Sentiment Analysis of Steam Review Datasets using Naive Bayes and Decision Tree Classifier

Zhen Zuo[1]

***Abstract*— Sentiment analysis or opinion mining is one of the major topics in Natural Language Processing and Text Mining. This paper will provide a complete process of sentiment analysis from data gathering and data preparation to final classification on a user-generated sentimental dataset with Naive Bayes and Decision Tree classifiers. The dataset used for analysis is the product reviews from Steam, a digital distribution platform. The performance of different feature selection models and classifiers will be compared. The trained classifier can be used to make prediction for unlabeled reviews and help companies to increase potential profits in global digital product market.**

Index Terms: Sentiment Analysis, Naive Bayes, Decision Tree, Feature Selection, Supervised Machine Learning, Text mining

I. INTRODUCTION

According to Fang 2015 [1], sentiment is an attitude or judgment based on feelings or experiences. One special type of sentiment is people's opinions after they consumed some products, such as watching a movie or having a dinner. For online stores or online market, former reviews can play an important role in customer's decision making process [2], which indicates that an accurate prediction of sentiment from a certain review can increase potential profit. In this case, Sentiment analysis can be applied. The one important goal of sentiment analysis is to measure the sentiment polarity of text data [3]. On the other hand, social media such as Twitter, Facebook and Yelp allow people to share their opinions in a real time manner with each other. The exponential of information includes an overwhelming amount of information about people's sentiments. For instance, the average tweets people sent per day is approximately 500 million referred to Krikorian [4], the VP in Twitter Inc. It makes Internet a resourceful place to gather sentiment information. Fortunately, many social media sites released the application programming interfaces (APIs), which makes data collection and data cleaning possible. For example, Twitter published Analytics Tools in 2014 to help users build free interactive dashboard. It can be used to summary Engagements of a tweet through Engagement Rate, Number of Link clicks, Number of Retweets, Number of Likes and Number of Replies (For more details in https://analytics.twitter.com). Twitter also provides Twitter Ads Campaigns services based on the output from Analytics for business partner.

The reviews analyzed are from the global game market. From Figure 1, there is a large market with more than 100 billion dollars annually and the market is still increasing

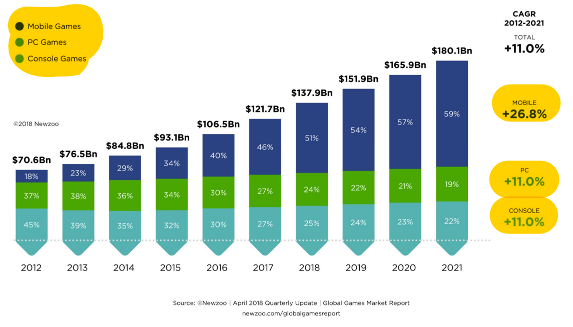


Fig. 1: Global Game Market Report April 2018 by Newzoo.

rapidly. The values after 2018 in Figure 1 are predictions from Newzoo. Unlike traditional products, digital products such as games in most of cases can be only purchased online. People cannot try the product before purchasing them. In addition, people are less likely to get refund after they make the purchases. Unlike movies, most of products does not have trailers. Therefore, reviews may be the only source that people can know more about the products' user experience before they actually buying them. Sentiment analysis can be very necessary and meaningful for game products. The data used for this research is the Steam Review Dataset [5]. It is a binary sentiment classification dataset containing over hundreds of millions reviews in multiple languages. It is also well labeled by Steam community members. Steam community members are providing insightful information in terms of their opinions on games or other softwares. Steam Database also provides API to access reviews and user information. More details about the data can be found in the Data Gathering section.

II. METHOD

A. Data Gathering

Several methods have been experimented to gather review data from Steam. Three main methods are listed as following:

1) Steam Database: Real time data has been collected and processed in several sites such as SteamDB and SteamSpy. All of them are free third-party tools designed to give better insight into the applications and packages. It provides more comprehensive information than Steam official API. Some digits were calculated and estimated with the algorithms from other sources. It is one of the most convenient way to know the data with interactive dashboard. In Figure 2, it is an instance of estimated daily online player for Dota 2 in past three years. The red line is showing how many Twitch viewers of this game. Like Youtube, Twitch is a large

[1]Email: `zhenzuo2@illinois.edu`

online game steaming platform. The interactive dashboard can be accessed with https://steamdb.info/app/570/graphs/. It includes past six years data. The vitalizations can be extremely helpful but they are not used in this study because the raw data behind those plots were not allowed to be scrapped. However, the latest updates allow user to download the csv or XLs format of the data with a provided link. Now, it may be the most effective way among only a few available methods to get information on how many real time players are online, the estimated number of sales and the number of viewers in direct broadcast.

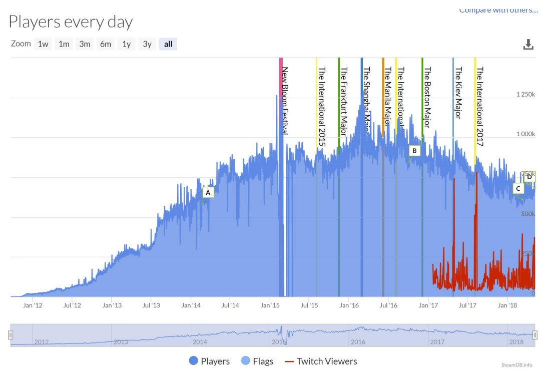
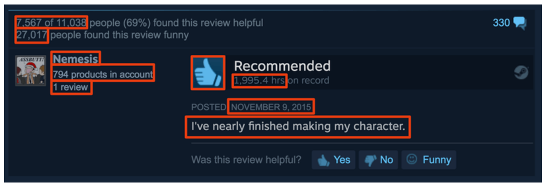


Fig. 2: Interactive dashboard for Players every day for Dota 2 in past three years

2) *Data Scrapping:* In Github, some scrapper were developed to scrap reviews data directly from the official website. Most of scrappers were built with Scrapy in Python. Both products information and user reviews information can be scrapped. The advantage of this method is the reviews are most complete and updated. The disadvantage is very time consuming. Error may occur if the scrapping speed or the number of scrappers exceed the maximum limits. However, the maximum limits are not indicated clearly. The scrapping process is done by iterations. 20 reviews can be found from each iteration. However, one error in one iteration can lead to the fails of all the subsequent iterations. Moreover, it may take a few days to scrap all reviews from Steam using this method.

3) *API:* This is the finally chosen method. The first step is to get all IDs and names for all products and the IDs were saved to a list. The second step is to pull all the product information with a product API by each ID obtained from the step 1. The file obtained from API are json files. It is saved after transforming to a csv data table. The next step is using the ID again with a review API by each ID to get all the reviews. Each product generates one json file. By merging all the reviews together, a csv file is retained for analysis. The reviews got from API are not in a real time manner but they are updated regularly. Additionally, user information can be download with user information API. An API key is needed to use that API. There are lots of missing values so they are removed from the final dataset.


7,567 of 11,038 people (69%) found this review helpful
27,017 people found this review funny
330

Nemesis
794 products in account
1 review

Recommended
1,995.4 hrs on record

POSTED NOVEMBER 9, 2015

I've nearly finished making my character.

Fig. 3: An example of review from Steam API

In Figure 3, it shows an example of raw data from Steam Review Dataset. Areas in red boxes are the areas API can access. More details about columns can be got from API can be found in the Code & Data section.

B. Data Processing

1) *Remove special characters and digits:* Punctuations were removed first. Even though the majority of reviews are in English, there are still a small portion of reviews are in Russia, Chinese, and other languages. Special characters and digits were removed by only keeping the lower and upper cases of English letters.

2) *Lower case:* From the last step, all texts other than letters have been removed. Then it is necessary to transform all the letters appeared in the reviews to lower case so as to reduce the size of words.

3) *Remove stop word:* The stop words lists are from NLTK package in Python. Other stop words are selected manually based on the world cloud plot. For example, ”get” is not a word in the original NLTK package but actually appeared lots of times in the text and should be safely ignored, it is treated as a stop word.

4) *Stemming:* Stemming was used to reduce the size of words. The stemmer used is Porter Stemmer in NLTK.

5) *Remove links:* Links are removed because links may not include sentiment information. Links are removed with regular expression. If a string matches the form ”http://”, it is removed.

6) *Remove most frequently words:* The threshold N is defined as if a word appeared more than N reviews, this word is removed. Words appear too frequently may contain less information for sentiment analysis. The threshold N is chosen with cross validation.

7) *Remove most infrequently words:* The threshold N is defined as if a word appeared less than N reviews, this word is removed. Words appear too infrequently may contain less information for sentiment analysis. Most of the time those words are misspelled or crated by a user. Such as ”Yaaaaaaaaaaaaaaaaaaaaaa”. The threshold N is chosen with cross validation.

8) *Correct misspelled word:* Since there are some misspelled words, methods have been attempted to correct them. Packages in Python such as Hunspell and autocorrect have been tried. Packages such as PyEnchant can also help to check whether a word exists in English. However, there are two reasons to explain why those methods may not be very effective: (1) All of them are time-consuming. The basic idea of Hunspell and autocorrect are calculating the distance of possible misspelled worlds and correct words in a dictionary

based on some distance function then output the most similar one . The whole process will take a long time for a large amount of text since it takes some time to compute each word in it. And it may still not reduce much word size compared lower case and stemming. (2) Now automatic correcting tools are almost everywhere. Most of the input method editors (IME) can underline the misspelled words with a red line. As a result, the remaining majority of misspelled words may be misspelled on purpose, or in a more complex case, be automatically corrected to another correct word by input method editor. (3) All of those methods do not consider the context. The correction is not accurate by ignoring the context. So finally, this step is skipped.

9) Remove short reviews: After all of the steps above, some reviews only have a few words. They are too short and may bring noise to the classifier. The threshold N is defined as once the number of words in the cleaned review is less than N, then that review is removed.

After precessing, data is converted to a sparse two-dimensional matrix. The rows and reviews have the same number, the columns and unique words have the same number as well. Each element represents the number of appearances of each word in each review.

C. *Feature Selection and Vectorizer*

According to Forman, feature selection is essential to make an efficient and more accurate classifier. Sharma in 2012 [8] investigated the performance of different feature selection methods for sentiment analysis. It shows that information gain gives consistent results while gain ratio performs best overall. This result is from a movie reviews data with 2000 documents. This experiment will be repeated for the Steam Dataset.

1) N-gram: N-gram is a way to generate feature. Dave [7] found that in some settings bigram and trigram perform better than unigram. For bigram, every pair of words (w1,w2) next to each other in the document is selected as a feature. In this representation, a feature is associated with all the bigrams in the document. The feature value is defined as the number of times the bigram occurs in the document. For bigram features, features in the unigram are also included. This is the same for trigram, features from both bigram and unigram are also included. The performance of unigram, bigram, and trigram are tested with cross validation.

D. *Popularity score*

The number of positive, negative and neutral words can be sat in three extra columns. However, they are removed from the final model because they are accurate for stemmed word in SentimentIntensityAnalyzer in Python's NTLK package. For example, awesome is a positive word with polarity_scores greater than 0.5, but the stemmed version of awesome, which is awesom is a neutral word. Also, like and likes have different polarity_scores. So popularity score was not used.

E. *Information Gain*

Information Gain (IG) is a measure of information gained (in bits) for classifying a text document by evaluating the presence or absence of a feature in a text document. A useful feature should decrease more entropy than a useless feature. For a binary classification problem, the entropy of a partition D is given by

$$Info(D) = -\sum_{i=1}^{2}(P_i)log_2(P_i)$$

where p_i is the proportion of instances of each class or category. For example, P_1 represents the proportion of positive reviews and P_2 represents the proportion of negative reviews.

To classify the documents in D on some feature with attributes $A\{a_1, ...a_v\}$, the whole documents D will be split into v partitions $\{D_1, D_2, ..., D_v\}$.The memory need to store those small partitions are the entropy after splitting:

$$Info_A(D) = -\sum_{j=1}^{v}\frac{|D_j|}{|D|} \times Info(D_j)$$

where $|D_j|$ is the number of documents in D_j and Info(D_j) can be calculated the same way as the formula above.

The formula for IG is simple but it may take a long time to compute for a large dataset. IG for each word need to be computed separately. For my dataset, the word vectorization is saved to a sparse matrix in python and the label is saved to another dataframe. It is time-consuming to count how many instances in each label by summarizing columns in a sparse matrix one by one. To simply the computation, the word vectorization is transformed to a binary matrix rather than the times of appearance. Each element is a binary variable indicating the present of the word in the review. So v in the above formula should always equal to 2. Also, only one hundred thousand reviews are randomly selected for the IG part. The number of features used for the final model is selected with cross validation.

F. *TF-IDF*

TF-IDF is the short for term frequencyinverse document frequency. It is a vectorizer and TFIDF-transformed data can be used directly for the classifier. The term frequency is calculated for each term in the review as

$$TF(t,d) = \frac{number\ of\ times\ terms\ t\ appears\ in\ document\ d}{total\ number\ of\ terms\ in\ documentd}$$

where t is the term and d is the document.

The Inverse Document Frequency is calculated by

$$IDF(t,d) = log(\frac{total\ number\ of\ documents\ D}{number\ of\ documents\ wth\ the\ term\ in\ it})$$

After calculating TF and IDF with the two formulas above, the TFIDF is calculated by

$$TFIDF(t,d,D) = TF(t,d) \times IDF(t,D)$$

TFIDF can be used as a feature selection method to select top n features from all features with the largest weights. The weight for each term is defined by taking average of that term in all documents. The number of features selected for the final model is chosen by cross validation.

Other improved version of TFIDF has been developed to improve the performance. Wang 2010 [10], distribution information among classes and inside a class is used to develop the weights function. Moreover, Martineau in 2011 proposed Delta TFIDF [11] and the accuracy has dramatically improved with SVM classier.

III. Model

A grid search is performed to find the best combination of hyperparameters. The possible parameters can be tuned including review minimum length, maximum of number of reviews appeared, minimum of number of reviews appeared, gram range, number of features, feature selection method, include additional features or not, and model. The space of paymasters are listed in the TABLE I:

TABLE I: Grid Search of hyperparameter space

Minimum Length	5,10
Minimum Appearance	10,50
Maximum Appearance	20,000,100,000
Gram Range	(1,1),(1,2),(1,3)
Number of Features	100,500
Feature selection method	IG, TFIDF
Include Additional Feature or not	YES, NO
Model	Tree, Gaussian Naive Bayes

The binary variable Include Additional Features is whether include extra features from the Steam Review Dataset. The whole list of additional features can be accessed with the shared files in the Code & Data section. The six additional features selected are viewer's playtime forever, viewer's playtime in last two weeks, viewer number of games owned, number of reviews viewer made in total, how many other viewers votes up this review, and how many other reviewers think this review is funny.

Cross validation is used for model evaluation, 75% of all the data are randomly selected as training data and the remaining 25% are testing data.

IV. Result

In gathered data, there are 7,705,997 reviews from 22,548 games are collected. Among those reviews, 6,410,832 reviews are positive and 1,295,165 reviews are negative. Data was balanced by keeping all the negative reviews and randomly subset the same number of positive reviews from the data. After gathering and cleaning the data, a world cloud is showed in Figure 5.

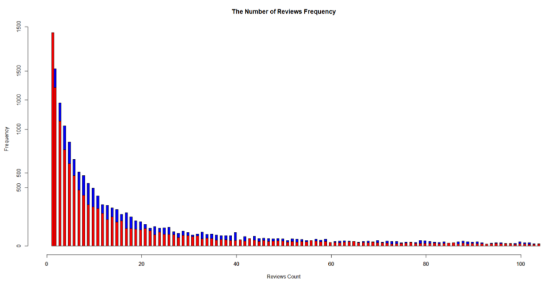


Fig. 4: How many positive and negatives reviews for each game?

The red parts in the histogram in Figure 4 are the number of positive reviews and the blue parts are the number of negative reviews. Most of games have less than 20 reviews. On the other hand, the top five most popular games got about 30% of all reviews.

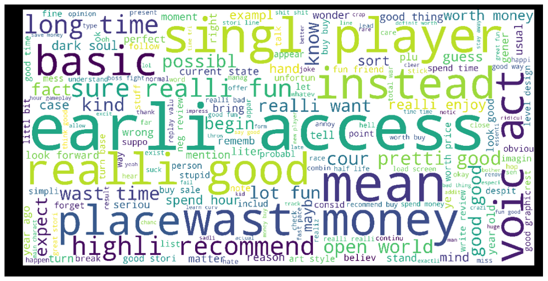


Fig. 5: Word Cloud for all words

The size of the word in this word cloud indicates the frequency. This plots includes the frequency of all the unigrams and bigrams. In this plot, there are some positive words such as really fun, highly recommended, good story and spend hours. Also, there are some negative words such as waste of money, waste of time, unfortunately, and mean. All the words here are stemmed format because the data used was cleaned. This word cloud can also help to identify stop word.

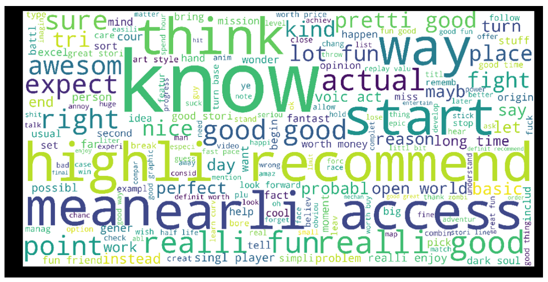


Fig. 6: Word Cloud for all words in positive reviews

In Figure 6, it shows the word cloud for all words appeared in positive reviews. The word mean should be a negative word but it seems to appear a lot of times in the positive reviews. Other than mean, most words are positive and neutral.

Fig. 7: Word Cloud for all words in negative reviews

In figure 7, it shows the word cloud for all words appeared in negative reviews. Most of those words are negative other than some verbs such as know, tell, and turn.

Finally, 196 different model were fitted. 192 out of 196 models are fitted with TFIDF and only four models are fitted with IG for comparison. Marginal distribution and density plot is applied to visualize the results for the 192 models with TFIDF. The assumption behind this marginal distribution is due to the balance design of this grid search, the influence of other factors can be safely ignored when studying the marginal distribution of one variable. Figure 8 compares the accuracy with or without six extra features. It is not clear to see any pattern from this plot. This distribution may be influenced by other variables.

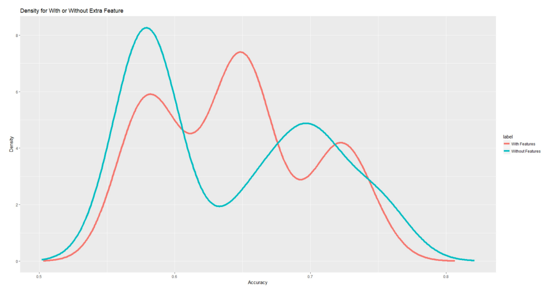


Fig. 8: The Density Distribution of Accuracy: With or Without Extra Feature

In Figure 9, it shows the comparison of how accuracy varies among different number of features selected. The red line is the density distribution of 100 features and the blue line is the density distribution of 500 features. It is clear to see that overall the blue line is on the right side of the red line, which indicates that overall models with 500 features are more accurate than the models with only 100 features. The 100 variables models may underfitting the data.

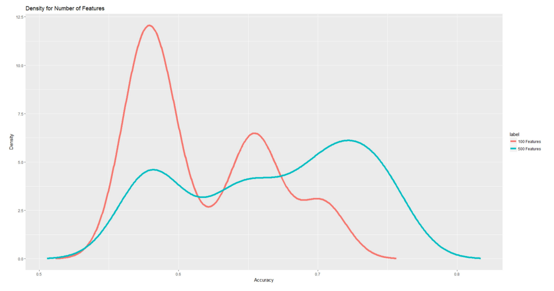


Fig. 9: The Density Distribution of Accuracy: 100 Features and 500 Features

Figure 10 compares the accuracy of Decision Tree Model and Gaussian Naive Bayes Model. The Decision Tree Model performs much better overall than the Gaussian Naive Bayes Model. The reason to explain this may be the feature selection method is TFIDF. The results should be different if the feature selection method is IG.

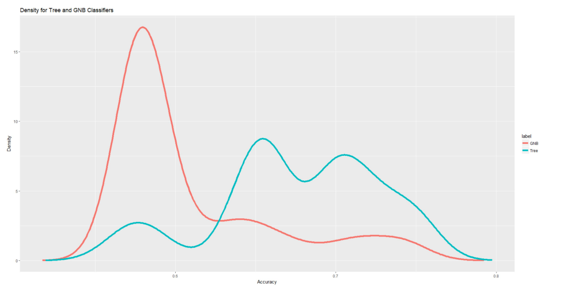


Fig. 10: The Density Distribution of Accuracy: Decision Tree Model and Gaussian Naive Bayes Model

Of course more density plots can be draw with the results from the 192 models. Performance can be also measured with precision and recall rather than accuracy alone.

Next, it is better to take a closer look at each model and focus on one of the best models by ranking the accuracy, precision and recall. A screen shot of top models is listed in Figure 12. This is only the head of the performance table with subset columns. The full table can be found in files in the Code & Data section.In Figure 11, it shows the Receiver Operating Characteristic Curve of the top model.

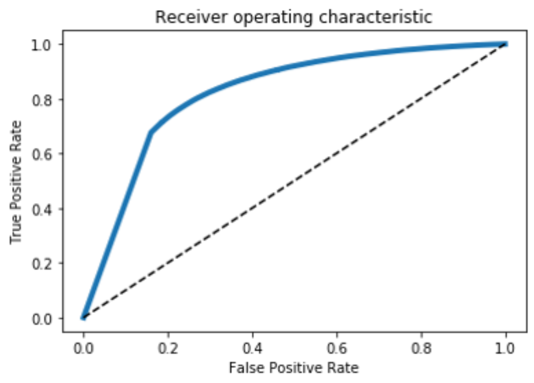
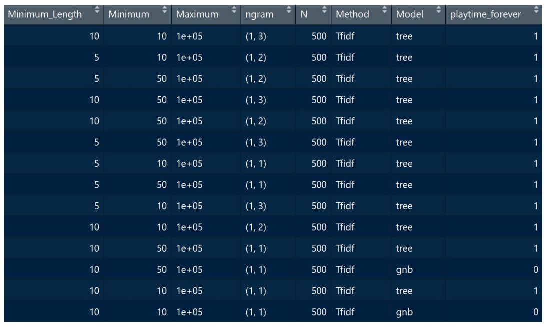


Fig. 11: ROC curve of the best model

The accuracy of the top model is about 74.95% with 70.25% recall. Among top models, the minimum length for reviews with 10 is better than 5. The minimum number appearance of reviews most are 10 rather than 50. Also, trigram and bigram perform a little better than the unigram but with a overall similar average. Models with 500 features perform totally better than models with 100 features. Decision Tree model also performs better than the Gaussian model. Models with extra features always perform better than the models without extra features.

The accuracy of the models with features selected by IG is about 60%. This is not comparable with TFIDF because model with TFIDF uses all the reviews data but models with IG only use ten thousand reviews for limited computing power.



Minimum_Length	Minimum	Maximum	ngram	N	Method	Model	playtime_forever
10	10	1e+05	(1, 3)	500	Tfidf	tree	1
5	10	1e+05	(1, 2)	500	Tfidf	tree	1
5	50	1e+05	(1, 2)	500	Tfidf	tree	1
10	50	1e+05	(1, 3)	500	Tfidf	tree	1
10	50	1e+05	(1, 2)	500	Tfidf	tree	1
5	50	1e+05	(1, 3)	500	Tfidf	tree	1
5	10	1e+05	(1, 1)	500	Tfidf	tree	1
5	50	1e+05	(1, 1)	500	Tfidf	tree	1
5	10	1e+05	(1, 3)	500	Tfidf	tree	1
10	10	1e+05	(1, 2)	500	Tfidf	tree	1
10	50	1e+05	(1, 1)	500	Tfidf	tree	1
10	50	1e+05	(1, 1)	500	Tfidf	gnb	0
10	10	1e+05	(1, 1)	500	Tfidf	tree	1
10	10	1e+05	(1, 1)	500	Tfidf	gnb	0

Fig. 12: Parameters for the best models

Compared with Decision Tree, Gaussian Naive Bayes requires the assumption class conditional independence else loss of accuracy. Also, by controlling the tee height, maximum leaf node size, and maximum number of features, Decision Tree at each node can make feature selection again by Information Gain or Gini Index. However, Gaussian Naive Bayes can only use the features selected from the former steps. Also, for a term, if there is no occurrences of a class label then the frequency-based probability estimate will be zero. Those are all disadvantages of Gaussian Naive Bayes and they can lead to lower accuracy compared with Decision Tree.

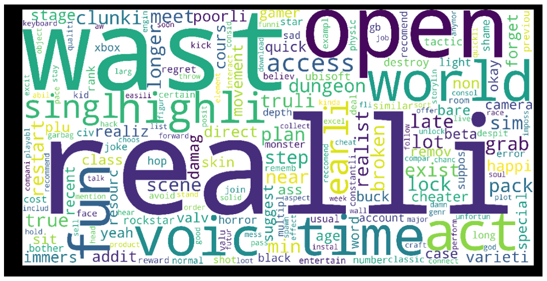
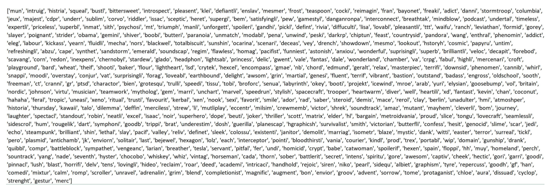


Fig. 13: What are terms in the best model?

In Figure 13, it shows the features selected by the best model. Some terms have very strong sentiment, such as waste, highly, fine, and really. The terms selected by TFIDF do make sense.



```
['mun', 'intruig', 'histria', 'squeal', 'bustl', 'bittersweet', 'introspect', 'pleasent', 'klei', 'defiantli', 'enslav', 'mesmer', 'frost', 'teaspoon', 'cocki', 'reimagin', 'fran', 'bayonet', 'freaki', 'adict', 'danni', 'stormtroop', 'columbia',
'jeux', 'majest', 'cdpr', 'underr', 'sublim', 'corvo', 'riddler', 'issac', 'sceptic', 'heret', 'supergi', 'bem', 'satisfyingli', 'pew', 'gamestyl', 'danganronpa', 'interconnect', 'breathtak', 'mindblow', 'podcast', 'undertal', 'timeless',
'expertli', 'priceless', 'superbl', 'inmat', 'sith', 'psychosi', 'mt', 'triumph', 'manli', 'unforgett', 'spoileri', 'gandhi', 'pickl', 'defint', 'rivia', 'diffuculti', 'lisa', 'lovabl', 'pleasantli', 'ttt', 'waifu', 'ranch', 'leviathan', 'formid', 'gorey',
'slayer', 'poignant', 'strider', 'obama', 'gemini', 'shiver', 'boobi', 'butteri', 'paranoia', 'unmatch', 'modabl', 'pena', 'unwind', 'peski', 'darkrp', 'chiptun', 'feast', 'countrysid', 'pandora', 'wang', 'enthral', 'phenomin', 'addict',
'eleg', 'labour', 'kickass', 'yearn', 'fluidli', 'mecha', 'nors', 'blackwel', 'totalbiscuit', 'sunshin', 'ocarina', 'scenari', 'deceas', 'vey', 'drench', 'showdown', 'mesmo', 'lookout', 'hstoryh', 'cosmic', 'papyru', 'untim',
'refreshingli', 'abzu', 'cape', 'synthet', 'sandstorm', 'emerald', 'soundscap', 'regim', 'flawless', 'homag', 'pacifist', 'funniest', 'astonish', 'anxiou', 'wonderful', 'suprisingli', 'superb', 'brilliantli', 'veloc', 'decapit', 'forebod',
'scavang', 'corn', 'redon', 'inexpens', 'chernobyl', 'stardew', 'glado', 'headphon', 'lightsab', 'princess', 'delic', 'gwent', 'vale', 'fantas', 'dale', 'wonderland', 'chamber', 'va', 'crpg', 'fabul', 'highli', 'mercenari', 'croft',
'playground', 'bard', 'wheat', 'theif', 'shooti', 'baker', 'flour', 'lightheart', 'lsd', 'crytek', 'hexcel', 'encompass', 'gmae', 'nb', 'chord', 'edmund', 'geralt', 'relax', 'masterpiec', 'terrifi', 'downsid', 'phenomen', 'cannib', 'whirl',
'snappi', 'moodi', 'overstay', 'conjur', 'vat', 'surprisingli', 'forag', 'loveabl', 'earthbound', 'delight', 'awsom', 'grin', 'martial', 'genesi', 'fluent', 'terrif', 'vibrant', 'bastion', 'outstand', 'badass', 'engross', 'oldschool', 'sooth',
'freeman', 'ct', 'cranni', 'gr', 'ptsd', 'charactor', 'bien', 'grotesqu', 'trulli', 'speedi', 'tissu', 'tobi', 'broforc', 'senua', 'labyrinth', 'okey', 'booti', 'projekt', 'icewind', 'mroe', 'arab', 'yuri', 'elysian', 'goosebump', 'vol', 'britain',
'nordic', 'johnson', 'virtu', 'musician', 'teamwork', 'mytholog', 'gem', 'marri', 'unchart', 'marvel', 'speedrun', 'stylish', 'spacecraft', 'trooper', 'heartwarm', 'diver', 'well', 'heartili', 'xd', 'fantast', 'kevin', 'chan', 'coconut',
'hahaha', 'feral', 'tropic', 'uneasi', 'xeno', 'ritual', 'trusti', 'favourit', 'kerbal', 'xen', 'nook', 'sexi', 'favorit', 'smile', 'ador', 'rad', 'saber', 'steroid', 'demis', 'mace', 'rerol', 'clay', 'berlin', 'unadulter', 'hmi', 'atmoshper',
'historia', 'thursday', 'kawaii', 'talo', 'dilemma', 'deffin', 'merciless', 'strew', 'li', 'mutliplay', 'eccentr', 'milsim', 'crewmemb', 'victor', 'shrek', 'soundtrack', 'amaz', 'mutant', 'mayhem', 'cleverli', 'bom', 'journey',
'laughter', 'spectacl', 'standout', 'robin', 'neatli', 'excel', 'isaac', 'noir', 'superhero', 'dope', 'beuti', 'joker', 'thriller', 'scott', 'matrix', 'elder', 'hl', 'bargain', 'metroidvania', 'proud', 'slice', 'tongu', 'lovecraft', 'seamlessli',
'sidescrol', 'hum', 'rougelik', 'dart', 'symphoni', 'goodb', 'trippi', 'brat', 'underestim', 'dosh', 'guerilla', 'planescap', 'hgraphicsh', 'survivalist', 'smith', 'victorian', 'butterfli', 'confess', 'hesit', 'genocid', 'slime', 'scar', 'jedi',
'echo', 'steampunk', 'brilliant', 'shin', 'lethal', 'slay', 'pacif', 'valley', 'reliv', 'definet', 'sleek', 'colossu', 'existenti', 'janitor', 'demolit', 'marriag', 'isometr', 'blaze', 'mystic', 'dank', 'witti', 'easter', 'terror', 'surreal', 'tickl',
'pero', 'plasmid', 'antichamb', 'jk', 'enviorn', 'solitair', 'last', 'bejewel', 'hexagon', 'lolz', 'each', 'interceptor', 'pointi', 'bloodthirsti', 'vania', 'courier', 'kindl', 'prod', 'trex', 'portabl', 'wip', 'domain', 'gunship', 'drank',
'quibbl', 'compr', 'battleblock', 'sympathet', 'vengeanc', 'larian', 'breather', 'tesla', 'servant', 'pitfal', 'fer', 'undi', 'homicid', 'crypt', 'babe', 'catwoman', 'spoilerif', 'hexen', 'spain', 'floppi', 'hh', 'muy', 'homeland', 'perch',
'sountrack', 'yang', 'nade', 'seventh', 'hyster', 'chocobo', 'whiskey', 'whiz', 'vintag', 'horseman', 'cada', 'thorn', 'sober', 'battlerit', 'secret', 'intens', 'spiritu', 'gore', 'awesom', 'captiv', 'cheek', 'hectic', 'gori', 'garri', 'goodi',
'pinnacl', 'lush', 'blast', 'horrifi', 'delv', 'tens', 'lovingli', 'hideo', 'reclaim', 'roar', 'deed', 'academi', 'intricaci', 'handhold', 'rejoic', 'siren', 'niko', 'pearl', 'sidequ', 'albiet', 'graphism', 'tyre', 'repercuss', 'goodh', 'gf', 'han',
'comedi', 'mixtur', 'calm', 'romp', 'scroller', 'unravel', 'adrenalin', 'grim', 'blend', 'completionist', 'magnific', 'augment', 'bon', 'envior', 'groov', 'advent', 'sorrow', 'tome', 'protaganist', 'chloe', 'aura', 'dissuad', 'cyclop',
'strenght', 'gestur', 'merc']
```

Fig. 14: What terms selected by IG?

In Figure 14, it shows what words are selected by IG. It may not agree with words selected by TFIDF because different datasets are used for those methods. Also, for IG, only unigram is used without bigram and trigram.

V. Conclusion

The aim of this study is to evaluate the performance for sentiment analysis in terms of accuracy, precision, and recall. We compared two supervised machine learning algorithms of Decision Tree and Gaussian Naive Bayes for sentiment analysis of the large scale Steam Review Dataset. The experimental results show that for the Steam Review Dataset, Decision Tree outperformed the Gaussian Naive Bayes and achieved around 75% accuracy. It also proves that extra features in Review dataset can help the Decision Tree to get better performance.

VI. Code & Data

Code can be found in https://github.com/zhenzuo2/IS_590_Final. All the ipython notebooks are in knitted version. Also a script shell file is prvided to run the py files. The folder Gather_ Data includes all the codes needed to gather data in Python. The code should be run with the following order (1) get_ detail.ipynb: Get all products ID and product details. Product details are optional. (2) getreview.ipynb: It can be run after getting the ID list from the first file. Review data will be save to json files. (3) to_ csv.ipynb: Convert the json review files to csv format. (4) game_ detail.R: This is optional. Convert the json product detail files to csv format.

The folder Classifier includes code to run the classifier. IG and TFIDF are saved to separate files.

Output results are saved to the result folder.

The raw data and processed data can be found in the Box folder https://uofi.box.com/s/z6enk2qdl2dmgqau9wyp77jnr9cmqgil. Four files are in this folder totally. (1) RawData.csv: the raw data from Steam API. (2) CleanData.csv: processed data. The specific processes are described in the data preparation section. (3) BalanceData.csv: A balanced dataset wit equal amount of positive reviews and negative reviews. (4) GameDetail.csv: A detailed information of each game.

Appendix

References

[1] Fang, X., & Zhan, J. (2015). Sentiment analysis using product review data. Journal of Big Data, 2(1). doi:10.1186/s40537-015-0015-2

[2] Utz, S., Kerkhof, P., & Bos, J. V. (2012). Consumers rule: How consumer reviews influence perceived trustworthiness of online stores. Electronic Commerce Research and Applications, 11(1), 49-58. doi:10.1016/ j.elerap.2011.07.010

[3] Tan, L. K., Na, J., Theng, Y., & Chang, K. (2011). Sentence-Level Sentiment Polarity Classification Using a Linguistic Approach. Digital Libraries: For Cultural Heritage, Knowledge Dissemination, and Future Creation Lecture Notes in Computer Science, 77-87. doi:10.1007/ 978-3-642-24826-9_13

[4] Krikorian, Raffi. (VP, Platform Engineering, Twitter Inc.). "New Tweets per second record, and how!" Twitter Official Blog. August 16, 2013.

[5] Sobkowicz & Stokowiec (2016). Steam Review Dataset - new, large scale sentiment dataset.

[6] Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. The Journal of Machine Learning Research, 3, pp. 1289-1305.

[7] Kushal Dave, Steve Lawrence & David M. Pennock (2003). Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. In Proceedings of the 12th International Conference on World Wide Web, WWW 03, pages 519528, New York, NY, USA, 2003. ACM

[8] Sharma & Dey (2012). Performance Investigation of Feature Selection Methods and Sentiment Lexicons for Sentiment Analysis. Special Issue of International Journal of Computer Applications, June 2012

[9] Li, P., & Huang, H. (2002). Improved feature selection approach TFIDF in text mining - IEEE Conference Publication. Retrieved from http://ieeexplore.ieee.org/document/1174522/

[10] Wang, N., Wang, P., & Zhang, B. (2010). An improved TF-IDF weights function based on information theory. 2010 International Conference on Computer and Communication Technologies in Agriculture Engineering. doi:10.1109/ cctae.2010.5544382

[11] Martineau & Tim Finin. (2011). Delta TFIDF: An Improved Feature Space for Sentiment Analysis. Association for the Advancement of Artificial Intelligence.