COE 379l

Project 1: Breast Cancer Classification

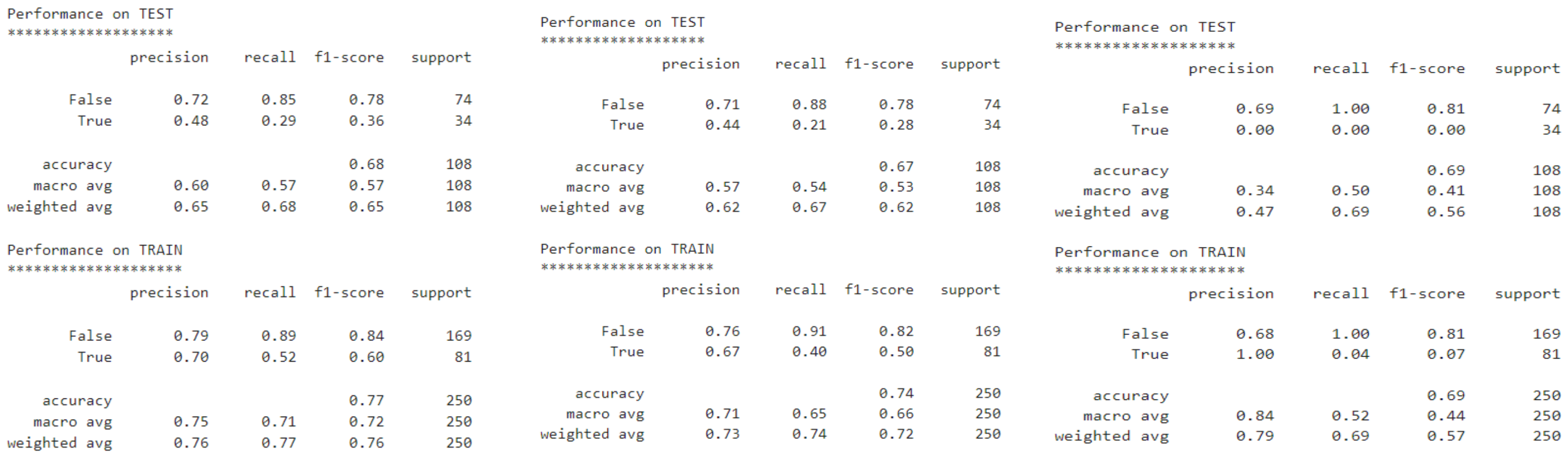
By: Jeet Patel EID: jcp4345

Data Preparation

Before changing and playing with the dataset it is important to analyze it so we know what we're working with. First I analyzed the shape of the dataset using the .shape command finding that the dataset is 386 rows by 10 columns, which using the .size command confirmed this as the datasets size is 3860 inputs. I then looked at the data itself using the .info() function which showed me the column names, null count, and data types of each column and more interestingly found that there are 2 Null values and all the data types are objects other than one which is int64. I then used the .head() function to see the actual inputs of the dataset which allowed me to see that all of the object datatypes needed to be converted to category data types. Investigating the dataset further i found that there were 11 duplicate rows within the dataset which were removed from the dataset and that there were 2 missing (Null) values one in "tumor-size" and one in "inv-nodes" which I replaced with the mode of the data since both were a range of values. Furthermore, in the columns "node-caps" and "breast-quad" there were values of "?" and "*" which were also removed since there was no way to assign a value to each category correctly. Finally I performed one-hot encoding on the nine category data types to make them all bools due to their categorical nature. Doing this makes sure for the data is clean, consistent, and suitable for analysis and our machine learning models.

Model Analysis

For each of the three models that were trained (K-Nearest Neighbor Classifier, K-Nearest Neighbor Classifier using Grid search CV, and Linear classification) I used the same X_test, X_train, y_test, and y_train data to ensure the best comparison of the models this included using the same random state which was set to 0, the same split which was 70/30, and the same feature being predicted which was "class_recurrence-events." Each model was also optimized for recall to minimize false negatives, If we have a large amount of false negatives than people who are at risk of recurrence will be told they won't have one when they are likely to which should be minimized. For the setup differences for each method: for K-Nearest Neighbor we set the n_neighbors parameter to 5 and used knn.fit to train the model, however when we used gridsearch we found that the best n_neighbors value was 3, when searching from 1 to 100, to optimize recall when using gridsearch.fit to train the model, and finally for linear classification we used clf with the loss set to “perceptron” and also set it at random state 0 with an alpha of 0.01 finally using clf.fit to train the model.



```
Performance on TEST
*******************
              precision    recall  f1-score   support

       False       0.72      0.85      0.78        74
        True       0.48      0.29      0.36        34

    accuracy                           0.68       108
   macro avg       0.60      0.57      0.57       108
weighted avg       0.65      0.68      0.65       108

Performance on TRAIN
********************
              precision    recall  f1-score   support

       False       0.79      0.89      0.84       169
        True       0.70      0.52      0.60        81

    accuracy                           0.77       250
   macro avg       0.75      0.71      0.72       250
weighted avg       0.76      0.77      0.76       250
```

```
Performance on TEST
*******************
              precision    recall  f1-score   support

       False       0.71      0.88      0.78        74
        True       0.44      0.21      0.28        34

    accuracy                           0.67       108
   macro avg       0.57      0.54      0.53       108
weighted avg       0.62      0.67      0.62       108

Performance on TRAIN
********************
              precision    recall  f1-score   support

       False       0.76      0.91      0.82       169
        True       0.67      0.40      0.50        81

    accuracy                           0.74       250
   macro avg       0.71      0.65      0.66       250
weighted avg       0.73      0.74      0.72       250
```

```
Performance on TEST
*******************
              precision    recall  f1-score   support

       False       0.69      1.00      0.81        74
        True       0.00      0.00      0.00        34

    accuracy                           0.69       108
   macro avg       0.34      0.50      0.41       108
weighted avg       0.47      0.69      0.56       108

Performance on TRAIN
********************
              precision    recall  f1-score   support

       False       0.68      1.00      0.81       169
        True       1.00      0.04      0.07        81

    accuracy                           0.69       250
   macro avg       0.84      0.52      0.44       250
weighted avg       0.79      0.69      0.57       250
```

Figure 1: Results from KNN (Middle), KNN Gridsearch (Left), and Linear Classification (Right)

K-Nearest Neighbor Classifier predicts the class of a data point by finding the majority class among its 'k' closest neighbors in the feature space. K-Nearest Neighbor Classifier using Grid Search CV optimizes the value of 'k' and other hyperparameters through cross-validation to improve the accuracy of the KNN classifier. Linear Classification predicts the class of a data point by drawing a linear decision boundary (e.g., a line or hyperplane) that best separates the classes in the feature space. As we see from the classification scores KNN gridsearch has the second highest accuracy on the TEST (0.68) and highest on the TRAIN (0.77) datasets. It also has the highest F1-score for the "True" class on both datasets, indicating a better balance between precision and having the best recall in TRAIN (0.52). Linear Classification performs poorly on the TEST dataset with an accuracy of 0.69 and fails to correctly classify any "True" instances (precision, recall, and F1-score are 0.00). It also has a low F1-score on the TRAIN dataset. KNN has slightly lower accuracy and F1-scores compared to KNN gridsearch on both datasets showing that KNN gridsearch is a more optimized method which is to be expected as it finds the best amount of neighbors to use. As for confidences in these models I have no confidence in Linear Classification as it fails to provide an “True” instances for anything other than recall. However, for KNN gridsearch I do trust it more as the recall of 0.29 and accuracy of 0.68 on the TEST set, but due to the nature of the problem a sub 70% correct diagnostic can be very harmful when seeing recurrences of breast cancer, which is the same for regular KNN leading me to have very low confidence in each of these models as well.