기계학습 중간고사 대체과제

프로젝트 보고서

202021497 생명공학과 이주철

- 지도학습을 이용한 분류 모델 구현 및 분석 –

제출일: 2025-04-26

1. 개발 환경 설정

1) 개발 환경

- 운영체제: Windows 11

- 언어: Python 3.13.2

2) 사용 라이브러리

- pandas: 데이터 불러오기, 정리 (read_csv, DataFrame, 결측치 처리 등)

- Numpy: 수치 계산, 결측값 처리 (np.nan, np.abs 등)

- matplotlib.pyplot: 결과 시각화 (산점도, 히트맵 등)

- seaborn: 상관관계 히트맵, countplot 등 고급 시각화

- scikit-learn (sklearn): 머신러닝 전처리, 모델, 평가 전체를 하기 위한 패키지

2. 프로젝트 개요

1) **문제 정의**: 버섯의 독성 여부를 버섯의 특성에 따라 예측하는 분류 문제를 정의합니다.

2) **데이터셋 설명**: 버섯의 생김새, 무늬, 냄새 같은 특성에 따라 독성과 식용 가능한 버섯을 예측할 수 있습니다.

- 출처: https://www.kaggle.com/datasets/rinichristy/uci-mushroom-dataset

- 데이터셋 크기: 8124 * 23

- 종속변수(target) 및 주요 독립변수(features):

종속 변수: Mushroom_quality (e: 식용, p: 독성)

독립 변수: odor, gill_size, spore_print_color, ring_type, gill_spacing, bruises 등으로 구성되어있습니다.

3. 데이터 전처리 및 탐색적 분석 (EDA)

1) 결측치 처리

- 사용한 방법: '?'값을 최빈값으로 대체하여 진행했습니다.

2) 범주형 변수 처리

범주형 변수를 명목형과 서열형으로 분류하여 인코딩을 진행하였습니다. 명목형 변수는 LabelEncoder를 사용하여 문자를 순서에 관여 받지 않게 숫자로 바꿔 분류하였습니다. 서열형 변수는 순서를 반영하여 수작업 매핑 후 넓고 좁음을 구별하였습니다.

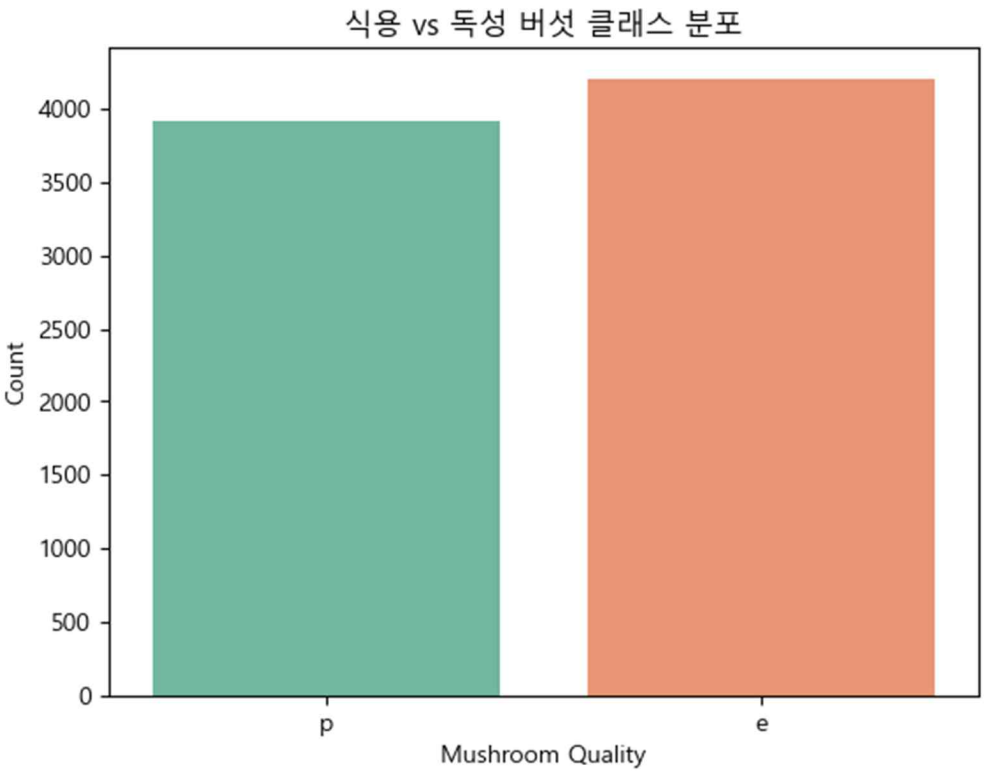
명목형 변수는 OneHotEncoder(handle_unknown="ignore")를 사용하여 각 범주를 고유한 벡터로 변환했습니다. 서열형 변수는 순서를 반영하여 수치로 변환한 후, StandardScaler를 적용하여 평균 0, 표준편차 1로 정규화하였습니다.

- 인코딩 방식: OneHotEncoder, StandardScaler

- 스케일링: Tree 기반 모델(RandomForest)에서는 스케일링이 필요 없지만, Logistic Regression 모델에서는 수치형 피처의 스케일링이 모델 성능에 영향을 줄 수 있어 적용하였습니다. (모든 전처리는 파이프라인 내부에서 자동 처리되었습니다.)

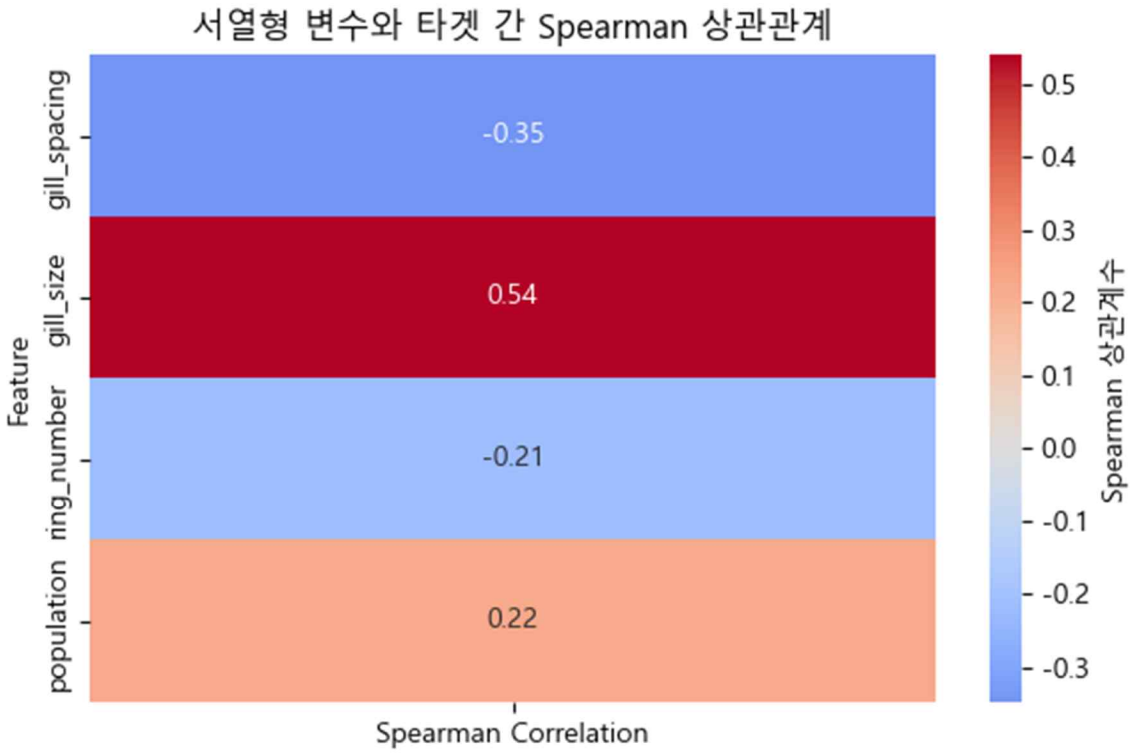
3) EDA 시각화:

- 클래스 분포 시각화: countplot을 사용하여 Mushroom_quality(식용/독성) 클래스 분포를 시각화하였습니다.



- 주요 변수 분포: 명목형 변수로 식용/독성 클래스를 그룹화하여 각 범주 내 비율(ratio)을 계산한 후, barplot을 통해 시각화하였습니다. 이를 통해 odor, gill_size 등 주요 변수들이 식용/독성 여부에 미치는 영향을 분석하였습니다.

- 상관관계 분석: 서열형 변수와 타겟(Mushroom_quality) 간의 관계를 분석하기 위해 spearmanr 상관계수를 계산하였고, 이를 heatmap으로 시각화하였습니다. 이를 통해 서열형 변수 중 타겟과 강한 상관관계를 갖는 변수를 식별할 수 있습니다.



4) 모델 구축 및 학습

- 사용한 알고리즘: RandomForestClassifier, LogisticRegression

- 데이터 분할 방식: train_test_split(test_size=0.2, stratify=y, random_state=42)를 사용하여 학습용/테스트용 데이터를 8:2 비율로 분할하였습니다.

- 파이프라인 사용 여부: Pipeline과 ColumnTransformer를 활용하여 전처리와 모델 학습을 통합 처리하였습니다.
명목형 변수는 OneHotEncoder(handle_unknown="ignore")를 사용하여 인코딩하고, 서열형 변수는 수치 변환 후 StandardScaler를 적용하여 스케일링을 진행하였습니다.
모든 전처리는 파이프라인 내부에서 자동으로 수행되었습니다.

- 학습 코드 요약

```
preprocessor = ColumnTransformer(
    transformers=[
        ('onehot', OneHotEncoder(...), nominal_columns),
        ('scale', StandardScaler(), ordinal_columns)
    ]
)
```

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.linear_model import LogisticRegression

model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', RandomForestClassifier())  # or LogisticRegression()
])
```

4. 성능 평가

1) 사용한 지표:

- **Accuracy**: 전체 예측 중에서 정답을 맞춘 비율을 평가하였습니다.

- **Precision**: 독성 버섯(Positive)을 예측했을 때 실제 독성일 확률을 평가하였습니다.

- **Recall**: 실제 독성 버섯을 모델이 놓치지 않고 잘 예측했는지를 평가하였습니다.

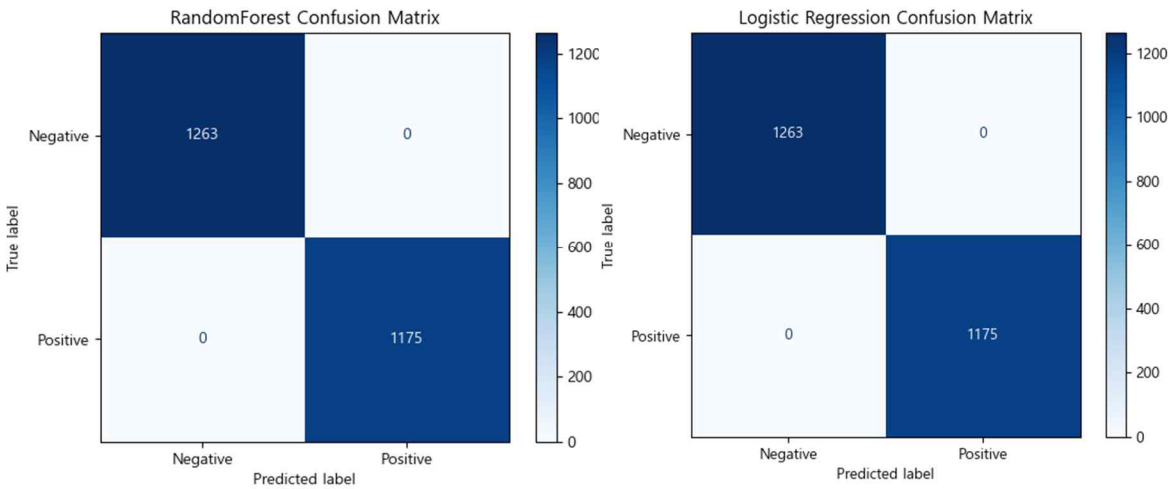
- **F1-score**: Precision과 Recall의 조화 평균으로, 두 지표의 균형을 평가하였습니다.

- **ROC-AUC (Area Under the Curve)**: 모델이 식용/독성 클래스를 구분하는 능력을 종합적으로 평가하였습니다.

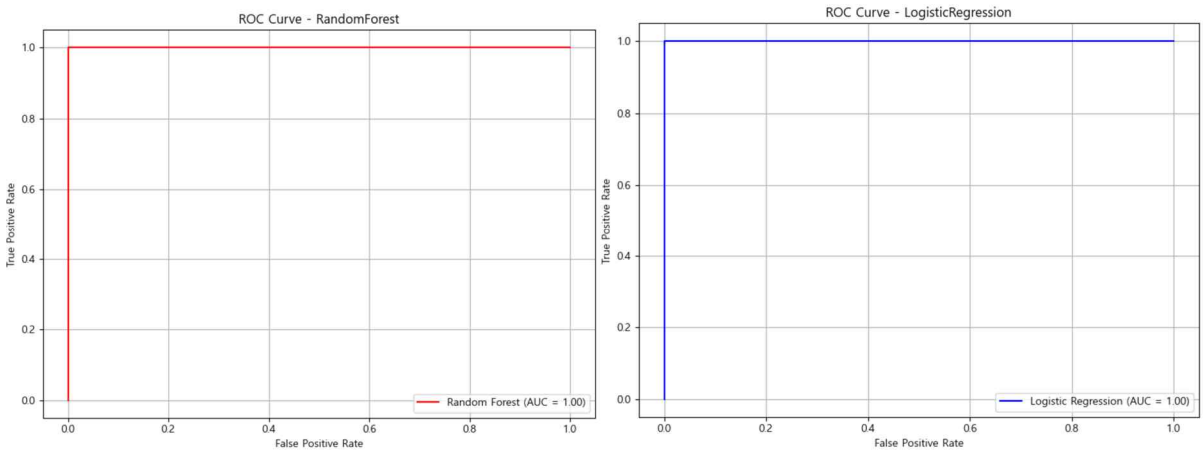
2) 예측 결과 시각화

- Confusion Matrix 시각화

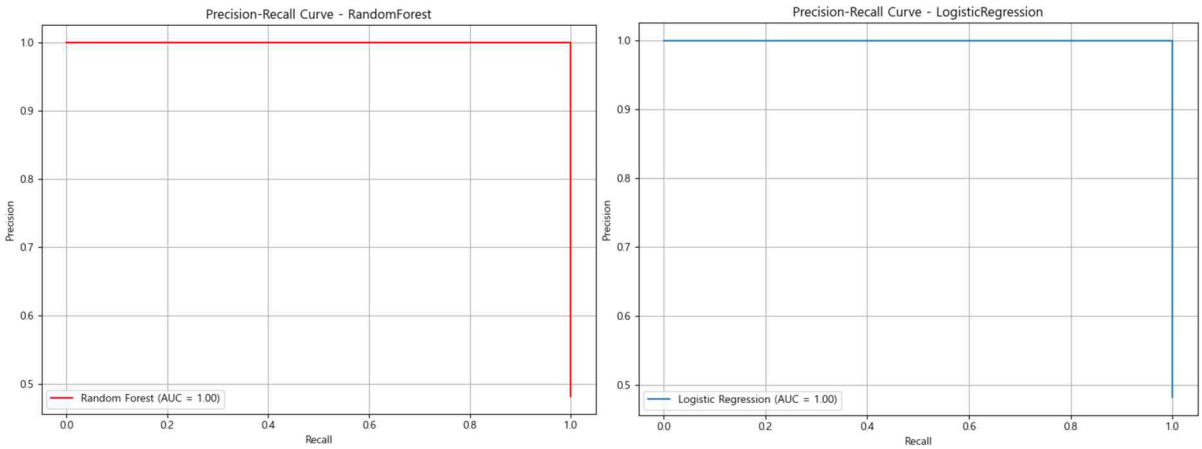
RandomForestClassifier와 LogisticRegression 각각에 대해 Confusion Matrix를 시각화하여, 실제 클래스와 예측 클래스 간의 관계를 한눈에 파악할 수 있도록 하였습니다.



- ROC Curve로 두 모델을 시각화 하였습니다.



- Precision-Recall Curve로 두 모델을 시각화 하였습니다.



3) 해석

- RandomForestClassifier와 LogisticRegression 두 모델을 사용하였을 때 두 모델 성능이 좋게 나왔습니다. 독성을 분명하게 예측할 거 같은 두 'odor', 'spore_print_color' 컬럼으로만 독성 예측을 진행하였을 때 LogisticRegression이 더 높게 나왔습니다. 이는 데이터셋이 잘 구분되어 있어서 위 모델이 정확도가 더 높게 나올 수 있다고 판단하였습니다.

```
Test Set 성능 (RandomForestClassifier)
- Accuracy: 0.9938
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      1263
           1       1.00      0.99      0.99      1175

    accuracy                           0.99      2438
   macro avg       0.99      0.99      0.99      2438
weighted avg       0.99      0.99      0.99      2438
```

```
Test Set 성능 (Logistic Regression)
- Accuracy: 0.9938
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      1263
           1       1.00      0.99      0.99      1175

    accuracy                           0.99      2438
   macro avg       0.99      0.99      0.99      2438
weighted avg       0.99      0.99      0.99      2438
```

- Confusion Matrix와 ROC Curve, Precision-Recall Curve를 분석한 결과, 식용(e)과 독성

(p) 클래스 모두 정확도와 정밀도 모두 유사하거나 완벽한 수준으로 나타났습니다.
→ 따라서 클래스 간 성능 차이는 거의 존재하지 않았으며, 모델이 두 클래스를 균형 있게 잘 분류했음을 확인할 수 있었습니다.

5. 하이퍼파라미터 튜닝

1) 튜닝 방법

- GridSearchCV를 활용하여 최적화하였습니다. 이를 통해 모든 조합을 체계적으로 탐색하였습니다.

2) 튜닝한 하이퍼파라미터

- 최적 파라미터 조합은 다음과 같습니다.

- RandomForest 모델을 사용했을 경우 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100

```
Fitting 5 folds for each of 108 candidates, totalling 540 fits
최적 파라미터: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators':
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      1263
           1       1.00      0.99      0.99      1175

    accuracy                           0.99      2438
   macro avg       0.99      0.99      0.99      2438
weighted avg       0.99      0.99      0.99      2438
```

- LogisticRegression 모델을 사용했을 경우 'C': 1, 'max_iter': 100, 'penalty': 'l2', 'solver': 'lbfgs'

```
Fitting 5 folds for each of 30 candidates, totalling 150 fits
최적 파라미터: {'C': 1, 'max_iter': 100, 'penalty': 'l2', 'solver': 'lbfgs'}
              precision    recall  f1-score   support

           0       0.99      1.00      0.99      1263
           1       1.00      0.99      0.99      1175

    accuracy                           0.99      2438
   macro avg       0.99      0.99      0.99      2438
weighted avg       0.99      0.99      0.99      2438
```

3) 튜닝 결과 분석

- **odor**, **gill_size**, **spore_print_color** 컬럼만을 가지고 튜닝하기 전 RandomForestClassifier, LogisticRegression 기본 모델 대비, 튜닝 된 모델은 Accuracy 및 F1-score에서 소폭 성능 향상을 보였습니다.

6. 결론 및 고찰

- 하이퍼파라미터 튜닝 결과, LogisticRegression 모델의 Test Set 정확도는 튜닝 전과 비교하여 유의미한 변화는 없었습니다. 이는 초기 설정에서도 모델이 데이터셋에 적합하게 학습되어 있음을 말해준다고 판단하였습니다.

- 주어진 데이터셋은 odor와 spore_print_color 등 일부 feature에 의존하는 경향이 매우 컸기 때문에, 상대적으로 쉽게 분류가 되었습니다.

- 이 모델은 버섯 독성 여부를 빠르게 판별하는 데 활용할 수 있으며, 식품 안전 분야, 야생 탐색 앱 등에 적용 가능하다고 생각합니다.

- 본 데이터는 매우 깨끗한 상태였지만, 실제 상황에서는 클래스 불균형 문제를 고려하고, 결측치 또한 최빈값 이외의 다양한 방법으로 처리하는 방안을 검토할 필요가 있다습니다.

7. 참고자료

- https://www.kaggle.com/datasets/rinichristy/uci-mushroom-dataset

- 참고한 논문, 블로그, scikit-learn 문서 등

- 문단 정리 및 코드 가독성을 위해 GPT 사용하였습니다.

- 혼자 공부하는 머신러닝, 딥러닝 (교재를 보며 코드를 해석하고 인용하였습니다.)