Kaggle2

목차 및 출처

3. feature engineering - 모델을 세우기에 앞서, 모델의 성능을 높일 수 있도록 feature 들을 engineering 합니다. one-hot encoding, class로 나누기, 구간으로 나누기, 텍스트 데이터 처리 등을 합니다.
4. model 만들기 - sklearn 을 사용해 모델을 만듭니다. 파이썬에서 머신러닝을 할 때는 sklearn 을 사용하면 수많은 알고리즘을 일관된 문법으로 사용할 수 있습니다. 물론 딥러닝을 위해 tensorflow, pytorch 등을 사용할 수 도 있습니다.
5. 모델 학습 및 예측 - trainset 을 가지고 모델을 학습시킨 후, testset 을 가지고 prediction 합니다.
6. 모델 평가 - 예측 성능이 원하는 수준인지 판단합니다. 풀려는 문제에 따라 모델을 평가하는 방식도 달라집니다. 학습된 모델이 어떤 것을 학습하였는 지 확인해봅니다.

- 출처 : https://kaggle-kr.tistory.com/17?category=868316
- https://kaggle-kr.tistory.com/18?category=868316

3. Feature engineering(Fill Null)

```
# 호칭을 추출한다.
train['Initial']= train.Name.str.extract('([A-Za-z]+)₩.')
test['Initial']= test.Name.str.extract('([A-Za-z]+)₩.')

# 성별 별로 호칭을 보여준다.
pd.crosstab(train['Initial'], train['Sex']).T.style.background_gradient(cmap='summer_r')
```

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

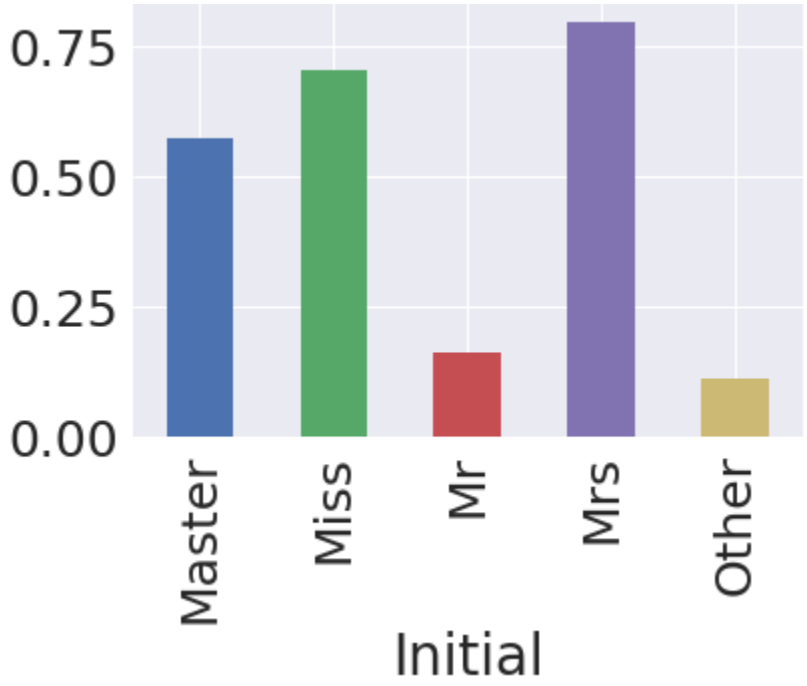
3. Feature engineering(Fill Null)

```
# 호칭을 바꾼다.(이전 값, 바꿀 값)
train['Initial'].replace(['Mlle','Mme','Ms','Dr','Major','Lady','Countess','Jonkheer','Col','Rev','Capt','Sir','Don', 'Dona'],
                         ['Miss','Miss','Miss','Mr','Mr','Mrs','Mrs','Other','Other','Other','Mr','Mr','Mr', 'Mr'],inplace=True)

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

train.groupby('Initial').mean()
```

Initial	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare	FamilySize
Master	414.975000	0.575000	2.625000	4.574167	2.300000	1.375000	3.340710	4.675000
Miss	411.741935	0.704301	2.284946	21.860000	0.698925	0.537634	3.123713	2.236559
Mr	455.880907	0.162571	2.381853	32.739609	0.293006	0.151229	2.651507	1.444234
Mrs	456.393701	0.795276	1.984252	35.981818	0.692913	0.818898	3.443751	2.511811
Other	564.444444	0.111111	1.666667	45.888889	0.111111	0.111111	2.641605	1.222222

3. Feature engineering(Fill Null Age)

```
train.groupby('Initial')['Survived'].mean().plot.bar()
```




```
# column Age에 null값이 있다면 그 값을 Initial의 평균값으로 채워넣는다.
train.loc[(train.Age.isnull())&(train.Initial=='Mr'),'Age'] = 33
train.loc[(train.Age.isnull())&(train.Initial=='Mrs'),'Age'] = 36
train.loc[(train.Age.isnull())&(train.Initial=='Master'),'Age'] = 5
train.loc[(train.Age.isnull())&(train.Initial=='Miss'),'Age'] = 22
train.loc[(train.Age.isnull())&(train.Initial=='Other'),'Age'] = 46

test.loc[(test.Age.isnull())&(test.Initial=='Mr'),'Age'] = 33
test.loc[(test.Age.isnull())&(test.Initial=='Mrs'),'Age'] = 36
test.loc[(test.Age.isnull())&(test.Initial=='Master'),'Age'] = 5
test.loc[(test.Age.isnull())&(test.Initial=='Miss'),'Age'] = 22
test.loc[(test.Age.isnull())&(test.Initial=='Other'),'Age'] = 46
```

3. Feature engineering(Fill Null Embarked)

```
print('Embarked has ', sum(train['Embarked'].isnull()), ' Null values')
train['Embarked'].fillna('S', inplace=True)
```

Null 개수 확인 및 'S'(가장 많은 값)으로 변경

3. Feature engineering(Change Age)

```
def category_age(x):
    if x < 10:
        return 0
    elif x < 20:
        return 1
    elif x < 30:
        return 2
    elif x < 40:
        return 3
    elif x < 50:
        return 4
    elif x < 60:
        return 5
    elif x < 70:
        return 6
    else:
        return 7

train['Age_cat_2'] = train['Age'].apply(category_age)
```

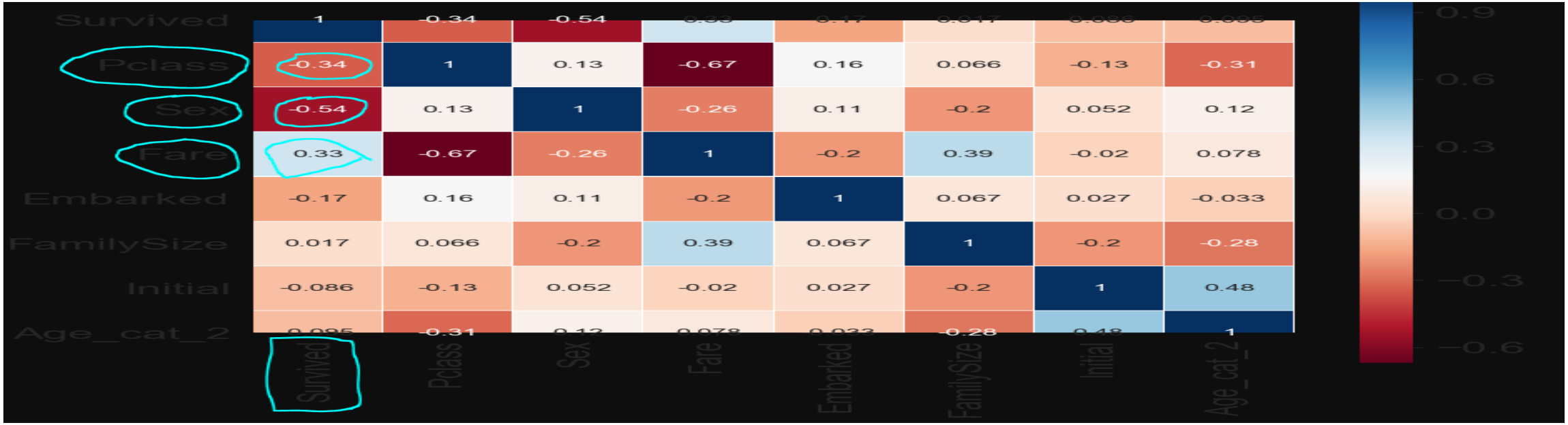
Train의 'Age' column의 값들에 대해 category_age 함수를 적용시킨다.

3. Feature engineering(Change String to numerical)

```
train['Initial'] = train['Initial'].map({'Master': 0, 'Miss': 1, 'Mr': 2, 'Mrs': 3, 'Other': 4})
test['Initial'] = test['Initial'].map({'Master': 0, 'Miss': 1, 'Mr': 2, 'Mrs': 3, 'Other': 4})
train['Embarked'] = train['Embarked'].map({'C': 0, 'Q': 1, 'S': 2})
test['Embarked'] = test['Embarked'].map({'C': 0, 'Q': 1, 'S': 2})
train['Sex'] = train['Sex'].map({'female': 0, 'male': 1})
test['Sex'] = test['Sex'].map({'female': 0, 'male': 1})
```

데이터 학습을 위해 String을 숫자로 바꿔준다.

3. Feature engineering(변수간상관관계)



(-1 to 1)
-1 음의 상관관계
: 한 변수가 증가 시 다른 변수가 감소
0은 상관없음을 의미
1 양의 상관관계
: 한 변수가 증가 시 다른 변수도 증가

$$r_{xy} = \frac{Cov(x,y)}{S_x S_y} = \frac{\frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})}{S_x S_y}$$

Pclass, Sex, Fare가 Survived와 관련성이 있다.

3. Feature engineering(one-hot encoding)

- 데이터.unique()를 (0,1)로 이루어지게 매핑

	Initial_Master	Initial_Miss	Initial_Mr	Initial_Mrs	Initial_(
Master	1	0	0	0	0
Miss	0	1	0	0	0
Mr	0	0	1	0	0
Mrs	0	0	0	1	0
Other	0	0	0	0	1

```
# 가변수 생성, prefix : 이름앞에 일정하게 붙이는 값
train = pd.get_dummies(train, columns=['Initial'], prefix='Initial')
test = pd.get_dummies(test, columns=['Initial'], prefix='Initial')
print(train.head())

train = pd.get_dummies(df_train, columns=['Embarked'], prefix='Embarked')
test = pd.get_dummies(df_test, columns=['Embarked'], prefix='Embarked')
```

Initial은 5개(Master, Miss, Mr, Mrs, Other)가 존재하므로 2^5로 표현

Initial_0	Initial_1	Initial_2	Initial_3	Initial_4
0	0	1	0	0
0	0	0	1	0
0	1	0	0	0
0	0	0	1	0
0	0	1	0	0

3. Feature engineering(drop columns)

- 필요 없는 columns 드랍

```
# drop columns
train.drop(['PassengerId', 'Name', 'SibSp', 'Parch', 'Ticket', 'Cabin'], axis=1, inplace=True)
test.drop(['PassengerId', 'Name',  'SibSp', 'Parch', 'Ticket', 'Cabin'], axis=1, inplace=True)
```

4. Building machine learning model and prediction using the trained model

```
#importing all the required ML packages
from sklearn.ensemble import RandomForestClassifier # 유명한 randomforestclassfier 입니다.
from sklearn import metrics # 모델의 평가를 위해서 씁니다
from sklearn.model_selection import train_test_split # traning set을 쉽게 나눠주는 함수입니다.
```

```
X_train = train.drop('Survived', axis=1).values
target_label = train['Survived'].values
X_test = test.values
X_tr, X_vld, y_tr, y_vld = train_test_split(X_train, target_label, test_size=0.3, random_state=2018)
```

1. 학습에 쓰일 데이터와 target table(Survived)를 분리한다. 보통 train과 test만 언급되지만 valid set을 따로 만들면 더 좋은 모델을 만들 수 있다. 축구 대표팀이 팀훈련(train)을 하고 바로 월드컵(test)에 나가는 것이 아니라 평가전(valid)를 거친 뒤 월드컵(test)에 나가는 것과 같다.

4. Building machine learning model and prediction using the trained model

```
# 모델(hypothesis)
model = RandomForestClassifier()
model.fit(X_tr, y_tr)
# 예측
prediction = model.predict(X_vld)
# 정확도
print('총 {}명 중 {:.2f}% 정확도로 생존을 맞춤'.format(y_vld.shape[0], 100 * metrics.accuracy_score(prediction, y_vld)))
```

```
총 268명 중 83.21% 정확도로 생존을 맞춤
```

2. 모델을 세우고(랜덤포레스트) 예측한 뒤 성능평가

4. Building machine learning model and prediction using the trained model

```
# 목표 파일
submission = pd.read_csv('gender_submission.csv')
# 예측
prediction = model.predict(X_test)
submission['Survived'] = prediction
# 파일저장
submission.to_csv('my_first_submission.csv', index=False)
```

3. 정답 파일 저장

기타 주의사항

- train과 test 파일은 똑같은 필터링을 해줘야함