Analysis 3: Analysis of the FWA in public sector organizations using data mining techniques [in Korean]

Sophisticated policy design is required for the public sector flexible work arrangement(FWA) to achieve the original purpose of work-family balance. In addition, the use of FWA due to external pressure (such as department evaluation of penalizing for not choosing to use FWA) rather reduces the adoption rate in a long-term perspective (i.e. stigmatization or psychological issue). Therefore, it is necessary to design a meticulous personnel system that facilitates the adoption through both psychological and material incentives.

Sangwon Ju, SNU GSPA

Dec/20/2021

00/

```
%%capture
!pip install nltk
!pip install konlpy
```

```
%%capture
!pip install -U pandas-profiling
```

```
%%capture
!pip install seaborn
```

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
import plotnine as pn
from plotnine import *
```

```
!pip install jsonlines
```

```
Requirement already satisfied: jsonlines in /usr/local/lib/python3.7/dist-packages (3.0.0)
Requirement already satisfied: attrs>=19.2.0 in /usr/local/lib/python3.7/dist-packages (from
Requirement already satisfied: typing-extensions in /usr/local/lib/python3.7/dist-packages (
```

```
import requests
import json
import jsonlines
import re
import pprint
```

```
%%capture
!sudo apt-get install -y fonts-nanum
!sudo fc-cache -fv
!rm ~/.cache/matplotlib -rf

# jupyter notebook
%matplotlib inline

# unicode minus           (minus     )
plt.rcParams['axes.unicode_minus'] = False
plt.rcParams['font.family'] = 'NanumGothic'
```

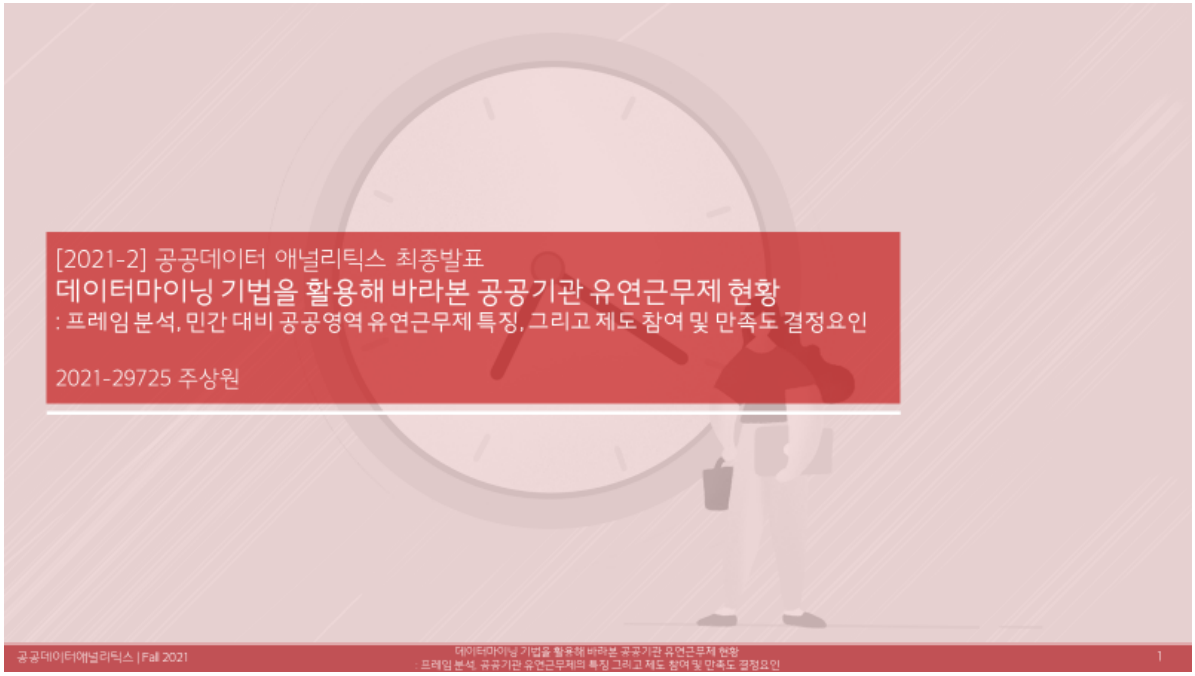
```
%%capture
!pip install dfply
from dfply import *
```

```
from google.colab import auth
auth.authenticate_user()

import gspread
from google.auth import default
creds, _ = default()
```

```
gc = gspread.authorize(creds)
```

```
from IPython.display import Image
```

```
Image(filename='/1.png')
```




```
Image(filename='/2.png')
```

01/ Case Intro
– 연구 배경 및 유연근무제 소개

"공공조직 노동생산성 향상"

- OECD(2019) 36개 국가들의 연간 평균 노동시간 1,746시간
  대한민국 근로시간 2,024시간으로 평균보다 278시간 더 근무.
- 하지만 시간당 생산성의 경우 36개 국가들 중 29위로 확인 됨.

- 노동 집약적 업무구조에서 지식 집약적 업무로의 수요 변화

"Work-Life Balance"

- 우리나라 공공기관 근로자들의 전통적인 저효율 장시간의 근로 방식은 일과 가정 양립에 부정적 영향을 미쳐왔을 (한국노동연구원, 2011) 뿐만 아니라 지식기반 사회로의 변화에 대응하는데 있어 많은 제약(황순옥·한상일, 2012)을 부여하였음.
- 일가정양립(Work-Life Balance)과 성과강조 (Result-Oriented) 문화의 확산(박한준, 2013)의 수단으로서 공공기관의 유연근무제도 도입의 당위성

근로자가 자신의 **근무시간, 방식, 장소**등을 선택하는 재량권을 부여하여 (안세연·김효선, 2019; 강현주, 2018)

(1) 일과 삶의 균형에 기여하고
(2) 공직생산성 향상을 높이는 것 목표
(인사혁신처, 2021)

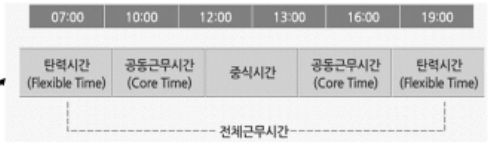

```
Image(filename='/3.png')
```

01/ Case Intro
– 유연근무제의 세부 유형 및 현황

유형	세부형태	개념
시간제근무		➤ 주 40시간보다 짧은 시간 근무
탄력근무제		➤ 주 40시간 근무하되, 출퇴근시각 · 근무시간 · 근무일을 자율 조정
	시차출퇴근형	➤ 1일 8시간 근무체제 유지 - 매일 같은 출근시각(07:00~10:00 선택) - 요일마다 다른 출근시각(07:00~10:00 선택)
	근무시간선택형	➤ 1일 4~12시간 근무 ➤ 주 5일 근무
	집약근무형	➤ 1일 10~12시간 근무 ➤ 주 3.5~4일 근무
	재량근무형	➤ 출퇴근의무 없이 프로젝트 수행으로 주 40시간 인정 ※고도의 전문적 지식과 기술이 필요해 업무수행 방법이나 시간배분을 담당자의 재량에 맡길 필요가 있는 분야
원격근무제		➤ 특정한 근무장소를 정하지 않고 정보통신망을 이용하여 근무
	재택근무형	➤ 사무실이 아닌 자택에서 근무
	스마트워크 근무형	➤ 자택 인근 스마트워크센터 등 별도 사무실 근무

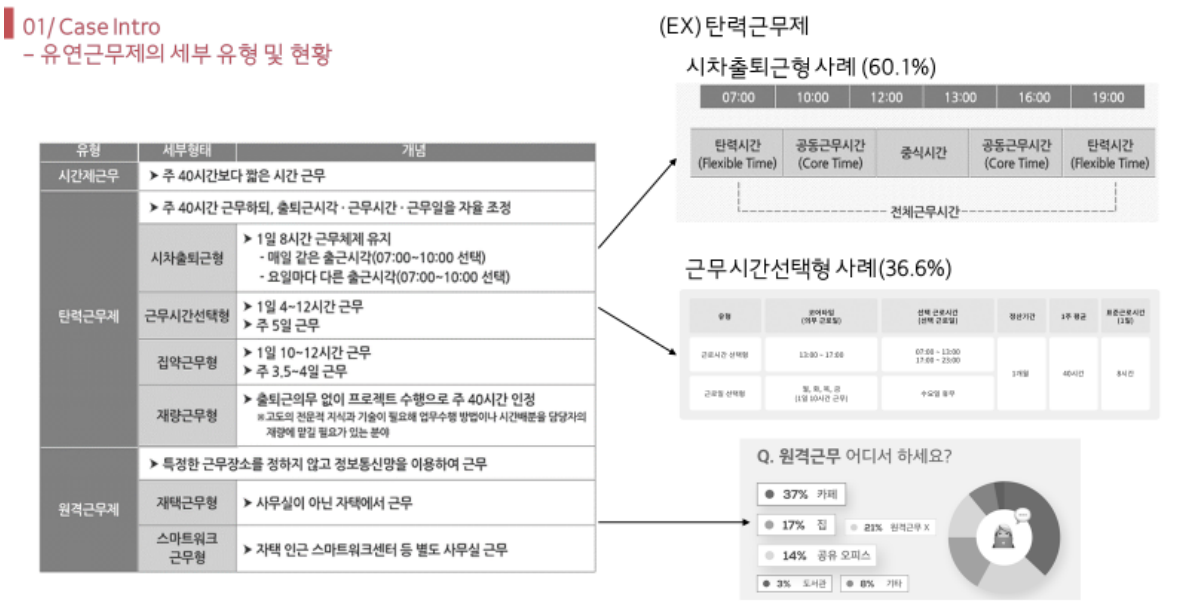
(EX) 탄력근무제

시차출퇴근형 사례 (60.1%)



근무시간선택형 사례(36.6%)

```
Image(filename='/4.png')
```




```
Image(filename='/5.png')
```

01/ Case Intro
– 공공영역 유연근무제 선행 연구

공공기관 유연근무제에 대한 연구 유형

- 국내 외 유연근무제 연구는 세부적으로 (1) 유연근무제의 효과, (2) 유연근무제 선택(참여)의 영향요인, (3) 유연근무제 실태 및 개선 방안으로 나누어진다

유연근무제 연구 흐름

- 초기연구들에서는 **유연근무제가 업무 효율성과 동기부여 정도에 긍정적 영향을 미침을 확인**할 수 있었고, **직무 만족도나 조직 몰입에 긍정적인 영향을 미쳐 조직의 성과와 목표를 달성하는데 긍정적인 영향**을 미쳤음을 확인할 수 있었다. (Caillier, 2012; Overmyer, 2011; Wadsworth, Facer and Arbon, 2010)
- 그러나, **유연근무제의 성격이나 연구 집단의 속성에 따라 결과변수에 부정적인 영향을 줄 수 있음을 확인하였다.** (Kim & Wiggins, 2011; Saltzstein et al., 2001) Greta et al. (2018)의 메타연구에 따르면 **원격근무제가 일과 생활의 통합으로 인한 스트레스**(Rau and Hyland, 2002), **사회적 고립이 조직 몰입에 방해요인으로 작용할 수 있음을 제시**하였다(Kurland and Cooper, 2002; Golden and Veiga 2008).
- 최근에는 제도의 성공적인 도입에 필요한 맥락을 파악하기 위해 여러 상황적 요인들을 조절변수로 활용한 연구가 진행되거나 (Choi, 2017). **조직 및 문화적 부작용(Backlash)에 의해 유연근무제를 꺼려하는**(Caillier, 2013; Kwon and Jeon, 2018) **원인을 확인하기 위해 선행요인들이 참여와 이탈에 어떤영향을 미치는지**(Kwon et al, 2019)에 **대한 연구가** 진행되었다.


```
Image(filename='/6.png')
```

01/ Case Intro
– 공공영역 유연근무제 논의의 필요성

- 전반적으로 공공영역 유연근무제의 사용이 **일과 삶의 균형에 긍정적인 영향을 미치고 있음이 확인되고 있고**(Mullins, 2020), **COVID-19 확산방지를 위해 정부에서 장려하면서 유연근무제에 대한 사회적 공감대가 형성되었다는 점에서 제도의 정착이 된 것**으로 보인다.
- 그러나, **일과 삶의 균형수준을 향상**시키기 위해서는 **제도의 양적인 확대하는** 것만이 중요한 것이 아니라 조직문화가 유연근무제에 적합한가에 대한 논의(민경률·박성민, 2013)가 필요하다는 점에서 **사용확산에 대한 논의를 뿐만 아니라 사용자들이 만족할 수 있는 유연근무제에 대한 논의**가 필요함을 확인할 수 있다.

```
Image(filename='/7.png')
```

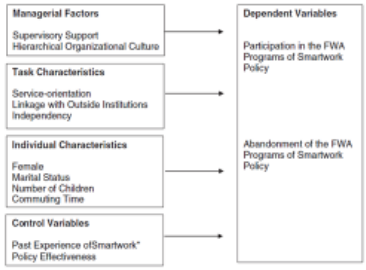
02/ Literature Review
– 선행연구1 (Kwon, Cho and Song, 2020): 선행요인

How do managerial, task, and individual factors influence flexible work arrangement participation and abandonment?

Research Question

How do **managerial factors, task factors, and individual characteristics** influence public employee participation in and abandonment of FWA (Flexible work arrangement)?

Framework: Social Exchange / Organization Justice Theory



Hypothesis

Managerial Support
Hypothesis 1: Supervisory support for WLB
➔ More likely to participate in FWA / less likely to abandon FWA
Hypothesis 2: Hierarchical organizational culture
➔ Less likely to participate in FWA / more likely to abandon FWA

Task Characteristics
Hypothesis 3: Service-oriented
➔ Less likely to participate in FWA / More likely to abandon FWA
Hypothesis 4: Public employees linked with outside institutions
➔ Less likely to participate in FWA / More likely to abandon FWA
Hypothesis 5: Public employees a high level of independence
➔ More likely to participate in FWA / Less likely to abandon FWA

Individual Characteristics
Hypothesis 6: Females, Married, More children, Longer commuting
➔ More likely to participate in FWA / Less likely to abandon FWA

Control Variable: Policy effectiveness and Past experience of FWA

Research Design

2016 SPEPS responses of public officials in the Korean government (11 government agencies)


```
Image(filename='/8.png')
```

02/ Literature Review
– 선행연구1 (Kwon, Cho and Song, 2020): 선행요인

How do managerial, task, and individual factors influence flexible work arrangement participation and abandonment?

Results: Logistic Regression

Table 5 Logit regression results

	FWA participation								FWA abandonment							
	통제변수만		+ 개인속성		+ 업무속성		+관리자속성		통제변수만		+ 개인속성		+ 업무속성		+관리자속성	
	Odds ratio	Mean effect	Odds ratio	Mean effect	Odds ratio	Mean effect	Odds ratio	Mean effect	Odds ratio	Mean effect	Odds ratio	Mean effect	Odds ratio	Mean effect	Odds ratio	Mean effect
Past Experience of Smartwork	53.78*** (27.76)	.52	74.18*** (43.05)	.53	121.6*** (79.47)	.54	157.1*** (110.4)	.55								
Policy Effectiveness	1.83*** (.28)	.36	2.05*** (.35)	.39	2.22*** (.40)	.38	2.47*** (.50)	.42	.72** (.05)	−.30	.79** (.05)	−.31	.71** (.09)	−.28	.71** (.09)	−.28
Female			.67 (.28)	−.03	.55 (.26)	−.03	.49 (.24)	−.04			1.84* (.63)	.08	1.89* (.67)	.09	1.87* (.67)	.08
Marital Status			.23* (.14)	−.14	.16** (.11)	−.15	.15** (.11)	−.14			3.43** (1.57)	.15	3.37** (1.56)	.14	3.23* (1.52)	.13
Number of Children			2.03** (.54)	.57	1.96* (.56)	.47	2.03* (.61)	.48			.93 (.19)	−.05	.99 (.21)	−.01	1.00 (.21)	.00
Commuting Time			1.70** (.28)	.55	1.96*** (.35)	.67	2.07*** (.39)	.71			.72* (.12)	−.16	.70* (.12)	−.16	.69* (.12)	−.16
Service Orientation					.93 (.11)	−.03	.94 (.11)	−.02					.92 (.08)	−.07	.90 (.08)	−.08
Linkage with Outside Institutions					.67** (.10)	−.23	.69* (.11)	−.19					1.13 (.12)	.08	1.11 (.12)	.07
Independency					1.49** (.207)	.16	1.53** (.222)	.15					.83* (.080)	−.15	.81* (.080)	−.17
Supervisory Support							1.52* (.33)	.10							.78* (.12)	−.23
Hierarchical Culture							.51* (.14)	−.48							1.39* (.26)	.20


```
Image(filename='/9.png')
```

02/ Literature Review
– 선행연구1 (Kwon, Cho and Song, 2020): 선행요인

How do managerial, task, and individual factors influence flexible work arrangement participation and abandonment?

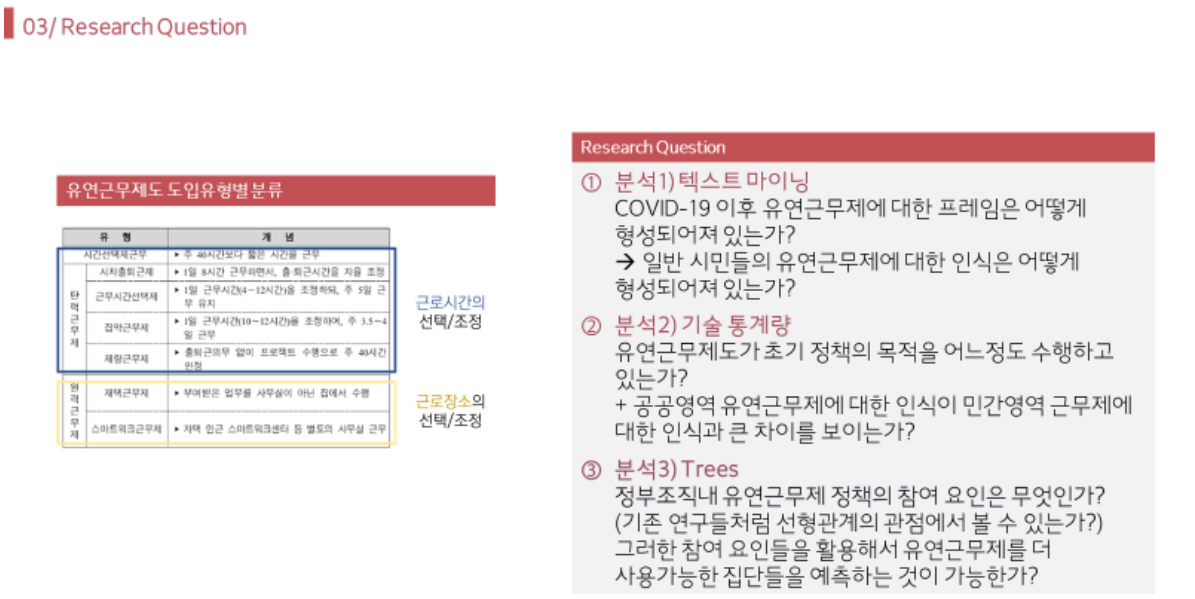
결론

1. 공공부문 유연근무제 과거 참여 경험은 유연근무제에 대한 지속적 참여로 이어지고 있으며, 유연근무제에 대한 긍정적인 인식은 참여도를 높이고 있다.
2. 성별, 혼인 여부, 자녀 수, 출퇴근 시간은 유연근무제 참여에는 통계적으로 유의한 영향을 미치고 있음. 업무가 외부와 연결되어 있을 경우 유연근무제 참여도가 감소하였고 업무 자유도가 높을 경우 유연근무제 참여도가 증가하였음.
3. 위계적인 문화는 유연근무제 참여도에 부정적인 영향을 미쳤으며, Supervisory support는 유연근무제 참여도에 긍정적인 영향을 미쳤음.
4. 유연근무제 포기에 관련된 회귀분석은 많은 변수들이 통계적으로 유의하지 않은 것으로 보아 이후에 연구가 더 진행되야 함.

유연근무제를 시행하는지 여부보다 직원들이 자유롭게 유연근무제를 사용할 수 있는지 여부가 더 중요하다면, 어떤 요인들이 유연근무제 사용에 영향을 미치는지 연구하는 것은 중요함.

➔ *Managerial, task, individual factor 모두 유연근무제 사용에 영향을 미침.*

```
Image(filename='/10.png')
```



03/ Analysis 1 – Text Mining – COVID-19 ?

?

COVID-19 (Big Kinds)

. .

-

1. (Count) - BoW: - DTM (Document-Term Matrix): , - **TF-IDF (Term Frequency - Inverse Document Frequency):** - > TF: D T DF: T IDF: DF . ln(/(DF)) TF-IDF: TF * IDF (TF) 2. - (Co-Occurrence) . “ ” . (, ,) . 3. (Association Rules) - Apriori Algorithm(Agrawal et al., 1993):

(DTM) (Latent) . (Blei, 2014; Blei and Lafferty, 2007;Blei, Ng and Jordan, 2003) DTM - , - LDA Singular Value Decomposition .

LDA(Latent Dirichlet Allocation): ? PLSA PLSA . LDA , . , . 1 ( ) . 1 . delta , beta

fyi) (Dirichlet distribution): . (K 2) LDA , k (p_k) . Multinomial . dirichlet k 1 , multinomial (p_k) -> .

$$f(x_1, x_2, \cdots, x_K) = \frac{1}{\mathrm{B}(\alpha)} \prod_{i=1}^{K} x_i^{\alpha_i - 1}$$

$$\mathrm{B}(\alpha) = \frac{\prod_{i=1}^{K} \Gamma(\alpha_i)}{\Gamma(\sum_{i=1}^{K} \alpha_i)}$$

:

$$\sum_{i=1}^{K} x_i = 1$$

https://donghwa-kim.github.io/distributions.html ( )

COVID-19 2020 2 2021 12 ‘ ’ , ’ , ’ , ’ , 11,334 (BIGKINDS) . ( )

a.

```
worksheet = gc.open('NewsResult_20200201-20211202 (1)').sheet1
rows = worksheet.get_all_values()
```

```
bigkinds=pd.DataFrame.from_records(rows)
bigkinds.columns=bigkinds.iloc[0]
bigkinds=bigkinds.drop(bigkinds.index[0])
```

```
import pandas_profiling
bigkinds.profile_report()
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]

Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]

Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]

<IPython.core.display.HTML object>
```

b.

-

```
from datetime import datetime
number=bigkinds >> group_by(X. ) >> summarize(count=n(X. ))
number.head(10)
```

		count
0	20200201	1
1	20200202	1
2	20200203	5
3	20200204	8
4	20200205	7
5	20200206	48
6	20200207	15
7	20200208	2
8	20200209	3
9	20200210	3

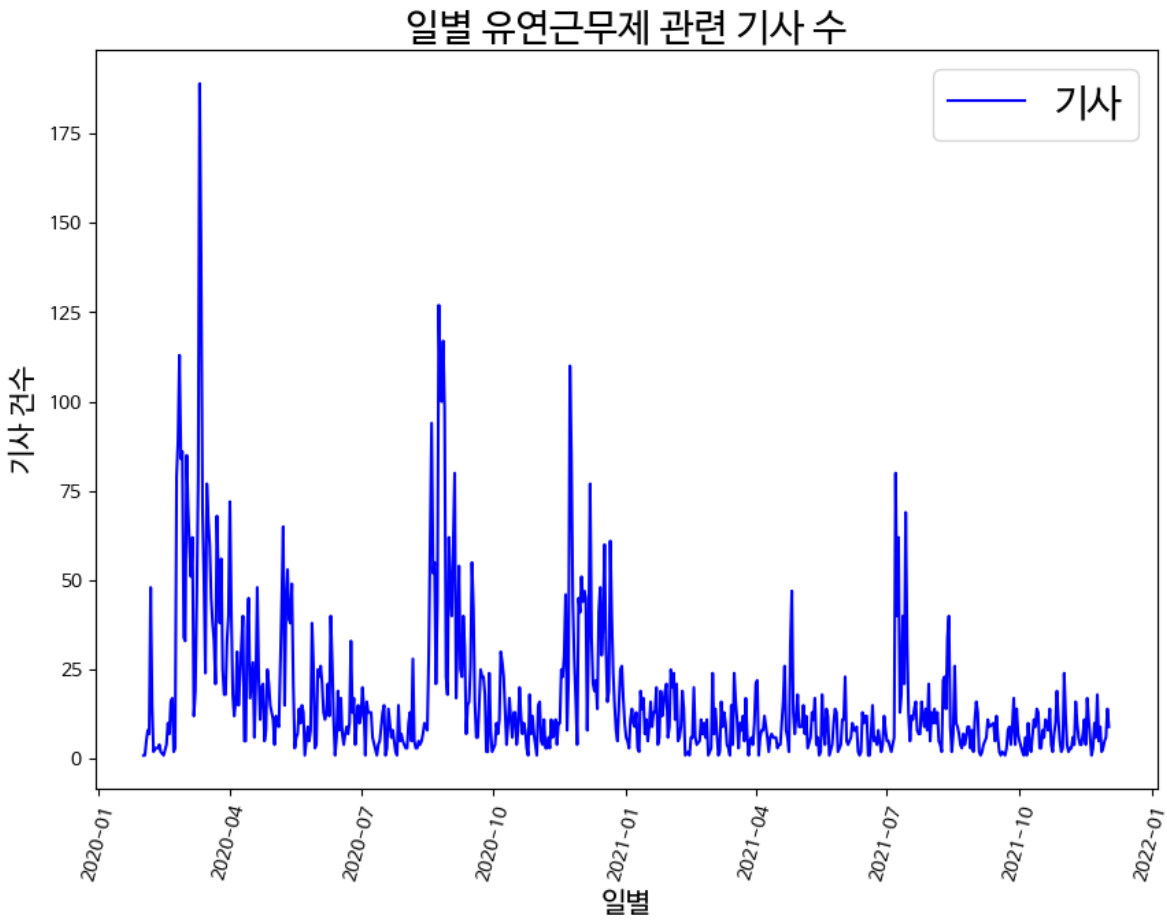
```
# list
kdate = [ datetime.strptime(d, '%Y%m%d') for d in number[" "] ]

head(kdate)
```

```
number['date'] = kdate
```

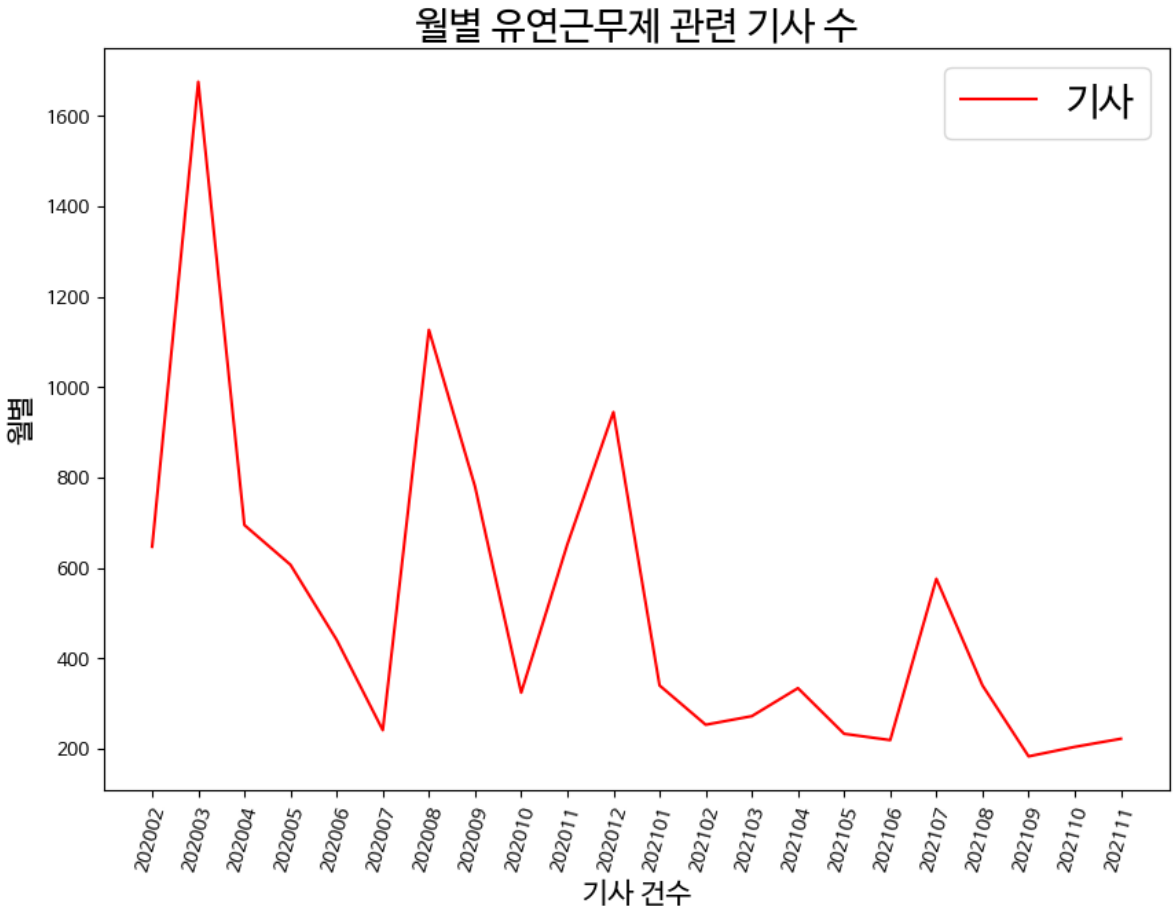
```
kdate1 = [datetime.strftime(d, '%Y%m') for d in number["date"] ]

number['date1'] = kdate1
number.head(10)
number2=number[number. .astype("int64")<20211131] >> select(["date1","count"])  >> rename(
```

```
fig=plt.figure(figsize = (10,7))

plt.plot(number["date"], number["count"],color='blue', label=str(" "))
plt.title("           ",fontsize=20)
plt.style.use("default")
plt.rc('font', family='NanumBarunGothic')

plt.legend(fontsize=20)
plt.xticks(rotation=75,fontsize=10)
plt.yticks(fontsize=10)

ax = fig.add_subplot(1, 1, 1)
ax.set_xlabel(" ", fontsize="15")
ax.set_ylabel("   ", fontsize="15")
plt.show()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:12: MatplotlibDeprecationWarning
  if sys.path[0] == '':
```




```
fig2=plt.figure(figsize = (10,7))
ax = fig2.add_subplot(1, 1, 1)
ax.set_xlabel("    ", fontsize="15")
ax.set_ylabel(" ", fontsize="15")

plt.plot(number2["date1"], number2["num"],color='red', label=str(" "))
plt.title("           ",fontsize=20)
plt.style.use("default")
plt.rc('font', family='NanumBarunGothic')

plt.legend(fontsize=20)
plt.xticks(rotation=75,fontsize=10)
plt.yticks(fontsize=10)
plt.show()
```



•

```
#
article=bigkinds >> select(" ") >> rename(title=" ")
print(article["title"].head(10))
article["title"]=article["title"].astype(str)
article.shape #11334
```

```
1                                    ,
2                                "     "
3                      ?    5    ,
4                                [  ]
5                                 MZ     [ 360]
6                            "  ?            ?"
7                    5000         "        "
```

```
8                                5       ' '
9     "           "               [   ...
10                       ,
Name: title, dtype: object
```

```
(11334, 1)
```

```
def text_preprocess(text):
    """

    1. span tag
    2. br tag
    3.  ,  ,  ,
    4.
    """
    text = re.sub("(<span class='quot[0-9]'>|</span>|<br/>|<br />|([^0-9 - A-Za-z. ]))","",
    return [sen.strip() for sen in text.split('.') if sen.strip()]
```

```
df = pd.DataFrame(index=np.arange(1,article.shape[0]+1), columns=['title'])
```

```
#
for i in np.arange(1,article.shape[0]+1):
  df["title"][i]=text_preprocess(article["title"][i])
```

```
df
```

	title
1	[ ]
2	[ ]
3	[5]
4	[ ]
5	[MZ 360]
...	...
11330	[ ]
11331	[ ]
11332	[ ]
11333	[ ]
11334	[6 2]

```
sentence_arr = []
for i in np.arange(1,article.shape[0]):
    text=df.title[i][0]
    sentence_arr.insert(0,text)
```

```
pprint.pprint(sentence_arr[1:10])
```

```
['                    ',
 '                     ',
 '                       ',
 '                    ',
 '                     ',
 '      2 4          20           ',
 '     16             42      ',
 '       1              ',
 '                      ']
```

```
check = [' ', ' ', ' ']
matching1 = [s for s in sentence_arr if " " in s]
matching2 = [s for s in sentence_arr if " " in s]
matching3 = [s for s in sentence_arr if " " in s]
matching =matching1 + matching2 + matching3
matching=list(set(matching))
```

```
print(len(matching))
type(matching)
# 1226
```

```
1226
```

```
list
```

NLP (morpheme) (morpheme tokenization) .
Okt . : https://wikidocs.net/21698

```
from konlpy.tag import *
```

```
kkma = Kkma()
okt = Okt()
```

- OKT

```
okt_list=[]
for title in matching:
  tokenized3=okt.pos(title)
  okt_list.insert(len(okt_list)+1,tokenized3)
```

```
okt_list[:3]
```

```
[[(' ', 'Noun'),
  ('19', 'Number'),
  (' ', 'Noun'),
  (' ', 'Josa'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  ('3000', 'Number'),
  (' ', 'Noun'),
  ('3', 'Number'),
  (' ', 'Noun'),
  ('  ', 'Noun'),
  (' ', 'Noun')],
 [('  ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Josa'),
  (' ', 'Noun'),
  (' ', 'Suffix'),
  (' ', 'Josa'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Josa'),
  (' ', 'Noun')],
 [('KBS', 'Alpha'),
```

```
  (' ', 'Noun'),
  ('3', 'Number'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  ('1', 'Number'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  (' ', 'Noun'),
  ('  ', 'Noun')]]
```

```
okt_morphs_list=[]
for title in matching:
  tokenized4=okt.morphs(title)
  okt_morphs_list.insert(len(okt_morphs_list)+1,tokenized4)
```

```
okt_morphs_list[:3]
```

```
[[' ',
  '19',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  '3000',
  ' ',
  '3',
  ' ',
  '  ',
  ' '],
 ['  ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
```

```
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' '],
 ['KBS', ' ', '3', ' ', ' ', '1', ' ', ' ', ' ', ' ', ' ', '  ']]
```

•

```
kkma_list=[]
for title in matching:
  tokenized=kkma.pos(title)
  kkma_list.insert(len(kkma_list)+1,tokenized)
```

```
kkma_list[:5]
```

```
[[(' ', 'NNG'),
  ('19', 'NR'),
  (' ', 'NNG'),
  (' ', 'JKM'),
  (' ', 'NNG'),
  (' ', 'NNG'),
  (' ', 'NNG'),
  ('3000', 'NR'),
  (' ', 'NNM'),
  ('3', 'NR'),
  (' ', 'NNG'),
  ('  ', 'NNG'),
  (' ', 'NNG')],
 [('  ', 'NNG'),
  (' ', 'NNB'),
  (' ', 'NNG'),
  (' ', 'JX'),
  (' ', 'NNG'),
  (' ', 'XSV'),
  (' ', 'ECD'),
  (' ', 'NNG'),
  (' ', 'NNG'),
  (' ', 'NNG'),
  (' ', 'NNG'),
```

```
 ('  ', 'NNG')],
[('KBS', 'OL'),
 (' ', 'NNG'),
 ('3', 'NR'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 ('1', 'NR'),
 (' ', 'NNM'),
 (' ', 'MAG'),
 (' ', 'VV'),
 (' ', 'ETD'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 ('  ', 'NNG')],
[(' ', 'NNG'),
 ('LS', 'OL'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 ('  ', 'NNG'),
 (' ', 'MAG'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 ('  ', 'NNG'),
 ('  ', 'NNG')],
[('  ', 'NNG'),
 (' ', 'NNG'),
 ('  ', 'NNG'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 (' ', 'VV'),
 (' ', 'EPT'),
 (' ', 'ECE'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 (' ', 'NNG'),
 (' ', 'VV'),
 (' ', 'ECS')]]
```

```
kkma_morphs_list=[]
for title in matching:
  tokenized1=kkma.morphs(title)
  kkma_morphs_list.insert(len(kkma_morphs_list)+1,tokenized1)
```

```
kkma_morphs_list[:5]
```

```
[[' ',
  '19',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  '3000',
  ' ',
  '3',
  ' ',
  '  ',
  ' '],
 [' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', '  '],
 ['KBS',
  ' ',
  '3',
  ' ',
  ' ',
  '1',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  ' ',
  '  '],
 [' ', 'LS', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', '  '],
 [' ', ' ', '  ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ']]
```

OKT .|

- : https://www.ranks.nl/stopwords/korean

```
worksheet2 = gc.open('stopwords').sheet1
rows2 = worksheet2.get_all_values()
stopwords=pd.DataFrame.from_records(rows2)
```

```
stopwords=pd.Series.tolist(stopwords[0])
```

```
okt_morphs_list_stop=[]
for words in okt_morphs_list:
  tokenized=[]
  for word in words:
    if not word in stopwords:
      tokenized.insert(len(tokenized)+1,word)
  okt_morphs_list_stop.insert(len(okt_morphs_list_stop)+1,tokenized)
```

```
okt_morphs_list_stop[:5]
```

```
[[' ', '19', ' ', ' ', ' ', ' ', '3000', ' ', '3', ' ', '  ', ' '],
 ['  ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' '],
 ['KBS', ' ', '3', ' ', ' ', '1', ' ', ' ', ' ', ' ', ' ', '  '],
 [' ', 'LS', ' ', ' ', ' ', ' ', ' ', ' ', ' ', '  '],
 [' ', ' ', '  ', ' ', '  ', ' ', ' ', ' ']]
```

```
kkma_morphs_list_stop=[]
for words in kkma_morphs_list:
  tokenized=[]
  for word in words:
    if not word in stopwords:
      tokenized.insert(len(tokenized)+1,word)
  kkma_morphs_list_stop.insert(len(kkma_morphs_list_stop)+1,tokenized)
```

```
kkma_morphs_list_stop[:5]
```

```
[[' ', '19', ' ', ' ', ' ', ' ', '3000', ' ', '3', ' ', '  ', ' '],
 ['  ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', ' ', '  '],
 ['KBS',
  ' ',
  '3',
```

```
 ' ',
 ' ',
 '1',
 ' ',
 ' ',
 ' ',
 ' ',
 ' ',
 ' ',
 ' ',
 '  '],
[' ', 'LS', ' ', ' ', ' ', '  ', ' ', ' ', ' ', '  ', '  '],
['  ', ' ', '  ', ' ', ' ', ' ', ' ', ' ', ' ']]
```

- Word Cloud

```
from wordcloud import WordCloud, STOPWORDS
import nltk as nltk
```

```
total_okt= []
for element in okt_morphs_list_stop:
  total_okt+=element
```

```
total_okt_over2=pd.Series(total_okt)[pd.Series(total_okt).str.len()>=2]
```

```
count_okt=pd.DataFrame(total_okt).value_counts().rename_axis('unique_values').reset_index(
count_okt.columns=["word","counts"]
```

```
count_okt2=count_okt >> mask(X.word.str.len() >=2)
count_okt2=count_okt2.reset_index().iloc[:,1:]
count_okt2.head(60)
```

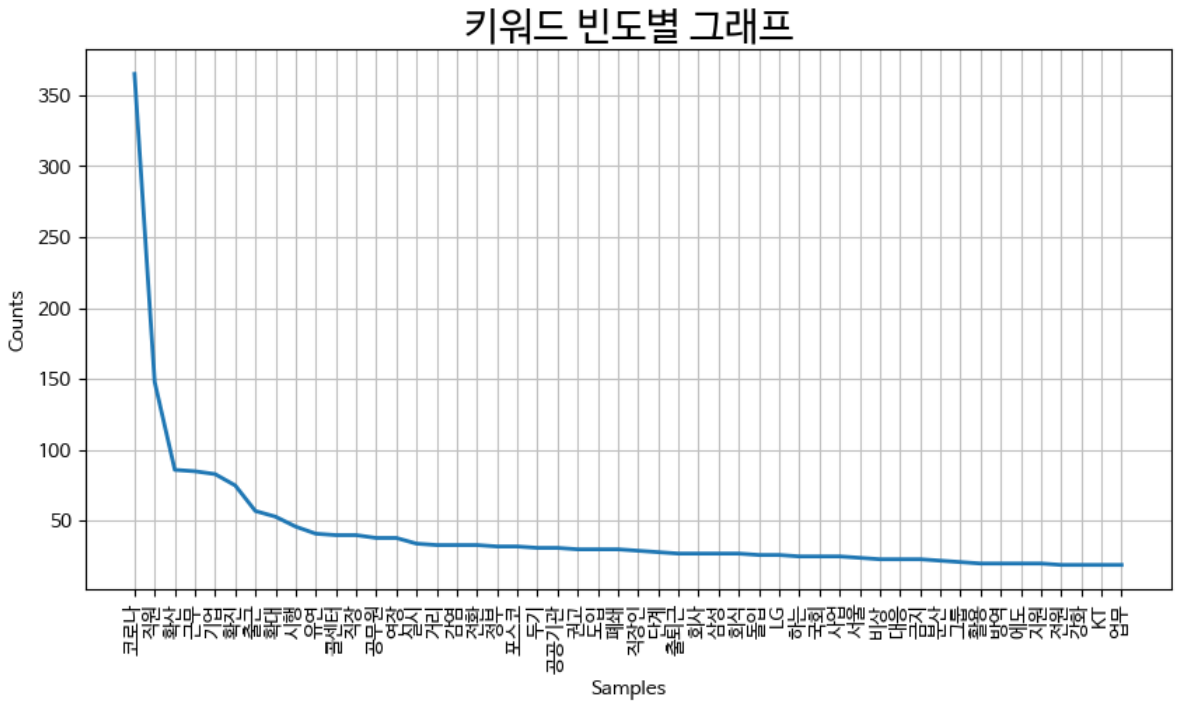
	word	counts
0		968
1		365
2	19	191
3		163
4		148
5		86

	word	counts
6		85
7		83
8		75
9		57
10		53
11		46
12		41
13		41
14		41
15		40
16		40
17		38
18		38
19		34
20		33
21		33
22		33
23		32
24		32
25		31
26		31
27		30
28		30
29		30
30		29
31		28
32		27
33		27
34		27
35		27
36		26
37		26
38	LG	26
39		25
40		25
41		25
42		24
43		23
44		23
45		23
46		22

	word	counts
47		21
48		20
49		20
50		20
51		20
52		19
53		19
54		19
55		19
56		19
57	KT	19
58		18
59		18

```
total_okt_over2 # 9467
total_okt_over3=[]
for element in total_okt_over2:
  if element not in ["  ","   ","19"," "," "," "," "]:
    total_okt_over3.insert(0,element)
```

```
oktplot=nltk.Text(total_okt_over3,name="Test")
fig=plt.figure(figsize = (10,5))
ax = fig.add_subplot(1, 1, 1)
plt.title("       ",fontsize=20)
oktplot.plot(50)
plt.show(ax)
```




```
data=oktplot.vocab().most_common(50)
plt.figure(figsize = (10,5))
path="/usr/share/fonts/truetype/nanum/NanumBarunGothic.ttf"
wc=WordCloud(font_path=path,relative_scaling=0.2,background_color="white",width=1200, heig

plt.imshow(wc, interpolation='bilinear')
plt.axis("off")
plt.tight_layout(pad=0)
plt.show()
```

- . . , , , LG
. , .

- TF-IDF

```
from collections import defaultdict

vectorizer = TfidfVectorizer()
tdm = vectorizer.fit_transform(sentence_arr)
```

```
word_count = pd.DataFrame({
    ' ': vectorizer.get_feature_names(),
    ' ': tdm.sum(axis=0).flat
})
```

```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/deprecation.py:87: FutureWarning: Functi
  warnings.warn(msg, category=FutureWarning)
```

```
word_count.sort_values(" ",ascending=False).reset_index(drop=True).head(30)
```

0	19	248.149943
1		225.877234
2		189.862035
3		166.912066
4		141.297835
5		134.722720
6		116.244045
7		82.505198
8		79.311515
9		77.250924
10		64.115543
11		63.758580
12		60.624583
13		59.834789
14	3	54.067897
15		53.735398
16		51.893703
17		51.615911
18		48.585535
19	1	47.710836
20		45.337426
21		44.223253
22		43.779173
23		43.749493
24		43.617216
25		40.211961
26		38.943360
27		37.046574
28		35.974340
29		35.499295

c. (Co-Occurrence)

: https://bab2min.tistory.com/598

```
# 2          /
cooccur=[]
for elements in okt_morphs_list_stop:
  new_elements=[]
  for i in elements:
    if i not in ["  "," "]:
      text = re.sub(r'[a-zA-Z0-9]',' ',i).strip()
      if len(text)>=2:
        new_elements.append(text)
  cooccur.append(new_elements)
```

```
count = {}   #         dict
for line in cooccur:
    words = line
    for i, a in enumerate(words):
        for b in words[i+1:]:
            if a == b: continue   #
            if a > b:
              a, b = b, a   #A, B  B, A                a < b
            count[a, b] = count.get((a, b), 0) + 1   #
```

```
def dict_to_df(df):
        data_df = pd.DataFrame({'keys': df.keys(), 'features': df.values()})
        data_df['word1'] = data_df['keys'].apply(lambda x: x[0])
        data_df['word2'] = data_df['keys'].apply(lambda x: x[1])
        return data_df[['word1','word2','features']]
```

```
cooccur_df.sort_values("features",ascending=False).head(10)
```

	word1	word2	features
884			89
98			62
24			52
65			46
94			39
770			36
174			34
585			32
1238			30

	word1	word2	features
745			27

```
cooccur_df=dict_to_df(count)
cooccur_df.features.describe()
```

```
count    9559.000000
mean        2.456219
std         2.635716
min         1.000000
25%         1.000000
50%         2.000000
75%         3.000000
max        89.000000
Name: features, dtype: float64
```

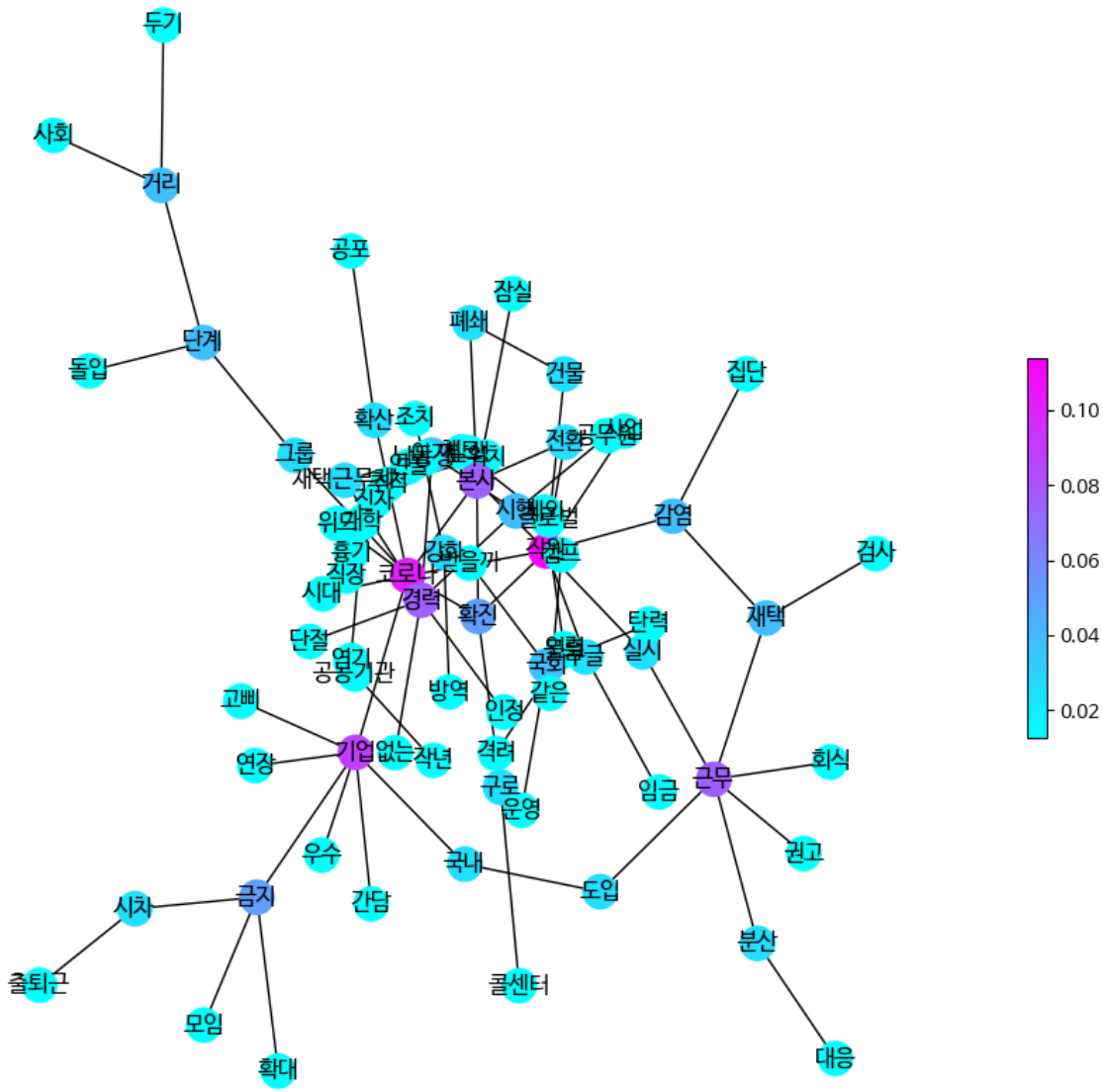
d.

```
import networkx as nx
import operator
import matplotlib.colors as mcolors
import matplotlib.cm as cm
```

```
cooccur_df_major=cooccur_df >> mask(X.features>=13)
```

```
# generate sample graph
plt.figure(figsize = (9,9),facecolor='k')
plt.rcParams['font.sans-serif'] = ['NanumBarunGothic']
g = nx.from_pandas_edgelist(cooccur_df_major, 'word1', 'word2')


# centrality
deg_centrality = nx.degree_centrality(g)
centrality = np.fromiter(deg_centrality.values(), float)
# plot
pos = nx.kamada_kawai_layout(g,scale=3)
nx.draw(g, pos, node_color=centrality,with_labels=True)
plt.title("        ",size=15)
```

```
plt.cool()


sizes = centrality / np.max(centrality) * 200
normalize = mcolors.Normalize(vmin=centrality.min(), vmax=centrality.max())
colormap = cm.cool

scalarmappaple = cm.ScalarMappable(norm=normalize, cmap=colormap)
scalarmappaple.set_array(centrality)

plt.colorbar(scalarmappaple,shrink=0.3)

plt.show()
```




```
def dict_to_df_1(df):
        data_df = pd.DataFrame({'keys': df.keys(), 'features': df.values()})
        data_df['word1'] = data_df['keys']
```

```
        return data_df[['word1','features']]

dict_to_df_1(deg_centrality).sort_values("features",ascending=False)[1:20]
```

	word1	features
5		0.101266
10		0.088608
34		0.075949
44		0.075949
6		0.075949
1		0.050633
48		0.050633
57		0.037975
7		0.037975
16		0.037975
36		0.037975
18		0.037975
73		0.037975
33		0.037975
55		0.025316
26		0.025316
66		0.025316
42		0.025316
41		0.025316

- ( )
- .
- .
- , .

03/ Analysis 2 – Descriptive Statistics ?

?

1. (2015~2021)

- : / - : - : (, ,) -
  - : - : ( )→ ( )→ - : - : - : 2017 9

```
from mizani.breaks import date_breaks, minor_breaks
from mizani.formatters import date_format
import matplotlib.font_manager as fm
```

```
path = '/usr/share/fonts/truetype/nanum/NanumBarunGothic.ttf'
font = fm.FontProperties(fname=path, size=13)
```

a.

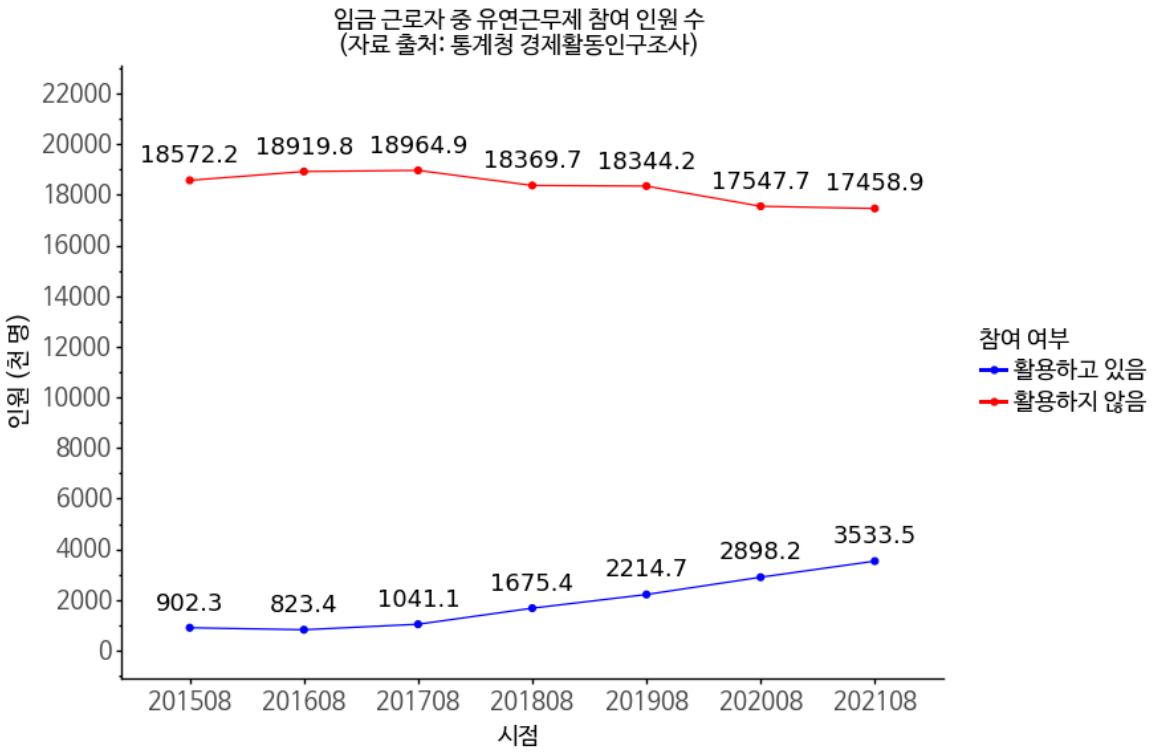
```
yes=pd.read_json("https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJ
no=pd.read_json("https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJm
```

```
pj_usege=yes >> bind_rows(no, join='outer') >> select(X.PRD_DE,X.C1_NM,X.DT)
pj_usege=pj_usege.reset_index(drop=True)
```

```
plt.figure()
pn.options.figure_size = (8,6)
(ggplot(pj_usege, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(0,22001),breaks= np.arange(0,22001,2000))
    + geom_line()
    + geom_point(pj_usege,aes(x='factor(PRD_DE)',y="DT",group="C1_NM",color="C1_NM"))
    + theme_classic()
    + ylab("  (  )")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="   ")
    + scale_color_manual(values=("blue","red"))
    + geom_text(aes(label="DT"),nudge_x=0, nudge_y=1000,size=13,color="black")
    + ggtitle('              \n(   :       )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762697391465)>
```

b.

```
link_list=[]
link="https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJmNDI2NWE1N2U
for i in range(201508,202109,100):
  link_list.append(link+str(i))
```

```
pj_gender=pd.read_json(link_list[0])
for i in link_list[1:]:
  pj_gender = pj_gender >> bind_rows(pd.read_json(i), join='outer')
```

```
pj_gender=pj_gender.reset_index(drop=True)
```

```
pj_gender2=pj_gender >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(~X.C1_NM==" ") >>  ma
pj_gender2=pj_gender2.reset_index(drop=True)
```

```
pj_total=pj_gender >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT)  >> mask(~X.C1_NM==" ") >> mask
```

```
pj_total
```

	PRD_DE	C1_NM	C2_NM	DT
3	201508			11006.6
6	201508			8467.9
12	201608			11085.6
15	201608			8657.6
21	201708			11188.2
24	201708			8817.9
30	201808			11171.3
33	201808			8873.7
39	201908			11395.7
42	201908			9163.3
48	202008			11361.3
51	202008			9084.6
57	202108			11516.6
60	202108			9475.9

```
pj_total2=pj_total >> select(~X.C2_NM) >> spread(X.PRD_DE, X.DT) >> select(~X.C1_NM)
pj_total2.index=[" "," "]
pj_total2.transpose() >> mutate(ratio=np.round(X. /(X. +X. ),2),  =np.round(X. /(X. ),2) )
#               /
```

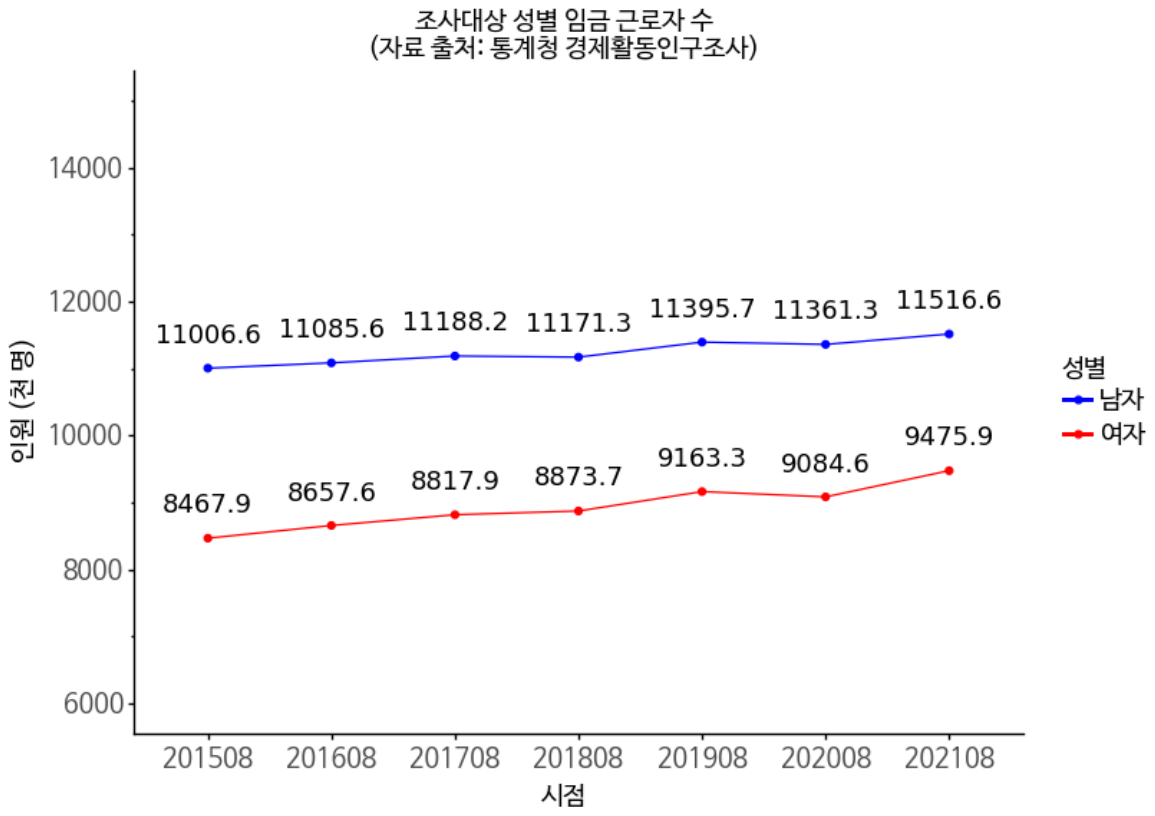
			ratio	
201508	11006.6	8467.9	0.57	1.30
201608	11085.6	8657.6	0.56	1.28
201708	11188.2	8817.9	0.56	1.27
201808	11171.3	8873.7	0.56	1.26
201908	11395.7	9163.3	0.55	1.24
202008	11361.3	9084.6	0.56	1.25

			ratio	
202108	11516.6	9475.9	0.55	1.22

```
plt.figure()
pn.options.figure_size = (8,6)
(ggplot(pj_total, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(6000,15001),breaks= np.arange(0,22001,2000))
    + geom_line()
    + geom_point(pj_total,aes(x='factor(PRD_DE)',y="DT",group="C1_NM",color="C1_NM"))
    + theme_classic()
    + ylab("  (  )")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour=" ")
    + scale_color_manual(values=("blue","red"))
    + geom_text(aes(label="DT"),nudge_x=0, nudge_y=500,size=13,color="black")
    + ggtitle('          \n(   :       )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```

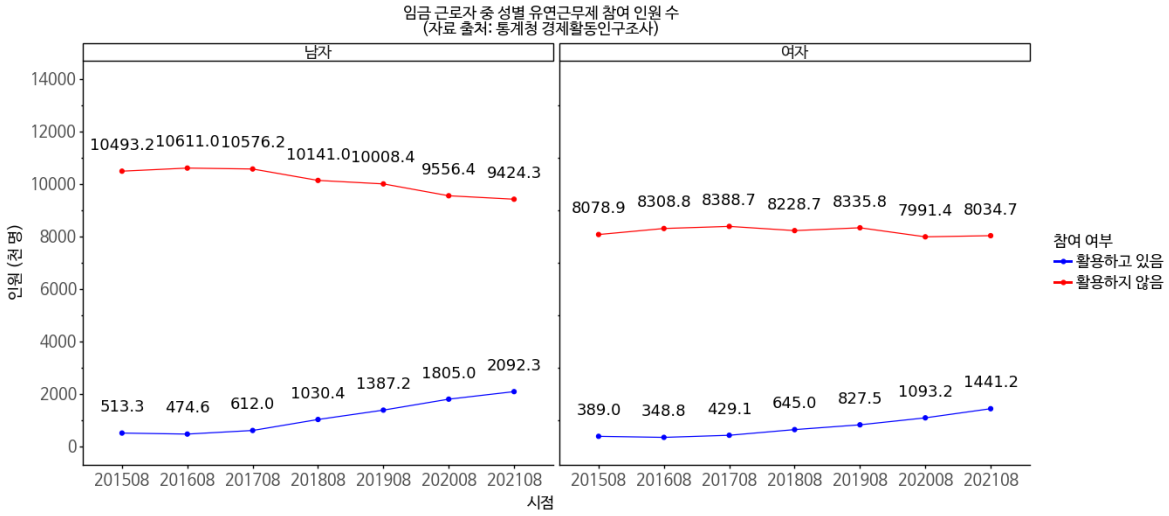


<ggplot: (8762697419985)>

```
plt.figure()
pn.options.figure_size = (14,6)
(ggplot(pj_gender2, aes(x='factor(PRD_DE)', y="DT",group="C2_NM",color="C2_NM"))
    + facet_wrap('C1_NM')
    + scale_y_continuous(limits=(0,14001),breaks= np.arange(0,22001,2000))
    + geom_line()
    + geom_point(pj_gender2, aes(x='factor(PRD_DE)', y="DT",group="C2_NM",color="C2_NM"))
    + theme_classic()
    + ylab("  (  )")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="   ")
    + scale_color_manual(values=("blue","red"))
    + geom_text(aes(label="DT"),nudge_x=0, nudge_y=1000,size=13,color="black")
    + ggtitle('                \n(   :       )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762697654101)>
```

```
#
ratio_total=(pj_gender >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM==" ") >> s
  >> mutate(ratio_use=np.round(X.used/X. ,3)*100,ratio_notuse=np.round(X.notused/X. ,3)*100
```

```
ratio_male=(pj_gender >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM==" ") >> se
  >> mutate(ratio_use=np.round(X.used/X. ,3)*100,ratio_notuse=np.round(X.notused/X. ,3)*100
```

```
ratio_female=(pj_gender >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM==" ") >>
  >> mutate(ratio_use=np.round(X.used/X. ,3)*100,ratio_notuse=np.round(X.notused/X. ,3)*100
```

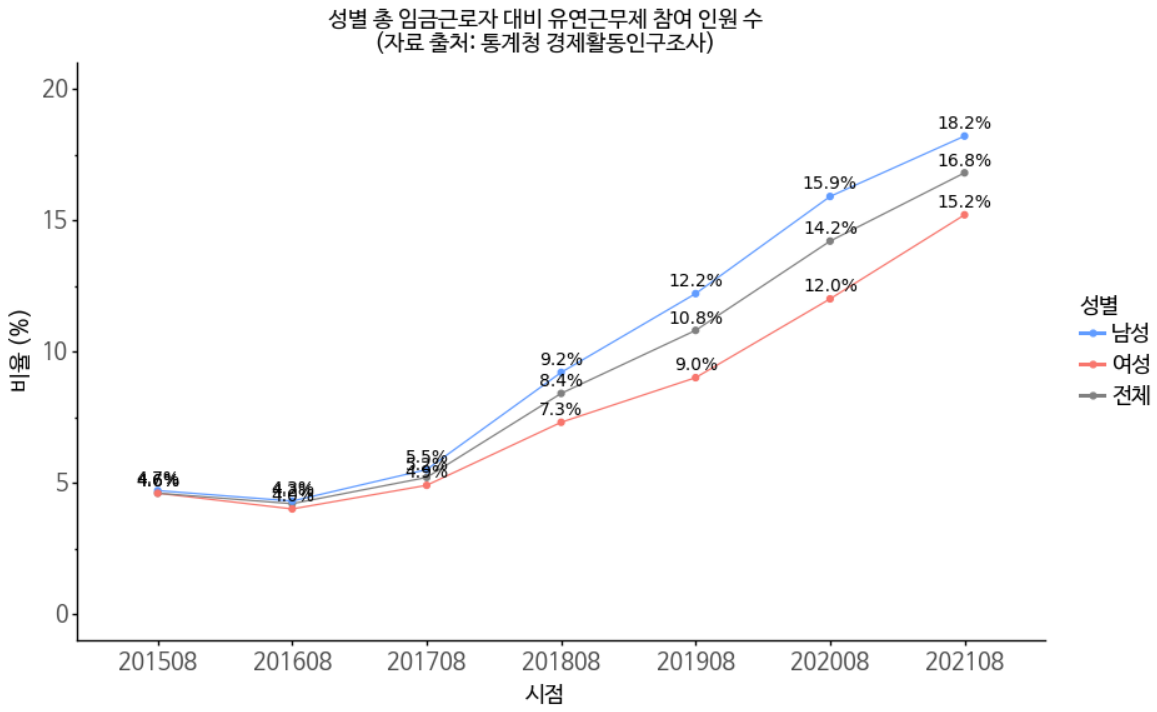
```
ratio=pd.concat([ratio_total,ratio_male,ratio_female], keys=[" "," "," "]).reset_index() >
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(ratio, aes(x='factor(PRD_DE)', y="ratio_use",group=" ",color=" "))
```

```
    + scale_y_continuous(limits=(0,20),breaks= np.arange(0,101,5))
    + geom_line()
    + geom_point(ratio, aes(x='factor(PRD_DE)', y="ratio_use",group=" ",color=" "))
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour=" ")
    + scale_color_manual(values=("#619CFF","#F8766D","gray"))
    + geom_text(aes(label="ratio_use"),nudge_x=0, nudge_y=+0.5,size=10,color="black",forma
    + ggtitle('                    \n(    :         )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762696383501)>
```

```
pj_age_2021=pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.PRD_DE==202108) >> m
```

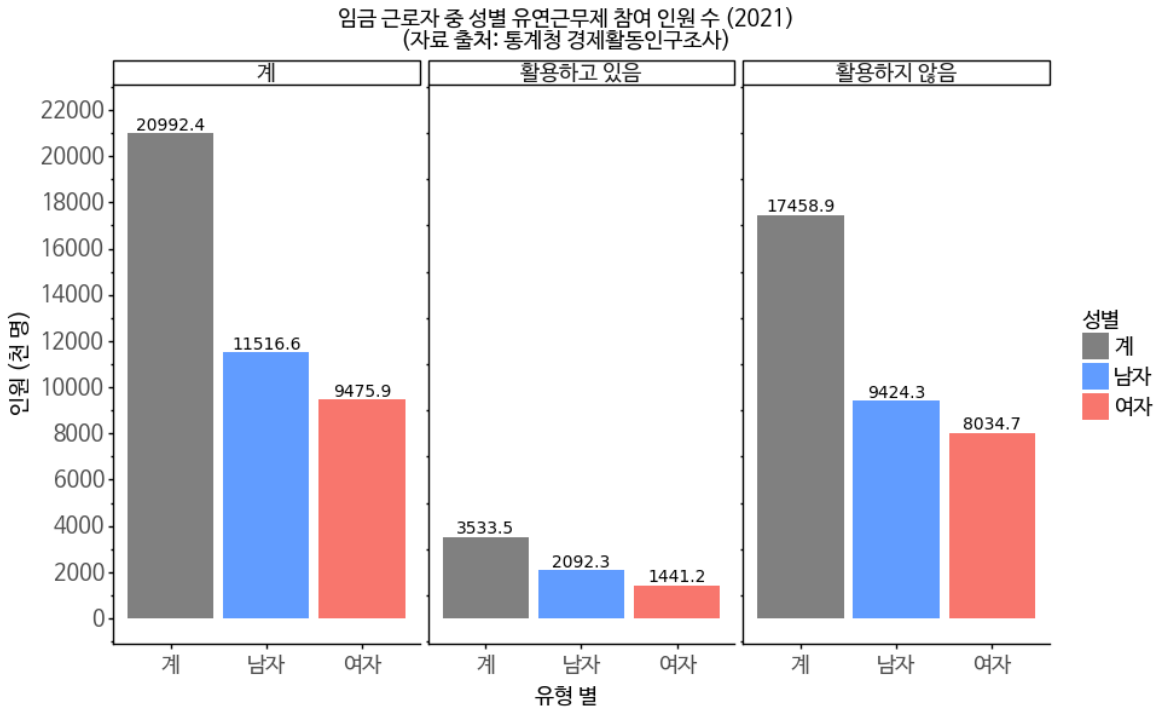
```
# 2021
```

```
pj_2021=pj_gender >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.PRD_DE==202108)
```

```
plt.figure()
pn.options.figure_size = (10,6)
dodge_text = position_dodge(width=0.9)
(ggplot(pj_2021,aes(x='C1_NM',y="DT",fill="C1_NM",group="C1_NM"))
    + scale_y_continuous(limits=(0,22000),breaks= np.arange(0,22001,2000))
    + geom_bar(stat='identity',position = position_dodge(width = 0.9))
    + theme_classic()
    + ylab("  (  )")
    + xlab("   ")
    + theme(text=element_text(fontproperties=font))
    + labs(fill=" ")
    + facet_wrap('C2_NM')
    + scale_fill_manual(values=("gray","#619CFF","#F8766D"))
    + geom_text(aes(label='DT'), position=dodge_text,size=10, va='bottom', format_string='
    + ggtitle('                  (2021)\n(    :        )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```



<ggplot: (8762697263013)>

c.

```
link_list=[]
link="https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJmNDI2NWE1N2U
for i in range(201508,202109,100):
  link_list.append(link+str(i))
```

```
pj_age=pd.read_json(link_list[0])
for i in link_list[1:]:
  pj_age = pj_age >> bind_rows(pd.read_json(i), join='outer')
```

```
ratio_age_total=(pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM==" ") >> s
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_age_15=(pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM=="15 - 29 ")
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_age_30=(pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM=="30 - 39 ")
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_age_40=(pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM=="40 - 49 ")
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_age_50=(pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM=="50 - 59 ")
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_age_60=(pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM=="60  ") >> s
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

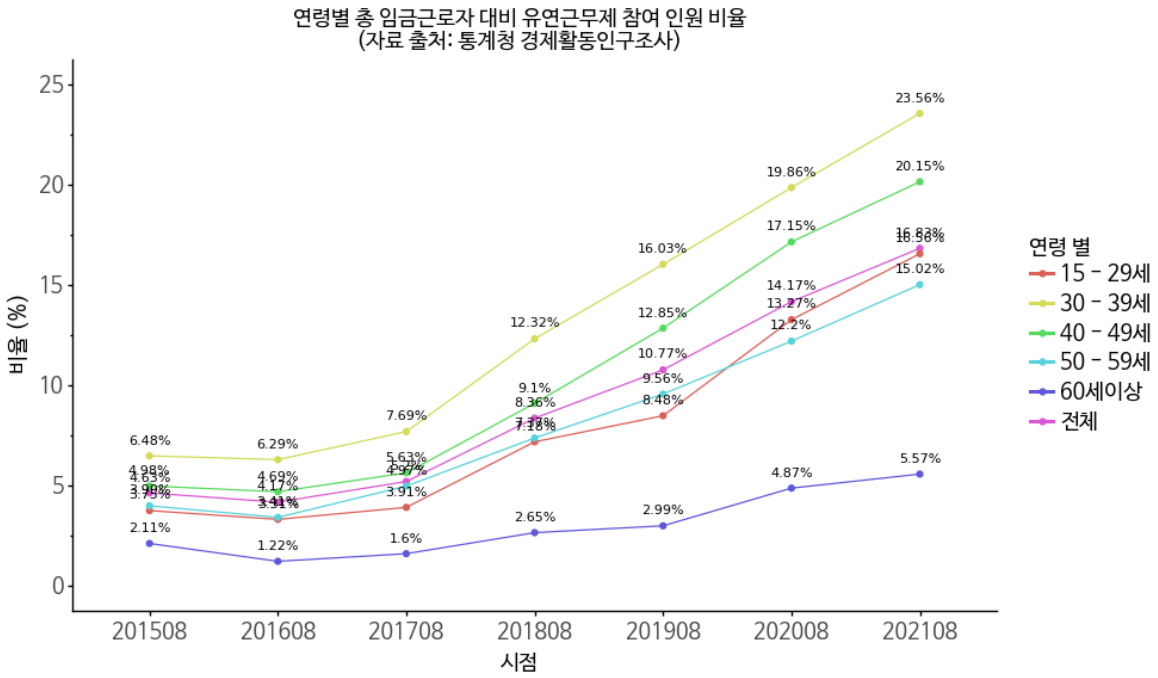
```
ratio_age=pd.concat([ratio_age_total,ratio_age_15,ratio_age_30,ratio_age_40,ratio_age_50,r
```

```
ratio_age["ratio_use"]=np.round(ratio_age["ratio_use"].astype(np.float64),2)
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(ratio_age, aes(x='factor(PRD_DE)', y="ratio_use",group="  ",color="  "))
    + scale_y_continuous(limits=(0,25),breaks= np.arange(0,101,5))
    + geom_line()
    + geom_point(ratio_age, aes(x='factor(PRD_DE)', y="ratio_use",group="  ",color="  "))
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="   ")
    + geom_text(aes(label="ratio_use"),nudge_x=0, nudge_y=+0.8,size=8,color="black",format
    + ggtitle('                  \n(    :        )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```



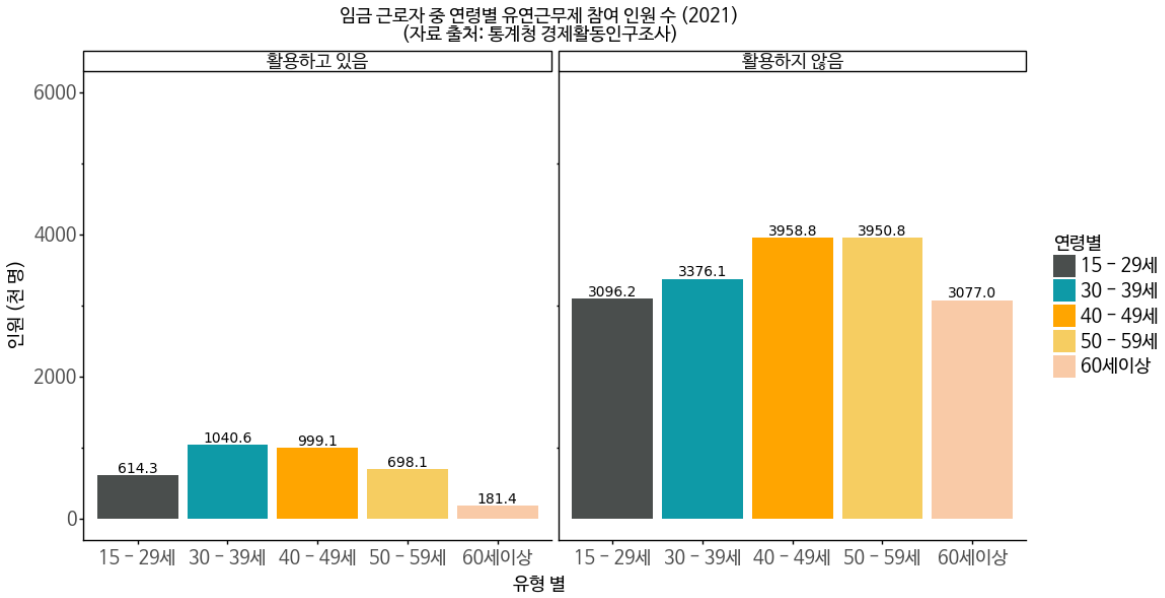
<ggplot: (8731672718093)>

```
pj_age_2021=pj_age >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.PRD_DE==202108) >> m
```

```
plt.figure()
pn.options.figure_size = (12,6)
dodge_text = position_dodge(width=0.9)
(ggplot(pj_age_2021,aes(x='C1_NM',y="DT",fill="C1_NM",group="C1_NM"))
    + scale_y_continuous(limits=(0,6000),breaks= np.arange(0,22001,2000))
    + geom_bar(stat='identity',position = position_dodge(width = 0.9))
    + theme_classic()
    + ylab("  (  )")
    + xlab("  ")
    + theme(text=element_text(fontproperties=font))
    + labs(fill="  ")
    + facet_wrap('C2_NM')
    + geom_text(aes(label='DT'), position=dodge_text,size=10, va='bottom', format_string='
    + scale_fill_manual(values=("#4a4e4d","#0e9aa7" ,"orange","#f6cd61","#f9caa7"))
    + ggtitle('                  (2021)\n(   :       )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762696667145)>
```

d.

```
link_marriage_list=[]
link="https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJmNDI2NWE1N2U
for i in range(201508,202109,100):
  link_marriage_list.append(link+str(i))
```

```
pj_marriage=pd.read_json(link_marriage_list[0])
for i in link_marriage_list[1:]:
  pj_marriage = pj_marriage >> bind_rows(pd.read_json(i), join='outer')
```

```
ratio_marriage_total=(pj_marriage >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_marriage_no=(pj_marriage >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM=="
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

```
ratio_marriage_yes=(pj_marriage >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.C1_NM==
  >> mutate(ratio_use=np.round(X.used/X. ,4)*100,ratio_notuse=np.round(X.notused/X. ,4)*100
```

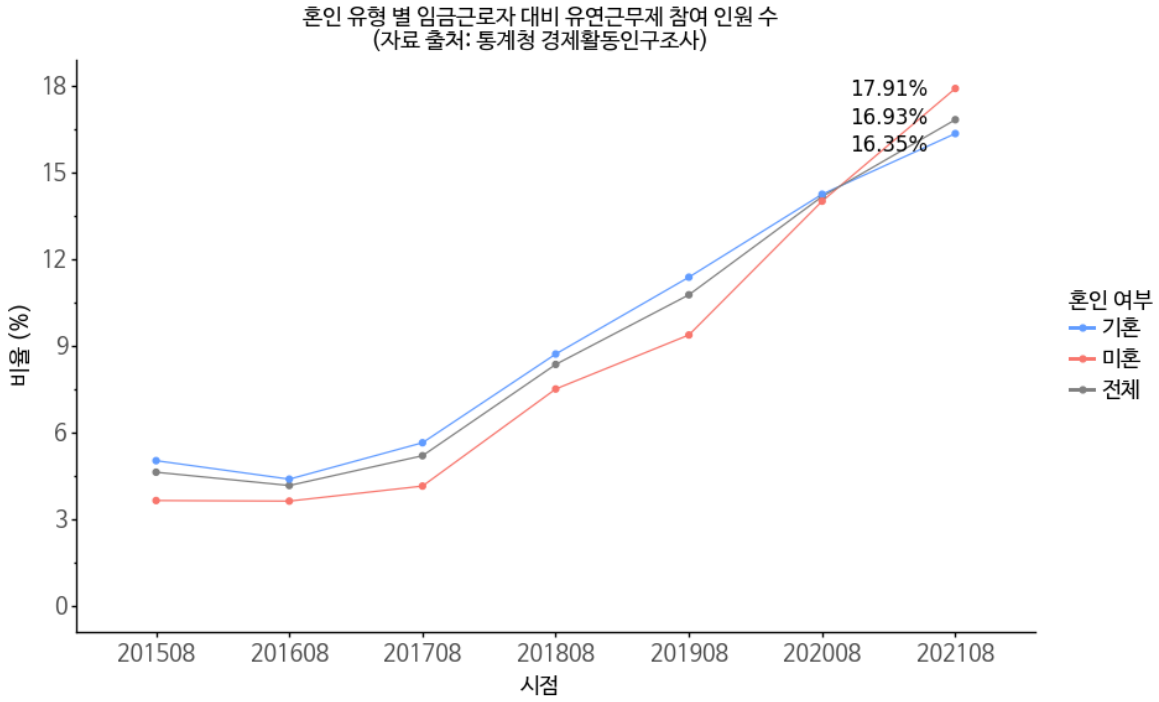
```
ratio_marriage=pd.concat([ratio_marriage_total,ratio_marriage_yes,ratio_marriage_no], keys
```

```
ratio_marriage["ratio_use"]=np.round(ratio_marriage["ratio_use"].astype(np.float64),2)
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(ratio_marriage, aes(x='factor(PRD_DE)', y="ratio_use",group=" ",color=" "))
    + scale_y_continuous(limits=(0,18),breaks= np.arange(0,101,3))
    + geom_line()
    + geom_point(ratio_marriage, aes(x='factor(PRD_DE)', y="ratio_use",group=" ",color=" "
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="   ")
    + scale_color_manual(values=("#619CFF","#F8766D","gray"))
    + annotate("text", x=6.5, y=17.91, label="17.91%", size=12,color="black")
    + annotate("text", x=6.5, y=16.93, label="16.93%", size=12,color="black")
    + annotate("text", x=6.5, y=16, label="16.35%", size=12,color="black")
    + ggtitle('                    \n(   :       )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```



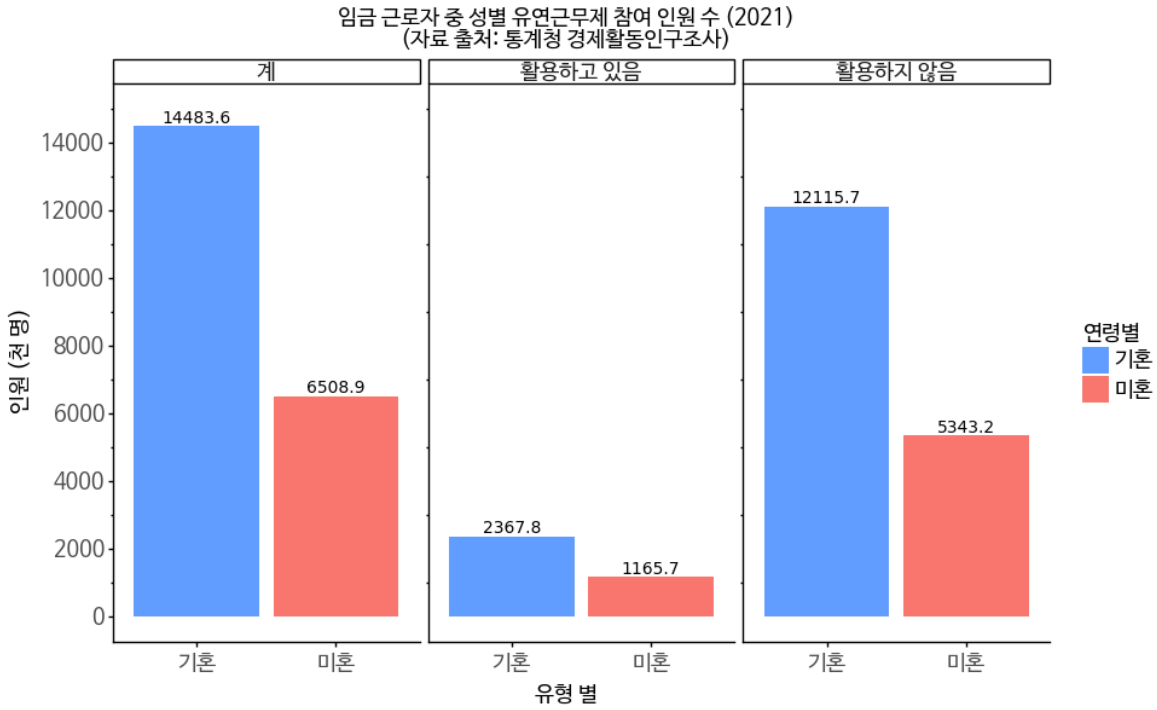
<ggplot: (8762696204773)>

```
pj_marriage_2021=pj_marriage >> select(X.PRD_DE,X.C1_NM,X.C2_NM,X.DT) >> mask(X.PRD_DE==20
```

```
plt.figure()
pn.options.figure_size = (10,6)
dodge_text = position_dodge(width=0.9)
(ggplot(pj_marriage_2021,aes(x='C1_NM',y="DT",fill="C1_NM",group="C1_NM"))
    + scale_y_continuous(limits=(0,15000),breaks= np.arange(0,22001,2000))
    + geom_bar(stat='identity',position = position_dodge(width = 0.9))
    + theme_classic()
    + ylab("  (  )")
    + xlab("  ")
    + theme(text=element_text(fontproperties=font))
    + labs(fill="  ")
    + facet_wrap('C2_NM')
    + geom_text(aes(label='DT'), position=dodge_text,size=10, va='bottom', format_string='
    + scale_fill_manual(values=("#619CFF","#F8766D"))
    + ggtitle('                (2021)\n(   :       )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762696952981)>
```

e. **( )**

```
link_type_list=[]
link="https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJmNDI2NWE1N2U
for i in range(201508,202109,100):
  link_type_list.append(link+str(i))
```

```
pj_type=pd.read_json(link_type_list[0])
for i in link_type_list[1:]:
  pj_type = pj_type >> bind_rows(pd.read_json(i), join='outer')
```

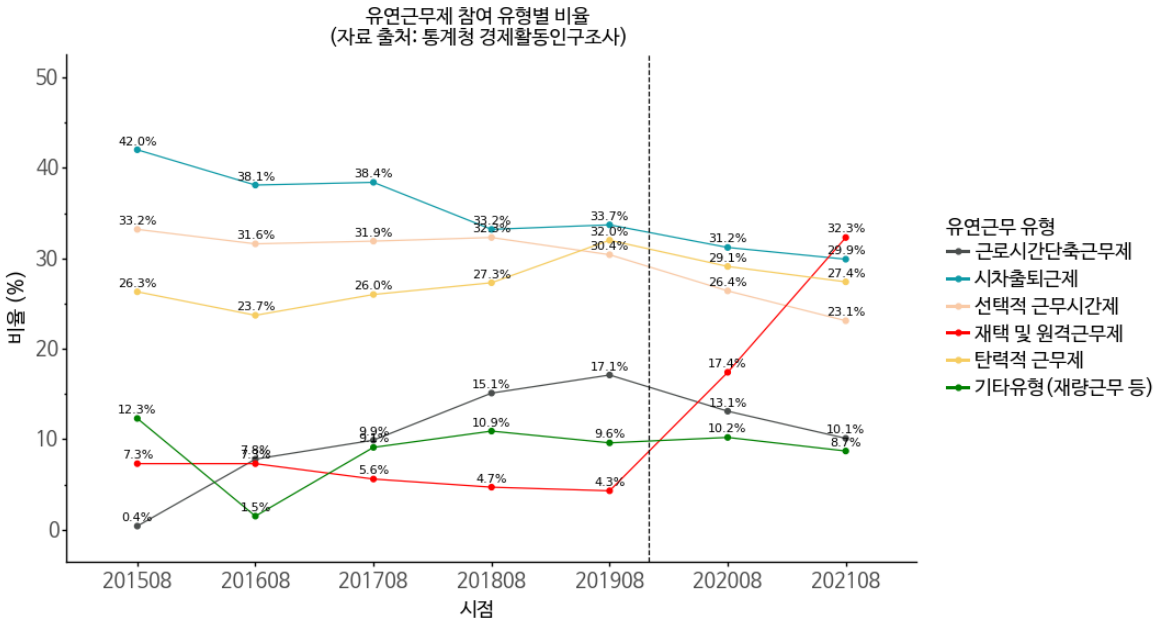
```
ratio_type=pj_type >> mask(X.ITM_NM_ENG=="ratio") >> select(X.PRD_DE,X.C1_NM,X.DT)
```

```
ratio_type['C1_NM']=pd.Categorical(ratio_type['C1_NM'], ordered=True,categories=['     ','
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(ratio_type,aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + geom_line()
    + scale_y_continuous(limits=(-1,50),breaks= np.arange(0,101,10))
    + geom_point(ratio_type, aes(x='factor(PRD_DE)', y="DT",color="C1_NM"))
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="    ")
    + scale_color_manual(values=("#4a4e4d","#0e9aa7" ,"#f9caa7","red","#f6cd61","green"))
    + geom_text(aes(label="DT"),nudge_x=0, nudge_y=+1,size=8,color="black",format_string='
    + geom_vline(xintercept=5+(4/12),linetype="dashed")
    + ggtitle('         \n(   :        )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```



<ggplot: (8762696647101)>

( )

- COVID-19 .
- . (2021 8 18.2%, 15.2%)
- 30 40 20 50 . 60 50
.
- COVID-19 , .(2020 8 17.91%, 16.35%)
- COVID-19 ( ) , 2021 .
- , .

2. (2017~2020)

2017~2020 ( ) 5 ‘ ’ . -
46 17 / 10 -
(95%) 2% 3% - E-mail .

```
#                    .
link="https://kosis.kr/openapi/statisticsData.do?method=getList&apiKey=YjNjZjJmNDI2NWE1N2U

def kosis(link,a,b):
```

```
  link_list=[]
  for i in range(int(a),int(b)+1,1):
    new_link=link+str(i)
    link_list.append(new_link)

  df= pd.read_json(link_list[0])
  if len(link_list)>=2:
    for i in link_list[1:]:
      df = df >> bind_rows(pd.read_json(i), join='outer')
  result=df.reset_index(drop=True)
  return result
```

```
pj_gov=kosis(link,2017,2020)
```

a.

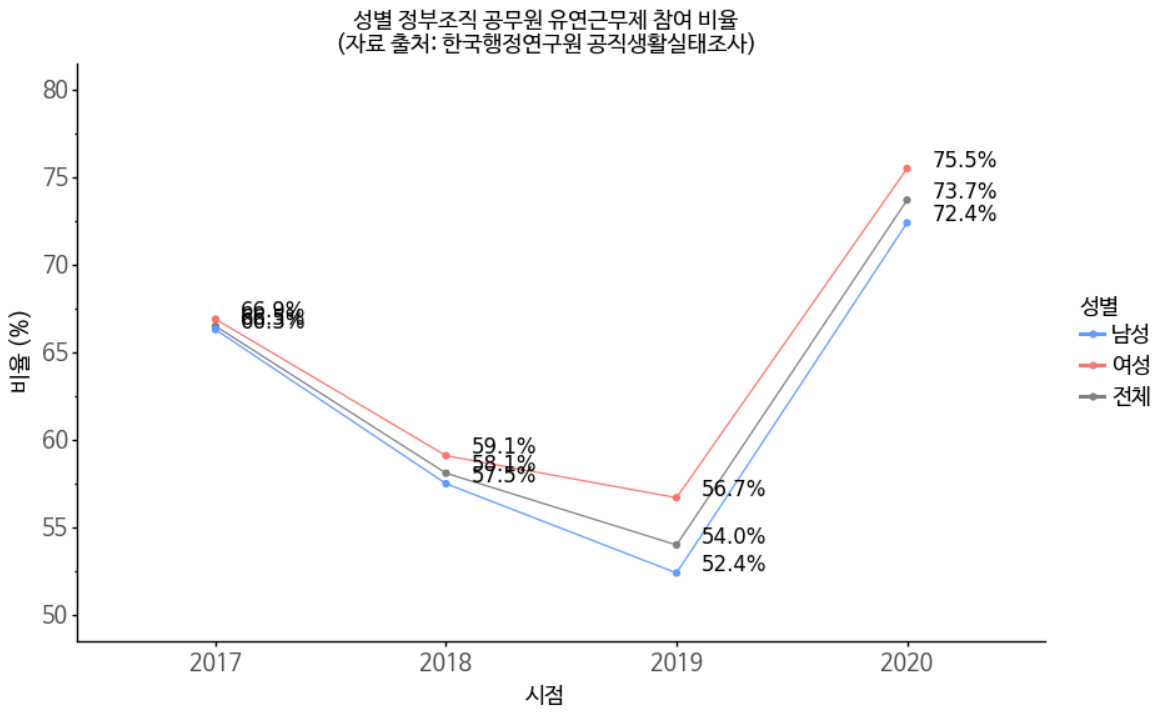
```
pj_gov=pj_gov>>mask(X.C2_NM==" ")
```

```
pj_gov_gender=pj_gov >> mask((X.C1_NM==" ")|(X.C1_NM==" ")|(X.C1_NM==" "))
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(pj_gov_gender, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(50,80),breaks= np.arange(0,101,5))
    + geom_line()
    + geom_point(pj_gov_gender, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour=" ")
    + scale_color_manual(values=("#619CFF","#F8766D","gray"))
    + geom_text(aes(label="DT"),nudge_x=.25, nudge_y=+.5,size=12,color="black",format_stri
    + ggtitle('              \n(   :          )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762697292525)>
```

b.

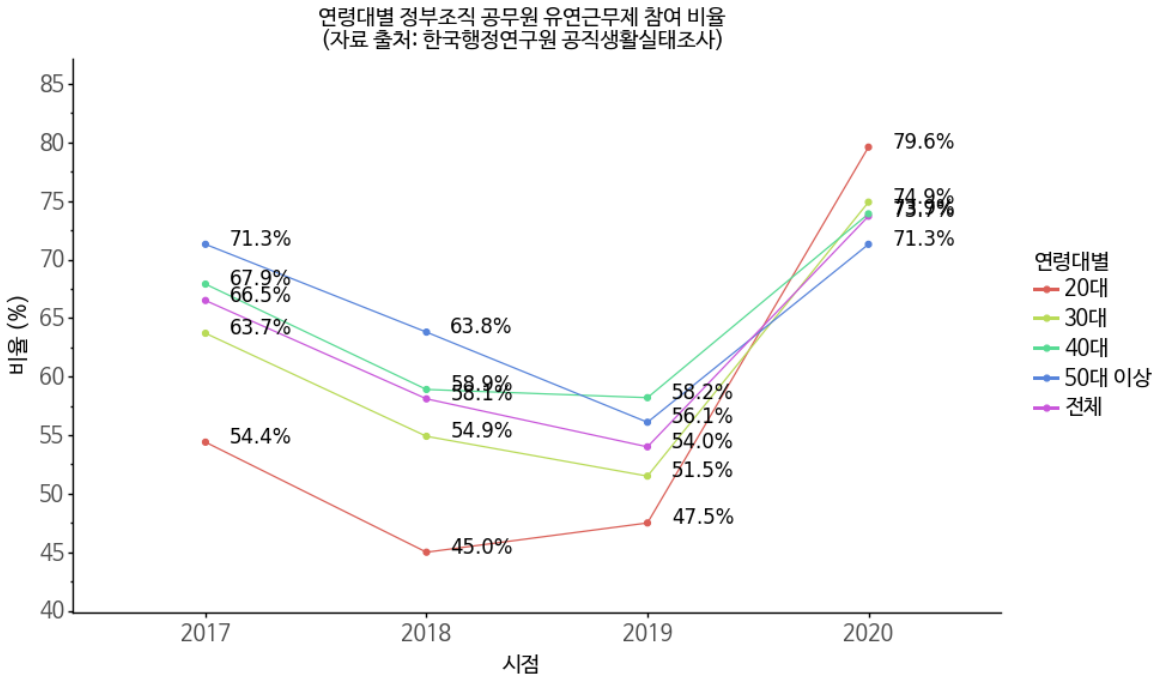
```
pj_gov_age=pj_gov >> mask((X.C1_NM=="20 ")|(X.C1_NM=="30 ")|(X.C1_NM=="40 ")|(X.C1_NM=="50
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(pj_gov_age, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(42,85),breaks= np.arange(0,101,5))
    + geom_line()
    + geom_point(pj_gov_age, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="  ")
```

```
    + geom_text(aes(label="DT"),nudge_x=.25, nudge_y=+.5,size=12,color="black",format_stri
    + ggtitle('                \n(    :          )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762698177993)>
```

c.

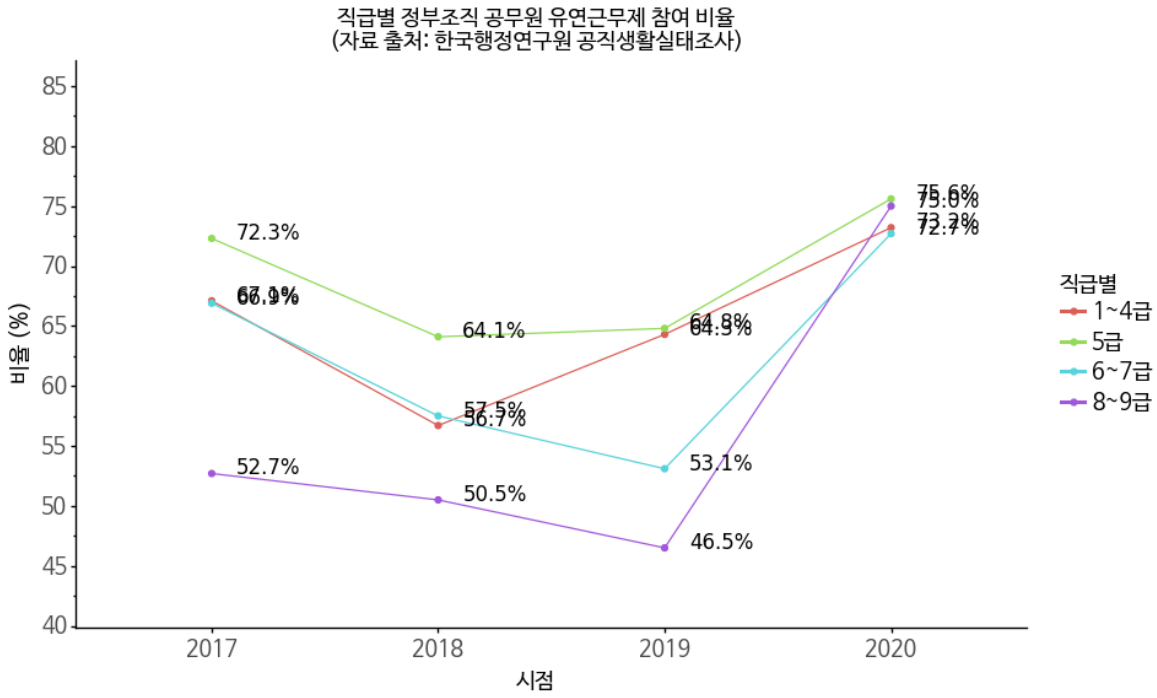
```
pj_gov_rank=pj_gov >> mask((X.C1_NM=="1~4 ")|(X.C1_NM=="5 ")|(X.C1_NM=="6~7 ")|(X.C1_NM=="8
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(pj_gov_rank, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(42,85),breaks= np.arange(0,101,5))
```

```
    + geom_line()
    + geom_point(pj_gov_rank, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="  ")
    + geom_text(aes(label="DT"),nudge_x=.25, nudge_y=+.5,size=12,color="black",format_stri
    + ggtitle('                \n(    :          )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762697124449)>
```

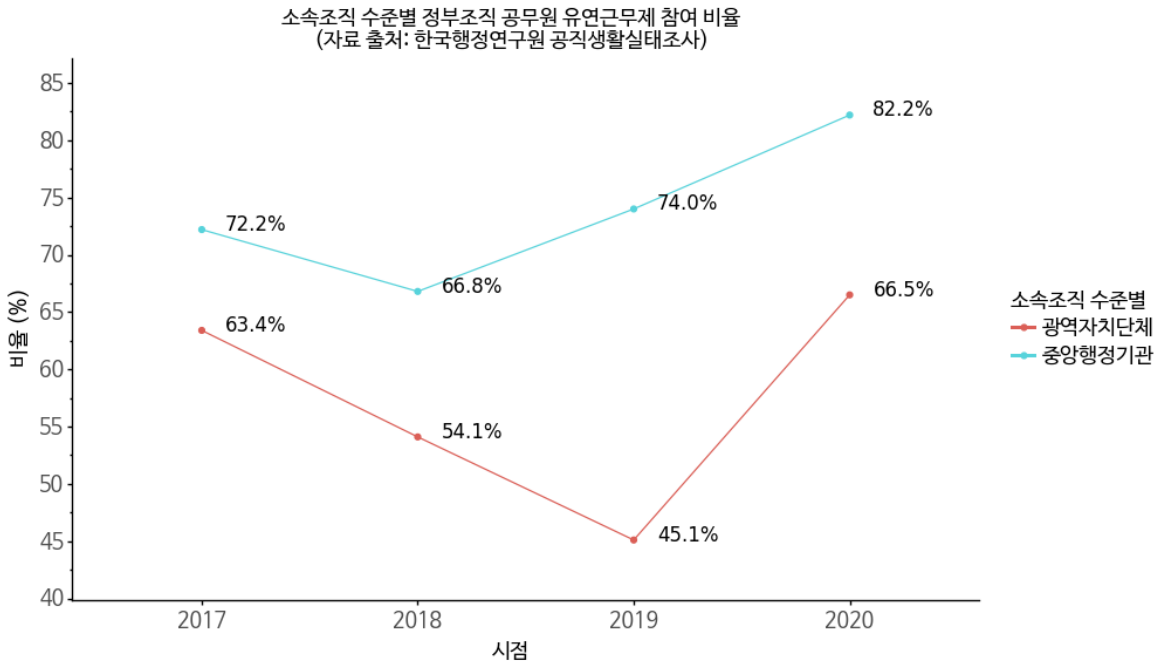
d.

```
pj_gov_level=pj_gov >> mask((X.C1_NM=="    ")|(X.C1_NM=="    "))
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(pj_gov_level, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(42,85),breaks= np.arange(0,101,5))
    + geom_line()
    + geom_point(pj_gov_level, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM")
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="     ")
    + geom_text(aes(label="DT"),nudge_x=.25, nudge_y=+.5,size=12,color="black",format_stri
    + ggtitle('                    \n(    :          )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```



<ggplot: (8762696037269)>

e.

```
pj_gov_length=pj_gov >> mask((X.C1_NM=="5   ")|(X.C1_NM=="6~10 ")|(X.C1_NM=="11~15 ")|(X.C1
```

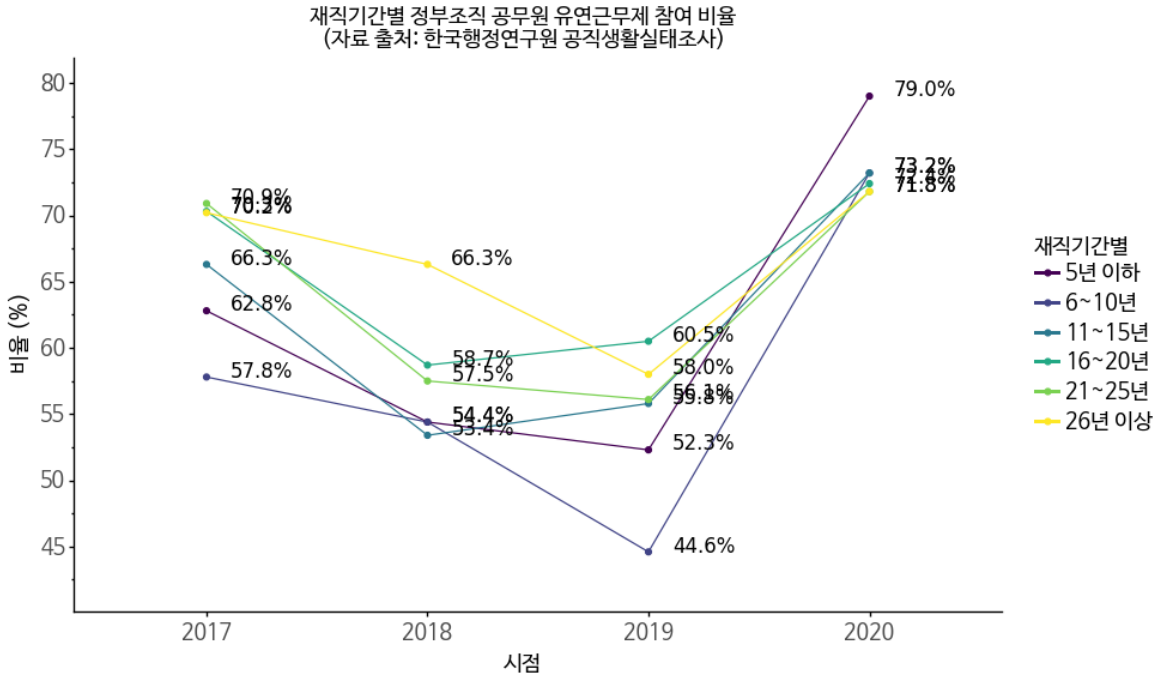
```
pj_gov_length["C1_NM"]=pd.Categorical(pj_gov_length["C1_NM"], ordered=True,categories=["5
```

```
plt.figure()
pn.options.figure_size = (10,6)
(ggplot(pj_gov_length, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"))
    + scale_y_continuous(limits=(42,80),breaks= np.arange(0,101,5))
    + geom_line()
    + geom_point(pj_gov_length, aes(x='factor(PRD_DE)', y="DT",group="C1_NM",color="C1_NM"
    + theme_classic()
    + ylab("  (%)")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(colour="   ")
    + geom_text(aes(label="DT"),nudge_x=.25, nudge_y=+.5,size=12,color="black",format_stri
```

```
    + ggtitle('                \n(    :          )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8762696138673)>
```

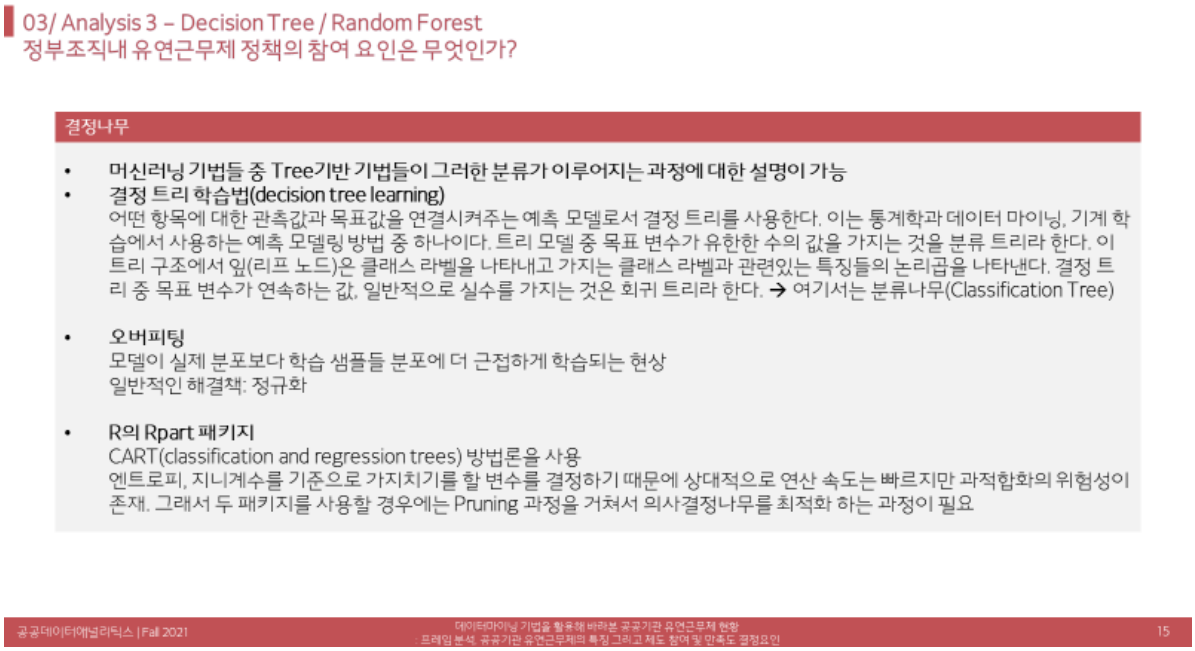
( )

- COVID-19 , .
- 2 .
- . (2020 72.4%, 75.5%) 20 COVID-19 , COVID-19 . (2019 47.5% 2020 79.6%) .(, ... or 2015)
- , , COVID-19 , 2020 , 5 79% .

- , (Choi, 2017)
.

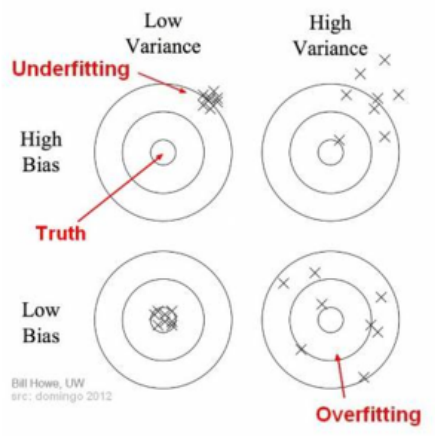
03/ Analysis 3 – Decision Tree / Random Forest ?
?

```
Image(filename='/21.png')
```




```
Image(filename='/22.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

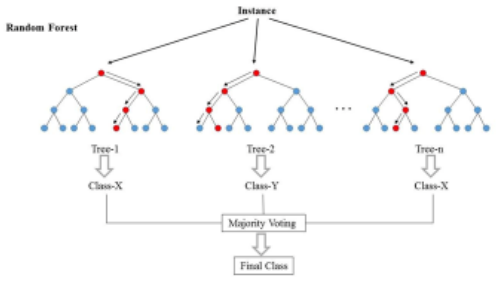



```
Image(filename='/23.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

랜덤포레스트

- **결정나무 분석 방법의** 앙상블(Ensemble method)
- **배깅**:
  배깅(bagging)은 bootstrap aggregating의 약자로, 부트스트랩(bootstrap)을 통해 조금씩 다른 훈련 데이터에 대해 훈련된 기초 분류기(base learner)들을 결합(aggregating)시키는 방법이다.
- 트리는 작은 Bias와 큰 variance을 갖기 때문에, 매우 깊이 성장한 트리는 훈련 데이터에 대해 과적합(overfitting)하게 된다.
  부트스트랩과정은 트리들의 편향은 그대로 유지하면서, 분산은 감소
  한 개의 결정 트리의 경우 훈련 데이터에 있는 노이즈에 대해서 매우 민감하지만, 트리들이 서로 상관화(correlated)되어 있지 않다면 여러 트리들의 평균은 노이즈에 대해 강건



- (2017 ~2020)

```
gc = gspread.authorize(GoogleCredentials.get_application_default())

worksheet3 = gc.open('kipa_pre').sheet1
rows1 = worksheet3.get_all_values()
```

```
kipa_pre=pd.DataFrame.from_records(rows1)
```

```
kipa_pre.columns=kipa_pre.iloc[0,:]
kipa_pre=kipa_pre.iloc[1:,1:]
```

```
kipa_pre
```

	year	fwa	smartwork	gender	age	work_year	rank	marriage	children	workamount	extra
1	2017	0	1	1	4	6	2	1	0	4	30
2	2017	1	0	1	2	3	3	1	1	4	25
3	2017	1	0	2	3	6	3	1	1	4	8
4	2017	1	1	1	2	2	2	1	1	5	20
5	2017	0	1	1	4	6	1	1	0	4	18
...	...	...	...	...	...	...	...	...	...	...	...
15563	2020	1	1	1	3	2	4	0	0	4	3
15564	2020	1	1	1	3	4	1	0	1	3	3
15565	2020	1	1	1	2	1	4	0	1	5	2
15566	2020	1	1	1	3	3	1	0	1	4	4
15567	2020	0	1	1	4	5	2	0	1	4	4

-

```
kipa_pre.isnull().sum() #
```

```
0
year            0
fwa             0
smartwork       0
gender          0
age             0
work_year       0
```

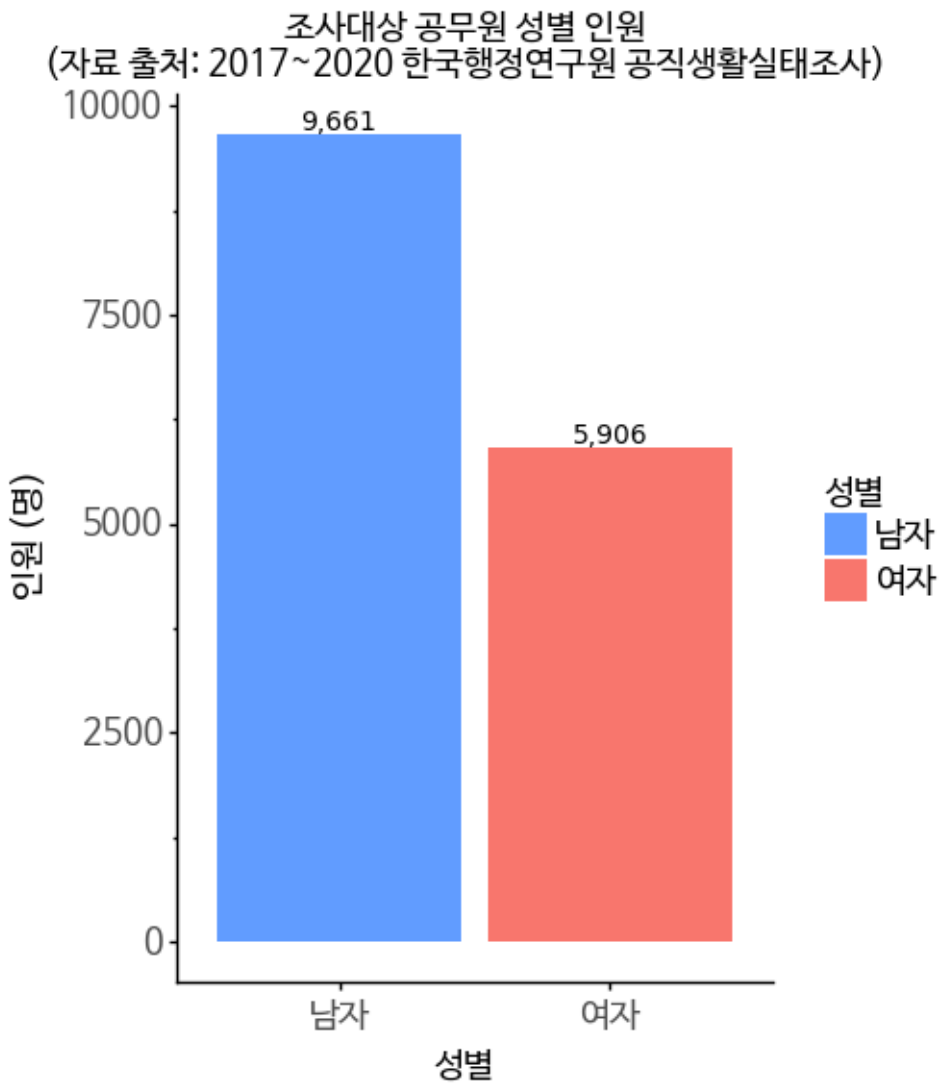
```
rank            0
marriage        0
children        0
workamount      0
extrawork       0
orglevel        0
promotion_1     0
promotion_2     0
promotion_3     0
transform_1     0
transform_2     0
tranform_3      0
culture_per_1   0
culture_per_2   0
culture_tra_1   0
culture_tra_2   0
culture_sta_1   0
culture_sta_2   0
culture_par_1   0
culture_par_2   0
mng_support_1   0
mng_support_2   0
mng_support_3   0
dtype: int64
```

```
gender=pd.DataFrame(kipa_pre['gender'].value_counts())
gender[" "]=[" "," "]
gender.columns=["value","gender"]
```

```
plt.figure()
pn.options.figure_size = (4,6)
(ggplot(gender,aes(x='gender',y="value",fill="gender"))
    + geom_bar(stat='identity',position = position_dodge(width = 0.9))
    + theme_classic()
    + ylab("  ( )")
    + xlab(" ")
    + theme(text=element_text(fontproperties=font))
    + labs(fill=" ")
    + scale_fill_manual(values=("#619CFF","#F8766D"))
    + geom_text(aes(label='value'),size=10, va='bottom', format_string='{:,}')
    + ggtitle('         \n(    : 2017~2020          )'))
```

```
/usr/local/lib/python3.7/dist-packages/plotnine/utils.py:1246: FutureWarning: is_categorical
  if pdtypes.is_categorical(arr):
```

```
<Figure size 432x288 with 0 Axes>
```




```
<ggplot: (8753139113205)>
```

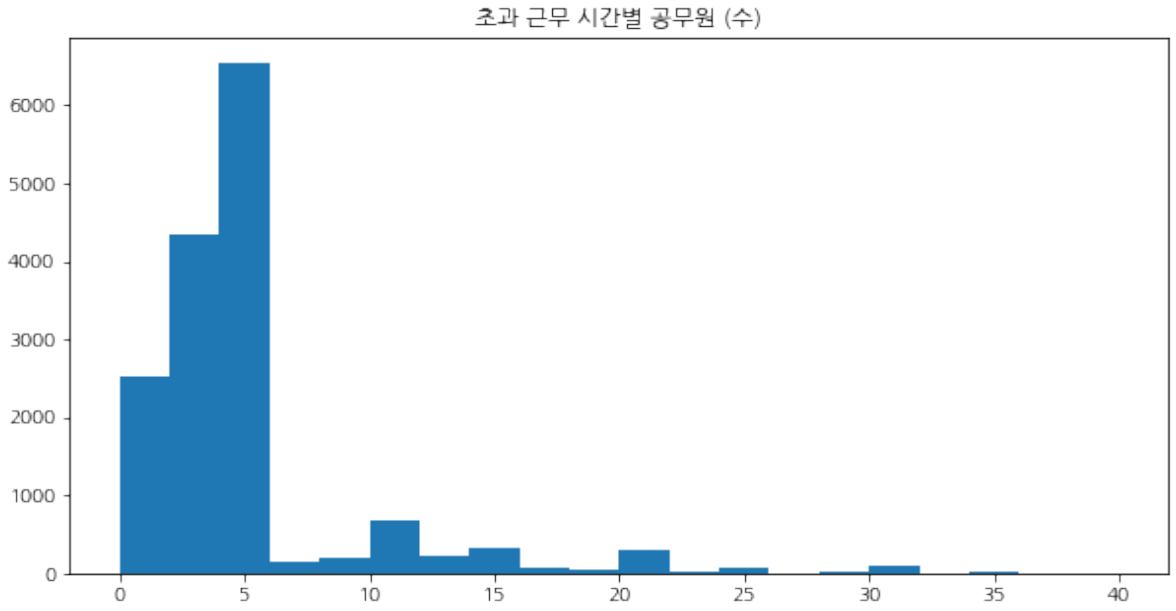
```
kipa_pre['gender']=pd.Categorical(kipa_pre['gender'].str.replace("1","0").str.replace("2",
```

•

```
for i in ["year","fwa","smartwork","age","work_year","rank","marriage","children","orgleve
  kipa_pre[i]=pd.Categorical(kipa_pre[i])
```

```
kipa_pre["extrawork"]=kipa_pre["extrawork"].astype(np.float)
```

```
#
plt.figure(figsize = (10,5))
plt.hist(kipa_pre["extrawork"],bins=20)
plt.title("          ( )")
plt.show()
```




```
for i in kipa_pre.columns:
  if i not in ["year","fwa","smartwork","gender","age","work_year","rank","marriage","chil
    kipa_pre[i]=pd.Categorical(kipa_pre[i], ordered=True)
```

•

```
import pandas_profiling
kipa_pre.profile_report()
```

- : Correlation Matrix and Plot Cramer's V correlation . :
  V (Cramér's V) . . 0 1 . 0.6 0.2~0.6
  . –> .

```
 #
data=kipa_pre >> select(~X.extrawork,   ~X.work_year)
```

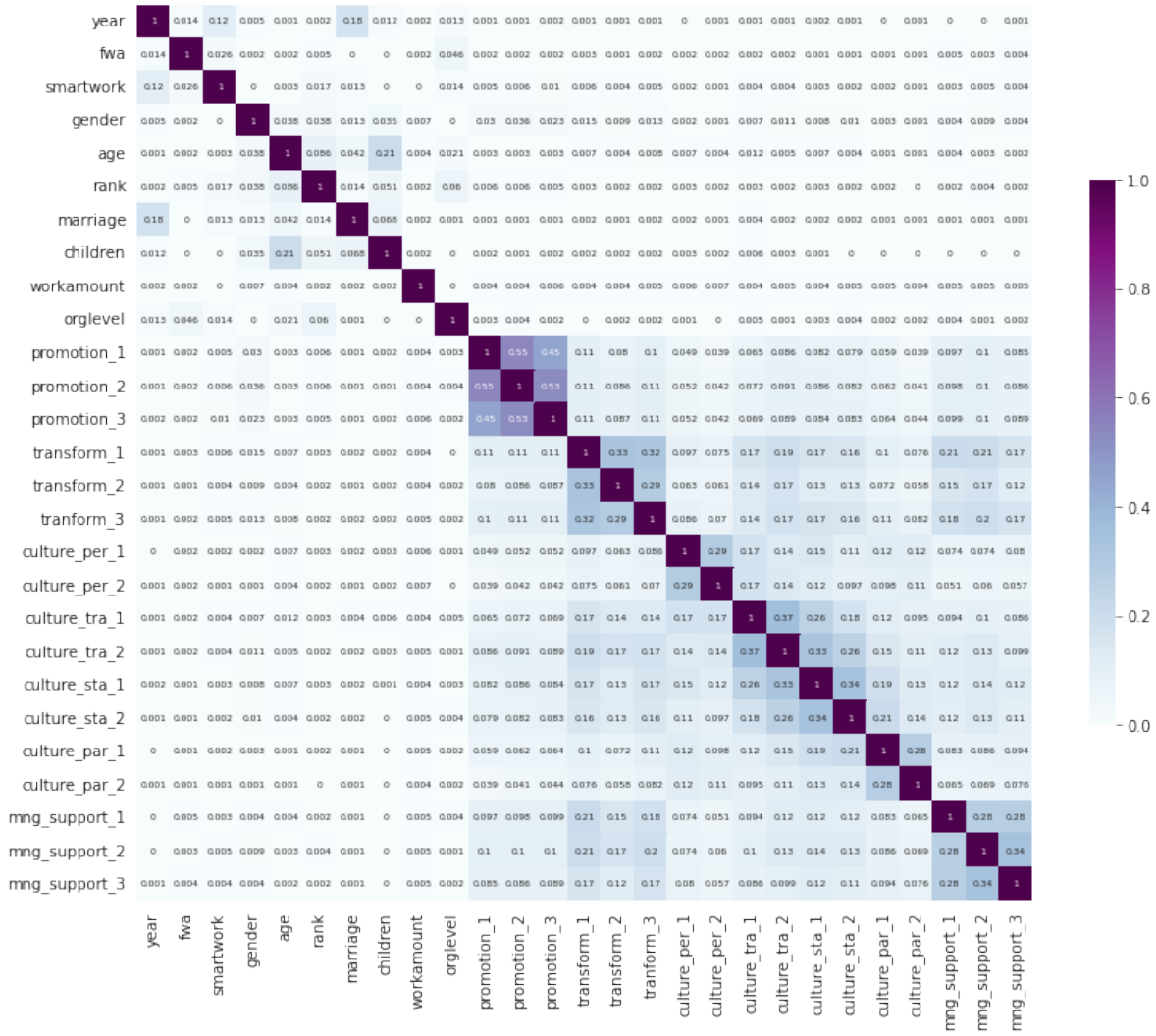
```
from sklearn import preprocessing

label = preprocessing.LabelEncoder()
data_encoded = pd.DataFrame()

for i in data.columns :
  data_encoded[i]=label.fit_transform(data[i])
```

```
# Building of the Cramer's V function
from scipy.stats import chi2_contingency

def cramers_V(var1,var2) :
  crosstab =np.array(pd.crosstab(var1,var2, rownames=None, colnames=None)) # Cross table b
  stat = chi2_contingency(crosstab)[0] # Keeping of the test statistic of the Chi2 test
  obs = np.sum(crosstab) # Number of observations
  mini = min(crosstab.shape)-1 # Take the minimum value between the columns and the rows o
  return (stat/(obs*mini))
```

```
rows= []
for var1 in data_encoded:
  col = []
  for var2 in data_encoded :
    cramers =cramers_V(data_encoded[var1], data_encoded[var2]) # Cramer's V test
    col.append(round(cramers,3)) # Keeping of the rounded value of the Cramer's V
  rows.append(col)
cramers_results = np.array(rows)
df = pd.DataFrame(cramers_results, columns = data_encoded.columns, index =data_encoded.col

df
```

```
plt.figure(figsize = (13,13))
with sns.axes_style("white"):
  ax = sns.heatmap(df,vmin=0., vmax=1, square=True,cbar_kws={"shrink": .5},cmap="BuPu",ann
```

```
plt.show()
```



- (Decision Tree, Random Forest)

```
!pip install graphviz
```

```
Requirement already satisfied: graphviz in /usr/local/lib/python3.7/dist-packages (0.10.1)
```

```
from sklearn.model_selection import train_test_split
import sklearn.metrics as mt

# sklearn    tree import
from sklearn import tree
```

```
data2=kipa_pre[kipa_pre["year"]<=2019]
data2=data2 >> select(~X.year)
```

```
data2.iloc[:,2:]
```

	gender	age	work_year	rank	marriage	children	workamount	extrawork	orglevel	promoti
1	0	4	6	2	1	0	4	30.0	2	4
2	0	2	3	3	1	1	4	25.0	2	2
3	1	3	6	3	1	1	4	8.0	2	4
4	0	2	2	2	1	1	5	20.0	1	2
5	0	4	6	1	1	0	4	18.0	2	3
...	...	...	...	...	...	...	...	...	...	...
11224	0	4	6	1	1	0	5	2.0	2	4
11225	0	4	6	1	1	1	3	3.0	2	4
11226	0	4	6	1	1	0	3	3.0	2	4
11227	0	4	6	1	1	0	3	2.0	2	3
11228	0	4	6	1	1	0	3	2.0	2	4

```
DT_label=data2 >> select(X.fwa)
DT_data=data2.iloc[:,2:]
```

```
train_data,test_data,train_label,test_label=train_test_split(DT_data,DT_label,test_size=0.
```

```
# DT
dt_clf = tree.DecisionTreeClassifier()
dt_clf.fit(train_data,train_label)

#
pred_label = dt_clf.predict(test_data)
```

```
# Train            -
print('Train_Accuracy: ', np.round(dt_clf.score(train_data, train_label)),'\n')
```

```
Train_Accuracy:  1.0
```

```
# Test
accuracy = mt.accuracy_score(test_label, pred_label)
accuracy
#                .
```

```
0.5894924309884239
```

. : .

.

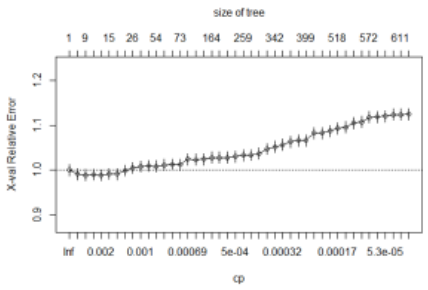
, , , R .

```
Image(filename='/31.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 결정나무

- Train data : Valid data = 7:3
- Train data를 활용해 결정나무를 만들고 나머지 Valid data를 활용해서 결정나무의 성능 평가
- Default tree는 가지를 깊이 뻗지 않는다는 점에서 깊이 있는 분석이 어려웠음.
- 따라서, cp값을 cpplot을 통해서 조정해서 최적의 cp값을 찾음 (2.128666e-03)
- 혼란행렬을 통해서 예측의 효과를 검증



```
Image(filename='/32.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 결정나무

		Observed			
		Present	Absent		
Predicted	Present	a	b	Positive predictive power	a/(a+b)
	Absent	c	d	Negative predictive power	d/(c+d)
		Sensitivity	Specificity	Accuracy = (a+d)/(a+b+c+d)	
		a/(a+c)	d/(b+d)		


```
Image(filename='/33.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 결정나무

- 분석 결과:
  Random하게 뽑았을 때 정확도: 0.5
  결정나무를 사용: 0.6654
  정확도가 생각보다 높지 않았음
- Sensitivity 가 높지만 Specificity가 심각하게 낮음
  (즉 유연근무제를 참여할 의향이 실제로 있는 사람들 중 참여할 것이라고 예측한 사람들의 비율은 높으나, 참여하지 않을 사람들 중 실제로 참여하지 않을 것이라고 예측한 사람의 비율은 낮다)
- 이러한 모델의 한계의 원인을 찾아보았는데, 공무원 개인의 소속기관이 중앙정부인지 광역자치단체인지에 의해서 유연근무제 사용 여부가 엄청난 차이를 보여서 그러함.
  (1= 중앙 / 0 = 지방자치단체)

```
> confusionMatrix(dt.validation.pred.class, valid.df$fwa, positive="1")
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0  157  136
         1  906 1915

               Accuracy : 0.6654
                 95% CI : (0.6485, 0.682)
    No Information Rate : 0.6586
    P-Value [Acc > NIR] : 0.2195

                  Kappa : 0.0986

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.9337
            Specificity : 0.1477
         Pos Pred Value : 0.6788
         Neg Pred Value : 0.5358
             Prevalence : 0.6586
         Detection Rate : 0.6150
   Detection Prevalence : 0.9059
      Balanced Accuracy : 0.5407

       'Positive' Class : 1
```



```
Image(filename='/34.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 결정나무 (1이 유연근무제 참여, 0이 미참여)

```
Image(filename='/35.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 랜덤포레스트

- 하이퍼파라미터인 ntree (만드는 여러 개의 나무의 개수), mtry (각 노드 설정 시 설명변수 후보 개수) 또한 학습을 통해서 확인해야 하지만 시간과 연산능력의 한계로 인해서 mtry는 모든 변수, ntree는 100, 500, 1000, 1500, 2000, 2500개로 분석. 그 중 나무의 개수를 100개로 설정한 것.
- 혼란행렬과 ROC 커브, 곡선 하 면적 (AUC) 등으로 모델을 평가하였으나 여전히 Specificity가 너무 낮게 나왔다는 단점이 있었음. 이는 데이터의 한계로 인해서 그러한 것인지 아니면 하이퍼파라미터 학습을 조금 더 시켜야 하는 것인지 확인이 필요해 보인다.
- 따라서, 이후 보다 본격적인 분석을 위해서는 설문조사 데이터를 보다 면밀하게 고찰할 필요성이 존재한다.

```
> confusionMatrix(rf.validation.pred.class, valid.df$fwa, positive="1")
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0  239  254
         1  824 1797

               Accuracy : 0.6538
                 95% CI : (0.6368, 0.6705)
    No Information Rate : 0.6586
    P-Value [Acc > NIR] : 0.7214

                  Kappa : 0.116

 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.8762
            Specificity : 0.2248
         Pos Pred Value : 0.6856
         Neg Pred Value : 0.4848
             Prevalence : 0.6586
         Detection Rate : 0.5771
   Detection Prevalence : 0.8417
      Balanced Accuracy : 0.5505

       'Positive' Class : 1
```

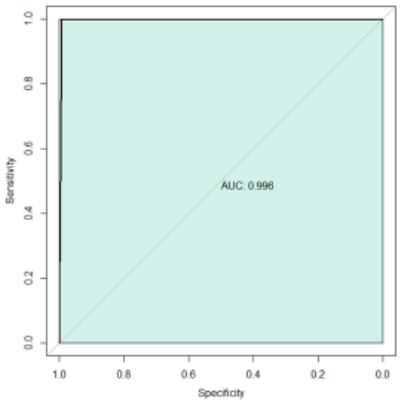


```
Image(filename='/36.png')
```

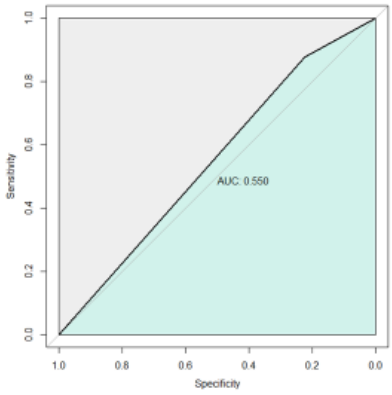
03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 랜덤포레스트

train data에 모델 적용시 심각한 오버 피팅



valid data에 모델 적용시 낮은 specificity의 문제

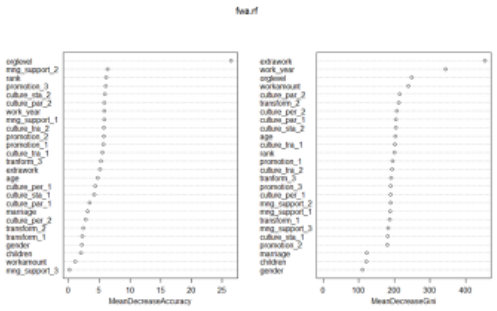



```
Image(filename='/37.png')
```

03/ Analysis 3 – Decision Tree / Random Forest
정부조직내 유연근무제 정책의 참여 요인은 무엇인가?

분석결과: 랜덤포레스트

- 오른쪽에 있는 변수중요도 Plot을 통해서 어떤 변수들이 유연근무제의 선택에 영향을 미쳤는지를 확인할 수 있었음.
- 참고)
- **The Mean Decrease Accuracy** expresses how much accuracy the model losses by excluding each variable
- **The Mean Decrease Gini** expresses how each variable contributes to the homogeneity of the nodes and leaves in the resulting random forest
- The Mean Decrease Gini를 기준으로 초과근무 시간, 재직기간, 조직 수준, 일의 양 등이 유연근무제를 참여하고 참여하지 않는 일관성을 결정하는데 영향을 많이 미친데 반해, 성별, 20세 이하의 아이가 있는지 여부, 혼인 여부 등은 큰 영향력을 미치지 못하였음. 선형모형에서 발견한 결과와 다소 일치하지 않음.



```
Image(filename='/38.png')
```

04/ 결론

결론: 분석 1)

- 유의미한 텍스트 마이닝을 위해서는 데이터의 원천에 대한 고찰이 필요하다
- 신문기사 특히 제목을 가지고 텍스트 마이닝을 하는 경우 어떤 맥락에 의해서 기사가 쓰였는지를 분류해야 하는데 이는 크롤링 특성상 많은 수의 기사들을 다루기 때문에 개별 기사들의 특성에 대해서 다소 무시하게 되는 문제가 발생한다.
- 따라서, 이후 텍스트 크롤링에 앞서서 먼저 그러한 텍스트의 일부 정도일지라도 분석가가 직접 읽고 나서 작성 맥락에 대해서 고려하고, 분석에 적합하지 않은 경우 그러한 문제를 어떻게 극복할 것인지에 대해서 고찰 할 필요성을 확인할 수 있었다.

결론: 분석 2) 3)

- **기존의 선행연구 들에서 사용한 일반화선형모형 기준 분석기법들은 성별, 혼인 여부, 20세 이하 자녀 존재 여부 등의 공무원 개인의 개인적 특성에 주목하였는데, 유연근무제를 사용할 수 있는 조직적 맥락이 더욱 중요하다는 것을 분석2)와 함께 보았을 때 다시 한번 확인할 수 있었다.**
- 따라서, 유연근무제가 현재 일가정양립이라는 정책적 목적을 달성하기 위해서는 보다 정교한 정책 설계가 요구되어질 것으로 생각된다. 부서평가 등으로 외부적 압력에 의해서 유연근무제를 사용하도록 하는 것은 장기적인 관점에서 오히려 선택율이 떨어지도록 하는 문제가 존재하므로 이러한 문제를 극복하기 위한 자발적 유인에 의한 선택이 가능해지도록 하는 인사제도 설계가 필요할 것으로 생각된다. 그러한 과정에 있어서, 어떤 요인에 의해서 유연근무제를 선택할 것인지를 예측할 수 있는 모형을 만드는 것이 도움이 될 것으로 생각된다. 이번 분석에서 specificity가 낮다는 모델의 심각한 한계에도 불구하고 일반화 선형모형들에서 확인할 수 없는 여러 변수들의 직간접적인 상호작용을 확인할 수 있었다는 점에서 Tree류 (Decision Tree, Xgboost, Random Forest 등) 기반 모델들의 가능성을 확인하였다. (cf. blackbox – Deep learning, SVM, Naïve Bayes 등)


```
Image(filename='/39.png')
```

Citations

- Blau, P. M., (1964). Exchange and Power in Social Life. Wiley, New York.
- Gouldner, H. P.(1960), "Dimensions of organizational commitment," Administrative Science Quarterly, 4. 468-490.
- Croucher, R. & Kelliher, C. (2005). The right to request flexible working in Britain: The law and organizational realities. The International Journal of Comparative Labour Law and Industrial Relations, 21(3): 503-520.
- Choi, Sungjoo. (2017). Managing Flexible Work Arrangements in Government: Testing the Effects of Institutional and Managerial Support. Public Personnel Management. 47. 0091026017738540. 10.1177/0091026017738540.
- Herzberg, Frederick. (1968). One More Time: How do You Motivate Employees?. Harvard business review. 81. 87-96. 10.1007/978-1-349-02701-9_2.
- Lee, D., & Kim, S. Y. (2018). A Quasi-Experimental Examination of Telework Eligibility and Participation in the U.S. Federal Government. Review of Public Personnel Administration, 38(4),
- Lee, Soo-Young & Hong, Jeong. (2011). Does Family-Friendly Policy Matter? Testing Its Impact on Turnover and Performance. Public Administration Review. 71. 870-879. 10.2307/41317386.
- Myungjung Kwon, Yoon Jik Cho, Hyun Jin Song (2019), How do managerial, task, and individual factors influence flexible work arrangement participation and abandonment?, Asia Pacific Journal of Human Resources
- 민경률, 박성민 (2013). 유연근무제가 조직결과에 미치는 영향력에 관한 연구. 한국정책논집, 25(4), 1211-1249
- 미수영, 홍정화 (2011) Does Family-Friendly Policy Matter? Testing its Impact on Turnover and Performance, Public Administration Review
- 이재호, 김태진 (2016), 유연근무제 확대에 따른 스마트워크 활성화 방안, 한국정책연구원.
- 이상천, (2014), 유연근무제(FlexibleWork)가 직무만족과 조직몰입에 미치는 영향에 관한 연구, 서울대 행정대학원
- 진종순, 장은명 (2015). 유연근무제와 직무만족:시차출퇴근제와 스마트워크제를 중심으로, 정부학연구, 236-263
- 정상호 (2016). 유연근무제 도입과 활용이 공기업 경영성과에 미치는 영향 분석, 서울대학교 행정대학원 1-80
- 정진택, 이윤룡 (2013). 스마트 유연근무제 유절에 관한 연구, 한성대학교 행정대학원
- 차동석-강대석(2006), "언식된 조직의 팀 지원과 변혁적 리더십이 팀 구성원들의 교환관계, 에피커시, 그리고 몰입에 미치는 영향," 인사관리연구, 30(4), 175-208.
- 황순옥, 한상일 (2013). 유연근무제 시절이 만족도와 효능감, 업무성과에 미치는 영향. 지방정부연구, 17(2), 391-414
- 행정규칙 (2014). 통계청 유연근무제 운영지침, 국가법령정보센터 Retrieved June 17, 2020, from http://www.law.go.kr/admRulInfoP.do;jsessionid=70BPwa0Ort7PRMMSYSLc1k7153xe511U54bbbkmf8zjdsc9mOg1aMMak1CGPVe.de_kl_a3_servlet_LSW3?admRulSeq=2100000007669
- 공공기관 알리오 (http://www.alio.go.kr/home.do)