Feature Engineering(특성공학)

특정 애플리케이션에서 가장 적합한 데이터 표현을 찾는 것을 특성 공학(Feature Engineering)이라 한다.

학습 목표

가. get_dummies에 대해 실습을 통해 One-Hot-Encoding을 확인해 본다.

학습 내용

가. panadas에서의 One-Hot-Encoding 해보기

1-1. pandas를 이용한 One-Hot-Encoding

In [104]:
```
import pandas as pd
train = pd.read_csv("D:/dataset/data_titanic/train.csv", index_col=["PassengerId"])
print(train.shape)
train.head()
```

```
(891, 11)
```

Out[104]:

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

In [105]:
```
test = pd.read_csv("D:/dataset/data_titanic/test.csv", index_col=["PassengerId"])
print(test.shape)
test.head()
```

```
(418, 10)
```

Out[105]:

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
892	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
893	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
894	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
895	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

가. Preprocessing

In [106]:
```
train.loc[train["Sex"] == "male", "Sex"] = 0
train.loc[train["Sex"] == "female", "Sex"] = 1

print(train.shape)
train.head()
```

```
(891, 11)
```

Out[106]:

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
1	0	3	Braund, Mr. Owen Harris	0	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	1	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	1	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	1	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	0	35.0	0	0	373450	8.0500	NaN	S

In [107]:
```
test.loc[test["Sex"] == "male", "Sex"] = 0
test.loc[test["Sex"] == "female", "Sex"] = 1

print(test.shape)
test.head()
```

```
(418, 10)
```

Out[107]:

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
892	3	Kelly, Mr. James	0	34.5	0	0	330911	7.8292	NaN	Q
893	3	Wilkes, Mrs. James (Ellen Needs)	1	47.0	1	0	363272	7.0000	NaN	S
894	2	Myles, Mr. Thomas Francis	0	62.0	0	0	240276	9.6875	NaN	Q
895	3	Wirz, Mr. Albert	0	27.0	0	0	315154	8.6625	NaN	S
896	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	1	22.0	1	1	3101298	12.2875	NaN	S

In [108]:
```
print(train.info())
print(test.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
Survived    891 non-null int64
Pclass      891 non-null int64
Name        891 non-null object
Sex         891 non-null int64
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(4)
memory usage: 83.5+ KB
None
<class 'pandas.core.frame.DataFrame'>
Int64Index: 418 entries, 892 to 1309
Data columns (total 10 columns):
Pclass      418 non-null int64
Name        418 non-null object
Sex         418 non-null int64
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(4)
memory usage: 35.9+ KB
None
```

Fill in missing age

In [109]:
```
tr_mean_age = train["Age"].mean()
test_mean_age = test["Age"].mean()

train.loc[pd.isnull(train["Age"]), "Age"] = tr_mean_age
train[pd.isnull(train["Age"])]

test.loc[pd.isnull(test["Age"]), "Age"] = test_mean_age
test[pd.isnull(test["Age"])]
```

Out[109]:

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

In [110]:
```
# 요금
test_mean_Fare = test["Fare"].mean()

test.loc[pd.isnull(test["Fare"]), "Fare"] = test_mean_Fare
test[pd.isnull(test["Fare"])]
```

Out[110]:

PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked

In [111]:
```
print(train.info())
print(test.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
Survived    891 non-null int64
Pclass      891 non-null int64
Name        891 non-null object
Sex         891 non-null int64
Age         891 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(4)
memory usage: 83.5+ KB
None
<class 'pandas.core.frame.DataFrame'>
Int64Index: 418 entries, 892 to 1309
Data columns (total 10 columns):
Pclass      418 non-null int64
Name        418 non-null object
Sex         418 non-null int64
Age         418 non-null float64
SibSp       418 non-null int64
Parch       418 non-null int64
Ticket      418 non-null object
Fare        418 non-null float64
Cabin       91 non-null object
Embarked    418 non-null object
dtypes: float64(2), int64(4), object(4)
memory usage: 35.9+ KB
None
```

나. One-Hot-Encoding

In [112]:
```
print(train.columns)
```

```
Index(['Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket',
       'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [113]:
```
# train["Embarked_C"] = train["Embarked"] == "C"
# train["Embarked_S"] = train["Embarked"] == "S"
# train["Embarked_Q"] = train["Embarked"] == "Q"

# test["Embarked_C"] = test["Embarked"] == "C"
# test["Embarked_S"] = test["Embarked"] == "S"
# test["Embarked_Q"] = test["Embarked"] == "Q"

sel_col = ['Pclass' , 'Sex', 'Age', 'SibSp', 'Parch', 'Fare',  'Embarked' ]
X_train = train[sel_col]
X_train.head()
```

Out[113]:

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
1	3	0	22.0	1	0	7.2500	S
2	1	1	38.0	1	0	71.2833	C
3	3	1	26.0	0	0	7.9250	S
4	1	1	35.0	1	0	53.1000	S
5	3	0	35.0	0	0	8.0500	S

In [114]:
```
print(test.columns)

X_test = test[sel_col]
X_test.head()
```

```
Index(['Pclass', 'Name', 'Sex', 'Age', 'SibSp', 'Parch', 'Ticket', 'Fare',
       'Cabin', 'Embarked'],
      dtype='object')
```

Out[114]:

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
892	3	0	34.5	0	0	7.8292	Q
893	3	1	47.0	1	0	7.0000	S
894	2	0	62.0	0	0	9.6875	Q
895	3	0	27.0	0	0	8.6625	S
896	3	1	22.0	1	1	12.2875	S

get_dummies()

가. pandas에서는 get_dummies 함수를 사용해서 데이터 인코딩이 가능하다.
나. 희소행렬(sparse의 기본은 False)
다. get_dummies함수는 숫자 자료형은 연속형이라고 판단하여 가변수를 만들지 않음.
   -> 이 경우 Scikit-learn의 OneHotEncoder를 사용할 수 있다.

In [118]:
```
X_train = pd.get_dummies(X_train)
print(list(X_train.columns))
X_train.head()
```

```
['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked_C', 'Embarked_Q', 'Embarked_S']
```

Out[118]:

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_C	Embarked_Q	Embarked_S
1	3	0	22.0	1	0	7.2500	0	0	1
2	1	1	38.0	1	0	71.2833	1	0	0
3	3	1	26.0	0	0	7.9250	0	0	1
4	1	1	35.0	1	0	53.1000	0	0	1
5	3	0	35.0	0	0	8.0500	0	0	1

In [119]:
```
## test 데이터
X_test = pd.get_dummies(X_test)
print(list(X_test.columns))
X_test.head()
```

```
['Pclass', 'Sex', 'Age', 'SibSp', 'Parch', 'Fare', 'Embarked_C', 'Embarked_Q', 'Embarked_S']
```

Out[119]:

PassengerId	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_C	Embarked_Q	Embarked_S
892	3	0	34.5	0	0	7.8292	0	1	0
893	3	1	47.0	1	0	7.0000	0	0	1
894	2	0	62.0	0	0	9.6875	0	1	0
895	3	0	27.0	0	0	8.6625	0	0	1
896	3	1	22.0	1	1	12.2875	0	0	1

다. 모델 적용

In [117]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score

# feature 선택
# X_train, X_test 위에서 선택됨.
label_name = "Survived"
y_train = train[label_name]
print(X_train.shape, y_train.shape)
#== 데이터 셋 나누기
X_tr, X_te, y_tr, y_te = train_test_split(X_train, y_train, random_state=0)

print(X_tr.shape, y_tr.shape)

model = LogisticRegression()
model.fit(X_tr, y_tr)

prediction = model.predict(X_te)
prediction[1:5]

accuracy_score(y_te, prediction)
```

```
(891, 9) (891,)
(668, 9) (668,)
```

Out[117]:
```
0.7982062780269058
```

라. Test 데이터 셋으로 예측 후, 제출해보기

In [102]:
```
prediction = model.predict(X_test)
prediction[1:5]

submission = pd.read_csv("D:/dataset/data_titanic/gender_submission.csv", index_col="PassengerId")

submission["Survived"] = prediction

print(submission.shape)
submission.head()
submission.to_csv("D:/dataset/data_titanic/result_script.csv")
```

```
(418, 1)
```

실습1

OneHotEncoding 의 내용을 나의 현재 프로젝트 데이터 셋에 적용시켜보자.

적용시킨 내용을 간단하게 ppt로 정리해보자.

In [ ]:
```

```