

2022 미래차 충전인프라구축운영 전문인력 양성교육
전기차 충전기 빅데이터 심화

Python 프로그래밍

2022.11



Python 소개

Python 이란? / Anaconda 소개 및 설치 / Jupyter Notebook 소개 / Spyder 소개 / 사용자 패키지 소개 / 패키지 설치

Matplotlib - IT위키

날씨마루 사용방법과 이용신청

bd.kma.go.kr/kma2020/dta/reqst/dtaReqstInfo.do?menuCd=F030102000&pageNum=4

날씨마루 기상기후 빅데이터 분석 플랫폼

소개 기상융합서비스(분석사례) 분석환경 게시판 커뮤니티 콘테스트

검색어를 입력하세요

로그인

분석환경

기상기후 빅데이터를 다른 분야 데이터와 접목시켜 분석할 수 있는 환경을 제공합니다.

VIEW

분석교육실습 R을 활용한 분석 교육 동영상(R|Python) Python을 활용한 분석 Fortran을 활용한 분석

비정형 도구

데이터 시각화

빅데이터 분석도구 Rstudio Python Fortran

데이터 기상데이터 업로드 데이터 웹데이터 분석결과 다운로드

마이페이지 나의 이용 현황 1:1 상담 비밀번호 재설정

사용자별 바로가기

융합서비스 이용자

분석교육 실습자

분석환경 이용신청 데이터 이용 신청 비정형데이터 시각화도구 분석도구(Rstudio) **분석도구(Python)** 분석도구(Fortran)

분석도구(Python) 따라하기

분석도구 Python Data 클릭

STEP 5 파일 복제

https://bd.kma.go.kr/kma2020/svc/intro/analyzeEnvinfo.do?pageNum=5&menuCd=F040000000

2022 미래차 중전인프라구축운영 전문인력 양성교육

Python 이란?

- (1991년) 귀도 반 로섬(Guido van Rossum)[프로그래머]이 발표한 고급 프로그래밍 언어
- 플랫폼 독립적
- 인터프리터(Interpreter)식
- 객체지향적(Object oriented)
- 동적 타이핑(Dynamically typed) 대화형 언어 - 실행 시간에 자료형을 검사 \leftrightarrow 정적 타이핑: 컴파일 시점과 같이 실행 전 시점에 타입이 올바른지를 체크



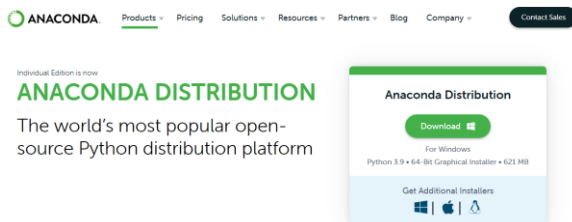
Anaconda 소개 및 설치

- Anaconda?
 - Continuum Analytics에서 개발된 무료 배포판
 - 콘솔판과 GUI판 함께 제공
 - Python을 포함해 수학, 과학, 시각화 등에 필요한 거의 모든 패키지들(Numpy, IPython, Matplotlib 등)이 한번에 설치
- 다운로드
 - <https://www.anaconda.com/download/>

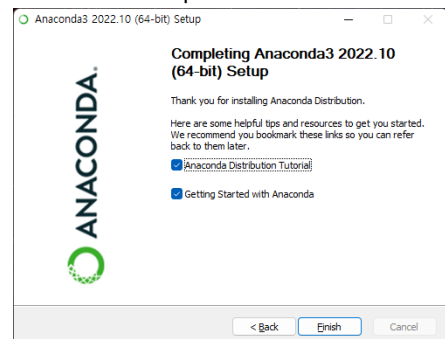


Anaconda 소개 및 설치

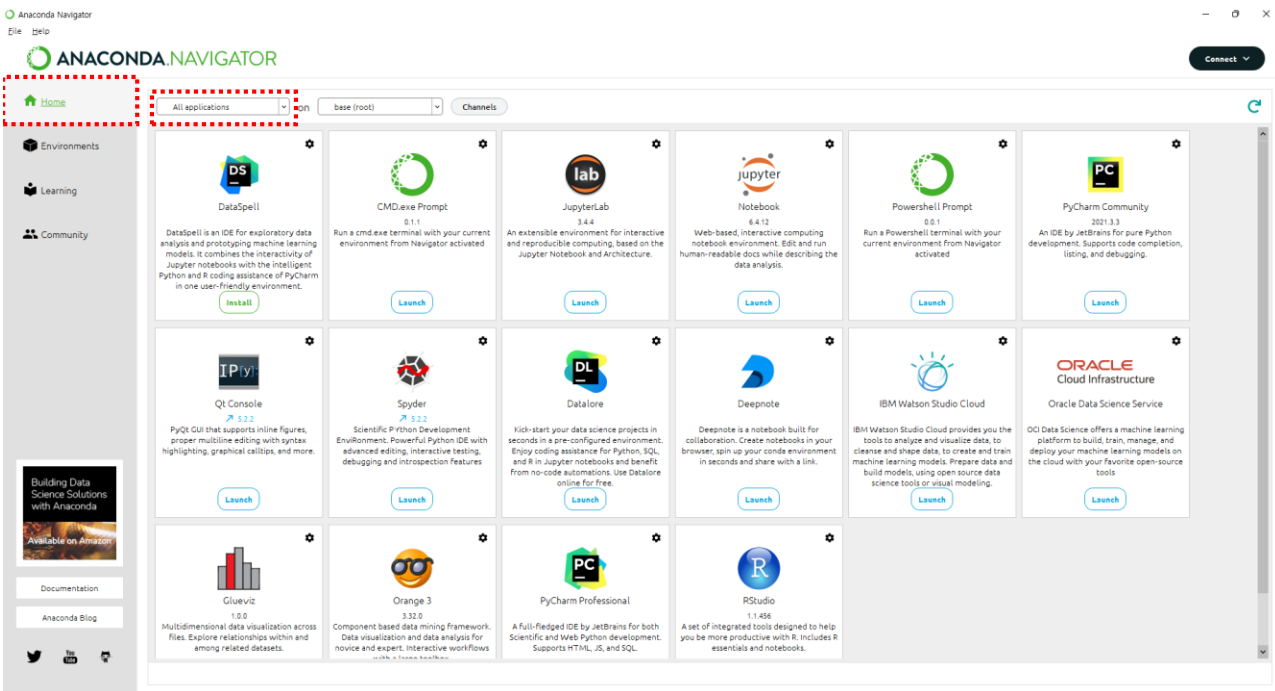
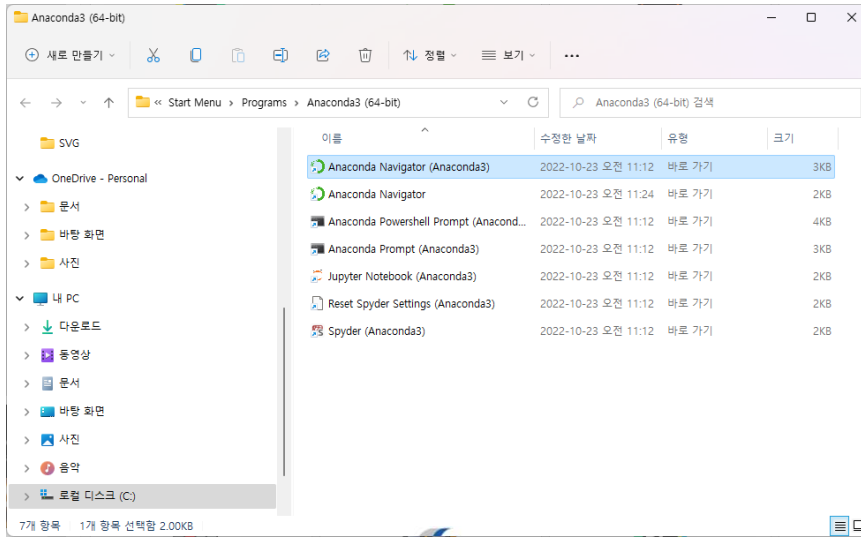
- 설치 - 2GB 이상의 디스크 여유 공간



- Step1 - Welcome to Anaconda3 2022.10(64-bit) Setup
- Step2 - License Agreement
- Step3 - Select Installation Type
- Step4 - Choose Install Location
- Step5 - Advanced Installation Options
- Step6 - Installation Complete



Anaconda 소개 및 설치



Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Connect

Home

Environments

Learning

Community

base (root)

Search Environments

Installed Channels Update index...

Search Packages

| Name | Description | Version |
|-----------------------|---|---------|
| _jupyterlab_nb_ext... | A configuration metapackage for enabling anaconda-bundled jupyter extensions | 0.1.0 |
| alabaster | Configurable, python 2+3 compatible sphinx theme. | 0.7.12 |
| anaconda | Simplifies package management and deployment of anaconda | 2022.10 |
| anaconda-client | Anaconda.org command line client library | 1.11.0 |
| anaconda-project | Tool for encapsulating, running, and reproducing data science projects | 0.11.1 |
| anyio | High level compatibility layer for multiple asynchronous event loop implementations on python | 3.5.0 |
| appdirs | A small python module for determining appropriate platform-specific dirs. | 1.4.4 |
| argon2-cffi | The secure argon2 password hashing algorithm. | 21.3.0 |
| argon2-cffi-bindings | Low-level python cffi bindings for argon2 | 21.2.0 |
| arrow | Better dates & times for python | 1.2.2 |
| astroid | A abstract syntax tree for python with inference support. | 2.11.7 |
| astropy | Community-developed python library for astronomy | 5.1 |
| atomicwrites | Atomic file writes | 1.4.0 |
| attrs | Attrs is the python package that will bring back the joy of writing classes by relieving you from the drudgery of implementing object protocols (aka dunder methods). | 21.4.0 |
| automat | Self-service finite-state machines for the programmer on the go | 20.2.0 |
| autopep8 | A tool that automatically formats python code to conform to the pep 8 style guide | 1.6.0 |
| babel | Utilities to internationalize and localize python applications | 2.9.1 |
| backcall | Specifications for callback functions passed in to an api | 0.2.0 |
| backports | Namespace for backported python features. | 1.1 |
| backports.functio... | | 1.6.4 |

416 packages available

Create Clone Import Backup Remove

Anaconda Navigator

File Help

ANACONDA.NAVIGATOR

Connect

Home






























Environments

Learning

Community

Documentation (29)

Search

| | | | | | | | |
|---|---|---|--|---|---|--|--|
|  Python Tutorial Read |  Python Reference Read |  Anaconda Package List Read |  Pandas Documentation Read |  Numpy Documentation Read |  Scipy Documentation Read |  Matplotlib Documentation Read |  Bokeh User Guide Read |
|  Anaconda Nucleus Documentat... |  Anaconda Documentation Read |  Anaconda Navigator Documentation Read |  Using MRO or R with Conda Read |  Microsoft R Open: The Enhance R Distribution Read |  The Comprehensive R Archive Network (CRAN) Read |  The Python Package Index (PyPi) Read |  Dask documentation Read |
|  Conda & Conda-Build Read |  Jupyter documentation Read |  Spyder documentation Read |  VSCode (jython) Read |  Orange documentation Read |  PyCharm Documentation Read |  PyTorch Documentation Read |  PyViz Documentation Read |
|  Scikit-Learn Documentation |  Tensorflow Documentation |  Conda-Forge |  Jupyterlab Documentation |  Numba Documentation | | | |

Anaconda Notebooks

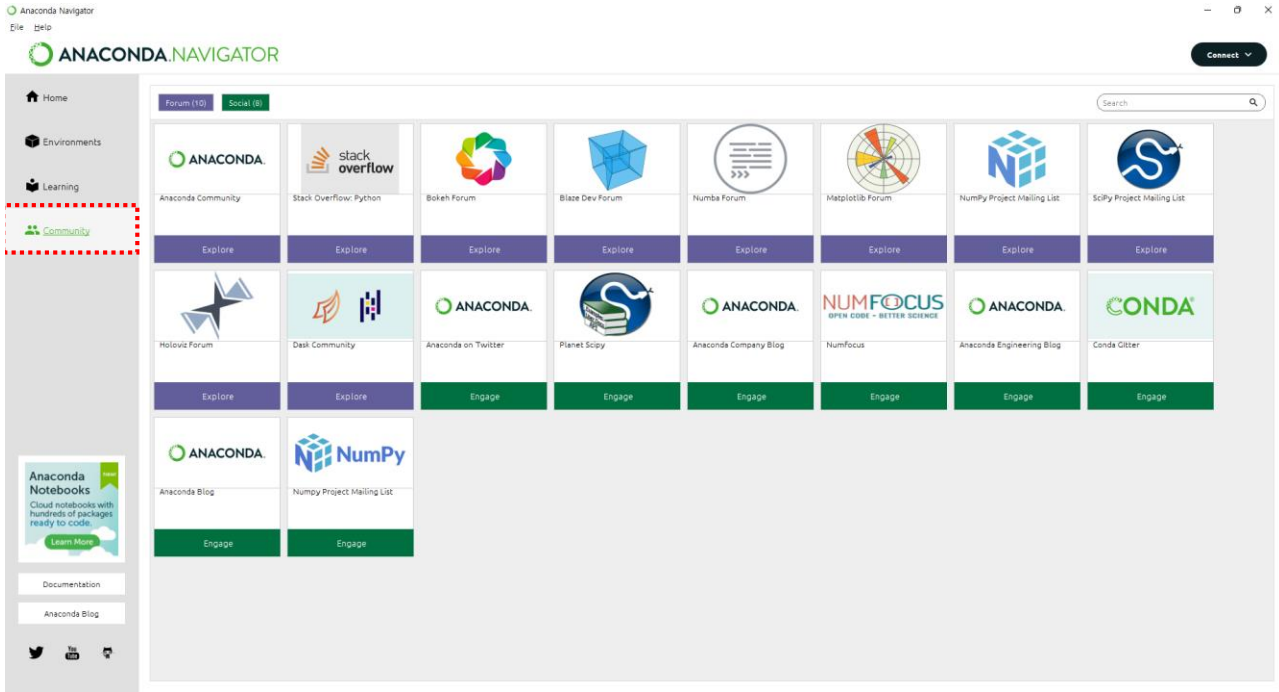
Cloud notebooks with hundreds of packages ready to code.

Learn More

Documentation

Anaconda Blog

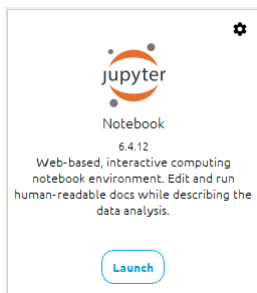
Twitter YouTube GitHub



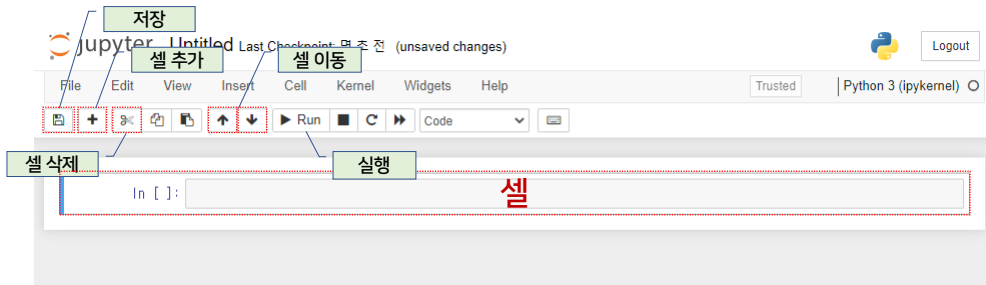
2022 미래차 중전인프라 구축 운영 전문인력 양성 교육

Jupyter Notebook 소개

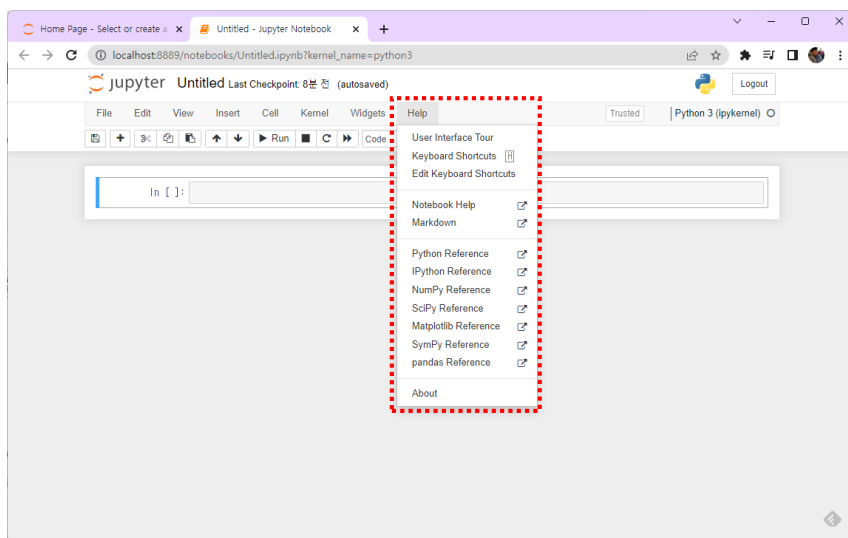
- Anaconda 설치 시 함께 서치되는 프로그램들 중 하나로 Python을 프로그래밍 하는데 필요한 도구들을 제공하는 무료 통합개발환경(IDE)
- Jupyter Notebook 실행



Jupyter Notebook 소개

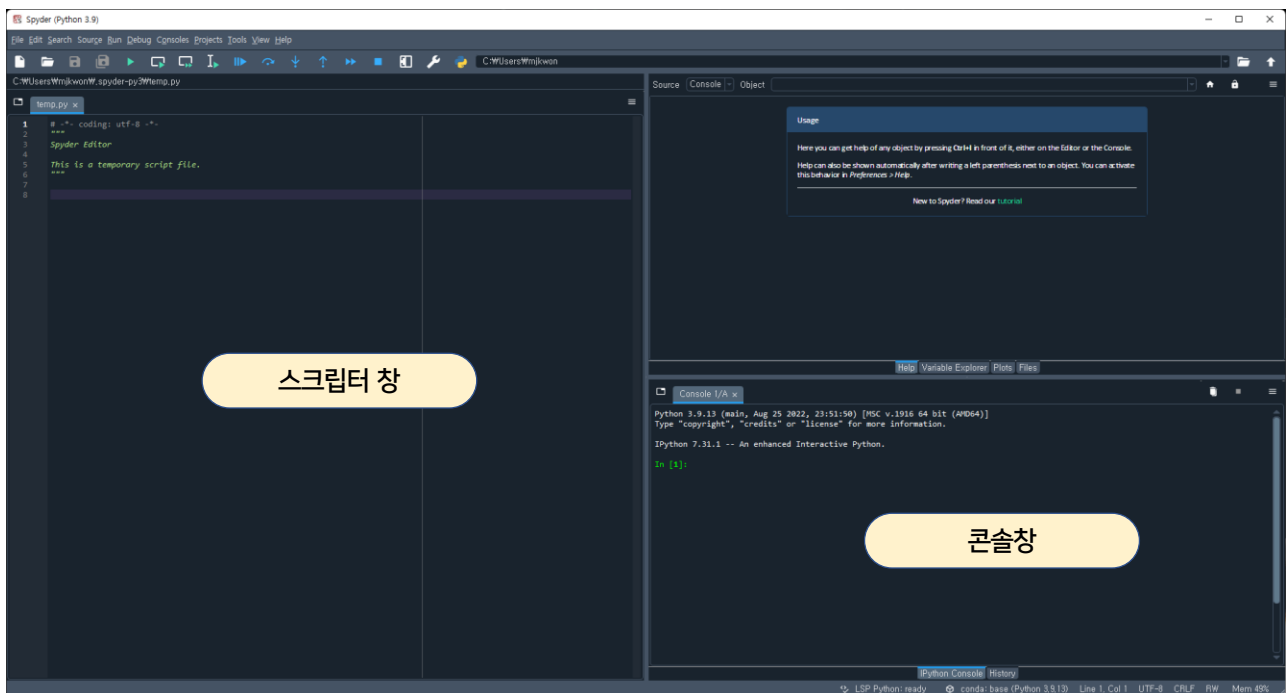
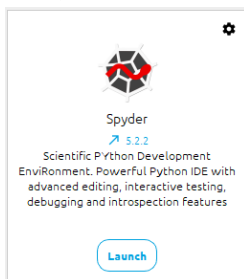


Jupyter Notebook 소개 - Help



Spyder 소개

- Spyder(Scientific PYthon Development EnviRonment)는 Anaconda 설치시 함께 설치되는 프로그램 중 하나로 Python으로 프로그래밍 하는데 필요한 도구들을 제공하는 무료 통합개발환경(IDE)
- Spyder 실행



사용 패키지 소개 - Numpy

- Numpy - Numerical Python의 약자
- **고성능 과학 계산용 패키지**
- Matrix와 Vector와 같은 Array 연산을 할 때 사용하며 표준 라이브러리 처럼 사용
- 발음 - [넘파이][넘피/눔파]
- 특징
 - 일반 List에 비해 빠르고, 메모리를 효율적으로 사용
 - 반복문 없이 데이터 배열에 대한 처리를 지원하여 빠르고 편리
 - 선형대수와 관련된 다양한 기능을 제공
 - C, C++, 포트란 등의 언어와 통합이 가능



사용 패키지 소개 - Pandas

- Python 데이터 분석을 효율적으로 하기 위한 **데이터 분석** 라이브러리(오픈 소스(BSD 라이선스)로 공개)
- 데이터를 **읽어** 들이거나 **총 계산량을 표시**, **그래프화** 등 데이터 분석이나 **기계 학습**에 필요한 작업을 간단히 실행
- 주요한 코드는 Cpython 또는 C로 작성되어 있으므로, Python 만으로 데이터를 분석할 때와 비교하면 더욱 **고속의 처리**가 가능
- Python 머신 러닝에 필수 - 데이터 분석(데이터 사이언스)는 기계학습을 실행하기 위한 **전처리**(데이터 읽어들이기, 클리닝, 결측값의 보완, 정규화 등)이 작업 전체의 8~9할을 차지



사용 패키지 소개 - Matplotlib

- Python에서 그래프를 그릴 수 있는 라이브러리
- 자료를 차트(Chart)나 플롯(Plot, 선형 도표, 산점도, 막대 도표, 히스토그램 등)으로 시각화(Visualization)하는 패키지
- 특징
 - 표준 플롯을 쉽게 그릴 수 있을 뿐만 아니라 복잡한 플롯과 세부적인 변경도 자유로운 유연한 라이브러리
 - Numpy 및 Pandas(Series, DataFrame)가 제공하는 자료들과도 잘 연동



사용 패키지 소개 - StatsModels

- 통계 분석을 위한 Python 패키지
- 통계(Statistics)
 - 각종 검정(test) 기능
 - 커널 밀도 추정
 - Generalized Method of Moments
- 회귀분석(Linear Regression)
 - 선형 모형(Linear Model)
 - 일반화 선형 모형(Generalized Linear Model)
 - 강인 선형 모형(Robust Linear Model)
 - 선형 혼합 효과 모형(Linear Mixed Effects Model)
 - ANOVA(Analysis of Variance)
 - Discrete Dependent Variable(Logistic Regression 포함)
- 시계열 분석(Time Series Analysis)
 - ARMA / ARIMA Process
 - Vector ARIMA Process
- 관련 사이트: <http://www.statsmodels.org>



패키지 설치(1/4)

• 명령 프롬프트(cmd)창을 통한 설치

```

Anaconda Prompt (Anaconda3)
(base) C:\Users\mjkwon> pip install StatsModels
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: StatsModels in c:\programdata\anaconda3\lib\site-packages (0.13.2)
Requirement already satisfied: numpy>=1.17 in c:\programdata\anaconda3\lib\site-packages (from StatsModels) (1.21.5)
Requirement already satisfied: scipy>=1.3 in c:\programdata\anaconda3\lib\site-packages (from StatsModels) (1.9.1)
Requirement already satisfied: pandas>=0.25 in c:\programdata\anaconda3\lib\site-packages (from StatsModels) (1.4.4)
Requirement already satisfied: patsy>=0.5.2 in c:\programdata\anaconda3\lib\site-packages (from StatsModels) (0.5.2)
Requirement already satisfied: packaging>=21.3 in c:\programdata\anaconda3\lib\site-packages (from StatsModels) (21.3)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in c:\programdata\anaconda3\lib\site-packages (from packaging>=21.3->StatsModels) (3.0.9)
Requirement already satisfied: pandas>=0.25->StatsModels (2022) in c:\programdata\anaconda3\lib\site-packages (from pandas>=0.25->StatsModels) (2.0.2)
Requirement already satisfied: six in c:\programdata\anaconda3\lib\site-packages (from patsy>=0.5.2->StatsModels) (1.16.0)
(base) C:\Users\mjkwon>
  
```

기 설치된 경우



패키지 설치 (2/4)

• Jupyter를 통한 설치

```

In [1]: !pip install scikit-learn
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: scikit-learn in c:\programdata\anaconda3\lib\site-packages (1.0.2)
Requirement already satisfied: joblib>=0.11 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.1.0)
Requirement already satisfied: numpy>=1.14.6 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.21.5)
Requirement already satisfied: threadpoolctl>=2.0.0 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (2.2.0)
Requirement already satisfied: scipy>=1.1.0 in c:\programdata\anaconda3\lib\site-packages (from scikit-learn) (1.9.1)
  
```

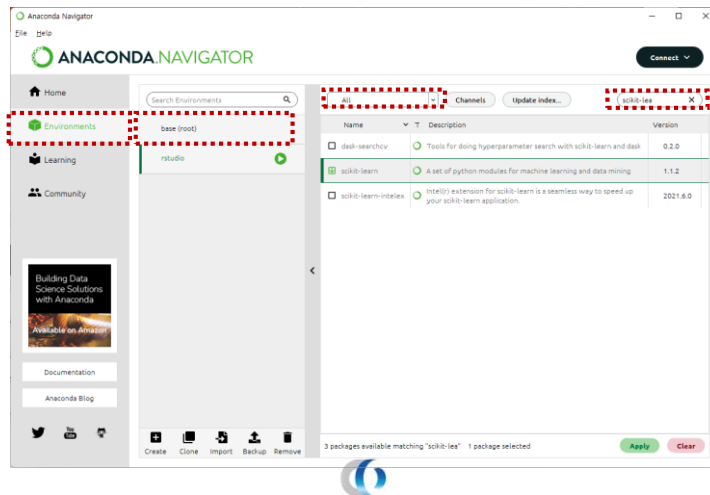
주의: 느낌표(!) 붙이기

기 설치된 경우



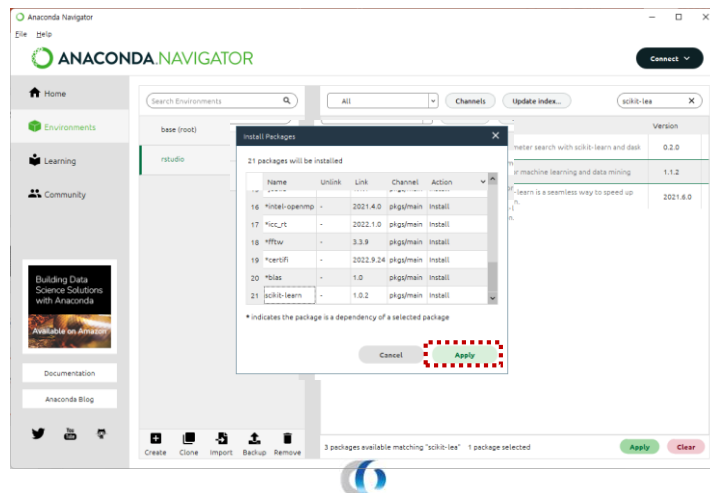
패키지 설치 (3/4)

• Anaconda Navigator를 통한 설치



패키지 설치 (4/4)

• Anaconda Navigator를 통한 설치



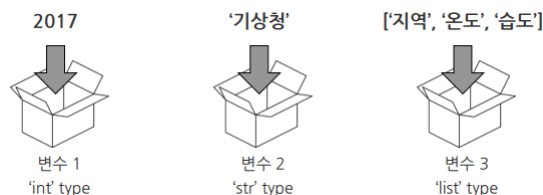
Python 프로그래밍

자료형 / 제어문(연산자) / 함수

2022 미래차 중전인프라구축운영 전문인력 양성교육

자료형(Data Type) - 변수

- 변수(Variable), 부울(Bool), 숫자(Numeric), 문자(String), 리스트(List), 딕셔너리(Dictionary), 튜플(Tuple)
- **변수(Variable)**
 - 데이터를 담는 메모리 공간, 이름 지정
 - 변수에 부울, 숫자, 문자, 목록, 이미지 등을 담을 수 있음
 - Python은 코드가 실행될 때 변수의 자료형을 판단하는 동적 형식 언어(Dynamic typed language)임



자료형(Data Type) – 부울 / 숫자

• 부울

- 논리 자료형 TRUE(1) or FALSE(0)
- 참(True) 또는 거짓(False)을 표현
- 숫자의 대소나 논리연산을 통한 결과로 참과 거짓을 출력
- 프로그램의 흐름을 제어할 때 사용

• 숫자

- 정수, 실수, 복소수
- 정수(int): 음의 정수, 0, 양의 정수
- 실수(float): 소수점을 표현하는 방식 중 하나인 부동 소수형을 제공
- 복소수(complex): 실수와 허수의 합으로 이루어진 수(단, $i \rightarrow j$)
- 숫자형을 이용한 기본적인 사칙연산과 math 모듈을 이용해 파이, 자연상수, 팩토리얼, 라디안, 삼각함수, 로그 등을 계산 가능



자료형(Data Type) – 문자

- 텍스트를 다루는 자료형
- 작은 따옴표('text') 또는 큰 따옴표("text")의 쌍으로 텍스트를 감싸서 표현
- 여러 줄로 이루어진 문자열은 작은 따옴표 3개("""strings""") 또는 큰 따옴표 3개("""strings""")의 쌍으로 텍스트를 감싸서 표현
- 숫자와 문자는 int(), float(), complex(), str() 등의 함수를 이용해 형태를 변경
- 문자열을 합칠 때에는 +연산자 사용



자료형(Data Type) – 리스트

- 데이터의 목록을 다루는 자료형
- 리스트 안에는 어떤 데이터든 담을 수 있으며, 리스트 안의 개별 데이터를 요소(Element)라 함
- 리스트를 만들 때 대괄호 [와] 사이에 데이터 또는 변수 목록을 입력하며, 각 데이터는 콤마(,)로 구분
- 리스트 안에 있는 개별 데이터를 참조할 때는 리스트의 이름 뒤에 대괄호를 붙이고 대괄호 사이에 참조하고자 하는 첨자(index)를 입력
- 리스트를 결합할 때 + 연산자 사용

[‘기상청’, 2017, ‘파이팅’]



자료형(Data Type) – 딕셔너리

- 키(Key)-값(Value)의 쌍으로 구성, 첨자는 키라하고, 이 키가 가리키는 데이터를 일컬어 값(Value)라 함
- 리스트처럼 첨자를 이용해서 요소에 접근, 변경 가능
- 리스트는 데이터를 저장할 요소의 위치를 인덱스를 사용하는 반면, 딕셔너리는 키 데이터를 그대로 사용
- 딕셔너리를 만들 때는 중괄호 { 와 }을 이용
- 새로운 키-값을 입력하거나 그 안에 있는 요소를 참조할 때는 리스트처럼 대괄호 [와]를 이용

{ ‘key’: ‘value’ }



자료형(Data Type) – 튜플

- 튜플 – ‘N개의 요소로 된 집합’
- 리스트는 변경 가능, 튜플은 **변경 불가능**
- 튜플은 만드는 방법은 괄호 (와)를 이용해 만들거나 괄호를 생략해 만들 수 있음
- 리스트와 동일한 방식의 더하기 / 곱하기 연산 존재
- 튜플은 변형이 불가능한 자료형으로서 제공하는 메소드는 index() 와 count() 두개뿐

(‘기상청’, 2017, ‘파이팅’)



연산자

• 논리연산자

| 연산자 | 설명 |
|--------|-------------------------------------|
| not(!) | 피연산자 부정 한 결과를 True/False로 반환 |
| and(&) | 두 피연산자 간의 논리곱 결과를 반환 |
| or() | 두 피연산자 간의 논리합 결과를 반환 |

• 비교연산자

| 연산자 | 연산자 |
|-----|-----|
| == | != |
| > | >= |
| < | <= |



제어문

• if 제어문

if 조건 :
(들여쓰기) 실행문

- 조건에 맞는 경우 프로그램이 수행되도록 할 수 있음

[구조]

```
if 조건 :
    수행할 코드 1
else :
    수행할 코드 2
```

[예제]

```
alarm = "강우"
if alarm == "강우":
    print("우산을 챙기세요")
else :
    print("좋은 하루 되세요!")
```

[구조]

```
if 조건1 :
    수행할 코드 1
elif 조건2 :
    수행할 코드 2
else :
    수행할 코드 3
```

[예제]

```
RN_DAY = 10
if RN_DAY >= 20 :
    print("폭우경보")
elif RN_DAY >= 10 :
    print("폭우주의보")
else :
    print("정상")
```



제어문

• while: 조건이 참인 동안 코드를 반복 수행

[구조]

```
while 조건 :
    수행할 코드
```

[예제]

```
i = 0
while i < 10 :
    print i
    i = i + 1
```

• for: 순서열의 끝에 도달할 때까지 반복 수행

[구조]

```
for 반복변수 in 순서열 :
    수행할 코드
```

[예제]

```
for i in [1, 2, 3] :
    print (i)
```

```
for i in range(0, 10, 2) :
    print (i)
```



제어문

• 반복문 제어하기

continue / break

• continue

- 특정 조건에서만 코드 실행없이 다음 반복으로 넘어갈 때 사용

```
num = 0
while num < 10:
    num += 1
    if num == 5:
        continue
    print(num)
```

• break

- 특정 조건에서만 반복문을 중단할 때 사용

```
num = 0
while 1:
    print(num)
    if num == 10:
        break
    num += 1
```



함수(Function)

• 정의 / 호출 / 반환 과정

• 정의(Define) [구조]

* 함수 정의의 기본 구조

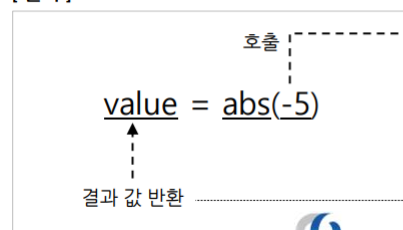
```
def 함수명(매개변수 목록):
    수행할 코드
    return 결과
```

[예제]

```
def hello():
    print("hello world")
```

• 호출(Call)과 반환(Return) 과정

[원리]



[예제]

```
def abs(num):
    if(num < 0):
        result = num * -1
    else:
        result = num
    return result
```



데이터 다루기

데이터 로드 / 데이터 생성 / 데이터 접근 / 데이터 변환 / 데이터 저장

2022 미래차 중전인프라 구축운영 전문인력 양성교육

데이터 로드

- 데이터를 불러올 때 Pandas 패키지 이용
- Pandas **데이터 로드(Load) 함수**

| Pandas 함수 | 설명 |
|---------------------------|-------------------|
| <code>read_csv()</code> | 쉼표(,)로 구분된 파일 읽기 |
| <code>read_excel()</code> | Excel 파일 읽기 |
| <code>read_table()</code> | 탭(Wt)으로 구분된 파일 읽기 |
| <code>read_fwf()</code> | 구분자가 없는 파일 읽기 |

- csv 파일 불러오기

[구조]

```
import pandas as pd

데이터명 =
pd.read_csv('파일경로/파일명.csv')
```

[예제]

```
import pandas as pd

mydata =
pd.read_csv('C:/Users/user/Desktop/example/mydata.csv')
```

* 함수의 이름이 길거나 어떤 필요에 의해 함수의 이름에 가명(Alias)을 주고 싶은 경우, 위 예시처럼 "함수명 as Alias" 와 같은 표현을 사용할 수 있습니다.



데이터 로드

- Pandas를 이용해 데이터 로드 시 옵션 키워드

| 옵션 종류 | 설명 |
|-----------------|----------------------|
| sep / dilimiter | 구분자를 지정 |
| header | header를 컬럼이름으로 지정 |
| index_col | index로 사용할 컬럼을 지정 |
| encoding | 파일의 인코딩을 지정 (한글 !!!) |

- 파일 불러올 때 index 지정하기

[구조]

```
import pandas as pd
데이터명 =
pd.read_csv('파일경로/파일명.csv',
index_col='컬럼명or컬럼번호')
```

[예제] * 첫번째 컬럼을 index로 지정하는 경우

```
import pandas as pd
mydata =
pd.read_csv('C:/Users/user/Desktop/ex
ample/mydata.csv', index_col='0')
```

• .txt 파일 불러오기 (구분자가 |로 되어 있는 경우)

[구조]

```
import pandas as pd
데이터명 =
pd.read_csv('파일경로/파일명.csv',
sep='구분자')
```

[예제]

```
import pandas as pd
mydata =
pd.read_csv('C:/Users/user/Desktop/ex
ample/mydata.csv', sep='|')
```



데이터 로드

- 기타 옵션

| 옵션 종류 | 설명 |
|-------------|-----------------------|
| skiprows | 읽지 않을 row 번호 list를 지정 |
| na_values | NA로 인식할 값 list를 지정 |
| comment | 주석으로 분류할 문자열을 지정 |
| paste_date | 날짜로 구분할 컬럼 list를 지정 |
| date_parser | 날짜 변환 시 사용할 함수 지정 |
| converters | 컬럼을 읽어올 때 적용할 함수 지정 |
| nrows | 몇 번째 줄까지 읽을 것인지 지정 |
| skipfooter | 무시할 파일의 마지막 줄 수를 지정 |



데이터 생성

- 생성한 데이터를 DataFrame형식으로 반환
- 데이터 생성(dictionary 사용해 입력하여 생성)

[구조]

```
import pandas as pd

# dictionary 생성
dictionary 파일명
= {'컬럼명1': ['chr1','chr2','chr3'], '컬럼명2':
[num1, num2, num3]}
# 변수타입이 character인 경우 값을 작은 따옴표
사이에 입력, 숫자인 경우 숫자만 입력

# dictionary를 DataFrame으로 변환
데이터 프레임 파일명 =
pd.DataFrame(dictionary 파일명,
index=['a','b','c','d','e'])
# index 키워드 뒤에 row 수만큼 index 명칭을 작은
따옴표 사이마다 입력
```

[예제]

```
import pandas as pd

data={'ID': ['A1','A2','A3','A4','A5'],
'X1': [1, 2, 3, 4, 5],
'X2': [3.0, 4.5, 3.2, 4.0, 3.5]}

mydata = pd.DataFrame(data,
index=['a','b','c','d','e'])
```

```
In [2]: 1 import pandas as pd
2 data={'ID': ['A1','A2','A3','A4','A5'],
3 'X1': [1,2,3,4,5],
4 'X2': [3.0, 4.5, 3.2, 4.0, 3.5]}
5
6 mydata = pd.DataFrame(data,
7 index=['a','b','c','d','e'])
```

```
In [4]: 1 whos

Variable Type Data/Info
-----
data dict n=3
mydata DataFrame ID X1 X2 n=5 a 1 4.0 b 2 4.0 c 3 3.5
pd module <module 'pandas' from 'C:\...\es\pandas\__init__.py'>
```



데이터 접근

- 접근하는 방법 - [] 활용

| 코드 | 설명 |
|--|------------------|
| mydata.columns | 데이터가 보유한 변수명을 확인 |
| mydata.x1 mydata['x1'] | 변수열(columns)을 선택 |
| mydata.ix[1] mydata.ix['row2'] | 특정행(row)을 선택 |
| del mydata['x1'] | 변수를 삭제 |
| mydata.rename(columns= {'x1': 'new_name'}, inplace=True) | 변수명을 변경 |

[예제]

```
import pandas as pd

# x1 변수를 선택해 상위 행 보기
mydata['x1'].head()

# x1 변수의 두번째 행부터 4번째 행까지 보기
mydata[1:5]['x1']

# x1 변수명을 item1 변수명으로 변경
mydata.rename(columns={'x1': 'item1'}, inplace=True)
```



데이터 변환

| 내장함수 | 설명 |
|--------------------------|--|
| <code>int()</code> | base 진법의 수를 10진 정수형으로 변환 |
| <code>long()</code> | base 진법의 수를 10진 Long 정수형으로 변환 |
| <code>float()</code> | 수를 실수형으로 변환 |
| <code>complex()</code> | 복소수를 만들 |
| <code>str()</code> | 객체를 출력할 수 있는 문자열 그 자체로만 변환 |
| <code>repr()</code> | 객체를 출력 가능하고 eval함수의 입력으로 쓰일 수 있는 문자열로 반환 |
| <code>eval()</code> | 문자열로 된 파이썬 식을 실행 |
| <code>tuple()</code> | 튜플로 변환 |
| <code>list()</code> | 리스트로 변환 |
| <code>set()</code> | 리스트나 튜플 등을 세트로 변환 |
| <code>dict()</code> | 사전 객체를 생성 |
| <code>frozenset()</code> | 변경이 불가능한 세트로 변환 |
| <code>chr()</code> | 코드 값을 문자로 변환 |
| <code>unichr()</code> | 유니코드 값을 유니코드 문자로 변환 |
| <code>ord()</code> | 문자의 코드값을 구함 |
| <code>hex()</code> | 10진수에서 16진수로 변환 |
| <code>oct()</code> | 10진수에서 8진수로 변환 |



데이터 저장

- DataFrame형식으로 변환하여 pandas 패키지의 DataFrame.to_포맷(csv, excel, json) 함수를 사용해 파일로 저장

| 함수 | 설명 |
|-----------------------|--------------|
| <code>to_csv</code> | csv 파일로 저장 |
| <code>to_excel</code> | excel 파일로 저장 |
| <code>to_json</code> | json 파일로 저장 |

- 데이터 저장

[구조]

```
import pandas as pd
```

데이터명.to_csv('파일을 저장할
경로/파일명.csv', 옵션)

[예제]

```
import pandas as pd
```

```
mydata.to_csv('C:/Users/user/Desktop/  
example/mydata.csv',  
...: sep=';', # 값 구분자  
...: na_rep='NaN') # 결측값 표기 방법
```



데이터 탐색과 시각화

개요 / 히스토그램 / 기술통계 분석 / 상자도표(Box plot) / 상관관계 분석 / 산점도 (Scatter plot) / 기타

2022 미래차 중전인프라구축운영 전문인력 양성교육

개요

- 데이터 분석을 위해 히스토그램, 기술통계, 상자도표, 상관분석, 산점도 등을 적용하여 정보를 확인
- 데이터 탐색 과정은 데이터의 신뢰성, 필요한 데이터의 정제의 정도, 모형 구축 시 활용에 적합한 변수 탐색 등을 알 수 있게 하고 전처리 과정과 함께 매우 중요한 부분
- **히스토그램(Histogram)**
 - 표로 되어 있는 도수 분포를 시각적으로 나타냄
 - [가로축] 속성값(계급, 범주 등), [세로축] 빈도(frequency)
- **기술통계 분석**
 - [기술통계] 측정이나 실험에서 수집한 자료의 정리, 표현, 요약, 해석 등을 통해 자료의 특성을 규명하는 통계적 방법
 - 자료의 특성을 표현하는 지표로 대푯값(평균값, 중앙값, 최빈값), 산포도(분산, 표준편차, 범위, 사분위수, 평균편차, 표준오차, 변이계속) 왜도 및 첨도 있음



개요

• 상자 도표(box plot)

- 5개의 통계(최소값, 첫 번째 사분위수, 중앙값, 두 번째 사분위수, 최대값)을 시각적으로 나타냄
- 변수가 가진 값의 분포와 함께 이상치가 표시되어 있어 자료의 중심과 흩어진 정도를 살펴 볼 수 있음

• 상관관계 분석

- 두 개의 변수 간 어떤 선형적 관계를 가지고 있고, 그 관계의 강도는 어떠한지를 분석하는 방법
- 상관분석으로 도출되는 상관계수는 -1부터 1사이의 값을 가짐
- + 부호인 경우는 정적 상관관계(positive relationship, x가 커질수록 y도 커짐)를 나타냄
- -부호인 경우는 부적 상관관계(negative relationship, x가 커질수록 y도 커짐)를 나타냄
- 상관계수의 절대값이 1에 가까울 수록 관련성이 깊은 것



개요

• 산점도(Scatter plot)

- 주어진 데이터를 점으로 표현하여 시각적으로 나타냄
- 데이터의 실제값들의 분포를 파악하는데 유용



히스토그램(Histogram)

- 히스토그램 출력하기 위해 보통 **matplotlib** 패키지 이용
- matplotlib을 기반으로 하는 파이썬 시각화 라이브러리인 **seaborn**을 이용하여 가독성이 좋은 디자인 표/그래프를 출력
- matplotlib를 이용한 히스토그램

[구조]

```
import matplotlib.pyplot as plt

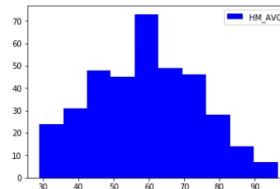
plt.hist(데이터명.변수명, 옵션)
plt.legend() # 변수명을 그래프에 표기
```

[예시]

```
import pandas as pd
import matplotlib.pyplot as plt
mydata = pd.read_csv('mydata.csv')

plt.hist(mydata.HM_AVG, color='b',
label='HM_AVG')
plt.legend()
```

[예시의 결과: 평균 상대습도(HM_AVG)]



히스토그램(Histogram)

- seaborn를 이용한 히스토그램

[구조]

```
import seaborn as sns
import matplotlib.pyplot as plt

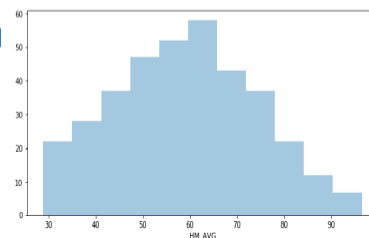
plt.figure(figsize=(가로 길이, 세로 길이))
sns.distplot(데이터명.변수명)
```

[예시]

```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10,5))
sns.distplot(mydata.HM_AVG, kde =
False) # kde: 가우시안 확률밀도 여부
```

[예시의 결과: 평균 상대습도(HM_AVG)]



기술통계 분석

- 데이터의 기본적인 통계량은 numpy, pandas, scipy 패키지를 이용해 출력
- numpy를 이용한 기술통계 분석

[구조]

* x는 array

```
import numpy as np

np.mean(x) # 평균
np.var(x) # 분산
np.str(x) # 표준편차
np.max(x) # 최대값
np.min(x) # 최소값
np.median(x) # 중앙값
np.percentile(x, 25) # 1사분위수
np.percentile(x, 50) # 2사분위수
np.percentile(x, 75) # 3사분위수

# x 에 데이터명 또는 변수명 입력
```

[예시]

```
import numpy as np

# 데이터에 있는 모든 변수의 결과
np.var(mydata)

# 데이터 특정 변수(x1)의 평균
np.mean(mydata['x1']) # 평균
```



기술통계 분석

- pandas를 이용한 기술통계 분석

[구조]

```
import pandas as pd

데이터명.describe()
```

[예시]

```
import pandas as pd

s = pd.DataFrame(mydata)
s.describe()
```

- SciPy를 이용한 기술통계 분석

[구조]

```
import scipy as sp

sp.stats.describe(데이터명)
```

[예시]

```
import scipy as sp

sp.stats.describe(mydata)
```



Box Plot

- matplotlib 패키지 중 pyplot 모듈 이용
- matplotlib를 이용한 Box plot

[구조]

```
import matplotlib.pyplot as plt

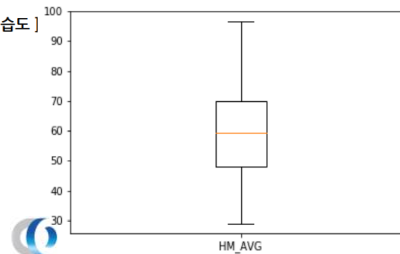
데이터명[['변수명']].plot(kind='box')
plt.xticks([1], ['변수명'])
```

[예시]

```
import pandas as pd
import matplotlib.pyplot as plt

mydata=pd.read_csv('mydata.csv')
#데이터 로드
plt.boxplot(mydata.HM_AVG)
plt.xticks([1], ['HM_AVG'])
```

[예시의 결과: 평균 상대습도]



상관관계 분석

- numpy패키지와 pandas패키지 활용
- numpy를 이용한 상관분석

[구조]

```
import numpy as np

np.corrcoef(변수명1, 변수명2)
```

[예시]

```
import numpy as np

a=np.array([1,2,3,4,5])
b=np.array([3,2,3,1,2])

np.corrcoef(a,b)
```

[예시의 결과]

```
array([[ 1.          , -0.56694671],
       [-0.56694671,  1.          ]])
```



상관관계 분석

• pandas를 이용한 상관분석

[구조]

```
import pandas as pd

데이터명.corr(method='pearson')
```

[예시]

```
import pandas as pd
mydata=pd.read_csv('mydata.csv')

X = mydata.iloc[:,1:4] # 두번째~
세번째 컬럼 선택

X.corr(method='pearson')
```

[예시의 결과]

| | CA_TOT | HM_AVG | RN_DAY |
|--------|----------|----------|----------|
| CA_TOT | 1.000000 | 0.665570 | 0.595266 |
| HM_AVG | 0.665570 | 1.000000 | 0.614174 |
| RN_DAY | 0.595266 | 0.614174 | 1.000000 |



Scatter plot

- matplotlib 패키지의 pyplot 모듈 활용
- matplotlib를 이용한 Scatter plot

[구조]

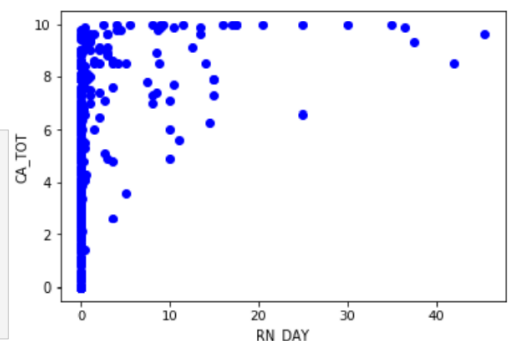
```
import matplotlib.pyplot as plt

plt.scatter('x축 변수명', 'y축 변수명', 옵션)
```

[예시]

```
import matplotlib.pyplot as plt

mydata=pd.read_csv('mydata.csv')
plt.scatter(mydata['RN_DAY'],
mydata['CA_TOT'],color='b', marker='o')
plt.xlabel('RN_DAY')
plt.ylabel('CA_TOT')
```



| 산점도 옵션 | 설명 |
|--------|------------------------------|
| label | 그래프 레이블을 입력 |
| size | 도형의 크기를 입력 |
| alpha | 도형 색상의 투명도를 입력(0: 투명, 1:불투명) |
| color | 그래프를 이용할 색상 입력 |
| marker | 표시할 도형 종류 입력 |

| | | | | | | | |
|-----|-----|-----|-----|-------|-----|-----|------|
| '.' | 'o' | '^' | 'x' | 'D' | 'p' | 's' | '+ |
| 점 | 원 | 삼각형 | 엑스형 | 다이아몬드 | 오각형 | 별 | 덧셈모양 |



기타

• 상위 또는 하위 행 선택해 출력하기

[구조]

```
import pandas as pd

mydata.head() # 상위 행 보기
mydata.tail() # 하위 행 보기
```

• 결측값 처리

[구조]

```
import pandas as pd

mydata.fillna(999) # 데이터에서 결측값은 999로 치환
mydata.dropna() # 데이터의 결측값 제외
mydata.isnull() # 데이터에서 결측인지
mydata.notnull() # 데이터에서 결측이 아닌지
```



기타

• 데이터 타입 변환

[구조]

```
# 데이터 타입 변환
데이터명[ " 변수명 " ]=데이터명[ " 변수명 " ].astype('변환할 타입')
```

[예시]

```
# target 변수를 string 타입으로 변환
mydata["target"]=mydata["target"].astype('category')

# 데이터 타입 보기
mydata.dtypes
```

• 선형, 누적, 바 그래프

[구조]

```
데이터명.변수명.plot.line() # 선형

데이터명.변수명.plot.bar() # 바

데이터명.변수명.plot.density() # 누적
```

[예시]

```
mydata.x1.plot.line()

mydata.x1.plot.bar()

mydata.x1.plot.density()
```



기타

• 행병합

[구조]

```
import pandas as pd

pd.concat([데이터명1, 데이터명2],axis=0)
```

[예시]

```
import pandas as pd

pd.concat([mydata1, mydata2],axis=0)
```

• 열병합

[구조]

```
import pandas as pd

pd.concat([데이터명1, 데이터명2],axis=1)
```

[예시]

```
import pandas as pd

pd.concat([mydata1,mydata2],axis=1)
```

• 데이터병합

[구조]

```
import pandas as pd

pd.merge(왼쪽 위치 데이터명, 오른쪽
위치 데이터명, on='고유키',how="left")
#how 옵션 종류: left, right, outer, inner
```

[예시]

```
import pandas as pd

pd.merge(mydata1, mydata2,
on='key',how="left")
```



기타

• 특정 조건에 따른 데이터 변환

[구조]

```
import numpy as np
import pandas as pd

데이터명["생성할 컬럼명"] =
np.where(데이터명[ ' 참조
컬럼명']==조건 값, 'True일 경우 값',
'False일 경우 값')
```

[예시]

```
import numpy as np

mydata.target.unique()
mydata["new"] =
np.where(mydata['target']==0.0, 'up',
'down')
mydata.head()
```

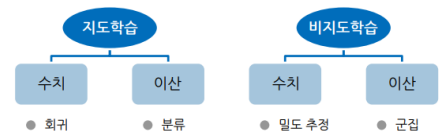


분석 알고리즘 맛보기

알고리즘 선정 / 예측: 단순선형회귀 / 분류: k-최근접 이웃

알고리즘 선정

2022 미래차 중전인프라구축운영 전문인력 양성교육



• 분석을 위한 기계학습 알고리즘

- 지도학습(Supervised learning): 회귀(Regression)과 분류(Classification)
- 비지도학습(Unsupervised learning)

• 분석 목적, 데이터 고려

- 목적값(Target value)을 예측하거나 예견하려고 한다면 지도학습방법 검토, 그렇지 않다면 비지도학습방법 검토
- [지도학습방법] 목적 값이 수치(Number)로 나타나길 원한다면 **회귀**를 검토, 'Yes/No', '그룹1/그룹2/그룹3'등과 같이 이산적인 값이라면 **분류**를 검토
- [비지도학습방법] 가지고 있는 데이터가 각각의 무리에 알맞은 어느 정도의 수치인지를 평가하려 한다면 **밀도 추정 알고리즘** 검토, 어떤 이산적인 무리에 알맞는지 알아보고자 한다면 **군집화 알고리즘**을 검토
- 보유한 데이터의 특성 파악 우선-속성이 명목형/연속형인지, Null값이 존재하는지, Outlier가 있는지 등에 따라서 적용 가능한 알고리즘의 폭을 줄일 수 있음



용어

- 전운량?
 - 대기 중 구름양을 측정해 0~10사이의 실수 값으로 표현된 수치
- 일조?
 - 하루 중 햇볕이 구름이나 안개 따위에 가려지지 안니하고 실제로 내리쬐는 시간을 말함



예측: 단순선형회귀

- 회귀분석-수치형 목적 값을 예측하기 위함
 - (장점)결과를 해석하기가 쉽고 계산 비용이 적음
 - (단점)비선형 데이터를 모델링하기에 적합하지 않음
- 회귀 방정식과 회귀 가중치
 - 전운량에 따른 일조를 예측할 때 독립변수인 전운량을 x_1 로, 종속변수인 일조를 y 로 하였을 때, 아래와 같은 방정식(회귀 방정식)으로 정의

$$y = \beta_0 + \beta_1 * x_1 + \varepsilon$$

$$\hat{y} = \beta_0 + \beta_1 * x_1$$

y : 종속변수 관측치 β_0 절편
 \hat{y} : 종속변수 추정치 β_1 회귀 가중치
 x : 독립변수 관측치
 ε : 오차

- (회귀분석)회귀 가중치를 찾는 과정, 일단 모수(절편, 회귀 가중치)를 찾게 되면 새로운 독립변수(x_1)가 주어졌을 때 종속변수 값(\hat{y})을 예측



예측: 단순선형회귀

• Python을 활용한 분석

- 청주지역에서 2015년에 일 단위로 측정된 ASOS 예제 데이터를 활용해 일평균전운량에 따른 하루의 일조 합계 시간을 예측
- 설치된 Spyder를 실행해 스크립트 편집 창에 아래의 코드를 붙여 넣고 전체 실행

```

### 패키지 불러오기
import pandas as pd
import statsmodels.formula.api as sm
import matplotlib.pyplot as plt

### 예제 데이터 로드
mydata = pd.read_csv('E:/data_regression.csv') # 데이터 위치 지정
print(mydata.head()) # 로드 데이터 확인

### 데이터 정제
mydata.loc[mydata['SS_DAY'] == -9] # 20150520 일조합 결측치(-9) 확인
mydata.loc[(mydata['SS_DAY'] != -9) & (mydata['CA_TOT'] == 0.5)] # 20150520을 제외한
일평균전운량이 0.5였던 날들 확인
mydata.loc[mydata['SS_DAY'] == -9, 'SS_DAY'] = 10.7 # 일조합 결측치에 평균값 10.7 입력

### 단순선형회귀 모델링
result = sm.ols(formula='SS_DAY ~ CA_TOT', data=mydata).fit()

### 회귀분석 결과 요약
print(result.summary())

### 세부 분석 결과 확인
print('< Parameters > \n', result.params) # 회귀계수 출력
print('< Prob (Parameters) > \n', result.pvalues) # 회귀계수에 대한 P-value 출력
print('< Adj. R-squared > \n', result.rsquared_adj) # 조정된 R-squared 출력
print('< Prob (F-statistic) > \n', result.f_pvalue) # 모형의 적합도 출력

### 그래프 그리기
fig, ax = plt.subplots(figsize=(8, 5))
plt.xlabel('SS_DAY', size=12)
plt.ylabel('CA_TOT', size=12)
ax.plot(mydata.CA_TOT.values, mydata.SS_DAY.values, 'o', label='Data')
ax.plot(mydata.CA_TOT.values, result.fittedvalues, 'b-', label='Regression')
ax.legend(loc='best')

```



예측: 단순선형회귀

• 분석결과

```

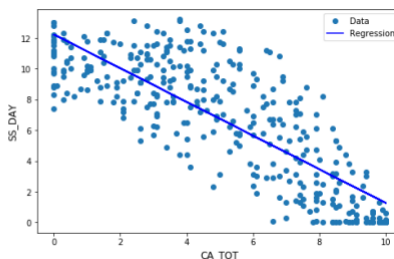
< Parameters >
Intercept    12.221711
CA_TOT       -1.095343
dtype: float64

< Prob (Parameters) >
Intercept    1.545615e-168
CA_TOT       2.877431e-90
dtype: float64

< Adj. R-squared >
0.672674117436

< Prob (F-statistic) >
2.87743055453e-90

```



• 분석결과 해석

- < Parameters >에는 회귀모형의 절편과 기울기가 제시되어 있습니다. 위 결과를 식으로 표현하면 다음과 같습니다.

$$\hat{Y} = 12.22 - 1.90X$$

\hat{Y} : 일조합(시간), X : 일평균전운량

- < Prob (Parameters) >에는 각 모수(절편, 기울기)의 통계적 유의확률이 제시되어 있으며, 일반적으로 0.05 이하일 경우, 모수가 유의미한 것으로 해석합니다.
- < Adj. R-squared >에는 회귀모형의 결정계수가 제시되어 있습니다. 결정계수는 모형의 설명력을 뜻하며, 위 모형의 설명력은 67.27%입니다.
- < Prob (F-statistic) >에는 모형의 F검정의 유의확률이 제시되어 있습니다. 이 유의확률 역시 0.05 이하일 경우 모형이 유의미한 것을 의미합니다.



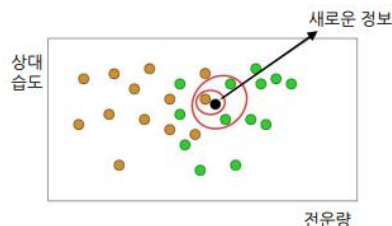
분류: k-최근접 이웃

- k-최근접 이웃(k-NN; k-nearest Neighbors) 분류기는 목적 대상(종속변수)의 범주를 활용 변수(독립변수)의 유사성에 기반해 범주를 지정하는 분류 알고리즘
 - (장점) 이상치(outlier)에 대해 둔감하고 데이터에 대한 가정이 없으며, 높은 정확도를 가짐
 - (단점) 계산 비용이 크고 많은 메모리 공간을 요구
- 전운량과 습도를 이용해 비(또는 눈)이 온 날 예측
 - 기상청에서 제공하는 종관기상관측장비(ASOS)데이터를 활용해 일평균전운량(CA_TOT)과 일평균상대습도(HM_AVG)를 이용해 비(혹은 눈)이 온 날(일 강수량(RN_DAT) > 0 mm)을 예측



분류: k-최근접 이웃

- k-최근접 이웃
 - 새로운 데이터가 주어졌을 때 학습한 데이터 가운데 가장 가까운 k개 이웃 정보를 이용하여 새로운 종속변수가 어떤 결과 값을 갖을지 예측
 - 선형 회귀분석과 같은 예측모형처럼 종속변수의 예측값을 추정하는 것이 아니라 분류하는 정보에 기반해 종속변수가 타겟에 속할 확률을 산출
 - 전운량과 상대습도에 따른 비 온 날을 분류하는 것을 예로 들어보면
 - 새로운 상대습도와 전운량 정보가 주어졌을 때, 기존의 분류 학습 데이터와 이웃한 1개의 정보만 반영하면 주황색으로 예측
 - 그러나 이웃한 3개의 정보를 반영하게 되면 주황색일 확률은 33.3%, 녹색일 확률은 66.6%



분류: k-최근접 이웃

• Python을 활용한 분석

- 서울지역에서 2015년에 일 단위로 측정된 ASOS 예제 데이터를 활용해 일평균전운량과 평균 상대습도에 따른 비(또는 눈)이 온 날을 예측
- 설치된 Spyder를 실행해 스크립트 편집창에 아래의 코드 전체 실행

```

### 패키지 불러오기
import numpy as np
from sklearn import neighbors # sklearn 은 scikit-learn 의 줄임말 입니다. 아니콘다에서 패키지 설치 시 scikit-learn 으로 검색합니다.
import pandas as pd
from matplotlib import colors as c
from sklearn.metrics import classification_report

### 예제 데이터 로드
mydata = pd.read_csv('E:/data_kNN.csv') # 데이터 위치에서 데이터 불러오기
print(mydata.head()) # 로드 데이터 확인

### 데이터 정제
mydata.loc[mydata['RN_DAY'] <= 0, 'RN_DAY'] = 0 # 강수량 0mm 이하(결측 포함)에 0 입력
mydata.loc[mydata['RN_DAY'] > 0, 'RN_DAY'] = 1 # 강수량 0mm 초과에 1 입력
mydata['RN_DAY'] = mydata['RN_DAY'].astype('category') # 강수량 변수 범주형 변환
mydata.loc[mydata['CA_TOT'] == -9, 'CA_TOT'] = 0 # 전운량 결측치(-9)에 0 입력
mydata.loc[mydata['HM_AVG'] == -9, 'HM_AVG'] = 0 # 상대습도 결측치(-9)에 0 입력

### 종속변수, 독립변수 설정
X = mydata.iloc[:,2:4] # 세번째 컬럼(CA_TOT)과 네번째 컬럼(HM_AVG)을 독립변수로 설정
y = data['RN_DAY'] # 이분형으로 변환한 강수량 변수를 종속변수로 설정

### k-최근접 이웃 모델링
kNN = neighbors.KNeighborsClassifier()
kNN.fit(X, y)

Z = kNN.predict(X) # 예측값 산출

### k-최근접 결과
target_names = ['0', '1']
print(classification_report(y, Z, target_names=target_names))

```



분류: k-최근접 이웃

• Python을 활용한 분석

- 서울지역에서 2015년에 일 단위로 측정된 ASOS 예제 데이터를 활용해 일평균전운량과 평균 상대습도에 따른 비(또는 눈)이 온 날을 예측
- 설치된 Spyder를 실행해 스크립트 편집창에 아래의 코드 전체 실행

```

# 도표 눈금(의사결정 경계를 얼마나 촘촘하게 할 것인지) 설정
plot_step = 0.02

# 경계 그리기
x_min, x_max = X['CA_TOT'].min() - 1, X['CA_TOT'].max() + 1
y_min, y_max = X['HM_AVG'].min() - 1, X['HM_AVG'].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, plot_step), np.arange(y_min, y_max, plot_step))
Z = kNN.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# kNN의 결과를 반영해 색상 입히기
pl.figure(1, figsize=(8, 6))
pl.set_cmap(pl.cm.Paired)
cMap = c.ListedColormap(['powderblue', 'ivory'])
pl.pcolormesh(xx, yy, Z, cmap=cMap)

# 독립변수 시각화
plt.scatter(X['CA_TOT'], X['HM_AVG'], c=y, cmap=plt.cm.Set3, edgecolor='k')
plt.xlabel('CA_TOT')
plt.ylabel('HM_AVG')

# 축이름 및 도표명
plt.xlim(xx.min(), xx.max())
plt.ylim(yy.min(), yy.max())
plt.xticks(())
plt.yticks(())
plt.title('Decision Boundary')

```

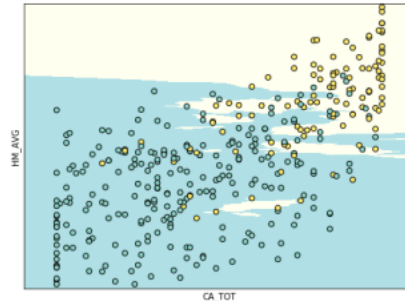


분류: k-최근접 이웃

• 분석 결과

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
                      metric_params=None, n_jobs=1, n_neighbors=5, p=2,
                      weights='uniform')
```

| | precision | recall | f1-score | support |
|-------------|-----------|--------|----------|---------|
| 0 | 0.96 | 0.99 | 0.97 | 254 |
| 1 | 0.97 | 0.90 | 0.93 | 111 |
| avg / total | 0.96 | 0.96 | 0.96 | 365 |



분류: k-최근접 이웃

• 분석결과 해석

- `kNN.fit(X, y)` 를 실행하면 모형의 옵션 구성 결과가 출력됩니다. 모형의 옵션은 값을 지정하여 고정할 수 있습니다. `n_neighbors` 옵션 예시는 아래와 같습니다(`k=3`일 때).

```
kNN = neighbors.KNeighborsClassifier(n_neighbors=3)
kNN.fit(X,y)
```

- 분석 결과표에서 정밀도(precision)란 타겟이 1일 경우, 모형에서 예측한 발생(1) 사건 중 실제 발생(1) 사건의 비율을 뜻합니다. 위 모형의 평균 정밀도는 96%입니다.
- 분석 결과표에서 재현율(recall)이란 타겟이 1일 경우, 실제 발생(1) 사건 중 모형이 예측한 발생(1) 사건의 비율을 뜻합니다. 위 모형의 평균 재현율은 96%입니다.
- 분석 결과표에서 f1 점수는 정밀도와 재현율 두 지수를 종합해 계산한 결과로 그 식은 아래와 같습니다.

$$F1 = 2 \times \frac{\text{정밀도} \times \text{재현율}}{\text{정밀도} + \text{재현율}}$$

f1 점수는 1에 가까울 수록 좋으며 위의 f1 점수는 0.96입니다.

- 미래의 사건을 예측하는 모형을 구축하기 위해선 독립변수와 종속변수 간 시간 간격을 조정, 변수 추가 등 여러 데이터 분석 방법을 적용 할 수 있습니다.



참고

- 기상기후 빅데이터 분석 플랫폼 - 날씨마루
 - <https://bd.kma.go.kr/kma2020/dta/reqst/dtaReqstInfo.do?menuCd=F030102000&pageNum=4>
- 최소자승법
 - <https://bkshin.tistory.com/entry/DATA-17-Regression>

