1. **Python 및 주요 패키지 설치**
2. **Python설치**

<https://www.anaconda.com/distribution/#download-section> (version – 3.6.5 이상)

아나콘다를 이용해서 설치하면 기본적으로 필요한 패키지가 같이 설치되어 환경설정에 용이합니다.

1. **주요패키지**

설치된 패키지 이외에 추가로 설치가 필요한 패키지나 주요패키지의 사용용도입니다.

(프로그램을 실행하다가 해당 패키지가 필요하다는 메시지가 뜨면 **window+R** 키를 둘러서 **cmd**실행을 해주고 **Pip install 패키지명** 을 해주면 설치가 됩니다.)

* **NLTK** (natural language toolkit) : 자연어처리 및 문서 분석용 패키지(문장단위 토큰화)
* **KoNLPy** : 한국어 정보처리를 위한 패키지(그 중 Twitter 형태소 분석기 사용)
  + 참고: <https://konlpy-ko.readthedocs.io/ko/v0.4.3/>
* **Pandas :** 파이썬에서 많이 사용되는 데이터 다루는 패키지(데이터를 엑셀파일처럼 다루기 용이)
* **Re :** 텍스트 데이터를 정규표현식으로 다루기 위한 패키지
* **Selenium, bs4 :** 동적으로 웹페이지 정보를 가져오기 위한 패키지
* **Pdfminer :** pdf파일을 텍스트 파일로 변환
* **Urllib :** url을 열고, 읽고, 에러를 다루는 패키지
* **Webbrowser :** 웹 기반 문서를 다룰 때 사용
* **Request :** http 요청을 보내는 모듈

1. **Jupyter notebook**

<https://github.com/jupyter/notebook/blob/master/docs-translations/ko-KR/Installation.md>

jupyter notebook으로 코드를 작성하면 라인단위로 실행해볼 수 있습니다.

1. **파일실행 방법**
2. 코드가 있는 폴더에서 **shift + 우클릭** -> 여기에 PowerShell 창 열기
3. **python 파이썬파일명** 입력
4. 파일에 따라 폴더나 파일경로를 입력해주고 실행
5. **파이썬 파일 설명**

**Naver Finance**

1. **crawler\_naver\_finance.py** : 네이버 경제에서 데이터를 크롤링한다
   * 입력
     + Chromedriver의 위치( \*.chrome.exe 까지 입력해야한다.)
     + 크롤링한 데이터를 저장할 위치(디폴트는 현재폴더)
     + 검색시작일자
     + 검색종료일자
     + 페이지 수(너무 데이터가 많을 시 체크해보기 위함)
   * 출력
     + Naver\_fiance\*.xlsx : 파일명이 같으면 덮어쓰기 때문에 중복되지 않는 숫자를 덧붙여서 파일 손상을 예방
2. **pdfdown.py** : 크롤링한 엑셀파일에 있는 pdf를 다운로드한다.
   * 입력
     + excel파일 명
     + pdf를 저장할 위치(폴더가 없으면 생성해준다)
   * 출력
     + 입력한 위치에 pdf파일을 다운
     + url이 잘못돼서 다운받지 못한 경우 다운받지못한pdf 텍스트파일에 목록 저장(이 txt파일은 항상 같은 이름이기 때문에 pdfdown.py를 다시 실행하면 덮어쓰기된다)
3. **pdfread.py** : pdf를 txt로 변환한다.

**( 제일 오래 걸리는 부분이기 때문에 중간에 종료했다가 다시 실행해도 이어서 실행되도록 작성함. 즉, 중간에 컴퓨터를 끄고 다시 실행해도 중복 실행되지 않아 문제없음 )**

* + 입력
    - pdf폴더 위치
    - txt를 저장할 폴더 위치
  + 출력
    - 입력한 위치에 txt변환한 파일 저장
    - 읽지못한pdf텍스트 파일에 읽지못한 목록저장(마찬가지로 파일명은 같기 때문에 다시 실행하면 덮어쓰기된다.)

1. **Combine\_txt.py** : 변환된 txt파일들을 합치고 1차 전처리한다.
   * 입력
     + txt폴더 위치
     + 데이터를 합칠 excel파일
   * 출력
     + Excel파일\_text.xlsx 파일이 생성
2. **Find\_analyst.py :** 룰베이스로 분석가를 찾는다. (이때는 텍스트를 전처리 하기전 데이터가 필요하기 때문에 txt폴더 위치를 다시 사용.)
   * 입력
     + txt폴더 위치
     + 분석가를 저장할 excel파일
   * 출력
     + 분석가가 추가된 excel파일
3. **Text\_pos\_neg.py** : 단어 수, 긍부정단어 수를 센다.
   * 입력
     + Excel 파일 명(text가 있는 파일)
     + Dart/naver 확인
     + 긍부정사전 위치
   * 출력
     + Excel파일\_text\_wordcnt.xlsx 파일 생성
4. Make\_dict.py : 긍부정txt를 명사추출해서 사전으로 만든다**.( 사전으로 한번 만들면 다시 실행할 필요 없으나, 불용어를 추가하고 싶으면 다시 실행하면 된다.)**
   * 입력
     + Pos\_pol\_word와 neg\_pol\_word가 있는 위치
   * 출력
     + Dictionary로 저장된 json파일
5. **crawler\_dart\_code.py** : 전자공시시스템(dart)에서 사업보고서를 크롤링한다.
   * 입력
     + API key(자신이 받은 api key를 넣는다. 현재 박수영의 api key가 입력되었으나 url 과다 요청으로 인해 닫힐 때가 종종 있다.)
     + 상장법인목록 엑셀 자료(현재 폴더에 넣기)
     + 크롤링한 데이터를 저장할 위치(디폴트는 현재 폴더)
     + 검색시작일자(start\_date)와 검색종료일자(end\_date)
   * 출력
     + Dart\_crp\_list.xlsx : 회사의 사업보고서 리스트. 회사당 발표한 사업보고서가 많으면 여러 행을 가진다.
     + Dart\_text\_list.xlsx : 데이터가 많아 오래 걸릴 시, naver와 마찬가지로 숫자를 넣어 따로 하기. 위 데이터와 다른 점은, 1) 이사의 경영진단 및 분석의견 부분의 html 소스 전체 텍스트가 포함되어 있다. 2) 이사의 경영진단 및 분석의견 부분이 없을 경우 사업보고서를 읽지 않았다. 따라서 약 200개 정도의 사업보고서를 읽지 않았다. (총12769개)
6. **Analyzer\_dart\_text.py** : dart 사업보고서의 텍스트의 단어수, 긍부정 단어수 세기

* 위의 text\_pos\_neg.py와 결합

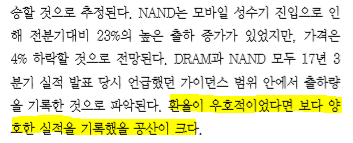
1. **한계점**

* **Naver: pdf파일을 읽는데 한계**

pdf파일을 텍스트로 변환하는 패키지를 사용해서 읽고 있습니다.

이때, pdf파일이 페이지마다 다른 형태로 저장되어 있는데 그림이나 표가 많이 들어있는 경우 간혹 읽을 수 없는 형태로 저장되어 있는 경우가 있습니다.

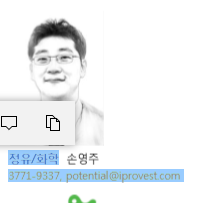
그럴 경우 그 페이지는 읽지 못하고 넘기게 되었습니다. 그리고 **그 목록을 txt파일로 저장했습니다. 또는 pdf전체를 읽지 못하는 경우에도 같이 목록을 저장했습니다.**



위의 사진과 같이 "환율이 우호적이었다면 보다 **양 호한** 실적을 기록했을 공산이 크다" 이렇게

양 호한 으로 텍스트가 읽혀지기 때문에 이러한 부분에서 오차가 발생합니다.

* **Naver: 분석가를 찾는데 한계**



위와 같이, 정유/화학 및 이메일 주소는 드래그해서 읽을 수 있는 텍스트이나 이름은 사진으로 파이썬 패키지가 글자로 인식할 수 없는 경우 분석가의 이름을 찾을 수 없습니다.



위와 같이 분석가가 2명인 경우도 존재합니다. 룰베이스로 규칙을 찾아서 두 명의 분석가를 냈습니다.

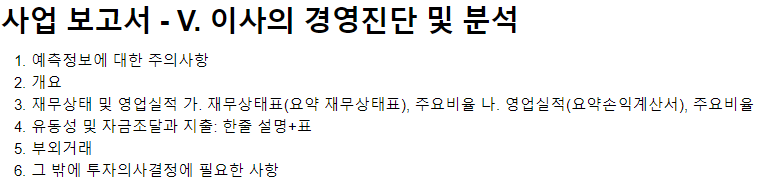
확실하지 않은 결과는 **표시했습니다.**

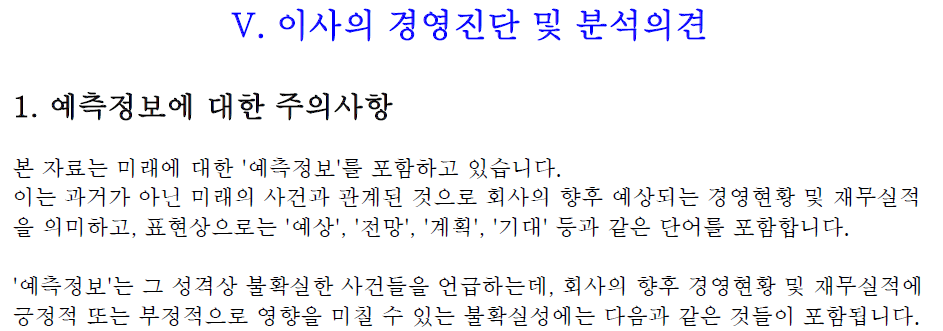
* **Dart: 데이터 읽기**

API key로 웹에 요청을 받아 데이터를 읽어오는 데에서 한번에 너무 많은 요청을 할 경우 해당 사이트에서 요청을 차단하는 경우가 자주 발생하였습니다. 짧게는 하루, 길게는 며칠 간 해당 사이트에 접속조차 되지 않았는데, 이 점 유의하시어 한 번에 확실한 코드를 돌리거나, 데이터를 짧게 끊어(저는 12989개의 데이터를 1000개 단위로 끊었습니다.) 데이터를 일일이 저장하고 나중에 다시 합치는 작업을 하였습니다. url 요청과 관련하여 오래 걸린 것 외에 데이터를 읽는 데에 발생한 문제는 대부분 해결되었습니다.

* **Dart: 텍스트 고유의 특징**

사업보고서 특성상, 형식적인 어구와 틀이 존재하여, 이를 유의하여 단어수의 형태를 파악한다.





* **트위터 형태소 분석기를 사용한 명사 추출의 한계**

예를 들어) ‘한샘’ 을 명사 추출하면 ‘샘’이 되어버린다.

* **긍부정을 명사만 비교하는 한계**

‘대우전자’ 의 대우도 긍정어로 체크된다. (대우라는 단어가 문서에서 잘 쓰이지 않는다면 불용어로 처리하는 방법이 존재한다)

* **텍스트 단어 수를 세기 위한 전처리의 한계**

1. 한글만 고려하여 단어 수를 센다.
2. ‘R&D’와 같은 단어들을 최대한 고려한다.

두 방법 모두 오차는 발생한다.