

강원대학교
AI 소프트웨어학과

머신러닝2
- 지도학습 -
기계학습의 회귀분석

통계적 방법과 기계학습 방법의 차이

기계학습의 방법(지도학습)

- **다중 회귀분석(Regression)**
- **로지스틱 회귀분석(Logistic Regression)**
- **신경망(Artificial Neural Network)**
- **서포트 벡터 머신(Support Vector Machine)**
- **의사결정나무(Decision Tree)**
- **앙상블(Ensemble)**
- **K-근접 이웃기법(k-Nearest Neighbor)**

로지스틱 회귀(Logistic Regression)

연속형 종속변수 경우?

- 독립변수를 바탕으로 종속변수를 예측함
- 강아지, 고양이와 같은 연속적이지 않은 범주형 변수를 어떻게 예측할까?

지점번호	홍보비용	직원수	지점성과	고객수	대출정도	유동인구	예금금액
1	40	15					2,000
2	50	20					3,500
3	30	14					4,000
4	60	22					3,500
5	70	30					4,900
6	60	24					3,100
7	30	16					4,200
8	60	20					2,300
9	20	14					6,000
10	80	32					5,000

로지스틱 회귀(Logistic Regression)

연속형 종속변수가 아닌 범주형 종속변수일 경우?

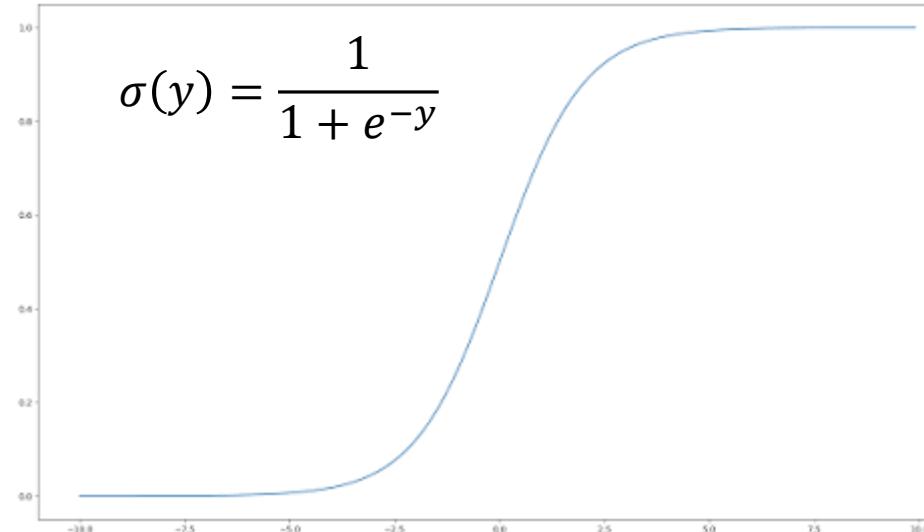
- 질병 진단(0 = 양성, 1 = 악성)
- 강아지, 고양이 판단

지점번호	홍보비용	직원수	지점성과	고객수	대출정도	유동인구	예금유지
1	40	15					0
2	50	20					1
3	30	14					1
4	60	22					1
5	70	30					1
6	60	24					1
7	30	16					0
8	60	20					1
9	20	14					0
10	80	32					1

로지스틱 회귀(Logistic Regression)

연속형 종속변수가 아닌 범주형 종속변수일 경우?

- 시그모이드 함수(Sigmoid Function) : $0 < \sigma(y) < 1$ 사이의 값으로 구성되어 확률처럼 해석하기 좋음
- 이진분류의 문제를 해결하기 위해 많이 사용
- 활성화 함수(Activation Function)으로도 많이 사용됨(Forward → Loss → Backward → Optimizer)

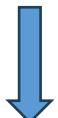


로지스틱 회귀(Logistic Regression)

연속형 종속변수가 아닌 범주형 종속변수일 경우?

- **로지스틱 회귀분석(Logistic Regression)** : 로지스틱 회귀 모델은 시그모이드 함수라고도 알려진 로지스틱 함수를 사용하여 예측값을 확률에 매핑해 0과 1로 도출함
- 질병 진단(0 = 양성, 1 = 악성)
- 강아지, 고양이 판단

$$y = \beta_0 + \beta_1 x_1 + \beta_n x_n + \cdots + \beta_n x_n \quad \rightarrow \text{연속적인 수치정보를 제공함}$$



$$\sigma(y) = \frac{1}{1 + e^{-y}} \quad \rightarrow \text{이분류 문제로 데이터 값을 변경함}$$

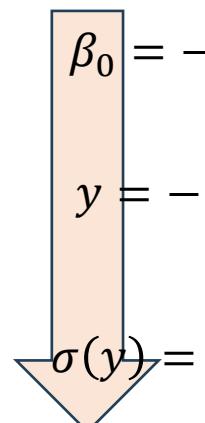
로지스틱 회귀(Logistic Regression)

연속형 종속변수가 아닌 범주형 종속변수일 경우?

- 로지스틱 회귀분석(Logistic Regression) : 로지스틱 회귀 모델은 시그모이드 함수라고도 알려진 로지스틱 함수를 사용하여 예측값을 확률에 매핑해 0과 1로 도출함
- 질병 진단(0 = 양성, 1 = 악성)
- 강아지, 고양이 판단

$$y = \beta_0 + \beta_1 x_1 + \beta_n x_n + \cdots + \beta_n x_n$$

$$\sigma(y) = \frac{1}{1 + e^{-y}}$$



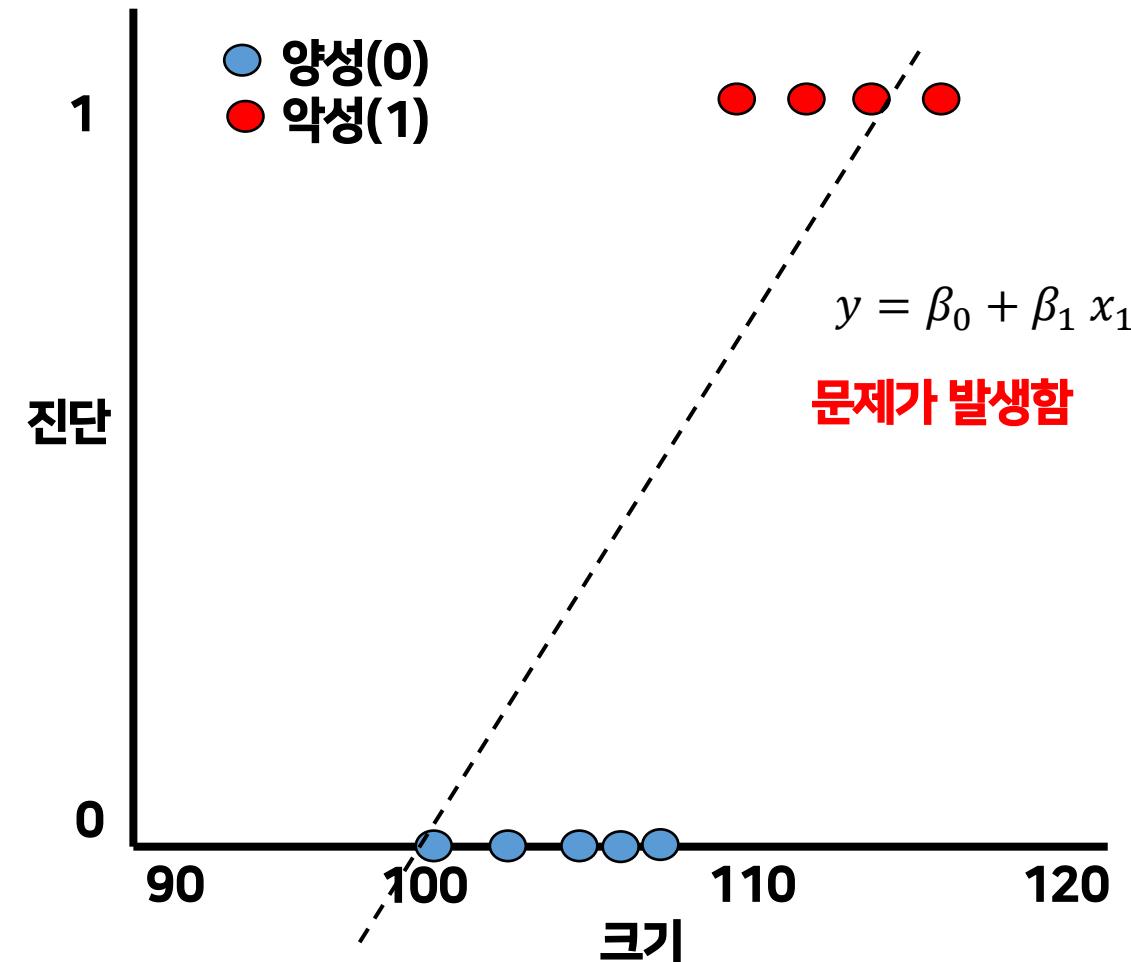
$$\begin{aligned} \beta_0 &= -1, \quad \beta_1 = 0.8, \quad x_1 = 2, x_2 = 3 \\ y &= -1 + (0.8 * 2) - (0.5 * 3) = -0.9 \\ \sigma(y) &= \frac{1}{1 + e^{-(-0.9)}} = 0.289 \end{aligned}$$

고양이 판단

로지스틱 회귀(Logistic Regression)

로지스틱 회귀

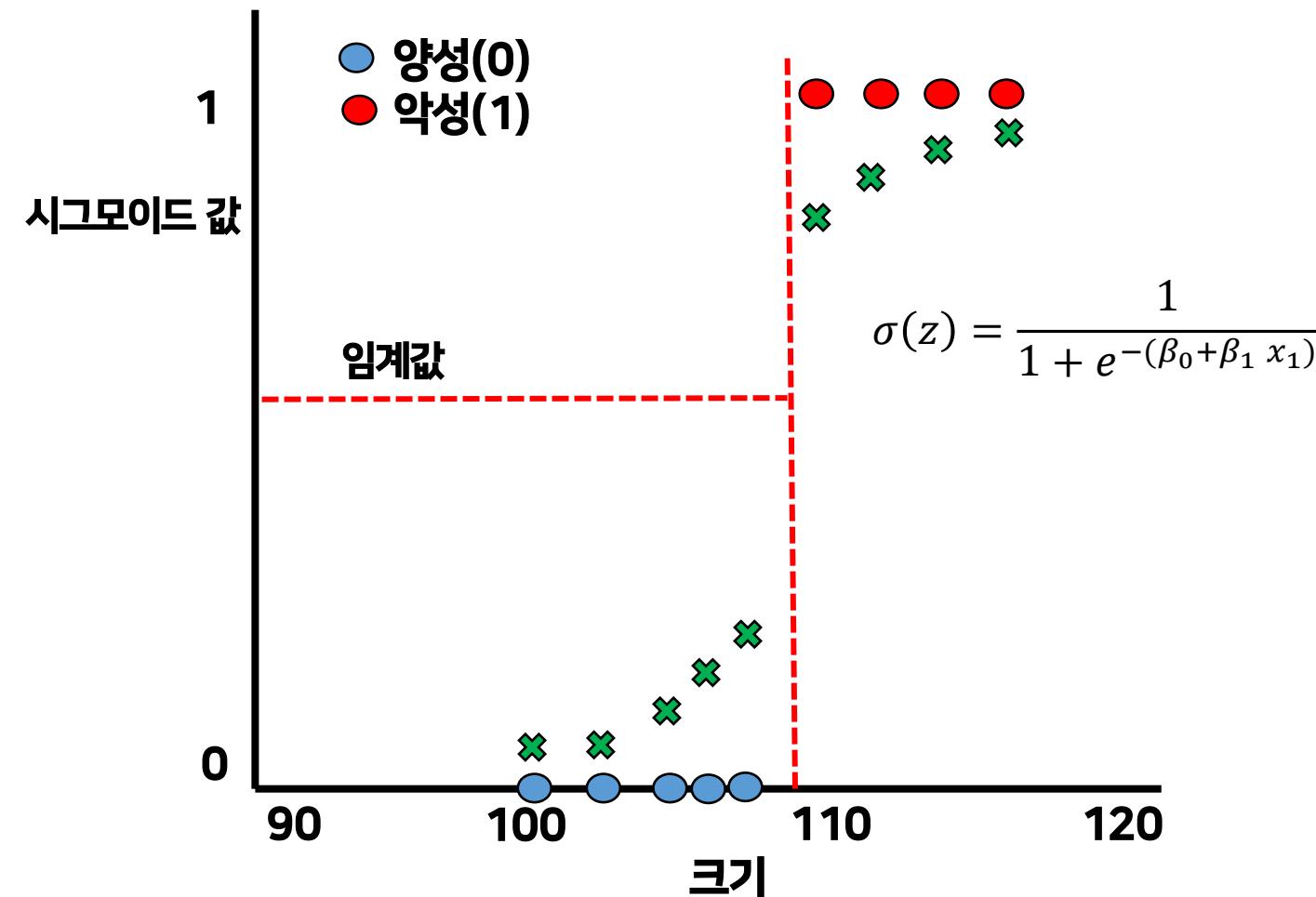
크기	진단
100	0
101	0
102	0
103	0
104	0
105	0
106	0
107	0
108	0
109	0
110	1
111	1
112	1
113	1
114	1



로지스틱 회귀(Logistic Regression)

로지스틱 회귀

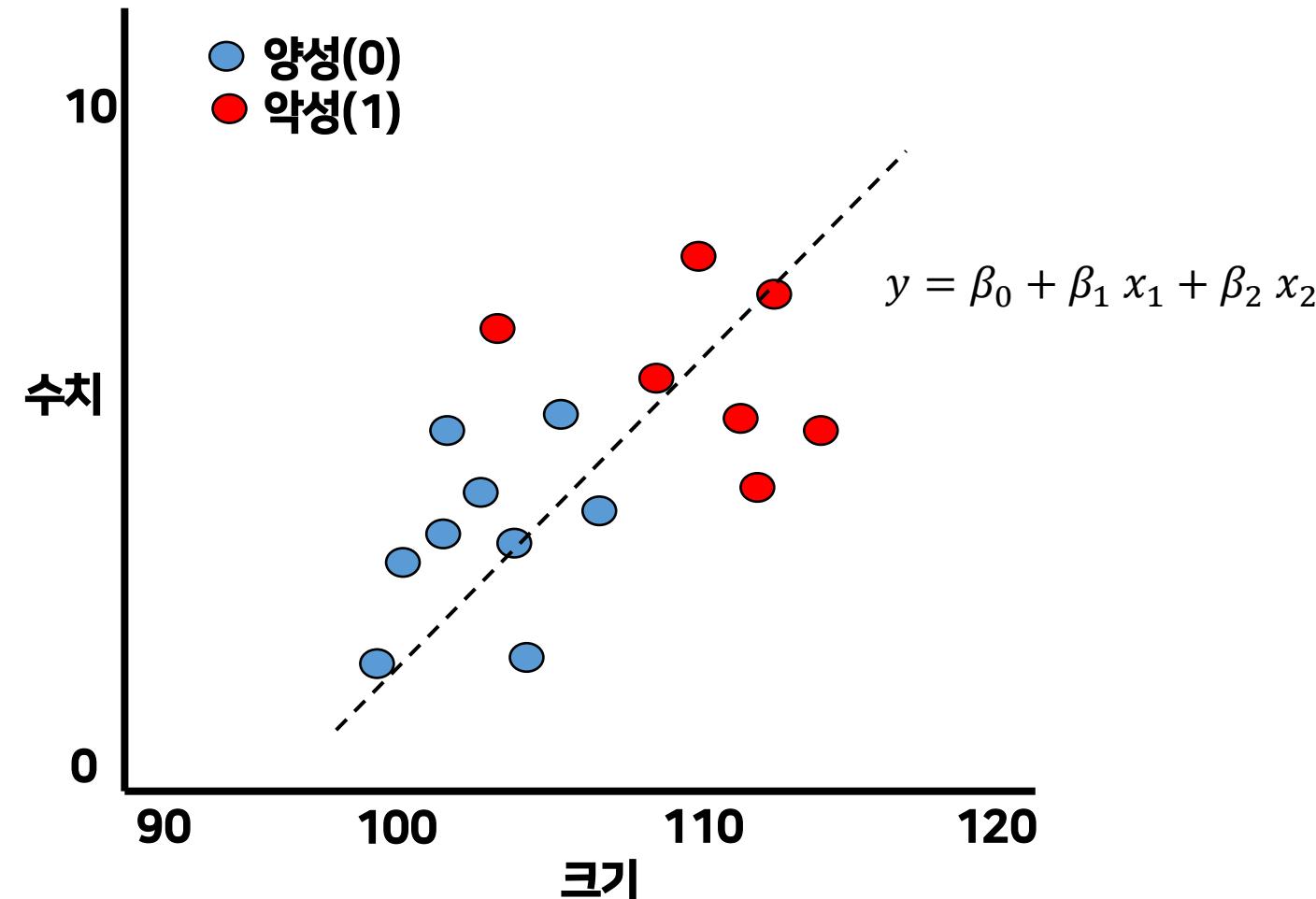
크기	예측확률	진단
100	0.005	0
101	0.007	0
102	0.009	0
103	0.01	0
104	0.02	0
105	0.05	0
106	0.1	0
107	0.4	0
108	0.5	0
109	0.6	0
110	0.65	1
111	0.7	1
112	0.75	1
113	0.8	1
114	0.9	1



로지스틱 회귀(Logistic Regression)

로지스틱 회귀

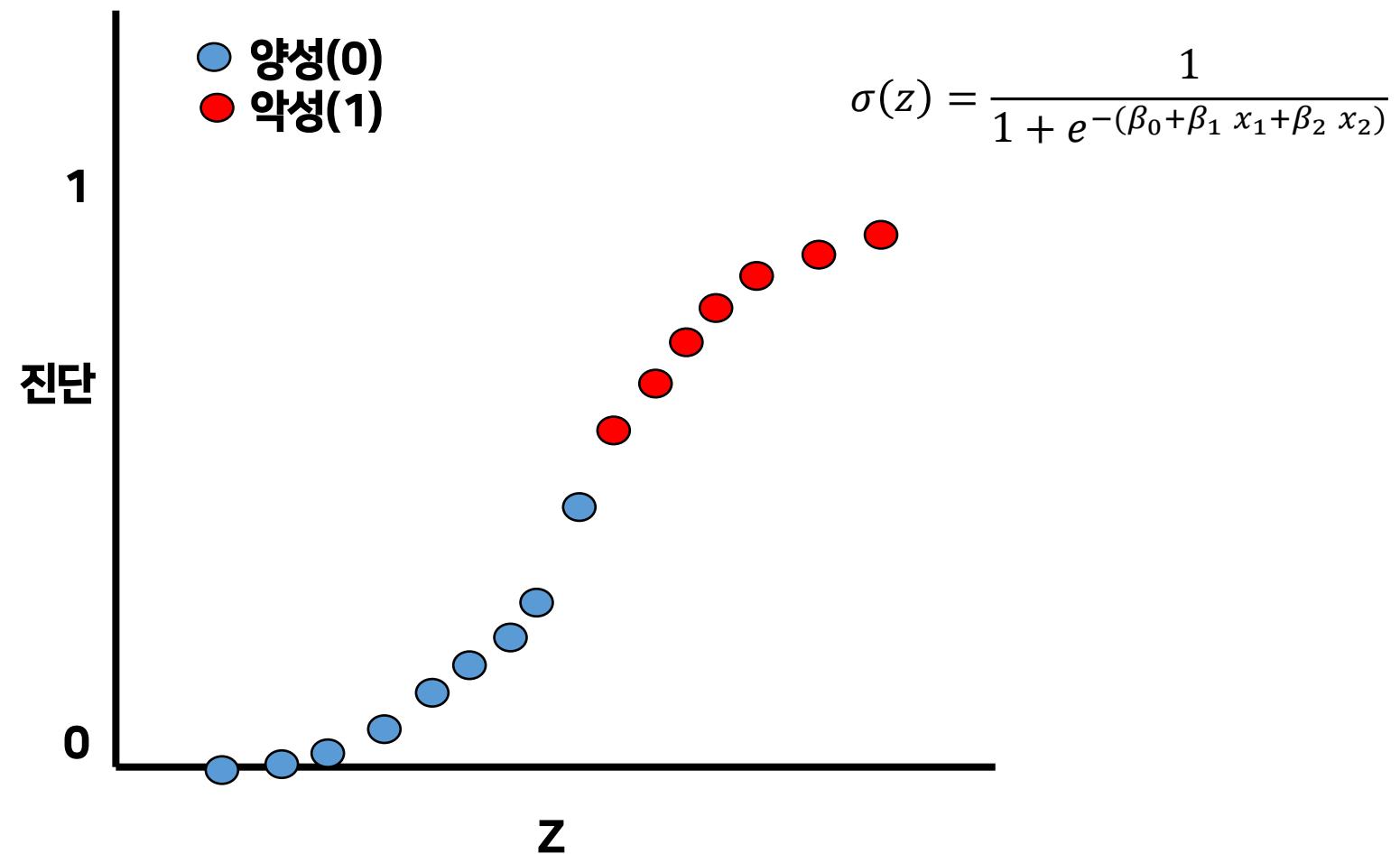
크기	수치	진단
100	5	0
101	6	0
102	7	0
103	4	0
104	5	0
105	7	0
106	4	0
107	6	0
108	8	0
109	5	0
110	8	1
111	5	1
112	7	1
113	9	1
114	8	1



로지스틱 회귀(Logistic Regression)

로지스틱 회귀

크기	수치	진단	예측확률
100	5	0	0.005
101	6	0	0.007
102	7	0	0.009
103	4	0	0.01
104	5	0	0.02
105	7	0	0.05
106	4	0	0.1
107	6	0	0.4
108	8	0	0.5
109	5	0	0.6
110	8	1	0.65
111	5	1	0.7
112	7	1	0.75
113	9	1	0.8
114	8	1	0.9



로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석

```
library(GGally)
library(skimr)
library(tidyverse)
library(tidymodels)
library(caret)

df <- read.csv("UniversalBank.csv", header = TRUE, na = ".")
skim(df)

df$Personal.Loan <- ifelse(df$Personal.Loan == 1, "yes", "no")
df$Personal.Loan <- as.factor(df$Personal.Loan)

trainIndex <- createDataPartition(df$Personal.Loan, p = 0.8, list = FALSE)
train_data <- df[trainIndex,]
test_data <- df[-trainIndex,]
```

로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석

```
result=glm(train_data$Personal.Loan ~., data= train_data, family = "binomial")  
  
tidy_result <- tidy(result)  
tidy_result <- mutate_if(tidy_result, is.numeric, round, 3)  
  
print(tidy_result)
```

로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석

```
library(car)
library(broom)
library(dplyr)

# 다중공선성(변수의 중복성) : 8이상 GVIF^(1/(2*DF))
vif(result)

# 다중공선성 위배 변수제거 후 다시 분석
result=glm(train_data$Personal.Loan ~., data= train_data, family = "binomial")

# Cook's distance 계산 (glm에도 사용 가능)
influencePlot(result, id.method="identify")
```

로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석

```
# 기준 설정  
# 상위 몇 %를 영향점으로 봄  
cooks_d <- cooks.distance(result)  
n <- nrow(train_data)
```

```
#Cooks'D값을 cutoff보다 큰 관측치만 추출  
cutoff <- 4 / n  
influential_idx <- which(cooks_d > cutoff)
```

```
influential_idx
```

로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석

```
# Test 데이터셋 결과
test_predictions <- predict(result, newdata = test_data, type = "response")
predicted_class <- ifelse(test_predictions > 0.5, "yes", "no")
predicted_class <- as.factor(predicted_class)
print(predicted_class)

# 모델 저장
saveRDS(result, file = "logistic_model.rds")

# 모델 로드
Load <- readRDS("logistic_model.rds")

tidy_result <- tidy(Load)
tidy_result
tidy_result <- mutate_if(tidy_result, is.numeric, round, 3)
```

로지스틱 회귀(Logistic Regression)

모델검정

- 수치형 : RMSE, R-Squared(R^2)
- 범주형 : Accuracy, ROC Curve
 - Precision(정밀도) : 모델에 의해 수행된 모든 긍정적인 예측 중 실제 긍정적인 예측의 비율
 - 장난감 상자에 10번 장난감을 꺼낼 경우, 10번 중에서 장난감 7개와 기타 물건 3개(책, 양말, 연필 등)를 선택함 → 내가 “장난감이다”라고 집어 든 것들 중에서 진짜 장난감 비율
 - Recall(재현율) : 모든 실제 양성 사례 중 참양성 예측의 비율(남아있는 장난감)
 - 상자에 10개의 장난감이 있다고 상상할 때(더 있을 수도 있음), 10개의 장난감 중에서 7개를 찾음 → 리콜은 당신이 발견한 상자 속 장난감의 총 개수에 관한 것 → 상자에 장난감이 총 10개 있었고 그중 7개를 찾았다면 7/10
 - Accuracy(정확도) : 전체 예측 정답 중 얼마나 올바르게 예측했는가에 대한 비율
 - F1 Score : Precision와 Recall의 조화 평균 → Label의 성능이 불균형일 경우 정확하게 평가할 수 있음

로지스틱 회귀(Logistic Regression)

모델검정

- **Precision(정밀도)** : $\frac{TP}{TP+FP}$
- **Recall(재현율)** : $\frac{TP}{TP+FN}$
- **Accuracy(정확도)** : $\frac{TP+TN}{TP+FN+FP+TN}$
- **F1-Score** : $2 \times \frac{Precision \times Recall}{Precision + Recall}$

		실제 정답	
		True(환자)	False(정상)
분류결과	True(환자)	TP(20)	FP(40)
	False(정상)	FN(30)	TN(10)

Precision →

Recall ↓

로지스틱 회귀(Logistic Regression)

모델검정

- 높은 Precision, 낮은 Recall → Precision : 내가 잡은 것들 중에 장난감의 비율 → 내선택에서 참의 비율

		실제 정답	
		True(장난감)	False(물건)
분류결과	True(장난감)	TP(5)	FP(0)
	False(물건)	FN(4)	TN(2)

- 높은 Recall, 낮은 Precision → Recall : 상자 안 모든 장난감 중 내가 찾아낸 장난감의 비율 → 참을 얼마나 놓치지 않았나?

		실제 정답	
		True(장난감)	False(물건)
분류결과	True(장난감)	TP(4)	FP(2)
	False(물건)	FN(2)	TN(2)

- 높은 Precision 및 높은 Recall: 9개 물건을 집었고 9개 모두 장난감이었고 상자에 장난감이 1개만 남아 있었다면 높은 Precision(장난감만 집은 것)과 높은 Recall(거의 모든 장난감을 찾았음)을 모두 갖게 됨

로지스틱 회귀(Logistic Regression)

모델검정

- ROC(Receiver Operating Characteristic)
- TPR(True Positive Rate) : 실제 클래스 True중에 잘 맞춘 것 → 4/6
- FPR(False Positive Rate) : 실제 클래스 False중에 못 맞춘 것 → 2/4

		실제 정답	
		True(장난감)	False(물건)
분류결과	True(장난감)	TP(4)	FP(2)
	False(물건)	FN(2)	TN(2)

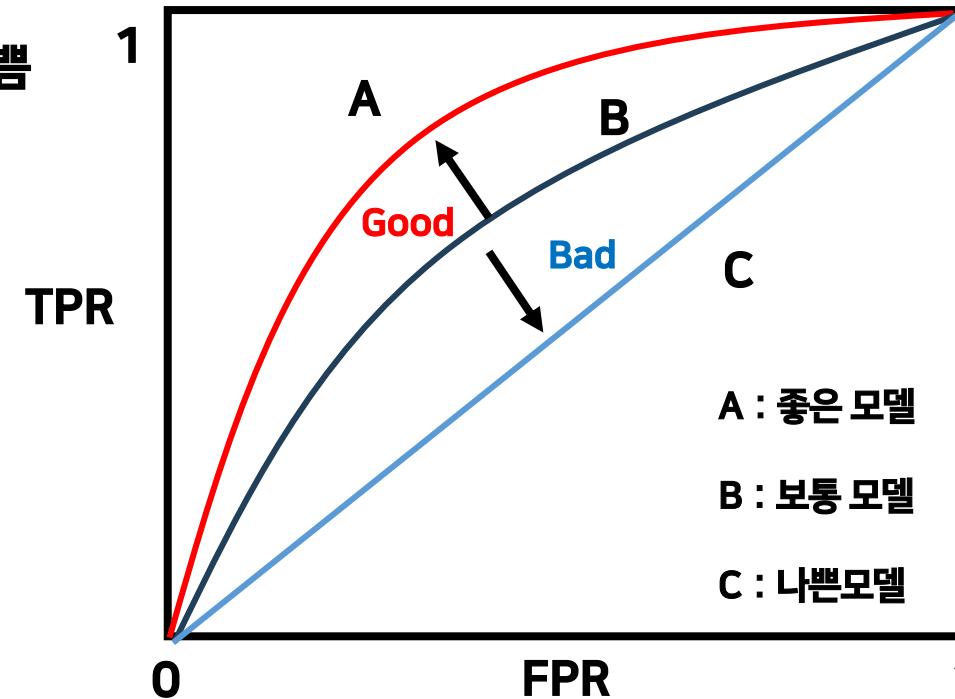
0.5 : 무작위 수준 → 나쁨

0.6~0.7 : 낮음

0.7~0.8 : 보통

0.8~0.9 : 좋음

0.9 이상 : 매우좋음



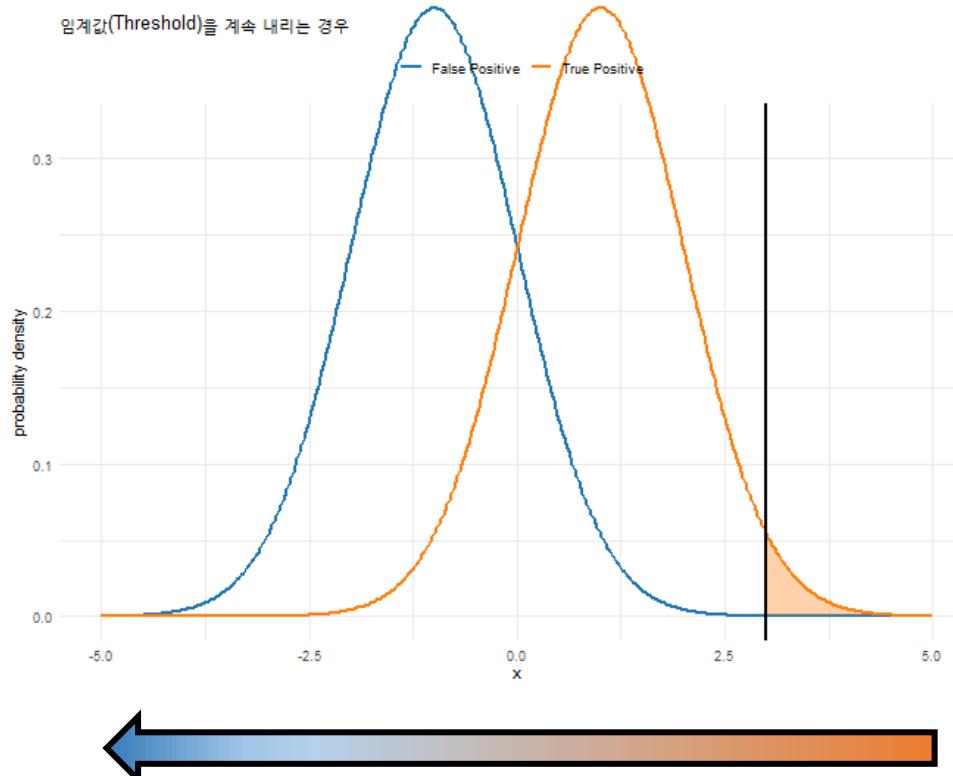
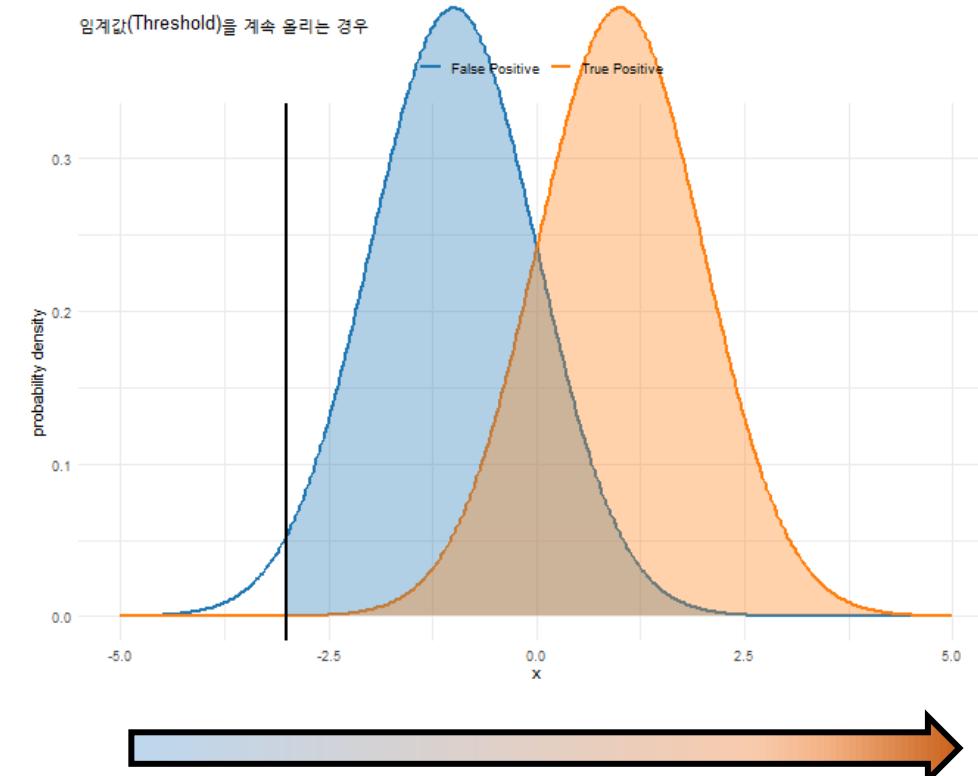
진짜를 얼마나 놓치지 않고 찾았는가?

$$TPR = \frac{TP}{TP + FN}$$

가짜를 얼마나 착각했는가?

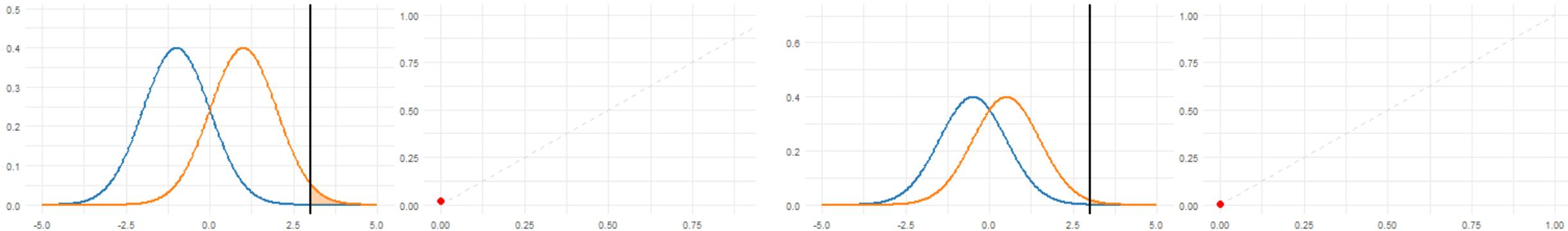
$$FPR = \frac{FP}{FP + TN}$$

로지스틱 회귀(Logistic Regression)

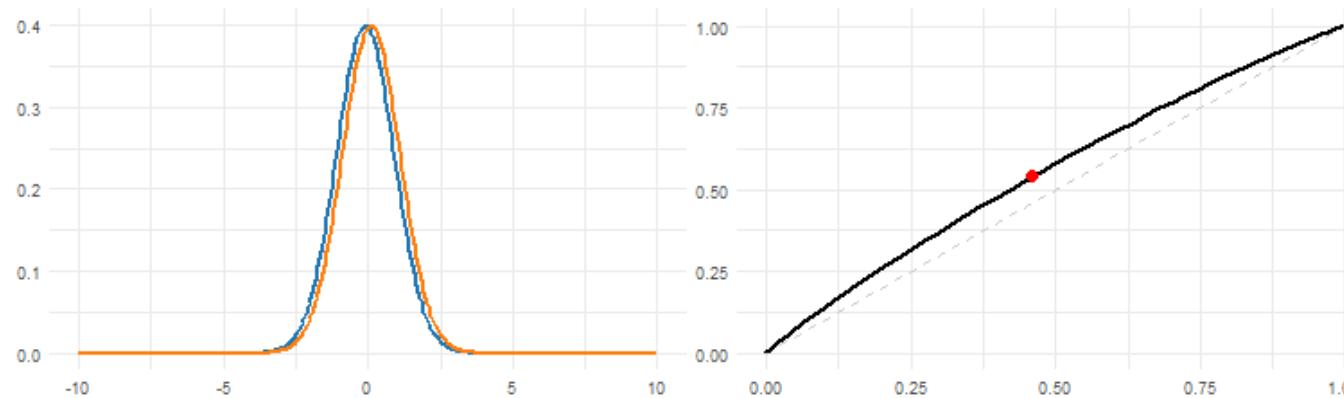
임계값을 내리는 경우**임계값을 올리는 경우**

로지스틱 회귀(Logistic Regression)

ROC커브가 그려지는 과정



두 집단의 거리가 멀어질수록 ROC커브가 왼쪽 위로 올라감 → 좋은 결과

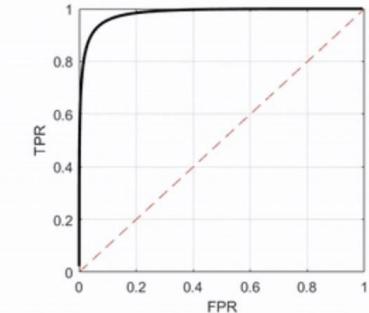
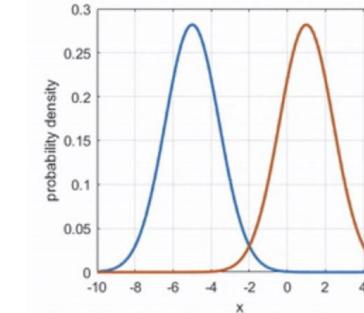
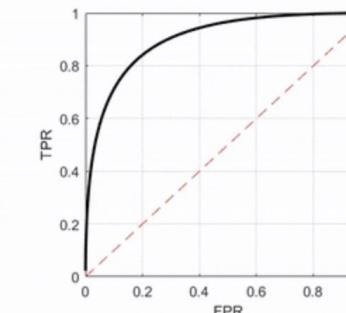
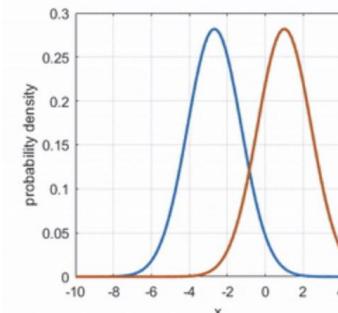
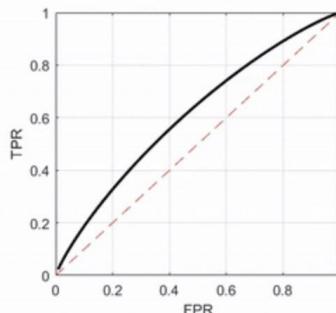
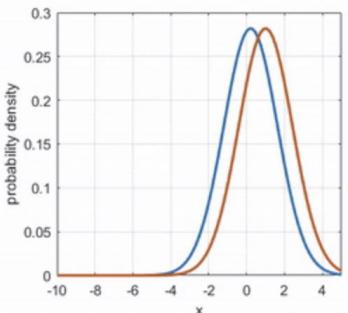


로지스틱 회귀(Logistic Regression)

모델검정

- ROC(Receiver Operating Characteristic)**
- TPR(True Positive Rate) :** 실제 클래스 True중에 잘 맞춘 것 → 4/6
- FPR(False Positive Rate) :** 실제 클래스 False중에 못 맞춘 것 → 2/4

		실제 정답	
		True(장난감)	False(물건)
분류결과	True(장난감)	TP(4)	FP(2)
	False(물건)	FN(2)	TN(2)



로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석(ROC Curve)

```
library(pROC)
library(caret)
library(e1071)

# 모델의 Confusion matrix
confusion_matrix <- confusionMatrix(predicted_class, test_data$Personal.Loan)
confusion_matrix

#Mcnemar's Test P-Value(혼동행렬에서 오류가 비대칭인지 파악 : 0.05보다 크면 데이터가 대칭적)
#Kappa(데이터가 한쪽이 너무 많을 때, 한쪽의 값을 도출해도 정확도가 높기 때문에 우연으로 맞췄는지 실제로 맞췄는지를 평가함)
#Kappa값이 1에 가까울 수록 좋은 모델, -값을 가지면 무작위로 예측하는 것 보다 정확도가 낮다는 의미(과적합 판단)

# ROC curve and AUC
roc_curve <- roc(test_data$Personal.Loan, test_predictions)
plot(roc_curve)
auc(roc_curve)
```

로지스틱 회귀(Logistic Regression)

로지스틱 회귀분석(ROC Curve)

```
# 새로운 데이터를 활용한 모델 적용  
df_test <- read.csv("UniversalBank_test.csv", header=TRUE)
```