Performance comparison of Regression Model, Regression Tree, Bagging Tree and Random Forest

Machine Learning and Applications

Tufan Bostan

01/05/2021

In this work, bunch of model performance will be compared using R. Data to be used here is "dragons" from “DALEX” package. Here, the life time of dragons was tried to be estimated using the data obtained from fossils. The data contains no missing values. First, some descriptive statistics will be obtained and interpreted for the data. Then, the data will be split into two parts which are “Train” and “Test”. Then using the “Train” part of data models will be trained. Finally, performance of the models on train and test sets will be interpreted for all model and their performances will be compared for this data.

Data Structure

First of all, “dragons” data introduced to R. The “dragons” data assigned to “data”. All variables were correctly classified. First 6 observations of the data are listed below. Then, structure of the data displayed. (`str(data)`) After that, data summary displayed to gather more information about variables. (`summary(data)`)

```
data <- DALEX::dragons
head(data)
```

```
##   year_of_birth   height   weight scars colour year_of_discovery
## 1         -1291 59.40365 15.32391     7    red              1700
## 2          1589 46.21374 11.80819     5    red              1700
## 3          1528 49.17233 13.34482     6    red              1700
## 4          1645 48.29177 13.27427     5  green              1700
## 5            -8 49.99679 13.08757     1    red              1700
## 6           915 45.40876 11.48717     2    red              1700
##   number_of_lost_teeth life_length
## 1                   25   1368.4331
## 2                   28   1377.0474
## 3                   38   1603.9632
## 4                   33   1434.4222
## 5                   18    985.4905
## 6                   20    969.5682

str(data)

## 'data.frame':    2000 obs. of  8 variables:
##  $ year_of_birth       : num  -1291 1589 1528 1645 -8 ...
##  $ height              : num  59.4 46.2 49.2 48.3 50 ...
##  $ weight              : num  15.3 11.8 13.3 13.3 13.1 ...
##  $ scars               : num  7 5 6 5 1 2 3 7 6 32 ...
##  $ colour              : Factor w/ 4 levels "black","blue",..: 4 4 4 3 4 4 1 2 4 4 ...
##  $ year_of_discovery   : num  1700 1700 1700 1700 1700 1700 1700 1700 1700 1700 ...
##  $ number_of_lost_teeth: num  25 28 38 33 18 20 28 29 2 22 ...
##  $ life_length         : num  1368 1377 1604 1434 985 ...

summary(data)
```

```
##  year_of_birth          height          weight          scars
##  Min.   :-1999.00   Min.   :29.94   Min.   : 7.524   Min.   : 0.00
##  1st Qu.:-1017.25   1st Qu.:44.98   1st Qu.:12.043   1st Qu.: 3.00
##  Median :  -66.00   Median :49.68   Median :13.385   Median : 7.00
##  Mean   :  -79.54   Mean   :50.05   Mean   :13.493   Mean   : 9.94
##  3rd Qu.:  858.00   3rd Qu.:54.50   3rd Qu.:14.826   3rd Qu.:14.00
##  Max.   : 1800.00   Max.   :76.28   Max.   :22.372   Max.   :76.00
##    colour     year_of_discovery number_of_lost_teeth  life_length
##  black:  27   Min.   :1700      Min.   : 0           Min.   : 511.2
##  blue : 576   1st Qu.:1724      1st Qu.:10           1st Qu.:1034.9
##  green: 370   Median :1751      Median :20           Median :1314.8
##  red  :1027   Mean   :1750      Mean   :20           Mean   :1371.0
##               3rd Qu.:1776      3rd Qu.:30           3rd Qu.:1614.4
##               Max.   :1800      Max.   :40           Max.   :3952.7
```

The data has 8 variables and 2000 observations. There is a factor and 7 numeric variables. Here, the target variable is “life_length”. The other variables are the independent variables, to be used to predict the “life_length”.

- life_length represents life length of the dragon
- scars represents number of scars.
- colour represents colour of the dragon.
- height represents height of the dragon in yards.
- weight represents weight of the dragon in tons.
- year_of_birth represents year in which the dragon was born. (Negative year means year BC, eg: -1200 = 1201 BC)
- year_of_discovery represents year in which the dragon was found.
- number_of_lost_teeth represents number of teeth that the dragon lost.

Here, the information about the variables in the data is listed as found in its source. Descriptive statistics of all variables are given. year_of_birth has -79,54 mean and ranges -1999 to 1800. height has 50,05 mean and ranges 29,94 to 76,28. weight has 13,493 mean and ranges 7,524 to 22,372. Scars has 9,94 mean and ranges 0 to 76. colour is a factor and it has 4 levels which are black, blue, green and red and levels observed 27 times, 576 times and 370 times and 1027 times respectively. year_of_discovery has 1750 mean and ranges 1700 to 1800. nuber_of_lost_teeth has 20 mean and ranges 0 to 40. Finally, life_length is the target variable and it has 1371 mean and ranges 511.2 to 3952.7.

Splitting Data

A sample were created using "dragons" data. This sample has two parts one of them is going to be used for training to model and the other part for testing to the model. These are assigned to variables named `train` and `test`, respectively. Also, using `nrow(train)` and `nrow(test)` how many observations they contain is shown.

```
library(caret)

set.seed(2380)
index <- createDataPartition(data$year_of_birth, p = 0.75,
list = FALSE, times = 1)
train <- data[index,]
test <- data[-index,]
nrow(train)

## [1] 1502

nrow(test)
```

```
## [1] 498
```

`train` has 1502 and observations and `test` has 498 observations. After this process, models are ready to be trained.

Linear Regression Model

```
#Traing Model
model_LM <- lm(life_length~. ,data=train)
#Performance for Train
predicted_train_LM <- predict(model_LM, train)
rmse_train_LM <-sqrt(mean((predicted_train_LM - train$life_length) ^ 2))

#Performance for Test
predicted_test_LM <- predict(model_LM, test)
rmse_test_LM <- sqrt(mean((predicted_test_LM - test$life_length) ^ 2))

cat("test_rmse:", rmse_test_LM,"\n")

## test_rmse: 40.69285

cat("train_rmse:", rmse_train_LM)

## train_rmse: 40.72499
```

In this part, the Linear Regression Model were trained and the model errors were calculated. This is square root of mean square error (RMSE). RMSE of Train is 40,72499 and for Test this value 40,69285. Test RMSE is smaller than Train RMSE. Even model RMSE on train and test sets, still there might be underfitting problem. Getting more training data, increasing the size or number of parameters in the model or increasing the complexity of the model may decrease or fix this problem.

Decision Tree

```
# Training a regression tree on the dragons data
library(rpart)

## Warning: package 'rpart' was built under R version 4.0.5

library(rpart.plot)

## Warning: package 'rpart.plot' was built under R version 4.0.5

model_DT <- rpart(life_length ~. , method = "anova", data = train)

#Performance for test
predicted_test_DT <- predict(model_DT, test)
rmse_test_DT <- sqrt(mean((predicted_test_DT - test$life_length) ^ 2))

#Performance for train
predicted_train_DT <- predict(model_DT, train)
rmse_train_DT <- sqrt(mean((predicted_train_DT - train$life_length) ^ 2))

cat("test_rmse_dt:", rmse_test_DT,"\n")

## test_rmse_dt: 157.3396

cat("train_rmse_dt:", rmse_train_DT)

## train_rmse_dt: 153.8834
```

In this part, the model was trained by Decision Tree method and the model errors were calculated. RMSE of Train is 153,3396 and for Test this value 157.8834. Train RMSE is smaller than Test

RMSE. Even model RMSE on train and test sets, still there might be overfitting problem. Training whit more data, removing some features or training an ensemble model can be applied to remove overfitting problem.

Training Begging Tree

```
library(randomForest)

model_BT <- randomForest(life_length~. , data=train, mtry= 7)

#Performance for test
predicted_test_BT <- predict(model_BT, test)
rmse_test_BT <- sqrt(mean((predicted_test_BT - test$life_length) ^ 2))

#Performance for train
predicted_train_BT <- predict(model_BT, train)
rmse_train_BT <- sqrt(mean((predicted_train_BT - train$life_length) ^ 2))

cat("test_rmse_dt:", rmse_test_BT,"\n")

## test_rmse_dt: 49.97751

cat("train_rmse_dt:", rmse_train_BT)

## train_rmse_dt: 23.76681
```

In this part, the model was trained by Begging Tree method and the model errors were calculated. RMSE of Train is 23,76681 and for Test this value 49,97751. Train RMSE is smaller than Test RMSE. According to RMSE, there might be overfitting problem. Training whit more data, removing some features or training an ensemble model can be applied to remove overfitting problem.

Random Forest

```
model_RF <- randomForest(life_length ~ ., data = train)
#Performance for test
predicted_test_RF <- predict(model_RF, test)
rmse_test_RF <- sqrt(mean((predicted_test_RF - test$life_length) ^ 2))

#Performance for train
predicted_train_RF <- predict(model_RF, train)
rmse_train_RF <- sqrt(mean((predicted_train_RF - train$life_length) ^ 2))

cat("test_rmse_dt:", rmse_test_RF,"\n")

## test_rmse_dt: 128.7516

cat("train_rmse_dt:", rmse_train_RF)

## train_rmse_dt: 52.95512
```

Finally, RMSE were calculated for Random Forest to see performance of the model on train and test sets. RMSE of Train is 52,95512 and for Test this value 128,7516. Train RMSE is way smaller than Test RMSE so probably there is overfitting problem. Training whit more data, removing some features or training an ensemble model can be applied to fix or decrease differences between train and test error.

Comparing Models Performances

```
data.frame(
  "TEST" = c(rmse_test_BT,rmse_test_DT,rmse_test_LM,rmse_test_RF),
  "TRAIN" = c(rmse_train_BT,rmse_train_DT,rmse_train_LM,rmse_train_RF),
  row.names = c("BEGGING TREE", "DECISION TREE","LINEAR MODEL","RANDOM FOREST")
)

##                     TEST     TRAIN
## BEGGING TREE   49.97751  23.76681
## DECISION TREE 157.33959 153.88342
## LINEAR MODEL   40.69285  40.72499
## RANDOM FOREST 128.75157  52.95512
```

For better interpretations Errors of all models were displayed on a table. It is possible to say that the lower the error, the higher the predictive performance of the model. Based on this, if the error of the model is lower than the other model, it can be concluded that the model with low error is better. It is clear that the Linear regression model has the lowest error scores. In addition, the error scores on the test and train are very close to each other, so the probability of experiencing underfitting and overfitting problems is very low. As a result, it is concluded that the Linear regression model is the one that is successful in predicting the lifetime of dragons from 4 models.