**Final**

21500131김은총

**1.**

## linear regression

traffic\_train <- read\_csv("C:/Users/21500/Desktop/2019\_summer\_final/traffic\_train.csv")

tra.train <- traffic\_train

tra.train$date\_time <- as.character.Date(tra.train$date\_time)

tra.train <- separate(tra.train, date\_time, c("year", "month", "day"),sep="-")

str(tra.train)

tra.train <- tra.train[,-8]

str(tra.train)

tra.train <- tra.train[,-9]

# month 변수를 만들기 위해 data\_time 변수를 character 형식으로 바꿔주고 separate로 분리시킨 뒤 month만 남겨두고 다 삭제했다.

tra.test <- traffic\_test\_student

tra.test$date\_time <- as.character.Date(tra.test$date\_time)

tra.test <- separate(tra.test, date\_time, c("year", "month", "day"),sep="-")

str(tra.test)

tra.test <- tra.test[,-8]

str(tra.test)

tra.test <- tra.test[,-9]

str(tra.train)

names(tra.train)

######### full model

tra\_model.full <- lm(traffic\_volume ~ holiday + temp+ rain\_1h + snow\_1h

+ clouds\_all+weather\_main +month+weather\_description, tra.train)

summary(tra\_model.full)

###Stepwise Regression

library(MASS)

step <- stepAIC(tra\_model.full, direction = "both")

summary(step)

# step 함수를 사용해서 모델을 축소시켰다.

#display result

step$anova

# new

tra\_model\_remove <- lm(traffic\_volume ~ holiday + temp+I(temp^2) + clouds\_all+I(clouds\_all^2) + month + weather\_description, tra.train)

# temp와 clouds\_all 변수를 제곱한 값을 추가했다.

summary(tra\_model\_remove)

**2.**

# classification

#### logistic

occupancy\_train <- read\_csv("occupancy\_train.csv")

occupancy\_test <- read\_csv("occupancy\_test\_student.csv")

occ.train <- occupancy\_train

occ.test <- occupancy\_test

###Stepwise Regression

names(occ.train )

occ.train$Occupancy <- as.factor(occ.train$Occupancy)

###########

str(occ.train)

occ.train$date\_time <- as.character.Date(occ.train$date)

occ.train <- separate(occ.train, date, c("year", "month", "day"),sep="-")

# date 변수를 year, month, day로 나눠주는 작업을 했다

str(occ.train)

occ.train <- occ.train[,-3]

str(occ.train)

occ.train <- occ.train[,-1]

str(occ.train)

occ.train <- occ.train[,-8]

names(occ.train)

#########

model <- glm(Occupancy ~ Temperature+ Humidity + Light + CO2 +HumidityRatio, data = occ.train, family = binomial(link='logit'))

summary(model)

library(MASS)

step <- stepAIC(model, direction = "both")

# stepAIC 변수를 사용해서 입력 변수를 줄인 모델을 만들었다.

summary(step)

model2 <- glm(Occupancy ~ Temperature + Light + CO2 + HumidityRatio,

family = binomial(link = "logit"), data = occ.train)

summary(model2)

occ.train$pred <- predict(model2, newdata = occ.train, type = 'response')

# threshold를 여러가지 했을 때 0.6으로 설정했을 때가 가장 accuracy값이 높았다.

train\_pred\_vector <- ifelse(occ.train$pred < 0.6 , "0","1")

mean(occ.train$Occupancy == train\_pred\_vector) # 0.986

calAUC(occ.train$pred , occ.train$Occupancy) # 0.9945

## test (pred, act)

prob\_occupancy\_test <- predict(model2, newdata = occ.test, type = 'response')

pred\_occupancy\_test <- ifelse(occ.test$pred < 0.6 , "0","1")

pred\_occupancy\_test <- as.numeric(pred\_occupancy\_test)

**3.**

1) 두 그래프의 교차점 이전에는 같은 tpr값에 도달할 때 증가하는 fpr값은 A보다 B가 더 크다.

교차점 이후에는 같은 tpr값에 도달할 때 증가하는 fpr값은 A가 더 크다.

만약 암환자 분류의 경우 recall 값이 더 중요하기 때문에 적은 fpr값에서 높은 recall값을 얻을 수 있는 모델 B를 선택해야 할 것이다.

2)

- 스팸메일에서 자주등장하는 단어를 확인하고 최대 빈도 단어들을 약 10개~20개가 나왔을 때 진짜 스팸메임일 확률, 그 단어가 나왔는데 진짜 스팸메일이 아닌 확률, 최대 빈도 단어가 안나왔는데 스팸일 확률, 최대 빈도 단어가 안나왔는데 스팸메일이 아닐 확률을 만든다.

- 만약 구할 수 있다면 계정 사용자의 주소록에 있는 사람을 조사해서 발신자가 계정 사용자의 주소록에 있는지 유무를 판단하여 스팸메일을 구분할 수 있을 것이다.

- 또한 직장에서 자주 사용하는 단어 (업무, 대리, 과장, 상사 등)들을 모아 놓은 단어장을 만들어서 그 단어가 포함되어있는지 유무를 판단하는 변수도 만들 수 있을 것이다.