ELEKTROTEHNIŠKI VESTNIK 88(4): 151–162, 2021
ORIGINAL SCIENTIFIC PAPER

Predicting the Popularity of Games on Steam

Andraž De Luisa[1], Jan Hartman[1], David Nabergoj[1], Samo Pahor[1], Marko Rus[1], Bozhidar Stevanoski[1], Jure Demšar[1], Erik Štrumbelj[1]

[1]*Faculty of Computer and Information Science, University of Ljubljana, Slovenia*

Abstract. The video-game industry has seen a rapid growth over the last decade. Thousands of video-games are released and played by millions of people every year, creating a large community of players. Steam is a leading gaming platform and social networking site allowing its users to purchase and store games. A by-product of Steam is a large database of information about games, players, and gaming behavior. In the paper, we investigate the relation between the game popularity and features that can be acquired through Steam. We predict the popularity of Steam games in the early stages after their release and we use a Bayesian approach to determine the impact of a game price, size, supported languages, release date, and genres on its player count. We implement several models and show that a genre-based hierarchical approach achieves the best performance. We further analyze the model and interpret its coefficients which indicate that games released at the beginning of the month and games of certain genres correlate with the game popularity.

Keywords: video-games, Bayesian inference, hierarchical modeling, Stan

Napovedovanje popularnosti videoiger na platformi Steam

Industrija video iger je v zadnjem desetletju doživela izredno hiter razvoj. Vsako leto je izdanih na tisoče video iger, ki jih igrajo milijoni igralcev. Steam je vodilna igralna platforma in socialno omrežje, ki uporabnikom omogoča nakup in shranjevanje video iger. Podatki platforme Steam nam omogočajo vpogled v igralne navade njenih uporabnikov in popularnosti iger na platformi. V prispevku raziščemo razmerje med različnimi lastnostmi iger na platformi Steam in njihovo popularnostjo. Naloge se lotimo preko implementacije različnih Bayesovskih napovednih modelov, s katerimi skušamo razumeti kako pri dani igri njena cena, velikost, število jezikov, žanr in druge lastnosti vplivajo na končno število igralcev. Najbolj uspešne napovedi dosežemo z žanrskim hierarhičnim modelom.

1 INTRODUCTION

Steam is a video-game digital distribution service owned by the Valve Corporation. It is currently the most widely used video-game platform on personal computers, having published more than 8000 games in 2019 alone.[*] Some of these games instantly reached a large community of players, while many others remained unpopular. It is difficult to fully understand how a game's popularity changes after its release, as this is affected by many factors which are often difficult to measure or quantify. For example, marketing and budget information may be of a great value for such a model but are generally difficult to obtain. Furthermore, popularity heavily relies on a great player experience, which is affected by many factors, such as the complexity of the game story, graphics, player interaction, etc. These attributes can only be measured through some kind of operationalization, for instance by analyzing reviews or observing the number of Google searches for a game.

However, the basic properties of a game may also affect its popularity. In the study, we focus on discovering how such properties may improve chances for a successful release of a game. For instance, we observe the game price and reason whether it is more sensible to release a game for free and reach many players, or release it at a higher cost, which may indicate that the game is of a higher quality. We also address questions like how popularity differs among different genres and of what genre a game should be to have chances to succeed. Any kind of such information can help video-game developers make design decisions about their games. We approach this problem from a Bayesian perspective, as it allows to compute robust uncertainty estimates that may be highly relevant for making potential business decisions. For example, if a model predicts the player count with a high certainty, the developer may confidently incorporate changes to the game development process. However, if a model uncertainty is high, then it may be better to further analyze the state of the development.

Below we briefly review some of the work analyzing the success of the Steam games. In Section 2, we describe the procedure of obtaining the Steam game data and the preprocessing steps taken to improve its

Received 13 Maj 2021
Accepted 14 July 2021

[*]Source: steamspy.com, accessed May 1, 2021.

quality. In Section 3, we describe how the data was transformed to be used in our models and describe the models and the motivation behind them. In Section 4 we list the quantitative results and provide visualizations of the model performance and discuss effects of different features on generated predictions. In Section 5 we review the key points of the paper and give some directions for our future work.

1.1 Related Work

The game popularity can be defined and measured in numerous ways. The simplest and most intuitive operationalization of the game popularity is the number of players in a period of time since a game is played by more players if they know about it. Budiarto et al. [1] use four other metrics and combine them to calculate the game popularity: user count, unique page views, average time on the page, difference of the unique page views, and average time on the page from the day before. They gather this information using data from the Google Analytics platform, which is updated daily. They also emphasize that the player count is the primary explanatory variate for understanding the game popularity, which is why we use it as the only target variable.

Ahn et al. [2] characterize popular and unpopular games by observing game reviews obtained from Steam. They identify different types of reviews that are associated with the popularity of a game. This reveals what users like and what makes them dislike a game. If certain reviews associated with a low popularity appear on Steam, one can then take an action to improve the game. Lin et al. [3] close the gap between the domain of game reviews and the domain of app reviews and predict when negative reviews are most likely to be posted. This helps understand the player satisfaction with the game and explain the popularity of the game. Since not many reviews are posted for unpopular games and because of possible bias or misleading reviews, we do not use them in our analysis.

Sometimes we are only concerned about a specific property of a video-game and want to explore how it affects its popularity. Lin et al. [4] analyze the advantages of *early-access* games, where players can purchase and play a game before its official release. In an early-access stage, the average rating of the reviews is much higher than later, suggesting that players are more tolerant of imperfections in early-access games. Therefore, it is reasonable to make the game accessible early, as positive reviews may attract more players later. We follow this fact in our analysis and generalize it to consider not only the *early-access* tag, but also other game tags (mainly genres).

2 Dataset

We use several data sources to collect diverse information about the Steam games. To capture the most recent state of the video-gaming market, we focus only on the games released after 2015.

In the next sections, we list the sources of our collected data and provide a short account on how the data is collected and processed.

2.1 Data Collection

The data is collected from several sources: Steam, SteamSpy [5], and SteamDB [6].

SteamSpy and SteamDB are independent sources unaffiliated with Steam itself, but offer data that cannot be acquired through Steam, e.g. SteamDB collects and stores the past numbers of concurrent players, while Steam only offers the current number of concurrent players. We scrape the data over the course of one week in mid December 2020 using the relevant APIs for each data source. To make sure that our scraper functions correctly, we manually verify the scraped information on a small random sample by checking if what we retrieve matches the data displayed on websites.

Steam official API allows us to access the list of all items available in the Steam store. However, in addition to games, the list also includes various other items, like additional downloadable content or even non-game software. API does not provide an easy way to filter such items. To address the issue, we use a SteamSpy listing of standalone games. After obtaining the list of standalone games, we use Steam API to get all the relevant information for each individual game. The data are obtained for each game including a list of genres, developers and publishers, game price, release date, number of supported languages, and description of the system requirements. The final step in our data collection is providing the history of concurrent player counts from SteamDB. Here, we provide the daily player count history for the games released after 2015.

2.2 Data Preprocessing

To make the collected data usable for our analysis and modeling, the data are preprocessed and cleaned. The first modifications are related to the price and system requirements. Steam provides the game prices in different currencies. We convert the prices to Euros and apply the static conversion rates: 1 USD = 0.82 EUR and 1 GBP = 1.09 EUR (rates of November 2020). The system requirements are retrieved as unstructured text data. We extract the storage requirements from the text data and convert them to MB. We perform a manual inspection and remove the game selection with unreasonably high storage requirements (e.g. *HuH?: and the Adventures of something* which requires 9000 GB of storage). Our experiments are performed on a dataset of 8000 games.

3 Methodology

In this section, we present our feature engineering process, the collected raw features and provide an additional insight and statistical information about both the raw and engineered features. Afterwards, we present our predictive models and explain our choice of priors.

3.1 Feature Engineering

We operationalize the response variate as a median player count in the second month after a game release. This is regarded as the game future player count. We consider the game median player count in the first month after its release as the main explanatory variate. Analogously this is regarded as the game past player count. We also construct features by considering specific information about the game.

Contrary to the past player count, which is a reflection of the game performance after it has already been released, some information is known at the moment of the release and plays its role when a player decides to buy a game. For instance, a player might opt not to buy a certain game due to its price. The game properties we use as features are the game price, number of supported languages, and game storage requirements.

There are different motivations for considering the game price as an explanatory variate. The simplest one is that cheaper games are generally more popular since more people can afford them. This is already suggested in [7] related to the price dynamics of video-game consoles. Policy simulations show that Nintendo could have won the commercial sales competition with Sony by reducing the cost of its products, thus reaching a larger user base. The reason for our including the number of supported languages is similar. We expect that games with a larger number of supported languages reach a larger player base. We show the distribution of the top supported languages in Figure 1.

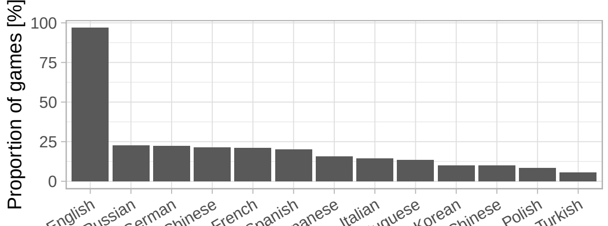


Figure 1.: Support of the most popular languages. As seen, English is by far the most supported language. Note that one game can support more languages.

The game storage requirement is another analyzed feature. In [8], Limelight Networks reinforces the need of including this feature. The data based on worldwide survey results reports that 87 percent of players find the process of downloading games frustrating. Larger games may therefore suffer in popularity due to their longer download times as well as the longer download times of their potential updates. Oppositely, a larger storage size may indicate that a game has more playable content, resulting in a longer player engagement. The storage size can also indicate the general scale and budget of the game, since larger games tend to come from more prestigious studios.

Lastly, we construct different temporal features from the release date timestamp. For each game, we extract the *release year* value and information about the actual release day, both on a yearly as well as monthly basis. We denote these features as a *release day* and *monthly release day*, respectively. To illustrate, that a game is released on March 2, 2020, it would have the release year value of 2020, the release day value of 62, since it is released on the sixty-second day of the year, and a monthly release day value of 2, since it is released on the second day of the month. The motivation behind these features is to capture different seasonalities present in the player behavior or particular to the Steam platform or to capture the effect of different commercial years on the game popularity. Specifically, the release day (an integer between 1 and 366) is included to capture the yearly seasonality related to possible annual online sales, holiday purchases, increase in gaming in a particular season, etc. Past studies observe a statistically significant impact of seasonality on the monthly playtime [9]. The monthly release day (an integer between 1 and 31) is included to capture the monthly seasonality, such as the inclination to purchase or play games after receiving a monthly paycheck.

3.2 Data Insights

In this section, we present a few insights into the data which are important for explaining our modeling. We cover the variates present in the data, show some of their properties and determine correlations between them and the response variate.

3.2.1 Basic Numeric Variates: The three basic numeric variates are the number of the supported languages, storage requirements and price. We show a few statistics in Table 1. All of them are heavily skewed towards 0 as most games will not have support for many languages, are not large, not particularly expensive or are actually attained for free. We can also observe outliers present in all variates by noting their high standard deviations. This is particularly notable in the case of storage requirements.

3.2.2 Genres, Developers, and Publishers: Besides the numeric variates, we also deal with the following categorical variates: game genres,[*] developers, and pub-

[*]We follow the Steam broader definition of the word genre which includes the information such as early-access.

Table 1.: Numeric variates statistics.

Name	mean	median	stdev	min	max
Number of languages	4.800	2.000	5.500	1.000	29.000
Storage requirements (GB)	4.600	1.000	10.100	0.001	256.000
Price (EUR)	9.920	6.750	12.390	0.000	325.910

lishers. These variables have a common property. A game can have more than one of them and vice versa (i.e. a many-to-many relationship). Due to this, they require transformations to be usable in our models.

In our dataset, there are 33 genres, over 19000 developers, and over 23000 publishers. A large number of developers and publishers relative to the number of games is not a good sign for using them as features in our model. This means that most of them have probably not made many games and thus it is very hard to learn anything about them. In fact, over 15000 developers and 18000 publishers have made only one game. Over 18000 developers and 22000 publishers have made less than five games, leaving us with very few of those who have made more games and will bring a benefit if they are added into our models. Another possibility is to group small developers and publishers, but this would bring a significant confusion in the model, and as the cardinality would still remain fairly high, we decide not to use them.

Oppositely, the genres are more useful. The distribution of the number of games belonging to genres is still skewed since there are many genres with a small number of games, but not as extremely as in the cases of developers and publishers. There are also only 33 genres, so cardinality is not a problem and utilizing genres in our models is thus much easier. To see how the different genres connect, we visualize the counts of games and the connections between genres in Figure 2. We observe that indie, action, casual, and adventure games are the most common. Interestingly, two clusters emerge, i.e. the actual games and game-related utilities. They have only one common genre, i.e. an early-access. From the opacity of the edges (i.e. the proportion of games the genres share relative to the total number of games that belong to those genres), we can see which genres have more in common.

3.2.3 Player Counts: We are interested in the distribution of our target variable, the median count of players in the second month after a game release. It relates heavily to our main predictor, the median player count of the first month. When visualized, they both appear to be similar to the power law distribution, which is often the case with the popularity of the online media [10]. We simplify the visualization by transforming the relevant data into the difference between the player counts of the first and second months of the game – more specifically, the difference between their medians. We show a histogram of the differences between the medians in Figure 3. We can observe that most games do not see large changes in the number of players since most of the mass is near zero and that most games lose players as the distribution turns more towards the left. The median of the differences is -2 and the mean is -187 as it is more affected by the outliers. We also generate equivalent plots for the third, fourth, and fifth month after the release, which all appear very similar to Figure 3.

We visualize the connection between the main predictor and the target (median player counts of the first two months) in Figure 4. As seen, the trend is almost linear. The Pearson's correlation coefficient between them is 0.998. This means that a model that predicts only the first month median can be a decent baseline. Besides the general linear correlation of the median player counts, there are also some games that open with nearly zero players in their first month and slowly increase and an inverse behavior, where some games drop to zero players in their second month. We also compute the Pearson's correlation coefficients for each predictor and the target. They are all very low, i.e. from -0.026 to 0.011, meaning that there is a very little linear correlation between them. However, they may still affect the target in a non-linear manner.

3.3 Feature transformations

The features of the past median player count, price, number of supported languages, and storage requirements are transformed using the below equation:

$$f(x) = \log\left(\frac{1+x}{\overline{x}}\right), \tag{1}$$

where x is the input feature and $\overline{x}$ is the mean of the feature in the training set. This approach is useful because our feature values are on average relatively small, but contain very large outliers (for example, the average value of the *past median player count* is roughly 1000, while its extremes range from 1 to 3,000,000). By adding 1 to our feature value, we avoid possible log-transformation problems in the following step. By dividing with the feature mean, we transform the raw feature value into a multiplier of the average observed value. Following the above, this would transform our extreme values into 3000 and 0.001 multiplier values, respectively. Finally, a log-transformation is performed

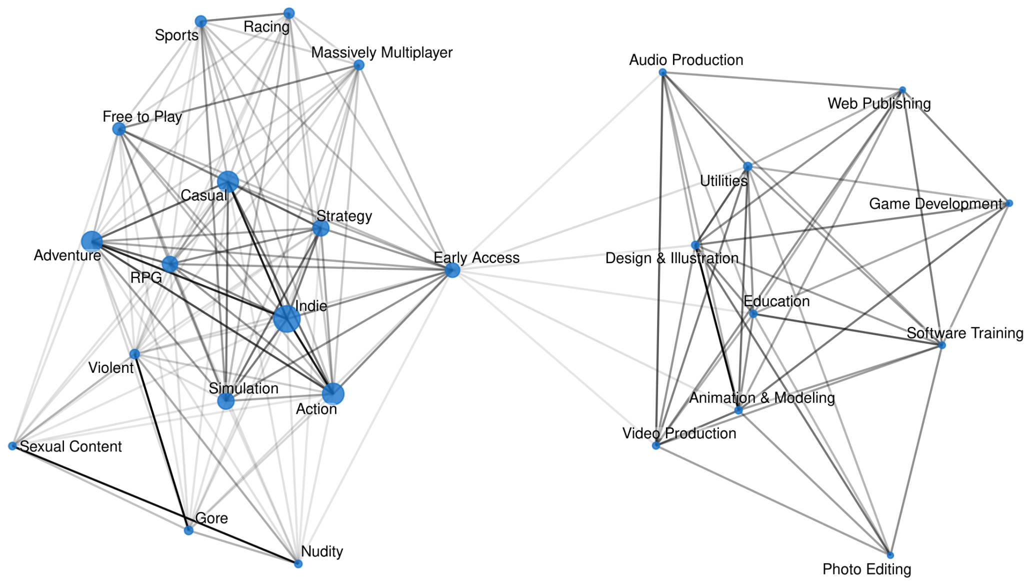


Figure 2.: **Network plot of the game genres**. The nodes represent the genres and the edges the games of both genres. The opacity of the edge is the proportion of the games the connected genres share. The node size is the number of the games in a genre.

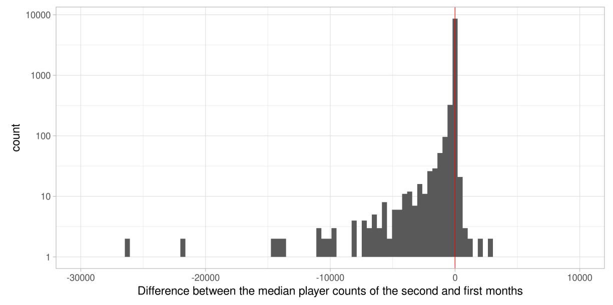


Figure 3.: Difference between the main predictor and the target, i.e. the medians of the second and first months. The y-axis is in a log scale.

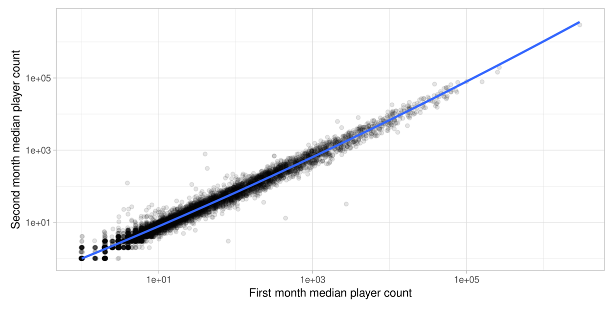


Figure 4.: Correlation between the main explanatory variate and target, i.e. medians of the second and first months. The blue line shows the trend and the dots are the games. Both axes are in the log scale.

to reduce the magnitude of our multipliers and make our features easier to model.

The value of the *release year* is treated as a categorical variable. Since outliers are not possible in the context of temporal features, we simply perform scaling, so both the *release day* and *monthly release day* are moved to the $[0, 1]$ interval. Afterwards, we replace both values with a tuple in the following manner:

$$x \mapsto [\sin(2\pi x),\ \cos(2\pi x)].$$

The replacement is performed due to the cyclical nature of the temporal features. It ensures that the games released at the beginning of the year and the games released at the end of the year have similar release day values.

3.4 Models

Given the vector of the explanatory variates for each game, we build a model predicting its median number of players in the second month after release. Given the transformed features, we construct a final feature vector containing all attributes, except for the genres, publishers, and developers. Given the final feature vector $\mathbf{x_i}$ of the i-th game, we model its target number of players y_i.

3.4.1 Normal model: We transform the target in the same manner as we do for the majority of the other features, using the transformation in Equation 1. We model the transformed real-valued target as

$$f(y)|\beta_0, \beta, \sigma^2 \sim \mathrm{N}(\beta_0 + \beta^T \mathbf{x}, \sigma^2),$$

$$\beta \sim \text{Cauchy}(0, 5),$$

$$\sigma \sim \text{Half-Cauchy}(0, 5).$$

For β, we introduce $\text{Cauchy}(0, 5)$ as a weakly-informative and zero-centered prior for not knowing neither the positive nor the negative feature effect, and as a wider-tailed distribution allowing for less penalization of larger parameters. The uncertainty in this model is assumed not to vary with different inputs and is modeled with input-invariant parameter σ having a similar weakly-informative $\text{Half-Cauchy}(0, 5)$ prior.

3.4.2 Folded normal model: We empirically determine that the normal model is sometimes unstable. As a possible solution, we do not transform the target as in Equation 1 and instead keep it in its original form. Using a normal distribution is no longer suitable, because its support is over the entire real line, whereas our target is now non-negative. To account for this, we look for a distribution that assigns a nonzero density to all points in $[0, \infty)$. Having a nonzero density at zero, we consider games with no players.

The folded normal distribution is one such candidate. It is based on the normal distribution, but only has support on the non-negative reals with a positive density at zero. It generalizes the half-normal distribution in allowing the point with the highest density to be different from zero.

Given that (i) our transformed target measurements is non-negative, (ii) games with no players are possible, (iii) unpopular games are more probable, and (iv) most probable prediction is not necessarily zero for every game in our dataset, we incorporate the folded normal distribution to model our prediction uncertainty for each game. Formally, probability density function (PDF) f_{FN} of folded normal distribution $\text{FN}(\mu, \sigma^2)$ parametrized with $\mu \in \mathbb{R}$ and $\sigma^2 \in \mathbb{R}^+$ is defined as:

$$\begin{aligned} f_{FN}(x|\mu, \sigma^2) =& \frac{1}{\sqrt{2\pi\sigma^2}} \left(e^{-\frac{(x-\mu)^2}{2\sigma^2}} + e^{-\frac{(x+\mu)^2}{2\sigma^2}} \right) = \\ =& f_N(x|\mu, \sigma^2) + f_N(-x|\mu, \sigma^2), \end{aligned}$$

where $f_N(x)$ is PDF of the normal distribution and x is an arbitrary non-negative real. We transform the location parameter with near-linear function $h(x) = \log(1 + e^x)$ for a better model convergence and we model target y as

$$y|\beta_0, \beta, \sigma^2 \sim \text{FN}(\log(1 + e^{\beta_0 + \beta^T \mathbf{x}}), \sigma^2),$$

$$\beta \sim \text{Cauchy}(0, 5),$$

$$\sigma \sim \text{Half-Cauchy}(0, 5).$$

We use the same priors on β and σ as for the normal model, following the same reasoning.

In the preliminary testing, we find the varying order of magnitude of the target to be problematic. Targets range from zero to more than a million players, which may lead to convergence difficulties during model training. As a solution, we replace target y with $\log(1 + y)$. This keeps the target on the same interval, but makes its distribution significantly less skewed. Our model is thus trained with the transformed target[†], however, we may still transform the target back to the original space to simplify the interpretation.

3.4.3 Hierarchical folded normal model: As said above, the publisher and developer variables are unsuitable for the model because of the large number of unique values and difficulties in meaningfully transforming them. On the other hand, the genre variable is more manageable. If a game belongs to some particular genres, then their general properties can be used to further improve predictions for the game. We implement this idea by adding a hierarchical component to the folded normal model.

We associate the distinct β_0 coefficients to different genres. If $\beta_{0,j}$ denotes the corresponding coefficient of genre j and $\overline{\beta_{0,G_i}} = \frac{1}{|G_i|} \sum_{j \in G_i} \beta_{0,j}$ denotes the coefficient mean over genres in genre set G_i of game i, then the proposed target attribute of the hierarchical model is:

$$y_i|\overline{\beta_{0,G_i}}, \beta, \mathbf{x_i}, \sigma^2 \sim \text{FN}(\log(1 + e^{\overline{\beta_{0,G_i}} + \beta^T \mathbf{x_i}}), \sigma^2),$$

$$\beta_{0,j}|\beta_0, \sigma_{\beta,0} \sim \text{N}(\beta_0, \sigma_{\beta,0}),$$

$$\beta_0, \beta \sim \text{Cauchy}(0, 1),$$

$$\sigma_{\beta,0} \sim \text{Half-Cauchy}(0, 1),$$

$$\sigma \sim \text{Half-Cauchy}(0, 5).$$

We link the individual genre-specific coefficients in a hierarchical structure by imposing $\beta_{0,j}|\beta_0, \sigma_{\beta,0} \sim \text{N}(\beta_0, \sigma_{\beta,0})$. Similarly as above, we impose $\text{Cauchy}(0, 1)$ priors on β_0 and β, and a $\text{Half-Cauchy}(0, 1)$ prior on $\sigma_{\beta,0}$. Again, we use $\log(1 + y)$ as the target.

3.4.4 Heteroscedastic models: The predictive variance plays an important role because it estimates the uncertainty in our predictions. The shared variance after transforming the target indicates how much the predicted player count will vary when considering the order of magnitude. For example, suppose we transform the target with $\log(1+y)$ and observe $\sigma^2 = 1$. Now consider two games with 10 and 10^4 predicted players each, and use a base-10 logarithm for ease of understanding. Generally speaking, this particular variance implies that the predicted player count will vary somewhere between 1 and 100 for the first game. The count will vary between 10^3 and 10^5 for the second game. This shows how uncertain the model is relative to the prediction magnitude.

[†]We say that $\log(1 + y)$ is distributed according to the folded normal, where y is the player count.

Though this approach is generally useful, a single scalar is not very flexible and may yield unreasonably high uncertainty estimates for some games with many players. It is better to consider features of each game and use them to compute its particular variance. With this approach the model provides better results for individual games and is still flexible enough to learn a single shared scalar if needed. We refer to models which make use of this idea as heteroscedastic.

We thus replace the σ^2 parameter of all the three models with the function of the game features. More precisely, we use $e^{\gamma_0+\gamma^T\mathbf{x}}$ for the normal and the folded normal model, and $e^{\overline{\gamma_{0,G_i}}+\gamma^T\mathbf{x_i}}$ for the hierarchical normal model with $\gamma_{0,j}|\gamma_0,\sigma_{\gamma,0} \sim N(\gamma_0,\sigma_{\gamma,0})$. This approach yields three new models with an attribute-dependent variance. The priors for the γ coefficients are the same as those for the β coefficients in the homoscedastic models. The heteroscedastic hierarchical folded normal model is the most complex of the three. We show it in Figure 5.

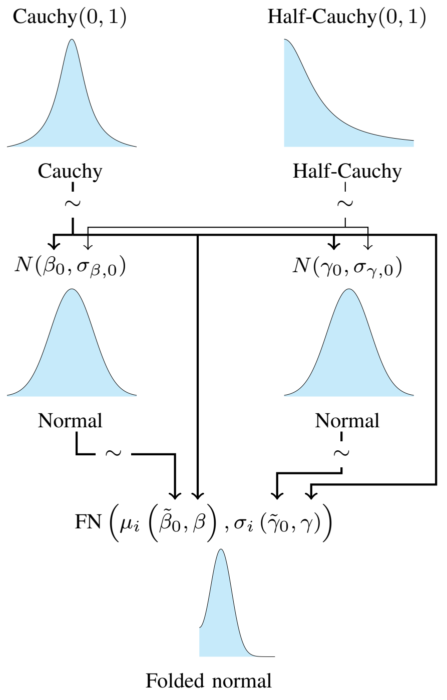


Figure 5.: Heteroscedastic hierarchical folded normal model where for conciseness we write: $\mu_i(\tilde{\beta}_0,\beta) = \log\left(1+e^{\frac{1}{|G_i|}\sum_{j\in G_i}\beta_{0,j}+\beta^T\mathbf{x_i}}\right)$ and $\sigma_i(\tilde{\gamma}_0,\gamma) = e^{\frac{1}{|G_i|}\sum_{j\in G_i}\gamma_{0,j}+\gamma^T\mathbf{x_i}}$.

4 Results

4.1 Visual check of the predictive distributions

By sampling from the posterior, we compute different game-specific parameters μ and σ which correspond to a distribution that describes the median player count for that game. We show some predictions with the heteroscedastic hierarchical folded normal model in Figure 6.

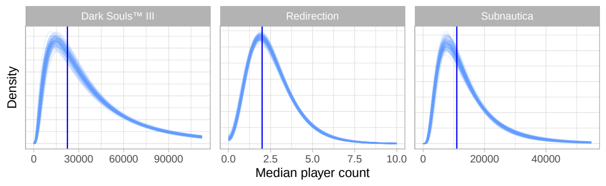


Figure 6.: Predicted median player count distributions using the hierarchical folded normal. Vertical lines are the targets. Dark Souls$^{\text{TM}}$ has the highest overall budget, followed by Subnautica, and Redirection. Based on these examples, the model performs well for the games with such differences.

4.2 Model comparison

We evaluate and compare the models performance with an approximation of the leave-one-out cross-validation (LOOCV) technique, the Pareto smoothed Importance sampling leave-one-out information criterion (PSIS-LOOIC), first presented by Vehtari et al. in 2016 [11]. We decide use the LOOCV technique instead of a simple holdout estimation to get more accurate results. LOOCV is an exhaustive model evaluation technique. It requires refitting the model once per each data instance (infeasible on a big dataset like ours). Therefore, we approximate it with LOOIC, i.e. a fast, robust and stable model evaluation method, based on the log-likelihood of the posterior at the actual target, and designed specifically for Bayesian models.

For each data instance, LOOIC approximates the expected log-predictive density (*elpd*) of the model fitted on the dataset from which the selected game would be removed.* The LOOIC value is then computed as LOOIC $= -2 \cdot elpd$ to get on the deviance scale. The principal advantage that LOOIC provides is its low time complexity. While LOOCV requires refitting the model N times (with N the size of the dataset), LOOIC requires only one evaluation of the model. We show the models performances over time in Figure 7. All the models exhibit a similar (and expected) behavior, with their performance dropping when predicting player counts further in the future. We present a proper pairwise comparison between the models in Figure 8. The heteroscedastic hierarchical model (which is the most complex one) turns out to be the best performing model. The other models might provide viable alternatives if a faster fitting process is used.

*PSIS-LOOIC is thoroughly described by Vehtari et al. (2016) [11] and Vehtari et al. (2002) [12].

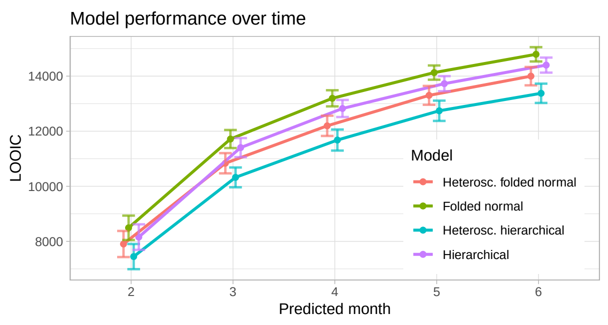


Figure 7.: LOOIC estimated for multiple predicted months. For each model-predicted month pair, we show the LOOIC estimate and its standard error. The performance of the models expectedly drops over time, with the greatest decrease between the 2nd and 3rd month (notice that the lower the LOOIC estimate is the better is the model performing). We include multiple models on this plot to give the reader the idea of their performances, but no conclusion about which one is better should be made from it.

4.3 *Coefficient interpretation*

We analyze the posterior distributions of the sampled coefficients. Because different features use somewhat different scales, we do not claim that a certain feature is more important in predicting the outcome than the other.[†] However, we quantify the contribution of each feature towards the computed parameters of the folded normal distribution. We focus on the heteroscedastic folded normal model. The posterior for this model is presented in Table 2. Since the hierarchical variant of the model has a very similar posterior, we only analyze the genre-specific intercepts.

First, we consider the main non-negative features, i.e. the price, number of supported languages, system requirements, and median player count. It should be noted that the folded normal model and its hierarchical variant use the exponential of the dot product to compute the transformation-specific parameter (the mean and the variance). We use an intercept term as a reference, so each coefficient individually affects the dot product in an additive manner. Since the four considered features are all transformed using $\log\left(\frac{1+x}{\overline{x}}\right) = \log(1+x) - \log(\overline{x})$, coefficient β_i causes one of the following changes in the computed parameter:

- if $\beta_i < 0$, the dot product value is decreased by $|\beta_i| \log(1+x)$, then increased by $|\beta_i| \log(\overline{x})$;
- if $\beta_i > 0$, the dot product value is increased by $|\beta_i| \log(1+x)$, then decreased by $|\beta_i| \log(\overline{x})$;
- if $\beta_i = 0$, then there is no change in the dot product value.

[†]We further analyzed these features to see if they could be transformed to a common distribution, which would make the posteriors of the coefficients easier to compare. Since no pair of features could meaningfully be transformed to distributions of similar shapes, we concluded that pairwise comparisons would likely be unreliable and did not pursue this any further.

Note that $\overline{x} > 1$ for all four features, so each $\log(\overline{x})$ term is strictly positive. We can treat all such terms as constants and focus only on $|\beta_i| \log(1+x)$, which are different across games.[‡] We show their contributions in Figure 9. The median player count feature is associated with an increase in the mean and variance parameters, but is omitted for a clearer visualization. Its contribution is larger than that of the other three features, which is evident from the above shown side-by-side correlation plot with the target (Figure 4).

The temporal features also affect the computed parameters. In a similar manner as above, we examine the impact of the *release day* feature on the computed parameters in Figure 10.

The posterior for the *monthly release day* coefficient is negative for the mean, which suggests that releasing a game later in the month decreases the number of players. According to the model, releasing the game on the 31st day of a month is associated with an expected decrease of the folded normal mean by -0.682, which is non-negligible considering the contributions of other features. We did not find a clear explanation for this phenomenon, however we still use the feature as it improves model predictions.

The heteroscedastic folded normal uses individual β_0 and γ_0 intercepts to compute its mean and variance parameters. On the other hand, the hierarchical variant considers β_0 and γ_0 parameters for each genre individually, then computes the average over all of the game's genres. Each genre thus influences predictions to a different degree. A visualization of these coefficients is presented in Figure 11.

Since we do not use a GLM in its proper sense, we cannot state how the model predictions behave after we change some of the input features. To provide some intuition, we show such changes for a particular game in Figure 12.

5 Conclusion

We implement and evaluate models to predict the popularity of games on Steam in early stages after their release and analyze the effect of several other features. We represent the game popularity with a number of persons playing a game over time.

We manually collect the game data from different sources and construct various features through different

[‡]If we explicitly account for the four feature means $\overline{x}$ by subtracting them from the base intercept, we arrive at an adjusted intercept $\widetilde{\beta_0}$ with mean 1.443 and 90% confidence interval (CI) $(1.326, 1.555)$. The corresponding result for the variance is $\widetilde{\gamma_0}$ with mean -0.670 and 90% CI $(-0.714, -0.425)$.

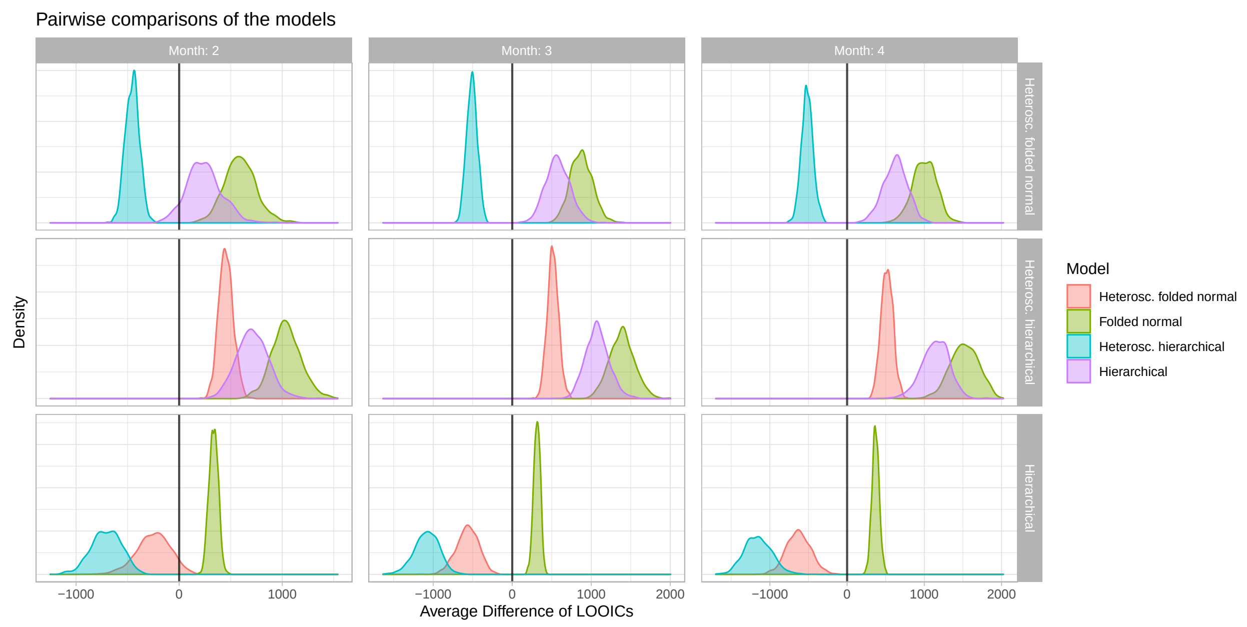


Figure 8.: Distribution of the LOOIC differences between the models. We compute the differences by subtracting the pointwise LOOIC estimations of the compared models. We obtain the distributions by bootstrapping the computed differences. Each model should be compared only with that line's reference model (multiple models are shown simultaneously just for convenience). For a selected model and the reference to which we compare it, the percentage of the area under its density curve that lies on the left of the $x = 0$ line is the confidence with which we can claim the selected model is better than the reference. For example, from the visualization, we can conclude that irrespective of the predicted month, the heteroscedastic hierarchical model is almost surely the best one. Furthermore, we can be almost sure that the difference between the heteroscedastic folded normal and heteroscedastic hierarchical models LOOIC is at least 200.

Feature		μ	q_5	q_{95}		μ	q_5	q_{95}
Intercept	β_0	7.583	7.539	7.627	γ_0	-0.259	-0.316	-0.203
Price	β_1	-0.001	-0.002	0.000	γ_1	-0.002	-0.003	0.000
Number of languages	β_2	-0.039	-0.049	-0.029	γ_2	-0.068	-0.082	-0.054
Storage requirements	β_3	0.013	0.000	0.027	γ_3	-0.026	-0.044	-0.007
Median players feature	β_4	0.988	0.982	0.994	γ_4	0.102	0.094	0.110
Day of month	β_5	-0.022	-0.030	-0.016	γ_5	-0.060	-0.070	-0.051
Day of year, cosine	β_6	0.055	0.041	0.070	γ_6	0.057	0.037	0.076
Day of year, sine	β_7	-0.005	-0.018	0.010	γ_7	0.068	0.049	0.088

Table 2.: Coefficients to compute the mean (using β_i) and variance (using γ_i) of the folded normal distribution. The columns represent the posterior mean, the 5th, and 95th percentiles. The 90% confidence intervals for the price, storage requirements, and the day of the year (sine component) include 0, so their contribution to the mean and the variance is probably negligible.

transformations. Our main prediction target is the median player count in the second month after a game release. Our experiments show that the models prediction performance for later periods of time decreases. Since our main aim is to understand the features effect, our focus is mostly on the second month target which is the easiest to model. We also notice that models not using the median player count of the first month as a feature do not converge, probably because the model predicts the player counts in absolute terms and requires a good starting point, which is then shifted to obtain the prediction. This issue could be resolved by using additional features to identify the starting point or by redesigning the model to predict relative changes in the player count as a percentage instead of an absolute value. There is no guarantee that such approach would solve all the problems, because some games would still experience unexpectedly quick changes in the number of players, but it could serve as a good starting point.

We evaluate Bayesian normal, folded normal, and

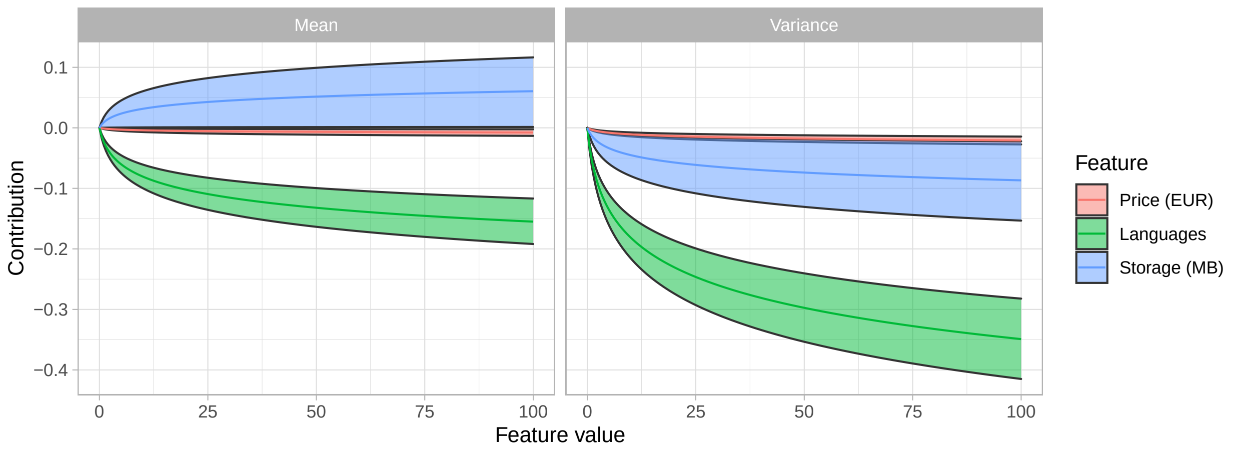


Figure 9.: Parameter contribution plot for three of the four main features. The median player-count feature is omitted for an easier visualization. The parameter contribution is computed by multiplying the parameter with x, which is a feature in the original space, i.e. before the $\log(1 + x)$ transformation. Each feature is represented by its mean contribution and a 90% confidence interval across different values. The left plot shows that as the number of the languages increases, the dot product, and thus the mean of the folded normal, decrease. On the other hand, increasing the storage is associated with an increase in the dot product and consequently the mean. On the right plot, we see that all three features cause a drop in the variance. The price feature has an almost negligible effect on the computed parameters.

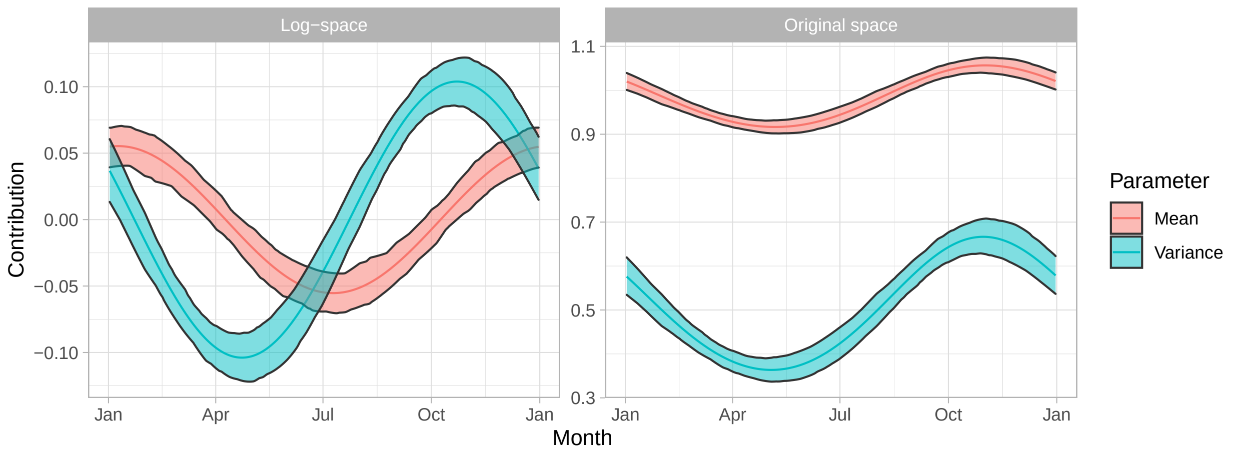
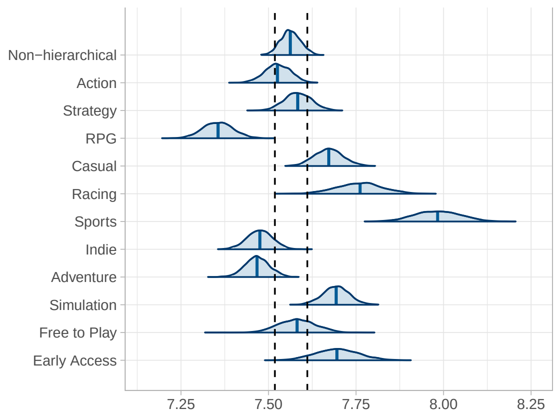
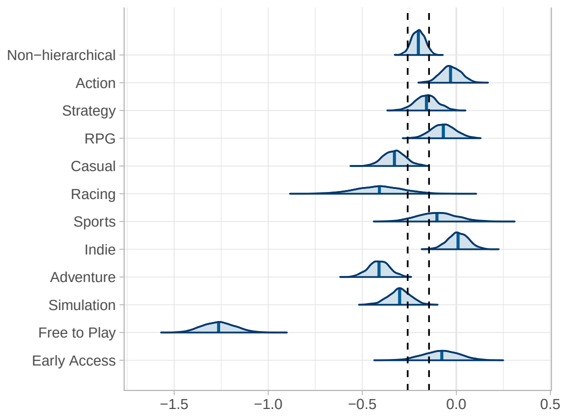


Figure 10.: Parameter contribution plot for the *release-day* feature. The plot on the left represents the parameter contribution in log-space, whereas the plot on the right is its transformation to the original space with the actual player counts. Releasing a game around January is associated with the highest increase in the mean parameter, whereas releasing it around July is associated with the highest decrease. In log-space (i.e. referring to the folded normal parameters), the variance parameter follows a similar pattern, shifted a few months backwards. In the original space (i.e. considering actual player counts), the mean and the variance are aligned. We can see that the highest contribution is roughly between October and January, and the lowest roughly between May and August. A seasonal behavior is observed for movie releases [13], where the peak release counts occur before holidays. Our data includes the total player counts from all countries where Steam is available, so national holidays do not present a significant contribution. A possible explanation for the peak is that publishers release games before the globally celebrated New Year's holidays, so the games are available during large sales and thus gain more visibility.



(a) Mean intercepts



(b) Variance intercepts

Figure 11.: Visualization of the genre-specific intercepts for computing the mean (left) and variance (right) of the heteroscedastic hierarchical folded normal model. Dashed lines represent the $[q_5, q_{95}]$ interval for the *non-hierarchical* heteroscedastic folded normal posterior. Some genres were skipped due to high-variance posteriors. The left plot indicates that RPG and indie games tend to a smaller intercept, whereas sports, racing, simulation, and possibly casual games attain a higher intercept. In the right plot, we see that most genres are similar to the regular model. The exceptions are free to play games, which have a significantly smaller variance and thus result in more stable predictions. The predictions for adventure games are also more stable with this model, whereas the variance of predictions for indie games is somewhat greater. In conclusion, the heteroscedastic hierarchical model estimates the folded normal mean more precisely for these genres, because their distributions are clearly different from the regular model and also achieve a reasonably small variance. The model also helps by reducing variance in free to play game predictions. From a simple visual check, the variance of other genres revolves around the regular model's variance and does negatively affect predictive performance by comparison.

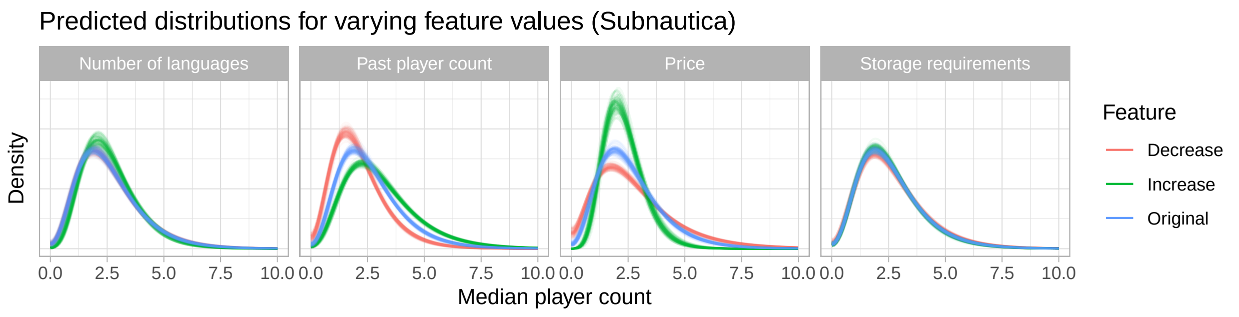


Figure 12.: Intuition of how changing the feature values affects the predicted distributions for the Subnautica game. WA heteroscedastic hierarchical folded normal model is used for predictions. Listed as triples, decreased, original, and increased feature values are (1, 18, 29) for the number of languages; (20000, 27143, 40000) for the past median player count; (0 EUR, 21.2 EUR, 60 EUR) for the price; (10 GB, 20 GB, 100 GB) for the storage requirements. In this example, changing the number of languages and the storage has a very small effect on the generated distributions. Changing the price results in noticeably different predictions. As said above, the target variable is heavily correlated with the past median player count feature, so the changes in the generated distributions are the largest here.

hierarchical folded normal models in both the homo- and heteroscedastic variants for this task. We augment the baseline, i.e. the first month median player count with other features and then examine their impact based on their posterior distributions. The heteroscedastic hierarchical model achieves the best performance with respect to LOOIC. The most important predictor in each model is the median player count, which makes sense as it is also a strong baseline. However, adding additional features improves the models and enables analyzing and interpreting their effect.

We see a great potential for future work in the area, especially in its analyzing. Gathering and adding more game features will improve the models substantially and will make it easier as Steam will gather and offer more data.

Another similar improvement would be to perform a thorough analysis of the missing values of player counts and incorporate them into the models. A more complex and difficult goal to attain would be to model the player counts from and to any arbitrary pair of points in time. This would enable a much deeper level of understanding how the play popularity changes over time.

A possible enhancement likely to improve the modeling approach would be adding other popularity-related features such as Twitch views, Google searches, reviews, etc. To fully utilize reviews, it is also possible to use natural language processing to extract certain features. Seeing that predicting a game's popularity accurately would be very valuable for publishers or even streamers, we see a possible practical application for predictive models in this area.

References

[1] Joseph Alexander Budiarto. Game popularity tracking system. *International Journal of Industrial Research and Applied Engineering*, 3(2):79–85, 2018.

[2] Sangho Ahn, Juyoung Kang, and Sangun Park. What makes the difference between popular games and unpopular games? analysis of online game reviews from steam platform using word2vec and bass model. *ICIC Express Letters*, 11(12):1729–1737, 2017.

[3] Dayi Lin, Cor-Paul Bezemer, Ying Zou, and Ahmed E Hassan. An empirical study of game reviews on the steam platform. *Empirical Software Engineering*, 24(1):170–207, 2019.

[4] Dayi Lin, Cor-Paul Bezemer, and Ahmed E Hassan. An empirical study of early access games on the steam platform. *Empirical Software Engineering*, 23(2):771–799, 2018.

[5] SteamSpy. https://steamspy.com. Accessed: 2020-12-17.

[6] SteamDB. https://steamdb.info. Accessed: 2020-12-18.

[7] Hongju Liu. Dynamics of pricing in the video game console market: Skimming or penetration? *Journal of Marketing Research - J MARKET RES-CHICAGO*, 47:428–443, 06 2010.

[8] Market research: The state of online gaming – 2020. https://www.limelight.com/resources/white-paper/state-of-online-gaming-2020/. Accessed: 2021-02-21.

[9] Anthony Palomba. Digital seasons: How time of the year may shift video game play habits. *Entertainment Computing*, 30:100296, 03 2019.

[10] Jacob Ratkiewicz, Santo Fortunato, Alessandro Flammini, Filippo Menczer, and Alessandro Vespignani. Characterizing and modeling the dynamics of online popularity. *Physical review letters*, 105(15):158701, 2010.

[11] Aki Vehtari, Andrew Gelman, and Jonah Gabry. Practical bayesian model evaluation using leave-one-out cross-validation and waic. *Statistics and Computing*, 27(5):1413–1432, Aug 2016.

[12] Aki Vehtari and Jouko Lampinen. Bayesian model assessment and comparison using cross-validation predictive densities. *Neural computation*, 14:2439–68, 11 2002.

[13] Sonja Radas and Steven M Shugan. Seasonal marketing and timing new product introductions. *Journal of Marketing Research*, 35(3):296–315, 1998.

Andraž De Luisa is a Data Science Master's student at the Faculty of Computer and Information Science, University of Ljubljana. His current research interests include Bayesian statistics and machine learning.

Jan Hartman is a Data Science Master's student at the Faculty of Computer and Information Science, University of Ljubljana. His current research interests include large scale machine learning and neural network optimization.

David Nabergoj is a Data Science Master's student at the Faculty of Computer and Information Science, University of Ljubljana. His current research interests are deep learning and Bayesian statistics.

Samo Pahor is a Data Science Master's student at the Faculty of Computer and Information Science, University of Ljubljana. His current research interests include data scraping and machine learning.

Marko Rus is a Data Science Master's student at the Faculty of Computer and Information Science, University of Ljubljana. His current research interests are in computer vision.

Bozhidar Stevanoski is a Data Science Master's student at the Faculty of Computer and Information Science, University of Ljubljana. His current research interests include data stream mining and multi-target prediction.

Jure Demšar is an assistant professor at the Faculty of Computer and Information Science, University of Ljubljana and a researcher at the Department of Psychology, Faculty of Arts, University of Ljubljana. His research interests lie in machine learning, Bayesian statistics and neuroinformatics.

Erik Štrumbelj is an associate professor at the Faculty of Computer and Information Science, University of Ljubljana. His main research interests lie in Bayesian statistics and machine learning.