**Machine Learning (2021 Fall semester)**

**Programming Assignment: Classification of Titanic Data Set**

**2021712943 이찬종**

**1. Benchmark Dataset**:

**이번 과제에서는 모델 학습 및 테스트 모두 train.csv파일을 이용합니다.**

텍스트이(가) 표시된 사진

자동 생성된 설명

**과제를 수행하실 때 다음의 feature들은 모델 학습에 이용하지 않도록 주의해주세요.**

**제외할 feature: PassengerId, Name, Ticket, Cabin**

텍스트이(가) 표시된 사진

자동 생성된 설명

**2. Preprocessing**

**2-1. train data에 결측치(NULL)가 있는 feature들이 있습니다. 이 값들을 어떻게 처리할 것인지 아이디어를 제시하고 실제로 구현하세요.**

-> 'Age' column에선 다수의, 'Embarked' column에선 극소수의 샘플에 대해 NULL값이 있음.  
a) 'Embarked'는 (눈으로 세어보니 2개) 워낙 갯수도 적고, 생존 여부에 영향을 주는 attribute이면서도 평균이나 중간값 등으로 처리하기 애매하므로 그냥 해당 sample은 drop할 생각. 하지만 지금 drop해버리면 pre-processing의 과정이 번거로워지므로 현재는 None 문자열로 잠깐 바꿔 두었다가 2-4) 파트에서 drop 예정  
  
b) 'Age'에 column에서 나오는 NULL 갯수는 무시할 수 없을 만큼 많은데, 그렇다고 전부 다 평균이나 중간값으로 대체하는 것은 실제 데이터(기록되진 않았지만 그 사람의 실제 나이)와는 괴리가 너무 커질 것으로 예상. 따라서 NULL값은 최빈값처럼 취급하고자, csv 파일을 살펴본 결과 가장 다수를 차지하는 20~30대의 범주로 포함시키기 위해 28로 설정

텍스트이(가) 표시된 사진

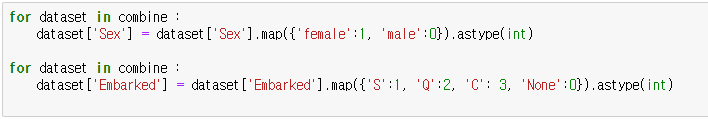
자동 생성된 설명

**2-2. One-Hot encoding을 수행할 필요가 있는 feature들이 무엇이며 그 이유는 무엇인지 서술하세요. 또한 실제로 어떻게 구현했는지 보고서에 나타내세요.**

-> scikit-learn을 사용할 때 문자열로 나타나는 feature들은 Label이든 One-Hot이든 숫자로 encoding해 줘야 함. 그 중에서도 'Sex', 'Embarked' 을 대상으로 One-Hot encoding이 필요해 보여 이를 수행. 이유는 'Pclass'는 3 -> 1로 갈수록 더 좋은 선실을 의미하여 숫자의 크기에 따라서도 의미가 달라지지만, 'Sex', 'Embarked'같은 feature의 경우는 숫자의 크기에 따른 의미 차이가 없기 때문텍스트이(가) 표시된 사진

자동 생성된 설명

one-hot encoding을 하기 위해 우선 labeling을 먼저 시행



그 다음 카테고리별로 column을 새로 만들며 one-hot encoding

텍스트이(가) 표시된 사진

자동 생성된 설명

**2-3. train.csv의 sample을 7대 3으로 학습 데이터와 테스트 데이터로 사용하세요.**

텍스트이(가) 표시된 사진

자동 생성된 설명

(ipynb 코드 상에선 **1. Benchmark Dataset** 의 내용에 곧이어 작성) “train.csv” 파일 중간에 column 이름을 나타내는 row를 한 번 더 삽입한 후 pandas read\_csv함수의 nrows, skiprows 옵션을 이용해 “train.csv”만으로 train\_df와 test\_df를 나누어 실습을 진행하였음

**2-4. 그 외 진행한 전처리 과정이 있다면 서술하세요.**

-> 범주에 따른 분류가 가능해야 하므로, 원래는 연속적 성질을 가진 feature인 'Age'와 'Fare'를 일정 구간 별로 나누어 범주화 시켰음. (One Hot Encoding까지 진행시킨 경우도 테스트해 보았는데, Labeling 까지만 한 경우와 비교했을 때 Logistic regression의 accuracy가 조금 높아졌을 뿐, 나머지 모델에서의 전반적인 accuracy는 범주화 까지만 진행한 경우가 더 높았기에 보고서는 이를 바탕으로 작성)

텍스트이(가) 표시된 사진

자동 생성된 설명

그래프도 참고하고, csv 파일에서 대강 범위 별 개수를 세어본 것을 바탕으로 Age는 20 40 60 기준으로 4덩이로 나누고(20 미만, 20 <= 나이 <40 이런 식) Fare는 21.0 53.0 75.0 100.0 이렇게 5덩이로 범주화 하였음

텍스트이(가) 표시된 사진

자동 생성된 설명

-> 마지막으로 'Embarked\_NaN'이 1로 되어있는 row들을 날린 후, 원래 존재하지 않았어야 했던 'Embarked\_NaN'도 날려버림으로써 pre-processing 완료

텍스트이(가) 표시된 사진

자동 생성된 설명

**3. Machine Learning Models: scikit-learn을 이용해 후술할 세 가지 machine learning model를 구현하고 성능을 평가하세요.**

각 모델에 fit 하기 전의 준비 단계

텍스트이(가) 표시된 사진

자동 생성된 설명

ipynb 파일의 거의 처음 부분에서 진행하긴 했지만 필요한 각종 import도 수행했음

텍스트이(가) 표시된 사진

자동 생성된 설명

**3-1 K-Nearest Neighbors(KNN): K(이웃)의 개수를 [1~5]까지 변화시키면서 test data에서 결과가 어떻게 변하는지 분석하세요.** 테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

K를 3으로 설정했을 때의 accuracy가 가장 높았고, f1 score의 경우는 K를 5로 설정했을 때 가장 높은 값을 보임. K가 커지거나 작아질수록 accuracy 및 f1 score가 규칙성을 갖고 변화한다기보다는, 다양한 K값을 시도해보면서 가장 높은 performance를 보여주는 K를 찾는 것이 좋은 방법으로 보인다.

**3-2 Logistic Regression: Iteration 횟수를 [0~100] 범위에서 20씩 변화시키면서 test data에서 결과가 어떻게 변하는지 분석하세요. Iteration 횟수를 100으로 고정한 후 regularization term(scikit-learn에서는 C)를 [0~5]의 범위에서 1씩 변화시키면서 test data에서 결과가 어떻게 변하는지 분석하세요**.

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

Iteration이 0일 때 현격히 낮은 성능을 보여준다. Iter가 20일 때 그나마 좋은 성능을 보여주고, 이후는 Iteration 횟수를 늘린다고 더 좋은 결과를 보여주거나 하진 않고 똑같은 결과를 보여줌. 0이 아닌 Iter값을 점점 숫자를 늘려가며 시도하다 최고의 성능을 보이는 Iteration에서 멈추는 것이 현명한 방법으로 보인다.

텍스트이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

C가 0일 때는 깔끔하게 division by zero 에러가 남. 양수 값으로만 써야 함을 알 수 있음

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명

C가 0일 때를 제외하곤 전부 고만고만한 성능을 보인다. 이 또한 특별한 규칙성을 갖지는 않아서, 성능 높은 C값을 찾아가야 하는 것으로 보인다.

**3-3 Decision Tree: Information Gain을 통해 test data에서 결과가 어떻게 나오는지 분석하세요. 또한 적절한 tool을 이용하여 각 depth에서의 조건과 gain값을 알 수 있도록 tree를 시각화하세요.**

각 depth에서의 information을 시각화하기 위해 criterion = ‘entropy’를 설정

테이블이(가) 표시된 사진

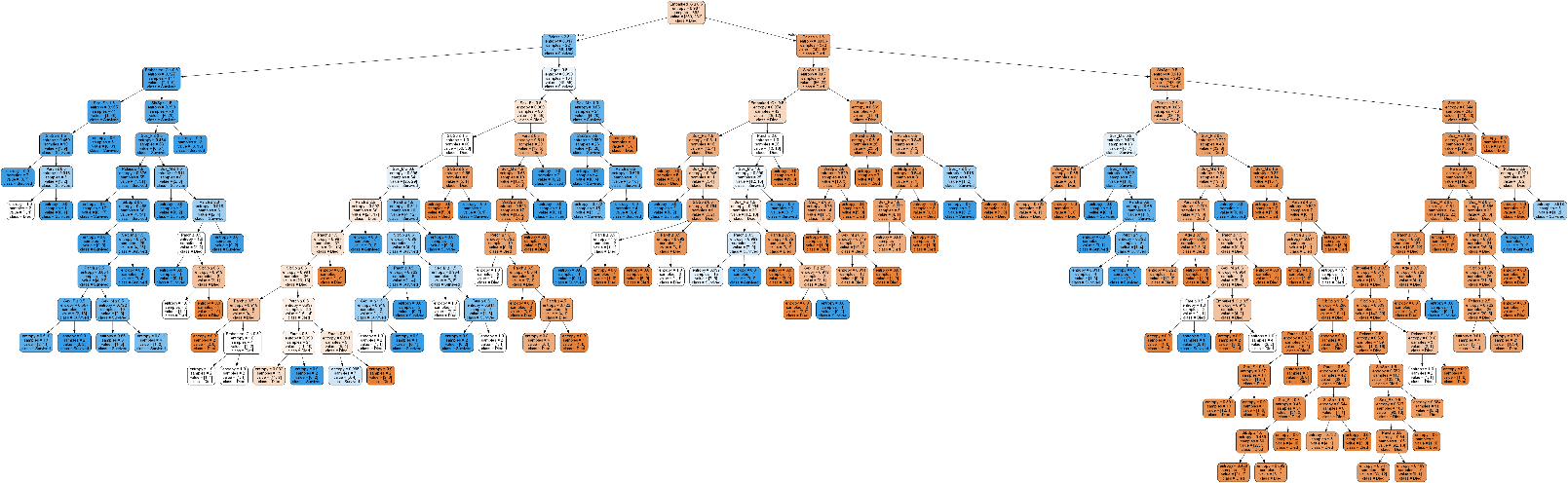
자동 생성된 설명

Accuracy 측면에선 현재까지 model 중 가장 뛰어난 성능을 보인다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Jupyter notebook 브라우저 상에서도 이미지가 나오긴 하지만, 위 코드를 통해 생성되었을 ‘tree.png’ 를 열어 봄으로써 더 자세히 확인할 수 있음



**3-3-1 Bagging with Decision Tree: Decision Tree 기반으로 Bagging 기법을 이용하여 bag의 수에 따라 test data에서 결과가 어떻게 변하는지 decision tree와 비교하여 분석하세요. Bag의 수는 [1~5]에서 1씩 변화시키세요.**

테이블이(가) 표시된 사진

자동 생성된 설명 테이블이(가) 표시된 사진

자동 생성된 설명테이블이(가) 표시된 사진

자동 생성된 설명 테이블이(가) 표시된 사진

자동 생성된 설명

테이블이(가) 표시된 사진

자동 생성된 설명 앞선 모델들과 구별되는 특이한 점은, 러닝 및 score()를 실행할 때마다 매번 Accuracy값이 바뀐다는 것. 하지만 Bag 의 값이 클수록 평균적으로(똑같은 n\_estimator로 여러 번 실행해보았을 때) 더 높은 Accurracy를 보인다는 점을 알 수 있음

**4. Evaluation Methods: 각각 모델에 따른 성능을 Accuracy, F1-Score을 통해 나타내세요.**

**3. Machine Learning Models** 파트에서 각 모델별로 함께 기술하였음