Steam Games Reviews NLP analysis and ML modelling

by Taras Ivaniv

1. Introduction

The ever-growing volume of user-generated text content on digital platforms has made Natural Language Processing (NLP) a crucial tool in understanding public opinions trough statistical tools. One such domain where NLP is particularly valuable is the gaming industry, where reviews and community posts provide critical insights into player satisfaction, game quality, and industry trends. With the expansion of online game stores such as Steam or Epic Games, vast amounts of textual data in the form of user reviews, community posts and chats are continuously generated, making it a rich source for text mining applications and insightful business decisions.

The dataset under analysis consists of user reviews for a list of both popular and more niche games from the online game retailer Steam by Valve. The dataset contains user reviews with unique review ids and a Steam-generated label of the review sentiment (positive/negative). By employing techniques such as tokenization, lemmatization, and TF-IDF vectorization, we aim to transform raw text into a format suitable for machine learning algorithms.

This report outlines the methodology used to preprocess the data, visualize key features, and train multiple supervised models to achieve satisfactory classification accuracy of positive and negative reviews. The project's objectives include:

- **Data Preprocessing**: cleaning and transforming the text data into a structured format using NLP techniques.
- **Data Visualisation**: analysing word and bigram frequency distributions across different aggregations of the pre-processed dataset.
- **Model Development and Evaluation**: training and evaluating supervised classification model suitable to the dataset.

The significance of this study lies in its potential to enhance the insights from automated review processing and classification, aiding both developers and consumers in decision-making processes. For game developers, understanding sentiment trends, and especially topic based sentiment trends, can drive product improvement and targeted marketing strategies. For consumers, sentiment-based insights can serve as a guide to making informed purchasing decisions and can be further explored to develop a continuous score metric for a Product recommendation algorithm.

---

2. Methodology

This section details the machine learning models, and analytical techniques employed for the review classification task. The methodology focuses on supervised learning approaches optimized for text classification, with comparisons between model architectures and their technical trade-offs. All models were implemented using scikit-learn (as shown in `Lab1_Modelling.ipynb `).

2.1 **Model Architecture**

Three distinct model architectures were selected and evaluated though hyperparameter optimization.

A. Logistic Regression

The first model chosen is Logistic Regression as it is generally a good baseline model for text classification. This is because it is a highly interpretable and efficient model that performs effectively with highly dimensional sparse data such as the result of TF-IDF Vectorization. Hyperparameter tuning of the model mainly consisted of selecting a max_iter in the range 1k-10k. The model offers a parameter to balance class weights which was included in the hyperparameter selection as the reviews presented a heavy prevalence of positive sentiment. Unfortunately the main drawback of this model is its susceptibility of unbalanced label distribution, which is noticeable in the model evaluation as it struggles to predict negative reviews as accurately as positive ones regardless of the implementation of class weight balance. Moreover, the model assumes linear separability between classes which is not always easily verifiable and can lead to significant struggles with nuanced sentiment expressions.

B. Complement Naive Bayes

To tackle the main issue of the Logistic Regression model we turned to Complement Naive Bayes Classifier, a model that in classification of datasets that present heavily unbalanced labels. Once again, the model is also well performant on high dimensional sparse data but in addition it has a relatively fast training time of *O(n_features),* emerging as the quickest training model. The hyperparameter tuning of the model was focus on the Laplace smoothing 'alpha' which was set to a range between 0.1 and 10.
While introducing a solution to unbalanced labels the model also assumes strong feature independence. That is a big disadvantage in the study framework as it means that any context of the phrase is lost in training leading to an underperforming model.

C. SGD Classifier (Log Loss)

The Stochastic Gradient Descent is a memory-efficient linear solver with elastic-net regularization capabilities. This means that depending on the selected loss function, the SGD can fit different types of linear models such as Linear and Logistic Regression, Support Vector Machines both in their traditional setting and the squared variant which is quadratically penalized. We decided to fit the model with a log loss function, offering an alternative solver for a logistic regression classifier. The main advantage of the model is its ability to handle large feature spaces better and in a more memory-efficient manner. This means we should be able to obtain similar performance to model A while improving computational time. Hyperparameter tuning of the model was mainly focused on the regularization strategies and strength, maximum number of iterations and initial learning rate value. While the training performance beats all other models the accuracy and evaluation metrics are just below what was possible to achieve with the linear model, showing how careful iteration tuning is necessary to prevent premature convergence with these models.

2.2 Validation Protocol

To evaluate the model performance and pick the best hyperparameters the following performance assessment steps were implemented:

1. **Randomized Hyperparameter Search**: 25 iterations per model with 5-fold cross-validation
2. **Stratified Accuracy Metrics**: ROC-AUC scoring prioritized due to class imbalance

3. **Diagnostic Visualization**: Confusion matrices and class-specific recall analysis

2.3 Considered Alternatives

To handle some of the issues presented by our selection of models the following alternatives can be considered in a future follow up to the study:

- **Neural Architectures**: Recurrent networks were excluded due to hardware constraints and dataset size limitations; however, they can be tuned more precisely to better estimate a review sentiment.
- **Ensemble Methods**: Random Forests were discarded from the initial analysis due to high computational requirements and a natural tendency to overfitting.
- **Transformer Models**: transformer model such as BERT are expected to perform significantly better due to their architecture and most importantly their text representation algorithms which in some cases allows for little to none of the context to be lost.
- **Metadata Integration**: Ownership tiers were pre-processed but excluded from final models to isolate text features. The models could be further improved across the board thanks to additional review information available directly from the steam API, such as purchase type, gameplay hours at reviews and to date and additional user review metrics.

Finally, the conscious exclusion of oversampling techniques preserves real-world class distributions at the cost of recall performance for minority classes.

---

3. Dataset description

The dataset from this study was originally published on Kaggle by *"Filip Kin"* with the name "*Steam Review & Games Dataset*". The repository contains two tables:

- The first table contains the list of all the games referenced in the reviews alongside with some additional information such as a range representing estimated game owners.
- The second table contains a total of 201151 user reviews uniquely labelled. Each review is associated with a user ID and a binary variable indicating whether the content is positive of negative. Finally, each review also references the game ID.

This is what the reviews table presents like:

ID	APP_ID	CONTENT	AUTHOR_ID	IS_POSITIVE
181331361	100	At least its a counter strike -1/100	76561199556485100	Negative
180872601	100	Uh... So far my playthrough has not been great...	76561199230620391	Negative
177836246	100	Better mechanics than cs2	76561198417690647	Negative
177287444	100	buggy mess and NOT fun to play at all	76561199077268730	Negative

The dataset should have only contained English user reviews, however, upon further inspection the dataset contained numerous non-English reviews in addition to a subset of "empty" content. For this reason, it was necessary to filter the original dataset by removing empty values and strings containing less than 10 characters. This is because in the following step the `langdetect` library was used to correctly identify English only reviews, returning 131208 English reviews which were sampled 3 times and manually inspected to verify that the language filtering process was successful.
For this reason, a decision to expand the dataset of reviews was made by leveraging the Steam API. This allowed to collect recent data (Jan 3rd-31st 2025) on the same subset of games, extending the total number of observations with up-to-date user reviews. Since the main Kaggle dataset was created in mid-late December 2024 no concerns as to deduplications arose. The additional dataset then underwent the same preprocessing stages before being saved for later merge and use in the dataset analysis and modelling sections.
The final pre-processed corpus of the study contained 384931 game reviews with 138238 unique terms and an average of 20.0 tokens per review.

4. Text cleaning and Preprocessing

Since both study dataset underwent the same cleaning and preprocessing procedures these will be explained once in the following section.
After removing Null and extremely short observations (<10 characters) both datasets were processed by 'langdetect' to select English comments. This was necessary since not only the original dataset contained non-English comments but also the data directly collected from the steam API specifically requesting English only comments contained mislabelled data.

After the datapoints of interest were correctly identified, text data was processed using NLP techniques implemented in as follows:

1. Lowercasing the text (using `.lower()`) and Removing punctuation and numbers (using re.sub() with a specific regex expression)
2. Fixing misstyped contractual forms (i.e. cant →can't, im → i'm) by using the ` `contractions.fix()` from the contractions library.
3. Tokenising the text strings using a word tokenizer from `nltk`.
4. Removing stop words (using the stop word dataset included in `nltk`) while lemmatizing the remaining tokens.

The preprocessing function was then applied to the datasets storing the cleaned content and a list of pre-processed word-tokens.

An inner merge was also performed between both datasets and the games.csv table in order to store all relevant information in a single table. Since a range of game owners was provided and not much additional information was given, for each game an "average owners" value was computing by averaging the extremes of the range.

Both datasets were then separately stored as 'preprocessed_reviews_games.csv' (output to 'Lab1_preprocessing.ipynb`) and 'preprocessed_validationset.csv` (output to `Lab1_DatasetExpansion.ipynb`). Since no temporal overlap can occur the data was then loaded and merged in 'Lab1_CorpusAnalysis' outputting the 'merged_data.csv' dataset.

Follows a table showing the final pre-processed structure of the analysed dataset:

Variable Name	Dtype	Description
id	Integer	The id column contains the unique identifier for each review.
app_id	Integer	The app id column contains the unique identifier of the game reviewed. It is matched with the appid in the games table to get additional game information.
content	Object: String	The content column contains the original unprocessed and unfiltered content of the reviews.
author_id	Integer	The author id column contains a unique identifier for each user. It can be used in later research to develop a recommendation engine.
score	Integer: [0, 1]	The score column represents the Negative/Positive (0, 1) labelling of the content provided by Steam API.
name	Object: String	The name column contains the name of the game reviewed.
average_owners	Integer	The column average owners contain the estimated average owners of the game computed from the game owner range provided.
tokens	Object: List of Strings	The column tokens contain a list of preprocessed tokens (words) from the original review content.
cleaned_content	Object: string	The column clean content contains a string of the fully preprocessed reviews content.

5. Explanatory Data Analysis

The first topic of interest in the dataset is the distribution of the review labels which will later be used for sentiment classification with ml models. As it can be observed in **Figure 1**, the data seems to heavily over-represent positive reviews. It is not known whether the original dataset publisher applied any specific transformations or filtering so we will assume that this distribution represents a real-world scenario where customer sentiment on a category of products tends to be positively skewed.

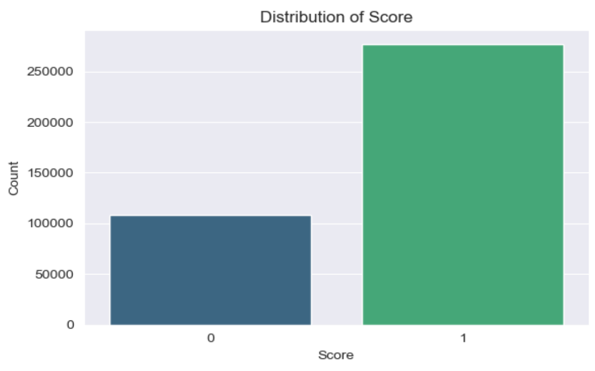

Figure 1

5.1 Token Frequency and Filtering

Preliminary analysis of the pre-processed dataset identified several high-occurrence tokens; words such as “game,” “fun,” “time,” “like,” “play,” “get,” “good,” and “one” appear in over 10% of reviews. Moreover, it was discovered that a high amount of extremely low-occurrence tokens were present; upon further inspection it turned out that most of these terms were caused by user typos or niche terms.

Given the findings some additional preprocessing for better analysis and modelling was needed. The key steps of data preprocessing for model use are:

- **Token Filtering**: by implementing a dual-threshold filtering method we were able to remove uninformative terms:
  - *Rare terms*: tokens appearing in less than 0.05% of reviews were removed after analysis revealed they mainly consisted of unique user typos or uncommon slang terms.
  - *Ubiquitous terms*: tokens appearing in more than 7% were removed as they were comprised of generic overly occurring terms common in game reviews.

  This process yielded significant dimensional reduction results as the number of unique tokens was decreased by 93.56% (from 138238 to 8897 tokens).

- **Ownership Tiering**: Games were categorized into Low/Medium/High popularity tiers using 33rd/66th percentile splits of owner counts

5.2 Visualisation and Statistical Measures

A series of visualisations were generated to further explore the properties of the dataset.

Visualising token frequency distribution in **Figure 2** highlights a still strong presence of uncommon terms. This can be due to the dataset nature, as it is not uncommon to find "gamers" using terminology that colloquially and academically would not be understood by the public. It is for this reason that such skewness is to be expected and if not severe can be accepted for further analysis.

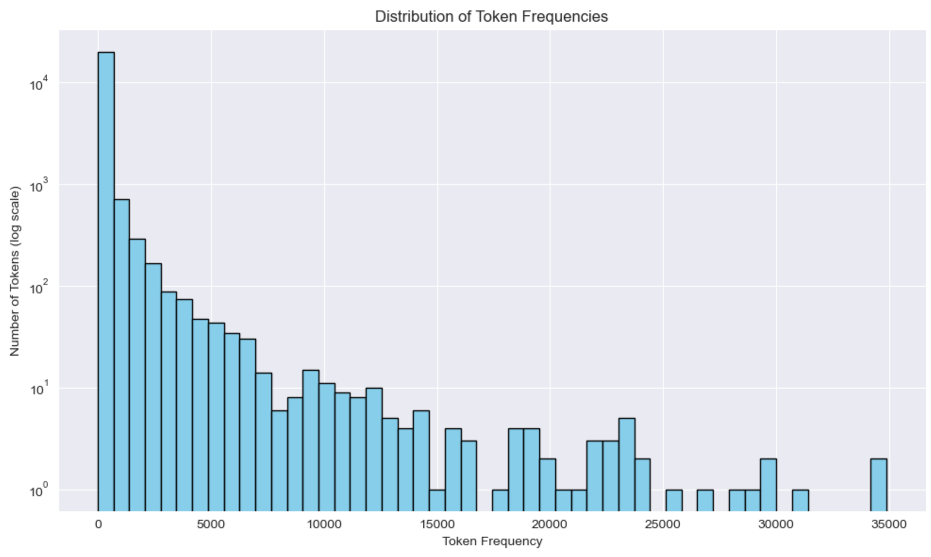

Figure 2

To analyse the general behaviour of the language corpora we leverage Zipf's Law, a well-documented empirical phenomenon in quantitative linguistics which posits that the frequency of a word is inversely proportional to its rank in the frequency distribution of a corpus. In simpler terms, the most common word in a language appears approximately twice as often as the second most common word, three times as often as the third, and so on. We analysed token frequencies by ranking them from the most to the least frequent. When plotted on a log-log scale (**Figure 3**), the relationship between token frequency and rank approximates a straight line—a characteristic signature of Zipf's law. This linearity on logarithmic scales confirms that a few tokens (such as "game," "fun," and "play") dominate the

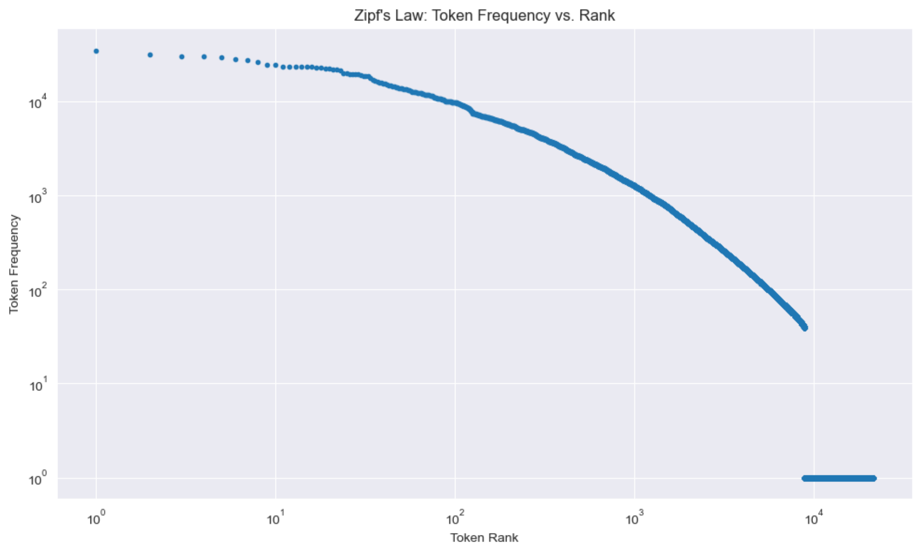

Figure 3

corpus, while a long tail of tokens appears infrequently. The heavy-tailed distribution also confirms the presence of uncommon terms already highlighted by the token distribution plot.

The issues brough up by the two plots may lead to high-frequency tokens overshadowing less common but potentially informative words. Therefor techniques such as TF-IDF weighting can be crucial to help balance the representation of words based on their importance across the corpus.

To visualise in and immediate way most frequently occurring terms we resorted to a word cloud (**Figure 4**).

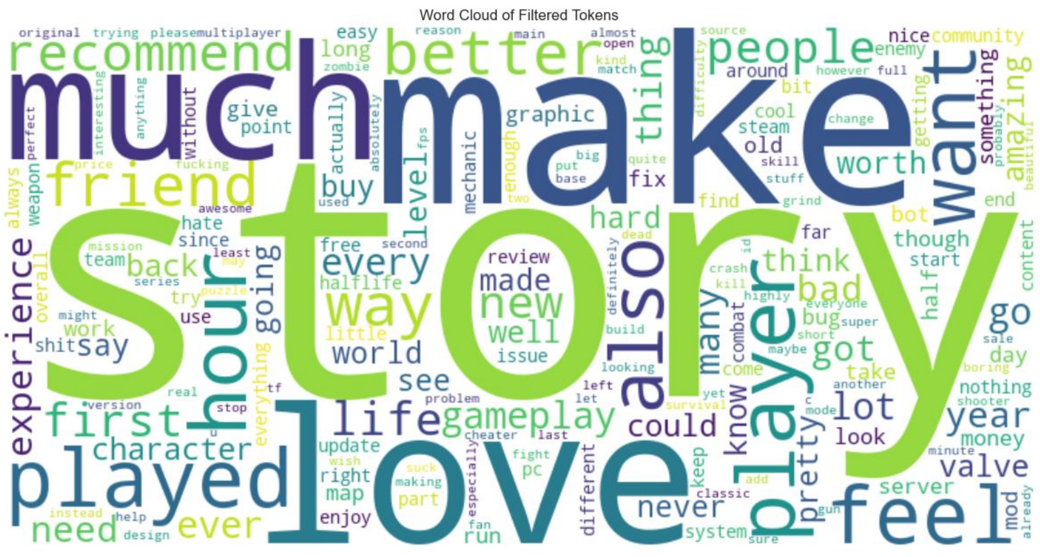

Figure 4: Word cloud of most used terms.

Some terms typical of the gaming community can be spotted, such as “gameplay”, “bot”, “update”, “story” and others. Moreover, it is interesting to note how the company owning the platform (“valve”) is often mentioned, probably because of being also game publishers of incredibly popular products such as “Counter Strike”, “Team Fortress” and “Half Life”.

The analysis was then extended to term popularity across reviews with differing sentiment. To investigate commonly used words by users leaving positive and negative reviews two plots

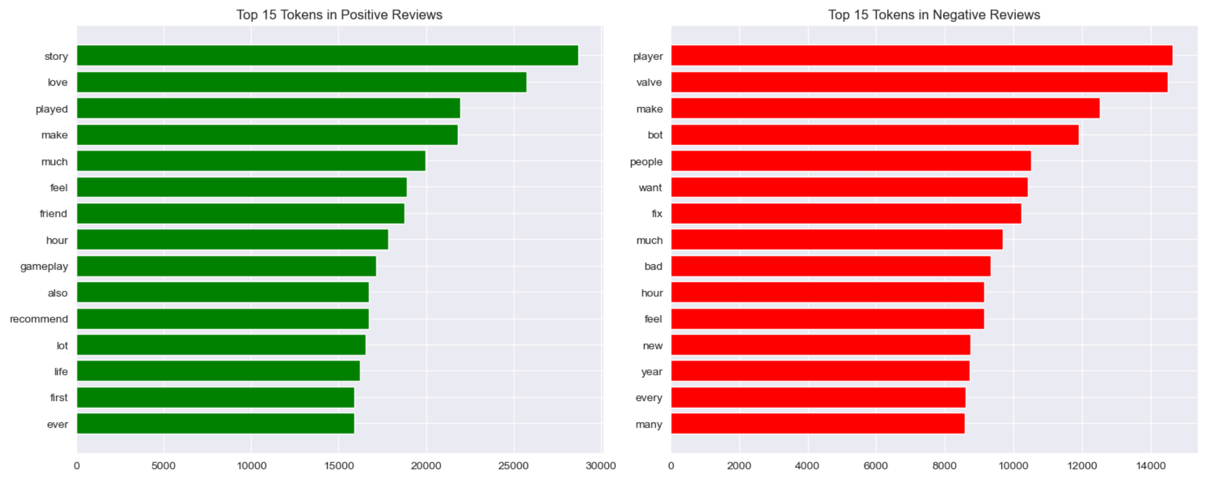

Figure 6

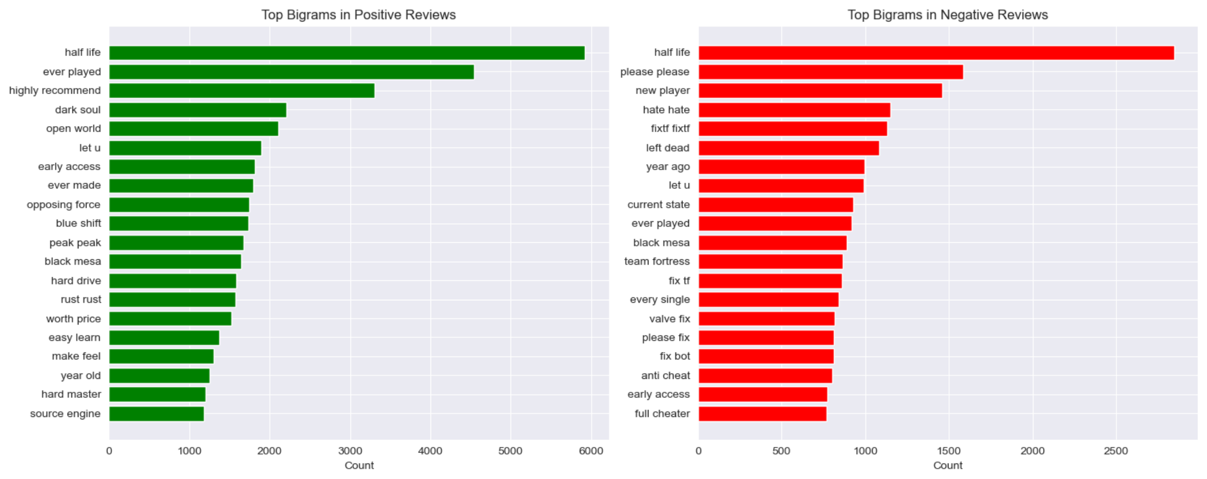

Figure 5

were developed: a plot of most common words grouped by sentiment (**Figure 5**) and a plot of most common bigrams (pairs of words occurring in sequence) in **Figure 6**.

A common element that will show up in other plots as well is the extremely high occurrence of the bigram “half life”, a popular game series developed by Valve which is considered one of the most influential game series for modern gaming. Another commonly mentioned game is “team fortress”.

In general, it can be said that users leaving positive reviews tend to focus mainly on gameplay and story related topics, which can be seen by the use of words as “story”, “played”, “gameplay” and bigrams such as “ever played”, “open world”, “make feel”. Another interesting pair of words present in positive reviews is “easy learn”, which indicates that a leaner learning curve is generally well received by customers. Negative reviews tell a completely different story, focusing on aspects related to game issues (“fix”, “bad”, “please please”, “current state”) and multiplayer issues in competitive games such as bots and cheaters (“bot”, “player”, “anti cheat”, “full cheater”, “fix bot”).

Given the availability of ownership information, an additional set of plots was introduced to study term usage in different levels of “popularity”. The plots in **Figure 7** showcase the distribution of bigrams across different ownership tiers.

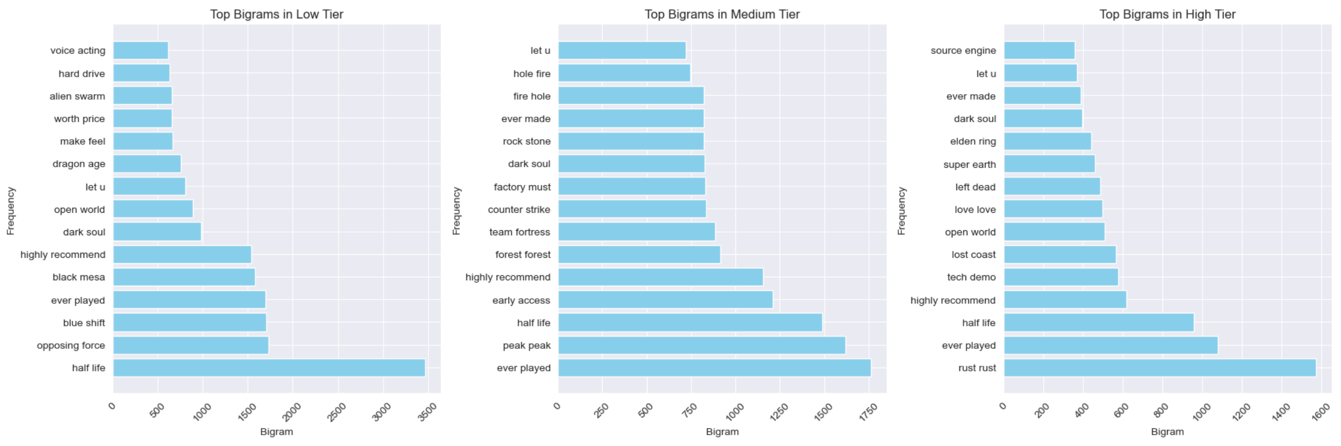

Figure 7

Notable differences in term usage can be highlighted; reviews for Low Ownership tier games seem to be mostly positive, containing terms such as “worth price” and “highly recemented”. Medium and High ownership tier game reviews seem to maintain a similar overall sentiment but game-specific terminology for popular releases emerges (“forest”, probably referring to “The Forest” game, “Half Life”, “Elden Ring”, “Rust”, “Counter Strike”).

Overall, the analysis yielded high dimensionality reduction while maintaining a high level of variety across the different segments of user reviews analysed.

6. Model Results and Discussion

The following section summarizes the outcomes of our modelling experiments and provides an in-depth discussion of their implications, limitations, and directions for future work.

6.1 Results

Our experiments leveraged three supervised learning models—logistic regression, Complement Naive Bayes, and an SGD classifier configured with logistic loss—each embedded in a TF-IDF pipeline and rigorously tuned through randomized hyperparameter searches.

The method of representation of the pre-processed content picked was Term Frequency-Inverse Document Frequency (TF-IDF). This is a numerical statistic that reflects the importance of a word within a document relative to a collection of documents; to do so it increases with the number of times the term appears in a document (single user review) but is offset by how commonly used the word is across the corpus of all review. These two terms are respectively the term frequency and the document frequency and the TF-IDF transformer can be described by the following equation:

$$TF - IDF(t, d, C) = TF(t, d) \times log(\frac{N}{DF(t, C)})$$

Where t is the term, d is the document and C is the corpus; TF(t, d) represents the term frequency of t in document d and finally DF(t, C) represents the document frequency of t in C.

Logistic Regression results:

Label	Precision	Recall	F1-score	Support
0	0.71	0.81	0.76	21480
1	0.92	0.87	0.90	55506
Weighted Avg	0.86	0.85	0.86	76986

Accuracy: 0.85

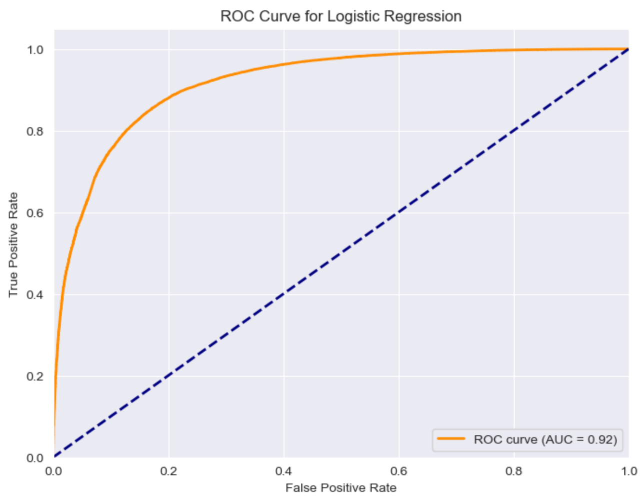

Figure 8

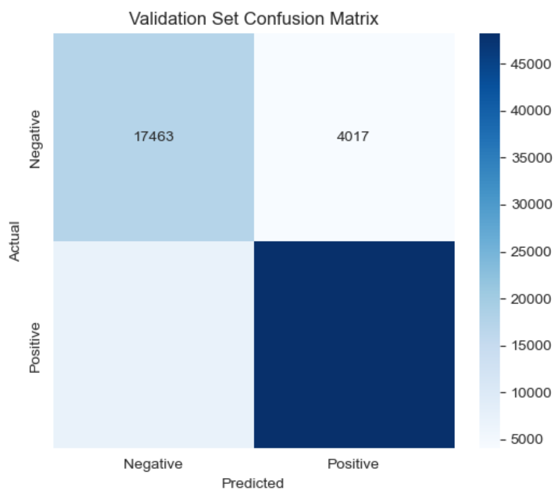

Figure 9

The logistic regression classifier was able to achieve a validation set accuracy of approximately 85.3% after extensive hyperparameter tuning and selection using Random Search Cross Validation from sklearn. The performance of the model is balanced across classes, recording a high precision and recall for positive class prediction and adequate performance on the negative class. It is however noticeable that the skewness in the distribution of labels in the training set caused the model to excel at identifying positive reviews rather than negative ones. The ROC curve for the model shows strong discriminative abilities, confirmed by the high AUC value.

Complement Naive Bayes results:

Label	Precision	Recall	F1-score	Support
0	0.61	0.79	0.69	21480
1	0.91	0.80	0.85	55506
Weighted Avg	0.82	0.80	0.81	76986

Accuracy: 0.80

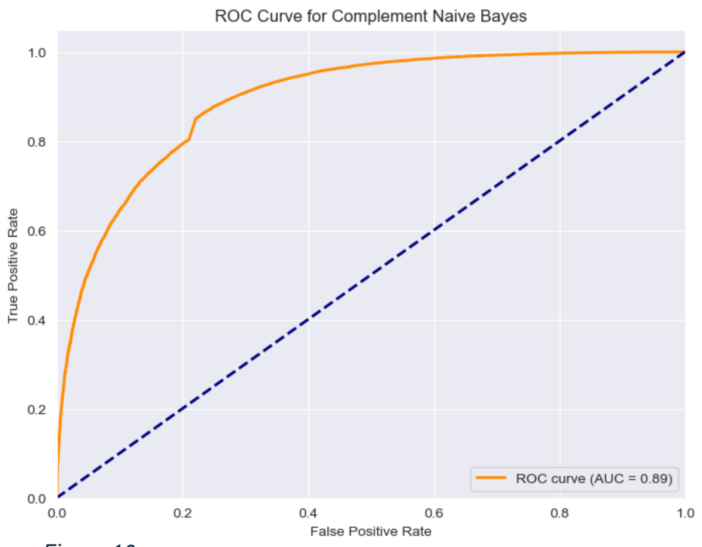

Figure 10

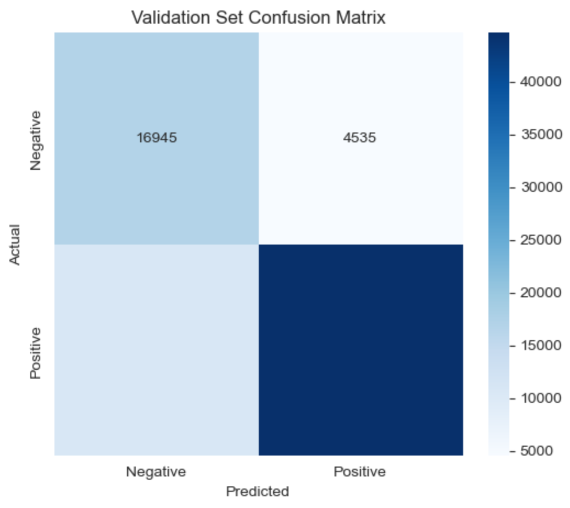

Figure 11

The Complement Naive Bayes achieved an accuracy of 80% on the validation set, showing struggles in the correct identification of negative reviews despite being a model that should be able to easily handle unbalanced labels. During hyperparameter tuning it emerged that a relatively high smoothing parameter was required for optimal performance. This adjustment, while talking issues regarding data sparsity inherent to short review texts in TF-IDF representation, also caused the precision and recall for negative class to be impacted. The ROC curve shows an acceptable discriminant ability with room for improvement as confirmed by the AUC.

Stochastic Gradient Descent Classifier with Log-Loss results:

Label	**Precision**	**Recall**	**F1-score**	**Support**
0	0.85	0.52	0.65	21480
1	0.84	0.96	0.90	55506
Weighted Avg	0.84	0.84	0.83	76986

Accuracy: 0.84

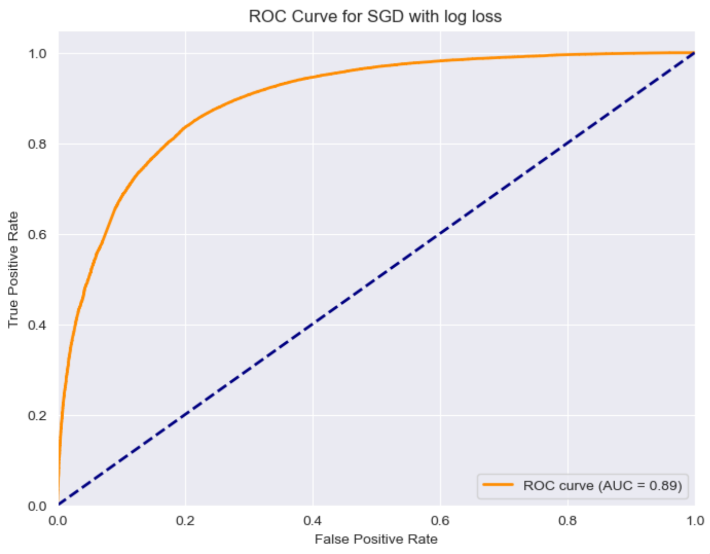

Figure 12

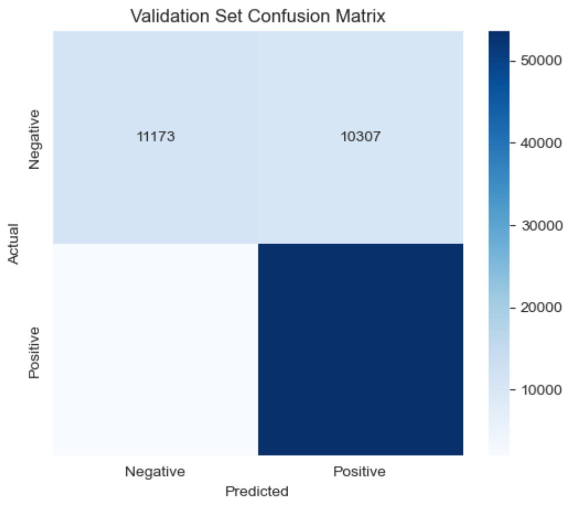

Figure 13

The SGD Classifier with log loss managed to achieve an accuracy of 0.84 on the validation set, while showing worrying performance issues in the prediction of negative reviews. This is again due to the labels distribution imbalance which is not tackled in this model as setting up the model to automatically balance labels does not yield performance improvements. While showing competitive performance with the traditional logistic regression approach, the model requires further study and tuning to outperform the latter. Once again, a satisfactory ROC curve and AUC value can be observed, suggesting the model still has an effective discriminant ability.

6.2 Discussion

The evaluation outcomes reveal a clear stratification in the model performance that could be licked both to technical configurations and inherent dataset challenges. The top performing model is the Logistic Regression, largely due it its robust implementation of class imbalance handling trough the incorporation of balanced class weights. This approach also minimized misclassifications and yielded the best F1-Scores, suggesting that the final model configuration was well-suited for the task given the dataset of reviews.

A different story is told by the results of Complement Naive Bayes (CNB), which faced significant challenges posed by the sparse feature space typical of a text data analysis framework. Despite the simplicity and computational efficiency of CNB it was not able to outperform any other model and required a high smoothing parameter (alpha) to handle the sparsity of the data, which stabilized model performance at the cost of possibly underfitting certain terms. Notable is the performance gap in precision and recall for negative reviews, indicating that a simple probabilistic model struggles to balance bias and variance when preprocessing and text representation decisions are not as finely calibrated.

The performance of the SGD classifier illustrates the trade-offs between scalability and fine-grained classification performance, with an accuracy nearly matching the one of logistic regression, it still demarks a strong discrepancy in its recall metric for the negative class. This indicated that further finetuning of hyperparameters may be required to achieve optimal performance across classes. The overall performance of the models suggest it is suitable for large-scale applications due to its memory efficiency, with a possibility to outperform other models presented with some additional hyperparameter tuning. One promising avenue for future research might involve the integration of ensemble techniques, wherein the complementary strengths of logistic regression and SGD are combined. Such an approach could potentially mitigate the weaknesses observed in the individual models.

A common element connecting the models was the critical role of feature transformation via TF-IDF vectorization. Extensive analysis and hyperparameter tuning revealed the choice of parameters such as maximum vocabulary size, n-gram range and document frequency thresholds have a great impact on model efficiency. Parameters that yielded near optimal results on some models turned out to be suboptimal for other models, highlighting the importance of a tailored model selection process based on the specific characteristics of the dataset.

Beyond the immediate performance indicators, these findings hold broader implications for applications involving sentiment classification in textual reviews. While it is true that logistic regression presented with the best balance of performance and computational efficiency, it is undeniable that the nuanced behaviours observed in the CNB and SGD models indicate that there is still ample room for improvements. Future work could explore more sophisticated text representation models such as BOW (Bag of Words) or Word Embeddings to improve the representation of the data. Moreover, it should be explored in Transformer based models such as BERT could handle the data sparsity and class imbalance better than the presented models. Finally, examining alternative feature engineering methods and perhaps the inclusion of additional variables from the Steam API could further improve model robustness and bias mitigation.

The modelling phase indicates that even standard machine learning algorithms can yield competitive performance on sentiment classification given proper attention to pre-processing, parameter tuning and model selection. While logistic regression is currently the best of all the techniques tried, the learning from the comparison between Both Complement Naive Bayes and SGD points to future opportunities. The detailed results and further discussions here not only authenticate current work but also provide a solid basis for more advanced future works.

---

7. Summary and Conclusion

This study highlights the importance of NLP and ML tools when it comes to extracting information on an extensive customer base such as the one present on steam. The experiment presented 3 solutions to predicting user sentiment in game reviews trough the application of common statistical models, achieving satisfactory classification metrics across the board. It is to be noted that room for improvement is always present, whether trough the application of more advance modelling techniques such as Transformer based architecture or trough more attentive and detailed data preprocessing and feature extraction. Moreover, limitations in the dataset structure were highlighted and possible bias due to the selection of games could be further inspected to combat class imbalance in further studies.