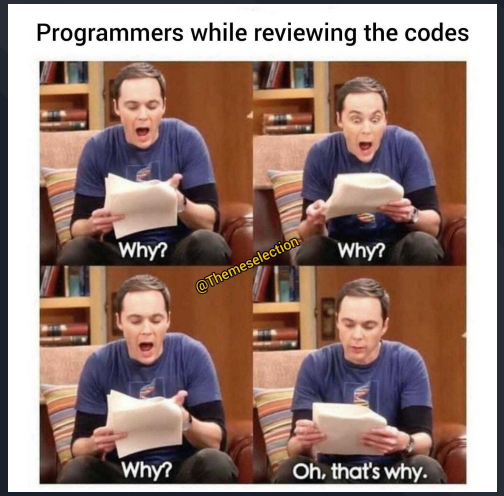
COGS 108 Week 7



Slides Adapted from Previous Quarter

AGENDA FOR TODAY

1. LOGISTICS
2. DISCUSSION
   LAB 6

LOGISTICS

DUE DATES

- Data Checkpoint due THIS FRIDAY May 17, 11:59 PM
  - Submit on GitHub
  - Refer to feedback on your proposal and make revisions!
  - If you successfully addressed the issues on proposal feedback, you will get points back!
- D6, Q7 are due next Monday May 20, 11:59PM

DATA SCIENCE

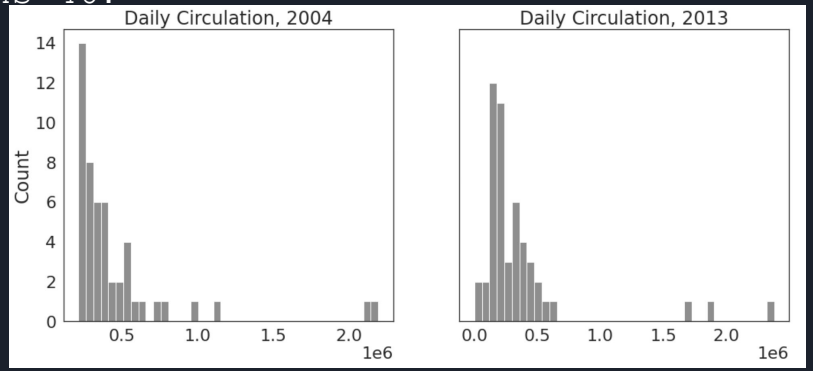
DISCUSSION LAB 6 INFERENTIAL ANALYSIS

Get rid of the commas in the numbers for Daily Circulation:

```
df['Series'].str.replace(old_str,
new_str).astype(float)
```

#Look at daily circulation distribution in 2004 and in 2013

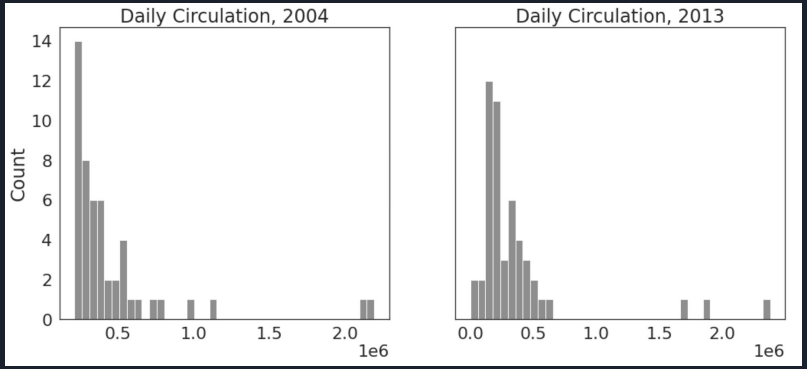
Plot using `sns.histplot()`. Parameters used for plot below: `bins=40`.



Tips: To create subplots in a plot

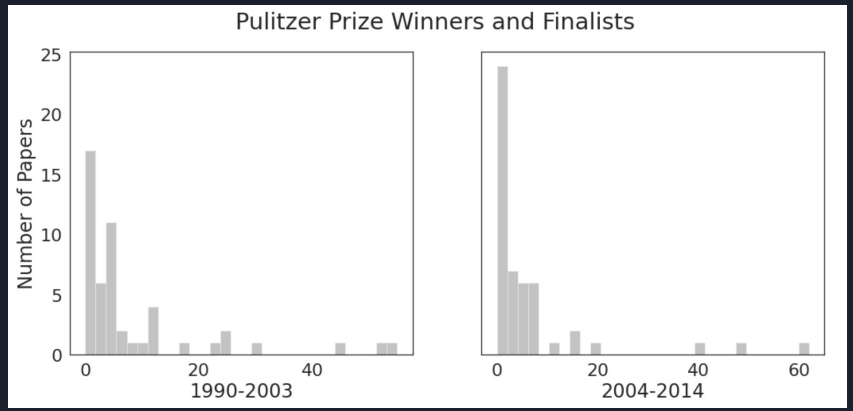
```
fig, (ax1, ax2) = plt.subplots(ncols=2,
sharey=True)

sns.histplot(….., ax=ax1)
sns.histplot(….., ax=ax2)
```

ax1 ax2



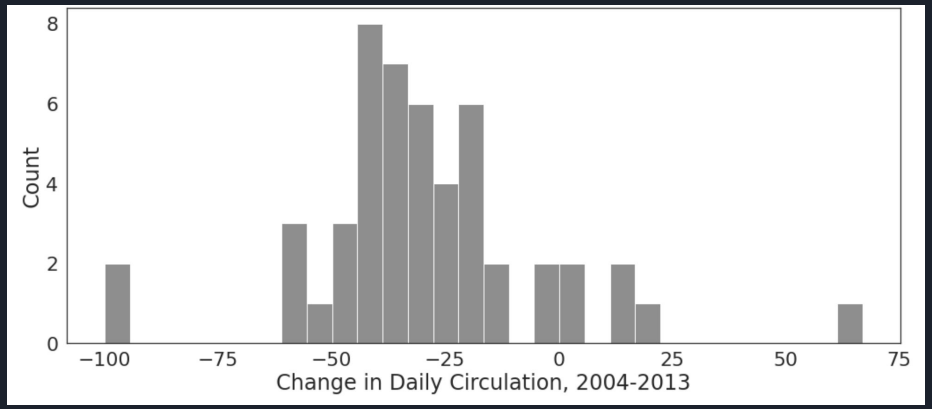
Let's look at the distribution of Pulitzer prize winners for the same time period.

Plot using sns.histplot() Parameters used for plot below: bins=30.



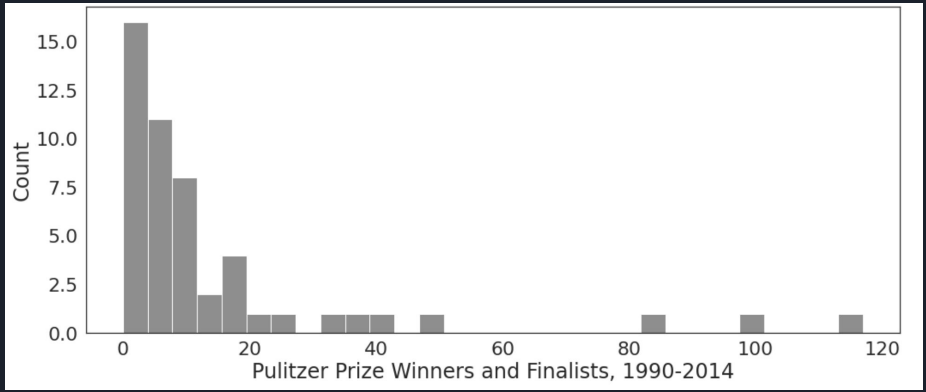
Plot the distribution of "change" in daily circulation:

Plot using sns.histplot(),
parameters used for plot below: kde=False, bins=30, color="dimgrey"



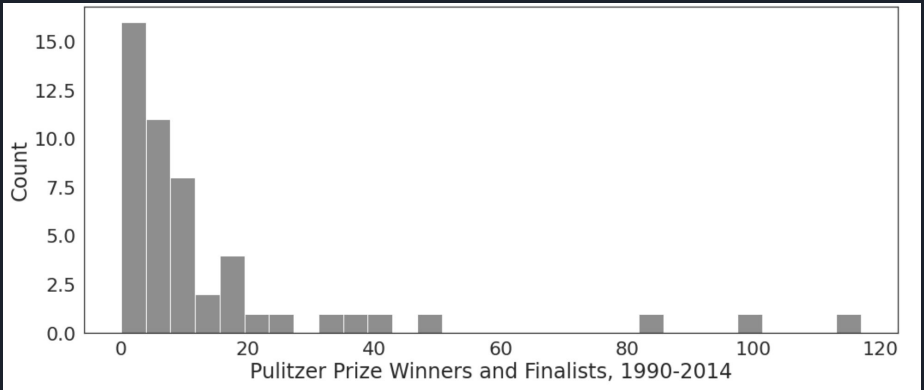
Look at pulitzer prize winner distributions:

Plot using sns.histplot()

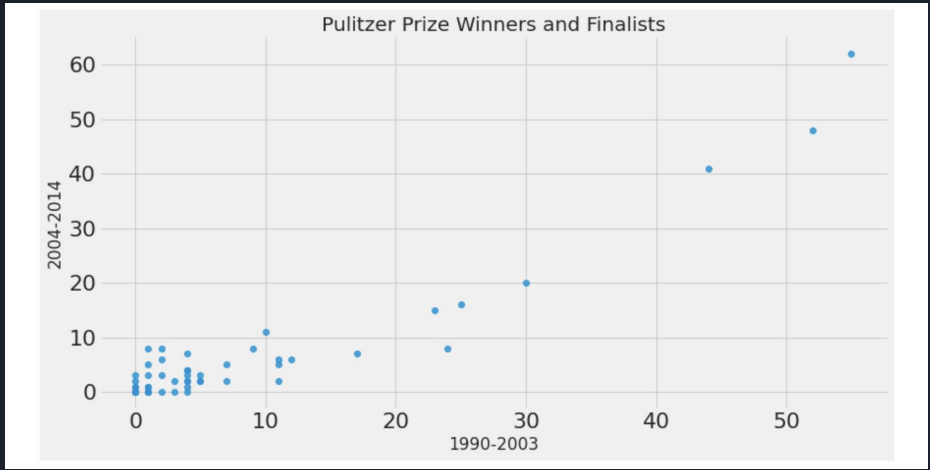


In the cell below look at pulitzer prize winner distributions

Plot using sns.histplot()

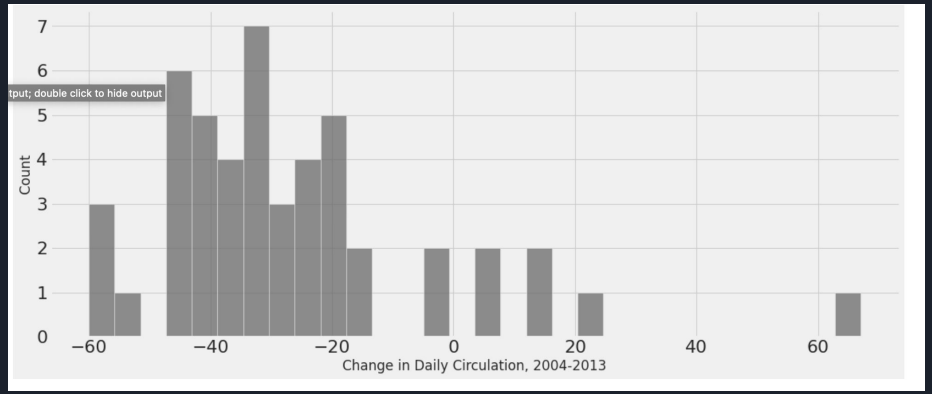


plot relationship between pulitzer prize winners/finalists in each time period and look at number of pulitzers between two time periods
Plot using sns.lmplot(x = 'Series1', y = 'Series2',data = DataframeName, fit_reg = False, height = 6, aspect=2)

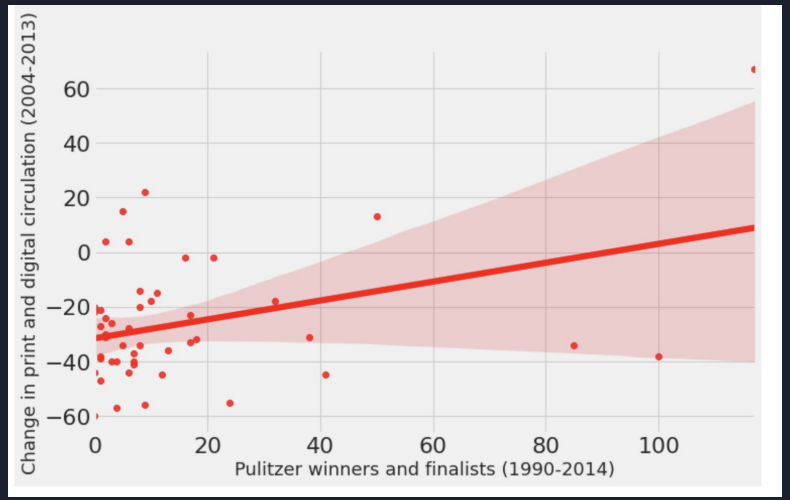


#Who has won the most pulitzers during the years we're looking at?
Use sort_values() to look at the top values

#Plot the distribution of daily change in circulation after outlier removal



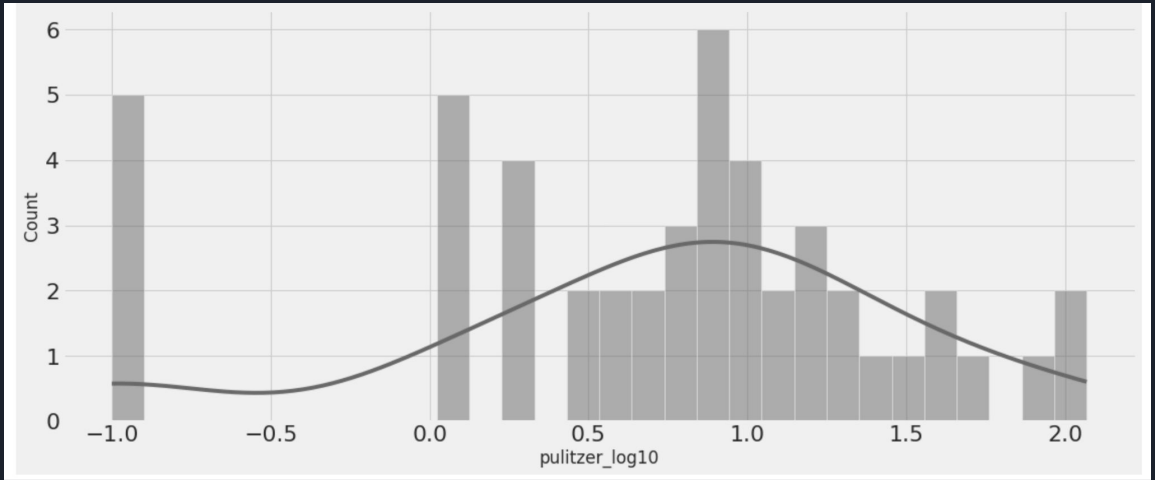
#Relationship between the total number of Pulitzers and change in readership (daily circulation)
Use sns.lmplot(x = 'Series1', y = 'Series2', data = dataFrameName, fit_reg = True,
height = 6, aspect = 1.7, line_kws={'color': 'red'}, scatter_kws={'color': 'red'})



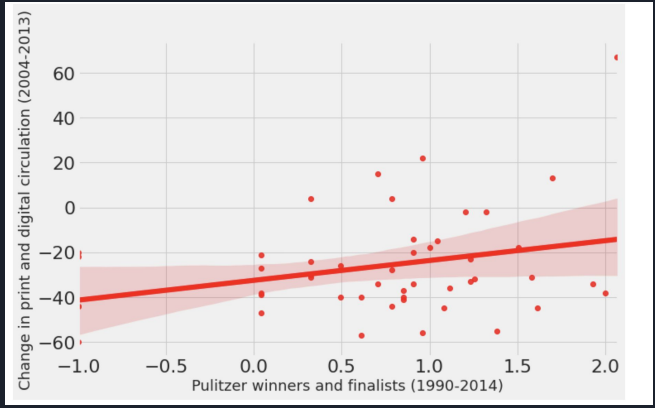
#Apply a log10-transformation the Pulitzer count data, with an offset of 0.1

Use pulitzer[‘pulitzer_log10’] = np.log10 (Series +0.1)

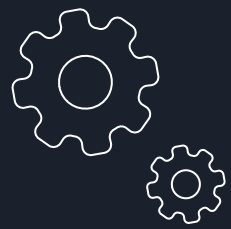
#In the next cell, visualize the distribution of the log10 column
Use sns.histplot()



#plot the relationship between our two variables of interest
Use sns.lmplot()

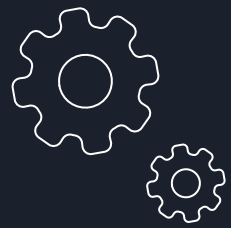


#Carry out linear regression; Now use statsmodels to initialize an OLS linear model This step initializes the model, and provides the data (but does not actually compute the model); fit the model; and Check out the results.



```
df = pulitzer[['Change in Daily Circulation, 2004-2013',
                    'pulitzer_log10']]
df.columns = ['circulation', 'pulitzer_log10']
df.head()

outcome, predictors = patsy.dmatrices('circulation ~ pulitzer_log10', df)

# Now use statsmodels to initialize an OLS linear model
#  This step initializes the model, and provides the data (but does not actually compute the model)
mod_log = sm.OLS(outcome, predictors)

# fit the model
res_log = mod_log.fit()

# Check out the results
print(res_log.summary())
```

#Carry out linear regression => (A modern way)
Fit the model; and Check out the results.



```
df = pulitzer[['Change in Daily Circulation, 2004-2013',
                'pulitzer_log10']]
df.columns = ['circulation', 'pulitzer_log10']
df.head()

# We can use statsmodels.formula.api to run this linear regression without patsy

# Go back up to the first cell and ‘import statsmodels.formula.api as smf’

# This step initializes the model, and provides the data (but does not actually compute the model)
mod_log = smf.ols(formula=‘circulation ~ pulitzer_log10’, data=df)

# fit the model
res_log = mod_log.fit()

# Check out the results
print(res_log.summary())
```

THANKS!


Someone literally bought a domain to do this
guthib.com
You spelled it wrong.