Data Science Assessment Conclusions and Discussion

In this assessment, we were tasked with analyzing to the provided data set and drawing four conclusions from the data. The four conclusions from the data that were gathered are:

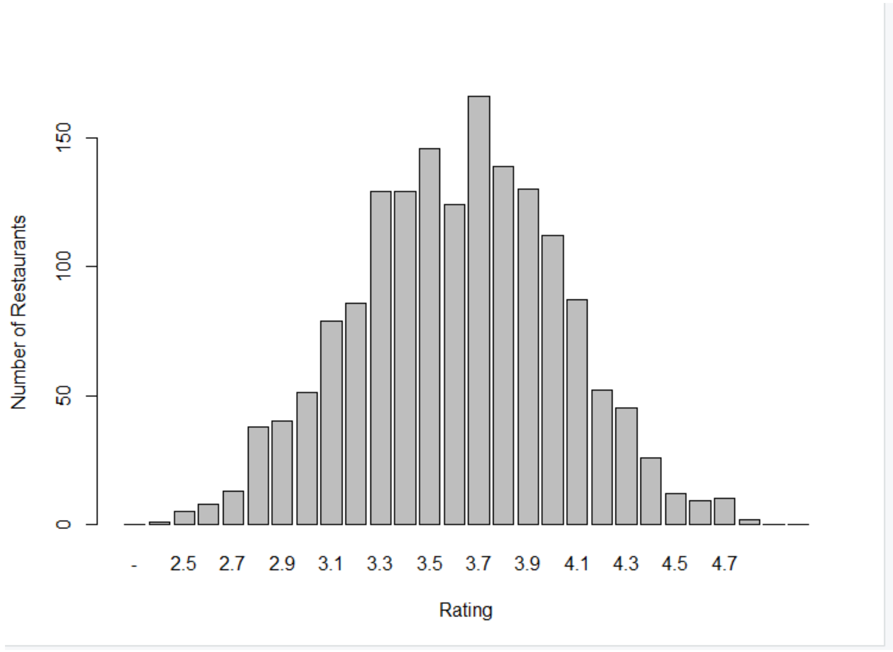
1. Using linear regression to develop a scoring algorithm based on Ratings versus Votes
2. Find the highest scoring restaurants from the given scoring algorithm
3. Take a cuisine type and find all the restaurants that serve that kind of cuisine. Then, of those restaurants, find the restaurant(s) with the highest score, lowest average cost, and lowest minimum order amount
4. Find the closest restaurant to a given restaurant and provide the score of that restaurant and cook time of that restaurant

<u>Discussion of Conclusion 1:</u>

To start, I set out wanting to develop a scoring algorithm for the restaurants in the data set. This would allow users to find the restaurants that other users have found highly enjoyable, without having to decide for themselves if they should weigh Ratings or Vote count higher.

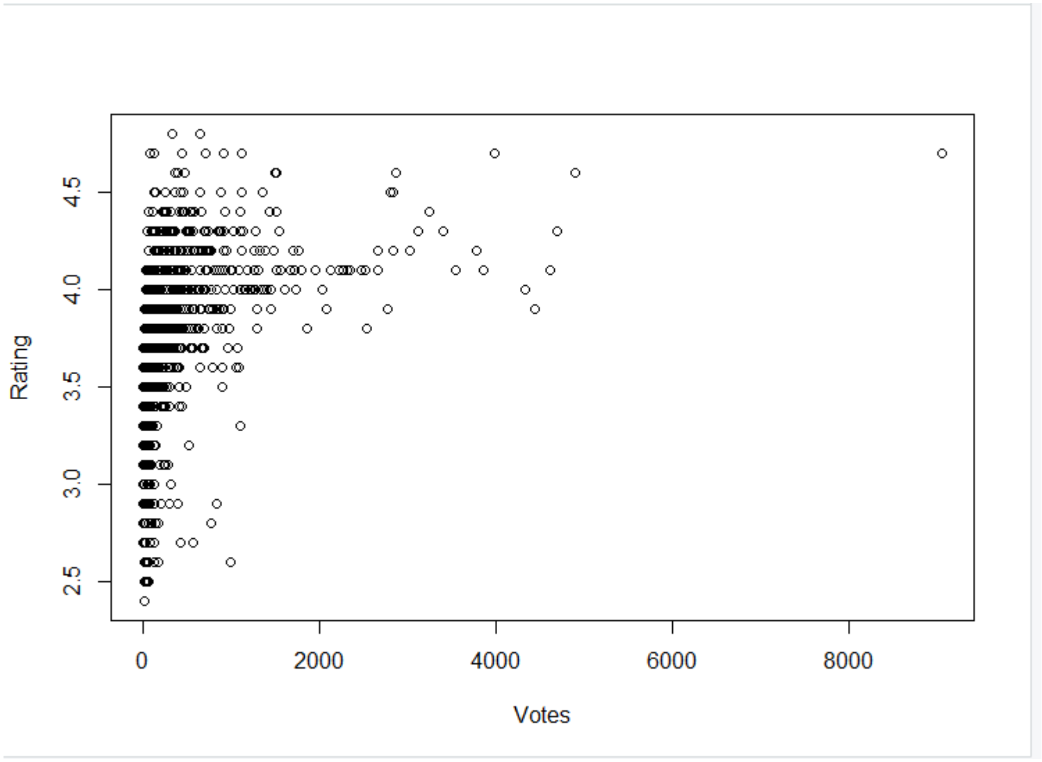
I start out all of my analysis by cleaning the data set, removing any rows that don't have an entry for any column. This includes rows where a restaurant had a rating of NEW, restaurants that had a rating but no votes, or any permutation of missing variables. While there could be some valuable analysis that could be done on the NEW restaurants, without any sort of rating besides new it would be hard to assess anything besides location and cook time.

After cleaning the data set, I plot the ratings to see if any analysis done would be significant

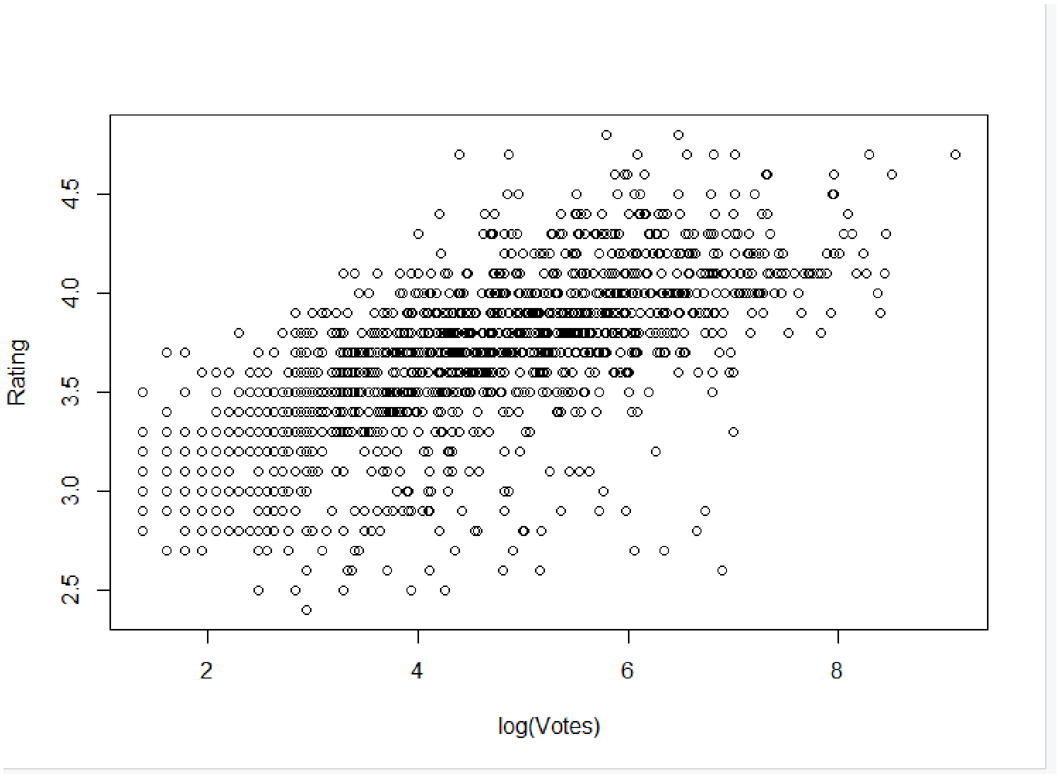


.

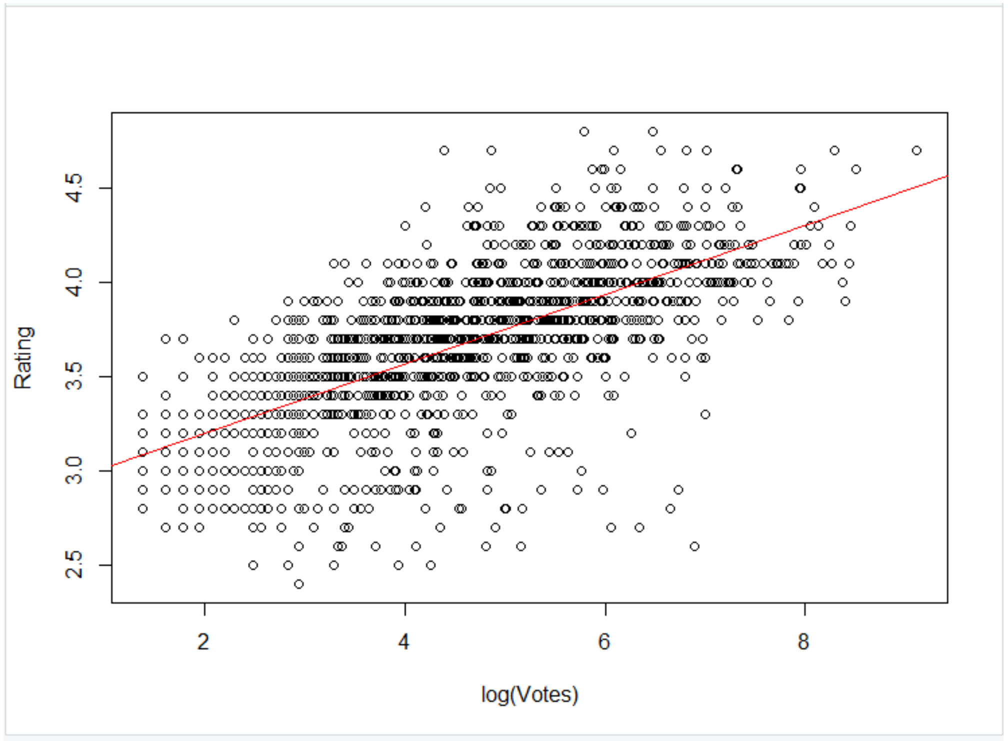
Seeing that the data was approximately normally distributed, I kept going with the analysis. All of the data in the file was stored as a factor class, so conversions were necessary to be able to form a simple linear regression model. Afterwards, I create a scatterplot of Rating versus Amount of votes, which produces this graph:

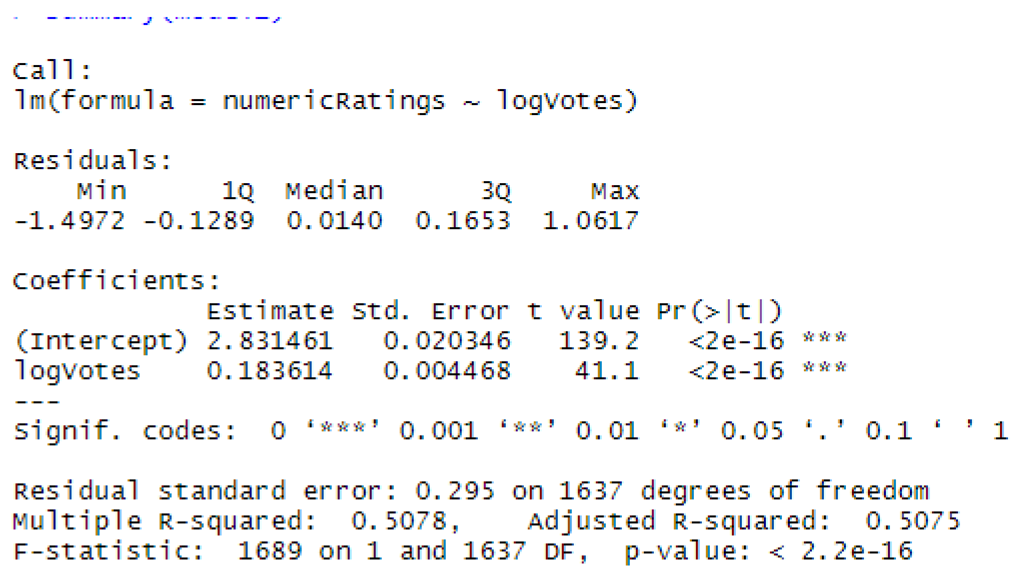


This shows that the data is skewed and the only way to get meaningful analysis is to transform the data. So, I take the natural log of votes and replot the data.



This data looks much more suitable to regression analysis, so I create a simple linear regression model of rating versus natural log(Votes). I think plotted the linear regression line and analyzed the calculated coefficients.





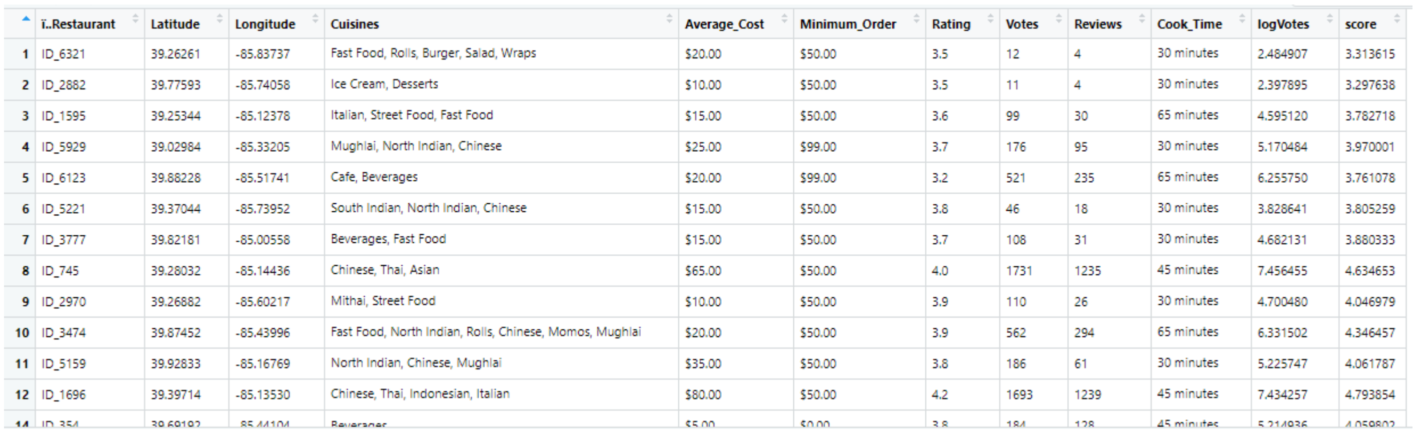
```
Call:
lm(formula = numericRatings ~ logVotes)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.4972 -0.1289  0.0140  0.1653  1.0617 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 2.831461   0.020346   139.2   <2e-16 ***
logVotes    0.183614   0.004468    41.1   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.295 on 1637 degrees of freedom
Multiple R-squared:  0.5078,	Adjusted R-squared:  0.5075 
F-statistic:  1689 on 1 and 1637 DF,  p-value: < 2.2e-16
```

Looking at the outputs, we can see that it is statistically significant that the natural log(Votes) has a positive effect on the ratings. This is reflected in the regression line plotted in red. This conclusion is what I used in my next one, which was developing the rating system for each restaurant.

Discussion of Conclusion 2:

After performing the linear regression, I decided to create a scoring system using both ratings and votes, and to use the calculated coefficient for the natural log(Votes) as the weight in algorithm. Since the calculated coefficient was 0.1836, I use a weight of 0.18 on the natural log(Votes) a restaurant receives, and a weight of (1 – 0.18) on the rating the restaurant has. This gives me values that are almost always contained within a 0 – 5 scale, where only restaurants that have a very high score (> 4.5) and a high amount of votes (> 800 votes) have a value above 5. I go back through and set any values above 5 to 5, creating a scoring system that has 0 as its lowest value and 5 as its highest value.

To do all of this, I add the natural log(Votes) as a column to the cleaned data set, and then use the described algorithm above to create a new column that contains the calculated score. The algorithm can be fully seen in the code. Here is a snippet of the data set that contains the newly calculated score:



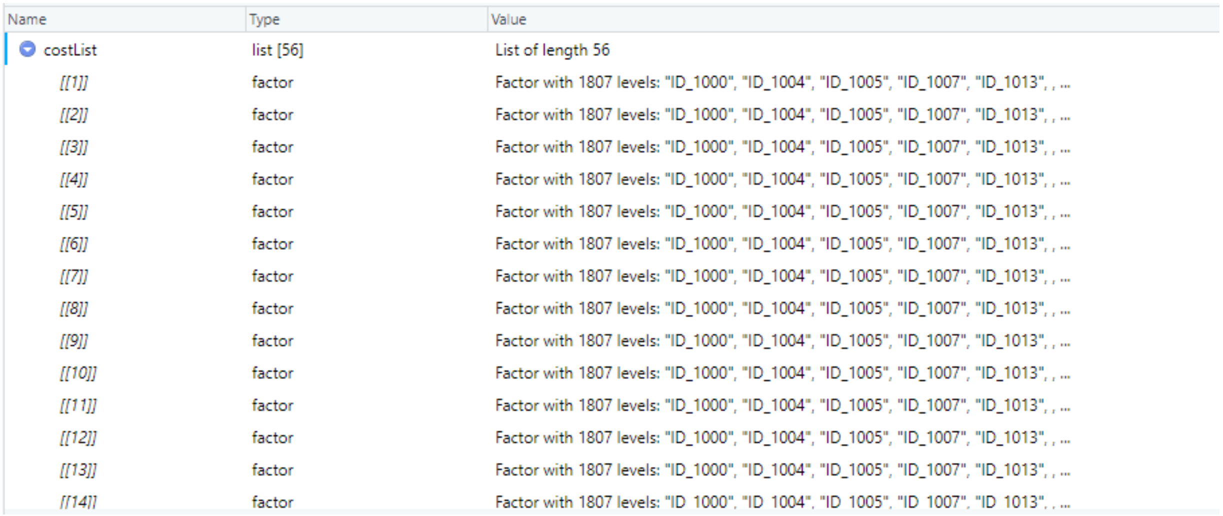
	ï..Restaurant	Latitude	Longitude	Cuisines	Average_Cost	Minimum_Order	Rating	Votes	Reviews	Cook_Time	logVotes	score
1	ID_6321	39.26261	-85.83737	Fast Food, Rolls, Burger, Salad, Wraps	$20.00	$50.00	3.5	12	4	30 minutes	2.484907	3.313615
2	ID_2882	39.77593	-85.74058	Ice Cream, Desserts	$10.00	$50.00	3.5	11	4	30 minutes	2.397895	3.297638
3	ID_1595	39.25344	-85.12378	Italian, Street Food, Fast Food	$15.00	$50.00	3.6	99	30	65 minutes	4.595120	3.782718
4	ID_5929	39.02984	-85.33205	Mughlai, North Indian, Chinese	$25.00	$99.00	3.7	176	95	30 minutes	5.170484	3.970001
5	ID_6123	39.88228	-85.51741	Cafe, Beverages	$20.00	$99.00	3.2	521	235	65 minutes	6.255750	3.761078
6	ID_5221	39.37044	-85.73952	South Indian, North Indian, Chinese	$15.00	$50.00	3.8	46	18	30 minutes	3.828641	3.805259
7	ID_3777	39.82181	-85.00558	Beverages, Fast Food	$15.00	$50.00	3.7	108	31	30 minutes	4.682131	3.880333
8	ID_745	39.28032	-85.14436	Chinese, Thai, Asian	$65.00	$50.00	4.0	1731	1235	45 minutes	7.456455	4.634653
9	ID_2970	39.26882	-85.60217	Mithai, Street Food	$10.00	$50.00	3.9	110	26	30 minutes	4.700480	4.046979
10	ID_3474	39.87452	-85.43996	Fast Food, North Indian, Rolls, Chinese, Momos, Mughlai	$20.00	$50.00	3.9	562	294	65 minutes	6.331502	4.346457
11	ID_5159	39.92833	-85.16769	North Indian, Chinese, Mughlai	$35.00	$50.00	3.8	186	61	30 minutes	5.225747	4.061787
12	ID_1696	39.39714	-85.13530	Chinese, Thai, Indonesian, Italian	$80.00	$50.00	4.2	1693	1239	45 minutes	7.434257	4.793854
14	ID_354	39.69192	-85.44104	Beverages	$5.00	$0.00	3.8	184	128	45 minutes	5.214936	4.059802

Finally, this calculated score can be used to find restaurants that are above a certain score, or to possibly find the restaurants with the maximum score. For the sake of this program, the only further conclusion that was implemented was finding the maximum score of all the restaurants. However, there are a lot more conclusions that could be drawn using solely the calculated score, showing how powerful of a metric it is.

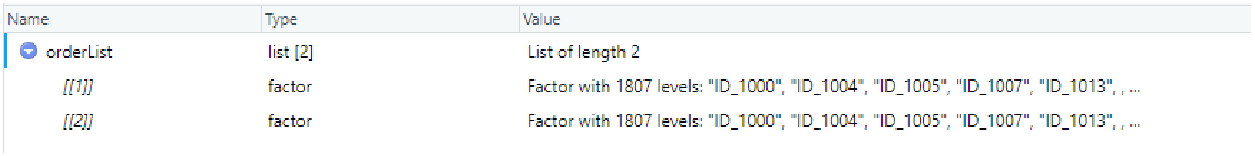
<u>Discussion of Conclusion 3:</u>

The purpose of Conclusion 3 is to allow a user to search restaurants based on the kind of cuisine they want, and to give that user the restaurants of that cuisine that have the lowest average cost, the lowest minimum order amount, and the restaurant with the highest score.

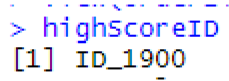
This was done by looping through data set and adding any row that contained the desired cuisine type to a new data frame. From there, the lowest average cost was identified, the lowest minimum order amount was identified, and the highest score was identified. Finally, every restaurant that shared the lowest average cost had its ID stored in a list, every restaurant that shared the lowest minimum order amount had its ID stored in a list, and the restaurant with the highest score was stored. For the sake of this program, I used the cuisine type of "Fast Food", but this analysis will work on any of the provided cuisine types. Below or the list of ID's of "Fast Food" restaurants that have the lowest average cost, the lowest minimum order amount, and the highest score.



Name	Type	Value
costList	list [56]	List of length 56
[[1]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[2]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[3]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[4]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[5]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[6]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[7]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[8]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[9]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[10]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[11]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[12]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[13]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[14]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...



Name	Type	Value
orderList	list [2]	List of length 2
[[1]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...
[[2]]	factor	Factor with 1807 levels: "ID_1000", "ID_1004", "ID_1005", "ID_1007", "ID_1013", , ...



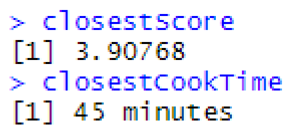
```
> highscoreID
[1] ID_1900
```

The final output shows that the restaurant with ID_1900 is the highest scoring “Fast Food” restaurant.

<u>Discussion on Conclusion 4:</u>

For the final conclusion, I wanted to find the restaurant that was closest to a given restaurant and give the user that restaurant’s score and cook time. This could serve as a possible restaurant recommendation to the user.

This was done by taking the latitude and longitude of a restaurant, for this program’s case it was for the restaurant ID_332 (but it will work with any given restaurant ID), and looping through the data set and finding the minimum of the sum of the absolute value in the difference between the given latitude and the data’s latitude and the given longitude and the data’s longitude. The algorithm for this can be found in the code. From there, I read in the calculated restaurant and print the score and the cook time to the user. Those outputs can be found below:



```
> closestScore
[1] 3.90768
> closestCookTime
[1] 45 minutes
```