Data_Science

데이터프레임

- 2차원 배열의 원소로 구성된 하나의 스프레드시트를 의미, 표와 같음
- 엑셀에서의 시트와 같은 의미

인덱스

- 데이터 원소들을 효율적으로 그룹화하고 참조가 용이하도록 제공하는 구분자
- 원소의 값을 구분하기 위한 고유의 식별 번호
- 자료를 효율적으로 관리할 수 있도록 제공

행데이터 선택

- `loc[1,3]` : 검색범위 마지막도 포함 1, 2, 3
- `iloc[1,3]` : 검색범위 마지막 미포함 1, 2

시리즈

- 시리즈 = 값 + 인덱스
- ⇒ 하나의 행 + 복수의 열

데이터 전처리

- 파일의 기본 구성요소인 레코드를 기반으로 필드를 조작하는 것
- 데이터를 체계적으로 수집하는 과정
- 데이터를 효율적으로 수집하기 위해서는 데이터베이스 관리 시스템 DBMS 사용하는 것이 좋음

데이터 발생 → 수집 → DBMS → 읽기 → 데이터프레임 → 데이터 분석

레코드

- 행

필드

- 열

to_excel() 메소드

- openpyxl 라이브러리가 설치되어 있어야 함

json 파일

- 데이터를 공유할 목적으로 개발된 특수한 형태의 파일

파일 경로명 지정

- 절대경로명, 상대경로명

행 인덱스 reset_index

- 행 인덱스를 초기화 하게 되면 행인덱스는 열로 이동하게 됨

기술통계정보

- count
- unique
- top : 최빈값
- freq : 빈도수
- mean
- min

value_count()

- 이 메서드를 사용하게 되면 고유값이 행인덱스가 되고 고유값의 개수가 데이터의 값이 되는 **시리즈** 객체가 생성

---

std()

df[['col1', 'col2']].corr()

데이터 정규화

- 중복이 최소화되도록 데이터를 구조화하는 프로세스를 의미

- 목적
  - 하나의 데이터프레임에서 데이터의 삽입, 삭제, 변경이 정의된 관계들로 인해 데이터 프레임의 나머지 부분들로 전파되게 하는 것
- 목표
  - 이상이 있는 관계를 재구성하여 작고 잘 조직된 관계를 생성하는 것

데이터 정규화 필요성 !!!!!!

- 데이터 프레임에 존재하는 데이터의 중복성 최소화
- 데이터의 무결성 강화
- 하나의 데이터 프레임을 둘 이상으로 분리하는 효율성 증대

데이터 분석

- 수집된 데이터를 기반으로 패턴을 추출하고 그 결과를 분석해 가치가 있는지 판단하는 프로세스
- 수집된 데이터는 결함이 존재하면 신뢰도가 떨어지므로 정규화는 필요함

데이터 분석의 정확도

- 데이터 분석의 정확도에 대한 신뢰성 확보는 수집된 데이터의 품질에 따라 됨

NaN

- 유효한 값이 존재하지 않거나 누락된 데이터
- Not a Number

데이터 시각화

5가지 주요 방법

- 시간 시각화 : 막대그래프, 누적 막대 그래프, 점 그래프
- 관계 시각화 : 히스토그램, 버블차트, 박스 플롯
- 비교 시각화 : 히트맵, 스타 차트, 평행 좌표계, 다차원 척도법
- 분포 시각화 : 파이 차트, 도넛 차트, 트리맵, 누적 연속 그래프
- 공간 시각화 : 지도 맵핑

7단계 활용 절차

1. 데이터 수집
2. 데이터 저장
3. 데이터 처리
4. 데이터 분석
5. 시각화
6. 활용
7. 폐기

박스 그래프

- 최댓값
- 3분위값
- 중간값
- 1분위값
- 최솟값

플로팅 함수

- 선 스타일 지정
- 마커 기호 지정
- 색상 지정

내용	함수 및 속성	비고
데이터프레임 생성	`pd.DataFrame()`	pandas 패키지 필요
행 인덱스 변경	`df.index = '바꿀idx명'`	
열 이름 변경	`df.columns = ‘바꿀col명’`	
선택적 행 인덱스|열 이름 변경	`df.rename(index = {’기존인덱스’ : ‘새로운 인덱스’)`	
열 데이터 선택1 ([])	`df[’col’]`	
열 데이터 선택2 (.)	`df.col`	

행 추가	`df.loc['추가하려는 새로운 행인덱스'] = 값`	
열 추가	`df['새로운 열'] = 값`	
열→인덱스	`df.set_index(['col'], inplace=True)`	
행 | 열 삭제	`df.drop('행|열'. axis=0|1)`	
Excel파일로 저장	`df.to_excel('저장할 파일명.xlsx')`	openpyxl 라이브러리 설치 필요
df 2개를 Excel로 저장	`변수명3 = pd.ExcelWriter('저장할 파일명'.xlsx)`	워크북을 생성
	`df1.to_excel(변수명3, sheet_name = '시트명1'`	df1을 시트로
	`df2.to_excel(변수명3, sheet_name = '시트명2'`	df2을 시트로
	`변수명3.save()`	
csv로 저장	`df.to_csv('파일명.csv')`	
json으로 저장	`df.to_json('파일명.json')`	
csv / json 파일 읽어오기	`df.read_csv('파일명.csv', encoding='cp949')`	
행 인덱스 재배열	`df.reindex(인덱스 배열 변수)`	인덱스를 원하는 것으로 재배열
행 인덱스 초기화	`df.reset_index()`	
데이터프레임 정렬	`df.sort_index(ascending=False)`	ascending : 오름차순
df 기본정보 확인	`df.info()`	dtype확인
자료형 정보 확인	`df.dtypes`	속성이라는 점 유의
기술통계정보 확인	`describe()`	모든 정보를 보려면 `include=all`
각 열의 고유값 개수 확인	`df.value_counts()`	`dropna=False` 를 사용하면 누락값(NaN)도 볼 수 있다.
평균값	`df.mean()`	
중간값	`df.median()`	
표준편차	`df.std()`	
상관계수	`df[['col1', 'col2']].corr()`	매우 중요
누락데이터 치환	`df['col'].fillna(변수명, inplace=True)`	`method='파라미터'` ffill, bfill

	`df.interpolate()`	양쪽 행의 중간값으로 치환
누락데이터 제거(열)	`df.dropna(axis=1, thresh=개수)`	개수만큼 존재하지 않은 열 전체 삭제
누락데이터 제거(행)	`df.dropna(axis=0, how='all')`	전체가 NaN으로 존재한다면 해당 행 삭제
특정 열에 결측치 존재 시 행 삭제	`df.dropna(subset='col_name')`	
중복 데이터 확인	`df.duplicated(['col'])`	boolean값으로 반환
중복 데이터 삭제	`df.drop_duplicated(['col'])`	
그래프 시각화(기본)	df.plot()	kind옵션이 있는데 default = 'line'
	kind옵션	barh : 수평막대
		his : 히스토그램
		kde : 커널 밀도 그래프
		area : 면적 그래프
		hexbin : 고밀도 산점도 그래프
선 스타일 지정자	'-' : 실선	
	'—' : 파선	
	':' : 점선	
	'-.' : 일점 쇄선	
선 굵기 지정자	linewidth	
마커테두리 색 지정	markeredgecolor	
마커 면 색 지정	markerfacecolor	
마커 크기 지정	markersize	
선형 회귀 시각화	`sns.lmplot(x='col1', y='col2', data = df, hue = 'col3')`	col1과 col2로 선형 회귀 그래프를 그리고 hue = col3로 col3에 해당하는 데이터를 색상으로 구분
서브 플롯 시각화	`sns.catplot(data=df, kind='옵션', col = '강조할 열이름', col_wrap = 개수)`	