labassignment12bn

April 29, 2025

1 Lab Assignment 12: Interactive Visualizations

1.1 DS 6001: Practice and Application of Data Science

1.1.1 Instructions

Please answer the following questions as completely as possible using text, code, and the results of code as needed. Format your answers in a Jupyter notebook. To receive full credit, make sure you address every part of the problem, and make sure your document is formatted in a clean and professional way.

1.2 Problem 0

Import the following libraries:

[36]:
```
import numpy as np
import pandas as pd
import plotly.graph_objects as go
import plotly.express as px
import plotly.graph_objects as go
import plotly.figure_factory as ff
import dash
from jupyter_dash import JupyterDash
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output
from IPython.display import Image
external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']
```

For this lab, we will be working with the 2019 General Social Survey one last time.

[16]:
```
%%capture
gss = pd.read_csv("https://github.com/jkropko/DS-6001/raw/master/localdata/
 ↪gss2018.csv",
                  encoding='cp1252', na_values=['IAP','IAP,DK,NA,uncodeable',␣
 ↪'NOT SURE',
                                                'DK', 'IAP, DK, NA, uncodeable',␣
 ↪'.a', "CAN'T CHOOSE"])
```

Here is code that cleans the data and gets it ready to be used for data visualizations:

[17]:
```
mycols = ['id', 'wtss', 'sex', 'educ', 'region', 'age', 'coninc',
          'prestg10', 'mapres10', 'papres10', 'sei10', 'satjob',
          'fechld', 'fefam', 'fepol', 'fepresch', 'meovrwrk']
gss_clean = gss[mycols]
gss_clean = gss_clean.rename({'wtss':'weight',
                              'educ':'education',
                              'coninc':'income',
                              'prestg10':'job_prestige',
                              'mapres10':'mother_job_prestige',
                              'papres10':'father_job_prestige',
                              'sei10':'socioeconomic_index',
                              'fechld':'relationship',
                              'fefam':'male_breadwinner',
                              'fehire':'hire_women',
                              'fejobaff':'preference_hire_women',
                              'fepol':'men_bettersuited',
                              'fepresch':'child_suffer',
                              'meovrwrk':'men_overwork'},axis=1)
gss_clean.age = gss_clean.age.replace({'89 or older':'89'})
gss_clean.age = gss_clean.age.astype('float')
```

The `gss_clean` dataframe now contains the following features:

- `id` - a numeric unique ID for each person who responded to the survey
- `weight` - survey sample weights
- `sex` - male or female
- `education` - years of formal education
- `region` - region of the country where the respondent lives
- `age` - age
- `income` - the respondent's personal annual income
- `job_prestige` - the respondent's occupational prestige score, as measured by the GSS using the methodology described above
- `mother_job_prestige` - the respondent's mother's occupational prestige score, as measured by the GSS using the methodology described above
- `father_job_prestige` -the respondent's father's occupational prestige score, as measured by the GSS using the methodology described above
- `socioeconomic_index` - an index measuring the respondent's socioeconomic status
- `satjob` - responses to "On the whole, how satisfied are you with the work you do?"
- `relationship` - agree or disagree with: "A working mother can establish just as warm and secure a relationship with her children as a mother who does not work."
- `male_breadwinner` - agree or disagree with: "It is much better for everyone involved if the man is the achiever outside the home and the woman takes care of the home and family."
- `men_bettersuited` - agree or disagree with: "Most men are better suited emotionally for politics than are most women."
- `child_suffer` - agree or disagree with: "A preschool child is likely to suffer if his or her mother works."
- `men_overwork` - agree or disagree with: "Family life often suffers because men concentrate too much on their work."

1.3 Problem 1

Our goal in this lab is to build a dashboard that presents our findings from the GSS. A dashboard is meant to be shared with an audience, whether that audience is a manager, a client, a potential employer, or the general public. So we need to provide context for our results. One way to provide context is to write text using markdown code.

Find one or two websites that discuss the gender wage gap, and write a short paragraph in markdown code summarizing what these sources tell us. Include hyperlinks to these websites. Then write another short paragraph describing what the GSS is, what the data contain, how it was collected, and/or other information that you think your audience ought to know. A good starting point for information about the GSS is here: http://www.gss.norc.org/About-The-GSS

Then save the text as a Python string so that you can use the markdown code in your dashboard later.

It should go without saying, but no plagiarization! If you summarize a website, make sure you put the summary in your own words. Anything that is copied and pasted from the GSS webpage, Wikipedia, or another website without attribution will receive no credit.

(Don't spend too much time on this, and you might want to skip it during the Zoom session and return to it later so that you can focus on working on code with your classmates.) [1 point]

In America, women make on average 85 cents for every dollar a man makes. This is a concerning number, but this number alone does not tell the whole story. It does not tell the story of how far we've come to achieving parity. The average pay gap in 1982 was 65%, suggesting we've come a long way in the past 20 years, though it seems to have stalled a bit only increasing from 81% in 2003. Despite this not all hope is lost about achieving parity in pay. When broken down by age group, women age 25-34 earn 95 cents to the dollar. This suggests a recent trend of near parity that, if it continues, will drive the pay gap down. Source: https://www.pewresearch.org/short-reads/2025/03/04/gender-pay-gap-in-us-has-narrowed-slightly-over-2-decades/#:~:text=The%20U.S.%20Census%20Bureau%20has,Census%20Bureau's%20most%20recent%20analysis

The General Social Survey (GSS) is a representative sample of adults in the United States. The survey contains data about demoagraphics, behaviors, and attitudes amoung other spical interest topics. The GSS has been collecting this data since 1972. Because the survey is so big and large ranging over the course of decades it becomes the best source to compare trends over the course of time and over different subgroups about the fabric of America.

[18]:
```
p1 = "The General Social Survey (GSS) is a representative sample of adults in␣
↪the United States. The survey contains data about demoagraphics, behaviors,␣
↪and attitudes amoung other spical interest topics. The GSS has been␣
↪collecting this data since 1972. Because the survey is so big and large␣
↪ranging over the course of decades it becomes the best source to compare␣
↪trends over the course of time and over different subgroups about the fabric␣
↪of America."
```

1.3.1 I cannot express how frustrated I am in trying to convert this notebook to a pdf. None of the graphs or images of the graphs are coming through. So I am going to submit anyway. BUT I will give you this:

1.3.2 https://drive.google.com/file/d/1Xytv4EQpkb4InyyHXerqc9Jy1Yu9Nfta/view?usp=drive_

1.3.3 This is a link to the html file that has a working version of all the interactive plotly graphs plus a scrollable dashboard. Please look at it to see that everything actually does work. Please note that the html file was exported before the code was written to put images in there or this note was written, but the actual code to generate the graphs is the same. You may have to download it and open the download to actually view it. Thank you for your consideration.

1.4 Problem 2

Generate a table that shows the mean income, occupational prestige, socioeconomic index, and years of education for men and for women. Use a function from a `plotly` module to display a web-enabled version of this table. This table is for presentation purposes, so round every column to two decimal places and use more presentable column names. [3 points]

[29]:
```
p2 = gss_clean.groupby('sex').agg({'income':'mean', 'job_prestige':'mean',␣
  ↪'socioeconomic_index':'mean', 'education':'mean'}).reset_index().round(2)
p2 = p2.rename(columns={'sex': 'Sex',
                        'income':'Mean Income',
                        'job_prestige':'Mean Job Prestige Score',
                        'socioeconomic_index':'Mean Socioeconomic Index',
                        'education':'Mean Years of Education'})
p2 = ff.create_table(p2)
p2.show(renderer='notebook')
```

[38]:
```
Image(url="lab12_p2.png")
```

[38]:
```
<IPython.core.display.Image object>
```

1.5 Problem 3

Create an interactive barplot that shows the number of men and women who respond with each level of agreement to `male_breadwinner`. Write presentable labels for the x and y-axes, but don't bother with a title because we will be using a subtitle on the dashboard for this graphic. [3 points]

[20]:
```
p3df = gss_clean.groupby(['male_breadwinner', 'sex']).size().
  ↪reset_index(name='count')
p3df
```

[20]:

	male_breadwinner	sex	count
0	agree	female	152
1	agree	male	158
2	disagree	female	377
3	disagree	male	337
4	strongly agree	female	48

```
5     strongly agree    male     40
6  strongly disagree  female    286
7  strongly disagree    male    147
```

[30]:
```
p3 = px.bar(p3df, x='male_breadwinner', y='count', color='sex',
            labels={'male_breadwinner':'Level of Agreement', 'count':'Count'},
            category_orders={'male_breadwinner': ['strongly disagree',␣
 ↪'disagree', 'agree', 'strongly agree']},
            color_discrete_map={'male':'blue', 'female':'red'},
            barmode='group')
p3.show(renderer='notebook')
```

[39]:
```
Image(url="lab12_p3.png")
```

[39]:
```
<IPython.core.display.Image object>
```

1.6 Problem 4

Create an interactive scatterplot with `job_prestige` on the x-axis and `income` on the y-axis. Color code the points by `sex` and make sure that the figure includes a legend for these colors. Also include two best-fit lines, one for men and one for women. Finally, include hover data that shows us the values of `education` and `socioeconomic_index` for any point the mouse hovers over. Write presentable labels for the x and y-axes, but don't bother with a title because we will be using a subtitle on the dashboard for this graphic. [3 points]

[31]:
```
p4 = px.scatter(gss_clean, x='job_prestige', y='income', color='sex',
                labels={'job_prestige':'Job Prestige Score', 'income':'Income',␣
 ↪'sex':'Sex'},
                color_discrete_map={'male':'blue', 'female':'red'},
                trendline='ols',
                hover_data=['education', 'socioeconomic_index'])
p4.show(renderer='notebook')
```

[40]:
```
Image(url="lab12_p4.png")
```

[40]:
```
<IPython.core.display.Image object>
```

1.7 Problem 5

Create two interactive box plots: one that shows the distribution of `income` for men and for women, and one that shows the distribution of `job_prestige` for men and for women. Write presentable labels for the axis that contains `income` or `job_prestige` and remove the label for `sex`. Also, turn off the legend. Don't bother with titles because we will be using subtitles on the dashboard for these graphics. [3 points]

[32]:
```
p5i = px.box(gss_clean, x='income', y='sex', color='sex',
             labels={'income':'Income', 'sex':''},
             color_discrete_map={'male':'blue', 'female':'red'},)
```

```
p5i.update_layout(showlegend=False)
p5i.show(renderer='notebook')
```

[41]:
```
Image(url="lab12_p5i.png")
```

[41]:
```
<IPython.core.display.Image object>
```

[33]:
```
p5p = px.box(gss_clean, x='job_prestige', y='sex', color='sex',
             labels={'income':'Income', 'sex':''},
             color_discrete_map={'male':'blue', 'female':'red'},)
p5p.update_layout(showlegend=False)
p5p.show(renderer='notebook')
```

[42]:
```
Image(url="lab12_p5j.png")
```

[42]:
```
<IPython.core.display.Image object>
```

1.8 Problem 6

Create a new dataframe that contains only `income`, `sex`, and `job_prestige`. Then create a new feature in this dataframe that breaks `job_prestige` into six categories with equally sized ranges. Finally, drop all rows with any missing values in this dataframe.

Then create a facet grid with three rows and two columns in which each cell contains an interactive box plot comparing the income distributions of men and women for each of these new categories.

(If you want men to be represented by blue and women by red, you can include `color_discrete_map = {'male':'blue', 'female':'red'}` in your plotting function. Or use different colors if you want!) [3 points]

[25]:
```
p6cols = ['sex', 'job_prestige', 'income']
p6df = gss_clean[p6cols]
p6df = p6df.assign(prestige_group =
                   pd.cut(p6df.job_prestige,
                          6,
                          labels=['Very Low','Low','Somewhat Low','Somewhat␣
 ↪High','High','Very High']))
p6df = p6df.dropna()
```

[34]:
```
p6 = px.box(p6df, x='income', y='sex', color='sex', facet_col='prestige_group',␣
 ↪facet_col_wrap=2,
            labels={'income':'Income', 'sex':''},
            category_orders={'prestige_group': ['Very Low','Low','Somewhat␣
 ↪Low','Somewhat High','High','Very High']},
            color_discrete_map={'male':'blue', 'female':'red'},
            )
p6.for_each_annotation(lambda a: a.update(text=a.text.
 ↪replace("prestige_group=", "")))
p6.update_layout(showlegend=False)
```

```
p6.update(layout=dict(title=dict(x=0.5)))
p6.show(renderer='notebook')
```

[43]:
```
Image(url="lab12_p6.png")
```

[43]:
```
<IPython.core.display.Image object>
```

1.9 Problem 7

Create a dashboard that displays the following elements:

- A descriptive title
- The markdown text you wrote in problem 1
- The table you made in problem 2
- The barplot you made in problem 3
- The scatterplot you made in problem 4
- The two boxplots you made in problem 5 side-by-side
- The faceted boxplots you made in problem 6
- Subtitles for all of the above elements

Use `JupyterDash` to display this dashboard directly in your Jupyter notebook.

Any working dashboard that displays all of the above elements will receive full credit. [4 points]

Here is some code for Challenge 2. You can see it in the app above for number 7. I had to put it here so the dashboard would run.

[35]:
```
c2x = ['satjob', 'relationship', 'male_breadwinner', 'men_bettersuited',␣
 ↪'child_suffer', 'men_overwork']
c2g = ['sex', 'region', 'education']

c2 = px.bar(gss_clean, x='male_breadwinner', color='sex',
            labels={'male_breadwinner':'Level of Agreement', 'count':'Count'},
            barmode='group')
c2.show(renderer='notebook')
```

[44]:
```
Image(url="lab12_c2.png")
```

[44]:
```
<IPython.core.display.Image object>
```

Here starts the code for the dashboard for problem 7.

[28]:
```
app = JupyterDash(__name__, external_stylesheets=external_stylesheets)

app.layout = html.Div(
    [
        html.H1("GSS Gender Paygap Dashboard"),
```

```
        dcc.Markdown(p1),

        html.H2("Relevant Pay Variables by Sex"),
        dcc.Graph(figure=p2),

        html.H2("Response to 'It is much better for everyone involved if the␣
↪man is the achiever outside the home and the woman takes care of the home␣
↪and family.'"),
        dcc.Graph(figure=p3),

        html.H2("Income vs Job Prestige by Sex"),
        dcc.Graph(figure=p4),

        html.Div([

            html.H2("Distribution of Income by Sex"),
            dcc.Graph(figure=p5i),
        ], style = {'width':'48%', 'float':'left'}),

        html.Div([
            html.H2("Distribution of Job Prestige by Sex"),
            dcc.Graph(figure=p5p),
        ], style = {'width':'48%', 'float':'right'}),

        html.H2("Distribution of Income by Sex and Prestige Group"),
        dcc.Graph(figure=p6),

        html.H2("Challenge 2"),

        html.Div([
            html.H3("X Variable"),

            dcc.Dropdown(
                id='x-axis',
                options=[{'label': i, 'value': i} for i in c2x],
                ),

            html.H3("Grouped Variable"),

            dcc.Dropdown(
                id='groups',
                options=[{'label': i, 'value': i} for i in c2g])

        ], style={'width': '25%', 'float': 'left'}),

        html.Div([
            dcc.Graph(id='bar-graph'),
```

```
        ], style={'width': '70%', 'float': 'right'}),

    ]
)

@app.callback(
    dash.Output('bar-graph', 'figure'),
    [dash.Input('x-axis', 'value'),
     dash.Input('groups', 'value')]
)
def update_graph(x, group):
    c2 = px.bar(
        gss_clean,
        x=x,
        y='income',
        color=group,
        barmode='group',
        title=f'Grouped Bar Chart for {x} by {group}'
    )
    return c2

if __name__ == '__main__':
    app.run(mode='inline', debug=True)
```

```
c:\Users\brian\AppData\Local\Programs\Python\Python313\Lib\site-
packages\dash\dash.py:587: UserWarning:

JupyterDash is deprecated, use Dash instead.
See https://dash.plotly.com/dash-in-jupyter for more details.


<IPython.lib.display.IFrame at 0x232d9ae2a50>
```

[46]:
```
Image(url="lab12_p7-1.png")
```

[46]:
```
<IPython.core.display.Image object>
```

[47]:
```
Image(url="lab12_p7-2.png")
```

[47]:
```
<IPython.core.display.Image object>
```

[48]:
```
Image(url="lab12_p7-3.png")
```

[48]:
```
<IPython.core.display.Image object>
```

[49]:
```
Image(url="lab12_p7-4.png")
```

[49]:
```
<IPython.core.display.Image object>
```

[50]:
```
Image(url="lab12_p7-5.png")
```

[50]:
```
<IPython.core.display.Image object>
```

[51]:
```
Image(url="lab12_p7-6.png")
```

[51]:
```
<IPython.core.display.Image object>
```

[52]:
```
Image(url="lab12_p7-7.png")
```

[52]:
```
<IPython.core.display.Image object>
```

1.10 Extra Credit (up to 10 bonus points)

Dashboards are all about good design, functionality, and accessability. For this extra credit problem, create another version of the dashboard you built for problem 7, but take extra steps to improve the appearance of the dashboard, add user-inputs, and host it on the internet with its own URL.

Challenge 1: Be creative and use a layout that significantly departs from the one used for the ANES data in the module 12 notebook. A good place to look for inspiration is the Dash gallery. We will award up to 3 bonus points for creativity, novelty, and style.

Challenge 2: Alter the barplot from problem 3 to include user inputs. Create two dropdown menus on the dashboard. The first one should allow a user to display bars for the categories of `satjob`, `relationship`, `male_breadwinner`, `men_bettersuited`, `child_suffer`, or `men_overwork`. The second one should allow a user to group the bars by `sex`, `region`, or `education`. After choosing a feature for the bars and one for the grouping, program the barplot to update automatically to display the user-inputted features. One bonus point will be awarded for a good effort, and 3 bonus points will be awarded for a working user-input barplot in the dashboard.

Challenge 3: Follow these steps to host the dashboard publicly on PythonAnywhere: https://docs.google.com/document/d/1lYxsRQ_J0llM5Ztk0CN4c5JQkzlyr8dv6qMjcfLsMh0/edit?usp=sharing 4 bonus points will be awarded for a working PythonAnywhere link.

The code for challenge 2 is above problem 7 so the dashboard would run.