2주차 예비보고서

전공 : 경영학과 학년 : 4학년 학번 : 20190808 이름 : 방지혁

1. 다음 링크에 있는 Machine Learning에 대한 정의와 설명을 읽고, 다음 2가지 질문에 대한 자신만의 생각을 적으세요.

1-1. Supervised Learning과 Unsupervised Learning의 차이점을 서술하세요.

Supervised learning은 지도학습으로 컴퓨터에게 입력 데이터와 정답(라벨)을 같이 학습시키는 방법입니다. 이는 비지도학습보다 더 단순하고 일반적입니다. 우리는 정답을 미리 알려주면 모델은 입력 데이터의 특징을 분석하여 해당 특징을 통해 분류하는 방법을 스스로 배웁니다. 이는 출력 중 연속적인 값을 출력하는 회귀에 사용될 수 있습니다. 회귀의 예시로는 환자의 입원 및 수술 기간을 예측하는 것이 있겠습니다. 또한 입력 데이터가 어떤 종류의 값인지 예측하는 분류에 사용될 수 있습니다. 분류의 예시로는 새의 종류를 예측하는 것, 손글씨로 쓴 숫자를 예측하는 것이 있으며 머신러닝 입문 시 가장 많이 접하게 되는 MNIST 데이터베이스가 대표적 사례라고 볼 수 있겠습니다.

반면 unsupervised learning은 비지도 학습으로 지도학습과 달리 정답이 없는 데이터를 입력시킵니다. 이는 정답을 주지 않고 숨겨진 패턴 혹은 구조를 파악하기 위해서입니다. 이는 clustering, association, dimensionality reduction과 같은 세 가지 작업에 사용될 수 있습니다. Clustering이란 데이터들을 데이터 사이의 거리에 따라 군집들로 나누는 것으로 고객 구매 패턴 분석, 이미지 압축 등에 사용됩니다. Association은 입력된 데이터에서 변수 간의 관계를 찾는 것으로 추천 엔진에 사용됩니다. Dimensionality reduction은 차원 수가 너무 많을 때 무결성을 유지하면서도 크기를 줄이는 것으로 노이즈를 제거하는 데이터 전처리 단계에서 사용됩니다.

결국 주요 차이점은 데이터에 label이 지정되어 있는지 아닌지 여부입니다. 또한 목표의 측면에서 보자면 지도 학습은 높은 정확도로 결과를 예측하는 것이지만 비지도 학습은 데이터에 대해 우리가 미처 발견하지 못한 패턴을 통해 통찰력을 얻는 것입니다.

1-2. 모델을 학습한다는 것에 대한 과정을 서술하세요.

우선 **문제정의**를 해야 합니다. 데이터를 토대로 무엇을 얻고자 하는지 정의를 해야 합니다. 이 정의가 이루어져야 어떤 데이터, 알고리즘 및 어떻게 평가할지 결정할 수 있게 될 것입니다. 그 다음으로는 **데이터 준비 및 전처리**가 있습니다. 전처리에 대해 설명하자면 데이터는 학습되기 전에 노이즈 제거, 결측치 처리, 이상치 탐지 및 제거를 해야 합니다. 또한, 스케일을 맞추기 위해 모든 변수를 비슷한 범위로 조정하는 표준화, 정규화 등의 과정을 거쳐야 합니다. 세 번째는 **모델 선택 및 구축**입니다. 문제의 성격 및 데이터의 특성, 컴퓨터 자원에 따라 적절한 모델을 선택하는 것입니다. 또한, 알고리즘에 따라 적절한 매개변수의 크기, 노드 개수, 레이어의 크기, 활성함수와 손실함수 중 어떤 것을 사용할 것인지도 결정해야합니다. 네 번째는 **반복 학습**입니다. 훈련 데이터를 활용하여 이루어지며 함수를 사용하여 손실을 계산합니다. 이를 줄여나가기 위해 파라미터를 계속 조정해나갑니다. 마지막은 **평가**입니다. 실제로 이 모델이 얼마나 잘 작동하는지 확인하는 것으로 이전까지 사용하지 않은 새로운 데이터를 사용해야 합니다. 만약 평가 기준에 부합하지 않는다면 이전 단계로 돌아가는 것을 반복하며 계속 개선해 나가야 합니다.

2. Python에서 사용되는 테이블 형태의 데이터를 다루기 위한 라이브러리인 `Numpy`, `Pandas`, `Matplotlib`에 대해 조사하고, 각 라이브러리는 어떤 용도로 만들어졌으며 어떤 특징을 가지고 있는지 자세하게 정리하세요.

Numpy는 python의 수치 계산 라이브러리입니다. Import numpy as np로 축약해서 호출하여 사용할 수 있습니다. 기존의 python의 리스트 자료형은 느리고 메모리 측면에서 비효율적이었습니다. 예를 들어 수백만개의 숫자로 이루어진 두 리스트를 더하려면 하나씩 순회해야 했기 때문입니다. 그러나 numpy의 경우 상당부분 C로 작성되어 빠르기에 이를 해결할 수 있습니다. 특히 ndarray(다차원배열) 처리에 특화되어 벡터화된 연산을 통해 반복문 없이 한번에 수행할 수 있습니다. 또한 선형 대수, 푸리에 변환 등 여러 복잡한 수학적 연산을 위한 함수를 제공합니다.

Pandas는 테이블 형태의 데이터에 특화된 라이브러리입니다. Import pandas as pd로 축약해서 호출하여 사용할 수 있습니다. CSV, SQL등 다양한 포맷을 지원하여 데이터 수집을 간편하게 해줍니

다. 실제 데이터는 결측치가 있거나 중복된 데이터가 포함되어 있기 마련인데 pandas는 결측치 찾기 및 제거, 복잡한 데이터 필터링 및 그룹화의 기능을 제공합니다. 그렇기에 머신러닝 단계 중 데이터 전처리 과정에서 매우 유용합니다. 또한, Numpy를 기반으로 구축되어 있어 매우 빠른 연산 속도를 보여주며 테이블 형태 데이터에 특화된 인터페이스를 제공합니다.

Matlplotlib는 데이터 시각화에 특화된 라이브러리입니다. 핵심모듈인 pyplot을 import matplotlib.pyplot as plt로 축약해서 호출하여 사용할 수 있습니다. 이름에서 알 수 있듯이 matlab의 인터페이스를 python으로 구현한 것에서 시작했습니다. 사용자는 선의 스타일, 두께, 눈금 등 그래프의 여러 요소를 커스텀할 수 있습니다. 또한, 기본적인 선 그래프, 막대 그래프, 히스토그램 등 여러 종류의 그래프를 지원하며 여러 개의 그래프를 한 화면에서 배치할 수도 있습니다. 또한, 머신러닝에서 역할을 설명하자면 EDA 단계에서 데이터의 분포를 빠르게 파악하는데 유용하며 성능을 평가할 때 ROC 곡선 같은 시각적 도구를 제공합니다.

3. Python에서 머신러닝에 사용되는 `Scikit-learn` 라이브러리를 조사하고, scikit-learn에서 제공하는 모델을 3가지 이상 찾아, 각 모델의 사용법에 대해 자세히 서술하세요.

Scikit-learn은 유명한 파이썬 머신러닝 라이브러리 중 하나입니다. 아나콘다라는 개발환경을 설치하는 것만으로 쉽게 이용할 수 있습니다. 단 numpy, scipy, pandas와 같은 라이브러리에 대한 설치도 필요합니다. 분류, 회귀, 군집화 등 다양한 머신러닝 문제를 해결하기 위한 도구를 제공합니다.

Linear regression은 앞선 언급한 지도 학습의 한 종류인 회귀 문제에 사용됩니다. 여러 입력 변수와 타겟 변수 간의 선형 관계를 모델링하여 새로운 데이터에 대한 예측을 하는 것입니다. 사용방법에 대해 설명하자면 from sklearn.linear_model import LinearRegression로 모델을 불러옵니다. 이후 model = LinearRegression()로 모델 객체를 생성하는데 이는 특별한 매개변수 설정 없이도 가능합니다. 그 후 model.fit(X_train, y_train)로 모델을 학습시킵니다. X_train은 학습 데이터이고 y_train은 정답 데이터로 이를 인자로 받는 것입니다. 학습이 완료되면 predictions = model.predict(X_test)로 새로운 테스트 데이터에 대한 예측값을 반환받습니다. 해당 모델은 해석

하기 쉽고 빠르다는 장점이 있습니다. model.coef_와 model.intercept_를 통해 계수와 절편도 확인 가능합니다.

K-Means Clustering은 비지도 학습의 군집화에 사용됩니다. 입력된 데이터를 k개의 클러스터로 나누고자 하며 내재된 패턴을 스스로 찾아냅니다. 사용방법에 대해 설명하자면 from sklearn.cluster import KMeans로 모델을 불러옵니다. 이후 모델 객체를 생성하는데 model = KMeans(n_clusters=3, random_state=42) 이러한 방법으로 n_clusters라는 변수로 몇 개의 클러스터로 나눌지 결정합니다. random_state 변수는 랜덤 시드 값을 설정하는 매개변수입니다. 비지도 학습이기에 이전과 다르게 model.fit(X)로 입력데이터만 사용합니다. 학습이 완료되면 labels = model.labels_로 각 데이터가 어떤 클러스터에 속하는지 확인할 수 있습니다. 또한, model.cluster_centers_로 중심점 좌표도 얻을 수 있습니다.

RandomForest는 분류와 회귀문제 둘 다 가능합니다. 여러 독립적인 결정 트리를 만들어 예측 결과를 종합하여 최종 결정을 내립니다. 사용방법에 대해 설명하자면 from sklearn.ensemble import RandomForestClassifier로 모델을 불러옵니다. 마찬가지로 모델 객체를 생성하는데 model = RandomForestClassifier(n_estimators=100, random_state=42)로 모델을 생성합니다. Random_state는 앞서 설명한 바와 같이 랜덤 시드 값이면 n_estimators는 결정트리의 개수를 의미합니다. 마찬가지로 fit()과 predict()를 사용합니다. 지도학습이기에 model.fit(X_train, y_train)로 학습 및 정답 데이터 두 개를 모두 인자로 받습니다. model.predict(X_test)로 새로운 데이터에 대한 클래스를 예측합니다. model.feature_importances_를 통해 각 변수가 얼마나 중요한 역할을 하는지 확인 가능합니다.