# 2025 학년도 2 학기 출석수업대체과제물(온라인 제출용)


❍ 교 과 목 명 : 파이썬컴퓨팅
❍ 학 번 : 202334-153257
❍ 성 명 : 양희석
❍ 연 락 처 : 010-4340-2326


※ A4용지 편집 사용

※ 과제물 표지등에 개인정보(주민번호,운전면허번호)가 포함될 경우 삭제처리로 과제물을 다시 제출해야 하는 경우가 발생할 수 있습니다.

---

[1]

## 1. Scikit-learn의 개념과 목적

### 가. 개념: 통합된 머신러닝 개발 환경

Scikit-learn은 파이썬 프로그래밍 언어 기반의 오픈소스 머신러닝 라이브러리이다. NumPy, SciPy, Matplotlib과 같은 과학 계산용 라이브러리를 기반으로 구축되어, 데이터 분석 및 머신러닝 작업을 위한 강력하고 통합된 환경을 제공한다. Scikit-learn의 핵심 철학은 **'일관성(Consistency)', '단순성(Simplicity)', '효율성(Efficiency)'**으로 요약할 수 있다. 이는 다양한 머신러닝 알고리즘을 동일한 인터페이스(fit, predict, transform)로 사용할 수 있게 하여, 개발자가 알고리즘의 내부 구현보다 문제 해결 자체에 집중할 수 있도록 돕는다.

### 나. 목적: 머신러닝의 대중화와 생산성 극대화

Scikit-learn의 주된 목적은 복잡한 머신러닝 알고리즘을 누구나 쉽게 접근하고 활용할 수 있도록 하는 것이다. 과거 머신러닝 모델을 구현하기 위해서는 깊은 수학적 지식과 복잡한 코딩이 필요했지만, Scikit-learn은 이를 추상화하여 몇 줄의 코드로도 분류, 회귀, 군집화, 차원 축소 등 다양한 작업을 수행할 수 있게 한다.

궁극적으로 Scikit-learn은 데이터 전처리부터 모델 학습, 평가, 튜닝에 이르는 머신러닝 워크플로우 전반을 지원함으로써, 연구자와 개발자의 생산성을 극대화하고, 프로토타이핑부터 실제 서비스 배포까지의 과정을 단축시키는 것을 목표로 한다.

## 2. 주요 기능 및 특징

### 가. 일관성 있는 API와 포괄적인 알고리즘 제공

Scikit-learn의 가장 큰 특징은 일관된 API 디자인이다. 모든 모델(Estimator)은 다음과 같은 공통된 메서드를 따른다.

fit(X, y): 훈련 데이터 X와 레이블 y를 사용하여 모델을 학습시킨다.
predict(X): 학습된 모델을 사용하여 새로운 데이터 X의 결과를 예측한다.
transform(X): 데이터를 변환한다. (주로 전처리 단계에서 사용)

fit_transform(X, y): fit과 transform을 동시에 수행하여 효율성을 높인다.
이러한 일관성은 개발자가 Support Vector Machine(SVM), RandomForest, K-Means 등 전혀 다른 알고리즘을 사용하더라도 코드 구조를 거의 변경하지 않고 모델을 교체하며 실험할 수 있게 한다. 이는 모델 성능 비교 및 최적 모델 선택 과정을 매우 용이하게 만든다.

또한, Scikit-learn은 전통적인 머신러닝 분야의 거의 모든 알고리즘을 포함하는 포괄적인 라이브러리이다.

분류 (Classification): SVM, Logistic Regression, Decision Tree, RandomForest 등
회귀 (Regression): Linear Regression, Ridge, Lasso, SVR 등
군집 (Clustering): K-Means, DBSCAN, Hierarchical Clustering 등
차원 축소 (Dimensionality Reduction): PCA, t-SNE, NMF 등

이처럼 방대한 알고리즘을 하나의 라이브러리에서 일관된 방식으로 제공하는 것은 Scikit-learn의 강력한 경쟁력이다.

**나. 머신러닝 워크플로우 전체를 지원하는 도구 체계**

Scikit-learn은 단순히 모델링 알고리즘만 제공하는 것을 넘어, 머신러닝 프로젝트의 전체 사이클을 지원하는 풍부한 유틸리티를 내장하고 있다.

데이터 전처리 (sklearn.preprocessing):

StandardScaler, MinMaxScaler: 피처 스케일링을 통해 모델 성능을 향상시킨다.
OneHotEncoder, LabelEncoder: 범주형 데이터를 수치형 데이터로 변환한다.
Imputer: 결측치를 처리한다.

모델 선택 및 평가 (sklearn.model_selection, sklearn.metrics):

train_test_split: 데이터를 훈련 세트와 테스트 세트로 분리한다.
cross_val_score: 교차 검증을 통해 모델의 일반화 성능을 안정적으로 평가한다.
GridSearchCV, RandomizedSearchCV: 하이퍼파라미터 튜닝을 자동화하여 최적의 모델 파라미터를 탐색한다.
accuracy_score, f1_score, mean_squared_error: 분류, 회귀 등 문제 유형에 맞는 다양한 평가 지표를 제공한다.

파이프라인 (sklearn.pipeline):

데이터 전처리, 피처 선택, 모델 학습 등 여러 단계를 하나의 Pipeline 객체로 묶어 코드의 가독성과 재사용성을 높인다.
특히 교차 검증 시 훈련 데이터의 정보가 검증 데이터로 유출(Data Leakage)되는 것을 방지하여, 모델 평가의 신뢰도를 높이는 데 결정적인 역할을 한다.

```
# 파이프라인 예시: 데이터 스케일링과 SVM 모델 학습을 하나로 묶음
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVC

pipe = Pipeline([
    ('scaler', StandardScaler()),
    ('svm', SVC(kernel='rbf', C=1.0))
])
# 이제 pipe.fit(X_train, y_train) 한 줄로 전처리 및 학습이 모두 수행된다.
```

## 3. 실제 활용 사례

Scikit-learn은 그 안정성과 범용성 덕분에 학계뿐만 아니라 산업계 전반에서 폭넓게 활용되고 있다.

### 가. Spotify: 음악 추천 시스템

세계적인 음원 스트리밍 서비스인 Spotify는 사용자의 음악 청취 패턴을 분석하여 새로운 음악을 추천하는 데 머신러닝을 적극 활용한다. Scikit-learn은 이러한 추천 시스템의 프로토타이핑 및 일부 모델링에 사용될 수 있다. 예를 들어, 사용자의 청취 기록, 장르 선호도, 아티스트 팔로우 정보 등을 피처로 사용하여 특정 플레이리스트에 대한 사용자의 선호도를 예측하는 분류(Classification) 문제를 풀 수 있다. 또한, 음원의 음향적 특징(템포, 에너지, 춤 가능성 등)을 분석하여 유사한 곡들을 그룹화하는 군집화(Clustering) 알고리즘(예: K-Means)을 적용하여 '유사한 곡 추천' 기능을 구현하는 데 활용된다.

### 나. 금융권: 사기 탐지 및 고객 신용 평가

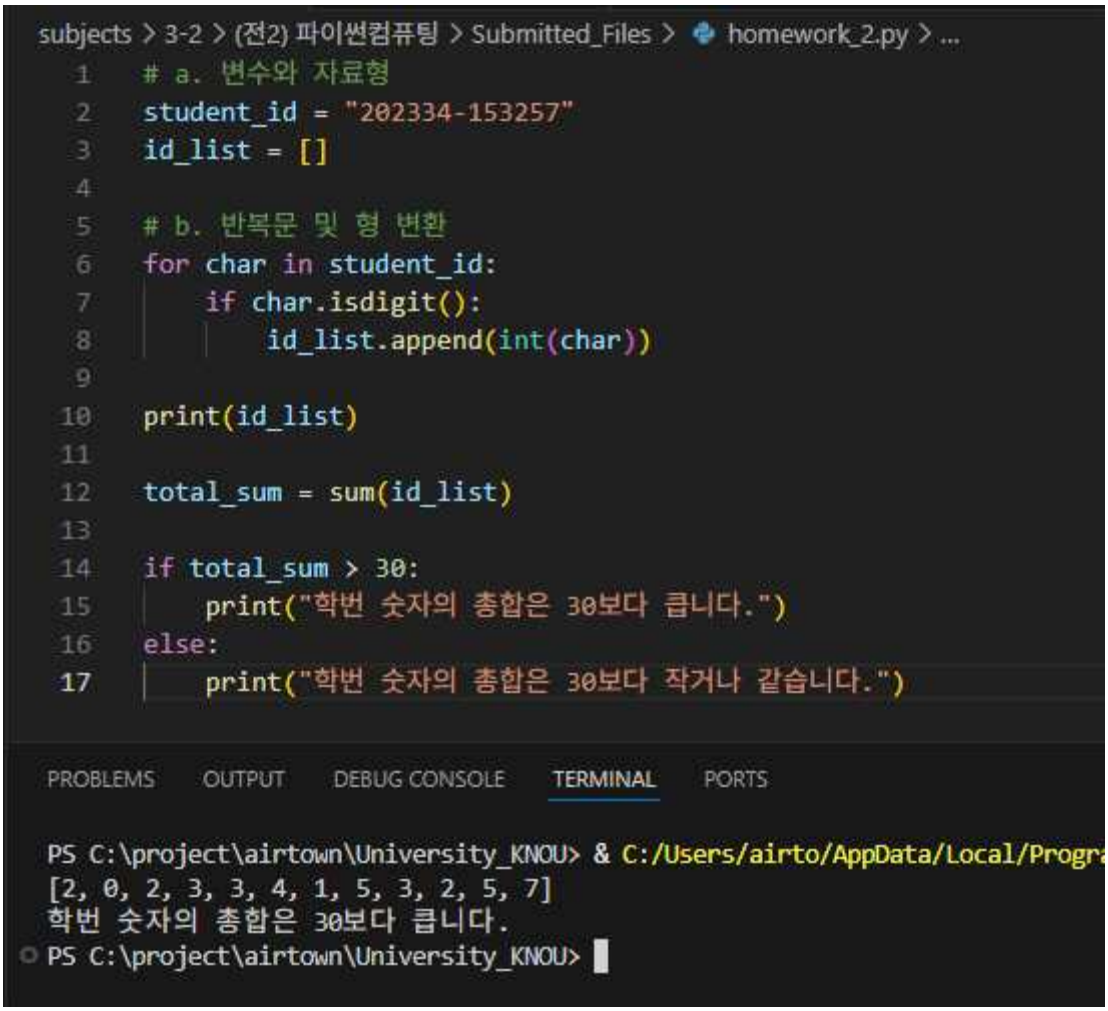
금융 분야에서는 비정상적인 거래 패턴을 탐지하는 사기 탐지 시스템(Fraud Detection System) 구축에 Scikit-learn이 널리 사용된다. 이는 소수의 사기 거래(이상치)를 다수의 정상 거래 속에서 찾아내는 이상치 탐지(Anomaly Detection) 문제로, One-Class SVM이나 Isolation Forest와 같은 Scikit-learn의 알고리즘이 효과적으로 사용된다. 또한, 고객의 수입, 직업, 연체 기록 등 다양한 정보를 바탕으로 대출 상환 능력을 예측하는 신용 평가 모델을 구축하는 데 Logistic Regression이나 Gradient Boosting과 같은 분류 모델이 활용된다.

### 다. Evernote: 이미지 내 텍스트 분류

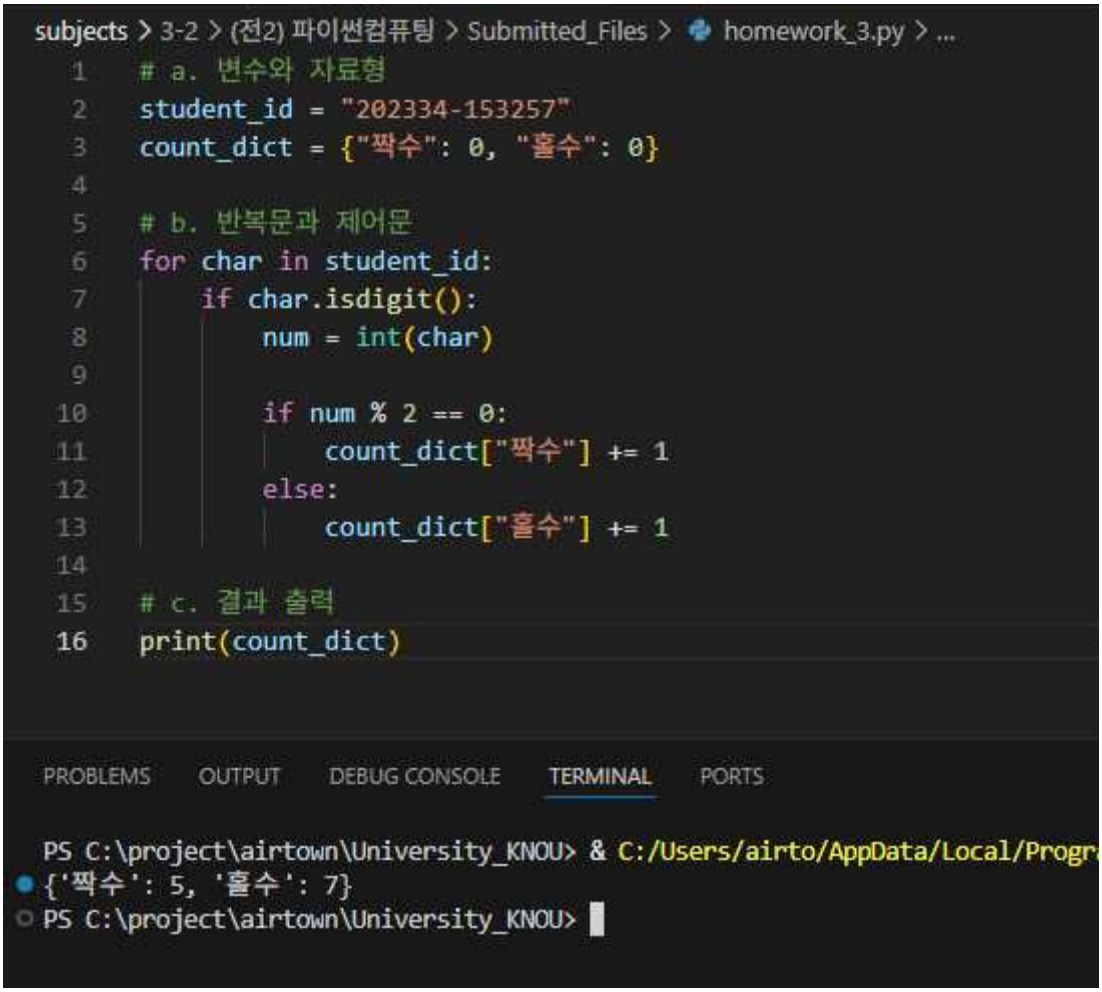
클라우드 노트 서비스인 Evernote는 사용자가 업로드한 이미지 속 텍스트를 인식하고 자동으로 분류하는 기능에 머신러닝을 사용한다. 이미지에서 추출된 텍스트(OCR)를 자연어 처리(NLP) 기술로 분석하여 문서의 주제를 예측한다. 이 과정에서 Scikit-learn의 TfidfVectorizer를 사용하여

텍스트를 수치 벡터로 변환하고, Naive Bayes나 LinearSVC와 같은 분류 모델을 적용하여 '영수증', '명함', '메모' 등 문서의 종류를 자동으로 태깅하는 기능을 구현할 수 있다. 이는 사용자가 수많은 노트를 효율적으로 관리하도록 돕는다.

[2]




[3]


subjects > 3-2 > (전2) 파이썬컴퓨팅 > Submitted_Files > homework_3.py > ...
# a. 변수와 자료형
student_id = "202334-153257"
count_dict = {"짝수": 0, "홀수": 0}
# b. 반복문과 제어문
for char in student_id:
if char.isdigit():
num = int(char)
if num % 2 == 0:
count_dict["짝수"] += 1
else:
count_dict["홀수"] += 1
# c. 결과 출력
print(count_dict)
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
PORTS
PS C:\project\airtown\University_KNOU> & C:/Users/airto/AppData/Local/Progr
{'짝수': 5, '홀수': 7}
PS C:\project\airtown\University_KNOU>