COSC 425 — Introduction to Machine Learning — Fall 2019

Project 3: k-Nearest-Neighbors and Decision Trees

Due Nov 4, 2020

In this project you will apply *k*-nearest-neighbors and decision trees to the diagnosis of breast cancer. The file is `breast-cancer-wisconsin.data` in the Project 3 folder. There are the following attributes:

1. Sample code number: id number
2. Clump Thickness: 1 – 10
3. Uniformity of Cell Size: 1 – 10
4. Uniformity of Cell Shape: 1 – 10
5. Marginal Adhesion: 1 – 10
6. Single Epithelial Cell Size: 1 – 10
7. Bare Nuclei: 1 – 10
8. Bland Chromatin: 1 – 10
9. Normal Nucleoli: 1 – 10
10. Mitoses: 1 – 10
11. Class: (2 for benign, 4 for malignant)

Your task will be to classify sample instances as benign or malignant. As usual, your report should cover the relevant steps in the workflow, reporting what you did and the results. Implement your own kNN and DT functions, except for basic numerical libraries.

1. To evaluate the performance of your classification algorithm, you will need to compare the correct classification $\{r^t\}_{t=1}^N$ with your classifications $\{y^t\}_{t=1}^N$ (on the training, evaluation, or testing data). Therefore, implement a function that prints performance metrics determined as follows. Let TP = # true positives, TN = # true negatives, FP = # false positives, FN = # false negatives. Print out a *confusion matrix* formatted as follows:

	Predicted Class	
True Class	**benign**	**malignant**
benign	TN	FP
malignant	FN	TP

(Some people arrange confusion matrices as the transpose of this.) In addition to the confusion matrix, print out the following performance metrics:

- Accuracy = (TN + TP) / (TN + TP + FN + FP)
- TPR (true positive rate, recall, or sensitivity) = TP / (TP + FN)
- PPV (positive predictive value or precision) = TP / (TP + FP)
- TNR (true negative rate or specificity) = TN / (TN + FP)
- F_1 Score = 2 × PPV × TPR / (PPV + TPR)

There are a variety of other metrics you can use to quantify performance (see https://en.wikipedia.org/wiki/Confusion_matrix).

2. **Part 1**: Implement a *k*-nearest-neighbors algorithm to predict whether data indicate a malignant or benign tumor. Run your algorithm with *k* values 2 to 8, 17, and 33, and use cross-validation to determine the best *k*. Report the performance (confusion matrix, accuracy, etc.) on the training and validation data for each *k*, and the performance on the test data for the optimal *k*.
3. **Part 2**: Implement a univariate decision tree classifier. One function should construct a decision tree from provided data. Your user should be able to specify which impurity measure to use, including at least *entropy*, *gini*, and *misclassification error*. The user should be able to limit tree construction by either an impurity threshold or maximum tree depth. Your function should report the depth of the constructed tree. For this project, it's sufficient if your classifier handles only dichotomous (two-class) classification and numeric attributes.
4. A second function should take a constructed decision tree and apply it to a data set, returning an array of predicted classifications.
5. Apply your decision tree functions to the breast cancer training data with a variety of maximum depths ($k = 2$ to 10) and impurity thresholds to determine the best hyperparameters (as determined by your validation data). Report these along with its performance on the testing data with the best hyperparameters as determined by cross-validation.
6. **Extra credit**: Use PCA to reduce the dimension of your data and try various reduced dimensions to determine if dimension reduction will improve decision tree classification on this data. That is, repeat step 5, but with the reduced data.