2/7/2018

CSE 5243
Instructor: Jason Van Hulse
Homework 2

Due Date: 2/21/2018 5:30pm

In this lab, you will write a program to implement (from scratch) a k-Nearest Neighbor (kNN) classification algorithm and test this method on the **Wine_quality** dataset. You may work as either an individual or as a team of 2.

The wine_quality dataset can be found at the following URL:
https://archive.ics.uci.edu/ml/machine-learning-databases/wine-quality/winequality-red.csv

This homework uses the *winequality-red.csv* dataset.

Data Preprocessing Tasks:

1. You will need to add a binary class variable according to the following rule:
   quality <= 5 —> class = "Low", quality > 5 —> class = "High"
2. You will need to remove the *quality* attribute after you create the class label
3. Partition the dataset into a training set (75%) and test set (25%). The partitioning should be done once randomly.

You will write a kNN classifier to predict the class of the test records, given the training set. *k* should be a model parameter that you can easily vary. Individuals should pick a single proximity measure, while teams of 2 should implement 2 different measures. The program that you write should create the following output:

> An *mx3* data frame or table, where *m* is the number of examples in the *test* dataset. The *Actual Class* is the class of the record that is listed in the Test data, *Predicted Class* is the prediction that is made based on the kNN algorithm and training dataset, and *Posterior Probability* is the probability that the record belongs to the predicted class, based on the kNN algorithm.

Actual Class	Predicted Class	Posterior Probability
High	Low	0.550
Low	Low	0.731

(this example is illustrative).

You will also compare your kNN implementation to any other off-the-shelf kNN implementation (e.g., the kNN classifier in R).

What you need to turn in to Carmen:

1) Program
2) Readme - contains all the important information about the directory, including how to run the program and how to view the resulting output.
3) Report
   - *Please hand in a hard-copy of the written report in class on the due date.*

Program (50% of grade): You can use either R, Python or Java for this assignment. You can use built-in statistical functions, data transformations or graphing packages for exploratory data analysis, **but you should code the kNN algorithm from scratch**. Please add comments to your code to improve readability.

Report (50% of grade):
In addition to turning in the program, you should create a report (maximum of approximately 9 pages for individuals, 14 pages for teams of 2) which describes your program and the design decisions and analyzes the performance of your kNN classification algorithms.

1) **Preliminary data analysis** - discuss any observed trends with the data, include interesting graphs (histograms, scatterplots), summary statistics, correlations among features, etc. I would suggest including some of the interesting output of this preliminary data analysis to help augment the presentation.

2) **Data Transformations** - What if any transformation or processing of the data is needed (prior to actually building the models) e.g., treatment of outliers and missing values, discretizing, feature subset selection? Even if you decide not to handle these issues, please discuss why.

3) **Model development** - Describe the approach you took to train and test the model, and where applicable the rationale for the approach. Detailing what you did is very important even if it did not work. Describe any difficulties you may have encountered and any assumptions you are making.

4) **Model evaluation -**
   - Produce confusion matrices, classification and error rates for the test dataset for a few different values of *k*.
   - Compute True Positive, False Positive, True negative and False negative rates, Recall, Precision and F-Measure, as well as the ROC curve.
   - Provide detailed analysis of the results. What trends did you observe? Provide graphs and/or statistics to back up and support your observations.

5) **Comparison**: Compare your kNN classifier to any off-the-shelf implementation of the kNN classifier. How did your algorithm compare? You do not need to turn in this code, just briefly describe the implementation you used and the parameter settings selected. Focus on comparing the classification performance of the off-the-shelf implementation to your program.

Your report will be graded on the completeness of each of these sections, the depth of the analysis performed, the reasonableness of the choices you make, and readability (is it well organized and does the presentation flow in a logical manner; are there many grammar and spelling mistakes; do the charts/graphs relate to the text, etc…)

Working as a team of 2: Teams of 2 should add extra, deeper analysis. Some ideas to consider:

- Implement a second proximity measure, and compare the performance of your kNN classifier with these two different measures.
- Implement a weighted posterior when determining the predicted class (i.e., adjust the voting algorithm to include the distance to each nearest neighbor somehow). How does this weighted posterior perform compared to the un-weighted version?
- Analyze what happens when you reduce the size of the training dataset. You can do this by randomly selecting 50% of the examples, or 25% of the examples, and using only these to classify instances in the Test dataset. What impact does size of the training dataset have on performance?

How to hand in your work:

Please choose one of the programming languages from: JAVA, Python, R.

Submit to Carmen: All the related files except for the data will be tarred in a **single** *.zip file or *.tgz file. Please use this naming convention: ”Project2_Surname_DotNumber.zip" or “Project2_Surname_DotNumber.tgz.” The submitted file should be less than 5MB.

Hand in a hard copy of your report in class on the due date.