CSC570AL Machine Learning Assignment 2
Hands on with KNN and Naïve Bayes Classifiers
(35 points)

Problem 1: Applying k-Nearest Neighbors to predict income

For this assignment, we will be using the census adult dataset from UCI ML repository. The Adult dataset was extracted by Barry Becker from the 1994 US Census Database. Each row in the dataset has de-identified dempgraphic information of an individual worker and their income. The income is a categorical variable with two levels: <50K and >50K. The goal of this assignment is to create a binary classifier to predict whether a person makes more than 50K based on the other attributes in the dataset.

Please see the description of dataset and its attributes here: https://archive.ics.uci.edu/ml/datasets/Adult
Then go to data folder at https://archive.ics.uci.edu/ml/machine-learning-databases/adult/ and download adult.data . This would be the dataset you will use to answer the following questions.

Data Exploration

1. (1pt) Download the dataset and store it in a dataframe in R. The dataset does not have header, you should add the headers manually to your dataframe based on the list of attributes provided in https://archive.ics.uci.edu/ml/datasets/Adult. Also please note that some entries have extra white space. So to read the data properly, use the option strip.white=TRUE in read.csv function.

2. (1pt) Explore the overall structure of the dataset using the str() function. Get a summary statistics of each variable. How many categorical and numeric variables you have in your data? Is there any missing values?

3. (1pt) Get the frequency table of the "income" variable to see how many observations you have in each category of the income variable. Is the data balanced? Do we have equal number of samples in each class of income?

4. (3 pts) Explore the data in order to investigate the association between income and the other features. Which of the other features seem most likely to be useful in predicting income.

   - To explore the relationship between numerical features and "income" variable, you can use side by side box plot and t.test

   - To explore the relationship between categorical features and "income" variable, you can use frequency table and chisquare test (note that chisquare test might throw a warning if there are cells whose expected counts in the frequency table is less 5. This warning means the p-values reported from chisquare test may be incorrect due to low counts and are not reliable. You can ignore the warning for this assignment).

Based on your data exploration above, decide which attributes you are going to use to predict income. **Explain your reason for selecting these attributes.**

Data Preparation:

5. (1 pt) An initial data exploration shows that the missing values in the dataset are denoted by "?" not NA. Change all the "?" characters in the dataframe to NA

6. (1pt) Use the command colSums(is.na(<your dataframe>) to get the number of missing values in each column of your dataframe. Which columns have missing values?

7. (3 pt) There are several ways we can deal with missing values. The easiest approach is to remove all the rows with missing values. However, if a large number of rows have missing values removing them will result in loss of information and may affect the classifier performance. If a large number of rows have missing values, then it is typically better to replace missing data with some values. This is called data imputation. Several methods for missing data imputation exist. The most naïve method (which we will use here) is to replace the missing values with mean of the column (for a numerical column) or mode/majority value of the column (for a categorical column). We will use a more advanced data imputation method in a later module. For now, replace the missing values in a numerical column with the mean of the column and the missing values in a categorical column with the mode/majority of the column. After imputation, use colSums(is.na(<your dataframe>) to make sure that your dataframe no longer has missing values.

8. (2 pt) This dataset has several categorical variables. With the exception of few models (such as Naiive Bayes and tree-based models) most machine learning models require numeric features and cannot work directly with categorical data. One way to deal with categorical variables is to assign numeric indices to each level. However, this imposes an artificial ordering on an unordered categorical variable. For example, suppose that we have a categorical variable primary color with three levels: "red","blue","green". If we convert "red" to 0 , "blue" to 1 and "green" to 2 then we are telling our model that red < blue< green which is not correct. A better way to encode an unordered categorical variable is to do **one-hot-encoding.** In one hot-encoding we create a dummy binary variable for each level of a categorical variable. For example we can represent the primary color variable by three binary dummy variables, one for each color (red, blue, and green) . If the color is red, then the variable red takes value 1 while blue and green both take the value zero.

   Do one-hot-encoding of all your unordered categorical variables (except the income variable). You can use the function one_hot from mltools package to one-hot encode all categorical variables in a dataset. Please refer to https://rdrr.io/cran/mltools/man/one_hot.html . Use option DropUnusedLevels=True to avoid creating a binary variable for unused levels of a factor variable.

   Please note that the one_hot function takes a data table not a dataframe. You can convert a dataframe to datatable by using as.data.table method https://www.rdocumentation.org/packages/data.table/versions/1.12.8/topics/as.data.table. Make sure to use library(data.table) before using as.data.table method. You can covert a datatable back to a dataframe by using as.data.frame method https://www.rdocumentation.org/packages/base/versions/3.6.2/topics/as.data.frame

Training and Evaluation of ML models

9. Set the seed of the random number generator to a fixed integer, say 1, so that I can reproduce your work:

```
> set.seed(1)
```

10. (1 pt) Scale all numeric features using Min-Max scaling

11. (1pt) Randomize the order of the rows in the dataset.

12. (4 pts) Use **5-fold** cross validation with KNN to predict the "income" variable and report the cross-validation error. (You can find an example in slides 51-53 of module 4 lecture notes).

13. (2 pts) Tune K (the number of nearest neighbors) by trying out different values (starting from k=1 to k=sqrt(n) where n is the number of observations in the dataset (for example k=1,5,10,20 50,100, sqrt(n)). Draw a plot of cross validation errors for different values of K. Which value of *K* seems to perform the best on this data set? (You can find an example in slides 54-55 of module 4 lecture notes) Note: This might a long time to run on your machine, be patient (It took about 30 minutes on my machine to run 5-fold cross validation for 6 different K values)

14. (3 pt) Use **5-fold** cross validation with KNN to predict the income variable and report the average **false positive rate (FPR)** and **false negative rate (FNR)** of the classifier. . FPR is the proportion of negative instances classified as positive by the classifier. Similarly, FNR is the proportion of positive instances classified as negative by the classifier.

It does not matter which class you designate as positive or negative. For instance, you can designate income>50K as positive and income<50K as negative

Note : don't need to tune K again, you can use the best K you found in the previous question.

Note: you can find an example of computing false negative rate and false positive rate for each fold in module 5 lecture notes slides 56-59. These slides use a naiive bayes classifier for prediction but you can modify it to use knn and report the average FPR and FNR

15. (2 pt) Consider a majority classifier which always predicts income <50K. Without writing any code, explain what would be the training error of this classifier? (Note the training error of this majority classifier is simply the proportion of all examples with income>50K because they are all misclassified by this majority classifier). Compare this with the cross validation error of KNN you computed in question 8. Does KNN do better than this majority classifier?

16. (2 pt) Explain what is the False Positive Rate and False Negative Rate of the majority classifier and how does it compare to the average FPR and FNR of KNN classifier you computed in question 10. You don't need to write any code to compute FPR and FNR of the majority classifier. You can just compute it based on the definition of FNR and FPR.

Problem 2: Applying Naïve Bayes classifier to predict income

Use the same dataset as problem1. This time we are going to use Naïve Bayes to predict income.

1. Set the seed of the random number generator to a fixed integer, say 1, so that you can reproduce your work:
   > set.seed(1)

2. Based on your data exploration in problem1, decide which features you want to keep to predict the income and remove other features.

Note: As we learned in the lectures, Naïve Bayes uses frequency tables to compute conditional probabilities. Frequency tables are only computed for categorical (or discrete features). For a continuous feature, you can either

1. manually convert it to a categorical variable by binning (as explained in the lectures), or
2. pass it directly to naiveBayes() function in R and it will use **Gaussian Naïve Bayes** to compute the conditional probabilities. More specifically, If x is a continuous feature and y is the target variable (i.e., the variable we want to predict) with $c_1, c_2, \ldots, c_k$ classes, then the Gaussian Naïve Bayes assumes that the likelihood $p(x|y = C_i)$ has a Gaussian/normal distribution and estimates its mean and variance from the training data. This distribution is used (instead of a frequency table) to compute the likelihood for each observed value of x in the data. Fortunately, the naiveBayes function in R does this computation for you. You just pass the dataset as is and it will automatically use Gaussian Naïve Bayes to compute conditional probabilities for continuous features.

3. (4 pt) Use 5-fold cross validation with Naïve Bayes to predict the “income” variable and report the cross-validation error.

4. (2 pt)Compare the cross validation error of Naiive Bayes with that of KNN. Which one performs better on this dataset?

5. (2 pt) Compare the False Positive Rate and False Negative Rate of naiive Bayes with those of the majority classifier which always predicts income <50K.

Bonus Problem--Optional (+5 pt)

A classification model that is trained on an imbalanced outcome variable (e.g., income in this dataset) is likely biased. There are several ways to deal with the imbalanced outcome variable, including undersampling and oversampling. Read this article for a short reference **https://www.marcoaltini.com/blog/dealing-with-imbalanced-data-undersampling-oversampling-and-proper-cross-validation** . Try to use at least two different methods to balance the income variable in the adult dataset and compute the cross validation errors of knn and naïve Bayes on this balanced dataset. Refer to the above article to see how oversampling or downsampling are performed together with cross validation. Pay special care not to oversample your data before cross validation. Does balancing data increase your classifier performance?

What to Turn in:

You need to create an R notebooke consisting of your answers to the questions outlined above for problems 1 and 2 together with your R code you used to answer each question.

Your submission must be in two formats:

1. **A .html file which contains the preview of your notebook**. When you click on preview in R studio to preview an R notebook, an html file is created in the same directory as your notebook. You must submit this .html file or your submission will not be graded.
2. **An .rmd file which contains your R notebook**.

Please do not hesitate to email me if you have any question.