From Pantry to Plate: Predicting Recipe Properties from Ingredients and Instructions

A Data-Driven Study Using the Extended Recipes Dataset

Dylan OBrien
School of Data Science
Wentworth Institute of Technology
Boston, MA USA
obriend7@wit.edu

ABSTRACT

This project investigates how the text of ingredient lists and short cooking instructions can be used to explain and predict key properties of recipes, and to help home cooks decide what to make with the ingredients they already have. Using the publicly available *Extended Recipes Dataset: 64K Dishes*, which contains over 62,000 recipes enriched with cuisine labels, taste profiles, dietary tags, and health indicators, the work models relationships between ingredients, directions, and outcomes such as primary taste, difficulty level, and approximate preparation and cooking time. Exploratory data analysis is first performed to summarize the structure of the dataset and the distribution of tastes, cuisines, and health profiles across recipes. Supervised machine learning methods, including linear models and tree-based ensemble models, are then applied to predict taste and difficulty from textual and numeric features. To address the practical "pantry problem," a simple recipe recommendation system is designed that, given a list of available ingredients, returns recipes whose ingredient profiles are most similar. Throughout the project, emphasis is placed on interpretable models and feature importance analysis to understand which ingredients and steps drive taste, time, and difficulty. The resulting system functions both as a case study in text-driven machine learning for food data and as a tool for planning meals with fewer leftover ingredients.

KEYWORDS

Recipe recommendation, natural language processing, supervised learning, text mining, food computing.

1 Introduction

Online recipe platforms have made it easy to search for dishes by title or cuisine, yet many home cooks still face a familiar problem: they look into their pantry or refrigerator and are unsure what they can make with the ingredients on hand. Standard search tools rarely allow users to start from a detailed ingredient list, account for preparation time or difficulty, and consider taste preferences and dietary constraints at the same time. This gap motivates the use of machine learning, that is, data-driven methods that automatically learn patterns from past examples, to better connect ingredients and short textual instructions with the properties and outcomes that matter to cooks.

Recent work in food computing and recipe recommendation has shown that text-based features such as ingredient lists and instructions can be effective signals for predicting cuisine and flavor profiles, and for suggesting similar recipes. At the same time, modern recipe datasets now include rich annotations about diet (for example, vegan or gluten-free), healthiness, and taste. These annotations make it possible to move beyond simple keyword search toward systems that can estimate how "sweet" or "spicy" a dish will be, how long it will take to prepare, and how difficult it is to cook, based purely on its text. Natural language processing (NLP), the area of machine learning focused on analyzing human language, provides tools to convert this text into numerical features that can be used by predictive models.

In this project, *the Extended Recipes Dataset: 64K Dishes* is used, a large collection of recipes enriched with cuisine tags, taste labels, dietary flags, and health-related indicators, to study how ingredient lists and directions relate to these key recipe properties. The analysis focuses on a set of practical questions: which ingredients and instructions are most predictive of a recipe's primary taste (for example, sweet, spicy, or savory); whether preparation and cooking times can be approximated from the structure of a recipe and its text; how well difficulty level (easy, medium, hard) can be predicted based on ingredients, number of steps, and estimated times; and whether it is possible to recommend feasible and appealing recipes based on a user's pantry while respecting time, taste, and dietary constraints.

The contributions of this work are threefold. First, a structured exploratory analysis of the extended recipes dataset is provided,

[*]Article Title Footnote needs to be captured as Title Note
[†]Author Footnote to be captured as Author Note

WOODSTOCK'18, June, 2018, El Paso, Texas USA

highlighting its coverage across tastes, cuisines, dietary patterns, and health indicators. Second, supervised learning models are designed and evaluated for predicting taste, time, and difficulty from a combination of text-based and numeric features. These models include logistic regression, a linear classification technique that estimates probabilities for discrete classes, and tree-based ensembles such as random forests, which combine many decision trees to produce more robust predictions. Third, a pantry-aware recommendation prototype is built that uses text similarity between a user's ingredient list and recipe ingredient lists to return suitable recipes. Together, these components form a small but complete end-to-end system that demonstrates how machine learning can help bridge the gap between available ingredients and concrete cooking decisions.

2 Data

This section describes the dataset used in the project, including its source and main characteristics. The dataset is stored in a single comma-separated values (CSV) file and contains approximately 62,000 recipes with a rich set of textual, numeric, and categorical features. Each row corresponds to one recipe, with columns describing the recipe title, category and subcategory, ingredients, directions, estimated times, and various derived attributes.

The core textual fields include the original ingredient list and directions as arrays, as well as preprocessed versions such as *ingredient_text*, *directions_text*, and *combined_text* where the text has been lowercased and cleaned. Structural fields such as *num_ingredients*, *num_steps*, *est_prep_time_min*, and *est_cook_time_min* capture the number of ingredients, number of instruction steps, and estimated preparation and cooking times in minutes. Additional columns encode cuisine and course information, for example *cuisine_list* and *course_list*, taste profiles through fields such as tastes, *primary_taste*, and *secondary_taste*, and dietary properties including indicators like *is_vegan*, *is_vegetarian*, and *is_gluten_free*. Health-related attributes such as *healthiness_score*, *health_flags*, *health_level*, and *dietary_profile* summarize nutritional aspects and flag characteristics like processed or fried foods. This combination of text, counts, and categorical labels makes the dataset well suited for both descriptive analysis and predictive modeling.

2.1 Source of dataset

The recipes used in this project come from the *Extended Recipes Dataset: 64K Dishes*, hosted on Kaggle, a widely used platform for sharing datasets and data science projects. The dataset was created by augmenting an existing collection of roughly 64,000 recipes with additional features that capture cuisine types, taste characteristics, dietary labels, and health-related indicators such as healthiness scores and flags for processed or fried foods. Kaggle's public review and versioning system, along with the detailed dataset description provided by the creator, make this a credible and transparent source of data for academic work. For this project, the most recent version of the dataset was downloaded and used in its original form, with only minor preprocessing such as parsing list-like fields performed within the analysis code.

2.2 Characters of the datasets

The extended recipes dataset contains on the order of 62,000 rows and several dozen columns. Each row represents a single recipe, and the columns can be grouped conceptually into textual, structural, taste and cuisine, dietary and health, and categorical summary fields. The textual fields include the recipe title, description, ingredients, and directions, along with cleaned counterparts intended for natural language processing. The cleaned fields are particularly useful because they contain lowercased, standardized text that can be converted directly into numerical features.

Structural and time-related fields such as *num_ingredients* and *num_steps* count the number of ingredients and instruction steps. The distribution of these counts reveals both very simple recipes and more complex dishes with long ingredient lists and multiple steps. The estimated preparation and cooking time fields (*est_prep_time_min* and *est_cook_time_min*) exhibit a skewed distribution typical of real-world cooking, with many moderate-length recipes and a long tail of slower dishes. Summary statistics for these fields are examined in the exploratory analysis.

Taste and cuisine fields provide labels such as "sweet," "spicy," "savory," "sour," "bitter," "umami," and "neutral" in the tastes column, while *primary_taste* and *secondary_taste* give single-label summaries. The *cuisine_list* field stores one or more cuisine tags per recipe, covering a wide variety of global cuisines including American, Italian, Indian, Mexican, Mediterranean, Middle Eastern, and many others. A similar multi-label structure is used for *course_list*, which identifies whether a recipe is, for instance, a main dish, side dish, dessert, or appetizer.

Dietary and health fields include Boolean columns such as *is_vegan*, *is_vegetarian*, *is_halal*, *is_kosher*, *is_nut_free*, *is_dairy_free*, and *is_gluten_free*, which capture whether a recipe satisfies common dietary constraints. The *dietary_profile* field combines these indicators into a list of diet tags per recipe. The *healthiness_score* column provides a numerical score, roughly on a 0–100 scale, summarizing the nutritional quality of a recipe. The *health_flags* and *health_level* fields give more interpretable categories, such as whether a recipe is considered "healthy," "moderate," or "unhealthy," and whether it includes elements like fried foods, whole grains, or processed meats. Categorical summary fields such as category, subcategory, and *main_ingredient* group recipes into broader types or highlight the central ingredient.

Overall, the dataset is relatively clean, with most key fields populated for the majority of recipes. Its main strengths for this project are the detailed ingredient and instruction text, the rich set of labels for taste, diet, and health, and the presence of time and

difficulty information, all of which can be connected through machine learning models.

3 Methodology

The modeling approach in this project consists of four main stages: preprocessing and feature extraction from the recipe text and structured fields, defining supervised learning tasks for predicting taste, time, and difficulty, training and comparing several predictive models, and building a similarity-based recipe recommendation system to support pantry-based queries. The chosen techniques are intended to be expressive enough to capture interesting patterns, yet remain interpretable and computationally manageable.

The overall project is framed as a set of related prediction and recommendation tasks. The first task is taste prediction, in which the goal is to predict the *primary_taste* of a recipe, such as sweet, spicy, or savory, from its ingredient text and directions. This is a multi-class classification problem where each recipe is assigned to one of several discrete categories. The second task is difficulty prediction, where the aim is to infer the difficulty label (easy, medium, hard) based on ingredients, number of steps, estimated times, and other features. This is also a multi-class classification problem focused on approximating how challenging a recipe feels to a home cook. A third task is time estimation, which treats *est_prep_time_min* and *est_cook_time_min* as numeric targets and models them using regression techniques. In this case, the intention is not to predict time exactly, but to provide a reasonable approximation that can help users filter recipes. Finally, the project considers pantry-based recommendation, in which a list of available ingredients is used as a query to recommend recipes that use many of those ingredients and satisfy optional constraints such as maximum prep time, difficulty, or dietary requirements. This last task is treated as a similarity search in an ingredient feature space.

Because the central features are ingredient lists and directions, natural language processing plays a key role in converting text into numerical representations. The analysis begins with the preprocessed fields *ingredient_text* and *directions_text*, which already contain lowercased and cleaned text. Additional cleaning removes obvious punctuation artifacts and tokenizes the text into words or simple word-like units. For a first representation, a bag-of-words model is employed, which records the counts of tokens in each recipe and ignores word order. This representation is simple but effective for capturing which ingredients and terms are present. Term frequency–inverse document frequency (TF–IDF) features are then computed, a standard weighting scheme in information retrieval and NLP that down-weights very common words and up-weights distinctive ones. This produces a sparse numeric vector for each recipe that reflects which terms are most informative. These text-based features are combined with numeric features such as *num_ingredients*, *num_steps*, and estimated times to form a joint feature vector for each recipe.

For the classification and regression tasks, several supervised learning algorithms are considered. Logistic regression serves as a baseline linear classification model that estimates the probability of each class as a function of the input features. Despite its name, logistic regression is used for classification rather than regression and is attractive because the learned weights can be interpreted as indicating how strongly specific features push the prediction toward particular classes. Decision trees provide a more flexible model that splits the feature space into regions by asking a series of yes or no questions about the features, which can sometimes be read as a flowchart of conditions. However, a single decision tree can easily overfit the training data, so random forests are also employed. Random forests are ensemble models that train many decision trees on different random subsets of the data and features and then average their predictions; they tend to be more accurate and robust than single trees while still providing feature importance scores. Gradient boosting models, such as gradient boosting classifiers and regressors available in scikit-learn, are also explored. These models build an ensemble of small trees in a stage-wise fashion, with each new tree focusing on correcting the errors of the previous ones, and can capture complex patterns with relatively modest model size. For regression tasks such as time estimation, regression versions of these models, including linear regression and gradient boosting regressors, are used. Performance is evaluated using metrics such as accuracy and macro-averaged F1-score for classification, and standard error measures for regression, with train–validation–test splits or cross-validation to obtain reliable estimates.

To support the pantry-based recommendation use case, a simple similarity-based system grounded in the TF–IDF representation of ingredients is designed. Given a user's ingredient list, that list is converted into the same TF–IDF space used for recipe ingredients, treating the list as a "query recipe." Cosine similarity, a vector-based similarity measure that compares the angle between two feature vectors, is then computed between the query vector and each recipe's ingredient vector. Higher cosine similarity indicates more similar ingredient profiles. Recipes are ranked by similarity score, and optional filters on maximum preparation time, maximum difficulty level, or dietary requirements are applied. The system then returns the top-ranked recipes as recommendations, along with their estimated taste profile, difficulty, and time. This k-nearest neighbors style approach, where neighbors are recipes most similar in the ingredient feature space, is straightforward to implement and does not require additional training beyond fitting the TF–IDF model. It aligns well with the practical scenario of making the best use of what is already in the pantry and forms the basis for the final recommendation component of the project.

3.1 Heading Level 2

3.2 Heading Level 2

…

Example format: The updated template, user manuals, samples, and required fonts, all are available at the URL https://www.acm.org/publications/proceedings-template. It contains said information for all three versions of MS Word (Windows and 2 versions of Mac). There are also separate links to the user guide, which can be referred to by the user. This URL also contains some useful video links, which describe how to add the template, structure the paper, and generate the layout, in different clips. **Display Formula with Number**

$$\sqrt{b^2-4ac}\,\frac{-b\pm\sqrt{b^2-4ac}}{2a} \tag{1}$$

Continuation part of Paragraph Text The user must style this paragraph in **ParaContinue** style, which follows immediately after the **DisplayFormula** (numbered equation). The **DisplayFormula** style is applied only in case of a numbered equation. A numbered equation always has a number to its right. Insert paragraph text here. **Display Formula without Number**

$$\sqrt{b^2-4ac}\,\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

The **DisplayFormulaUnnum** style is applied only in case of an unnumbered equation. An unnumbered display equation never contains an equation number to its right, and this unique property distinguishes it from a numbered equation.

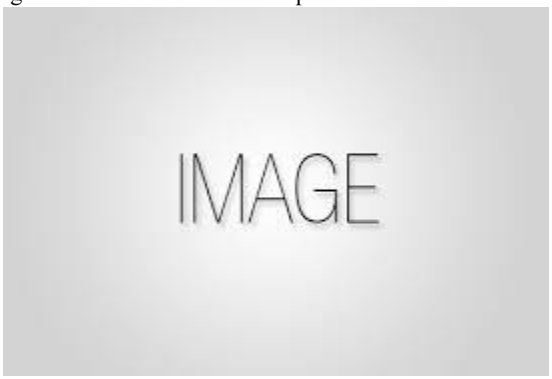

Figure 1: Figure Caption and Image above the caption [In draft mode, Image will not appear on the screen]

Theorem/Proof/Lemma. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement. Insert text here for the enunciation or Math statement.

> ....Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract, Insert text here for the Quotation or Extract.

4 Results

In this part, you need to select a reasonable way to deliver the result of your topic. For example, equation or numerical results, or visualization of your result. You also need to provide a clear explanation of all results and how to understand the results. If there exist any unexpected results, please explain why or possible cause of this special result. You can use subsection 4.1, 4.2, … to separate your results.

4.1 Heading Level 2

Example format: In the below paragraph, it is explained how alt-txt value is placed in **MS Word 2010**. To add alternative text to a picture in Word 2010, follow these steps:

1. In a Word 2010 document, insert a picture.
2. Right click on the inserted picture and select the **Format Picture** option.
3. Select the **Alt Txt** option from the left-side panel options.
4. In the "Title:" and "Description:" text boxes, type the text you want to represent the picture, and then click "Close".

Below are steps to place alt-txt value in **MS Word 2013/2016**. To add alternative text to a picture in Word 2013/2016, follow these steps:

1. In a Word 2013/2016 document, insert a picture.
2. Right click on the inserted picture and select the **Format Picture** option.
3. In the settings at the right side of the window, click on the "Layout & Properties" icon (3rd option).
4. Expand **Alt Txt** option.
5. In the "Title:" and "Description:" text boxes, type the text you want to represent the picture, and then click "Close".

1.1.1 Heading Level 3. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

1.1.1.1 Heading Level 4. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here. Insert paragraph text here.

5 Discussion

Every method/project has its shortage or weakness. Please discuss the unsatisfied results in your project. And discuss the feasible suggestions of future work to revise/improve your result.

6 Conclusion

In this part, you should summarize your project. What important results did you find for your topic and what's the effect of this result on the real-world?

ACKNOWLEDGMENTS

Insert paragraph text here. Insert paragraph text here.

REFERENCES

Use the following ACM Reference format for your citation

FirstName Surname, FirstName Surname and FirstName Surname. 2018. Insert Your Title Here: Insert Subtitle Here. In *Proceedings of ACM Woodstock conference (WOODSTOCK'18). ACM, New York, NY, USA, 2 pages.* https://doi.org/10.1145/1234567890

[1] Patricia S. Abril and Robert Plant, 2007. The patent holder's dilemma: Buy, sell, or troll? *Commun. ACM* 50, 1 (Jan, 2007), 36-44. DOI: https://doi.org/10.1145/1188913.1188915.

[2] Sten Andler. 1979. Predicate path expressions. In *Proceedings of the 6th. ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages (POPL '79)*. ACM Press, New York, NY, 226-236. DOI:https://doi.org/10.1145/567752.567774

[3] Ian Editor (Ed.). 2007. *The title of book one* (1st. ed.). The name of the series one, Vol. 9. University of Chicago Press, Chicago. DOI:https://doi.org/10.1007/3-540-09237-4.

[4] David Kosiur. 2001. *Understanding Policy-Based Networking* (2nd. ed.). Wiley, New York, NY..