CS3263 – Foundations of Artificial Intelligence
Term Project Report

SmartFridge, an AI agent that recommends the best recipe for you

Lei Jianwen (A0281480E, e1155608@u.nus.edu)
Chen Yixun (A0281405L, e1155533@u.nus.edu)
San Muyun (A0271888J, e1121291@u.nus.edu)

November 7, 2025

1 Introduction

1.1 Motivation

Research has shown that households often waste food because expiration dates are overlooked and meal planning is not done systematically. The amount of avoidable food waste is equivalent to each household throwing away a 2.5kg bag of rice every week. And over half such waste can be reduced if meals are better planned. Recognizing this issue, our team aims to develop an AI-driven solution. An **AI-driven recipe suggestion system** can address this challenge by tracking food items, reasoning about their freshness, and proposing meal options that prioritize ingredients with shorter expiration dates.

1.2 Project Scope

Our solution contains the following features

- Keep track(Add/Delete) food items from the fridge
- Keep track(Add/Delete) recipes from a recipe book
- Recommend the most suitable recipe from the recipe book

1.3 Techniques Used

The recommendation is simulated by a decision-making process to decide which recipe to recommend. A decision tree is constructed to solve this problem. The parameters of the decision tree can be categorized into two types. Bayesian networks are implemented to infer the probability of successfully preparing the recipe and probabilities of expiration of food items. The utility function for each recipe is calculated using an unsupervised machine learning model.

1.4 Terminology & Notations

For clarity, the main symbols and assumptions used in this report is defined below:

- $\mathcal{F} = \{f_1, f_2, \ldots, f_n\}$ denotes the set of food items available and currently stored in the fridge.
- Each food item f_i is associated with food names $f\_name_{f_i}$ days in the fridge $days_{f_i}$, food type fty_{f_i}, storage type sty_{f_i}.
- $\forall fty_{f_i}, fty_{f_i} \in FT, FT = \{\text{CANNED}, \text{VEG\_OR\_FRUIT}, \text{MEAT}\}$. To simplify the problems, we only focus on three types of food: canned food, vegetable and fruits, and meat.

- $\forall sty_{f_i}, sty_{f_i} \in ST, ST = \{\text{REFRIGERATE}, \text{FROZEN}, \text{NORMAL\_TEMP}\}$. Similarly, we limit the scope of the discussion to only three types of storage condition required by each food: refrigerated, frozen and normal temperature.
- $\forall f\_name_{f_i}, f\_name_{f_i} \in \mathcal{N}$, $\mathcal{N}$ is the set of food names available in the fridge.
- $\mathcal{M}$ is the set of recipes r_i that will be available for users. A fixed set of recipes has been redefined with its relevant attributes.
- Each recipe r_i has an unique attribute of recipe name $rname_{r_i}$.
- Each recipe r_i has a requirements in form of a set of names of the required food items $\mathcal{R}_{r_i} = \{f\_name_F | f\_name_F \in \mathcal{N}\}$.
- Each recipe r_i has a corresponding list of tags $\mathcal{C}_{r_i}$, which will be then used in determining the desirability of a recipe.
- We assume the quantity of food available in the fridge is sufficient for any recipe that required this food as part of the ingredient requirements.

2 Decision Making

2.1 Decision Tree construction

The essence of the problem can be viewed as a decision-making task — specifically, choosing a recipe from a recipe book. To model this process systematically, a decision tree is constructed to represent the possible choices and outcomes. In this case, some problems would be:

- What are the possible outcomes after recommending a recipe?
- How to find the probability of these outcomes?
- How to find the utility of these outcomes?

Our design is simple. The recipe will either be feasible or not, given the foods in the fridge. The feasibility of the recipe is assessed using a Bayesian network, and the utility of a feasible recipe is assessed using a supervised machine learning algorithm. The utility of an infeasible recipe is set to 0. A diagram illustration is given in appendix as Figure 1.

2.2 Decision Making Process

As a rational agent, our bot aims to choose the recipe that maximizes the expected utility. Here is the meaning of the notations we used in our formula:

- Rec: The final recommendation

- R: The recipe book (set of all recipes)
- r: A specific recipe
- s: A binary value, representing feasible or not
- P: The feasibility function
- U: The utility function

With these notations, our algorithm can be represented as:

$$\text{Rec} = \arg\max_{r \in R} \mathbb{E}[U \mid r] = \arg\max_{r \in R} \sum_{s} P(s \mid r)\, U(s, r)$$

Since U(fail, r) = 0 by design, the formula can be simplified to:

$$\text{Rec} = \arg\max_{r \in R} P(feasible \mid r)\, U(feasible, r)$$

3 Bayesian Network Inferencing

The **feasibility** of preparing recipe r_i depends on state of foods available in the **fridge** that is in the requirement set $\mathcal{R}_{r_i}$. We assume that the state of each food is dependent on three characteristics of food f_i: **food type** fty_{f_i}, **storage type** sty_{f_i}, and **days in the fridge** $days_{f_i}$.

Due to the partial observability of the true state of food items in the fridge, **Bayesian Network** is used for handling the uncertainty and infer the probability of the successful preparation of a specific recipe. The **pgmpy** (Ankan and Textor [2024]) library is used to construct **Bayesian Network** and perform the inference task.

3.1 BN structure

The visual representation of the overall structure can be seen from the picture Figure Appendix 1

The feasibility of the recipe depends on feasibility of the food items, in other words, whether the food has expired. To model this relationship, we first construct a sub-Bayesian Network for each of the food item. In the sub-network, the node $expired_{f_i}$, which is a random variable denoting the event whether the food is expired, has three parents nodes which representing the random variables for food type fty_{f_i}, storage type sty_{f_i}, and days in the fridge $days_{f_i}$ respectively. The parent nodes representing the observed characteristics of the food.

The probability of successfully preparing the recipe is denoting by random variable $feasibility_{r_i}$. We assume that the feasibility of each food item is conditionally independent of each other. As such, the node $feasibility_{r_i}$ will have multiple parents that denoting the random variable of $expired_{f_i}$ for each f_i in recipe's requirements respectively.

Hence, we can derive following formulae for the probability of successfully preparing a recipe, i.e. feasibility of the recipe (Assume all food items required by the recipe can be found in the fridge):

$$
\begin{aligned}
&P(feasibility_{r_i}|\mathcal{F}) \\
&= P(feasibility_{r_i}|expired_{f_1}, \dots expired_{f_n}) \prod_{k=1}^{n} P(expired_{f_i}|f_i) \\
&= P(feasibility_{r_i}|expired_{f_1}, \dots expired_{f_n}) \prod_{k=1}^{n} P(expired_{f_i}|fty_{f_i}, sty_{f_i}, days_{f_i})
\end{aligned}
$$

$\mathcal{F}$ denotes the current state of the fridge with all the food available
$f_1 \dots f_n \in \mathcal{F}$

3.2 Inference Framework

Here is the process of building the Bayesian Network and perform the inference process with the help of pgmpy.

- Extract food items from the fridge that is required by a recipe
- Build BN_{f_i} for each food extracted
- Connect BN_{f_i} to node $feasibility_{r_i}$
- Extract the evidence from each food items
- Run variable elimination inference algorithm

The code that generates this part is presented in **??**

3.3 Maximum Likelihood Estimate

We estimate the Conditional Probability Table of each node using a maximum likelihood estimate. We would estimate $P(a) = \frac{\#(a)}{N}$ where $\#(a)$ denotes the count of occurrences of a in the dataset and N is the number of samples. We can estimate conditional probability in a similar way: $P(a|b) = \frac{\#(a \wedge b)}{\#(b)}$. The complete set of data is in Appendix 2

4 Utility

We calculate the utility of the recipe in order to make recommendations. Since our aim is to reduce the number of near-expiry food left in the fridge, the overall utility of a recipe consists of two parts, the utility of the recipe which depends the recipe itself, and an expiry score which depends on the available ingredients of this recipe.

$$Overall\ utility = utility\ of\ recipe + expiry\ score$$

4.1 Utility of recipe

The utility of each recipe is modeled as a function of multiple attributes, including the estimated cooking time, ease of preparation, nutritional profile, dietary classification (e.g., vegetarian), and course type (e.g., dessert). We apply the k-means clustering algorithm to group recipes with similar attribute profiles and assign a utility value to each cluster. Model training was conducted on an online Kaggle dataset (Wei [2023]).

4.2 Learning for utility

The original dataset (Appendix 2) contains a large number of headers, many of which are redundant. For model development, we therefore restrict the feature set to the first 25 attributes (see Appendix 3 for the full list of 25 attributes). Because we aim to cluster recipes by similarity, we apply feature scaling so that all variables contribute comparably to the distance-based objective used by k-means clustering. Most attributes are binary indicators which are 0 or 1, whereas "step_cnt" and "ingredient_cnt" are integer-valued and can be > 1. To prevent these count variables from disproportionately influencing Euclidean distances between vectorized recipes, we normalize them to the interval [0,1] via Min–Max scaling:

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

In order to have a large diversity for the recipe utility, we set the number of clusters to $k = 2000$. The k-means algorithm is trained using $scikit - learn$ implementation. The trained model is saved into a ".joblib" file for future predictions.

After clustering, each cluster is mapped to a utility value. For new recipes, we use the trained k-means algorithm to group it into a cluster, and map it to the corresponding utility value.

4.3 Expiry Score

The Expiry score is taken from the expiry probability model, which is the same node *expired* in the Bayesian Network mentioned in section 3.1 including the three parent nodes representing random variables for food type *fty*, storage type *sty* and days in the fridge *days*. *expired* is a random variable with three outcomes, *fresh*, *near_expiry* and *expired*. Since we want to consider the number of food close to expiry and how close they are to expiry, we only consider the probability of near_expiry given the parent nodes as evidence. To be more precise, we are only looking at:

$$\mathrm{P}(near\_expiry | fty, sty, days)$$

Since a recipe has multiple ingredients and we prefer to use more ingredients which will be expired soon, we calculate the average probability of near_expiry probability of all ingredients in the fridge, denoted by N_{stored}, multiplied by the number of ingredients in fridge N_{stored}. Hence, we have the following:

$$Expiry\ Score = \sum_i P(near\_expiry_i | fty_i, sty_i, days_i), i \in fridge.foods \cap ingredients$$

Since we are only checking the expiry score of food items stored in the current fridge, we can always pass in the evidence from the food attribute as we query for the conditional probability.

4.4 Expected Utility

Since some ingredient of the recipe could be expired, hence not feasible, the expected utility is taken as:

$$P(feasible|recipe) * U(recipe)$$

where $U(recipe)$ is the utility of the recipe assuming that it is feasible.

5 Future Improvement

5.1 Connection to Internet of Things

Currently, we estimate the expiration condition of a food item indirectly, using its type, storage condition, and storage duration. In real life, people use the appearance and smell to more directly estimate the expiration condition of a food item. We should also adopt this method and add the appearance and smell factor to our expiration probability model.

To train the expiration Bayesian Net work, data about appearance and smell of food are required. Seeing this need, we plan to integrate the internet of things into our project. The following component will be implemented:

- Camera: To collect appearance data of food items
- Electronic Nose: To detect smell/chemical component in the fridge
- Data Transmitter: Transmit data from the fridge to the computer

With the help of the Internet of Things, we can obtain expiration data with appearance and smell. We can retrain the updated BN with these data to make more realistic prediction.

5.2 Reinforcement Learning

Currently, our agent is fixed after deployment. In other words, it will recommend the same recipe under the same conditions, even if the user is not satisfied with the result. Seeing this shortage, we plan to integrate model-based reinforcement learning into our agent. Currently, our agent does not have an "immediate reward" for recommending a recipe. Or we can say that the "immediate reward" is always 0 for all recipes.

We plan to introduce a learnable function R(r), representing the reward for recommending the recipe r. Hence, the mathematical representation for the final choice becomes:

$$\text{Rec} = \arg\max_{r \in R} \mathbb{E}[U \mid r] + R(r) = \arg\max_{r \in R} \sum_{s} P(s \mid r)\, U(s, r) + R(r)$$

This reward function, R, can be learned in a method similar to how the reward function is learned in model-based reinforcement learning. We will ask the user to rate the recommended dish after the recommendation from one to five. This user feedback will be used as the observed environmental reward. When feedback is given about recipe r, the function R is updated as follows:

$$\text{R'(r)} = (1 - a)R(r) + a(S(f))$$

Where the notation has the following meaning:

- R' The updated reward function
- f the user input feedback score
- S the function that normalizes the feedback score
- a the learning rate

Here, the learning rate for recipe r will decrease with the increasing number of feedback provided by for r from the user. This ensures the stability of our agent in the long run.

With this new integration of reinforcement learning, our agent will be able to adopt the eating habits of each user and better scope its job to minimize food waste.

6 Responsible AI

Our product (RecipeAI) recommends dishes based on the estimated freshness and availability of ingredients. The goal is to reduce food waste by prioritizing dishes that use items likely to expire soon. This will inevitably lead to some responsible-AI considerations.

6.1 Food to be used should not be expired to minimize harm

While we prioritize dishes that consume ingredients nearing expiry, we must minimize the chance of recommending dishes that include items that are already expired or unsafe. We

use a Bayesian network to estimate the probability that an ingredient is spoiled given its type, storage condition, and time in storage.

However, even with such a model in place, it is still difficult to accurately predict the expiration of each food item. Different food items, even of the same type of food, can expire over a range of days. Currently, more advanced models use neural networks and are trained using large datasets of all types, including temperature, moisture level, microbial growth and more (Shehzad [2025]) to predict the spoilage of food. Currently, for our model, we do not have such large datasets, and we can only estimate whether the food has expired or not based on simple user inputs. We also do not have sensors to monitor the condition of the food items in real time. This will inevitably lead to inaccuracy in our predictions. To avoid safety issues, we make it clear to users that this app is only meant for recommendation. Users still need to check whether the food items in storage are in good condition for consumption.

6.2 Transparency and Explainability

For every recommendation, the expected utility is calculated using feasibility and a utility score of the recipe. When a user queries for a recommendation, RecipeAI will return the feasibility, utility, as well as expected utility for each recipe, before displaying the final recommendation at the end. This allows user to have more information on how the recommendation is made, without going too deep into exactly how each of the values is computed.

6.3 Unintended side effect and reward hacking

Unintended side effects can arise when the system optimizes the stated objectives and more likely to recommend food that is closer to expiry. This can potentially conflict with user's welfare or safety. For example, aggressive waste reduction may over-prioritize soon-to-expire items into every meal, yielding monotonous menus, poor nutrition balance, or encouraging users to keep borderline-fresh items.

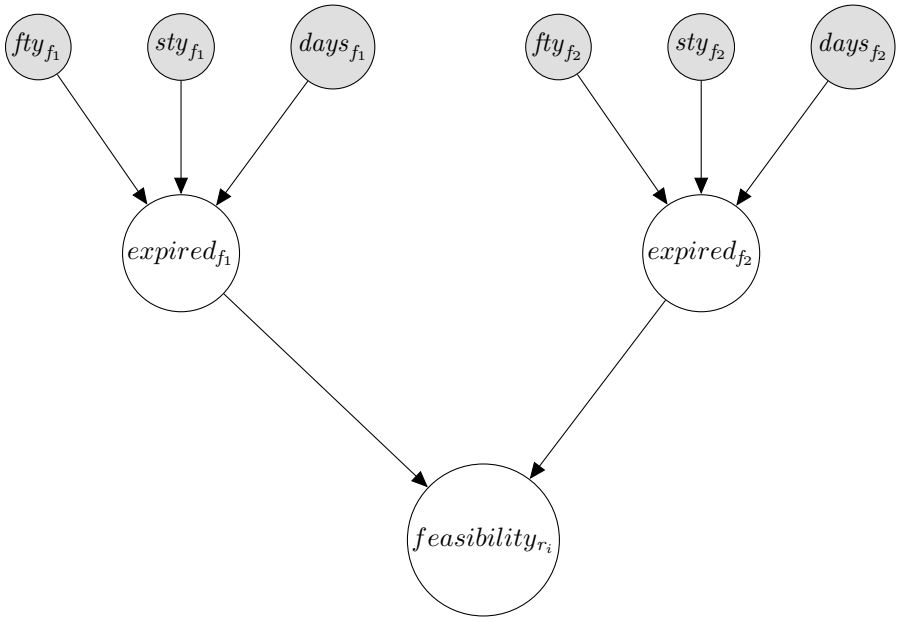
To mitigate this issue, we can impose a threshold when estimating feasibility of recipe. If a certain percentage of ingredients exceeds a certain threshold in expiry probability, we reduce the final feasibility, hence reducing the expected utility of the dish, reducing the likelihood of it being recommended. This helps to ensure that users have a balanced and healthy diet, and prevent itself from encouraging users to leave food in storage until it is close to expiry.

7 Conclusion

This prototype of application has utilized various skills and knowledge from CS3263. Reasoning techniques such as Bayesian Networks has been applied to represent uncertain knowledge

and draw new conclusions and evidence from existing knowledge. We then utilize a decision system for the recommendation of recipes. Lastly, machine learning will be employed to improve the performance of the system. In the future, we plan to apply reinforcement learning and integration with IoT to improve the capacity of the agent to make more favorable decisions and better sense the environment.

Appendix 1 Diagrams



Appendix 2 Data

- attributes of recipe vector: 'preparation', 'time-to-make', 'course', 'main-ingredient', 'dietary', 'easy', 'occasion', 'cuisine', 'low-in-something', '60-minutes-or-less', 'main-dish', 'equipment', '30-minutes-or-less', 'number-of-servings', 'meat', '4-hours-or-less', 'desserts', 'vegetables', '3-steps-or-less', 'taste-mood', 'north-american', 'low-sodium', 'low-carb', 'healthy', '15-minutes-or-less', 'step_cnt', 'ingredient_cnt'
- Training data for Utility function: Kaggle — Food.com Recipes with Ingredients and Tags
- Training data for Bayesian Network: GitHub — food_condition.csv

Appendix 3 Algorithm

Appendix 3.1 Algorithms for inference machine

```
from Models.FoodAndRecipe import Fridge, Recipe
from Generators.EvidenceBuilder import EvidenceBuilder
from Generators.BNGenerator import BNGenerator
from pgmpy.inference import VariableElimination
from pgmpy.factors.discrete import DiscreteFactor
```

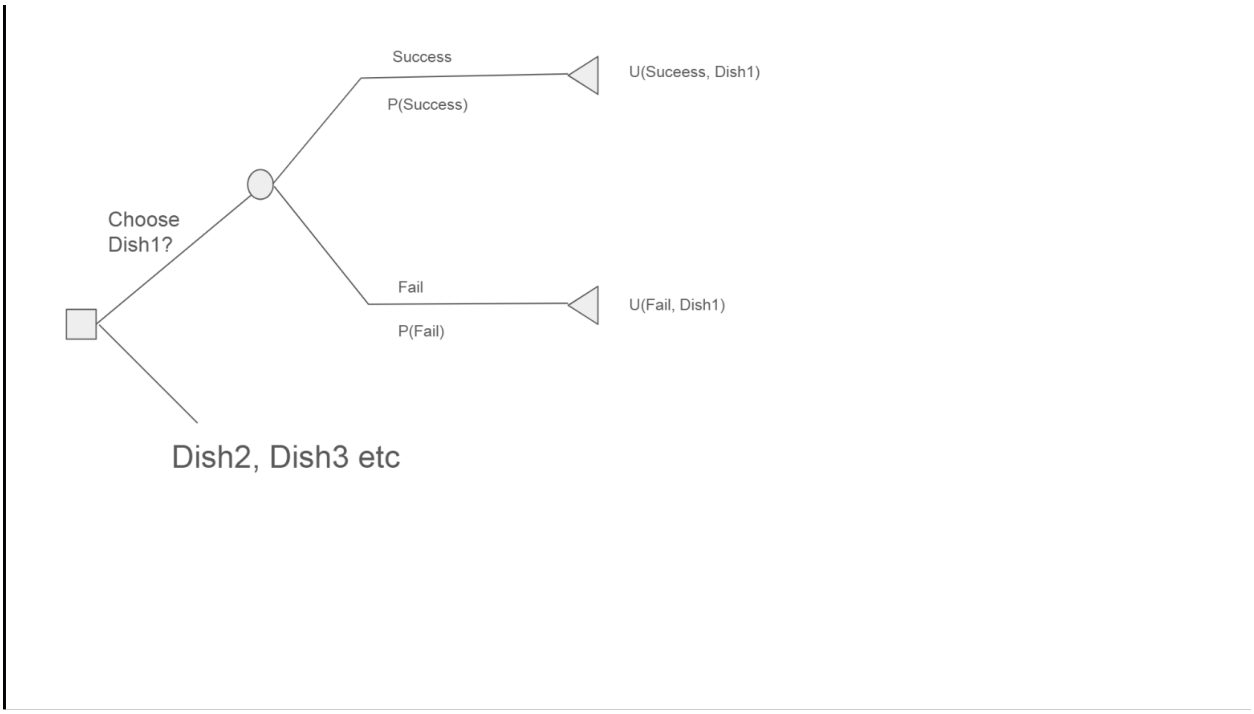


Figure 1: Decision Network structure

```
class InferenceMachine:
    @staticmethod
    def infer(fridge: Fridge, recipe: Recipe):
        # if cannot cook, just return a
        # DiscreteFactor of success rate = 0
        if not fridge.can_cook(recipe):
            return DiscreteFactor(
                variables=['Feasibility'],
                cardinality=[2],
                values=[1, 0])

        food_xs = [
            fridge.extract_food(r)
            for r in recipe.requirements]

        SampleBN = (
            BNGenerator().build_bn(recipe=recipe)
        )
        infer = VariableElimination(SampleBN)

        evidence = {}
```

```
for f in food_xs:
    evidence.update(
        EvidenceBuilder.build_food(f)
    )

return infer.query(
    ["Feasibility"],
    evidence=evidence
)
```

Appendix 4 Acknowledgment of the use of Generative AI

We utilized Generative AI tools primarily to understand APIs of libraries used such as **pgmpy** and **scikit-learn**, and learn Latex syntax and packages to format and generate the report. The main design and implementation of different components of the application were done independently. AI tools were only used as reference and learning aids to become familiar with tools and syntax

Bibliography

Ankur Ankan and Johannes Textor. pgmpy: A python toolkit for bayesian networks. *Journal of Machine Learning Research*, 25(265):1–8, 2024. URL http://jmlr.org/papers/v25/23-0487.html.

Khuram Shehzad. Predictive ai models for food spoilage and shelf-life estimation. *Global Trends in Science and Technology*, 2025.

Alexander Wei. Food.com recipes with ingredients and tags. Technical report, Kaggle.com, 2023. URL https://www.kaggle.com/datasets/realalexanderwei/food-com-recipes-with-ingredients-and-tags/data.