Open in Colab

Assignment 8

Weeks 10 & 11 - matplotlib & seaborn

- In this homework assignment, you will explore and analyze a public dataset of your choosing. Since this assignment is “open-ended” in nature, you are free to expand upon the requirements below. However, you must meet the minimum requirments as indicated in each section.

- The preferred method for this analysis is in a .ipynb file. Feel free to use whichever platform of your choosing.

Some data examples:

- https://www.data.gov/

- https://opendata.cityofnewyork.us/

- https://datasetsearch.research.google.com/

- https://archive.ics.uci.edu/ml/index.php

Resources:

- https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html

- https://www.oreilly.com/library/view/python-data-science/9781491912126/ch04.html

- https://www.data-to-viz.com/

Headings or comments

You are required to make use of comments, or headings for each section. You must explain what your code is doing, and the results of running your code. Act as if you were giving this assignment to your manager - you must include clear and descriptive information for each section.

You may work as a group or indivdually on this assignment.

Introduction

In this section, please describe the dataset you are using. Include a link to the source of this data. You should also provide some explanation on why you choose this dataset.

I'll continue with the dataset I've used for my project proposal. It is a dataset of statements evaluated by POLITIFACT, a website focused on fact checking statements from public figures. The data contains more than 10,000 statements from a variety of public figures from a variety of contexts. Each statement is rated for its veracity on the following scale, ranging from totally bogus to totally true: "pants-fire", "false", "barely-true", "half-true", "mostly-true", and "true".

The dataset is hosted here: https://datasets.activeloop.ai/docs/ml/datasets/liar-dataset/.

Data Exploration

Import your dataset into your .ipynb, create dataframes, and explore your data.

Include:

- Summary statistics means, medians, quartiles,
- Missing value information
- Any other relevant information about the dataset.

The hosts offer an API to query the data directly, but it requires installation of a separate library. Instead, I'll download the zipped data directly from the following site: https://www.cs.ucsb.edu/~william/data/liar_dataset.zip.

In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import os
import requests
from zipfile import ZipFile
from IPython.display import Markdown
```

In [2]:

```
# download zip file
url = 'https://www.cs.ucsb.edu/~william/data/liar_dataset.zip'

request = requests.get(url)
with open('data/liar_dataset.zip', 'wb') as fd:
    fd.write(request.content)

# unzip and see what's inside
zf = ZipFile('data/liar_dataset.zip')
zf.extractall(path = 'data')
zf.close()
zf.namelist()
```

Out[2]:

```
['README', 'test.tsv', 'train.tsv', 'valid.tsv']
```

In [3]:

```
# define list of column names
columns = [
    'id',
    'label',
    'statement',
    'subjects',
    'speaker',
    'speaker_job',
    'state',
    'party',
    'barely_true',
    'false',
    'half_true',
    'mostly_true',
    'pants_on_fire',
    'context'
]

# loop through each file and read in as a pandas dataframe
files = ['test', 'train', 'valid']

for file in files:
    filename = 'data/' + file + '.tsv'
    globals()[file] = pd.read_csv(filename, sep = '\t', names = columns)
    os.remove(filename)

    print(f'{file} - No. observations: {len(globals()[file])}')
    display(globals()[file].head())

# delete the zip files to keep our directory clean
os.remove('data/liar_dataset.zip')
os.remove('data/README')
```

```
test - No. observations: 1267
```

	id	label	statement	subjects	speaker	speaker_job	state	par
0	11972.json	true	Building a wall on the U.S.-Mexico border will...	immigration	rick-perry	Governor	Texas	republic
1	11685.json	false	Wisconsin is on pace to double the number of l...	jobs	katrina-shankland	State representative	Wisconsin	democr
2	11096.json	false	Says John McCain has done nothing to help the ...	military,veterans,voting-record	donald-trump	President-Elect	New York	republic
3	5209.json	half-true	Suzanne Bonamici supports a plan that will cut...	medicare,message-machine-2012,campaign-adverti...	rob-cornilles	consultant	Oregon	republic
4	9524.json	pants-fire	When asked by a reporter whether hes at the ce...	campaign-finance,legal-issues,campaign-adverti...	state-democratic-party-wisconsin	NaN	Wisconsin	democr

```
train - No. observations: 10240
```

	id	label	statement	subjects	speaker	speaker_job	state	party	bare
0	2635.json	false	Says the Annies List political group supports ...	abortion	dwayne-bohac	State representative	Texas	republican	
1	10540.json	half-true	When did the decline of coal start? It started...	energy,history,job-accomplishments	scott-surovell	State delegate	Virginia	democrat	
2	324.json	mostly-true	Hillary Clinton agrees with John McCain "by vo...	foreign-policy	barack-obama	President	Illinois	democrat	
3	1123.json	false	Health care reform legislation is likely to ma...	health-care	blog-posting	NaN	NaN	none	
4	9028.json	half-true	The economic turnaround started at the end of ...	economy,jobs	charlie-crist	NaN	Florida	democrat	

```
valid - No. observations: 1284
```

	id	label	statement	subjects	speaker	speaker_job	state	pa
0	12134.json	barely-true	We have less Americans working now than in the...	economy,jobs	vicky-hartzler	U.S. Representative	Missouri	republic
1	238.json	pants-fire	When Obama was sworn into office, he DID NOT u...	obama-birth-certificate,religion	chain-email	NaN	NaN	no
2	7891.json	false	Says Having organizations parading as being so...	campaign-finance,congress,taxes	earl-blumenauer	U.S. representative	Oregon	democ
3	8169.json	half-true	Says nearly half of Oregons children are poor.	poverty	jim-francesconi	Member of the State Board of Higher Education	Oregon	no
4	929.json	half-true	On attacks by Republicans that various program...	economy,stimulus	barack-obama	President	Illinois	democ

In [4]:

```
# examine the datatypes
train.dtypes
```

Out[4]:

```
id                  object
label               object
statement           object
subjects            object
speaker             object
speaker_job         object
state               object
party               object
barely_true        float64
false              float64
half_true          float64
mostly_true        float64
pants_on_fire      float64
context             object
dtype: object
```

The only numeric columns are the counts of statements under each rating. Per the README, it appears these counts are aggregated PER SPEAKER. So, rather than view summary stats directly, we'll first want to group by speaker.

In [5]:

```
train_byspeaker = train.groupby('speaker')[[
    'barely_true',
    'false',
    'half_true',
    'mostly_true',
    'pants_on_fire'
]].agg(np.mean)

display(
    train_byspeaker,
    train_byspeaker.agg(sum)
)
```

	barely_true	false	half_true	mostly_true	pants_on_fire
speaker					
18-percent-american-public	0.0	0.0	0.0	0.0	1.0
60-plus-association	2.0	0.0	0.0	0.0	1.0
AARP	0.0	0.0	0.0	1.0	0.0
Arizona-Citizens-Defense-League	0.0	1.0	1.0	0.0	0.0
Ballesteros	1.0	0.0	0.0	0.0	0.0
...	...	...	...	...	...
yvette-mcgee-brown	0.0	0.0	0.0	0.0	0.0
zack-space	0.0	0.0	0.0	0.0	1.0
zell-miller	0.0	1.0	0.0	2.0	0.0
zephyr-teachout	0.0	0.0	1.0	0.0	0.0
zoe-lofgren	0.0	0.0	0.0	1.0	0.0

2910 rows × 5 columns

```
barely_true      2040.607843
false            2284.500000
half_true        2494.931373
mostly_true      2380.921569
pants_on_fire    1010.941176
dtype: float64
```

Then, we create a column showing the sum of all statements by these folks. Then we can use that column to find the summary stats for the proportion of each type of statement per speaker.

```
In [6]: train_byspeaker['total_statements'] = train_byspeaker.apply(lambda x: sum(x), axis = 1

        for col in train_byspeaker.columns[0:-1]:
            train_byspeaker[col] = train_byspeaker[col] / train_byspeaker['total_statements']

        display(train_byspeaker.describe())
```

	barely_true	false	half_true	mostly_true	pants_on_fire	total_statements
count	2585.000000	2585.000000	2585.000000	2585.000000	2585.000000	2910.000000
mean	0.196017	0.229494	0.247526	0.235839	0.091125	3.509245
std	0.333151	0.356259	0.367378	0.362934	0.249224	14.958851
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
50%	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
75%	0.285714	0.333333	0.400000	0.400000	0.000000	2.000000
max	1.000000	1.000000	1.000000	1.000000	1.000000	473.000000

Data Wrangling

Perform data wrangling. You are free to use your best judgment here. If you are stuck, look at previous assignment.

It doesn't seem that a ton of wrangling is needed. But as an example, I'll split out the subjects column into a series of one-hot encoded values. This will be necessary when we move into modeling.

In [7]:

```
onehot_encodings = train.subjects.str.get_dummies(sep = ',')
onehot_encodings
```

Out[7]:

	10-news-tampa-bay	Alcohol	abc-news-week	abortion	afghanistan	after-the-fact	agriculture	animals	autism	bankrupt
0	0	0	0	1	0	0	0	0	0	
1	0	0	0	0	0	0	0	0	0	
2	0	0	0	0	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	
4	0	0	0	0	0	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	
10235	0	0	0	0	0	0	0	1	0	
10236	0	0	0	0	0	0	0	0	0	
10237	0	0	0	0	0	0	0	0	0	
10238	0	0	0	0	0	0	0	0	0	
10239	0	0	0	0	0	0	0	0	0	

10240 rows × 142 columns

We can see there are a LOT of unique subjects. This format might make things more difficult right now, so I'll add this in a new dataframe, keeping the original `train` df intact.

In [8]:

```
train_onehot = train.join(onehot_encodings)
train_onehot
```

Out[8]:

	id	label	statement	subjects	speaker	speaker_job	state	party
0	2635.json	false	Says the Annies List political group supports ...	abortion	dwayne-bohac	State representative	Texas	republica
1	10540.json	half-true	When did the decline of coal start? It started...	energy,history,job-accomplishments	scott-surovell	State delegate	Virginia	democra
2	324.json	mostly-true	Hillary Clinton agrees with John McCain "by vo...	foreign-policy	barack-obama	President	Illinois	democra
3	1123.json	false	Health care reform legislation is likely to ma...	health-care	blog-posting	NaN	NaN	none
4	9028.json	half-true	The economic turnaround started at the end of ...	economy,jobs	charlie-crist	NaN	Florida	democra
...	...	...	...	...	...	...	...	..
10235	5473.json	mostly-true	There are a larger number of shark attacks in ...	animals,elections	aclu-florida	NaN	Florida	none
10236	3408.json	mostly-true	Democrats have now become the party of the [At...	elections	alan-powell	NaN	Georgia	republica
10237	3959.json	half-true	Says an alternative to Social Security that op...	retirement,social-security	herman-cain	NaN	Georgia	republica
10238	2253.json	false	On lifting the U.S. Cuban embargo and allowing...	florida,foreign-policy	jeff-greene	NaN	Florida	democra

	id	label	statement	subjects	speaker	speaker_job	state	party
10239	1155.json	pants-fire	The Department of Veterans Affairs has a manua...	health-care,veterans	michael-steele	chairman of the Republican National Committee	Maryland	republican

10240 rows × 156 columns

Visualizations

The main purpose of this assignment is to practice creating various visualizations using the matplotlib and seaborn library.

Part 1:

Using matplotlib, create ***two or more plots*** that incorporate at least **5** of the following properties:

Note: these properties vary based on your data. The goal is to practice creating visualizations and modifying its properties.

- Use and change a legend position
- Change a legend font size
- Place a legend outside of the plot
- Create a single legend for all subplots
- Change the title and x/y labels
- Change the marker, line colors, and line width
- Add annotations
- Modify Axis Text Ticks/Labels
- Change size of axis Labels
- Your own choice not included above

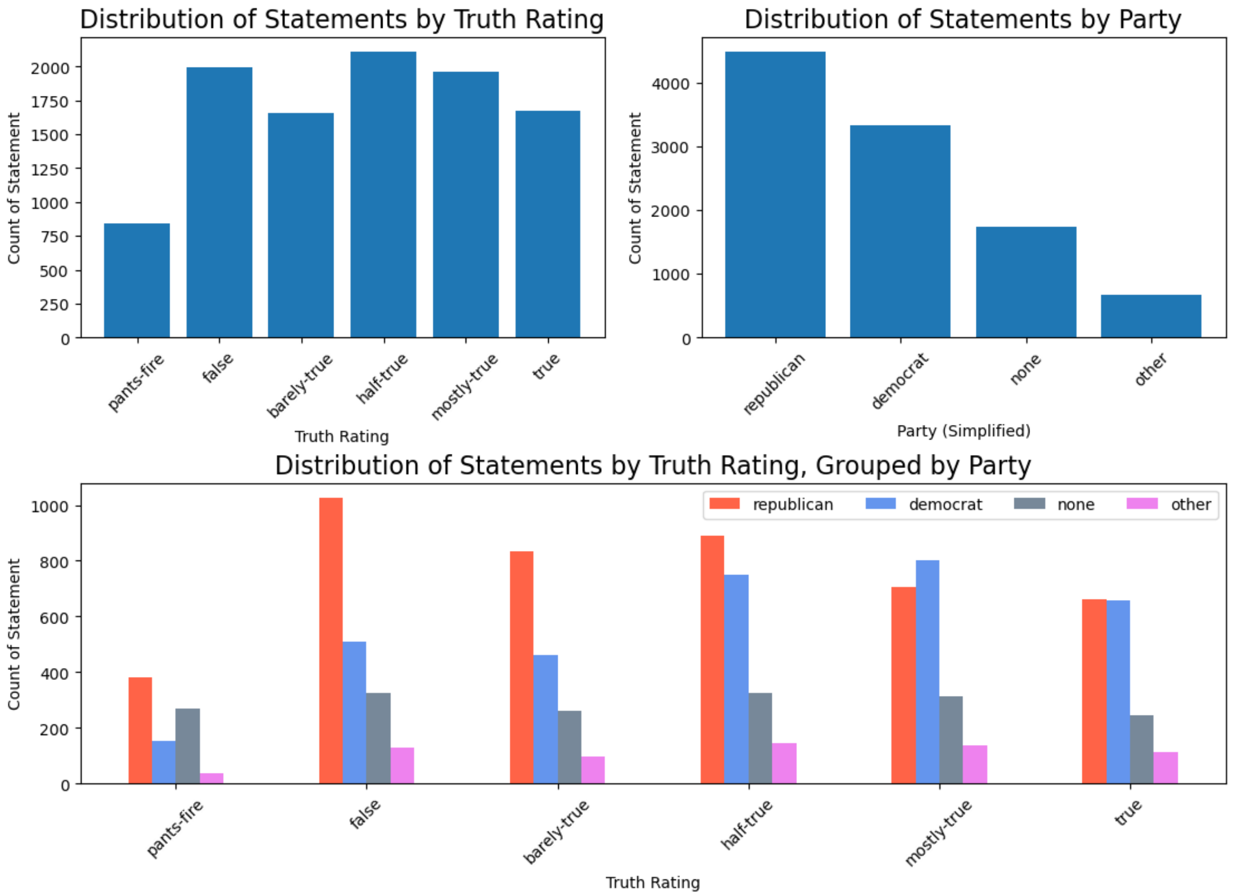
Plots that you can create **include**:

- Scatter Plot
- Bar plot
- Line Chart
- Multi Plots (e.g. using .subplot()
- Histogram

You can add another plot not listed here if it works better for your data. This is not a complete list of plots to create.

In [9]:

```
# Create simplified party column
train['party_simple'] = train.party.apply(
    lambda x: x if x == 'democrat' or x == 'republican' or x == 'none' else 'other'
)

# Convert party and label to Categorical variables
# in order to set order they will appear in the plots
truth_order = ['pants-fire', 'false', 'barely-true', 'half-true', 'mostly-true', 'true
party_order = ['republican', 'democrat', 'none', 'other']
train.label = pd.Categorical(train.label, truth_order)
train.party_simple = pd.Categorical(train.party_simple, party_order)

# Create figure and axes for the subplots
plt.figure(figsize = (11,8), layout = 'constrained')

ax1 = plt.subplot2grid((2, 2), (0, 0))
ax2 = plt.subplot2grid((2, 2), (0, 1))
ax3 = plt.subplot2grid((2, 2), (1, 0), colspan = 2)

# First subplot: Distribution of Statements by Truth Rating
ax1.bar(
    train.label.value_counts().reindex(truth_order).index,
    train.label.value_counts().reindex(truth_order).values
)
ax1.set_xlabel('Truth Rating')
ax1.set_ylabel('Count of Statement')
ax1.set_title('Distribution of Statements by Truth Rating', fontsize=16)
ax1.tick_params(axis = 'x', rotation = 45)

# Second subplot: Distribution of Statements by Party
ax2.bar(
    train.party_simple.value_counts().index,
    train.party_simple.value_counts().values
)
ax2.set_xlabel('Party (Simplified)')
ax2.set_ylabel('Count of Statement')
ax2.set_title('Distribution of Statements by Party', fontsize=16)
ax2.tick_params(axis = 'x', rotation = 45)

# Third subplot: Distribution of Statements by Truth Rating, Grouped by Party
(train
 .groupby('label').party_simple.value_counts().unstack()
 .plot(kind='bar', ax = ax3, color=['tomato', 'cornflowerblue', 'lightslategrey', 'vic
)
ax3.legend(ncol = 4, loc = 'upper right')
ax3.set_xlabel('Truth Rating')
ax3.set_ylabel('Count of Statement')
ax3.set_title('Distribution of Statements by Truth Rating, Grouped by Party', fontsize
ax3.tick_params(axis = 'x', rotation = 45)

plt.show()
```



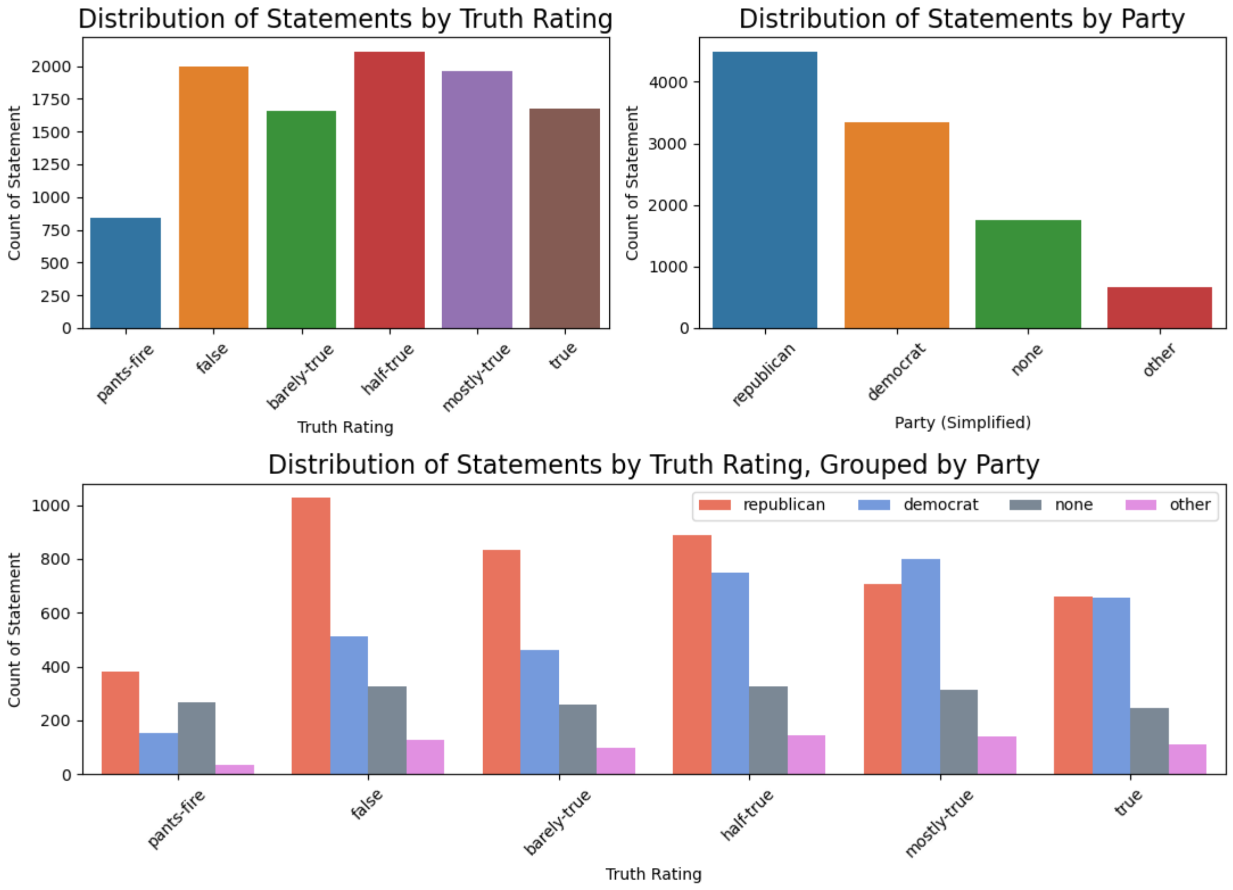
Part 2:

Recreate the visualizations above using the Seaborn library as best as possible.

You are required to explain what each of your plots is representing. Plots without comments will not be accepted. In addition, please explain the properties you are showcasing.

In [10]:

```
# Create figure and axes for the subplots
plt.figure(figsize = (11,8))

ax1 = plt.subplot2grid((2, 2), (0, 0))
ax2 = plt.subplot2grid((2, 2), (0, 1))
ax3 = plt.subplot2grid((2, 2), (1, 0), colspan = 2)

# First subplot: Distribution of Statements by Truth Rating
truth_countplot = sns.countplot(
    data = train,
    x = 'label',
    order = truth_order,
    ax = ax1
)
truth_countplot.set_xlabel('Truth Rating')
truth_countplot.set_ylabel('Count of Statement')
truth_countplot.set_title('Distribution of Statements by Truth Rating', fontsize=16)
truth_countplot.tick_params(axis = 'x', rotation = 45)

# Second subplot: Distribution of Statements by Party
party_countplot = sns.countplot(
    data = train,
    x = 'party_simple',
    order = party_order,
    ax = ax2
)
party_countplot.set_xlabel('Party (Simplified)')
party_countplot.set_ylabel('Count of Statement')
party_countplot.set_title('Distribution of Statements by Party', fontsize=16)
party_countplot.tick_params(axis = 'x', rotation = 45)

# Third subplot: Distribution of Statements by Truth Rating, Grouped by Party
grouped_barplot = sns.countplot(
    data = train,
    x = 'label',
    hue = 'party_simple',
    order = truth_order,
    palette=['tomato', 'cornflowerblue', 'lightslategrey', 'violet'],
    ax = ax3
)
grouped_barplot.legend(ncol = 4, loc = 'upper right')
grouped_barplot.set_xlabel('Truth Rating')
grouped_barplot.set_ylabel('Count of Statement')
grouped_barplot.set_title('Distribution of Statements by Truth Rating, Grouped by Part
grouped_barplot.tick_params(axis = 'x', rotation = 45)

# Adjust layout
plt.tight_layout()
plt.show()
```



Part 3:

In a comment or text box, explain the differences between creating a plot in matplotlib and seaborn, based on your above plots.

The primary difference is that, with matplotlib, we can directly leverage the Axes we creates as part of the `.subplots()` call. For example, we can call `ax1.plot()` to create the plot, then call various other methods to make adjustments (e.g. `ax1.set_title()`). In seaborn, we first need to name our initial plot as a new variable (e.g. `plotname = sns.plot()`). Then, we can make adjustments by calling methods on that new variable (e.g. `plotname.set_title()`).

Conclusions

After exploring your dataset, provide a short summary of what you noticed from this dataset.

Based on this initial analysis, I can highlight the following observations:

1. The data contains a relatively even distribution of statements across the various truth labels, with the exception of the `pants_on_fire` label, for which there are significantly fewer statements. This seems inuitive, given that the number of eggregious mistatements or intentional lies should be somewhat limited (or at least hopefully it will be!).

2. When breaking out the types of statements by party, there appears to be a greater number of false / misleading statements by Republicans as compared to Democrats. This point is confounded somewhat by the fact that there are more Republican statements overall. A normalized plot showing the proportion of statements under each category would probably help clarify whether this trend is meaningful.