**IS 6482 Spring 2017  
Assignment 4 – KNN, Clustering and Association Rule Mining**

**Due: 11:59 pm, March 19 (Sunday)**

**Resources:** Tutorials from weeks 6, 7 and 8.

**Input files:**

* Walmart\_2015\_visits\_sample6.csv for KNN-based classification and Kmeans clustering
* Walmart\_2015\_dept\_baskets.csv for Association Rule Mining

**Data Source: Walmart and Kaggle**

*Walmart held its* 3rd Kaggle “recruiting” competition (<https://www.kaggle.com/c/walmart-recruiting-trip-type-classification>) in Fall 2015 to attract data scientists interested in getting jobs at Walmart.

The raw data is a long market basket format of items purchased during each visit and some other item related information and trip information.

For Walmart’s competition in Kaggle, data scientists compete on classifying shopping trip types based on the items that customers purchased. To give a few hypothetical examples of trip types: a customer may make a small daily dinner trip, a weekly large grocery trip, a trip to buy gifts for an upcoming holiday, or a trip to buy seasonal items.

For classifying a customer visit’s trip\_type, the raw data needs to be processed to aggregate items by VisitNumber and to extract meaningful and potentially effective predictors for trip\_type. Besides being regarded as data preparation, this is also about feature (predictor) engineering.

Walmart\_2015\_visits\_sample6.csv was created this way to contain the following features:

* trip\_type - a categorical id representing the type of shopping trip the customer made. This is the ground truth that you are predicting. Trip type, 999, is an "other" category. To reduce the challenge, Walmart\_2015\_visits\_sample6.csv only contains 6 of the original 38 trip types.
* dow – Day of Week of the trip
* unique\_items – the number of unique UPC numbers of the products purchased in a visit
* total\_purchase\_quantity - the total number of the items that were purchased in a visit
* total\_return\_quantity - the total number of the items returned in a visit
* net\_quantity = total\_purchase\_quantity – total\_return\_quantity
* unique\_departments – the number of unique departments representing the purchased items in a visit.
* one\_item\_only\_departments – the number of unique departments representing single-departmental-product purchases in a visit
* return\_departments – the number of unique departments representing the returned items in a visit.

To practice Association Rule Mining, Walmart\_2015\_dept\_baskets.csv is derived from this data source to represent shopping baskets also in the long format based on VisitNumber as transaction id and DepartmentDescription as a high level indication of item type in a basket.

**Packages required:** Install C50, psych, rweka, caret, rminer, matrixStats, knitr and arules packages.

**Submission:** Submit two files – A4\_your\_initials.Rmd which is an R code file together with comments and text required by this assignment’s tasks, and A4\_your\_initials.html, (or another output format) generated from rendering (or knitting) A4\_your\_initials.Rmd.

**Task I (1 - 40%, 2 – 35%, 3 – 20%):** Create A4\_your initials.rmd to meet the following requirements:

Use beginning text (meta fields) to include assignment title, author name – you, and the file creation date. Set output to an output format of your choice. Create code chunks to meet the following requirements. (*Examine the results for your own learning and future Quiz and Exam questions.)*

1. Code chunk 1 – Understand Walmart\_2015\_visits\_sample6.csv using correlation analysis (pairs.panels from psych), decision trees (C5.0) and clustering (SimpleKmeans in RWeka)
   1. Package loading, and Walmart\_2015\_visits\_sample6.csv import and transformation. Show the overall structure of the input file. Transform factor variables, and show summary of the input data file. Use pairs.panels to exam variable distributions and correlations.
   2. Build a descriptive C5.0 decision tree using the default setting and the whole data set (trip\_type is the target variable). Show summary of the model to see the tree and the in-sample confusion matrix.
   3. Building and show clusterings to better understand visits in clusters of similar visits according to the following requirements.
      1. Use SimpleKMeans for all tasks. Remove trip\_type from input for building clusters. Show the standard deviations in addition to the centroids of the clusters.
      2. Generate and show 6 clusters using the default (i.e. random) initial cluster assignment and the default distance function (Euclidean).
      3. Keep the number of clusters to be 6 and the distance function to be Euclidean, change the initial cluster assignment method to the Kmeans++ method. Regenerate and show the clustering
      4. Keep the number of clusters to be 6 and the initial cluster assignment method to be the Kmeans++ method. the distance function to "weka.core.ManhattanDistance". Regenerate and show the clustering
      5. Choose your own distance function and initial cluster assignment method, increase the number of clusters to 9. Regenerate and show the clustering.
      6. Use the same distance function and initial assignment method selected for task C.v of this chunk, change the number of clusters to 3. Regenerate and show the clustering.
2. Code chunk 2 – KNN-based trip\_type classification using IBk of RWeka
   1. Define a few cross-validation functions that allows for changes in IBk’s parameters – K, X, I and/or F.
   2. Call the function or one of the functions defined in A of this chunk with the default parameter setting of IBk to set a base line out-of-sample performance of KNN-based trip\_type classification. Set the number of folds to 5 or more.
   3. Performance improvement based on the following requirements and suggestions:
      1. Change IBk’s parameters in the calls of functions defined in task A of this chunk to improve this classifier’s overall accuracy in cross validation.
      2. You can also selectively remove some predictors in this chunk to improve the effectiveness of selecting nearest neighbors.
      3. Use the same number of folds as what you set for task B of this chunk.
      4. While your goal is to improve performance, you also want to learn from the process the non-linear relationships between parameter values, predictors and classification performance. Experiment with the different parameter values, predictor selection as different combinations may give you similar performance improvements.
      5. Your final code for task C of this chunk should include only two calls to the function defined in A with two different combinations of IBk parameter values and/or predictor selections that have given you higher classification accuracies than that in B.
3. Code chunk 3 – Read and mine Walmart dept baskets in the long file format.
   1. Import Walmart\_2015\_dept\_baskets.csv file using the following read.transactions() with the “single” format (for long format) and save it in a sparse matrix called, e.g., Dept\_baskets.

Dept\_baskets <- read.transactions("Walmart\_2015\_dept\_baskets.csv", format="single", sep = ",", cols=c("VisitNumber","DepartmentDescription"))

* 1. Inspect the departments in the first 5 transactions.
  2. Use the itemFrequencyPlot command to perform the following tasks.
     1. View the frequency (in percentage, i.e., the relative format) of all of the item (i.e. dept) sets with support = 0.12 or higher.
     2. Plot the most frequent 8 items in the descending order of transaction frequency in percentage.
  3. Use the apriori command to generate about 50 to 100 association rules from the input data. Set your own minimum support and confidence threshold levels. Remember if the thresholds are too low, you will get too many rules, or if you set them too high, you may not get any or enough rules. Show the rules in the descending order of their lift values.

For each chunk:

* Add some simple descriptive text in the text area before the code chunk.
* Add a name or description of each code chunk in {r}. Be sure that you allow code and output from executing code to be included in the file from rendering A4\_your\_intials.Rmd.
* Add comment lines for each code requirement item.

**Task II (5%):**

Render 4\_your\_initials.Rmd. You can click on the “Knit html” button above the source code pane in RStudio. You can also change the output format by choosing a different Knit format option. To use the pdf\_document, you might need to first install Tex or LaTex software.