**Lights on the Horizon**

**Valence Density and Lexical Variance as a Significant Factor for Detecting Erroneous Reporting of Unidentified Aerial Phenomena at MUFON and NUFORC**

Graham G. Waters

Department of Graduate Studies, Brandeis University

RSAN 175: Analytics Strategy and Management

Dr. Paul Dooley

March 24, 2021

*Keywords: MUFON, NUFORC, Balanced Scorecard, Python*

**EXECUTIVE SUMMARY:**

This paper provides a cursory introduction to improvements that MUFON and NUFORC could make to their data warehousing solutions and gives some analysis of the data. MUFON and NUFORC are two civilian UFO reporting organizations that have been around for decades. I still believe that there is potential in using the Star Link Satellite sightings for a control group, allowing researchers to generate quantitative measures that gauge the potential value of a UAE report in near-real-time. Longer-term, I would like to see MUFON and NUFORC develop an autonomous detection system that integrates with webcams worldwide and potentially implement deep learning to detect outlying events.

My analysis of MUFON and NUFORC data led to the following findings:

Using two methods of pattern detection that reduce or eliminate bias and subjectivity, hierarchical clustering and density estimation, I found that the data was highly insight-rich. Therefore, hierarchical clustering was selected as a pilot study because it would introduce some pattern recognition capability into the analysis while being straightforward enough for exploratory analysis.

Site Valence Density Analysis led to some interesting correlations between erroneous reporting and various levels of UFO activity that are worth exploring more deeply. Unfortunately, the correlation was difficult to quantify under the time constraints placed on the project. Given time I believe that the report-site density analysis would benefit from a more sophisticated model that supports artificial intelligence or machine learning and can consider various levels of impact on the reported sighting.

**Abstract**

To many, the Mutual UFO Network (MUFON) and its quasi-competitor, the National UFO Reporting Center (NUFORC), are simply two of another group of conspiracy theorist organizations, and to others, they represent the lone gunmen searching for what is really happening, believing firmly that *the truth is out there.* As for MUFON, the nonprofit is one of the oldest of its kind and is entirely volunteer-based. As a 501(c)3 Charitable organization, they invite people from across the world to work together and collaborate on a subject they are passionate about (*MUFON - Recent UFO Sightings | Daily Alien News & Encounters*, n.d.). Close on their heels, in 1974, NUFORC was founded, also as a nonprofit (*National UFO Reporting Center*, n.d.). What these organizations may find difficult is how much data they process through their reporting dashboards and, even worse, the backlog of historical data sitting on a shelf in a warehouse. This study is to examine whether valence density is a viable factor to determine the value of a report or whether value can be determined without extensive machine learning or artificial intelligence backgrounds. It will also touch on how balanced score carding and identification of critical metrics for MUFON, in particular, could assist them with changing the organizational design, human capital, or decision-making practices to enable them to deal with a notoriously unruly data set: *American UFO Sightings.*

**Problem Statement:**

The two giants in the hunt for the unseen, NUFORC and MUFON, both maintain massive data repositories that they have collected from amateur and professional alien hunters around the world. The analytical challenge for these researchers is how they can get past the *hype mentality* that causes *many* erroneous reports of unusual events without governing the volume of data flowing into the website. The core of the problem truly lies in their inability to come to a consensus about what a valuable report looks like. So, what organizational challenges can strategic analytics address for these organizations, and how can the application of techniques in Python and SPSS inform an analysis of clusters within their data and improve their ability to generate high-quality data insights about sightings of unusual phenomena?

**Hypotheses:**

The reality is that such an in-depth issue generates many hypotheses and that it necessitates them. The limiting factor, besides mental block, always includes time, computing power, and the availability of data. With this in mind, consider the following hypotheses, some of which will become opportunities for further research.

1. Through careful analysis of reports from MUFON and NUFORC that have high similarity factors in lexical format and location of sighting, a more holistic understanding of the reporting individual will aid researchers to determine the value for their report.
2. In the year 2020, there were more people who were up late at night and homebound, and this could have caused higher volumes of data for analysis than ever before. This should be researched further through analysis of Social Media posted about phenomena.
3. The geospatial density, as defined by *valences*, of Reports has been mentioned as a factor in the successful efforts of some self-proclaimed *ufologists* of isolating real events.
4. Text analysis of unstructured and public reports would provide a method of determining the strategic value of each individual report. This will allow researchers at MUFON to more accurately determine centrality points within report clusters. Word Usage within clusters is an indicator of credibility

**Areas for MUFON to improve in organization and reporting**

***Data Warehousing***

The data storage solutions that the organization is currently using are increasingly arcane as they are rarely used by those who can afford to use a real solution with an actual GUI. They limit their users to fifty records at a time and provide no API access or developer solutions. This is sharp contrast to the NUFORC site, which is downloadable in moments and free. The data in the comments for NUFORC is sparse when compared to MUFON’s rich tomes of commentary from reporters and thus offers less in the way of sentiment analysis; however, their consistent maintenance of the geospatial data and integrity of records is evident as the quality of their dataset is very high. MUFON could do a better job at keeping their data clean and analyzable.

**Data Quality Assurance**

Some of the online comments are credible, and even possible scenarios that could occur. These types of sightings should be getting higher value and better ratings than sightings with low corroboration and core valance density. Extrapolating the concept of density analysis to the country requires a talented team of Quality focused professionals who are aware of the needs at the company and also have passion in their work. Below are several examples of low-quality data in MUFON’s Database.

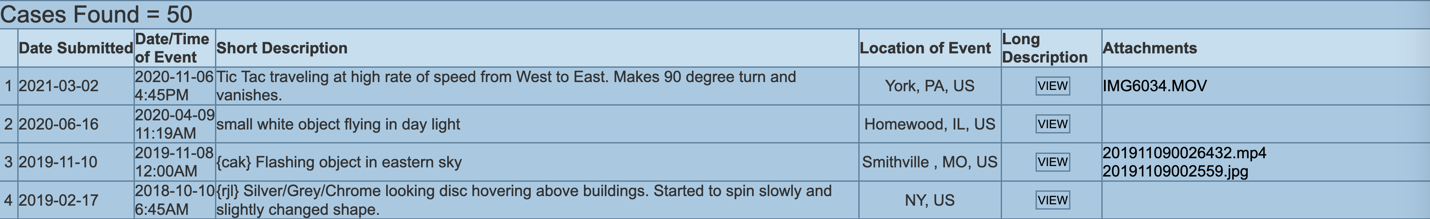
**The Overly Spiritual Commentary**

“At first, I thought it was a star, the north star, but it was in the west sky. As I was goin back in my house, it moved, I grabbed my door frame n case I was moving, and it moved again. I stepped back outside and hunkered down grabbing my knees so I could be sure I wasnt moving, it moved again. I went inside to get my brother n law, n we watched it for 45min. It would squiggle around, then elevate, then go into a serpantine motion, like a snake on water… Now about who they are, genesis 11:1-9 explains bout tower of babel, God said, I can see when man comes together, nothin is impossible,, then God destroyed the city not the tower. They are humans, in NASAs studies theyve learned that you cant develope babies without gravity. In Texas, in the 50s our govt did autopsies on pregnant aliens. They are in the skies developing their babies. They are humans that have lived up there for thousands of years, they are smarter than we are. This shouldnt surprise anyone, we have only been able to fly for around 109 years, and we can already keep people in space for years at a time. God wants them to be restored, malachi 3:5 do not deprive the aliens of justice. God never lets his people wander forever and he always reserves a remnant. Jesus said he is coming SOON, he said that 2000 years ago. Every star in the sky. They got my attention, then I awoken spiritually” (*Recent UFO Sightings | Daily Alien News & Encounters*, n.d.).

**The admittedly prepared but unsubstantiated post**

“I have a Sigma mirrorless camera that lI enjoy taking cloud pictures with. I looked up and saw this tiny white solid object moving across the sky. It was a dot but it was moving faster than most planes. So I started taking pictures using the viewfinder. I have trouble finding small objects in the viewfinder without my glasses. I didn't have my glasses own so, I was not sure if I actually got a picture of the object. It was a strange-looking because it didn't look like anything I have seen before. Luck was on my side because I did get 4 pictures. When I looked at the pictures zoomed in, it was fascinating. I thought it could be a satellite, but I doubt that it is” (*Recent UFO Sightings | Daily Alien News & Encounters*, n.d.).

This post admits to capturing four photos of an unidentified flying object yet they do no upload the photos to MUFON’s database as shown in the figure below.



**The dramatic and inappropriate commentary**

**“**It was a normal night. I had put my 1 year old son to bed and had laid down to go to sleep. I woke up a few hours later and the room I was in looked similar to my bedroom - large, king size bed and dresser in the same position inside the 20'x15' room. But surrounding the room was sheer curtains hanging from ceiling to floor and on the other side of the room were people, medical staff of some sort, watching everything I did. I couldn't see them clearly through the curtains…I then went back to sleep and woke the next morning to a normal day” (*MUFON - Recent UFO Sightings | Daily Alien News & Encounters*, n.d.).

There is a significant portion of this removed out of necessity. One thing to mention about this description is the inclusion of the dimensions of the room makes the story less believable. Something that could inform the value, mentioned earlier, to some investigators are comments that include rational thought. If a post says the phrase, “*it could be a satellite”* yet the person still reported the sighting then it suggests that the event, whatever it was, was at least out of the ordinary.

**The Rational Report**

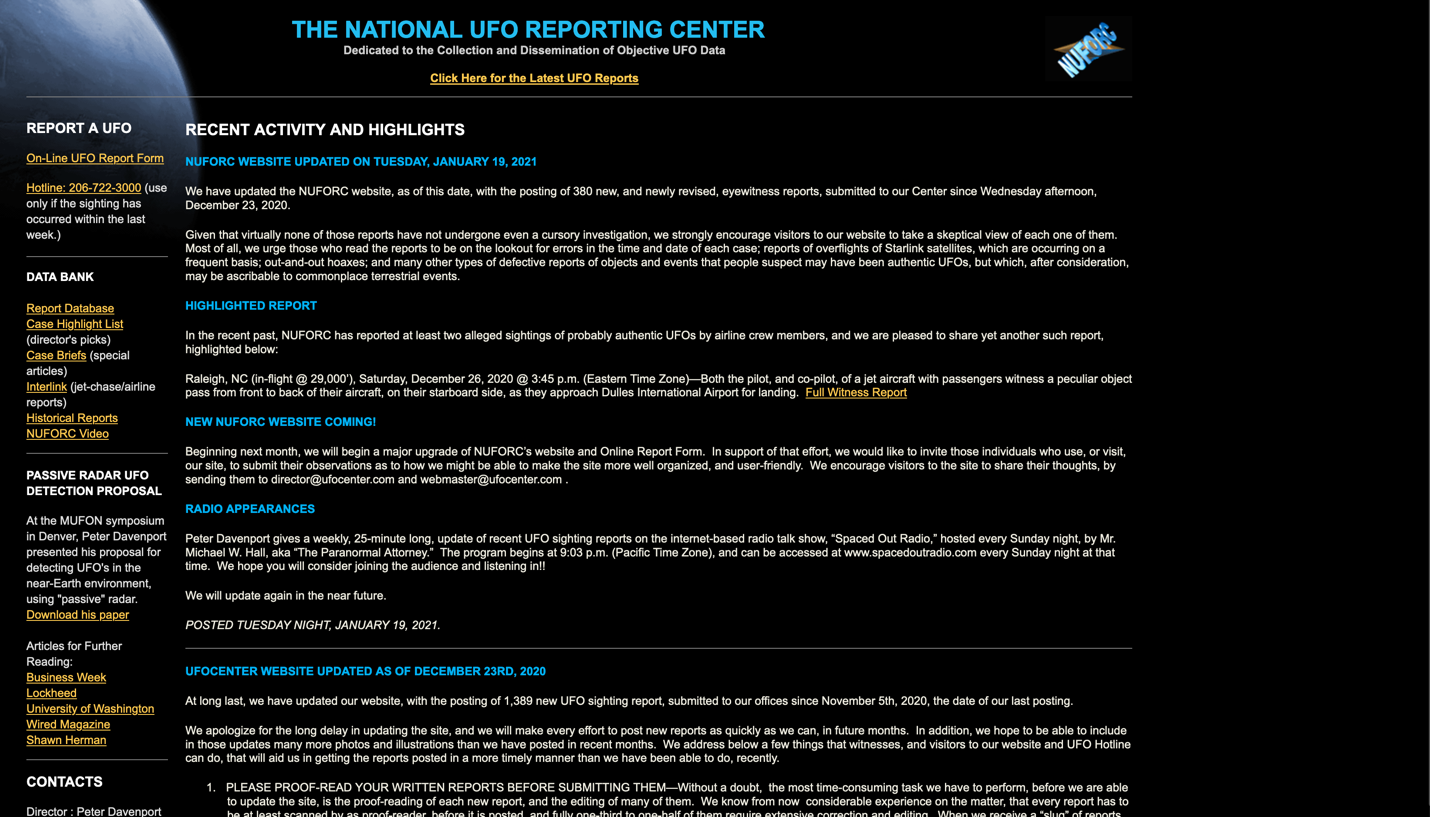
“I was laying on a swing outside while my kiddo was laying on her trampoline. We were both looking up at the sky (she was looking for constellations) when she pointed out a small (Star-like) bright light that was flying out of the NW sky to the SW. My first response was “it’s probably a satellite”. Note: These did not have any flashing lights (there were obvious planes flying over head at a much lower altitude (identifiable by their wing lights & some by engine sound) as they flew over. These star-like lights continued to fly over in a consistent line. My kiddo exclaimed, “mom, there’s more, they just keep coming” so I sat up and saw a continuing line of them flying towards us. After about 20 flew over (amongst airplanes which were clearly at a lower level) and more of the star-like lights were visible in the distance I went inside & called my husband out. In total, I probably saw between 40-50 of these star-like lights fly over. They made no noise, they did not blink or flash. They appeared the same size as the stars in the sky around them, except they were moving. They had a steady pace to them. My husband called a friend of ours who lives about 7 miles away to the west of us and she and her husband went out side and observed them as well. The lights were high enough that we would not have been able to pick them up on cell phone cameras” (*MUFON - Recent UFO Sightings | Daily Alien News & Encounters*, n.d.).

This report even has an additional group of witnesses seven miles away, it mentions planes flying *with* the lights, a believable plotline and lots of witnesses and relatable contexts. These are very challenging to isolate and even more challenging to do in a short period of time. For this reason, this study will not delve into this aspect of value; instead, consider the concept of valence density.

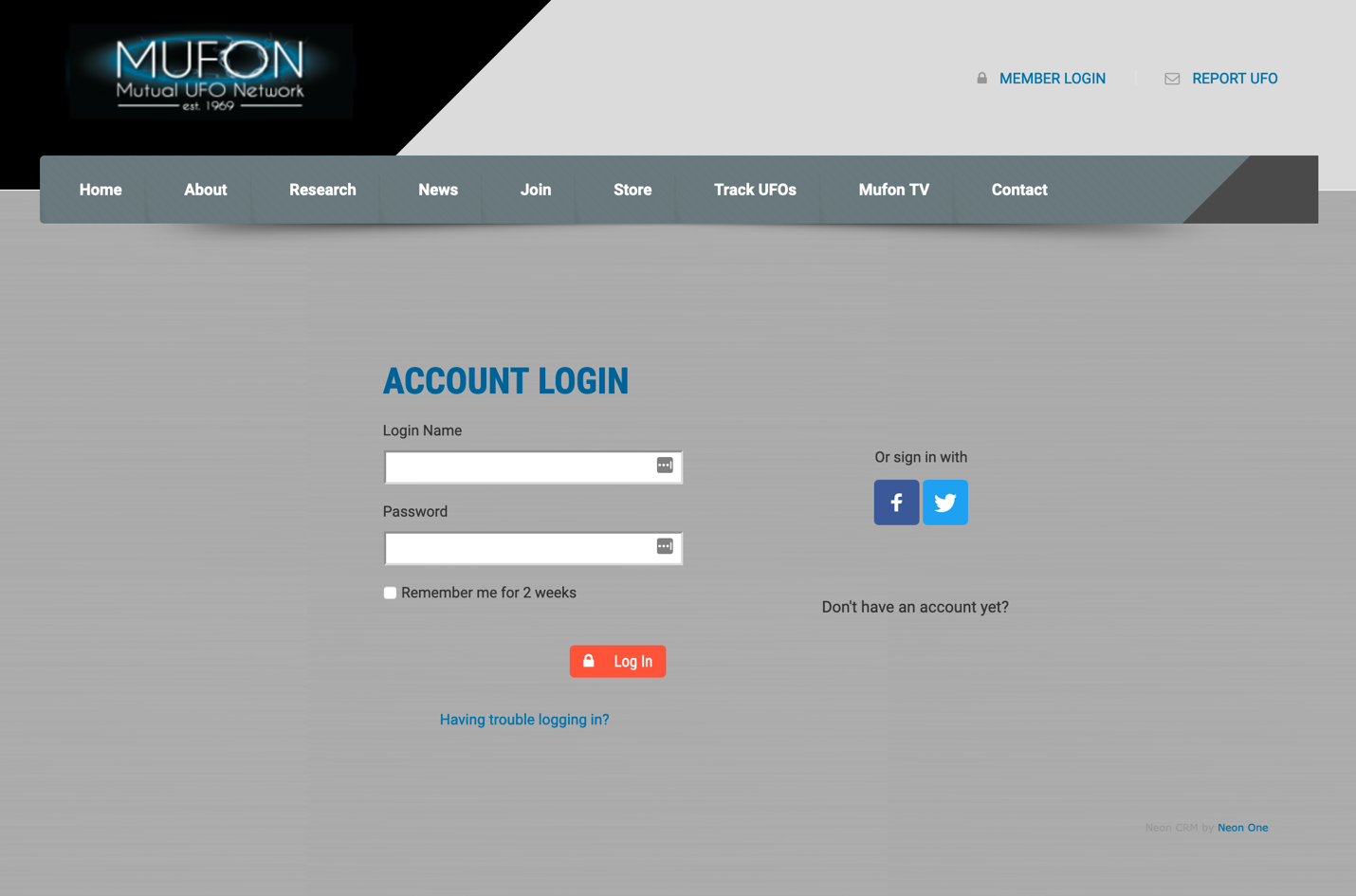
**Valence Density**

One aspect that neither MUFON nor NUFORC have made a clear part of their strategy online is the concept of valence density. In chemistry electrons live in spheres surrounding atoms that are called *valences* and this analogy is perfect for this study. In the rational commentary above the woman mentions that they called their neighbors seven miles away to see if they saw the same thing. This is an example of a valence radius. For this couple they have connections up to (as far as they revealed) seven miles away. This could be incorporated into the bio of the person that posts on MUFON and used to inform machine learning to learn about how these people and people in their sphere react to these kinds of unexplained events. One place that this shows up a lot is in classification. Clustering algorithms essentially find these valences through comparison and iteration. Some have even applied them to bee hive development (Gambhir et al., 2018).

**The Issues of these Companies’ Online Interfaces**

Another limiting factor for MUFON and NUFORC is the lack of imagination in their UI and thus UX on the websites. These websites are shown in the two figures below. 

**Figure 1**. NUFORC Main Page (*National UFO Reporting Center*, n.d.)

****

**Figure 2.** MUFON Interface for logging in (MUFON - Recent UFO Sightings | Daily Alien News & Encounters, n.d.).

If picking between the color schemes alone NUFORC has the better-looking site. This is not a marketing critique however, and there may be more utility in simply suggesting that higher UX may drive up profitability.

**Research Design/Data Collection**

The data used in this analysis has been paid for despite the strange loopholes that MUFON has in their database system that is secure in all the wrong ways. In total 419 rows of records were gathered from MUFON for analysis after cleaning and removing the most unusable of the reports.

**Data Visualizations**

* ***Plotly*** 
  + Plotly is a free, open-source library for creating interactive, publication-quality graphics on the web.

I plan to use these libraries to generate a variety of charts and maps of my data in order for me to understand the information I am analyzing, as well as share my findings with other collaborators that are not as familiar with statistics or software.

**What credibility factors should be considered when evaluating reports?**

1. Similar reporting in a certain mile radius of the report
2. Similar reports on that day
3. Airports in the area
4. Likelihood of falsification based on word choices
5. Key buzzwords that should flag interest
6. Key locations that have reputations for UFO activity.
7. Reported shape of UFO.­­

**Methods of data mining to improve the quality of MUFON reports**

The first step towards quality assurance in this large dataset is to narrow down the data into scannable subsets that represent clusters within the reports. One method used for this is known as K-means Clustering. This can be done in SPSS and will be shown later on in a cursory manner as there is much to explore on the data side.

**Initial Format of Extracted Corporal Data**

2021-03-15ø2021-03-15ø1955-06-012:10PMø1955-06-012:10PMøStopped to observe us, then moved slowly.øStopped to observe us, then moved slowly.øLos Angeles , CA, USøLos Angeles , CA, USøøøLong Description of Sighting ReportI was performing hydrographic surveys in the port of Los Angeles. At about 645 pm I went out on the back of the boat and layed down to take a break. I saw 3 white lights flying north to south very fast. They were in a parallel line and the distance between them didn't vary. All the lights we saw were the same white, circular, and slightly brighter than your average star. I called to my boat captain (Robert) to come out and look. It was clear out. He came out

…

and saw the first 3 lights. text/htmlø<selenium.webdriver.firefox.webelement.FirefoxWebElement (session="dea773d6-7ba6-ab4e-b4f8-03ddb1542ca0", element="45379dca-6836-f749-b09a-b247128ea959")>ø2021-03-07ø2021-03-07ø2021-03-071:05AMø2021-03-071:05AMøstrange light moving in different directionsøstrange light moving in different directionsøAldie, VA, USøAldie, VA, USøøø.

Adapted and extracted from: (*MUFON - Recent UFO Sightings | Daily Alien News & Encounters*, n.d.)

To clean the dataset and transform it into usable data for analytics required the creation of the following python codes snippet which is taken from a larger function included as Appendix A.

for k in tqdm(range(3,pagerows)):# following the xpath pattern on the page going down each the number of rows on the page.

#ic(k)

mufon\_file = open("MUFON\_Database\_Scraped\_Reports.txt",'a+')

mufon\_file.write(topicstring)

front = "" # init and reset every iteration

back = "" # init and reset every iteration

inner\_string = "" # init and reset every iteration

for index in range(1,len(front\_elements)+1): # go through [description\_x\_front,date\_sub\_x\_front,date\_time\_front,sdescr\_front,location\_of\_event\_front,attachments\_front]

#ic(index)

full\_path\_x = "" # init and reset every iteration

# generate the xpath

# in the third line down the 3 is unchanging while the 2 changes every iteration of this inner loop

front = front\_elements[index-1]

back = end\_elements[index-1]

full\_path\_x = front + str(k) + back # /html/body/center/form/table/tbody/tr[3]/td[2] for example.

#ic(full\_path\_x)

# extract the contents of the field

if index != 6: # is other column besides descr.

field\_contents = driver.find\_element\_by\_xpath(full\_path\_x).text

#ic()

# special case: long description field has a button to click first.

if index == 6: # index = 6 is a clickable button.

field\_text = "" # init and reset

field\_contents = descriptions[k] # should allow the browser to access the popup window

try:

field\_contents.click()

time.sleep(.25)

except Exception:

driver.quit()

#ic()

field\_contents = "n/a" # save nothing in the field

shifty(1,2) # pause for a moment then check field contents too see if click was successful above

if field\_contents != "n/a": #means the button was clicked

#get current window handle

p = driver.current\_window\_handle

#get first child window

window\_handles = driver.window\_handles

for w in window\_handles:

#switch to child window

if(w!=p):

driver.switch\_to.window(w)

break

shifty(.5,1)

el = driver.find\_element\_by\_tag\_name('body')

#print(el.text[0:30])

field\_text = el.text

inner\_string = inner\_string + "ø" +str(field\_text) # the delimeter is the ø (option o) symbol for this csv file.

#ic()

# if any error or any exception is flagged then this trips

p = driver.current\_window\_handle

driver.close()

driver.switch\_to\_window(window\_before)

shifty(.5,1) # patience...

descriptions = driver.find\_elements\_by\_class\_name("button") #refresh the descriptions array

else:

# all other indices

new\_field = "" #init

field\_contents2 = str(field\_contents)

fields = ""

for i in field\_contents:

if i != ',' and i != '\n': # remove all commas

fields += i

else:

fields += " "

field\_contents = fields

if field\_contents.find("\n") < len(str(field\_contents2)) and field\_contents.find("\n")>-1:

new\_field = field\_contents2[0:field\_contents2.find("\n")] + " " + field\_contents2[field\_contents2.find("\n")+1:len(field\_contents2)]

inner\_string = inner\_string + "ø" +str(new\_field) # the end of record delimiter is the end of line character

else:

# if the index is one less than length then the program is looking at the last field (attachments field)

if index < len(front\_elements)-1:

inner\_string = inner\_string + "ø" +str(field\_contents2) # the delimeter is the ø symbol for this csv file.

else:

inner\_string = inner\_string + "ø" +str(field\_contents2) # the end of record delimiter is the end of line character

#ic()

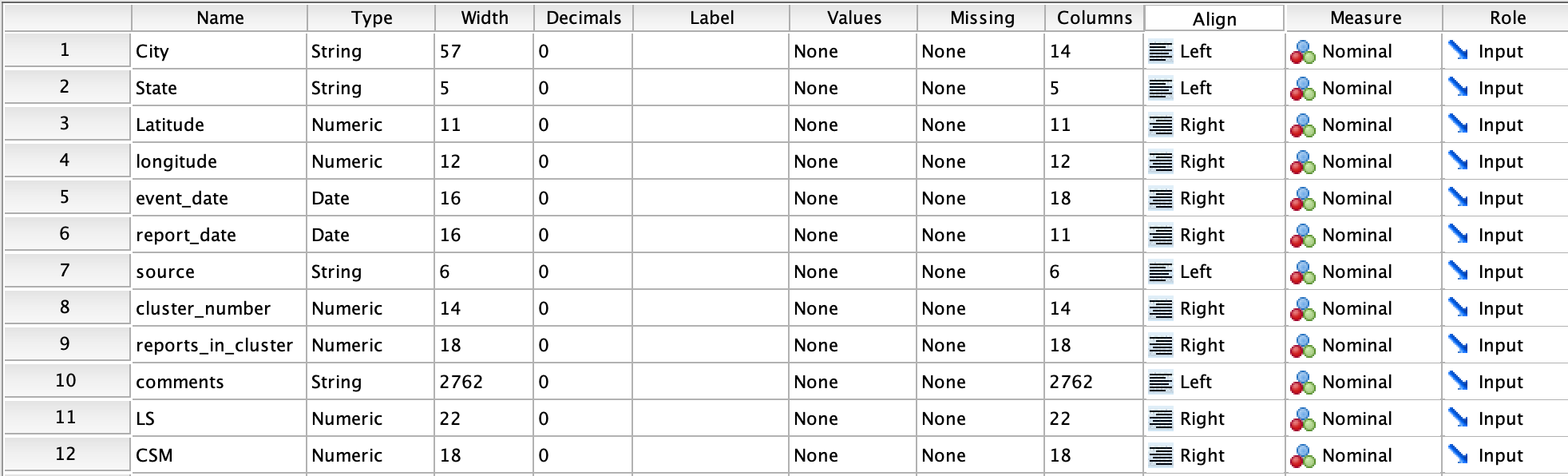
#print(inner\_string) #debugging in the terminal window

# now inner\_string contains the data from the row in string format

# it can now be saved to the file as a new line

#inner\_string += "ø" + timestamp\_string

Once the dataset was transformed into usable data that matched the formatting of the NUFORC database each record was cross-referenced with the records within NUFORC for the state of Texas. The scope of the NUFORC data is beyond the scope of this study and a more granular focus should improve the outcome. For this reason this case study will primarily focus on the state of Texas. Shown in the figure below, there are several variables that are considered in the paper.



**Figure 3**. SPSS Dataset generated with python code in Appendix A.

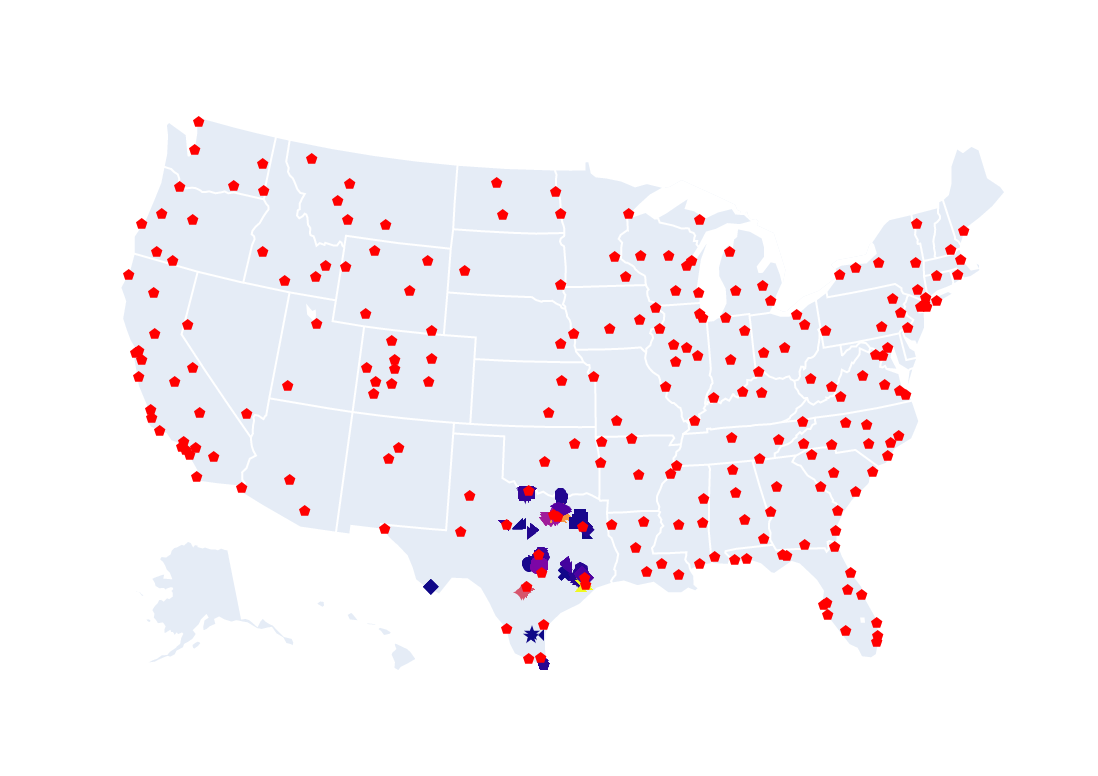
**Geospatial Considerations for MUFON and NUFORC data**

Two functions were developed to measure geospatial distances between points defined by coordinates. These are: *finding\_nemopolis() and chasm()*. These functions are included in Appendix A. As a metric this study considers *valances* around each report. The square, referred to as *valence area*, that surrounds each report will be tested for the following key metric:

1. How many other reports are there within the valence area? This will become known as the *reports in cluster* variable.

This, and sentiment of the clusters would be the two metrics are integral to MUFON’s strategic success in reaching their self-proclaimed goals of being the voice of UFO hunters everywhere.

**Considering Air Traffic**



**Figure 4.** Plotly-generated scatterplot geoplot map in Python. Shows locations of major airports in the country and previews the clusters of reporting in Texas.

***Airports***

Many people may be witnessing air traffic from local airports and mistaking it for UFO activity. Airports are a consideration when plotting reports on a map, but we must distinguish the difference between commercial and military airfields. This fails to mention the abundance of space debris that can be seen from the earth occasionally and the thousands of satellites there are in orbit around the earth right now (*Satellites/Debris/Rocket Bodies Currently in Orbit*, n.d.).

***Areal Coverage***

Sizing and comparing two valance areas presumes that there is some underlying variable that is being measured. What if there are multiple UFO reports within the same area? In addition to these metrics to describe UFO activity, another metric has been used as a proxy for human population density: reporting density. This metric was selected over a neural network or regression because it is more general in nature.

***Military Bases***

The subjective nature of UFO activity renders statistical inference methodologies nearly useless. In order to overcome the skepticism of those that do not believe in the facts of UFO phenomena, an unbiased, third-party approach was taken. The use of a machine-learning algorithm was explored but ultimately deemed beyond the scope due to the sparsity of data that such specialized silos represent.

**Quantifying a Lie**

The fundamental difficulty with this study has been quantifying lying or dishonesty in text. There have been many studies on what lying looks like in person but far fewer on written lies. Sometimes a lie may be nothing more than a truth that is misleadingly represented. This makes determining a lie so much more challenging for UFO reports. Not everyone experiences UFOs, but it seems that not everyone that does experience UFO-like phenomena reports it. It seems like a logical step to task a bot to reach out to people that lived in the areas of significant encounter clusters to find more people that witnessed potential events. Newman [8] found that across five studies, deceptive communications were characterized by fewer first-person singular pronouns, fewer third-person pronouns, more negative emotion words, fewer exclusive words, and more motion verbs. When compared to non-deceptive communications, deceptive communications were more likely to contain words related to lying and concealment. In a study by Bond and DePaulo, college students in the United States were recruited through an online crowdsourcing site (Mechanical Turk) to complete an online survey about their childhood experiences with deception. They found that those who were more likely to deceive others lied more often, but conversely, those who lied often were not necessarily more likely to be deceived (*Accuracy of Deception Judgments - Charles F. Bond, Bella M. DePaulo, 2006*, 2006). The link between lying as an action and being lied to as a recipient of lies is tenuous at best.

**Table 1.** SPSS clustering results and convergence details (Field, 2013) .

|  |  |  |
| --- | --- | --- |
| **Initial Cluster Centers** | | |
|  | Cluster | |
| 1 | 2 |
| latitude | 29.321586000000000 | 32.226666700000000 |
| longitude | -103.61601910000000 | -95.22527780000000 |

|  |  |  |
| --- | --- | --- |
| **Iteration Historya** | | |
| Iteration | Change in Cluster Centers | |
| 1 | 2 |
| 1 | .000 | 1.925 |
| 2 | .000 | .000 |

|  |
| --- |
| a. Convergence achieved due to no or small change in cluster centers. The maximum absolute coordinate change for any center is .000. The current iteration is 2. The minimum distance between initial centers is 8.879. |

**Findings**

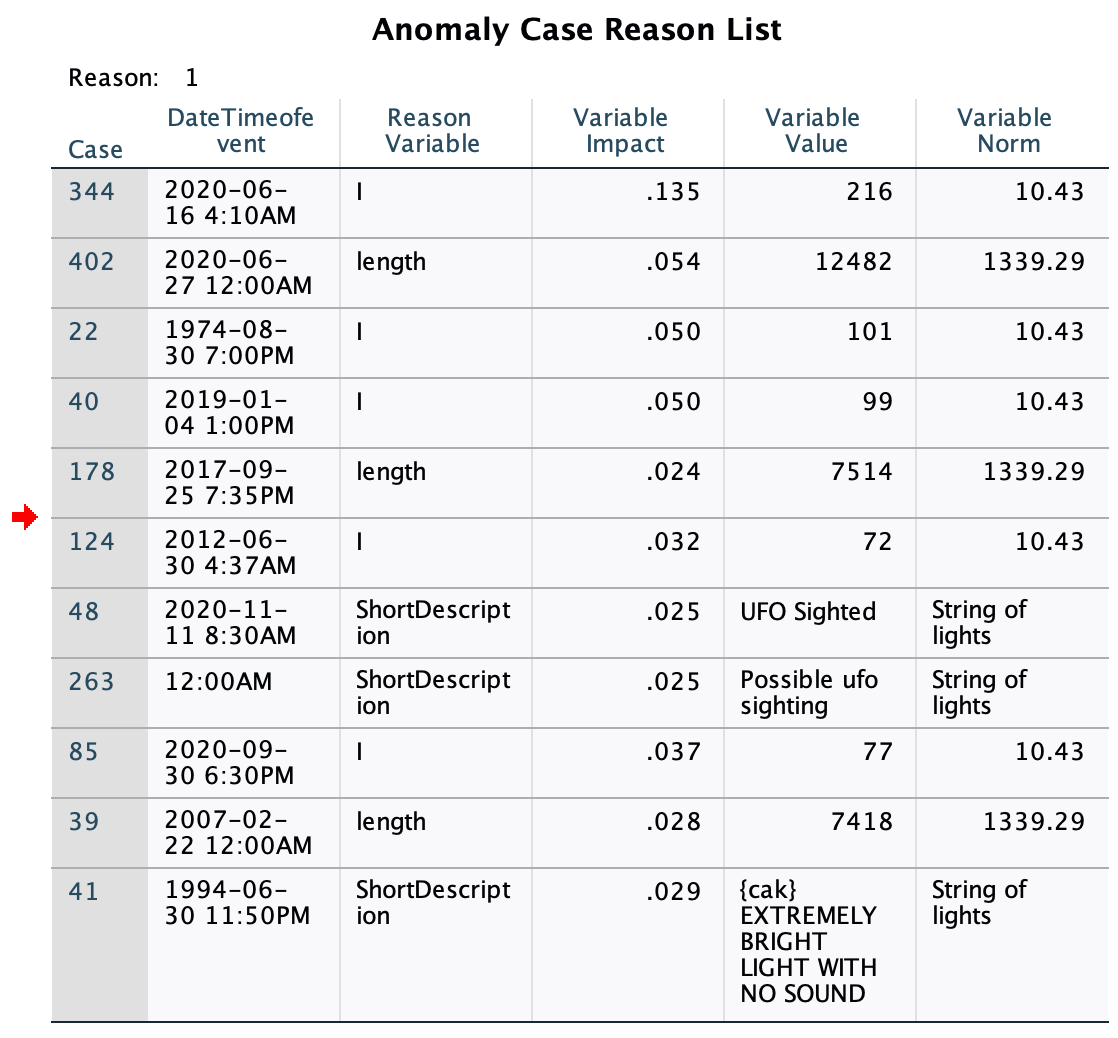
 This study focuses on reports in the state of Texas and investigates whether clustering reports based on proximity could yield higher lexical similarity[[1]](#footnote-1) within clusters. If MUFON data can be clustered in conjunction with the data that is procurable from NUFORC then there will be much more potential insight than if viewed alone. An analysis of the text of five-hundred MUFON Reports scraped from the database reveals several interesting significant connections between terms within the comments. The top used words in these reports collected from both sources were *I, light, my, star, red, low, line, time, object, and bright.* These will have been biased by the queries submitted to the database: *blinding, beautiful, lights in a line, moving star, like star, line, starlink, string of,* and *strange light*. Using SPSS to model the data yielded a two-cluster system within the state of Texas with centroid coordinates shown in the table above. Due to constraints on the research timeline there was very little lexical analysis that could be done programmatically on the dataset. There are other resources that have proven useful for this. An anomaly test returned several cases that should be investigated further to determine what causal factors there are, if any, that could be indicative of potential organizational improvements.

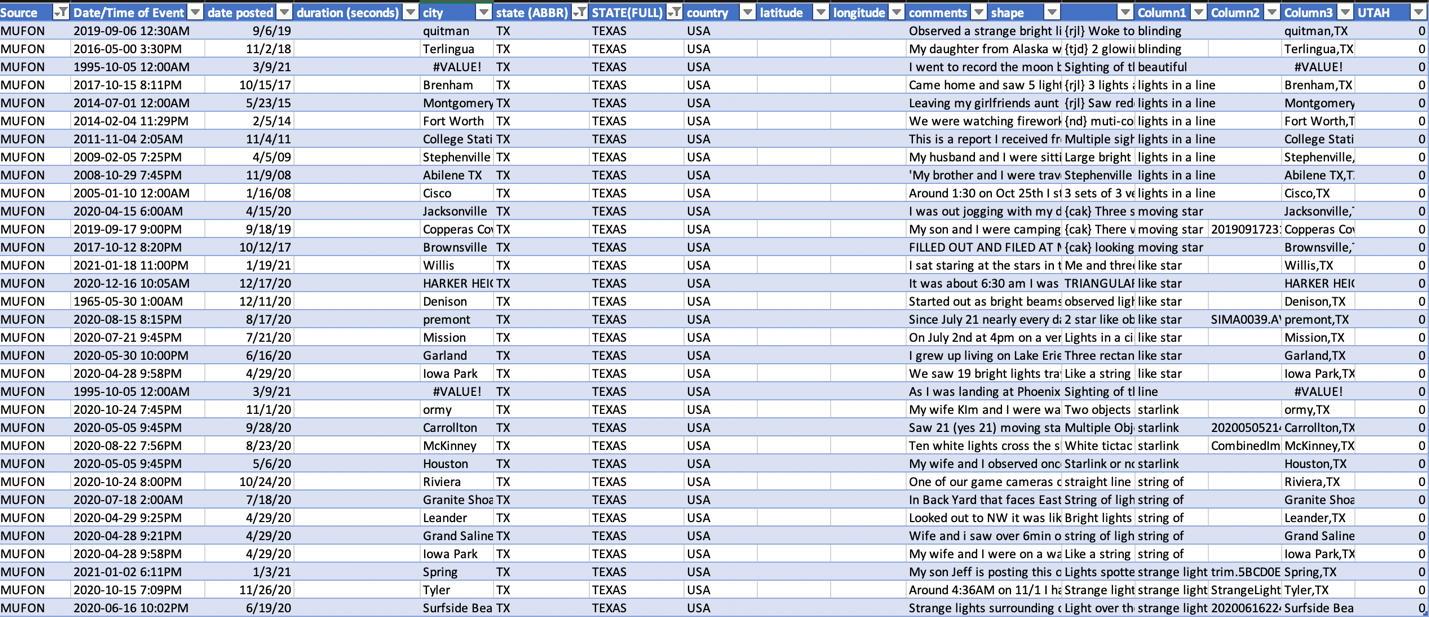
Figure 5. Anomaly Detection in SPSS to identify unusual cases within the dataset to investigate (Field, 2013).

**Data Analysis/ Results:**

Using python to parse results found in the NUFORC database as well as data from MUFON’s database, the following database was generated. There were two, one from MUFON and the other from NUFORC and each had separate issues that needed to be addressed before they could be used for insight and comparison.

**Data Formatting**

**Table 2.** Format of Data from MUFON in Excel.

****

Data procured from NUFORC is organized differently that the MUFON data available online which made combining the two data sets time-consuming as an intermediate Python Coder. One critical area for MUFON to improve is a standardization of their data formatting to allow for easier analysis and quicker insight generation. This is if their goal is to find if the truth is out there. The following code was written to determine distances between cities coordinates. It could be adaptable for further papers in the next months.

#todo use geopy to find coordinates of cities mentioned in the reports and return those

#todo cities' coordinates, google rated importance, and bounding box for lat and longitude.

**def** geopy\_city\_to\_coordinates(cityname,statename):

'''

README

city name must be like "Austin"

statename must be in form "Texas"

no spaces they will be combined with space between them

'''

inputstring = cityname + " " + statename

from geopy.geocoders import Nominatim

geolocator = Nominatim(user\_agent='myapplication')

location = geolocator.geocode(inputstring)

try:

coordinates = [location.point[0],location.point[1]]

other\_city\_data = [location.raw['importance'],location.raw['boundingbox']]

except AttributeError:

coordinates = ['','']

other\_city\_data = ['','']

returnable\_array = [coordinates,other\_city\_data] # [[lat,long],[importance,boundingbox]

return returnable\_array

'''

result: [[30.2711286, -97.7436995],[0.8694265717866669,['30.0984576', '30.5166255', '-97.9367663', '-97.5605288']]]

'''

#todo this function is working properly and gives coordinates of cities if available.

**def** finding\_nemopolis(city\_name,state\_name):

location\_data = ['']

if len(state\_name)>2:

location\_data = geopy\_city\_to\_coordinates(city\_name,state\_name)

else:

return location\_data

return location\_data # no coordinates found.

**Preliminary Findings**

As the research progressed in the NUFORC area it became clear that the volume of data available through their website was an unviable option given the scope of this project. As a result the data used in this study focuses on reports in the state of Texas from both MUFON and NUFORC and investigates whether clustering reports based on proximity with python could allow testing for higher lexical similarity within clusters. If MUFON data can be clustered in conjunction with the data that is procurable from NUFORC, the results could prove interesting. If we can identify high-frequency linguistic features associated with certain clusters, then we may be able to predict whether someone has a propensity for lying or not. Not every cluster in this study is considered a UFO cluster because there are no frequent mentions of occupation in these reports. There are several factors that were considered, but one of the most important was how close an area was to where the event occurred. Perhaps there is some correlation between human population density and UFO sightings that can be quantified. Anecdotally, it seems like Ufologists use this approach when searching for possible cases. Two distinct methods have been used to uncover the hidden patterns in the data while reducing or eliminating bias and subjectivity. These are hierarchical clustering and density estimation. Hierarchical clustering was selected as a pilot study because it would introduce some pattern recognition capability into the analysis while being straightforward enough for exploratory analysis. Hierarchical clustering was selected as a high-level means of organizing the data into meaningful clusters.

**Computing a Density Estimate for UFO Activity**

In order to understand UFO activity, both NUFORC and MUFON were considered from 1947 to the present day. The NUFORC dataset was downloaded in 2015, while the MUFON dataset has been continuously updated since its inception in 1974. The NUFORC data was filtered to only include reports originating from Texas, where the MUFON data includes all records since its inception. This process created a three-year window of sorts where each cluster was assumed to be valid UFO activity, with the exception of anomalous clusters that represent false positives or class anomalies.

import plotly.graph\_objects as go

import plotly.figure\_factory as ff

import pandas as pd

import icecream as ic

from plotly.subplots import make\_subplots

import plotly.graph\_objects as go

*#texas points of interest*

*#city,state,latitude,longitude,event\_date,report\_date,source,cluster\_number,reports\_in\_cluster,comments,Levenshtein\_Similarity,cosine\_sim\_vectors*

'''df = pd.read\_csv('TX\_pointsofinterest.csv')

df.head()

'''

import plotly.graph\_objects as go

import pandas as pd

df2 = pd.read\_csv('TX\_pointsofinterest.csv')

**def** plotthis():

for col in df2.columns:

df2[col] = df2[col].astype(str)

df2['text'] = df2['city'] + '<br>' + \

'event\_date ' + df2['event\_date'] + ' cluster size ' + df2['reports\_in\_cluster'] + '<br>'

fig = go.Figure(data=go.Choropleth(

locations=df2['city'],

z=df2['reports\_in\_cluster'].astype(float),

locationmode='USA-states',

colorscale='Reds',

autocolorscale=False,

text=df2['text'], *# hover text*

marker\_line\_color='white', *# line markers between states*

))

fig.update\_layout(

title\_text='MUFON and NUFORC Reporting Analysis',

geo = dict(

scope='usa',

projection=go.layout.geo.Projection(type = 'albers usa'),

showlakes=True, *# lakes*

lakecolor='rgb(255, 255, 255)'),

)

df = pd.read\_csv('TX\_pointsofinterest.csv')

df['text'] = df['city'] + ' ' + df['state'] + ', date:' + df['event\_date'] + '' + 'source: ' + df['source'].astype(str)

fig.add\_trace(go.Scattergeo(

lon = df['longitude'],

lat = df['latitude'],

text = df['text'],

mode = 'markers',

marker = dict(

size = 8,

opacity = 0.8,

symbol = df['cluster\_number'],

),

marker\_color = df['reports\_in\_cluster'],

showlegend = True,

*#size = 10,*

))

df = pd.read\_csv('2011\_february\_us\_airport\_traffic.csv')

df['text'] = df['airport'] + '' + df['city'] + ', ' + df['state'] + '' + 'Arrivals: ' + df['cnt'].astype(str)

fig.add\_trace(go.Scattergeo(

lon = df['long'],

lat = df['lat'],

text = df['text'],

mode = 'markers',

marker = dict(

size = 10,

opacity = 0.5,

symbol = 'pentagon',

),

marker\_color = 'red',

))

fig.update\_layout(

title = 'MUFON and NUFORC Data Clusters in Texas',

geo\_scope='usa',

)

*# focus point*

lat\_foc = 31.3915

lon\_foc = -99.1707

fig.update\_layout(

geo = dict(

projection\_scale=3, *#this is kind of like zoom*

center=dict(lat=lat\_foc, lon=lon\_foc), *# this will center on the point*

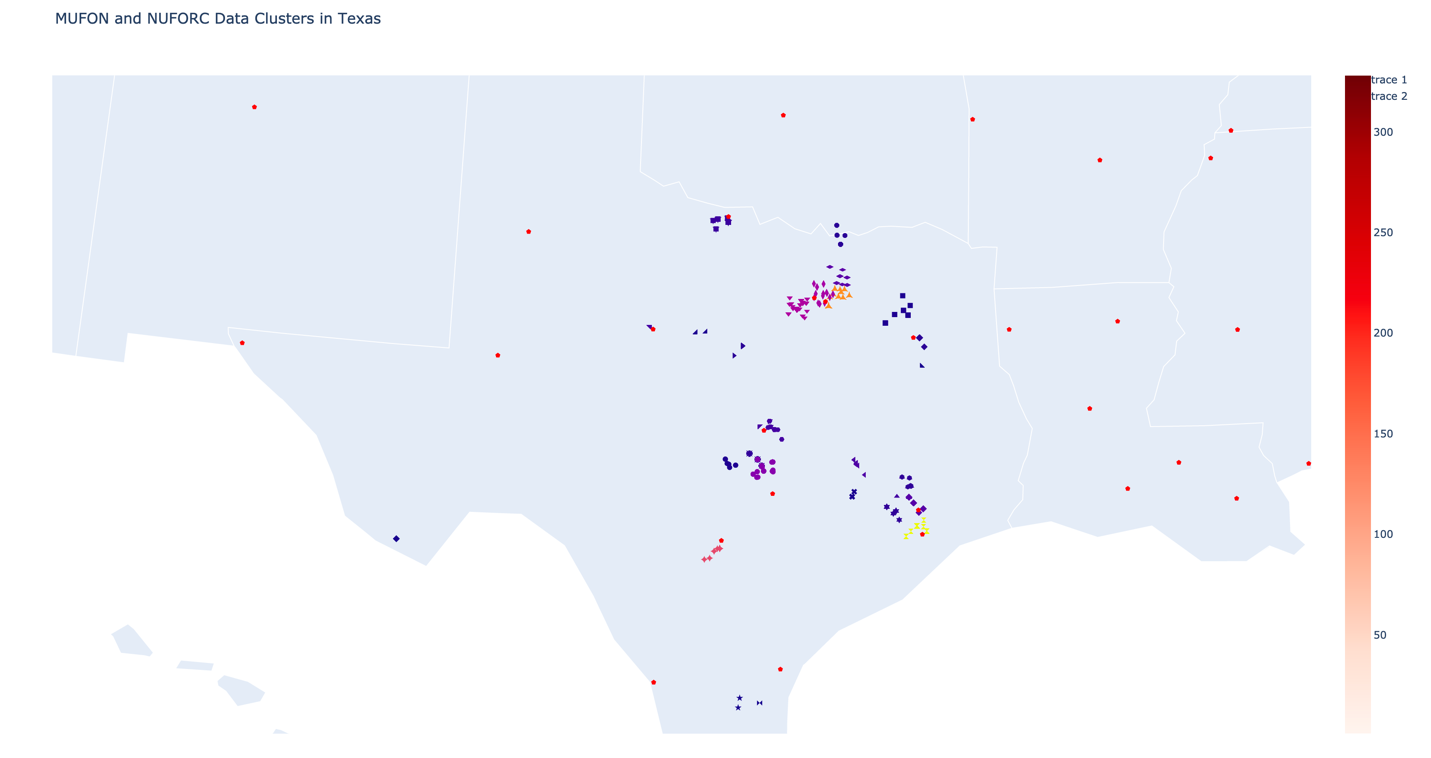
))

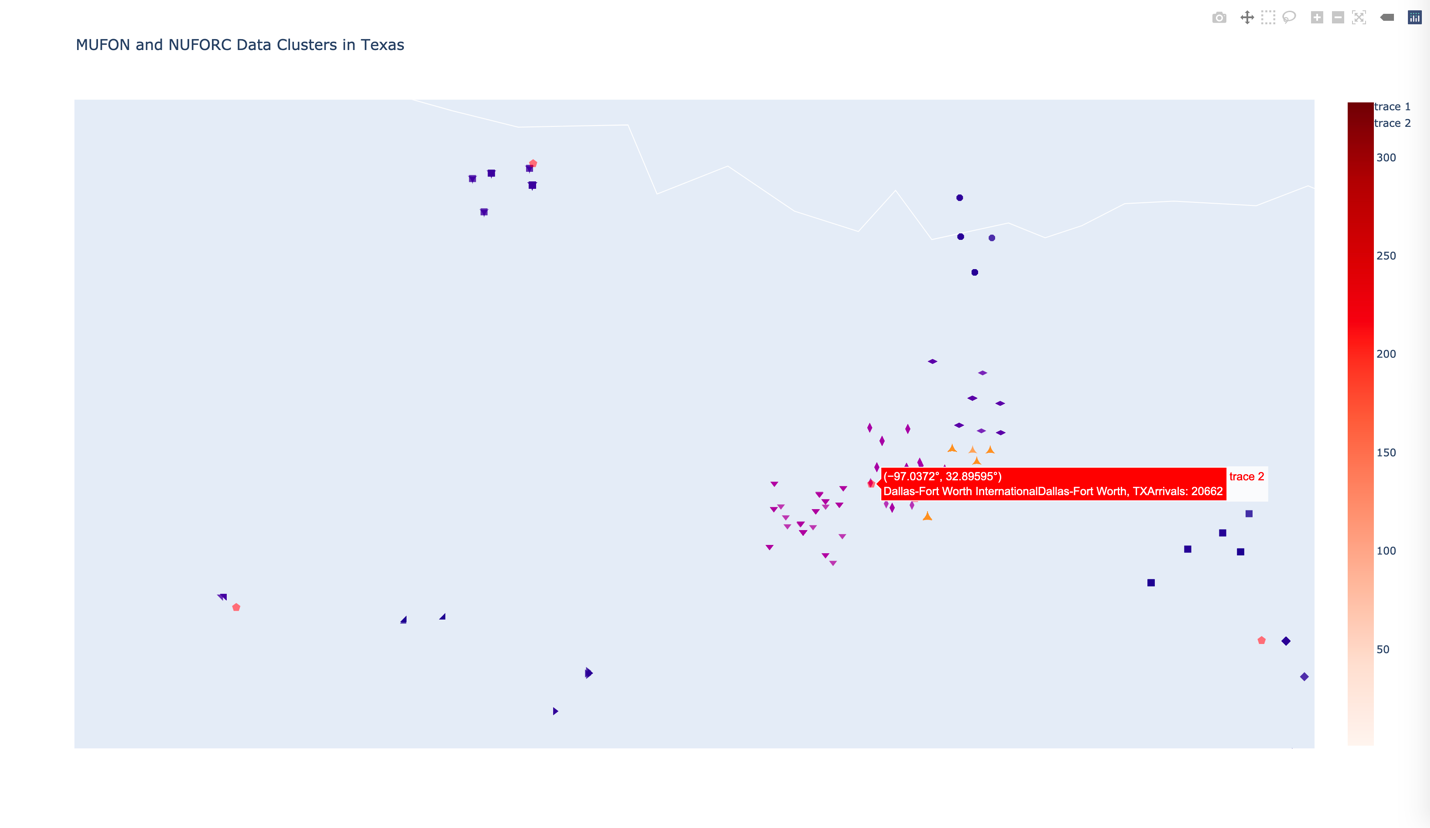
fig.show()

plotthis()

The above code was influenced by the works of (Dataman, 2020) and (plotly.com, 2020).

Getting deeper into sentiment analysis of each of these clusters is the next step in this research effort. As is noted by leaders in the field, deep learning is a key element of successful sentiment analysis, and especially with the speed that the world’s lexicon is expanding (*Understanding Emotions in Text Using Deep Learning and Big Data | Elsevier Enhanced Reader*, n.d.).





**Figure 6.** MUFON and NUFORC Data Cluster Illustration on the State of Texas using Plotly and Geoplots (Dataman, 2020).

Focusing on MUFON specifically, how can balanced scorecarding help them in their key performance areas?

**Business Units**

1. Data Collection
2. Research and Investigation
3. Education

**O: (Objective)**

MUFON will increase database quality[[2]](#footnote-2) by 10% and grow the database by an additional 25,000 reports from social media listening in the next year. Collect data on purported UFO sightings in the MUFON Database and disseminate this data to researchers around the world. One of the goals of MUFON is to educate the public on UFO phenomenon and their potential impacts on society. In the next year MUFON will increase their educational impact and reach[[3]](#footnote-3) by 10% year-over-year.

**A: (Advantage)**

MUFON has no direct competition as it is a nonprofit in a niche market. As an education platform it competes with YouTube Channels such as (insert name here) for viewers. It’s primary advantage over other potential competitors is its name recognition and compiled data.

**S: (Scope):**

MUFON’s scope is limited to the data available in current repositories and any social media data collected from primary sites. The company will not utilize any methodology that has been previously characterized as *pseudoscience* or *innuendo* in their education.

**Table 1.** Change Agenda for MUFON based on (Kaplan & Norton, 1996).

|  |  |  |  |
| --- | --- | --- | --- |
|  | | **PRESENT** | **FUTURE** |
| We are Getting | **Financial** | Subscriptions | Varying Levels of Subscriptions |
| Current tuition for MUFON University coursework. | MUFON University coursework tuition as well as paid influencers that teach online. |
|  | Grants based on meritorious research in the field. |
| Donations from hundreds of benefactors | Donations from more benefactors. |
|  |  |
| **Customer** | Random Website Traffic | Website Traffic Driven by Ads |
| Repeat Customers (subscribers) | Higher numbers of subscribers |
| Purchasers of Training Materials | More trainers until reaches economy of scale. |
| **Data** | Data from thousands of reporters | Data from thousands of reporters |
| n/a | Social Media Listening data |
| Data from news outlets | Data from more news outlets |
|  | Web Scraping Data, news, Google, RSS feeds, Webcams, Foreign country support. |
| Compiled UFO data from various data lakes dedicated to UFO knowledge management. | Maintain the compiling of these data |
| We are Doing | **Product Service Process** | Providing Data on the Website along with map data and other useful tools for UFO researchers. | Providing Data on the Website along with map data and other useful tools for UFO researchers that boost performance. |
|  | MUFON Research Library, New Website, MARRS Archive System Hosting, CUFOS Documents scanned into the CMS worldwide UFO Database. |
| Creating courses (MUFON University) to educate people about UFO phenomena and providing trainings to become a ufologist researcher in the field | Same |
| Testing Facilities around the world are occasionally used to test extraterrestrial materials. | Utilization of CISION for PR. ($13k/yr.), MUFON TV service for MUFON members. Highest Quality Materials Testing facilities around the world. |
|  |  |
| **People & Process** | Attracts people with a tendency to paranoia that typically have a conspiratorial view of the world. | Secure the services of a few trusted researchers and individuals that add credence to the ufology field, limiting the pseudoscientific approach to the study. |
| Processes are very opaque and do not promote a trustworthy face to the public. Average people are less likely to donate. | They employ high levels of transparency to promote public trust. They also show dedication to quality and accuracy to truly send out the MUFON message. |
|  | MUFON will grow their member population. |
| MUFON Partners with over 70 scientists and thousands of members worldwide. | MUFON will partner with over 200 scientists and labs around the world. |

**Table 2.** Strategy Map for Change based on (Kaplan & Norton, 1996).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Verb** | **Adjective** | **Noun** | **Additional** | **Statement** |
| Generate | highly attractive, new | subscription models | offering various tiers for different types of members. | MUFON Will Generate highly attractive, new subscription models offering various tiers for different types of members. |
| maintain current tuition and employ | talented, rational, trusted, and intelligent | online influencers on major social media platforms | to drive up interest in younger people and promote justifiable assertations to the new generation. [Brave new World Campaign] | MUFON Will maintain current tuition and employ talented, rational, trusted, and intelligent online influencers on major social media platforms to drive up interest in younger people and promote justifiable assertations to the new generation. [Brave new World Campaign] |
| Obtain | Grants | for researching UFO sightings or unexplained phenomena around the world | which shows a philanthropic motivation behind the effort to seek the truth that is out there. | MUFON Will Obtain Grants for researching UFO sightings or unexplained phenomena around the world which shows a philanthropic motivation behind the effort to seek the truth that is out there. |
| Secure | larger, more impactful, and visible | donations and funding | from various sources preferably in higher profile positions. | MUFON Will Secure larger, more impactful, and visible donations and funding from various sources preferably in higher profile positions. |
|  |  |  |  | MUFON Will |
| Generate | higher, and more fruitful, easy to convert, | web traffic on the MUFON website and social media pages | by using targeted ad campaigns and more strategic ad placement. | MUFON Will Generate higher, and more fruitful, easy to convert, web traffic on the MUFON website and social media pages by using targeted ad campaigns and more strategic ad placement. |
| Maintain | the current, brand-loyal, interested, and engaged | subscriber-base on MUFON.com |  | MUFON Will Maintain the current, brand-loyal, interested, and engaged subscriber-base on MUFON.com |
| Promote and Sell | higher volumes of higher quality and dynamic | training materials, bundles, and software | to more trainers online till economy-of-scale is reached. | MUFON Will Promote and Sell higher volumes of higher quality and dynamic training materials, bundles, and software to more trainers online till economy-of-scale is reached. |
| Collect | All kinds (unstructured, structured, high-quality, low-quality) | of data on UFO sightings or reports of other phenomena | around the globe. | MUFON Will Collect All kinds (unstructured, structured, high-quality, low-quality) of data on UFO sightings or reports of other phenomena around the globe. |
| scrape | relevant, suspicious, or potentially relevant | posts, pictures, videos, or commentary | that seems to relate to a UFO-like (or other) event occurring. | MUFON Will scrape relevant, suspicious, or potentially relevant posts, pictures, videos, or commentary that seems to relate to a UFO-like (or other) event occurring. |
| scrape | higher volumes of either freely accessible, or purchased | news material | from various websites that are likely to run stories on UFOs or MUFON-related phenomena. | MUFON Will scrape higher volumes of either freely accessible, or purchased news material from various websites that are likely to run stories on UFOs or MUFON-related phenomena. |
| scrape | higher volumes of either freely accessible, or purchased | google data, RSS feed data, Webcam footage, and other online sources | which will be scanned for any statistically significant possibility of a phenomena-event occurring. | MUFON Will scrape higher volumes of either freely accessible, or purchased google data, RSS feed data, Webcam footage, and other online sources which will be scanned for any statistically significant possibility of a phenomena-event occurring. |
| maintain | current, updated, secured | data from various repositories on UFO reports |  | MUFON Will maintain current, updated, secured data from various repositories on UFO reports |
| provide | free, accessible, and accurate | data on UFO reporting around the world as well as tools | for boosting UFO researcher performance | MUFON Will provide free, accessible, and accurate data on UFO reporting around the world as well as tools for boosting UFO researcher performance |
| Raise enough funding to begin scanning | the large, valuable, unformatted | CUFOS documents | into the CMS worldwide UFO Database | MUFON Will Raise enough funding to begin scanning the large, valuable, unformatted CUFOS documents into the CMS worldwide UFO Database |
| Create and keep creating | educational, nonbiased, science-based | courses through MUFON University | for users that wish to become professional ufologists, researching in the field. | MUFON Will Create and keep creating educational, nonbiased, science-based courses through MUFON University for users that wish to become professional ufologists, researching in the field. |
| Raise enough funding to secure the use of | trusted, scientific, transparent, and secure | science laboratories around the world | for the testing of all purported extraterrestrial matter obtained by MUFON | MUFON Will Raise enough funding to secure the use of trusted, scientific, transparent, and secure science laboratories around the world for the testing of all purported extraterrestrial matter obtained by MUFON |
| Raise enough funding to employ | the powerful | CISION PR platform | for public relations management. | MUFON Will Raise enough funding to employ the powerful CISION PR platform for public relations management. |
| Secure the services of | trusted, influential, nonbiased, and not conspiratorial | leaders’ researchers, and individuals | that add credence to the ufology field, limiting the pseudoscientific approach to the study. | MUFON Will Secure the services of trusted, influential, nonbiased, and not conspiratorial leaders researchers, and individuals that add credence to the ufology field, limiting the pseudoscientific approach to the study. |
| Create and cultivate | a transparent, trustworthy, quality-dedicated | public image | to promote public trust which will also show dedication to quality and accuracy to truly send out the MUFON message. | MUFON Will Create and cultivate a transparent, trustworthy, quality-dedicated public image to promote public trust which will also show dedication to quality and accuracy to truly send out the MUFON message. |
| generate | organic, findings-based | growth | in member populations by sound practices. | MUFON Will generate organic, findings-based growth in member populations by sound practices. |
| partner with | over 200 credible, sound-minded | scientists, and laboratory specialists around the world |  | MUFON Will partner with over 200 credible, sound-minded scientists, and laboratory specialists around the world |

**Further Opportunities for Research**

There are many other areas that would be fantastic for research in this field. For one there seems to be no social media element that is commonly being used to track unexplained phenomenon. This may be an area for opportunity for the usual UFO network as they were to expand their environmental sensing capabilities and social listening technology.

**Starlink Satellite Launches as potential control group**

The unique opportunity the UFO researchers have right now is that for the first time in history there is a control group for the experiment. Identifying reporters that were witnessing star link satellites but did not know this could possibly be the first step for man into the world of science-based deep learning about unidentified phenomena that haunt the skies. As recently as March 11, SpaceX launched 60 Star link satellites from Space Launch Complex 40 (SLC-40) at Cape Canaveral Space Force Station in Florida (*Starlink*, n.d.). According to amateur astronomers someone as the best chance of witnessing Starlink Satellites in the twilight moments, 30 minutes before sunrise or 30 minutes after sunset. Due to their heat-resistant coating they shine like pearls in a row (Visualization et al., 2015).

**Stellarium[[4]](#footnote-4) Integration**

Integrating Astronomy Application: Stellarium’s web-interface with Selenium in Python has potential as well. It has the ability to track these satellites and extrapolate their locations back in time. This would allow cross referencing the locations with coordinates in the databases of MUFON and NUFORC.

**References**

*Accuracy of Deception Judgments—Charles F. Bond, Bella M. DePaulo, 2006*. (2006). https://journals.sagepub.com/doi/10.1207/s15327957pspr1003\_2

Dataman, D. (2020, June 5). *Powerful Plots with Plotly*. Medium. https://medium.com/analytics-vidhya/plot-with-plotly-114ac106e25f

Gambhir, A., Payal, A., & Arya, R. (2018). Performance analysis of artificial bee colony optimization based clustering protocol in various scenarios of WSN. *Procedia Computer Science*, *132*, 183–188. https://doi.org/10.1016/j.procs.2018.05.184

Kaplan, R. S., & Norton, D. P. (1996). Linking the Balanced Scorecard to Strategy. *California Management Review*, *39*(1), 53.

*MUFON - Recent UFO Sightings | Daily Alien News & Encounters*. (n.d.). Retrieved March 13, 2021, from https://www.mufon.com/

*National UFO Reporting Center*. (n.d.). Retrieved March 24, 2021, from http://www.nuforc.org/

plotly.com. (2020). *Line and Scatter Plots*. Plotly.Com/Line-and-Scatter. https://plotly.com/python/line-and-scatter/

*Recent UFO Sightings | Daily Alien News & Encounters*. (n.d.). MUFON. Retrieved March 13, 2021, from https://www.mufon.com/

*Satellites/Debris/Rocket bodies currently in orbit*. (n.d.). Retrieved March 21, 2021, from https://www.n2yo.com/satellites/

*Starlink*. (n.d.). Starlink. Retrieved March 19, 2021, from https://www.starlink.com

*Understanding Emotions in Text Using Deep Learning and Big Data | Elsevier Enhanced Reader*. (n.d.). https://doi.org/10.1016/j.chb.2018.12.029

Visualization, D., Underst, T. W. T. D. I. T. O., Excel, ing of the world I. spend a lot of time with my face buried in, here, when I. find something interesting I. write about it, Cities, also as a G., & background, H. P. contributor M. about my. (2015, June 3). The Most Credible UFO Sightings And An Interactive Map. *Metrocosm*. http://metrocosm.com/map-of-ufo-sightings/

**Appendix A.**

#MUFONVARIABLES.py

#include all of these libraries

from tqdm import tqdm

from tqdm import tqdm\_gui

import time

import datetime

#from WebDriver import WebDriverWait

#from WebDriver import wait

from datetime import datetime

import string

'''

write() : Inserts the string str1 in a single line in the text file.

File\_object.write(str1)

writelines() : For a list of string elements, each string is inserted in the text file.Used to insert multiple strings at a single time.

File\_object.writelines(L) for L = [str1, str2, str3]

MUFON SEARCH DETAILS FOR KEYWORDS

This function searches through the Submitter‘s Detailed Description

To search through all cases in CMS be sure to clear the Country, State and/or Province using the button next to those menus.

Examples: (the underscore "\_" means a space in the examples below)

dog will find any word containing dog. ie hotdog, doghouse, endogeny etc.

dog\_ (add a space at the end) will find any word ending with dog\_ i.e. hotdog\_, birddog\_ etc.

\_dog (add space at beginning) will find any word beginning with dog i.e. \_doghouse, \_dogfish etc.

\_dog\_ (add space at the beginning and end) will only find "\_dog\_"

You can enter any phrase like "Over the trees", "Into the water" etc.

Searches are NOT case sensitive.

'''

import fileinput

import os

from selenium.webdriver.support.ui import WebDriverWait

import icecream

import random

from urllib3.exceptions import HTTPError as BaseHTTPError

from icecream import ic

import re #regex

from selenium.webdriver.common.action\_chains import ActionChains

import requests

from selenium import webdriver

from requests.exceptions import InvalidSchema

from nltk.stem import PorterStemmer

from nltk.tokenize import sent\_tokenize, word\_tokenize

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

from selenium import webdriver

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from selenium.webdriver.common.action\_chains import ActionChains

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from selenium.common.exceptions import NoSuchElementException

from selenium.common.exceptions import NoSuchFrameException

from selenium.common.exceptions import StaleElementReferenceException

from selenium.common.exceptions import InvalidArgumentException

from selenium.common.exceptions import WebDriverException

from selenium.common.exceptions import NoAlertPresentException

from selenium.webdriver.remote.webdriver import WebElement

from bs4 import BeautifulSoup as bs

from newspaper import Article

from selenium.webdriver import ActionChains

import urllib

import numpy as np

import pandas as pd

import requests

from bs4 import BeautifulSoup

from pprint import pprint

from newspaper import Article

from newspaper import Source

from newspaper import news\_pool

from htmldom import htmldom

from selenium.webdriver.support.ui import Select

import urllib.request

from bs4 import BeautifulSoup

import csv

import urllib.request, urllib.error, urllib.parse, ssl

import nltk

from nltk.tokenize import word\_tokenize, sent\_tokenize

import Levenshtein

from sklearn.metrics.pairwise import cosine\_similarity

from sklearn.feature\_extraction.text import CountVectorizer

from nltk.corpus import stopwords

import csv

#selenium stellarium Variables for space object tracking

'''# todo

1 - Main controls

2 - Time Jump

3 - Selection

4 - Stars

5 - Planets

6 - Atmosphere

Grids

8 - DSO

Landscape

10 - Starlore

Actions

11 - Scripts

10 - Location

12 - Projection

'''

#\* time tab

maincontrols\_tab\_dateyear\_id = "date\_year"

maincontrols\_tab\_datemonth\_id = "date\_month"

maincontrols\_tab\_dateday\_id = "date\_day"

maincontrols\_tab\_time\_hour\_id = "time\_hour"

maincontrols\_tab\_time\_minute\_id = "time\_minute"

maincontrols\_tab\_time\_second\_id = "time\_second"

#\* field of view

maincontrols\_tab\_fieldofview\_id = "view\_fov"

#\* location tab

location\_tab\_id = "ui-id-21"

location\_tab\_latitude\_id = "loc\_latitude"

location\_tab\_longitude\_id = "loc\_longitude"

location\_tab\_altitude\_id = "loc\_altitude"

location\_tab\_name\_id = "loc\_name"

#\* time jump tab

timejump\_tab\_id = "ui-id-10"

location\_tab\_latitude\_id = "loc\_latitude"

location\_tab\_longitude\_id = "loc\_longitude"

location\_tab\_altitude\_id = "loc\_altitude"

location\_tab\_name\_id = "loc\_name"

#selenium variable front/ends section

atmosphere\_tab\_id = "ui-id-6"

lightpollution\_class = "stelaction" #click this to enable real light polution sim from location

#\* actions tab

actions\_tab\_class = "checkableaction"

actions\_tab\_datetime\_value = "actionShow\_DateTime\_Window\_Global"

actions\_button\_id = "bt\_doaction" #click this to enable the above datetime value window.

actions\_optionvalue\_update = "actionRun\_AstroCalc\_UpdatePositions"

actions\_optionvalue\_screenshot = "actionSave\_Screenshot\_Global"

actions\_optionvalue\_show\_starlink = "actionShow\_Satellite\_Group\_starlink"

actions\_optionvalue\_show\_militarysats = "actionShow\_Satellite\_Group\_military"

addict = {0:"Date submitted",

1:"Date/Time of Event",

2:"Short Description",

3:"Location of Event",

4:"Long Description",

5:"Attachments"}

nearest\_city\_field\_x = "/html/body/center/div[2]/form/table/tbody/tr[8]/td[2]/input"

event\_stateusa\_x = '//\*[@id="id\_selbox\_state\_id"]'

description\_x\_front = "/html/body/center/form/table/tbody/tr[" #starts at 3

date\_sub\_x\_front = "/html/body/center/form/table/tbody/tr["#starts at 3

date\_time\_front = "/html/body/center/form/table/tbody/tr[" #starts at 3

sdescr\_front = "/html/body/center/form/table/tbody/tr[" # starts at 3

location\_of\_event\_front = "/html/body/center/form/table/tbody/tr[" #starts at 3

#todo could be attachments... idk if you want them but here is the xpath

attachments\_front = "/html/body/center/form/table/tbody/tr["# starts at 3

description\_css\_front = "body > center:nth-child(1) > form:nth-child(4) > table:nth-child(1) > tbody:nth-child(1) > tr:nth-child("

description\_css\_end = ") > td:nth-child(6) > input:nth-child(1)"

#body > center:nth-child(1) > form:nth-child(4) > table:nth-child(1) > tbody:nth-child(1) > tr:nth-child(3) > td:nth-child(6) > input:nth-child(1)

description\_x\_end = "]/td[6]/input"

date\_sub\_x\_end = "]/td[2]"

date\_time\_end = "]/td[3]"

sdescr\_end = "]/td[4]"

location\_of\_event\_end = "]/td[5]"

attachments\_end = "]/td[7]"

keywordfield\_x = "/html/body/center/div[2]/form/table/tbody/tr[9]/td[2]/input[1]"

date\_of\_event\_x = "/html/body/center/div[2]/form/table/tbody/tr[2]/td[2]/input[5]"

submit\_button\_x = "/html/body/center/div[2]/form/input[2]"

pagerows = 50 # number of rows allowed or visible on the MUFON screen

timestamp\_string = str(datetime.now())

popup\_text = "/html/body/center/table/tbody/tr[2]/td" # this holds the popup window's description text (long)

#popup\_x\_text = "/html/body/center/table/tbody"

popup\_x\_text = "/html/body/center/table/tbody/tr[2]/td"

popup\_text\_x\_button = "/html/body/center/form/table/tbody/tr[3]/td[6]/input"

casesfound\_css = "body > center:nth-child(1) > form:nth-child(4) > table:nth-child(1) > tbody:nth-child(1) > tr:nth-child(1) > td:nth-child(1) > font:nth-child(1)"

#xfiles section

xfiles\_original\_airdate\_x\_front = "/html/body/font[3]/table[1]/tbody/tr[" #starts at 2

xfiles\_original\_airdate\_x\_end = "]/td[3]"

#ancient aliens section

ancient\_aliens\_original\_airdate\_x\_front = ""

ancient\_aliens\_original\_airdate\_x\_front = ""

#for use with selenium automated data extraction

topics = ['low in the sky','line formation','bright','green','blue','fbi was there','swat','I saw police','Angels','lights in a line','strange light','cave'] # todo iterate the entire code through this topics array.

buzzwords = ['God','Jesus','Heaven'] # these will be eliminated from results as they are automatically biased and unlikely to be credible.

odd\_words = ['spouse']

Main\_Cities = ['','Austin','Miami','Reno','Tinley Park','New York','Houston','El Paso']

Main\_HotSpots = {'Tinley Park':['10/2004','10/2005']}

Primary\_States = ['Utah','Idaho','Montana','West Virginia','New Hampshire','Maine']

Cities\_to\_avoid = ['Chicago']

#todo add tags for data runs that can indicate tests for things like starlink satelites.

another\_search\_x = "/html/body/center/form/input[1]"

#functions section

**def** scrollment():

ic()

'''

readme

scrolls down one page length to see invisible content.

'''

driver.execute\_script("window.scrollTo(0,document.body.scrollHeight-1)")

**def** shifty(bottom,top):

ic()

#ic(bottom)

#ic(top)

temp = 0 # init

if bottom > top:

temp = top

top = bottom

bottom = temp

#makes lifelike shifts

x = random.uniform(top,top+4) #init higher so that the while loop kicks in.

while x < top and x > bottom:

x = random.random()\*float(random.uniform(0,top))

x = round(x)

time.sleep(x)

#ic(x)#debug

**def** xpathcreator(frontend,iterator,backend):

ic()

xpaths\_array = []

fullxpath = ""

'''for i in range(int(iter\_start),int(iter\_end)):

fullxpath = frontend + str(i) + backend

xpaths\_array.append(fullxpath)

fullxpath = frontend + str(i) + backend'''

if fullxpath == "":

fullxpath = frontend + str(iterator) + backend

return(fullxpath)

**def** xfiles\_spyder():

ic()

# this is the path to your firefox driver file

firefox\_driver = "/Users/GrahamAtWork/Documents/firefoxdriver"

driver = webdriver.Firefox()

url = "http://www.insidethex.co.uk/"

print("Beginning the program sequence: ",end="")

driver.get(url)

print(url)

#search page variables

from MUFONVARIABLES import original\_airdate\_x\_front

from MUFONVARIABLES import original\_airdate\_x\_end

elems = driver.find\_elements\_by\_tag\_name('a')

tagnumber = 2

for elem in tqdm(range(0,len(elems))):

href = elems[elem].get\_attribute('href')

if href is not None:

if href.find("transcrp") > -1:

airdate\_x = original\_airdate\_x\_front + str(tagnumber) + original\_airdate\_x\_end

try:

airdate = driver.find\_element\_by\_xpath(airdate\_x).text

except NoSuchElementException:

#ic(tagnumber)

print("no such element")

xfiles\_script = open("MUFON\_XFiles\_Episodes\_Scraped.txt",'a+') #opening text file in append mode

title = elems[elem].text

xfiles\_script.write(title)

xfiles\_script.write('\n')

this\_element = elems[elem] #select this element

this\_element.click()

time.sleep(.15)

xfiles\_script.write(airdate)

xfiles\_script.write('\n')

#find all p tagged elements

paragraphs = driver.find\_elements\_by\_tag\_name('p')

p\_counter = 0 #init

for paragraph in paragraphs:

first\_paragraph = 3 # first dial. paragraph

p\_counter+=1 #which paragraph is it on

if p\_counter >= first\_paragraph:

p\_text = paragraph.text

xfiles\_script.write(p\_text)

driver.back()

time.sleep(random.uniform(1,4))

elems = driver.find\_elements\_by\_tag\_name('a') #refresh

xfiles\_script.write('\n')

tagnumber+=1

#ic(title)

print("completed")

xfiles\_script.close()

driver.quit()

**def** ancient\_aliens\_spyder():

ic()

#extract any transcripts you can from this website for ancient aliens

# this is the path to your firefox driver file

firefox\_driver = "/Users/GrahamAtWork/Documents/firefoxdriver"

driver = webdriver.Firefox()

url = "https://subslikescript.com/series/Ancient\_Aliens-1643266"

print("Beginning the program sequence: ",end="")

driver.get(url)

#search page variables

#from MUFONVARIABLES import ancient\_aliens\_original\_airdate\_x\_front

#from MUFONVARIABLES import ancient\_aliens\_original\_airdate\_x\_end

elems = driver.find\_elements\_by\_tag\_name('a')

tagnumber = 2

for elem in tqdm(range(0,len(elems))):

href = elems[elem].get\_attribute('href')

if href is not None:

if href.find("/episode") > -1: # if the link is a link to an episode's transcript

#airdate\_x = original\_airdate\_x\_front + str(tagnumber) + original\_airdate\_x\_end

'''try:

airdate = driver.find\_element\_by\_xpath(airdate\_x).text

except NoSuchElementException:

#ic(tagnumber)'''

tv\_show\_script = open("MUFON\_Ancient\_Aliens\_Episodes\_Scraped.txt",'a+') #opening text file in append mode

this\_element = elems[elem] #select this element

this\_element.click()

time.sleep(random.uniform(1,4))

# now we are in the page that contains the transcript.

title = driver.find\_element\_by\_tag\_name('h1')

'''

# the title line above finds the upper heading as shown below

# Ancient Aliens (2009–…): Season 1, Episode 1 - The Evidence - full transcript

'''

real\_title = title.text # extract heading

tv\_show\_script.write(real\_title)

tv\_show\_script.write('\n')

real\_title = "" # clear up space

title = "" # clear up space

#tv\_show\_script.write(airdate)

#tv\_show\_script.write('\n')

#find all p tagged elements

transcript = driver.find\_element\_by\_class\_name("full-script") # the transc. object

transcript\_text = transcript.text

tv\_show\_script.write(transcript\_text)

driver.back()

time.sleep(random.uniform(1,4))

elems = driver.find\_elements\_by\_tag\_name('a') #refresh

tv\_show\_script.write('\n')

tagnumber+=1

#ic(real\_title)

print("completed")

tv\_show\_script.close()

driver.quit()

'''

This section of the code has been adapted from the source below:

Radečić, D. (2019, October 30). Calculating String Similarity in Python. Medium. https://towardsdatascience.com/calculating-string-similarity-in-python-276e18a7d33a

'''

**def** cosine\_sim\_vectors(vec1,vec2):

'''

This function has been reproduced from the source below:

Radečić, D. (2019, October 30). Calculating String Similarity in Python. Medium. https://towardsdatascience.com/calculating-string-similarity-in-python-276e18a7d33a

'''

ic()

vec1 = vec1.reshape(1,-1)

vec2 = vec2.reshape(1,-1)

return cosine\_similarity(vec1,vec2)[0][0]

**def** clean\_string(text):

'''

This code has been adapted from the source below:

Radečić, D. (2019, October 30). Calculating String Similarity in Python. Medium. https://towardsdatascience.com/calculating-string-similarity-in-python-276e18a7d33a

'''

ic()

try:

text = ''.join([word for word in text if word not in string.digits])

except TypeError:

text = "N/A"

if text != "N/A":

text = ''.join([word for word in text if word not in string.punctuation])

text = text.lower()

text = ' '.join([word for word in text.split() if word not in stopwords])

else:

text = "N/A"

return text

**def** Levenshtein\_Similarity(phrases):

'''

This code has been adapted from the source below:

Radečić, D. (2019, October 30). Calculating String Similarity in Python. Medium. https://towardsdatascience.com/calculating-string-similarity-in-python-276e18a7d33a

ADAPTATIONS

phrases is passed to the function and x number of phrases are analyzed to determine their similarity scores.

this can be iterated to find the most similar reports.

'''

ic()

from nltk.corpus import stopwords

stopwords = stopwords.words('english') # the stop words are english only

#?sentences = ['this is my mother','that is your mother','Jim has no father']

sentences = phrases # save the phrases to compare in this variable.

cleaned = list(map(clean\_string, sentences))

#ic(cleaned)

vectorizer = CountVectorizer().fit\_transform(cleaned)

vectors = vectorizer.toarray()

#ic(vectors)

csim = cosine\_similarity(vectors)

#ic(csim)

csimilarity = cosine\_sim\_vectors(vectors[0],vectors[1])

#ic(csimilarity)

return csimilarity

'''

end of adapted section from Radečić (2019)

'''

**def** my\_next\_step(driver):

ic()

driver.refresh()

time.sleep(10)

element=driver.switch\_to\_frame('iframe')

**def** wordstemmer(description\_string):

ic()

# will be used to find the stems of the words in all comments

words = description\_string.split(" ")# extract all words from the string

print(words)

ps = PorterStemmer()

for word in words:

print(ps.stem(word))

**def** transcript\_parser(transcript\_file,heading\_text):

'''

README

transcript\_file contains "file\_name.txt" for example as a string.

newfilename = "Scrubbed\_" added to the filename and will be saved as a txt file

heading\_text should contain text that recurrs in the text file as a heading that you want

to split into sep. records.

'''

ic()

heading\_text = str(heading\_text) # data validation step

newfilename = "" #init

myfile = open(transcript\_file,'r') # open transcript read only

newfilename = "MUFON\_scrubbed\_" + transcript\_file

outfile = open(newfilename,'w')

new\_item\_line = "" #init

for item in tqdm(myfile):

endofline = 0 # init

new\_item = "" # init and reset for every iteration

if item.find(heading\_text)>-1: # in the heading of the transcript

outfile.write(new\_item\_line) # save the current episode

outfile.write('\n') # new episode

outfile.write(item)

new\_item\_line = "" #reset the newline variable

else: #in the body of the transcript not the heading

if item.find("\n") > -1:

endofline = item.find("\n")

for i in range(0,endofline): #scan the line

if item[i] != '\n': #

new\_item = new\_item + item[i]

else:

#ic(new\_item)

continue

else: #there was a newline found

new\_item = item

#ic()

#ic(item)

#ic(new\_item)

new\_item\_line = new\_item\_line + new\_item

outfile.close()

myfile.close()

print("cleaned up the ",transcript\_file," file and saved in new location --> ",newfilename)

**def** initializer(download\_choice,scrape\_more):

'''

README

scrape\_more can either be Y, y, N, or True

download\_choice is to download the xfiles again as well as ancient aliens

'''

ic()

#ic(download\_choice)

#ic(scrape\_more)

if download\_choice == True:

xfiles\_spyder() # extract xfiles transcripts

ancient\_aliens\_spyder() # extract the transcripts of ancient aliens show. This will be saved to MUFON\_Ancient\_Aliens\_Episodes\_Scraped.txt

transcript\_parser("MUFON\_Ancient\_Aliens\_Episodes\_Scraped.txt","ANCIENT ALIENS")

else:

print("transcripts have been downloaded\nChecking updated MUFON Reports Database for Quality")

transcript\_parser("MUFON\_Database\_Scraped\_Reports.txt",'Long Description of Sighting Report')

#ic()

if scrape\_more == "NO" or scrape\_more == False or scrape\_more == "n" or scrape\_more == "N":

return 0 #does go here so that's working...

elif scrape\_more == 'y' or scrape\_more == 'Y' or scrape\_more == True:

print("crawling for more data")

selenium\_crawler(topics,Main\_Cities,Primary\_States,Main\_HotSpots,Cities\_to\_avoid)

else:

#ic()

print("major issues...")

**def** selenium\_crawler(topic,city,states,mainhotspots,citiestoavoid):

PRESET\_VARIABLE\_MAX = 30

'''mufon\_file = open("MUFON\_Database\_Scraped\_Reports.txt",'a+') #opening text file in append mode

mufon\_file.write(timestamp\_string)

mufon\_file.close() # save progress in the file and reopen it later'''

topic

#search page variables

from MUFONVARIABLES import nearest\_city\_field\_x

from MUFONVARIABLES import event\_stateusa\_x

from MUFONVARIABLES import keywordfield\_x

from MUFONVARIABLES import date\_of\_event\_x

from MUFONVARIABLES import submit\_button\_x

topics = [topic] # for continuity

maincities = [city] # for continuity

low = 3

high = 15 # maximum loaded (items on the page) at one time

highest = PRESET\_VARIABLE\_MAX

k = 0 # init

wallace = 0

richard = 0 #init

#?city in primarycities

# this is the path to your firefox driver file

firefox\_driver = "/Users/GrahamAtWork/Documents/firefoxdriver"

driver = webdriver.Firefox()

url = "https://mufoncms.com/cgi-bin/report\_handler.pl?req=search\_page"

print("Beginning the program sequence: ",end="")

driver.get(url)

#ic(topic)

#ic()

#critical section to fool the website. Need speed of use to mimic human and not trigger attention.

if topic != topics[0]: #skip first iteration

wallace = random.uniform(1,20)

richard = random.uniform(25,60)

print("waiting between ",wallace, " and ",richard, 'seconds...')

shifty(wallace,richard)

keywordfield = driver.find\_element\_by\_xpath(keywordfield\_x)

date\_of\_event\_field = driver.find\_element\_by\_xpath(date\_of\_event\_x)

nearest\_city\_field = driver.find\_element\_by\_xpath(nearest\_city\_field\_x)

event\_state\_field = driver.find\_element\_by\_xpath(event\_stateusa\_x)

submit\_button = driver.find\_element\_by\_xpath(submit\_button\_x)

keywordfield.send\_keys(topics[k])

shifty(1,random.uniform(2,4)) # act humanish

nearest\_city\_field.send\_keys(city)

shifty(1,2)

nearest\_city\_field.send\_keys(Keys.RETURN)

#event\_state\_field.send\_keys(state)

#shifty(2,3)

#?submit\_button.click()

shifty(1,3)

front\_elements = [date\_sub\_x\_front,date\_time\_front,sdescr\_front,location\_of\_event\_front,attachments\_front,description\_css\_front]

end\_elements = [date\_sub\_x\_end,date\_time\_end,sdescr\_end,location\_of\_event\_end,attachments\_end,description\_css\_end]

descriptions = [] #init

print("waiting till button is visible")

wait = WebDriverWait(driver, 10)

element = wait.until(EC.visibility\_of\_element\_located((By.CSS\_SELECTOR, casesfound\_css)))

descriptions = driver.find\_elements\_by\_class\_name("button") #save 52 buttons to this array to be clicked and extracted in order.

pagerows = len(descriptions) # this should equal 52 but this step prevents breakage at this point.

window\_before = driver.window\_handles[0] # identifies the id of the window with the search results for navigating back later on.

print("searching for phrase:",topic," and nearest the city:",city) # topic is the referenced "topic" from the uppermost array: "topics"

#ic()

topicstring = topic + ',' #will go in first column of csv

for k in tqdm(range(3,pagerows)):# following the xpath pattern on the page going down each the number of rows on the page.

#ic(k)

mufon\_file = open("MUFON\_Database\_Scraped\_Reports.txt",'a+')

mufon\_file.write(topicstring)

front = "" # init and reset every iteration

back = "" # init and reset every iteration

inner\_string = "" # init and reset every iteration

for index in range(1,len(front\_elements)+1): # go through [description\_x\_front,date\_sub\_x\_front,date\_time\_front,sdescr\_front,location\_of\_event\_front,attachments\_front]

#ic(index)

full\_path\_x = "" # init and reset every iteration

# generate the xpath

# in the third line down the 3 is unchanging while the 2 changes every iteration of this inner loop

front = front\_elements[index-1]

back = end\_elements[index-1]

full\_path\_x = front + str(k) + back # /html/body/center/form/table/tbody/tr[3]/td[2] for example.

#ic(full\_path\_x)

# extract the contents of the field

if index != 6: # is other column besides descr.

field\_contents = driver.find\_element\_by\_xpath(full\_path\_x).text

#ic()

# special case: long description field has a button to click first.

if index == 6: # index = 6 is a clickable button.

field\_text = "" # init and reset

field\_contents = descriptions[k] # should allow the browser to access the popup window

try:

field\_contents.click()

time.sleep(.25)

except Exception:

driver.quit()

#ic()

field\_contents = "n/a" # save nothing in the field

shifty(1,2) # pause for a moment then check field contents too see if click was successful above

if field\_contents != "n/a": #means the button was clicked

#get current window handle

p = driver.current\_window\_handle

#get first child window

window\_handles = driver.window\_handles

for w in window\_handles:

#switch to child window

if(w!=p):

driver.switch\_to.window(w)

break

shifty(.5,1)

el = driver.find\_element\_by\_tag\_name('body')

#print(el.text[0:30])

field\_text = el.text

inner\_string = inner\_string + "ø" +str(field\_text) # the delimeter is the ø (option o) symbol for this csv file.

#ic()

# if any error or any exception is flagged then this trips

p = driver.current\_window\_handle

driver.close()

driver.switch\_to\_window(window\_before)

shifty(.5,1) # patience...

descriptions = driver.find\_elements\_by\_class\_name("button") #refresh the descriptions array

else:

# all other indices

new\_field = "" #init

field\_contents2 = str(field\_contents)

fields = ""

for i in field\_contents:

if i != ',' and i != '\n': # remove all commas

fields += i

else:

fields += " "

field\_contents = fields

if field\_contents.find("\n") < len(str(field\_contents2)) and field\_contents.find("\n")>-1:

new\_field = field\_contents2[0:field\_contents2.find("\n")] + " " + field\_contents2[field\_contents2.find("\n")+1:len(field\_contents2)]

inner\_string = inner\_string + "ø" +str(new\_field) # the end of record delimiter is the end of line character

else:

# if the index is one less than length then the program is looking at the last field (attachments field)

if index < len(front\_elements)-1:

inner\_string = inner\_string + "ø" +str(field\_contents2) # the delimeter is the ø symbol for this csv file.

else:

inner\_string = inner\_string + "ø" +str(field\_contents2) # the end of record delimiter is the end of line character

#ic()

#print(inner\_string) #debugging in the terminal window

# now inner\_string contains the data from the row in string format

# it can now be saved to the file as a new line

#inner\_string += "ø" + timestamp\_string

#ic(inner\_string)

fields = ""

for i in inner\_string:

if i != ',' and i != '\n': # remove all commas

fields += i

else:

fields += " "

#now with no commas sub commas in place of delimiters

#ic(fields)

if fields.find(",")>-1 and len(fields)<=len(inner\_string):

#ic()

print("success")

fields\_ofgrain = ""

for i in fields:

if i != 'ø': # remove all commas

fields\_ofgrain += i # add to fields if not a ø

else:#ic(i)

fields\_ofgrain += ","

mufon\_file.writelines(fields\_ofgrain[1:]) # save the row

mufon\_file.write('\n')

time.sleep(10)

#shifty(1,3) #add randomness like a human

mufon\_file.close()

driver.close()

**def** corpora\_sherlock(text\_file,delimeter\_used):

'''

README

takes in a text file and creates a csv file

(textfile,delimeter\_used) for this case will be 'ø' this is option + o

'''

ic()

newfilename = "" #init

myfile = open(text\_file,'r') # open transcript read only

newfilename = "MUFON\_Sherlocked\_" + text\_file

outfile = open(newfilename,'w')

myfile = myfile.read()

print(myfile[0:200])

fieldtext = "" #init

fields = "" #init

linetext = "" #init

counter = 0 # init

columns = 12

rows = myfile.count("øøø") # how many rows of data will there be?

#ic(rows)

#initialize matrix

reports = [ '' for i in range(rows)]

matrix\_index = 0 # will be the record id in the matrix

column\_id = 0 # init

reportsline = [ '' for i in range(columns)]# holds each row of the matrix

for i in tqdm(range(0,len(fields))):

if fields[i] == 'ø': # this is a new field "2021-03-15"

#reportsline[column\_id] = [linetext] # add these into the matrix later

column\_id += 1 # this column id will increment by 1

counter+=1 # if goes above 3 there is a new record.

if counter == 3:

column\_id = 0 # go back to the first column

reports[matrix\_index] = reportsline

reportsline = [] #reset

matrix\_index += 1 # go to next row of the matrix

#ic(linetext)

outfile.write(linetext)

linetext = ""

counter = 0 #reset

else:

linetext = linetext + ',' + fieldtext

fieldtext = "" #reset the field text

else:

counter = 0 # reset counter that counts the number of ø it sees in a row.

fieldtext += fields[i]

#ic(reports)

#print(reports[10][1])

#print(reports[23][4])

#ic(myfile[i])

outfile.close()

#\*topics = 'lights'

city = 'Orlando'

#\*selenium\_crawler(topics,city,Primary\_States,Main\_HotSpots,Cities\_to\_avoid)

#selenium\_crawler(topics,Main\_Cities,Primary\_States,Main\_HotSpots,Cities\_to\_avoid)

#corpora\_sherlock("Scrubbed\_MUFON\_Database\_Scraped\_Reports.txt",'ø')

'''

Cape Canaveral Space Force Station

WASHINGTON — SpaceX successfully launched another set of Starlink satellites Feb. 4 in the first of back-to-back Falcon 9 launches scheduled from Cape Canaveral. The Falcon 9 rocket lifted off at 1:19 a.m. Eastern from Space Launch Complex 40 at Cape Canaveral Space Force Station.

'''

#!unused function

**def** convert\_csv\_to\_dict(filename):

ic()

#adapted from reading and writing CSV Files in Python Website

import csv

csv\_reader = csv.DictReader(csv\_file)

line\_count = 0

for row in csv\_reader:

if line\_count == 0:

print(**f**'Column names are {", ".join(row)}')

line\_count += 1

#print(f'\t{row["city"]} , {row["state\_name"]} is located at {row["lat"]} {row["lng"]}')

line\_count += 1

print(**f**'Processed {line\_count} lines.')

#ic(csv\_reader)

return csv\_reader

**def** convert\_csv\_to\_pandas(filename,indexcolumn):

ic()

#df = pandas.read\_csv(filename)

df = pd.read\_csv(filename, index\_col=indexcolumn)

return df

**def** write\_to\_csv\_from\_pandas(df\_name,new\_file\_name):

ic()

import pandas

#ic("NOT COMPLETED YET... CHECK THIS FUNCTION BEFORE IMPLEMENTATION")

df = pandas.read\_csv('hrdata.csv',

index\_col='Employee',

parse\_dates=['Hired'],

header=0,

names=['Employee', 'Hired', 'Salary', 'Sick Days'])

newfilename += ".csv"

df.to\_csv(newfilename)

#convert csv file to pandas df

#todofilename = "MUFON\_uscities.csv"

#todocities\_df = convert\_csv\_to\_pandas(filename)

# saved with cityname as index column

#todo this function is working properly and gives coordinates of cities if available.

**def** finding\_nemopolis\_old(city\_name,state\_name,cities\_df,cities2\_df):

ic()

#test dictionary

#print("testing Austin,TX in df: ",end="")

# Check whether city is in the dictionary and print feedback

#first check main one

from operator import indexOf

Mode2 = False #state id mode

if len(state\_name) == 2:

Mode2 = True

#ic()

null\_array = ['','','','']

if city\_name not in cities\_df['city\_ascii']:

if city\_name not in cities2\_df['city\_ascii']:

try:

null\_array = ['','','','']

#ic(cities2\_df[city\_name])

except KeyError:

null\_array = [city\_name,state\_name,'n/a','n/a']

return null\_array

return null\_array

else:

cities\_df = cities2\_df # go with the second option

null\_array = [city\_name,state\_name,'n/a','n/a']

#then check mufon 2014 us cities with no state id and find result

#ic()

if Mode2 == False: #state names

line\_count = 0 #init

location\_data = [] #init

null\_array = [city\_name,state\_name,'n/a','n/a']

for row in tqdm(range(0,len(cities\_df['city\_ascii']))):

if cities\_df['city\_ascii'][line\_count] == city\_name and cities\_df['state\_name'][line\_count] == state\_name:

#print(cities\_df['city\_ascii'][line\_count],", ",cities\_df['state\_name'][line\_count]," located: ",cities\_df['lat'][line\_count]," ",cities\_df['lng'][line\_count])

location\_data = [cities\_df['city\_ascii'][line\_count],cities\_df['state\_name'][line\_count],cities\_df['lat'][line\_count],cities\_df['lng'][line\_count]]

#ic("returning")

#ic(location\_data)

return location\_data

else:

#ic("else")

location\_data = [] # remains null

if row > len(cities\_df['city\_ascii']):

location\_data = geopy\_city\_to\_coordinates[city\_name,state\_name]

return location\_data

line\_count += 1

else:

line\_count = 0 #init

location\_data = [] #init

null\_array = [city\_name,state\_name,'n/a','n/a']

loc = np.nonzero(np.in1d(cities\_df['city\_ascii'], city\_name))[0]

loc = np.nonzero(np.in1d(cities\_df['city\_ascii'], city\_name))[0]

iterator = 0 #init

for row in tqdm(range(0,len(loc))):

intersection\_x = []

intersection\_y = []

var0 = []

var1 = []

x = np.array(cities\_df['city\_ascii'])

y = np.array(cities\_df['state\_id'])

intersection\_x = np.where(x == city\_name)

intersection\_y = np.where(y == state\_name)

intersection = np.intersect1d(intersection\_x,intersection\_y)

intersection\_bool = np.in1d(intersection, intersection\_y)

resultant = np.nonzero(np.in1d(intersection, intersection\_y))

var0 = intersection\_y

if len(intersection)>0:

var1 = np.where(var0 == intersection[0])

#print(var1[1][0])

truevar = var1[1][0]

var = intersection[0]

#print(intersection[0], " and ",intersection\_y[0][truevar])

city\_loc = intersection[0]

state\_loc = intersection\_y[0][truevar]

location\_data = [cities\_df['city\_ascii'][city\_loc],cities\_df['state\_id'][city\_loc],cities\_df['lat'][city\_loc],cities\_df['lng'][city\_loc]]

return location\_data

else:

#ic()

location\_data = geopy\_city\_to\_coordinates[city\_name,state\_name]

return location\_data

return null\_array # no coordinates found.

#todo this function works and finds distance between two points given in the df format returned by finding\_nemopolis

#todo: ['Austin', 'Texas', 30.3004, -97.7522]

**def** chasm\_between(point1,point2):

ic()

import geopy.distance

from geopy import distance

coords\_1 = (point1[2], point1[3])

coords\_2 = (point2[2], point2[3])

dist = geopy.distance.distance(coords\_1, coords\_2).mi

return dist

#todo julian date converter (complete)

**def** tojulian(day,month,year):

'''

README

dd/mm/yyyy

'''

ic()

#\* https://stackoverflow.com/questions/13943062/extract-day-of-year-and-julian-day-from-a-string-date

import datetime

import jdcal

fmt = '%m.%d.%Y'

s = str(month) + "." + str(day) + "." + str(year)

dt = datetime.datetime.strptime(s, fmt)

julian\_day = sum(jdcal.gcal2jd(dt.year, dt.month, dt.day))

return julian\_day

from geopy.extra.rate\_limiter import RateLimiter

from geopy.geocoders import Nominatim

from geopy.distance import geodesic

**def** city\_finder(city\_name,state\_name):

#coordinates data provided by https://simplemaps.com/data/us-cities.

#see their website for more details.

#finds coordinates of city

'''geolocator = Nominatim(user\_agent="mufon\_scanner")

location = geolocator.geocode("175 5th Avenue NYC")

print(location.address)

print((location.latitude, location.longitude))

print(location.raw)

#{'place\_id': '9167009604', 'type': 'attraction', ...}

#from geopy.distance import geodesic

newport\_ri = (41.49008, -71.312796)

cleveland\_oh = (41.499498, -81.695391)

print(geodes#ic(newport\_ri, cleveland\_oh).miles)

#538.390445368

df = pd.DataFrame({'name': ['paris', 'berlin', 'london']})

#from geopy.geocoders import Nominatim

geolocator = Nominatim(user\_agent="specify\_your\_app\_name\_here")

#from geopy.extra.rate\_limiter import RateLimiter

geocode = RateLimiter(geolocator.geocode, min\_delay\_seconds=1) #keeps my ping within respectful range

tqdm.pandas()

df['location'] = df['name'].progress\_apply(geocode)

df['point'] = df['location'].apply(lambda loc: tuple(loc.point) if loc else None)

import concurrent.futures

geolocator = OpenMapQuest(api\_key="...")

geocode = RateLimiter(geolocator.geocode, min\_delay\_seconds=1/20)

with concurrent.futures.ThreadPoolExecutor() as e:

locations = list(e.map(geocode, search))

'''

**def** city\_extraction(string):

ic()

string\_spaces = 0

s\_iters = []

string\_commas = 0

c\_iters = []

iterator = 0

for i in string:

if i == ' ':

string\_spaces +=1

s\_iters.append(iterator)

if i == ',':

string\_commas +=1

c\_iters.append(iterator)

iterator += 1

#blinding,2021-03-13,2010-08-14 9:15PM,it was a bright blue light flying over head,Bogalusa LA US,,Long Description of Sighting Report Driving home at dark we’re on a new road that boarded Ft Knox KY. Three huge brights started descending towards us. I thought it was an airplane though all we could see were these blinding lights. I said that I thought it was going to land on us. Then it was stationary over the road. Two more light turned on on either side of the three. Then as in one second they simultaneously darted down to the side I’d the road and all lights and evidence of it was gone. In the daytime I looked at spot where they went down and it was a depression of weeds and tree saplings. It bordered on Ft Know property line. It wasn’t until years later I realized there was no sound. In fact I felt numb after seeing it. Content-type: text/html

#count commas

commas = 0 # init

spaces = 0 # init

end\_of\_loc = string.find(',')

if end\_of\_loc == -1:

end\_of\_loc = len(string)

mode = 2

loc\_string = "" # holds Bogalusa LA US

iterator = 0

if mode == 1:

#ic(mode)

for i in string:

if iterator < end\_of\_loc:

if i == ",":

commas+=1

if commas == 4:

if i != ' ':

loc\_string += str(i)

elif i == ' ' and spaces < 1:

loc\_string += ','

spaces += 1

#ic(spaces)

#ic(i)

else:

#ic(spaces)

#ic(i)

continue

if spaces == 5:

return loc\_string #exit with the string saved

iterator+=1

else:

return loc\_string[1:] #returns city,Statecode (i.e. Austin,TX)

if mode == 2:

#ic(mode)

iterator = -1 # init one lower to compensate for line beneath for i in string below.

for i in string:

iterator+=1

nothing = 0

if iterator in c\_iters:

nothing = 1

#comma was in string

#ic('comma found in mode 2')

else:

continue

if iterator in s\_iters:

loc\_string += str(',')

if iterator in s\_iters and iterator < len(string):

#space was in string and we are on the last space before state code

loc\_string += str(',')

else:

loc\_string += str(i)

**def** writetofile(text,filename):

ic()

outfile = open(filename,'w+')

outfile.write(text)

outfile.close()

**def** city\_cleaner(filename):

ic()

csv\_data = pd.read\_csv(filename)

#csv\_data.head()

new\_locations = []

for location in csv\_data['Location of Event']:

new\_location = city\_extraction(location)

#ic(new\_location)

new\_locations.append(new\_location)

df = pd.read\_csv(filename)

df["Optimized Location"] = new\_locations

df.to\_csv(filename, index=False)

**def** plotly\_project():

ic()

#Final Project - RSAN 130 - Plotly Visualizations

import re

import string

import time

import os.path

import numpy as nmp

import numpy as np

import pandas as pd

import matplotlib

from tqdm import tqdm

from tqdm import tqdm\_gui

import plotly.express as px

import plotly.graph\_objects as go

import datetime

import csv

#! multi plot

import csv

import plotly.graph\_objects as go

from plotly.subplots import make\_subplots

ThisDay = 1 #! this holds the day of the week for each column

fig = make\_subplots(rows=7, cols=6, start\_cell="top-left",x\_title='Your master x-title',

y\_title='Your master y-title',

subplot\_titles = [**f**'Title\_{i+1}\_{j+1}' for i in range(10) for j in range(2)])

#!init for the figure with 7 rows and 6 columns

row = 1

col = 1

groupsize = 100 #how large the buckets are

data = list(csv.reader(open("nycpartyupd.csv")))

lat = pd.read\_csv("nycpartyupd.csv", index\_col="Latitude")

#!lat holds the latitudes column

long = pd.read\_csv("nycpartyupd.csv", index\_col="Longitude")

#!long holds the longitudes column

barlat = list(csv.reader(open("bar\_locations.csv")))

BXArray = []#\* holds the x coordinate arrays for the bars

BYArray = []#\* holds the y coordinate arrays for the bars

BCArray = []#\* holds the y coordinate arrays for the number of calls at these bars

i = 0

j = 0

for i in tqdm(range(0,len(barlat))): #saving these values for graphing later

# go down list of bars

BXArray.append(barlat[i][4])

BYArray.append(barlat[i][5])

BCArray.append(barlat[i][6])

#? 4 and 5 are coordinates and 6 is the number of calls to police

#!these arrays are for later sections in the code

CombArrays=[]#! this will hold the comb scores for each iteration of the whileloops below

XArrays = []

YArrays = []

XAxis\_Max = 0 # this variable tells what the maximum of the x axis coordinates is

XAxis\_Min = 0 # this variable tells what the minimum of the x axis coordinates is

YAxis\_Max = 0 # this variable tells what the maximum of the y axis coordinates is

YAxis\_Min = 0 # this variable tells what the minimum of the y axis coordinates is

#! more storage arrays

xbounds = [] #holds tuples with upper and lower bounds for the x axis on muliple graphs

ybounds = []

TimeArrays = [] #will hold event times for analysis

#!end

bucket = 0 #size limiter

while row<8:#! through the days

while col<7:#!through the locations

XArray = []#\* holds the x coordinate arrays

YArray = []#\* holds the y coordinate arrays

TimeArray = []#\* holds time of day when event occurred

CombArray = []#\* holds the combined rrt score for this day

#! this row will have six plots side by side.

day = 0 #? holds a certain row in a certain column

record = 0 #? this holds length of the whole list of records

#? https://www.geeksforgeeks.org/indexing-and-selecting-data-with-pandas/

#! the relevant indeces are: [0][] is the records column and [][4] is the Day of Week Column

rec = "" #! init

#?obtaining the coordinates columns

for record in range(1,len(data)):#for all of the records...

#! rec can now be used to search

rec = int(record)

#print("rec->",rec," day=",day," data[rec][2]=",data[int(rec)][2])

#print("record:",record)

day = data[rec][4]# ! day will hold the day of the week

if int(day) == int(ThisDay): #! if the day matches the current day of the week then save the

#!coordinates to the X and Y arrays to be graphed are stored in XArray and YArray

if bucket == groupsize:#? if the current bucket is the same size as groupsize then move forward

if data[rec][2] != 0 and data[rec][2] != "0" and data[rec][3] != 0 and data[rec][3] != "0": #and grouper==groupsize:

XArray.append(data[rec][2]) #! saves the latitude of that record to the XArray to be graphed

YArray.append(data[rec][3]) #! saves the longitude of that record to the YArray to be graphed

XArrays.append(data[rec][2]) #! saves the latitude of that record to the XArray to be graphed

YArrays.append(data[rec][3]) #! saves the longitude of that record to the YArray to be graphed

CombArray.append(data[rec][30]) #! saves the value of COMB\_AVGRRT combined average reponse time

CombArrays.append(data[rec][30]) #! saves the value of COMB\_AVGRRT combined average reponse time

TimeArray.append(data[rec][13]) #! saves time of event to array: TimeArray

TimeArrays.append(data[rec][13]) #! saves time of event to array: TimeArrays

if float(data[rec][2])>float(XAxis\_Max): #? these if statements show the bounds of the x and y coordinates

XAxis\_Max = data[rec][2]

if float(data[rec][3])>float(YAxis\_Max):

YAxis\_Max = data[rec][3]

print("[",YAxis\_Max,"],",record,end=",")# todo: erase this debug step

if float(data[rec][3])<float(YAxis\_Min):

YAxis\_Min = data[rec][3]

if float(data[rec][2])<float(XAxis\_Min):

XAxis\_Min = data[rec][2]

bucket = 0 #! reset the bucket value

else:

bucket+=1

continue

else:

#print("\*",end="") # debugging tool

continue

#!save the bounds

xbounds.append([XAxis\_Min,XAxis\_Max])

ybounds.append([YAxis\_Min,YAxis\_Max])

#!Now the X and Y arrays are plottable and a trace can be added to the figure.

fig.add\_trace(go.Scatter(x=XArray, y=YArray,mode="markers"), row=row, col=col)

fig.add\_trace(go.Scatter(x=BXArray, y=BYArray,fillcolor = '#ff0000',mode="markers",marker\_symbol='x'), row=row, col=col)

print("||||",col,",",row,"|||| ",end="") #! for visual reference

col+=1

col=1

ThisDay+=1 #! Goes to the next day of the week

print("")

row+=1

fig.update\_layout(title\_text="New York Noise Complaints by Geography")

fig.update\_yaxes(automargin=True)

fig2 = px.density\_heatmap(x=XArrays, y=YArrays) #! holds heatmap comparing the x coordinates to the y coordinates

fig2.show()

# Update xaxis properties

'''for i in range(0,7):

for j in range(0,6):

title = "xaxis title"

ytitle = "yaxis title"

print('fig.update\_xaxes(title\_text="',title,'", row=',i,' col=',j,')')

print('fig.update\_yaxes(title\_text="',ytitle,'", row=',i,' col=',j,')')

'''

fig.update\_xaxes(title\_text="xaxis 1 title", row=1, col=1)

# Update title and height

fig.update\_layout(title\_text="Customizing Subplot Axes", height=700)

fig.show()

**def** city\_csv\_to\_location(filename):

ic()

coordinates\_dict = {'Bogalusa/LA':['Bogalusa', 'LA', 30.7812, -89.8633]}

cities\_df = convert\_csv\_to\_pandas("MUFON\_uscities.csv","city")

cities2\_df = convert\_csv\_to\_pandas("MUFON\_2014\_Cities\_updated.csv","city")

cities\_df.head()

csv\_data = pd.read\_csv(filename)

#csv\_data.head()

new\_locations = []

csv\_data\_states = []

coordinates = []

for i in range(0,len(csv\_data)):

csv\_data\_states.append(csv\_data['state\_id'][i])

#ic(csv\_data\_states)

iterator = 0

spot = ['','','','']

for location in csv\_data['city\_ascii']:

new\_location\_city = location

new\_location\_state = csv\_data\_states[iterator]

new\_location = location + "," + new\_location\_state[0:2]

spot = finding\_nemopolis(new\_location\_city,new\_location\_state[0:2],cities\_df,cities2\_df)

city\_state = location + "/" + new\_location\_state[0:2]

coordinates\_dict[city\_state] = spot

coordinates.append(spot)

#ic(new\_location)

#new\_locations.append(new\_location)

print(coordinates\_dict[location + "/" + new\_location\_state[0:2]])

iterator +=1

#now find coordinates

'''for city\_state in tqdm(new\_locations):

dory\_bull = city\_state.split (",") #splits string into array city,state to [city,state]

spot = finding\_nemopolis(dory\_bull[0],dory\_bull[1])

coordinates.append(spot)'''

return coordinates\_dict

**def** save\_dict\_as\_csvfile(dict\_name):

ic()

filename = 'MUFON\_' + str("dictionary") + '.csv'

with open(filename, 'w+') as f:

for key in dict\_name.keys():

f.write("%s,%s\n"%(key,dict\_name[key]))

#\*coordinates = city\_csv\_to\_location("citiestocheck.csv")

#\*save\_dict\_as\_csvfile(coordinates)

#writetofile(coordinates,"MUFON\_dictionary\_cities\_locations.csv")

#ic()

#todo function to get closest "n" reports to a report (report\_i) based on the location of report\_i

**def** Jones\_Effect(point\_x,dataset):

'''

README

Q is len(dataset)/(2) round down so for len(dataset) = 1203 the Q = 1203/2 = 601.5 round down to 601

find the Q closest points by longitude

find the Q closest points by latitude

find which points occur in both sets

get their distances

point\_x = the dataframe below

Date/Time of Event date posted city state country latitude longitude Description shape duration (seconds)

10/10/49 20:30 4/27/04 san marcostx us 29.8830556 -97.9411111 cylinder 2700

Topic Date Submitted Date/Time of event Short Description Location of Event length upper Long Description Attachments705 DRIVING HOME AT DARK WE’RE ON A NEW ROAD THAT BOARDED FT KNOX KY.

blinding 3/13/21 2010-08-14 9:15PM it was a bright blue light flying over head Bogalusa LA US

'''

#will call distance functions and lat long from findingnempolis function

dataset

#todo function to get the closest "n" reports to a report (report\_i) based on date of event

#todo function to compare the text of a report A with a report B and give similarity

#todo function to generate the value of a report\_i given the similarity of their description to others in the area

#todo use geopy to find coordinates of cities mentioned in the reports and return those

#todo cities' coordinates, google rated importance, and bounding box for lat and longitude.

**def** geopy\_city\_to\_coordinates(cityname,statename):

'''

README

city name must be like "Austin"

statename must be in form "Texas"

no spaces they will be combined with space between them

'''

inputstring = cityname + " " + statename

from geopy.geocoders import Nominatim

geolocator = Nominatim(user\_agent='myapplication')

location = geolocator.geocode(inputstring)

print(location.address)

print(location.raw)

#coordinates = [0,0]

coordinates = [location.point[0],location.point[1]]

other\_city\_data = [location.raw['importance'],location.raw['boundingbox']]

returnable\_array = [coordinates,other\_city\_data] # [[lat,long],[importance,boundingbox]

return returnable\_array

'''

result: [[30.2711286, -97.7436995],[0.8694265717866669,['30.0984576', '30.5166255', '-97.9367663', '-97.5605288']]]

'''

#todo this function is working properly and gives coordinates of cities if available.

**def** finding\_nemopolis(city\_name,state\_name):

location\_data = ['']

if len(state\_name)>2:

location\_data = geopy\_city\_to\_coordinates(city\_name,state\_name)

else:

return location\_data

return location\_data # no coordinates found.

result = geopy\_city\_to\_coordinates("Austin","Texas")

var = finding\_nemopolis("Austin","Texas")

ic(var)

ic(result)

#city\_text = city\_extraction("blinding,2021-03-13,2010-08-14 9:15PM,it was a bright blue light flying over head,Bogalusa LA US,,Long Description of Sighting Report Driving home at dark we’re on a new road that boarded Ft Knox KY. Three huge brights started descending towards us. I thought it was an airplane though all we could see were these blinding lights. I said that I thought it was going to land on us. Then it was stationary over the road. Two more light turned on on either side of the three. Then as in one second they simultaneously darted down to the side I’d the road and all lights and evidence of it was gone. In the daytime I looked at spot where they went down and it was a depression of weeds and tree saplings. It bordered on Ft Know property line. It wasn’t until years later I realized there was no sound. In fact I felt numb after seeing it. Content-type: text/html")

#ic(city\_text)

'''loc\_data = finding\_nemopolis('Austin','Texas')

loc\_data2 = finding\_nemopolis('San Diego','California')

#ic(loc\_data)

#ic(loc\_data2)

distance = chasm\_between(loc\_data,loc\_data2)

#ic(distance)

#ic(loc\_data)

#ic()'''

#?city\_cleaner("MUFON\_Database\_Scraped\_Reports.csv")

#todo find coordinates for the cities

'''coordinates\_file = open("citiescoordinates.txt","w+")

coordinates\_file.write("city,state\_id\n")

coordinates\_file.close()

cities = open("citiestocheck.csv",'r')

commas = 0

newcity = ""

newstate = ""

for city in tqdm(cities):

if city != "\ufeffcity, state\_id\n":

coordinates\_file = open("citiescoordinates.txt","a+")

coordinates = "" # init and reset

for i in city:

if i == ',':

commas += 1

else:

if commas == 0:

newcity += i

else:

if i != ' ' and i != '\n':

newstate += i

else:

continue

try:

#ic(newcity)

#ic(newstate)

coordinates = finding\_nemopolis(newcity,newstate)

except Exception:

#ic(city)

nothing = 1

if coordinates != "":

coordinates\_file.write(coordinates)

newcity = ""

commas = 0 #reset

newstate = ""

coordinates\_file.close()

'''

**def** point\_finder(R\_x,R\_y):

'''

README

R\_x =

R\_y =

M\_x =

M\_y =

file headers are below

Source,Date/Time of Event,date posted,duration (seconds),city,state (ABBR),STATE(FULL),country,latitude,longitude ,comments,shape, ,,,

'''

import pandas as pd

csvfilename = "NUFORC\_CONSOLIDATED\_DB.csv"

data\_file = open('/Users/GrahamAtWork/Documents/Automate-with-Python-\_-Form-Filling-Script-/TextTester/NUFORC\_CONSOLIDATED\_DB.csv')

write\_file = open("POINTSOFINTEREST.csv","a+")

df = pd.read\_csv(csvfilename,usecols = [0,1,2,8,9])

#now parse

R\_x = 0

R\_y = 0

points\_of\_interest = []

iterator = 0 #Init

bounds = 0.5 # preset boundaries for significance

for R in df[0]:

if R == "MUFON": #source is mufon

R\_x = R[8] #get latitude for the report

iterator += 1 # what row number the report is on

for i in range(0,len(df)): # parse the other reports from NUFORC and MUFON

if i != iterator: # for all other records

lat = df.loc[i]['latitude'] #check latitude

if lat > R\_x-bounds and lat < R\_x + bounds: #if latitude is within bounds of R\_x latitude then check longitude

lng = df.loc[i]['longitude']

if lng > R\_y-bounds and lng < R\_y + bounds:

points\_of\_interest.append(df.loc[i]) # save the record as within the range

write\_file.write(points\_of\_interest)

data\_file.close()

write\_file.close()

return points\_of\_interest

results = point\_finder(30.2711286,-97.7436995)

ic(results)

1. Using the work of Brmson on Github. <http://brmlab.cz/project/brmson> [↑](#footnote-ref-1)
2. Quality in this context is defined as the number of potentially substantive records per 100 reports. For example: in 100 reports MUFON will be able to show 10% more records that are credible than last year. [↑](#footnote-ref-2)
3. Impact and Reach in this context will be defined as the number of educational corpora, videos, and courses offered by the MUFON education and research team. [↑](#footnote-ref-3)
4. If using these tools be sure to cite this source: Stellarium Remote Control Usage Citation: [Georg Zotti, Florian Schaukowitsch, and Michael Wimmer. The Skyscape Planetarium](http://www.cultureandcosmos.org/pdfs/21/CCv21_17Zotti.pdf). In Liz Henty, Bernadette Brady, Darrelyn Gunzburg, Frank Prendergast, and Fabio Silva, editors, The Marriage of Astronomy and Culture: Theory and Method in the Study of Cultural Astronomy (Papers from the 2016 SEAC Conference), volume 21 of Culture and Cosmos, pages 269–281, Lampeter, Ceredigion, Wales, Spring/Summer and Autumn/Winter 2017. Culture and Cosmos & Sophia Centre Press. [↑](#footnote-ref-4)