# CookSmart

Marc Mascarenhas, Sandeep Tiwari and Sandhya Tharanian

Master of Science in Data Science, University of Washington

DATA 515: Software Design for Data Science

Bernease Herman & Mark Friedman

March 2, 2021

**Table of Contents**

# Functional Specification

## Background

Many factors influence the meals being cooked at home today. People are always on the lookout for inspiration to cook something new provided they have the time, interest, creativity and the ingredients at hand. Knowing what different recipes one can cook given a list of ingredients at hand is one of the constant and colossal challenges in life.

However, with tons of cooking inspiration and exponential availability of recipes online, preparing a meal with available ingredients can become easier than one can think. Giving a twist to the traditional cooking approach where you search for what to cook and then create a list of ingredients to buy them at the grocery, our proposal was to reverse and combine these methods. By using the ingredients at hand and searching for related recipes, users not only save time but also discover recipes that incorporate heterogenous ingredient pairings allowing them to uncover recipes they otherwise wouldn't.

Our project aims to address this idea via a recipe recommendation system. The recommendation system will perform unsupervised topic modeling on the recipes by vectorizing the recipe name, ingredients and user-input ingredients list using TF-IDF. The vectorized representations are then provided as an input to the LDA model to generate a topic probability distribution. Finally, a similarity score is generated for each recipe in the dataset based on the user-input ingredients list and the recipes are recommended based on highest scores.

## Goals

The goal of this project is to use the dataset available in kaggle to build a recipe recommendation system that allows users to search for recipes based on the ingredients list.

- Given any recipe dataset, build a model to find most relevant recipes that match the user's ingredients list.
- Perform unsupervised Topic Modeling on the recipes to group recipes into topics.
- Rank the ingredients based on its frequency i.e. each successive ingredient in the list is weighted incrementally less.
- Create a search algorithm that utilizes similarity scoring to rank recipes according to the greatest similarity to the user-input list of ingredients and returns recipe recommendations based on the scores.

# Users

Users can range from developers to researchers, students or anyone with some proficiency in programming knowledge. When our library is integrated with a web interface, it can be used by users with different levels of cooking experience and people who cook often or occasionally. For the purpose of this project, we are scoping to 3 main types of personas described below.

**Persona #1:**

| **ZURI**<br><br>“So, what to cook with these?” | **BEHAVIOUR**<br><br>● Uses well-known recipes<br>● Searches new recipes on the internet<br>● Doesn’t have time or passion to cook<br>● Often in situations where she doesn’t know what to cook. |
|---|---|
| **FACTS**<br><br>● 22 years old<br>● Full time employee and a part-time student | **GOAL**<br><br>● Save time searching for recipes based on a list of ingredients<br>● Find relevant and interesting recipes |

**Persona #2:**

| **XANDER**<br><br>“Can I incorporate the existing libraries to build a web interface?” | **BEHAVIOUR**<br><br>● Loves to develop web based apps using existing libraries or API<br>● Prefers to focus on the front-end rather than the backend. |
|---|---|
| **FACTS**<br><br>● 25 years old<br>● UX/UI Developer at a start-up | **GOAL**<br><br>● Proto-typing user friendly UI<br>● Save time by using existing libraries to build out a web interface or an app |

**Persona #3:**

| **MATTEO**<br><br>“Can I use the existing models to incorporate additional features?” | **BEHAVIOUR**<br><br>● Likes to play around with existing models in github<br>● Adds additional features to the models |
|---|---|
| **FACTS**<br><br>● 34 years old<br>● Data analyst transitioned to a Data Scientist in a renowned company | **GOAL**<br><br>● To understand how the existing model works.<br>● Learn by incorporating additional features to the existing libraries |

# Data sources

The kaggle dataset[1] used for our recommendation system was created by scraping AllRecipes.com, a popular social network recipe site. With 49,085 recipes, the dataset represents a varied range of cuisines and tastes. For each recipe, the dataset provided the following information.

- Recipe ID: int
- Recipe Name: str
- Average Rating: int
- Image URL: str
- Review Numbers: int
- Ingredients: str
- Cooking Directions: dictionary
- Nutritions: dictionary
- Reviews: dictionary

# Use Case

1. Zuri wants to be able to search for recipes based on a set of ingredients. She wants some recipe recommendations given a list of input ingredients.

   Users would like to get recommendations for recipes based on the ingredients. The actors in this use case are anyone with basic programming knowledge in Python and an environment to import and run the Python libraries. The recommender class will take in a list of comma separated ingredients through a

calling function. And upon initiation, the function will present the user with a list of recommendations along with the cooking instructions.

2. Xander, a UX/UI developer, wants to build a user friendly web interface for recipe recommendations. He wants to use the existing libraries out there as his expertise is not around ML algorithms.

   Users would like to incorporate the libraries to build applications which lists the recommendations. The actor in this use case is a UX/UI developer who would integrate the libraries with a front-end that he/she has developed.

3. Matteo wants to enhance his skill set by fine tuning the existing algorithms and adding additional features to learn how the model performs.

   Users would like to be able to use the existing model and scale it to include additional features such as ratings or cuisine type or cuisine category (veg vs non-veg). The actor in this use case is a self taught data scientist who would like to learn how the model works and improve it. the model.

# Component Specification

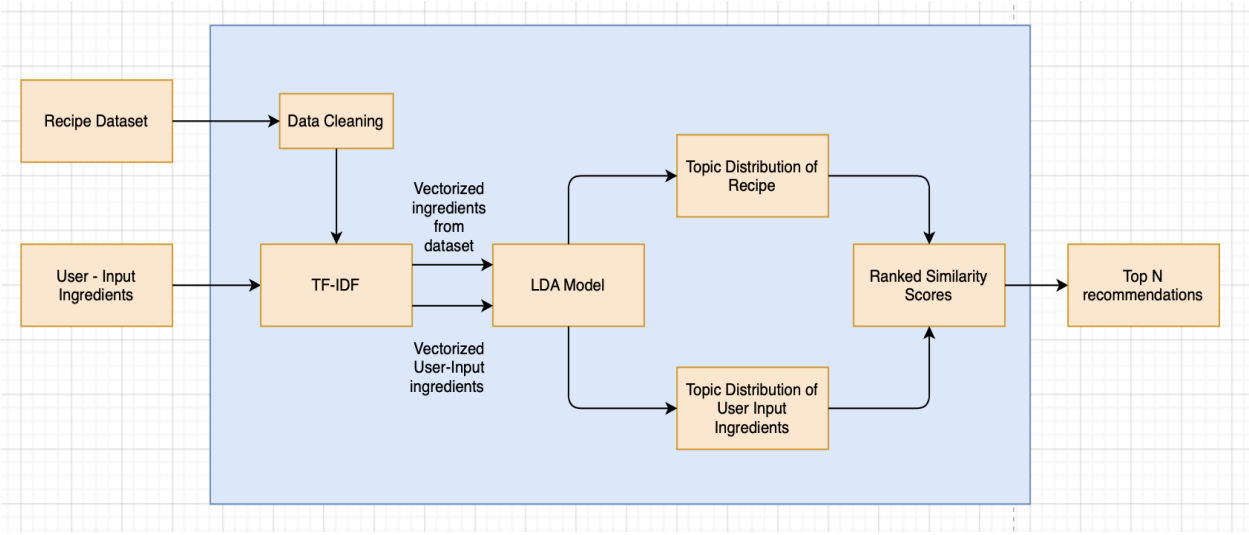


Fig 1: Component Diagram

## Software components

### Data Cleaning Pipeline

Before utilizing the Recipe Recommender, a user has the option of inputting their dataset into a pipeline that will clean and preprocess their dataset of recipes into a usable format

for the model. Due to the variability of potential datasets, this method will require some level of documentation to guide the user to provide a reasonable dataset on which to clean. This user would be able to input a CSV file path from which to read the data, a delimiting character, and the names of the columns that list a recipe's name, its ingredients, and its cooking instructions. A cleaned dataset will be saved to a CSV file path that can be specified by the user.

**Topic Modeling Algorithm**

The recommendation consists of two classes that leverage the capabilities of a topic modeling algorithm called Latent Dirichlet Allocation (LDA) in the **sklearn.decomposition**[2] package. This is an unsupervised algorithm that can group words into "topics," based on their frequency, structural context, and other latent semantic patterns across all recipes. LDA will transform the list of ingredients and produce a distribution of probabilities of a recipe being associated with each topic. This component has no output, but fits the LDA model with the input dataset for future reference in the recommender system.

**Recommender**

The second class will act as an interface for a user to query recipe recommendations based on a user-input list of ingredients. The user can input a list of ingredients into the recommender, which will use LDA to transform it and obtain a topic probability distribution. It then performs a cosine similarity search across the dataset to find and return the most relevant recipes according to similarity rank. Ideally, the top recommendations will primarily contain the ingredients listed by the user. The separation of topic modelling and recipe recommendation aims to emulate the general architecture heavily used in the **sklearn** library: decoupling the Estimator and Predictor of a model.

**LDA Visualization Tool**

The visualization will use the **pyLDAvis** library, and will provide a bit more interpretability to the recommendations provided to the user. The visualizations primarily show the various topics that it has generated, along with the most frequently seen words within each topic. Furthermore, it will have interactive features that will allow users to filter or drill down into the topic clusters.

# Interactions

Smooth interactions between the various software components as well as between the components and the user help ensure a seamless user experience. Our primary user is someone who wants to get serendipitous recommendations/ideas for recipes based on ingredients at hand. We also assume that the user knows basic python to interact with our library.

- The user would have to instantiate the class and specify a recipe dataset (in a certain format as stated in the documentation) along with a list of potential ingredients.
- The first interaction would be the preprocessing that would be done on the inputted dataset so that the data is preprocessed for the model which includes data cleaning, creating bigrams, as well as creating a TF-IDF vector.
- The output of the previous step is then passed to the LDA model which breaks down each recipe into weighted topics containing a set of ingredients.
- The model would then interact with the recommendation component by comparing the topics of each recipe in the dataset with the corresponding topics in ingredients that the user has specified.
- The top N recommendations would then be displayed to the user along with an interpretable visual of what caused the model to generate those recommendations.

# Preliminary plan

| Task | Start Date | End Date | Status |
|---|---|---|---|
| Create Requirement Document | 2/12/2021 | 2/15/2021 | Complete |
| Technology evaluation | 2/18/2021 | 2/21/2021 | Complete |
| Create design document | 2/24/2021 | 2/28/2021 | Complete |
| Create Unit Test | 2/25/2021 | 3/8/2021 | In Progress |
| Pre-process the dataset | 2/25/2021 | 3/8/2021 | In Progress |
| Build and test the model | 2/25/2021 | 3/8/2021 | In Progress |
| Perform integration testing | 3/9/2021 | 3/10/2021 | Not yet started |
| Package | 3/11/2021 | 3/13/2021 | Not yet started |
| Documentation | 3/14/2021 | 3/15/2021 | Not yet started |

# References

[1] Dataset - https://www.kaggle.com/elisaxxygao/foodrecsysv1

[2] Scikit Learn LDA - https://scikit-learn.org/stable/modules/generated/sklearn.decomposition.LatentDirichletAllocation.html