Pattern Product Review Analysis

Technical Report - Milestone 3

Team: Sehaj Chawla, Taro Spirig, Lotus Xia, Heather Liu

Outline

1. Problem Description

eCommerce is gaining dominance in the global retail market. Due to the lack of in-person interactions on virtual marketplaces, customer reviews become one of the most important channels for customers to communicate with each other and to provide feedback to retailers.

On the one hand, customers rely on reviews from past customers to have a peek at product quality and to make purchasing decisions accordingly. On the other hand, understanding customer reviews is also crucial to the success of online retailers. For instance, retailers can improve product quality or change marketing strategies based on customer preferences and suggestions. Retailers may also rely on review sentiment to predict sales volumes in the immediate future and to determine the quantity of inventories to acquire.

In this project, we work with Pattern, an eCommerce accelerator, to create a natural language processing (NLP) model based on customer reviews to predict product future performance. In particular, we are interested in conducting both short term predictions (e.g. from month to

month) as well as long term predictions (e.g. success of a newly launched product after one year). On the one hand, short term predictions can help retailers make sales projections and manage their inventories. On the other hand, long term predictions will inform the retailers if it is worth continuing a newly launched product or how they could change the product to boost its success.

We further separate each of the two prediction horizons into two subtasks:

1. We aim to accurately predict products' future sales performance based on customer reviews. In particular, we would like to gauge the effectiveness of using review texts, in addition to review metadata, in sales prediction.
2. Extract themes from review texts to gain insights on what keywords or topics are predictive of popular products

For subtask 1, we implement separate models using only review metadata and using only review texts. We compare the performance of these two classes of models to gain insights on the advantages of either class. We also construct an ensemble model of the two classes of models and evaluate the amount of improvement, if any, from using review texts in addition to review metadata.

For subtask 2, we hope to generate a dictionary of keywords or phrases that are predictive of sales performance that are intuitive to and are informative for online retailers in terms of future sales strategy.

2. Data

The BSR history dataset is a history of Best Seller Rank (BSR) for Amazon products within the Vitamins and Dietary Supplements category, ranging from July 2017 to July 2021. The data is scraped by a third-party service provider, Keepa. BSR is a performance evaluation calculated by Amazon using the product's current sales volume as well as the product's historical sales. Rank 1 in a category is the best selling product, 2 is the second best, etc. This dataset contains around 29 million rows and covers 9,991 unique products. The dataset's fields include ASIN (a unique identifier for the product), best seller rank, the average price of the product over the past 180 days, and the date of observation.

The review history dataset is a collection of Amazon reviews for products within the Amazon Vitamins and Dietary Supplements category that are on sale at the time of scraping.[1] The earliest review is in January 2004, and the last review in this dataset is in July 2021. There are around 5 million reviews on 9,977 unique products. The dataset's fields include ASIN, product name, review title, review rating, review date, review upvotes, review comment count, and a binary field indicating whether the purchase is verified (An "Amazon Verified Purchase" review means Amazon has verified that the person writing the review purchased the product at Amazon and didn't receive the product at a deep discount). The two datasets cover 9,958 products in common.

We note two main limitations of the datasets. Firstly, the time span is not equal across products. Keepa, the third party data provider, pays more attention to popular products on Amazon, scraping and updating their ranking information more often than unpopular products. Therefore, popular products are more likely to have frequent, continuous BSR information in our dataset than unpopular products, which tend to have long periods of missing BSR. As a

[1] Around fall 2021.

result, when we restrict to products that have frequent reviews and BSR records, the final sample may not be representative of the true population of products and reviews on Amazon. In addition, the dates at which we know the product ranking may not match the dates at which reviews were added for said product, sometimes leading to a small overlapping period for us to work with.

Secondly, the ordinal nature of the BSR hides some important and relevant information we may care about as an online retailer. For instance, the sales volume of rank 1 may differ drastically from the sales volume of rank 2, although they only differ by 1 in terms of ranking. In addition, Amazon does not reveal how exactly they calculate the BSRs. The calculation seems to depend most heavily on daily sales, with little weights on historical sales of the product. As a result, the ranks may suddenly spike or drop, adding noise to the BSR data.

To address the second limitation, Pattern further provides us with their estimated sales volumes corresponding to each BSR. The mapping is fitted based on actual sales volumes and ranks of their customer retailers, although the exact mapping function between BSR and sales volume is not revealed to us. A drawback is that the estimate for top products may not be accurate, since Pattern's customers rarely attain an extremely small rank (e.g. below 100). The estimates on these top products are essentially based purely on extrapolation of the fitted function.

3. Exploratory Data Analysis

3.1. BSR

Figure 1 plots the BSR over time for an example product (ASIN: B00005313T). The raw BSR data is oftentimes volatile. In particular, the median of BSR range among all products is over 170,000. The least volatile product has a difference of 474 between the highest and lowest ranking, whereas the most volatile product has a difference of 7,785,554.

We note three major causes of the volatility:

First, the product scope based on which the ranks are computed is larger than the scope of products we have in our datasets. In particular, our datasets consist of all products in Amazon's Vitamins and Supplements category, but the ranking is calculated based on all products in Amazon's Health and Household category, which is a much larger universe. As an example, the largest rank we see on a day may be on the scale of millions, although we only have data on approximately 9,000 products in our dataset.

Second, some products may go out of stock for a short period of time, during which the BSR of said product skyrockets.

Third, products may go on sale, leading to a higher sales volume, and, in turn, a lower BSR during the sale period. The main focus of this project is not to predict day to day spikes and drops that happen due to these external factors. It also appears unlikely for us to predict granular rank movement using just review data since the reviews may not immediately take effects on each rank movement. Therefore, we chose to take a bigger view of the rank to. We aggregate product ranks by taking the median rank per month for each product. We discuss more about our pre-processing procedures below.

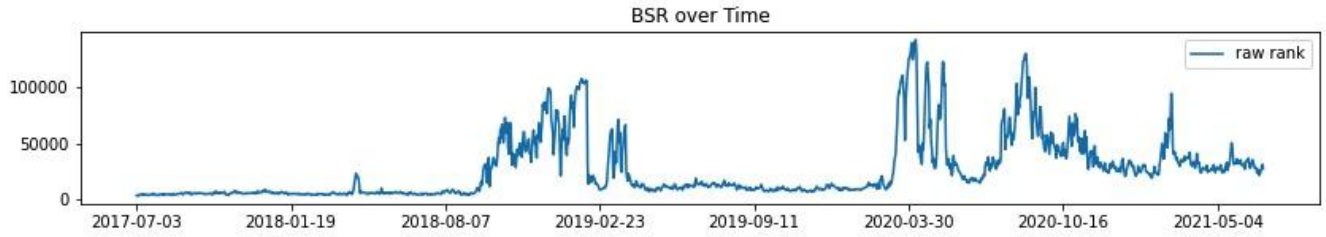

Figure 1: BSR over time for product B00005313T

3.2. Sales Volumes

Figure 2 plots the estimated sales volume corresponding to each BSR. This plot comes from the dataset Pattern used to derive the mapping function. We note that the sales volume decreases rapidly as BSR increases. Rank 1 product is estimated to have a daily sales volume of 5,214 units, whereas the sales volume of rank 7,000 is barely above 100 (not shown in the plot), and the sales volume of rank 201,934 dips below 1. The smallest sales volume in this dataset is at 0.05 units for a BSR of 454,302. As a comparison, the largest BSR observed in our dataset is on the order of 3 million, which is much larger than the largest BSR associated with an estimated sales volume. However, these products with extremely small ranks have essentially zero sales—we later describe a simple extrapolation method in “data processing” section.

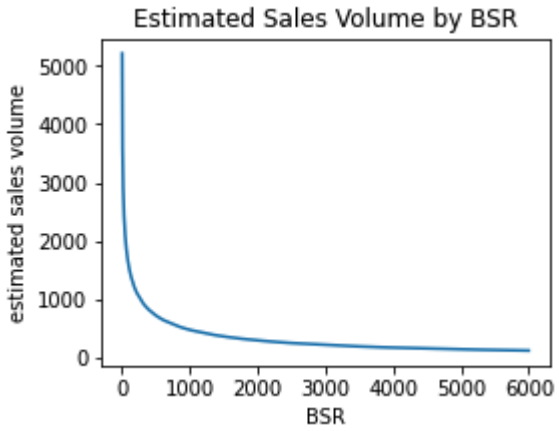

Figure 2: Estimated Sales Volume by BSR

3.3. Reviews

We aggregate the review data by product and month. The number of reviews per product-month varies quite drastically across products. The left pane of Figure 3 presents a histogram of the log number of reviews per month. The population is right skewed, with the most popular product attracting 1,392 reviews per month. Each review also has different lengths. The right pane of Figure 3 presents a histogram of the log number of words in each review. While approximately 28% of reviews have fewer than 10 words, the longest review contains 4,643 words.

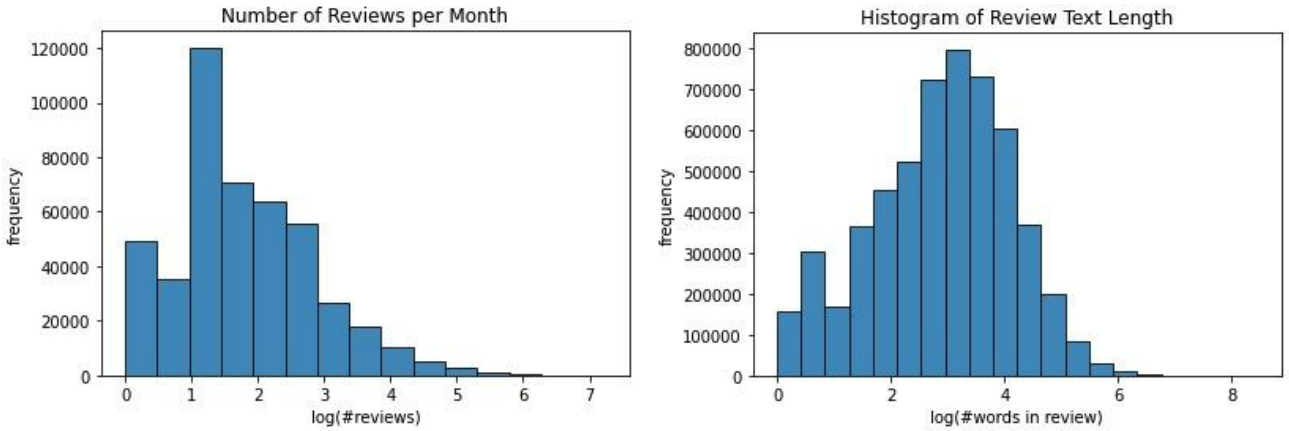

Figure 3: Histograms of the number of reviews and length of reviews

Among all review metadata, review rating is one of the most salient features that influence customer decision. Figure 4 presents some high-level statistics by review ratings. The vast majority of reviews are 5-star reviews. A potential explanation for this pattern is that popular products are more likely to elicit reviews, and these products are popular because they have good quality. In addition, reviews with 5-star ratings tend to be shorter than the reviews with fewer stars. 2-star reviews are the longest on average, which is intuitive—reviewers are more likely to use lengthy explanations to justify their low ratings.

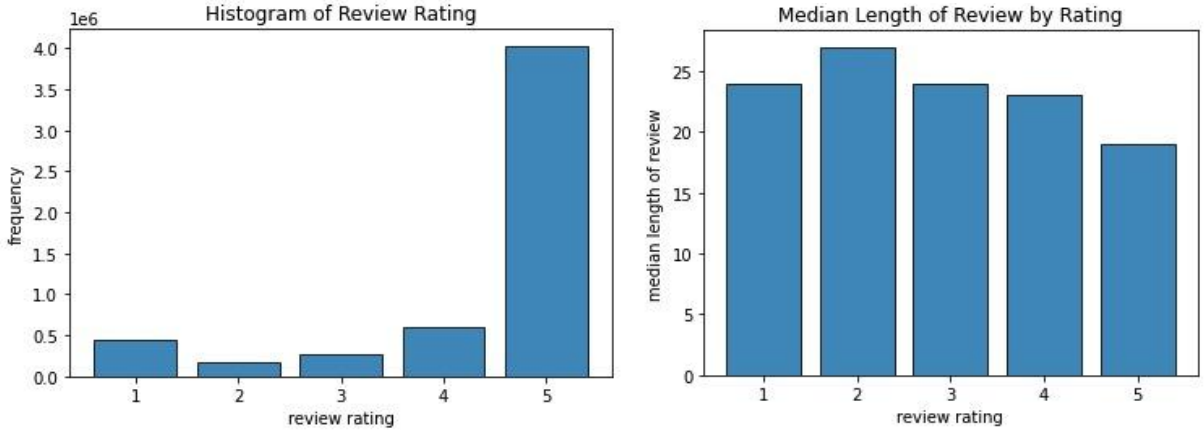

Figure 4: Frequency and review length by review rating

Lastly, we examine the number of upvotes corresponding to each review. Reviews with different numbers of upvotes are expected to have different impacts on customer decisions. In particular, the most voted review is displayed on top of the review feeds, so more customers may notice and trust this review. Figure 5 plots the distribution of the log number of upvotes per review. The distribution is again right-skewed. While over 60% of reviews have zero upvotes, the most liked review is voted by 16,368 customers.

Despite its indisputable impact on purchasing decisions, we would like to flag that the number of upvotes may be a form of data leakage. These upvotes are a snapshot as of the time of scraping, i.e. the number of upvotes represent the number of people who find the review useful between the time when the review is posted and when the reviews were scrapped.[2] We

[2] Roughly in January 2022.

do not know, and have no way of backtracking, the exact timing of the upvotes. As a result, directly using these upvote numbers in prediction essentially entails using future popularity of a review to predict past sales performance. For instance, using the number of upvotes as of January 2022 to predict the sales volumes in January 2021 is inappropriate, as the same review might be less popular in January 2021, and, in turn, has a lesser impact on consumer decisions.

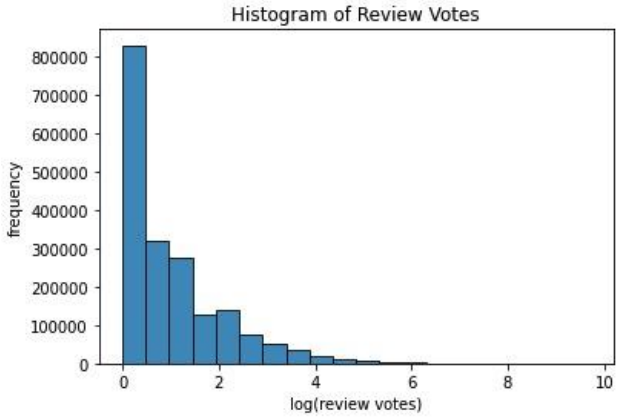

Figure 5: Histogram of the number of review upvotes

4. Data Processing

4.1 Short-term predictions

We take the following steps to pre-process the BSR:

1. We exclude products with more than 30 days of consecutive missing BSRs. For the remaining products, we fill in the missing values by taking the median of the BSR right before and after the missing range.
2. To reduce the noise in the data, we smooth the BSR data by calculating a 10-day rolling median. We refer to this rolling median BSR as “rolling BSR” for ease of reference in the rest of the report.
3. We then group the data by product-month and calculate the product’s median BSR for each month. We use the (lagged) monthly median BSR as the target variable in some of our models. We refer to this quantity as “*monthly median BSR*” throughout the report.
4. Lastly, we normalize the monthly BSR and the rolling BSR. In either case, we first compute its maximum and minimum values observed in the dataset, then we use the Max-Min method to scale down the value of BSR to between 0 and 1. In this way, the worst rolling (monthly) BSR is normalized to 1 and the best rolling (monthly) BSR is normalized to 0.
5. Finally, We reshape the data to yield a product-month level dataset.

Note that the rolling BSR has different sizes across months. For instance, there are 28 or 29 days in February, whereas there are 31 days in December. To obtain the same number of rolling BSRs across observations (recall each observation is a product-month), we drop the first rolling BSR in months with 31 days, and add the monthly mean of rolling BSR to the beginning in months with fewer than 30 days.

We take the following steps to pre-process the estimated sales data:

1. For ranks without an estimated sales volume (i.e. ranks larger than 454,302), we extrapolate the estimate using a line between 0.05 (which is the estimated sales volume for rank 454,302) and 0.
2. Then we merge the estimated sales data with the original BSR data using rank, to give each rank a corresponding estimated sales volume.
3. We then group the data by product-month and calculate the product’s median sales volume for each month. We use themonthly median sales volume as the target variable in some of our models. We refer to this quantity as “*monthly median sales*” throughout the report.

Figure 6 and 7 below plot the distribution of the two potential target variables, monthly median BSR and monthly median sales. We note that the distribution of monthly median sales is extremely right skewed—while the largest value is around 5,000, the vast majority of observations have monthly median sales below 100. The median observation has a monthly median sales volume around 40.

Lastly, we take the following steps to pre-process the review data:

1. We generate reviewvotes_num which is a numeric version of the number of upvotes for each review.
2. Then we group reviews by product-month. Texts/values in each product-month level are grouped in a list. We then merge the dataset with the processed BSR dataset to keep the

products in the intersection.

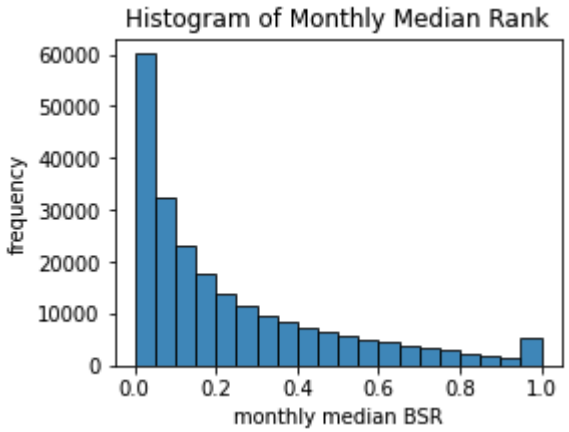

Figure 6: Histogram of monthly median rank

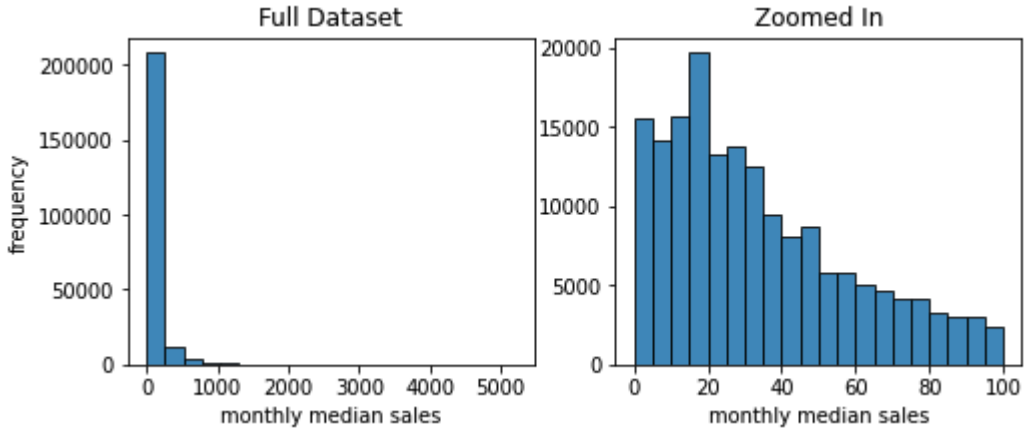

Figure 7: Histogram of monthly median sales volume

After we split the data, we have 2298 unique products containing 61199 observations in the training set and 851 unique products containing 21813 observations in the test set.

4.2 Long-term predictions

Based on the data processed in the previous steps, we take the following steps to generate new variables for the long-term prediction task:

1. We remove all products whose first review date is earlier than their first BSR date. We then define a product's launch date as its first BSR date.
2. Then, we calculate the optimal (minimum) BSR for a product over the subsequent one-year period after the first launching year. The time period of one year is chosen to minimize the effects due to seasonal changes (which affect the market of vitamins a lot).
3. We then use this optimal BSR to generate our new target variable. We define a product as successful if it has ever reached top 3000 in terms of Amazon's BSR during the one year period after the first launching year. For every product, we will predict if it will be classified as 1, a successful product, or 0, an unsuccessful product. We pick 3000 for two reasons. First, it is intuitively a good rank—being one of the top 3000 products in Amazon's Health and Household category (which contains millions of products) is an unambiguous achievement. Second, using this threshold, about 18% of products in our

datasets will be categorized as successful. The threshold is a nice balance between a real success and a balanced classification.

4. We then generate features by aggregating the reviews data and calculating some statistics on the BSR data for the first three months after a product's launch. By doing so, we are able to observe how the initial review and the initial BSR of a product after its launch affects its long-term BSR.

After we split the data, we have 2768 unique products in the training set and 923 unique products in the test set.

5. Literature Review

Recent studies have shown that predicting sales through online reviews is reliable. The study in [1] examines the effect of online reviews on new product sales from Amazon.com. The study also found that sales prediction models that include sentiment variables improve the ability to fit the data and the predictive power of the model. In [2], the authors use large-scale text mining to characterize the behavior of e-commerce consumers and model the relationship between product presentations and business outcomes. As described in [3], the authors establish a feature-level sentiment analysis using Amazon sales data and customer review data, to show that customers' preference for different features and texts can be used for predictive modeling of future changes in sales. In [4], the authors present a sentiment-aware model for predicting sales performance using blogs.

For sales prediction, some commonly used models are linear regression and tree-based models like XGBoost [5]. Regarding performance evaluation metrics, [6] used R2 and RMSE to evaluate the performance of regression models by comparing the predicted sales with the ground truth value.

For product review text analysis, Transformers and its variants are proved to be reliable, as described in [7]. Transformers, proposed in [8], are widely used for text encoding. A transformer is an encoder-decoder-based neural network that aims to solve sequence-to-sequence tasks while handling long-range dependencies with ease. Its main feature is the use of so-called self-attention (i.e., a mechanism that determines the importance of words to other words in a sentence or which words are more likely to appear together) to compute representations of sequential data, such as natural language text, without needing to process the data chronologically. BERT is a state-of-the-art language model proposed by researchers at Google AI language in [9]. It is a transformer-based technique for learning representations of language and is widely used for text representation as well. More specifically, it applies bidirectional training of Transformer to language modeling, which achieves a deeper sense of language context and flows than a single-direction Transformer model.

6. Short-term predictions

For both tasks presented in the introduction, i.e. short-term and long-term predictions, we have developed two different types of regression models

1. models based on review metadata, and
2. models based on review texts.

The idea behind this approach is to look at the predictive power of the text as an incremental

value-add on the model without any text. We do this by ensembling our two models. This helps us with the interpretability of the models (our second goal), while also giving valuable modeling insights into our prediction task.

6.1 Regression without Text

6.1.1 The models and the results

Since Milestone 1, we have completed all of our previously mentioned goals for the regression models without text. We have also changed several aspects of our models in terms of features and target variables considered, that we present below.

Our regression models without text consist of linear regression, XGBoost, and random forest models to predict the future estimated sales of a product based on most of the data available apart from the text of the reviews and their titles. For every product, we predict the next monthly median sales using the following features:

1. The rolling BSR (this constitutes 30 values).
2. The monthly median BSR.
3. The monthly mean review ratings.
4. The monthly mean sales.
5. The monthly median sales.
6. The monthly mean price.
7. The monthly median price.
8. The monthly mean review ratings, weighted by the truth function associated with the verification of the reviews, i.e. weight a review by one if it is verified and weight by zero if it is not verified.
9. The average over all the mean monthly review ratings from previous months, i.e. the average over the values of point 3 calculated for every month.
10. The average over all the means of the product's review ratings over the previous months, weighted by the verification truth functions, i.e. the average over the values of point 8 calculated for every month.
11. The cumulative number of reviews over the previous months.

In total this represents 40 features. The last three features are cumulative data which give the model information about the product's past performance, instead of having only the information about the current month. This is important as customers have access to all of the reviews which were ever posted for that particular product. In particular, the overall rating visible to the customer is computed based on all of the reviews for that product.

In comparison to the previous models developed for Milestone 1, we have added the features related to sales volume, as we have decided to change the target variable from change in rank to monthly median sales. This decision was made because a good product which stays good or a bad product which stays bad would both show very little change in rank. It is thus hard for the models to differentiate between those two cases based on review data. In other words, the reviews for a constantly bad product or constantly good product are very different but both have very little change in rank. We have also added the information on the price which could greatly influence the decision of a customer to buy a given product. The cumulative number of reviews was also added as a feature since Pattern was interested in understanding its predictive power. Finally, the features related to upvotes (or the count of people who found a review useful) were dropped as they were a form of data leakage. Indeed, as the numbers of upvotes were only recorded when the data set was created, they were not informative of the importance of a

review at a given month in the past. It could have been, for example, that a review got almost no upvotes at a given month and received a lot of upvotes only later just before the data was collected.

We have then inputted these features into linear regression, XGBoost, and random forest models imported from scikit-Learn. These models were then trained on a third of the products and tested on another third of the products[3] using the computing resources of AWS. After hyperparameter tuning, the resulting performances were found to be:

Models	Hyperparameter	R-squared score
Linear regression	———	0.938
Xgboost	learning rate = 0.05 n estimators = 100	0.946
Random forest	max depth = leaves pure[4] n estimators = 500	0.939

The R-squared scores are extremely high for these very simple models. It was first hypothesized that this was due to some problem in our data. Indeed, we thought that the estimated sales volume of most of the products we were feeding the model and/or testing on were very close to zero. This would result in a trivial prediction for most of our test data-points, which would result in a high score. However, after inspection of our data, it became clear that this was not the case as can be seen in the histograms of the distribution of the data in figure 7.

The reason for the very high scores became clearer with the analysis of our models which we present below.

6.1.2 Analysis of the models

We first explored the importance of our features using permutation feature importance. The following histograms present the importance of the five most relevant features for both the Xgboost and the random forest models respectively.

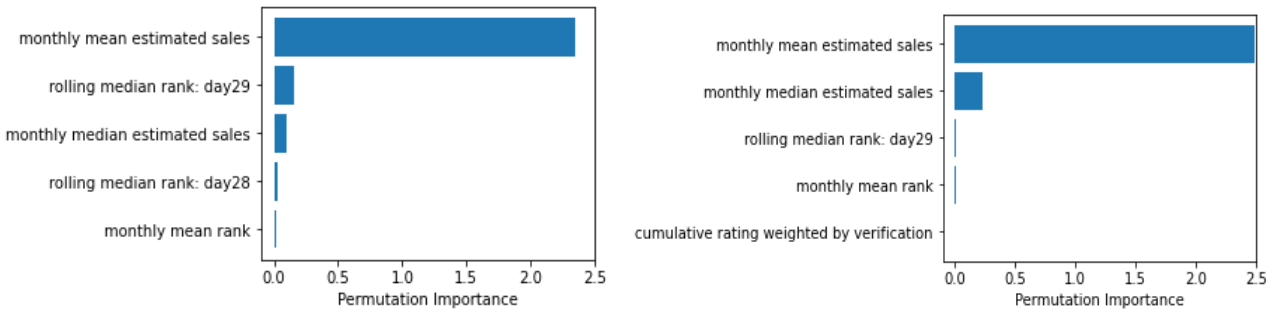

Figure 8: Histograms of feature importance (left: Xgboost, right: random forest)

[3] We have decided to restrict ourselves to a third of the products both for training such that we could train the models more rapidly. This is especially important for the text-based models which take much longer to train and as we wanted to compare performances with these simpler models, we had to train and test them on the same data.
[4] This is the default setting, i.e. if max depth is None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

We see that both our models are strongly dependent on a single feature, namely the monthly mean sales. It is thus not so surprising that the models were having very high R^2 scores. Indeed, it is intuitively clear and expected that the models will be very good at predicting the next month's sales volume given the current month's sales volume as the predictions are based on the principle of momentum, i.e. a product which is selling well this month will probably continue to sell well in the near future. It is, however, unclear why the models are both basing the next month's median sales prediction on the current month's mean sales. One would expect that the models would use the current month's median sales. This is probably due to the fact that the monthly median sales and monthly mean sales are highly correlated and the models picked up the monthly mean sales instead of the monthly median sales as means contain more information about the data than medians.

We have then considered partial dependence plots to analyze the dependency of the target variable on certain features. We discovered that the most relevant feature, i.e. the monthly mean sales, had a linear relationship with the target variable for both models, as can be seen in the following plots.

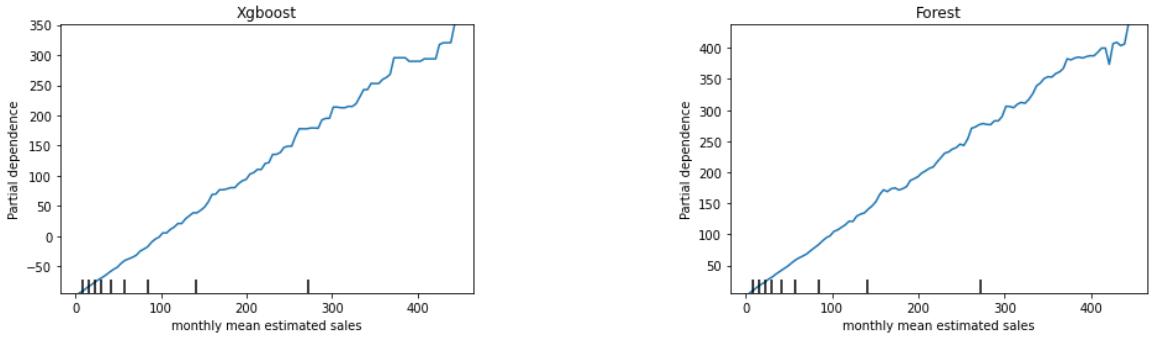

Figure 9: Partial dependence plots for the most relevant feature for both models

This linear relationship supports the intuition that predictions are based on momentum. We also computed the partial dependence of the second and third most important features of both models. The second and third most important feature for the Xgboost model and the random forest model respectively is the rolling median rank on the 29th day of the month. The resulting partial dependence plots also have the expected behavior as a small rank leads to high predicted monthly median sales for the following month (see figure 10). The third and second most important feature for the Xgboost model and the random forest model respectively is the monthly mean rank. In this case as well we observe the same expected behavior, i.e. a small rank leads to a high median estimated sales volume predicted[5]. We note that both of these features are correlated with the monthly mean estimated sales as we have a one-to-one function translating estimated sales volume to BSR and vice-versa.

[5] As the partial dependence plots for this feature are very similar to the rolling median rank on the 29th day of the month we omit the figure and refer the reader to figure 10 as a reference.

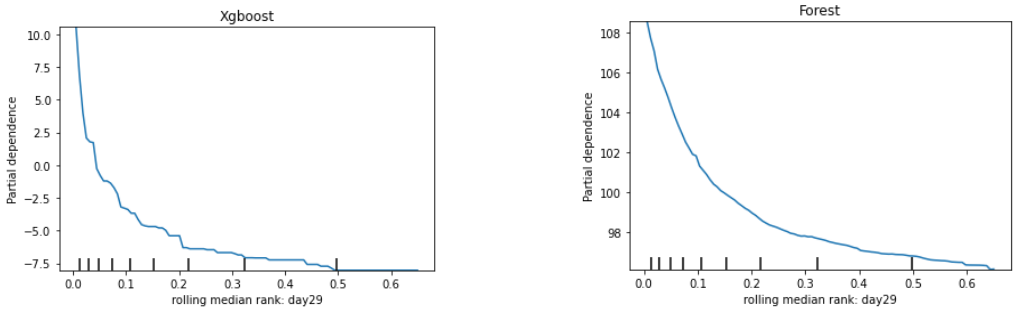

Figure 10: Partial dependence plots for the rolling median rank on the 29th day of the month.

We also considered the partial dependence plots of other features of interest such as the cumulative number of reviews, but they are hard to interpret as they are not as relevant for these models.

6.2 Regression with Text

Compared to Milestone 1, we have now added a set of bag of words based models, and have made multiple iterations on the transformer models.

6.2.1 Bag of Words Models with Linear Regression

We use a bag of words model as the benchmark for prediction using only reviews text. Such a benchmark model has two major merits. First, it is easily trainable and provides a baseline performance that can be compared with a more complex transformer based model. Second, a bag of words model is highly interpretable. For instance, we can simply look at the phrases that are associated with positive or negative coefficients in a linear regression, and gain some insights on what topics are associated with high or low sales performance.

We use the same training set and testing set as the non-text model. Recall that the target variable is the monthly median sales of the next month, identical to that in the non-text model. The predictors are the (weighted) frequency of the 500 most common phrases in the training corpus. For text processing, we experiment with the bag of words model, where we simply count the occurence of each of these 500 phrases, as well as the term frequency-inverse document frequency (TF-IDF) model, where the weighted frequency of each phrase is calculated from the term frequency and inverse document frequency. The TF-IDF model has the advantage of adjusting for the fact that some words are used more commonly in general. We then use a regularized linear regression model to predict the monthly median sales of the following month.

The set of hyper-parameters we experiment with includes: simple bag of words model vs. TF-IDF model, bigrams vs. trigrams, LASSO vs. ridge regression, and the different penalty strengths. Figure 11 tabulates the R^2 score on a hold-out validation set across different model/hyper-parameter combinations. With the bag of words model, the R^2 score is sensitive to changes in hyper-parameters; however, the R^2 score is more stable with the TF-IDF model. The best performance is attained with a bigram TF-IDF model with a LASSO regression using a penalty strength of 0.1. The R^2 score is quite high, especially given that we are only using a simple linear model on a very sparse model matrix.

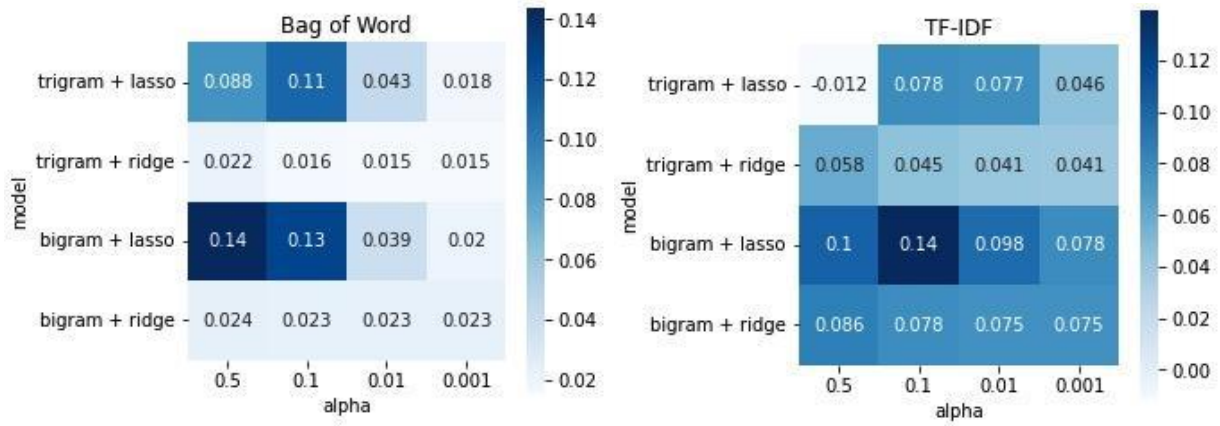

Figure 11: R^2 score across different model/hyper-parameter combinations

To better understand the model predictions, we show the binscatter plot of the target value against the model prediction in Figure 12. The binscatter plot is generated as follows: we divide all observations into 30 bins based on their model prediction values. Each bin has the same number of observations. Then we plot the average true target value against the average prediction value within each bin. Intuitively, the closer the points are to the 45 degree line, the better the prediction is compared to the real values of the target.

We see that the predictions below 200 are unbiased—the predictions match the corresponding target values closely. However, the predictions above 200 are biased downwards—the average true value is around 400 when the model predicts an average of less than 300. This pattern is not surprising given that a linear model can hardly capture the nonlinear pattern of the monthly median sales of the very successful products.

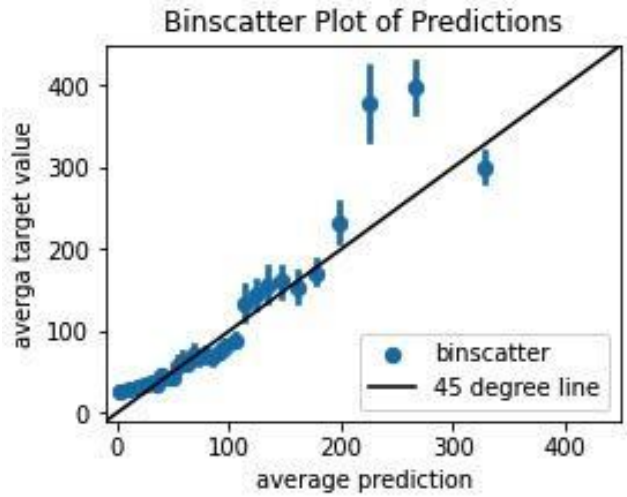

Figure 12: Binscatter Plot of Target Variable vs. BoW Predictions

Focusing on the best performing model, we further examine the bigrams that are the most predictive of sales volumes to gain some insights on influential keywords and topics.

The 10 bigrams associated with the largest positive coefficients are as follows: *cider vinegar, garden life, bowel movements, flu season, hair nails, taking probiotic, 2nd bottle, taste great, getting sick,* and *stopped taking*. The only 2 bigrams associated with negative coefficients are

joint pain and *pain relief*.[6]

Many of these bigrams are quite intuitive although some require more context that is not captured by a bigram model. We make two observations. First, "garden of life" is a famous brand name of vitamin products, and is one of the best sellers on Amazon. The model precisely picks up the brand[7] from the review texts. Second, "2nd bottle" is predictive of high sales performance. There may be two ways through which this keyword is associated with high sales volumes. First, it is the consequence of a repeated purchase, and therefore represents the increase of the intensive margin—the product is retaining existing buyers. Second, it increases the extensive margin—new customers may be convinced by such a genuine review and start to purchase this product.

6.2.2 Transformer Models

Since Milestone 1, we have added many iterations to our transformer model. This section will be structured by listing out each version of the model as a separate section, and then discussing their results in the final section.

6.2.2.1 Transformer Model Version 2

There were 3 major changes to the model described in Milestone 1, that led to this version of the model:

1. We first had to get rid of the upvotes as a feature, which were previously playing a central role in the model - both as input and as a weighting factor to create the embedding of the aggregate review information. This was required because, after discussions with Pattern, we realized that upvotes were a form of data leakage - i.e. the data was collected at the end of the timeline for each review, and so we didn't know the upvote values for different points in the timeline.
2. We wanted to use a simpler transformer model - specifically move from large BERT to tiny BERT. This was because the gain in performance for large BERT over tiny BERT wasn't very significant, and the training time for large BERT made prototyping of the model close to impossible because of the long training time.
3. We also trained our model for a significantly larger number of train steps (going from 200 steps previously to 50,000 now). We were able to make this change because we now had access to AWS, and could run experiments remotely. This change is very significant, but even then only regresses on a training point once.

A diagram for the new model is shown in Figure 13 below. It is important to note that the aggregate review embedding represented here is now no longer weighted by upvotes, but is instead a simple mean of all the review embeddings.

6.2.2.2 Transformer Model Version 3

After having trained the previous model, we realized we needed two more changes to the above described structure:

1. We had to get rid of the previous BSR input because as we saw from the non-text

[6] Among the 500 bigrams, 400 have coefficients equal to 0 in the LASSO regression. The intercept is approximately -26.

[7] In pre-processing, we take out all stopping words in the English dictionary (e.g. an, the, of). Therefore, "garden of life" is pre-processed to be "garden life."

analysis, any information about product revenue momentum was extremely predictive of the target, and as such we knew that our model was only going to consider that information if we passed it into the model. Instead, we needed a way to create a fair comparison between our transformer model and the bag of words model described above. We also wanted to answer the question - how much of the variance in the target can be explained by review text alone?

2. We also wanted to change the target variable from the BSR change to the next month's sales volume. This, again, was done to make the transformer models comparable to the other models we had.

The final structure of this model is described in Figure 14.

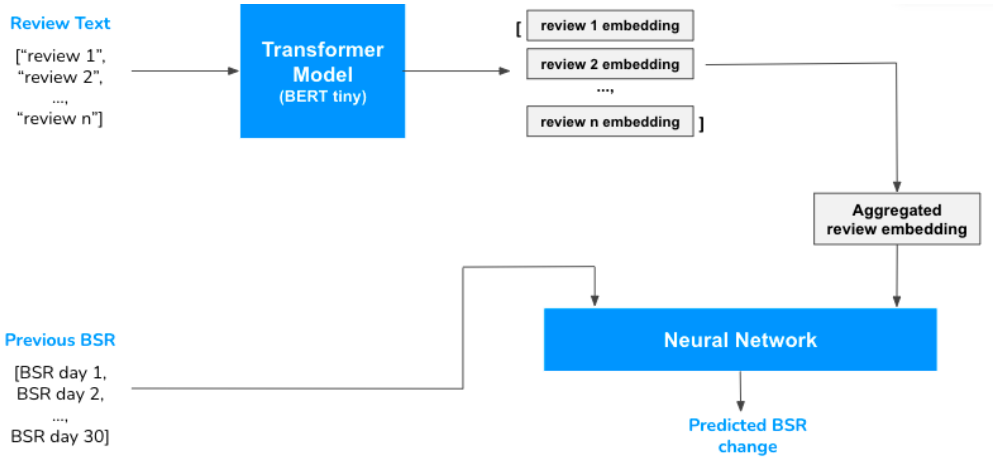

Figure 13: Transformer Model version 2

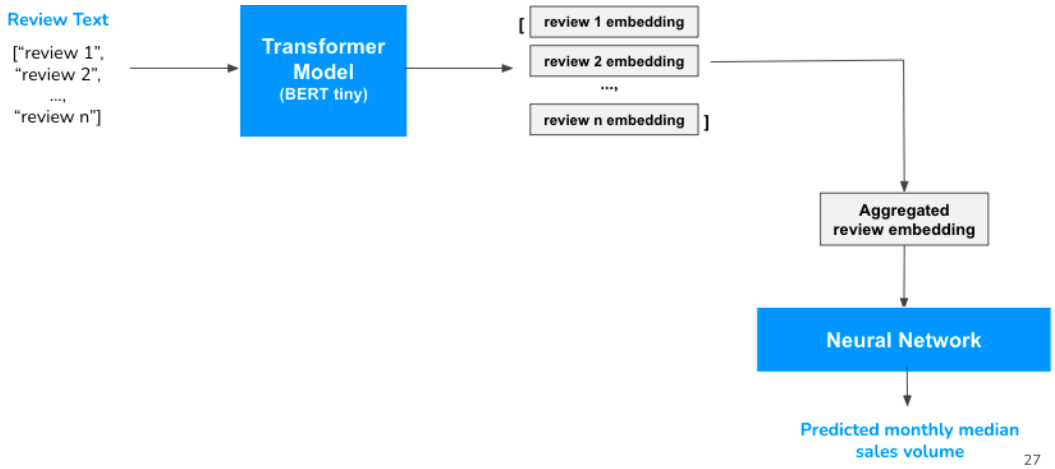

Figure 14: Transformer Model version 3

6.2.2.3 Transformer Model Results

The results (R^2 score) for the models is shown in the table below. Please note that the test set used for this table is exactly the same test set that is used for the other models in our report (namely the non text model and the bag of words model).

Transformer Model Version	Total Train Steps	R^2 Score

Version 1 (Milestone 1)	200	-0.132
Version 2	50,000	0.256
Version 3	50,000	0.164

6.2.2.4 Transformer Model Discussions

In this section, we give reasons why we believe the above R^2 scores were seen for each of the models described.

For the first model, from Milestone 1, the reason for the bad R^2 score is clear - the model was severely underfitting the data because it had only seen about 0.4% of the training data, which, of course, doesn't even begin to capture the variance of the target variable. However, the results from this model were included here to illustrate a point about how important the training steps are in improving the model performance. This might seem an obvious point to make, but it is one of the major points we will make for our discussion of model version 3.

For the version 2 model, which was the best performer out of these transformer models, we see the immediate impact of adding the train steps into the training process - even though version 2 lacked an important feature that version 1 had access to (review upvotes), it was able to capture up to 25% of the variance of the target variable. The reason for this high score is potentially because of the inclusion of the previous BSR values in the model. As we saw in the non-text analysis, when the time window is as short as 1 month, momentum information is the most important feature for predicting the next month's target. This leads us to believe that this transformer model structure was not getting much information gain out of the reviews themselves, but was instead using the previous rank values almost entirely to make the prediction.

For version 3 of the model, there is a drop in the score compared to version 2, but this is not a fair comparison to make for two very significant reasons: (1) they are both predicting on different targets, and (2) we have dropped the most important feature for version 2 from our version 3 model. However, we can make a fair comparison of this model with the bag of words models described previously. Specifically, the best performing bag of words model had an R^2 score of 0.138, while the transformer model has a score of 0.164. This, at first, did not seem like a significant improvement to us, but after discussing our results with Pattern, they mentioned that the results were what they expected to begin with. This is because the increase is a 10% increase in performance, and that is what is expected for most regression based tasks that do not have a very significant context based component compared to the vocabulary based component.

Nevertheless, we believe there are two ways this model can be improved, both of which are from the assumption that the model is currently underfitting the train data. The first of these involves looking at the training steps. Although the model was trained for 50,000 steps, this is not a large number compared to the training data (~60,000 rows of data), and it means the model weights have regressed on each training point just once during the training cycle - multiple such updates on each training point will allow the weights to be shaped even better for the train fit. We plan to increase this value to at least 100,000 steps. This conclusion of letting the model train more is also backed by evidence from the 5 train epochs themselves, which show the following trend in R^2 score: -0.213, -0.003, 0.018, 0.115, 0.139. It is worth noting that a single training epoch in this case includes just 10,000 training steps, and so is not

all-encompassing for the training dataset. It is clear from this trend that the R^2 score has only been increasing, and stopping at 5 epochs was a form of regularization because it was early stopping, and there was more potential for the R^2 to increase. The big challenge here is that the model currently takes 3 days to train, which leads to much longer prototyping cycles.

The second improvement we can make is related to the model structure- specifically, the number of trainable parameters. Currently, our model uses vector representations for reviews that are 40 elements in dimension. So, when we compare this to the 500 element vector that the bag of words model is making a prediction based on, we see a big difference in the amount of information the model has to make a prediction. To improve this, we can simply increase the size of our review descriptors, while also adding more dense layers in our neural networks, so that we can better capture the non-linear trends that we are likely currently missing. Moreover, we will also be looking into unfreezing some more transformer layers (rather than just the last layer), in order to make the model even better equipped to handle contextual information.

In Figure 15 below, we see a binscater plot for our version 3 model. This plot helps us make more sense of the model predictions. To understand this plot, recall that the closer the points are to the 45 degree line, the better the prediction is compared to the real values of the target. On plotting this, and comparing it to the binscatter for the bag of words model, we notice that the transformer model is better able to capture the trends for the tails of the distribution. i.e. it is doing a better job at predicting things for very successful products compared to bag of words. This is what we would expect, since BERT along with the neural network structure has the additional advantages of non-linearity. It is, however, not doing as good a job for the linear trend closer to the 50-100 mark, and this can potentially be explained by the smaller number of train steps this model has had to capture that trend successfully.

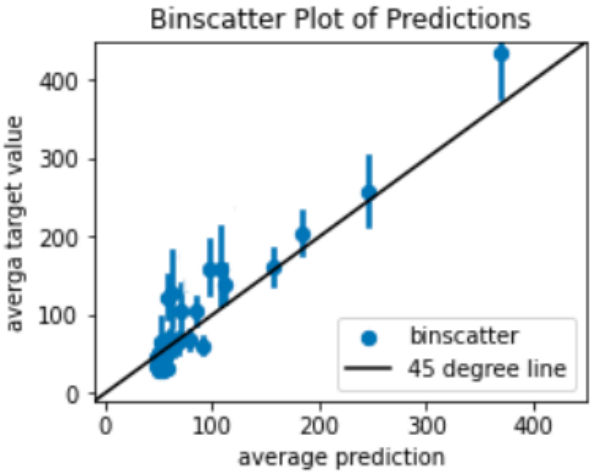

Figure 15: Binscatter Plot for BERT

7 Long-term predictions

For long-term predictions, we use confusion matrices, ROC and AUC to evaluate and tune the models, and use F1-score as the main evaluation criterion to compare the performance of different models. We can read precision and recall from the confusion matrix, where precision is the proportion of correctly identified success cases, and recall is the proportion of actual success cases that are correctly identified. We care about the precision of our models because

we want our models to make accurate predictions about the future performance of every product. We also care about the recall of our models because we do not want our models to easily overestimate the future performance of a product and thus give a false signal to our customers. To get the best precision and recall at the same time, we use F1-score, which is the harmonic mean of precision and recall values for a classification problem.

The above-mentioned metrics change with the changing threshold values in the binary classification problem. Thus, we generate AUC-ROC curves to easily visualize which threshold is giving us a better result. ROC curve is an evaluation metric for binary classification problems. It is a probability curve that plots the true positive rate against the false positive rate at various threshold values. The AUC is the measure of the ability of a classifier to distinguish between classes. The higher the AUC, the better the performance of the model at distinguishing between the positive and negative classes.

7.1 Classification without Text

7.1.1 The models and the results

Our classification models without text consist of logistic regression, XGBoost, and random forest models predict if a product will ever be successful during the year that follows the first three months after the product's launch based on most of the data available apart from the text of the reviews and their titles. For every product, we predict if it will be classified as 1, a successful product, or 0, an unsuccessful product, based on the following features, each calculated over the first three months after the product's launch if not specified:

1. Number of reviews
2. The mean review rating
3. The mean review rating weighted by the truth function associated with the verification of the reviews, i.e. weight a review by one if it is verified and weight by zero if it is not verified.
4. Number of verified reviews
5. Mean rank over the first month
6. Mean rank over the second month
7. Mean rank over the third month
8. Median rank over the first month
9. Median rank over the second month
10. Median rank over the third month
11. Minimum rank over the first month
12. Minimum rank over the second month
13. Minimum rank over the third month
14. Mean rank
15. Median rank
16. Minimum rank

In total this represents 16 features. The first four features are purely based on the non-text review data. These are summarizing the information that is available to customers during the first three months after the launch of the product. The last 12 features are encoding the rank information.

We input these features into logistic regression, XGBoost, and random forest models imported from scikit-Learn. We then train the models on the train set and we tune the hyperparameters of the models based on their F1-scores on the validation set. After hyperparameter tuning, we find

the following performances on the test data:

Model	Best hyperparameter	F1-score
Logistic Regression	————————	0.0121
Xgboost	learning_rate=0.2 n_estim=100	0.371
Random Forest	max_depth=5 n_estim=200	0.338

The F1-scores are comparable to our best text-based models (see below). In comparison to the short-term non-text models which were almost entirely making predictions based on features related to previous rank, we were hoping that the long-term classification would depend more on the review data as the long-term prediction is intuitively less correlated with momentum. We thus have tested the performances of the models when inputting only the first four features listed above, i.e. only the review based features.

Model	Best hyperparameter	F1-score
Logistic Regression	————————	0.0345
Xgboost	learning_rate=0.2 n_estim=500	0.318
Random Forest	max_depth=leaves pure[8] n_estim=500	0.245

The performances of these modes are surprisingly good, which means that the review data really has some significant information encoded in them.

7.1.2 Analysis of the models

We first explore the importance of our features using permutation feature importance. The following histograms present the importance of the five most relevant features for both the Xgboost and the random forest models respectively.

[8] This is the default setting, i.e. if max depth is None, then nodes are expanded until all leaves are pure or until all leaves contain less than min_samples_split samples.

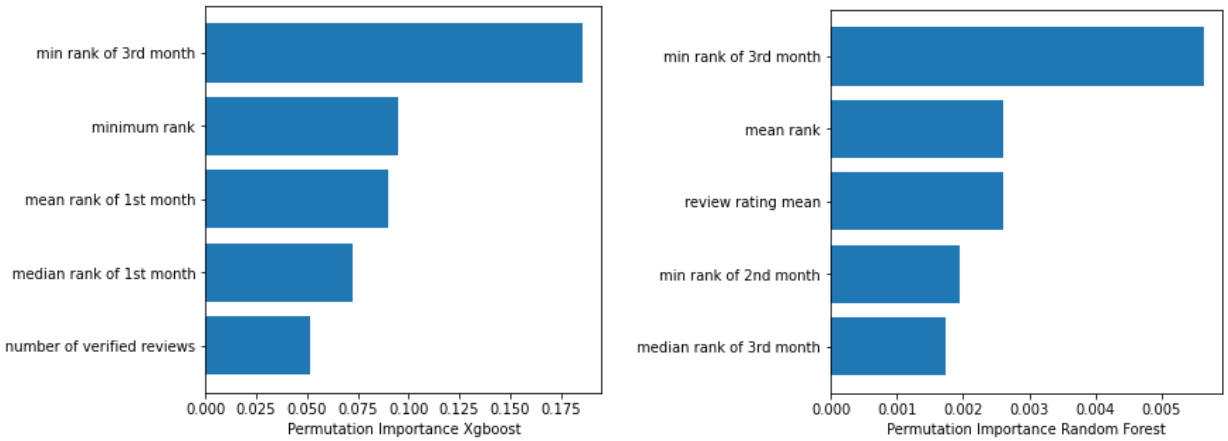

Figure 16: Histograms of feature importance (left: Xgboost, right: random forest)

We see that the long-term predictions of the non-text classifiers depend heavily on several features compared to the feature importance plots of the short-term prediction non-text models which were almost entirely relying on one feature. It is also interesting that as expected the long-term prediction depends not only on the rank features but also on the review data. The fifth most important feature for the Xgboost model is the number of verified reviews and the third most important feature for the random forest model is the mean review rating over the first three months after the launch of a product. This is particularly interesting as the way these features are used by the models can give us some interesting insights into review data.

We therefore consider partial dependence plots which give us an insight into how the target variable depends on certain features. Looking in particular at the review related features, we first plot the partial dependence plots of the number of verified reviews.

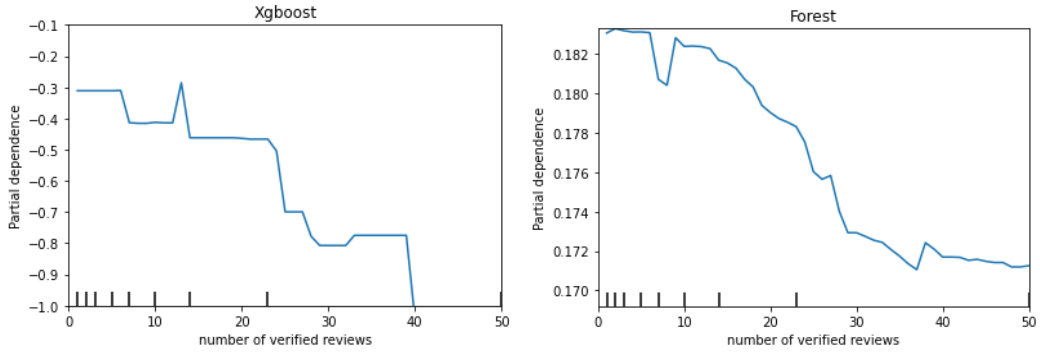

Figure 17: Partial dependence plots for the number of verified reviews

Both models are showing a downward trend at least in the region of zero to approximately forty verified reviews[9]. Although counterintuitive at first, as more reviews should normally mean that a product is successful, we believe that this trend is actually representative of the reality behind the data. Indeed, it is known that a lot of businesses create fake verified reviews by buying their own products and rating them well, especially right after launching their products to boost their products' popularity. The models are thus probably picking up on that. It thus means that the more verified reviews the more probable the reviews are fake which leads to a

[9] This is also the region where the plots should be taken most seriously as it is the region where the black bars on the abscissa are located and these black bars represent the distribution of the data

lower probability of success of the product. It is also important to note that the plots for the number of reviews (aggregate of verified and not verified) show the opposite trend, i.e. as expected more reviews lead to better chances of success.

We also consider the partial dependence plots for the mean of the review ratings over the first three months after the launch of the product. This feature is the third most important feature for the random forest model but it is the second to last feature for the xgboost model. We thus only show the partial dependence plot for the random forest model, as the plot for the xgboost model does not give any insight.

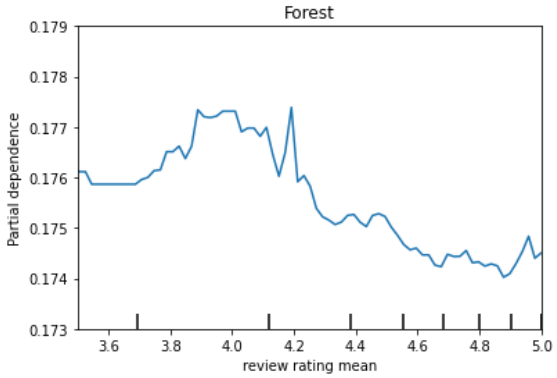

Figure 18: Partial dependence plots for the mean of review rating

We observe a similar counterintuitive trend for this feature as well, i.e. a lower mean rating gives a higher probability of success[10]. This is probably also due to the fake reviews produced by the businesses. The small bump at around 4 is probably much more representative of the distribution of real good reviews and thus leads to a higher probability of success than say a mean rating of five which is almost certainly fake.

7.2 Regression with Text

7.2.1 Bag of Words Models with Linear Regression

We again use a bag of words model as the benchmark for prediction using only reviews text.

The target variable in this case is also the dummy variable indicating whether the product is successful after 1 year of launch, identical to that in the non-text model. The predictors are computed in the same way as in the short-term prediction task, but based on only reviews within three months of product launch. Recall, they are the (weighted) frequency of the 500 most common phrases in the training corpus. Again, we experiment with the bag of words model, as well as the TF-IDF model. The set of hyper-parameters we experiment with includes: unigram vs. bigrams vs. trigrams, l1 vs. l2 penalty, and the different penalty strengths. Figure 19 tabulates the F1-score on a hold-out validation set across different model/hyper-parameter combinations.

[10] As mentioned in the previous footnote, the trend below 3.75 should not be taken too seriously as the data is mostly distributed above that value as indicated by the black bars on the abscissa.

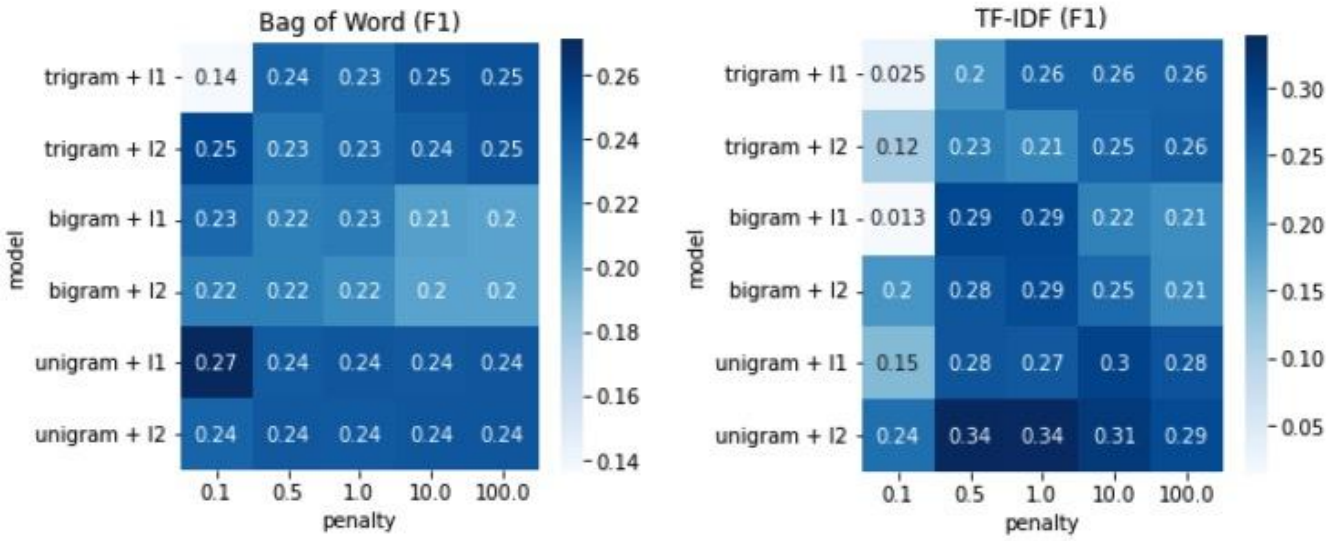

Figure 19: R^2 score across different model/hyper-parameter combinations

We make two major observations. First, trigram models overall perform the worst. This is potentially due to lack of variation in trigram features. Note that we are using only the first three months of reviews for long term predictions, which results in a small training corpus. In particular, there're on average approximately 5,300 words worth of reviews in the first three months for each product. The 500 most common trigrams collected from the training corpus usually have low frequency. Indeed, the median frequency of the 500 most common trigrams is 3.

Second, unigram models overall perform the best. One potential reason is that the variation in frequency of unigrams is much higher, making the model capable of explaining the variation in target variables. Specifically, the median frequency among the 500 most common unigrams is 22. The highest frequency we observed is 290, and the minimum frequency among the 500 unigrams is 5. Another potential reason is that simple word counts can already capture a lot of meaning or predictive power of the reviews.

The highest F1-score is achieved using a TF-IDF model with unigram and L2penalty at a strength of 0.5. Focusing on this best performing model, we again examine the unigrams that are the most predictive of sales volumes to gain some insights on influential keywords and topics. This time, however, the set of words are less suggestive.

The 10 unigrams associated with the largest positive coefficients are as follows: *gummy, gummies, fiber, week, keto, instead, morning, acv, b12,* and, *buying*.

The 10 unigrams associated with negative coefficients are *products, little, relief, digestion, levels, flavor, went, symptoms, helping,* and, *took*.

However, we note that unigrams associated with large positive coefficients are more likely to be product ingredients. For example, among the 50 most positive unigrams, we get the following ingredients: acv (apple cider vinegar), fish, oil, turmeric, coffee, and enzymes. We see no ingredient unigrams among the 50 most negative unigrams. This provides some insight on what reviews are associated with successful products. In particular, subjective feeling of the product appears to be less important than objective description of the ingredients. This is quite intuitive, as customers might be looking for certain effective ingredients in their desired vitamin products, whereas flavor and texture are secondary.

7.2.2 Transformer Model

Like the models described in the previous sections, we also train our latest version of the transformer model on this long term prediction task to try and predict whether a product will be successful or not after 1 year, given only the review texts from the first 3 months of the product.

The transformer model architecture that we used is shown in the diagram below:

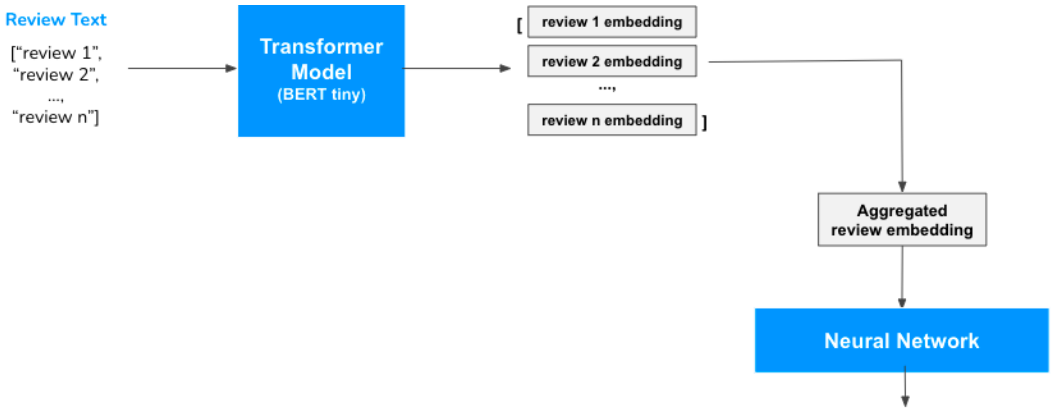

Figure 20: Transformer Model Structure

As a quick recap: we first pass each review through the transformer model, then we aggregate the review embeddings to create a single embedding for the model that is then passed through a neural network to make the final probability prediction of success.

While for the short term task we saw that the transformer model showed some improvement over a bag of words model, we also saw that any prediction made using text data was overshadowed by the predictions made using momentum data. This was perhaps expected, given we were only predicting for next month's performance given the current month. This trend changed significantly once we started predicting for longer term data.

The results for the BERT model are shown in the figure below:

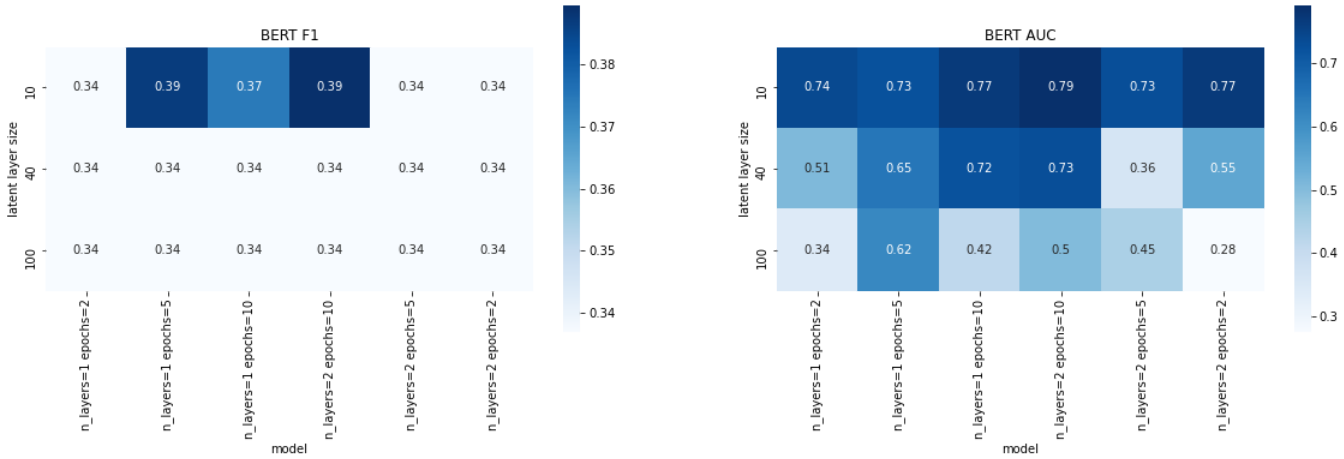

Figure 21: Transformer model performance for the long term task

We see from the results above that the F1-score is fairly stable, and is not very sensitive to the model choices we make. The AUC, however, does get affected significantly by the model choices. We see that a small embedding size (of 10 elements) works better - this is perhaps because a smaller number of trainable parameters reduces the variance of the model, not allowing it to overfit on the train data.

We also see that the best model was one that was trained on 10 epochs and had 2 dense layers for each FFNN structure. The 2 dense layers working better than 1 is probably indicating that having more layers allows for us to reduce the bias of the model and counteract the rigidity introduced by the small embedding size. The 10 epochs is an artifact of the model structure we adopted, where we take a mean across all the embeddings to make one prediction - this means each review embedding doesn't have its own feedback loop, but instead has to share one with other reviews in that data point. This is perhaps the reason why we need to regress on each data point 10 times for us to get the best result.

We have also completed an experiment on the max sequence length that we set for the transformer model, and look at the performance as a function of this below:

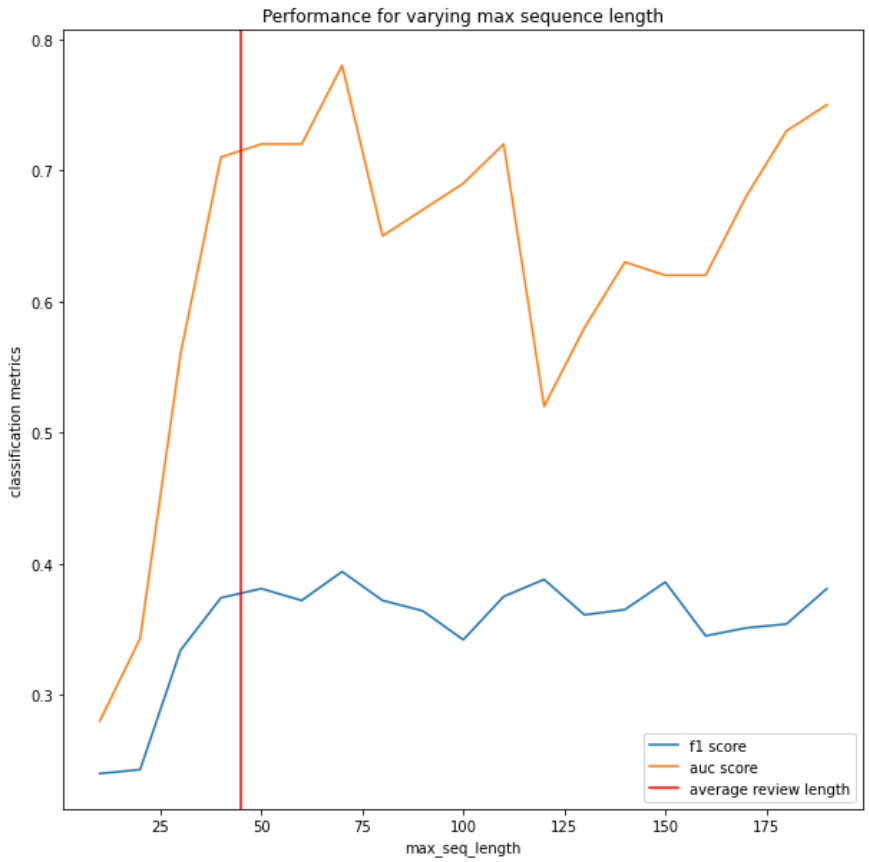

Figure 22: Transformer model performance w.r.t. max sequence length

We see in the above plot that the performance of the model is poor when the max sequence length is very small - this is because we are, perhaps, getting rid of important information and truncating reviews to be too small. We also see that the performance overall sees a plateau once the max sequence length exceeds the mean review length. This is an indication that the model doesn't need to read the entire review to get the information it requires for the prediction, and after a certain threshold, the longer review contains a similar signal to the start of the

review, so we don't need to read the whole review to make a prediction. This can also be taken a step further, and can be interpreted as the threshold to which most customers will stop reading a review before making a purchase decision.

We plan to complete more interpretability work on the BERT model that is highlighted in the future work section below.

7.3 Ensemble Model

To gain some insights into how review texts can improve model performance, we pass all of our individual model predictions through an ensemble model.

In the ensemble model, the feature variables are the predicted probability of being successful from the four individual models (two non-text-based models: XgBoost and random forest; two text-based models: bag of words and BERT). The target variable is the true label of whether the product is successful after one year. We experiment with two simple models to aggregate individual predictions: a logistic regression and a decision tree classifier. The ensemble model is intentionally kept simple to avoid overfitting, especially after an already exhaustive hyper-parameter tuning within each model class. In either case, we fit the model on the validation set[11] and make predictions on the test set to evaluate the performance of the ensemble models.

In the table below, we compare the performance of our individual models on a test set along different performance metrics. Note that all models are tuned to maximize F1-score—all other metrics, AUC, precision, and recall, are computed using the set of hyper-parameters that attains the highest F1-score.

Models	F1	Accuracy	AUC	Precision	Recall
Ensemble (logistics regression)	0.437	0.835	0.805	0.557	0.360
Ensemble (decision tree)	0.536	0.839	0.786	0.548	0.524
BERT	0.438	0.872	0.755	1.00	0.281
Bag of Words	0.321	0.762	0.670	0.325	0.317
XgBoost	0.388	0.829	0.758	0.532	0.305
Random forest	0.402	0.832	0.759	0.547	0.317

In addition, in Figure 23, we plot the ROC curve of the ensemble predictions as well as the individual model predictions.

[11] In particular, we fit the model on the validation set. The same validation set is used for cross-validation for all individual models. We choose not to fit the ensemble model on the training set, because the non-text-based models are overfit to the training set—they correctly predict almost 100% of observations; however, the text-based models, especially the BERT model, are not overfit to the training set. As a result, if we fit the ensemble model on the training set, the ensemble model will simply use the predictions from the non-text-based models.

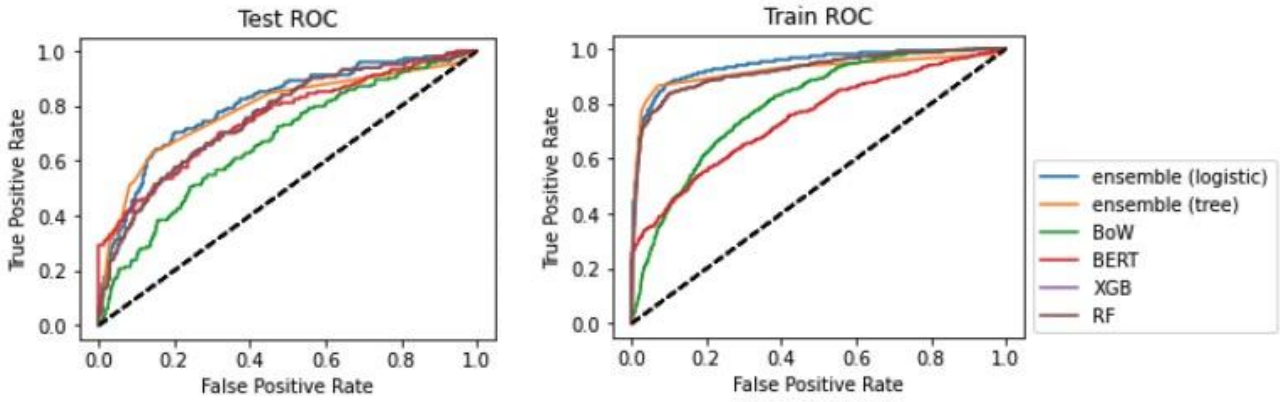

Figure 23: ROC curve of different models

Overall, the ensemble models outperform all individual models. The only exception is for precision score—while the BERT model has a precision of 1,[12] the ensemble model achieves a much lower precision. In particular, the BERT model makes 46 positive predictions, all of which are correct. As a comparison, 86 out of 152 and 59 out of 106 positive predictions are correct from the decision tree and logistic regression ensemble model, respectively.

The increase in performance of the ensemble model suggests that the non-text-based models and the text-based models have different advantages. Especially, using review texts helps model predictions even if we have historical sales data. Specifically, F1-score increases quite substantially from 0.4 in a random forest non-text model to over 0.5 in the decision tree ensemble model.

It is also hard to distinguish which of the two ensemble models is better—the decision tree results in higher F1 while the logistic regression results in higher AUC. In fact, depending on the relevant performance metric, neither ensemble model might be ideal to make the final prediction. To see this, recall that the BERT model is 100% correct on positive predictions. However, neither ensemble model sticks fully to the positive predictions from BERT. For example, the logistic regression ensemble model incorrectly predicts 24 of the 46 positive observations to be negative, and the decision tree ensemble model incorrectly predicts 2 of the 46 positive observations to be negative. In some real life applications, it might be detrimental for retailers to miss even a single successful product, in which case, an ensemble model that takes all positive predictions from the BERT model (and potentially some positive predictions from other individual models) might be more beneficial.

8. Future Work

We plan to focus on the interpretation of our text-based models for the rest of this project. In particular for BERT, we are looking to gain insights into what BERT performs well on and what it doesn't do so well on. We are also looking to gain insights on how BERT's predictions change with properties of the reviews.

More specifically, we will be looking at how BERT's predictions change for different numbers of

[12] BERT is especially stringent on positive predictions. The false negative predictions, however, do not show a clear pattern in terms of the actual underlying BSR. The false negative products have minimum BSR almost uniformly from 0 to 3000 (which is the cutoff of a successful product).

reviews fed in, different average lengths of reviews, different average review ratings, and different number of verified reviews. This will give us a better understanding of the different corpuses in our dataset and how BERT is interpreting these properties of reviews to make predictions for success.

For the above mentioned properties, we will be visualizing them using plots similar to the ones we have used before (specifically the binscatter plot). This will help us identify overall trends and gain interpretability for a complicated model like BERT.

We also want to look more at the contents of the reviews themselves, and will, therefore, be looking at creating a bag of words model on reviews BERT predicted to be for successful products and on reviews that were predicted to be unsuccessful. This will give us a better understanding of what tokens/keywords BERT identified as important while making decisions.

References:

[1] Geng Cui, Hon-Kwong Lui & Xiaoning Guo (2012) The Effect of Online Consumer Reviews on New Product Sales, International Journal of Electronic Commerce, 17:1, 39-58, DOI: 10.2753/JEC1086-4415170102

[2] Pryzant, Reid, Young-joo Chung and Dan Jurafsky. “Predicting Sales from the Language of Product Descriptions.” eCOM@SIGIR (2017).

[3] Nikolay Archak, Anindya Ghose, Panagiotis G. Ipeirotis, (2011) Deriving the Pricing Power of Product Features by Mining Consumer Reviews. Management Science 57(8):1485-1509.

[4] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. 2007. ARSA: a sentiment-aware model for predicting sales performance using blogs. In Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR '07). Association for Computing Machinery, New York, NY, USA, 607–614. DOI:https://doi.org/10.1145/1277741.1277845

[5] Chern, CC., Wei, CP., Shen, FY. et al. A sales forecasting model for consumer products based on the influence of online word-of-mouth. Inf Syst E-Bus Manage 13, 445–473 (2015). https://doi.org/10.1007/s10257-014-0265-0

[6] Elizabeth Fernandes, Sérgio Moro, Paulo Cortez, Fernando Batista, Ricardo Ribeiro, A data-driven approach to measure restaurant performance by combining online reviews with historical sales data, International Journal of Hospitality Management, Volume 94, 2021, 102830, ISSN 0278-4319, https://doi.org/10.1016/j.ijhm.2020.102830.

[7] Balakrishnan, V., Shi, Z., Law, C.L. et al. A deep learning approach in predicting products’ sentiment ratings: a comparative analysis. J Supercomput 78, 7206–7226 (2022). https://doi.org/10.1007/s11227-021-04169-6

[8] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 6000–6010.

[9] Devlin, Jacob, Ming-Wei Chang, Kenton Lee and Kristina Toutanova. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” ArXiv abs/1810.04805 (2019): n. pag.