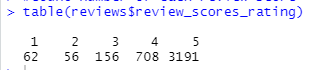
Tim Miller

15.071 | HW #6

**PROBLEM 1A**

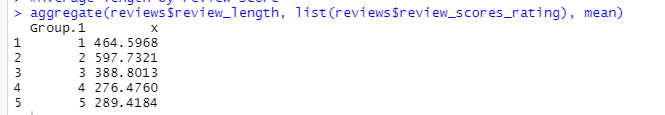


Using the results from above, it appears we have

* 62 “1” scores
* 56 “2” scores
* 156 “3” scores
* 708 “4” scores
* 3191 “5” scores

**PROBLEM 1B**

Aggregating by review\_scores\_rating, we find the following:



The immediate takeaway is that lower review scores have more characters in the text feedback. This is not surprising. Typically when someone leaves a bad review they will include a long list of complaints – driven by anger or frustration – justifying their low rating.

**PROBLEM 1C**

*corpus = tm\_map(corpus, tolower)*

This command changes all characters to lower case.

*corpus = tm\_map(corpus, removePunctuation)*

This command removes all punctuation from the corpus.

*corpus = tm\_map(corpus, removeWords, stopwords("english"))*

This command removes all stop works such as ‘the’ ‘and’ ‘but’ etc.

*corpus = tm\_map(corpus, removeWords, c("airbnb","apartment","location","place","room","host","stay"))*

This command removes words commonly found in Airbnb reviews. Since they are common based on the context of an Airbnb review, we do not want them to influence our predictions.

*corpus = tm\_map(corpus, stemDocument)*

This command returns the root of words (i.e., the word ‘stem’) in the corpus. This helps us remove some of the clutter and noise in the cluster and get to the most important part of what the review is saying.

Running the following command returns the following string for the first document:



**PROBLEM 1D**

*Positive reviews word cloud*



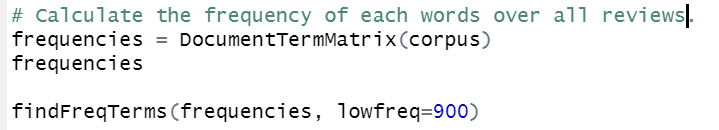
*Negative reviews word cloud*



Comments

* Different colors for different words, makes it easy to read
* In R Studio we can hover over each word to see the number of times it occurs
* Interesting to see ‘clean’ appear in both positive and negative word clouds. Likely in the negative reviews ‘not’ appears in front of the word ‘clean’ but that was filtered out with the stop words cleaning step. It would be interesting to concatenate words to see if ‘not\_clean’ would show up frequently.

**PROBLEM 1E**

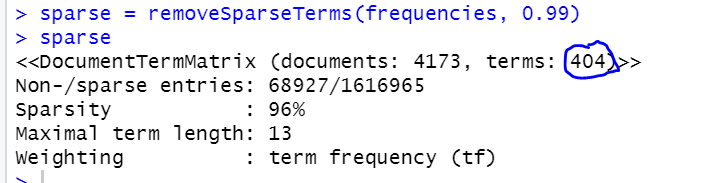


We find the following terms appear at least 900 times:

* recommend
* great
* clean
* nice

**PROBLEM 1F**

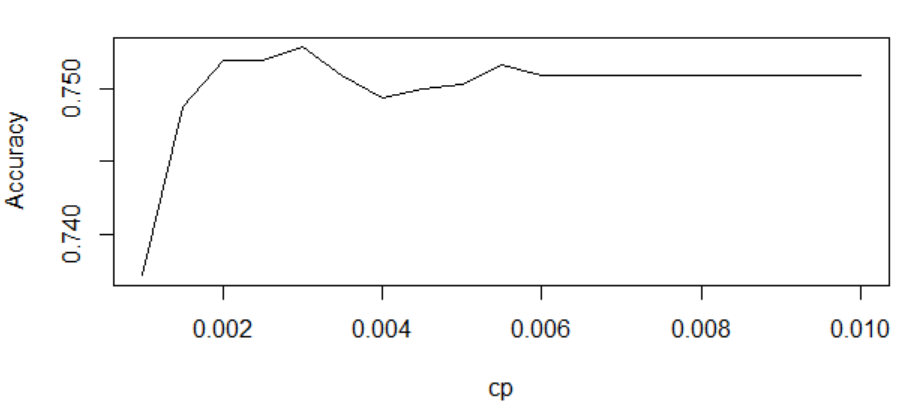
After removing the sparse words, we now only have 404 words in the corpus.



**PROBLEM 1G**

We want to model with the highest accuracy. So we plot cp vs Accuracy:

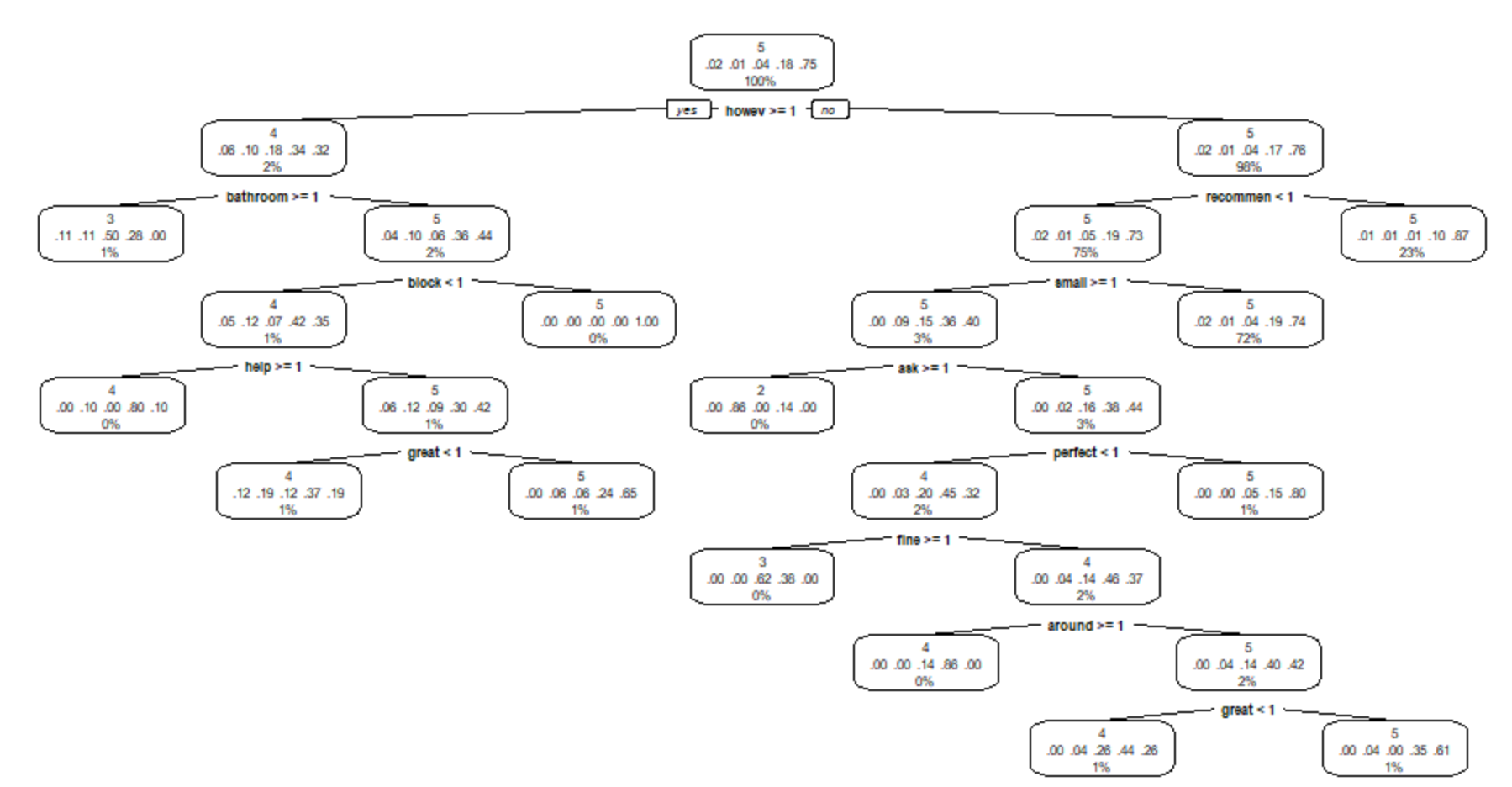




**We get a value of 0.0030 for cp.**

**PROBLEM 1H**

Plot of the tree with cp = 0.0030.



Looking at the tree there are 2 leaves with a score of 3 and 1 leaf with a score of 2. Breakdown of features for the node with predicted score of 2:

* ‘recommen’ is not mentioned
* ‘small’ is mentioned greater than or equal to 1 time
* ‘ask’ is mentioned greater than or equal to 1 time
* ~86% of the responses that fall in the bucket will be predicted to have a score of 2
* Remaining ~14% of the responses that fall in the bucket will be predicted to have a score of 4
* Contains <1% of responses

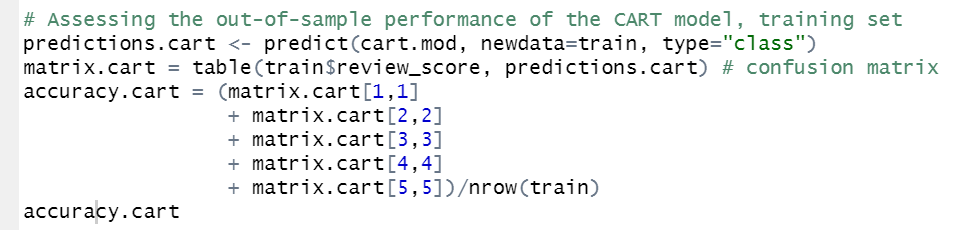
**PROBLEM 1I**

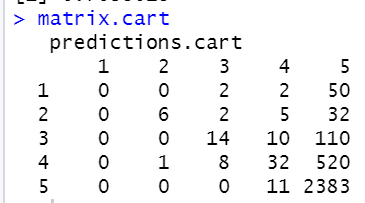
Yes, the tree aligns with my intuition. Looing at the predicted score of “2” node we see that the reviews will not contain “recommend, ” “small” & “ask will be mentioned more than 1 time. I can see a scenario where someone writes a review where they don’t include the word recommend because it was a bad experience, perhaps due to it being small. They might also include “ask” in there because they had to ask for a refund.

Additionally, for the node that predicts a score of 3 we see “however” and “bathroom” are mentioned more than 1 time. It is safe to assume that a reviewer might have started out the review saying the Airbnb was fine but then qualified it with “however” to add some additional detail e.g., the bathroom was not clean or had some other issue.

**PROBLEM 1J**

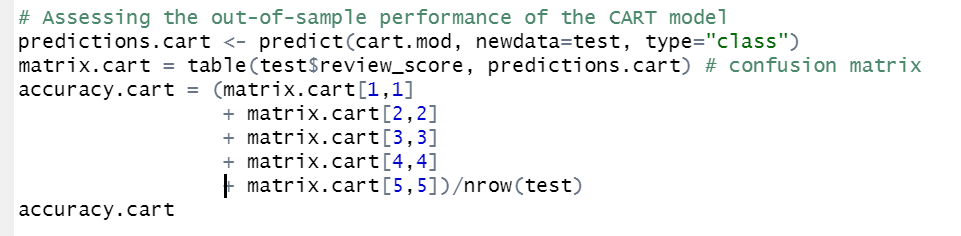
*Model accuracy for training set*

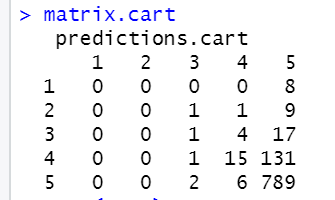




We find the model accuracy for the training set is **0.7638018**.

*Model accuracy for test set*

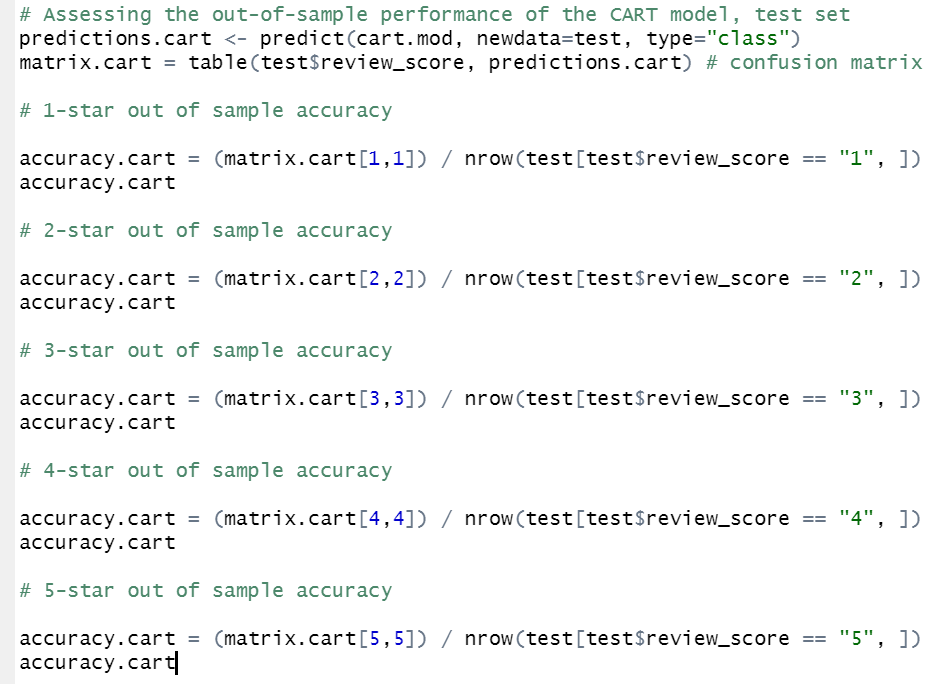




We find the model accuracy for the test set is **0.8172589**.

**PROBLEM 1K**

Code for calculating out of sample accuracy by star level:



Accuracy values by star level:

* 1 star – 0
* 2 star – 0
* 3 star - 0.04545455
* 4 star - 0.1020408
* 5 star - 0.9899624

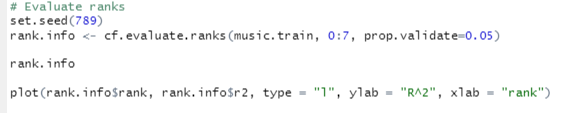
4 star and 5 star reviews have the highest out of sample accuracy. This makes sense given we have the most training data for those two categories.

**PROBLEM 1L**

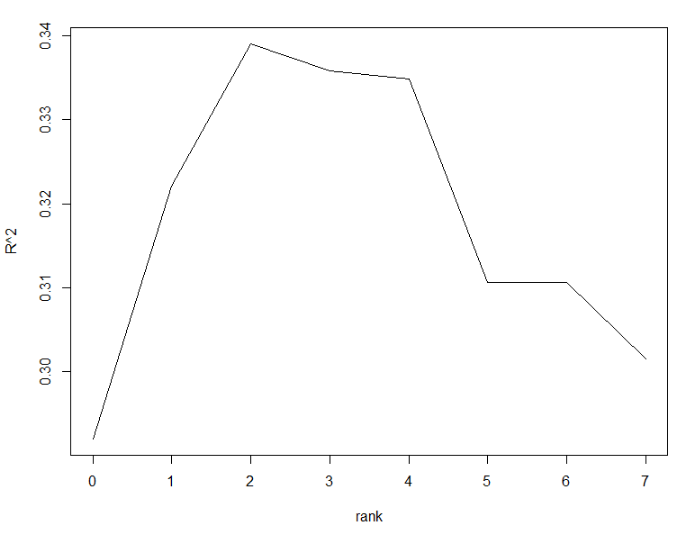
Bag of words representation might fail when there is a temporal component to the prediction. In other words, you cannot just create random bag of words: the order that the words occur in have some impact on the type of prediction that is made. For example, if you are predicting the sentiment of some time of text, you may need the words in a specific order to ensure you are pulling out the right context that is driving the sentiment of word source.

To improve bag of words, I would include some type of time tag or indicator on the training data to force the ordering of words and ensure that the model was trained with the right level of context.

**PROBLEM 2A**



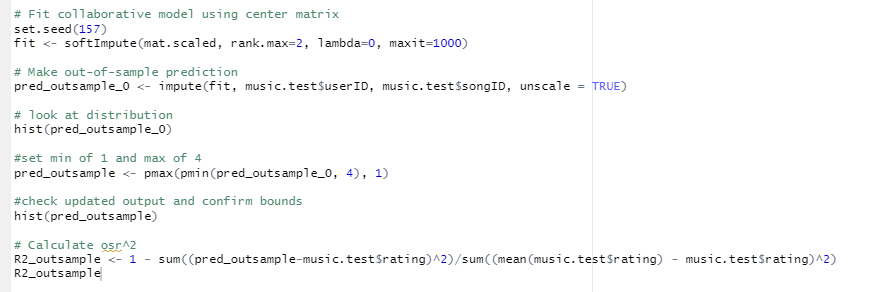
Plotting R2 vs rank we get



Given we want to maximize R2 we select a rank of 2.

Intuitively this makes sense. Thinking of how my friends and I listen to music, we typically prefer 2 or 3 different genres and focus on those. In the ‘Songs.csv’ database we see there are 7 different possible genres. With the optimal 2 ranks we have identified; the model is likely grouping people by their preference for ranking 2 – 3 genres. As stated before, this matches how I typically think people focus their music listening habits.

**PROBLEM 2B**

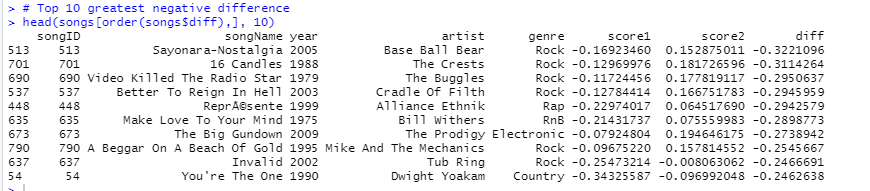


The out of sample R2 for the model is 0.3084477.

Note: in the above code you will see I set prediction scores at a minimum of 1 and a maximum of 4. This was based on the input data from ‘MusicRatings.csv’ where there was no score below 1 and no score above 4.

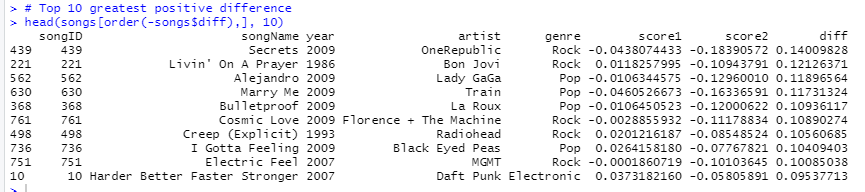
**PROBLEM 2C**

Top 10 songs with greatest negative difference



We see that archetype 2 has positive values for rock while archetype 1 generally has negative scores. Based on this sample of 10 songs we can generally say that archetype 2 likes rock while archetype 1 does not prefer it.

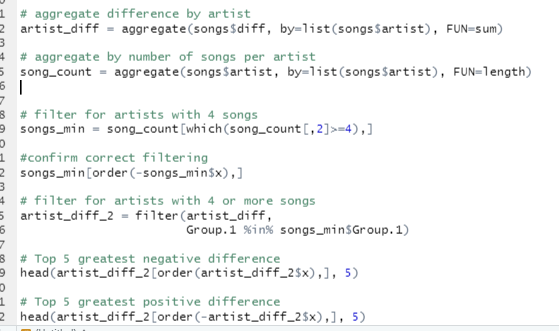
Top 10 songs with greatest positive difference



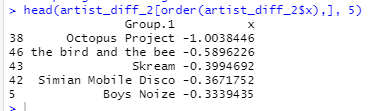
We see that archetype 1 and archetype 2 are quite different in their scoring on pop songs. Perhaps this suggests that archetype 1 might like pop sounding songs a little bit more than archetype 2. Interesting that archetype 2 has negative scores for rock songs, while previously they had positive scores for rock songs. My guess is that some of the bands classified as rock should really be under pop, therefore the more hardcore rock fans in archetype 2 are ranking them negatively.

**PROBLEM 2D**

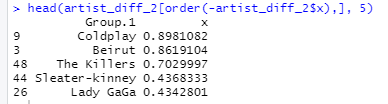
Data manipulation



Top 5 greatest negative differences



Top 5 greatest positive differences



We see that the top 5 negative / positive differences are different than what we saw the previous result in problem 2c. For the top 5 negative differences we see several electronic artists including Skream, Simian Mobile Disco, and Boys Noize. This suggests that archetype 2 likes electronic more than archetype 1.

For the top 5 positive differences we see different results than what we got in problem 2c. Most of the artists on this list are classified mostly as rock. Therefore this suggests overall that archetype 1 likes rock more than archetype 2.

**PROBLEM 2E**

*Unsure how to apply the model here*

**PROBLEM 2F**

*Unsure how to apply the model here*