Fall 2017
BUAN 6340 – Programming for Data Science
Programming Lab #3
Market Basket Analytics – given a purchasing history of products together, recommend additional product to customers making purchases

Kevin R. Crook

Scenario

A new startup company has been selling their products online with several million sales transactions. They have hired you as a data scientist to design a prototype of a market basket analytics system. The system will look at the products the customer has placed in their online shopping cart and recommend another product.

The company has provided us with a training set of 1 million sales of 2 or more products.

For simplicity for this first prototype, they have:

- Limited their training set to a maximum of 4 products per sales transaction
- Limited individual sales to no more than 1 of each product
- Temporal reasoning should not be considered (time and date of the sale should not be considered)
- Asked us to do a simple frequency count for recommendations

The company only has 10 products. The products are named P01, P02, …, P10. Some products may be new without any sales yet.

The company has provided us with a test set of 100 online shopping carts, and has asked us to recommend 1 additional product for each of the 100 online shopping carts.

For each combination of products in the training set, calculate a frequency count for that combination.

For each online shopping cart, find the product that when combined with the items in the shopping cart has the greatest frequency in the training set. Not all products may have sales yet. If you have a shopping cart with products without sales yet, use the frequency of the combination for products that do have sales.

Download the training set of 1 million sales transactions

Your Python program should download the training set from the following link:
http://kevincrook.com/utd/market_basket_training.txt

You Python program should load and analyze this training set in order to recommend products for the test set.

The training set has no header record.

The training set has 1 million records. Each record should be considered an historical sales transaction. Each record starts with a line number (starting with 0000001 and ending with 1000000), followed by a comma separated products list. There will be 2 or 3 or 4 products per record.

Download the test set of 100 online shopping carts

Your Python program should download the test set from the following link:
http://kevincrook.com/utd/market_basket_test.txt

You Python program should load this test set, apply the analytics from the training set, and recommend 1 product for each shopping cart.

The test set has no header record.

The test set has 100 records. Each record should be considered an online shopping cart. Each record starts with a line number (starting with 001 and ending with 100), followed by comma separated products in the shopping cart. There will be 1 or 2 or 3 products per shopping cart.

Create the recommendations file

You Python program will create a file of recommendations in the local directory called
market_basket_recommendations.txt

Do not create a header record for this file.

The file must be a proper text file using utf-8 encoding, with each line (including the last line) properly terminated by a machine independent end of line character.

Each line will be 1 recommendation. The line should start with the line number from the test file. Line numbers should all be 3 digits, with leading zeroes if necessary (starting with 001 and ending with 100). Follow the line number with a comma. Follow the comma with the recommended product. Follow the recommended product with an end of line. No spaces anywhere in the file!

Individual Assignment

This assignment in an individual assignment. You may consult with other student about general approaches to solving the problem and for asking for help to resolve stack traces, but all coding must be your own work. An electronic comparison for similarities in submissions will be made. Any similarities greater than 70% will be investigated by the instructor, with possible referrals for academic dishonesty.

Python Program

You will write a single file Python program in Jupyter Notebook format named **market_basket_analytics.ipynb** to accomplish them. The program must download and read all files correctly. The program must run without stack trace. The program must create the specified output files.

Your Python code must be algorithmic in nature

Your Python code must be algorithmic in nature. Hardcoding output statements that are not algorithmic in nature is considered cheating and is explicitly listed as an act of academic dishonesty in UTD official regulations, with possible referrals for academic dishonesty.

Documentation Strings and Ratio of Source Code to Comments

In your Python code all functions, classes, and methods should have a documentation string with at least 1 line of meaningful documentation. The ratio of non-empty source code lines to comments should be no more than 5 to 1.

Submission to eLearning

You must submit to eLearning 2 files only:

- market_basket_analytics.ipynb
- market_basket_recommendations.txt

Grading Rubrics

Pass / Fail Grading with 65% Threshold

The instructor will award a grade of **pass** if 65% of the objectives of the assignment have been met.

A grade of **fail** will be given for any <u>one</u> of the following conditions:

- in the instructor's sole opinion less than 65% of the objectives of the assignment have not been met
- in the instructor's sole opinion, the solution is not algorithmic in nature (no hard coding!)
- the assignment is submitted late (1 second late is late)
- the student emails the instructor the assignment instead of or in addition to submitting it in eLearning
- required files are not submitted with the correct names (case sensitive)
- not observing the documentation string and comments ratio detailed previously
- submitting any additional materials other than the explicitly mentioned submission files
- if there is a 70% similarity or greater between any part of the submission and another student's submission from any section in any semester

Timing of Submission for Rank Grading

The submission time of this assignment may be used in tie breaking for rank grading as detailed in the syllabus. The sooner you submit the assignment, the greater the probability of a higher grade ranking.

Late Penalty

No late submissions will be accepted. A grade of **fail** will be given. Students are strongly recommended to target a completion date 2 or 3 days prior to the deadline.

Resubmission

Students may resubmit the assignment prior to the due date and time. Only the last submission will be graded. Prior submissions will not be graded. The last submission time will be counted toward rank grading.