Tecnológico de Monterrey

Analítica de datos y herramientas de inteligencia artificial II

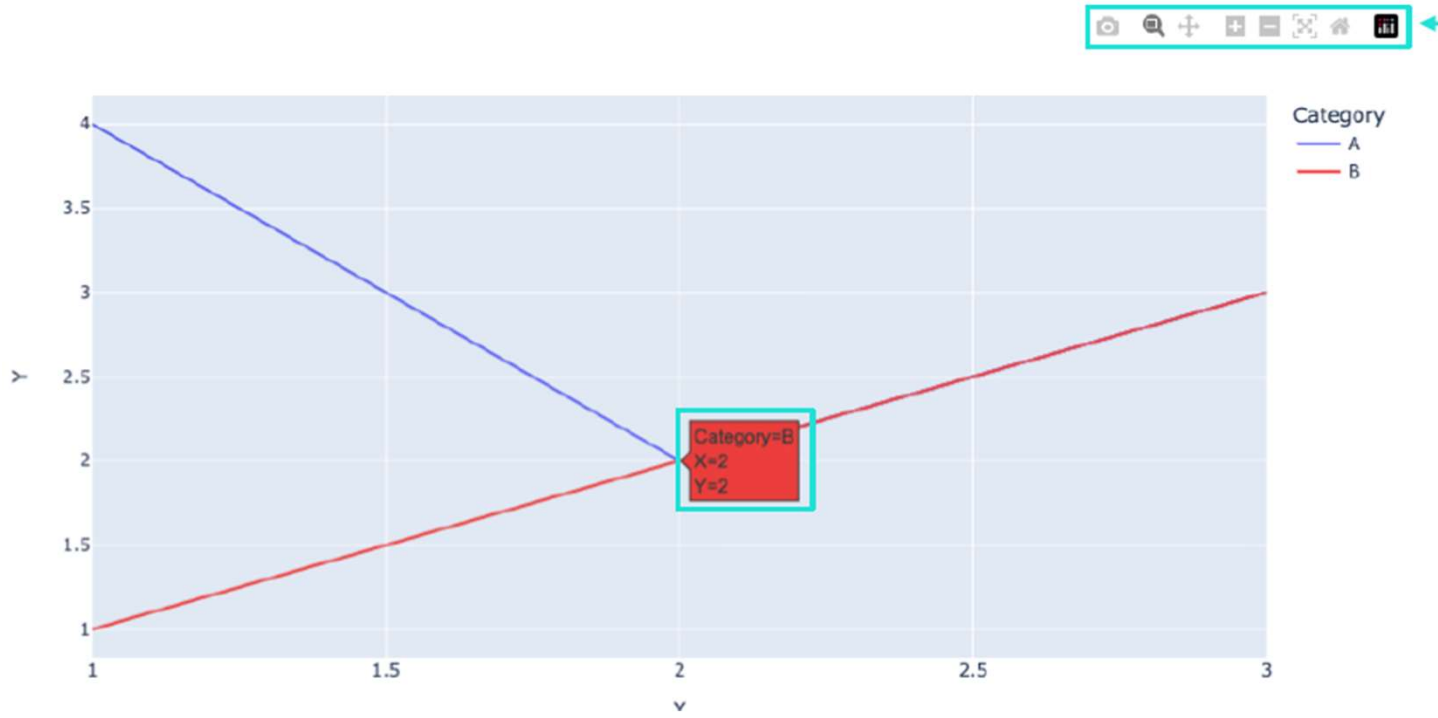
UF6. Desarrollo Web para Apps de Analítica

UF6.2.Frameworks web con Python – Dash Parte3 – Elementos Interactivos

Profr. Gustavo Romero

Interactividad Básica

- Los gráficos Plotly cuentan con funcionas básicas interactivas por defecto



Hover Menu Options:

- *Download Image*
- *Select Zoom Area*
- *Pan*
- *Zoom In*
- *Zoom Out*
- *Auto Scale*
- *Return to Default*

Elementos Interactivos

- Incluidos en el modulo “Dash Core Components” (dcc). Permiten ser usados en aplicaciones Dash para crear dashboards dinámicos con gráficos Plotly
- Su elección considera los **tipos de datos** y **cantidad de opciones** disponibles a usuarios
- Poseen operadores lógicos usados en funciones callback para filtrar visualizaciones seleccionadas

Component	Description	Logical Operators
dcc.Dropdown()	Dropdown list of options for the user to select (or multi-select)	==, !=, in, not in
dcc.Checklist()	Checkboxes with options for the user to select or deselect	==, !=, in, not in
dcc.RadioItems()	Radio buttons with options for the user to toggle between	==, !=, in, not in
dcc.Slider()	Slider with a handle for the user to drag and select values with	==, <, <=, >, >=
dcc.RangeSlider()	Slider with two handles for the user to drag and select ranges with	.between(value[0], value[1])
dcc.DatePickerSingle()	Dropdown calendar for the user to select a date with	==, <, <=, >, >=
dcc.DatePickerRange()	Dropdown calendar for the user to select a date range with	.between(start, end)

Elementos Interactivos

- *La correcta selección considera los tipos de dato, cantidad de opciones y criterio de filtrado (comparaciones lógicas).*
- Usa dropdowns, checklists, y radio buttons para **categorías**. *Checklists y radio buttons son adecuados para un conjunto limitado de opciones, mientras que los dropdown pueden incorporar docenas manteniendo las aplicaciones compactas.*
- Use sliders y date pickers para **números** y **fechas**. Sliders y date pickers simples funcionan bien para desigualdades, mientras que los que permiten rangos son idóneos para lógica “entre” opciones.
- Las funciones Callback pueden tener **múltiples inputs** & **outputs**
  - *Múltiples outputs permiten modificar diversas visualizaciones ó texto utilizando un solo elemento interactivo.*
  - *Múltiples inputs permiten utilizar varios elementos interactivos para modificar texto y visualizaciones en la aplicación*

Dropdown Menus

- *Provee un listado de opciones a seleccionar por el usuario (también permite multi-select)*

- dcc.Dropdown(id, options, value, multi)

Checklists

- *Provee un listado de opciones a seleccionar por el usuario (también permite multi-select)*
- Los checklists son adecuados para un conjunto de 2 a 8 opciones, más allá se convierten en algo complejo. Es mejor considerar un multi-dropdown menú.

- dcc.Checklist(id, options, value)

Radio Buttons

- *Provee un listado de opciones al usuario para seleccionar solo una.*

- dcc.RadioItems(id, options, value)

Sliders

- *Permite al usuario arrastrar un extremo para seleccionar un valor dentro de un rango definido*
- *Frecuentemente se alinea a operadores Booleanos, donde el usuario selecciona opciones* ***igual que****,* ***menor que*** *ó* ***mayor que*** *cierto valor.*

- dcc.Slider(min, max, step, value)

Sliders de Rango

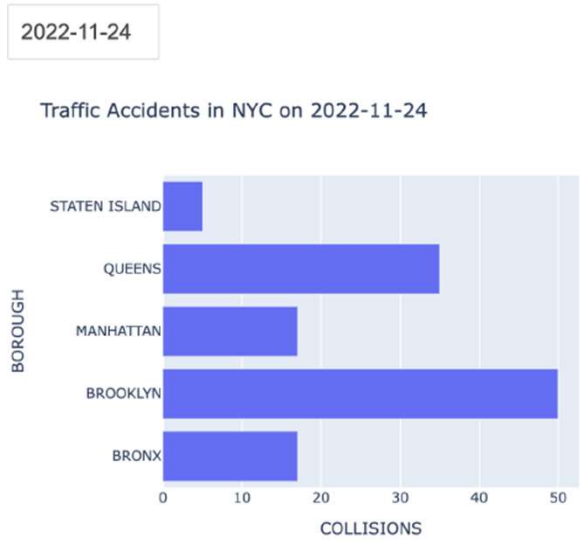
- *Permite al usuario arrastrar ambos extremos para seleccionar un rango de valores*
- *Permiten selecciones “entre” valores en vez de en una simple dirección*
- *Establece una lista ó tupla como el valor*
- *Se filtra el DataFrame usando el método .between() en lugar de un operador Booleano (los valores se acceden con su índice)*

- dcc.RangeSlider(min, max, step, value)

Date Pickers

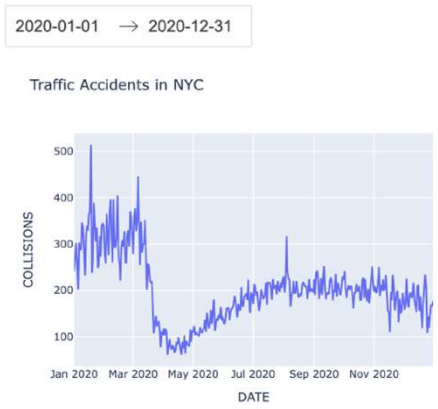
- *Single: Permite al usuario elegir una fecha a partir de un calendario*
- *Se establecen min y max de fechas para la data y max como valor de inicio*
- *La propiedad de input es “date”*
- dcc.DatePickerSingle(id, min_date_allowed, max_date_allowed, initial_visible_month, date, display_format)

```
app = JupyterDash(__name__)

app.layout = html.Div([
    dcc.DatePickerSingle(
        id="date picker",
        min_date_allowed=collisions["DATE"].min(),
        max_date_allowed=collisions["DATE"].max(),
        initial_visible_month=collisions["DATE"].max(),
        date=collisions["DATE"].max(),
        display_format="YYYY-MM-DD"
    ),
    dcc.Graph(id="graph")
])

@app.callback(Output("graph", "figure"), Input("date picker", "date"))
def plot_collisions_bar(date):
    fig = px.bar(
        collisions.loc[collisions["DATE"].eq(date)],
        x="COLLISIONS",
        y="BOROUGH",
        title=f"Traffic Accidents in NYC on {date}"
    )
    return fig

if __name__ == "__main__":
    app.run_server(mode="inline")
```



Date Pickers

- *Range: Permite al usuario elegir un rango de fechas a partir de calendarios*
- *Establece min y max de fechas como los valores de inicio y fin*
- *Se crean dos dropdowns*
- *Se requieren dos inputs que se pasan a la función callback*
- dcc.DatePickerRange(id, start_date, end_date, display_format)




```
app = JupyterDash(__name__)

app.layout = html.Div([
    dcc.DatePickerRange(
        id="dates",
        start_date=collisions["DATE"].min(),
        end_date=collisions["DATE"].max(),
        display_format="YYYY-MM-DD"
    ),
    dcc.Graph(id="graph")
])

@app.callback(
    Output("graph", "figure"),
    [Input("dates", "start_date"),Input("dates", "end_date")]
)
def plot_rev_line(start, end):
    fig = px.line(
        (collisions
         .loc[collisions["DATE"].between(start, end)]
         .groupby("DATE", as_index=False)
         .sum()),
        x="DATE",
        y="COLLISIONS",
        title=f"Traffic Accidents in NYC"
    )
    return fig

if __name__ == "__main__":
    app.run_server(mode="inline")
```

Tecnológico de Monterrey

Multiple Input and Output Callbacks

```
app.layout = html.Div([
    dcc.Dropdown(
        id="X Column Picker",
        options=list(education.select_dtypes(include='number').columns),
        value="expenditure_per_student"
),
    dcc.Dropdown(
        id="Y Column Picker",
        options=list(education.select_dtypes(include='number').columns),
        value="TOTAL_EXPENDITURE"
),
    dcc.Graph(id="graph"),
])

@app.callback(
    Output('graph', 'figure'),
    Input("X Column Picker", "value"),
    Input("Y Column Picker", "value")
)
```

```
app.layout = html.Div([
    html.H2(id="header"),
    dcc.Dropdown(
        id="X Column Picker",
        options=list(education.select_dtypes(include='number').columns),
        value="expenditure_per_student"
),
    dcc.Dropdown(
        id="Y Column Picker",
        options=list(education.select_dtypes(include='number').columns),
        value="TOTAL_EXPENDITURE"
),
    dcc.Graph(id="graph"),
])

@app.callback(
    Output('header', 'children'),
    Output('graph', 'figure'),
    Input("X Column Picker", "value"),
    Input("Y Column Picker", "value")
)
```