Table of Contents

Introduction

Pricing is a critical driver of success in the video game industry, yet it remains a complex and evolving challenge. The wide variation in game genres, platforms, and player bases means that a one-size-fits-all pricing strategy is rarely effective. While major studios may leverage brand recognition to support high launch prices, indie developers often benefit from more targeted, data-driven approaches.
A major obstacle to optimal pricing is the limitation of historical datasets, which usually record only a single global price per title—restricting the ability to analyze past price-performance relationships. Additionally, many pricing models neglect player sentiment, even though platforms like Metacritic offer extensive user feedback.

To address this gap, this study proposes a cluster-based pricing optimization framework that integrates sentiment-aware Natural Language Processing (NLP) techniques with group-specific deep learning models. Games are clustered based on shared characteristics (e.g., platform, genre, and rating), while sentiment features extracted from user reviews are incorporated to capture player reception. For each cluster, the model predicts global sales across different price points and identifies the optimal price that maximizes expected revenue. This approach enhances pricing recommendations for upcoming or unreleased games by combining historical trends with audience perception.

Problem Identification

Game publishers often struggle to determine the most profitable launch price for new titles due to the lack of granular historical price data and reliable early market signals. Most datasets contain only a single observed global price per game, limiting the ability to explore how different pricing strategies would impact sales performance. Moreover, while factors like sales volume and genre are commonly examined, qualitative signals such as player sentiment—which can influence sales through user ratings and word-of-mouth—are frequently overlooked.

In addition, temporal effects, such as the month of release, may significantly affect player behavior and sales but are often excluded from pricing models. Relying on generalized strategies risks missing important patterns shaped by player reception and seasonality.

This project addresses these limitations through three key innovations:

1. Sentiment-Enhanced Modeling: Leveraging NLP techniques to extract sentiment from user reviews and incorporate it into sales forecasting.
2. Simulation-Based Optimization: Training group-specific MLP models to simulate price–sales relationships and identify revenue-maximizing prices.

By combining data-driven segmentation with qualitative and seasonal insights, this approach offers more tailored and effective pricing recommendations for game publishers.

Data Collection, Organization, and EDA

- **Data Collection**

In this project, the real -world data would be collected from 3 websites, which are "VGChartz", "Metacric.com" and "Dekudeals.com". Sales data would be scrape by VGChartz.com, the websites provide various games and sales amount by global sales, and different region sales. Secondly, we would scrape the players' review data from Metacric.com. In this website, it also classified critic reviews and user reviews. In addition, it would base the score, and review would be asset by different platform such as PC, Xbox, Nintendo Switch… etc. Lastly, since we would only analyze Nintendo Switch games, the price of Nintendo games would be retrieved from Dekudeals.com.
The data would be merged by using the game's name. Since there some typo or punctuation of the names in those different websites, we will use fuzzy string matching based on game title and platform.

- **Data Organization**

To prepare the game dataset for analysis and modeling, a normalization process was applied to handle entries that contained multiple platforms and corresponding metascores within a single record. The following steps were carried out:

1. Load the raw dataset containing game information, where several fields (such as platforms and platform metascores) were stored as list-like strings within individual rows.
2. Implement a custom parsing function that:

- Extracted the list of platforms and the list of metascores.
- Matched each platform to its respective score.

3. Apply the function across the dataset to transform it into a normalized format where each row corresponds to one game on one platform.
4. Filter the normalized dataset to include only a specific platform (e.g., Nintendo Switch). This was done to maintain analytical consistency by removing platform-level variance due to technical specifications or user bases.
5. Perform a fuzzy join with external datasets (e.g., review scores, price data, and sales performance) using cosine similarity to match game titles across tables. Since titles may differ slightly in formatting, a vectorized similarity approach was necessary to align the datasets. Specifically:

- All titles were transformed into vector representations using, TfidfVectorizer, a token-based method.
- Cosine similarity was computed between title pairs across datasets.
- Pairs with a similarity score greater than a defined threshold (similarity_threshold = 0.7) were considered a valid match and joined together.

- This technique enabled accurate alignment of entries like "Mario Kart 8 Deluxe" and "Mario Kart 8 (Switch)", which would otherwise fail an exact string match.

6. Output the cleaned and fully merged dataset, which now consists of single-platform records, enriched with review sentiment, pricing, and sales data—ready for downstream feature engineering and machine learning applications.

This step-wise restructuring ensured that the game data was analytically coherent and aligned with the requirements of structured modeling pipelines.

- **EDA**

Once the initial structural reorganization was completed, further data cleaning and feature engineering were carried out. Before developing the NLP model, several preparatory steps were performed in the EDA notebook to clean, structure, and enrich the dataset for effective natural language analysis. These steps focused on improving data quality and constructing relevant attributes to support both statistical analysis and text modeling. The main procedures included:

1. Removal of duplicate records and null values to ensure data integrity. Duplicate game entries were dropped based on key identifiers, and rows with missing critical information (e.g., title, platform, review content) were filtered out.
2. Data feature extraction

   Feature engineering was performed primarily on temporal attributes. Specifically, the release date was converted into a structured datetime format and decomposed into separate features such as release year, release month, and release quarter. These features were designed to capture potential seasonal or temporal patterns in game distribution and user engagement.

3. Cleaning price-related fields, specifically by removing currency symbols (e.g., $) from the original price column and converting the values into numeric format. This allowed for quantitative comparison and transformation (e.g., discount calculation, log scaling).

These preprocessing steps laid the foundation for implementing NLP models by ensuring that the textual review data was well-structured, cleaned, and semantically relevant for tasks such as sentiment scoring, classification, or feature extraction.

NLP Processing

In this project, Natural Language Processing (NLP) techniques were applied to analyze the game review text found in the quote column. The goal was to extract meaningful features for exploratory analysis and sentiment modeling. The NLP pipeline involved the following key steps:

- Text Preprocessing
  This step included several sub-processes:
  - Tokenization: Splitting text into individual words.
  - Lowercasing: Converting all text to lowercase.
  - Stopword Removal: Removing common words like “the” and “is”.
  - Stemming and Lemmatization: Reducing words to their root or base form.
  - Named Entity Recognition (NER): Identifying names, places, and organizations in text.
- Frequency Analysis & Feature Creation
  The cleaned tokens were analyzed to find the most frequent terms. Text was then transformed into numerical vectors using TF-IDF (Term Frequency-Inverse Document Frequency) for use in machine learning.
- Sentiment Labeling
  Using the VADER sentiment analysis tool, each review was labeled as positive, neutral, or negative, providing supervised labels for training.
- Sentiment Classification Model
  To enhance sentiment classification, a Random Forest classifier was trained using TF-IDF features (X) and sentiment labels generated by the VADER tool (y). Although the labels were not manually annotated, they served as reliable pseudo-labels—machine-generated labels that enabled the supervised training of the classifier. This approach is a form of pseudo-labeling, where an existing model (VADER) is used to assign labels to unlabeled text data.

  The primary benefit of pseudo-labeling is that it allows us to leverage large amounts of unlabeled data without the need for costly manual annotation. By training on pseudo-labeled examples, the classifier can potentially improve performance and generalization beyond what the original labeling model (VADER) provides. In this context, the Random Forest model was designed to replicate or enhance VADER’s sentiment logic, resulting in more flexible and potentially more accurate sentiment predictions.

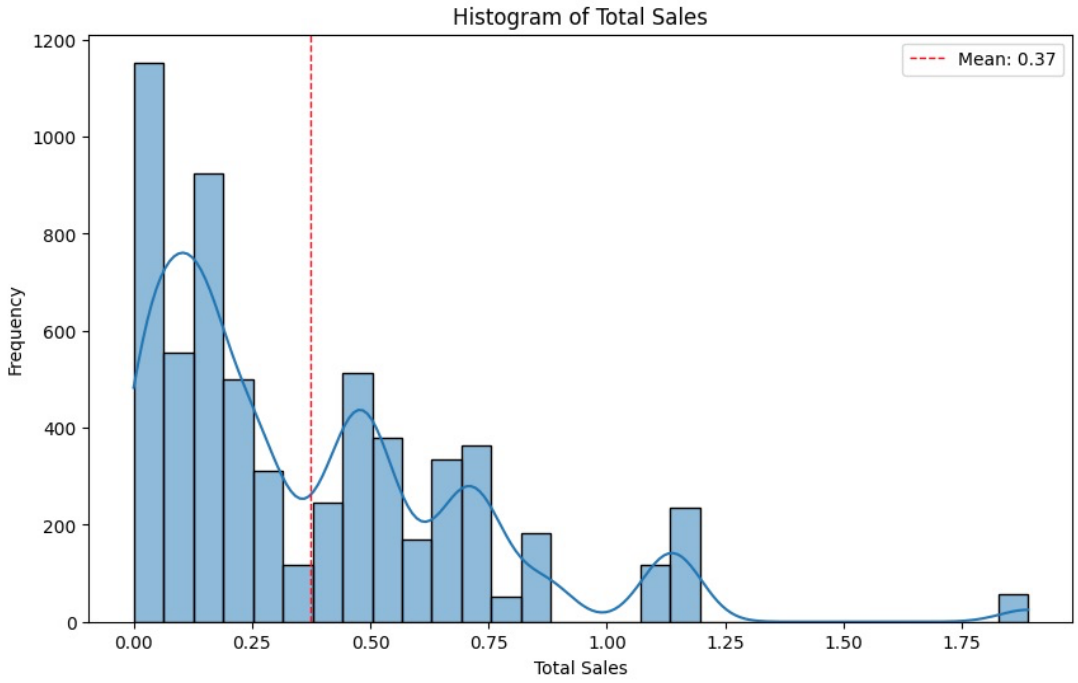
- Integration and Export
  The predicted sentiment labels were added to the original dataset as a new feature (predicted_sentiment) and saved for use in further EDA and modeling tasks.

This streamlined NLP pipeline enabled the conversion of raw text reviews into structured, sentiment-rich features for more effective data exploration and prediction.

Data Visualization

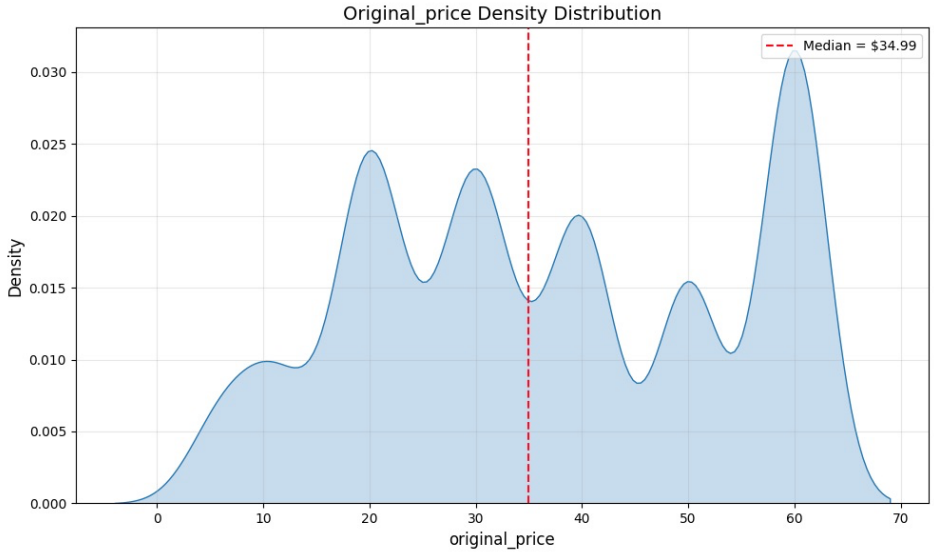
After completing data cleaning and NLP preprocessing, a comprehensive exploratory data analysis (EDA) was conducted to understand the structure, distribution, and key relationships within the dataset. The main findings are summarized below:

- **Sales Distribution**



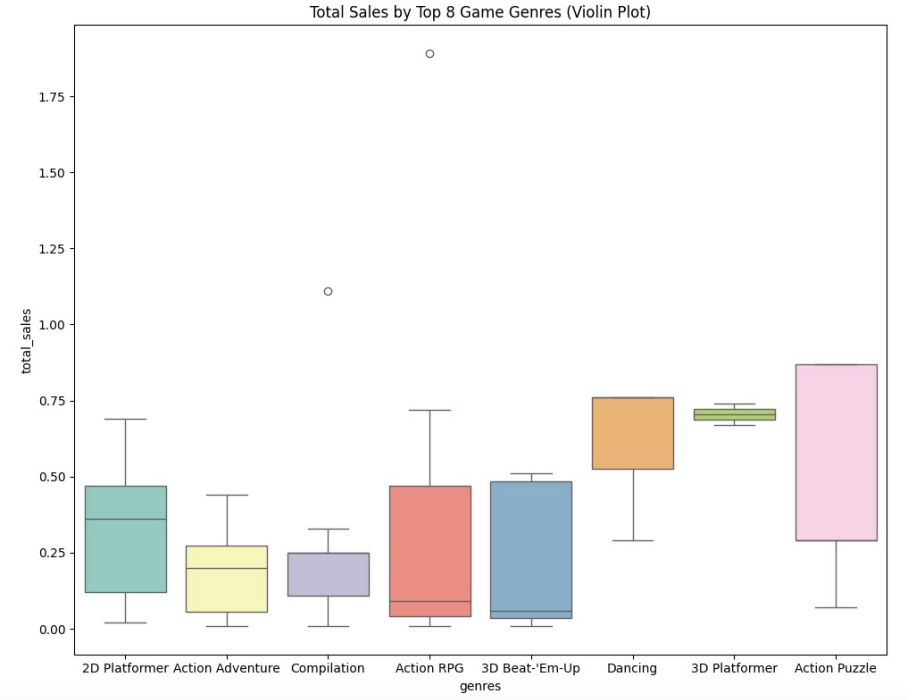
A histogram of total_sales revealed a right-skewed distribution, indicating that while most games have modest sales, a few titles account for very high sales.

- **Price Distribution**



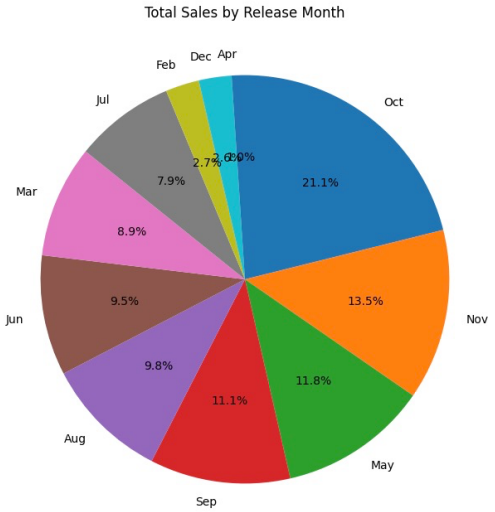
A KDE (Kernel Density Estimate) plot of original_price shows a **multi-modal distribution**, meaning that there are several common price clusters rather than one dominant pricing point. Notably, there are visible peaks around the $20, $30, $40, and $60 ranges, suggesting these are popular pricing tiers among similar games.

- **Genre vs. Sales**



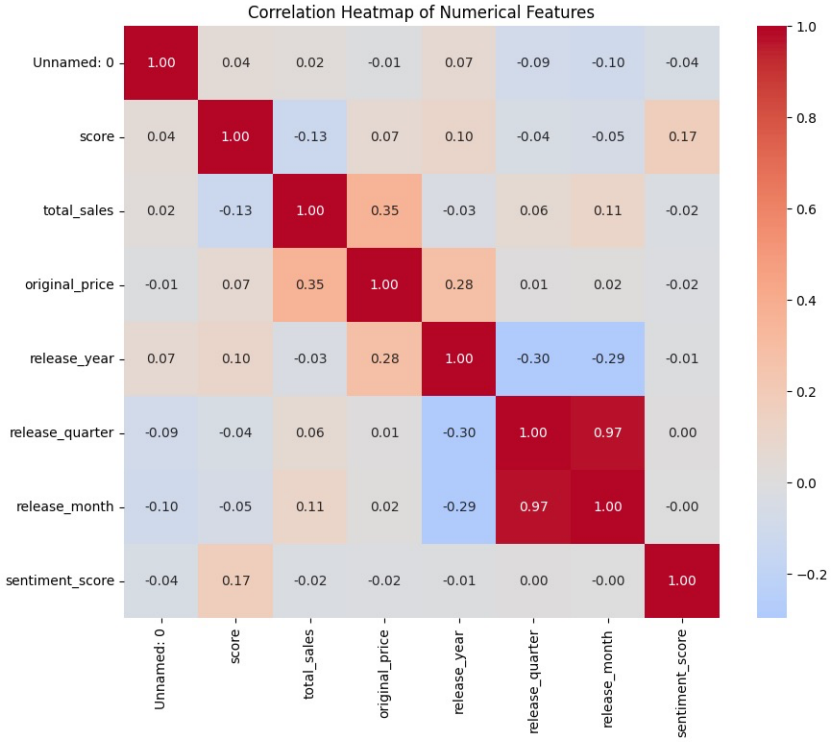
A boxplot of genres vs. total_sales highlighted that certain game genres (e.g., Action, Adventure) tend to perform better in terms of average sales, with considerable variability across titles.

- **Release Month Trend**



A pie chart of total sales by release_month revealed that games released in **October** have the highest overall sales. This may align with seasonal trends and strategic timing before the holiday shopping season.

- **Correlation Heatmap**



A heatmap of numerical features showed significant positive correlations between original price and total sales**,** as well as between release year and total sales**.**

Modeling and Evaluation

In this project, there are two types of models would be used. There are classified and linear regression models.

We used random forest classifier to do sentiment classification. The result showed that overall accuracy of 82%, with strong performance on the positive class (Precision: 0.85, Recall: 0.96, F1-score: 0.90). This is likely influenced by class imbalance, as positive reviews make up the majority of the dataset. The model also performs moderately on the neutral class but shows lower precision and recall.

However, the model performs poorly on negative reviews**,** with a very low recall of 0.12**,** indicating that most negative cases are being missed. To address this, class balancing techniques such as oversampling or weighting should be considered to improve recall and overall fairness across sentiment categories.

	precision	recall	f1-score	support
negative	0.59	0.12	0.20	180
neutral	0.55	0.66	0.60	83
positive	0.85	0.96	0.90	979
accuracy			0.82	1242
macro avg	0.67	0.58	0.57	1242
weighted avg	0.79	0.82	0.78	1242

Secondly, we use regression models to predict sales amount. There are some models were tested to predict the target variable (sales amount), including a simple linear regression, three random forest models (with and without tuning), and several multilayer perceptron (MLP) models with varying architectures.

- **Type 1 Model: Simple Linear Regression**
  Served as a baseline model. It achieved modest performance (**Test R^2 = 0.635**) and had the highest error among all models, indicating limited ability to capture complex patterns.
- **Type 2 Model: Random Forest (Untuned)**
  Showed almost perfect performance (**$R^2 \approx 0.9999$**) on both train and test sets.

- **Type 3 Models: Random Forest (Tuned)**

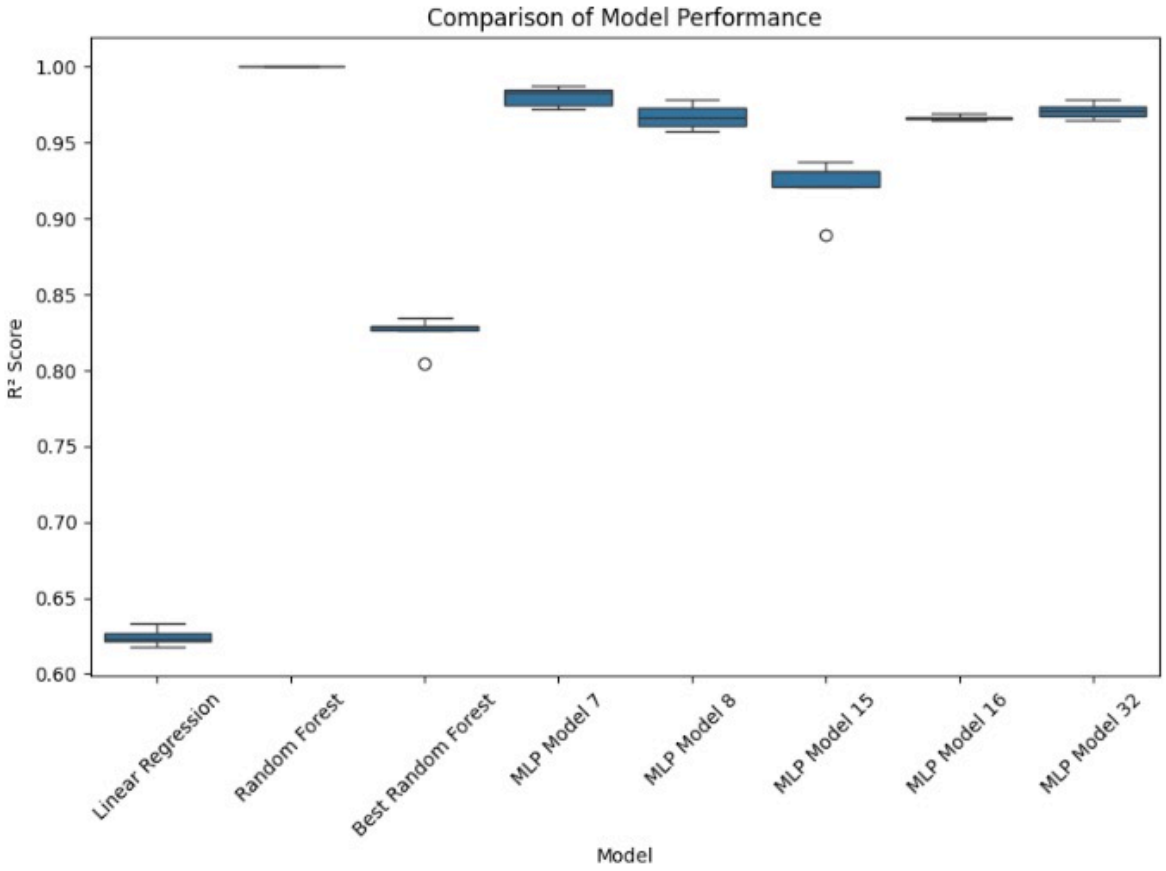
Grid search optimization in the Random Foreast		
Hyper Parameters	Value Test	Best Parameters
n_estimators	[100,200],	100
max_depth	[10,15],	10
min_samples_split	[2,5],	2
min_samples_leaf	[1,5],	5
max_features	['log2']	log2

After grid search optimization, it reached the R-squre of Test dataset **is** `0.849` and `RMSE` is low as 0.134, which significantly improved generalization. This model balances performance and interpretability well.

- **Type4 Models: MLP Models**

	MLP_Model_id	Architecture	Learning Rate	Alpha	Activation	CV RMSE	Test RMSE	Train R^2	Test R^2
14	15	(64,)	0.010	0.1000	relu	0.9216	0.0809	0.9494	0.9455
31	32	(100, 50)	0.010	0.1000	tanh	0.9709	0.0658	0.9646	0.9639
15	16	(64,)	0.010	0.1000	tanh	0.9662	0.0585	0.9717	0.9715
7	8	(64,)	0.001	0.1000	tanh	0.9670	0.0489	0.9818	0.9801
6	7	(64,)	0.001	0.1000	relu	0.9803	0.0488	0.9852	0.9802
0	1	(64,)	0.001	0.0001	relu	0.9670	0.0472	0.9905	0.9814
16	17	(100, 50)	0.001	0.0001	relu	0.9749	0.0463	0.9946	0.9821
20	21	(100, 50)	0.001	0.0100	relu	0.9783	0.0426	0.9949	0.9849
18	19	(100, 50)	0.001	0.0010	relu	0.9758	0.0414	0.9963	0.9857
23	24	(100, 50)	0.001	0.1000	tanh	0.9864	0.0406	0.9877	0.9863
2	3	(64,)	0.001	0.0010	relu	0.9784	0.0401	0.9939	0.9866
12	13	(64,)	0.010	0.0100	relu	0.9374	0.0367	0.9905	0.9888
8	9	(64,)	0.010	0.0001	relu	0.9694	0.0333	0.9932	0.9908
4	5	(64,)	0.001	0.0100	relu	0.9678	0.0327	0.9964	0.9911
30	31	(100, 50)	0.010	0.1000	relu	0.9873	0.0320	0.9913	0.9915
22	23	(100, 50)	0.001	0.1000	relu	0.9943	0.0257	0.9960	0.9945
3	4	(64,)	0.001	0.0010	tanh	0.9906	0.0254	0.9984	0.9946
1	2	(64,)	0.001	0.0001	tanh	0.9918	0.0231	0.9990	0.9955
5	6	(64,)	0.001	0.0100	tanh	0.9878	0.0230	0.9981	0.9956
29	30	(100, 50)	0.010	0.0100	tanh	0.9945	0.0220	0.9968	0.9960
10	11	(64,)	0.010	0.0010	relu	0.9718	0.0192	0.9977	0.9969
21	22	(100, 50)	0.001	0.0100	tanh	0.9961	0.0192	0.9982	0.9969
13	14	(64,)	0.010	0.0100	tanh	0.9872	0.0177	0.9975	0.9974
26	27	(100, 50)	0.010	0.0010	relu	0.9969	0.0173	0.9987	0.9975
24	25	(100, 50)	0.010	0.0001	relu	0.9965	0.0162	0.9984	0.9978
19	20	(100, 50)	0.001	0.0010	tanh	0.9956	0.0159	0.9992	0.9979
17	18	(100, 50)	0.001	0.0001	tanh	0.9954	0.0155	0.9992	0.9980
11	12	(64,)	0.010	0.0010	tanh	0.9893	0.0146	0.9988	0.9982
28	29	(100, 50)	0.010	0.0100	relu	0.9973	0.0143	0.9987	0.9983
9	10	(64,)	0.010	0.0001	tanh	0.9897	0.0135	0.9989	0.9985
27	28	(100, 50)	0.010	0.0010	tanh	0.9986	0.0130	0.9993	0.9986
25	26	(100, 50)	0.010	0.0001	tanh	0.9988	0.0125	0.9994	0.9987

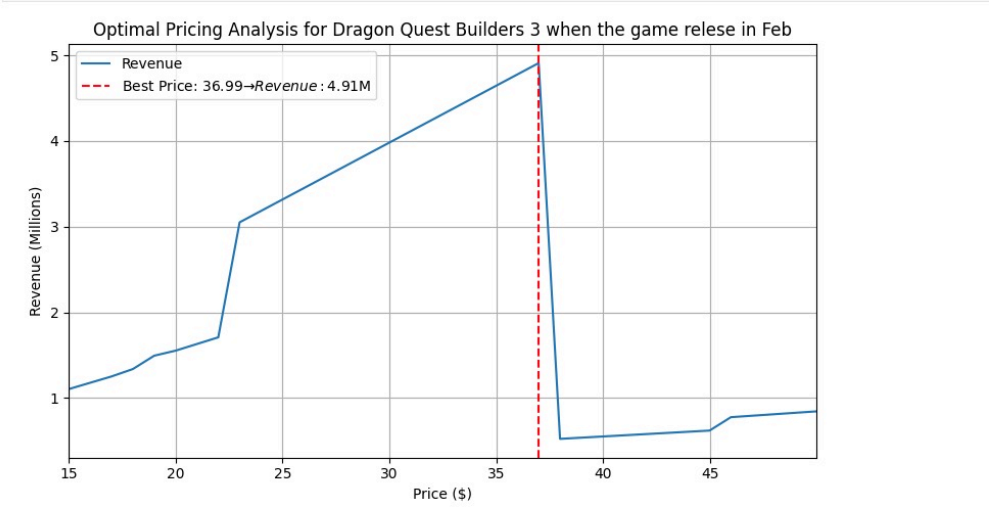
All three MLP models performed well, especially the third one (Architecture: 100, 50), which achieved the **lowest test RMSE (0.0557)** and the **highest Test R^2 (0.9739)**. This demonstrates the neural network's strong capacity to model nonlinear relationships.



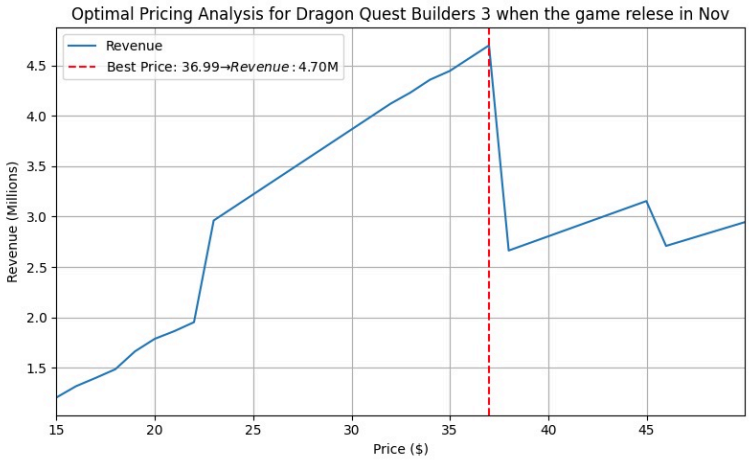
Based on the evaluation results, the best overall model was the third MLP (architecture: 100, 50), which achieved the highest test R^2 of 0.9739 and the lowest RMSE of 0.0557, indicating strong predictive performance and the ability to capture complex nonlinear patterns. Among traditional models, the tuned Random Forest (Best Random Forest) provided a good balance between accuracy and generalization, with a test R^2 of 0.8523 after hyperparameter optimization. In contrast, the simple linear regression model served as a baseline and showed limited effectiveness, highlighting its inability to handle the nonlinear structure of the data.

Conclusion and Case Study

This study employed a Random Forest regression model to simulate and optimize the pricing strategy using *Dragon Quest Builders 3* as the test case. The simulation was based on historical data of sandbox games genre and released on the Nintendo Switch platform. The model incorporated features such as review sentiment, quote length, pricing history, and game metadata to estimate future sales under various pricing scenarios.




```
Best price: $36.99, Predicted sales: 0.13270 million units. Maximum projected revenue: $4.91 million USD.
```




```
Best price: $36.99, Predicted sales: 0.12700 million units. Maximum projected revenue: $4.70 million USD.
```

This study demonstrates the practical application of data-driven simulation for game pricing optimization using *Dragon Quest Builders 3* as a case study. By leveraging a Random Forest regression model trained on historical data from similar sandbox titles released on the Nintendo

Switch platform, we were able to estimate optimal pricing strategies under different hypothetical release months.

Release Time	Price	Amount	Revenue
Feb-25	49.99(Same as the previous)	0.0169	0.84 M
Feb-25	36.99(Best price among same-genre)	0.1327	4.9M
Nov-25	36.99(Best price among same-genre)	0.127	4.7M

The model consistently identified $36.99 as the optimal price for both February and November release scenarios. In the February simulation, this price point yielded the highest projected revenue of $4.9 million from an estimated 0.1327 million units sold—significantly outperforming the historical price of $49.99, which resulted in only 0.84 million in revenue. When applied to a November release, the same optimal price generated slightly lower sales at 0.127 million units, leading to a projected revenue of $4.7 million. These results highlight not only the impact of pricing strategy, but also the sensitivity of revenue to seasonal release timing and prediction precision.

These findings emphasize several key insights:

1. Sentiment-Enhanced Forecasting: By incorporating NLP-based sentiment features, the model captured not only structural metadata (price, platform, genre) but also consumer reception, improving the realism of sales predictions.
2. Release Timing Matters: Although the optimal price remained consistent, subtle variations in projected revenue reflect the influence of seasonality and launch context—factors that game publishers must consider in tandem with pricing decisions.