Analysis of Prostate Cancer data

Marina Vallejo Vallés (marina.vallejo01@estudiant.upf.edu (mailto:marina.vallejo01@estudiant.upf.edu))

Last update: 10 abril, 2022

Data from Stamey et al. 1989.

During this report we will **train a model that predicts log of prostate-specific antigen (lpsa).**

Brief introduction.

In the publication, they examined the correlation between the level of prostate-specific antigen (PSA) and a number of clinical measures in men who were about to receive a radical prostatectomy. PSA is a protein that is produced by the prostate gland. The higher a man's PSA level, the more likely it is that he has prostate cancer.

The variables are log cancer volume (lcavol), log prostate weight (lweight), age, log of the amount of benign prostatic hyperplasia (lbph), seminal vesicle invasion (svi), log of capsular penetration (lcp), Gleason score (gleason), and percent of Gleason scores 4 or 5 (pgg45).

Prostate data info:

- Predictors (columns 1–8)

1. lcavol
2. lweight
3. age
4. lbph
5. svi
6. lcp
7. gleason
8. pgg45

- Outcome (column 9):

9. lpsa

EDA

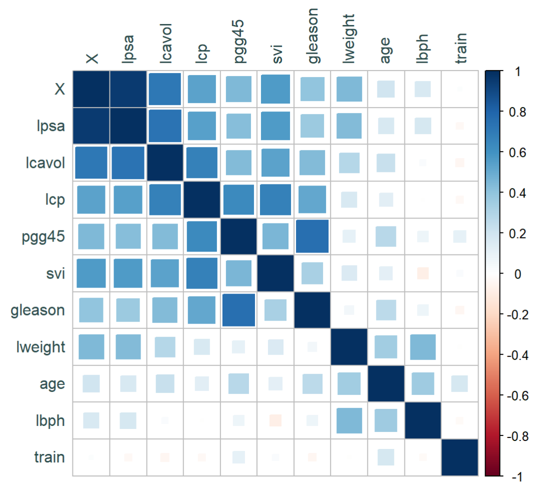
Work with the data set.

```
# Load data:
prostate_dataset <- read.csv("prostate_cancer_data.txt", sep="\t")

# Summary of the data:
summary(prostate_dataset)
```

```
##        X          lcavol           lweight           age
##  Min.   : 1   Min.   :-1.3471   Min.   :2.375   Min.   :41.00
##  1st Qu.:25   1st Qu.: 0.5128   1st Qu.:3.376   1st Qu.:60.00
##  Median :49   Median : 1.4469   Median :3.623   Median :65.00
##  Mean   :49   Mean   : 1.3500   Mean   :3.629   Mean   :63.87
##  3rd Qu.:73   3rd Qu.: 2.1270   3rd Qu.:3.876   3rd Qu.:68.00
##  Max.   :97   Max.   : 3.8210   Max.   :4.780   Max.   :79.00
##       lbph              svi              lcp             gleason
##  Min.   :-1.3863   Min.   :0.0000   Min.   :-1.3863   Min.   :6.000
##  1st Qu.:-1.3863   1st Qu.:0.0000   1st Qu.:-1.3863   1st Qu.:6.000
##  Median : 0.3001   Median :0.0000   Median :-0.7985   Median :7.000
##  Mean   : 0.1004   Mean   :0.2165   Mean   :-0.1794   Mean   :6.753
##  3rd Qu.: 1.5581   3rd Qu.:0.0000   3rd Qu.: 1.1787   3rd Qu.:7.000
##  Max.   : 2.3263   Max.   :1.0000   Max.   : 2.9042   Max.   :9.000
##      pgg45             lpsa           train
##  Min.   :  0.00   Min.   :-0.4308   Mode :logical
##  1st Qu.:  0.00   1st Qu.: 1.7317   FALSE:30
##  Median : 15.00   Median : 2.5915   TRUE :67
##  Mean   : 24.38   Mean   : 2.4784
##  3rd Qu.: 40.00   3rd Qu.: 3.0564
##  Max.   :100.00   Max.   : 5.5829
```

Pearson correlation:

```
prostate_correlations <- cor(prostate_dataset,method="pearson")

# Generate correlation plot:
corrplot(prostate_correlations, hclust.method = "ward",method = "square",
          order = "FPC", type = "full", tl.col = "darkslategray")
```



In order to select the variables for the model to predict **lpsa**, we check the Pearson correlation values:

```
prostate_correlations
```

```
##                   X      lcavol      lweight       age         lbph         svi
## X       1.00000000  0.71113628  0.440071938 0.1965557  0.167928486  0.56678035
## lcavol  0.71113628  1.00000000  0.280521380 0.2249999  0.027349703  0.53884500
## lweight 0.44007194  0.28052138  1.000000000 0.3479691  0.442264399  0.15538490
## age     0.19655569  0.22499988  0.347969112 1.0000000  0.350185896  0.11765804
## lbph    0.16792849  0.02734970  0.442264399 0.3501859  1.000000000 -0.08584324
## svi     0.56678035  0.53884500  0.155384903 0.1176580 -0.085843238  1.00000000
## lcp     0.53369604  0.67531048  0.164537142 0.1276678 -0.006999431  0.67311118
## gleason 0.39360794  0.43241706  0.056882093 0.2688916  0.077820447  0.32041222
## pgg45   0.44972672  0.43365225  0.107353785 0.2761124  0.078460018  0.45764762
## lpsa    0.95811486  0.73446033  0.433319382 0.1695928  0.179809404  0.56621822
## train   0.01115249 -0.04654347 -0.009940658 0.1776155 -0.029939957  0.02679950
##                  lcp     gleason      pgg45        lpsa        train
## X        0.533696039  0.39360794 0.44972672  0.95811486  0.011152493
## lcavol   0.675310484  0.43241706 0.43365225  0.73446033 -0.046543468
## lweight  0.164537142  0.05688209 0.10735379  0.43331938 -0.009940658
## age      0.127667752  0.26889160 0.27611245  0.16959284  0.177615517
## lbph    -0.006999431  0.07782045 0.07846002  0.17980940 -0.029939957
## svi      0.673111185  0.32041222 0.45764762  0.56621822  0.026799505
## lcp      1.000000000  0.51483006 0.63152825  0.54881317 -0.037427296
## gleason  0.514830063  1.00000000 0.75190451  0.36898681 -0.044171456
## pgg45    0.631528246  0.75190451 1.00000000  0.42231586  0.100516371
## lpsa     0.548813175  0.36898681 0.42231586  1.00000000 -0.033889743
## train   -0.037427296 -0.04417146 0.10051637 -0.03388974  1.000000000
```

For our linear regression model, as the highest correlation value is for **lpsa-lcavol : 0.73** (ignore X as it is an identifier), a first approach could be to have **lpsa** as response and **lcavol** as the predictor. It will be a simple linear regression.

MODEL BUILDING

First of all we have to split the data. 3/4 data will be retained for modeling and variable **lpsa** will be used to stratify the samples:

```
# Generate the split object:
prostate_split <- data_split <- initial_split(prostate_dataset, prop = 3/4, strata = lpsa)

# Build the training prostate dataset (with 3/4 of the data)
prostate_training <- prostate_split %>% training()

# Build the testing prostate dataset:
prostate_test <- prostate_split %>% testing()
```

Simple linear regression

- Response : lpsa
- Predictor: lcavol

Now generate the model (linear regression):

```
simple_lm <- linear_reg() %>% set_mode("regression") %>% set_engine("lm")
# set_mode is redundant, as it can only be regression, but we keep it in order to be verbose

# Check:
simple_lm
```

```
## Linear Regression Model Specification (regression)
##
## Computational engine: lm
```

Fit model with our training data previously generated, having **lcavol** as predictor and **lpsa** as response:

```
fit_simple_lm <- simple_lm %>% fit(lpsa ~ lcavol, data = prostate_training)

# Check summary
fit_simple_lm %>% pluck("fit") %>%summary()
```

```
##
## Call:
## stats::lm(formula = lpsa ~ lcavol, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65662 -0.41263  0.05257  0.45321  1.90565
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.52263    0.14553  10.462 7.00e-16 ***
## lcavol       0.71090    0.08016   8.869 5.17e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7817 on 69 degrees of freedom
## Multiple R-squared:  0.5327, Adjusted R-squared:  0.5259
## F-statistic: 78.66 on 1 and 69 DF,  p-value: 5.165e-13
```

Check the parameter estimates of the fit object:

```
tidy(fit_simple_lm)
```

```
## # A tibble: 2 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)    1.52     0.146      10.5  7.00e-16
## 2 lcavol         0.711    0.0802      8.87 5.17e-13
```

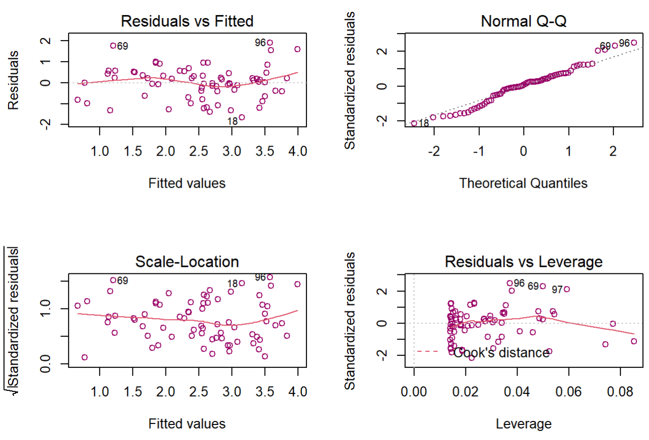
Extract the model statistics:

```
glance(fit_simple_lm)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.533         0.526 0.782      78.7 5.17e-13     1  -82.2  170.  177.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Obtain plots:

```
par(mfrow=c(2,2))

plot(fit_simple_lm$fit, pch = 21, col = '#990066')
```



The **Residuals vs. Fitted** plot is used to detect non-linearity of data, as well as unequal error variances, and outliers. In our plot the linearity seems to hold reasonably well. The red line is close to the dashed grey line. Points 69 and 96 could be considered as outliers, as they have large residual values. And finally we can confirm heteroskedasticity, when we move to the right (x-axis), the spread seems to increase.

The **Quantile-Quantile (QQ)** plot is a visual way to check if a variable is normal. It compares the quantiles of our data against the quantiles of another distribution, the desired one (in our case the normal distribution). As we can see our data mostly fall in the dashed line, so we can assume it has a normal distribution.

Prediction, with test dataset (1/4 of the original data).

```
results_prostate_test <- predict(fit_simple_lm, new_data = prostate_test) %>% bind_cols(prostate_test)
```

Root Mean Square Error (rmse):

```
# .pred is our predictor, previously generated by the predict function
rmse(results_prostate_test, truth = lpsa, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.803
```

The Root Mean Square Error (rmse) allows us to measure the error of a model when predicting quantitative data. A low value of rmse indicates that the model is able to properly fit a data set. For our **linear regression** model it is equal to 0.8, we will check later if it is improved with the **multiple regression** model.

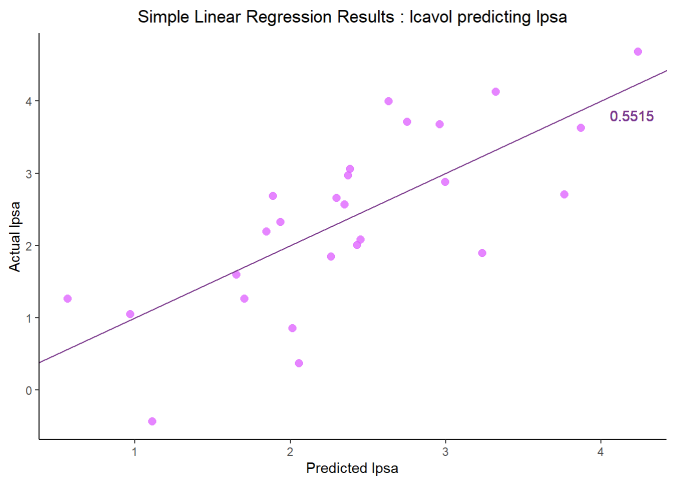
Square Error (rsq):

```
rsq_model <- rsq(results_prostate_test, truth = lpsa, estimate = .pred)
rsq_model
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.551
```

Plots:

```
ggplot(data = results_prostate_test,
       mapping = aes(x = .pred, y = lpsa)) +
       geom_point( size = 2.5,colour="mediumorchid1",pch=19,alpha = 0.8)+
       geom_abline(intercept = 0, slope = 1, color = 'mediumorchid4') +
       labs(title = 'Simple Linear Regression Results : lcavol predicting lpsa',
       x = 'Predicted lpsa',y = 'Actual lpsa')+
       theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), legend.position = "none",
       panel.background = element_blank(), axis.line = element_line(colour = "black"),
       plot.title = element_text(hjust = 0.5))+
       geom_text(x=4.2, y=3.8, label=format(round(rsq_model$.estimate, 4)), color = 'mediumorchid4')
```



Here we have the visual representation of the results of our simple linear regression model, with an R-squared equal to 0.55

Multiple linear regression

As the previously generated model is not optimal, we are going to make a second approach. In order to improve the model here we will include extra predictor variables and perform a multiple linear regression.

The predictor variables included in this new model will be the ones that had a correlation value greater than 0.5 in the first part of the report.

- Response : lpsa
- Predictors: lcavol, svi, lcp

```
# We can reuse simple_lm created before
fit_multiple_lm <- simple_lm %>% fit(lpsa ~ lcavol + svi + lcp, data = prostate_training)
# Check summary
fit_multiple_lm %>% pluck("fit") %>%summary()
```

```
##
## Call:
## stats::lm(formula = lpsa ~ lcavol + svi + lcp, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.46583 -0.50834  0.04841  0.47108  1.68900
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.51436    0.19425   7.796 5.62e-11 ***
## lcavol       0.63621    0.11409   5.576 4.76e-07 ***
## svi          0.52852    0.32187   1.642    0.105
## lcp         -0.03121    0.10903  -0.286    0.776
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7749 on 67 degrees of freedom
## Multiple R-squared:  0.5542, Adjusted R-squared:  0.5342
## F-statistic: 27.76 on 3 and 67 DF,  p-value: 8.777e-12
```

Check the parameter estimates of the fit object:

```
tidy(fit_multiple_lm)
```

```
## # A tibble: 4 x 5
##   term        estimate std.error statistic  p.value
##   <chr>          <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)   1.51       0.194     7.80  5.62e-11
## 2 lcavol        0.636      0.114     5.58  4.76e- 7
## 3 svi           0.529      0.322     1.64  1.05e- 1
## 4 lcp          -0.0312     0.109    -0.286 7.76e- 1
```

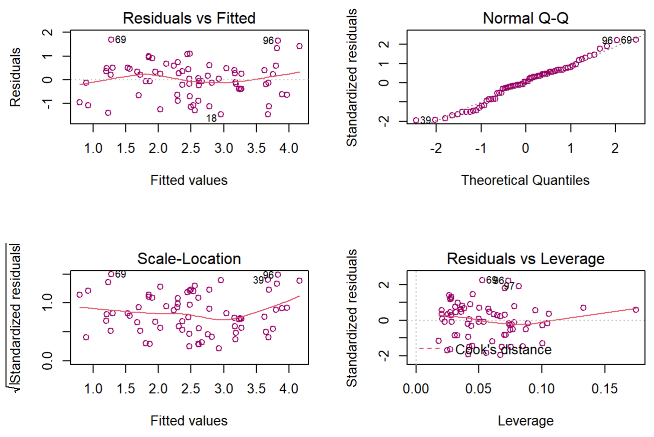
Extract the model statistics:

```
glance(fit_multiple_lm)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.554         0.534 0.775      27.8 8.78e-12     3  -80.6  171.  182.
## # ... with 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

Obtain plots:

```
par(mfrow=c(2,2))

plot(fit_multiple_lm$fit, pch = 21, col = '#990066')
```



In our **Residuals vs. Fitted** plot the linearity again seems to hold well, as the red line is close to the dashed grey line. Points 69 and 96 again can be considered as outliers, including this time point 18, as they have large residual values. In this case I wouldn't say that we have heteroskedasticity, the spread seem to be equal on both sides.

In the **Quantile-Quantile (QQ)** data again, mostly fall in the dashed line, so we can assume it has a normal distribution.

Prediction, with test data set (1/4 of the original data).

```
results_prostate_test2 <- predict(fit_multiple_lm, new_data = prostate_test) %>% bind_cols(prostate_test)
```

Root Mean Square Error (rmse):

```
rmse(results_prostate_test2, truth = lpsa, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.727
```

```
# .pred is our predictor, previously generated by the predict function
```

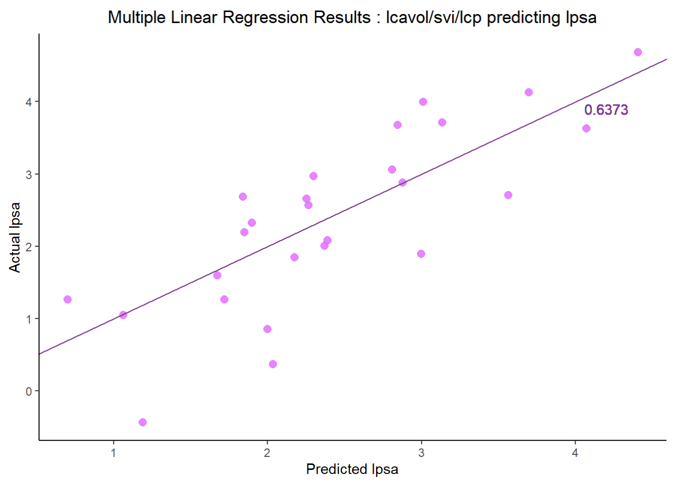
Here we have improved the rmse compared to the **linear regression** rmse, we have a lower value.

Square Error (rsq):

```
rsq_model2 <- rsq(results_prostate_test2, truth = lpsa, estimate = .pred)
rsq_model2
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.637
```

```
ggplot(data = results_prostate_test2,
       mapping = aes(x = .pred, y = lpsa)) +
       geom_point( size = 2.5,colour="mediumorchid1",pch=19,alpha = 0.8)+
       geom_abline(intercept = 0, slope = 1, color = 'mediumorchid4') +
       labs(title = 'Multiple Linear Regression Results : lcavol/svi/lcp predicting lpsa',
       x = 'Predicted lpsa',y = 'Actual lpsa')+
       theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(), legend.position = "none",
       panel.background = element_blank(), axis.line = element_line(colour = "black"),
       plot.title = element_text(hjust = 0.5))+
       geom_text(x=4.2, y=3.9, label=format(round(rsq_model2$.estimate, 4)), color = 'mediumorchid4')
```



Here we have the visual representation of the results of our multiple linear regression model, with an R-squared equal to 0.63

Model comparison. Discussion.

Overall, we have seen that the performance of the **multiple linear regression** model was better than the **simple linear regression** model.

For the **multiple linear regression** model we had a lower rmse value indicating that the model is able to properly fit a data set.

It also had a greater R-squared, indicating a better adjustment of the predicted data to the linear function.

Further models should be explored in order to improve the prediction of `log of prostate-specific antigen (lpsa)` , maybe using a different subset of predictors and increasing the number of predictor variables. But always considering the risk of overfitting the model and its negative effects.