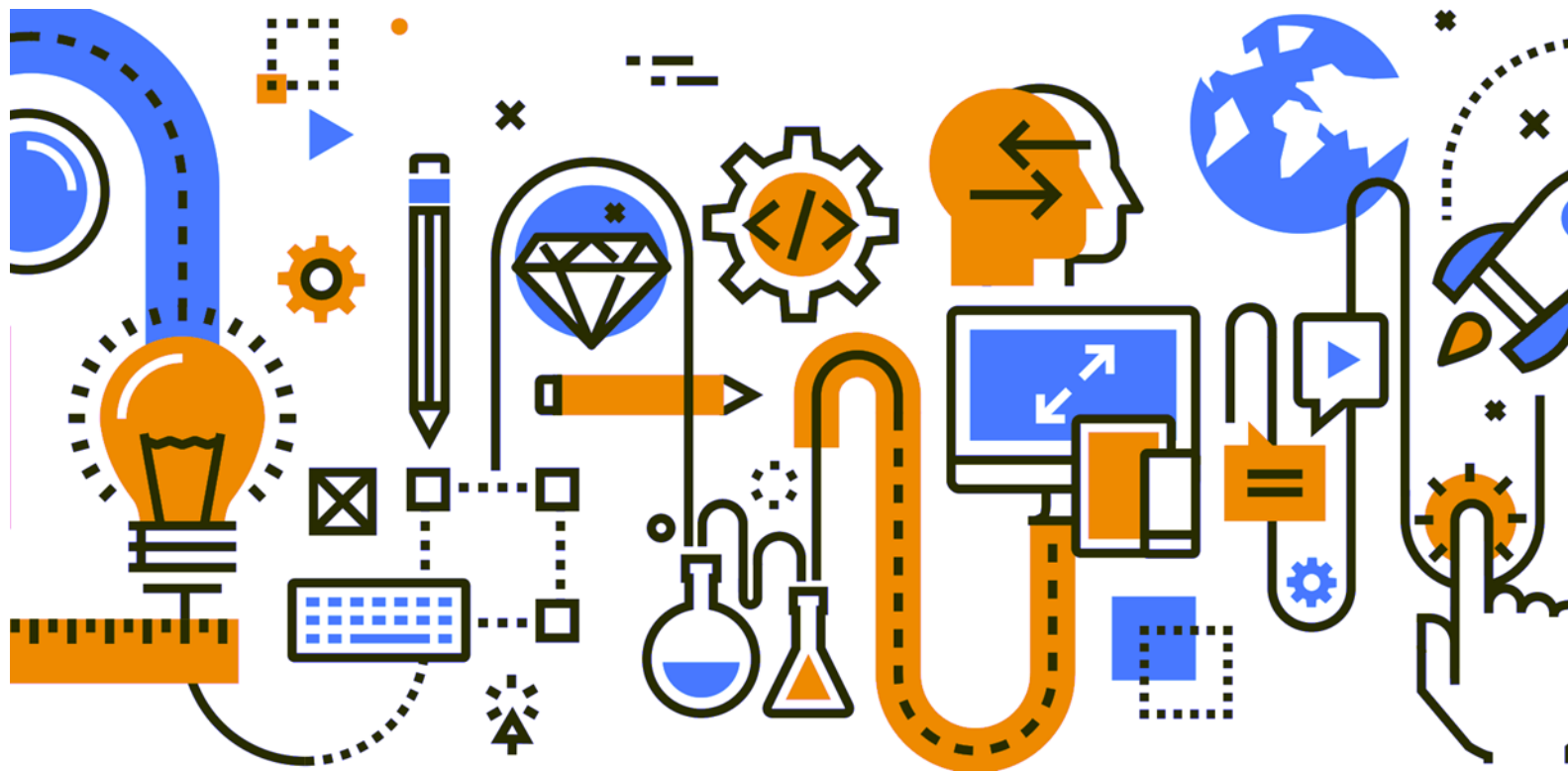


# 데이터 분석 및 시각화



# 행렬 (Matrix)

■ Matrix is an array of numbers

$$\begin{bmatrix} 6 & 4 & 24 \\ 1 & -9 & 8 \end{bmatrix}$$

A Matrix

(This one has 2 Rows and 3 Columns)

Notation

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,1} & a_{2,2} & a_{2,3} \end{bmatrix}$$

Adding

$$\begin{bmatrix} 3 & 8 \\ 4 & 6 \end{bmatrix} + \begin{bmatrix} 4 & 0 \\ 1 & -9 \end{bmatrix} = \begin{bmatrix} 7 & 8 \\ 5 & -3 \end{bmatrix}$$

These are the calculations:

$$3+4=7 \quad 8+0=8$$

$$4+1=5 \quad 6-9=-3$$

Transposing

$$\begin{bmatrix} 6 & 4 & 24 \\ 1 & -9 & 8 \end{bmatrix}^T = \begin{bmatrix} 6 & 1 \\ 4 & -9 \\ 24 & 8 \end{bmatrix}$$

# 행렬 곱(Matrix Multiplication)

To work out the answer for the **1st row** and **1st column**:

"Dot Product"

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 \\ \end{bmatrix}$$

The "Dot Product" is where we **multiply matching members**, then sum up:

$$(1, 2, 3) \bullet (7, 9, 11) = 1 \times 7 + 2 \times 9 + 3 \times 11 \\ = 58$$

We match the 1st members (1 and 7), multiply them, likewise for the 2nd members (2 and 9) and the 3rd members (3 and 11), and finally sum them up.

# 행렬 곱

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix}$$

$$(1, 2, 3) \bullet (8, 10, 12) = 1 \times 8 + 2 \times 10 + 3 \times 12 \\ = 64$$

We can do the same thing for the **2nd row** and **1st column**:

$$(4, 5, 6) \bullet (7, 9, 11) = 4 \times 7 + 5 \times 9 + 6 \times 11 \\ = 139$$

And for the **2nd row** and **2nd column**:

$$(4, 5, 6) \bullet (8, 10, 12) = 4 \times 8 + 5 \times 10 + 6 \times 12 \\ = 154$$

And we get:

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix} \times \begin{bmatrix} 7 & 8 \\ 9 & 10 \\ 11 & 12 \end{bmatrix} = \begin{bmatrix} 58 & 64 \\ 139 & 154 \end{bmatrix} \checkmark$$

# 행렬 곱(Matrix Multiplication)

$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \times 4 + 2 \times 5 + 3 \times 6 \end{bmatrix} = \begin{bmatrix} 32 \end{bmatrix}$$

$$\begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} \begin{bmatrix} 1 & 2 & 3 \end{bmatrix} = \begin{bmatrix} 4 \times 1 & 4 \times 2 & 4 \times 3 \\ 5 \times 1 & 5 \times 2 & 5 \times 3 \\ 6 \times 1 & 6 \times 2 & 6 \times 3 \end{bmatrix} = \begin{bmatrix} 4 & 8 & 12 \\ 5 & 10 & 15 \\ 6 & 12 & 18 \end{bmatrix}$$

# 행렬 계산

**Matrix Calculator**

**A: 3×3**

▲ ▼ ◀ ▶

3	2	1
6	3	3
1	4	5

**B: 3×3**

▲ ▼ ◀ ▶

3	2	1
6	3	3
1	4	5

A+B   A-B   AB   det(A)   det(A) details   example   inv(A)   A=I   B=I   A↔B   A<sup>T</sup>   B<sup>T</sup>

A<sup>2</sup>   B<sup>2</sup>   cA   cB   1

AB=

22	16	14
39	33	30
32	34	38

to A

to B

from A

from B

© 2019 MathsIsFun.com v0.641

# 넘파이(Numpy)

NumPy(Numerical Python)는 데이터 분석, 수학/과학연산을 위한 파이썬 기본 패키지로  
고성능의 다차원 배열 객체와 다양한 객체에 대해 고속 연산을 가능하게 합니다.

```
data = np.array([1,2,3])
```

data

1
2
3

data

1
2
3

.max() = 3

data

1
2
3

.min() = 1

data

1
2
3

.sum() = 6

```
np.random.random((4,3,2))
```

```
np.array([ [[1,2],[3,4]],  
          [[5,6],[7,8]] ])
```



	5	6
1	2	8
3	4	

	0.3	0.6	0.8
0.2	0.5	0.3	0.8
0.7	0.6	0.1	0.5
0.4	0.5	0.5	0.3
0.1	0.1	0.4	

4 3 2

# 넘파이 (Numpy)



NumPy.ipynb

## ■ Numpy 라이브러리 импорт

```
import numpy as np
```

## ■ Numpy Array 생성

```
my_list = [1, 2, 3]
```

```
np.array(my_list)
```

```
[Out] array([1, 2, 3])
```

```
my_matrix = [[1, 2, 3],[4, 5, 6],[7, 8, 9]]
```

```
np.array(my_matrix)
```

```
[Out] array([[1, 2, 3],  
            [4, 5, 6],  
            [7, 8, 9]])
```



# 데이터 분석 필수 라이브러리 판다스(Pandas)

판다스(Pandas)는 데이터 처리와 분석을 위해 널리 사용되는 파이썬 라이브러리

데이터사이언티스트에게 필요한 기본적이면서도 아주 중요한 도구  
행과 열로 이루어진 데이터 객체를 만들어 다룰 수 있음  
데이터를 수집하고 정리하는 데 최적화 된 도구



판다스는 시리즈(Series)와 데이터프레임(DataFrame)이라는 구조화된 데이터 형식을 제공

시리즈(Series) : 1차원 배열

데이터프레임(DataFrame) : 2차원 배열

# 판다스 시리즈(Series)

1차원의 배열의 값(values)과 각 값에 대응하는 인덱스(index)를 부여할 수 있는 데이터 구조

```
import pandas as pd
```

```
sr = pd.Series([20000, 18000, 5000])  
print(sr)
```

```
sr = pd.Series([20000, 18000, 5000], index = ['피자', '치킨', '맥주'])  
print(sr.index)  
print(sr.values)  
print(sr)
```

```
sr = pd.Series({'피자': 20000, '치킨': 18000, '맥주': 5000})  
print(sr)
```

index	values
피자	20000
치킨	18000
맥주	5000
dtype: int64	

# 판다스 데이터프레임(DataFrame)

데이터프레임은 행과 열을 가지는 자료구조로 인덱스(index), 열(columns), 값(values)으로 구성

열(columns)

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL
1	5575-GNVDE	Male	0	No	No	34	Yes	No	DSL
2	3668-QPYBK	Male	0	No	No	2	Yes	No	DSL
3	7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL
4	9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic

인덱스  
(index)

값(values)

# 판다스 실습



Pandas\_Series.ipynb

Pandas\_DataFrame.ipynb

Pandas\_MissingData.ipynb

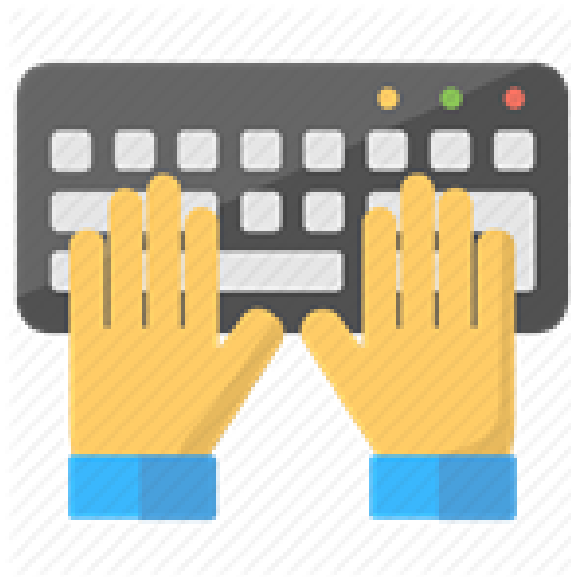
Pandas\_Groupby.ipynb

Pandas\_DataInputOutput.ipynb

Pandas\_Operation.ipynb

# 판다스 Exercise

<https://bit.ly/3bnwEHT>



# 판다스 Q&A



**Question 1 – Define Python Pandas.**

**Question 2 – What Are The Different Types Of Data Structures In Pandas?**

**Question 6 – What Are The Most Important Features Of The Pandas Library?**

**Question 8 – What are the different ways of creating DataFrame in pandas? Explain with examples.**

**Question 9 – Explain Categorical Data In Pandas?**

**Question 14 – How Can You Iterate Over Dataframe In Pandas?**

**Question 17 – What Is Groupby Function In Pandas?**

# 데이터 분석 실습 - 타이타닉 데이터셋



DataAnalysis\_Titanic.ipynb

## ■ Seaborn 라이브러리 импорт

```
import seaborn as sns
```

## ■ 파일에서 데이터를 로드

```
df = sns.load_dataset('titanic')
```

## ■ 데이터 확인

```
df.head()
```

```
df.head(20)
```

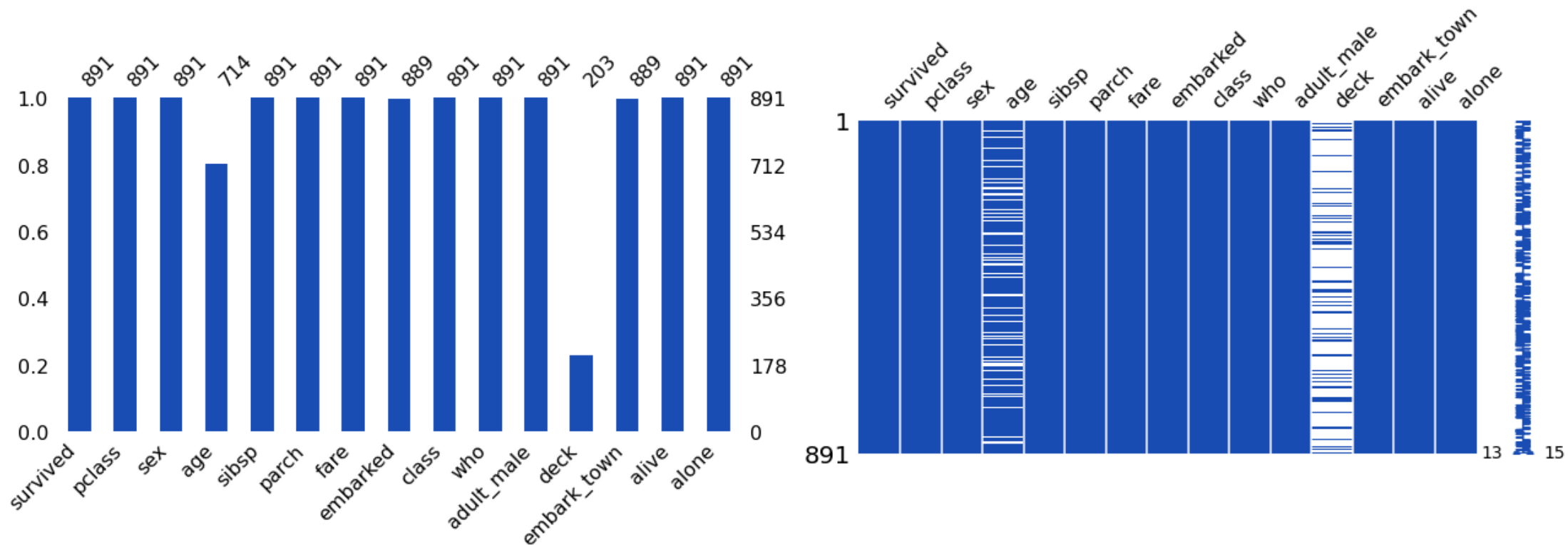
```
df.tail()
```

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True

# 데이터 분석 실습 - 타이타닉 데이터셋

## ■ 결측값 확인

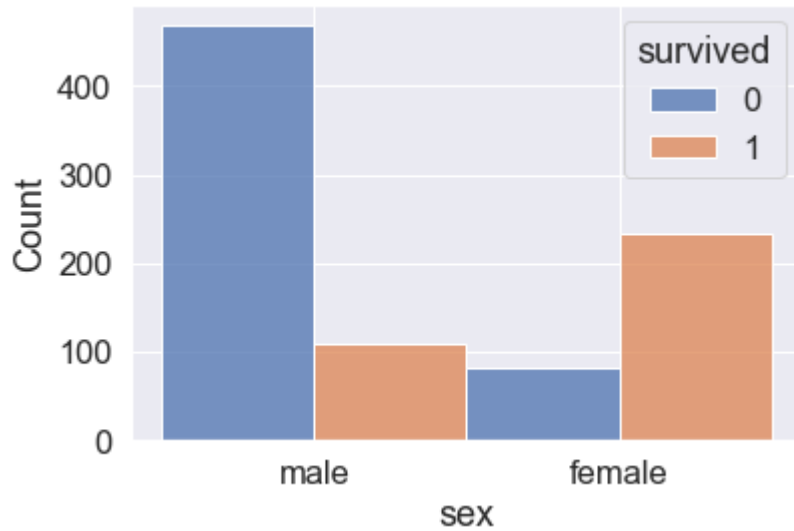
```
!pip install missingno
import missingno as msno
msno.bar(df, figsize=(10, 5), color=(0.1, 0.3, 0.7))
msno.matrix(df, figsize=(10, 5), color=(0.1, 0.3, 0.7))
```



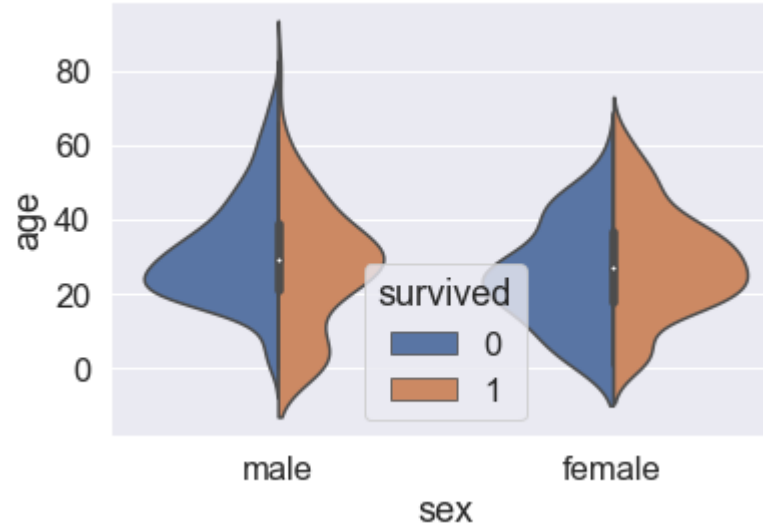


# 데이터 분석 실습 - 타이타닉 데이터셋

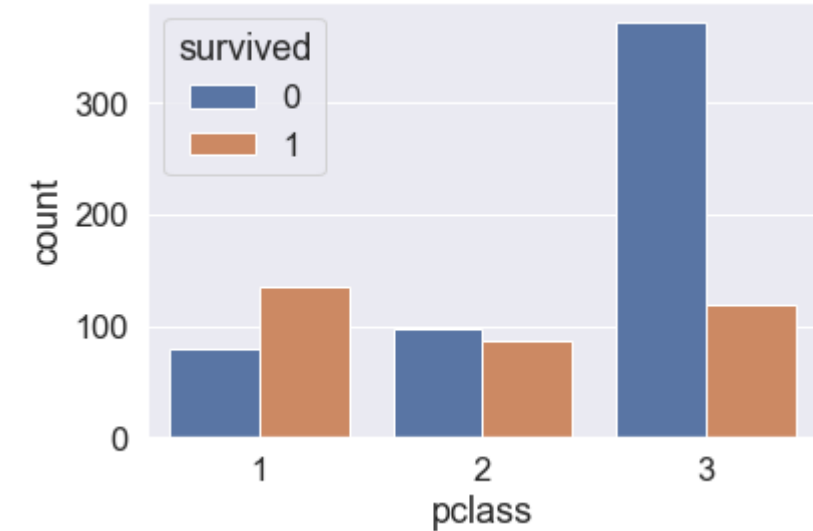
## ■ 성별(sex)에 따른 생존율 분포



## ■ 승객 나이와 생존 여부와의 관계



## ■ 객실등급과 생존율



```
sns.histplot(x='sex', hue='survived', multiple='dodge', data=df)
```

```
sns.violinplot(x='sex', y='age', hue='survived', data=df, split=True)
```

```
sns.countplot(x='pclass', hue='survived', data=df)
```

# 데이터 분석 실습 - 통신사 이탈고객 데이터셋



DataAnalysis\_Telecom.ipynb

## ■ Pandas 라이브러리 импорт

```
import pandas as pd
```

## ■ 파일에서 데이터 로드

```
df = pd.read_csv('churn_data.csv')
```

## ■ 데이터 확인

df.head()

df.head(20)

df.tail()

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
0	7590-VHVEG	Female	0	Yes	No	1	No	No phone service
1	5575-GNVDE	Male	0	No	No	34	Yes	No

# 데이터 분석 실습 - 통신사 이탈고객 데이터셋

## ■ 데이터구조 파악

`df.info()`

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 7043 entries, 0 to 7042
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype  
---  -
 0   customerID            7043 non-null  object 
 1   gender                7043 non-null  object 
 2   SeniorCitizen         7043 non-null  int64  
 3   Partner               7043 non-null  object 
 4   Dependents            7043 non-null  object 
 5   tenure                7043 non-null  int64  
.....
18  MonthlyCharges        7043 non-null  float64
19  TotalCharges          7043 non-null  object 
20  Churn                 7043 non-null  object 
dtypes: float64(1), int64(2), object(18)
memory usage: 1.1+ MB
```

## ■ 데이터 타입 확인

`df.dtypes`

```
customerID    object
gender        object
SeniorCitizen  int64
Partner       object
Dependents    object
tenure        int64
PhoneService  object
MultipleLines object
InternetService object
OnlineSecurity object
OnlineBackup  object
DeviceProtection object
TechSupport   object
StreamingTV   object
StreamingMovies object
Contract      object
PaperlessBilling object
PaymentMethod object
MonthlyCharges float64
TotalCharges  object
Churn         object
dtype: object
```

## ■ Null 데이터 확인

`df.isnull().sum()`

```
customerID    0
gender        0
SeniorCitizen  0
Partner       0
Dependents    0
tenure        0
PhoneService  0
MultipleLines  0
InternetService 0
OnlineSecurity 0
OnlineBackup  0
DeviceProtection 0
TechSupport   0
StreamingTV   0
StreamingMovies 0
Contract      0
PaperlessBilling 0
PaymentMethod 0
MonthlyCharges 0
TotalCharges  0
Churn         0
dtype: int64
```

# 데이터 분석 실습 - 통신사 이탈고객 데이터셋

## 통계 정보

df.describe()

	SeniorCitizen	tenure	MonthlyCharges
count	7043.000000	7043.000000	7043.000000
mean	0.162147	32.371149	64.761692
std	0.368612	24.559481	30.090047
min	0.000000	0.000000	18.250000
25%	0.000000	9.000000	35.500000
50%	0.000000	29.000000	70.350000
75%	0.000000	55.000000	89.850000
max	1.000000	72.000000	118.750000

## 데이터 상관관계 분석

df.corr()

	SeniorCitizen	tenure	MonthlyCharges
SeniorCitizen	1.000000	0.016567	0.220173
tenure	0.016567	1.000000	0.247900
MonthlyCharges	0.220173	0.247900	1.000000

# 데이터 분석 /전처리 실습 - 통신사 이탈고객 데이터셋

## ■ 데이터 전처리

입력 데이터에서 제외: drop()

Null 데이터 처리: dropna(), fillna()

누락데이터 처리: replace()

데이터타입 변환 : astype()

특성 추출 (feature engineering) : df['new\_feature'] = df['f\_1']/df['f\_2']

```
df.drop('customerID', axis=1, inplace=True)
```

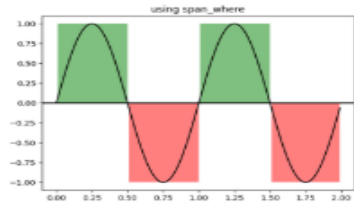
```
df['TotalCharges'].replace([' '], ['0'], inplace=True)
```

```
df['TotalCharges'] = df['TotalCharges'].astype(float)
```

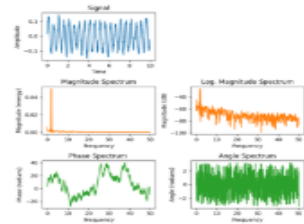
```
df['Churn'].replace(['Yes', 'No'], [1, 0], inplace=True)
```

# 데이터 시각화 - 맷플롯립 (Matplotlib)

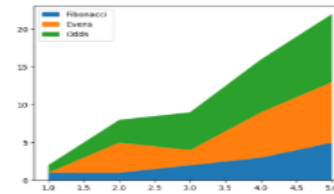
데이터를 차트나 플롯(Plot)으로 표시할 때 가장 많이 사용되는 데이터 시각화 라이브러리



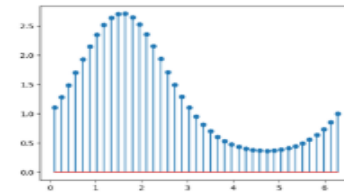
Using span\_where



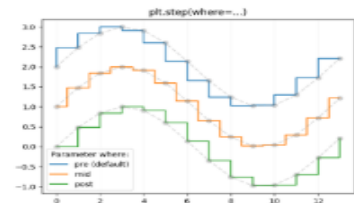
Spectrum Representations



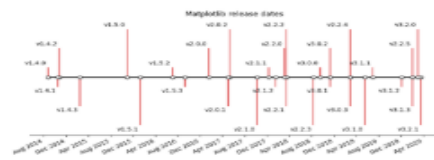
Stackplot Demo



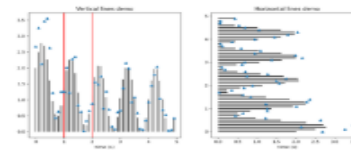
Stem Plot



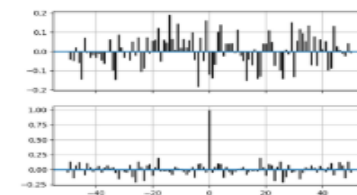
Step Demo



Creating a timeline with lines, dates, and text



hlines and vlines



Cross- and Auto-Correlation Demo

# 데이터 시각화 - 맷플롯립(Matplotlib)

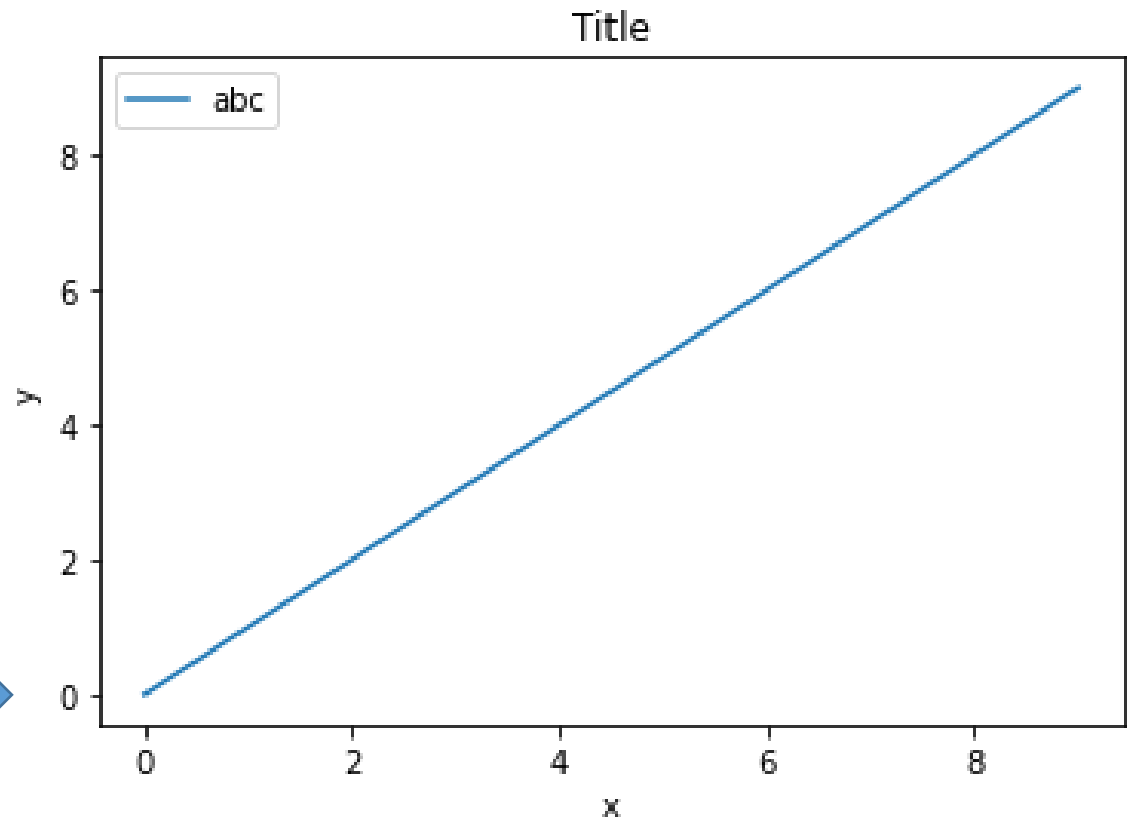
## ■ 라이브러리 импорт

```
import matplotlib.pyplot as plt  
%matplotlib inline
```

## ■ Matplotlib 사용법(예시)

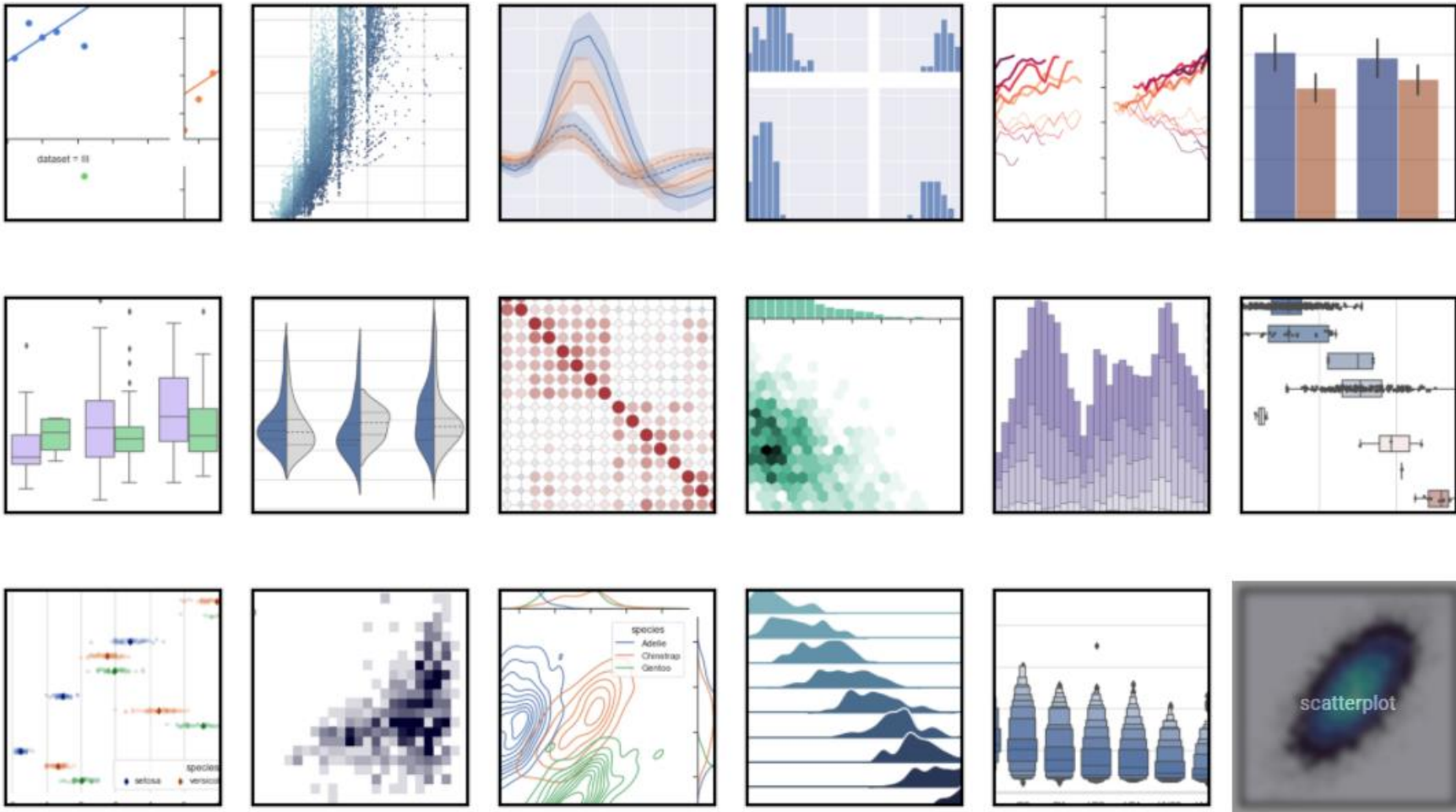
```
x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]  
y = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
plt.plot(x, y)  
plt.title('Title')  
plt.xlabel('x')  
plt.ylabel('y')  
plt.legend(['abc'])  
plt.show()
```



# 데이터 시각화 - 씨본(Seaborn)

Matplotlib을 기반으로 다양한 색상 테마와 통계용 차트 등의 기능을 추가한 시각화 라이브러리





# 데이터 시각화 - 씨본(Seaborn)

## ■ 패키지 설치

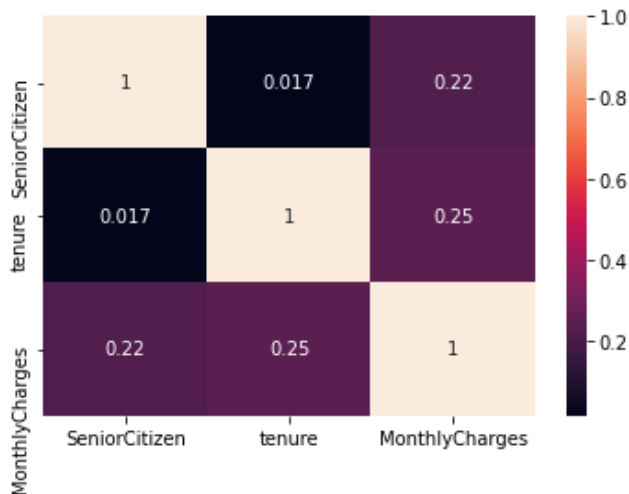
```
!pip install seaborn
```

## ■ 라이브러리 импорт

```
import seaborn as sns
```

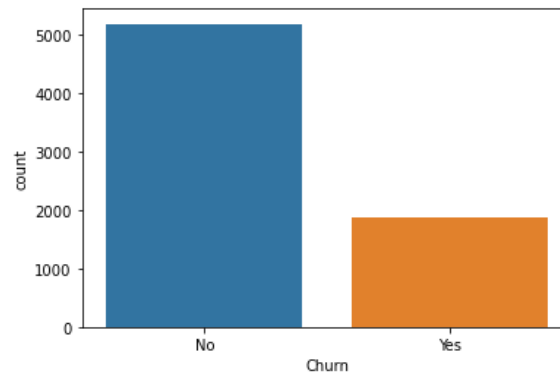
## ■ 상관관계 히트맵

```
sns.heatmap(df.corr(), annot=True)
```



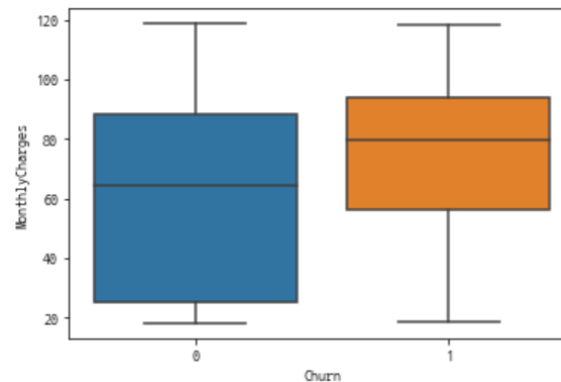
## ■ 카운트 플롯

```
sns.countplot(x='Churn', data=df)
```



## ■ 박스 플롯

```
sns.boxplot(x='Churn',  
            y='MonthlyCharges', data=df)
```



# JavaScript Graphing Library – ECharts

<https://echarts.apache.org/examples/en/index.html>



# JavaScript Graphing Library – Plotly

<https://plot.ly/javascript/>

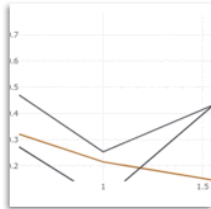
## Fundamentals



Configuration Options



Responsive / Fluid Layouts



uirevision in Plotly.react



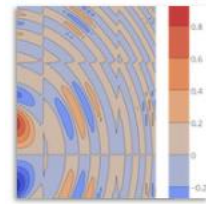
React Plotly.js



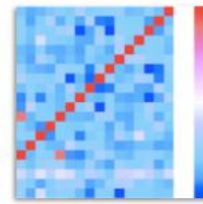
Analytical Apps with Dash

[More Fundamentals >](#)

## Scientific Charts



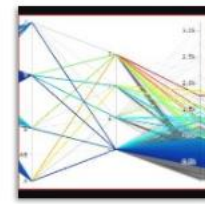
Contour Plots



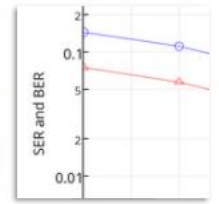
Heatmaps



Ternary Plots



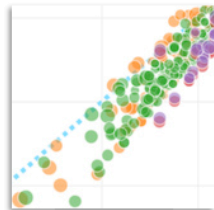
Parallel Coordinates Plot



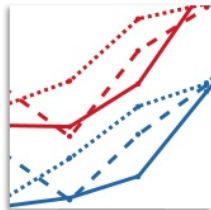
Log Plots

[More Scientific Charts >](#)

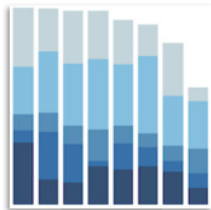
## Basic Charts



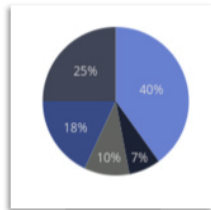
Scatter Plots



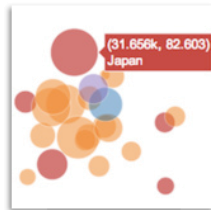
Line Charts



Bar Charts



Pie Charts



Bubble Charts

[More Basic Charts >](#)

## Financial Charts



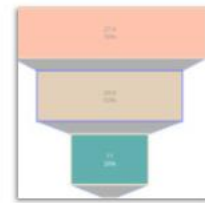
Waterfall Charts



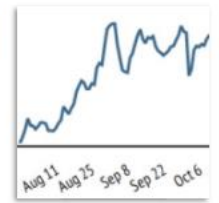
Indicators



Candlestick Charts



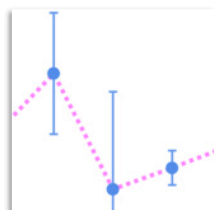
Funnel and Funnelarea Charts



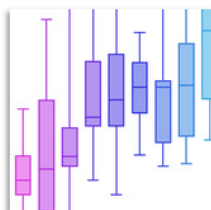
Time Series

[More Financial Charts >](#)

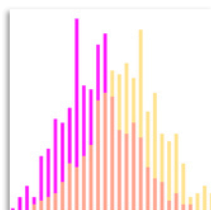
## Statistical Charts



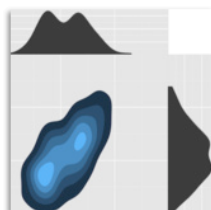
Error Bars



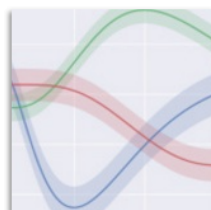
Box Plots



Histograms



2d Density Plots



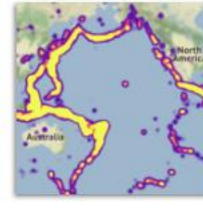
Continuous Error

[More Statistical Charts >](#)

## Maps



Mapbox Map Layers



Mapbox Density



Choropleth Mapbox



Lines on Maps

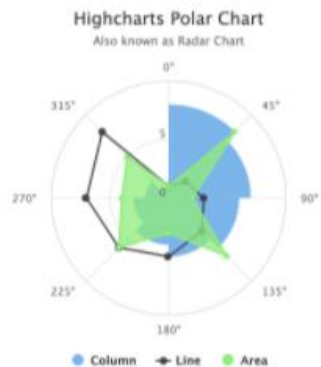


Bubble Maps

[More Maps >](#)

# JavaScript Graphing Library – Highcharts

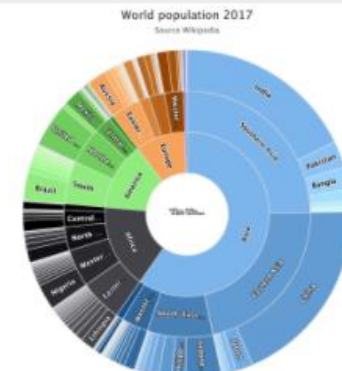
<https://www.highcharts.com/demo>



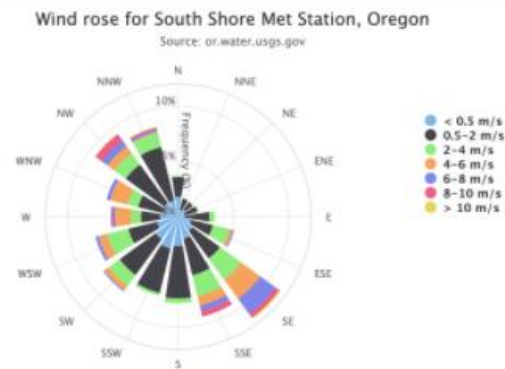
Polar (radar) chart



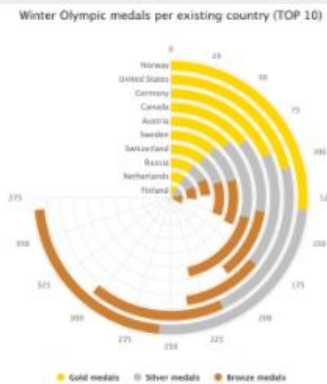
Spiderweb



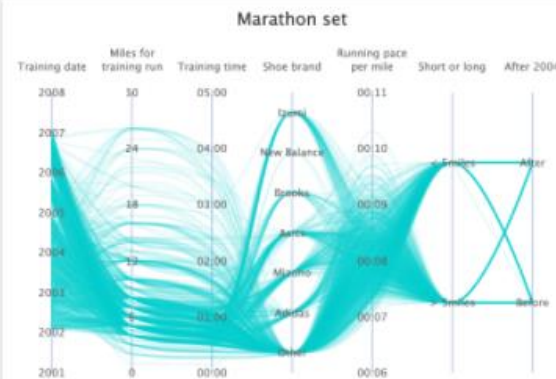
Sunburst



Wind rose



Radial bar chart



Parallel coordinates



통계적 사고관을 갖추고  
데이터 과학 기초를 이해하면  
변화하는 세상을 살아가는 데  
분명 도움이 될 겁니다.

# Thank you