UFO SIGHTINGS

Ali Krakowsky

INTRODUCTION

For the final project I'm looking at UFO Sightings again. Here's a quick refresher on the data.

SUMMARY

NUFORC is the National UFO Reporting Center where the reports of UFO sightings are stored. The goal of this project is to use machine learning to see if it's possible to predict the shape of the UFO by location.

PROCESS

- Used Jupyter Notebook to pull data

- Executable path created to search for table

- Looped through each link to create the data frame

- Result = Data pulled from almost 1,000 links

```
executable_path = {'executable_path': ChromeDriverManager().install()}
browser = Browser('chrome', **executable_path, headless=False)

url = 'http://www.nuforc.org/webreports/ndxevent.html'
browser.visit(url)



====== WebDriver manager ======
Current google-chrome version is 94.0.4606
Get LATEST driver version for 94.0.4606
Driver [C:\Users\alig_\.wdm\drivers\chromedriver\win32\94.0.4606.61\chromedriver.
```

```
data = browser.find_by_css("td a")
```

```
ufo_links = [x["href"] for x in data]
```

```
browser.quit()
```

```
df_list = []
for index,i in enumerate(ufo_links):
    df = pd.read_html(i)[0]
    df_list.append(df)
    print(index)
    time.sleep(1)
```
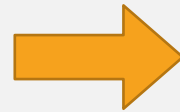
```
0
1
```

PROCESS

NUFORC Site

DATA CLEANUP

- Data frame created

- Prior to merging the csvs

  - The city and state were combined to a new column(Locations)

  - All sightings that were missing the location were dropped

  - Canadian sightings were dropped due to variation in data entry

- After cleaning- over 100,000 rows were left

```python
ufo_sightings['Location'] = ufo_sightings['City'] + ", " + ufo_sightings['State']
ufo_sightings
```

Date / Time	City	State	Shape	Duration

```python
ufo_sightings = ufo_sightings.dropna(how="all", subset=["Location"])
ufo_sightings
```

Date / Time	City	State	Shape	Duration
0 9/17/21 22:10	Laguna Hills	CA	Light	15 minutes At 10:10 pm I well

VISUAL #1

- Grouped the sightings by shapes

- Removed any sightings less than 5

- Created a pie chart with the name and percent inside the wedge

- Json.dumps- creates a trace to pass the data through as html

```
#Create the pie chart
fig = px.pie(df, values='ID', names='Shape',
            title='Shapes of UFO Sightings',
            hover_data=['ID'], labels={'ID':'Total'})
fig.update_traces(textposition='inside', textinfo='percent+label')
fig1JSON = json.dumps(fig, cls =plotly.utils.PlotlyJSONEncoder)
```
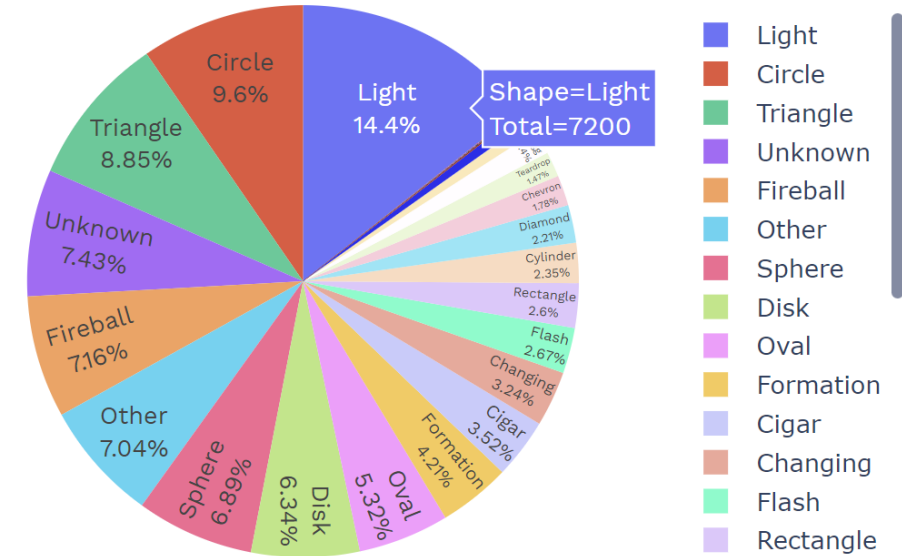
VISUAL #1

- Grouped the sightings by shapes

- Removed any sightings less than 5

- Created a pie chart with the name and percent inside the wedge

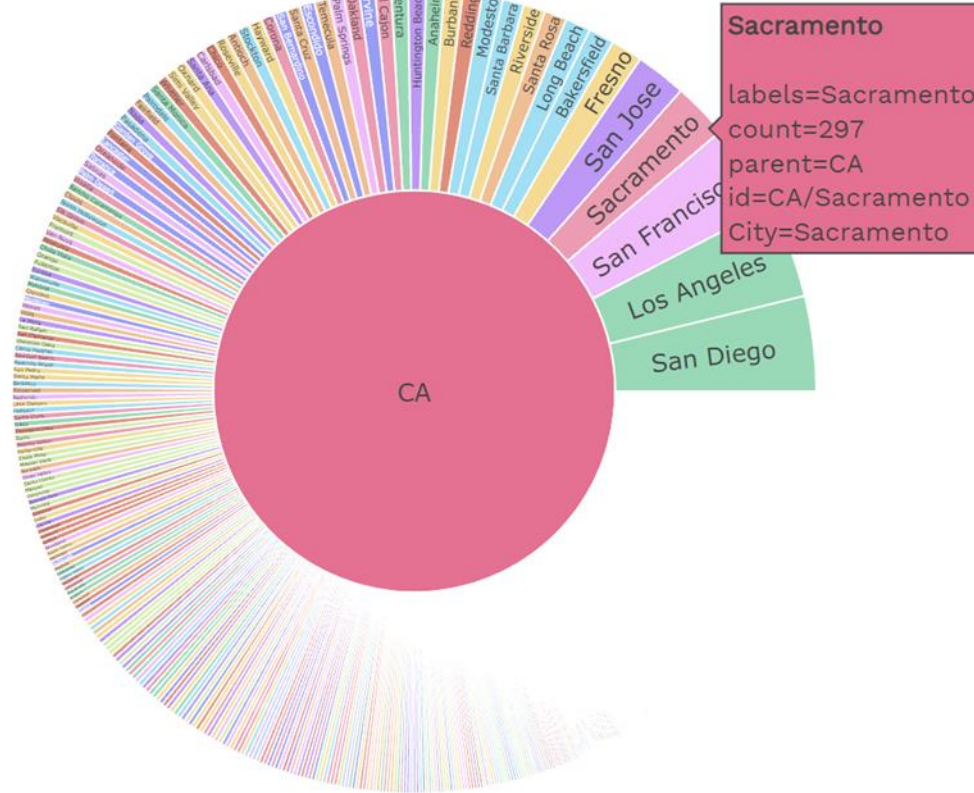- Json.dumps- creates a trace to pass the data through as html
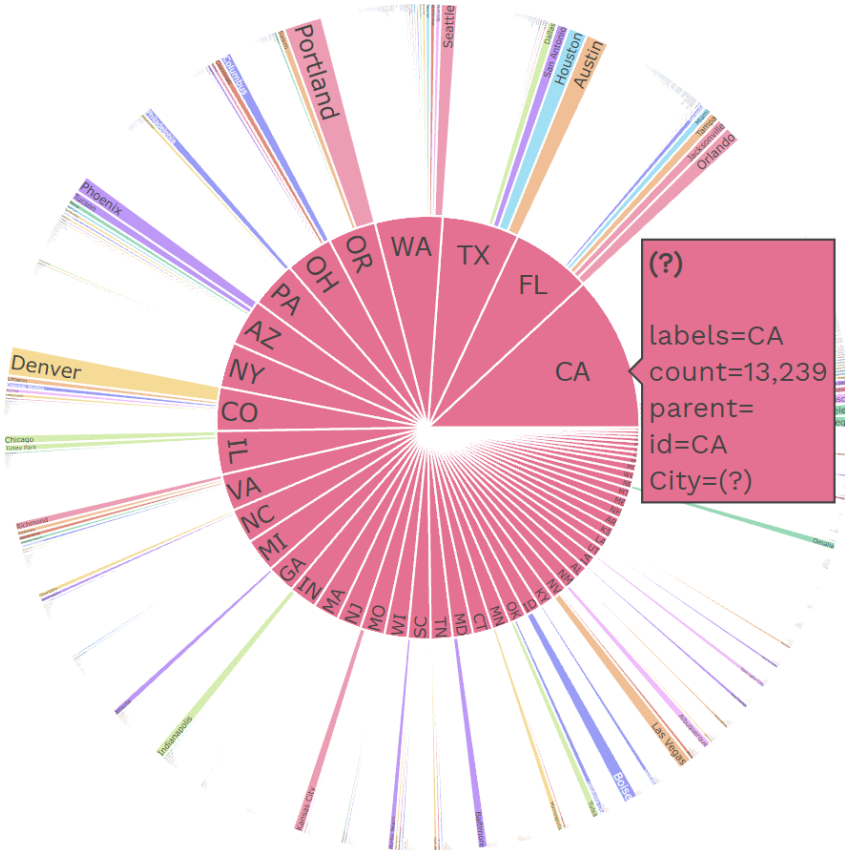

This pie chart is showing the reported shapes of the UFO sightings with the count and percentage. The most popular shape is Light. Sighting reports were created by viewers and submitted as free-text. This causes a variety in the data provided.

VISUAL #2

Sunburst Plot of Loca... ...ations

SETBACKS

- Submissions are dependent on how the user enters the data- this created a wide variety of data types that needed to be cleaned
- The varied entries limited the number of categories that could be used
- Initially running the machine learning crashed the notebook due to RAMs being used(too many columns)
- There is a very weak correlation between location and shape prediction
- Prediction is currently overfitting

DISCUSSION

The column Category was created by combing the shapes into 2 groups: Light and Dim. The shapes placed in each group was decided by me. This was to help with the accuracy of the training as having even the limited 9 shapes as the class gave poor predictions.

GOING FORWARD

- Interesting to try other categories such as date
- Require more time cleaning the data and making it uniform

QUESTIONS?