Dashboard by plotly

Dashboard by Plotly

سنتعرف على كيفية بناء Dash Plotly

لكن اولاً ماهو Dash ؟

هو إطار عمل لإنشاء تطبيقات تفاعلية بتصورات للبيانات في Python وغيرها.

بإستخدام Python نستطيع إنشاء dashboard تفاعلية online للبيانات الخاصة بنا فقط بالـpython, ولا نحتاج الخبرة العالية في HTML, CSS, JavaScript.

لماذا ننشئ Dashboard؟

تعمل Dashboard على زيادة كفاءة إعداد التقارير و توفر الوقت ايضاً في تحسين معرفة البيانات و فهمها و ذلك بدوره سوف ينتج إجابيات كثيرة .

قبل إن نبدا في بناء dash, نحتاج لتحميل بعض الـpackage:

لـPython :

```
pip install dash
```

لـJupyter notebook:

```
pip install jupyter-dash
```

وايضاً لنتعرف على المكتبات التي سوف نستخدمها:

- الاولى jupyter dash:
  هي فقط لإنشاء تطبيق Dash

- الثانية هي Dash HTML Components:
  بدلاً من كتابة HTML أو استخدام محرك قوالب HTML ، يمكنك إنشاء تخطيطك باستخدام Python مع Dash HTML Components
  و dash.html هو المودل الذي يحتوي على مكونات لكل علامة HTML مثل
  مثال على ذلك :

```
html.H2('DS Bootcamp - Tuwaiq Academy', style={'textAlign':'center'})
```

حين نكتب مثل هذا السطر لكن بالـ HTML syntax:

```
<h2 style = "text-align:center;">DS Bootcamp - Tuwaiq Academy <h2/>
```

للإطلاع أكثر هنا

- الثالثة هي Dash Core Components:
تحتوي مكتبة Dash Core Component على مجموعة من المكونات عالية المستوى مثل أشرطة التمرير والرسوم البيانية والقوائم المنسدلة والجداول, الخ.
و dash.dcc هو مودل الذي يحتوي على المكونات , و بعض هذه المكونات :
  - dropdown
  - slider
  - rangeSlider
  - input
  - textarea
  - checkboxes
  - Radio Items

للإطلاع أكثر على هذه المكونات هنا

لنقوم الان ببناء dash:

عادةً اول ما نقوم بكتابة في إي لغة برمجية جديدة "Hello, World!", لنقوم بكتبتها في أول Dash:

Hello, World!

- اولاً نقوم بإستتعاد المكتبات و الحزمات التي نحتاجها:

```
from jupyter_dash import JupyterDash
from dash import html
```

- ثانياً نقوم بإنشاء الـapp الخاص بـالـdash:

هنا قمنا بإنشاء dash object

```
app = JupyterDash(name)
```

هنا لتشغيل الـapp

```
if name__ == '__main':
app.run_server()
```

حين نشغل هذا الكود البسيط سوف تظهر لنا dash فارغة لا تحتوي على اي شيء.
لنقم الان بإضافة الجملة الشهيرة Hello, World!

- بنفس الخطوات السابقة و لكن نضيف:

```
app.layout = html.Div([
   html.H1('Hello, World!')
])
```

المثال الاول :

- نقوم بإستتدعاء الـpackages

```
from jupyter_dash import JupyterDash
import dash_html_components as html
import dash_core_components as dcc
```

- ننشئ JupyterDash

```
app = JupyterDash(__name__)
```

- نحدد التنسيق

```
app.layout = html.Div([
   dcc.Checklist(
   ['New York City', 'Montréal', 'San Francisco'],
   ['New York City', 'Montréal']
)
])
```

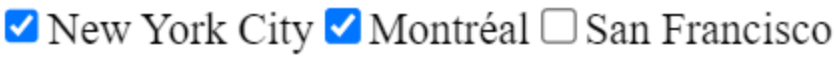
هنا في التنسيق سوف نتعامل مع html بسيط جداً , نلاحظ هنا ان layout = html.
و Div هنا يمثل حاوية عامة يتم استخدامها لتجميع العناصر. ثم من مكتبة Dash Core Component سوف نستدعي Checklist ثم نضيف الخيارات للقائمة .
اما في ['New York City', 'Montréal'] هنا يتم وضع check لهذه الخيارين

- تشغيل الـdash

```
if __name__ == '__main__':
```

```
app.run_server(mode="inline")
```

- المخرج



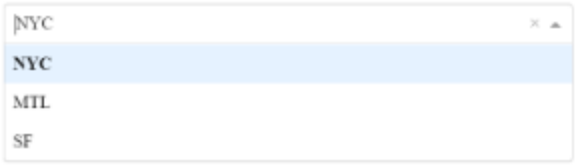
المثال الثاني:
مشابة جداً للمثال السابق لكن سوف نستخدم Dropdown بدلاً من Checklist.
بعد أن قمنا بإنشاء الـdash

- نحدد التنسيق

```
app.layout = html.Div([
   dcc.Dropdown(['NYC', 'MTL', 'SF'],
            'NYC',
            id='demo-dropdown')
])
```

نرا هنا أستخدما Dropdown و اضفنا القيم و هي المدن

- المخرج



المثال الثالث:

```
app.layout = html.Div([
   html.Label('Choose from RangeSlider'),
  dcc.RangeSlider(0, 20, 1, value=[5, 15], id='my-range-slider'),
   html.Div(id='output-container-range-slider')
])
```

نرا هنا تم إضافة label و استخدام مكون جديد وهو RangeSlider, بداخله يتم تحديد الـrange و في المثال اعلاه تم تحديده من 0 الى 20 و تم تحديد الـgap , نقصد به هنا عرض تسلسل الارقام.

نرا انه تم تحديد الـrange من 5 الى 15 و التي هي القيم المحددة داخل value

هذه امثلة بسيطة جداً و منفردة , سوف نتعرف على الكثير من Components ثم سوف نعمل على Dash متكامل.

الآن نحتاج إلى مكتبة جديدة وهي:

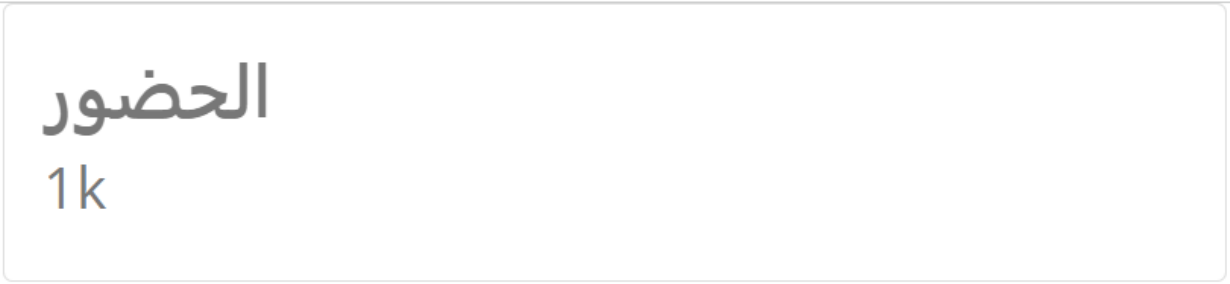
- Dash Bootstrap Components

مكتبة لـ Plotly Dash تتيح لك تخصيص تخطيط تطبيقك لـDash باستخدام مكونات ومظاهر Bootstrap وتمنح تطبيقك نمطًا متناسقًا

لتحميل هذه المكتبة :

```
pip install dash-bootstrap-components
```

لنكمل الآن , سوف نقوم بإضافة Card:



هذه ابسط card في dash بإستخدام bootstrap, لنقوم بها:

- اولاً المكتبات

```
import dash_bootstrap_components as dbc
from jupyter_dash import JupyterDash
from dash import html
```

- ثانياً نقوم إنشاء الـapp

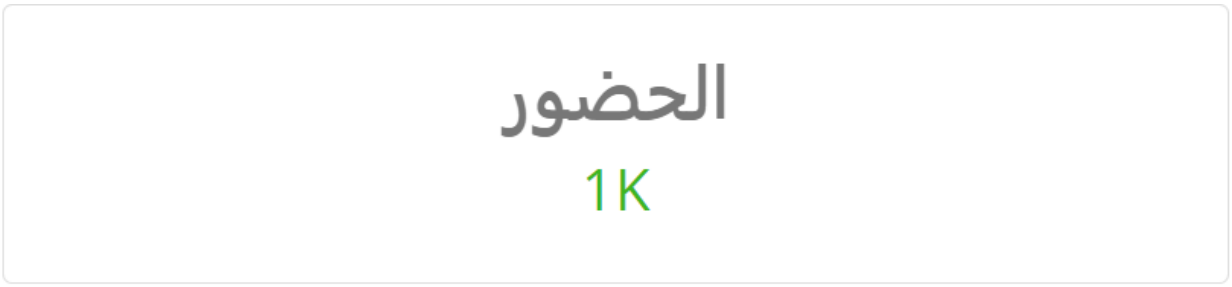
```
app = JupyterDash(__name__, external_stylesheets=[dbc.themes.SPACELAB,
dbc.icons.BOOTSTRAP])
```

- ثالثاً نقوم بإنشاء الـcard

```
card =  dbc.Card(
   dbc.CardBody(
      [
         html.H1("الحضور"),
         html.H3("1k")
      ],
   ),
)
app.layout=dbc.Container(card)
```

تم إستدعاء card من dbc وهي dash bootstrap components, ومن ثم CardBody . اما dbc.Container(card) فهي فقط تستخدم الـContainer لتوسيط محتوى تطبيقك ووضعه أفقيًا.

لنضيف بعض من الـstyle




```
card =  dbc.Card(
    dbc.CardBody(
       [
          html.H1("الحضور"),
          html.H3("1K", className="text-success")
       ],
    ),
    className="text-center"
 )

app.layout=dbc.Container(card)
```

نلاحظ انه مشابة جداً الكود السابق, لكن تم إضافة className وهي classes بتنسيق و تصميم جاهز في bootstrap.

ولان نضيف رمز و خلفية للـCard


```
card = dbc.Card(
    dbc.CardBody(
        [
            html.H1(children=[html.I(className="bi bi-people-fill"), " الحضور"]),
            html.H3("1K"),
            html.H6(html.I("في أكاديمية طويق", className=" text-success")),
        ],
    ),
    className="text-center m-4 bg-primary text-white",
)
```

هنا "className="bi bi-people-fill هو إضافة الرمز و يوجد الكثير من الرموز و لتصفحها هنا

Task One:

- [ ] عمل Dash بأربع Cards بسيطه مع إضافة icons.

- اولاً نقوم بإستدعاء المكتبات

```
import dash_bootstrap_components as dbc
```

```
from jupyter_dash import JupyterDash
from dash import html
```

- ثانياً نقوم بإنشاء الـapp

```
app = JupyterDash(name, external_stylesheets=[dbc.themes.SPACELAB])
```

نلاحظ هنا أنا قمنا بإضافة external_stylesheets و هنا نحدد dash style, فانقوم بإستتدعاء bootstrap, للإطلاع على جميع themes هنا.

- ثالثاً نقوم بتعرف على Card

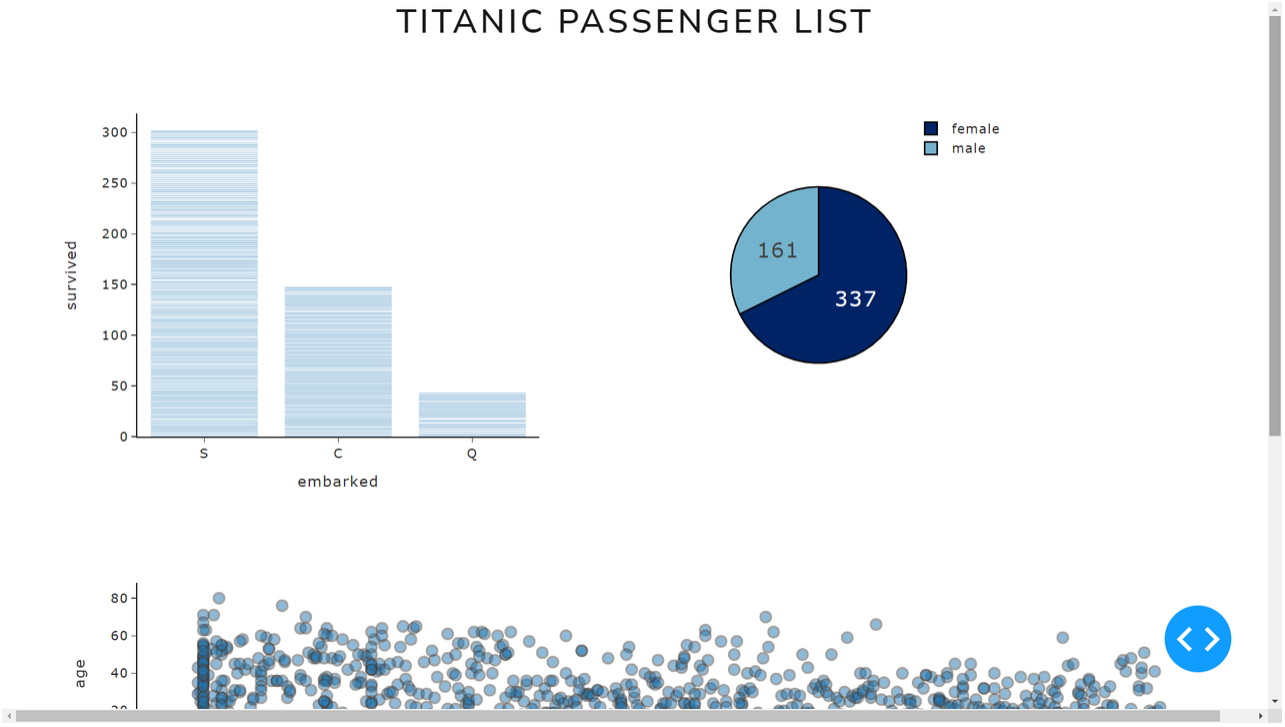
```
img = 'الصوره.png'
card = dbc.Card([
   dbc.CardImg(src=img, top=True),
   dbc.CardBody(
      [
         html.H3("Data Science Bootcamp", className="text-primary"),
         html.Div("5 weeks, 6 hours"),
         html.Div("Tuwaiq Academy,2023."),
      ]
   )],
   className="shadow my-2",
   style={"maxWidth": 350},
)
```

نلاحظ ان تم إستدعاء card من dbc وهي dash bootstrap components, ثم تم استدعاء CardImg و تحديد الصوره و من بعدها CardBody وهو محتوى card و نلاحظ className="text-primary" هو class في bootstrap جاهز و نستطيع الاستغناء عنه , وايضا className="shadow my-2" هو class جاهز يتصميم معين و {style={"maxWidth": 350}, لتحديد عرض الـcard .

Task Two:

☐ كيف نضيف background لـ Card

لنقوم الان بعمل Dash بسيط خاص ببيانات خاصة بنا :



اولاً لنتعرف على ماهي Data لهذا الـdash :

- الـdata :

هي titanic passenger list, مفتوحة المصدر

- المكتبات:

```
import pandas as pd
from dash import html
from dash import dcc
import dash_bootstrap_components as dbc
from jupyter_dash import JupyterDash
import pandas as pd
import plotly.express as px
```

- قراءة الـcsv:

```
df = pd.read_csv('titanic passenger list.csv')
df.head()
```

- إنشاء الـdash app:

```
app = JupyterDash(__name__ ,external_stylesheets = [dbc.themes.LUX])
```

- إنشاء الرسمات:

```
colors = ['#002366', '#74b3ce']

fig = px.bar(data_frame=df, x ='embarked', y ='survived', template="simple_white")

fig2 = px.pie(df, values='survived', names='sex' , template="simple_white")
fig2.update_traces(hoverinfo='label+percent', textinfo='value', textfont_size=20,
                   marker=dict(colors=colors, line=dict(color='#000000', width=1.5)))

fig3 = px.scatter(data_frame=df, x="home.dest", y="age", hover_name="survived",
                  template="simple_white",width=1200, height=600)
fig3.update_traces(marker=dict(size=10,
                               line=dict(width=2,
                               color=colors)),
                    opacity=0.5,
                   selector=dict(mode='markers'))
```

- كتابة اكواد html و bootstrap:

```
app.layout = dbc.Container([
    dbc.Row([
        dbc.Col([
            html.H1("Titanic Passenger List", style={'textAlign': 'center'})
        ], width=12)
    ]),

    dbc.Row([
        dbc.Col([
            dcc.Graph(id='our-plot', figure=fig)
        ], width=6),
         dbc.Col([
            dcc.Graph(id='our-plot', figure=fig2)
        ], width=4)

    ]),

    dbc.Row([
           dbc.Col([
            dcc.Graph(id="scatter-plot", figure=fig3)
        ], width=50)
    ]),

])
```

- تشغيل الـdash:

```
if __name__ == '__main__':
    app.run_server()
```

Dash Bootstrap Cheatsheet here