SEEDS SEASON2

Seeds
FIRST STEP
-4 week-

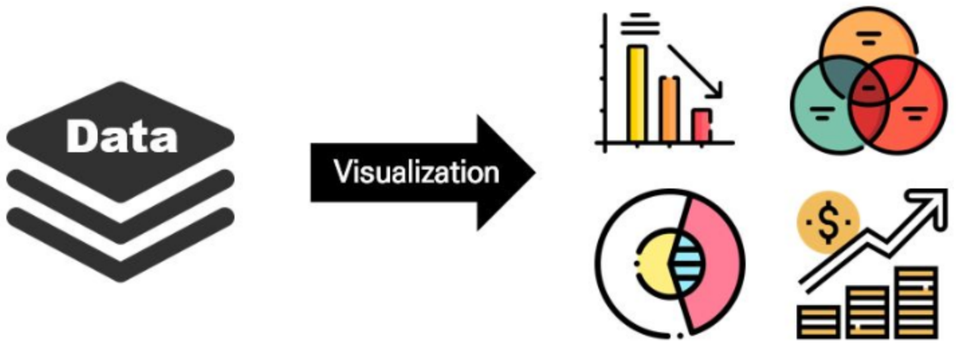
Seeds운영진 **김민성**

INDEX

01 시각화란?

- 시각화(Visualization)는 데이터나 정보를 시각적인 형태로 표현하는 과정 또는 결과물을 말한다.

- 시각화를 통해 숫자와 텍스트로 표현되는 정보를 그래프,차트,도표, 등의 시각적 요소로 변환하고, 이를 통해 데이터 패턴, 관계, 추세 등을 쉽게 파악할 수 있다.



01 시각화를 하는 목적

- 데이터의 이해: 우리가 보통 처음 받아보는 데이터는 엑셀과 같은 Table형식의 데이터이다. 이런 데이터는 한눈에 파악하기 어려운 특징을 가지고 있다. 따라서 이런 복잡한 데이터를 시각적으로 표현해서 데이터의 특성과 구조를 빠르게 파악하기위해 시각화를 진행한다.

- 판단과 의사 결정: 데이터의 핵심 내용을 이해하고 앞으로 어떻게 분석을 진행할지 계획을 세울 수도 있다. 데이터를 시각적으로 표현하면 추후에 어떻게 모델링을 진행할 지 등과 같은 다양한 의사결정을 내릴 때 더 정확하고 효율적으로 판단할 수 있다.

- 인사이트 도출 및 보고 : 모델링과 검증이 끝나고 나서 결과를 해석할때 사용하기도 한다. 또한 나중에 클라이언트들에게 보고할 때나 의견을 표현할 때 시각화된 자료를 활용하여 자신의 의견과 결과를 효과적으로 전달할 때도 사용할 수 있다.

02 파이썬 주요 시각화 라이브러리

- Matplotlib
  - 매력적인 그래프, plot , 히스토그램 및 막대 차트를 만드는데 주로 사용되는 패키지
  - Scipy , Numpy 및 Pandas의 그래프 데이터를 지원한다.
  - 다른 종류의 그래프 도구에 대한 사전 지식이 있다면 Matplotlib이 가장 자연스러운 선택이 될 수 있음.

02 파이썬 주요 시각화 라이브러리

- 주요 특징:
  - 선 그래프, 막대 그래프 , 히스토그램과 같은 다양한 그래픽 표현을 표시할 수 있다.
  - Numpy 배열 및 SciPy 스택과 호환된다.
  - 여기에는 패턴을 식별하고 관계를 만드는데 도움이 되는 다양한 차트가 포함되어 있다,

- 장점:
  - 상호작용을 위한 플랫폼
  - 적응형 라이브러리

- 단점:
  - 사용법이 다소 복잡하고, 기본적인 시각화 결과가 세련되지 않을 수 있음.
  - 코드의 길이가 길어질 수 있고, 복잡한 시각화를 구현하기 어렵다.

02 파이썬 주요 시각화 라이브러리

- Ploty
  - Python 그래프 모듈을 사용하면 대화형 고품질 그래프를 간단하게 생성할 수 있다.
  - 선 그래프, 산점도 , 영역차트 , 막대 차트 및 Matplotlib 및 Seaborn과 유사한 기타 차트 스타일이 포함된다.
  - Interactive 한 시각화 가능하여 사용자가 시각화된 그래프를 쉽게 줌인, 줌아웃 및 툴팁을 활용한 데이터확인이 가능합니다.

02 파이썬 주요 시각화 라이브러리

- 주요 특징:
  - 포괄적인 API는 로컬 및 웹 브라우저 모드에서도 잘 작동한다.
  - 높은 수준의 대화형 오픈 소스 시각화 라이브러리다.
  - Jupyter 노트북, 독립 실행형 HTML 파일 , 심지어 온라인에서도 볼 수 있다.

- 장점:
  - 등고선도 , 치수 차트 , 덴드로그램을 사용할 수 있다.
  - 40개의 개별 차트 및 Plot 유형을 허용한다.

- 단점:
  - 활용하기가 어렵다.

02 파이썬 주요 시각화 라이브러리

- Seaborn
  - Seaborn은 matplotlib를 기반으로 하는 Python 데이터 시각화 라이브러리이다. 매력적이고 유익한 통계 그래픽을 그리기 위한 고급 인터페이스를 제공한다.

02 파이썬 주요 시각화 라이브러리

- 주요 특징:
  - 정보 시각화를 생성하는데 필요한 매핑 및 집계를 수행한다.
  - 사용자가 데이터를 더 잘 탐색하고 이해할 수 있도록 돕기 위해 Pandas의 dataframe과 잘 통합되어 있으므로 데이터 분석결과를 직접 시각화하기 쉽다.
  - 아름답고 유익한 대수 이미지 생성을 위한 높은 수준의 융합을 제공한다.

- 장점:
  - 훨씬 더 매력적인 시각적 묘사.
  - 다른 데이터 형식으로 변경이 가능.

- 단점:
  - 커스터마이징이 제한되어 있다.

02 파이썬 주요 시각화 라이브러리

- ggplot
  - ggplot은 R의 유명한 시각화 라이브러리인 ggplot2의 Pyhton 버전으로, 데이터 시각화를 위한 강력하고 지능적인 도구이다.

02 파이썬 주요 시각화 라이브러리

- 주요 특징:
  - 향상된 표현을 통해 훨씬 더 유익한 시각화를 생성할 수 있다.
  - Pandas와 함께 사용하여 dataframe에 데이터를 저장할 수 있다.
  - R 프로그래밍 언어로 작성된 시각화라이브러리인 ggplot2를 기반으로 작동한다.

- 장점:
  - 문서는 간단하고 이해하기 쉽다.
  - 그래프의 구성 요소를 명확하게 이해하고 표현할 수 있다.

- 단점:
  - ggplot이 제공하는 기본 설정 밖에서 매우 특수하거나 복잡한 시각화를 만드는 것이 어렵다.

02 파이썬 주요 시각화 라이브러리

- Altair
  - Altair는 Vega-Lite 기반 Python용 선언적 통계 시각화 패키지이다. 복잡한 통계 수정이 필요한 차트를 구성하는데 탁월한 라이브러리이다.

- Vega-Lite란?
  - JSON 형식으로 데이터 시각화를 선언적으로 정의하도록 설계되었으며, 이를 통해 사용자는 간결하게 복잡한 시각화를 만들 수 있다.

02 파이썬 주요 시각화 라이브러리

- 주요 특징:
  - Vega-Lite JSON 표준을 기반으로 사용하기 쉽고 일관된 API를 제공한다.
  - 소스 코드는 GitHub에서 사용할 수 있다.
  - Python 3.6이상 지원 , jsonschema, Numpy, Pandas 및 Toolz가 모두 필요하다.

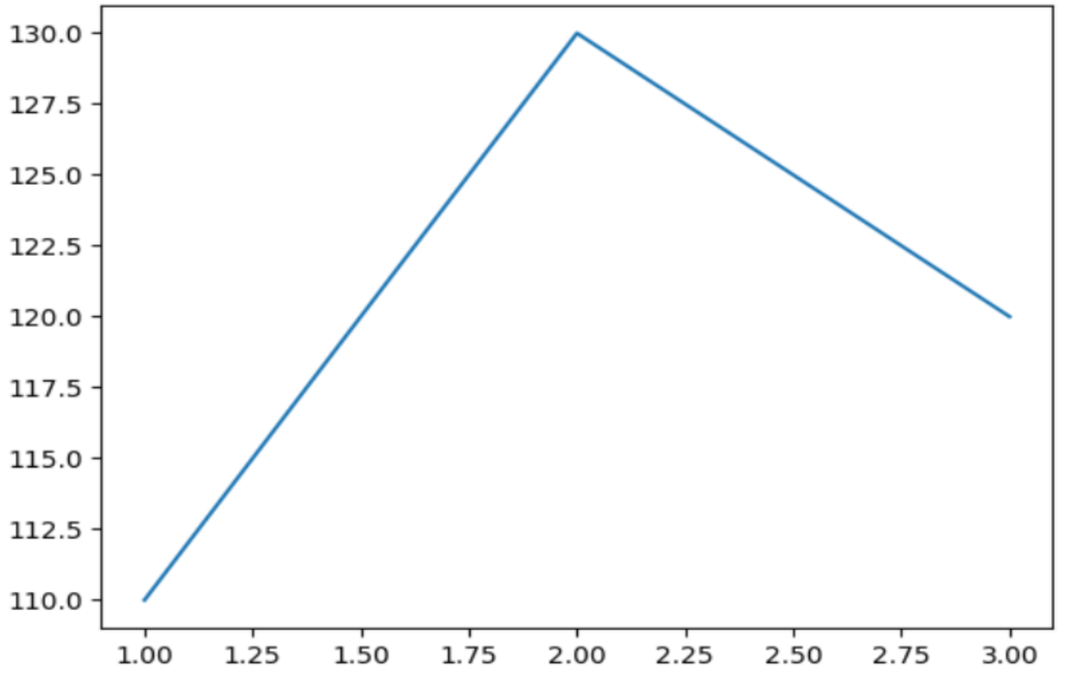
- 장점:
  - 최소한의 코드로 최고의 비주얼을 만들 수 있다.
  - JSON 형식으로 출력할 수 있으므로, 웹에서 쉽게 사용할 수 있다.

- 단점:
  - 기본적으로 웹 브라우저에서 동작하기 때문에, 대규모 데이터 셋에 대한 시각화를 생성하면 웹 브라우저가 느려질 수 있다.
  - 선언적 접근법을 사용하기 때문에, 사용자 정의 옵션이 명령형 라이브러리(Matplotlib)에 비해 제한적일 수 있다.

03 matplotlib 사용

- matplotlib 사용

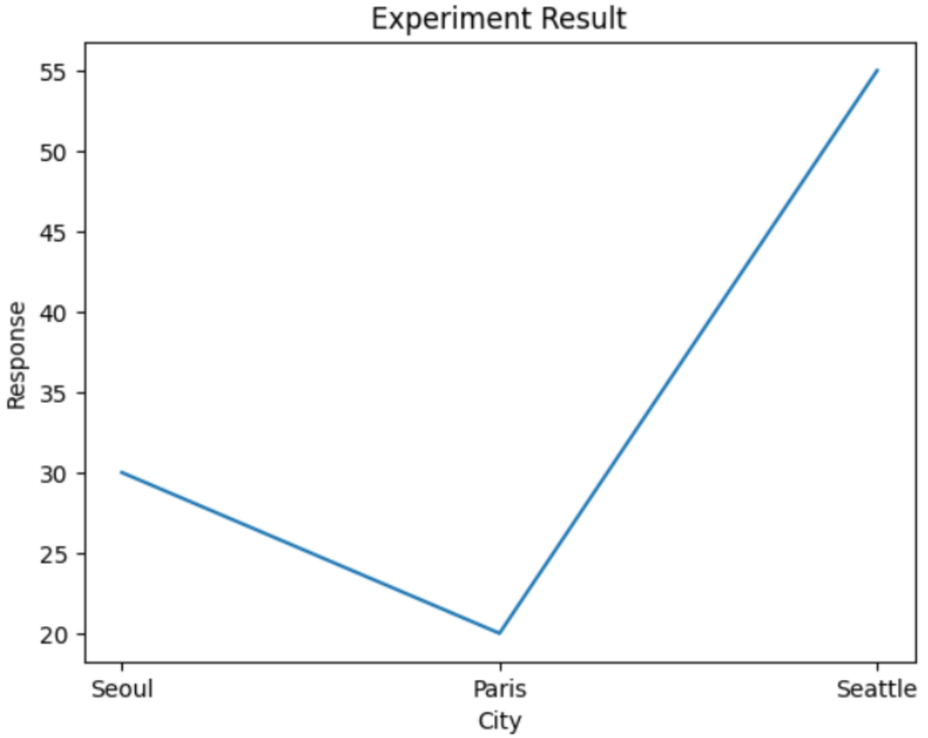
```
import matplotlib.pyplot as plt   <- matplotlib 라이브러리를 불러옴

plt.plot([1,2,3] , [110,130,120])  #선 그래프를 그림
plt.show() # 그래프 출력
```



03 matplotlib 사용

- **제목과 축 레이블 설정**

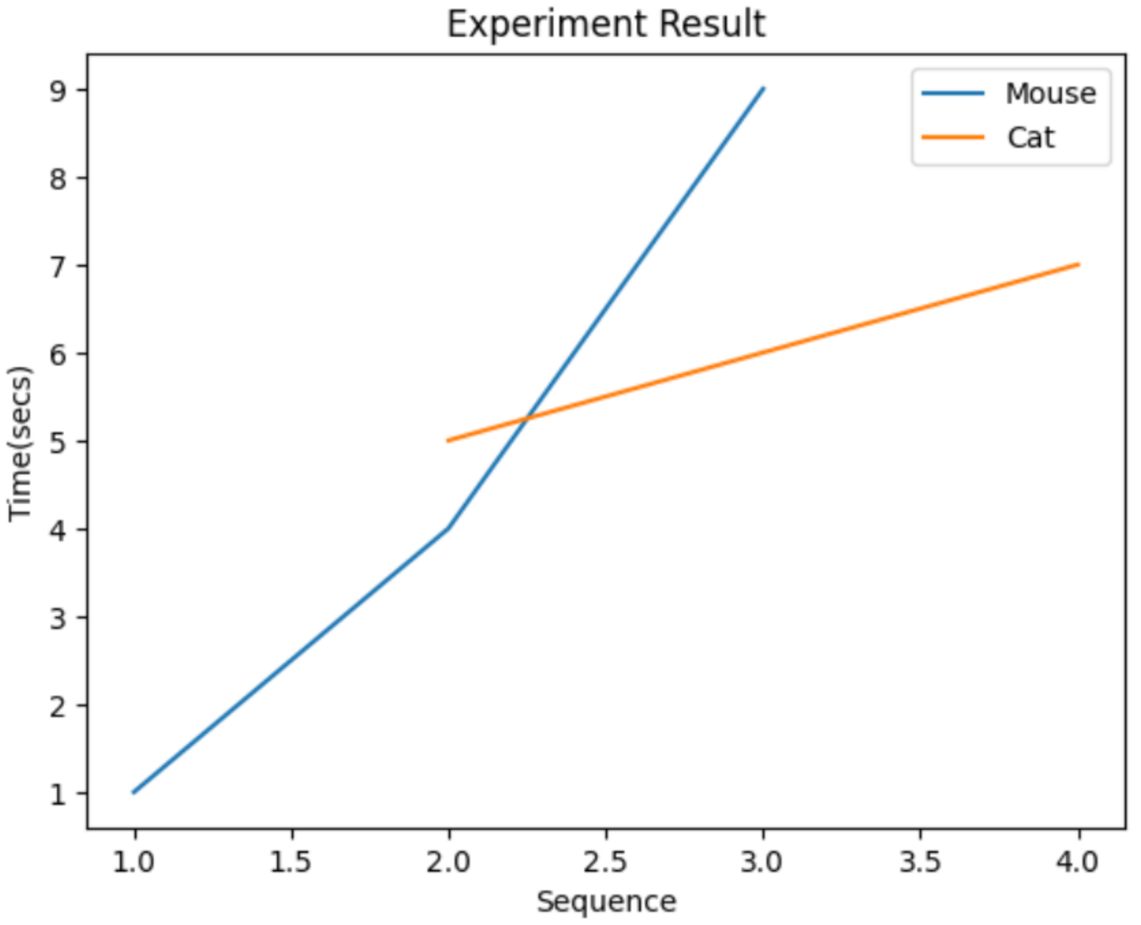
```
plt.plot(["Seoul" , "Paris" , "Seattle"] , [30,20,55]) #선 그래프를 그림
plt.xlabel("City") #x축의 레이블을 설정
plt.ylabel("Response") #y축의 레이블을 설정
plt.title("Experiment Result") #그래프의 제목을 설정
plt.show()
```



03 matplotlib 사용

- 범례 추가

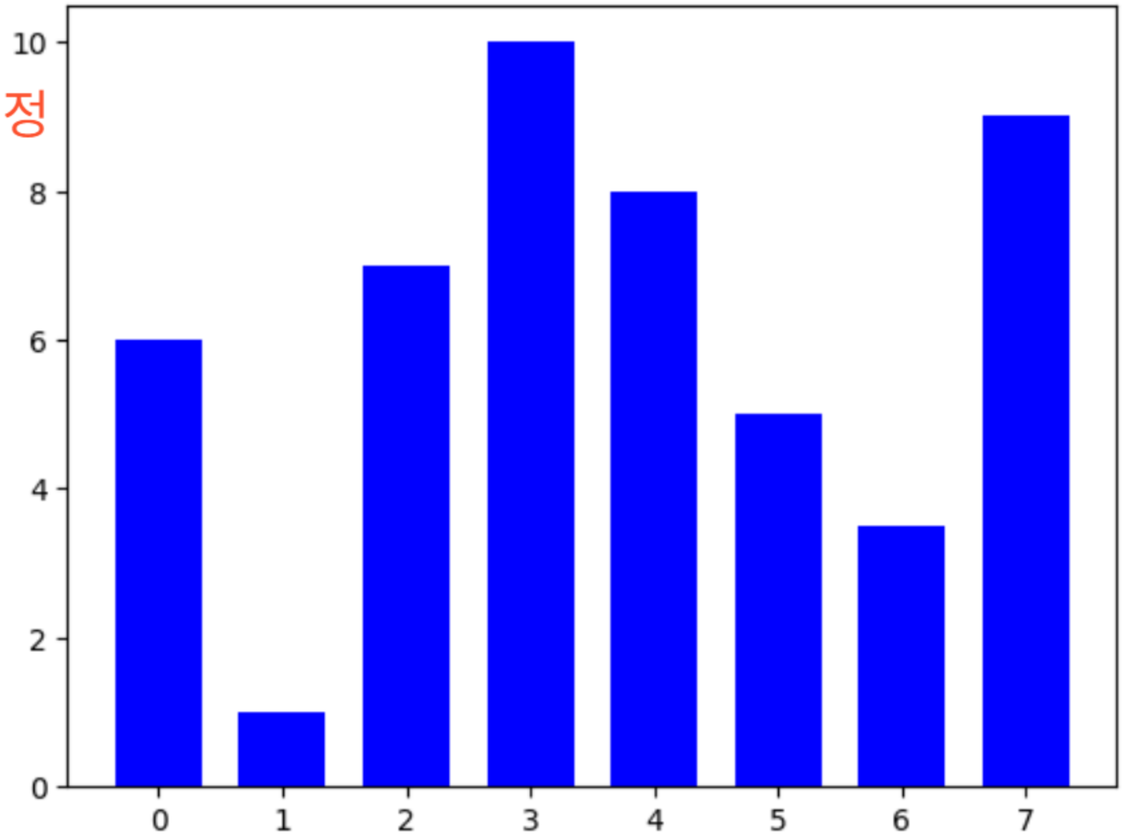
```
plt.plot([1,2,3], [1,4,9])
plt.plot([2,3,4],[5,6,7])
plt.xlabel('Sequence')
plt.ylabel('Time(secs)')
plt.title('Experiment Result')
plt.legend(['Mouse', 'Cat'])   #범례 표시
plt.show()
```



03 matplotlib 사용

- 다양한 차트 및 플롯

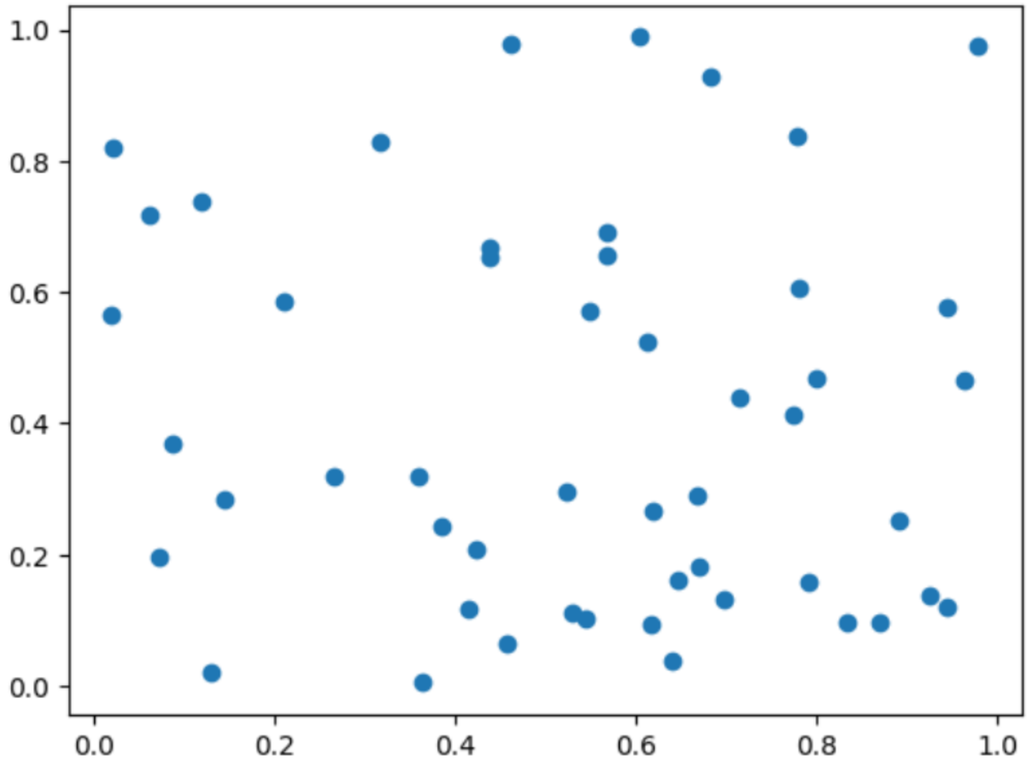
```
y = [5, 3, 7, 10, 9, 5, 3.5, 8]
x = range(len(y))
plt.bar(x, y, width=0.7, color="blue")   # 바 차트 설정
plt.show()
```



03 matplotlib 사용

- 산점도 표기 예제

```
import numpy as np      # numpy는 수치 계산을 위한 라이브러리
np.random.seed(0)       #시드 설정
n = 50
x = np.random.rand(n)   # 난수 생성
y = np.random.rand(n)
plt.scatter(x, y)
plt.show()
```



과제

Matplotlib Tutorial - 파이썬으로 데이터 시각화하기
https://wikidocs.net/141537