26:711:685, Fall 2019
Homework 2 **Due date:** 10/23/19

Instructions: Please read carefully

Please answer the following questions in **electronic** form and upload and submit your files to Sakai Assignment site, before the due date. Make sure to click on the **submit** button. For this homework you should submit **three files**.

For questions 1 and 2 you may use a word processor software like Word or similar, or use hand-written answers. Either way, transform your solution into a **single pdf file for both questions** and upload and submit it on the course Sakai site.

For each of questions 3 and 4 you should submit a **A Single R or Python file**. For each part of questions 3 and 4 have the necessary R/Python code along with answers to questions in the form of an output. For example, here is a sample R code for homework answers:

```
# Question 3)
print("Question 3a)\n")
knnModel <- knn (y ~ x, data = someData)
print(summary(knnModel))
print("Answer to Q2a):\n")
print("error rate is ...\n")
#Question 3b)
. . . .
```

Include ample comments explaining what you are doing for each part of your code.

1. Consider the following table where certain keywords from daily Twitter messages of stock brokers and traders are collected, along with whether the stock market (as indicated by the S&P 500 index) went up or down on that day. Each word is represented by a binary feature variable, which is 1 if a participating brokers or trader used it on that day, and 0 otherwise. (In this toy example we are following 11 broker or trader.) The data is given below:

	ipo	profitUp	loss	interestLow	dividendUp	bankrupcy	hostileTakeOver	SP500
1	1	0	0	0	1	0	1	Up
2	0	0	0	1	0	0	1	Up
3	1	0	0	1	1	0	0	Up
4	1	1	1	1	1	0	0	Up
5	1	0	0	1	1	0	0	Up
6	0	1	0	1	1	0	1	Up
7	1	1	0	1	1	0	1	Up
8	1	0	1	0	1	1	1	Down
9	1	0	1	0	0	1	0	Down
10	0	0	1	0	0	0	0	Down
11	1	0	1	0	1	1	0	Down
12	1	0	1	0	0	0	0	Down
Pr[1|UP]								
Pr[1|Down]								
Pr[0|UP]								
Pr[0|Down]								

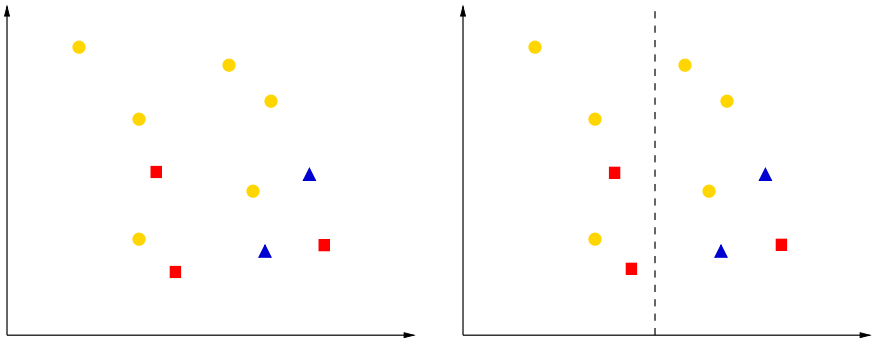
1a) Fill in the last four rows of the table, giving estimated conditional probabilities of 1 and 0 values in each column given S&P index is up, and given it is down. Also in the last column give the estimated *prior* probability of the index going up and going down.

1b) A new data comes in for today, containing the words "interestLow", and "bankrupcy". Compute the estimated probability of market going up and market going down using Naive Bayes method. What will be the prediction for the market for today?

1c) Repeat questions 1a) and 1b), but this time use the Laplace Smoothing technique, with $\alpha = 1$ and $\beta = 2$. Will the prediction for the market behavior for today change?

While it is possible to do this homework by hand, I wrote simple R scripts to compute them. I did not use any naive Bayes packages. All calculations are done explicitly. See files `Q1nbTwitter.r`and the accompanying input file `twitData.csv`.

2. Consider the following set and its partition to the right.



2a) Compute the misclassification rate for both before and after

- **Before:** $5/11 = 0.45$
- **After:** $5/11 \times 2/5 + 6/11 \times 3/6 = 5/11 = 0.45$

2b) Compute the Gini index for both before and after partition.

- **Before:** $2 \times \Big(5/11 \times (1-5/11) + 3/11 \times (1-3/11) + 2/11 \times (1-2/11)\Big) = 144/121 = 1.19$
- **After:** 5/11*gini(left) + 6/11*gini(right).
  gini(left)=$2 \times \big(3/5 \times (1-3/5) + 2/5 \times (1-2/5) + 0/5 \times (1-0/5)\big) = 24/25 = 0.96$
  gini(right)=$2 \times \big(3/6 \times (1-3/6) + 2/6 \times (1-2/6) + 1/6 \times (1-1/6)\big) = 44/36 = 1.22$
  gini(partition)=$5/11 * 0.96 + 6/11 * 1.22 = 1.10$

2c) Compute the cross entropy for both before and after partition. Use
All logarithms are in base 2:

- **Before:** $-6/11 \times \log(6/11) - 3/11 \times \log(3/11) - 2/11 * \log(2/11) = 1.435$

- **After:** 5/11*entropy(left) + 6/11*entropy(right).
  entropy(left)=$-3/5 \times \log(3/5) - 2/5 \times (2/5) - 0 * \log(0) = 0.97$
  entropy(right)=$-3/6 \times \log(3/6) - 2/6 \times \log(2/6) - 1/6 \times \log(1/6) = 1.46$
  entropy(partition)=$5/11 * 0.97 + 6/11 * 1.46 = 1.24$

Note that the misclassification rate does not change before and after partition. Before the set would be classified as yellow, and after partition both partitions will also be classified as yellow. This will make misclassification rate unchanged. On the other hand, both Gini and entropy measures are reduced after partition, indicating that the partition improved.

We could code these in R like this:

```
> # Define functions:
> misClass<-function(v){return (1-max(v)/sum(v))} #for one partition
> gini<-function(v){return(sum(v/sum(v)*(1-v/sum(v)))*2)} #for one partition
> entropy<-function(v){return(1/log(2)*sum(ifelse(v==0,0,-v/sum(v)*log(v/sum(v)))))}
> ff<-function(V,f)return(sum(apply(V,1,f)*rowSums(V))/sum(V)) #One function for all
> fr<-c(6,3,2) #Before partition (yellow, red, blue)
> fr1<-c(3,2,0) #After partition, left (yellow, red, blue)
> fr2<-c(3,2,1) #After partition, right (yellow, red, blue)
> misClass(fr)
[1] 0.4545455
> gini(fr)
[1] 1.190083
> entropy(fr)
[1] 1.435371
> ff(matrix(c(fr1,fr2),nrow=2),misClass)
[1] 0.4545455
> ff(matrix(c(fr1,fr2),nrow=2),gini)
[1] 1.10303
> ff(matrix(c(fr1,fr2),nrow=2),entropy)
[1] 1.23724
```

3. **R/Python Project: Simulation to compare the kNN and the Bayes error rates** In this project you are going to generate a classification data according to a known distribution. Since you know the distribution of the data you can draw the decision boundaries of classes predicted by the Bayes decision rule.

   Next, you will generate data according to this distribution and use it along with the kNN model for best value of k for prediction; the best value of k is determined by 1-folding cross validation. You will compare the boundaries generates by the kNN and by the Bayes decision rule.

   The simulation step:

   3a) Generate two sets of data according to the bivariate normal distribution. Recall that for the

random vector $\mathbf{x} \in \mathbb{R}^d$ the density is given by

$$\phi(\mathbf{x} \mid \boldsymbol{\mu}, \Sigma) = \frac{1}{\left(2\pi \det(\Sigma)\right)^{d/2}} \exp\left(-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right)$$

where $\boldsymbol{\mu}$ is the vector of means, and Σ is the covariance matrix. For $n = 2$, $\boldsymbol{\mu}$ is a vector of length 2 and Σ is a 2×2 matrix.

You need two sets of data. For the first one generate 300 2-dimensional points (X_1, X_2) drawn from a bivariate normal distribution with $\boldsymbol{\mu}_R = (680, 700)$, and

$$\Sigma_R^{1/2} = \begin{pmatrix} 149 & -47 \\ -47 & 209 \end{pmatrix},$$

and give the response value Y of `red` for all of them.

For the second group generate another 300 2-dimensional points (X_1, X_2) drawn from a bivariate normal distribution with mean vector $\boldsymbol{\mu}_B = (10, 550)$ and a covariance matrix

$$\Sigma_B^{1/2} = \begin{pmatrix} 200 & 15 \\ 15 & 250 \end{pmatrix},$$

and give the response value Y of `blue` for all of them.

To generate such points, one easy method is the following:

A. Generate 600 independent numbers normally distributed with mean 0 and standard deviation 1.

B. Organize these numbers into a 2×300 matrix, called `data`. The first row is going to be the X_1 coordinates of these points, and the second row is going to be the X_2 coordinates. Currently these 300 (X_1, X_2) points have a bivariate normal distribution with mean $(0, 0)$ and the covariance equal to the identity matrix.

C. Multiply the matrix `data` by $\Sigma_R^{1/2}$: `data` $\leftarrow \Sigma_R^{1/2} \times$ `data`. The new data points are going to have mean $(0, 0)$ and covariance Σ_R.

D. Add to each column of `data` the vector $\boldsymbol{\mu} = (680, 700)$. Now you have the 300 points with a distribution specified for the red points.

A similar approach generates the 300 blue points. Just use $\boldsymbol{\mu}_B$ and Σ_B instead.

3b) Next, put these 600 point, along with their color value Y in a data frame called `simData`.

3c) Plot the red and blue points in a scatter plot, with the right color.

3d) Generate another 200 red and 200 blue points just as above, for testing purpose. Collect these 400 points in a data frame called `testDF`.

3e) Generate a 200×200 grid of in the range of X_1 and X_2. Using Bayes decision rule color these points red or blue according to the Bayes rule, and add them to the scatter plot of the `simData`

(use a small dot for the grid points.) Draw the boundary in maroon color between the blue and red areas as prescribed by the Bayes rule.

3f) Using Bayes decision rule predict the Y value of this test data and compare it to their actual value. Print the confusion table for `testDF` and also print the test error rate.

Learning Step:

3g) Load the `kknn` library for R or the `KNeighborsClassifier` for Python. Use the `simData` data frame you generated earlier. Using 1-folding cross validation find the best k for the kNN model for this data. Using this k build a kNN model called `bestKnnModel` on the training data `simData`. Predict the Y values of the test data `testDF`. Print the confusion table and the test error rate of the best kNN model.

3h) Predict the Y values of the 200×200 grid data you generated. Plot the scatter plot of the `testDF` with their correct class color (red or blue) and the grid points with their predicted color. On the same plot draw both the boundary line for the Bayes decision rule (in maroon) and the kNN decision rule (in dark green.)

4. **R/Python Project for building the Naive Bayes and CART Models:** For this question you will work with the *Adult* data set in the UC-Irvine archive. This data is mined from the US Census Bureau. It attempts to relate whether a person's income is over $50,000 per year or below that level to multiple variables, both quantitative and qualitative. The data comes in two separate files. One file, containing over 32,000 items should be used as the training set. The second, containing over 16,000 items should be used as the test set. Go to the UCI archive site, look for the "adult" data and try to learn as much about it as possible, including the variables used in the it.

   Follow this link and study the origin and contents of this data.

   4a) Read the data file found in this link and save the data into a data frame called `adultTrain`. Since the column headings are not in the data you must make sure that when reading the data, you take this fact into account. Remember that the `read.csv` function in R and the `pd.read.csv` function in Pandas library in Python by default treat the first row as the title row. You must turn off this action. Furthermore, pay attention that missing data is represented by " ?" symbol (note the blank followed by "?"). As mentioned, this data does not contain headers. The headers can be found in a different file in the same directory containing the adult data. To make your life easier, we have provided the headers, in the correct order below:
   `"age", "workclass", "fnlwgt", "education", "educationNum", "maritalStatus", "occupation", "relationship", "race", "sex", "capitalGain", "capitalLoss", "hoursPerWeek", "nativeCountry", "income"`

   Print the head and tail to make sure it is read properly. Also print the list of data that are categorical, along with the levels in each of them. These should be done to make sure the data is read correctly.

The data has over 32,000 items in it. It may be helpful to build your model and answer the following questions for a small sample, say 500 items first. When you are confident that your model works, you can run it on the whole data.

4b) Load the `naivebayes` package or the analogeous package in Python. Build a model called `naiveModelA` with `income` as the response variable, and other features as independent variables. The model should use no Laplace smoothing, and treat numerical data non-parametrically using a kernel based method. Print a summary of the model.

4c) From the model extract the estimated class conditional probabilities of males and females under the `sex` feature for people with income over \$50K and people with income under \$50K per year.

4d) From the model extract the estimated class conditional mean and standard deviation of `hoursPerWeek` feature for people with income over \$50K and people with income under \$50K per year. Also draw boxplots for this feature for each income class. (You may wish to examine this information for all numerical and categorical features to get a feel of each one's relation with income level.)

4e) Form the UCI archive read the test data from this link and save it in the data frame `adultTest`. If you examine the original file, you will see that the first few lines are comments, and start with the character "`|`". Make sure to set the comments character while reading the data, so these lines are skipped. Also, like the training data, there are missing items represented by " ?" (with a blank before "?"). Make sure to account for them. Finally, unfortunately in this test data the values of income are "`<=50K.`" and "`>50K.`" with an extra"." You need to pre-process this data and remove these dots.

Once the test data is read, predict the income level of the test data, and save them in the vector `naivePredA`. Then compare it to the actual values. Print the confusion matrix and the error rate for the test data.

4f) Repeat parts 4b)-4e) but this time assume all numerical variables follow the normal distribution for numerical features. How does this change affect the test error rate?

4g) Now load the `tree` library in R or the analogeous library in Python and repeat parts 4b)-4e) for the default tree model. Also, print the resulting tree. Compare the error rate with the naive Bayes models.

4h) Next apply the cost complexity pruning and find the tree it produces and name it `bestTree`. Plot the best tree, and also print its confusion matrix and error rate on the test data.

4i) Finally, develop a random forest model where at each partition only a random subset of features of size $\mathtt{m} = \sqrt{\mathtt{d}}$ is used, where `d` is the number of features. Print the confusion matrix and the error rate for the test data. Plot the importance of parameters in order of importance using bar

charts. Plot importance with respect to changes on the classes "`<=50K`" and "`>50K`". Also, plot the importance with respect to the mean decrease in accuracy and with respect to decrease in the Gini index.