Linköping University | Department of Computer and Information Science
732A92 Text Mining, 6 ECTS
January 15, 2021

Temporal Sentiment Analysis

- Trends in reviews over time

Martin Svensson, marsv079

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

Abstract

Number or reviews a product can get on the internet in an international stage is vast, which is why this has to be automatically checked and interpreted. This report presents a pipeline containing text digestion, sentiment classification and visualization of the reviews over time. Looking at games or other software the reviews can change depending on updates, new content or bugs introduced to the users. To validate the pipeline the author predicts how the reviews change over time for a game using prior knowledge and patch notes. The results shows that common machine learning models can be used with simple text features to make accurate predictions on the data. Machine learning models used are Multinomial Naive Bayes, Linear Support Vector Machines, Logtistic Regression, K-Nearest Neighbors and Random Subsampling. For the text features only Term Frequency tested with unigrams. Further the visualization from the validation data resembles the prediction made before viewing and analysing the validation data.

This report and code produced can be found in the project repository on GitHub.

Acknowledgments

I’m deeply indebted to Zelda, my cat, for her unparalleled support and patience.

Contents

List of Figures

List of Tables

1 Introduction

Opinions are often presented as the count of positive and negative views of the audience. This is common for elections, but also on social media platforms like Twitter or product reviews where it can be used for information retrieval [4]. However, for certain products or situations the opinion might change over time e.g. stock market or presidential elections. This is also more common for software products (e.g. applications, games or services) since these are continuously changed over time [3]. As described in [3] this is an important view of the users and developers for either deeming if the game is a good investment or what issues to prioritize.

1.1 Aim

For this report the aim is to implement an automated approach for performing sentiment analysis on text data in a time-series approach. The outcome is a package with modular components that has the ability to digest text with a certain topic, classify each entry based on sentiment and then visualize with respect to time. It is also of interest to use as simple methods as possible creating a modular framework for future improvement. Usability and understandability over complexity.

This report and code produced will be available in the project repository on GitHub [8], data excluded as described in chapter Data.

1.2 Motivation

This project was chosen to utilize text mining for temporal insights of a topic, which can be important for decision making. This will also result in an usable application that can be used and improved in the future.

1.3 Delimitations

This work is planned to take approximately 90 hours and audience is expected to have basic knowledge in Machine Learning and Text Mining. No ethical or societal aspects is included within the scope.

2 Related Research

In this chapter a short review of related work in the areas of sentiment analysis and classification will be presented. The focus is on the different parts of the project, which is text processing for the models and used models and their results. Since prior knowledge is assumed it will not cover basic evaluation of models or the principle of the common machine learning models.

2.1 Text Processing

In [11] the authors compare different text feature sets and different models and how the features used affect the results. Their research focuses on average improvement using Term Frequency (TF), Term Present (TP) or Term Frequency-Inverse Document Frequency (TF-IDF) and unigrams or bigrams. Results shown are that with the datasets used the difference between unigram-TF and unigram-TP is small, while using bigrams can have 5-10% improvement or perform worse depending on the text data. TF-IDF, however, does only improve over bigrams in certain datasets with certain models. It can be concluded that there are ways to improve the accuracy of models, but that it cannot be said certain which of the models will perform best. Noticeable is that the unigram TF or TP performs well with usually quite high test accuracy. [11]

Using Position of Speech (POS) tagging can depending on the text origin further improve accuracy of the model, as it can capture other dimensions than just n-grams. [5] The authors further discusses the subject that different n-grams are better for different types of texts, like movie reviews, product reviews or social media posts. The paper from [6] also tests different combinations of unigrams, bigrams and POS e.g. emoticons. Their test accuracy for just unigrams is close to the best results of different combinations, being the simplest feature. However, their result is better with unigrams-TP for some models and combination models have better overall performance.

In [9] the authors show that using sentiment-specific word embeddings together with deep learning can increase accuracy compared to non-deep learning models. This also show that more advanced ways of engineering the features could give better results.

2.2 Classification Models and Methods

Throughout the literature review there are some more frequent occurring ML models in sentiment classification. [14] uses Multinomial Naive Bayes, Maximum Entropy (Multinomial logistic regression) and Support Vector Machines (SVM), which is also used by [6]. In the work of [11] Bagging with SVM, AdaBoost and Random Subspaces with SVM are used. The Random Subspace model often has similar accuracy, except for when it is has a quite higher accuracy than the other models. Then Naive Bayes is also used by [5].

Further [2] shows that the concept of Majority Voting could be used to increase accuracy from just using a single model. The authors uses SVM, SVM with Bagging and Naive Bayes models and then combines these into a majority vote classifier. The combined classifier always performs better than each of the stand-alone models. They further encourages the use of what they call Multiple classifier systems for sentiment classifications.

3 Data

The data used is queried from Steam Web API [10] using Python for the games used in the project. Data is stored in parquet file format [1] for efficient storing and good interaction with Pandas. Code used for extracting and storing is available in the project repository [8].

The data from the reviews API consists of a summary of the statistics for the game, information about the author and the review itself with meta data. The full specification is available on Steam review API page [10].

Processing of the data consisted of converting between different data types and creating a consistent dataset from the API. Data types were converted to the most suited for use with the Python packages, such as Scikit-learn, Numpy and Pandas. Mostly this involved datetime convertions.

The data was also stripped of personal information about the author, recommendationid and author, which could easily be used to trace the user accounts. Due to the Term of Use for Steam Web API [7] data is not provided with the project.

The datasets used consisted of 137982 reviews for training and testing from GTA V and 24333 reviews from Wolcen for validation.

4 Method

In this chapter the implementation and methods used will be given. For this work the aim has not been to create the most accurate classifier, but to utilize methods to create a framework that can later be improved.

4.1 Data Selection

The two games selected, for training and test and validation, have different purposes. The game chosen for validation is a game the author has prior knowledge about and is fairly new, meaning it has a limited amount of updates and also reviews. For training and testing a larger dataset is wanted so a more known and older game that is still actively developed was chosen. The language is quite specific and can be unique in the gaming community, which is why another game is used as training corpus. However, review language is also genre specific and the games chosen are from somewhat different genres. Only reviews in English were used for this project.

4.2 Corpus Preparation

Before the reviews were incorporated into a corpus the dataset has to be balanced between the classes, the reviews for GTA V are severely unbalanced. This also reduces the amount of training data, which is helpful for reducing the time taken for training the different models. Data is then divided into training and test data for model selection, with 70% training and 30% test data.

The training reviews are then digested by CountVectorizer from scikit-learn to create the corpus. Term Frequency (TF) of Unigrams was used as it is one of the simplest methods and should produce similar results as Term Present (TP) as described in chapter Related Research. Reviews for testing are then transformed into the corpus created. For validation the complete train/test dataset will be used for training and validated on the full validation set.

4.3 Model Selection

Different models were tested before a selection was made. Since computational resources are not abundant models with fairly low complexity was used, also adhering to best practice to start with simple models. If the simple models, Multinomial Naive Bayes, Linear SVM, Logistic Regression and K-Nearest Neighbors, would produce unsatisfying results further investigation would be required. This would include more precise parametrization of the models or testing more complex models. Above 75% accuracy on the test data would be deemed good enough for a model as this is in parity with results from [11, 14] and this dataset is assumed be fairly simple. For further proof of future improvement a Random Subspace model with SVM base was used with very basic parametrization, due to the complexity of the model.

Models were trained and tested with the confusion matrix and accuracy evaluated for both training and testing data. Then the best models on the test data was also combined into a Majority Vote system as seen in chapter Related Research.

For validation the best outcome from testing would be used as sentiment classifier for the visualization analysis.

4.4 Evaluation

For evaluation the similar Steam Reviews will be used for a game the author has knowledge about. The author will predict the behaviour of the reviews based on knowledge about the game and by reading update notes from the developers to see what issues have been present in the game. Then in this section there will be a prediction made before investigating the reviews and comparing with the true label graphs. This will later be used as a benchmark for the predictions on validation data in section Evaluation.

The game chosen is a PC game named *Wolcen: Lords of Mayhem*, which started as a kickstarter with the name *Umbra*. [12] The game is available at Steam [13] and was released on February 13 2020 after a 4 year early access for a limited audience. The audience had very high hopes for the game from the very successful kickstarter campaign and glimpses of the game during development. However, the release was not as smoothed as anyone could have hoped for with server issues and unstable gameplay. At the same time it was a somewhat new take on the genre and had many positive elements that could evolve into the game that was expected.

Visualization of positive and negative reviews will be used with both the true labels and the predicted labels, along with the confusion matrix and accuracy. Also the top 5 words of each bin can be browsed using the visualization tool.

Review Prediction

The development team then released small patches and hotfixes to the game without any new content for almost a year. The 3 December 2020 a large new content update was released that have been very sought for in the community. This were another implementation of what was promised during the kickstarter campaign along with rebalancing and restarting the game and economy that had been affected by the early bugs at release. During the last month more hotfixes and small patches has been released without any new content.

With the information about the game, over 100 hours of gameplay and looking over the patch notes and updates certain patterns are expected. There has been lots of game breaking issues around release of the game, suggesting that there should be a fair amount of negative reviews. Also looking at where in the development process according to the kickstarter campaign, these are not yet fulfilled at release meaning that content is missing that has been promised.

During a game release there will always be most active players just after release and later if there is large content releases. This is where peaks in the reviews are expected, both negative and positive. There should be several magnitudes of reviews more around release date than around the large content update in December. It is also expected that there are a very low number of reviews in between these releases. Since the content update still not completed the kickstarter roadmap, there is probably a high number of negative reviews at the time for this update.

5 Results

This chapter is divided into results for selecting models and the evaluation of the unseen validation data using those models.

5.1 Model Selection

With the balanced training and test data different models were trained and the results are shown in tables. The best values are marked in bold along with the true numbers for the confusion matrix.

Model	TP	FP	FN	TN	Accuracy
Dummy Classifier (stratified)	24015	24172	24179	24221	0.50
Multinomial Naive Bayes	41980	6207	5356	43044	0.88
Linear SVM (SGD)	40443	7744	3350	45050	0.89
Logistic Regression	**42155**	**6032**	3049	45351	**0.91**
KNN	34689	13498	**2900**	**45500**	0.83
Random Subspaces (SVM)	38322	9865	6028	42372	0.84
True Classification	**48187**	**0**	**0**	**48400**	

Table 5.1: Model result on train data.

Result from training data is listed in table 5.1. For training Logistic Regression (LR) had best precision, as seen with high number of TP and low number of FP. KNN instead has the best negative predictive value, low number of FN and high number of TN. The dummy classifier performs worst with 50% accuracy which is however the perfect score for a balance dataset with stratified dummy classifier. Linear SVM and Multinomial Naive Bayes (MNB) have similar performance where MNB has slightly better values for positive predictions and SVM for negative predictions. Best accuracy is gotten from Logistic Regression with 91% accuracy. The more complex Random Subspace model with very basic parametrization still gets 84% accuracy, about same as KNN and slightly less than the other models.

Model	TP	FP	FN	TN	Accuracy
Dummy Classifier (stratified)	10480	10324	10276	10315	0.50
Multinomial Naive Bayes	**18042**	**2762**	2841	17750	**0.86**
Linear SVM (SGD)	16851	3953	1794	18797	**0.86**
Logistic Regression	17084	3720	2015	18576	**0.86**
KNN	13461	7343	**1709**	**18882**	0.78
Random Subspaces (SVM)	16446	4358	3005	17586	0.82
True Classification	**20804**	**0**	**0**	**20591**	

Table 5.2: Model result on test data.

Using the test data, table 5.2 there is quite a big shift in results, suggesting the different generalizations between the models. The dummy classifier is the same as on the training data, as expected with balanced dataset. Now MNB has the best precision and KNN the best negative predictive value. KNN has the lowest accuracy at 78% of the real models, despite having the best negative predictive value. Overall the accuracy is just a bit lower than the training data with MNB, SVM and LR all on 86% and will be used for the Majority Voting.

Model	TP	FP	FN	TN	Accuracy
Majority Voting	17210	3594	1891	18700	0.87

Table 5.3: Majority Voting on test data.

In table 5.3 is the result from Majority Voting, which was only performed on the test data. Compared to the results in table 5.2 the confusion matrix is a combination of the three models used, while accuracy is slightly improved to 87%. This makes Majority Voting using Multinomial Naive Bayes, Linear SVM and Logistic Regression the most accurate of the methods presented.

5.2 Evaluation

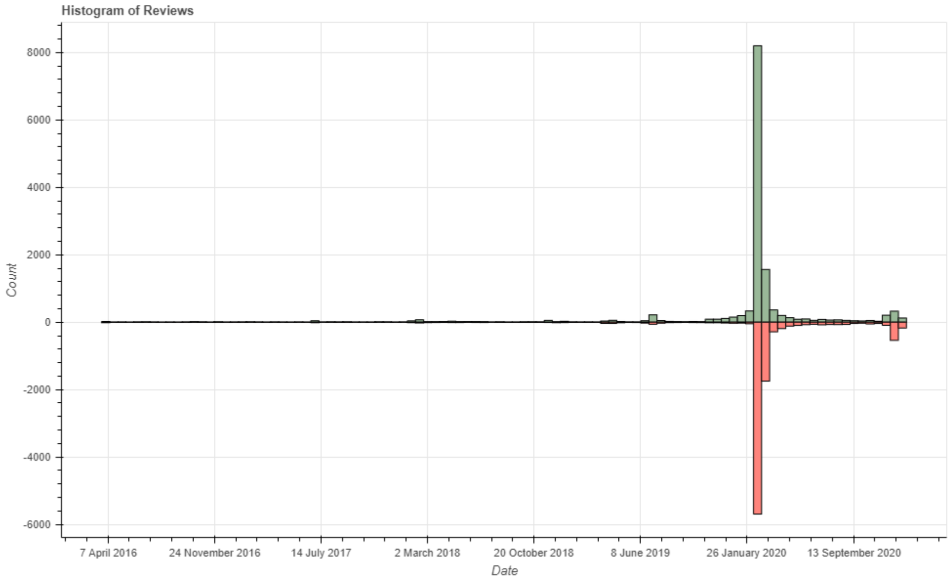
For the evaluation all training and test data is used for training the final models Multinomial Naive Bayes, Linear SVM, Logistic Regression and Majority Voting.

Model	TP	FP	FN	TN	Accuracy
Multinomial Naive Bayes	**8839**	**1828**	2945	10721	0.80
Linear SVM (SGD)	8137	2530	**1746**	**11920**	0.82
Logistic Regression	8342	2325	2092	11574	0.82
Majority Voting	8420	2247	1914	11752	**0.83**
True Classification	**10667**	**0**	**0**	**13666**	

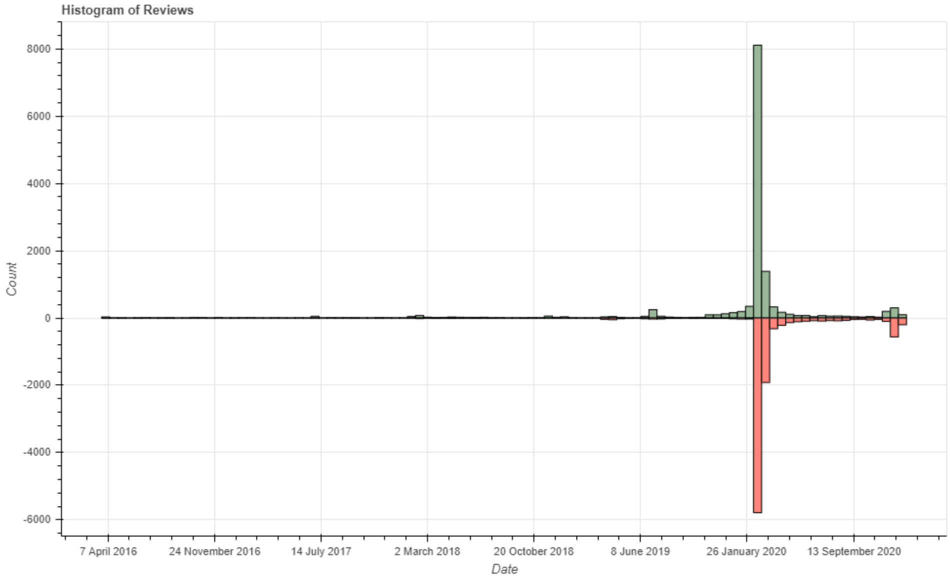
Table 5.4: Model result on validation data.

From table 5.4 it is shown that MNB has best precision while SVM has best negative predictive value. The accuracies for the models are about the same with the combination through Majority Voting being the highest at 83%.

Comparing the predicted, figure 5.1a, with the true labels, figure 5.1b, by plotting positive and negative reviews. There is only a slight difference seen in the comparison due to the scaling. Looking at the axis there is a small difference in maximum negative reviews and a small difference in positive reviews just behind the peak.



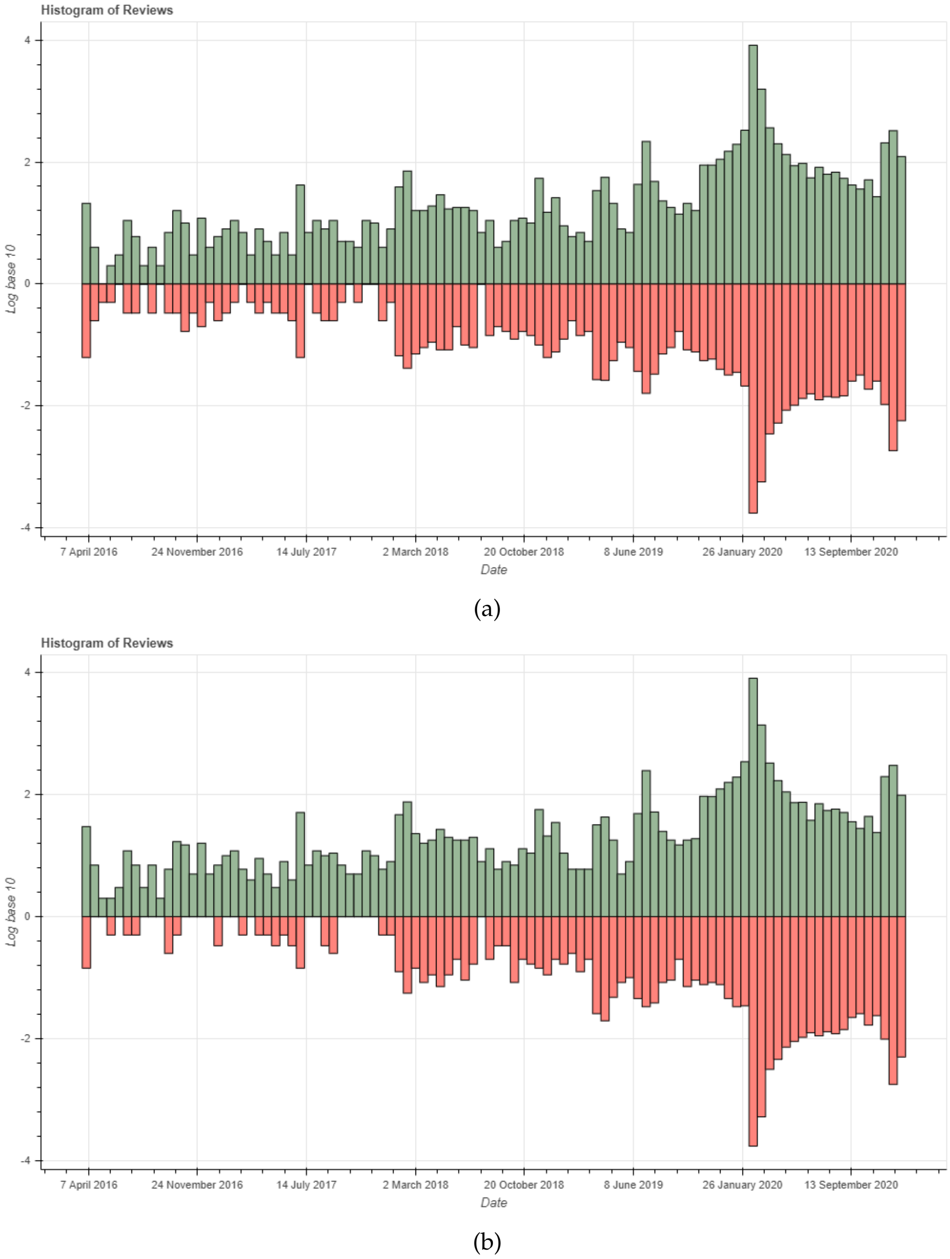
(a)



(b)

Figure 5.1: (a) The predicted sentiment from the reviews. (b) The true labels of the reviews.

If the plots are scaled using log base 10 a larger difference can be observed between the predicted, figure 5.2a, and true figure 5.2a.



(a)

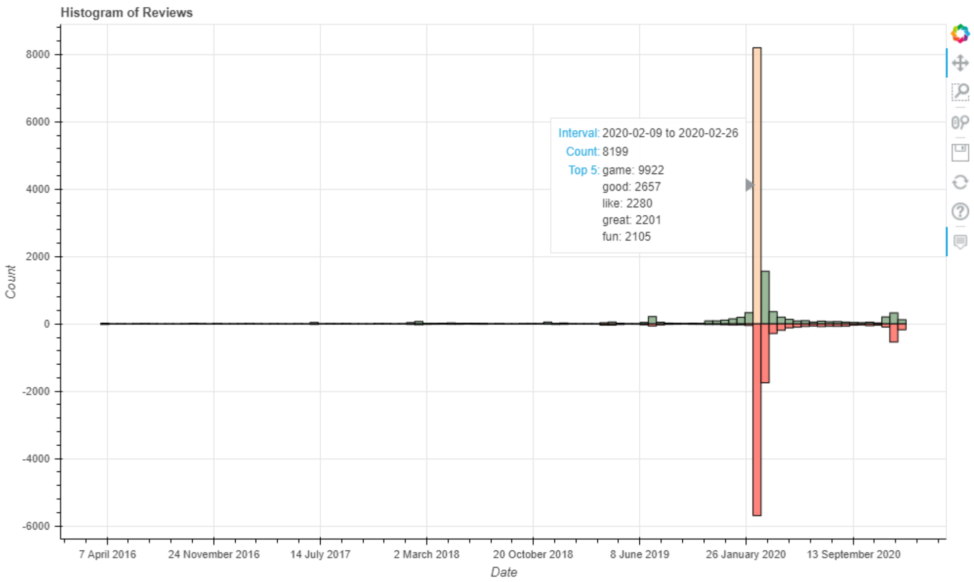
(b)

Figure 5.2: (a) The predicted sentiment from the reviews in log base 10. (b) The true labels of the reviews in log base 10.

Here a few more differences are found due to the scaling, mostly in the left part of the plots with few reviews since this is sensitive to scaling.

Using the tool tips in the hover tool, as demonstrated in figure 5.3, keywords can be extracted from the different bins. This could be useful for a further analysis and see if the top words for positive and negative reviews respectively are changing over time. An example of words found is listed in table 5.5.

Figure 5.3: Plot showing the tool tip when hovering a bin.



Positive	Negative
game	game
good	just
like	play
great	bugs
fun	like
great	time

Table 5.5: Example of most common positive and negative words.

6 Discussion

This chapter analyse the results obtained and the method used, both described in respective chapter.

6.1 Results

The results from training and testing the models are closely related to the results seen in related work, described in chapter Related Research. The K-Nearest Neighbors (KNN) model was not used in any of the referenced works, but was included since it is a common ML model and results in non-linear classification borders. It was unexpected to see that KNN performed so unbalanced, classifying negative reviews so well, but very poor on positive reviews. Also the potential of Random Subspaces (RS) is hinted, it is used in a poorly parametrized way, but the generalization i.e. performance on test data is very good considering. However, since it uses SVM as base model it should still be close to the Linear SVM model that is a more basic model than the SVM used in RS.

The majority voting increased performance as expected from referenced papers in a similar way. It could be argued that this increases complexity over just using one model, but it also balanced the FP and FN classifications. Possibly using majority voting the classifications will be more stable over different datasets, this would require more investigation and could be future work. Using low complexity models made the prediction time fast even when using the voting system.

Validation

Since classifications of the validation dataset was good with over 80% accuracy, the dataset used for training was representative enough. The games, Wolcen and GTA V, are both similar and very different being in different sub-genres. However, both are games and reviews resemble one another but genre specific words are probably not included in training. The result could possibly be improved with a broader set of games used for training.

The models performed otherwise as expected with majority voting having the highest accuracy, just as described in related research. Analysing the plots created had very similar result, since FP and FN is balancing cancelling each other when looking at the figures. It also looks like the misclassifications are distributed along the time axis, not causing any large difference in the overall shapes.

As for the most common positive and negative words they are quite expected. Writing *good*, *great* or *fun* in a positive review or *bugs* in a negative review is probably common. It looks like *just* is very common in negative reviews, which might not be expected. This is probably due to writing *...just like *another game*..* or *...just too many bugs...*, this analysis could also be future work.

The prediction made by the author in Method are quite accurate missing only the fact that the number of reviews are almost zero in-between the game release and release of the new content. At the time of the release of Wolcen the number of reviews are around 8000 positive and 6000 negative. Number of reviews increases just before release and start to decline fast after release down to a few reviews per month. Then the review numbers go up just before new content release, peaks just after the release and then goes down again. More reviews were predicted to happen during content release, but the difference in orders of magnitude was correctly captured.

6.2 Method

The training dataset was around five times as large as the validation set, this was at least in this application good enough based on the results. Training data could be from multiple games, some from a similar sub-genre might increase the performance of the model. Mining the data trough the Steam Web API was good in order to format the data at the same time to a usable format. This also ensured some consistency within the data, no missing data points etc.

For this work the most simple text processing, term frequency, was used since it was good enough as seen in Related Research. However, as the literature review conclude there are much that can be done to increase the accuracy by a considerable amount. With the modular approach this is possible and could also be used with majority voting, using different corpus features.

It is similar with the models used, most of them commonly used in related works but there are suggestions that more complex models and parametrization would increase accuracy. Because of the modular design this is easy to implement and could also be used with majority voting. It was interesting to use KNN and see the results, even though it was not within the commonly used ML models for sentiment analysis.

The evaluation made for the complete application on validation dataset was successful and the aim is completed. Using these kind of end-to-end applications does span over several large fields in terms of literature review and what to cover. Therefore it can be difficult to get any depth in a project with the time limit described in section Delimitations. At the same time the project is exposed to real world scenarios, combining the problem space from industry with academia. With the approach made there are several future work areas available within this package, allowing for much improvement.

A more structured approach for literature study could also change the method used in this report, which could also have changed the outcome. Since popular models were selected from the related research, if all referenced material would change the method would change as well. For a more stable method a larger literature study and as stated a more structured one could render a better quality foundation for this project.

7 Conclusion

The sentiment classification were very similar to the true labels and since some misclassifications are TN or FN, looking at a plot of both is very similar to the true plot. With a high accuracy of 83% on validation data using Majority Voting the concept is working. The outcome is a modular package that can digest text, classify based on sentiment and visualize the results. Both code used and models were simple and would be easy to improve or change for the future. The application is a complete pipeline and fulfils the aim of the project.

For future work all components could be worked with separately or together. With digestion and processing of text testing e.g. Term present, Term frequency–Inverse document frequency or Position of Speech tagging could improve the accuracy. The models have not been tuned, e.g. using grid search for better parametrization, which is suggested to improve performance without using more computational models. Then if there is computational resources other models such as Random Subspaces, Bagging or switching to deep learning models should improve performance. However, it is proved that simple models can give a good result for everyday applications.

Bibliography

[1] *Apache Parquet*. URL: https://parquet.apache.org/.

[2] Cagatay Catal and Mehmet Nangır. "A Sentiment Classification Model Based On Multiple Classifiers". In: *Applied Soft Computing* 50 (Nov. 2016). DOI: 10.1016/j.asoc.2016.11.022.

[3] Dayi Lin, Cor-Paul Bezemer, Ying Zou, and Ahmed E. Hassan. "An Empirical Study of Game Reviews on the Steam Platform". In: *Empirical Software Engineering* 24 (Feb. 2019). DOI: 10.1007/s10664-018-9627-4.

[4] Alexander Pak and Patrick Paroubek. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining". In: vol. 10. Jan. 2010.

[5] Alexander Pak and Patrick Paroubek. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining". In: vol. 10. Jan. 2010.

[6] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. "Thumbs up? Sentiment Classification Using Machine Learning Techniques". In: *EMNLP* 10 (June 2002). DOI: 10.3115/1118693.1118704.

[7] *Steam Web Api Terms of Use*. URL: https://steamcommunity.com/dev/apiterms.

[8] Martin Svensson. *Project GitHub Repository*. URL: https://github.com/MiniDlicious/temporalsentimentproject.

[9] Duyu Tang, Furu Wei, Nan Yang, Ming Zhou, Ting Liu, and Bing Qin. "Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification". In: vol. 1. June 2014, pp. 1555–1565. DOI: 10.3115/v1/P14-1146.

[10] *User Reviews Steamworks Documentation*. URL: https://partner.steamgames.com/doc/store/getreviews.

[11] Gang Wang, Jianshan Sun, Jian Ma, Kaiquan Xu, and Jibao Gu. "Sentiment classification: The contribution of ensemble learning". In: *Decision Support Systems* 57 (Jan. 2013). DOI: 10.1016/j.dss.2013.08.002.

[12] *Wolcen: Lords of Mayhem on Kickstarter*. URL: https://www.kickstarter.com/projects/wolcenstudio/umbra/description.

[13] *Wolcen: Lords of Mayhem on Steam*. URL: https://store.steampowered.com/app/424370/Wolcen_Lords_of_Mayhem/.

[14] Rui Xia, Chengqing Zong, and Shoushan Li. "Ensemble of feature sets and classification algorithms for sentiment classification". In: *Inf. Sci.* 181 (Mar. 2011), pp. 1138–1152. DOI: `10.1016/j.ins.2010.11.023`.