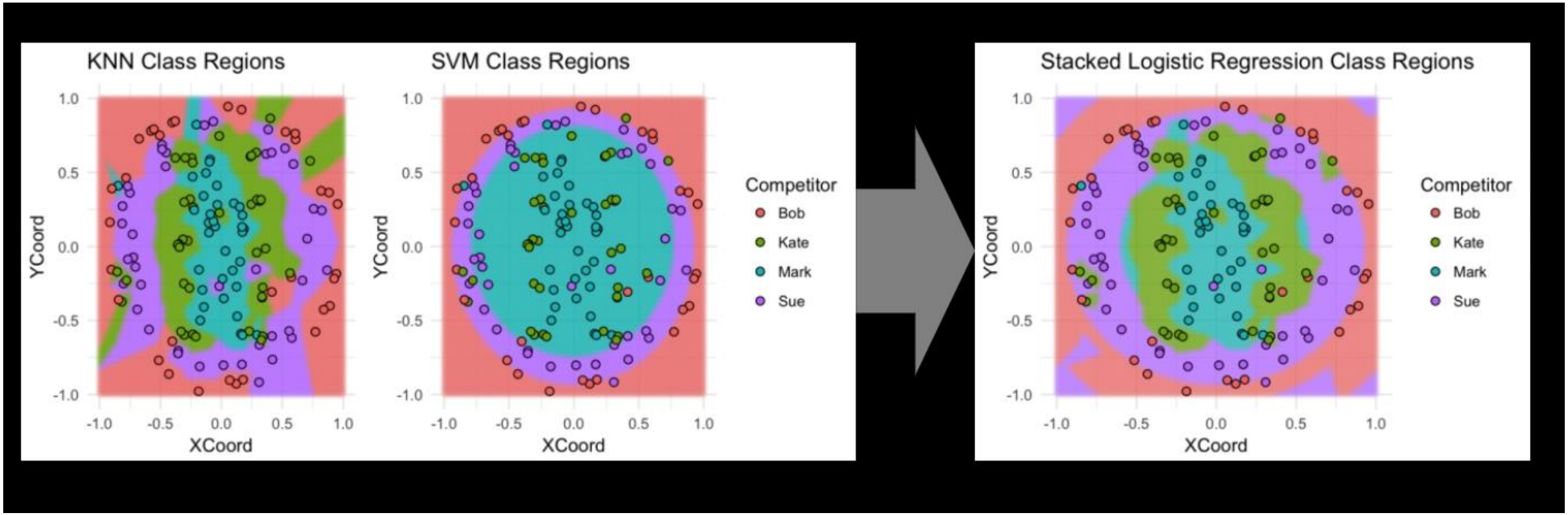
Stacking

2018.12.23 박이삭

- Staking or Meta Ensembling



Kaggle 공식 홈페이지에 있는 수상자 리뷰 중 Stacking 관련 인터뷰를 참고
링크

- Why Staking

  - 좀 더 나은 성능을 위해 ★

  - 여러 모델을 Stack해서 어떤 이점이 있는가?

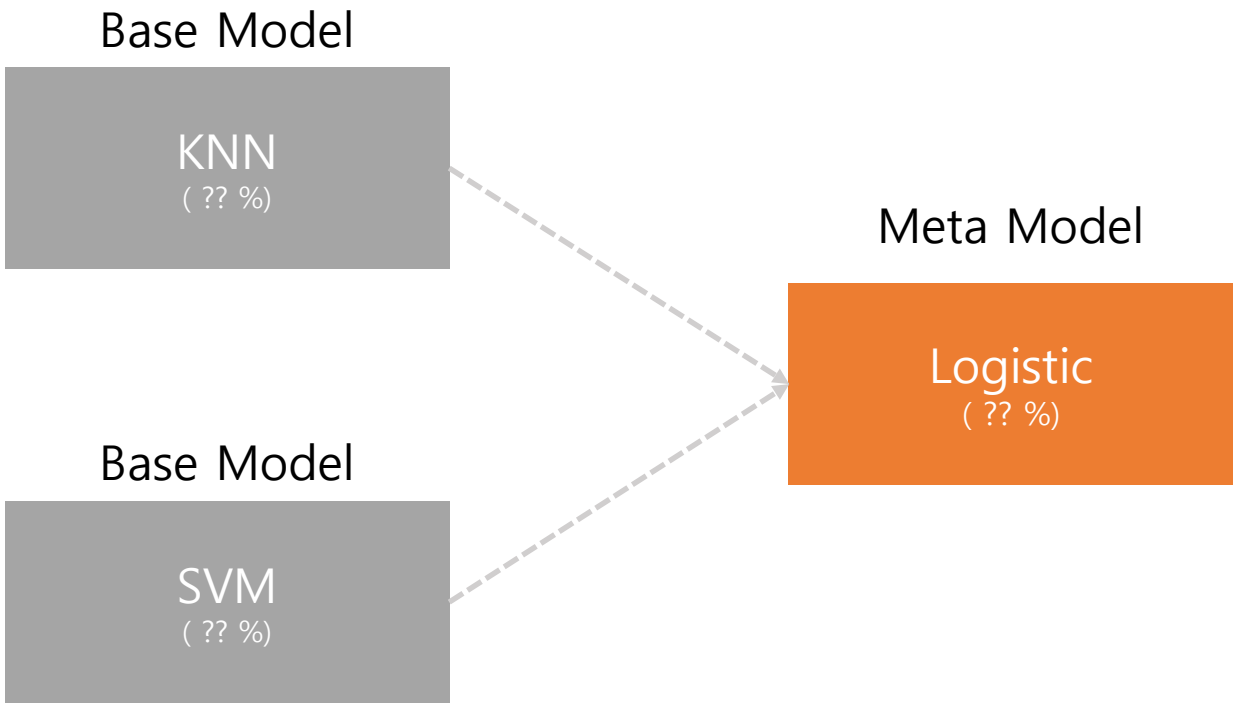
    Q1. 단일 모델을 좀 더 잘 만들면 되는 것이 아닌가?

    A1. 거의 모든 모델은 mistake를 유발한다. 또한 모델마다 가지고 있는 장단점이 달라서,
    데이터를 보는 관점이 다르다. 따라서, 여러 모델을 통합할 때 더 좋은 결과를 유도 할 수 있다.

    Q2. 여러 모델을 만들면서 증가되는 계산 복잡도는?

    A2. 좀더 나은 성능을 위해서 잠시만 접어두자.

- Stack structure



- Stacking의 구조는 다음과 같이 생겼음
- 우선, Base Model을 만들어야 한다.

→ Hyper parameter를 가지는 Base Model를 튜닝한다

- Base Model 만들기

Train Data			
	x1	x2	Y
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

KNN 모델 사용

	x1	x2	Y
1			
2			
	x1	x2	Y
3			
4			
	x1	x2	Y
5			
6			
	x1	x2	Y
7			
8			
	x1	x2	Y
9			
10			

for i in 1:10{

	x1	x2	Y
1	Test		
2			
3	Train		
4			
5			
6			
7			
8			
9			
10			

}

Test Data		
	x1	x2
1		
2		
3		
4		

Cross-Validation으로 최적의 Hyper parameter를 찾는다.

- Base Model 만들기

Train Data			
	x1	x2	Y
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

KNN 모델 사용

	x1	x2	Y
1			
2			
	x1	x2	Y
3			
4			
	x1	x2	Y
5			
6			
	x1	x2	Y
7			
8			
	x1	x2	Y
9			
10			

for i in 1:10{

	x1	x2	Y
1	Train		
2			
3	Test		
4			
5	Train		
6			
7			
8			
9			
10			

}

Test Data		
	x1	x2
1		
2		
3		
4		

Cross-Validation으로 최적의 Hyper parameter를 찾는 중.

- Base Model 만들기

Train Data			
	x1	x2	Y
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

KNN 모델 사용

	x1	x2	Y
1			
2			
	x1	x2	Y
3			
4			
	x1	x2	Y
5			
6			
	x1	x2	Y
7			
8			
	x1	x2	Y
9			
10			

for i in 1:10{

	x1	x2	Y
1	Train		
2			
3			
4			
5	Test		
6			
7	Train		
8			
9			
10			

}

Test Data		
	x1	x2
1		
2		
3		
4		

Cross-Validation으로 최적의 Hyper parameter를 찾는 중..

- Base Model 만들기

Train Data			
	x1	x2	Y
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

KNN 모델 사용

	x1	x2	Y
1			
2			
	x1	x2	Y
3			
4			
	x1	x2	Y
5			
6			
	x1	x2	Y
7			
8			
	x1	x2	Y
9			
10			

for i in 1:10{

	x1	x2	Y
1	Train		
2			
3			
4			
5			
6			
7	Test		
8			
9	Train		
10			

}

Test Data		
	x1	x2
1		
2		
3		
4		

Cross-Validation으로 최적의 Hyper parameter를 찾는 중...

- Base Model 만들기

Train Data			
	x1	x2	Y
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

KNN 모델 사용

	x1	x2	Y
1			
2			
	x1	x2	Y
3			
4			
	x1	x2	Y
5			
6			
	x1	x2	Y
7			
8			
	x1	x2	Y
9			
10			

for i in 1:10{

	x1	x2	Y
1	Train		
2			
3			
4			
5			
6			
7			
8			
9	Test		
10			

}

Test Data		
	x1	x2
1		
2		
3		
4		

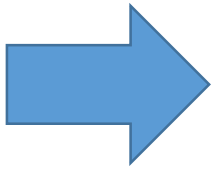
Cross-Validation으로 최적의 Hyper parameter를 찾는 중....

- Base Model 만들기

Train Data			
	x1	x2	Y
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			

KNN 모델 사용

	x1	x2	Y
1			
2			
	x1	x2	Y
3			
4			
	x1	x2	Y
5			
6			
	x1	x2	Y
7			
8			
	x1	x2	Y
9			
10			



K = 3

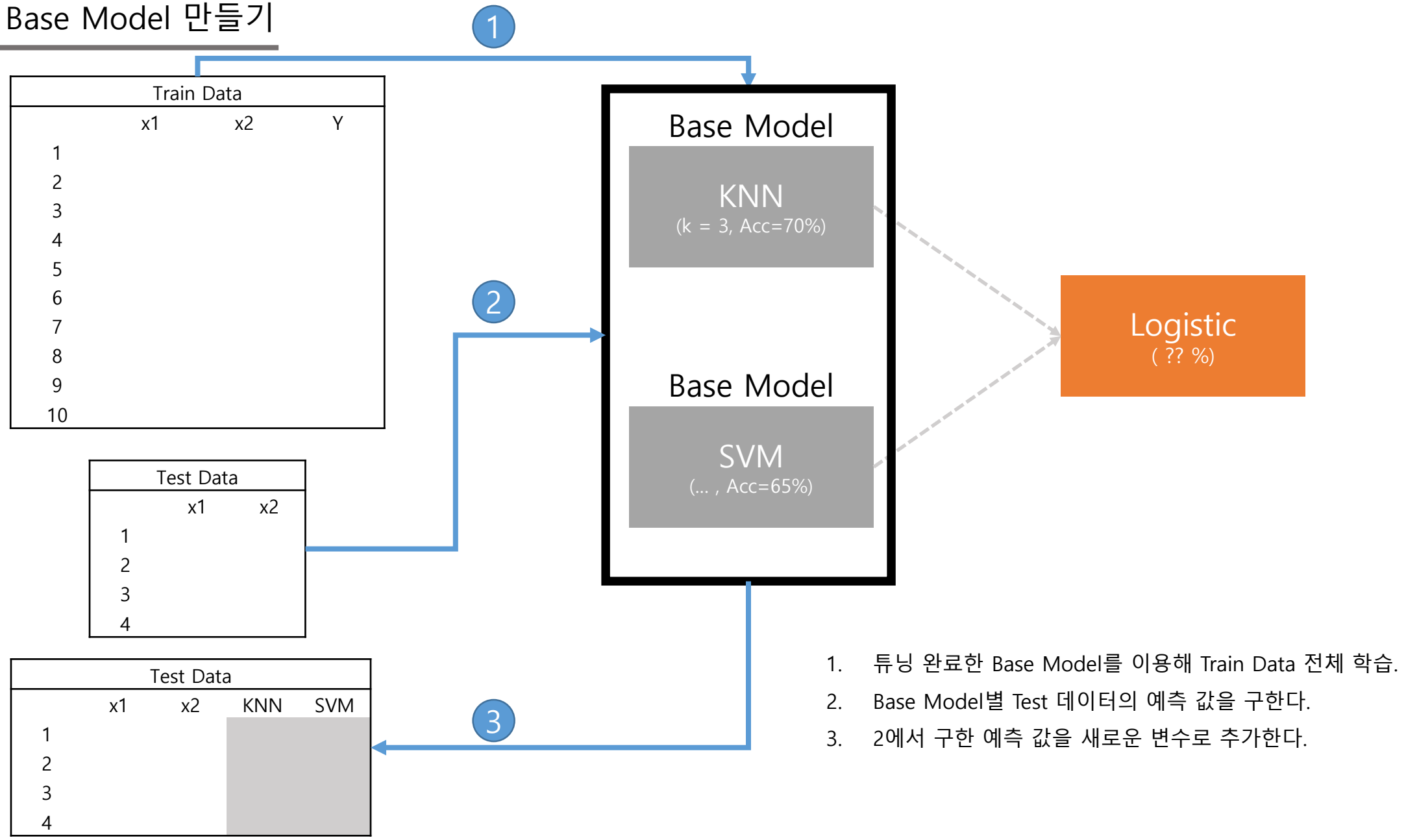
Test Data		
	x1	x2
1		
2		
3		
4		

최적의 Hyper parameter를 찾았습니다.

CV, Grid search 이용

'**Random search**' 또는 '**베이지안 최적화**'도 사용가능

- Base Model 만들기



1. 튜닝 완료한 Base Model를 이용해 Train Data 전체 학습.
2. Base Model별 Test 데이터의 예측 값을 구한다.
3. 2에서 구한 예측 값을 새로운 변수로 추가한다.

- Meta Model 만들기

Train Data					
	x1	x2	KNN	SVM	Y
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

- Test와 마찬가지로 Train에도 Meta 변수를 만들어야 한다.
- **단,** 모델 튜닝을 할 때와 마찬가지로 CV로 Meta변수를 얻는다. ⟸ 왜?

- Meta Model 만들기

Stacking의 핵심은 Base Model의 **예측 결과값**을 Meta model의 **변수**로 사용한다는 점

→ 어떤 부분에서 정확도가 높고, 낮은지 덕분에 알 수 있다.

이를 위해선,

- Meta변수를 만들 때 i번째 행을 학습하지 않고 test로 두어야 한다.
- 만약, 학습하게 된다면 데이터 누출(링크: Data leakage)로 인해 오버피팅 될 수 있다.

※ 데이터 누출 :
ML알고리즘을 훈련하기 위해 사용하는 데이터가 예측하려는 정보를 가지고 있는 상황

Train Data					
	x1	x2	KNN	SVM	Y
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					

	x1	x2	KNN	SVM	Y
1					
2					
	x1	x2	KNN	SVM	Y
3					
4					
	x1	x2	KNN	SVM	Y
5					
6					
	x1	x2	KNN	SVM	Y
7					
8					
	x1	x2	KNN	SVM	Y
9					
10					

- Meta Model 만들기

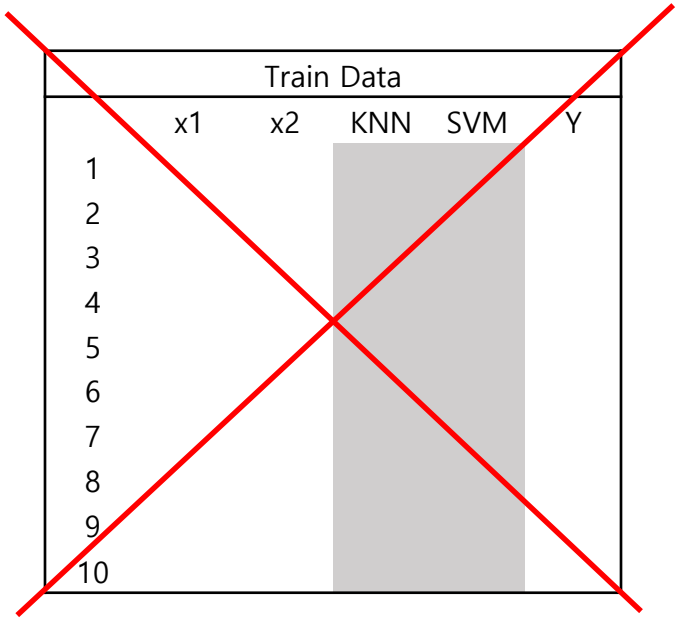
Stacking의 핵심은 Base Model의 **예측 결과값**을 Meta model의 **변수**로 사용한다는 점

→ 어떤 부분에서 정확도가 높고, 낮은지 덕분에 알 수 있다.

이를 위해선,

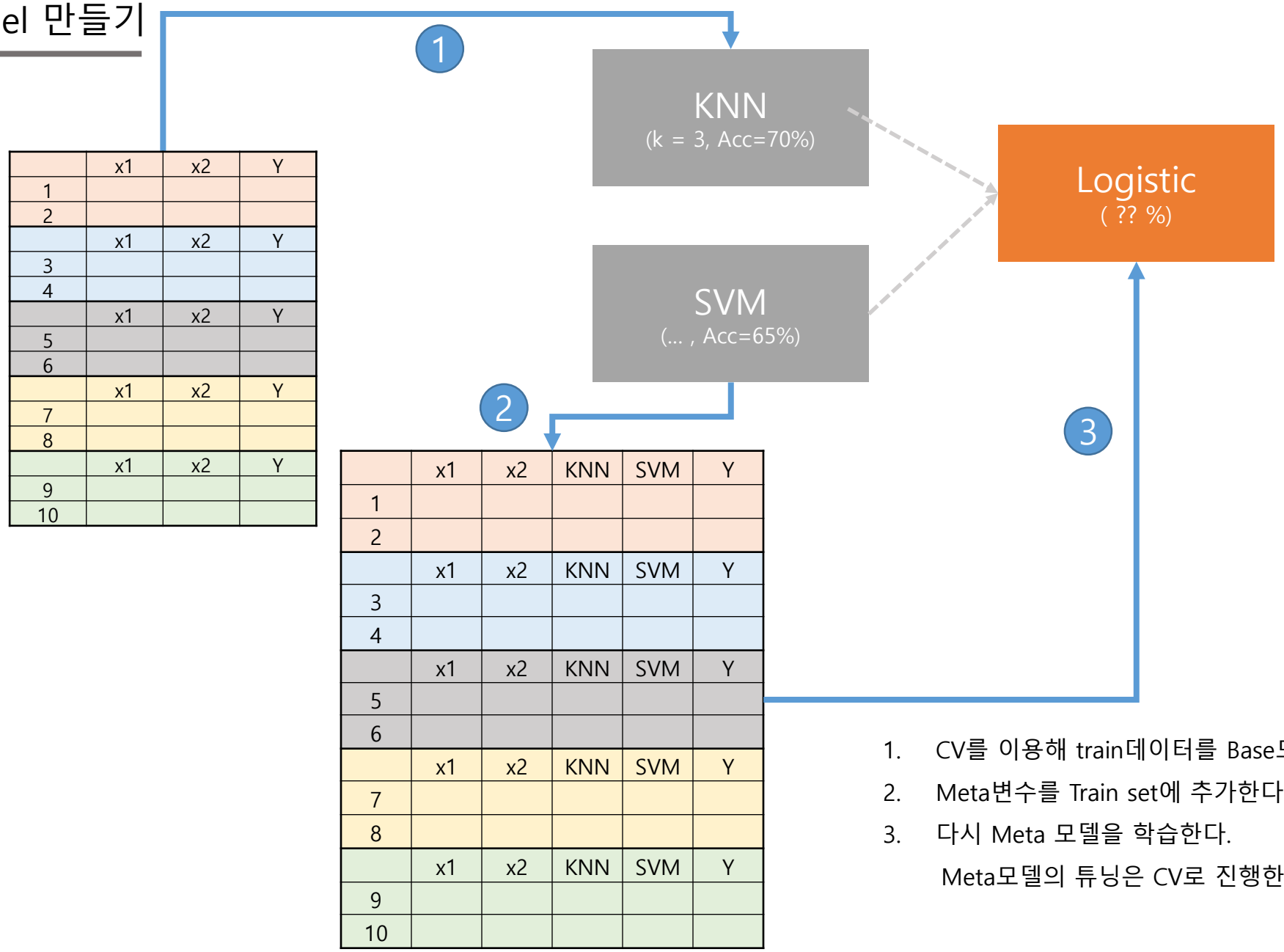
- Meta변수를 만들 때 i번째 행을 학습하지 않고 test로 두어야 한다.
- 만약, 학습하게 된다면 데이터 누출(링크: Data leakage)로 인해 오버피팅 될 수 있다.

※ 데이터 누출 :
ML알고리즘을 훈련하기 위해 사용하는 데이터가 예측하려는 정보를 가지고 있는 상황



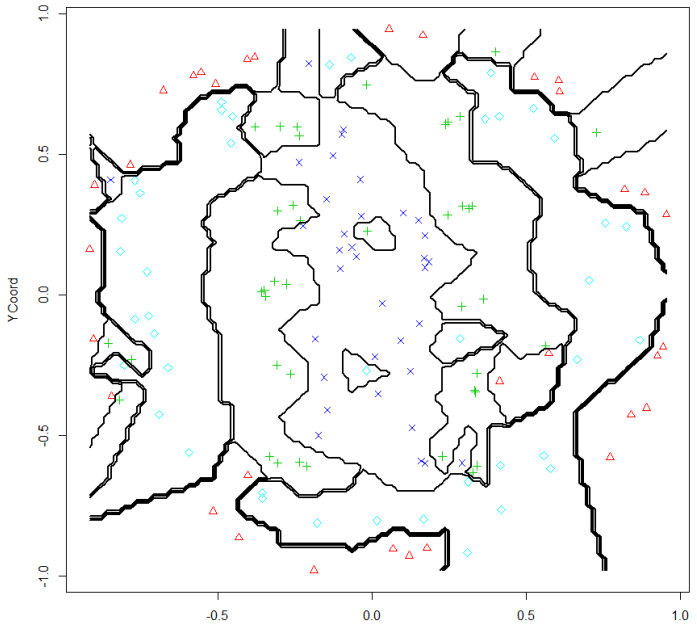
	x1	x2	KNN	SVM	Y
1					
2					
	x1	x2	KNN	SVM	Y
3					
4					
	x1	x2	KNN	SVM	Y
5					
6					
	x1	x2	KNN	SVM	Y
7					
8					
	x1	x2	KNN	SVM	Y
9					
10					

- Meta Model 만들기

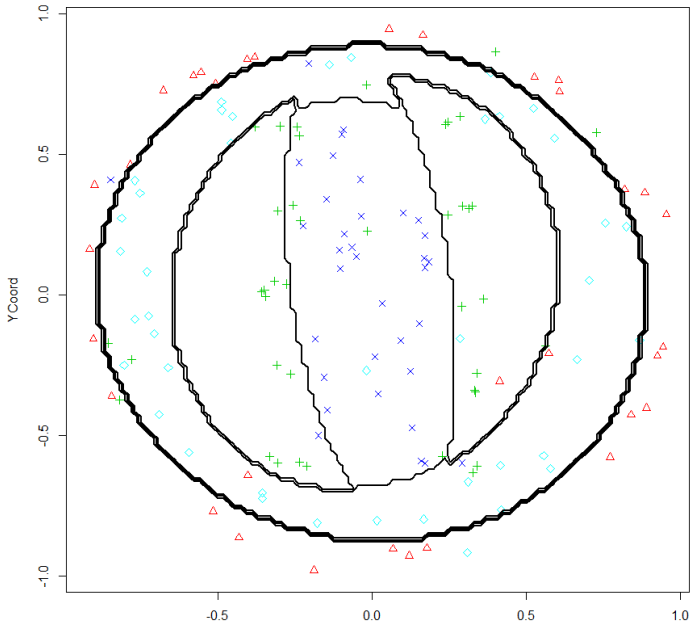


1. CV를 이용해 train데이터를 Base모델에 적합
2. Meta변수를 Train set에 추가한다.
3. 다시 Meta 모델을 학습한다.
   Meta모델의 튜닝은 CV로 진행한다.

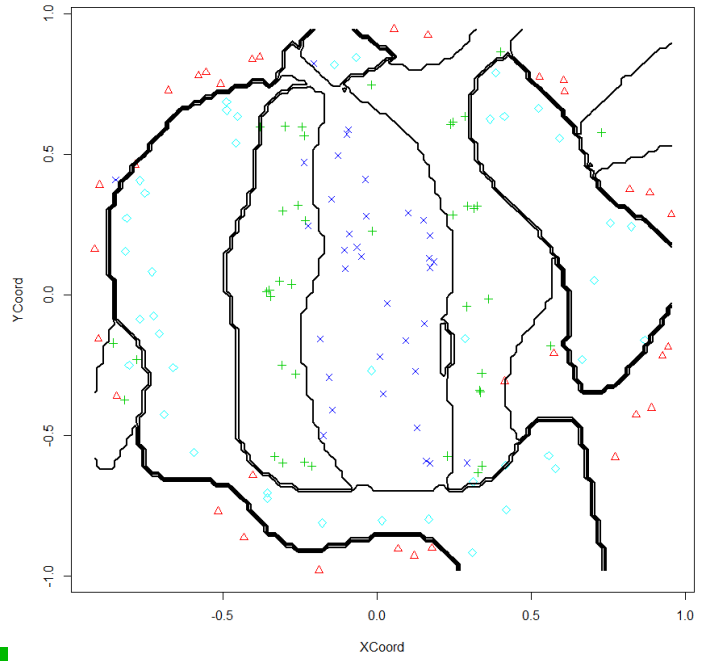
KNN(70%)



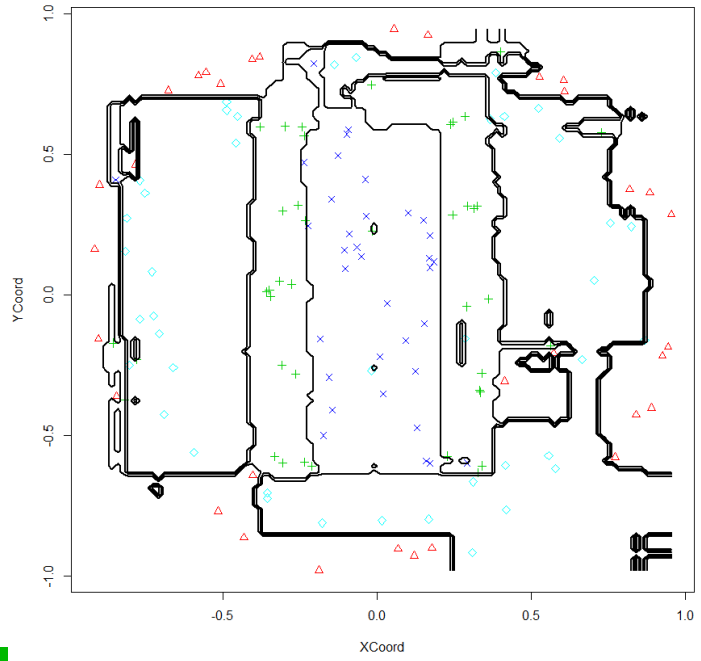
SVM_poly(86%)

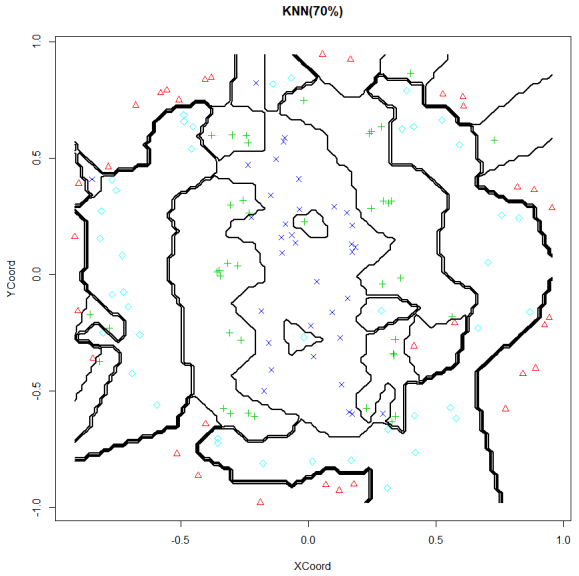


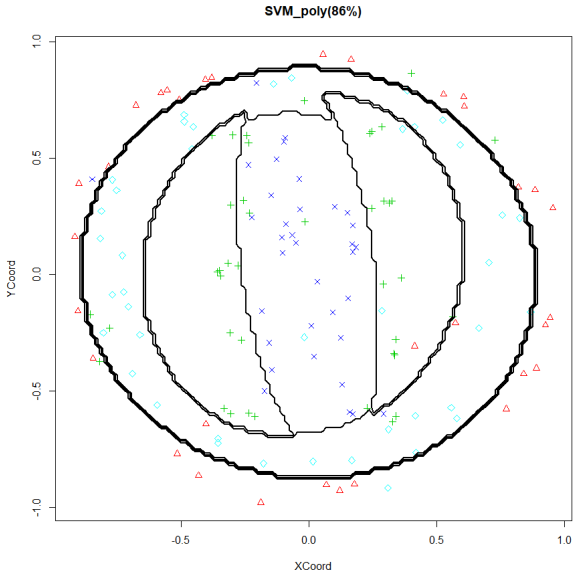
SVM_radial(75%)

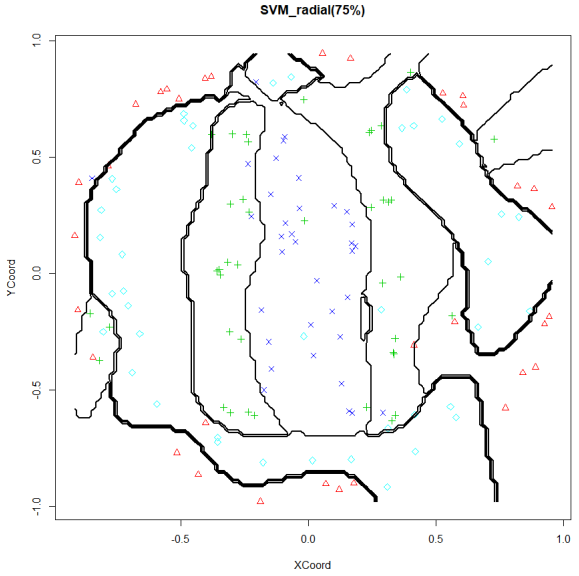


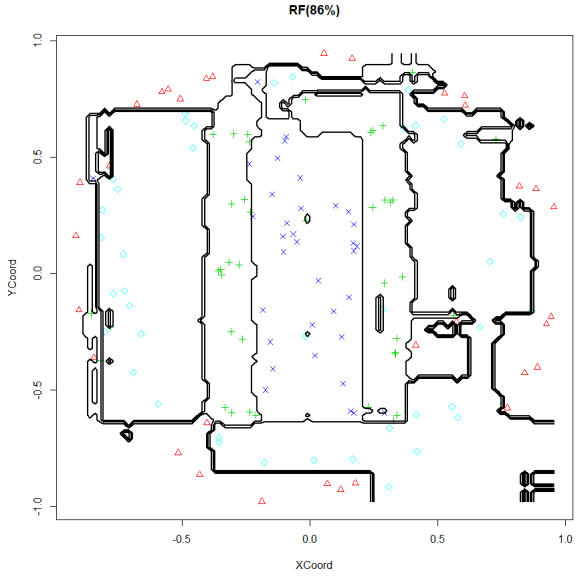
RF(86%)

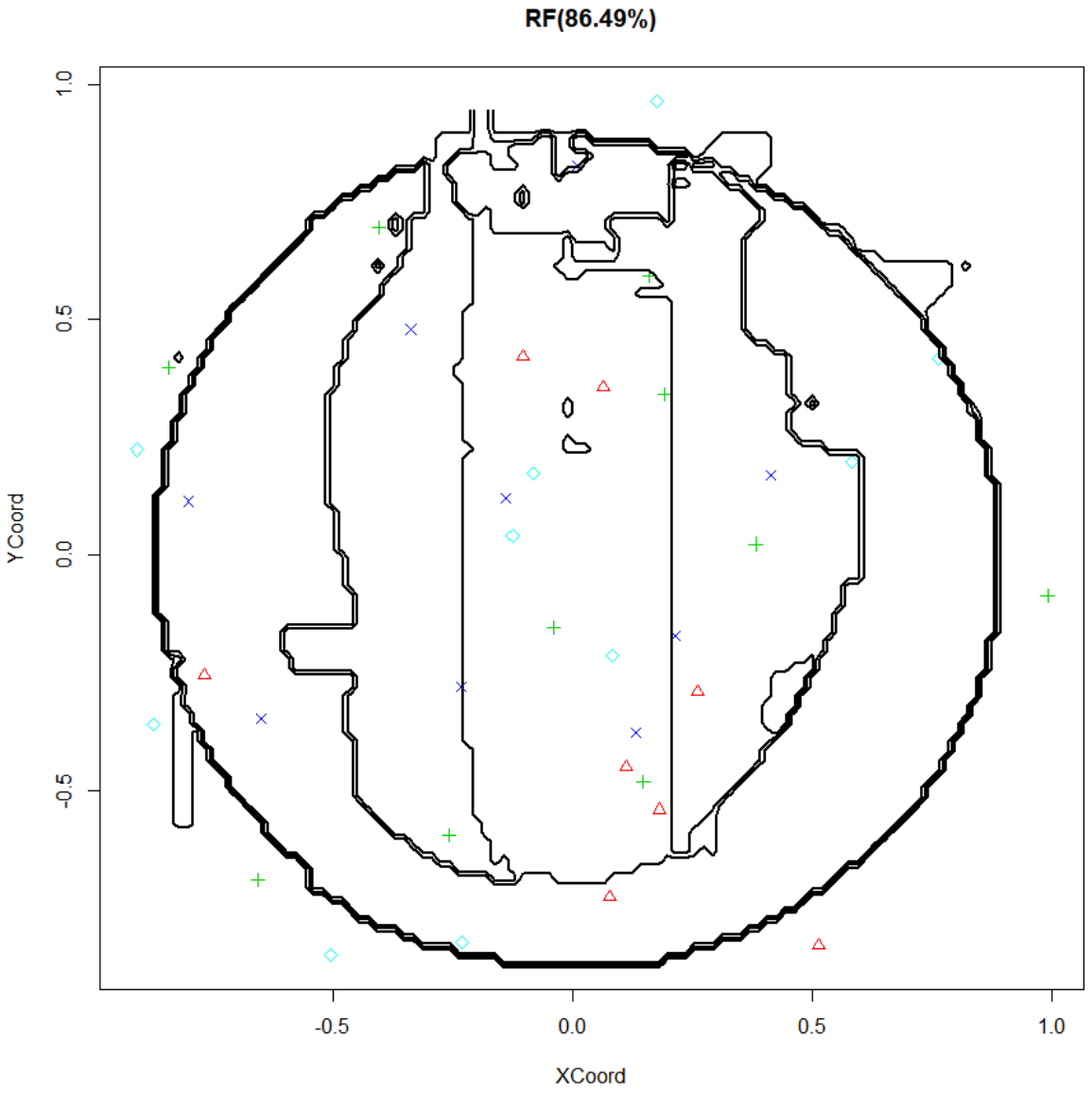


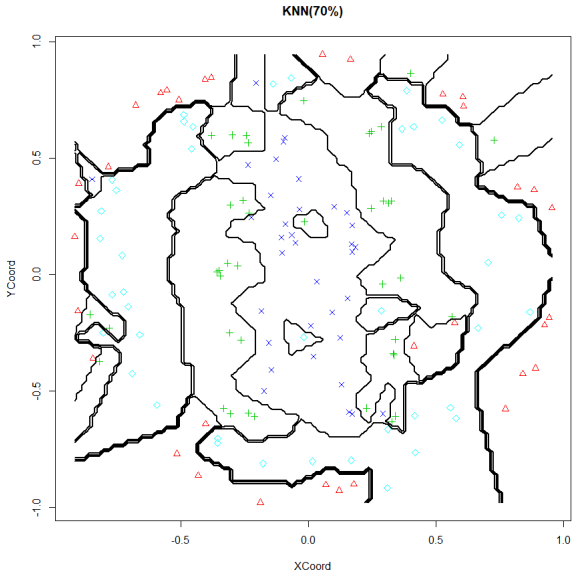

KNN(70%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0

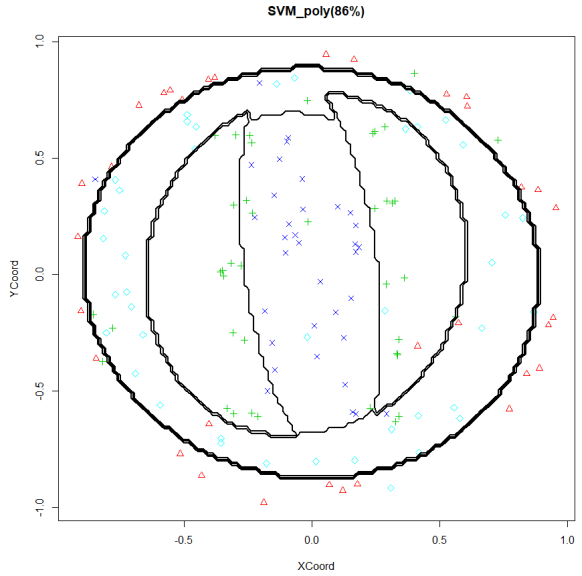

SVM_poly(86%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0

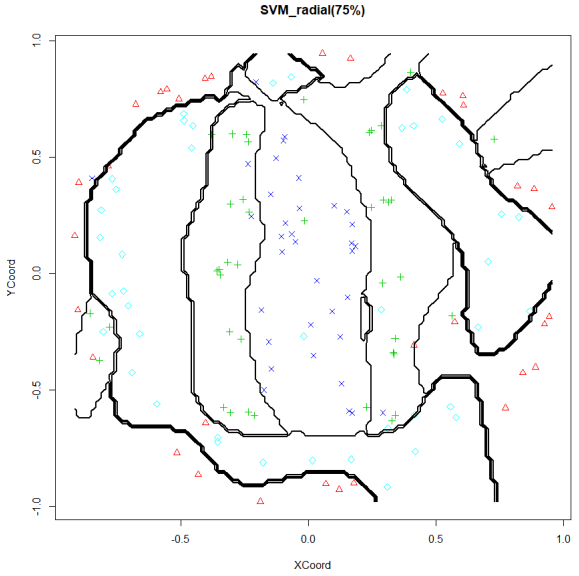

SVM_radial(75%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0

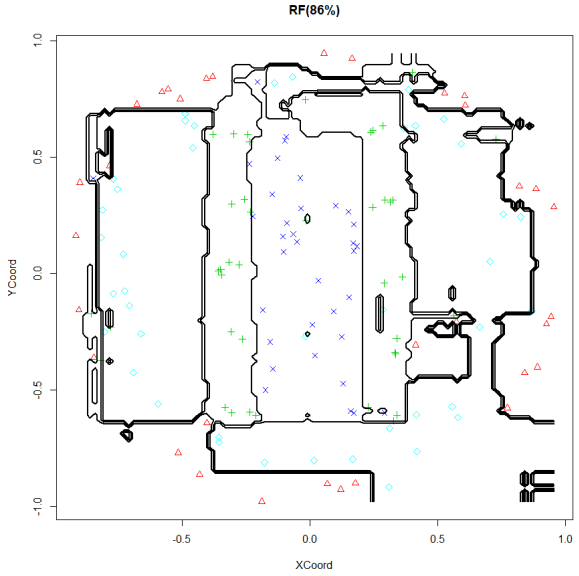

RF(86%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0

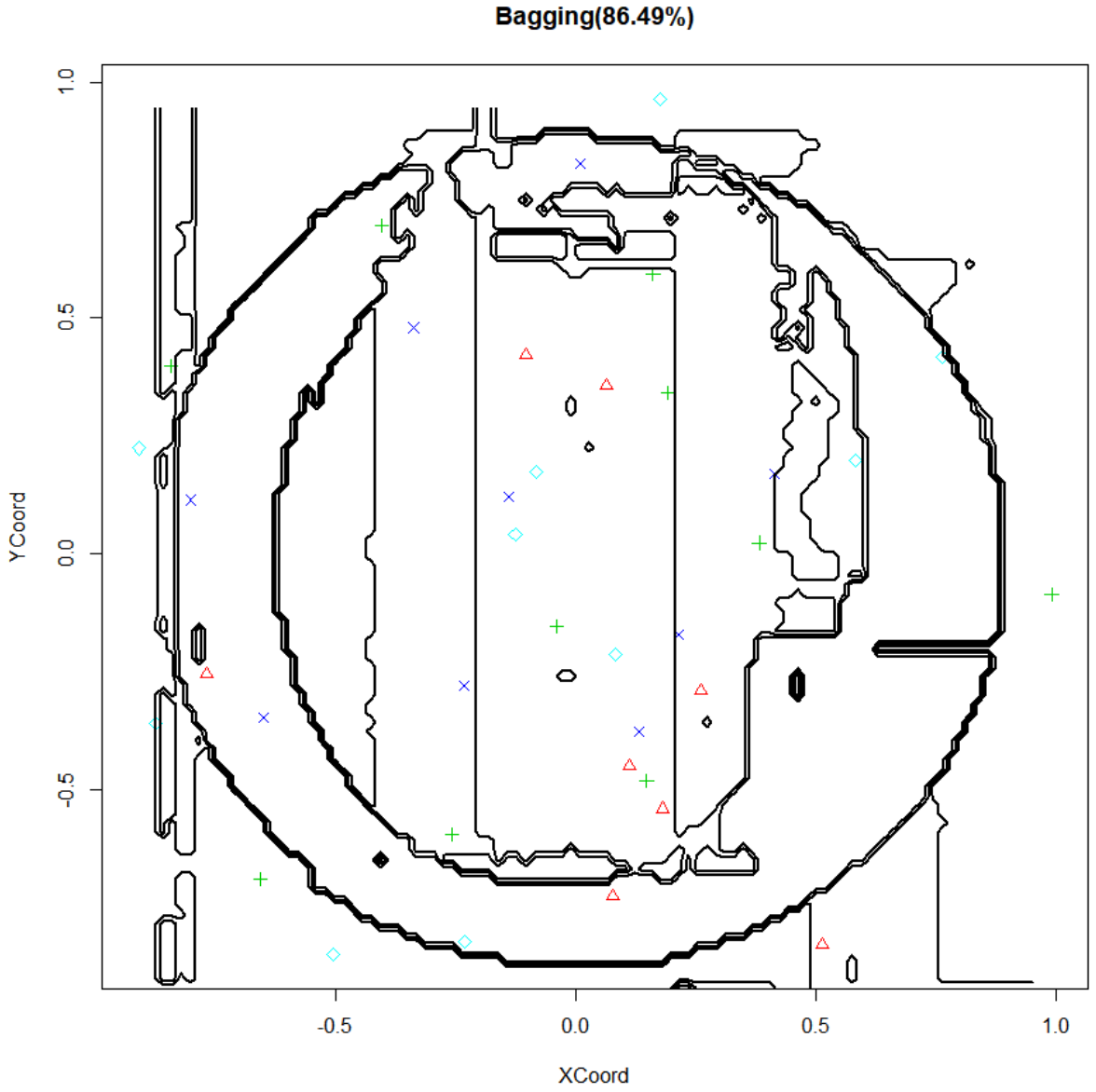

RF(86.49%)
YCoord
XCoord
1.0
0.5
0.0
-0.5

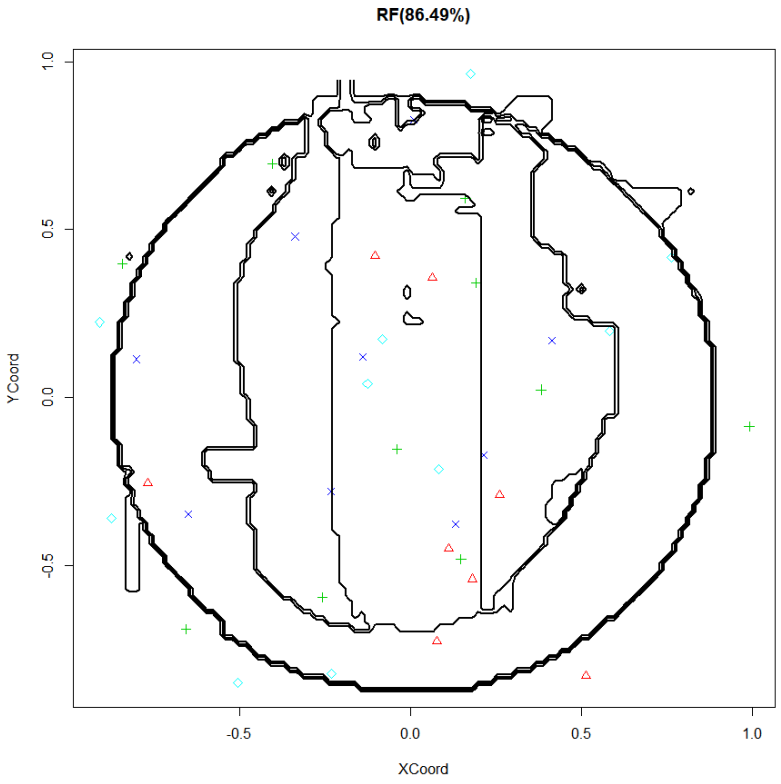

KNN(70%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0

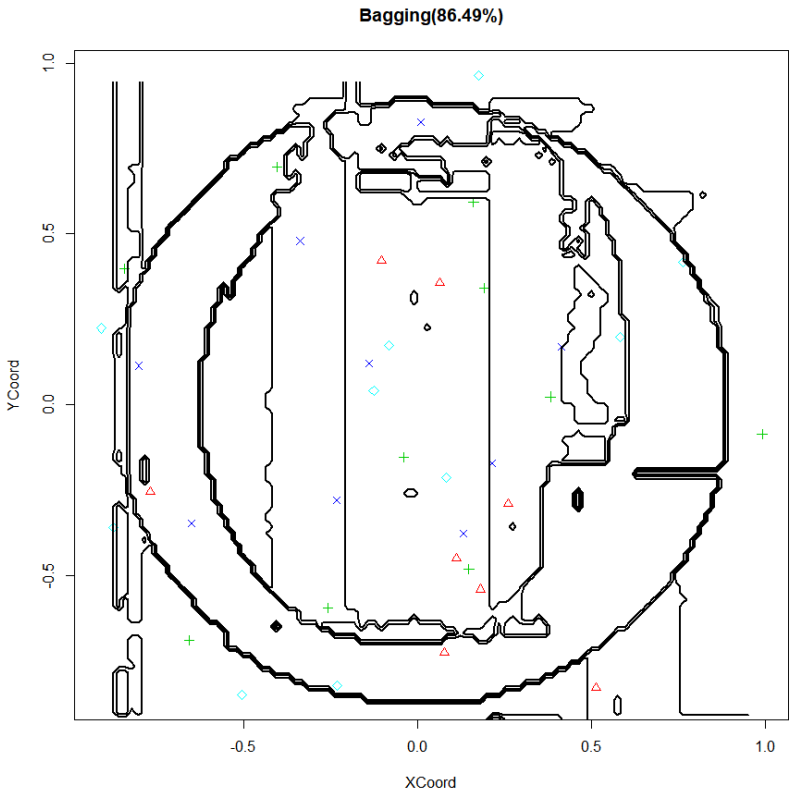

SVM_poly(86%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0

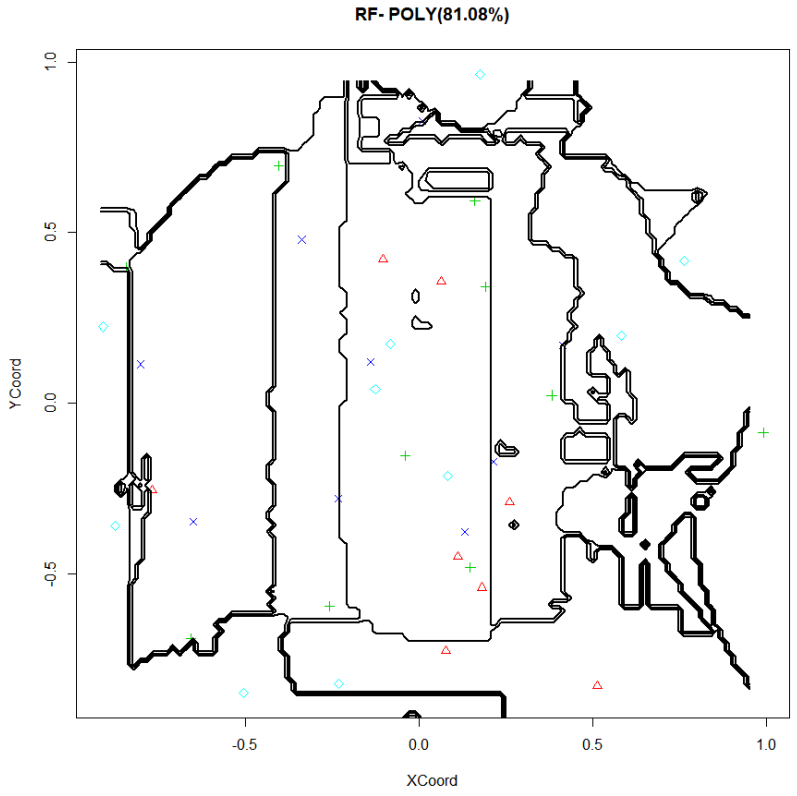

SVM_radial(75%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0


RF(86%)
YCoord
XCoord
1.0
0.5
0.0
-0.5
-1.0


Bagging(86.49%)
YCoord
XCoord
1.0
0.5
0.0
-0.5







어떤 메타 모델을 사용해야 할지는 정답이 없다.

참고자료

- http://kweonwooj.tistory.com/36
- https://www.kaggle.com/serigne/stacked-regressions-top-4-on-leaderboard#
- https://topepo.github.io/caret/train-models-by-tag.html#l1-regularization
- http://otzslayer.github.io/machine-learning/feature-selection/
- http://michael.hahsler.net/SMU/EMIS7332/R/viz_classifier.html