Patient Readmission Prediction (K-Nearest Neighbor)

Ashley Traore

Research Question:

Can the k-nearest neighbor's method be used to predict patient readmission status, allowing the company to make informed business decisions to reduce readmission rates?

Goals for Analysis:

The primary goal of this analysis is to develop a machine learning model using k-nearest neighbor to help the company identify patients who have a higher risk of readmission. The company can then make more informed business decisions to reduce readmission rates.

K-Nearest Neighbor (KNN) Classification Analysis and Expected Outcomes:

The k-nearest neighbor (KNN) classification method is a simple and effective algorithm that determines the class of an unlabeled data point by examining its 'k' nearest neighbors in the feature space. KNN calculates the distance between the unlabeled point and all points in the training dataset using metrics such as Euclidean, Manhattan, or Minkowski distance to measure proximity. Smaller distances indicate greater similarity, enabling the algorithm to identify the 'k' closest points and classify the unlabeled data point based on the most frequently occurring class label among its neighbors. In this analysis, a k value of 10 was used, meaning the classification was based on the 10 closest data points.

Each distance metric has specific use cases. Euclidean distance measures the straight-line relationship, making it ideal for continuous data. Manhattan distance calculates the total of the absolute differences between the coordinates of two points. It is useful for situations where movement occurs along a grid or fixed paths, such as navigating city streets. Minkowski distance generalizes these metrics, offering flexibility through its p parameter (DataLensBlogger). By adjusting the p parameter, the model can be fine-tuned to focus more on smaller differences, when p is lower or give greater weight to larger differences as p increases. In Python's scikit-learn implementation of KNN, Minkowski distances is the default distance metric, when the p parameter is 2, it is equivalent to Euclidean distance. After calculating the distances, the algorithm selects the k-nearest points and in classification tasks, determines the final classification by evaluating how many of these neighbors belong to each class. The choice of metric, the value of 'k', and preprocessing steps like scaling the data significantly impact the performance of a KNN model. When parameters are optimally adjusted and the data is high quality, the algorithm performs exceptionally well.

One outcome is accurately classifying new data points by assigning them to the correct categories based on their nearest neighbors. An accuracy of 90% or higher indicates strong model performance.

Assumption:

One assumption of KNN classification is that similar data points are likely to share the same labels. Essentially, the closer two data points are in the feature space, the more likely they belong to the same class.

Libraries Used:

Python was chosen as the programming language for this project for two key reasons. Firstly, Python has simple and consistent syntax, which facilitates easier coding and debugging. Secondly, Python has a large and active community, providing an abundance of resources and tutorials for support and knowledge sharing.

Several modeling and data analysis libraries were utilized for this project:

Packages and Libraries	**Usage**
pandas	Used to handle and prepare the data.
numpy	For performing numerical operations on arrays.
sklearn.preprocessing import LabelEncoder, StandardScaler	For encoding the data.
sklearn.model_selection import train_test_split, cross_val_score	For splitting the data and identifying the optimal k value.
sklearn.neighbors import KNeighborsClassifier	To perform KNN.

sklearn.metrics import accuracy_score, roc_auc_score	To calculate the accuracy score and Area Under the Curve value for the model.
sklearn.feature_selection import SelectKBest	To select the features with highest p-values.
matplotlib.pyplot	To visualize the KNN k values.

Data Preprocessing Goal:

When utilizing K-Nearest Neighbors (KNN) to predict which patients are at higher risk of readmission, one crucial preprocessing goal is to scale the data within a defined range. This scaling process is vital, as it ensures the features contribute equally to the distance calculations, thereby improving the model's performance (Brown, 2024). By standardizing the data before applying KNN, the accuracy and reliability of the predictions are significantly enhanced. This preprocessing step is key to leveraging the full potential of the KNN algorithm in predicting patient readmission risk.

Variable Identification:

Variable	Data Type	Data Class
TotalCharge	Continuous	Qualitative
Intial_days	Continuous	Qualitative
Children	Continuous	Qualitative

Cleaned Dataset:

Refer to the “CleanedData” csv file for the prepared data set.

CleanedData.csv

Training and Testing Sets:

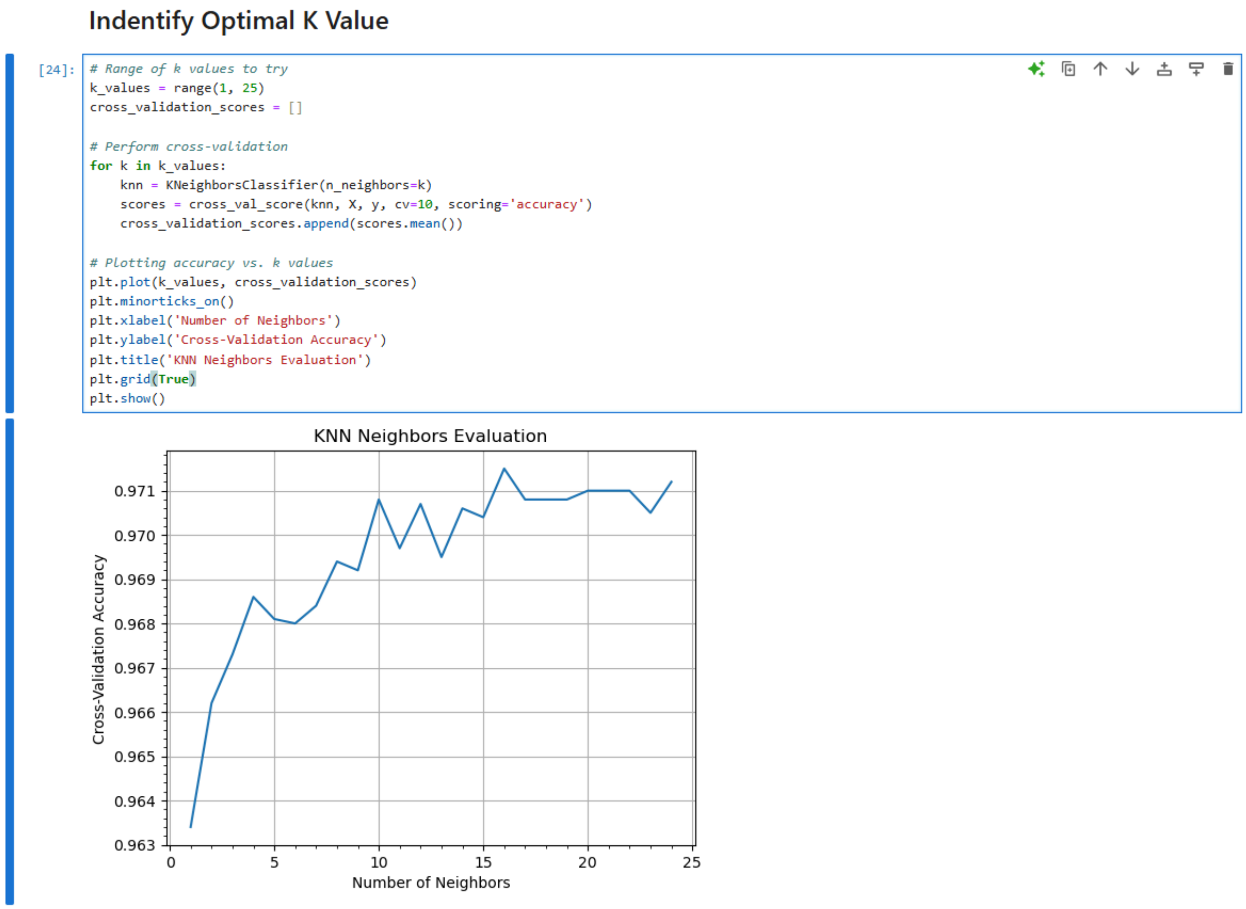
Refer to the “X_test”, “X_train”, “y_test”, and “y_train” csv files.

X_test.csv X_train.csv y_test.csv y_train.csv

Analysis Technique and Calculations:

The analysis technique used to predict readmission for a patient is K-Nearest Neighbor or KNN. KNN is a technique that predicts the values of a data point by examining the ‘k’ closest labeled data points in its vicinity. The new data point is then assigned the value of the most common value among these ‘k’ neighbors. To determine the most optimal value for ‘k’, cross-validation was employed. Cross-validation is a robust evaluation method that assesses the performance of the model across different ‘k’ values by splitting the dataset into multiple subsets and iteratively training and validating the model on these subsets. This ensures a more generalized and reliable model performance measure.

In the analysis, 10-fold cross-validation was used to evaluate the performance of the KNN model across ‘k’ values ranging from 1 to 25. The graph shown below visually represents how the cross-validation accuracy score varies with different ‘k’ values. It was observed that the most optimal ‘k’ value is 16, as it yields the highest cross-validation accuracy score. This indicates that when predicting patient readmission, considering the 10 nearest neighbors provided the most accurate and reliable predictions. This thorough evaluation ensures that the model performs well on unseen data, leading to more accurate predictions and better healthcare outcomes.



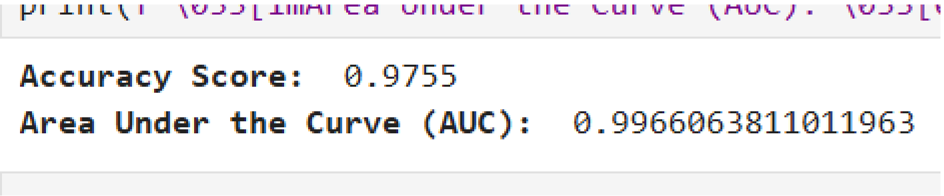
Indentify Optimal K Value

```
# Range of k values to try
k_values = range(1, 25)
cross_validation_scores = []

# Perform cross-validation
for k in k_values:
    knn = KNeighborsClassifier(n_neighbors=k)
    scores = cross_val_score(knn, X, y, cv=10, scoring='accuracy')
    cross_validation_scores.append(scores.mean())

# Plotting accuracy vs. k values
plt.plot(k_values, cross_validation_scores)
plt.minorticks_on()
plt.xlabel('Number of Neighbors')
plt.ylabel('Cross-Validation Accuracy')
plt.title('KNN Neighbors Evaluation')
plt.grid(True)
plt.show()
```

Accuracy and Area Under the Curve (AUC):

The accuracy score is a measure of how well the KNN model correctly predicts the outcome for new, unseen data. It is expressed as a percentage, calculated by dividing the number of correctly predicted observations by the total number of observations in the dataset.

The Area Under the Curve (AUC) is a metric used to evaluate the performance of a binary classification model. Specifically, it measures the area under the Receiver Operating Characteristic (ROC) curve. The ROC curve is a graphical representation of a classifier's performance across all possible threshold values. The ROC curve plots the True Positive Rate (TPR) on the y-axis, and the False Positive Rate (FPR) on the x-axis. AUC values range from 0 to 1, where 1 indicates that the model correctly classifies all positives and negatives, and 0.5 or less indicates the model performs no better than random guessing (GeeksForGeeks, 2025).



```
print(f"\033[1mArea Under the Curve (AUC): \033[
```

```
Accuracy Score:  0.9755
Area Under the Curve (AUC):  0.9966063811011963
```

Results and Implications:

This KNN model for predicting patient readmission has notable implications in the healthcare industry. By identifying patients with a high risk of readmission and enabling target interventions, it can significantly improve the patient care and lead to better health outcomes. The model also plays a vital role in reducing costs for the company by minimizing readmission rates and optimizing resource utilization, such as personnel and equipment. Additionally, it contributes to shaping healthcare policies and protocols, enhancing patient satisfaction and engagement.

In this case, the accuracy score of the model is 97%, meaning the model's predictions are correct 97% of the time, reflecting a high level of predictive reliability. Additionally, the model has an Area Under the Curve (AUC) value of 0.99, which signifies that the model performs exceptionally well, demonstrating its strong capability to distinguish between positive and negative outcomes. The features used in the model, TotalCharge, Initial_days, and Children, are critical factors that capture meaningful patterns related to patient readmission. For example, total charge and initial days may reflect the intensity of the initial hospital stay, while the number of children could provide insights into a patient's support system or external responsibilities, both of which might influence readmission likelihood. These strategically selected features enhance the model's ability to make accurate predictions. The data driven insights derived from the model can inspire further research and refinement of predictive models, providing broader benefits to the healthcare system overall.

Limitation of Method Used:

Data balance is a critical factor in ensuring the effective performance of a KNN model. An imbalanced dataset, where one class significantly outnumbers the other, can lead to biased predictions. In such cases, the model may disproportionately favor the majority class, resulting in poor performance for the minority class. A balanced dataset with an equal distribution of classes allows the model to learn patterns from all classes effectively and reduces the risk of misclassifying the less common class.

For instance, in this case, the dataset consists of 10,000 observations, with 6,331 patients not being readmitted and 3,669 patients being readmitted. This imbalance means the majority class, not readmitted, could dominate the model's predictions, causing it to underperform on the minority class, readmitted. As a result, high-risk patients who have a higher likelihood of being readmitted might not be correctly identified, limiting the effectiveness of the predictions.

Recommendations:

Based on the results of the KNN model, it is recommended that the company develop targeted patient care plans for individuals who are at higher risk of being readmitted to reduce readmission rates. Additionally, the company should consider implementing follow-up care for patients who have had a more intense initial hospital stay and who may have inadequate external support, as these patients are more likely to be readmitted. Lastly, the model can be further enhanced by fine-tuning the algorithm or incorporating additional features to enhance performance.

Code Sources:

Cross_val_score. scikit. (n.d.-b). https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.cross_val_score.html

Model Selection and Evaluation. scikit. (n.d.-d). https://scikit-learn.org/stable/model_selection.html

Sources:

Brown, J. (2024, December 6). *K-nearest neighbors: An in-depth guide*. 33rd Square. https://www.33rdsquare.com/knn-the-distance-based-machine-learning-algorithm/

W3schools.com. Python. (n.d.). https://www.w3schools.com/python/python_ml_cross_validation.asp

Model Selection and Evaluation. scikit. (n.d.-d). https://scikit-learn.org/stable/model_selection.html

GeeksForGeeks. (2025, February 7). *Auc Roc Curve in machine learning*. GeeksforGeeks. https://www.geeksforgeeks.org/auc-roc-curve/

DataLensBlogger. (n.d.). *Mastering K-nearest neighbors (KNN): A 101 guide to this simple yet powerful supervised learning algorithm*. Coding Island. https://codingisland.net/supervised-learning/mastering-k-nearest-neighbors