Looking for your next obsession? Recommending *Steam* Games using Machine Learning Algorithms

Luciano Elish
SUNY New Paltz
CPS493-04
Professor Sreya Banerjee
elishl1@newpaltz.edu

Abstract

Steam is a popular online platform and storefront developed by Valve for PC video games. It allows users to purchase, download, and play games right from the platform, and includes social features like friends lists, chat, and community forums on game pages. For people who love video games and computer scientists (usually goes hand and hand) alike, this binary classification task is to predict whether a game is recommended or not. This recommendation algorithm is implemented by logistic regression, Gaussian Naïve bayes, and [third algorithm yet to be used, probably SVM]. The data set used includes over 85,000 games listed in the Steam store as of March 2025, and many features to go along with it. Through the previously stated algorithms, I was able to get a baseline accuracy of about 61-66% on the testing set, which essentially means it was 61-66% accurate in predicting whether a game should be recommended or not.

I. INTRODUCTION

Steam, being one of the most widely used platforms for online PC gaming, is something that I use on a regular basis to play video games. One of the main things that gets me to buy and play new games are the reviews of said game, which got me thinking once I came across this huge data set of Steam games on Kaggle. I thought that I could make my own recommendation algorithm, and with help from Professor Banerjee, figured a good measure for whether the game should be recommended or not could be based on positive reviews of that game. This recommendation would be the target output for the algorithm to predict, based on several features in the data set, *not* including anything to do with reviews as the algorithm would just see positive reviews being high means that the game is recommended.

This is a binary classification task, and common machine learning algorithms such as Naïve Bayes, logistic regression and [SVM?] were used in this work. The data set used is tabulated as a csv file, and only 14000 entries were used after cleaning the dataset. Since there were so many entries that didn't need to be worked with, I chose the number 14000 based on how many available entries met the conditions on the recommendation column and number of reviews a game has in each row.

For the task, I divided the newly cleaned and shortened data set into training and testing components, an 80%-20% split respectively. After applying the models to the data set, the baseline results came in of ~61% and ~66% accuracy for Gaussian NB and logistic regression respectively.

II. RELATED WORKS

The idea of using machine learning for recommendation algorithms is not a new one, I found when looking for related works for this project, a very similar recommendation algorithm right on Kaggle where I downloaded the data set from. The code used an

SVM algorithm and created a recommendation column based on total reviews and percentage of positive reviews for it to predict. https://www.kaggle.com/code/bernardolmedeiros/recommendation-2025

III. METHODOLOGY

3.1 Methods and Results

I took the final, cleaned and randomized data set and created a model from it, with logistic regression and Naïve Bayes having been successfully implemented with some preliminary results.

3.1.1 Gaussian Naïve Bayes

My data contains numeric and categorical data, but as a classification task, I saw Naïve Bayes as a good fit, only after some tinkering and encoding is done to the dataset to make it trainable for the algorithm. Naive Bayes is a very simple probabilistic classification algorithm that makes strong assumptions about the independence of each attribute of data and is based on the classical Bayes theorem. Nevertheless, it has been shown to be effective in many problems.

To make Naïve Bayes a viable algorithm for my data set, I had to encode and transform several of the columns being used from categorical to numerical. Now, some of these columns, namely "genres", "categories", and "tags", have many values attached to each individual, making the transform task quite a gargantuan one. I ended up with over 34000 columns, each a column for a specific tag/ genre/ category/ publisher etc., all individual and turned numerical using OneHotEncoder import from scikitlearn. After running the score, the accuracy was ~61%.

3.1.2 Logistic Regression

Much like with Naïve Bayes, logistic regression is a great algorithm to use when dealing with binary classification and predicting an output. It assigns weights to each input attribute and outputs a value between 0 and 1. This output can be seen as a probability of success relative to the target variable (in this case, recommending a game), with any probability above a certain threshold (over 80% positive reviews that meets the criteria of at least 100 total reviews) is considered a success. This is how I formatted the classification problem. After running the score, the accuracy was ~66%. This is just the preliminary finding and does not yet fully entail the specifics of each algorithm and results.