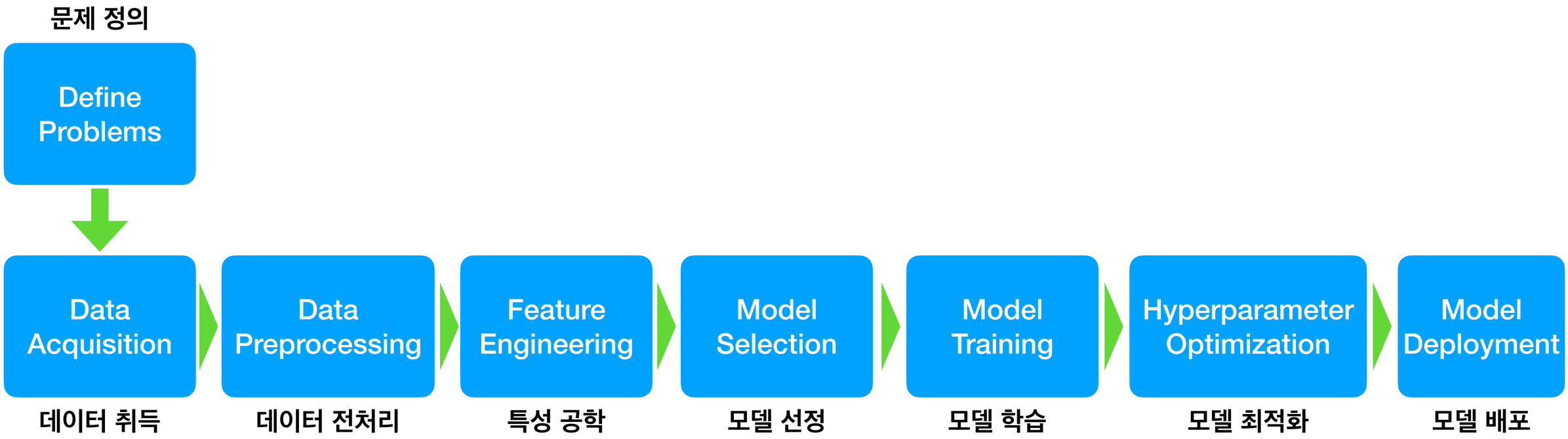
Chapter 8. Metric, CV, Optimization

머신 러닝 모델 개발 프로세스



성능 평가

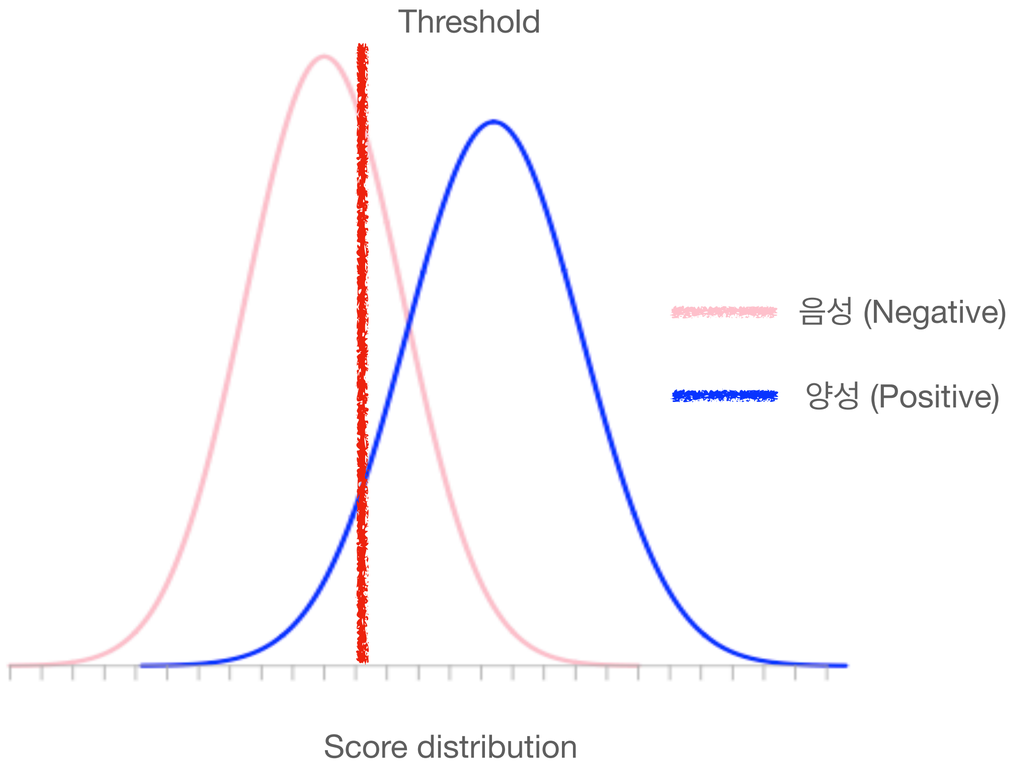
- Classification : Accuracy, F1 score, AUROC
- Regression : Mean absolute error, Mean squared error, R-squared

성능평가 - 분류

- 분류 결과 얼마나 잘 맞는지를 정량화하여 수치적으로 환산해야 함.
- 가장 단순한 binary classification(이진분류) 기준으로 다음을 계산할 수 있음.
- 0: Negative, False
- 1: Positive, True
- 이진분류가 잘 되는 경우 두가지, 이진분류가 잘못되는 경우에도 두가지 경우가 존재함.
- Type 1 error: 실제 False 클래스를 True 클래스로 오분류,
- Type 2 error : 실제 True 클래스를 False 클래스로 오분류,

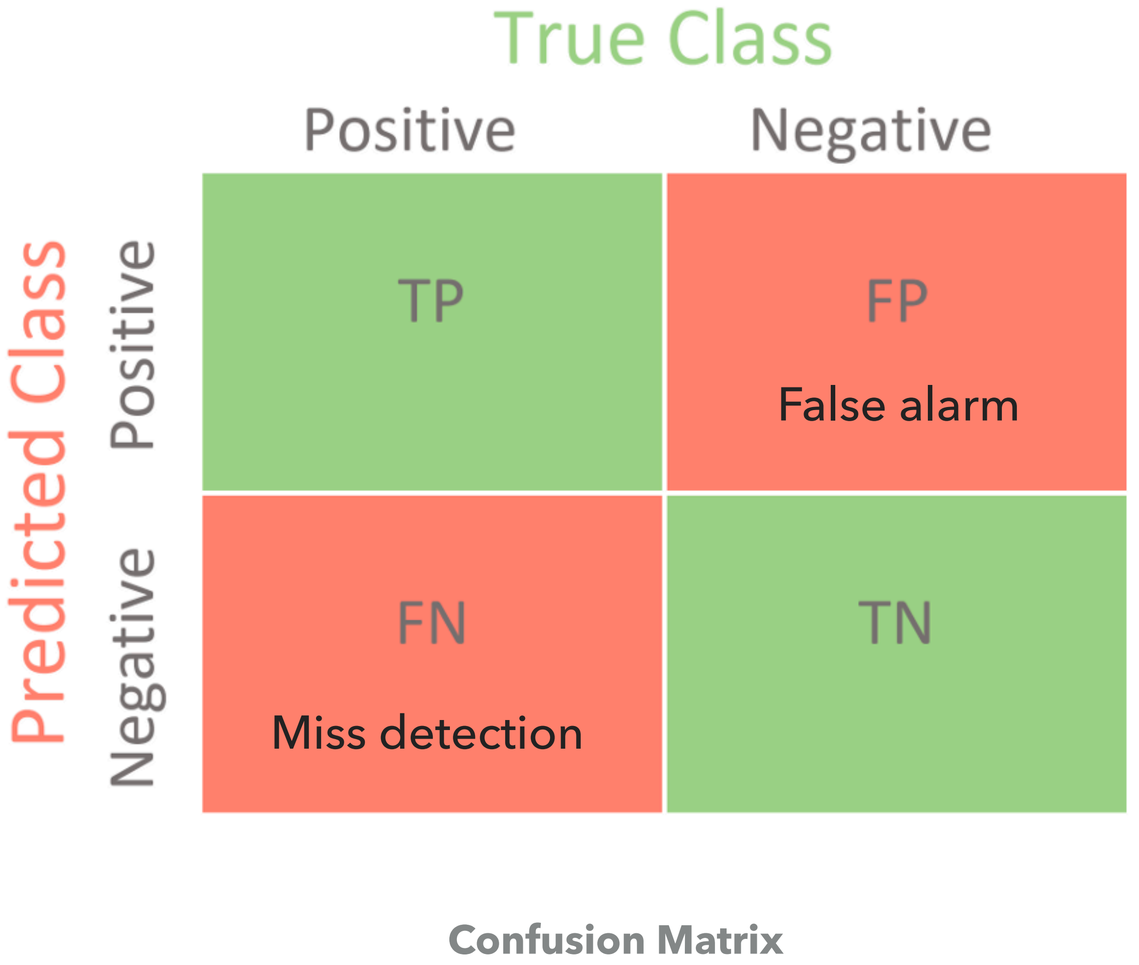
분류 문제 메트릭 - 기초 요소

- True Positive (TP) : (기준선 위의 파란 영역)
  - 실제 양성을 양성으로 분류.
- True Negative (TN): (기준선 밑의 붉은 영역)
  - 실제 음성을 음성으로 분류.
- False Positive (FP): (기준선 위의 붉은 영역)
  - 실제 음성을 양성으로 분류, False alarm.
- False Negative (FN): (기준선 밑의 파란 영역)
  - 실제 양성을 음성으로 분류, Miss detection.



분류 문제 메트릭 - Confusion matrix

- Recall, True Positive Rate (TPR) :
  - TP/(TP+FN), 양성 예측 / 실제 양성
  - 값이 클수록 좋음.
- Fall-out, False Positive Rate (FPR)
  - FP/(FP+TN), 양성 예측 / 실제 음성
  - 값이 작을수록 좋음
- Precision
  - TP/(TP+FP), 실제 양성 / 양성 예측
  - 값이 클수록 좋음
- Accuracy
  - (TP+TN)/(TP+TN+FP+FN), 맞게 예측 / 전체 데이터
  - 값이 클수록 좋음



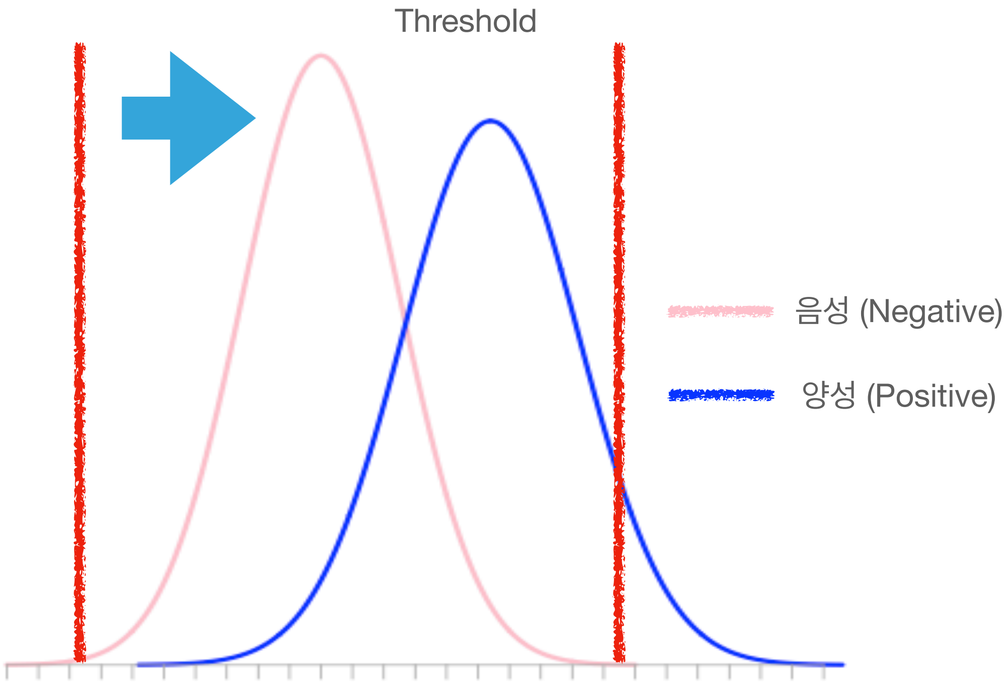
Confusion Matrix

분류 문제 메트릭 - F1 score

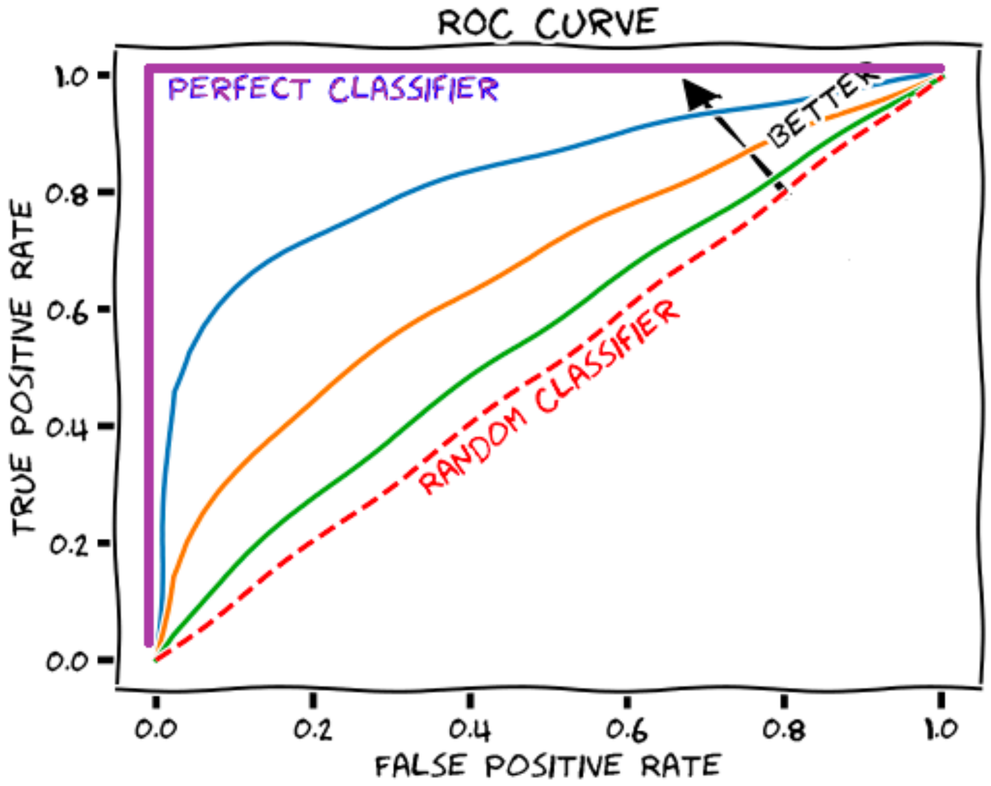
- F1-score
  - 2 * (Precision * Recall) / (Precision + Recall)
  - Precision과 recall의 조화평균

분류 문제 메트릭 - AUROC

- Perfect classifier : TPR(Recall)=1, FPR = 0, Precision=1
- 0.5 : Random, 1 = Perfect classifier



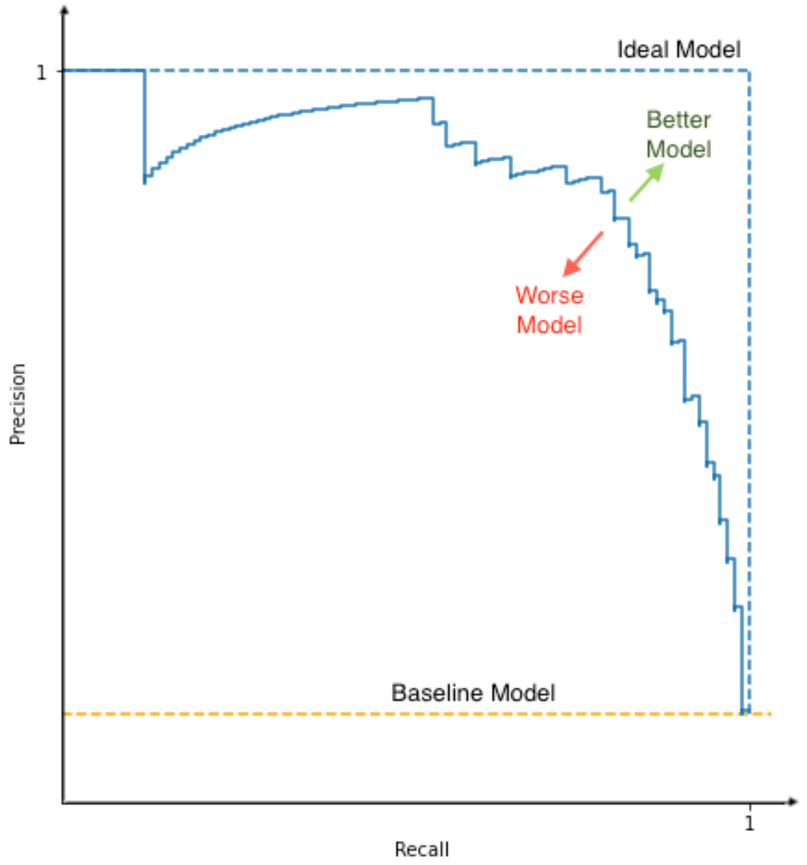
Score distribution



ROC curve and AUROC (AUC)

분류 문제 메트릭 - PRAUC

- Perfect classifier : TPR(Recall)=1, FPR = 0, Precision=1
- 1 = Perfect classifier
- 랜덤모델인 경우의 기준선 = precision ~ 전체 데이터 중 양성데이터 비율



다중분류의 경우

- 개, 고양이, 토끼 클래스를 구분하는 경우?

클래스 (12)	1(양성)	0(음성)
1(양성)	TRUE POSITIVE 3명	FALSE NEGATIVE 1명
0(음성)	FALSE POSITIVE 2명	TRUE NEGATIVE 6명

전체샘플 (27)	강아지	고양이	토끼
강아지	5	1	3
고양이	2	5	2
토끼	1	3	5

다중분류의 경우

- 기본적으로 각 클래스 별로 이진 분류를 수행한 것으로 가정한 confusion matrix를 통해 지표 계산
- 전체 지표에 대한 평균값 활용 >> 두가지 평균 방법 (micro average, macro average)
- Micro average : 각 이진분류 confusion matrix의 TP, FP, TN, FN 요소들을 가지고 precision, recall 등을 계산
- Macro average : 각 이진분류 confusion matrix에서 먼저 precision, recall 등을 계산하고 평균

Micro Average

	강아지	그외
강아지	TP_1	FN_1
그외	FP_1	TN_1

	고양이	그외
고양이	TP_2	FN_2
그외	FP_2	TN_2

	토끼	그외
토끼	TP_3	FN_3
그외	FP_3	TN_3

- Precision : $TP/(TP+FP)$
- Micro precision : $(TP_1+TP_2+TP_3)/(TP_1+TP_2+TP_3+FP_1+FP_2+FP_3)$
- Recall : $TP/(TP+FN)$
- Micro recall : $(TP_1+TP_2+TP_3)/(TP_1+TP_2+TP_3+FN_1+FN_2+FN_3)$

Macro Average

	강아지	그외
강아지	TP_1	FN_1
그외	FP_1	TN_1

$precision_1, recall_1, F1_1$

	고양이	그외
고양이	TP_2	FN_2
그외	FP_2	TN_2

$precision_2, recall_2, F1_2$

	토끼	그외
토끼	TP_3	FN_3
그외	FP_3	TN_3

$precision_3, recall_3, F1_3$

- Macro Precision = (Precision$_1$ + Precision$_2$ + Precision$_3$) / 3
- Macro Recall = (Recall$_1$ + Recall$_2$ + Recall$_3$) / 3
- Macro F1 = (F1$_1$ + F1$_2$ + F1$_3$) / 3

회귀 문제 메트릭 - MAE, MAPE, MSE

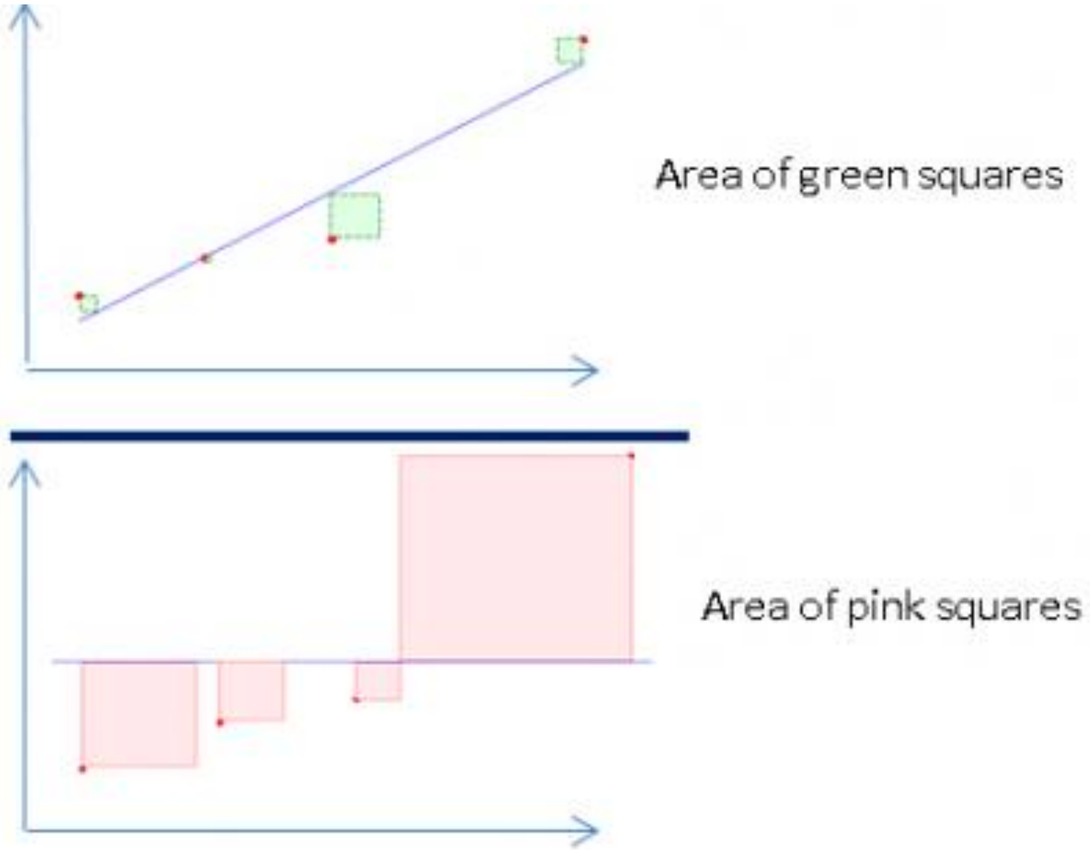
- N : data 샘플 수, y : 실제값, $\hat{y}$: 예측값
- MAE (mean absolute error) : $\frac{1}{N}\sum|y-\hat{y}|$
- MAPE (mean absolute percentage error) : $\frac{100}{N}\sum|\frac{y-\hat{y}}{y}|$
- MSE (mean squared error) : $\frac{1}{N}\sum(y-\hat{y})^2$
- 모두 작을 수록 좋음.

회귀 문제 메트릭 - R-squared

- N : data 샘플 수, y : 실제값, $\hat{y}$: 예측값, $\bar{y}$: y값들의 평균
- SST (total sum of squares) : $\sum (y - \bar{y})^2$
- SSE (residual sum of squares): $\sum (y - \hat{y})^2$
- R_squared = 1 - (SSE/SST) → 1에 가까울 수록 좋음.
  - 작은 SSE : 예측 오차가 작음.
  - 큰 SST : 예측이 어려움 (variance가 큼)

R-squared

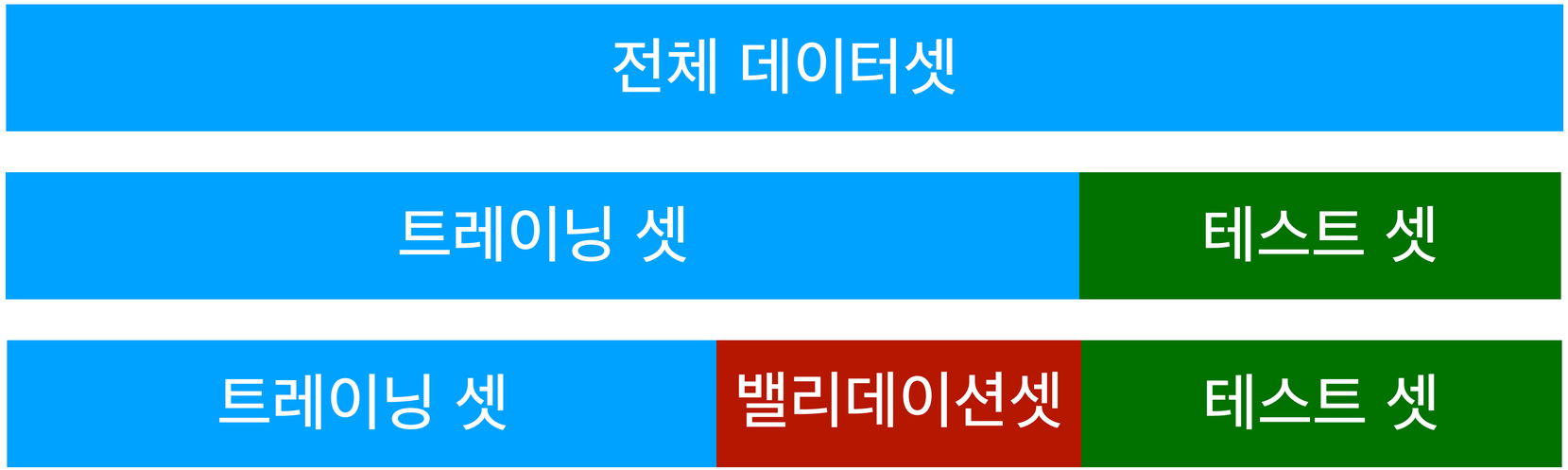
$$R^2 = 1 - \frac{\text{Area of green squares}}{\text{Area of pink squares}}$$



평가지표의 적용 대상

- Training set: 학습 과정에 문제점은 없는지 체크
- Validation set: 하이퍼 파라미터 변경에 따른 평가지표 비교를 통해 최적 하이퍼파라미터 계산
- Test set: 모델 일반화 성능에 대한 평가

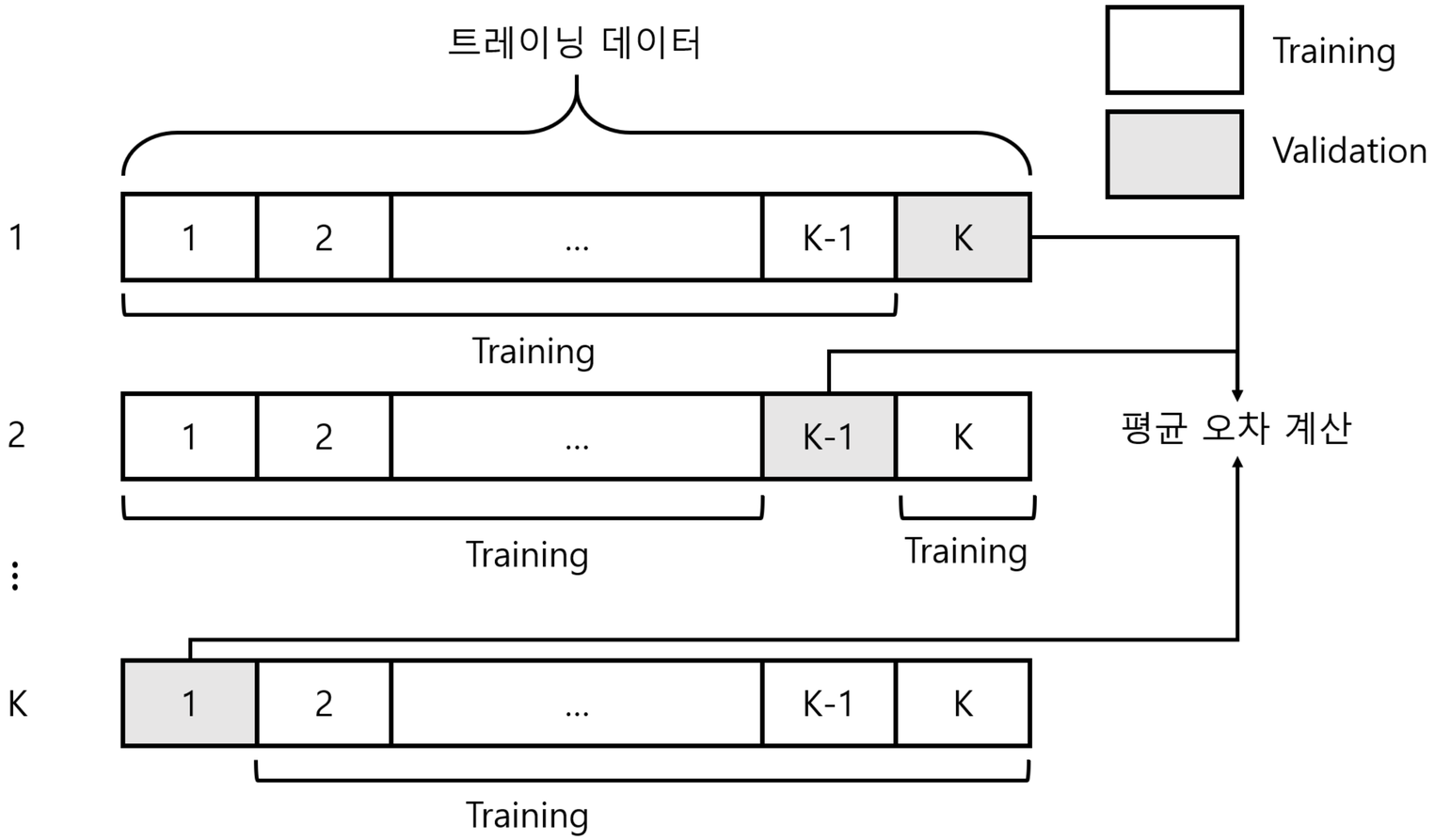
Hold-out validation



Hold-out validation

- 충분히 큰 크기의 데이터셋에서 사용
- 학습데이터/테스트데이터가 서로 독립적이면서 전체 데이터의 분포를 모델링하기에 충분해야 함.
- 한번의 평가로 충분하지 않은 경우 Cross validation 적용
- 일반적인 비율 : training/test : 7:3, 8:2, training/validation/test : 6:2:2, 7:1:2 등
- 대규모 데이터 또는 프로토타이핑에 적합

K-fold cross validation



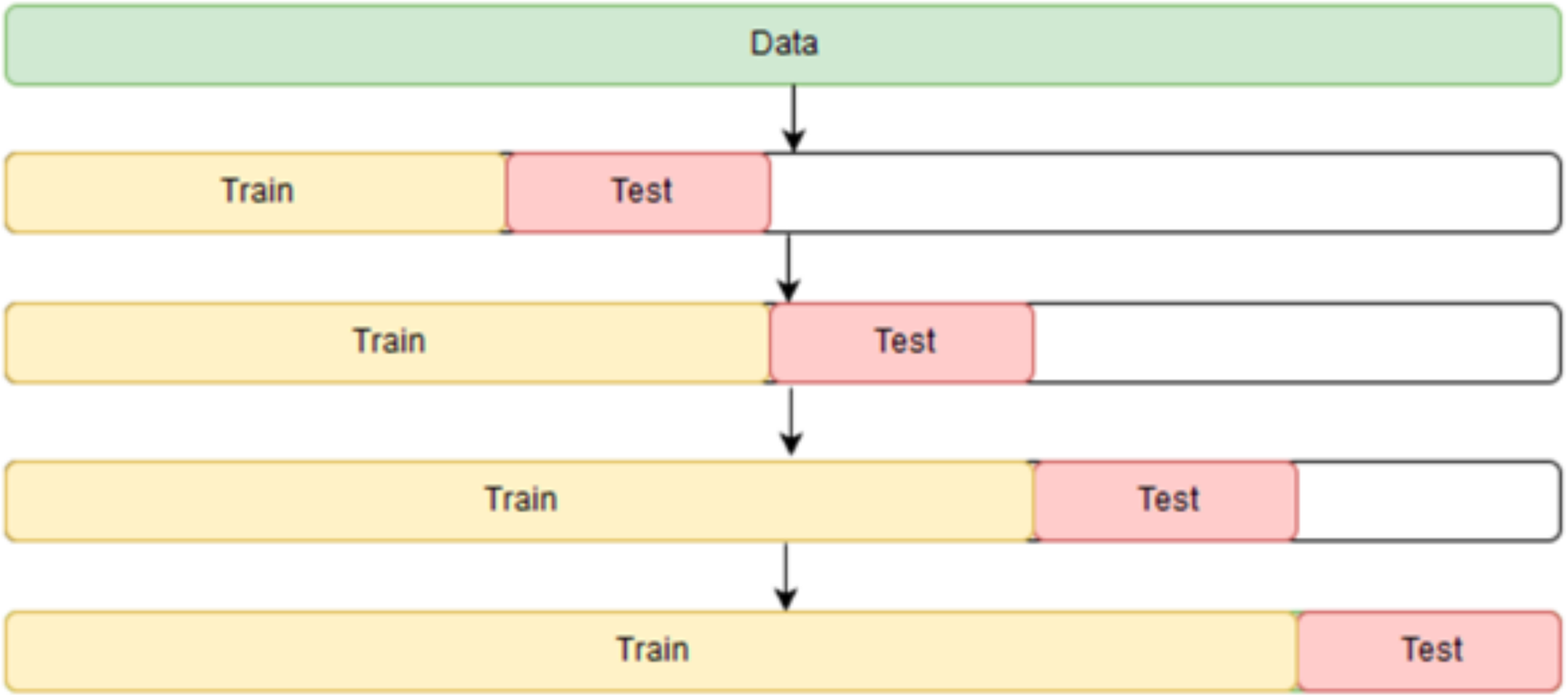
K-fold CV pseudo code

```
Initialize:
    scores = []

Split data into K folds → [(X_train_1, X_valid_1), ..., (X_train_K, X_valid_K)]

For k in 1 to K:
    X_train_k, y_train_k ← training set in fold k
    X_valid_k, y_valid_k ← validation set in fold k

   model_k.fit(X_train_k, y_train_k)

    y_pred_k ← model_k.predict(X_valid_k)
    score_k ← score_fn(y_valid_k, y_pred_k)

    Append score_k to scores

Final Score ← average(scores)
```

K-fold CV

- 전체 데이터를 K개의 서브셋(폴드)으로 분할, 특정 서브셋을 테스트, 나머지는 학습하여 K회 측정 성능의 통계 비교
- 장점 :
  - 학습/테스트 데이터의 구성이 다른 수회의 결과에 대해 평가하므로 신뢰성 증가.
  - 적은 데이터에도 활용가능
- 단점 :
  - K번의 훈련과 검증을 수행하므로 시간과 계산비용이 증가함.
- K-fold의 변형
  - Stratified K fold : 각 폴드를 구성하는 클래스의 비율이 전체 클래스의 비율과 동일하게 구성
  - Leave-One-Out : 테스트 폴드가 단일 데이터 샘플로 구성

Time series split



하이퍼 파라미터

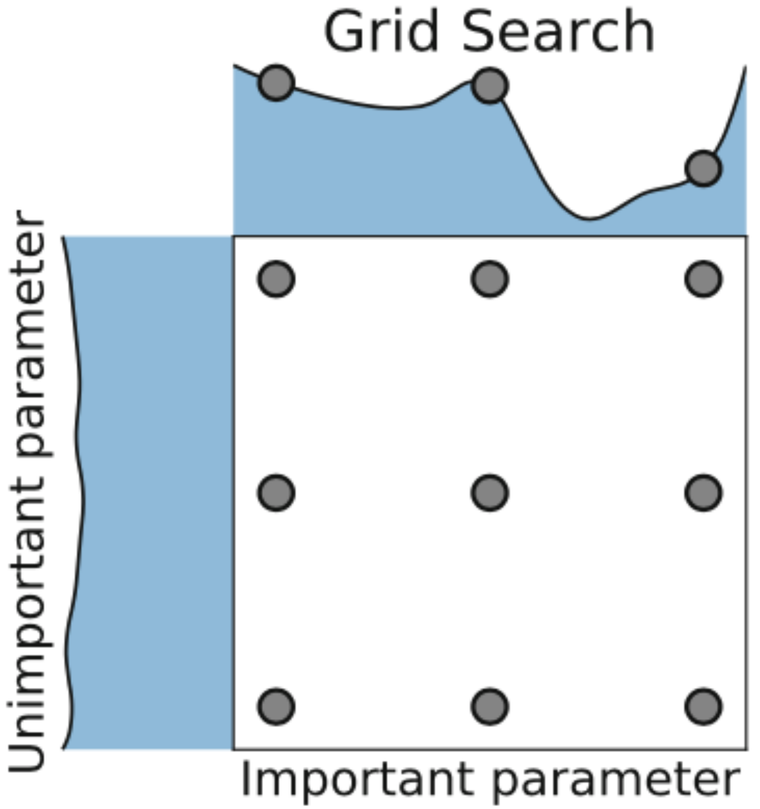
- 하이퍼 파라미터 :
  - 학습하고자 하는 머신러닝 모델에 대해서 모델 구조, 학습 과정 등과 관련하여 사전에 설정해주는 변수들.
  - 학습을 통해 배우는 파라미터는 아니지만, **모델 학습 결과에 큰 영향을 미침.**
- 머신러닝 모델 별 하이퍼 파라미터 예시
  - 신경망 : 뉴런의 수, 레이어의 수, 모델 구조, 활성 함수, learning rate, optimizer, etc.
  - 서포트벡터머신 : 커널의 종류, 마진(margin)
  - 의사결정트리, 랜덤포레스트, lightGBM : 트리의 수, 최대 깊이, 서브샘플링 여부 등

하이퍼 파라미터 탐색 방법

- Manual Search
- Grid Search
- Random Search
- Bayesian Optimization

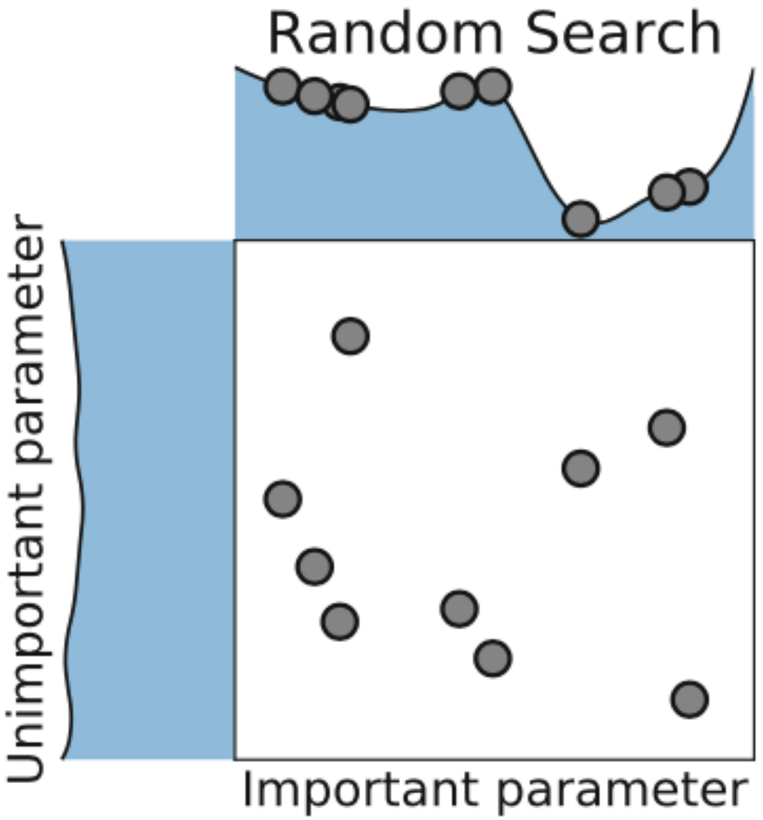
Grid search

- 하이퍼 파라미터 변수 별로 후보 군을 선정
- 모든 조합에 대하여 모델을 평가함.
- Curse of dimensionality :
  하이퍼 파라미터 Λ의 차원이 증가함에 따라서 필요한 평가 횟수가 지수적으로 증가.



Random search

- 하이퍼 파라미터 변수의 분포를 설정.
- 설정된 분포 안에서 샘플링하여 평가.
- 특정 변수의 영향이 클때 더욱 잘 동작함.
- 무작위로 샘플링하기때문에 변수값이 매번 달라짐.



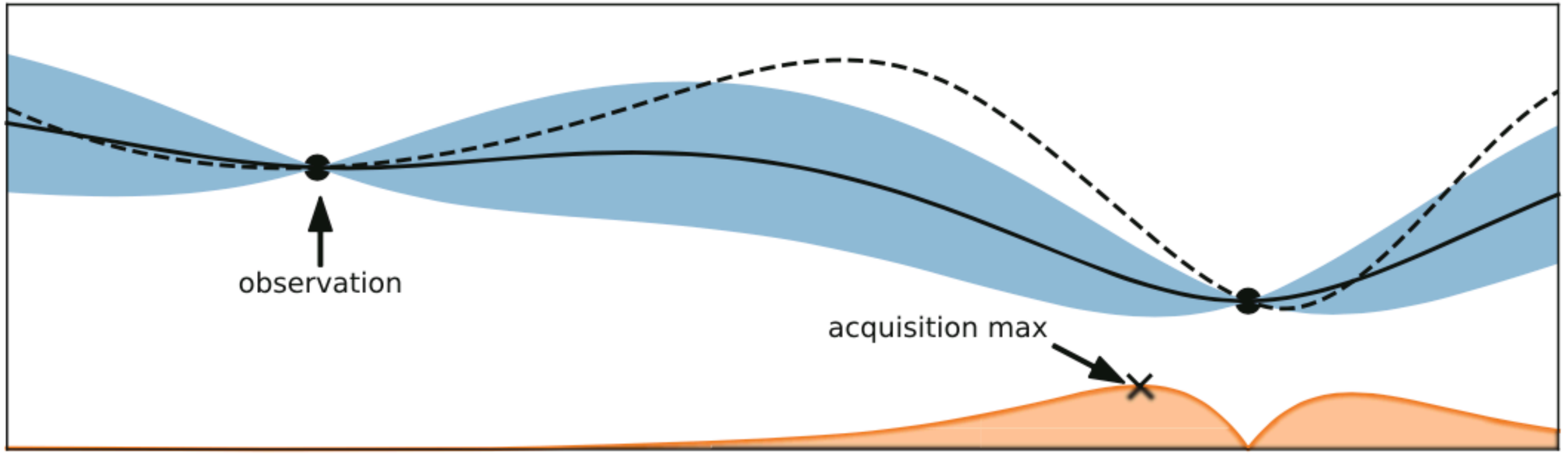
Bayesian Optimization

- 베이지안 최적화 :
  함수의 형태를 가정하지 않는 블랙박스 함수의 전역 최적화를 위한 순차적 전략

- Surrogate model:
  원래 함수를 대체 하는 모델.
  HPO에서는 하이퍼파라미터에 대한 머신러닝 모델의 성능 함수의 대리 모델로,
  주로 가우시안 프로세스(GP)를 활용. GP 대신 NN, RF 기반 방법도 존재.

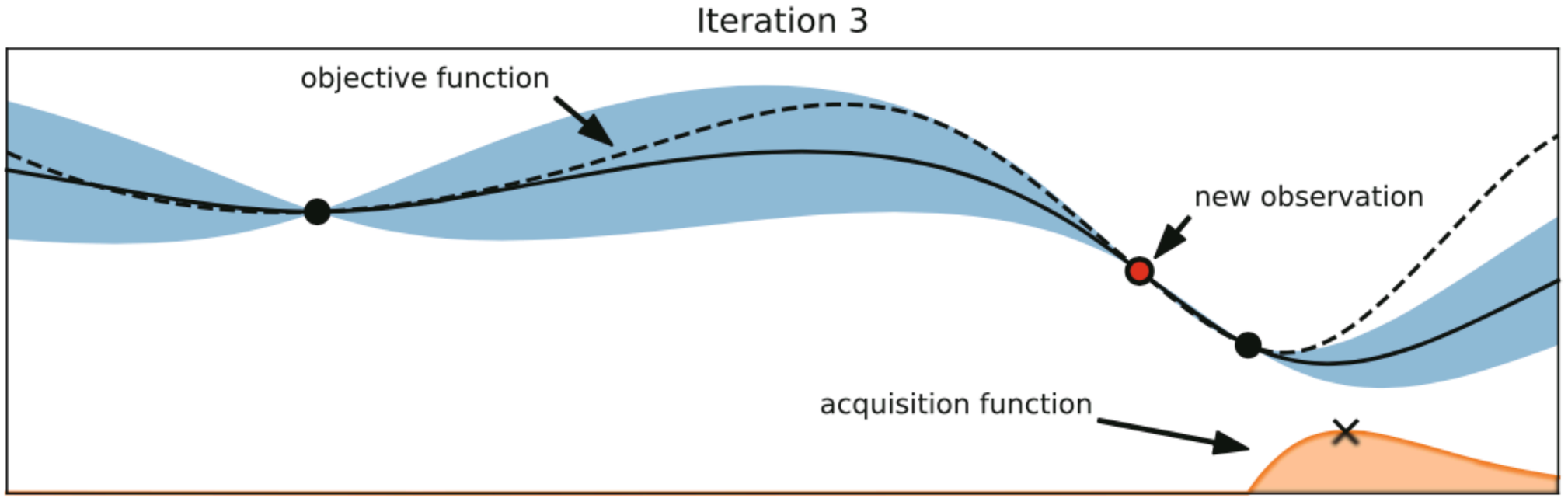
- Acquisition function:
  과거의 결과로부터 탐색 해야 할 파라미터를 결정해주는 함수.

베이지안 최적화 예시

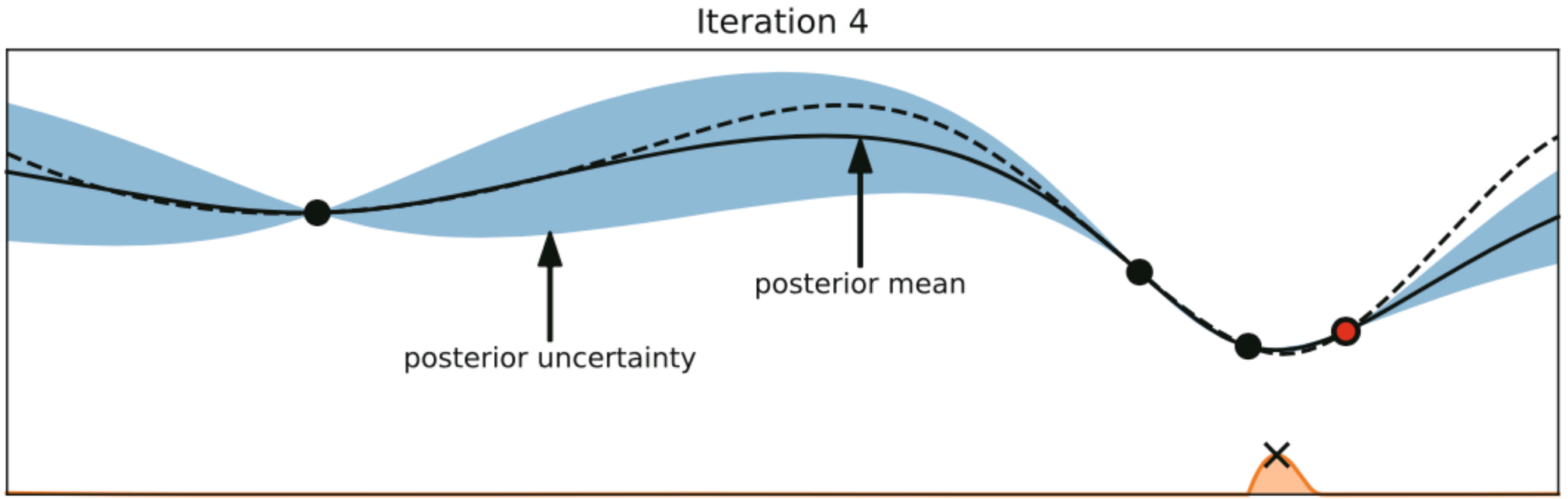


Iteration 3

베이지안 최적화 예시


Iteration 3
objective function
new observation
acquisition function

베이지안 최적화 예시



하이퍼 파라미터 탐색 효율성

- 세개의 하이퍼파라미터를 가정 : A, B, C (e.g., 신경망의 레이어 수)
- 비교를 위해서는 3번의 모델 학습과 평가가 필요 (validation set 사용)
- CV 적용 시 validation set의 수를 곱한 만큼 모델 학습과 평가가 필요
- 하이퍼파라미터가 여러개라면?
- $|H_1| \times |H_2| \times \cdots \times |H_K| \times |CV|$ >> 컴퓨팅 자원 + 시간의 문제

Multi-fidelity optimization

- Fidelity (충실도) : High-fidelity (Hi-Fi) & Low-fidelity (Lo-Fi)
- Recall : 함수 평가가 매우 비용이 많이 들 수 있음.
  → 하이퍼파라미터가 다른 여러 함수에 대한 평가가 수반되어야 함.
- 더 어려운 태스크, 더 복잡한 모델 구조, 더 많은 데이터 → HPO에서의 주요 문제점
- 직관적인 접근은? 간단한 태스크, 간단한 데이터에 대해서 탐색하고 실제 문제로 확장
- 실제 loss function에 대한 Low-fidelity approximation.

Successive halving

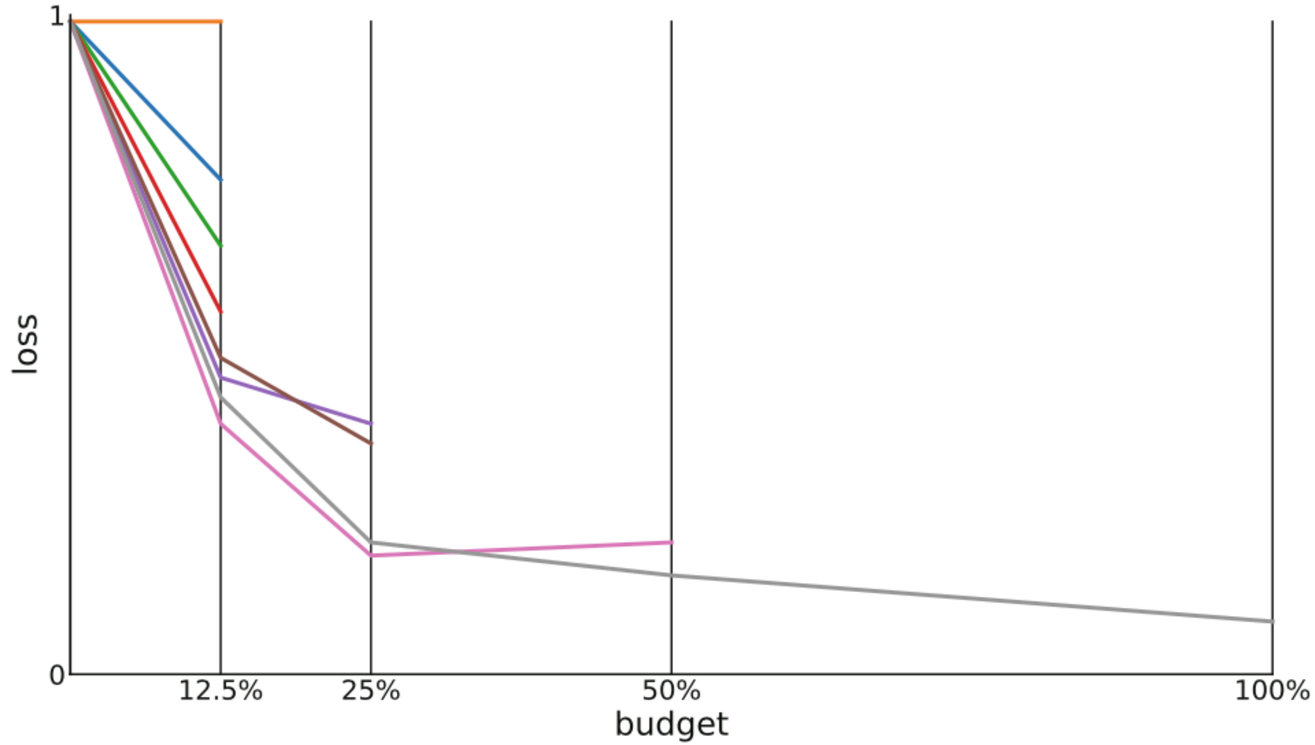


Fig. 1.3 Illustration of successive halving for eight algorithms/configurations. After evaluating all algorithms on $\frac{1}{8}$ of the total budget, half of them are dropped and the budget given to the remaining algorithms is doubled