Business Analytics & Data Mining

Final Project

포르투갈 은행의 "정기 예금 가입 유도" 전략

[정기예금 가입 예측 모형]

2019.11.21.Thur

Team 4.이도현 이승한 이해린 정다솜

TABLE OF – CONTENTS

1. 연구 목표

포르투갈 은행의 "정기 예금 가입자" 유도 방안

'데이터 분석' 목표 :

고객들의 특성 및 신용 정보를 바탕으로, "정기예금(Term Deposit) 가입 여부" 예측!

이를 통해, 정기예금 유도에 영향을 미칠 주요한 변수들을 파악한다!

포르투갈 은행은 "어떠한 방식"으로 이들의 정기예금 가입을 유도해야 할까?

1. 연구 목표

포르투갈 은행의 "정기 예금 가입자" 유도 방안

'데이터 분석 목표'

Suggestion 1.모든 고객들을 대상으로 한 보편적 전략

고객들의 특성 및 신용 정보를 바탕으로, "정기예금(Term Deposit) 가입 여부" 예측!

이를 통해, 정기예금 유도에 영향을 미칠 주요한 변수들을 파악한다!

Suggestion 2.세분화된 고객군을 대상으로 한 선별적 전략

포르투갈 은행은 "어떠한 방식"으로 이들의 정기예금 가입을 유도해야 할까?

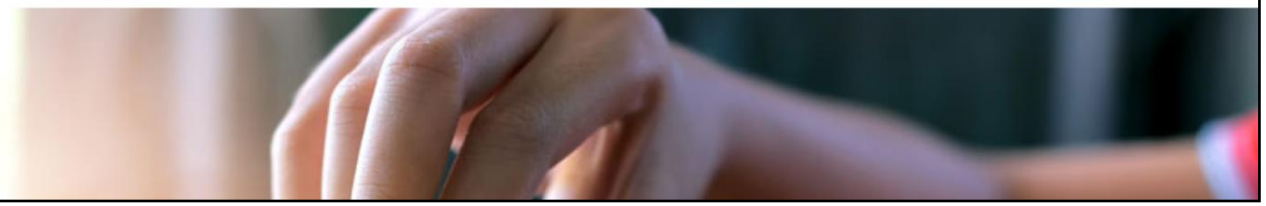
1. 연구 목표

[배경] 포르투갈의 정기예금 시장 현황은?

- 유럽에서 가장 낮은 수준의 가계 저축률!
- 지난 20년 간 저축률의 지속적 감소 (1995, 12.9% -> 2017, 5.4% -> 2018 Q2, 4.4%)
- 포르투갈의 가계 저축 문화를 바꾸려는 움직임도!

Ageas is looking to promote a shift in Portugal's savings culture

Portugal has one of the lowest household savings rates in Europe. In order to offer a more secure financial future for its citizens, the country must look to promote financial literacy and change the Portuguese savings culture



12.9%

Portuguese household savings as a percentage of available income in 1995

5.4%

Portuguese household savings as a percentage of available income in 2017

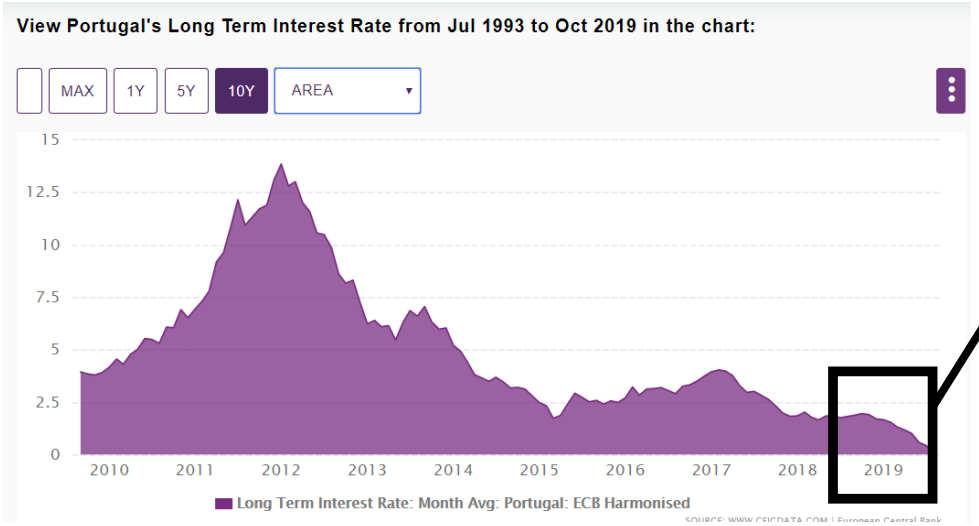
4.4%

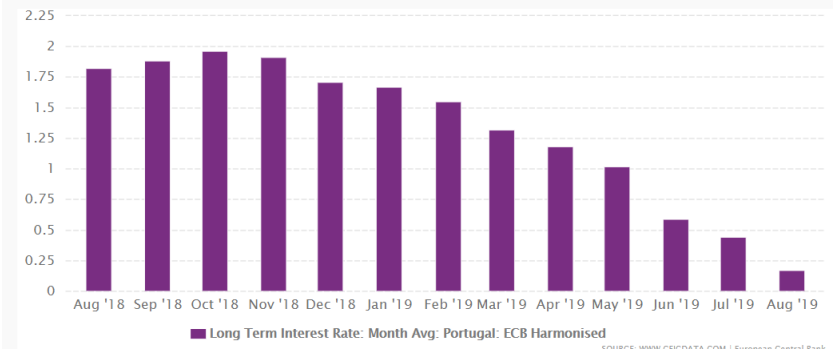
Portuguese household savings as a percentage of available income in Q2 2018

1. 연구 목표

[배경] 낮은 가계 저축률의 원인

- 1. 낮은 가계저축 incentive (낮은 이자율 + 높은 이자세율)
- 2. 국민들의 낮은 금융 이해도 (복잡한 자산에 대한 투자 거부)
- 3. 젊은 세대의 지나친 소비 > 저축 문화





이러한 현실 속에서, 어떻게 하면 각 은행은

고객들의 '정기 예금 가입'을 유도할 수 있을까?

2. 데이터 소개

Bank-full.csv

(포르투갈 고객들의 bank marketing data)

- 포르투갈 은행 기관의 고객 개인.신용 정보 및 마케팅 관련 데이터

- 17개의 변수를 가진 45211명의 데이터

- 45211명의 고객 中

  정기예금 가입자 5289명

  정기예금 미가입자 39922명

출처 : https://archive.ics.uci.edu/ml/datasets/bank+marketing

2. 데이터 소개

Bank-full.csv : 2008.05 ~ 2010.11까지 포르투갈 고객들의 다양한 정보

Data Overview

45211명의 개인 정보 및 신용 정보

data shape :(45211,17)

[개인 정보] Age, Job, Marital, Education, Housing

[신용 정보] Default, Loan, Balance,

[마케팅 정보] Contact, Day, Month, Duration, Campaign, Pdays, Previous, Poutcome

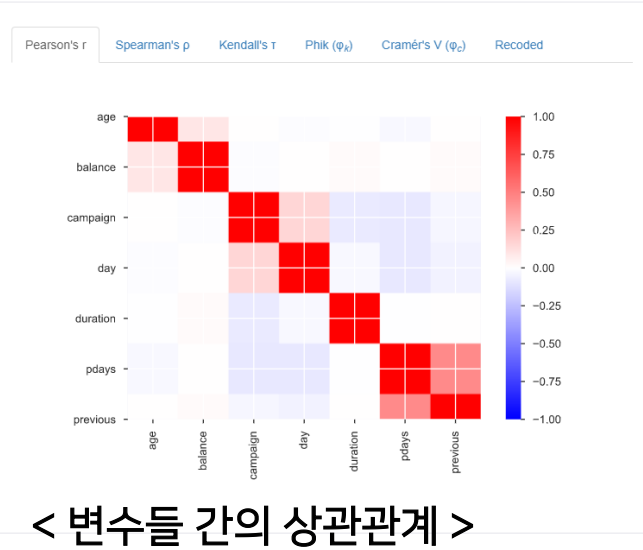
* (ex. 홍보 여부, 최근 연락 일시, 등)

Y : 정기예금 가입 여부

3. EDA

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no

Correlations

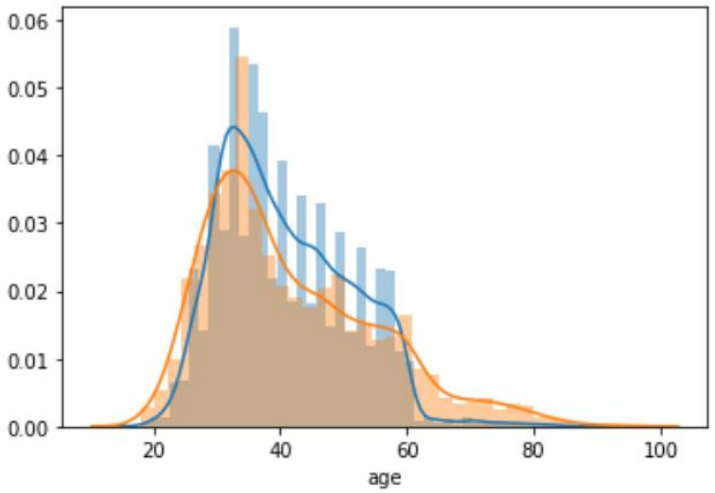


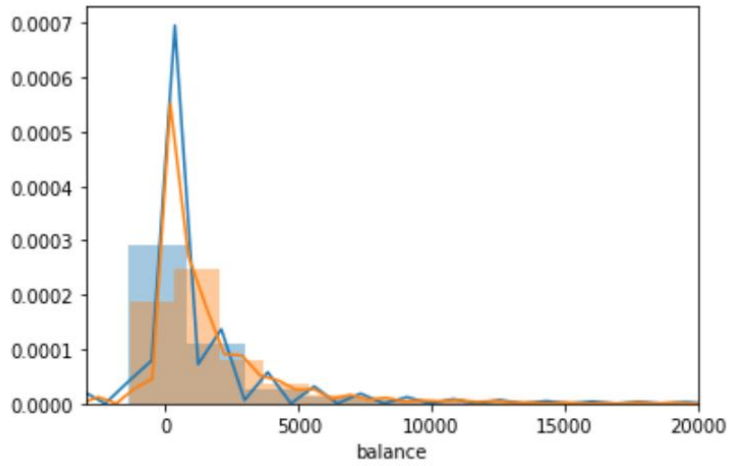
< 변수들 간의 상관관계 >

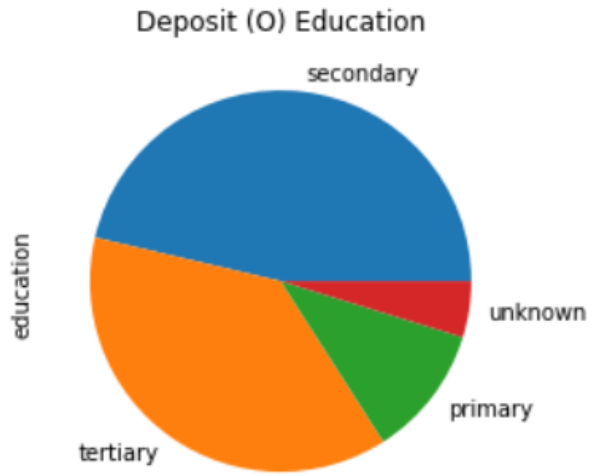
y	age	balance	day	duration	campaign	pdays	previous
no	40.84	1303.71	15.89	221.18	2.85	36.42	0.50
yes	41.67	1804.27	15.16	537.29	2.14	68.70	1.17

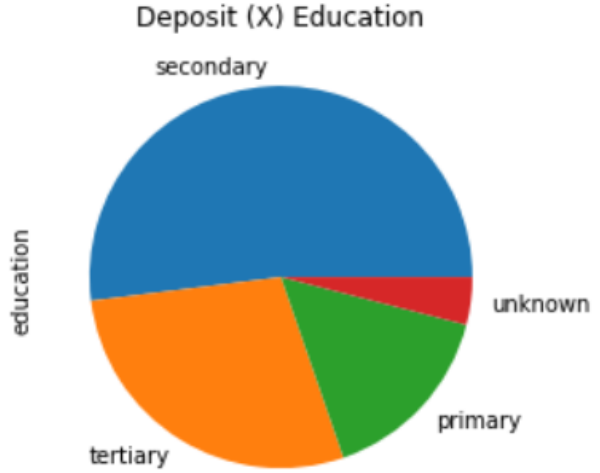
< 정기예금 가입 여부에 따른 numeric feature들의 차이 >

3. EDA










```
1  no['default'].value_counts()
```

```
no     39159
yes      763
Name: default, dtype: int64
```

```
1  yes['default'].value_counts()
```

```
no     5237
yes      52
Name: default, dtype: int64
```

정기예금 가입자와

비가입자 간의 차이는?

3. 데이터 전처리

1. Categorical 변수인 'education'을 Numeric 변수로 바꿔줌

(primary – secondary – tertiary 라는 categorical 변수로 표시되어 있지만,

이는 1,2,3학년이라는 order가 있는 변수이므로 numeric 변수로 바꿔주는 것이 적절)

2. Category 변수들의 Dummy화

(모델에 학습 가능한 형태로 만들기 위해, categorical 변수들을 dummy variable로 바꿔주어야 한다.

Column 개수는 17개에서 41개로 늘어났다)

```
1 dummied_data.head()
```

	age	education	default	balance	housing	loan	day	duration	campaign	pdays	...	month_jul	month_jun	month_mar	month_may	month_nov	mont
0	58	3	0	2143	1	0	5	261	1	-1	...	0	0	0	1	0	
1	44	2	0	29	1	0	5	151	1	-1	...	0	0	0	1	0	
2	33	2	0	2	1	1	5	76	1	-1	...	0	0	0	1	0	
3	47	2	0	1506	1	0	5	92	1	-1	...	0	0	0	1	0	
4	33	2	0	1	0	0	5	198	1	-1	...	0	0	0	1	0	

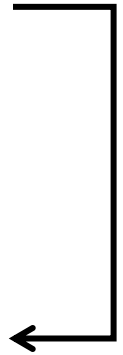
3. 데이터 전처리

3. Yes/No -> 1/0

4. 변수들 간의 규모(단위) 차이가 커서 Standard Scaling을 함

(변수들 간의 규모차이를 무시할 수 있게 되어 모델의 성능을 높일 수 있다)

	age	education	default	balance	housing	loan	day	duration	campaign	pdays	...	month_jul	month_jun	month_mar	month_may	month_nov	mont
0	58	3	0	2143	1	0	5	261	1	-1	...	0	0	0	1	0	
1	44	2	0	29	1	0	5	151	1	-1	...	0	0	0	1	0	
2	33	2	0	2	1	1	5	76	1	-1	...	0	0	0	1	0	
3	47	2	0	1506	1	0	5	92	1	-1	...	0	0	0	1	0	
4	33	2	0	1	0	0	5	198	1	-1	...	0	0	0	1	0	



	age	education	default	balance	housing	loan	day	duration	campaign	pdays	...	month_jul	month_jun	month_mar	month
0	1.606965	1.314507	-0.13549	0.256419	0.893915	-0.436803	-1.298476	0.011016	-0.569351	-0.411453	...	0	0	0	
1	0.288529	-0.218740	-0.13549	-0.437895	0.893915	-0.436803	-1.298476	-0.416127	-0.569351	-0.411453	...	0	0	0	
2	-0.747384	-0.218740	-0.13549	-0.446762	0.893915	2.289359	-1.298476	-0.707361	-0.569351	-0.411453	...	0	0	0	
3	0.571051	-0.218740	-0.13549	0.047205	0.893915	-0.436803	-1.298476	-0.645231	-0.569351	-0.411453	...	0	0	0	
4	-0.747384	-0.218740	-0.13549	-0.447091	-1.118674	-0.436803	-1.298476	-0.233620	-0.569351	-0.411453	...	0	0	0	

3. 데이터 전처리

5-fold Cross Validaton

5. Train & Test Split (0.7 : 0.3)

Train 과 Test를 0.7 : 0.3으로 나누고, Train data의 일부를 validation data로 이용하였다

Train (0.7)					Test (0.3)
				validation	
			validation		
		validation			
	validation				
validation					

4. 모델링

(1) Classification을 위해 사용할 Model

[Basic Models]	[Ensemble]	[Neural Net]
1. KNN	4. Random Forest	7. Neural Net
2. Naïve Bayes	5. XGBoost	
3. Decision Tree	6. LightGBM	

(2) Metric : Accuracy (Validation Score)

4. 모델링

(1) Classification을 위해 사용할 Model

[Basic Models]	[Ensemble]	[Neural Net]
1. KNN	4. Random Forest	예측력은 좋지만, 앞의 모델들에 비해 해석력은 BAD 7. Neural Net
2. Naïve Bayes	5. XGBoost	
3. Decision Tree	6. LightGBM	(앞의 6개 모델로도 성능이 좋다면, 굳이 이용 X)

(2) Metric : Accuracy (Validation Score)

4. 모델링

(1) Classification을 위해 사용할 Model

[Basic Models]	[Ensemble]	[Neural Net]
1. KNN	4. RandomForest	7. MLP Net
2. Naïve Bayes	5. XGBoost	
3. Decision Tree	6. LightGBM	

이 7가지 Model을 사용하여

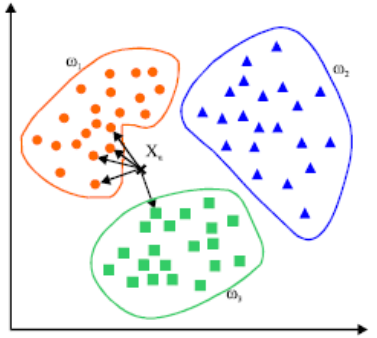
Validation Score가 가장 높은 모델을 사용!

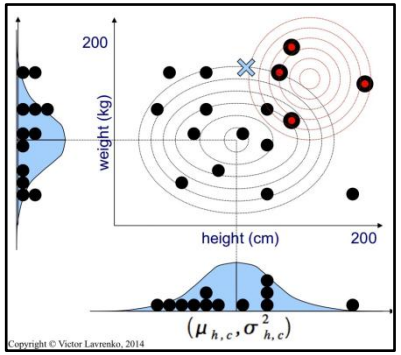
(2) Metric : Accuracy (Validation Score)

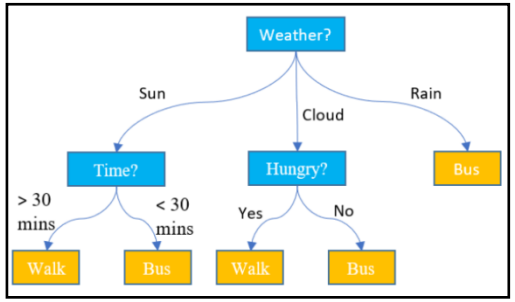
4. 모델링

Model Outline 1. Basic Model

1. KNN	가장 가까운 K개의 데이터와 같은 class에 속하도록 분류
(Gaussian) 2. Naïve Bayes	(변수들이 numeric일 때 사용하는 Naïve Bayes) 변수들의 분포를 정규분포로 가정하고, 각 데이터가 각 class에 속할 확률을 계산하여 가장 높은 확률의 class로 분류
3. Decision Tree	규칙을 가지고 데이터를 구분해 나가서, 최종적으로 같은 node에 속하게 되면 같은 class로 분류





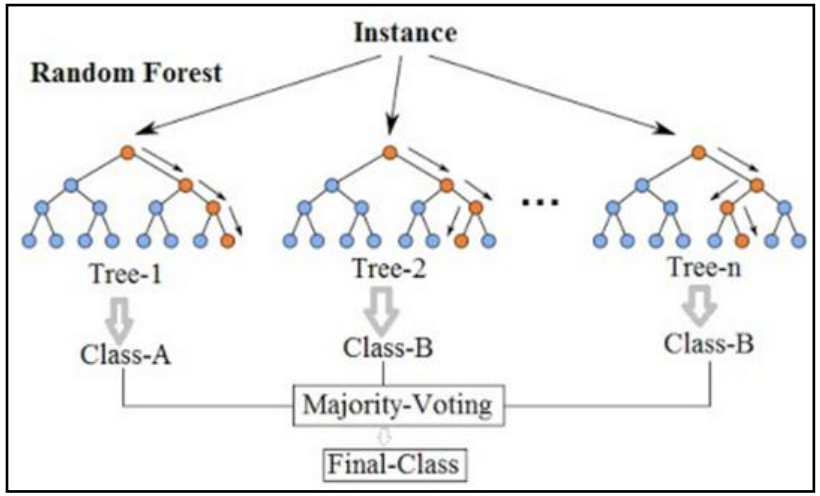


4. 모델링

Model Outline 2. Ensemble

Ensemble : 여러 개의 basic model을 만든 뒤 이 결과들을 종합해서 결과값 산출!

4. Random Forest	여러 Decision Tree를 만들어서, 이들의 결과를 종합해서 class 분류 (하나의 decision tree는 모든 feature를 사용하지 않고, random하게 선택해서 모델을 만든다)



DT : 하나의 잘 짜여진 모델(strong learner)

RF : 여러 개의 쉽게 짜여진 모델(weak learners)

하나의 분류기(Tree)를 만들 때, 주어진 모든 feature를 사용하지 않고,

일부의 변수만을 사용하여 random 하게 Tree들을 생성!

-> Tree들간의 De correlation -> Overfitting 방지!

Ex) (a,b,e,f 만을 사용한 Tree 1) + (b,d,e,k 만을 사용한 Tree 2) …

4. 모델링

Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

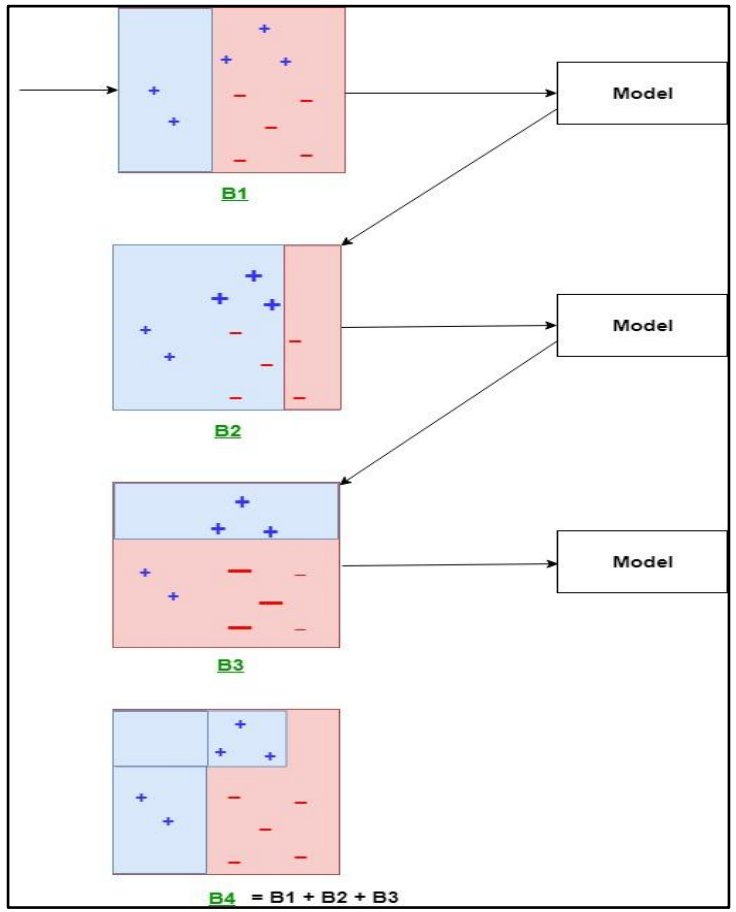
Model Outline 2. Ensemble

5. XGBoost	여러 모델을 순차적으로 만들어, **이전 모델이 정확하게 분류해내지 못한 데이터에 더 가중치를 부여**하여(신경을 쓰며) 모델을 만들어 나감. * Kaggle 대회에서 높은 순위권을 차지하는 좋은 성능의 모델!
6. Light GBM	**XGBoost와 유사**한 알고리즘 * XGBoost 대비 성능 향상 및 자원 소모 최소화 & 대용량 데이터 학습 가능

- Ensemble기법으로, "여러 개의" Classifier를 만들어서 예측을 함!
- Boosting Method) **"잘못 예측"한 답에 대해 더 많은 가중치**를 두어서,
  틀린 것을 더 잘 학습하도록 훈련함! -> **더 좋은 결과(정확도)!**

4. 모델링

Model Outline 2. Ensemble

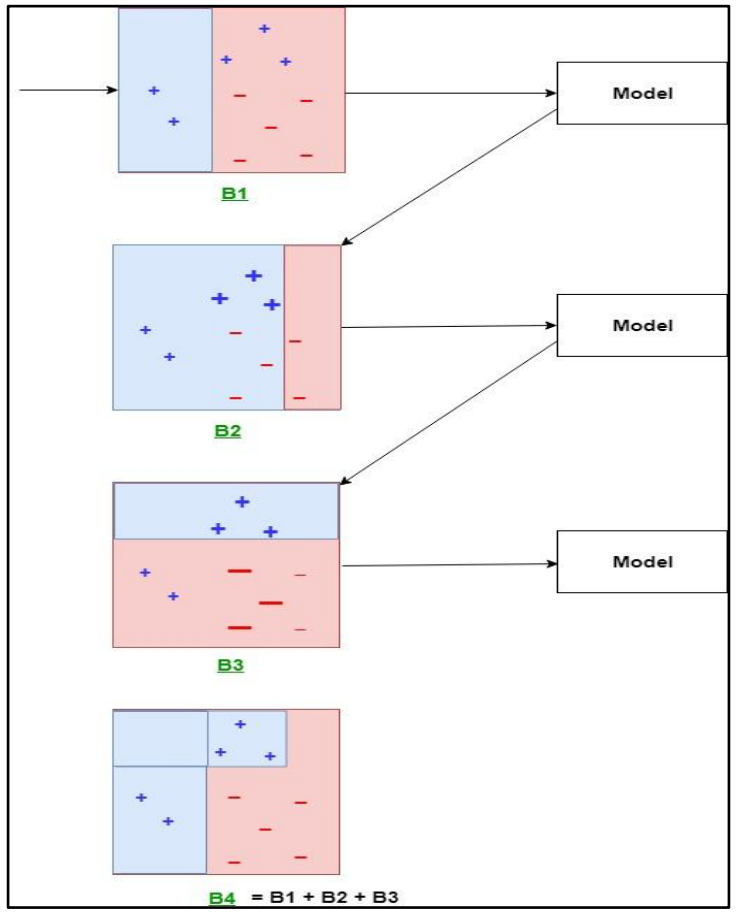


Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

<- 앞의 Model이 틀리게 예측한 것을 더 잘 예측하기 위해 노력!

4. 모델링

Model Outline 2. Ensemble



Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

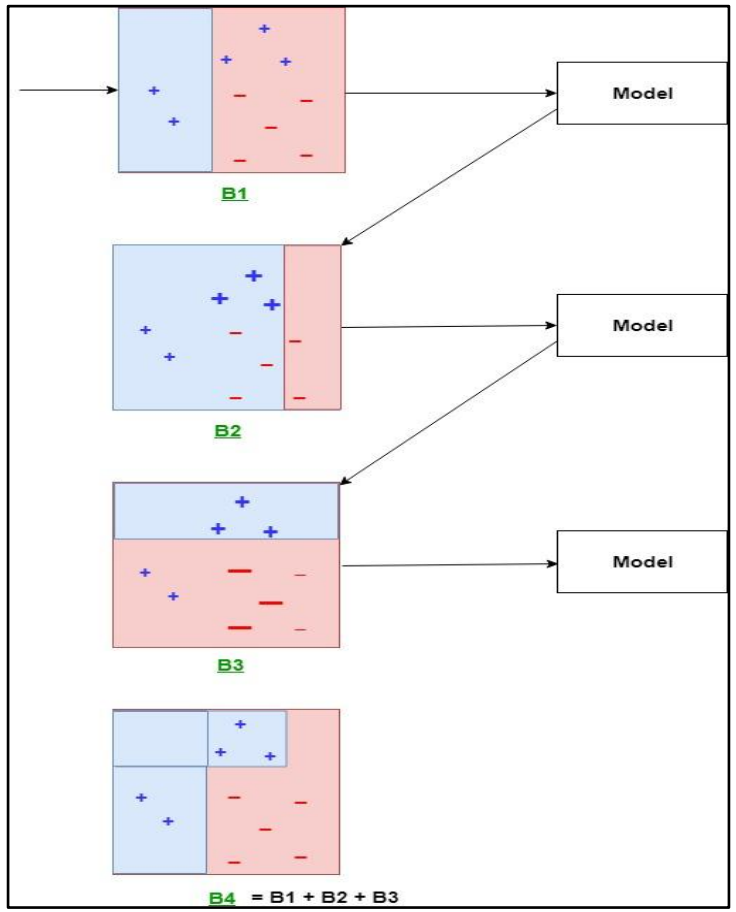
오답!

$$Y = M(x) + error$$

<- 앞의 Model이 틀리게 예측한 것을 더 잘 예측하기 위해 노력!

4. 모델링

Model Outline 2. Ensemble



Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

오답!

$$Y = M(x) + \underline{error}$$

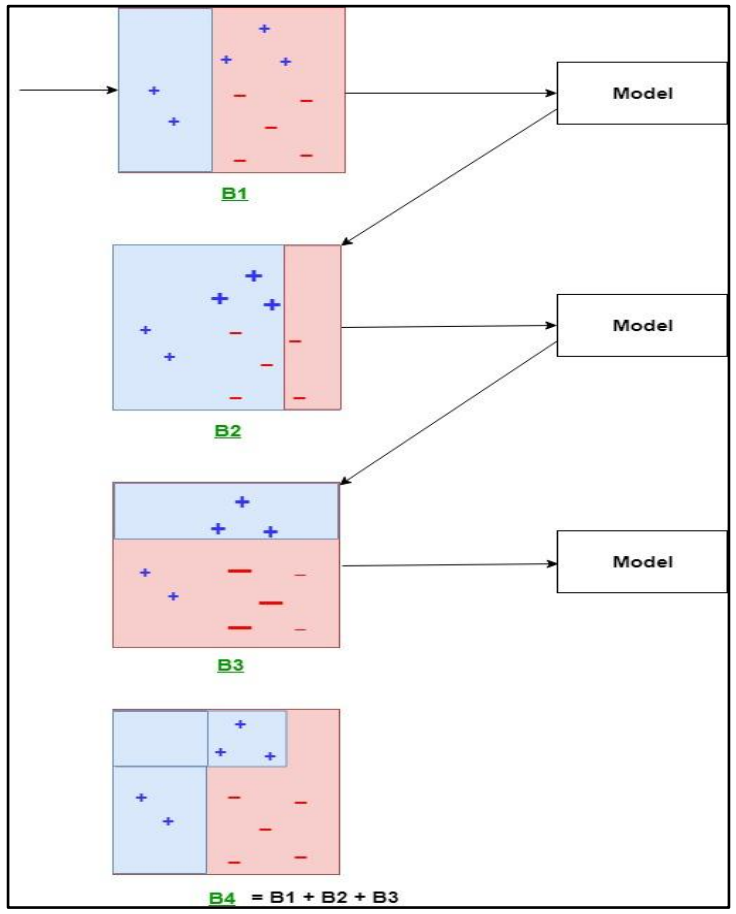
$$error = G(x) + error2$$

(오답을 예측하는 또 다른 모델)

<- 앞의 Model이 틀리게 예측한 것을 더 잘 예측하기 위해 노력!

4. 모델링

Model Outline 2. Ensemble



Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

오답!

$$Y = M(x) + \underline{error}$$

오답!

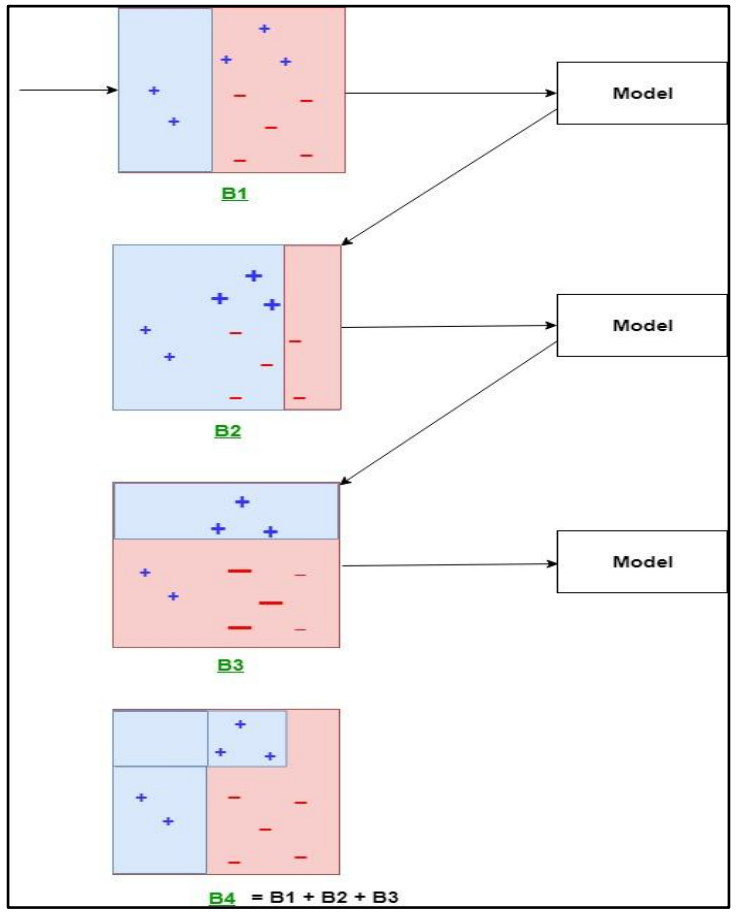
$$error = G(x) + error2$$

(오답을 예측하는 또 다른 모델)

<- 앞의 Model이 틀리게 예측한 것을 더 잘 예측하기 위해 노력!

4. 모델링

Model Outline 2. Ensemble



Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

오답!

$Y = M(x) + error$

오답!

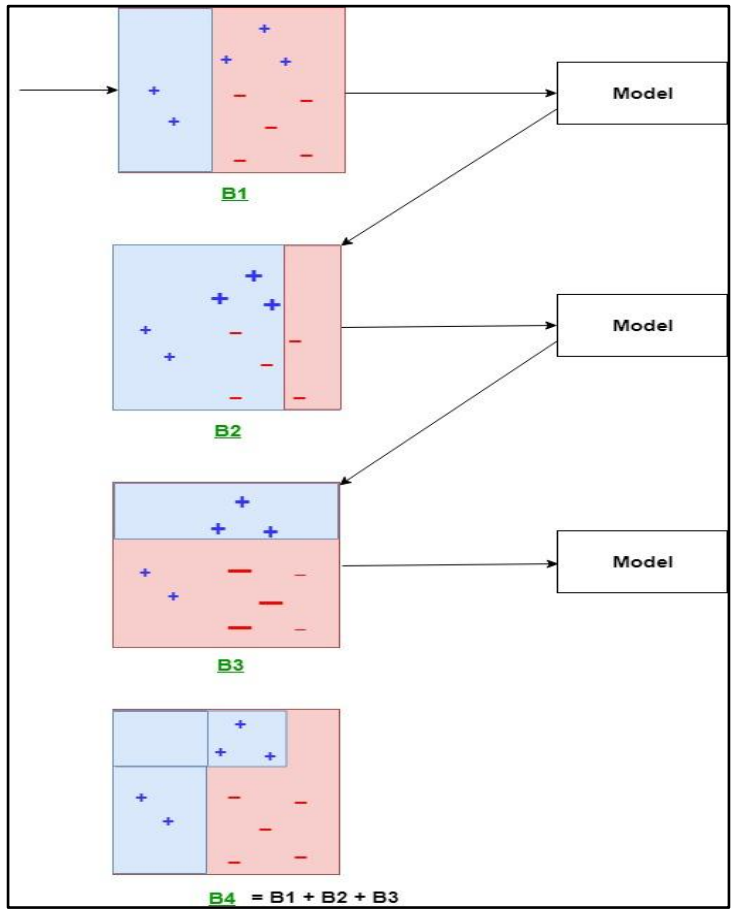
$error = G(x) + error2$ (오답을 예측하는 또 다른 모델)

$error2 = H(x) + error3$ (오답을 예측하는 또 다른 모델)

<- 앞의 Model이 틀리게 예측한 것을 더 잘 예측하기 위해 노력!

4. 모델링

Model Outline 2. Ensemble



Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

오답!

$Y = M(x) + error$

오답!

$error = G(x) + error2$ (오답을 예측하는 또 다른 모델)

$error2 = H(x) + error3$ (오답을 예측하는 또 다른 모델)

$$Y = M(x) + G(x) + H(x) + error3 \ (4)$$

앞의 Model이 "틀리게 예측한 것"을 더 잘 예측하기 위해 노력!

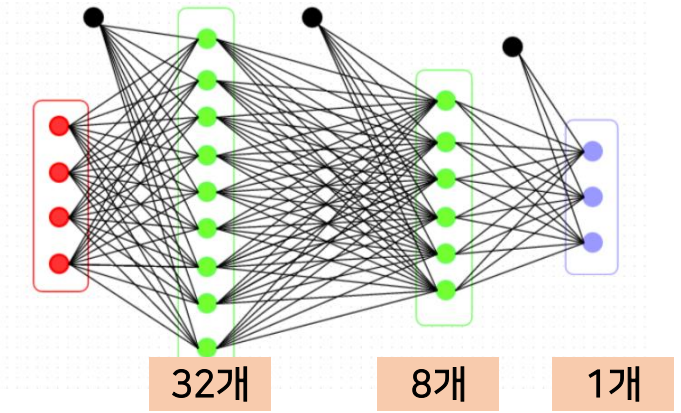
4. 모델링

Model Outline 3. Neural Net

7. Neural Net

인공 신경망을 이용한 Model

- 기존의 Machine Learning보다 복잡 + 정교!
- BUT 기존의 모델들보다 해석력이 낮음! (Black Box 모델)



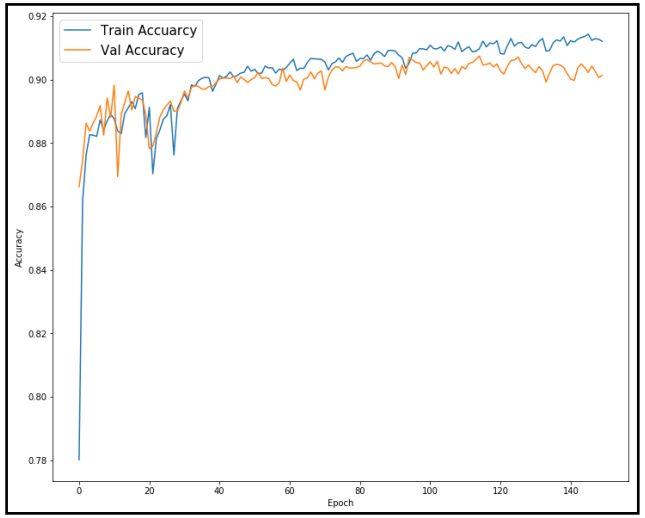
(1) Model의 구성

Layer (type)	Output Shape	Param #
dense_12 (Dense)	(None, 32)	1312
batch_normalization_4 (Batch	(None, 32)	128
dense_13 (Dense)	(None, 8)	264
dense_14 (Dense)	(None, 1)	9

Total params: 1,713
Trainable params: 1,649
Non-trainable params: 64

32 node – BN – 8 node – 1 node

(2) Epoch 별 Accuracy



(3) Result (Train & Validation Accuarcy)

	accuracy	val_accuracy
149	0.912	0.901

4. 모델링

Ensemble : 여러 개의 basic model을 만든 뒤 **이 결과들을 종합해서** 결과값 산출!

Model Outline 2. Ensemble

7. Neural Net

- 기존의 Machine Learning보다 복잡 + 정교!

기존의 모델(1~6번)로도 충분히 해결 가능한 모델이라면,

Neural Net 대신 기존의 모델을 이용! (Parameter만 1,713개)

(아주 복잡한 문제가 아닌 이상, Neural Net보다는 다른 모델을 사용!)

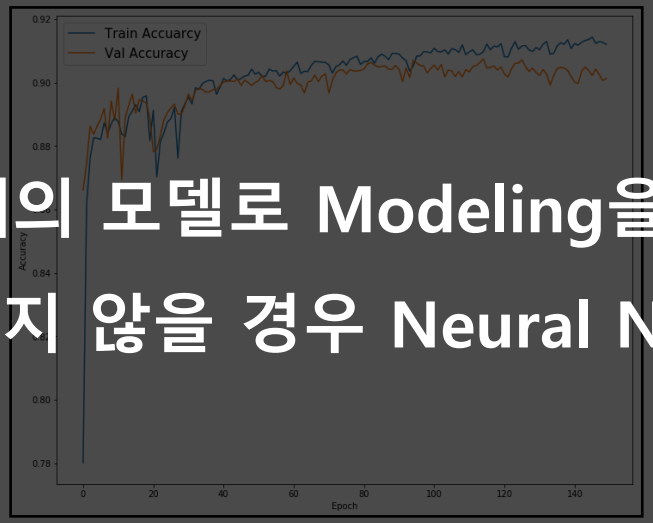
(1) Model의 구성

Layer (type)	Output Shape	Param #
dense_12 (Dense)	(None, 32)	1312
batch_normalization_4 (Batch	(None, 32)	128
dense_13 (Dense)	(None, 8)	9
dense_14 (Dense)	(None, 1)	9

Total params: 1,713
Trainable params: 1,649
Non-trainable params: 64

32 node – BN – 8 node – 1 node

(2) Epoch 별 Accuracy



기존의 6개의 모델로 Modeling을 해보고,

결과가 좋지 않을 경우 Neural Net으로!

(3) Result
(Train & Validation Accuarcy)

	accuracy	val_accuracy
149	0.912	0.901

4. 모델링

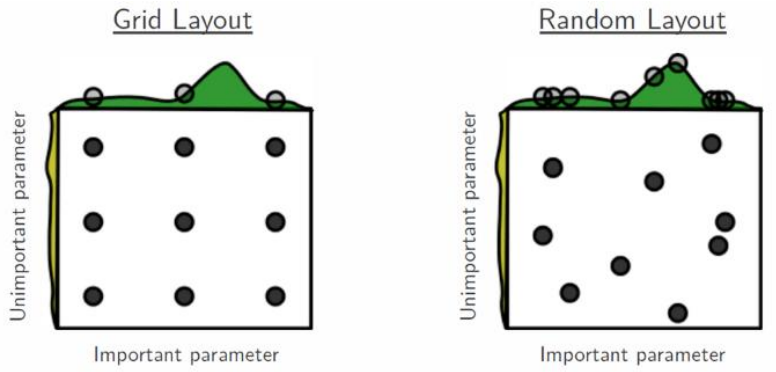
(CV의 Accuracy를 가장 높게 하는 것으로!)

보다 정교한 모델을 위한 Hyperparameter Tuning

KNN : Grid Search

Decision Tree, Random Forest, XGBoost, LightGBM : Random Search

Naïve Bayes : No parameter



1. KNN	n_neighbors (k,즉 최 근접 데이터의 개수)
2. Decision Tree	max_depth (나무 최대 깊이) , min_sample_split (최소 분기 데이터 수)
3. Random Forest	max_depth, n_estimator (분류기 개수)
4. XGBoost	max_depth, learning_rate (학습율)
5. Light GBM	num_leaves, n_estimator, learning_rate (학습율)

4. 모델링

각 모델 별 최적의 hyperparameter값은?

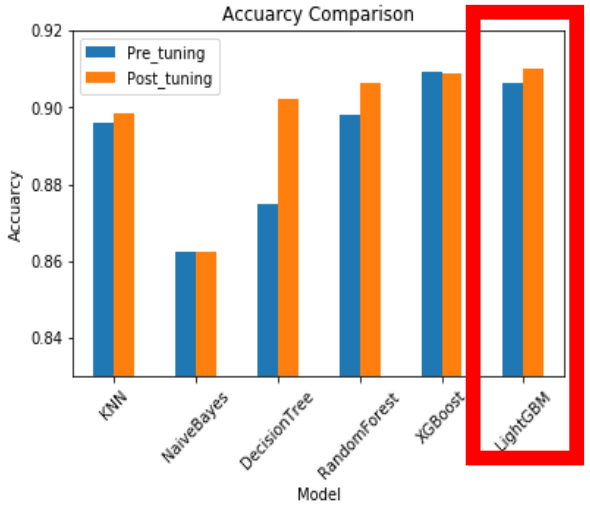
1. KNN	{'n_neighbors': 9}
2. Decision Tree	{'min_samples_split': 4, 'max_depth': 6}
3. Random Forest	{'n_estimators': 136, 'max_depth': 29}
4. XGBoost	{'max_depth': 6, 'learning_rate': 0.1}
5. Light GBM	{'num_leaves': 49, 'n_estimators': 146, 'max_depth': 29, 'learning_rate': 0.05}

5. 모델 평가

모델 간의 성능 비교 (기준 : Cross-Validation Score)

Hyper Parameter Tuning 이후 , LightGBM이 가장 좋은 성능을 보였다.

Cross-Validation Score에서 약 91%의 정확도를 보였다



	Pre_tuning	Post_tuning
KNN	0.896	0.898
NaiveBayes	0.862	0.862
DecisionTree	0.875	0.902
RandomForest	0.898	0.906
XGBoost	0.909	0.909
LightGBM	0.906	0.910

< Model 별 순위 >

(Tuning 이후의 CV score)

1. Light GBM (91.0%)
2. XGBoost (90.9%)
3. Random Forest (90.6%)
4. Decision Tree (90.2%)
5. Neural Net (90.1%)
6. KNN (89.8%)
7. Naïve Bayes (86.2%)

5. 모델 평가

가장 정확하게 정기예금자를 분류해냈던 "Light GBM"….

이 모델을 이용하여 예측한 train & test dataset 의 'confusion matrix' & 'accuracy'는?

```
1 confusion_matrix(y_train,train_pred)
```

```
array([[27312,   641],
       [ 1346,  2348]], dtype=int64)
```

```
1 accuracy_score(y_train,train_pred).round(3)
```

```
0.937
```

Train Accuracy

```
1 confusion_matrix(y_test,test_pred)
```

```
array([[11479,   490],
       [  801,   794]], dtype=int64)
```

```
1 accuracy_score(y_test,test_pred).round(3)
```

```
0.905
```

Test Accuracy

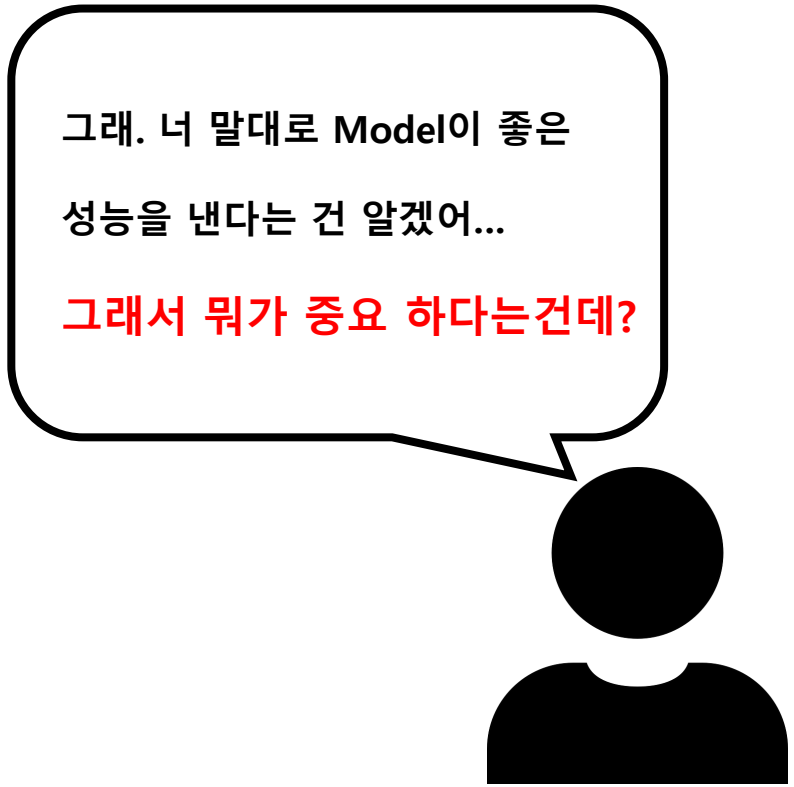
5. 모델 해석

높은 Accuracy 만큼이나 중요한 것이, **"모델의 해석력"**

우리가 만든 모델에서, 어떠한 변수가 중요한 역할을 했는가?

즉, 사람 A는 정기 예금자로, 사람 B는 비정기예금자로 분류하는데

있어서 "가장 중요한 역할을 한 변수"가 무엇인가?



5. 모델 해석

높은 Accuracy 만큼이나 중요한 것이 **"모델의 해석력"**

모델을 해석해주는 여러 지표들

Feature Importance

PDP (Partial Dependence Plot)

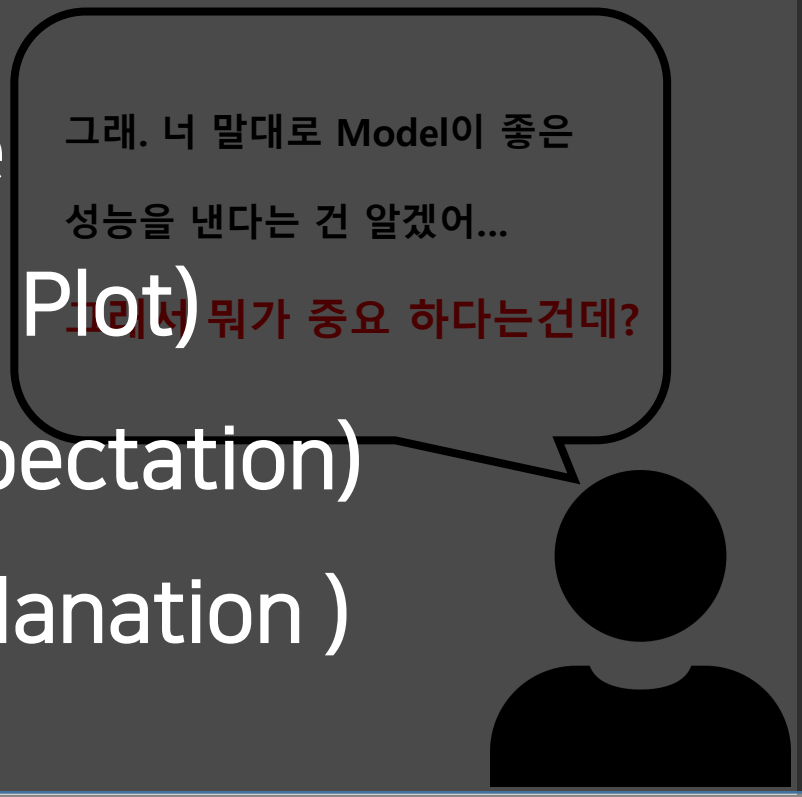
ICE(Individual Conditional Expectation)

SHAP (Shapely Additive eXplanation)

우리가 만든 모델에서, 어떠한 변수가 중요한 역할을 했는가?

즉, 사람 A는 정기 예금자로, 사람 B는 정기 예금자로 분류되었을 때

있어서 **"가장 중요한 변수"**가 무엇인가?



5. 모델 해석

높은 Accuracy 만큼이나 중요한 것이 **"모델의 해석력"**

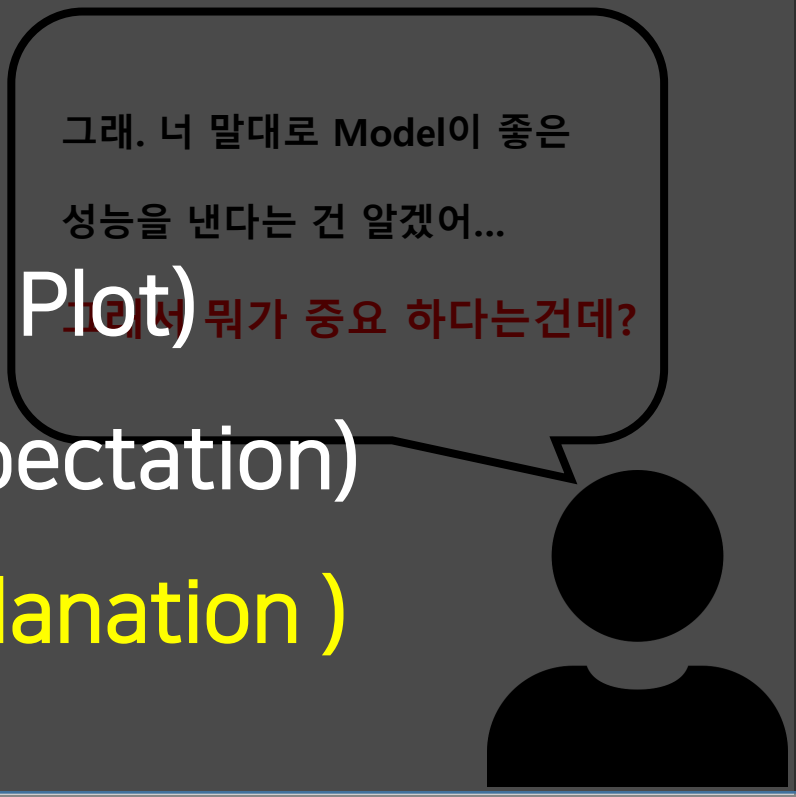
모델을 해석해주는 여러 지표들

Feature Importance

PDP (Partial Dependence Plot)

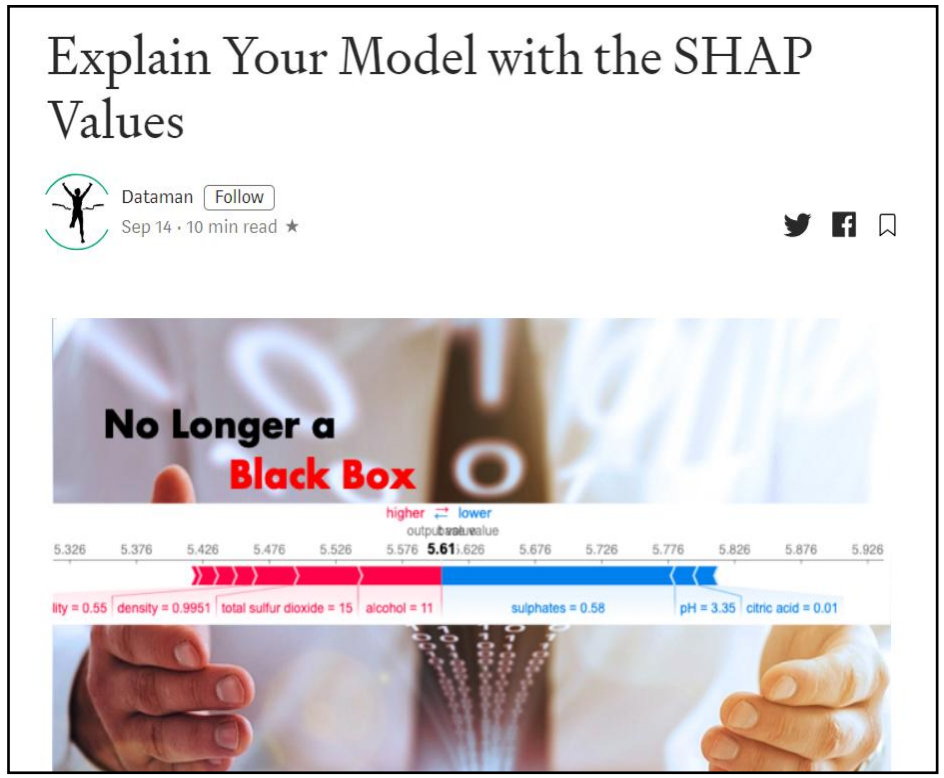
ICE (Individual Conditional Expectation)

SHAP (Shapely Additive eXplanation)



5. 모델 해석

SHAP (SHapley Additive exPlanations)를 통한 모델 해석



5.10.1 Definition

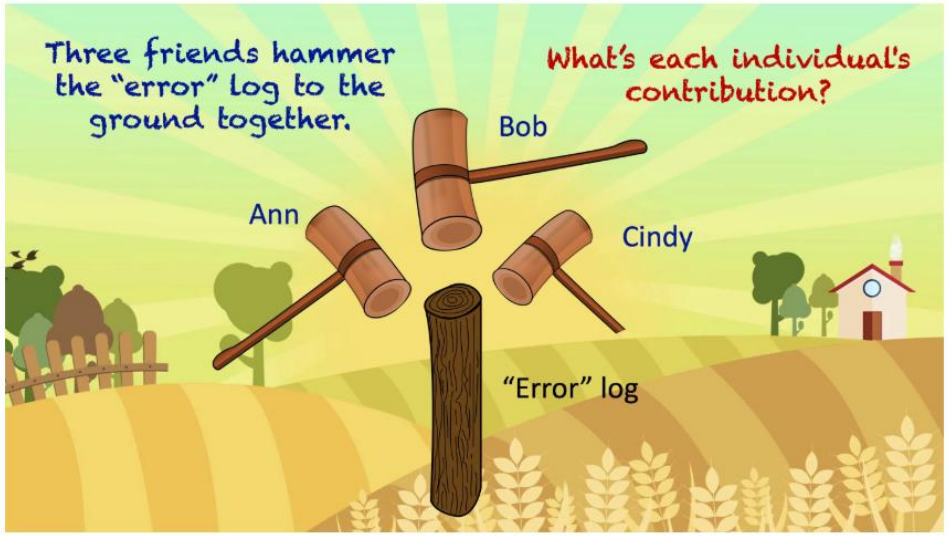
The goal of SHAP is to explain the prediction of an instance x by computing the contribution of each feature to the prediction. The SHAP explanation method computes Shapley values from coalitional game theory. The feature values of a data instance act as players in a coalition. Shapley values tell us how to fairly distribute the “payout” (= the prediction) among the features. A player can be an individual feature

SHAP값이 높을 수록, 해당 모델에

큰 기여(즉, 큰 영향)을 미친 변수라는 것을 의미!

5. 모델 해석

SHAP (SHapley Additive exPlanations)를 통한 모델 해석



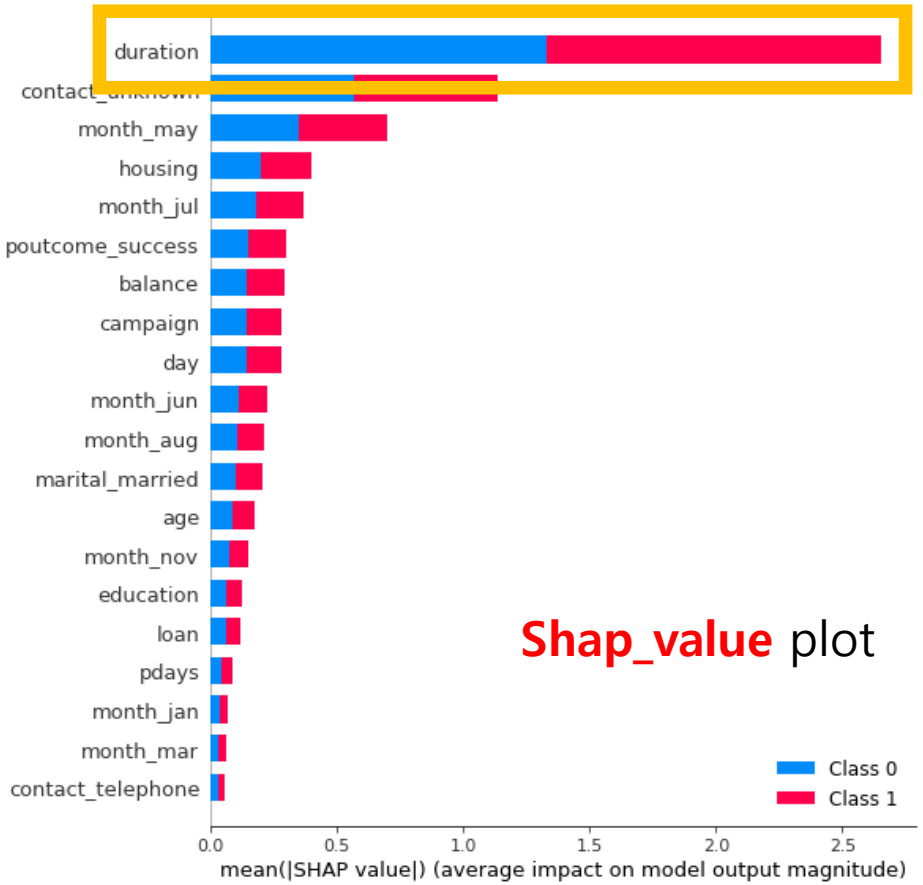
	Marginal contribution			inches
Combination	Ann	Bob	Cindy	Total
A, B, C	2	32	4	38
A, C, B	4	34	0	38
B, A, C	2	32	4	38
B, C, A	0	28	10	38
C, A, B	2	36	0	38
C, B, A	0	28	10	38
Average	2	32	4	38

해당 변수를 "넣었을 때"와, "뺐을 때"의 결과 차이를 통해 기여도 파악!

$$\phi_{ij} = \sum_{\text{All.orderings}} val(\{\text{features.before.j}\} \cup x_{ij}) - val(\{\text{features.before.j}\})$$

5. 모델 해석

```
1 shap.summary_plot(shap_values, X_test)
```



Shap_value plot

duration : last contact duration

(가장 최근 (은행기관-고객) 연락한 시간 (단위:초))

은행 기관이 "고객에게 가장 마지막으로 연락했을 때의 통화 시간(=duration)" 이 정기 예금 가입자에게 가장 큰 영향을 미친 것을 확인할 수 있었다!

SUGGESTION 1. 보편적 전략 (고객군 별 X. 모든 고객군 O)

(= duration 변수)

각각의 고객이 가장 마지막으로 통화했을 때의 시간을 파악하여,

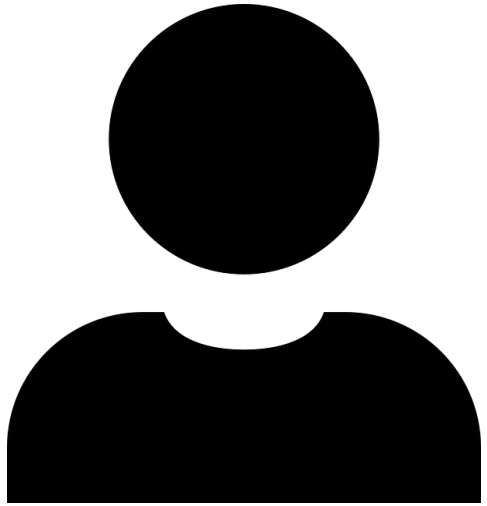
이 시간이 크지만, 정기예금을 아직 가입하지 않은 사람이 있다면,

이들을 대상으로 집중적인 정기예금 가입 캠페인/마케팅을 하면 좋을 것이다.

이 45000명의 고객들을 일괄적인 전략뿐만 아니라,

각각의 고객군의 특징에 맞는 최적의 방식으로

전략을 세우면 보다 효과적이지 않을까?

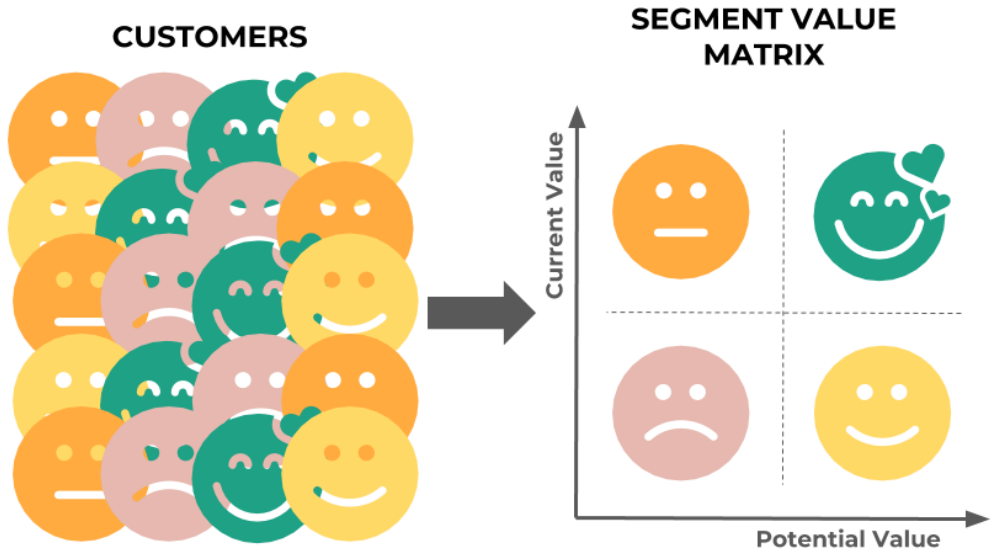


6. Clustering

고객을 n개의 고객 군으로 클러스터링하여,

"각 고객 군에 맞는 최적의 방식"으로 "정기예금을 유도"하는 전략 세우기

1. Clustering을 위한 전처리
2. Clustering Method
3. Cluster(고객 군)간의 비교



6. Clustering – clustering을 위한 전처리

1. PCA (Principal Component Analysis)를 통한 차원축소

(기존) 40개의 Feature를 가진 data

(NEW) 9개의 주성분(PC)으로 구성된 data

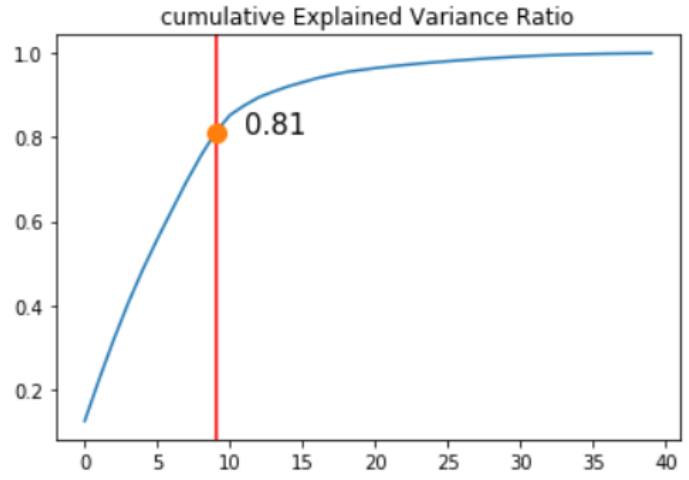
```
# 차원이 축소된 것을 확인할 수 있다
X.shape, pcadf.shape
```

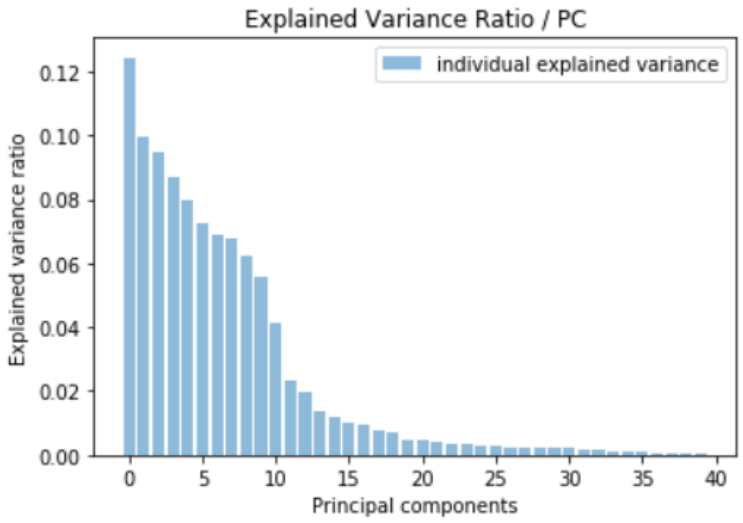
((45211, 40), (45211, 9))

```
pcadf = pd.DataFrame(pca_values2, columns = ['PC'+str(i) for i in range(1,10)])
pcadf.index = X.index
pcadf.head()
```

	PC1	PC2	PC3	PC4	PC5	PC6	PC7	PC8	PC9
0	0.013812	0.618380	-0.068002	-1.286128	0.027843	-0.717798	-0.210573	0.630735	0.440883
1	0.139942	-0.676180	-0.663513	-1.149365	-0.357059	-1.120233	0.212301	0.026766	0.227911
2	0.066223	-1.770193	-1.075489	-0.507574	0.819477	-0.941089	-1.866937	0.512142	0.404800
3	0.051579	-0.157233	-1.137654	-0.993902	-0.568032	-1.198240	0.059776	0.535708	0.223792
4	-0.177710	-0.209624	0.114222	-1.221593	0.386855	-1.166395	0.406823	-0.747862	0.214874

9차원으로도 약 81%의 설명력을 가짐





6. Clustering – clustering을 위한 전처리

2. Outlier 제거

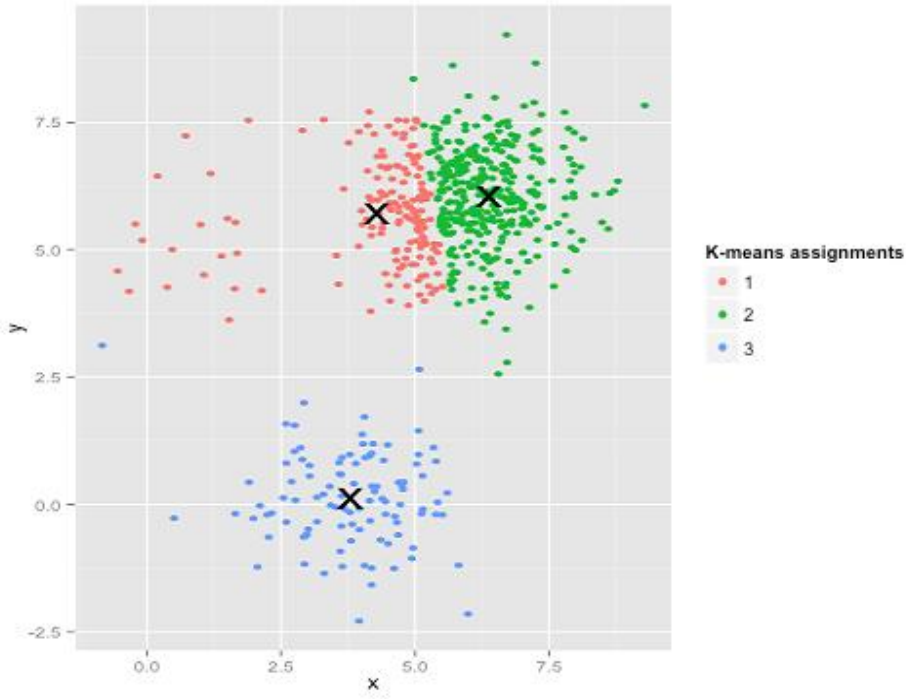
- Z score (-3~3)에서 벗어나는 데이터 제거!
- 45211명 -> 41396명

```
from scipy import stats

for i in pcadf.columns:
    pcadf = pcadf[np.abs(pcadf[i]-pcadf[i].mean()) <= (3*pcadf[i].std())]
```

```
# 차원이 축소된 것을 확인할 수 있다
X.shape, pcadf.shape
```

```
((45211, 40), (41396, 9))
```



Ex) Outlier로 인해 제대로 구분되지 않은 Cluster

6. Clustering – clustering을 위한 전처리

2. Outlier 제거

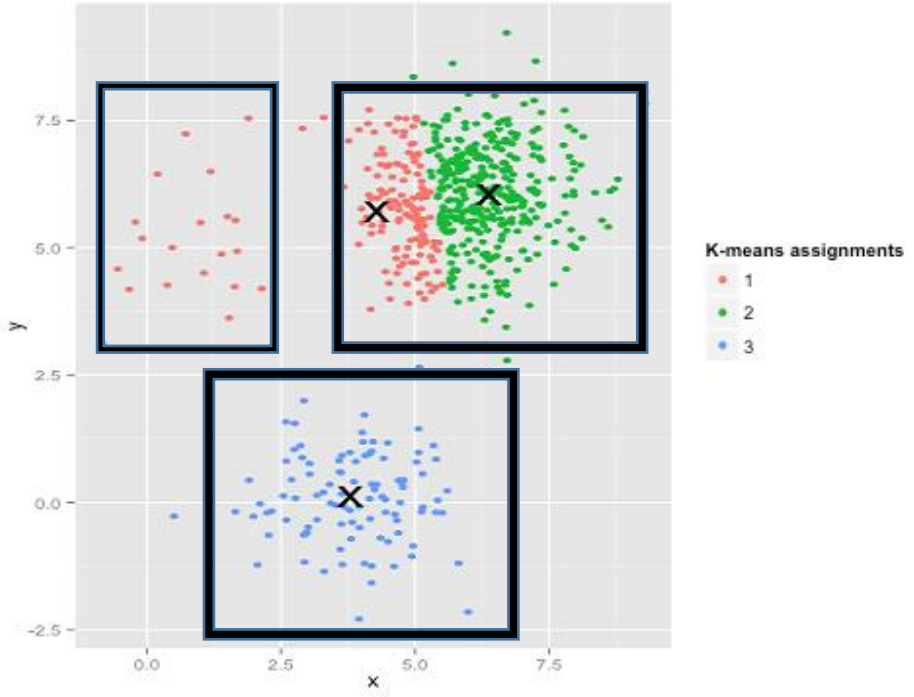
- Z score (-3~3)에서 벗어나는 데이터 제거!
- 45211명 -> 41396명

```
from scipy import stats

for i in pcadf.columns:
    pcadf = pcadf[np.abs(pcadf[i]-pcadf[i].mean()) <= (3*pcadf[i].std())]
```

```
# 차원이 축소된 것을 확인할 수 있다
X.shape, pcadf.shape
```

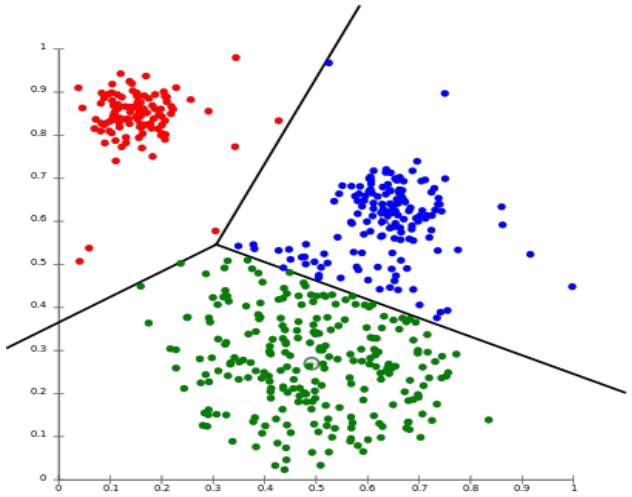
```
((45211, 40), (41396, 9))
```



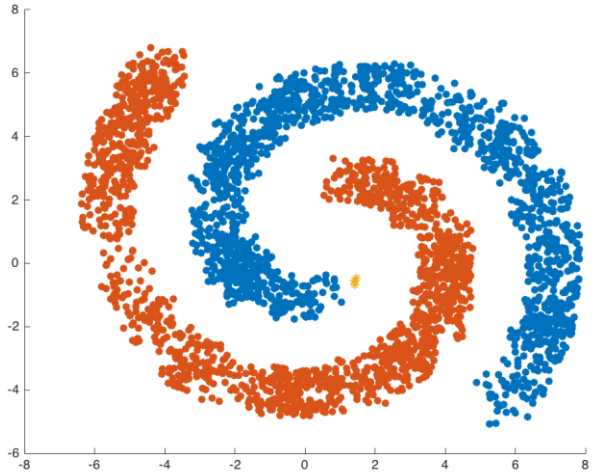
Ex) Outlier로 인해 제대로 구분되지 않은 Cluster

6. Clustering – clustering method

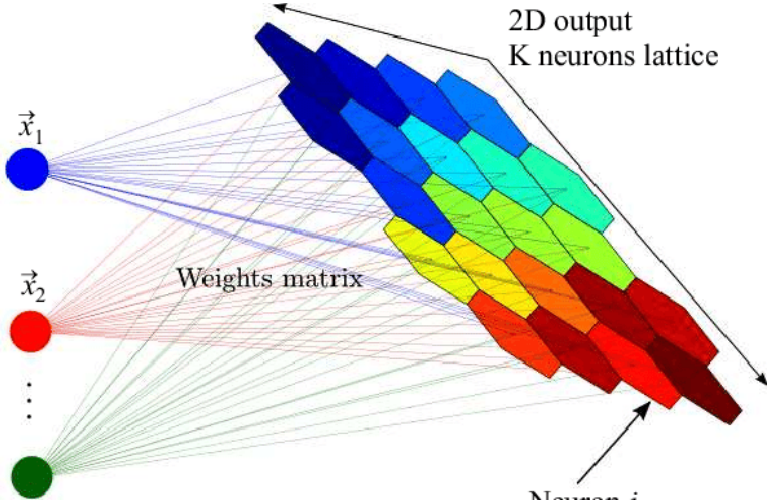
1. K-Means (Distance-Based)
2. DBSCAN (Density-Based)
3. SOM (= Self-Organizing Map)



K-Means



DBSCAN

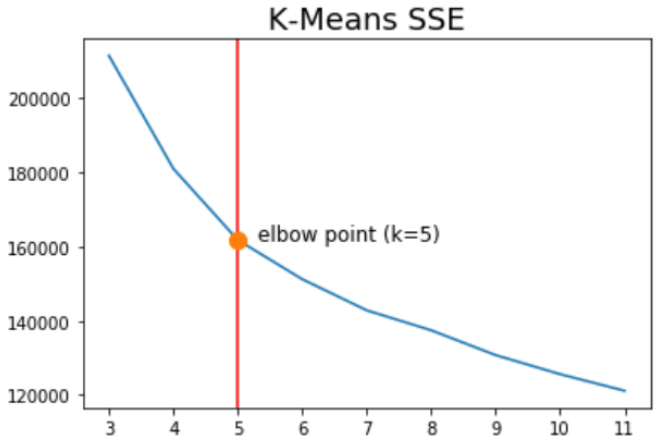


Self Organizing Map

6. Clustering – clustering method

1. K-Means (Distance-Based)

- 적절한 k의 개수 : K=5



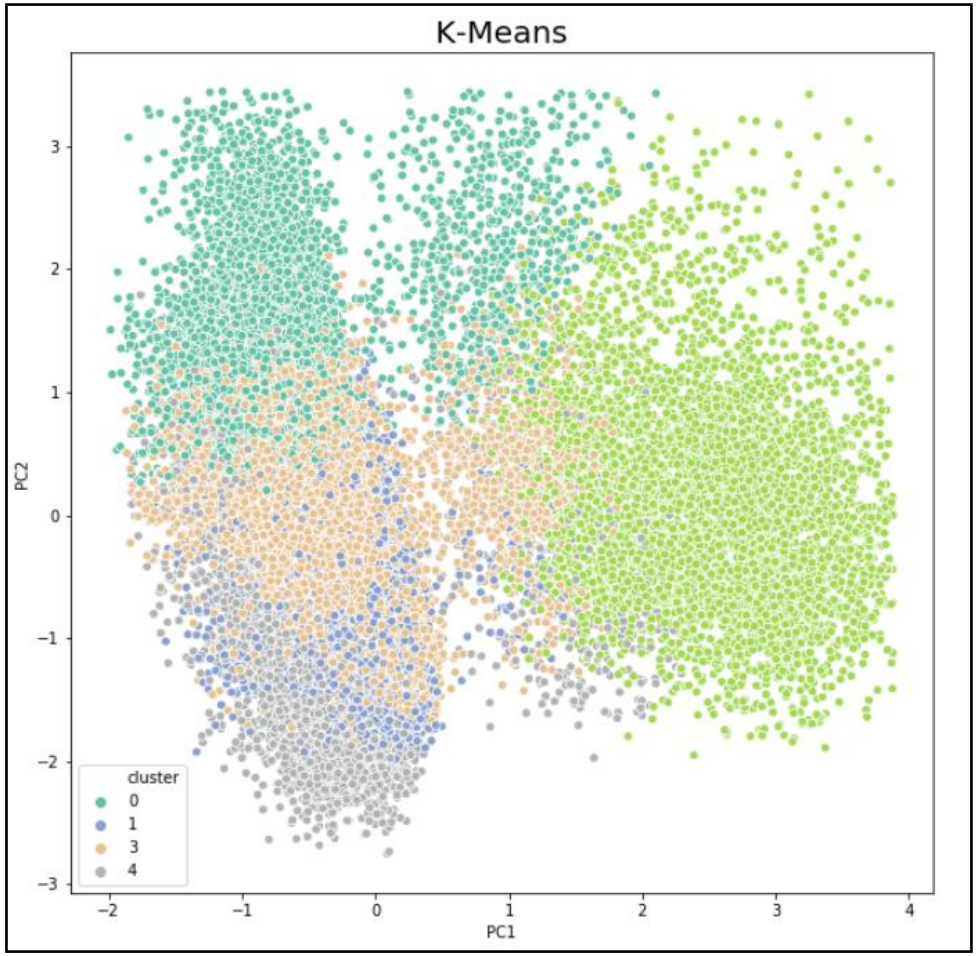
Cluster 별 고객 수?

1	11643
3	10864
0	7648
4	5956
2	5285

Name: cluster,

어느 정도 구분은 되어 보이지만,

아주 완벽히 잘 구분되어 보이진 않는다



(주성분)PC1 & PC2로 2차원 시각화

6. Clustering – clustering method

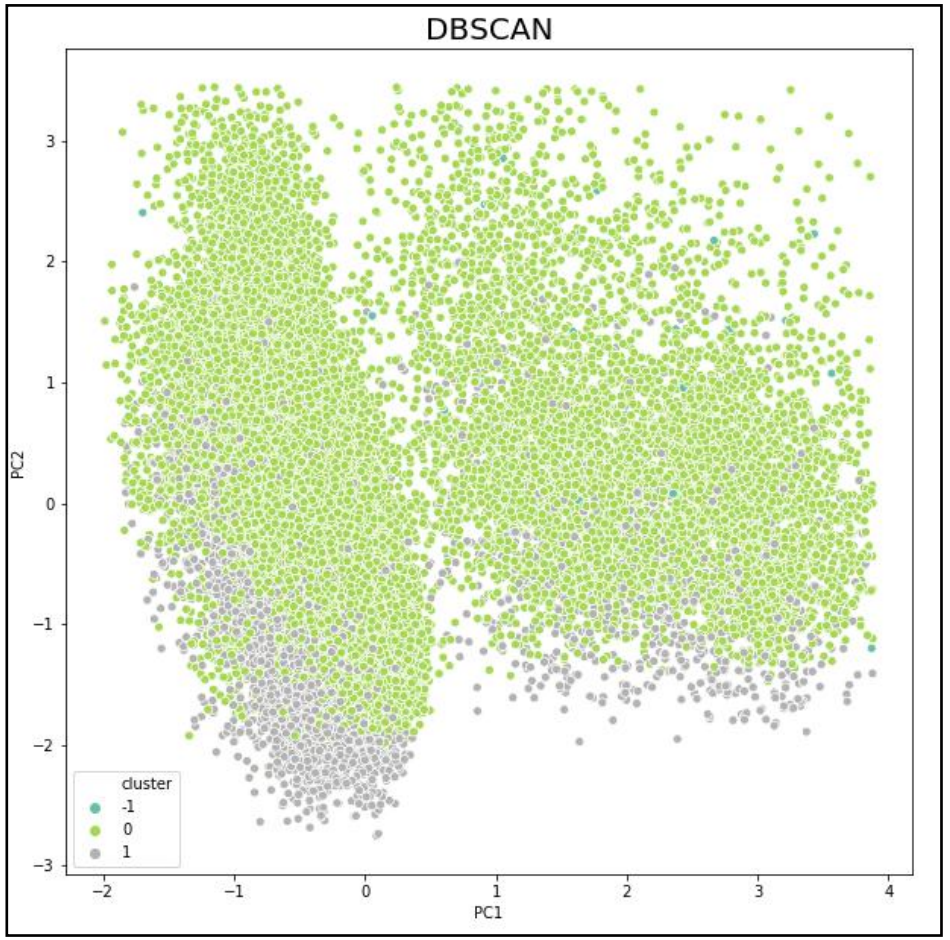
2. DBSCAN (Density-Based)

Density(밀도)를 기반으로 한 Clustering !

좋은 구분을 하지 못한 것으로 보임.

K-Means보다 못한 clustering…

보다 나은 방법은 없을까?

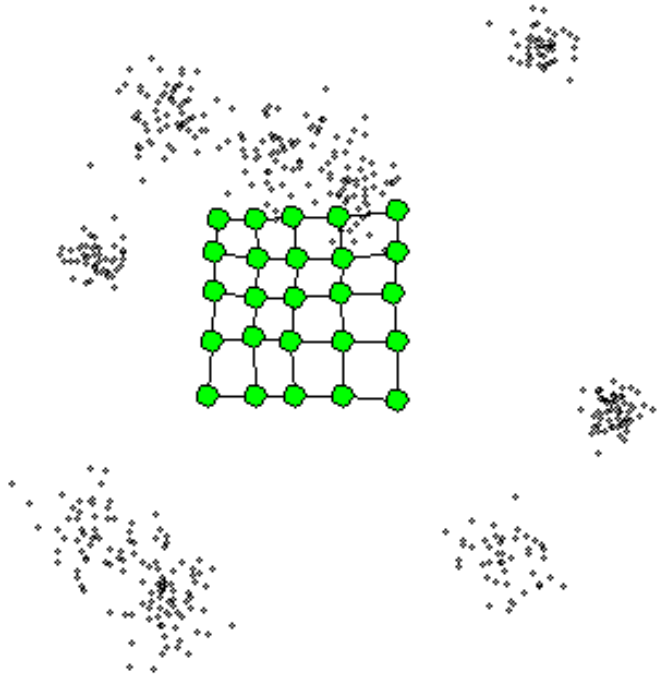


(주성분)PC1 & PC2로 2차원 시각화

6. Clustering – clustering method

3. SOM (Self-Organizing Map)

- Neural Net을 통한 Clustering 방법!
- 시간은 좀 걸리지만, 가장 고객 군을 잘 구분하는 방법이다!
- 격자 형식으로 Cluster를 생성하여, 인접한 Cluster끼리는 비슷한 특징을 가짐!



(1) 생성된 6개의 Cluster

Cluster 1 [0 0]	**Cluster 2 [0 1]**	**Cluster 3 [0 2]**
Cluster 4 [1 0]	**Cluster 5 [1 1]**	**Cluster 6 [1 2]**

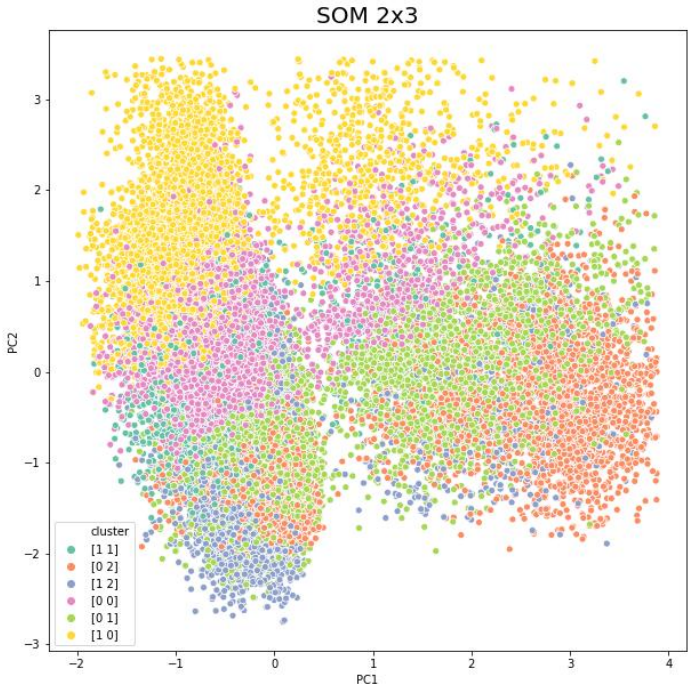
(2) Cluster 별 고객 수

```
[0 2]    8274
[1 2]    7962
[1 0]    7710
[0 0]    7551
[0 1]    5941
[1 1]    3958
Name: cluster,
```

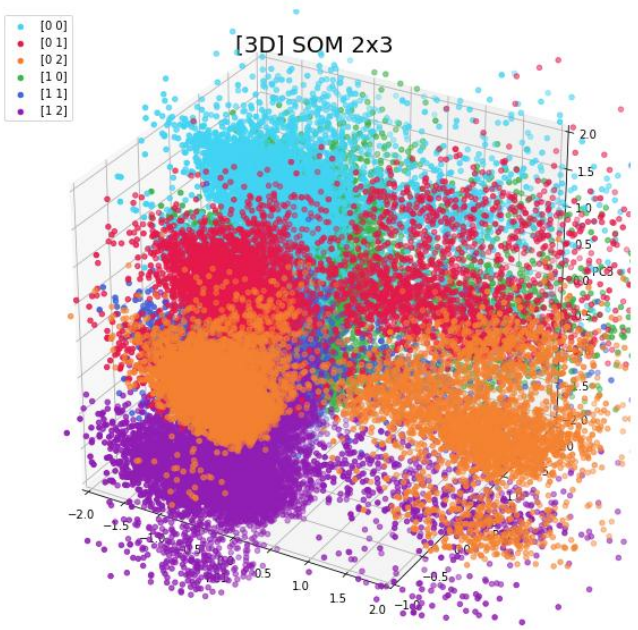
6. Clustering – clustering method

3. SOM (Self-Organizing Map)

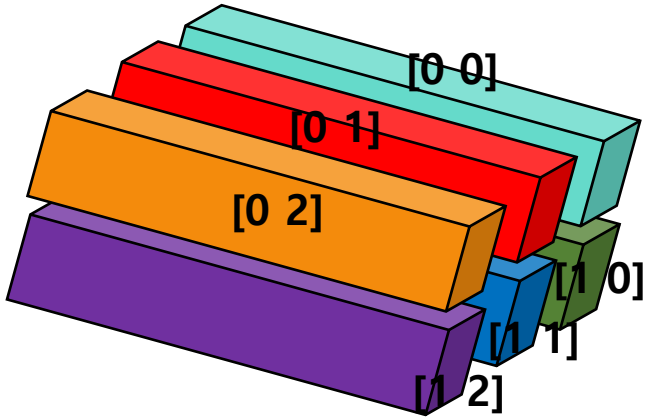
(PC1, PC2, PC3) 3차원으로 보아도, 잘 구분된 Cluster(고객군)



PC1 & PC2로 2차원 시각화



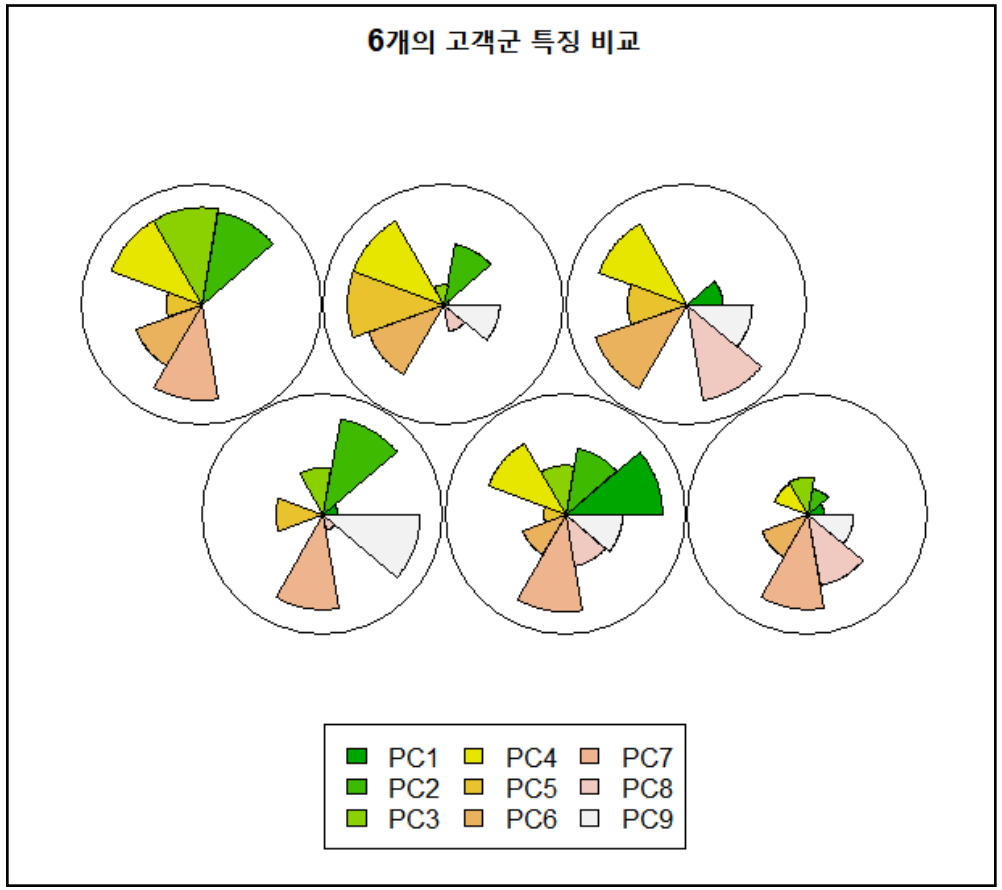
총 6개(2x3)의 Cluster



인접한 Cluster끼리 비슷한 특징 (Feature)를 가짐을 알 수 있다

PC1 & PC2 & PC3으로 3차원 시각화

6. Clustering – clustering method



SOM 방법을 통한 clustering의 장점

"고객 군 간의 유사 정도" 파악 가능!

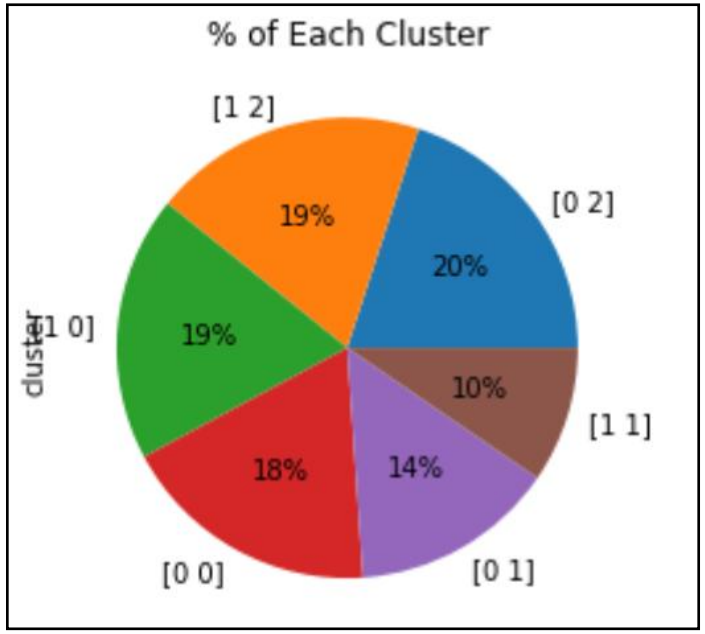
(추후 은행이고객이 더 늘어난 이후에)

유사한 고객 군끼리 공동의 전략을 세우는 것도 가능!

부채꼴 하나는 '하나의 특징' & 부채꼴 크기는 '해당 특징의 정도'

(인접한 cluster끼리 유사한 특징을 지님을 알 수 있다!)

6. Clustering – cluster 간의 비교



(일부 변수에서 눈에 띄는 차이를 보이는 6개의 고객 군)

cluster	age	balance	day	duration	campaign	pdays	previous
[0 0]	36.7	1358.9	16.0	243.2	2.6	12.8	0.3
[0 1]	34.5	1117.8	15.1	243.1	2.2	56.1	0.9
[0 2]	34.5	902.2	15.2	255.2	2.3	99.1	0.9
[1 0]	52.7	1315.5	17.1	233.6	2.8	13.5	0.3
[1 1]	45.9	964.2	16.7	214.9	2.6	11.3	0.2
[1 2]	42.1	731.7	14.6	236.3	2.4	19.9	0.2

앞에서 했던 classification 과정을,

6개의 고객 군 각각에 다시 반복하여 적용

6. Clustering – cluster 간의 비교

Light GBM + (5-fold CV) Random Search로 Hyperparameter tuning

Cluster 구분 하기 전…

Train Accuracy : 93.7%

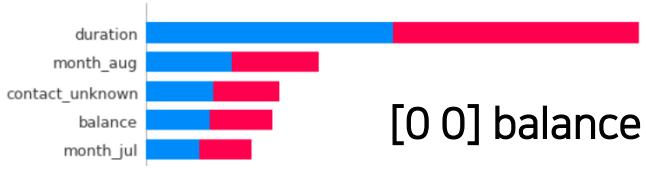
Test Accuracy : 90.5%

Cluster	[0 0]	[0 1]	[0 2]	[1 0]	[1 1]	[1 2]
Train Accuracy (%)	89.8	89	95.5	89.1	94.7	96.8
Test Accuracy (%)	83.5	86	92.5	87.6	93.9	95.9

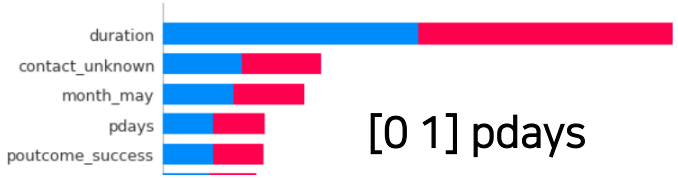
1. Train & Test 정확도 비교

6. Clustering – cluster 간의 비교

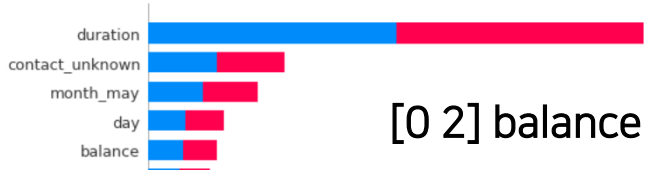
고객 군 마다, "정기 예금 유도"하는 방안을 다르게!



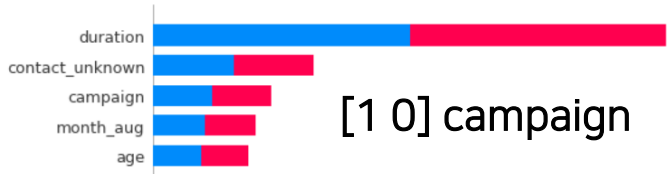
[0 0] balance



[0 1] pdays

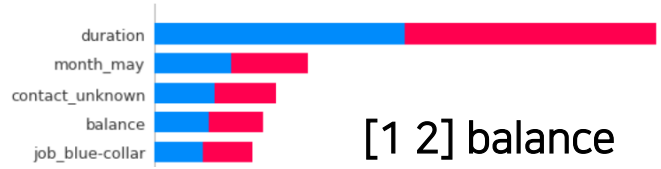


[0 2] balance



[1 0] campaign

[1 1] balance



[1 2] balance

모델 해석

어느 고객 군이든, 정기예금 가입에 있어서 'duration'이 가장 중요한 요소로 들어났다.

하지만, 그 이후의 중요한 요소에서는 군집 별로 차이를 보였다.

2. SHAP value 비교

7. 결론

Suggestion to the Bank

[보편적 전략]	[선택적 전략]
모든 고객 군에게 동일한 전략	고객 군 별로 차별화된 전략

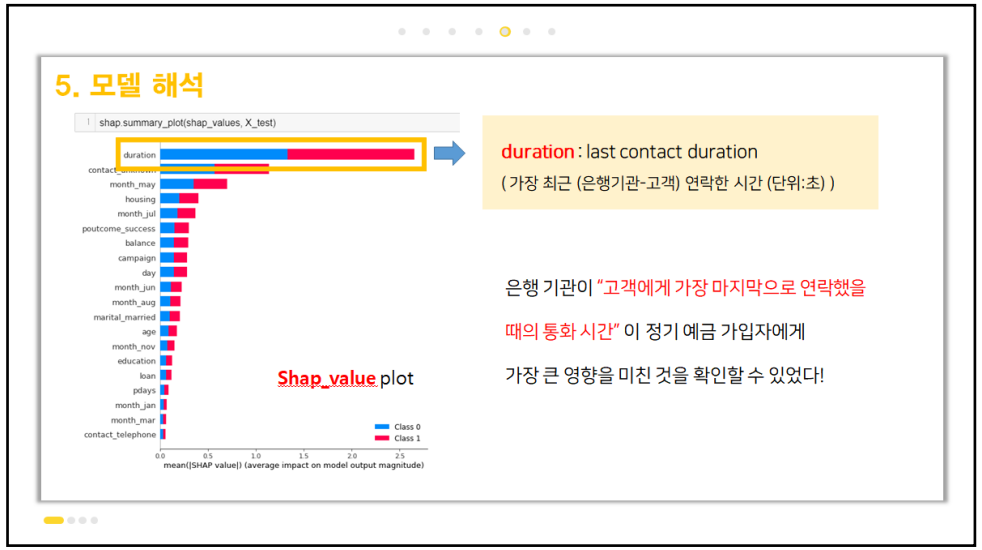
7. 결론

Suggestion to the Bank

[보편적 전략]	[선택적 전략]
모든 고객 군에게 동일한 전략	고객 군 별로 차별화된 전략

7. 결론

"모든 고객"으로 만든 모델의 해석



[duration]

과거 통화 시간을 확인하여, **"통화 시간이 짧았던 사람들"**을 대상으로 보다 집중적인 정기예금가입 유도 MKT 전략 세우기!

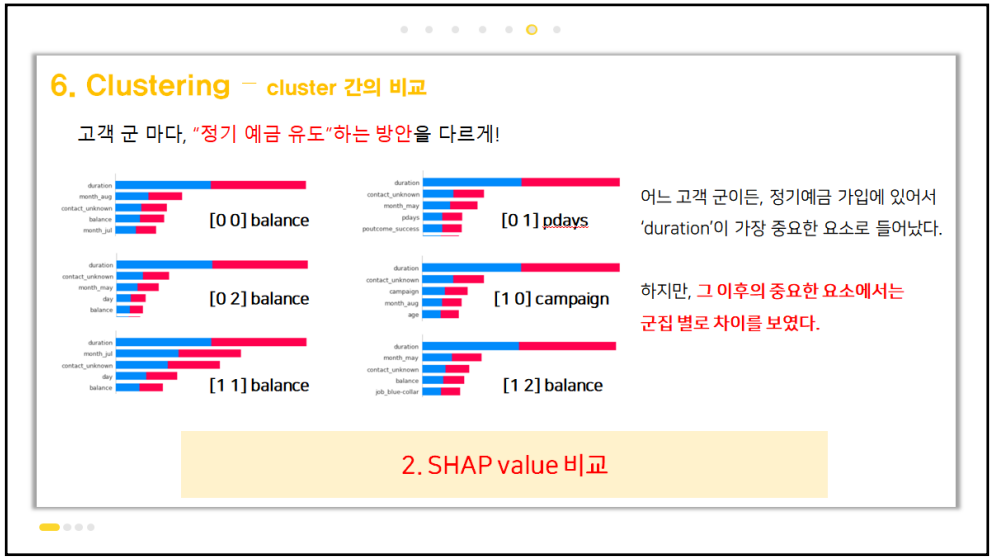
7. 결론

Suggestion to the Bank

[보편적 전략]
모든 고객 군에게 동일한 전략

[선택적 전략]
고객 군 별로 차별화된 전략

7. 결론



고객군 [0 0] [0 2] [1 1] [1 2]

은행 잔고 [Balance]

은행 잔고가 많지만, 정기 예금에 가입하지 않은
고객들을 대상으로 집중 MKT을 실시!

고객군 [0 1]

캠페인 후 경과 일수 [Pdays]

캠페인 직후, 가장 먼저 관리를 해야 할 고객 군.
빠른 시일 내에 정기예금 관련 정보 메시지를 전송!

고객군 [1 0]

캠페인 [Campaign]

캠페인에 영향을 가장 많이 받으면서 ,연령대가 가장 높은(52.7세)
고객군. 노후 대비 관련하여 정기예금 가입 유도 캠페인

8. 느낀점

After Project

데이터 분석 결과를 기반으로 고객군별로 마케팅 방식에 차별성을 둘 수 있다는 것이 흥미로웠다.
포르투갈의 낮은 저축률의 원인을 데이터를 통해 이해할 수 있었고, 이를 해결함으로써 국가 경제의 지속 가능성을 높일 수 있다는 것이 매력적이었다.

이전까지 마케팅 전략에 대해 생각할 때, 주로 사회적 흐름이나 논리적인 생각으로만 타켓팅을 하고 마케팅 전략을 짰다. 그런데 이번에 실행한 데이터 분석을 통해 정량적인 근거를 바탕으로 마케팅 전략을 짤 수 있는 것을 배웠고, 이를 현실에서 잘 활용하면 더 효과적인 마케팅 전략을 구성할 수 있을 것이라고 생각했다.

데이터 분석, 마이닝에 대한 기초지식도 없는 상황에서 걱정을 많이 하면서 시작했지만 이 수업을 수강하면서 다양한 기법과 마인드 셋을 익혀가는 스스로의 모습에 뿌듯함을 느꼈다. 데이터 해석이 점점 중요한 시대에서 기초 지식이지만 실제적인 데이터를 다양하게 다루어보고 마케팅에도 적용할 기회를 가질 수 있어서 좋았다.

추후 가능 분석

지역에 따라 고객들의 유의미한 금융생활의 차이가 나올 수도 있다고 생각.
고객들의 지역 관련 데이터가 있다면, 보다 정교한 고객군 별 분석이 가능할 것!

Thank you。