Microsoft

자동화된 ML, 나도 해보자

Episode 1: 자동화된 ML이 왜 필요한가

한석진
마이크로소프트

Episode 1
자동화된 ML이 왜 필요한가

머신러닝 (간단하게) / 모델 생성 과정

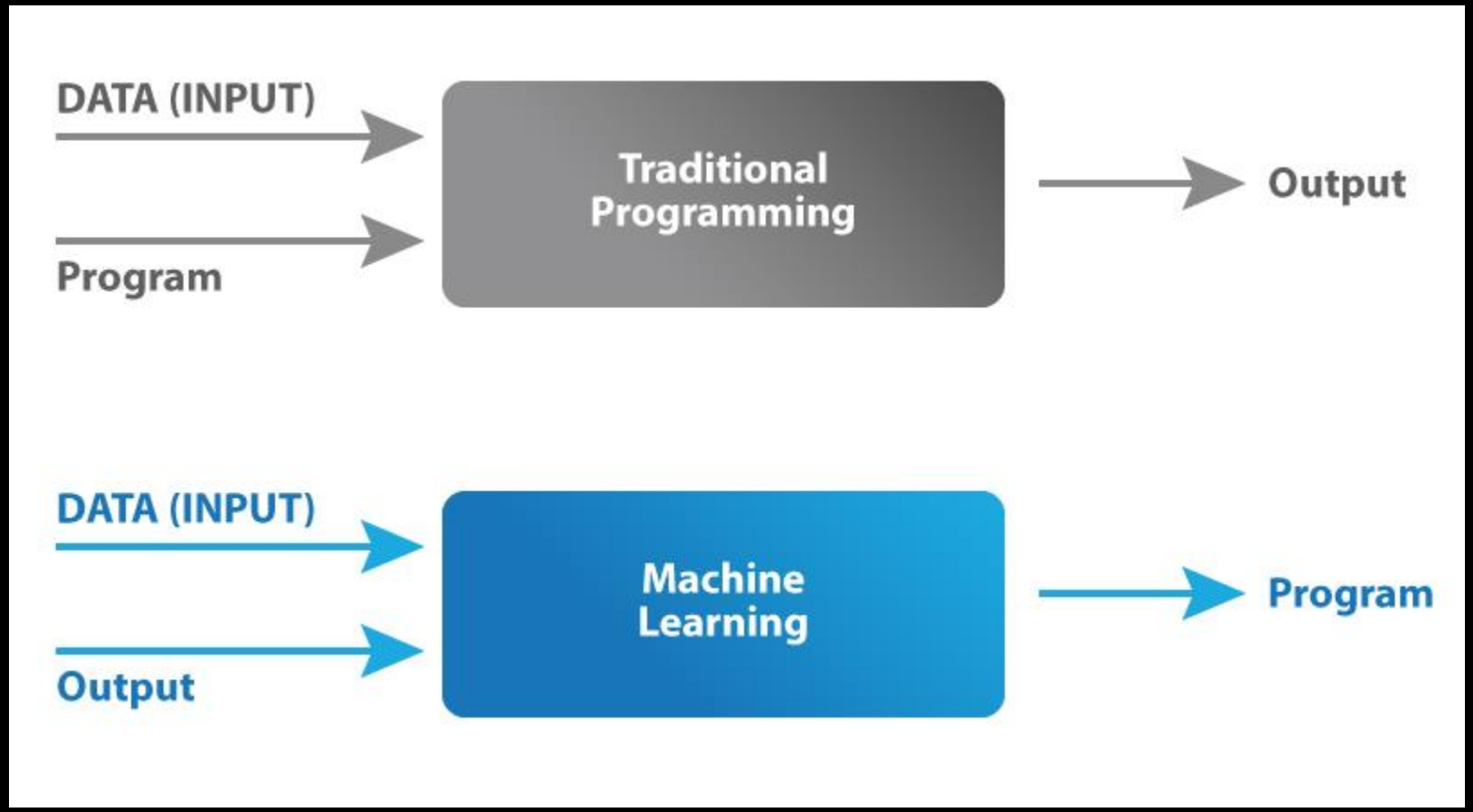
자동화된 ML 기본 구조

오버피팅 (과적합) / 클래스 불균형

자동화된 ML의 역할 / 사람의 역할

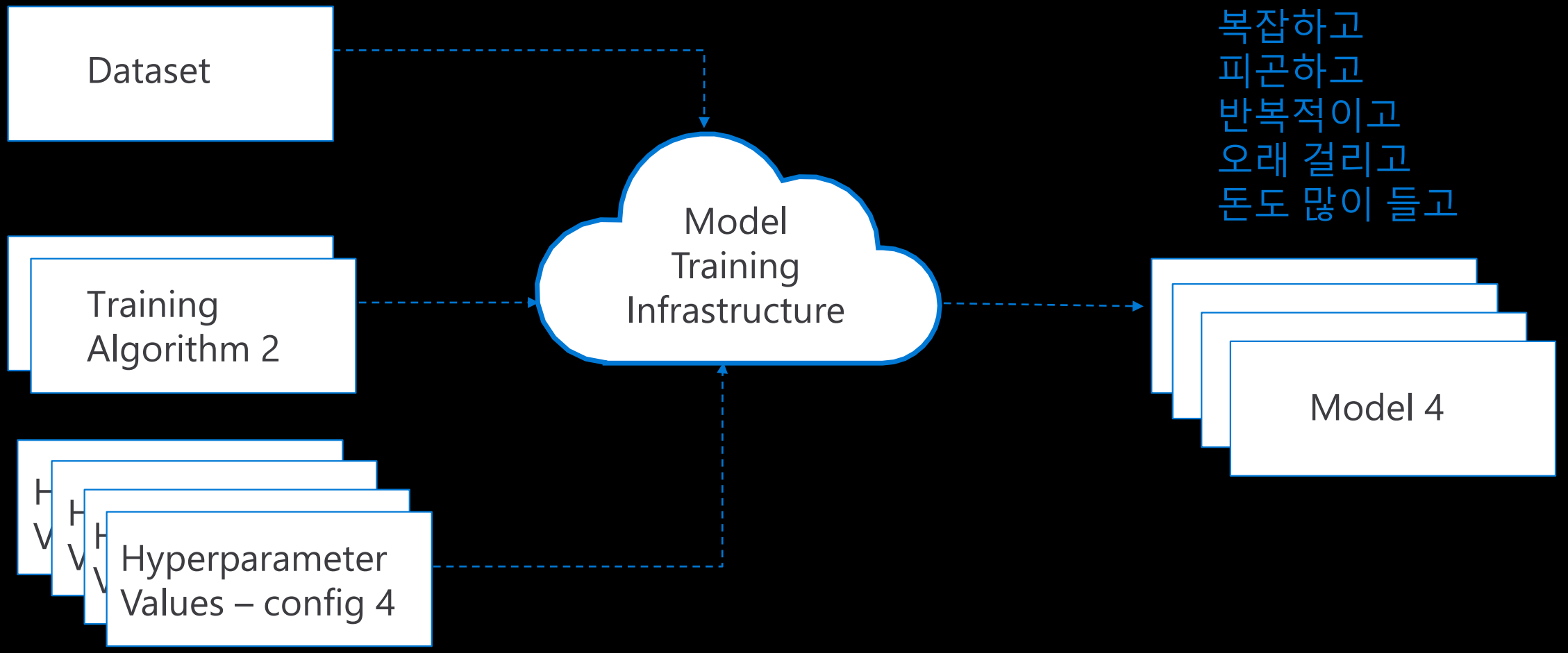
자동화된 부분 더 알아보기

머신러닝 (간단하게)

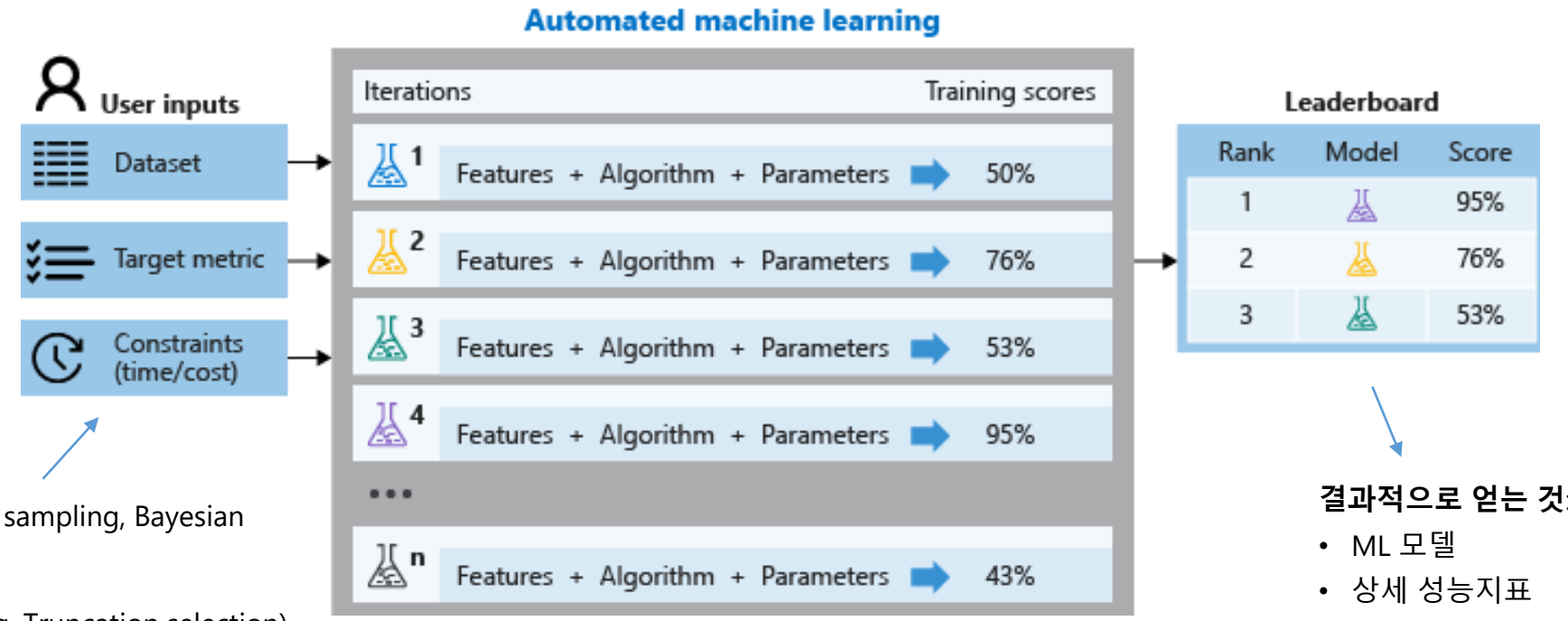


https://www.greatlearning.in/blog/what-is-machine-learning/

모델 생성 과정


Dataset
Training Algorithm 2
Hyperparameter Values – config 4
Model Training Infrastructure
Model 4
복잡하고
피곤하고
반복적이고
오래 걸리고
돈도 많이 들고

기본 구조



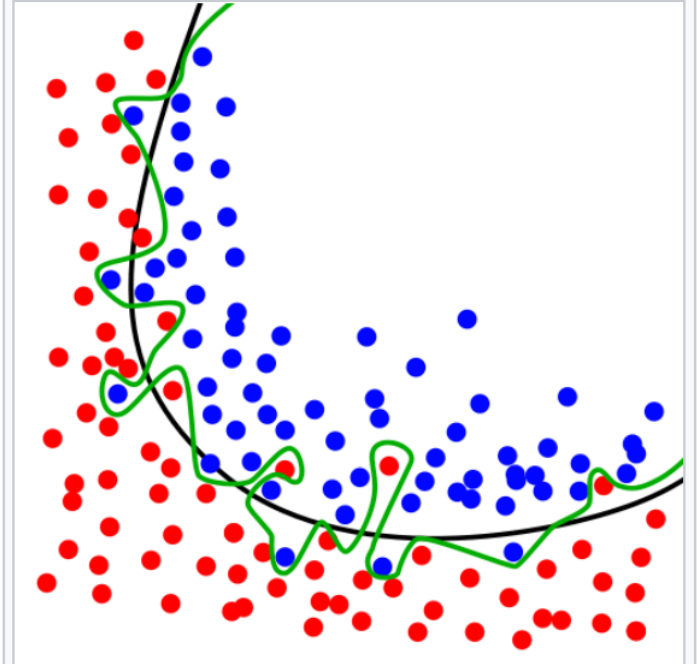
그 외 선택할 수 있는 것:

- Hyperparameter 범위
  (Grid sampling, Random sampling, Bayesian optimization)
- Early Termination Policy
  (Bandit, Median stopping, Truncation selection)
- 동시에 실행할 파이프라인 개수
- 안 쓸 알고리즘
- 원격 실행 (대용량 클러스터에서)
- 기존 실행한 것에서 이어서 하기
- Sample weight 지정
- Subsampling

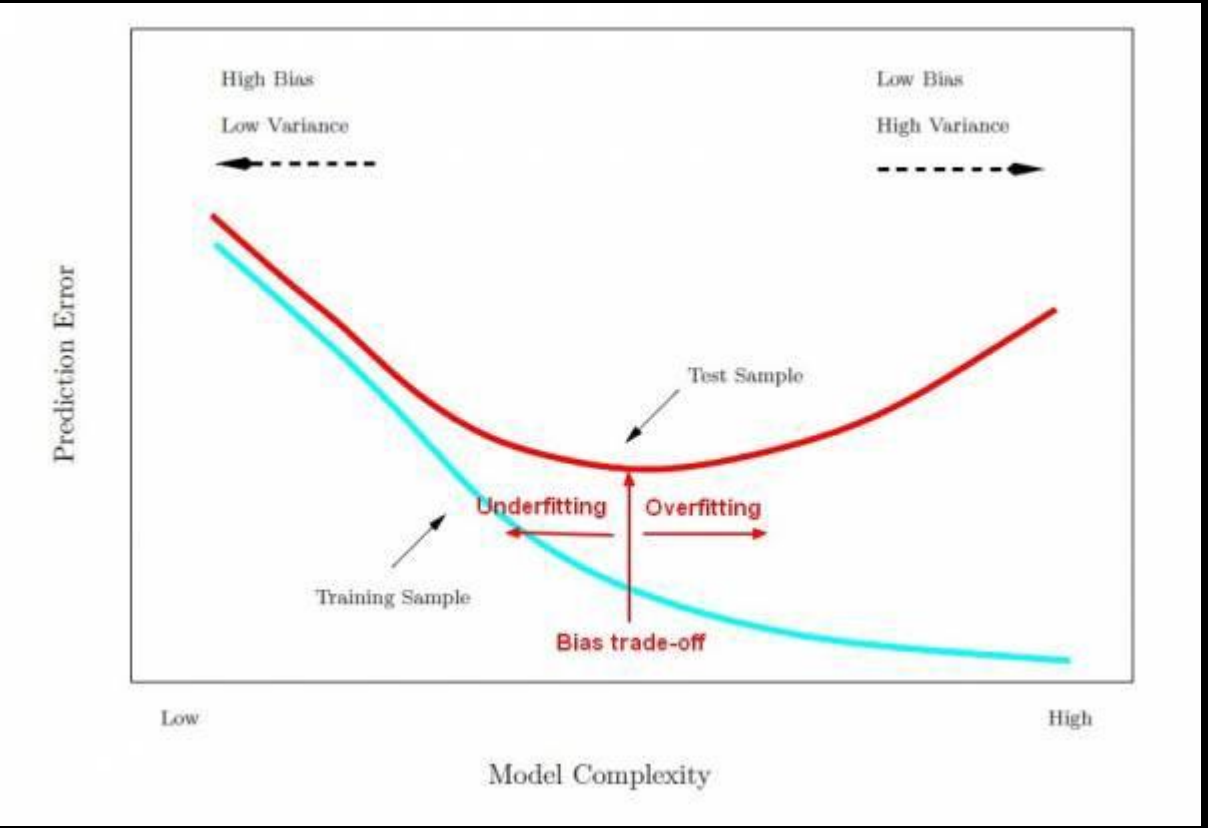
결과적으로 얻는 것:

- ML 모델
- 상세 성능지표
- 상세 hyperparameter 설정

Overfitting (과적합)



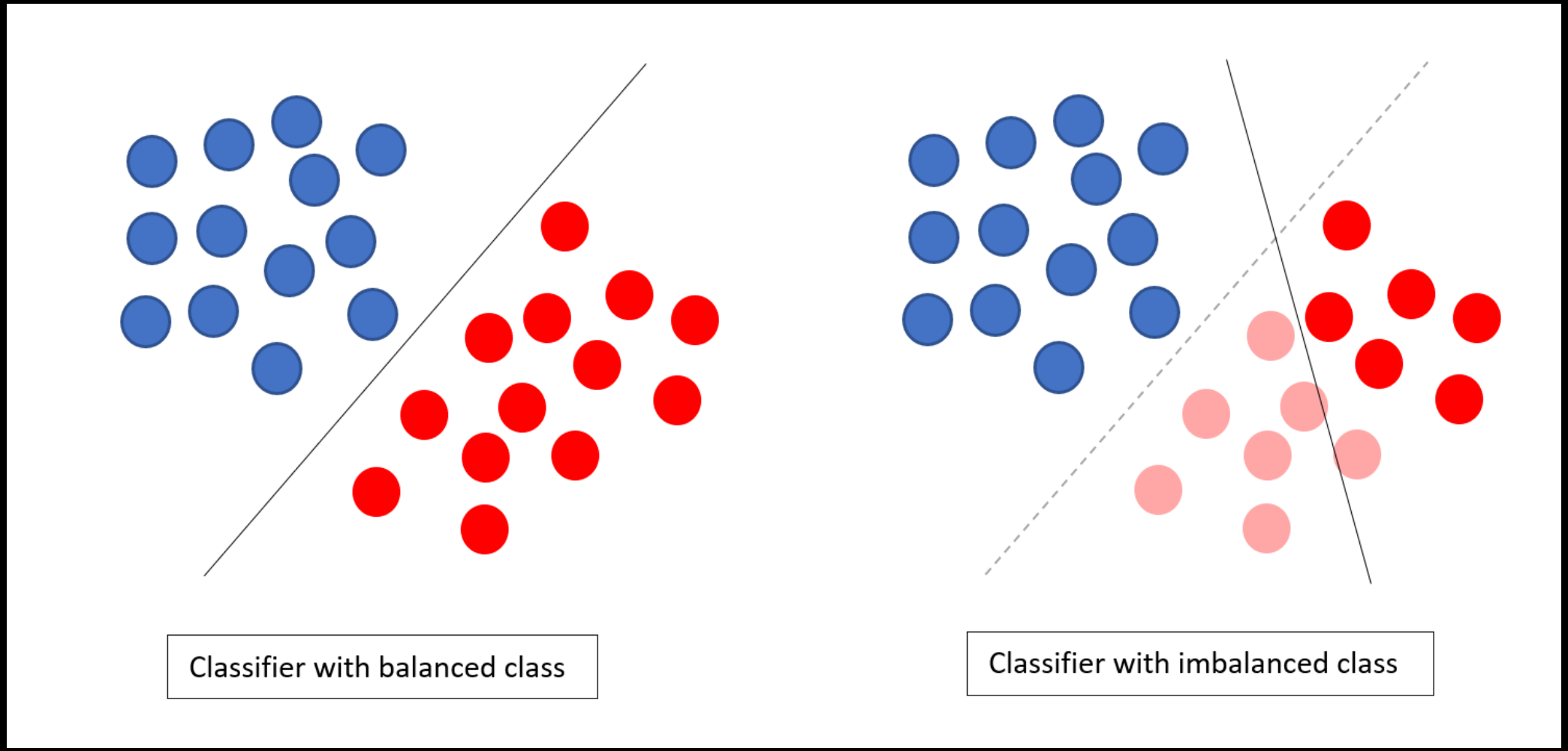
초록색 선은 과적합된 모델을, 검은색 선은 일반 모델을 나타낸다.



https://ko.wikipedia.org/wiki/%EA%B3%BC%EC%A0%81%ED%95%A9

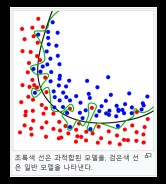
https://medium.com/ml-research-lab/chapter-3-bias-and-variance-trade-off-in-machine-learning-a449fa1e2729

Class Imbalance (클래스 불균형)

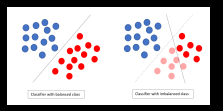


https://medium.com/x8-the-ai-community/solving-class-imbalance-problem-in-cnn-9c7a5231c478

자동화된 ML이 해주는 것과 우리가 할 것



과적합



클래스 불균형

자동화됨

- Regularization (정칙화/정규화), Hyperparameter (하이퍼파라미터) 튜닝
- 명시적으로 모델 복잡도 제한 (Tree 계열)
- Cross Validation (교차 검증)

- 불균형을 확인할 수 있는 시각화 생성
  - Confusion Matrix, Precision-Recall, ROC Curve 등
- Weight 컬럼 (중요한 row를 강조)
- 불균형을 측정하고 sub-sampling 시도
- 불균형에 robust한 AUC_weighted 등 평가

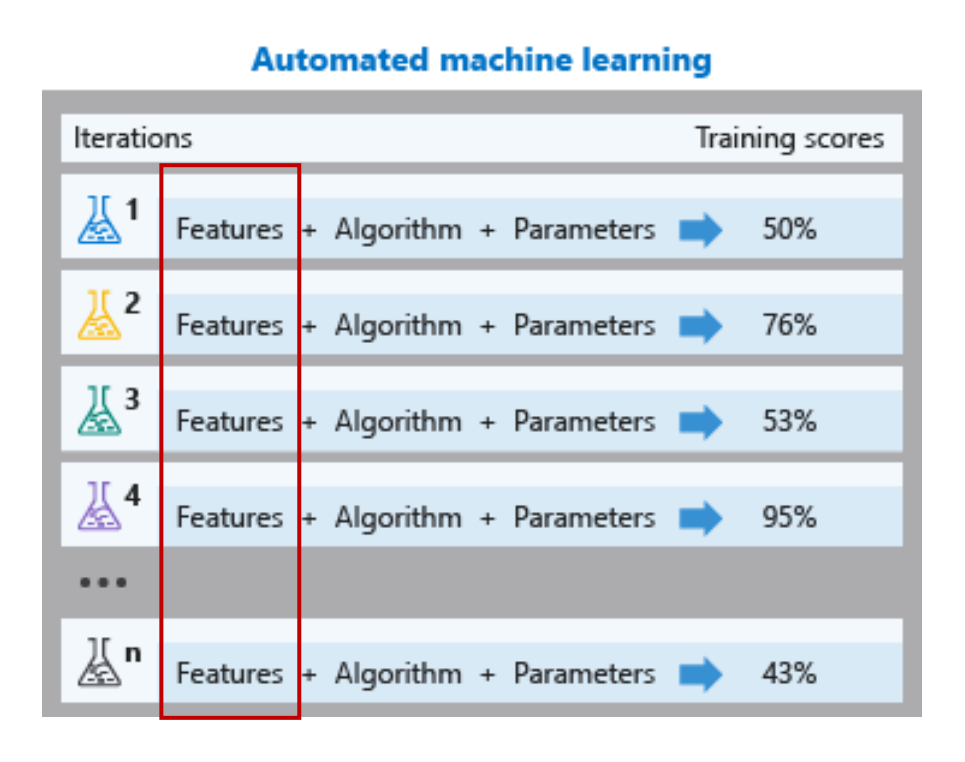
우리가 할 일

- 학습데이터 추가 확보, 통계적 bias 제거
- Target Leakage 방지
- Feature 수 줄이기

- Up-sampling 또는 Down-sampling
- 성능지표 평가
  - F1: Precision과 Recall의 조화평균
  - Precision이 높으면 FP가 적음
  - Recall이 높으면 FN이 적음

https://docs.microsoft.com/en-us/azure/machine-learning/concept-manage-ml-pitfalls

Feature 가공 단계



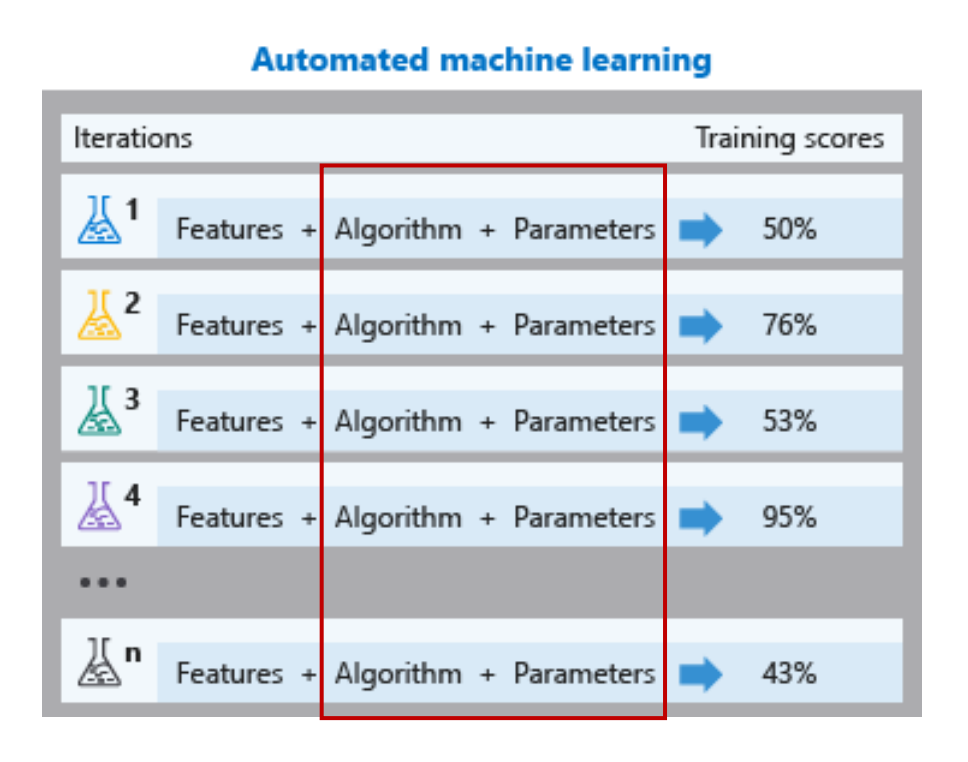
기본적으로

- **StandardScaleWrapper / MinMaxScaler / MaxAbsScaler / RobustScalar / PCA**
- **TruncatedSVDWrapper** (scipy.sparse 행렬에 유리) / **SparseNormalizer**

더 깊이 들어가서

- **카디널리티가 높거나 분산이 0인 속성은 제거 / 빠진 값 채우기 (imputation)**
- **추가 속성 만들기**: DateTime, Text (TF 기반)
- **가공 및 인코딩**
  - 숫자형 -> 범주형 (카디널리티가 낮을 때)
  - One-hot 인코딩 (카디널리티가 낮을 때), One-hot-hash 인코딩 (카디널리티가 낮을 때)
- **워드 임베딩**
  - 토큰 -> sentence 벡터 (pre-trained 모델 사용: GloVe)
  - 워드 임베딩 -> document feature 벡터
- **타겟 인코딩** (범주형 속성의 경우)
  - 특히 오버피팅과 sparsity로 인한 노이즈를 피하기 위해 frequency-based weighting과 k-fold CV를 적용
- **텍스트 타겟 인코딩:** 각 class별 확률 계산을 위해 stacked linear model with bag-of-words를 사용
- **Weight of Evidence** (WoE): class별로 in-class vs out-of-class 확률비율의 log로 계산, 명시적으로 missing 값이나 outlier 처리할 필요성을 없앰
- **Cluster distance** (k-means)

알고리즘과 하이퍼파라미터 선택



Classification	Regression	Time Series Forecasting
Logistic Regression*	Elastic Net*	Elastic Net
Light GBM*	Light GBM*	Light GBM
Gradient Boosting*	Gradient Boosting*	Gradient Boosting
Decision Tree*	Decision Tree*	Decision Tree
K Nearest Neighbors*	K Nearest Neighbors*	K Nearest Neighbors
Linear SVC*	LARS Lasso*	LARS Lasso
Support Vector Classification (SVC)*	Stochastic Gradient Descent (SGD)*	Stochastic Gradient Descent (SGD)
Random Forest*	Random Forest*	Random Forest
Extremely Randomized Trees*	Extremely Randomized Trees*	Extremely Randomized Trees
Xgboost*	Xgboost*	Xgboost
Averaged Perceptron Classifier	Online Gradient Descent Regressor	Auto-ARIMA
Naive Bayes*	Fast Linear Regressor	Prophet
Stochastic Gradient Descent (SGD)*		ForecastTCN
Linear SVM Classifier*		

+ 앙상블 모델 (Ensemble Model) 비교평가

- **Stack Ensemble**
- **Voting Ensemble**

Episode 1
자동화된 ML이 왜 필요한가

머신러닝 (간단하게) / 모델 생성 과정

자동화된 ML 기본 구조

오버피팅 (과적합) / 클래스 불균형

자동화된 ML의 역할 / 사람의 역할

자동화된 부분 더 알아보기

{다음 시간에는}

Episode 2
애저ML
처음 시작하기

애저포털과 애저ML포털
애저ML 워크스페이스 만들기
애저ML 스튜디오 둘러보기