Economic Data Analysis with Fred and Pandas

```
!pip install fredapi
```

```
Collecting fredapi
  Downloading fredapi-0.5.2-py3-none-any.whl.metadata (5.0 kB)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from f
Requirement already satisfied: numpy>=1.23.2 in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.11/dist-packages (
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (fron
Downloading fredapi-0.5.2-py3-none-any.whl (11 kB)
Installing collected packages: fredapi
Successfully installed fredapi-0.5.2
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import time

plt.style.use('fivethirtyeight')
pd.set_option('display.max_columns',500)
color_pal=plt.rcParams['axes.prop_cycle'].by_key()['color']

from fredapi import Fred

fred_key= 'enter your key here'
```

Create Fred object

```
fred= Fred(api_key=fred_key)
```

Search Economic data

```
sp_search=fred.search('S&P', order_by='popularity')
```

```
sp_search.shape
```

```
(1000, 15)
```

```
sp_search
```

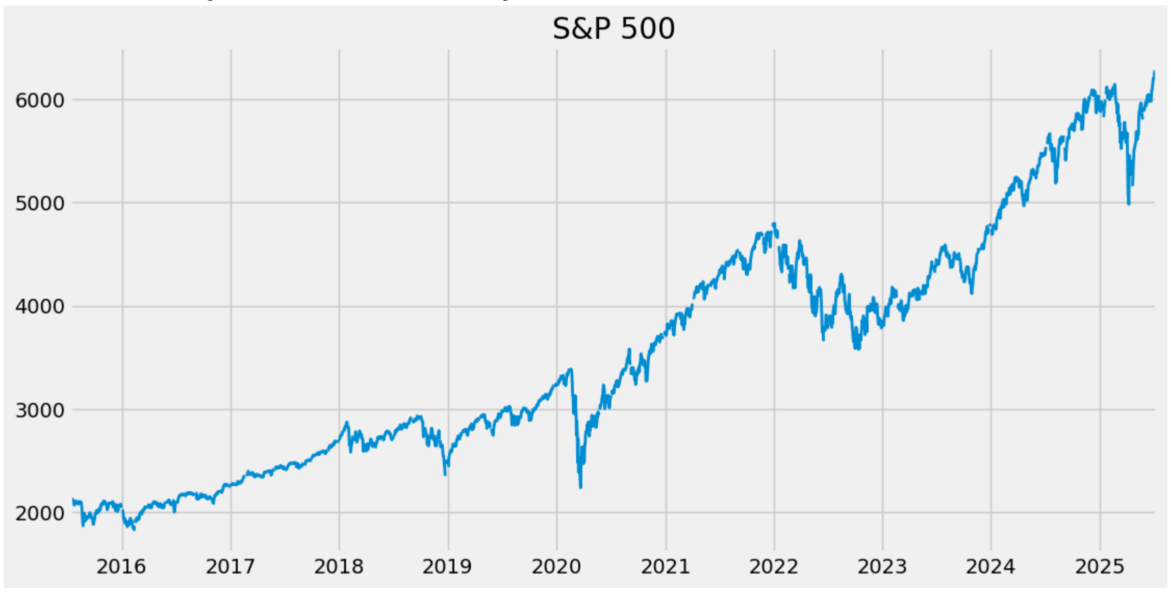
	id	realtime_start	realtime_end	tit
series id				
BAMLH0A0HYM2	BAMLH0A0HYM2	2025-07-14	2025-07-14	ICE BofA High Yi Inc Optic Adjusted S
CSUSHPINSA	CSUSHPINSA	2025-07-14	2025-07-14	S CoreLo Case-Shi U Natio Home
SP500	SP500	2025-07-14	2025-07-14	S&P 5
BAMLH0A0HYM2EY	BAMLH0A0HYM2EY	2025-07-14	2025-07-14	ICE BofA High Yi Inc Effect Yi
BAMLC0A0CM	BAMLC0A0CM	2025-07-14	2025-07-14	ICE BofA Corpora Inc Optic Adjust Sp
...	...	...	...	
DDDI12SMA156NWDB	DDDI12SMA156NWDB	2025-07-14	2025-07-14	Priva Credit Depo Mor Banks a Othe
Q03069USQ605NNBR	Q03069USQ605NNBR	2025-07-14	2025-07-14	Rever Freight To Originate Less Th Ca
CSHICPCZA156NRUG	CSHICPCZA156NRUG	2025-07-14	2025-07-14	Share Gro Cap Formation Current Pr
				Share

Pull raw data

```
sp500=fred.get_series(series_id='SP500')
```

```
sp500.plot(figsize=(12,6), title='S&P 500', lw=2)
```

```
<Axes: title={'center': 'S&P 500'}>
```



Pull and join Multiple Data series

```
unemployment_results=fred.search('unemployment')
```

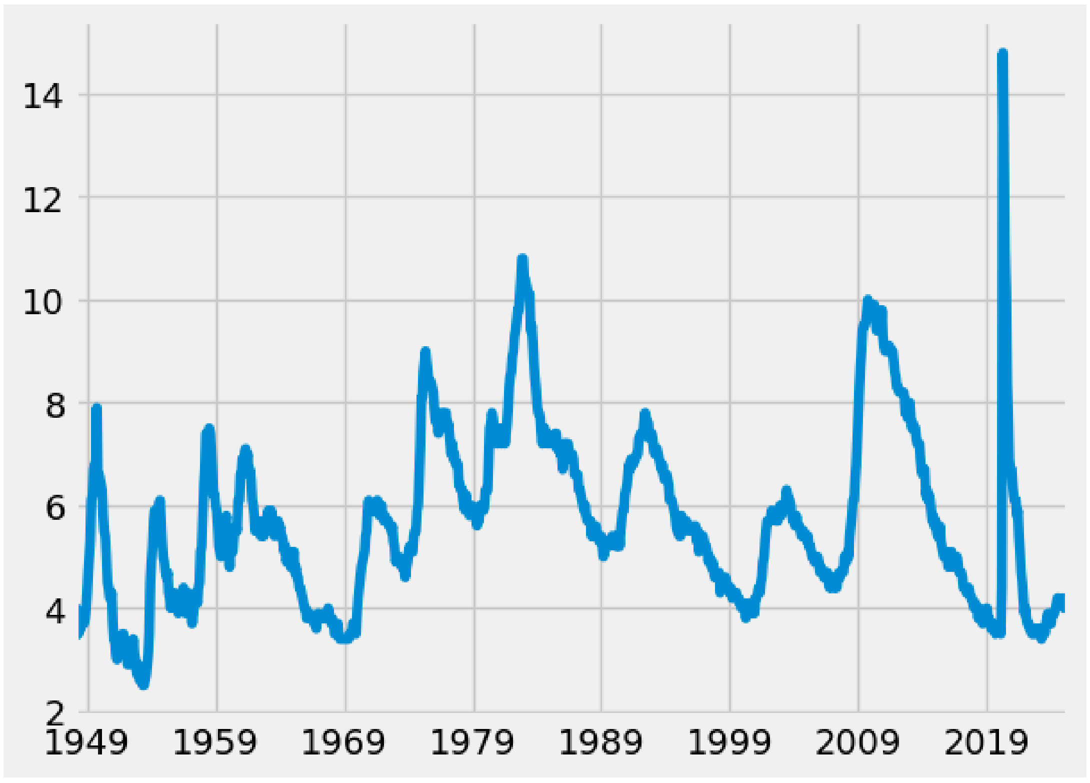
```
unemployment_results
```

	id	realtime_start	realtime_end	t
series id				
UNRATE	UNRATE	2025-07-14	2025-07-14	Unemploy
UNRATENSA	UNRATENSA	2025-07-14	2025-07-14	Unemploy
UNEMPLOY	UNEMPLOY	2025-07-14	2025-07-14	Unemploy
NROU	NROU	2025-07-14	2025-07-14	Noncy R Unemploy
CCSA	CCSA	2025-07-14	2025-07-14	Cont Claims (In Unemploy
...	...	...	...	
ENUC317430010	ENUC317430010	2025-07-14	2025-07-14	Total Qua Wag Manhatta (
IPUEN3116L020000000	IPUEN3116L020000000	2025-07-14	2025-07-14	Compens Manufact Anim
IPUEN3116U110000000	IPUEN3116U110000000	2025-07-14	2025-07-14	Compens Manufact Anim
ENUC148640110SA	ENUC148640110SA	2025-07-14	2025-07-14	Av Weekly W for Emplo in Fede
ENUC148640110	ENUC148640110	2025-07-14	2025-07-14	Av Weekly W for Emplo in Fede

1000 rows × 15 columns

```
unrate=fred.get_series('UNRATE')
unrate.plot()
```

```
<Axes: >
```



```
unemp_df=fred.search('unemployment rate state', filter=('frequency', 'Monthly'))
```

```
unemp_df
```

	id	realtime_start	realtime_end	
series id				
UNRATE	UNRATE	2025-07-14	2025-07-14	Unem
UNRATENSA	UNRATENSA	2025-07-14	2025-07-14	Unem
LNS14000006	LNS14000006	2025-07-14	2025-07-14	Unem Rate - A
UNEMPLOY	UNEMPLOY	2025-07-14	2025-07-14	Unem
LNU03000000	LNU03000000	2025-07-14	2025-07-14	Unem
...	...	...	...	
LAUCN300190000000005	LAUCN300190000000005	2025-07-14	2025-07-14	E P Co
LAUCN470950000000005	LAUCN470950000000005	2025-07-14	2025-07-14	E P Lake
LAUCN220350000000005	LAUCN220350000000005	2025-07-14	2025-07-14	E P Ea P
LAUCN021300000000005	LAUCN021300000000005	2025-07-14	2025-07-14	E P K Bor
				E P

```
unemp_df = unemp_df.query('seasonal_adjustment == "Seasonally Adjusted" and units == "Percent
unemp_df.shape
```

```
(52, 15)
```

```
unemp_df=unemp_df.loc[unemp_df['title'].str.contains('Unemployment Rate')]
```

```
unemp_df.shape
```

```
(46, 15)
```

```
all_results=[]
for myid in unemp_df.index:
  results=fred.get_series(myid)
  results=results.to_frame(name=myid)
  all_results.append(results)
```

```
type(all_results)
```

```
list
```

```
all_results[4]
```

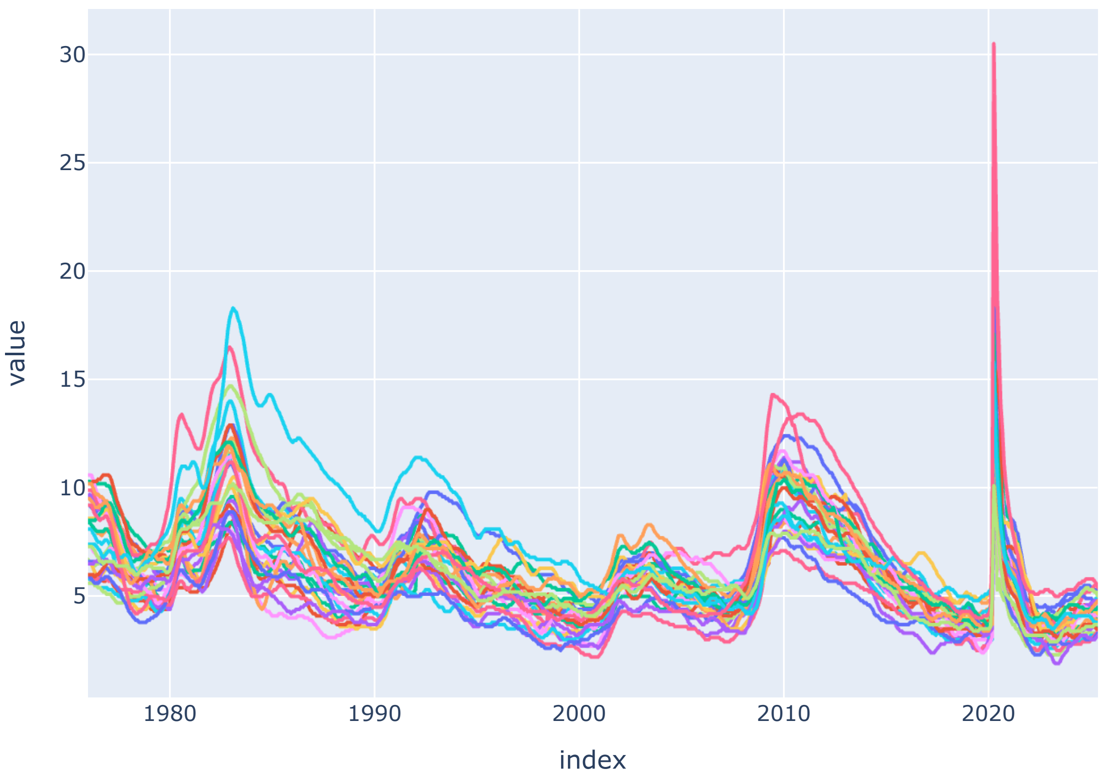
	M0892AUSM156SNBR
1929-04-01	0.69
1929-05-01	1.65
1929-06-01	2.06
1929-07-01	0.79
1929-08-01	0.04
...	...
1942-02-01	3.56
1942-03-01	3.22
1942-04-01	2.33
1942-05-01	1.22
1942-06-01	0.24

159 rows × 1 columns

```
unemp_results=pd.concat(all_results, axis=1)
```

```
cols_to_drop = []
for i in unemp_results:
    if len(i) > 4:
        cols_to_drop.append(i)
unemp_results = unemp_results.drop(columns = cols_to_drop, axis=1)
```

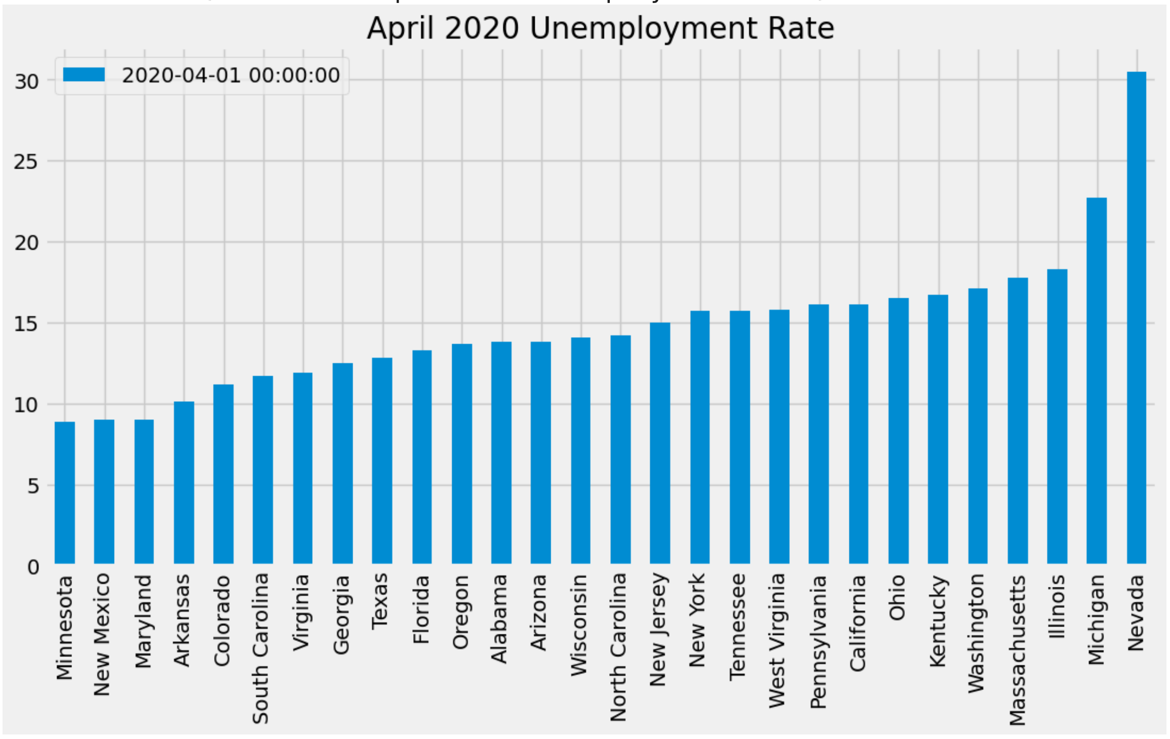
```
uemp_states = unemp_results.copy()  #.drop('UNRATE', axis=1)
uemp_states = uemp_states.dropna()
id_to_state = unemp_df['title'].str.replace('Unemployment Rate in ','').to_dict()
uemp_states.columns = [id_to_state[c] for c in uemp_states.columns]


# Plot States Unemployment Rate
px.line(uemp_states)
```



Pull april 2020 Unemployment rate per state

```
apr_2020rate=uemp_states.loc[uemp_states.index == '2020-04-01'].T.sort_values('2020-04-01')

apr_2020rate.plot(kind='bar', figsize=(12,6), title='April 2020 Unemployment Rate')
```

```
<Axes: title={'center': 'April 2020 Unemployment Rate'}>
```



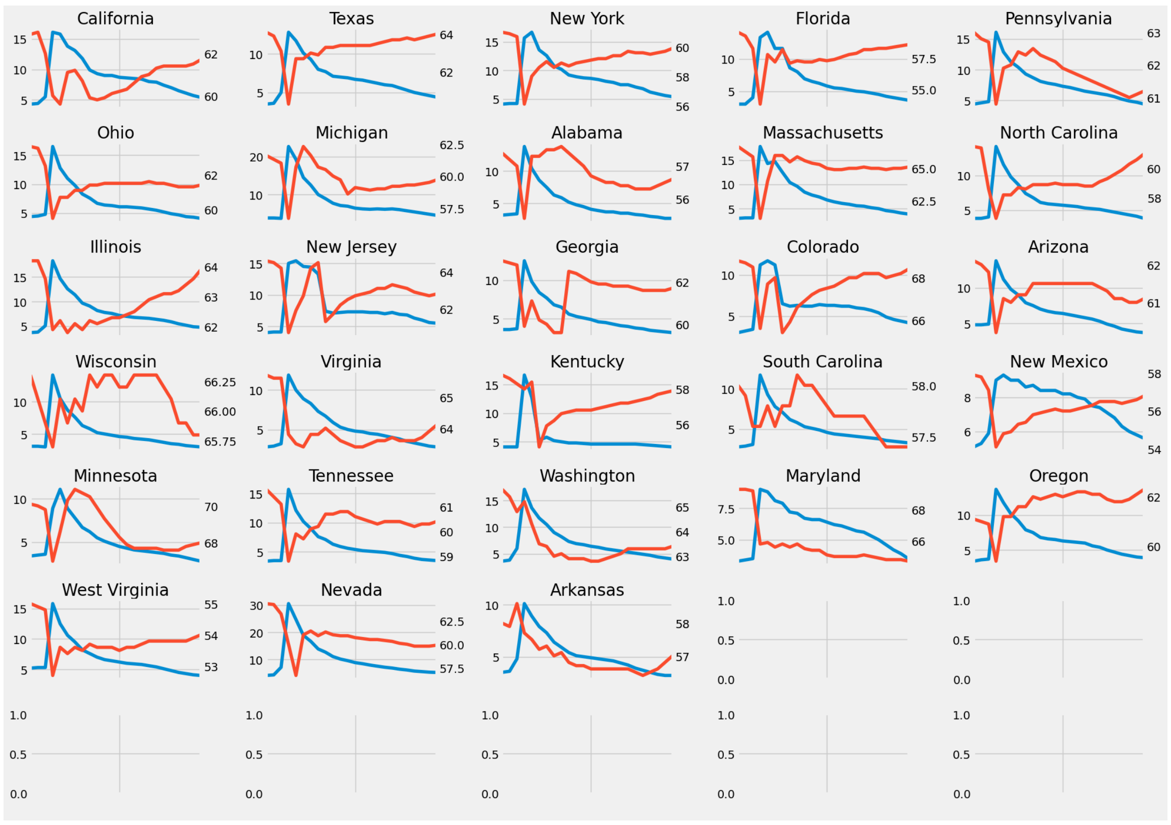
Pull Participation rate

```
part_df = fred.search('participation rate state', filter=('frequency','Monthly'))
part_df = part_df.query('seasonal_adjustment == "Seasonally Adjusted" and units == "Percent'
```

```
part_id_to_state = part_df['title'].str.replace('Labor Force Participation Rate for ','').t

all_results = []

for myid in part_df.index:
    results = fred.get_series(myid)
    results = results.to_frame(name=myid)
    all_results.append(results)
    time.sleep(0.1) # Don't request to fast and get blocked
part_states = pd.concat(all_results, axis=1)
part_states.columns = [part_id_to_state[c] for c in part_states.columns]
```

Plot Unemp vs Participation

```
# Fix DC
uemp_states = uemp_states.rename(columns={'the District of Columbia':'District Of Columbia'
fig, axs = plt.subplots(10, 5, figsize=(20, 20), sharex=True)
axs = axs.flatten()

i = 0
for state in uemp_states.columns:
    if state in ["District Of Columbia","Puerto Rico"]:
        continue
    ax2 = axs[i].twinx()
    uemp_states.query('index >= 2020 and index < 2022')[state] \
        .plot(ax=axs[i], label='Unemployment')
    part_states.query('index >= 2020 and index < 2022')[state] \
        .plot(ax=ax2, label='Participation', color=color_pal[1])
    ax2.grid(False)
    axs[i].set_title(state)
    i += 1
plt.tight_layout()
plt.show()
```


California
Texas
New York
Florida
Pennsylvania
Ohio
Michigan
Alabama
Massachusetts
North Carolina
Illinois
New Jersey
Georgia
Colorado
Arizona
Wisconsin
Virginia
Kentucky
South Carolina
New Mexico
Minnesota
Tennessee
Washington
Maryland
Oregon
West Virginia
Nevada
Arkansas