# 텍스트 마이닝의 소개와 파이썬 작업 환경의 구축

이기황

2017.04.13

## 1 강좌 안내

"텍스트 마이닝 캠프" 강좌의 성격과 목표, 강의 내용, 그리고 자료 제공 방법 등에 대하여 소개한다.

### 1.1 강좌의 성격과 목표

이 강좌는 텍스트 마이닝에 관한 "입문" 또는 "개론"의 성격을 지닌다. 다만 다양한 텍스트 마이닝 기법에 대한 이론적 탐구, 혹은 수리적 이해보다는 텍스트 처리 현장에서의 실질적인 활용을 위한 실마리를 제공하는 것을 목표로 한다. 이에 따라 이 강좌는 다음과 같이 진행한다.

- **파이썬 언어를 이용한 텍스트 마이닝 실습을 진행한다.** 매회 즉시 실행하여 결과를 확인할 수 있는 파이썬 예제 코드를 제공하며 주요 라이브러리 모듈의 사용법을 실제 데이터에 적용하며 학습한다. 강좌의 목표가 파이썬 코딩 기술의 습득에 있는 것은 아니지만, 효율적인 파이썬 코딩 기법도 틈나는 대로 소개한다.
- **실제 업무에서 흔히 만날 수 있는 텍스트 데이터를 실습에 이용한다.** 신문 기사, 논문 초록, 연설문 등을 예제 데이터로 다룬다. 참여자들은 개인 프로젝트를 진행하며 "현업" 데이터를 활용할 수 있으며 가능한 경우 사용자 제공 데이터를 강의 시간에 공유하며 다루어 볼 수도 있다.
- **주피터 노트북을 이용한 자료 처리를 경험한다.** 텍스트 자료 처리는 대규모의 일괄 작업으로 이루어지기도 하지만, 초기 단계에서는 데이터에 대한 탐색적 접근이 필요할 때가 많다. 이러한 상황에서는 코드와 데이터, 시각화, 그리고 문서화까지 통합적으로 접근할 수 있는 주피터 노트북이 매우 편리하다.
- **텍스트 마이닝의 실제 업무 적용에 관한 경험과 고민을 공유한다.** 텍스트 마이닝은 다양한 학술과 산업 현장에서 널리 활용되고 있다. 텍스트 마이닝을 어떻게 학문 연구와 비지니스에 이용할 수 있는지에 대하여 강사와 조교, 그리고 참여자들이 함께 고민하며 경험을 나눈다.

10주는 위의 목표를 달성하기에 빠듯한 시간이다. 강의 시간 중에 실습을 진행하기는 하지만 충분하지는 않을 것이다. 그러므로 효과적인 학습을 위해서는 개인별 반복 강화 학습이 필수적으로 요청된다.

### 1.2 주별 강의 계획서

이 강좌는 다음과 같은 일정과 내용으로 진행할 예정이다. 세부적인 내용은 상황에 맞게 바뀔 수 있다. 개인 혹은 팀 프로젝트는 필수는 아니지만 실력 향상을 위해 권장된다.

- **제1강**: 텍스트 마이닝의 소개 및 파이썬 작업 환경 구축
  - 텍스트 마이닝의 개념과 응용 사례
  - 파이썬 언어 소개
  - 아나콘다 파이썬과 보조 도구의 설치
  - 주피터 노트북을 이용한 파이썬 실습
- **제2강**: 파이썬 텍스트 처리 (1)
  - 숫자와 문자, 그리고 문자열
  - 리스트, 딕셔너리, 세트
  - 파이썬의 단문과 복문
  - 내장 함수와 표준 라이브러리
- **제3강**: 파이썬 텍스트 처리 (2): 웹 크롤링
  - 웹 크롤링의 원리와 절차
  - 정적 컨텐츠의 수집
  - 동적 컨텐츠의 수집
  - 트윗의 수집
- **제4강**: 형태소 분석과 후처리
  - 형태소와 형태소 분석의 개념
  - 형태소 분석 라이브러리의 설치와 사용
  - 형태소 분석 결과의 구조화와 저장
  - 형태소 분석 결과의 후처리
- **제5강**: 키워드 분석
  - 형태소 빈도 계수
  - 형태소 빈도의 시각화
  - 용어 빈도와 문헌 빈도
  - 분류 사전의 활용
- **제6강**: 어휘 공기 분석
  - 어휘 공기의 개념
  - 어휘 공기의 추출과 계수
  - 어휘 공기 행렬의 구성
  - 어휘 공기 네트워크의 생성
- **제7강**: 문서 군집
  - 문서 군집의 개념
  - 문서의 속성과 유사도 측정
  - 계층적 군집 분석

  - 비계층적 군집 분석

- **제8강**: 문서 분류
  - 문서 분류의 개념
  - 나이브 베이즈 모델을 이용한 문서 분류
  - 문서 분류의 성능 평가
  - 그리드 탐색에 의한 문서 분류 파라미터 최적화

- **제9강**: 토픽 모델링과 워드 임베딩
  - 토픽 모델링의 개념
  - 토픽 모델링을 이용한 주제 분석
  - 워드 임베딩의 개념
  - 워드 임베딩을 이용한 유사도 분석

- **제10강**: 감성 분석
  - 감성 분석의 개념과 방법
  - 문서 분류 기법을 이용한 긍부정 분석
  - 감성어 사전 기반 세부 감성 분석
  - 개인별 텍스트 마이닝 프로젝트 경험 공유

### 1.3 강의 자료 제공과 페이스북 그룹 운영

PDF 형식의 강의안과 주피터 노트북, 그리고 예제 데이터는 압축 파일의 형태로 웹 링크를 통하여 제공한다. 강의 시간에서 다루지 못한 내용의 보충 설명이나 질의, 응답은 페이스북 그룹 페이지를 이용하여 이루어진다. 강의에서 직접 다루지 않은 내용이라 하더라도 텍스트 마이닝에 관련된 내용이라면 어떤 내용이든지 질문과 공유가 가능하다.

## 2 텍스트 마이닝 개관

### 2.1 개요

텍스트 마이닝은 비정형 텍스트로부터 **유의미한** 정보를 얻는 과정을 말하는데, **텍스트**와 **데이터 마이닝**의 결합이라고 볼 수 있다. 비지니스 인텔리전스의 전통에서는 텍스트 애널리틱스라고 부르기도 한다. "유의미하다" 혹은 "유가치하다"는 개념은 간단히 정의하기 어렵다. 위키피디어에서는 이와 관련하여 연관성(relavance), 참신성(novelty), 그리고 흥미성(interestingness)을 텍스트 마이닝을 통해 얻을 수 있는 고품질 정보의 속성으로 제시한다.

텍스트 마이닝의 핵심은 자연언어처리(natural language processing) 기술과 다양한 분석(analytical) 기법을 이용하여 텍스트를 분석(analysis) 가능한 **데이터**로 변환하는 것이라고 할 수 있다. 텍스트 마이닝의 절차는 대략 다음과 같이 요약된다.

- 입력 텍스트의 구조화

- 구조화된 텍스트로부터 패턴의 도출
- 결과의 해석

### 2.2 세부 과제

텍스트로부터 유의미한 정보를 추출하기 위한 텍스트 마이닝의 세부 과제에는 다음과 같은 것들이 있다. 이들은 독립적으로 활용되기보다는 결합하여 사용되어 인사이트 도출에 기여한다.

- 텍스트 분류(text categorization): 주어진 학습 자료로부터 기계 학습을 통해 분류 모델의 구축하여 새로운 자료를 분류, 혹은 꼬리붙이기(labeling)를 하는 일로 지도 학습(supervized learning)이라고도 부름.
- 텍스트 군집(text clustering): 학습 자료 없이 주어진 텍스트에서 추출 가능한 자질(feature) 혹은 속성(attribute)을 이용하여 비슷한 텍스트들끼리 묶는 일로 비지도 학습(unsupervized learning)이라고도 부름.
- 개념/개체 추출(concept/entity extraction): 텍스트의 내용 혹은 의미를 대표, 요약할 수 있는 개념 혹은 개체명(인명, 장소명, 조직명 등)을 인식하고 추출하는 일.
- 분류 사전 생성(taxonomy production): 텍스트로부터 유의어 사전, 시소러스(thesaurus) 등을 유도하는 일.
- 감성 분석(sentiment analysis): 텍스트에 표현된 긍부정 감성을 인식하는 일로 오피니언 마이닝이라고 부름.
- 텍스트 요약(text summarization): 긴 텍스트를 짧은 텍스트로 요약하는 일.
- 개체 관계 모델링(entity relation modeling): 텍스트에서 추출한 개체관의 관계(예: 조직-소속원)를 인식하는 일.

### 2.3 관련 기술

앞서 언급한 바와 같이 텍스트 마이닝은 자연언어처리 기술을 기반 기술로 활용하며 다음과 같은 자료 처리 기술을 광범위하게 활용한다.

- 정보 검색
- 키워드 분석
- 패턴 분석
- 태깅/주석
- 정보 추출
- 데이터 마이닝(링크/연관 분석, 시각화, 예측 분석)

### 2.4 응용 분야

텍스트 마이닝은 여러 산업과 학문 분야에서 활용되고 있다.

- 보안
- 생의학

- 소셜 미디어
- 마케팅
- 디지털인문학과 전산사회학 등의 학문

## 2.5 활용 사례

### 2.5.1 학술 연구

- 박성희. 2009. 제17대 대통령 후보 합동 토론 언어네트워크 분석 : 북한 관련 이슈를 중심으로. 『한국언론정보학보』45: 220-254. http://academic.naver.com/view.nhn?doc_id=57814796&dir_id=0&field=0&unFold=false&gk_adt=0&sort=0&qvt=1&query=%EC%A0%9C17%EB%8C%80%20%EB%8C%80%ED%86%B5%EB%A0%B9%20%ED%9B%84%EB%B3%B4%20%ED%95%A9%EB%8F%99%20%EC%97%B0%EC%84%A4&gk_qvt=0&citedSearch=false&page.page=1&ndsCategoryId=10319&library=111
- 배정환, 손지은, 송민. 2013. 텍스트 마이닝을 이용한 2012년 한국대선 관련 트위터 분석.『지능정보연구』19(3): 141-156. http://jiisonline.evehost.co.kr/files/DLA/20131108225932_9-19-3.pdf

### 2.5.2 마케팅

- http://analyticsstory.com/60
- 유유제약
- 아이들을 위한 진통소염제로 분류되어 판매
- 빅데이터 분석을 통한 기회 발굴
- 50% 이상 매출 상승

### 2.5.3 방송 컨텐츠

- 썰전: 국민 여동생
- http://news.jtbc.joins.com/article/article.aspx?news_id=NB10385733
- https://www.youtube.com/watch?v=rCQyxEn6TRQ

### 2.5.4 정보 서비스

- 언론진흥재단 BIGKinds
- http://www.kinds.or.kr

## 2.6 텍스트 파일의 유형과 특징


〈그림 1〉 베노플러스


〈그림 2〉 국민 여동생


〈그림 3〉 언론진흥재단 BIGKinds 서비스

### 2.6.1 텍스트 파일과 바이너리 파일

텍스트 마이닝의 대상인 텍스트 데이터는 보통 파일의 형태로 저장되어 있다. 텍스트 데이터만 포함하고 있는 파일은 당연하게도 **텍스트 파일**이라 부르며 텍스트가 아닌 데이터도 포함하고 있는 파일은 **바이너리 파일**이라 부른다.

그런데 이 구분은 파일이 포함하고 있는 데이터의 "내용"에 의한 구분이 아니라 "형식"에 의한 구분이다. 예를 들어 대학생의 기말 보고서가 담긴 워드프로세서 데이터 파일은 내용적으로는 텍스트 데이터를 포함하고 있지만 형식적으로 텍스트 파일이 아니다. 스프레드시트나 프레젠테이션 프로그램의 데이터 파일도 마찬가지이다. 그래서 여기서 말하는 텍스트 파일을 좀 더 명확히 부르는 명칭은 **플레인 텍스트 파일**이다.

바이너리 파일과 대별되는 플레인 텍스트 파일의 형식적 특성은 다음과 같다.

- 텍스트 파일은 라인의 연속체(sequence)로 구성된다.
- 각 라인은 문자로 구성된다.
- 문자는 인쇄 가능(printable) 문자와 인쇄 불가능(non-printable) 문자로 구분되며, 인쇄 불가능 문자에는 터미널 등에서 특수한 기능을 수행하는 제어(control) 문자가 포함된다.
- 라인과 라인은 새줄 혹은 개행(newline), 또는 줄끝(EOL) 문자로 구분된다. 새줄 문자로는 캐리지 리턴(CR) 문자나 라인 피드(LF) 문자가 사용된다.
- 경우에 따라 파일 끝을 표시하는 파일끝(EOF) 문자가 사용되기도 한다.

위와 같은 특성을 지닌 텍스트 파일은 특정 플랫폼이나 프로그램에 종속되지 않는다. 그러므로 많은 프로그램들이 텍스트 파일을 불러오는 기능, 그리고 해당 프로그램의 파일 형식을 텍스트 파일로 내보내는

기능을 가지고 있다.

많은 운영체제, 특히 윈도우는 파일의 형식을 파일 명칭 확장자(extension)로 나타내는데 일반적으로 텍스트 파일의 확장자로는 `.txt`가 쓰인다. 그러나 이는 엄격한 규칙이 아니다. 윈도우에서는 메모장 프로그램으로 내용을 볼 수 있는 파일을 텍스트 파일이라고 보면 된다.

### 2.6.2 텍스트 파일의 유형

텍스트 파일은 그 구조적 형태와 속성에 따라 몇 가지 유형으로 나눌 수 있다.

- 정형 텍스트 파일
- 비정형 텍스트 파일
- 반정형 텍스트 파일

정형(structured) 텍스트 파일은 엑셀로 대표되는 스프레드시트에서 흔히 볼 수 있는 가로와 세로로 짜여진 표 형식의 텍스트 파일을 가리킨다. 이러한 유형의 텍스트 파일은 항목 구분자로 쉼표 혹은 탭 문자를 사용하는 것이 일반적이다. 특히 쉼표로 항목이 구분된 파일을 CSV(Comma-Separated Values)라고 부르며 스프레드시트, 혹은 데이터베이스에 저장된 데이터를 텍스트 파일로 변환할 때에 널리 사용된다. 그런데 현장에서는 구분자로 사용되는 쉼표가 텍스트의 일부에도 사용될 수 있기 때문에 쉼표 대신에 탭 문자를 구분자로 사용하는 경우가 많다. 탭 문자가 구분자로 사용된 파일을 TSV(Tab-Separated Values) 파일이라고 특정하여 부르기도 한다.

앞서 언급한 바와 같이 정형 텍스트 파일은 스프레드시트, 혹은 구조적 데이터의 저장과 처리에 널리 사용되는 관계형 데이터베이스의 자료를 텍스트 파일로 변환할 때에 널리 사용되기에 데이터베이스의 개념을 일부 적용할 수 있다. 쉼표 혹은 탭 문자로 구분되어 가로로 나열되는 항목들은 열, 컬럼, 혹은 필드라고 부른다. 그리고 정해진 수의 열로 구성된 하나의 단위를 행, 로우, 또는 레코드라고 부른다. 하나의 레코드는 하나의 라인으로 나타내는 것이 일반적이다. 그러므로 라인 수를 세는 것은 해당 파일에 포함된 레코드의 수를 세는 것이 된다.

비정형(unstructured) 텍스트는 신문 기사, 소설, 전자우편, 소셜 미디어 포스팅과 같이 라인 구분 외의 별다른 구조적 속성이 보이지 않는 텍스트 파일 유형이다. 물론 기사 제목, 날짜, 기자명 등이 구분되기도 하고 단락의 구분이 있기도 하지만 비정형 텍스트 파일에서는 이들의 구분이 명시적으로 이루어지지 않는다. 즉, 구조적 항목의 명시적 구분자(delimiter)가 존재하지 않거나 중의적이다. 이러한 텍스트를 자유 형식(free-form) 텍스트라 부르기도 한다.

반정형(semi-structure) 텍스트 파일은 비정형 텍스트 파일에 다양한 서지정보 등의 메타정보, 인쇄와 화면 렌더링을 위한 조판 정보, 텍스트 구조적 정보를 표시하기 위해 구분자를 도입하여 텍스트에 주석, 혹은 표식을 부가한 유형의 텍스트 파일이다. 텍스트 주석 방식으로는 SGML(Standard Generalized Markup Language)에 기반을 둔 HTML(Hyper Text Markup Language, XML(eXtensible Markup Language) 등이 널리 쓰인다.

최근에는 텍스트 마크업보다 복잡한 데이터를 텍스트 형식으로 표현하는 데에 집중하는 YAML(YAML Ain't Markup Language), JSON(JavaScript Object Notation) 등의 형식이 주목을 끌고 있으며, 렌더링을 위한 간략한 마크업 언어, 이른바 "경량 마크업 언어"(light-weight markup language)로 분류되는 마크다운(Markdown), 텍스타일(Textyle), 리스트럭쳐드(reStructured) 텍스트 등도 인기를 얻고 있다.

### 2.6.3 문자 부호화 체계

컴퓨터는 숫자를 다루는 기계로 알려져 있다. 따라서 컴퓨터에서 문자를 다루려면 문자를 숫자로 변환하고, 반대로 숫자를 문자로 바꾸는 과정이 필요하다. 이 과정을 부호화(encoding)와 복호화(decoding)이라 부른다. 보통은 이 과정이 컴퓨터의 운영체제에서 자동적으로 이루어지기 때문에 사용자들은 이를 인식하지 못하며 자세한 내용은 이 강좌의 범위를 넘어선다. 다만, 텍스트 자료를 다룰 때에 흔히 언급되는 몇 가지 부호화 방식은 알아 둘 필요가 있을 것이다.

- 유니코드: ISO와 유니코드 컨소시움(http://www.unicode.org)에서 유지, 관리하는 국제 표준 문자 부호화 체계로 전 세계에서 사용했거나 사용하는 모든 문자를 표현하는 것을 목표로 한다. 이에 해당하는 대한민국의 표준은 KS C 5700이었다가 KS X 1005-1로 개정되었다. 11,172자의 한글 음절 영역과 현대어와 중세어 한글 모두를 표현할 수 있는 한글 자모 영역이 포함되어 있다.
- ASCII: 미국에서 산업 표준으로 정해진 라틴 문자 부호화 방식이다. 이를 바탕으로 국제 표준인 ISO 8859가 만들어졌으며 유럽어 문자 표현을 위한 여러 확장이 이루어졌다.
- UTF-8: 유니코드를 실제로 컴퓨터에서 사용할 때에 기존 소프트웨어 환경과의 호환성을 최대한 유지할 수 있도록 만들어진 문자 부호화 방식이다. 유니코드 텍스트 파일은 사실 UTF-8 부호화가 적용된 파일인 경우가 대부분이다.
- EUC-KR: 유니코드 이전에 한글을 유닉스 운영체제 상에서 표현하기 위해 벨 연구소에서 제안한 부호화 체계. EUC-KR은 한글의 부호화에 대한 대한민국 표준인 KS C 5601과 로마자 부호화의 표준인 KS C 5636의 합이다. 한글 음절 가운데 2,350자만 표현 가능하다. KS C 5601은 이후 KS X 1001로 개정되었다.
- CP949: 마이크로소프트에서 사용하는 비표준 부호화 체계로 EUC-KR을 확장하여 한글 음절 8,822자를 추가하여 이론적으로 생성 가능한 현대 한글 음절 11,172자를 모두 표현한다.

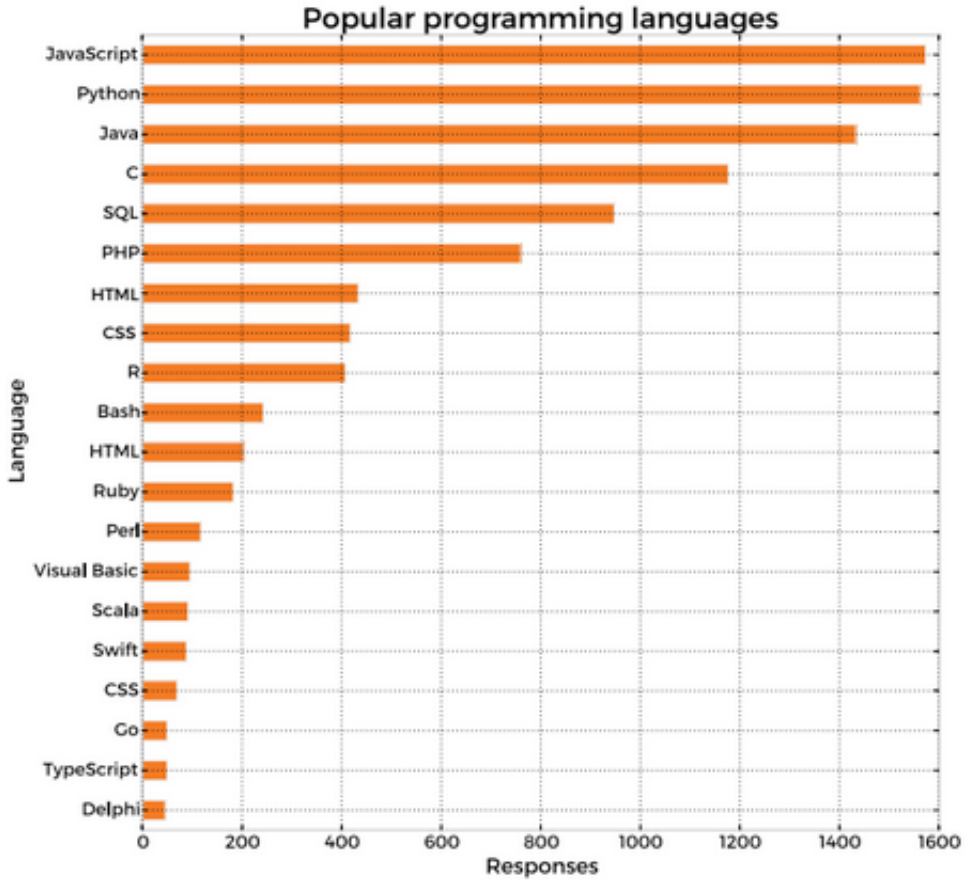
문자 부호화의 사실 상의 표준은 UTF-8이다. 우리나라의 경우 상당히 많은 텍스트가 EUC-KR 혹은 CP949 부호화가 적용된 상태로 존재하는데 점차 UTF-8로 이동하고 있다.

## 2.7 텍스트 마이닝의 핵심

텍스트 마이닝의 핵심은 비정형 혹은 반정형의 텍스트 또는 텍스트 집합으로부터 정보 가치가 있는 항목들을 추출하여 정형 텍스트, 즉 표로 구성하는 것이라고 할 수 있다. 이 과정은 대체로 정보 가치가 있는 것으로 판단되는 항목, 혹은 단위의 인식, 그리고 특정 범위 내에서의 이 단위의 출현 빈도 계수로 구성된다.

한편 위와 같이 구성된 표가 최종 결과물일 때도 있으나 이 표를 입력으로 하는 추가 분석이 이루어질 수도 있다. 이 때 서로 다른 항목, 또는 단위의 비교, 유사성의 측정 등이 이루어질 수 있다.

> 예를 들어 다음 기사의 작성을 위해 진행된 계량적 텍스트 분석의 일부를 컴퓨터를 이용하여 진행할 수 있을 것이다. http://www.ccdm.or.kr/xe/vote/214859




〈그림 4〉 파이썬의 인기 (Skill Up Repor 2016, Packt Publishing)

## 2.8 텍스트 마이닝을 위한 프로그래밍 언어의 선택

이 강좌에서는 파이썬 (Python) 언어를 텍스트 마이닝을 위한 도구로 사용한다. 컴퓨터 언어는 도구일 뿐이다. 목표 달성을 위해서는 어느 도구든지 편리하고 익숙한 것을 쓰면 된다. 파이썬은 최근 여러 분야에서 큰 인기를 얻고 있는데, 특히 데이터 과학 분야에서 가장 널리 사용되고 있는 것으로 알려져 있다.

널리 사용되는 프로그래밍 언어를 사용하는 것은 유행에 동참하는 것 이상의 의미가 있다. 파이썬은 널리 사용되는 만큼 관련 서적을 비롯한 수많은 자료가 넘쳐난다. 텍스트 마이닝 관련 서적도 상당수 출간되었는데 아쉽게도 외국 서적이 더 많고, 한국어의 언어적 특성 때문에 외국의 사례를 직접 적용하기 어려운 부분도 있다.

### 2.8.1 파이썬의 특징

파이썬은 간결하고 명확한 문법과 다양한 사용자 라이브러리의 지원으로 큰 주목을 받고 있는 프로그래밍 언어이다. 간결하고 명확한 문법을 지녔다는 것은 파이썬이 다른 언어에 비해 배우기 쉽다는 의미이다. 다소 과장되었다고 할 수 있겠으나 파이썬을 가리켜 "실행 가능한 의사 코드 (executable pseudo-code)"라 하기도 한다. 비교적 적은 수의 키워드를 지닌 동적 언어인 파이썬 코드가 그만큼 읽고 쓰기 쉽다는 뜻이다.

> "If Python is executable pseudo-code, then Perl is executable line noise." - ***Randall Munroe***

오해하지 말아야 할 것은 파이썬이 배우기 쉽다고 해서 심각한 일에는 쓸 수 없는 장난감 언어가 아니라는 점이다. 파이썬은 다양한 산업 현장에서 널리 사용되고 있으며, 특히 자료 처리와 분석을 위한 언어로 각광을 받고 있다.

파이썬을 지칭하는 또 다른 표현으로 "건전지 포함(batteries included)"이라는 말이 있다. 이는 지금도 유지되고 있는 파이썬의 철학이기도 한데 가능한 한 사용성이 좋은 라이브러리들을 파이썬 표준 라이브러리에 포함하여 배포하려는 노력을 뜻한다. 그러나 파이썬의 활용 분야가 급격히 확대되며 새로운 라이브러리가 계속 등장하고 있는 요즈음 건전지 포함을 외치는 것은 현실적으로 불가능한 일이다. 사용자 입장에서는 자신이 파이썬으로 하려는 일, 혹은 그와 비슷한 일을 누군가 먼저 했을 것이며, 그 목표를 달성하기 위한 도구를 만들었고, 그 도구를 공유했을 가능성이 크다는 것을 염두에 두면 된다. 주의해야 할 것은 다양한 사용자 라이브러리가 존재한다는 것이 파이썬의 고유 속성이 아니라는 것이다. 즉, 파이썬 언어의 능력이 필요한 기능의 사용자 라이브러리의 존재 여부로 판단되어야 하는 것은 아니다.

파이썬이 여러 가지 장점을 지닌 것을 틀림없지만 당연히 단점도 가지고 있다. 가장 널리 알려진 단점은 파이썬으로 작성된 프로그램의 실행 속도가 느리다는 것이다. 이는 분명한 사실이다. 그러나 컴퓨터로 해결해야 하는 모든 과제가 어셈블리나 C 언어로 작성된 프로그램처럼 빠른 실행이 보장되어야 하는 것은 아니다. 경우에 따라서는 프로그램의 실행 속도보다는 작성 속도, 유지 관리의 용이성이 더 중요하다.

> 최근 https://hackernoon.com/yes-python-is-slow-and-i-dont-care-13763980b5a1에 파이썬의 성능(실행 속도)과 생산성에 관한 흥미로운 블로그 포스트가 실렸다.

더 기술적인 사안이기는 하지만 또 하나의 잘 알려진 파이썬의 약점은 소위 GIL(global interpreter lock)이라 불리는 것으로 특정한 상황에서 멀티 스레딩 프로그램이 멀티 코어 시피유를 제대로 활용하지 못하여 병목이 발생하는 현상이다. 이에 관한 논의는 이 강좌의 범위를 벗어나므로 여기서 그친다. 분명한 것은 GIL에 대해서 너무 큰 걱정은 하지 않아도 된다는 것이다.

마지막으로 단점이라기보다는 불평에 가까운 것인데 파이썬이 들여쓰기에 민감하다는 것이다. 파이썬은 들여쓰기의 크기와 방법의 일관성을 강제한다. 이 때문에 자유로운 코딩 스타일에 익숙한 사용자는 일시적으로 불편을 겪을 수도 있다. 그러나 이는 읽고 쓰기가 쉬운 코딩 스타일을 유지하기 위한 방편일 뿐이다.

### 2.8.2 2인가 3인가?

현재 사용되고 있는 파이썬에는 두 가지 판(버전)이 있다. 파이썬 2.X와 파이썬 3.X가 그것이다. 보통 한 언어의 새 판이 나오면 자연스럽게 새 것으로 이동이 이루어지기 마련인데 파이썬의 경우에는 그렇지 않다. 그 이유는 파이썬 3.X가 파이썬 2.X와의 완전한 하위 호환성을 지원하지 않기 때문이다.

> 여기서 말하는 '파이썬'은 가장 널리 사용되는 C 언어로 구현된 씨파이썬(CPython)을 가리킨다. 씨파이썬 이외에 자바로 구현된 자이썬(Jython), C#으로 구현된 아이언파이썬(IronPython)도 있다.

파이썬 3.X에서 도입된 새로운 문법의 대부분은 2.7.X에 역이식되었기 때문에 단순한 문법의 변화는 큰 문제가 아니다. 문제가 되는 것은 모든 문자열을 유니코드 문자열로 다루는 것, 기존의 문자열은 바이트로 다루는 것으로 우리처럼 소위 알파누머릭이 아닌 문자를 사용하는 문화권의 자료 처리에서는 매우 중요한 사안이다. 파이썬 2.X에서 한글 문자열의 인코딩/디코딩에서 발생하는 문제로 골머리를 썩은 경험은 겪어보지 않은 사람은 상상도 하지 못할 것이다.

> http://www.snarky.ca/why-python-3-exists에 파이썬 3가 어떻게 해서 나오게 되었는지에 대한 간략한 소회가 있다.

짧게 말하면 파이썬 프로그래밍을 새로 시작하는 사람이면 무조건 파이썬 3.X로 시작하라는 것, 그리고 아직 파이썬 2.X를 쓰는 사람은 역이식된 파이썬 3.X의 문법에 익숙해지면서 빠른 시일 안에 3.X로 이전하는 것이 좋다.

물론 중요한 모듈이 파이썬 3.X를 지원하지 않는 등의 이유로 파이썬 2.X를 반드시 사용해야 하는 경우도 있다. 이런 경우에는 파이썬 2.X와 3.X를 동시에 설치하여 사용해야 하는 경우가 있을 수 있다. 이 경우에도 파이썬 2.X 코드를 최대한 파이썬 3.X 코드에 가깝게 작성하는 것이 좋다.

더 진보한 방법은 Virtualenv(https://virtualenv.pypa.io/en/latest)와 같은 도구를 활용하여 디렉토리별로 분리된 파이썬 환경을 유지하는 것이다. 이 강좌에서 권장하는 아나콘다 파이썬의 경우는 conda라는 패키지 관리 도구를 이용하여 이와 같은 일을 쉽게 할 수 있다.

#### 2.8.3 배포판의 선택

파이썬 배포판이란 파이썬을 이용한 소프트웨어의 개발에 필요한 파이썬 인터프리터, 표준 라이브러리 모듈, 디버거, 테스팅 도구, 통합 개발 환경, 사용자 설명서 등을 한 번에 설치할 수 있도록 설치 프로그램으로 묶어서 배포하는 소프트웨어 패키지를 말한다.

파이썬 인터프리터와 표준 라이브러리, 그리고 통합개발환경인 아이들(IDLE)을 포함한 표준 파이썬 배포판은 파이썬 공식 사이트(http://www.python.org)에서 배포된다. 리눅스와 오엑스텐 운영체제에는 기본적으로 파이썬이 포함되어 있기도 하다. 그러므로 파이썬 배포판의 선택은 중요한 문제가 아닐 수도 있다. 그러나 앞서 언급한 바와 같이 다양한 사용자 라이브러리를 원활하게 사용하기 위해서는 조금 고민이 필요하다. 자주는 아니지만 가끔 여러 개의 라이브러리들이 의존성 문제로 복잡하게 얽히기도 하고 라이브러리가 파이썬이 아닌 C/C++ 언어로 작성되었는데 컴파일된 바이너리가 제공되지 않고 소스만 제공되면 특히 C 컴파일러의 설치가 까다로운 윈도우 운영체제에서는 큰 곤란을 겪을 수도 있다. 이러한 이유로 이 강좌에서는 널리 사용되는 라이브러리 모듈과 통합개별환경 등을 패키지로 제공하는 아나콘다 파이썬 배포판의 사용을 권장한다.

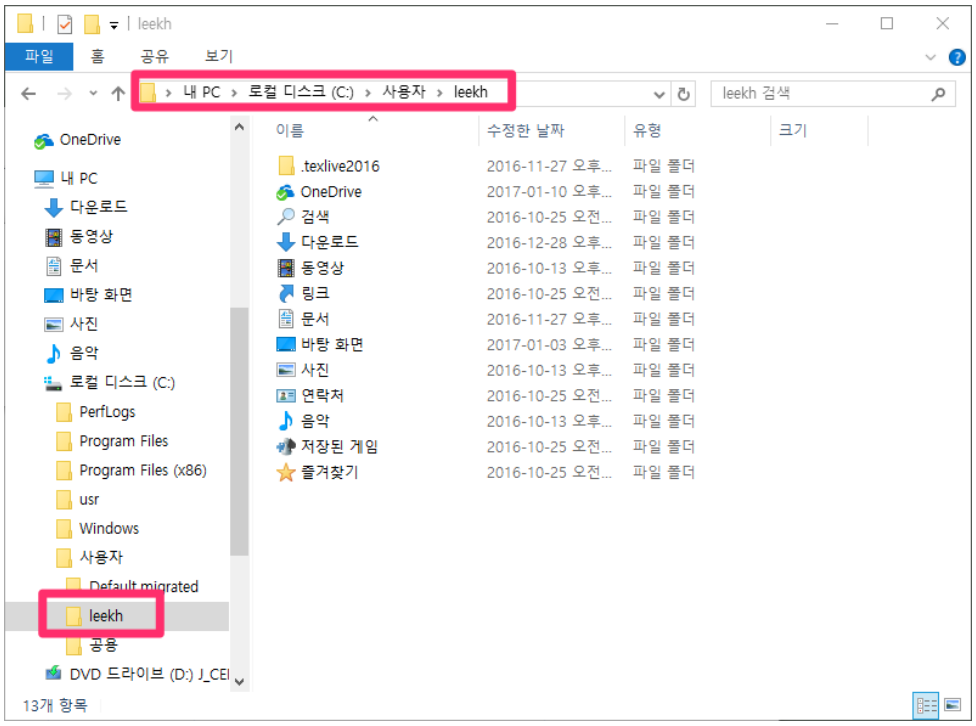
# 3 작업 환경 구축

## 3.1 아나콘다 파이썬의 설치

### 3.1.1 준비

이 글에서는 마이크로소프트 윈도우 운영체제에 아나콘다 파이썬 배포판을 설치하는 과정을 안내한다. 설명의 기준이 되는 윈도우의 버전은 윈도우 10이지만 윈도우 7 이상이라면 동작에 큰 차이가 없다. 특별히 유의해야 할 점은 **사용자의 홈디렉토리명에 공백이나 한글이 포함되지 않아야 한다**는 것이다. 즉, 다음 그림과 같이 탐색기를 실행하였을 때에 표시되는 홈디렉토리명이 한글과 공백이 없어야 한다.




〈그림 5〉 사용자 홈디렉토리 확인

### 3.1.2 설치 프로그램 내려받기

아나콘다 파이썬 배포판의 내려받기 페이지(https://www.continuum.io/downloads)에서 볼 수 있는 것처럼 현재 아나콘다 파이썬 배포판의 최신판은 4.3.1이다. 그러나 이 강좌에서는 4.2.0을 사용하므로 그림과 같이 이전판 저장소 페이지(https://repo.continuum.io/archive/index.html)에서 설치 프로그램 파일 `Anaconda3-4.2.0-Windows-x86_64.exe`를 내려받아야 한다.

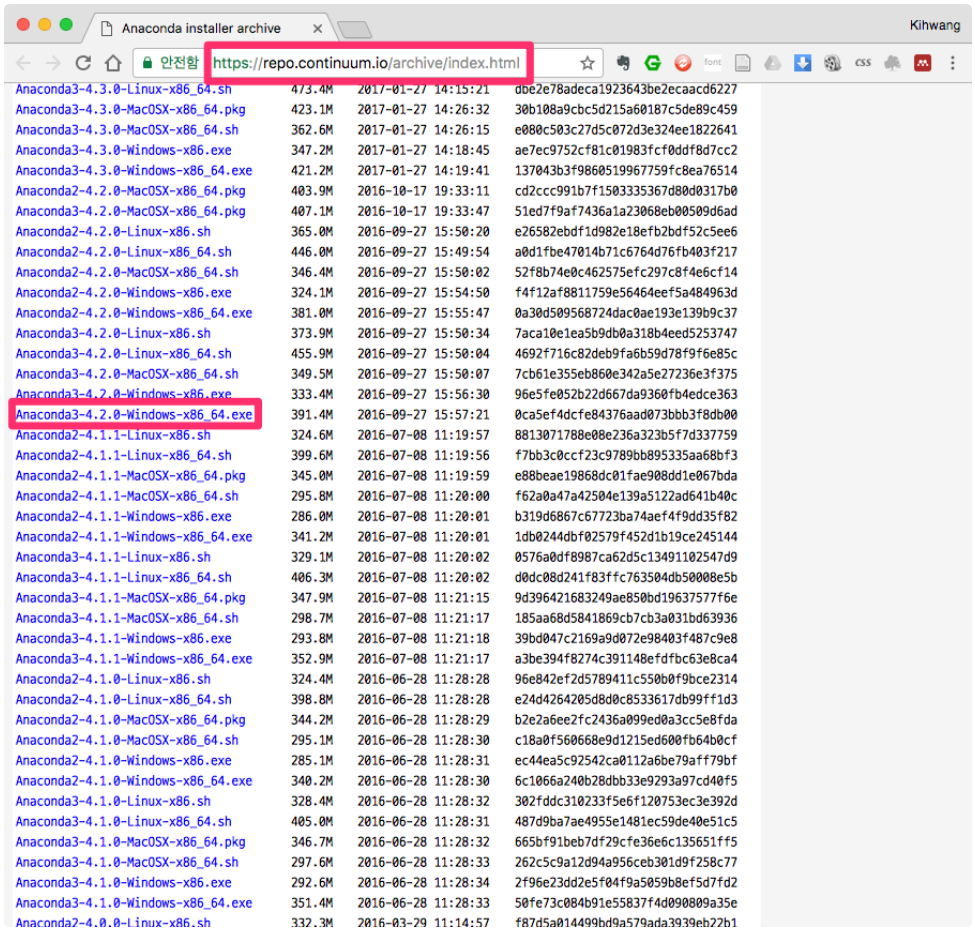
> 아나콘다 파이썬 4.2.0을 사용하는 것은 파이썬 3.5를 사용하기 위해서이다. 이전판 저장소 페이지의 주소를 입력하기 어려울 때에는 아나콘다 파이썬 내려받기 페이지(https://www.continuum.io/downloads)에서 "Anaconda installer archive" 링크를 따라가도 된다.

### 3.1.3 설치 프로그램 실행

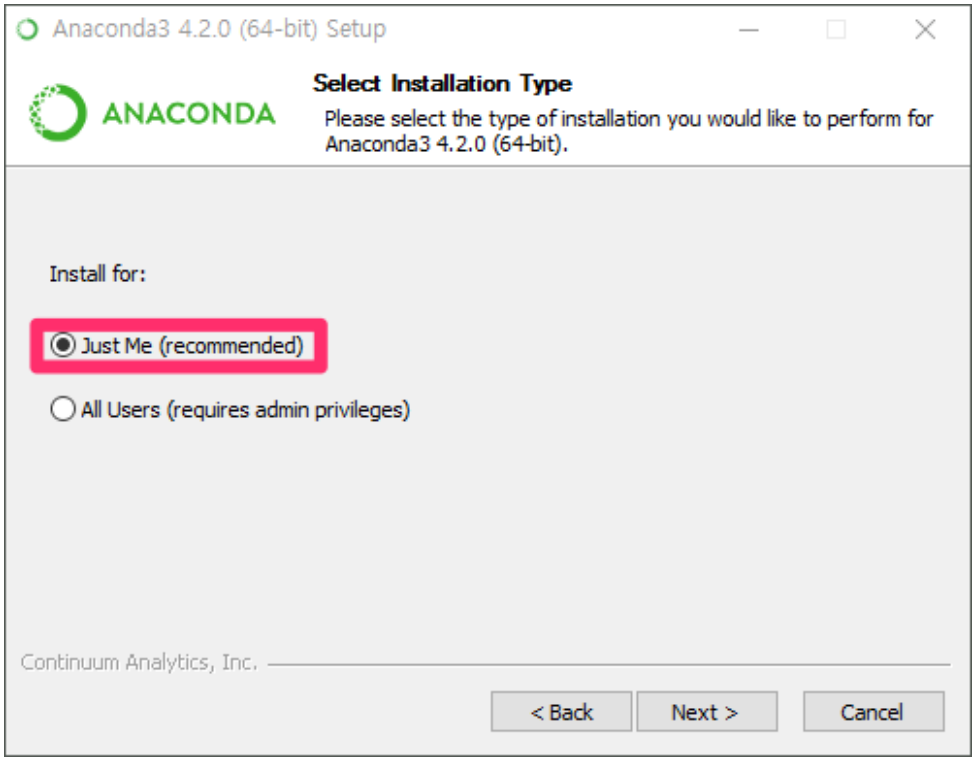
내려받은 설치 프로그램을 실행하여 주어진 기본 선택 사항대로 설치를 진행한다. 특히 아래의 두 그림과 같이 "Select Installation Type" 창과 "Advanced Installation Options" 창의 기본 선택 사항을 반드시 유지하도록 하자.

### 3.1.4 참고

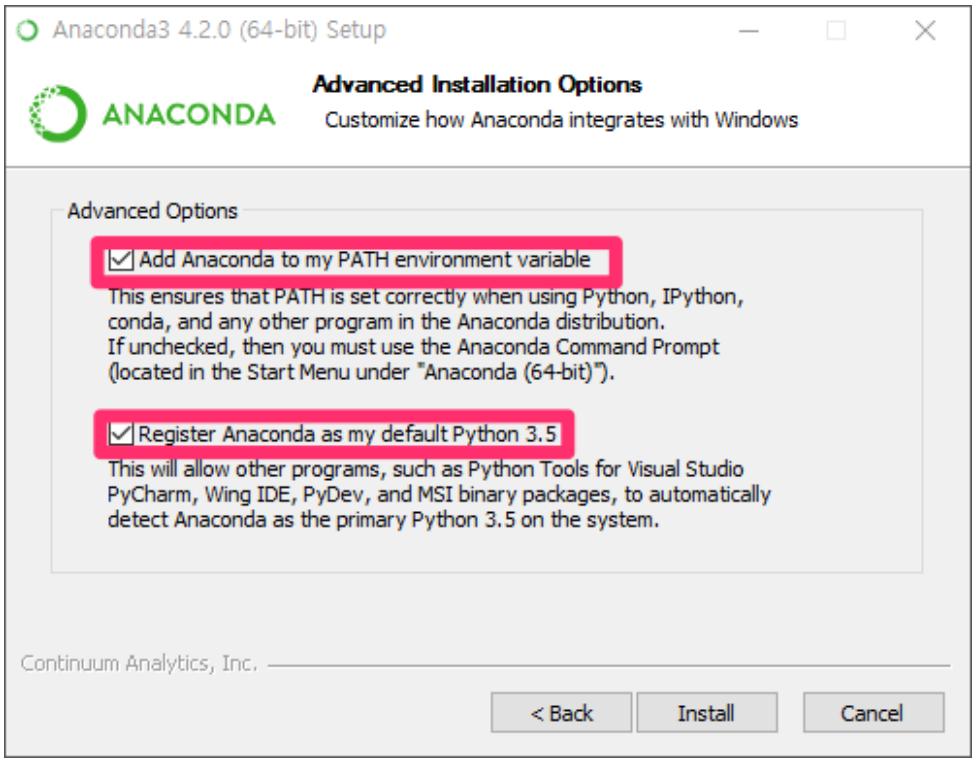
- 맥오에스, 또는 리눅스 사용자는 해당 운영체제용 아나콘다 파이썬 배포판을 설치할 수 있다. 설치 과정은 윈도우용과 그리 다르지 않다.
- 사용자의 선택에 따라 다른 배포판을 사용할 수도 있는데 라이브러리 패키지 설치 과정 등에 있어서 약간의 차이가 있을 수 있다.




〈그림 6〉 아나콘다 파이썬 설치 프로그램 내려받기




〈그림 7〉 설치 유형 선택




〈그림 8〉 설치 선택 사항 선택

맥오에스 사용자의 경우 운영체제에 포함된 파이썬을 이용하는 것보다 아나콘다 파이썬을 설치하여 사용하는 것이 훨씬 편리하다. 맥오에스에서의 본격적인 개발은 위해서 홈브루(https://brew.sh)를 이용한 작업 환경 구축이 권장되기도 한다.

## 3.2 보조 도구의 설치

### 3.2.1 텍스트 편집기

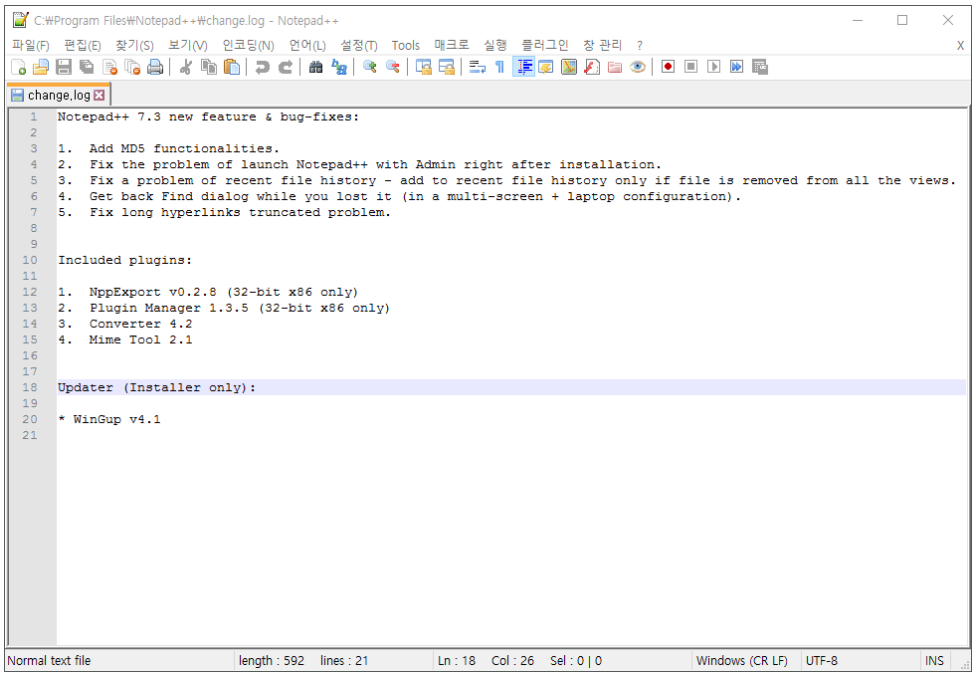
텍스트 마이닝의 분석 대상인 텍스트는 파일의 형태로 주어진다. 텍스트 마이닝 작업을 수행하면서 텍스트 파일의 내용을 눈으로 확인해야 경우가 종종 생긴다. 이 때에 쓸만한 텍스트 편집기가 큰 도움이 된다. 쓸만한 텍스트 편집기란 대체로 다음과 같은 조건을 만족해야 한다.

- 실행 속도가 빨라야 한다.
- 규모가 큰 파일도 열 수 있어야 한다.
- 여러 가지 인코딩의 파일을 열 수 있고 변환할 수 있어야 한다.

그밖에 플러그인을 통한 확장이 가능하다거나 무료로 쓸 수 있는 편집기라면 더욱 좋을 것이다.

이 강좌에서는 위의 조건을 만족하는 텍스트 편집기로 노트패드++를 추천한다. 현재 최신 버전은 7.3.3으로 배포 페이지(https://notepad-plus-plus.org/download/v7.3.3.html)에서 설치 프로그램을 내려받아 실행하여 기본 설치 사항을 따르며 간단히 설치할 수 있다. 32비트용과 64비트용이 있는데 대부분의 사용자는 64비트용을 설치하면 된다. 이 프로그램은 설치 프로그램과 본 프로그램에서 한국어 인터페이스를 제공한다. 설치를 마치고 프로그램을 실행하면 다음과 같은 화면이 표시된다.




〈그림 9〉 노트패트++ 처음 실행 화면

> 최근 인기를 끌고 있는 멀티 플랫폼 텍스트 편집기로 에이톰(https://atom.io)이 있다. 플러그인을 통해 다양한 코딩 관련 기능을 제공한다. 다만, 실행 속도가 다소 느리고 큰 파일을 열지 못한다는 약점이 있다.

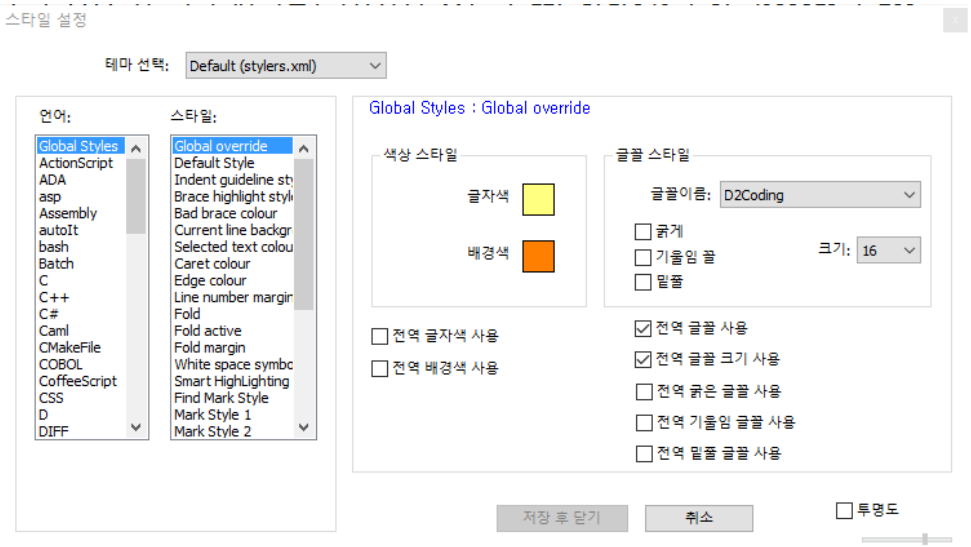
### 3.2.2 스프레드시트

자주 사용되는 정형 텍스트 파일 형식인 CSV 혹은 TSV 형식의 파일을 열어서 검색을 하거나 정렬을 할 때에 가장 유용한 도구가 스프레드시트이다. 스프레드시트의 최강자는 마이크로스프트 엑셀이고 이미 많은 사용자들이 사용하고 있으므로 따로 설명할 필요가 없을 것이다. 무료 공개 소프트웨어인 리브리오피스나 오픈오피스에 포함된 캘크도 엑셀 못지 않은 기능을 제공한다.

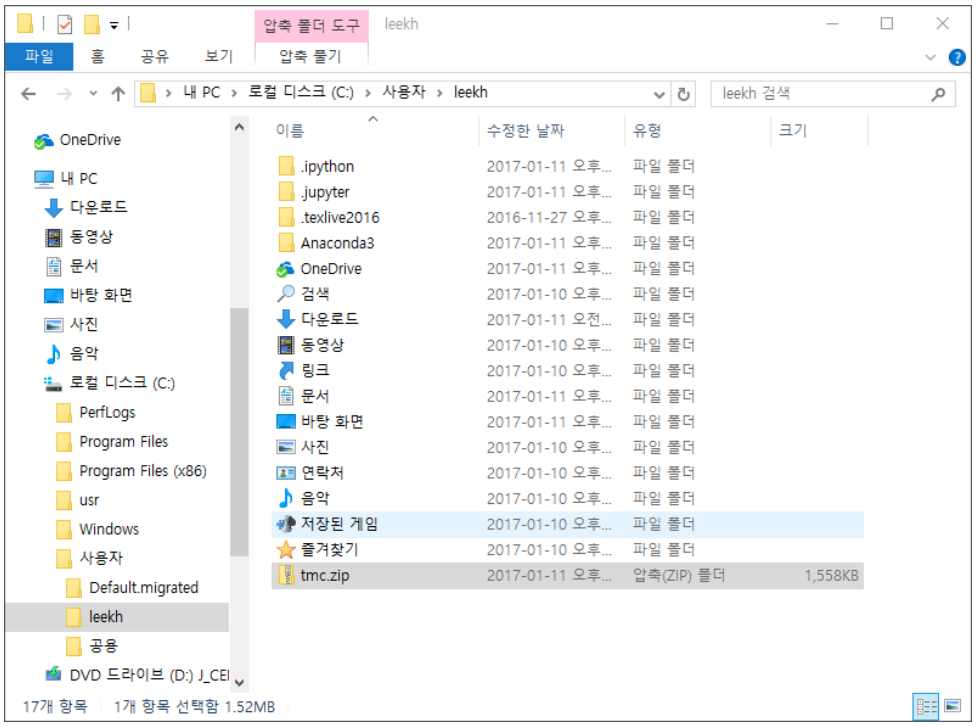
### 3.2.3 코딩용 글꼴

부차적인 것일 수도 있지만 텍스트 처리나 코딩을 많이 하는 사람들에게는 가독성이 좋은 글꼴이 매우 중요하다. 특히 코딩을 위해서는 영문과 숫자가 자폭이 일정한 고정폭 글꼴이 편리하다. 글꼴의 선택은 매우 주관적인 것이지만 이 강좌에서는 무난한 무료 글꼴로 네이버에서 배포하는 D2 Coding 글꼴을 추천한다. 깃헙 페이지(https://github.com/naver/d2codingfont)에서 제공하는 압축 파일을 내려받아 압축을 풀면 한 개의 `.ttc` 파일과 두 개의 `.ttf` 파일이 들어있는데 각각의 파일을 더블클릭하여 열어서 "설치"를 누르면 설치가 이루어진다.

설치한 D2 Coding 글꼴을 노트패드++에서 사용하려면, 메뉴에서 **설정** -- **스타일 설정**을 실행한다. 이어서 표시되는 대화창에서 그림과 같이 설정한다. 글꼴의 크기는 본인에게 적합하게 맞춘다.




〈그림 10〉 노트패드++의 글꼴 설정




〈그림 11〉 홈디렉토리의 `tmc.zip`

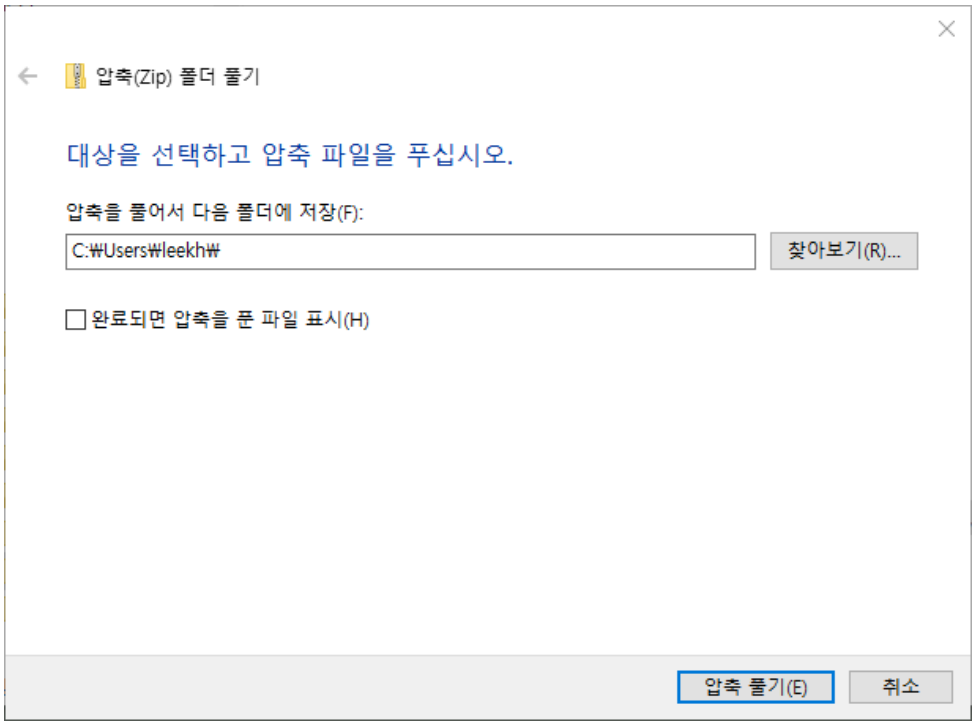
## 3.3 작업 디렉토리의 생성

제공하는 `tmc.zip` 파일을 내려받아 사용자의 홈디렉토리 (예: `C:\Users\leekh\`)에 놓는다. 탐색기 창으로 보면 다음과 같은 상태가 된다.

`tmc.zip` 파일 이름 위에 마우스 오른쪽 버튼을 클릭하여 표시되는 컨텍스트 메뉴에서 "압축 풀기"를 실행한다. 표시되는 압축 해제 디렉토리 설정 창에서 맨 끝의 `tmc`를 지워서 압축 파일이 사용자 홈디렉토리에 바로 풀리도록 한다.

그러면 `text-mining-camp`라는 이름의 디렉토리가 사용자 홈디렉토리 밑에 생성된다. 매 회차마다




〈그림 12〉 압축 해제 디렉토리 설정

제공되는 `tmc.zip` 파일을 같은 방법으로 내려받아 압축을 풀면 된다.

# 4 주피터 노트북을 이용한 파이썬 실습

## 4.1 주피터 노트북의 실행과 종료

주피터 노트북은 시스템 메뉴에서 [윈도우 - Anaconda3 (64-bit) - Jupyter Notebook]을 선택하여 실행할 수 있다. 그러면 명령행 창이 표시되면서 시스템의 기본 브라우저를 통해 주피터 노트북 앱이 실행된다. 주피터 노트북을 처음 실행할 때에는 구동에 시간이 오래 걸린다.

> 주피터 노트북은 어떤 브라우저에서나 동작하지만 크롬 브라우저에서 가장 잘 동작하는 것으로 알려져 있다. 크롬 브라우저를 주피터 노트북에서 사용하려면 시스템 기본 브라우저로 등록하면 된다.

주피터 노트북을 종료할 때에는 표시된 명령행 창을 마우스 포인터로 클릭하여 활성화한 다음 `Ctrl-C`를 두 번 누르거나 그 창 자체를 닫고 브라우저 창을 닫으면 된다.

본 강좌에서는 강의 자료로 제공되는 노트북 파일에 쉽게 접근하기 위해서 작업 디렉토리인 `text-mining-camp` 디렉토리에서 주피터 노트북을 실행하는 배치 파일(`run-notebook.bat`)을 제공한다. 탐색기에서 이 파일을 더블클릭하면 주피터 노트북이 실행된다.

## 4.2 주피터 노트북

주피터 노트북은 실행 가능한 코드, 수식, 시각화 결과, 설명문 등을 포함한 복합 문서를 생성하여 공유할 수 있도록 하는 웹 응용 프로그램으로 최근 데이터 정제와 변환, 시뮬레이션, 통계 모델링, 기계 학습 등의 다양한 분야에서 큰 인기를 끌고 있다.

주피터 노트북의 주요 요소는 다음과 같다.

- **노트북 문서**: 노트북 문서, 줄여서 노트북은 주피터 노트북 앱에 의해 생성된 복합 문서로 실행 가능한 코드(파이썬, 줄리아, R, 스칼라 등)와 그림, 수식 등을 포함하고 있는 파일이다. 확장자는 `.ipynb` 이다. 이 파일은 다른 사용자들과 공유가 가능하며, HTML, PDF, 마크다운 등의 형식으로 변환할 수도 있다.
- **주피터 노트북 앱**: 주피터 노트북 앱은 클라이언트-서버 방식의 웹 응용 프로그램으로 노트특 분서의 편집과 실행을 웹 브라우저를 통해 수행할 수 있도록 해준다. 주피터 노트북 앱은 로컬 기계에서 인터넷 연결 없이 실행할 수도 있고 원격 서버에 설치된 앱을 인터넷을 통하여 실행할 수도 있다. 주피터 노트북 앱은 노트북 파일의 목록 등이 표시되는 대시보드와 노트북 문서를 편집하고 실행하는 편집기로 구성된다.

## 4.3 주피터 노트북의 실행과 종료

주피터 노트북은 시스템 메뉴에서 [윈도우 - Anaconda3 (64-bit) - Jupyter Notebook]을 선택하여 실행할 수 있다. 그러면 명령행 창이 표시되면서 시스템의 기본 브라우저를 통해 주피터 노트북 앱이 실행된다. 주피터 노트북을 처음 실행할 때에는 구동에 시간이 오래 걸린다.

주피터 노트북은 어떤 브라우저에서나 동작하지만 크롬 브라우저에서 가장 잘 동작하는 것으로 알려져 있다. 크롬 브라우저를 주피터 노트북에서 사용하려면 시스템 기본 브라우저로 등록하면 된다.

주피터 노트북을 종료할 때에는 표시된 명령행 창을 마우스 포인터로 클릭하여 활성화한 다음 `Ctrl-C`를 두 번 누르거나 그 창 자체를 닫고 브라우저 창을 닫으면 된다.

## 4.4 주피터 노트북 기본 사용법

주피터 노트북을 실행하면 브라우저 창에 대시보드가 표시된다. 대시보드는 주피터 노트북 앱의 홈페이지 역활을 하며 현재 작업 디렉토리에 있는 노트북과 다른 파일들을 목록으로 보여준다. 앞서 설명한 방법으로 주피터 노트북을 실행하면 다음과 같은 창이 표시된다.

기본 작업 디렉토리는 사용자의 홈디렉토리 밑의 `Documents`(예: `C:\Users\leekh\Documents`)로 설정되고 그 아래에 `Python Scripts` 디렉토리가 생성된다.

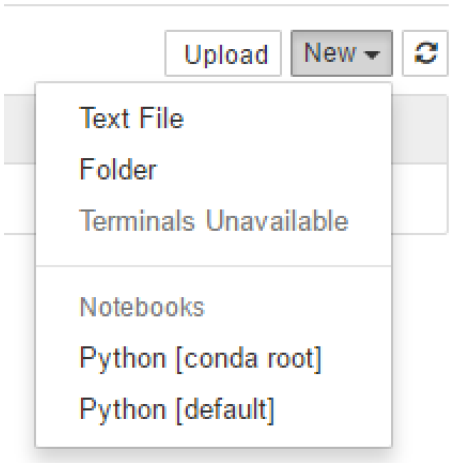
노트북을 새로 만들려면 상단의 "New" 단추를 누르고 드롭다운 목록에서 원하는 커널, 우리의 경우에는 "Python [conda root]" 혹은 "Python [default]"를 선택한다.

그리고 나면 다음과 같이 빈 노트북이 생성된다.

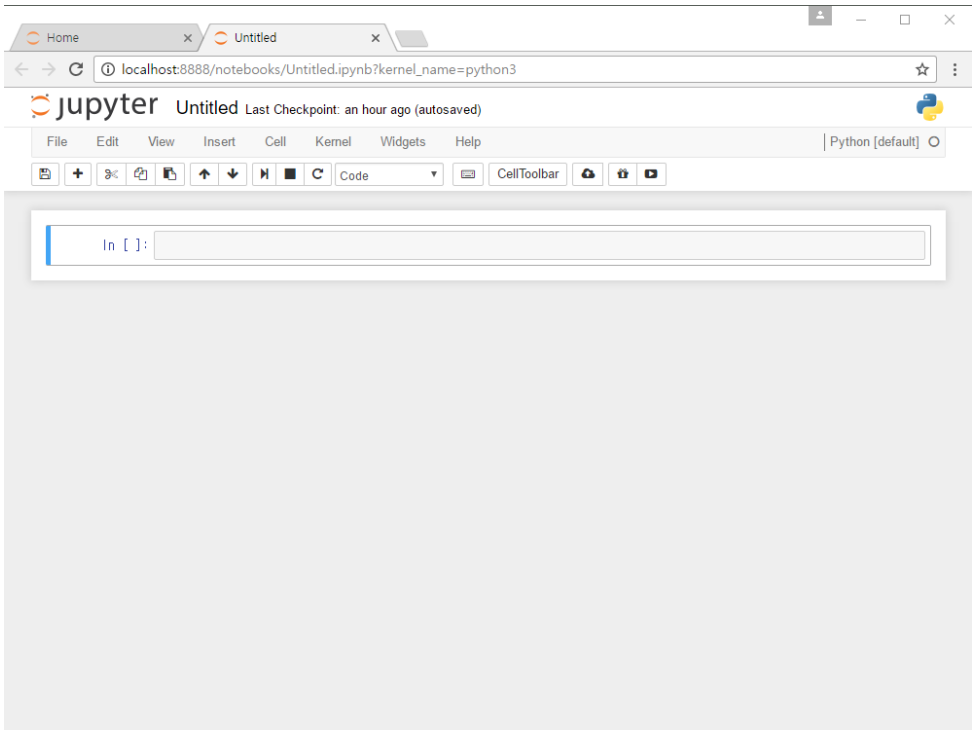
위에서 코드와 문서 마크업 등을 입력할 수 있는 칸을 "셀"이라고 부른다. 각 셀은 이른바 "모덜 편집기"로 구성이 되어 있어서 **명령** 모드와 **편집** 모드의 두 가지 모드를 가지고 있다.

셀이 편집 모드에 있을 때에는 셀의 왼쪽 경계선이 녹색으로 표시되며 커서가 깜빡거리며 대기한다. 이 상태에서는 일반 편집기에서와 마찬가지로 코드나 텍스트의 입력과 편집을 자유롭게 할 수 있다.


〈그림 13〉 주피터 노트북 대시보드




〈그림 14〉 새 노트북 만들기




〈그림 15〉 생성된 노트북

명령 모드는 회색 셀 경계선과 청색 왼쪽 여백선으로 표시된다. 이 상태에서는 셀 안의 코드를 실행하거나 셀 단위 이동, 셀의 복사와 붙이기 등의 동작을 실행할 수 있다.

주피터 노트북 편집기의 모든 동작은 마우스를 이용하여 수행할 수 있지만 단축키를 익혀서 키보드를 사용하는 것이 훨씬 더 효율적이다. 처음부터 모든 단축키를 기억하는 것은 어려운 일이고 다음의 핵심 단축키들부터 먼저 익숙해지자.

- `Shift-Enter`: 셀 실행. 커서가 위치하는 현재 셀을 실행하여 결과를 출력하고 다음 셀로 이동한다. `Shift-Enter`가 마지막 셀에서 눌려지면 새로운 코드 셀이 생성된다. 이 단축키는 메뉴에서 [Cell-Run Cells]을 실행하는 것과 같다.
- `Ctrl-Enter`: 제자리 셀 실행. 커서가 있는 현재 셀을 실행하지만 커서는 현재 셀에 머문다. 현재 셀을 계속 편집하면서 작은 실험을 해볼 때에 편리하다.
- `Alt-Enter`: 셀 실행, 다음 셀 삽입. 커서가 위치하는 현재 셀을 실행하고 바로 아래에 새로운 셀을 삽입한다.
- `ESC`: 편집 모드에서 명령 모드로 전환한다.
- `Enter`: 명령 모드에서 편집 모드로 전환한다.

## 4.5 파이썬 실습

복잡한 실세계의 객체를 컴퓨터로 효과적으로 표현하여 처리하려면 복합 자료형을 잘 다룰 수 있어야 한다. 여기서는 먼저

### 4.5.1 리스트

```
# 언패킹
a_list = [1, 2, 3, 4, 5, 6, 7]

[a, b, c, d, e, f, g] = a_list
print(a, b, c, d, e, f, g)

a, b, c, d, e, f, g = a_list
print(a, b, c, d, e, f, g)

list_of_tuple= [(1, 2), (3, 4), (5, 6)]

X, Y, Z = list_of_tuple
print(X, Y, Z)

(u, v), (w, x), (y, z) = list_of_tuple
print(u, v, w, x, y, z)
```

```
# 스플라이싱
a_list = [1, 2, 3, 4, 5, 6, 7]

print(a_list[1])
print(a_list[3:5])
print(a_list[-3])
print(a_list[-4:])

b_list = a_list.copy()
b_list[2:4] = ["w", "x", "y", "z"]
print(b_list)
```

```
# range() 함수
a_range = range(0, 11)
print(a_range)

a_range_list = list(range(0, 11))
print(a_range_list)
```

```
# for 문을 이용한 반복(iteration): if, continue, break, enumerate(),
# reversed(), sorted(), zip()
a_list = [1, 2, 3, 4, 5, 6, 7]

for n in a_list:
    print(n)

for n in a_list:
    if n % 2 == 0:
        continue

    print(n)

for n in a_list[:-1]:
    if n > 4:
        break

    print(n)
```

```
for i, n in enumerate(a_list):
    print(i, n)

for i, n in enumerate(a_list, 4):
    print(i, n)

b_list = reversed(a_list)

for n_a, n_b in zip(a_list, b_list):
    print(n_a, n_b)

for n_a, n_b in zip(a_list, sorted(reversed(a_list))):
    print(n_a, n_b)
```

```
# split()과 join() 함수
a_str = "패스트캠퍼스 貝水投感破水 fastcampus"

a_char_list = list(a_str)
print(a_char_list)

b_str = "★".join(a_char_list)
print(b_str)

a_word_list = a_str.split()
print(a_word_list)

a_csv_str = ",".join(a_word_list)
print(a_csv_str)

a_tsv_str = "\t".join(a_word_list)
print(a_tsv_str)
```

### 4.5.2 딕셔너리

```
# keys(), values(), items()

a_dict = {"사람": "발이 두 개인 동물", "개": "발이 네 개인 동물",
          "고양이": "발이 네 개인 동물"}
print(a_dict["사람"], a_dict["고양이"])

b_dict = {"이경규": {"성별": "남", "직업": "개그맨"},
          "유시민": {"성별": "남", "직업": "작가"}}
print(b_dict["유시민"]["직업"])

print(a_dict.keys())
print(a_dict.values())
print(a_dict.items())
```

```
# for 문을 이용한 반복
a_dict = {"사람": "발이 두 개인 동물", "개": "발이 네 개인 동물",
          "고양이": "발이 네 개인 동물"}

for k in a_dict:
    v = a_dict[k]
    print("키: {}, 값: {}".format(k, v))
```

```
        for k in a_dict.keys():
            v = a_dict[k]
            print("키: {}, 값: {}".format(k, v))

        for k, v in a_dict.items():
            print("키: {}, 값: {}".format(k, v))
```

### 4.5.3 연습 문제

1. 띠별 오늘의 운세 페이지(http://www.unsin.co.kr/unse/free/todayline/result)에서 띠별 운세를 복사하여 문자열의 리스트로 구조화하라.
2. 사용자로부터 한글 이름을 입력 받아 전체 자모수를 구하여 이를 12로 나눈 나머지를 구하여 해당 인덱스의 운세를 위의 리스트에서 찾아서 표시하라. (unicodedata 모듈 사용)
3. (선택) 위의 코드를 스크립트 파일 `fortune.py`로 저장하고 명령행 창에서 실행해 보라.(`%%writefile` 매직 명령 사용)
4. (선택) 추가로 `%lsmagic`, `%load`, `%run` 매직 명령의 사용법을 알아보라.