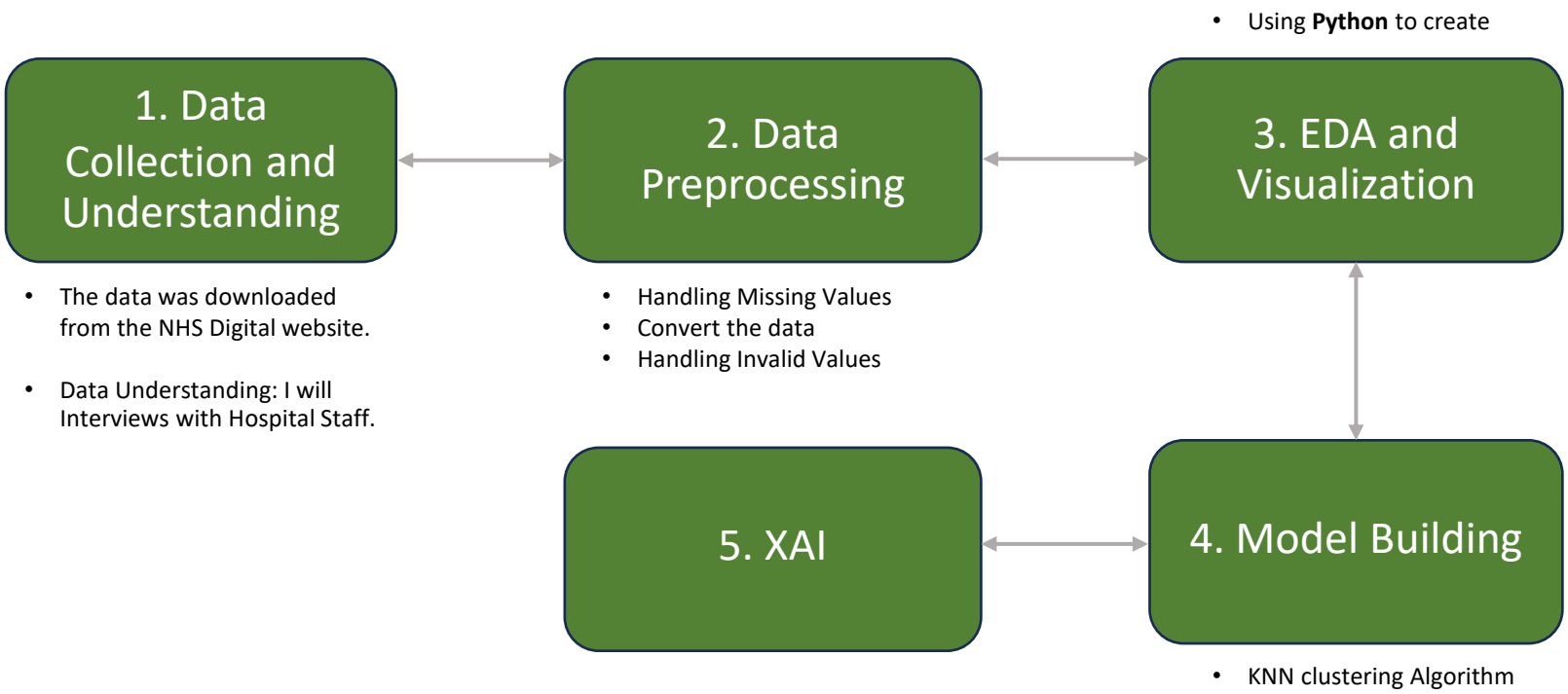
Objectives

- To conduct data visualization of diagnosis-specific flow patterns in order to identify which medical conditions are most closely associated with overcrowding during different times. This will aid in targeted resource allocation and improve patient flow management.
- To develop robust methods for integrating historical patient diagnosis data into predictive models, enhancing the hospital's ability to accurately forecast diagnosed patient influx and manage future overcrowding proactively.
- To improve the explainability of predictive models used for forecasting patient flow and overcrowding, ensuring that healthcare professionals can trust and comprehend the outputs, leading to better-informed decisions and improved management of hospital resources.

Scope of study

- **Source:** NHS Digital [Hospital Episode Statistics (HES)]
- **Publication Title:** "Hospital Admitted Patient Care Activity 2022-23“
- **Dataset Used:** Hospital Episode Statistics Admitted Patient Care Diagnosis
- **It Includes :** Details on diagnoses, consultancy episodes, and demographic attributes of patients

Methodology



Objective 1: Conduct data visualization

- **Python** is use to create various visualizations that can help uncover insights and effectively communicate data patterns.

- **The libraries used** include :

  Seaborn, Plotly Express, and Matplotlib.

Objective 1: Conduct data visualization

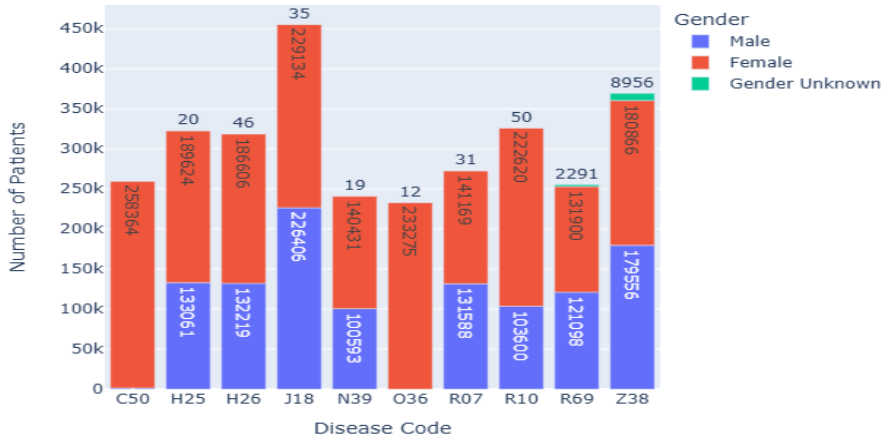


Fig. 1 Total number of patients by gender

- This graph show the top 10 diseases with the highest total number of patients, categorized by gender.
- The graph is a **stacked bar plot**
- **Library used:** Plotly Express (px.bar) function to create a bar chart

- **Axes:**
  - **Y-axis:** Show the total number of patients for each diagnosis.
  - **X-axis:** Show the diagnosis codes.

- **Colors within each bar:**

  - **Blue**: Male patients

  - **Red**: Female patients

  - **Green**: Gender unknown patients

Objective 1: Conduct data visualization

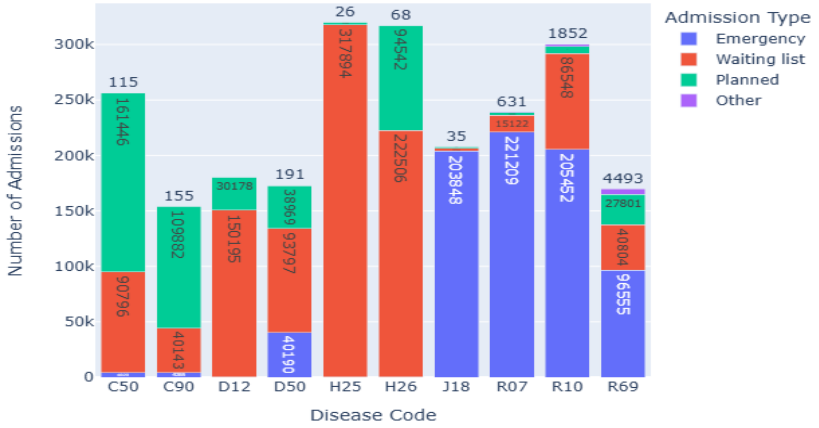


Fig. 2 Total Number of Admission by Admission Types

- This graph shows the number of patient admissions by type of admission. For the top 10 diseases.
- The graph is a **stacked bar plot**
- **Library used:** Plotly Express (px.bar)

- **Axes:**
  - **Y-axis:** Show the total number of patients for each diagnosis.
  - **X-axis:** Show the diagnosis codes.

- **Colors within each bar:**
  - **Blue**: Emergency
  - **Red**: Waiting list
  - **Green**: Planned
  - **Purple**: Other

Objective 1: Conduct data visualization

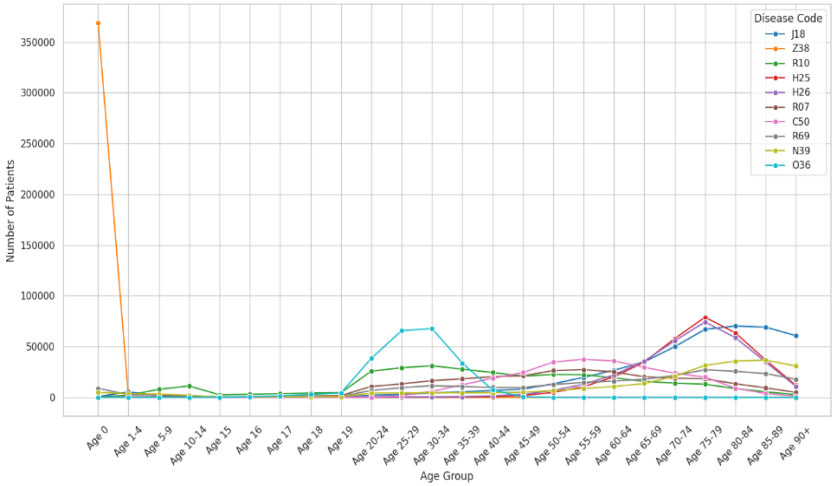


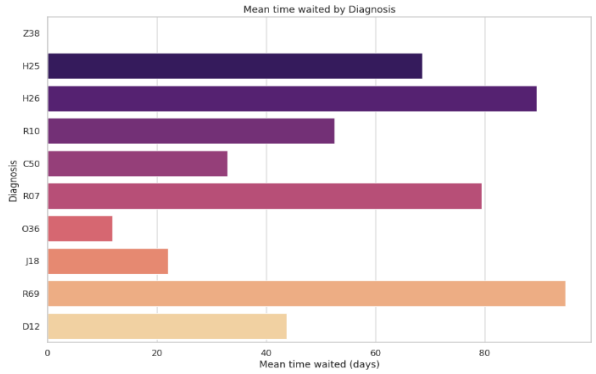
Fig. 3 Total Number of Patients by Age Group

- **Axes:**
  - **X-axis:** shows the age group.
  - **Y-axis:** shows the number of patients.

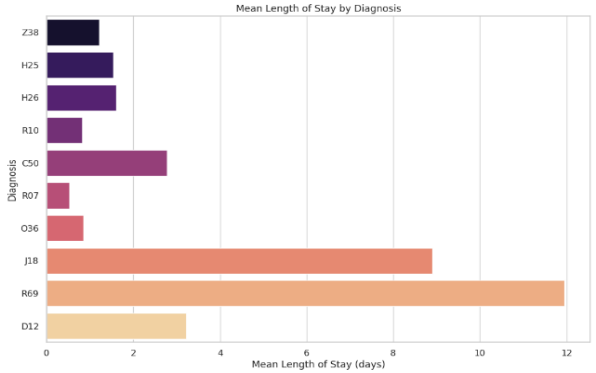
- This line plot shows the distribution of various diagnosis across different age groups.
- The graph is a **Line plot**
- Each line represents one disease, differentiated by color, and the points on the line indicate the number of patients in each age group.

- **Library used:**
  - Seaborn: Used to create line charts, by using `sns.barplot()` function, use `hue` to separate each line by disease code, with each disease having a different color.
  - Matplotlib: Basic library used in Seaborn to display the chart, using `plt.show()` to display the chart.

Objective 1: Conduct data visualization



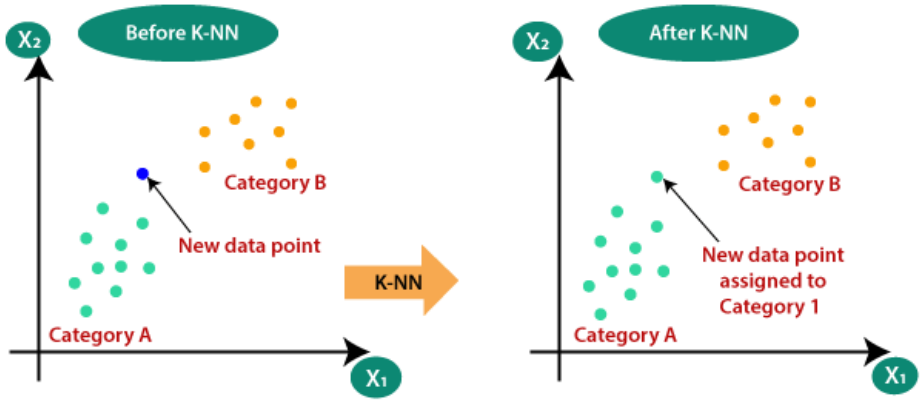
- This graph shows the average waiting time (Mean time waited) before treatment for various diseases.
- The graph is a **Bar plot**: It help to see which diseases have the longest waiting times before receiving treatment.
- **Axes:**
  - **X-axis:** shows the number of waiting days.
  - **Y-axis:** shows the diagnosis codes.



- This graph shows the average length of hospital stay for the top 10 diseases.
- The graph is a **Bar plot**: It helps to see which diseases have the longest hospital stays.
- **Axes:**
  - **X-axis:** shows the average length of stay (in days).
  - **Y-axis:** shows the diagnosis codes.

- **Library used:**
  - Seaborn: Used to create bar charts, using sns.barplot() function to create the chart.
  - Matplotlib: Basic library used in Seaborn to display the chart, using plt.show() to display the chart.
- **Color Palette:** Use the "magma" option.

Objective 2: KNN Clustering



- KNN works by finding 'k' data points that are closest to the new data in the existing data set, using distance to calculate the closeness.
- 'k' is a pre-defined value of how many neighboring data points to use in the decision.
- Once the k closest data points are selected, KNN will perform a majority voting to predict how the new data should be classified or in which group it should be. The predicted result will be the group with the largest number of k neighboring data points.

- Euclidean Distance

$$dist = \sqrt{\sum_{k=1}^{n}(p_k - q_k)^2}$$

Objective 2: KNN Clustering

Step :

- Library: scikit-learn
- Train/Test Split (Training Set and Test Set)
- Building Model KNN (Choose the value for K)
- Make Prediction
- Model Evaluation (Accuracy, Precision, Recall และ F1-score)
- Hyperparameter Tuning (Using Cross Validation to Get the Best Value of k)

Plan for FYP 2

- Create and run model KNN
- XAI

Thank you for your attention

…

Any questions and suggestions