



MEALMATE

R Shiny App Report

TEAM: BAIM Alcoholics Anonymous

Xueying (Alicia) Yang

Alex Hartman

Uma Aiyer

MealMate

Alicia Yang, Uma Aiyer, Alexander Hartman

Purdue University, Department of Management, 403 W. State Street, West Lafayette, IN 47907
yang551@purdue.edu; uaiyer@purdue.edu; hartma82@purdue.edu

Abstract

[Epicurious](#), a popular food website, currently provides more than 330,000 recipes and is a leading source of information for novice cooks. As consultants for Epicurious we are tasked with suggesting additional features that may add value for their users.

Preliminary information from test users and secondary info sources suggests that many young adults do not explore cooking because they are not sure how to go about finding recipes (based on difficulty levels, ingredients or time). The website's current features do not support such granular suggestions, and we do strongly believe that this would greatly increase Epicurious' appeal in the target segment.

As a primary suggestion, we have implemented a functionality that will help people filter meals based on specific inputs (proteins and vegetables) seeking to optimize one major nutrient type (in this case, proteins, since young adults prefer higher protein content in their foods). We gathered secondary data relating to suggested content on various nutrients in a healthy diet and built an optimization model to provide the user with maximum protein. We are in the early stages of development and are testing the current R Shiny Application to see its popularity among target users and incorporating any feedback.

Keywords: Healthy Lifestyle, Calories, Fat, Sodium, Food, Rush Meals, MealMate, Cooking, Protein, Ingredients, Health Conscious, ShinyApp, R-Studio

Business Problem:

Epicurious, a popular food website, wants to expand its reach to a wider target audience and is considering additional functionalities that may help them achieve this. We are working with Epicurious to create 2 new functionalities on their website that will provide recipes based on input criterion from website visitors:

1. Provide 2 recipes to users that would take into account user's protein and vegetable options and is optimized for protein
2. Provide 2 recipes to users that can be prepared in under 22 minutes that would take into account user's protein options and is optimized for protein

Utility for Users:

1. Easy to use (checkboxes and sliders) and automatically optimizing for maximum protein
2. Considers both protein options, ensuring reduced food wastage
3. Provides an option for quick-to-cook recipes with relevant inputs (current site functionality does not provide such a feature)

Benefits to Epicurious:

1. Engage with new users and provide "value" to their website visit

2. Cleaning their dataset to identify irregularities which may have crept it
3. Application now provides functionality to visitors who are constrained on time
4. Potential for increasing ad revenues indirectly over time (from increased user interest)

Analytics Problem

In looking at the data, we decided that we would focus on maximizing protein per meal, which is a nutrient group that users typically care a lot about, and allow for functionalities to enable users to input the total amount of meals they would like to eat. The model's key drivers would be the ingredients, the total amount of meals, and the total calorie per meal; any change in input would alter the recipes being presented to the user. For example, picking a leaner meat like turkey will display healthier recipes than a fattier meat such as beef. The output is also randomized (within the overall constraints); a user choosing the same 2 proteins over 2 instances may not give the same output, since we want the user to have a more diverse meal experience as well.

Key metrics in our application, as mentioned above, are related to finding the highest amount of protein per meal based on the limitations set during user input. Additionally, as users will be given two recipes we want each recipe to be optimized regarding serving size per meal.

Data

We have sourced data for this application from Kaggle, where the original dataset was built off Epicurious' website. Data includes rows indicating different recipes and columns, which are key tags attached to the recipe. The columns contain binary variables (0 or 1) illustrating if they are present or not in the recipe and provides information for rating (on website), calories, proteins, fat, and sodium content. The dataset is slightly dated and, while large, has a lot of blanks.

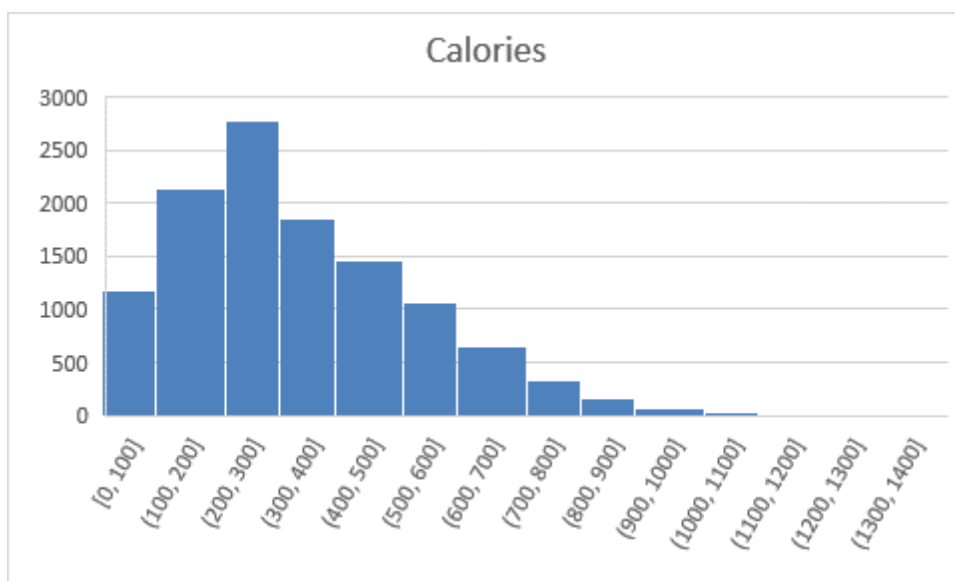
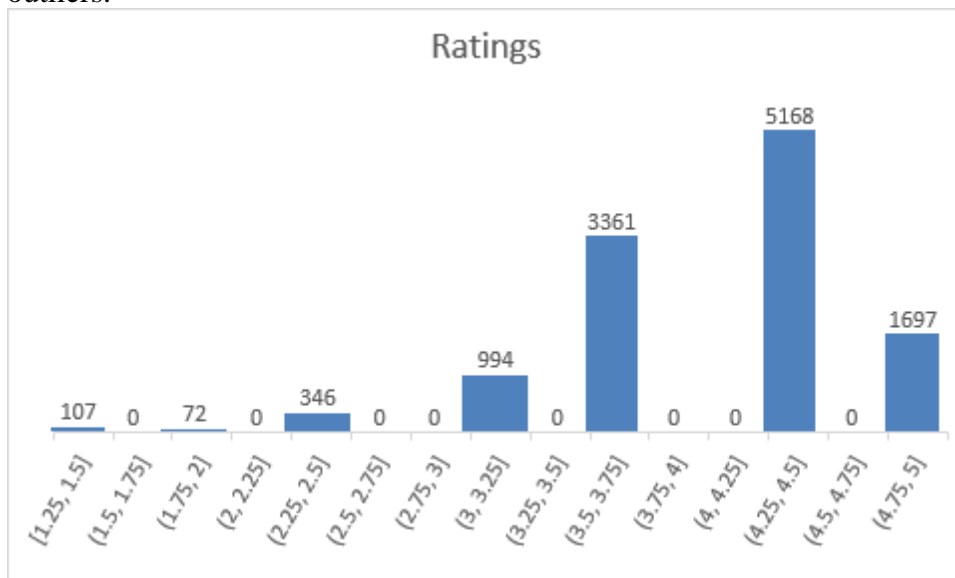
To create a functional app, we first cleaned the dataset through the removal of unnecessary columns and through the removal of any data points that were blank.

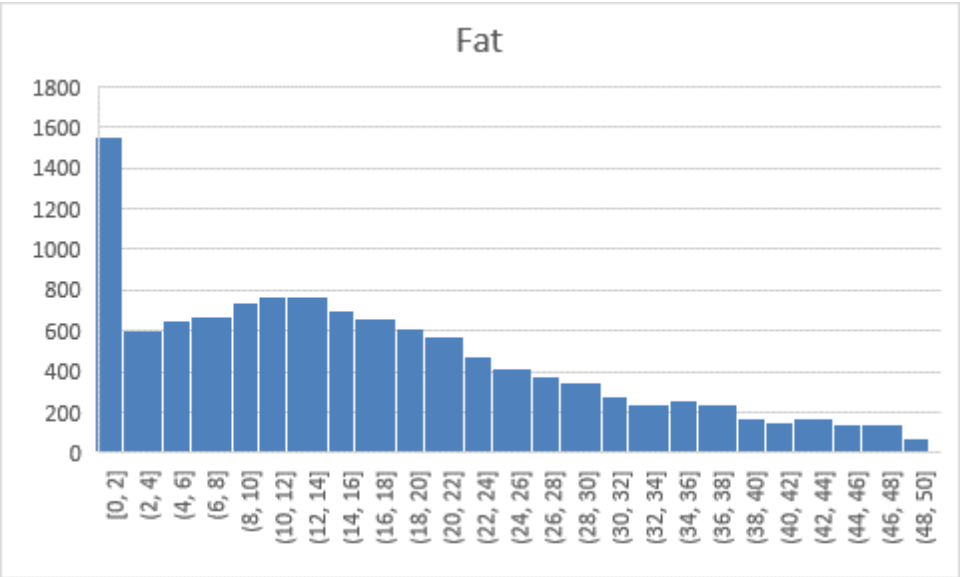
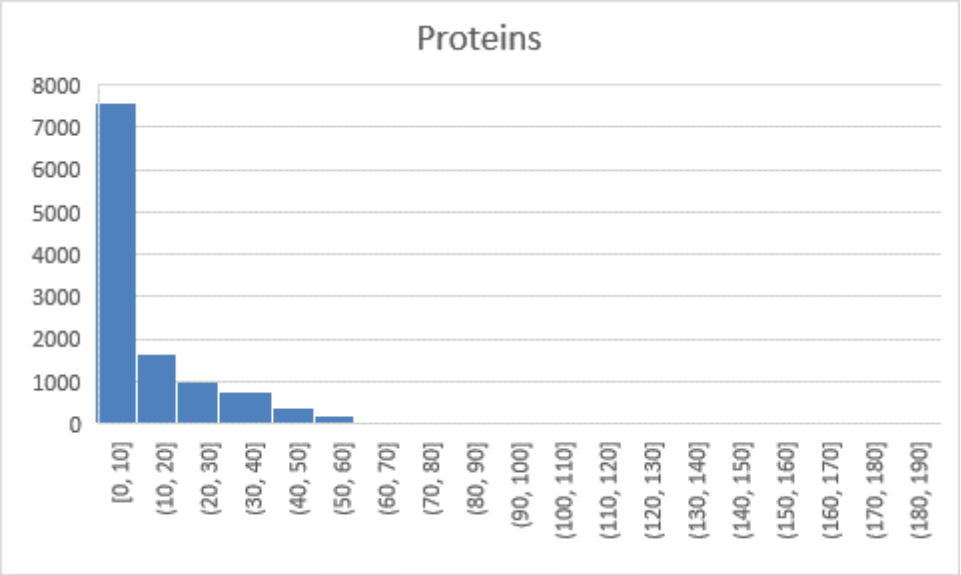
Some of the issues we found:

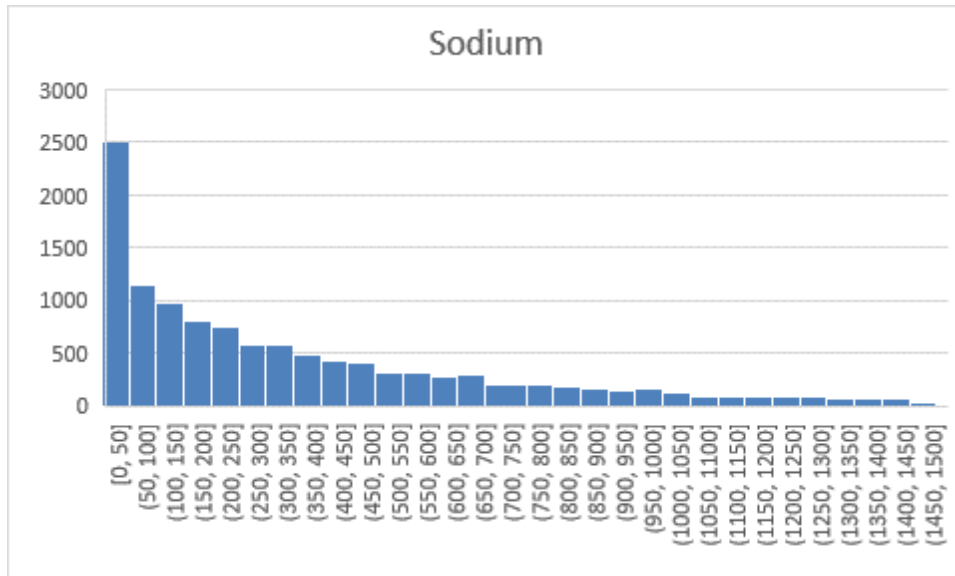
1. Some data points were missing info relating to calories, fat, protein or sodium, and needed to be excluded
2. Several observations had 0 rating (i.e. extremely unpopular among site users) and needed to be excluded to present optimal recipes for users
3. Ratings are directly related to users and can alter future models
4. Some columns headings were tags that weren't ingredients and may not be needed for our analysis (e.g. Anniversary rather than an ingredient)
5. Unicode text for certain recipes, such as **SautÃ©ed** instead of Saut  ed
6. Duplicate recipes on the list
7. Some recipes included more than one protein – which meant that we could not necessarily subset based on just one protein

Findings:

1. We have a total of 15864/20052 rows of data and 405/680 columns after removing blanks. Further filtering out the recipes with “0” rating, we have a dataset with 14,568 recipes.
2. Some recipes have abnormally high calories, protein, fat or sodium content – they may be potential issues with the data creation. So, we had to create a filter to ensure that these recipes do not unduly influence recipe recommendations [1500 calories, 50g of fat and 1500mg of sodium per recipe was found to be within a healthy range]. After this filtering, we have 11,745 records.
3. Summary statistics from data: Our data is significantly right skewed after filtering for the outliers.







Methodology Selection

Based on the preliminary findings from our dataset and the requirement to create a model that can present optimized recipes, our methodology needs to account for user inputs (as a filter on our overall data) and then pass any outputs from those recommendations into an optimization model.

On the “filtering model”, we may need to limit our user inputs to create the filters, otherwise we end up with null recommendations (i.e. we know that are limitations in our dataset, so we shouldn’t stress our model any more than required). Some testing may be required to create a stress-tested model. It may be worthwhile to interact with potential users to get a sense of important ingredients and/or ingredients that they would most often search for and build that into the app in the first run (allowing to scaling up later). Given that calories, fat and sodium had certain healthy limits, we could explore building in that constrain early on in the filtering methodology, or use it during the optimization process.

A Simplex Linear Programming Model may be an adequate model to meet our user requirements for the optimization model, as long as we are able to set intelligible bounds on the variables.

R provides an excellent platform to create and test such a model for us - we can use in-built functionality and conditional statements to create a “shortlist” of recipes and can then use R’s capability to create optimization models (lpSolve and lpSolverAPI package) that can provide us the output we need. R’s capability to handle medium to large datasets (such as our current dataset) is a positive as well.

Model Building

Our initial tests to create the “shortlist” dataset incorporated only basic bounds on parameters: ensuring that they lined up with user input. However, there were several cases in which we were unable to create a legible shortlist (empty datasets were a frequent issue). Taking cues from findings from our EDA, we incorporated bounds on our user inputs, allowing a limited number of proteins and vegetables (in our case 6 proteins and 6 vegetables). Additionally, we needed to use an “or” approach on the vegetable constraint - very few recipes actually contained all the vegetables in many of our test runs, and we needed to factor that in ahead of time. Our initial assessment that we needed to interact with test users to select that limited set of ingredients was very valuable, and gave us a list of ingredients on which our “filtering” model could be adequately applied.

On our optimization model, we discovered that our initial decision to apply stringent “healthy filters” on calories, sodium and fat were unnecessary; our model had the capability to optimize for these parameters and we didn’t need to create very restrictive filters. Therefore, we created less stringent filters for calories, fat and sodium based on the upper bounds of healthy limits (1500 cals, 50 g of fat and 1500 mg of sodium) and then allowed our optimization model to work on them further (considering user inputs). After multiple beta tests, we found no reason to suggest that these model constraints would not work as desired. We needed to create intelligible bounds i.e. ensure that the minimum number of servings per recipe were whole numbers ≥ 1 (to ensure that users didn’t need complex math to decipher a recipe!).

Since our goal is to maximize protein, the initial model ranked the filtered “shortlist” on protein and returned the top recipe from the list as output for each protein. However, after some beta runs, we noticed that this would result in the same output if the same input is selected by a user. To enhance the user experience and provide recipe variety, especially for return users, we changed the final selection model to return a random sample. Tests provided evidence that the previous filtering is sufficient to ensure the quality of these recipes.

After building up both our filtering and optimization models, we carried out multiple beta runs to identify potential logical loopholes and supply user-friendly error messages when required. We have built the model (and the related code) to support additional ingredients and/or nutrients (such as carbohydrates or fibre) as well; however, it would need to scaled up with additional testing to catch any logical loopholes.

Findings:

This model would need additional functionality such as limiting carbohydrates / sugars...etc. for users to obtain the full functionality of this application.

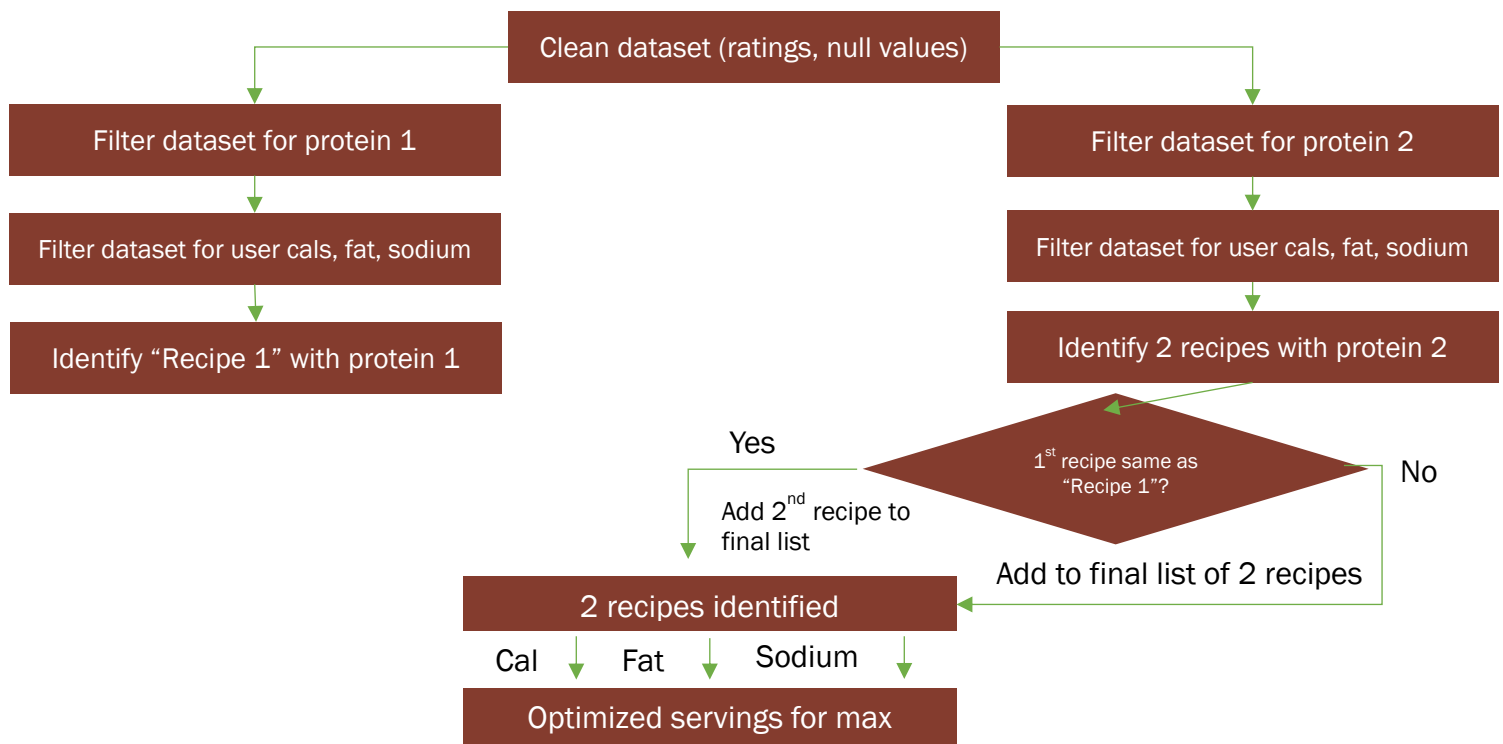
Since this model is an optimization model it’s ability is limited to the dataset and does not have any forecasting capabilities. Furthermore, since our application focuses on providing users with recipes, regression and clustering cannot be used for the output.

Functionality

The DSS can take in user inputs for ingredients and create a meal plan that would optimize the user's protein intake (maximized), while at the same time ensuring some benchmark on calories, fat and sodium. In keeping with the most popular proteins and vegetables in student households, we tailored our input list (so that it isn't overwhelming for users) and present only minimal output that is most relevant to our users.

We used the R shiny (to create the app), shinythemes (to prettify the app interface), markdown (to provide added functionality with layouts) and DT (to create html-supportive tables) packages.

A flowchart for our app functionality is as follows:



Additional functionality to get a link to view the recipe is also provided to user.

We would like to include additional functionality

- to scrape Epicurious' website to get additional data about recipes
- embed the relevant recipe links within the shiny app (based off of the web-scraping perhaps)
- to include more ingredients (currently limited to 12, we would like to classify more ingredients as proteins or veggies, so we can increase the user input over time). We would, however, restrict that to an upper limit of say 20 – the user shouldn't be overwhelmed with input options.

GUI Design and Quality:

- Initial Page:

Meal Mate

Protein Optimizer

Short on Time?

How many meals do you want this for?

3

Calories Limit per Meal

0 750 1,500

Select 2 Proteins

☐ beef

☐ chicken

☐ shrimp

☐ turkey

☐ salmon

☐ tofu

Select 3 Vegetables

☐ potato

☐ tomato


☐ onion

☐ carrot

☐ broccoli

☐ bell,pepper

GO



According to the 2010 FDA Dietary Guidelines:
The recommended calories for a 21-25 year-old moderately active adult is 1400 calories per lunch for Male and 1100 calories per lunch for female.
- The suggested fat intake is around 35% of total calories
- The Protein intake should be in the range of 10-35% of total calories
- The healthy range of sodium intake is 1100-1400 milligram per lunch.

- Protein Optimizer Tab functionality:

- Output:

Meal Mate

Protein Optimizer

Short on Time?

How many meals do you want this for?

3

Calories Limit per Meal

0 750 1,500

Select 2 Proteins

☒ beef

☒ chicken

☐ shrimp

☐ turkey

☐ salmon

☐ tofu

Select 3 Vegetables

☐ potato

☒ tomato


☒ onion

☒ carrot

☐ broccoli

☐ bell,pepper

GO



According to the 2010 FDA Dietary Guidelines:
The recommended calories for a 21-25 year-old moderately active adult is 1400 calories per lunch for Male and 1100 calories per lunch for female.
- The suggested fat intake is around 35% of total calories
- The Protein intake should be in the range of 10-35% of total calories
- The healthy range of sodium intake is 1100-1400 milligram per lunch.

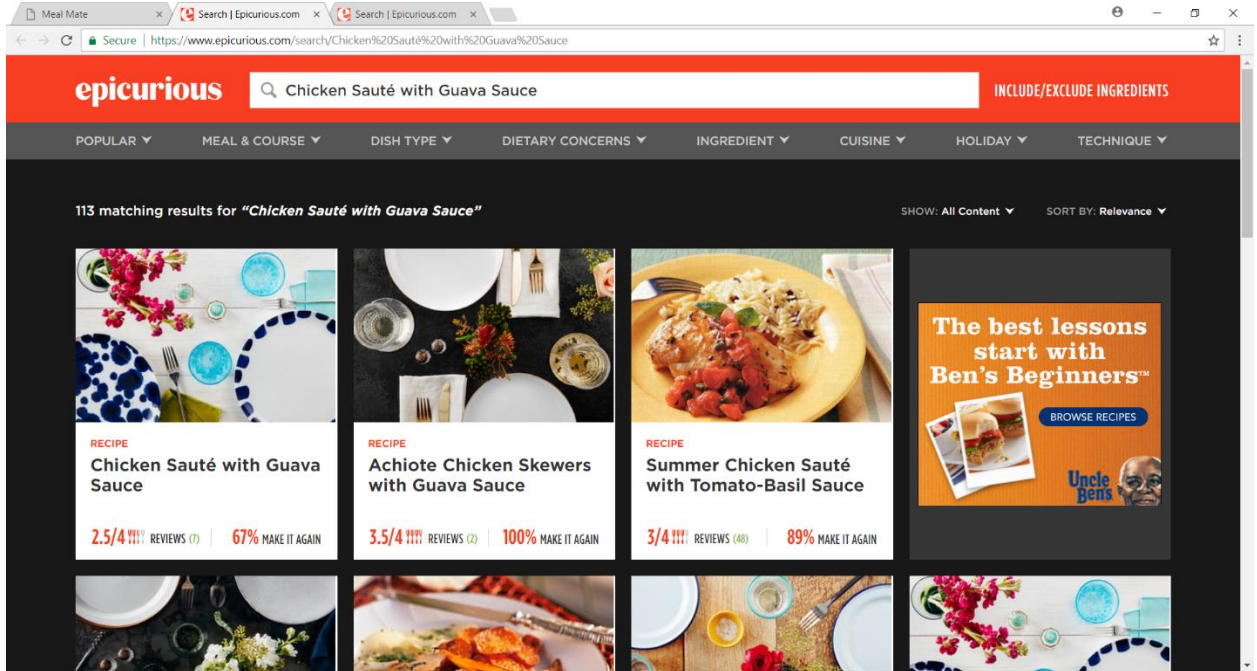
Search:

	title	rating	calories	protein	fat	sodium	serving	links
15749	Open-Face Roquefort Burgers with Grilled Onions	3.75	385	32	25	774	1	Recipe!
9362	Chicken Sauté with Guava Sauce	3.125	491	56	16	116	4	Recipe!

Showing 1 to 2 of 2 entries

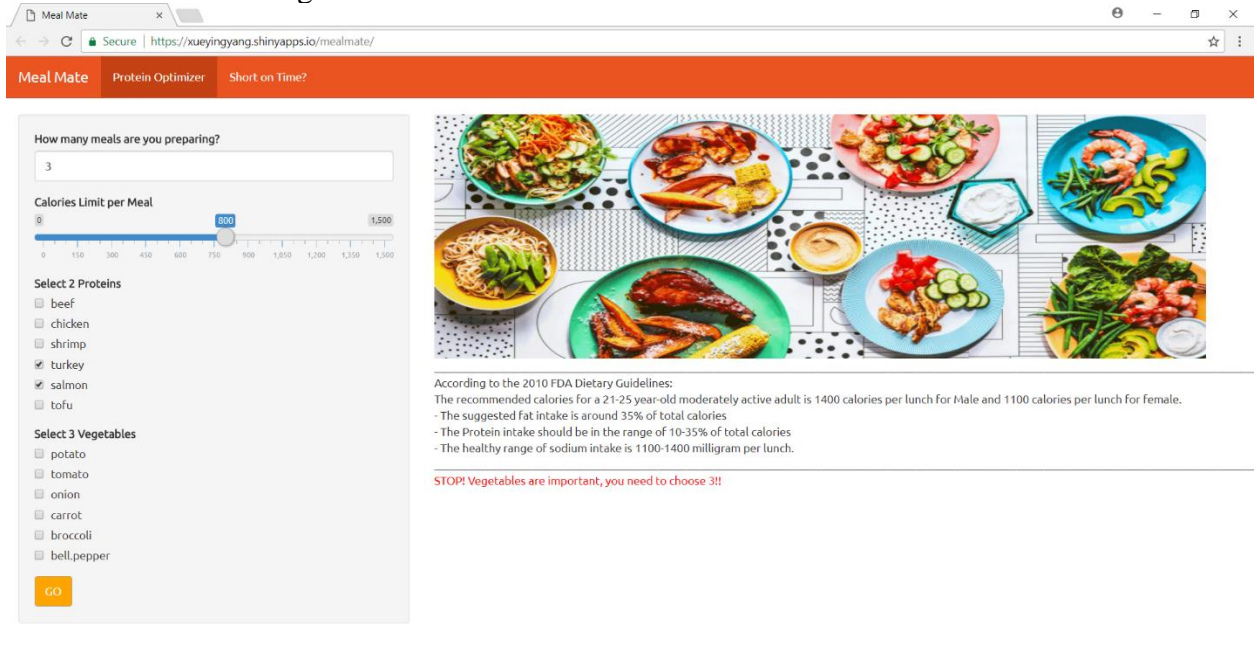
Previous 1 Next

2. Link to corresponding recipe:



3. Identify and guide user to solve problem with selections:

When there is a missing item:



When the choice of calories is too little:

Meal Mate

Protein Optimizer

Short on Time?

How many meals are you preparing?

3

Calories Limit per Meal

0

157

1,500

Select 2 Proteins

☐ beef

☐ chicken

☐ shrimp

☒ turkey

☒ salmon

☐ tofu

Select 3 Vegetables

☒ potato

☒ tomato


☒ onion

☐ carrot

☐ broccoli

☐ bell,pepper

GO



According to the 2010 FDA Dietary Guidelines:
The recommended calories for a 21-25 year-old moderately active adult is 1400 calories per lunch for Male and 1100 calories per lunch for female.
- The suggested fat intake is around 35% of total calories
- The Protein intake should be in the range of 10-35% of total calories
- The healthy range of sodium intake is 1100-1400 milligram per lunch.

Search:

1

NOOO!!!! You don't wanna get too skinny too fast!!! Eat some more!!

1

Showing 1 to 1 of 1 entries

Previous 1 Next

Short on Time? Tab:

Meal Mate

Protein Optimizer

Short on Time?

How many meals are you preparing?

6

Calories Limit per Meal

0

800

1,500

Select 2 Proteins


☒ beef

☐ chicken

☒ shrimp

☐ pork

GO



The usual meal preparer in a household spent an average of 51 minutes in meal preparation on an average day.
STOP EATING FROZEN PIZZA
These are REAL FOOD that can be prepared in 22 min!!

Search:

	title	rating	calories	protein	fat	sodium	serving	links
10560	Easy Beef Lo Mein	4.375	687	48	25	1892	5	Recipe!
4433	Roasted Hot-Honey Shrimp With Bok Choy and Kimchi Rice	3.75	448	29	17	1919	3	Recipe!

Showing 1 to 2 of 2 entries

Previous 1 Next

Conclusions

The R Shiny App MealMate has a lot of functionality and provides benefits to Epicurious in the form of engagement with users, recipes based on specific inputs, and has the potential to provide increased ad revenues from increased user interest and pure website traffic.

As the website currently does not support granular suggestions, this app would greatly increase Epicurious' appeal to their target segment. Furthermore, the application provides utility to users in the form of automatic optimizations for maximum protein, reduction in food waste due to optimal serving size, and provides users with the ability to cook specific recipes with relevant ingredients.

We discovered that a lot of our target audience wants to maximize protein intake per meal. These users, who usually care a lot about health, tend to count calories as well and this application solves both of those problems by incorporating a Maximum Calories per meal slider.

While the app has many uses today, we want to improve upon several key areas:

- 1) Up-to-date data:
 - i. The app currently is using an older dataset and some of the current nutritional values have changed as the recipes are being altered by the users. This could be solved in two ways: i) API and/or ii) Web Scraping
- 2) Added functionality:
 - i. Ideally, we would like to have the app display recipe instructions and nutritional facts within an output screen that users can follow rather than clicking the link to the website.
 - ii. Increasing optimization parameters would enable users to better create personalized meals.
 - iii. The addition of ingredients not found on their website to add unknown recipes in the future.
- 3) Improved Speed:
 - i. The application speed is not where we would like it to be. For this application to be user-facing ready, there would need to be improvements made to the base code.
- 4) Additional user constraints
 - i. We would like to add more user constraints (i.e. food allergies) that would help provide users with a better experience.

Altogether, this app provides users with a very interactive experience to a real life daily problem. MealMate also provides Epicurious with the tools necessary to stay competitive as an online customer-facing website that is reliant on customer traffic to generate revenue.

References

- 1) [Epicurious.com](https://www.epicurious.com)
- 2) [Kaggle](https://www.kaggle.com)
- 3) Food and Drug Administration (FDA)