

Week 3

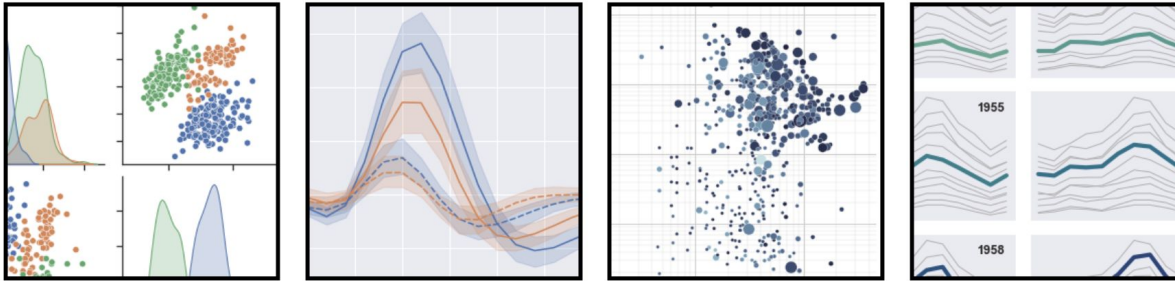
COGS 108 sp 2025

Due dates

- Q2: Monday (04/14)
- A1: Wednesday (04/16)
- Group Signup: Wednesday (04/16)
- D2: Friday (04/18)

Pyplot and seaborn

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics. Alias : **sns**



Pandas Series and Dataframes

Python3

```
#importing pandas library
import pandas as pd

#Creating a list
author = ['Jitender', 'Purnima', 'Arpit', 'Jyoti']
#Creating a Series by passing list variable to Series() function
auth_series = pd.Series(author)
#Printing Series
print(auth_series)
```

Output:

```
0    Jitender
1    Purnima
2     Arpit
3     Jyoti
dtype: object
```

Pandas Series and Dataframes

We have created two lists 'author' and 'article' which have been passed to Series() functions to create two Series.

After creating Series, we have created a dictionary and passed Series objects as values of the dictionary and keys of the dictionary will be served as Columns of the dataframe.

Python3

```
#Importing Pandas library
import pandas as pd

#Creating two lists
author = ['Jitender', 'Purnima', 'Arpit', 'Jyoti']
article = [210, 211, 114, 178]

#Creating two Series by passing lists
auth_series = pd.Series(author)
article_series = pd.Series(article)

#Creating a dictionary by passing Series objects as values
frame = { 'Author': auth_series, 'Article': article_series }

#Creating DataFrame by passing Dictionary
result = pd.DataFrame(frame)

#Printing elements of Dataframe
print(result)
```

Output:

	Author	Article
0	Jitender	210
1	Purnima	211
2	Arpit	114
3	Jyoti	178

Part I : Cheating

```
feature_counts =  
dataFrame['feature'].value_counts()
```

`df['your_column'].value_counts()` - this will return the count of unique occurrences in the specified column.

It is important to note that `value_counts` only works on pandas series, not Pandas dataframes. As a result, we only include one bracket `df['your_column']` and not two brackets `df[['your_column']]`.

Parameters

- **normalize (bool, default False)** - If True then the object returned will contain the relative frequencies of the unique values.
- **sort (bool, default True)** - Sort by frequencies.
- **ascending (bool, default False)** - Sort in ascending order.
- **bins (int, optional)** - Rather than count values, group them into half-open bins, a convenience for `pd.cut`, only works with numeric data.
- **dropna (bool, default True)** - Don't include counts of NaN.

Part I : Cheating

```
sns.countplot(x, y, hue, data=df);
```

Python3

```
# importing the required library

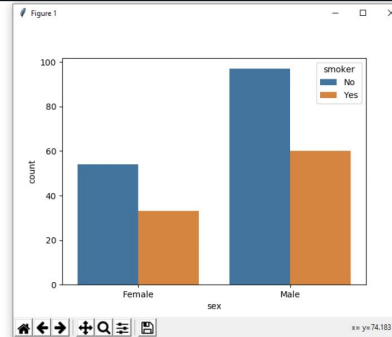
import seaborn as sns
import matplotlib.pyplot as plt

# read a tips.csv file from seaborn library
df = sns.load_dataset('tips')

# count plot on two categorical variable
sns.countplot(x='sex', hue='smoker', data=df)

# Show the plot
plt.show()
```

Output :



Part I : Cheating

create a DataFrame `prop_df` with three columns, one for gender, one for cheated, and one including the proportion of respondents who cheated within each gender

```
prop_df = (survey['cheated']  
           .groupby(...)  
           .value_counts(normalize=True)  
           .rename(...)  
           .reset_index())
```


Part I : Cheating

Regenerate your barplot using the proportion data you just generated to determine which gender cheats more frequently.

Assign your seaborn plot to a variable named `plot_proportion`

```
plot_proportion = sns.barplot(x=' ',  
                              y=' ',  
                              hue=' ',  
                              data=dataFrame);
```

Y axis is cheated
Y axis is proportion
The hue is the gender

Swapping: include
`hue_order=["Male","Female"],`

Your time to ...

- Talk to your classmates to find potential teammates!
- Work on D2 and A1