Homework 3

Anish Bhurtyal

2022-11-04

```
library(tidyverse)
library(ggplot2)
```

Data Overview

Sports Analytics Project

In this project, we consider the 1992 baseball salary data set, which is available from http://jse.amstat.org/datasets/baseball.dat.txt This data set (of dimension 337 18) contains salary information (and performance mea- sures) of 337 Major League Baseball players in 1992. Detailed information about this data set can be found at http://jse.amstat.org/datasets/baseball.txt

```
co <- c("salary" ,"batting.avg", "OBP", "runs", "hits", "doubles",
        "triples" ,"homeruns" ,"RBI", "walks" ,"strike.outs",
        "stolen.bases", "errors", "free.agency.elig", "free.agent.91",
        "arb.elig", "arb.91", "name")
baseball <- read.table(file= "http://jse.amstat.org/datasets/baseball.dat.txt",
                       header = F, col.names=co)
names(baseball)
```

```
##  [1] "salary"           "batting.avg"      "OBP"              "runs"
##  [5] "hits"             "doubles"          "triples"          "homeruns"
##  [9] "RBI"              "walks"            "strike.outs"      "stolen.bases"
## [13] "errors"           "free.agency.elig" "free.agent.91"    "arb.elig"
## [17] "arb.91"           "name"
```

1: Exploration Data Analysis (EDA)

(a) (i) Are there any missing data?

```
sum(is.na(baseball) )
```

```
## [1] 0
```

We dont have any missing values in the dataframe

(a) (ii) Among all the predictors, how many of them are continuous, integer counts,and categorical, respectively?

```
glimpse(baseball)
```

```
## Rows: 337
## Columns: 18
## $ salary           <int> 3300, 2600, 2500, 2475, 2313, 2175, 600, 460, 240, 20~
```

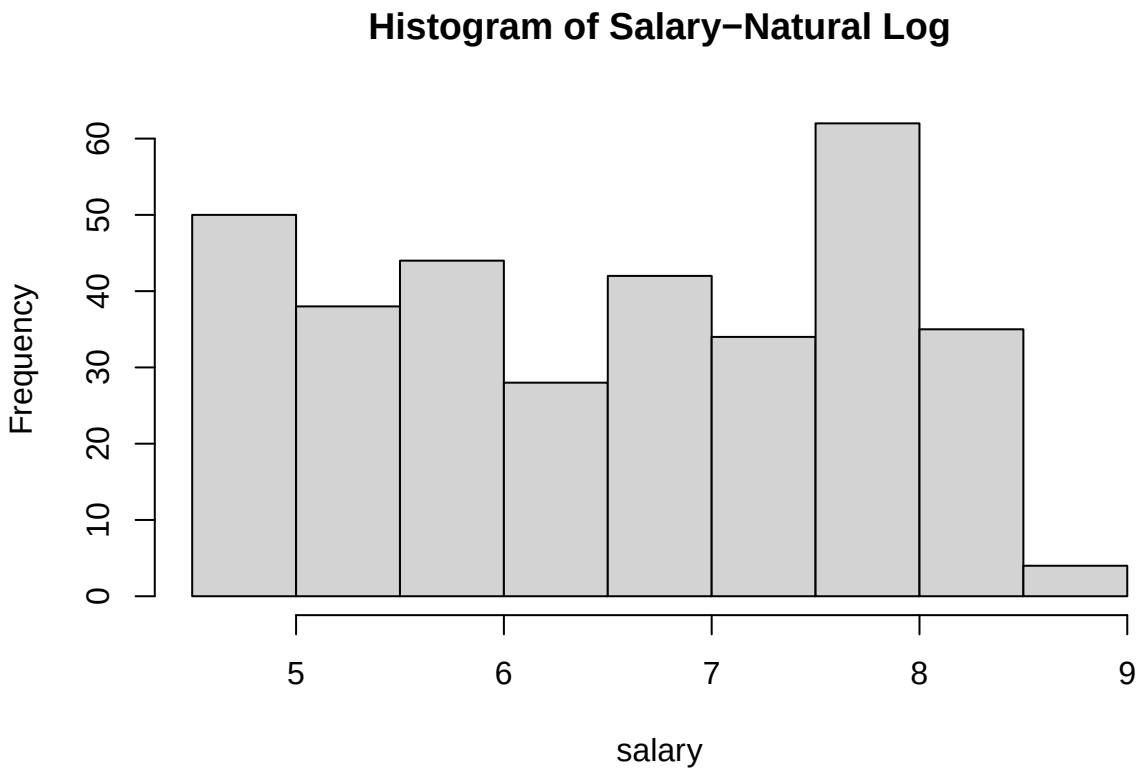
```
## $ batting.avg      <dbl> 0.272, 0.269, 0.249, 0.260, 0.273, 0.291, 0.258, 0.22~
## $ OBP              <dbl> 0.302, 0.335, 0.337, 0.292, 0.346, 0.379, 0.370, 0.27~
## $ runs             <int> 69, 58, 54, 59, 87, 104, 34, 16, 40, 39, 7, 21, 4, 1,~
## $ hits             <int> 153, 111, 115, 128, 169, 170, 86, 38, 61, 64, 38, 45,~
## $ doubles          <int> 21, 17, 15, 22, 28, 32, 14, 7, 11, 10, 5, 9, 2, 0, 22~
## $ triples          <int> 4, 2, 1, 7, 5, 2, 1, 2, 0, 1, 0, 0, 0, 0, 3, 1, 8, 2,~
## $ homeruns         <int> 31, 18, 17, 12, 8, 26, 14, 3, 1, 10, 0, 6, 1, 0, 19, ~
## $ RBI              <int> 104, 66, 73, 50, 58, 100, 38, 21, 18, 33, 10, 22, 3, ~
## $ walks            <int> 22, 39, 63, 23, 70, 87, 15, 11, 24, 14, 5, 19, 2, 4, ~
## $ strike.outs      <int> 80, 69, 116, 64, 53, 89, 45, 32, 26, 96, 18, 56, 1, 3~
## $ stolen.bases     <int> 4, 0, 6, 21, 3, 22, 0, 2, 14, 13, 2, 3, 0, 0, 31, 2, ~
## $ errors           <int> 3, 3, 5, 21, 8, 4, 10, 3, 2, 6, 7, 3, 0, 0, 7, 14, 8,~
## $ free.agency.elig <int> 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1,~
## $ free.agent.91    <int> 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ arb.elig         <int> 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,~
## $ arb.91           <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,~
## $ name             <chr> "Andre Dawson     ", "Steve Buchele    ", "Kal Daniel~
```

- Continuous: salary, batting.avg, OBP
- integer counts: runs, hits,doubles, triples, homeruns, RBI, walks, strike.outs,stolen.bases, error
- categorical: name, arb.91, arb.elig, free.agent.91, free.agency.elig

(b) Obtain the histograms of salary and the logarithm (natural base) of salary and comment on the two visualizations.Proceed with the log-transformed salary from this step on.

```
hist(baseball$salary, xlab="salary", main="Histogram of Salary")
```


```
baseball$salary = log(baseball$salary)
hist(baseball$salary, xlab="salary", main="Histogram of Salary-Natural Log")
```



The majority of players' salaries are below $1,000, as indicated by the right-skewed histogram for player salaries. In contrast, the second histogram for the logarithmic salary has a more uniform distribution.

2. Multiple Linear Regression:

(a) Partition the data randomly into two sets: the training data D0 and the test data D1 with a ratio of about 2:1.

```
set.seed(123)
# Sample Indexes
Index = sample(1:nrow(baseball), size = 0.7*nrow(baseball))
# Splitting Data
TrainData = baseball[Index,]
dim(TrainData)
```

```
## [1] 235  18
```

```
TestData = baseball[-Index,]
dim(TestData)
```

```
## [1] 102  18
```

(b) Using the training data D0, fit a multiple linear regression model called "fit.train".

```
fit.train <- lm(salary ~ batting.avg + OBP + runs + hits +
                  doubles + triples + homeruns + RBI + walks +
                  strike.outs + stolen.bases + errors + free.agency.elig +
                  free.agent.91 + arb.elig + arb.91, data=TrainData)
```

(c) Output the necessary fitting results, e.g., selected variables and their corre-sponding slope parameter estimates.

```
summary(fit.train)# Get the table of parameter estimates
```

```
##
## Call:
## lm(formula = salary ~ batting.avg + OBP + runs + hits + doubles +
##     triples + homeruns + RBI + walks + strike.outs + stolen.bases +
##     errors + free.agency.elig + free.agent.91 + arb.elig + arb.91,
##     data = TrainData)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.37496 -0.29873 -0.00851  0.30219  1.27706
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       5.485e+00  3.008e-01  18.238  < 2e-16 ***
## batting.avg      -7.170e-01  2.469e+00  -0.290  0.77183
## OBP              -1.037e+00  2.196e+00  -0.472  0.63712
## runs              7.418e-03  5.590e-03   1.327  0.18588
## hits              3.582e-05  3.247e-03   0.011  0.99121
## doubles           1.660e-03  8.166e-03   0.203  0.83909
## triples          -9.641e-03  1.980e-02  -0.487  0.62680
## homeruns         -6.148e-04  1.168e-02  -0.053  0.95806
## RBI               1.309e-02  4.623e-03   2.832  0.00506 **
## walks             3.120e-03  4.404e-03   0.708  0.47945
## strike.outs      -5.894e-03  1.959e-03  -3.009  0.00293 **
## stolen.bases      3.581e-03  4.544e-03   0.788  0.43161
## errors           -5.497e-03  7.128e-03  -0.771  0.44145
## free.agency.elig  1.662e+00  1.033e-01  16.087  < 2e-16 ***
## free.agent.91    -2.388e-01  1.248e-01  -1.913  0.05704 .
## arb.elig          1.430e+00  1.132e-01  12.636  < 2e-16 ***
## arb.91           -3.012e-01  2.146e-01  -1.404  0.16176
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5361 on 218 degrees of freedom
## Multiple R-squared:  0.8068, Adjusted R-squared:  0.7926
## F-statistic: 56.91 on 16 and 218 DF,  p-value: < 2.2e-16
```

- *We can see the parameter estimates in the summary table above.*

(d) Is there a relationship between the response and predictors?

- *The quality of a linear regression fit is typically assessed using the R squared statistic (R^2). The above R2 value of 0.8068 obtained using the training dataset explains a good proportion of the variability in the response*
- *The p-value is less than 0.05, so the result is significant to say we have some relationship and F-statistic value is also high*
- *To Reject the NULL hypothesis, we would need the p-value < 0.05. The p-values of variables RBI, strike.outs, free.agent.91, free.agency.elig, and arb.elig are <0.05 but others donot which is why we can't reject the Null Hypothesis*

(e) Using the model (i.e., "fit. train") you developed above, predict using the test data D1. Output the mean squared error (MSE).

```
data <- data.frame(actual= TestData$salary, predicted=predict(fit.train, TestData))
head(data)
```

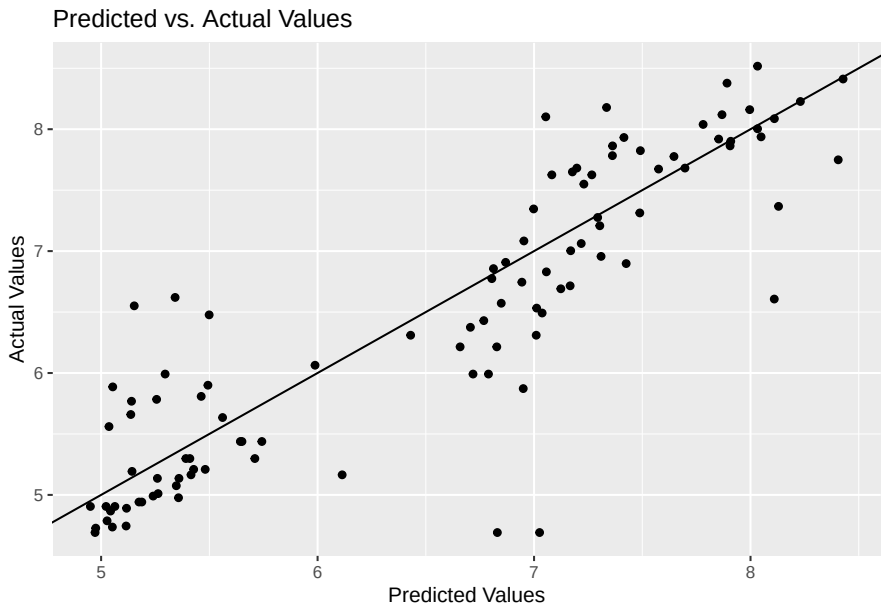
```
##       actual predicted
## 2  7.863267  7.362725
## 3  7.824046  7.491215
## 12 4.941642  5.187980
## 15 7.863267  7.905792
## 18 6.897705  7.425591
## 19 6.829794  7.056972
```

```
#calculate MSE

#library("Metrics")
#mse(data$actual, predict(fit.train , TestData))

mean((data$actual - data$predicted)^2)
```

```
## [1] 0.3205818
```

The mean squared error (MSE) is 0.3205818. The lower the value of MSE the better and 0 means the model is perfect.

```
ggplot(data, aes(x=predicted, y= actual)) +
  geom_point() +
  geom_abline(intercept=0, slope=1) +
  labs(x='Predicted Values', y='Actual Values', title='Predicted vs. Actual Values')
```



The predicted values from the model on TestData are displayed on the x-axis, while the actual values from the TestData dataset are displayed on the y-axis.