An Analysis on the Facebook Profiles of Quezon, Manila, and Cebu City PIO

Garing,
Lumawig,
Vallester,
Villafuerte,
Yu

Table of Contents

Context

The group seeks to find correlations between the posts of different LGUs, the length of post VS engagement, time of post VS engagement, and others. These would be done through **data scraping** (using facebook-scraper and Pandas), **sentiment analysis** (using NLTK), **Pearson correlations** between variables (through NumPy), and other statistical and visualization tools.

→

→

Context

In this digital generation, social media has been the primary way for citizens to voice out their opinions and thoughts, especially when it comes to the government. For the purposes of this project, we scraped data from the public information office Facebook pages of 3 LGUs—**2 from Metro Manila and 1 from outside Metro Manila**. We decided to scrape Quezon City and the city of Manila because they are the 2 most populated cities in Metro Manila, while Cebu was selected as a city outside Metro Manila.

→

→

Significance

This will allow the LGUs to **efficiently gather thoughts from citizens** regarding their decisions, mandates, and programs without the need for surveys/interviews.

Because of the restrictions brought about by the pandemic, this can be utilized to collect useful data and improve the systems in place while limiting face-to-face interaction with residents/constituents of the LGU.

→

→

Significance

Below are the corresponding like counts of each LGU PIO.

Quezon City PIO

Quezon City Government
Page · 672K like this

Welcome to the official Facebook account of the Quezon City Government handled by the Public Affairs and Information Services Department.

672k likes

Manila City PIO

Manila Public Information Office
Page · 764K like this

Welcome to the official Facebook page of the new Manila Public Information Office (@ManilaPIO) under Manila City Mayor Francisco "Isko Moreno" Domagoso.

764k likes

Cebu City PIO

Cebu City Public Information Office
Place · Government Organization
Public Information Office, First Floor, Legislative Building, Cebu City Hall, City Hall Lane, Cebu City, Cebu City, Philippines
Opens Tomorrow

213k likes

Cebu City Public Information Office
213K like this · Government Organization

Solution

1. Web Scraping (8 – 9)
2. Format datasets (10 – 12)
3. Average reacts & comments (13 – 15)
4. Correlate length of post & engagement (16 – 18)
5. Correlate hour posted & engagement (19 – 20)
6. Scatter matrix (21)
7. Visualize per city (22 – 23)
8. Compare 3 cities (24)
9. Sentiment Analysis (25 – 29)

Solution

Web Scraping

```
import csv
from facebook_scraper import get_posts

'''change file name according to LGU
itm-final-project-manila.csv
itm-final-project-cebu.csv
'''
with open('itm-final-project-qc.csv','w') as csvfile:
    field_names =['post','time','image','num of likes','num of comments','comments','reacts']
    writer = csv.DictWriter(csvfile, fieldnames=field_names)
    writer.writeheader()

    '''change page ID and cookies file according to LGU
    ManilaPIO, cookies_manila.txt
    CityofCebuOfficial, cookies_cebu.txt
    '''
    try:
        for post in get_posts('QCGov', cookies="cookies_qc.txt", options={"comments":True}, pages=200, extra_info = Tru
            text = post['text']
            time = post['time']
            image = post['image']
            num_likes = post['likes']
            num_comments = post['comments']
            reactions = post['reactions']

            comments = []
            #for posts without comments
            try:
                for i in post['comments_full']:
                    comments.append(i['comment_text'])
            except:
                pass

            writer.writerow({'post': text,'time':time,'image':image,'num of likes':num_likes,'num of comments':
                             num_comments,'comments':comments, 'reacts':reactions})
    except:
        print('Error')
```

- Install facebook-scraper library (which scrapes Facebook pages without an API key) to scrape the posts
- Indicate field names (type of data to be scraped from each FB page)
- Add try/except for possible interruptions while scraping (e.g. internet connection problems)
- Replace unique FB page ID

Solution

Web Scraping

```
import csv
from facebook_scraper import get_posts

'''change file name according to LGU
itm-final-project-manila.csv
itm-final-project-cebu.csv
'''
with open('itm-final-project-qc.csv','w') as csvfile:
    field_names =['post','time','image','num of likes','num of comments','comments','reacts']
    writer = csv.DictWriter(csvfile, fieldnames=field_names)
    writer.writeheader()

    '''change page ID and cookies file according to LGU
    ManilaPIO, cookies_manila.txt
    CityofCebuOfficial, cookies_cebu.txt
    '''
    try:
        for post in get_posts('QCGov', cookies="cookies_qc.txt", options={"comments":True}, pages=200, extra_info = Tru
            text = post['text']
            time = post['time']
            image = post['image']
            num_likes = post['likes']
            num_comments = post['comments']
            reactions = post['reactions']

            comments = []
            #for posts without comments
            try:
                for i in post['comments_full']:
                    comments.append(i['comment_text'])
            except:
                pass

            writer.writerow({'post': text,'time':time,'image':image,'num of likes':num_likes,'num of comments':
                             num_comments,'comments':comments, 'reacts':reactions})
    except:
        print('Error')
```

- Get page cookies and pass True to the 'comments' parameter
- Indicate the number of pages to be scraped
- Pass True to the 'extra_info' parameter to get the number of reacts (haha, wow, like, love, etc.)

Solution

Format datasets

```
def get_total_reacts(reacts_str):
    total_reacts = 0
    if type(reacts_str) == float:
        reacts_str= str(reacts_str)
    # remove commas and {}
    reacts_str = reacts_str.replace(","," ").replace("}", "")
    # use list comprehension to get integers
    reacts_list = [int(word) for word in reacts_str.split() if word.isdigit()]
    # add each element of reacts_list
    for react_amount in reacts_list:
        total_reacts += react_amount
    return(total_reacts)

filename_list = ['./data/itm-final-project-qc.csv','./data/itm-final-project-manila.csv','./data/itm-final-project-cebu

for filename in filename_list:
    df = pd.read_csv(filename)
    df['total_reacts'] = df['reacts'].apply(get_total_reacts)
    df.to_csv(filename, index = False)
```

- Get the # of reactions per emoji and add up to get the total # of reactions per post
- Add another column to the './data/itm-final-project-{LGU}.csv' files (for total reactions per post)

Solution

Format datasets

```
filename_list = ['./data/itm-final-project-qc.csv','./data/itm-final-project-manila.csv','./data/itm-final-project-cebu
keywords = 'coronavirus|covid|covid-19|delta|variant|symptomatic|asymptomatic|social distancing|quarantine|lockdown|gcq
def get_topic(post):
    try:
        re.search(keywords,post).group()
        return 'covid'
    except:
        return 'noncovid'

for filename in filename_list:
    df = pd.read_csv(filename)
    df['post'].str.lower()
    df['topic'] = df['post'].apply(get_topic)
    df.to_csv(filename,index=False)
```

- Indicate the keywords that can indicate if the post is COVID-related or not
- Use Regular Expressions (RegEx) to check if keywords are present in each post
  - if it returns the word, it will add 'covid' to the specific column
  - if it returns None (an error), it will run the code under except (return 'noncovid')

Solution

Format datasets

```
data_qc = pd.read_csv('./data/itm-final-project-qc.csv')
data_manila = pd.read_csv('./data/itm-final-project-manila.csv')
data_cebu = pd.read_csv('./data/itm-final-project-cebu.csv')

master_data = data_qc.append(data_manila.append(data_cebu))
master_data.reset_index(drop = True, inplace = True)

master_data.to_csv('./data/master-data.csv')
```

- Combine into one csv file

Solution

Average reacts and comments

```
# Average reacts and comments (COVID vs Non-COVID)

avestats = pd.DataFrame(index = ['cebu','manila','qc'], columns = ['reacts_covid','comments_covid','reacts_noncovid','c
avestats['reacts_covid'] = 0
avestats['comments_covid'] = 0
avestats['reacts_noncovid'] = 0
avestats['comments_noncovid'] = 0
cebu_covid_count = 0
cebu_noncovid_count = 0
manila_covid_count = 0
manila_noncovid_count = 0
qc_covid_count = 0
qc_noncovid_count = 0

# CEBU

cebu_read = cebu_read.astype({"total_reacts": int, "num of comments": int})

for i in range(len(cebu_read['topic'])):
    if cebu_read.topic[i] == "covid":
        avestats.reacts_covid["cebu"] += cebu_read.total_reacts[i]
        avestats.comments_covid["cebu"] += cebu_read["num of comments"][i]

        cebu_covid_count += 1

    if cebu_read.topic[i] == "noncovid":
        avestats.reacts_noncovid["cebu"] += cebu_read.total_reacts[i]
        avestats.comments_noncovid["cebu"] += cebu_read["num of comments"][i]

        cebu_noncovid_count += 1

avestats.reacts_covid["cebu"] = avestats.reacts_covid["cebu"]/cebu_covid_count
avestats.comments_covid["cebu"] = avestats.comments_covid["cebu"]/cebu_covid_count
avestats.reacts_noncovid["cebu"] = avestats.reacts_noncovid["cebu"]/cebu_noncovid_count
```

- Process average # of reacts and comments for COVID-related posts and non COVID-related posts in each LGU

Solution

Average reacts and comments

```
avestats.comments_noncovid["cebu"] = avestats.comments_noncovid["cebu"]/cebu_noncovid_count

# MANILA

manila_read = manila_read.astype({"total_reacts": int, "num of comments": int})

for i in range(len(manila_read['topic'])):
    if manila_read.topic[i] == "covid":
        avestats.reacts_covid["manila"] += manila_read.total_reacts[i]
        avestats.comments_covid["manila"] += manila_read["num of comments"][i]

        manila_covid_count += 1

    if manila_read.topic[i] == "noncovid":
        avestats.reacts_noncovid["manila"] += manila_read.total_reacts[i]
        avestats.comments_noncovid["manila"] += manila_read["num of comments"][i]

        manila_noncovid_count += 1

avestats.reacts_covid["manila"] = avestats.reacts_covid["manila"]/manila_covid_count
avestats.comments_covid["manila"] = avestats.comments_covid["manila"]/manila_covid_count
avestats.reacts_noncovid["manila"] = avestats.reacts_noncovid["manila"]/manila_noncovid_count
avestats.comments_noncovid["manila"] = avestats.comments_noncovid["manila"]/manila_noncovid_count

# QC

qc_read = qc_read.astype({"total_reacts": int, "num of comments": int})

for i in range(len(qc_read['topic'])):
    if qc_read.topic[i] == "covid":
        avestats.reacts_covid["qc"] += qc_read.total_reacts[i]
        avestats.comments_covid["qc"] += qc_read["num of comments"][i]

        qc_covid_count += 1
```

- Process average # of reacts and comments for COVID-related posts and non COVID-related posts in each LGU

Solution

Average reacts and comments

```
    if qc_read.topic[i] == "noncovid":
        avestats.reacts_noncovid["qc"] += qc_read.total_reacts[i]
        avestats.comments_noncovid["qc"] += qc_read["num of comments"][i]

        qc_noncovid_count += 1

avestats.reacts_covid["qc"] = avestats.reacts_covid["qc"]/qc_covid_count
avestats.comments_covid["qc"] = avestats.comments_covid["qc"]/qc_covid_count
avestats.reacts_noncovid["qc"] = avestats.reacts_noncovid["qc"]/qc_noncovid_count
avestats.comments_noncovid["qc"] = avestats.comments_noncovid["qc"]/qc_noncovid_count

# FILE
avestats.to_csv("./data/itm-average-stats.csv")
```

	reacts_covid	comments_covid	reacts_noncovid	comments_noncovid
cebu	461	191	208	42
manila	1516	453	801	127
qc	992	216	870	276

Solution

Correlate length of post and engagement

```
# Add column for post's word count
all_len = []

for k in master_data['post'].astype(str):
    all_len.append(len(k.split()))

master_data['post_word_count'] = all_len
```

```
# categorize word count and make new column for category
def categorize_word_count(word_count):
    if word_count <= 30:
        return "0-30"
    elif word_count <= 60:
        return "31-60"
    elif word_count <= 90:
        return "61-90"
    elif word_count <= 120:
        return "91-120"
    elif word_count <= 150:
        return "121-150"
    elif word_count <= 180:
        return "151-180"
    else:
        return "180+"

master_data['word count category'] = master_data['post_word_count'].apply(categorize_word_count)
master_data['count'] = 1
master_data
```

- Create a new column for word count
- Categorize word count and create a new column to indicate the category

Solution

Correlate length of post and engagement

```
# create bar graph based on category and number of likes

# format and create a new dataframe for bar graph
category_likes_df = master_data[['word count category', 'total_reacts','count']].copy()
bar_df = category_likes_df.groupby(['word count category']).sum()
bar_df.reset_index(inplace = True)

def get_word_count_start(word_count):
    try:
        j = int(word_count.split('-')[0])
        return(j)
    except:
        return(int(word_count[:len(word_count)-1]))

#def get_average(total_reacts):
bar_df['ave_reacts'] = bar_df['total_reacts']/bar_df['count']
bar_df['word count start'] = bar_df['word count category'].apply(get_word_count_start)

bar_df.sort_values('word count start', ascending = True, inplace = True)
bar_df.drop(columns = ['word count start'])
```

Solution

Correlate length of post and engagement

```
# for bar graph
fig, ax = plt.subplots(figsize = (12,7))

ax.bar(height=bar_df['ave_reacts'],x=bar_df['word count category'], color = 'BurlyWood')
ax.set_xlabel('Word Count Category')
ax.set_ylabel('Average Reacts per Post')
ax.set_title('Average Reactions for Each Word Count Category')
```

```
# create scatter plot for total reacts VS. word count
fig, ax = plt.subplots(figsize = (12,7))

ax.scatter(master_data['post_word_count'], master_data['total_reacts'], color = 'BurlyWood')
ax.set_xlabel('Word Count')
ax.set_ylabel('Total Reacts')
ax.set_title('Total Reacts VS Word Count')
```

```
# Pearson's Correlation Test - word count vs no. of reactions
cor_df = master_data[['post_word_count','total_reacts']].dropna()
cor_df.corr()
```

Solution

Correlate hour posted and engagement

```
filename = "./data/itm-final-project-qc.csv"
df = pd.read_csv(filename)

df['hour'] = pd.to_datetime(df['time']).dt.hour
count = df.groupby(['hour']).count()
grouped_df = df.groupby(['hour'])['total_reacts'].sum()
grouped_df = grouped_df.to_frame()
grouped_df['count'] = count['post']
grouped_df['average_reacts'] = grouped_df['total_reacts']/grouped_df['count']
grouped_df.reset_index(inplace=True)
# for bar graph
fig, ax = plt.subplots(figsize = (12,7))

ax.bar(height=grouped_df['average_reacts'] , x=grouped_df['hour'], color = 'DarkBlue')
ax.set_xticks(list(grouped_df['hour']))
ax.set_xlabel('Hour (24H Format)')
ax.set_ylabel('Average Reacts')
ax.set_title('Average Reacts per Hour in Quezon City')
plt.show()
```

- Get the total number of reacts and # of posts per time period to calculate the average reacts

Solution

Correlate hour posted and engagement

```
# create plot for average reacts VS. posting time
fig, ax = plt.subplots(figsize = (12,7))

ax.plot(grouped_df['hour'], grouped_df['average_reacts'])
ax.set_xlabel('Hour')
ax.set_ylabel('Average Reacts')
ax.set_title('Average Reacts VS Hour in Quezon City')
```

```
# Pearson's Correlation Test - posting time vs total reactions - QC
grouped_df[['hour','average_reacts']].corr()
```

Solution

Scatter matrix

```
city_comp = pd.concat([qc_df['total_reacts'],mnl_df['total_reacts'],cebu_df['total_reacts']], axis = 1)
city_comp.columns = ['QC Reacts','Manila Reacts','Cebu Reacts']

scatter_matrix(city_comp, figsize=(12,12), hist_kwds={'bins':25})
```

- Check if there are any correlations between each city's respective total number of reacts

Solution

Visualize (per city)

```
qc_df = pd.read_csv("./data/itm-final-project-qc.csv")
mnl_df = pd.read_csv("./data/itm-final-project-manila.csv")
cebu_df = pd.read_csv("./data/itm-final-project-cebu.csv")

# create new dataframes for just time and total_reacts columns
qc_time_df = qc_df[['time','total_reacts']].copy()
qc_time_df.sort_values('time', inplace = True)
qc_time_df.set_index('time', drop=False, inplace=True)

mnl_time_df = mnl_df[['time','total_reacts']].copy()
mnl_time_df.sort_values('time', inplace = True)
mnl_time_df.set_index('time', drop=False,inplace=True)

cebu_time_df = cebu_df[['time','total_reacts']].copy()
cebu_time_df.sort_values('time', inplace = True)
cebu_time_df.set_index('time', drop=False, inplace=True)

# make column for date one (no time)
def get_date(time):
    return(time[0:10])

qc_time_df['date'] = qc_time_df['time'].apply(get_date)
mnl_time_df['date'] = mnl_time_df['time'].apply(get_date)
cebu_time_df['date'] = cebu_time_df['time'].apply(get_date)
```

- Create dataframes for time and total reacts of each city/LGU (through pandas)
- Create a column for date (not including time)

Solution

Visualize (per city)

```
# QC
fig, ax = plt.subplots(figsize = (12,7))

ax.scatter(qc_time_df['date'],qc_time_df['total_reacts'], color = "DarkBlue")
ax.set_xlabel('Date')
ax.set_ylabel('Total Reacts')
ax.set_title('Quezon City Facebook Page: Total Reacts VS Time')
ax.tick_params(axis='x', rotation=45)
```

Solution

Compare 3 cities

```
# make new col if date it < 2021-07-24 (True or False)
def check_25(date):
    return not(int(date.replace("-","")) >= 20210725)

qc_time_df['before 07-25'] = qc_time_df['date'].apply(check_25)
mnl_time_df['before 07-25'] = mnl_time_df['date'].apply(check_25)
cebu_time_df['before 07-25'] = cebu_time_df['date'].apply(check_25)

# make copy dataframes for only False
qc_after_07_25_df = qc_time_df.loc[qc_time_df.loc[:, 'before 07-25']== False,:].copy()
mnl_after_07_25_df = mnl_time_df.loc[mnl_time_df.loc[:, 'before 07-25']== False,:].copy()
cebu_after_07_25_df = cebu_time_df.loc[cebu_time_df.loc[:, 'before 07-25']== False,:].copy()
```

```
fig, axes = plt.subplots(figsize = (12,7))

axes.scatter(qc_after_07_25_df['date'],qc_after_07_25_df['total_reacts'], color = 'DarkBlue', label = 'Quezon City')
axes.scatter(mnl_after_07_25_df['date'],mnl_after_07_25_df['total_reacts'], color = 'DarkGreen', label = 'Manila City')
axes.scatter(cebu_after_07_25_df['date'],cebu_after_07_25_df['total_reacts'], color = 'DarkRed', label = 'Cebu City')
axes.set_xlabel('Date')
axes.set_ylabel('Total Reacts')
axes.set_title('Quezon City, Manila City, Cebu City: React per Post Over Time')
axes.legend()
axes.tick_params(axis='x', rotation=45)
```

- compare from July 25 to August 6 since these are the dates present in the collected LGU data

Solution

Sentiment Analysis

```
import googletrans
from googletrans import Translator

# make csv for just posts
qc_posts = data_qc[['post']].copy()
# translate to english
def translate(post):
    translator = Translator()
    result = translator.translate(str(post), scr='tl', dest=
    return(result.text)

qc_posts['post'] = qc_posts['post'].apply(translate)
qc_posts.to_csv('./data/qc-posts-english.csv')
```

- translate post text to English (through Google Translate) because Tagalog/Bisaya is not supported by the sentiment analyzer

Solution

Sentiment Analysis

```
import time
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import nltk
import io
import unicodedata
import numpy as np
import re
import string

from numpy import linalg
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from nltk.tokenize import sent_tokenize, word_tokenize
from nltk.tokenize import PunktSentenceTokenizer
from nltk.tokenize import PunktSentenceTokenizer
from nltk.corpus import webtext
from nltk.stem.porter import PorterStemmer
from nltk.stem.wordnet import WordNetLemmatizer

with open('./data/qc-posts-english.csv', encoding ='ISO-8859-2') as f:
    text = f.read()
```

- Import modules/libraries needed for sentiment analysis

Solution

Sentiment Analysis

```
sent_tokenizer = PunktSentenceTokenizer(text)
sents = sent_tokenizer.tokenize(text)

porter_stemmer = PorterStemmer()

nltk_tokens = nltk.word_tokenize(text)

wordnet_lemmatizer = WordNetLemmatizer()
nltk_tokens = nltk.word_tokenize(text)

text = nltk.word_tokenize(text)

sia = SentimentIntensityAnalyzer()
qc_list_scores = []
with open('./data/qc-posts-english.csv',encoding='utf-8') as f:

    for text in f.read().split('\n'):
        scores = sia.polarity_scores(text)
        qc_list_scores.append(scores)
```

- Taken from Geeks for geeks
  (slightly edited)

Solution

Sentiment Analysis

```
qc_scores_df = pd.DataFrame(qc_list_scores)

# 0 and 1 refer to newline and other irrelevant expressions
qc_scores_df= qc_scores_df[qc_scores_df['neu'] != float(0)]
qc_scores_df= qc_scores_df[qc_scores_df['neu'] != float(1)]
qc_scores_df.reset_index(inplace = True)

# show in histogram
qc_scores_df['compound'].hist(bins=25, color= 'Darkblue', figsize = (13,7))
plt.xlabel('Compound Value')
plt.ylabel('Amount')
plt.title('Histogram of the Sentiment Analysis Compound Value - Quezon City')
```

- Take 'Compound Value' from the dictionary output of the analyzer

```
qc_scores_df['compound'].hist(bins=25, label='Quezon City', color='DarkBlue', alpha = 0.5, figsize=(13,6))
mnl_scores_df['compound'].hist(bins=25, label='Manila City', color='DarkGreen', alpha = 0.5)
cebu_scores_df['compound'].hist(bins=25, label='Cebu City', color = 'DarkRed',alpha = 0.5)
plt.legend()
plt.xlabel('Compound Value')
plt.ylabel('Amount')
plt.title('Histogram of the Sentiment Analysis Compound Value per City')
```

- Make histogram based on that value per city

Solution

Sentiment Analysis

```
# To scale accordingly (since Manila and Quezon City have a bigger reach)
# to visualize the Probability Density
qc_scores_df['compound'].plot(kind='kde',label='Quezon City', color = 'DarkBlue',figsize=(13,6))
mnl_scores_df['compound'].plot(kind='kde', color = 'DarkGreen', label='Manila City')
cebu_scores_df['compound'].plot(kind='kde', color = 'DarkRed', label='Cebu City')
plt.legend()
plt.title('Kernel Density Estimation (KDE) of Sentiment Analysis Compound Value per City')
```

- Use Kernel Density Estimation (KDE) to estimate the probability density function (curve)

Insights

Insights

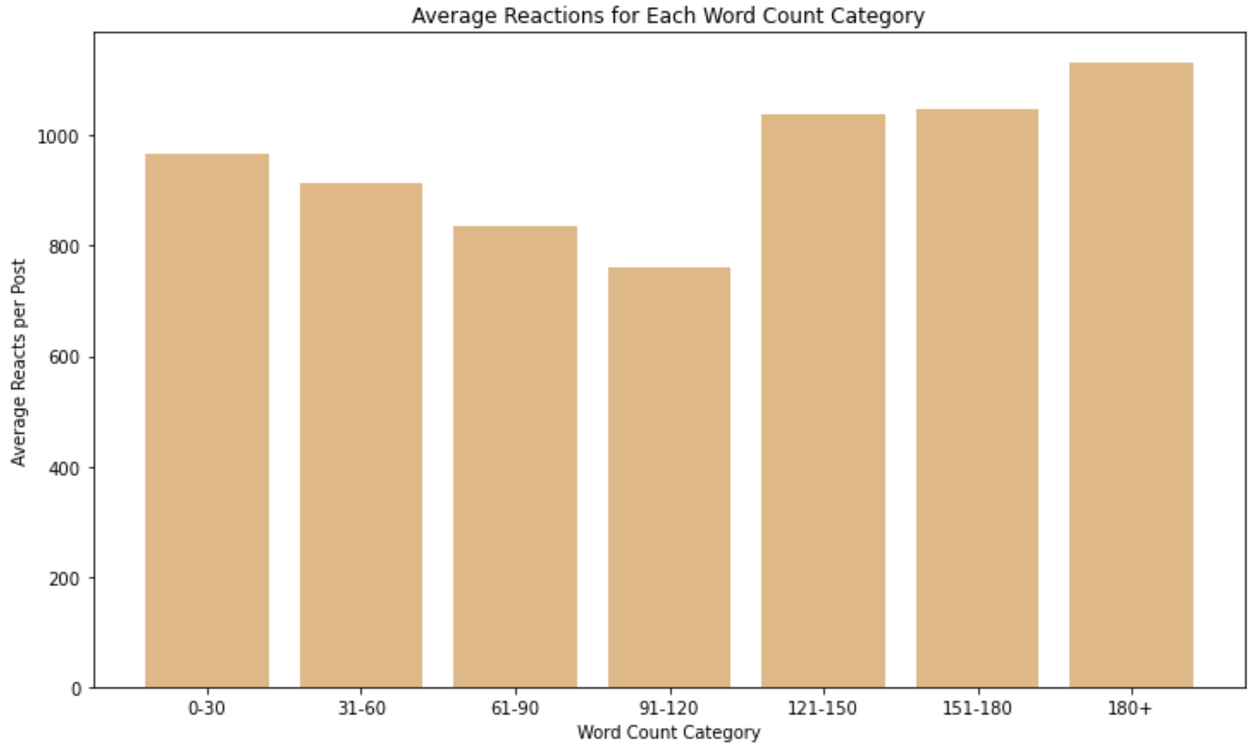
Average Reactions for Each Word Count Category

"The shorter the better." To find out if the length of the post has an effect on the number of reactions it gets, a simple **bar graph** was initially used.

As seen in the figure, the shortest post length does indeed do well with an **average of close to 1000 reactions**, and the longer the post gets the less engagement it receives.

However, there comes a **turning point at the 121-150 words category** where engagement suddenly **rises by 36%** and continues this trend as the word count increases.

With this unexpected result, a correlation test was done.



	word count category	total_reacts	count	ave_reacts
0	0-30	79181	82	965.621951
4	31-60	108694	119	913.394958
5	61-90	95188	114	834.982456
6	91-120	61733	81	762.135802
1	121-150	47737	46	1037.760870
2	151-180	35582	34	1046.529412
3	180+	81417	72	1130.791667

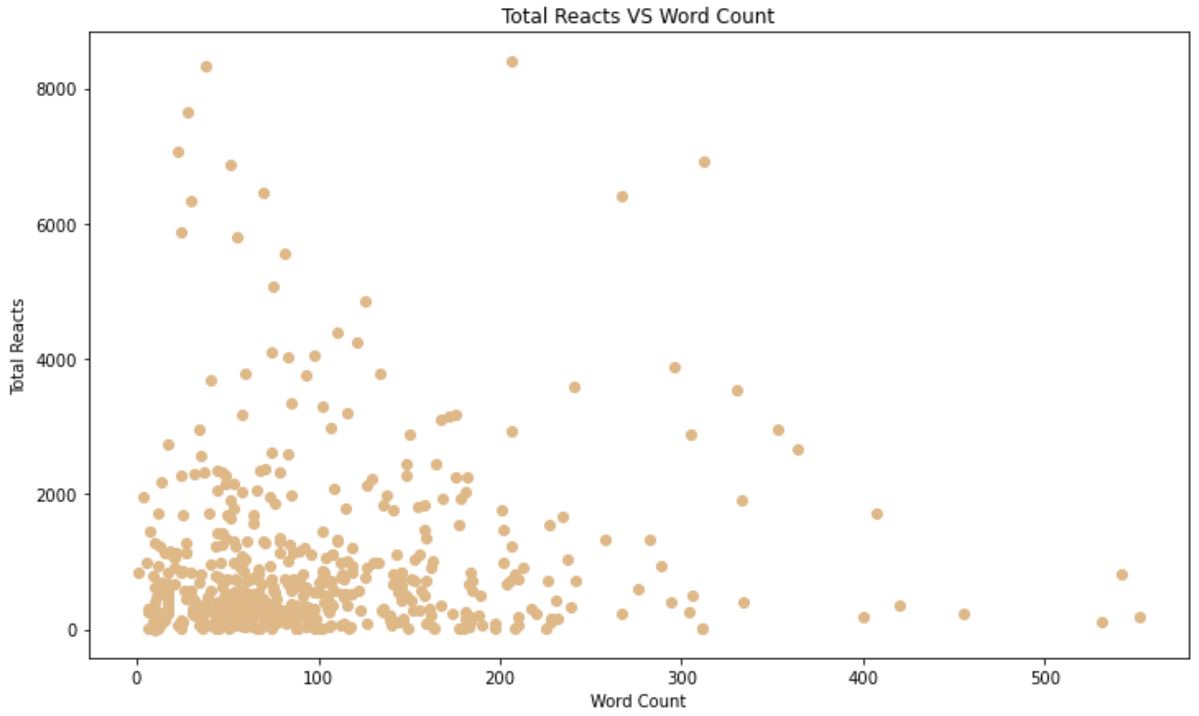
Insights

Total Reacts VS Word Count

The Pearson's R Correlation Test was used. For the relationship between the total number of reacts of a post and its word count, a correlation coefficient of **0.066877** was outputted. With the number being very close to zero, it can be said that the **length of the post does not directly correlate with the reactions it receives.**

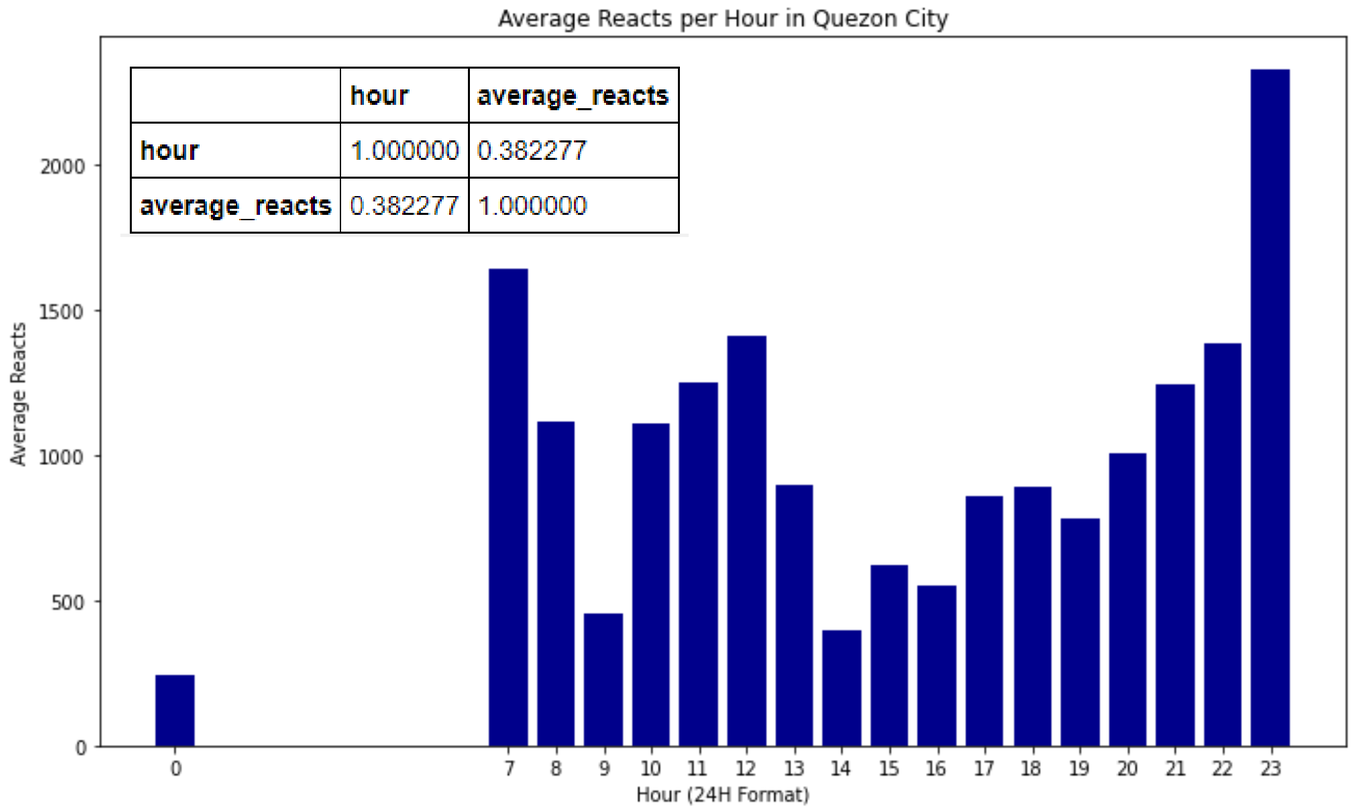
To visualize this, a **scatter plot** seen to the right was used. Nowhere does the plot resemble a line of any sort. This may be due to the fact that the longer posts may contain important facts. Contrary to the belief "the shorter post, the better", LGUs may not need to shorten their posts, especially if it contains valuable information.

	post_word_count	total_reacts
post_word_count	1.000000	0.066877
total_reacts	0.066877	1.000000



Insights

Average Reactions per Hour in Quezon City



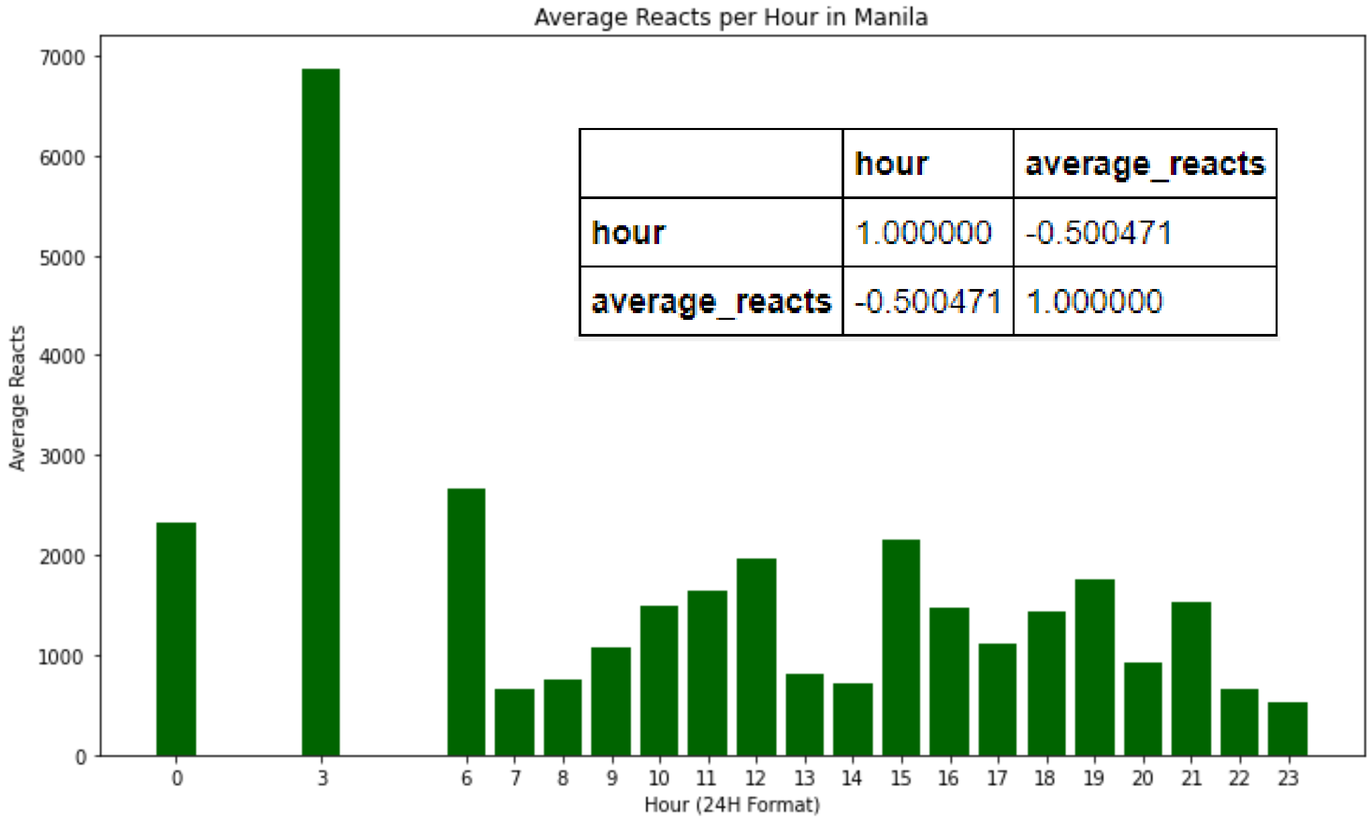
	hour	average_reacts
hour	1.000000	0.382277
average_reacts	0.382277	1.000000

It may be assumed that the best time to post is at night since this is the time when the Filipinos are resting and are online. However, it can be seen from the bar graph of Quezon City that while the engagement is high from 8-11PM, the posts in the morning get as much attention. The lowest average reactions are those posted at 12MN-1AM, 9-10AM, and 2-3PM.

Since the Pearson coefficient is low (below 0.4), it can be concluded that in Quezon City, there is a **weak correlation** between the time an announcement is posted and its engagement in terms of the average post reactions.

Insights

Average Reactions per Hour in Manila City



	hour	average_reacts
hour	1.000000	-0.500471
average_reacts	-0.500471	1.000000

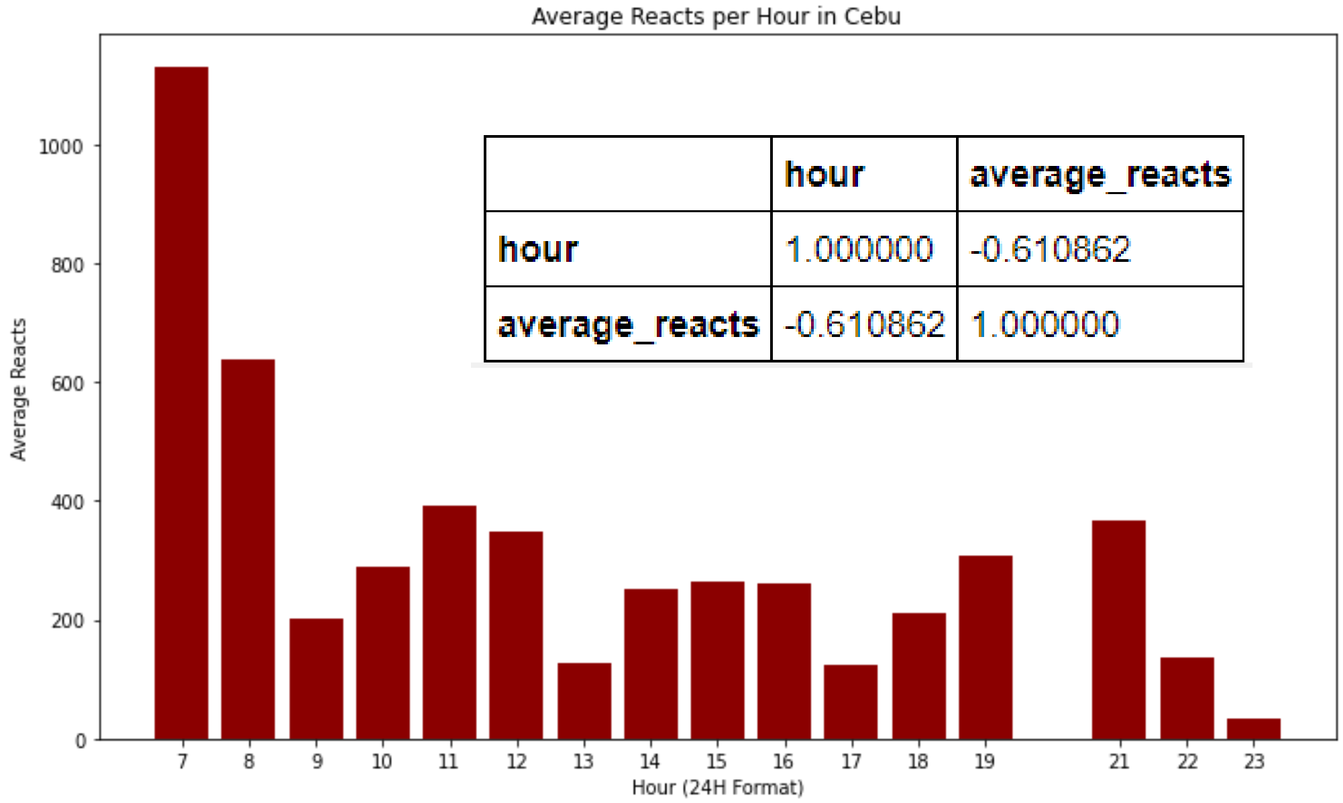
The posts from Manila PIO generally have the same average reactions (around 1000-2000 reactions). However, at 3-4AM, the average number of reactions reached almost 7000. This is due to a single post at around 3AM about the commotion regarding Manila's vaccination program.

Aside from this, the next 3 with the highest engagements are those posted at 12MN-1AM, 6-7AM, and 3-4PM, while the lowest engagements were posted at 2-3PM, 10PM-12MN.

The Pearson coefficient suggests that in Manila, the average reactions per time period and the time posted have **moderate correlation.**

Insights

Average Reactions per Hour in Cebu City



	hour	average_reacts
hour	1.000000	-0.610862
average_reacts	-0.610862	1.000000

Cebu's bar graph shows that the highest engagement occurs at 7-8AM. Generally, Cebu's PIO has an average of 300 reactions per time period, but the posts that gained the lowest attention were posted at 1-2PM, 5-6PM, and 10-12MN.

There is a **moderate correlation** between Cebu's average reactions and the time posted. The LGU may refer to this bar graph when they will post important announcements.

Insights

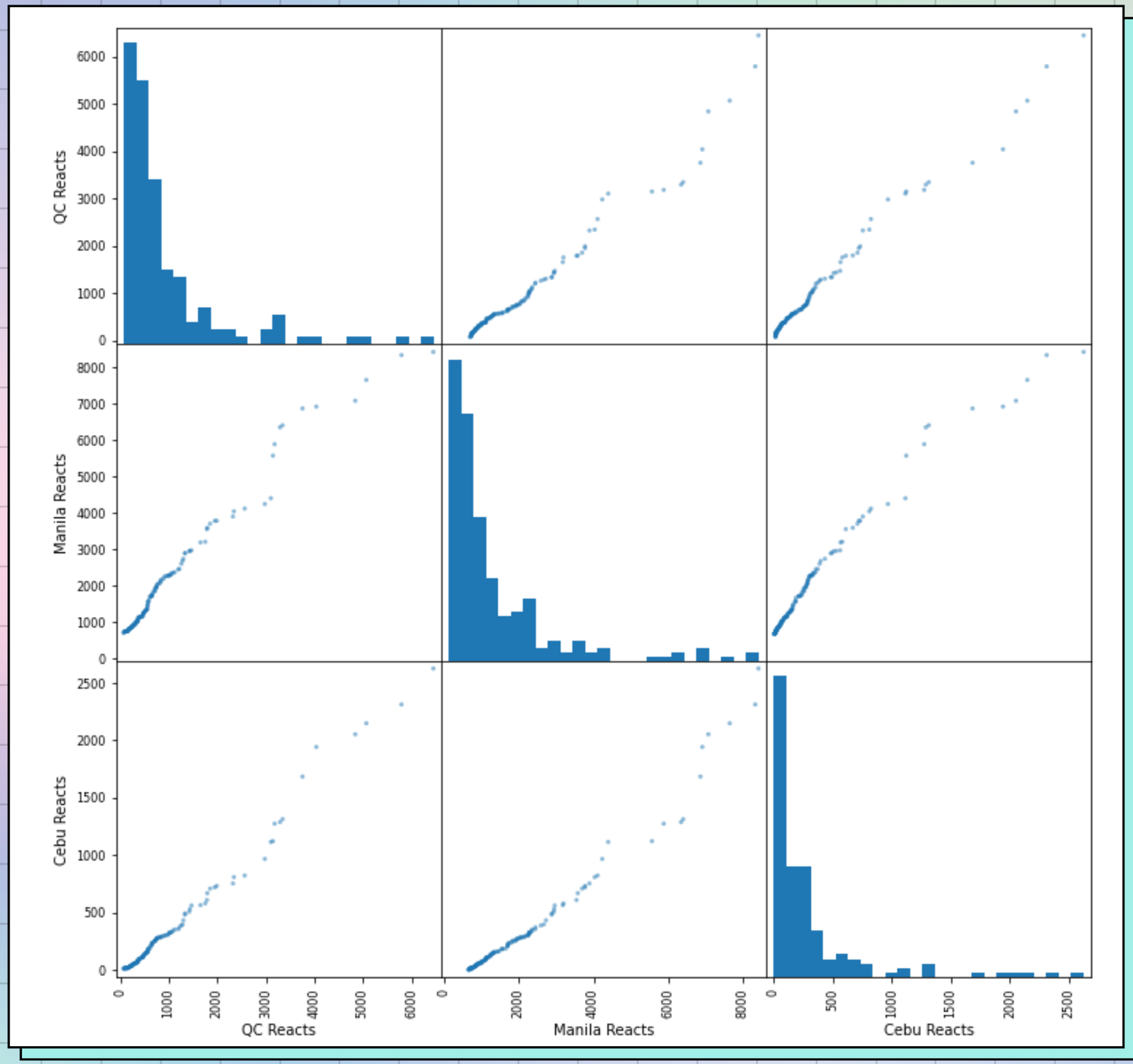
Scatter Matrix for Correlation between LGUs

To see the relationship between the engagement (number of reactions) of different cities' posts, a scatter matrix was used.

As seen from the figure, all the intersections between different cities have a scatter plot that exhibits the **shape of a line with a positive slope.**

This suggests that engagement between cities is related. Although one may not directly affect the other, they are **likely bounded by the same content and context.**

Even the city outside Metro Manila performed similarly to that of the cities in the capital.

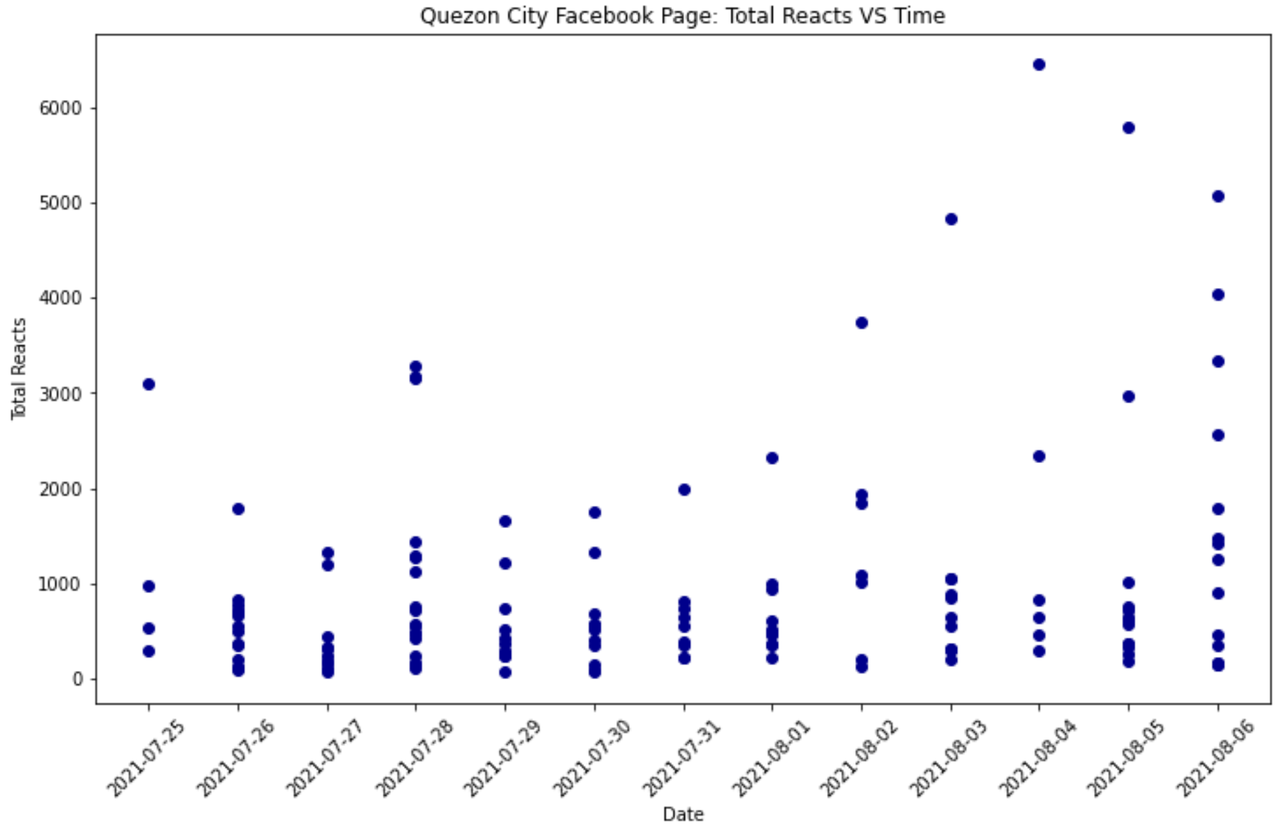


Insights

Quezon City Facebook Page Total Reactions VS Date

To further analyze the engagement performance of each LGU, scatter plots were used, but on this occasion, each post was plotted according to the **date of when it was posted.**

For Quezon City, we see that the post with the most engagement has **increased** in reactions from the start of **August**. Moreover, on 08-06, there were **4 posts** that stood out and had **over 2500 reactions.**



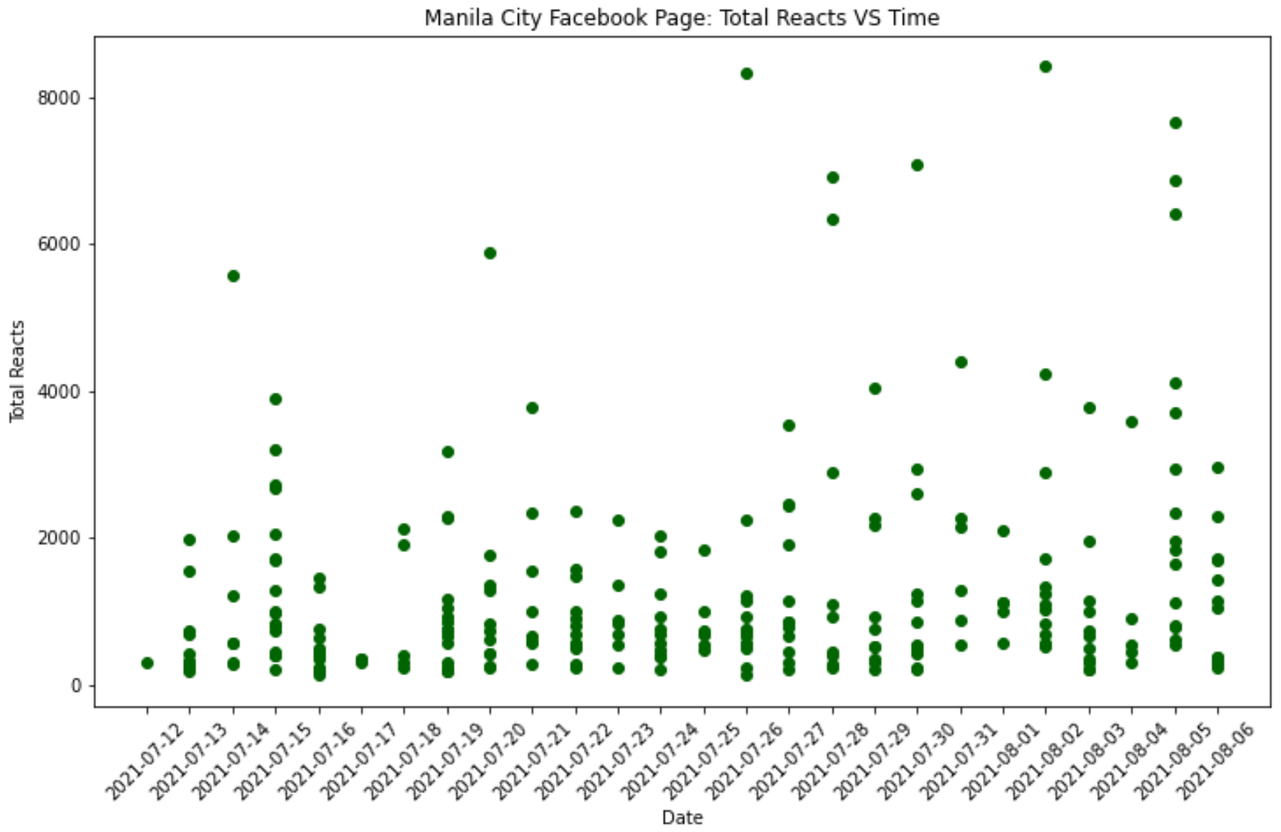
Insights

Manila City Facebook Page Total Reactions VS Date

With Manila's following base to be larger than Quezon City by 90,000, there is no surprise that their posts have **more reactions.**

However, similar to the case of Quezon City, there was a spike of **5 cases that stood out, each with 3000+ reactions**, but this time on 08-05.

More on this spike will be discussed in the following slides.



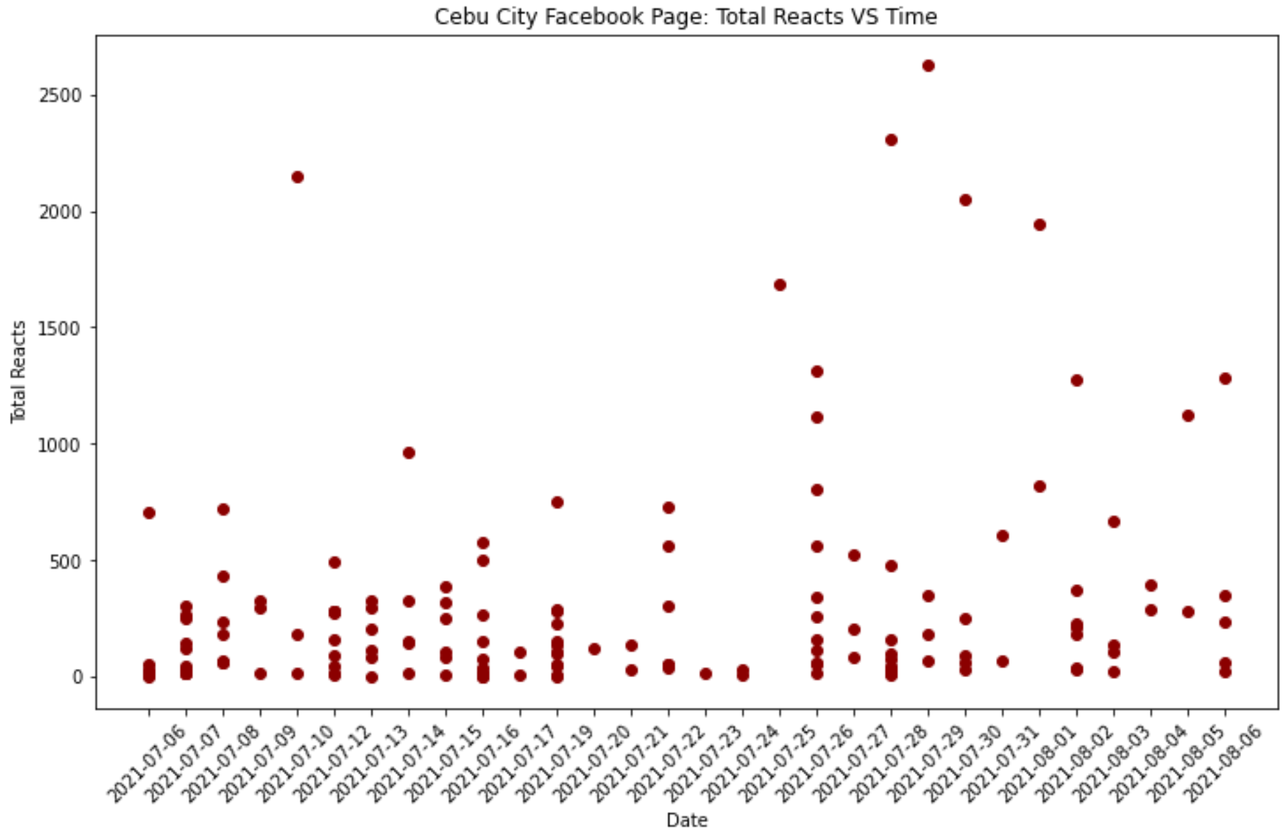
Insights

Cebu City Facebook Page Total Reactions VS Date

With the smallest following base among the two other LGUs, Cebu's Facebook Page shows still **similar engagement patterns** with Quezon City and Manila.

Most of the posts have less than 500 reactions with the occasional few reaching over **2000.**

To see the big picture of the patterns of the different Facebook Pages, it would be beneficial to combine the graphs into one



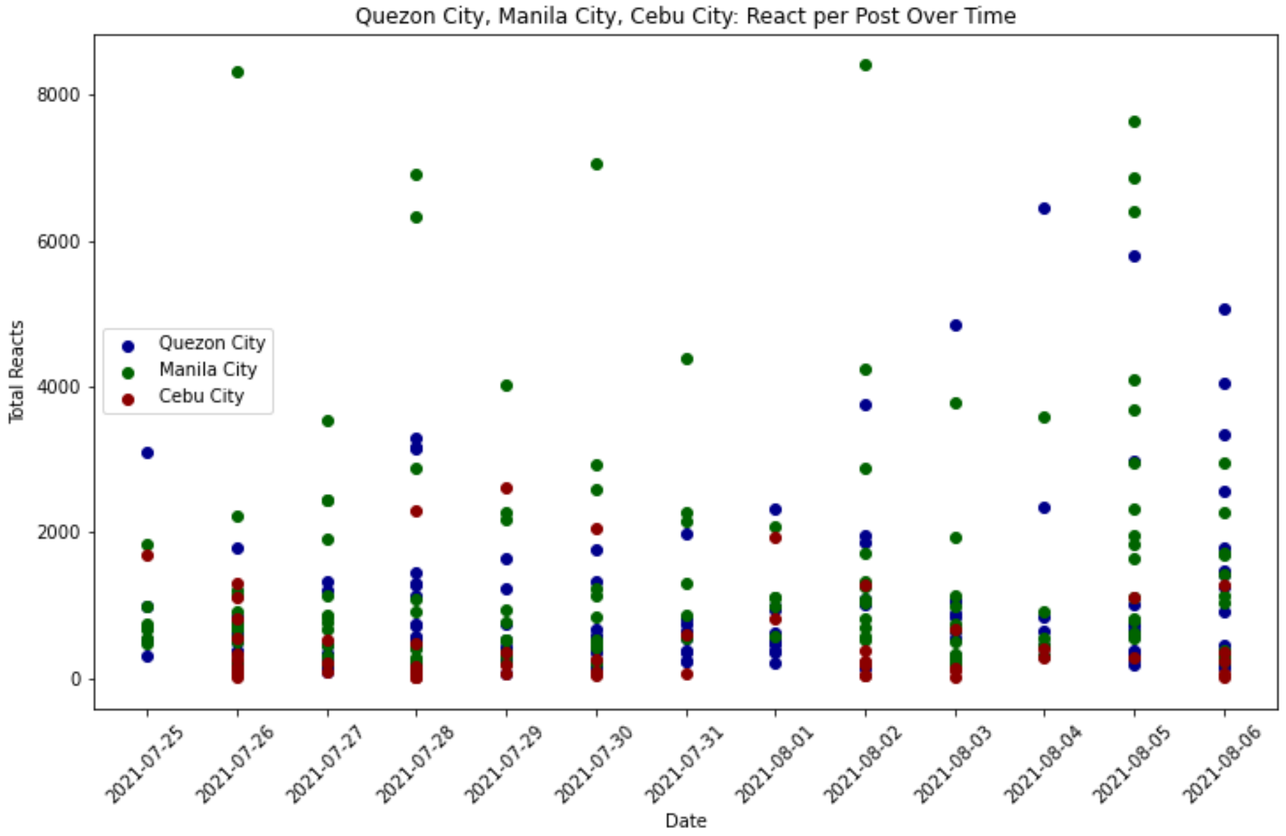
Insights

Quezon City, Manila City, Cebu City Reactions per Post Over Date

To have a clearer graph, the date of all 3 LGU's Facebook engagement data was narrowed down to **07-25 to 08-06** as these are the dates available to all LGU's data.

Again, the data during **August 5** stands out as the number of posts with above-average reactions from that day is much higher compared to that of other days.

It would be best to examine these posts with significantly higher engagement to understand why this took place.



Insights

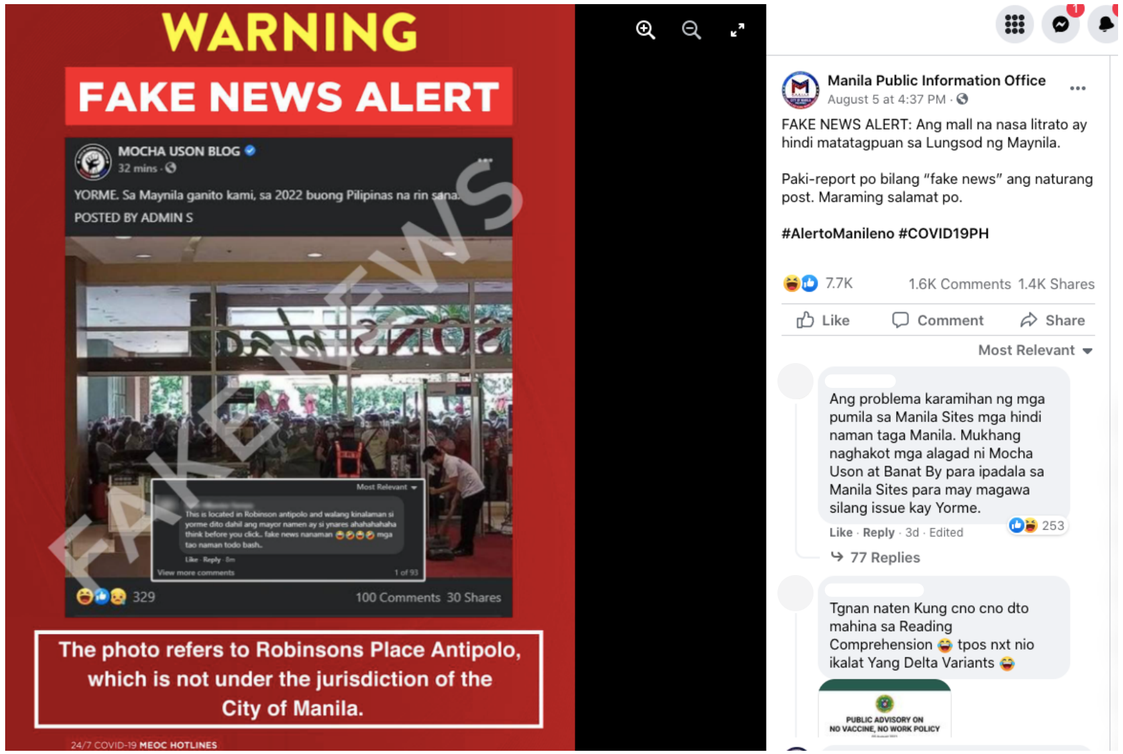
Quezon City, Manila City, Cebu City
Reactions per Post Over Date

Analysis of Post with High Engagement

The photo is a screenshot taken of the top performing post on **Manila PIO**'s page. Looking into the other top posts on August 5, most, if not all, seemed to be related to the spread of **fake news** and **misinformation** regarding its vaccination program**.**

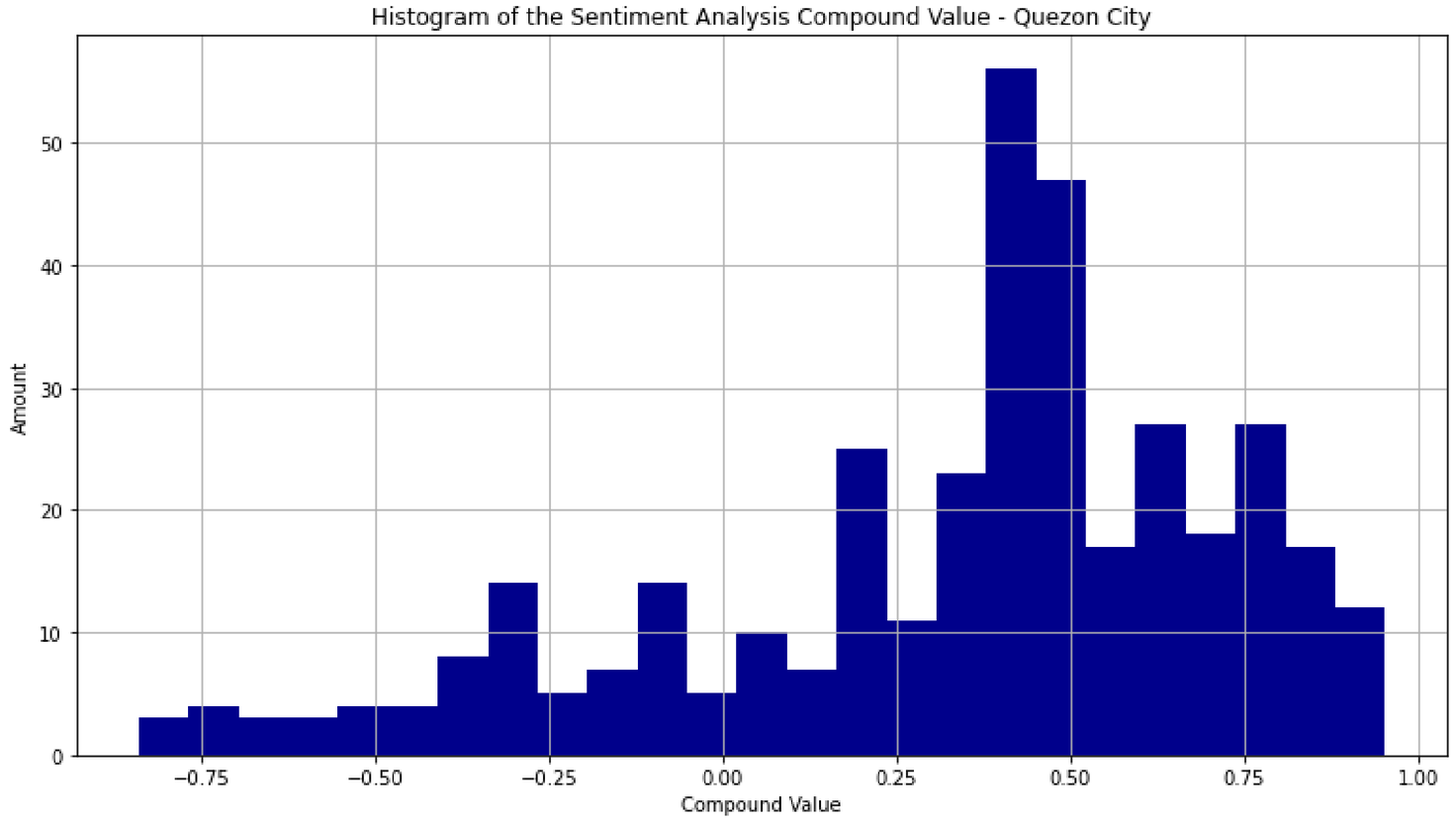
This one in particular is a warning against a post by Mocha Uson bearing a photo of people supposedly overcrowding a vaccination facility against COVID-19 that Uson alleges to be located in Manila. The picture is actually an image of a mall in Antipolo.

Coincidence or not, with a rising number of these kinds of posts circulating in the media these days, **we may be seeing more posts like this in the near future**.



Insights

Histogram of the Sentiment Analysis Compound Value – Quezon City



The Compound Value of the Sentiment Analysis is a number between -1 and 1 that **quantifies if a text is positive or negative.**

The figure on the left, for the compound value of QC, is **skewed to the left**. This suggests that most of the post description of Quezon City is **positive**.

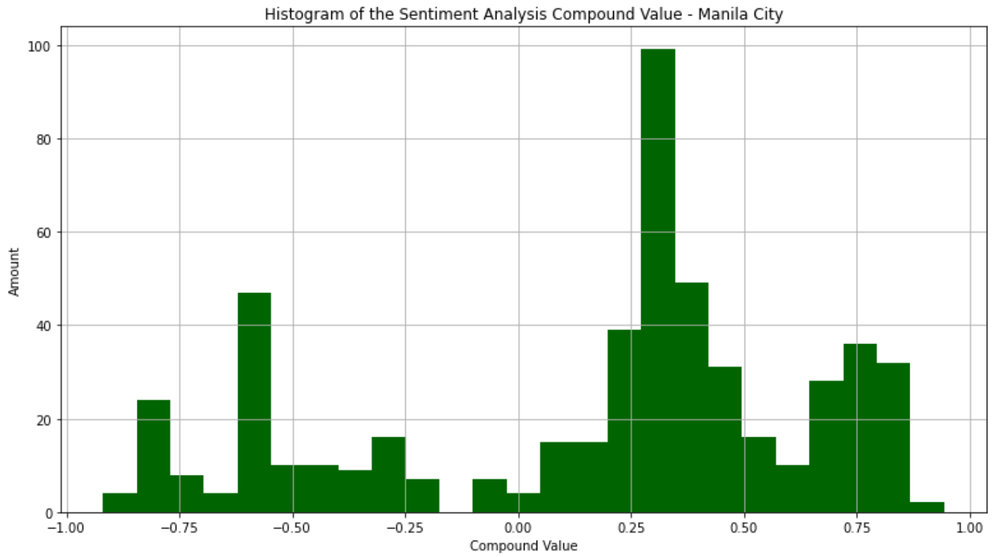
Newberry, C. (2020) says that this skewness could imply that the **brand messaging of the organization, QC LGU, is doing great!**

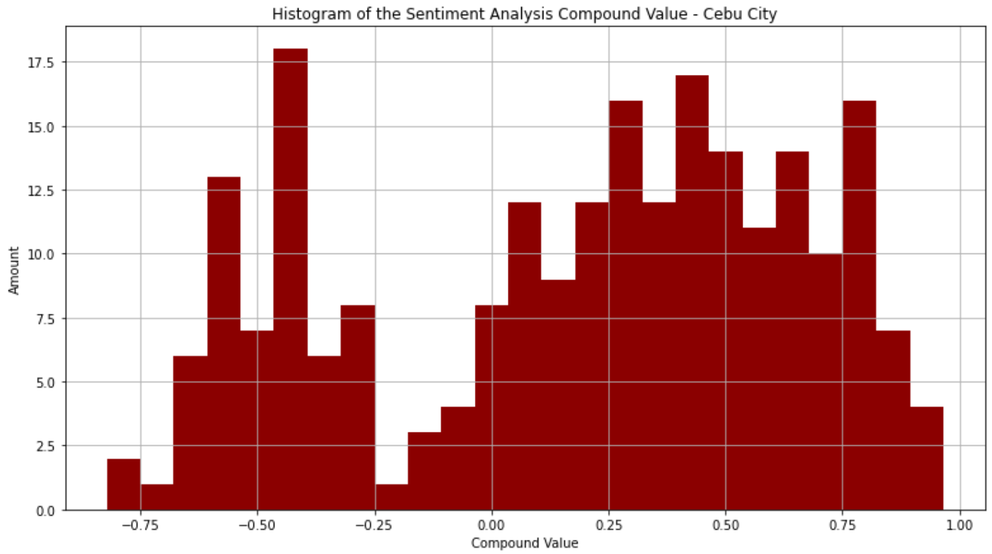
Insights

Histogram of the Sentiment Analysis Compound Value – Manila and Cebu City

For both Manila and Cebu, although most of their posts are positive, a significant amount of them is also negative, thus the **two "hill-like" curves** seen in the graphs.

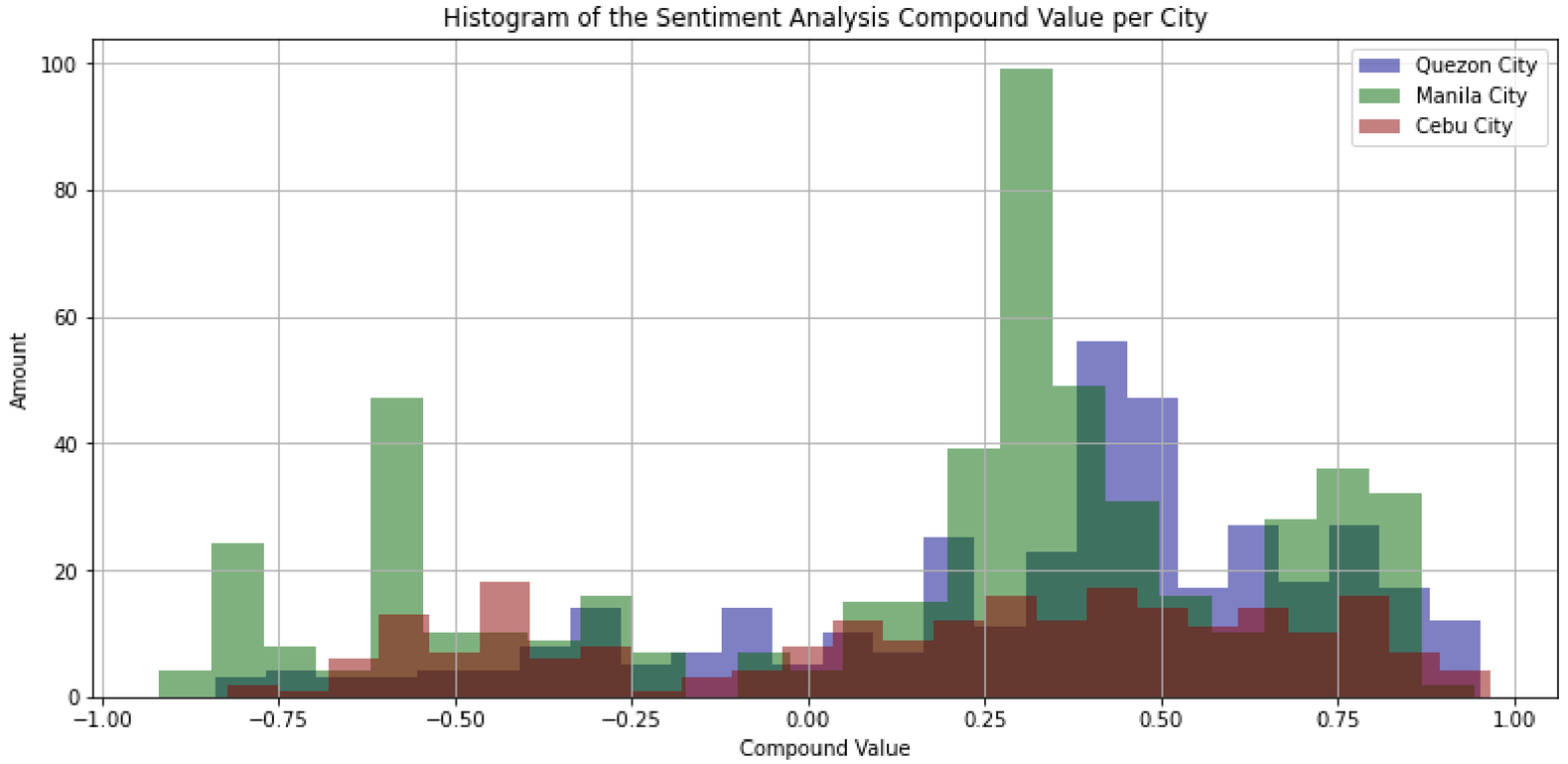
Although having posts with a negative compound value is not necessarily bad, it does **have an effect on how social media users may perceive the organization** (Botchway et al., 2019). LGUs may take note of this.





Insights

Histogram of the Sentiment Analysis Compound Value per City



With the histograms of the 3 cities combined, we see that a **majority of their posts are positive.**

It may seem like the City of Manila has the most positive posts, but again, the reach of Manila's Facebook page is far greater than the other cities'.

To determine which FB page is truly the most positive, **another type of graph should be used.**

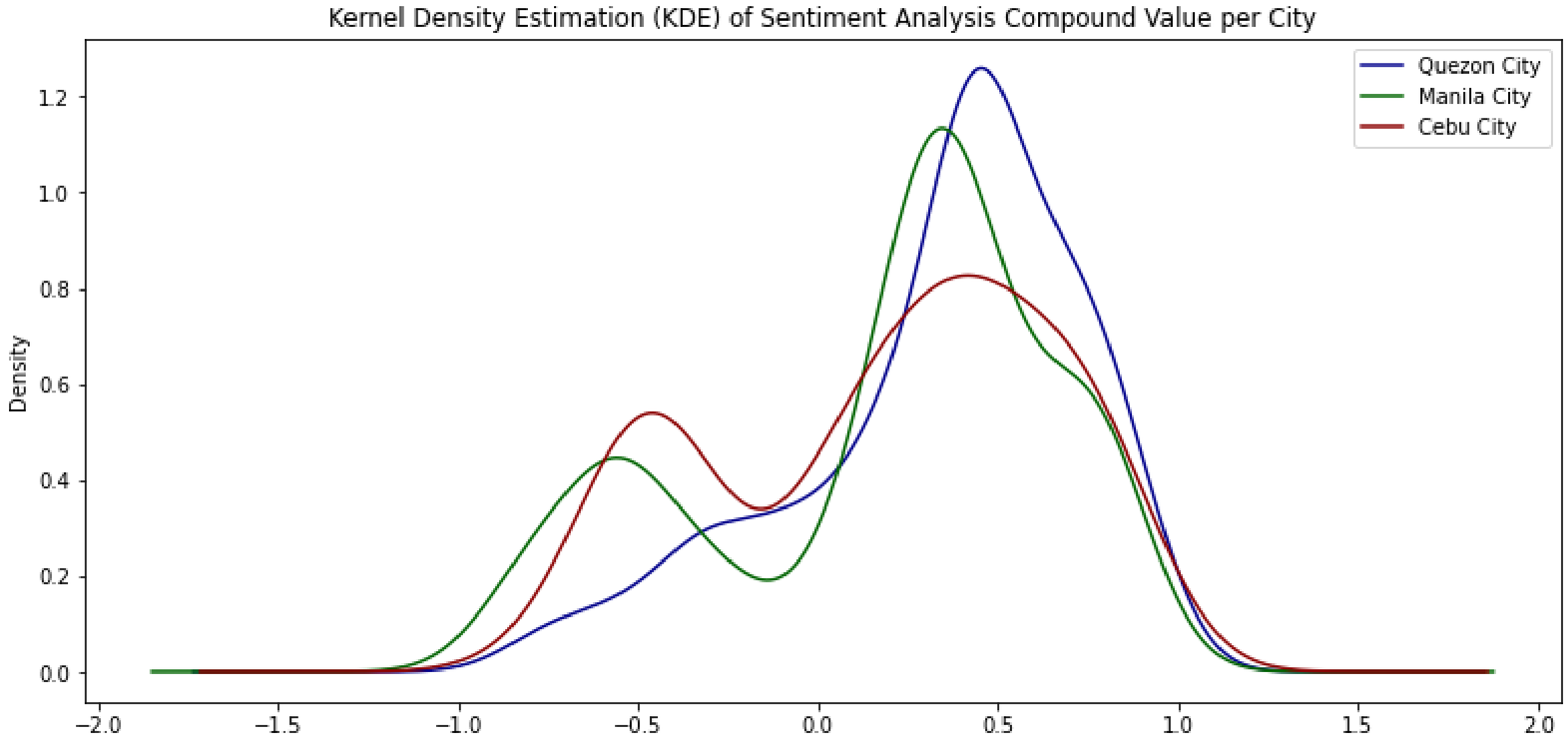
Insights

Kernel Density Estimation (KDE) of Sentiment Analysis Compound Value per City

The KDE estimates the probability density function (PDF). The value of the PDF at any given sample can be interpreted as "providing a relative likelihood that the value of the random variable would equal that sample." (AP Statistics Review, 2015)

This also means that it **does not take bias of LGU reach size**.

Here, we can clearly see that not only is the page of **Quezon City more positive** (more right) but its **peak density is also higher** compared to both cities.



Key Insights

From our findings, we were able to gather the following key insights.

Insight 1

The length of the Facebook posts by the analyzed LGUs does not significantly affect the number of reactions it recieves.

Insight 2

Although one may not directly affect the other, the analyzed LGUs are likely bounded by the same content and context.

Insight 3

A primary cause for the highest engaged posts is related to fake news, specifically local governments pointing them out.

Insight 4

Based on the Sentiment Analysis, Quezon City tops the other LGUs in being more positive in their post descriptions.

Moving Forward

Recommendations for future analysis

1. Explore analyzing the Facebook pages of other LGUs. Take larger amounts of historical data.
2. Have a Sentiment Analysis from the comment section of different posts. Find the causal factors of the sentiment.
3. Study the correlation between Sentiment Analysis compound value and the engagement of the post.

Thank you!

Have a great school year ahead!