Newbies as a Data Scientist in East Asia!

2021 Kaggle Machine Learning & Data Science Survey

Team: Green_YHJW

김지원 박윤화

Team Member

YH

박윤화

JW

김지원

Table of Contents

01 분석 배경 및 목적

1.1 Introduction : Kaggle 정의
1.2 Introduction : Kaggle 대회 개요
1.3 Introduction : Data Science
1.4 Introduction : 분석 배경과 목적
1.5 캐글 제출

PART 1

1. Introduction : Kaggle 정의

kaggle

AI / 디지털 트랜스포메이션 / 로봇/자동화 / 머신러닝/딥러닝 / 비즈니스|경제 2017.03.20

구글의 캐글(Kaggle) 인수, AI 분야에 미칠 영향은?

Clint Boulton | CIO

AI 기술 경쟁에 뛰어 들었다면, 수많은 데이터 과학자가 참여하는 커뮤니티에 주목할 만한 이유가 충분하다.

구글은 지난주 공개되지 않은 금액에 크라우드소싱 플랫폼인 캐글(Kaggle)을 인수하면서 유리한 고지를 점령했다. 오늘날 캐글을 이용하는 데이터 전문가는 60만 명에 달한다. 이들은 암 발견과 심장병 진단 등 도전 과제를 극복하는 예측 모델을 구축하기 위해 이 커뮤니티에 참여하고 있다. 업계 전문가들은 구글이 캐글 인수를 통해 AI기술을 더 광범위하게 보급시키려 시도할 것으로 내다보고 있다.

싱귤래러티 대학(Singularity University)에서 인공지능(AI)과 로봇을 책임지고 있는 닐 제이콥 스타인은 "전세계 곳곳에서 데이터와 머신러닝 연구가 진행되고 있다. 캐글 인수는 똑똑한 사람들 중 상당수가 이미 다른 누군가를 위해 일하고 있다는 점을 구글이 인식하고 있음을 보여준다. AI 기술의 발전과 확산 관점에서 아주 긍정적인 움직임이다"라고 말했다.

구글 클라우드에서 AI 및 머신러닝 부문 최고 과학자(Chief Scientis)를 담당하고 있는 페이페이 리는 지난주 구글 클라우드 넥스트(Google Cloud Next) 이벤트에서 캐글팀이 오픈소스 머신러닝 소프트웨어인 텐소플로우(TensorFlow)용 개발자 라이브러리 등 구글의 다른 AI자산과 통합될 것이라고 밝혔다.

캐글(Kaggle)

2010년 설립된 예측모델 및 분석 대회 플랫폼

대회개요

기업 및 단체에서 데이터와 해결과제를 등록하면, 데이터 과학자들이 이를 해결하는 모델을 개발하고 경쟁한다.

2. Introduction : Kaggle 대회 개요

2021 Kaggle Machine Learning & Data Science Survey

2017년부터 이어져온 대회로,
21/09/01부터 21/10/04동안 Kaggler들을 대상으로 진행한 설문조사의 결과를
데이터로 분석하여 각자의 창의적인 'Story'를 진행하고 데이터를 시각화하는 대회이다.

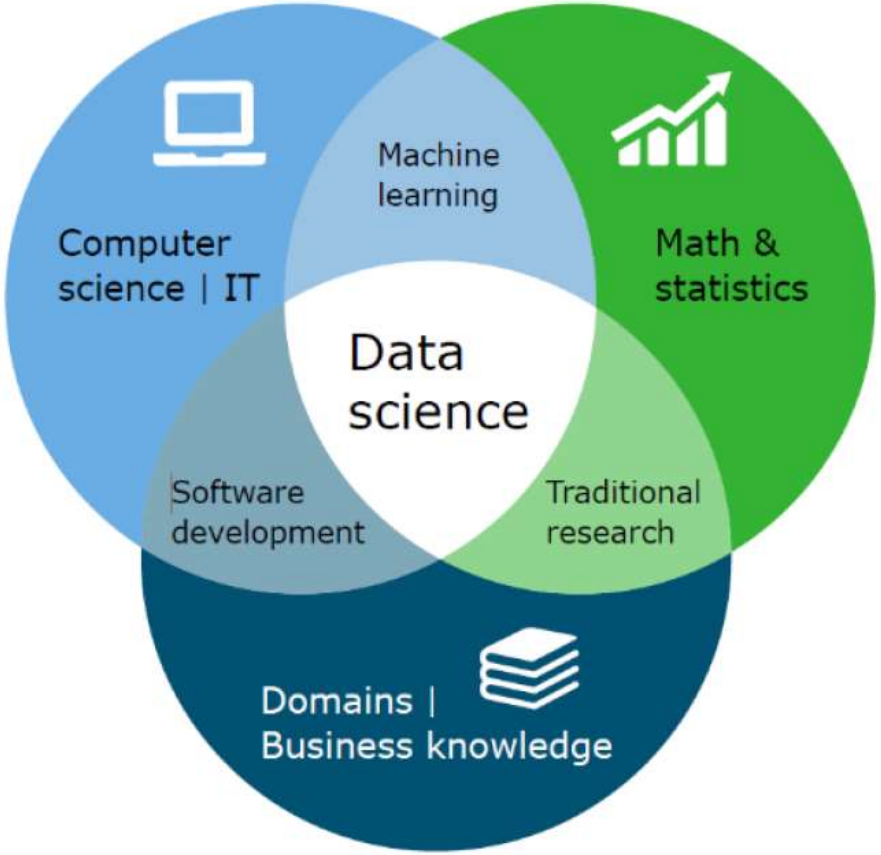
3. Introduction : Data Science

- Data Science

데이터 사이언스는 통계, 과학적 방법, 인공지능(AI) 및 데이터 분석을 포함한 여러 분야를 결합하여 데이터에서 가치를 추출한다.

- Data Scientist

데이터 과학자는 현장에 존재하는 대량의 데이터를 모으고, 분석에 적합한 형태로 가공하고, 데이터가 의미하는 바를 이 야기(story)에 담아 다른 사람에게 효과적으로 전달하는 역할을 한다.



PART 1

4. Introduction : 분석 배경과 목적

분석 배경

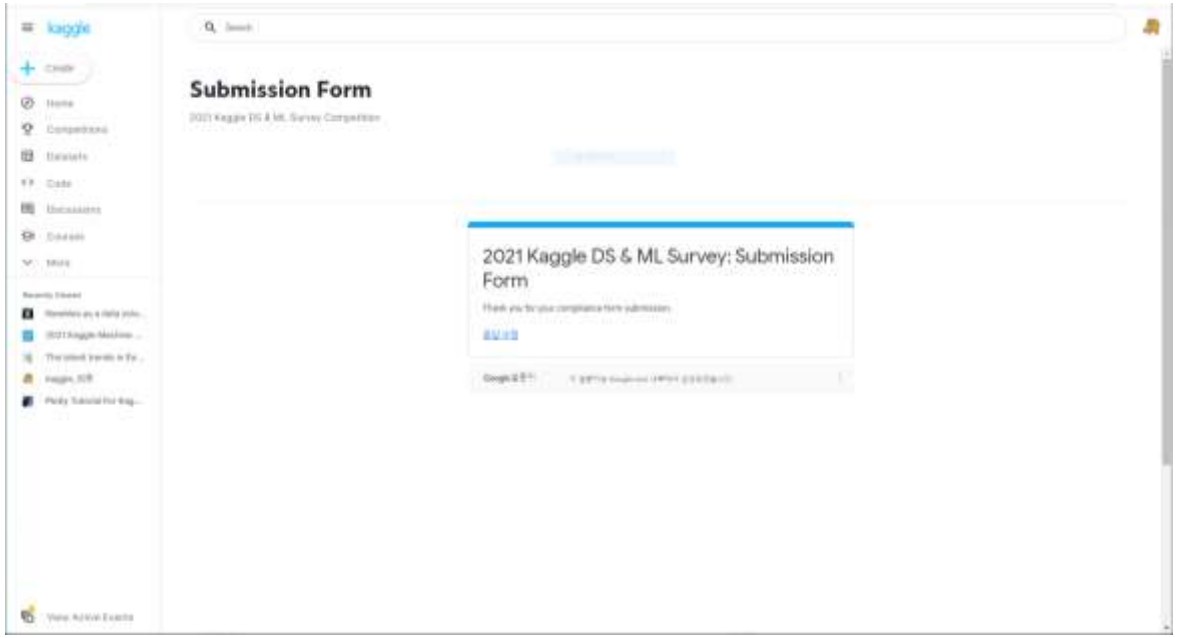
현재 동아시아의 Data Scientist의 꿈나무의 일원으로서 무엇을 준비해야할까?
빅데이터 분야에서 변화하는 트렌드를 알아보자

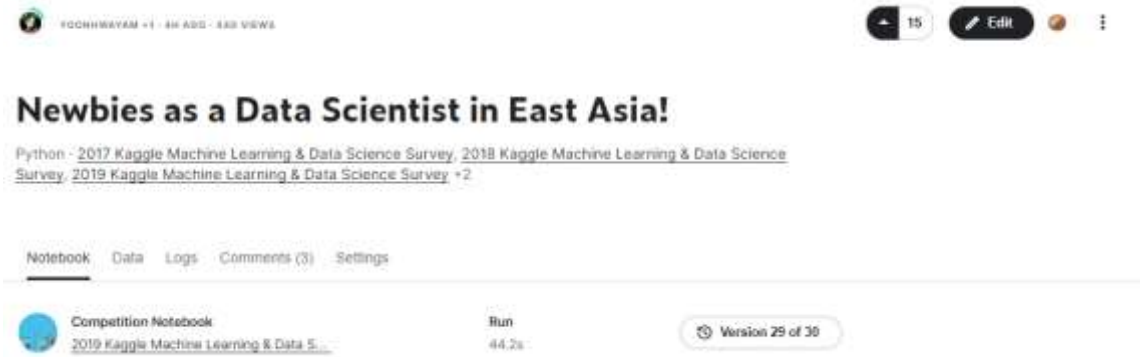
분석 목적

Data Scientist의 직무 분석, 연봉, 사용한 언어에 대해
연도별로 어떻게 증가하는지, 트렌드는 무엇인지 분석하고
분석한 자료를 토대로 데이터를 시각화 해보자

PART 1

5. 캐글 제출


Submission Form
2021 Kaggle DS & ML Survey: Submission Form


Newbies as a Data Scientist in East Asia!
Python · 2017 Kaggle Machine Learning & Data Science Survey, 2018 Kaggle Machine Learning & Data Science Survey, 2019 Kaggle Machine Learning & Data Science Survey +2
Notebook
Data
Logs
Comments (3)
Settings
Competition Notebook
2019 Kaggle Machine Learning & Data S...
Run
44.2s
Version 29 of 30
Edit

02 데이터 분석

2.1 사용한 언어와 Library

2.2 Plotly

2.3. Kaggle Survey in 2021 Data Set

2.3 Data Import &

2.4 Data Preprocessing

PART 2

1. 사용한 언어와 Library

Python

: 파이썬은 플랫폼에 독립적이며 인터프리터식, 객체 지향적, 동적 타이핑이 가능한 대화형 프로그래밍 언어이다.

Library

Numpy

: 행렬이나 일반적으로 대규모 다차원 배열을 쉽게 처리 할 수 있도록 지원하는 파이썬의 라이브러리

Metplotlib

: 다양한 데이터를 그래프로 만들어주는 파이썬 라이브러리

Seaborn

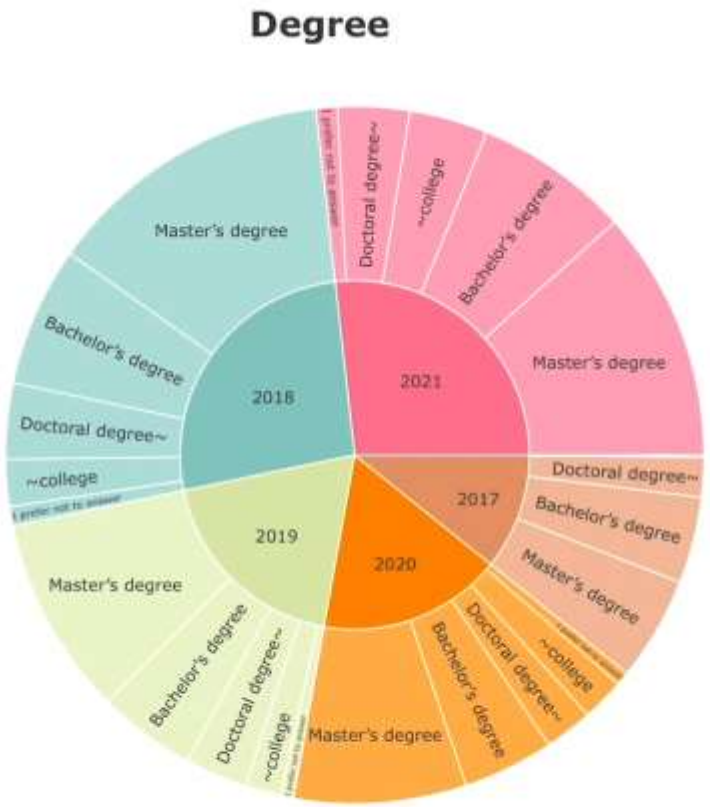
: Matplotlib을 기반으로 다양한 색상 테마와 통계용 차트 등의 기능을 추가한 시각화 패키지이며, Matplotlib 보다 더 세세한 작업이 가능하다.

Plotly (아래는 여러가지 모듈)

- plotly.express 그래프를 쉽고 빠르게 그릴 수 있는 인터페이스
- plotly.graph_objects express보다 더 상세한 작업이 가능해 자신이 원하는 방식으로 커스터마이징이 가능
- plotly.figure_factory express가 존재하기 전에 사용했음 이전 버전과의 호환성을 위해 모듈에 남음
- plotly.subplots 여러 개의 그래프를 하나의 그림에 나타내도록 하는 모듈
- plotly.offline 로컬에 저장되며 웹 브라우저에서 열리는 HTML을 만들고 독립 실행 형태로 만듬

PART 2

2. Plotly

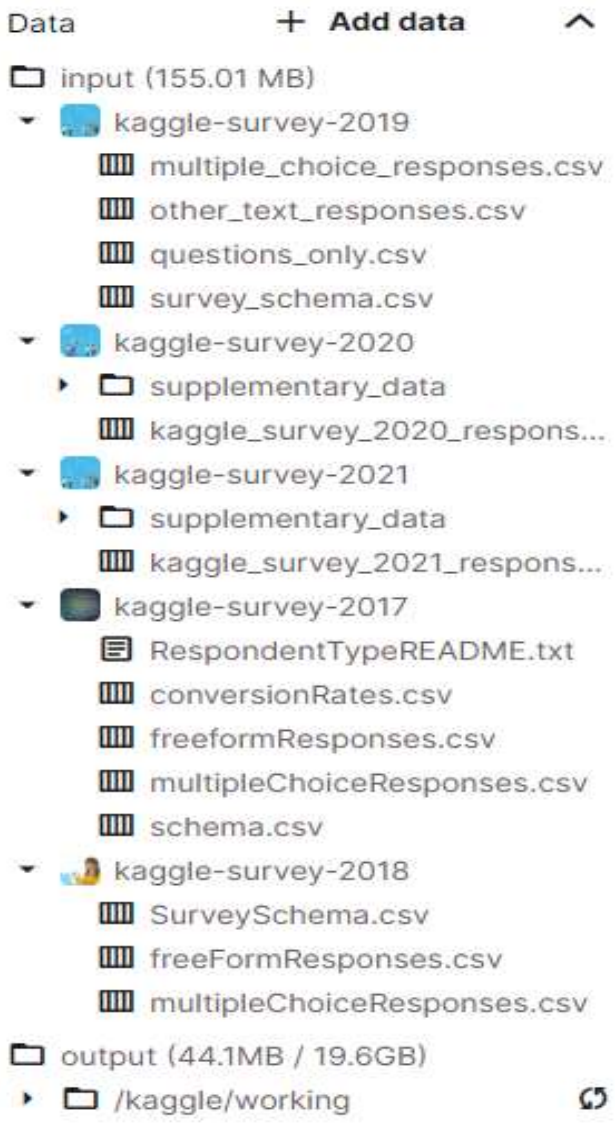


Plotly의 장점

Matplotlib는 정적인 그래프를 그리지만,
Plotly는 동적으로 클릭해서 움직일 수 있고, 줌 아웃, 줌 인, 그래프 다운로드를 지원

PART 2

3. Data Import



Data Import

- ✓ Data 의 종류 (2021~2017)
  - ✓ 2017년부터 2021년까지 사용할 데이터 5개의 Data를 Loading 하였다.
  - ✓ 아래는 Kaggle notebook에서 데이터를 Loading

```
df17= pd.read_csv("/kaggle/input/kaggle-survey-2017/multipleChoiceResponses.csv", encoding="ISO-8859-1")
df18= pd.read_csv("/kaggle/input/kaggle-survey-2018/multipleChoiceResponses.csv", )
df19= pd.read_csv("/kaggle/input/kaggle-survey-2019/multiple_choice_responses.csv", )
df20= pd.read_csv("/kaggle/input/kaggle-survey-2020/kaggle_survey_2020_responses.csv", )
df21= pd.read_csv("/kaggle/input/kaggle-survey-2021/kaggle_survey_2021_responses.csv", )
```

PART 2 4. Data Preprocessing

1. East Asia와 World
2. Gender
3. Job
4. Age
5. Degree
6. Experience
7. Salary
8. Language

PART 2

4. Data Preprocessing : 1. East Asia와 World

East Asia

동아시아에는 몽골, 러시아 극동, 중국, 북한, 남한, 일본, 베트남, 대만 등의 나라가 있지만

Kaggle 설문조사를 참여한 국가는 중국, 일본, 남한, 대만의 총 4개국이 있다.
2018년도에 대만은 포함되지 않았다.

https://namu.wiki/w/%EB%8F%99%EC%95%84%EC%8B%9C%EC%95%84

```
df21.head()
```

Time from Start to Finish (seconds)	Q1	Q2	Q3	Q4	Q5	Q6	Q7_Part_1	Q7_Part_2	Q7_Part_3	...	Q38_B_Part_5	Q38_B_Part_6	Q38_B_Part_7	Q38_B_Part_8	Q38_B_Part_9	Q38_B_Part_10	Q38_B_Part_11	Q38_B_OTHER	region	year
1	910	22-24	Man	India	Bachelor's degree	Other	5-10 years	Python	4	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	World	2021
2	784	50-54	Man	Indonesia	Master's degree	Program/Project Manager	20+ years	NaN	NaN	SQL	...	NaN	NaN	NaN	NaN	NaN	NaN	None	NaN	World	2021
3	924	22-24	Man	Pakistan	Master's degree	Software Engineer	1-3 years	Python	NaN	NaN	...	TensorBoard	NaN	NaN	NaN	NaN	NaN	NaN	NaN	World	2021
4	575	45-49	Man	Mexico	Doctoral degree	Research Scientist	20+ years	Python	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	None	NaN	World	2021
5	781	45-49	Man	India	Doctoral degree	Other	< 1 years	Python	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	World	2021

```
df21_Ea.head()
```

Time from Start to Finish (seconds)	Q1	Q2	Q3	Q4	Q5	Q6	Q7_Part_1	Q7_Part_2	Q7_Part_3	...	Q38_B_Part_5	Q38_B_Part_6	Q38_B_Part_7	Q38_B_Part_8	Q38_B_Part_9	Q38_B_Part_10	Q38_B_Part_11	Q38_B_OTHER	region	year
19	249	22-24	Man	Japan	Master's degree	Software Engineer	3-5 years	Python	NaN	SQL	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	EastAsia	2021
28	807	22-24	Man	China	Master's degree	Student	3-5 years	Python	NaN	NaN	...	TensorBoard	NaN	NaN	NaN	NaN	NaN	NaN	NaN	EastAsia	2021
29	820	22-24	Man	Japan	No formal education past high school	Software Engineer	1-3 years	Python	NaN	SQL	...	NaN	NaN	NaN	NaN	NaN	NaN	None	NaN	EastAsia	2021
47	1024	22-24	Man	China	Some college/university study without earning ...	Data Analyst	< 1 years	Python	NaN	SQL	...	TensorBoard	NaN	NaN	NaN	NaN	NaN	NaN	NaN	EastAsia	2021
61	382	18-21	Man	Japan	Some college/university study without earning ...	Student	< 1 years	NaN	NaN	NaN	...	TensorBoard	NaN	NaN	NaN	NaN	NaN	NaN	NaN	EastAsia	2021

```
df21_Wo.head()
```

Time from Start to Finish (seconds)	Q1	Q2	Q3	Q4	Q5	Q6	Q7_Part_1	Q7_Part_2	Q7_Part_3	...	Q38_B_Part_5	Q38_B_Part_6	Q38_B_Part_7	Q38_B_Part_8	Q38_B_Part_9	Q38_B_Part_10	Q38_B_Part_11	Q38_B_OTHER	region	year
2	784	50-54	Man	Indonesia	Master's degree	Program/Project Manager	20+ years	NaN	NaN	SQL	...	NaN	NaN	NaN	NaN	NaN	NaN	None	NaN	World	2021
3	924	22-24	Man	Pakistan	Master's degree	Software Engineer	1-3 years	Python	NaN	NaN	...	TensorBoard	NaN	NaN	NaN	NaN	NaN	NaN	NaN	World	2021
4	575	45-49	Man	Mexico	Doctoral degree	Research Scientist	20+ years	Python	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	None	NaN	World	2021
5	781	45-49	Man	India	Doctoral degree	Other	< 1 years	Python	NaN	NaN	...	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	World	2021
6	1020	25-29	Woman	India	I prefer not to answer	Currently not employed	< 1 years	Python	NaN	NaN	...	NaN	Scikit-...	NaN	NaN	NaN	NaN	NaN	NaN	World	2021

데이터 프레임을 World와 East Aisa로 나눔
⇒ df21_Wo, df21_EA
17년부터 21년까지 연도별로 각각 나누어줌
⇒ df20_Wo, df20_EA

PART 2 **4. Data Preprocessing : 2. Gender**

Data Frame 에서 필요한 값을 Colums으로 묶어서 테이블 생성함

```
Gender_21 = (
    df21['Q2']
    .replace(['Prefer not to say', 'Prefer to self-describe', 'Nonbinary'], 'Others')
    .replace(['Man', 'Woman'], ['Male', 'Female'])
    .fillna('Others')
    .value_counts()
    .to_frame()
    .reset_index()
    .rename(columns={'index':'type', 'Q2':'Gender'})
    .groupby('type')
    .sum()
    .reset_index())
```

Gender_21

	type	Gender
0	Female	4890
1	Male	20598
2	Others	485

```
#data preprocessing
gender21= df21_Ea.loc[:, ['Q3', 'Q2', 'year']].rename(columns={'Q3':'Country', 'Q2':'Gender'})
gender20= df20_Ea.loc[:, ['Q3', 'Q2', 'year']].rename(columns={'Q3':'Country', 'Q2':'Gender'})
gender19= df19_Ea.loc[:, ['Q3', 'Q2', 'year']].rename(columns={'Q3':'Country', 'Q2':'Gender'})
gender18= df18_Ea.loc[:, ['Q3', 'Q1', 'year']].rename(columns={'Q3':'Country', 'Q1':'Gender'})
gender17= df17_Ea.loc[:, ['Country', 'GenderSelect', 'year']].rename(columns={'index':'type', 'GenderSelect':'Gender'})

Gender5y= pd.concat([gender17, gender18, gender19, gender20, gender21])
Gender5y= (Gender5y.replace(['Prefer not to say', 'Prefer to self-describe', 'Nonbinary', 'A different identity'], 'Others')
           .replace(['Man', 'Woman'], ['Male', 'Female'])
           .groupby(['year', 'Gender'])
           .size()
           .reset_index()
           .rename(columns = {0:"Count"}))

gen17_5y = Gender5y[Gender5y['year'] == "2017"].reset_index(drop = True)
gen18_5y = Gender5y[Gender5y['year'] == "2018"].reset_index(drop = True)
gen19_5y = Gender5y[Gender5y['year'] == "2019"].reset_index(drop = True)
gen20_5y = Gender5y[Gender5y['year'] == "2020"].reset_index(drop = True)
gen21_5y = Gender5y[Gender5y['year'] == "2021"].reset_index(drop = True)

Gen5y_ = pd.concat([gen17_5y, gen18_5y, gen19_5y, gen20_5y, gen21_5y], ignore_index = True)
Gen5y_= pd.pivot(Gen5y_, index = "year", columns = "Gender", values = "Count").reset_index()
Gen5y_
```

Gen5y_

Gender	year	Female	Male	Others
0	2017	183	1004	8
1	2018	341	2110	49
2	2019	227	1537	39
3	2020	214	1401	30
4	2021	327	2037	64

처음 East Asia로 분류해놓은 df21_Ea DataFrame을 이용해,

East Asia 내에서 연도별로 성별 카운트 값을 한 테이블에 묶어서 보여줌

PART 2

4. Data Preprocessing : 3. Job


```
df21_job =df21job.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df20_job =df20job.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df19_job =df19job.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df18_job =df18job.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df17_job =df17job.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})

merge11=pd.merge(df21_job,df20_job, how='outer',on='JOB')
merge21=pd.merge(df19_job,df18_job, how='outer',on='JOB')
merge31=pd.merge(merge11,merge21, how='outer',on='JOB')
merge_Wo=(pd.merge(merge31,df17_job, how='outer',on='JOB')
            .rename(columns = {'Count_x_x':'2021','Count_y_x':'2020','Count_x_y':'2019','Count_y_y':'2018','Count':'2017'}).fillna(0)
            .reindex(columns = ['JOB','2017','2018','2019','2020','2021' ]))

df21job_Ea = df21job[df21job['region'] == 'EastAsia'].loc[:,['region','JOB']].rename(columns={'region':'EastAsia'})
df20job_Ea = df20job[df20job['region'] == 'EastAsia'].loc[:,['region','JOB']].rename(columns={'region':'EastAsia'})
df19job_Ea = df19job[df19job['region'] == 'EastAsia'].loc[:,['region','JOB']].rename(columns={'region':'EastAsia'})
df18job_Ea = df18job[df18job['region'] == 'EastAsia'].loc[:,['region','JOB']].rename(columns={'region':'EastAsia'})
df17job_Ea = df17job[df17job['region'] == 'EastAsia'].loc[:,['region','JOB']].rename(columns={'region':'EastAsia'})

df21job_Ea =df21job_Ea.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df20job_Ea =df20job_Ea.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df19job_Ea =df19job_Ea.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df18job_Ea =df18job_Ea.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})
df17job_Ea =df17job_Ea.groupby(['JOB']).size().reset_index().rename(columns = {0:"Count"})

merge1=pd.merge(df21job_Ea,df20job_Ea, how='outer',on='JOB')
merge2=pd.merge(df19job_Ea,df18job_Ea, how='outer',on='JOB')
merge3=pd.merge(merge1,merge2, how='outer',on='JOB')
merge=(pd.merge(merge3,df17job_Ea, how='outer',on='JOB')
        .rename(columns = {'Count_x_x':'2021','Count_y_x':'2020','Count_x_y':'2019','Count_y_y':'2018','Count':'2017'}).fillna(0)
        .reindex(columns = ['JOB','2017','2018','2019','2020','2021' ]))
```

```
#data preprocessing
df21job= df21.loc[:,['region','Q5']].rename(columns={'Q5':'2021'}).fillna('Others')
df20job= df20.loc[:,['region','Q5']].rename(columns={'Q5':'2020'}).fillna('Others')
df19job= df19.loc[:,['region','Q5']].rename(columns={'Q5':'2019'}).fillna('Others')
df18job= df18.loc[:,['region','Q6']].rename(columns={ 'Q6':'2018'}).fillna('Others')
df17job= df17.loc[:,['region','CurrentJobTitleSelect']].rename(columns={'CurrentJobTitleSelect':'2017'}).fillna('Others')

df21job['JOB']=["Data Analyst" if x in Data_Analyst
                else "Data Scientist" if x in Data_Scientist # Data Scientist
                else "Developer" if x in Developer
                 else "NotEmployed" if x in Not_Employed
                else "Others"
                for x in df21job['2021']]
```

```
Data_Analyst =['Data Analyst','Data Miner,Information technology','Data Miner',
               'Predictive Modeler','Information technology, networking, or system administration',
               'A business discipline (accounting, economics, finance, etc.)', 'Business Analyst', 'Humanities',
               'Statistician', 'Mathematics or statistics', 'Medical or life sciences (biology, chemistry, medicine, etc.)',
               'Physics or astronomy', 'Social sciences (anthropology, psychology, sociology, etc.)', 'Environmental science or geology',
               'Humanities (history, literature, philosophy, etc.)']
Data_Scientist =['Data Scientist', 'Research Scientist', 'Researcher',
                 'Machine Learning Engineer', 'Scientist/Researcher']
Developer=['Developer Relations/Advocacy','Data Engineer','Engineer','Engineering (non-computer focused)',
           'Programmer','Software Engineer','Computer Scientist','Computer science (software engineering, etc.)',
           'Fine arts or performing arts','Product Manager','Software Developer/Software Engineer',
           'Product/Project Manager','Program/Project Manager','DBA/Database Engineer']
Not_Employed =['Currently not employed', 'Not employed', 'Student']
Others = ['I never declared a major', 'Other']
```

merge

	JOB	2017	2018	2019	2020	2021
0	Data Analyst	146.0	297	180	165	309
1	Data Scientist	296.0	291	411	367	555
2	Developer	249.0	502	492	310	535
3	NotEmployed	0.0	938	490	538	855
4	Others	505.0	472	230	265	174

총 다섯개로 직업을 분류
: Data_Analyst, Data_Scientist, Developer, Not_Employed, Others

연도별로 직업을 분류하고, Dataframe을 합침

PART 2 4. Data Preprocessing : 4. Age

World와 East Asia를 각각 전처리 (오른쪽은 World)

-Age21_W
연도별로 age, 국가, year의 데이터 프레임을 만듬

-Age5y_W
연도별 데이터를 하나의 테이블로 합쳐줌

-Age21_percent_W
단일연도 데이터만 뽑아냄
Count 값에서 Percent 값으로 바꿔줌

-Age5y_percent_W
연도별로 그룹화

Age5y

	year	age	Count
0	2018	18-21	294
1	2018	22-29	1492
2	2018	30-39	454
3	2018	40-49	193
4	2018	50-59	38
5	2018	60+	29
6	2019	18-21	153
7	2019	22-29	823
8	2019	30-39	449
9	2019	40-49	255
10	2019	50-59	97
11	2019	60+	26
12	2020	18-21	205
13	2020	22-29	690
14	2020	30-39	361
15	2020	40-49	232
16	2020	50-59	116
17	2020	60+	41
18	2021	18-21	319
19	2021	22-29	1004
20	2021	30-39	538
21	2021	40-49	317
22	2021	50-59	180
23	2021	60+	70

```
Age21_W = df21.loc[:,['Q3','Q1', 'year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q1':'age'}).fillna('etc')
Age20_W = df20.loc[:,['Q3','Q1','year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q1':'age'}).fillna('etc')
Age19_W = df19.loc[:,['Q3','Q1','year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q1':'age'}).fillna('etc')
Age18_W = df18.loc[:,['Q3','Q2','year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q2':'age'}).fillna('etc')

Age5y_W= pd.concat([Age21_W, Age20_W, Age19_W, Age18_W])
Age5y_W= (Age5y_W.replace(['60-69', '70+', '70-79', '80+'], '60+')
           .replace(['22-24', '25-29'], '22-29')
           .replace(['30-34', '35-39'], '30-39')
            .replace(['40-44', '45-49'], '40-49')
         .replace(['50-54', '55-59'], '50-59')
           .groupby(['year', 'age'])
           .size()
           .reset_index()
           .rename(columns = {0:"Count"}))

Age21_percent_W = Age5y_W[Age5y_W['year'] == "2021"].reset_index(drop = True)
Age21_percent_W['percentage'] = Age21_percent_W["Count"] / Age21_percent_W["Count"].sum()
Age21_percent_W['%'] = np.round(Age21_percent_W['percentage'] * 100, 1)

Age20_percent_W = Age5y_W[Age5y_W['year'] == "2020"].reset_index(drop = True)
Age20_percent_W['percentage'] = Age20_percent_W["Count"] / Age20_percent_W["Count"].sum()
Age20_percent_W['%'] = np.round(Age20_percent_W['percentage'] * 100, 1)

Age19_percent_W = Age5y_W[Age5y_W['year'] == "2019"].reset_index(drop = True)
Age19_percent_W['percentage'] = Age19_percent_W["Count"] / Age19_percent_W["Count"].sum()
Age19_percent_W['%'] = np.round(Age19_percent_W['percentage'] * 100, 1)

Age18_percent_W = Age5y_W[Age5y_W['year'] == "2018"].reset_index(drop = True)
Age18_percent_W['percentage'] = Age18_percent_W["Count"] / Age18_percent_W["Count"].sum()
Age18_percent_W['%'] = np.round(Age18_percent_W['percentage'] * 100, 1)

Age5y_percent_W = pd.concat([Age18_percent_W, Age19_percent_W, Age20_percent_W, Age21_percent_W], ignore_index = True)
Age5y_percent_W= pd.pivot(Age5y_percent_W, index = "year", columns = 'age', values = "%").reset_index()
Age5y_percent_W

Age21 = df21_Ea.loc[:,['Q3','Q1', 'year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q1':'age'}).fillna('etc')
Age20 = df20_Ea.loc[:,['Q3','Q1','year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q1':'age'}).fillna('etc')
Age19 = df19_Ea.loc[:,['Q3','Q1','year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q1':'age'}).fillna('etc')
Age18 = df18_Ea.loc[:,['Q3','Q2','year']].reset_index().rename(columns={'Q3':'East_Asia', 'Q2':'age'}).fillna('etc')

Age5y= pd.concat([Age21, Age20, Age19, Age18])
Age5y= (Age5y.replace(['60-69', '70+', '70-79', '80+'], '60+')
           .replace(['22-24', '25-29'], '22-29')
           .replace(['30-34', '35-39'], '30-39')
            .replace(['40-44', '45-49'], '40-49')
         .replace(['50-54', '55-59'], '50-59')
           .groupby(['year', 'age'])
           .size()
           .reset_index()
           .rename(columns = {0:"Count"}))
```

PART 2

4. Data Preprocessing : 5. Degree


```
#data preprocessing
df21_Ea_degree=(df21_Ea['Q4'].replace(['No formal education past high school', 'Some college/university study without earning a bachelor's degree'],'~college')
                             .replace(['Doctoral degree','Professional doctorate'],'Doctoral degree~')
                             .value_counts().to_frame().rename(columns={'Q4':'2021'}))
df20_Ea_degree=(df20_Ea['Q4'].replace(['No formal education past high school', 'Some college/university study without earning a bachelor's degree'],'~college')
                             .replace(['Doctoral degree', 'Professional degree'],'Doctoral degree~')
                             .value_counts().to_frame().rename(columns={'Q4':'2020'}))
df19_Ea_degree=(df19_Ea['Q4'].replace(['No formal education past high school','Some college/university study without earning a bachelor's degree'],'~college')
                             .replace(['Doctoral degree', 'Professional degree'],'Doctoral degree~')
                             .value_counts().to_frame().rename(columns={'Q4':'2019'}))
df18_Ea_degree=(df18_Ea['Q4'].replace(['No formal education past high school', 'Some college/university study without earning a bachelor's degree'],'~college')
                             .replace(['Doctoral degree', 'Professional degree'],'Doctoral degree~')
                             .value_counts().to_frame().rename(columns={'Q4':'2018'}))
df17_Ea_degree=(df17_Ea['FormalEducation']
                .replace(['No formal education past high school', 'Some college/university study without earning a bachelor's degree'],'~college')
                .replace(['Doctoral degree', 'Professional degree'],'Doctoral degree~')
                .value_counts().to_frame()
                .rename(columns={'FormalEducation':'2017'} ,index = {'I did not complete any formal education past high school':'No formal education past high school'

concat1 = pd.concat([df21_Ea_degree,df20_Ea_degree],axis=1, join='outer')
concat2 = pd.concat([df19_Ea_degree,df18_Ea_degree],axis=1, join='outer')
concat3 = pd.concat([concat1,concat2],axis=1, join='outer')
df21_Ea_degree_yearly_=concat3.join(df17_Ea_degree).fillna(0).transpose() #.transpose() 행 열 바꾸기

df21_Ea_degree_yearly=df21_Ea_degree_yearly_.stack().to_frame().reset_index().rename(columns={'level_0':'year','level_1':'degree',0:'value'})
df21_Ea_degree_yearly
```

df21_Ea_degree_yearly

	year	degree	value
0	2021	Master's degree	1056.0
1	2021	Bachelor's degree	651.0
2	2021	~college	330.0
3	2021	Doctoral degree~	301.0
4	2021	I prefer not to answer	90.0
5	2020	Master's degree	723.0
6	2020	Bachelor's degree	382.0
7	2020	~college	181.0
8	2020	Doctoral degree~	203.0
9	2020	I prefer not to answer	55.0
10	2019	Master's degree	837.0
11	2019	Bachelor's degree	411.0
12	2019	~college	152.0
13	2019	Doctoral degree~	262.0
14	2019	I prefer not to answer	54.0
15	2018	Master's degree	1216.0
16	2018	Bachelor's degree	588.0
17	2018	~college	197.0
18	2018	Doctoral degree~	319.0
19	2018	I prefer not to answer	81.0
20	2017	Master's degree	438.0
21	2017	Bachelor's degree	363.0
22	2017	~college	0.0
23	2017	Doctoral degree~	168.0
24	2017	I prefer not to answer	8.0

> **학력을 총 다섯개로 분류**
:Master's degree, Banchelor's degree, Doctoral degree, college, I prefer not to answer

연도별로 학력의 count 값을 구해 Data frame 으로 만든 다음 하나의 Data frame으로 합침

PART 2 **4. Data Preprocessing : 6. Experience**


```
Exp21_Wo = df21.loc[:,['Q3','Q6', 'year']].reset_index().rename(columns={'Q3':'Country', 'Q6':'Exp'}).fillna('etc')
Exp20_Wo = df20.loc[:,['Q3','Q6','year']].reset_index().rename(columns={'Q3':'Country', 'Q6':'Exp'}).fillna('etc')
Exp19_Wo = df19.loc[:,['Q3','Q15','year']].reset_index().rename(columns={'Q3':'Country', 'Q15':'Exp'}).fillna('etc')
Exp18_Wo = df18.loc[:,['Q3','Q8','year']].reset_index().rename(columns={'Q3':'Country', 'Q8':'Exp'}).fillna('etc')
Exp17_Wo = df17.loc[:,['Country','Tenure', 'year']].reset_index().rename(columns={'Country':'Country', 'Tenure':'Exp'}).fillna('etc')

Exp21_Wo= Exp21_Wo.replace({'I have never written code': '< 1 years',  '1-3 years': '1-2 years'}).replace(['10-20 years', '20+ years'], '10+ years' )
Exp20_Wo= Exp20_Wo.replace({'I have never written code': '< 1 years'}).replace(['10-20 years', '20+ years'], '10+ years' )
Exp19_Wo= Exp19_Wo.replace({'I have never written code': '< 1 years'}).replace(['10-20 years', '20+ years'], '10+ years' )
Exp18_Wo= (Exp18_Wo.replace({'0-1': '< 1 years', '1-2': '1-2 years', '5-10':'5-10 years'})
       .replace(['2-3', '3-4',  '4-5'],'3-5 years')
      .replace(['10-15', '15-20','20-25', '30 +','25-30'],'10+ years'))
Exp17_Wo=(Exp17_Wo.replace({'More than 10 years':'10+ years', '1 to 2 years':'1-2 years',  'Less than a year':'< 1 years',
      '3 to 5 years':'3-5 years', "I don't write code to analyze data":'< 1 years',
      '6 to 10 years':'5-10 years'}))

#data 결제(한재번에 이름바꾸기)
Exp5y_Wo= pd.concat([Exp17_Wo, Exp18_Wo, Exp19_Wo, Exp20_Wo, Exp21_Wo]).reset_index()
Exp5y_Wo=(Exp5y_Wo.groupby(['year', 'Exp'])
          .size()
          .reset_index()
          .rename(columns = {0:"Count"}))
```

```
#percent data 넣기
Exp21_per_W= Exp5y_Wo[Exp5y_Wo['year'] == "2021"].reset_index(drop = True)
Exp21_per_W['percentage'] = Exp21_per_W["Count"] / Exp21_per_W["Count"].sum()
Exp21_per_W['%'] = np.round(Exp21_per_W['percentage'] * 100, 1)

Exp20_per_W = Exp5y_Wo[Exp5y_Wo['year'] == "2020"].reset_index(drop = True)
Exp20_per_W['percentage'] = Exp20_per_W["Count"] / Exp20_per_W["Count"].sum()
Exp20_per_W['%'] = np.round(Exp20_per_W['percentage'] * 100, 1)

Exp19_per_W = Exp5y_Wo[Exp5y_Wo['year'] == "2019"].reset_index(drop = True)
Exp19_per_W['percentage'] = Exp19_per_W["Count"] / Exp19_per_W["Count"].sum()
Exp19_per_W['%'] = np.round(Exp19_per_W['percentage'] * 100, 1)

Exp18_per_W = Exp5y_Wo[Exp5y_Wo['year'] == "2018"].reset_index(drop = True)
Exp18_per_W['percentage'] = Exp18_per_W["Count"] / Exp18_per_W["Count"].sum()
Exp18_per_W['%'] = np.round(Exp18_per_W['percentage'] * 100, 1)

Exp17_per_W = Exp5y_Wo[Exp5y_Wo['year'] == "2017"].reset_index(drop = True)
Exp17_per_W['percentage'] = Exp17_per_W["Count"] / Exp17_per_W["Count"].sum()
Exp17_per_W['%'] = np.round(Exp17_per_W['percentage'] * 100, 1)

#data 완성
Exp5y_per_W = pd.concat([Exp17_per_W, Exp18_per_W, Exp19_per_W, Exp20_per_W, Exp21_per_W], ignore_index = True)
Exp5y_per_W= pd.pivot(Exp5y_per_W, index = "year", columns = 'Exp', values = "%").reset_index()
Exp5y_per_W.fillna('0')
Exp5y_percent_order = Exp5y_per_W['year'].tolist()
```

> 경력을 총 5개로 분류
: 1년이하, 1-2년, 3-5년, 5-10년, 기타

> 연도별로 경력 columns를 가진 데이터 프레임을 만들고,
경력 5개 카테고리로 분리

>전체 values를 퍼센트로 비율로 바꾼 후
Concat 메서드를 이용해 하나의 데이터 프레임으로 합침

Exp5y_per_W

Exp	year	1-2 years	10+ years	3-5 years	5-10 years	< 1 years	etc
0	2017	20.5	12.1	20.1	10.3	18.0	19.0
1	2018	15.7	13.1	24.4	10.6	24.7	11.6
2	2019	20.6	8.2	17.1	9.6	23.8	20.7
3	2020	22.5	15.4	22.7	12.7	22.1	4.6
4	2021	30.3	15.5	15.6	11.9	26.6	NaN

PART 2

4. Data Preprocessing : 7. Salary


```
Salary21= df21.loc[:, ['region', 'Q25', 'year']].rename(columns={'Q3':'Country', 'Q25':'Salary'})
salary21_Index=['< 999', '1,000-20,000', '20,000-59,999', '60,000-99,999','100,000-199,999', '200,000~']

Salary21=(Salary21
          .replace(['0-999','$0-999','0'], '< 999')
          .replace({'>$1,000,000':'200,000~'})
          .replace(['1,000-1,999','2,000-2,999','3,000-3,999', '4,000-4,999','5,000-7,499','7,500-9,999','10,000-14,999', '15,000-19,999'],'1,000-20,000')
          .replace(['20,000-24,999''25,000-29,999','30,000-39,999', '40,000-49,999',  '50,000-59,999'],'20,000-59,999')
           .replace(['60,000-69,999', '70,000-79,999', '80,000-89,999','90,000-99,999'], '60,000-99,999')
          .replace(['100,000-124,999', '300,000-499,999', '125,000-149,999', '125,000-149,999', '150,000-199,999'],'100,000-199,999')
          .replace(['200,000-249,999', '250,000-299,999','1,000,000','$500,000-999,999'], '200,000~')).fillna('0')
sal_order=['< 999', '1,000-20,000', '20,000-59,999', '60,000-99,999','100,000-199,999', '200,000~']

Salary21=(Salary21.groupby(['region', 'Salary'])
           .size()
           .reset_index()
           .rename(columns = {0:"Count"}))

Salary21_Ea = Salary21[Salary21['region'] == "EastAsia"].reset_index(drop = True)
Salary21_Ea['%']=((Salary21_Ea['Count'] / Salary21_Ea['Count'].sum())*100).round(2)
Salary21_Wo = Salary21[Salary21['region'] == "World"].reset_index(drop = True)
Salary21_Wo['%']=((Salary21_Wo['Count'] / Salary21_Wo['Count'].sum())*100).round(2)

Dgr_Sal_21= df21.loc[:, ['region', 'Q25', 'Q4']].rename(columns={'Q4':'Dgree', 'Q25':'Salary'})
Dgr_Sal_21 = (Dgr_Sal_21.replace(['0-999','$0-999','0'], '< 999')
          .replace({'>$1,000,000':'200,000~'})
          .replace(['1,000-1,999','2,000-2,999','3,000-3,999', '4,000-4,999','5,000-7,499','7,500-9,999','10,000-14,999', '15,000-19,999'],'1,000-20,000')
          .replace(['20,000-24,999''25,000-29,999','30,000-39,999', '40,000-49,999',  '50,000-59,999'],'20,000-59,999')
          .replace(['60,000-69,999', '70,000-79,999', '80,000-89,999','90,000-99,999'], '60,000-99,999')
          .replace(['100,000-124,999', '300,000-499,999', '125,000-149,999', '125,000-149,999','150,000-199,999'],'100,000-199,999')
          .replace(['200,000-249,999', '250,000-299,999','1,000,000','$500,000-999,999'], '200,000~')
          .replace({'I prefer not to answer':'etc'})
          .replace(['No formal education past high school', 'Some college/university study without earning a bachelor's degree'],'~college')
          .replace(['Doctoral degree', 'Professional doctorate'],'Doctoral degree~'))
```

Dgr_Sal_21

	region	Salary	Dgree
1	World	25,000-29,999	Bachelor's degree
2	World	60,000-99,999	Master's degree
3	World	< 999	Master's degree
4	World	20,000-59,999	Doctoral degree~
5	World	20,000-59,999	Doctoral degree~
...	...	...	...
25969	World	1,000-20,000	Bachelor's degree
25970	EastAsia	NaN	Master's degree
25971	World	< 999	Doctoral degree~
25972	World	NaN	Master's degree
25973	World	< 999	Bachelor's degree

> 연봉 구간 분류

: 999달러 이하, 1,000-20,000달러, 20,000-59,999 달러, 60,000-99,999 달러, 100,000-199,999 달러, '200,000달러 이상

Df21에서 region, Salary, Degree columns 를 뽑아 데이터 프레임을 만들어주고, World와 East Asia로 분리

PART 2

4. Data Preprocessing : 8. Language


```
#data preprocessing
#world
programming_list = ["Python", "R", "SQL", "Java", "C", "Bash", "Javascript", "C++"]
programming_df = pd.Series(programming_list)

df_2019 = df19[df19['Q19'].isin(programming_df)]
df_2020 = df20[df20['Q8'].isin(programming_df)]
df_2021 = df21[df21['Q8'].isin(programming_df)]

df19Lag = df_2019.loc[:, ['region', 'Q5', 'Q19', 'year']]
df19Lag = df19Lag.rename(columns = {'Q19':'Language'}, inplace = False) # To match with other datasets
df20Lag = df_2020.loc[:, ['region', 'Q5', 'Q8', 'year']].rename(columns = {'Q8':'Language'}, inplace = False)
df21Lag = df_2021.loc[:, ['region', 'Q5', 'Q8', 'year']].rename(columns = {'Q8':'Language'}, inplace = False)

df3y_Lag = pd.concat([df19Lag, df20Lag, df21Lag])
df3y_Lag = df3y_Lag.groupby(['year', 'Language']).size().reset_index().rename(columns = {0:"Count"})
df3y_Lag

# 2019
dfLang_19 = df3y_Lag[df3y_Lag['year'] == "2019"].reset_index(drop = True)
dfLang_19['percentage'] = dfLang_19["Count"] / dfLang_19["Count"].sum()
dfLang_19['%'] = np.round(dfLang_19['percentage'] * 100, 1)

# 2020
dfLang_20 = df3y_Lag[df3y_Lag['year'] == "2020"].reset_index(drop = True)
dfLang_20['percentage'] = dfLang_20["Count"] / dfLang_20["Count"].sum()
dfLang_20['%'] = np.round(dfLang_20['percentage'] * 100, 1)

# 2021
dfLang_21 = df3y_Lag[df3y_Lag['year'] == "2021"].reset_index(drop = True)
dfLang_21['percentage'] = dfLang_21["Count"] / dfLang_21["Count"].sum()
dfLang_21['%'] = np.round(dfLang_21['percentage'] * 100, 1)

dfLang_19=dfLang_19.sort_values(by='%', ascending=False)
dfLang_20=dfLang_20.sort_values(by='%', ascending=False)
dfLang_21=dfLang_21.sort_values(by='%', ascending=False)
```

> Language는 2019, 2020, 2021년의 데이터를 사용
> Language
:Python, R, SQL, C, C++, Java, Javascript, Bash

연도별로 Language 정리하고
퍼센트 값으로 바꾼 후, 각각 테이블로 만들어줌

dfLang_19

	year	Language	Count	percentage	%
5	2019	Python	1016	0.844555	84.5
6	2019	R	88	0.073150	7.3
7	2019	SQL	36	0.029925	3.0
1	2019	C	27	0.022444	2.2
2	2019	C++	18	0.014963	1.5
3	2019	Java	12	0.009975	1.0
4	2019	Javascript	4	0.003325	0.3
0	2019	Bash	2	0.001663	0.2

dfLang_20

	year	Language	Count	percentage	%
5	2020	Python	1165	0.859779	86.0
6	2020	R	71	0.052399	5.2
1	2020	C	38	0.028044	2.8
7	2020	SQL	30	0.022140	2.2
2	2020	C++	28	0.020664	2.1
3	2020	Java	17	0.012546	1.3
0	2020	Bash	3	0.002214	0.2
4	2020	Javascript	3	0.002214	0.2

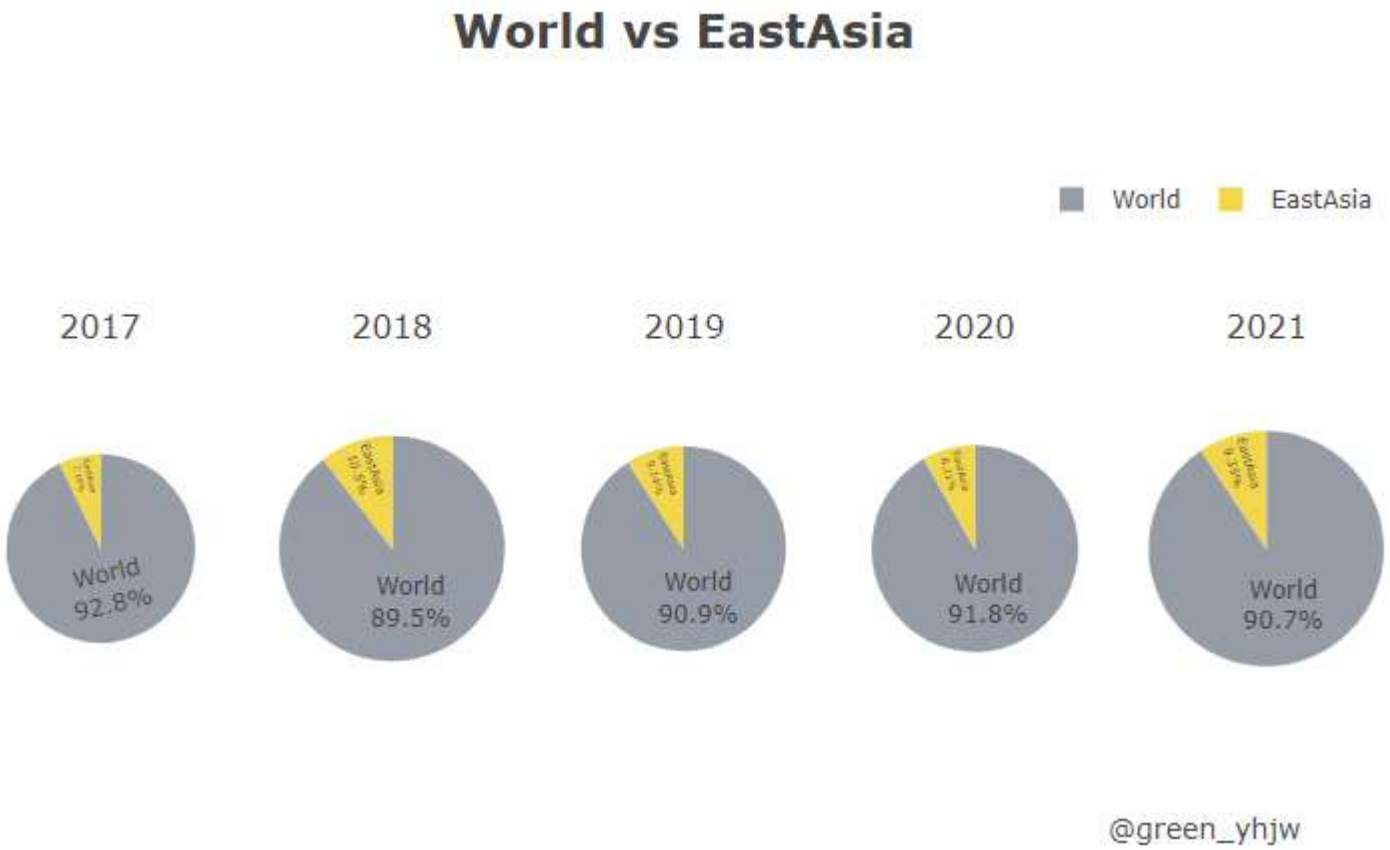
dfLang_21

	year	Language	Count	percentage	%
5	2021	Python	1974	0.853806	85.4
6	2021	R	103	0.044550	4.5
7	2021	SQL	70	0.030277	3.0
1	2021	C	64	0.027682	2.8
2	2021	C++	63	0.027249	2.7
3	2021	Java	29	0.012543	1.3
0	2021	Bash	5	0.002163	0.2
4	2021	Javascript	4	0.001730	0.2

03 분석 결과

PART 3

1.1 User Transformation in East Asia: Prediction of Outliers(18')

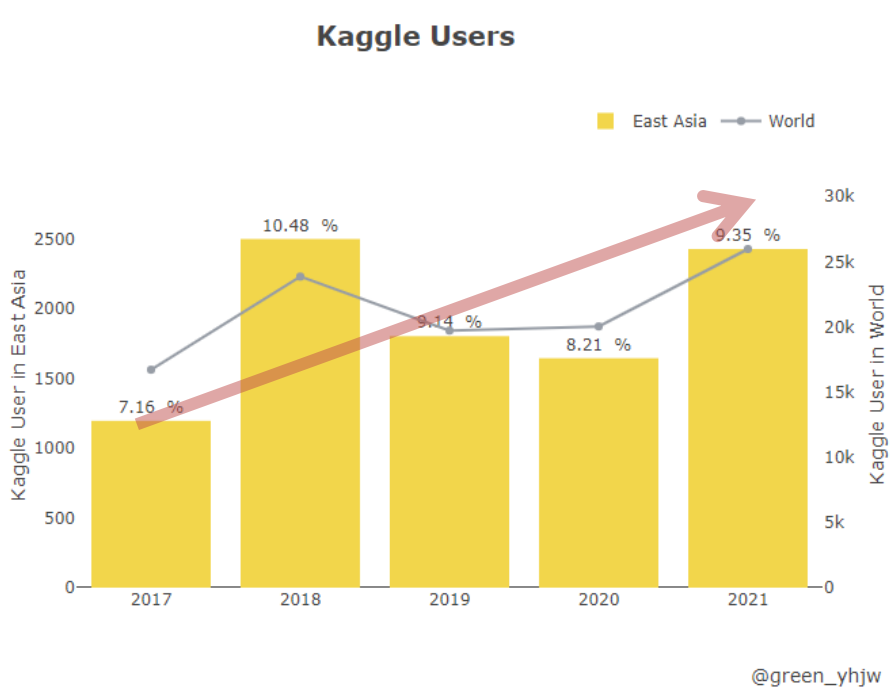


응답자 수 (Pie plot : World와 East Asia 응답자 수)

- 원의 크기: 응답자의 수에 비례, %는 연도별 비율
- 2018년도 East Asia의 비율 감소하지 않음 : 이상치= East Asia의 수 증가 연관

PART 3

1.1 Kaggle's Transformation: World Vs East Asia



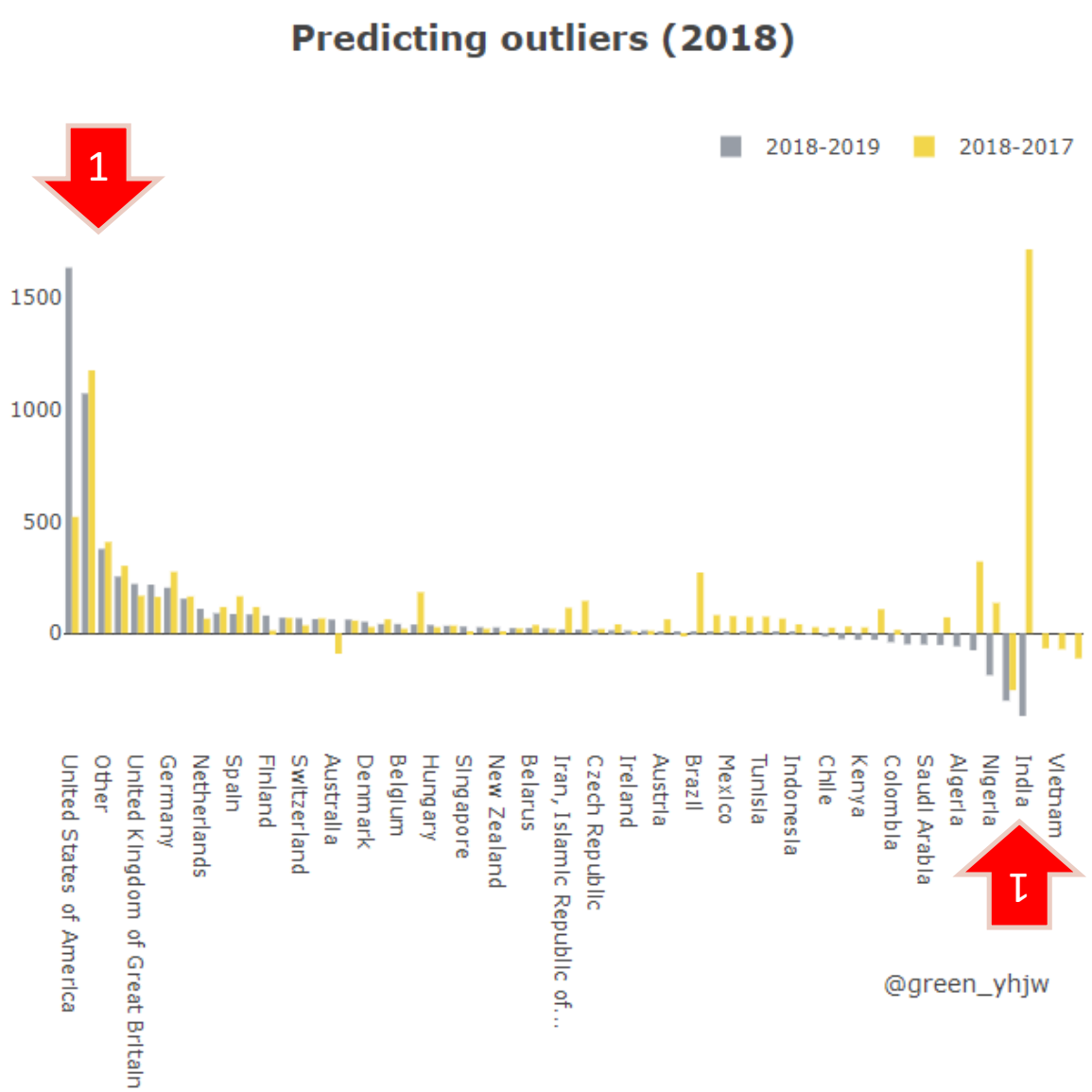
응답자 수 (bar, scatter plot : World와 East Asia 응답자 수, Map plot : East Asia 응답자 수)

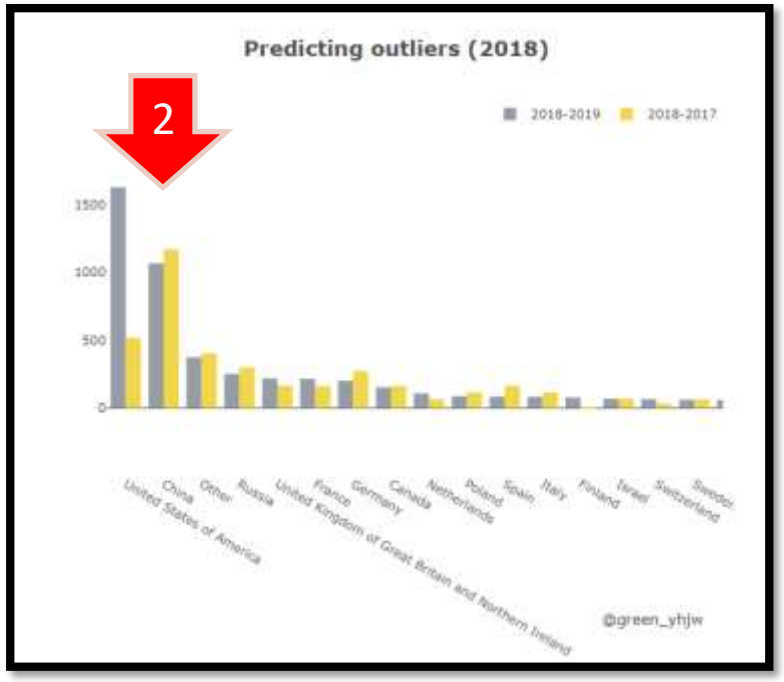
- World와 East Asia : 같은 추세.
- East Asia : 전체 대륙의 15%, 인구수는 20.3% (16/78.7 : Ea/Wo)
- 2018 Issue : 응답자의 유의미하게 증가->가설: china의 급격한 증가 때문
- 2018년 이상치 감안 : 2022년 Kaggle survey 응답자 수: World와 East Asia 모두에서 증가

우리 팀도 2022년 Kaggle survey 응답자가 되는 영광을 기를….

PART 3

1.2 User Transformation in East Asia: Prediction of outliers(18')





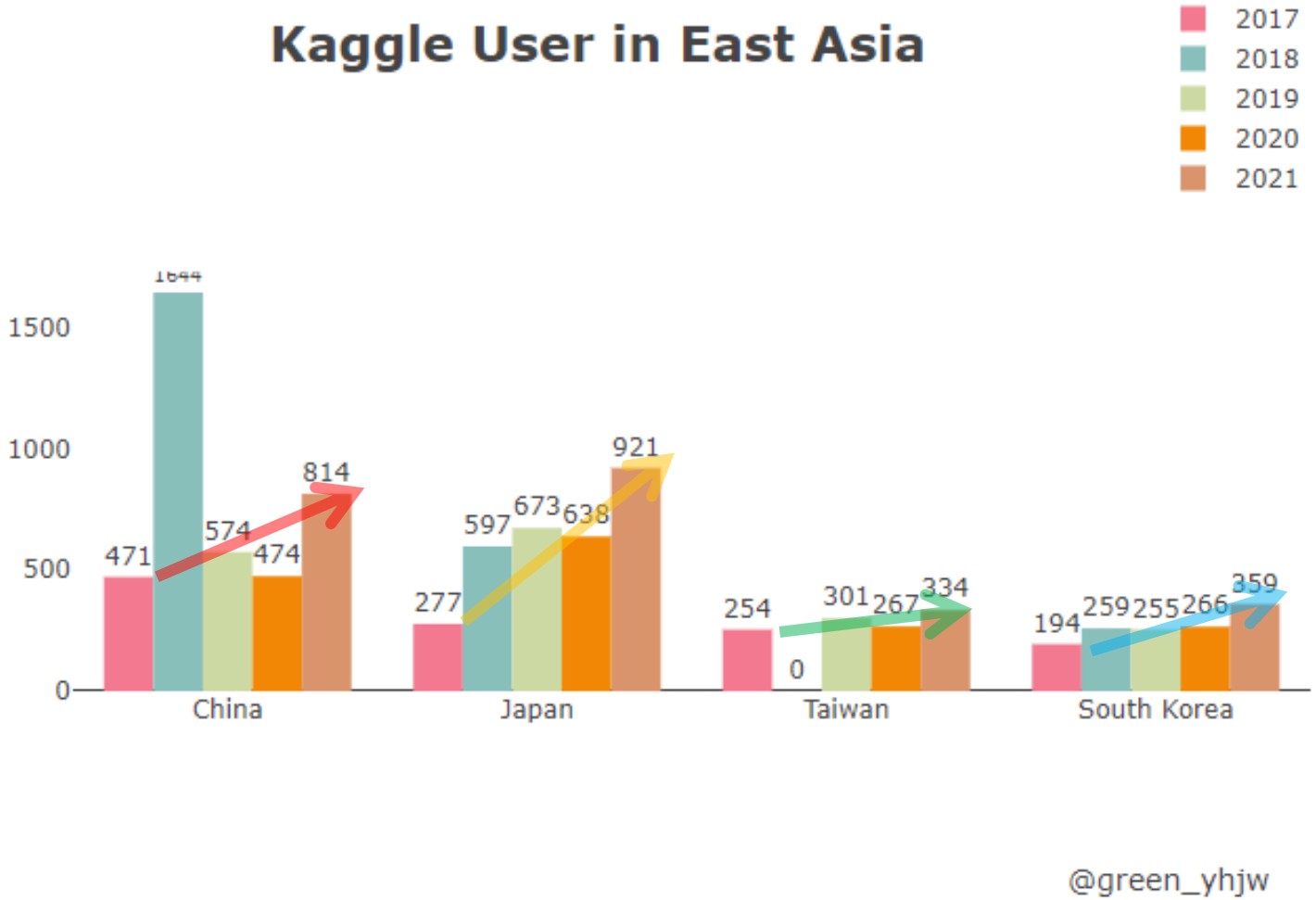
18' :

1. United Satates와 India 사이의 유저 변동
2. China가 2018년에 기록적 증가

\+ 대만은 없고, 중국만 증가. : 동아시아 정치 상황 Issue 의심

PART 3

1.3 user Transformation in East Asia



총 인구수	1.3억	0.5억	2.6
Resp.	일본	한국	배율
2017	277	194	1.43
2018	597	259	2.31
2019	673	255	2.64
2020	638	266	2.40
2021	921	359	2.57

총 인구 수 :

중국 14억 (85%), 일본 1.3억, 한국 0.5억, 대만 0.2억

- **중국** : 인구수에 비해 적은 응답자 수
- **일본** : 2019년 부터 중국 추월
- **대만** :
  - 2018년도 data 0 : 외교문제?
  - 증가 추세가 미약하지만 있다.
- **한국** :
  - 일본과 인구수 대비 비슷한 수준의 응답자.
- **동아시아** : 응답자 수가 더 증가 할 것이다.

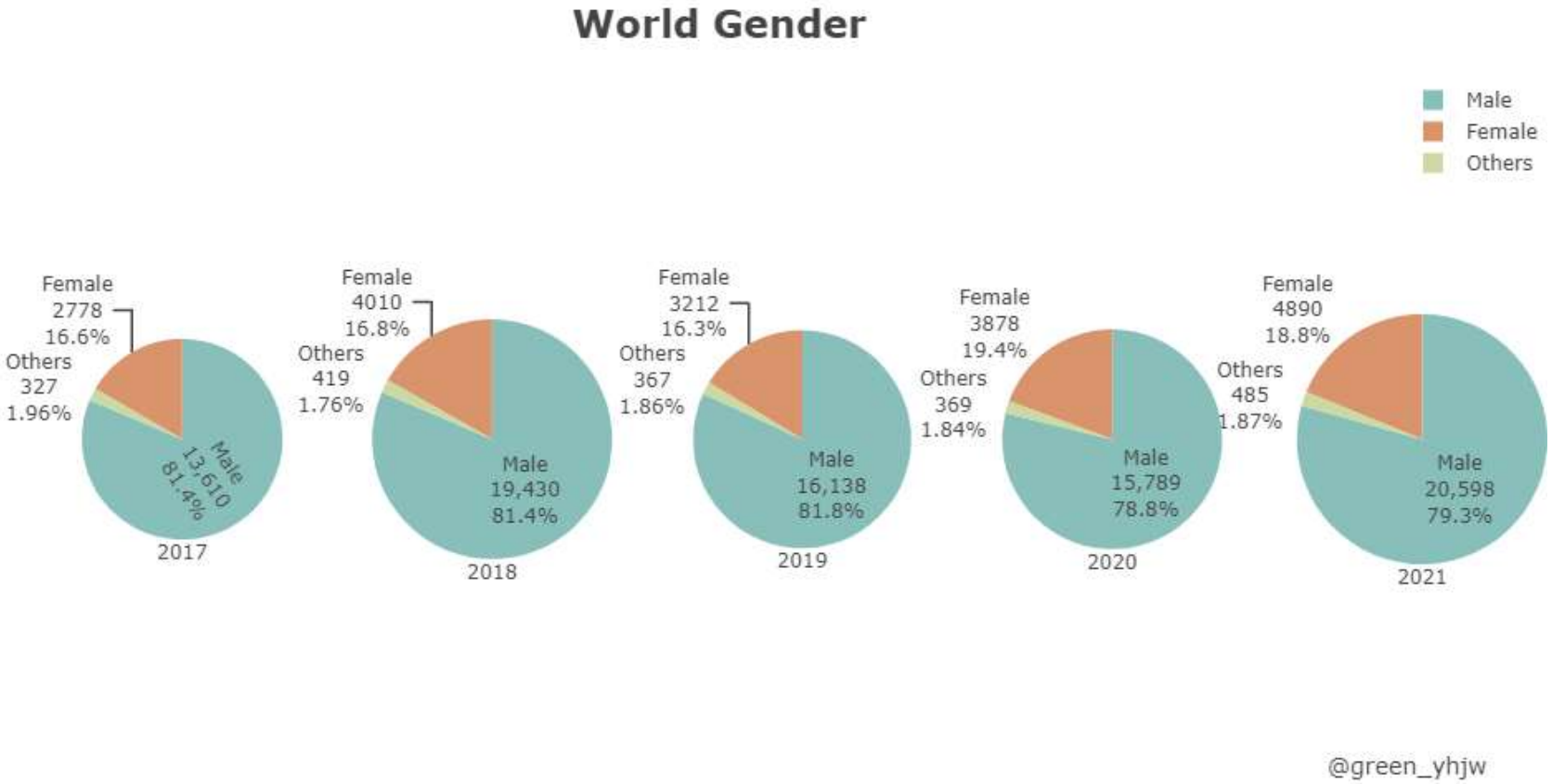
PART 3

2.1 Gender Transformation: World Vs East Asia

World: 여성 응답자의 비율이 증가 (여전히 20% 이하)

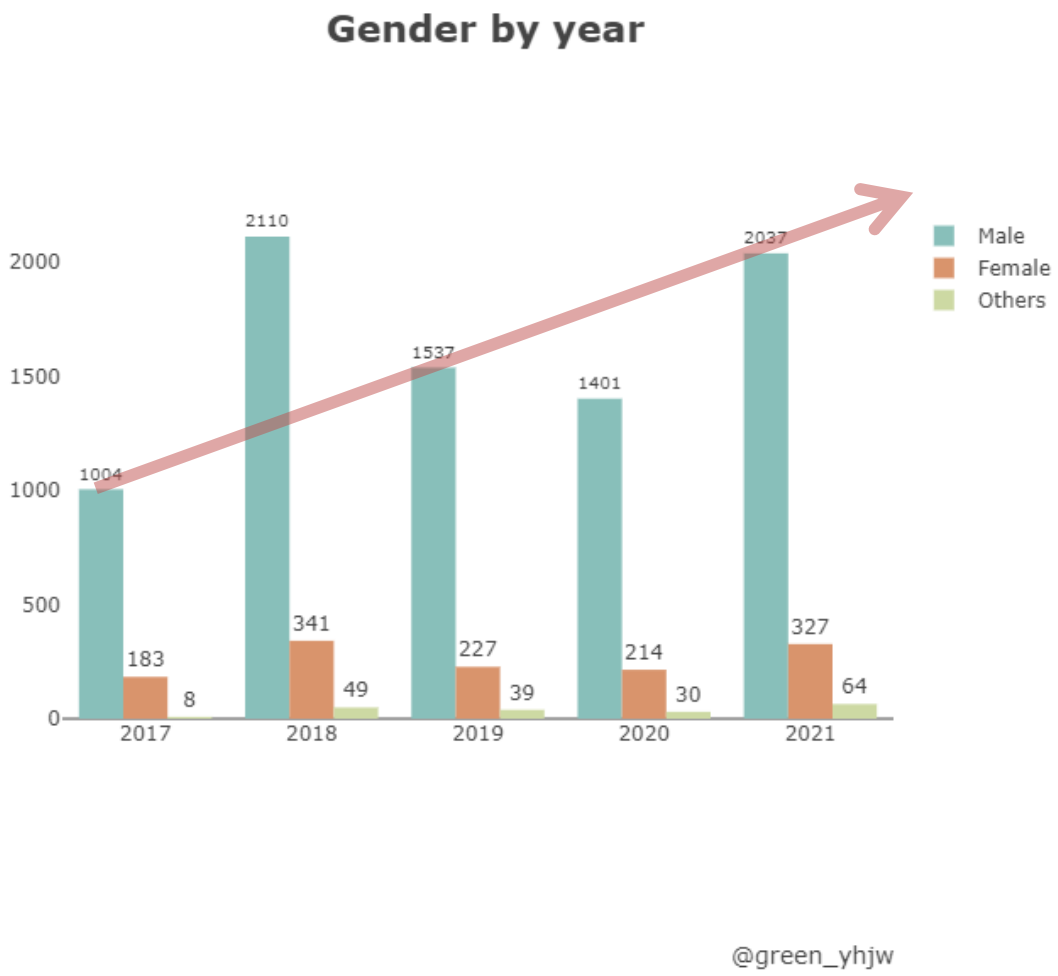
응답자 수는 전 성별이 증가하는 추세

우리 팀도 여성 구성원이 높은 팀으로써 2022년도 응답자로 기여하고싶다.



PART 3

2.2 Gender Transformation in East Asia



East Asia의 남성 응답자의 수(1004->**2037** : 2017->**2021**) 2배 상 증가

East Asia 의 여성 응답자의 수(183->**327** : 2017->**2021**) 1.8배 증가

East Asia 의 기타 응답자의 수(8->**64** : 2017->**2021**) **8배 증가**

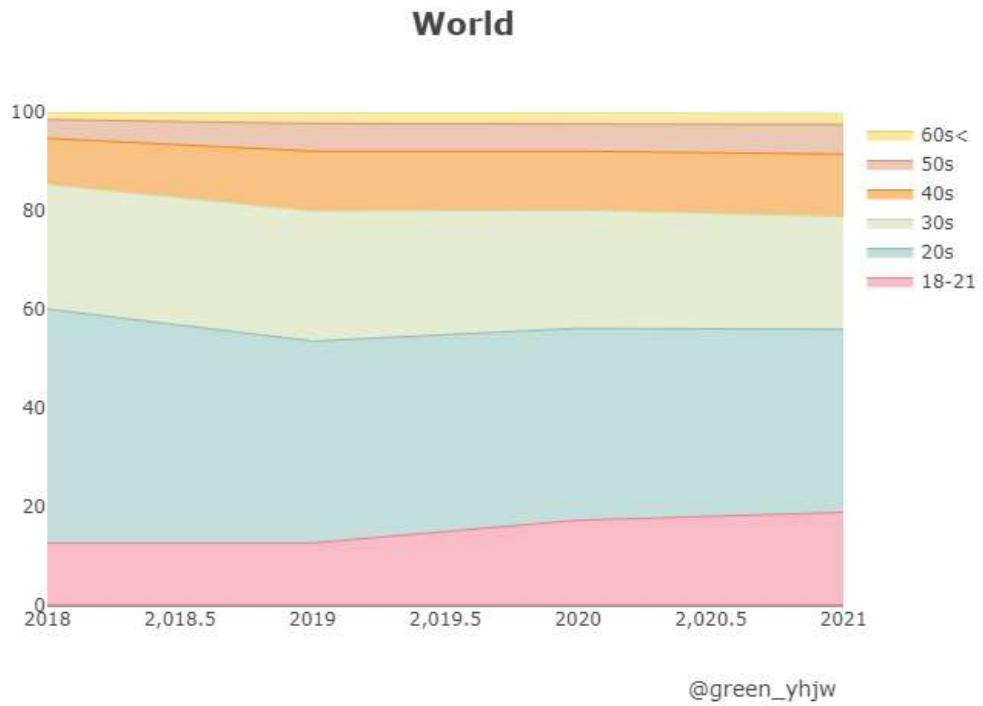
[고점대비 저점 비교]

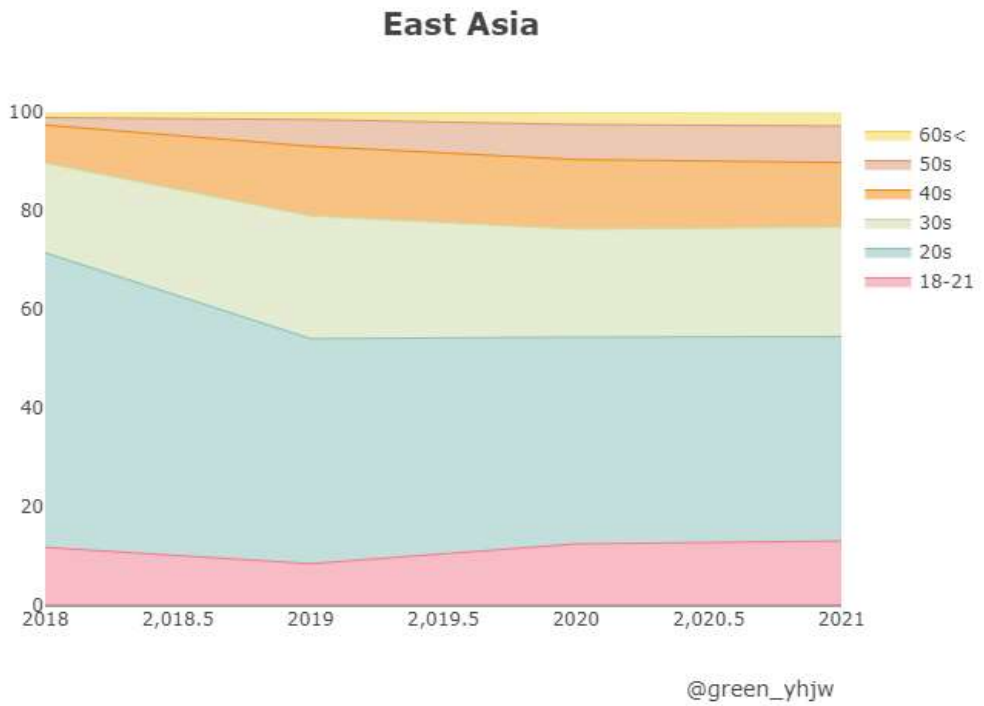
+ 여성 응답자의 수와 남성 응답자의 **비율이 거의 변하지 않는 것**을 알 수 있으며, 이는 World 자료와 비교하여 차이가 나는 부분이다.

상대적으로 East Asia내의 성별 자유도가 증가 한 것을 알 수 있다. World자료와 비교 해보자면, 상대적으로 보수적이었던 2017년(1.96 : 0.7 = Wo : Ea)에 비해 2021년 (1.87 : 2.6= Wo : Ea)에는 오히려 더 자유로워 진 것을 볼 수 있다.

PART 3

3.1 Age Transformation: World Vs East Asia



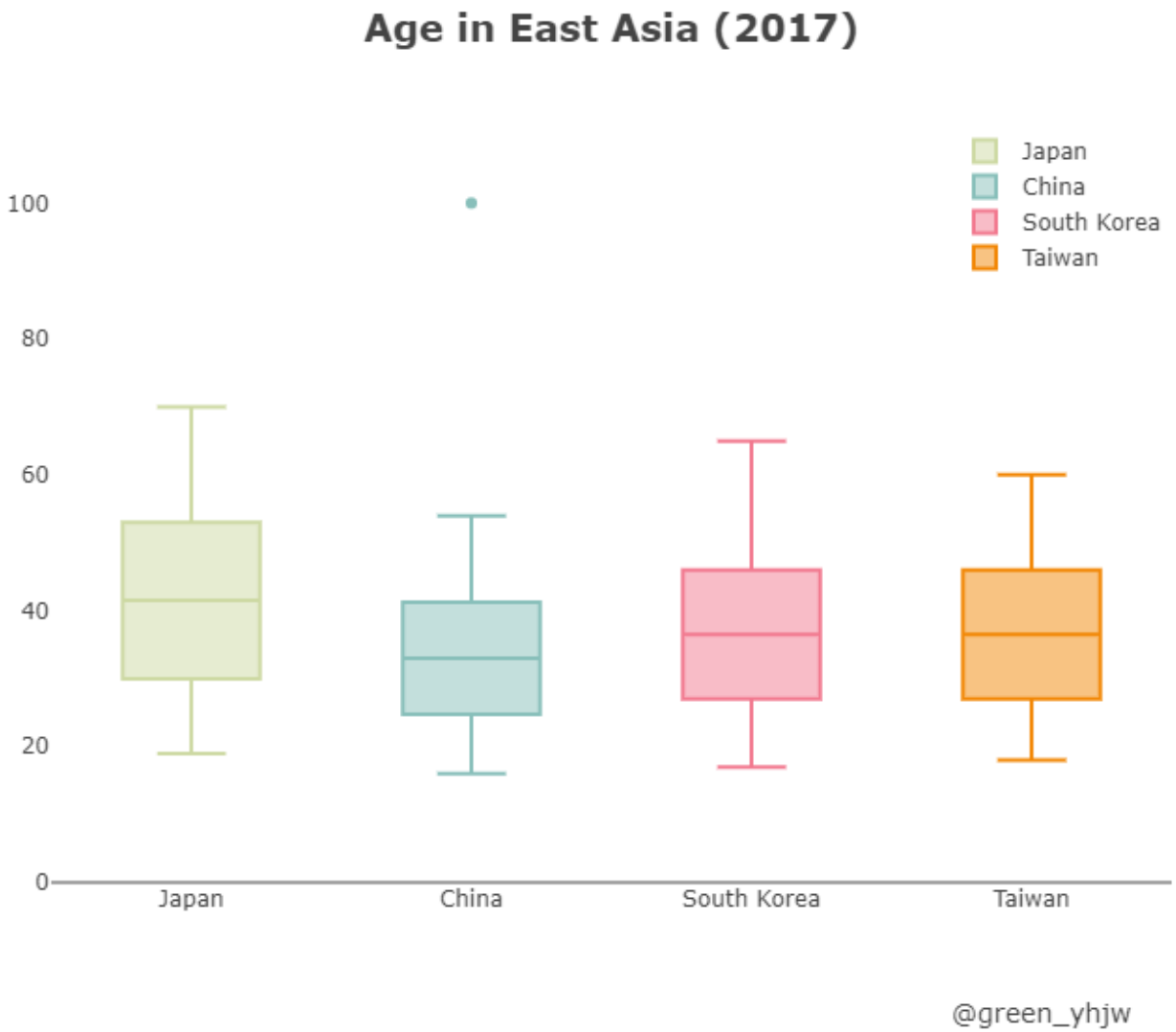


> 연도별 World와 East Asia의 연령 변화 : Stacked scatter plot

- Age data의 경우 2017년 data가 없음.
- World 응답자의 **70%가 20~30s**
- East Asia 응답자의 **70%가 20~30s**
- 응답자 숫자는 증가하지만 비율이 안정화 된 것으로 보임

PART 3

3.2 Age Transformation in 2017

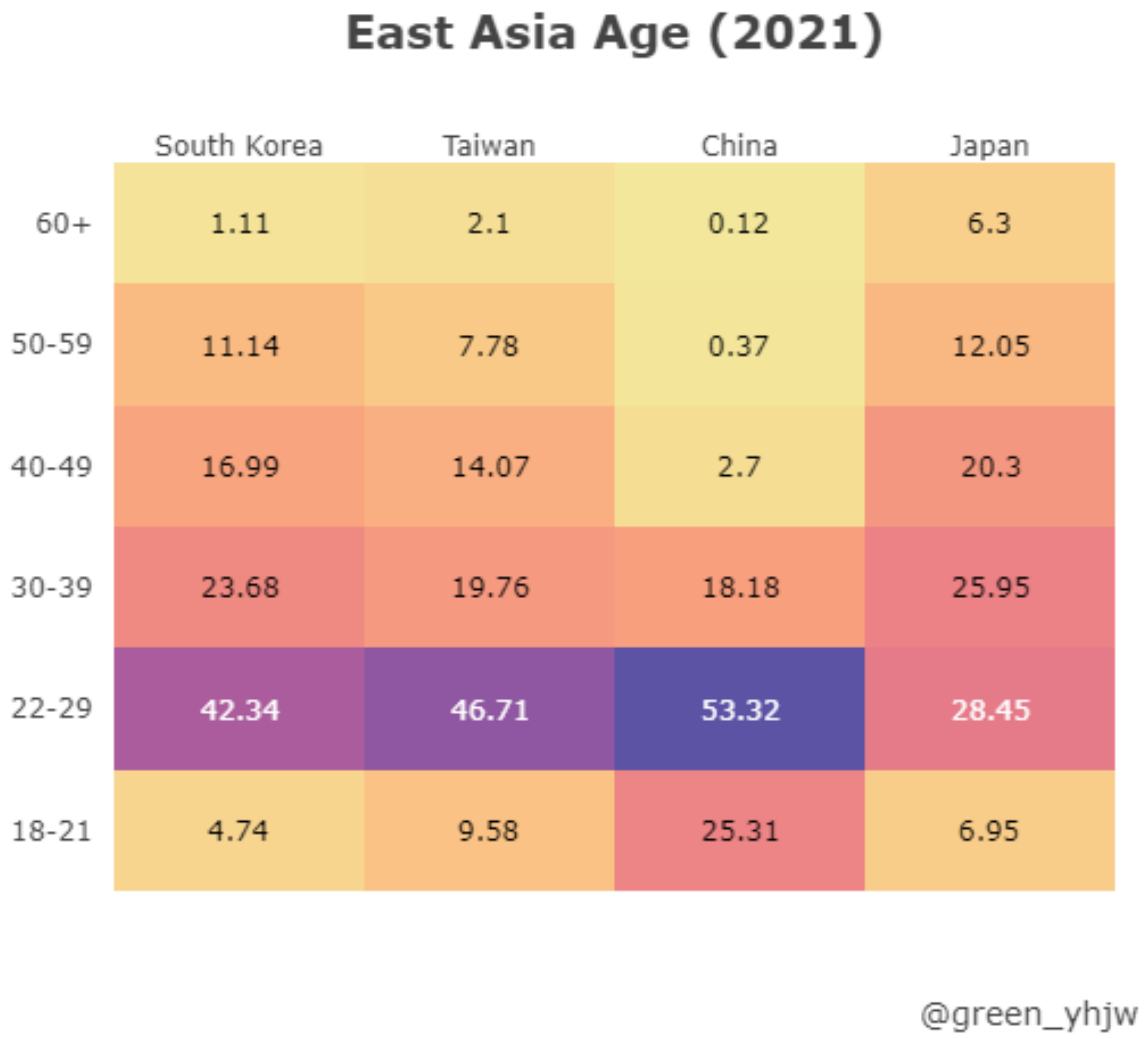


> 17' East Asia의 연령 비율 : Box plot

- 2017 : data가 구간이 아닌 개별 숫자
- 구간을 나누면 앞의 그래프에 추가 가능
- Bar plot을 그릴 수 있는 data 였기 때문에 그려 봄.
- 중국에 100세가 보이는데, 결측치 제거는 일부러 하지 않음.

PART 3

3.3 Age Transformation in East Asia(21')

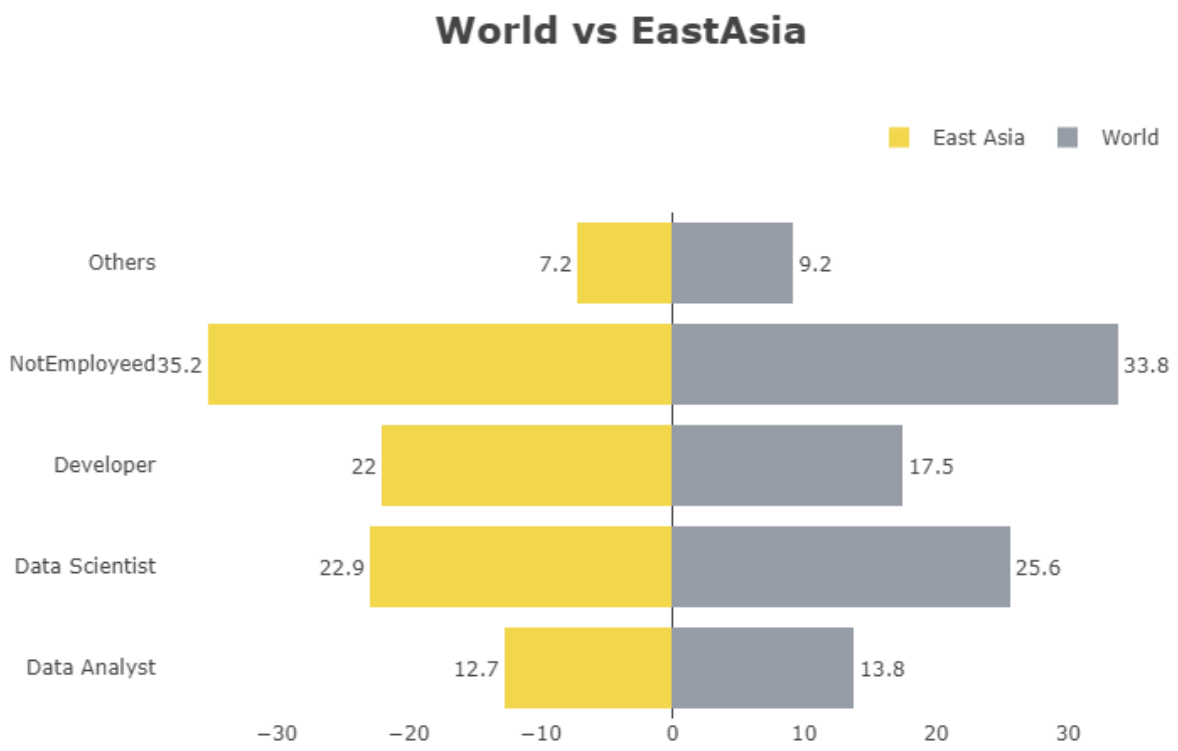


> 17' East Asia 각 나라의 연령 비율 : Heat Map

- East Asia :
  - 50% 이상 20~30대
- 한국:
  - 20대가 가장 높음
  - 50대 이상 응답자의 수도 많다.
- 대만:
  - 비교적 30대 이상 응답자의 수가 적다.
- 중국:
  - 30대 이하 응답자 70% 이상
  - 평균 수명과 연관?
- 일본:
  - 고령화 나라 답게 전 연령 고르게 분포
  - 나이가 많아도 Kaggle을 응답자 많음

PART 3

4.1 Job Transformation: World Vs East Asia (21')



@green_yhjw

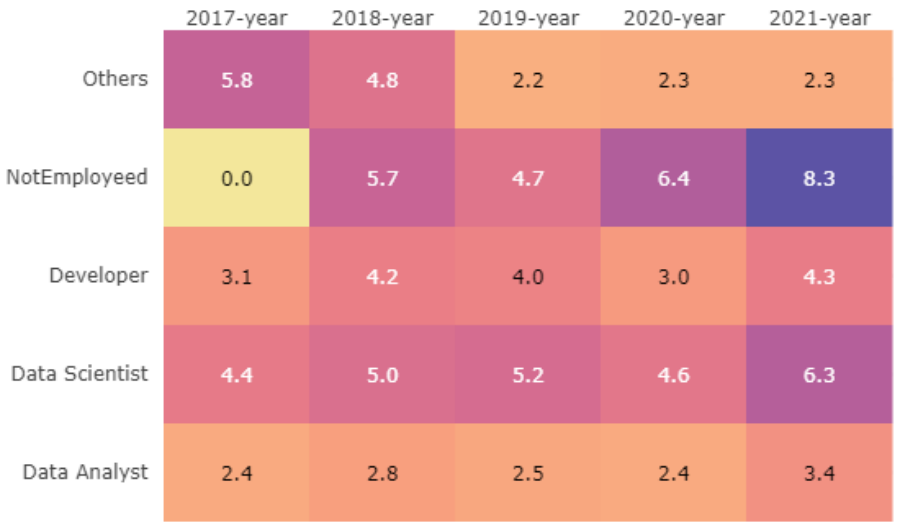
> 21' World Vs East Asia 연령 비율 : bar plot

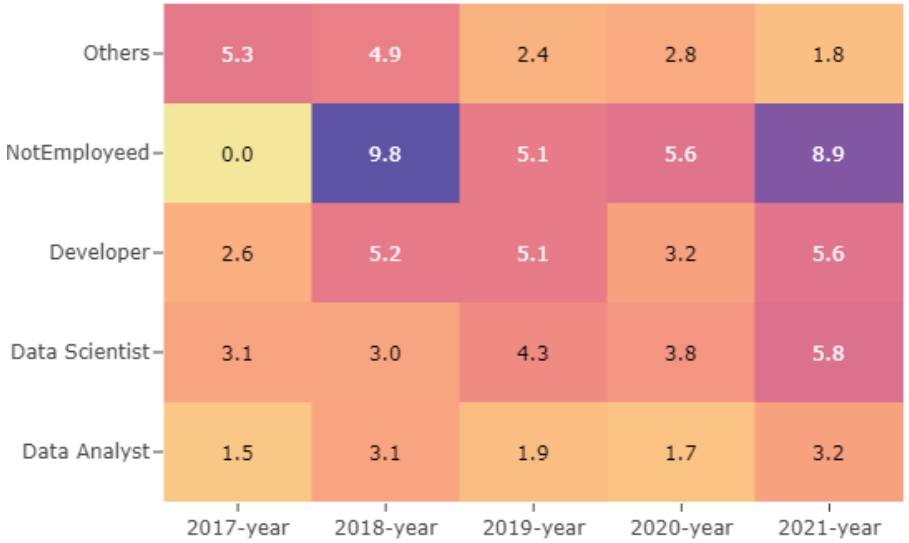
- **Not Employed :**
  East Asia 와 world모두에서 30% 이상, 가장 높음
  Students가 포함 되어 있기 때문
- **Data Scientist :**
  - World와 East Asia에서 높은 비율
  - 상대적으로 East Asia에서 **비율 상 낮음**
    = 절대적 **'수' 부족**

우리는 East Asia에서 수가 부족한 data scientist를 선택하여 앞으로 나아가고자 한다.

PART 3

4.2 Job Transformation: World Vs East Asia





> World 각 나라의 직업 비율 : Heat Map

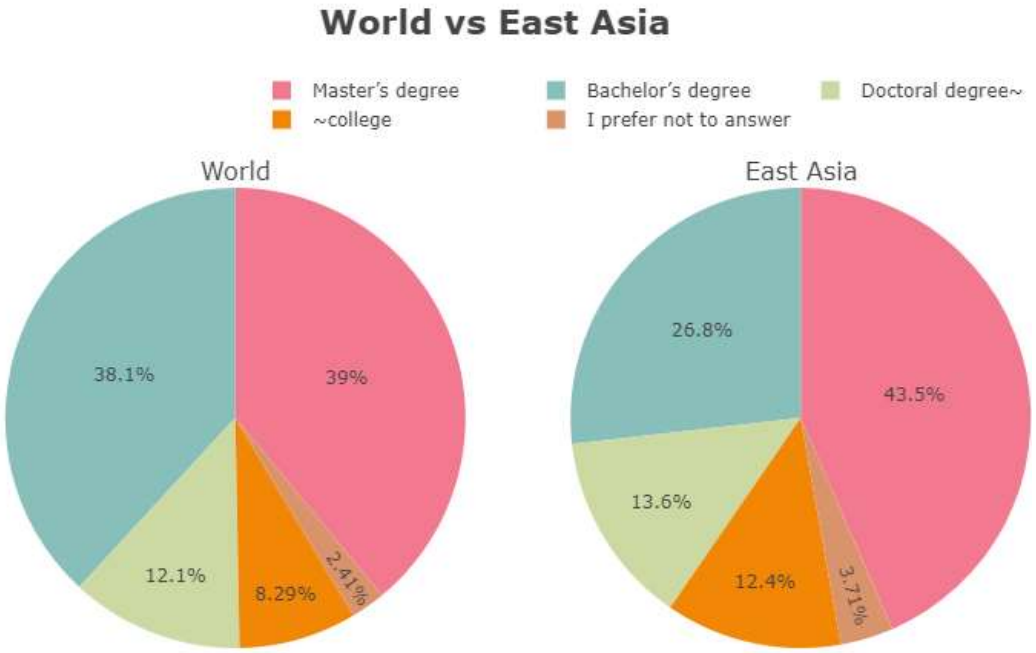
- ✓ Others를 제외한 각 직업들이 증가하는 추세.
- ✓ Data Scientist가 높은 비중, 2022년에 더 증가 할 추세.

> East Asia 각 나라의 직업 비율 : Heat Map

- ✓ East Asia :
- ✓ data scientist, 의 비율이 증가.
- ✓ Not Employed

PART 3

5.1 Degree Transformation: World Vs East Asia (21')



> World 각 나라의 직업 비율 : pie plot

World:

✓ 90% 이상 학사 이상의 학위

East Asia

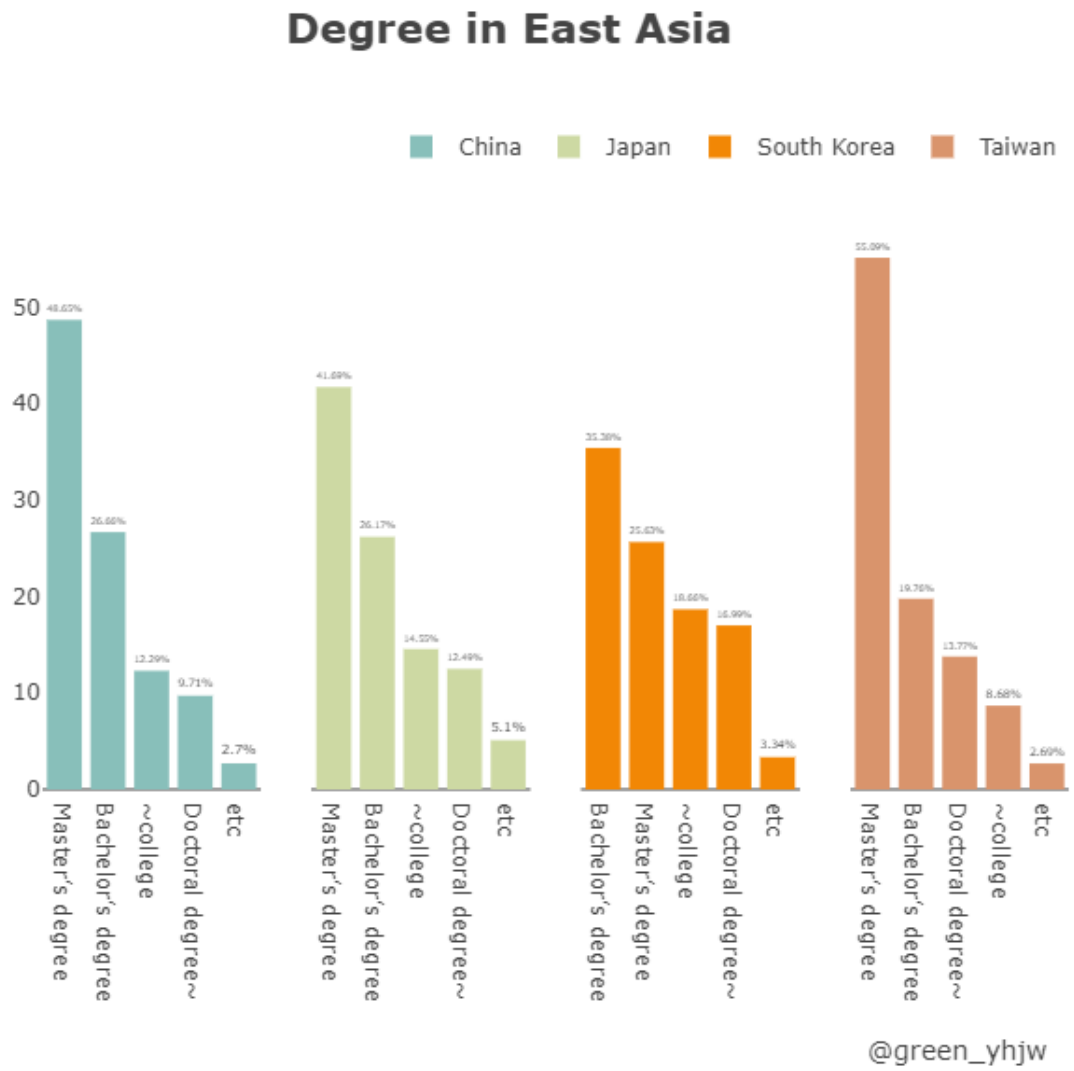
✓ 85% 정도 학사 이상의 학위

> 연도별 East Asia 학위의 비율 : sunburst plot

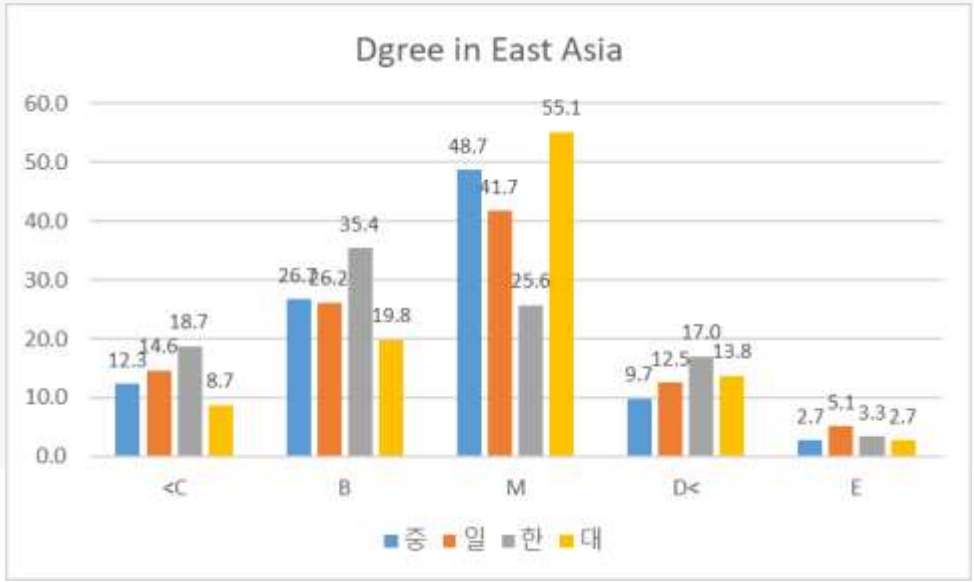
석사 학위를 가진 응답자가 매년 가장 높은 비율

PART 3

5.3 Degree Transformation in East Asia detail

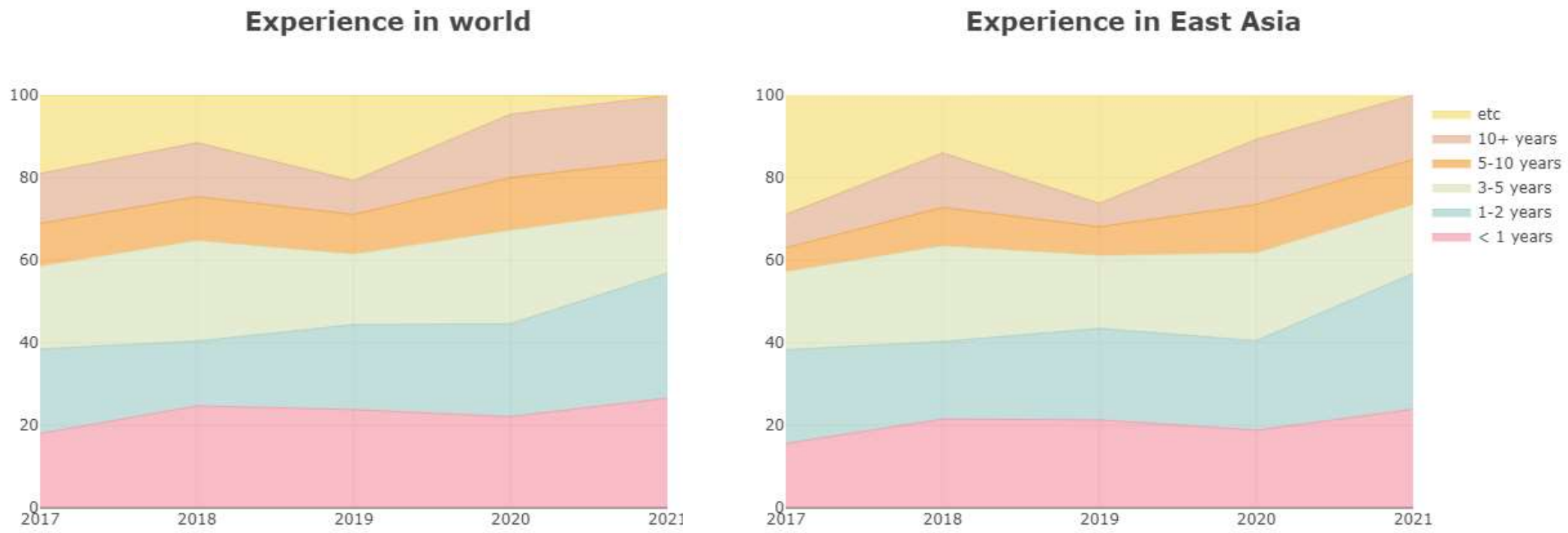


East Asia내의 국가별로 살펴 보더라도, 대부분의 나라는 석사 학위를 취득한 응답자들이 40%이상이다. 특히 대만의 경우 응답자의 50%이상이 석사학위를 가지고 있다.
특이하게도 한국은 학사 학위를 가진 응답자가 석사학위를 가진 응답자 보다 10% 더 많았으며, 박사학위 이상을 가진 응답자의 비율이 17%로 2위와 약 3%이상 차이 나는 것을 알 수 있다.



PART 3

6.1 Experience Transformation : World Vs East Asia

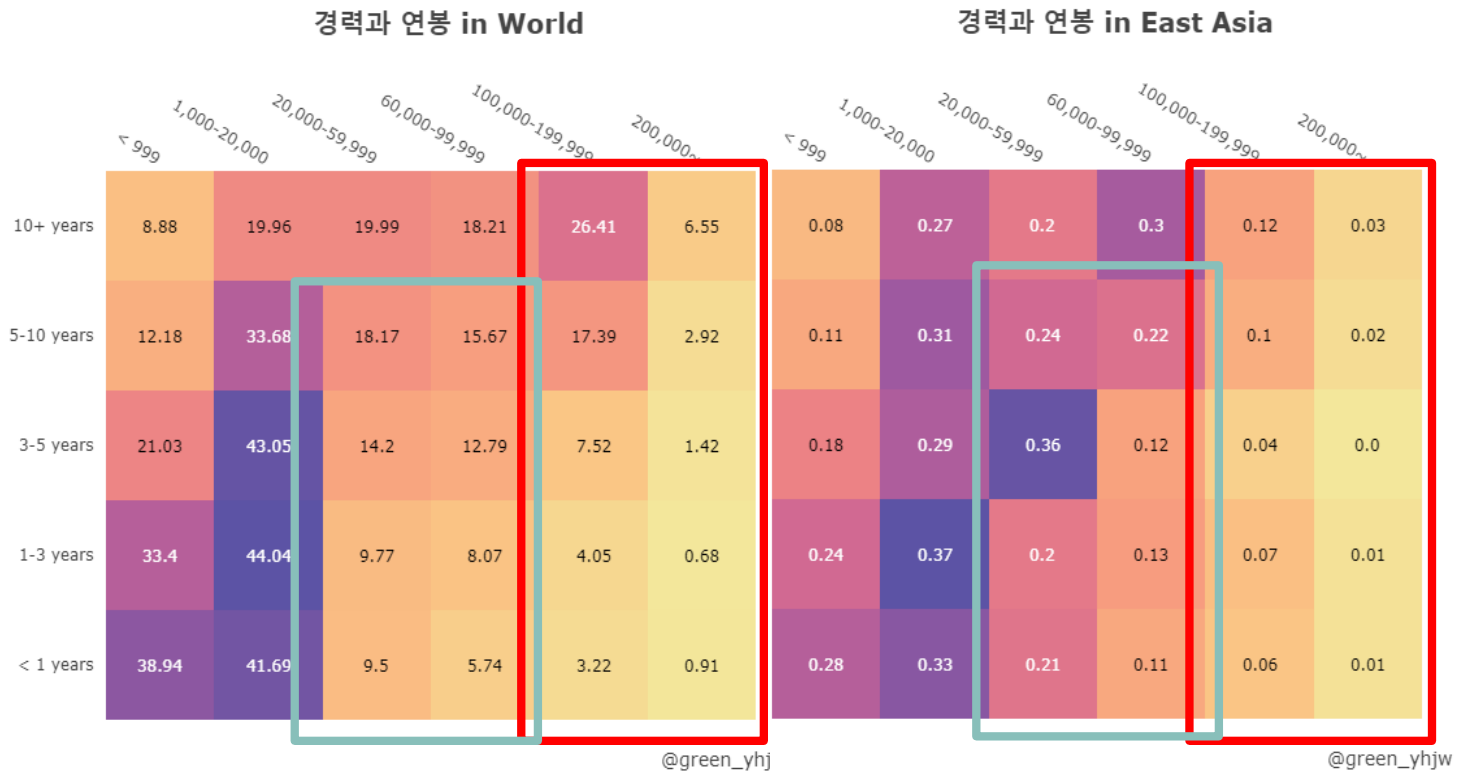


> World & East Asia 경력의 추세 : Stacked Scatter plot

- ✓ < 2년 : 전체 50%를 차지
- ✓ 3-5 years : World에서는 감소, East Asia 비율 유지
- ✓ 2021 기타 data 사라짐

PART 3

6.1 Experience Transformation : World Vs East Asia



> World 경력과 연봉 : Heat Map

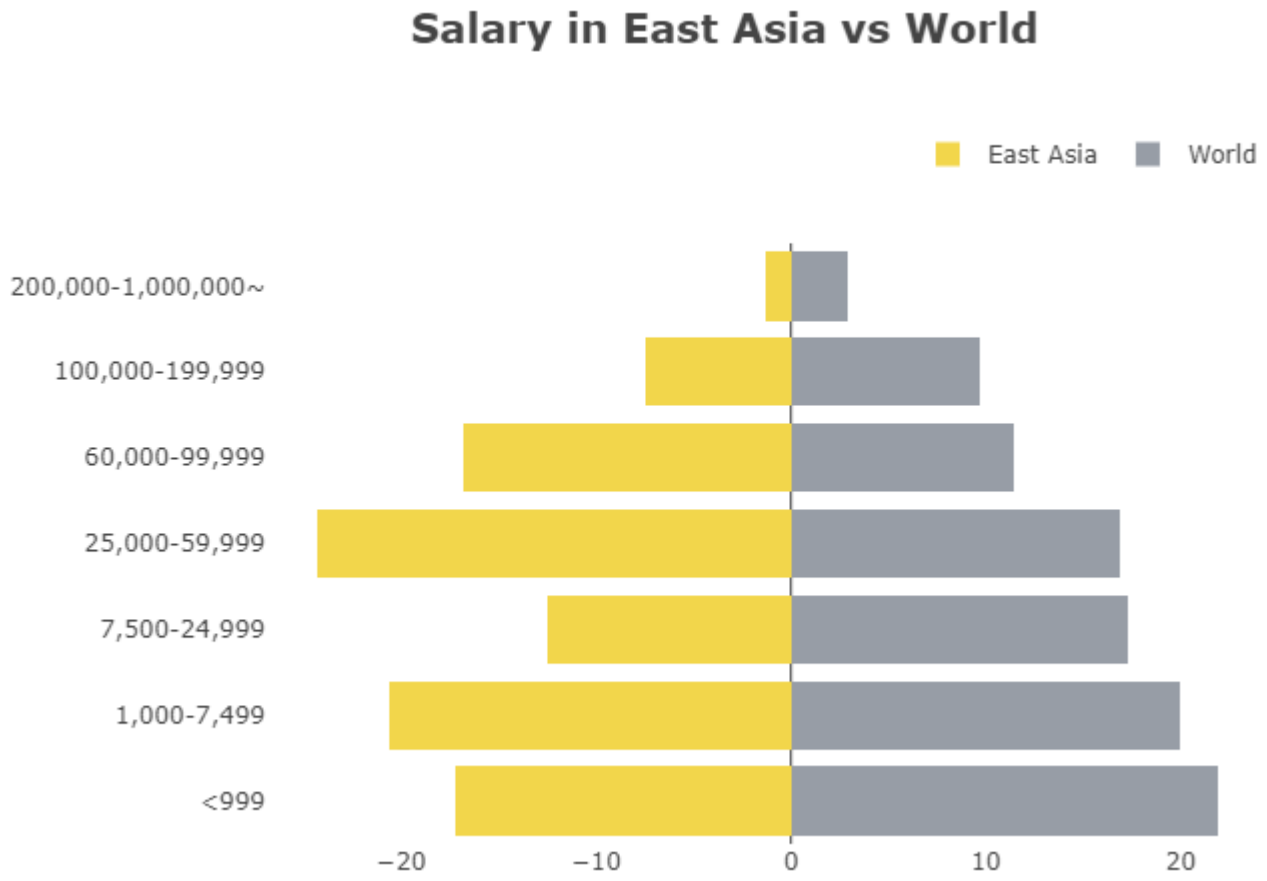
- ✓ 비교적 양의 상관관계
- ✓ 5-10년의 경력에도 45% 이상이 연봉 20,000$ 이하
- ✓ 10년 이상 경력에는 30% 이상이 연봉 100,000$을 받음

> East Asia 경력과 연봉 : Heat Map

- ✓ World 보다 더 양의 상관관계를 명확하게 보여줌
- ✓ 3-5년 경력의 36%가 위가 목표하는 연봉 수령.
- ✓ World에 비해 높은 연봉을 받기 힘들지만, 낮은 경력에도 최저의 연봉을 받지 않음

PART 3

7.1 Salary Transformation : WorldVs East Asia (21')



> World & East Asia 연봉 : Bar-H plot

$ 200,000~ :

World(2.9%)는 East Asia (1.3%)에 비해 50%이상

$ ~ 250,000 :

World(59.2%)는 East Asia (50.3%)에 비해 적음

= East Asia의 연봉 빈부격차가 덜하다.

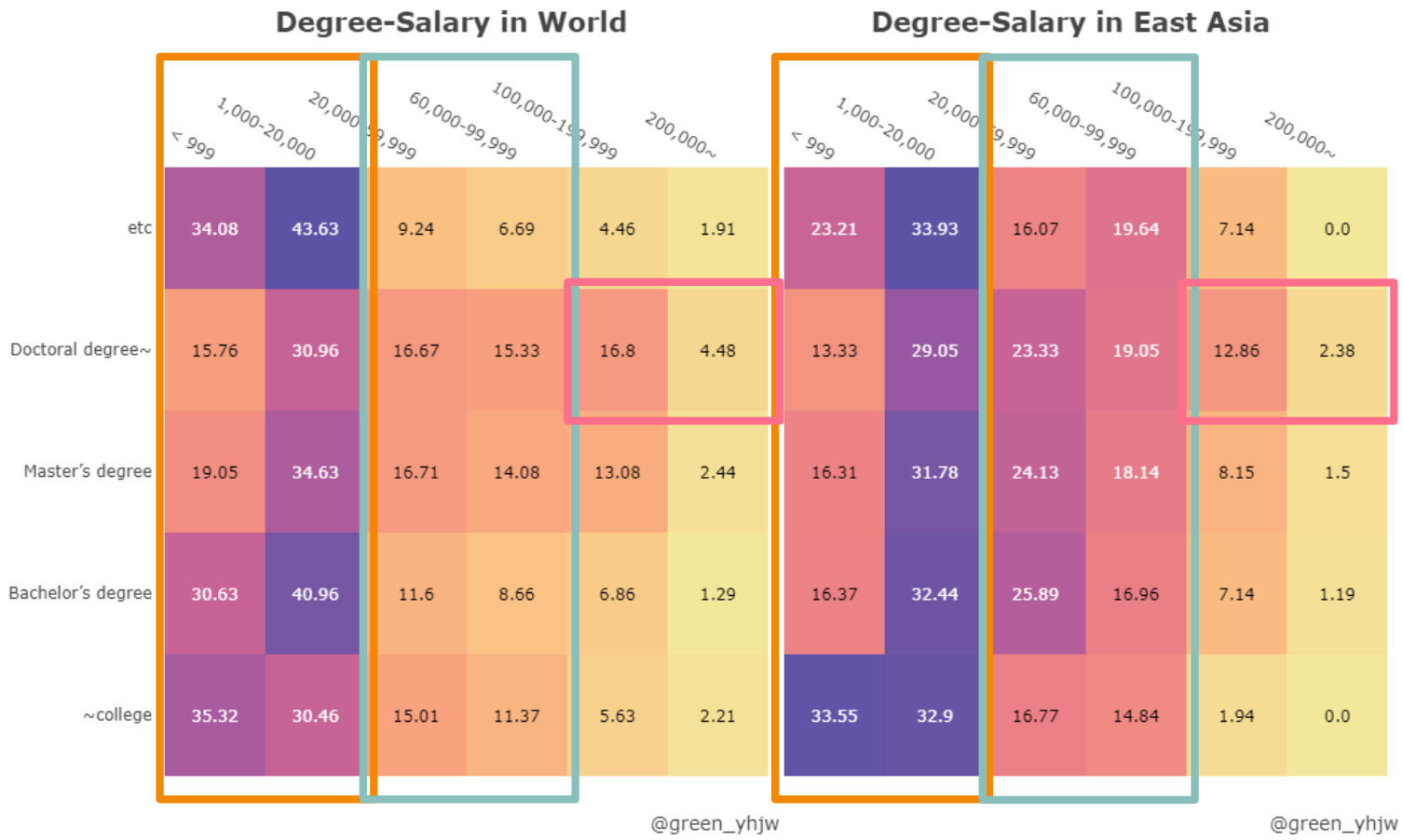
$ 25,000~60,000:

East Asia에서 24%로 가장 높은 구간

= 우리가 목표하는 연봉 구간.

PART 3

7.2 Salary Transformation : Salary and Dgree



> World & East Asia 학위/연봉 : Heat Map

$ ~20,000 :

✓ 학위에 상관없이 40% 정도는 연봉 20,000$ 이하. **학생**에서 오는 비율 일거라 추측.

$ 25,000~100,000:

✓ East Asia에서 학사 학위만으로도 40% 이상이 받음

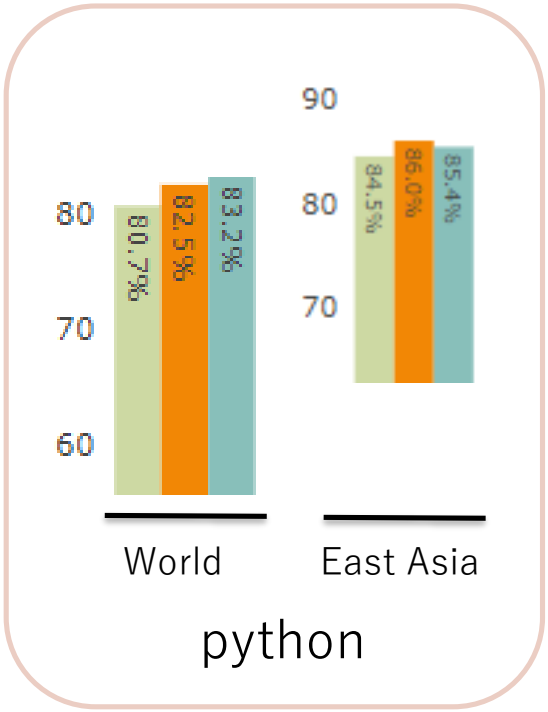
✓ (World : 20% 미만)

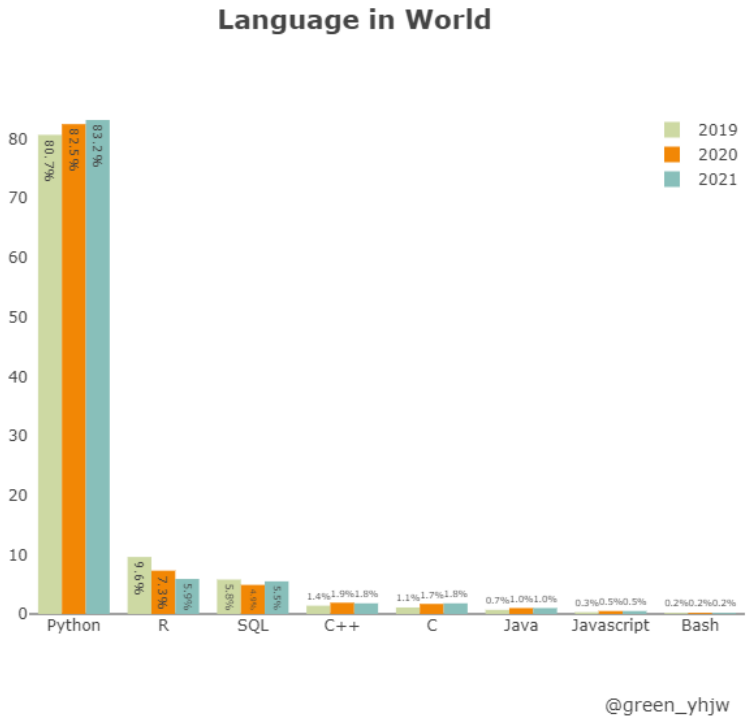
$ 200,000~:

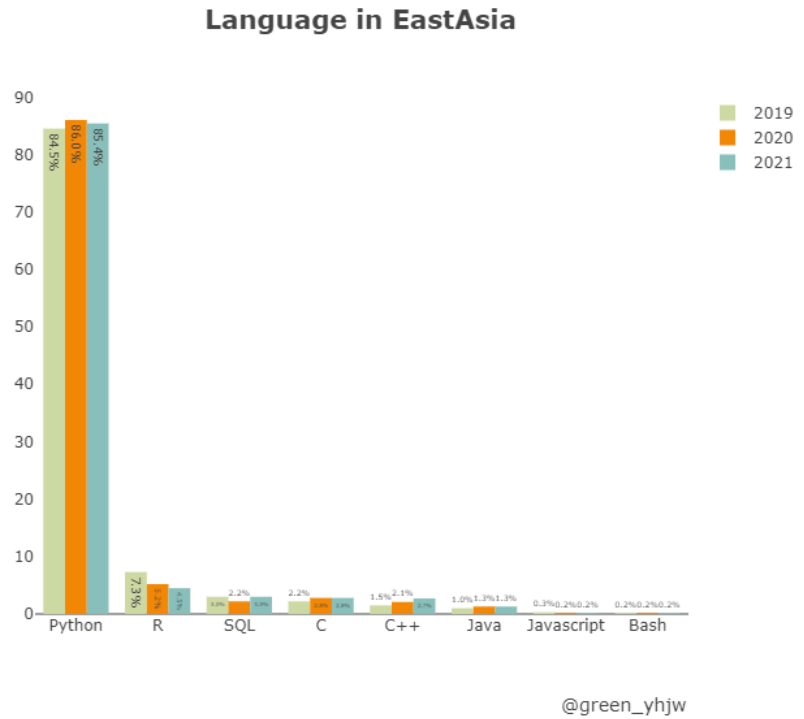
✓ 박사 학위 이상을 받아도 East Asia에서 받기 힘듦.

PART 3

8.1 Language Transformation. (World/East Asia)







> **World & East Asia 프로그래밍 언어 : Bar plot**

Python :

World 의 80% , **East Asia의 경우 85%** 정도가 사용.

현재 우리도 python으로 project를 진행 했는데 앞으로도 python을 열심히 배워서 Data scientist의 경력자가 될 수 있기를 !!

03 분석 결과

3.2 position of data scientist in East Asia

- 3.2.1 Salary
- 3.2.2 Salary-Experience
- 3.2.3 Degree
- 3.2.4 Salary-Degree
- 3.2.5 Language
- 3.2.6 Summary

PART 3 3.1 Kaggle's Transformation (World/East_Asia)

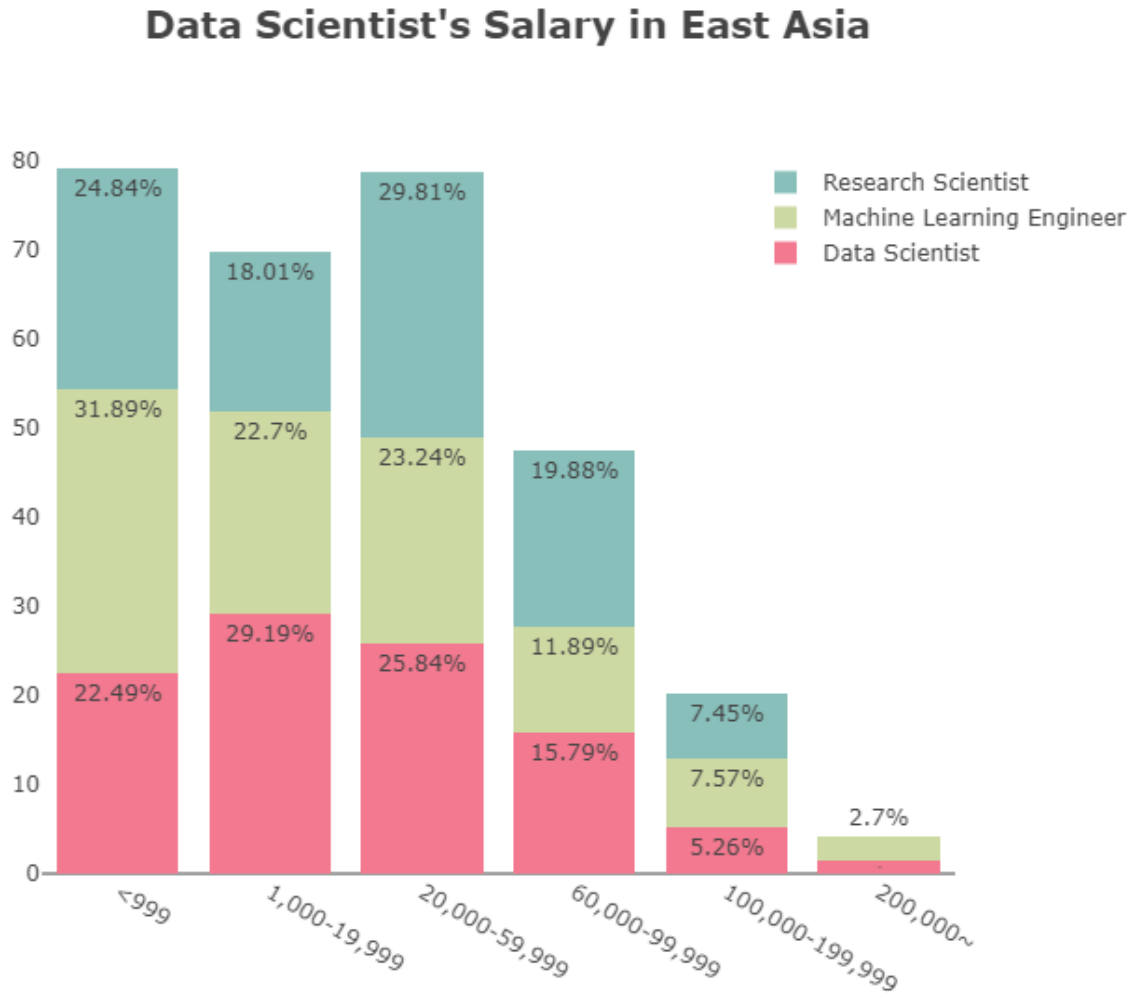
> 동아시아에서 Data Scientist의 연봉

Research Scientist의 연봉
: 2-6만달러의 비율이 29.81%로 가장 높음

Machine Learning Engineer의 연봉
: 999달러의 비율이 31.89%로 가장 높음

Data Scientist의 연봉은
: 1,000-2만달러의 비율이 29.19%로 가장 높음

⇒ 연봉이 높을수록 전체적인 직업 비율은 감소



PART 3

3.2 Kaggle's Transformation (Experience & Salary)

> Data Scientist의 경력과 연봉의 상관 관계

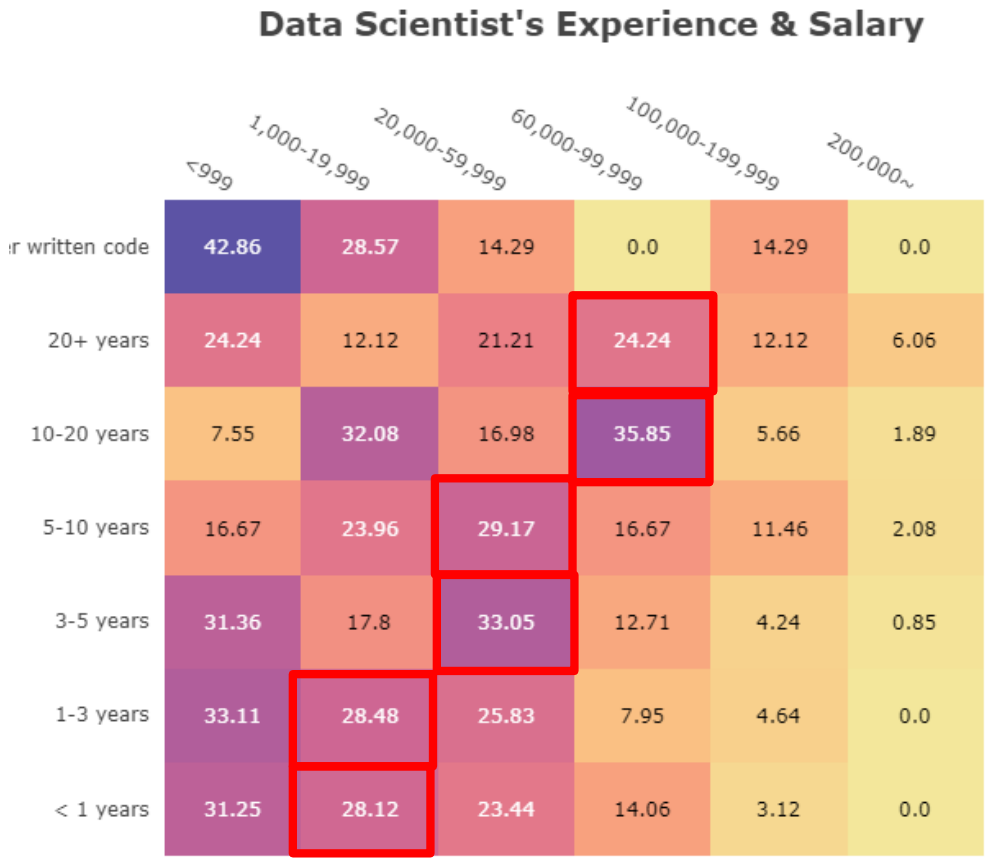
경력이 없는 경우 999달러 비율이 가장 높음

1년 이하, 1-3년은 999달러의 비율이 가장 높음

3-10년은 2만 달러-6만달러의 비율이 가장 높음

10-20년은 6만-10만 달러의 비율이 가장 높음

⇒ 따라서 경력이 많을수록 연봉은 증가했다



PART 3

3.3 Kaggle's Transformation (Degree)

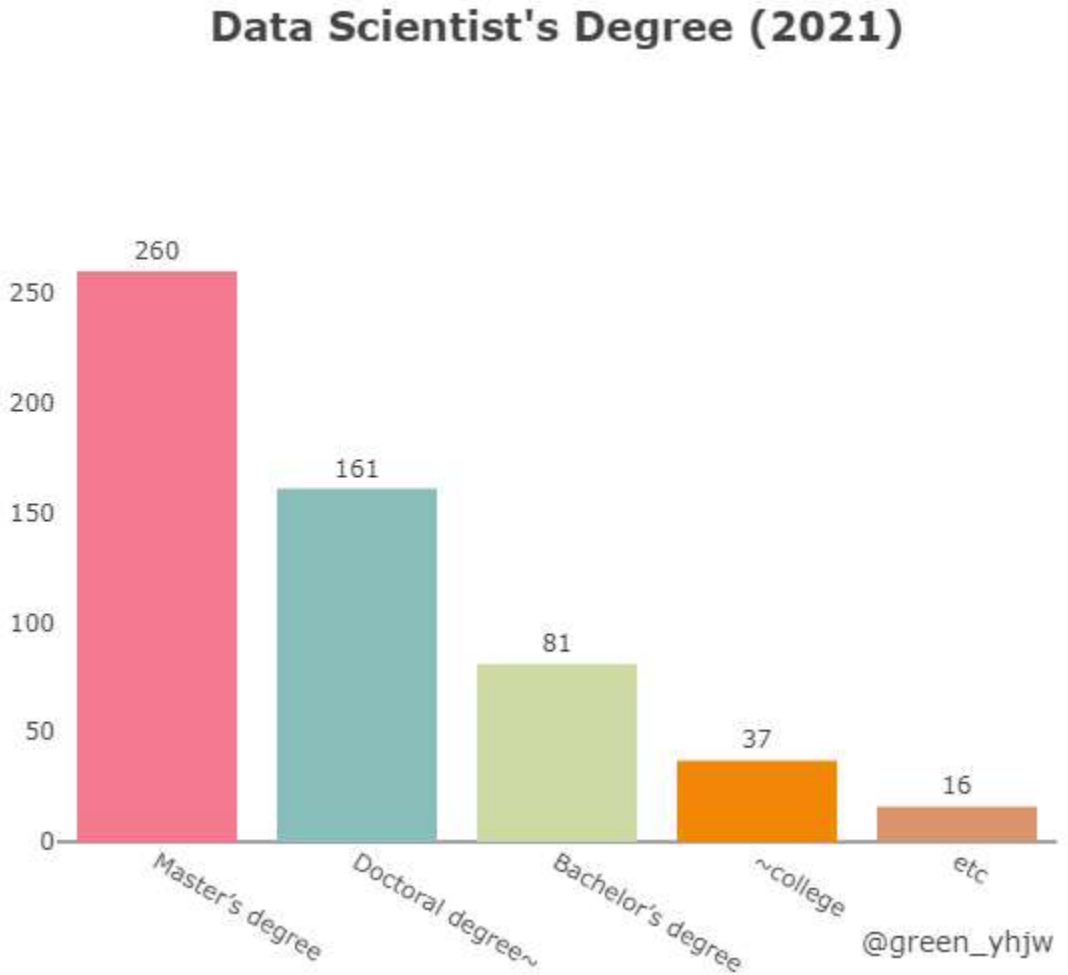
> Data Scientist의 학력 비교

석사학위(Master's Degree) 가장 높은 수치를 차지함

이어서 박사학위(Doctoral Degree),

학사 학위(Bachelor's Degree) 순으로 높은 수치를 보임

⇒ Data Scientist는 석사 학위 이상이 많은 비율을 차지함



PART 3

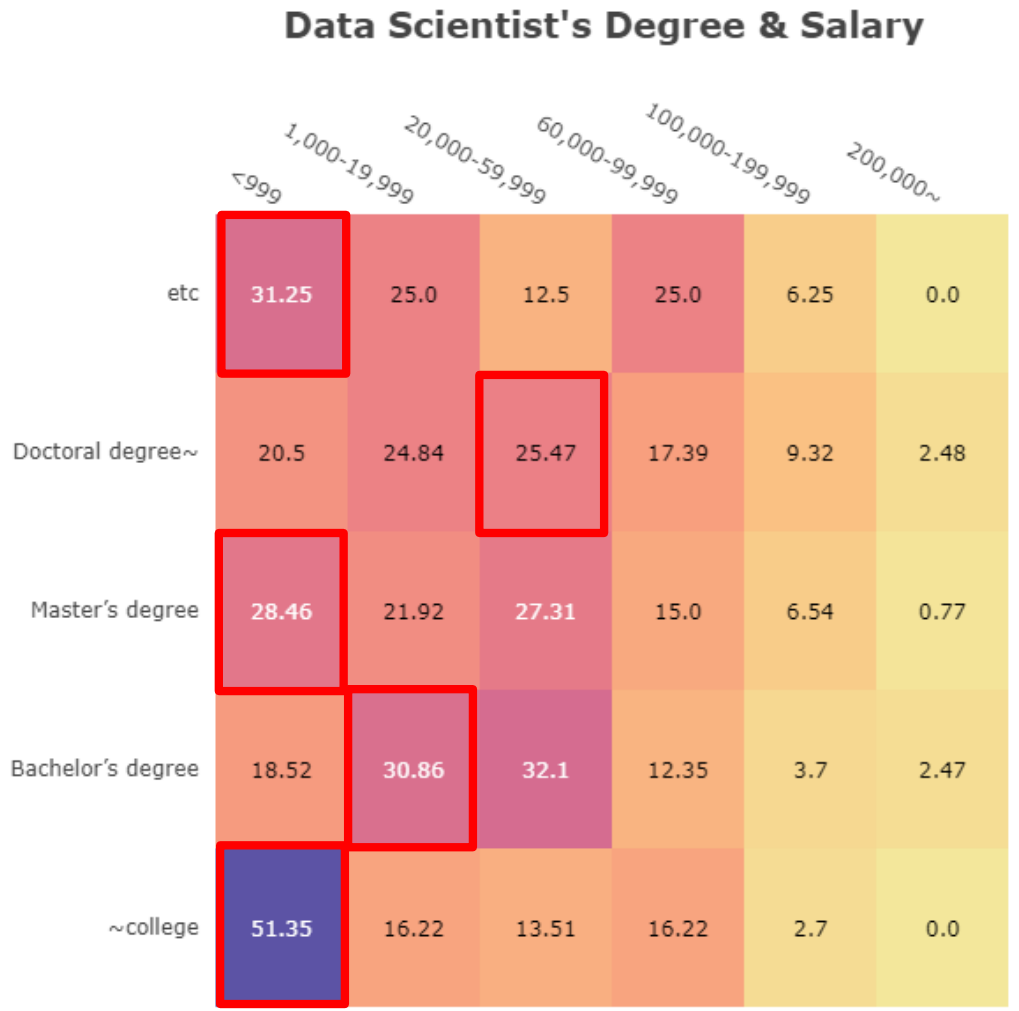
3.4 Kaggle's Transformation (Degree & Salary)

> Data Scientist의 학력와 연봉의 관계

학력이 대졸(college)이하 일 경우
: 999달러 이하
가장 낮은 연봉이 가장 높은 비율을 차지함

학사학위(Bachelor's degree),
석사학위 (Master Degree),
박사학위(Doctoral degree)
:2-6만달러 가 많은 비율을 차지함

⇒ 학력이 높을 수록 대체적으로 연봉도 높아짐



PART 3

3.5 Kaggle's Transformation (Language)

> Data Scientist가 많이 사용하는 언어

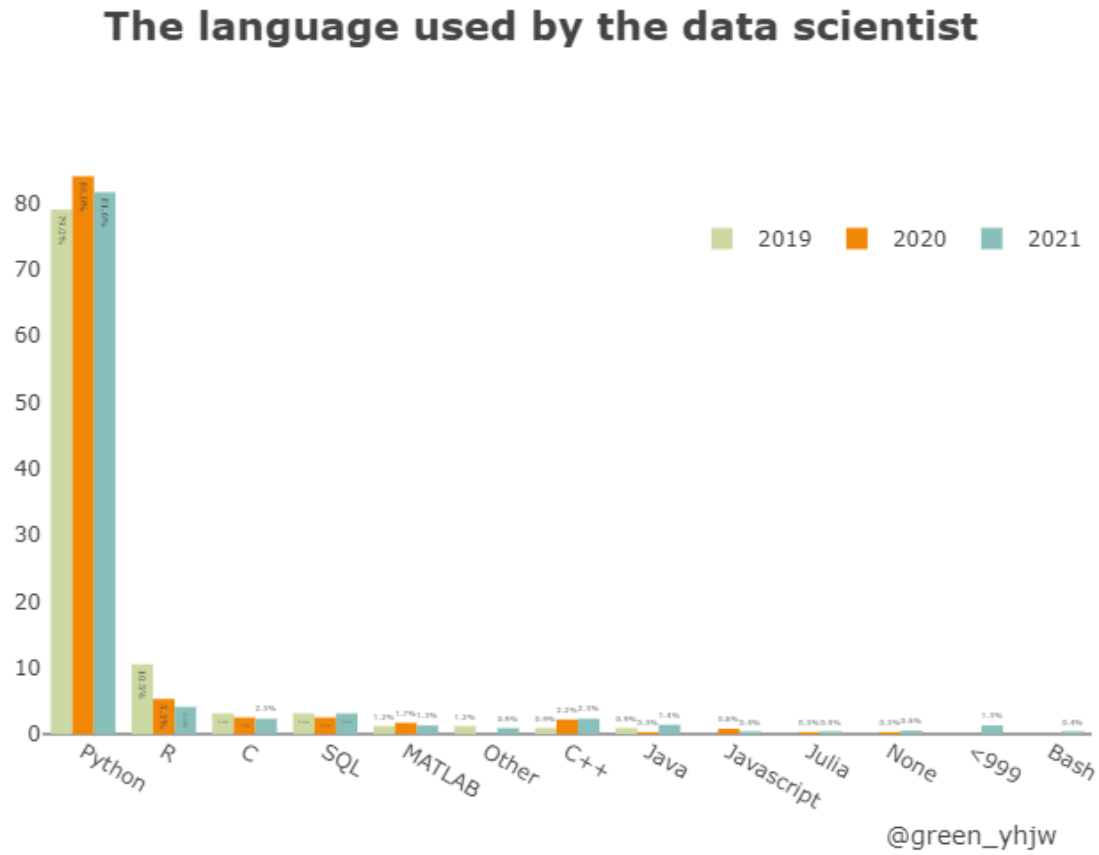
Python이 80% 이상인 가장 높은 비율을 차지함

두번째로는 R을 가장 많이 사용
19년 20년 21년 순으로 R의 사용빈도가 낮아짐
19년에서 21년에 따라 사용 비율이
10% -> 4%, 총 6%감소

세번째로 많이 사용하는 언어는 SQL
SQL은 20년도가 21년도보다 0.6 % 증가

네번째로 사용 빈도가 높은 언어는 C언어와 C++

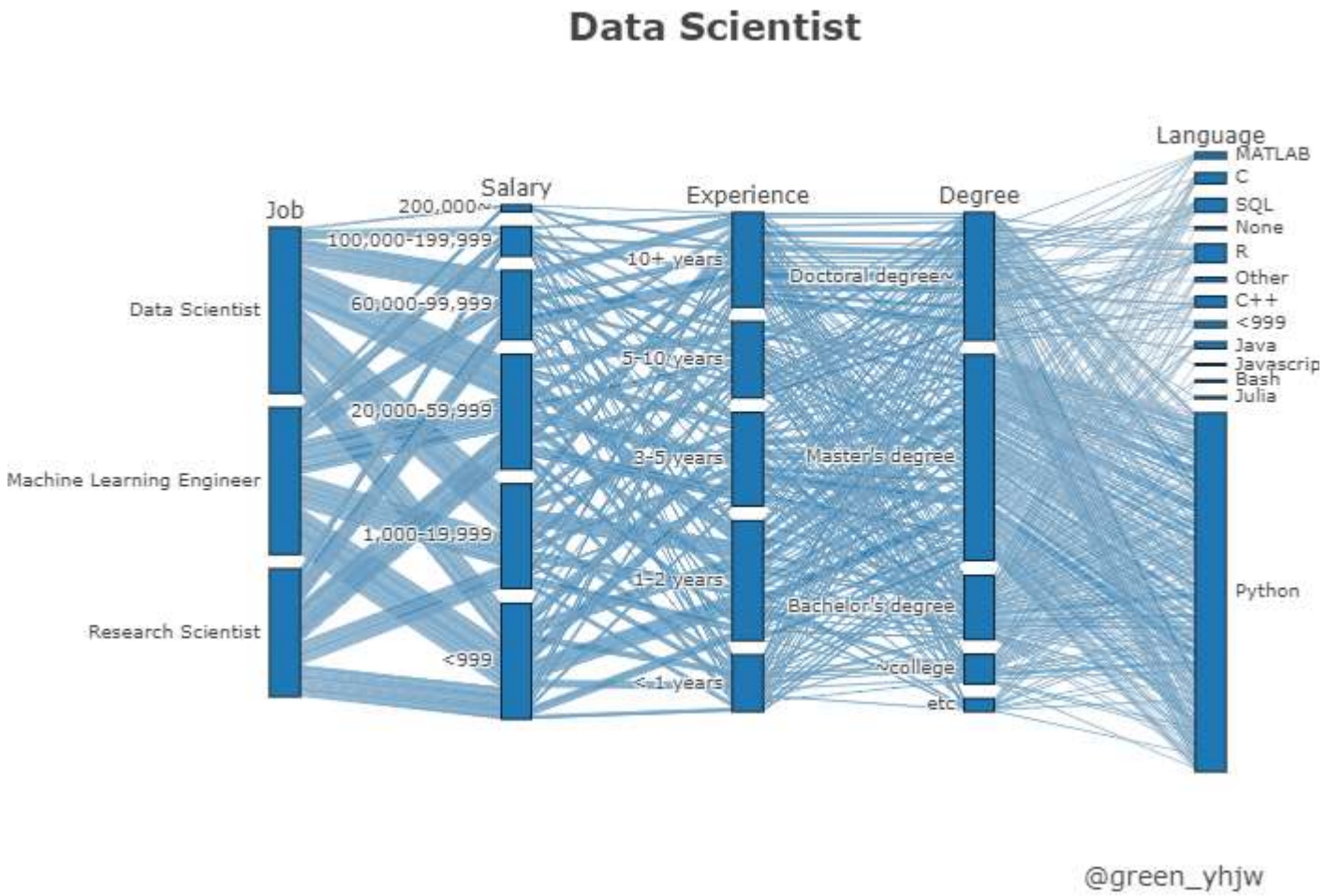
⇒ Data Scientist가 되기 위해서는
Python을 우선적으로 공부하자!



PART 3

주요 분석

3.6 Kaggle's Transformation



Parallel Categories Diagram

: 다차원 범주형 데이터 집합의 시각화

총 555명의 Data Scientist Job에 대해서 시각화 한다

카테고리의 높이가 높을수록 데이터의 발생 빈도가 증가 함을 나타낸다

04 고찰

4.1 2017 Not Employed data

4.2 Plotly Stacked scatter plot x축 설정

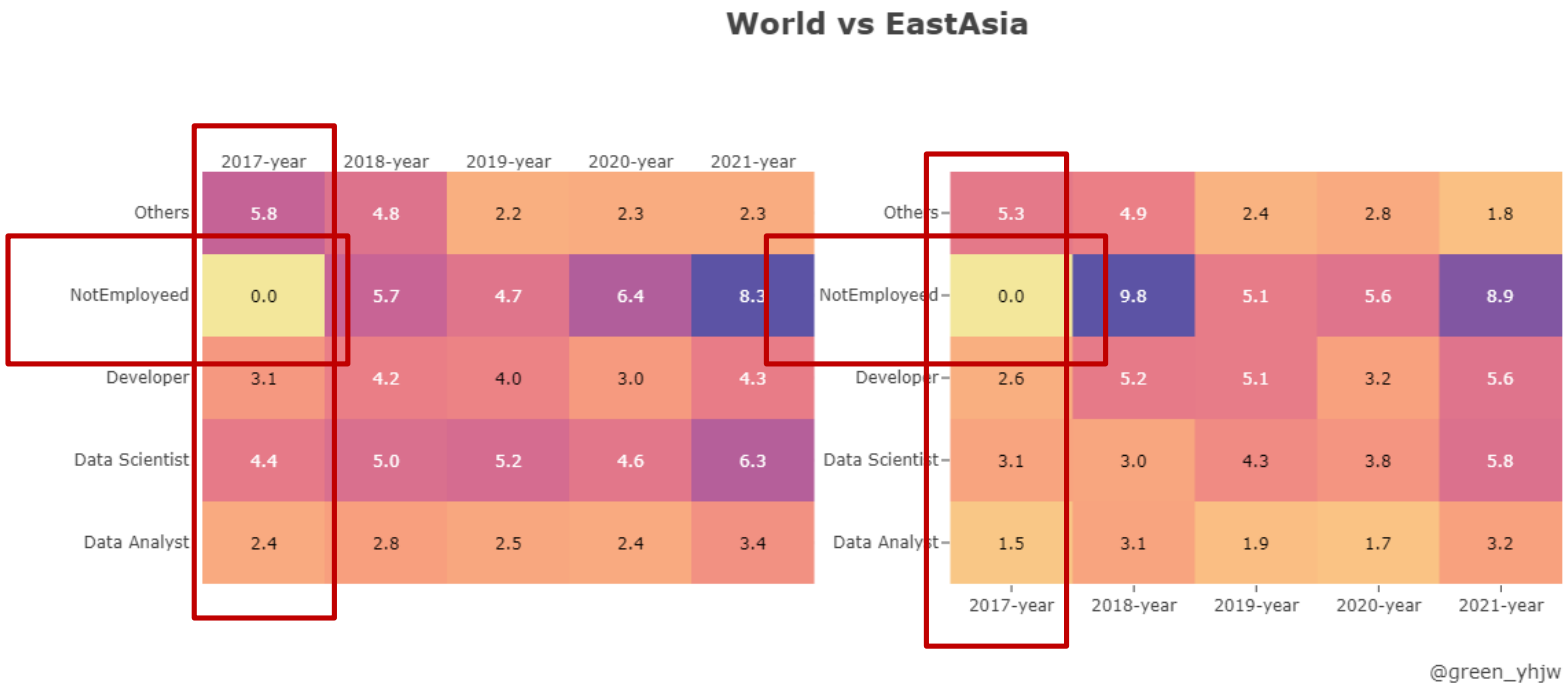
4.3 2017년 dfAge 구간 분획 후 첨가

4.4 Language x축에 None 값 발생

4.5 더 분석 해 보고 싶었던 부분

PART 4

1. 2017 Not Employed data

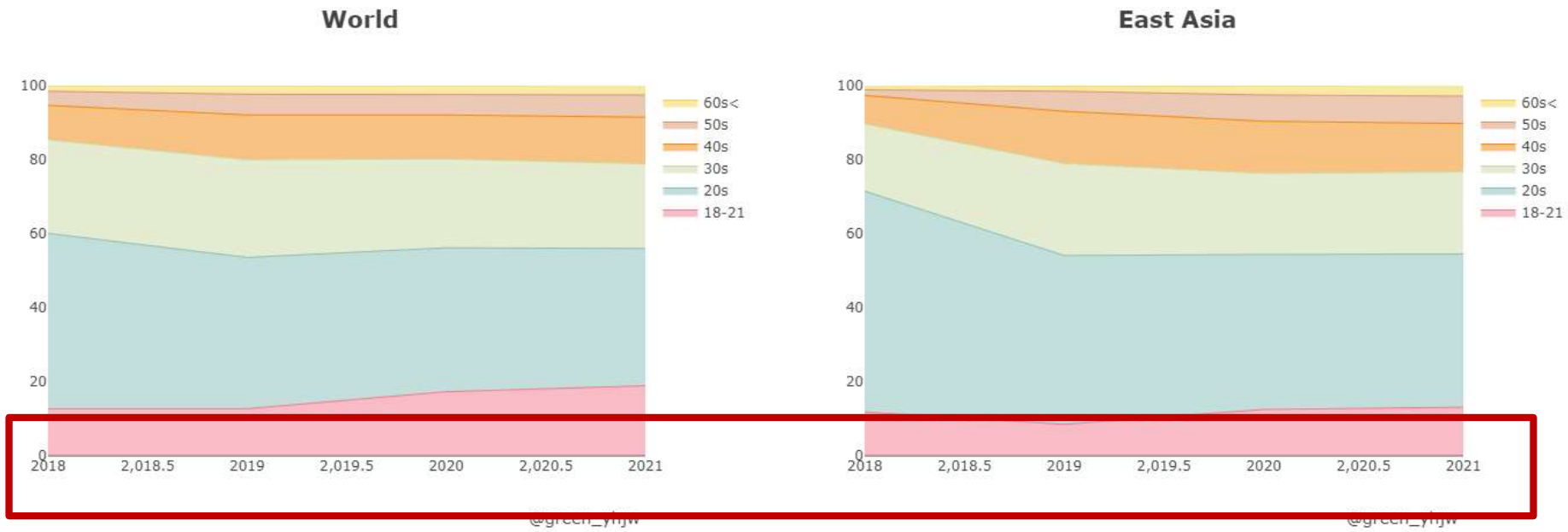


2017 Not Employed data

- ✓ 17′, 18′ data의 경우 19′~21′ data frame과 질문 위치가 다른 부분이 많다.
- ✓ 17′ data set 다시 확인 해 봐야 할 듯

PART 5

2. Plotly Stacked scatter plot x축 설정

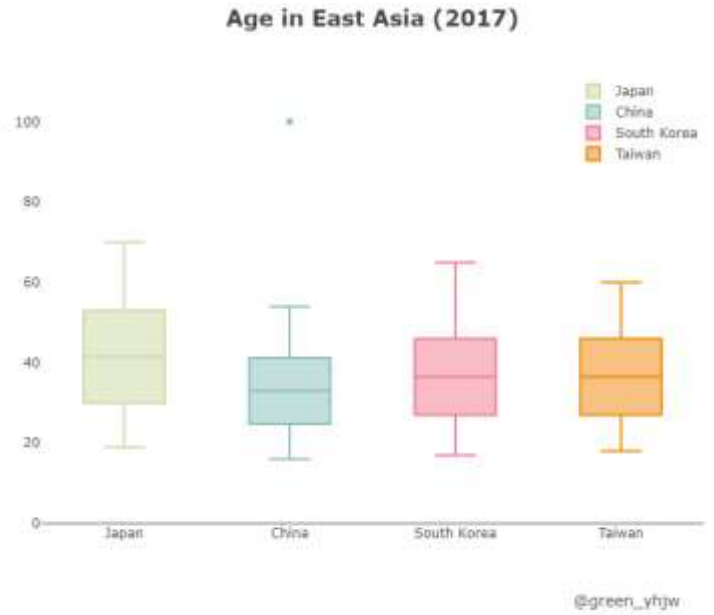


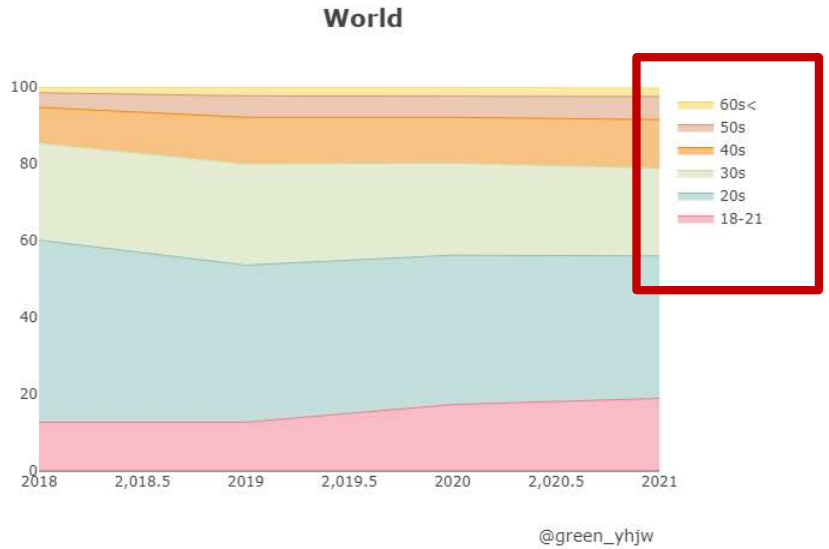
plotly

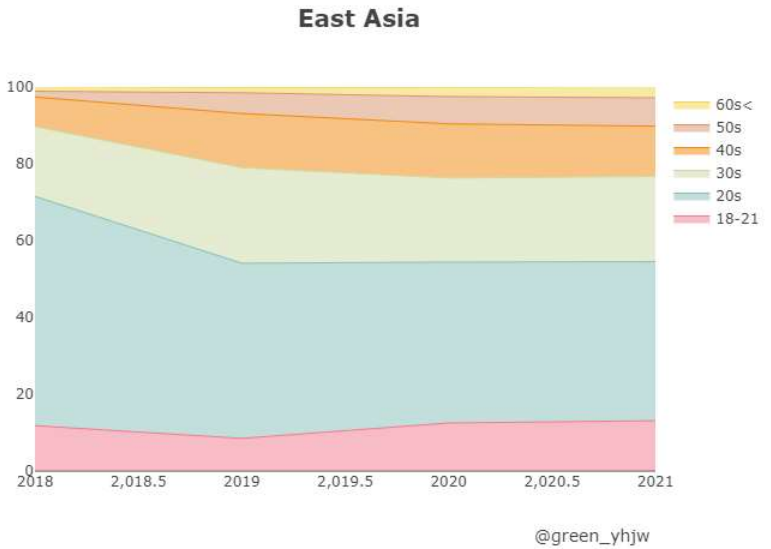
- ✓ Data를 String 형태로 집어 넣었음에도 불구하고 숫자로 인식되어 크기를 키우면 x축이 변한다.

PART 5

3. 2017년 'dfAge' 구간 분획 후 첨가





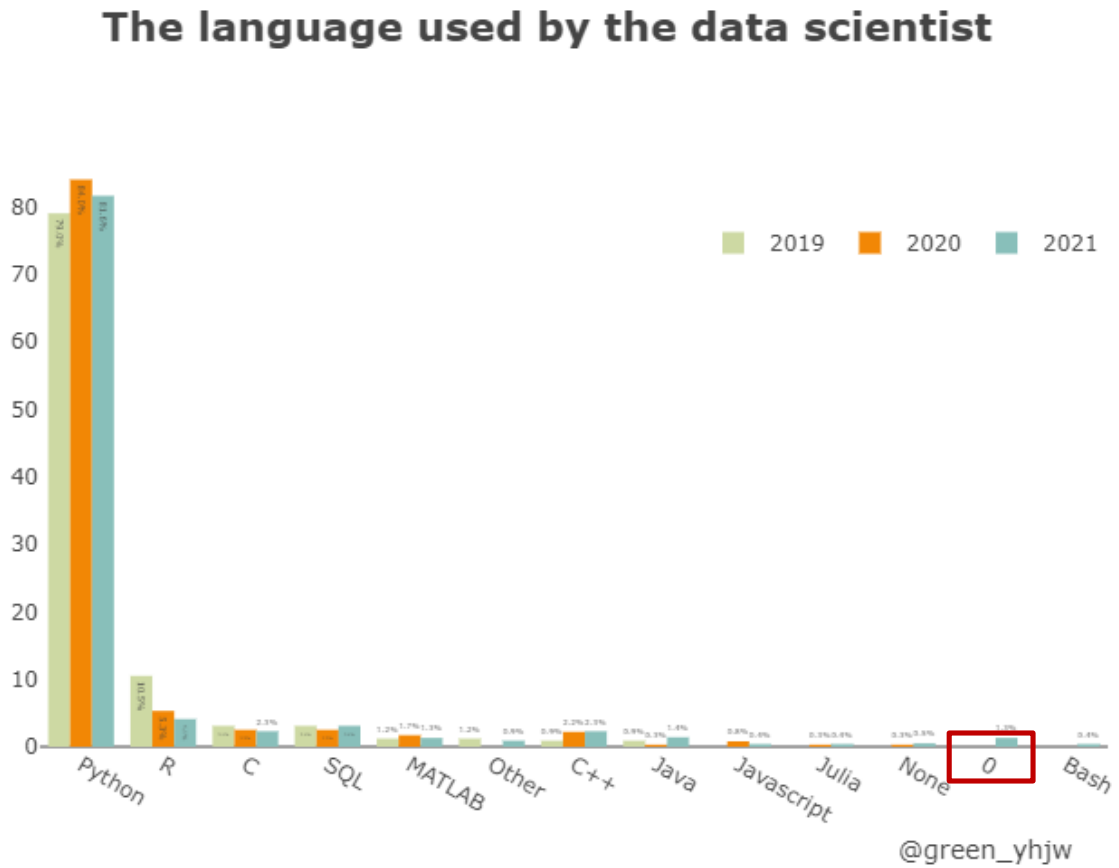


2017 East Asia 구간 설정하여 World, East Asia Stacked scatter에 집어넣기

- ✓ 시간이 충분 했다면, 2017 data를 구간(오른쪽과 같이)으로 나누어 연속 그래프에 집어 넣을 수 있었다.
- ✓ 대신 bar graph를 그려 봄으로써 2017년도의 대략적인 연령 분포를 확인 해 볼 수 있었다.

PART 5

4. Language x축에 0 값 발생



x축에 0 이라는 columns이 생긴 오류 발생 치환

위의 Swift라는 columns이 누락되었는데, 아마도 Swift의 데이터 값이라고 추측한다.
'df21_Ea_DS' 데이터 프레임 확인 필요하다.

PART 5

5. 더 분석 해 보고 싶었던 부분

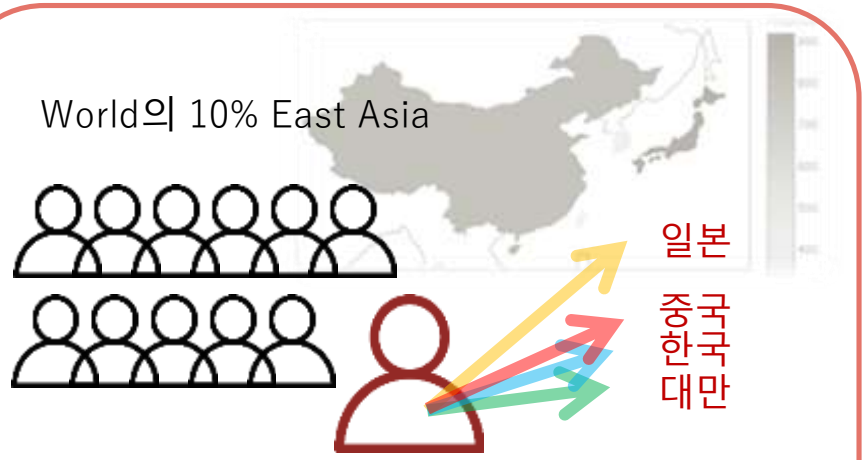
- ✓ 회사 규모에 따른 연봉 분석 (Heat Map or bar plot ,%)
- ✓ 상관관계 분석 (relationship plot, scatter)
- ✓ 나이-연봉 (TreeMap)
- ✓ WordCloud : 사용하는 IDE, Language등을 WordCloud로 만들어 보고 싶었음.

05 요약

5.1 Summary : Kaggle in East Asia
5.2 Kaggle in East Asia as Data Scientist

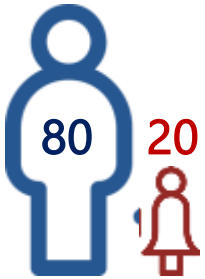
PART 5

1. Summary : Kaggle in East Asia



(22')사용자수 증가 예상

Gender

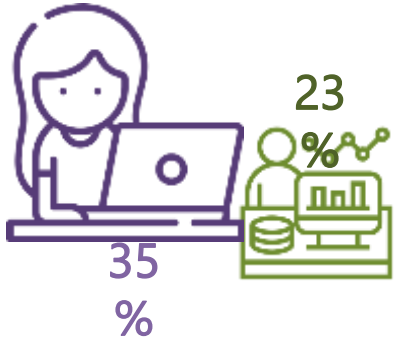


성비 80:20

Age

20~30대 70%

Job



Not Employed : 35%
Data scientist : 23%

- ✓ 18' 중국의 급격한 증가
- ✓ 학사 학위:
  $ 25,000~100,000연봉 40%
- ✓ 3-5년 경력:
  $ 25,000~100,000연봉 48%
- ✓ 가능!!

85% ~ 학사 학위

Degree

50%~ 2년 이하

Experience

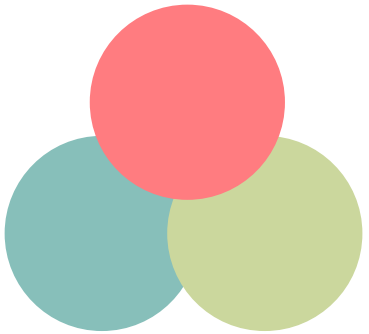
$ 25,000~60,000: 24%

Salary

Python 85%

Language

2 Kaggle in East Asia as Data Scientist



Research Scientist
Machine Learning Engineer
Data Scientist

Data Scientist

	1위구간	석사	학사	박사 이상	Python
Data Scientist	1,000-20,000$ 29.2%	2 1 3	2 1 3	2 1 3	80%~
Research Scientist	20,000-60,000$ 29.8%				
Mucnie Leanint Engineer	<900$ 31.9%				

$ 25,000~100,000 연봉

~전문 : 29%
학사 : 44%
석사 : 32%
박사~: 42%

~3y :14%
3~5 :16.9%
5~10: 28%
10~: 38%

학사 이상 10년 이상

06 참고문헌

6.1 Reference

참고 문헌

동아시아 지역	https://ko.wikipedia.org/wiki/%EB%8F%99%EC%95%84%EC%8B%9C%EC%95%84
동아시아 인구	https://ko.wikipedia.org/wiki/%EC%95%84%EC%8B%9C%EC%95%84%EC%9D%98_%EC%9D%B8%EA%B5%AC
세계 인구	https://ko.wikipedia.org/wiki/%EC%84%B8%EA%B3%84_%EC%9D%B8%EA%B5%AC https://ko.wikipedia.org/wiki/%EC%9D%B8%EA%B0%84_%EA%B0%9C%EB%B0%9C_%EC%A7%80%EC%88%98#2020%EB%85%84
동아시아 인간개발지수	https://namu.wiki/w/%EB%8F%99%EC%95%84%EC%8B%9C%EC%95%84
Data Scientist란	https://dataprofessional.tistory.com/126 https://terms.naver.com/entry.naver?docId=1691563&cid=42171&categoryId=42183
Kaggle이란 Python이란	https://ko.wikipedia.org/wiki/%EC%BA%90%EA%B8%80 https://ko.wikipedia.org/wiki/%ED%8C%8C%EC%9D%B4%EC%8D%AC
flaricon	Icons made by Freepik from www.flaticon.com
Kaggle competition Ref.	https://www.kaggle.com/miguelfzzz/the-typical-kaggle-data-scientist-in-2021 https://www.kaggle.com/desalegngeb/how-popular-is-kaggle-in-africa

Thank you

2021 Kaggle Machine Learning & Data Science Survey

Team: Green_YHJW

김지원 박윤화