<코드 설명서 - 모델 학습 (IsolationForest)>

| FDS 모델 학습 (IsolationForest) 소스 코드 설명서 | |
| --- | --- |
| 작성자 | 김동훈 |
| 작성 일자 | 2024/09/23 |
| 팀명 | Sniffers |

| 순서 | 작업 내용 | 페이지 |
| --- | --- | --- |
| 1 | 모델 학습 | 2 |
| 모델 선택 | 2 |
| 모델 초기화 | 3 ~ 4 |
| 모델 학습 | 5 ~ 6 |
| 2 | 모델 검증 | 7 |
|  | 예측 수행 및 성능 평가 | 7 ~ 8 |
|  | 하이퍼파라미터 튜닝 | 9 ~ 11 |
| 3 | 모델 테스트 | 12 |
|  | 예측 수행 및 최종 성능 평가 | 12 ~ 13 |
| 4 | 모델 배포 | 14 |
|  | 모델 저장 및 로드 | 14 ~ 15 |

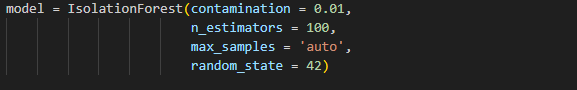
# <1. 모델 학습>

<1.1. 모델 선택>



| 기능 | |
| --- | --- |
| 이상 탐지에 사용할 모델을 선택한다. | |
| 코드 설명 | |
| from sklearn.ensemble import IsolationForest | sklearn.ensemble에서 IsolationForest를 임포트한다. |
| 입력 | |
| 없음 | |
| 출력 | |
| 없음 | |
| 매개변수 | |
| 없음 | |
| 예외 처리 | |
| 없음 | |
| 알고리즘 | |
| IsolationForest는 앙상블 기반 이상 탐지 알고리즘이다. | |
| 여러 개의 Isolation Tree를 생성하여 데이터 포인트를 격리한다. | |
| 정상 데이터 포인트는 Isolation Tree에서 깊은 곳에 위치하고, 이상치는 얕은 곳에 위치하는 경향이 있다. | |
| 추가 설명 | |
| Isolation Forest는 비지도 학습 알고리즘으로, 레이블이 없는 데이터에서 이상치를 탐지할 수 있다. | |
| 다른 이상 탐지 모델 (One-Class SVM, Local Outlier Factor)과 비교하여 Isolation Forest는 대용량 데이터셋에서도 빠르고 효율적으로 이상치를 탐지할 수 있다는 장점이 있다. | |
| IsolationForest는 특히 고차원 데이터 셋에서 이상치 탐지에 효과적이다. | |

<1.2. 모델 초기화>



| 기능 | |
| --- | --- |
| 선택한 IsolationForest 모델을 초기화하고, 하이퍼파라미터를 설정하여 모델을 커스터마이징 한다. 이 단계에서 설정하는 하이퍼파라미터는 모델의 학습 방식과 성능에 영향을 미친다. | |
| 코드 설명 | |
| model = IsolationForest(contamination = 0.01, n\_estimators = 100, max\_samples = ‘auto’, random\_state = 42) | IsolationForest 모델을 초기화한다.  - contamination: 이상치 비율을 설정한다.  - n\_estimators: Isolation Tree의 개수를 설정한다.  - max\_samples: 각 Isolation Tree를 학습할 때 사용할 샘플 수를 설정한다.  - random\_state: 랜덤 시드를 설정하여 결과를 재현 가능하도록 한다. |
| 입력 | |
| 없음 (모델 초기화 단계이므로 입력 데이터를 사용하지 않는다.) | |
| 출력 | |
| 초기화된 IsoLationForest 모델 객체 | |
| 매개변수 | |
| contamination | 이상치 비율 (float) |
| n\_estimators | Isolation Tree 개수 (int) |
| max\_samples | 각 Isolation Tree 학습에 사용할 샘플 수 (int 또는 ‘auto’) |
| random\_state | 랜덤 시드 (int 또는 None) |
| 예외 처리 | |
| contamination | 값이 0과 0.5 사이의 float 형식이 아닌 경우 ValueError를 발생시킨다. |
| n\_estimators | 값이 int 형식이 아닌 경우 TypeError를 발생시킨다. |
| max\_samples | 값이 int 또는 ‘auto’ 형식이 아닌 경우 TypeError를 발생시킨다. |
| random\_state | 값이 int 또는 None 형식이 아닌 경우 TypeError를 발생시킨다. |
| 알고리즘 | |
| IsolationForest는 앙상블 기반 이상 탐지 알고리즘이다. 여러 개의 Isolation Tree를 생성하여 데이터 포인트를 격리한다. 정상 데이터 포인트는 Isolation Tree에서 깊은 곳에 위치하고, 이상치는 얕은 곳에 위치하는 경향이 있다. | |
| 추가 설명 | |
| IsolationForest는 비지도 학습 알고리즘으로, 레이블이 없는 데이터에서 이상치를 탐지할 수 있다. | |
| 하이퍼파라미터를 조정하여 모델의 성능을 개선할 수 있다. | |
| contamination, n\_estimators, max\_samples, random\_state 외에도 다양한 하이퍼파라미터를 설정할 수 있다. | |

# 

<1.3. 모델 학습>



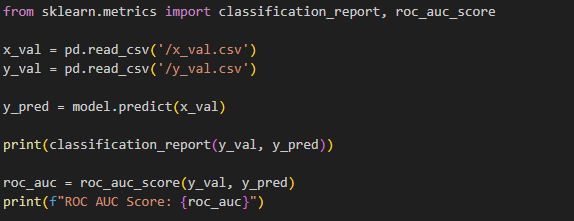
| 기능 | |
| --- | --- |
| 초기화된 IsolationForest 모델에 x\_train 데이터를 입력하여 모델을 학습시킨다. 모델 학습은 IsolationForest 알고리즘을 사용하여 데이터의 정상적인 패턴을 학습하고, 이를 기반으로 이상치를 탐지할 수 있도록 모델을 조정하는 과정이다. | |
| 코드 설명 | |
| model.fit() | 초기화된 model 객체의 fit() 메서드를 호출하여 모델을 학습시킨다. |
| x\_train | 학습에 사용할 데이터셋으로, 이 데이터는 모델이 정상적인 패턴을 학습하는 데 사용된다. |
| 입력 | |
| x\_train | 학습 데이터셋 (pandas DataFrame) |
| 출력 | |
| 학습된 IsolationForest 모델 객체 (자기자신) - fit() 메서드는 학습된 모델 자체를 반환한다. | |
| 매개변수 | |
| 없음 (fit() 메서드는 입력 데이터셋 외에 별도의 매개변수를 받지 않는다. | |
| 예외 처리 | |
| 입력 데이터셋(x\_train)의 형식이 올바르지 않은 경우(예: pandas DataFrame이 아닌 경우) TypeError가 발생할 수 있다. | |
| 입력 데이터셋에 결측값이나 이상값이 포함된 경우 ValueError가 발생할 수 있다. | |
| 알고리즘 | |
| IsolationForest 알고리즘은 데이터 포인트를 격리하기 위해 여러 개의 Isolation Tree를 생성한다. 각 Isolation Tree는 데이터셋에서 무작위로 선택된 특성과 분할 값을 사용하여 데이터를 재귀적으로 분할한다. 정상 데이터 포인트는 Isolation Tree에서 깊은 곳에 위치하고, 이상치는 얕은 곳에 위치하는 경향이 있다. 모델 학습 과정에서 IsolationForest는 Isolation Tree들을 생성하고, 각 데이터 포인트의 이상치 점수를 계산한다. | |
| 추가 설명 | |
| 모델 학습은 IsolationForest 모델을 사용하여 이상 탐지 시스템을 구축하는 데 있어 가장 중요한 단계이다. | |
| 학습 데이터셋의 품질은 모델의 성능에 큰 영향을 미치므로, 학습 데이터셋을 신중하게 선택하고 전처리 해야한다. | |
| fit() 메서드는 모델을 학습시키고, 학습된 모델은 이후 이상치 탐지에 사용된다. | |
| 학습된 모델은 predict() 메서드를 사용하여 새로운 데이터에 대한 이상치 여부를 예측할 수 있다. | |

# 

# 

# <2. 모델 검증>

<2.1. 예측 수행 및 성능 평가>



| 기능 | |
| --- | --- |
| 학습된 IsolationForest 모델을 사용하여 x\_val 데이터에 대한 예측을 수행하고, 예측 결과와 실제 값(y\_val)을 비교하여 모델의 성능을 평가한다. 이 단계는 모델의 과적합을 방지하고 일반화 성능을 평가하는 데 중요한 역할을 한다. | |
| 코드 설명 | |
| from sklearn.metrics import classification\_report, roc\_auc\_score | sklearn.metrics 모듈에서 classification\_report 함수와 roc\_auc\_score 함수를 가져온다.  classification\_report 함수는 정밀도, 재현율, F1-score 등의 분류 성능 지표를 계싼하고, roc\_auc\_socre 함수는 ROC AUC 점수를 계싼하는 데 사용된다. |
| x\_val = pd.read\_csv(‘/x\_val.csv’) | pandas 라이브러리의 read\_csv() 함수를 사용하여 지정된 경로의 CSV 파일을 읽어와 pandas DataFrame 형태로 x\_val 변수에 저장한다. 이는 검증에 사용될 데이터를 불러오는 과정이다. |
| y\_val = pd.read\_csv(‘y\_val.csv’) | x\_val과 동일한 방식으로 검증 데이터의 실제 값을 나타내는 y\_val 데이터를 불러온다. |
| y\_pred = model.predict(x\_val) | 학습된 모델(model)의 predict() 메서드를 사용하여 검증 데이터(x\_val)에 대한 예측을 수행한다. 예측 결과는 y\_pred 변수에 저장된다. |
| print(classification\_report(y\_val\_pred)) | classification\_report() 함수를 사용하여 실제 값(y\_val)과 예측 값(y\_pred)을 비교하여 정밀도, 재현율, F1-score 등의 분류 성능 지표를 계싼하고 겨로가를 출력한다. |
| roc\_auc = roc\_auc\_score(y\_val, y\_pred) | roc\_auc\_score() 함수를 사용하여 실제 값(y\_val)과 예측 값(y\_pred)을 기반으로 ROC AUC 점수를 계싼하고 roc\_auc 변수에 저장한다. ROC AUC 점수는 이진 분류 모델의 성능을 평가하는 데 널리 사용되는 지표이다. |
| 입력 | |
| x\_val | 검증 데이터셋 (pandas DataFrame) |
| y\_val | 검증 데이터셋의 실제 값 (pandas Series 또는 numpy array) |
| 출력 | |
| y\_pred | 검증 데이터셋에 대한 예측 결과 (numpy array) |
| 성능 평가 지표 (classification report, ROC AUC score) - 콘솔에 출력된다. | |
| 매개변수 | |
| 없음 (코드에서 직접적으로 매개변수를 사용하지 않는다. classification\_report()와 roc\_auc\_score() 함수는 내부적으로 매개변수를 사용한다.) | |
| 예외 처리 | |
| 입력 데이터셋(x\_val, y\_val)의 형식이 올바르지 않은 경우 TypeError가 발생할 수 있다. | |
| 입력 데이터셋에 결측값이나 이상값이 포함된 경우 ValueError가 발생할 수 있다. | |
| 알고리즘 | |
| IsolationForest 모델은 학습 단계에서 학습한 정상적인 패턴을 기반으로 새로운 데이터 포인트의 이상치 점수를 계산한다. 이상치 점수가 특정 임계값보다 낮으면 해당 데이터 포인트를 이상치로 예측한다. predict() 메서드는 이러한 과정을 통해 예측 결과를 생성한다. | |
| 추가 설명 | |
| 모델 검증은 모델의 성능을 객관적으로 평가하고, 과적합을 방지하는 데 중요하다. | |
| 다양한 성능 평가 지표를 사용하여 모델의 성능을 다각적으로 분석할 수 있다. | |
| ROC AUC 점수는 이진 분류 모델의 성능을 평가하는 데 널리 사용되는 지표이며, 1에 가까울수록 모델의 성능이 우수함을 나타낸다. | |
| 모델 검증 결과를 바탕으로 모델의 하이퍼파라미터를 조정하여 성능을 개선할 수 있다. | |

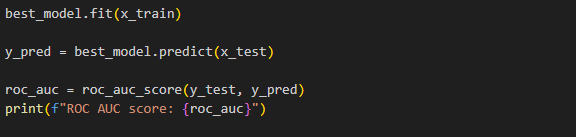
<2.2. 하이퍼파라미터 튜닝>



| 기능 | |
| --- | --- |
| 검증 결과를 바탕으로 모델의 하이퍼파라미터를 조정하여 최적의 성능을 찾는다. 이 단계에서는 Grid Search와 같은 방법을 사용하여 다양한 하이퍼파라미터 조합을 시도하고, 가장 좋은 성능을 보이는 조합을 선택한다. | |
| 코드 설명 | |
| best\_params = {}, best\_roc\_auc = 0 | 최적의 하이퍼파라미터와 ROC AUC 점수를 저장할 변수를 초기화한다. |
| param\_grid = {...} | 탐색할 하이퍼파라미터 범위를 지정한다.  contamination, n\_estimators, max\_samples는 IsolationForest 모델의 주요 하이퍼파라미터이다. |
| model = IsolationForest(...) | 지정된 하이퍼파라미터 값으로 IsolationForest 모델을 초기화한다. |
| y\_pred = model.fit\_predict(x\_train) | 학습 데이터(x\_train)를 사용하여 모델을 학습하고 예측을 수행한다. |
| roc\_auc = roc\_auc\_score(y\_train, y\_pred) | ROC AUC 점수를 계산한다. |
| if roc\_auc > best\_roc\_auc: … | 현재 하이퍼파라미터 조합으로 얻은 ROC AUC 점수가 이전보다 높으면 최적의 하이퍼파라미터와 ROC AUC 점수를 업데이트 한다. |
| best\_model = IsolationForest(...) | 최적의 하이퍼파라미터를 사용하여 최종 모델을 생성한다. |
| 입력 | |
| x\_train | 학습 데이터셋 (pandas DataFrame) |
| y\_train | 학습 데이터셋의 실제 값 (pandas Series 또는 numpy array) - roc\_auc\_score 계산에 사용된다. |
| 출력 | |
| best\_params | 최적의 하이퍼파라미터 (dictionary) - 콘솔에 출력된다. |
| best\_roc\_auc | 최적의 ROC AUC 점수 (float) - 콘솔에 출력된다. |
| best\_model | 최적의 하이퍼파라미터로 학습된 IsolationForest 모델 객체 |
| 매개변수 | |
| param\_grid | 탐색할 하이퍼파라미터 범위 (dictionary) |
| 예외 처리 | |
| 입력 데이터셋(x\_train, y\_train)의 형식이 올바르지 않은 경우 TypeError가 발생할 수 있다. | |
| 입력 데이터셋에 결측값이나 이상값이 포함된 경우 ValueError가 발생할 수 있다. | |
| param\_grid에 지정된 하이퍼파라미터 값이 IsolationForest 모델에서 허용되지 않는 값인 경우 ValueError가 발생할 수 있다. | |
| 알고리즘 | |
| Grid Search는 지정된 하이퍼파라미터 범위 내에서 모든 가능한 조합을 시도하여 최적의 하이퍼파라미터를 찾는 방법이다. 각 하이퍼파라미터 조합에 대해 모델을 학습하고 성능을 평가하여, 가장 좋은 성능을 보이는 조합을 선택한다. 본 코드에서는 Grid Search 대신 반복문을 사용하여 하이퍼파라미터 튜닝을 수행한다. | |
| 추가 설명 | |
| 하이퍼파라미터 튜닝은 모델의 성능을 최적화하는 데 중요한 단계이다. | |
| Grid Search는 하이퍼파라미터 튜닝을 위한 효과적인 방법이지만, 계산 비용이 많이 들 수 있다. | |
| 하이퍼파라미터 튜닝을 수행할 때는 교차 검증(cross-validation)을 사용하여 모델의 일반화 성능을 평가하는 것이 좋다. | |
| 본 코드에서는 Grid Search 대신 반복문을 사용하여 하이퍼파라미터 튜닝을 수행하지만, 실제로는 Grid Search와 같은 더 효율적인 방법을 사용하는 것이 좋다. | |

# <3. 모델 테스트>

<3.1. 예측 수행 및 최종 성능 평가>

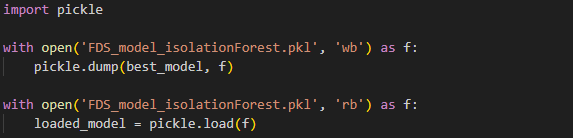


| 기능 | |
| --- | --- |
| 검증을 통해 최적화된 모델(best\_model)을 사용하여 x\_test 데이터에 대한 예측을 수행하고, 예측 결과(y\_pred)와 실제 값(y\_test)을 비교하여 모델의 최종 성능을 평가한다. 이 단계는 실제 환경에서 모델이 얼마나 잘 작동할지를 예측하는 데 중요한 과정이다. | |
| 코드 설명 | |
| best\_model.fit(x\_train) | 최적의 하이퍼파라미터로 찾은 모델(best\_model)을 학습 데이터셋(x\_train)을 사용하여 다시 학습시킨다. |
| y\_pred = best\_model.predict(x\_test) | 학습된 best\_model을 사용하여 테스트 데이터셋(xt\_test)에 대한 예측을 수행한다. 예측 결과는 y\_pred 변수에 저장된다. |
| roc\_auc = roc\_auc\_score(y\_test, y\_pred) | 실제 값(y\_test)과 예측 값(y\_pred)을 사용하여 ROC AUC 점수를 계산하고 roc\_auc 변수에 저장한다. |
| 입력 | |
| x\_train | 학습 데이터셋(pandas DataFrame) - best\_model을 학습하는데 사용된다. |
| x\_test | 테스트 데이터셋(pandas DataFrame) - 모델의 최종 성능을 평가하는 데 사용된다. |
| y\_test | 테스트 데이터셋의 실제 값(pandas Series 또는 numpy array) - ROC AUC 점수 계산에 사용된다. |
| 출력 | |
| y\_pred | 테스트 데이터셋에 대한 예측 결과 (numpy array) |
| ROC AUC score | 모델의 최종 성능 지표 (float) - 콘솔에 출력된다. |
| 매개변수 | |
| 없음 (코드에서 직접적으로 매개변수를 사용하지 않는다.) | |
| 예외 처리 | |
| 입력 데이터셋 (x\_train, x\_test, y\_test)의 형식이 올바르지 않은 경우 TypeError가 발생할 수 있다. | |
| 입력 데이터셋에 결측값이나 이상값이 포함된 경우 ValueError가 발생할 수 있다. | |
| 알고리즘 | |
| Isolation Forest 모델은 학습 단계에서 학습한 정상적인 패턴을 기반으로 새로운 데이터 포인트의 이상치 점수를 계산한다. 이상치 점수가 특정 임계값보다 낮으면 해당 데이터 포인트를 이상치로 예측한다. predict() 메서드는 이러한 과정을 통해 예측 겨로가를 생성한다. | |
| 추가 설명 | |
| 모델 테스트는 학습된 모델의 최종 성능을 평가하는 단계인다. | |
| 테스트 데이터셋은 모델 학습에 사용되지 않은 데이터여야 한다. | |
| ROC AUC 점수는 모델의 성능을 평가하는 데 널리 사용되는 지표이며, 1에 가까울수록 모델의 성능이 우수함을 나타낸다. | |
| 모델 테스트 결과를 바탕으로 모델의 배포 여부를 결정할 수 있다. | |

# 

# <4. 모델 배포>

<4.1. 모델 저장 및 로드>



| 기능 | |
| --- | --- |
| 학습된 IsolationForest 모델을 파일로 저장하고, 필요할 때 다시 불러와 사용할 수 있도록 한다. 이를 통해 모델을 반복적으로 학습시키지 않고 재사용할 수 있다. pickle을 Python 객체를 직렬화하여 파일에 저장하고, 다시 불러올 수 있도록하는 모듈이다. | |
| 코드 설명 | |
| import pickle | pickle 모듈을 가져온다. |
| with open(‘FDS\_model\_isolationForest.pkl’, ‘wb’) as f: | ‘FDS\_model\_isolationForest.pkl’파일을 쓰기 바이너리 모드 (‘wb’)로 열고, 파일 객체를 f로 지정한다. with 문을 사용하면 파일을 자동으로 닫아준다. |
| pickle.dump(best\_model, f): | pickle.dump() 함수를 사용하여 학습된 모델(best\_model)을 파일 객체(f)에 저장한다. |
| with open(‘FDS\_model\_isolationForest.pkl’, ‘rb’) as f: | ‘FDS\_model\_isolationForest.pkl’파일을 읽기 바이너리 모드(‘rb’)로 열고, 파일 객체를 f로 지정한다. |
| loaded\_model = pickle.load(f) | pickle.load() 함수를 사용하여 파일 객체(f)에서 모델을 불러와 loaded\_model 변수에 저장한다. |
| 입력 | |
| best\_model | 저장할 IsolationForest 모델 객체 (모델 저장 시) |
| FDS\_model\_isolationForest.pkl | 모델이 저장된 파일 경로 (모델 로드시) |
| 출력 | |
| FDS\_model\_isolationForest.pkl | 모델이 저장된 파일 (모델 저장 시) |
| loaded\_model | 불러온 IsolationForest 모델 객체 (모델 로스 시) |
| 매개변수 | |
| 없음 (코드에서 직접적으로 매개변수를 사용하지 않는다.) | |
| 예외 처리 | |
| 파일을 열거나 쓸 때 발생할 수 있는 IOError, OSError 등의 예외를 처리해야 한다. | |
| pickle 파일이 손상되었거나 올바르지 않은 경우 pickle.UnpicklingError 예외가 발생할 수 있다. | |
| 알고리즘 | |
| pickle 모듈은 Python 객체를 직렬화하여 바이트 스트림으로 변환하고, 이를 파일에 저장한다. 반대로, 파일에서 바이트 스트림을 읽어와 다시 Python 객체로 변환할 수 있다. | |
| 추가 설명 | |
| pickle을 사용하여 모델을 저장하면 모델을 다시 학습시키지 않고 재사용할 수 있어서 시간을 절약할 수 있다. | |
| pickle 파일은Python에서만 사용할 수 있으므로, 다른 언어와의 호환성이 필요한 경우 다른 형식 (예: PMML)을 사용해야 한다. | |
| 보안: 신뢰할 수 없는 출처에서 받은 pickle 파일을 로드하는 것은 보안 위험이 있으므로 주의해야 한다. 악의적인 코드가 포함된 pickle 파일을 로드하면 시스템이 손상될 수 있다. | |