What Video Game Genre will be popular next?

By Max Kogan

Github link: https://github.com/MaxKoganUNCC/MachineLearningFinal

Introduction

My goal for this project is to enable two machine learning algorithms to predict what genre of video games may become popular, and to see if machine learning is viable in this scenario. The landscape of the video game industry has become increasingly competitive over the years. The technology and knowledge required to produce a video game have also become more accessible, leading to new competition. To overcome these challenges, a useful avenue would be to incorporate machine learning to create a predictive model based on historical data.

The motivations behind my decision are both from a personal and industry perspective. I've grown up with video games my whole life, and I plan to partake in developing video games as well. Beyond this, the industry has become extremely competitive, as I touched on earlier. Almost anyone can produce a video game given the time and knowledge, but only a few become viral or well-known. By exploring these trends, this data can help developers pick what genres may be best suited for success. There are several challenges present in this endeavor, such as the fact that most video games fall under a multitude of genres, along with factoring the aspect of time into the data and results, as trends are known to change with time.

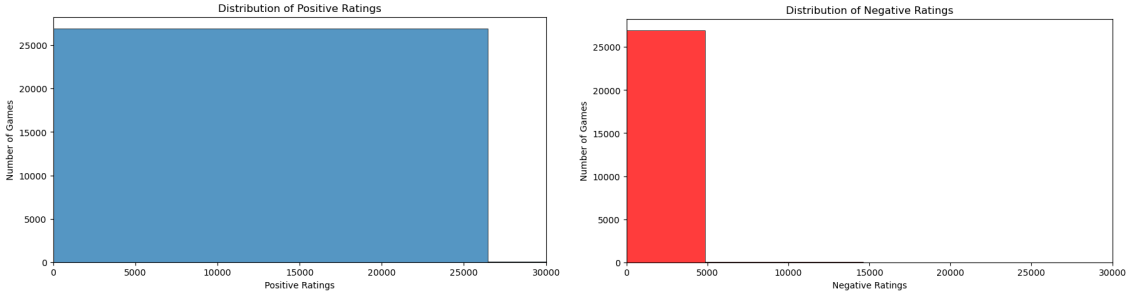
To meet my goals, several steps must be taken. First, it is necessary to collect and preprocess data, in which I have chosen a dataset from Kaggle called "Steam Store Games (Clean dataset)"(Davis). For clarity, Steam is an online game storefront for PC, being one of the most popular and widely used ones. After performing an analysis of the data, I will conclude which features are the most helpful for this cause and preprocess such data so that it is ready for the two machine learning algorithms I have chosen, which will be K-Nearest Neighbors and Linear Regression. After using the two algorithms, a conclusion will be made on the results.

Taking these steps is necessary to gain an accurate dataset so that our algorithms can properly analyze and test their predictive capabilities.

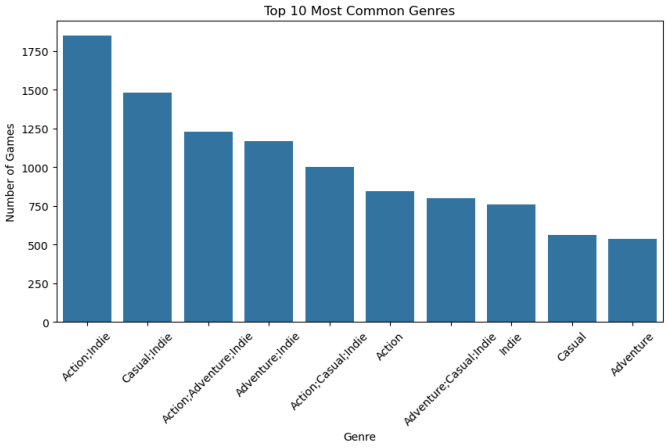
Data/Environment

As mentioned before, the data I have chosen to use is called “Steam Store Games (Clean dataset)”[1]. The dataset includes 27,075 different titles (samples) across 18 features, which include attributes such as genre, user score, release date, and more. There are both qualitative and quantitative features present in the dataset, which enable better insights into what may or may not contribute to a video game’s popularity. Furthermore, something to note is that Steam uses a Positive/Negative rating system rather than something like an average user score. Due to this, I must create a ratio or percentage of positive and negative scores for a game to determine its rating.

The large amount of data samples in the dataset also provides a better representation of the possible trends in the relationship between genres and game success. A large dataset also reveals large amounts of diversity, allowing more subtle patterns to be explored. For example, there may be a niche combination of game genres that does surprisingly well in terms of the ratio of its score.



In the above figure, we have the distribution of positive and negative ratings across all games. While positive ratings far outnumber negative ratings, it's important to consider that negative ratings will still have an impact on a game's performance. I will be combining these two attributes to create a "score" value that can be used for the machine learning algorithms later.



The above figure displays the top 10 most common game genres on Steam. A notable fact about the data is the combination of multiple genres, which, in the context of video games, is not only accurate but also beneficial to how one should consider a game's appeal. Many games incorporate several genres into their gameplay, creating a distinct blend of entertainment. I also anticipate that just because a game genre is common, it doesn't inherently mean that it will be a guaranteed success.

My method for preprocessing the data not only considers the two machine learning algorithms I have chosen, but also the nature of the data at hand. As mentioned before, I will create a new score attribute by combining the data between the positive and negative ratings, allowing us to properly score a game out of 100%. Preprocessing also involved splitting the genres each game had into a form that would be friendly for the machine learning algorithms to use, alongside normalizing the numeric rating columns.

Method

As previously mentioned, the two machine learning algorithms I intend to use are K-Nearest Neighbors (KNN) and Logistic Regression. Firstly, KNN is a rather simple but powerful algorithm used for classification and regression tasks, which works well for my problem. In KNN, data points are classified based on their nearest neighbors in the feature space, going off the general logic that similar points in data sets tend to have a lot in common.

To classify a new data point, the algorithm determines the distance between that new data point and all other data points. In the context of this problem, a data point that is near other data points labeled as “positive” will be predicted to be positive. The number of data points selected to be compared to the new data point is known as the “k” value.

Adjusting the k value is the most important part of KNN, as smaller values make it more sensitive to noise while larger values may reduce the accuracy overall, if too big. However, KNN has incredible strengths due to its simplicity, and it should help identify trends in a game genre’s performance.

The other method, Logistic Regression, has greater emphasis on predicting the probability that a data point will fall into a certain class. Logistic regression uses the softmax function to calculate that probability and is generally more efficient than KNN. This does come with the issue of Logistic regression typically underperforming when put up against more complicated data.

Fortunately, the data we’re looking to predict is how positive a genre is, which goes hand-in-hand with the classification nature of Logistic regression. Logistic regression is trained by using maximum likelihood estimation, which is done by implementing gradient descent to

minimize any cross-entropy loss. Thanks to these factors, Logistic regression is another powerful tool to help solve the problem at hand.

Between Logistic regression and KNN, the two will provide strong methods for predicting a genre's popularity. Linear regression also has the advantage of being highly interpretable, giving us a probability for the predictions. I can combine the results of the two algorithms to help determine the most accurate predictions for the genres I'm testing.

Results

To set up my machine learning algorithms, I had to first split the data into training and testing groups. This is done to make sure that the model has exposure to new, unseen data and also prevents overfitting. Afterwards, I was able to train and test the two algorithms. This was accomplished by using various methods from the Python scikit-learn library("Scikit-Learn: Machine Learning in Python — Scikit-Learn 1.1.3 Documentation") that enable the use of the two algorithms I decided to use, along with several metrics to analyze the results.

KNN Classification Report:

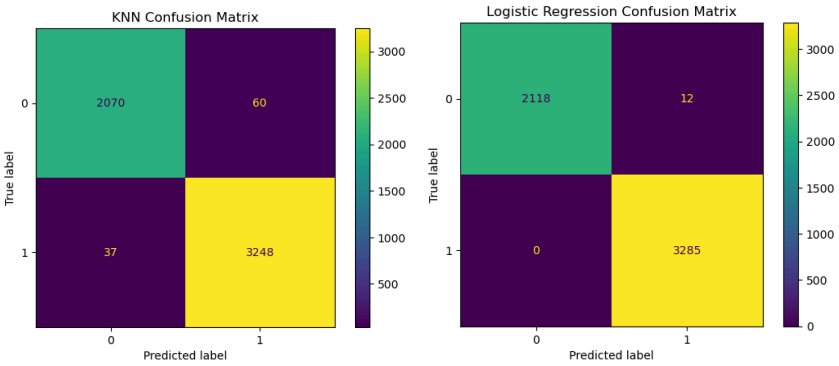
	precision	recall	f1-score	support
0	0.98	0.97	0.98	2130
1	0.98	0.99	0.99	3285
accuracy			0.98	5415
macro avg	0.98	0.98	0.98	5415
weighted avg	0.98	0.98	0.98	5415

KNN Accuracy: 0.9820867959372115

Logistic Regression Classification Report:

	precision	recall	f1-score	support
0	1.00	0.99	1.00	2130
1	1.00	1.00	1.00	3285
accuracy			1.00	5415
macro avg	1.00	1.00	1.00	5415
weighted avg	1.00	1.00	1.00	5415

Logistic Regression Accuracy: 0.9977839335180055



As pictured above, the results of the two algorithms are very pleasing. I both did a standard results metric and a confusion matrix to determine the success of my results for each algorithm, with high accuracy in both! The confusion matrices also show a low amount of misidentified positives and negatives, which supports the high accuracy.

Despite the very high accuracy results, I was still able to experiment with the KNN results I was able to bump the accuracy from about 97% to almost 99% simply by playing around with the K value hyperparameter. The results for Logistic Regression were nearly perfect, however, so I did not feel the need to adjust anything there.

Conclusion

Overall, I was intimidated by this project at first, but in the end it was quite enjoyable. I am more than happy to see the results of my machine learning models perform well, and it was fun applying the concepts learned into something that interests me, which in this case was video games. When it comes to the bigger picture, machine learning can certainly help predict successful genres in video games, in which I was expecting the algorithms to struggle with some of the more nuanced predictions. I'm glad that tools like these exist to help people both create new things and make better decisions.

Acknowledgements

All resources used for this project are listed below in the References & Citations, including the Scikit-Learn documentation and the dataset I used.

References & Citations

Davis, Nik. “Steam Store Games (Clean Dataset).” *Www.kaggle.com*, 2019, www.kaggle.com/datasets/nikdavis/steam-store-games.

“Scikit-Learn: Machine Learning in Python — Scikit-Learn 1.1.3 Documentation.” *Scikit-Learn.org*, scikit-learn.org/stable/#.