Assigned: 02/21/2021
Due: Thurs. 03/4/2021, 11:59pm

Instructions: This project will cover topics of classification.

Submission Requirements:

This assignment is split into two main parts.

Part A: You will be asked to walk through methods **manually**, that is not using the available functions to perform clustering analysis. You can use R, Matlab, or Python for plotting of the data when asked and for support in calculations (compute distances, averages, etc.). But, the process should be done manually with your answer presented in **typed form**.

You may then prepare your solution to Part A as a separate document using Google Docs, Word, LaTeX, with computer generated plots inserted. Note, if you prepare your solutions in Rmd, Sweave, or Python notebooks, your solution to Part A can also be included in that document using Markdown or LaTeX to enter your answers (not a separate document).

Part B: You will again have a final document submission that includes text responses to questions, tables, R/Matlab/Python code used to calculate answers, and figures.

Formatting of submissions will follow the same approaches as those used in P1 and P2. Namely: Matlab + markup, Matlab LiveScript, Rmd, Sweave, Jupyter notebook, Colab notebook.
Any other packages or tools, outside those listed in the assignments, examples, or those for reproducible research should be cleared by Dr. Brown before use in your submission.

For this project you are allowed to work in **groups of up to 3**. All students must sign-up on Canvas into a group (e.g., P3-Rho, P3-Tau, etc.).

Name your main submission files as *P2_GroupName*, create a zip-file called *Project3_GroupName.zip* and submit on Canvas. For example, if I was using R, with a separte response for Part A, and member of GroupChi, I would submit either:

- *P3_PartA_GroupChi.pdf*, *P3_PartB_GroupChi.Rmd*, *P3_PartB_GroupChi.pdf*, or
- *P3_PartA_GroupChi.pdf*, *P3_PartB_GroupChi.Snw*, *P3_PartB_GroupChi.pdf*

Questions:

Part A - WRITTEN QUESTIONS

1. (1 point) List your project GroupName and all group members names.

2. Decision Trees
Suppose you are building a decision tree on a data set with three classes A, B, C. At the current position in the tree you have the following samples available:

$$N = \begin{pmatrix} A & 100 \\ B & 50 \\ C & 60 \end{pmatrix}.$$

You are examining two ways to split the data. The first splits the data as,

$$N_{1,1} = \begin{pmatrix} A & 56 \\ B & 12 \\ C & 0 \end{pmatrix}, \qquad N_{1,2} = \begin{pmatrix} A & 44 \\ B & 38 \\ C & 60 \end{pmatrix}.$$

The second splits the data as,

$$N_{2,1} = \begin{pmatrix} A & 62 \\ B & 18 \\ C & 0 \end{pmatrix}, \qquad N_{2,2} = \begin{pmatrix} A & 28 \\ B & 11 \\ C & 24 \end{pmatrix}, \qquad N_{2,3} = \begin{pmatrix} A & 10 \\ B & 21 \\ C & 36 \end{pmatrix}.$$

(a) (10 points) Compute the gain in GINI index for the two splits. Show the form of the calculations, not just the final numbers.

(b) (2 points) Which node would be preferred to include next in the decision tree?

(c) (10 points) Compute the information gain (based on entropy) for the two splits. Show the form of the calculations, not just the final numbers.

(d) (2 points) Which node would be preferred to include next in the decision tree?

3. Naïve Bayes Classification
Consider the fruit data set, `fruit.txt`. You will construct and evaluate a Naïve Bayes classifier to predict whether a fruit is an apple or orange (ignore the data for lemon) using the remaining features (weight, height, and width). Please refer to the file `fruit_info.txt` to understand the attributes and classes of the data.

(a) (12 points) Estimate the probabilities needed for Naïve Bayes classification using Laplace smoothing,

$$P(X_i = j \,|\, Y = y_k) = \frac{\#(X_i = j, Y = y_k) + 1}{\#(Y = y_k) + |domain(X_i)|}$$

You can use code to estimate the counts, $\#(X_i = j, Y = y_k)$ and $\#(Y = y_k)$ in the data. Fill in the following tables:

Prior	Prob.
$P(apple)$	
$P(orange)$	

Cond.	Prob.
$P(Wt = 0 \,|\, apple)$	
$P(Wt = 1 \,|\, apple)$	
$P(Wt = 0 \,|\, orange)$	
$P(Wt = 1 \,|\, orange)$	

Cond.	Prob.
$P(Ht = 0 \,|\, apple)$	
$P(Ht = 1 \,|\, apple)$	
$P(Ht = 2 \,|\, apple)$	
$P(Ht = 0 \,|\, orange)$	
$P(Ht = 1 \,|\, orange)$	
$P(Ht = 2 \,|\, orange)$	

Cond.	Prob.
$P(Wid = 0 \,|\, apple)$	
$P(Wid = 1 \,|\, apple)$	
$P(Wid = 2 \,|\, apple)$	
$P(Wid = 0 \,|\, orange)$	
$P(Wid = 1 \,|\, orange)$	
$P(Wid = 2 \,|\, orange)$	

(b) (12 points) Report the predicted class on the following test samples using the estimated parameters.

Sample Num.	Test Data
1	[1, 1, 1, 0]
2	[1, 0, 0, 1]
3	[2, 0, 0, 1]
4	[2, 1, 0, 0]

Do these calculation writing out the probabilities that go into the estimates using the values from part (b).

Do not use a supplied function/package in R, Matlab, or Python.

(c) (4 points) Report which of the test samples are predicted correctly/incorrectly state for each whether it is a TP, FP, FN, TP. Assume apple is the positive class / orange the negative class.

Part B - CODE QUESTIONS

4. (15 points) For the following data set, you will compute the true positive rate, false positive rate, and accuracy. Threshold the Y_{pred} classifier output at each possible value (use a greater than equal to comparison).

   Report the results as a matrix/table with rows corresponding with 10 rows and columns for the different thresholds, the true positive rate (TPR), false positive rate (FPR), and accuracy (ACC).

Sample	Y_{true}	Y_{pred}
1	1	0.98
2	0	0.92
3	1	0.85
4	0	0.77
5	0	0.71
6	1	0.64
7	1	0.50
8	1	0.39
9	0	0.34
10	0	0.31

5. (5 points) Use the results from Prob. 4 to plot the ROC curve for the data. *Note, plot this curve using the standard plotting tools rather than any special library/package available in* R, Python, *or* Matlab.

6. Classification of Spam: Trees

   For this problem, you will work to classify e-mail messages as spam or not. The data set to be used is given with the project (note, this is not the same spam data set that is available from UCI ML repository).

   (a) Load in the `spam` data. You should **not** include the following columns in the classification task: isuid, id, domain, spampct, category, and cappct.

   (b) (6 points) To see whether a classifier is actually working, we should compare it to a constant classifier which always predicts the same class, no matter what the input features actually are.

      i. What fraction of the e-mails are actually spam?

      ii. What should the constant classifier predict?

      iii. What is the error rate of the constant classifier?

   (c) (3 points) Split the data into a training and test set with an 80/20 split of the data.
   Set the see for the random generator to "124".
   Helpful functions: R - `sample`, Matlab - `cvpartition`, Python - `train_test_split` from `sklearn.model_selection`.

Note for Python users, the sklearn decision tree functions do not work with categorical data. Therefore, you will need to transform the categorical/ordinal data to Numeric - `OrdinalEncoder` or `OneHotEncoder` from `sklearn.preprocessing`

(d) (8 points) Construct a classification tree to predict spam on the training data. Then, print out the tree found.
Helpful functions: R - `rpart, plot` library, Matlab - `fitctree, view`,
Python - `DecisionTreeClassifier, export_graphviz` from `sklearn.tree`.

(e) (2 points) Which selection criteria is used by default when learning the tree model?

(f) (6 points) Estimate the performance of the decision tree on the training set and the testing set. Report accuracy, sensitivity, specificity, and AUC (if a method returns a probability rather than a label use a threshold of 0.5).
Helpful functions: R - `pROC, ROCR` library and `confusionMatrix` of the `caret`, Matlab - `confusionmat, perfcurve`, Python - `sklearn.metrics` library.

(g) (8 points) Try pruning the tree, print out the tree that is a different size and report the classification performance (accuracy, sensitivity, specificity, and AUC).

7. Spam, Spam, Spam

For this problem, you will continue to work with the spam data set from above.

(a) Prepare the data for a 10-fold cross-validation (using the do-it-yourself approach described in the example code).
Helpful functions: R - `cvFolds`, Matlab - `cvpartition`, Python - `StratifiedKFold` from `sklearn.model_selection`.

Note: Decision trees in `R` and `Matlab` work for a mixture of categorical and numeric data, the other learning methods do not. Therefore, you will need to convert the data to numeric. Once converted to numeric, you should also ensure the data is scaled using min max scaling from 0 to 1.
Helpful functions: R - `preProcess, predict` from `caret` library, Matlab - `normalize`, Python - `MinMaxScaler` from `sklearn.preprocessing`.

(b) For each of the following methods, estimate the generalization performance over the 10-folds, calculate and report the accuracy, sensitivity, specificity, and AUC performance on the testing data (for each split and averaged over the splits). Show the results in a table.

Hint: You can use one for loop and create all the models required below within it to avoid duplicating code.

i. (14 points) Did you set up the experimental evaluation correctly?

ii. (14 points) Use kNN to predict whether an email is spam. Show performance results for these values of $k = 3, 7, 11, 15$.
Helpful functions: R - `knn` from `class` library, Matlab - `fitcknn`,
Python - `KNeighborsClassifier` from `sklearn.neighbors`.

iii. (8 points) Use Decision Trees to predict whether an email is spam. Estimate the generalization performance over the 10-folds, calculate and report the accuracy, sensitivity, specificity, and AUC performance on the testing data (for each split and averaged over the splits). Show the results for two different sized decision trees (consider different amounts of pruning).
Helpful functions: described above.

iv. (8 points) Use a Naive Bayes classifier to predict whether an email is spam.
Helpful functions: R - `naiveBayes` from `e1071` library, Matlab - `fitcnb`, Python - `GaussianNB` from `sklearn.naive_bayes`.

Questions for CS5831 ONLY!

8. (18 points) Write a function, `classifierPerf`, that will evaluate the predictions of a classifier. The function takes two arguments: a vector (or data frame column) of predicted values, $Ypred$, and vector of actual class values, $Ytrue$, and returns the following quantities:

   - number of true positives (TP),
   - number of false positives (FP),
   - number of true negatives (TN),
   - number of false negatives (FN),
   - true positive rate (TPR),
   - true negative rate (TNR),
   - sensitivity (Sens.),
   - specificity (Spec.),
   - precision (Prec.),
   - recall (Rec.),
   - accuracy (Acc.), and
   - error rate (Err)

   For this function you create, you **can not use** other libraries functions to compute any of the desired values or functions to compute the confusion matrix.