BRAIN FOOD

A recipe recommender system

By Corey Miller

The DATA

- **The goal is to minimize time and hassle at meal times by suggesting recipes based on ingredients.**
- **A string of ingredients is input into the system and a list of 10 recipes generated from both similar and matching ingredients will be returned.**
- **The recipe are returned in order of how well they matched the input ingredients.**

The DATA

- **The system was built from an aggregation of multiple recipe databases :**
  - **Allrecipes.com**
  - **bbc.com/food/recipes**
  - **Epicurious.com**
  - **Cookstr.com**
- **The final dataset used to build the system contained over 144K recipes!**

The BRAIN

- **The true power behind the system is driven by a term frequency – inverse document frequency (TFIDF) matrix and supplemented with a word2vec model, both created from the recipe corpus.**
- **The recipes were cleaned to prepare for the TFIDF matrix and word2vec model**
  - **Removed stop words**
  - **Removed punctuation and numbers**
  - **Removed most common non food words**
  - **Lemmatized**
- **The TFIDF matrix was built and a word2vec model was trained on the corpus**

The BRAIN

- **Ten keywords for each recipe were selected based on the highest TFIDF scores**
  - **Missing titles for recipes were filled with the top 3 keywords for that recipe**
  - **The rank order (score) for recipes returned is based on how well the input string matches the 10 keywords in each recipe**
- **Most similar words to the input string are generated with the word2vec model**
  - **The word2vec model returns the most similar word from the word2vec vocabulary (when applicable) to each word input into the system**
  - **The most similar words are added to the input string to be fed into the system with the original string – increasing diversity of results returned**

The BRAIN

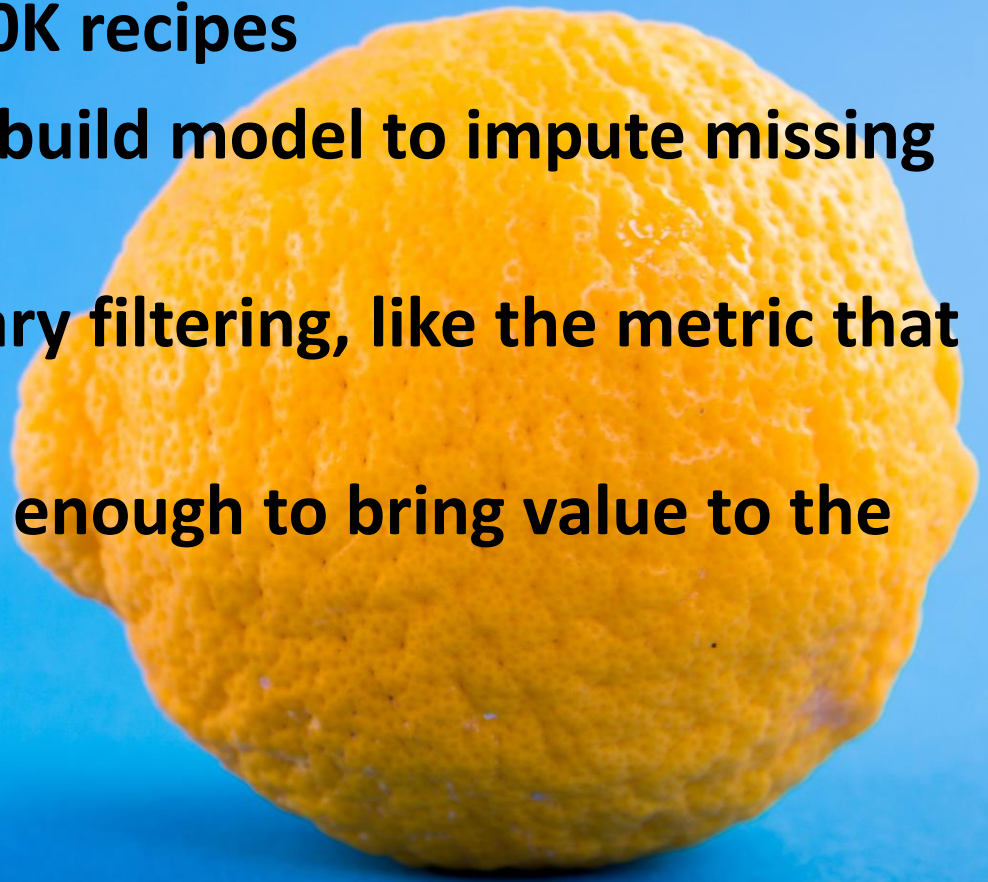
- **Once a string is input into the system, matched with its most similar words, the new string of input and similar words is vectorized**
  - **The length of the input vector is the same length of the number of words in the TFIDF matrix**
  - **One hot encoding for each word that matches from the input to the TFIDF vocabulary**
  - **The TFIDF matrix is multiplied by the input vector and the top 10 highest scores are the results**
- **The results are ranked by how well the original input matches the top ten keywords**
  - **A perfect score (1.00) represents a recipe who's keywords contain *ALL* of the original input words**

The RESULTS

- The final system was tested with 3 different custom built tests
  - Test 1 = first 10 ingredients from *n* randomly sampled recipes
  - Test 2 = 10 randomly selected words (ingredients) from cleaned text from *n* randomly sampled recipes
  - Test 3 = top 10 keywords from *n* randomly sampled recipes
- The test was to use 1000 randomly sampled recipes with the inputs from each of the tests and measure the accuracy as how many times the inputs from a recipe returned itself
- The accuracy results increased with each test
  - Test 1 = 53.2%
  - Test 2 = 66%
  - Test 3 = 100%
- Test 3 performing with 100% accuracy makes sense because the recipes are generated based on how well the inputs match the TFIDF vocabulary and the top ten keywords are selected as the highest scoring TFIDF words for each recipe

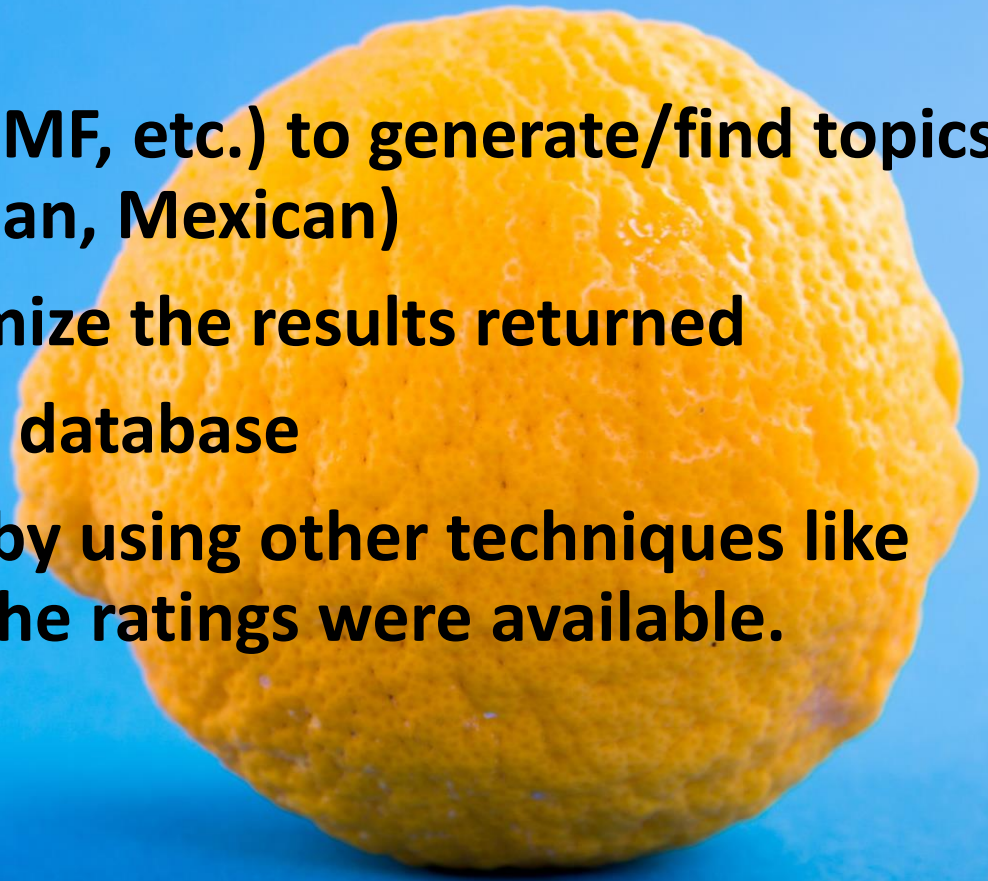
Additional thoughts

- **Ratings were available for about 100K recipes**
- **Attempted to use known recipes to build model to impute missing ratings**
- **Ratings were to be used for secondary filtering, like the metric that was created**
- **None of the models performed well enough to bring value to the system**



Additional Thoughts

- **Clustering techniques (LDA, LSA, NNMF, etc.) to generate/find topics to allow for topic searching (i.e. Italian, Mexican)**
- **Add a dithering algorithm to randomize the results returned**
- **Allow for recipes to be added to the database**
- **Further improve the recommender by using other techniques like collaborative filtering, especially if the ratings were available.**
- **Get a website and go live!**



Questions and Examples