Лабораторна робота №5. Побудова регресійних моделей

[Іващенко А.В.]

2024-11-25

Table of Contents

### 1. Исследование данных

Сначала загрузим данные и рассмотрим их структуру:

# Загрузка данных  
library(dplyr)

##   
## Присоединяю пакет: 'dplyr'

## Следующие объекты скрыты от 'package:stats':  
##   
## filter, lag

## Следующие объекты скрыты от 'package:base':  
##   
## intersect, setdiff, setequal, union

library(randomForest)

## randomForest 4.7-1.2

## Type rfNews() to see new features/changes/bug fixes.

##   
## Присоединяю пакет: 'randomForest'

## Следующий объект скрыт от 'package:dplyr':  
##   
## combine

library(doParallel)

## Загрузка требуемого пакета: foreach

## Загрузка требуемого пакета: iterators

## Загрузка требуемого пакета: parallel

library(engsoccerdata)  
data("england")  
  
# Просмотр первых строк  
head(england)

## Date Season home visitor FT hgoal  
## 1 1888-09-08 1888 Bolton Wanderers Derby County 3-6 3  
## 2 1888-09-08 1888 Everton Accrington F.C. 2-1 2  
## 3 1888-09-08 1888 Preston North End Burnley 5-2 5  
## 4 1888-09-08 1888 Stoke City West Bromwich Albion 0-2 0  
## 5 1888-09-08 1888 Wolverhampton Wanderers Aston Villa 1-1 1  
## 6 1888-09-15 1888 Aston Villa Stoke City 5-1 5  
## vgoal division tier totgoal goaldif result  
## 1 6 1 1 9 -3 A  
## 2 1 1 1 3 1 H  
## 3 2 1 1 7 3 H  
## 4 2 1 1 2 -2 A  
## 5 1 1 1 2 0 D  
## 6 1 1 1 6 4 H

Мы получим таблицу, содержащую столбцы: Season, Date, home, visitor, hgoal, vgoal, tier, и другие.

### 2. Постановка задачи

Будем прогнозировать количество голов домашней команды (hgoal) на основе:

Среднего количества забитых голов домашней команды в сезоне (mean\_home\_goals). Разницы забитых и пропущенных голов домашней команды в сезоне (goal\_diff\_home). Среднего количества пропущенных голов гостевой командой (mean\_visitor\_goals). ### 3. Подготовка данных Создадим агрегированные метрики и добавим их в основной набор данных:

library(dplyr)  
  
# Агрегированные метрики  
team\_stats <- england %>%  
 group\_by(Season, home) %>%  
 summarize(  
 mean\_home\_goals = mean(hgoal),  
 goal\_diff\_home = sum(hgoal) - sum(vgoal)  
 )

## `summarise()` has grouped output by 'Season'. You can override using the  
## `.groups` argument.

visitor\_stats <- england %>%  
 group\_by(Season, visitor) %>%  
 summarize(mean\_visitor\_goals = mean(vgoal))

## `summarise()` has grouped output by 'Season'. You can override using the  
## `.groups` argument.

# Объединяем с основным набором данных  
data\_model <- england %>%  
 left\_join(team\_stats, by = c("Season", "home")) %>%  
 left\_join(visitor\_stats, by = c("Season", "visitor"))

### 4. Построение моделей

Построим две модели:

Множественная линейная регрессия. Модель на основе случайного леса (random forest). Линейная регрессия

# Линейная регрессия  
model\_lm <- lm(hgoal ~ mean\_home\_goals + goal\_diff\_home + mean\_visitor\_goals, data = data\_model)  
summary(model\_lm)

##   
## Call:  
## lm(formula = hgoal ~ mean\_home\_goals + goal\_diff\_home + mean\_visitor\_goals,   
## data = data\_model)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -4.4275 -0.9651 -0.1859 0.7649 10.0391   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 0.253886 0.015841 16.027 < 2e-16 \*\*\*  
## mean\_home\_goals 1.082844 0.010535 102.781 < 2e-16 \*\*\*  
## goal\_diff\_home -0.002825 0.000417 -6.774 1.25e-11 \*\*\*  
## mean\_visitor\_goals -0.323104 0.009273 -34.844 < 2e-16 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 1.332 on 203952 degrees of freedom  
## Multiple R-squared: 0.1329, Adjusted R-squared: 0.1329   
## F-statistic: 1.042e+04 on 3 and 203952 DF, p-value: < 2.2e-16

Random Forest

library(randomForest)  
set.seed(42) # Для воспроизводимости  
sample\_data <- data\_model %>% sample\_n(5000)   
cl <- makeCluster(detectCores() - 1) # Используем все ядра, кроме одного  
registerDoParallel(cl)  
# Random Forest  
model\_rf <- randomForest(hgoal ~ mean\_home\_goals + goal\_diff\_home + mean\_visitor\_goals,  
 data = sample\_data,  
 ntree = 100)  
print(model\_rf)

##   
## Call:  
## randomForest(formula = hgoal ~ mean\_home\_goals + goal\_diff\_home + mean\_visitor\_goals, data = sample\_data, ntree = 100)   
## Type of random forest: regression  
## Number of trees: 100  
## No. of variables tried at each split: 1  
##   
## Mean of squared residuals: 2.055446  
## % Var explained: 1.67

### 5. Оценка моделей

Оценим качество моделей по метрикам RMSE и R²:

# Предсказания для линейной модели  
pred\_lm <- predict(model\_lm, data\_model)  
rmse\_lm <- sqrt(mean((data\_model$hgoal - pred\_lm)^2))  
  
# Предсказания для Random Forest  
pred\_rf <- predict(model\_rf, data\_model)  
rmse\_rf <- sqrt(mean((data\_model$hgoal - pred\_rf)^2))  
  
# Вывод результатов  
cat("RMSE Линейная модель:", rmse\_lm, "\n")

## RMSE Линейная модель: 1.331711

cat("RMSE Random Forest:", rmse\_rf, "\n")

## RMSE Random Forest: 1.377579

### 6. Визуализация

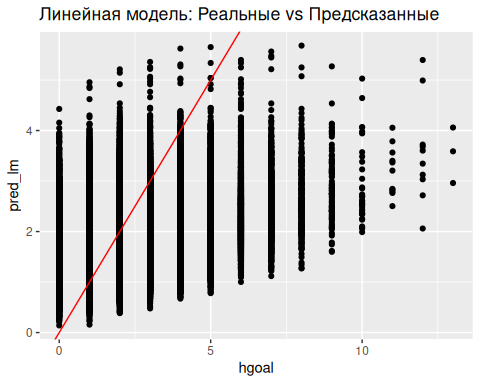
Построим графики для сравнения реальных и предсказанных значений:

library(ggplot2)

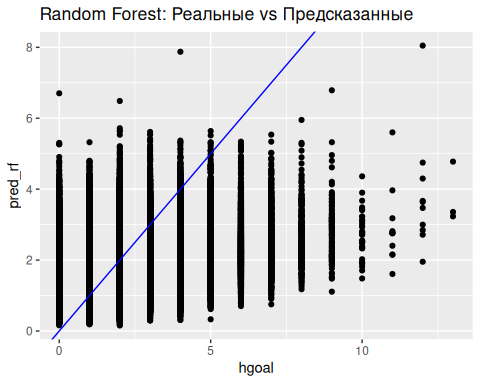
##   
## Присоединяю пакет: 'ggplot2'

## Следующий объект скрыт от 'package:randomForest':  
##   
## margin

# Линейная модель  
ggplot(data\_model, aes(x = hgoal, y = pred\_lm)) +  
 geom\_point() +  
 geom\_abline(slope = 1, intercept = 0, color = "red") +  
 ggtitle("Линейная модель: Реальные vs Предсказанные")



# Random Forest  
ggplot(data\_model, aes(x = hgoal, y = pred\_rf)) +  
 geom\_point() +  
 geom\_abline(slope = 1, intercept = 0, color = "blue") +  
 ggtitle("Random Forest: Реальные vs Предсказанные")

 ### 7. Выводы После выполнения всех шагов можно:

Сравнить качество моделей (например, по RMSE). Сделать вывод о том, какие переменные оказывают наибольшее влияние. Выбрать наиболее подходящую модель для прогнозирования. Готов продолжить работу или доработать код под конкретные ваши цели!