

30기 겨울방학 세미나

1팀

김민우

김진혁

변석주

채희지

INDEX

1. 데이터 확인 및 EDA
2. 데이터 전처리
3. 모델링
4. 최종 모델
5. 의의 및 한계

1

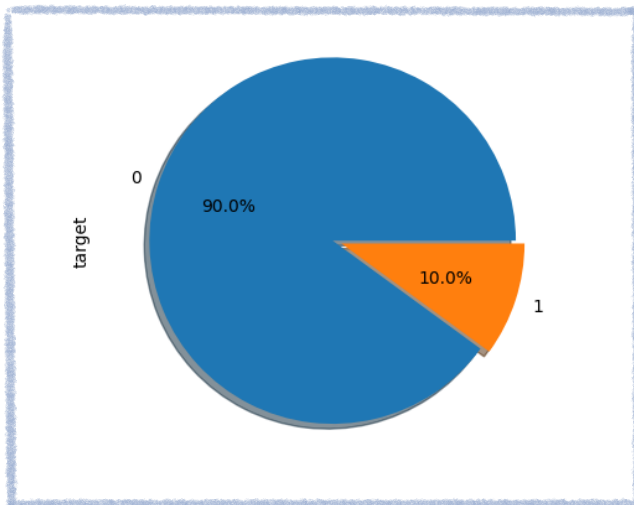
데이터 확인 및 EDA

1

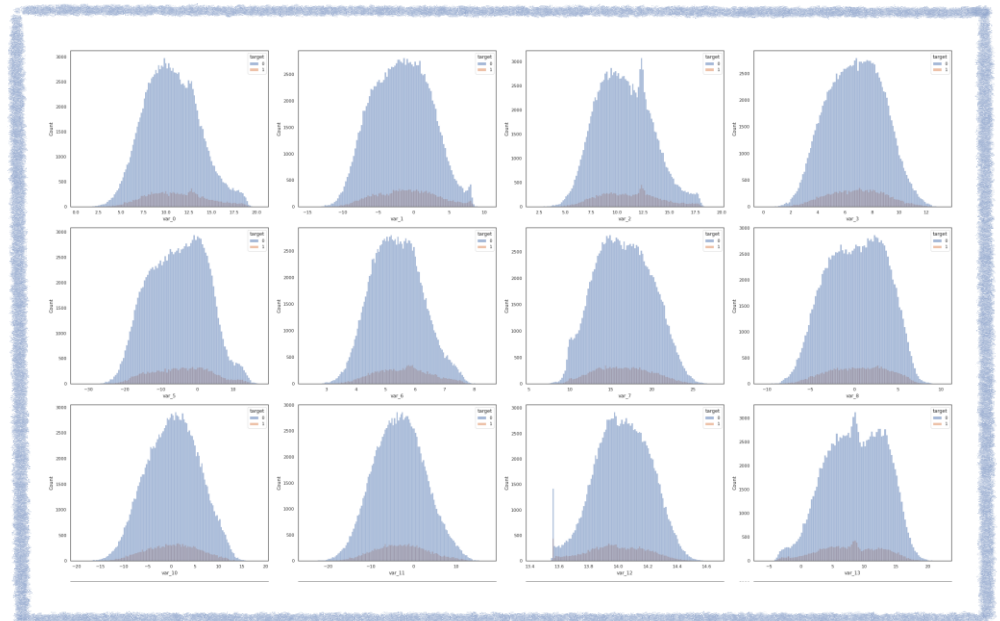
데이터 확인 및 EDA

데이터 확인 - 종속변수 비율, 클래스 불균형

종속변수 비율



클래스 불균형



종속 변수 비율이 9:1으로, 클래스 불균형이 존재함을 확인

1

데이터 확인 및 EDA

데이터 확인

	ID_code	target	var_0	var_1	var_2	var_3	var_4	var_5	var_6	var_7	...	var_190
0	train_1	0	15.4140	-2.1016	10.4773	4.8941	12.6506	-3.7205	5.1426	17.7048	...	-2.8810
1	train_2	0	12.3576	-8.1666	11.7785	2.8869	12.3183	-6.9847	4.2671	9.6710	...	0.2397
2	train_3	0	9.4142	-8.6132	7.2196	3.2496	10.6550	-3.3245	5.1010	18.5389	...	8.1638

3 rows × 202 columns

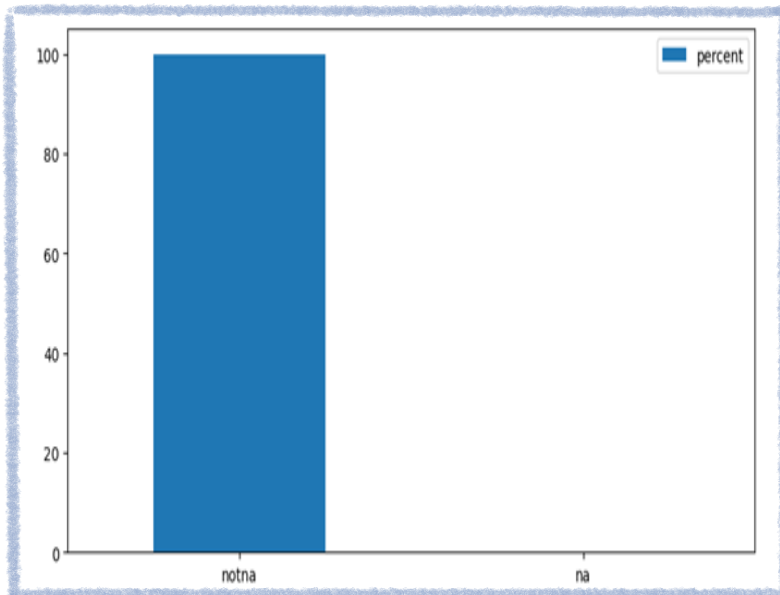
- 전체 변수가 수치형이고, masked data임을 확인.
- 독립 변수가 200개로 나타나 차원 축소를 고려해봐야 함.

1

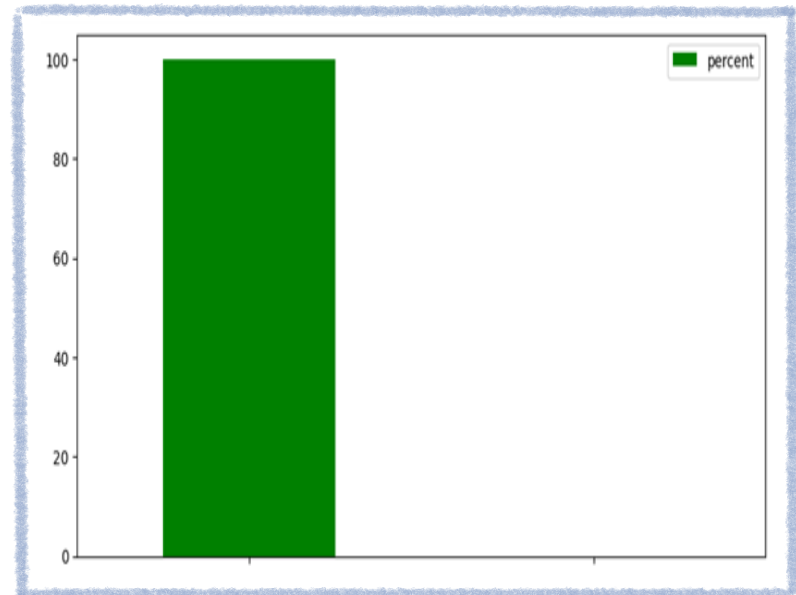
데이터 확인 및 EDA

데이터 시각화 - 결측값 / 중복값 비율

결측값



중복값

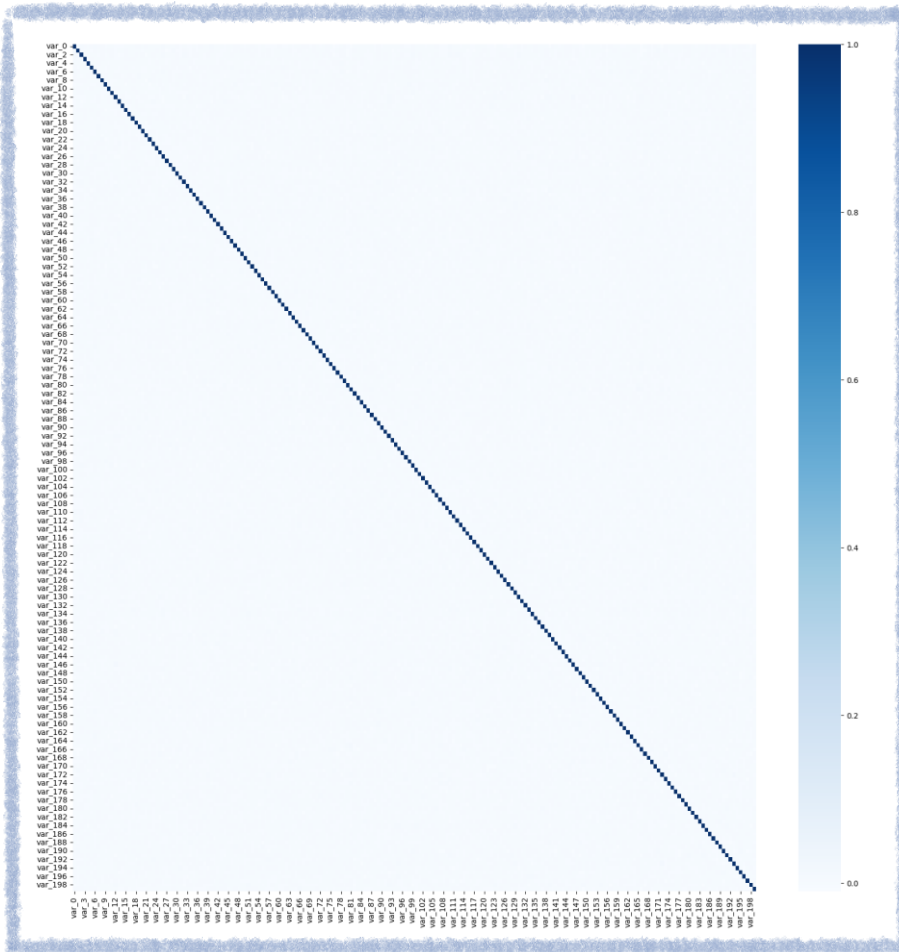


Train data에 결측값, 중복값 모두 존재하지 않음을 확인!

1

데이터 확인 및 EDA

데이터 시각화 - 피어슨 상관계수



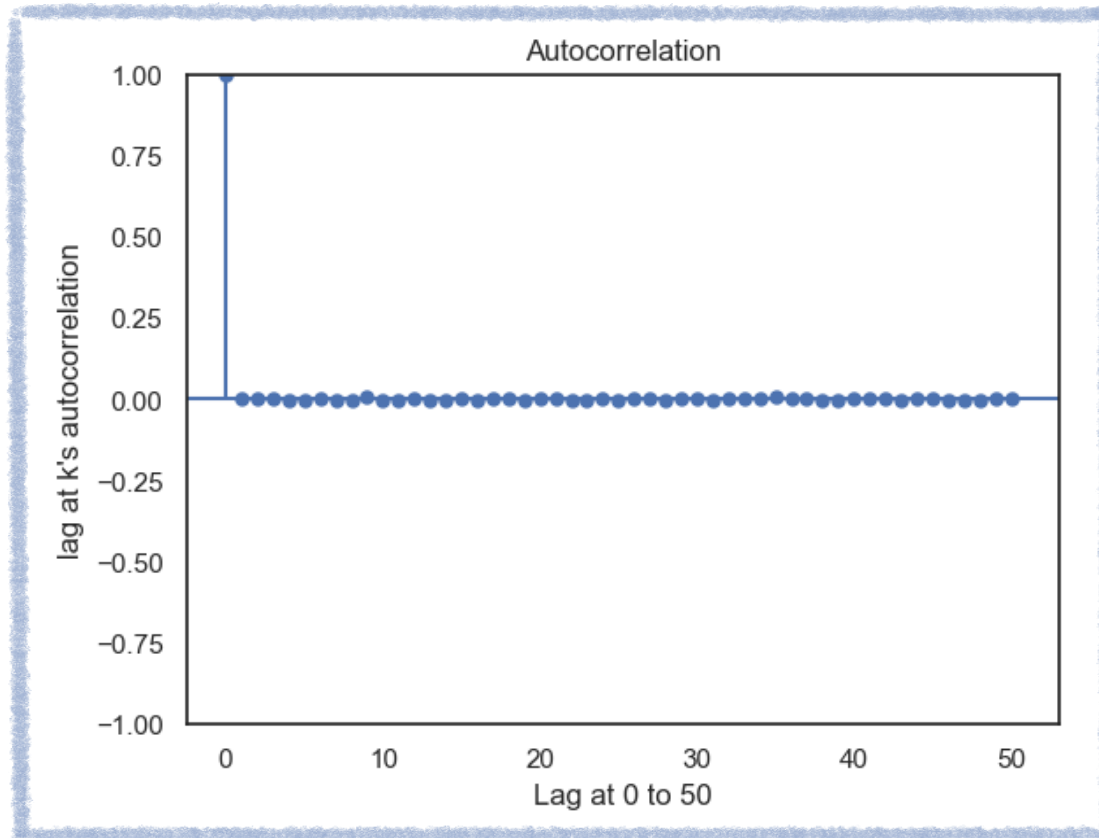
독립변수 간 상관계수가
모두 0.1을 넘지 않음

⋮

변수들 간에 유의미한 상관관계가 나타나지
않으므로 **PCA** 차원 축소 **부적절**

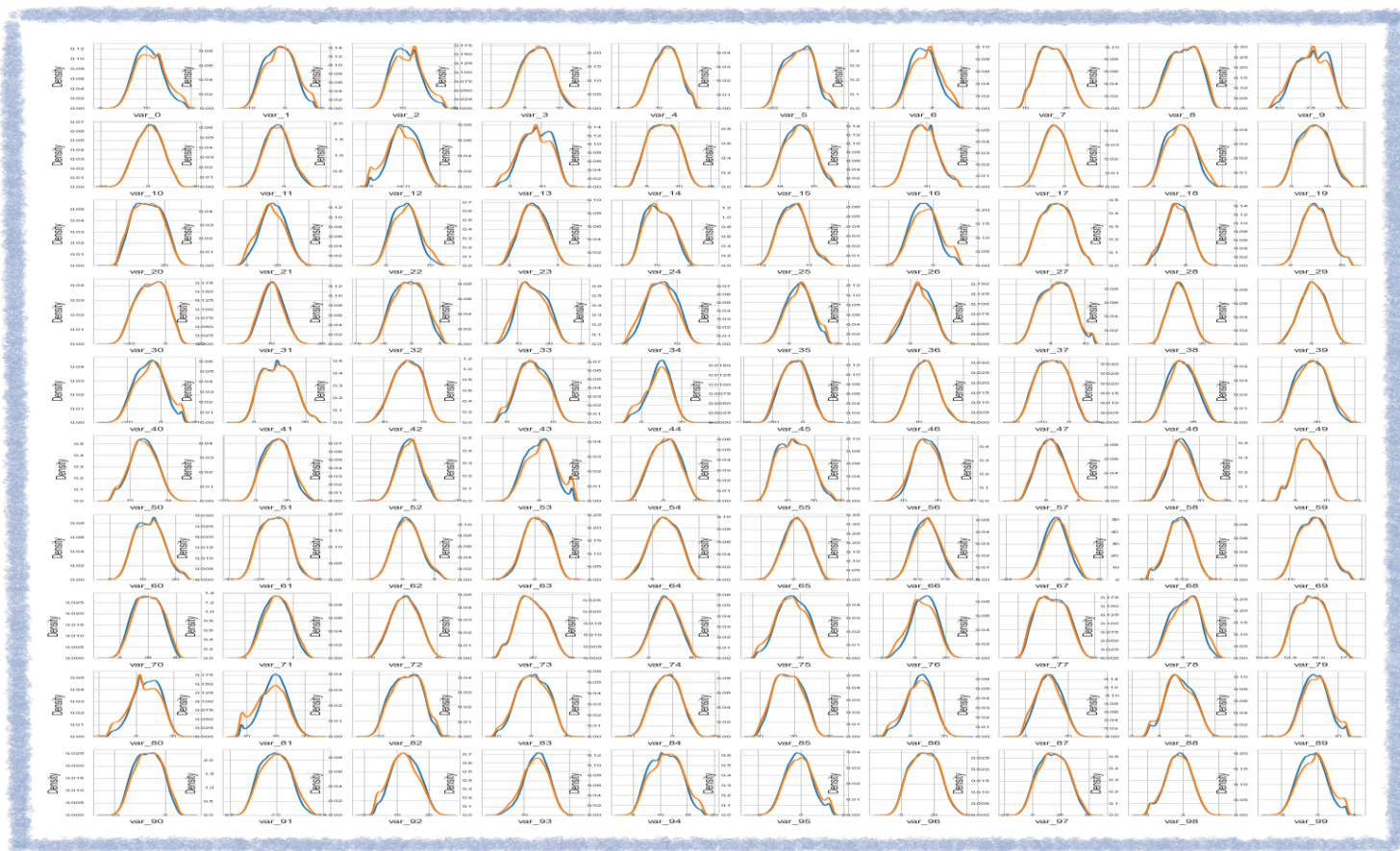
*PCA*는 데이터의 분산을 최대한 보존하면서 고차원
공간의 데이터를 저차원 공간으로 변환하는 차원 축소

데이터 시각화 - 자기상관계수



모든 변수의 자기상관계수가 0에 수렴

정규성 검정



데이터 분포가 벨 형태를 띄고 있으므로, 정규성 검정(Anderson Test 등) 진행

동질성 검정(KS TEST)

```
ho_list=hr[hr['p_value']>=0.05].iloc[2:]['Variable'].to_list()  
ho_list
```

```
['var_14',  
'var_17',  
'var_27',  
'var_29',  
'var_30',  
'var_38',  
'var_41',  
'var_46',  
'var_61',  
'var_79',  
'var_96',  
'var_100',  
'var_103',  
'var_117',  
'var_126',  
'var_129',  
'var_136',  
'var_158',  
'var_160',  
'var_161',  
'var_183',  
'var_185']
```



동질성 검정 결과, 위의 22개 변수들이 추출됨

2

데이터 전처리

변수 선택(Feature Selection)

필요성

```
✓ 0초 ▶ 1 train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 160000 entries, 0 to 159999
Columns: 202 entries, ID_code to var_199
dtypes: float64(200), int64(1), object(1)
memory usage: 246.6+ MB
```

Train data가 var_0부터 var_199까지 200개의 변수와 160,000개의
행으로 이루어진 고차원 데이터이므로, 효율성을 높이기 위해
중요한 변수만 선택하는 **변수 선택(Feature Selection)** 고려!



변수 선택(Feature Selection)

종류

Filter
Method

변수들의 상관관계가 높은,
적합한 변수 선택



계산 속도 빠름.
고차원 데이터에 **적합!**

Wrapper
Method

변수의 Subset을 Input으로
하여 최적화된 변수 집합 선택



느리지만 정확함.
고차원 데이터에 **부적합!**

Embedded
Method

변수 선택 모델링 과정
안에서 변수 선택



Filter, Wrapper 결합.
고차원 데이터에 **적합!**

변수 선택(Feature Selection)

종류

Filter
Method

변수들의 상관관계가 높은,
적합한 변수 선택



계산 속도 빠름.
고차원 데이터에 **적합!**

Wrapper
Method

변수의 Subset을 Input으로
하여 최적화된 변수 집합 선택



느리지만 정확함.
고차원 데이터에 **부적합!**

Embedded
Method

변수 선택 모델링 과정
안에서 변수 선택



Filter, Wrapper 결합.
고차원 데이터에 **적합!**

변수 선택(Feature Selection)

Variance Importance - Xgboost

	XGB_gain	XGB_cover	weight	XGB_totalgain	XGB_totalcover
3	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0
8	0.0	0.0	0.0	0.0	0.0
10	0.0	0.0	0.0	0.0	0.0
11	0.0	0.0	0.0	0.0	0.0
...
187	0.0	0.0	0.0	0.0	0.0
189	0.0	0.0	0.0	0.0	0.0
193	0.0	0.0	0.0	0.0	0.0
196	0.0	0.0	0.0	0.0	0.0
199	0.0	0.0	0.0	0.0	0.0

108 rows × 5 columns

108개의 변수 추출됨

XGB_gain, XGB_totalgain :

해당 변수가 모델 예측에 미친 영향 측정

XGB_cover, XGB_totalcover :

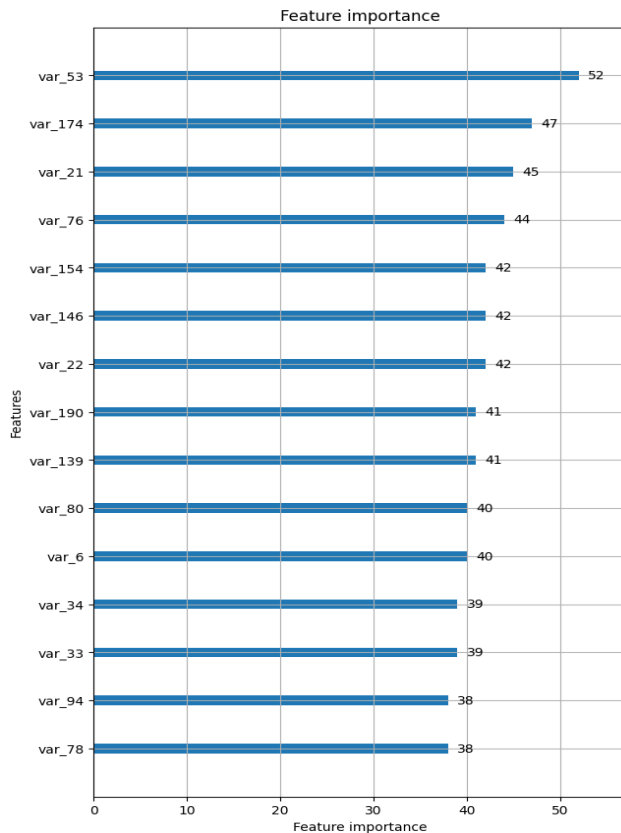
해당 변수와 관련 있는 샘플의
상대적 개수

weight :

해당 변수가 노드 분기에 사용된 횟수

변수 선택(Feature Selection)

Variable Importance - LGBM



Gain :

변수가 트리에서 사용될 때 평균
이득 계산

Split :

모든 트리에서 데이터를 분할하는 데
변수가 사용되는 횟수

변수 선택(Feature Selection)

Variable Importance - LGBM



XGBoost, LGBM 등을 통해 변수 선택한 데이터를 사용하려 했으나
성능 확인 후 **전체 데이터**를 사용하기로 결정!

Sampling

Over Sampling

- RandomOverSampler
- SMOTE
- ADASYN

Under Sampling

- RandomUnderSampler
- TomekLinks: 고차원 데이터에 부적합
- ENN: 고차원 데이터에 부적합

Sampling

Over Sampling

- RandomOverSampler
- SMOTE
- ADASYN

Under Sampling

- RandomUnderSampler
- TomekLinks: 고차원 데이터에 부적합
- ENN: 고차원 데이터에 부적합

Sampling

Over Sampling

- RandomOverSampler

- SMOTE

Combine Sampling

- RandomUnderSampler + ADASYN

- RandomUnderSampler + SMOTE

- SMOTEENN

Under Sampling

- RandomUnderSampler

- TomekLinks: 고차원 데이터에 부적합!

- CNN: 고차원 데이터에 부적합!

- OneSidedSelection: CNN + TomekLinks

Under Sampling과 Over Sampling을 합친 Combine Sampling 또한 시도!

Sampling

```
In [13]: from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_sm, y_sm)
predict = model.predict(x_test)
from sklearn.metrics import f1_score
f1 = f1_score(y_test, predict, average='macro')
print(f1)

0.7000203861001255
```

Over/Under/Combine Sampling을 통해,
Logistic Regression과 같은 간단한 모델에서는 성능 향상!



3

모델링

사용한 모델

Logistic
Regression

나이브 베이즈

LDA/QDA

GMM

SVM

Random Forest

XGBOOST

Light GBM

ADABOOST

사용한 모델

변수 선택

Raw Data / Test of Homogeneity / Feature Importance



샘플링

Under Sampling / Over Sampling / SMOTE



모델링

Logistic Regression / 나이브 베이즈 / LGBM 등

모델 조합



여러 조합을 시도한 결과!

LGBM 변수선택 + 나이브 베이즈

Raw Data + Under Sampling + LGBM

⋮

Under Sampling / Over Sampling / SMOTE



Logistic Regression / 나이브 베이즈 / LGBM 등

모델 조합



여러 조합을 시도한 결과!

LGBM 변수선택 + 나이브 베이즈

Raw Data + Under Sampling + LGBM

⋮



Raw Data가 가장 좋은 성능을 보임

Logistic Regression / 나이브 베이즈 / LGBM 등



모델 조합

성능 향상 모델

Raw Data + Logistic Regression 0.65

SMOTE + Logistic Regression **0.70**

Raw Data + LDA 0.67

SMOTE + LDA **0.71**

성능 하락 모델

Raw Data + 나이브 베이즈 0.72

SMOTE + 나이브 베이즈 **0.55**

Raw Data + LGBM 0.70

SMOTE + LGBM **0.62**

모델 조합

성능향상 모델

Linear Regression, LDA 등의 성능향상이

Raw Data를 사용한 다른 모델의 성능보다 좋지 않음

Raw Data + Logistic Regression 0.65

SMOTE + Logistic Regression 0.70

Raw Data + LDA 0.67

SMOTE + LDA 0.70

성능하락 모델

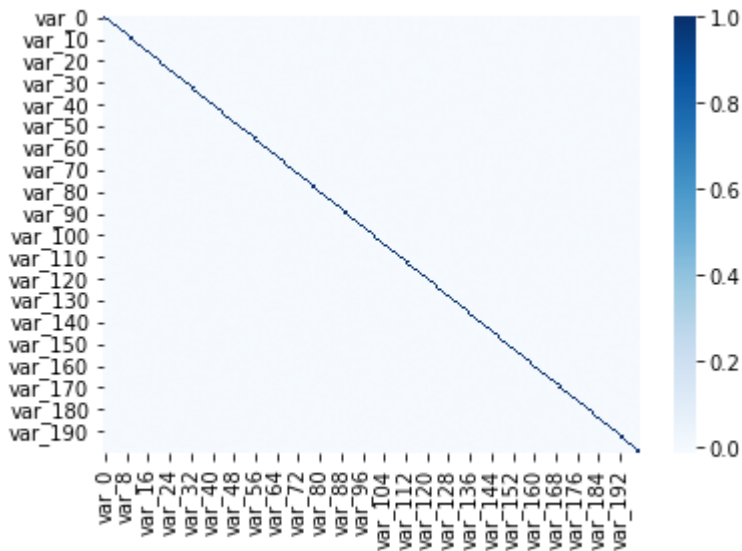
Raw Data + 나이브 베이즈 0.72

SMOTE + 나이브 베이즈 0.55

Raw Data + LGBM 0.70

따라서 변수선택 및 샘플링 X + LGBM 0.62

① Gaussian Naïve Bayes Classifier

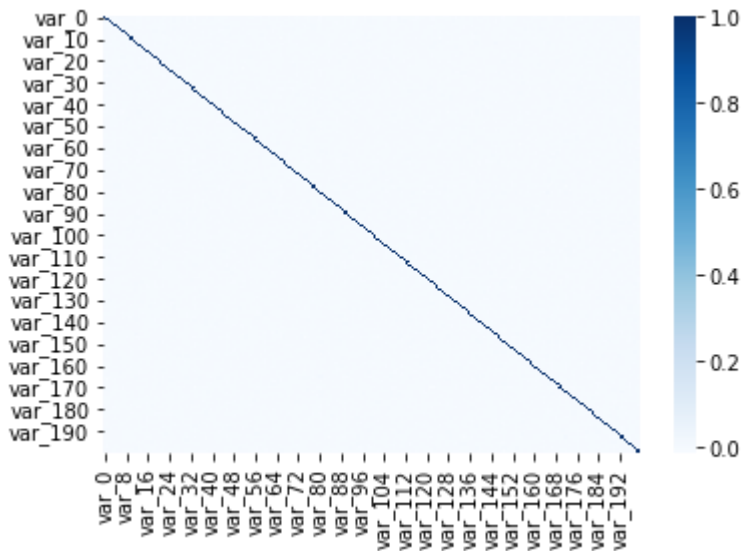


매우 낮은 상관관계



Bell-shaped Distribution

① Gaussian Naïve Bayes Classifier



변수가 정규분포를 따르며 독립이라고 가정하는

Gaussian Naïve Bayes Classifier

① Gaussian Naïve Bayes Classifier

The diagram shows the formula for the posterior probability in a Naïve Bayes classifier, with arrows pointing from descriptive labels to the corresponding parts of the equation:

$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

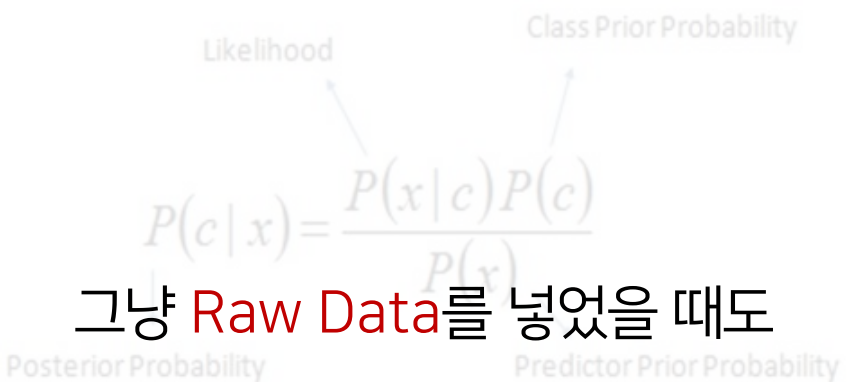
Labels and their corresponding parts in the formula:

- Likelihood: $P(x|c)$
- Class Prior Probability: $P(c)$
- Posterior Probability: $P(c|x)$
- Predictor Prior Probability: $P(x)$

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

정규분포를 가정한 표본들을 대상으로 **조건부 독립**을 나타내,
확률이 가장 높은 경우를 선택하여 분류하는 모델

① Gaussian Naïve Bayes Classifier


$$P(c|x) = \frac{P(x|c)P(c)}{P(x)}$$

그냥 Raw Data를 넣었을 때도

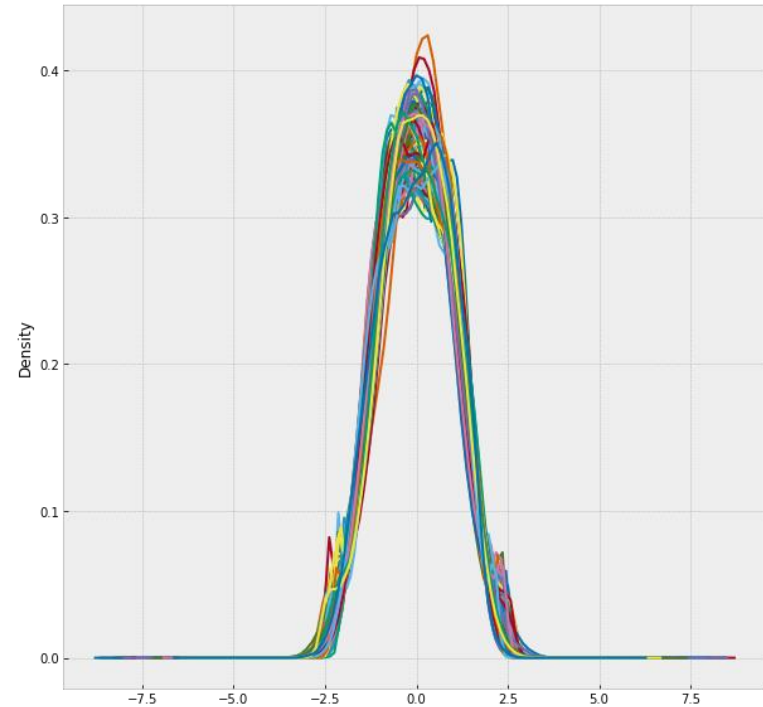
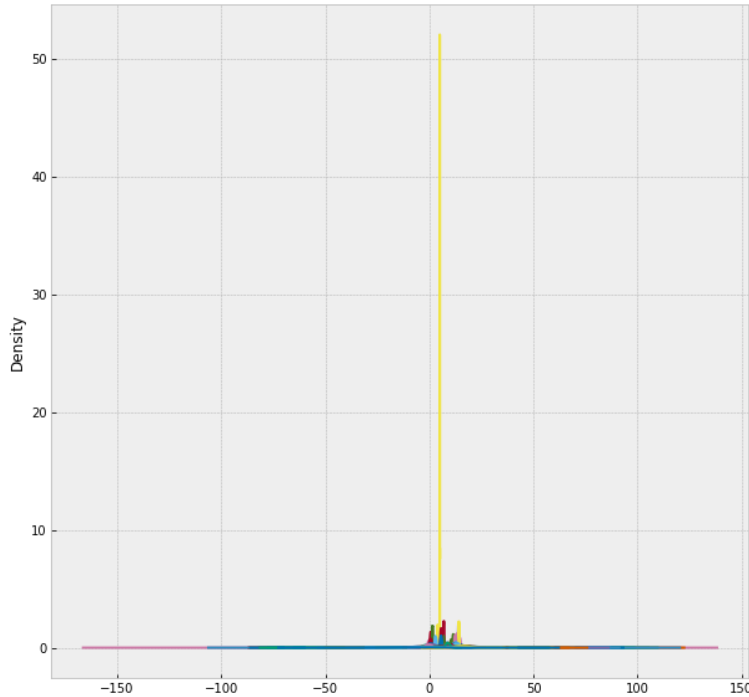
0.7149로 나쁘지 않은 성능을 보임

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

정규분포를 가정한 표본들을 대상으로 조건부 독립을 나타내,

확률이 가장 높은 경우를 선택하여 분류하는 모델

② Gaussian Mixture Model (GMM)



KDE Plot을 그렸을 때, 평균과 분산이 다른 bell-shaped 분포

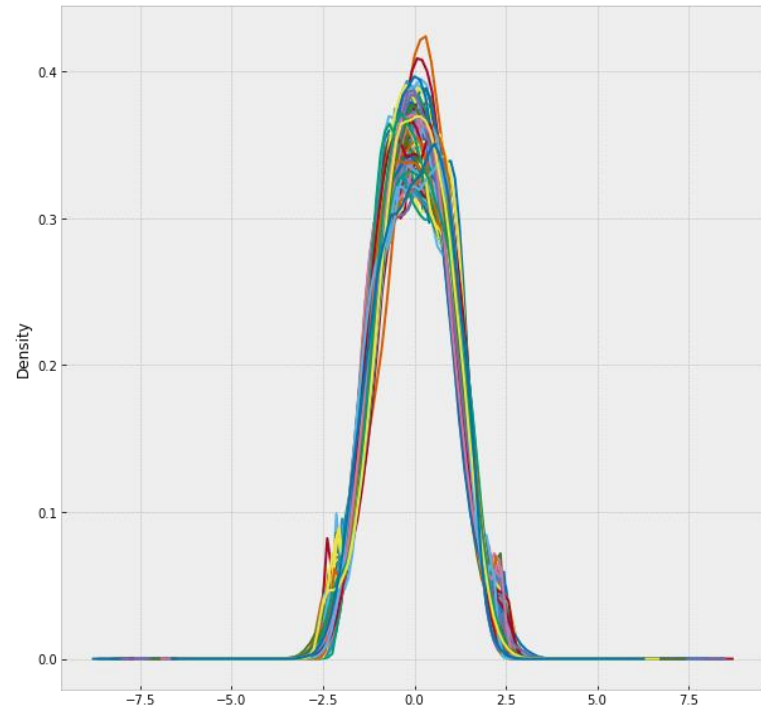
② Gaussian Mixture Model (GMM)

KDE: Kernel Density Estimator

밀도 추정 방법 중 하나로
kernel function을 사용해
부드러운 밀도 함수를 도출함



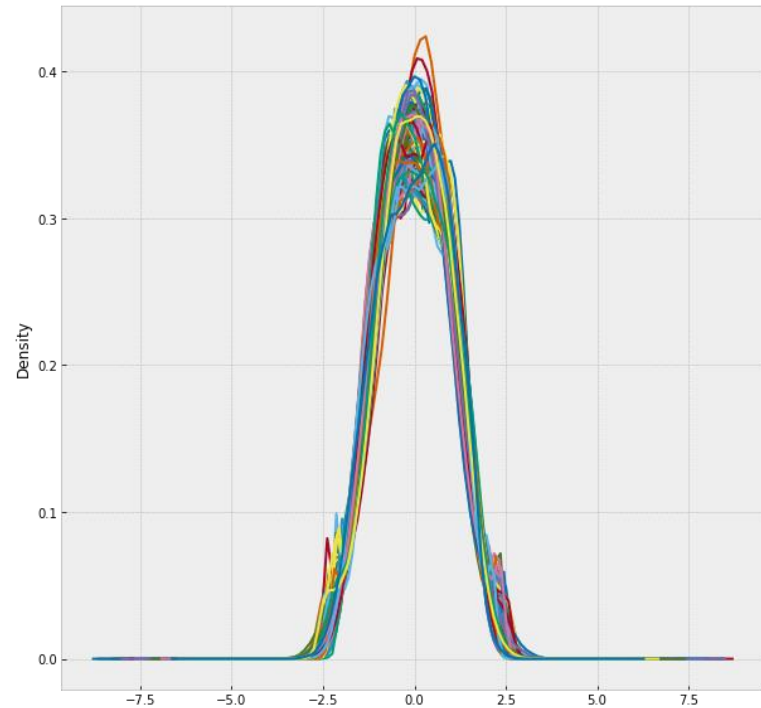
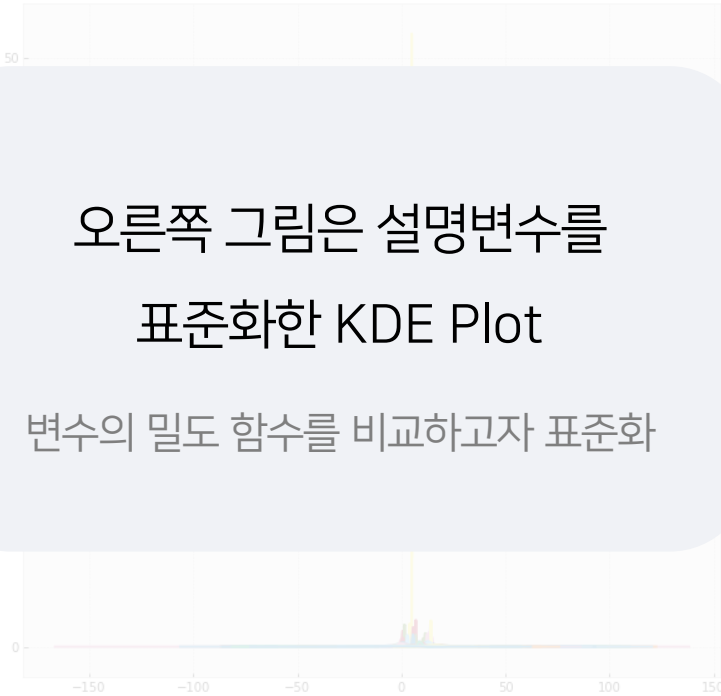
KDE Plot을 그렸을 때, 평균과 분산이 다른 bell-shaped 분포



② Gaussian Mixture Model (GMM)

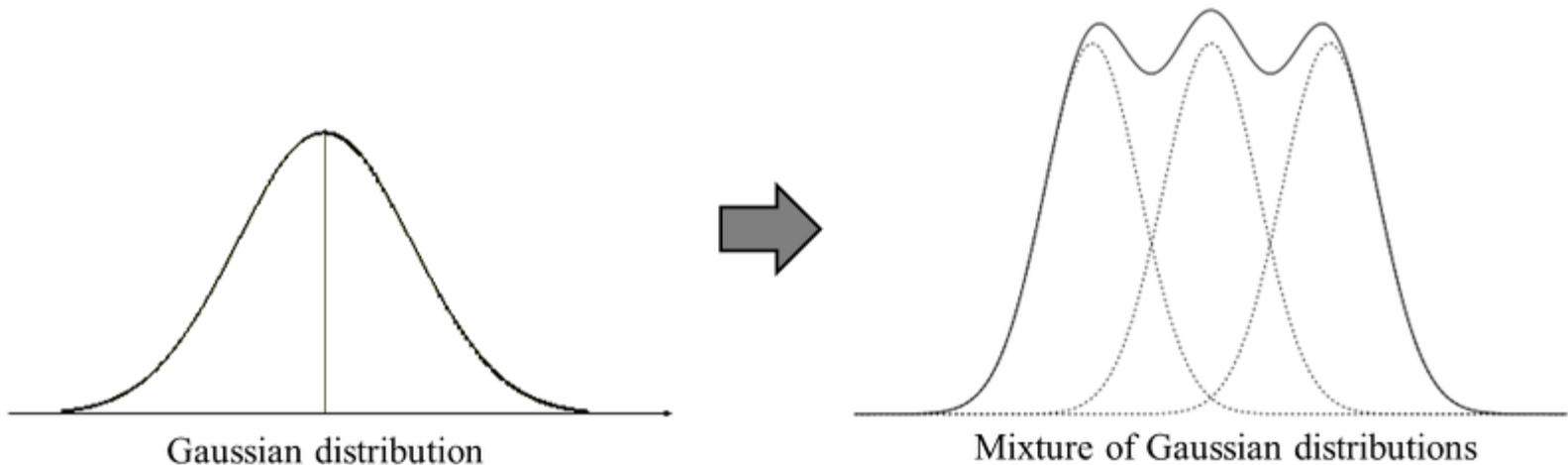
오른쪽 그림은 설명변수를
표준화한 KDE Plot

변수의 밀도 함수를 비교하고자 표준화



여러 정규분포가 혼합된 모델처럼 볼 수 있음!

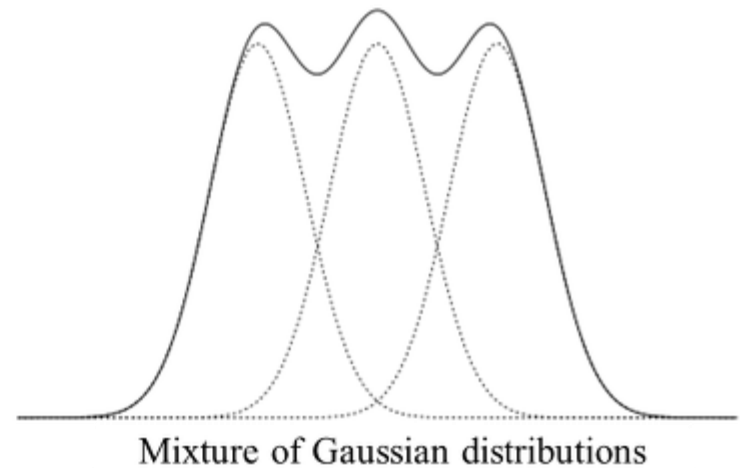
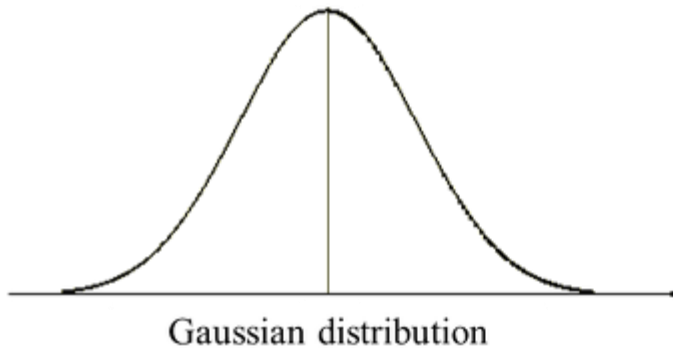
② Gaussian Mixture Model (GMM)



복잡한 형태의 확률 분포를

K개의 Gaussian Distribution을 혼합하여 표현하는 GMM 시도!

② Gaussian Mixture Model (GMM)



Stratified K-fold CV 이용
per-sample average log-likelihood를 계산해
최적 군집 개수 3개 선정!

② Gaussian Mixture Model (GMM)

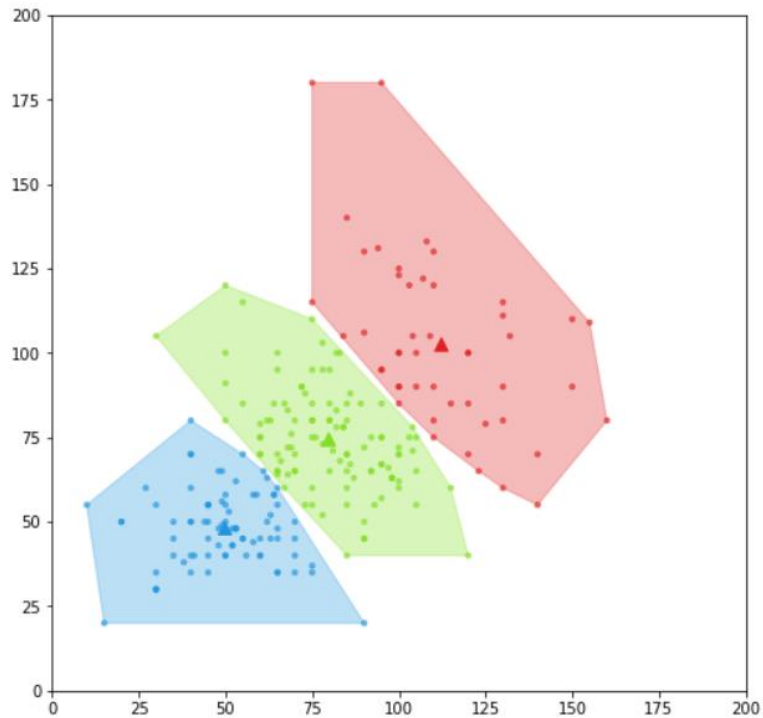


Stratified K-fold CV 이용

per-sample average log-likelihood를 계산해

최적 군집 개수 3개 선정!

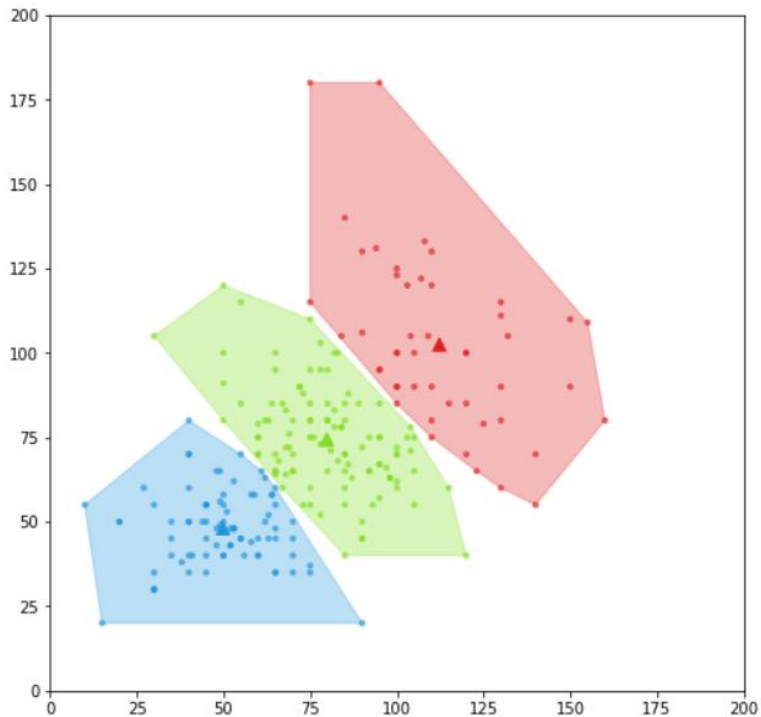
② Gaussian Mixture Model (GMM)



	0	1
Red	80	15
Green	50	10
Blue	10	20

군집 별 종속 변수 비율이 다름!

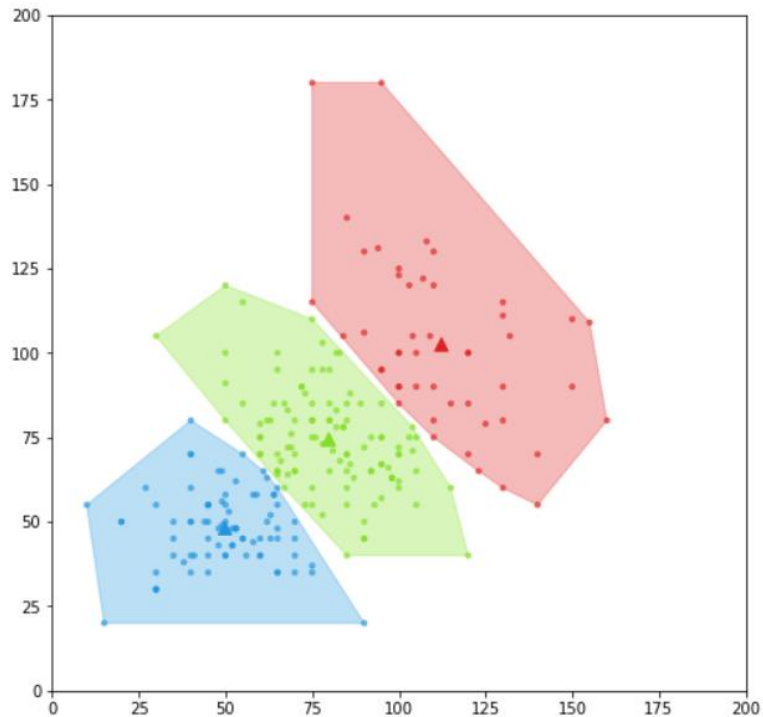
② Gaussian Mixture Model (GMM)



	0	1
Red	80	15
Green	50	10
Blue	10	20

Red와 Green 군집은
0의 비율이 훨씬 높음

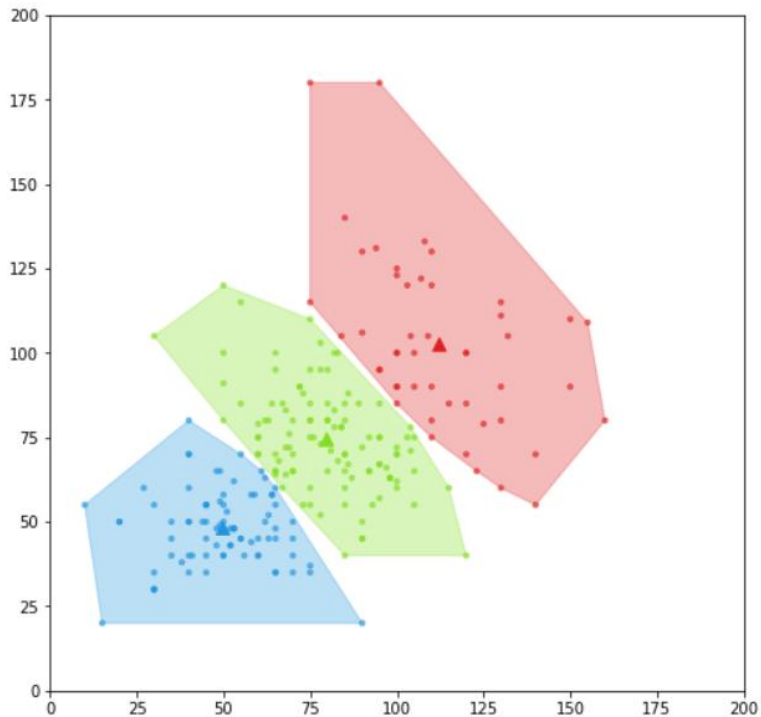
② Gaussian Mixture Model (GMM)



	0	1
Red	80	15
Green	50	10
Blue	10	20

Blue 군집은
반대로 1의 비율이 높음

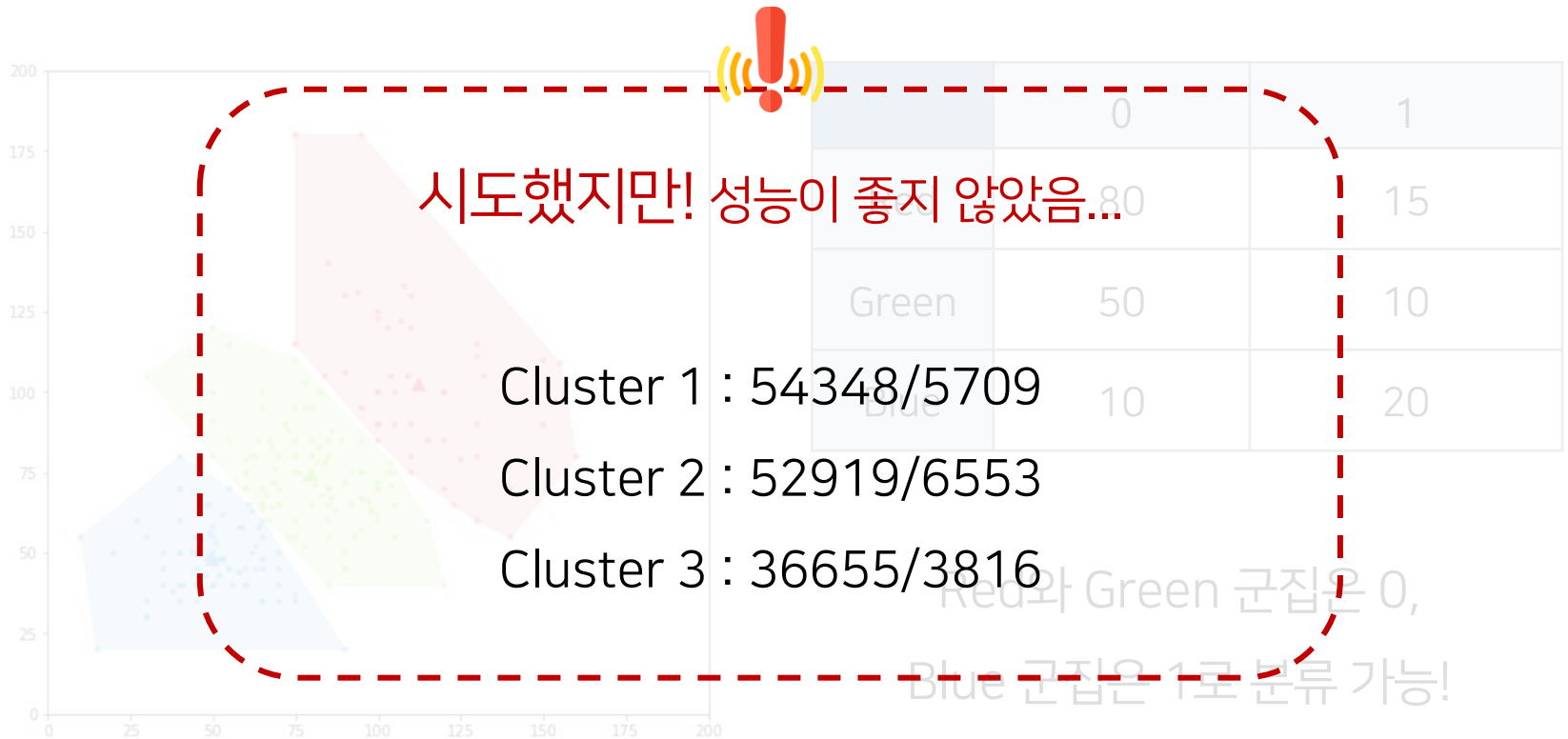
② Gaussian Mixture Model (GMM)



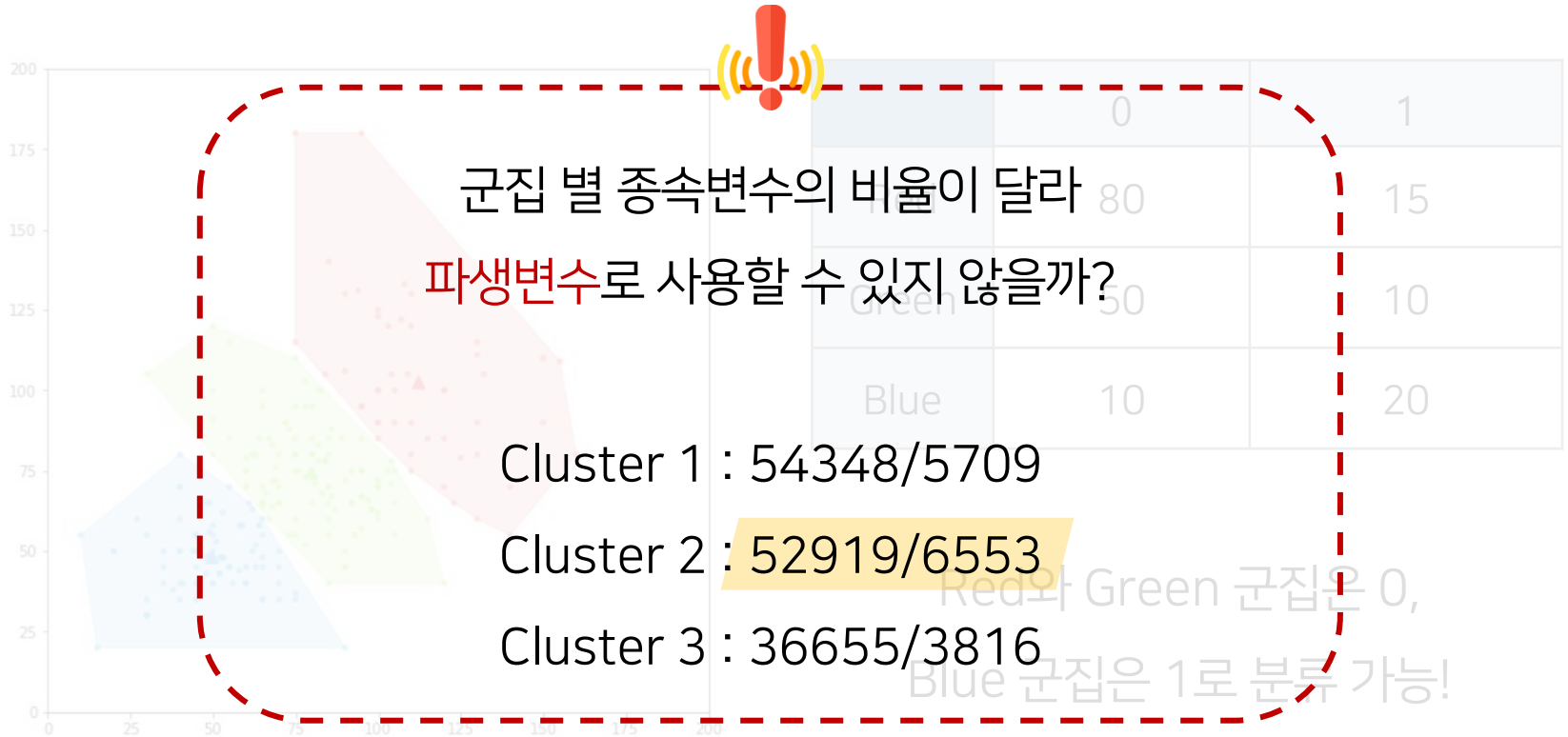
	0	1
Red	80	15
Green	50	10
Blue	10	20

Red와 Green 군집은 0,
Blue 군집은 1로 분류 가능!

② Gaussian Mixture Model (GMM)



② Gaussian Mixture Model (GMM)



② Gaussian Mixture Model (GMM)

파생변수 추가

군집 별 종속변수의 비율이 달라

파생변수로 사용할 수 있지 않을까?

원래 200개의 설명변수가 너무 많다고 생각

따라서 변수선택, 차원축소 고려했지만 성능 하락

Cluster 2 : 52919/6553

Cluster 3 : 36655/3816

② Gaussian Mixture Model (GMM)

파생변수 추가

군집 별 종속변수의 비율이 달라

파생변수로 사용할 수 있지 않을까?

GMM 클러스터를 파생변수로 추가했을 때

LGBM 성능 향상!

Cluster 2 : 52919/6553

Cluster 3 : 36655/3816

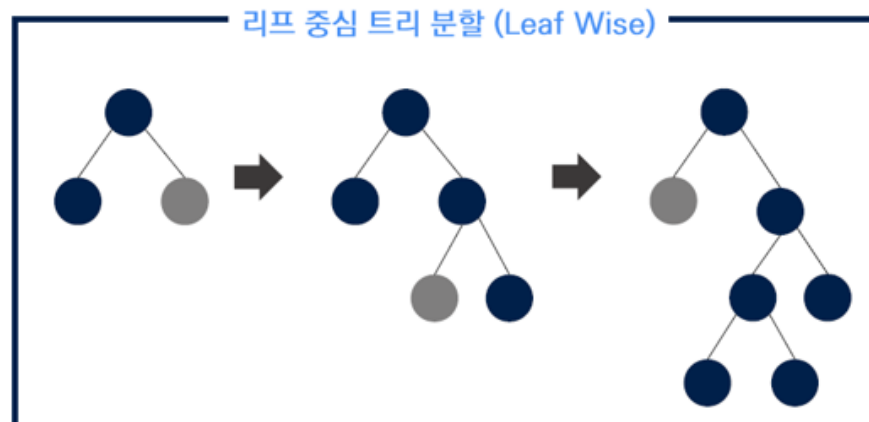
③ Light Gradient Boosting Model(LGBM)

LGBM

트리 기반 Gradient Boosting 모형

다른 트리 모형과 다르게 리프 중심으로 노드를 분할

학습 속도가 빨라 하이퍼 파라미터 튜닝 용이



③ Light Gradient Boosting Model(LGBM)

OPTUNA

하이퍼 파라미터 최적화를 도와주는 프레임 워크

베이지언 최적화 기반으로 우수한 성능



OPTUNA

③ Light Gradient Boosting Model(LGBM)

boosting_type

부스팅 타입

num_leaves

하나의 트리가 가지는 최대 리프의 개수

learning_rate

부스팅 iteration 마다 곱해지는 가중치

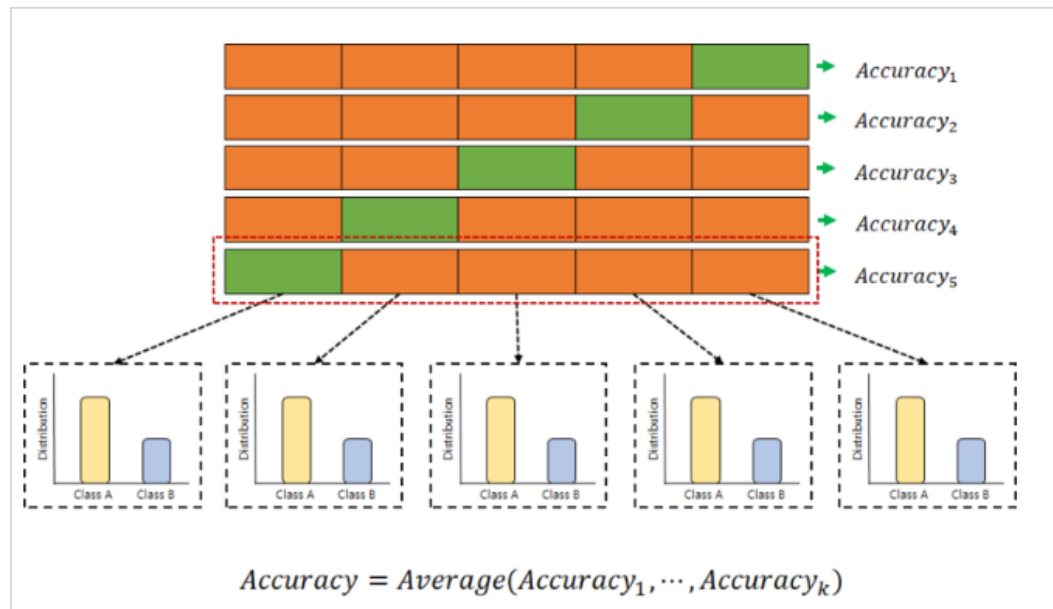
n_estimators

부스팅에 활용되는 트리의 개수

교차검증

StratifiedKFold

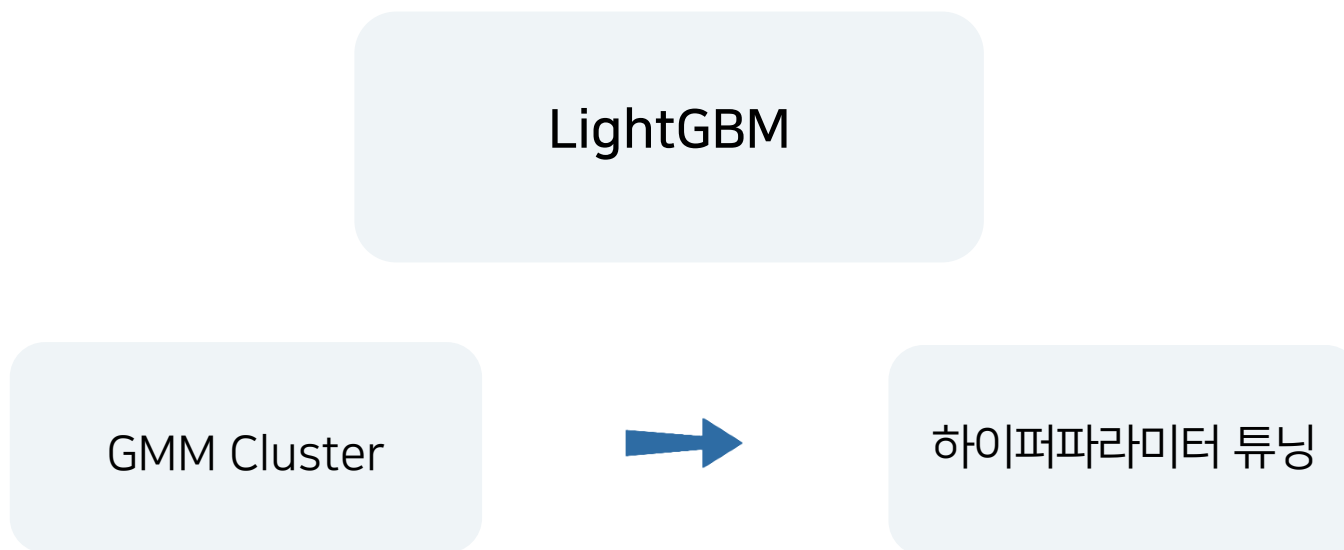
- 전체 데이터의 분포를 고려하여 학습 데이터셋과 검증 데이터셋을 분배
 - 불균형한 데이터를 사용하는 모델 성능을 측정하는데 용이



4

최종 선정 모델

최종 모델



하이퍼 파라미터 튜닝

하이퍼 파라미터 최종 수치

- Boosting_type : goss
 - Objective : binary
 - Num_leaves : 11
- Learning_rate : 0.024351317319143
 - n_estimator : 3881