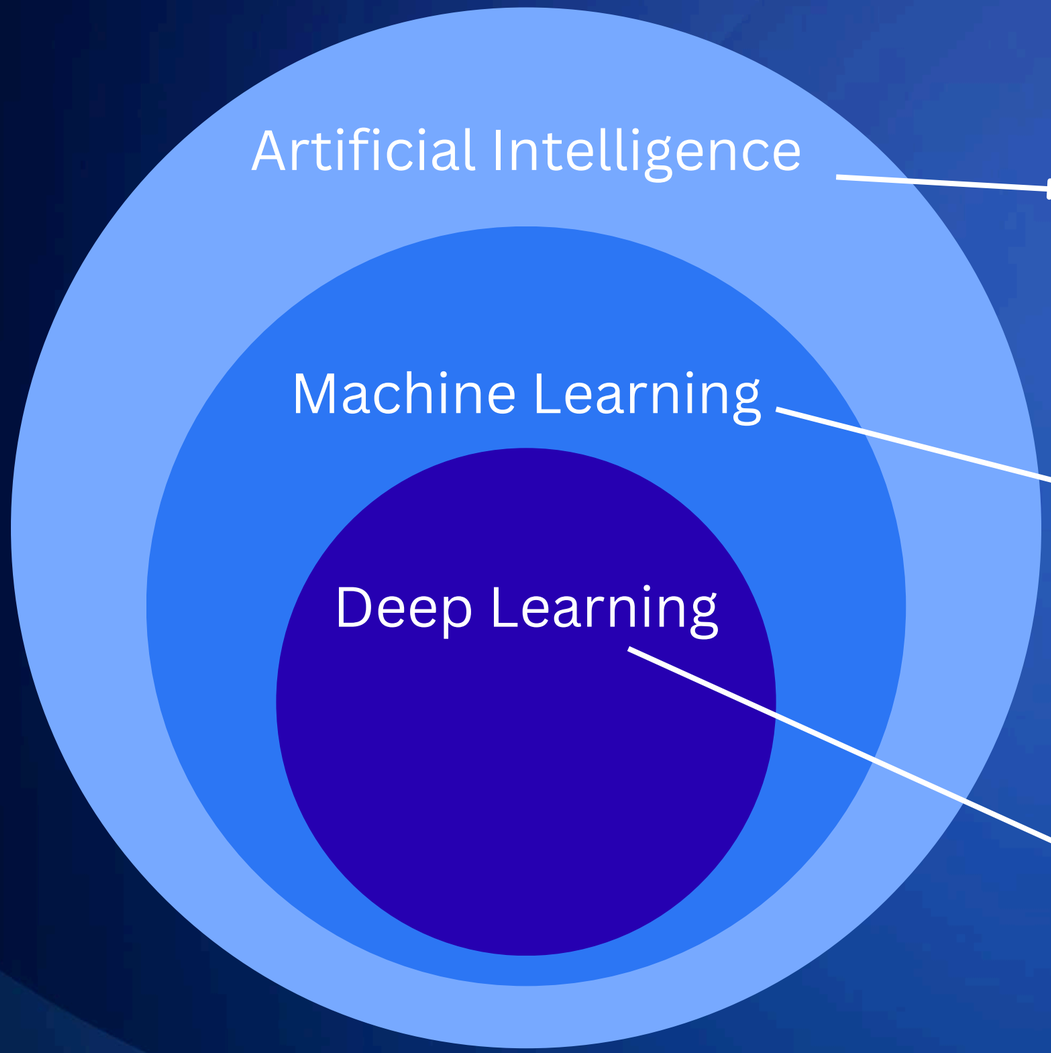
EDA & DATA-PREPROCESSING

Kuggle 10-th 서현규 & 김태현

CONTENT

1. 인공지능이란?



사람이 해야 할 일을 기계가 대신할 수 있는
모든 자동화

명시적으로 규칙을 프로그래밍하지 않고
데이터로부터 의사결정을 위한 패턴을
기계가 스스로 학습

인공신경망 기반의 모델로, 비정형 데이터로부터
특징 추출 및 판단까지 기계가 한 번에 수행

KEY WORDS

1. 데이터
2. 모델
3. 학습
4. 모델 파라미터
5. 손실함수
6. 최적화

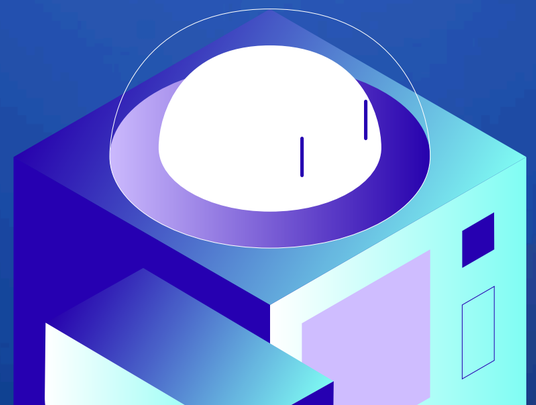
Example

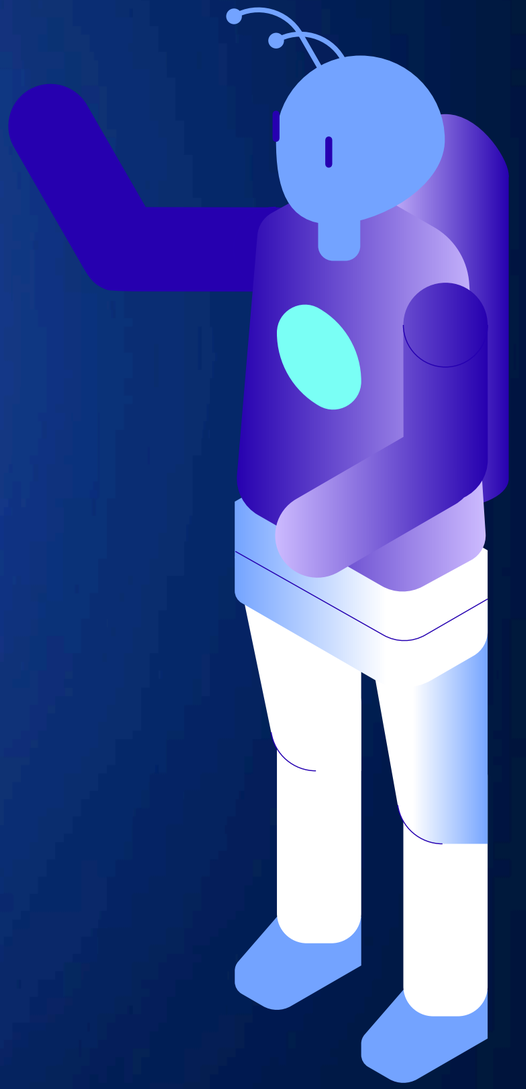
키	몸무게	시력	재산	성적	성별
180	60	1.0	A	100	남
170	50	2.0	C	90	여
160	80	1.5	B	80	남
150	70	2.0	A	50	남
121	59	0.5	B	70	여
153	85	0.1	A	50	여
169	53	0.6	A	60	남
158	49	0.3	B	80	남
201	100	0.9	C	90	여
147	50	1.0	A	80	남

2. EDA

1 EDA란?

2 어떤 경우에 사용하는가?





EDA란?

- EDA == 탐색적 데이터 분석

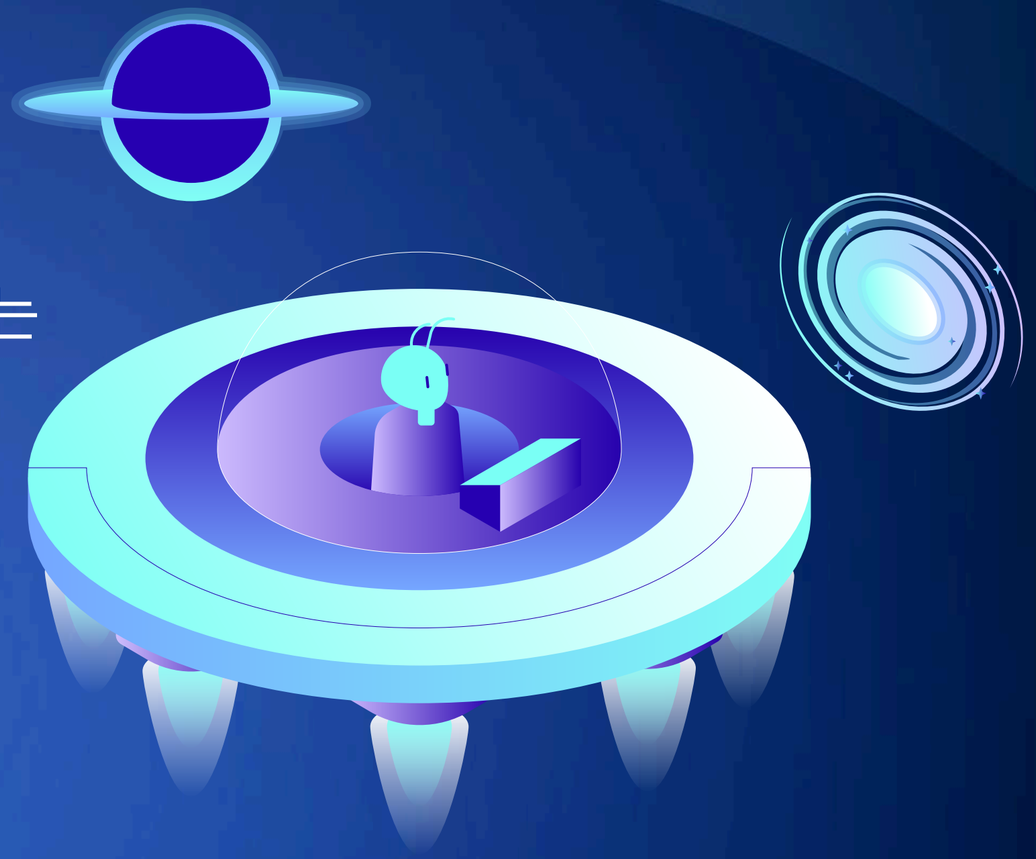
- 그래픽과 시각화를 이용하여 데이터 집합을 탐색하고 분석하는 일

어떤 경우에 사용하는가?

- 데이터의 형태
- 변수 간 관계
- 데이터 품질 문제를 나타내거나 흥미로운 인사이트를 유도할 이상치 또는 비정상 점이 데이터에 존재 여부
- 데이터에 시간 경과에 따른 패턴 존재 여부

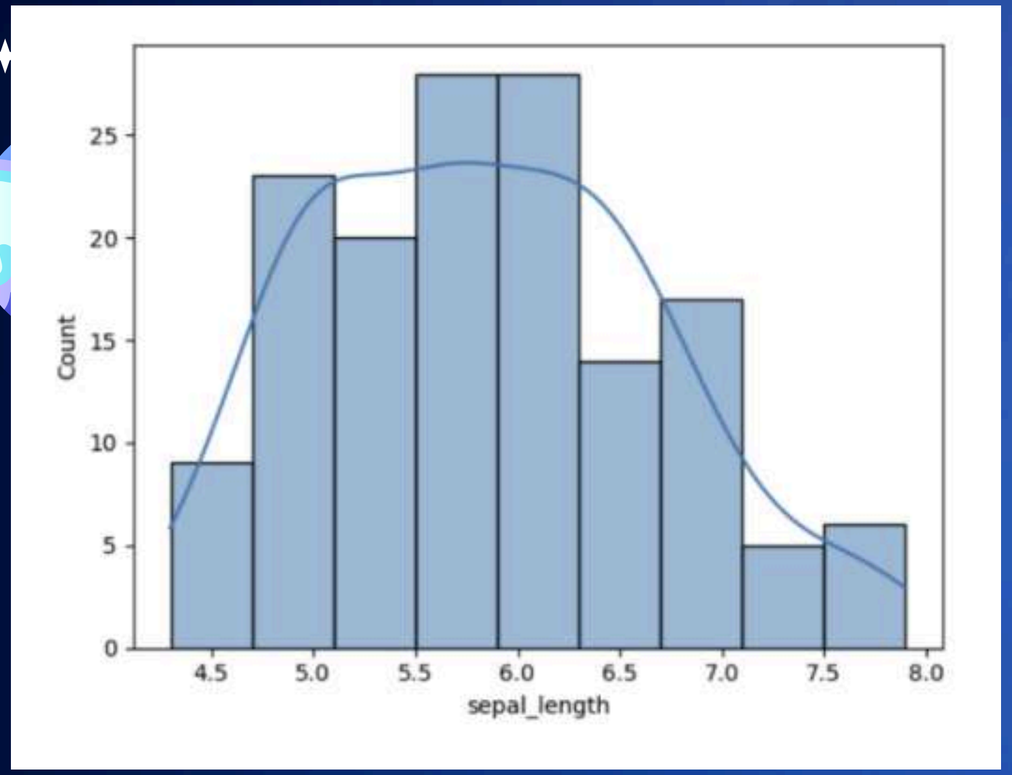
LIBRARY

↳ 프로그래밍에 사용할 수 있게 미리 만들어져 있는
Class or function들의 모임

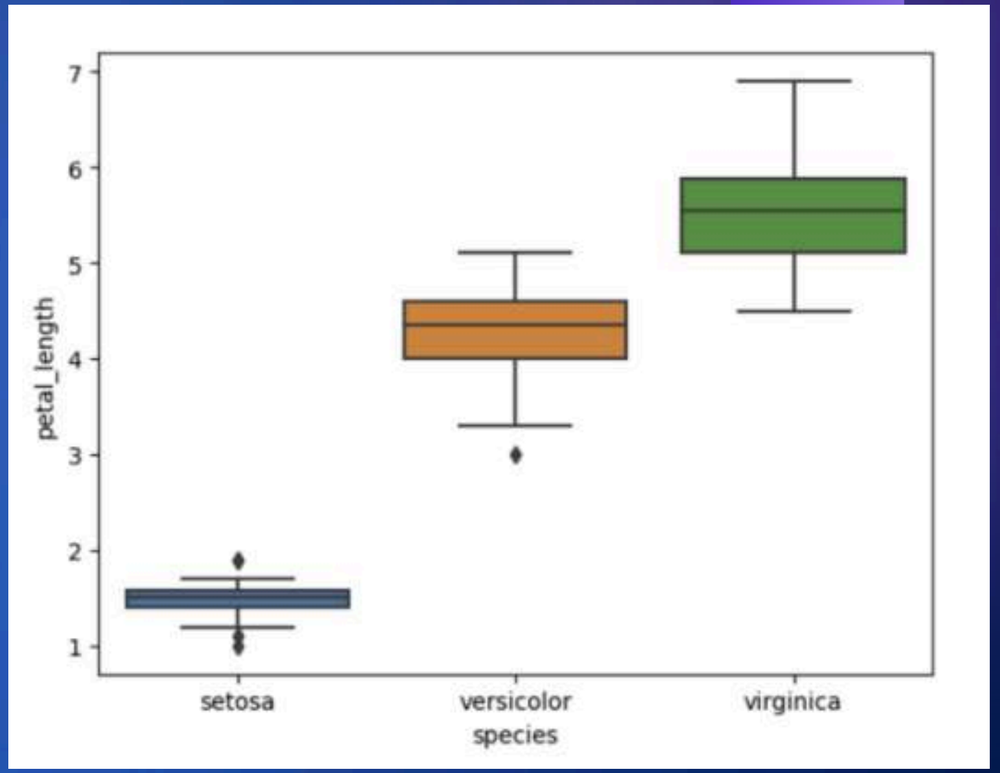


- Pandas : 데이터 조작과 분석
- Matplotlib : 데이터 시각화
- Seaborn : Matplotlib 기반으로 고급 시각화
- Numpy : 선형 대수 함수들 제공

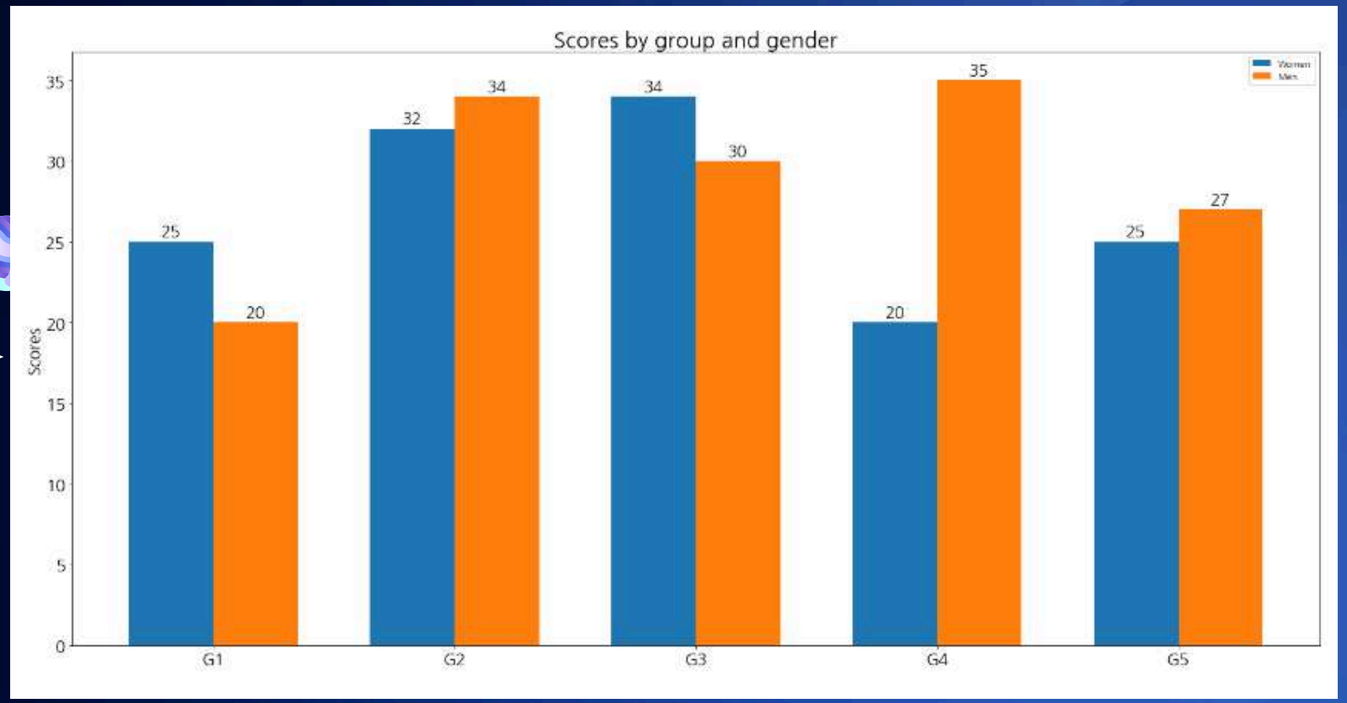
시각화



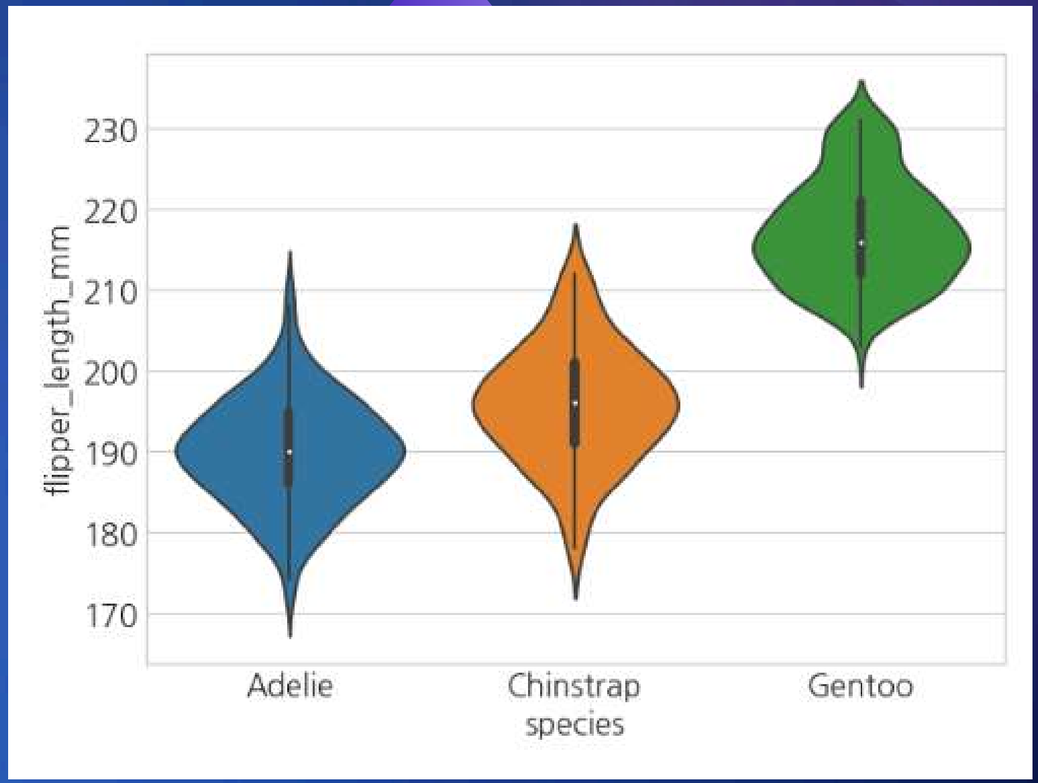
seoborn 라이브러리
histplot() 사용



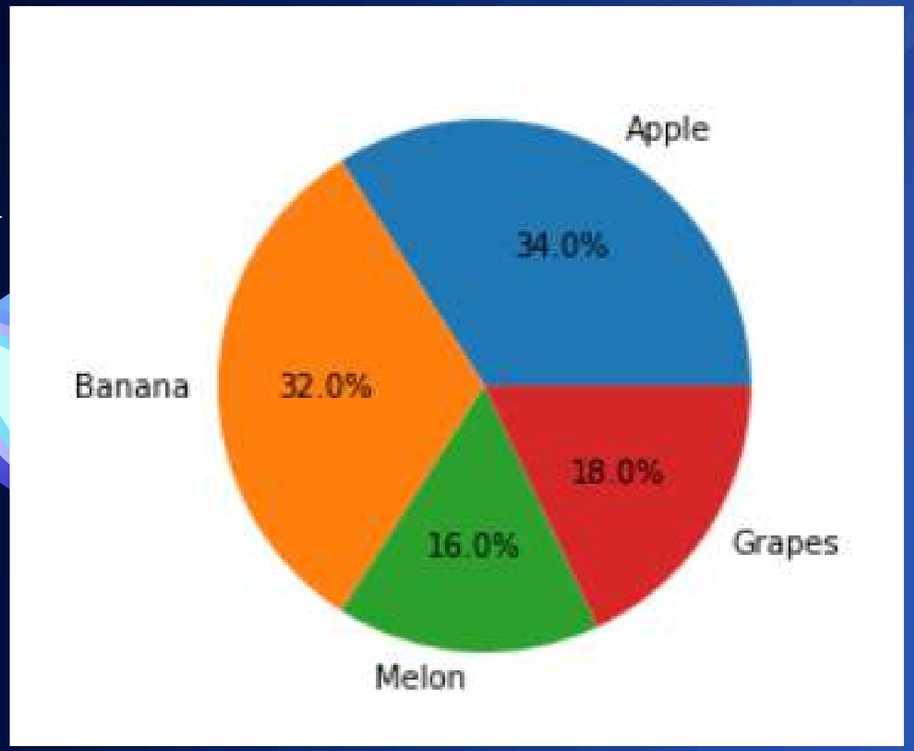
seaborn 라이브러리
boxplot() 사용



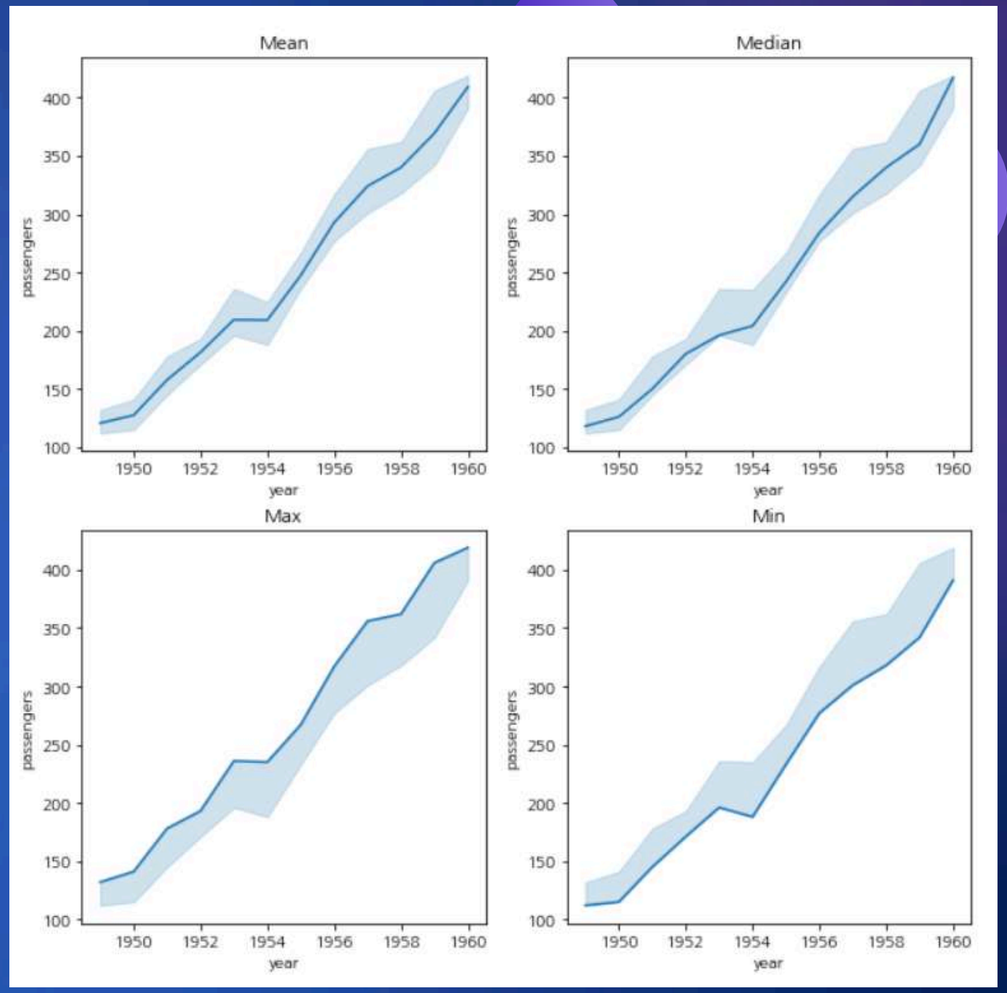
seaborn 라이브러리
countplot() 사용



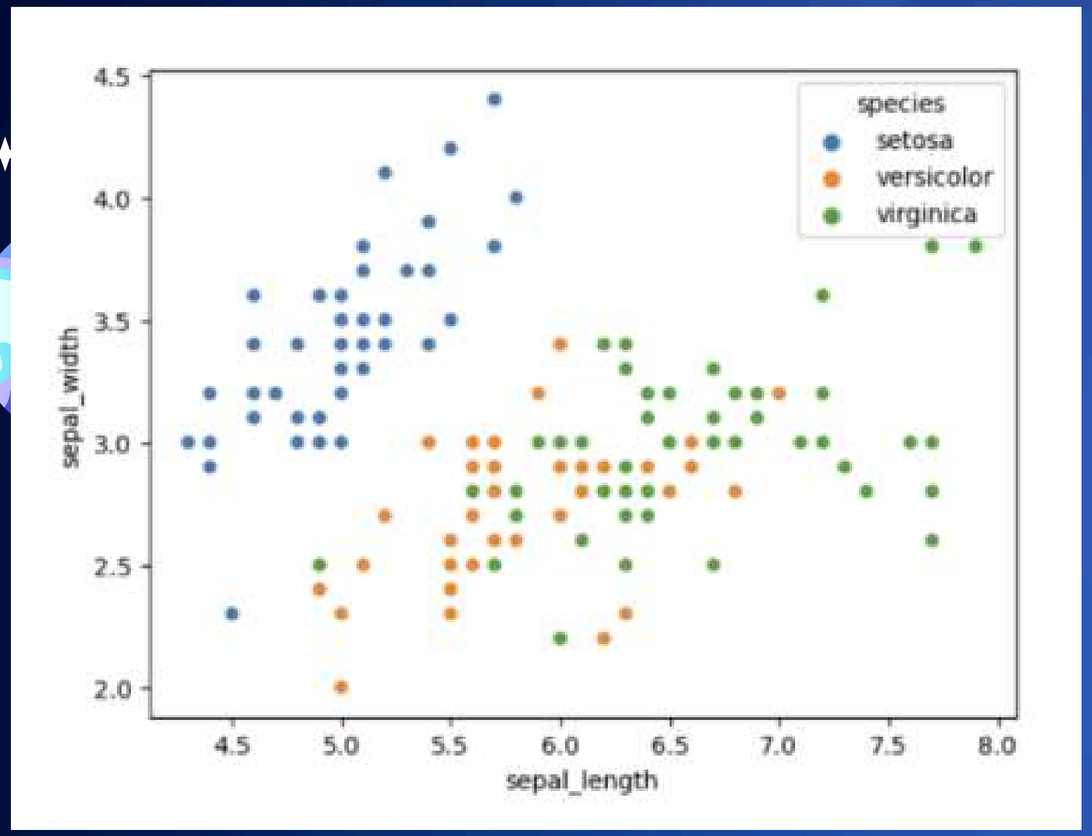
seaborn 라이브러리
viloionplot() 사용



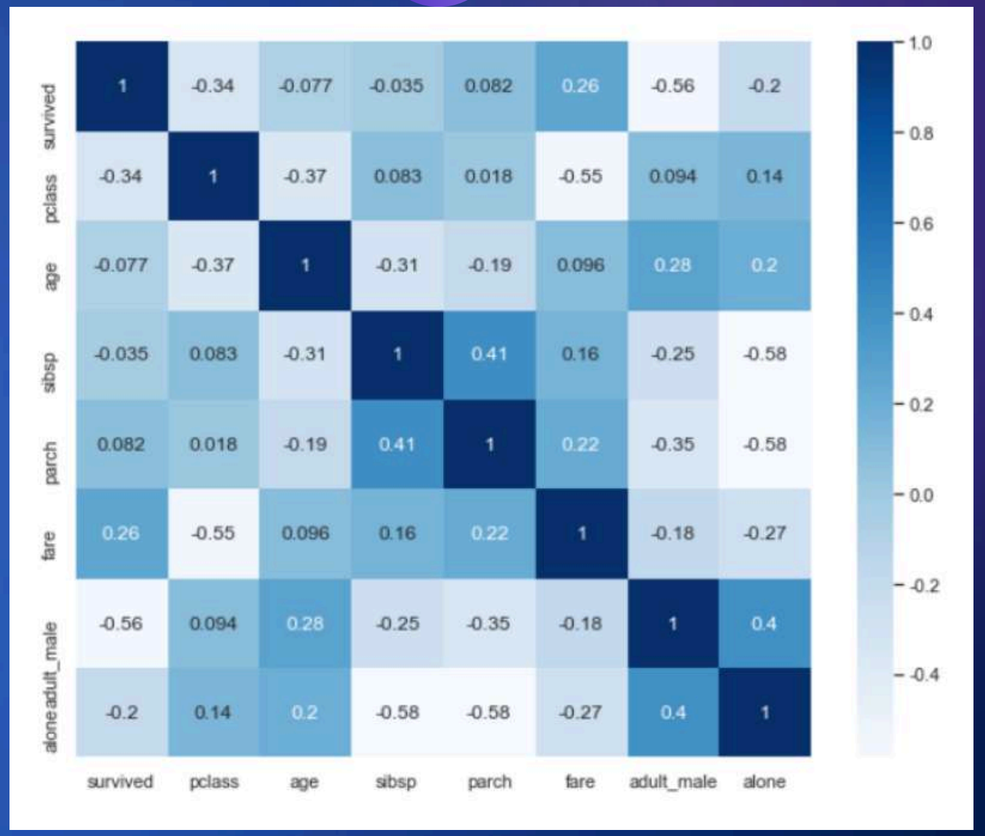
matplotlib 라이브러리
pie() 사용



seaborn 라이브러리
lineplot() 사용



seoborn 라이브러리
scatterplot() 사용



seoborn 라이브러리
heatmap() 사용

비시각화

요약통계량 확인 : describe()

```
[3]  iris.describe()
```

	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

데이터 파악 : info()

```
[2]  iris.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   sepal_length  150 non-null    float64
 1   sepal_width   150 non-null    float64
 2   petal_length  150 non-null    float64
 3   petal_width   150 non-null    float64
 4   species       150 non-null    object 
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

데이터 분포 파악 : value_counts()

```
[6]  iris.value_counts()
```

sepal_length	sepal_width	petal_length	petal_width	species	
5.8	2.7	5.1	1.9	virginica	2
6.2	2.2	4.5	1.5	versicolor	1
	2.9	4.3	1.3	versicolor	1
	3.4	5.4	2.3	virginica	1
6.3	2.3	4.4	1.3	versicolor	1
					..
5.4	3.9	1.3	0.4	setosa	1
		1.7	0.4	setosa	1
5.5	2.3	4.0	1.3	versicolor	1
	2.4	3.7	1.0	versicolor	1
7.9	3.8	6.4	2.0	virginica	1

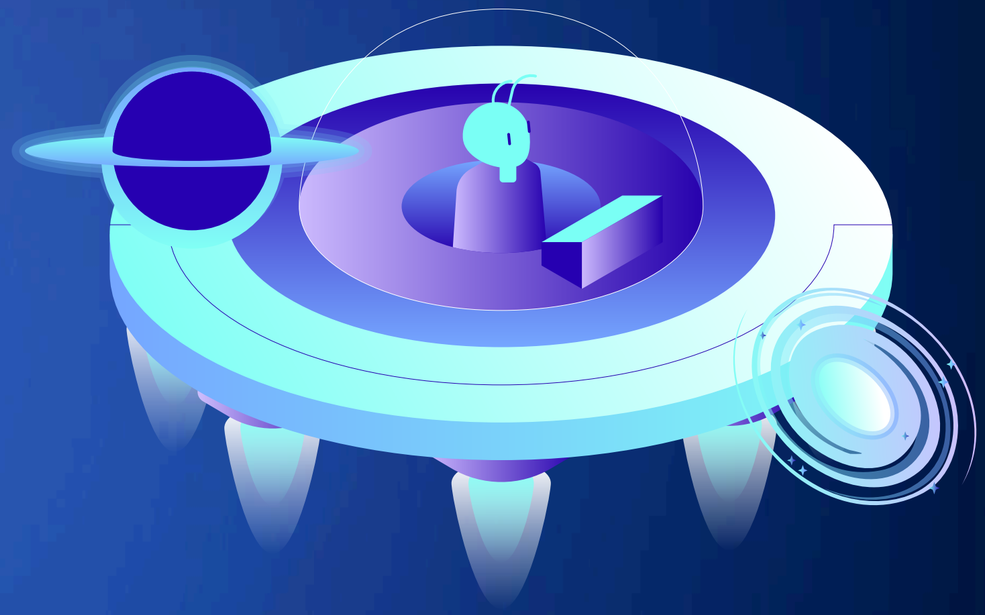
Length: 149, dtype: int64

데이터 크기 확인 : shape

```
[13]  iris.shape
```

```
(150, 5)
```

3. 데이터 전처리



데이터 전처리란?

데이터를 분석 및 처리에 적합한 형태로 만드는 작업

일반적으로 주어진 데이터는 비어있는 부분이 있거나 정합성이 맞지 않는 경우가 대다수임

데이터가 가지고 있는 본래의 정보를 최대한 왜곡시키지 않으며, 사용 목적에 맞게 효과적으로 가공해야 함

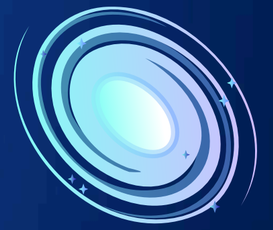
데이터 전처리 종류

- 데이터 정제(Cleansing)
- 데이터 변환(Transformation)
- 데이터 필터링(Filtering)
- 데이터 축소(Integration)
- 데이터 통합(Reduction)

데이터 전처리 과정

데이터 수집 → 데이터 정제 → 데이터 통합

데이터 축소 → 데이터 변환



데이터 전처리

결측치 처리

결측치 확인 코드
: df.isna().sum()

inplace=True
: 변경된 설정으로 덮어 씌움

1, 결측지 제거
-dropna() <-> fillna()

2, 평균**(mean),** 중앙치**(median),** 최빈값**(mode)**으로 대체

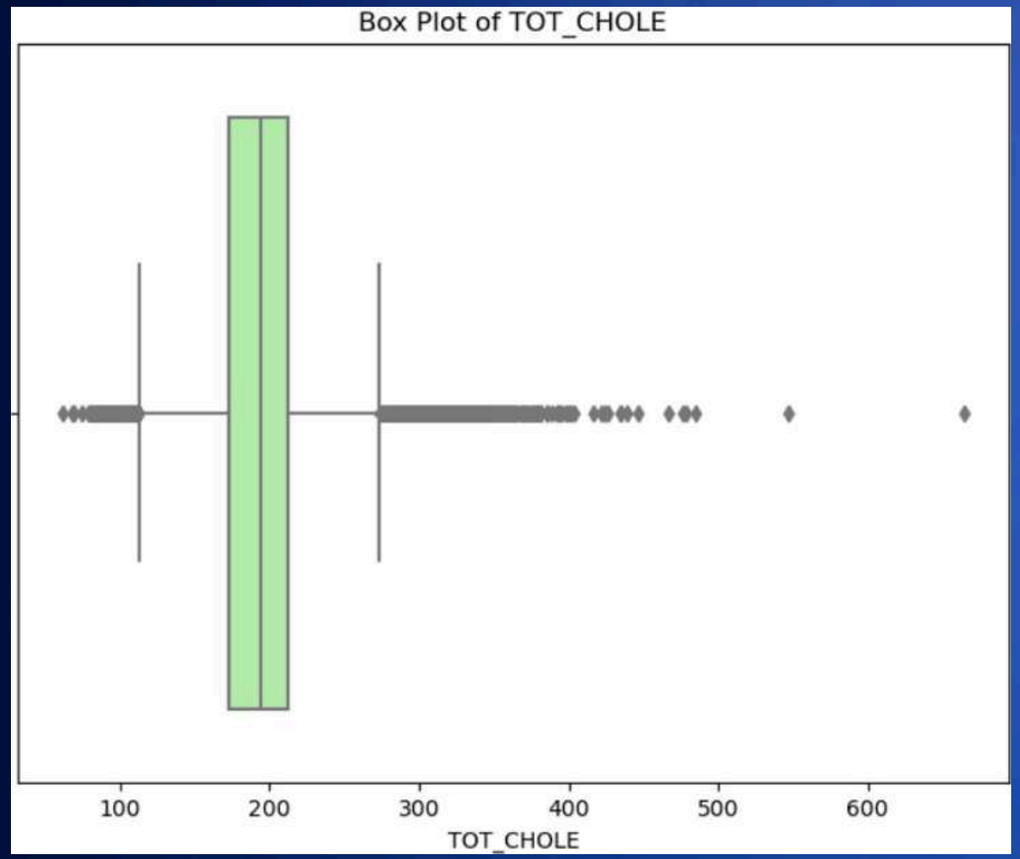
```
df['cylinders']=df['cylinders'].fillna(df.cylinders,mean())
```

3, 보간법
선형보간법**: interpolate(method='linear')**

```
df['cylinders']=df['cylinders'].interpolate(method='linear')
```

데이터 전처리

이상치 처리



df.drop(df[df['TOT_CHOLE']>=500].index)

–**drop()** 함수로 이상치 제거

–이상치를 **np.nan** 값으로 바꾼 후 원하는 값으로 채우기

df.loc[df[TOT_CHOLE]>=500, 'TOT_CHOLE']=np.nan

데이터 전처리

중복값 처리 및 데이터 보간

<중복값 처리>

-drop_duplicates()

-> 데이터가 중복되어 들어간 것인지, 아니면 실제 데이터의 값이 같은 것인지에 대한 구분 필요

<데이터 분포 변환>

-대부분의 모델**:** 변수가 특정 분포를 따른다는 가정을 기반으로 함

->log(), exp(), sqrt() 등의 함수를 이용하여 데이터 분포 변환 ⟶

데이터 전처리

데이터 단위 변환

데이터의 스케일(측정 단위)이 다를 경우,
모델에 부정적인 영향을 끼침

스케일링을 통해 단위를 일정하게 맞추는 작업 필요

- Standard Scailing: 평균 0, 분산 1 -> 분포로 변환
- MinMax Scaling: 특정 범위(ex, 0~1)로 모든 데이터 변환

데이터 전처리

데이터 선택 – loc, iloc 차이

1) 행 선택

<loc 속성>
-인덱스를 기준으로 행 데이터 추출
-df.loc[0], df.loc[[0,10,20]]
-df.loc[-1] -> 인덱스에 없는 값을 이용하면 오류가 발생함.

<iloc 속성>
-행 번호를 기준으로 행 데이터 추출
-df.iloc[1], df.iloc[[0,10,20]]
-df.iloc[-1] -> 마지막 행 데이터를 추출

```
In [7]: df.loc[-1]
---------------------------------------------------------------------------
ValueError                                Trace
File ~₩anaconda3₩Lib₩site-packages₩pandas₩c
eIndex.get_loc(self, key, method, tolerance)
    390 try:
--> 391     return self._range.index(new_key)
    392 except ValueError as err:

ValueError: -1 is not in range

The above exception was the direct cause of the

KeyError                                  Trace
Cell In[7], line 1
----> 1 df.loc[-1]
```

```
In [13]: df.iloc[-1]
Out[13]: mpg                31.0
         cylinders           4
         displacement      119.0
         horsepower         82.0
         weight             2720
         acceleration       19.4
         model_year           82
         origin              usa
         name          chevy s-10
         Name: 397, dtype: object
```

데이터 전처리

데이터 선택 – **loc, iloc** 차이

2) 열 선택

<loc 속성>
-df.loc[:, ‘manufacturer’]

```
In [17]: df.loc[:,'manufacturer']

Out[17]: 0            audi
         1            audi
         2            audi
         3            audi
         4            audi
                     ...
         229    volkswagen
         230    volkswagen
         231    volkswagen
         232    volkswagen
         233    volkswagen
         Name: manufacturer, Length: 234, dtype: object
```

<iloc 속성>
-df.iloc[:, 0]

```
In [16]: df.iloc[:,0]

Out[16]: 0            audi
         1            audi
         2            audi
         3            audi
         4            audi
                     ...
         229    volkswagen
         230    volkswagen
         231    volkswagen
         232    volkswagen
         233    volkswagen
         Name: manufacturer, Length: 234, dtype: object
```

데이터 전처리

데이터 선택 – **loc, iloc** 차이

3) 범위 선택

<loc 속성>
-df.loc[1:3]

```
In [12]: df.loc[1:3]
```

Out [12]:

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	category
1	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
2	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact
3	audi	a4	2.0	2008	4	auto(av)	f	21	30	p	compact

<iloc 속성>
-df.iloc[1:3]

```
In [11]: df.iloc[1:3]
```

Out [11]:

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	category
1	audi	a4	1.8	1999	4	manual(m5)	f	21	29	p	compact
2	audi	a4	2.0	2008	4	manual(m6)	f	20	31	p	compact

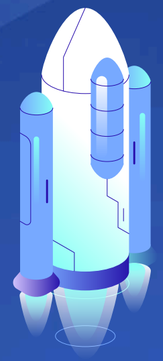
4. 과제

4. 과제

1. 데이터 전처리 하기
2. 데이터 시각화 하기
3. 딥러닝 관련 문제 풀기

	gender	race_ethnicity	parental_level_of_education	lunch	test_preparation_course	math_score	reading_score	writing_score
0	female	group B	bachelor's degree	standard	none	72	72	74
1	female	group C	some college	standard	completed	69	90	88
2	female	group B	master's degree	standard	none	90	95	93
3	male	group A	associate's degree	free/reduced	none	47	57	44
4	male	group C	some college	standard	none	76	78	75
...	...	...	...	...	...	...	...	...
995	female	group E	master's degree	standard	completed	88	99	95
996	male	group C	high school	free/reduced	none	62	55	55
997	female	group C	high school	free/reduced	completed	59	71	65
998	female	group D	some college	standard	completed	68	78	77
999	female	group D	some college	free/reduced	none	77	86	86

1000 rows × 8 columns



THANK YOU

https://github.com/konkuk-kuggle/9th_Seminar