Appendix:

Comparing Decision Tree and K-Nearest Neighbor for Cerebral Stroke Prediction

1. Glossary:

Accuracy: A classification metric that indicates the percentage of correctly predicted observations.

Bayesian Optimization: An algorithm that minimizes an objective function, by setting optimal hyperparameters in a bounded domain.

Bias: A characteristic of predictions from a model, that are systematically skewed in one direction.

Class imbalance: Describes a data set where one class is more common than another.

Classification: Prediction of a categorical variable.

Correlation Matrix: A table visually showing the correlations between different variables.

Correlation: Metric that describes the strength of the linear relationship between two variables.

Cross-validation: A method to estimate the model's ability to generalize to new data, by repeatedly training and testing it on non-overlapping subsets of the training data.

Decision Tree: A machine learning model that is represented as a sequence of branching statements.

F1-Score: A classification metric, calculated as the harmonic mean of precision and recall.

Feature: The variables that the model uses to make predictions.

Greedy: Refers to the partitioning algorithm of decision trees, where at each step of the tree building the split that improves the prediction power the most is added (instead of looking ahead).

Hyperparameters: Higher level properties of a model.

Instance-based: A type of algorithm that does not try to model an underlying relationship, but simply compares new instances to the training observations.

K-Nearest Neighbor: A machine learning model, where predictions of new observations are based on the values of the k closest observation as defined by a distance function.

Leaves: Also referred to as leaf nodes, represents a class label, where no further splits are applied.

Loss Function: Also called objective function. Is the function a machine learning algorithm tries to minimize during training.

Monotone transformation: A transformation by a strictly increasing function.

Noise: Random fluctuations in the data.

Non-parametric: Non-parametric models, do not make strong assumptions about the form of the distribution of the underlying training data.

One-hot-encoding: Technique to represent categorical variables, where each category of a categorical feature is represented as a binary vector.

Outliers: A data point that is very different from the rest.

Overfitting: Describes a model that fits the training data too well, so that it models the noise in the training data, and does not generalize well to unseen data.

Precision: A classification metric that measures the model's ability to classify the positive class, i.e., the percentage of correctly predicted observations out of all predictions of the positive class.

Preprocessing: Describes the manipulation or deletion of data before it is used in a machine learning algorithm to ensure or improve performance.

Recall: Also called sensitivity. A classification metric that indicates the "sensitivity" of the model to observations of the positive class, i.e., the percentage of correctly predicted positive observations out of all positive observations in the sample

Regression: Prediction of a numerical variable.

Root: In decision tree models, the root marks the beginning of the tree. It is the first split and based on the entire data.

Sensitivity: Also called Recall. A classification metric that indicates the "sensitivity" of the model to observations of the positive class, i.e., the percentage of correctly predicted positive observations out of all positive observations in the sample

SMOTE: Synthetic minority oversampling technique (SMOTE) is an algorithm that creates new synthetic examples of the minority class.

Specificity: A classification metric that measures the model's ability to classify the negative class, i.e., the percentage of correctly predicted observations out of all predictions of the negative class.

Standardization: The process of putting different variables on the same scale.

Stratified cross-validation: Selects cross validation folds, so that the class distribution stays the same across all folds.

Supervised learning: A machine learning problem in which the models tries to learn a function that maps an input to an output based on example labelled input-output pairs.

Target Variable: The variable that the model is trying to predict.

Top-Down: Refers to the partitioning algorithm of decision trees, where the entire data or root is at the top, then splits are successively added.

Variance: Describes the variability of a model's predictions.

White-box: A type of model where the behavior can be explained, and the process that led to a particular prediction is transparent. Opposite of a black-box model.

References:

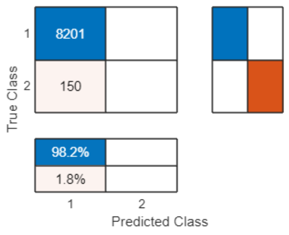
- https://developers.google.com/machine-learning/glossary
- https://ml-cheatsheet.readthedocs.io/en/latest/glossary.html
- https://www.analyticsvidhya.com/glossary-of-common-statistics-and-machine-learning-terms/
- Friedman, J. H. (2017). The elements of statistical learning: Data mining, inference, and prediction.

2. Intermediate / Negative Results:

2) Hyperparameter Tuning Models (CV = k-fold, loss = accuracy)

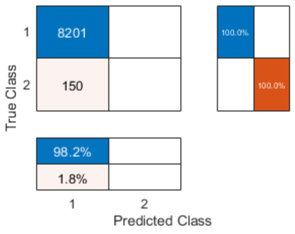
K-fold cross-validation with accuracy loss constitutes the baseline model. With this approach both models drastically overfit the majority class and do not predict the minority class at all.

Confusion Matrix DT



DT	Metric	KNN
98.2%	Accuracy	98.2%
NaN	Precision	NaN
0	Recall	0
1	Specificity	1
0	F2-Measure	0

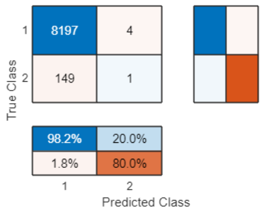
Confusion Matrix KNN



3) Hyperparameter Tuning Models(CV = k-fold, loss = F2)

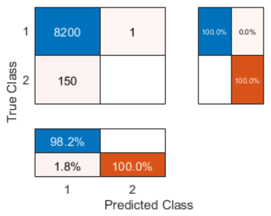
Applying F2-measure instead of accuracy as the objective function has a positive effect, as both final models predict the minority class at least sometimes. However, they still overfit strongly to the majority class.

Confusion Matrix DT



DT	Metric	KNN
98.17%	Accuracy	98.19%
20%	Precision	0.00%
0%	Recall	0%
99.95%	Specificity	99.99%
0.83%	F2-Measure	0%

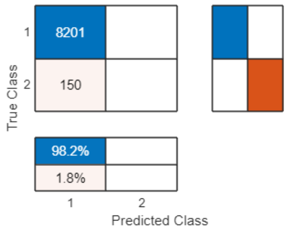
Confusion Matrix KNN



4) Hyperparameter Tuning Models (CV = stratify, loss = accuracy)

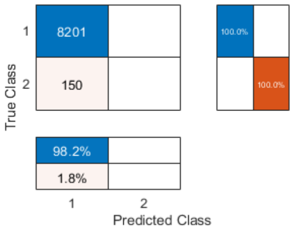
Stratified cross-validation with accuracy loss does not yield any improvements over the baseline model. Again, both models fail to predict the minority class at all.

Confusion Matrix DT



DT	Metric	KNN
98.2%	Accuracy	98.2%
NaN	Precision	NaN
0%	Recall	0%
100%	Specificity	100%
0%	F2-Measure	0%

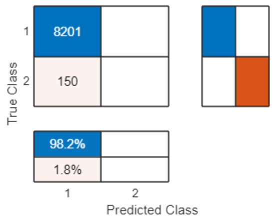
Confusion Matrix KNN



5) Hyperparameter Tuning Models (CV = stratify, loss = F2)

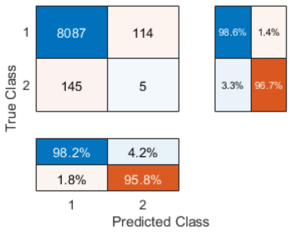
When applying F2 loss to stratified cross validation, DT does not improve its ability to generalize to new data. On the other hand, the final KNN model improves significantly at predicting the minority class.

Confusion Matrix DT



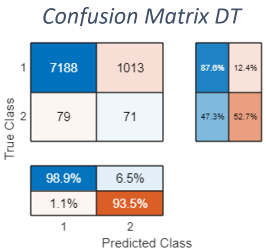
DT	Metric	KNN
98.2%	Accuracy	96.9%
NaN	Precision	4.2%
0%	Recall	3.3%
100%	Specificity	98.6%
0%	F2-Measure	3.5%

Confusion Matrix KNN

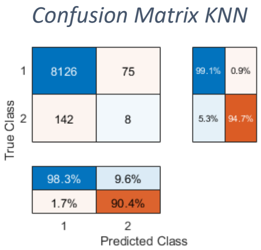


6) Main Results: Hyperparameter Tuning Models (CV = k-fold, loss = accuracy, SMOTE)

Oversampling the minority class with SMOTE, indeed improves the models' ability to learn an effective decision boundary for the minority class. Although the evaluation metrics are identical to those of using the F2 loss in Bayesian optimization, the accuracy loss specification was defined as the "best" model because, as suggested by Occam's razor, given the same power, the simpler model (or in this case, the simpler evaluation metric) is the better one.

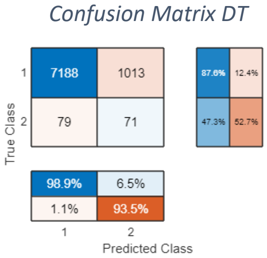


DT	Metric	KNN
86.9%	Accuracy	97.4%
6.6%	Precision	9.6%
47.3%	Recall	5.3%
87.7%	Specificity	99.1%
21.1%	F2-Measure	5.9%

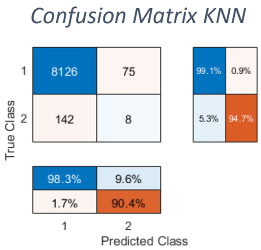


7) Hyperparameter Tuning Models (CV = k-fold, loss = F2, SMOTE)

Using the F2-measure as the objective function in Bayesian optimization yields identical results of the final model, as accuracy loss.



DT	Metric	KNN
86.9%	Accuracy	97.4%
6.6%	Precision	9.6%
47.3%	Recall	5.3%
87.7%	Specificity	99.1%
21.1%	F2-Measure	5.9%



3. Brief Description of Main Implementation Choices

Python pre-processing: The entire data pre-processing was completed in python, as the 'pandas' library allows for simple implementation of all necessary data cleaning steps, like removal of missing values, imputation, and one-hot encoding. The SMOTE oversampling was carried out using the 'imbalanced-learn' package. The training and test data, as well as the oversampled training data, were saved as csv files.

MATLAB modeling: The MATLAB functions 'fitctree()' and 'fitcknn()' were used to train the two models. Both can be easily used in cross-validation by passing a 'cvpartition' object. Here, the 'Stratify' parameter of MATLAB's 'cvpartition()' function determines whether stratified or regular cross-validation is used to create the folds. For hyperparameter tuning the 'bayesopt()' function in combination with a custom loss function was used.

Reproducibility: Following the MATLAB documentation a seed for the MATLAB random number generator is set via the 'rng()' function. This fixes the folds for the 'cvpartition' function. Because exhaustive search is used in the KNN algorithm, no further specifications are needed. For the DT algorithm, the parameter the 'Reproducible' parameter is set to 'true'. Lastly, the bayesopt() parameter 'AcquisitionFunctionName', is set to 'expected-improvement-plus', as the default values that consider expected improvement per second can be influenced by other processes on the computer.