

Fundamentals of machine learning

김연우

2019-01-08

4.1. Four branches of machine learning

- 해결할 문제에 맞는 학습 알고리즘 선택
 - Supervised learning
 - Unsupervised learning
 - Self-supervised learning
 - Reinforcement learning

4.1. Four branches of machine learning

- Supervised learning
 - mapping input data to known targets(annotations)
 - classification, scalar regression
 - Sequence generation
 - Given a picture, predict a caption describing it.
 - *Syntax tree prediction*
 - *Given a sentence, predict its decomposition into a syntax tree.*
 - Object detection
 - Given a picture, draw a bounding box around certain objects inside the picture.
 - *Image segmentation*
 - Given a picture, draw a pixel-level mask on a specific object.

4.1. Four branches of machine learning

- Unsupervised learning
 - transformations of the input data without the help of any targets
 - data visualization, data compression, or data denoising
 - to better understand the correlations present in the data at hand
 - Dimensionality reduction and clustering

4.1. Four branches of machine learning

- Self-supervised learning
 - learning without human-annotated labels
 - autoencoders
 - to predict the next frame in a video, given past frames
 - to predict the next word in a text, given previous words
- Reinforcement learning
 - an *agent* receives information about its environment and learns to choose actions that will maximize some reward.
 - self-driving cars
 - robotics
 - resource management

4.2 Evaluating machine-learning models

- 머신 러닝의 목표
 - 신규 데이터에서 잘 작동하는 일반화된 모델을 얻는 것
 - how to measure generalization
 - how to evaluate machine-learning models.

4.2.1 Training, validation, and test sets

- 데이터 분할에 관한 튜닝
 - Training / validation / test
 - one of the hyperparameters (others: choosing the number of layers or the size of the layers etc.)
 - 주의 사항
 - information leak: tuning the configuration of the model based on its performance on the validation set can quickly result in overfitting to the validation set
- 주요 방법
 - simple hold-out validation
 - K-fold validation
 - iterated K-fold validation with shuffling

4.2.1 Training, validation, and test sets

- SIMPLE HOLD-OUT VALIDATION

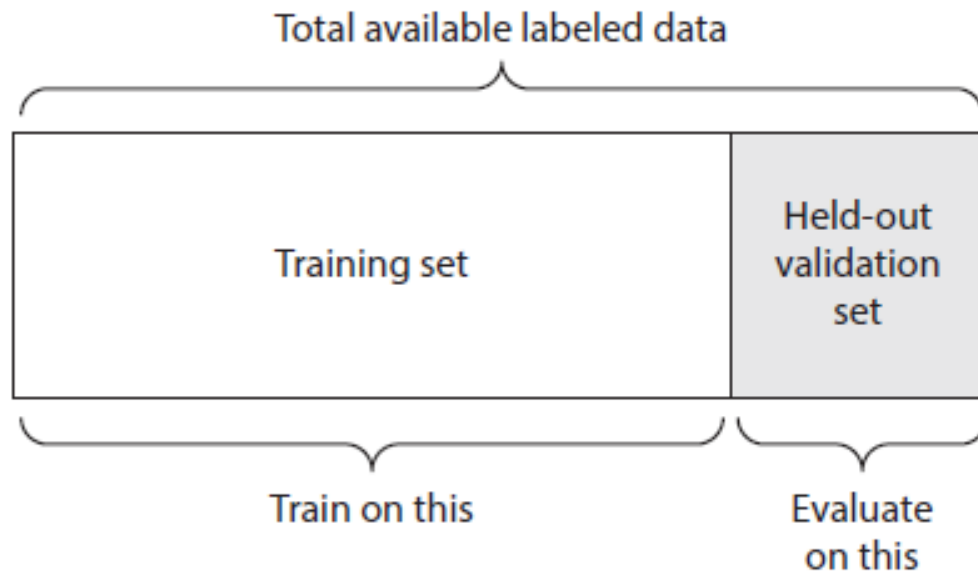


Figure 4.1 Simple hold-out validation split

4.2.1 Training, validation, and test sets

- K-FOLD VALIDATION

- 데이터가 적을 때 사용
- 최종 스코어: 각 K-fold 실행에서 얻은 스코어의 평균

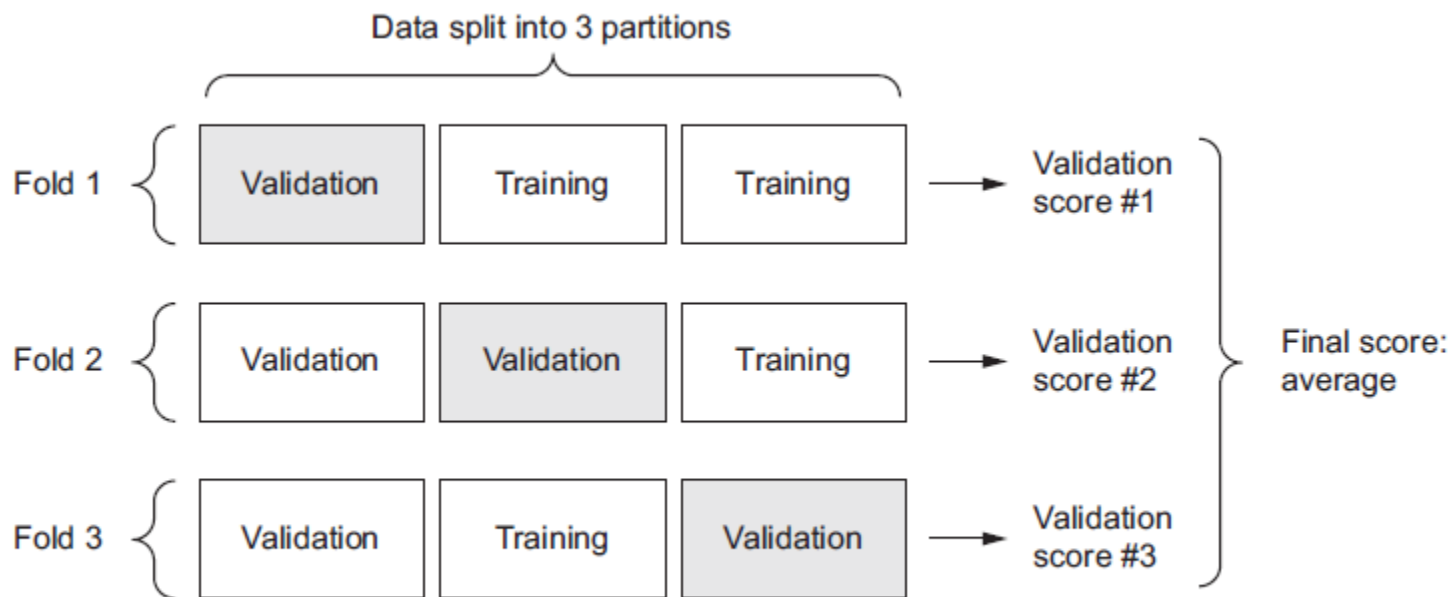


Figure 4.2 Three-fold validation

4.2.1 Training, validation, and test sets

- ITERATED K-FOLD VALIDATION WITH SHUFFLING
 - 데이터가 적을 때 사용
 - k 개로 분할 하기 전에 데이터를 셔플링
 - 최종 스코어: 각 k -fold 실행에서 얻은 스코어의 평균

4.2.2 Things to keep in mind

- Data representativeness
 - traning/test 데이터 모두 대표성이 있는 데이터 사용
- The arrow of time
 - 시간선이 있는 데이터는 셔플링하면 안됨
 - traning 데이터가 test 데이터보다 시간선이 앞서야 함
- Redundancy in your data
 - traning 데이터와 test 데이터에 중복된 데이터 포인트가 있으면 안됨

4.3.1 Data preprocessing for neural networks

- VECTORIZATION
 - 단어 리스트를 정수 인덱스 리스트로 변환
 - one-hot encoding
- VALUE NORMALIZATION
 - 각 데이터 포인트를 동일한 스케일로 인코딩
 - 스케일이 다른 자질은 각 자질을 독립적으로 정규화
 - Take small values—Typically, most values should be in the 0–1 range.
 - Be homogenous—That is, all features should take values in roughly the same range.
- HANDLING MISSING VALUES
 - 어떤 데이터 포인트에 누락된 값이 있다면 0으로 채움

4.3.2 Feature engineering

- 더 좋은 데이터 표현으로 변환



Raw data: pixel grid		
Better features: clock hands' coordinates	$\{x1: 0.7, y1: 0.7\}$ $\{x2: 0.5, y2: 0.0\}$	$\{x1: 0.0, y1: 1.0\}$ $\{x2: -0.38, y2: 0.32\}$
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

4.4 Overfitting and underfitting

- Optimization:
 - refers to the process of adjusting a model to get the best performance possible on the training data.
- generalization:
 - refers to how well the trained model performs on data it has never seen before.
 - 머신 러닝의 목표는 generalization
 - generalization을 위해서는 가능한 많은 훈련 데이터를 모으는 것

4.4 Overfitting and underfitting

- underfitting
 - 모델의 성능이 계속 발전될 여지가 있는 경우
- overfitting
 - Training 세트 혹은 Validation 세트에 지나치게 최적화되어 일반화를 잃은 경우

4.4 Overfitting and underfitting

- to prevent overfitting (training set에 대한)
 - Reducing the network's size
 - 레이어의 수를 줄이거나
 - 각 레이어의 유닛 수를 줄임
 - Adding weight regularization
 - weight이 복잡할 수록 패널티를 부여함 (loss score에 반영)
 - L1 regularization—The cost added is proportional to the absolute value of the weight coefficients (the L1 norm of the weights).
 - L2 regularization—The cost added is proportional to the square of the value of the weight coefficients (the L2 norm of the weights).
 - Adding dropout
 - 훈련 시 층의 일부 출력을 0으로 만드는 것
 - [0.2, 0.5, 1.3, 0.8, 1.1] -> [0, 0.5, 1.3, 0, 1.1]

4.5 The universal workflow of machine learning

1. Defining the problem and assembling a dataset
2. Choosing a measure of success
3. Deciding on an evaluation protocol
4. Preparing your data
5. Developing a model that does better than a baseline
6. Scaling up: developing a model that overfits
7. Regularizing your model and tuning your hyperparameters

4.5 The universal workflow of machine learning

1. Defining the problem and assembling a dataset

- 입력 데이터의 형식 및 특성 파악
- 해결할 문제의 종류 파악 classification? Scalar regression?
- 가설 설정
 - You hypothesize that your outputs can be predicted given your inputs.
 - You hypothesize that your available data is sufficiently informative to learn the relationship between inputs and outputs.

4.5 The universal workflow of machine learning

2. Choosing a measure of success

- loss function을 선택하는 기준
- accuracy? Precision and recall?
- area under the receiver operating characteristic curve (ROC AUC)

3. Deciding on an evaluation protocol

- Maintaining a hold-out validation set—The way to go when you have plenty of data
- Doing K-fold cross-validation—The right choice when you have too few samples for hold-out validation to be reliable
- Doing iterated K-fold validation—For performing highly accurate model evaluation when little data is available

4.5 The universal workflow of machine learning

4. Preparing your data

- As you saw previously, your data should be formatted as tensors.
- The values taken by these tensors should usually be scaled to small values: for example, in the $[-1, 1]$ range or $[0, 1]$ range.
- If different features take values in different ranges (heterogeneous data), then the data should be normalized.
- You may want to do some feature engineering, especially for small-data problems.

4.5 The universal workflow of machine learning

5. Developing a model that does better than a baseline
 - 적어도 랜덤한 분류보다 높은 정확도를 가지게 만드는 것
 - 마지막 레이어(출력 레이어)를 어떻게 구성할 것인가

Table 4.1 Choosing the right last-layer activation and loss function for your model

Problem type	Last-layer activation	Loss function
Binary classification	<code>sigmoid</code>	<code>binary_crossentropy</code>
Multiclass, single-label classification	<code>softmax</code>	<code>categorical_crossentropy</code>
Multiclass, multilabel classification	<code>sigmoid</code>	<code>binary_crossentropy</code>
Regression to arbitrary values	None	<code>mse</code>
Regression to values between 0 and 1	<code>sigmoid</code>	<code>mse</code> or <code>binary_crossentropy</code>

- Optimization configuration—In most cases, it's safe to go with rmsprop and its default learning rate.

4.5 The universal workflow of machine learning

6. Scaling up: developing a model that overfits

- 일단은 충분히 overfited 된 모델을 만들어 본다
 - Add layers.
 - Make the layers bigger.
 - Train for more epochs.

4.5 The universal workflow of machine learning

7. Regularizing your model and tuning your hyperparameters

- training set에 대한 overfitting 패턴을 보고 더 좋은 모델을 만드는 과정
- 이 단계가 대부분의 시간을 차지함
 - Add dropout
 - Try different architectures: add or remove layers.
 - Add L1 and/or L2 regularization.
 - Try different hyperparameters (such as the number of units per layer or the learning rate of the optimizer) to find the optimal configuration.
 - Optionally, iterate on feature engineering: add new features, or remove features that don't seem to be informative.
- 여기에 지나치게 집중하면 validation set에 대한 overfitting 발생 information leak