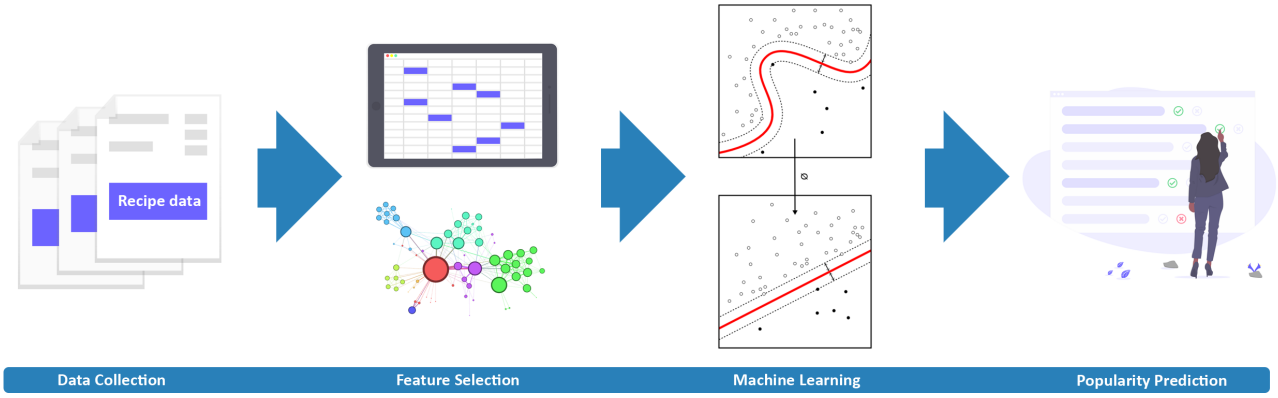
Spilling the beans: Food recipe popularity prediction using ingredient networks

F. A. Ruis
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
f.a.ruis@student.utwente.nl



ABSTRACT *APERITIF*

Food plays a central role in all of our lives, it affects our health and even our mood. There are millions of different (online) recipes to choose from, a lot of which haven't been vetted yet. This research aims to glean new insights in which features drive the popularity of recipes by way of network analysis, and use these insights to train a predictive model. While the best of a pair of similar recipes can be determined with an accuracy of 90%, a more general rating predicting proves to be a much tougher nut to crack. We haven't been able to accurately predict a general rating for recipes, but we believe we can provide some food for thought.

Keywords

food, recipe recommendation, ingredient networks, data mining, machine learning

1. INTRODUCTION *HORS D'OEUVRE*

There is a wealth of information on recipes and cooking available online. Over the last few years even larger and more complete, structured datasets have become available for use in food recipe analysis.

A lot has been done to discover how we choose to eat what we eat, but there is a lot more information to be gained. Research by Ahn et. al [1] and Teng et al. [10] show promising results in the use of networks for better understanding food and recipes.

31th Twente Student Conference on IT July 5th, 2019, Enschede, The Netherlands.

A better understanding of food, recipes and what makes them popular can aid not only recommender systems but also authors, chefs or people who want to learn how to cook. A few use cases could be predicting how well a recipe would be received, recommending ingredients to omit, add or replace, or optimizing the recipe for efficient use of ingredients the user has available in their home.

We will download a large set of recipes from online recipe sharing sites, parse their ingredient lists and build networks showing the relations between different ingredients. We will extract information about the structure of these networks and use this to train machine learning models to predict popularity measures for unrated recipes.

In a nutshell, this research attempts to find out which features are the bread and butter when it comes to food recipe popularity.

2. RESEARCH QUESTIONS *LA CARTE*

The research can be broadly summarized by the questions below and is elaborated upon in sections 3 and 5.

RQ1. Which features are the most important factors in recipe popularity?

RQ2. Can we predict the popularity of an unrated recipe?

- **RQ2.1** Can we identify the best of 2 similar recipes?
- **RQ2.2** Can we assign an overall popularity measure?
- **RQ2.3** How does this compare to related work?

3. BACKGROUND *PLATS D'ACCOMPAGNEMENT*

You are what you read; this section succinctly explains some background about the proposed methodology.

3.1 Data and Format

The data will be scraped from Allrecipes[1], the cream of the crop when it comes to food recipe data. Allrecipes has

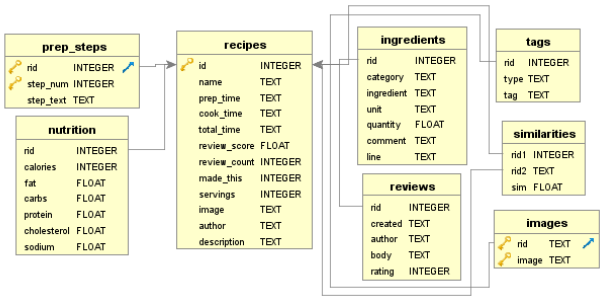
[1] `https://www.allrecipes.com/`

a large community of active users that leave reviews and upload recipes.

The data includes ingredient lines, reviews (score, user, date, text), how many people made the recipe, tags (cuisine, course, occasion), images, cooking time, servings, detailed nutritional information, preparation steps and the author.

A SQLite[2] database will be used to query the data. An ERD of the data can be seen in figure 1.

Figure 1. Entity Relationship Diagram

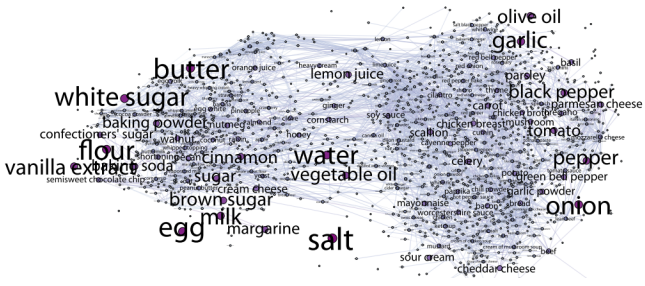


3.2 Network Science

The field of Network Science studies complex networks such as social networks, or in the case of this study, ingredient networks. An example of an ingredient network can be seen in figure 2, where two ingredients share an edge if they occur together more than would be expected by chance according to their pointwise mutual information.

A network science approach can, among other things, be used for data mining and feature extraction for machine learning methods. Teng et al. [10] used centrality measures and community structure from their networks to build feature sets that had a much lower dimensionality than a full ingredient list. This lower dimensionality addresses the sparsity problem of a full ingredient list, where often only about 10 ingredients of a list of several thousands are present in a single recipe.

Figure 2. Ingredient network by Teng et al. [10]



3.3 Machine Learning

The field of Machine Learning studies algorithms and models that can perform a specific task without using explicit instructions. Using the data from section 3.1 and the features extracted from it one can train a model to predict the popularity of a recipe.

Supervised machine learning is a subset of machine learning tasks where output functions are learned from labeled training data. In the case of this study, the labels are review scores and favorite counts. Supervised machine learning can be further divided into classification and regression methods.

Classification methods, e.g. support-vector machines[3], are used for data with discrete labels such as which in a set of two recipes is considered the best. Regression methods, e.g. linear regression[4], are used for continuous labels such as an overall popularity score.

4. RELATED WORK *AMUSE-BOUCHE*

There has not been a lot of research in the domain of food recipe recommendation specifically in recent years. There are some papers with promising results however.

Networks

Ahn et al. [1] spiced things up by using networks of shared flavor compounds in ingredients to investigate the food pairing hypothesis[5]. They show how data-driven network analysis can yield new insights into food science. Simas et al. [9] recently used the same methods to investigate a food-bridging hypothesis.

Regional cuisines

Jain et al. [8] investigated the food pairing (or lack thereof) in regional cuisines in India. They find that the regional cuisines follow a similar trend of negative food pairing and that spices play a crucial role in this trend.
Zhu et al. [11] investigate the impact of climate and geographical proximity on the similarity of regional cuisines in China. They find that geographical distance plays a key factor in how regional cuisines are shaped over time.

User-centered

Freyne and Berkovsky [3], joined by Smith [4] a year later, take a user-centered approach to recipe recommender systems. They surveyed users about their preferences on a set of recipes and investigate which factors influence a user's rating.
Ge et al. [5] built an actual recommendation system demo for Android. They take health factors into consideration and allow users to adjust the relative importance of health and taste.

The whole picture

In terms of related work, Teng et al. [10] take the cake. They built two ingredient networks, one complement network and one substitution network. The complement network shows which ingredients tend to co-occur. The substitution network is a directed network generated by analyzing user reviews, it shows which ingredients tend to be substituted by other ingredients. They have shown how centrality and community features derived from the network structure can significantly increase the accuracy of their predictions as opposed to using a full ingredient list.

[2] https://www.sqlite.org/

[3] https://en.wikipedia.org/wiki/Support-vector_machine

[4] https://en.wikipedia.org/wiki/Linear_regression

[5] https://en.wikipedia.org/wiki/Foodpairing

5. METHODS *RECETTE*

The proof is in the pudding; in this section we show our methodology for answering the research questions.

5.1 Data Acquisition

Several recipe sites have been considered to use as the subject of this research, most notably allrecipes.com, yummly.com and epicurious.com. These sites were chosen for their large volume of recipes as well as their active communities in both reviewing and uploading recipes. Yummly initially seemed as the best option because they have over 2 million recipes and 16 million active users. However, only the first couple thousand recipes get a significant amount of exposure resulting in less than 10.000 recipes that have more than 5 reviews. Allrecipes and Epicurious have a similar amount of recipes, around 60.000 vs 40.000, but Allrecipes has significantly more reviews, 3.6 million vs 700.000. Combining the datasets of several sites would result in the machine learning models learning to differentiate the source of the recipe rather than the quality (more on that in section 6). For this reason Allrecipes was chosen as the data source.

58.305 recipes have been scraped from allrecipes.com along with 3.591.506 reviews. The data includes nutritional information, serving size, cook time, tags such as course type or dietary concerns, and preparation steps. The reviews are a score from 1 to 5. An ERD of the data can be seen in figure 1.

5.2 Preprocessing

The ingredients of the recipes are given as unstructured ingredient lines, usually in the form of "quantity unit comment ingredient comment" e.g. "1 cup chopped tomatoes, as ripe as possible". There is a lot of variation possible though and it can be hard to differentiate between what is an unnecessary descriptor or an important part of the ingredient name. A melon for example is a completely different fruit from a bitter melon, but a red bell pepper can safely be seen as more or less equivalent to a green bell pepper.

The New York Times has published a dataset of roughly 180.000 ingredient lines that have been manually tagged and the model they trained for tagging ingredient lines with their quantity, unit, and ingredient [6]. Their approach uses Conditional Random Fields, which might be overkill for this application since even though the input is somewhat dependant (a different unit would result in a different quantity if you want the total amount to stay the same), that is not a relation that the model needs to learn for this application. Inference might be faster with e.g. an LSTM because of a more narrow search space, but since the run time is still manageable for the amount of data used in this research we have opted to stick with their CRF approach. An example of the CRF output (converted to JSON) can be seen in listing 1.

Listing 1. Ingredient Phrase Tagger Output

```
{
    "qty": "5",
    "unit": "cup"
    "name": "all-purpose flour",
    "other": ",",
    "comment": "or more as needed",
    "input": "5 cups all-purpose flour, or more
        ↪ as needed",
}
```

The ingredients returned by the ingredient phrase tagger are further preprocessed by stemming them using the Natural Language Toolkit's[6] PorterStemmer and by removing certain common adjectives that don't affect ingredient flavor such as 'large' or 'medium'.

5.3 Ingredient Networks

With the structured ingredient data in place it is time to build the ingredient networks. To build the networks we used the NetworkX[7] Python library.

Cooccurence Network

We counted the cooccurrences of each ingredient and added an edge between two ingredients when they occur together in at least 0.0005% of all recipes, with the *log* of their cooccurrence count as the edge weight. This results in a network with 740 nodes and 10.391 edges. A couple of different thresholds were tested, 0.0005% filtered out the ingredients that were either too verbose (e.g. "Dunkin' Donuts® Caramel Coffee Cake Artificially Flavored K-Cup® pod") or misspelled, and didn't impact the network feature performance while increasing efficiency due to fewer nodes and edges. This value will of course depend on the size of your dataset.

5.3.1 Complement Network

The complement network is similar to the cooccurrence network but its edges are based on the ingredients' pointwise mutual information (PMI). The PMI value for two ingredients x and y is calculated as follows:

$$PMI(x, y) = log(\frac{p(x, y)}{p(x)p(y)})$$

Where $p(x, y)$ is the number of recipes where x and y occur together divided by the total number of recipes, and $p(z)$ is the number of recipes containing ingredient z divided by the total number of recipes.

The PMI tells us how 'important' a connection between two ingredients is. Onions, for example, have a high cooccurrence rate with almost every other ingredient, but there are no ingredients that onions cooccur with more often than would be expected by chance. Therefore there is not much information to be gained if onions are used in a recipe. Saffron on the other hand does not occur frequently, but when it does it is usually with spanish chorizo and vice versa, giving that combination a high PMI value (much to the chagrin of Valencian Paella chefs).

For visualization purposes the threshold of when to add an edge between two ingredients was set at three times the standard deviation above the mean of all PMI values. When using the networks to extract features for the machine learning methods a threshold of one standard deviation above the mean had the best results. The weight of the edges was set to the actual PMI value.

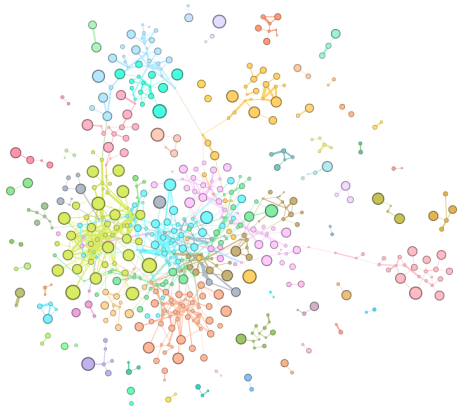
An example of the comlement network can be seen in figure 3, or by visiting the link at the end of this section. Clusters are colored by the multilevel algorithm by Blondel et al. [2]. Ingredients found in Indian, Thai, and Japanese cuisines and cake ingredients and alcoholic drinks form the 5 most densely packed clusters.

5.4 Recipe Similarity and Node2vec

To answer **RQ2.1** and to calculate some recipe similarity measures we used the node2vec algorithm by Grover et

[6] `https://www.nltk.org/`
[7] `https://networkx.github.io/`

Figure 3. Complement Network
An example of the complement network with a PMI threshold of $mean + 3 * std$, colored by cluster membership. Nodes scaled by their occurrence count, edge width scaled by PMI, isolates removed.



al.[7]. It maps network nodes to a low-dimensional space of features that maximizes the likelihood of preserving their neighborhoods by performing many random walks originating from each node in the network. These embeddings can be converted for use in a Gensim word2vec model[8]. This allows us to use all word2vec functions such as *doesnt_match* to find the odd ingredient in a list and *n_similarity* where you can input two lists of ingredients and get a cosine similarity score between 0 and 1.

The algorithm has a return hyperparameter and an in-out hyperparameter controlling the probability of the walk backtracking and choosing nodes that are close to the previous node respectively. The algorithm has been run on both networks with a return hyperparameter of 0.3 and in-out hyperparameter of 0.7 and the following regular parameters: 128 dimensions, walk length of 50, 240 walks per node. The algorithm takes the edge weights mentioned in the network sections into account.
The chosen return and in-out parameters ensure that there is an increased likelihood for the walks to backtrack as well as choose a node further away from the previous node, encouraging the walk to explore different paths but not stray too far from its origin.

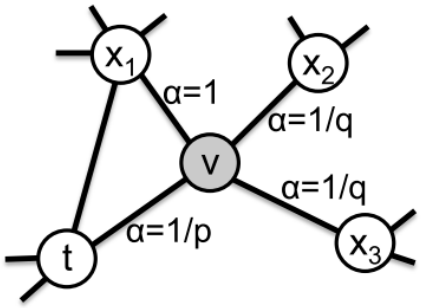
The benefit of using node2vec as opposed to simply calculating the cosine similarity of the ingredient lists directly is that even though the lists ['chicken breast', 'canned tomatoes', 'thyme'] and ['chicken thigh', 'vine tomatoes', 'rosemary'] are very similar, their cosine similarity would be 0. The node2vec algorithm results in a similarity of 0.68 on these same lists.

5.5 Recipe Pairs

For comparison's sake we used the same restrictions as Teng et al. [10] to generate recipe pairs for the recipe pair prediction task, with some important differences: We used different methods to extract ingredient information and to calculate recipe similarity, and their constraint that 50% of users should have rated both recipes would result in only 3 out of a possible 80 million combinations.

[8] `https://radimrehurek.com/gensim/models/word2vec.html`

Figure 4. hyperparameters, as seen in [7]
The parameters return p, in-out q and search bias α (dependant on p and q). The walk has transitioned from t to v and is evaluating its next step.



The pairs were generated as follows. A recipe should have at least 10 reviews by users that have reviewed at least 8 recipes in total. For all combinations of those recipes a and b, consider those where:

- Cosine similarity of a and b is strictly higher than 0.5 (as calculated by the node2vec algorithm)
- It has at least 10 reviews by users that have reviewed 8 or more other recipes
- At least 5 of those reviews are by users that have reviewed both recipe a and b
- Of those reviews, 75% of the scores for recipe a are strictly higher than recipe b
- The total aggregated review score of recipe a is strictly higher than recipe b

This results in 53.810 recipe pairs where it is safe to say that the users collectively agree that recipe a is better than recipe b.

5.6 Features

The features are a combination of recipe information, nutritional information, network structure and a node2vec similarity measure, as seen in table 2.

The basic recipe features are the servings, a boolean value indicating if it has an image, the ingredient count, the number of preparation steps, the cook time and the course type (main, dessert, appetizer, etc.).

The nutritional information features are the amount of calories, fat, carbs, protein, cholesterol and sodium in the recipe.

The network features consist of aggregated degree, betweenness, closeness, eigenvector and pagerank centrality measures, the average degeneracy (core number), and cluster membership as calculated by the Blondel et al. multilevel algorithm [2].
The centrality measures and degeneracy are aggregated by a dot product of a vector with the values for the measure and a binary vector of ones and zeroes when the corresponding ingredient is or is not present in the recipe.
The cluster membership is represented as a column for each cluster with a value between 0 and 1 for the percentage of ingredients that belong to that cluster.
The node2vec similarity measure is the average similarity of all combinations of the ingredients in the recipe.

Table 1. Precicion, Recall, and F1-score

All	Network and Nutrition	Network and Basic	Network	Nutrition and Basic	Nutrition	Basic
0.90	0.89	0.87	0.89	0.84	0.82	0.72

Table 2. Features

Basic	Nutrition	Network
Servings	Calories	Degree centrality
Image	Fat	Closeness centrality
Ingredient count	Carbs	Betweenness centrality
Prep step count	Protein	Eigenvector centrality
Cook time	Cholesterol	Pagerank centrality
Course type	Sodium	Degeneracy
		Cluster membership
		Node2vec similarity

Machine Learning Methods

To avoid putting all our eggs into one basket these features have been used to train multiple models including Random Forest classifiers and regressors, SVMs, and k-nearest neighbors and linear regression algorithms. All models were trained using the Scikit Learn[9] library. For the pair classification the features of both recipes are passed to the model with a 0 or a 1 indicating if recipe a or b is considered to be better. For the regression the features of one recipe with its aggregated review score between 1 and 5 is used to train the model.
The apple doesn't fall far from the tree, just like the other models don't fall far from the random forest in terms of accuracy. For that reason we have chosen to focus on the Random Forest, since it is both very efficient and allows us to evaluate the relative performance of all features.

We performed some hyperparameter tuning to determine the best values for the prediction tasks, which were: n_estimators = 100, min_samples_leaf = 0.0005, max_features = 0.33 for the pair prediction and n_estimators = 100, max_depth = 25, min_samples_leaf = 0.0005 for the regression.
Over 90% of reviews rate a 4 or a 5. To tackle this imbalance we have looked at oversampling the minority classes, undersampling the overrepresented classes and assigning a higher weight to minority classes.
For the general popularity measure we have also used the pair prediction model. Two approaches have been attempted:

- Splitting all recipes in groups where their review scores round to 1, 2, 3, 4 and 5 and making pairs of the recipe being tested together with the similar recipes in the aforementioned classes. We then used the verdict of the majority of these classes to determine which class the test recipe belongs to. If e.g. 90% of recipes in class 1 say the test recipe is better, and only 40% of the recipes in class 2, decreasing for each class, we would classify the test recipe as a 2.

- A tug-o-war where all similar recipes are sorted in a list and for each recipe the position of the test recipe is pushed forward or backward in the list based on whether the model thinks that recipe is better or worse. The push backwards or forwards is weighted by the current distance to the recipe, where a recipe scoring a 2 saying the test recipe is worse will have a larger impact than a recipe scoring a 4 if the test recipe position is near the end of the list. We have experimented with the starting position of the recipe and the recipe pointers (alternatingly picking recipes from the front and end of the list to compare, placing the recipe in the middle at the start, etc.).

All experiments have been repeated on a subset of the data where the recipes have been explicitly tagged as a main course to see if this would impact the ingredient networks and model accuracy.

Web App

To explore the ingredient complement network and node2vec model we have cooked up a little web app which is available on `https://food.frank-ruis.nl/viz/` until at least a couple of weeks after the conference.

6. RESULTS *PLAT PRINCIPAL*

6.1 Pair Prediction

As mentioned in section 5, 53.810 pairs of recipes were tagged with either a 0 or a 1 indicating which recipe is considered to be better and evaluated in a 2/3 train, 1/3 test split. The model has been evaluated with all combinations of feature groups from table 2. The precision and recall (and therefore also f1-score) are the same in all instances and can be seen in table 1. An interesting phenomenon is that the basic feature set improves the accuracy by roughly 2% in all instances except when paired with just the network features where they result in a 2% reduction in accuracy. The combination of nutrition and network features doesn't improve compared to just using the network features, but combining all 3 feature sets results in the best performance of 90%.

Feature importance

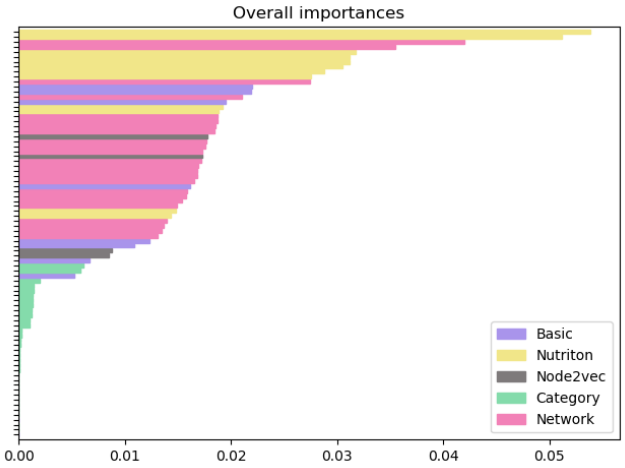
Because Random Forests are en ensemble of decision trees it is possible to inspect the feature importance, which scikit-learn implemented with the Gini Impurity approach. The importance for the combined set can be seen in figure 5. The nutrition features rank the highest and contribute 35.32% to the total importance, the basic features contribute 14.11% and the ingredient networks contribute the most at 50.57%. It quickly became apparent that the cluster features were all sizzle and no steak. They were very sparse with a high dimensionality (66 per recipe), and more often than not reduced accuracy, so they were excluded from the feature set.

Nutrition importance

The nutrition features can be seen in figure 6, including the standard deviation of the importances reported by the individual trees. Surprisingly, this is an almost complete reversal of the nutrition importances found by Teng et al. [10] on the same data set 8 years ago. While in their findings fat scores the lowest closely followed by sodium, they are the 2 most important features in our set.
Since high fat and sodium content are often seen as indicators unhealthy food we explored the data and decision tree paths for these features. The recipes in the 'better' set have 14.52% more fat and 19.82% more sodium than those in the 'worse' set, hinting at unhealthier recipes scoring higher. On the other hand, following the decision paths in

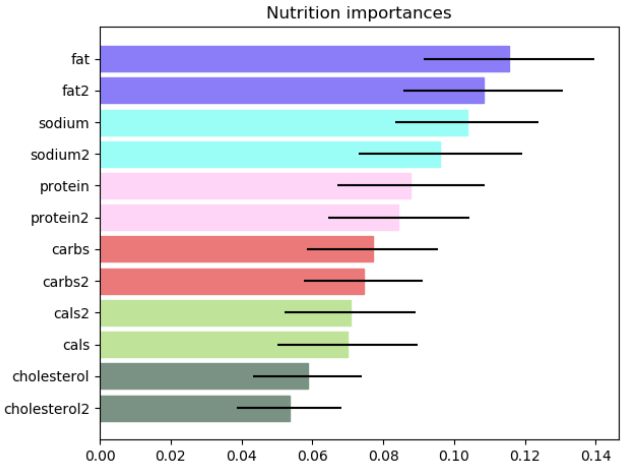
[9]https://scikit-learn.org/

Figure 5. Feature importance
Feature importance breakdown for the combined set.



all trees in the forest and keeping track of the lower and upper bounds for the fat and sodium values that are being evaluated suggests it doesn't matter which value is higher. These upper and lower bounds tell us what the decision tree knows about these values, e.g. the fat content for the recipe classed as 'better' is between 5 and 30 and that of the worse recipe between 0 and 20. For fat about 47% of the 'better' recipes have a higher average lower and upper bound, and 45% for sodium. The average lower bound of fat for the better recipes is 9.20 and 9.21 for the worse recipes. The upper bound is 13.93 and 13.92 respectively. Similarly the values for sodium also fall within 0.1 of each other. This indicates that the model does not see which value is higher, but more how they relate to other features.

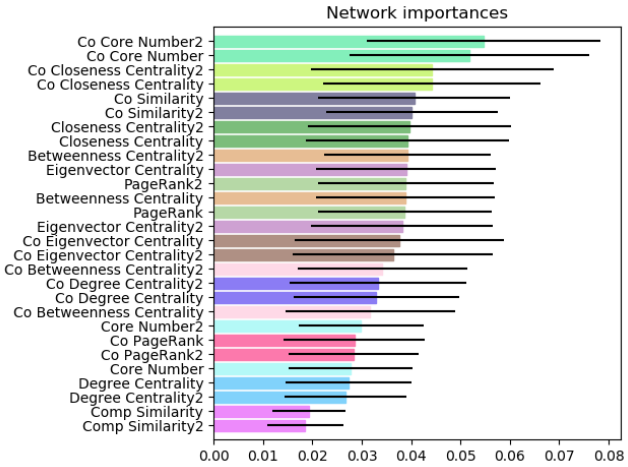
Figure 6. Nutrition importance
Feature importance breakdown for the nutrition set.



Network importance

A breakdown of the importance of the network features can be seen in figure 7. These features have a strikingly high variance. Since the Random Forest only considers a third of all features per split this suggests that they perform better or worse based on what other features they are paired with. The cooccurrence graph provides 53.88% of the importance, the complement graph 46.12%. The degeneracy and closeness centrality are the most influential features, followed by the cooccurrence network similarity score as calculated by the node2vec algorithm in third place.

Figure 7. Network importance
Feature importance breakdown for the network set.



6.2 Intermezzo

When looking at a scatter plot of the recipe review scores and their increment id, as seen in figure 9, an interesting correlation can be seen where the lower the increment id, the lower the average score. The earliest 5000 recipes have an average review score of 4.2 while the latest 5000 recipes score a 4.5. Performing Welch's t-test on a sliding window of pairs of 5000 recipes over the entire recipe set results in 2 out of 8 instances with a p-value above 5% (0.08 and 0.70), with the remaining p-values in the range $[3.31 \times 10^{-26}, 1 \times 10^{-4}]$.

The first gap in the increment ids can be explained by the fact that Allrecipes consisted of 38 separate websites with each website hosting a different type of food recipes such as cookies, chicken, cake, etc., which later migrated into Allrecipes. The other gaps can be explained by another site redesign around 2006 and our webscraper crashing near the end of one of the sitemaps, missing a couple recipes.

The earliest recorded review was placed in 1998, over 20 years ago. This plot shows how the user base and their voting habits have changed over time, and highlights a potential problem when attempting to use the aggregated review score of a recipe as a ground truth for an overall popularity measure.

6.3 General popularity

The general popularity approach is similar to the pair prediction, but it looks at all recipes with more than 5 reviews and does regression on their aggregated review score. The most accurate in this case was again the combined feature set. Sadly, our methods were about as effective as reading tea leaves. After extensive hyperparameter tuning and feature engineering the highest attained accuracy was an r2-score of 0.12 with a mean squared error that is roughly 9% better than always guessing the mean of all recipe scores. The learning curve for this and other approaches can be seen in figure 8.

The model performed slightly better (r2 0.16) when including the increment id, which prompted the investigation in the previous section. To control for recipe age we grouped

Figure 8. Learning Curves

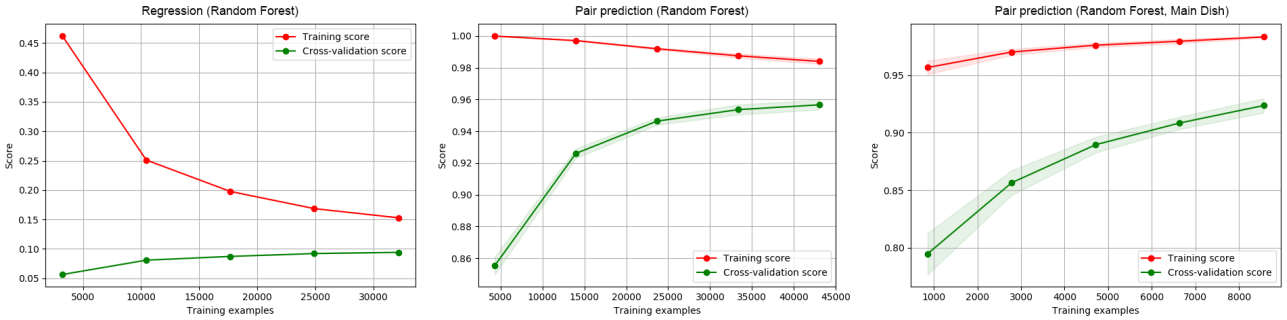


Figure 9. Review score vs Increment id
The older the recipe, the lower the average score. Some guesses as to what may explain the shift in score distribution are highlighted at the top.

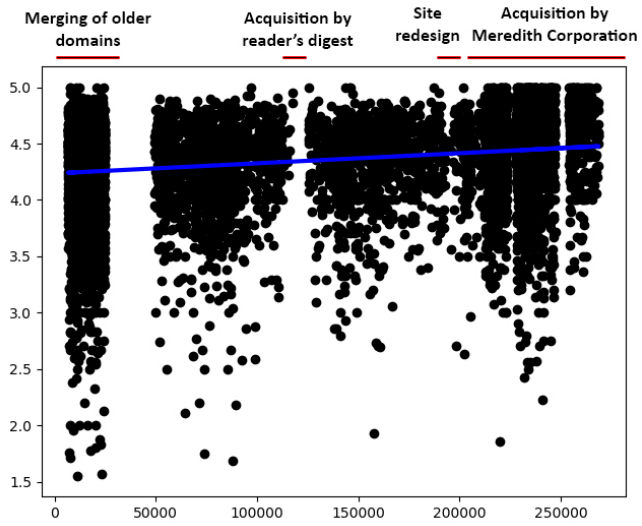
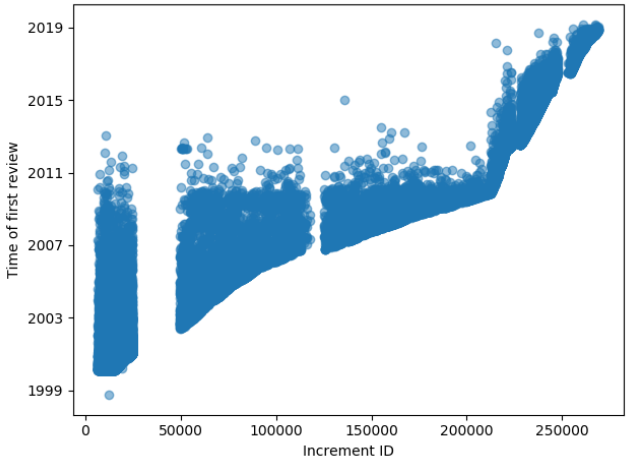


Figure 10. Time of first review vs Increment id



the recipes in groups of 2 years, their age guessed by following the line as seen at the bottom of figure 10. This did not result in an increase in accuracy, with the learning curves flattening out indicating that it is not a case of too little data.

The next approach attempted to use the pair prediction model to determine the score of a recipe. The reasoning was that a 90% chance at determining the best of two similar recipes should allow us to compare a new, unseen recipe with all similar known recipes and assign a score based on these verdicts. This seemed promising in a few cherry-picked examples but further testing resulted in an accuracy worse than assigning a random number between 1 and 5, and a precision of 15% when treated as a classification method.

Controlling for dish type

The repeat experiments where only recipes tagged as 'Main Dish' were taken into consideration (mainly affecting the ingredient network structure) did not yield hard results. The accuracy was fairly similar in all instances, with the learning curve (figure 8) suggesting that more data would put it around the same performance as including the entire set of recipes.

7. DISCUSSION *DESSERT*

The model can accurately predict user preference for two similar recipes given that there is enough overlap in users that have reviewed both recipes. The highest accuracy model outperforms that of the related work by Teng et al. [10] (90% vs 79%), though due to the time difference of 8 years between the data sets and not being able to replicate their exact approach in generating the recipe pairs this should be taken with a pinch of salt. Due to time constraints we were not able to include a substitution network which complemented their feature set well, so some improvements may be had there.

The node2vec model is excellent at recognizing similar recipes where a cosine similarity calculation on the ingredient arrays would fail to see the connection. The cooccurrence network similarity score was among the top performing features. Anecdotal evidence suggests that the node2vec model can perform the same function as the substitution network mentioned above, suggesting matching and alternative ingredients when given a list of ingredients and determining the least fitting ingredient, but that would require further investigation.

The aggregated review score on its own does not seem to be an adequate ground truth for predicting a general recipe rating, or the features that perform well on pair prediction don't necessarily translate to a general prediction. The dataset is very imbalanced, with the vast majority of reviews rating either 4 or 5 stars, a phenomenon which occurs on the Yummly and Epicurious datasets as well, making it more difficult to train the model to detect

the lower ratings. The pair prediction model has a much stronger ground truth (a group of the same users collectively agreeing), but falls flat when used in general since it only works when we know for certain that one recipe is significantly better than the other.

It seems a different approach is needed to tackle the prediction of a general score. However, these features and methods could be very useful for determining a personal user-centered preference through collaborative filtering, since that is what the pair prediction model is effectively doing.

8. CONCLUSION *DIGESTIF*

We have shown how to build ingredient networks and use them to extract features for use in machine learning methods. We have seen how users' voting habits can change drastically over time, and that we may have bitten off more than we can chew with the general popularity measure prediction task. The pair prediction might not be as useful for cold-start recommendations as it first seemed, but it could perform well in a more user-centered approach.

In the future we would investigate the use of these ingredient networks in a personalized recommendation system, and experiment more with features extracted from the node2vec model.

9. ACKNOWLEDGEMENTS

I want to thank Doina Bucur for her contagious enthusiasm and for always being available for support despite her busy schedule.

10. REFERENCES

[1] Y. Y. Ahn, S. E. Ahnert, J. P. Bagrow, and A. L. Barabási. Flavor network and the principles of food pairing. *Scientific Reports*, 1:1–7, 2011.

[2] V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. mar 2008.

[3] J. Freyne and S. Berkovsky. Intelligent food planning: Personalized recipe recommendation. *Proceedings of the 15th international conference on Intelligent user interfaces - IUI '10*, page 321, 2010.

[4] J. Freyne, S. Berkovsky, and G. Smith. Recipe recommendation: Accuracy and reasoning. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6787 LNCS:99–110, 2011.

[5] M. Ge, F. Ricci, and D. Massimo. Health-aware food recommender system. *RecSys 2015 - Proceedings of the 9th ACM Conference on Recommender Systems*, pages 333–334, 2015.

[6] E. Green. Extracting Structured Data From Recipes Using Conditional Random Fields - The New York Times, 2015.

[7] A. Grover and J. Leskovec. node2vec: Scalable Feature Learning for Networks.

[8] A. Jain, N. K. Rakhi, and G. Bagler. Analysis of food pairing in regional cuisines of India. *PLoS ONE*, 10(10):1–17, 2015.

[9] T. Simas, M. Ficek, A. Diaz-Guilera, P. Obrador, and P. R. Rodriguez. Food-bridging: a new network construction to unveil the principles of cooking. 4(June):1–9, 2017.

[10] C.-Y. Teng, Y.-R. Lin, and L. A. Adamic. Recipe recommendation using ingredient networks. 2011.

[11] Y. X. Zhu, J. Huang, Z. K. Zhang, Q. M. Zhang, T. Zhou, and Y. Y. Ahn. Geography and similarity of regional cuisines in China. *PLoS ONE*, 8(11):2–9, 2013.