1. Introduction

1.1 Data sets

This report aims to explore data from Reddit, a community network where people can gain insight into their interests, hobbies, and passions. From the cluttered data we collected from website, we simply selected subreddits related to Ukraine, finance, and NSFW (Not Safe at Work), based on what's trending. The main focus is on changes in the number of current views across different time series, including daily and hourly changes from February to May.

All Reddit data used for this analysis came from a .csv file named *reddit.csv*. The basic information is shared as follows: There are 12 attributes and 2,705,192 pieces of data in the file. Select "RunDate", "sub_reddit" and "current_view_count" from all 12 properties to complete the tasks set for this report.

The file we chose to use has detailed data organization and clear data structure. The main dataset generated has meaningful data, and the columns used have no obvious missing values or errors, making them ideal for data analysis and visualization.

1.2 Tasks

Two main analytical goals will be achieved:

1. Reflecting the hourly and daily changes in the current view count value will be the primary task of this report, and three main subreddit topics have been selected to accomplish this task, which are topics related to Ukraine, Finance, and NSFW.
2. As for the second task of this report is to mine these three subreddits and show the details of each topic.

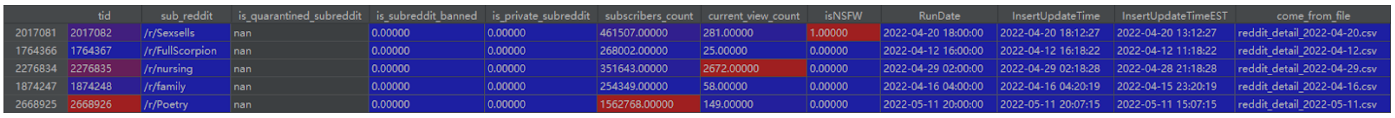
2. Data Processing

2.1 Data Importation

Import data and select columns to be used:

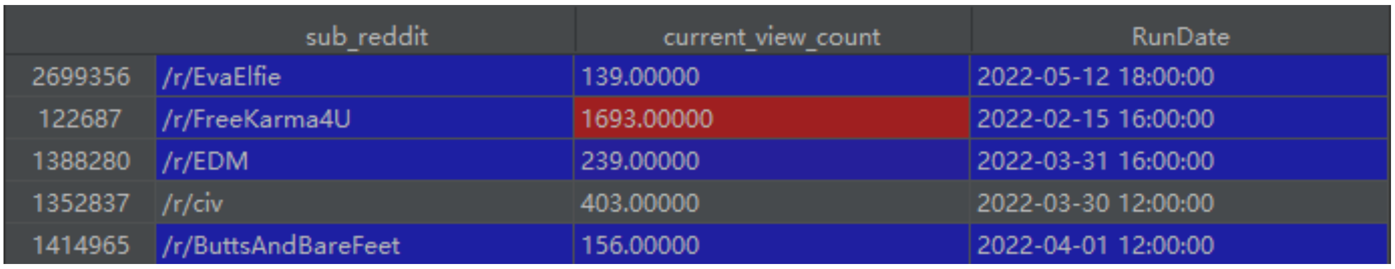
```
import pandas as pd

csv_file_original = r'.\datavisualization\reddit\reddit.csv'

# df1 is the original .csv file
df1 =pd.read_csv(csv_file_original)
# select 3 columns to for further analysis
df = pd.read_csv(csv_original, usecols=['sub_reddit', 'RunDate', 'current_view_count'])
```

Then use the sample() method to randomly select 5 sample data previews, the result graph of original .csv file shown below.



	tid	sub_reddit	is_quarantined_subreddit	is_subreddit_banned	is_private_subreddit	subscribers_count	current_view_count	isNSFW	RunDate	InsertUpdateTime	InsertUpdateTimeEST	come_from_file
2017081	2017082	/r/Sexsells	nan	0.00000	0.00000	461507.00000	281.00000	1.00000	2022-04-20 18:00:00	2022-04-20 18:12:27	2022-04-20 13:12:27	reddit_detail_2022-04-20.csv
1764366	1764367	/r/FullScorpion	nan	0.00000	0.00000	268002.00000	25.00000	0.00000	2022-04-12 16:00:00	2022-04-12 16:18:22	2022-04-12 11:18:22	reddit_detail_2022-04-12.csv
2276834	2276835	/r/nursing	nan	0.00000	0.00000	351643.00000	2672.00000	0.00000	2022-04-29 02:00:00	2022-04-29 02:18:28	2022-04-28 21:18:28	reddit_detail_2022-04-29.csv
1874247	1874248	/r/family	nan	0.00000	0.00000	254349.00000	58.00000	0.00000	2022-04-16 04:00:00	2022-04-16 04:20:19	2022-04-15 23:20:19	reddit_detail_2022-04-16.csv
2668925	2668926	/r/Poetry	nan	0.00000	0.00000	1562768.00000	149.00000	0.00000	2022-05-11 20:00:00	2022-05-11 20:07:15	2022-05-11 15:07:15	reddit_detail_2022-05-11.csv

And the result graph of .csv file after selecting three main columns:



	sub_reddit	current_view_count	RunDate
2699356	/r/EvaElfie	139.00000	2022-05-12 18:00:00
122687	/r/FreeKarma4U	1693.00000	2022-02-15 16:00:00
1388280	/r/EDM	239.00000	2022-03-31 16:00:00
1352837	/r/civ	403.00000	2022-03-30 12:00:00
1414965	/r/ButtsAndBareFeet	156.00000	2022-04-01 12:00:00

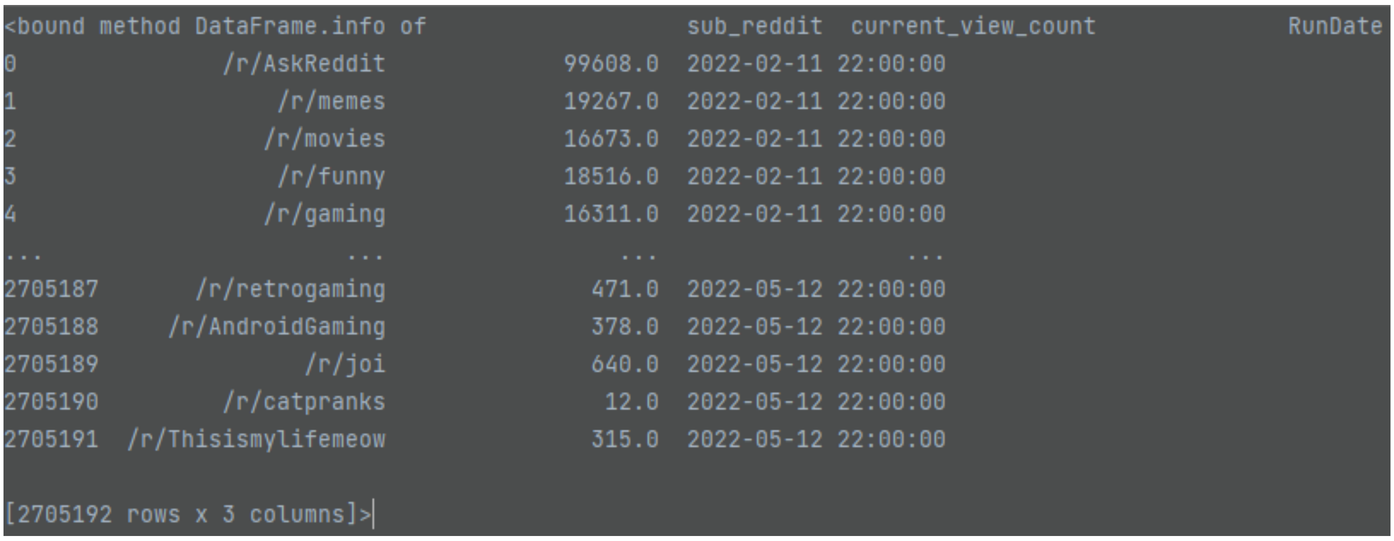
2.2 Create/Insert New Columns

Two functions are shared below for adding the "hour" column and "day" column to the pandas data frame respectively, so that these columns can be used to analyze the hourly and daily changes of the current view count value, and the main idea is to add The 'RunDate' type is changed from string type to datetime type. The built-in datetime function of pandas can directly extract the hour value from the datetime type value.

```
def insert_day(df):
    def date_formation(x):
        try:
            return pd.to_datetime(df["RunDate"][x]).strftime('%Y-%m-%d')
        except ValueError as v:
            print(v)
    df['day'] = df.index.to_series().apply(date_formation)
    return df

def format_date(df):
    def date_formation(x):
        return pd.to_datetime(df["RunDate"][x]).hour

    df['insert_h'] = df.index.to_series().apply(date_formation)
    return df
```

2.3 View Data Types

Info() function, as build-in method of pandas is used for viewing the data types of each field



```
<bound method DataFrame.info of                  sub_reddit  current_view_count            RunDate
0                /r/AskReddit           99608.0  2022-02-11 22:00:00
1                    /r/memes           19267.0  2022-02-11 22:00:00
2                   /r/movies           16673.0  2022-02-11 22:00:00
3                    /r/funny           18516.0  2022-02-11 22:00:00
4                   /r/gaming           16311.0  2022-02-11 22:00:00
...                       ...               ...                  ...
2705187         /r/retrogaming            471.0  2022-05-12 22:00:00
2705188       /r/AndroidGaming            378.0  2022-05-12 22:00:00
2705189                 /r/joi            640.0  2022-05-12 22:00:00
2705190            /r/catpranks            12.0  2022-05-12 22:00:00
2705191   /r/Thisismylifemeow             315.0  2022-05-12 22:00:00

[2705192 rows x 3 columns]>
```

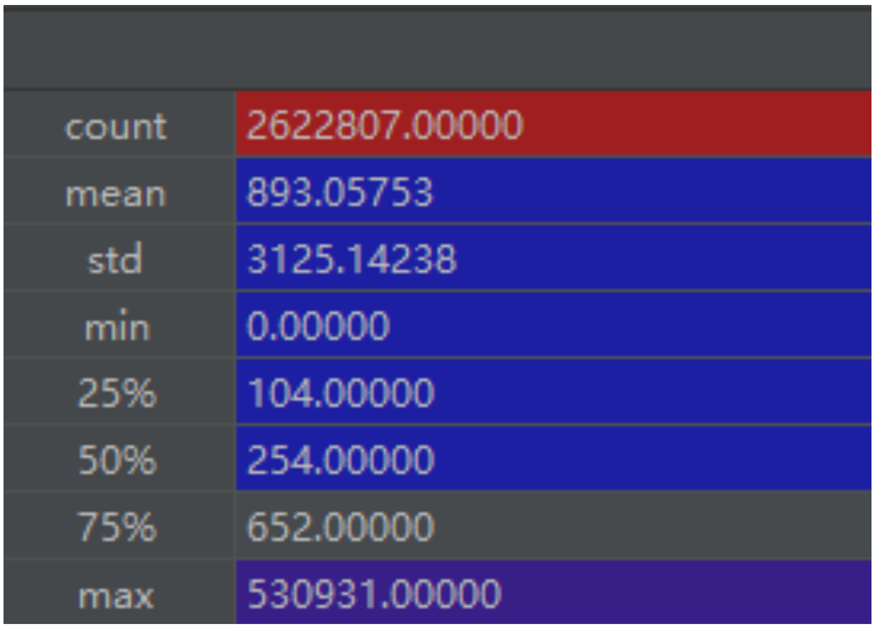
2.4 Data Cleaning

```
# delete duplicated data
df = df.drop_duplicates()
# fill Nan value with "None"(only two lines contain Nan value in this case)
df = df.fillna('None')
```

2.5 Descriptive Statistics

Here simply describe the basic statistics of the 'playerCount' attribute.

```
df[['current_view_count']].describe()
```



count	2622807.00000
mean	893.05753
std	3125.14238
min	0.00000
25%	104.00000
50%	254.00000
75%	652.00000
max	530931.00000

3. Data Analysis

The data visualization in this report uses pandas and matplotlib library as main tools, which can simplify the process of data processing and draw amazing figures. All code in this report use Python as programming language.

3.1 Hourly Current View Count of Three Sub Reddits

This section can be divided into three small parts, which are sub reddits related to Ukraine, finance and NSFW, with the main goal of refelcting how the current view count has changed over time.

3.1.1 Data Processing

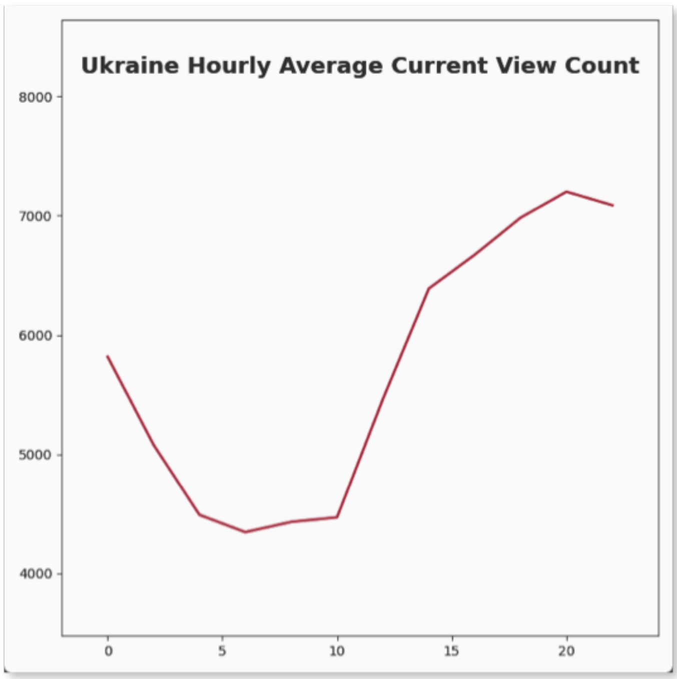
Figure 1 is created to show how current view count of Ukraine, finace and NSFW changes over hourly time. To plot lines displyed in Figure 1 , first step is to find all sub reddits that related to those topic and extract the data as the main dataset for further processing. Code of this part is as follows:

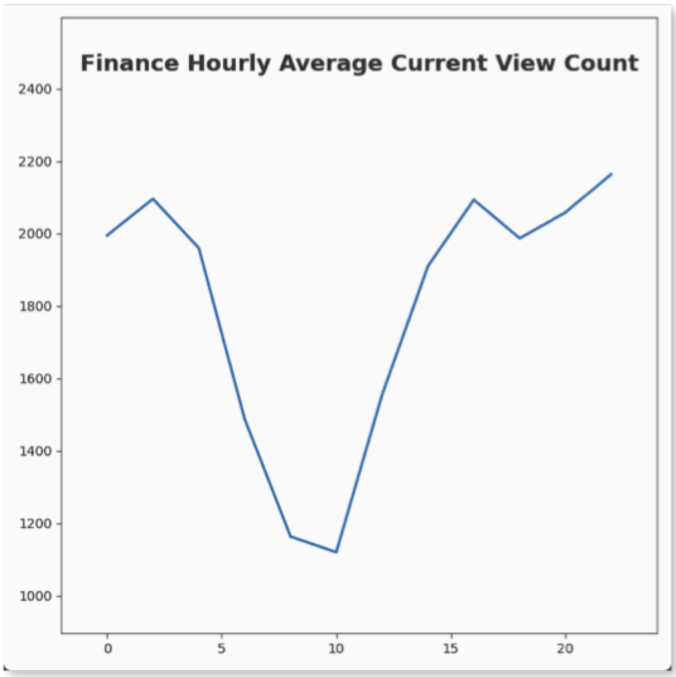
```
# add new column to label the sub_reddit that is relative to ukraine
def get_ukraine(df):
    def find_keyword(x):
        try:
            # for finance part, just simply change key_word to 'finance'
            key_word = 'ukraine'
            sub_reddit_x = df["sub_reddit"][x].lower()
            if key_word in sub_reddit_x:
                return 'T'
            else:
                return None
        except ValueError as v:
            print(v)
    df['check_keyword'] = df.index.to_series().apply(find_keyword)
    return df

def get_csv_files():
    df = pd.read_csv(csv_original, usecols=['sub_reddit', 'RunDate',
'current_view_count'])
    df = get_financial(df)
    df = df[df.check_keyword == 'T']
    df = df.reset_index(drop=True)
    df = format_date(df)
    df = df.reset_index(drop=True)
```

Code part shown below is about plotting the mean value of current view count for Ukraine-related communities, while the process for plotting the other two subreddits is similar, and the process for plotting the other two subreddits is similar. The main idea is to group the data by day values and calculate the average result of the current view count for each group, then use the matplotlib library for data visualization.

```
def plot_hourly_ukraine(df):
    df = df.groupby('insert_h').agg(total_current_count=('current_view_count', 'mean'))
    plt.rcParams["figure.figsize"] = [7.00, 7.00]
    plt.rcParams["figure.autolayout"] = True
    fig, ax = plt.subplots()

    min_x, max_x = df.index[0], df.index[-1]
    ax.set_xlim([-2, 24])
    min_y = df.total_current_count.min() * 0.8
    max_y = df.total_current_count.max() * 1.2
    ax.set_ylim([min_y, max_y])
    plt.title("Ukraine Hourly Total Current View Count", color="k", fontsize=17,
weight='bold', y=0.9)
    line, = ax.plot([], [], linestyle='-', color='C0', linewidth=2)

    x_data = df.index
    y_data = df.total_current_count
    line.set_data(x_data, y_data)
    plt.show()
```

3.1.2 Visualization & Analysis





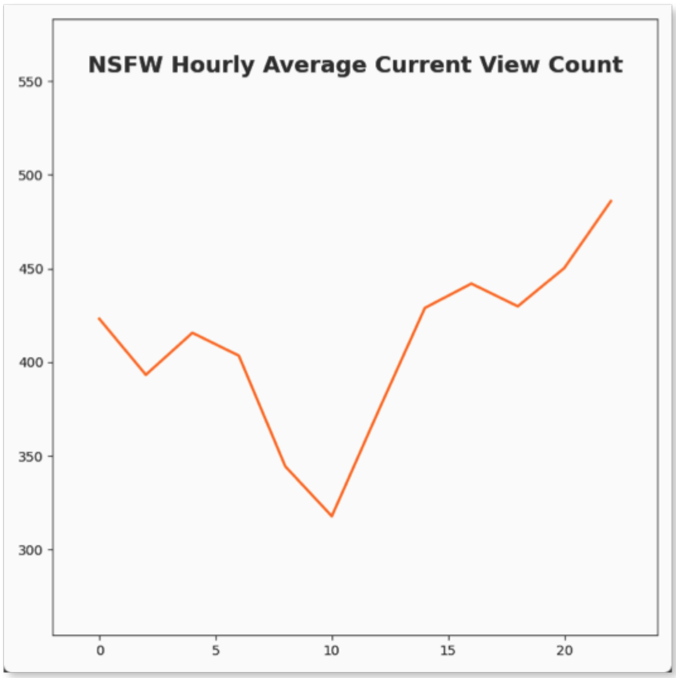


Figure 1: Hourly Current View Count of Ukraine, finance and NSFW

Figure 1 shows three lines that serve the same purpose: to display how the current number of views changes over the course of the day. Similar trends can be found in the three subplots: there is a clear drop in morning hours, and more people dive into these topics over time. Night is the time most people choose to go online to view those sub reddits. This may be because most people need to work during the day and night is the main time zone for people to relax.

3.2 Daily Current View Count of Three Sub Reddits

3.2.1 Data Processing

Since the dataset generation part is exactly the same as Task 3.1, here we only discuss the plotting part of this task.

```
def plot_with_rundate(df):
  # add new column about day datetime
    df = insert_day(df)
    df = df.groupby('day', as_index=False).agg(average_current_count=
('current_view_count', 'total'))
    df["format_day"] = pd.to_datetime(df["day"], format="%Y-%m-%d")
    df = df.set_index('format_day')
    plt.rcParams["figure.figsize"] = [7.00, 7.00]
    plt.rcParams["figure.autolayout"] = True
    fig, ax = plt.subplots()

    min_x, max_x = df.index[0], df.index[-1]
    ax.set_xlim([min_x, max_x])
    min_y = df.total_current_count.min() * 0.8
    max_y = df.total_current_count.max() * 1.2
    ax.set_ylim([min_y, max_y])
    line, = ax.plot([], [], linestyle='-', color='C2', linewidth=2)

    x_data = df.index
    y_data = df.total_current_count
    line.set_data(x_data, y_data)
    plt.legend([line], ['Ukraine Current View Count Sum'], loc='upper left',
fontsize='x-large')
    plt.show()
```

3.2.2 Visualization & Analysis

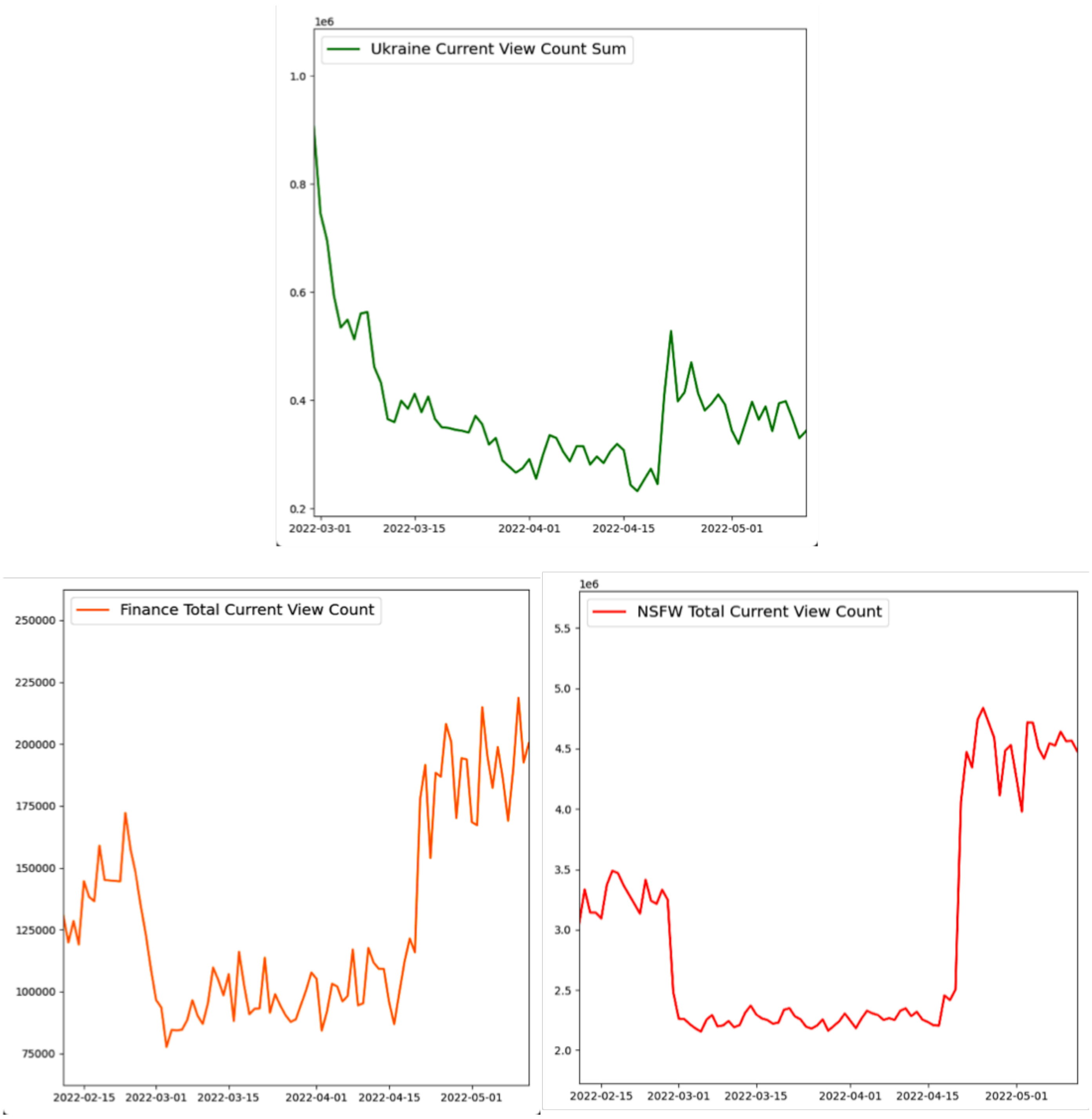


Figure 2: Daily Current View Count of Ukraine, Finance And NSFW

It can be seen from *Figure 2* that that finance and NSFW have the most similar trends. However, the overall trend of the three lines is roughly the same. The number of total current views from late March to early April were the lowest number of stages in both months.

3.3 Detailed Information of Ukraine and Finance

This section is mainly intended to display detailed information on Ukraine-related and finance-related topics (for example, sub reddit titles). Both subreddits have six subtopics.

3.3.1 Data Processing

After finding the detailed names of Ukraine, extract this data as a dataset and draw lines based on the sum of the current views for each subtitle. The code of this part is shown below:

```python
def plot_ukraine_6lines(df):
    name_list = [_.replace('/r/', '') for _ in list(df.sub_reddit.unique())]
    plt.rcParams["figure.figsize"] = [7.00, 7.00]
    plt.rcParams["figure.autolayout"] = True
    fig, ax = plt.subplots()
    ax.set_ylabel("Ukraine Sub_Reddit Current View", color="k", fontsize=13,
weight='bold')

    y_datas = {}
    names = list(df.sub_reddit.unique())
    for i in range(len(names)):
        df_new = df[df.sub_reddit == names[i]].groupby(by='insert_h').agg(new=
('current_view_count', 'sum'))
        y_datas[name_list[i]] = df_new.new
    x_data = list(df.insert_h.unique())
    for y_data in y_datas.keys():
        plt.plot(x_data, y_datas[y_data], marker='$\U0001F601$', linewidth=2,
label=y_data)
    plt.legend()
    plt.show()
```

3.3.2 Visualization & Analysis

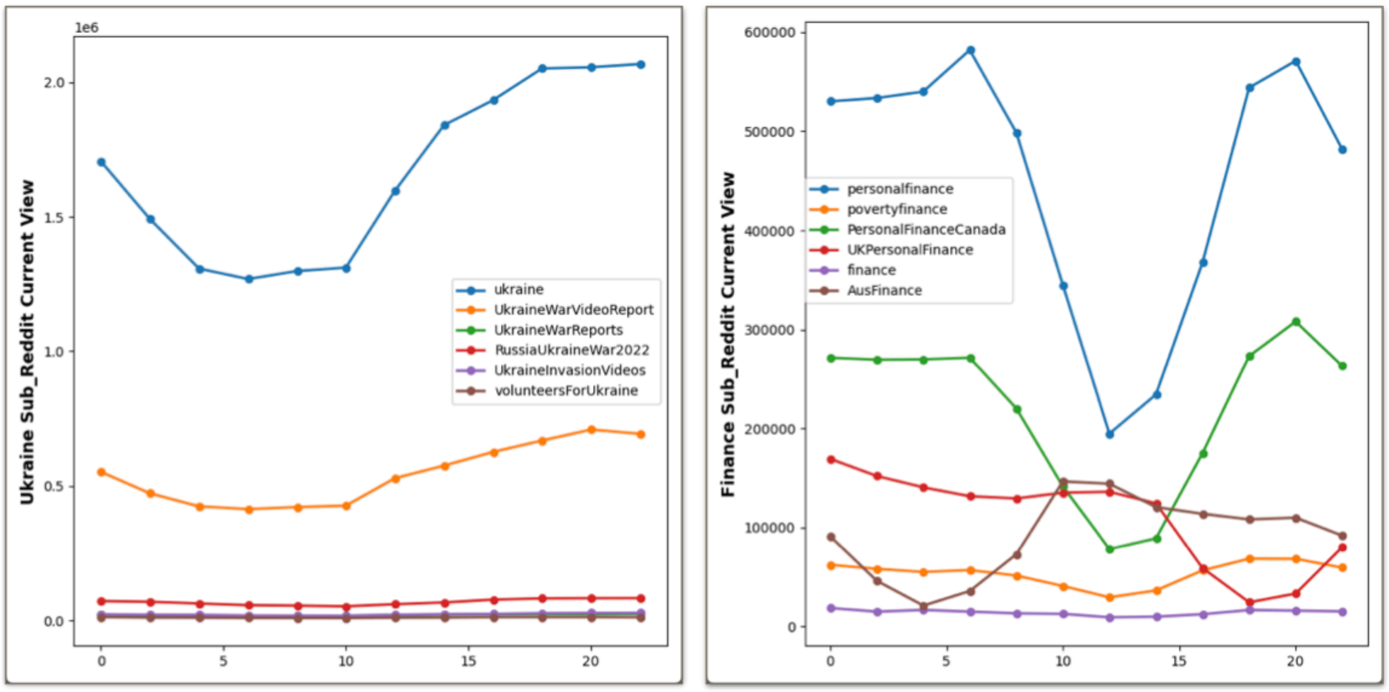


Figure 3: Hourly Current View Count of Sub Topics of Ukraine And Finance and NSFW

From Figure 3, we can get the information that the titles named "ukraine" and "personalfinance" are currently the most viewed titles.

3.4 Multiple Charts of Top Current View Count

This part aims to show top 5 current view count and top 10 most frequently appearing in the .csv file, using multiple charts type to reflect the variation.

3.4.1 Top 5 Current View Count

3.4.1.1 Data Processing (Stacked Bar Charts)

The main process in this section is firstly find top 5 max current view count of the whole dataset by sorting 'current_view_count' column, then extract those 5 sub reddits and get the total value of current view count. Finally, visualize the data using stacked bar chart. Core code for ploting those five lines could be found below.

```
# top 5 sub reddit sorted by max current view count
def plot_top_sub_reddit_day_bar(df):
    df_count = df.groupby(by='sub_reddit', as_index=False).agg(view_count=
('current_view_count', 'sum'))
    df_count = df_count.sort_values(by='view_count', ascending=False)
    df_count = df_count.reset_index(drop=True)
    name_list_original = list(df_count.sub_reddit.head(5))
    name_list = [_.replace('/r/', '') for _ in name_list_original]
    df["format_day"] = pd.to_datetime(df["day"], format="%Y-%m-%d")

    plt.rcParams["figure.figsize"] = [7.00, 7.00]
    plt.rcParams["figure.autolayout"] = True
    fig, ax = plt.subplots()
    ax.set_ylabel("Top5 Sub_Reddit Current View", color="k", fontsize=13,
weight='bold')
    y_datas = {}
    y_datas_list = []

    for i in range(len(name_list_original)):
        df_new = df[df.sub_reddit ==
name_list_original[i]].groupby(by='format_day').agg(
            new=('current_view_count', 'sum'))
        y_datas[name_list[i]] = df_new.new
        y_datas_list.append(df_new.new)
    x_data = list(df.format_day.unique())
    color_list = ['#1f77b4', '#ff7f0e', '#2ca02c', '#9467bd']
    for i in range(len(y_datas_list)):
        if i == 0:
            plt.bar(x_data, y_datas_list[i], color='r')
        else:
            plt.bar(x_data, y_datas_list[i], color=color_list[i-1])
    plt.legend(name_list)
    plt.title("Top5 Sub_Reddit Current View", color="k", fontsize=17, weight='bold',
y=1)
    plt.show()
```

3.4.1.2 Visualization & Analysis (Stacked Bar Charts)

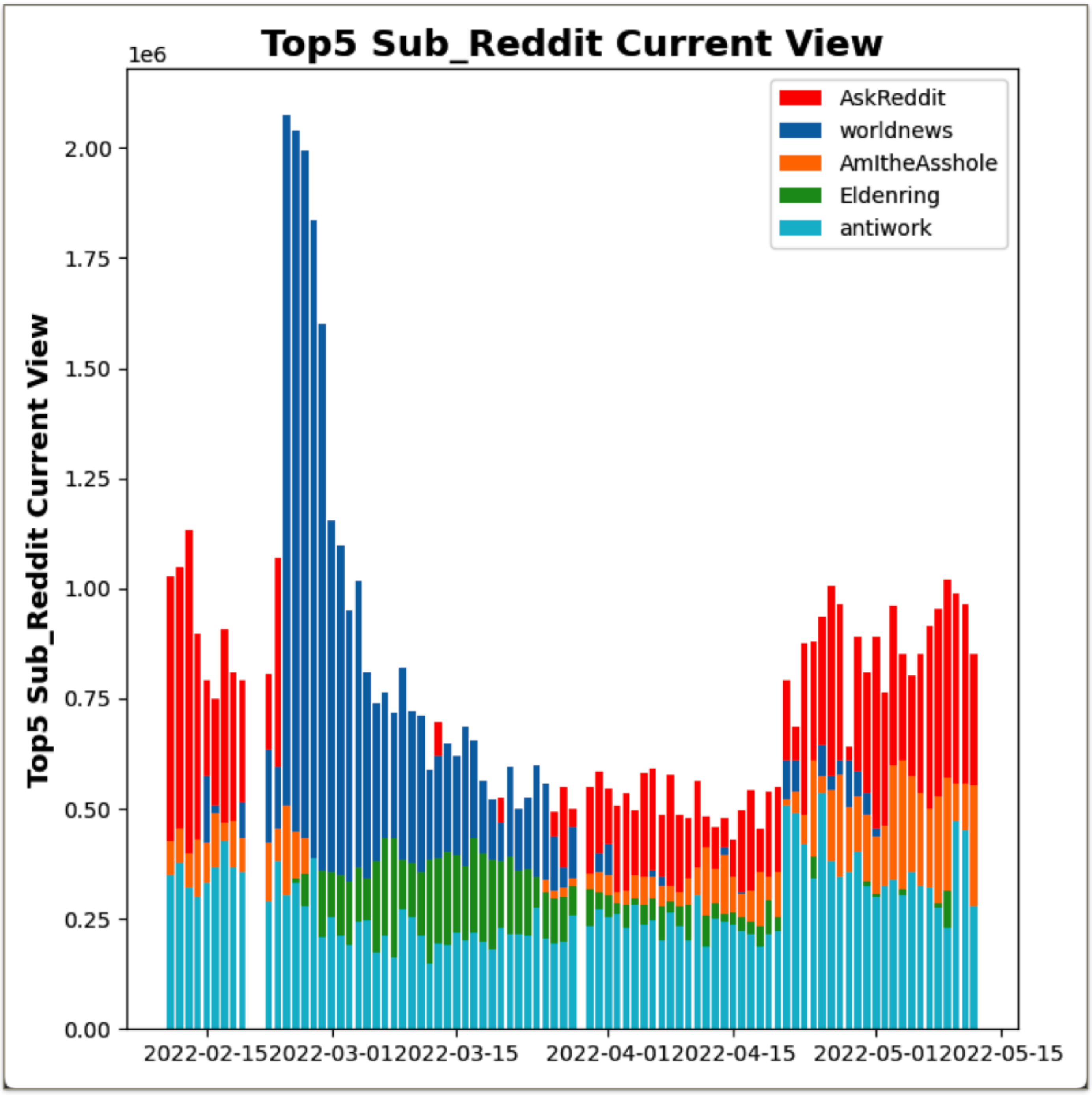


Figure 4: Stacked Bar Chart of Top5 Sub_Reddit Current View

We can see from the sorted overall bar heights that the 'worldnews' has the highest current view count in March and the 'antiwork' is lowest the whole time from February to May. Another interesting thing can be noticed by 'Eldenring' and 'worldnews' is that people only seem to suddenly increase their attention to some topics within a month.

3.4.2.1 Data Processing (Line Charts)

This section is similar to Task 3.4.1.1, the only difference is that we use line charts as the primary data visualization method. Share the code to draw 5 lines here.

```
# 5 line chart for displaying how current view count changes in April.
def plot_hourly_top_sub_reddit(df):
    df_count = df.groupby(by='sub_reddit', as_index=False).agg(view_count=
('current_view_count', 'sum'))
    df_count = df_count.sort_values(by='view_count', ascending=False)
    df_count = df_count.reset_index(drop=True)
    name_list_original = list(df_count.sub_reddit.head(5))
    name_list = [_.replace('/r/', '') for _ in name_list_original]
    df["format_day"] = pd.to_datetime(df["day"], format="%Y-%m-%d")
    df = df.set_index(pd.DatetimeIndex(df['format_day']))
    df = df.drop(['format_day'], axis=1)

    plt.rcParams["figure.figsize"] = [7.00, 7.00]
    plt.rcParams["figure.autolayout"] = True
    fig, ax = plt.subplots()
    ax.set_ylabel("Top5 Sub_Reddit Current View", color="k", fontsize=13,
weight='bold')
    y_datas = {}
    for i in range(len(name_list_original)):
        df_new = df[df.sub_reddit ==
name_list_original[i]].groupby(by='format_day').agg(
            new=('current_view_count', 'sum'))
        y_datas[name_list[i]] = df_new.new
    x_data = list(dict.fromkeys(list(df.index)))

    for y_data in y_datas.keys():
        plt.plot(x_data, y_datas[y_data], linewidth=2.5, label=y_data)
    plt.title("Top5 Sub_Reddit Current View", color="k", fontsize=17, weight='bold',
y=1)
    plt.legend()
    plt.show()
```

3.4.2.2 Visualization & Analysis (Line Charts)

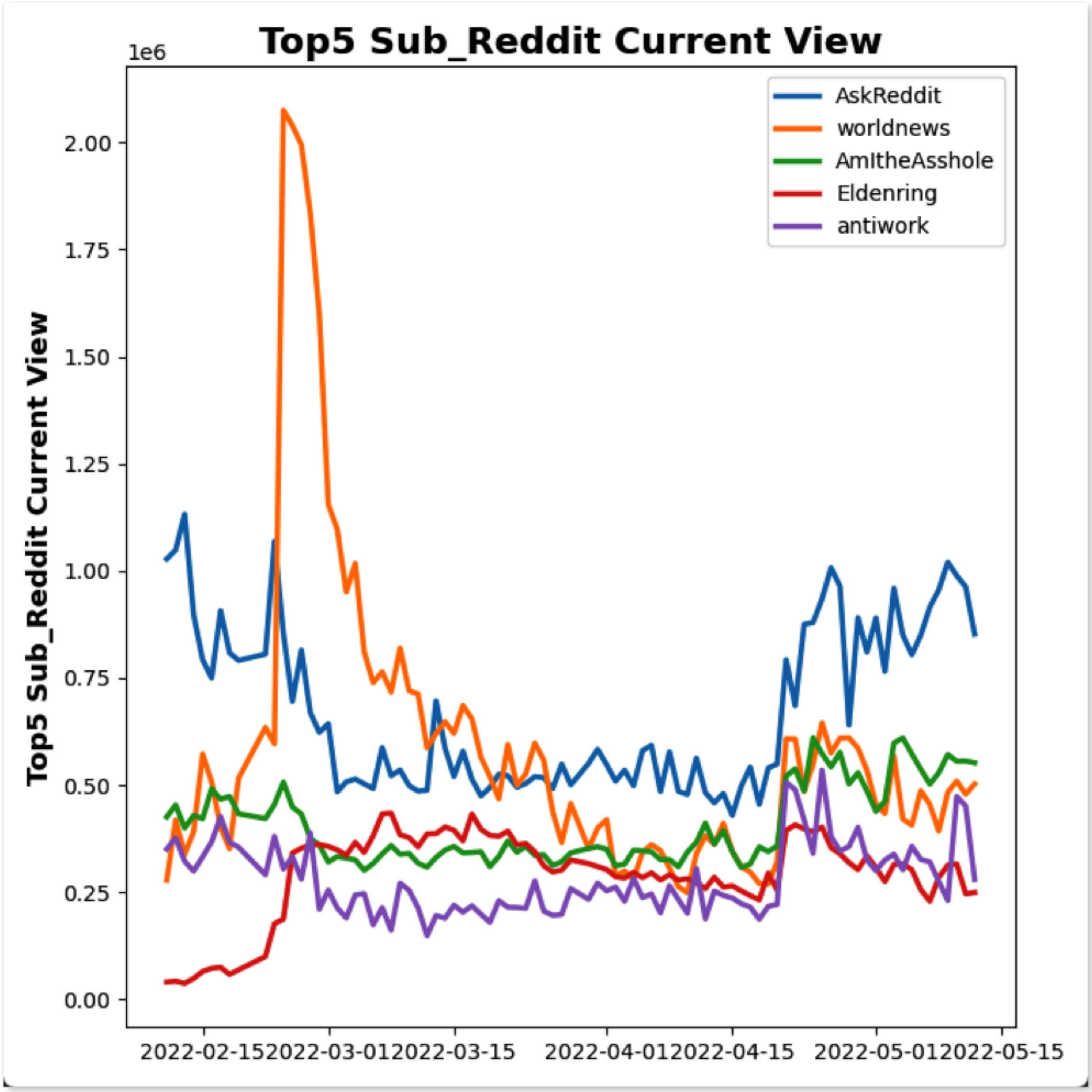


Figure 5: Line Chart of Top5 Sub_Reddit Current View

3.4.2 10 Lines

3.4.2.1 Data Processing

This section aims to analysis top 10 sub reddits that are the most frequent in the .csv file, the data procession part is similar to Task 3.4.1. Here mainly shares code for ploting 10 lines.

```
def plot_top_hourly_sub_lines(df):
    # to get group by name result -- select top 10 for further analysing
    df_name = df.groupby('sub_reddit', as_index=False).count()
    name_list_original = list(df_name.sub_reddit.head(10))
    name_list = [_.replace('/r/', '') for _ in name_list_original]
    plt.rcParams["figure.figsize"] = [7.00, 7.00]
    plt.rcParams["figure.autolayout"] = True
    fig, ax = plt.subplots()
    ax.set_ylabel("Top10 Sub_Reddit Current View", color="k", fontsize=13,
weight='bold')

    y_datas = {}
    for i in range(len(name_list_original)):
        df_new = df[df.sub_reddit ==
name_list_original[i]].groupby(by='insert_h').agg(new=('current_view_count', 'sum'))
        y_datas[name_list[i]] = df_new.new
    x_data = list(df.insert_h.unique())
    for y_data in y_datas.keys():
        plt.plot(x_data, y_datas[y_data], marker='$\U0001F601$', linewidth=2,
label=y_data)
    plt.legend(bbox_to_anchor=(1.01, 1), borderaxespad=0, loc='upper left')
    plt.show()
```

3.4.2.2 Visualization & Analysis

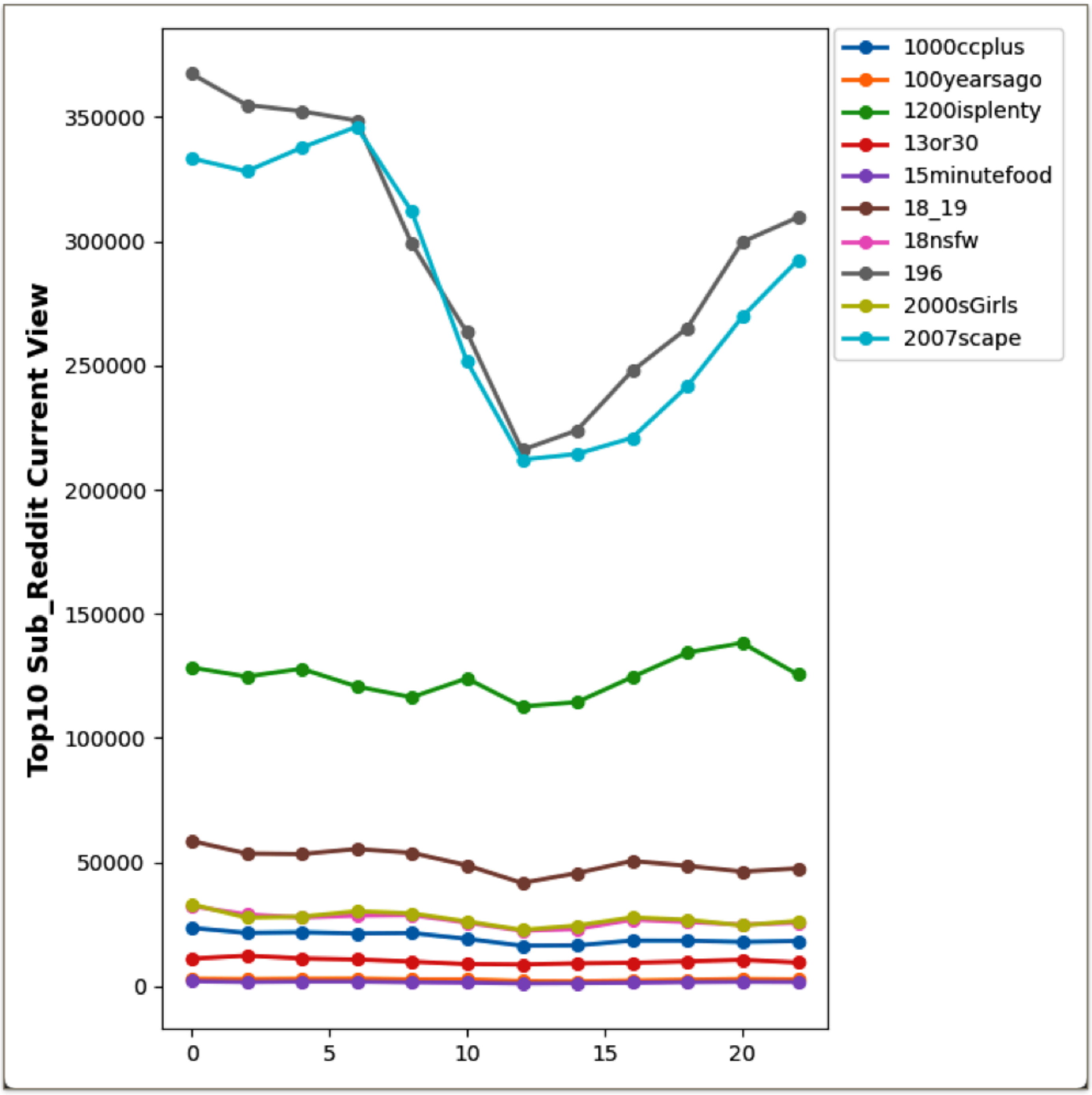


Figure 6: Line Charts of Top10 Sub_Reddit Current View

It appears that '196' and '2007scape' are the most two frequent sub reddits shown in dataset, with similar trends to Figure 1.