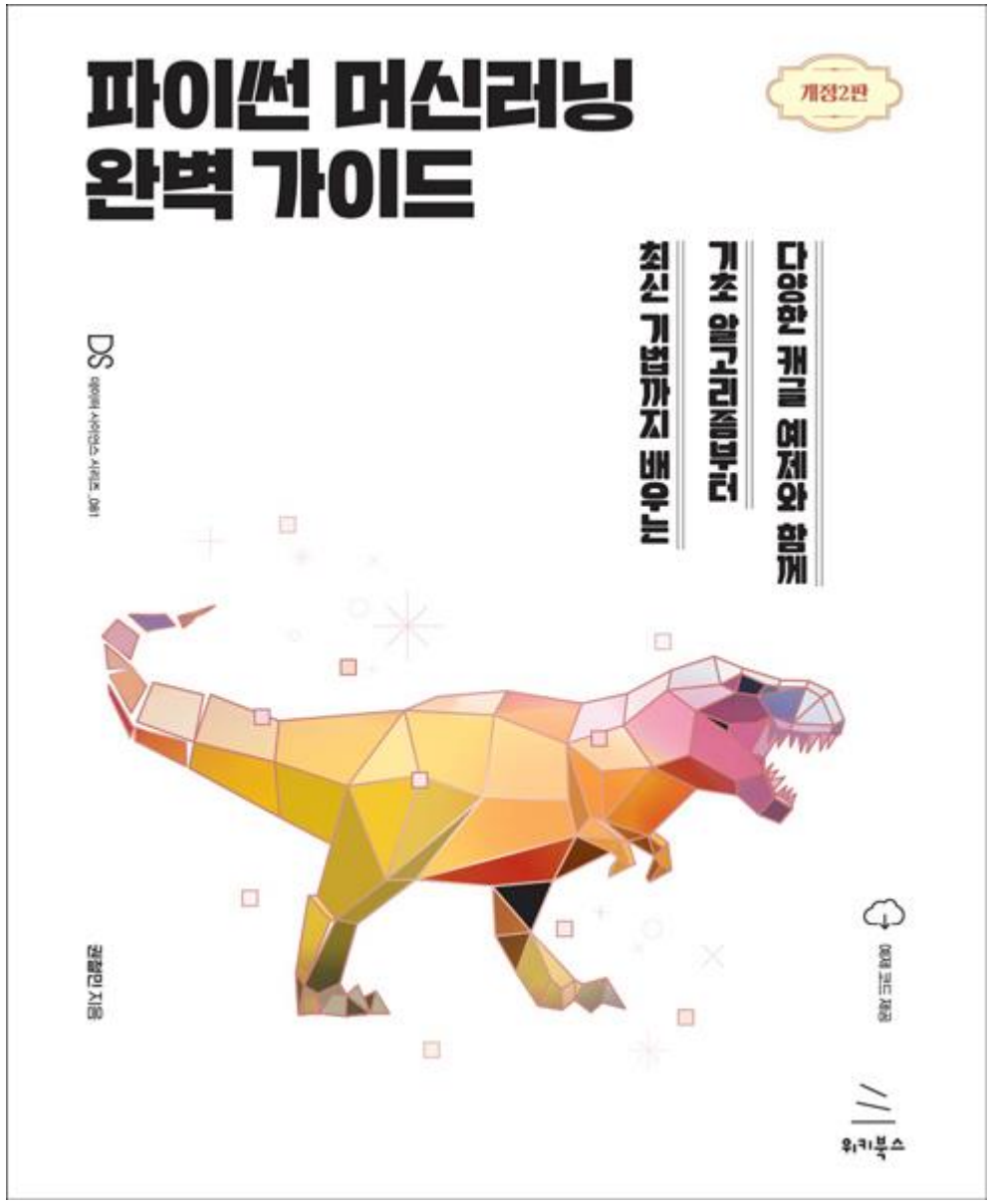
2023 Machine Learning Odyssey Pt. 2

Seungeun Lee

“2023 Machine Learning Odyssey”

Part 1	2023.07.24.	Tabular data, Data preprocessing, Evaluation metrics, Cross validation, imputation
Part 2	2023.07.24., 2023.08.18.	AutoML, Feature selection/extraction, Data Imbalance, Data Preprocessing 2, XGBoost 1
Part 3	TBD	XGBoost 2, Hyperparameter tuning, Gridsearch
Part 4	TBD	SVM, RandomForest, Clustering, Dimension reduction
Part 5	TBD	LightGBM, Ensemble Models
Part 6	TBD	ML v.s. DL (TabNet, ⋯), Future of Machine Learning (Causal Inference, Bayesian)
Part 7	TBD	Meta Learning & Meta Reinforcement Learning
Part 8	TBD	MLOps, Multi-modal analysis



https://github.com/wikibook/pymlrev2

AutoML이란

AutoML-pycaret

- 적은 코드로 머신러닝 워크 플로우를 자동화하는 오픈 소스 라이브러리
- 코딩, 전처리, 모델 선택, 파라미터 튜닝 작업을 한 번에!

- https://pycaret.org/guide/

- 첨부한 Jupyter Notebook 파일

AutoML 사용법

Machine Learning 문제를 푼다면…

(1) AutoML로 분석
(2) AutoML Best 3 or 5 정도 모델 선정
(3) Best 3 or 5개의 모델을 직접 모델링 (python wrapper, scikit-learn)
(4) (선택) Ensemble 기법 적용

AutoML 사용법

- 주의점

- AutoML을 맹신하지 말 것

- python wrapper / scikit-learn / AutoML로 각각 같은 모델 (e.g. XGBoost)을 만들어도 결과는 다르게 나온다

- 보통, AutoML < scikit-learn < python wrapper로 알려져 있다

- 또한, AutoML이 찾은 최적의 하이퍼파라미터라고 주장하는 값은 직접 python wrapper나 scikit-learn으로 모델링해보면 최적의 값이 아닐 확률이 높다

AutoML 사용법

- 주의점

- AutoML을 설치할 때 다른 패키지 (numpy, pandas, matplotlib) 등과 충돌할 가능성이 크다 -> 독립적인 가상환경을 만들어서 설치하는 것 추천

- 그래서 AutoML 패키지 자체의 모델을 저장하는 것 (.model, .pkl)은 추천하지 않음

- “AutoML은 어떤 모델을 점검해볼 것인가?”의 우선 순위를 세우는 역할을 할 뿐
- 모델의 “경향성“을 보여주는 가이드로서의 역할만 기대하자

To be Updated

- Data Preprocessing 2:
- Feature selection/extraction, Multicollinearity, Data Imbalance, Real-world issues

- XGBoost 1

Multicollinearity [다중공선성]

- 회귀분석에서 독립 변수들 간에 강한 상관관계가 나타나는 문제
- 독립 변수는 종속 변수와만 관계가 있어야 하며, 독립변수들 간에 관계가 생기면 안된다
- 다중공선성이 관찰되는 경우, 부정확한 회귀 결과가 도출될 수 있다

< 다중공선성 처리 가이드 라인 >
VIF 10 이상: 삭제 (이론적으로 권장)
VIF가 10 이상인 feature가 70% 이상인 경우:
VIF가 10 이상인 feature 중 큰 순서대로
30-50% 정도만 삭제

(경험적인 수치일 뿐, 이론적인 것 아님)

Multicollinearity [다중공선성]

- VIF (Variance Inflation Factors, 분산팽창요인/인수)

$$VIF_i = \frac{1}{1 - r_i}$$

r_i = i 번째 변수를 제외한 회귀식의 R^2값

$$VIF_i > 10 \Leftrightarrow \frac{1}{1 - r_i} > 10$$

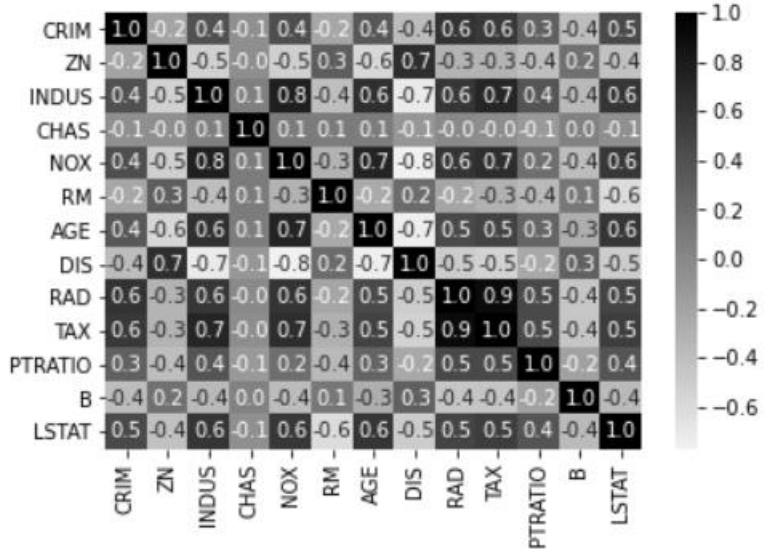
$$1 > 10 - 10 r_i$$

$$r_i > 0.9$$

- i번째 독립변수가 빠져도 나머지 변수들이 종속 변수 (y)를 90% 이상 설명한다는 것
- i번째 독립변수 없이도 충분히 종속 변수 (y)를 잘 설명할 수 있으므로, i번째 독립변수는 없어도 된다는 것을 의미

Multicollinearity [다중공선성]

- https://ysyblog.tistory.com/122



	VIF Factor	features
0	1.152952	CHAS
1	2.100373	CRIM
2	2.844013	ZN
3	11.102025	LSTAT
4	14.485758	INDUS
5	14.699652	DIS
6	15.167725	RAD
7	20.104943	B
8	21.386850	AGE
9	61.227274	TAX
10	73.894947	NOX
11	77.948283	RM
12	85.029547	PTRATIO

- 산점도 (scatter plot)
- Pearson correlation coefficient

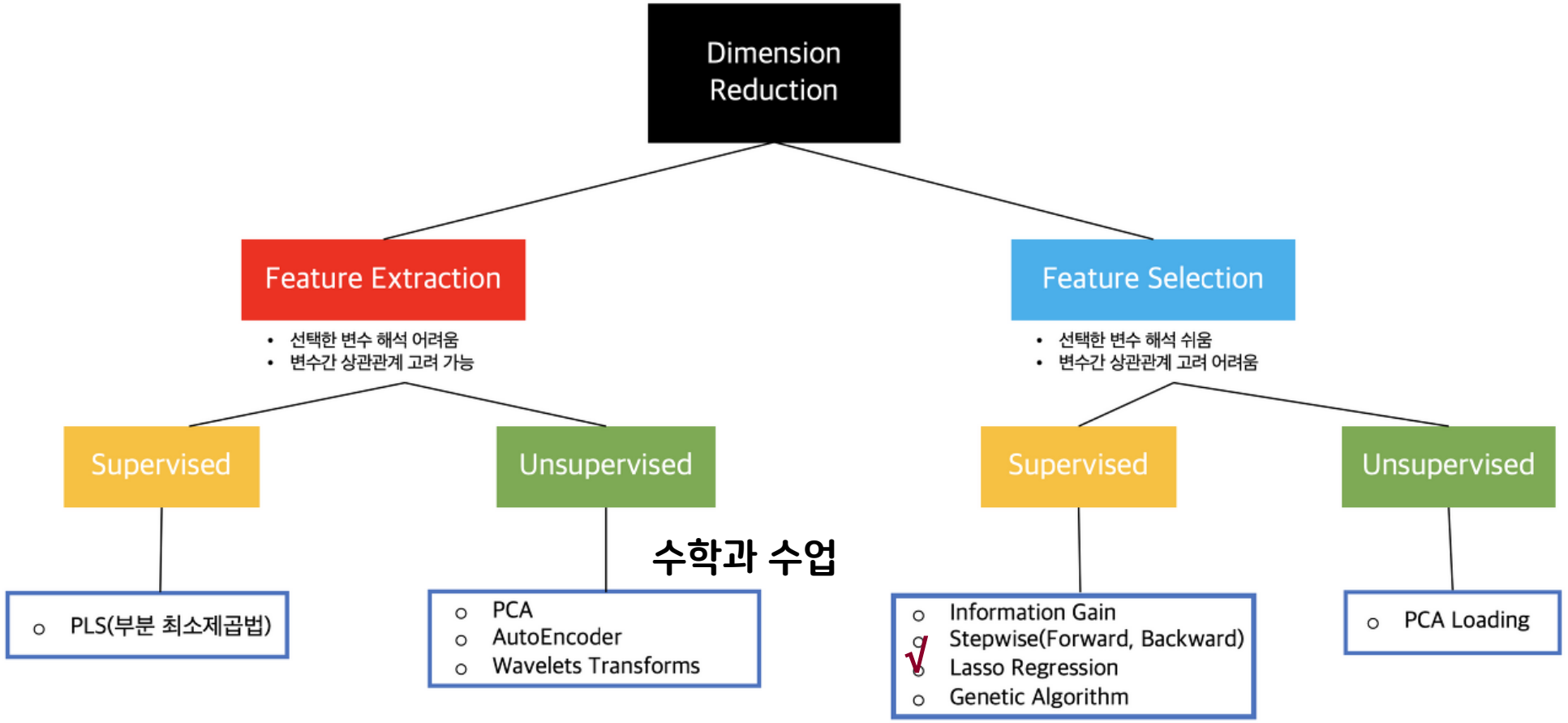
Multicollinearity [다중공선성]

- BUT!!

- Multicollinearity를 제거하는 것이 좋은 것인지, 그렇지 않은지는 연구적으로 정확히 밝혀진 바가 없다
- 실제로 여러 실험을 해보면 multicollinearity가 어느 정도 있는 상황에서 예측 결과가 더 좋게 나오기도 한다 (특히 어려운 task를 하는 경우. 실제 데이터에서 multicollinearity가 어느 정도 있는 것은 당연한 것이 아닐까?)

- 쉬운 task일수록 multicollinearity 제거가 중요(하다고 알려져 있다).
- 현업에서 multicollinearity를 민감하게 처리한다고 한다

Feature selection/extraction



Ridge / Lasso / Elastic Net Regression

- (딥러닝: Loss 계산 시 term을 추가해서 규제. 단, 테스트 데이터셋에 대해서
  성능을 평가하거나 예측할 때에는 사용 x)

- Regularization: Bias를 조금 높이는 대신, Variance를 낮춘다
- 의의: 꼭 쓸모없는 변수를 줄이는 것이 아니라, 단순하게 다중공선성을 해결하고자 하는 방법

- 계산된 회귀 계수가 작은 것부터 drop (얼마나 없앨 것인지는 선택!)

Ridge / Lasso / Elastic Net Regression

1. Ridge Regression

```
rr = Ridge(alpha = 0.01)
rr.fit(train_x,train_y)
rr.coef_
```

```
array([-3.71283678e-03,  7.37570775e-03,  3.54973975e-01, -5.28579506e-02,
        7.83404224e-02,  4.12823466e-03,  3.62504712e-02,  3.27385112e-03,
        1.73105480e-06, -1.91297381e-02, -8.77388670e-02])
```

Ridge / Lasso / Elastic Net Regression

1. Ridge Regression (L2-norm)

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \alpha \frac{1}{2} \sum_{i=1}^{n} \theta_i^2$$

2. Lasso Regression (L1-norm)

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + \alpha \sum_{i=1}^{n} |\theta_i|$$

3. Elastic Net Regression (1, 2 합친 것)

$$J(\boldsymbol{\theta}) = \mathrm{MSE}(\boldsymbol{\theta}) + r\alpha \sum_{i=1}^{n} |\theta_i| + \frac{1-r}{2} \alpha \sum_{i=1}^{n} \theta_i^2$$

Ridge / Lasso / Elastic Net Regression

- 무엇을 언제 사용해야 할까?

우선, Ridge를 기본적으로 사용

Lasso는 변수의 개수가 샘플 수 (n)보다 많으면 최대 n개의 변수를 선택하고,
여러 변수간 강한 상관관계를 띄면 임의의 변수 하나를 선택
즉, 다중공선성이 발생할 경우 덜 중요한 변수의 coefficient를 0으로 만들어버려 극단적으로 처리하게 됨.

이 경우에는 Elastic Net으로 적당히 확인.

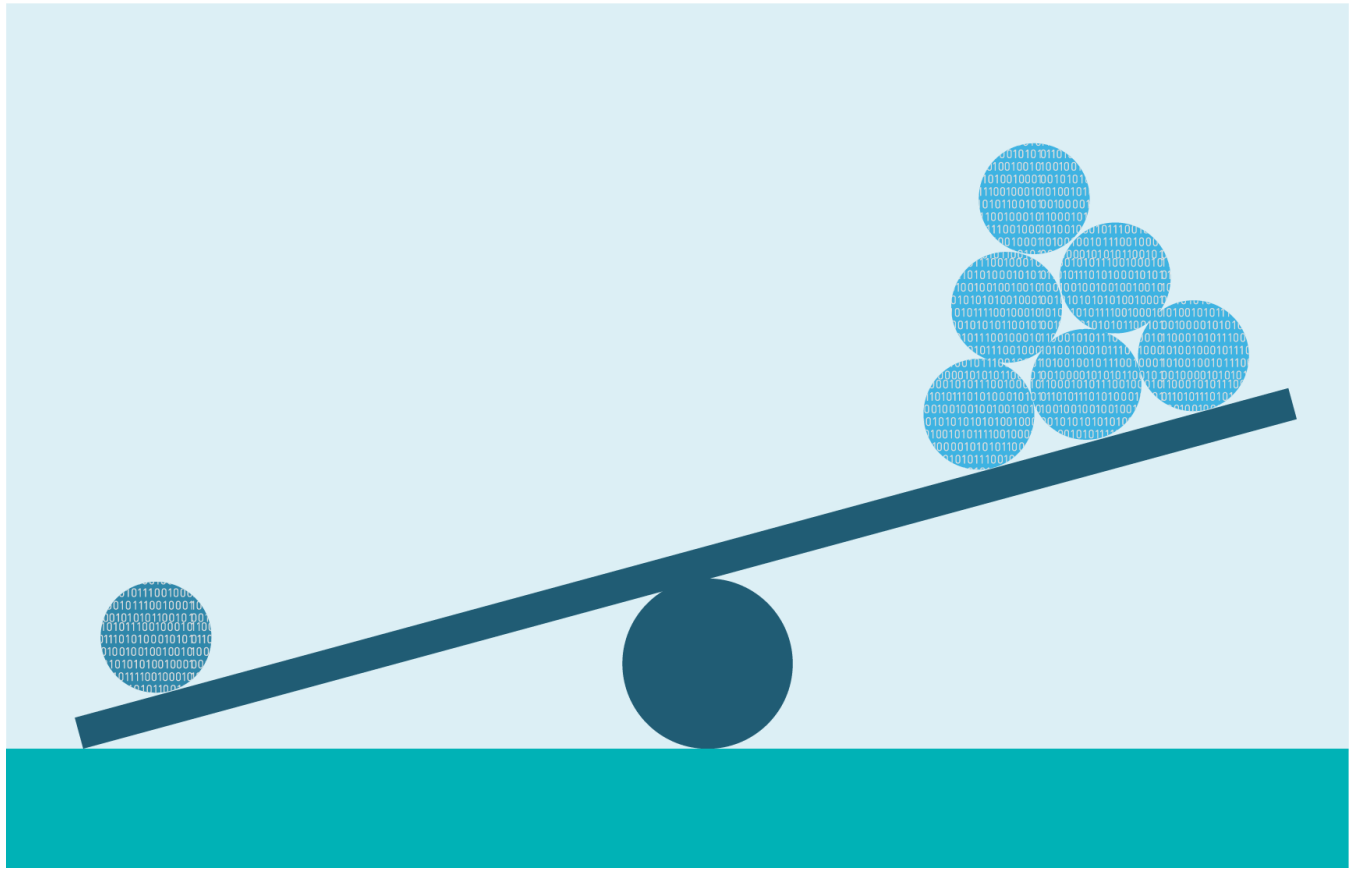
Ridge / Lasso / Elastic Net Regression

- BUT!!

- 최근 머신러닝 연구 트렌드: impact가 작은 feature에서도 배울 점이 있다
- Impact가 작은 feature도 모두 학습에 포함시키는 것이 좋다!!
- Weak learner를 앙상블하는 방식으로 사용

- 먼저 전체 feature에 대해서 머신러닝을 적용해본 후에 (baseline),
- Ridge / Lasso / Elastic net regression은 추가적으로 성능을 높이기 위해 튜닝할 때 사용해보는 것 추천 (조금이지만 성능이 높아지는 경우가 꽤 있다)

Data Imbalance [Class Imbalance]



데이터를
균형 있게 수집!!

Data Imbalance [Class Imbalance]

- BUT!!

- Class Imbalance는 당연한 결과일 수 있음.

- e.g. 공장에서 양품과 불량품 관련 데이터의 개수가 비슷하다면?
-> 그 공장은 이미 문을 닫았을 것.

- e.g. 병원에서 골절 판독용 X-ray를 찍을 때 정상/비정상 군의 샘플 수가 비슷할 수 없음. 다른 질환으로 X-ray를 찍는 환자도 많기 때문에 정상 군 샘플 수가 훨씬 많음.

=> <u>전처리 차원에서의 해결 필요</u>

Data Imbalance [Class Imbalance]

1. Class Imbalance 관련 처리를 하지 말고 모델링 해보기

- Class Imbalance가 항상 문제 되는 것은 아님.
- Task가 어려울 때 문제 발생. 쉬운 task는 별다른 문제 없이 해결 가능한 경우가 많음.

- 단, Accuracy 등 일부 성능 지표는 정확하지 않을 확률이 높으니 task에 맞게 바꾸어 사용할 것 (정상 : 비정상 = 9 : 1인 상황에서 모두 정상이라고 예측하고 90%의 accuracy를 달성한다면 ??!!) -> MCC, domain에서 잘 사용하는 지표…

-> task가 어려운 경우에는??

Data Imbalance [Class Imbalance]

2. Class Imbalance 관련 처리 방법

- Automl-pycaret
- fix_imbalance로 원하는 방법 선택 가능

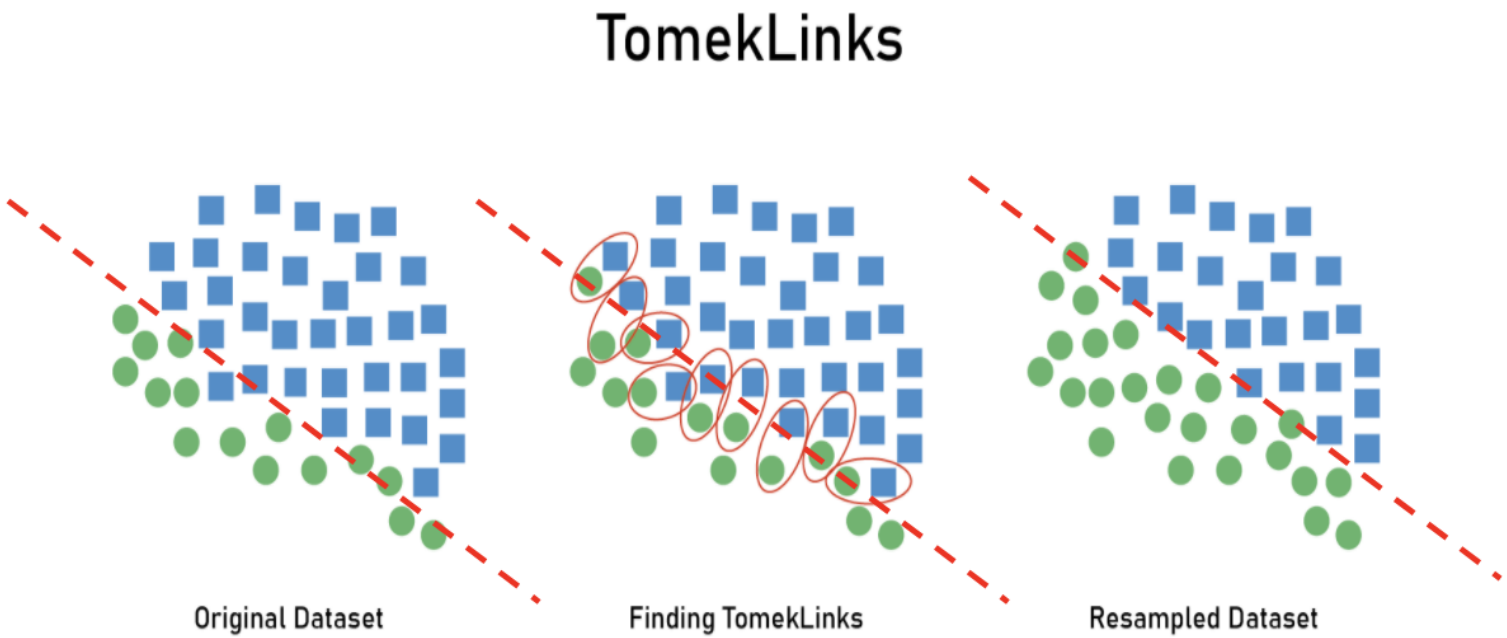
(1) Under sampling

강제로 데이터 개수 맞추기 (정보 유실 문제)

(2) Tomek link

서로 다른 클래스가 있을 때 서로 다른 클래스끼리 가장 가까운 데이터들이
토멕링크로 묶여서 토멕링크 중 분포가 높은 데이터를 제거하는 방법론
클래스를 나누는 Treshold를 분포가 높은 쪽으로 밀어 붙이는 효과

Data Imbalance [Class Imbalance]



Data Imbalance [Class Imbalance]

(3) CNN (Condensed Nearest Neighbor)

최근접인 클래스 분포 데이터를 삭제하면서 샘플

1. 분포가 작은 클래스를 S분포로 둔다
2. 분포가 큰 클래스를 랜덤으로 하나 선택한 뒤 그 데이터 위치에서 가장 가까운 데이터를 선택했을 때 S 분포에 포함 되어 있지 않은 데이터라면 제거
3. 가장 가까운 값이 S분포가 나올 때까지 2번 반복

(4) Oversampling

Real-word Issues

- Multi-center 학습

다양한 기관, 기기, 데이터 획득 포인트에서의 데이터 활용 (generality 확보)
-> 각 상황에서의 파라미터를 반영하여 전처리, 불필요한 표지 제거 (e.g. ICU mark 등)

- Noisy samples 관련

- Labeling이 어려울 때

XGBoost 1

- Jupyter Notebook
https://colab.research.google.com/drive/1huqOXFrhVXuF3TP-cHMCHHJ_Zmv2Bqo7?authuser=5#scrollTo=rMLd6TrhUA92

- https://kubig-2023-1.tistory.com/134

Reference

(1) 파이썬 머신러닝 완벽 가이드
(2) https://jaeworld.github.io/data%20science/PyCaret/
(3) https://dodonam.tistory.com/387
(4) AI 기술, 제조업에 생기를 불어넣다 (삼성전기㈜ AI Solution Lab. 조한상 Master)
(5) https://romg2.github.io/mlguide/03_%EB%A8%B8%EC%8B%A0%EB%9F%AC%EB%8B%9D-%EC%99%84%EB%B2%BD%EA%B0%80%EC%9D%B4%EB%93%9C-04.-%EB%B6%84%EB%A5%98-XGBoost/
(6) https://laoonlee.tistory.com/12
(7) https://todayisbetterthanyesterday.tistory.com/14
(8) https://shinminyong.tistory.com/34

COMING UP NEXT…!

XGBoost 2 (논문 나머지 및 세부 parameter 관련)
Hyperparameter tuning, GridSearch

Q & A