Data Analysis

1차시
데이터분석 개론

20 . .

지은이: 이은재

Ⅰ. 데이터와 데이터 분석이란 무엇인가?

- **dare (주어진 것, 주다) → Datum / Data (주어진 것, 사실, 자료)**

일반적인 의미: 관찰, 측정 또는 실험을 통해 수집된 사실이나 수치. 해석과 분석을 통해 지식이나 통찰로 전환될 수 있는 정보의 원재료가 된다.

컴퓨터 과학에서의 의미: 컴퓨터가 처리할 수 있는 형태로 저장된 기호, 신호, 텍스트, 숫자 등의 집합.

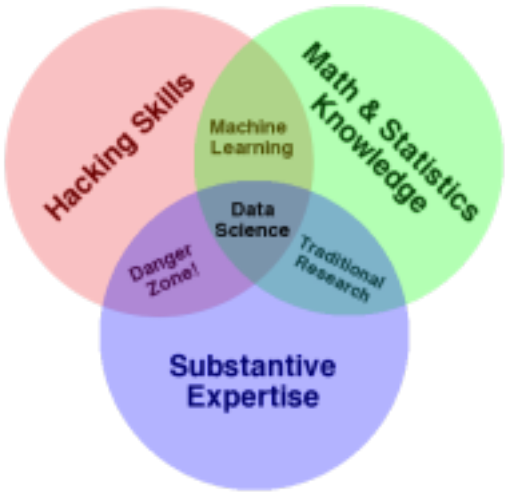
→ 데이터는 해석, 분석, 가공해야 의미있는 정보가 된다!

데이터 분석과 데이터 과학

1. 데이터 과학

- **데이터를 수집, 정제, 분석, 활용하고 그 모든 과정을 자동화하는 데에 필요한 수학적 원리와 기술을 연구하는 학문 분과**

- **일반적으로 통계학, 데이터 분석학, 소프트웨어 공학, 데이터 마이닝, 데이터 시각화, 머신러닝 등 다양한 학문 영역을 융합하고 있음**



드류 콘웨이의 데이터 과학 벤다이어그램

2. 데이터 분석

- **유용한 정보를 발견하고 결론을 유추하거나, 의사 결정을 돕기 위해 데이터를 조사, 정제, 변환, 모델링하는 모든 과정**

- **데이터 수집, 데이터 처리, 데이터 정제, 모델링**

- **데이터 분석의 주요 분야**

분야	설명
기술통계	관측이나 실험을 통해 수집한 데이터를 정량화하거나 요약하는 기법
탐색적 데이터 분석	데이터를 시각적으로 표현하여 주요 특징을 찾고 분석하는 방법
가설검정	주어진 데이터를 기반으로 특정 가정이 합당한지 평가하는 방법

→ 데이터 분석가는 프로그래밍, 수학, 통계학, 도메인 지식을 모두 갖추어야 한다!

Q. 도메인 지식이란? 나는 어떤 도메인 지식을 가지고 있을까?

3. 데이터 분석을 위한 도구

- 프로그래밍 언어

항목	Python	R	MATLAB
주요 용도	범용 프로그래밍, 데이터 과학, AI, 웹 개발	통계 분석, 데이터 시각화, 머신러닝	수치 해석, 공학적 모델링, 시뮬레이션
언어 특성	동적 타이핑, 객체 지향, 스크립트 언어	통계적 계산에 최적화된 스크립트 언어	행렬 연산 및 수치 계산 중심의 프로그래밍 언어
강점	- 대규모 커뮤니티 - 다양한 라이브러리 - 범용성 뛰어남 - 웹 개발 연동 용이 (Django, Flask)	- 통계 분석 및 시각화에 최적화 - 데이터 과학자와 연구자에게 인기 - 강력한 시각화 기능 (ggplot2)	- 과학적 계산에 최적화 - 강력한 수치해석 기능 - Simulink와 통합 가능 - 공학적 모델링에 적합
단점	- 실행 속도 상대적으로 느림 - 복잡한 수치 계산 시 최적화 필요	- 범용성 부족 - 대규모 시스템 개발에 부적합 - 속도 느림 (Python보다 느릴 수 있음)	- 상용 소프트웨어 (유료) - 오픈 소스 생태계 부족 - 문법의 유연성 부족
학습 난이도	쉬움 (가독성 높은 문법)	쉬움 (데이터 분석 관점에서 직관적)	중간 (수학 및 공학적 지식 요구)
속도	보통 (C/C++로 확장 시 빠름)	느림 (대용량 데이터 처리 시 속도 저하)	빠름 (수치 계산 및 행렬 연산에서 우수)
생태계	매우 활발 (다양한 오픈 소스 프로젝트)	분석과 연구 중심의 커뮤니티 존재	공학 및 학술 커뮤니티에서 강세
시각화 도구	Matplotlib, Seaborn, Plotly	ggplot2, Shiny, plotly	기본 제공 플로팅 함수, Simulink
확장성	웹, 모바일, IoT 등 다양한 분야 확장 가능	데이터 분석과 머신러닝에 국한되는 경향	주로 공학적 애플리케이션에 한정
라이선스	오픈 소스 (무료)	오픈 소스 (무료)	상용 라이선스 (유료)
통합성	C/C++, Java, R과 통합 용이	Python, C++과 통합 가능	Python, C/C++와 연동 가능

- 파이썬 필수 패키지

패키지	주요 기능	특징	사용 목적	의존성
NumPy	- 다차원 배열 객체(ndarray) 제공 - 고성능 수학 연산 지원	- 과학 계산을 위한 핵심 패키지 - C 기반으로 빠른 연산 지원 - 배열 연산의 표준	- 수치 연산 - 배열 처리 - 선형대수	대부분의 과학 관련 Python 패키지에 의존
Pandas	- 데이터프레임(DataFrame) 및 시리즈(Series) 지원 - 데이터 조작 및 분석 도구	- CSV, Excel 등 다양한 데이터 파일 읽기/쓰기 - 결측치 처리, 필터링, 그룹화 지원 - SQL과 유사한 데이터 처리	- 데이터 분석 - 데이터 정제 - 시계열 데이터 처리	NumPy에 의존
Matplotlib	- 2D 및 3D 시각화 지원 - 다양한 그래프 생성	- MATLAB과 유사한 시각화 인터페이스 - 커스터마이징 가능한 플롯 - 다양한 출력 형식 지원(PNG, PDF 등)	- 데이터 시각화 - 그래프 및 플롯 생성	NumPy에 의존
SciPy	- 고급 과학 및 수학 계산 도구 - 최적화, 통계, 신호 처리 등	- NumPy 위에 구축된 과학 계산 확장 라이브러리 - 선형 대수, 미분방정식, FFT 등 고급 기능 포함	- 과학적 계산 - 최적화 문제 해결 - 신호 및 이미지 처리	NumPy에 의존
Scikit-learn	- 머신러닝 알고리즘 구현 - 분류, 회귀, 클러스터링 등 지원	- 일관된 API와 문서화된 사용법 제공 - 모델 평가, 하이퍼파라미터 튜닝 지원 - 간단한 통합 파이프라인 구축 가능	- 머신러닝 모델링 - 데이터 마이닝 - 예측 분석	NumPy, SciPy, Matplotlib에 의존

- 데이터베이스

- DBMS (MySQL, Oracle, MS ...)
- SQL

- 프로그래밍 환경

- Google Colab
- R Studio
- SPSS

Ⅱ. 첫 번째 실습

1. Google Colab 소개

: https://colab.google/

- 설치 필요 없음
웹 기반이기 때문에 따로 개발 환경을 구축할 필요 없이 구글 계정만 있으면 바로 사용할 수 있다.

- 무료 GPU & TPU 지원
머신러닝이나 딥러닝 모델 학습할 때 GPU나 TPU를 무료로 사용할 수 있다.
(런타임→ 런타임 유형 변경→ GPU/TPU 선택)

- Jupyter Notebook과 동일한 인터페이스
.ipynb파일 형식을 사용해서 Jupyter Notebook과 거의 동일한 사용 경험을 제공한다.

- Google Drive와 통합
Colab 노트북 파일을 Google Drive에 저장하고 언제든지 불러와 작업할 수 있다.

- 라이브러리 설치 간편
!pip install명령어로 필요한 파이썬 라이브러리를 즉시 설치할 수 있고, 주요 라이브러리는 이미 설치되어 있다.

- 협업 기능
Google Docs처럼 실시간으로 다른 사람들과 노트북을 공유하고 공동 작업할 수 있다.

2. 데이터 불러오기

Iris Species Dataset

a) 직접 다운로드한 데이터 불러오기

- Kaggle (https://www.kaggle.com/datasets/uciml/iris)

```
from google.colab import drive
drive.mount('/content/drive')
# 드라이브 경로의 CSV 파일 읽기 예제
df = pd.read_csv('/content/drive/MyDrive/path_to_your_file/iris.csv')
print(df.head())
```

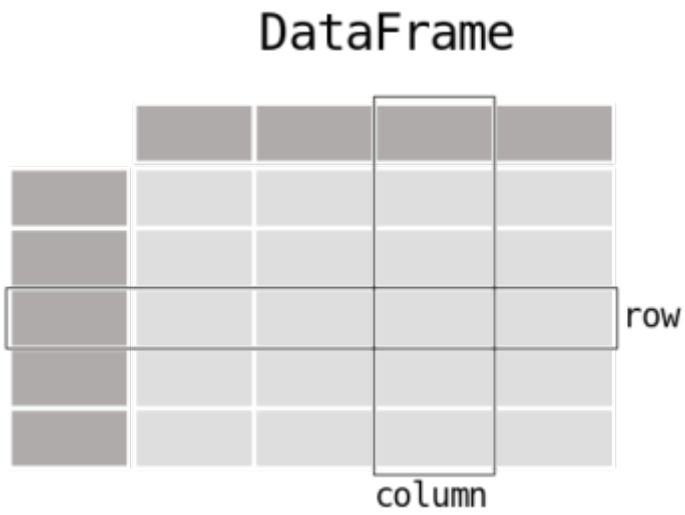
b) scikit-learn 내장 데이터 불러오기

```
# scikit-learn 라이브러리에서 iris 데이터셋을 불러오는 함수 load_iris를 임포트합니다.
from sklearn.datasets import load_iris

# pandas 라이브러리를 pd라는 이름으로 임포트합니다. (데이터 프레임 처리에 사용됩니다.)
import pandas as pd

# load_iris 함수를 사용하여 iris 데이터셋을 불러와 변수 iris에 저장합니다.
# 이 데이터셋에는 꽃받침 길이/너비, 꽃잎 길이/너비 등의 특성과 클래스(품종) 정보가 포함되어 있습니다.
iris = load_iris()

# pandas의 DataFrame 객체로 iris 데이터셋을 변환합니다.
# data 매개변수에는 iris 데이터 배열을, columns 매개변수에는 각 특성의 이름 리스트를 전달합니다.
df = pd.DataFrame(data=iris.data, columns=iris.feature_names)

# target(숫자형 클래스 라벨)을 사람이 읽을 수 있는 품종 이름으로 변환합니다.
# pd.Categorical.from_codes를 사용하여 target 배열의 숫자를 target_names(품종 이름)와 매핑합니다.
df['species'] = pd.Categorical.from_codes(iris.target, iris.target_names)

# DataFrame의 상위 5개 행을 출력하여 데이터가 제대로 로드되었는지 확인합니다.
print(df.head())
```

3. Dataframe 다루기

Q. Dataframe?

:행(row)과 열(column)로 구성된 2차원 표 형태의 데이터 구조



Q. CSV? JSON?

- CSV

: Comma-Separated Values의 약자로, 쉼표(,)를 구분자로 데이터를 저장하는 파일 형식

```
Name,Age,City
Alice,25,Seoul
Bob,30,Busan
Charlie,35,Incheon
```

- JSON

: JavaScript Object Notation의 약자로, 키-값 쌍으로 구성된 데이터 구조

```
[
  {"Name": "Alice", "Age": 25, "City": "Seoul"},
  {"Name": "Bob", "Age": 30, "City": "Busan"},
  {"Name": "Charlie", "Age": 35, "City": "Incheon"}
]
```

함수/기능 이름	기능	설명
`info()`	데이터 요약 정보 확인	데이터프레임의 구조, 데이터 타입, 결측치 여부 등을 출력합니다.
`describe()`	통계 요약	수치형 컬럼의 평균, 표준편차, 최소/최대값 등의 통계 정보를 제공합니다.
`shape`	크기 반환	데이터프레임의 (행, 열)형태를 반환합니다.
`columns.tolist()`	컬럼명 리스트화	데이터프레임의 컬럼명을 리스트 형태로 반환합니다.
`df['column']`	특정 컬럼 선택	선택한 컬럼의 데이터를 반환합니다.
`df[['col1', 'col2']]`	여러 컬럼 선택	선택한 여러 컬럼의 데이터를 반환합니다.
`loc[]`	라벨 기반 선택	행과 열을 라벨(이름)로 선택합니다. 슬라이싱 시 마지막 포함됩니다.
`iloc[]`	정수 기반 선택	행과 열을 정수 인덱스로 선택합니다. 슬라이싱 시 마지막 제외됩니다.
`df[조건]`	조건 기반 필터링	조건에 해당하는 행만 반환합니다.
`sort_values()`	데이터 정렬	특정 컬럼을 기준으로 데이터프레임을 정렬합니다.
`ascending`	오름차순/내림차순 선택	sort_values()에서 오름차순(True), 내림차순(False) 여부 지정합니다.
`df['new_col'] = ...`	새로운 컬럼 추가	계산 등을 통해 새로운 컬럼을 추가합니다.
`groupby()`	그룹화	특정 컬럼을 기준으로 데이터를 그룹화합니다.
`mean()`	평균 계산	그룹화된 데이터의 평균값을 계산합니다.
`value_counts()`	고유값 개수 세기	특정 컬럼 내 각 고유값의 개수를 반환합니다.
`isnull().sum()`	결측치 개수 확인	각 컬럼별 결측치(NaN) 개수를 계산합니다.
`reset_index()`	인덱스 초기화	인덱스를 기본값으로 초기화합니다. (drop=True사용 시 기존 인덱스 삭제)

함수/기능 이름	기능	설명
to_csv()	CSV 파일로 저장	데이터프레임을 CSV 파일로 저장합니다.
read_csv()	CSV 파일 불러오기	CSV 파일을 데이터프레임으로 읽어옵니다.
unique()	고유값 추출	특정 컬럼의 고유한 값을 배열로 반환합니다.
matplotlib.pyplot.figure()	플롯 생성	플롯의 크기와 구조를 정의합니다.
matplotlib.pyplot.hist()	히스토그램 생성	데이터 분포를 나타내는 히스토그램을 생성합니다.
legend()	범례 표시	플롯에 범례를 추가합니다.
title(), xlabel(), ylabel()	그래프 제목 및 축 레이블 설정	플롯의 제목과 x, y축 레이블을 지정합니다.
show()	그래프 출력	생성된 플롯을 화면에 표시합니다.

4. Dataframe 파일로 내보내기

: Colab 환경에서는 왼쪽 파일트리에서 다운로드 할 수 있다.

```
# CSV 파일로 저장 및 다시 불러오기
csv_file ='iris_dataset.csv'
df.to_csv(csv_file, index=False)
print(f"\n DataFrame이 '{csv_file}' 파일로 저장되었습니다.")

df_loaded = pd.read_csv(csv_file)
print("\n 저장된 CSV 파일을 다시 불러온 결과:")
print(df_loaded.head())
```