COMPENG 4SL4 Assignment 3 Report

Raeed Hassan
hassam41
400188200

November 18, 2022

Data Set Loading and Splitting

The data set was loaded using `sklearn.datasets.load_breast_cancer` and split into a training set (containing 80% of the data sample points), and a test set (containing the other 20% of the data sample points). The `sklearn.model_selection.train_test_split` function was used to perform the split, using the last 4 numbers of my student ID (8200) as the random state for the function.

K-Fold Cross-Validation Setup

For K-fold cross-validation in this lab, the value of K was set to 5. Arrays containing the training and test indexes for each fold were created using `sklearn.model_selection.KFold`. These indices are used for all K-Fold cross-validation done throughout the lab. These indices are used for row indexing to generate the feature and target matrices before doing regression.

Feature Normalization

Feature normalization was performed using the scikit-learn StandardScaler code provided in the lab. The training and test feature data sets were both normalized prior to doing any training, and the same normalized training and test sets are used for all models.

Logistic Regression

Logistic regression was implemented with a logistic regression classifier. The probabilities for the positive class are were calculated.

The thresholds used for the probability of the positive class (malignant) were between 0.05–0.95 with 0.05 increments, with the precision, recall and F1 score calculated at each threshold. The PR curve can be seen in Figure 1. The test error (misclassification error) for each threshold was also calculated. The best performing thresholds (in terms of best F1 score and lowest test score) were when the threshold was between 0.45 and 0.05.

Logistic Regression with scikit-learn

Logistic regression was performed using the `LogisticRegression` classifier implemented in `sklearn.linear_model`. A logistic regression classifier and created and fit to the training data. The probabilities for each class are obtained using the `predict_proba` function.

The same thresholds as the previous part are used to compute the precision and recall with the training data (0.05–0.95 with 0.05 increments), with the precision, recall and F1 score calculated at each threshold. The PR curve can be seen in Figure 2. The test error (misclassification error) for each threshold was also calculated. The best performing thresholds (in terms of best F1 score and lowest test score) were when the threshold was between 0.45 and 0.55.

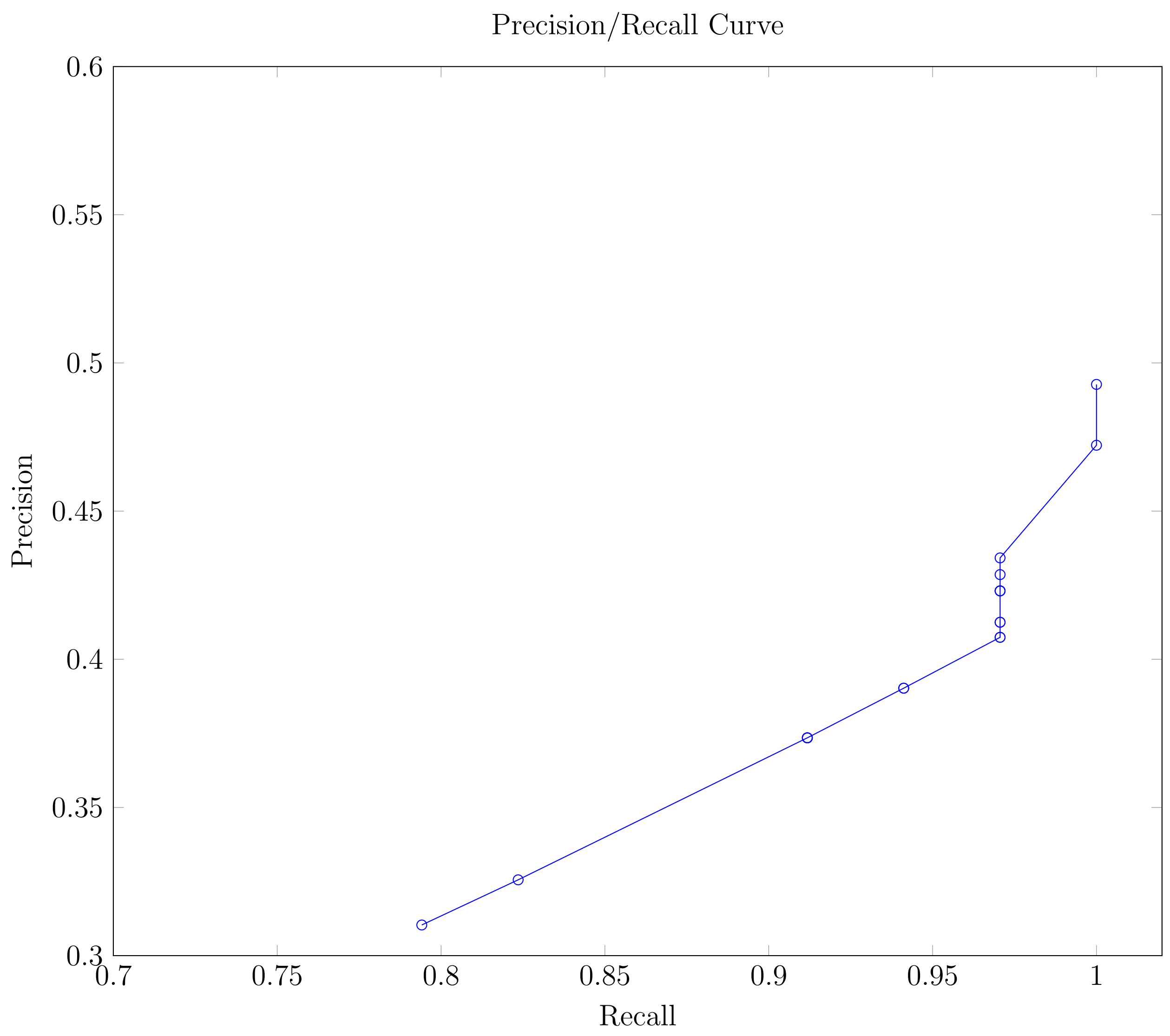

Figure 1: Precision/Recall Curve for Logistic Regression

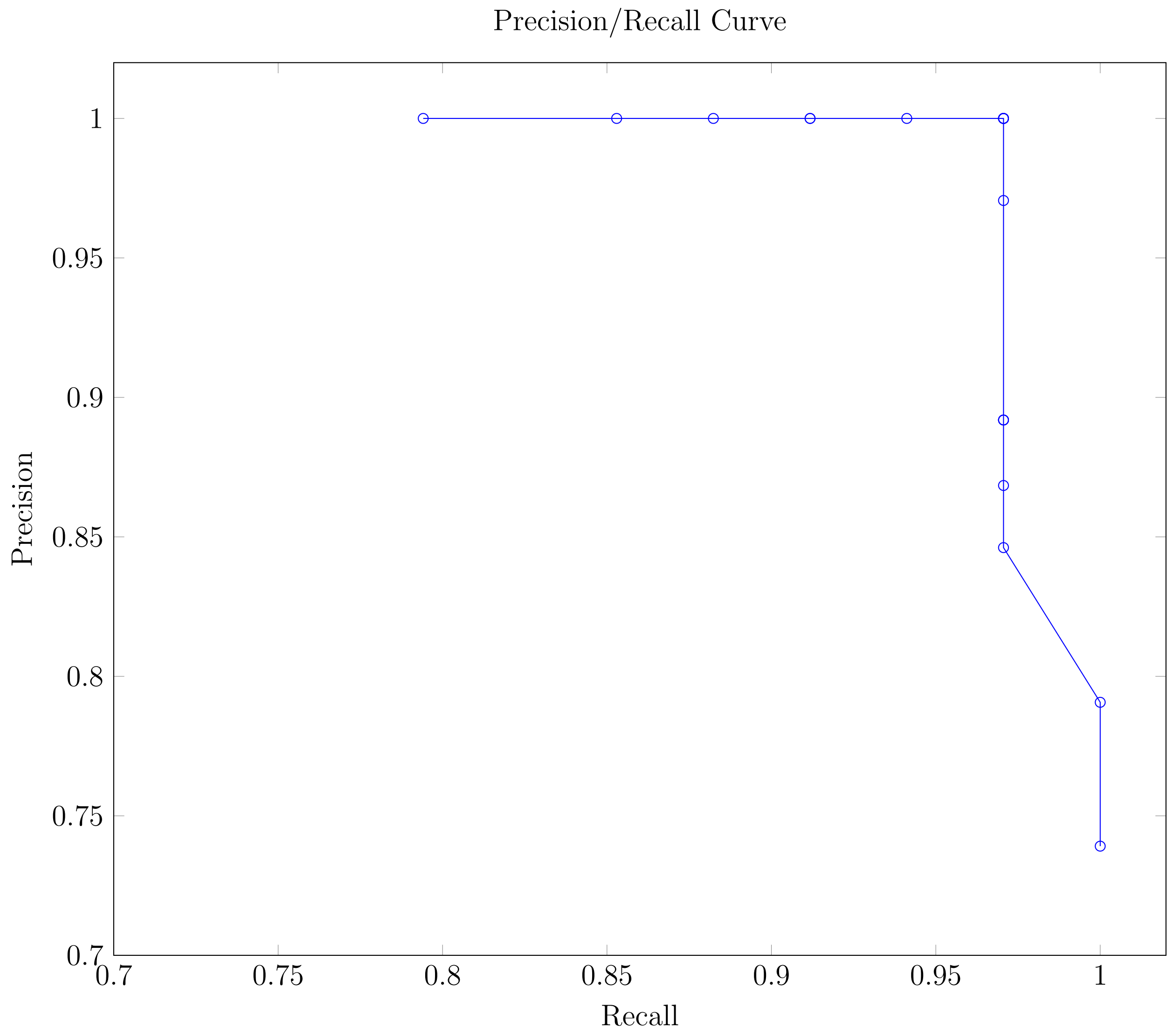

Figure 2: Precision/Recall Curve for scikit-learn Logistic Regression

Logistic Regression Comparison

The two models produced similar misclassification rates and recall, the precision with my own model was significantly worse, leading to a significantly worse F1 score.

k-Nearest Neighbour Classifier with K-fold cross-validation

k-Nearest neighbour classifier for each $k = 1, 2, 3, 4, 5$ were implemented with K-fold cross-validation. The K-fold cross-validation with $K = 5$ was implemented using the scikit-learn function `sklearn.model_selection.KFold` which provides the training and test indices for each fold. The cross-validation error (misclassification error) for each k was calculated, and $k = 3$ was determined to be the best k as it produced the lowest misclassification error. The cross-validation errors are shown in Table 1. The test error (misclassification error) with $k = 3$ was found to be 0.05263157894736842.

There are two situations where ties might occur. When two or more training examples are at the same distance from the current example, the first n training examples that appear in the matrix within the k-nearest neighbour cutoff. When half of the neighbours are each split between the classes twos, we resolve the tie by predicting malignant, as a malignant case predicted as benign is a worse outcome than predicting a benign case as malignant.

Table 1: Cross-Validation Error for k = 1–5 for k-Nearest Neighbour Classifier

k	Cross-Validation Error
1	0.04835164835164835
2	0.039560439560439566
3	0.03296703296703297
4	0.03736263736263736
5	0.035164835164835165

k-Nearest Neighbour Classifier with scikit-learn with K-fold cross-validation

k-Nearest neighbour classifier for each $k = 1, 2, 3, 4, 5$ were implemented with K-fold cross-validation and `sklearn.neighbors.KNeighborsClassifier` from scikit-learn. Tie-handling is also handled by scikit-learn's implementation. The K-fold cross-validation with $K = 5$ was implemented in the same way as the previous part. The cross-validation error (misclassification error) for each k was calculated, and $k = 3$ was determined to be the best k as it produced the lowest misclassification error. The cross-validation errors are shown in Table 2. The test error (misclassification error) with $k = 3$ was found to be 0.05263157894736842. The cross-validation errors for all k and test error with best k were all found to be the same as our own implementation of the k-nearest neighbour classifier.

k-Nearest Neighbour Classifier Comparison

The two implementations of the k-nearest neighbour classifier had identical performance with the Wisconsin breast cancer data set. The predictions between the two models are identical, and the cross-validation errors and test errors in both cases are also identical.

Table 2: Cross-Validation Error for k = 1–5 for k-Nearest Neighbour Classifier

k	Cross-Validation Error
1	0.04835164835164835
2	0.039560439560439566
3	0.03296703296703297
4	0.03736263736263736
5	0.035164835164835165

Best Model

For logistic regression, the cases where the threshold is 0.5 is used for comparison. The misclassification rates and F1 scores are shown in Table 3. The best model is the logistic regression implementation by scikit-learn, which produces the lowest F1 score and misclassification error.

The misclassification rates between both logistic regression models is the same, but the F1 score with my own implementation is significantly lower. The misclassification rates and F1 scores are identical between both k-nearest neighbour models. In both cases, the there is no winner with the misclassification rate as a metric as both comparisons have the same misclassification rates between the models. Similar, with k-nearest neighbour models there is no winner for F1 scores as they have the same misclassification rate. However, with logistic regression, the scikit-learn model significantly wins with a much higher F1 score than all other models.

Table 3: Metric Scores for All Models

Model	Misclassification Rate	F1 score
Logistic Regression	0.008771929824561403	0.5739130434782609
Logistic Regression (scikit-learn)	0.008771929824561403	0.9850746268656716
k-Nearest Neighbour	0.05263157894736842	0.5172413793103449
k-Nearest Neighbour (scikit-learn)	0.05263157894736842	0.5172413793103449