McKinsey & Company

20200904 DS Coffee

by Zecai Liang

Dash Tutorial: interactive SHAP plots

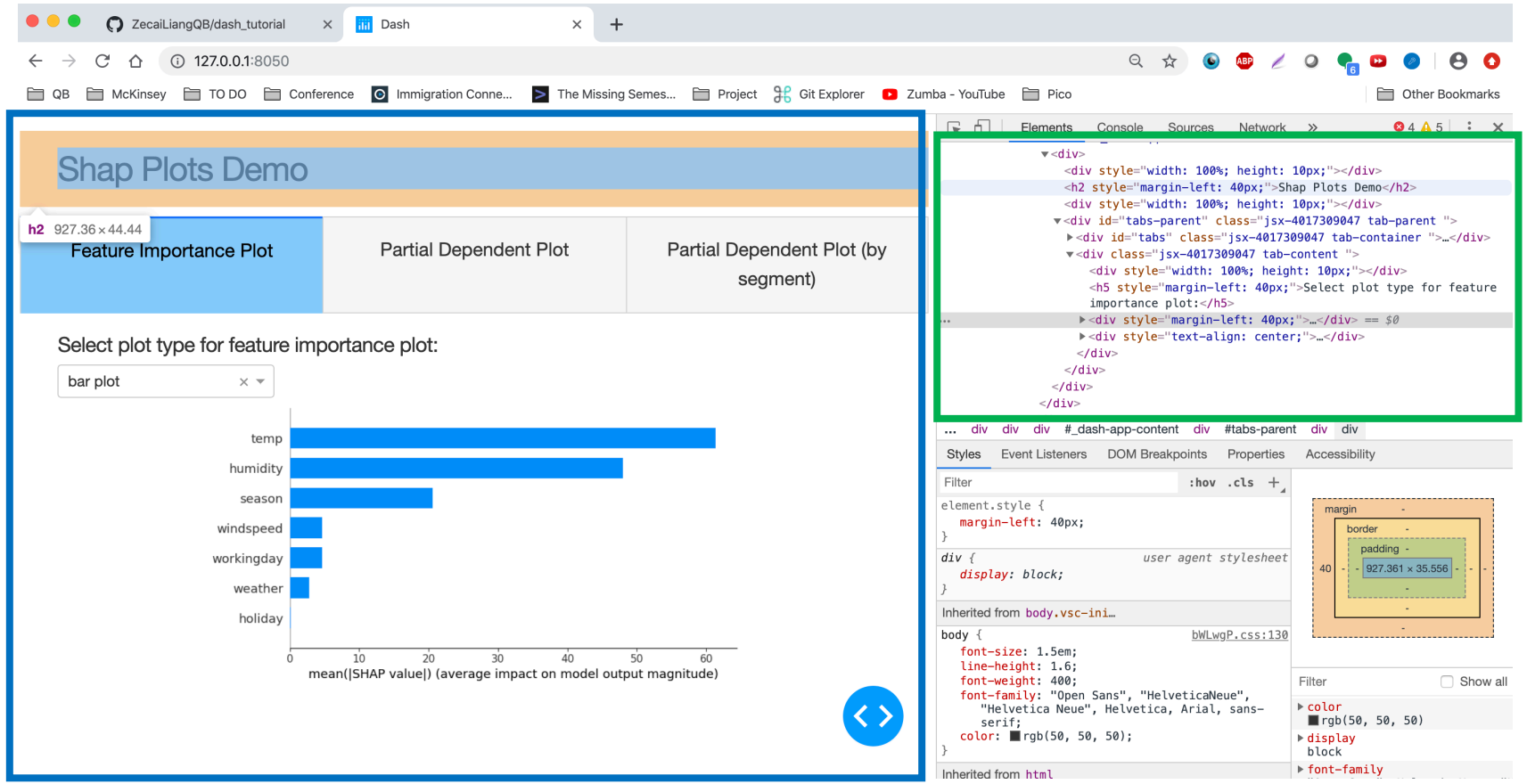
Outline:

1. How does Dash work
2. Minimal components to build a Dash page
3. Demo

code repo:
https://github.com/ZecaiLiangQB/dash_tutorial

1. How does Dash work

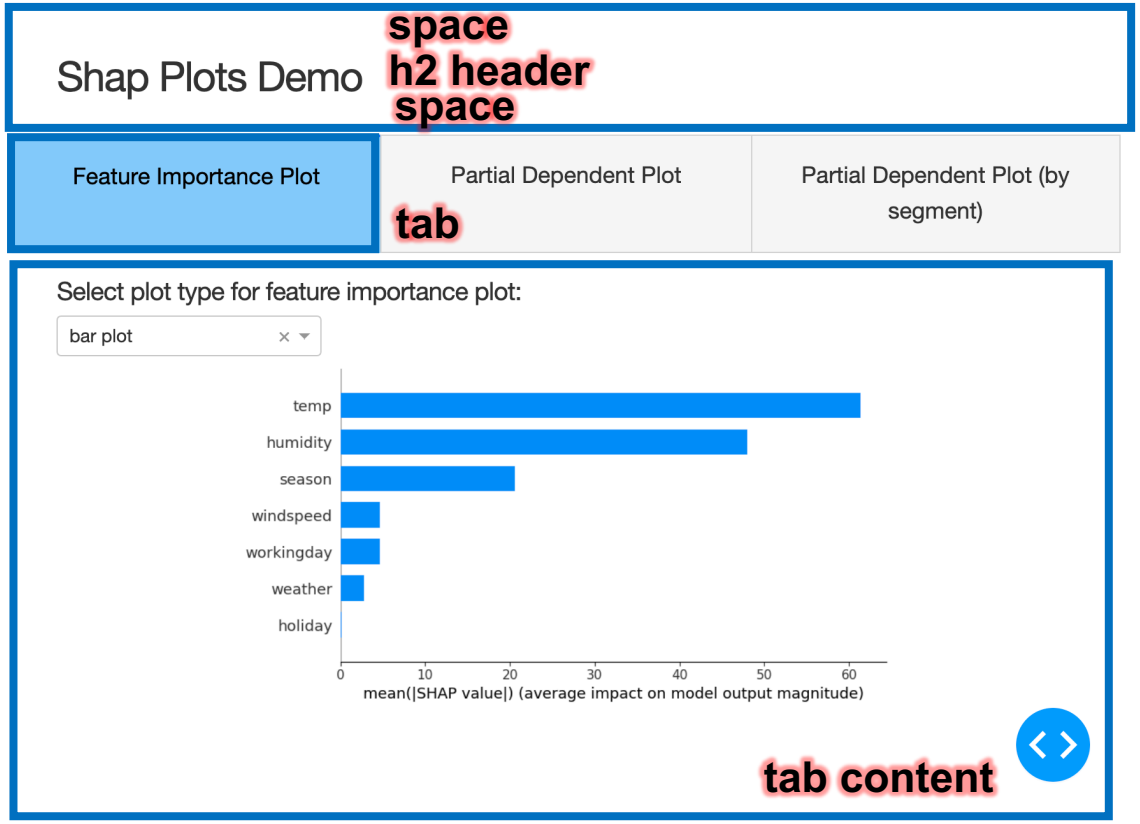
From website to html



html tutorial: https://www.w3schools.com/html/default.asp

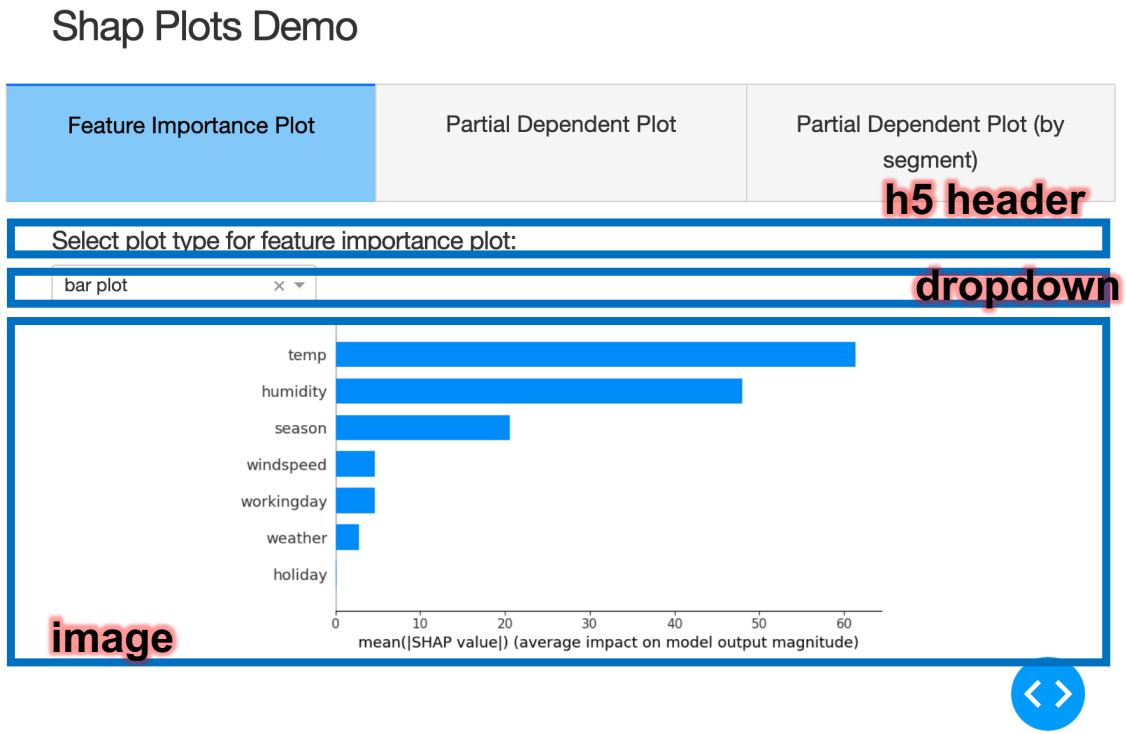
1. How does Dash work

From website to html



1. How does Dash work

From website to html




```
▼<div>
    <div style="width: 100%; height: 10px;"></div>
    <h2 style="margin-left: 40px;">Shap Plots Demo</h2>
    <div style="width: 100%; height: 10px;"></div>
  ▼<div id="tabs-parent" class="jsx-4017309047 tab-parent ">
    ▶<div id="tabs" class="jsx-4017309047 tab-container ">…</div>
    ▼<div class="jsx-4017309047 tab-content ">
        <div style="width: 100%; height: 10px;"></div> space
        <h5 style="margin-left: 40px;">Select plot type for feature
        importance plot:</h5> h5 header
      ▶<div style="margin-left: 40px;">…</div> == $0 dropdown
      ▶<div style="text-align: center;">…</div> image
      </div>
    </div>
  </div>
</div>
```

1. How does Dash work

From website to Dash to html


```
###############
# 0.Header of the whole website
html.Div(style={"width": "100%", "height": 10}),  # space
html.H2(children="Shap Plots Demo", style={"margin-left": left_margin}),
html.Div(style={"width": "100%", "height": 10}),  # space
###############
```

Dash: create html components and gcc components as python object

```
▼<div>
    <div style="width: 100%; height: 10px;"></div>
    <h2 style="margin-left: 40px;">Shap Plots Demo</h2>
    <div style="width: 100%; height: 10px;"></div>
```

space

h2 header

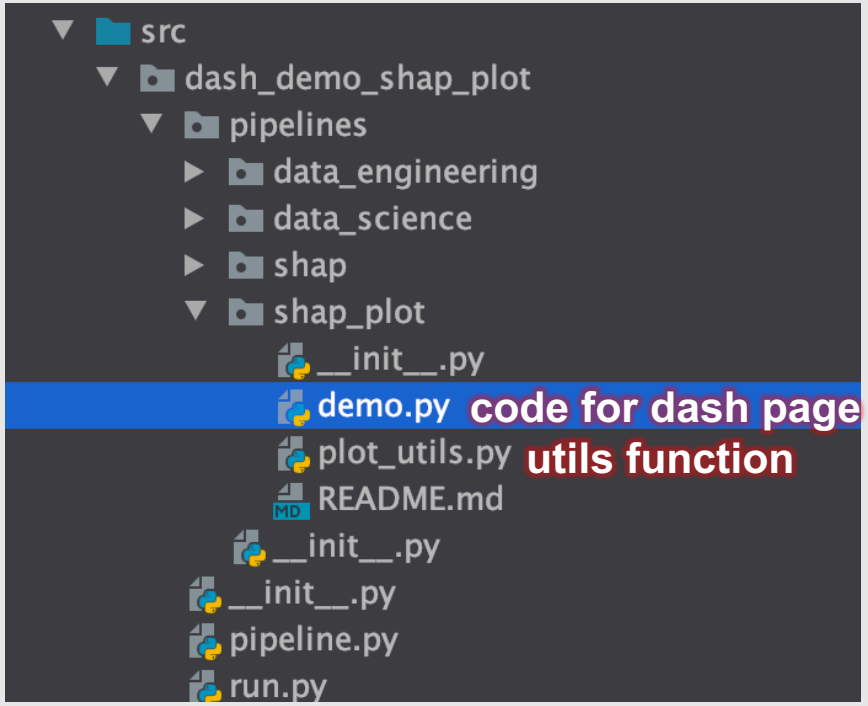
space

2. Minimal components to build a Dash page

- data
- style
- app layout
- callbacks
- run app

Before dash:

- raw data: download bike sharing data (https://www.kaggle.com/c/bike-sharing-demand)
- de pipeline: split into train and test
- ds pipeline: train a random forest model
- shap pipeline: calculate and save shap_values



2. Minimal components to build a Dash page

- **data**
- style
- app layout
- callbacks
- run app

Code inside src/dash_demo_shap_plot/pipslines/shap_plot/demo.py

```
############################## data ################################
# load data (hard coded)
train_x = pd.read_csv("data/05_model_input/train_x.csv")
shap_values = pd.read_csv("data/08_reporting/shap_values.csv")
```

*let me know if there's a smarter way to load from kedro catalog!

2. Minimal components to build a Dash page

- data
- **style**
- app layout
- callbacks
- run app

Code inside src/dash_demo_shap_plot/pipslines/shap_plot/demo.py

```
############################ styles ##############################
external_stylesheets = ["https://codepen.io/chriddyp/pen/bWLwgP.css"]
```

css template for the whole page, don't need to change

```
left_margin = 40
tab_default_style = {"fontSize": 20}
tab_selected_style = {"fontSize": 20, "backgroundColor": "#86caf9"}
dropdown_style = {"fontSize": 16, "width": "50%"}
summary_plot_style = {
    "height": "30%",
    "width": "60%",
}
pdp_plot_stype = {
    "height": "50%",
    "width": "50%",
}
```

(optional)

template css style for each html/gcc components,

attributes from html tag

```
# Set Matplotlib backend to a non-interactive
plt.switch_backend("Agg")
```

more attributes for html tag: https://www.w3schools.com/tags/tag_img.asp

2. Minimal components to build a Dash page

- data
- style
- app layout
- callbacks
- **run app**

Code inside src/dash_demo_shap_plot/pipslines/shap_plot/demo.py

```
if __name__ == "__main__":
    app.run_server(debug=True)
```

- debug=True: you can change code and refresh the website
- host="0.0.0.0": you can specify host
- to launch the dash app, run in terminal

>> *python src/dash_demo_shap_plot/pipslines/shap_plot/demo.py*

2. Minimal components to build a Dash page

- data
- style
- **app layout**
- callbacks
- run app

Code inside src/dash_demo_shap_plot/pipslines/shap_plot/demo.py

```
############################ layout ##############################
app = dash.Dash(__name__, external_stylesheets=external_stylesheets)
app.layout = html.Div(                          create app, don't need to change
    children=[
        ###############
        # 0.Header of the whole website
        html.Div(style={"width": "100%", "height": 10}),  # space
        html.H2(children="Shap Plots Demo", style={"margin-left": left_margin}),
        html.Div(style={"width": "100%", "height": 10}),  # space
        ###############
        dcc.Tabs(
            id="tabs",
            children=[
                ###############
                # 1. First tab
                dcc.Tab(
                    label="Feature Importance Plot",
                    children=[
                        html.Div(style={"width": "100%", "height": 10}),  # space
                        ###############
                        # 1.1 Dropdown menu
```

Will go through the details in part3.

2. Minimal components to build a Dash page

- data
- style
- app layout
- **callbacks**
- run app

Code inside src/dash_demo_shap_plot/pipslines/shap_plot/demo.py

```
######################## callbacks for Tab 1 ############################
# choose plot type from dropdown menu,
# generate summary plot
@app.callback(Output("summary_plot", "src"), [Input("plot_type", "value")])
def _generate_summary_plot(plot_type):
    feature_cols = shap_values.columns
    shap.summary_plot(
        shap_values.to_numpy(), train_x[feature_cols], plot_type=plot_type, show=False
    )
    plt.tight_layout()
    fig = plt.gcf()
    # Convert plt.figure object to base64 encoding
    # Don't need if use plotly library (plotly.express) to plot
    plotly_fig = _fig_to_uri(fig)
    return plotly_fig
```

(3) put output value into this html component with id = “summary_plot”

(1) take value from this html component with id = “plot_type”

(2) put the input into the function and returns the output

Will go through the details in part3.

3. Demo

code repo: https://github.com/ZecaiLiangQB/dash_tutorial
ready to use as a recipe for interactive SHAP plots.