dashboard_webapps

June 19, 2020

# Extending Plotting

It's time to extend your plotting skills. Over the past two lessons, you've learned how to create a range of interactive plots using hvPlot and Plotly Express; however, you haven't had one centralized location to embed these plots. Now you do! Integrate Plotly map visualizations with hvPlot scatter plots to create a Population and Crimes dashboard

[1]:
```
import plotly.express as px
import panel as pn
import pandas as pd
import os
from pathlib import Path
from dotenv import load_dotenv
```

### Use extension function to specify plugin

[2]:
```
# Set up Panel Plotly extension
pn.extension('plotly')
```

0.0.1 Import hvplot.pandas after pn.extension

[3]:
```
# Import hvplot.pandas after pn.extension
# This avoids plotly initialization failure
import hvplot.pandas
```

### Set up Mapbox token and prepare data

[4]:
```
# Read the Mapbox API key
load_dotenv()
map_box_api = os.getenv("mapbox")

# Set token using Plotly Express set function
px.set_mapbox_access_token(map_box_api)

# Read in data
city_pop = pd.read_csv(Path("../Resources/population_counts.csv")).
 ↪drop_duplicates()
crime_rates = pd.read_csv(Path("../Resources/crime_rates.csv")).
 ↪drop_duplicates()
```

```
pop_with_index = city_pop.set_index("city")
crime_with_index = crime_rates.set_index("city")
population_crime = (
    pd.concat([pop_with_index, crime_with_index], axis=1, sort=True)
    .dropna()
    .reset_index()
)
```

Create plots

[5]:
```
# Create plots
def get_population_plot():
    population_plot = px.scatter_mapbox(
        population_crime,
        lat="latitude",
        lon="longitude",
        size="pop_2015",
        color="index",
        color_continuous_scale=px.colors.cyclical.IceFire,
        title="City Population",
        zoom=3,
        width=1000,
    )
    return population_plot


def get_crime_plot():
    crime_plot = px.scatter_mapbox(
        population_crime,
        lat="latitude",
        lon="longitude",
        size="violent_crime",
        color="index",
        color_continuous_scale=px.colors.cyclical.IceFire,
        title="City Crime",
        zoom=3,
        width=1000,
    )
    return crime_plot


def get_population_violence():
    population_violence = population_crime.hvplot.scatter(
        x="pop_2015",
        y="violent_crime",
        title="Violent Crime by Population Correlation",
        width=1000,
    ).opts(yformatter="%.0f")
    return population_violence


def get_violent_murder():
    violent_murder = population_crime.hvplot.scatter(
        x="violent_crime",
        y="murder",
        title="Correlation Between Number of Violent Crimes and Murder",
        width=1000,
    ).opts(yformatter="%.0f")
    return violent_murder
```

### Create Panel columns and tabs

[6]:
```
# Create panels to structure the layout of the dashboard
geo_column = pn.Column(
    "## Population and Crime Geo Plots", get_population_plot(), get_crime_plot()
)

scatter_column = pn.Column(
    "## Correlation of Population and Crime Plots",
    get_population_violence(),
    get_violent_murder(),
)

# Create tabs
crime_pop_dashboard = pn.Tabs(
    ("Geospatial", geo_column), ("Correlations", scatter_column)
)
```

### Execute the servable function

[7]:
```
# Execute Panel dashboard using servable function
crime_pop_dashboard.servable()
```

[7]:
```
Tabs
    [0] Column
        [0] Markdown(str)
        [1] Plotly(Figure)
        [2] Plotly(Figure)
    [1] Column
        [0] Markdown(str)
        [1] HoloViews(Scatter)
        [2] HoloViews(Scatter)
```