



데이터 시각화 1

인공지능 RPA – 5주차

학습 내용

1. 데이터 시각화
2. FastAPI 기본



인공지능 이론 및 실습

_데이터의 처리 과정



1. 데이터 정제 (Data Cleaning) : 불필요하거나 일치하지 않는 데이터를 제거
2. 데이터 통합 (Data Integration) : 다수의 데이터 소스들을 결합
3. 데이터 선택 (Data Selection) : 필요한 데이터들을 데이터 저장소로부터 검색
4. 데이터 변환 (Data Transformation) : 데이터 마이닝/모델링을 하기에 적합한 형태로 데이터 가공
5. 데이터 마이닝 / 모델링 (Data Mining, Modeling) : 지능적 방법들을 적용하여 지식(데이터 패턴, 관계 등) 추출
6. 데이터 검증 (Data Evaluation) : 찾아낸 지식(데이터 패턴, 관계 등)를 검증
7. 데이터 시각화 (Data Presentation) : 발견한 지식을 사용자에게 효과적으로 보여주기 위해 시각화

_데이터 시각화

1. matplotlib 라이브러리 불러오기

- `pip install matplotlib`
- `import matplotlib.pyplot as plt`

2. 데이터 불러오기

- `pd.read_csv('ch4-1.csv')`

3. 차트(그래프) 함수 사용

- Bar 차트
`plt.bar()`
- 상자 차트
`plt.boxplot()`



_데이터 읽기 : CSV 파일

CSV 파일읽기

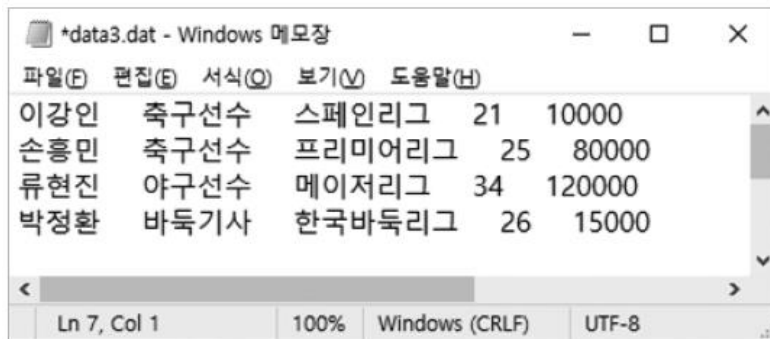
- CSV 파일을 읽을 때는 read_csv() 함수를 사용

```
import pandas as pd
```

```
ns_df = pd.read_csv('ch4-1.csv', encoding='utf-8')
```

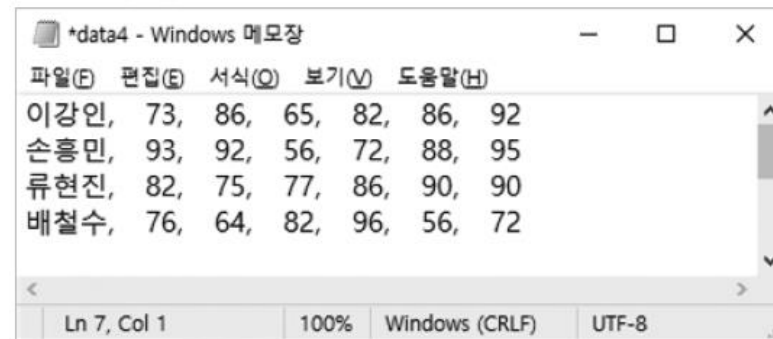
CSV(Comma-Separated Values) : 쉼표(.)로 구분된 텍스트 데이터

- 한 줄이 하나의 레코드(record)이며 레코드는 콤마로 구분된 여러 필드(field)로 구성



이강인	축구선수	스페인리그	21	10000
손흥민	축구선수	프리미어리그	25	80000
류현진	야구선수	메이저리그	34	120000
박정환	바둑기사	한국바둑리그	26	15000

(a) 공백으로 구분



이강인,	73,	86,	65,	82,	86,	92
손흥민,	93,	92,	56,	72,	88,	95
류현진,	82,	75,	77,	86,	90,	90
배철수,	76,	64,	82,	96,	56,	72

(b) 콤마로 구분(CSV 파일)



_데이터 시각화

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

```
def addtext(x,y):
    for i in range(len(x)):
        plt.text(i,y[i]+0.5,y[i], ha = 'center')
```

```
hat = pd.read_csv('ch4-1.csv') # hat 변수에 데이터셋 입력
print(hat, end="\n\n")
print(hat.head(), end="\n\n") # 위에서 부터 5개 데이터 확인
```

```
font_path = "malgun.ttf"
font_name = font_manager.FontProperties(fname=font_path).get_name()
plt.rc('font', family=font_name)
```



_데이터 시각화 : Bar 차트 그리기

```
plt.figure(figsize=(15, 10))  
plt.bar(hat['hatchery'], hat['chick'], color =  
('red','orange','yellow','green','blue','navy','purple'))
```

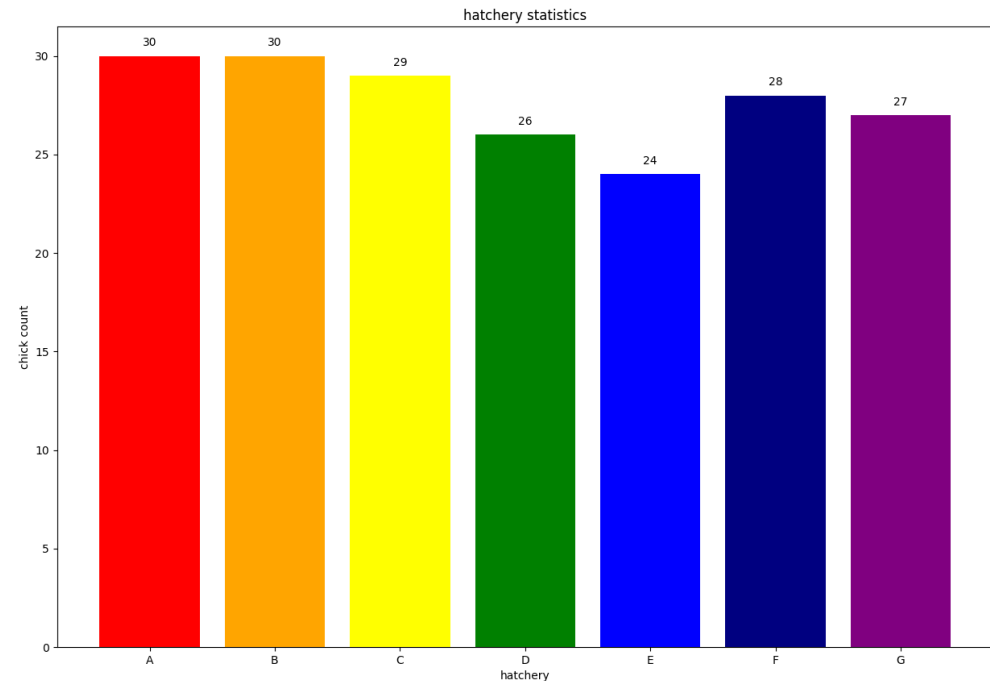
```
plt.title('hatchery statistics')
```

```
plt.xlabel('hatchery')
```

```
plt.ylabel('chick count')
```

```
addtext(hat['hatchery'], hat['chick'])
```

```
plt.show()
```



_데이터 시각화 : Pie 차트 그리기

파이차트를 그리기 위해 비율 계산

```
pct = hat['chick']/hat['chick'].sum()
```

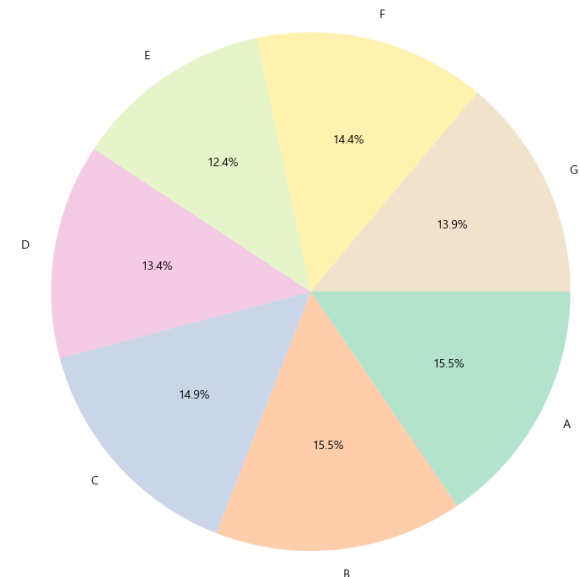
```
col7 = sns.color_palette('Pastel2', 7)
```

파이차트 그리기

```
plt.figure(figsize=(10, 10))
```

```
plt.pie(pct, labels = hat['hatchery'], autopct='%0.1f%%', colors=col7, counterclock = False)
```

```
plt.show()
```



_데이터 시각화 : Line 차트 그리기

라인 차트 그리기

```
plt.figure(figsize=(10, 7))
```

```
plt.plot(hat.hatchery, hat.chick, marker='*', color='y', linestyle='--', linewidth=4)
```

```
plt.title('부화장별 병아리 부화현황')
```

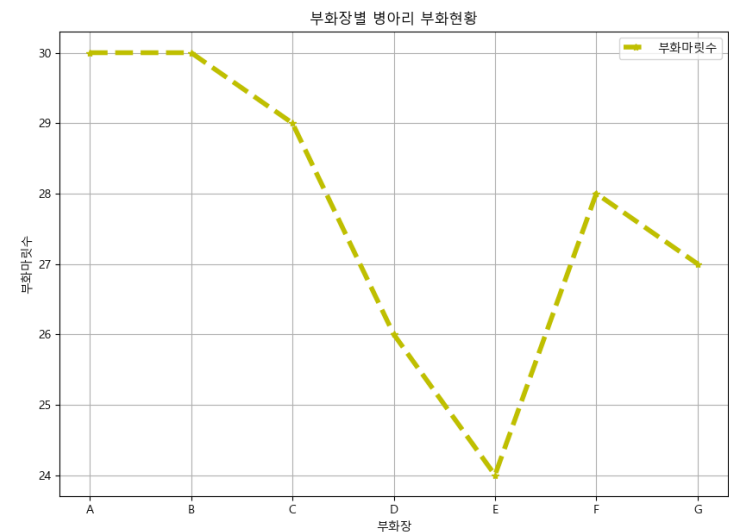
```
plt.xlabel('부화장')
```

```
plt.ylabel('부화마릿수')
```

```
plt.grid(True)
```

```
plt.legend(['부화마릿수'], fontsize=10, loc='best')
```

```
plt.show()
```



_데이터 시각화

(a) linestyle

종류	지정 방법
실선(Solid)	'—'
대쉬선(Dashed)	'--'
점선(Dotted)	'.'
대쉬점선(Dash-dot)	'-.'

(b) marker

종류	지정 방법
o	'o'
x	'x'
+	'+'
사각형	's'
오각형	'p'
마름모	'd'
영문자	'\$영문자\$'

RPA 실습

FastAPI

Rest API를 만들기 위한 Python 웹 프레임워크

<https://fastapi.tiangolo.com/ko/>



FastAPI 프레임워크, 고성능, 간편한 학습, 빠른 코드 작성, 준비된 프로덕션

Test **passing** coverage **100%** pypi package **v0.111.1**



_FastAPI

1. Fast API 설치

- 라이브러리 설치 (처음 1회)

```
pip install fastapi
```

```
pip install "uvicorn"
```

- 모듈 import (파일 마다)

```
from fastapi import FastAPI
```

2. 코드 입력

```
fapi1.py
```

3. 실행

```
python -m uvicorn fapi1:app --reload
```

4. 브라우저에서 확인

```
http://127.0.0.1:8000
```

```
from fastapi import FastAPI
```

```
app = FastAPI()
```

```
1 @app.get("/")
```

```
2 def read_root():  
    return {"Hello": "World"}
```



_데이터 형식 : JSON과 CSV

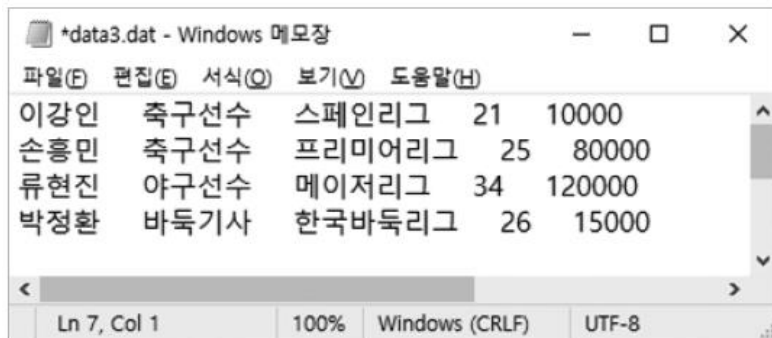
JSON(JavaScript Object Notation) : 'key : value' 쌍으로 구성된 텍스트 데이터

→ 모든 키는 큰 따옴표로 감싸야 한다

```
{  
  "name": "혼자 공부하는 데이터 분석",  
  "author": ["박해선", "홍길동"],  
  "year": 2022  
}
```

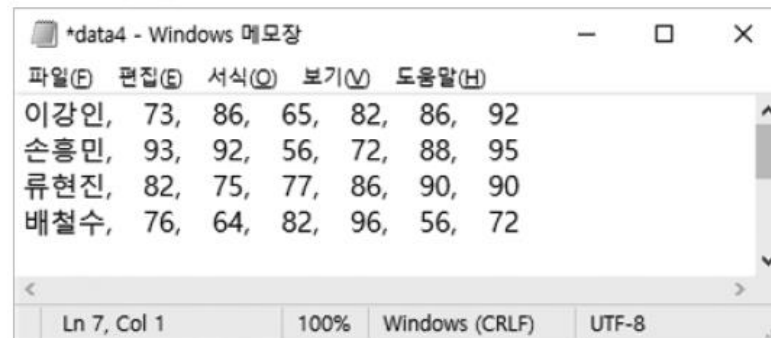
CSV(Comma-Separated Values) : 쉼표(,)로 구분된 텍스트 데이터

- 한 줄이 하나의 레코드(record)이며 레코드는 콤마로 구분된 여러 필드(field)로 구성



이강인	축구선수	스페인리그	21	10000
손흥민	축구선수	프리미어리그	25	80000
류현진	야구선수	메이저리그	34	120000
박정환	바둑기사	한국바둑리그	26	15000

(a) 공백으로 구분



이강인	73	86	65	82	86	92
손흥민	93	92	56	72	88	95
류현진	82	75	77	86	90	90
배철수	76	64	82	96	56	72

(b) 콤마로 구분(CSV 파일)



_FastAPI

파라미터 받기

http://127.0.0.1:8000/item?item_id=20000&name=홍길동&age=20

- 함수의 인자를 통해서 파라미터를 받음 (타입 힌트를 이용)

이름 : 형식 = 디폴트값

```
from fastapi import FastAPI
```

```
app = FastAPI()
```

```
@app.get("/")
```

```
def read_root():
```

```
    return {"Hello": "World"}
```

```
@app.get("/item")
```

```
async def read_item(item_id: int, name: str = None, age: int = 0):
```

```
    return {"item_id": item_id, "name": name, "age": age}
```

int (정수)

float (실수)

bool (불리언)

str (문자열)

None (null, void)

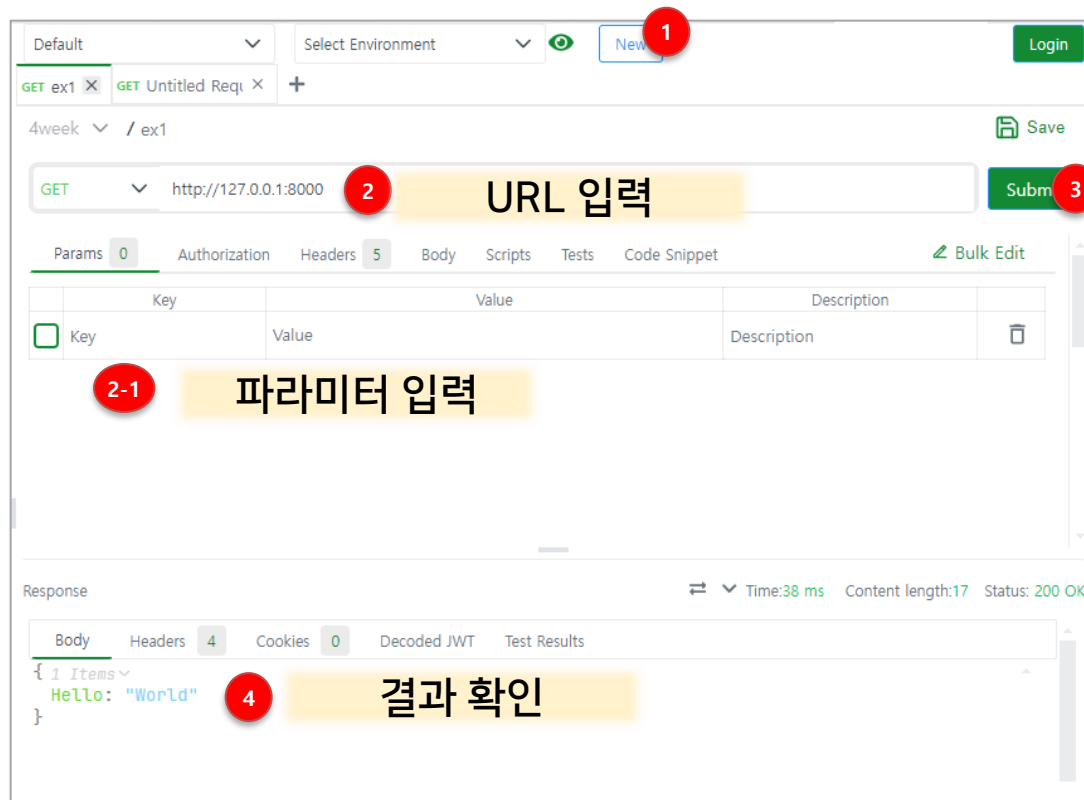
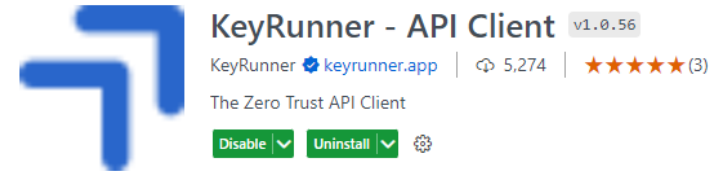
List (리스트)

Dict (딕셔너리)



_FastAPI

VS Code Extension에서 확인 KeyRunner - API Client 설치



호출 - 실행

파라미터 입력

결과 확인



_FastAPI

실습 : 이름, 학번, 학과 정보를 보내면 "___학과 ___님 (___) 반갑습니다" 를 출력하는 RestAPI 추가
현재 실습 예제에서 이어서 코딩

- API : /user (method는 GET)
- 파라미터 : name, studentcode, major
- 출력 : { "msg" : "의료정보과 홍길동님(202223) 반갑습니다" }

The screenshot shows a REST client interface with a GET request to `http://127.0.0.1:8000/info?major=의료정보과&name=홍길동&studentcode=202223`. The request is saved and submitted. The response is a JSON object: `{ "msg": "의료정보과 홍길동님 (202223) 반갑습니다." }`.

GET ex2 × +

4week ▾ / ex2 Save

GET ▾ `http://127.0.0.1:8000/info?major=의료정보과&name=홍길동&studentcode=202223` Submit

Params 3 Authorization Headers 5 Body Scripts Tests Code Snippet Bulk Edit

	Key	Value	Description	
<input type="checkbox"/>	name	홍길동	Description	
<input type="checkbox"/>	major	의료정보과	Description	
<input type="checkbox"/>	studentcode	202223	Description	
<input type="checkbox"/>	Key	Value	Description	

Response ↔ ▾ Time:12 ms Content length:64 Status: 200 OK

Body Headers 4 Cookies 0 Decoded JWT Test Results

```
{ 1 Items ▾  
  msg: "의료정보과 홍길동님 (202223) 반갑습니다."  
}
```

