How To Train Your Palate

Victor Treaba and Herbert Li

Introduction

Background:

- Finding recipes is easy!
- Finding *good* recipes is bit more difficult!

Problem:

- We would like to tackle the problem of telling how good a recipe is without cooking it.
- Specifically, if we can learn to correctly predict recipe ratings
- We will also explore the question of whether or not we can learn to rank recipes

Dataset

- 0 ratings: 1834 (9.43%)
- 1 ratings: 162 (0.85%)
- 2 ratings: 124 (0.65%)
- 3 ratings: 2011 (9.84%)
- 4 ratings: 13194 (65.50%)
- 5 ratings: 2710 (13.73%)

- Total recipes: 20035

Each recipe in the dataset includes:

1. Cooking Instructions
2. Ingredients
3. Rating (out of 5 stars)
4. Title
5. Nutritional Information
6. Categories/Keywords

Experiment Setup

- Initial Remarks:
    1. Data is *highly* skewed towards the higher echelon of ratings
    2. Preliminary tests with SVM showed that this would cause most “predictions” to be merely guessing the same class (in this case, 4)
    3. It’s almost completely absent of 1 and 2 star ratings, making it more difficult to accurately fit a model that would correctly predict them
- Our Fix:
    - 4 class classification model where we would push 0 and 1 ratings up to 2, (recipes rated 0, 1, and 2, are all pretty bad)
- Classification: SVM, Adaboost, and Neural Networks
    - Grid Search, 10-fold cross-validation
- Ranking case: Adaboost and SVM in a similar setup

Featurization

1. Ingredient Grouping
   a. `ALL`: ~800 features, each corresponding to a unique ingredient string
   b. `SPARSE`: ~600 features, each corresponding to a unique ingredient, here we grouped singular and plural forms together, and we grouped all alcoholic beverages together
   c. `GROUPED`: ~70 features, each corresponding to a group of ingredients that all have similar flavors (sweet, savory, bitter, etc.) and textures
2. Value of Feature
   a. `EXISTENCE`: Binary numbering (0-1) representing the presence of a given feature column (i.e. ingredient or ingredient category)
   b. `WEIGHTS`: Floating point number representing the percentage of the mass in grams of the ingredient group
3. Cooking Directions
   a. `DIRECTIONS`: ~20 features, each corresponding to a unique preparation (e.g. sauté, boil, bake, roast), also features for cooking times and oven temperatures

Results - 4-Class Scenario

- Tried to keep an even distribution 25 - 25 - 25 - 25 split
- Split classes by ratings [0, 1, 2] [3] [4] [5] combining all the bad recipes into one lower category
- Beating the distribution by at most 16%
  - Best SVM results using `GROUPED EXISTENCE` ingredients (40.95% accuracy)
  - Best Adaboost using `GROUPED EXISTENCE DIRECTIONS` (40.60% accuracy)
  - Best Neural Networks using `GROUPED WEIGHTS DIRECTIONS` (41.63% accuracy)

Results - 3-Class Scenario

- Distribution of 33 - 33 - 33 split
- Classes split into ratings of [0, 1, 2, 3] [4] [5], separating bad, average, and good
- Beating the distribution by at most 19%
    - Best SVM results using `SPARSE EXISTENCE` (52.56% accuracy)
    - Best Adaboost results were obtained by using `ALL EXISTENCE` (49.27% accuracy)
    - Best Neural Networks results using `GROUPED EXISTENCE DIRECTIONS` (46.00%)

Results - 2-Class Scenario

- Distribution was 50 - 50 split
- Recipes were in two classes, [0, 1, 2, 3] vs [4, 5] rating, essentially separating the good from the rest.
- Ended up only beating the distribution by at most 15%.
  - Best SVM accuracy was obtained using `SPARSE EXISTENCE DIRECTIONS` (65.43%)
  - Best Adaboost accuracy was obtained using `GROUPED EXISTENCE DIRECTIONS` (62.18%)
  - Best Neural Networks accuracy was obtained using `GROUPED WEIGHTS` (59.72%).
  - Use of `FREQUENCY` instead of `EXISTENCE` in no significant change in test accuracy.

Results - Ranking

- Distribution was 33 - 33 - 33 split
- Recipes were in three classes, [-1], [0], [1] rating, denoting if recipe A was worse than, equal to, or better than recipe B, respectively
- SVM and Adaboost performed roughly 10% better (43% acc) than just randomly assigning -1, 0, 1 to test points.

Analysis

- Including `DIRECTIONS` led to comparable if not better results
- Using `WEIGHTS` instead of `EXISTENCE` led to comparable if not worse results.
- Using of `FREQUENCY` instead of `EXISTENCE` resulting in no significant change in test accuracy.
- Using SVM, Adaboost, and Neural Networks we were able to predict ratings better than just randomly guessing
- Our methods were not accurate to the extent of considering this problem of rating recipes as definitely learnable
- Same can be said of the problem of ranking recipes

Challenges

- Unexpectedly low amount of recipe APIs, most don’t provide exhaustive details about the recipes. They would always miss some key information such as recipe rating, or directions.
- We tried scraping the web from website such as allrecipes.com, but after a few hundred calls we would get banned from their website.
- Predicting user preference is akin to trying to predict psychological results, where most machine learning experiments in that area have resulted in under a 50% correlation coefficient.
- A lot of data to collect, where the data is highly skewed towards high ratings.

Further Work

- More methods of featurization, possibly look at features composed of bigrams and trigrams of ingredients.
    - For example, “lamb and rosemary” might be one such bigram, and “carrots, celery, and onions” might be one such trigram. Deciding on which bigrams and trigrams to select is also something to think about.
- Lots of room to incorporate some natural language processing methods in our classification and ranking scenarios.
- Possibilities in narrowing the scope of recipe prediction. Initial concern was that the data spanned across too many dishes and recipes.
    - It would be interesting to see whether it is possible to predict recipe ratings that used chicken as their main ingredient, or even more specifically for example, predict the ratings of only lasagna recipes.

Questions?

References

1. https://www.kaggle.com/hugodarwood/epirecipes
2. N. Yu, D. Zhekova, C. Liu, and S. Kubler. Do good recipes need butter? predicting user ratings of online recipes. In Proceedings of the Cooking with Computer workshop at the International Joint Conference on Artificial Intelligence (IJCAI2013), 2013. URL: http://cl.indiana.edu/~skuebler/papers/epi13.pdf