Time Series Analysis on the Rating Trend of Genshin Impact on Two Representative Platforms

Zishi Wang
Center of Data Science
New York University
New York, NY, 10011
zw3254@nyu.edu

Jinqian Pan
Center of Data Science
New York University
New York, NY, 10011
jp6218@nyu.edu

1 Introduction

Genshin Impact is a fantasy-themed open-world action role-playing game released in September 2020 by miHoYo, an emerging Chinese video game development and animation studio. This video game made an estimated 2 billion dollars solely from mobile phones in its first year of release [1]. According to the latest statistics from an app monitoring company Sensor Tower, it was the third highest-earning mobile game in the globe in previous 12 months, only surpassed by Honour of Kings and PUBG Mobile [1].

Genshin Impact is such a unique game that, in addition to its immense financial success, its ratings vary drastically across different gaming forums. In its early days, IGN, the world's leading gaming and entertainment publication, gave it a 9 out of 10 [2]. In the Google Play store, it got a 4.1 out of 5. However, in Bilibili Game Center, one of the most popular gaming forum in China, it only scored a 5.9 out of 10. Thus, we want to analyze Genshin Impact's rating trend on these two platforms and predict its ratings in the recent two months. Moreover, we want to consider a classic and enduring operation research question: how can we produce better results with fewer resources? Thus, we are also going to test if a model can make an acceptable prediction with a limited number of data points.

2 Problem Definition and Algorithms

2.1 Task

We have two main difficulties with this project. First and foremost, can we use the information gained throughout the training to create models that forecast the Genshin Impact rating trend over the recent two months? Is it possible to train models using a smaller training set without having a substantial influence on the models we developed in the first question?

We would leverage the ratings data from Bilibili Game Center and Google Play to tackle these two problems and our fundamental methods to model rating trend in this project are ARIMA and Gaussian Process. The findings will include line graphs that depict the predicted outcomes of two models.

2.2 Algorithm

ARIMA model is composed of autoregressive model and moving average model.

According to Time series analysis and its applications, the theory behind the **autoregressive (AR) model** [3] is that the current value of a series, x_t, may be described as a function of p previous values, $x_{t-1}, x_{t-2}, \ldots, x_{t-p}$, where p is the number of steps into the past required to anticipate current value. We have assumed that current value is a linear function of previous values.

Definition: An autoregressive model of order p, abbreviated $\mathbf{AR}(p)$, is of the form

$$x_t = \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t,$$

where x_t is stationary, $w_t \sim \mathcal{N}\left(0, \sigma_w^2\right)$, and $\phi_1, \phi_2, \ldots, \phi_p$ are constants ($\phi_p \neq 0$). The mean of x_t is zero. If the mean, μ, of x_t is not zero, replace x_t by $x_t - \mu$,

$$x_t - \mu = \phi_1\left(x_{t-1} - \mu\right) + \phi_2\left(x_{t-2} - \mu\right) + \cdots + \phi_p\left(x_{t-p} - \mu\right) + w_t,$$

or write

$$x_t = \alpha + \phi_1 x_{t-1} + \phi_2 x_{t-2} + \cdots + \phi_p x_{t-p} + w_t,$$

where $\alpha = \mu\left(1 - \phi_1 - \cdots - \phi_p\right)$.

The **moving average (MA) model** [3] of order q, abbreviated as $\mathbf{MA}(q)$, assumes the white noise w_t on the right-hand side of the defining equation are combined linearly to form the observed data as an alternative to the autoregressive representation in which the x_t on the left-hand side of the equation are assumed to be combined linearly.

Definition: The moving average model of order q, or $\mathbf{MA}(q)$ model, is defined to be

$$x_t = w_t + \theta_1 w_{t-1} + \theta_2 w_{t-2} + \cdots + \theta_q w_{t-q},$$

where $w_t \sim \mathcal{N}\left(0, \sigma_w^2\right)$, and $\theta_1, \theta_2, \ldots, \theta_q$ $\left(\theta_q \neq 0\right)$ are parameters.

So that the full **ARMA model** could be written as:

$$x_t - \lambda_1 x_{t-1} - \ldots \lambda_p x_{t-p} = w_t + \theta_1 w_{t-1} + \cdots + \theta_q w_{t-q}$$

Same as the previous three models, **autoregressive integrated moving average (ARIMA) model** is also based on a smooth time series. The models above can be seen as the special forms of ARIMA. It is expressed as $\mathbf{ARIMA}(p, d, q)$, where p is the autoregressive order, q is the moving average order, and d is the number of differences made when the time becomes smooth, which is the meaning of the word “Integrate" here.

In the traditional method, the values of p, d, and q for ARIMA model would be decided based on the autocorrelation function (ACF) and partial autocorrelation function (PACF) graphs. Our choice is to make our decision by evaluating the quality of models with Akaike information criterion (AIC).

AIC is a predictor of prediction error and hence relative model quality for a given set of data [4]. It measures the quality of each model in relation to other models and thus may be used to tune hyperparameters. Thus, we set the upper and lower bounds for hyperparameters, compute AIC for each model, and go with the model with the lowest AIC.

Gaussian Process Model is a probability distribution across the many functions that can be used to fit a set of points [5].

We can compute the means as the function and the variances to indicate how confident the forecasts are since we have the probability distribution across all potential functions. The main points are: (1) the function (posteriors) updates with new observations, (2) a Gaussian process model is a probability distribution over possible functions, and any finite sample of functions is jointly Gaussian distributed, and (3) the mean function calculated from the posterior distribution of possible functions is the function used for regression predictions [5].

Here is the definition in the book Gaussian Processes for Machine Learning [6]:

Definition: A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.

A Gaussian process is completely specified by its mean function and covariance function. We define mean function $m(\mathbf{x})$ and the covariance function $k\left(\mathbf{x}, \mathbf{x}^{\prime}\right)$ of a real process $f(\mathbf{x})$ as

$$\begin{aligned} m(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})] \\ k\left(\mathbf{x}, \mathbf{x}^{\prime}\right) &= \mathbb{E}\left[(f(\mathbf{x}) - m(\mathbf{x}))\left(f\left(\mathbf{x}^{\prime}\right) - m\left(\mathbf{x}^{\prime}\right)\right)\right] \end{aligned}$$

and will write the Gaussian process as

$$f(\mathbf{x}) \sim \mathcal{GP}\left(m(\mathbf{x}), k\left(\mathbf{x}, \mathbf{x}'\right)\right).$$

Usually, for notational simplicity we will take the mean function to be zero [6]. k defines the form (smoothness) of the Gaussian processes model, which is a distribution over functions. If the kernel considers points $\mathbf{x}$ and $\mathbf{x}'$ to be comparable, the function outputs of the two points, $f(\mathbf{x})$ and $f(\mathbf{x}')$, should be similar. Figure 1 depicts the procedure for doing regressions using the Gaussian processes model: given observable data (red points) and a mean function f (blue line) derived from these observed data points, we generate predictions at new points $\mathbf{x}_1$ as $\mathbf{y}_1$ [5].

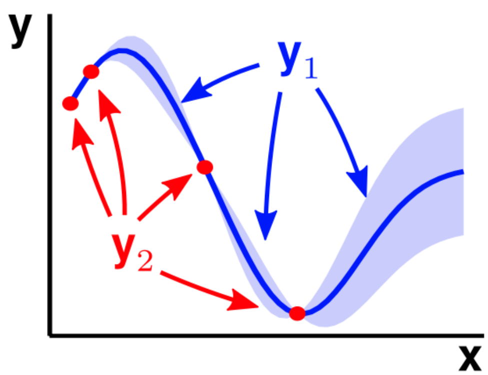

Figure 1: A diagram showing how to use Gaussian processes to execute regressions. The red dots represent observed data, the blue line represents the mean function inferred from the observed data points, and fresh blue points will be used to make predictions [5].

3 Experimental Evaluation

3.1 Data

We collect around 0.3 million reviews respectively from Bilibili Game Center and Google Play to compute daily average ratings for our training and test sets. For the data in Bilibili Game Center, we build a web scraper by ourselves with Python package requests and parse the raw data in json format to get our desired information, which are ratings and review time. For the data in Google Play, we directly use an existing package called "google-play-scraper" [7]. After providing the suffix of link to the review page and specifying country and language, we can get all US users' reviews and ratings.

Besides the regular pre-processing such as dropping duplicates and resetting index, three additional steps are introduced: (1) we remove all the reviews before the initial release of this game; (2) we unify the rating scales of two platforms to 5 for a direct comparison; (3) we convert the time in string format to timestamp for visualizations and only keep the date to compute daily average ratings.

3.2 Methodology

The criteria that we use to evaluate our models is mean squared error (MSE). It is the standard to compare the prediction performance of two models. For train-test split, since one of our goals is to explore if a model can make an acceptable prediction with a limited number of data points, we have two distinct ways to split our data.

The first method is to build the test set with ratings in the recent two months and build the training set with the rest data, which has a test ratio of approximately 0.15. To be more specific, the training set consists of the daily average ratings of Genshin Impact on two platforms from its initial release date (September 28, 2020) to October 11, 2021, while the test set consists of its daily average ratings from October 12, 2021 to December 11, 2021.

The second method is related to the update cycle of Genshin Impact. There are two main events in each version of this game and each event lasts for 20 days [8]. Thus, in order to fully reflect players' attitudes towards an event, we decide to build the training set with ratings on the start and middle dates of every event until two months ago and build the test set with the rest data. In other words, we pick up a data point approximately every 10 days and build a training set with 36 daily average ratings, which is less than one-tenth of data in the training set built with the first method.

In addition, for the gaussian process model, besides commonly used white kernel and radial basis function (RBF) kernel, we add two extra kernels to introduce periodicity (exp-sine-squared) and increasing mean (dot-product) for a better extrapolation performance.

3.3 Results

For ARIMA model, we use the first method to do train-test split. The results of extrapolation are shown in Figure 2 and 3. In both cases, ARIMA model has a relatively good performance based on MSE (0.112 for Bilibili Game Center, 0.137 for Google Play) when it extrapolates. However, it suffers from a high risk of overfitting problem. From Figure 4 and 5, we can see that the interpolations given by fine-tuned ARIMA model fit exactly against the data points in training sets, which will negatively affect the performance of this model on new data.

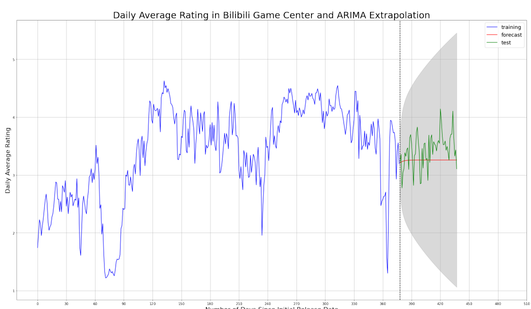

Figure 2: ARIMA Extrapolation (Bilibili)

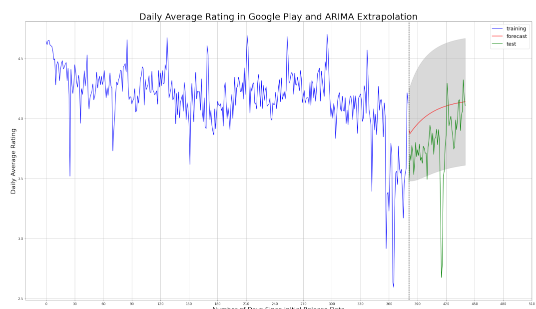

Figure 3: ARIMA Extrapolation (Google)

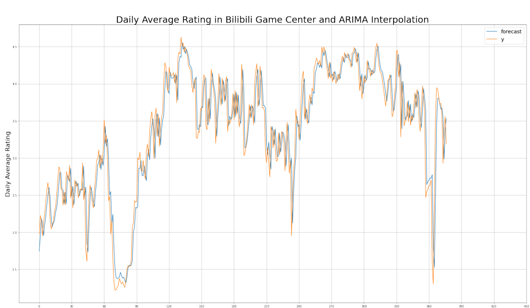

Figure 4: ARIMA Interpolation (Bilibili)

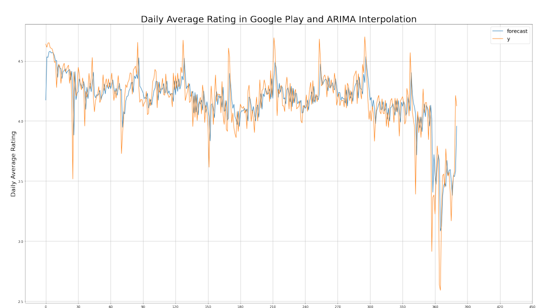

Figure 5: ARIMA Interpolation (Google)

For gaussian process model, we use the second method to do train-test split. In both cases, gaussian process model has a worse performance than ARIMA model based on MSE (1.244 for Bilibili Game Center, 0.263 for Google Play). In Figure 6, it even shows a similar sign of overfitting in the case of Bilibili Game Center. However, in the case of Google Play, although the extrapolation of gaussian process model has twice as much MSE as ARIMA model, this result is good enough for a model with training set of one-tenth size. Moreover, in Figure 7, we can see that it keeps a reasonable level of randomness when interpolating, which outperforms ARIMA model from this perspective.

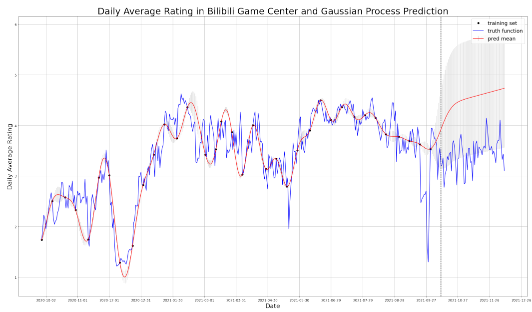

Figure 6: Gaussian Process Prediction (Bilibili)

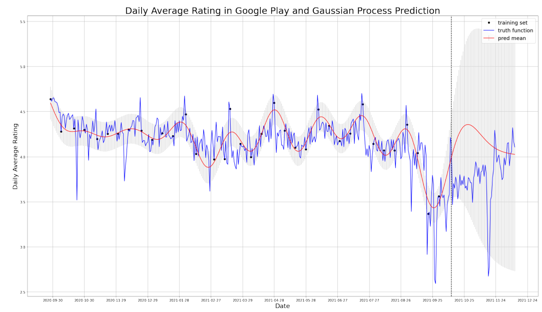

Figure 7: Gaussian Process Prediction (Google)

3.4 Discussion

As we expected, ARIMA and gaussian process are two suitable models to analyze the rating trend of Genshin Impact. We also successfully anticipate the overfitting problem of ARIMA and the relatively less demand of gaussian process for training data. The only exception is the low performance of gaussian process in the case of Bilibili Game Center. From our perspective, one of the possible reasons why gaussian process model dose not work well here is because the rating trend in Bilibili Game Center is not quite stationary and does not show a clear periodicity.

4 Conclusions and Future Work

In this project, we gain a more comprehensive understanding of the pros and cons of ARIMA and gaussian process through applying them to real-world time-series data:

(1) ARIMA has more precise extrapolations while having a dependence on sufficient training data, which makes it suffers from the risk of overfitting problem.

(2) Gaussian process is not as precise as ARIMA, but it can give an acceptable prediction interval with a small amount of data. Notice that its performance may vary greatly due to the property of data.

From the time series analysis itself, we can see that both models successfully capture the upward rating trend of Genshin Impact in the recent two months. Based on this result, we can reasonably speculate that this game will maintain a relatively good reputation and continue its financial success in its second year's operation.

If we were to continue working on this project, we have two main directions for future work:

(1) Although our models capture the general trend of ratings, the prediction is still not detailed enough. We will try to introduce more advanced modeling methods such as deep neural network to improve model performance and extend our prediction to the ratings in future two months.

(2) There are still spaces for improvement even if we stick to current two models. For ARIMA, we can find a more ideal test ratio to mitigate the negative impact of overfitting problem. For gaussian process, we can design a better combination of kernels and find the rules for setting their initial states.

References

[1] M. Baggs, "Genshin impact earns 2 billion dollars after 'unheard of' success in first year." Website, 2021. `https://www.bbc.com/news/newsbeat-58707297`.

[2] T. Northrop, "Genshin impact review." Website, 2021. `https://www.ign.com/articles/genshin-impact-review`.

[3] R. H. Shumway, D. S. Stoffer, and D. S. Stoffer, *Time series analysis and its applications*, vol. 3. Springer, 2000.

[4] P. Stoica and Y. Selen, “Model-order selection: a review of information criterion rules,” *IEEE Signal Processing Magazine*, vol. 21, no. 4, pp. 36–47, 2004.

[5] J. Wang, “An intuitive tutorial to gaussian processes regression,” *arXiv preprint arXiv:2009.10862*, 2020.

[6] C. K. Williams and C. E. Rasmussen, *Gaussian processes for machine learning*, vol. 2. MIT press Cambridge, MA, 2006.

[7] Website. `https://pypi.org/project/google-play-scraper/`.

[8] “Wishes/history.” Website, 2021. `https://genshin-impact.fandom.com/wiki/Wishes/History`.

Student Contributions

Zishi Wang:

(1) Built the web scrappers to collect data
(2) Optimized models based on our evaluation metrics
(3) Wrote sections 3 and 4 of report

Jinqian Pan:

(1) Built the prototype of models
(2) Wrote sections 1 and 2 of report

Link to Github Repository with Code and Data

https://github.com/zwang1999/DS-GA-1018-Final-Project