

AI+X 인재양성 확대를 위한 교수자 연수

한국폴리텍대학 대구캠퍼스
SI엔지니어링학과 강현우

2강

-기계학습 개념 및 타이타닉 생존자 예측 실습



기계학습의 등장 배경

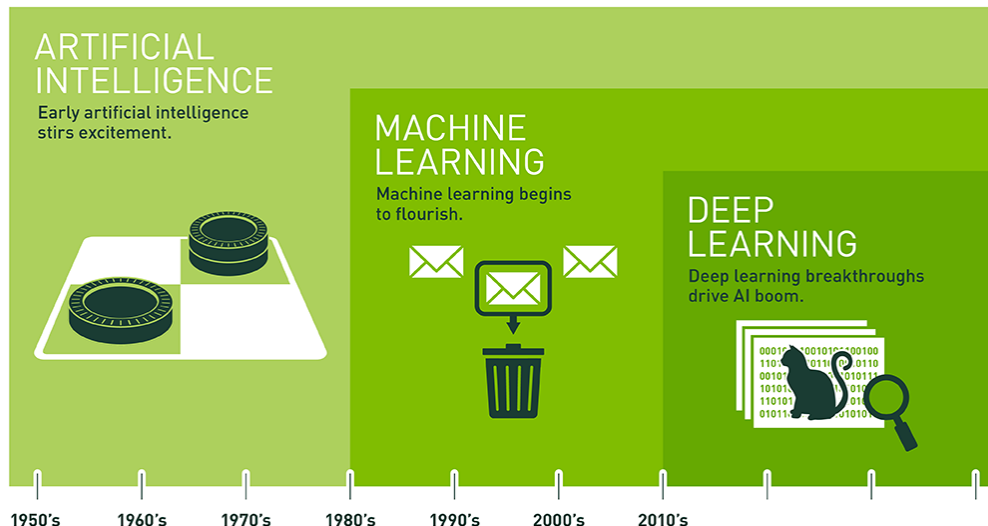
◆ 왜? 기계 학습?

- 1980년대 인공 지능 연구의 대표적인 방법
= 전문가 시스템
- 사람이 직접 많은 수의 규칙을 만드는 것을 전제
- 규칙을 정확하게 규정할 수 없는 분야는 어떻게?
- 사람조차 정확한 원리를 모르는 영역에 대해 요구

기계가 학습한다는 것?

◆ Machine Learning

- 어떤 컴퓨터 프로그램이 T라는 작업을 수행한다.
- 이 프로그램의 성능을 P 라는 척도로 평가했을 때
- 경험 E를 통하여 성능이 개선된다면
- 이 프로그램은 학습을 한다고 말할 수 있다.



Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

출처: Nvidia.com



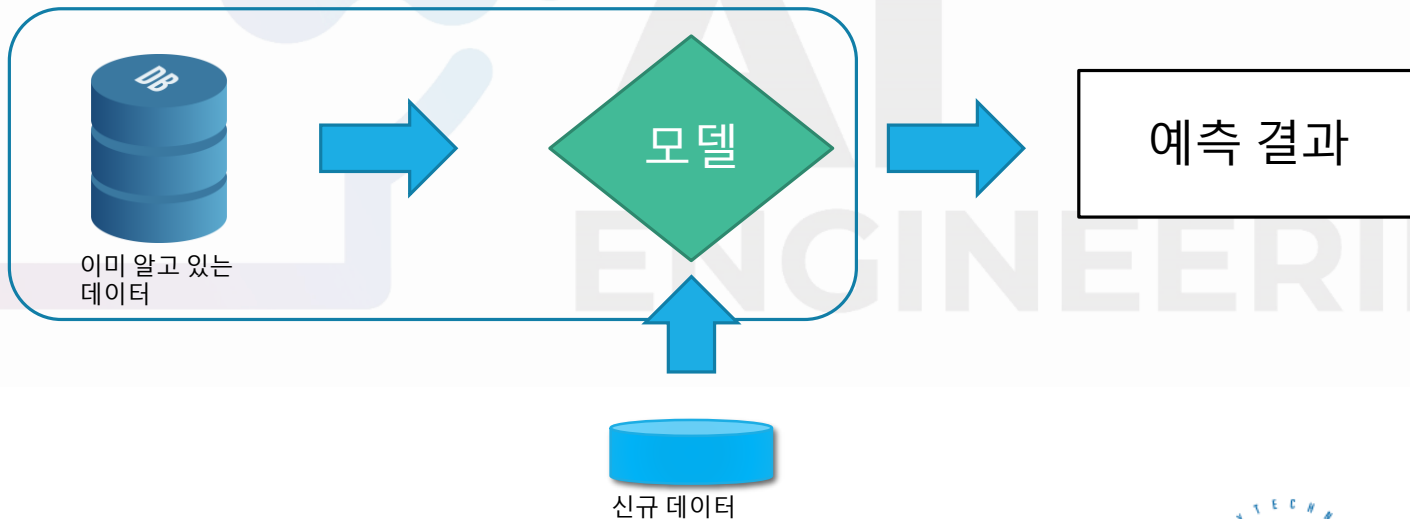
한국폴리텍대학
대구캠퍼스

기계 학습

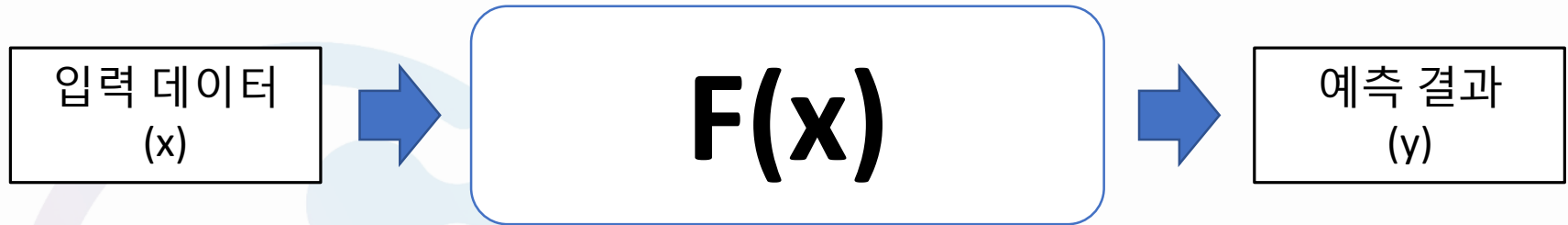
◆ Machine Learning

➤ 이미 알고 있는 데이터 (학습 데이터)로
모델을 생성해내는 과정

✓ 데이터에서 패턴을 추출하여 스스로 지식을 획득



기계 학습



- ◆ 과거에는 $F(x)$ 를 만드는데 집중
- ◆ 머신 러닝은 알고있는 데이터 x 와 결과 y 로 $F(x)$ 를 만들어 내는 것

F(x) 어디까지?

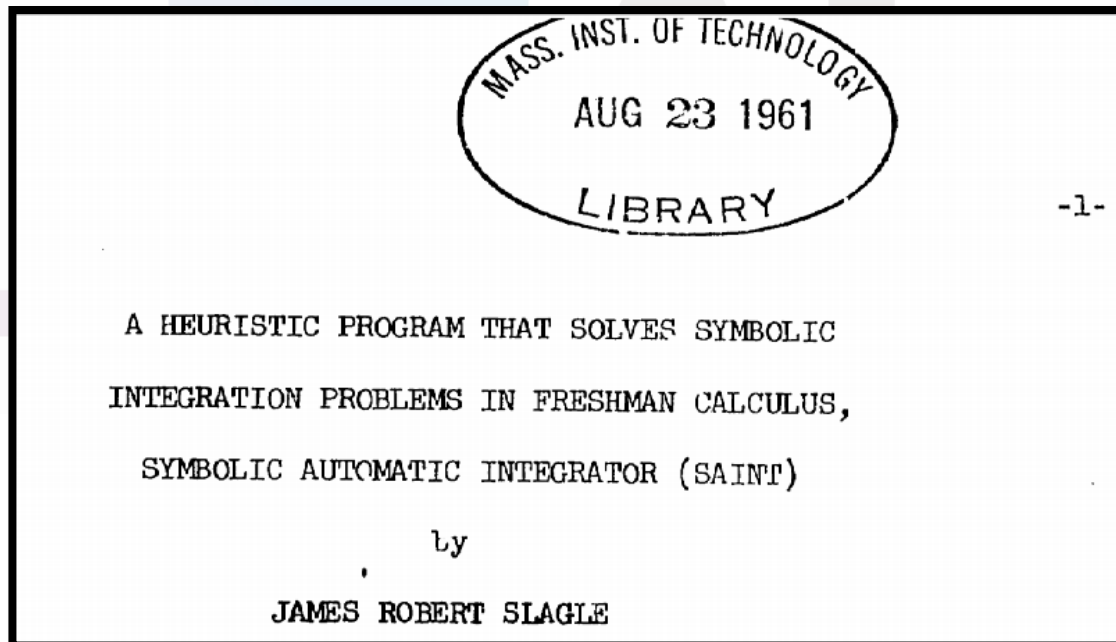
◆ 다음 적분 문제를 풀어봅시다

$$\int \frac{x^4}{(1-x^2)^{5/2}} dx$$

◆ 만약 컴퓨터가 이 문제를 푼다면?

Saint

- ◆ “A heuristic program that solves symbolic integration problems in freshman calculus,
symbolic **a**utomatic **i**ntegrator (Saint)



기계 학습 준비

◆ 어떤 문제를 머신 러닝으로 풀고 싶다면

➤ 어떤 부류의 문제인지 파악

➤ **데이터 세트**

- ✓ 학습 데이터
- ✓ 테스트 데이터
- ✓ Optional – 검증 데이터

➤ **모델을 설계**

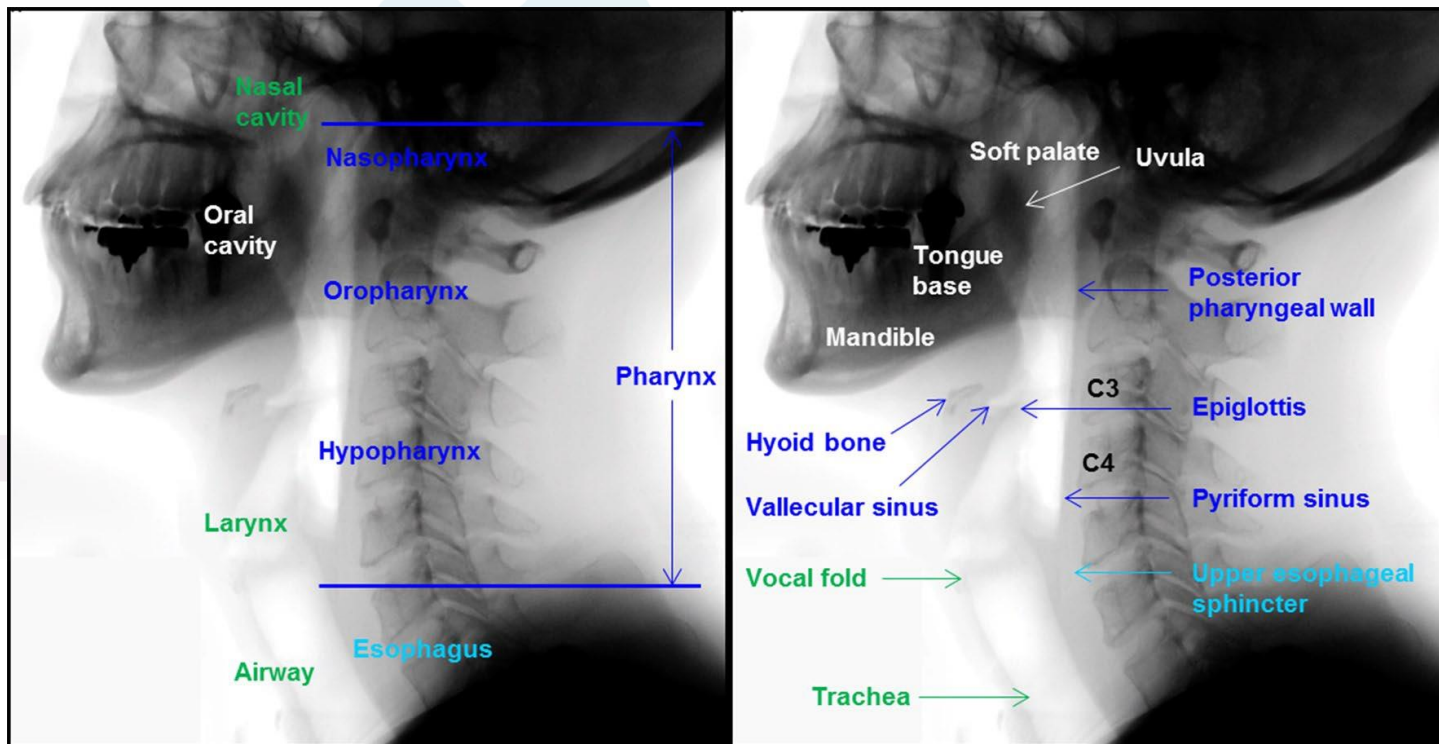
- ✓ 알고리즘



Domain Knowledge

◆ Engineer 는 엔지니어...?

- 이 사진은 뭐죠? → 논문
- Kaggle에서 제공되는 것이 뭐였죠?



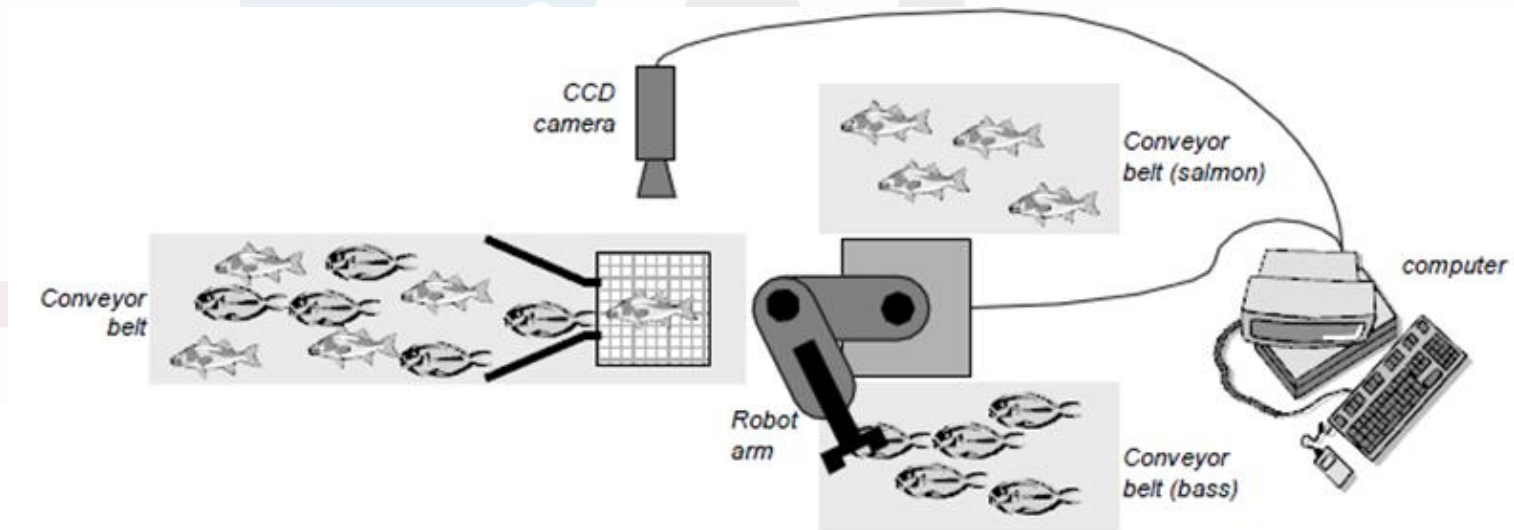
여러분들은

- ◆ 여러분의 전문 분야에서
혹은 전문분야가 아니더라도
인공지능이나, 머신 러닝을 이용하여
풀고 싶었던 문제가 있나요?
- ◆ Domain knowledge >>>> 기계학습 알고리즘

기계 학습 예제

◆ 시나리오

- 생선처리 공장에서 연어(salmon), 농어(sea bass)를 분류
- CCD 카메라를 갖춘 비전 시스템
- 영상을 분석하여 로봇 암을 제어하여 생선을 이동



기계 학습 설계

◆ 데이터 수집



◆ 전처리



◆ 특징?

➤ 길이, 밝기 ... Domain 지식이 없으니까...

◆ 분류기 설계

➤ 모델 선정, 분류기 학습

◆ 성능 테스트

➤ 학습에 사용하지 않은 데이터 사용

Feature – 특징

◆ 구분 대상을 어떻게 표현해야 하는가?

- 연어와 농어를 2가지 특징으로 표현
- [특징1: 길이, 특징2: 밝기, ... 특징N: something]

◆ Feature

- 관찰 대상에게서 발견된 개별적이고 측정가능한 경험적 속성
- 독립적인 변수를 잘 선택하는 것이 성공적인 분류를 위해 중요

지도 학습

◆ 지도 학습

- 훈련 데이터로부터 하나의 함수를 유추해내기 위한 기계 학습의 한 방법
- 훈련 데이터는 일반적으로 입력 객체에 대한 속성을 벡터 형태로 표현
- 각각의 벡터에 대해 원하는 결과가 무엇인지 표시

첫 번째 물고기
데이터

첫 번째 물고기
클래스

$[X_{11} \ X_{12}]$
 $[X_{21} \ X_{22}]$
...
 $[X_{n1} \ X_{n2}]$

$Y_1 = \text{salmon}$
 $Y_2 = \text{salmon}$
...
 $Y_n = \text{bass}$

Decision
Tree 학습



Feature Space – 특징 공간

◆ Feature Space

- 특징 벡터를 표현하는 공간
- 특징의 개수에 따라 다차원 공간으로 구성

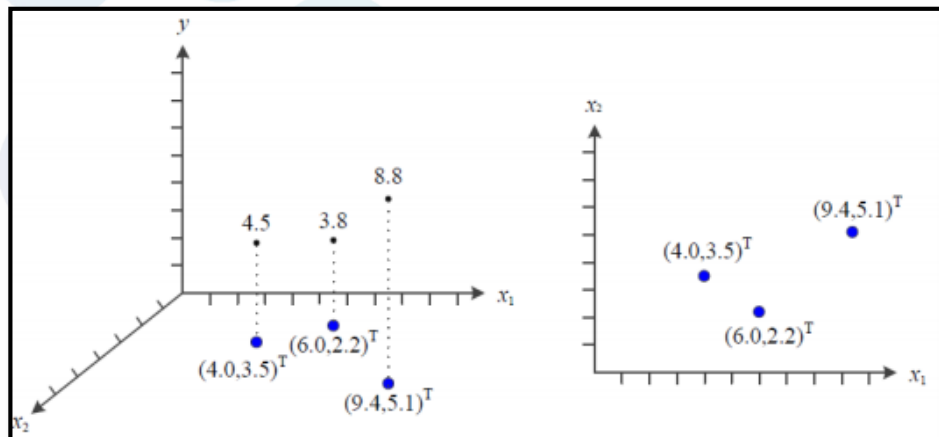
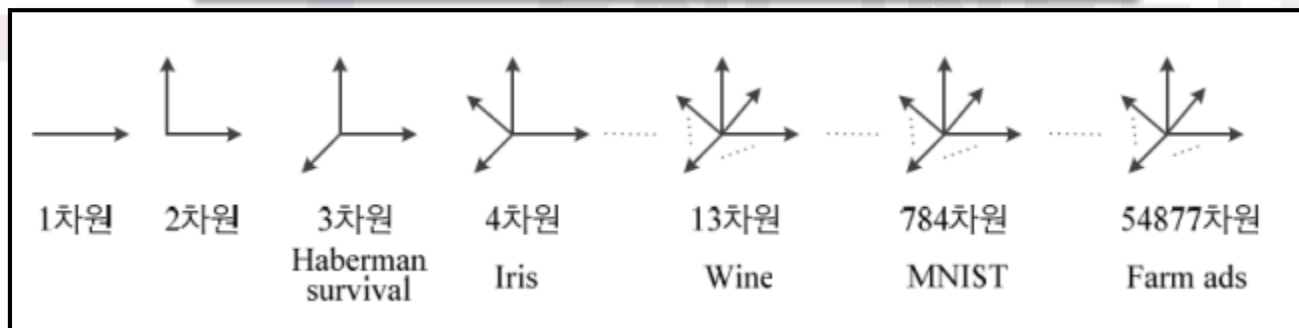


그림 출처: 기계학습(오일석 저)



Classification - 분류

◆ 특징 공간에서 Classification

- 특징 공간에서 대상을 분류할 수 있는 결정 경계를 구하는 것

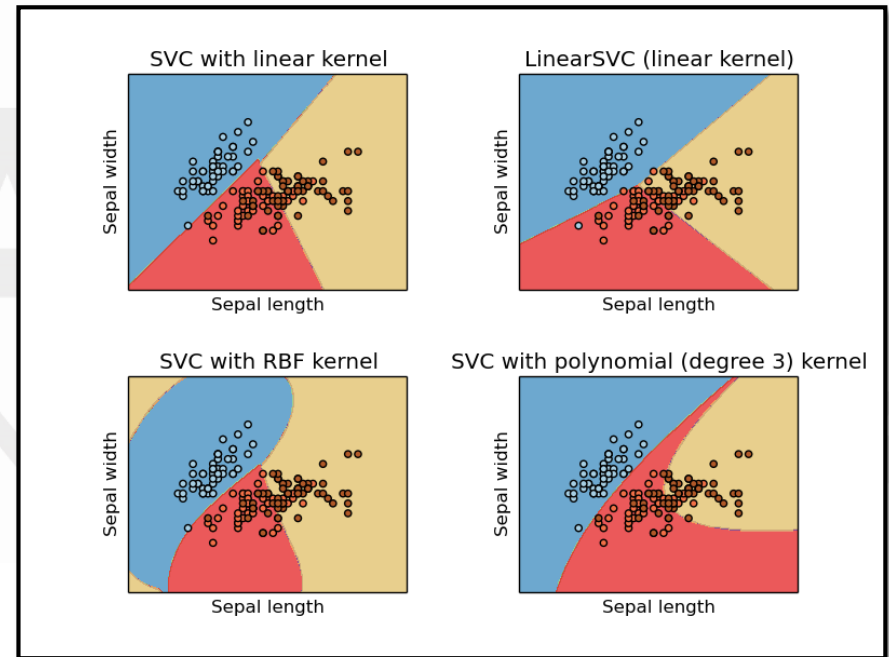
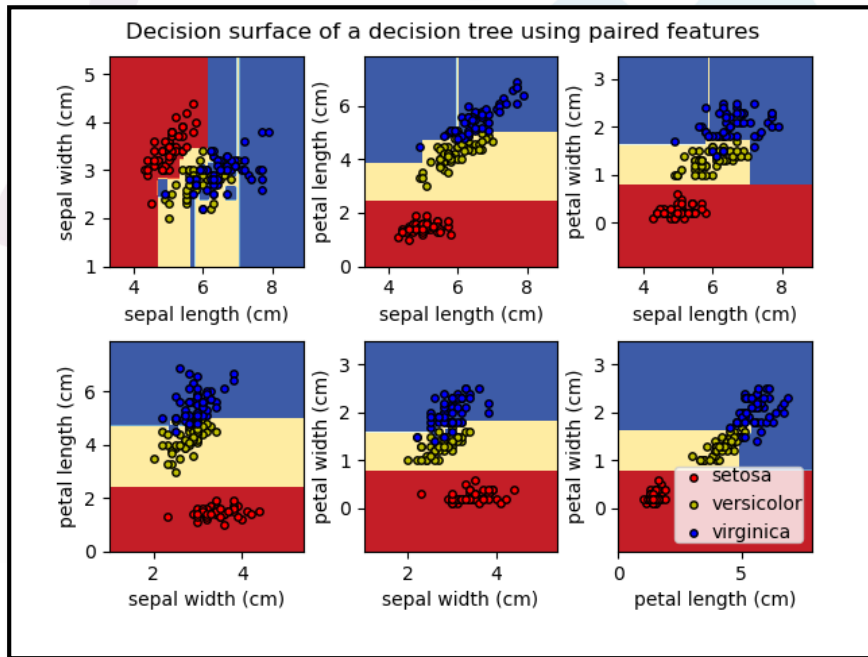


그림 출처: scikit-learn.org



한국폴리텍대학
대구캠퍼스

모델 설계

◆ Domain Knowledge

➤ 농어(sea bass)는 연어(salmon)보다 일반적으로 길다

◆ 선정 특징: Length

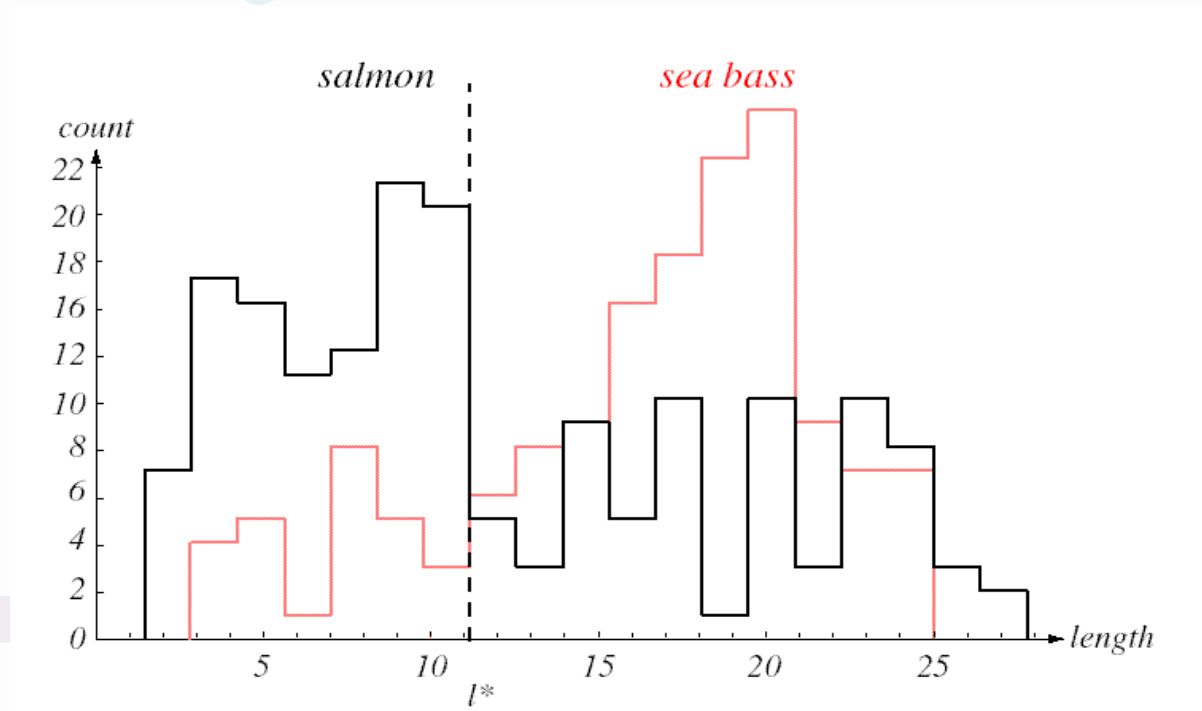
◆ 분류 규칙

If *Length* $\geq l^*$ then *sea bass*
otherwise *salmon*

◆ l^* 를 고르는 방법?

모델 학습

◆ 두 생선에 대한 Length 히스토그램



오분류가 제일 적은 지점

Training error: $90 / 316 = 28\%$



학습 결과

◆ 실험 결과

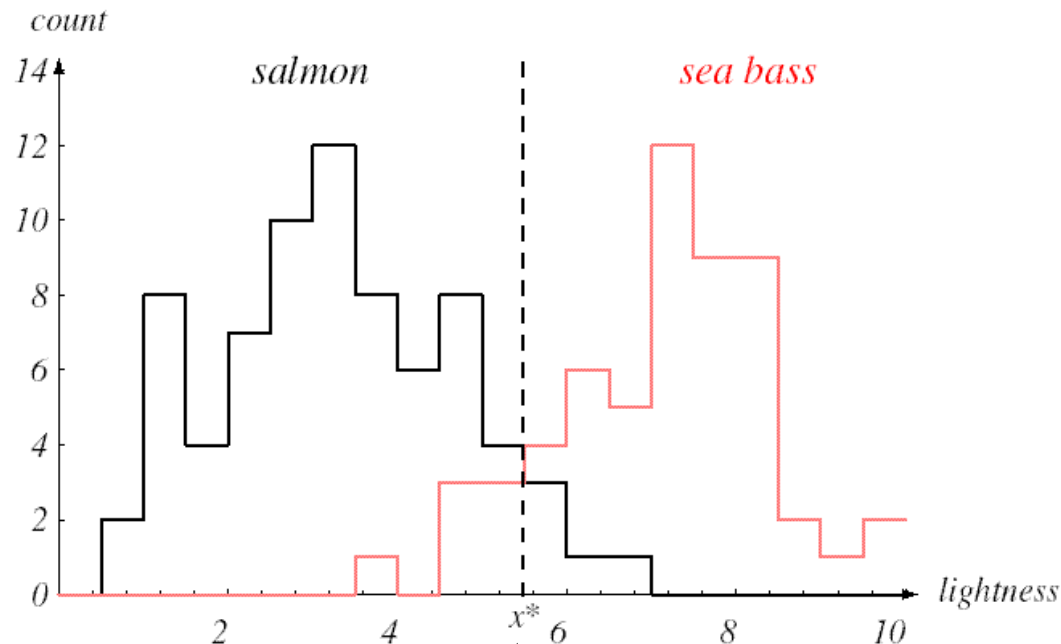
- 학습 데이터에 대한 분류율: 28%
- 너무 낮다!!!
- 다른 특징을 시도

◆ 밝기?

- New Feature → Lightness

모델 학습

◆ 두 생선에 대한 Lightness 히스토그램



오분류가 제일 적은 지점 Training error: $16 / 316 = 5\%$

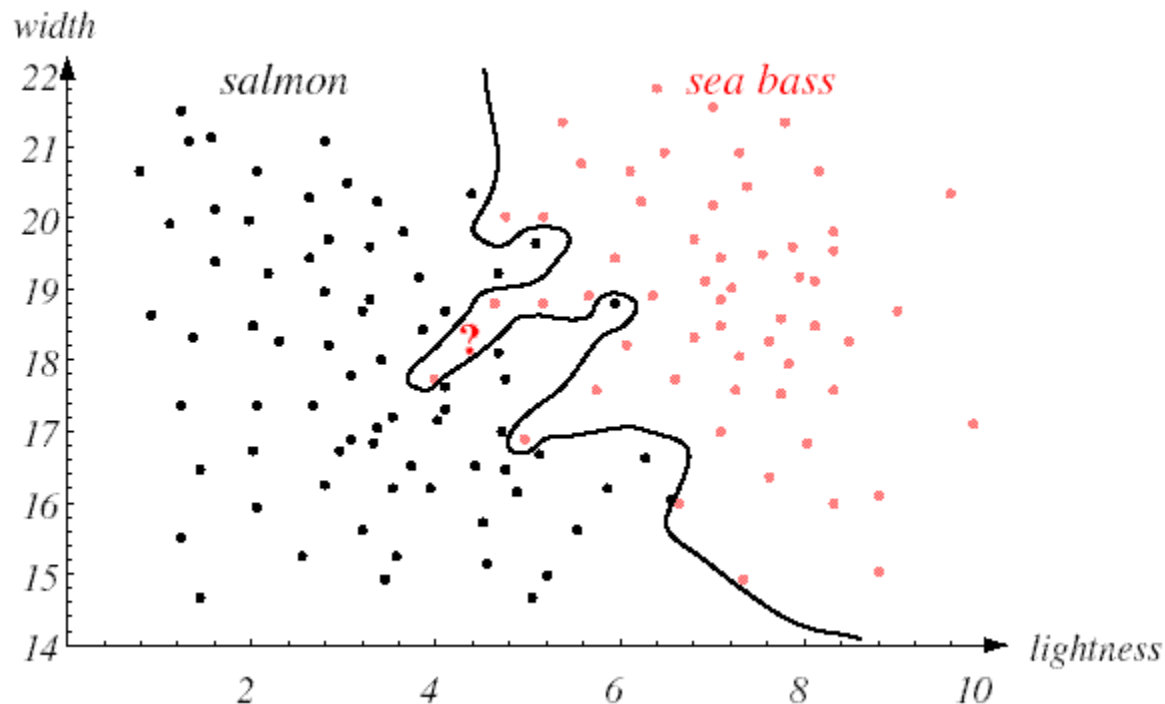
분류가 아까 보다 잘 되었음!

최선입니까?

- ◆ 단일 특징으로는 만족스럽지 못하다
- ◆ 복합 특징을 사용
 - Sea bass 가 salmon 보다 보통 폭이 넓다
- ◆ 일반적으로 특징 공간의 차원이 높을 수록
 - 분리에 유리
- ◆ 보다 복잡한 분류 모델을 사용할 필요

분류 함수를 좀 더 복잡하게?

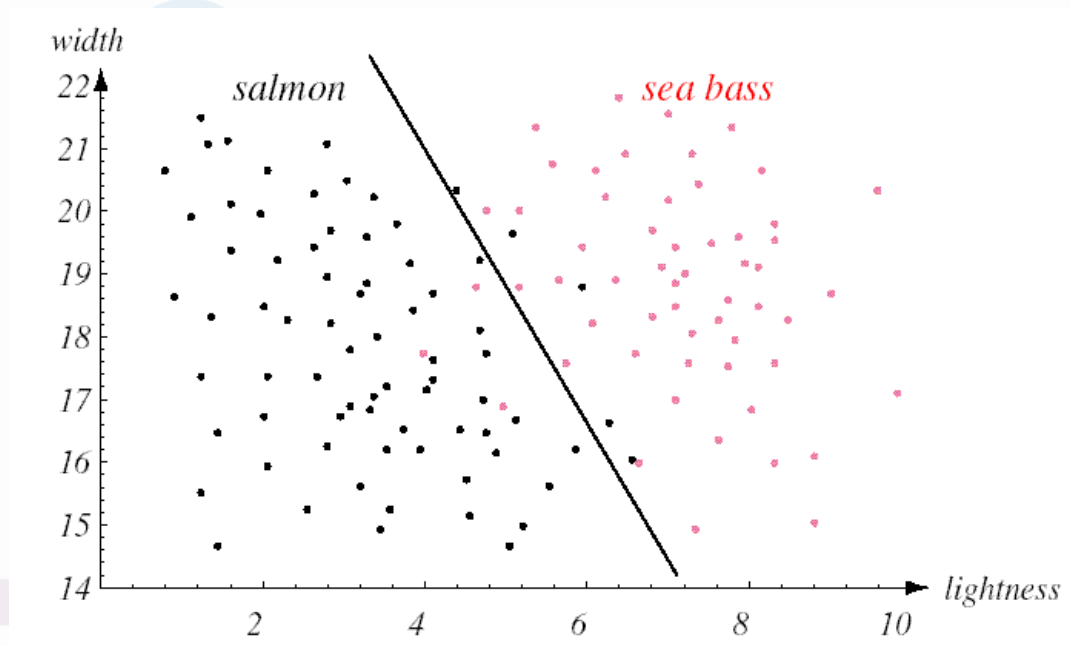
◆ 학습 데이터를 완벽하게 분류하는 모델



Complex decision function
Training error: $0 / 316 = 0\%$

일반화

◆ 앞 슬라이드의 분류 모델은 Overfitting



Linear decision function

Training error: $8 / 316 = 2.5\%$



Overfitting

◆ Overfitting 은 왜 생기나?

- 우리가 가지고 있는 데이터는 전체 데이터 중 얼마나 될까?
- 얼마나 일반화가 잘 되었는지는 어떻게 알지?
- 우리가 가지고 있는 데이터가 무진장 많다면?

모델 – ML Algorithm

◆ 분류

- 지도 학습 (Supervised Learning)
- 비지도 학습 (Unsupervised Learning)
- 강화 학습 (Reinforcement Learning)

◆ Algorithm

- Support Vector Machine (SVM), Bayesian Network, Decision Tree, Random Forest, Neural Network...

◆ Deep Neural Network (심층 신경망)



ML Application

◆ 컴퓨터 비전

- 컴퓨터에서 카메라 등의 시각 매체를 통해 입력 받은 영상을 분석하여 유용한 정보를 생성하는 기술
[ex. 보행자 검출, 얼굴 인식, 번호판 인식 등등]

◆ 데이터 마이닝

- 데이터 베이스 내에서 유용한 정보를 발견하는 기술
[ex. 상품 추천, 마케팅]

◆ 자연어 처리

- 컴퓨터를 이용해 사람의 자연어를 분석하고 처리하는 기술
- 대량의 말뭉치 데이터를 활용하는 기계 학습 기반의 자연어 처리 기법이 주류

타이타닉 생존자 예측 실습

- Kaggle 데이터를 다운로드하여 Pandas 라이브러리를 사용하여 살펴보고 시각화를 진행한다.
- 머신러닝을 사용하여 생존자를 예측한다.



머신 러닝 입문

◆ Here comes new challenger!

- Titanic – Machine Learning from Disaster
- <https://www.kaggle.com/c/titanic/overview>

◆ 타이타닉의 생존자를 예측하라

- kaggle 머신 러닝 입문 예제를 풀어보자

◆ Data 를 받아보자

- train.csv
- test.csv

Data Set 구조

Variable	Definition	Key
survival	Survival	0 = No, 1 = Yes
pclass	Ticket class	1 = 1st, 2 = 2nd, 3 = 3rd
sex	Sex	
Age	Age in years	
sibsp	# of siblings / spouses aboard the Titanic	
parch	# of parents / children aboard the Titanic	
ticket	Ticket number	
fare	Passenger fare	
cabin	Cabin number	
embarked	Port of Embarkation	C = Cherbourg, Q = Queenstown, S = Southampton

문제 정의

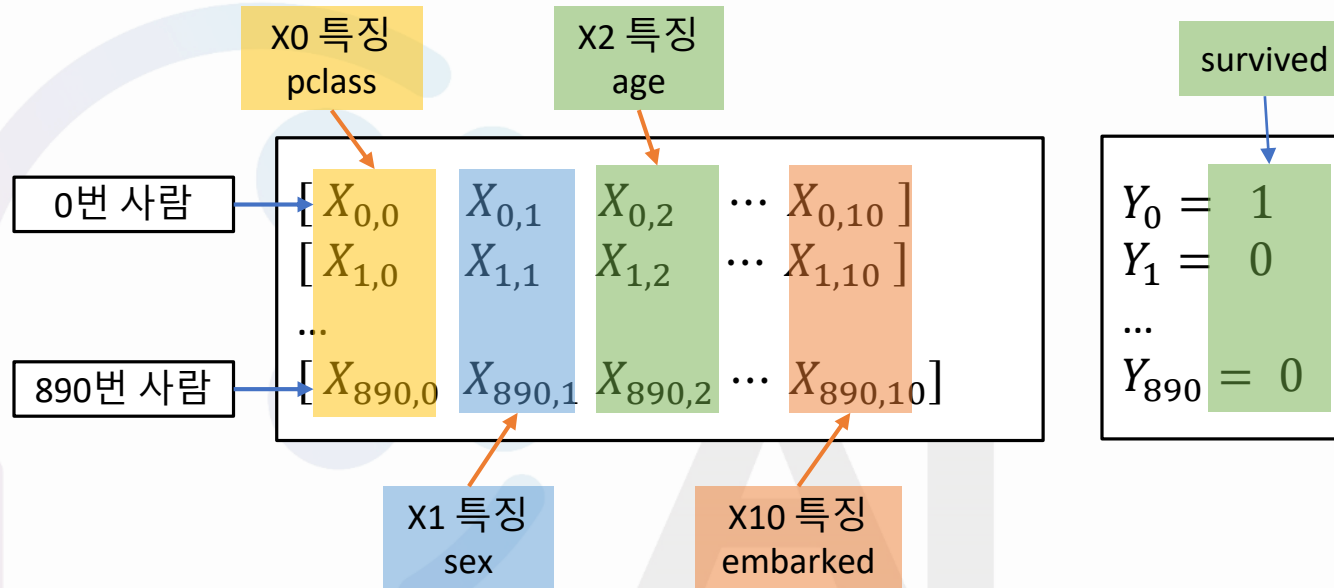
◆ train.csv 와 test.csv의 차이점?

- train = 12개의 column
- test = 11 개의 column
- Survived column 이 없다!

◆ 생존 여부를 알고 있는 891명의 데이터로

◆ 419명의 생존 여부를 맞춰보라!

Classification 으로 풀면 될 거 같은데?!



Machine
Learning

Train Set / Test Set

◆ Train Set (학습 데이터)

- 모델을 학습하기 위해 사용

◆ Test Set (테스트 데이터)

- 학습된 모델의 성능을 평가하기 위해 사용

◆ Test Set 까지 학습하면??

- 학습된 모델이 학습 데이터에 오버피팅 되었는지
- 잘 일반화 되었는지 알 수 없다

- 학습 데이터를 모두 맞춘다고 좋은 모델은 아니다

Pandas DataFrame



Pandas
소프트웨어

영어에서 번역됨 - 컴퓨터 프로그래밍에서 pandas는 데이터 조작 및 분석을 위해 Python 프로그래밍 언어로 작성된 소프트웨어 라이브러리입니다. 특히 숫자 테이블과 시계열을 조작하기 위한 데이터 구조와 연산을 제공합니다. 3 절 BSD 라이선스에 따라 출시 된 무료 소프트웨어입니다. 위키백과(영어)

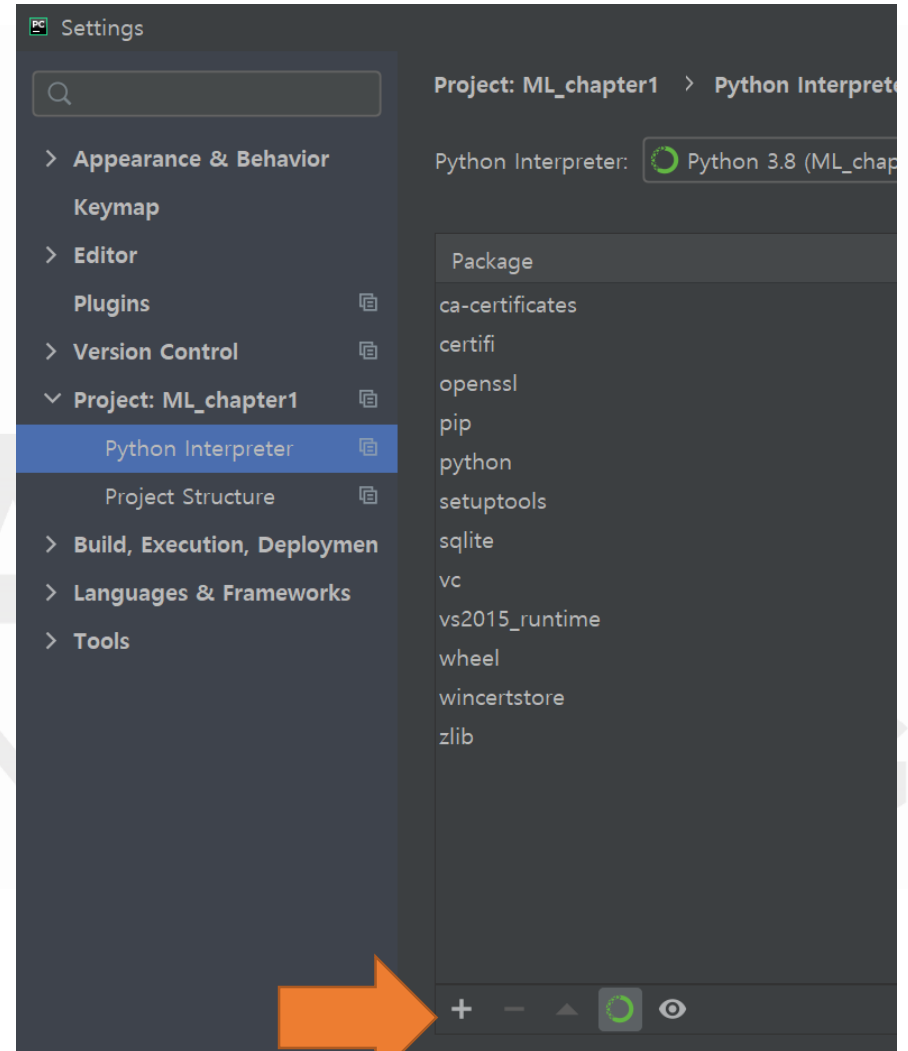
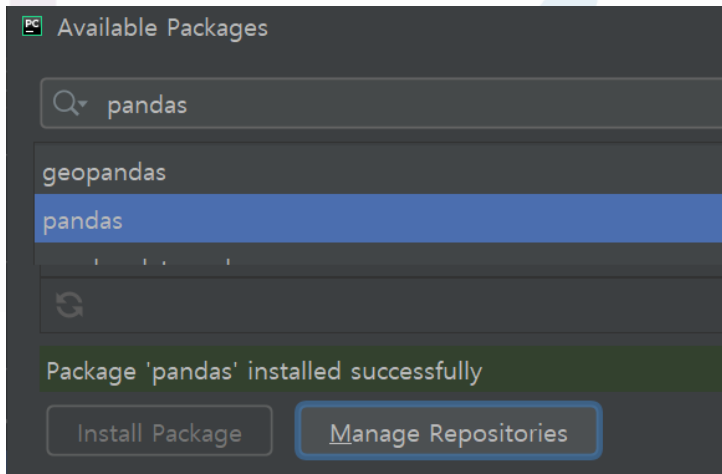
[원래 설명 보기](#) ▼

DataFrame

- 행/열 구조
- 가장 많이 활용되는 데이터 형태
- 스프레드 시트 형태

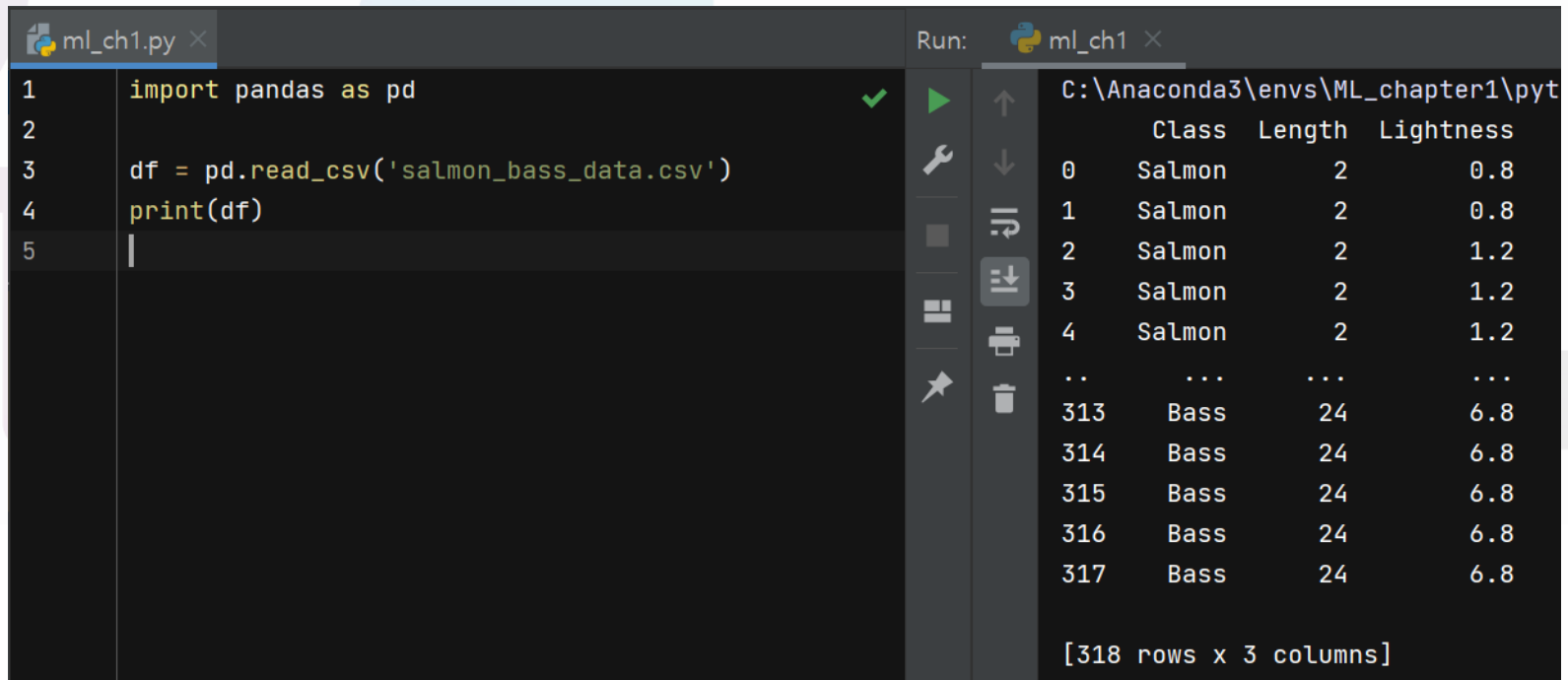
설정에서 Pandas 설치

- ◆ File → Settings
- ◆ Project:프로젝트이름
 - Python Interpreter → + 버튼 (insert)
 - Pandas 검색
 - Install Package



Pandas 동작 확인

- ◆ import pandas 가 동작하는지 확인
- ◆ pandas 내부 함수 들이 보이는지 확인
- ◆ csv 파일 로딩



The screenshot shows a Jupyter Notebook interface with a code cell on the left and a console output on the right. The code cell contains the following Python code:

```
1 import pandas as pd
2
3 df = pd.read_csv('salmon_bass_data.csv')
4 print(df)
5
```

The console output displays the result of the code execution, showing a DataFrame with 318 rows and 3 columns: Class, Length, and Lightness. The output is as follows:

	Class	Length	Lightness
0	Salmon	2	0.8
1	Salmon	2	0.8
2	Salmon	2	1.2
3	Salmon	2	1.2
4	Salmon	2	1.2
...
313	Bass	24	6.8
314	Bass	24	6.8
315	Bass	24	6.8
316	Bass	24	6.8
317	Bass	24	6.8

At the bottom of the console output, it indicates the dimensions of the DataFrame: [318 rows x 3 columns].

DataFrame

◆ DataFrame 으로 불러오기

```
import pandas as pd

df_train = pd.read_csv("titanic/train.csv")
df_test = pd.read_csv("titanic/test.csv")

print(df_train.head())
print(df_test.head())

print(df_train.info())
print(df_test.info())
```

정보가 비어 있는 칼럼도 있다.

1. Value_counts()

◆ 특정 칼럼의 도수의 값을 카운트

생존자 / 사망자 숫자

```
# value count로 확인  
df_train["Survived"].value_counts()
```

0 549

1 342

Name: Survived, dtype: int64

사망자

생존자

```
df_train["Sex"].value_counts()
```

male 577

female 314

Name: Sex, dtype: int64

```
df_train["Pclass"].value_counts()
```

3 491

1 216

2 184

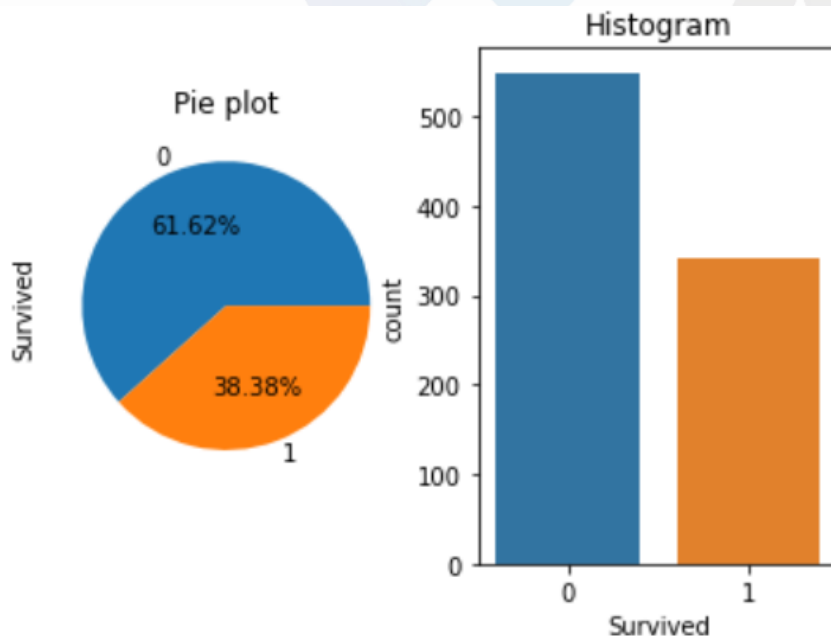
Name: Pclass, dtype: int64

2. Histogram / Pie Plot

```
# 파이플롯 / 카운터 플롯으로 시각화
# 그림은 1행 2열
fig, ax = plt.subplots(1,2)

df_train["Survived"].value_counts().plot.pie(autopct='%1.2f%%', ax=ax[0])
ax[0].set_title("Pie plot")

sns.countplot("Survived", data=df_train, ax=ax[1])
ax[1].set_title("Histogram")
plt.show()
```



3. Correlation

우리가 관심 있는 부분은
생존 여부

```
df_train.corr()
```

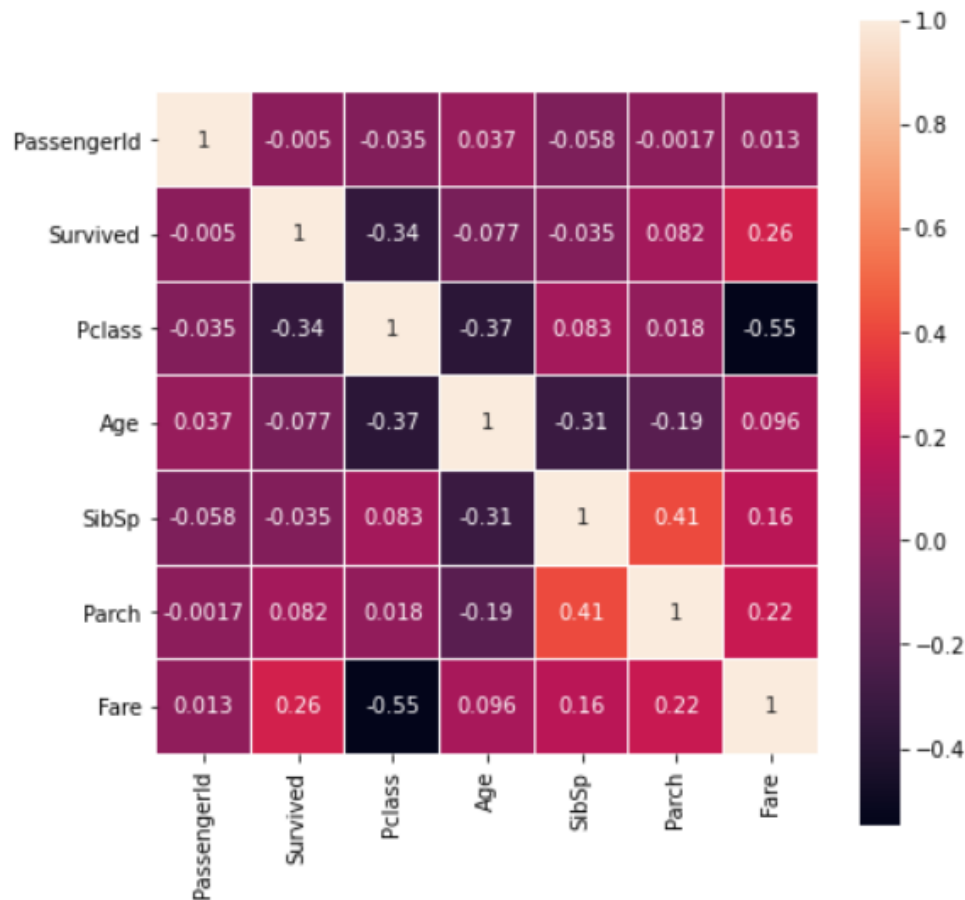
	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
PassengerId	1.000000	-0.005007	-0.035144	0.036847	-0.057527	-0.001652	0.012658
Survived	-0.005007	1.000000	-0.338481	-0.077221	-0.035322	0.081629	0.257307
Pclass	-0.035144	-0.338481	1.000000	-0.369226	0.083081	0.018443	-0.549500
Age	0.036847	-0.077221	-0.369226	1.000000	-0.308247	-0.189119	0.096067
SibSp	-0.057527	-0.035322	0.083081	-0.308247	1.000000	0.414838	0.159651
Parch	-0.001652	0.081629	0.018443	-0.189119	0.414838	1.000000	0.216225
Fare	0.012658	0.257307	-0.549500	0.096067	0.159651	0.216225	1.000000

Pclass, fare 가 어느정도
관련이 있어보임



3. Correlation – heatmap

```
# 상관도 시각화, seaborn의 heatmap 사용
plt.figure(figsize=(7, 7))
sns.heatmap(df_train.corr(), linewidths=0.01, annot=True, square=True)
plt.show()
```

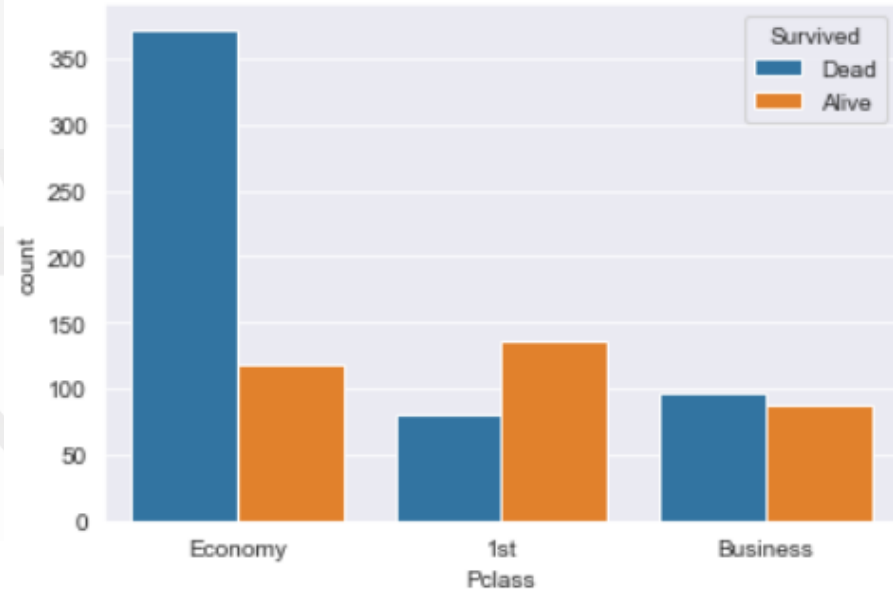


보기 좋게 꾸미기

```
df_train["Pclass"] = df_train["Pclass"]\
.replace(1, "1st")\
.replace(2, "Business")\
.replace(3, "Economy")

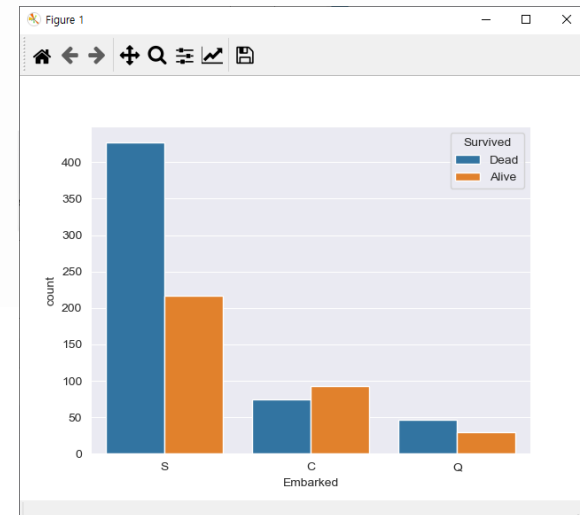
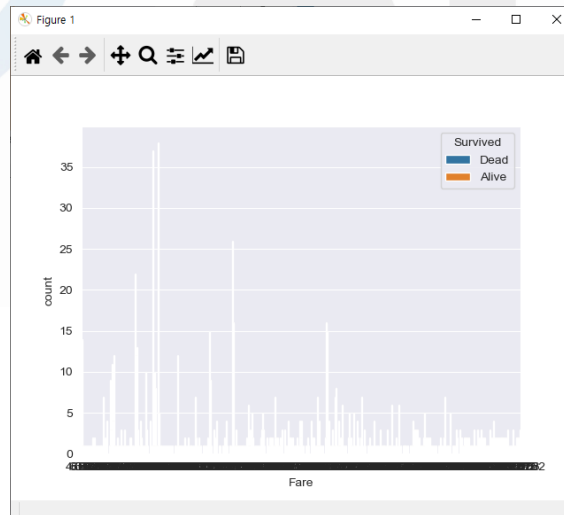
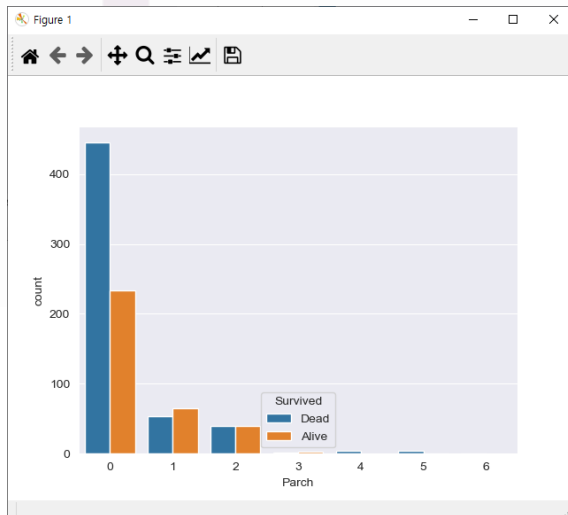
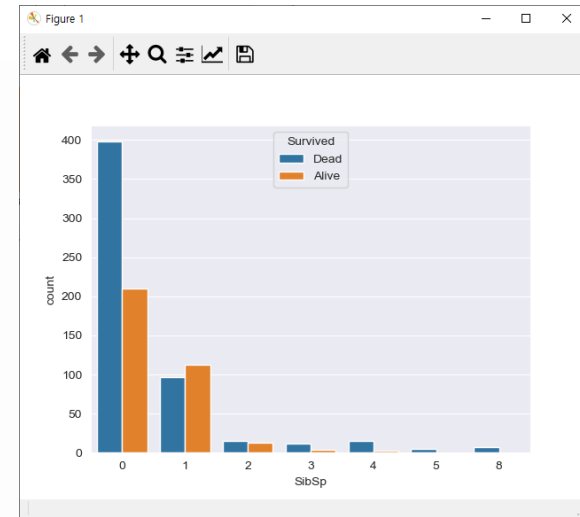
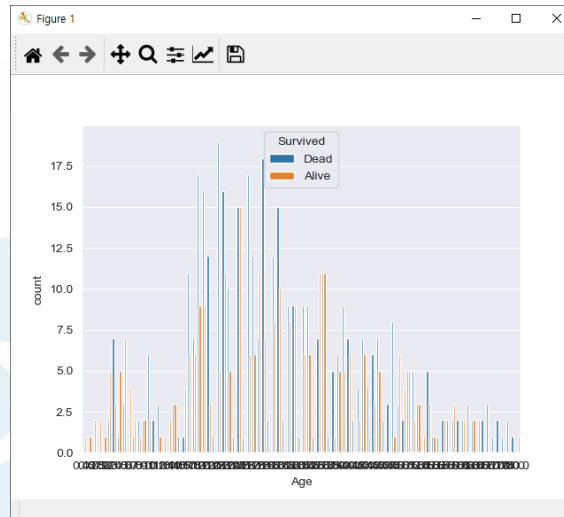
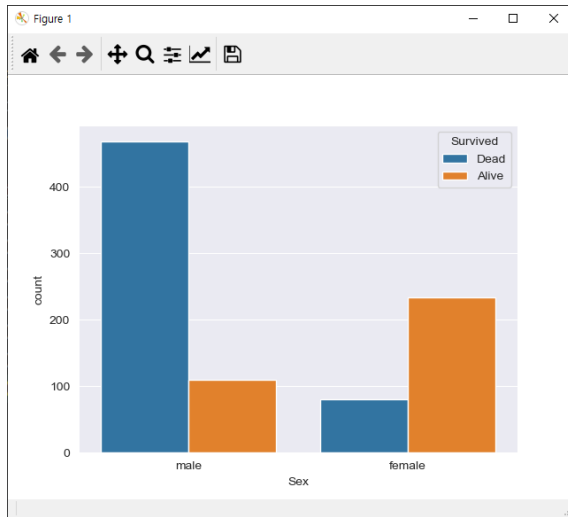
df_train["Survived"] = df_train["Survived"]\
.replace(1, "Alive")\
.replace(0, "Dead")

sns.set_style(style="darkgrid")
sns.countplot(data=df_train, x="Pclass", hue="Survived")
plt.show()
```



➤ 1등석 승객의 생존 비율이 높은 것을 확인

다른 칼럼들 생존자 숫자 시각화



그룹별 생존 비율로 코드를 바꾸어 보자

◆ 예 1) 성별에 따른 생존 비율

- 남성 생존자 수 / 남성 승객 수
- 여성 생존자 수 / 여성 승객 수

◆ 예 2) 좌석등급에 따른 생존 비율

- 1등급 생존자 수 / 1등급 승객 수
- 2등급 생존자 수 / 2등급 승객 수
- 3등급 생존자 수 / 3등급 승객 수

모듈화

◆ Subplot 을 사용한다.

생존자와 죽은자의 data frame을 분리 했다.

```
df_survive = df_train.loc[df_train["Survived"] == "Alive"]
```

```
df_dead = df_train.loc[df_train["Survived"] == "Dead"]
```

각 칼럼의 인원수 대비 생존자 비율

```
def show_group_rate(feature):
```

```
    sur_info = df_survive[feature].value_counts(sort=False)
```

```
    dead_info = df_dead[feature].value_counts(sort=False)
```

```
    fig = plt.figure()
```

```
    plt.title("Survival rete of " + feature)
```

```
    for i, index in enumerate(sur_info.index):
```

```
        fig.add_subplot(1, len(sur_info), i+1)
```

```
        plt.pie([sur_info[index], dead_info[index]], labels=["Survived", "Dead"], autopct='%0.1f%%')
```

```
        plt.title("Survial rate of " + index)
```

```
plt.show()
```

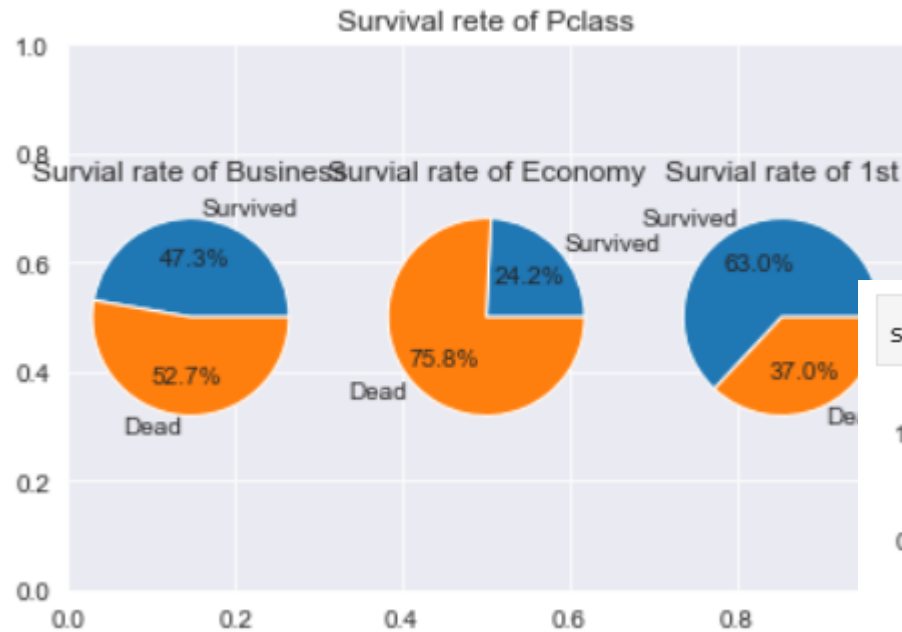
루프를 몇 번 돌았는지
확인하고자 할 때 사용

행, 열, 그림번호

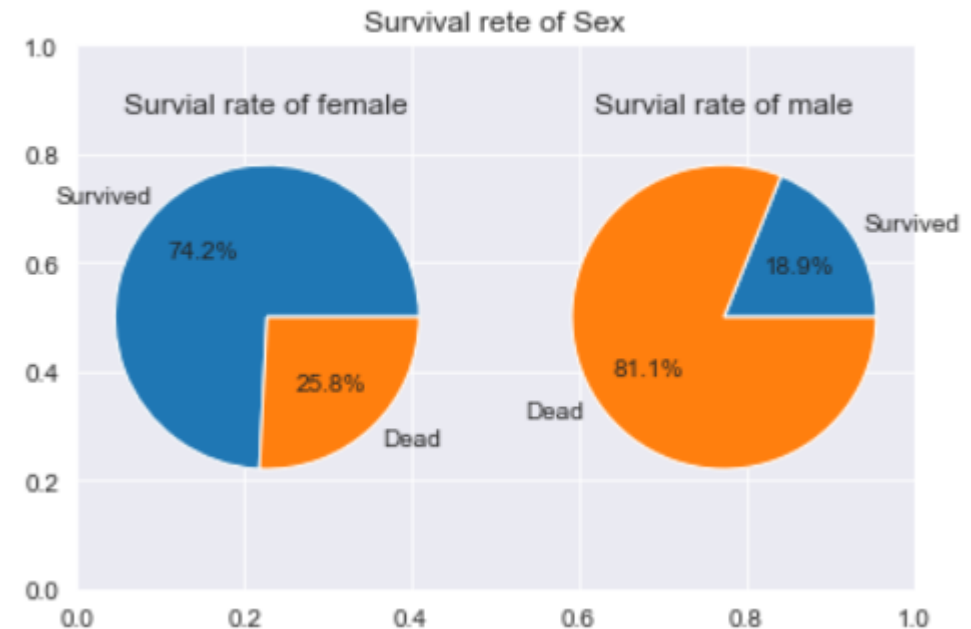


그룹별 생존율

```
show_group_rate("Pclass")
```



```
show_group_rate("Sex")
```



기계학습을 위한 전처리

◆ 기계 학습

- 데이터의 양, 학습 모델, 특징 등에 의해 영향

◆ 전처리

- 효율적인 학습을 위해 데이터를 가공
- 특징 선택 (Feature Selection)
- 노이즈 데이터 제거
- 차원 감소 (Demension Reduction)

비어 있는 값들 채우기

```
print(df_train.isnull().sum())
```

PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	177
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	687
Embarked	2
dtype:	int64

Data가 충분히 많다면,
비어 있는 데이터는 삭제하기도 함

```
# 비어있는 값을 채움
```

```
df_train["Age"] = df_train["Age"].fillna(df_train["Age"].mean())
```

```
# Embarked 는 가장 많은 S로 채웠다.
```

```
df_train["Embarked"] = df_train["Embarked"].fillna("S")
```


전처리

나이를 그룹별로 설정

```
for i in range(len(df_train)):
    age = int(df_train.loc[i, "Age"] / 10)
    df_train.loc[i, "Age"] = age
```

문자 데이터는 숫자로

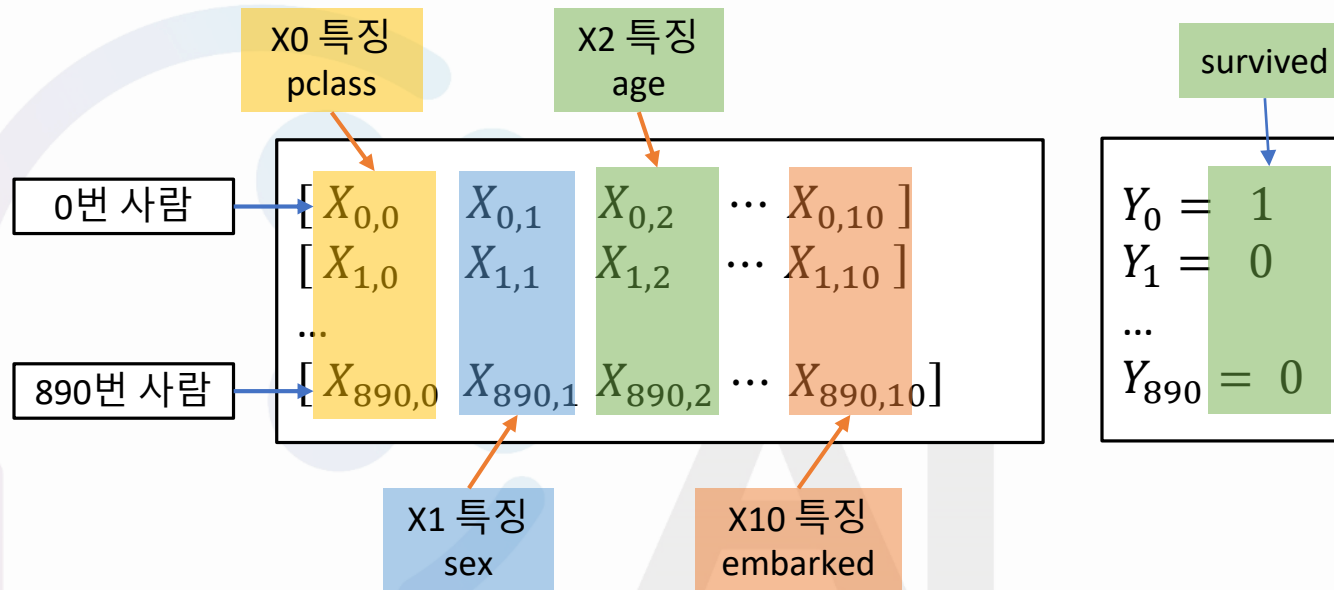
#문자 데이터를 숫자로 변환하기

```
df_train["Sex"] = df_train["Sex"].map({"male": 0, "female": 1})
df_train["Embarked"] = df_train["Embarked"].map({"Q": 0, "C": 1, "S": 2})
df_train["Pclass"] = df_train["Pclass"].map({"1st": 1, "Business": 2, "Economy": 3})
df_train["Survived"] = df_train["Survived"].map({"Alive": 1, "Dead": 0})
```

상관관계가 낮은 칼럼은 삭제

```
df_train = df_train.drop(["Name", "PassengerId", "Ticket", "Fare", "Cabin"], axis=1)
```

Train



Machine
Learning

Train

```
# 1. 분류기 생성. 여기서는 decision tree
#classifire = DecisionTreeClassifier()
classifire = RandomForestClassifier()
#classifire = LinearSVC()

# 2. 특징데이터와 정답데이터 분리
ground_truth = df_train["Survived"]
train_data = df_train.drop("Survived", axis=1)

# 3. 학습
classifire.fit(train_data, ground_truth)

# 4. 학습 결과 확인
print("Train Accuracy: ", round(classifire.score(train_data, ground_truth), 2))
```

Train Accuracy: 0.88

학습 데이터에 대한 정확도

Test

```
# 테스트 셋에 대한 결과 생성
df_test = pd.read_csv("data/test.csv")

# 1. 학습 데이터와 똑같은 형태로 만들어준다.
# 결과물 제출시 passengerId는 필요하므로 따로 빼두었다.
pId = df_test["PassengerId"]
df_test = df_test.drop(["Name", "PassengerId", "Ticket", "Fare", "Cabin"], axis=1)

df_test["Age"] = df_test["Age"].fillna(df_test["Age"].mean())
df_test["Embarked"] = df_test["Embarked"].fillna("S")
df_test["Sex"] = df_test["Sex"].map({"male": 0, "female": 1})
df_test["Embarked"] = df_test["Embarked"].map({"Q": 0, "C": 1, "S": 2})

# 2. 분류기에 넣고 돌린다.
test_result = classifier.predict(df_test)

# 3. 결과물을 만든다.

submit = pd.DataFrame({"PassengerId": pId, "Survived": test_result})
submit.to_csv("data/submit.csv", index=False)
```

Test Result

Your most recent submission

Name	Submitted	Wait time	Execution time	Score
submit.csv	2 days ago	1 seconds	0 seconds	0.72248

Complete

[Jump to your position on the leaderboard](#) ▼

1 submissions for [AIEngineering #2](#)

Sort by Most recent ▼

All Successful Selected

Submission and Description

Public Score

[submit.csv](#)

0.72248

2 days ago by [AIEngineering](#)

최초 dtree 학습 결과 테스트

결과가 왜 이래?!

◆ 이유 분석

- 딱히 전처리에서 해준 것이 없다.
- Decision Tree 로는 부족 하다.
- 다른 모델을 써 본다.

◆ 남들은 어떻게 하는가?

- <https://github.com/mdepero/titanic/blob/master/Final%20Report.ipynb>
- best random forest result: 79.904% accuracy