다변량분석 과제 5

2015170378

정은영

전체 데이터셋을 임의로 250 개의 Training dataset 과 71 개의 Validation dataset 으로 구분한 뒤 다음 각 물음에 답하시오.

```r
#Assignment5_2015170378정은영

# Install necessary packages
# glmnet: Ridge, Lasso, Elastic Net Logistic Regression
install.packages("glmnet")
install.packages("GA")

library(glmnet)
library(GA)

# Load the data & Preprocessing
df <- read.csv("Weather_Ankara.csv")
df_input <- df[,-10]
df_input_scaled <- scale(df_input, center = TRUE, scale = TRUE) #scaling,정규화 수행
df_target <- df$Mean_temperature
df_data_scaled <- data.frame(df_input_scaled, df_target)

nR <- nrow(df_data_scaled)
nC <- ncol(df_data_scaled)

set.seed(12345) #랜덤 알고리즘 지정
df_trn_idx <- sample(1:nR, 250)
df_trn <- df_data_scaled[df_trn_idx,] #250개를 training dataset으로
df_val <- df_data_scaled[-df_trn_idx,] #training 아닌것들
```

MLR 을 위해 데이터를 불러와 scaling 해준다. 10 열이 종속변수이다. 랜덤으로 250 개의 Training set 과 71 개의 validation set 으로 구분해준다.

[Q1] 모든 변수를 사용하여 Multiple Linear Regression (MLR) 모형을 학습하시오. 학습된 모형의 Adjusted R^2 는 얼마인가? 또한, 유의수준 1% (significance level = 0.01)에서 통계적으로 유의미한 변수는 어떤 것이 있는가? 학습한 모형을 이용하여 Validation dataset 에 대한 RMSE, MAE, MAPE 를 산출해 보시오.

```r
#[Q1]
# RMSE, MAE, MAPE 산출 함수
perf_eval_reg <- function(tgt_y, pre_y){ #tgt=정답, pre=예측 제공

  # RMSE
  rmse <- sqrt(mean((tgt_y - pre_y)^2))
  # MAE
  mae <- mean(abs(tgt_y - pre_y))
  # MAPE
  mape <- 100*mean(abs((tgt_y - pre_y)/tgt_y))

  return(c(rmse, mae, mape))

}
perf_mat <- matrix(0, nrow = 6, ncol = 6)
# Initialize a performance summary
rownames(perf_mat) <- c("Weather data(full model)","Weather Data(Adjusted R^2)","Weather Data(Forward)"
,"Weather Data(Backward)","Weather Data(Stepwise)","Weather Data(GA)")
colnames(perf_mat) <- c("input_formula","Adj.R^2","Time","RMSE", "MAE", "MAPE")
```

RMSE, MAE, MAPE 를 산출하기 위한 함수를 선언하고 performance matrix 를 초기화한다.

```r
#MLR
full_model <- lm(df_target ~ ., data = df_trn)
summary(full_model)
#adjusted R^2 = 0.9882, **(0.01) or ***(0.001) 을 고르면 Max_tem, Min_tem, Dewpoint, Sea_level_pressure
perf_mat[1,1] <- as.character(summary(full_model)$call)[2]
perf_mat[1,2] <- summary(full_model)$adj.r.squared
perf_mat[1,3] <- NA

#함수 사용하여 MAE, MAPE, RMSE 계산
mlr_df_haty <- predict(full_model, newdata = df_val)
perf_mat[1,4:6] <- perf_eval_reg(df_val$df_target, mlr_df_haty)
perf_mat
```

전체 input 으로 MLR 을 학습해본 결과는 다음과 같다.

```
> summary(full_model)

Call:
lm(formula = df_target ~ ., data = df_trn)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0922 -1.0702 -0.0671  0.9986  5.2698

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       48.94226    0.10369 471.992  < 2e-16 ***
Max_termperature   9.36488    0.28363  33.018  < 2e-16 ***
Min_temperature    4.18076    0.33520  12.472  < 2e-16 ***
Dewpoint           0.77053    0.27984   2.753  0.00635 **
Precipitation     -0.05339    0.10975  -0.486  0.62706
Sea_level_pressure -2.09996    0.73293  -2.865  0.00454 **
Standard_pressure  1.16432    0.57948   2.009  0.04563 *
Visibility         0.23765    0.13570   1.751  0.08118 .
Wind_speed         0.21669    0.16236   1.335  0.18324
Max_wind_speed    -0.04640    0.13831  -0.335  0.73758
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.635 on 240 degrees of freedom
Multiple R-squared:  0.9887,    Adjusted R-squared:  0.9882
F-statistic:  2326 on 9 and 240 DF,  p-value: < 2.2e-16
```

학습된 모델의 Adjusted R-squared 는 0.9882 로 input data 들이 종속변수인 평균온도를 그냥 평균온도들의 평균으로 예측했을 때 보다 상당히 잘(98%) 예측할 수 있을 것이라는 것을 확인할 수 있다.

또한 유의수준 1%(0.01)에서 유의미한 변수들을 살펴보면 위의 summary 의 *이 2 개 이상인 것들을 살펴보면 되는데, Max_temperature, Min_temperature, Dewpoint, Sea_level_pressure 이다. 최고온도, 최저온도, 이슬점, 해면기압이 평균온도에 유의미한 변화를 준다는 것을 알 수 있다. 최고기온이 1 단위 높아지면 평균온도는 약 9 단위 높아지며, 최저기온이 1 단위 높아질 때 평균온도가 약 4 단위 높아질 것이라는 것을 알 수 있다. 또한 이슬점이 1 높아지면 평균온도가 0.77 정도 높아지며, 해면기압이 1 높아질수록 평균온도는 약 2.1 정도 낮아짐을 알 수 있다.

```
> perf_mat
                           input_formula   Adj.R^2             Time
Weather data(full model)   "df_target ~ ." "0.988241162779006" NA
Weather Data(Adjusted R^2) "0"             "0"                 "0"
Weather Data(Forward)      "0"             "0"                 "0"
Weather Data(Backward)     "0"             "0"                 "0"
Weather Data(Stepwise)     "0"             "0"                 "0"
Weather Data(GA)           "0"             "0"                 "0"
                           RMSE               MAE                MAPE
Weather data(full model)   "1.48096892976632" "1.04811766863543" "2.32711707340505"
Weather Data(Adjusted R^2) "0"                "0"                "0"
Weather Data(Forward)      "0"                "0"                "0"
Weather Data(Backward)     "0"                "0"                "0"
Weather Data(Stepwise)     "0"                "0"                "0"
Weather Data(GA)           "0"                "0"                "0"
```

full model 에 대한 RMSE, MAE, MAPE 는 위와 같다.

[Q2] Exhaustive Search 를 수행하는 함수를 직접 구현하고 Training Dataset 에 대한 Adjusted R^2 기준으 로 가장 높은 값이 산출된 변수 집합을 제시하시오. 또한 Exhaustive Search 에 소요된 시간을 산출하시오. 학습한 모형을 이용하여 Validation dataset 에 대한 RMSE, MAE, MAPE 를 산출하고 모든 변수를 사용한 MLR 모형의 결과와 비교하시오.

```
#[Q2]
es <- function(trn_data){
  target <- c(0,0,0,0,0,0,0,0,0)
  dfs <- function(len, idx, target,trn_data,adjR2_max){
    a <- c(1,2,3,4,5,6,7,8,9)
    if(len == l){
      tmp_x <- paste(colnames(df_trn)[target[1:l]], collapse=" + ")
      tmp_xy <- paste("df_target ~ ", tmp_x, collapse = "")
      es_model <- lm(as.formula(tmp_xy), data = df_trn)
      adjR2 <- round(summary(es_model)$adj.r.squared,digits = 5)
      print(paste("adj.R^2:",adjR2,",formular: ",tmp_xy))
      return()
    }
    nxt <- a[(idx+1)]
    target[(len+1)] <- nxt
    dfs((len + 1), (idx + 1),target,trn_data,adjR2_max)
    target[len+1] <- 0
    if((9-idx)>(l-len)) {
      return(dfs(len, (idx+1),target,trn_data,adjR2_max))
    }
  }
  for(l in 1:9){
    dfs(0,0,target, trn_data,0)
  }
}
```

위의 함수는 exhaustive search 를 수행하는 함수이다. 1-9 까지 열, 즉 9 가지 변수에 대한 조합을 DFS 로 가능한 모든 조합에 대해 구한 뒤 그 변수들을 model 에 넣어 adjusted R squared 와 사용된 변수를 출력한다.

```
85  start_time <- proc.time()
86  es(df_trn)
87  end_time <- proc.time()
88  perf_mat[2,3] <- (end_time - start_time)[3]
```

함수를 실행하여 걸린 시간을 matrix 에 넣고, 결과를 확인한다.

```
> es(df_trn)
[1] "adj.R^2: 0.94542 ,formular:  df_target ~  Max_termperature"
[1] "adj.R^2: 0.8671 ,formular:  df_target ~  Min_temperature"
[1] "adj.R^2: 0.7699 ,formular:  df_target ~  Dewpoint"
[1] "adj.R^2: 0.00056 ,formular:  df_target ~  Precipitation"
[1] "adj.R^2: 0.37286 ,formular:  df_target ~  Sea_level_pressure"
[1] "adj.R^2: 0.05148 ,formular:  df_target ~  Standard_pressure"
[1] "adj.R^2: 0.25959 ,formular:  df_target ~  Visibility"
[1] "adj.R^2: 0.03001 ,formular:  df_target ~  Wind_speed"
[1] "adj.R^2: 0.02426 ,formular:  df_target ~  Max_wind_speed"
[1] "adj.R^2: 0.98632 ,formular:  df_target ~  Max_termperature + Min_temperature"
[1] "adj.R^2: 0.96944 ,formular:  df_target ~  Max_termperature + Dewpoint"
[1] "adj.R^2: 0.94865 ,formular:  df_target ~  Max_termperature + Precipitation"
[1] "adj.R^2: 0.96311 ,formular:  df_target ~  Max_termperature + Sea_level_pressure"
[1] "adj.R^2: 0.95775 ,formular:  df_target ~  Max_termperature + Standard_pressure"
[1] "adj.R^2: 0.947 ,formular:  df_target ~  Max_termperature + Visibility"
[1] "adj.R^2: 0.95548 ,formular:  df_target ~  Max_termperature + Wind_speed"
[1] "adj.R^2: 0.9486 ,formular:  df_target ~  Max_termperature + Max_wind_speed"
[1] "adj.R^2: 0.87276 ,formular:  df_target ~  Min_temperature + Dewpoint"
[1] "adj.R^2: 0.87517 ,formular:  df_target ~  Min_temperature + Precipitation"
[1] "adj.R^2: 0.86664 ,formular:  df_target ~  Min_temperature + Sea_level_pressure"
[1] "adj.R^2: 0.87171 ,formular:  df_target ~  Min_temperature + Standard_pressure"
[1] "adj.R^2: 0.87419 ,formular:  df_target ~  Min_temperature + Visibility"
[1] "adj.R^2: 0.87491 ,formular:  df_target ~  Min_temperature + Wind_speed"
[1] "adj.R^2: 0.86752 ,formular:  df_target ~  Min_temperature + Max_wind_speed"
[1] "adj.R^2: 0.7839 ,formular:  df_target ~  Dewpoint + Precipitation"
[1] "adj.R^2: 0.77906 ,formular:  df_target ~  Dewpoint + Sea_level_pressure"
[1] "adj.R^2: 0.76904 ,formular:  df_target ~  Dewpoint + Standard_pressure"
[1] "adj.R^2: 0.81657 ,formular:  df_target ~  Dewpoint + Visibility"
[1] "adj.R^2: 0.77434 ,formular:  df_target ~  Dewpoint + Wind_speed"
[1] "adj.R^2: 0.77064 ,formular:  df_target ~  Dewpoint + Max_wind_speed"
[1] "adj.R^2: 0.39902 ,formular:  df_target ~  Precipitation + Sea_level_pressure"
[1] "adj.R^2: 0.06258 ,formular:  df_target ~  Precipitation + Standard_pressure"
[1] "adj.R^2: 0.26243 ,formular:  df_target ~  Precipitation + Visibility"
[1] "adj.R^2: 0.03347 ,formular:  df_target ~  Precipitation + Wind_speed"
[1] "adj.R^2: 0.02635 ,formular:  df_target ~  Precipitation + Max_wind_speed"
[1] "adj.R^2: 0.88784 ,formular:  df_target ~  Sea_level_pressure + Standard_pressure"
[1] "adj.R^2: 0.466 ,formular:  df_target ~  Sea_level_pressure + Visibility"
[1] "adj.R^2: 0.37277 ,formular:  df_target ~  Sea_level_pressure + Wind_speed"
[1] "adj.R^2: 0.37054 ,formular:  df_target ~  Sea_level_pressure + Max_wind_speed"
[1] "adj.R^2: 0.27949 ,formular:  df_target ~  Standard_pressure + Visibility"
[1] "adj.R^2: 0.05971 ,formular:  df_target ~  Standard_pressure + Wind_speed"
[1] "adj.R^2: 0.05963 ,formular:  df_target ~  Standard_pressure + Max_wind_speed"
```

...생략

```
[1] "adj.R^2: 0.98793 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Standard_pressure + Visibility + Wind_speed"
[1] "adj.R^2: 0.98781 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Standard_pressure + Visibility + Max_wind_speed"
[1] "adj.R^2: 0.9877 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Standard_pressure + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98723 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98832 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
[1] "adj.R^2: 0.98824 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
[1] "adj.R^2: 0.98818 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98813 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98793 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98797 ,formular:  df_target ~  Max_termperature + Min_temperature + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
[1] "adj.R^2: 0.98795 ,formular:  df_target ~  Max_termperature + Min_temperature + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
[1] "adj.R^2: 0.98789 ,formular:  df_target ~  Max_termperature + Min_temperature + Precipitation + Sea_level_pressure + Standard_pressure + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98764 ,formular:  df_target ~  Max_termperature + Min_temperature + Precipitation + Sea_level_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98737 ,formular:  df_target ~  Max_termperature + Min_temperature + Precipitation + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98797 ,formular:  df_target ~  Max_termperature + Min_temperature + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98056 ,formular:  df_target ~  Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
[1] "adj.R^2: 0.97891 ,formular:  df_target ~  Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
[1] "adj.R^2: 0.98064 ,formular:  df_target ~  Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.97971 ,formular:  df_target ~  Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.97896 ,formular:  df_target ~  Max_termperature + Dewpoint + Precipitation + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98077 ,formular:  df_target ~  Max_termperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.97221 ,formular:  df_target ~  Max_termperature + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.93511 ,formular:  df_target ~  Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
[1] "adj.R^2: 0.93127 ,formular:  df_target ~  Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
[1] "adj.R^2: 0.93181 ,formular:  df_target ~  Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.9015 ,formular:  df_target ~  Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.90166 ,formular:  df_target ~  Min_temperature + Dewpoint + Precipitation + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.93154 ,formular:  df_target ~  Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.93493 ,formular:  df_target ~  Min_temperature + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.91333 ,formular:  df_target ~  Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98828 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
[1] "adj.R^2: 0.9882 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
[1] "adj.R^2: 0.98814 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98809 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98789 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98828 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98792 ,formular:  df_target ~  Max_termperature + Min_temperature + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.9807 ,formular:  df_target ~  Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.9351 ,formular:  df_target ~  Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
[1] "adj.R^2: 0.98824 ,formular:  df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
```

다음과 같이 511 개의 경우의 수가 나오며, (아무것도 안포함시켰을 때 1 뺌. 2^9-1)

그 중 adjusted R squared 가 가장 큰 경우는 df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed 이다.(0.98832)

```
es_Var <-   "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure +
Standard_pressure + Visibility + Wind_speed"

es_model <- lm(as.formula(es_Var), data = df_trn)
summary(es_model)
perf_mat[2,2] <- summary(es_model)$adj.r.squared
perf_mat[2,1] <- es_Var
# Make prediction
mlr_df_haty2 <- predict(es_model, newdata = df_val)
perf_mat[2,4:6] <- perf_eval_reg(df_val$df_target, mlr_df_haty2)
perf_mat
```

따라서 위의 경우를 이용하여 LR 모델을 만들어 결과를 확인한다.

```
> summary(es_model)

Call:
lm(formula = as.formula(es_Var), data = df_trn)

Residuals:
    Min      1Q  Median      3Q     Max
-4.0817 -1.1040 -0.0582  0.9627  5.3158

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         48.9442     0.1033 473.870  < 2e-16 ***
Max_termperature     9.3911     0.2730  34.399  < 2e-16 ***
Min_temperature      4.1951     0.3307  12.684  < 2e-16 ***
Dewpoint             0.7399     0.2740   2.701  0.00741 **
Sea_level_pressure  -2.0854     0.7295  -2.859  0.00462 **
Standard_pressure    1.1641     0.5772   2.017  0.04483 *
Visibility           0.2322     0.1348   1.723  0.08612 .
Wind_speed           0.1866     0.1315   1.419  0.15712
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.629 on 242 degrees of freedom
Multiple R-squared:  0.9886,    Adjusted R-squared:  0.9883
F-statistic:  3011 on 7 and 242 DF,  p-value: < 2.2e-16
```

마찬가지로 유의수준 0.01 에서 평균기온에 영향을 주는 유의미한 변수들은 최고기온, 최저기온, 이슬점, 해면기압이다.

```
> perf_mat
                        input_formula
Weather data(full model)   "df_target ~ ."
Weather Data(Adjusted R^2) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Forward)      "0"
Weather Data(Backward)     "0"
Weather Data(Stepwise)     "0"
Weather Data(GA)           "0"
                        Adj.R^2             Time               RMSE               MAE                MAPE
Weather data(full model)   "0.988241162779006" NA                 "1.48096892976632" "1.04811766863543" "2.32711707340505"
Weather Data(Adjusted R^2) "0.988321493056102" "1.25100000000384" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Forward)      "0"                 "0"                "0"                "0"                "0"
Weather Data(Backward)     "0"                 "0"                "0"                "0"                "0"
Weather Data(Stepwise)     "0"                 "0"                "0"                "0"                "0"
Weather Data(GA)           "0"                 "0"                "0"                "0"                "0"
```

걸린시간은 약 "1.25100000000384", RMSE 는 "1.47648481469331" MAE "1.04837333694827" MAPE "2.32839580198534" 이다. full model 과 비교했을 때 RMSE 는 살짝 낮아지고, MAE 와 MAPE 는 살짝 높지만 큰 차이는 없는 것 같아 만약 내가 선택하게 된다면 full model 이 아닌 exhaustive

search 를 통한 모델, 즉 위와 같이 Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed 를 사용한 모델을 선택할 것이다.

[Q3] Forward Selection, Backward Elimination, Stepwise Selection 방식을 사용하여 MLR 변수 선택 과정을 수행해 보시오. 각 방법론마다 [Q2]와 같이 Training dataset 에 대한 Adjusted R^2 및 소요 시간, Validation dataset 에 대한 RMSE, MAE, MAPE 를 산출하고 [Q1] 및 [Q2]의 결과와 비교하시오.

```
#[Q3]
#첫 번 째 - Foward, 두 번 째 - Backward, 세 번 째 - Stepwise
model_v <- c("df_target~1", "full_model", "full_model")
model_d <- c("forward", "backward", "both")

# Variable selection method: Forward Selection, Backward Elimination, Stepwise Selection

for(i in 1:3){
  start_time <- proc.time()
  i_model <- step(lm(as.formula(model_v[i]), data =df_trn),
                  scope = list(upper = full_model, lower = df_target ~ 1),
                  direction= model_d[i], trace = 1)
  end_time <- proc.time()
  perf_mat[i+2,3] <- (end_time - start_time)[3]
  summary(i_model)
  perf_mat[i+2,2] <- summary(i_model)$adj.r.squared
  perf_mat[i+2,1] <- as.character(summary(i_model)$call)[2]
  # Make prediction
  mlr_df_haty2 <- predict(i_model, newdata = df_val)
  perf_mat[i+2,4:6] <- perf_eval_reg(df_val$df_target, mlr_df_haty2)
}
perf_mat
```

코드를 짧게 하기 위해 for 문을 사용하여 i=1 일 때 forward, 2 일때 backward, 3 일때 stepwise 로 각각을 구하여 matrix 에 집어넣는다.

```
[1] "forward"
Start:  AIC=1357.54
df_target ~ 1

                  Df Sum of Sq   RSS    AIC
+ Max_termperature  1     53515  3076  631.49
+ Min_temperature   1     49100  7491  853.98
+ Dewpoint          1     43622 12969  991.22
+ Sea_level_pressure 1    21243 35348 1241.88
+ Visibility        1     14859 41732 1283.39
+ Standard_pressure 1      3129 53462 1345.31
+ Wind_speed        1      1919 54672 1350.91
+ Max_wind_speed    1      1595 54996 1352.39
<none>                          56591 1357.54
+ Precipitation     1       259 56332 1358.39

Step:  AIC=631.49
df_target ~ Max_termperature

                  Df Sum of Sq    RSS    AIC
+ Min_temperature   1    2307.96  768.11 286.62
+ Dewpoint          1    1360.82 1715.26 487.46
+ Sea_level_pressure 1   1005.45 2070.62 534.54
+ Standard_pressure 1     704.36 2371.72 568.48
+ Wind_speed        1     576.67 2499.41 581.59
+ Precipitation     1     193.55 2882.52 617.24
+ Max_wind_speed    1     190.43 2885.64 617.51
+ Visibility        1     100.61 2975.46 625.17
<none>                           3076.08 631.49

Step:  AIC=286.62
df_target ~ Max_termperature + Min_temperature

                  Df Sum of Sq    RSS    AIC
+ Sea_level_pressure 1    73.916 694.20 263.32
+ Standard_pressure 1     55.217 712.90 269.97
+ Wind_speed        1     20.972 747.14 281.70
+ Visibility        1     20.811 747.30 281.75
+ Dewpoint          1     15.698 752.42 283.46
+ Max_wind_speed    1     10.675 757.44 285.12
<none>                           768.11 286.62
+ Precipitation     1      2.377 765.74 287.85

Step:  AIC=263.32
df_target ~ Max_termperature + Min_temperature + Sea_level_pressure

                  Df Sum of Sq    RSS    AIC
+ Standard_pressure 1    22.6014 671.60 257.05
+ Dewpoint          1    13.2330 680.97 260.51
+ Visibility        1    12.8249 681.37 260.66
+ Wind_speed        1     7.2957 686.90 262.68
<none>                           694.20 263.32
+ Max_wind_speed    1     2.4959 691.70 264.42
+ Precipitation     1     0.0185 694.18 265.32

Step:  AIC=257.05
df_target ~ Max_termperature + Min_temperature + Sea_level_pressure +
    Standard_pressure

                  Df Sum of Sq    RSS    AIC
+ Dewpoint          1    9.0154 662.58 255.67
+ Visibility        1    8.4348 663.16 255.89
<none>                           671.60 257.05
+ Wind_speed        1    5.2953 666.30 257.07
+ Max_wind_speed    1    1.8255 669.77 258.37
+ Precipitation     1    0.0128 671.58 259.05

Step:  AIC=255.67
df_target ~ Max_termperature + Min_temperature + Sea_level_pressure +
    Standard_pressure + Dewpoint

                  Df Sum of Sq    RSS    AIC
+ Visibility        1   14.9193 647.66 251.98
+ Wind_speed        1   12.3833 650.20 252.95
<none>                           662.58 255.67
+ Max_wind_speed    1    2.8874 659.69 256.58
+ Precipitation     1    0.2852 662.30 257.56

Step:  AIC=251.98
df_target ~ Max_termperature + Min_temperature + Sea_level_pressure +
    Standard_pressure + Dewpoint + Visibility

                  Df Sum of Sq    RSS    AIC
+ Wind_speed        1    5.3460 642.32 251.91
<none>                           647.66 251.98
+ Max_wind_speed    1    0.8222 646.84 253.66
+ Precipitation     1    0.7212 646.94 253.70

Step:  AIC=251.9
df_target ~ Max_termperature + Min_temperature + Sea_level_pressure +
    Standard_pressure + Dewpoint + Visibility + Wind_speed

                  Df Sum of Sq    RSS    AIC
<none>                           642.32 251.91
+ Precipitation     1   0.62603 641.69 253.66
+ Max_wind_speed    1   0.29426 642.02 253.79
[1] "backward"
Start:  AIC=255.54
df_target ~ Max_termperature + Min_temperature + Dewpoint + Precipitation +
    Sea_level_pressure + Standard_pressure + Visibility + Wind_speed +
    Max_wind_speed

                  Df Sum of Sq    RSS    AIC
- Max_wind_speed    1      0.30  641.7 253.66
- Precipitation     1      0.63  642.0 253.79
- Wind_speed        1      4.76  646.2 255.39
<none>                           641.4 255.54
- Visibility        1      8.20  649.6 256.72
- Standard_pressure 1     10.79  652.2 257.71
- Dewpoint          1     20.26  661.7 261.32
- Sea_level_pressure 1    21.94  663.3 261.95
- Min_temperature   1    415.72 1057.1 378.46
- Max_termperature  1   2913.46 3554.8 681.65

Step:  AIC=253.66
df_target ~ Max_termperature + Min_temperature + Dewpoint + Precipitation +
    Sea_level_pressure + Standard_pressure + Visibility + Wind_speed

                  Df Sum of Sq    RSS    AIC
- Precipitation     1      0.63  642.3 251.90
<none>                           641.7 253.66
- Wind_speed        1      5.25  646.9 253.70
- Visibility        1      8.20  649.9 254.84
- Standard_pressure 1     10.89  652.6 255.87
- Dewpoint          1     19.96  661.7 259.32
- Sea_level_pressure 1    22.00  663.7 260.09
- Min_temperature   1    427.25 1068.9 379.24
- Max_termperature  1   2927.48 3569.2 680.66
```

```
Step:  AIC=251.9
df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure +
    Standard_pressure + Visibility + Wind_speed

                    Df Sum of Sq    RSS    AIC
<none>                           642.3 251.90
- Wind_speed         1      5.35  647.7 251.98
- Visibility         1      7.88  650.2 252.95
- Standard_pressure  1     10.80  653.1 254.07
- Dewpoint           1     19.36  661.7 257.33
- Sea_level_pressure 1     21.69  664.0 258.21
- Min_temperature    1    427.00 1069.3 377.33
- Max_termperature   1   3140.67 3783.0 693.20
[1] "both"
Start:  AIC=255.54
df_target ~ Max_termperature + Min_temperature + Dewpoint + Precipitation +
    Sea_level_pressure + Standard_pressure + Visibility + Wind_speed +
    Max_wind_speed

                    Df Sum of Sq    RSS    AIC
- Max_wind_speed     1      0.30  641.7 253.66
- Precipitation      1      0.63  642.0 253.79
- Wind_speed         1      4.76  646.2 255.39
<none>                           641.4 255.54
- Visibility         1      8.20  649.6 256.72
- Standard_pressure  1     10.79  652.2 257.71
- Dewpoint           1     20.26  661.7 261.32
- Sea_level_pressure 1     21.94  663.3 261.95
- Min_temperature    1    415.72 1057.1 378.46
- Max_termperature   1   2913.46 3554.8 681.65

Step:  AIC=253.66
df_target ~ Max_termperature + Min_temperature + Dewpoint + Precipitation +
    Sea_level_pressure + Standard_pressure + Visibility + Wind_speed

                    Df Sum of Sq    RSS    AIC
- Precipitation      1      0.63  642.3 251.90
<none>                           641.7 253.66
- Wind_speed         1      5.25  646.9 253.70
- Visibility         1      8.20  649.9 254.84
+ Max_wind_speed     1      0.30  641.4 255.54
- Standard_pressure  1     10.89  652.6 255.87
- Dewpoint           1     19.96  661.7 259.32
- Sea_level_pressure 1     22.00  663.7 260.09
- Min_temperature    1    427.25 1068.9 379.24
- Max_termperature   1   2927.48 3569.2 680.66
```

```
Step:  AIC=251.9
df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure +
    Standard_pressure + Visibility + Wind_speed

                    Df Sum of Sq    RSS    AIC
<none>                           642.3 251.90
- Wind_speed         1      5.35  647.7 251.98
- Visibility         1      7.88  650.2 252.95
+ Precipitation      1      0.63  641.7 253.66
+ Max_wind_speed     1      0.29  642.0 253.79
- Standard_pressure  1     10.80  653.1 254.07
- Dewpoint           1     19.36  661.7 257.33
- Sea_level_pressure 1     21.69  664.0 258.21
- Min_temperature    1    427.00 1069.3 377.33
- Max_termperature   1   3140.67 3783.0 693.20
```

다음과 같은 단계로 변수가 선택되며 선택된 결과는 다음과 같다.

```
> perf_mat
                          input_formula
Weather data(full model)   "df_target ~ ."
Weather Data(Adjusted R^2) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Forward)      "df_target ~ Max_termperature + Min_temperature + Sea_level_pressure + Standard_pressure + Dewpoint + Visibility + Wind_speed"
Weather Data(Backward)     "df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Stepwise)     "df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(GA)           "0"
                          Adj.R^2             Time                 RMSE               MAE                MAPE
Weather data(full model)   "0.988241162779006" NA                   "1.48096892976632" "1.04811766863543" "2.32711707340505"
Weather Data(Adjusted R^2) "0.988321493056102" "1.25100000000384"   "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Forward)      "0.988321493056102" "0.0599999999976717" "1.47648481469331" "1.04837333694827" "2.32839580198535"
Weather Data(Backward)     "0.988321493056102" "0.0249999999941792" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Stepwise)     "0.988321493056102" "0.0299999999988358" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(GA)           "0"                 "0"                  "0"                "0"                "0"
```

Forward selection, Backward Elimination, Stepwise selection 모두 같은 결과가 나왔다. 즉 두번째 행인 Exhaustive search 방법과 같은 결과이다. 다만 실행시간의 차이가 있는데, 3 가지 방법이 exhaustive search 보다 빠른 시간에 수행되므로 효율적이다. 수행시간은 3 가지 방법 모두 작기 때문에 셋 중 아무거나 선택해도 될 것이다. 위의 문제와 같은 이유로 full 모델보다는 선택된 모델을 사용한다.

[Q4] Adjusted R^2 를 Fitness function 으로 하는 Genetic Algorithm 기반의 변수 선택 함수를 작성해 보시오. 작성한 함수를 이용하여 GA 를 이용한 변수 선택을 수행한 결과를 소요 시간, Adjusted R^2, Validation dataset 에 대한 RMSE, MAE, MAPE 관점에서 앞선 결과들과 비교해 보시오.

```r
#[Q4]
fit_F1 <- function(string){
  sel_var_idx <- which(string == 1) #사용하는 변수의 index 항에 집어넣기.
  # Use variables whose gene value is 1
  sel_x <- x[, sel_var_idx] #사용하라고 지정된 것만 사용
  xy <- data.frame(sel_x, y)
  # Training the model
  GA_lr <- lm(y ~ ., data = xy)
  return(summary(GA_lr)$adj.r.squared)
}
```

fitness function 의 반환값을 adjusted R^2 로 설정한다.

```
x <- as.matrix(df_trn[,-10])
y <- df_trn[,10]

start_time <- proc.time()
GA_F1 <- ga(type = "binary", fitness = fit_F1, nBits = ncol(x),
            names = colnames(x), popSize = 100, pcrossover = 0.5,
            pmutation = 0.01, maxiter = 150, elitism = 2, seed = 123)
end_time <- proc.time()
perf_mat[6,3] <- (end_time - start_time)[3]
best_var_idx <- which(GA_F1@solution == 1)
colnames(df_trn)[best_var_idx]

tmp_x <- paste(colnames(df_trn)[best_var_idx], collapse=" + ")
tmp_xy <- paste("df_target ~ ", tmp_x, collapse = "")
perf_mat[6,1] <- tmp_xy
as.formula(tmp_xy) #string을 formula라는 새로운 데이터 형태로 바꿈(작동시키기위해)

ga_model <- lm(tmp_xy, data = df_trn)
summary(ga_model)
perf_mat[6,2] <- summary(ga_model)$adj.r.squared

mlr_df_haty5 <- predict(ga_model, newdata = df_val)
perf_mat[6,4:6] <- perf_eval_reg(df_val$df_target, mlr_df_haty5)
perf_mat
```

수업시간에 교수님께서 말씀하신 대로 popSize 는 100 이상으로(100), maxiter 는 100 세대보다 많이(150) 설정한다. 또한 elitism 은 2 로 설정했다.

```
GA | iter = 1 | Mean = 0.8778117 | Best = 0.9882374
GA | iter = 2 | Mean = 0.9058851 | Best = 0.9882374
GA | iter = 3 | Mean = 0.9322432 | Best = 0.9882374
GA | iter = 4 | Mean = 0.9244320 | Best = 0.9882374
GA | iter = 5 | Mean = 0.9288704 | Best = 0.9882374
GA | iter = 6 | Mean = 0.9364720 | Best = 0.9883215
GA | iter = 7 | Mean = 0.9398498 | Best = 0.9883215
GA | iter = 8 | Mean = 0.9405508 | Best = 0.9883215
GA | iter = 9 | Mean = 0.9402719 | Best = 0.9883215
GA | iter = 10 | Mean = 0.9305880 | Best = 0.9883215
GA | iter = 11 | Mean = 0.9426953 | Best = 0.9883215
GA | iter = 12 | Mean = 0.9354120 | Best = 0.9883215
GA | iter = 13 | Mean = 0.9384761 | Best = 0.9883215
GA | iter = 14 | Mean = 0.9454426 | Best = 0.9883215
GA | iter = 15 | Mean = 0.9452731 | Best = 0.9883215
GA | iter = 16 | Mean = 0.9519599 | Best = 0.9883215
GA | iter = 17 | Mean = 0.9481704 | Best = 0.9883215
GA | iter = 18 | Mean = 0.9535101 | Best = 0.9883215
GA | iter = 19 | Mean = 0.9581605 | Best = 0.9883215
GA | iter = 20 | Mean = 0.9656025 | Best = 0.9883215
GA | iter = 21 | Mean = 0.9661190 | Best = 0.9883215
GA | iter = 22 | Mean = 0.9719581 | Best = 0.9883215
GA | iter = 23 | Mean = 0.9757875 | Best = 0.9883215
GA | iter = 24 | Mean = 0.9780009 | Best = 0.9883215
GA | iter = 25 | Mean = 0.9819727 | Best = 0.9883215
GA | iter = 26 | Mean = 0.9834973 | Best = 0.9883215
GA | iter = 27 | Mean = 0.9830653 | Best = 0.9883215
GA | iter = 28 | Mean = 0.9847169 | Best = 0.9883215
GA | iter = 29 | Mean = 0.9848040 | Best = 0.9883215
GA | iter = 30 | Mean = 0.9845393 | Best = 0.9883215
GA | iter = 31 | Mean = 0.9850052 | Best = 0.9883215
GA | iter = 32 | Mean = 0.9845993 | Best = 0.9883215
GA | iter = 33 | Mean = 0.9853370 | Best = 0.9883215
GA | iter = 34 | Mean = 0.9854511 | Best = 0.9883215
GA | iter = 35 | Mean = 0.9861658 | Best = 0.9883215
GA | iter = 36 | Mean = 0.9858882 | Best = 0.9883215
GA | iter = 37 | Mean = 0.9864089 | Best = 0.9883215
GA | iter = 38 | Mean = 0.9867632 | Best = 0.9883215
GA | iter = 39 | Mean = 0.9869484 | Best = 0.9883215
GA | iter = 40 | Mean = 0.9876493 | Best = 0.9883215
GA | iter = 41 | Mean = 0.9875831 | Best = 0.9883215
GA | iter = 42 | Mean = 0.9877384 | Best = 0.9883215
GA | iter = 43 | Mean = 0.9878262 | Best = 0.9883215
GA | iter = 44 | Mean = 0.9878033 | Best = 0.9883215
GA | iter = 45 | Mean = 0.9876624 | Best = 0.9883215
GA | iter = 46 | Mean = 0.9875133 | Best = 0.9883215
GA | iter = 47 | Mean = 0.9872922 | Best = 0.9883215
GA | iter = 48 | Mean = 0.9874445 | Best = 0.9883215
GA | iter = 49 | Mean = 0.9880586 | Best = 0.9883215
GA | iter = 50 | Mean = 0.9879811 | Best = 0.9883215
GA | iter = 50 | Mean = 0.9879811 | Best = 0.9883215
GA | iter = 51 | Mean = 0.9880609 | Best = 0.9883215
GA | iter = 52 | Mean = 0.9882879 | Best = 0.9883215
GA | iter = 53 | Mean = 0.9882899 | Best = 0.9883215
GA | iter = 54 | Mean = 0.9882868 | Best = 0.9883215
GA | iter = 55 | Mean = 0.9882873 | Best = 0.9883215
GA | iter = 56 | Mean = 0.9882134 | Best = 0.9883215
GA | iter = 57 | Mean = 0.9882165 | Best = 0.9883215
GA | iter = 58 | Mean = 0.9880655 | Best = 0.9883215
GA | iter = 59 | Mean = 0.9882148 | Best = 0.9883215
GA | iter = 60 | Mean = 0.9882916 | Best = 0.9883215
GA | iter = 61 | Mean = 0.9882962 | Best = 0.9883215
GA | iter = 62 | Mean = 0.9882987 | Best = 0.9883215
GA | iter = 63 | Mean = 0.9883001 | Best = 0.9883215
GA | iter = 64 | Mean = 0.9882986 | Best = 0.9883215
GA | iter = 65 | Mean = 0.9883048 | Best = 0.9883215
GA | iter = 66 | Mean = 0.9883093 | Best = 0.9883215
GA | iter = 67 | Mean = 0.9881499 | Best = 0.9883215
GA | iter = 68 | Mean = 0.9883047 | Best = 0.9883215
GA | iter = 69 | Mean = 0.9883051 | Best = 0.9883215
GA | iter = 70 | Mean = 0.9883033 | Best = 0.9883215
GA | iter = 71 | Mean = 0.9883009 | Best = 0.9883215
GA | iter = 72 | Mean = 0.9883037 | Best = 0.9883215
GA | iter = 73 | Mean = 0.9882266 | Best = 0.9883215
GA | iter = 74 | Mean = 0.9877371 | Best = 0.9883215
GA | iter = 75 | Mean = 0.9882302 | Best = 0.9883215
GA | iter = 76 | Mean = 0.9880754 | Best = 0.9883215
GA | iter = 77 | Mean = 0.9877758 | Best = 0.9883215
GA | iter = 78 | Mean = 0.9883095 | Best = 0.9883215
GA | iter = 79 | Mean = 0.9883110 | Best = 0.9883215
GA | iter = 80 | Mean = 0.9883138 | Best = 0.9883215
GA | iter = 81 | Mean = 0.9883125 | Best = 0.9883215
GA | iter = 82 | Mean = 0.9882326 | Best = 0.9883215
GA | iter = 83 | Mean = 0.9883159 | Best = 0.9883215
GA | iter = 84 | Mean = 0.9883166 | Best = 0.9883215
GA | iter = 85 | Mean = 0.9883158 | Best = 0.9883215
GA | iter = 86 | Mean = 0.9883178 | Best = 0.9883215
GA | iter = 87 | Mean = 0.9883154 | Best = 0.9883215
GA | iter = 88 | Mean = 0.9883192 | Best = 0.9883215
GA | iter = 89 | Mean = 0.9883200 | Best = 0.9883215
GA | iter = 90 | Mean = 0.9877518 | Best = 0.9883215
GA | iter = 91 | Mean = 0.9883188 | Best = 0.9883215
GA | iter = 92 | Mean = 0.9881663 | Best = 0.9883215
GA | iter = 93 | Mean = 0.9882397 | Best = 0.9883215
GA | iter = 94 | Mean = 0.9877491 | Best = 0.9883215
GA | iter = 95 | Mean = 0.9873693 | Best = 0.9883215
GA | iter = 96 | Mean = 0.9883194 | Best = 0.9883215
GA | iter = 97 | Mean = 0.9877529 | Best = 0.9883215
GA | iter = 98 | Mean = 0.9871832 | Best = 0.9883215
GA | iter = 99 | Mean = 0.9877514 | Best = 0.9883215
GA | iter = 100 | Mean = 0.9883192 | Best = 0.9883215
GA | iter = 100 | Mean = 0.9883192 | Best = 0.9883215
GA | iter = 101 | Mean = 0.9882416 | Best = 0.9883215
GA | iter = 102 | Mean = 0.9882431 | Best = 0.9883215
GA | iter = 103 | Mean = 0.9883191 | Best = 0.9883215
GA | iter = 104 | Mean = 0.9883207 | Best = 0.9883215
GA | iter = 105 | Mean = 0.9882435 | Best = 0.9883215
GA | iter = 106 | Mean = 0.9874444 | Best = 0.9883215
GA | iter = 107 | Mean = 0.9867984 | Best = 0.9883215
GA | iter = 108 | Mean = 0.9866452 | Best = 0.9883215
GA | iter = 109 | Mean = 0.9877004 | Best = 0.9883215
GA | iter = 110 | Mean = 0.9875850 | Best = 0.9883215
GA | iter = 111 | Mean = 0.9875909 | Best = 0.9883215
GA | iter = 112 | Mean = 0.9881589 | Best = 0.9883215
GA | iter = 113 | Mean = 0.9880017 | Best = 0.9883215
GA | iter = 114 | Mean = 0.9882339 | Best = 0.9883215
GA | iter = 115 | Mean = 0.9882333 | Best = 0.9883215
GA | iter = 116 | Mean = 0.9883120 | Best = 0.9883215
GA | iter = 117 | Mean = 0.9883169 | Best = 0.9883215
GA | iter = 118 | Mean = 0.9883189 | Best = 0.9883215
GA | iter = 119 | Mean = 0.9883147 | Best = 0.9883215
GA | iter = 120 | Mean = 0.9883189 | Best = 0.9883215
GA | iter = 121 | Mean = 0.9883167 | Best = 0.9883215
GA | iter = 122 | Mean = 0.9883172 | Best = 0.9883215
GA | iter = 123 | Mean = 0.9883189 | Best = 0.9883215
GA | iter = 124 | Mean = 0.9883193 | Best = 0.9883215
GA | iter = 125 | Mean = 0.9882421 | Best = 0.9883215
GA | iter = 126 | Mean = 0.9881671 | Best = 0.9883215
GA | iter = 127 | Mean = 0.9881667 | Best = 0.9883215
GA | iter = 128 | Mean = 0.9883208 | Best = 0.9883215
GA | iter = 129 | Mean = 0.9882430 | Best = 0.9883215
GA | iter = 130 | Mean = 0.9882428 | Best = 0.9883215
GA | iter = 131 | Mean = 0.9882424 | Best = 0.9883215
GA | iter = 132 | Mean = 0.9882427 | Best = 0.9883215
GA | iter = 133 | Mean = 0.9883200 | Best = 0.9883215
GA | iter = 134 | Mean = 0.9883204 | Best = 0.9883215
GA | iter = 135 | Mean = 0.9883204 | Best = 0.9883215
GA | iter = 136 | Mean = 0.9883208 | Best = 0.9883215
GA | iter = 137 | Mean = 0.9883203 | Best = 0.9883215
GA | iter = 138 | Mean = 0.9883199 | Best = 0.9883215
GA | iter = 139 | Mean = 0.9883189 | Best = 0.9883215
GA | iter = 140 | Mean = 0.9883120 | Best = 0.9883215
GA | iter = 141 | Mean = 0.9883162 | Best = 0.9883215
GA | iter = 142 | Mean = 0.9883184 | Best = 0.9883215
GA | iter = 143 | Mean = 0.9883189 | Best = 0.9883215
GA | iter = 144 | Mean = 0.9883185 | Best = 0.9883215
GA | iter = 145 | Mean = 0.9883174 | Best = 0.9883215
GA | iter = 146 | Mean = 0.9877492 | Best = 0.9883215
GA | iter = 147 | Mean = 0.9883148 | Best = 0.9883215
GA | iter = 148 | Mean = 0.9883152 | Best = 0.9883215
GA | iter = 149 | Mean = 0.9883136 | Best = 0.9883215
GA | iter = 150 | Mean = 0.9877440 | Best = 0.9883215
```

다음과 같이 생식이 이루어진다.

```
> colnames(df_trn)[best_var_idx]
[1] "Max_termperature"   "Min_temperature"    "Dewpoint"           "Sea_level_pressure" "Standard_pressure"  "Visibility"         "Wind_speed"
```

뽑힌 변수들은 다음과 같다.

```
> perf_mat
                                input_formula
Weather data(full model)        "df_target ~ ."
Weather Data(Exhaustive Search) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Forward)           "df_target ~ Max_termperature + Min_temperature + Sea_level_pressure + Standard_pressure + Dewpoint + Visibility + Wind_speed"
Weather Data(Backward)          "df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Stepwise)          "df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(GA)                "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
                                Adj.R^2             Time                RMSE               MAE                MAPE
Weather data(full model)        "0.988241162779006" NA                  "1.48096892976632" "1.04811766863543" "2.32711707340505"
Weather Data(Exhaustive Search) "0.988321493056102" "1.25100000000384"  "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Forward)           "0.988321493056102" "0.0599999999976717" "1.47648481469331" "1.04837333694827" "2.32839580198535"
Weather Data(Backward)          "0.988321493056102" "0.0249999999941792" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Stepwise)          "0.988321493056102" "0.0299999999988358" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(GA)                "0.988321493056102" "11.7480000000069"   "1.47648481469331" "1.04837333694827" "2.32839580198534"
```

GA 를 통한 결과도 full model 을 제외한 다른 방법들을 사용한 결과와 일치했다. 다만 수행시간이 매우 길었다.

```
GA | iter = 1 | Mean = 0.8778117 | Best = 0.9882374
GA | iter = 2 | Mean = 0.8913230 | Best = 0.9881741
GA | iter = 3 | Mean = 0.9121073 | Best = 0.9882412
GA | iter = 4 | Mean = 0.8818339 | Best = 0.9882412
GA | iter = 5 | Mean = 0.8603000 | Best = 0.9881837
GA | iter = 6 | Mean = 0.8888332 | Best = 0.9881741
GA | iter = 7 | Mean = 0.8678039 | Best = 0.9882845
GA | iter = 8 | Mean = 0.8617951 | Best = 0.9882845
GA | iter = 9 | Mean = 0.8573258 | Best = 0.9881837
GA | iter = 10 | Mean = 0.8328126 | Best = 0.9881837
GA | iter = 11 | Mean = 0.8401817 | Best = 0.9881837
GA | iter = 12 | Mean = 0.8288108 | Best = 0.9881837
GA | iter = 13 | Mean = 0.8545315 | Best = 0.9882784
GA | iter = 14 | Mean = 0.8361220 | Best = 0.9882784
GA | iter = 15 | Mean = 0.8187965 | Best = 0.9879791
GA | iter = 16 | Mean = 0.8317755 | Best = 0.9879791
GA | iter = 17 | Mean = 0.8133397 | Best = 0.9881403
GA | iter = 18 | Mean = 0.8170988 | Best = 0.9879791
GA | iter = 19 | Mean = 0.8424222 | Best = 0.9874214
GA | iter = 20 | Mean = 0.8756150 | Best = 0.9873713
GA | iter = 21 | Mean = 0.8615453 | Best = 0.9874232
GA | iter = 22 | Mean = 0.8914410 | Best = 0.9874729
GA | iter = 23 | Mean = 0.8993643 | Best = 0.9879696
GA | iter = 24 | Mean = 0.8977944 | Best = 0.9880191
GA | iter = 25 | Mean = 0.9103996 | Best = 0.9880191
GA | iter = 26 | Mean = 0.9245079 | Best = 0.9880191
GA | iter = 27 | Mean = 0.9253157 | Best = 0.9880191
GA | iter = 28 | Mean = 0.9303564 | Best = 0.9880191
GA | iter = 29 | Mean = 0.9317334 | Best = 0.9880191
GA | iter = 30 | Mean = 0.9274963 | Best = 0.9880191
GA | iter = 31 | Mean = 0.9276158 | Best = 0.9880191
GA | iter = 32 | Mean = 0.9267594 | Best = 0.9880191
GA | iter = 33 | Mean = 0.9316361 | Best = 0.9880191
GA | iter = 34 | Mean = 0.9179073 | Best = 0.9880191
GA | iter = 35 | Mean = 0.9261386 | Best = 0.9880191
GA | iter = 36 | Mean = 0.9219508 | Best = 0.9880191
GA | iter = 37 | Mean = 0.9198150 | Best = 0.9880191
GA | iter = 38 | Mean = 0.9199491 | Best = 0.9880191
GA | iter = 39 | Mean = 0.9256756 | Best = 0.9880191
GA | iter = 40 | Mean = 0.9209437 | Best = 0.9880191
GA | iter = 41 | Mean = 0.8921213 | Best = 0.9880191
GA | iter = 42 | Mean = 0.8872591 | Best = 0.9880191
GA | iter = 43 | Mean = 0.8914628 | Best = 0.9880191
GA | iter = 44 | Mean = 0.8846093 | Best = 0.9882030
GA | iter = 45 | Mean = 0.8618356 | Best = 0.9880191
GA | iter = 46 | Mean = 0.8377397 | Best = 0.9880191
GA | iter = 47 | Mean = 0.8581909 | Best = 0.9880191
GA | iter = 48 | Mean = 0.8518414 | Best = 0.9880191
GA | iter = 49 | Mean = 0.9082083 | Best = 0.9880009
GA | iter = 50 | Mean = 0.8787510 | Best = 0.9879768
> perf_mat
                                input_formula
Weather data(full model)        "df_target ~ ."
Weather Data(Exhaustive Search) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Forward)           "df_target ~ Max_termperature + Min_temperature + Sea_level_pressure + Standard_pressure + Dewpoint + Visibility + Wind_speed"
Weather Data(Backward)          "df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(Stepwise)          "df_target ~ Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
Weather Data(GA)                "df_target ~  Max_termperature + Min_temperature + Dewpoint + Standard_pressure + Visibility + Wind_speed"
                                Adj.R^2             Time                RMSE               MAE                MAPE
Weather data(full model)        "0.988241162779006" NA                  "1.48096892976632" "1.04811766863543" "2.32711707340505"
Weather Data(Exhaustive Search) "0.988321493056102" "1.25100000000384"  "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Forward)           "0.988321493056102" "0.0599999999976717" "1.47648481469331" "1.04837333694827" "2.32839580198535"
Weather Data(Backward)          "0.988321493056102" "0.0249999999941792" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(Stepwise)          "0.988321493056102" "0.0299999999988358" "1.47648481469331" "1.04837333694827" "2.32839580198534"
Weather Data(GA)                "0.987976801104931" "3.86499999999069"   "1.46580010666664" "1.07378118786553" "2.43155657465721"
```

maxiter 를 50 으로, elitism 을 0 으로 바꾸어 한번 더 실행해 본 결과는 다음과 같다. 시간은 약 4 초로 전보다 매우 줄었지만, Adjusted R^2 는 약간 줄었고, MAE, MAPE 가 늘었다.

선택된 변수는 모두 6 개로 위에서 선택된 변수에서 Sea level 을 제외한 변수들이다. 변수가 1 개가 줄었기 때문에 모델 복잡도는 줄었으며, Adjusted R^2 가 차이가 많이 안나기 때문에 위에서 설정한 것보다 지금 선택된 변수를 사용할 것이다.

하지만 GA 보다는 시간이 적게 걸리고 performance 도 좋은 3 번의 3 가지 방법을 선택하여 변수선택을 할 것이다.

[Q5] Genetic Algorithm 에서 변경 가능한 하이퍼 파라미터들(population size, Cross-over rate, Mutation rate 등)에 대해 최소한 세 가지 이상의 후보 값들을 선정하고 각 조합에 대한 변수 선택 결과를 비교해 보시오. 최종 결과에 가장 큰 영향을 미치는 하이퍼파라미터는 무엇으로 나타났는가? 왜 그런 결과가 나타났다고 생각하는가?

```
#[Q5]
gmat <- matrix(0, nrow = 10, ncol = 6)
rownames(gmat) <- c("population Size(100)","population Size(20)","Cross-over rate(0.5)","Cross-over rate(0.2)"
,"mutation(0.005)","mutation(0.08)","maxiter(150)","maxiter(50)","elitism(0)","elitism(2)")
colnames(gmat) <- c("input_formula","Adj.R^2","Time","RMSE", "MAE", "MAPE")

pS <- c(100,20,100,100,100,100,100,100,100,100)
pco <- c(0.5,0.5,0.5,0.2,0.5,0.5,0.5,0.5,0.5,0.5)
pm <- c(0.005,0.005,0.005,0.005,0.005,0.005,0.08,0.005,0.005,0.005,0.005)
mt <- c(150,150,150,150,150,150,150,50,150,150)
el <- c(0,0,0,0,0,0,0,0,0,2)

for(i in 1:10){
  start_time <- proc.time()
  GA_F1 <- ga(type = "binary", fitness = fit_F1, nBits = ncol(x),
              names = colnames(x), popSize = pS[i], pcrossover = pco[i],
              pmutation = pm[i], maxiter = mt[i], elitism = el[i], seed = 123)
  end_time <- proc.time()
  gmat[i,3] <- (end_time - start_time)[3]
  best_var_idx <- which(GA_F1@solution == 1)
  colnames(df_trn)[best_var_idx]

  tmp_x <- paste(colnames(df_trn)[best_var_idx], collapse=" + ")
  tmp_xy <- paste("df_target ~ ", tmp_x, collapse = "")
  gmat[i,1] <- tmp_xy
  as.formula(tmp_xy) #string을 formula라는 새로운 데이터 형태로 바꿈(작동시키기위해)

  ga_model <- lm(tmp_xy, data = df_trn)
  summary(ga_model)
  gmat[i,2] <- summary(ga_model)$adj.r.squared

  mlr_df_haty5 <- predict(ga_model, newdata = df_val)
  gmat[i,4:6] <- perf_eval_reg(df_val$df_target, mlr_df_haty5)
}

gmat
```

총 5 개의 parameter 에 대해 각각 2 가지 값으로 총 10 가지의 경우를 측정해보았다.

교수님께서 말씀하신 보통 많이 쓰인다는 값을 표준으로 하고, 그것보다 작거나 큰 경우로 바꾸어서 실행해보았다.

pop Size 100 을 표준값으로, prossover 를 0,5, pmutation 을 0.005 로 maxiter 를 150, elitism 을 0 으로 , 바꿔주는 값을 pop Size 20, pcrossover 0.2, pmutation 0.08(크게), maxiter 를 50, elitism 을 2 로 하였다.

그에 따른 결과는 아래와 같다.

```
> gmat
                    input_formula
population Size(100) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
population Size(20)  "df_target ~  Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed"
Cross-over rate(0.5) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
Cross-over rate(0.2) "df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
mutation(0.005)      "df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
mutation(0.08)       "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
maxiter(150)         "df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
maxiter(50)          "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility"
elitism(0)           "df_target ~  Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed"
elitism(2)           "df_target ~  Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed"
                     Adj.R^2             Time               RMSE               MAE                MAPE
population Size(100) "0.988203037830603" "11.3879999999917" "1.49131318092423" "1.07788544422336" "2.3746966331086"
population Size(20)  "0.980700020649472" "2.94700000000012" "2.07311495595073" "1.56348640821799" "3.56811371004538"
Cross-over rate(0.5) "0.988203037830603" "10.8610000000044" "1.49131318092423" "1.07788544422336" "2.3746966331086"
Cross-over rate(0.2) "0.988203037830603" "5.18300000000454" "1.49131318092423" "1.07788544422336" "2.3746966331086"
mutation(0.005)      "0.988203037830603" "10.9470000000001" "1.49131318092423" "1.07788544422336" "2.3746966331086"
mutation(0.08)       "0.988239242180714" "11.2219999999943" "1.48700756746147" "1.07670534859699" "2.37364887331092"
maxiter(150)         "0.988203037830603" "11.0020000000077" "1.49131318092423" "1.07788544422336" "2.3746966331086"
maxiter(50)          "0.988272751906896" "3.52000000000407" "1.49147746130414" "1.08410118807141" "2.38635563854184"
elitism(0)           "0.988203037830603" "11.0249999999942" "1.49131318092423" "1.07788544422336" "2.3746966331086"
elitism(2)           "0.988321493056102" "11.4580000000133" "1.47648481469331" "1.04837333694827" "2.32839580198534"
```

>population Size

100: Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed 로 8 개의 변수가 사용되었으며, Adjusted R^2 가 0.9882 로 높았다.

20: Max_termperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed + Max_wind_speed 로 8 개의 변수가 사용되었으며, 위와 비교하여 Min_tem -> Wind_speed 만 바뀌었다. poplution size 가 작기 때문인지 시간은 엄청 줄었으며, (약 3 초, 100 일 때 11 초) 하지만 adjusted R^2 가 0.98 로 줄었고, RMSE, MAE, MAPE 가 1.49-> 2.07, 1.1 -> 1.56, 2.37->3.57 로 크게 증가했다. 따라서 population size 가 작을수록 에러가 커질 것을 예상할 수 있다.

>Cross over rate

0.5, 0.2: Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed 로 8 개의 변수가 사용되었으며, 이는 population size 의 첫번쨰 경우와 같다. cross over rate 는 결과에 많은 영향을 미치지 않는 것으로 보인다. 하지만 작을수록 시간이 적게 듬(약 5 초, 0.5 일때 11 초)을 알 수 있다.

> pMutation

0.005: Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed 로 8 개 변수가, cross over rate 의 결과와 같게 나타났다.

0.08: Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed 로 7 개 변수가 사용되었다. Adjusted R^2 값이

0.98823 으로 (0.05 일 때 0.9882) 매우 미세하게 큼을 볼 수 있다. 또한 RMSE, MAE, MAPE 가 1.48700756746147" "1.07670534859699" "2.37364887331092"으로 아주 미세하게 0.05 일때보다 작은 것을 알 수 있다. 돌연변이 확률이 크면 미세한 영향을 미친다는 것을 알 수 있다. 선택된 변수 개수에 이 경우에 영향을 미쳤다.

> Maxiter

150: Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed 로 cross over rate 와 같은 결과이다.

50: Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility 로 6 개의 변수가 선택되었다. 이 경우 변수 개수가 적음에도 Adjusted R^2 가 0.98827 로 큰 값을 보였고(미세한 차이(큼)), 적은 시간이 걸렸으며, RMSE, MAE, MAPE 가 "1.49147746130414" "1.08410118807141" "2.38635563854184" 로 150 일 때랑 적은 차이를 보였다. maxiter 가 100 이상이어야 한다고 수업시간에 교수님께서 말씀하셨는데 이 경우 50 일 때도 많은 차이는 보이지 않으며 시간대비 효율적이라고 생각하여 데이터에 따라 다른 기준이 적용될 수도 있다고 생각하였다.

>elitism

0: Max_termperature + Min_temperature + Dewpoint + Precipitation + Sea_level_pressure + Standard_pressure + Visibility + Max_wind_speed 로 cross over rate 와 같은 결과이다.

2: Max_termperature + Min_temperature + Dewpoint + Sea_level_pressure + Standard_pressure + Visibility + Wind_speed 로 총 7 개의 변수가 선택되었으며, Adjusted R^2 는 0.9883 으로 모든 경우에서 가장 좋은 performance 를 보인다. RMSE, MAE, MAPE 도 "1.47648481469331" "1.04837333694827" "2.32839580198534" 로 가장 작다. 우수한 유전자는 계속 남아있게(불멸) 하는 수가 많을 수록 좋은 performance 를 보일 것이라고 예측된다.

>> 최종 결과에 가장 큰 영향을 미치는 parameter 는 popsize 같다. 물론 임의로 설정한 파라메타 값이 달라짐에 따라 영향이 다르겠지만 위에서 설정한 대로 해석을 한다면 popSize 가 영향을 많이 미친다. 한 세대의 염색체를 적게 가져가게 된다면 데이터의 샘플의 개수가 적어지는 것과 비슷한 효과를 가져오게 될 것이라고 예상되기 때문이다.