Kickstarter 사이트의 펀딩 페이지의 content 및 risk and challenges 텍스트 마이닝

- 2015년~2019년 데이터만 수집
- 총 수집된 데이터 수/행 : 153,576 (608.5MB)

In [ ]:
```
# Kickstarter의 펀딩 페이지는 javascript를 통해 동적으로 텍스트 내용이 불러와지기에 Requests가 아닌 Selenium 사용
# 웹크롤링으로 데이터 수집 중 지속적으로 차단을 당하여 여러번의 수정을 거쳤다
# 크롤링 block 당했거나 오류가 났을 때, continue하고 다음 url로 바로 넘어가는게 아니라 약 2분 정도 간격으로 10번까지 더 시도해서 scrape하고 안 되면 continue되도록 설계
```

In [ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import os
import json
from datetime import datetime, timedelta

import time
import random
import requests
from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver import Chrome
from selenium.webdriver import ChromeOptions
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

options = ChromeOptions()
options.add_argument('headless')
options.add_argument("--user-agent=Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/79.0.3945.117 Safari/537.36")
#path = '/Users/Python/chromedriver'
#driver = Chrome(executable_path=path, options=options)
driver = Chrome(options=options)

content_list = []

for idx, url in enumerate(test):
    driver.get(url)
    time.sleep(random.uniform(8,20))
    req = driver.page_source
    soup = BeautifulSoup(req, 'lxml')
    try:
        content_tag = soup.select_one('div.rte__content')
        contents = content_tag.select('p')
        contents_collected = []
        for c in contents:
            content = c.get_text().strip()
            contents_collected.append(content)

        try:
            risk_challenge_tag = soup.select_one('div#risksAndChallenges')
            risk_challenge_list = risk_challenge_tag.select('p.js-risks-text.text-preline')
            for rc in risk_challenge_list:
                risk_challenge = rc.get_text().strip()
        except:
            risk_challenge=" "

        content_list_since_error = []
        content_list_since_error.append([idx, contents_collected, risk_challenge])
        pd.DataFrame(content_list_since_error, columns=['index','content','risk_challenge']).to_csv('failed_ks_content_crawled.csv', index=False, header=False, mod
e='a', encoding='UTF-8')
        print(idx)

    except Exception as ex:
        print("Unexpected error at {}".format(idx), ex)
        error_url = url
        count = 0
        while count < 11:
            print("Retrying:", count)
            driver.get(error_url)
            time.sleep(random.uniform(8,20))
            req = driver.page_source
            soup = BeautifulSoup(req, 'lxml')
            try:
                content_tag = soup.select_one('div.rte__content')
                contents = content_tag.select('p')
                contents_collected = []
                for c in contents:
                    content = c.get_text().strip()
                    contents_collected.append(content)

                try:
                    risk_challenge_tag = soup.select_one('div#risksAndChallenges')
                    risk_challenge_list = risk_challenge_tag.select('p.js-risks-text.text-preline')
                    for rc in risk_challenge_list:
                        risk_challenge = rc.get_text().strip()
                except:
                    risk_challenge=" "

                content_list_since_error = []
                content_list_since_error.append([idx, contents_collected, risk_challenge])
                pd.DataFrame(content_list_since_error, columns=['index','content','risk_challenge']).to_csv('ks_content_crawled.csv', index=False, header=False, mo
de='a', encoding='UTF-8')
                print(idx)
                break

            except Exception as et:
                print("Error while retrying:", et)
                time.sleep(random.uniform(100,130))
                count += 1
                continue

        #만약 차단이나 네트워크 오류로 크롤링이 안 되었을 시, 재시도를 10번 한 후 그래도 크롤링에 실패한다면 원래 했던대로 오류 index, url, nan값을 csv파일들에 저장
        if count == 11:
            error_list = []
            error_list.append([idx, error_url])
            pd.DataFrame(error_list, columns=['index','error_url']).to_csv('ks_middle_errors.csv', index=False, header=False, mode='a', encoding='UTF-8')

            content_list.append([idx, error_url, np.nan])

            error_into_saved_csv = []
            error_into_saved_csv.append([idx, error_url, np.nan])
            pd.DataFrame(error_into_saved_csv, columns=['index','content','risk_challenge']).to_csv('ks_content_crawled.csv', index=False, header=False, mode='a',
encoding='UTF-8')

        continue

driver.close()
```

In [ ]:

```
```

In [ ]:

```
# Google Colab에서 Selenium 사용
# 재시도 횟수 --> 4번으로 변경
```

In [ ]:

```
!apt update
!apt install chromium-chromedriver
!pip install selenium
```

In [ ]:

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

import os
import json
from datetime import datetime, timedelta

import time
import random
import requests
from bs4 import BeautifulSoup
from selenium import webdriver
from selenium.webdriver import Chrome
from selenium.webdriver import ChromeOptions
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC

options = webdriver.ChromeOptions()
options.add_argument('--headless')
options.add_argument('--no-sandbox')
options.add_argument('--disable-dev-shm-usage')
driver = Chrome('chromedriver', options=options)

#저장할 csv파일 이름/경로 지정 (Google Drive)
filename_contents = '/content/drive/My Drive/PlayData 빅데이터 분석 전문가 과정/Final_Project/Kickstarter_2019_12_12/content_crawled_2016.csv'
filename_errors = '/content/drive/My Drive/PlayData 빅데이터 분석 전문가 과정/Final_Project/Kickstarter_2019_12_12/errors_crawled_2016.csv'

for idx, url in enumerate(url_ks_2016[2165:], start=2165):
    driver.get(url)
    time.sleep(random.uniform(8,20))
    req = driver.page_source
    soup = BeautifulSoup(req, 'lxml')

    try:
        content_tag = soup.select_one('div.rte__content')
        contents = content_tag.select('p')
        contents_collected = []
        for c in contents:
            content = c.get_text().strip()
            contents_collected.append(content)

        try:
            risk_challenge_tag = soup.select_one('div#risksAndChallenges')
            risk_challenge_list = risk_challenge_tag.select('p.js-risks-text.text-preline')
            for rc in risk_challenge_list:
                risk_challenge = rc.get_text().strip()
        except:
            risk_challenge=" "

        content_list_since_error = []
        content_list_since_error.append([idx, contents_collected, risk_challenge])
        pd.DataFrame(content_list_since_error, columns=['index','content','risk_challenge']).to_csv(filename_contents, index=False, header=False, mode='a', encodin
g='UTF-8')
        print(idx)

    except Exception as ex:
        print("Unexpected error at {}".format(idx), ex)
        error_url = url
        count = 0
        while count < 5:
            print("Retrying:", count)
            driver.get(error_url)
            time.sleep(random.uniform(8,20))
            req = driver.page_source
            soup = BeautifulSoup(req, 'lxml')

            try:
                content_tag = soup.select_one('div.rte__content')
                contents = content_tag.select('p')
                contents_collected = []
                for c in contents:
                    content = c.get_text().strip()
                    contents_collected.append(content)

                try:
                    risk_challenge_tag = soup.select_one('div#risksAndChallenges')
                    risk_challenge_list = risk_challenge_tag.select('p.js-risks-text.text-preline')
                    for rc in risk_challenge_list:
                        risk_challenge = rc.get_text().strip()
                except:
                    risk_challenge=" "

                content_list_since_error = []
                content_list_since_error.append([idx, contents_collected, risk_challenge])
                pd.DataFrame(content_list_since_error, columns=['index','content','risk_challenge']).to_csv(filename_contents, index=False, header=False, mode='a',
encoding='UTF-8')
                print("Retry successful")
                print(idx)
                break

            except Exception as et:
                print("Error while retrying:", et)
                time.sleep(random.uniform(100,130))
                count += 1
                continue

        if count == 5:
            error_into_saved_csv = []
            error_into_saved_csv.append([idx, np.nan, np.nan])
            pd.DataFrame(error_into_saved_csv, columns=['index','content','risk_challenge']).to_csv(filename_contents, index=False, header=False, mode='a', encodin
g='UTF-8')

            error_list = []
            error_list.append([idx, error_url])
            pd.DataFrame(error_list, columns=['index','error_url']).to_csv(filename_errors, index=False, header=False, mode='a', encoding='UTF-8')
            print("Retry failed")

        continue

driver.close()
```