codecademy

VISUALIZATION CUMULATIVE PROJECTS

Twitch

Visualizing Data With Matplotlib

Alex Ricciardi

Date: 06/20/2020

Project Overview

Twitch is the world's leading video platform and community where millions of people and thousands of interests collide in a beautiful explosion of video games, pop culture, and conversation. Its live and on-demand video platform forms the backbone of a distribution network for video game broadcasters including professional players, tournaments, leagues, developers and gaming media organizations.
In this project, we have partnered with Twitch's Science Team and you will be working with two related tables that describe user engagement with Twitch video and chat. The project is broken down into two parts:
•Part-1: Analyze Data with SQL
•Part-2: Visualize Data with Matplotlib

The following presentation is for part -2 of the twitch project, for part-1 please see README_SQLite.text file.

Project part-2 Overview

In this part of the project, you will be taking your findings from the SQL queries and visualize them using Python and Matplotlib, in the forms of:

- Bar Graph: Featured Games
- Pie Chart: Stream Viewers' Locations
- Line Graph: Time Series Analysis

Note: With Twitch Part-1 , Analyze Data with SQL, two data sets were provided:

- video_play.csv, the stream table
- chat.csv, the chat table

Project Tasks Overview

Bar Graph: Featured Games:

1. Use SQL to find the top 10 trending games (on January 1st, 2015) and their number of viewers.
2. Use the plt.bar() to plot a bar graph.
3. Make the graph more informative.

Pie Chart: League of Legends (LoL) Viewers' Whereabouts:

1. Using SQL, create a list of countries and their number of LoL viewers.
2. With the lists labels and countries create a pie chart.
3. Make the pie chart more visually appealing and more informative.

Line Graph: Time Series Analysis:

1. Use SQL to find the number of US viewers at different hours of the day on January 1st, 2015.
2. Whist the lists hour and viewers_hour plot a line graph.
3. Let’s account for a 15% error in the viewers_hour data.

1. Bar Graph: Featured Games

Task-1
Use SQL to find the top 10 trending games

1.1.1 overview

Twitch's home page has a Featured Games section where it lists the "Games people are watching now".

In the previous part of the project, we used SQL to find the top 10 trending games (on January 1st, 2015) and their number of viewers.

In the next few tasks, we take this data and plot a bar graph using Matplotlib.

1.1.2 Top 10 trending games

Output

game	Number of Views
League of Legends	1070
Dota 2	472
Counter-Strike: Global Offensive	302
DayZ	239
Heroes of the Storm	210
The Binding of Isaac: Rebirth	171
Gaming Talk Shows	170
Hearthstone: Heroes of Warcraft	90
World of Tanks	86
Agar.io	71

Code

```
SELECT game, COUNT(*) AS 'Number of Views' FROM stream
GROUP BY game
ORDER BY COUNT(*) DESC
LIMIT 10;
```

Task-2
Use the plt.bar() to plot a bar graph

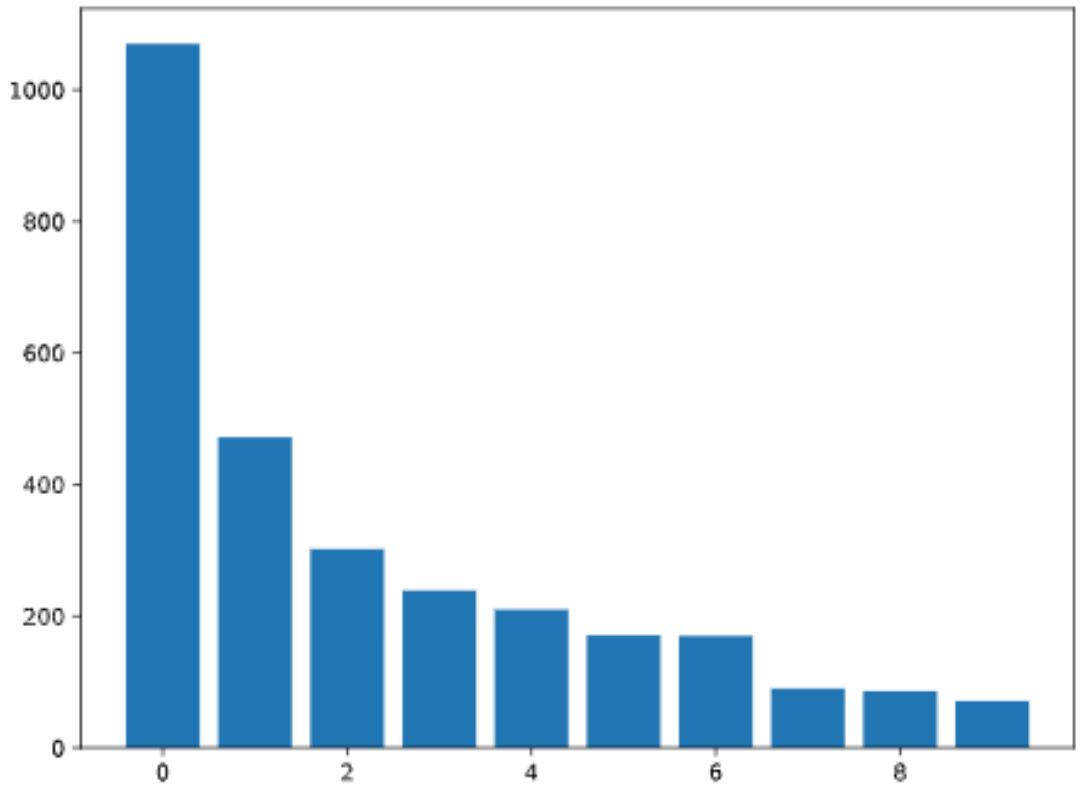
1.2.1 overview

To reflect the SQL query use the lists:
games = ["LoL", "Dota 2", "CS:GO", "DayZ", "HOS", "Isaac", "Shows", "Hearth", "WoT", "Agar.io"]
viewers = [1070, 472, 302, 239, 210, 171, 170, 90, 86, 71]

Now, use the plt.bar() to plot a bar graph using range(len(games)) and viewers as arguments.

Then, use plt.show() to visualize it.

1.2.2 Top 10 trending games bar graph



Code

```
# Config figure size, Personal preference, n
ot a project requirement
plt.figure(figsize=(8,6))
# 2. Use the plt.bar() to plot a bar graph
plt.bar(range(len(games)), viewers)
# ouput
plt.show()
# Clear the current figure
plt.clf()
```

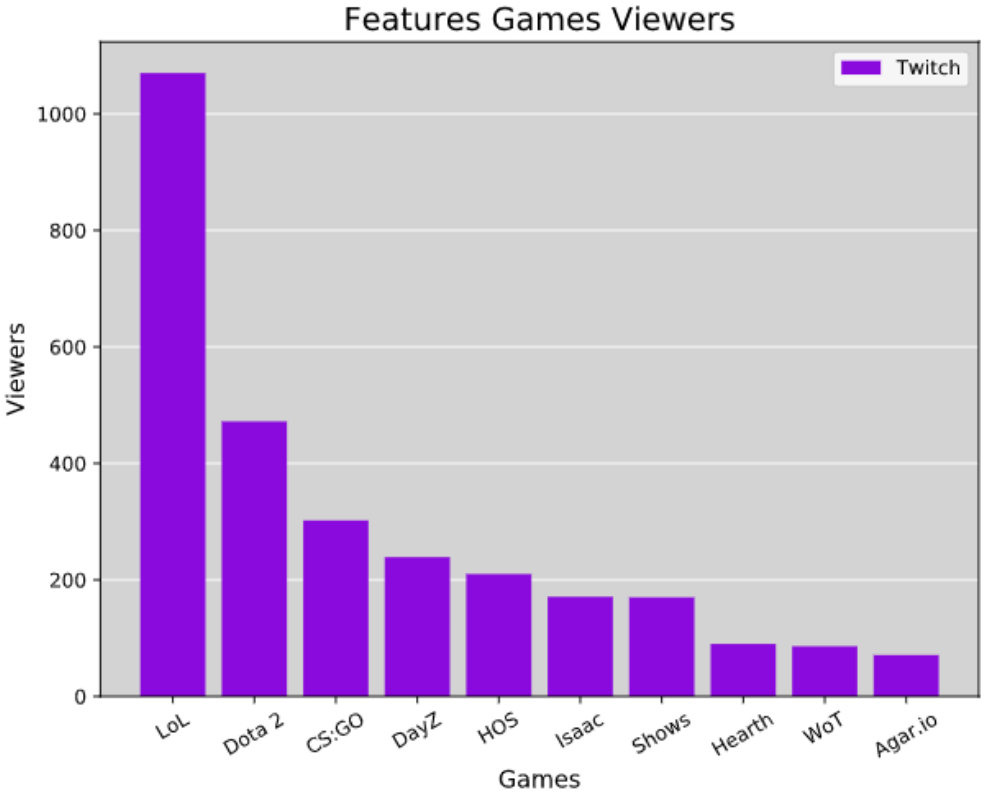
Task-3
Make the graph more informative

1.3.1 overview

Make the graph more informative by doing the following:

- Add a title
- Add a legend
- Add axis labels
- Add ticks
- Add tick labels (rotate if necessary)

1.3.2 Top 10 trending games more informative bar graph



Code

```
plt.bar(range(len(games)), viewers, color='#8A2BE2')
# Graph title
plt.title('Features Games Viewers', fontsize=16)
# Legend
plt.legend(["Twitch"])
# axis Label
plt.xlabel('Games', fontsize=12)
plt.ylabel('Viewers', fontsize=12)
# axis object
ax = plt.subplot()
# x ticks axis object
ax.set_xticks(range(len(games)))
ax.set_xticklabels(games, rotation=30)
# Graph background color
ax.set_facecolor('lightgray')
# Draws grid lines behind other graph elements
ax.set_axisbelow(True)
# Draw a grid
plt.grid(axis='y', color='w', linestyle='solid')
# ouput
plt.show()
# Close figure
plt.close()
```

2. Pie Chart: League of Legends Viewers' Whereabouts

Task-1
Using SQL,
query the number of LoL viewers per country

2.1.1 overview

There are 1070 League of Legends viewers from this dataset.
Where are they coming from?

As well as other countries that accounted for another 279 stream viewers.

In the next few tasks, we take this data and make a pie chart.

2.1.2 Number of LoL viewers per country

Output

country	Number of LoL Views
US	447
DE	66
CA	64
∅	49
GB	45
TR	28
BR	25
DK	20
PL	19
BE	17
NL	17

Code

```
SELECT country, COUNT(*) AS 'Number of LoL Views'
FROM stream
WHERE game = 'League of Legends'
GROUP BY 1
ORDER BY 2 DESC
Limit 11;
```

Task-2
Using the LoL viewers per country plot a pie chart.

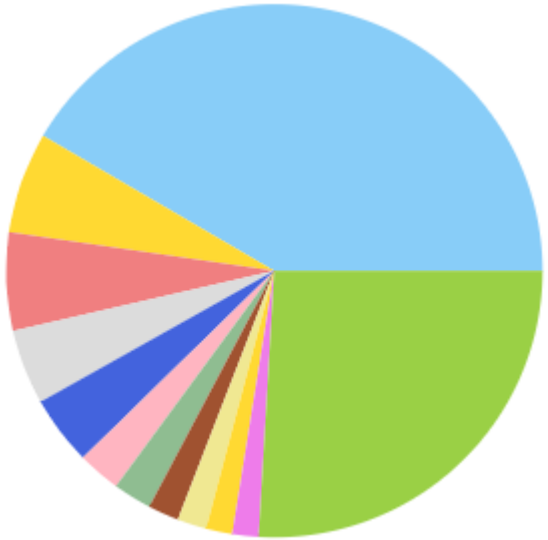
2.2.1 overview

To reflect the SQL query use the lists:
labels = ["US", "DE", "CA", "N/A", "GB", "TR", "BR", "DK", "PL", "BE", "NL", "Others"]
countries = [447, 66, 64, 49, 45, 28, 25, 20, 19, 17, 17, 279]

Then, use plt.pie() to plot a pie chart.

Lastly, use plt.show() to visualize it.

2.2.2 LoL viewers per country pie chart



Code

```
# Config figure size, Personal preference,
# not a project requirement
plt.figure(figsize=(5,5))
# Color countries list
colors = ['lightskyblue', 'gold', 'lightcoral', 'gainsboro', 'royalblue',
'lightpink', 'darkseagreen', 'sienna', 'khaki', 'gold', 'violet', 'yellowg
reen']
# Makes pie plot
plt.pie(countries, colors=colors)
# ouput
plt.show()
# Clear the current figure
plt.clf()
```

Task-3
Make the chart more informative

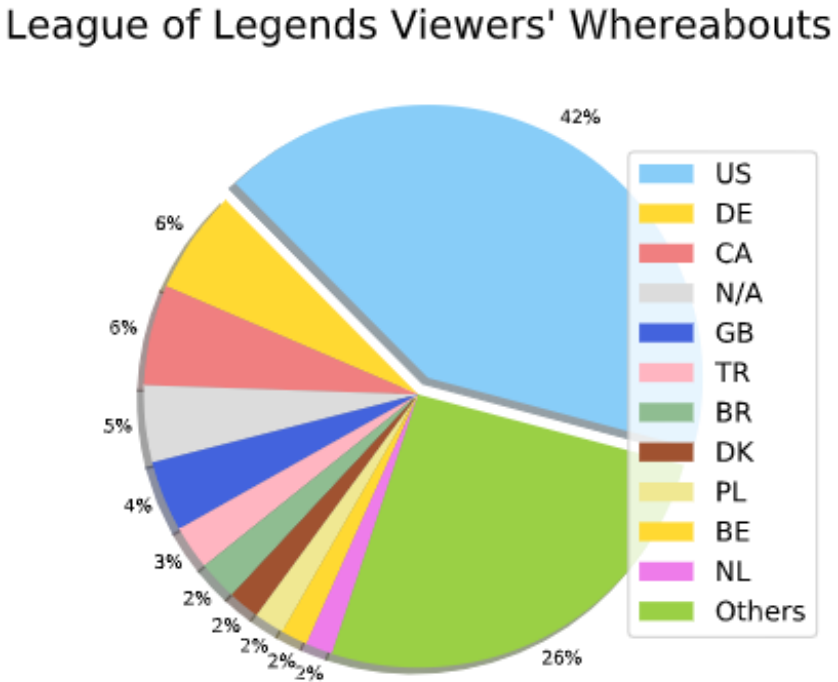
2.3.1 overview

Make the pie chart more visually appealing and more informative.

Use “explode”, or break out, the 1st slice (United States):

- Add the explode
- Add shadows
- Turn the pie 345 degrees
- Add percentages
- Configure the percentages’ placement

2.3.2 LoL viewers per country more informative pie chart



Code

```
# Explode list
explode = (0.07, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
# Makes visually appealing pie plot
plt.pie(countries, explode=explode, colors=colors,
        shadow=True, startangle=345, autopct='%1.0f%%',
        pctdistance=1.1, textprops={'fontsize': 7.4})
# Title
plt.title("League of Legends Viewers' Whereabouts",
          fontsize=16)
# Legend
plt.legend(labels, bbox_to_anchor=(1.1, 0.5),
           prop={'size': 11}, loc="right")
# ouput
plt.show()
# Close figure
plt.close()
```

3. Line Graph:
Time Series Analysis

Task-1
Use SQL to find the top 10 trending games

3.1.1 overview

In the previous part of the project, we used SQL and we were able to find the number of US viewers at different hours of the day on January 1st, 2015.

In the next few tasks, we take this data and plot a line graph using Matplotlib

3.1.2 Number of US viewers per hour

Code

```
SELECT strftime('%H', time) AS 'Hour',
       COUNT(*) AS 'Number of Viewers'
FROM stream
WHERE country = 'US'
GROUP BY 1;
```

Output

Hour	Number of Viewers
00	30
01	17
02	34
03	29
04	19
05	14
06	3
07	2
08	4
09	9
10	5
11	48
12	62
13	58
14	40
15	51
16	69
17	55
18	76
19	81
20	102
21	120
22	71
23	63

Task-3
Using the number US viewers per hour plot a line graph.

3.2.1 overview

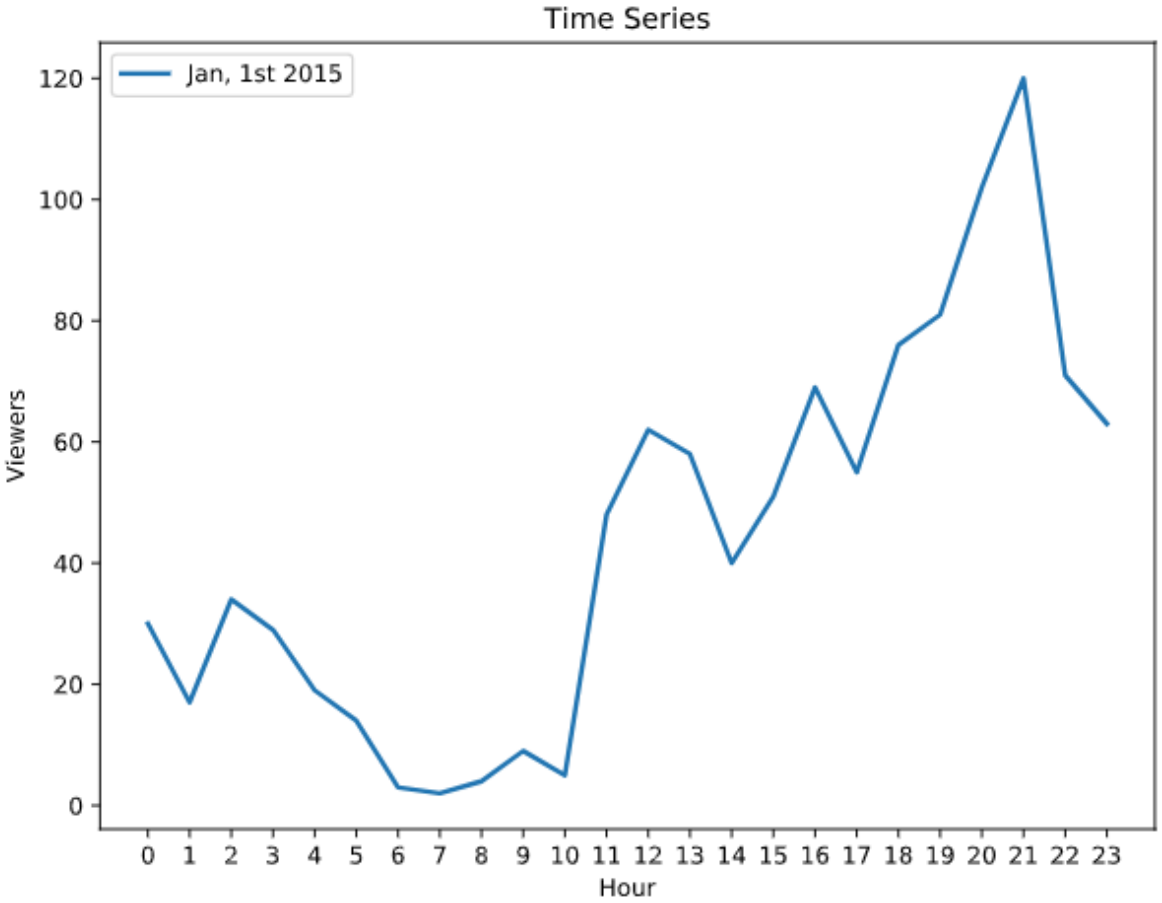
To reflect the SQL query use the lists:
hour = range(24)
viewers_hour = [30, 17, 34, 29, 19, 14, 3, 2, 4, 9, 5, 48, 62, 58, 40, 51, 69, 55, 76, 81, 102, 120, 71, 63]

Use plt.plot() to plot a line graph.

Then, add the title, the axis labels, legend, and ticks.

Lastly, use plt.show() to visualize.

3.2.2 Us viewers per hour line graph



Code

```
# Config figure size, Personal preference,
# not a project requirement
plt.figure(figsize=(8,6))
# Title
plt.title("Time Series")
# axis labels
plt.xlabel("Hour")
plt.ylabel("Viewers")
# Makes line plot
plt.plot(hour, viewers_hour, linewidth=2)
# Legend
plt.legend(['Jan, 1st 2015'])
# Adding x Ticks
ax = plt.subplot()
ax.set_xticks(hour)
# ouput
plt.show()
# Clear the current figure
plt.clf()
```

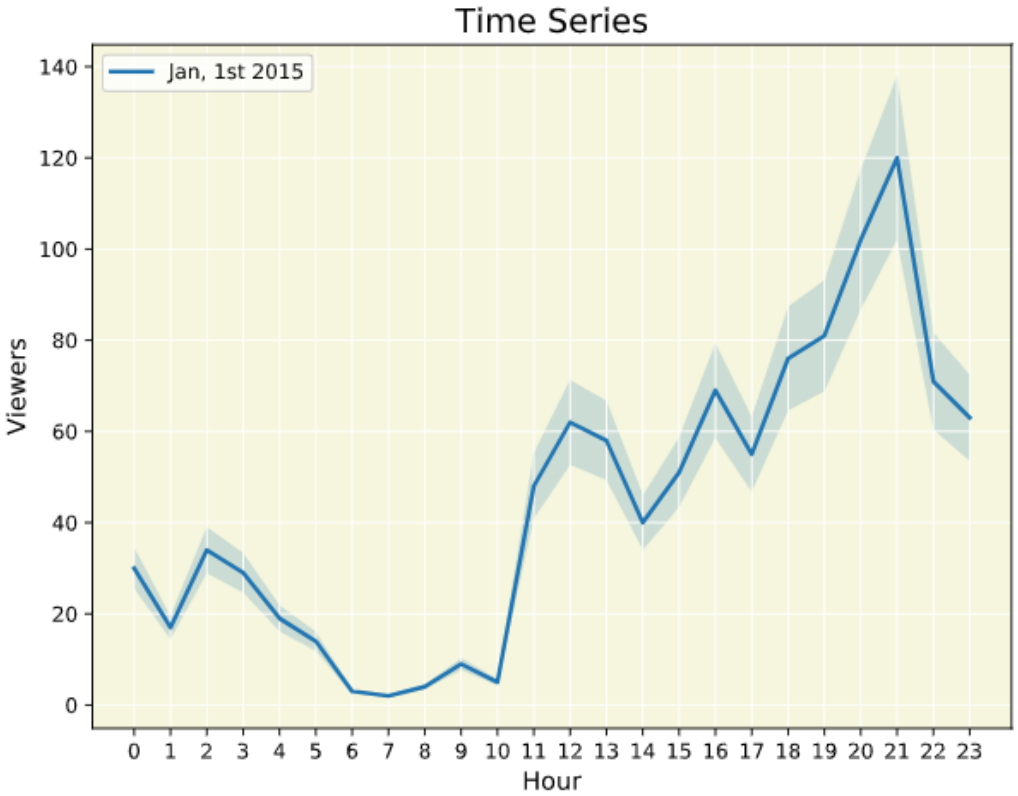
Task-3
15% error in the viewers_hour data

3.3.1 overview

There is some uncertainty in these numbers because some people leave their browsers open. Let’s account for a 15% error in the viewers_hour data.

Use plt.fill_between() to shade the error, with an alpha of 0.2.

3.3.2 Us viewers per hour line graph accounting for 15% error



Code

```
# Title
plt.title("Time Series", fontsize=16)
# axis labels
plt.xlabel("Hour", fontsize=12)
plt.ylabel("Viewers", fontsize=12)
# Makes line plot
plt.plot(hour, viewers_hour, linewidth=2)
# Legend
plt.legend(['Jan, 1st 2015'], loc='upper left')
# %15 error range upper and lower
y_upper = [i + (i*0.15) for i in viewers_hour]
y_lower = [i - (i*0.15) for i in viewers_hour]
# shade 15% error
plt.fill_between(hour, y_lower, y_upper, alpha=0.2)

ax = plt.subplot()
# Graph background color
ax.set_facecolor('#F5F5DC')
# x ticks
ax.set_xticks(hour)
# Draw a grid
ax.grid(color='w', linestyle='solid')
# ouput
plt.show()
# Close figure
plt.close()
```