Predicting Breast - Cancer Recurrence: Machine Learning Models

Jakob Long

COE 379L

Software Design for Responsible Intelligent Systems

Introduction

The purpose behind this project was to apply our newly learned & refined exploratory data analysis skills to a dataset that needs refining. Then use this newly refined dataset to train & develop different machine learning classification models and find the most optimal model.

The data used for this project was a breast cancer dataset holding key features of a patient that could be used as indicators to aid if a cancer recurrence was likely to occur. These features include age, malignancy of the cancer, the size of the tumor, the number of nodes etc. Thus, this project focused on providing univariate visualizations for the dataset & providing three different machine learning classification models to predict if a patient would be likely to have a cancer recurrence.

The three models chosen to be investigated were a simple Logistic Regression Classifying model, the K-Nearest-Neighbors classifying model, and the Decision-Tree Classifier. For all three models' analysis of the statistics including, F1 Score, accuracy, recall, precision were all investigated and tested as a way to validate the models.

Data Preparation

The data has 286 entries, with 10 variable columns. The data originally held all object datatypes, aside from the malignancy variable, which was an integer datatype. The variables being:
class (recurrence or no recurrence of breast cancer), age, menopause stage, tumor size, invasive nodes count, node caps count, degree of malignancy, breast (left/right), breast quadrant, and irradiation (yes/no).
What this meant was that there was a lot of datatype conversion that needed to occur, to have our dataset primed for classification modeling. Thus to prepare, the following steps were taken:

1. **Finding & Correcting 'Bad' Values:** While there were no true missing values within the dataset, some of the columns held '?' values. This of course was bad. Rather than

dropping the columns that held these values, I replaced them with the most common value for that column, or the mode.

2. **Datatype Conversion:** Converting the object datatypes to Booleans, or even integers was something that needed to be done as well. Since the object datatypes do not work well with classification models.
    a. **One-Hot Encoding**: For the categorical variables I used one-hot encoding as way to convert these objects into Boolean variables that could be used for modeling. Such as the menopause & breast quadrant variables
    b. **Ordinal Encoding**: For the numeric variables that were binned together, I used a hierarchical system that corresponds to each bin value. This is shown in the jupyter notebook.
    c. **Binary Encoding**: For simple variables such as breast & Irradiated, I used binary encoding of just simple 0 & 1 to indicate left or right for the breast variable, and yes or no for Irradiated.
3. **Removal of Duplicates:** Following tidying all of the data I ensured that duplicates were removed if present, of which I found there was 14 duplicate rows.

The data has now been cleaned and prepared, I visualized all variables into count plots to demonstrate and show exactly how the data was broken up.

Model Fitting & Learning

To begin I used the Scikit learn library to develop classification models. The objective of these models was to predict the recurrence of breast cancer for patients, using the other variables that have been tidied within our dataframe. The procedure is the following:

1. **Data Splitting:** First the target variable “mpg”, was isolated and put into variable Y, with the rest of the columns going into variable X. The data was then split using scikit learns ‘train_test_split’ function, with 70% of the data allocated to training & the remining 30% for testing.
2. **Model Training:** The training data was fed into the classification models. The model was then fit to the training data, of which the model represented can be found in the conjoining notebook.
    a. **Parameter-Grid:** For the KNN (K-Nearest-Neighbors), and Decision Tree models, a hyperparameter grid was used to test many different models and identify the best one. This process allowed for the fine tuning of hyperparameters in order to find the most optimal model, for each algorithm.
    b. **Grid Search Cross Validation:** For the KNN and Decision Tree models in addition to using this hyperparameter grid a 5-fold cross validation was used, where the scoring was based on the recall metric, which is our most important metric for the problem being predicted.

Model Analysis

Ensuring proper evaluation of the performance for each of our models, the F1 score, precision, accuracy, and recall performance scores were all used. However, due to the nature of the problem being investigated the recall metric is the most important for each model. The number of false negatives should be minimized to reduce risk for patients that could be affected. False positives are less important, as further testing would identify that the patient is truly fine, whereas a false negative could quite literally be a fatal mistake for an individual. Thus, it was concluded that the model needs to be more accurate in predicting true negatives, as this would decrease the number of false ones.

The following are table of each model's overall performance on the Test & Train data splits.

Decision Tree Classifier Metrics

Decision Tree	Accuracy	Recall	Precision	F1-Score
Test Data	0.71	0.71	0.71	0.71
Train Data	0.72	0.72	0.72	0.72

K-Nearest Neighbor Classifier Metrics

K-Nearest Neighbor	Accuracy	Recall	Precision	F1-Score
Test Data	0.72	0.72	0.69	0.70
Train Data	0.78	0.78	0.77	0.76

Logistic Regression Classifier Metrics

Logistic Regression	Accuracy	Recall	Precision	F1-Score
Test Data	0.54	0.54	0.71	0.54
Train Data	0.56	0.56	0.72	0.57

From the above, we can see the KNN model had the strongest performance in the recall metric as compared to our other two models. However, the decision tree performed nearly identically as well as our KNN model, with a score of 0.71 compared to 0.72 on the test data. Indicating that both models score about a 70% when identifying true positives, meaning there are quite a few false negatives detected. As for the other metrics, the two models perform nearly identically with: 0.71 to 0.72 for accuracy, 0.71 and 0.69 for precision, and lastly 0.71 and 0.70 for F1-Score.

As the logistic regression model failed quite significantly with a low accuracy, F1-Score and recall value, but with a precision score of 0.71 that competes evenly with our Decision Tree model.

Thus, our K-Nearest Neighbor (KNN) classifier is the most recommended model to predict recurrence in breast cancer. However, with the recall value being only 0.72, or having only a

72% likelihood of predicting a true positive, thus implying false negatives are likely to occur. It's recommended that this model is only used as a tool to help decide if a patient should need further testing, and not be the deciding factor. As the goal of these models is to predict recurrence of breast cancer, the models need to be incentivized to keep the number of false negatives minimal or classifying someone as non-recurrent when they in reality do have recurrent breast cancer.

Resources

1. Data: https://raw.githubusercontent.com/joestubbs/coe379Lsp24/master/datasets/unit02/project2.data
2. Decision Tree Hyper-Parameterization Help: https://plainenglish.io/blog/hyperparameter-tuning-of-decision-tree-classifier-using-gridsearchcv-2a6ebcaffeda