Kasra Eshaghi, 1000512144, Kaggle team ID: Kasra Eshaghi

1. Exploratory Analysis and Data Preprocessing

The basic statistics of the dataset are summarized in Table 1 below. One may note that the dataset is clearly unbalanced.

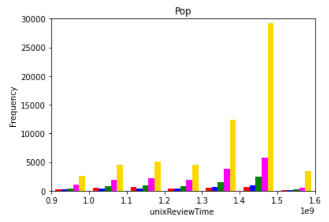
Table 1 – Dataset summary statistics

	Overall rating				
	1	2	3	4	5
# of data points	6366	6952	16140	39516	131026

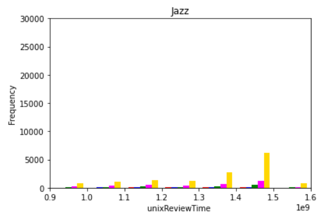
Analysis of Category and Review Time

The dependence of overall rating of a user-item pair on the category and review time is shown by analyzing the histogram of overall ratings for all categories. Fig. 1 (a)-(e) shows the abovementioned histogram for the 5 categories in the dataset. Two observations can be concluded from the results:

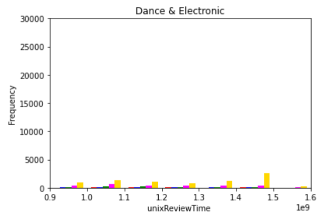
1. For a given category, the ratio of the number of reviews for a given overall rating changes over time. For example, in the pop category, the number of 5-star reviews is approximately twice the number of 4-star reviews at the beginning of the time frame, but this ration increases to approximately five times near the end. This indicates that the review time should be taken into consideration by the developed model.
2. The shape of the distributions for each category is different. In the pop category, the distribution has one peak, whereas it has two peaks in the alternative rock category. This indicates that the given category of the album should also be taken into consideration.



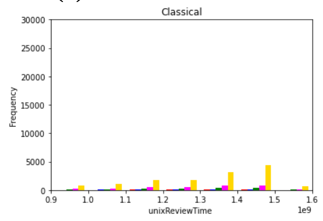
(a)



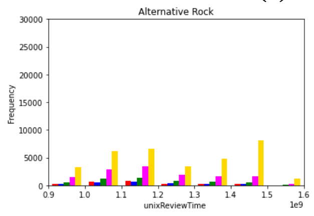
(b)



(c)



(d)

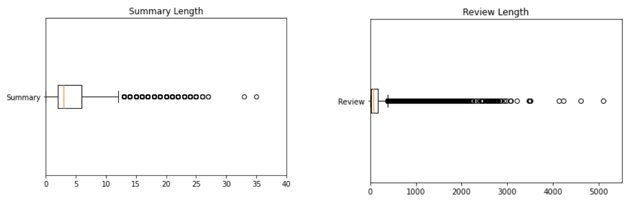


(e)

Fig. 1 – Histogram of overall ratings for each category

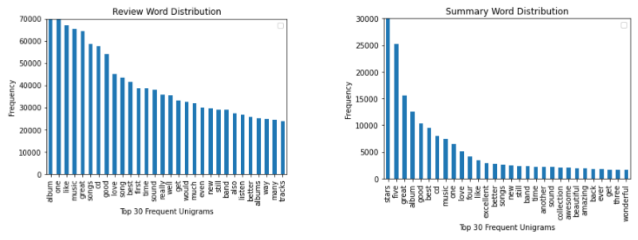
Analysis of Review and Summaries:

The distribution of the number of words in each of the two features was found. These are shown through Fig. 2 (a)-(b) for summary and review texts, respectively. Results indicate that summaries are at most approximately 35 words, and reviews are at most about 5000 words (stop words excluded). Furthermore, the summaries are composed of 32139 unique words, and the reviews are composed of 169519 unique words.



(a) (b)

Fig. 2 – Number of words



(a) *(b)*

Fig. 3 – Word histogram

The top 30 frequent unigrams of the review and summary text are shown in Fig 3 (a) and Fig. 3 (b), respectively. These unigrams were found by first preprocessing each review which included: turning all characters into lower case, tokenizing the sentence, discarding non alphanumeric strings, normalizing each string (i.e., lemmatization), and removing stop words. The results shown in Fig. 3 indicate that the top 30 words generally convey a positive sentiment in summary and review texts. These correspond to higher overall ratings and as such, a large vocabulary is required to observe words with negative sentiments. Bigrams were also considered, but were discarded as they further reduced the frequency of words with negative sentiments.

2. Models

Naïve Bayes

As a preliminary model, the naïve Bayes classifier was implemented using the review text, summary, and categories as features. This decision was taken because Naïve Bayes is suitable for categorical data and, by counting the number of times certain words appear in a sentence, it would be able to predict the overall rating of a user-item pair. Namely, this model assumes that the overall rating is dependent on the number of times certain words appear in a review/summary. In this regard, the review and summary were encoded through the bag of words (BoW) model based on a vocabulary of 200 words. The text preprocessing steps described above were used. The categories were also encoded as one-hot vectors, and no preprocessing was necessary. The complement naïve Bayes classifier was implemented as it is deemed to be most suitable for text data and unbalanced datasets. The classifier achieving a mean squared training error of 1.17, and a mean squared validation error of 1.2 on a 30% validation data split.

It is conjectured that the naïve Bayes classifier achieves poor results due to the sparsity of the encoded data. Namely, a large vocabulary (of the 200 most common words) was created for the summary and review data. However, the number of unique words found in the initial data analysis step far exceeded this number. As such, many of the input vectors were zero, and the classifier could not use them to make accurate predictions. The naïve Bayes model also assumes that the words are uncorrelated, which is not the case in text data. This problem is addressed through the next model, which aims to find relationships between words in the vocabulary, and allows the model to make better predictions.

One may also note that using the Naïve Bayes classifier inhibited the model from considering the review time as features as it is only limited to categorical features. A natural progression would have been to implemented decision trees to address this issue. This was avoided, because decision trees would also suffer from the sparse input vectors that were used to represent text for this model.

Neural Network #1

The next model that was implemented to address the sparse bag of words inputs (for review and summary text) was a neural network. The architecture of the proposed network is shown in Fig. 4 below. The inputs to the network consist of the review texts and summaries, in BoW format (preprocessed as above). The vocabulary for this tokenization process consisted of the top 200 most frequent words. This review and summary texts, in BoW format, are concatenated and fed into an embedding layer.

The purpose of this embedding layer is to address the sparse input problem. Namely, this embedding layer casts each word into a higher dimensional space, where words that have similar meanings are placed closer to each other. In this space, the words are not merely described as vectors that convey a certain sentiment. This allows the model to learn the meaning of commonly used words, and to use their meanings to infer the overall sentiment of the review. The output of the embedding layer is concatenated with the item categories (one-hot encoded), and the review times. Note that review times were normalized to 0-1 during the preprocessing step.

The combined inputs were then fed into a dense (*i.e.*, fully connected) layer, which then learns the sentiment of the inputs. In the implemented model, this layer consisted of 100 units, activated through the rectifier (*i.e.*, ReLU) function. Finally, the output of this dense layer is fed into a single perceptron (dense layer #2, Fig. 4), that combines all measurements and estimates the overall rating of the review through regression using mean squared error between the predicted and true overall scores as its optimization measure.

The proposed model achieved a mean squared validation error of 0.58, and a mean squared training error of 0.50 on a 30% validation data split. Although the proposed method exceeds the required error, it may fail to capture the true sentiment of the text data due to the BoW model. Namely, this model does not see the text data as a sequence (*i.e.*, time series) of inputs and as such, maybe inhibited from understanding the true meaning of the text. The final proposed model aims to address this issue.

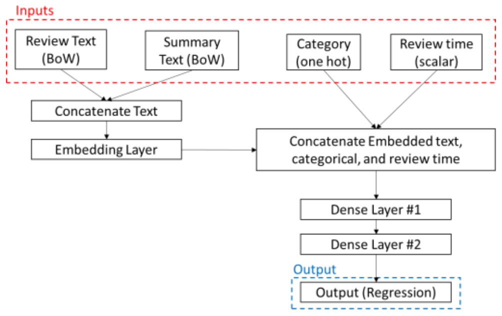


Fig. 4 - Neural Network #1

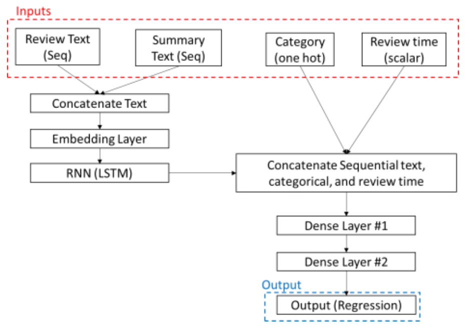


Fig. 5 – Final Model

Final Model

The architecture of the final (and submitted) model is shown in Fig. 5. The model looks very similar to the one in Fig. 4, except for:

1. The summary and review texts are embedded as a sequence, rather than through the BoW model. Namely, the sentences are encoded as a sequence of numbers, where each number represents the index of the associated word in the devised vocabulary. For example, the sentence "I love this course" would be represented as [1, 2, 3, 4], rather than as [1, 1, 1, 1]

through the BoW model. It is conjectured that this representation would enhance the model's ability to understand the sentiment of a review/summary. Note that in the implemented model, a vocabulary of 5000 words was used, and the maximum sequence length was limited to 100 words. Furthermore, stop words were not removed as it was found that doing so reduced the accuracy of the model, possibly because contextual information is being lost.

2. Once the summary and review texts are concatenated and embedded, the results are passed onto a recurrent neural network (a long short-short term memory (LSTM) was used in this implementation). This layer views the summary and review texts as a sequence, and estimates the overall score accordingly. By viewing the summary and review text as a sequence, the model is able to interpret the sentiment of a sentence more clearly and it can establish relationships between successive words.

As in the previous section, the output of the LSTM is concatenated with the one-hot encoded categories and normalized review times, and fed into two successive dense layers. The first dense layer uses the ReLU activation function, while the latter is equipped with a single neuron and a linear activation to achieve regression with mean squared error as the metric.

The proposed model was tuned with respect to three hyper parameters: the dimension of the embedding layer, the number of units in the LSTM layer, and the number of units in dense layer #1. The hyperparameter tuning was completed through a grid search, with **some** of the results (*i.e.*, minimum achieved validation error on 30% data split) shown in Table 2. The results illustrate that all hyperparameter sets achieved similar validation errors. As such, the one with the highest training error (*i.e.*, model 2) was chosen as the final model to reduce overfitting.

Table 2 – Hyper parameter optimization

Model	Embedding Layer (number of embedding dimensions)	LSTM layer (number of units)	Dense layer #1 (number of neurons)	Validation error	Training error
1	16	32	100	0.439	0.267
2	16	64	200	0.434	0.353
3	32	64	100	0.423	0.295

The training and validation error for the selected model is shown in Table 3 for each overall rating. The models achieved very good results for high overall ratings, but extremely poor results for lower ratings. This is because of the unbalanced dataset. To accommodate this, the dataset was balanced by down sampling the points with higher ratings, and up sampling the points with lower ratings. However, the overall model performed worse and was therefore discarded.

Table 3 – Training and validation error per rating

Overall	1	2	3	4	5
Train/validation error	2.84 / 3.72	1.58 / 1.92	0.72 / 0.85	0.29 / 0.34	0.13 / 0.18

Conclusion

The implemented models suggest that the best rating predictions can be obtained by extracting meaningful information from the text data. Thus, representation of sentences as sequences of words achieved the best results, especially when stop words were not used. One may note that the final model could have been improved with a more exhaustive hyperparameter tuning, which also considers the batch size and learning rate of the optimizing search engine. For future steps, it would be useful to consider using reviewer ID and item ID information to achieve more personalized results.