Food Recipe Recommendation System and Rating Prediction

Abstract—**In recent years, due to the COVID-19, people have begun to try to cook at home, which has led to a large number of applications of online food recipes. As some of people getting benefit from online food recipes, we decided to work on this problem.**

We designed a food recipe recommendation system using content-based approach. Also, we developed a star rating prediction using recommendation system methods based on user-recipe interactions and traditional machine learning method based on recipe info only. Compared with rating prediction mean-squared-error, we found traditional machine learning methods worked better partly because of large data-set memory problem.

Keywords: Food Recipe Recommendation System, Rating Prediction, Machine Learning

I. Introduction

Food is something that people look for naturally, so they are constantly searching for appetizing recipes. And with increasing use of online food, recipe recommendation system and rating prediction can be economically beneficial to contributors improve their recipes and users to get what they can cook based on their ingredients.

Considering what we dealt with in Assignment 1, we designed a food recipe recommendation system and rating prediction model. Recipe recommendation systems play an important role in helping people find recipes that are of their interest and fit their eating habits. For food recipe recommendation system, we used content-based approach. After cleaning data and extracting content features like name and ingredients, we recommended related recipes based on top K cosine similarity recipes.

Also, we developed a star rating prediction using user-recipe interactions and recipe own information separately. For user-recipe interactions, we used complete latent factor model. And we combined content-based filtering and collaborative filtering as a mix model as recommendation system. And we tried a traditional machine learning approach by combining review text feature and numerical features. What's more, we got the kernel method idea from paper and tried this machine learning method using one single text feature. By comparing these three methods, we found traditional ML methods working better considering computer memory limit and running time limit. And with large enough data-set, traditional machine learning method seems to improve performance to some extent.

II. Related Literature

A. Past Similar Dataset

There are several other datasets that are frequently used like food recipe dataset. These differ from food recipe dataset in terms of size, shape and context of interaction. Table 1 shows most frequently cited datasets besides food recipe dataset ordered by number of ratings. Shown in table II-A

TABLE I
Prominent Alternative Datasets

Name	Domain	# Ratings
Book-Crossing	books	1.1m
EachMovie	movies	2.7m
Jester	jokes	4.1m
Amazon	many	82.8m
Netflix Prize	movies	100.5m
Yahoo Music	music	262.8m

B. Food Recipe Rating Prediction

Research focuses on developing a recommendation system for recipes based on a particular user's ratings of recipes. Majumder et al. propose a personalized recipe generator, which accepts a user's ratings and reviews of recipes, key ingredients, a recipe title, and calorie level, and returns a set of instructions for a dish to create[1]. The dataset used in this project was originally developed by Majumder et al. for this work. While this work has interesting results, it only implicitly predicts whether a user will like the result, instead of explicitly rating the recipe. The work done by Harvey et al. builds a recipe recommendation model tailored towards recommending healthy recipes based on users past preferences[2]. Similarly, Freyne and Berkovsky develop a healthy recipe recommendation system, with a focus on how to relate user preferences on food preparation to possible recipes[3]. Teng et al. develop a recipe recommendation model that attempts to capture ingredient similarities and possible substitutions[4]. Finally, work by Mao et al. attempts to predict whether a recipe will be popular by doing sentiment analysis of reviews[5].

C. Combining Content-based and Collaborative Filtering

Recommendation systems have formulated in parallel with the web. Initially Recommendation systems were based on demographic, content-based filtering and collaborative filtering. Currently, these systems are incorporating social information for enhancing a quality of recommendation process.

For betterment of recommendation process in the future, Recommendation systems will use personal, implicit and local information from the Internet. [6]

The hybrid filtering is a combination of more than one filtering approach [7]. The hybrid filtering approach is introduced to overcome some common problem that are associated with above filtering approaches such as cold start problem, over-specialization problem and sparsity problem. Another motive behind the implementation of hybrid filtering is to improve the accuracy and efficiency of recommendation process.

III. DATASET INTRODUCTION

In this Assignment, we selected "Food.com Recipe & Review datasets for our study. The website can be found on Kaggle. This dataset consists of 180K+ recipes and 700K+ recipe reviews covering 18 years of user interactions and uploads on Food.com (formerly GeniusKitchen). Data includes cooking recipes and review texts.

This dataset contains 2 original files that is described below.

1 **RAW_recipes.csv** contains recipe information. In total, there are 231636 recipes are stored in this dataset while each recipe has 11 features including recipe name, id, cooking minutes *etc*. Some of the features are the description and other features are integers that represent the basic numeric information of this recipe.

2 **RAW_interactions.csv** contains the users' evaluation towards every recipe they made. There are 1132366 reviews included in this dataset. Every review has the features of basic information for both the user who posted this review and the recipe they evaluated. Also, each review includes user's rating and their review text toward the recipe in the data.

Data Pre-processing for the data To give our model better applicability on the dataset, we first preprocessed the dataset. First of all, we noticed that the nutrition feature in the recipe dataset includes several numerical values which represent each value of food for human nutritional intake such as calories, fat, protein and so on. We believe that these values are essential for our rating model. Therefore, we take these nutritional indicators out of nutrition as a new feature and add each one as a separate feature into the dataset. Moreover, We believe that each food has a very different level of health depending on its nutrient content, and that the healthiness of a food can greatly affect the final rating of a recipe. So we calculated the proportion of nutritional value for each recipe according to the nutritional formula and added a new binary feature (healthy_index) for better prediction.

After we add those new feature into the recipe dataset, we now can merge these two datasets together for our predictions. According to the recipe, merge all of data in two datasets into one dataset called "data". Then split the data into training, validation and test sets for formal prediction. All the features are demonstrated down below.III

TABLE II
FEATURES OF DATASET

Feature	Type
user_id	int64
recipe_id	int64
review_date	datetime64
descriptions	string
total fat(PDV)	float64
sugar(PDV)	float64
sodium(PDV)	float64
protein(PDV)	float64
saturated fat(PDV)	float64
carbohydrates(PDV)	float64
healthy index	binary
minutes	int64
n_steps	int64
n_ingredients	int64
reviews	string

IV. ANALYSIS OF THE DATASET

Right before we begin our predictive task, we first analyze several important features for predicting the final rating in the dataset. Finding the connections between those features and conduct necessary truncation of the data set based on distribution to ensure accuracy of predictions.

A. Cooking minutes and n_steps

At first, we investigate the distribution of cooking minutes and number of steps features in the dataset. From the graph down below, we can find out that most of dishes will be cooked among in 10 to 150 minutes, and there are almost no dishes that takes longer than 200 minutes. Apparently, for those dishes that take more than 200 minutes to prepare, we will truncate these data to ensure the accracy of the prediction. Similarly, for n_steps feature, we use the same operation as minutes feature.IV-A

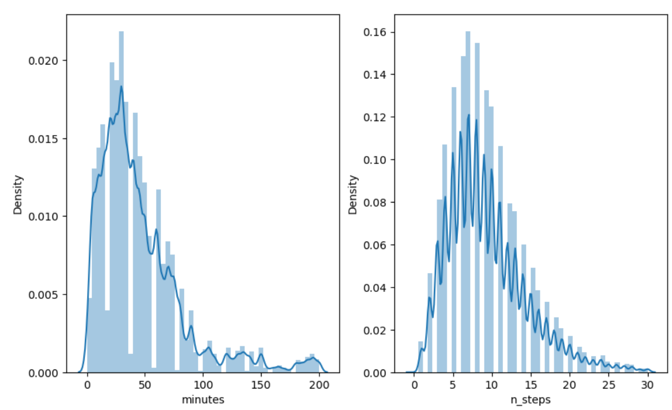


Fig. 1. Density distribution of minutes and n_steps features

B. rating and n_ingredients

The Fig 2 IV-B demonstrates the distribution of the rating and the number of ingredients in boxplot. Although n_ingredients feature in most of data is between 5 to 10, there are still some data with more than 20 materials, which should also be discarded as outlier.

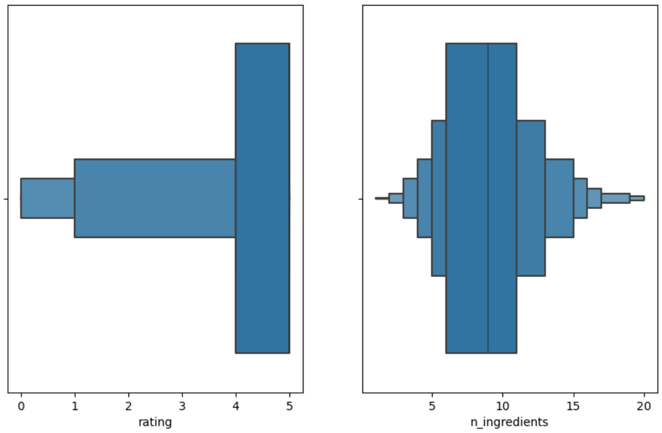


Fig. 2. Density distribution of rating and n_ingredients features

C. Correlated heatmap

Figure 3 IV-C represents the heat map of the correlation between the features. The correlation decreases from 1 to 0. A correlation of less than 0 indicates that a negative correlation is formed between two features. Since we want to make prediction for the rating features, we can pick the most relevant features and adjust their weights in the model according to this graph for our subsequent prediction.

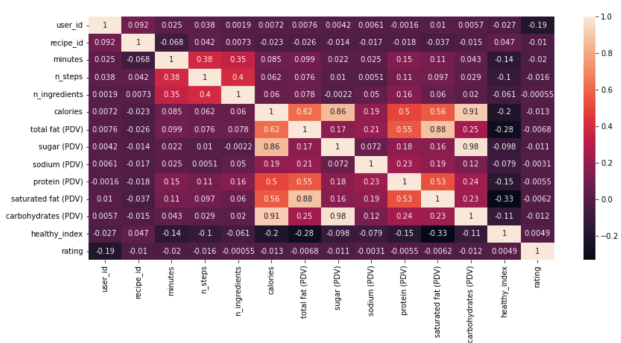


Fig. 3. Correlated heatmap between all features

D. relationship between rating and feature pair

1) relation of minutes, n_steps and rating: The graph down below IV-D1 represents the rating value for each minutes and n_steps pair. It is easy to see from the graph that neither the cooking minutes nor the number of steps has a particularly significant effect on the final rating.

2) relation of minutes, n_steps and rating: The graph down below IV-D2 represents the rating value for each minutes and n_ingredients pair. The graph also shows that neither the cooking minutes nor the number of ingredients is the decisive factor of final rating.

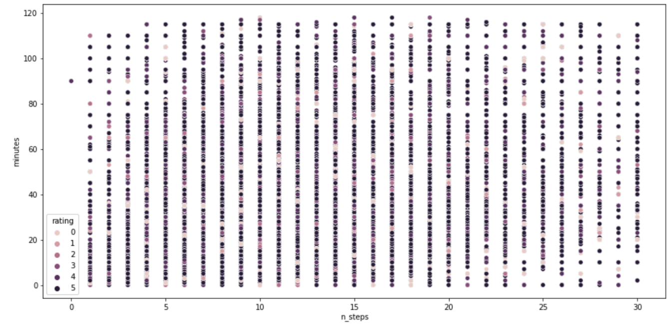


Fig. 4. Scatter plot of rating, minutes and n_steps

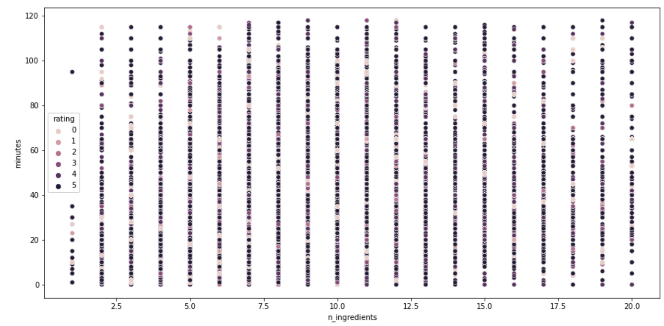


Fig. 5. Scatter plot of rating, minutes and n_ingredients

E. relation of healthy_index and rating

From the Fig 6 IV-E, we can find out that for those dishes that are unhealthy, their rating will be relatively low (or we can say the density of low rating feature are high). On the Contrary, for those dished that are healthy, the rating would be high. This phenomenon means that healthy_index might be an important feature, which may have a great influence on the final prediction results.

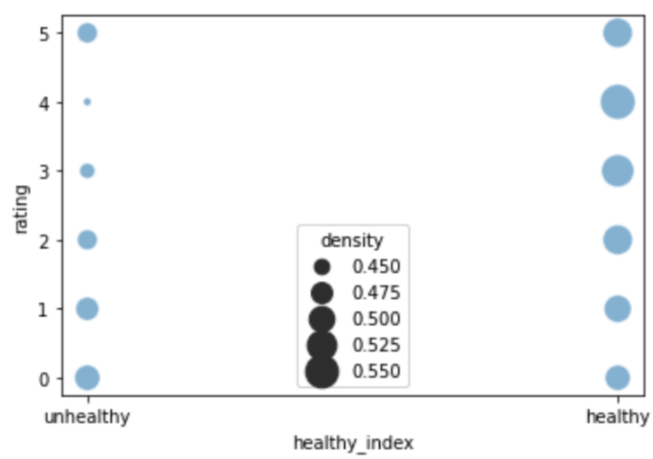


Fig. 6. Density distribution of rating and healthy_index features

V. TASKS

A. Data Pre-processing

From data analysis section, we can find that there are some missing values and outliers in the dataset. For missing values, we dropped the missing values because compared with the large dataset, the missing values are a small proportion. As for outliers, we truncated the scale of minutes from 0 to 200, the scale of step counts from 0 to 30 and the scale of ingredient numbers from 0 to 30.

For recipe recommendation system, we used recipe names, recipe tags, recipe ingredients and recipe descriptions as our content-based filtering method to recommend top K similar recipe.

We chose different features in rating prediction. For traditional machine learning method, we select minutes, number of steps, number of ingredients, healthy index and reviews as our model feature. For recommendation system method, we used user id and recipe id for collaborative filtering, review text for content-based filtering. Shown in table V-A.

TABLE III
RATING PREDICTION MODEL FEATURE

Model	Feature	Type
Traditional ML	minutes	int
	# steps	int
	# ingredients	int
	healthy index	binary
	reviews	string
Recommend System	user id	string
	recipe id	string
	reviews	string

B. Recommendation System

For food recipe recommendation system, we used content-based approach on recipe data-set. After cleaning data and extracting content features like name and ingredients, we recommended related recipes based on top K cosine similarity recipes.

C. Rating Prediction

We aim to predict the rating of a recipe, given the features in the recipe like review, minutes, tags, steps to make it, ingredients, number of ingredients, calories, total fat, sugar, sodium, protein, saturated fat, carbohydrates, healthy or not.

VI. MODELS

A. Recommendation System

We first cleaned and parsed the recipe name, ingredients and tags for each recipe (for example, 1 diced onion becomes onion). Next we encoded each recipe ingredient list using TF-IDF. From here we applied a similarity function to find the similarity between ingredients for known recipes and the ingredients given by the end-user. Finally, we can get the top-recommended recipes according to the similarity score. Pipeline shown in figure VI-A

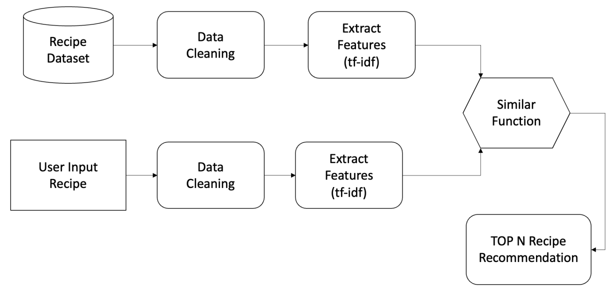


Fig. 7. Recommendation System Pipeline

$$k(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x}\mathbf{y}^T}{\|\mathbf{x}\|\|\mathbf{y}\|} \tag{1}$$

Mathematically, cosine similarity measures the cosine of the angle between two vectors. For the mathematically inclined out there, this is the same as the inner product of the same vectors normalized to both have length 1. With cosine similarity, the smaller the angle, the higher the cosine similarity: so we are trying to maximize this score.

B. Rating Prediction

1) Baseline Model:

We implement the baseline model from the same method of baseline model in assignment1, which will use the existing rating if user exist or the global average rating if user doesn't exist and we got the MSE result is about 1.6.

2) Latent Factor Model:

We used the latent factor model as below. This approach is simple to implement and works well, the only drawback is the lack of performance when dealing with sparse matrices. Because of too much features, each time we tune parameters will cost a very long time. However, the MSE is only around 1.4 which is not so satisfied.

$$\arg\min_{\alpha,\beta,\gamma} \text{ objective } (\alpha, \beta, \gamma) \tag{2}$$

3) Machine Learning Model Combining Numerical and Text Feature:

Firstly, we did review text content analysis and used the principles of natural language process (TF-IDF). After getting text corresponding feature, we combined text feature and numerical features we selected(number of steps, minutes, number of ingredients, health index). Finally, we used xgboost regression and logistic regression to fit the model. Pipeline shown in figure VI-B3

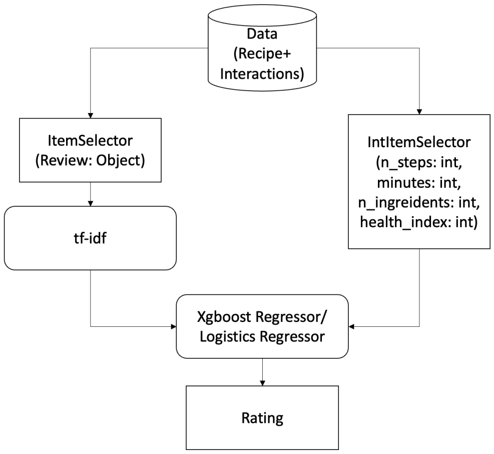


Fig. 8. Rating Prediction Pipeline

4) Model Combining Content-based Filtering and Collaborative Filtering:

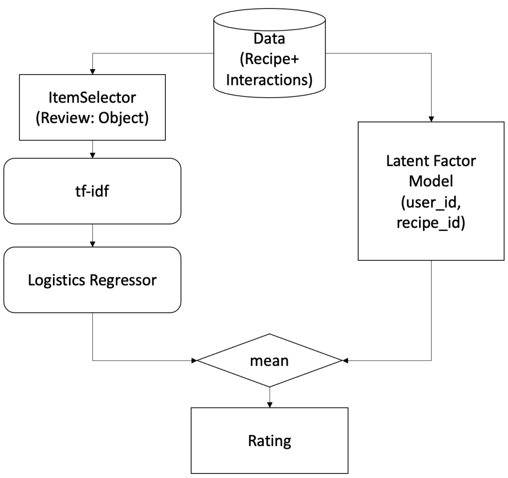


Fig. 9. Rating Prediction Pipeline

There are two main methods to approach this problem. Firstly, we did review text content analysis and used the principles of natural language process (TF-IDF and logistics regression). This method lacks the insights that can be drawn from the relationship between costumers and items. The second one is based on recommendation systems, specifically on collaborative filtering, and focuses on the reviewer's point of view. Use latent factor model. This method ignores any information from the review text content analysis. And after getting two results, we simple get the mean of two methods as this method result. Pipeline shown in figure VI-B4

5) Rating Prediction Model using Ingredients Text Feature: VI-B5 VI-B5

We applied linear regression and kernel methods, and we only choose ingredients[4] as feature. We defined our loss

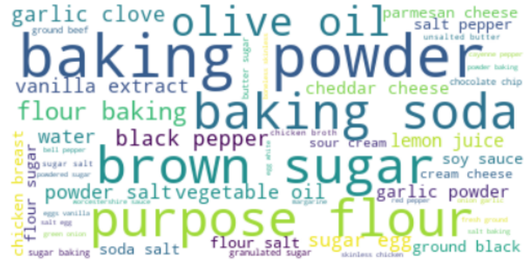


Fig. 10. Word Cloud of Ingredients with Rating 1

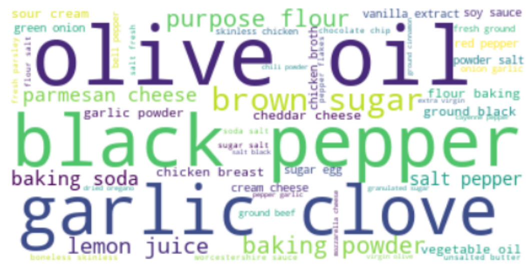


Fig. 11. Word Cloud of Ingredients with Rating 5

function(shown in below). For linear regression, we learn the vector θ to minimize the loss, where predictions are shown in below.

$$J(\theta)=\frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\left(h\left(x^{(i)}\right)-y^{(i)}\right)^{2} \tag{3}$$

$$h_{\text{lin reg}}(x)=\theta^{T}x \tag{4}$$

With this prediction function and loss function, we update θ iterantly using full-batch gradient descent. Letting X be the design matrix and letting α our learning rate hype-parameter, we update as below.

$$\theta \leftarrow \theta+\alpha(Y-X\theta)X \tag{5}$$

Our prediction for a new embedding is the inner product with θ, and so each present ingredient contributes some weight (defined by θ) to the overall ingredient prediction. Thus, at a high level, we may understand of linear regression as thinking of each recipe to be the sum of its parts. With the BERT[8] embedding, each rating is a linear combination of the feature vector elements. Next, we applied kernel methods. With this, we aim to learn β to predict as

$$h(x)=\sum_{i=1}^{n}\beta_{i}K\left(x^{(i)},x\right) \tag{6}$$

We have previously shown that a linear kernel is equivalent to linear regression. A kernel function K(x1, x2) acts as an inner product between virtual feature vectors derived from x1 and x2. Given a kernel function, to learn the parameter β, we update as below. We considered two kernel functions. We first considered a square kernel, as

$$\beta \leftarrow \beta+\alpha(Y-K\beta) \tag{7}$$

$$K_{\text{square}}(x, z) = \left(x^T z\right)^2 \tag{8}$$

Recipes with entirely different ingredients should not influence the ratings of each other. The Gaussian kernel satisfies exactly that, with large values of KGaussian(x, z) when the distance between x and z is small, and small values of KGaussian(x, z) otherwise. The value of σ controls the rate at which the weight contribution from increasingly. Next, we consider a Gaussian kernel. Letting σ be a spread hyperparameter, we define as

$$K_{\text{Gaussian}}(x, z) = \exp\left(-\frac{\|x - z\|_2^2}{2\sigma^2}\right) \tag{9}$$

VII. Results and Conclusions

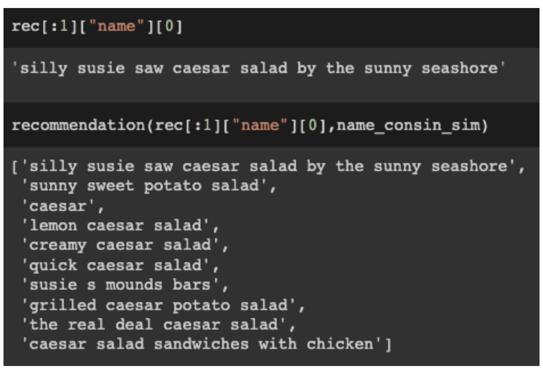
A. Recommendation System

1) Evaluation:

It's worth noting that there was no concrete way of assessing the performance of the model so I had to evaluate the recommendations manually. To be honest, this ended up actually being quite fun. We also discovered a tonne of new recipes!

2) Result:

Shown in Figure VII-A2.



```
rec[:1]["name"][0]
'silly susie saw caesar salad by the sunny seashore'
recommendation(rec[:1]["name"][0],name_consin_sim)
['silly susie saw caesar salad by the sunny seashore',
 'sunny sweet potato salad',
 'caesar',
 'lemon caesar salad',
 'creamy caesar salad',
 'quick caesar salad',
 'susie s mounds bars',
 'grilled caesar potato salad',
 'the real deal caesar salad',
 'caesar salad sandwiches with chicken']
```

Fig. 12. Recommendation System Result

3) Conclusion:

Take one recipe for example. When we want a caesar salad, it will recommend something about caesar salad to us. It makes sense.

B. Rating Prediction

1) Evaluation:

We used mean-squared-error as our rating prediction model evaluation. MSE has several pros. One is that mean-squared-error is more sensitive to outlier than mean-absolute-error. Another is that for SGD, mean-squared-error is easy to compute gradient as loss function.

$$MSE = \frac{\sum_{i=1}^{N}(\text{Predicted}_i - \text{True}_i)}{N} \tag{10}$$

2) Result: VII-B2 VII-B2

Latent Factor Parameter: k = 2; λ = 0.0001

TABLE IV
MSE Results of Rating Prediction

Model	MSE
Baseline	1.5973
Latent Factor Model	1.3544
Combining Numerical and Text Feature	0.2963
Combining Numerical and Text Feature	0.6257
Content-based and Collaborative Filtering	0.7302

TABLE V
The MSE achieved by different models on test dataset

Model & Kernel	Test loss
Linear Reg. one-hot	0.1730
Gaussian one-hot	0.6054
Linear Reg. BERT	0.1826
Gaussian BERT	0.3994

3) Conclusion:

- Traditional machine learning method works better than recommendation system method. Partly because the dataset is too large to run it thoroughly, which makes tuning parameter in latent factor model too hard.
- Review feature is quite important for food recipe rating prediction model. Both (recommendation system and traditional machine learning) works better combining with review text model.
- As the result shows VII-B2VII-B2, using one-hot embedding with linear regression model is much better. Future studies could examine increase the size of the training dataset. This may allow kernel methods to outperform linear regression.
- We can try fast factorization machine or rank factorization machine model in the future, which is a better recommendation system method.

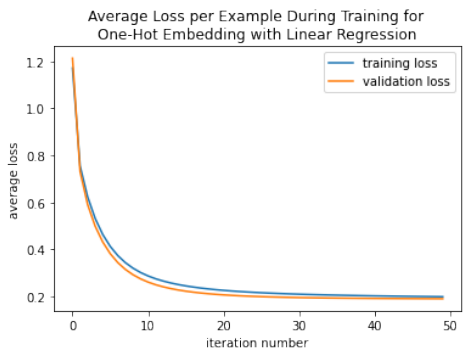


Fig. 13. The loss over the training and validation sets for One-Hot Embedding with linear regression

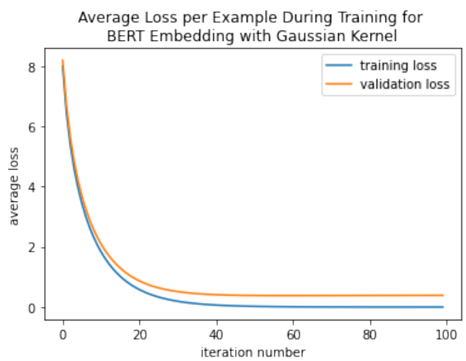


Fig. 14. The loss over the training and validation sets for BERT Embedding with Gaussian Kernel

REFERENCES

[1] Majumder, B. P., Li, S., Ni, J., & McAuley, J, (2019), *Generating personalized recipes from historical user preferences*, arXiv preprint arXiv:1909.00105.

[2] Harvey, M., Ludwig, B., & Elsweiler, D, (2013, October), *You are what you eat: Learning user tastes for rating prediction*, In International symposium on string processing and information retrieval (pp. 153-164). Springer, Cham.

[3] Freyne, J., & Berkovsky, S, (2010, February), *Intelligent food planning: personalized recipe recommendation*, In Proceedings of the 15th international conference on Intelligent user interfaces (pp. 321-324).

[4] Teng, C. Y., Lin, Y. R., & Adamic, L. A, (2012, June), *Recipe recommendation using ingredient networks*, In Proceedings of the 4th annual ACM web science conference (pp. 298-307).

[5] Mao, X., Rao, Y., & Li, Q, (2013, November), *Recipe popularity prediction based on the analysis of social reviews*, In 2013 International Joint Conference on Awareness Science and Technology & Ubi-Media Computing (iCAST 2013 & UMEDIA 2013) (pp. 568-573). IEEE.

[6] Thorat, P. B., Goudar, R. M., & Barve, S, (2015), *Survey on collaborative filtering, content-based filtering and hybrid recommendation system* International Journal of Computer Applications, 110(4), 31-36.

[7] Geetha, G., Safa, M., Fancy, C., & Saranya, D, (2018, April), *A hybrid approach using collaborative filtering and content based filtering for recommender system* In Journal of Physics: Conference Series (Vol. 1000, No. 1, p. 012101). IOP Publishing.

[8] Devlin, Jacob, et al. "Bert: Pre-training of deep bidirectional transformers for language understanding." arXiv preprint arXiv:1810.04805 (2018).