Titanic Data 분석을 통한 생존자 예측

(1)

목차 - Titanic Data 분석 (1)

- 데이터 불러오기, 데이터 확인
- 결측치 확인 및 대체
- 데이터 시각화 (Survived, Sex, Embarked, Parch, SibSp)
- 파생변수 생성 (Family size, Alone)
- Feature Engineering (One hot Encoding)
- Feature Selection
- 분류 모델 정의 (Logistic Regression)
- 교차검증(Cross Validation)
- Train, Test Data Split
- Model Train (Logistic Regression)
- Model Evaluation
- Other Models (Decision Tree, RandomForest)
- Predict (나의 생존여부 예측)

Titanic Data 분석을 위한 준비사항

파이썬 패키지 설치

- (시작메뉴) Anaconda Prompt 실행
- pip install numpy 입력 후 Enter (pandas 사용을 위해)
- pip install pandas 입력 후 Enter
- pip install matplotlib 입력 후 Enter
- pip install seaborn 입력 후 Enter
- pip install sklearn 입력 후 Enter

Titanic Data 분석

- Titanic Data를 분석하여 생존자와 사망자의 특징 분석
- 해당 Data를 토대로 생존여부를 예측하는 모델을 만들어보자
- 나의 생존여부도 예측해보자

(1) Data 불러오기, Data 확인

승선객 ID(**PassengerId**)
생존여부(**Survived**)
등급(**Pclass**)
이름(**Name**)
성별(**Sex**)
나이(**Age**)
형제/배우자 수(승선)(**SibSp**)
부모, 자식 수(승선)(**Parch**)
티켓(**Ticket**)
지불운임(**Fare**)
승객 선실(**Cabin**)
승선 항구(**Embarked**)

```
# 데이터분석을 위한 pandas, 시각화를 위한 matplotlib, seaborn을 불러옴
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# pd.read_csv(경로): 해당 csv를 읽어와 DataFrame Type으로 변환
# DataFrame명.head() : data의 상단을 보여준다(default는 5)

titanic_df= pd.read_csv('titanic.csv') # csv와 ipynb파일이 같은 폴더내에 있을 때 파일이름 only(상대경로)
titanic_df.head(3)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S

```
# DataFrame명.info() => Data의 개수, null 값의 개수, Data Type 확인
titanic_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          714 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Cabin        204 non-null    object
 11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

<Data Type>
Int: 정수(ex: 1, 2, 3..)
Object: 문자
Float: 실수(ex: 1.0, 2.5, 3.7 ..)

<null이란?> - 결측치
"**값이 없음**"
(ex: Age, Cabin의 경우 다른 열의 data 개수인 891보다 적다 – null값 존재)

=> 분석 또는 머신러닝 수행을 위해서는 null을 채우거나 제거해야 함

(2) 시각화 (Survived)

Survived

```
# DataFrame[원하는 column명].value_counts(): 값의 종류별로 도수를 보여준다
# 0은 사망자, 1은 생존자
titanic_df['Survived'].value_counts()
```

```
0    549
1    342
Name: Survived, dtype: int64
```

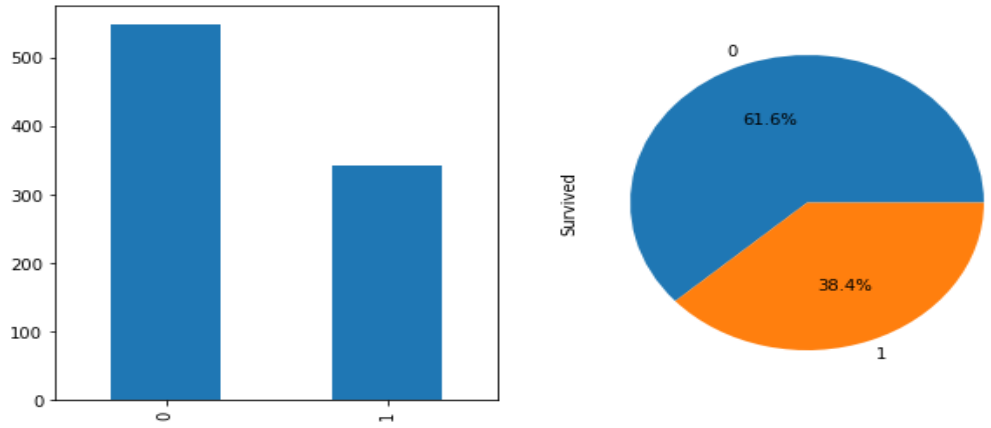
```
# plt.figure(figsize=(10,5)): 가로 10, 세로 5inch의 백지 생성
plt.figure(figsize=(10,5))
plt.subplot(1,2,1) # 백지를 1행 2열 공간으로 나누고 첫번째 자리에 그래프에 그려준다

titanic_df['Survived'].value_counts().plot(kind='bar') # 막대(bar)그래프

plt.subplot(1,2,2) # 1행 2열로 나눈 백지의 두번째 자리에 그래프 그려준다

titanic_df["Survived"].value_counts().plot(kind = "pie", autopct='%1.1f%%') # 파이(pie)그래프
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939de9fe08>
```



- 사망자가 생존자보다 많은 것을 확인

plt.figure(figsize = 10, 5): 가로10, 세로 5인치의 도화지

plt.subplot(1, 2, 1) 1행2열의 첫번째 그래프	plt.subplot(1, 2, 2) 1행2열의 두번째 그래프

(2) 시각화 (Embarked, Sex)

Sex

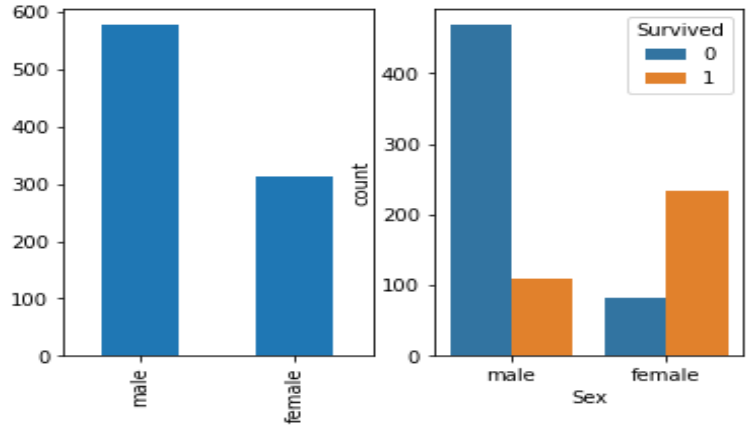
```
plt.subplot(1,2,1)

# 성별 분포에 대한 막대그래프
titanic_df["Sex"].value_counts().plot(kind = "bar")

plt.subplot(1,2,2)

# 성별에 따른 생존자, 사망자 분포
sns.countplot(data=titanic_df, x="Sex", hue="Survived")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939d1a9208>
```

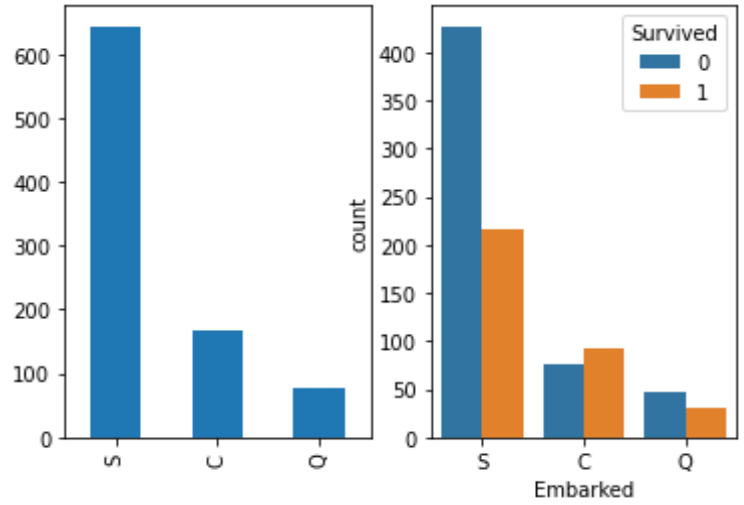


- **남성승객**이 여성승객보다 **많았다**
- **남성의 생존비율**이 여성보다 **극히 낮았다**

Embarked

```
plt.subplot(1,2,1)
titanic_df["Embarked"].value_counts().plot(kind = "bar")
plt.subplot(1,2,2)
sns.countplot(data=titanic_df, x="Embarked", hue="Survived")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939d21ff08>
```



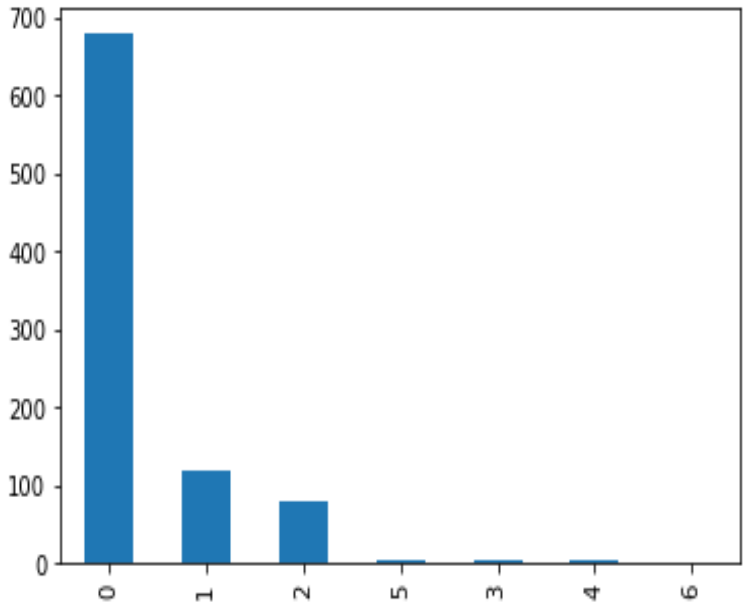
- **S 선착장** 탑승승객이 가장 많았으나 **사망자 수가 가장 높았다.**
- **Q 선착장** 탑승승객이 가장 적었으나 **사망자 수가 가장 적었다**

(2) 시각화 (Parch, SibSp)

Parch

```
titanic_df["Parch"].value_counts().plot(kind = "bar")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939df15208>
```

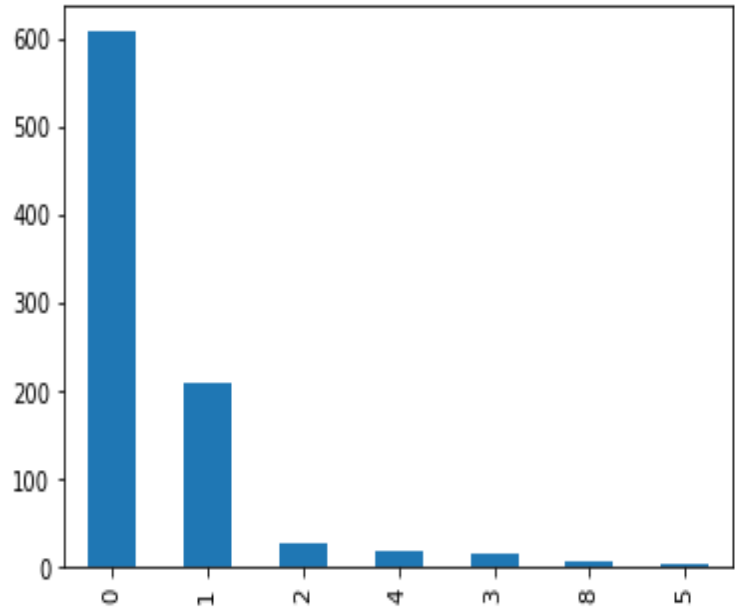


- **동행한 부모 또는 자식 수가 0, 1, 2가 많았다**
- **많은 가족이 동행**한 **승객**들도 있었다 **(5, 6)**

SibSp

```
titanic_df["SibSp"].value_counts().plot(kind = "bar")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939d992848>
```



- **형제나 배우자의 수**가 **0 또는 1**인 승객이 많았다
- **많은 가족이 동행**한 **승객**들도 있었다 **(5, 8)**

(3) 데이터 전처리 – Null 제거

결측치(null), **오류 데이터**(ex: 이상치), **모순된 데이터**(ex: 나이가 음수) **제거**, 또는 적절한 값으로 **대체**

```
titanic_df.isnull()
# null이면 True, not-null이면 False
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	False	False	False	False	False	False	False	False	False	False	True	False
1	False	False	False	False	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False	False	False	True	False
3	False	False	False	False	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False	False	False	True	False
...	...	...	...	...	...	...	...	...	...	...	...	...
886	False	False	False	False	False	False	False	False	False	False	True	False
887	False	False	False	False	False	False	False	False	False	False	False	False
888	False	False	False	False	False	True	False	False	False	False	True	False
889	False	False	False	False	False	False	False	False	False	False	False	False
890	False	False	False	False	False	False	False	False	False	False	True	False

```
titanic_df.isnull().sum()
# True(숫자 1), False(숫자 0)이므로
# 합계(sum)는 null data의 개수를 나타낸다
```

```
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
```

Age의 null 개수: 177

Cabin의 null 개수: 687

Embarked의 null 개수: 2

- **Cabin**의 결측치는 너무 많으므로 **변수에서 제거**하는 것이 적절
- **Age, Embarked**의 경우 결측치를 **적절한 값으로 대체**해보자!

(3) 데이터 전처리 – 결측치 채우기(Age)

```
# mean_age: Age의 평균값
mean_age = titanic_df['Age'].mean()
print(mean_age)
```

```
29.69911764705882
```

```
# fillna(x): x로 null값을 채운다(fill)
titanic_df['Age'] = titanic_df['Age'].fillna(mean_age)
```

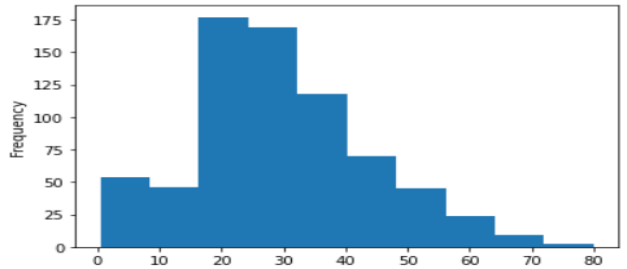
BEFORE

```
# 결측치 채우기 전
# Age의 값이 null인 data만 조회
titanic_df[titanic_df['Age'].isnull()]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emb
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	
17	18	1	2	Williams, Mr. Charles Eugene	male	NaN	0	0	244373	13.0000	NaN	
19	20	1	3	Masselmani, Mrs. Fatima	female	NaN	0	0	2649	7.2250	NaN	
26	27	0	3	Emir, Mr. Farred Chehab	male	NaN	0	0	2631	7.2250	NaN	
28	29	1	3	O'Dwyer, Miss. Ellen	female		0	0	330959	7.8792	NaN	

```
# 결측치 채우기 이전
titanic_df['Age'].plot(kind='hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939e088248>
```



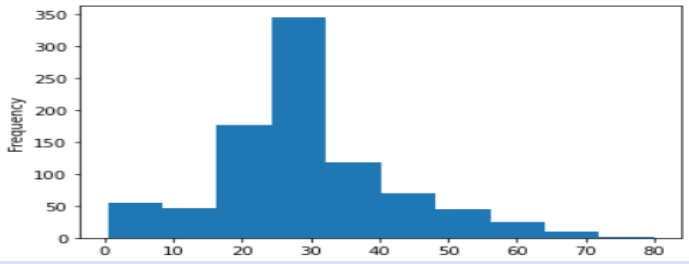
AFTER

```
titanic_df[titanic_df.index.isin(age_before.index)]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
5	6	0	3	Moran, Mr. James	male	29.699118	0	0	330877	8.4583	NaN
17	18	1	2	Williams, Mr. Charles Eugene	male	29.699118	0	0	244373	13.0000	NaN
19	20	1	3	Masselmani, Mrs. Fatima	female	29.699118	0	0	2649	7.2250	NaN
26	27	0	3	Emir, Mr. Farred Chehab	male	29.699118	0	0	2631	7.2250	NaN
28	29	1	3	O'Dwyer, Miss. Ellen	female	29.699118	0	0	330959	7.8792	NaN

```
# 결측치 채운 후
titanic_df['Age'].plot(kind='hist')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x2939e16dbc8>
```



177개의 null값을 Age평균값으로 대체

(3) 데이터 전처리 – 결측치 채우기(Embarked)

```
titanic_df[['Embarked']].mode()
```

	Embarked
0	S

Embarked의 최빈값 : '**S**'

```
# 최빈값('S')로 Embarked의 null값 대체
mode_em = titanic_df['Embarked'].mode
titanic_df['Embarked'] = titanic_df['Embarked'].fillna('S')
```

BEFORE

```
em_before = titanic_df[titanic_df['Embarked'].isnull()]
em_before.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
61	62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	NaN
829	830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	NaN

AFTER

```
# 결측치 채운 후
# Embarked의 값이 null이었던 data 조회
titanic_df[titanic_df.index.isin(em_before.index)]
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
61	62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	S
829	830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	S

2개의 null값을 최빈값인 'S'로 대체

(4) 파생변수 생성 – FamilySize, Alone

기존에 존재하는 변수들로 **새로운 유용한 변수**를 생성

FamilySize

SibSp + Parch +1
형제배우자 + 부모아이 + 자기자신

```
# FamilySize라는 새로운 변수 column 생성
# 형제, 배우자 수 + 부모, 자식 수 + 1(혼자일 경우 1로 표시)
titanic_df['FamilySize'] = (titanic_df['SibSp'] + titanic_df['Parch'] + 1)
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	FamilySize
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	2
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	1

Alone

FamilySize = 0 일 때 1

```
# 'Alone' column 생성하고 값을 0으로 초기화
titanic_df['Alone'] = 0

# FamilySize가 1인 승객들은 Alone 값을 1로 변경
titanic_df.loc[titanic_df['FamilySize'] == 1, 'Alone'] = 1
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	FamilySize	Alone
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S	2	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C	2	0
2	3	1	3	Heikkinen, Miss.	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S	1	1

FamilySize=0 일 때
Alone = 1

(5) Feature Engineering – One Hot Encoding

```
x1 = pd.get_dummies(titanic_df['Pclass']) #원핫인코딩 실시
x2 = pd.get_dummies(titanic_df['Sex'])
x3 = pd.get_dummies(titanic_df['Embarked'])

data = pd.concat([titanic_df, x1, x2, x3], axis=1)
```

```
x3.head(3)
```

	C	Q	S
0	0	0	1
1	1	0	0
2	0	0	1

```
data.head(3)
```

ived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	...	FamilySize	Alone	1	2	3	female	male	C	Q	S
0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	...	2	0	0	0	1	0	1	0	0	1
1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	...	2	0	1	0	0	1	0	1	0	0
1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	...	1	1	0	0	1	1	0	0	0	1

One Hot Encoding이란?

명목형 변수를 1과 0으로 이루어진 수치형 값으로 변환

Embarked : S, Q, C

S: 0 0 1
Q: 0 1 0
C: 1 0 0

Why?

- 크기비교나 연산의 의미가 없는 명목형 변수들의 값을 수치로 나타낼 수 있다
- **Machine Learning**을 위해서는, 모든 data를 **수치형**으로 변환해야

(6) Feature Selection

생존여부 예측에 **중요하지 않은 변수**를 제외

```
drop_cols = ['PassengerId','Name',"Ticket",'Cabin','Survived','Pclass','Sex', 'Embarked']
X = data.drop(drop_cols, axis=1)
y = data['Survived']
```

```
X.head(3)
```

	Age	SibSp	Parch	Fare	FamilySize	Alone	1	2	3	female	male	C	Q	S
0	22.0	1	0	7.2500	2	0	0	0	1	0	1	0	0	1
1	38.0	1	0	71.2833	2	0	1	0	0	1	0	1	0	0
2	26.0	0	0	7.9250	1	1	0	0	1	1	0	0	0	1

```
y.head(3)
```

```
0    0
1    1
2    1
Name: Survived, dtype: int64
```

분석에 유용한 변수들을 선택 후
유용하지 않은 변수 제거

⇒ 승선객ID, 이름, 티켓종류, 숙박선실, 생존여부(종속변수로 설정하기 위해)

⇒ One Hot Encoding하기 전 변수들(등급, 성별, 선착장)

독립변수(X)와 종속변수(Y) 설정

⇒ 종속변수(Y): 예측할 변수(Survived – 생존여부)

⇒ 독립변수(X): 예측을 위해 사용될 변수들(나이, 성별..)

Summary

- 데이터 불러오기, 데이터 확인 (info, head, tail)
- 결측치 확인 및 대체 (isnull.sum, mean, mode)
- 데이터 시각화 (Matplotlib Seaborn)
- 파생변수 생성 (Family size, Alone)
- 명목변수 인코딩 (One Hot Encoding)
- 변수 선택 (drop)
- 분류 모델 정의 (Logistic Regression)
- 교차검증 (cross_val_score)
- Train, Test data split (train_test_split)
- Model Train (Logistic Regression)
- Model Evaluation (accuracy Score)
- Other Models (Decision Tree, RandomForest)
- Predict (나의 생존여부 예측)

수고하셨습니다!