Stony Brook University

DASH AND PLOTLY FOR INTERACTIVE PLOTTING

Mayank Tiwari
September 15, 2020

FAR BEYOND

DASH+PLOTLY

Dash https://plotly.com/dash/

- Dash is a productive Python framework for building web applications
- Written on top of Flask, Plotly.js, and React.js, Dash is ideal for building data visualization apps with highly custom user interfaces in pure Python
- Dash abstracts away all of the technologies and protocols that are required to build an interactive web-based application
- Knowledge of HTML & JS is not strictly necessary, but it can help as the function and call back names of Dash Core Components as it is same as HTML tags & JS functions

Plotly https://plotly.com/

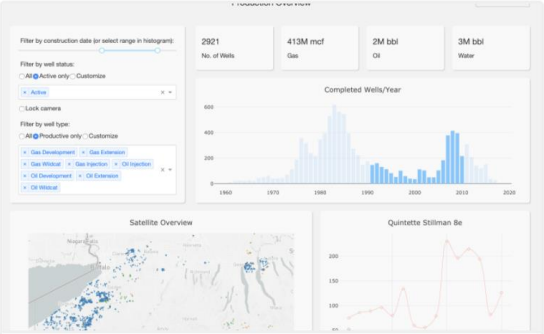
- Plotly is a free and open-source graphing library for Python
- Has many ways to customize graphs
- Works with or without Dash
  - Good & illustrated documentation: https://plot.ly/python/

Stony Brook University

All Apps (65)

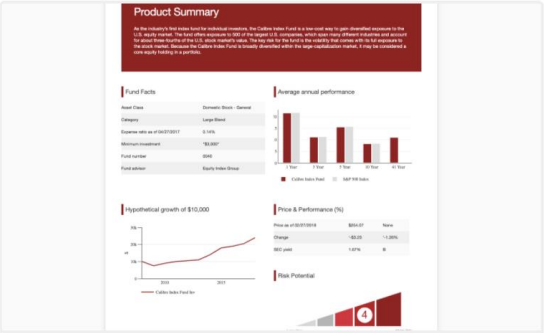
Dash App Gallery

Search applications...



New York Oil and Gas

Energy | Cross-filtering | Geospatial

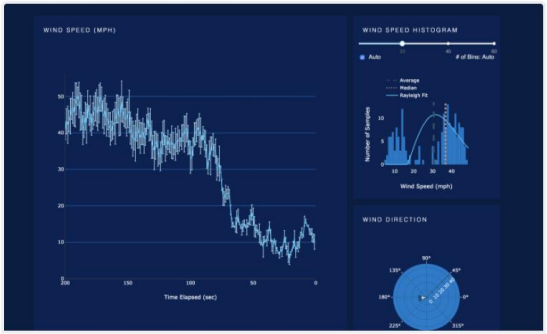


Financial Report

Finance

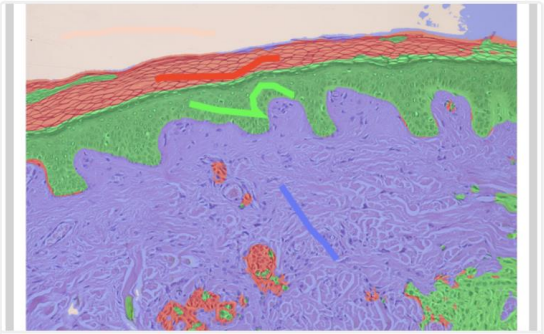
Manufacturing SPC Dashbo

Data Acquisition | Streaming



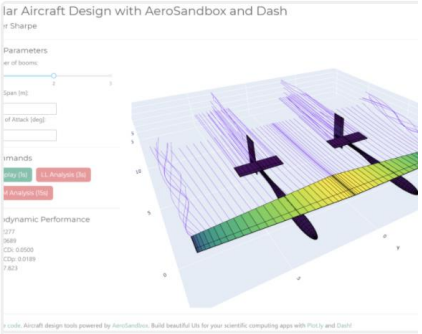
Wind Streaming

Streaming



Interactive Image Segmenta...

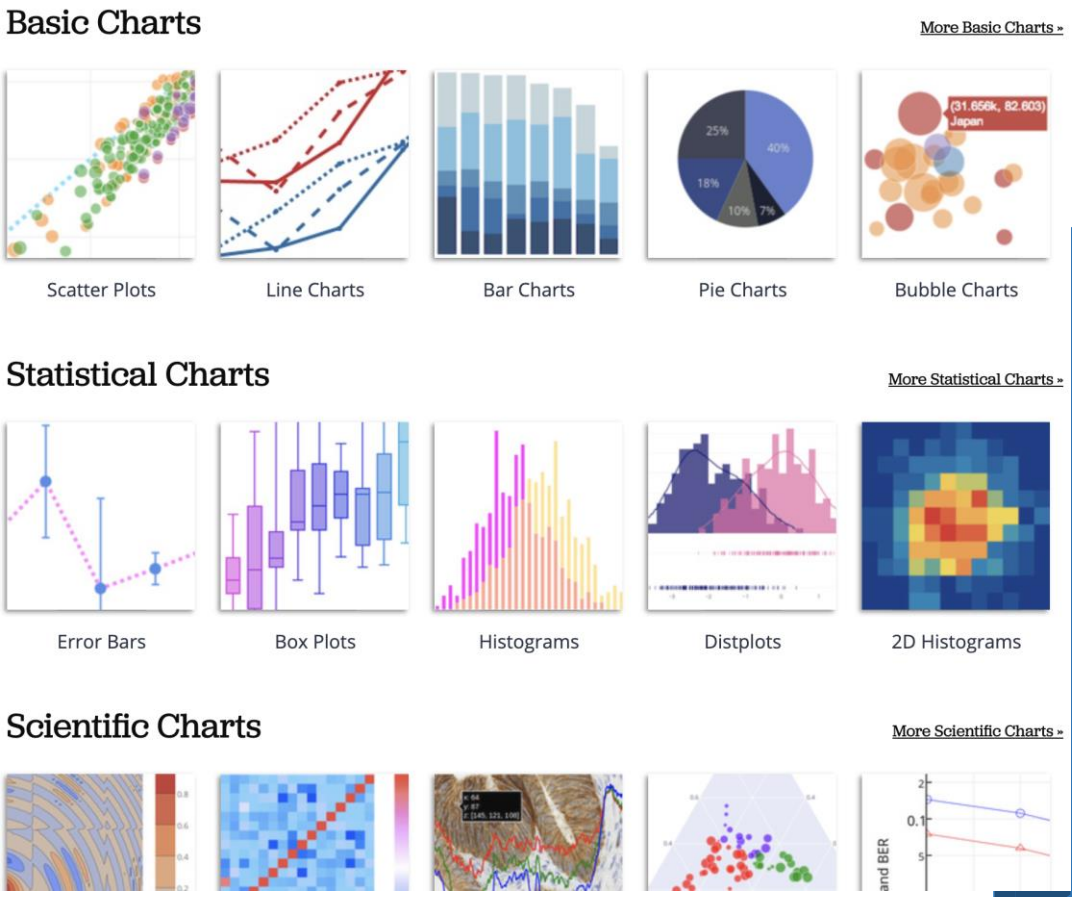
machine learning | image processing



AeroSandbox Demo

aerosandbox

FAR BEYOND



Plot.ly Chart Gallery

Website: https://plotly.com/python/

Dash Installation

pip install dash==1.16.0

A quick note on checking your versions and on upgrading. These docs are run using the versions listed above and these versions should be the latest versions available. To check which version that you have installed, you can run e.g.

```
>>> import dash_core_components
>>> print(dash_core_components.__version__)
```

Hello Dash!

```
import dash
import dash_html_components as html

app = dash.Dash()

app.layout = html.Div('Hello Dash!')

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

Dash – Main Components

- Layout (UI) - describes what the application looks like
  - Html components: https://dash.plotly.com/dash-html-components
  - Core components: https://dash.plotly.com/dash-core-components
- Callbacks - describes the interactivity of the application

Dash Layout

HTML ... in Python

```
import dash_html_components as html
app.layout = html.Div(children=[
    html.H1(
        'Hello Dash',
         style={'text-align': 'center'}),

    html.Div(
        id='my-div',
        children='Dash: A web app
    framework for Python. ',
         style={'textAlign': 'center'}),
    ]
)
```

Dash Layout

HTML ... in Python ... plus core components

```
import dash_core_components as dcc

component1 = dcc.Dropdown(value='MTL', options=[
    {'label': 'New York City', 'value': 'NYC'},
    {'label': 'Montréal', 'value': 'MTL'},
    {'label': 'San Francisco', 'value': 'SF'}])

component2 = dcc.Checklist(value=['MTL'], options=[
    {'label': 'New York City', 'value': 'NYC'},
    {'label': 'Montréal', 'value': 'MTL'},
    {'label': 'San Francisco', 'value': 'SF'}])

component3 = dcc.Slider(min=0, max=9, value=5)

component4 = dcc.Tabs(value='tab-2-example', children=[
    dcc.Tab(label='tab one', value='tab-1-example', children=[
        component1,
        component3
    ]),
    dcc.Tab(label='tab two', value='tab-2-example', children=component2)])

app.layout = html.Div(component4)
```

Dash Core Component - Graphs

Core component that accepts plotly.py go.Figure object!

```
import dash_core_components as dcc
import plotly.graph_objs as go
import dash

app = dash.Dash()
app.layout = html.Div(children=[
    html.H1('Hello Graph', style={'text-align': 'center'}),

    dcc.Graph(
        id='my-first-graph',
        figure=dict(data=[dict(x=[0, 1, 2], y=[3, 4, 2])])
    )
])

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

Scatter Plot Graph Example

```
import dash
import dash_core_components as dcc
import dash_html_components as html
import pandas as pd
import plotly.express as px

external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

app = dash.Dash(__name__, external_stylesheets=external_stylesheets)

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/gapminderDataFiveYear.csv')

fig = px.scatter(
    df, x="gdpPercap", y="lifeExp", size="pop",
    color="continent", hover_name="country", log_x=True, size_max=60)

app.layout = html.Div([
    dcc.Graph(
        id='life-exp-vs-gdp',
        figure=fig
    )
])

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

Callbacks

Callbacks are Python functions that are **automatically called** by Dash whenever an input component's **property changes**.

Reference: https://dash.plotly.com/basic-callbacks

Dash Callbacks

You can get callbacks from

- Button Clicks, Text(Div/P) clicks
- Dropdown list value entered/changed
- Graph Hover/Click on Value
- Period timers, URL address change,…

From Dash Callbacks, you can

- Update input values
- Generate new HTML elements
- Update the CSS style of the layout HTML elements
- Generate any kind of plot.ly graph

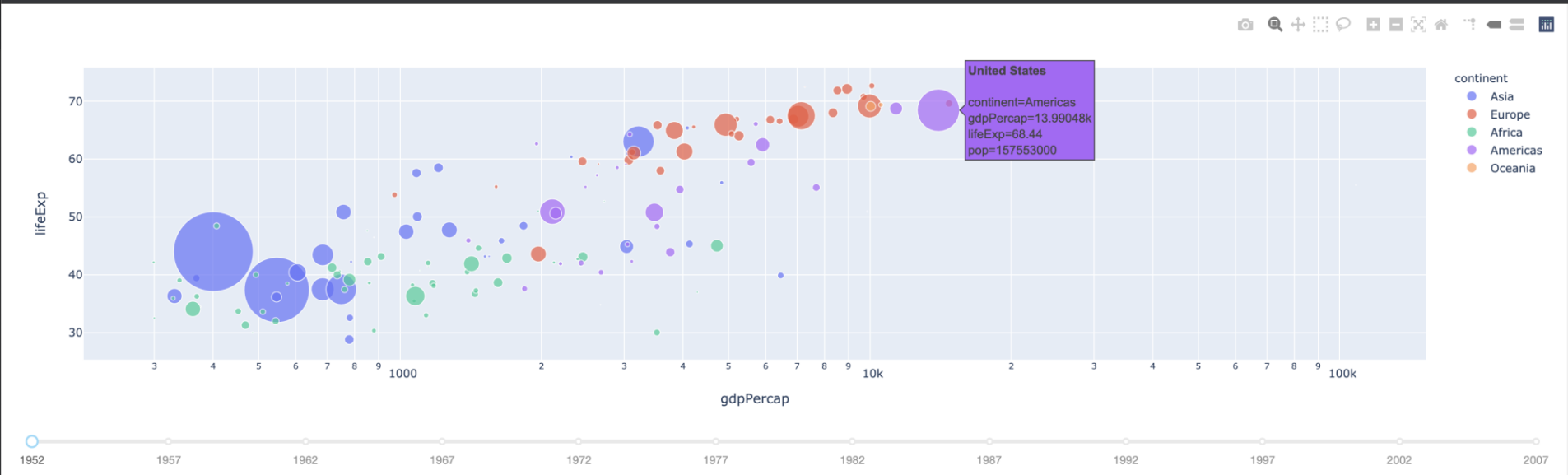
Callbacks - Example

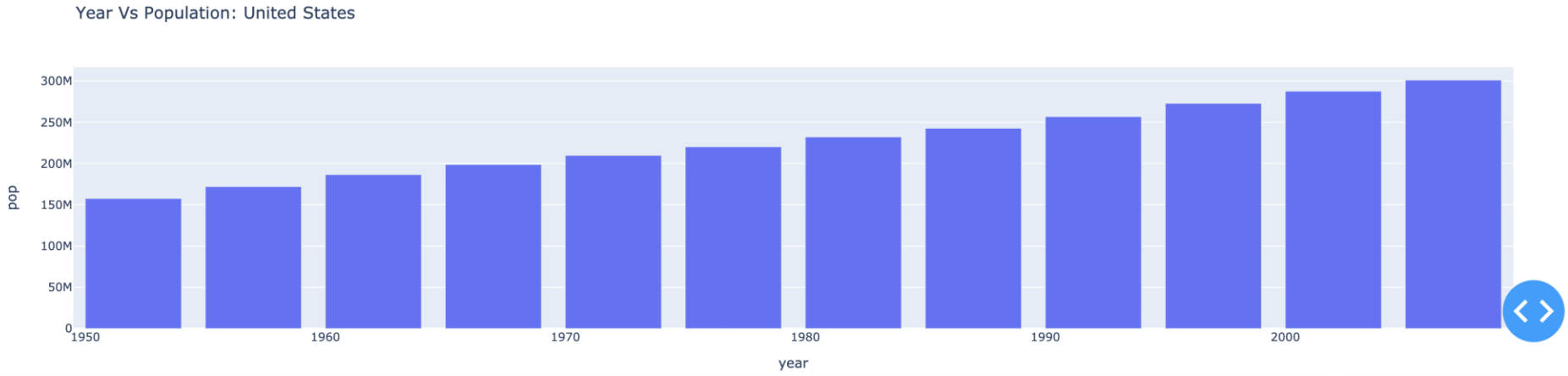
```
from dash.dependencies import Input, Output

df =
pd.read_csv('https://raw.githubusercontent.com/plotly/datasets/master/gapminderDataFiveYear.csv
')

@app.callback(
    Output('life-exp-vs-gdp', 'figure'),
    Input('year-slider', 'value')
)
def update_figure(selected_year):
    filterDf = df[df.year == selected_year]
    fig = px.scatter(filterDf, x="gdpPercap", y="lifeExp", size="pop", color="continent",
hover_name="country", log_x=True, size_max=60)
    fig.update_layout(transition_duration=500)
    return fig

app.layout = html.Div([
    dcc.Graph(id='life-exp-vs-gdp'),
    dcc.Slider(
        id='year-slider', min=df['year'].min(), value=df['year'].min(),
        max=df['year'].max(), marks={str(year): str(year) for year in df['year'].unique()}, step=None
    )
])
```

Callback - Linking

```
@app.callback(
    Output('year-pop', 'figure'),
    [dash.dependencies.Input('life-exp-vs-gdp', 'hoverData')]
)
def update_output_div(hoverData):
    if not hoverData:
        country = ''
    else:
        country = hoverData['points'][0]['hovertext']
    filterDf = df[df.country == country]
    fig = px.bar(filterDf, x='year', y='pop', title='Year Vs Population: {}'.format(country))
    # return 'Output: {}'.format(hoverData['points'][0]['hovertext'])
    return fig

app.layout = html.Div([
    dcc.Graph(id='life-exp-vs-gdp'),
    dcc.Slider(
        id='year-slider', min=df['year'].min(), value=df['year'].min(),
        max=df['year'].max(), marks={str(year): str(year) for year in df['year'].unique()}, step=None
    ),
    dcc.Graph(id='year-pop'),
])
```


continent
Asia
Europe
Africa
Americas
Oceania
lifeExp
70
60
50
40
30
China
continent=Asia
gdpPercap=400.4486
lifeExp=44
pop=556263528
1000
10k
100k
gdpPercap
1952
1957
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007
Year Vs Population: China
pop
1.2B
1B
0.8B
0.6B
0.4B
0.2B
0
1950
1960
1970
1980
1990
2000
year


United States
continent=Americas
gdpPercap=13.99048k
lifeExp=68.44
pop=157553000
continent
Asia
Europe
Africa
Americas
Oceania
lifeExp
gdpPercap
1000
10k
100k
1952
1957
1962
1967
1972
1977
1982
1987
1992
1997
2002
2007


Year Vs Population: United States
pop
year
300M
250M
200M
150M
100M
50M
0
1950
1960
1970
1980
1990
2000

References

- https://dash-gallery.plotly.host/Portal/
- https://dash.plotly.com/
- https://plotly.com/python/
- https://github.com/plotly/dash