

# Smart Career Project

홍성민, 김은비, 김은수, 백서연

Mentor: 송재희님

# Contents

01 프로젝트 개요

02 방법론

03 결과

04 Q&A



# 01

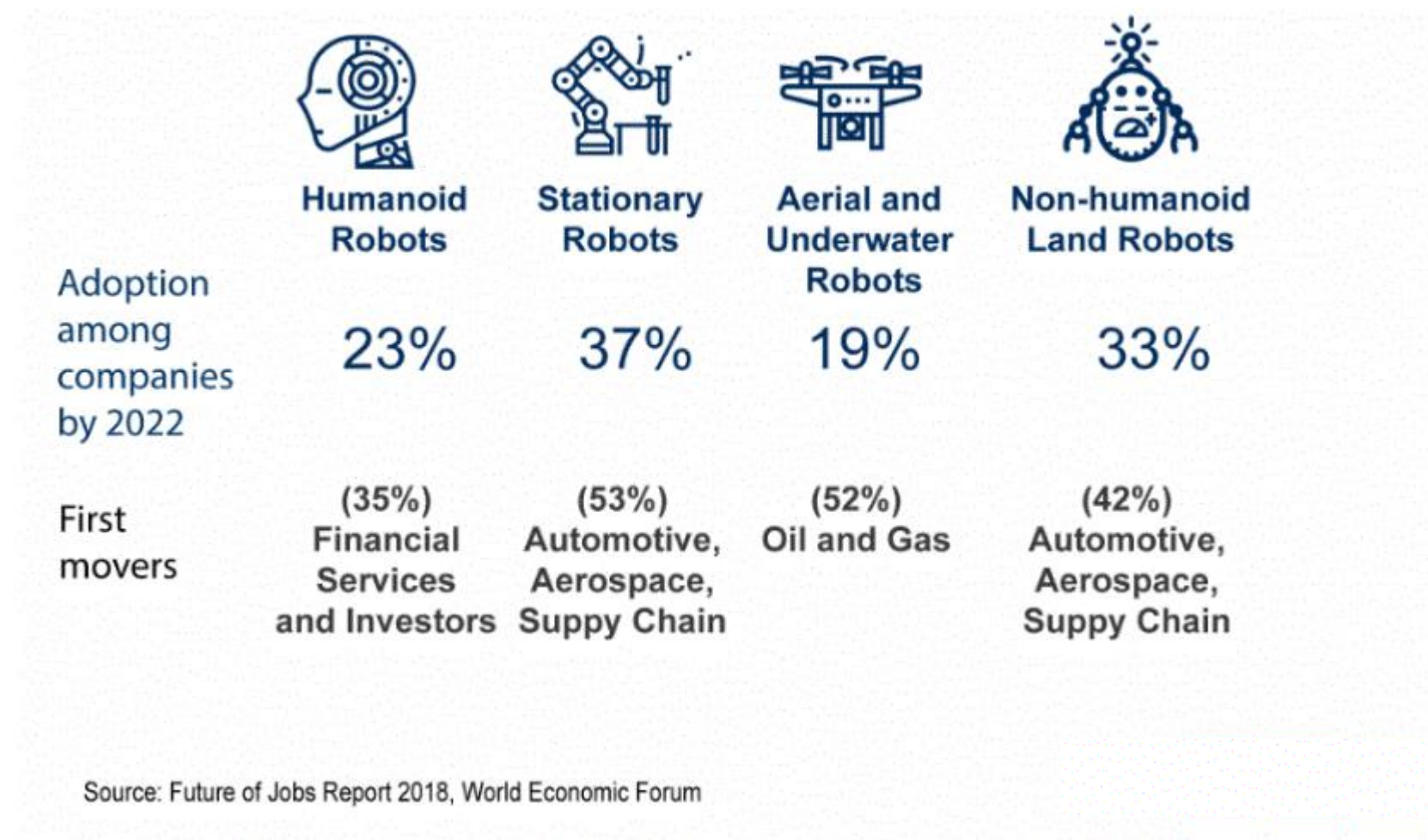
AI, Robotics, 5G, 자율주행차 등 새로운 기술의 등장...

## ▼ 프로젝트 개요

방법론

결과

Q&A



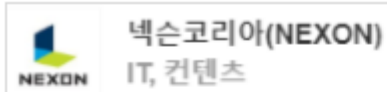
이에 따라 급변하는 구직 시장

## ▼ 프로젝트 개요

방법론

결과

Q&amp;A



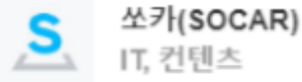
## 자격요건

- (필수) 게임에 대한 열정과 및 게임 산업에 대한 이해
- (필수) 논리적이고 체계적인 문제 해결 및 커뮤니케이션 능력
- (필수) 새로운 기술 습득에 능동적이고 거부감이 없는 분
- (선택) R 혹은 Python을 이용한 데이터분석 능력
- (선택) SQL 기본 활용 능력
- (선택) 통계 및 분석에 대한 이론 지식
- (선택) 시각화 도구 활용 능력(예: Tableau 등)



## [우대 조건]

- react.js, angularJS, vue.js 등 신기술 트렌드 파악 및 이해
- 반응형 웹 개발 경험 보유자
- Java/Spring 기반 개발 경력 1년 이상
- DW/OLAP 개발에 대한 관심과 이해가 있는 자
- Tableau, Spotfire 솔루션을 통한 데이터 시각화 가능자



## 우대사항

- 불확실한 데이터에서 확신을 가질 수 있도록 확률과 통계 및 데이터에 대한 지식을 가지신 분
- BigQuery (SQL), MySQL, Tableau, Pandas, Spark, TensorFlow, Excel (Google Sheet) 등 데이터 분석 도구를 원활히 다루시거나, 앞으로 원활히 다루고자 하는 의지가 높으신 분
- Python, R 등의 프로그래밍 언어 혹은 데이터 분석 언어를 1개 이상 능숙하게 사용 가능하신 분
- AWS, GCP 등의 클라우드 서비스 사용 경험이 있으신 분

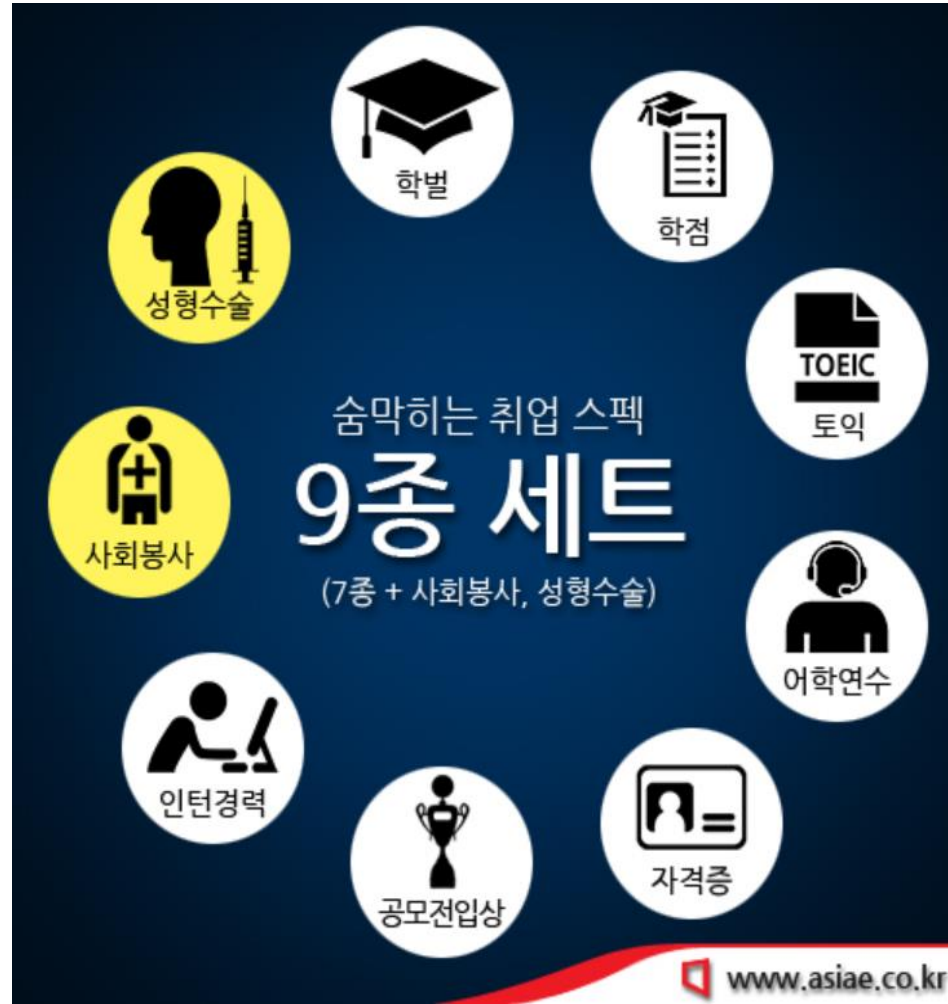
더 험난해진 Dream Job을 얻기 위한 여정

## ▼ 프로젝트 개요

방법론

결과

Q&amp;A



## 기업이 원하는 스펙, 원하지 않는 스펙

불필요한 스펙 (단위: %, 중복 응답)



지원자가 갖춰야 할 스펙 (단위: %, 중복 응답)



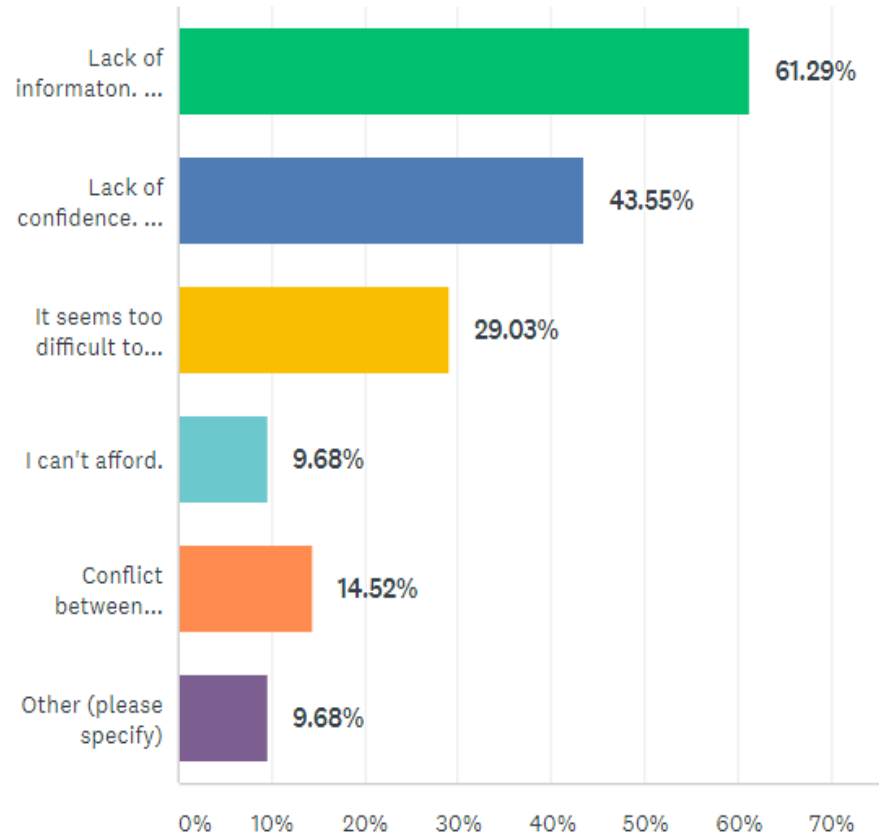
다양한 의견과 정보들

# 01

## 불투명한 미래 때문에

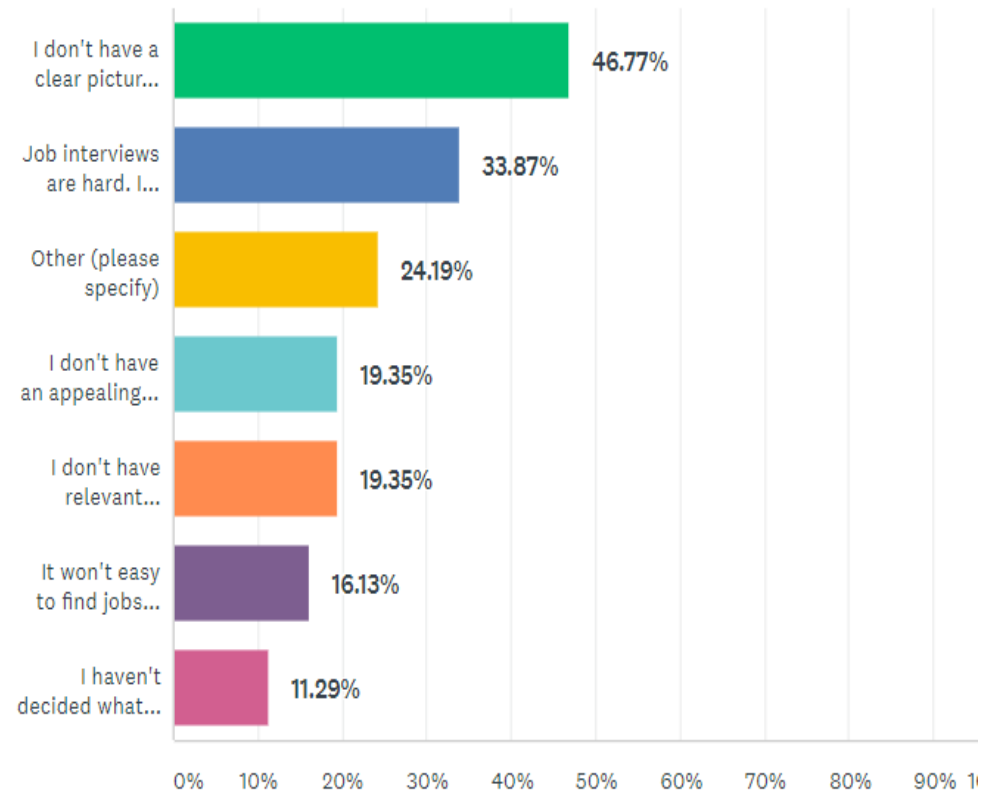
Why was it hard to choose your career?

Answered: 62 Skipped: 23



What are your pain points in getting a job or career development? (Select all that apply.)

Answered: 62 Skipped: 23



### ▼ 프로젝트 개요

방법론

결과

Q&A

다양한 컨설팅 서비스를 사용하는 하는 구직자들





# 01

편향성을 극복하고 개인화된 컨설팅 위해 빅데이터를 이용하는 것이 어떨까?

## ▼ 프로젝트 개요

방법론

결과

Q&A



이렇게 시작된 프로젝트 “SmartCareer”



# 01

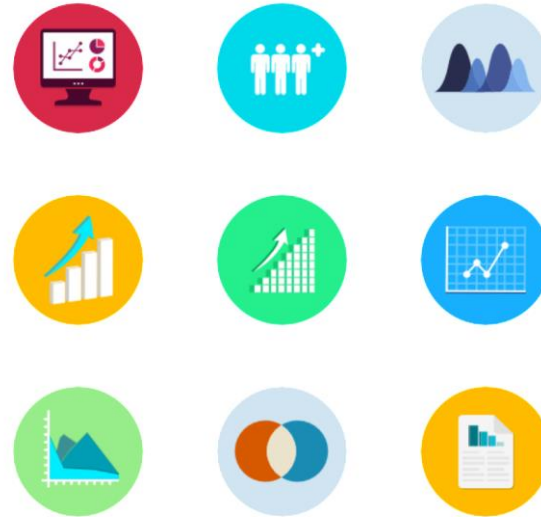
## 웹에 있는 이력서를 데이터베이스화 하여

### ▼ 프로젝트 개요

방법론

결과

Q&A



키워드에 맞는 정보를 시각화하여 유저에게 제공

# 02

## General Process

프로젝트 개요

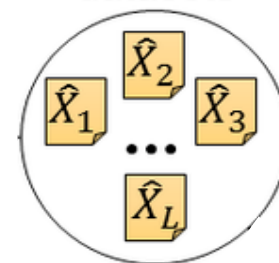
▼ 방법론

결과

Q&A



Web Scraping



Pre-processing



Data Exploration



Data Visualization

프로젝트 개요

▼ 방법론

결과

Q&amp;A



Webpages



Web Scraping

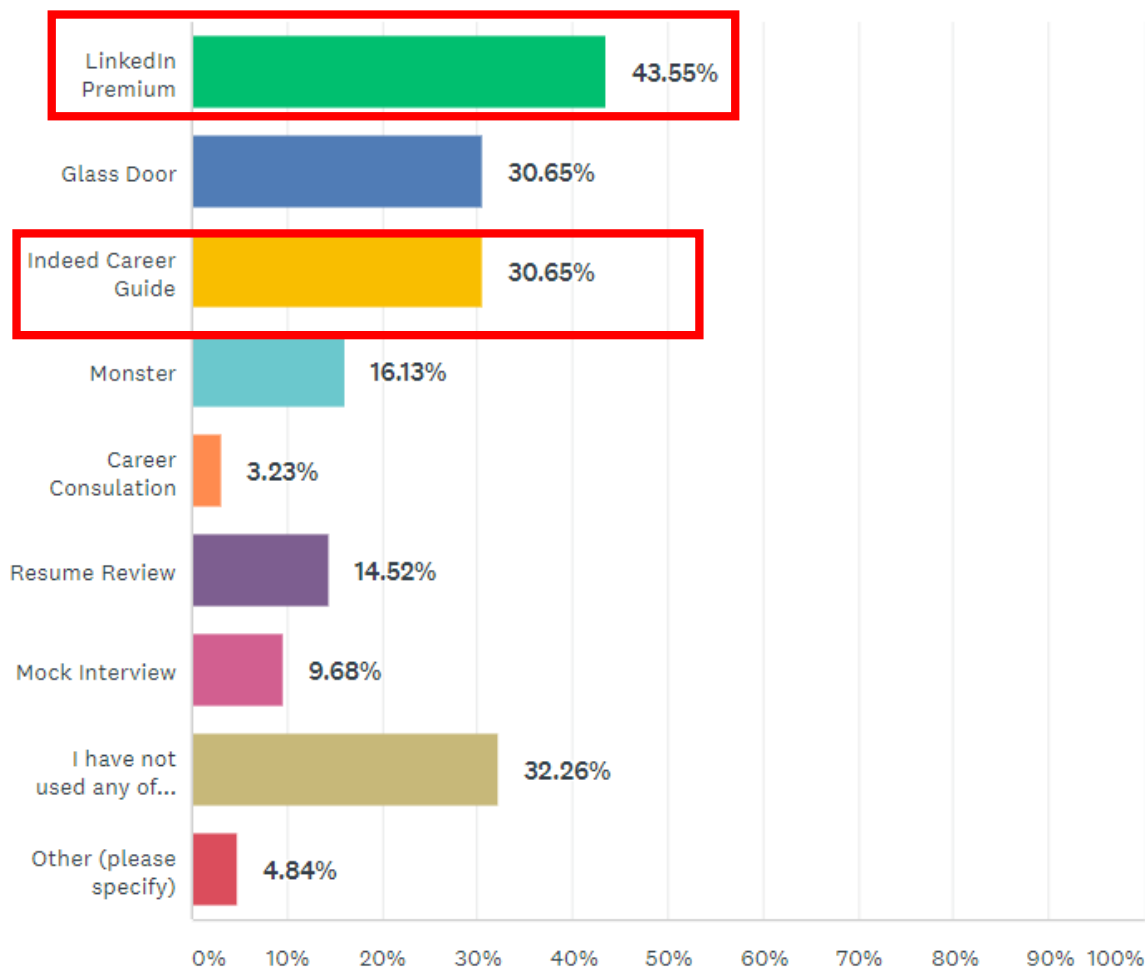


Mongo DB

# 1. Web Scraping

In the past 6 months, which of the following career services have you used? (Please select all that apply.)

Answered: 62 Skipped: 23



프로젝트 개요

▼ 방법론

결과

Q&A

## 1. Web Scraping: LinkedIn vs. Indeed







프로젝트 개요

▼ 방법론

결과

Q&amp;A



BeautifulSoup		
Scrapy		
Selenium		



## 02

## 2. Data Exploration

프로젝트 개요

▼ 방법론

결과

Q&amp;A

Keyword

Analyst

Data Engineer

Len(data)

1890

799

Example

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1890 entries, 0 to 1889
Data columns (total 15 columns):
   Date Captured      437 non-null object
   Education          1890 non-null object
   Experience          1890 non-null object
   Industry Knowledge  1079 non-null object
   Interpersonal Skills 927 non-null object
   Job Title          1890 non-null object
   Languages          151 non-null object
   Location           1890 non-null object
   Other Skills       851 non-null object
   Profile Summary    1890 non-null object
   ProfileID          1890 non-null object
   Skills & Endorsements 1890 non-null object
   Tools & Technologies 1021 non-null object
   _id                1890 non-null object
dtypes: object(15)
memory usage: 221.6+ KB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 799 entries, 0 to 1006
Data columns (total 11 columns):
   Education          799 non-null object
   Experience          799 non-null object
   Industry Knowledge  715 non-null object
   Interpersonal Skills 276 non-null object
   Job Title          799 non-null object
   Languages          24 non-null object
   Location           799 non-null object
   Other Skills       700 non-null object
   ProfileID          799 non-null object
   Skills & Endorsements 799 non-null object
   Tools & Technologies 725 non-null object
dtypes: object(11)
memory usage: 74.9+ KB
```

프로젝트 개요

▼ 방법론

결과

Q&amp;A



mongoDB



pymongo

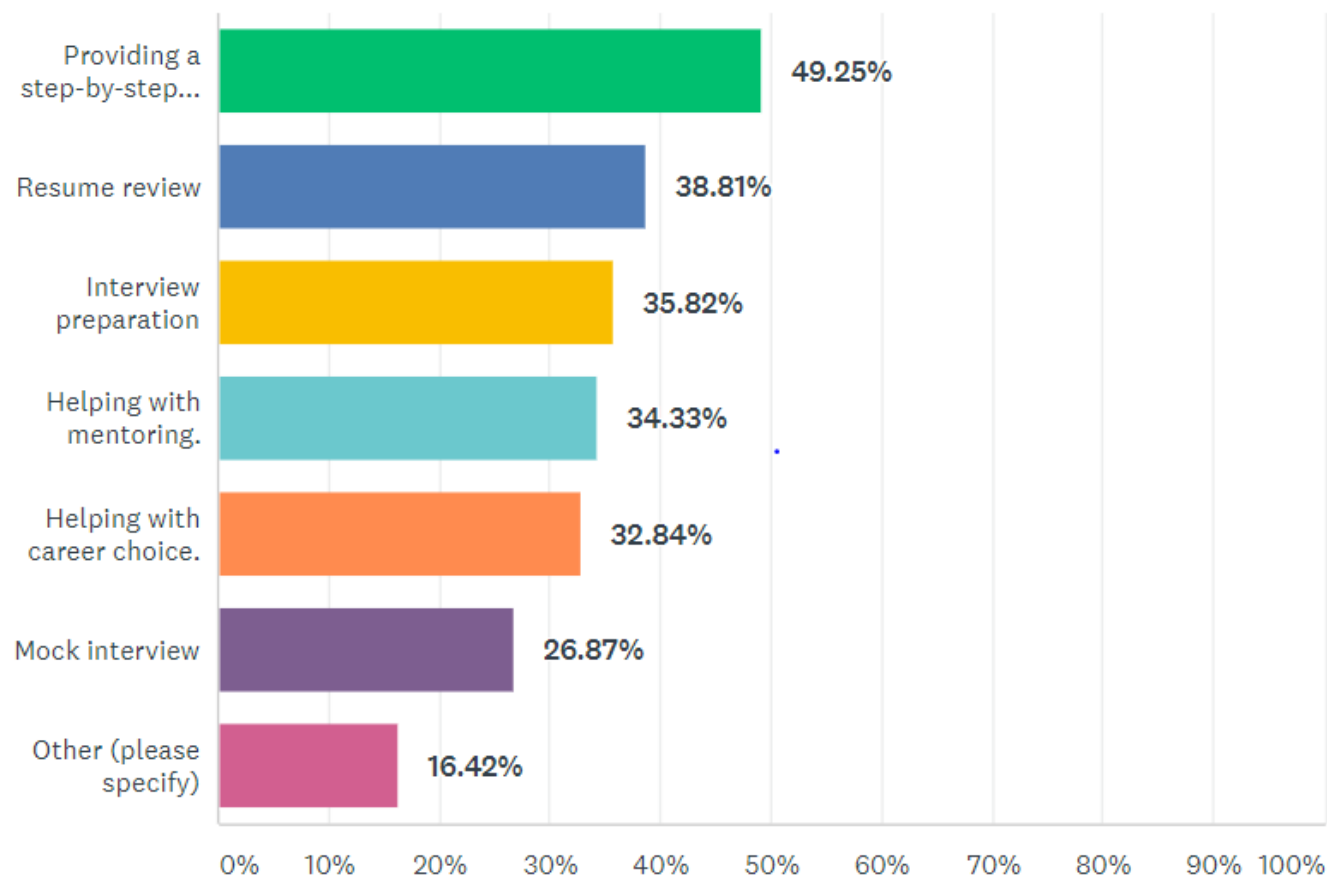


python

## 2. Data Exploration

Which of the following services would you be willing to pay for?

Answered: 67 Skipped: 18



프로젝트 개요

▼ 방법론

결과

Q&A

# 02

## 2. Data Exploration

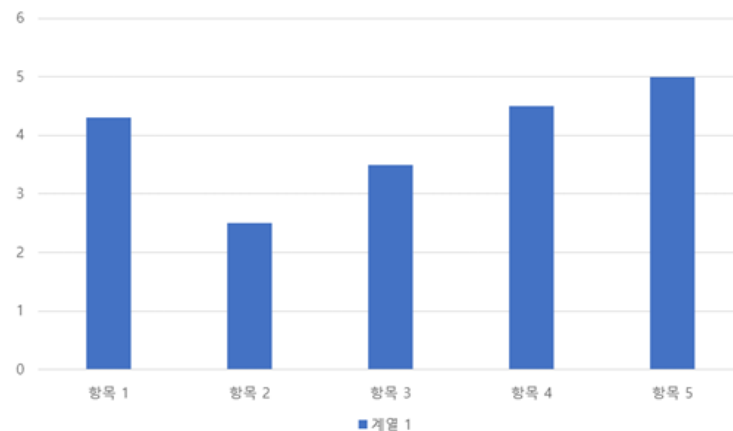
프로젝트 개요

▼ 방법론

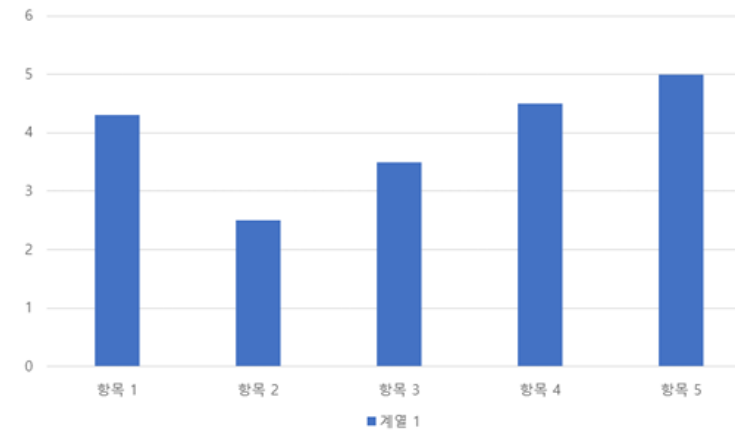
결과

Q&A

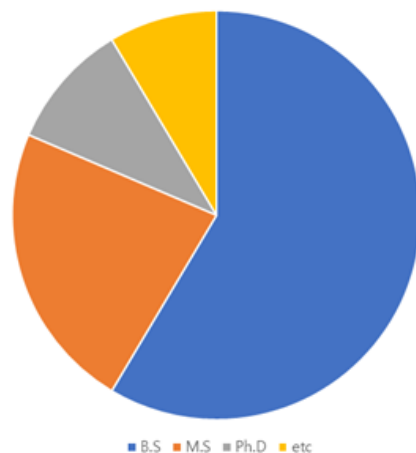
Top 10 Skill



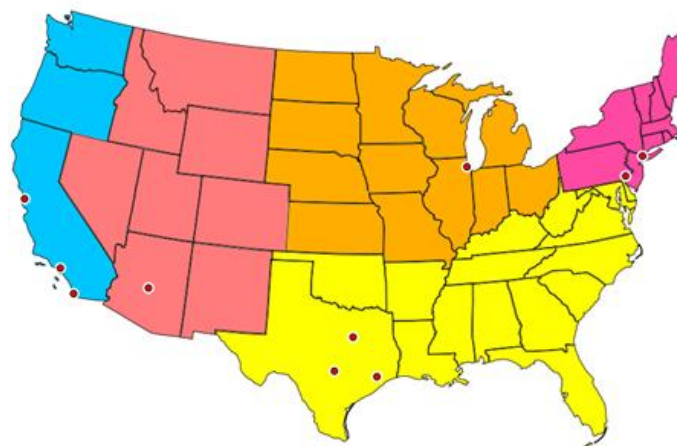
Top 10 Major



Degree



Location



Years of experience Word cloud



## job title normalization

```

1 # Job_list 소문자로 변경
2 job_list = []
3
4 for job in df['Job Title']:
5     job_list.append(job.lower())
6
7 job_list

```

```

['senior data engineer at change healthcare',
'sr. principal data engineer at coupang global llc.',
'data engineer at playstation',
'architect / data engineer',
'data scientist and ml engineer',
'data engineer at aetna, a cvs health company',
'data engineer at smartdrive systems',
'sr. data engineer at coupang',
'student at university of pennsylvania',
'incoming data engineer at amazon robotics',
'data engineer at united technologies',
'data engineer at google',
'data engineer at facebook',
...

```

```

de_list = ['engineer', 'platform', 'infrastructure', 'science',

```

```

1 check=[]
2 del_list = []
3
4 for i, job in enumerate(job_list):
5     for de in de_list:
6         check.append(de in job)
7     # print(check)
8     if 'data' in job and any(check):
9         pass
10    else:
11        del_list.append(i)
12    del check[:]
13 print(del_list)
14 len(del_list)

```

```

[8, 13, 16, 23, 24, 156, 157, 158, 159, 174, 313, 317, 344, 345, 366,
560, 561, 563, 571, 577, 601, 605, 607, 613, 626, 629, 632, 637, 645,
715, 721, 723, 724, 725, 728, 739, 743, 762, 781, 783]

```



## Education normalization

```
1 # 인덱스 재 설정 (중복행 제거로 뒤섞임)
2 df = df.reset_index(drop=True)
3 df.index
```

RangeIndex(start=0, stop=799, step=1)

```
1 df['Education'][0][0]
```

```
{'School': 'University of Washington',
'Degree': 'Biochemistry Applied Mathematics/Chemistry minor',
'Date Attended': '2008 ~ 2013'}
```

```
1 df['Education']
```

```
0    [{'School': 'University of Washington', 'Degree': 'Biochemistry Applied Mathematics/Chemistry minor', 'Date Attended': '2008 ~ 2013'}]
1    [{'School': 'Korea University', 'Degree': 'Doc...', 'Date Attended': '2008 ~ 2013'}]
2    [{'School': 'University of Virginia', 'Degree': '...', 'Date Attended': '2008 ~ 2013'}]
3    [{'School': 'University of California, Berkeley', 'Degree': '...', 'Date Attended': '2008 ~ 2013'}]
4    [{'School': 'Flatiron School', 'Degree': 'Data Science', 'Date Attended': '2008 ~ 2013'}]
5    [{'School': 'Carnegie Mellon University', 'Degree': '...', 'Date Attended': '2008 ~ 2013'}]
6    [{'School': 'University of Oregon', 'Degree': '...', 'Date Attended': '2008 ~ 2013'}]
7    [{'School': 'Sungkyunkwan University', 'Degree': '...', 'Date Attended': '2008 ~ 2013'}]
8    [{'School': 'The Wharton School', 'Degree': 'B...', 'Date Attended': '2008 ~ 2013'}]
9    [{'School': 'Cornell University', 'Degree': 'B...', 'Date Attended': '2008 ~ 2013'}]
10   [{'School': 'Western University', 'Degree': 'M...', 'Date Attended': '2008 ~ 2013'}]
```

```
1 # Education key 값 확인.
2 edu_key = []
3
4 for i in range(len(df)):
5     if df['Education'][i] == []:
6         pass
7     else:
8         for k in df['Education'][i][0].keys():
9             edu_key.append(k)
10
11 set(edu_key)
```

{'Date Attended', 'Degree', 'School'}

```
1 df['School'] = None
2 df['Degree'] = None
3 # df['Date Attended'] = None
4
5 for i in range(len(df)):
6     if df['Education'][i] == []:
7         pass
8     else:
9         df['School'][i] = df['Education'][i][0]['School']
10        df['Degree'][i] = df['Education'][i][0]['Degree']
11        # df['Date Attended'][i] = df['Education'][i][0]['Date Attended']
```

## Location normalization

```
1 df_loc = pd.DataFrame(df['Location'].str.replace(' ', '').str.replace(',', '').str.lower())
2 df_loc
3 df_loc.to_csv('df_loc.csv')
```

## city, state list 불러오기

```
1 city_state = pd.read_csv('city_state.csv')
2 city_state
```

	city	state	state_s	city_s
0	New York	New York	newyork	newyork
1	Los Angeles	California	california	losangeles
2	Chicago	Illinois	illinois	chicago
3	Houston	Texas	texas	houston
4	Phoenix	Arizona	arizona	phoenix
5	Philadelphia	Pennsylvania	pennsylvania	philadelphia
6	San Antonio	Texas	texas	sanantonio
7	San Diego	California	california	sandiego
8	Dallas	Texas	texas	dallas
9	San Jose	California	california	sanjose
10	Austin	Texas	texas	austin
11	Jacksonville	Florida	florida	jacksonville
12	Fort Worth	Texas	texas	fortworth

```
1 df_loc = pd.DataFrame(df['Location'].str.replace(' ', '').str.replace(',', '').str.lower())
```

```
1 for i in range(len(df_loc)):
2     for j in range(len(city_state)):
3         if city_state['city_s'][j] in df_loc['Location'][i]:
4             df['City'][i] = city_state['city'][j]
5             df['State'][i] = city_state['state'][j]
6             break
7
8         elif city_state['state_s'][j] in df_loc['Location'][i]:
9             df['State'][i] = city_state['state'][j]
10            break
11
12 df.head()
```

## Years Experience

```
1 df_exp = df[['Experience']]
2 df_exp
```

## Experience

```
0    [{'Job Title': 'Senior Data Engineer', 'Experience': '10 yrs'}]
1    [{'Job Title': 'Director of Research', 'Experience': '10 yrs'}]
2    [{'Job Title': 'Data Engineer', 'Experience': '10 yrs'}]
3    [{'Job Title': 'Senior Site Reliability Engineer', 'Experience': '10 yrs'}]
4    [{'Job Title': 'Data Scientist', 'Experience': '10 yrs'}]
5    [{'Job Title': 'Data Engineer', 'Experience': '10 yrs'}]
```

```
1 def get_year(tr):
2     mo_tr = 0
3     ye_tr = 0
4     year = 0
5
6     for i in range(len(tr)):
7         tr[i]['Years'] = tr[i]['Years'].replace(' ', '').replace('yrs', 'yr').replace('mos', 'mo')
8         if 'yr' in tr[i]['Years'] and 'mo' in tr[i]['Years']:
9             ye_tr += np.int(tr[i]['Years'].replace('mo', '').split('yr')[0])
10            mo_tr += np.int(tr[i]['Years'].replace('mo', '').split('yr')[1])
11        elif 'yr' not in tr[i]['Years'] and 'mo' in tr[i]['Years']:
12            mo_tr += np.int(tr[i]['Years'].replace('mo', ''))
13            ye_tr += 0
14        elif 'yr' in tr[i]['Years'] and 'mo' not in tr[i]['Years']:
15            ye_tr += np.int(tr[i]['Years'].replace('yr', ''))
16            mo_tr += 0
17        else:
18            ye_tr += 0
19            mo_tr += 0
20
21    year = int(round(((ye_tr*12)+mo_tr)/12,0))
22
23    return year
```

```
1 df['Exp_years'] = df_exp['Experience'].apply(lambda x: get_year(x))
```

프로젝트 개요

▼ 방법론

결과

Q&amp;A

# 02

프로젝트 개요

▼ 방법론

결과

Q&A

## 3. PRE-processing

### Skill

```
1  ## 바로 apply를 하면 nan은 float type이라서 len가 안먹힌다.  
2  ## 각 column을 공통 type으로 맞춰주기 위해 nan값을 찾아 빈리스트로 대체  
3  def nan_to_list(num):  
4      if num!=num:  
5          return list()  
6      else:  
7          return num  
8  
9  for i in df:  
10     df[i] = df[i].apply(lambda x: nan_to_list(x))
```

```
1  ## 복잡한 형태의 column을 일정한 형태로 맞춰주기위함  
2  def prep_edu(ro):  
3      if ro == []:  
4          ro = [{'School': 'No School', 'Degree': 'No Degree', 'Date Attend': 'No attend'}]  
5      else:  
6          ro=ro  
7      return ro  
8  
9  def prep_exp(ro):  
10     if ro== []:  
11         ro = [{'Job Title': '', 'Company': '', 'Period': '', 'Years': '', 'Location': '', 'Description': ''}]  
12     else:  
13         ro = ro  
14     return ro  
15  
16 def prep_sk(ro):  
17     if ro== []:  
18         ro=[{'Skills': ''}]  
19     else:  
20         ro=ro  
21     return ro  
22
```

# 02

프로젝트 개요

▼ 방법론

결과

Q&A

## 3. PRE-processing

```
1 df['Skills & Endorsements'] = df['Skills & Endorsements'].apply(lambda x:prep_sk(x))
2 df['Tools & Technologies'] = df['Tools & Technologies'].apply(lambda x:prep_sk(x))
3 df['Interpersonal Skills'] = df['Interpersonal Skills'].apply(lambda x:prep_sk(x))
4 df['Other Skills'] = df['Other Skills'].apply(lambda x:prep_sk(x))
5 df['Industry Knowledge'] = df['Industry Knowledge'].apply(lambda x:prep_sk(x))
6 df['Experience'] = df['Experience'].apply(lambda x:prep_exp(x))
7 df['Education'] = df['Education'].apply(lambda x:prep_edu(x))
```

```
1 ## nlp에 필요한 function정의
2 def remove_punct(text):
3     text_nopunct = "".join([char for char in text if char not in string.punctuation])
4     return text_nopunct
5 def tokenize(text):
6     tokens = re.split('\\\\W+', text)
7     return tokens
8 def remove_stopwords(tokenized_list):
9     text = [word for word in tokenized_list if word not in stopwords]
10    return text
11 def lemmatizing(tokenized_text):
12    text = [wn.lemmatize(word) for word in tokenized_text]
13    return text
```

```
1 ## skill들을 빼올수 있는 함수 생성
2 def get_skill(sel):
3     skill_list = []
4     for i in range(len(sel)):
5         skill_list.append(sel[i]['Skills'])
6     return skill_list
```

```
1 for col in ['Skills & Endorsements', 'Tools & Technologies', 'Interpersonal Skills', 'Other Skills', 'Industry Knowledge']:
2     df[col] = df[col].apply(lambda x:get_skill(x))
3
4 df.head()
```



## Degree Normalization

```
1 df_deg = pd.DataFrame(df['Degree'].str.replace('.', '').str.replace(' ', '').str.lower())
2 df_deg
```

	Degree
0	biochemistryappliedmathematics/chemistryminor
1	doctorofphilosophy(phd)computerscience
2	masterofscience-msstatistics
3	bselectricalengineeringandcomputerscience
4	datasciencedatascienceimmersivebootcamp
5	bachelorofscience-bsstatisticsandmachinelearning
6	bachelor'sdegreecomputerandinformationsciences...
7	masterofengineering(meng)itconsulting,
8	bachelorofscience-bscomputerscience
9	masterofscience(msc)computerscience
10	bachelor'sdegreecomputerscience
11	bachelorofscience-bscomputerscience
12	master'sdegreecomputerscience3909/40gpa

```
1 # degree 분류 함수
2 def deg_class(tr):
3     if tr[0:1]=='m':
4         ret = 'Master'
5     elif tr[0:2]=='bs' or tr[0:2]=='ba' or tr[0:2]=='be' or tr[0:2]=='bt':
6         ret = 'Bachelor'
7     elif tr[0:2]=='no':
8         ret = 'No Degree'
9     elif tr[0:2]=='do' or tr[0:2]=='ph':
10        ret = 'Ph.D'
11    else:
12        ret = 'etc'
13    return ret
```

02

## 4. Visualization

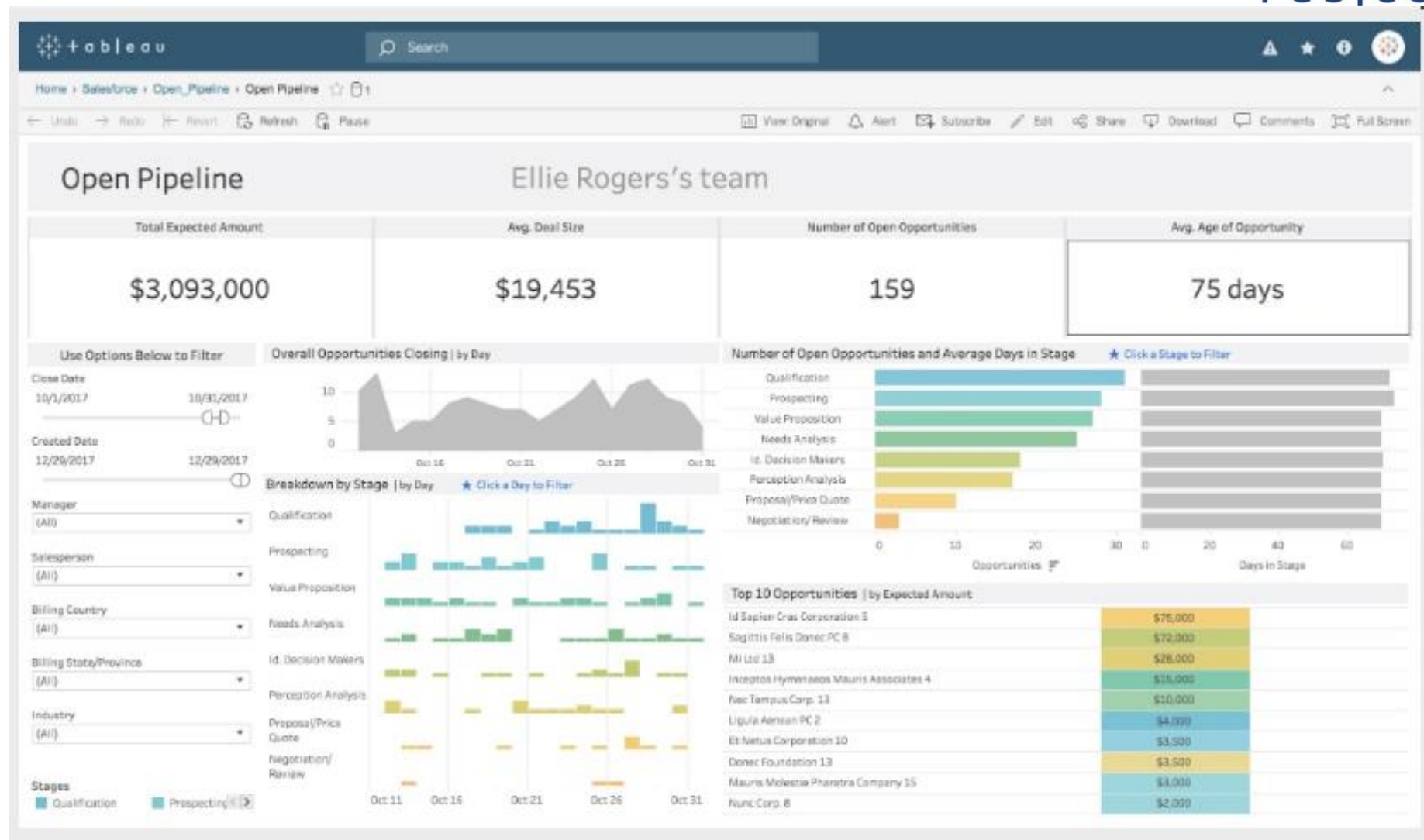


프로젝트 개요

▼ 방법론

결과

Q&amp;A



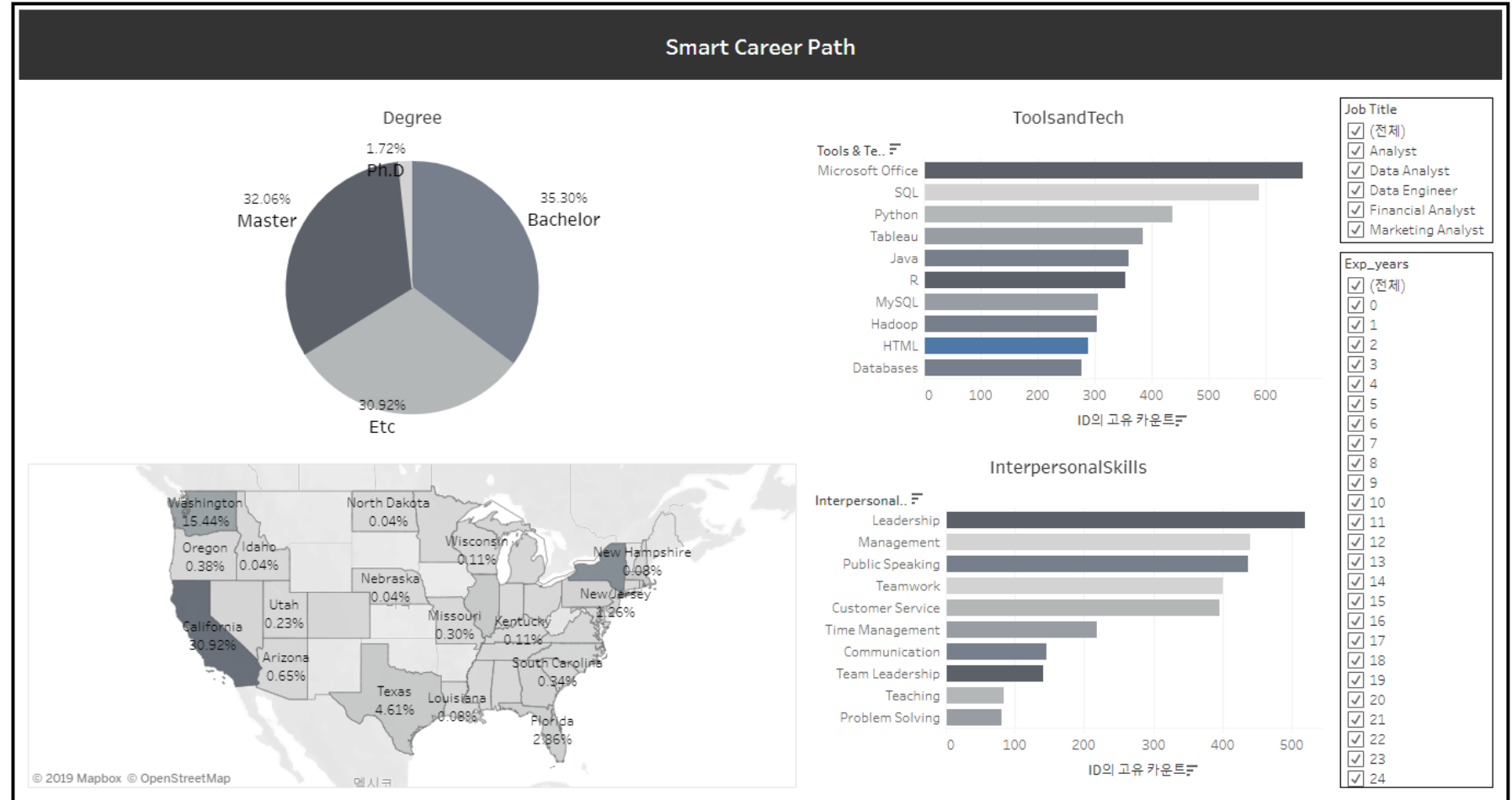
프로젝트 개요

방법론



결과

Q&amp;A



# 03

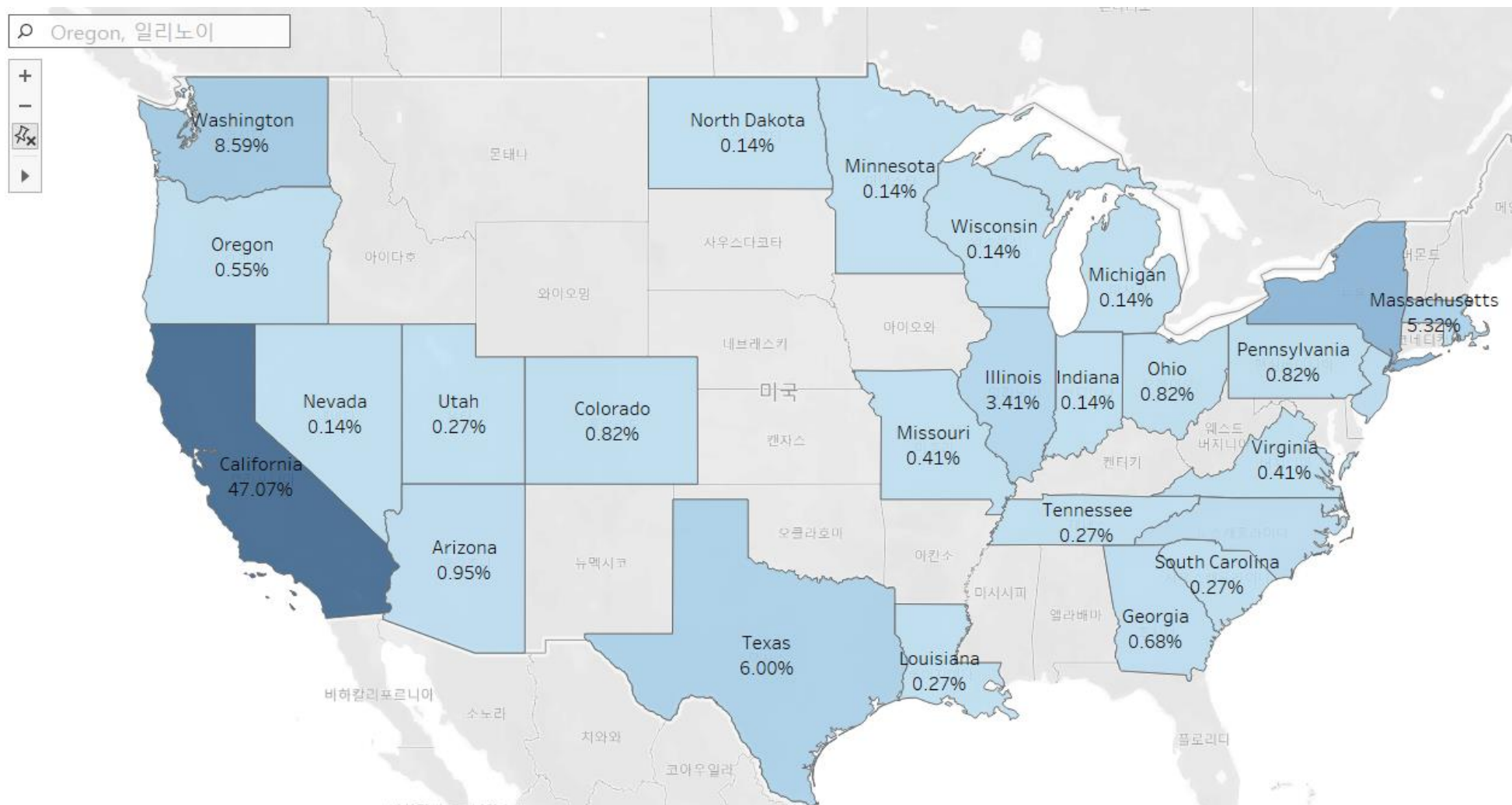
## Tableau 예시 : Data Engineer 직업이 가장 많은 곳은 어디인가?

프로젝트 개요

방법론

▼ 결과

Q&A



# 04

## 느낀점

프로젝트 개요

방법론

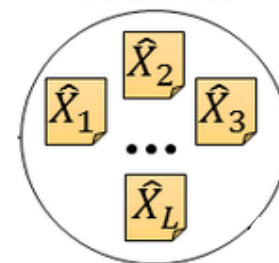
결과



Q&A



Web Scraping



Pre-processing



Data Exploration



Data Visualization



# 04

## 프로젝트 개요

## 방법론

## 결과



## Q&A

01 AI, Robotics, SG, 자율주행차 등 새로운 기술의 등장

이제 따라 갈뻔하는 구조 시장

01 구직자들에게 요구되는 점점 다양한 능력

여 힘낸세진 Dream Job을 찾기 위한 여정

01 일반화된 취업과정 앞에 남으려는

다양한 의견과 정보들

01 선택할 때의 고민

다양한 취업 정보 서비스들을 제공하는 것은 구조적일

01 현재 취업이 현실일 시장

전날선들의 경험의 바탕이되는 표현하기가 흔한데

01 인공지능과 관련된 다양한 직업의 미래

미래를 위한 프로젝트 "SmartCareer"

01 현재 있는 직업들을 데이터베이스화 하여

기업문화 또는 정보를 시각화하여 유익하게 제공

02 General Process

02 1. Web Scraping

02 1. Web Scraping

02 1. Web Scraping: LinkedIn vs. Indeed

02 2. Data Exploration

02 2. Data Exploration

02 2. Data Exploration

02 3. PRE-processing: Job Title

02 3. PRE-processing: Education

02 3. PRE-processing: Location

02 3. PRE-processing: Years of Experience

02 3. PRE-processing: Skills

02 3. PRE-processing: Other Information

Thank You