Abbas Dharamsey | A10886858
Antriksh Yadav | A98002799
Hung Nguyen | A11299922
Tim Hedstrom | A11649142

Assignment 2 CSE 190 | Dahta Mining and Predictive Analytics

In this paper, we explore patterns in the relation between user review metadata and user ratings in an online store. We analyze a sufficiently large and broad dataset comprising of Amazon user reviews on movies. Specifically, we try to predict a given user's rating for a given movie using several pieces of metadata from the user's review of the same movie. We analyze reviews and study various predictive analytics methods to see what works best when predicting the user's rating for the reviewed item.

Exploratory Analysis

Dataset statistics

Number of reviews	*7,911,684*
Number of users	*889,176*
Number of movies	*253,059*
Timespan	*August 1997 - October 2012*

Review structure

The following example represents what the data points (reviews) looked like.

```
product/productId: B00006HAXW
review/userId: A1RSDE90N6RSZF
review/profileName: Joseph M. Kotow
review/helpfulness: 9/9
review/score: 5.0
review/time: 1042502400
review/summary: Pittsburgh - Home of the OLDIES
review/text: I have all of the doo wop DVD's and this one is as good or better than
the 1st ones. Remember once these performers are gone, we'll never get to see them
again. Rhino did an excellent job and if you like or love doo wop and Rock n Roll
you'll LOVE this DVD !!
```

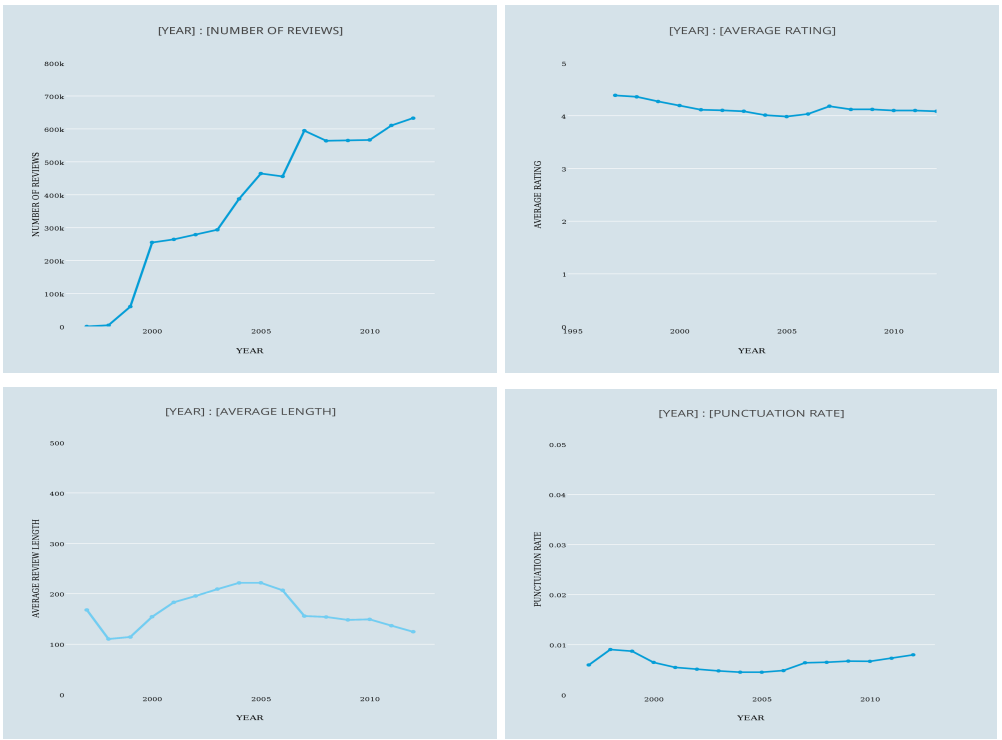
product/productId	*asin, e.g. amazon.com/dp/B00006HAXW*
review/userId	*id of the user, e.g. A1RSDE90N6RSZF*
review/profileName	*name of the user*

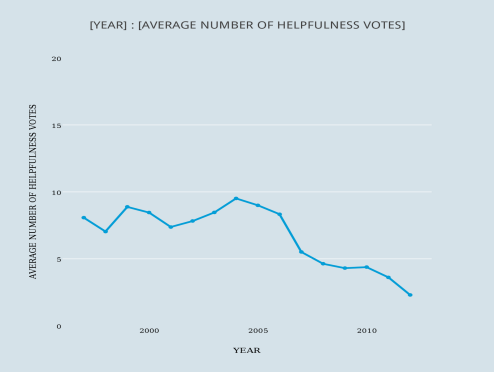
review/helpfulness	*fraction of users who found the review helpful*
review/score	*rating of the product*
review/time	*unix timestamp of the review*
review/summary	*summary/title of the review*
review/text	*body text of review*

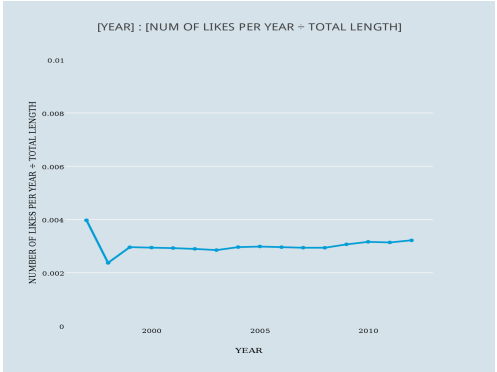
We performed some exploratory analysis on this dataset to get an idea of which feature we can possibly make a predictive model for. We calculated the following averages for each year:

- Average rating and number of reviews
- Length of reviews (in words)
- Number of punctuation characters as a fraction of review length in characters
- Number of helpfulness votes on reviews
- Number of instances of the word “like” in a review body

Exploratory Analysis Results



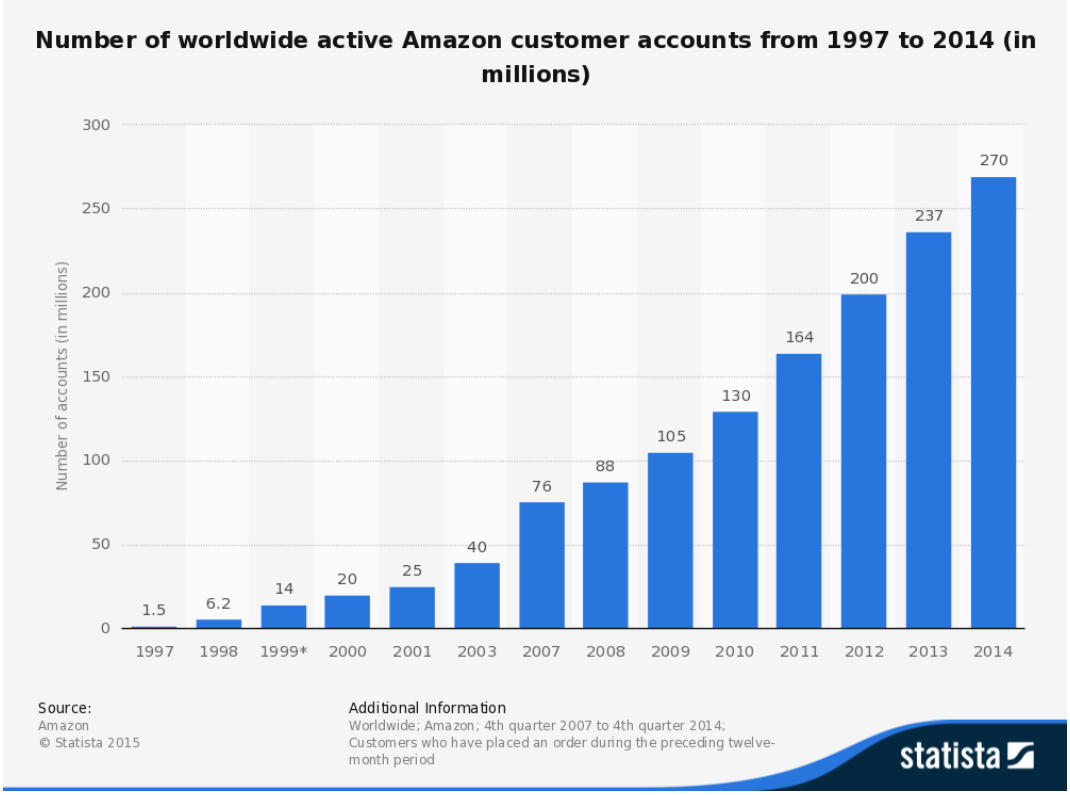




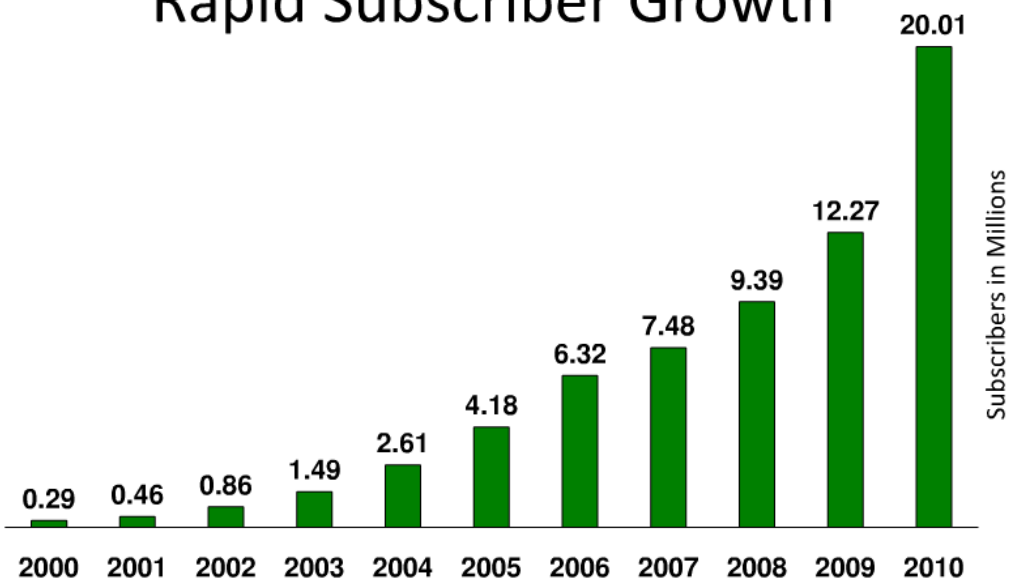
Notable findings

For several of these metrics, there was a peak around the mid-2000s. Number of reviews went up over time for an obvious reason - Amazon got larger over time.

The dip may be because Amazon started to become a less popular place to share and consume user reviews. This may explain the gradual dip in the average number of helpfulness votes after 2006.



Source: http://www.statista.com/statistics/237810/number-of-active-amazon-customer-accounts-worldwide/



Source: http://entertainment.slashdot.org/story/11/09/16/0043201/Netflix-To-Lose-1-Million-Subscribers

The average fraction of punctuation characters over length of review text also gradually and consistently increased after 2006 as well, before when there were several ups and downs. A reason for this could be the rise of the use of emoticons. The popularity of utilizing various elements of punctuation to express emotion has increased over the past couple of years.

One interesting trend was the number of movie reviews submitted to Amazon each year. After the initial spike from inception, the number of reviews grew linearly with respect to the number of Amazon users between 2000 and 2007, as seen in the charts. Since 2008, despite the number of Amazon users growing by millions per year, the number of movie reviews per year remained mostly stagnant. While we could not find any data to confirm, we hypothesize that the number of movies distributed through Amazon remained stagnant as well. Around the same time (2008), Netflix Instant was gaining users by millions per year. Amazon offered its own streaming service (Amazon Instant Video) in 2006, but Netflix seemed to be more popular with the audience. We believe this may have driven Amazon movie customers away and may also be a reason why the rise in the count of Amazon's movie reviews per year didn't correspond with the rise in users.

It is important to note that we have significantly less data from the first two years in this dataset. We have only 5 months of data from 1997, and Amazon was simply too unpopular until the turn of the millennium for us to be able to collect enough data points for data from those years to be meaningful. We are also missing the last two months of the last year in this dataset, 2012. While the number of reviews is still higher than the number from 2011 (because of Amazon's rapid growth), the metrics may be a little skewed when trends are considered.

Predictive Task

We decided to predict the user's star rating of a movie using a linear regression model. Linear regression seemed to be the correct choice based on our experience with homework problems that had us predict other Amazon review ratings. Training on a dataset of almost 8

million reviews seemed to be frivolous as we planned to have a test set of 100,000, so we cut our data set to ~2 million for training.

For a baseline, we simply predicted a movie's average rating if it was in our training set. Otherwise, we predicted the average score given by reviews of all movies in our training set.

We used mean squared error (MSE) and absolute error (AE) to measure the validity of our model's prediction. This combination will tell us how close our predictions are on average, and how close our predictions are in total.

We processed the raw text data into dictionaries where we could access each value using the field name as a key. For example, we used data['review/text'] to retrieve the review text.

For our predictive model, we attempted linear regression with different combinations of various metrics from the review data:

- Length of review text
  To extract the length of review text, we simply accessed *review/text*, split it at space characters and found the length of the resulting list
- Number of helpfulness votes
  Accessed the *review/helpfulness* field, split on a '/' and accessed second item in the resulting list (the *outOf* value) of the resulting list.
- Year of review
  Accessed *review/time* field, converted this Unix time into year using a standard time library function.
- Average rating by given user
  We computed average overall *review/score* for each user, over all of their movie reviews
- Average rating for given movie
  We calculated average *review/score* for each movie, similar to what Amazon likely does behind the scenes
- Most popular words
  We went through all the review text, removed stopwords and punctuation, and created a set of words with their counts and used only the thousand most popular.

We trained our predictor on a set of 2 million reviews and validated it on a set of 100,000 reviews.

It's important to note that not all of the features listed improved our results. Any time the number of helpfulness votes or the average rating for a given movie was included in regression, our MSE and ASE always decreased. For this reason, we decided to exclude those features from any of our models described below, though we included them here to present what we thought would be useful features.

Related Literature

Our movie review dataset comes from the Stanford Large Network Dataset Collection. The full dataset consists of around 8 million movie reviews from Amazon. The data spans the years between August 1997 to October 2012.

From Amateurs to Connoisseurs: Modeling the Evolution of User Expertise through Online Reviews by Julian McAuley and Jure Leskovec
Source:
http://i.stanford.edu/~julian/pdfs/www13.pdf

We're using data from Stanford that Julian McAuley and Jure Leskovec previously did research on. They used the Amazon movie reviews to analyze how experienced reviewers rate items differently. They learned that by taking into account user experience they can achieve a 14% reduction in MSE when compared to standard latent factor models. By introducing temporal dynamics into their models, they drastically improved predictive accuracy. They encoded temporal information into standard latent factor models through user experience. The same research was conducted on beer, fine food, and wine reviews. They had similar results across the board. They also noted that the mean squared error went down drastically as user experience level increased.

Predicting Star Ratings of Movie Review Comments by Aju Thalappillil, Rose Marie Philip, and Sagar V Mehta
Source:
http://cs229.stanford.edu/proj2011/MehtaPhilipScaria-Predicting%20Star%20Ratings%20from%20Movie%20Review%20Comments.pdf

Another group at Stanford analyzed how to predict star ratings of movie review comments. Aju Thalappillil, Rose Marie Philip, and Sagar V Mehta obtained a data set containing movie review comments from IMDB and the star ratings. They used text classification consisting of eliminating stop words, identifying negation of words, stemming, and dimensionality reduction. They evaluated the Naive Bayes and Multiclass SVM models. Naive Bayes ended up giving butter results than SVM when trying to predict the star rating of a user comment on the test data. They achieved a mean deviation of 1.1934, which is not as good as McAuley and Leskovec's introduction of temporal dynamics.

The conclusions of the similar work that has been done on these datasets goes to show that that there are methods of predicting star rating that are superior to anything that we attempted. Latent factor models incorporating temporal dynamics and naive bayes classifiers are far more useful than linear regression.

Results and Conclusions

Baseline

Baseline prediction: Predict the average rating of that movie, or predict the average rating of all movies if that movie is not in the training data.

Model evaluated by:
2,000,000 reviews in training set
100,000 reviews in test set

Results: MSE, AE: (1.6125386617239681, 100401.16348689367)

Despite this model being fairly naive, it wasn't an overly terrible predictor. On average, we predicted 1.6 (out of 5.0) off of the real value. However, we were not able to improve on it by a significant amount using linear regression, a method that is quite versatile.

Other Models

Split the average length of text by year, and average user ratings
MSE, ASE: (1.6057, 100087.6303)

This model was introduced after seeing that the average length of reviews changed significantly almost every year. After comparing this with other factors, we added the average user ratings to our regression because it produced a slightly lower MSE and ASE.

Split length of text by year, average user rating, and most popular word count
MSE, ASE: (2.3028, 90789.8114)

This model came about after evaluating the previous model and from a model that Professor McAuley provided, which predicted ratings for Amazon reviews from the popular words in all reviews. What is interesting is how the MSE and AE differ from the previous review. While our AE went down by almost ten-thousand, our MSE went up significantly. This meant that while the model was more accurate for certain predictions, our predictions were far off from others

Before using this model, we wanted to see if the most popular words for reviews varied per year. When we calculated this, we found that the most popular words were almost exactly the same each year, even after we removed stopwords ("the", "an" etc). This is because there were still too many common words specific to movie reviews that people used. In retrospect, we should have decided on a list of these specialized, movie-related stopwords and ignored those too. Using common words could have given us some more insight on specific users. Their writing style could have helped us predict their ratings better. A model like this could possibly work well when combined with movie metadata, which the dataset did not contain.

Year, length of text
MSE, ASE: (1.5192, 100029.1042)

By simply using linear regression using the length of the review and the year of reviews as features, we were able to reduce our MSE the most. We included both of these, and some other features in other models and derived worse results. This was probably because using those other features caused us to overfit the training set.

Latent factor models incorporating temporal dynamics and naive Bayesian classifiers work much better than linear regression when predicting rating because the features we used weren't linearly related to the rating score. The feature vector that lowered the absolute error the most was taking into account positive and negative words. Our failed model parameters included length of the review text, the total number of helpfulness votes, average rating for movie, and average rating for user.

Simply using a feature vector that included the year (16 slots for each year) beat our baseline MSE by 0.05.

The length of review didn't lower our MSE. We found that length of review is not linearly related to review score. In fact, the more extreme the rating the reviewer gave, the longer the review text was. More extreme ratings are ratings that are farther from 3, the middle review).

Our model:
y = X*(theta)
y → 'review/score'
X → feature vector
theta → feature weights

Our linear regression model failed because we found that features we used were not linearly correlated to our label, the score rating.

Why other methods worked

Latent factor models generally perform better than linear regression because they treat features independently. Linear regression does not treat features independently, which isn't always necessarily true. Introducing temporal dynamics into latent factor models improves predictive accuracy because it takes into account how different reviewers previously rated movies. It improves accuracy by taking into account how more experienced reviewers will rate certain items.

Linear regression may have worked better if the dataset contained more features. The dataset did not contain movie titles or genre data, so we could not incorporate those into our model. Ideally, we would have isolated each user's preferences towards various movie metadata and incorporated that information when making a rating prediction also. Featured cast and crew data could have been worth trying as well. If we were to try to use linear regression to predict the movie rating in the future, we should try to gather data that allows us to use these

features. Although we may or may not achieve a better predictor than other models, features that allow us to tune predictors to user preferences would likely yield better models.