DATA7201 Data Analytics at Scale (2023)

Project Report on Facebook Dataset Analytics

Semester 1, 2023

Student Name: Shuo Yuan

Student ID: 46920348

Email: s4692034@student.uq.edu.au

Abstract

This report provides an exhaustive analysis of the multidimensional impact of gun-related advertisements on Facebook in the United States from March to August 2021, harnessing a dataset of nearly 20 million records. By leveraging robust data analytics and distributed system solutions, specifically tools like Hadoop, Spark on the DATA7201 cluster, and Tableau for data visualization, the study delves into the intricate subject matter through five distinct dimensions: funding entity contributions, temporal patterns in advertisement activity, demographic targeting trends, regional disparities in advertisement spending and gun violence incidents, and the influence of specific legislation on advertisement activity. The findings highlight a significant influence by a minority of entities, prominent fluctuations in ad activity, a bias towards older demographics, disparities in regional ad spend and violence incidents, and the notable impact of legislative milestones on ad activity. These insights demonstrate the value of a multidimensional approach to big data analysis and its potential to guide strategic decision-making. The report also gleans important considerations for future big data projects, emphasizing the necessity of data preprocessing, the effectiveness of tool selection, and the benefits of examining multiple analytical dimensions.

Table of Contents

Introduction 1

Dataset Analytics 2

  Funding Entity Analysis 3

  Temporal Patterns in Ad Activity 5

  Demographic Analysis 6

  Regional Analysis: Advertisement Spending, Impression and Incident Correlations 7

Discussion and conclusions of the analysis 10

Appendix 13

Introduction

The digital age is characterized by an overwhelming surge of big data, stemming from diverse sources like social media, IoT devices, and digitized business operations. Traditional data processing tools, such as SQL, struggle with this mounting data volume, velocity, and variety, necessitating the use of robust big data analytics (Chai et al., 2021). Managing this enormous data and its complexities requires powerful distributed system solutions. Tools like Hadoop and Spark, utilizing advanced algorithms like MapReduce, have become indispensable in effectively addressing these challenges (Pathak, n.d.).

Facebook, a social media behemoth that handles a vast amount of data from billions of users daily, posing significant data management challenges. To overcome these, Facebook leverages big data technologies such as HBase for distributed storage and Apache Hive for real-time user data analytics, thereby optimizing data processing and enabling precise, data-driven marketing approaches.

Similarly, Uber, a provider of app-based taxi services, addressing vehicle optimization issues by utilizing big data methodologies. By collecting data points such as GPS locations and traffic information, Uber enhances its services and fare calculations. It employs Hadoop for data management and Apache Spark for data processing, exemplifying the effectiveness of a data-centric business model in solving real-world problems (Marr, 2016).

Overall, big data analytics and distributed system solutions are integral to various industries including healthcare, finance, transportation, and more. Thus, understanding and implementing them has become a primary requirement for any data-driven industry (M, 2023).

Dataset Analytics

This report focuses on the impact of Facebook political gun-related ads in the U.S. from March to August 2021. The primary source of data for this analysis is the Facebook Ad Library API, an exhaustive collection of sponsored political posts aimed at U.S. users from March 2020 to January 2022. This API data is coupled with two additional data sets to enrich the analysis:

1. The Gun Violence dataset provides detailed statistics on gun-related incidents in the U.S., facilitating a clear understanding of the real-world context.
2. The Legislation dataset, sourced from congress.gov, covers all gun-related bills introduced during the same period, offering a governmental perspective on the gun debate and its influence.

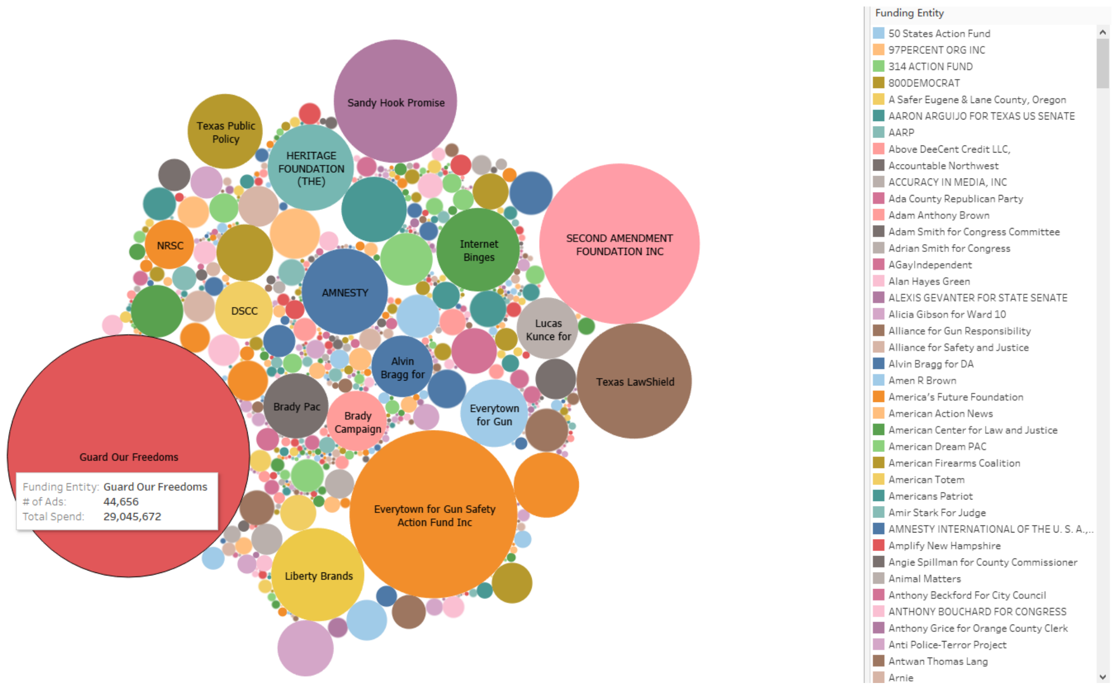
While the data for analysis spans only half a year, it still amounts to nearly 20 million records. Therefore, selecting appropriate big data processing tools is vital. I utilize Spark and HDFS on the DATA7201 cluster for data processing, and Tableau for visualization.

Data pre-processing was crucial to ensure accuracy and relevance. The Facebook Ad Library API was manipulated to identify gun-related ads from March to August 2021. The “ad_creative_body” was converted to lowercase utilizing the lower() function and filtered for the keyword "gun". Missing values were removed, duplicates eliminated, and lower/upper bounds derived for “spend” and “impressions” to compute “average_spend” and “average_impressions” by adding the lower and upper bounds and dividing by 2.

A comprehensive approach is necessary to address the complex and socially relevant gun-related issues. This report will explore them from five dimensions, including: the key contributors to the ads; temporal patterns in ad activity; demographic targets of ad activity; regional variations in ad spending and the correlation with incident rates; and the influence of specific legislations on ad activity.

Funding Entity Analysis

This analysis starts by looking at the funding entity to see who is leading the gun conversation on Facebook, and the scale of their financial commitments. I used the groupBy() function to group the dataset by "funding_entity" and then two key metrics were calculated for each entity: total number of ads and cumulative spend - using count() and sum() functions.



Funding Entity: Guard Our Freedoms
# of Ads: 44,656
Total Spend: 29,045,672

Figure 1. Advertisement Quantity Sponsored by Different Entities

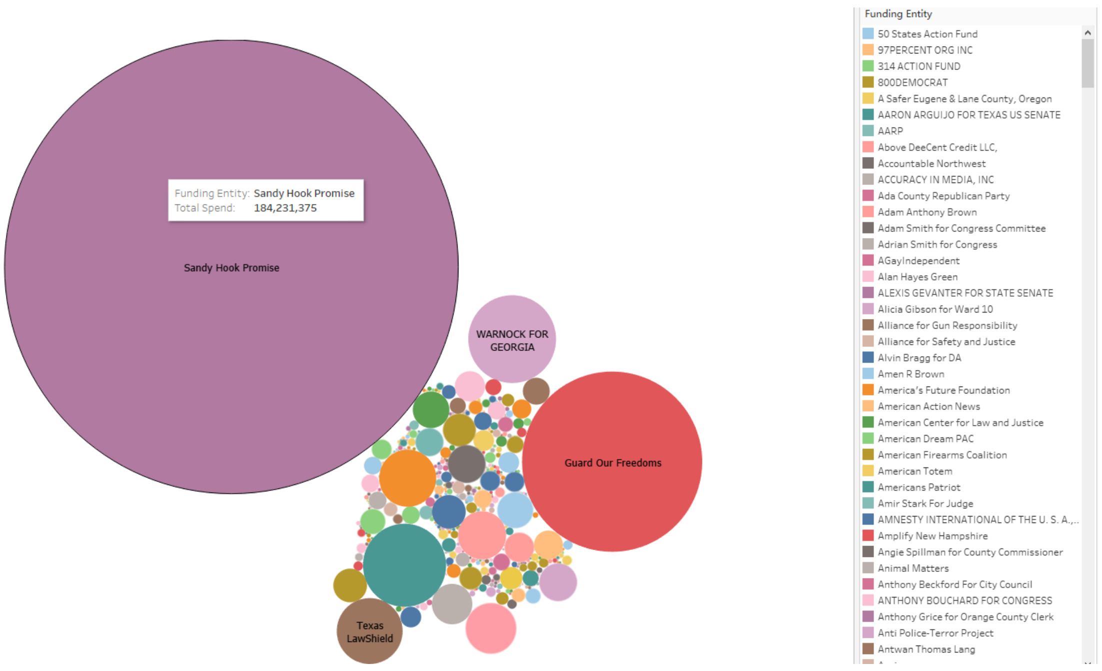


Figure 2. Sponsorship Spend by Different Entities

The distribution of entities, as shown in Figures 1 and 2, demonstrates that a minority of dominant entities is responsible for a significant portion of ad spend, while a long list of smaller entities exists. For example, "Sandy Hook Promise" sponsored fewer ads but spent six times more than "Guard Our Freedoms".

Other notable entities included "Everytown for Gun Safety Action Fund Inc" is known for its advocacy for gun control measures. In contrast, entities such as "GunAssociation.org" signified the involvement of pro-gun rights groups (Crummett, 2021). Stakeholders should closely monitor these influential entities to the gun-related ads to understand their objectives and strategies.

Temporal Patterns in Ad Activity

I aim to identify potential correlations by examining the timing, spending, and impressions of gun-related ads. I aggregated the data monthly by grouping it based on the “Month”, which is extracted from the “ad_delivery_start_time” using groupBy() and todate() function. Then, I used the sum() function on the 'average_spend' and 'average_impressions' fields to calculate monthly total spend and impressions, respectively.

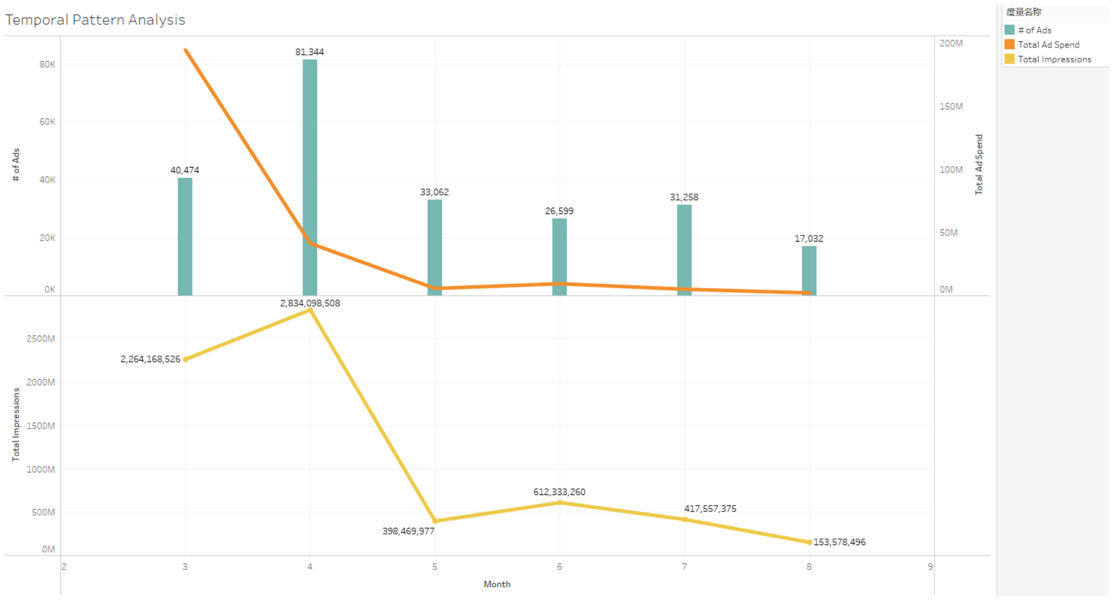


Figure 3. Monthly Trends in number of Ads, Ad Spend, and Impressions

As depicted in Figure 3, advertising spending in March was relatively high. However, the number of ads increased to 81,344 in April despite lower total spending. The number of Ads and impressions decreased significantly as summer approached, with only 17,032 ads by August. Stakeholders must pay attention to the monthly fluctuations in number of ads and impressions to devise effective timing strategies for their campaigns.

Demographic Analysis

With a comprehensive grasp of the monthly advertising trends, my next step was to analyze the targeted audience. I initially utilized the explode() function on the "demographic_distribution" field to break it down into its components of "age", "gender", and "percentage". An additional "estimated_impressions" column was calculated by multiplying the "average_impressions" by the corresponding "percentage" for each demographic. Finally, I applied the groupBy() function to group the data by "age" and "gender".

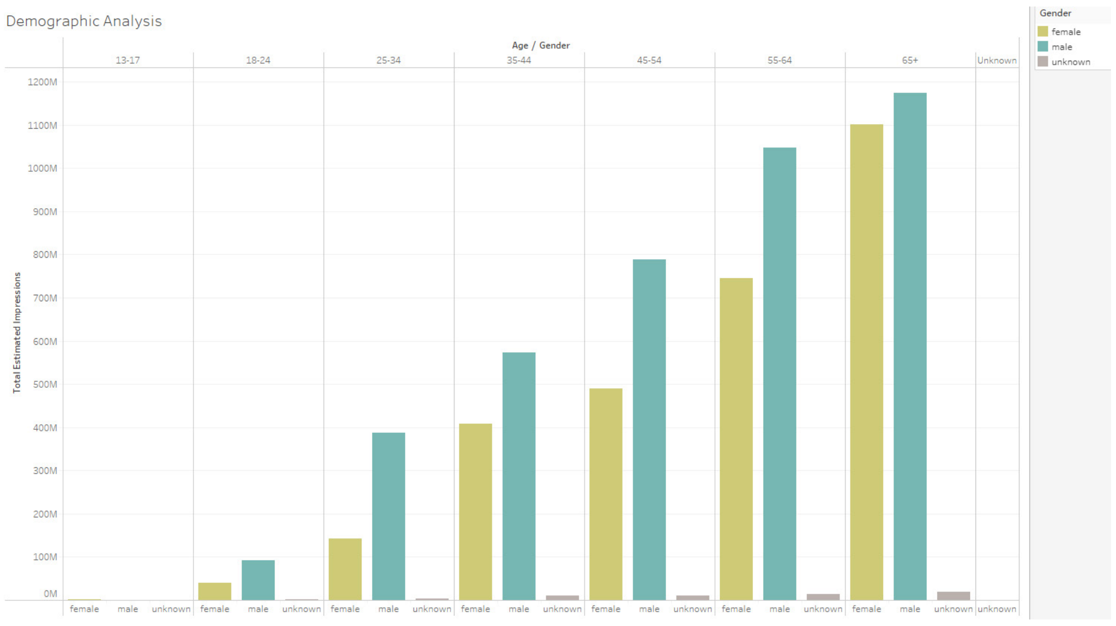
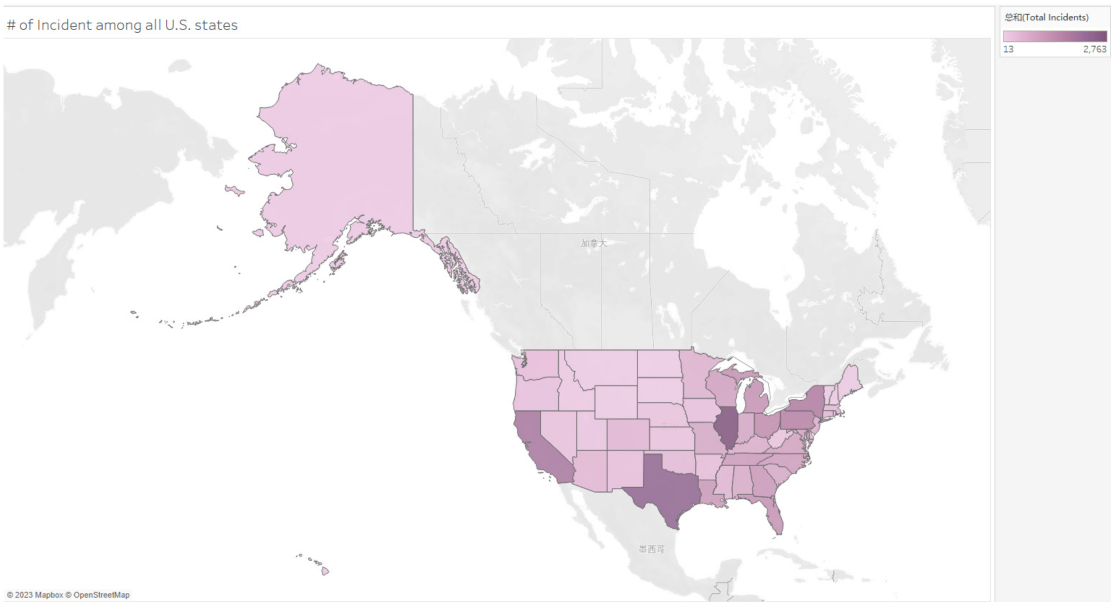


Figure 4. Age and Gender Distribution of Estimated Impressions

As evident in Figure 4, the majority of the ads target older people, especially over 65 years old. As the age group gets younger, the number of impressions keeps decreasing. For instance, men aged 25-34 only received 3.88E+08 impressions, almost a third of the 65+ age group. Stakeholders targeting a broader or younger audience may need to revise their strategies.

Regional Analysis: Advertisement Spending, Impression and Incident Correlations

This section investigates correlations between ad activity and levels of gun violence across states. Similar to the demographic analysis, I exploded the “region_distribution” field and grouped the data by “region”, summing up the “estimated_spend” and “estimated_impressions”. For the Gun Violence data, I grouped the data by “State” and calculate the total number of gun violence incidents, then join it with Facebook API on “region”.



# of Incident among all U.S. states
总和(Total Incidents)
13
2,763
加拿大
墨西哥
© 2023 Mapbox © OpenStreetMap

Figure 5. Gun-related Incidents Across States

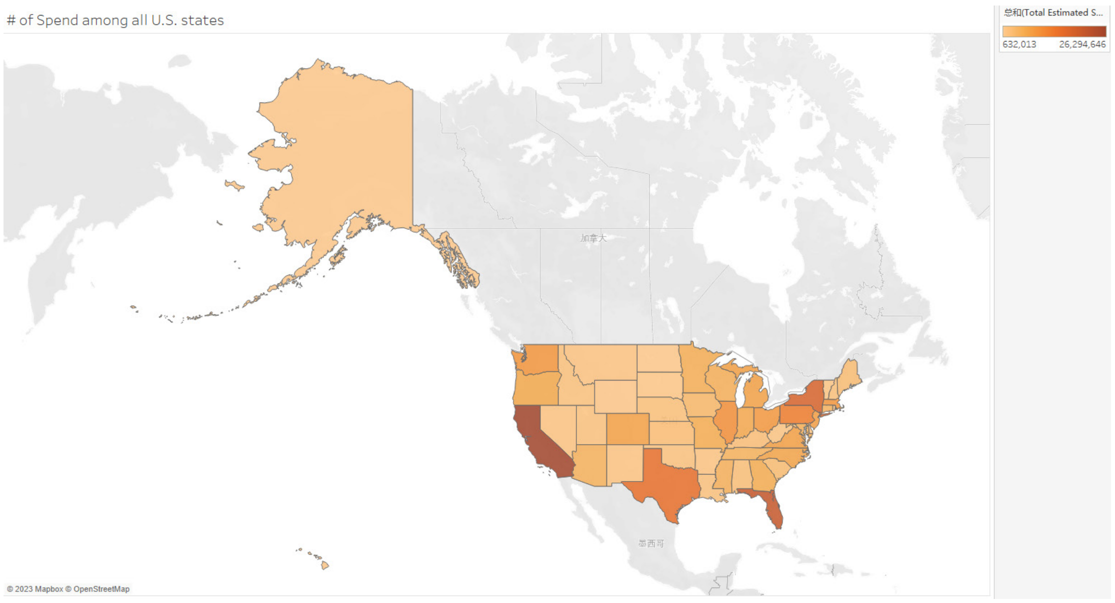

# of Spend among all U.S. states
总和(Total Estimated S...
632,013
26,294,646
加拿大
墨西哥
© 2023 Mapbox © OpenStreetMap

Figure 6. Regional Distribution of Gun-related Ad Spending

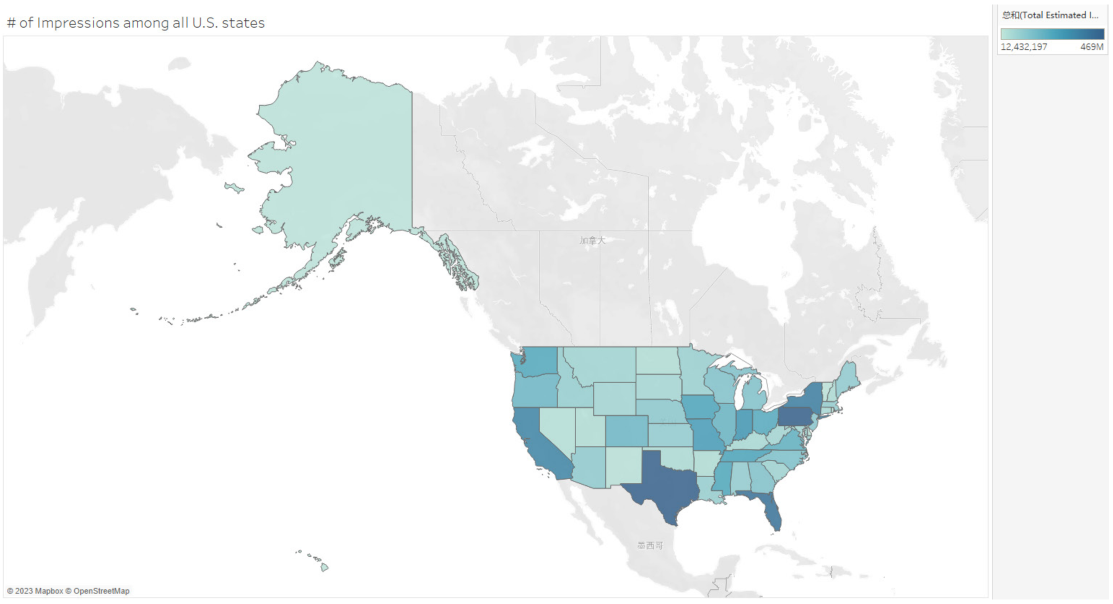

# of Impressions among all U.S. states
总和(Total Estimated I...
12,432,197
469M
加拿大
墨西哥
© 2023 Mapbox © OpenStreetMap

Figure 7. Impressions of Gun-related Ads Across States

Unsurprisingly, states with higher ad spend, such as California, Texas and Florida, also had higher impressions. This aligns with the common understanding that more spending leads to more ad visibility and reach. However, the correlation between spending and the number of incidents is not always straightforward. For instance, Illinois has a lower spend than Texas or California, but has the highest number of incidents. Overall, the correlation between ad spends, impressions, and incidents volume is more complex due to various factors such as state laws, cultural attitudes toward firearms, and socioeconomic conditions. Stakeholders should account for regional variations in ad spend and gun violence incidents.

Impact of Specific Legislation on Ad Activity

Finally, I want to explore the changes in ad activity around the introduction of legislation. I utilized date_sub() and date_add() functions to extract three days "start window" and "end window" surrounding the introduction of legislation. Only ads with "ad_delivery_start_time" falling within these windows were included. Finally, I aggregated the Legislation data by “Name” field, counting unique ads and summing “average_spend” and “average_impressions” for each piece of legislation.

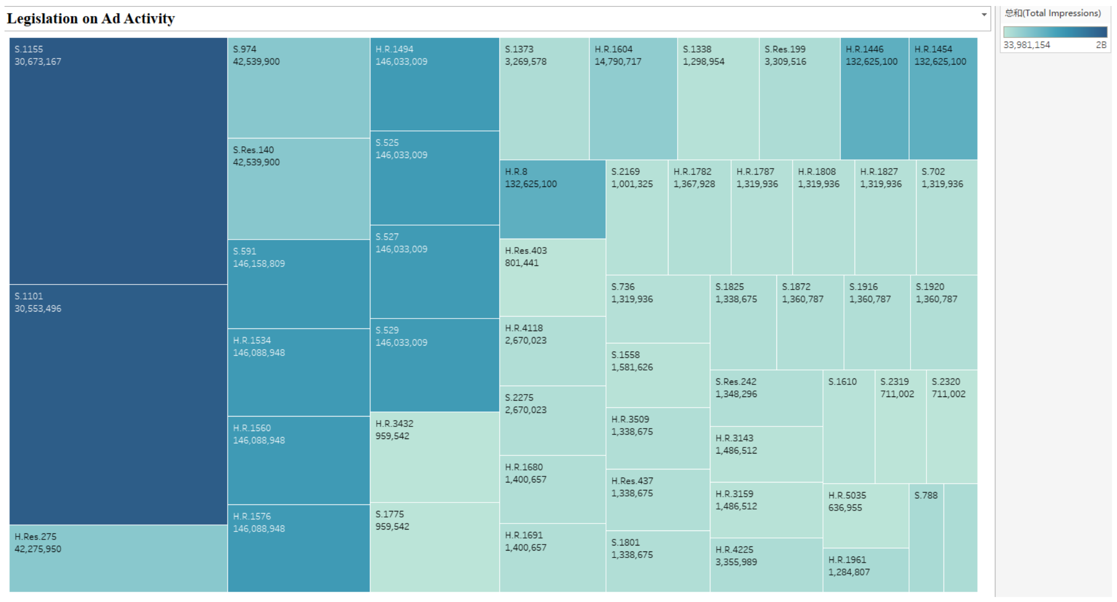


Figure 8. Legislative Impact on Gun Control Advertisements

Figure 8 reveals diverse advertising responses to legislation. For example, S.1155 and S.1101 triggered over 51,000 ads with spending exceeding \$30.7 million and impressions exceeding 2 billion each. Conversely, legislations like H.R.1961 and S.788 incited a quieter response with 3,700 ads and \$1.28 million spend.

Certain legislations exhibited a significant spend despite a lower volume of ads. Specifically, the number of ads for H.R.5035 was slightly over 5300, but the total spend during this period was close to $636,954.5. In general, the introduction of specific legislation has had a significant impact on ad activity. Monitoring changes in advertising activities around new legislation could provide critical insights for stakeholders, guiding their advertising strategies.

Discussion and conclusions of the analysis

This report examined the multidimensional impact of gun-related ads on Facebook in U.S. between March and August 2021 using 1,587 words. Different perspectives highlight the complexity of the issue and underline the importance of a multi-dimensional approach to big data analysis. The main conclusions from the various analytical dimensions can be summarized as follows:

- Funding Entity Analysis: A minority of entities contribute to a significant proportion of the ad spend, shaping the gun conversation on Facebook. Key entities such as "Sandy Hook Promise" was observed to spend heavily and run numerous ads, revealing a clear need to monitor these entities closely.
- Temporal Patterns in Ad Activity: The temporal distribution of gun-related ads exhibited conspicuous fluctuations, with number of ads peaking in April.
- Demographic Analysis: The analysis highlighted a clear trend towards older demographics in ad targeting. It suggests a potential need to reassess the strategies for reaching a younger audience.
- Regional Analysis: There were disparities in ad spend, impressions, and gun violence incidents across different regions. The findings indicate that high ad spending did not necessarily translate into fewer violence incidents, pointing to the influence of localized factors.

- Impact of Specific Legislation on Ad Activity: The introduction of new legislation induced varying responses in ad activity, revealing the need to strategize advertising efforts based on legislative milestones.

Through the course of this project, the following insights for future big data projects were gleaned:

- Data preprocessing is crucial: The process of data cleaning and transformation, including handling missing values, duplicates, and format standardization, is a vital step to ensure accuracy and relevance in big data analysis.
- Effective tool selection: Given the massive volume of data, selecting suitable big data processing tools, such as Spark in this case, is essential. Conventional methods may quickly run out of memory or significant slowdowns, hampering project efficiency.
- Multi-dimensional approach: Taking into account multiple perspectives such as funding entities, temporal patterns, demographics, regional differences and legislative impacts allows for a more comprehensive understanding of the topic. It reveals complex patterns and correlations that might be overlooked in a single-dimension analysis.

The diverse findings derived from different analytical dimensions provide a holistic perspective of the subject, enabling stakeholders to strategize their future actions more effectively. The lessons learned in this project emphasize the importance of a robust and multidimensional approach to big data analysis, further underscoring its value in influencing data-driven decisions.

Reference

Chai, W., Labbe, M., & Stedman, C. (2021). big data analytics. *Business Analytics*. https://www.techtarget.com/searchbusinessanalytics/definition/big-data-analytics

Crummett, D. (2021). Freedom, Firearms, and Civil Resistance. *The Journal of Ethics*, *25*(2), 247–266. https://doi.org/10.1007/s10892-021-09365-3

M, S. (2023). Top Big Data Applications Across Industries. *Simplilearn.com*. https://www.simplilearn.com/tutorials/big-data-tutorial/big-data-applications

Marr, B. (2016). *Big Data in Practice: How 45 Successful Companies Used Big Data Analytics to Deliver Extraordinary Results*. http://doi.wiley.com/10.1002/9781119278825

Pathak, R. (n.d.). *Top 10 Big Data Analytics Tools | Analytics Steps*. https://www.analyticssteps.com/blogs/top-10-big-data-analytics-tools

External Datasets:

- Congress.gov | Library of Congress. (n.d.). Legislation Data
- Gun Violence Archive. (n.d.). Gun Violence Data:

Appendix

```
# %%
from pyspark.sql import SparkSession
from pyspark.sql.functions import to_date, year, month, count, col,
sum, lower, explode, date_add, date_sub

spark = SparkSession.builder.appName('s4692034-project').getOrCreate()
spark

# %%
# Define a function to generate file paths for a given month
def generate_file_paths(year, start_month, end_month):
    file_paths = []
    for month in range(start_month, end_month + 1):
        file_paths.append(f'/data/ProjectDatasetFacebook/FBads-US-
{year}{month}' + "*")
    return file_paths

# Generate file paths for the desired months
file_paths = generate_file_paths(2021, 3, 8)

# Define the columns you're interested in
cols = ["id", "ad_creative_body", "demographic_distribution",
"region_distribution", "spend", "impressions", "funding_entity",
"ad_delivery_start_time"]

# Read each file into a DataFrame and concatenate them into one
dfs = [spark.read.json(path).select(*cols) for path in file_paths]
df = dfs[0]
for other_df in dfs[1:]:
    df = df.union(other_df)

# %%
path = "/user/s4692034/projectData/df_all"
df.write.mode('overwrite').json(path)

# %%
# Convert ad_creative_body to lowercase and add as a new column
```

```
df = df.withColumn("ad_creative_body_lower",
lower(col("ad_creative_body")))

# Filter rows where ad_creative_body_lower contains the word 'gun'
df_gun = df.filter(col("ad_creative_body_lower").rlike("\\bgun\\b"))

# Remove the temporary column
df_gun = df_gun.drop("ad_creative_body_lower")

# Drop missing values
df_gun = df_gun.na.drop(subset=cols)

# Remove duplicates
df_gun = df_gun.dropDuplicates()

# Extract lower_bound and upper_bound from spend and impressions
df_gun = df_gun.withColumn("spend_lower_bound",
df_gun["spend.lower_bound"])\
              .withColumn("spend_upper_bound",
df_gun["spend.upper_bound"])\
              .withColumn("impressions_lower_bound",
df_gun["impressions.lower_bound"])\
              .withColumn("impressions_upper_bound",
df_gun["impressions.upper_bound"])

# Calculate the average spend and impressions
df_gun = df_gun.withColumn("average_spend",
(df_gun["spend_lower_bound"] + df_gun["spend_upper_bound"]) / 2)
df_gun = df_gun.withColumn("average_impressions",
(df_gun["impressions_lower_bound"] + df_gun["impressions_upper_bound"])
/ 2)

# Convert the 'ad_delivery_start_time' to datetime
df_gun = df_gun.withColumn("ad_delivery_start_time",
to_date(col("ad_delivery_start_time"), "yyyy-MM-dd"))
df_gun = df_gun.withColumn("Month",
month(col("ad_delivery_start_time")))
df_gun = df_gun.withColumn("Year", year(col("ad_delivery_start_time")))

print(df_gun.count())
print(df_gun.columns)
# df_gun.show(10)
# %%
path = "/user/s4692034/projectData/df_gun"
df_gun.write.mode('overwrite').json(path)

# %%
path = "/user/s4692034/projectData/df_gun"
df_gun = spark.read.json(path)
print(df_gun.columns)

# %% [markdown]
# ### Look at funding entities and spend to see which ones are spending
and putting the most

# %%
# Group by funding entity
df_funding = df_gun.groupBy("funding_entity")\
                  .agg(count("id").alias("# of Ads"),
                       sum("average_spend").alias("Total Spend"))

# Order by number of ads and total spend
df_funding = df_funding.orderBy(["# of Ads", "Total Spend"],
ascending=False)

df_funding.show()

# %%
path = "/user/s4692034/projectData/df_funding"
df_funding.write.mode('overwrite').json(path)
df_funding.coalesce(1).write.mode('overwrite').csv("/user/s4692034/proj
ect/funding_entity", header=True)

# %% [markdown]
# ### Load GVA and LGS datasets

# %%
# Load Gun Violence data
df_violence = spark.read.csv('/user/s4692034/external-
dataset/GVA_2021.3.1-8.31.csv', header=True)

# Convert the 'Incident Date' to datetime
```

```
df_violence = df_violence.withColumn('Incident Date',
to_date(col("Incident Date"), "d-MMM-yy"))

# Group by month and count the number of incidents
df_violence = df_violence.withColumn("Month", month(col("Incident
Date")))
df_violence = df_violence.withColumn("Year", year(col("Incident
Date")))
df_violence_grouped = df_violence.groupBy("Month",
"Year").agg(count("Incident ID").alias("# of Violence
Incidents")).sort("Month", "Year")
df_violence_grouped = df_violence_grouped.filter(col("Year") == 2021)

df_violence_grouped.show()

# Load Legislation data
df_legislation = spark.read.csv('/user/s4692034/external-
dataset/LGIS_2021.3.1-8.31.csv', header=True)

# Convert the 'Introduced Date' to datetime
df_legislation = df_legislation.withColumn("Introduced Date",
to_date(col("Introduced Date"), "MM/dd/yyyy"))

# Group by month and count the number of bills
df_legislation = df_legislation.withColumn("Month",
month(col("Introduced Date")))
df_legislation = df_legislation.withColumn("Year", year(col("Introduced
Date")))
df_legislation_grouped = df_legislation.groupBy("Month",
"Year").agg(count("Name").alias("# of Legislation
Introduced")).sort("Month", "Year")
df_legislation_grouped = df_legislation_grouped.filter(col("Year") ==
2021)

df_legislation_grouped.show()

# %%
path = "/user/s4692034/projectData/df_violence"
df_violence.write.mode('overwrite').json(path)

path = "/user/s4692034/projectData/df_legislation"
df_legislation.write.mode('overwrite').json(path)
```

```
# %%
path = "/user/s4692034/projectData/df_violence"
df_violence = spark.read.json(path)

path = "/user/s4692034/projectData/df_legislation"
df_legislation = spark.read.json(path)

# %% [markdown]
# ### Change in the number of ads per month

# %%
df_gun_grouped = df_gun.groupBy("Month",
"Year").agg(sum("average_spend").alias("Total Ad Spend"),
sum("average_impressions").alias("Total Impressions"),
count("*").alias("# of Ads")).filter("Year == 2021")

# Join on 'Month'
df_pattern = df_gun_grouped.join(df_violence_grouped, on=["Month",
"Year"], how="outer")
df_pattern = df_pattern.join(df_legislation_grouped, on=["Month",
"Year"], how="outer")
df_pattern = df_pattern.select("Month", "Year", "# of Ads", "Total Ad
Spend", "Total Impressions",
                                "# of Violence Incidents", "# of
Legislation Introduced")

df_pattern.show()

# %%
path = "/user/s4692034/projectData/df_pattern"
df_pattern.write.mode('overwrite').json(path)
df_pattern.coalesce(1).write.mode('overwrite').csv("/user/s4692034/proj
ect/pattern_analysis", header=True)

# %% [markdown]
# ### The relationship between age, gender and impression

# %%
# Explode the demographic_distribution array
df_demographics = df_gun.select("id", "average_impressions",
explode("demographic_distribution").alias("demographics"))
```

```
# Select useful columns and calculate the estimated impressions for
each demographic group
df_demographics = df_demographics.select(
    "id",
    col("demographics.age").alias("age"),
    col("demographics.gender").alias("gender"),
    col("demographics.percentage").alias("percentage"),
    (col("average_impressions") *
col("demographics.percentage")).alias("estimated_impressions"),
)

# Group by age_group and gender and sum the estimated impressions
df_demographics_grouped = df_demographics.groupBy("age",
"gender").agg(sum("estimated_impressions").alias("total_estimated_impre
ssions")).sort("total_estimated_impressions", ascending=False)

df_demographics_grouped.show()

# %%
path = "/user/s4692034/projectData/df_demographics"
df_demographics.write.mode('overwrite').json(path)
df_demographics_grouped.coalesce(1).write.mode('overwrite').csv("/user/
s4692034/project/demographics", header=True)

# %% [markdown]
# ### The relationship among the number of incidents, the number of
ads, spend and impression in different regions

# %%
# Explode the region_distribution array
df_region = df_gun.select("id", "average_spend", "average_impressions",
explode("region_distribution").alias("regions"))

# Select useful columns and calculate the estimated spend and
impressions for each region
df_region = df_region.select(
    "id",
    col("regions.region").alias("region"),
    col("regions.percentage").alias("percentage"),
    (col("average_spend") *
col("regions.percentage")).alias("estimated_spend"),
    (col("average_impressions") *
col("regions.percentage")).alias("estimated_impressions"),
)
df_region = df_region.filter(col('region') != 'Unknown')

# Group by region and sum the estimated spend and impressions
df_region_grouped = df_region.groupBy("region").agg(
    sum("estimated_spend").alias("total_estimated_spend"),
    sum("estimated_impressions").alias("total_estimated_impressions")
)

# Group the already loaded Gun Violence data by State and count the
number of incidents and victims
df_violence_state = df_violence.groupBy("State").agg(
    count("Incident ID").alias("# of Violence Incidents"),
    sum("# Victims Injured").alias("Total Victims Injured"),
    sum("# Victims Killed").alias("Total Victims Killed")
)

# Add Total Victims Injured and Total Victims Killed into one column
named "Total Victims"
df_violence_state = df_violence_state.withColumn("Total Victims",
col("Total Victims Injured") + col("Total Victims
Killed")).withColumnRenamed("# of Violence Incidents", "Total
Incidents")

# Drop the original columns
df_violence_state = df_violence_state.drop("Total Victims Injured",
"Total Victims Killed")

# Join the ad data and violence data on region/state
df_regional_analysis = df_region_grouped.join(df_violence_state,
df_region_grouped.region == df_violence_state.State, how="inner")
df_regional_analysis = df_regional_analysis.select("region",
"total_estimated_spend", "total_estimated_impressions", "Total
Victims", "Total Incidents")

df_regional_analysis.show()

# %%
# Calculate the correlation between ad spend and gun violence
```

```
spend_correlation =
df_regional_analysis.stat.corr("total_estimated_spend", "Total
Incidents")
impression_correlation =
df_regional_analysis.stat.corr("total_estimated_impressions", "Total
Incidents")

print("Correlation between ad spend and gun violence incident: ",
spend_correlation)
print("Correlation between ad impressions and gun violence incident: ",
impression_correlation)

# %%
path = "/user/s4692034/projectData/df_region"
df_region.write.mode('overwrite').json(path)
df_regional_analysis.coalesce(1).write.mode('overwrite').csv("/user/s46
92034/project/regional", header=True)

# %% [markdown]
# ### Ads spend efficiency

# %%
# Add a new column for cost per impression (average_spend /
average_impressions)
df_efficiency = df_gun.withColumn("cost_per_impression",
col("average_spend") / col("average_impressions"))

# Explode the region_distribution array into separate rows
df_efficiency = df_efficiency.select("*",
explode("region_distribution").alias("region_info"))

# Add a column for the region
df_efficiency = df_efficiency.withColumn("region",
df_efficiency["region_info.region"])

# Add a column for the impressions by region (percentage * total
impressions)
df_efficiency = df_efficiency.withColumn("impressions_by_region",
df_efficiency["region_info.percentage"] *
df_efficiency["average_impressions"])

# Add a column for the spend by region (percentage * total spend)
```

```
df_efficiency = df_efficiency.withColumn("spend_by_region",
df_efficiency["region_info.percentage"] *
df_efficiency["average_spend"])

# Group by region and calculate average cost per impression (total
spend by region / total impressions by region)
df_efficiency = df_efficiency.groupBy("region").agg(
    (sum("spend_by_region") /
sum("impressions_by_region")).alias("average_cost_per_impression")
)

# Order by average cost per impression
df_efficiency = df_efficiency.orderBy("average_cost_per_impression",
ascending=False)

df_efficiency.show()
df_efficiency.count()

# %%
# Define a list of valid US states
us_states =
["Alabama","Alaska","Arizona","Arkansas","California","Colorado",
  "Connecticut","Delaware","Florida","Georgia","Hawaii","Idaho","Illinoi
s",
  "Indiana","Iowa","Kansas","Kentucky","Louisiana","Maine","Maryland",
  "Massachusetts","Michigan","Minnesota","Mississippi","Missouri","Monta
na",
  "Nebraska","Nevada","New Hampshire","New Jersey","New Mexico","New
York",
  "North Carolina","North
Dakota","Ohio","Oklahoma","Oregon","Pennsylvania",
  "Rhode Island","South Carolina","South
Dakota","Tennessee","Texas","Utah",
  "Vermont","Virginia","Washington","West
Virginia","Wisconsin","Wyoming"]

# Filter df_efficiency to include only the valid US states
df_efficiency =
df_efficiency.filter(df_efficiency.region.isin(us_states))

df_efficiency.show()
df_efficiency.count()
```

```
# %%
path = "/user/s4692034/projectData/df_efficiency"
df_efficiency.write.mode('overwrite').json(path)
df_efficiency.coalesce(1).write.mode('overwrite').csv("/user/s4692034/p
roject/efficiency", header=True)

# %% [markdown]
# ### Changes in the amount of advertising before and after the
introduction of the bill

# %%
# Adding columns with the dates for three days before and after the
introduced date
df_legislation = df_legislation.withColumn("Start Window",
date_sub(col("Introduced Date"), 3))
df_legislation = df_legislation.withColumn("End Window",
date_add(col("Introduced Date"), 3))

# Joining df_gun and df_legislation on the condition that the
ad_delivery_start_time falls within the start and end window
df_join = df_gun.join(df_legislation,
                     (df_gun["ad_delivery_start_time"] >=
df_legislation["Start Window"]) &
                     (df_gun["ad_delivery_start_time"] <=
df_legislation["End Window"]))

# Group by Name of the legislation and aggregate on the necessary
columns
df_legislation_period = df_join.groupBy(df_legislation["Name"]).agg(
                   count(df_gun["id"]).alias("Number of Ads"),
                   sum(df_gun["average_spend"]).alias("Total Spend"),
                   sum(df_gun["average_impressions"]).alias("Total
Impressions")).sort("Number of Ads", ascending=False)

df_legislation_period.show()

# %%
path = "/user/s4692034/projectData/df_legislation_period"
df_legislation_period.write.mode('overwrite').json(path)
df_legislation_period.coalesce(1).write.mode('overwrite').csv("/user/s4
692034/project/legislation_period", header=True)
```

```
# %%
spark.stop()
```