목 차

문법 및 함수 정리

plt.subplots

f, ax = plt.subplots(1, 2, figsize=(20, 8))

-> 1 x 2 개의 그래프를 만들어냄.

-> 각각의 크기는 20x8

-> 총 2개의 그래프는 ax[index]로 접근 가능.

subplots 원형 그래프

df_train['Survived'].value_counts().plot.pie(explode=[0, 0.1],

autopct='%1.1f%%', ax=ax[0], shadow=True)

-> 표의 'Survived' 의 갯수를 원형그래프로 표현

-> explode는 각 지표 값에 대해 원점을 기준으로 거리를 나타냄.

-> explode의 원소들은 표현하려는 value 종류의 갯수와 같아야함.

seaborn 연속 데이터 값 그래프

fig, ax = plt.subplots(1,1, figsize=(10,5))

sns.kdeplot(df_train[df_train['Survived'] == 1]['Age'], ax=ax)

sns.kdeplot(df_train[df_train['Survived'] == 0]['Age'], ax=ax)

plt.legend(['Survived == 1', 'Survived == 0'])

plt.show()

-> seaborn 의 kdeplot 이용, 조건에 맞는 feature 데이터를 넣어줌

groupby

df.groupby(['city', 'fruits']).mean()

-> 좌측의 행을 city 별 fruits 종류로 정하고 그 외 나머지 feature 들의 평균을 보여줌

		price	quantity
city	fruits		
부산	apple	100.0	1.0
	banana	275.0	3.5
	orange	200.0	2.0
서울	apple	175.0	5.5
	banana	400.0	7.0

-> as_index 를 false 로 하면 위 그림처럼 그룹이 index 로 지정되지 않음

데이터 열의 합으로 정렬

pd.crosstab(df_train['Initial'], df_train['Sex'])

-> 위의 결과는 정렬되지 않은 데이터지만 female + male을 기준으로 정렬하고 싶음.

pd.crosstab(df_train['Initial'], df_train['Sex']).sum(axis=1)

.sort_values(ascending=False)

-> 위의 명령어처럼 열을 기준으로 더한 뒤(sum(axis=1)) 내림차순으로 정렬하면 된다.

Sex	female	male
Initial		
Capt	0	1
Col	0	2
Countess	1	0
Don	0	1
Dr	1	6
Jonkheer	0	1
Lady	1	0
Major	0	2
Master	0	40
Miss	182	0
Mlle	2	0
Mme	1	0
Mr	0	517
Mrs	125	0
Ms	1	0
Rev	0	6
Sir	0	1

```
Initial
Mr          517
Miss        182
Mrs         125
Master       40
Dr            7
Rev           6
Col           2
Mlle          2
Major         2
Countess      1
Don           1
Sir           1
Jonkheer      1
Lady          1
Mme           1
Ms            1
Capt          1
dtype: int64
```

String feature 를 수치화 시키기.

df_train['Initial'] = df_train['Initial'].map({'Master': 0, 'Miss': 1, 'Mr': 2, 'Mrs': 3, 'Other': 4})

-> pandas의 map을 이용해서 직접 바꿔주는 방법.

```
import numpy as np
from sklearn.preprocessing import LabelEncoder
X_train = np.array(['PC', 'MOBILE', 'PC'       ])
X_test    = np.array(['PC', 'TABLET', 'MOBILE']) # X_test 에만 TABLET 데이터가 있음
# 라벨 인코더 생성
encoder = LabelEncoder()

# X_train 데이터를 이용 피팅하고 라벨숫자로 변환한다
encoder.fit(X_train)
X_train_encoded = encoder.transform(X_train)

# X_test 데이터에만 존재하는 새로 출현한 데이터를 신규 클래스로 추가한다 (중요!!!)
for label in np.unique(X_test):
    if label not in encoder.classes_: # unseen label 데이터인 경우( )
```

```
            encoder.classes_ = np.append(encoder.classes_, label) # 미처리 시 ValueError 발생
X_test_encoded = encoder.transform(X_test)
```

-> LabelEncoder를 이용해서 바꿔주는 방법.

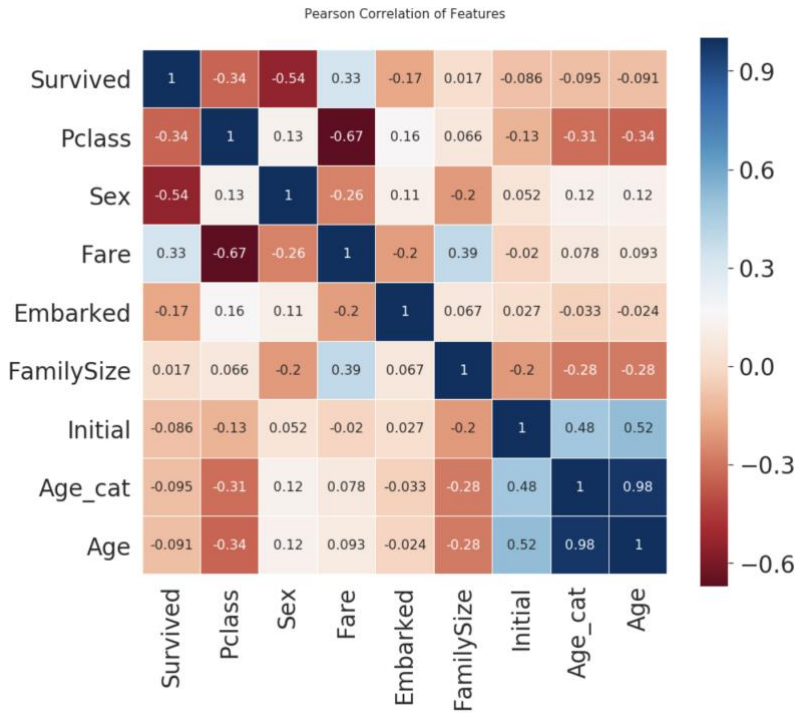
Label Encoding vs One Hot Encoding

Label Encoding	One Hot Encoding
숫자 이외의 Label 값을 가질 경우 사용 특정 Label 값에 따라 숫자로 변환	Loss 를 계산하기 쉽게 만들어주기 위해 벡터의 한개의 요소만 1 로 설정, 나머지는 0 으로 설정한다.
EX) KR, US, UK, CN 의 값을 가지는 feature KR -> 0 US -> 1 UK -> 2 CN -> 3	EX) KR, US, UK, CN 의 값을 가지는 feature KR -> (1, 0, 0, 0) US -> (0, 1, 0, 0) UK -> (0, 0, 1, 0) CN -> (0, 0, 0, 1)

상관관계 분석

data.corr()

-> 데이터 간의 상관관계를 알 수 있다.



-> 상관관계에서 주의할 점

1) 연속형(숫자로 표현 가능한) 데이터에 대해서만 상관관계 분석이 가능한점
2) -1 부터 1 까지의 값으로 표현됨
3) 인과관계를 뜻하진 않음.

Pandas apply() 함수

df.apply(np.average, axis=1)

-> 현재 df 를 연산에 맞게 새롭게 만듬.

```
df = pd.DataFrame({'A': [1,2,3], 'B': [4,5,6]}, index=['I1', 'I2', 'I3'])
df
```

	A	B
I1	1	4
I2	2	5
I3	3	6

```
def fun_s(df_temp):
    return df_temp.sum()

df.apply(fun_s,axis=1)
```

```
I1    5
I2    7
I3    9
dtype: int64
```

문제 풀이

Titanic tutorial

시작하기에 앞서 데이터 feature 들의 크기를 살펴보자.
-> df.describe() 함수를 이용하여 각 feature 들의 수, 평균 등을 확인한다.

null 값을 포함하고 있는 feature 에 대해 null 값을 채워줬지만, 그 다음 진행에 있어 막힘.

feautre A 에 따른 B 의 값을 보고 싶을 때 groupby 함수 사용한다.
groupby 는 좌측의 행의 기준을 정해준다고 생각하면 됨.