**Prediction of Breast Cancer Survival**

Practical Data Science (COSC1295)

May 2018

Team Members:

Janarth Punniyamoorthy

(s3706154@student.rmit.edu.au)

Charles Galea

(s3688570@student.rmit.edu.au)

Lecturer:

Dr. Yongli Ren

Table of contents

Table of Contents

[Dr. Yongli Ren 1](#_Toc513468644)

# Abstract

# Introduction

Breast cancer is a common cancer accounting for 1 in every 4 cancer diagnoses. It is estimated that in 2018 over 18,000 Australians will be diagnosed with breast cancer and more than 3,000 will die from this disease (<https://breast-cancer.canceraustralia.gov.au/statistics>). Early diagnosis can lead to a dramatic increase in long term survival rates from 56% to more than 86%. It has been reported that shown that cancer that has not spread beyond the breast to the lymph nodes has a better prognosis. Metastatic breast cancers which have spread to the lymph nodes and more distant sites account for 90% of all deaths from this disease (Fouad et al., 2015, Peart, 2017). In this report, we have explored whether patient information, including the patient’s age and number of auxillary nodes detected, could be used to predict whether a patient survived longer than 5 years.

# Methodology

## Data

The Haberman’s Survival dataset (Haberman, 1976) was obtained from the University of California, Irvine Machine Learning Repository (<https://archive.ics.uci.edu/ml/datasets/l>). The dataset contained data from a study conducted between 1958 and 1970 at the University of Chicago’s Billings Hospital on the survival of 306 patients who had undergone surgery for breast cancer. The data comprised: patient age at time of operation, year of operation, the number of auxillary nodes detected and the patient survival status (designated with a value of 1 for patients surviving for 5 years or longer or 2 for patients who died within 5 years) (Table 1).

## Data Analysis

Data manipulation and statistical analyses were performed using the python packages pandas (<https://pandas.pydata.org/>) and NumPy (<http://www.numpy.org/>). The data were initially checked for errors and missing data points. Statistical parameters (mean, standard deviation, minimum, maximum, quantiles) were then determined for the entire patient cohort as well as for each target group (based on survival status).

## Data Visualization

The python plotting package Matplotlib (<https://matplotlib.org/>) was used for data visualization.

## Machine Learning

### *k*-Nearest Neighbors

The *k*NN classifier defines the class of a test instance based on the majority vote of its *k*-nearest neighbors derived from the training data (Friedman et al., 2001). Due to the relatively small size of the dataset *k*NN hyperparameters (*k*, distance metric p and weights) were optimized, to achieve the best balance between the bias and variance of the model, using 5-fold cross validation. The distance metric was determined by the following equation:

Where and .

For 5-fold cross-validation the data was divided into 5 folds and the *k*NN model was applied to make predictions on the 5th segment. Predictions were performed using the equation:

Where is the ith case of the sample and is the prediction of the query point.

The process was then applied successively applied to other 4 segments. The computed errors were then averaged to yield a measure of performance of the model. These steps were then repeated for various parameter values and the value achieving the lowest error (or highest classification accuracy) was then chosen as the optimal value. The *k*NN classifier (*K*NeighborsClassifier) was implemented using the python sklearn package (Pedregosa et al., 2011).

### Data Scaling for *k*NN classifier

The *k*NN classifier was also tested on the dataset following feature scaling to compensate for differences in the measurement scale of the various features. Two scaling methods (standardization and normalization) (Kelleher et al., 2015) were used to rescale each feature prior to applying the *k*NN classifier. The data were standardized to a standard normal distribution with and using the formula:

Where is the standard score (also called the z-score) is the mean (average) and is the standard deviation from the mean. The mean was calculated using the equation:

Where is the total number of observations and is the ith observation. While the standard deviation was derived from the equation:

The data were normalized (also know as Min-Max scaling) by applying the following formula:

Where is the normalized feature value, is the original value, is the minimum value of the feature and is the maximum value of the feature.

### Decision Trees

The decision tree was built by partitioning instances into local subsets by performing binary splits recursively using the most significant variable and value of variable which gives the best homogeneous sets. By default, we used the *Gini method* to determine the homogeneity of sets. Gini method compares homogeneity of sets by computing the sum of squares of probability for success and failure.

To control overfitting of the dataset we put constraints on the decision tree. We controlled the depth, minimum required samples in a node to split, minimum required samples to be in a leaf node, and maximum number of leaf nodes. *GridSearch*, a tool from *Sklearn* library, makes it easy to find which values for the parameters by trying out many combinations of values and returning the values which created the best decision tree based on accuracy of prediction.

## Model Evaluation

Model performance was evaluated using the following parameters:

Where was the number of false positives (negative target features instances incorrectly predicted as positive), was the number of false negatives (positive target features instances incorrectly predicted as negative), was the number of true positives (correctly predicted positive target feature instances) and was the number of true negatives (correctly predicted negative target feature instances).

The average class accuracy was used to determine classification accuracy to account for the imbalance between patient survival status values in the target feature.

Where is the set of levels that the target feature, , can assume; is the size of the set; and refers to the recall achieved by the model for level (Kelleher et al., 2015).

The Area Under Curve (AUC) metric was utilized to compare the performance of the *k*NN and Decision Tree classifiers. AUC is the area under the ROC (Receiver Operator Characteristic) curve, which is a plot of False Positive Rate (FPR) versus True Positive Rate (TPR). Where the TPR is identical to the Recall term outlined above and the FPR is:

# Results

## Data Pre-processing

### Statistical Analysis

Data collected for the Haberman Survival Studies (Haberman, 1976) for 306 breast cancer patients was used to analysis several classification models for the prediction of cancer patient survival. The data consisted of the features: patient age, year of operation, number of positive auxillary nodes detected and survival status (Table 1).

***Table 1.*** Haberman Breast Cancer Dataset

|  |  |
| --- | --- |
| Number of patients | 306 |
| Patient ages | 30-83 |
| Year of operation | 1958-1969 |
| Number of nodes | 0-52 |
| Patient survival status |  |
| Less than 5 years | 81 (73.5%) |
| 5 years or longer | 225 (26.5%) |

We initially examined the data distribution of each feature (Figs. 1 and 2). The distribution of patient ages was normally distributed with a mean of 52.46 and standard deviation of 10.80 (Fig. 1A). The youngest patient was 30 years old while the eldest was 83.

***Table 2.*** Statistical parameters for the entire breast cancer patient cohort.

|  | **Patient Age** | **Year of Operation** | **Number of Nodes** | **Survival Status** |
| --- | --- | --- | --- | --- |
| **count** | 306 | 306 | 306 | 306 |
| **mean** | 52.46 | 62.85 | 4.03 | 1.26 |
| **std** | 10.80 | 3.25 | 7.19 | 0.44 |
| **min** | 30.0 | 58.0 | 0.0 | 1.0 |
| **25%** | 44.0 | 60.0 | 0.0 | 1.0 |
| **50%** | 52.0 | 63.0 | 1.0 | 1.0 |
| **75%** | 60.8 | 65.8 | 4.0 | 2.0 |
| **max** | 83.0 | 69.0 | 52.0 | 2.0 |

Similar numbers of patients (between 25 and 30) aged between 45 and 55 were operated on between 1958 and 1969 (Figs. 1B and 2A). Interesting, in 1959 nearly twice the number of patients underwent an operation with a these being younger cancer patients (~ 40 years old) (Fig. 2A).

A screenshot of a cell phone

Description generated with very high confidence

***Fig. 1.*** (A) Ages, (B) year of operation and (C) number of auxiliary nodes for patients in the breast cancer cohort. The data were plotted as a histogram (top) and boxplot (bottom) to highlight the distribution of the data.

A close up of text on a white background

Description generated with high confidence

***Fig. 2.*** 2D contour plots examining the distribution of breast cancer patient age, year of operation and umber of auxiliary nodes.

No auxiliary nodes were detected in the majority of cancer patients (Figs. 1C and 2B). The remaining patients displayed less than 25 nodes while several patients had 25 or more nodes (Figs. 1C and 2C). Interestingly, survival status did not appear to correlate with the number of auxiliary nodes detected. The range of auxiliary nodes detected was similar for those surviving for 5 years or longer and individuals surviving for less than 5 years (Fig. 2D).

A close up of a map

Description generated with high confidence

***Fig. 3.*** Comparison of the distribution of (A) ages, (B) year of operation and (C) number of auxiliary nodes for breast cancer patients surviving for 5 years or longer (blue) or less than 5 years (orange). The data were plotted as a histogram overlaid with the corresponding density plot (solid line).

The distribution of patient ages and those undergoing an operation each year, based on survival status, were similar (Fig. 3A, 3B and 4A) with comparable mean and standard deviation values (Tables 2 to 4). Comparison of the distribution of the number of auxiliary nodes detected based on survival status indicated that patients surviving for less than 5 years appeared to have more nodes compared to those surviving for longer (Fig. 3C, 4C and Tables 3 and 4).

A picture containing text

Description generated with high confidence

Fig. 4. Comparison of survival status (blue - < 5 years and orange - > 5 years) for (A) patient age, (B) year of operation and (C) number of auxiliary nodes.

***Table 3.*** Statistical parameters for the entire breast cancer patients surviving for 5 years or longer.

|  | **Patient Age** | **Year of Operation** | **Number of Nodes** | **Survival Status** |
| --- | --- | --- | --- | --- |
| **count** | 225 | 225 | 225 | 225 |
| **mean** | 52.02 | 62.86 | 2.79 | 1.00 |
| **std** | 11.01 | 3.22 | 5.87 | 0.00 |
| **min** | 30.0 | 58.0 | 0.0 | 1.0 |
| **25%** | 43.0 | 60.0 | 0.0 | 1.0 |
| **50%** | 52.0 | 63.0 | 0.0 | 1.0 |
| **75%** | 60.0 | 66.0 | 3.0 | 1.0 |
| **max** | 77.000000 | 69.000000 | 46.000000 | 1.0 |

***Table 4.*** Statistical parameters for the entire breast cancer patients surviving for less than 5 years.

|  | **Patient Age** | **Year of Operation** | **Number of Nodes** | **Survival Status** |
| --- | --- | --- | --- | --- |
| **count** | 81 | 81 | 81 | 81 |
| **mean** | 53.68 | 62.83 | 7.46 | 2.00 |
| **std** | 10.17 | 3.34 | 9.18 | 0.00 |
| **min** | 34.0 | 58.0 | 0.0 | 2.0 |
| **25%** | 46.0 | 59.0 | 1.0 | 2.0 |
| **50%** | 53.0 | 63.0 | 4.0 | 2.0 |
| **75%** | 61.0 | 65.0 | 11.0 | 2.0 |
| **max** | 83.0 | 69.0 | 52.0 | 2.0 |

A screenshot of a cell phone

Description generated with high confidence

Fig. 6. Scatter plot for patient age, year of operation and cube root transformed data for the number of auxiliary nodes.

The target feature comprised cancer patients who had survived for 5 years or longer and those surviving for less than 5 years. The target data was slightly imbalanced with more long term survivors (73.5%) compared to those who had succumbed to the disease (26.5%) (Fig. 1).



***Fig. 1.*** Pie chart illustrating the proportion of breast cancer patients surviving for 5 years or longer (blue) or less than 5 years (orange).

### Data processing

### Feature transformation

The histogram plotting the distribution of auxiliary nodes was highly right skewed due to the presence of a number high value outliers (Fig. 1C). Since the *k*NN classifier is a ‘distance-based’ predictor the presence of outliers can potentially have a significant detrimental effect upon its performance. To try to minimize the effect of outliers the data was transformed with a number of different transformation functions (i.e. log10, log2, square root, cubed root) to try to obtain a more normal distribution. The best results were obtained by applying a cubed root function to the dataset which resulted in an improvement in the overall distribution and shape of the plotted data (Fig. 3).

A close up of a map

Description generated with high confidence

Fig. 3. Cubed root transformation of the data for the number of auxiliary feature results in an improved distribution of the data. The highly right skewed data for the number of auxiliary nodes (A) was processed with a cubed root function to obtain a better distribution of the data (B). The histogram is show at top while the corresponding boxplot is on the bottom of the figure. (C) and (D) show the histograms and density plots (solids lines) for breast cancer patients surviving for 5 years or longer (blue) or less than 5 years (orange).

### Feature Scaling

To compensate for differences in the range of measurements used for each feature we applied several different functions to standardization and normalization the data (Figs. 8). This ensured the data being compared had similar ranges while having negligible effect upon the overall distribution of the data (Figs. 9 and 10).

### Feature Scaling

A close up of a map

Description generated with high confidence

Fig. 8. Comparison of the distribution of breast cancer data before (green) and after standardization (red) or normalization (blue) of the data to compensate for differences in the measurement range for each feature. Plots of patient age versus (A) year of operation and (B) number of auxiliary nodes and year of operation versus number of auxiliary nodes.

A close up of a map

Description generated with high confidence

Fig. 9. Comparison of (A) standardized and (B) normalized data for the breast cancer dataset.

A close up of a map

Description generated with very high confidence

Fig. 10. Scatter plots for the (A) original, (B) standardized and (C) normalized breast cancer data.

# Prediction Modelling

## *k*-Nearest Neighbor (*k*NN) Classifier

### Grid Search for Optimizing Hyperparameters

Parameters influencing the performance of the *k*NN classifier were searched simultaneously via a grid search using the GridSearchCV algorithm provided in the python package sklearn (Boschetti and Massaron, 2015). Initially, the whole standardized dataset was randomly divided into two series, a learning and validation series, to train and validate the final models using a ratio of 1:1. Five-fold cross-validation was used within the learning sample, which was randomly divided into five learning partitions of roughly equal size. The model was trained on four of these partitions and performance was estimated using the last one (the test partition). The five estimates of performance were then averaged to provide the cross-validation performance for the model.

The *k*NN algorithm is a non-parametric statistical model based on combinations of the various descriptive features (i.e. patient age, year of operation and number of auxiliary nodes). Model performance was measured by the ability of model to predict class target labels (survival for 5 years or longer versus less than 5 years) pre-defined in the test portion of the dataset.

For each combination of the descriptive features using either the Euclidean or Manhattan distance to measure the distance between nearest neighbours did not have an effect upon the performance of the model (Appendix Tables 1-4). Model performance was optimal for a k (nearest neighbours) value of 2 (accuracy of 83.41%).

The accuracy performance score provides a measure of the ability of the *k*NN classifier to correctly predict target labels. However, this is not the best measure to determine model performance for these studies. Since we were primarily interested in identifying individuals with a low rate of survival rather than individuals with a high chance of long term survival. To gain a more detailed understanding of how the model performed when predicting breast cancer patients with a high risk of succumbing to the disease within 5 years, we examined the confusion matrix for the model while varying each hyperparameter in turn. We also determined various performance scores (precision, recall, F1-score and overall model score) to gain further insight into how the model performed for the prediction of these high-risk patients.

We tested the performance *k*NN model against the original ‘un-processed’, cube root transformed, normalized and standardized datasets (Appendix Tables 5-16). Examination of the confusion matrices for each dataset revealed that the *k*NN classifier performed poorly at predicting high risk breast cancer patients. In the best scenario the classifier could only predict 50% (8 out of 16) of the high-risk patients (Table ).

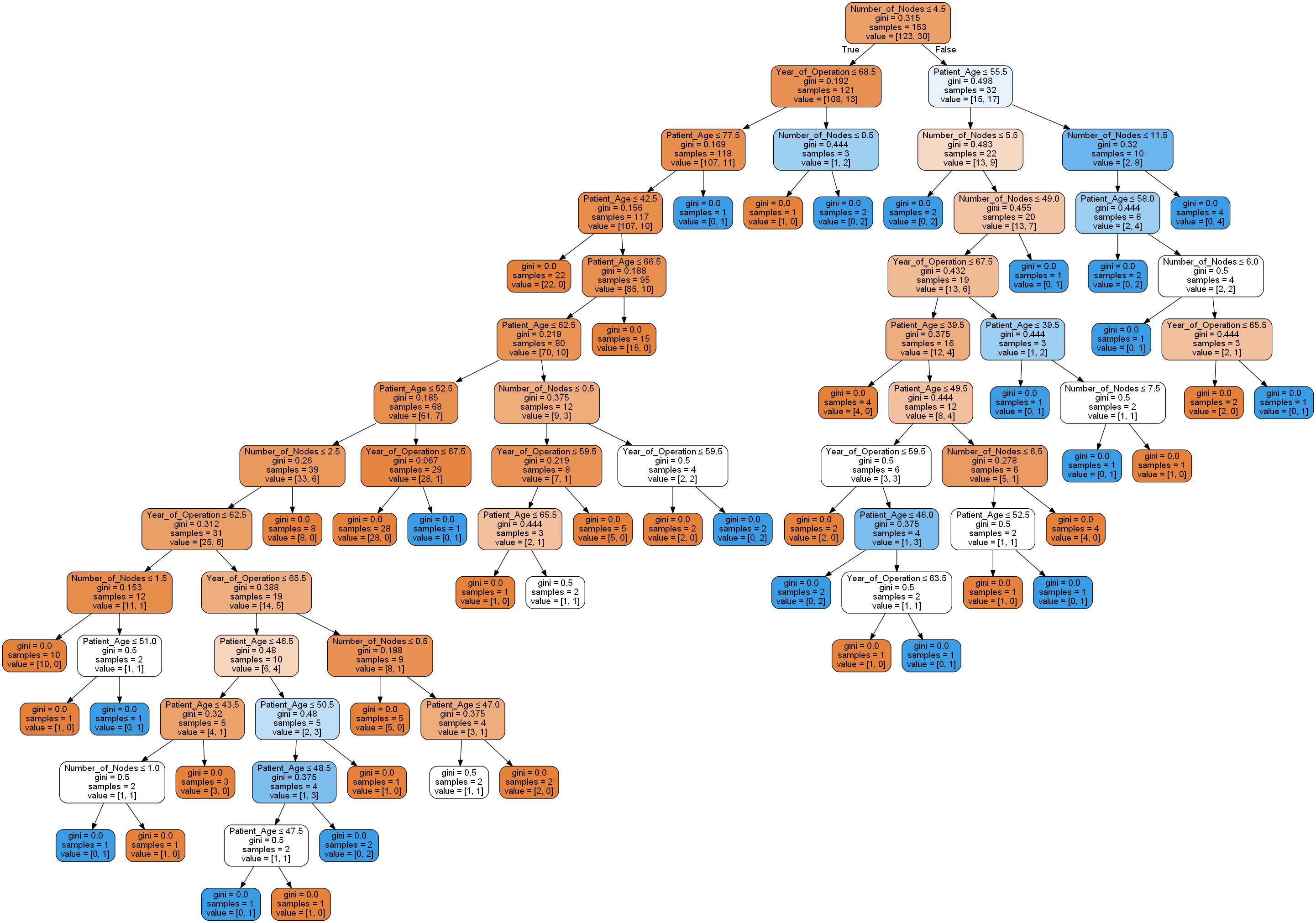
Table Best combinations according to model performance for the prediction of high-risk breast cancer patients.

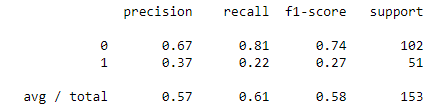
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Dataset | K value | P value | Precision | Recall | F1-score | Model score (SCV) | Prediction of high risk patients  (%) |
| Cube root transformed | 3 | 2 | 0.44 | 0.50 | 0.47 | 0.611 | 50 |
| Cube root transformed | 4 | 2 | 0.55 | 0.50 | 0.50 | 0.618 | 50 |
| Normalized | 1 | 1 | 0.40 | 0.50 | 0.44 | 0.650 | 50 |
| Normalized | 1 | 2 | 0.41 | 0.44 | 0.42 | 0.660 | 44 |
| Standardized | 1 | 2 | 0.41 | 0.44 | 0.42 | 0.660 | 44 |

## Decision Tree Classifier

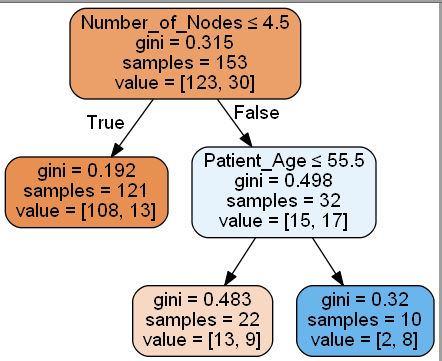
We created two trees using default parameters and parameters given from gridsearchcv to see the effects of overfitting. Grid search was used to find best recall score.

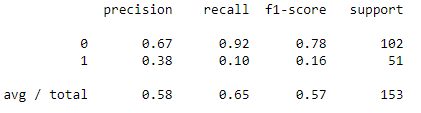
Default parameter Tree:





Optimum Parameter Tree:





It is clear that putting constraints on the decision tree drastically reduces it’s complexity. We can also see that avoiding overfitting resulted in a higher overall recall value. However we would likely be interested in the recall value when target=1 because we want to know which patients are in risk of dying within five years. In which case the optimized tree, with a recall score of 10% for target=1, is worse than the overfitted tree, with a recall score of 22% for target=1.

# Appendix

Table 1. Hyperparameter optimization for nearest neighbours using a grid search approach while keeping the distance metric (i.e. euclidean distance) and weighting (i.e. uniform) constant.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **k value** | **Distance Metric** | **Weights** | **Accuracy**  **(%)** | **Optimal neighbors** | **AUC** |
| 1, 2, 3, 4, 5, 6, 7, 10 | Euclidean | Uniform | 78.17 | 10 | 0.645 ± 0.116 |
| 1, 2, 3, 4, 5, 6, 7 | Euclidean | Uniform | 77.29 | 7 |  |
| 1, 2, 3, 4, 5, 6 | Euclidean | Uniform | 83.41 | 2 |  |
| 1, 2, 3, 4, 5 | Euclidean | Uniform | 83.41 | 2 |  |
| 1, 2, 3, 4 | Euclidean | Uniform | 83.41 | 2 |  |
| 1, 2, 3 | Euclidean | Uniform | 83.41 | 2 |  |

Table 2. Hyperparameter optimization for nearest neighbours using a grid search approach while keeping the distance metric (i.e. manhattan distance) and weighting (i.e. uniform) constant.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **k value** | **Distance Metric** | **Weights** | **Accuracy**  **(%)** | **Optimal neighbors** | **AUC** |
| 1, 2, 3, 4, 5, 6, 7, 10 | Manhattan | Uniform | 78.17 | 10 | 0.645 ± 0.116 |
| 1, 2, 3, 4, 5, 6, 7 | Manhattan | Uniform | 77.29 | 7 |  |
| 1, 2, 3, 4, 5, 6 | Manhattan | Uniform | 83.41 | 2 |  |
| 1, 2, 3, 4, 5 | Manhattan | Uniform | 83.41 | 2 |  |
| 1, 2, 3, 4 | Manhattan | Uniform | 83.41 | 2 |  |
| 1, 2, 3 | Manhattan | Uniform | 83.41 | 2 |  |

Table 3. Hyperparameter optimization for nearest neighbours using a grid search approach while keeping the distance metric (i.e. euclidean distance) and weighting (i.e. distance) constant.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **k value** | **Distance Metric** | **Weights** | **Accuracy**  **(%)** | **Optimal neighbors** | **AUC** |
| 1, 2, 3, 4, 5, 6, 7, 10 | Euclidean | Distance | 78.17 | 10 | 0.645 ± 0.116 |
| 1, 2, 3, 4, 5, 6, 7 | Euclidean | Distance | 77.29 | 7 |  |
| 1, 2, 3, 4, 5, 6 | Euclidean | Distance | 83.41 | 2 |  |
| 1, 2, 3, 4, 5 | Euclidean | Distance | 83.41 | 2 |  |
| 1, 2, 3, 4 | Euclidean | Distance | 83.41 | 2 |  |
| 1, 2, 3 | Euclidean | Distance | 83.41 | 2 |  |

Table 4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **k value** | **Distance Metric** | **Weights** | **Accuracy**  **(%)** | **Optimal neighbors** | **AUC** |
| 1, 2, 3, 4, 5, 6, 7, 10 | Manhattan | Distance | 78.17 | 10 | 0.645 ± 0.116 |
| 1, 2, 3, 4, 5, 6, 7 | Manhattan | Distance | 77.29 | 7 |  |
| 1, 2, 3, 4, 5, 6 | Manhattan | Distance | 83.41 | 2 |  |
| 1, 2, 3, 4, 5 | Manhattan | Distance | 83.41 | 2 |  |
| 1, 2, 3, 4 | Manhattan | Distance | 83.41 | 2 |  |
| 1, 2, 3 | Manhattan | Distance | 83.41 | 2 |  |

Table 5. Confusion Matrix for *k*NN classifier predictions using the original ‘un-processed’ dataset. Correct prediction are highlighted in red while incorrect predictions are black.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | k value |  |  |  |
| p value | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| 1 | 42 3  13 3 | 42 3  14 2 | 43 2  13 3 | 42 3  13 3 | 42 3  13 3 | 43 2  13 3 | 42 3  13 3 | 42 3  13 3 |
| 2 | 43 2  13 3 | 43 2  13 3 | 42 3  13 3 | 41 4  13 3 | 42 3  13 3 | 41 4  13 3 | 42 3  13 3 | 42 3  13 3 |
|  |  |  |  |  |  |  |  |  |

Distance metric (p = 1 manhattan distance; p=2 euclidean distance)

Table 6. *k*NN classifier metrics (p = 1)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | k value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.82  0.29 | 0.75  0.4 | 0.77  0.60 | 0.76  0.50 | 0.76  0.50 | 0.77  0.60 | 0.76  0.50 | 0.76  0.50 |
| Recall | 0.67  0.47 | 0.93  0.12 | 0.96  0.19 | 0.93  0.19 | 0.93  0.19 | 0.96  0.19 | 0.93  0.19 | 0.93  0.19 |
| F1-score | 0.74  0.36 | 0.83  0.19 | 0.85  0.29 | 0.84  0.27 | 0.84  0.27 | 0.85  0.29 | 0.84  0.27 | 0.84  0.27 |
| Model score (5CV) | 0.74828 | 0.69968 | 0.69318 | 0.67705 | 0.70597 | 0.741830 | 0.72221 | 0.72538 |

5CV – 5-fold cross-validation

Table 7. *k*NN classifier metrics (p = 2)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | K value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.77  0.60 | 0.77  0.60 | 0.76  0.50 | 0.76  0.43 | 0.76  0.50 | 0.76  0.43 | 0.76  0.50 | 0.76  0.50 |
| Recall | 0.96  0.19 | 0.96  0.19 | 0.93  0.19 | 0.91  0.19 | 0.93  0.19 | 0.91  019 | 0.93  0.19 | 0.93  0.19 |
| F1-score | 0.85  0.29 | 0.85  0.29 | 0.84  0.27 | 0.83  0.26 | 0.84  0.27 | 0.83  0.26 | 0.84  0.27 | 0.84  0.27 |
| Model score (5CV) | 0.68780 | 0.69935 | 0.68667 | 0.68990 | 0.74828 | 0.74167 | 0.74162 | 0.73834 |

Cube root transform of Number of Nodes feature (survival 2) (weights = distance)

Table 8. Confusion Matrix for *k*NN classifier predictions using the original dataset but where the auxiliary nodes data has been processed with a cubed root function. Correct prediction are highlighted in red while incorrect predictions are black.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | k value |  |  |  |
| p value | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| 1 | 39 9  10 6 | 39 6  10 6 | 40 5  12 4 | 41 4  11 5 | 44 1  15 1 | 45 0  15 1 | 43 2  14 2 | 45 0  15 1 |
| 2 | 40 5  10 6 | 40 5  10 6 | 35 10  8 8 | 37 8  8 8 | 43 2  13 3 | 45 0  14 2 | 45 1  15 1 | 45 0  16 0 |
|  |  |  |  |  |  |  |  |  |

Distance metric (p = 1 manhattan distance; p=2 euclidean distance)

Table 9. *k*NN classifier metrics (p = 1)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | k value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.78  0.40 | 0.80  0.50 | 0.77  0.44 | 0.76  0.56 | 0.75  0.50 | 0.75  1.00 | 0.75  0.50 | 0.75  1.00 |
| Recall | 0.80  0.38 | 0.87  0.38 | 0.89  0.25 | 0.91  0.31 | 0.98  0.06 | 1.00  0.06 | 0.96  0.12 | 1.00  0.06 |
| F1-score | 0.79  0.39 | 0.83  0.43 | 0.82  0.32 | 0.85  0.40 | 0.85  0.11 | 0.86  0.12 | 0.84  0.20 | 0.86  0.12 |
| Model score (5CV) | 0.63723 | 0.67652 | 0.64093 | 0.67023 | 0.68646 | 0.68969 | 0.69603 | 0.71866 |

5CV – 5-fold cross-validation

Table 10. *k*NN classifier metrics (p = 2)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | K value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.80  0.55 | 0.80  0.55 | 0.81  0.44 | 0.82  0.50 | 0.77  0.60 | 0.76  1.00 | 0.75  0.50 | 0.74  0.00 |
| Recall | 0.89  0.38 | 0.89  0.38 | 0.78  0.50 | 0.82  0.50 | 0.96  0.19 | 1.00  0.12 | 0.98  .06 | 1.00  0.00 |
| F1-score | 0.84  0.44 | 0.84  0.44 | 0.80  0.47 | 0.82  0.50 | 0.85  0.29 | 0.87  0.22 | 0.85  0.11 | 0.85  0.00 |
| Model score (5CV) | 0.60471 | 0.61782 | 0.61148 | 0.61798 | 0.6832 | 0.68302 | 0.70549 | 0.70544 |

Normalized features (df\_std) (weights = distance)

Table 11. Confusion Matrix for *k*NN classifier predictions using the normalized dataset. Correct prediction are highlighted in red while incorrect predictions are black.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | k value |  |  |  |
| p value | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| 1 | 36 9  10 6 | 38 7  12 4 | 40 5  11 5 | 38 7  12 4 | 41 4  13 3 | 41 4  13 3 | 41 4  13 3 | 41 4  13 3 |
| 2 | 35 10  9 7 | 40 5  14 2 | 42 3  15 1 | 41 4  14 2 | 43 2  13 3 | 40 5  13 3 | 42 3  13 3 | 41 4  13 3 |
|  |  |  |  |  |  |  |  |  |

Distance metric (p = 1 manhattan distance; p=2 euclidean distance)

Table 12. *k*NN classifier metrics (p = 1)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | k value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.78  0.40 | 0.76  0.36 | 0.78  0.50 | 0.76  0.36 | 0.76  0.43 | 0.76  0.43 | 0.76  0.43 | 0.76  0.43 |
| Recall | 0.80  0.38 | 0.84  0.25 | 0.89  0.31 | 0.84  0.25 | 0.91  0.19 | 0.91  0.19 | 0.91  0.19 | 0.91  0.19 |
| F1-score | 0.79  0.39 | 0.80  0.30 | 0.83  0.38 | 0.80  0.30 | 0.83  0.26 | 0.83  0.26 | 0.83  0.26 | 0.83  0.26 |
| Model score (5CV) | 0.65045 | 0.66028 | 0.66383 | 0.65404 | 0.71253 | 0.69947 | 0.69926 | 0.71560 |

5CV – 5-fold cross-validation

Table 13. *k*NN classifier metrics (p = 2)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | k value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.80  0.41 | 0.74  0.29 | 0.74  0.25 | 0.75  0.33 | 0.77  0.60 | 0.74  0.38 | 0.76  0.50 | 0.76  0.43 |
| Recall | 0.78  0.44 | 0.89  0.12 | 0.93  0.06 | 0.91  0.12 | 0.96  0.19 | 0.89  0.19 | 0.93  0.19 | 0.91  0.19 |
| F1-score | 0.79  0.42 | 0.81  0.17 | 0.82  0.10 | 0.82  0.18 | 0.85  0.29 | 0.82  0.25 | 0.84  0.27 | 0.83  0.26 |
| Model score (5CV) | 0.66039 | 0.66695 | 0.67372 | 0.68339 | 0.71243 | 0.70582 | 0.72216 | 0.74172 |

Standardized features and distance weighting (df\_std) (weights = uniform)

Table 14. Confusion Matrix for *k*NN classifier predictions using the standardized dataset. Correct prediction are highlighted in red while incorrect predictions are black.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  | k value |  |  |  |
| p value | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| 1 | 33 12  8 8 | 41 4  13 3 | 39 6  11 5 | 42 3  15 1 | 41 4  13 3 | 41 4  13 3 | 41 4  13 3 | 44 1  14 2 |
| 2 | 35 10  9 7 | 44 1  15 1 | 41 4  15 1 | 44 1  15 1 | 42 3  13 3 | 43 2  13 3 | 43 2  13 3 | 42 3  14 2 |
|  |  |  |  |  |  |  |  |  |

Distance metric (p = 1 manhattan distance; p=2 euclidean distance)

Table 15. *k*NN classifier metrics (p = 1)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | k value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.80  0.40 | 0.76  0.43 | 0.78  0.45 | 0.74  0.25 | 0.76  0.43 | 0.76  0.43 | 0.76  0.43 | 0.76  0.67 |
| Recall | 0.73  0.50 | 0.91  0.19 | 0.87  0.31 | 0.93  0.06 | 0.91  0.19 | 0.91  0.19 | 0.91  0.19 | 0.98  0.12 |
| F1-score | 0.77  0.44 | 0.83  0.26 | 0.82  0.37 | 0.82  0.10 | 0.83  0.26 | 0.83  0.26 | 0.83  0.26 | 0.85  0.21 |
| Model score (5CV) | 0.65045 | 0.69963 | 0.65420 | 0.70910 | 0.72226 | 0.71238 | 0.69937 | 0.71872 |

5CV – 5-fold cross-validation

Table 16. *k*NN classifier metrics (p = 2)

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | K value |  |  |  |  |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 10 |
| Precision | 0.80  0.41 | 0.754  0.50 | 0.73  0.20 | 0.75  0.50 | 0.76  0.50 | 0.77  0.60 | 0.77  0.60 | 0.75  0.40 |
| Recall | 0.78  0.44 | 0.98  0.06 | 0.91  0.06 | 0.98  0.06 | 0.93  0.19 | 0.96  0.19 | 0.96  0.19 | 0.93  0.12 |
| F1-score | 0.79  0.42 | 0.85  0.11 | 0.81  0.10 | 0.85  0.11 | 0.84  0.27 | 0.85  0.29 | 0.85  0.29 | 0.83  0.19 |
| Model score (5CV) | 0.66039 | 0.70920 | 0.66721 | 0.74166 | 0.71243 | 0.72538 | 0.72872 | 0.73501 |

# Discussion

Decision Tree was useful because the ability to see the visualization of the tree really helps to understand what the classification algorithm is doing. It also helps to decide which parameters to tune to improve the classification. An example would be if the gini index is not improving from one level of the tree to the next, it might be smart to reduce the maximum depth of the tree to avoid growing the tree unnecessarily.

Decision tree method has the benefit of having more variables which can be tuned to fit the data compared to KNN method, but grid search for KNN will be less intensive. It is possible that the recall score for the decision tree could have been improved if more values were added to the parameter grid for the grid search, however it would have exponentially increased the time to compute the optimum parameters.

As a result, k-neighbours classification with a recall score of 50% for target =1 clearly beats decision tree classification with a recall score of 10% for target =1.

# Conclusions

# References

BOSCHETTI , A. & MASSARON, L. 2015. *Python Data Science Essentials,* Birmingham, PACKT publishing.

FOUAD, T. M., KOGAWA, T., LIU, D. D., SHEN, Y., MASUDA, H., EL-ZEIN, R., WOODWARD, W. A., CHAVEZ-MACGREGOR, M., ALVAREZ, R. H., ARUN, B., LUCCI, A., KRISHNAMURTHY, S., BABIERA, G., BUCHHOLZ, T. A., VALERO, V. & UENO, N. T. 2015. Overall survival differences between patients with inflammatory and noninflammatory breast cancer presenting with distant metastasis at diagnosis. *Breast cancer research and treatment,* 152**,** 407-416.

FRIEDMAN, J., HASTIE, T. & TIBSHIRANI, R. 2001. *Elements of Statistical Learning: Data mining, inference, and prediction.,* New York, NY, Springer.

HABERMAN, S. J. 1976. Generalized residuals for log-linear models. *Proceedings of the 9th International Biometrics Conference.* Boston.

KELLEHER, J., MAC NAMEE, B. & D'ARCY, A. 2015. *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*.

PEART, O. 2017. Metastatic Breast Cancer. *Radiologic technology,* 88.

PEDREGOSA, F., VAROQUAUX, G., GRAMFORT, A., MICHEL, V., THIRION, B., GRISEL, O., BLONDEL, M., PRETTENHOFER, P., WEISS, R., DUBOURG, V., VANDERPLAS, J., PASSOS, A., COURNAPEAU, D., BRUCHER, M., PERROT, M. & DUCHESNAY, É. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research,* 12**,** 2825-2830.