**Final Project**

**Marketing Analytics Application for a Game Streaming Service**

**Team 6**

INFO7374: Algorithmic Digital Marketing

Nikita Ravindran

Anugraha Rose Varkey

Rean Rodrigues

# 

# **What is a video game streaming application?**

Who does not like video games? From the very famous mario series to Fortnite, video gaming has come a long way. Don't you remember fighting for a joystick with your siblings or friends when video games allowed only two players at a time ? Now with the boom in technology, the gaming industry has leveled up its standards by allowing 100’s to 1000 players at a time to not just play but also to interact and live stream their play time.

For those unfamiliar with it, Twitch is a streaming video website where content creators attract wide audiences of viewers and subscribers by streaming themselves while they play popular video games or other entertaining content. In 2018 alone, over **1 million years** of content was consumed on Twitch, with over 4 million unique monthly streamers providing it. As you can imagine, with so many choices of what to watch or what to play, the streamers are competing with each other for viewers and dedicated subscribers.

## 

# **Objective**

## Why?

The application helps the marketing team build marketing strategies for the company. It will be useful for making data driven decisions based on the subscribers usage for the previous year and come up with a strategy for next year or campaign.

## What?

A web application with recommendation, RFE and churn information required for the marketing team which will help them view information of streamers/viewers all at one place and easier to use.

## Who?

The web app will be used by the marketing and data analysts

## How?

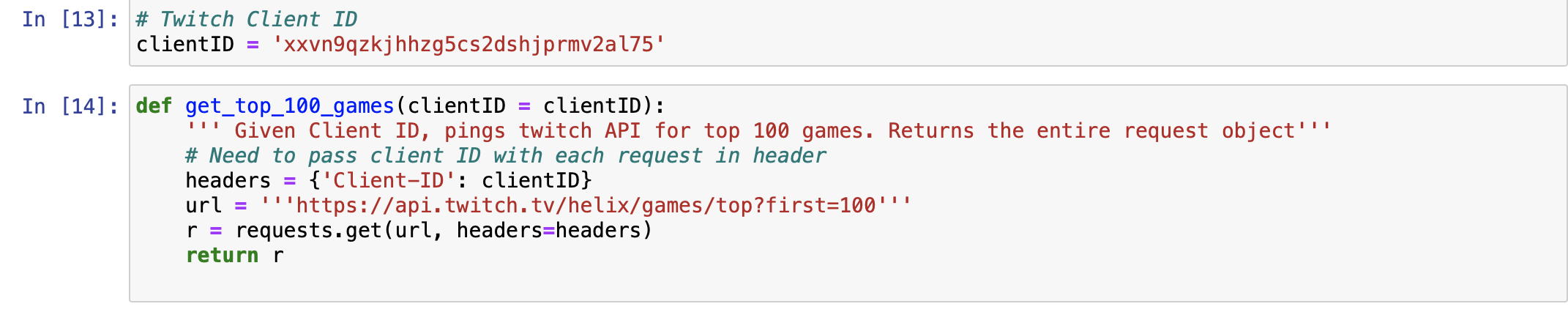
We created a pipeline on AWS with a PostgreSQL database which stores the data. The Streamlit application which is built in Python with Jupyter Notebooks is deployed on Heroku

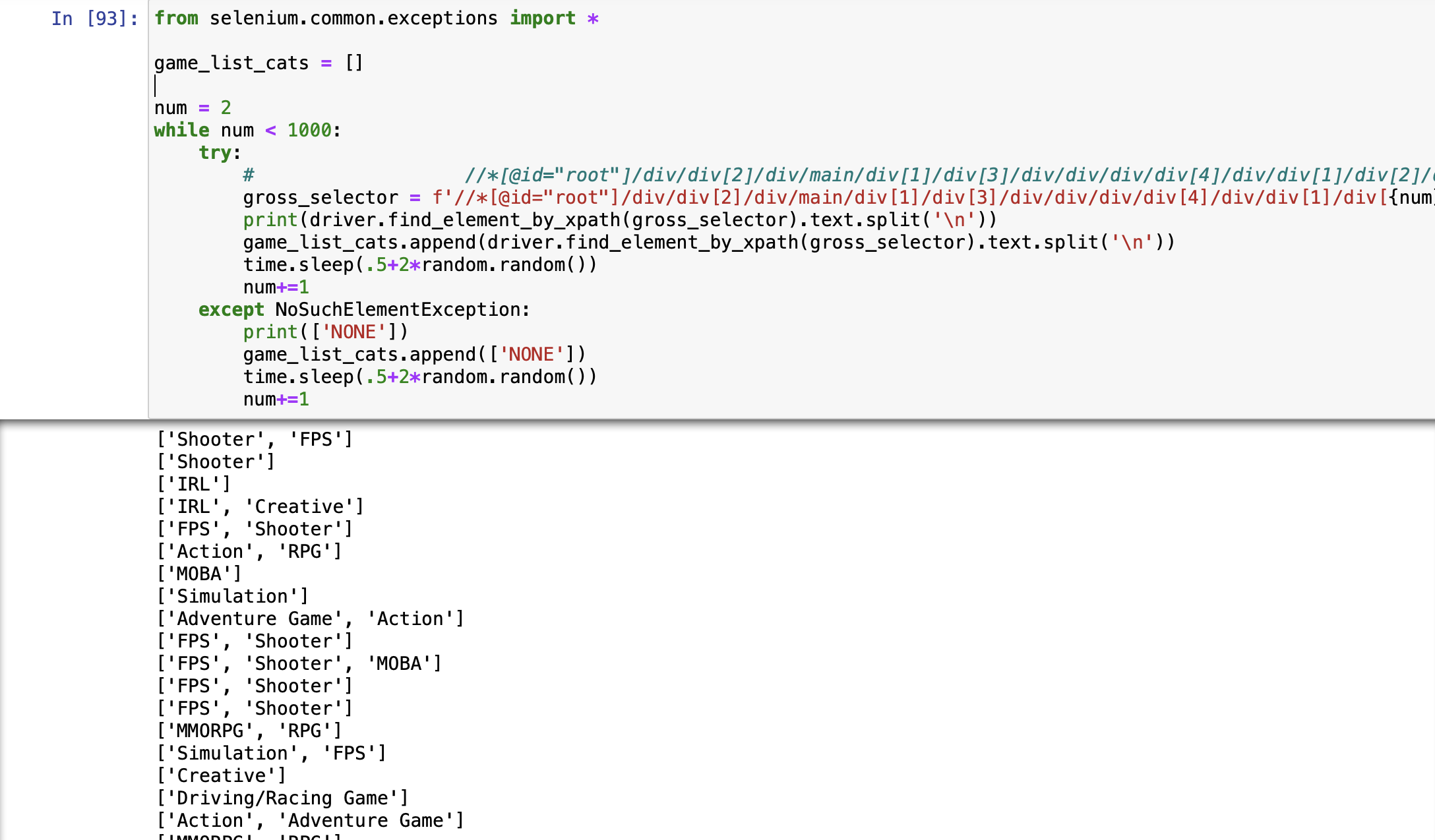
# **Data Pipeline for the Application**



### How did we get the data for video games streaming?

By 2 main ways :

* Pinged the Twitch API for streamer and game related information  
    
  
* Scraped Twitch Browse categories page for genres using selenium



Pinged the Twitch API to pull in the live stream data from the Twitch website using our AWS EC2 instance and funneling the data into a POSTGRESQL database using Amazon RDS services.

We collected the stream level information for top 100 games and their top 100 streamers over the course of approximately two weeks. Surprise needs numeric ratings between 1-5 for every streamer and game they've played.

# **ER Diagram**

# 



# **Use Cases**

## Use case 1- Collaborative Filtering

## Use case 2- Customer Segmentation

## Use case 3- Propensity Rule Based Model

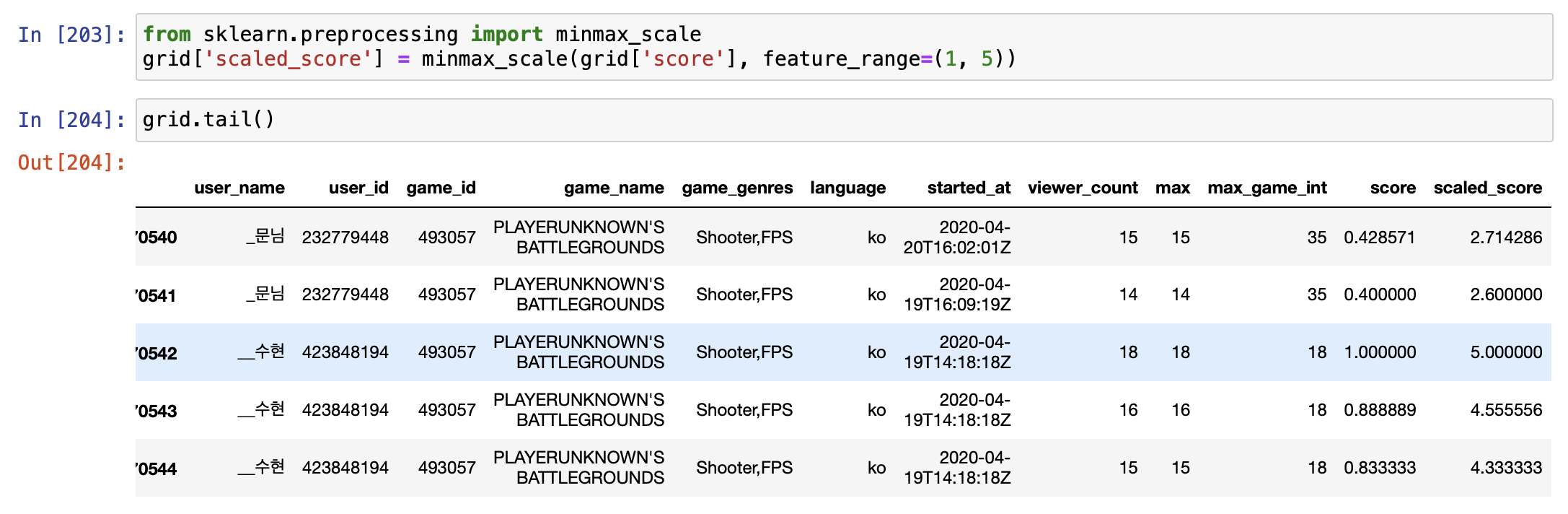
# 

# **Use case 1- Collaborative Filtering**

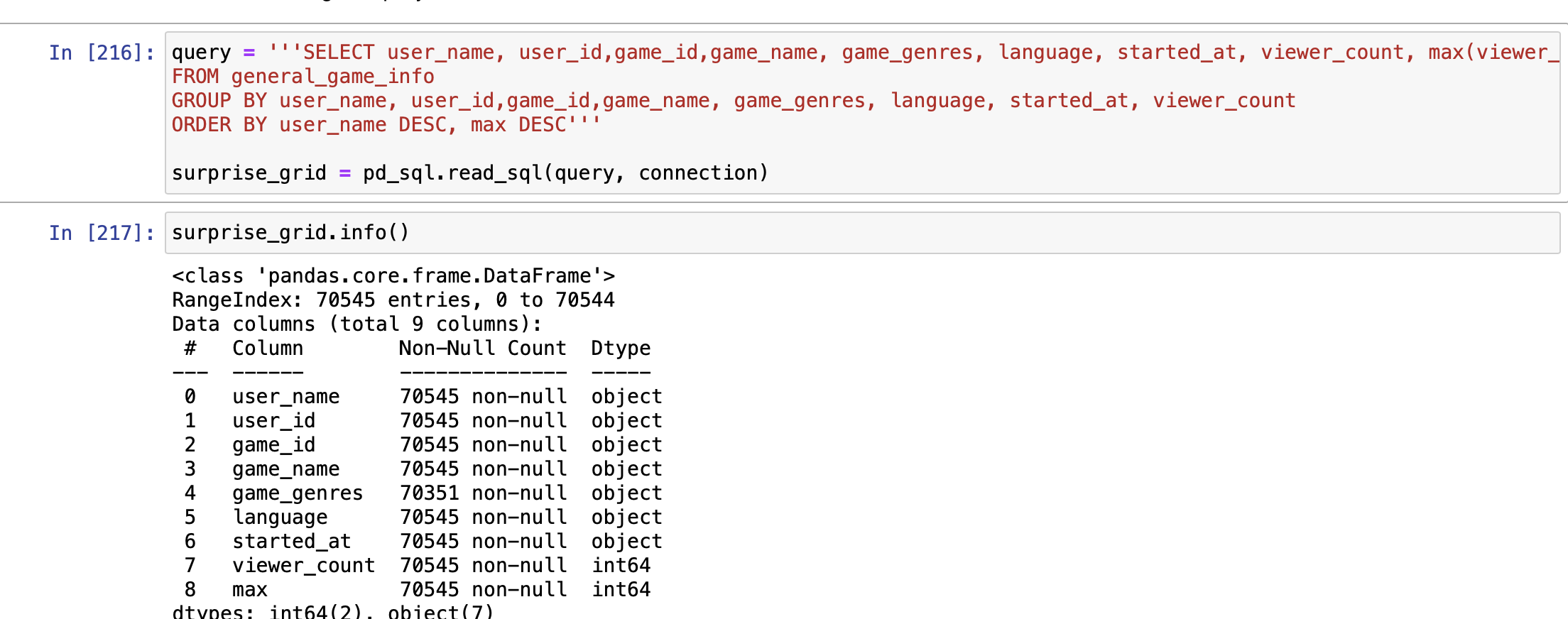
Our application aims to **recommend games/genres** based on streamer’s similar tastes/preferences liked in the past , like viewing history. For those streamers who already stream their favorite games we intend to recommend other similarly appealing games for them to try and stream.This will recommend the right shows for targeting the right people which will in turn **reduce churn rate**

### What are we doing to recommend?

Since we are focussed on streamers , we do not have the traditional “I like this movie, so I will rate it 4 on 5” ratings.  
How many viewers each streamer attracted with a particular game compared to their max viewer count over the period we pinged the API. For each user ,one game is their ultimate streaming “5 out of 5” benchmark, all the other games they play are compared to that one and normalized to ratings between 1 & 5.



The most granular level of recommendations would apply to those who are already a streamer on the platform, and are looking for advice on what to try perhaps as a change from their existing game or content. For these streamers, we would see if their user\_id is already contained in our database of top 100 streamers of top 100 games. If it is, we can use their existing streaming history to recommend a game that they have not yet streamed but which is trending and is similar to their existing stream history.



For genre data, SlopeOne seems to do the best algorithm. To drastically reduce overfitting, improve performance and ease implementation, the Slope One family of easily implemented Item-based Rating-Based collaborative filtering algorithms was proposed. For games data, we use the Baseline algorithm which predicts the baseline estimate for the given streamer and game.

### Why do marketing analysts need a recommendation system?

Of course, to help streamers/gamers to find the correct games to stream and to enjoy an individualized experience. On top of that, it is a great tool to retain the existing viewer base to make marketing more efficient.

# 

# **Use case 2- Customer Segmentation**

**Goal:** Gain deeper knowledge of our viewer and tailor targeted marketing campaigns and in turn improves user engagement and retention.

Inorder to analyze the entire viewer base we segmented them into homogeneous groups based on their usage of the streaming application.

The Key Metrics we used for segmenting our viewers are:

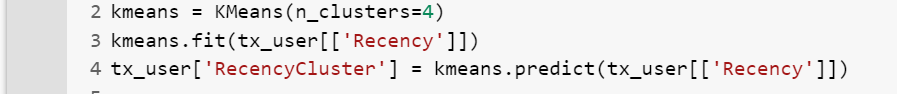
**Recency, Frequency, Engagement (RFE)**

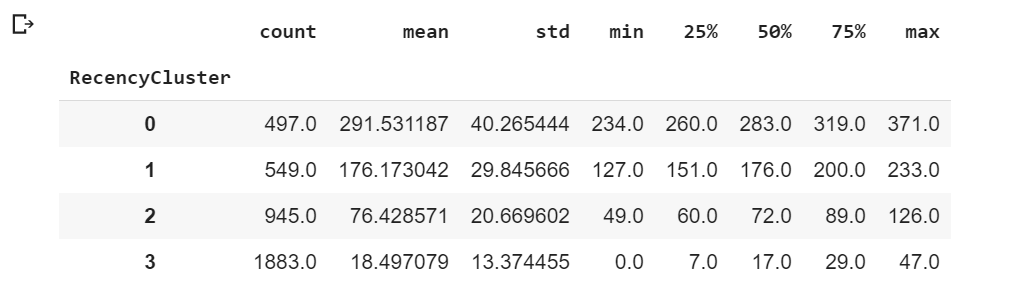
These metrics are important indicators of a customer’s behavior because frequency and engagement value affects a [customer’s lifetime value](https://clevertap.com/blog/customer-lifetime-value/), and recency affects retention.

**Methodology**: K- Means Clustering

Using the Elbow Method we classified our viewers into 4 clusters

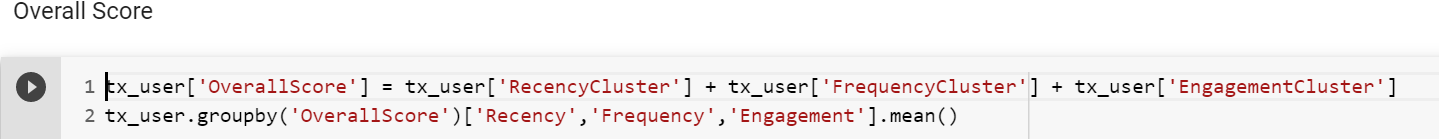
We used K- means Clustering Algorithm to Assign a Recency, Frequency and Engagement score





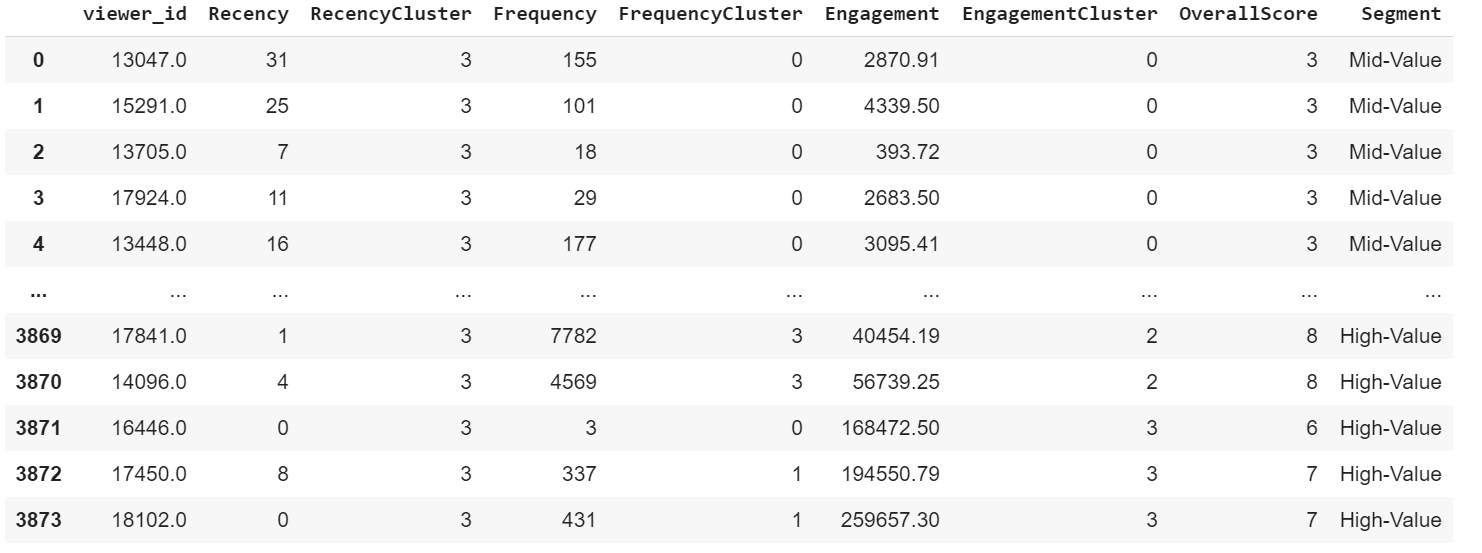
As shown in the above snippet, 3 covers the most recent customers and 0 has the most inactive customers.

Next we created an Overall Score by adding the individual RFE scores



To simplify things we name the scores as:

* 0 to 2: Low Value
* 3 to 4: Mid Value
* 5+: High Value



**Strategy recommendations based on the score** :

* High Value: Improve Retention
* Mid Value: Improve Retention + Increase Frequency
* Low Value: Increase Frequency

# **Use case 3- Propensity Rule Based Model**

Customer Churn helps a company evaluate how they are performing in their business. According to sources “on an average a company loses around **1.6 trillion through customer churn** and in order to acquire new customers , they need to spend 5 times more than to keep an existing one. Likewise, in Video Gaming Stream service too, it is important to hold streamers as they bring in revenue through paid partnership programs.

Hence our Propensity Rule based model, will help determine the # of streamers who have churned and not churned.

## Approach

**Step: 1** **Defining Business needs**

According to our company’s business requirement, we will decide on how they want to churn their historical data. There are two types

(i) Monthly Churn

This has a lead time of 31 days, (i.e) the model will calculate the # of churn based on streamers who have gone without a membership/subscription for more than a month

(ii) Bimonthly Churn

This has a lead time of 15 days, (i.e) the model will calculate the # of churn based on streamers who have gone without a membership/subscription for more than 2 weeks

**Step:2 Translating needs to problem parameters**

After the decision is made, the next step is to convert those problem statements into machine learning tasks. The main parameters we are using for finding our churn dates in our model are “user\_id”, “transaction\_date” and “membership\_expiry\_date”.

->By calculating the difference between one membership\_expire\_date and the next transaction\_date, the period of churn can be determined. If this value is greater than the days selected for a churn, then it is a positive churn.

->Our model then generates a set off cut off times using the prediction date parameter and then for each positive label, we generate the cut off time for churn.

For eg: If the churn occurs on 09-15 with a lead time of 1 month and a prediction window of 1 month, then this churn gets the cutoff time 08-01. Cutoff times where the customer was active 1-2 months out (for this problem) will receive a negative label, and, cutoff times where we cannot determine whether the customer was active or was a churn, will not be labeled.

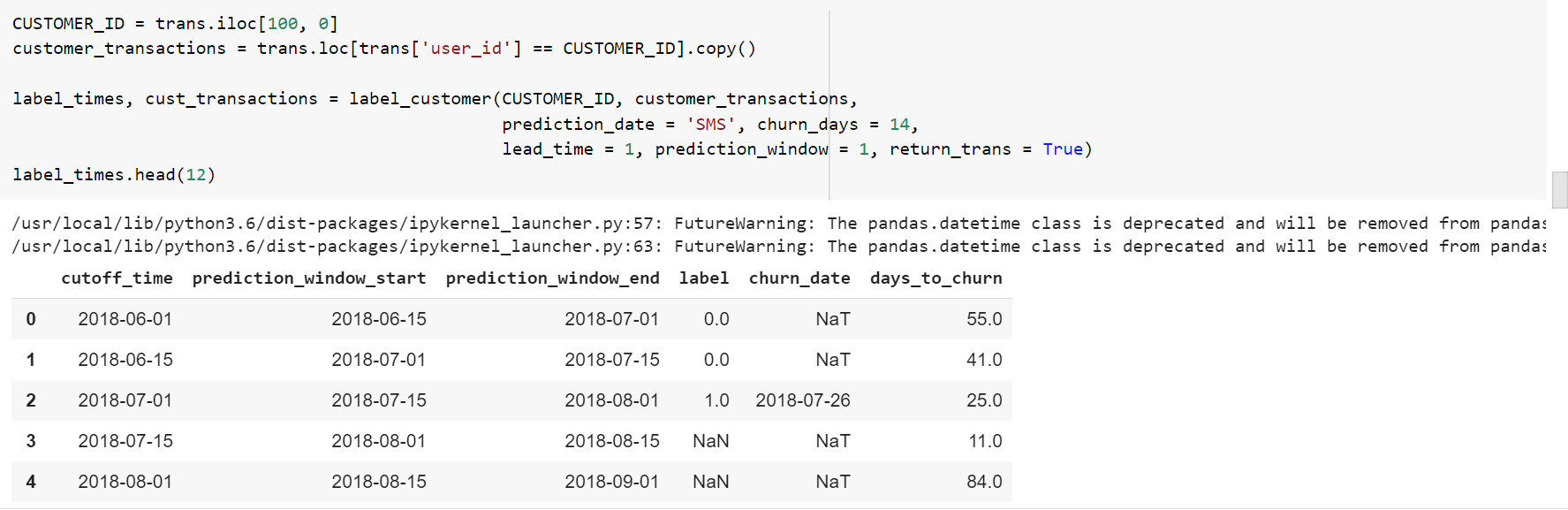
We can very rapidly label customer transactions by shifting each transaction\_date back by one and matching it to the previous membership\_expire\_date. We then find the difference in days between these two (transaction - expire) and if the difference is greater than the number of days established for churn, this is a positive label. Once we have these positive labels, associating them with a cutoff time is straightforward

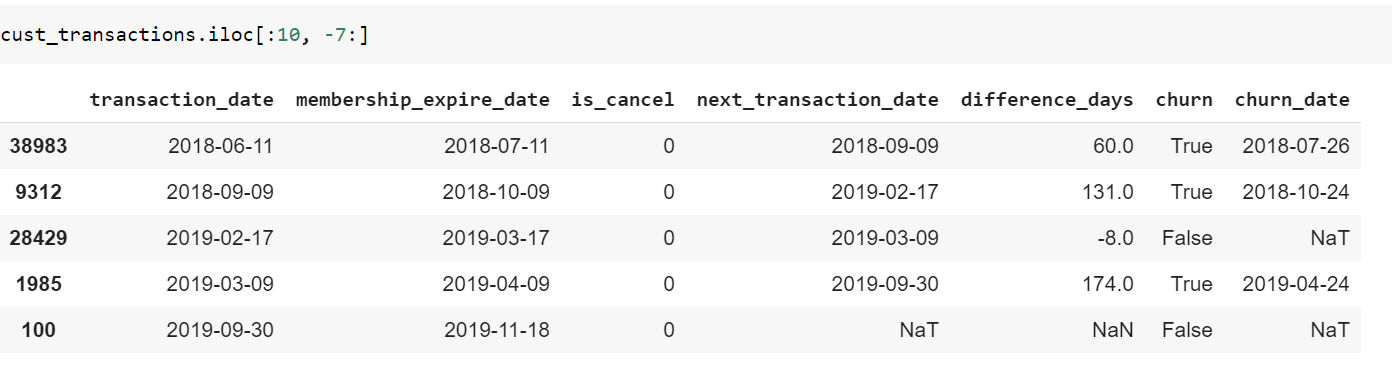
**Step:3 The general framework**

-> label\_customer(customer\_id, transactions, \*\*params)

-> make\_label\_times(transactions, \*\*params)

The first takes a single member and returns a table of cutoff times for the member along with the associated labels. The second goes through all of the customers and applies the customer\_to\_label\_times function to each one. The end outcome is a single table consisting of the label times for each customer. In the end, we have built two functions that will generate labels for customers based on churn or no churn.

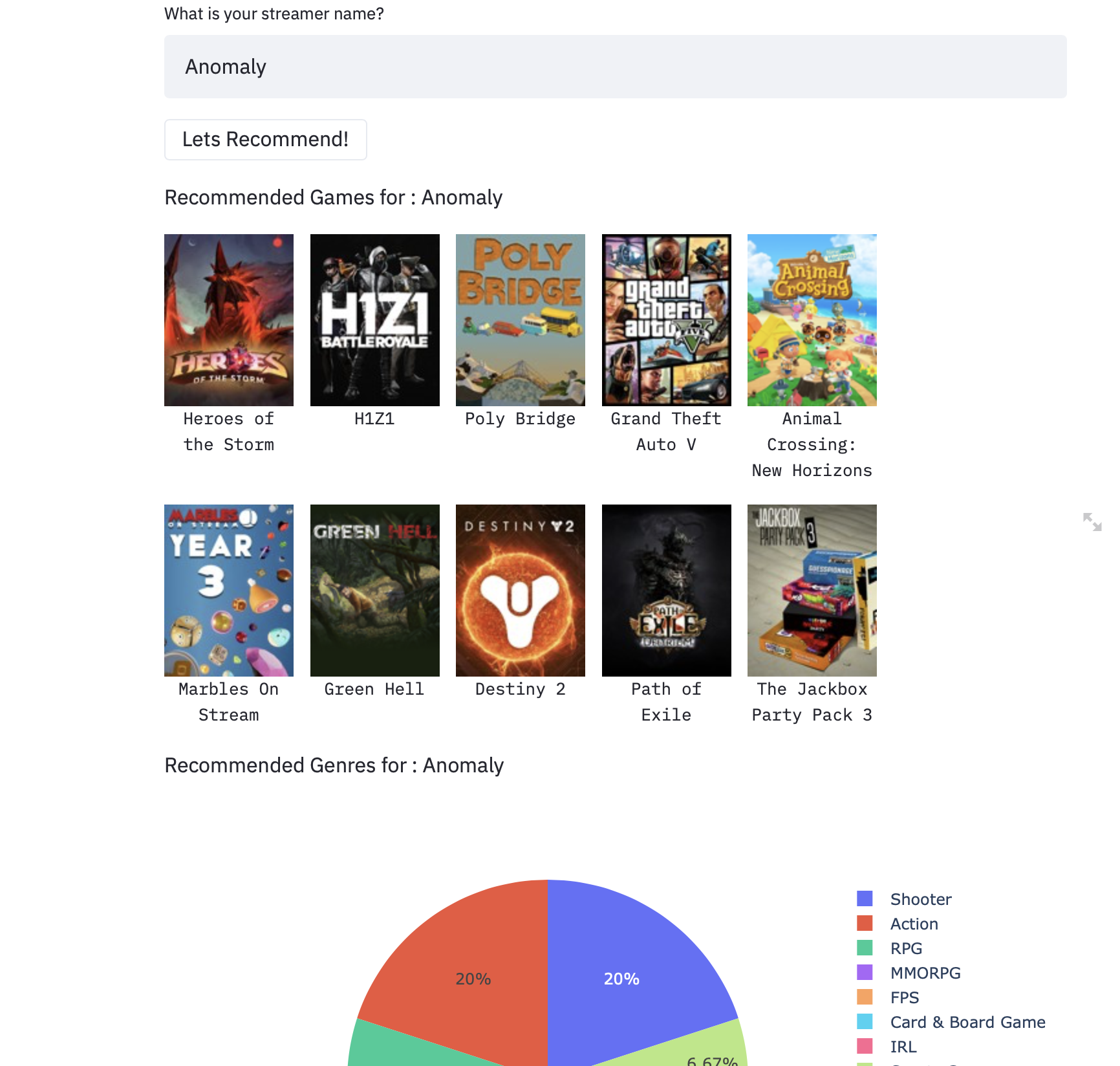
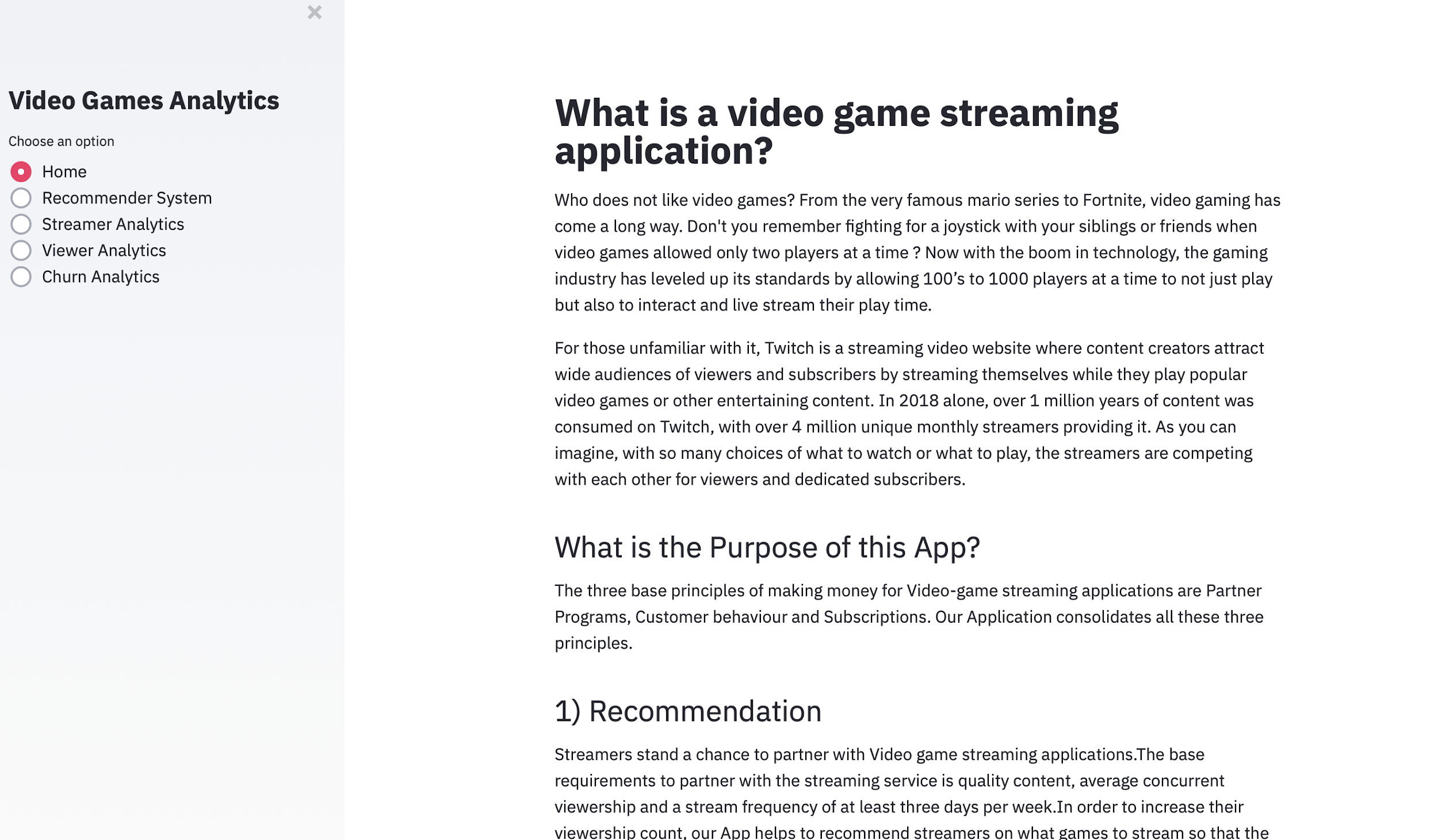


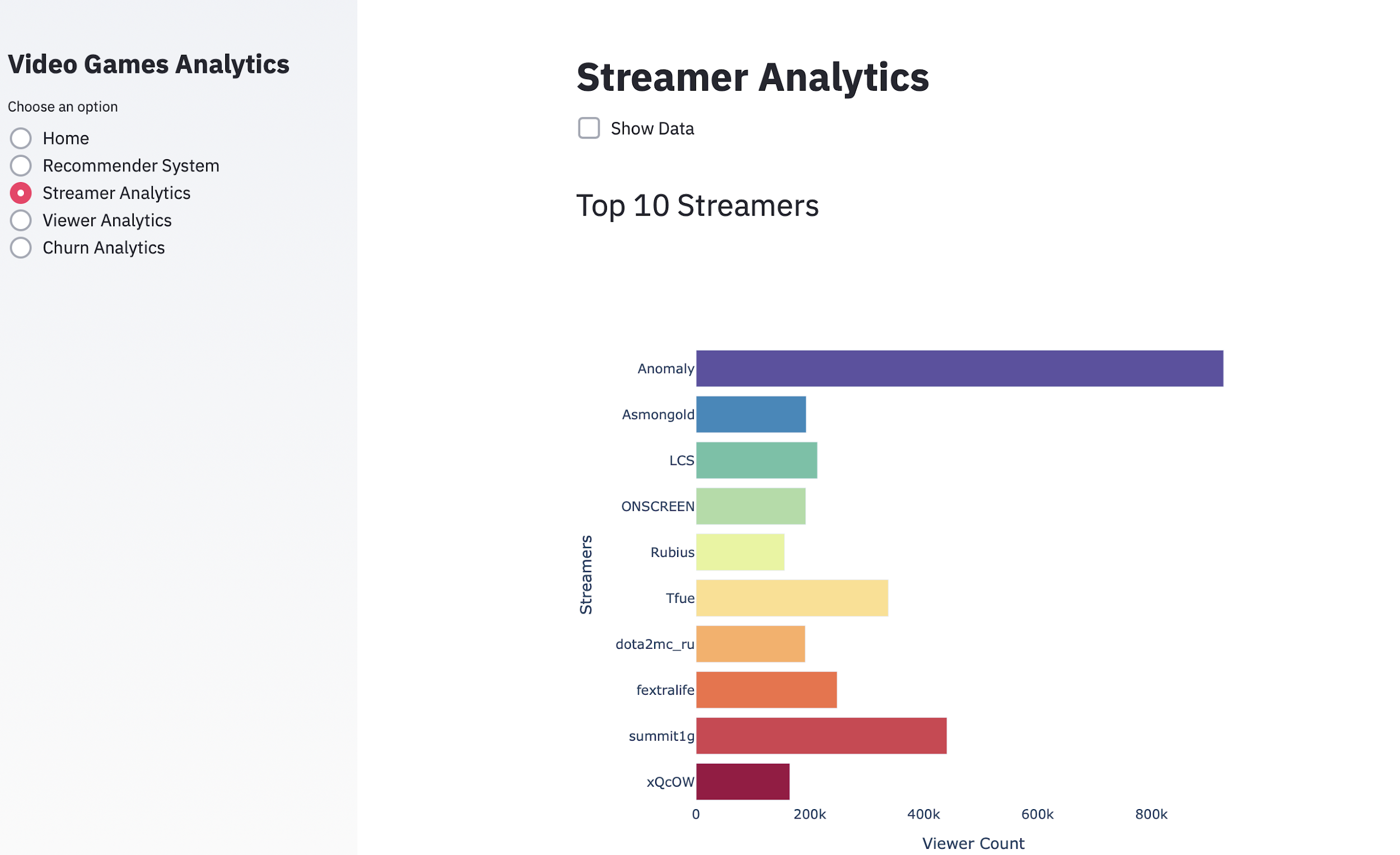


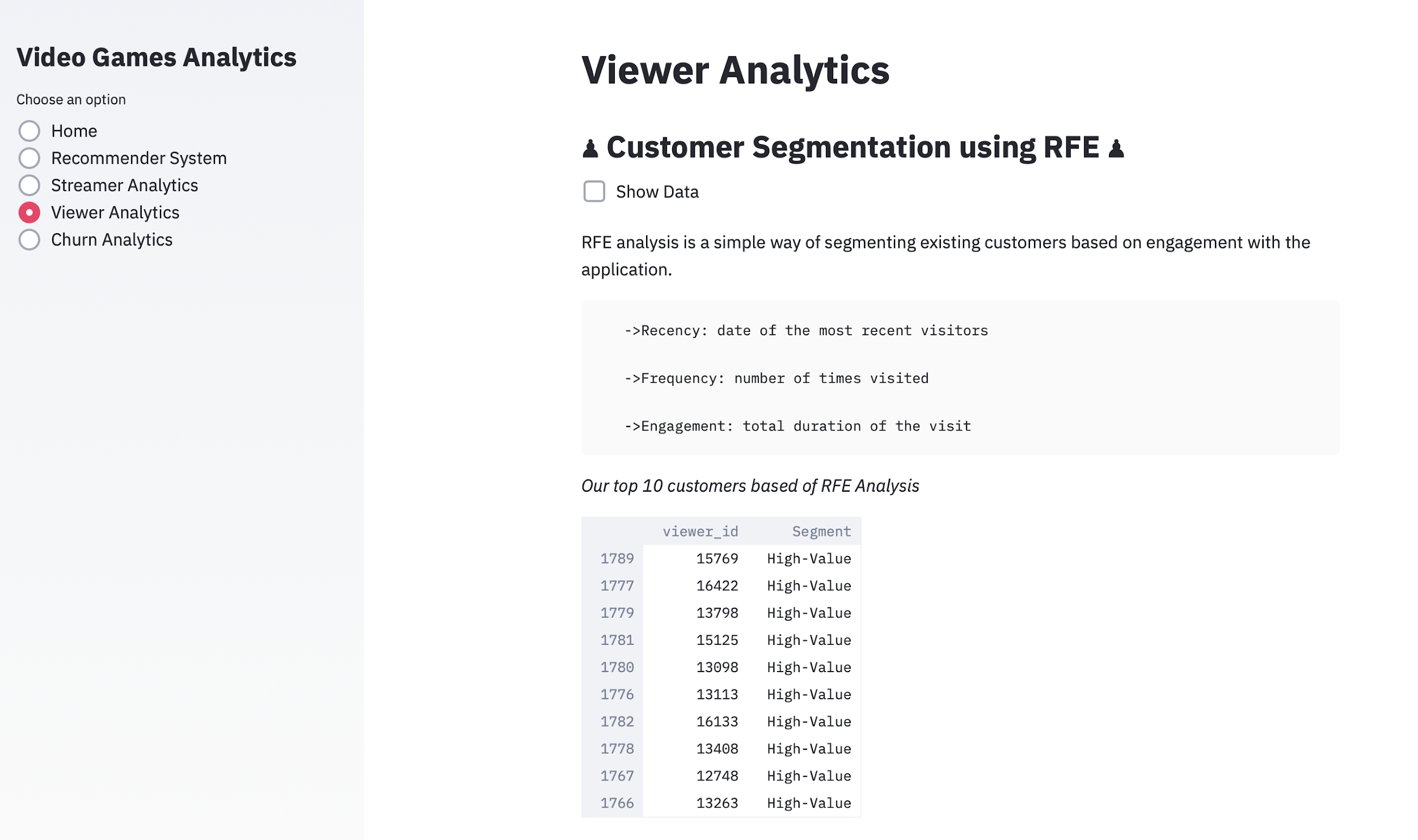
# Application & Github links

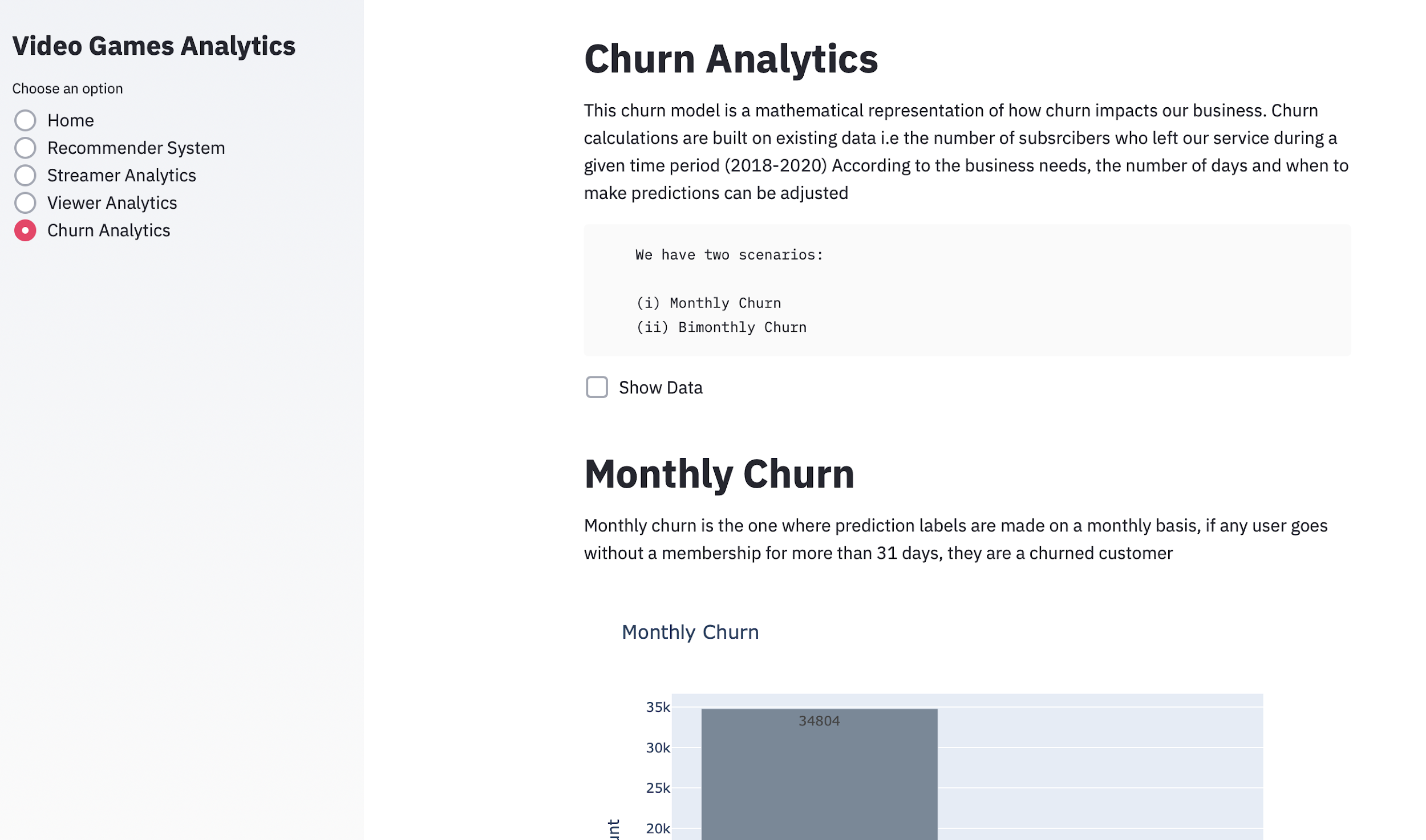
Application link:

[**https://videogames-analytics-app.herokuapp.com**](https://videogames-analytics-app.herokuapp.com)

****

****

****

****

Github link :

[**https://github.com/nikiravi04/INFO7374DigitalMarketingAnalytics/tree/master/Video\_Games\_Streaming\_Analytics**](https://github.com/nikiravi04/INFO7374DigitalMarketingAnalytics/tree/master/Video_Games_Streaming_Analytics)

# 

# Technology

1. Amazon RDS for cloud storage
2. PostgreSQL for the database integrated with Python
3. Jupyter Notebook with Pandas,Sci-kit learn,Seaborn,Plotly for exploratory data analysis and modelling
4. Streamlit to build the dashboard and display the predictive analytics for the marketing analyst
5. Deploy the application on Heroku