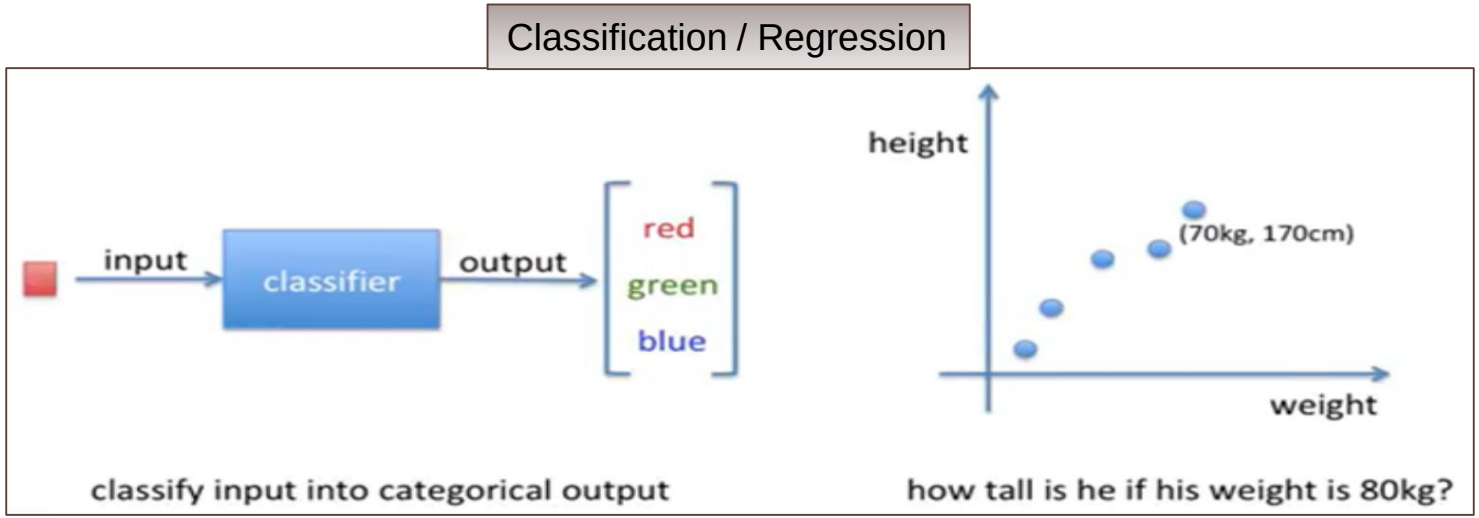
Deep Learning
week 4

목차

4.1.1 지도 학습

- 샘플 데이터가 주어지면 알고 있는 타깃에 입력 데이터를 매핑하는 방법 학습
- 크게 분류(classification)와 회귀(regression)이 있다.

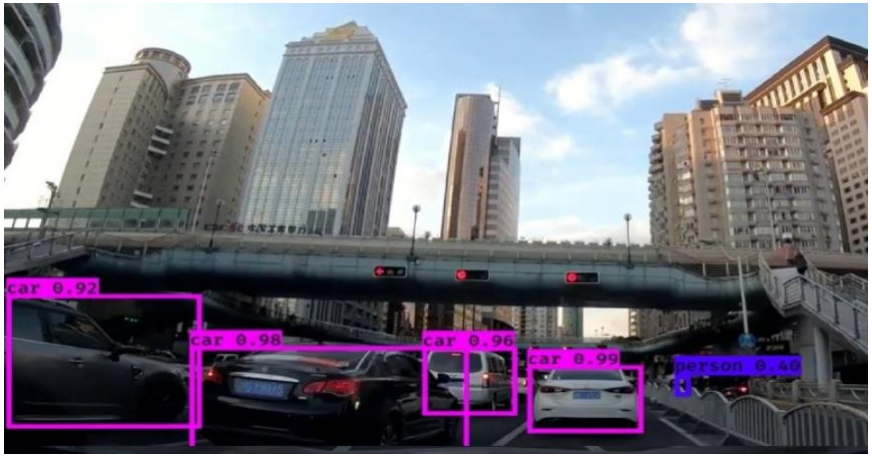


4.1.1 분류와 회귀에서 사용하는 용어

- **샘플** 또는 **입력**: 모델의 주입될 하나의 데이터 포인트
- **예측** 또는 **출력**: 모델로부터 나오는 값
- **타깃**: 정답, 외부데이터 소스에 근거하여 모델이 완벽하게 예측해야 하는 값
- **예측 오차** 또는 **손실 값**: 모델의 예측과 타깃 사이의 거리를 측정한 값
- **클래스**: 분류 문제에서 선택할 수 있는 가능한 레이블의 집합
- **레이블**: 분류 문제에서 클래스 할당의 구체적인 사례.
- **참 값**(ground-truth) 또는 **꼬리표**(Annotation): 데이터셋에 대한 모든 타깃. 일반적으로 사람에 의해 수집됨
- **이진 분류**: 각 입력 샘플이 2개의 배타적인 범주로 구분되는 분류 작업
- **다중 분류**: 각 입력 샘플이 2개 이상의 범주로 구분되는 분류 작업
- **다중 레이블 분류**: 각 입력 샘플이 여러 개의 레이블에 할당될 수 있는 분류 작업.
- **스칼라 회귀**: 타깃이 연속적인 스칼라 값인 작업.
- **벡터 회귀**: 타깃이 연속적인 값의 집학인 작업
- **미니 배치** 또는 **배치**: 모델에 의해 동시에 처리되는 소량의 샘플 묶음(일반적으로 8개 ~ 128개 사이), 훈련할 때 미니 배치마다 한 번씩 모델의 가중치에 적용할 경사 하강법 업데이트 값을 계산

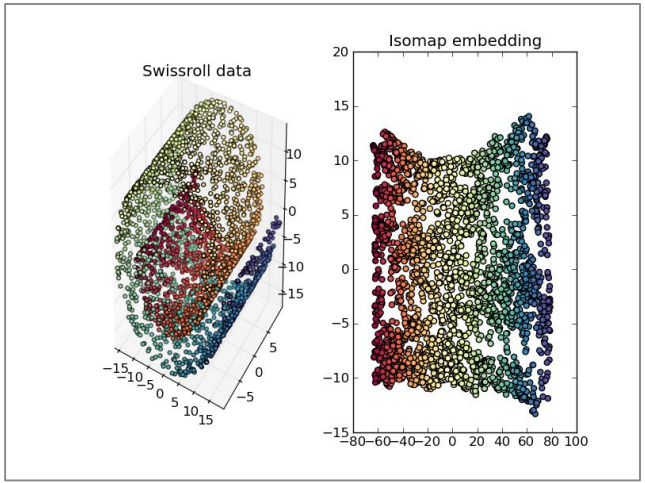
4.1.1 지도 학습

1. 시퀀스 생성 (sequence generation)
2. 구문 트리 예측(syntax tree)
3. 물체 감지(object detection)
4. 이미지 분할(image segmentation)

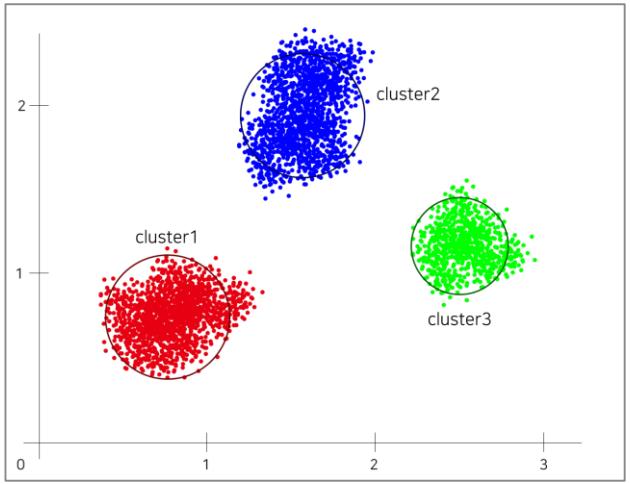


4.1.2 비지도 학습

- 타깃을 사용하지 않고 입력 데이터에 대해 규칙성을 찾는 학습 방법
- 데이터 시각화, 데이터 압축, 데이터 노이즈 제거 또는 데이터의 상관관계를 이해하기 위해 사용
- 차원 축소(dimensionality reduction)과 군집(clustering)이 잘 알려져 있음



차원 축소

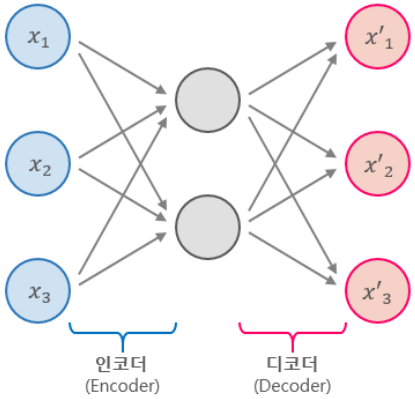


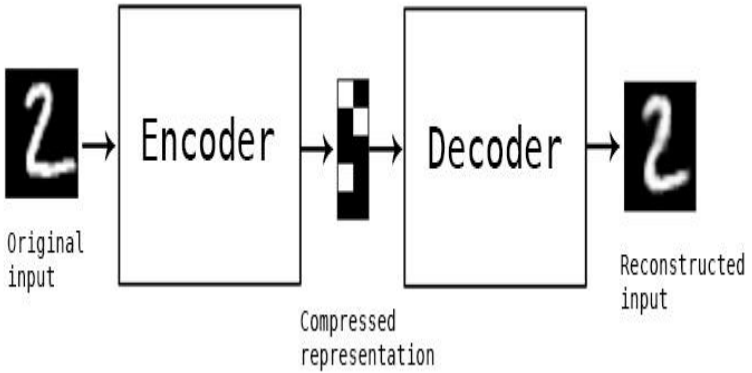
군집

4.1.3 자기 지도 학습

- 지도 학습의 특별한 경우.
- 사람이 만든 레이블을 사용하지 않음. 주로 경험적인 알고리즘 (heuristic algorithm)을 사용.

Ex) 오토인코더(autoencoder)

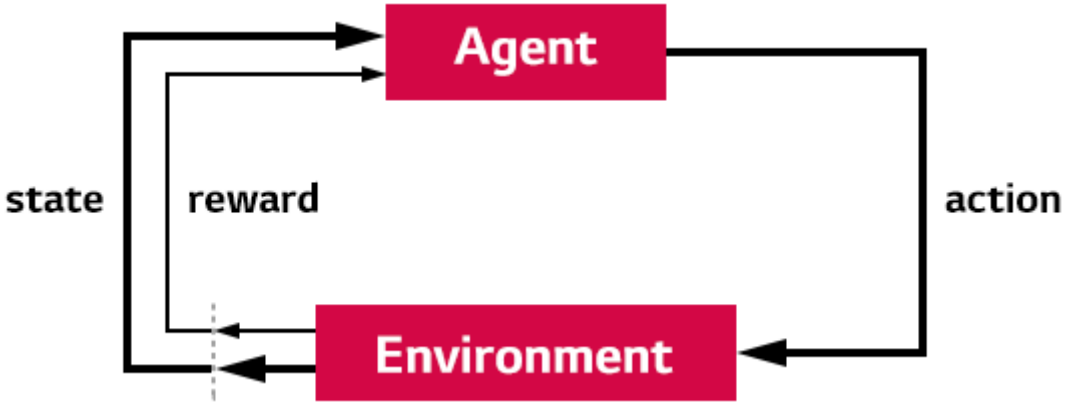




$n

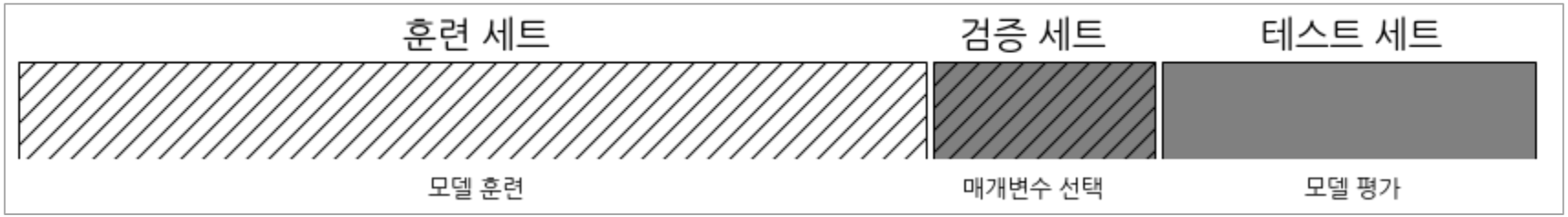
4.1.4 강화 학습

- 주어진 환경에 대한 정보를 받아 보상을 최대화하는 행동을 선택하도록 학습 방법
- 대부분 연구 영역에 속해 있으며 자율 주행 자동차, 자원 관리, 교육 등에 확대될 것임.



4.2.1 훈련, 검증, 테스트 세트

- 모델 평가의 핵심은 가용한 데이터를 훈련, 검증, 테스트 3개의 세트로 준비하는 것.
- 훈련 세트에서 모델을 훈련하고 검증 세트에서 모델을 평가, 테스트 세트에서 모델을 테스트



4.2.1 훈련, 검증, 테스트 세트

- 검증 세트가 필요한 이유?

  ➡ 모델을 개발할 때 항상 모델의 설정을 튜닝하기 때문 (ex : 층의 수 or 층의 유닛 수)

  하이퍼 파라미터

- 검증 세트도 오버피팅이 될 수 있음

  ➡ 정보 누설 (information leak)

  검증 성능에 기반하여 모델의 하이퍼파라미터를 조정할 때마다 검증 데이터에 관한 정보가 모델로 새는 것

4.2.1 단순 홀드아웃 검증

- 전체 데이터셋에서 일정량을 테스트셋으로 설정
- 남은 데이터에서 훈련하고 테스트 세트로 평가

➢ 데이터가 적은 경우, 검증 세트와 테스트 세트의 샘플이 너무 적어 주어진 전체 데이터를 통계적으로 대표하지 못할 수 있음.

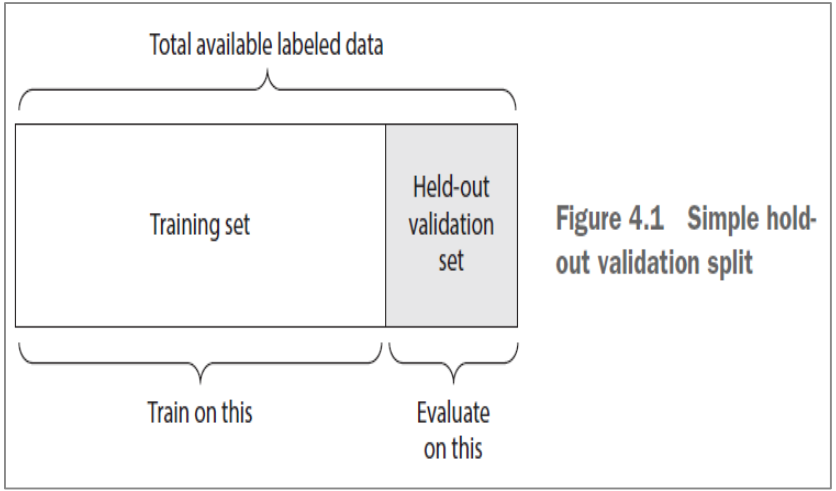


Figure 4.1 Simple hold-out validation split

4.2.1 단순 홀드아웃 검증

```
num_validation_samples = 10000

np.random.shuffle(data)                          --------> 데이터 셔플링

validation_data = data[:num_validation_samples]  --------> 검증 세트 만들기
data = data[num_validation_samples:]

training_data = data[:]                          --------> 훈련 세트 만들기

model = get_model()                              --------> 훈련세트에서 모델 훈련
model.train(training_data)                                 검증 세트로 평가
validation_score = model.evaluate(validation_data)

모델 튜닝,
다시 훈련, 평가, 튜닝 반복 …

model = get_model()                              --------> 하이퍼파라미터 튜닝이 끝나면
model.traing(np.concatenate([training_data,                테스트 데이터를 제외한
                             validation_data]))            모든 데이터를 사용해
test_score = model.evalutate(test_data)                    모델을 다시 훈련시킴.
```

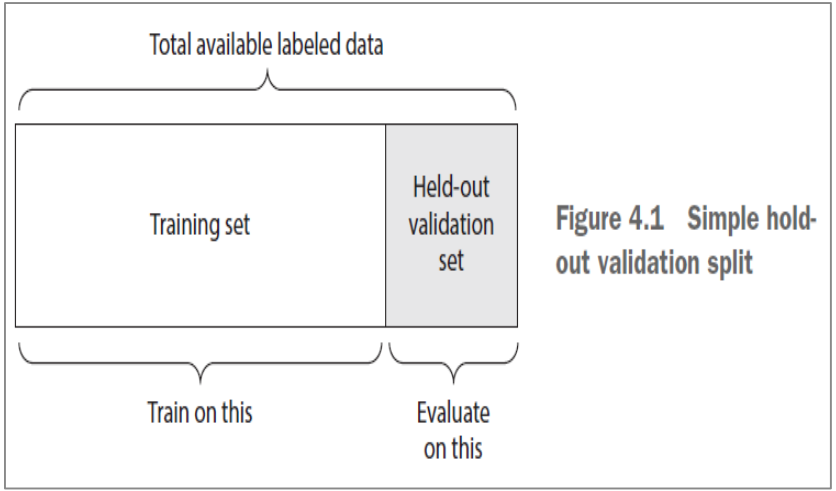
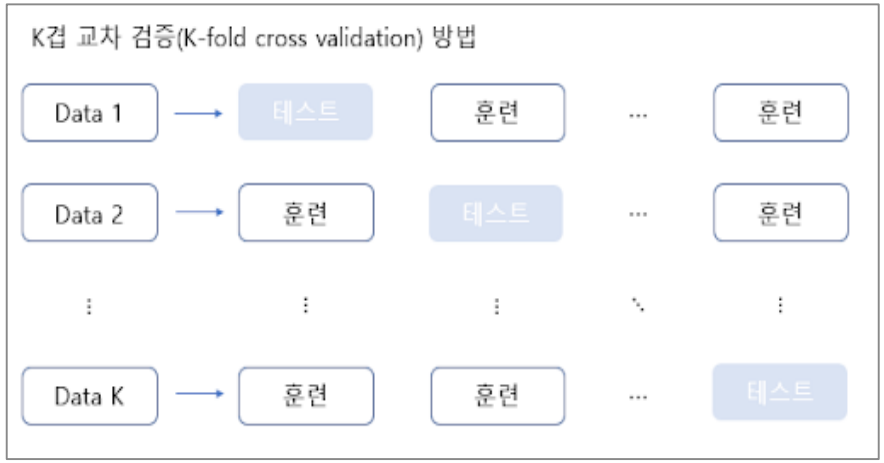


Figure 4.1 Simple hold-out validation split

4.2.1 K-겹 교차 검증

1) 데이터를 동일한 K개 분할로 나눔
2) 각 분할 i에 대해 남은 K-1개의 분할로 모델을 훈련
3) 분할 i에서 모델을 평가
4) K개 점수 평균으로 최종 점수

➢ 데이터 분할에 따라 편차가 클 때 도움이 됨.



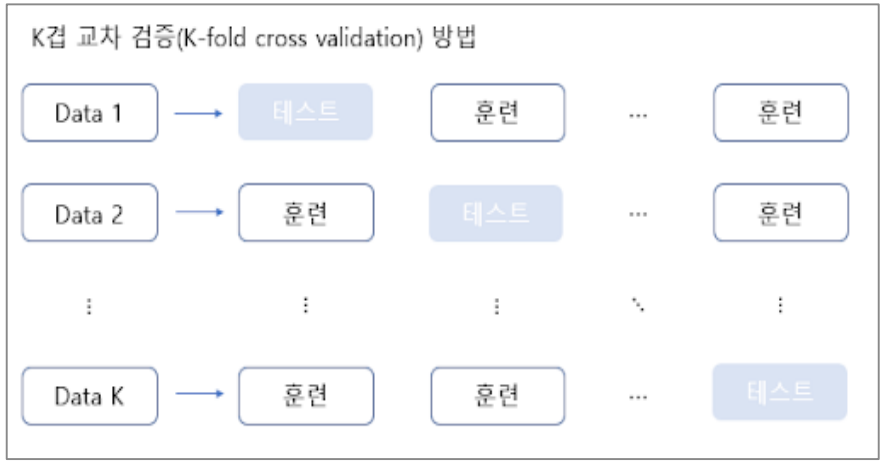
4.2.1 K-겹 교차 검증

```
k = 4
num_validation_samples = len(data) // k

np.random.shuffle(data)
validation_scores = []

for fold in range(k):
    validation_data = data[num_validation_samples * fold:
    num_validation_samples * (fold + 1)]
    training_data = data[:num_validation_samples * fold] +
    data[num_validation_samples * (fold + 1):]

    model = get_model()
    model.train(training_data)
    validation_score = model.evaluate(validation_data)
    validation_scores.append(validation_score)

validation_score = np.average(validation_scores)

model = get_model()
model.train(data)
test_score = model.evaluate(test_data)
```

검증 데이터 부분 선택

남은 데이터를 훈련 데이터로 사용

검증 점수
: K개 폴드의 검증 점수 평균



4.2.1 셔플링을 사용한 반복 K-겹 교차 검증

- 비교적 가용 데이터가 적고 가능한 정확하게 모델을 평가하고자 할 때 사용
- K-겹 교차 검증을 여러 번 적용하되, K개의 분할로 나누기 전에 매번 데이터를 무작위로 섞음
- 최종 점수는 모든 K-겹 교차 검증(P * K) 점수의 평균

반복 횟수

➢ 모든 모델을 훈련하고 평가하므로 비용이 많이 든다.

4.2.2 유념 사항

- ✓ **대표성 있는 데이터** : 훈련 세트와 테스트 세트가 주어진 데이터에 대한 대표성이 있어야 함.
- ✓ **시간의 방향** : 과거로부터 미래를 예측할 때는 훈련 세트는 테스트 세트보다 과거 데이터여야 함.
- ✓ **데이터 중복** : 훈련 세트와 검증 세트가 중복되지 않도록 함.

4.3.1 신경망을 위한 데이터 전처리

- 목적: 주어진 원본 데이터를 신경망에 적용하기 쉽도록 하는 것
- 벡터화(vectorization), 정규화(normalization), 누락된 값 다루기, 특성 추출 등이 포함

4.3.1 신경망을 위한 데이터 전처리

벡터화

- 신경망에서 모든 입력과 타깃은 부동 소수 데이터로 이루어진 텐서여야 함 (특정 경우 정수 텐서)
- 데이터를 신경망에 넣기 위해서 텐서로 변환하는 것.

값 정규화

- 작은 값. 일반적으로 0~1 사이
- 균일. 모든 특성이 대체로 비슷한 범위를 가져야 함.

누락된 값 다루기

- 데이터셋에서 누락된 값을 다룰수 있어야 함
- 훈련 데이터세트에 누락된 값을 평균(mean), 중간 값(median)으로 대체할 경우 그 값을 저장해야 함
- 훈련 데이터세트, 테스트 데이터세트에 둘다 누락된 값을 동일하게 넣어주어야 함

4.3.2 특성 공학

- 모델에 데이터를 주입하기 전에 하드코딩된 변환을 적용하여 알고리즘이 더 잘 수행되도록 만들어 줌.

2차원 픽셀 데이터

시침, 분침의 직교좌표

시침 분침의 극좌표

Raw data: pixel grid		
Better features: clock hands' coordinates	{x1: 0.7, y1: 0.7} {x2: 0.5, y2: 0.0}	{x1: 0.0, y2: 1.0} {x2: -0.38, 2: 0.32}
Even better features: angles of clock hands	theta1: 45 theta2: 0	theta1: 90 theta2: 140

Figure 4.3 Feature engineering for reading the time on a clock

4.5 보편적인 머신 러닝 작업 흐름

1. 주어진 문제 정의

- 입력 데이터, 예측하는 정보 : 가용 훈련 데이터의 유무
- 문제의 유형 : 이진 분류, 다중 분류, 스칼라 회귀 등

2. 성공 지표 선택

- 클래스 분포가 균일한 분류 문제에서는 정확도와 ROC AUC가 일반적인 지표
- 클래스 분포가 균일하지 않은 문제에서는 정밀도와 재현율을 사용할 수 있음
- 랭킹 문제, 다중 레이블 문제에는 평균 정밀도를 사용

4.5 보편적인 머신 러닝 작업 흐름

3. 평가 방법 선택

- 홀드아웃 검증 세트 분리: 데이터가 풍부할 때 사용
- K-겹 교차 검증: 홀드아웃 검증을 사용하기에 샘플의 수가 너무 적을 때 사용
- 반복 K-겹 교차 검증: 데이터가 적고 매우 정확한 모델 평가가 필요할 때 사용

4. 데이터 준비

- 데이터는 텐서로 구성됨
- 텐서에 있는 값은 일반적으로 작은 값으로 스케일 조정되어 있음 (ex. [-1,1] or [0,1] 범위)
- 특성마다 범위가 다르면(여러 종류의 값으로 이루어진 데이터라면) 정규화해야 함.

4.5 보편적인 머신 러닝 작업 흐름

5. 기본모다 나은 모델 훈련하기

-> 통계적 검정력을 달성하는 것.

- 마지막 층의 활성화 함수 : 네트워크의 출력에 필요한 제한을 가함.
- 손실 함수: 풀려고 하는 문제의 종류에 적합해야 함.
- 최적화 설정 : 옵티마이저와 학습률을 확인해야 함.

->주로 rmsprop과 기본 학습률 사용

모델에 맞는 마지막 층의 활성화 함수와 손실 함수 선택

문제 유형	마지막 층의 활성화 함수	손실함수
이진 분류	시그모이드	binary_crossentropy
단일 레이블 다증 분류	소프트맥스	categorical_crossentropy
다중 레이블 다중 분류	시그모이드	binary_crossentropy
임의 값에 대한 회귀	없음	mse
0과 1사이 값에 대한 회귀	시그모이드	mse 또는 binary_crossentropy

4.5 보편적인 머신 러닝 작업 흐름

6. 몸집 키우기 : 과대적합 모델 구축

- 층 추가
- 층 크기 키우기
- 더 많은 epoch 동안 훈련

7. 모델 규제와 하이퍼파라미터 튜닝

- 드롭아웃 추가
- 층 추가 또는 제거
- L1이나 L2 또는 두 가지 모두 추가
- 하이퍼파라미터 바꾸기
- 특성 공학 시도

Q & A