모델 선택 소개

안녕하세요. 오늘 제가 여러분에게 소개할 주제는 '모델 선택(Model Selection)'입니다.

모델 선택은 머신러닝에서 중요한 단계로, 우리가 가진 데이터로부터 최적의 학습 결과를 얻기 위해 적절한 모델을 선택하는 과정을 말합니다. 이 과정에는 데이터를 학습 세트와 테스트 세트로 분리하고, 적절한 하이퍼파라미터를 설정하는 등의 단계가 포함됩니다. 오늘 발표에서는 이러한 모델 선택 과정의 각 단계를 자세히 살펴보고, 왜 이 과정이 중요한지에 대해 알아보겠습니다.

데이터 분할(train_test_split)

모델을 학습하기 전에, 우리는 가지고 있는 데이터를 학습용과 테스트용으로 분리해야 합니다. 이를 위해 사용되는 함수가 바로 'train_test_split()'입니다. 데이터를 분리하는 목적은 학습된 모델의 성능을 객관적으로 평가하기 위함입니다. 이 때, 데이터를 어떻게 분리하느냐가 모델의 성능을 크게 좌우할 수 있으므로, 적절한 비율로 학습과 테스트 세트를 나누는 것이 중요합니다.

test_size : 전체 데이터 세트에서 테스트 데이터 세트 크기를 얼마로 샘플링할 것인지 결정하는 파라미터입니다. 디폴트 값은 0.25 입니다.

random_state : 동일한 결과가 나오게 해주기 위한 파라미터입니다.

shuffle : 데이터를 분리하기 전에 데이터를 미리 섞을지 결정하는 파라미터입니다. 디폴트값은 True 이며 데이터를 분산시켜 보다 효율적인 학습/테스트 데이터 세트를 만드는데 사용합니다.

교차 검증(Cross Validation)

이와 같은 고정된 학습용 데이터와 테스트 데이터를 나누어 모델을 평가할 때는 과적합(Overfitting)에 취약합니다. 이를 보완하기 위해 교차 검증이라는 기법이 사용됩니다. 교차 검증은 데이터를 여러 개의 세트로 나누어, 각 세트를 검증 데이터로 사용하는 동안 나머지는 학습 데이터로 사용하여 모델을 평가하는 방법입니다. 이 방식을 통해 모델의 성능을 더 정확하게 평가할 수 있습니다. 쉽게 생각하면 수능을 보기위해 모의고사를 여러번 보는 것이라고 생각할 수 있습니다.

KFold 교차 검증

교차 검증의 한 방법으로, KFold 방법에 대해 알아보겠습니다. 전체 데이터를 K 개의 부분 집합으로 나누고, 이 중 하나의 폴드를 테스트 세트로 사용하고 나머지를 학습 세트로 사용하여 모델을 평가하는 과정을 K 번 반복하여, 각 폴드에 대한 모델 성능을 평가한 뒤, 이들의 평균값을 최종 모델 성능으로 사용합니다.

KFold 의 문제점

그러나 화면과 같은 불균형한 분포도를 가진 레이블 데이터에서는 K 폴드 방식이 전체 레이블 값의 분포를 반영하지 못하는 문제가 생깁니다. 이러한 문제를 해결한 방식이 필요한데 이 방식은 무엇일까요?

Stratified KFold

정답은 Stratified KFold 입니다. 앞서 말한 문제를 해결하기 위해 Stratified KFold 방식을 사용할 수 있습니다. 이 방법은 각 폴드에서 데이터의 레이블 분포가 전체 데이터셋의 분포와 유사하도록 유지하며, 이를 통해 보다 균형 잡힌 학습과 검증이 이루어질 수 있도록 합니다. 일반적으로 분류에서의 교차검증은 KFold 가 아니라 Stratified Kfold 로 분할이 되어야합니다. 반면, 회귀에서는 결정값이 이산형의 레이블이 아니라 연속된 숫자 값이기 때문에 Stratified Kfold 가 지원되지 않습니다.

---

cross_val_score

사이킷런은 교차 검증을 좀 더 편리하게 수행할 수 있게 해주는 API 를 제공합니다. 대표적으로 cross_val_score 입니다. 이는 폴드 세트 설정, 학습/테스트 데이터 추출, 학습/평가 반복을 일련의 과정을 한꺼번에 수행해주는 API 입니다. 이 API 는 내부적으로 stratified kFold 를 이용합니다.

Hyperparameter Optimization

다음으로는 하이퍼파라미터 튜닝을 소개하겠습니다. 하이퍼파라미터는 머신러닝 알고리즘을 구성하는 주요 구성 요소이며, 알고리즘 성능을 개선할 수 있습니다. 사이킷런에서는 대표적으로 GridSearchCV 와 Random Search 가 있습니다.

GridSearchCV

사이킷런은 그리드서치를 이용해 classifier 나 regressor 와 같은 알고리즘에 사용되는 하이퍼 파라미터를 순차적으로 입력하면서 최적의 파라미터를 도출할 수 있는 방안을 제공합니다.

GridSearchCV 클래스의 생성자로 들어가는 주요 파라미터는 화면과 같이 있습니다.

- **estimator** 는 classifier, regressor, pipeline 이 사용될 수 있습니다.

- **param_grid** 는 key 와 리스트 값을 가지는 딕셔너리가 주어집니다. estimator 의 튜닝을 위해 파라미터명과 사용될 여러 파라미터 값을 지정합니다.

- **scoring** 은 예측 성능을 측정할 평가 방법을 지정합니다.

- **cv :** 교차 검증을 위해 분할되는 학습/검증 세트의 개수를 지정합니다. K 값으로 생각 하시면 됩니다.

- **refit :** 디폴트가 True 이며 True 로 생성 시 가장 최적의 하이퍼 파라미터를 찾은 뒤 입력된 estimator 객체를 해당 하이퍼 파라미터로 재학습 시킵니다.

이번에는 퀴즈인데요, 결정트리에서 최대 깊이를 결정하는 파라미터와 노드를 분할하기 위한 최소 샘플 수를 결정하는 파라미터 총 두가지를 적어서 구글폼에 제출해주시면 됩니다.

정답은 max_depth 와 min_samples_split 입니다.

다음으로 바로 퀴즈가 있습니다. cv=3, 다음의 파라미터로 모델을 학습시킬 때, 총 학습 및 평가 횟수는 얼마나 될까요? 파라미터는 max_depth=[1,2,3], min_samples_split=[2,3]입니다.

정답은 파라미터의 개수를 곱한 3 x 2 에 폴드 수 3 을 곱한 18 입니다.

다음으로 랜덤서치에 대해 간략히 설명하겠습니다.

랜덤서치는 랜덤으로 숫자를 뽑아서 실험을 합니다. 그리드 서치에서 실험할 하이퍼 파라미터를 명시적으로 정해줘야 한다면, 랜덤 서치는 하이퍼파라미터로 시도할 숫자의 구간과 횟수를 정해줍니다. 랜덤서치는 특히, 파라미터의 수가 많고 데이터의 양이 방대할 때 빛을 발합니다.

이상으로 2 조 발표를 마치겠습니다. 감사합니다.