

VIII. Plotly 사용과 배포

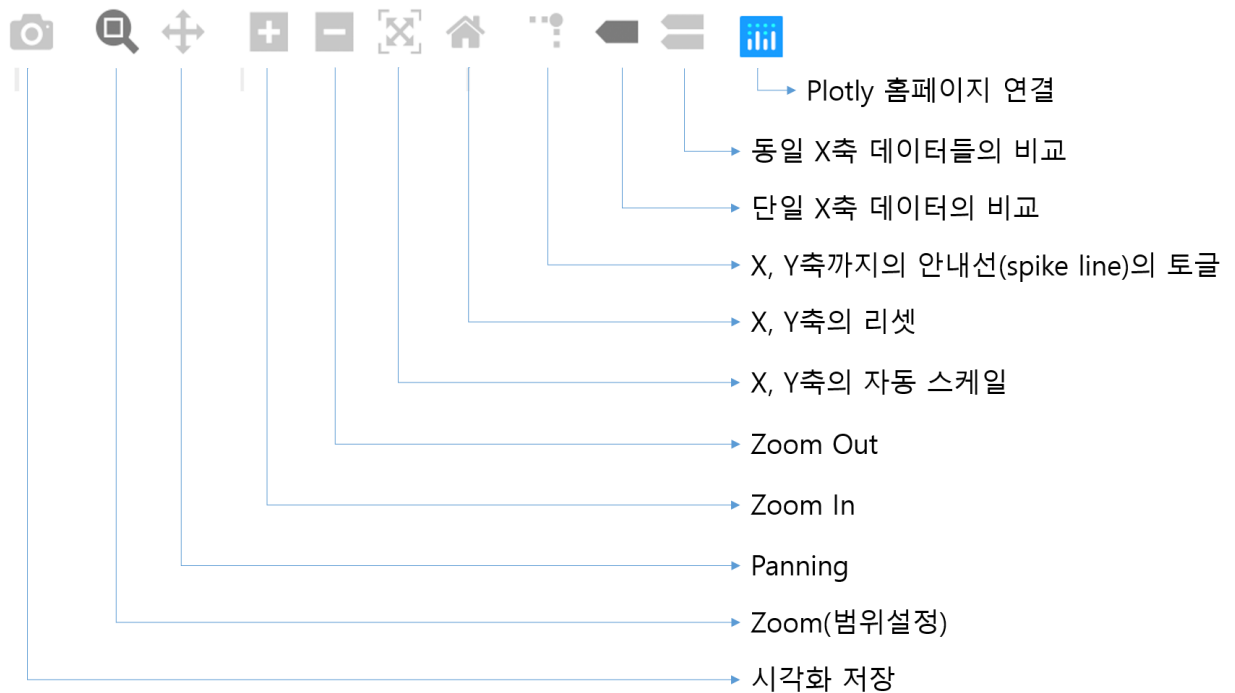
~~plotly 는 동적 시각화인 만큼 Zoom in, Zoom out 뿐 아니라 사용자와의 상호 작용을 통한 많은 기능을 제공한다.~~ 이번 장에서는 plotly 시각화에서 제공하는 다양한 기능을 알아본다.

1. plotly 시각화 사용하기

R 의 `ggplot2` 나 python 의 `matplotlib`, `seaborn` 로 만든 정적 시각화는 그래프를 만들때 시각화한 데이터 외에 시각화 자체에서 추가적인 데이터를 얻기가 어렵다. 따라서 시각화에 추가적인 데이터를 제공하기 위해서는 다시 코딩해서 만들어야하는 불편함이 따른다. 특히 특정 위치의 데이터 값을 확인하거나 특정 구간 데이터를 zoom 하기 위해서도 다시 코딩해야하는데 사용자의 사용을 예상하여 따라 수없이 많은 시각화를 만들어 놓을 수는 없다. 반면 plotly 와 같은 동적 시각화에서는 특징적 데이터 값의 확인, zoom, zoom out, 특정 데이터만의 표기 등과 같은 탐색적 데이터 분석에 자유롭게 사용할 수 있는 다양한 기능을 제공한다.

1.1. modebar 의 사용

plotly 가 시각화 사용자와의 상호작용을 위한 주요 기능을 제공하는 메뉴가 'modebar'이다. 'modebar'는 plotly 가 실행되는 R-Studio, Jupiter Notebook 이나 웹 브라우저의 오른쪽 상단에 나타나는 버튼 메뉴를 말한다. scatter 트레이스에 나오는 'modebar'는 다음의 그림과 같이 8 개의 기능을 버튼을 통해 제공한다. 각각의 트레이스에 따라 제공하는 'modebar'가 달라지는데 scatter 트레이스에 가장 기본적인 'modebar'가 나온다.



modebar 버튼과 기능

1.1.1. 시각화 저장 버튼

시각화 저장 기능은 기본적으로 모드바의 가장 왼쪽에 카메라 아이콘으로 표현된다. 이 기능은 현재 표시되는 plotly 시각화를 정적 이미지로 저장한다. plotly에서는 기본적으로 'png'타입으로 이미지의 저장이 가능하다.

1.1.2. Zoom()과 Pan() 기능 버튼

plotly 시각화는 기본적으로 마우스로 시각화의 플롯 영역에서 좌클릭한 상태로 드래그하면 시각화의 확대 영역을 설정할 수 있다. 이 기능은 plotly 시각화에서 마우스의 기본 기능 설정이기 때문에 'Zoom' 버튼을 누르지 않고 가능하다. Zoom 기능을 완료하고 원 상태의 시각화로 돌아가기 위해서는 모드바의 리셋 버튼을 누르거나 마우스 더블 클릭으로 돌아갈 수 있다.

Pan 기능은 시각화의 표현 비율을 그대로 두고 그래프를 이동시키는 기능이다. 이 기능을 사용하려면 모드바의 'Pan' 버튼을 누른 다음 마우스 좌클릭한 상태에서 움직이면 그래프가 같이 움직이게 된다. 그래프의 이동에 따라 X 축과 Y 축도 같이 이동하는 것을 확인할 수 있다.

Pan 기능을 사용하고 원래 시각화로 돌아가기 위해서는 모드바의 리셋 버튼을 누르거나 마우스 더블 클릭으로 돌아갈 수 있다.

1.1.3. Zoom In()과 ZoomOut() 기능 버튼

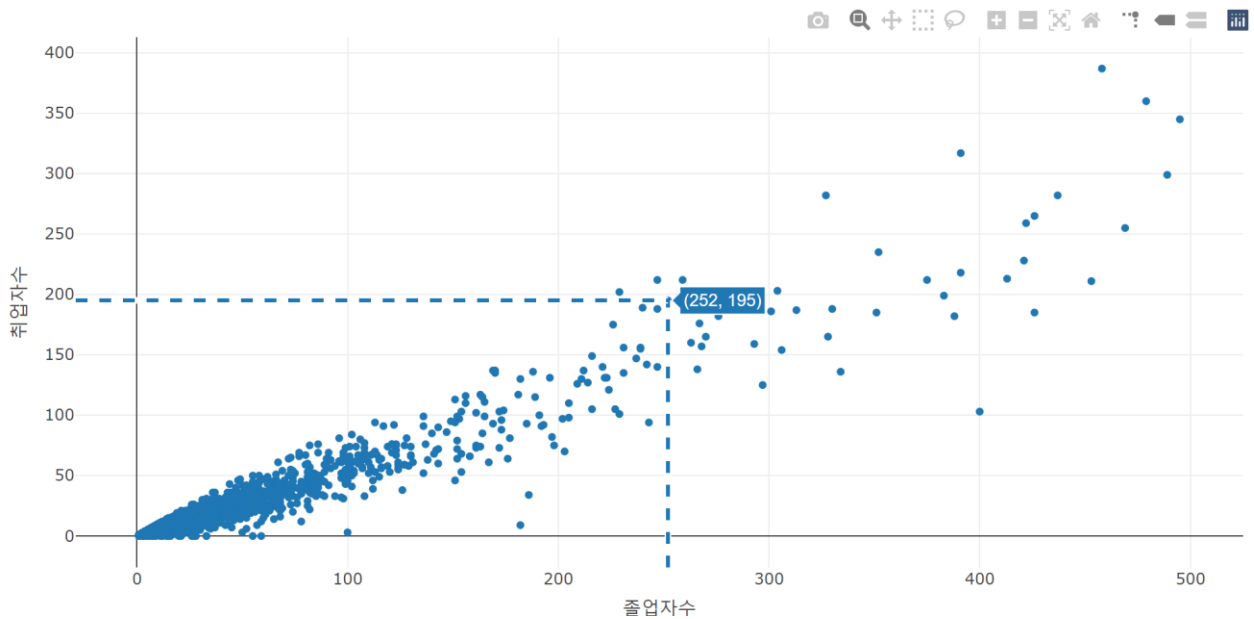
앞서 Zoom 버튼은 마우스 드래그를 통해 Zoom 기능을 활용하였다. 하지만 Zoom In/Out 기능 버튼의 ‘+’ 버튼을 누를때 마다 현재 시각화의 중심으로 Zoom In 이 되고 ‘-’ 버튼을 누를때 마다 현재 시각화의 중심에서 Zoom Out 이 된다.

1.1.4. Autoscale()과 Reset Axes() 기능 버튼

plotly 를 만들때 X 축과 Y 축의 범위를 설정하지 않으면 plotly 는 표시되는 데이터의 사이즈에 맞게 X 축과 Y 축의 범위를 자동 설정한다. 이 기능을 ‘Autoscale’이라고 하는데 ‘Autoscale’ 버튼은 plotly 가 시각화를 생성할 당시 자동적으로 설정했던 축 설정으로 되돌아 가는 버튼이다. ‘reset axes’ 버튼은 초기 시각화로 돌아가는 기능을 제공한다.

1.1.5. toggle spike line() 기능 버튼

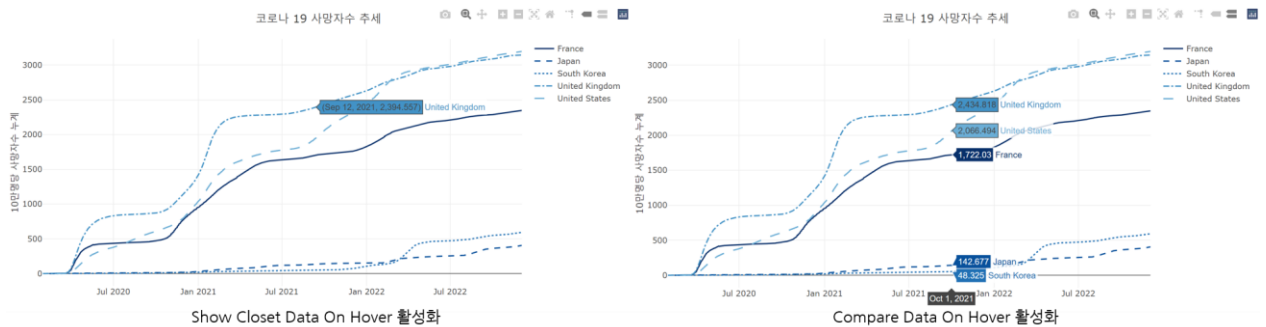
‘Toggle Spike Line’ 버튼은 X 축, Y 축으로 그려지는 보조선의 설정을 변경하는 버튼이다. 이 버튼으로 스파이크 라인을 활성화하면 X 축과 Y 축으로 ‘toaxis’가 설정된 스파이크 라인이 설정되고 다시 한번 누르면 스파이크 라인이 없어진다.



spike 기능버튼 사용시 결과

1.1.6. Show Closet Data On Hover()/Compare Data On Hover() 기능

이 두 버튼은 호버의 설정과 관련한 기능을 조절하는 버튼이다. 'Show Closet Data On Hover' 버튼은 데이터에 가장 가까운 데이터에 호버가 나타나는 기능으로 설정하는 버튼이고 'Compare Data On Hover'는 X 축으로 동일한 데이터에 대한 호버가 표시되는 기능, 즉 'hovermode'를 "x"로 설정하는 버튼이다.



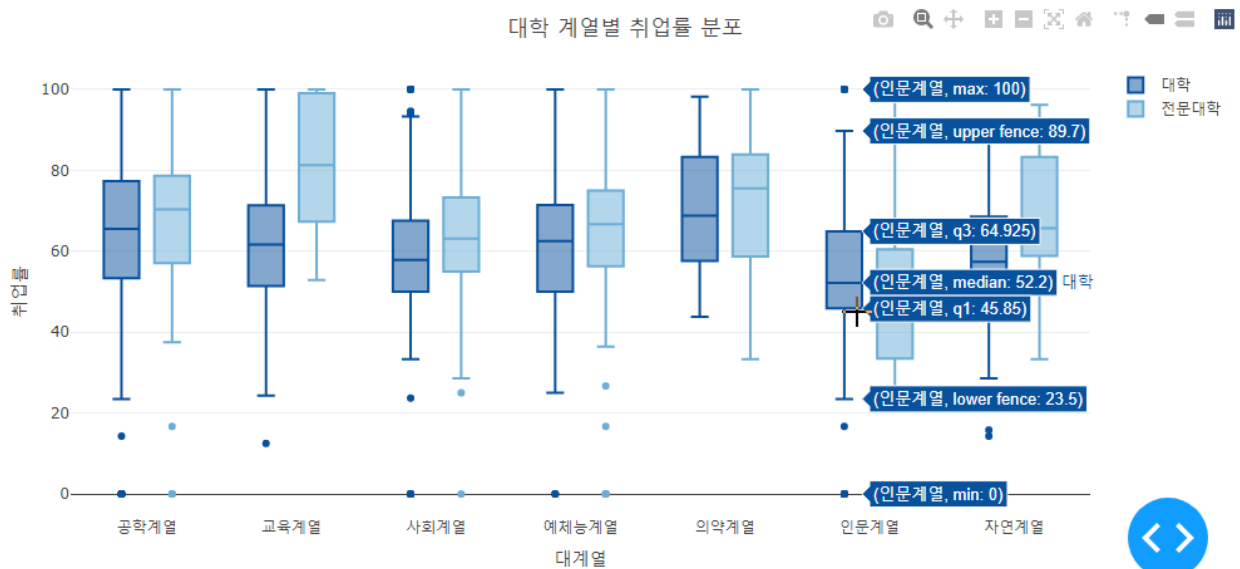
호버 기능 버튼을 사용한 결과

1.2. 마우스의 기능

1.2.1. 호버를 통한 데이터 확인

plotly 시각화에서 가장 쉽게 사용하는 기능은 마우스를 사용하여 해당 위치의 데이터 정보를 확인하는 기능이다. plotly 객체로 생성된 시각화 위에 표현된 각 트레이스들은 자체 데이터를 JSON 의 형태로 포함하고 있기 때문에 마우스 포인터를 트레이스위에 위치시키면 호버를 통해 해당 트레이스의 정보가 표시된다.

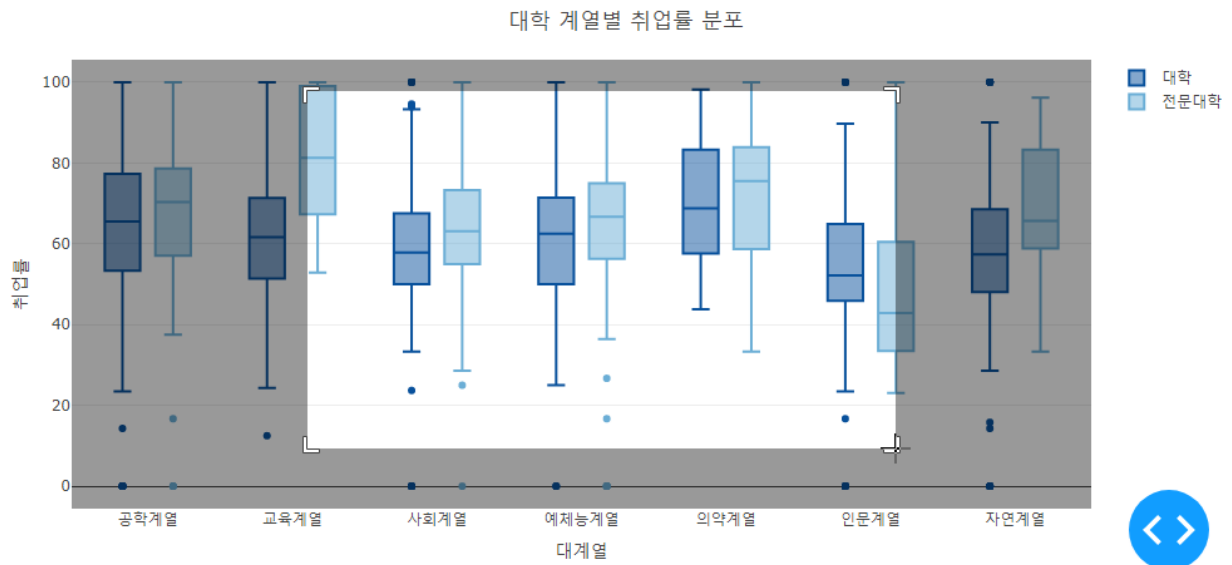
마우스 포인터에 의해 표시되는 호버의 정보는 각 트레이스의 'hoverinfo', 'hovertemplate' 등 호버 속성 설정에 따라 표시된다.



박스 트레이스의 호버 표시

1.2.2. 드래그를 통한 Zoom In

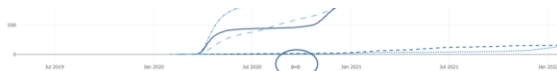
plotly 로 완성된 시각화에서 트레이스가 나타나는 플롯팅 영역(plotting area)에서 마우스 왼쪽 버튼을 클릭한 상태에서 드래그하면 다음 그림과 같이 줌 인 영역을 선택할 수 있다. 이렇게 영역을 선택한 후에 마우스 클릭을 놓으면 해당 부분이 줌인 되어 표시되게 된다. 만약 다시 처음의 상태로 돌아가려면 모드바의 집 아이콘인 'Reset Axes' 버튼을 사용한다.



마우스 드래그를 통한 줌

1.2.3. 드래그를 통한 축 이동과 압축(팽창)

plotly 에서 마우스를 드래그를 통해 추가적으로 할 수 있는 기능은 축 이동과 축 압축이다. X 축과 Y 축의 위치에서 마우스의 왼쪽 버튼을 클릭한 상태에서 상하 또는 좌우로 드래그를 하면 축의 범위가 조절된다. 이를 통해 데이터가 표현되는 축의 범위를 변경할 수 있다. 만약 처음 눈금 라벨의 아래나 마지막 눈금 라벨의 위(Y 축), 처음 눈금 라벨의 왼쪽이나 마지막 눈금 라벨의 오른쪽(X 축)에서 마우스를 클릭하고 드래그하면 축의 원점이나 최종점이 고정된 채 축을 압축하거나 팽창 할 수 있다.



축의 이동

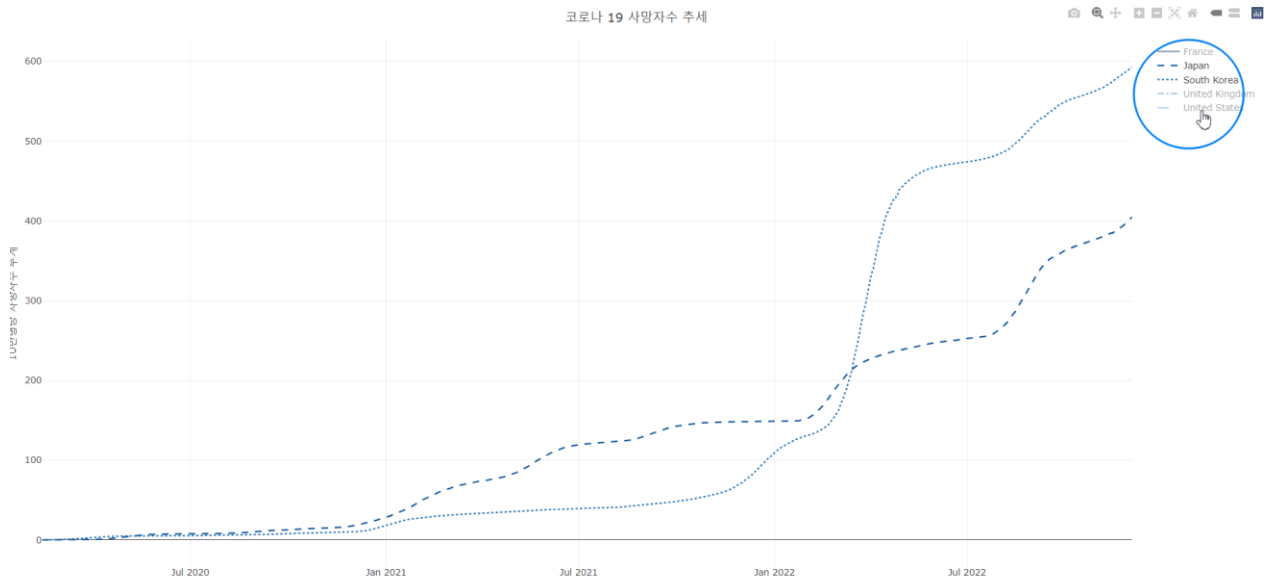


축의 압축

1.3. 범례의 사용

plotly 에서 범례는 단순히 트레이스 이름과 표현방식을 매핑해주는 역할을 넘어 트레이스들의 표시를 조절할 수 있는 기능이 있다. 범례의 아이템을 클릭하면 해당 트레이스의 표시를 토글하는 역할을 하는 것으로, 여러 데이터 트레이스 중에 특정한 트레이스만을 확인하기 위해서 해당 트레이스만 남기고 다른 트레이스의 표시를 제거함으로써 자신이 보기 원하는 데이터만 선별하여 볼 수 있게 된다. 이 기능은 대시보드에서 흔히 제공하는 옵션 중에

하나인데 plotly 에서는 범례를 사용하여 이 기능을 기본적으로 제공한다는 것이다. 특히 이 기능이 더 편리한 것은 현재 표시되고 있는 데이터의 범위에 따라 X 축과 Y 축의 범위가 자동적으로 다시 설정되어 남은 데이터의 특징이 더 잘 보인다는 장점이 있다. 다음은 범례를 사용하여 한국과 일본의 데이터만 남긴 결과이다.



범례의 토글

2. Plotly 배포

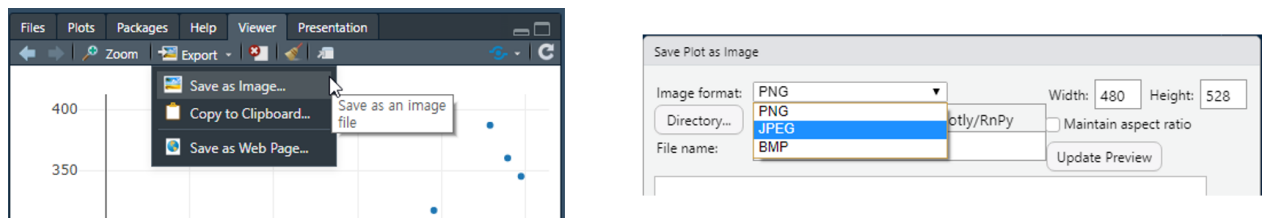
2.1. Off-line 배포

데이터 시각화는 보통 데이터 분석의 결과가 요약되는 보고서에 첨부되어 데이터 분석의 결과를 보다 설득력있게 제공하는데 많이 활용된다. plotly 의 결과도 정적 이미지로 저장하여 보고서에 포함시켜 사용할 수 있다. 이를 위해서는 앞에서 살펴본 모드바의 다운로드 기능을 사용하여 'png' 형태의 파일로 다운로드 받아 사용할 수 있지만 R 과 python 코드에서 바로 저장할 수도 있다.

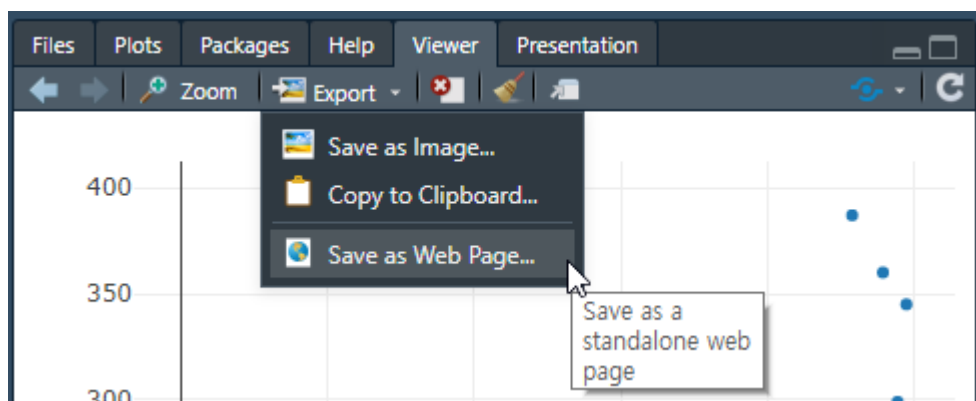
- R

R 에서 plotly 시각화를 정적 이미지로 다운로드 받기 위해서는 R-Studio 를 사용하는 방법과 코드를 사용하여 다운로드 받는 방법의 두 가지로 나눌 수 있다.

R-Studio 를 사용하는 사용자라면 보통 우측 하단 ‘Viewer’ 패널에 plotly 시각화가 표현된다
 ‘Viewer’ 패널의 ‘Export’ 기능을 사용하면 **plotly** 시각화를 jpg, png, bmp 파일로 저장할 수 있다. 이 기능을 사용하면 저장할 파일의 크기를 조절할 수 있다는 장점도 있다.



또 ‘Viewer’ 패널에서 plotly 시각화를 html 형식의 파일로도 저장이 가능하다. 파일로 저장해서 웹브라우저에서 열면 해당 시각화를 웹브라우저에서 사용할 수 있고 앞서 png, jpg, bmp 이미지 파일과는 달리 동적 시각화의 모든 기능을 사용할 수 있다는 장점이 있다.



이렇게 사용자가 직접 이미지나 HTML 로 저장하는 방식이 아닌 코드에서 자동적으로 이미지 파일을 저장하기 위해서는 **plotly** 패키지에서 제공하는 **export()**를 사용하거나 **htmlwidgets** 패키지의 **savewidget()**을 사용할 수 있다. **export()**는 plotly 시각화 객체를 매개변수로 지정하고 파일명을 지정하면 해당 파일명으로 plotly 시각화가 저장된다. **export()**에서 지원하는 파일 포맷은 jpg, png, pdf 등으로 파일명 지정시 파일포맷 확장자를 지정하면 해당 파일 포맷으로 저장된다.¹

```
fig <- df_취업률_500 |>
  ## x 축은 졸업자수, y 축은 취업자수로 매핑한 plotly 객체 생성
```

¹ plotly 는 export()보다는 orca()를 사용하라고 권고하고 있으나 orca()는 한글 변환에 문제가 있어 여기서는 export()를 위주로 설명한다. 또 WebGL 이나 svg 파일 포맷으로 저장할 때는 매개변수로 **RSelenium::rsDriver()** 설정이 필요하다.


```

plot_ly() |>
  add_trace(type = 'scatter', mode = 'markers',
            x = ~졸업자수, y = ~취업자수)

export(fig, file = 'fig.png')

```

만약 HTML 파일로 저장하기 위해서는 `htmlwidgets` 패키지를 설치하고 `savewidget()`을 다음과 같이 사용할 수 있다.

```
htmlwidgets::saveWidget(widget = fig, 'fig.html')
```

- python

python 에서 plotly 시각화를 정적 이미지로 다운로드 받기 위해서는 `Kaleido` 라이브러리나 `orca` 라이브러리를 설치하여야 한다. plotly 에서는 `Kaleido` 라이브러리를 설치를 권장한다.² 이 라이브러리는 python 의 plotly 객체를 저장하기 위해 꼭 필요하지만 직접 임포트를 하지 않아도 되고 `plolty` 라이브러리의 `write_image()`을 사용할 때 내부적으로 사용된다.

```
> pip install Kaleido
```

plotly 객체를 저장하기 위해 `write_image()`을 사용한다. `write_image()`에서 저장 가능한 파일 포맷은 래스터 이미지 파일인 png, jpg, webp 와 벡터 이미지 파일인 svg, pdf 등 이다. 이들 파일 포맷의 결정은 `write_image()`의 저장 파일명을 설정할 때 파일 확장자에 따라 결정된다.

```
fig.write_image("fig.png")
```

만약 HTML 파일 포맷으로 저장해야 한다면 `write_html()`을 사용한다.

```
fig.write_html("fig.html")
```

2.2. On-line 배포

`plolty` 와 같은 동적 시각화는 사실 보고서에 넣는 것 보다는 온라인 웹페이지를 통해서 배포할때 그 효과가 극대화된다. 앞서 오프라인 배포에서도 HTML 파일로 저장이 가능한데 이 HTML 파일은 plotly 의 동적 기능을 위한 자바 스크립트(plotly.js)를 포함하기 때문에 HTML 파일의 크기가 3~4MB 를 넘는다. 따라서 이렇게 큰 사이즈의 HTML 을 웹 서버에 올려서 서비스하는 것은 가능하지만 블로그에 포스팅 할 경우 파일 사이즈가 커서 업로드가 불가능한

² 윈도우 10 에서는 kaleido 라이브러리와 plolty 라이브러리가 충돌하여 `write_image()`가 hang 이 걸리는 경우가 있다. 이 경우 kaleido 라이브러리를 삭제하고 실행하면 정상 작동한다.

경우가 있다. 이런 경우는 plotly 에서 제공하는 차트 스튜디오(chart studio)를 사용하여 포스팅할 수 있다.

차트 스튜디오는 자신이 생성한 plotly 객체를 업로드하고 이 파일을 블로그에 임베딩하는 기능을 제공한다. 또한 plotly 객체의 배포외에도 웹 브라우저에서 차트를 쉽게 생성하거나 편집할 수 있는 편집기 기능도 제공한다. 차트 스튜디오는 무료 계정으로 사용이 가능한데 무료 계정은 자신이 생성한 plotly 시각화 결과를 누구나 볼 수 있는 퍼블릭 모드로만 사용할 수 있고 유료 계정의 경우 자신의 plotly 시각화 결과를 다른 사람에게 공개하지 않는 프라이빗 모드를 사용할 수 있다.

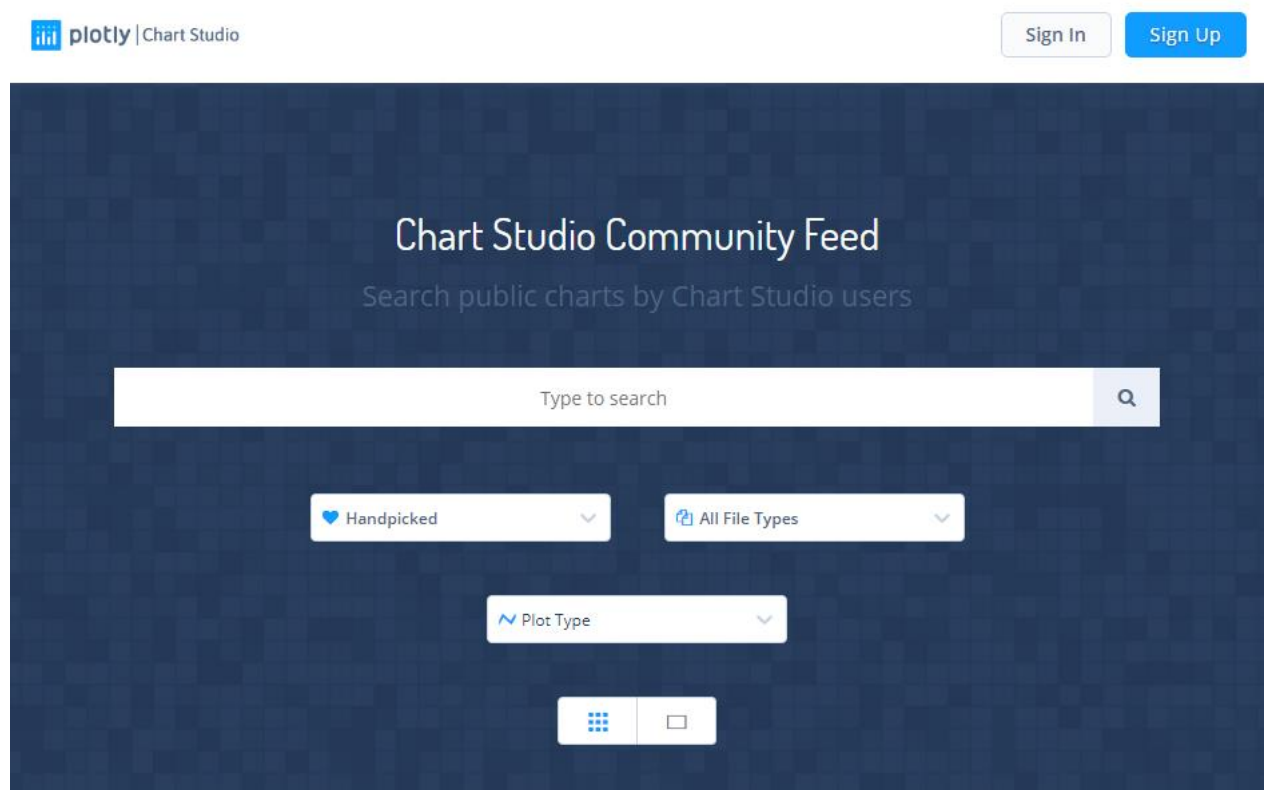
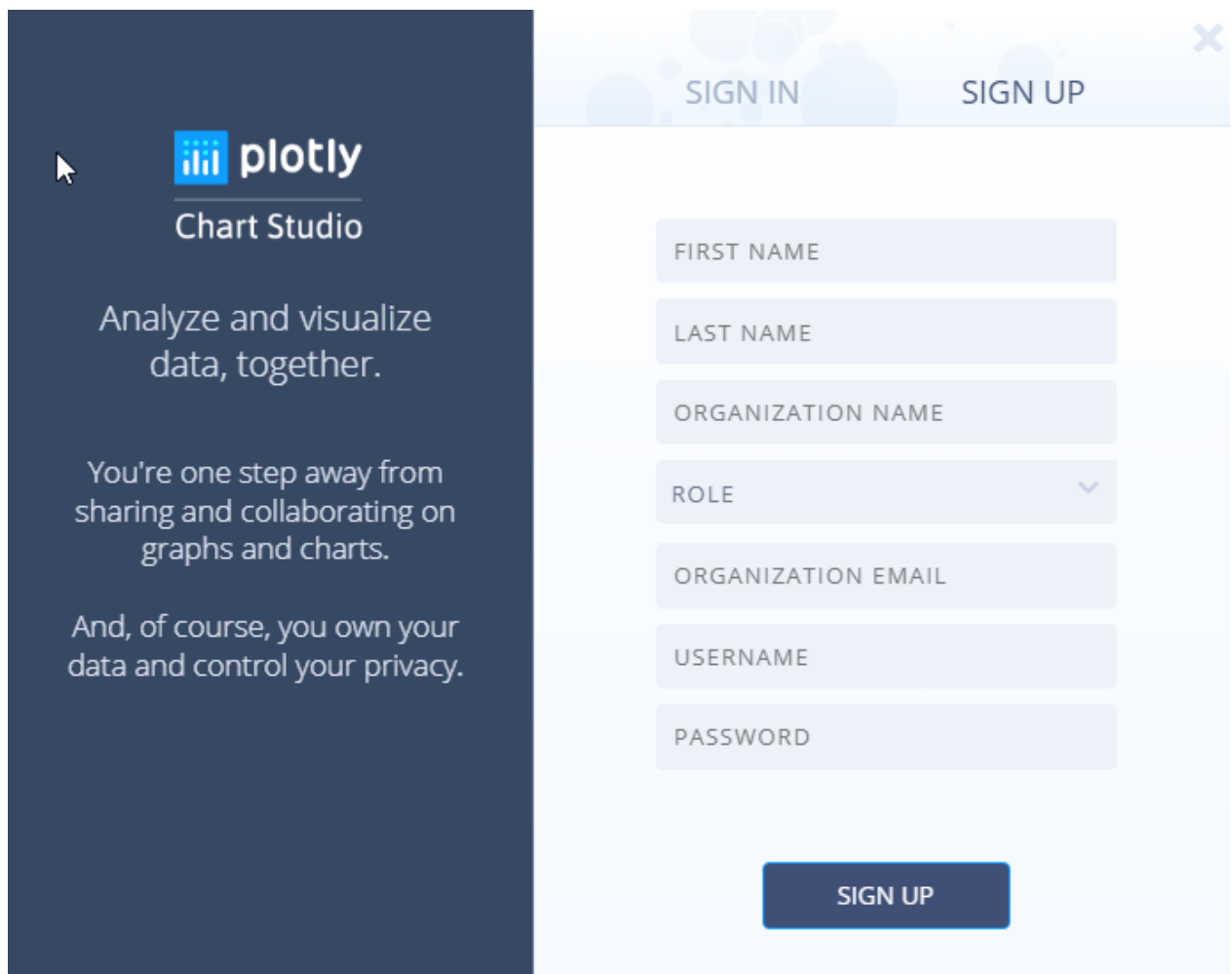


차트 스튜디오를 사용하기 위해서는 먼저 차트 스튜디오에 가입을 해야한다.



The image shows the Plotly Chart Studio sign-up interface. On the left, a dark blue sidebar contains the Plotly logo, the text 'Chart Studio', and a message: 'Analyze and visualize data, together. You're one step away from sharing and collaborating on graphs and charts. And, of course, you own your data and control your privacy.' On the right, a light blue sign-up form is displayed. At the top of the form are links for 'SIGN IN' and 'SIGN UP'. The form includes input fields for 'FIRST NAME', 'LAST NAME', 'ORGANIZATION NAME', 'ROLE' (a dropdown menu), 'ORGANIZATION EMAIL', 'USERNAME', and 'PASSWORD'. A dark blue 'SIGN UP' button is located at the bottom of the form.

plotly
Chart Studio

Analyze and visualize data, together.

You're one step away from sharing and collaborating on graphs and charts.

And, of course, you own your data and control your privacy.

SIGN IN SIGN UP

FIRST NAME

LAST NAME

ORGANIZATION NAME

ROLE

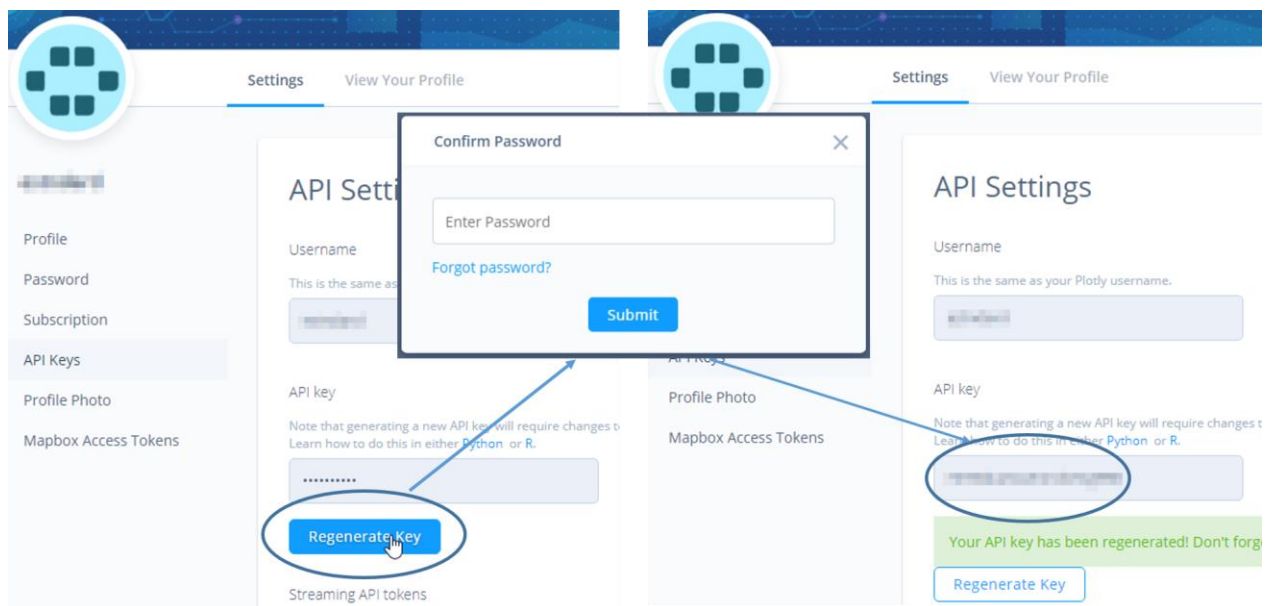
ORGANIZATION EMAIL

USERNAME

PASSWORD

SIGN UP

가입을 신청하면 사용자 인증을 위한 이메일이 발송되는데 이 이메일에 인증해주면 가입이 완료된다. 이 후 로그인하면 먼저 R-Studio 과 Jupyter Notebook 에서 업로드를 하기 위한 토큰을 받아야 한다. 로그인 후 'Setting' 메뉴의 'API Keys' 메뉴를 선택하고 패스워드를 입력하면 다음과 같은 키를 키 값이 나오는데 이 키 값을 잘 기록해 두어야 한다.



2.2.1. 환경 설정과 업로드

이 키 값을 R-Studio 와 Jupyter notebook 에서 다음과 같이 세팅하고 plotly 시각화 객체를 업로드할 수 있다.

- R

R 에서 다음과 같이 환경변수를 설정해주면 해당 R 세션내에서는 차트 스튜디오를 사용할 수 있다.

```
Sys.setenv("plotly_username"="your_plotly_username")
Sys.setenv("plotly_api_key"="your_api_key")
```

만약 R 세션이 열릴때 자동적으로 환경변수를 설정하기 위해서는 '.Rprofile' 파일에 환경변수를 넣어준다.

환경변수가 설정된 후에는 plotly 시각화를 만들고 `api_create()`를 사용하여 차트 스튜디오에 올려준다.

```
fig <- df_취업률_500 |>
  ## x 축은 졸업자수, y 축은 취업자수로 매핑한 plotly 객체 생성
  plot_ly() |>
  add_trace(type = 'scatter', mode = 'markers',
            x = ~졸업자수, y = ~취업자수)

api_create(fig, filename = "업로드할 파일 이름")
```

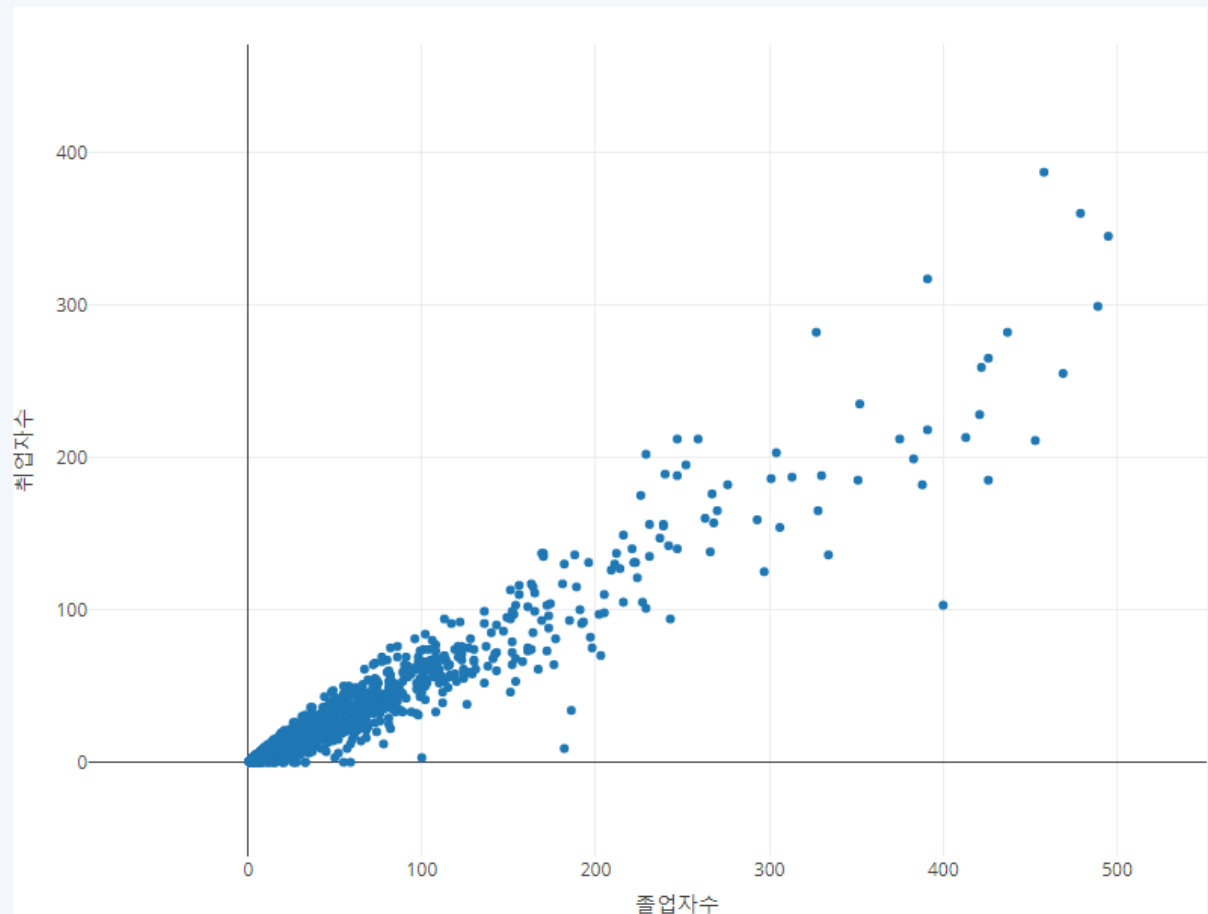
@estndard

Plot

Data

Python & R

Forking History



- python

python 에서 차트 스튜디오에 접속하기 위해서는 먼저 `pip` 를 사용하여 `chart_studio` 라이브러리를 설치하여야 한다.

```
> pip install chart_studio
```

라이브러리가 설치 되었다면 `chart_studio.tools.set_credentials_file()` 을 사용하여 차트 스튜디오의 ID 와 API Key 를 다음과 같이 설정한다.

```
import chart_studio
chart_studio.tools.set_credentials_file(username='your_plotly_username', api_key='your_api_key')
```

이 과정을 python 실행시 자동으로 설정하려면 python 홈 디렉터리에 '.plotly/credentials' 파일을 다음과 같이 만들어야 한다.

```
{  
  "username": "your_plotly_username",  
  "api_key": "your_api_key"  
}
```

이제 차트 스튜디오를 사용할 환경설정이 완료되었다면 업로드할 plotly 시각화를 생성하고 완성되면 `chart_studio.plotly` 라이브러리를 임포트하고 `plot()`이나 `ipplot()`을 사용하여 plotly 시각화 객체를 차트 스튜디오에 업로드할 수 있다. 다음의 코드를 실행시키면 업로드된 결과가 웹 브라우저를 통해 자동으로 실행되고 실행 결과로 해당 URL 이 나타난다. `ipplot()`을 사용하면 업로드된 결과가 웹 브라우저를 통해 실행되고 jupyter notebook 에도 동일한 결과가 나타난다.

```
import chart_studio.plotly as py  
py.plot(fig, filename = '업로드할 파일 이름', auto_open=True)
```

2.2.2. 블로그에 임베딩

앞과 같이 R 과 python 에서 만든 그래프가 차트 스튜디오에 올라가면 이 시각화를 임베딩하기 위한 주소를 추출해야 한다. 주소의 추출은 업로딩된 그래프의 바로 아래 버튼을 클릭하면 다음과 같이 나타난다.

여기에 나타난 임베딩 태그를 복사하여 포스팅에 붙여넣으면 해당 위치에 plotly 시각화가 나타난다.



3. 정적 시각화(ggplot, matplotlib)의 plotly 변환

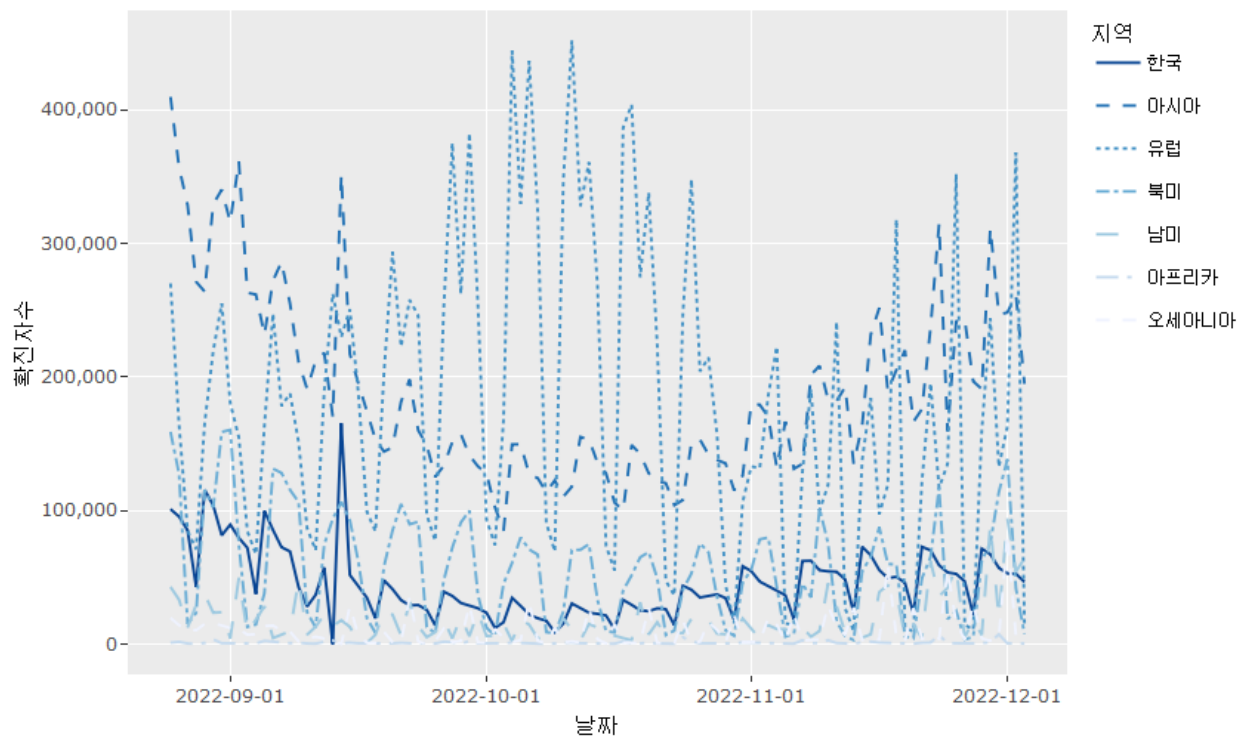
기존의 데이터 분석에서 시각화를 위해 많이 사용하는 패키지가 R 은 **ggplot2** 일 것이고 python 사용자는 **matplotlib** 이나 **seaborn** 을 사용한다. plotly 는 이들 정적 시각화를 인터랙티브 데이터 시각화로 변환하는 기능도 제공하기 때문에 기존의 시각화도 재활용할 수 있다.

- R

R 의 경우는 plotly 패키지에서 제공하는 **ggplotly()** 를 사용하면 간단히 변환된다. **ggplot2** 로 생성된 객체를 **ggplotly()** 에 매개변수로 전달해 주면 plotly 객체로 변환된다.

```
ggplotly <- df_covid19_100 |>
  ggplot(aes(x = date, y = new_cases, color = location )) +
  geom_line(aes(group = location, linetype = location)) +
  scale_x_date(breaks = '1 months') +
  scale_y_continuous(labels = scales::comma) +
  labs(x = '날짜', y = '확진자수', linetype = '지역', color = '지역')

## ggplot 객체를 plotly 객체로 변환
ggplotly(ggplotly)
```



실행결과 IV - 1 R의 ggplot2 객체의 전환

- python

python의 경우는 `plotly.tools` 패키지의 `mpl_to_plotly()`를 사용하여 `matplotlib`과 `seaborn` 패키지로 생성된 정적 시각화를 plotly의 동적 시각화로 변환할 수 있다.

```
import seaborn as sns
import plotly.tools as tls

sns.lineplot(x="date", y="new_cases",
             hue="location",
             data=df_covid19_100)

mpl_fig = plt.gcf()
plotly_fig = tls.mpl_to_plotly(mpl_fig)
plotly_fig.show()
```

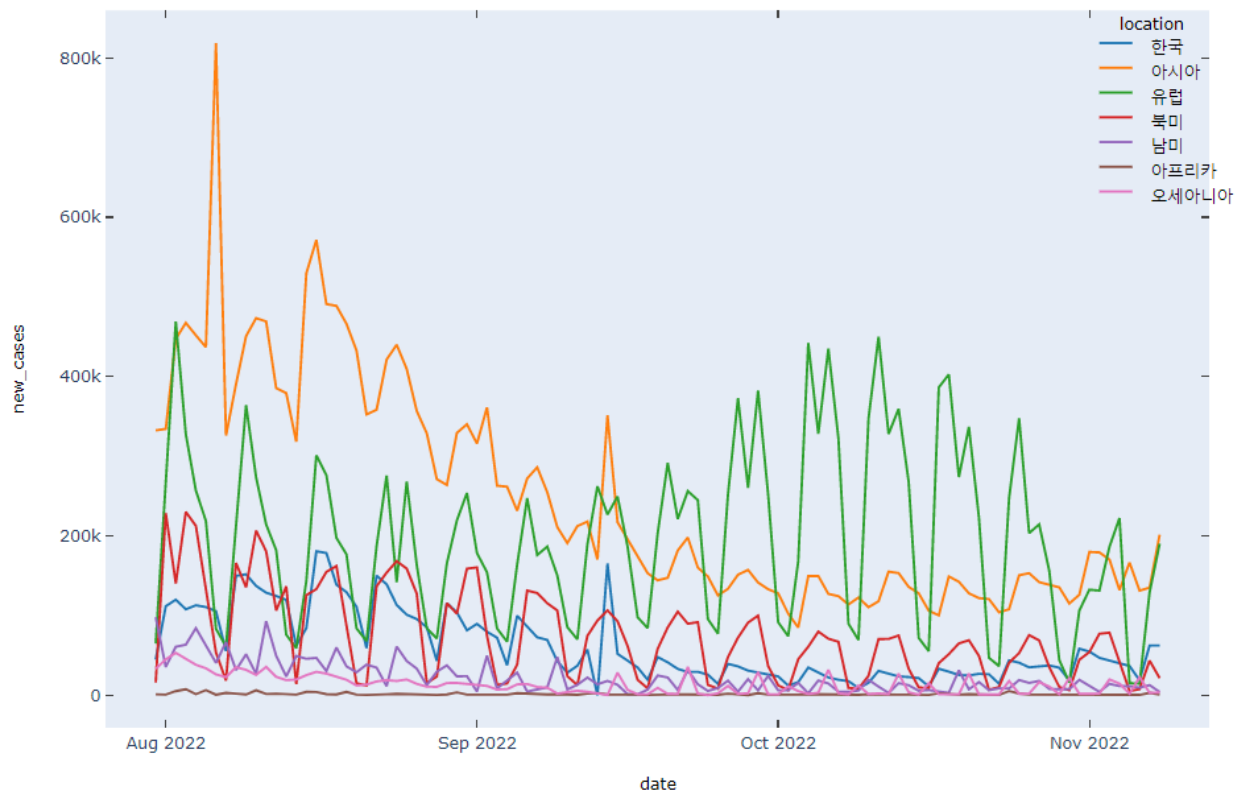



그림 II-30p. plotly 의/seaborn 객체의 전환