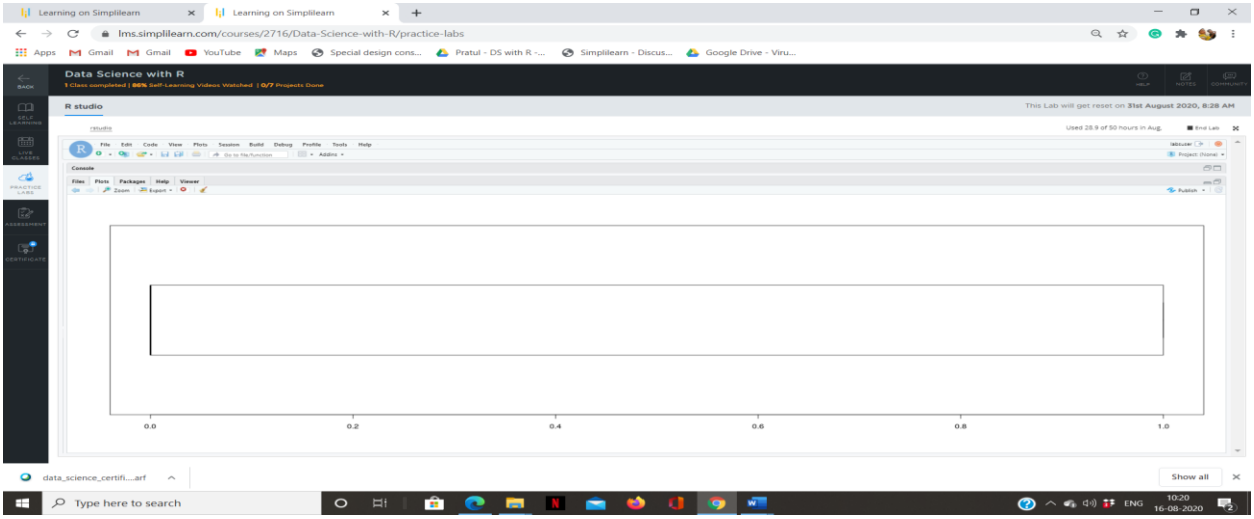
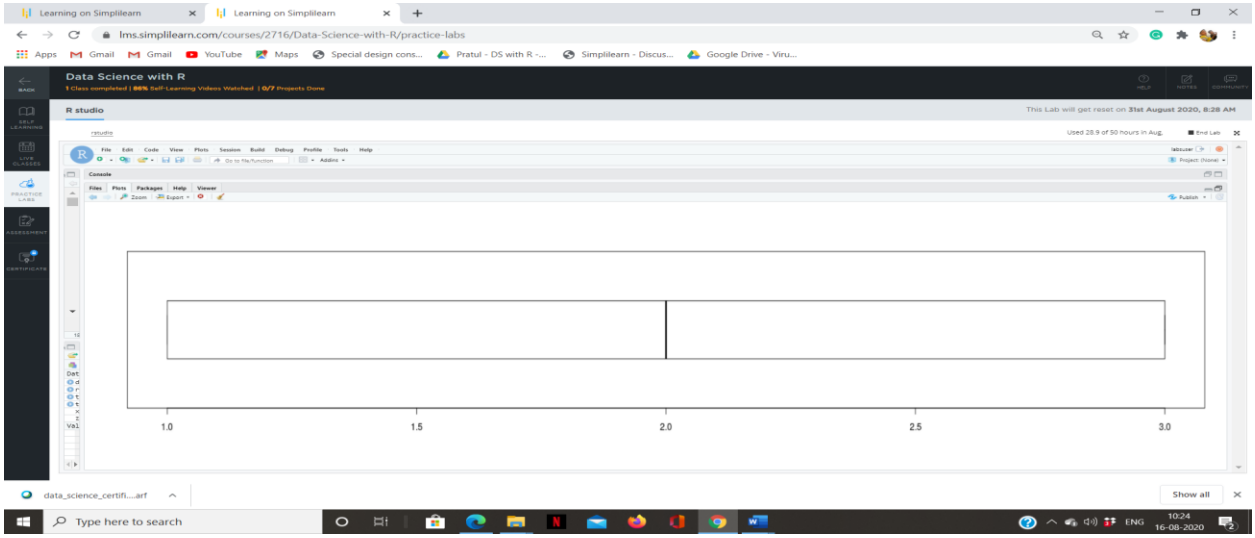
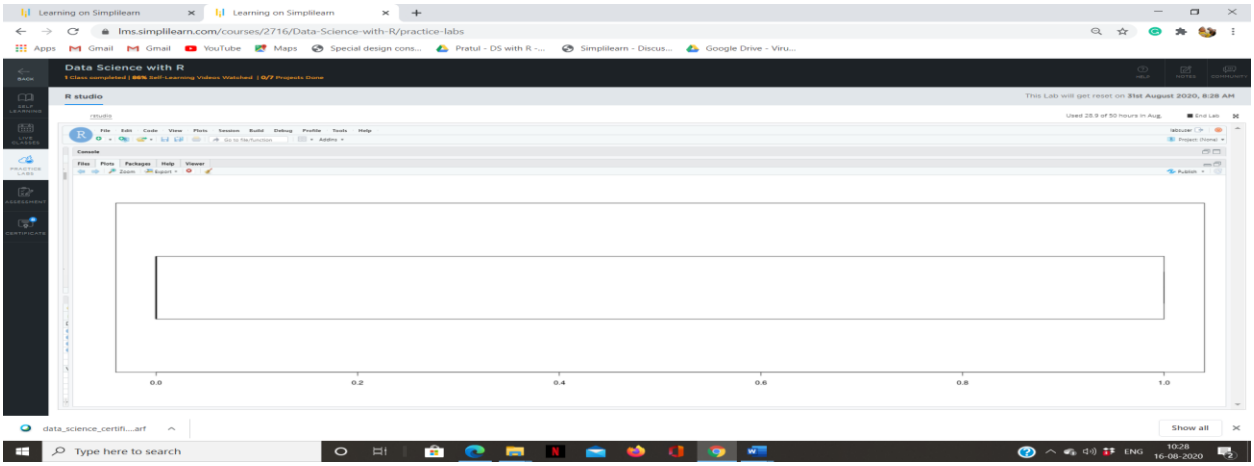
```
> #Analysis Tasks: Analyze the historical data and determine the key drivers for
admission.
> getwd()
[1] "/home/labsuser"
> dataset <- read.csv("College_admission.csv")
> View(dataset)
> #A->predictive tasks
> #task1 -to find missing value in data set.?if found perform missing value treat
ment.
> x=is.na(dataset)
> dataset[x]
numeric(0)
> #task2-Find outliers (if any, then perform outlier treatment)
> #although admit,gpa,ses,Gender_Male,Race,rank are cetagorical data still as per
 said lets check it out for ouliers.
> boxplot(dataset$admit,horizontal = T)
```



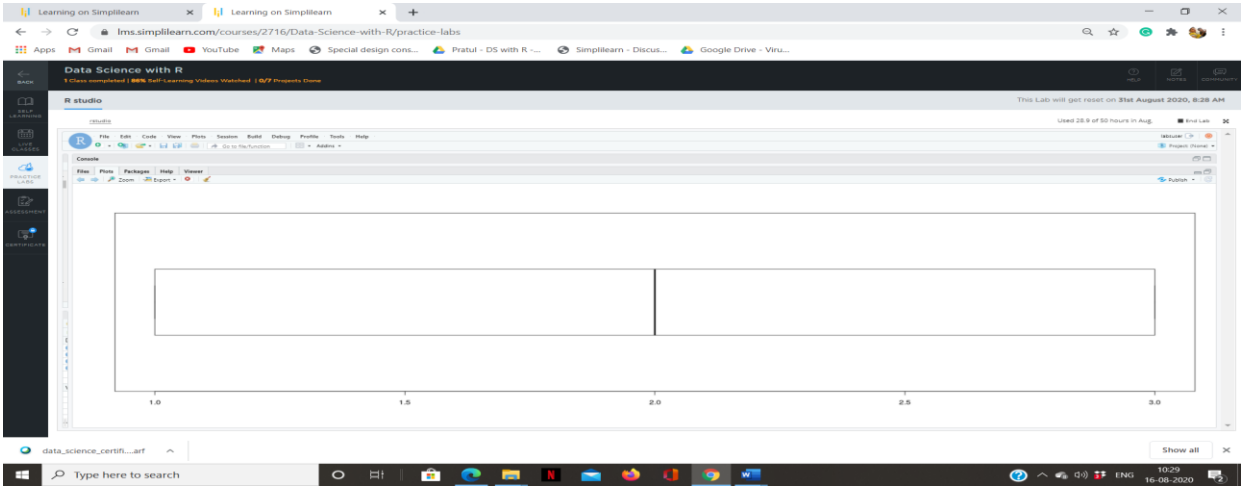

```
> boxplot(dataset$ses,horizontal = T)
```



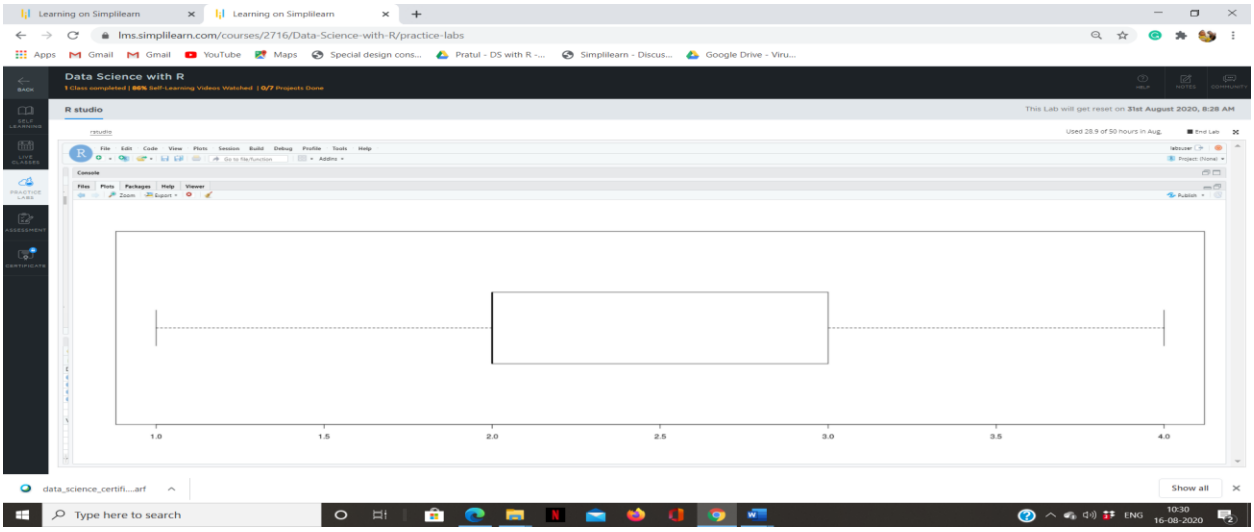
```
> boxplot(dataset$Gender_Male,horizontal = T)
```



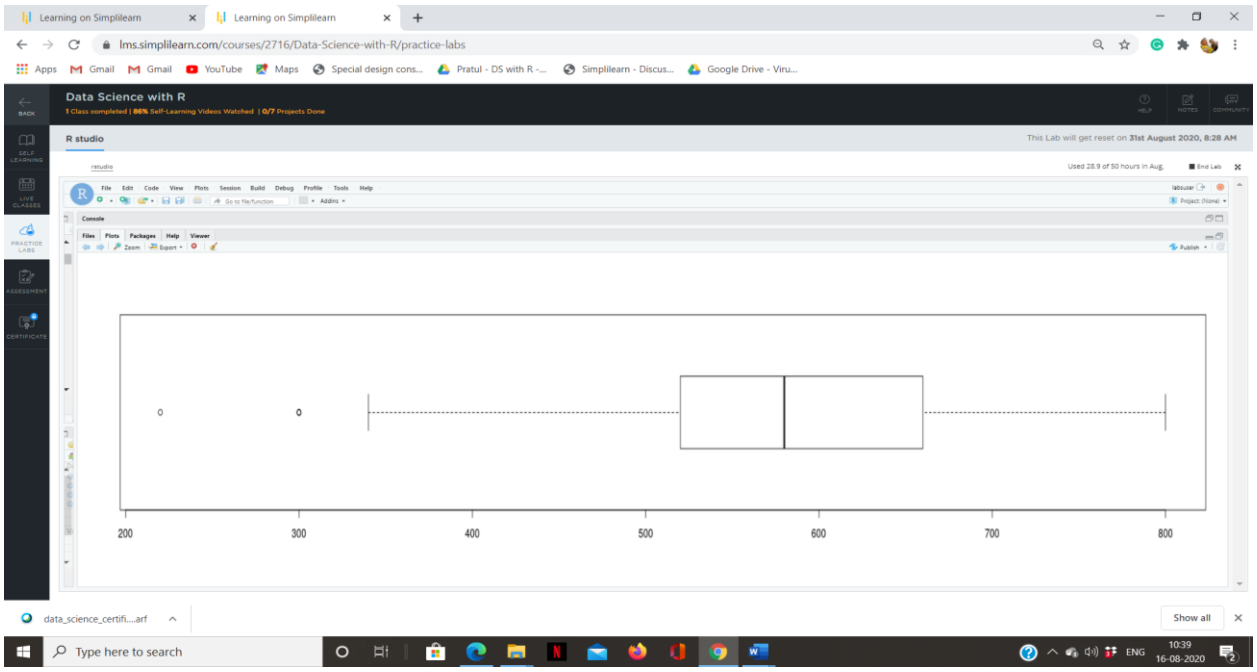

```
> boxplot(dataset$Race,horizontal = T)
```



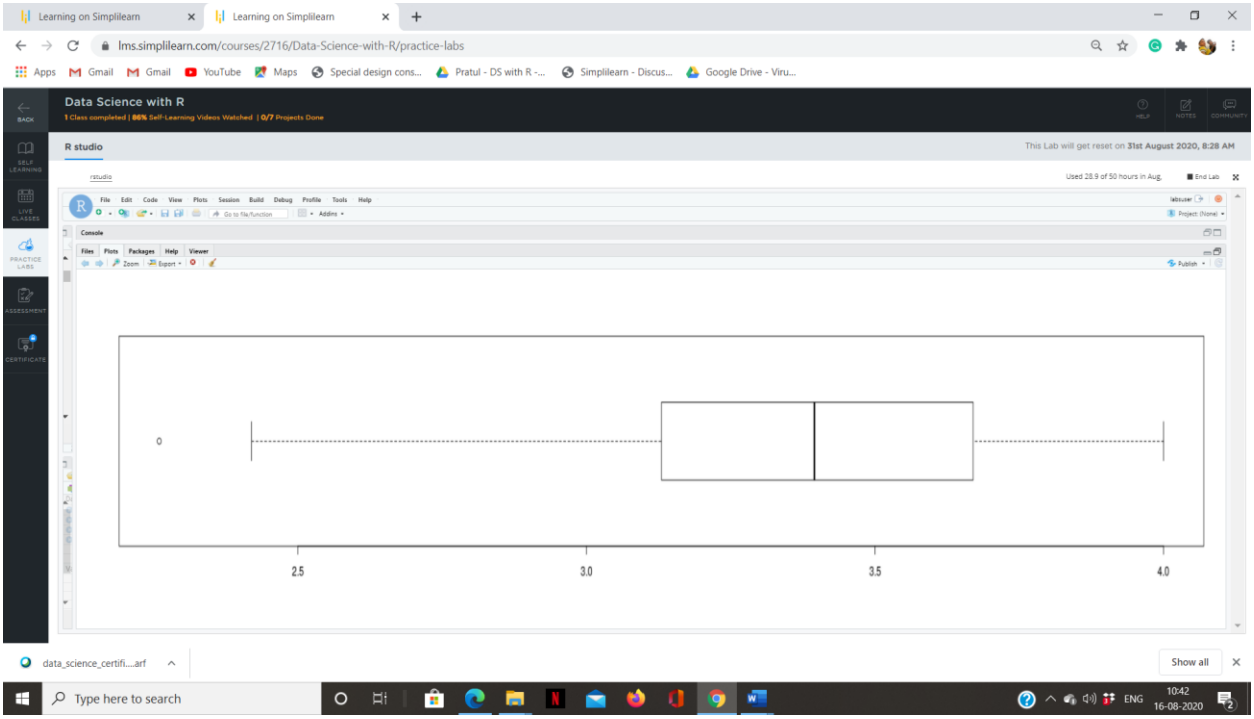

```
> boxplot(dataset$rank,horizontal = T)
```



```
OutVals = boxplot(dataset$gre,horizontal = T)$out
```

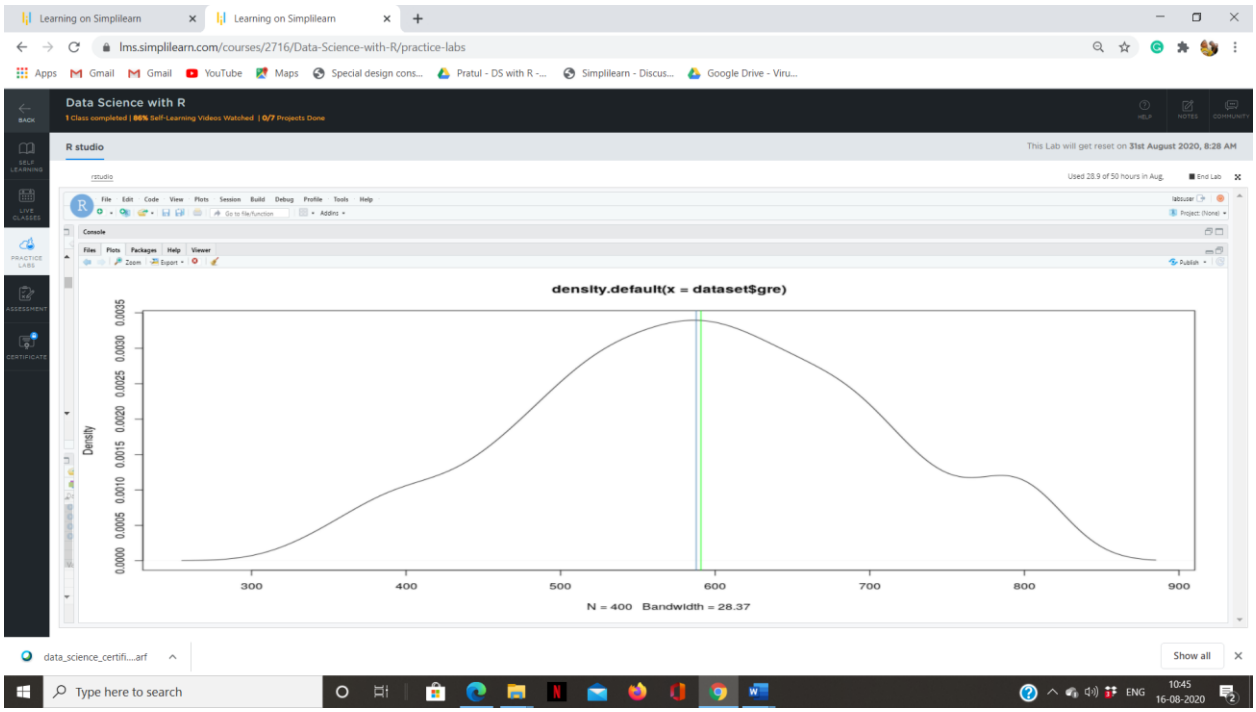



```
> OutVals
[1] 300 300 220 300
> #there are 4 outliers in dataset$gre or gre column
> outvals1=boxplot(dataset$gpa,horizontal = T)$out
```

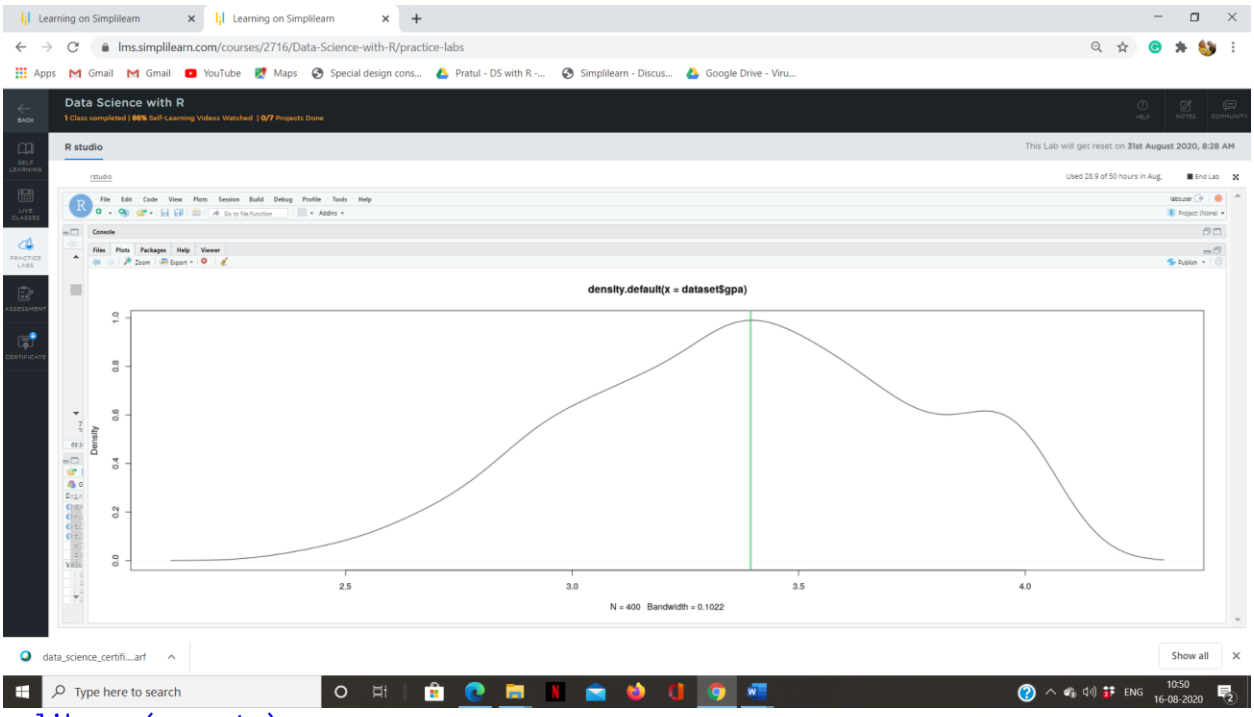



```
> outvals1
[1] 2.26
```

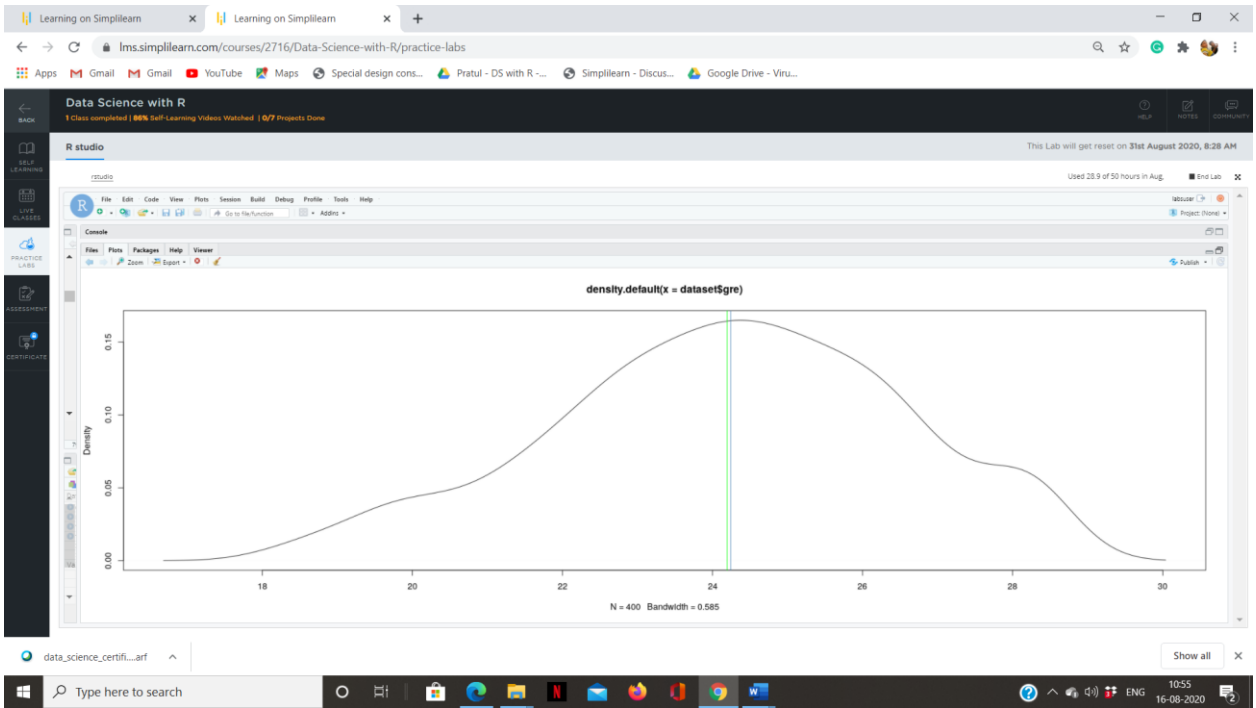
```
> #task4-Find whether the data is normally distributed or not. Use the plot to de
termine the same.
> mean(dataset$gre)
[1] 590.777
> median(dataset$gre)
[1] 587.7
> #here mean is not equal to median in case of gre column of dataset hance data i
s not normaly distributed
> #by visualization
> plot(density(dataset$gre))
> abline(v = c(mean(dataset$gre),
+              median(dataset$gre)),
+        col = c('green', 'steelblue'))
```



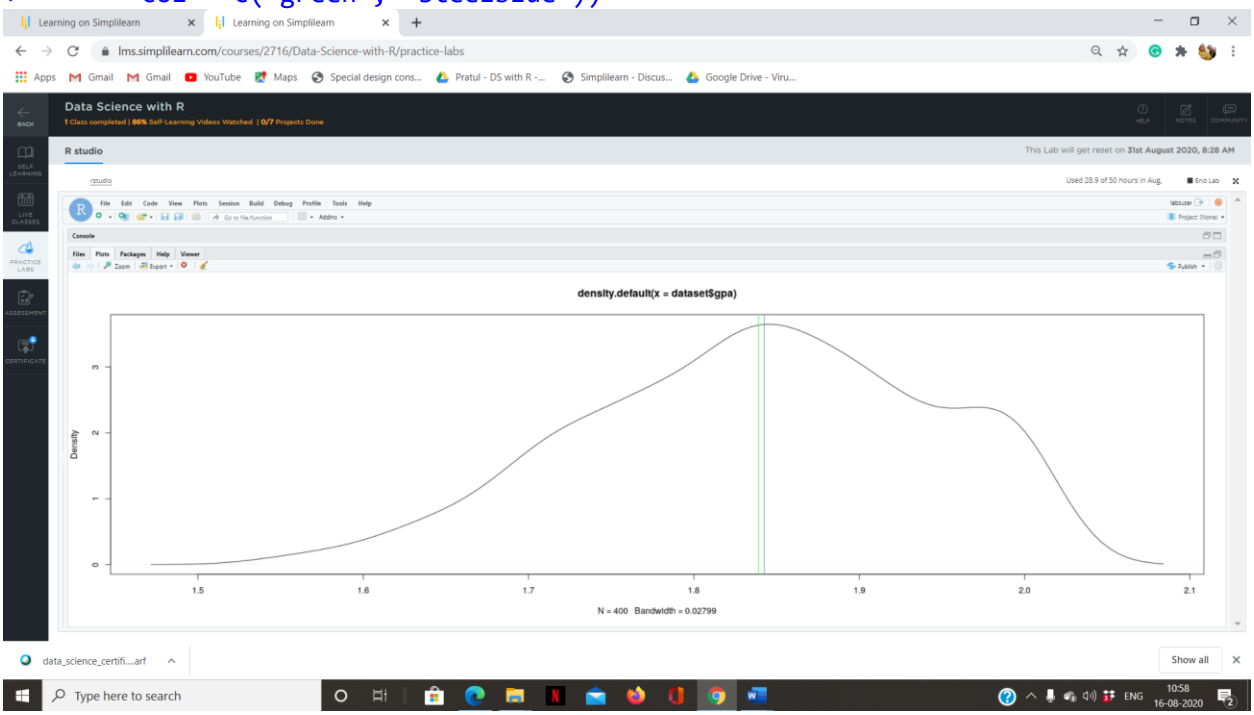

```
> #in visualization also mean is not equal to median in case of gre column of dat
aset hance data is not normaly distributed

> mean(dataset$gpa)
[1] 3.392725
> median(dataset$gpa)
[1] 3.395
> #here mean is not equal to median in case of gpa column of dataset hance data i
s not normaly distributed
> #by visualization
> plot(density(dataset$gpa))
> abline(v = c(mean(dataset$gpa),
+              median(dataset$gpa)),
+        col = c('green', 'steelblue'))
```




```
> library(moments)
> skewness(dataset$gre, na.rm = TRUE)
[1] -0.02328672
> skewness(dataset$gpa, na.rm = TRUE)
[1] -0.1696461
> #both gre and gpa distribution have negative skewness.
> install.packages("ggpubr")
> library(magrittr)
> library(ggpubr)
> dataset$gre = sqrt(dataset$gre)
> dataset$gpa = sqrt(dataset$gpa)
> plot(density(dataset$gre))
> abline(v = c(mean(dataset$gre),
+              median(dataset$gre)),
+        col = c('green', 'steelblue'))
```



```
> plot(density(dataset$gpa))
> abline(v = c(mean(dataset$gpa),
+              median(dataset$gpa)),
+        col = c('green', 'steelblue'))
```




```
> mean(dataset$gre)
[1] 24.19508
> median(dataset$gre)
[1] 24.24252
> mean(dataset$gpa)
[1] 1.839056
> median(dataset$gpa)
[1] 1.842552
```

```
> #now the data is normally distributed
> #task6-Use variable reduction techniques to identify significant variables.
> # Splitting the dataset into the Training set and Test set
> install.packages('caTools')
> library(caTools)
> set.seed(125)
> split = sample.split(dataset$admit,SplitRatio = 0.8)
> training_set = subset(dataset,split == TRUE)
> test_set = subset(dataset,split == FALSE)
> ## Feature Scaling
> training_set[,2:3] =scale(training_set[,2:3])
> test_set[,2:3] =scale(test_set[,2:3])
> # Fitting Multiple Linear Regression to the Training set
> regressor = lm(formula = admit ~.,data = training_set)
> summary(regressor)
```

```
Call:
lm(formula = admit ~ ., data = training_set)

Residuals:
    Min      1Q  Median      3Q     Max
-0.7908 -0.3375 -0.1726  0.4673  0.9247

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.619667   0.083188   7.449 9.47e-13 ***
gre           0.061700   0.026881   2.295 0.022385 *
gpa           0.048105   0.026803   1.795 0.073673 .
ses2         -0.009216   0.060221  -0.153 0.878463
ses3         -0.029756   0.061847  -0.481 0.630773
Gender_Male1 -0.059993   0.050035  -1.199 0.231439
Race2        -0.124931   0.060298  -2.072 0.039104 *
Race3        -0.065092   0.061066  -1.066 0.287286
rank2        -0.140065   0.075624  -1.852 0.064962 .
rank3        -0.299029   0.079183  -3.776 0.000191 ***
rank4        -0.345314   0.088168  -3.917 0.000111 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4422 on 309 degrees of freedom
Multiple R-squared:  0.1304,    Adjusted R-squared:  0.1023
F-statistic: 4.634 on 10 and 309 DF,  p-value: 3.678e-06

> #backward elimination for eliminating
> regressor = lm(formula = admit ~gre+gpa+rank+Race+ses+Gender_Male,data = traini
ng_set)
> summary(regressor)

Call:
lm(formula = admit ~ gre + gpa + rank + Race + ses + Gender_Male,
    data = training_set)

Residuals:
    Min      1Q  Median      3Q     Max
-0.7908 -0.3375 -0.1726  0.4673  0.9247

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   0.619667   0.083188   7.449 9.47e-13 ***
gre           0.061700   0.026881   2.295 0.022385 *
gpa           0.048105   0.026803   1.795 0.073673 .
rank2        -0.140065   0.075624  -1.852 0.064962 .
rank3        -0.299029   0.079183  -3.776 0.000191 ***
rank4        -0.345314   0.088168  -3.917 0.000111 ***
Race2        -0.124931   0.060298  -2.072 0.039104 *
Race3        -0.065092   0.061066  -1.066 0.287286
ses2         -0.009216   0.060221  -0.153 0.878463
ses3         -0.029756   0.061847  -0.481 0.630773
```

```
Gender_Male1 -0.059993   0.050035  -1.199 0.231439
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4422 on 309 degrees of freedom
Multiple R-squared:  0.1304,    Adjusted R-squared:  0.1023
F-statistic: 4.634 on 10 and 309 DF,  p-value: 3.678e-06

> regressor = lm(formula = admit ~gre+gpa+rank+Race+ses,data = training_set)
> summary(regressor)

Call:
lm(formula = admit ~ gre + gpa + rank + Race + ses, data = training_set)

Residuals:
    Min      1Q  Median      3Q     Max
-0.7540 -0.3380 -0.1639  0.4571  0.9529

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.58534    0.07816   7.489 7.28e-13 ***
gre          0.06102    0.02689   2.269 0.023950 *
gpa          0.04955    0.02680   1.849 0.065389 .
rank2       -0.13603    0.07560  -1.799 0.072950 .
rank3       -0.29130    0.07898  -3.688 0.000266 ***
rank4       -0.34285    0.08821  -3.887 0.000124 ***
Race2       -0.11880    0.06012  -1.976 0.049051 *
Race3       -0.06316    0.06109  -1.034 0.301985
ses2        -0.01258    0.06020  -0.209 0.834602
ses3        -0.02855    0.06188  -0.461 0.644872
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4425 on 310 degrees of freedom
Multiple R-squared:  0.1264,    Adjusted R-squared:  0.101
F-statistic: 4.983 on 9 and 310 DF,  p-value: 2.839e-06

> regressor = lm(formula = admit ~gre+gpa+rank+ses,data = training_set)
> summary(regressor)

Call:
lm(formula = admit ~ gre + gpa + rank + ses, data = training_set)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6831 -0.3413 -0.1701  0.4849  0.9472

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.52598    0.07127   7.380 1.44e-12 ***
gre          0.06191    0.02693   2.299 0.022155 *
gpa          0.04800    0.02679   1.792 0.074128 .
```

```
rank2       -0.12157    0.07522  -1.616 0.107052
rank3       -0.28682    0.07890  -3.635 0.000325 ***
rank4       -0.33988    0.08844  -3.843 0.000147 ***
ses2        -0.02341    0.05996  -0.390 0.696434
ses3        -0.03625    0.06168  -0.588 0.557135
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4439 on 312 degrees of freedom
Multiple R-squared:  0.1153,    Adjusted R-squared:  0.09545
F-statistic: 5.809 on 7 and 312 DF,  p-value: 2.408e-06

> regressor = lm(formula = admit ~gre+gpa+rank,data = training_set)
> summary(regressor)

Call:
lm(formula = admit ~ gre + gpa + rank, data = training_set)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6880 -0.3348 -0.1719  0.4777  0.9666

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.50776    0.06356   7.988 2.62e-14 ***
gre          0.06195    0.02680   2.312 0.021440 *
gpa          0.04774    0.02670   1.788 0.074758 .
rank2       -0.12267    0.07493  -1.637 0.102624
rank3       -0.28872    0.07860  -3.673 0.000281 ***
rank4       -0.34111    0.08809  -3.872 0.000131 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4427 on 314 degrees of freedom
Multiple R-squared:  0.1143,    Adjusted R-squared:  0.1002
F-statistic: 8.103 on 5 and 314 DF,  p-value: 3.306e-07

> regressor = lm(formula = admit ~gpa+rank,data = training_set)
> summary(regressor)

Call:
lm(formula = admit ~ gpa + rank, data = training_set)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6282 -0.3312 -0.1832  0.5010  0.9303

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.51839    0.06383   8.121 1.06e-14 ***
gpa          0.06993    0.02509   2.787 0.005645 **
rank2       -0.12915    0.07539  -1.713 0.087701 .
```

```
rank3       -0.30661    0.07875  -3.893 0.000121 ***
rank4       -0.35841    0.08837  -4.056 6.31e-05 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4458 on 315 degrees of freedom
Multiple R-squared:  0.09921,   Adjusted R-squared:  0.08777
F-statistic: 8.673 on 4 and 315 DF,  p-value: 1.186e-06

> #task7-Run logistic model to determine the factors that influence the admission
 process of a student (Drop insignificant variables)
> #Encoding the target feature as factor
> dataset$admit = factor(dataset$admit,)
> class(dataset$admit)
[1] "factor"
> levels(dataset$admit)
[1] "0" "1"
> # Splitting the dataset into the Training set and Test set
> install.packages('caTools')
Error in install.packages : Updating loaded packages
> library(caTools)
> set.seed(169)
> split = sample.split(dataset$admit,SplitRatio = 0.75)
> training_set = subset(dataset,split == TRUE)
> test_set = subset(dataset,split == FALSE)
> ## Feature Scaling
> training_set[,2:3] =scale(training_set[,2:3])
> test_set[,2:3] =scale(test_set[,2:3])
> # Fitting Logistic Regression to the Training set
> classifier = glm(formula = admit ~.,
+                  family = binomial,
+                  data = training_set)
> summary(classifier)

Call:
glm(formula = admit ~ ., family = binomial, data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8154  -0.8470  -0.5981   1.0547   2.1786

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.6269     0.4091   1.533 0.125381
gre            0.3978     0.1520   2.617 0.008874 **
gpa            0.2210     0.1481   1.492 0.135590
ses2          -0.2520     0.3240  -0.778 0.436850
ses3          -0.4869     0.3277  -1.486 0.137297
Gender_Male1  -0.2514     0.2713  -0.926 0.354210
Race2         -0.5256     0.3329  -1.579 0.114408
Race3         -0.3322     0.3215  -1.033 0.301466
rank2         -0.6246     0.3648  -1.712 0.086886 .
```

```
rank3         -1.2316     0.3943  -3.123 0.001787 **
rank4         -1.7735     0.4936  -3.593 0.000327 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.6  on 299  degrees of freedom
Residual deviance: 333.0  on 289  degrees of freedom
AIC: 355

Number of Fisher Scoring iterations: 4

> # Predicting the Test set results
> prob_pred = predict(classifier, type = 'response',newdata = test_set[,-1])
> y_pred = ifelse(prob_pred>0.5 ,1 ,0)
> #making confusion matrix for test set
> m  = table(test_set[,1],y_pred>0.5)
> m

    FALSE TRUE
  0    64    4
  1    27    5
> #checking accuracy of the model over test set (acc0rding to task 8)
> accuracy =(64+5)/(64+27+5+4)*100 #69
> classifier = glm(formula = admit ~gre+gpa+rank+Race+ses+Gender_Male,
+                  family = binomial,
+                  data = training_set)
> summary(classifier)

Call:
glm(formula = admit ~ gre + gpa + rank + Race + ses + Gender_Male,
    family = binomial, data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8154  -0.8470  -0.5981   1.0547   2.1786

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.6269     0.4091   1.533 0.125381
gre            0.3978     0.1520   2.617 0.008874 **
gpa            0.2210     0.1481   1.492 0.135590
rank2         -0.6246     0.3648  -1.712 0.086886 .
rank3         -1.2316     0.3943  -3.123 0.001787 **
rank4         -1.7735     0.4936  -3.593 0.000327 ***
Race2         -0.5256     0.3329  -1.579 0.114408
Race3         -0.3322     0.3215  -1.033 0.301466
ses2          -0.2520     0.3240  -0.778 0.436850
ses3          -0.4869     0.3277  -1.486 0.137297
Gender_Male1  -0.2514     0.2713  -0.926 0.354210
---
```

```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.6  on 299  degrees of freedom
Residual deviance: 333.0  on 289  degrees of freedom
AIC: 355

Number of Fisher Scoring iterations: 4

> classifier = glm(formula = admit ~gre+gpa+rank+Race+ses,
+                  family = binomial,
+                  data = training_set)
> summary(classifier)

Call:
glm(formula = admit ~ gre + gpa + rank + Race + ses, family = binomial,
    data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8027  -0.8546  -0.5903   1.0647   2.2240

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.4760     0.3731   1.276 0.201998
gre           0.3940     0.1517   2.598 0.009379 **
gpa           0.2282     0.1478   1.544 0.122566
rank2        -0.6128     0.3632  -1.687 0.091550 .
rank3        -1.1959     0.3906  -3.062 0.002198 **
rank4        -1.7505     0.4919  -3.559 0.000372 ***
Race2        -0.4983     0.3306  -1.507 0.131781
Race3        -0.3182     0.3208  -0.992 0.321215
ses2         -0.2654     0.3227  -0.822 0.410839
ses3         -0.4740     0.3269  -1.450 0.147093
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.60  on 299  degrees of freedom
Residual deviance: 333.86  on 290  degrees of freedom
AIC: 353.86

Number of Fisher Scoring iterations: 4

> classifier = glm(formula = admit ~gre+gpa+rank,
+                  family = binomial,
+                  data = training_set)
> classifier = glm(formula = admit ~gre+gpa+rank+Race,
+                  family = binomial,
+                  data = training_set)
```

```
> summary(classifier)

Call:
glm(formula = admit ~ gre + gpa + rank + Race, family = binomial,
    data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7666  -0.8674  -0.6088   1.1129   2.2986

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.2665     0.3405   0.783 0.433772
gre           0.3986     0.1503   2.651 0.008018 **
gpa           0.2194     0.1471   1.492 0.135796
rank2        -0.6333     0.3625  -1.747 0.080606 .
rank3        -1.2158     0.3892  -3.124 0.001785 **
rank4        -1.7357     0.4893  -3.547 0.000389 ***
Race2        -0.5377     0.3281  -1.639 0.101278
Race3        -0.3012     0.3195  -0.943 0.345697
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.60  on 299  degrees of freedom
Residual deviance: 336.03  on 292  degrees of freedom
AIC: 352.03

Number of Fisher Scoring iterations: 4

> summary(classifier)

Call:
glm(formula = admit ~ gre + gpa + rank + Race, family = binomial,
    data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7666  -0.8674  -0.6088   1.1129   2.2986

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.2665     0.3405   0.783 0.433772
gre           0.3986     0.1503   2.651 0.008018 **
gpa           0.2194     0.1471   1.492 0.135796
rank2        -0.6333     0.3625  -1.747 0.080606 .
rank3        -1.2158     0.3892  -3.124 0.001785 **
rank4        -1.7357     0.4893  -3.547 0.000389 ***
Race2        -0.5377     0.3281  -1.639 0.101278
Race3        -0.3012     0.3195  -0.943 0.345697
---
```

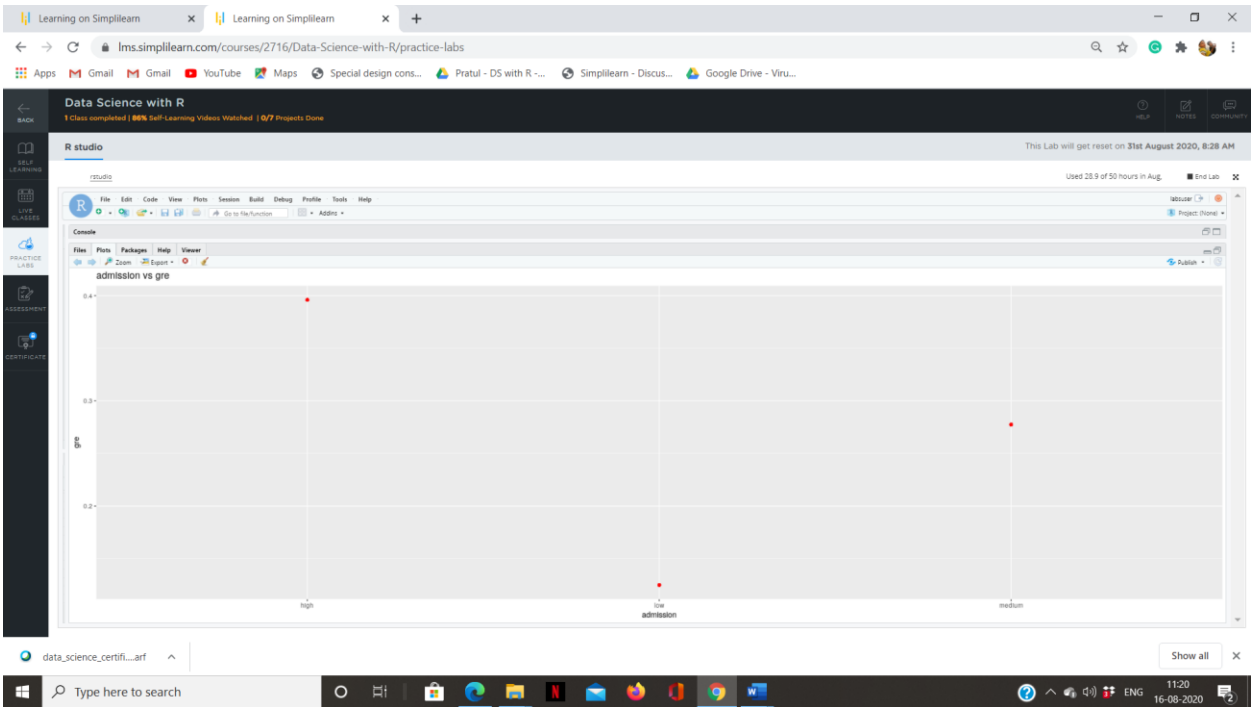
```
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.60  on 299  degrees of freedom
Residual deviance: 336.03  on 292  degrees of freedom
AIC: 352.03

Number of Fisher Scoring iterations: 4

> classifier = glm(formula = admit ~gpa+rank,
+                  family = binomial,
+                  data = training_set)
> summary(classifier)

Call:
glm(formula = admit ~ gpa + rank, family = binomial, data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4310  -0.8831  -0.6258   1.1694   2.1385

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.0295     0.2828   0.104 0.916898
gpa           0.3497     0.1356   2.578 0.009941 **
rank2        -0.5647     0.3487  -1.620 0.105317
rank3        -1.2696     0.3790  -3.350 0.000809 ***
rank4        -1.7255     0.4783  -3.608 0.000309 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.60  on 299  degrees of freedom
Residual deviance: 346.15  on 295  degrees of freedom
AIC: 356.15

Number of Fisher Scoring iterations: 4

> classifier1 = glm(formula = admit ~rank+gpa,
+                  family = binomial,
+                  data = training_set)
> summary(classifier1)

Call:
glm(formula = admit ~ rank + gpa, family = binomial, data = training_set)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.4310  -0.8831  -0.6258   1.1694   2.1385
```

```
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.0295     0.2828   0.104 0.916898
rank2        -0.5647     0.3487  -1.620 0.105317
rank3        -1.2696     0.3790  -3.350 0.000809 ***
rank4        -1.7255     0.4783  -3.608 0.000309 ***
gpa           0.3497     0.1356   2.578 0.009941 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 374.60  on 299  degrees of freedom
Residual deviance: 346.15  on 295  degrees of freedom
AIC: 356.15

Number of Fisher Scoring iterations: 4

> #task 8-Calculate the accuracy of the model and run validation techniques
> # Predicting the Test set results
> prob_pred1 = predict(classifier1, type = 'response',newdata = test_set[,-1])
> y_pred1 = ifelse(prob_pred1>0.5 ,1 ,0)
> #making confusion matrix for test set
> m  = table(test_set[,1],y_pred1>0.5)
> m

    FALSE TRUE
  0    66    2
  1    26    6
> accuracy_of_logistic_regression = ((66+6)/(66+26+6+2))*100 #72
> #task 9-Try other modelling techniques like decision tree and SVM and select a
champion model
> #1.svm
> # Fitting SVM to the Training set, with kernel as linear
> install.packages(e1071)
> classifier = svm(formula = admit~.,
+                  data = training_set,
+                  type = 'C-classification',
+                  kernel = 'linear')
> library(e1071)
> # Predicting the Test set results
> y_pred = predict(classifier,newdata = test_set[,-1])
> # Making the Confusion Matrix
> cm  = table(test_set[,1],y_pred)
> cm
   y_pred
     0  1
  0 68  0
  1 32  0
> accuracy_of_svm= (68/100*100)
> classifier = svm(formula = admit~.,
+                  data = training_set,
```

```
+                 type = 'C-classification',
+                 kernel = 'linear')
> # Predicting the Test set results
> y_pred = predict(classifier,newdata = test_set[,-1])
> # Making the Confusion Matrix
> cm  = table(test_set[,1],y_pred)
> cm
   y_pred
     0  1
  0 68  0
  1 32  0
> # Fitting SVM to the Training set, with kernel as redial
> classifier = svm(formula = admit~.,
+                 data = training_set,
+                 type = 'C-classification',
+                 kernel = 'radial')
> # Predicting the Test set results
> y_pred = predict(classifier,newdata = test_set[,-1])
> # Making the Confusion Matrix
> cm  = table(test_set[,1],y_pred)
> cm
   y_pred
     0  1
  0 68  0
  1 32  0
> accuracy =68/100*100 #68
> #2.knn
> # Fitting KNN to the Training set
> install.packages(class)
> y_pred = knn(train = training_set[,-1],
+              test = test_set[,-1],
+              cl = training_set[,1],
+              k = 25,
+              prob = T)  #3,5,7,9
> library(class)
> # Making the Confusion Matrix
> cm  = table(test_set[,1],y_pred)
> cm
   y_pred
     0  1
  0 62  6
  1 26  6
> accuracy_of_knn = (62+6)/(6+62+6+26)*100 #68
> #3-decision tree
> # Fitting Decision Tree Classification to the Training set
> # Fitting Decision Tree Classification to the Training set
> install.packages('rpart')
> library(rpart)
> classifier = rpart(formula = admit~ .,
+                    data = training_set)
> # Predicting the Test set results
> y_pred = predict(classifier, newdata = test_set[-1],type = 'class')
```

```
> # Making the Confusion Matrix
> cm = table(test_set[,1], y_pred)
> cm
   y_pred
     0  1
  0 58 10
  1 22 10
> accuracy_of_decision_tree = (58+10)/(58+10+22+14)*100 #68

> #task10-Determine the accuracy rates for each kind of model.
> accuracy_of_logistic_regression = ((66+6)/(66+26+6+2))*100 #72
> accuracy_of_svm= (68+0) /(68+32+0+0)*100 #68%
> accuracy_of_knn = (62+6)/(6+62+6+26)*100 #68
> accuracy_of_decision_tree = (54+10)/(54+10+22+14)*100 #64
> #hance we chose logistic regression model and hance it is the chempion model.
> #task10-Determine the accuracy rates for each kind of model.
> accuracy_of_logistic_regression = ((66+6)/(66+26+6+2))*100 #72
> accuracy_of_svm= (68+0) /(68+32+0+0)*100 #68%
> accuracy_of_knn = (62+6)/(6+62+6+26)*100 #68
> accuracy_of_decision_tree = (54+10)/(54+10+22+14)*100 #64
> #hance we chose logistic regression model and hance it is the chempion model.
> # Importing the dataset
> # Fitting Random Forest Classification to the Training set
> install.packages('randomForest')
>library(randomForest)
randomForest 4.6-14
Type rfNews() to see new features/changes/bug fixes.

Attaching package: ‘randomForest’

The following object is masked from ‘package:ggplot2’:

    margin

> set.seed(321)
> classifier = randomForest(x = training_set[-1],
+                           y = training_set$admit,
+                           ntree = 180)
> # Predicting the Test set results
> y_pred = predict(classifier, newdata = test_set[-1])
> # Making the Confusion Matrix
> cm = table(test_set[, 1], y_pred)
> cm
   y_pred
     0  1
  0 55 13
  1 24  8



#B->Descriptive:
> #Categorize the average of grade point into High, Medium, and Low (with admission probabilit
 a point chart.Cross grid for admission variables with GRE Categorization is shown below:
```

```
> #GRE  Categorized
> #0-440        Low
> #440-580Medium
> #580+ High
> dataset <- read.csv("College_admission.csv")
> gre_grade1 = dataset$gre <= 440
> gre_grade2 = dataset$gre > 440 & dataset$gre <=580
> gre_grade3 = dataset$gre > 580
> dataset$gre[gre_grade1] = "low"
> dataset$gre[gre_grade2] = "medium"
> dataset$gre[gre_grade3] = "high"
> #Encoding the target feature as factor
> # Splitting the dataset into the Training set and Test set
> install.packages('caTools')
> library(caTools)
> set.seed(169)
> split = sample.split(dataset$admit,SplitRatio = 0.75)
> training_set = subset(dataset,split == TRUE)
> test_set = subset(dataset,split == FALSE)
> #now there can be two possible outcome of the task.
> #1>to compaire admission withe the gre column and making the visualization
> # Fitting Regression to the dataset
> regressor = lm(formula = admit ~ gre,data =dataset)
> summary(regressor)

Call:
lm(formula = admit ~ gre, data = dataset)

Residuals:
    Min      1Q  Median      3Q     Max
-0.3959 -0.3959 -0.2774  0.6041  0.8750

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.39594    0.03266  12.122  < 2e-16 ***
grelow      -0.27094    0.07379  -3.672 0.000274 ***
gremedium   -0.11852    0.04922  -2.408 0.016501 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4584 on 397 degrees of freedom
Multiple R-squared:  0.03738,   Adjusted R-squared:  0.03253
F-statistic: 7.708 on 2 and 397 DF,  p-value: 0.00052

> y =predict(regressor, type = 'response')
> library(ggplot2)
> ggplot() +
+         geom_point(aes(x=dataset$gre, y =predict(regressor, type = 'response')),
+                    colour = 'red') +
+                 ggtitle('admission vs gre ') +
+         xlab('admission')+
+         ylab('gre')
```




```
> #2>to compare the admission with all the other column and than making the visualization.
> regressor = lm(formula = admit ~ .,data =dataset)
> y =predict(regressor, type = 'response')
> summary(regressor)

Call:
lm(formula = admit ~ ., data = dataset)

Residuals:
    Min      1Q  Median      3Q     Max
-0.6606 -0.3434 -0.1892  0.4977  0.9811

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.24330    0.24206   1.005   0.3155
grelow      -0.16446    0.07675  -2.143   0.0327 *
gremedium   -0.07140    0.04977  -1.435   0.1522
gpa          0.15397    0.06324   2.435   0.0153 *
ses         -0.02667    0.02765  -0.965   0.3353
Gender_Male -0.03037    0.04465  -0.680   0.4968
Race        -0.03175    0.02740  -1.159   0.2473
rank        -0.10884    0.02376  -4.581 6.22e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.4449 on 392 degrees of freedom
Multiple R-squared:  0.1049,    Adjusted R-squared:  0.08892
F-statistic: 6.563 on 7 and 392 DF,  p-value: 2.423e-07
```

```
> library(ggplot2)
> ggplot() +
+         geom_point(aes(x=dataset$gre, y =predict(regressor, type = 'response')),
+                    colour = 'red') +
+         ggtitle('admission vs gre ') +
+         xlab('admission')+
+         ylab('gre')
```