2 - Plotly

June 27, 2022

https://plotly.com/python-api-reference/plotly.express.html

https://plotly.com/python/

1. Plotly express
   - bar chart
   - line chart
   - scatterplot
   - exercise - pick two, create share
2. plotly graph objects (go)
   - figure structure - The structure of a figure - data, traces and layout explained
     - https://plotly.com/python/figure-structure/
     - tree of attributes
     - data (aka traces)
     - layout
     - frames (used in animated plots)
   - display figures
     - in a notebook or script… fig.show()
     - renderers png, jpeg, etc. fig.show(renderer="png", width=800, height=300)
     - export to html
     - static using Kaleido…..https://plotly.com/python/static-image-export/
   - bar charts
   - line charts
   - scatterplot
   - map
3. subplots
   - https://plotly.com/python/creating-and-updating-figures/
   - go down to subplot section

1 Getting the data

These next few steps read data from data.cdc.gov and do some clean-up and data prep.

[54]:
```
import requests
import pandas as pd
import numpy as np
import plotly.express as px
```

[55]:
```
# Get the data from CDC and look at it in json format

response = requests.get("https://data.cdc.gov/resource/saz5-9hgg.json")
jsonhold = response.json()
#jsonhold
```

[56]:
```
# Put the data into a DataFrame

vaccines = pd.DataFrame(jsonhold)

# Create month and week columns

vaccines['month'] = pd.to_datetime(vaccines['week_of_allocations']).dt.month
vaccines['week'] =  pd.to_datetime(vaccines['week_of_allocations']).dt.week


# Changing the datatypes & column names

vaccines['month'] = vaccines.month.astype(str)
vaccines['week'] = vaccines.week.astype(str)
vaccines['_1st_dose_allocations'] = pd.
 ↪to_numeric(vaccines['_1st_dose_allocations']).astype(int)
vaccines['_2nd_dose_allocations'] = pd.
 ↪to_numeric(vaccines['_2nd_dose_allocations']).astype(int)
vaccines['_2nd_dose_allocations'] = vaccines._2nd_dose_allocations*1.2


short_names = {'_1st_dose_allocations':'first',
               '_2nd_dose_allocations':'second'}
vaccines.rename(columns=short_names, inplace=True)
vaccines = vaccines[vaccines.jurisdiction.isin(['Massachusetts','New␣
 ↪Hampshire', 'Rhode Island'])]


vaccines.head()
```

```
/var/folders/bg/jzzhjp857hv08kcqg3jdptcr0000gn/T/ipykernel_7488/3172776109.py:8:
FutureWarning:

Series.dt.weekofyear and Series.dt.week have been deprecated. Please use
Series.dt.isocalendar().week instead.
```

[56]:

	jurisdiction	week_of_allocations	first	second	month	week
2	Massachusetts	2021-06-21T00:00:00.000	104580	125496.0	6	25
3	New Hampshire	2021-06-21T00:00:00.000	21420	25704.0	6	25

```
4    Rhode Island  2021-06-21T00:00:00.000   17280   20736.0     6   25
65  Massachusetts  2021-06-14T00:00:00.000  104580  125496.0     6   24
66  New Hampshire  2021-06-14T00:00:00.000   21420   25704.0     6   24
```

[57]:
```
vaccines.shape
```

[57]:
```
(48, 6)
```

[58]:
```
vaccines = vaccines.sort_values(by='month')

fig = px.line(vaccines,
              x = 'month',
              y = 'first',
              color = 'jurisdiction',
              markers = True,
              symbol = 'jurisdiction')
fig.show()
```

[59]:
```
v_week = vaccines.groupby('week').sum().reset_index()
v_week.head()
```

[59]:
```
   week   first    second
0   10  119340  143208.0
1   11  127530  153036.0
2   12  131040  157248.0
3   13  159120  190944.0
4   14  131040  157248.0
```

[60]:
```
v_month = vaccines.groupby('month').sum().reset_index()
v_month.head()
```

[60]:
```
   month   first    second
0      3  537030  644436.0
1      4  548730  658476.0
2      5  714240  857088.0
3      6  429840  515808.0
```

[61]:
```
v_sm = vaccines.groupby(['jurisdiction','month']).sum().reset_index()
v_sm.head()
```

[61]:
```
     jurisdiction month   first    second
0  Massachusetts     3  394290  473148.0
1  Massachusetts     4  403650  484380.0
2  Massachusetts     5  525780  630936.0
3  Massachusetts     6  313740  376488.0
4  New Hampshire     3   79560   95472.0
```

2 plotly express

https://plotly.com/python-api-reference/plotly.express.html

[62]:
```
fig = px.line(v_week, x = 'week', y = 'first')
fig.show()
```

[63]:
```
fig = px.line(v_sm,
              x = 'month',
              y = 'first',
              color = 'jurisdiction',
              markers = True,
              symbol = 'jurisdiction',
              text = 'first')
fig.show()
```

[64]:
```
fig = px.scatter(v_sm,
              x = 'first',
              y = 'second')
fig.show()
```

[65]:
```
# Using aggregated data
fig = px.bar(v_sm,
              x = 'month',
              y = 'first')
fig.show()
```

[66]:
```
fig = px.bar(v_sm,
             x = 'month',
             y = 'first',
             color = 'jurisdiction')
fig.show()
```

[67]:
```
# Continuous color
fig = px.bar(v_month,
             x = 'month',
             y = 'first',
             color = 'second')
fig.show()
```

[68]:
```
# Unaggregated data

fig = px.bar(vaccines, x = 'jurisdiction', y = 'first', color = 'month')
fig.show()
```

[69]:
```
# A more dramatic example of same phenomena
```

```
df = px.data.tips()
fig = px.bar(df,
             x="sex",
             y="total_bill",
             color='time')
fig.show()
```

[70]:
```
# Stacked unaggregated data

fig = px.bar(vaccines, x = 'jurisdiction', y = 'first', color = 'month')
fig.show()
```

[71]:
```
# Side-by-side unaggregated data

fig = px.bar(vaccines,
             x = 'jurisdiction',
             y = 'first',
             color = 'month',
             barmode = 'group')
fig.show()
```

[72]:
```
# Use histogram to aggregate

fig = px.histogram(vaccines,
             x = 'jurisdiction',
             y = 'first',
             color = 'month',
             barmode = 'group')
fig.show()
```

[73]:
```
# faceted subplots   ##### Different dataset!

df = px.data.tips()
fig = px.bar(df,
             x="sex",
             y="total_bill",
             color="smoker",
             barmode="group",
             facet_row="time",
             facet_col="day",
             category_orders={"day": ["Thur", "Fri", "Sat", "Sun"],
                              "time": ["Lunch", "Dinner"]})
fig.show()
```

[ ]:
```

```

3 Exercise 1 - 10 minutes

[74]:
```
# Exercise 1 plotly express 1 - pie chart
```

[75]:
```
# Exercise 1 plotly express 2 - boxplot
```

4 plotly graph objects

4.1 Getting the data ready

[76]:
```
import plotly.graph_objects as go
import pandas as pd
ob = pd.read_csv('https://raw.githubusercontent.com/jimcody2014/Python-Data/
 ↪main/outbreaks-dashboard.csv')
ob.head()
```

[76]:

	Year	Month	State	Location	\
0	1998	January	California	Restaurant	
1	1998	January	California	NaN	
2	1998	January	California	Restaurant	
3	1998	January	California	Restaurant	
4	1998	January	California	Private Home/Residence	

	Food	Ingredient	Species	\
0	NaN	NaN	NaN	
1	Custard	NaN	NaN	
2	NaN	NaN	NaN	
3	Fish, Ahi	NaN	Scombroid toxin	
4	Lasagna, Unspecified; Eggs, Other	NaN	Salmonella enterica	

	Serotype/Genotype	Status	Illnesses	Hospitalizations	Fatalities
0	NaN	NaN	20	0.0	0.0
1	NaN	NaN	112	0.0	0.0
2	NaN	NaN	35	0.0	0.0
3	NaN	Confirmed	4	0.0	0.0
4	Enteritidis	Confirmed	26	3.0	0.0

[77]:
```
ob_month = ob.groupby('Month')[['Illnesses','Hospitalizations', 'Fatalities']].
 ↪sum().reset_index()
```

[78]:
```
oby = ob.groupby('Year')[['Illnesses','Hospitalizations', 'Fatalities']].sum().
 ↪reset_index()
```

[79]:
```
obs = ob.groupby('State')[['Illnesses','Hospitalizations', 'Fatalities']].sum().
 ↪reset_index()
```

4.2 Bar Charts

[80]:
```
# Basic graph object
fig = go.Figure(
    data=[go.Bar(x=['apples', 'oranges', 'bananas'], y=[1, 3, 2])],
    layout=go.Layout(
        title=go.layout.Title(text="A Figure Specified By A Graph Object")
    )
)

fig.show()
```

[81]:
```
print(fig)
```

```
Figure({
    'data': [{'type': 'bar', 'x': ['apples', 'oranges', 'bananas'], 'y': [1, 3,
2]}],
    'layout': {'template': '…', 'title': {'text': 'A Figure Specified By A
Graph Object'}}
})
```

[82]:
```
# Very minimal

fig = go.Figure([go.Bar(x=['apples', 'oranges', 'bananas'], y=[1, 3, 2])])
fig.show()
```

[83]:
```
# With dataframe data - version 1
fig = go.Figure(go.Bar(x=ob['Month'], y = ob['Illnesses'],hovertemplate = "%{x}:
 ↪ <br>Illnesses: %{y} </br> %{y}"))
fig.show()
```

[84]:
```
# With dataframe data - version 2 - just a different way of accessing the␣
 ↪variables

fig = go.Figure(go.Bar(x=ob.Month, y = ob.Illnesses))
fig.show()
```

[85]:
```
# With aggregated dataframe data
fig = go.Figure(go.Bar(x=ob_month.Month, y = ob_month.Illnesses))
fig.show()
```

[86]:
```
# Multiple traces
fig = go.Figure(
    data=[go.Bar(name = 'ill', x=ob_month.Month, y = ob_month.Illnesses),
          go.Bar(name = 'hosp', x=ob_month.Month, y = ob_month.
 ↪Hospitalizations)],
    layout=go.Layout(
```

```
        title=go.layout.Title(text="A Figure Specified By A Graph Object")
    )
)

fig.show()
```

[87]:
```
# Layout update
fig = go.Figure(
    data=[go.Bar(name = 'ill', x=ob_month.Month, y = ob_month.Illnesses),
          go.Bar(name = 'hosp', x=ob_month.Month, y = ob_month.
 ↪Hospitalizations)],
    layout=go.Layout(
        title=go.layout.Title(text="A Figure Specified By A Graph Object")
    )
)
fig.update_layout(barmode='stack')
fig.show()
```

[88]:
```
# From the documentation - Adding multiple 'traces'
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun',
          'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']

fig = go.Figure()
fig.add_trace(go.Bar(
    x=months,
    y=[20, 14, 25, 16, 18, 22, 19, 15, 12, 16, 14, 17],
    name='Primary Product',
    marker_color='indianred'
))
fig.add_trace(go.Bar(
    x=months,
    y=[19, 14, 22, 14, 16, 19, 15, 14, 10, 12, 12, 16],
    name='Secondary Product',
    marker_color='lightsalmon'
))

# Here we modify the tickangle of the xaxis, resulting in rotated labels.
fig.update_layout(barmode='group', xaxis_tickangle=-45)
fig.show()
```

[89]:
```
# Modifying the Hover text & traces update

fig = go.Figure(go.Bar(x=ob_month.Month, y = ob_month.Illnesses,
                       hovertext=['A lot', 'medium', 'Big']))

fig.update_traces(marker_color='rgb(158,202,225)',␣
 ↪marker_line_color='rgb(8,48,107)',
```

```
                  marker_line_width=1.5, opacity=0.6)
fig.update_layout(title_text='Outbreaks by Month')
fig.show()
```

[90]:
```
# Modifying colors

# amts = [37,27,33,30,29,30,35,33,37,32,27,24]
colors = ['lightslategray',] * 12
colors[11] = 'crimson'

fig = go.Figure(go.Bar(x=ob_month.Month, y = ob_month.Illnesses,
                       hovertext=['A lot', 'medium', 'Big'],
                       text = ob_month.Illnesses,
                       textposition = 'auto',
                       marker_color = colors)
                )

fig.update_layout(title_text='Outbreaks by Month')

fig.update_traces(texttemplate='%{text:.2s}', textposition='outside')
fig.update_layout(uniformtext_minsize=8, uniformtext_mode='hide')

fig.show()
```

[91]:
```
# Sorting as part ogf the layout

fig = go.Figure(
    data=[go.Bar(name = 'ill', x=ob_month.Month, y = ob_month.Illnesses),
          go.Bar(name = 'hosp', x=ob_month.Month, y = ob_month.
 ↪Hospitalizations)],
    layout=go.Layout(
        title=go.layout.Title(text="A Figure Specified By A Graph Object")
    )
)
fig.update_layout(barmode='stack', xaxis={'categoryorder':'total ascending'}) ␣
 ↪# descending
fig.show()
```

4.3 Scatterplot

Reminder: ob is outbreaks. ob_month is outbreak data aggregated to the month

[92]:
```
fig = go.Figure(data=go.Scatter(x=ob_month.Illnesses, y=ob_month.Fatalities,␣
 ↪mode = 'markers'))
fig.show()
```

[93]:
```
# Same figure as above
fig = go.Figure()
fig.add_trace(go.Scatter(
    x=ob_month.Illnesses,
    y=ob_month.Fatalities,
    mode = 'markers',
    marker_color='indianred'
))
```

When using Plotly graphic objects, **Scatter** is also used to create line charts. The marker used chnges the style.

[94]:
```
# From documentation

import numpy as np
np.random.seed(1)

N = 100
random_x = np.linspace(0, 1, N)
random_y0 = np.random.randn(N) + 5
random_y1 = np.random.randn(N)
random_y2 = np.random.randn(N) - 5

# Create traces
fig = go.Figure()
fig.add_trace(go.Scatter(x=random_x, y=random_y0, mode='lines', name='lines'))
fig.add_trace(go.Scatter(x=random_x, y=random_y1, mode='lines+markers',␣
 ↪name='lines+markers'))
fig.add_trace(go.Scatter(x=random_x, y=random_y2, mode='markers',␣
 ↪name='markers'))

fig.show()
```

[95]:
```
# Change the marker size
fig = go.Figure()
fig.add_trace(go.Scatter(
    x=ob_month.Illnesses,
    y=ob_month.Hospitalizations,
    mode = 'markers',
    marker_size=ob_month.Fatalities,
    marker_color='indianred'

# Below are different formatting options to try.

    #marker_color = ob_month.Fatalities
    #marker=dict(
    #    size=16,
```

```
    #    color=ob_month.Fatalities, #set color equal to a variable
    #    colorscale='inferno', # one of plotly colorscales
    #    showscale=True
    #)
))
#fig.update_traces(mode='markers', marker_line_width=2, marker_size=ob_month.
 ↪Fatalities)
# If multiple traces exist, the update will be applied to all traces.

#fig.update_layout(title='Sized Scatterplot')

# Update the x axes
#fig.update_xaxes(tickangle = 90,title_text = "Illnesses",title_font={"size":␣
 ↪20},title_standoff = 25)
#fig.update_xaxes(showline=True, linewidth=2, linecolor='black')
#fig.update_xaxes(showgrid=False)

# Update the x axes
#fig.update_yaxes(title_text = "Hospitalizations",title_standoff = 25)
#fig.update_yaxes(showline=True, linewidth=2, linecolor='black')
#fig.update_yaxes(title_font=dict(size=18, family='Courier', color='crimson'))
#fig.update_yaxes(ticklabelposition="inside top", title='Hospitalizations')



fig.show()

# https://plotly.com/python/builtin-colorscales/
```

[96]:
```
# Using a large dataset - from documentation
N = 100000
fig = go.Figure(data=go.Scattergl(
    x = np.random.randn(N),
    y = np.random.randn(N),
    mode='markers',
    marker=dict(
        color=np.random.randn(N),
        colorscale='Viridis',
        line_width=1
    )
))
fig.show()
```

4.4 Line Charts

When using graph objects, line charts are scatter charts with connected marks.

[97]:
```
# Line charts are Scatter charts with connected markers.
# The default scatter creates a line

fig = go.Figure(go.Scatter(x=oby.Year, y=oby.Illnesses))
fig.show()
```

[98]:
```
fig = go.Figure()

fig.add_trace(go.Scatter(x=oby.Year,
                         y=oby.Illnesses,
                         name = 'Illnesses'))

fig.add_trace(go.Scatter(x=oby.Year,
                        y=oby.Hospitalizations,
                        name = 'Hospitalizations',
                        line=dict(color='lightgrey', width=4, dash='dot')))
# dash options include 'dash', 'dot', and 'dashdot'

fig.add_trace(go.Scatter(x=oby.Year,
                         y=oby.Fatalities,
                         name = 'Fatalities'))

fig.update_layout(title='Illnesses by Year',
                  xaxis_title='Year',
                  yaxis_title='Number of Illnesses')


fig.show()
```

5 Exercise - 30 minutes

- Create a new notebook (don't forget the imports)
- Name the notebook **Diabetes Analysis Dashboard**
- read in the diabetes_for_plotly dataset
- group data as needed
- Use express or graph objects
- Create a scatter plot of any two measures. Use a third measure to adjust the size. Color by a categorical value. Add hover text to show the age group.
- Create a side-by-side bar chart showing number of lab procedures and number of non lab procedures by gender.
- Create a line chart showing number of number of medications by month.
- Create a line chart showing number of number of procedures by month.
- Create a fifth chart of your choice (NOT scatter, bar or line) using the documentation.

https://bitbucket.org/jimcody/sampledata/raw/b2aa6df015816ec35afc482b53df1b7ca7a31f80/diabetes_for_plotly

Instructor solution below.

[99]:
```
import pandas as pd
import plotly.express as px
import plotly.graph_objects as go

diabetes = pd.read_csv('https://bitbucket.org/jimcody/sampledata/raw/
 ↪b2aa6df015816ec35afc482b53df1b7ca7a31f80/diabetes_for_plotly.csv')
diabetes.head()
```

[99]:

	encounter_id	patient_nbr	race	month	year	gender	age
0	2278392	8222157	Caucasian	2	2020	Female	[20-30)
1	149190	55629189	Caucasian	11	2021	Female	[20-30)
2	64410	86047875	AfricanAmerican	6	2019	female	[20-30)
3	500364	82442376	Caucasian	10	2020	Mle	[30-40)
4	16680	42519267	Caucasian	1	2020	M	[40-50)

	admission_type_id	discharge_disposition_id	time_in_hospital
0	6	25	1
1	1	1	3
2	1	1	2
3	1	1	2
4	1	1	1

	num_lab_procedures	num_procedures	num_medications	diag_1	A1Cresult
0	41	0	1.0	250.83	None
1	59	0	NaN	276	None
2	11	5	13.0	648	None
3	44	1	NaN	8	None
4	51	0	8.0	197	None

	insulin	diabetesMed	readmitted
0	No	No	NO
1	Up	Yes	>30
2	No	Yes	NO
3	Up	Yes	NO
4	Steady	Yes	NO

[100]:
```
diabetes['gender'] = diabetes['gender'].replace({'M':'Male', 'Mle':'Male', 'F':
 ↪'Female',
                                                  'female':'Female', 'male':
 ↪'Male',
                                                  '?':'Female', 'Unknown/
 ↪Invalid':'Female'})
```

[101]:
```
# Create a scatter plot of any two measures. Use a third measure to adjust the␣
 ↪size. Color by a categorical value.
# Add hover text to show the age group.
fig = px.scatter(diabetes, x=diabetes.num_lab_procedures,
```

```
                  y=diabetes.num_medications,
                  size = diabetes.time_in_hospital,
                  color = diabetes.gender,
                  hover_data = ['age'])
fig.show()
```

[102]:
```
fig = go.Figure()
fig.add_trace(go.Scatter(
    x=diabetes.num_lab_procedures,
    y=diabetes.num_medications,
    mode = 'markers',
    #marker_color='indianred'
    marker_color = diabetes.time_in_hospital
))

fig.show()
```

[103]:
```
# Create a side-by-side bar chart showing number of lab procedures and number␣
 ↪of non lab procedures by gender.
d_gender = diabetes.groupby('gender').sum().reset_index()
fig = px.bar(d_gender, x='gender', y=['num_lab_procedures', 'num_procedures'],␣
 ↪barmode = 'group')
fig.show()
```

[104]:
```
fig = go.Figure(
    data=[go.Bar(name = 'labs', x=d_gender.gender, y = d_gender.
 ↪num_lab_procedures),
          go.Bar(name = 'non labs', x=d_gender.gender, y = d_gender.
 ↪num_procedures)],
    layout=go.Layout(
        title=go.layout.Title(text="A Figure Specified By A Graph Object")
    )
)
fig.show()
```

[105]:
```
# Create a line chart showing number of number of medications by monmth.

d_month = diabetes.groupby('month').sum().reset_index()
#d_month = d_month.sort_values('month')
fig = px.line(d_month,x='month', y='num_medications')
fig.show()

# fig = go.Figure(go.Scatter(x=d_month.month, y=d_month.
 ↪num_medications,mode='lines')) DEFAULT is a line
```

[106]:
```
# Create a line chart showing number of number of procedures by month.

#d_month = diabetes.groupby('month').sum().reset_index()
#d_month = d_month.sort_values('month')
fig = px.line(d_month,x='month', y='num_procedures')
fig.show()

# fig = go.Figure(go.Scatter(x=d_month.month, y=d_month.num_procedures,␣
  ↪mode='lines'))
```