Assignment 4.2 - Supervised Regression Models

Arpan Sapkota

2023-06-01

Do as follows in R Studio and submit/attach the compiled PDF report file with codes and outputs here:

1. Generate a 1000 random data with 10 variables [five continuous: age (18 to 90 years), height (150 - 180 cm), weight (50 - 90 kg), income (10000 - 200000), diastolic blood pressure (70 - 170 mm Hg) and five categorical: sex (male/female), education (no education, primary, secondary, tertiary), place of residence (rural/urban), socio-economic status (low/medium/high) and exercise (yes/no)] using set.seed(your roll number and save it as SR object

```r
set.seed(07)
age <- sample(18:90, 1000, replace = TRUE)
height <- sample(150:180, 1000, replace = TRUE)
weight <- sample(50:90, 1000, replace = TRUE)
income <- sample(10000:200000, 1000, replace = TRUE)
blood_pressure <- sample(70:170, 1000, replace = TRUE)
sex <- sample(c("male", "female"), 1000, replace = TRUE)
education <- sample(c("no education", "primary", "secondary", "tertiary"), 1000, replace = TRUE)
residence <- sample(c("rural", "urban"), 1000, replace = TRUE)
socio_status <- sample(c("low", "medium", "high"), 1000, replace = TRUE)
exercise <- sample(c("yes", "no"), 1000, replace = TRUE)

sex<-as.factor(sex)
education<-as.factor(education)
residence<-as.factor(residence)
socio_status<-as.factor(socio_status)
exercise<-as.factor(exercise)

SR <- data.frame(age, height, weight, income, blood_pressure, sex = factor(sex), education, residence, 

head(SR)
```

```
##   age height weight income blood_pressure    sex    education residence
## 1  59    154     76 167366            144   male no education     rural
## 2  48    169     76 164480             98 female     tertiary     urban
## 3  83    167     89 114245            117   male no education     urban
## 4  32    156     90  44150            140 female no education     rural
## 5  25    169     90 176693             74 female      primary     rural
## 6  84    153     88 129242            134   male    secondary     urban
##   socio_status exercise
## 1       medium      yes
## 2         high       no
## 3       medium      yes
## 4         high      yes
## 5         high       no
## 6          low      yes
```

2. Randomly split the SR object data as SR.train (70%) and SR.test (30%) with replacement sampling and fit multiple linear regression with diastolic blood pressure as dependent variable and rest of variables as independent variable and get fit indices (R-Square, MSE, RMSE and MAE) for the SR.test data

```
#Split data and fit multiple linear regression
set.seed(07)
train_indices <- sample(1:nrow(SR), 0.7 * nrow(SR), replace = TRUE)
SR.train <- SR[train_indices, ]
SR.test <- SR[-train_indices, ]

lm_model <- lm(blood_pressure ~ ., data = SR.train)
lm_predictions <- predict(lm_model, newdata = SR.test)

# Calculate fit indices
library(Metrics)
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following objects are masked from 'package:Metrics':
##
##     precision, recall
```

```
R_square <- R2(lm_predictions, SR.test$blood_pressure)
R_square
```

```
## [1] 2.883933e-05
```

```
RMSE <- RMSE(lm_predictions, SR.test$blood_pressure)
RMSE
```

```
## [1] 29.81087
```

```
MAE <- MAE(lm_predictions, SR.test$blood_pressure)
MAE
```

```
## [1] 25.22291
```

3. Fit the multiple linear regression model with Leave One Out Cross-Validation, k-fold cross validation, repeated k-fold cross validation methods and get fit indices for SR.test data and, compare the fit indices of supervised regression models fitted in step 2 and 3 above with careful interpretation

```
# Fit regression models with different cross-validation methods

# Leave One Out Cross-Validation (LOOCV)
lm_model_loocv <- train(blood_pressure ~ ., data = SR.train, method = "lm", trControl = trainControl(me
lm_predictions_loocv <- predict(lm_model_loocv, newdata = SR.test)

# Calculate fit indices for LOOCV
R_square_loocv <- R2(lm_predictions_loocv, SR.test$blood_pressure)
R_square_loocv
```

```
## [1] 2.883933e-05
```

```
RMSE_loocv <- RMSE(lm_predictions_loocv, SR.test$blood_pressure)
RMSE_loocv
```

```
## [1] 29.81087
```

```
MAE_loocv <- MAE(lm_predictions_loocv, SR.test$blood_pressure)
MAE_loocv
```

```
## [1] 25.22291
```

```
# k-fold Cross-Validation (k = 10)
lm_model_kfold <- train(blood_pressure ~ ., data = SR.train, method = "lm", trControl = trainControl(me
lm_predictions_kfold <- predict(lm_model_kfold, newdata = SR.test)

# Calculate fit indices for k-fold CV
R_square_kfold <- R2(lm_predictions_kfold, SR.test$blood_pressure)
R_square_kfold
```

```
## [1] 2.883933e-05
```

```
RMSE_kfold <- RMSE(lm_predictions_kfold, SR.test$blood_pressure)
RMSE_kfold
```

```
## [1] 29.81087
```

```
MAE_kfold <- MAE(lm_predictions_kfold, SR.test$blood_pressure)
MAE_kfold
```

```
## [1] 25.22291
```

```r
# Repeated k-fold Cross-Validation (k = 10, repeats = 3)
lm_model_repeated_kfold <- train(blood_pressure ~ ., data = SR.train, method = "lm", trControl = trainC
lm_predictions_repeated_kfold <- predict(lm_model_repeated_kfold, newdata = SR.test)

# Calculate fit indices for repeated k-fold CV
R_square_repeated_kfold <- R2(lm_predictions_repeated_kfold, SR.test$blood_pressure)
R_square_repeated_kfold
```

```
## [1] 2.883933e-05
```

```r
RMSE_repeated_kfold <- RMSE(lm_predictions_repeated_kfold, SR.test$blood_pressure)
RMSE_repeated_kfold
```

```
## [1] 29.81087
```

```r
MAE_repeated_kfold <- MAE(lm_predictions_repeated_kfold, SR.test$blood_pressure)
MAE_repeated_kfold
```

```
## [1] 25.22291
```

A higher R-squared value indicates a better fit, while lower values of MSE, RMSE, and MAE indicate better predictive accuracy. We get

In Step 2:

R_square = 2.883933e-05

RMSE = 29.810872

MAE = 25.22291

in Step 3:

R_square_loocv = 2.883933e-05

RMSE_loocv = 29.81087

MAE_loocv = 25.22291

R_square_kfold = 2.883933e-05

RMSE_kfold = 29.81087

MAE_kfold = 25.22291

R_square_repeated_kfold= 2.883933e-05

RMSE_repeated_kfold = 29.81087

MAE_repeated_kfold = 25.22291

The R-squared value (R_square), root mean squared error (RMSE), and mean absolute error (MAE) are identical for each method. The performance of the multiple linear regression model, in terms of predicting diastolic blood pressure, is consistent across different cross-validation techniques. The R-squared value being close to zero indicates that the model explains very little of the variance in the dependent variable. The RMSE and MAE values indicate the average prediction error of approximately 29.81 mm Hg and 25.22 mm Hg, respectively.

Given that the fit indices are the same across different cross-validation methods, it indicates that the model's performance is relatively stable and not significantly affected by the choice of cross-validation technique.

However, it's important to note that the model's overall performance, as reflected in the fit indices, is relatively poor, as indicated by the low R-squared value and the relatively high RMSE and MAE values.

It might be worth considering alternative regression models, such as KNN regression, decision tree regression, SVM regression, or neural network regression, to see if they can improve the prediction performance on the test data.

4. Fit KNN regression, Decision Tree regression, SVM regression and Neural Network regression using the same dependent and independent variables, get and compare fit indices of these models for SR.test data

```
# KNN regression
library(caret)
knn_model <- train(blood_pressure ~ ., data = SR.train, method = "knn", trControl = trainControl(method
knn_predictions <- predict(knn_model, newdata = SR.test)
knn_R_square <- R2(knn_predictions, SR.test$blood_pressure)
knn_R_square
```

```
## [1] 0.00281484
```

```
knn_RMSE <- RMSE(knn_predictions, SR.test$blood_pressure)
knn_RMSE
```

```
## [1] 30.38297
```

```
knn_MAE <- MAE(knn_predictions, SR.test$blood_pressure)
knn_MAE
```

```
## [1] 25.71751
```

```
# Decision tree regression
library(rpart)
tree_model <- rpart(blood_pressure ~ ., data = SR.train)
tree_predictions <- predict(tree_model, newdata = SR.test)
tree_R_square <- R2(tree_predictions, SR.test$blood_pressure)
tree_R_square
```

```
## [1] 0.0001422495
```

```
tree_RMSE <- RMSE(tree_predictions, SR.test$blood_pressure)
tree_RMSE
```

```
## [1] 31.49071
```

```
tree_MAE <- MAE(tree_predictions, SR.test$blood_pressure)
tree_MAE
```

```
## [1] 26.37546
```

```
# SVM regression
library(e1071)
svm_model <- svm(blood_pressure ~ ., data = SR.train)
svm_predictions <- predict(svm_model, newdata = SR.test)
svm_R_square <- R2(svm_predictions, SR.test$blood_pressure)
svm_R_square
```

```
## [1] 0.0001146666
```

```
svm_RMSE <- RMSE(svm_predictions, SR.test$blood_pressure)
svm_RMSE
```

```
## [1] 31.78105
```

```
svm_MAE <- MAE(svm_predictions, SR.test$blood_pressure)
svm_MAE
```

```
## [1] 26.62667
```

```
# Neural network regression
library(nnet)
nn_model <- nnet(blood_pressure ~ ., data = SR.train, size = 5, linout = TRUE)
```

```
## # weights:  71
## initial  value 10359454.233829
## final  value 572021.000000
## converged
```

```
nn_predictions <- predict(nn_model, newdata = SR.test)
nn_R_square <- R2(nn_predictions, SR.test$blood_pressure)
```

```
## Warning in cor(obs, pred, use = ifelse(na.rm, "complete.obs", "everything")):
## the standard deviation is zero
```

```
nn_R_square
```

```
##      [,1]
## [1,]   NA
```

```
nn_RMSE <- RMSE(nn_predictions, SR.test$blood_pressure)
nn_RMSE
```

```
## [1] 29.34516
```

```
nn_MAE <- MAE(nn_predictions, SR.test$blood_pressure)
nn_MAE
```

```
## [1] 25.14845
```

In Step 4

knn_R_square = 0.00281484

knn_RMSE = 30.38297

knn_MAE = 25.71751

tree_R_square = 0.0001422495

tree_RMSE = 31.49071

tree_MAE = 26.375465

svm_R_square = 0.0001146666

svm_RMSE = 31.78105

svm_MAE = 26.62667

nn_R_square = NA

nn_RMSE = 29.34516

nn_MAE = 25.14845

It appears that the KNN regression model performs slightly better than the other models, as it has a higher R-squared value and lower RMSE and MAE values. However, R-squared values for all the models are relatively low, indicating that they explain a small proportion of the variance in the blood pressure.

5. Which supervised regression model is the best model for doing prediction in the SR.test data? Why?

The KNN regression model appears to perform slightly better than the other models in terms of R-squared, root mean squared error (RMSE), and mean absolute error (MAE). It has a higher R-squared value and lower RMSE and MAE values compared to the other models.

However, the overall performance of all the models is relatively weak, as indicated by the low R-squared values and relatively high RMSE and MAE values. This suggests that none of the models have a strong ability to accurately predict diastolic blood pressure based on the given set of independent variables.

6. Predict diastolic blood pressure of a person with 50 years, 175mm height, 80 kg weight, 90000 income, male, tertiary level education, living in urban area, medium socio-economic status and no exercise and interpret the result carefully

```r
# Predict diastolic blood pressure using the KNN regression model
new_data <- data.frame(
  age = 50,
  height = 175,
  weight = 80,
  income = 90000,
  sex = "male",
  education = "tertiary",
  residence = "urban",
  socio_status = "medium",
  exercise = "no"
)
```

```r
knn_prediction <- predict(knn_model, newdata = new_data)
knn_prediction
```

```
## [1] 122.2222
```

The predicted diastolic blood pressure for a person with the given characteristics is 122.2222 mm Hg. The predicted diastolic blood pressure of 122.2222 mm Hg should be interpreted as an estimate based on the KNN regression model. However, due to model limitations and individual variations, it should be considered as an approximation rather than an absolute value. Additional medical assessments are recommended for a comprehensive evaluation of an individual's health status.

7. Write a reflection of the assignment on your own words focusing on "what did I learn with this assignment?"

1. Data Generation: I learned how to generate random data with specific characteristics using R, including both continuous and categorical variables.
2. Model Fitting and Evaluation: I explored multiple linear regression models and evaluated their performance using fit indices such as R-squared, mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE). I also learned about different cross-validation techniques like leave-one-out cross-validation, k-fold cross-validation, and repeated k-fold cross-validation.
3. Model Comparison: I compared the fit indices of different regression models, including linear regression, KNN regression, decision tree regression, SVM regression, and neural network regression. This allowed me to assess the relative performance of these models in predicting diastolic blood pressure.
4. Interpretation of Results: I practiced interpreting the results of regression models, considering the limitations of the models and the implications of the predictions. I learned to be cautious in interpreting the results and to view them as estimates rather than definitive values.
5. Model Selection: I explored which regression model performed best for the given data based on the fit indices. However, I also recognized the importance of considering other factors such as interpretability, computational efficiency, and scalability in determining the most suitable model for prediction.