GRIP Task 1: Prediction using Supervised ML

Name: Debartha Paul

July 4, 2021

Importing libraries and visualising the data

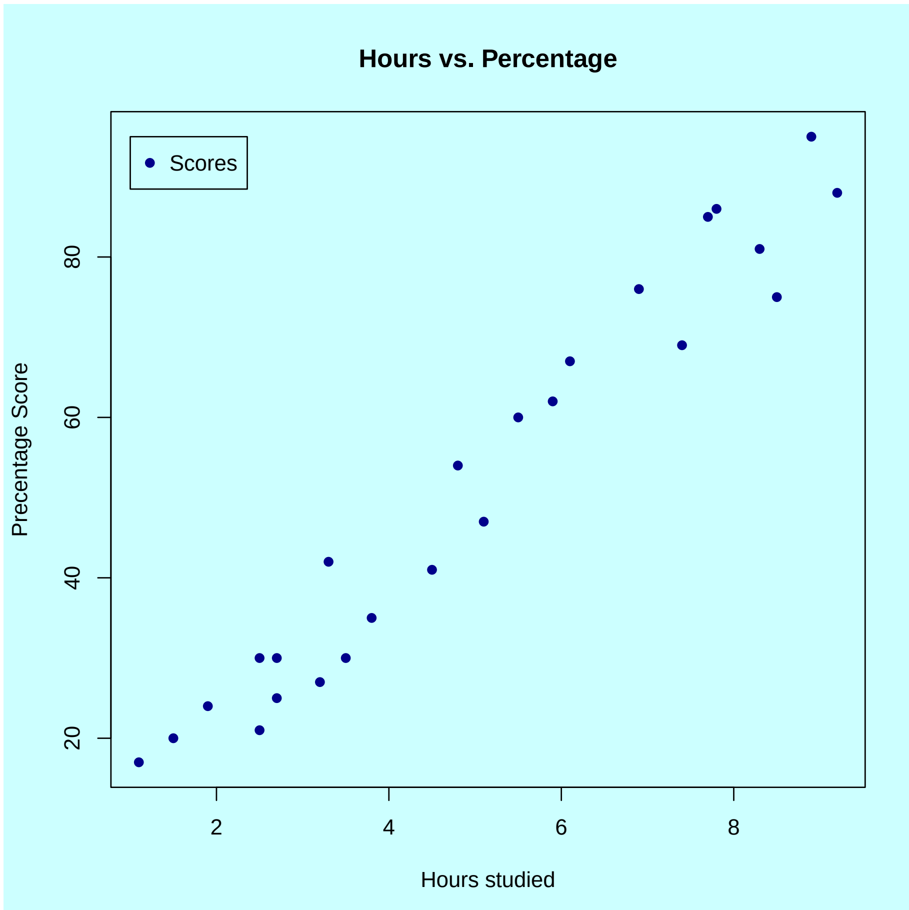
We first load the libraries required for our work and then read the dataset

```
#Importing the necessary libraries
library(Metrics)
```

```
#Reading the dataset
s_data<-read.csv('http://bit.ly/w-data',header=T)
dim(s_data)#the dimensions of the dataset

## [1] 25  2

head(s_data,n=5)#a brief preview of the dataset

##   Hours Scores
## 1   2.5     21
## 2   5.1     47
## 3   3.2     27
## 4   8.5     75
## 5   3.5     30

names(s_data)#column names of the dataset

## [1] "Hours"  "Scores"
```

Now we plot the data points on a 2-D graph to check if there's any visible correlation between the variables

```
#Plotting the distribution of the scores
par(bg='#CCFFFF')
plot(s_data$Hours,s_data$Scores,pch=16,col='dark blue',
        xlab='Hours studied',ylab='Precentage Score',
        main='Hours vs. Percentage')
legend(x=1,y=95,'Scores',pch=16,col='dark blue')
```



From the graph above, we can clearly see that there is a positive linear relation between the number of hours studied and percentage of score.

Preparing the data

We now divide the data into attributes (inputs or `x`) and labels (outputs or `y`)

```
X<-s_data$Hours;Y<-s_data$Scores
```

Then, we split our data into training and test sets. We do this using the `sample()` method in `R`.

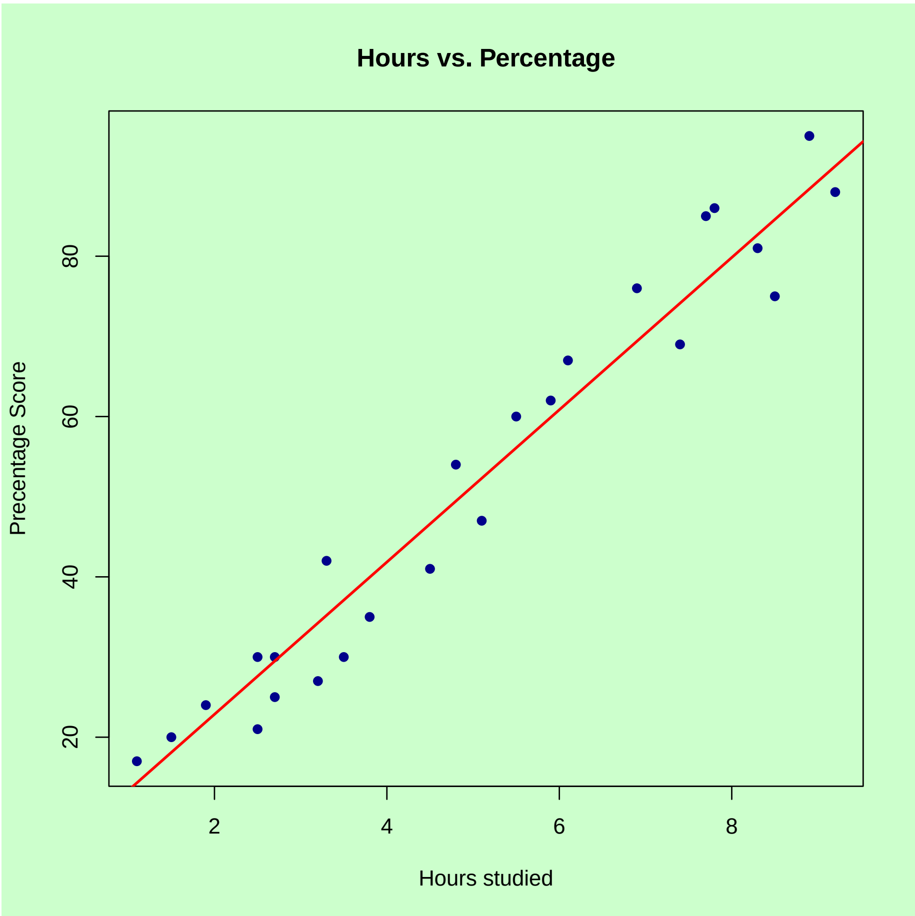
```
t_sample<-sample(nrow(s_data),floor(0.8*nrow(s_data)),replace=F)
X_train<-X[t_sample];Y_train<-Y[t_sample]
X_test<-X[-t_sample];Y_test<-Y[-t_sample]
```

Training the Algorithm

We have split our data into training and testing sets. Now we train our algorithm and then plot the regression line along with the observed marks.

```
model<-lm(Y_train~X_train)
```

```
#Plotting the regression line and the test data
par(bg='#CCFFCC')
plot(s_data$Hours,s_data$Scores,pch=16,col='dark blue',
        xlab='Hours studied',ylab='Precentage Score',
        main='Hours vs. Percentage')
abline(model,col='red',lwd=2)
```



Making Predictions

Now, it's time to make some predictions using our trained agorithm.

```
X_test#Testing data

## [1] 3.2 8.3 7.7 8.9 7.4

Y_predicted<-predict(model,newdata=data.frame(X_train=X_test))#Predicting the scores
```

```
#Comparing Actual vs. Predicted values
df<-data.frame('Hours'=X_test,'Actual score'=Y_test,'Predicted score'=Y_predicted)
df

##   Hours Actual.score Predicted.score
## 1   3.2           27        34.24819
## 2   8.3           81        82.68557
## 3   7.7           85        76.98705
## 4   8.9           95        88.38408
## 5   7.4           69        74.13780
```

Predicting the score of a student who studied for 9.25 hours:

```
pred<-predict(model,newdata=data.frame(X_train=9.25))
pr_data<-data.frame('Hours'=9.25,'Predicted Score'=pred)
pr_data

##   Hours Predicted.Score
## 1  9.25        91.70822
```

Evaluating the model

Finally, we evaluate the performance of the model. We chose the mean absolute error as the measure of performance of the algorithm (lower is better):

```
mae(df$Actual,df$Predicted)

## [1] 5.740084
```