LCBO Whiskey Recommendation Algorithm

By Nelson Siegel for CSDA1050

Research Question

There are two parts to this research question:

1. Given a whiskey that the user enjoys, how can we tell other whiskeys that they will also enjoy?

2. Based on the prices available for these whiskies at the LCBO, what is the best value purchase the user could choose that they would enjoy the most for the least cost?

Data

Whiskey Reviews

Data Source

The first part of the data will be whiskey reviews. To gather these, there is a google sheet that contains whisky names and links to reddit whiskey reviews:

Reddit Whiskey Network Review Archive

What I want to use, however, is not just the ratings but the descriptions themselves.

To gather the descriptions I will need to use the reddit api and the links in the google sheet in order to scrape the text of the reviews.

Data Gathering

There is a great python package developed to interact with the Reddit API called praw:

Documentation for PRAW python wrapper for Reddit API

This makes it easy for me to extract the content of the reviews by taking the following steps:

1. Read the csv extract of the Reddit Whiskey Network Review Archive into a dataframe

2. Use regular expressions to extract the submission ID from each link.

3. Input this submission ID into praw to interact with the submission.

4. Grab all submission content as well as any replies by the original poster in the submission (this is because sometimes the review is in a comment rather than the submissions text.

This leaves me with all text a user has in their review. Later I will need to process it to get the parts of the review that are relevant to me.

LCBO Prices

Data Source

The second part of the data is the LCBO price data. Unfortunately, a user run LCBO-api has recently shut down so I will not be able to use that. However, his github is now public. To gather the data I will use a combination of two LCBO data sources:

1. Vintages.com: This will be used to search for Product IDs based on Whiskey Names

2. FoodAndDrink.Ca: This is a real LCBO API, but it requires Product ID to get info.

Data Gathering

This is a two step process:

1. I first search on Vintages.com for the whiskey name given in the review, and record all results (Sometimes there are more than one result so I will need to double check the match up later). This step is completed using urllib2 to request the site and beautifulSoup to find the elements I need to process.

2. Next, the product IDs collected are pulled and used to query the foodanddrink.ca API. This returns and XML document that I can parse in order to extract all of the fields.

3. At this point, I realized I probably was not capturing all whiskeys due to name changes, so for future reference I have begun collecting data from all LCBO products by blindly looping through every possible product ID. Since this was running quite slowly I've parallelized the code to run on all 16 cores available on my server. This speeds it up a lot so I expect it to be finished within a couple of days, assuming I don't get blacklisted.

Here is an example link to see the XML result returned for a given product ID:
http://www.foodanddrink.ca/lcbo-webapp/productdetail.do?itemNumber=12385

Analysis Methods

In order to give proper recommendations I do not think a user-based recommendation system is appropriate. Rather I will be using NLP processes in order to determine similarities in the text. Each whiskey review is divided into three different sections:

- Nose
- Taste
- Finish

For each of these categories I will use NLP processes. The procedure will be as follows:

1. Data cleaning
2. Stemming and lemmatization
3. Topic extraction (Most likely using LDA but will consider other methods)
4. Key word identification
5. Find similar whiskies, likely using something like Word2vec or LSA to compare.

The other portion will be analysis of review ratings versus value. Most likely the difficult part of this will be normalizing the user reviews, as some users will tend to review things in certain ways, and users in general will tend towards more favorable reviews. To offset this I will attempt a two step approach:

1. For users who have above a certain threshold of reviews, I will normalize their reviews based on that user's own reviews.

2. For users who have less reviews, I will need to normalize amongst all the users who have less reviews. This will be less accurate but hopefully there will be enough data to balance out the noise.

Once these reviews are normalized I can properly aggregate them to create a "Rating vs Cost" metric which will inform all other calculations.

Interface

In order for users to interact with the results, I will need to create a user interface. At this point I expect to use a python Dash app, as they are quick to develop and there are time limitations to the project.

Limitations / Constraints

Due to the time constraints, there are some things that would be good to include but I will likely skip for now to focus on aspects I would rather learn. These are:

- Ideally the data would be stored in a database, but for this project I will instead use dataframes saved as parquet files.
- Scripts would be run each day to update prices and check for changes. In this case I will collect data at the beginning and not set up daily runs.

Timeline

Week 2: (July 9 - 15)

- Begin data collection, assembling methodology for Exploratory Data Analysis
- ETL work into data warehousing
- Begin exploring data and documenting issues/limitations/needs understanding needs/limitations regarding research question (might need to adjust question, scope, data, etc.).

Week 3: (July 16-22)

- Submission of EDA sprint (via GitHub/Moodle)
- Collect/Augment/Refine project according to Sprint 1 findings
- Develop several analytical methodologies (these should be methods you are interested in learning – like Topic modelling or sentiment analysis, or social network analysis, etc.)

Week 4: (July 23-29)

- The halfway point.

o All data collection should be completed
o The methodology should be finalized
Week 5 (July 30 - August 5)

- Submission of Sprint 2 to GitHub/.

o Includes codebase, report (brief), and plan for analysis
Week 6 (August 6 - August 12)

- Start final sprint prepare for final submission.

Week 7 (August 13 - August 19)

- Work on final project.

Week 8 (August 19 - August 27)

- Final Submission.