Unlocking Game Success: *Can We Predict Game Reviews from Key Attributes?*

Ananya Achan, Elaine Keung, Mia Kobayashi, Francisco Munoz

Research Question

Can we predict the review tier of a game based on its features such as price, genres, publishers, and other relevant features?

Interest: Guidance for developers to create well received games with respect to metadata (writing descriptions), discount strategy, etc… based on existing “successful” games; exploring consumer habits towards game preference.

95 - 100	500+ reviews	positive	overwhelming
85 - 100	50+ reviews	positive	very
80 - 100	1+ reviews	positive	
70 - 79	1+ reviews	positive	mostly
40 - 69	1+ reviews	mixed	
20 - 39	1+ reviews	negative	mostly
0 - 19	1+ reviews	negative	
0 - 19	50+ reviews	negative	very
0 - 19	500+ reviews	negative	overwhelming

Data

Date: 19 May 2024

Source: Kaggle. Steam Store: a site dedicated to “playing, discussing, and creating games”

Rows: 42,496

Columns: 24

Total number of non-null values in column

		Non-Null Values
	content_descriptor	2,375
	discount_percentage	4,859
	original_price	4,859
	recent_review_%	5,503
	recent_review_count	5,503
	recent_review	5,503
	overall_review	40,020
1	overall_review_count	40,020
2	overall_review_%	40,020
3	discounted_price	42,257
	publisher	42,286
	developer	42,307
4	about_description	42,359
5	genres	42,410
6	release_date	42,440
	categories	42,452
7	awards	42,497
	app_id	42,497
	mac_support	42,497
	win_support	42,497
	dlc_available	42,497
	title	42,497
	linux_support	42,497
	age_rating	42,497

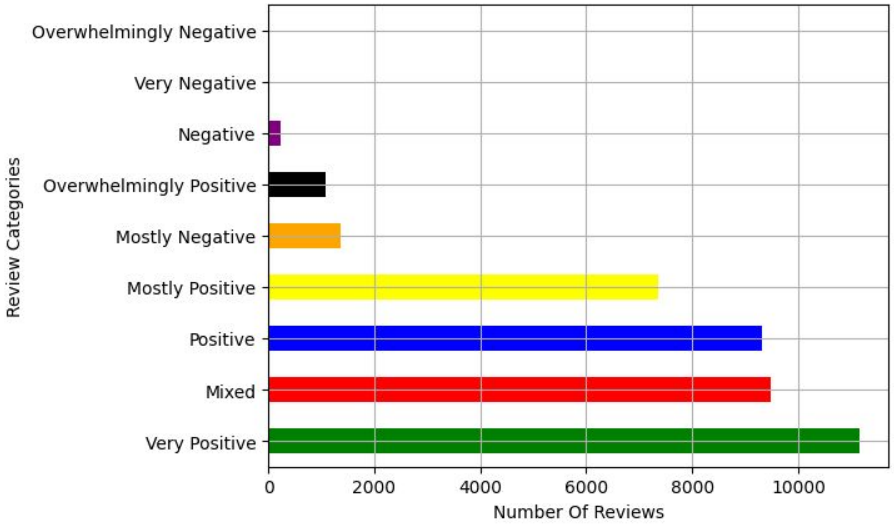
Percentage of null values in column

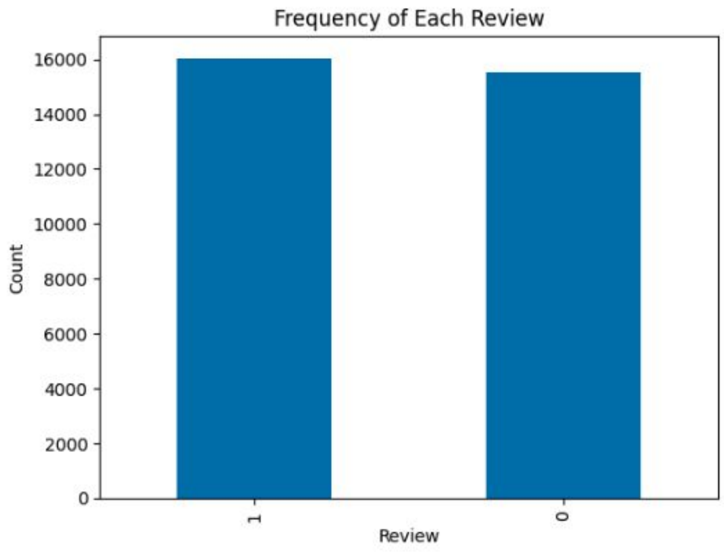
	Missing Values	Percentage
app_id	0	0.00%
awards	0	0.00%
linux_support	0	0.00%
mac_support	0	0.00%
win_support	0	0.00%
dlc_available	0	0.00%
age_rating	0	0.00%
title	0	0.00%
categories	45	0.11%
release_date	57	0.13%
genres	87	0.20%
about_description	138	0.32%
developer	190	0.45%
publisher	211	0.50%
discounted_price	240	0.56%
overall_review_count	2,477	5.83%
overall_review_%	2,477	5.83%
overall_review	2,477	5.83%
recent_review	36,994	87.05%
recent_review_count	36,994	87.05%
recent_review_%	36,994	87.05%
discount_percentage	37,638	88.57%
original_price	37,638	88.57%
content_descriptor	40,122	94.41%

Summary of Data

	discounted_price	awards	age_of_game	overall_review_%	overall_review_count
mean	355.76	0.33	4.68	77.17	2499.93
std	448.47	1.29	3.30	17.60	49201.71
median	250	0	4	81	59.00
min	0	0	0	0	10.00
max	8325	41	27	100	8062218

Summary of Review Class





Features of Interest

genres: List of genres the game belongs to (one hot)

release_date (age): Date the game was published

awards: Number of awards the game has received

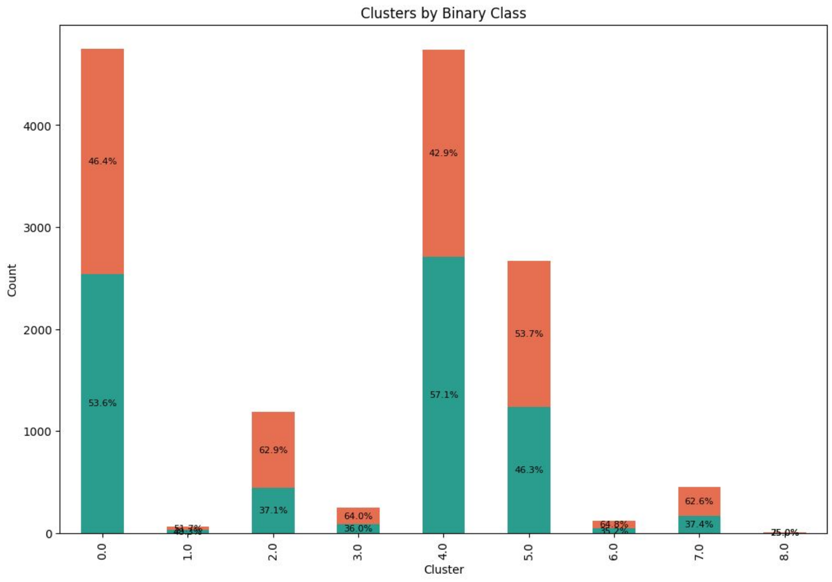
discounted_price: Price after the discount (as of the time the data was scraped)

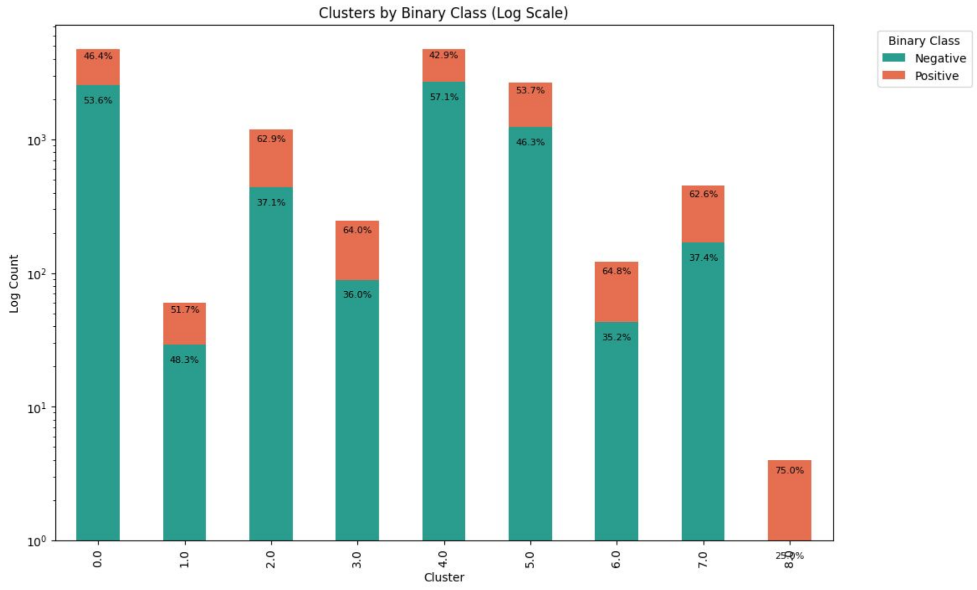
about_description: Short description of the game from the publisher (text embedding)

overall_review: The overall review category classification based on the rating score of the game

- Converted to **binary classification**: split games as positive or negative based on the review score percentage in order to account for class imbalance

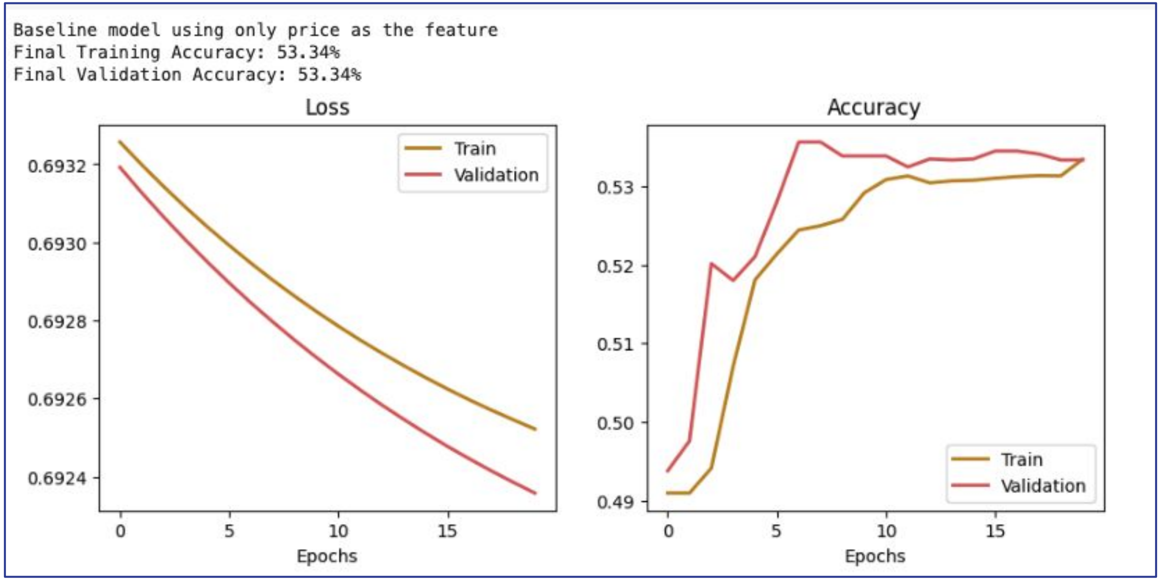
Features and Binary Review Class





Baseline Prediction Model

- Binary Logistic Classification
- Feature: Price, Output: Binary Review Class
- Final Validation Accuracy: 53.3%



Prediction Algorithms

- **Decision Trees**
  Are intuitive and easy to interpret, allowing us to visualize how different features impact the final decision, which facilitates explaining the results to non-technical users.

- **XGBoost**
  An optimized version of decision trees, is highly effective in classification due to its ability to handle large datasets and enhance performance through boosting techniques. We will assign a higher penalty to misclassifications of the minority class using scale weights parameter.

- **Neural networks + hidden layers**
  Powerful for capturing complex non-linear relationships and patterns in the data. Their ability to model intrinsic complexities is essential in a domain as variable as video game preferences, where feature interactions can be highly sophisticated.

Performance Metrics

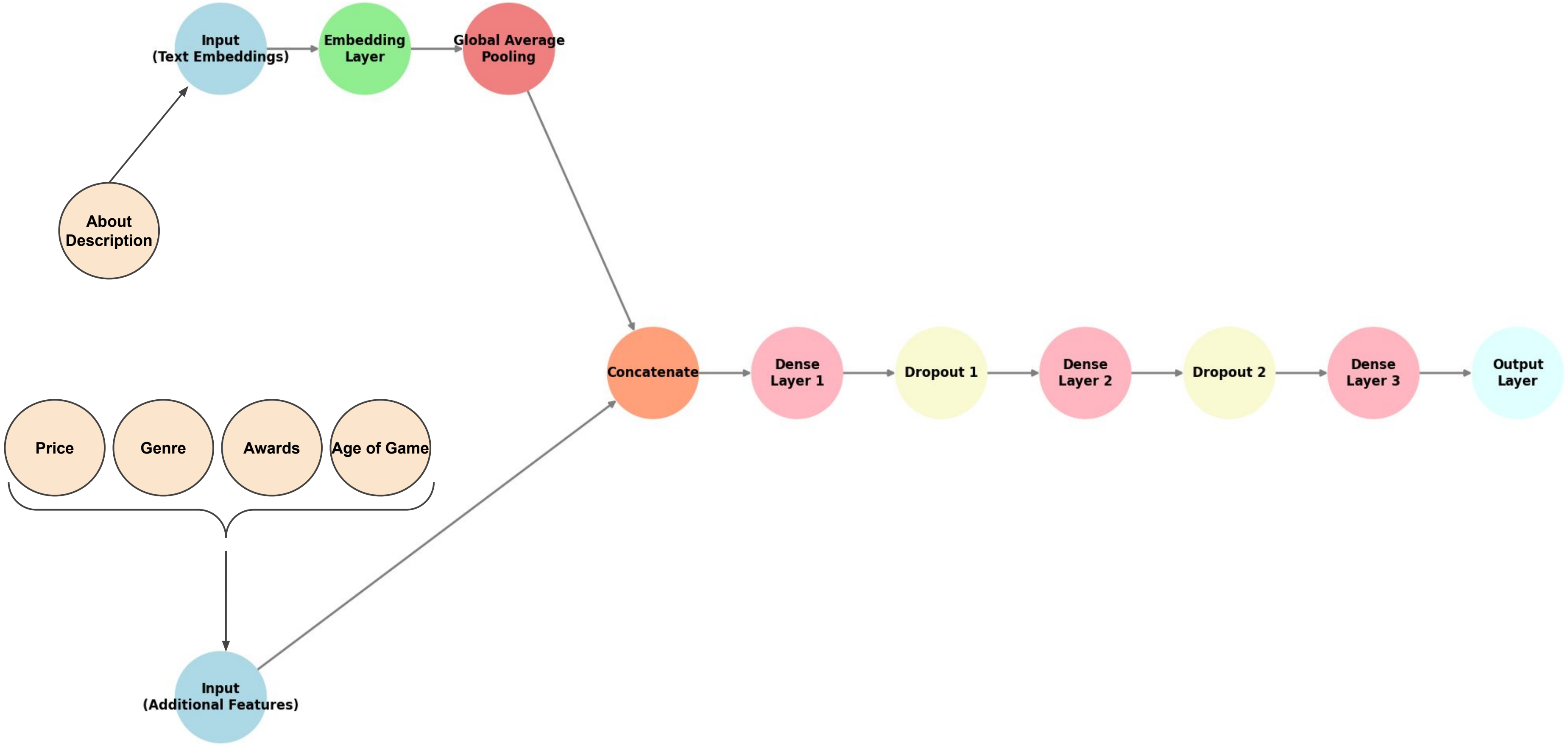
Accuracy:

Measures the ratio of correctly predicted instances to the total instances

All Models & Accuracies

Model	Type	Features	Training Accuracy	Validation Accuracy
Baseline	Binary Logistic	Price	53.5%	53.4%
Model 2	Binary Logistic	Price, Age of Game	57.6%	56.9%
Model 3	Binary Logistic	Price, Genres, Age of game	58.4%	58.0%
Model 4	Binary Logistic	Price, Genres, Age of game, Awards	59.8%	59.4%
Model 5	Binary Logistic	Price, Genres, Age of game, Awards, TF-IDF on Description	60.1%	59.3%
Model 6	Binary Logistic	Price, Genres, Age of game, Awards, Learned Embeddings on Description	66.3%	63.3%
Model 7	Neural Network	Price, Genres, Age of game, Awards, TF-IDF on Description	68.2%	63.2%
Model 8	Neural Network	Price, Genres, Age of game, Awards, Learned Embeddings on Description	64.0%	65.0%
Model 9	XGBoost	Price, Genres, Age of game, Awards, TF-IDF on description	71.0%	63.67%
Model 10	Decision Tree	Price, Genres, Age of game, Awards, TF-IDF on description	67.3%	61.7%

Final Model: Neural Network Architecture



Final Model: Hyperparameter Tuning

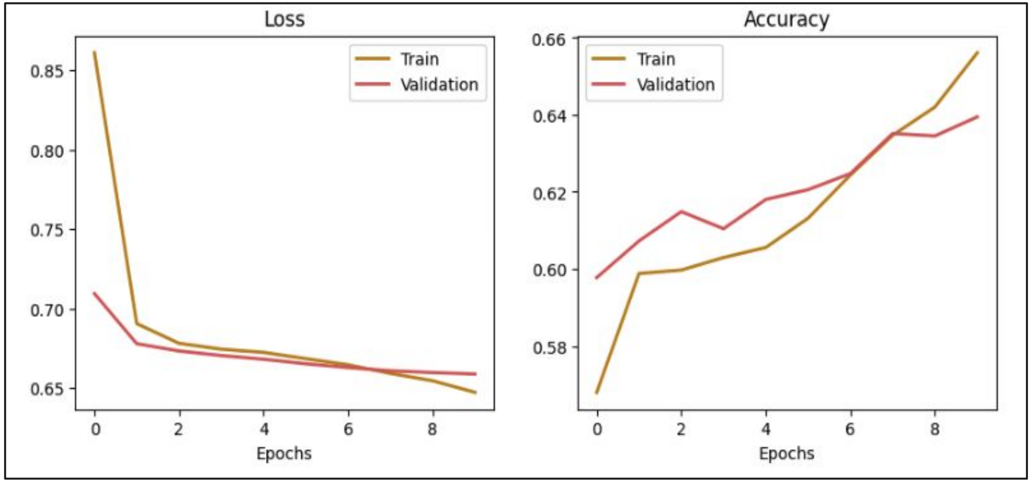
Parameter Grid		
Embedding Dimensions	2	4
Dense Units	32	64
Dropout Rate	0.3	0.4
Learning Rate	0.001	0.005
Batch Size	32	64
Epochs	5	10

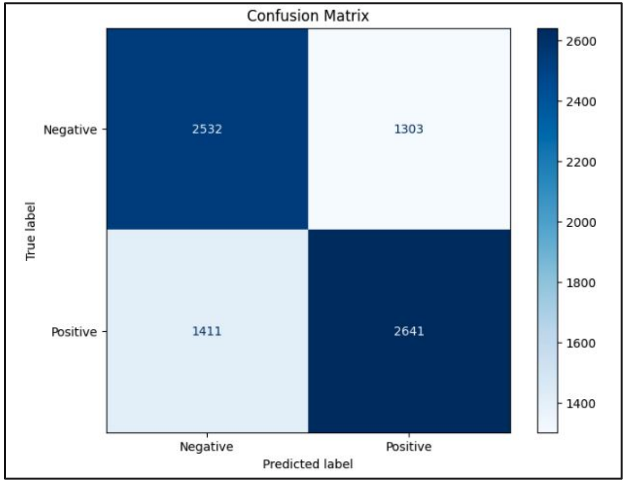
Random Search
cv = 3

Best Performing Model	
Embedding Dimensions	2
Dense Units	32
Dropout Rate	0.4
Learning Rate	0.001
Batch Size	64
Epochs	10

Final Model: Generalization Abilities

- Test Accuracy: 65%
- Improvement over baseline: +12%





Conclusions

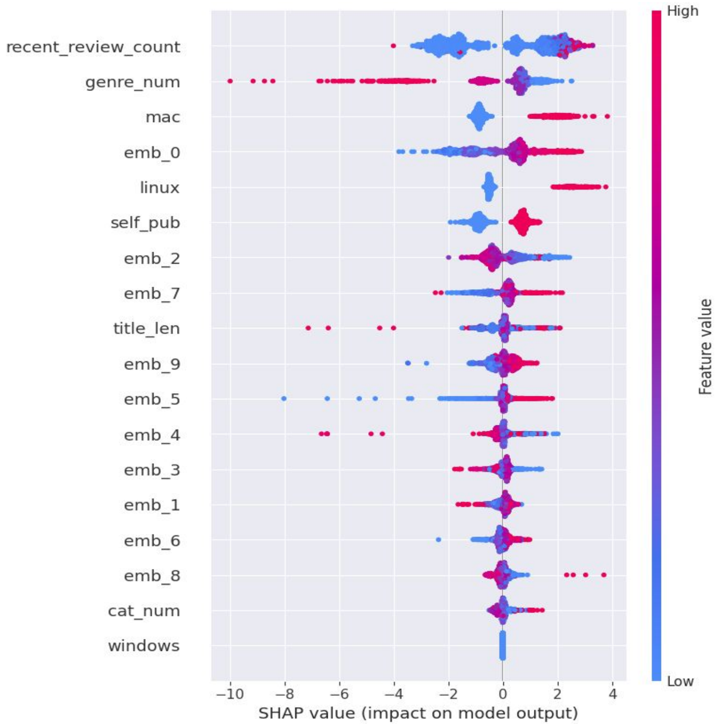
Q: Can we predict the review tier of a game based on its features such as price, genres, publishers, and other relevant features?

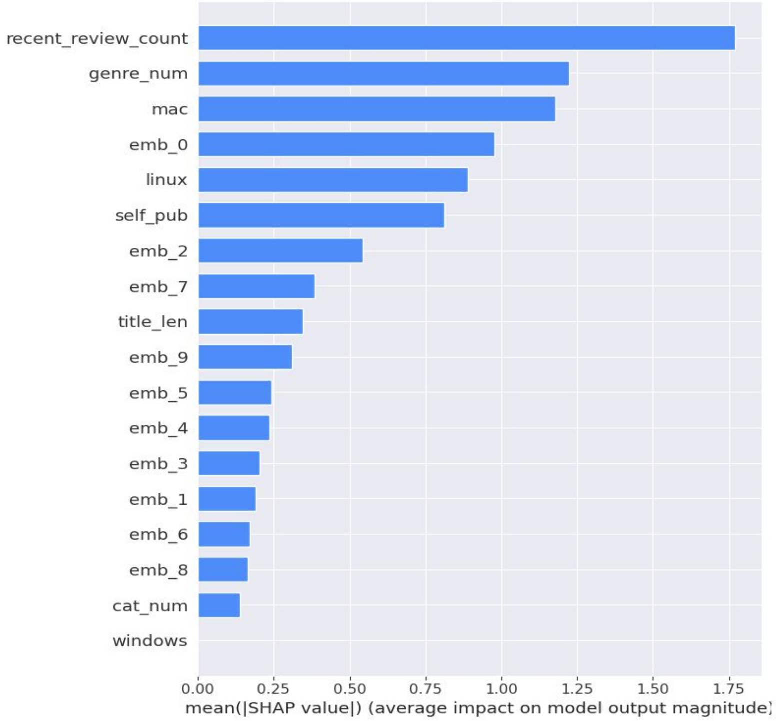
- Binary Logistic Model
    - Best, >10% increase in accuracy from baseline
- More complex models != better performance
- Future work
    - Binary classification: Choosing more robust features
    - Create a multiclass classification model
    - Further investigate clustering
    - Combine neural network with XGBoost

Questions?

Experiments

The combination of both charts provides a clear view of which features are most influential in the model and how these values impact the predictions. High values of **recent_review_count** and **genre_num** are the most determinative for predictions. Other features like **mac**, **linux**, and certain embeddings also play an important role, though to a lesser extent.





Contributions

TODO
- Move to git?

Project Member	Contributions
Ananya	
Elaine	
Mia	
Francisco	

Final Project (I)

- **Final presentation. Your slides should include: (12 minutes + 2 minutes total (no Q&A included). You will be timed!)**
    - Title, Authors
    - (15%) Motivation: Introduce your question and why the question is interesting. Explain what has been done before in this space. Describe your overall plan. Provide a summary of your results.
    - (15%) Data: Describe in detail the data that you are using, including the source(s) of the data and relevant statistics.
    - (15%) Modeling: Describe in detail the models (baseline + improvement over baseline) that you use in your approach.
    - (30%) Experiments: Provide insight into the effect of different hyperperameter choices. Please include tables, figures, graphs to illustrate your experiments.
    - (10%) Conclusions: Summarize the key results, what has been learned, and avenues for future work.
    - (15%) Code submission: Provide link to your GitHub repo. The code should be well commented and organized.
    - Contributions: Specify the contributions of each author (e.g., data processing, algorithm implementation, slides etc.).

Final Project (II)

- **Final Report**
  - You should have a very detailed README file in the repo:
    - The README should cover how would you run or read your repo.
    - Use proper Markdown and links to the codespace
  - Try breaking down your code into different processes instead of a long document
    - If possible, think about how to productionalize the code.
    - Any team that provides a functional pipeline to run the training and testing outside of the notebook will be rewarded with 10% of Extra Credit.
  - Code comments are very valuable - code as a Machine Learning Engineer and less like a Data Scientist
  - Data should be hosted outside of Git
    - So the notebook/code should explicitly mention how to get the data if we want to reproduce the results.

question

Originally multiclass classification -> not enough negatives

Review count as a hive mind thing

Maybe regression to get a percentage score but after that should we then take that score and try to put it into one of the original categories

- Steam categories are based on overall % score + number of reviews so it can be directly mapped to a category

1. Directly do multiclass classification
2. Do classification with custom categories based on the quantiles of the percentages, but bottom quantile is a wide range
3. Do regression with the percentage score and then map to categories (as there are general categories)

Presentation notes

Don’t mention accuracy

Which metric to prioritize/tune (top 1 metric)

Weighted classes have same issues for accuracy, focus on label

Dividing data into 9 buckets but 2 of them don’t have many, **probably just do the quartile thing**

Award, price, sentiment graph hard to read but maybe unsupervised learning/clustering (k-means about numerical features, 9 clusters/however many)

Embeddings in the text/sentiment analysis on about description

Describe less (pandas)