****Predicting Drug Prescriptions Based on Patient’s Health Metrics: A Machine Learning Approach****

****Submitted By:****

****Sreerag M (AM.LS.P2BIF24028)****

****Neeraj Krishna R (AM.LS.P2BIF24019)****

****MSc Bioinformatics S2****

****AIM****

**To develop and evaluate machine learning models that can accurately classify which type of drug should be prescribed to a patient based on their medical attributes such as age, sex, blood pressure, cholesterol levels, and sodium-to-potassium ratio.**

**METHODS**

We used a dataset named drug200.csv containing 200 records from kaggle (<https://www.kaggle.com/datasets/prathamtripathi/drug-classification/data>).

Each record represents a patient with several features:

* **Numerical features:** Age, Na\_to\_K
* **Categorical features:** Sex, BP, and Cholesterol
* **Target variable:** Drug

**To make the data suitable for machine learning models, several preprocessing steps were taken:**

* **We e**nsured there were no null values.
* ****Encoding:****
  + Categorical features like Sex, BP, Cholesterol, and Drug were converted into numeric values using **Label Encoding**.
* ****Normalization:****
  + Since models like KNN and SVM are sensitive to feature scales, you applied **MinMaxScaler** to normalize all features to a 0–1 range.

**We separated the input features and the target variable into** X and y, and also split the dataset into training and testing sets using an 80–20 ratio.

**We evaluated four popular Machine Learning models:**

### ****Decision Tree****

A tree-structured classifier that splits data based on feature conditions. It's intuitive, easy to visualize, and handles both numerical and categorical data. However, it may over-fit if the tree is too deep or unpruned.

### ****K-Nearest Neighbors (KNN)****

A distance-based model that assigns a class to a data point based on the majority class among its closest neighbors. It’s simple and effective for small datasets but requires data normalization for best results.

### ****Random Forest****

An ensemble technique that combines multiple decision trees using bagging. It boosts predictive accuracy and minimizes over-fitting. Also useful for determining the most important features.

### ****Support Vector Machine (SVM)****

A powerful classifier that finds the best separating hyperplane between classes. It works well for complex datasets, especially in high-dimensional spaces, but relies heavily on proper scaling of input features.

**For each model, we calculated multiple performance metrics:**

* ****Accuracy**:** Ratio of correct predictions to total predictions.
* ****Classification Report**:** Includes:
  + **Precision** – TP / (TP + FP)
  + **Recall** – TP / (TP + FN)
  + **F1-score** – Harmonic mean of precision and recall
* ****Macro Average**:** Average score treating all classes equally.
* ****Weighted Average**:** Average score weighted by support (number of true instances per class).
* ****Confusion Matrix**:** Breakdown of prediction vs actual labels.

In this project, we utilized the **Scikit-learn (sklearn)** library to implement and evaluate various machine learning models, and employed **Pandas** for efficient data loading, preprocessing, and analysis.

**RESULT**

From the evaluation table, we observe that **Decision Tree** and **Random Forest** models performed perfectly on the training dataset, achieving 100% scores in accuracy, precision, recall, and F1 score (both macro and weighted)

The **K-Nearest Neighbors (KNN)** model showed a strong performance with an accuracy of **85%**, and balanced precision and recall values, indicating decent generalization.

The **Support Vector Machine (SVM)** model also performed reasonably well, with an accuracy of **82.5%** and slightly lower precision and recall than KNN.

**DATA AVAILABILITY**

The dataset and the complete implementation code for this project are available on the [GitHub](https://github.com/Sree-ragM/BDML) page for easy access and reproducibility.