dashboard

February 13, 2022

1 Toronto Dwellings Analysis Dashboard

In this notebook, you will compile the visualizations from the previous analysis into functions to create a Panel dashboard.

[1]:

```python
# imports
import panel as pn
pn.extension('plotly')
import plotly.express as px
import pandas as pd
import hvplot.pandas
import matplotlib.pyplot as plt
import os
from pathlib import Path
from dotenv import load_dotenv
```

```
Bad key text.latex.preview in file
C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-packages\matplotlib\mpl-
data\stylelib\_classic_test.mplstyle, line 123 ('text.latex.preview : False')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key mathtext.fallback_to_cm in file
C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-packages\matplotlib\mpl-
data\stylelib\_classic_test.mplstyle, line 155 ('mathtext.fallback_to_cm : True
# When True, use symbols from the Computer Modern')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key savefig.jpeg_quality in file
C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-packages\matplotlib\mpl-
data\stylelib\_classic_test.mplstyle, line 418 ('savefig.jpeg_quality: 95
# when a jpeg is saved, the default quality parameter.')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
```

```
or from the matplotlib source distribution

Bad key savefig.frameon in file C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-
packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle, line 421
('savefig.frameon : True')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key verbose.level in file C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-
packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle, line 472
('verbose.level  : silent      # one of silent, helpful, debug, debug-annoying')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key verbose.fileo in file C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-
packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle, line 473
('verbose.fileo  : sys.stdout  # a log filename, sys.stdout or sys.stderr')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key keymap.all_axes in file C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-
packages\matplotlib\mpl-data\stylelib\_classic_test.mplstyle, line 490
('keymap.all_axes : a                 # enable all axes')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key animation.avconv_path in file
C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-packages\matplotlib\mpl-
data\stylelib\_classic_test.mplstyle, line 501 ('animation.avconv_path: avconv
# Path to avconv binary. Without full path')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution

Bad key animation.avconv_args in file
C:\Users\antho\anaconda3\envs\pyvizenv\lib\site-packages\matplotlib\mpl-
data\stylelib\_classic_test.mplstyle, line 503 ('animation.avconv_args:
# Additional arguments to pass to avconv')
You probably need to get an updated matplotlibrc file from
https://github.com/matplotlib/matplotlib/blob/v3.5.0/matplotlibrc.template
or from the matplotlib source distribution
```

[2]:

```
# Initialize the Panel Extensions (for Plotly)
import panel as pn
pn.extension("plotly")
```

[3]:

```
# Read the Mapbox API key
load_dotenv()
map_box_api = os.getenv("mapbox")
px.set_mapbox_access_token(map_box_api)
```

2 Import Data

[4]:

```
# Import the CSVs to Pandas DataFrames
file_path = Path("Data/toronto_neighbourhoods_census_data.csv")
to_data = pd.read_csv(file_path, index_col="year")

file_path = Path("Data/toronto_neighbourhoods_coordinates.csv")
df_neighbourhood_locations = pd.read_csv(file_path,index_col='neighbourhood')

file_path = Path("sum_dwelling.csv")
sum_dwelling = pd.read_csv(file_path, index_col="year")
```

2.1 Panel Visualizations

In this section, you will copy the code for each plot type from your analysis notebook and place it into separate functions that Panel can use to create panes for the dashboard.

These functions will convert the plot object to a Panel pane.

Be sure to include any DataFrame transformation/manipulation code required along with the plotting code.

Return a Panel pane object from each function that can be used to build the dashboard.

Note: Remove any .show() lines from the code. We want to return the plots instead of showing them. The Panel dashboard will then display the plots.

2.1.1 Global available data

[5]:

```
# Getting the data from the top 10 expensive neighbourhoods
# YOUR CODE HERE!

richest_neighbourhoods = to_data.groupby('neighbourhood').mean()
richest_neighbourhoods_sorted = richest_neighbourhoods.
 ↪sort_values('average_house_value', ascending=False)
richest_neighbourhoods_sorted_top_10 = richest_neighbourhoods_sorted[0:10]
richest_neighbourhoods_sorted_top_10_housevalue =␣
 ↪richest_neighbourhoods_sorted_top_10['average_house_value']
```

[6]:

```
# Join the average values with the neighbourhood locations
# YOUR CODE HERE!

df_mean_neighbourhood_locations = pd.concat([richest_neighbourhoods,␣
 ↪df_neighbourhood_locations], axis=1)
```

[7]:

```
# Calculate the mean number of dwelling types units per year
# YOUR CODE HERE!

to_data_reset_plot = to_data.groupby(['year', 'neighbourhood']).mean()
```

[8]:

```
# Calculate the average monthly shelter costs for owned and rented dwellings
# YOUR CODE HERE!

costs_mean = to_data_reset_plot[
    ['shelter_costs_owned','shelter_costs_rented']
]
```

[9]:

```
data_types = sum_dwelling[
    ['single_detached_house',␣
 ↪'apartment_five_storeys_plus','movable_dwelling','semi_detached_house','row_house',
    'duplex', 'apartment_five_storeys_less', 'other_house']
]
```

[10]:

```
### Data organsining for plots
df_mean = to_data[
    ['neighbourhood', 'average_house_value']
]
```

[11]:

```
## Data for Dwelling Type Units per year
row = data_types.iloc[0]
row1 = data_types.iloc[1]
row2 = data_types.iloc[2]
row3 = data_types.iloc[3]
```

[12]:

```
### Data for Sunburst Chart - Top 10 Expensive Neighbourhoods by Year
df_mean_reset = to_data.reset_index()
df_mean_reset_2001 = df_mean_reset.loc[df_mean_reset['year'] == 2001].
 ↪sort_values('average_house_value', ascending=False)[0:10]
df_mean_reset_2006 = df_mean_reset.loc[df_mean_reset['year'] == 2006].
 ↪sort_values('average_house_value', ascending=False)[0:10]
df_mean_reset_2011 = df_mean_reset.loc[df_mean_reset['year'] == 2011].
 ↪sort_values('average_house_value', ascending=False)[0:10]
df_mean_reset_2016 = df_mean_reset.loc[df_mean_reset['year'] == 2016].
 ↪sort_values('average_house_value', ascending=False)[0:10]
```

```
top10_richest_neighbourhoods_byyear = pd.concat([df_mean_reset_2001,␣
↪df_mean_reset_2006, df_mean_reset_2011,df_mean_reset_2016])
```

[13]:

```
## Bar Chart with Facet Row Data Reorder
avg_house_value_year = to_data[
    ['neighbourhood','average_house_value']
].reset_index()
```

2.2 Panel Visualization Functions

[14]:

```
""" Year 2001 Dwelling Types """

def dwelling_types_per_year(
    row = row.hvplot(kind='bar',title='Year 2001 Dwelling Types',␣
↪color='m',rot=20, yformatter='%.0f')
):
    return row
```

[15]:

```
""" Year 2006 Dwelling Types """

def dwelling_types_per_year_1(
    row1 = row1.hvplot(kind='bar',title='Year 2006 Dwelling␣
↪Types',figsize=(12,6), color='c',rot=20,yformatter='%.0f')
):
    return row1
```

[16]:

```
""" Year 2016 Dwelling Types """

def dwelling_types_per_year_2(
    row2 = row2.hvplot(kind='bar',title='Year 2011 Dwelling␣
↪Types',figsize=(12,6), color='y',rot=20,yformatter='%.0f')
):
    return row2
```

[17]:

```
""" Year 2016 Dwelling Types """

def dwelling_types_per_year_3(
    row3 = row3.hvplot(kind='bar',title='Year 2016 Dwelling␣
↪Types',figsize=(12,6), color='b',rot=20,yformatter='%.0f')
):
    return row3
```

[18]:

```
row_of_bar = pn.Row(dwelling_types_per_year,dwelling_types_per_year_1)
row_of_bar2 = pn.Row(dwelling_types_per_year_2,dwelling_types_per_year_3)
```

[19]:

```
    """Neighbourhood Map"""
```

```
# Define Panel visualization functions
def neighbourhood_map(

    fig = px.scatter_mapbox(df_mean_neighbourhood_locations, lat="lat",␣
↪lon="lon",   color="average_house_value",
                  color_continuous_scale=px.colors.cyclical.IceFire,␣
↪size='average_house_value', zoom=10,mapbox_style='open-street-map',␣
↪height=600, width=1200, title='Average House Value Scatterplot',
                           ␣
↪hover_data=['single_detached_house','apartment_five_storeys_plus','movable_dwelling','semi_
↪'row_house', 'duplex', 'apartment_five_storeys_less',
                                                                     ␣
↪'other_house', 'shelter_costs_owned','shelter_costs_rented'])

):
        return fig
```

[20]:

```
""" Top 10 Richest Neighbourhoods in Toronto"""

def top_10_richest_neighbourhoods(

top_10_richest = richest_neighbourhoods_sorted_top_10_housevalue.hvplot.
↪bar(width=750,height=600, rot=60,
                                                           title='Top 10 Most␣
↪Expensive Neighbourhoods In Toronto',
                                                           ylabel='Price',
                                                          xlabel='Price',␣
↪fontscale=1.2,yformatter="%.0f")
):
    return top_10_richest

pn.interact(top_10_richest_neighbourhoods)
```

[20]:

```
Column
    [0] Column()
    [1] Row
        [0] HoloViews(Bars, name='interactive01958')
```

[21]:

```
"""Average house values by neighbourhood."""

    # YOUR CODE HERE!

def barchar_house_value(
    df_mean_bar = df_mean.hvplot(x='year', y='average_house_value', width=550,␣
↪height=500, color='chartreuse',
               line_width= 5, groupby='neighbourhood',
               widget_location='top_left', yformatter='%.0f'
```

```
              ,xlabel='Year',ylabel='Average House Value Price')
):
    return df_mean_bar
```

[22]:

```
    """Average Shelter Owned and Rented Costs for all Toronto's neighbourhoods␣
 ↪per year."""

    # YOUR CODE HERE!

# Create two line charts, one to plot the monthly shelter costs for owned␣
 ↪dwelleing and
## other for rented dwellings per year

def average_shelter_value(
    costs_mean_plot = costs_mean.hvplot(x='year',␣
 ↪y=['shelter_costs_owned','shelter_costs_rented'], width=800, height=500,
                            line_width= 5,␣
 ↪groupby='neighbourhood',widget_location='top_left', yformatter='%.
 ↪0f',xlabel='Year',ylabel='Average House Value Price',
                                       fontscale=1.5)

):
        return costs_mean_plot
```

[23]:

```
row_of_bar_housevalue = pn.Row(average_shelter_value,barchar_house_value)
```

[24]:

```
 """ Number of dwelling types per year"""

def number_dwelling_types(

    number_dwelling_types_plot = to_data_reset_plot.hvplot.bar(x='year',␣
 ↪y=['single_detached_house','apartment_five_storeys_plus','movable_dwelling',
                                              'semi_detached_house',␣
 ↪'row_house', 'duplex', 'apartment_five_storeys_less',
                                              'other_house'],
                                 stacked=False, width=900,height=600,␣
 ↪rot=90,groupby='neighbourhood', widget_location='top_left',
                                 yformatter='%.0f"', xlabel= 'Year', ylabel =␣
 ↪'Dwelling Type Units', colormap='rainbow', fontscale=1.25)

):
    return number_dwelling_types_plot
```

[25]:

```
""" Bar Chart Facet Row"""

def barchart_facet(
    fig = px.bar(avg_house_value_year, x='neighbourhood',␣
 ↪y='average_house_value', color='average_house_value', facet_row='year',␣
 ↪height=800, width=1000)

):
    return fig
```

[26]:

```
""" Sunburst Chart for average House Value"""

def sunburst_house_chart(

    sunburst_chart = px.sunburst(top10_richest_neighbourhoods_byyear,␣
 ↪path=['year','neighbourhood'],values='average_house_value',
                         color='average_house_value',␣
 ↪hover_data=['single_detached_house','apartment_five_storeys_plus','movable_dwelling','semi_
 ↪'row_house', 'duplex', 'apartment_five_storeys_less',
                                                                                   ␣
 ↪'other_house', 'shelter_costs_owned','shelter_costs_rented'],
                             width=900,height=900)
):
    return sunburst_chart
```

2.3 Panel Dashboard

In this section, you will combine all of the plots into a single dashboard view using Panel. Be creative with your dashboard design!

[27]:

```
# Create a Title for the Dashboard
# Define a welcome text
# Create a tab layout for the dashboard
# Create the main dashboard
# YOUR CODE HERE!

plot_as_column = pn.Column(
    ("# Average House Dashboard"),
    ("Welcome to an interaction Real Estate Analysis of Toronto Dashboard."),
    ("This dashborad includes an analysis of the average house value per␣
 ↪neighouhood, sheltter costs for owned and rented properties per␣
 ↪neighbourhood and the types of dwelling units."),
    ("To see the data just click through the tabs on the top of the dashboard␣
 ↪and it will direct you to the relevant information. Enjoy!")
    , neighbourhood_map
)
```

```
plot_as_column_row = pn.Column(
    ("# Dwelling Types Per Year"),
    row_of_bar, row_of_bar2)

plot_as_column_2 = pn.Column(
    ("# Shelter Costs vs. House Value"),
    row_of_bar_housevalue
)

plot_as_column_3 = pn.Column(
    ("# Dwelling Type  Units"),
    ("This graph includes all dwelling type units per neighbourhood for the years 2001, 2006, 2011 and 2016."),
    number_dwelling_types, barchart_facet
)

plot_as_column_4 = pn.Column(
    ("# Average House Value"),
 barchart_facet, sunburst_house_chart
)



plot_as_column_5 = pn.Column(
    "# Top 10 Richest Neighbourhoods", "Click year to enter that year data only. Hover over neighbourhoods to find all combined data."
    ,sunburst_house_chart,top_10_richest_neighbourhoods
)



tabs = pn.Tabs(
    ## hello
    ("Welcome", plot_as_column),
    ("Dwelling Types Per Year", plot_as_column_row),
    ("Shelter Costs vs. House Value", plot_as_column_2),
    ("Neighbourhood Analysis", plot_as_column_3),
    ("Top 10 Richest Neighbourhoods",plot_as_column_5)
)
```

2.4 Serve the Panel Dashboard

[28]:

```
tabs.servable()
```

[28]:

```
Tabs
    [0] Column
        [0] Markdown(str)
        [1] Markdown(str)
        [2] Markdown(str)
        [3] Markdown(str)
        [4] Column
            [0] Column()
            [1] Row
                [0] Plotly(Figure, name='interactive02164')
    [1] Column
        [0] Markdown(str)
        [1] Row
            [0] Column
                [0] Column()
                [1] Row
                    [0] HoloViews(Bars, name='interactive01843')
            [1] Column
                [0] Column()
                [1] Row
                    [0] HoloViews(Bars, name='interactive01853')
        [2] Row
            [0] Column
                [0] Column()
                [1] Row
                    [0] HoloViews(Bars, name='interactive01864')
            [1] Column
                [0] Column()
                [1] Row
                    [0] HoloViews(Bars, name='interactive01874')
    [2] Column
        [0] Markdown(str)
        [1] Row
            [0] Column
                [0] Column()
                [1] Row
                    [0] Column
                        [0] Row
                            [0] WidgetBox
                                [0] Select(margin=(20, 20, 20, 20),
name='neighbourhood', options=['Agincourt North', …], value='Agincourt North',
width=250)
                            [1] HSpacer()
                        [1] HoloViews(DynamicMap, widget_location='top_left')
            [1] Column
                [0] Column()
                [1] Row
                    [0] Column
                        [0] Row
                            [0] WidgetBox
                                [0] Select(margin=(20, 20, 20, 20),
name='neighbourhood', options=['Agincourt North', …], value='Agincourt North',
width=250)
                            [1] HSpacer()
                        [1] HoloViews(DynamicMap, widget_location='top_left')
    [3] Column
        [0] Markdown(str)
        [1] Markdown(str)
        [2] Column
            [0] Column()
            [1] Row
                [0] Column
                    [0] Row
                        [0] WidgetBox
                            [0] Select(margin=(20, 20, 20, 20),
name='neighbourhood', options=['Agincourt North', …], value='Agincourt North',
width=250)
                        [1] HSpacer()
                    [1] HoloViews(DynamicMap, widget_location='top_left')
        [3] Column
            [0] Column()
            [1] Row
                [0] Plotly(Figure, name='interactive02187')
    [4] Column
        [0] Markdown(str)
        [1] Markdown(str)
        [2] Column
            [0] Column()
            [1] Row
                [0] Plotly(Figure, name='interactive02216')
        [3] Column
            [0] Column()
            [1] Row
                [0] HoloViews(Bars, name='interactive02223')
```

3 Debugging

Note: Some of the Plotly express plots may not render in the notebook through the panel functions.

However, you can test each plot by uncommenting the following code

[29]:

```python
# neighbourhood_map().show()
```

[30]:

```python
# create_bar_chart(data, title, xlabel, ylabel, color)

# # Bar chart for 2001
# create_bar_chart(df_dwelling_units.loc[2001], "Dwelling Types in Toronto in 2001", "2001", "Dwelling Type Units", "red")
# # Bar chart for 2006
# create_bar_chart(df_dwelling_units.loc[2006], "Dwelling Types in Toronto in␣
 ↪2006", "2006", "Dwelling Type Units", "blue")

# # Bar chart for 2011
# create_bar_chart(df_dwelling_units.loc[2011], "Dwelling Types in Toronto in␣
 ↪2011", "2011", "Dwelling Type Units", "orange")

# # Bar chart for 2016
# create_bar_chart(df_dwelling_units.loc[2016], "Dwelling Types in Toronto in␣
 ↪2016", "2016", "Dwelling Type Units", "magenta")
```

[31]:

```
# create_line_chart(data, title, xlabel, ylabel, color)

# # Line chart for owned dwellings
# create_line_chart(df_avg_costs["shelter_costs_owned"], "Average Monthly␣
 ↪Shelter Cost for Owned Dwellings in Toronto", "Year", "Avg Monthly Shelter␣
 ↪Costs", "blue")

# # Line chart for rented dwellings
# create_line_chart(df_avg_costs["shelter_costs_rented"], "Average Monthly␣
 ↪Shelter Cost for Rented Dwellings in Toronto", "Year", "Avg Monthly Shelter␣
 ↪Costs", "orange")
```

[32]:

```
# average_house_value()
```

[33]:

```
# average_value_by_neighbourhood()
```

[34]:

```
# number_dwelling_types()
```

[35]:

```
# average_house_value_snapshot()
```

[36]:

```
# top_most_expensive_neighbourhoods()
```

[37]:

```
# sunburts_cost_analysis()
```