In [345…

```
import pandas as pd
import numpy as np
import panel as pn
pn.extension('tabulator')
import pycountry_convert as pc
import hvplot.pandas
import holoviews as hv
```

Install some libraries

pip install country_converter -upgrade

pip install dataprep

pip install pycountry-convert

In [224…

```
 # importing the data
 df = pd.read_csv("GCB2022v27_MtCO2_flat.csv")
df
```

Out[224]:

	Country	ISO 3166-1 alpha-3	Year	Total	Coal	Oil	Gas	Cement	Flaring	Other	Per Capita
0	Afghanistan	AFG	1750	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN
1	Afghanistan	AFG	1751	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	Afghanistan	AFG	1752	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	Afghanistan	AFG	1753	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	Afghanistan	AFG	1754	0.000000	NaN	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...	...	...
63099	Global	WLD	2017	36096.739276	14506.973805	12242.627935	7144.928128	1507.923185	391.992176	302.294047	4.749682
63100	Global	WLD	2018	36826.506600	14746.830688	12266.016285	7529.846784	1569.218392	412.115746	302.478706	4.792753
63101	Global	WLD	2019	37082.558969	14725.978025	12345.653374	7647.528220	1617.506786	439.253991	306.638573	4.775633
63102	Global	WLD	2020	35264.085734	14174.564010	11191.808551	7556.290283	1637.537532	407.583673	296.301685	4.497423
63103	Global	WLD	2021	37123.850352	14979.598083	11837.159116	7921.829472	1672.592372	416.525563	296.145746	4.693699

63104 rows × 11 columns

DATA CLEANING

In [225…

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 63104 entries, 0 to 63103
Data columns (total 11 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   Country            63104 non-null  object 
 1   ISO 3166-1 alpha-3 61472 non-null  object 
 2   Year               63104 non-null  int64  
 3   Total              62904 non-null  float64
 4   Coal               21744 non-null  float64
 5   Oil                21717 non-null  float64
 6   Gas                21618 non-null  float64
 7   Cement             20814 non-null  float64
 8   Flaring            21550 non-null  float64
 9   Other              1620 non-null   float64
 10  Per Capita         18974 non-null  float64
dtypes: float64(8), int64(1), object(2)
memory usage: 5.3+ MB
```

In [226…

```
df.describe()
```

Out[226]:

	Year	Total	Coal	Oil	Gas	Cement	Flaring	Other	Per Capita
count	63104.000000	62904.000000	21744.000000	21717.000000	21618.000000	20814.000000	21550.000000	1620.000000	18974.000000
mean	1885.500000	55.224788	73.968916	55.760624	23.504285	4.330443	1.712695	10.951389	4.413363
std	78.519728	824.845435	598.986992	519.034563	247.674772	50.305770	16.727067	39.034073	17.432815
min	1750.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	1817.750000	0.000000	0.000000	0.091600	0.000000	0.000000	0.000000	0.520885	0.197866
50%	1885.500000	0.000000	0.271852	1.044240	0.000000	0.022756	0.000000	1.255329	1.303949
75%	1953.250000	0.549342	6.736411	8.339752	0.581628	0.568502	0.000000	4.385471	5.077994
max	2021.000000	37123.850352	15051.512770	12345.653374	7921.829472	1672.592372	439.253991	306.638573	834.192642

In [227…

```
# Checks total duplicate rows
duplicate = df.duplicated(keep=False).sum()
duplicate
```

Out[227]: 0

In [228…

```
## check how many columns contains null values
df.isnull().sum().sort_values()
```

Out[228]:

```
Country                   0
Year                      0
Total                   200
ISO 3166-1 alpha-3     1632
Coal                  41360
Oil                   41387
Gas                   41486
Flaring               41554
Cement                42290
Per Capita            44130
Other                 61484
dtype: int64
```

In [229…

```
# Asumming that if a country does not have a
df.fillna(0, inplace = True)
```

In [ ]:
```
# some countries places in the dataset are not countires, so i dropped them.
indexAge = df[ (df['Country'] == "St. Kitts-Nevis-Anguilla")].index
df.drop(indexAge , inplace=True)
df.head(15)
```

In [ ]:
```
df["Country"].unique()
```

In [371…
```
df.to_csv('file1.csv')
```

In [387…
```
df.rename(columns={"Total": "Total_CO2"}, inplace=True)
```

In [366…
```
# Make DataFrame Pipeline Interactive
idf = df.interactive()
```

map each country to its continent

In [247…
```
# see the pycountry_ onverter documentation
def convert (row):
    # convert country name to country code
    cn_code = pc.country_name_to_country_alpha2(row.Country, cn_name_format= "default")

    # convert country code to continent code
    conti_code = pc.country_alpha2_to_continent_code(cn_code)
    return conti_code
```

In [275…
```
# crete a new column with continent code
df["continent"]= df.apply(convert, axis = 1)
```

In [276…
```
df["continent"].unique()
```

Out[276]:
```
array(['AS', 'EU', 'AF', 'NA', 'SA', 'OC'], dtype=object)
```

In [277…
```
# map continent code to continent name
conti_name = {"AS": "Asia",
             "SA": "South America",
             "EU":"Europe",
             "AF": "Africa",
             "NA":"North America",
             "OC": "Oceania"}
df["continent"] = df["continent"].map(conti_name)
```

CO2 emission over time by continent

In [403…
```
# Define Panel widgets
year_slider = pn.widgets.IntSlider(name='Year slider', start=1750, end=2021, step=5, value=1850)
year_slider
```

Out[403]: Year slider: **1850**

In [404…
```
idf = df.interactive()
```

In [420…
```
# Radio buttons for CO2 measures
yaxis_Total_CO2 = pn.widgets.RadioButtonGroup(
    name='Y axis',
    options=['Total_CO2', 'Per Capita',],
```

```
    button_type='success'
)
yaxis_Total_CO2
```

Out[420]:

Total_CO2 | Per Capita

In [ ]:

```
co2_pipeline = (
    idf[
        (idf.Year <= year_slider)
    ]
    .groupby(['continent', 'Year'])["Total_CO2"].mean()
    .to_frame()
    .reset_index()
    .sort_values(by='Year')
    .reset_index(drop=True)
)
co2_pipeline
```

In [421…

```
# plot grapgh using the c02 pipeline
co2_plot = co2_pipeline.hvplot(x = 'Year', by='continent', y=yaxis_Total_CO2,line_width=2, title="CO2 emission by continent")
co2_plot
```

Out[421]:

Year slider: **1925**

Total_CO2 | Per Capita

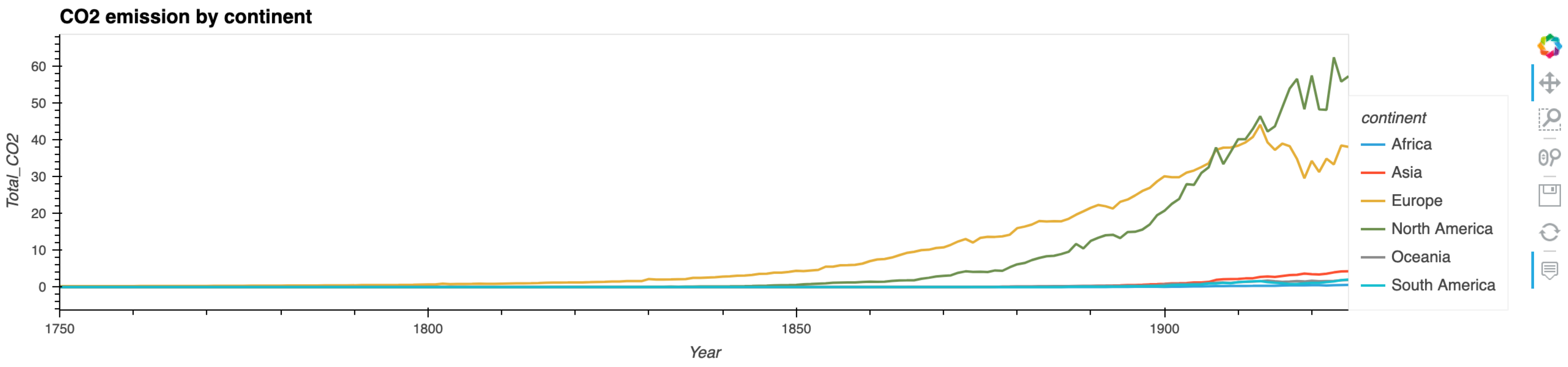


Table - CO2 emission over time by continent

In [392…

```
co2_table = co2_pipeline.pipe(pn.widgets.Tabulator, pagination='remote', page_size = 9, sizing_mode='stretch_width')
co2_table
```

Out[392]: Year slider: **1990**

index	continent	Year	Total_CO2
0	Africa	1,750	0.0
1	Oceania	1,750	0.0
2	North America	1,750	0.0
3	Europe	1,750	0.233763
4	Asia	1,750	0.0
5	South America	1,750	0.0
6	North America	1,751	0.0
7	South America	1,751	0.0
8	Europe	1,751	0.233763

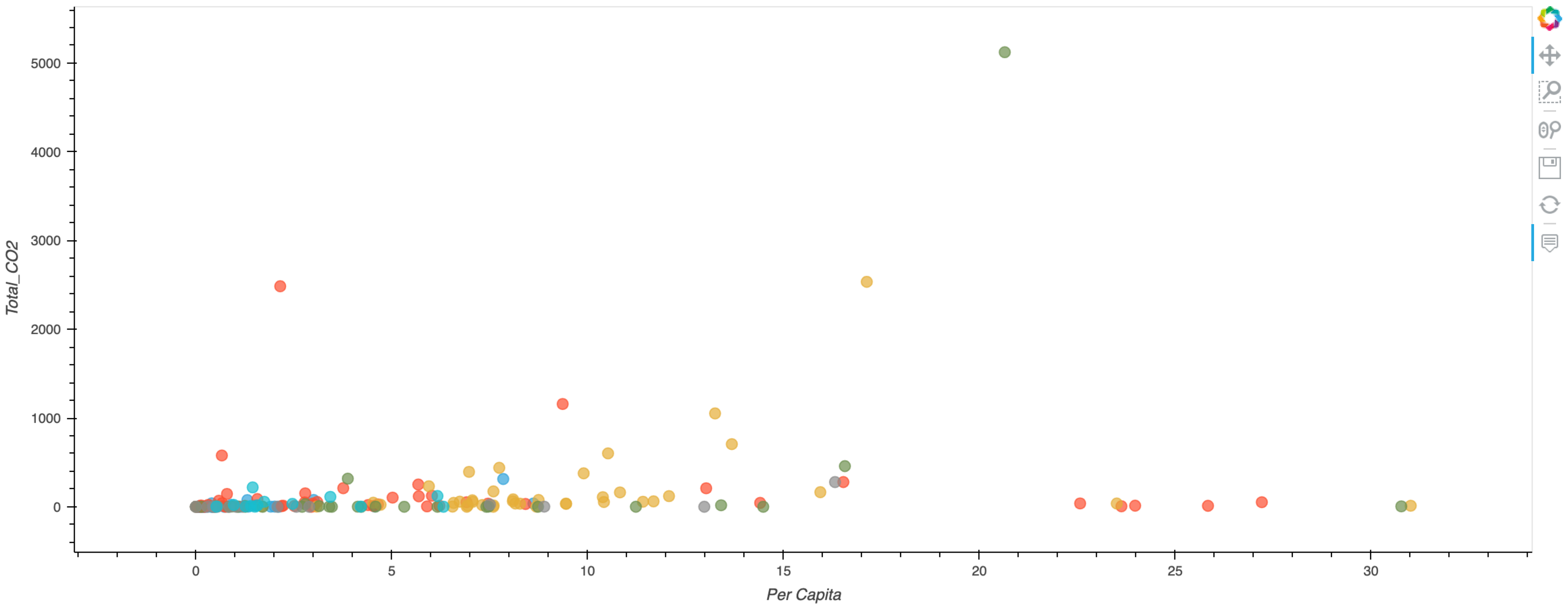
First Prev 1 2 3 4 5 Next Last

CO2 vs GDP scatterplot

In [ ]:
```
co2_vs_per_capita_scatterplot_pipeline = (
    idf[
        (idf.Year == year_slider)
    ]
    .groupby(['continent', 'Year', 'Per Capita'])['Total_CO2'].mean()
    .to_frame()
    .reset_index()
    .sort_values(by='Year')
    .reset_index(drop=True)
)
co2_vs_per_capita_scatterplot_pipeline
```

In [424…
```
co2_vs_per_capita_scatterplot = co2_vs_gdp_scatterplot_pipeline.hvplot(x='Per Capita',
                                                                        y='Total_CO2',
                                                                        by='continent',
                                                                        size=80, kind="scatter",
                                                                        alpha=0.7,
                                                                        legend=False,
                                                                        height=500,
                                                                        width=500)
co2_vs_per_capita_scatterplot
```

Out[424]: Year slider: 1990



Bar chart with CO2 sources by continent

In [394…

```
yaxis_co2_source = pn.widgets.RadioButtonGroup(
    name='Y axis',
    options=['Coal', 'Oil', 'Gas', "Cement", "Flaring", "Other"],
    button_type='success'
)



co2_source_bar_pipeline = (
    idf[
        (idf.Year == year_slider)
    ]
    .groupby(['Year', 'continent'])[yaxis_co2_source].sum()
    .to_frame()
    .reset_index()
    .sort_values(by='Year')
    .reset_index(drop=True)
)
co2_source_bar_pipeline
```

Out[394]:

Year slider: **1990**

Coal | Oil | Gas | Cement | Flaring | Other

	Year	continent	Coal
0	1990	Africa	295.753071
1	1990	Asia	3252.448897
2	1990	Europe	2980.499273
3	1990	North America	1930.964950
4	1990	Oceania	147.035089
5	1990	South America	66.518716

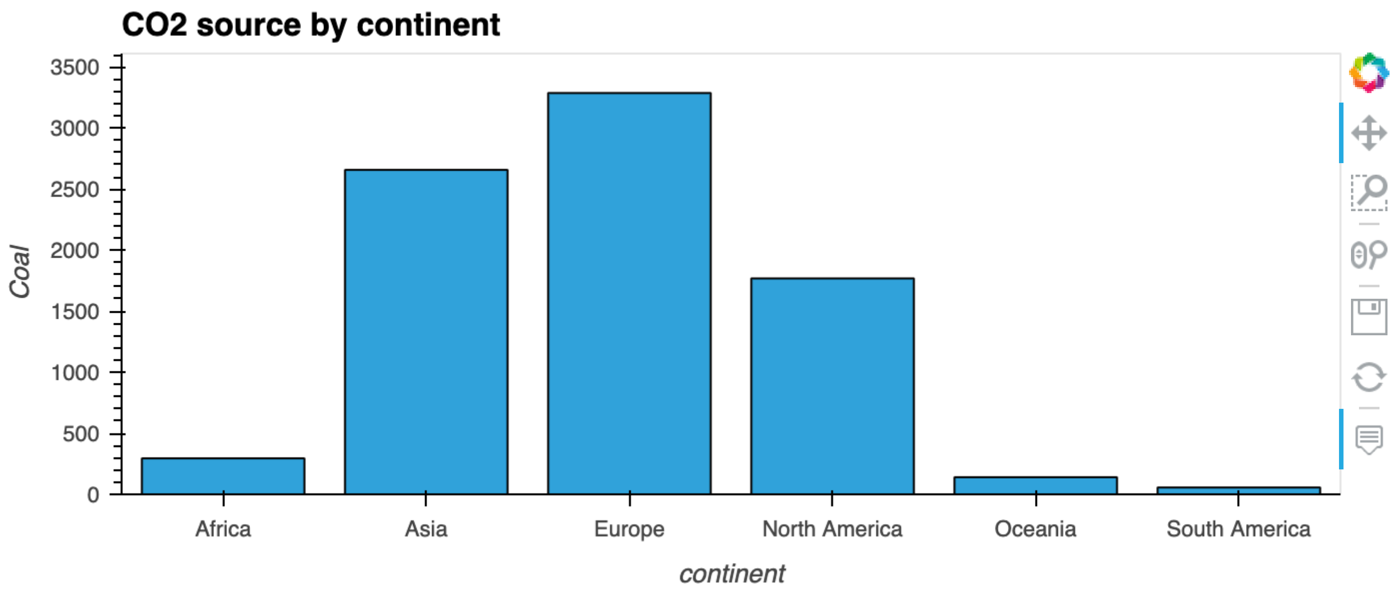
In [342…

```
co2_source_bar_plot = co2_source_bar_pipeline.hvplot(kind='bar',
                                                     x='continent',
                                                     y=yaxis_co2_source,
                                                     title='CO2 source by continent')
co2_source_bar_plot
```

Out[342]:

Year slider: **1985**

Coal | Oil | Gas | Cement | Flaring | Other



Creating Dashboard

In [ ]:

```
#Layout using Template
pn.extension(sizing_mode = 'stretch_width')
template = pn.template.FastListTemplate(
    title='World CO2 emission dashboard',
    sidebar=[pn.pane.Markdown("# CO2 Emissions and Climate Change"),
             pn.pane.Markdown("#### Carbon dioxide emissions are the primary driver of global climate change. It’s widely recognised that to avoid the worst impacts of climate cl
             pn.pane.PNG('climate_day.png', sizing_mode='scale_both'),
             pn.pane.Markdown("## Settings"),
             year_slider],
```

```
    main=[pn.Row(pn.Column(yaxis_co2,
                           co2_plot.panel(width=700), margin=(0,25)),
                 co2_table.panel(width=500)),
          pn.Row(pn.Column(co2_vs_gdp_scatterplot.panel(width=600), margin=(0,25)),
                 pn.Column(yaxis_co2_source, co2_source_bar_plot.panel(width=600)))],
   accent_base_color="#88d8b0",
    header_background="#88d8b0",
)
# template.show()
template.servable();
template.show()
```

In [ ]:

In [ ]:

In [ ]:

In [ ]:

In [ ]: