k-Nearest Neighbours for Stroke Risk Prediction

Abstract

Stroke is a leading cause of death and long-term disability worldwide. Early prediction of stroke risk from routine clinical data could help clinicians prioritise patients for further investigation, but such data are often noisy and strongly imbalanced: only a small minority of patients will have a stroke. In this tutorial-style report, I use the public Stroke Prediction Dataset and the k-Nearest Neighbours (k-NN) algorithm to explore how the choice of the hyperparameter k affects performance. I outline a simple preprocessing pipeline, show descriptive plots of key risk factors, and evaluate k-NN for a range of k values using accuracy, F1-score for the stroke class, confusion matrices, and ROC and precision–recall curves. The results illustrate why accuracy alone is misleading on imbalanced datasets and how larger k values can completely suppress the minority class. I conclude with a brief discussion of limitations, potential improvements, and ethical considerations when applying machine learning to healthcare.

1 Introduction

Stroke occurs when blood flow to part of the brain is interrupted, leading to tissue damage and potentially severe disability or death. Early identification of high-risk patients can improve outcomes by enabling earlier treatment and lifestyle interventions. Machine learning methods can support this process by detecting patterns in routinely collected health data.

The aim of this project is to build an educational tutorial rather than a clinical tool. I use the k-Nearest Neighbours (k-NN) classifier to predict whether a patient has had a stroke based on demographic and clinical features, and I focus on a single question: how does changing k, the number of neighbours considered in the vote, affect performance on a strongly imbalanced dataset?

The primary learning objectives are to:

- Implement a small but non-trivial machine learning pipeline in Python.

- Analyse how the hyperparameter k influences minority-class performance.

- Evaluate models using appropriate metrics and visualisations.

- Reflect briefly on the ethical implications of applying such models in healthcare.

2 Background: k-Nearest Neighbours and Imbalanced Data

k-Nearest Neighbours is a simple supervised learning algorithm originally formalised by Cover and Hart (1967). Given a new sample x, k-NN computes its distance to every training point (here using Euclidean distance), selects the k closest training points, and assigns x to the most common class among those neighbours. Intuitively, this assumes that points that are close in feature space tend to share the same label.

In this tutorial, Euclidean distance is used to measure the similarity between samples, defined as

$$d(x, x_i) = \sqrt{\sum_{j=1}^{p} (x_j - x_{i,j})^2}$$

where x is the test point, x_i is a training point, and p is the number of features. Other distance measures, such as Manhattan, Minkowski, or Hamming distance, may be more appropriate for certain types of data, particularly when features are categorical or not inherently Euclidean.

The choice of k controls a bias–variance trade-off. A very small k (e.g., k = 1) leads to low bias and high variance, as predictions can follow noisy local patterns. Larger k smooths decisions, reducing variance but increasing bias because the classifier averages over more distant points and may ignore local structure. On imbalanced datasets, large k also increases the influence of the majority class, because most neighbours are likely to belong to it. The minority class can therefore be excluded from predictions, even when it is clinically important.

2.1 Why use k-Nearest Neighbours?

k-NN is particularly appealing as a baseline classifier because it is straightforward and intuitive, and it makes no assumptions about the underlying data distribution. It can model highly nonlinear decision boundaries and requires no explicit training phase, as all processing is deferred until prediction. For these reasons, k-NN is frequently used for exploratory analysis, benchmarking, and small to medium-sized datasets where interpretability and simplicity are important.

3 Data and Preprocessing

3.1 Stroke Prediction Dataset

I use the Stroke Prediction Dataset from Kaggle (Fedesoriano, n.d.). Each row describes one individual and includes demographic variables (age, gender, marital status, work type, residence type), clinical risk factors (hypertension, heart disease, smoking status), and continuous measurements (average glucose level, body mass index). The target label stroke is 0 for non-stroke and 1 for stroke.

Only about 5% of individuals experience stroke = 1. Figure 1 shows the class distribution. A trivial classifier that always predicts "no stroke" would therefore achieve about 95% accuracy while completely failing to detect any strokes, which already hints that accuracy alone will not be a suitable evaluation metric.

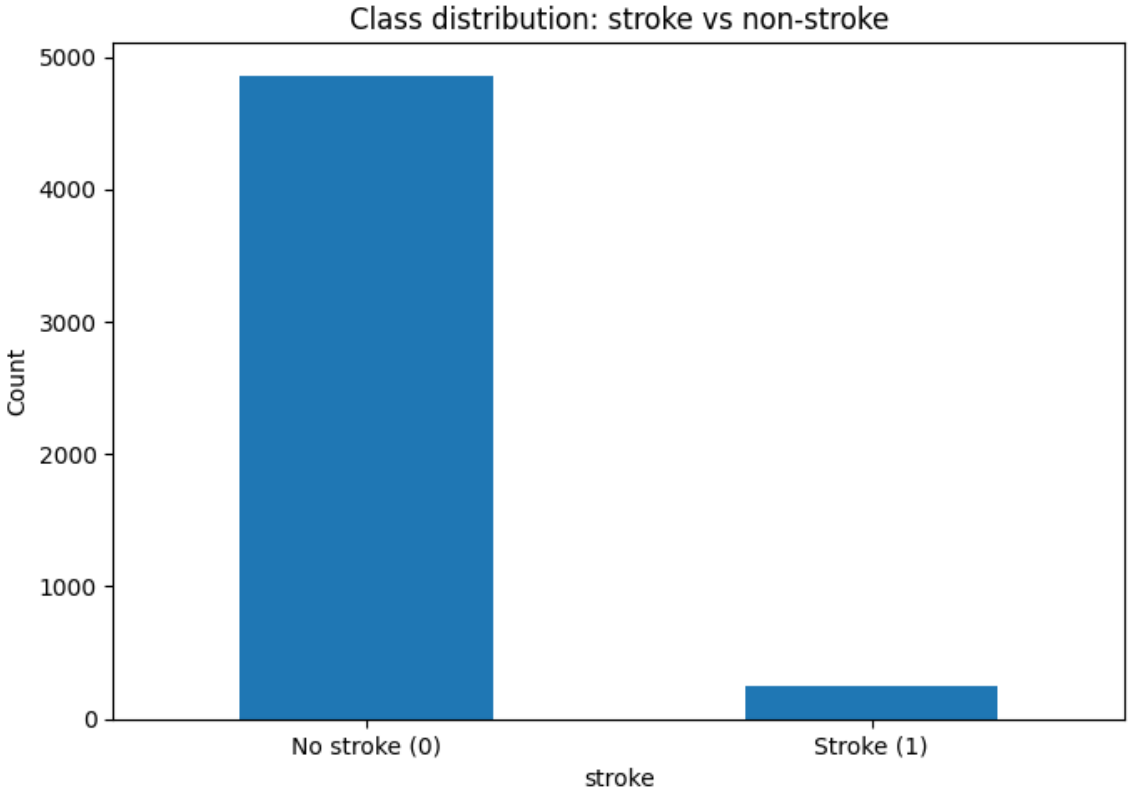

Figure 1: Bar plot of the number of non-stroke (0) and stroke (1) cases, illustrating the strong class imbalance.

To better understand the data, I also examine the distribution of two key numeric risk factors by class. Figure 2 shows histograms of age for stroke vs non-stroke cases, and Figure 3 shows histograms of average glucose level. In both plots, the stroke group is shifted

towards higher values, which matches medical expectations, but there is still substantial overlap between the distributions.

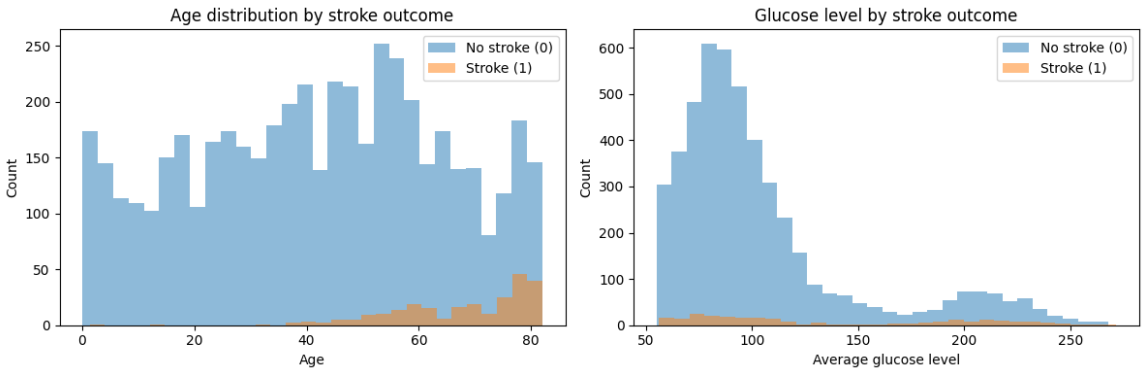

Figure 2: Histogram of age for stroke and non-stroke cases.

Figure 3: Histogram of average glucose level for stroke and non-stroke cases.

3.2 Preprocessing Pipeline

Before fitting k-NN, I apply a simple preprocessing pipeline using scikit-learn (Pedregosa et al., 2011):

- Missing values: the bmi feature contains missing values, which I replace with the median BMI across the dataset.

- Train–test split: I split the data into 80% training and 20% test sets using stratified sampling on stroke, so that the imbalance is preserved in each split.

- Encoding categorical variables: gender, ever_married, work_type, Residence_type and smoking_status are one-hot encoded with OneHotEncoder.

- Feature scaling: numeric features (age, hypertension, heart_disease, avg_glucose_level, bmi) are standardised using StandardScaler so that all contribute comparably to Euclidean distance.

The encoder and scaler are combined in a ColumnTransformer, and this preprocessor is then placed in a Pipeline together with KNeighborsClassifier. Using a pipeline keeps the code concise and makes it easier to reproduce the analysis.

4 Experimental Design

The key experimental question is how the choice of k affects classifier performance on this imbalanced dataset. I therefore evaluate k-NN for a range of k values:

$k \in \{1, 3, 5, 7, 9, 11, 15, 21\}$.

For each k, I train the pipeline on the training data and evaluate it on the held-out test set. To capture both overall and minority-class performance, I compute:

- Test accuracy.

- F1-score for the stroke class (label 1).

I then plot accuracy and F1-score as a function of k (Figures 4 and 5). Finally, I pick k = 1 for a more detailed analysis: I show a confusion matrix heatmap and compute ROC and precision–recall curves based on predicted probabilities (Figures 6–8). These visualisations give complementary views of how well the model separates stroke from non-stroke cases.

5 Results

5.1 Descriptive Patterns

The class distribution in Figure 1 confirms that stroke cases are rare. Figures 2 and 3 highlight that stroke patients tend to be older and to have higher average glucose levels than non-stroke patients, but the large overlap between groups suggests that these features alone cannot perfectly separate the classes. This motivates the use of probabilistic predictions and richer evaluation metrics.

5.2 Effect of k on Accuracy and F1-score

Figure 4 shows test accuracy as a function of k. Accuracy is relatively high for all values (around 0.90–0.95) and tends to increase slightly as k increases. If we looked only at accuracy, we might conclude that large k values, such as k = 15 or k = 21, are best.

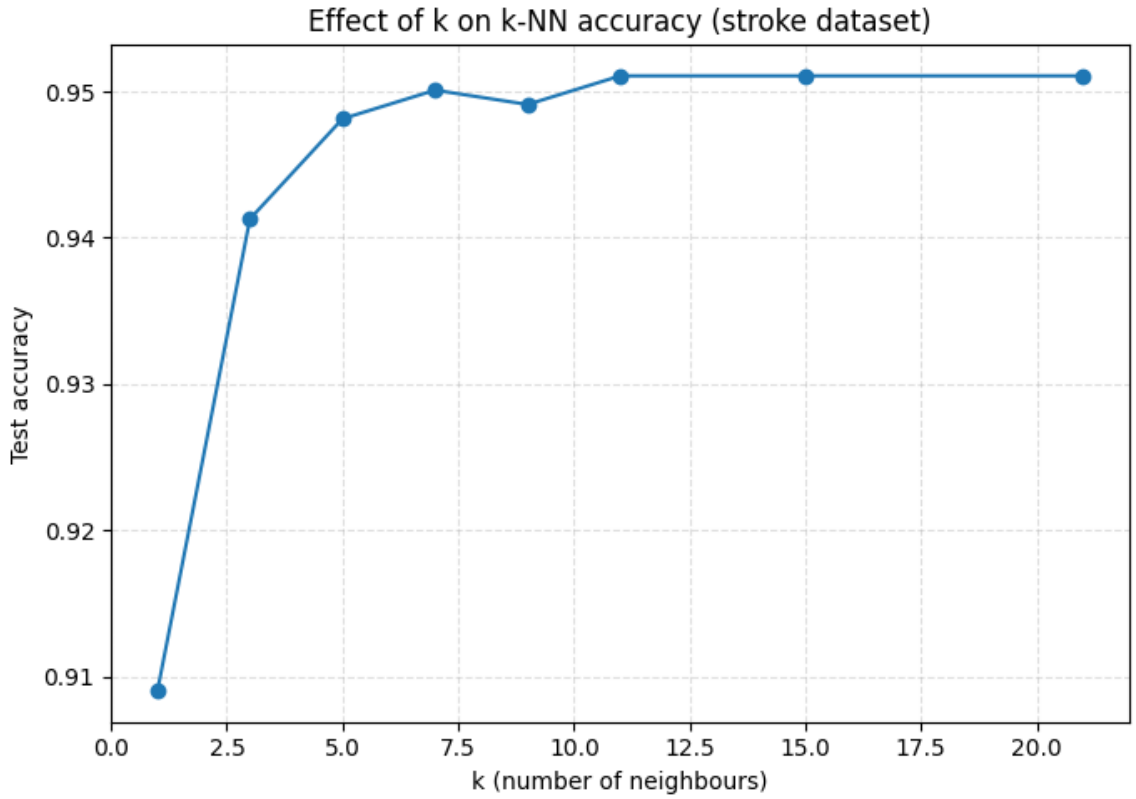

Figure 4: Test accuracy of k-NN as a function of k.

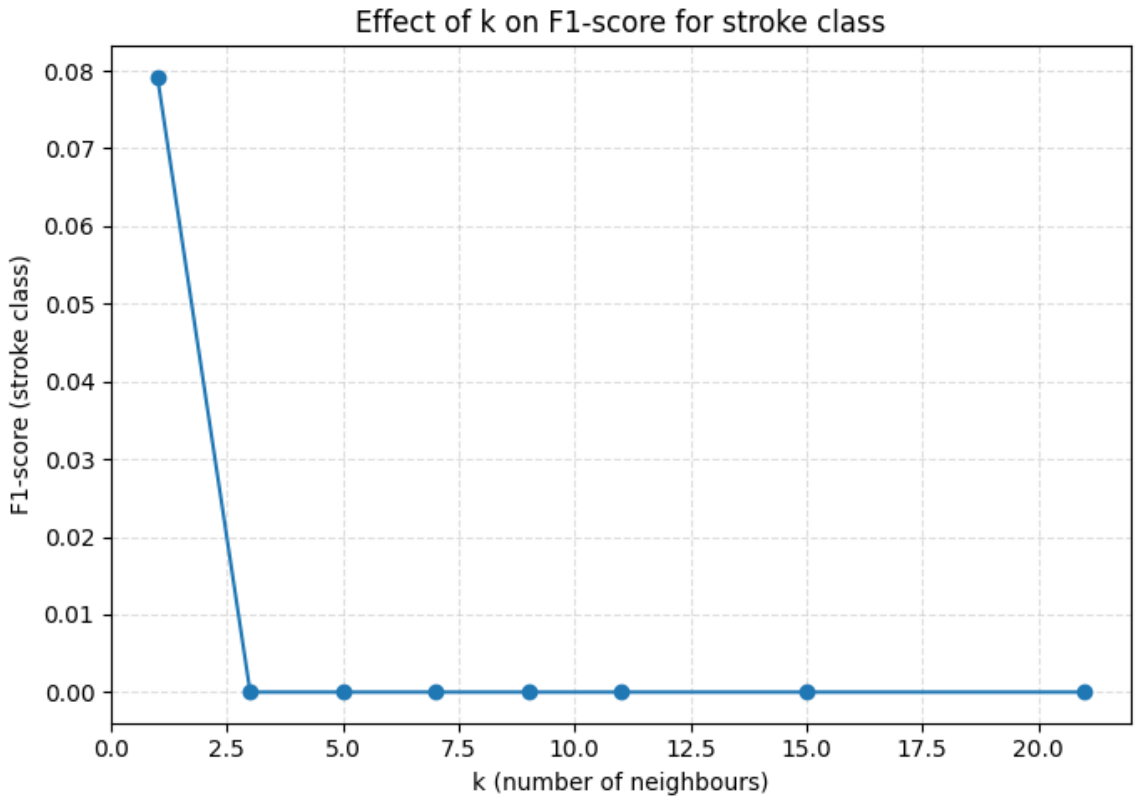

Figure 5, however, shows the F1-score for the stroke class as a function of k. Here we see a very different picture. For k = 1, the F1-score is low but non-zero: the classifier detects a small number of stroke cases but also makes some false positive predictions. For k ≥ 3, the F1-score drops to zero because the classifier almost never predicts the stroke class. In other words, increasing k improves overall accuracy but destroys the ability to detect the minority class.

Figure 5: F1-score for the stroke class (label 1) as a function of k.

This behaviour is a direct consequence of the class imbalance combined with majority voting in k-NN. When k is large, almost all neighbours of any given point are non-stroke patients, so the majority vote is almost always “no stroke”.

5.3 Detailed Evaluation for k = 1

To understand the trade-offs more concretely, I examine k = 1 in detail. Figure 6 shows the confusion matrix as a heatmap. The model correctly classifies most non-stroke patients (true negatives), but it misses many stroke cases (false negatives) and produces some false positive stroke predictions.

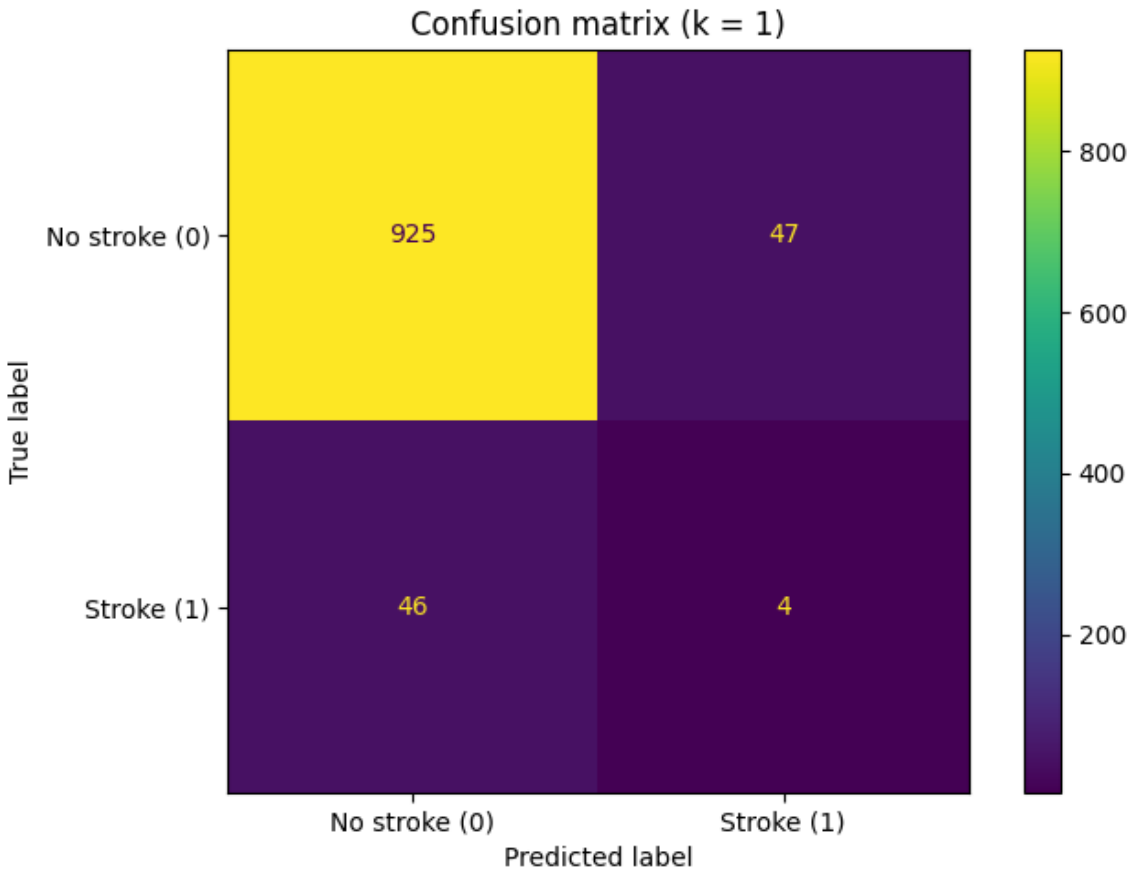

Figure 6: Confusion matrix heatmap for k = 1.

Because stroke is rare, ROC and precision–recall curves provide additional insight. Figure 7 shows the ROC curve for k = 1 based on the predicted probabilities (obtained from the proportion of stroke cases among the nearest neighbours). The area under the ROC curve (ROC AUC) summarises the trade-off between true positive and false positive rates across all possible thresholds. In my run, the ROC AUC is clearly above 0.5 (better than random) but far from perfect, reflecting the difficulty of the task.

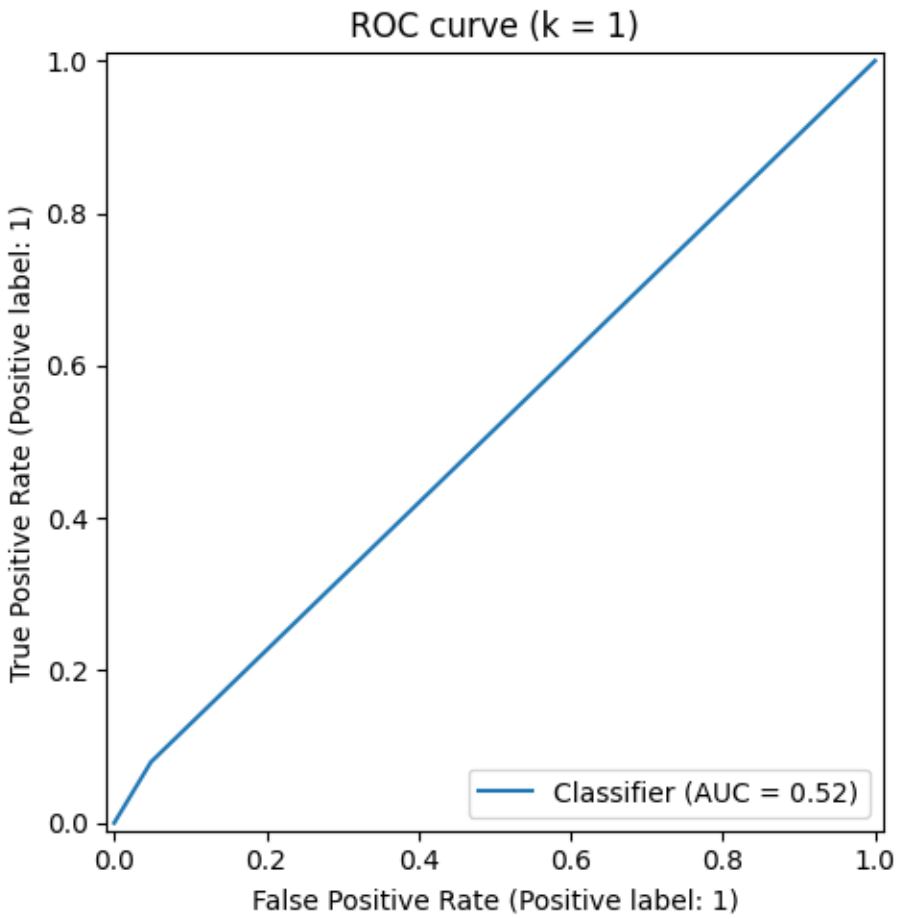

Figure 7: ROC curve for k = 1 with the corresponding AUC.

Figure 8 shows the precision–recall (PR) curve, which is often more informative than the ROC curve in highly imbalanced settings. The area under the PR curve (average precision) indicates how well the model retrieves stroke cases among the patients it flags as high risk. The curve starts at low recall with relatively high precision and drops as recall increases, highlighting the difficulty of maintaining precision when trying to catch more stroke cases.

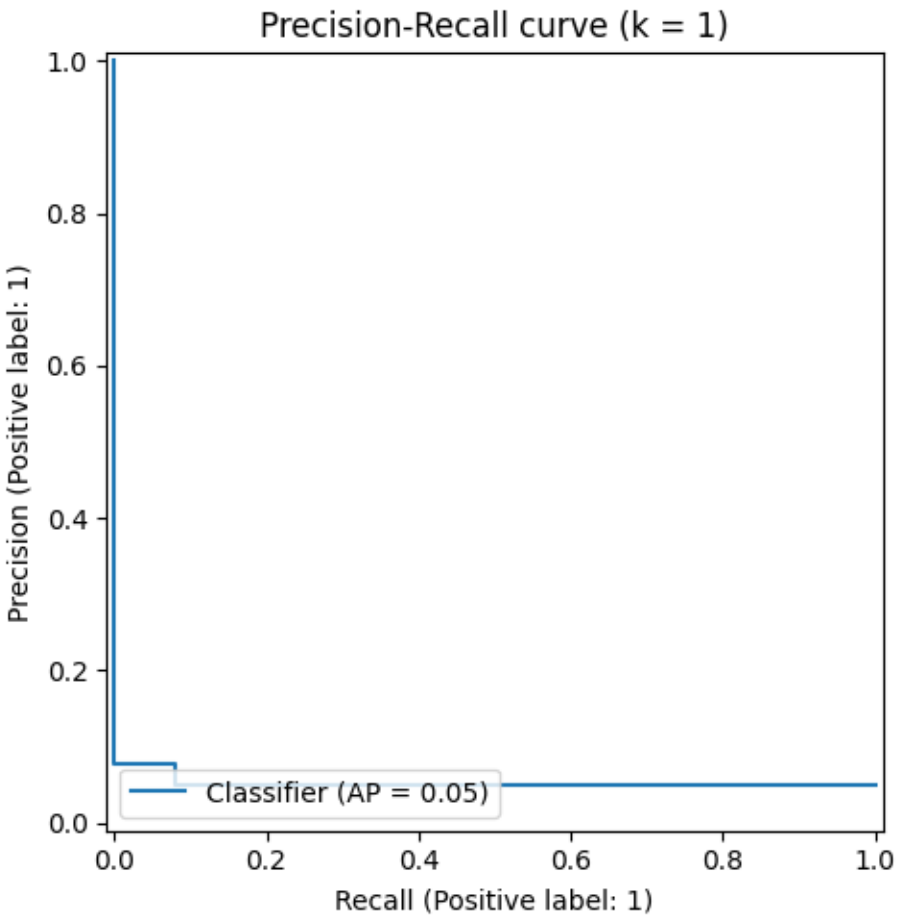

Figure 8: Precision–recall curve for k = 1 and its average precision.

6 Discussion and Ethical Considerations

The experiments demonstrate several key machine learning concepts. First, the choice of k strongly affects performance: small k values allow the model to adapt to local structure and occasionally pick out stroke cases, but at the cost of higher variance and more false positives; large k values produce smooth, stable decision boundaries but can erase the minority class entirely. Second, accuracy alone is misleading on this dataset: a classifier that predicts "no stroke" for everyone would achieve high accuracy but be clinically useless. F1-score for the stroke class, together with confusion matrices and PR curves, gives a more realistic view of performance.

The analysis also highlights limitations of vanilla k-NN on imbalanced medical data (He & Garcia, 2009). With uniform class weights and no resampling, the algorithm is biased towards the majority class. In practical applications, one would consider alternatives such as resampling (e.g., SMOTE), cost-sensitive learning, class-weighted voting, or other models such as logistic regression or tree-based ensembles.

From an ethical perspective, applying machine learning to healthcare raises at least three concerns. Patient safety is paramount: a model that misses most stroke cases could give clinicians or patients a false sense of security, so such models should only be used, if at all, as decision-support tools. Fairness is also important: if particular groups (for example, defined by age, gender, or work type) are underrepresented, the model may perform worse for them and thus widen existing health inequalities. Finally, privacy and data governance must be respected when using medical data, even when datasets are anonymised and publicly available.

7 Conclusion

In this project, I used the Stroke Prediction Dataset and the k-Nearest Neighbours algorithm to explore how the choice of k affects performance on a heavily imbalanced medical classification task. By combining a clear preprocessing pipeline, several complementary evaluation metrics, and a set of informative plots, the tutorial shows that increasing k can slightly improve overall accuracy while completely eliminating predictions of the minority (stroke) class. This highlights that accuracy is an inadequate metric on its own in high-stakes, imbalanced scenarios.

From a learning perspective, this project demonstrates the full workflow of planning and implementing a data analysis task, critically evaluating model behaviour, and reflecting on the ethical implications of applying machine learning to real-world problems. Future work could extend the analysis by exploring resampling techniques, alternative classifiers, and calibration methods that better reflect the uncertainties inherent in medical decision-making.

References

- Cover, T. M., & Hart, P. E. (1967). Nearest neighbour pattern classification. IEEE Transactions on Information Theory, 13(1), 21–27.

- Pedregosa, F. et al. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825–2830.

- Fedesoriano. (n.d.). Stroke Prediction Dataset. Kaggle. https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

- He, H., & Garcia, E. A. (2009). Learning from imbalanced data. IEEE Transactions on Knowledge and Data Engineering, 21(9), 1263–1284.