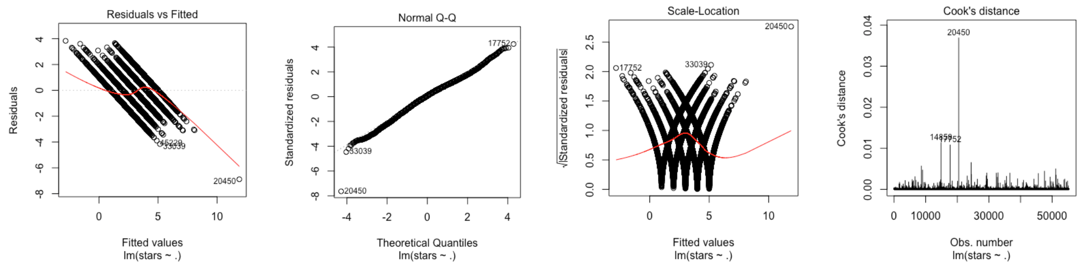
Yelp Review Ratings Prediction

By Aiden Song, Shane McIntyre (Group 12)

1. Introduction and Motivation

<u>Intro:</u> One of the most popular review websites used around the world is Yelp. The company holds an annual project in which students are challenged to interpret and analyze the information on their websites. Yelp gives out data sets for students to analyze. Some different types of analyzation Yelp likes to explore are Photo Classification, Natural Language Processing and Sentiment Analysis, and Graph Mining. For our project, we looked into the Natural Language Processing and Sentiment Analysis. From this, we were given 3 yelp data sets; training data, validation data, and test data. Our model seeks to analyze any given yelp review, and try to predict the number of stars the review will get, ranging from 1 to 5, with 1 being a very negative review and 5 being a very positive review. This allows us to analyze what words and phrases in a review text represent how a customer may perceive their experience at a restaurant. With this knowledge, you can predict how well a customer likes or dislikes a specific restaurant. In our project, we will try to prove that the relationship between a yelp review text and the amount of stars they get behave linearly, and that more positive words lead to higher ratings and more negative words lead to lower ratings.

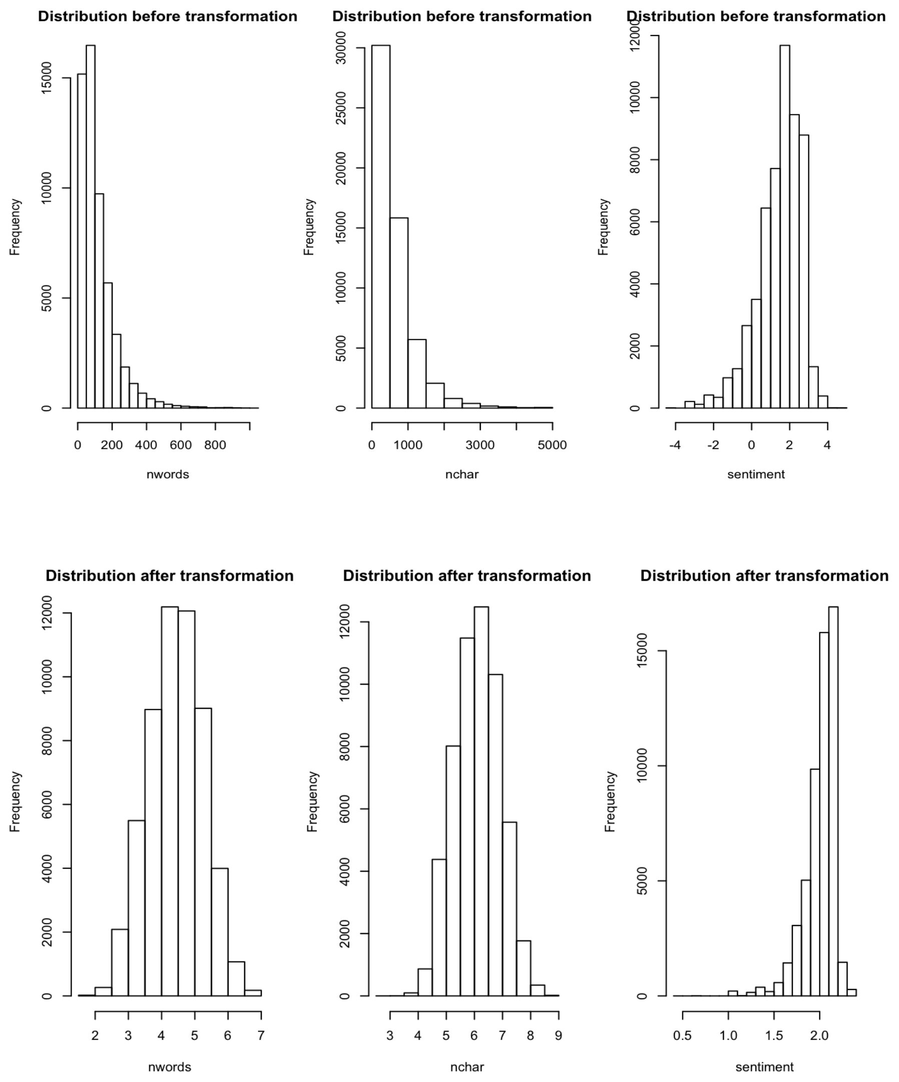
<u>Motivation</u>: Since stars on Yelp are between the values of 1-5. This lead us to look into how a linear regression model would predict star values. Originally, the benchmark model has 106 predictors used in its analyses of predicting star value. When checking the assumptions of linearity, the following graphs are explored:

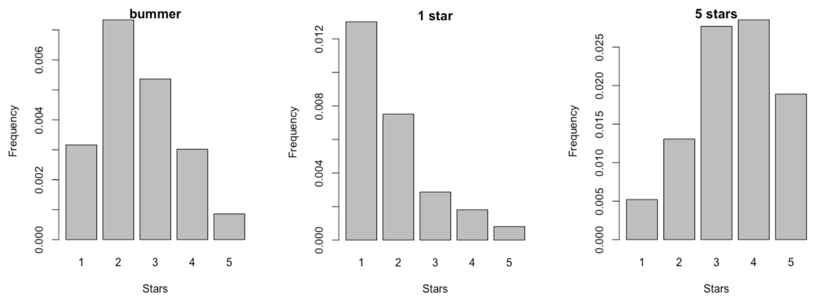


From these graphs, you can tell some of the assumptions of linearity are not followed. This led us to pursue some changes to the benchmark model to create a better prediction model. In terms of how accurate a prediction model is the most commonly analyzed value is the RMSE. The RMSE, or root mean square error, tells us how much error our residuals have in respect to the predicted mean line of our model. The benchmark model has a RMSE of .9313 when tested against the training data set. From the linearity assumptions, and the current RMSE, we wanted to investigate certain changes to the benchmark model to make the linearity assumptions fit better, as well as lower the RMSE.

2. Summary

Data Preprocessing: we started out by looking at the predictors that were already given to us. From looking at the distributions of some of the predictors, we realize that many of them are very skewed toward the right, hence we decided to apply a log transformation function so that the variables would look more normal. However for variables like cool, useful, funny, there are many that have many 0 values. Since log(0) does not make much sense, we decided to first add 1 to those variables and then apply the log transformation. Similarly, we apply the log transformation to the sensitivity variable so that everything would look more normal. The before and after of the distributions of the variables cool, useful, funny, nwords, nchars, and sensitivity can be seen in the plot below. After we transform some of our variables, we also look at some of the more commonly used words in the dataset and add those as predictors to help predict the star ratings. We added words like “5 stars”, “4 stars”, etc and it helped increase the overall accuracy.

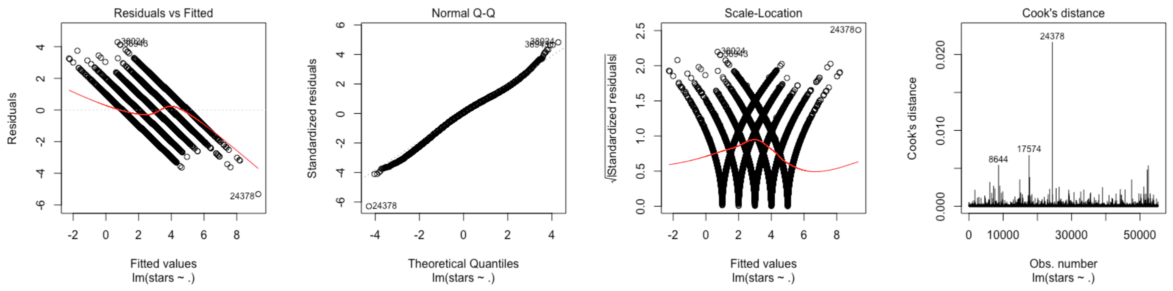




<u>Model Selection:</u> We trained different regression models and compare the results to see which one is performing well. We started with the multiple linear regression model since that was also the one we used for our benchmark model. Then we hypothesize that since we are adding a lot of predictors, we can use some penalized terms to make sure the model does not overfit. We added Lasso regularization, Ridge Regularization and the Elastic Net regularization. Unfortunately none of these model actually improves the accuracy of the model. In our final model, we ended up using just a multiple linear regression. We believe the reason why the penalized term didn't help is because we don't have that many predictors so the model doesn't overfit and by adding the regularization term, the model actually underfit. If we add more words as predictors, perhaps the regularization terms would help.

<u>Result:</u> In our final model, we use a multiple linear regression to predict star ratings. We also do some post processing to make sure that predictions that are greater than 5 round down to 5 and predictions that are less than 5 would round up to 1. Using this technique, along with the multiple linear regression model, our final RSME is around 0.8867.

3. **Model Assumption and Diagnostics**



From the above graphs, our new model, which has 89 new predictors and certain log transformations and also tests against the training data, makes the data fit better. While the assumption of linearity from the Residuals vs. Fitted graph drops off at the end, in our final model we test against the testing and validation data, and in those tests we have any predictions that are greater than 5 round down to 5, and any predictions that are less than 1 round up to 1. On Yelp, stars can only be between 1 and 5, and cannot go above or below. Doing this fixes the assumption of linearity. Similarly, in the Cook's distance graph the value at 24,378 appears to be skewing the data, but it's Cook's distance value is around 0.020, which is quite small. From the

QQ plot, you can see that homoscedasticity and normality are kept well. Overall, the assumptions of linearity, once star prediction values of more than 5 and less than 1 are rounded to their appropriate values, fit well. Now that we have a model that fits the data well we can analyze the RMSE. With the 89 new predictors and log transformations our RMSE when our model was tested against the training data was lowered to 0.8892, which puts us below a desired value of 0.9. This means are model is now more accurate at predicting a star value for any given review than the previous benchmark model.

4. Conclusion and Future work:

After trying out different models and data processing techniques, our final model has a RMSE of around 0.88 and the adjusted R-squared of around 0.53. The p-value is very very low and suggests that there are some predictors that are useful. Most of the predictors have p-value less than 0.05. However, words like “down”, “beer”, “wine” don’t seem to be a good predictor of the model. Words like “5 stars”, “amazing”, “highly” have positive coefficient and hence are more likely to be predicted to have a higher rating than words that don’t. And words like “gross”, “waste”, “terrible” have a negative coefficients hence are more likely to have a lower ratings than words that don’t. Interestingly, we found that nchar has a positive coefficient, meaning words that have more character are more likely to have a higher review. However, the predictor `nwords` has a very negative coefficient and suggests that more words would likely be predicted to have a lower rating.

Even though we were able to get the model RMSE to be below the 0.9 threshold, there are still a lot to improve upon. First, we can use more advanced NLP techniques for our predictors, for example, we use words like “5 stars” to be our predictors, however in some of the review, the user actually stated that “this is definitely not a 5 star!!”. As we can clearly see that this review states that this restaurant is nowhere near a 5 star, but because the model picks up the “5 stars” keyword and use it to predict that it might have a high rating. We can also try techniques such as the multinomial classification to see if it would lower the model accuracy.

5. Team Contribution:

Aiden: I started this project by first doing data preprocessing, I wrote the code to first bucket the predictors “cool”, “useful”, and “funny” since they were very skewed, however the bucketing technique didn’t work since most variables ended up become collinear with each other for some reason. I then use a log transform to transform the distribution of the data. I also wrote the code to do LASSO, Ridge, and Elastic Net regression.

Shane: I wrote the code to add in more predictors from the review text. I began by first looking at which words have a higher correlation with the ratings. I then wrote the code to add in those words. I also did a log transform for the sensitivity predictor and ended up this also helps lower the model RMSE. I also wrote the code to train the multiple linear regression model. I also use the technique to round the review > 5 to 5 and < 1 to 1.

During the presentation we both talked about things that we did to help improve the RSME.