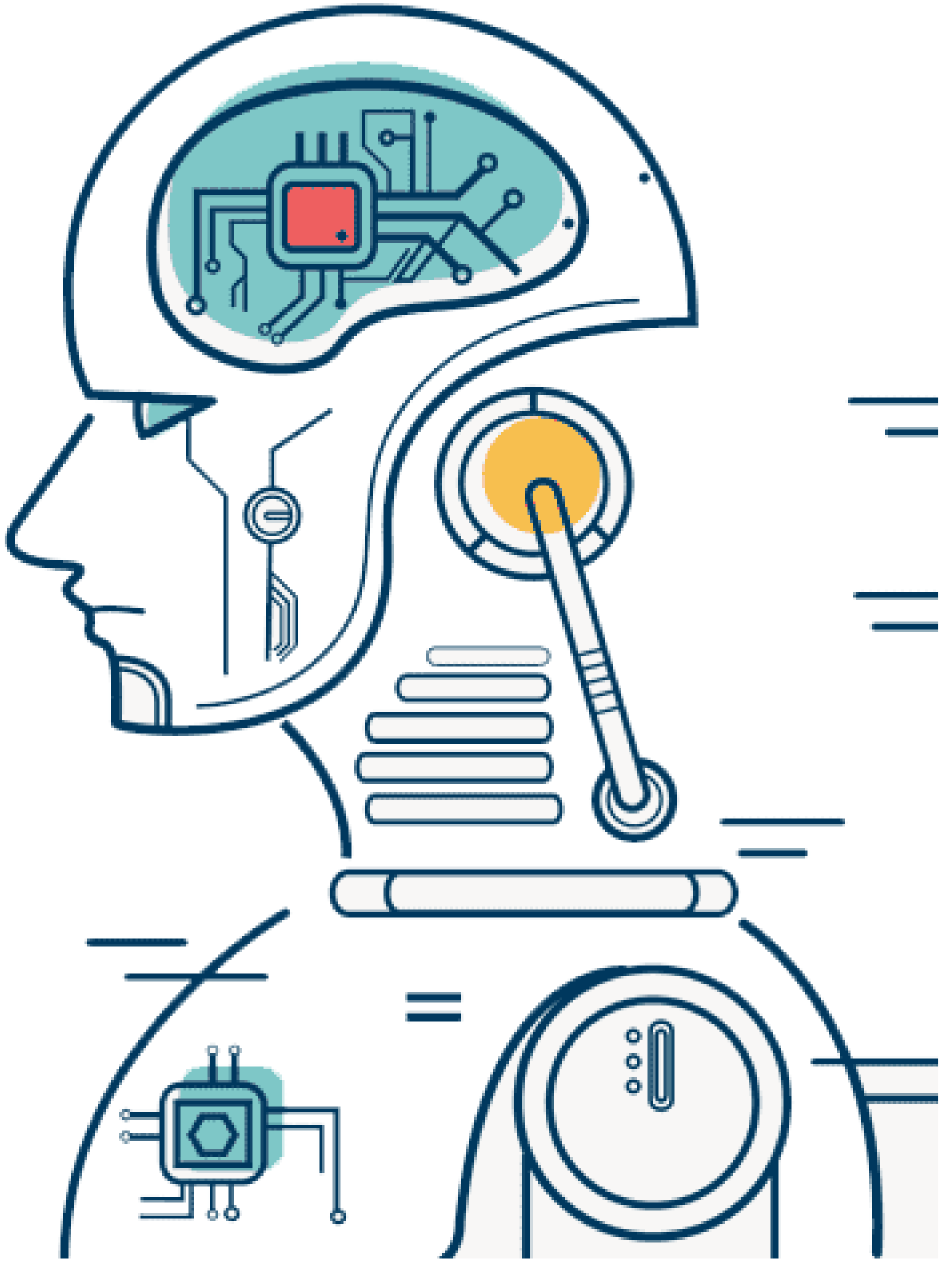
스마트인재개발원
Smart Human Resources Development

Machine Learning

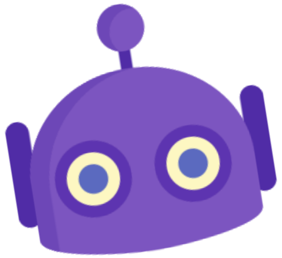
Chapter 4 지도학습
(타이타닉 데이터 실습)



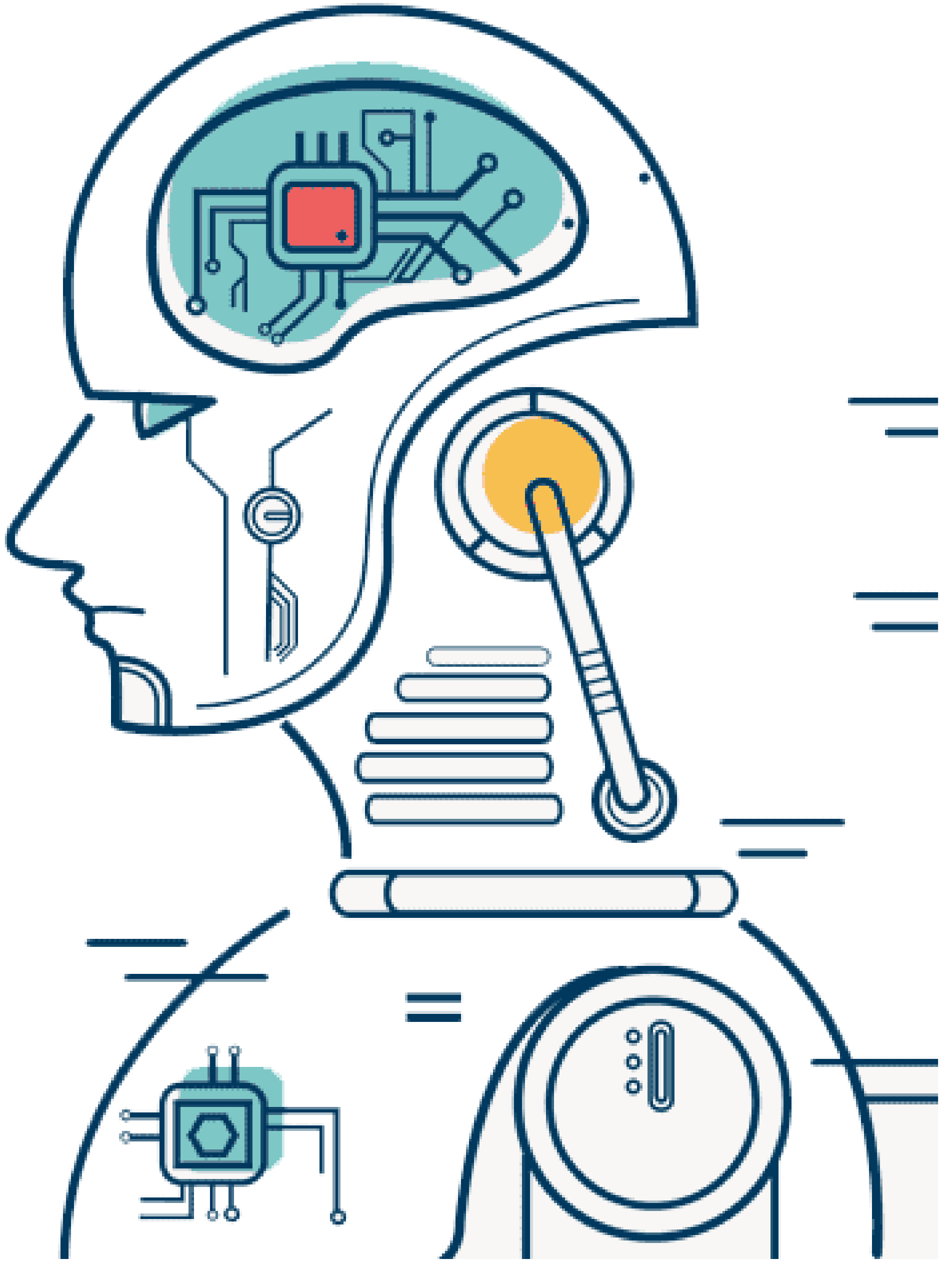
문제 정의 및 요구사항 분석

- 이전까지 학습한 내용을 적용할 수 있다.
- 다양한 특성 공학의 방법들을 활용할 수 있다.
- 탐색적 데이터 분석(EDA)을 수행할 수 있다.
- 머신러닝을 이용하여 타이타닉 데이터의 생존/사망자를 예측할 수 있다.

스마트인재개발원
Smart Human Resources Development



타이타닉 데이터를 이용한 머신러닝 학습 실습



데이터 수집

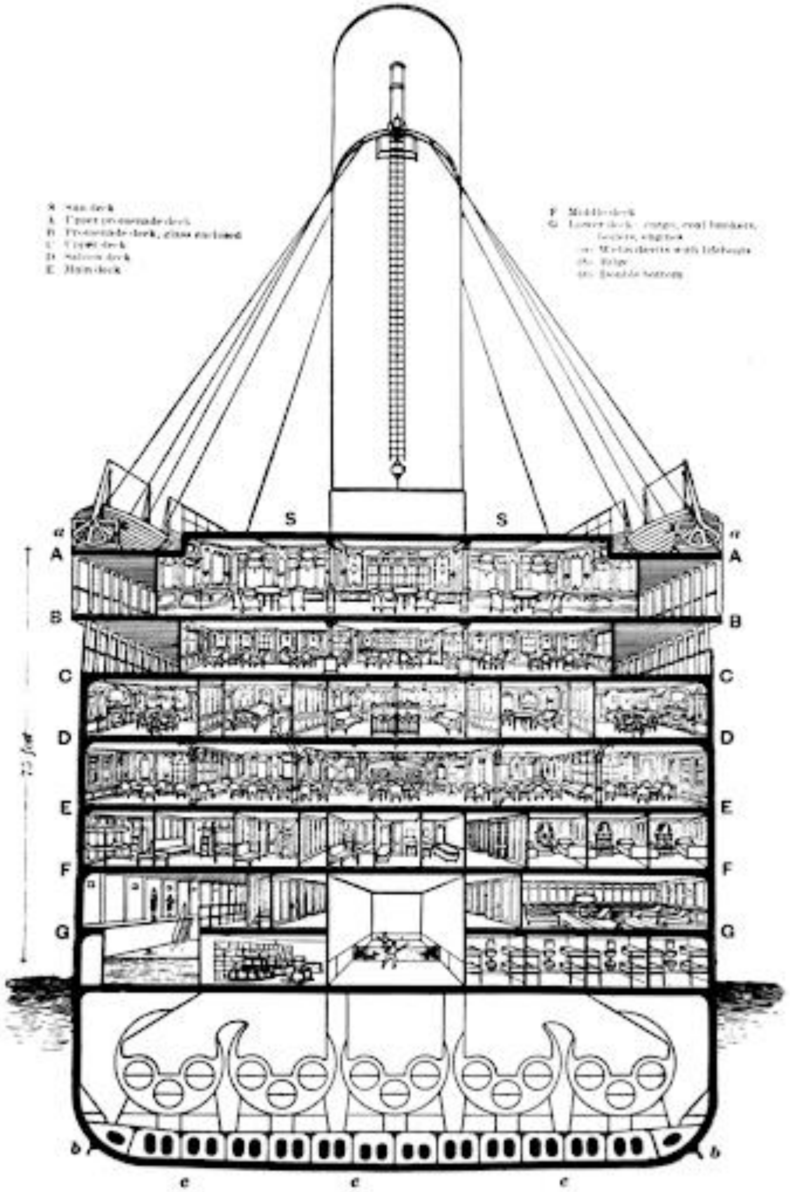
타이타닉 데이터 구조

- 훈련 데이터 891개, 테스트 데이터 418개
- 테스트 데이터에는 survived 컬럼이 없음
- 10개 특성으로 구성

feature	의미	설명	타입
Survivied	생존여부	target 라벨 (0 : 생존, 1 : 사망)	integer
Pclass	티켓의 클래스	1 = 1등석, 2 = 2등석, 3 = 3등석	integer
Sex	성별	male, female로 구분	string
Age	나이	0-80세	integer
SibSp	함께 탑승한 형제와 배우자의 수		integer
Parch	함께 탑승한 부모, 아이의 수		integer
Ticket	티켓 번호	alphabat + integer	integer
Fare	탑승료		float
Cabin	객실 번호	alphabat + integer	string
Embarked	탑승 항구	C = Cherbourg, Q = Queenstown, S = Southampton	string

데이터 수집

타이타닉 구조



데이터 전처리 (특성공학)

스마트인재개발원
Smart Human Resources Development

결측치(null data) 확인 – info() 함수

train 데이터

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
PassengerId    891 non-null int64
Survived       891 non-null int64
Pclass         891 non-null int64
Name           891 non-null object
Sex            891 non-null object
Age            714 non-null float64
SibSp          891 non-null int64
Parch          891 non-null int64
Ticket         891 non-null object
Fare           891 non-null float64
Cabin          204 non-null object
Embarked       889 non-null object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.6+ KB
```

test 데이터

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 418 entries, 0 to 417
Data columns (total 11 columns):
PassengerId    418 non-null int64
Pclass         418 non-null int64
Name           418 non-null object
Sex            418 non-null object
Age            332 non-null float64
SibSp          418 non-null int64
Parch          418 non-null int64
Ticket         418 non-null object
Fare           417 non-null float64
Cabin          91 non-null object
Embarked       418 non-null object
dtypes: float64(2), int64(4), object(5)
memory usage: 36.0+ KB
```

결측치(null data) 확인

train.isnull().sum()

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

이상치(outlier) / 간단한 통계 확인 – describe()

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

결측치 채우기 - Age

Pclass와 Sex 컬럼에 해당하는 Age의 평균을 계산

```
pt1 = train.pivot_table(values='Age',
                        index=['Pclass', 'Sex'],
                        aggfunc = 'mean')
pt1
```

Pclass	Sex	Age
1	female	34.611765
	male	41.281386
2	female	28.722973
	male	30.740707
3	female	21.750000
	male	26.507589

데이터 전처리 (특성공학)

apply() 함수 : 정의된 함수를 전체 데이터에 적용

```
컬럼명 = train.apply(함수명, axis=1).astype('int64')
```

데이터 전처리 (특성공학)

스마트인재개발원
Smart Human Resources Development

결측치 채우기 - Age

```
def fill_age(row):
    if np.isnan(row['Age']):
        return pt1.loc[row['Pclass'], row['Sex']]
    else:
        return row['Age']

train['Age'] = train.apply(fill_age, axis=1).astype('int64')
```

test 데이터의 Age 컬럼의 결측치를 피벗 테이블의 해당 값으로 채우고 결과 확인하기

결측치 채우기 - Embarked

결측치가 적으므로 가장 많은 수를 차지하는 클래스로 치환

train 데이터

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

```
train['Embarked'] = train['Embarked'].fillna('S')
```

결측치 채우기 - Fare

train 데이터의 Pclass와 Sex 컬럼에 해당하는 Fare의 평균을 계산

Pclass	Sex	Fare
1	female	106.125798
	male	67.226127
2	female	21.970121
	male	19.741782
3	female	16.118810
	male	12.661633

결측치 채우기 - Fare

test 데이터의 결측치가 있는 Pclass와 Sex 컬럼에 해당하는 Fare 값을 채우기

```
test[test['Fare'].isnull()]
```

	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
PassengerId										
1044	3	Storey, Mr. Thomas	male	60	0	0	3701	NaN	NaN	S

```
test['Fare'] = test['Fare'].fillna(12.661633)
```

데이터 전처리 (특성공학)

스마트인재개발원
Smart Human Resources Development

결측치 채우기 – Cabin

train['Cabin'].unique()

```
array([nan, 'C85', 'C123', 'E46', 'G6', 'C103', 'D56', 'A6',
       'C23 C25 C27', 'B78', 'D33', 'B30', 'C52', 'B28', 'C83', 'F33',
       'F G73', 'E31', 'A5', 'D10 D12', 'D26', 'C110', 'B58 B60', 'E101',
       'F E69', 'D47', 'B86', 'F2', 'C2', 'E33', 'B19', 'A7', 'C49', 'F4',
       'A32', 'B4', 'B80', 'A31', 'D36', 'D15', 'C93', 'C78', 'D35',
       'C87', 'B77', 'E67', 'B94', 'C125', 'C99', 'C118', 'D7', 'A19',
       'B49', 'D', 'C22 C26', 'C106', 'C65', 'E36', 'C54',
       'B57 B59 B63 B66', 'C7', 'E34', 'C32', 'B18', 'C124', 'C91', 'E40',
       'T', 'C128', 'D37', 'B35', 'E50', 'C82', 'B96 B98', 'E10', 'E44',
       'A34', 'C104', 'C111', 'C92', 'E38', 'D21', 'E12', 'E63', 'A14',
       'B37', 'C30', 'D20', 'B79', 'E25', 'D46', 'B73', 'C95', 'B38',
       'B39', 'B22', 'C86', 'C70', 'A16', 'C101', 'C68', 'A10', 'E68',
       'B41', 'A20', 'D19', 'D50', 'D9', 'A23', 'B50', 'A26', 'D48',
       'E58', 'C126', 'B71', 'B51 B53 B55', 'D49', 'B5', 'B20', 'F G63',
       'E121', 'D11', 'E77', 'F38', 'B3', 'D6', 'B82 B84', 'D17', 'A36',
       'B102', 'B69', 'E49', 'C47', 'D28', 'E17', 'A24', 'C50', 'B42',
       'C148'], dtype=object)
```

Cabin 번호는 첫 글자는 영문자로 나머지는 숫자로 구성

데이터 전처리 (특성공학)

결측치 채우기 – Cabin

(1) nan인 값을 영문자 M으로 채움

```
train['Deck']  = train['Cabin'].fillna('M')
```

(2) 첫 영문자만 잘라내어 Deck 컬럼에 저장

```
train['Deck'] = train['Deck'].str[0]
```

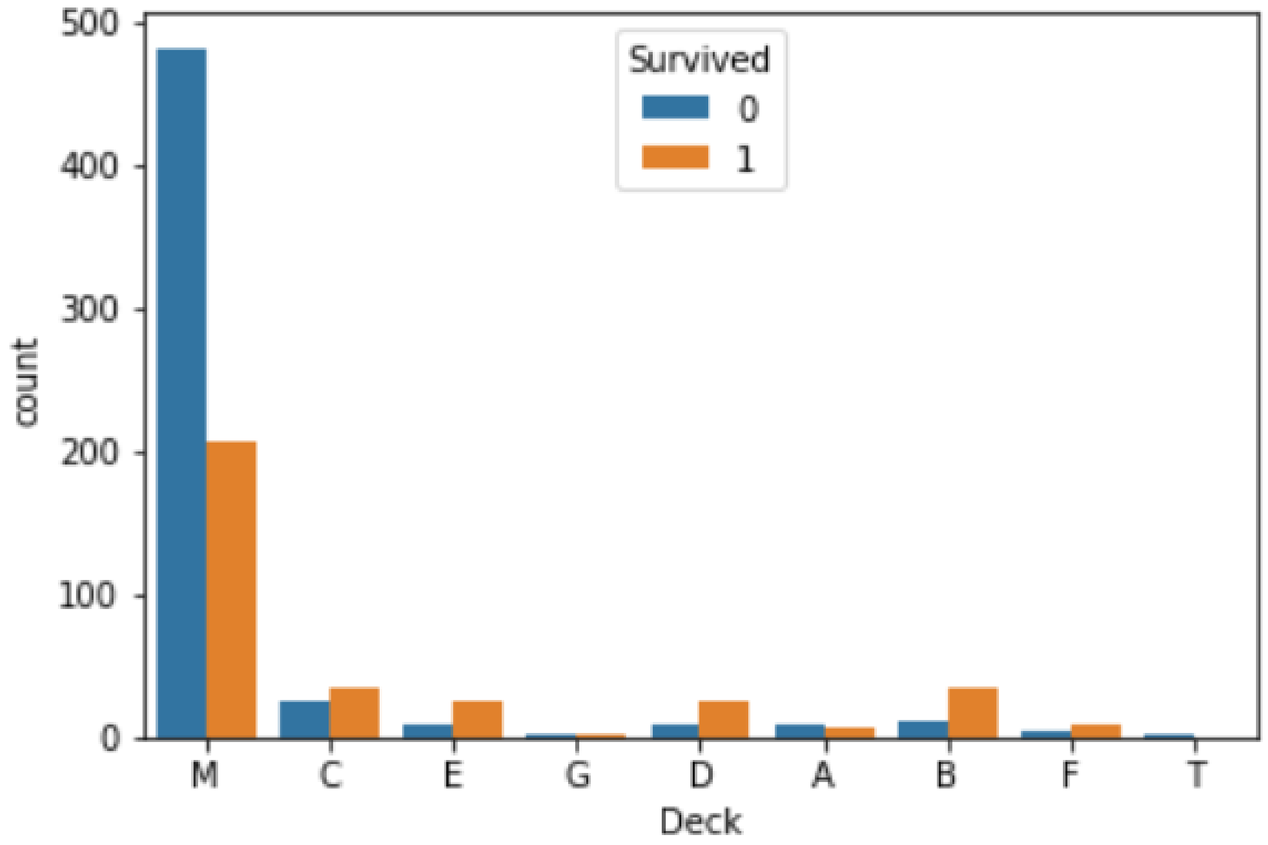
(3) Cabin 컬럼 삭제

```
train.drop('Cabin', inplace=True, axis=1)
```

탐색적 데이터 분석 (EDA) + 특성공학

Deck (객실번호) 시각화

객실번호에 따른 생존자/사망자 수 분석



groupby() 함수 : 컬럼을 그룹핑하는 기능

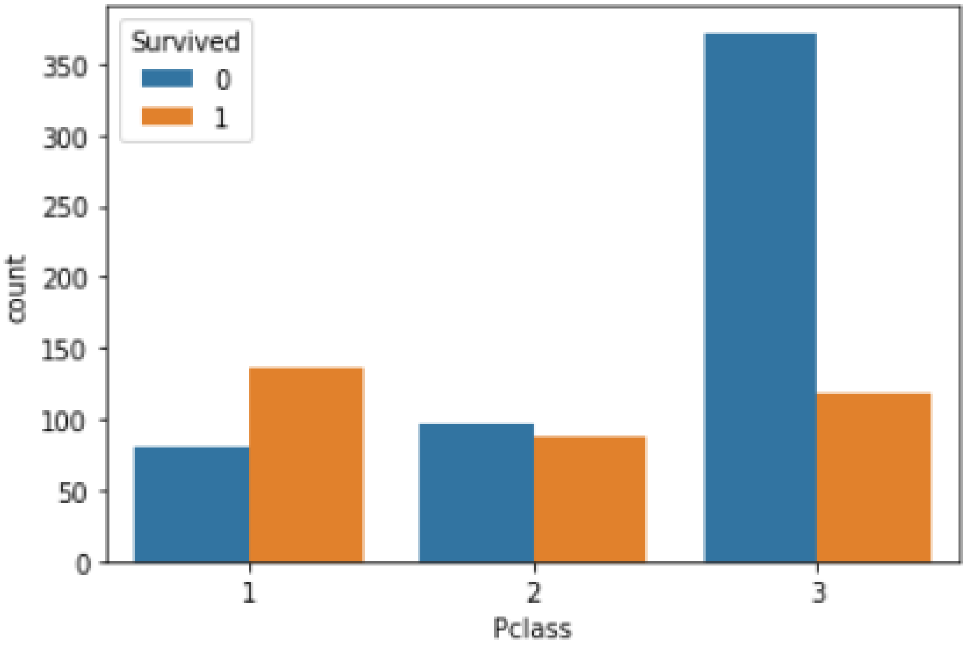
```
deck=train[['Deck','Survived','Name']]
.groupby(['Deck','Survived']).count()
```

```
sns.countplot(data=train, x='Deck',
 hue='Survived')
```

		Name
Deck	Survived	
A	0	8
	1	7
B	0	12
	1	35
C	0	24
	1	35
D	0	8
	1	25
E	0	8
	1	24
F	0	5
	1	8
G	0	2
	1	2
M	0	481
	1	206
T	0	1

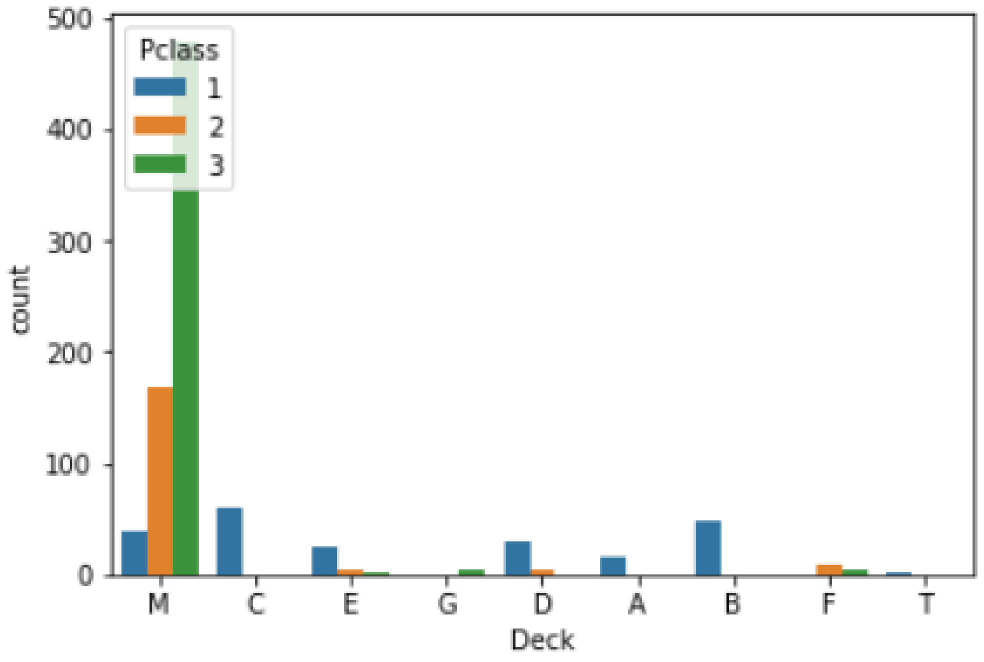
Pclass (등급) 시각화

등급에 따른 생존자/사망자 수 분석



Deck와 Pclass 시각화

Deck에 따른 Pclass 수 분석

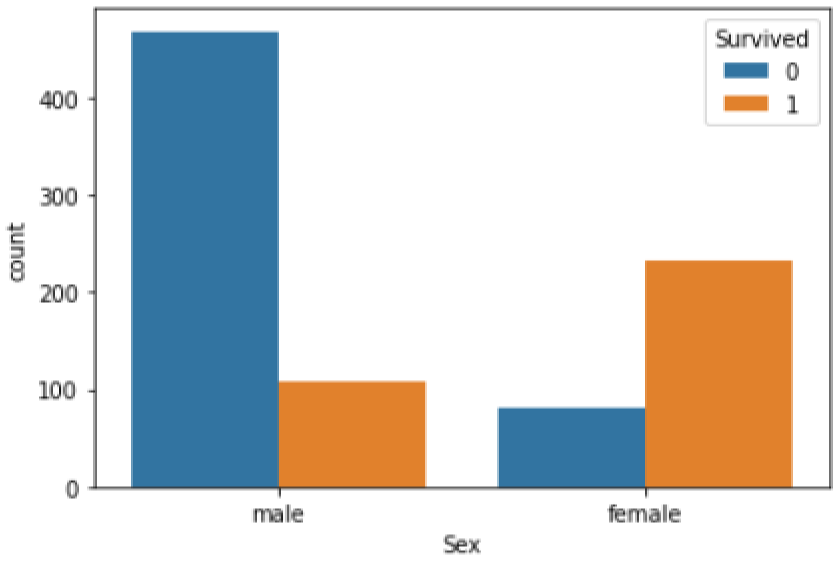


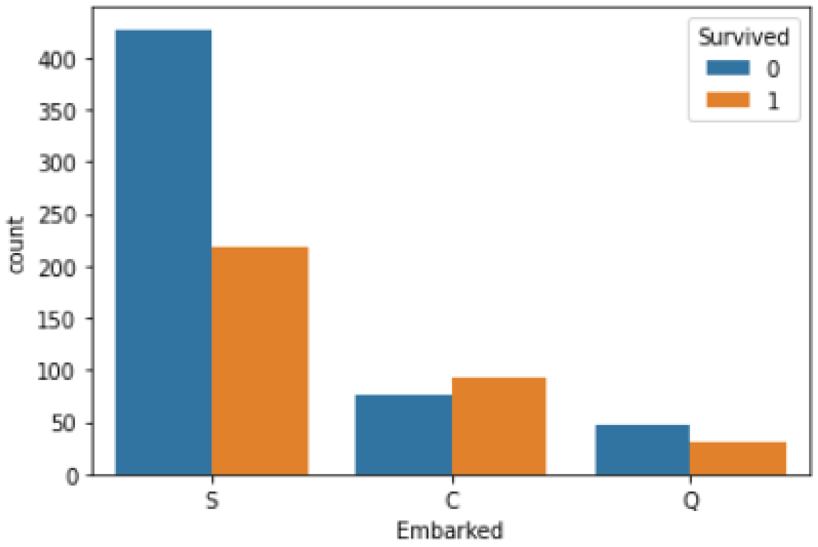
탐색적 데이터 분석 (EDA) + 특성공학

스마트인재개발원
Smart Human Resources Development

Sex, Embarked 시각화

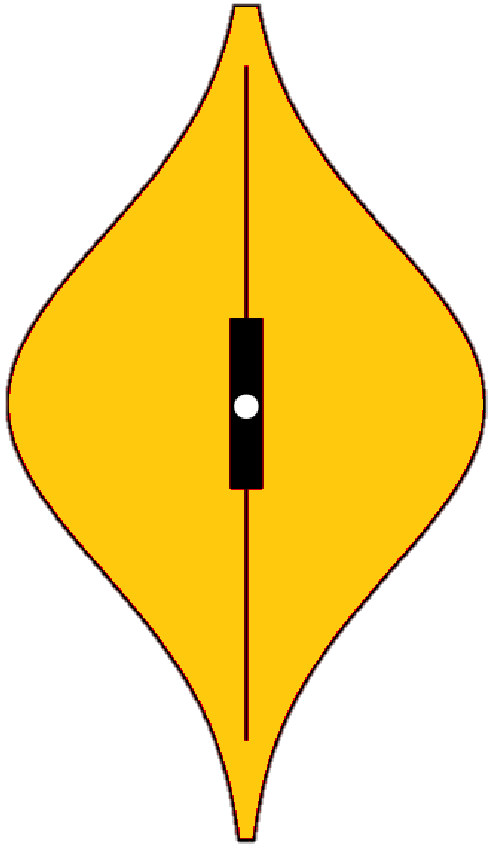
Sex와 Embarked에 따른 생존자/사망자 수 분석





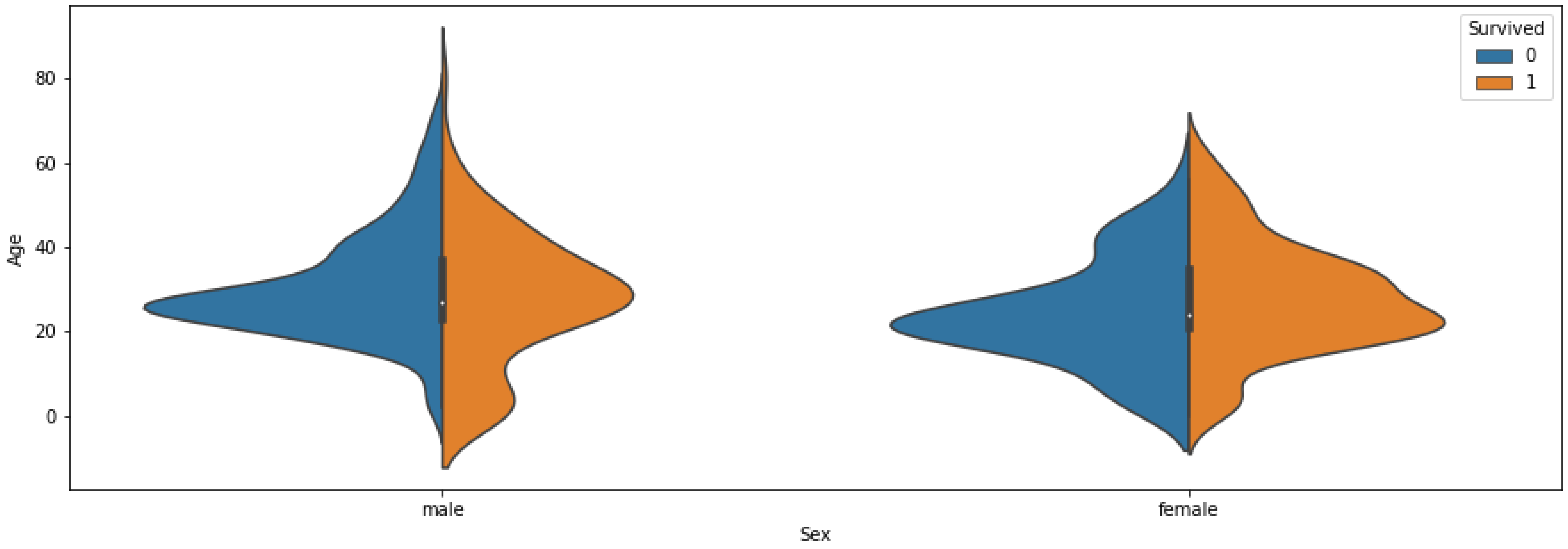
Violin plot : 3개의 변수를 시각화하는 도구

- KDE 플롯과 박스 플롯을 합쳐놓은 것
- 가운데 흰색점 : 중앙값
- 가운데 두꺼운 검정색 선
  - 흰점 아래쪽 끝이 25%,
  - 위쪽 끝이 75%
- 가운데 얇은 검정색 선 : 신뢰 구간



Sex, Age에 따른 생존자/사망자 수 시각화

```
plt.figure(figsize=(15,5))
sns.violinplot(data=train, x='Sex', y='Age',
        hue='Survived', split=True)
```



Age를 binning하여 범주형 데이터로 변경

- Age_cat 컬럼에 저장

Age < 10	**0**
10 <= Age < 20	1
20 <= Age < 30	2
30 <= Age < 40	3
40 <= Age < 50	4
50 <= Age < 60	5
60 <= Age < 70	6
70 <= Age	7

탐색적 데이터 분석 (EDA) + 특성공학

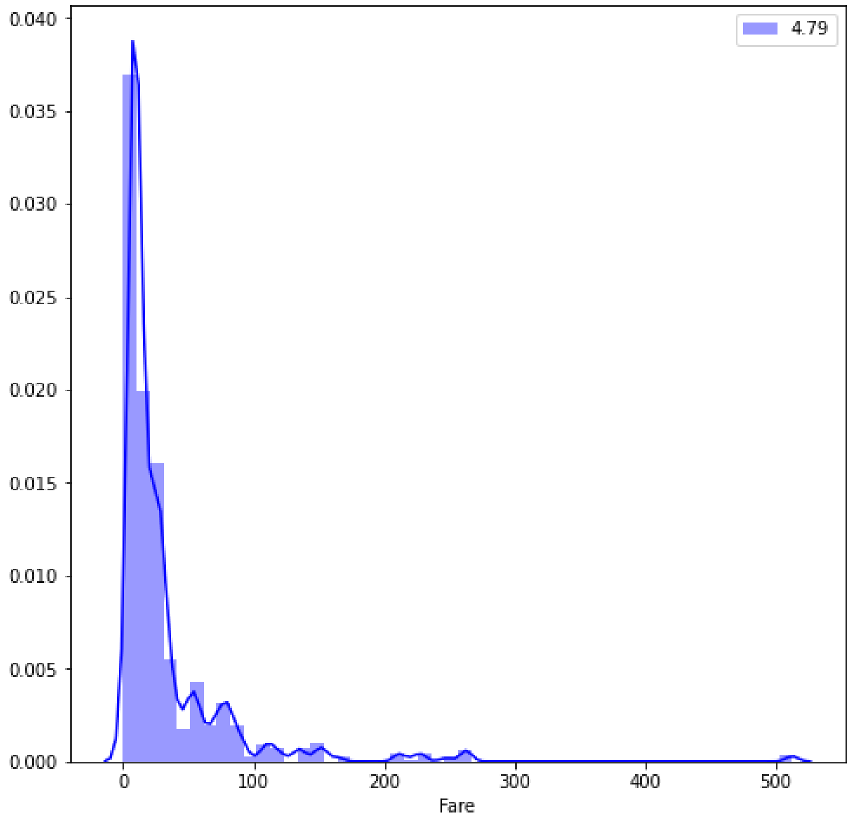
스마트인재개발원
Smart Human Resources Development

Fare를 정규분포로 변환

```
plt.figure(figsize=(8,8))

g = sns.distplot(train['Fare'],
color='b', label='{:.2f}'.format
(train['Fare'].skew()))

g.legend()
```

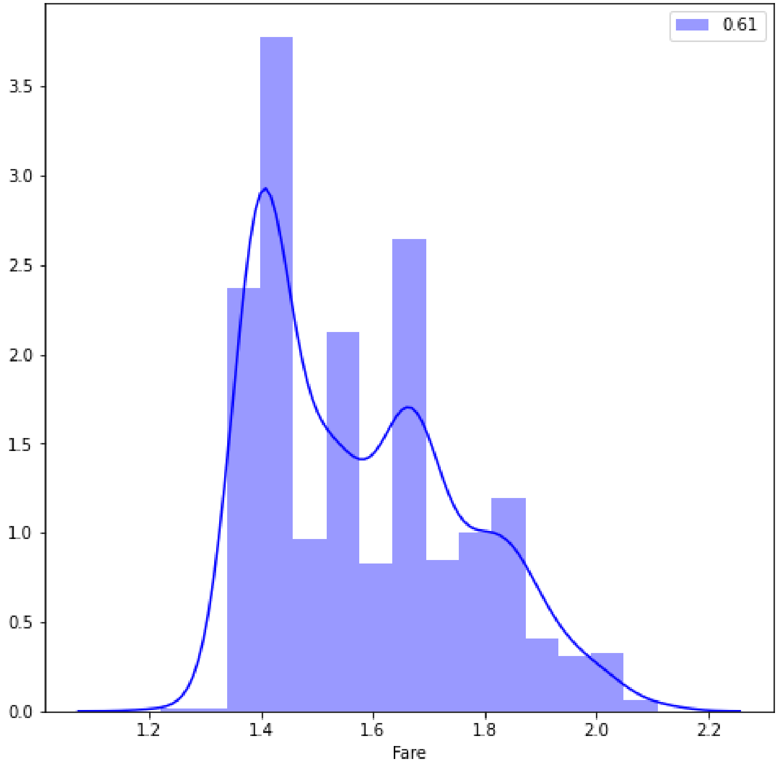


스마트인재개발원
Smart Human Resources Development

Fare를 정규분포로 변환

```
train['Fare'] = np.log(train['Fare'] + 1)
test['Fare'] = np.log(test['Fare'] + 1)
```

```
plt.figure(figsize=(8,8))
g = sns.distplot(train['Fare'],
color='b', label='{:.2f}'.format
(train['Fare'].skew()))
g.legend()
```



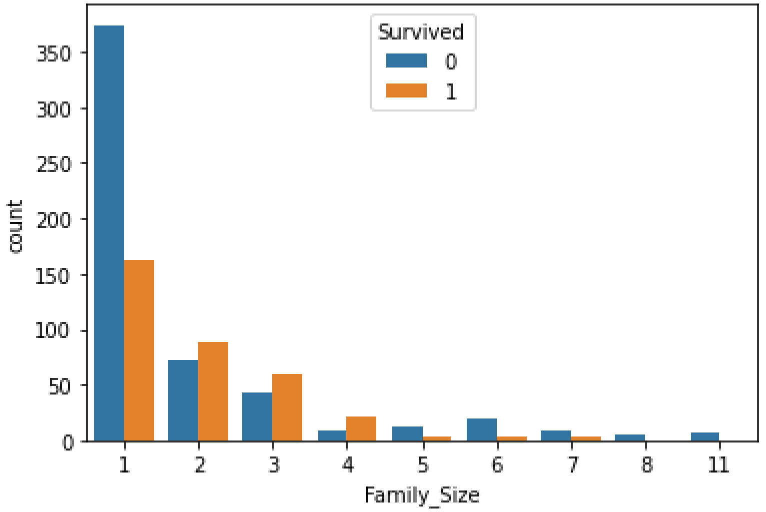
탐색적 데이터 분석 (EDA) + 특성공학

스마트인재개발원
Smart Human Resources Development

Parch(부모자식), Sibsp(형제자매) 통합

```
train['Family_Size'] = train['Parch'] + train['SibSp'] + 1
test['Family_Size'] = test['Parch'] + test['SibSp'] + 1
```

```
sns.countplot(data=train, x='Family_Size',
hue='Survived')
```



Parch(부모자식), Sibsp(형제자매) 통합

- 가족사이즈가 1이면 Alone, 2~4면 Small, 5이상이면 Large

```
bins = [0,1,4,11]
labels = ['Alone','Small','Large']

train['Family_Group'] = pd.cut(train['Family_Size'],
bins=bins, labels=labels)

test['Family_Group'] = pd.cut(test['Family_Size'],
bins=bins, labels=labels)
```

스마트인재개발원
Smart Human Resources Development

Name 시각화

- Name 값에 공통적으로 , 호칭. 이 공통적으로 포함

```
PassengerId
1                                Braund, Mr. Owen Harris
2      Cumings, Mrs. John Bradley (Florence Briggs Th...
3                                 Heikkinen, Miss. Laina
4           Futrelle, Mrs. Jacques Heath (Lily May Peel)
5                               Allen, Mr. William Henry
                              ...
887                                Montvila, Rev. Juozas
888                         Graham, Miss. Margaret Edith
889             Johnston, Miss. Catherine Helen "Carrie"
890                                Behr, Mr. Karl Howell
891                                  Dooley, Mr. Patrick
Name: Name, Length: 891, dtype: object
```

Name 시각화

- 호칭만 추출하여 Initial 컬럼을 생성하여 저장

```
def split_title(row):
    return row.split(',')[1].split('.')[0].strip()
```

```
train['Initial'] = train['Name'].apply(split_title)
test['Initial'] = test['Name'].apply(split_title)
```

Initial 통합

- Master, Mr, Mrs, Miss, Other

```
train['Initial'].replace(['Mlle','Mme','Ms','Dr','Major','Lady','the Countes
s','Jonkheer','Col','Rev','Capt','Sir','Don', 'Dona'],
['Miss','Miss','Miss','Mrs','Mr','Mrs','Mrs','Other','Other','Other','Mr','Mr','
Mr', 'Mr'],inplace=True)

test['Initial'].replace(['Mlle','Mme','Ms','Dr','Major','Lady','the Countess'
,'Jonkheer','Col','Rev','Capt','Sir','Don', 'Dona'],
['Miss','Miss','Miss','Mr','Mr','Mrs','Mrs','Other','Other','Other','Mr',
','Mr','Mr', 'Mr'],inplace=True)
```

Initial 통합

```
train['Initial'].unique()
```

```
array(['Mr', 'Mrs', 'Miss', 'Master', 'Other'], dtype=object)
```

Ticket 컬럼 삭제 / 필요 없는 칼럼 삭제

```
train.drop('Ticket',axis=1,inplace=True)
train.drop('Cabin', axis=1, inplace=True)
train.drop('Family_Size',axis=1,inplace=True)
train.drop('Age', axis=1, inplace=True)
train.drop('Parch',axis=1,inplace=True)
train.drop('SibSp',axis=1,inplace=True)
train.drop('Name',axis=1,inplace=True)
```

전처리가 완료된 최종 데이터 셋

```python
train.head()
```

	Survived	Pclass	Sex	Fare	Embarked	Deck	Age_cat	Family_Group	Initial
PassengerId									
1	0	3	male	1.981001	S	M	2	Small	Mr
2	1	1	female	4.266662	C	C	3	Small	Mrs
3	1	3	female	2.070022	S	M	2	Alone	Miss
4	1	1	female	3.972177	S	C	3	Small	Mrs
5	0	3	male	2.085672	S	M	3	Alone	Mr

상관관계 분석 (Label Encoding 수행)

Sex	
male	0
female	1

Embarked	
C	0
Q	1
S	2

Family Group	
Alone	0
Small	1
Large	2

Deck	
A	0
B	1
C	2
D	3
E	4
F	5
G	6
M	7
T	8

Initial	
Master	0
Miss	1
Mr	2
Mrs	3
Other	4

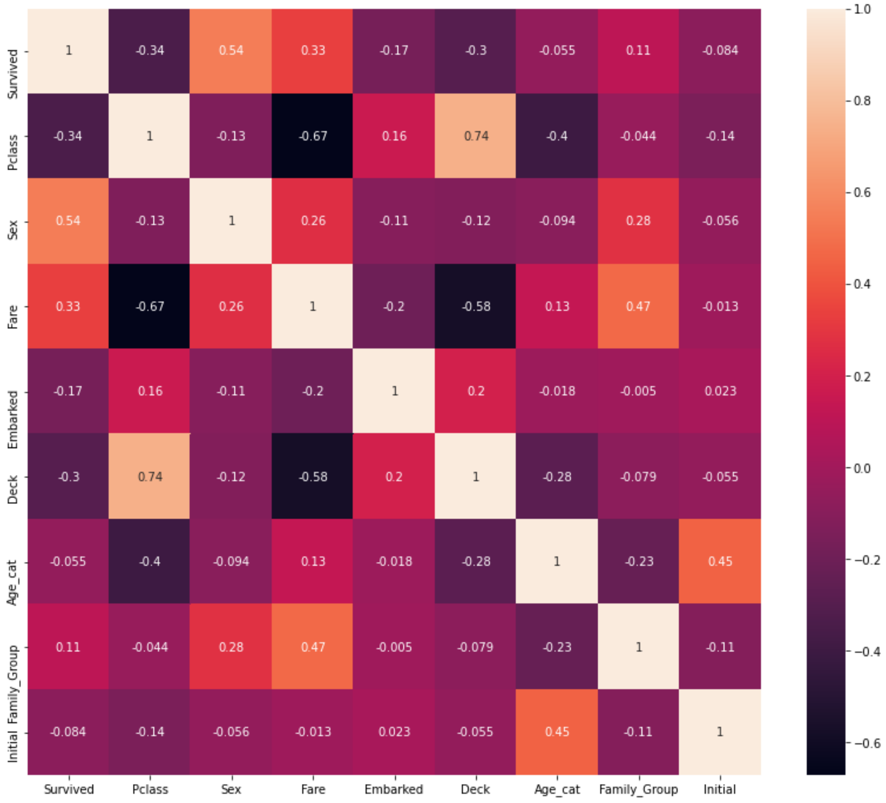
상관관계 분석 (Label Encoding 수행)

	Survived	Pclass	Sex	Fare	Embarked	Deck	Age_cat	Family_Group	Initial
PassengerId									
1	0	3	0	1.981001	2	7	2	1	2
2	1	1	1	4.266662	0	2	3	1	3
3	1	3	1	2.070022	2	7	2	0	1
4	1	1	1	3.972177	2	2	3	1	3
5	0	3	0	2.085672	2	7	3	0	2

탐색적 데이터 분석 (EDA) + 특성공학

상관관계 분석

```
plt.figure(figsize=(14,12))
sns.heatmap(train.corr(), annot=True)
```



스마트인재개발원
Smart Human Resources Development

One-hot Encoding

```
cat_feature = ['Pclass', 'Embarked', 'Deck', 'Age_cat', '
Family_Group', 'Initial']

for cat_name in cat_feature:
    dummy = pd.get_dummies(train[cat_name],
                           prefix=cat_name)
    X_train = pd.concat([X_train,dummy], axis=1)
    X_train.drop(cat_name,axis=1,inplace=True)
```

탐색적 데이터 분석 (EDA) + 특성공학

스마트인재개발원
Smart Human Resources Development

훈련데이터와 테스트데이터 구조를 같도록 처리

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 33 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Sex             891 non-null    int64
 1   Fare            891 non-null    float64
 2   Pclass_1        891 non-null    uint8
 3   Pclass_2        891 non-null    uint8
 4   Pclass_3        891 non-null    uint8
 5   Embarked_0      891 non-null    uint8
 6   Embarked_1      891 non-null    uint8
 7   Embarked_2      891 non-null    uint8
 8   Deck_0          891 non-null    uint8
 9   Deck_1          891 non-null    uint8
 10  Deck_2          891 non-null    uint8
 11  Deck_3          891 non-null    uint8
 12  Deck_4          891 non-null    uint8
 13  Deck_5          891 non-null    uint8
 14  Deck_6          891 non-null    uint8
 15  Deck_7          891 non-null    uint8
 16  Deck_8          891 non-null    uint8
 17  Age_cat_0       891 non-null    uint8
 18  Age_cat_1       891 non-null    uint8
 19  Age_cat_2       891 non-null    uint8
 20  Age_cat_3       891 non-null    uint8
 21  Age_cat_4       891 non-null    uint8
 22  Age_cat_5       891 non-null    uint8
 23  Age_cat_6       891 non-null    uint8
 24  Age_cat_7       891 non-null    uint8
 25  Family_Group_0  891 non-null    uint8
 26  Family_Group_1  891 non-null    uint8
 27  Family_Group_2  891 non-null    uint8
 28  Initial_0       891 non-null    uint8
 29  Initial_1       891 non-null    uint8
 30  Initial_2       891 non-null    uint8
 31  Initial_3       891 non-null    uint8
 32  Initial_4       891 non-null    uint8
dtypes: float64(1), int64(1), uint8(31)
memory usage: 87.9 KB
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 418 entries, 892 to 1309
Data columns (total 32 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Sex             418 non-null    int64
 1   Fare            418 non-null    float64
 2   Pclass_1        418 non-null    uint8
 3   Pclass_2        418 non-null    uint8
 4   Pclass_3        418 non-null    uint8
 5   Embarked_0      418 non-null    uint8
 6   Embarked_1      418 non-null    uint8
 7   Embarked_2      418 non-null    uint8
 8   Deck_0          418 non-null    uint8
 9   Deck_1          418 non-null    uint8
 10  Deck_2          418 non-null    uint8
 11  Deck_3          418 non-null    uint8
 12  Deck_4          418 non-null    uint8
 13  Deck_5          418 non-null    uint8
 14  Deck_6          418 non-null    uint8
 15  Deck_7          418 non-null    uint8
 16  Age_cat_0       418 non-null    uint8
 17  Age_cat_1       418 non-null    uint8
 18  Age_cat_2       418 non-null    uint8
 19  Age_cat_3       418 non-null    uint8
 20  Age_cat_4       418 non-null    uint8
 21  Age_cat_5       418 non-null    uint8
 22  Age_cat_6       418 non-null    uint8
 23  Age_cat_7       418 non-null    uint8
 24  Family_Group_0  418 non-null    uint8
 25  Family_Group_1  418 non-null    uint8
 26  Family_Group_2  418 non-null    uint8
 27  Initial_0       418 non-null    uint8
 28  Initial_1       418 non-null    uint8
 29  Initial_2       418 non-null    uint8
 30  Initial_3       418 non-null    uint8
 31  Initial_4       418 non-null    uint8
dtypes: float64(1), int64(1), uint8(30)
memory usage: 22.0 KB
```

```
X_test['Deck_8'] = 0
```

모델 선택 및 학습

KNeighborsClassifier

DecisionTreeClassifier

교차검증을 이용하여 모델을 평가해보자