2023 Machine Learning Odyssey Pt. 2

Aug 18, 2023
Seungeun Lee

“2023 Machine Learning Odyssey”

Part 1	July 25, 2023	Tabular data, Data preprocessing, Evaluation metrics, Cross validation, imputation
Part 2	July 25, 2023 & Aug 18, 2023	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

AutoML

AutoML-pycaret

- An open-source library that automates the machine learning workflow with minimal code, handling coding, preprocessing, model selection, and parameter tuning all at once!

- https://pycaret.org/guide/

How to AutoML

If tackling a Machine Learning problem...

(1) Analyze with AutoML
(2) Select the top 3 or 5 models from AutoML
(3) Manually model the top 3 or 5 models (using Python wrappers, scikit-learn)
(4) (Optional) Apply Ensemble Techniques

How to AutoML

! Caution !

- Do not rely entirely on AutoML
- Even with the same model (e.g., XGBoost), results may vary between Python wrappers, scikit-learn, and AutoML
- Generally, it is known that AutoML < scikit-learn < Python wrapper in terms of performance
- Also, <u>the "optimal" hyperparameters found by AutoML often turn out to be suboptimal when modeled directly using Python wrappers or scikit-learn</u>

How to AutoML

! Caution !

- AutoML installations often conflict with other packages (e.g., numpy, pandas, matplotlib) -> It's recommended to install AutoML in a separate virtual environment
- Therefore, saving models directly from the AutoML package (e.g., as .model or .pkl) is not recommended
- AutoML should only serve to prioritize <u>"which models to consider"</u> and act as a guide showing the overall <u>"trend"</u> of model performance.

Multicollinearity

Problem of High Correlation Among Independent Variables in Regression Analysis

- Independent variables should only relate to the dependent variable, not to each other.
- If relationships form among independent variables, it can lead to multicollinearity, resulting in inaccurate regression outcomes.

< **Guidelines for Handling Multicollinearity** >

- **VIF ≥ 10**: Remove (recommended in theory)
- If **70% or more features have VIF ≥ 10**: Remove approximately 30-50% of features with the highest VIF values

(Not a strict theory-based rule, just an empirical guide)

Multicollinearity

VIF (Variance Inflation Factors)

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

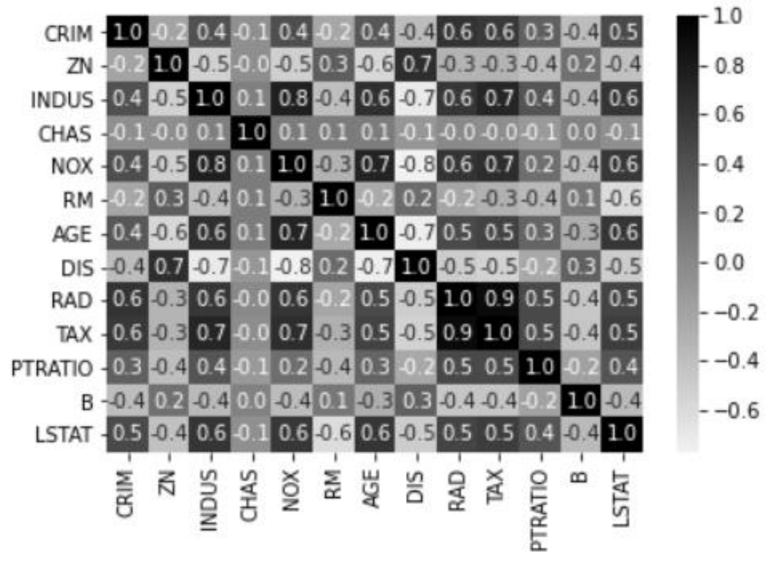
$$VIF_i > 10 \Leftrightarrow \frac{1}{1 - r_i} > 10$$

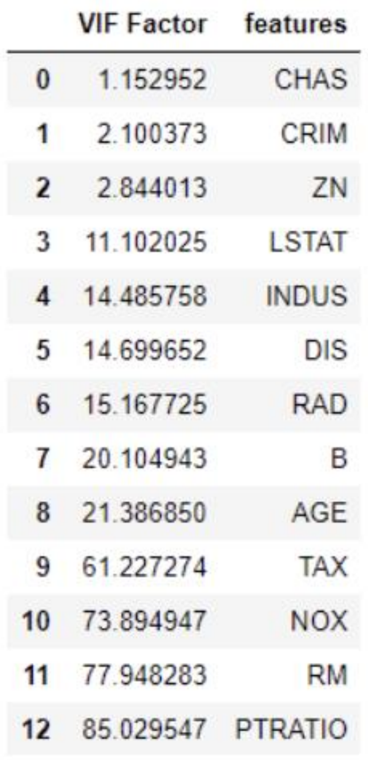
$$1 > 10 - 10r_i$$

$$r_i > 0.9$$

- This means that even without the *i*th independent variable, the remaining variables explain at least 90% of the dependent variable (y)
- In other words, *i*th independent variable is unnecessary, as *y* can be adequately explained without it

Multicollinearity





	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

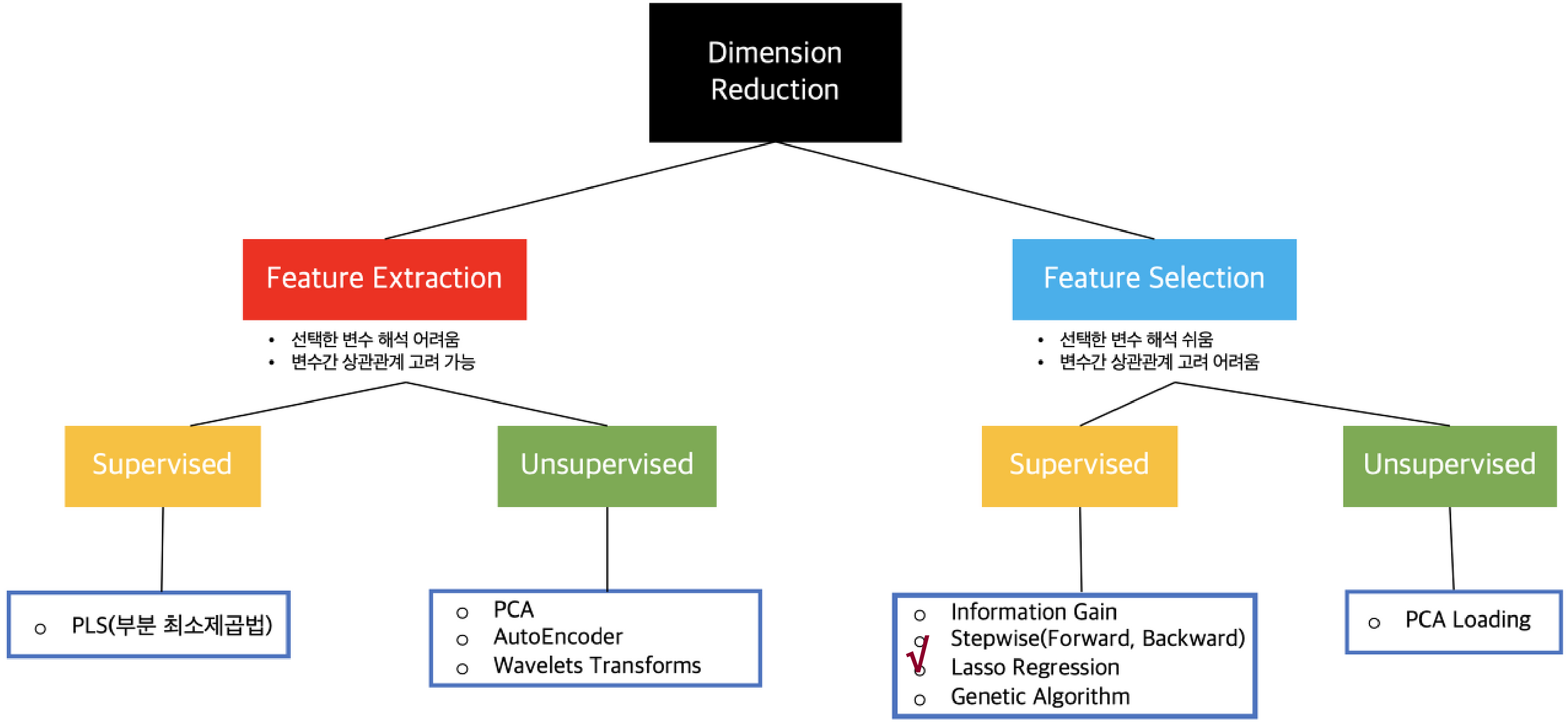
Can also be analyzed with …
- Scatter plot
- Pearson correlation coefficient

Multicollinearity

!! BUT !!

- Whether removing multicollinearity is beneficial or not remains uncertain in research.
- In practice, predictions sometimes perform better with a degree of multicollinearity, especially in complex tasks. It may even be natural for real-world data to exhibit some multicollinearity.
- For simpler tasks, removing multicollinearity is generally considered important. In industry, handling multicollinearity is treated with caution.

Feature selection/extraction



Ridge / Lasso / Elastic Net Regression

- Deep Learning: Add a regularization term during loss calculation, but do **not** use it for performance evaluation or prediction on the test dataset

- **Regularization**: Lowers variance at the cost of a slight increase in bias. Significance: A method to address multicollinearity, not merely to reduce unnecessary variables

- Drop variables with the smallest calculated regression coefficients (degree of removal is up to you!)

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 (A combination of 1 and 2)

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

Ridge / Lasso / Elastic Net Regression

When to Use What?

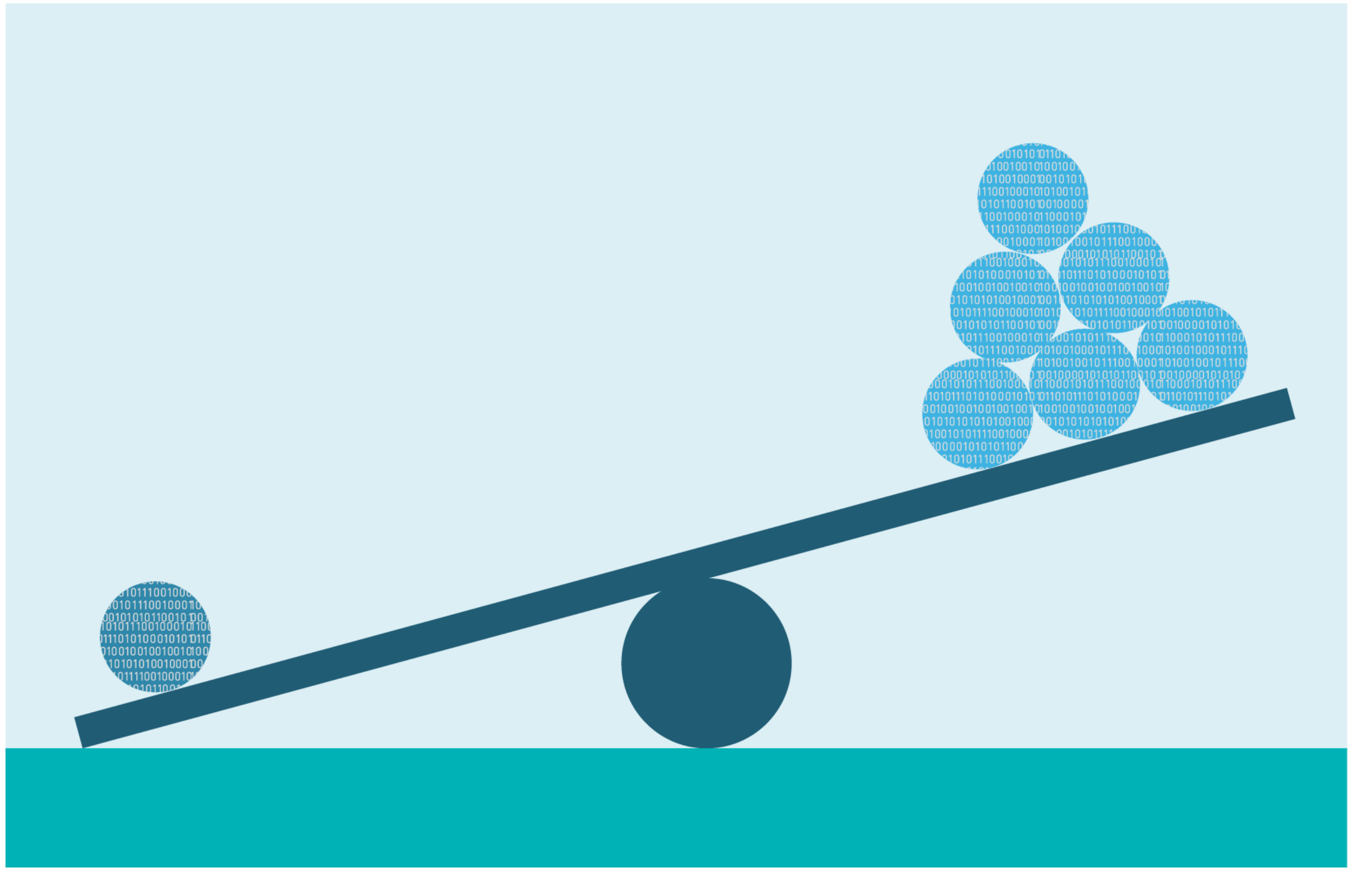
- Start with **Ridge** as a default.
- **Lasso**: If the number of variables exceeds the sample size (n), it selects up to n variables. When variables are strongly correlated, it arbitrarily selects one, setting coefficients of less important variables to zero, handling multicollinearity in an extreme manner.
- In such cases, **Elastic Net** offers a balanced alternative for checking results.

Ridge / Lasso / Elastic Net Regression

!! BUT !!

- Recent machine learning research trends suggest there is value in even low-impact features. It’s beneficial to include all features in the learning process!
- Using an ensemble of weak learners can be effective.
- First, apply machine learning to the full set of features as a baseline, then consider **Ridge / Lasso / Elastic Net** regression as additional tuning options to potentially improve performance (often yielding a small but meaningful increase).

Data Imbalance [Class Imbalance]



Collect data in a balanced manner!!

Data Imbalance [Class Imbalance]

!! BUT !!

- **Class imbalance can be a natural outcome.**

- For example, <u>if the number of good and defective products in a factory is roughly equal, that factory would likely be out of business.</u>

- In a hospital, when taking X-rays for fracture assessments, the sample sizes for normal and abnormal cases are unlikely to be equal. <u>There are often many more patients being X-rayed for other conditions, leading to a significantly larger sample size for the normal group.</u>

- Thus, addressing this issue requires preprocessing solutions.

Data Imbalance [Class Imbalance]

1. **Try modeling without addressing Class Imbalance**

- Class imbalance isn't always problematic. Issues typically arise with difficult tasks, while easier tasks can often be solved without significant issues.
- However, certain performance metrics, like accuracy, may be misleading. For example, in a scenario with a normal-to-abnormal ratio of 9:1, predicting all cases as normal would yield an accuracy of 90%.
- Instead, consider using metrics like **MCC** (Matthews Correlation Coefficient) or other domain-relevant indicators.

-> **What to do in cases of difficult tasks?**

Data Imbalance [Class Imbalance]

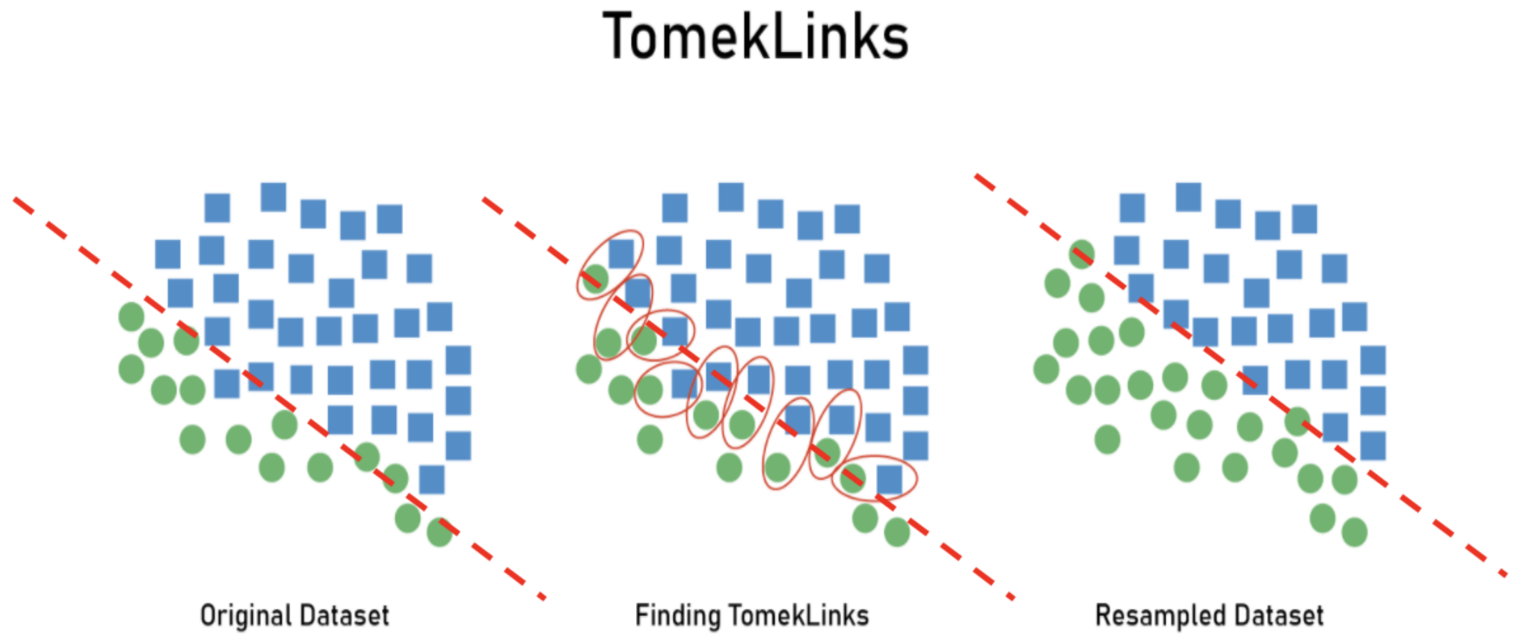
2. Methods for Addressing Class Imbalance

- **Automl-pycaret**: Use the `fix_imbalance` function to choose your desired method for handling class imbalance.

- **Under Sampling**: Forcefully equalize the number of samples, which may lead to information loss.

- **Tomek Link**: This method groups the closest data points of different classes into what are called Tomek links. It removes data points in the links that are heavily concentrated, effectively pushing the threshold for class separation toward the higher-density side.

Data Imbalance [Class Imbalance]



Data Imbalance [Class Imbalance]

3. CNN (Condensed Nearest Neighbor, not the convolution thing!)

- This method samples by removing data points from the nearest class distribution.

(1) Designate the class with a small distribution as **S distribution**.

(2) Randomly select one data point from the class with a large distribution, then identify the closest data points. If those points are not part of the **S distribution**, remove them.

(3) Repeat step 2 until the closest points belong to the **S distribution**.

4. Oversampling

- This technique involves increasing the number of samples in the minority class to balance the class distribution.

Real-word Issues

- **Multi-center Learning**

Utilization of data from various institutions, devices, and data acquisition points (ensuring generality)
-> Reflect parameters specific to each situation during preprocessing and remove unnecessary labels (e.g., ICU (Intensive Care Unit) mark of medical dataset, etc.)

- **How to deal with Noisy samples**

- **When we have both labeled and unlabeled dataset**

Q & A