Kaggle Titanic 분석

고은경

분석 프로세스

- 이름, 나이, 성별, 생존 여부 등 891명의 승객 정보를 가지고 생존 여부를 예측 ➔ train.csv 파일
- 생존 여부를 예측하는 모델을 가지고 418명의 생존 여부를 예측 ➔ test.csv 파일
- 승객 아이디, 생존 여부를 저장해 제출 ➔ sample_submissioin.csv 파일
- 제출된 데이터 가지고 점수 측정 ➔ LeaderBoard

데이터 확인

- Servived : 생존유무 (0 = 사망, 1 = 생존)
- Pclass : 티켓 클래스 (1 = 1st, 2 = 2nd, 3 = 3rd)
- Sex : 성별
- Age : 나이
- SibSp : 함께 탑승한 형제자매, 배우자 수 총합
- Parch : 함께 탑승한 부모, 자녀 수 총합
- Ticket : 티켓 넘버
- Fare : 탑승 요금
- Cabin : 객실 넘버
- Embarked : 탑승 항구

데이터 확인

In [186]:
```
import pandas as pd
train = pandas.read_csv('train.csv')
test = pandas.read_csv('test.csv')
```

In [187]:
```
train.shape
```

Out[187]: (891, 12)

In [188]:
```
test.shape
```

Out[188]: (418, 11)

In [189]:
```
train.head()
```

Out[189]:

	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
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [190]:
```
train.head(20)
```

In [191]:
```
train.info()
```

```
<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: 83.7+ KB
```

In [192]:
```
test.info()
```

```
<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
```

데이터 확인

Null 데이터 개수 체크

```
In [193]: train.isnull()
          train.isnull().sum()
```

```
Out[193]: 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
```

```
In [194]: test.isnull().sum()
```

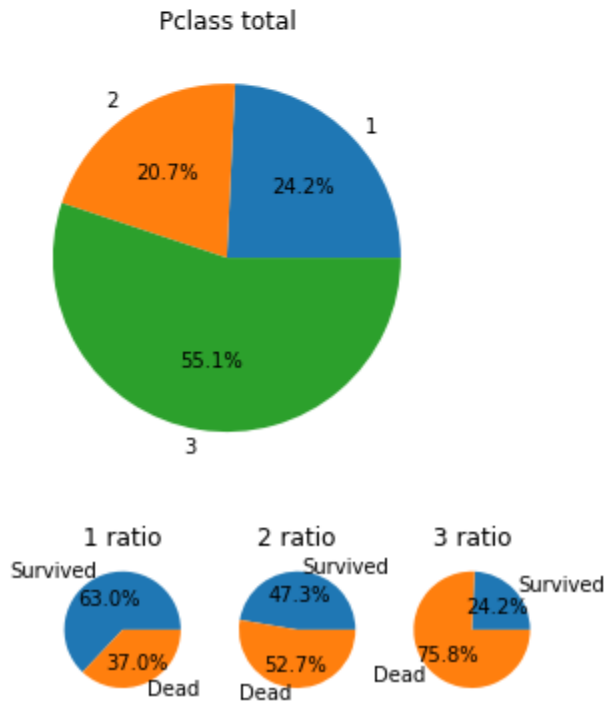
```
Out[194]: PassengerId      0
          Pclass           0
          Name             0
          Sex              0
          Age             86
          SibSp            0
          Parch            0
          Ticket           0
          Fare             1
          Cabin          327
          Embarked         0
          dtype: int64
```

데이터 시각화

Pie 차트

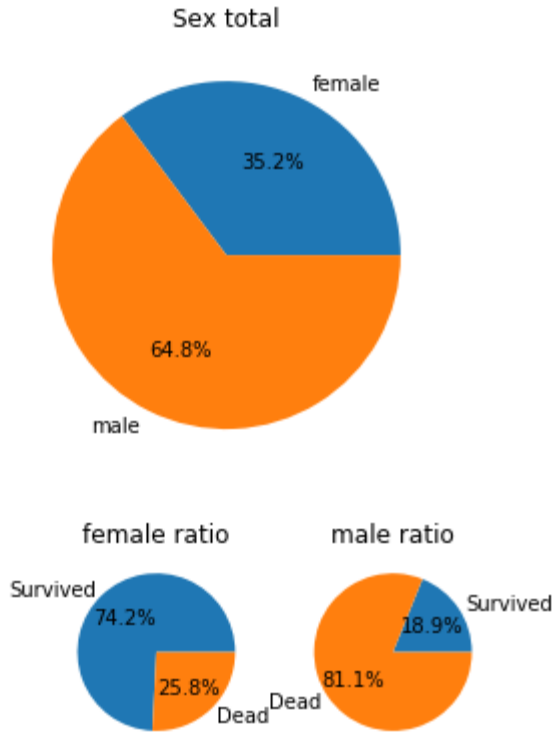
```
import matplotlib.pyplot as plt
%matplotlib inline
def pie_char(feature):
    feature_ratio = train[feature].value_counts(sort=False)
    feature_size = feature_ratio.size
    feature_index = feature_ratio.index
    survived = train[train['Survived'] == 1][feature].value_counts()
    dead = train[train['Survived'] == 0][feature].value_counts()
    plt.plot(aspect='auto')
    plt.pie(feature_ratio, labels=feature_index, autopct='%1.1f%%')
    plt.title(feature + ' total')
    plt.show()
    for i, index in enumerate(feature_index):
        plt.subplot(1, feature_size + 1, i + 1, aspect='equal')
        plt.pie([survived[index], dead[index]], labels=['Survived', 'Dead'], autopct='%1.1f%%')
        plt.title(str(index) + ' ratio')
    plt.show()
```

In [196]:
```
pie_char('Pclass')
```

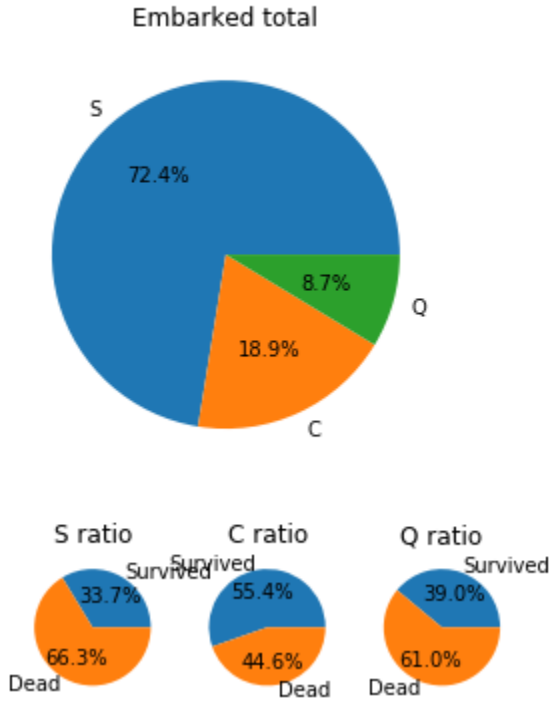


데이터 시각화 Pie 차트

```
In [197]: pie_char('Sex')
```




```
In [198]: pie_char('Embarked')
```

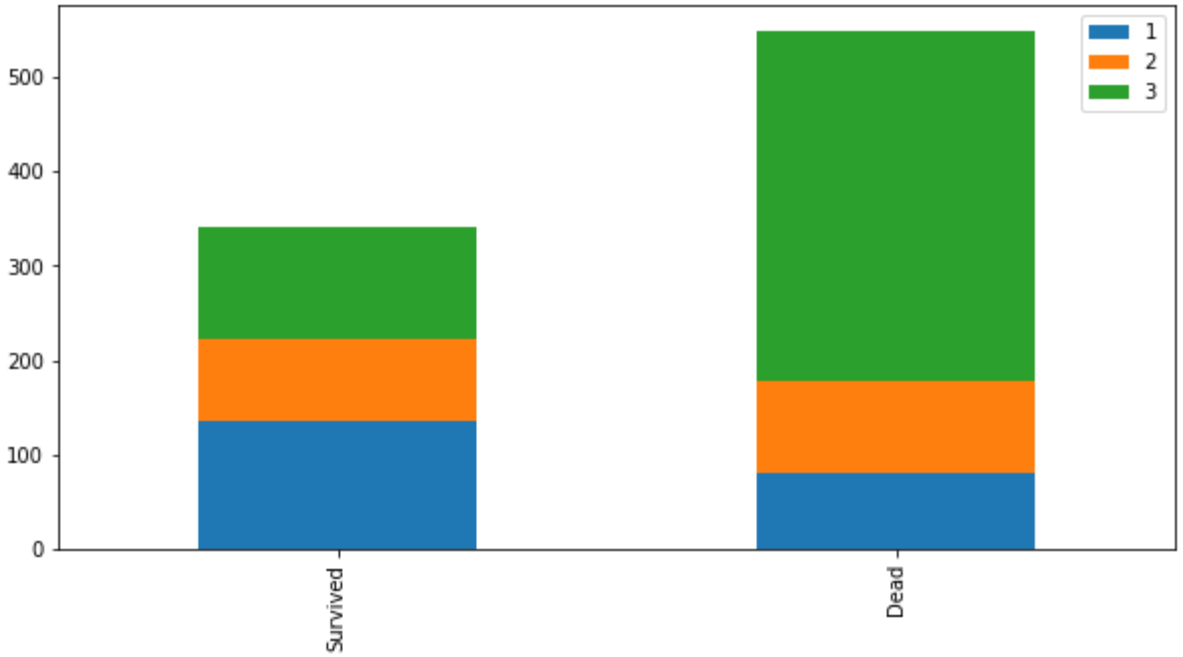


데이터 시각화

Bar 차트

```
In [199]: def bar_chart(feature):
              survived = train[train['Survived']==1][feature].value_counts()
              dead = train[train['Survived']==0][feature].value_counts()
              df = pd.DataFrame([survived,dead])
              df.index = ['Survived','Dead']
              df.plot(kind='bar',stacked=True, figsize=(10,5))
```

```
In [200]: bar_chart('Pclass')
```

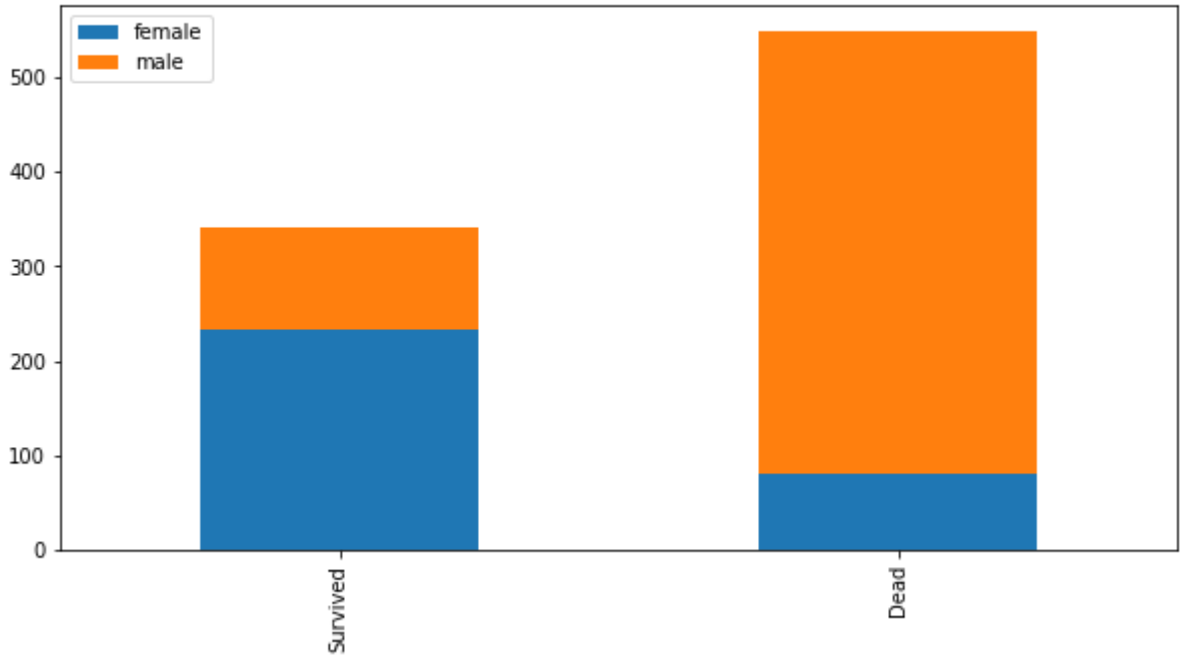


➔ 티켓 등급이 높을수록 생존율이 높다.

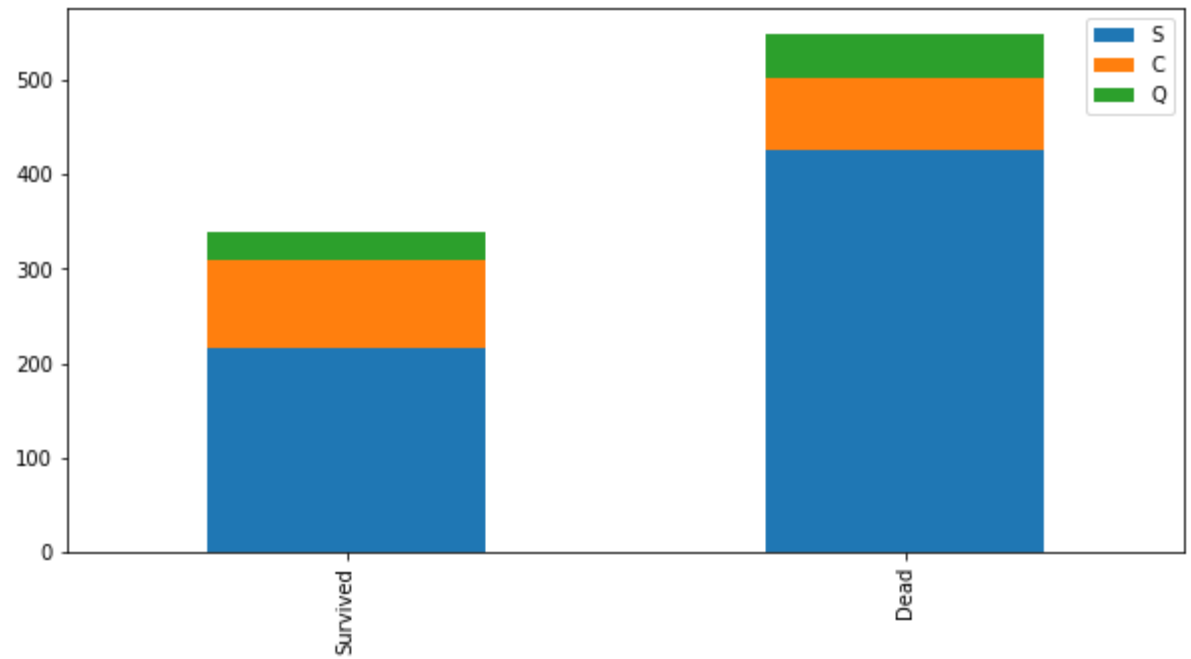
➔ 좌석 위치 관련 추측

데이터 시각화 Bar 차트

```
bar_chart('Sex')
```




```
bar_chart('Embarked')
```



➔ 남성보다 여성 생존률이 높다.

데이터 전처리

NaN값 처리 - 나이데이터

In [203]:
```
# 전체 탑승객의 평균 나이
train.mean()['Age']
```

Out [203]:
```
29.69911764705882
```

In [204]:
```
train.median()['Age']
```

Out [204]:
```
28.0
```

In [205]:
```
train.groupby('Pclass').mean()['Age']
```

Out [205]:
```
Pclass
1    38.233441
2    29.877630
3    25.140620
Name: Age, dtype: float64
```

In [206]:
```
train.groupby('Pclass').median()['Age']
```

Out [206]:
```
Pclass
1    37.0
2    29.0
3    24.0
Name: Age, dtype: float64
```

In [207]:
```
train['Age'].fillna(train.groupby('Pclass')['Age'].transform('median'), inplace=True)
test['Age'].fillna(train.groupby('Pclass')['Age'].transform('median'), inplace=True)
```

In [208]:
```
train.isnull().sum()
```

Out [208]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

데이터 전처리

NaN값 처리 - 탑승 항구

In [209]:
```
train.groupby('Embarked').count()
```

Out [209]:

Embarked	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
C	168	168	168	168	168	168	168	168	168	168	69
Q	77	77	77	77	77	77	77	77	77	77	4
S	644	644	644	644	644	644	644	644	644	644	129

In [210]:
```
train_test_data = [train, test]
for dataset in train_test_data:
    dataset['Embarked'] = dataset['Embarked'].fillna('S')

train.isnull().sum()
```

Out [210]:
```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         0
dtype: int64
```

데이터 전처리

이름: Mr, Miss, Mrs

```
In [211]: train.head(100)
```

Out [211]:

	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
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
95	96	0	3	Shorney, Mr. Charles Joseph	male	24.0	0	0	374910	8.0500	NaN	S
96	97	0	1	Goldschmidt, Mr. George B	male	71.0	0	0	PC 17754	34.6542	A5	C
97	98	1	1	Greenfield, Mr. William Bertram	male	23.0	0	1	PC 17759	63.3583	D10 D12	C
98	99	1	2	Doling, Mrs. John T (Ada Julia Bone)	female	34.0	0	1	231919	23.0000	NaN	S
99	100	0	2	Kantor, Mr. Sinai	male	34.0	1	0	244367	26.0000	NaN	S

100 rows × 12 columns

데이터 전처리

이름: Mr, Miss, Mrs

In [212]:
```
for dataset in train_test_data:
    dataset['Title'] = dataset['Name'].str.extract(' ([A-Za-z]+)\.', expand=False)

#A-Za-z부분 정규식?

train['Title'].value_counts()
```

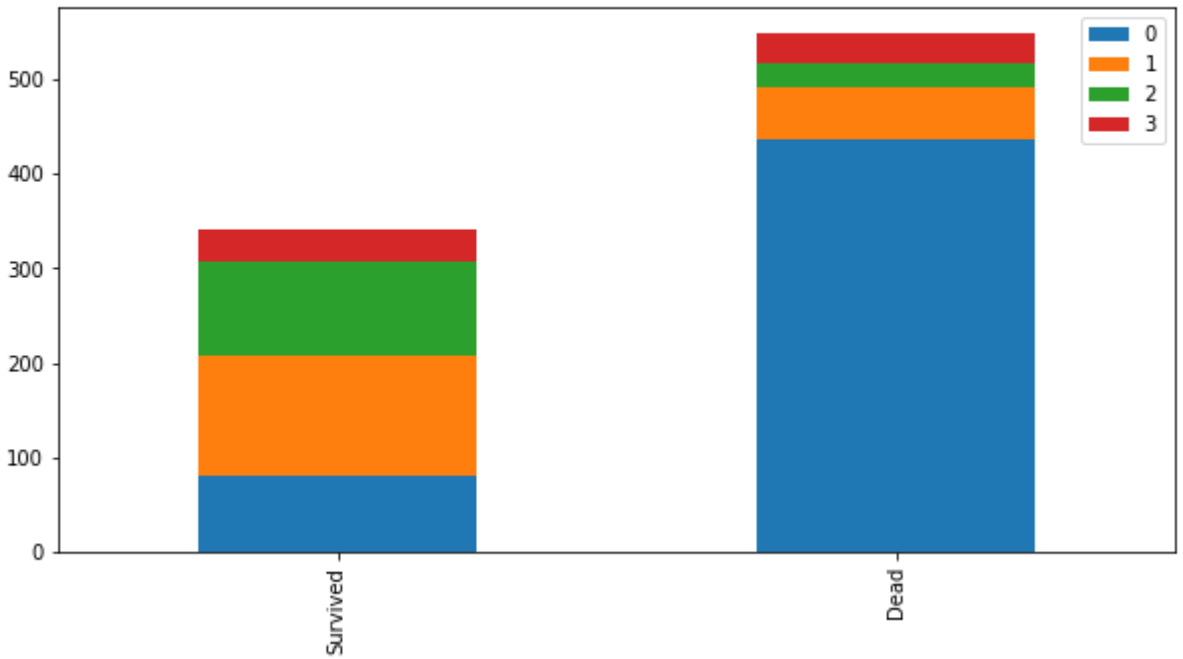
Out[212]:
```
Mr          517
Miss        182
Mrs         125
Master       40
Dr            7
Rev           6
Mlle          2
Major         2
Col           2
Ms            1
Capt          1
Countess      1
Don           1
Jonkheer      1
Sir           1
Lady          1
Mme           1
Name: Title, dtype: int64
```

In [213]:
```
title_mapping = {"Mr": 0, "Miss": 1, "Mrs": 2, "Master": 3, "Dr": 3, "Rev": 3, "Col": 3, "Major": 3, "Mlle": 3,"Countess": 3, "Ms": 3, "Lad
for dataset in train_test_data:    dataset['Title'] = dataset['Title'].map(title_mapping)
```

데이터 전처리

이름: Mr, Miss, Mrs

```
In [215]: bar_chart('Title')
```

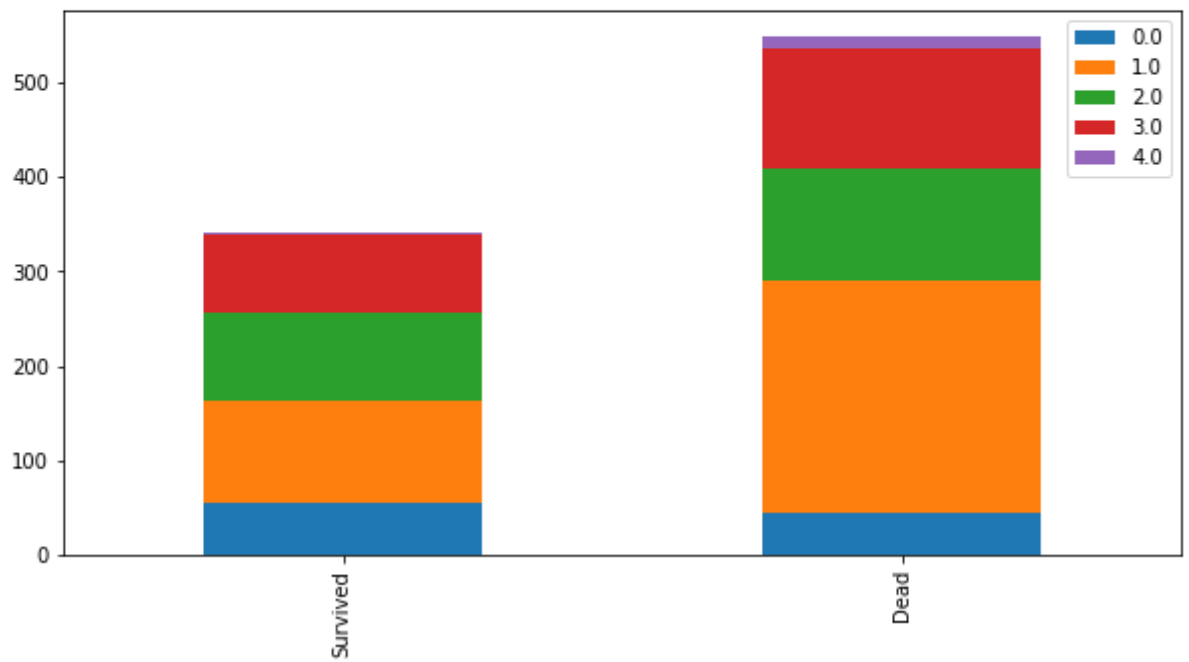


Feature Engineering

나이 데이터 수정

In [216]:
```
for dataset in train_test_data:
    dataset.loc[dataset['Age']<=16, 'Age'] = 0
    dataset.loc[(dataset['Age']>16) & (dataset['Age']<=26), 'Age'] = 1
    dataset.loc[(dataset['Age']>26) & (dataset['Age']<=36), 'Age'] = 2
    dataset.loc[(dataset['Age']>36) & (dataset['Age']<=62), 'Age'] = 3
    dataset.loc[dataset['Age']>62, 'Age'] = 4
```

In [217]:
```
bar_chart('Age')
```



Feature Engineering

성별 데이터 수정

In [218]:
```
sex_mapping = {"male" : 0, "female" : 1}
for dataset in train_test_data:
    dataset['Sex'] = dataset['Sex'].map(sex_mapping)

train.head(100)
```

Out [218]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title
0	1	0	3	Braund, Mr. Owen Harris	0	1.0	1	0	A/5 21171	7.2500	NaN	S	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	3.0	1	0	PC 17599	71.2833	C85	C	2
2	3	1	3	Heikkinen, Miss. Laina	1	1.0	0	0	STON/O2. 3101282	7.9250	NaN	S	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	2.0	1	0	113803	53.1000	C123	S	2
4	5	0	3	Allen, Mr. William Henry	0	2.0	0	0	373450	8.0500	NaN	S	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	96	0	3	Shorney, Mr. Charles Joseph	0	1.0	0	0	374910	8.0500	NaN	S	0
96	97	0	1	Goldschmidt, Mr. George B	0	4.0	0	0	PC 17754	34.6542	A5	C	0
97	98	1	1	Greenfield, Mr. William Bertram	0	1.0	0	1	PC 17759	63.3583	D10 D12	C	0
98	99	1	2	Doling, Mrs. John T (Ada Julia Bone)	1	2.0	0	1	231919	23.0000	NaN	S	2
99	100	0	2	Kantor, Mr. Sinai	0	2.0	1	0	244367	26.0000	NaN	S	0

Feature Engineering

탑승항구 데이터 수정

In [219]:
```
embarked_mapping = {"S" : 0, "C" : 1, "Q" : 2}
for dataset in train_test_data:
    dataset['Embarked'] = dataset['Embarked'].map(embarked_mapping)

train.head(100)
```

Out [219]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked	Title
0	1	0	3	Braund, Mr. Owen Harris	0	1.0	1	0	A/5 21171	7.2500	NaN	0	0
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	3.0	1	0	PC 17599	71.2833	C85	1	2
2	3	1	3	Heikkinen, Miss. Laina	1	1.0	0	0	STON/O2. 3101282	7.9250	NaN	0	1
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	2.0	1	0	113803	53.1000	C123	0	2
4	5	0	3	Allen, Mr. William Henry	0	2.0	0	0	373450	8.0500	NaN	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	96	0	3	Shorney, Mr. Charles Joseph	0	1.0	0	0	374910	8.0500	NaN	0	0
96	97	0	1	Goldschmidt, Mr. George B	0	4.0	0	0	PC 17754	34.6542	A5	1	0
97	98	1	1	Greenfield, Mr. William Bertram	0	1.0	0	1	PC 17759	63.3583	D10 D12	1	0
98	99	1	2	Doling, Mrs. John T (Ada Julia Bone)	1	2.0	0	1	231919	23.0000	NaN	0	2
99	100	0	2	Kantor, Mr. Sinai	0	2.0	1	0	244367	26.0000	NaN	0	0

문제 데이터, 의미 X 데이터 삭제

In [220]:
```
train.drop('Cabin', axis=1, inplace=True)
test.drop('Cabin', axis=1, inplace=True)
train.drop('Name', axis=1, inplace=True)
test.drop('Name', axis=1, inplace=True)
train.drop('Ticket', axis=1, inplace=True)
test.drop('Ticket', axis=1, inplace=True)
train.drop('Fare', axis=1, inplace=True)
test.drop('Fare', axis=1, inplace=True)
```

In [221]:
```
train.head(100)
```

Out[221]:

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Embarked	Title
0	1	0	3	0	1.0	1	0	0	0
1	2	1	1	1	3.0	1	0	1	2
2	3	1	3	1	1.0	0	0	0	1
3	4	1	1	1	2.0	1	0	0	2
4	5	0	3	0	2.0	0	0	0	0
...	...	...	...	...	...	...	...	...	...
95	96	0	3	0	1.0	0	0	0	0
96	97	0	1	0	4.0	0	0	1	0
97	98	1	1	0	1.0	0	1	1	0
98	99	1	2	1	2.0	0	1	0	2
99	100	0	2	0	2.0	1	0	0	0

머신러닝

In [223]:
```
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
import numpy as np

from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
k_fold = KFold(n_splits=10, shuffle=True, random_state=0)
```

In [225]:
```
train_data = train.drop('Survived', axis = 1)
target = train['Survived']

test_data = test.drop('PassengerId', axis = 1).copy()
test_data.head()

train_data = train_data.drop('PassengerId', axis = 1).copy()
train_data.head()
```

Out [225]:

	Pclass	Sex	Age	SibSp	Parch	Embarked	Title
0	3	0	1.0	1	0	0	0
1	1	1	3.0	1	0	1	2
2	3	1	1.0	0	0	0	1
3	1	1	2.0	1	0	0	2
4	3	0	2.0	0	0	0	0

머신러닝

KNN, Decision Tree

In [235]:
```
clf = KNeighborsClassifier(n_neighbors = 13)
#  이 때 Knn은 항상 홀수여야 함
scoring = 'accuracy'
score = cross_val_score(clf, train_data, target, cv=k_fold, n_jobs=1, scoring=scoring)
print(score)
round(np.mean(score)*100, 2)
```

```
[0.82222222 0.76404494 0.84269663 0.82022472 0.82022472 0.82022472
 0.82022472 0.78651685 0.84269663 0.84269663]
```

Out [235]: 81.82

In [231]:
```
clf = DecisionTreeClassifier()
scoring = 'accuracy'
score = cross_val_score(clf, train_data, target, cv=k_fold, n_jobs=1, scoring=scoring)
print(score)
round(np.mean(score)*100, 2)
```

```
[0.77777778 0.80898876 0.78651685 0.76404494 0.86516854 0.82022472
 0.83146067 0.83146067 0.76404494 0.7752809 ]
```

Out [231]: 80.25

머신러닝

Random Forest, Naïve Bayes

In [232]:
```
clf = RandomForestClassifier(n_estimators=13)
scoring = 'accuracy'
score = cross_val_score(clf, train_data, target, cv=k_fold, n_jobs=1, scoring=scoring)
print(score)
round(np.mean(score)*100, 2)
```

```
[0.77777778 0.82022472 0.80898876 0.75280899 0.85393258 0.80898876
 0.78651685 0.79775281 0.75280899 0.7752809 ]
```

Out[232]: 79.35

In [233]:
```
clf = GaussianNB()
scoring = 'accuracy'
score = cross_val_score(clf, train_data, target, cv=k_fold, n_jobs=1, scoring=scoring)
print(score)
round(np.mean(score)*100, 2)
```

```
[0.8        0.74157303 0.82022472 0.79775281 0.80898876 0.80898876
 0.83146067 0.79775281 0.83146067 0.85393258]
```

Out[233]: 80.92

SVM은 3.7.4 버젼에서는 실행 안됨

캐글 제출 파일 만들기 & 제출

Naïve Bayes

In [237]:
```
clf = GaussianNB()
clf.fit(train_data, target)

prediction = clf.predict(test_data)
submission = pd.DataFrame({
    "PassengerId" : test["PassengerId"],
    "Survived" : prediction
})

submission.to_csv('submission.csv', index = False)
```

In [238]:
```
submission = pd.read_csv('submission.csv')
submission.head()
```

Out[238]:

	PassengerId	Survived
0	892	0
1	893	1
2	894	0
3	895	0
4	896	1

+ 테스트 데이터 추가해보기

In [243]:
```
dicaprio_winslet = pd.read_csv('dicaprio_winslet.csv')
dicaprio_winslet.head()
```

Out [243]:

	PassengerId	Pclass	Sex	Age	SibSp	Parch	Embarked	Title
0	1	3	0	1	0	0	0	0
1	2	1	1	0	1	2	0	1
2	3	3	1	1	1	2	0	1

In [244]:
```
dicaprio_winslet_data = dicaprio_winslet.drop("PassengerId", axis = 1).copy()
dicaprio_winslet_data.head()
```

Out [244]:

	Pclass	Sex	Age	SibSp	Parch	Embarked	Title
0	3	0	1	0	0	0	0
1	1	1	0	1	2	0	1
2	3	1	1	1	2	0	1

In [245]:
```
prediction = clf.predict(dicaprio_winslet_data)
submission1 = pd.DataFrame({
    "PassengerId": dicaprio_winslet["PassengerId"],
    "Survied": prediction
})
submission1.to_csv('dicaprio_winslet_result.csv', index=False)
```