4-Plotly

January 28, 2021

1 Plotly

Plotly è una libreria grafica scritta in **javascript** che possiede un **wrapper** che ne permette l'utilizzo di funzionalità in python, per poterlo usare ricordate di andare a vedere il suo uso nella lezione 3-pandas.

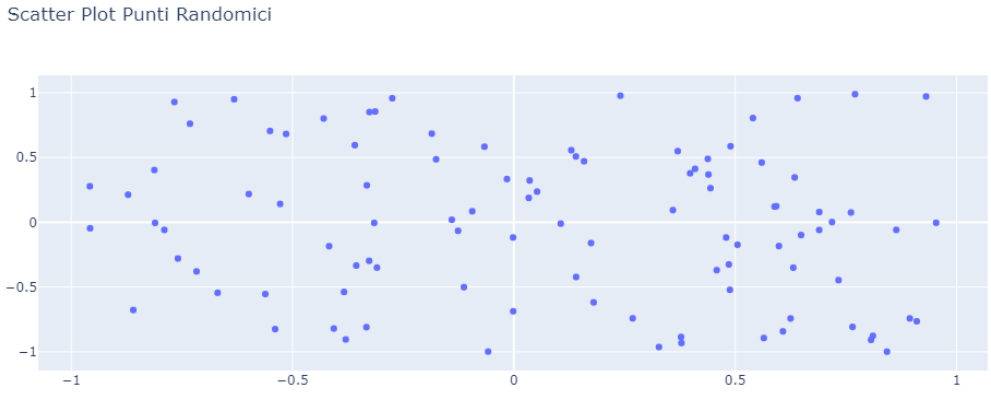
1.1 Creazione grafici

La libreria Plotly per i grafici si basa principalmente su due moduli plotly.graph_objects e plotly.express per sapere esattamente la struttura di una figurazione la documentazione le dedica un intera **pagina**. Entrambi i moduli richiedono che i dati siano passati in formato di **dizionario**, quindi qualora aveste dubbi tornate a fare un salto al capitolo 1 alla lezione 3-Strutture dati.

1.1.1 Graph objects

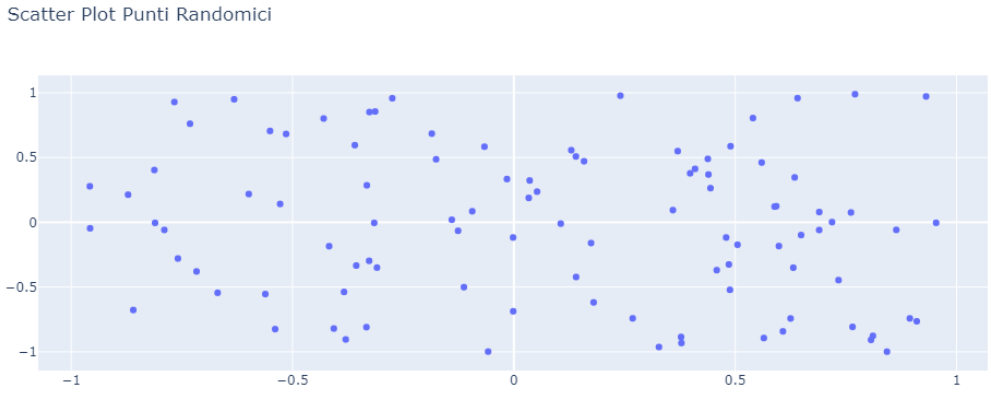
Il modulo graph_object richiede necessariamente un dizionario al suo interno, mentre express no in generale come vedremo in seguito. Per creare un immagine dobbiamo in prima battuta preparare dei dati, utilizzando una funzione **pseudo-randomica** creeremo 100 punti compresi tra -1 e 1.

[1]:
```
import numpy as np
import plotly.graph_objects as go

#La funzione restituisce un valore compreso tra 0 e 1
# per farlo diventare tra -1 e 1 moltiplico per due e sottraggo 1
x_values = 2 * np.random.random_sample((100,)) - 1
y_values = 2 * np.random.random_sample((100,)) - 1

#creiamo ora la figura
fig = go.Figure(data = go.Scatter(
    x = x_values,
    y = y_values,
    mode = "markers"
))

#aggiungiamo il titolo
fig.update_layout(title = "Scatter Plot Punti Randomici")

#mostriamo il grafico
fig.show()
```



Qualora voleste creare una figura da un dizionario ciò è possibile farlo usando questo modo

[2]:
```
import plotly.io as pio

setting = dict({
    "data": [{"type": "scatter",
              "x": x_values,
              "y": y_values,
              "mode" : "markers"}],
    "layout": {"title": {"text": "Scatter Plot Punti Randomici"}}
})

pio.show(setting)
```



Ovviamente sono presenti molti altri tipi di attributi all'interno di plotly, per avere un'idea di quali siano e come sia stato settato il grafico è sufficiente stampare l'**oggetto Figure**, pertanto facendolo otteniamo:

[3]:
```
print(fig)
```

```
Figure({
    'data': [{'mode': 'markers',
              'type': 'scatter',
              'x': array([-0.528071  ,  0.12963225, -0.05816906,  0.6408044 ,
0.95382341,
                           0.77061649,  0.58887074,  0.76493922,  0.48806006,
-0.33220472,
                          -0.81214686,  0.37848436, -0.53934425,  0.68962521,
0.39817738,
                          -0.00180495,  0.59863336,  0.84274641, -0.30930294,
0.37766199,
                          -0.32697453,  0.81102449,  0.76140481,  0.63427064,
-0.73166621,
                           0.60774914,  0.4394623 ,  0.05240348,  0.55974255,
-0.59893534,
                           0.18009226, -0.18543036, -0.31338913,  0.03560556,
-0.35609669,
                           0.10583991, -0.38370798, -0.71688682,  0.8638679 ,
0.73334794,
                          -0.76694465,  0.44413292,  0.47947131,  0.68985753,
-0.63192639,
                          -0.51460471, -0.56151652,  0.64880439, -0.14024612,
0.45827307,
                           0.63119533, -0.4067729 ,  0.80678894, -0.95729617,
0.56463758,
                           0.71866935, -0.81058596, -0.12610168, -0.27473014,
0.48564741,
                          -0.7895986 , -0.09407826, -0.42972159, -0.35937055,
0.14051284,
                          -0.00151989,  0.03357591,  0.50521355,  0.54003565,
0.93113521,
                          -0.6692469 , -0.37992293,  0.91007832,  0.14009455,
-0.75895789,
                          -0.11287798, -0.17577624,  0.15848698, -0.31552179,
0.59271589,
                          -0.55084691, -0.32629022,  0.4380671 , -0.87155887,
0.17428563,
                           0.24064611, -0.06647951,  0.26835715,  0.40923973,
0.48943243,
                           0.35921625,  0.89437596,  0.62503928, -0.33312298,
-0.95787324,
                          -0.41737473, -0.01563642,  0.32767829,  0.37003556,
```

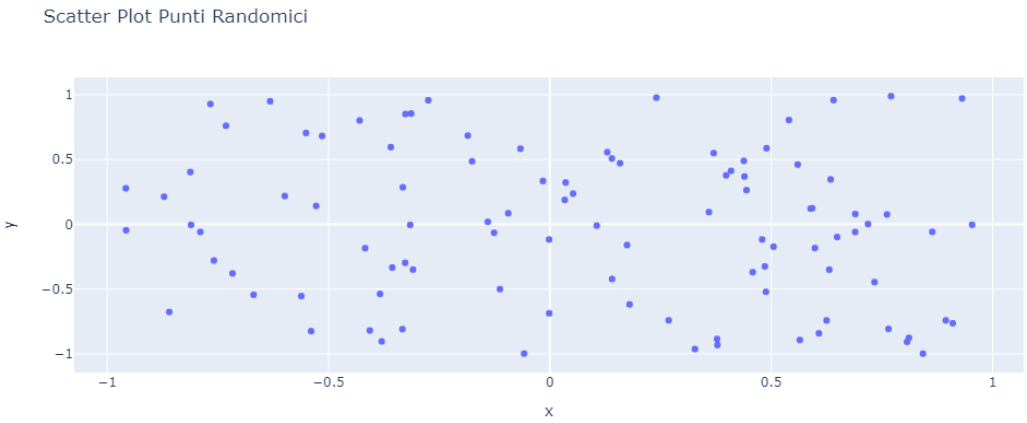
```
-0.8597695 ]),
              'y': array([ 0.14177518,  0.55682887, -0.99819617,  0.95833139,
-0.00391792,
                           0.98894901,  0.12131646, -0.8072405 , -0.52101223,
0.28547657,
                           0.40367768, -0.93175363, -0.82453253, -0.05956496,
0.37829579,
                          -0.11713811, -0.18336418, -0.99866669, -0.34964912,
-0.88511165,
                          -0.29750243, -0.87662508,  0.07594317,  0.34659729,
0.76101512,
                          -0.84187757,  0.36877971,  0.23679831,  0.46127258,
0.21803677,
                          -0.61837068,  0.68448372,  0.85516272,  0.32277987,
-0.33401889,
                          -0.01027115, -0.53744273, -0.37922794, -0.05853348,
-0.44615781,
                           0.92848839,  0.26380819, -0.11739469,  0.07923887,
0.94966043,
                           0.68192566, -0.55389344, -0.09830461,  0.01961537,
-0.36943675,
                          -0.35027736, -0.82002073, -0.90762691, -0.04621564,
-0.89326399,
                           0.00174467, -0.00434544, -0.06548718,  0.95747921,
-0.32548626,
                          -0.05896528,  0.08533725,  0.8012487 ,  0.595488  ,
-0.42211055,
                          -0.68689871,  0.18834034, -0.17312222,  0.80476964,
0.97148383,
                          -0.54410899, -0.90416195, -0.76409102,  0.50822998,
-0.27937849,
                          -0.50073509,  0.48651526,  0.47137192, -0.00449917,
0.12421092,
                           0.70519932,  0.8512218 ,  0.48947947,  0.21324501,
-0.16015209,
                           0.97745808,  0.58378244, -0.74152057,  0.41252825,
0.58729608,
                           0.09410244, -0.74193386, -0.74224274, -0.80921905,
0.27809596,
                          -0.18435918,  0.33389385, -0.9629176 ,  0.54959764,
-0.67711424])}],
    'layout': {'template': '…', 'title': {'text': 'Scatter Plot Punti
Randomici'}}
})
```

Qualora voleste le impostazioni e i dati importati in formato dizionario, potete usare il comando `fig.to_dict()` utile per esportare le vostre importazioni, qualora lo voleste in formato JSON(JavaScript Object Notation) è sufficiente usare `fig.to_json()`.

1.1.2 Plotly express

PLotly express è il modulo consigliato dalla documentazione per l'utilizzo, infatti questo modulo possiede molte funzionalità, dati e molto altro al suo interno, vediamo per esempio come ricreare i grafici precedenti.

[4]:
```
import plotly.express as px

fig = px.scatter(x = x_values, y = y_values, title = "Scatter Plot Punti␣
↪Randomici")
fig.show()
```



Come possiamo vedere questo modo è molto più compatto e veloce, anche qui funziona la possibilità di stampare ed esportare le impostazioni.

[5]:
```
print(fig)
```

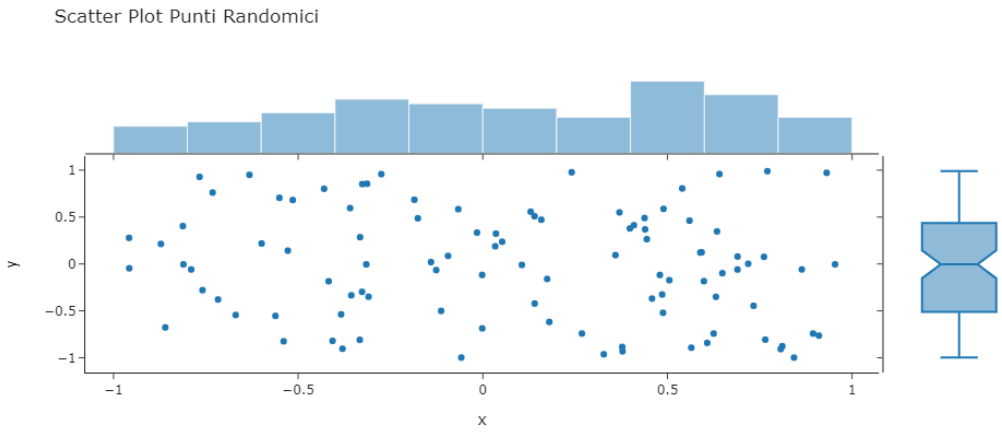
```
Figure({
    'data': [{'hovertemplate': 'x=%{x}<br>y=%{y}<extra></extra>',
              'legendgroup': '',
              'marker': {'color': '#636efa', 'symbol': 'circle'},
              'mode': 'markers',
              'name': '',
              'orientation': 'v',
              'showlegend': False,
              'type': 'scatter',
              'x': array([-0.528071  ,  0.12963225, -0.05816906,  0.6408044 ,
0.95382341,
                           0.77061649,  0.58887074,  0.76493922,  0.48806006,
-0.33220472,
                          -0.81214686,  0.37848436, -0.53934425,  0.68962521,
0.39817738,
```

```
                        -0.00180495,  0.59863336,  0.84274641, -0.30930294,
0.37766199,
                        -0.32697453,  0.81102449,  0.76140481,  0.63427064,
-0.73166621,
                         0.60774914,  0.4394623 ,  0.05240348,  0.55974255,
-0.59893534,
                         0.18009226, -0.18543036, -0.31338913,  0.03560556,
-0.35609669,
                         0.10583991, -0.38370798, -0.71688682,  0.8638679 ,
0.73334794,
                        -0.76694465,  0.44413292,  0.47947131,  0.68985753,
-0.63192639,
                        -0.51460471, -0.56151652,  0.64880439, -0.14024612,
0.45827307,
                         0.63119533, -0.4067729 ,  0.80678894, -0.95729617,
0.56463758,
                         0.71866935, -0.81058596, -0.12610168, -0.27473014,
0.48564741,
                        -0.7895986 , -0.09407826, -0.42972159, -0.35937055,
0.14051284,
                        -0.00151989,  0.03357591,  0.50521355,  0.54003565,
0.93113521,
                        -0.6692469 , -0.37992293,  0.91007832,  0.14009455,
-0.75895789,
                        -0.11287798, -0.17577624,  0.15848698, -0.31552179,
0.59271589,
                        -0.55084691, -0.32629022,  0.4380671 , -0.87155887,
0.17428563,
                         0.24064611, -0.06647951,  0.26835715,  0.40923973,
0.48943243,
                         0.35921625,  0.89437596,  0.62503928, -0.33312298,
-0.95787324,
                        -0.41737473, -0.01563642,  0.32767829,  0.37003556,
-0.8597695 ]),
             'xaxis': 'x',
             'y': array([ 0.14177518,  0.55682887, -0.99819617,  0.95833139,
-0.00391792,
                         0.98894901,  0.12131646, -0.8072405 , -0.52101223,
0.28547657,
                         0.40367768, -0.93175363, -0.82453253, -0.05956496,
0.37829579,
                        -0.11713811, -0.18336418, -0.99866669, -0.34964912,
-0.88511165,
                        -0.29750243, -0.87662508,  0.07594317,  0.34659729,
0.76101512,
                        -0.84187757,  0.36877971,  0.23679831,  0.46127258,
0.21803677,
                        -0.61837068,  0.68448372,  0.85516272,  0.32277987,
```

```
-0.33401889,
                           -0.01027115, -0.53744273, -0.37922794, -0.05853348,
-0.44615781,
                            0.92848839,  0.26380819, -0.11739469,  0.07923887,
0.94966043,
                            0.68192566, -0.55389344, -0.09830461,  0.01961537,
-0.36943675,
                           -0.35027736, -0.82002073, -0.90762691, -0.04621564,
-0.89326399,
                            0.00174467, -0.00434544, -0.06548718,  0.95747921,
-0.32548626,
                           -0.05896528,  0.08533725,  0.8012487 ,  0.595488  ,
-0.42211055,
                           -0.68689871,  0.18834034, -0.17312222,  0.80476964,
0.97148383,
                           -0.54410899, -0.90416195, -0.76409102,  0.50822998,
-0.27937849,
                           -0.50073509,  0.48651526,  0.47137192, -0.00449917,
0.12421092,
                            0.70519932,  0.8512218 ,  0.48947947,  0.21324501,
-0.16015209,
                            0.97745808,  0.58378244, -0.74152057,  0.41252825,
0.58729608,
                            0.09410244, -0.74193386, -0.74224274, -0.80921905,
0.27809596,
                           -0.18435918,  0.33389385, -0.9629176 ,  0.54959764,
-0.67711424]),
              'yaxis': 'y'}],
    'layout': {'legend': {'tracegroupgap': 0},
               'template': '…',
               'title': {'text': 'Scatter Plot Punti Randomici'},
               'xaxis': {'anchor': 'y', 'domain': [0.0, 1.0], 'title': {'text':
'x'}},
               'yaxis': {'anchor': 'x', 'domain': [0.0, 1.0], 'title': {'text':
'y'}}}
})
```

Questa libreria però è molto più flessibile permettendo di fare grafici molto più complessi, come:

[6]:
```
fig = px.scatter(x = x_values, y = y_values, title = "Scatter Plot Punti␣
↪Randomici",
                 marginal_x = "histogram", marginal_y = "box", template =␣
↪"simple_white" )
fig.show()
```



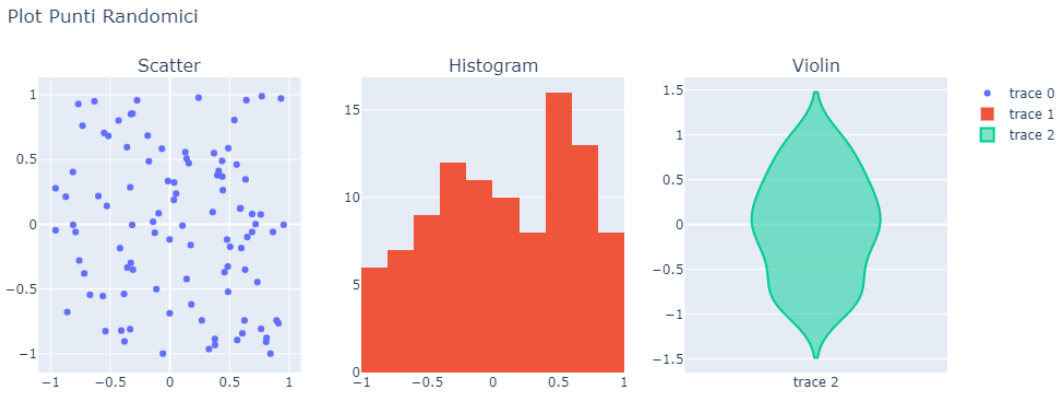
Attraversi questi comandi siamo riusciti ad effettuare grafici ai margini di diverso tipo.

1.1.3 Subplots

Qualora volessimo invece dividere i grafici fatti in precedenza, abbiamo la possibilità di usare i Subplot ovvero creare all'interno dell'oggetto Figure molteplici Plots di tipo diversi, qualora fossero ancora uguali è ancora possibile usare `plotly.express`(**link**), nel caso invece siano diversi è necessario ricorrere a `graph_object` e al metodo `make_subplots`.

[7]:
```
from plotly.subplots import make_subplots

#definiamo il numero di righe e colonne come una matrice
#questo impatterà sulla visione dei plots e aggiungiamo i titoli di ognuno
fig = make_subplots(rows = 1, cols = 3, subplot_titles=("Scatter", "Histogram",␣
 ↪"Violin"))

#creiamo i subplots diversi creando una traccia
fig.add_trace(
    #lo scatter plot
    go.Scatter(x = x_values, y = y_values, mode = "markers"),
    row=1, col=1 # è la posizione del plot
)

fig.add_trace(
    #l'istogramma
    go.Histogram(x = x_values, y = y_values),
    row = 1, col = 2
)

fig.add_trace(
    #Il Violino
```

```
    go.Violin(y = y_values),
    row = 1, col = 3
)

fig.update_layout(title_text = "Plot Punti Randomici")

fig.show()
```



Ovviamente le funzionalità di subplots sono numerose ed per questo vi consiglio di fare riferimento alla sua sezione nella **documentazione**.

1.2 Esportare grafici

Per esportare i grafici è molto importante aver installato il pacchetto kaleido nel caso delle **immagini statiche**, per le dinamiche in formato web non è necessario, per installarlo aprite **Anaconda Prompt(terminal)** e digitate:

[ ]:
```
conda install -c conda-forge python-kaleido
```

Qualora lo abbiate installato ora avete la possibilità di esportare i grafici in due modi: ***interattivi*** o ***statici***.

1.2.1 Grafici interattivi

I grafici interattivi sarebbe grafici in cui è possibile interagire sull'elemento zoomando, modificando o salvando parti l'immagine originale, in genere per poterlo fare dovete prendere **l'oggetto Figure** ed esportarlo, qualora abbiate necessità di navigare le directory usate la libreria **os** (se avete problemi usate questa **guida iniziale**.

[8]:
```
fig.write_html("../img/plot.html")
```

1.2.2 Grafici statici

I grafici statici sono invece grafici che possono essere solo visualizzati e a cui non è possibile interagire, tra i formati supportati da plotly sono presenti **webp(webpage), PNG, JPEG, SVG, PDF** in base alle vostre necessità potete scegliere il formato, ricordate solo che a differenza delle interattive si usa il comando `write_img` e non `write_html`.

[9]:
```
fig.write_image("../img/plot.pdf")
```

In alcuni casi il formato con cui si vede il file con altri programmi può essere diverso da quello impostato, la motivazione risiede nel fatto che plotly cerca di interfacciarsi con il software o web browser in uso per l'esportazione delle immagini, per risolvere ciò è possibile usare i **renders**.

COMPLIMENTI, AVETE COMPLETATO IL CAPITOLO 2 E LA LEZIONE DI PLOTLY, ADESSO SI PARTE CON IL MACHINE LEARNING!