Linear Regression Analysis

Johnny Lo

March 2020

Table of Contents

Linear Regression Analysis

Linear regression is a linear approach to modelling the relationship between a **continuous** outcome variable and one or more **continuous** predictors/features. Although this is beyond the scope of this unit, linear regression can be applied to model *non-linear* relationships as well with clever manipulation of the predictors. The term "linear" only applies to the parameter coefficients, and not the predictors in the model.

Applications of linear regression typically fall into two categories:

(1) Prediction - model is used to predict the outcome based on some unobserved or untested values of the predictors.

(2) Association - model estimates is used to explain the strength and nature of the relationship between the outcome and the predictors.

In machine learning, application (1) is the main focus, while (2) is secondary or of little interest.

To start, you need to install and load the relevant packages for this workshop.

```
#De-comment to install the packages below
#install.packages(c("tidyverse","GGally","caret","corrplot","car"))
library(tidyverse)  #For ggplot2
library(GGally)  #For scatter plot matrix
library(caret)  #Classification and Regression Training package
library(corrplot)  #For visualisng correlation matrix
library(car)  #For the VIF function
```

The *CPU.csv* data file contains performance data relating to computer hardaware from various vendors and models. The variables in the files are as follows:

(1) Vendor name: 27 (adviser, amdahl, apollo, basf, bti, burroughs, c.r.d, cambex, cdc, dec, dg, formation, four-phase, gould, harris, honeywell, hp, ibm, ipl, magnuson, microdata, nas, ncr, nixdorf, perkin-elmer, prime, siemens)
(2) Model name: many unique symbols
(3) MYCT: machine cycle time in nanoseconds (integer)
(4) MMIN: minimum main memory in kilobytes (integer)
(5) MMAX: maximum main memory in kilobytes (integer)
(6) CACH: cache memory in kilobytes (integer)
(7) CHMIN: minimum channels in units (integer)
(8) CHMAX: maximum channels in units (integer)
(9) PRP: published relative CPU performance (integer)

The goal here is to predict relative CPU performance based on its cycle time, memory and etc. The vendors' names and the hardware model, i.e. Variables (1) and (2) are inconsequential in this exercise.

Let us firstly import the data into R studio.

```
CPU <- read.csv("CPU.csv", header=TRUE, stringsAsFactors=FALSE);  #Read in
the data
str(CPU)  #Examine its structure

## 'data.frame':    164 obs. of  9 variables:
##  $ Vendor: chr  "adviser" "amdahl" "amdahl" "amdahl" ...
##  $ Model : chr  "32/60" "470v/7a" "580-5840" "580-5850" ...
##  $ MYCT  : int  125 29 23 23 23 23 400 60 50 350 ...
##  $ MMIN  : int  256 8000 16000 16000 16000 32000 1000 2000 4000 64 ...
##  $ MMAX  : int  6000 32000 32000 32000 64000 64000 3000 8000 16000 64 ...
##  $ CACH  : int  256 32 64 64 64 128 0 65 65 0 ...
##  $ CHMIN : int  16 8 16 16 16 32 1 1 1 1 ...
##  $ CHMAX : int  128 32 32 32 32 64 2 8 8 4 ...
##  $ PRP   : int  198 220 367 489 636 1144 38 92 138 10 ...
```

Given that **PRP** is the outcome of interest and that it is a numeric variable, linear regression modelling is therefore appropriate here. Not only do we want to build a linear regression model here, we also want to be evaluate the predictions from said model.

Hence, we will now split the dataset into *training* and *test* sets. The training set is used to build the model, and the test set is used to evaluate its predictive performance. Here, we will use a 75/25 split, i.e. randomly select 75% of the original data to be in the training set, and the remaining 25% form the test set.

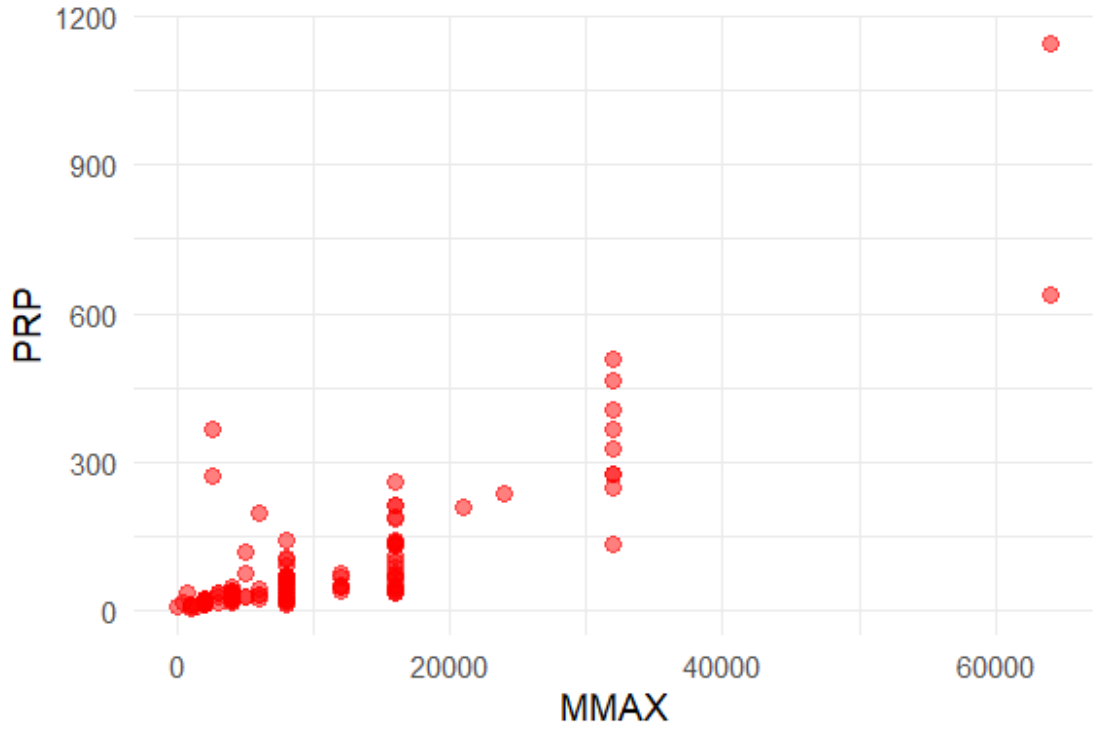
```
#Create the training and test datasets

set.seed(1)  #Set the random seed.

# Step 1: Get row numbers for the training data
trainRowNumbers <- createDataPartition(CPU$PRP, #The outcome variable
                                       p=0.75, #proportion of data to form
the training set
                                       list=FALSE  #Don't store the result in
a list
                                       );

# Step 2: Create the training  dataset
trainData <- CPU[trainRowNumbers,]

# Step 3: Create the test dataset
testData <- CPU[-trainRowNumbers,]
```

Simple Linear Regression

Simple linear regression refers to the situation where the outcome variable is regressed against **one** predictor or feature.

Suppose, at this moment, we are only interested in predicting **PRP** from maximum main memory, i.e. **MMAX**. Let us examine their relationship via a scatter plot and Person's correlation coefficient.

```
#Extract PRP and MMAX from the main training data
trainData.slr <- trainData[,c(5,9)]

#Plot PRP against MMAX
ggplot(trainData.slr,aes(x=MMAX,y=PRP)) +
  geom_point(size=3,colour="red",alpha=0.5) +
  theme_minimal(base_size=14)
```




```
#Pearson's correlation coefficient
cor(trainData.slr)

##           MMAX       PRP
## MMAX 1.0000000 0.8427903
## PRP  0.8427903 1.0000000
```

Exercise: What sort of relationship can you see here? Is it reasonable to model this relationship with a linear model?

Let us now build a simple linear regression model for predicting PRP with MMAX using the *lm(.)* function.

```
mod.slr <- lm(formula=PRP~MMAX,  #PRP as a function of MMAX
              data=trainData.slr)  #The relevant training data

summary(mod.slr)  #Summary of the model

##
## Call:
## lm(formula = PRP ~ MMAX, data = trainData.slr)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -208.61  -35.03    1.65   20.49  435.96
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.283e+01  9.887e+00  -2.309   0.0226 *
## MMAX         1.142e-02  6.603e-04  17.295   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 78.43 on 122 degrees of freedom
## Multiple R-squared:  0.7103, Adjusted R-squared:  0.7079
## F-statistic: 299.1 on 1 and 122 DF,  p-value: < 2.2e-16

#Show the estimated parameter coefficients to 3 dp
mod.slr$coefficients %>% round(digits=3)

## (Intercept)        MMAX
##     -22.828       0.011
```

Exercise: From the above summary of the simple linear regression model, determine:

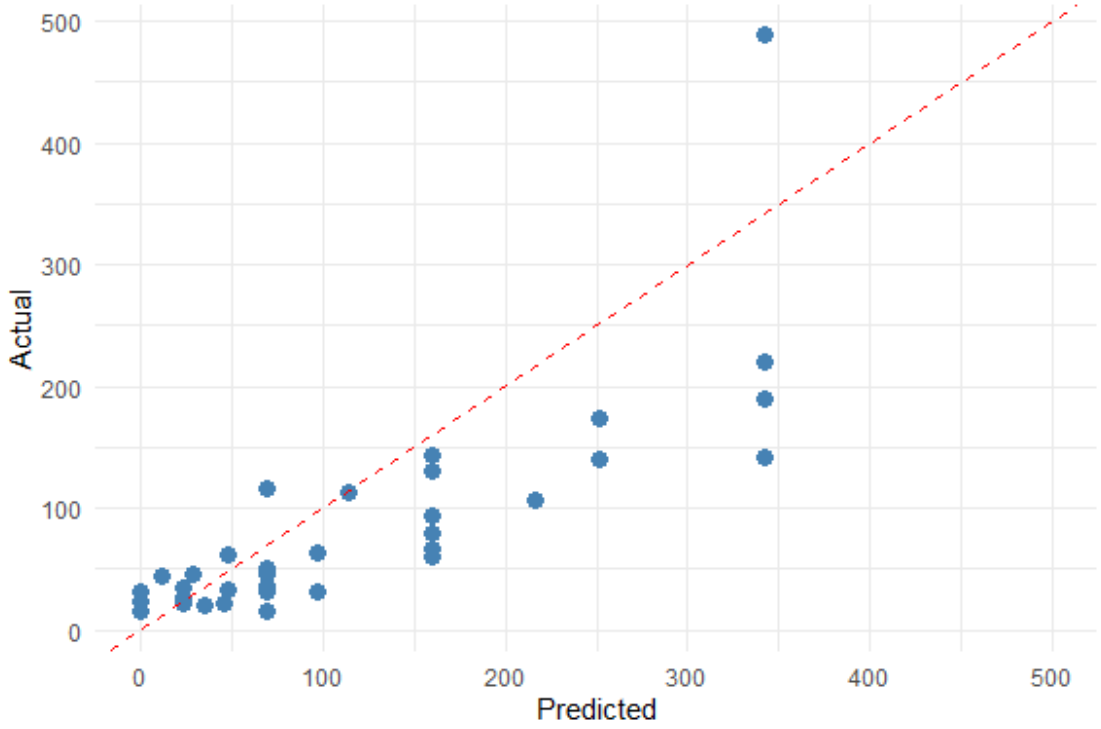
- The equation of the simple linear regression model
- The significance, i.e. p-value, of MMAX as a predictor of PRP, and interpret this value
- The coefficient of determination R^2, and interpret this value

We will now evaluate the predictive performance of this model on the test data and determine the prediction root mean squared error (RMSE) andbias, and the correlation and predicted R^2 between the actual and predicted PRP values for the test set. Prediction RMSE and bias that are close to zero, and a correlation value that is close to one are desired.

```
#Predict PRP with the test data
pred <- predict(mod.slr,  #model used to make prediction
                newdata=testData  #new data, i.e. test data, to predict on
                )

#Difference between the actual and estimated PRP in the test set
diff <- testData$PRP - pred

RMSE.slr <- diff^2 %>% mean %>% sqrt;  #RMSE
bias.slr <- diff %>% mean  #Bias
cor.slr <- cor(testData$PRP,pred)  #Correlation
predR2.slr <-cor.slr^2  #Prediction R2

#Show out the results for SLR
SLR.pf <- c(RMSE=RMSE.slr,
            Bias=bias.slr,
            Corr=cor.slr,
            PredR2=predR2.slr); SLR.pf

##        RMSE        Bias        Corr      PredR2
##  68.1102851 -26.3419105   0.8075859   0.6521950
```

Next, let us plot the actual PRP values from the test set against their predicted values.

```
df <- data.frame(Actual=testData$PRP,Predicted=pred)

ggplot(df,aes(x=Predicted,y=Actual))+
  geom_point(size=3,colour="steelblue")+
  xlim(0,500)+
  #Reference line given by y=x, i.e. slope=1 and intercept=0
  geom_abline(slope=1,
              intercept=0,
              colour="red",  #Colour of the line
              linetype=2) + #Dotted line
  theme_minimal()
```


Actual
Predicted
0
100
200
300
400
500

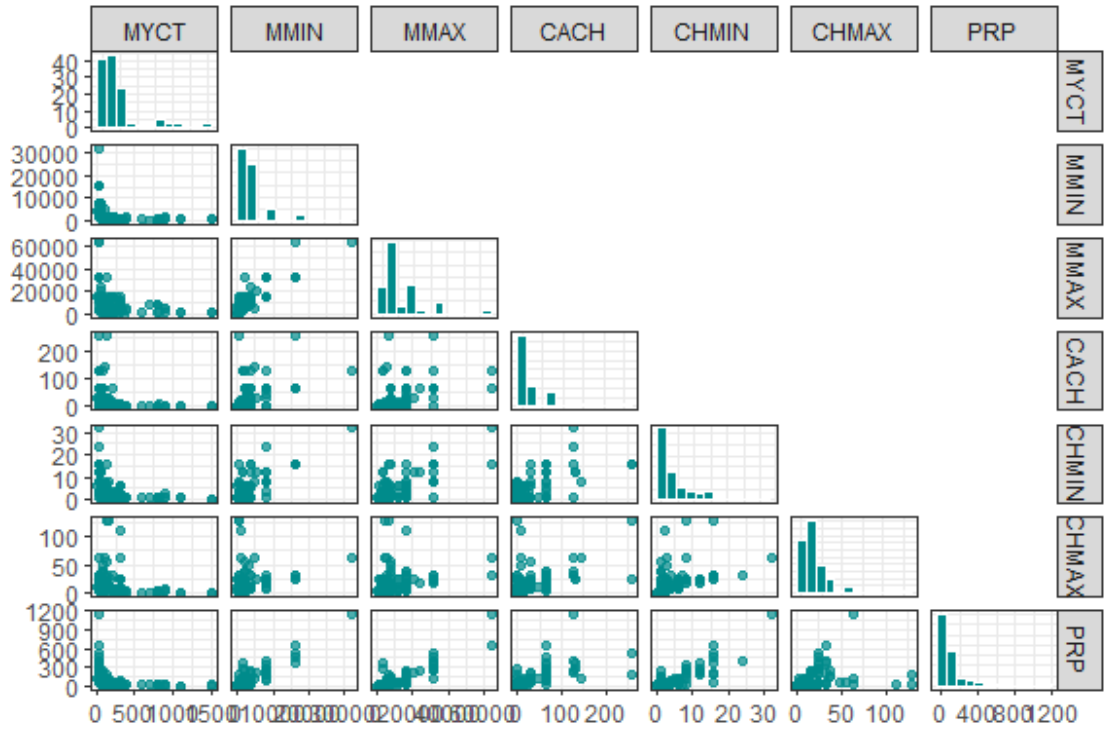
Exercise: What can you conclude from the above prediction performance measures and the scatter plot of Actual vs Predicted PRP?

Multiple Linear Regression

In MLR, the continuous outcome is linearly modelled as a function of **two or more** continuous predictors or features.

We will revisit the CPU data, but this time we will consider all six predictors, i.e. Columns 3 to 8. First, we will generate a scatter plot matrix between all the predictors and the outcome **PRP**.

```
ggpairs(trainData[,3:9],
        upper=NULL,  #Don't display the upper triangle
        ggplot2::aes(fill="cyan4"),
        lower=list(continuous = wrap("points",colour="cyan4",alpha=0.7)),
        diag=list(continuous=wrap("barDiag",colour="white",
                                  fill="cyan4",bins=12))) +
  theme_bw()
```

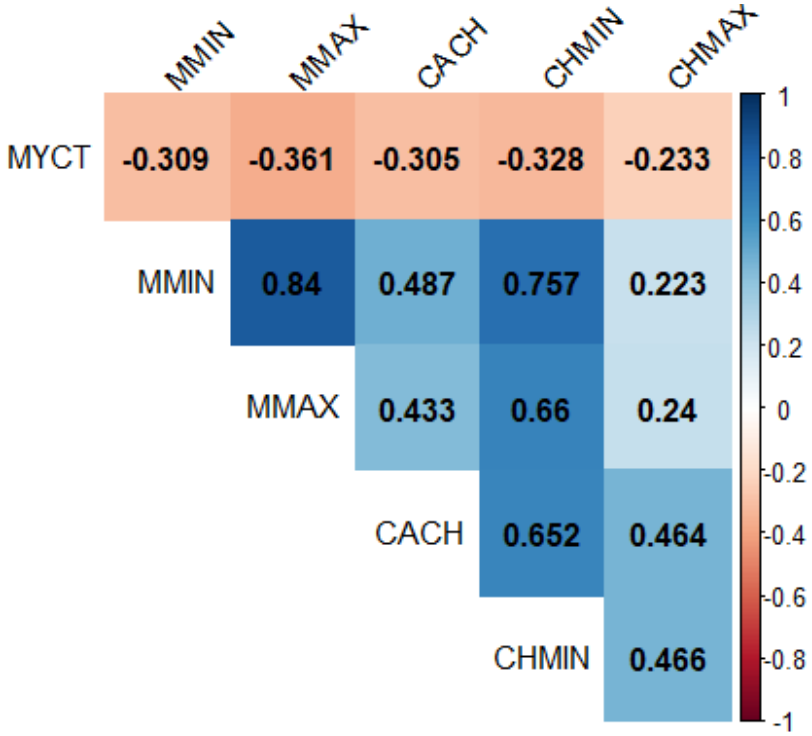


The last row of the scatter plot matrix shows the bivariate relationship between PRP with each of the 6 features. It is clear from the 1st scatter plot along this row that PRP is not linearly correlated with MYCT and that transforming the data may be useful here. In fact, the two are actually log-linearly related, but let's not worry about this. If we leave MYCT as is in the MLR model, then it is likely that the final result will shown that MYCT is not a

significant predictor of PRP. However, it may not be wise to conclude which variables are NOT important based solely on these plots. In particular for variables with uninteresting pattern that sometimes they can help explain certain aspects of the outcome variable that the visually important variables may not.

With the scatter plot matrix, it is also important to observe the relationship between the features, and in particular, those that are highly correlated with each other as this can imply collinearity in our data. We can dig deeper by examining the correlation matrix between the features.

```
cor(trainData[,3:8]) %>%
  corrplot(method="color", #escribe the magnitude of r by colours
           type="upper",  #Show the upper triangle only
           addCoef.col = "black",  #Add the correlation coefficient to the
matrix
           tl.col="black",  #Colour of the text label
           tl.srt=45,  #Angle of the text label
           number.digits=3,  #Number of decimal places for r
           diag=FALSE)  #Do not show the main diagnal values
```



Exercise: Do you believe collinearity could be an issue later for our MLR model? Explain.

Let us go ahead and build the multiple regression model and regress PRP against the 6 features.

```r
#Method 1 - Specify the names of all features
mod.mlr <- lm(PRP~MYCT+MMIN+MMAX+CACH+CHMIN+CHMAX,
              data=trainData[,3:9]);

#Method 2
mod.mlr <- lm(PRP~.,  #Use all available features in the dataset
              data=trainData[,3:9]);

summary(mod.mlr)  #Summary of the MLR model
```

```
##
## Call:
## lm(formula = PRP ~ ., data = trainData[, 3:9])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -123.027  -14.837   -1.283   17.422  211.118
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.749e+01  8.317e+00  -3.305 0.001259 **
## MYCT         2.212e-02  1.555e-02   1.423 0.157356
## MMIN         1.494e-02  2.129e-03   7.016 1.59e-10 ***
## MMAX         3.704e-03  7.191e-04   5.151 1.06e-06 ***
## CACH         6.642e-01  1.246e-01   5.333 4.78e-07 ***
## CHMIN        5.864e+00  1.458e+00   4.022 0.000103 ***
## CHMAX       -1.183e-01  2.278e-01  -0.519 0.604618
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 45.38 on 117 degrees of freedom
## Multiple R-squared:  0.907,  Adjusted R-squared:  0.9022
## F-statistic: 190.2 on 6 and 117 DF,  p-value: < 2.2e-16
```

```r
#Show the estimated parameter coefficients to 3 dp
mod.mlr$coefficients %>% round(digits=3)
```

```
## (Intercept)        MYCT        MMIN        MMAX        CACH       CHMIN
##     -27.490       0.022       0.015       0.004       0.664       5.864
##       CHMAX
##      -0.118
```

Exercise: From the above summary of the multiple linear regression model, determine:

- The equation of the simple linear regression model
- The feature(s) that are **NOT** significant predictors of PRP
- The multiple coefficient of determination R^2, and interpret this value

The variance inflation factor (VIF) can be generated using the *vif(.)* function from the **car** package.

```
vif(mod.mlr)
```

```
##     MYCT     MMIN     MMAX     CACH    CHMIN    CHMAX
## 1.200610 4.719909 3.543292 1.867197 3.478805 1.444968
```

Exercise: Is there any evidence of collinearity based on the VIF measure?

We will now evaluate the predictive performance of the MLR model.

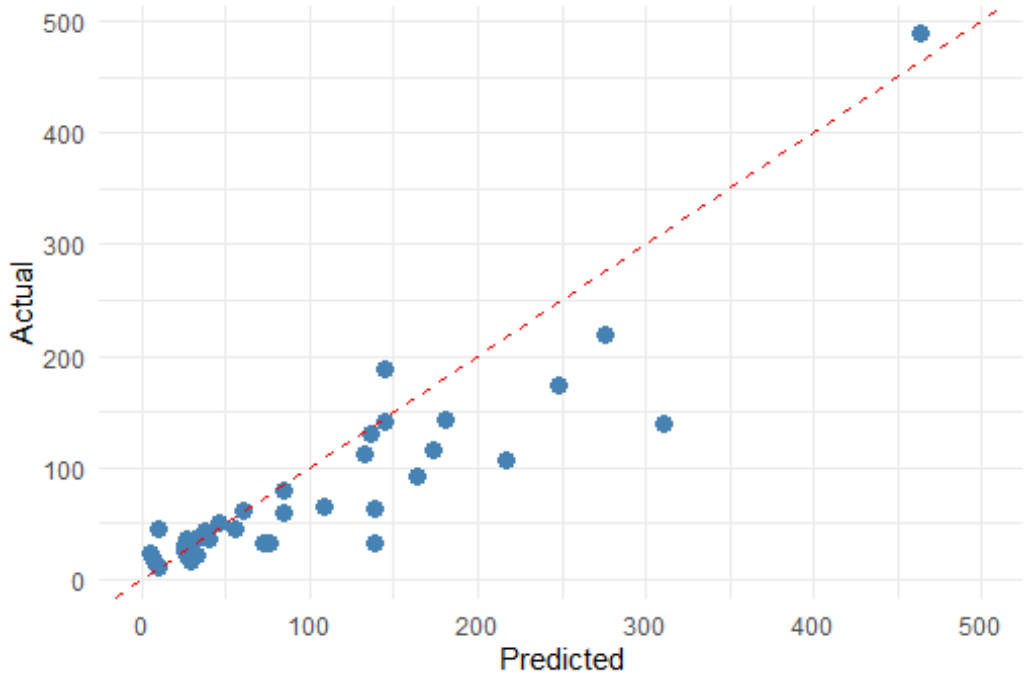
```
#Predict PRP with the test data
pred <- predict(mod.mlr,  #model used to make prediction
                newdata=testData  #new data, i.e. test data, to predict on
                )

#Difference between the actual and estimated PRP in the test set
diff <- testData$PRP - pred

RMSE.mlr <- diff^2 %>% mean %>% sqrt;  #RMSE
bias.mlr <- diff %>% mean  #Bias
cor.mlr <- cor(testData$PRP,pred)  #Correlation
predR2.mlr <-cor.mlr^2  #Predicted R2

#Predictive measures for MLR
MLR.pf <- c(RMSE=RMSE.mlr, Bias=bias.mlr,
            Corr=cor.mlr, PredR2=predR2.mlr); MLR.pf
```

```
##        RMSE        Bias        Corr      PredR2
##  47.2227780 -20.1941936   0.9025831   0.8146562
```

The scatter plot between the actual PRP values from the test set against the MLR predicted values is as follows.

```
df <- data.frame(Actual=testData$PRP,Predicted=pred)

ggplot(df,aes(x=Predicted,y=Actual))+
  geom_point(size=3,colour="steelblue")+
  xlim(0,500)+
  #Reference line given by y=x, i.e. slope=1 and intercept=0
  geom_abline(slope=1,
              intercept=0,
              colour="red",  #Colour of the line
```

```
              linetype=2) + #Dotted line
  theme_minimal()
```



Next, we compare the predictive performance of SLR to MRL.

```
#Display the results for SLR and MLR
data.frame(SLR=SLR.pf,MLR=MLR.pf)

##                SLR         MLR
## RMSE    68.1102851  47.2227780
## Bias   -26.3419105 -20.1941936
## Corr     0.8075859   0.9025831
## PredR2   0.6521950   0.8146562
```

Exercise: What can you conclude regarding the performance of SLR vs MLR?

Notice here for the MLR model that the Multiple $R^2 = 0.902$ is greater than the prediction $R^2 = 0.815$. This is an indication that the MLR model is perhaps overfitted, which is not unexpected based on the significance values of the features in the model summary.

There are a number of ways to overcome or alleviate this problem, including performing the log-likelihood test, stepwise regression or other variable selection methods. Here we will perform a recursive feature elimination (RFE). The RFE, a backward variable selection process, is as follows:

Step 1: Build a model to all features in the training set and rank each feature based on its importance to the model. In our case, it is the *p*-value.

Step 2: Keeping priority to the most important variables, iterate through by building models of given subset sizes, that is, subsets of most important predictors determined from step 1. Ranking of the features is recalculated in each iteration.

Step 3: The model performances are compared across different subset sizes to arrive at the optimal number and list of final predictors.

We can implement RFE using the *rfe(.)* function from the **caret** package.

```
set.seed(1)
options(warn=-1)  #Turn the warning messages off

subsets <- c(2:6)  #Subsets of features to test

#Set the cross validation parameters
ctrl <- rfeControl(functions = lmFuncs,
                   method = "repeatedcv",  #10-fold CV by default
                   repeats = 10,  #repeat CV 10 times
                   #Prevents copious amounts of output from being produced
                   verbose = FALSE)
#Perform RFE
lmProfile <- rfe(x=trainData[,3:8], #features only
                 y=trainData$PRP,  #outcome variable
                 sizes = subsets,
                 rfeControl = ctrl)
lmProfile

##
## Recursive feature selection
##
## Outer resampling method: Cross-Validated (10 fold, repeated 10 times)
##
## Resampling performance over subset size:
##
##  Variables  RMSE Rsquared   MAE RMSESD RsquaredSD MAESD Selected
##          2 79.95   0.6977 49.52  42.52     0.2223 18.35
##          3 80.39   0.6953 50.41  42.43     0.2229 18.50
##          4 72.32   0.7136 46.54  31.50     0.2203 14.33
##          5 54.02   0.8176 34.52  21.63     0.1573 11.00
##          6 49.98   0.8540 32.09  25.91     0.1428 11.48        *
##
## The top 5 variables (out of 6):
##    CHMIN, CACH, CHMAX, MYCT, MMIN
```

In this instance, the RFE algorithm suggests that all 6 features should be retained from a prediction accuracy standpoint, even though, statistically speaking, both CHMAX and MYCT are non-sigificant predictors.