# Research on the Prediction on the Sales of Electronic Games

Weiqi Huang

Computer Sciences, Wuhan University, Wuhan, 430072, China

* Corresponding Author Email: 2019302110409@whu.edu.cn

**Abstract.** This essay's primary goal is to predict and analyze the sales characteristics of electronic games based on different features, including but not limited to release time, ranking, publisher, and game genre. The methods employed in this paper include a neural network architecture where the activation function is ReLU , XGBoost model, and LightGBM model. Through experiments, this paper effectively constructs machine learning models using these three methods to predict video game sales. Furthermore, the performance of these three methods is compared and analyzed using three evaluation metrics: root mean squared error (RMSE), mean absolute error (MAE), and mean squared error (MSE). And the prediction error of each model is plotted as a related line chart to visually show the details of the three methods in analyzing this problem. Finally, by comparing the experimental results of these three methods in the field of electronic game sales prediction, this article analyzes the advantages and disadvantages of neural networks, XGBoost, and LightGBM in related problem analysis, providing readers with reference and reference.

**Keywords:** sales prediction, neural network architecture, XGBoost model, LightGBM model.

## 1. Introduction

The research field mentioned in this paper has always been a hot topic because it involves significant potential economic benefits. It is evident that the sales of many popular games reach hundreds of thousands or even millions of copies, resulting in substantial revenues ranging from tens of thousands to millions of dollars. However, there are also numerous games that underperform, with actual sales falling far below market expectations. The gaming industry can be regarded as a high-reward, high-risk industry. Therefore, conducting predictive analysis on game sales, which is an economic question, holds tremendous economic value in terms of potential returns.

The analysis of game sales in the gaming industry during the previous ten years is the primary subject of this article. By examining and processing various features such as sales, rankings, publishing platforms, genres, publishers, and regional sales proportions relative to total revenues, a model is constructed to achieve the prediction of game sales. Specifically, neural networks, XGBoost, and LightGBM methods are employed for model construction. Furthermore, a comparative analysis is conducted to assess the performance and strengths and weaknesses of these three methods.

There have been numerous research achievements in this field. Many previous studies have focused less on directly predicting game market sales, instead focusing on a specific feature of the game to analyze its impact on game sales and have not been able to combine a large number of multiple analyses and predictions of game sales as a whole in electronic games [1,2].

Or perhaps many previous studies only used traditional neural networks and regression analysis to predict the sales of electronic games. Although XGboost and LightGBM gradient boosting tree algorithms have been widely used in predicting many economic problems such as stock prices and housing prices, few people have previously used these two methods to analyze and predict sales in the gaming market [3,4].

Moreover, with the passage of time, the gaming market has witnessed the emergence of new game companies and genres. Many previous findings, due to the long-time interval, are no longer directly applicable to the analysis of sales in today's gaming market. As a result, it's important to update the pertinent data aspects and suggest new forecasting techniques that may be used with the market's current data.

Studying this issue can help investors understand market trends. By researching game sales in the market, one can gain insights into market trends and changes. These trends and changes may include

popular game genres, platforms, geographic locations, and target audiences. This information can assist game developers and publishers in better understanding market demand and formulating more targeted strategies.

Predicting sales performance: Analyzing historical sales data enables the prediction of future game sales performance. These forecasts can aid developers and publishers in planning development and marketing budgets, as well as optimizing game features and functionalities.

Guiding investment decisions: The sales situation in the game market is an important reference for investment decisions. Investors can assess the potential value and risks of game companies and projects through research on market trends and sales data.

Enhancing competitiveness: Understanding game market sales can help game developers and publishers gain better insights into competitors and formulate more effective strategies to improve competitiveness.

In summary, studying and predicting game market sales is crucial for various stakeholders in the gaming industry. It can help them make wiser decisions, improve performance, and increase profits.

## 2. Data and method

### 2.1. Data

#### 2.1.1 Introduction to data sources and datasets

The data used in this study is sourced from: https://gregorut.kaggle.com/videogamesales A list of video games with sales of more than 100,000 copies is included in the dataset. It was created via web scraping data from vgchartz.com (the data scraping script, which makes use of Python's BeautifulSoup, may be obtained at https://github.com/GregorUT/vgchartzScrape).

In the initial dataset, each game is treated as a separate data sample with different features. These features include the game's ranking, name, platform of release (e.g., PC, PS4), release year, genre (e.g., shooter, fighting), publisher (e.g., Sony, Nintendo), sales in North America, Europe, Japan, other regional sales, and worldwide sales. Among these features, the global sales of the games are the target variable that aim to predict, while the other features will be preprocessed and used as relevant predictors in the analysis.

#### 2.1.2 Data preprocessing

Handling Missing Values: The median value from the dataset is used to fill in the missing values in the "Year" column of the various data samples. The value "Unknown" is used to fill in the blanks in the "Publisher" column. This approach aims to minimize the issue of data loss due to missing values and ensure an adequate size of the training set for model training.

Encoding Categorical Features: The string-type categorical features such as "Platform", "Genre", and "Publisher" are converted into numerical representations. Each unique value of the relevant features is assigned a numerical value. This translation turns textual input into numerical form that machine learning algorithms can understand, allowing the computer to recognize and discriminate between various characteristics and facilitating model training.

Standardizing Numeric Features: The numeric feature "Year" is standardized, which scales the data in the "Year" column to have a standard deviation of 1 and a mean of 0. This approach guarantees that scales and distributions of certain attributes are comparable, allowing them to have comparable influence in the model. It helps improve the performance and accuracy of the model.

80% of the original dataset is used as training data, while 20% is used as test data. The random number generator's seed is controlled by parameters to ensure consistent results across script runs.

Overall, the purpose of preprocessing is to clean and prepare the data for model training and evaluation. Through these steps, the raw data is transformed into a format suitable for neural networks, enhancing the performance and accuracy of the model.

## 2.2. Method

### 2.2.1 Neuaral network

The five fully connected layers of the neural network model have output nodes that are 128, 64, 32, 16, and 1 correspondingly. Between each pair of completely linked layers, a ReLU activation function is performed. and a dropout regularization layer is added after the first four layers. In the forward propagation method, the input tensor x is first computed through the top layer with all connections and the ReLU activation function, then passed through the first dropout regularization layer. Subsequently, it goes through the remaining fully connected layers, activation functions, and dropout regularization layers. Finally, a scalar value is outputted as the prediction result of the model.

This neural network model uses methods like dropout regularization and ReLU activation function to improve the model's performance and generalizability. It also demonstrates a certain amount of complexity.

This neural network model demonstrates a certain level of complexity and utilizes methods like dropout regularization and ReLU activation function to improve the model's performance and generalizability. ReLU is a nonlinear function commonly used in neural networks, and it has the following characteristics [5].

Fast convergence: Compared to other traditional activation functions, ReLU allows for faster convergence during neural network training.

Fast computation: ReLU has low computational complexity, which speeds up the forward and backward propagation processes of the neural network.

Avoidance of gradient vanishing problem: ReLU prevents the issue of gradient vanishing, enabling better training of deep neural networks [6].

Enhancement of sparsity: ReLU outputs 0 for negative inputs, enhancing sparsity of the input values, reducing inter-neuron correlations, and improving the model's generalization ability.

In summary, in neural networks, the ReLU activation function is essential for expediting the training process., improving model performance and generalization ability, and avoiding issues like gradient vanishing. Due to these advantages, ReLU has become popular as a activation functions.

The neural network model ultimately outputs a scalar value as the prediction result of the model. This result is a float value representing the predicted global sales. Specifically, when a specific game data sample is inputted into the neural network model, after performing forward propagation calculations, the output layer will produce a float value that represents the predicted global sales for that game. It can be considered as the prediction result in a regression problem.

### 2.2.2 XGBoost

The key idea behind XGBoost is to sequentially add weak learners, typically decision trees, to form a strong predictive model. Each subsequent tree is constructed to correct the mistakes made by previously added trees. This iterative process allows XGBoost to continuously improve its predictions by minimizing a specific loss function [7].

Key features and advantages of XGBoost include:

Techniques for regularization: XGBoost offers a number of regularization methods, such as L1 and L2 regularization, to assist reduce overfitting and improve generalization performance.

Gradient-based optimization: XGBoost uses gradient descent optimization methods to efficiently find the best model parameters. The learning process is guided by the gradients of the loss function in relation to the model's predictions.

Tree pruning: XGBoost includes an innovative technique called tree pruning, which removes unnecessary branches from individual decision trees to reduce complexity and improve computational efficiency.

Handling missing values: XGBoost can efficiently use data even when some features have missing entries since it has built-in ability to manage missing values in the dataset.

Parallel processing and scalability: XGBoost supports parallel processing, enabling faster training on multi-core CPUs. Additionally, it can handle large-scale datasets efficiently and is optimized for distributed computing frameworks like Apache Spark [8].

Flexibility and extensibility: Users may fine-tune the model using a variety of XGBoost's programmable parameters to meet their own needs. It also supports custom loss functions to address unique problem domains.

A dictionary containing the model parameters is defined, and the xgb.train() function is used to train the XGBoost regression model.

First, the numpy arrays, pandas DataFrames, or other data types are transformed into the data format required by XGBoost. Then, the training set's feature matrix and label vector are passed as parameters to the model.

The maximum depth of the trees is defined as 6, the learning rate (the amount of weight shrinkage in each iteration) is set to 0.01, and the subsample ratio (the proportion of samples used in each boosting round) is 0.7 for each training iteration.

#### 2.2.3 LIghtGBM

LightGBM is designed with the following key features:

Histogram-based decision tree algorithm: LightGBM uses a technique called histogram to discretize continuous features into discrete bins and performs splits based on histograms. This approach speeds up the training process and improves memory usage efficiency.

Leaf-wise growth strategy: LightGBM uses a leaf-wise approach to growth, which focuses on growing leaves that contribute to a larger gradient. This strategy leads to faster convergence compared to the traditional level-wise growth strategy [9].

Optimized for classification tasks: LightGBM incorporates specialized optimization algorithms for classification problems, resulting in improved performance for handling classification tasks [10].

Handling large-scale datasets: LightGBM has a small memory footprint and offers fast training speed, making it effective for processing large-scale datasets. It also supports parallel training, allowing for faster training on multi-core CPUs or distributed environments.

High accuracy and generalization ability: LightGBM optimizes the objective function and combines feature discretization and histogram techniques to achieve high model accuracy and good generalization ability.

The model first defines the dataset format for the training set and test set. Then, it defines the model parameters, including the objective function as regression, the evaluation metric as root mean squared error (RMSE), the maximum number of leaves in each tree as 31, and the learning rate as 0.05.

In summary, the defined model is a regression model using the LightGBM algorithm. It is trained with specific parameters to minimize the RMSE on the training set and learn how to map features to target values.

## 3. Results

Three error visualization plots were made based on these three models, where each integer on the x-axis represents the arrangement of samples in the test set and the y-axis represents the difference (in millions) between the predicted and actual values. This line plot depicts the v and offers insights into the mistakes' patterns.

### 3.1. Neural network

#### 3.1.1 Training process

A training loop is used during the 200 epochs of the training procedure. The training and testing losses are printed and kept track of in an array for each epoch. After the training is completed, the

model is evaluated on the test set to obtain its predictions for root mean squared error (RMSE), mean absolute error (MAE), and mean squared error (MSE).

### 3.1.2 Test results

It can be observed that the performance of the conventional neural network is not satisfactory (Figure 1), with significant variations in predictions for many samples in the test set. Finally, the predicted values and true values are saved to a text file for further analysis.

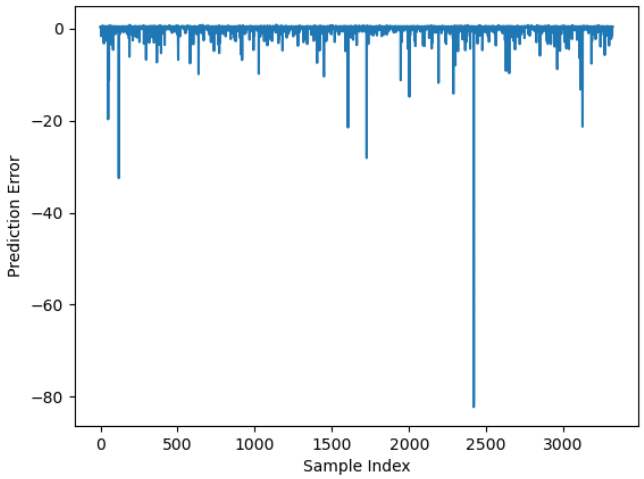


**Fig. 1.** The discrepancy between the neural network's real values and anticipated values, expressed in millions.

## 3.2. XGBoost

### 3.2.1 Training process

Variables are set to specify the number of iterations for training, and then the XGBoost regression model training begins. The feature matrix is passed as a parameter to the function, while specifying the number of iterations as 1000 rounds. During the training process, the model automatically optimizes the parameters to minimize the objective function (i.e., squared error) and evaluates the model's performance by calculating RMSE, MAE, and MSE.

### 3.2.2 Test results

The discrepancy between XGboost's real values and anticipated values is shown in Figure 2.

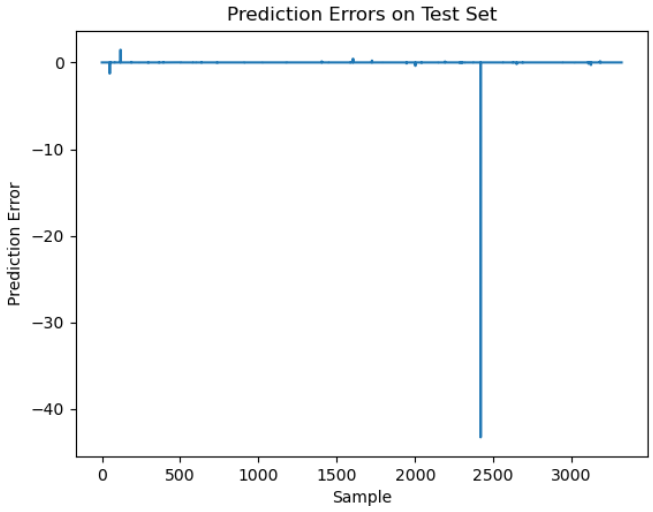


**Fig. 2.** The discrepancy between XGboost's real values and anticipated values, expressed in millions.

### 3.3. LightGBM

#### 3.3.1 Training process

The number of iterations is defined as 1000, indicating that the model iterates 1000 times during the training process. The model is trained by passing parameters, including model parameters, training set data, and the number of iterations. During the training process, the model iteratively optimizes based on the objective function and evaluation metric to minimize the root mean squared error on the training set and learn how to map features to target values.

#### 3.3.2 Test results

The discrepancy between LightGBM 's real values and anticipated values is shown in Figure 3.

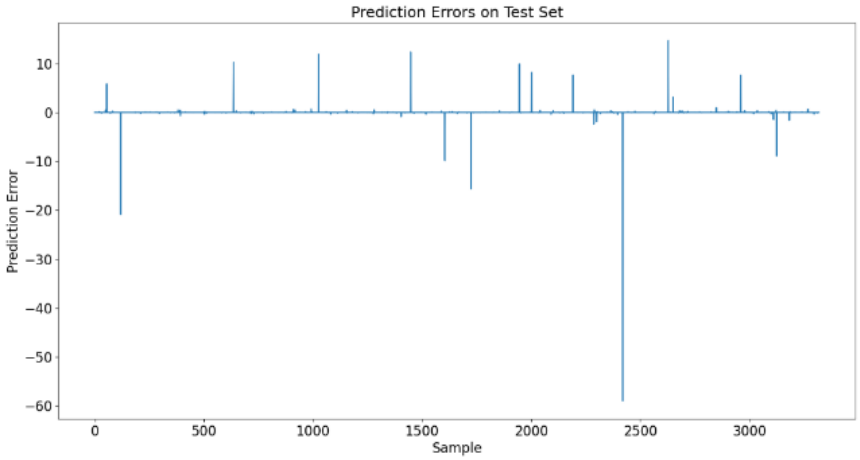


**Fig. 3.** The discrepancy between LightGBM 's real values and anticipated values, expressed in millions.

### 3.4. Prediction performance comparsion

#### 3.4.1 Evaluation criteria

This study uses RMSE, MSE, and MAE as performance metrics to evaluate different models for the research question at hand.

The average square root of the discrepancies between genuine values and anticipated values is represented by the term "RMSE" (Root Mean Squared Error). A lower RMSE suggests improved model performance and lower prediction errors.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{n}(y_i - predy_i)^2}{n}} \tag{1}$$

Here, n denotes the number of samples, $y_i$ represents the true values, and $predy_i$ represents the model's predicted values.

MSE (Mean Squared Error) represents the average squared differences between predicted values and true values. A smaller MSE indicates smaller prediction errors and better model performance.

$$MSE = \frac{\sum_{i=1}^{n}(y_i - predy_i)^2}{n} \tag{2}$$

Here, n denotes the number of samples, $y_i$ represents the true values, and $predy_i$ represents the model's predicted values.

The average absolute variances between anticipated values and actual values are represented by the term MAE (Mean Absolute Error). A lower MAE suggests improved model performance and lower prediction errors.

$$MAE = \frac{\sum_{i=1}^{n}|y_i - predy_i|}{n} \tag{3}$$

Here, n denotes the number of samples, $y_i$ represents the true values, and $predy_i$ represents the model's predicted values. Overall, RMSE and MSE are more sensitive to larger errors between predicted values and true values, while MAE gives equal weight to all errors.

### 3.4.2 Final comparsion

During the training and testing process, these three evaluation criteria are calculated and presented in Table 1.

**Table 1.** Overall Results comparison Among Different Methods Based on Three Criterias

| | Neural network | XGboost | LightGBM |
|---|---|---|---|
| RMSE | 2.046379 | 0.7515 | 1.2640 |
| MAE | 0-621843 | 0.8195 | 0.8256 |
| MSE | 4.187667 | 0.5647 | 1.5978 |

Based on the three-evaluation metrics, we can see that overall, the XGBoost model performs the best, while the conventional neural network shows poor performance for this research question.

### 3.4.3 Insights from the comarsion

For conventional neural networks with ReLU activation functions, their performance tends to be poor when analyzing predictive economic problems:

Sampling Issue: Market data is usually high-dimensional, nonlinear, and noisy. Traditional neural network models require more and larger training samples to handle such data. However, market data is often limited and incomplete, making it difficult to obtain sufficient training samples.

ReLU Activation Function Issue: Conventional neural network models typically use ReLU as the activation function, which has advantages such as simplicity, nonlinearity, and efficiency. However, ReLU may suffer from the problem of gradient vanishing when dealing with extreme points, causing the network to fail to learn the correct features.

Interpretability Issue: Conventional neural network models are often considered black box models, making it challenging to explain their prediction results. In fields like market forecasting, it is important to interpret and analyze the prediction results, thus requiring interpretable models.

XGBoost and LightGBM are popular gradient boosting decision tree (GBDT) algorithms that can achieve high performance in various datasets and application scenarios. However, which algorithm has a higher prediction accuracy depends on the specific problem and dataset.

In general, XGBoost and LightGBM can achieve similar performance in most cases. However, they each have their own advantages and disadvantages, so it is necessary to choose based on the actual situation.

The advantages of XGBoost include:

It can effectively avoid overfitting through regularization techniques.

It supports distributed computing and can handle large-scale datasets.

It can use various loss functions, including regression, classification, ranking, etc.

The advantages of LightGBM include:

It uses a histogram-based decision tree algorithm, which can quickly handle high-dimensional sparse data.

It uses a leaf-wise growth approach, which converges more quickly.

It can achieve higher performance when dealing with smaller datasets.

Therefore, if you are dealing with large-scale datasets and need to use multiple loss functions, XGBoost is more suitable. If you are dealing with high-dimensional sparse data or need fast training and prediction, LightGBM is more suitable.

Considering the current research on game sales prediction, since a list of video games with sales of more than 100,000 copies is included in the dataset, the dataset is large, and multiple loss functions including RMSE, MSE, and MAE are used to evaluate the performance of different models in this research problem, it is expected that the XGBoost model would perform better. On the other hand,

LightGBM sacrifices some prediction accuracy in order to achieve better performance and faster convergence, resulting in slightly inferior results.

## 4. Conclusion

In this study, three machine learning methods, neural networks, XGBoost, and LightGBM, were employed to analyze and predict the sales volume characteristics of different game samples using features such as game release date, ranking, publishing company, and genre. The outcomes demonstrate that this application problem is successfully solved by the XGBoost model, followed by the LightGBM model, while the conventional neural network performs poorly. The obtained research results also demonstrate that considering multiple characteristics and attributes of games can effectively help predict their sales volume, thereby assisting investors in making relevant market decisions.

However, there are limitations to the availability of datasets in the relevant field in the current market. The rapid development of the gaming industry has occurred only in recent decades, and the data in the gaming market varies widely. Some games have high sales while others have low sales, resulting in less representativeness and regularity in the samples compared to other economic research problems.

## References

[1] C E. Near, J. Sex roles, **68**, (2013)

[2] F. Adigüzel, J. İşletme Araştırmaları Dergisi, **13**, 3(2021)

[3] J. Li, Y. Zheng, H. Hu, et al. *Predicting Video Game Sales Based on Machine Learning and Hybrid Feature Selection Method*, 16th International Conference on Intelligent Systems and Knowledge Engineering (ISKE). IEEE, (2021)

[4] K. Saraswathi, N. T. Renukadevi, S, Nandhinidevi, et al. *Sales prediction using machine learning approaches*, AIP Conference Proceedings, AIP Publishing LLC, (2021)

[5] A. Novikov, D. Podoprikhin, A. Osokin, et al. J. Adv neural info proc syst, **28**, (2015)

[6] A F. Agarap, arXiv preprint, (2018)

[7] Chen T, Guestrin C. *Xgboost: A scalable tree boosting system*, in proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, (2016)

[8] X. Shi, Y. D. Wong, M. Li, et al. J. Acc Anal & Prev, **129**, (2019)

[9] G. Ke, Q. Meng, T. Finley, et al. J. Adv neural info proc syst, **30**, (2017)

[10] D. Wang, L. Li, D. Zhao. J. Info sci, **202**, (2022)