Lil' Report

1.1 Problem definition

- There is a huge amount of raw data on the internet that is generated every second which is not meaningful or comprehensible in any way. The objective of the project was to convert data into meaningful information by collecting data from sources like Twitch and Giantbomb, process it to detect interesting insights and observations about Twitch streaming service and game industry as a whole and visualize it. The major challenge was the process of data gathering and the absence of uniformity of the data. Along with that, blindly joining data would not give any value to the analysis, so understanding what kind of observations will be relevant and gripping was important.

1.2 Target and Challenges

1) Popularity of a game/category/stream/channel:

Challenges come from how we are going to measure 'popularity'. There are multiple perspectives out there and we need to make each of them looks convincing and meaningful. This means that the results we use must be reasonable and accurate so that they make sense. Such as number of streams vs. number of viewers regarding the popularity of a game. Which one makes more sense on conveying the concept?

And remember, we are dealing with 50 gigabytes' data. The challenge could arise from the very beginning of the process: how we are going to ETL the data so that we could fetch the necessary information to convey a sense of popularity of some object.

Furthermore, the data we collected could lack the required information we need to lead to a result. Actually, twitch does not provide category data for games. We may find more of this kind of issue as we are making progress

2) Popularity in time frame scale:

The challenge is simple: how do we define 'time frame'? We can arbitrarily make some definition, but the result could seems ridiculous based on our 'guess'. When is morning, noon, afternoon and night? What about time zones and do we really need to consider about that?

3) Make simple prediction about popularity of a streamer in the near future:

'Prediction' could be the most tricky part in our project. First of all, we actually lack useful data to conduct predictions. Twitch does not have any information about users: age, nationality, education level, and so on. Furthermore, how could we make prediction for a future object, when we actually do not know the value of a feature for that object in the future? For example, we want to use number of followers of channel as a part of feature vector, but number of followers is a variable that could change along the time. To make a prediction, we need the number of follower for that channel in the future. And that is impossible to know.

The limitation of useful information we collect from twitch becomes a challenge for us to dig deep into the data. If we want to conduct reasonable investigation on the data in depth, we have to make full use of existing information.

2. Methodology

- ETL:

  ETL work took a lot of effort because the data we download using twitch api and giantbomb api was dirty. First made advantage of python regular expression operation package to clean the collected json files, such as removing the tail comma, so that they could be read by pyspark. Afterwards, mainly use pyspark to convert json strings into featurized dataframes.

  Dataframe is suitable with our project, because it provides well-structured table frame, to which data with multiple features well fits. Furthermore, Dataframe is managed by using sql like functions. It also provides users with API that enables hardcoded sql queries. This sql-user-friendly feature makes it easy to handle complex data.

  The original data was splitted into three tables: stream table, which contains information about the stream, including stream id, game, created time of the stream and or so; channel table, which was separated from stream data, that describes the channel owner who was broadcasting corresponding stream; game table, which was a joined table from stream data and game details from giantbomb. The reason to separate data in this way, is that it matches the three main topics we want to conduct on the data, namely stream, channel and game.

- EDA:

  The exploratory data analysis, as explained on wikipedia, is usually a method used to describe the hidden characteristics of the data helping the formulation of the hypothesis regarding various aspects of the data through mainly visualization methods.

  First of all, we selected a group of results that are more appealing for visualization and teammate responsible building the website puts all of the selected results to our website and used

  As mentioned earlier, we used different standards for measuring popularities of games, channel, language and so on. The reason behind this is that we were trying to get a sense of for example, popularity, from each possible point of view and hence

getting a full depiction of the popularity on each subjects (eg.games, channels). However, the results on certain topics from each perspective do not vary much and sometimes stays the same. For example, fortinet is the most popular game played on twitch by both view count and number of streamer played. As you can see from one of our graph used on the website, fortnite, red hot as it is, dwarfs all other games, classic or newly released. Therefore, we only selected a few results from data analysis to represent some of the topics for our website.

Among all of the analysis of twitch data, not all of the results are suitable for making simple and concise graphs either because it contains so few numbers or because it is a bit too complicated.

For example is the one of the results for analyzing the what channels are popular and what languages does each streamer speaks, the resulting table looks like:

channel_id	name	broadcaster_language	max(channel_total_views)	total_followers	partner	mature
46108674	wildlands	en	1181979	62914	false	true
173162545	waveigl	pt	1591695	56507	false	true
98260357	drjarba	en	2254060	56030	false	true
272702367	voiceoverpete	en	77741	52841	false	false
50815446	saltyteemo	en	2897849	52596	false	false
36957524	glebleodota	ru	4671197	34229	false	false
92023003	ninjaxluke	en	237614	31006	false	false
152779293	the_nightmare	tr	287127	30612	false	true
98338285	louisejulie_	en	238221	30350	false	true
58683351	bigjoe_4u	ar	296580	29338	false	true
82383962	scottp86	en	124735	29053	false	true
91196150	triplegzgaming	en	315303	27930	false	false
89748886	xx_babygirl_xx	de	154489	26906	false	true
82885897	rajtantajta	en	441720	26800	false	true
76548748	soprecious	tr	438997	24761	false	true
37970439	sidekool	pt	455284	23793	false	true
134070181	instantgamingfr	fr	1546975	23197	false	false
176618772	mamadotv	pt	583582	23168	false	false
158936012	tiggra	ru	740818	21395	false	true
46310178	honmastertv	ru	429516	21281	false	true

Figure 2. sample channel and language analysis result

- MLLib:

  Since we had plenty of data collected, we could make predictions about the popularity of games and categories using Apache Spark Machine Learning Library. We have decided to make prediction on how many viewers will Top-20 games will have and which categories will be most popular on January 1, 2019. We identified it as a regression problem, since it requires the prediction of a continuous value.

  At first, we selected stream data that can be used as features and split it into training and validation sets, however, the model cannot be trained on textual data. To resolve

that we used SQLTransformer to transform the dates using UNIX_TIMESTAMP() to get number value of the date and hash the name of the games and categories. The features to train the model are: date in UNIX format, hashed name of the game or category and day of the week (since it also affects how many people are watching streams), to combine the features VectorAssembler was used. We ran several attempts to train a model using Linear Regressor, General Linear Regressor, but DecisionTreeRegressor has shown best predictions and that's what we kept in our final implementation. After the pipeline model was trained on the whole dataset, we attempted to make predictions on the popularity of the games and categories. The results are provided on our website.

3. Problems

ETL:

- After cleaning original data, we found out that it does not readable column names for features. So we make customized schema to solve the issue
- The original data was messed up. One significant problem was that twitch allows users to manually type in the game name they are broadcasting. This leads to severe problems such as typo, mixed lower and upper cases, or just random names. At very first we tried to use Levenshtein distance algorithm to deal with typo, but at last it was proved to be useless because the algorithm cannot tell whether the name had some typo or merely was another game. At the end, we have to roughly filter out 'irregular' names. That is, names of games which had few viewers. And we convert all names to lower cases in order to make a better join with data from giantbomb.
- In order to analyze data on time frame scale, we need a new column representing what period of day the stream was being broadcast. So we discussed and decided to break a day into six different time pieces.
- Original time data contains minitues, needed to be converted into simple yyyy-mm-dd format.

Data Analyze:

- Cannot simply count the number of streams, because we collect data every 30 minutes, and the same stream could appear multiples times in our data. Before counting, have to select distinct stream id
- Cannot simply sum up the number of viewers for a game. The reason is similar to that of number of streams, while this time, number of viewers for one stream could vary. We choose to compute the max or average value for number of viewers for each stream at first, and sum them up to the total number of viewers for each game or category or time frame afterwards

- Number of games exceeds number of categories that could be handled by StringIndexer. Instead, we hash code the game name

4. Results

Brief explaination of the charts

Game popularity:

Table shows the top 20 most popular games played on twitch with corresponding total number of streams(videos) in the duration from 2018-11-13 to 2018-11-26. The bar chart on the right shows that Fortnite is by far the most popular game played on twitch (about 4.6 times more than the 2nd game).

Category popularity

All of the web pages for this topic follows the same layout style of table on the left and chart(s) on the right.

Each game can have more than 1 categories (tags). For example, a game like Fortnite falls in both shooter and MMORPG categories.

1) **By stream count**

   The table listed top 20 most popular game genre/category by total stream count for each genre. Top 3 categories are Shooter, First-Person-Shooter and Action, they consumed more than half of the total streams played on twitch.

   In other words, there is 1 in every 2 streamers on twitch streamed themselves playing games in these 3 categories.

   The 'half-pie' chart provides a more focused view on rest of the categories by minimize the visual effects of the most popular category.

2) **By viewers**

   Same idea but using total view count as basis of measuring category popularity.

   Top 3 categories are the same. They still takes half of the pie chart means that half of the people who watched videos on twitch are watching games in these 3 categories.

3) **By followers**

   Using total followers count, the top 3 categories are the same, but only takes one third of the entire fan base. The fan base stood strong for other less popular categories.

4) **By languages**

   This time, bar charts shows the total number of viewers for each languages used on twitch. People watch streams in english predominantly since most of the streams are in english. The table on the left shows the most popular categories for each popular languages by viewers count.

Channel popularity

Table on the left shows the top 20 most popular channels on twitch by total follower counts with additional information of if the streamer's channel is labelled 'mature' and if this channel is a officially registered partner of twitch.

Ninja, the most popular streamer, has the largest fan base, counting almost 25% of the entire fan population.

Time frame popularity

All time used PST, first graph shows that the number of streams and number of viewers has a nice little phase difference which represents the accumulation of the audiences while the stream is playing.

Another interesting observation is that the total number of streams and audiences drops a little in late-night since only the true-hard-core streamers and viewers will stay up super late for the games they love.

The charts below are showing the distributions of total number of streams and viewers for each time frame throughout the duration of the period of data collection.

The diagram at the bottom shows the distribution of total viewers count through each days of the week. Tuesday seems to be the ramadan day for most of the gamers and streamers.

Language popularity

The line chart shows the most popular languages used for broadcasting on twitch by total number of channels.

As discussed earlier, majority of streamers on twitch use English but German, French, and Spanish twitch streamer also formed their own gaming community on twitch.

Predictions

The top table and bar chart illustrate the predicted number of viewers for the most popular games on January 1, 2019. As expected, Fortnite is topping the rating, followed by Call of Duty and Overwatch.

The bottom chart demonstrate top 20 categories that will be popular on January 1, 2019. As seen in the results, Shooter, First-Person Shooter and Action games will continue to be among the top categories in the future.

5. **Project Summary**

- Getting the data: Acquiring/gathering/downloading: 3.0
    - We collect data from twitch and giantbomb, in total 50 gigabytes
- ETL: Extract-Transform-Load work and cleaning the data set: 3.0
    - The data was noisy and huge so that ETL took much effort
- Problem: Work on defining problem itself and motivation for the analysis: 2.0
    - We held couple of meetings to discuss our target and topics
- Algorithmic work: Work on the algorithms needed to work with the data, including integrating data mining and machine learning techniques: 2.5
    - We mainly used agg functions to calculate our results. We used Spark ML toolkits to predict future viewers
- Bigness/parallelization: Efficiency of the analysis on a cluster, and scalability to larger data sets: 2.0
    - We conducted analysis work on 50 gigabytes' data. We used repartition, cache and broadcast join to ensure the efficiency of computation
- UI: User interface to the results, possibly including web or data exploration frontends: 3.5
    - We used react.js to develop our front-end and spent a decent amount of time implementing it. Check readme for more detailed tech stack about our frontend.
- Visualization: Visualization of analysis results: 3.5
    - Though one of our teammate has some prior experience in front-end development, none of us has experience visualizing data analysis before. We also spent time reformatting our json output in order to fit those framework.
- Technologies: New technologies learned as part of doing the project: 0.5
    - We use open-sourced frameworks like Charts.js and ReChart.js