IST 707 HW4: Regression and Artificial Neural Network/Deep Learning

Sanjeev Ramasamy Seenivasagamani

April 27, 2019

Load the required libraries:

```
require(foreign)
require(ggplot2)
require(MASS)
require(Hmisc)
require(reshape2)
library(rms)
library(data.table)
library(tidyverse)
library(tidytext)
library(caret)
library(rpart)
library("rpart.plot")
library(recipes)
library(randomForest)
library(keras)
```

Introduction

About:

This is an educational data set which is collected from learning management system (LMS) called Kalboard 360. Kalboard 360 is a multi-agent LMS, which has been designed to facilitate learning through the use of leading-edge technology. Such system provides users with a synchronous access to educational resources from any device with Internet connection.

The dataset consists of 305 males and 175 females. The students come from different origins such as 179 students are from Kuwait, 172 students are from Jordan, 28 students from Palestine, 22 students are from Iraq, 17 students from Lebanon, 12 students from Tunis, 11 students from Saudi Arabia, 9 students from Egypt, 7 students from Syria, 6 students from USA, Iran and Libya, 4 students from Morocco and one student from Venezuela.

This dataset includes also a new category of features; this feature is parent parturition in the educational process. Parent participation feature have two sub features: Parent Answering Survey and Parent School Satisfaction. There are 270 of the parents answered survey and 210 are not, 292 of the parents are satisfied from the school and 188 are not.

The students are classified into three numerical intervals based on their total grade/mark:

Low-Level: interval includes values from 0 to 69,

Middle-Level: interval includes values from 70 to 89,

High-Level: interval includes values from 90-100.

Data load:

```
data <- read.csv("Students Academic Performance.csv", header = TRUE)
str(data)
```

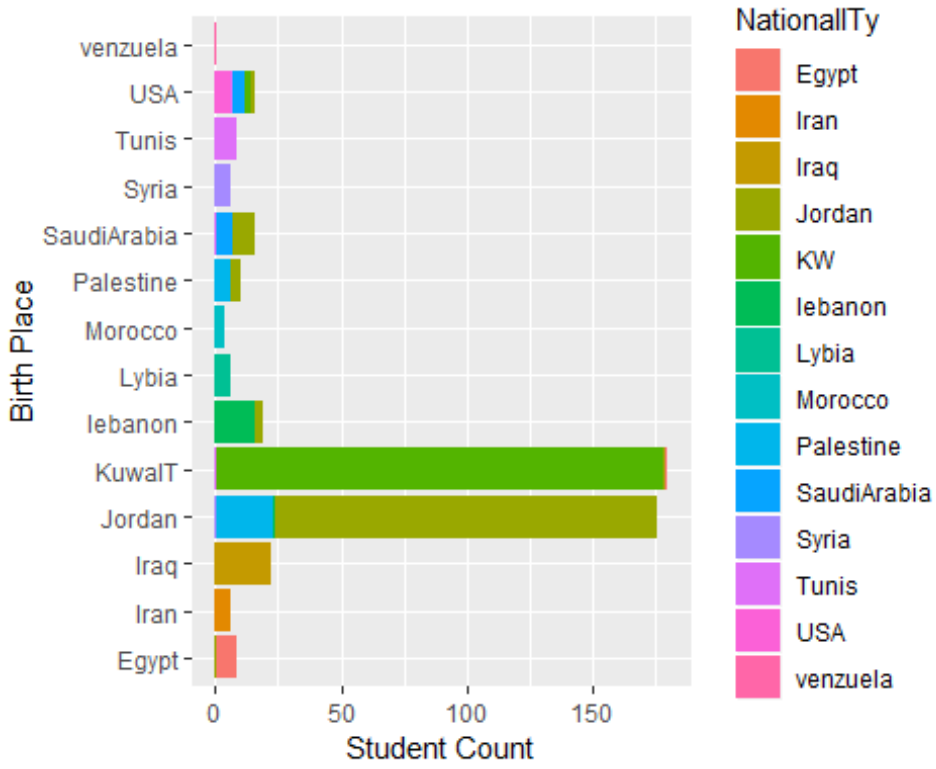
```
## 'data.frame':    480 obs. of  17 variables:
##  $ gender                  : Factor w/ 2 levels "F","M": 2 2 2 2 2 1 2 2 1
1 ...
##  $ NationalITy             : Factor w/ 14 levels "Egypt","Iran",..: 5 5 5
5 5 5 5 5 5 5 ...
##  $ PlaceofBirth            : Factor w/ 14 levels "Egypt","Iran",..: 5 5 5
5 5 5 5 5 5 5 ...
##  $ StageID                 : Factor w/ 3 levels "HighSchool","lowerlevel",
..: 2 2 2 2 2 2 3 3 3 3 ...
##  $ GradeID                 : Factor w/ 10 levels "G-02","G-04",..: 2 2 2 2
2 2 5 5 5 5 ...
##  $ SectionID               : Factor w/ 3 levels "A","B","C": 1 1 1 1 1 1 1
1 1 2 ...
##  $ Topic                   : Factor w/ 12 levels "Arabic","Biology",..: 8
8 8 8 8 9 9 9 8 ...
##  $ Semester                : Factor w/ 2 levels "F","S": 1 1 1 1 1 1 1 1 1
1 ...
##  $ Relation                : Factor w/ 2 levels "Father","Mum": 1 1 1 1 1
1 1 1 1 1 ...
##  $ raisedhands             : int  15 20 10 30 40 42 35 50 12 70 ...
##  $ VisITedResources        : int  16 20 7 25 50 30 12 10 21 80 ...
##  $ AnnouncementsView       : int  2 3 0 5 12 13 0 15 16 25 ...
##  $ Discussion              : int  20 25 30 35 50 70 17 22 50 70 ...
##  $ ParentAnsweringSurvey   : Factor w/ 2 levels "No","Yes": 2 2 1 1 1 2 1
2 2 2 ...
##  $ ParentschoolSatisfaction: Factor w/ 2 levels "Bad","Good": 2 2 1 1 1 1
1 2 2 2 ...
##  $ StudentAbsenceDays      : Factor w/ 2 levels "Above-7","Under-7": 2 2 1
1 1 1 1 2 2 2 ...
##  $ Class                   : Factor w/ 3 levels "H","L","M": 3 3 2 2 3 3 2
3 3 3 ...
```

Exploratory Data Analysis

Let us have a look at the data distribution through various plots.

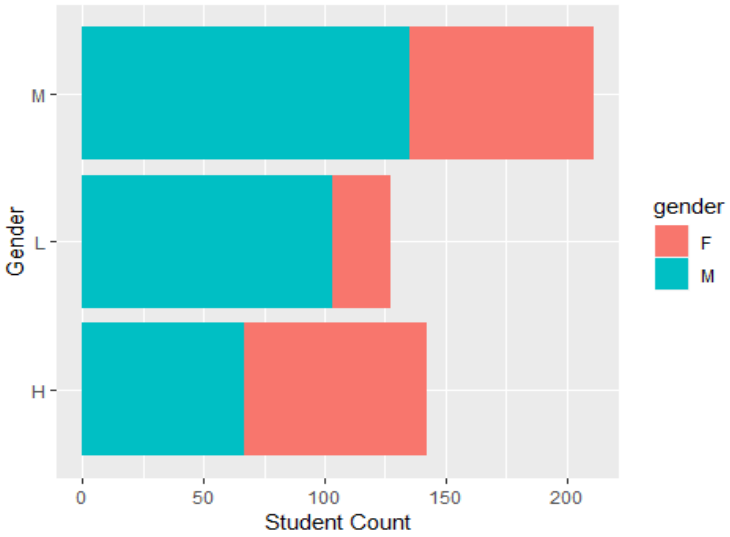
Studentcount vs Birthplace

```
ggplot(data = data, aes(x = PlaceofBirth)) + geom_bar(aes(fill = NationalITy)
) +
    labs(x = "Birth Place", y = "Student Count") + coord_flip()
```



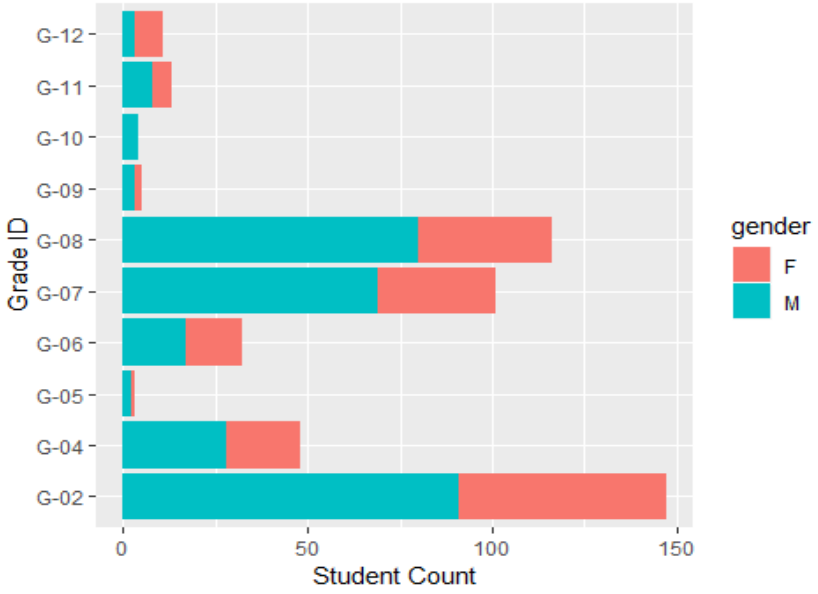
Studentcount vs Gender

```
ggplot(data = data, aes(x = Class, fill = gender)) + geom_bar() +
    labs(x = "Gender", y = "Student Count") + coord_flip()
```



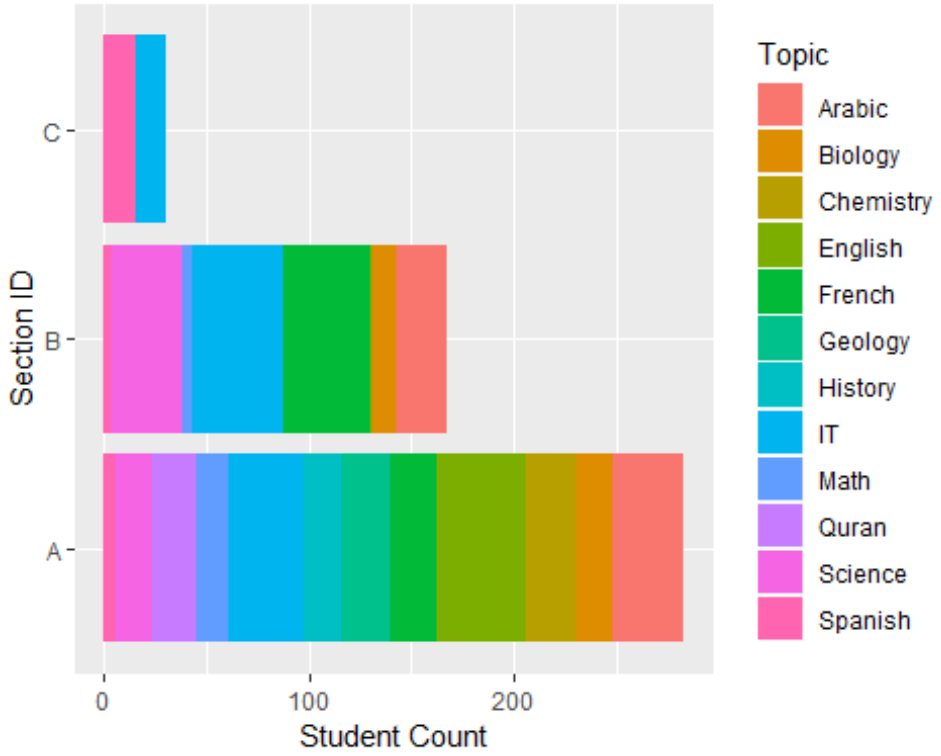
Studentcount vs GradeID

```
ggplot(data = data, aes(x = GradeID, fill = gender)) + geom_bar() +
    labs(x = "Grade ID", y = "Student Count") + coord_flip()
```



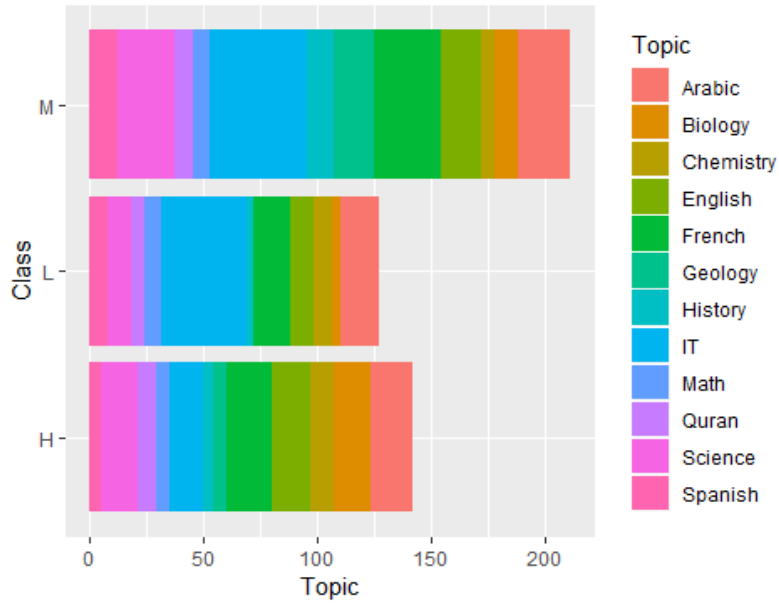
Studentcount vs Section ID

```
ggplot(data = data, aes(x = SectionID, fill = Topic)) + geom_bar() +
    labs(x = "Section ID", y = "Student Count") +
    coord_flip()
```



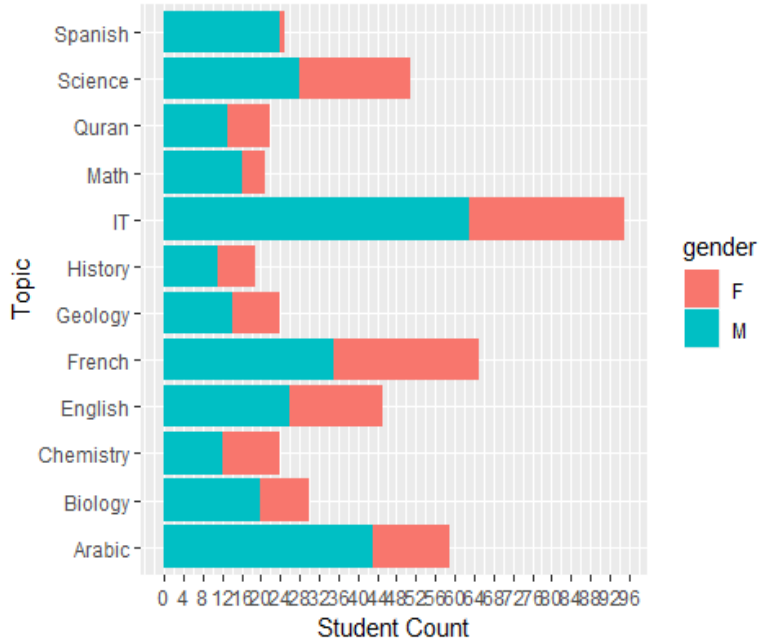
Topic vs Class

```
ggplot(data = data, aes(x = Class, fill = Topic)) + geom_bar() +
    labs(x = "Class", y = "Topic") +
    coord_flip()
```



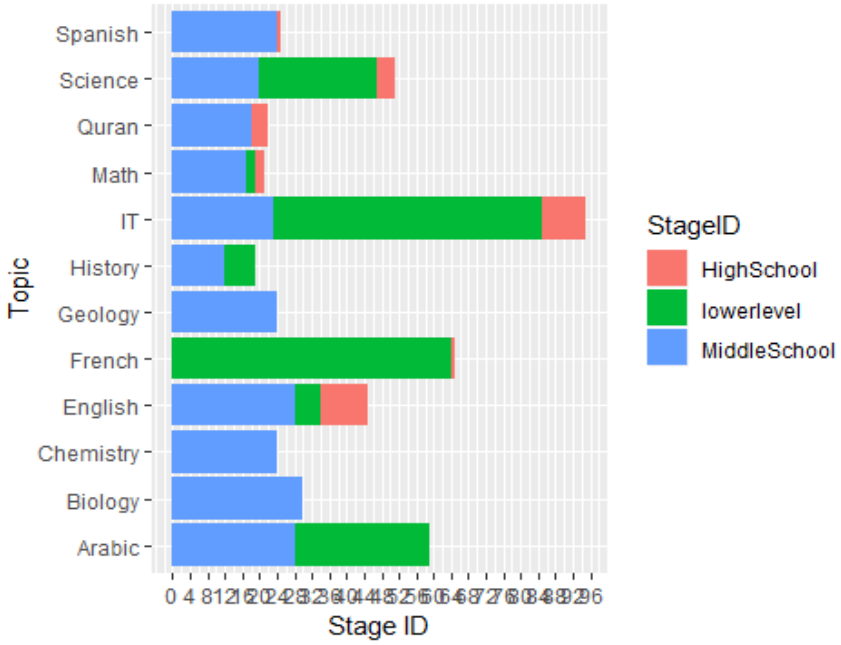
Gender vs Topic

```
ggplot(data = data, aes(x = Topic, fill = gender)) + geom_bar() +
    labs(x = "Topic", y = "Student Count") +
    scale_y_continuous(breaks = seq(0,100,4)) + coord_flip()
```



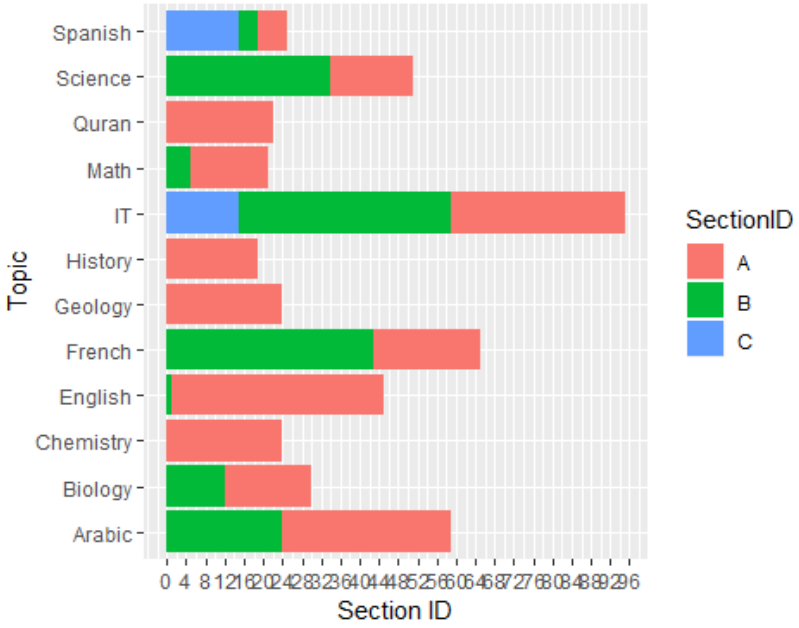
Topic vs Stage ID

```
ggplot(data = data, aes(x = Topic, fill = StageID)) + geom_bar() +
    labs(x = "Topic", y = "Stage ID") + coord_flip() +
    scale_y_continuous(breaks = seq(0,100,4))
```



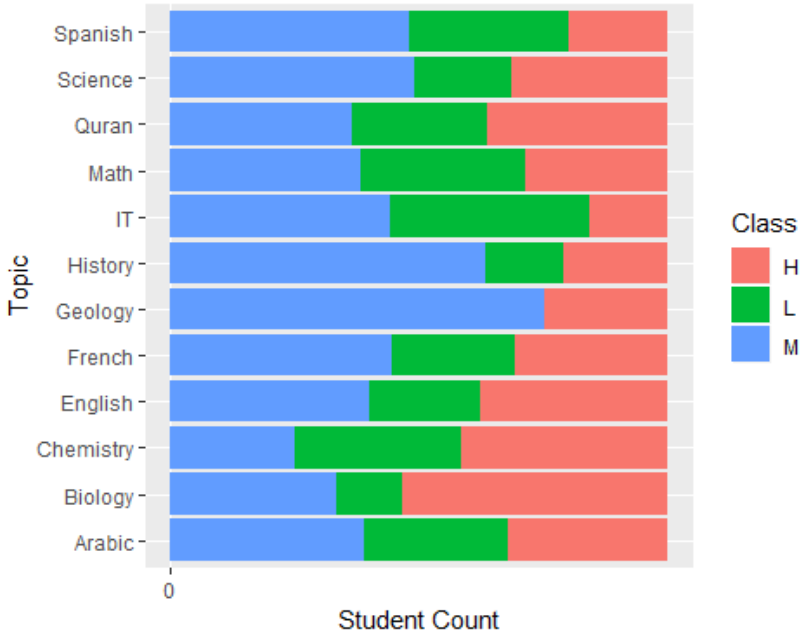
Topic vs SectionID

```
ggplot(data = data, aes(x = Topic, fill = SectionID)) + geom_bar() +
    labs(x = "Topic", y = "Section ID") + coord_flip() +
    scale_y_continuous(breaks = seq(0,100,4))
```



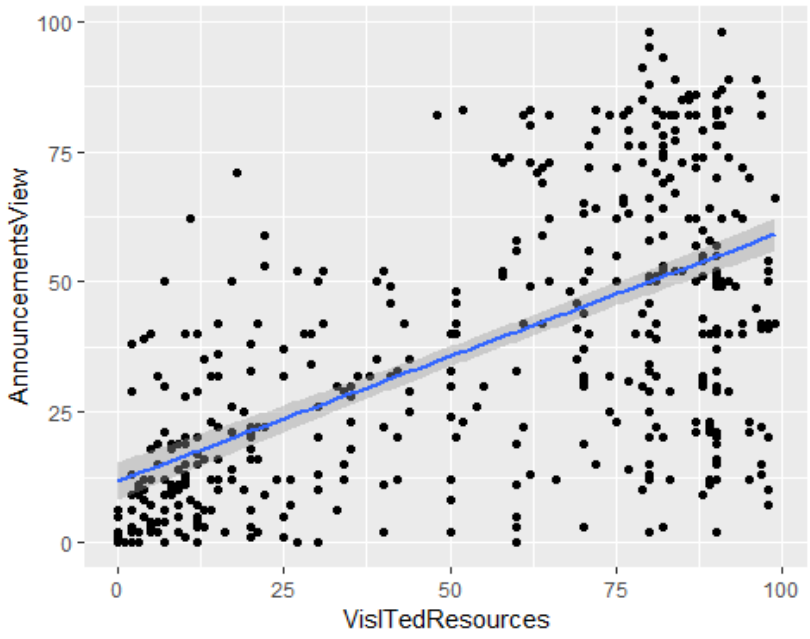
Topic vs Student Count

```
ggplot(data = data, aes(x = Topic, fill = Class)) + geom_bar(position = "fill
") +
    labs(x = "Topic", y = "Student Count") + coord_flip() +
    scale_y_continuous(breaks = seq(0,100,4))
```



AnnouncementViews vs VisitedResources

```
ggplot(data = data, aes( x = VisITedResources, y = AnnouncementsView)) + geom
_point() +
    geom_smooth(method = "lm")
```



Ordinal Logistic Regression

Data Manipulation:

As you can see that some of the categorical variables have more than 9 levels. This can cause issues while training the Ordinal Regression Model. So, let us try to convert the ordinal data to numbered ranking.

```
#Converting the variables to character prior to ranking them
data$NationalITy<- as.character(data$NationalITy)
data$PlaceofBirth<- as.character(data$PlaceofBirth)
data$GradeID<- as.character(data$GradeID)
data$Topic<- as.character(data$Topic)

#Nationality
data$NationalITy[data$NationalITy == "Egypt"] <- 1
data$NationalITy[data$NationalITy == "Tunis"] <- 1
data$NationalITy[data$NationalITy == "Lybia"] <- 1
data$NationalITy[data$NationalITy == "Iran"] <- 2
data$NationalITy[data$NationalITy == "Iraq"] <- 2
data$NationalITy[data$NationalITy == "Jordan"] <- 2
data$NationalITy[data$NationalITy == "KW"] <- 2
data$NationalITy[data$NationalITy == "lebanon"] <- 2
data$NationalITy[data$NationalITy == "SaudiArabia"] <- 2
data$NationalITy[data$NationalITy == "Syria"] <- 2
data$NationalITy[data$NationalITy == "Palestine"] <- 2
data$NationalITy[data$NationalITy == "Morocco"] <- 3
data$NationalITy[data$NationalITy == "venzuela"] <- 4
data$NationalITy[data$NationalITy == "USA"] <- 5

#Place of Birth
data$PlaceofBirth[data$PlaceofBirth == "Egypt"] <- 1
data$PlaceofBirth[data$PlaceofBirth == "Tunis"] <- 1
data$PlaceofBirth[data$PlaceofBirth == "Lybia"] <- 1
data$PlaceofBirth[data$PlaceofBirth == "Iran"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "Iraq"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "Jordan"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "KuwaIT"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "lebanon"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "SaudiArabia"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "Syria"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "Palestine"] <- 2
data$PlaceofBirth[data$PlaceofBirth == "Morocco"] <- 3
data$PlaceofBirth[data$PlaceofBirth == "venzuela"] <- 4
data$PlaceofBirth[data$PlaceofBirth == "USA"] <- 5

#Grade ID
data$GradeID[data$GradeID == "G-02"] <- 2
data$GradeID[data$GradeID == "G-04"] <- 4
```

```r
data$GradeID[data$GradeID == "G-05"] <- 5
data$GradeID[data$GradeID == "G-06"] <- 6
data$GradeID[data$GradeID == "G-07"] <- 7
data$GradeID[data$GradeID == "G-08"] <- 8
data$GradeID[data$GradeID == "G-09"] <- 9
data$GradeID[data$GradeID == "G-10"] <- 10
data$GradeID[data$GradeID == "G-11"] <- 11
data$GradeID[data$GradeID == "G-12"] <- 12

#Subject Topic
data$Topic[data$Topic == "Arabic"] <- 1
data$Topic[data$Topic == "Biology"] <- 2
data$Topic[data$Topic == "Chemistry"] <- 3
data$Topic[data$Topic == "English"] <- 4
data$Topic[data$Topic == "French"] <- 5
data$Topic[data$Topic == "Geology"] <- 6
data$Topic[data$Topic == "History"] <- 7
data$Topic[data$Topic == "IT"] <- 8
data$Topic[data$Topic == "Math"] <- 9
data$Topic[data$Topic == "Quran"] <- 10
data$Topic[data$Topic == "Science"] <- 11
data$Topic[data$Topic == "Spanish"] <- 12

#Converting the variables back to integer instead of factors
data$NationalITy<- as.integer(data$NationalITy)
data$PlaceofBirth<- as.integer(data$PlaceofBirth)
data$GradeID<- as.integer(data$GradeID)
data$Topic<- as.integer(data$Topic)
```

Splitting the data into train and test dataset:

```r
train_index <- createDataPartition(data$Class, p = 0.6, list = FALSE)
train <- data[train_index, ]
test <- data[-train_index, ]
```

Training the model:

```r
m <- polr(Class ~., data = train, Hess=TRUE)
summary(m)
```

```
## Call:
## polr(formula = Class ~ ., data = train, Hess = TRUE)
##
## Coefficients:
##                                Value Std. Error  t value
## genderM                     0.418532   0.252624  1.65674
## NationalITy                -0.471882   0.366569 -1.28729
## PlaceofBirth                0.362624   0.294197  1.23259
## StageIDlowerlevel          -0.112250   1.376508 -0.08155
## StageIDMiddleSchool         0.647616   0.796374  0.81321
## GradeID                    -0.045439   0.147805 -0.30743
## SectionIDB                 -0.049015   0.269727 -0.18172
```

```
## SectionIDC                      0.527530   0.499076  1.05701
## Topic                           0.014913   0.036930  0.40382
## SemesterS                      -0.176166   0.249826 -0.70515
## RelationMum                    -0.819127   0.280795 -2.91717
## raisedhands                    -0.010403   0.006287 -1.65456
## VisITedResources                0.017209   0.005641  3.05046
## AnnouncementsView              -0.001606   0.007020 -0.22884
## Discussion                     -0.002549   0.004711 -0.54102
## ParentAnsweringSurveyYes       -0.527418   0.307212 -1.71679
## ParentschoolSatisfactionGood    0.197260   0.291290  0.67719
## StudentAbsenceDaysUnder-7      -0.520893   0.305128 -1.70713
##
## Intercepts:
##     Value   Std. Error t value
## H|L -1.3125  1.8339    -0.7157
## L|M -0.0600  1.8324    -0.0327
##
## Residual Deviance: 581.8294
## AIC: 621.8294
```

We see the usual regression output coefficient table including the value of each coefficient, standard errors, t values, estimates for the two intercepts, residual deviance and AIC. AIC is the information criteria. Lesser the better.

Now we'll calculate some essential metrics such as p-Value, CI, Odds ratio:

```
ctable <- coef(summary(m))
```

Calculating the p-value from the t-value:

```
p <- pnorm(abs(ctable[, "t value"]), lower.tail = FALSE) * 2
ctable <- cbind(ctable, "p value" = p)
ctable
```

```
##                                    Value  Std. Error     t value
## genderM                         0.418532051 0.252624244  1.65673747
## NationalITy                    -0.471881534 0.366569411 -1.28729108
## PlaceofBirth                    0.362623694 0.294197263  1.23258691
## StageIDlowerlevel              -0.112250055 1.376507771 -0.08154698
## StageIDMiddleSchool             0.647615880 0.796373692  0.81320602
## GradeID                        -0.045439328 0.147805104 -0.30742733
## SectionIDB                     -0.049014579 0.269727017 -0.18171921
## SectionIDC                      0.527530213 0.499075626  1.05701458
## Topic                           0.014912917 0.036930024  0.40381553
## SemesterS                      -0.176166154 0.249826386 -0.70515431
## RelationMum                    -0.819126685 0.280795347 -2.91716616
## raisedhands                    -0.010402844 0.006287391 -1.65455642
## VisITedResources                0.017208910 0.005641422  3.05045634
## AnnouncementsView              -0.001606337 0.007019614 -0.22883550
## Discussion                     -0.002549009 0.004711496 -0.54101905
## ParentAnsweringSurveyYes       -0.527417539 0.307212204 -1.71678576
```

```
## ParentschoolSatisfactionGood  0.197259927 0.291290475  0.67719319
## StudentAbsenceDaysUnder-7    -0.520893057 0.305127712 -1.70713127
## H|L                          -1.312475888 1.833945265 -0.71565707
## L|M                          -0.059996194 1.832378992 -0.03274224
##                                  p value
## genderM                      0.097572564
## NationalITy                  0.197992851
## PlaceofBirth                 0.217729925
## StageIDlowerlevel            0.935006962
## StageIDMiddleSchool          0.416099949
## GradeID                      0.758518130
## SectionIDB                   0.855803092
## SectionIDC                   0.290504933
## Topic                        0.686348376
## SemesterS                    0.480714209
## RelationMum                  0.003532275
## raisedhands                  0.098014509
## VisITedResources             0.002284939
## AnnouncementsView            0.818996771
## Discussion                   0.588494450
## ParentAnsweringSurveyYes     0.086018323
## ParentschoolSatisfactionGood 0.498283387
## StudentAbsenceDaysUnder-7    0.087797657
## H|L                          0.474203128
## L|M                          0.973880139
```

Confidence Intervals:

```
ci <- confint(m)

## Waiting for profiling to be done...

exp(coef(m))

##                       genderM                   NationalITy
##                     1.5197290                     0.6238274
##                   PlaceofBirth              StageIDlowerlevel
##                     1.4370950                     0.8938207
##            StageIDMiddleSchool                       GradeID
##                     1.9109794                     0.9555776
##                     SectionIDB                    SectionIDC
##                     0.9521672                     1.6947415
##                         Topic                     SemesterS
##                     1.0150247                     0.8384787
##                    RelationMum                   raisedhands
##                     0.4408165                     0.9896511
##               VisITedResources              AnnouncementsView
##                     1.0173578                     0.9983950
##                    Discussion      ParentAnsweringSurveyYes
##                     0.9974542                     0.5901270
```

```
## ParentschoolSatisfactionGood    StudentAbsenceDaysUnder-7
##                    1.2180606                    0.5939898
```

```
## OR and CI
exp(cbind(OR = coef(m), ci))
```

```
##                                    OR      2.5 %     97.5 %
## genderM                     1.5197290 0.92635629  2.4976332
## NationalITy                 0.6238274 0.29626396  1.2776436
## PlaceofBirth                1.4370950 0.80629042  2.6299240
## StageIDlowerlevel           0.8938207 0.05992205 13.6713752
## StageIDMiddleSchool         1.9109794 0.40359712  9.3394435
## GradeID                     0.9555776 0.71426890  1.2786911
## SectionIDB                  0.9521672 0.56087049  1.6174612
## SectionIDC                  1.6947415 0.64540119  4.6236236
## Topic                       1.0150247 0.94394801  1.0912984
## SemesterS                   0.8384787 0.51331218  1.3688876
## RelationMum                 0.4408165 0.25298171  0.7622327
## raisedhands                 0.9896511 0.97739738  1.0018514
## VisITedResources            1.0173578 1.00634534  1.0288957
## AnnouncementsView           0.9983950 0.98469210  1.0122291
## Discussion                  0.9974542 0.98827002  1.0067293
## ParentAnsweringSurveyYes    0.5901270 0.32116240  1.0740384
## ParentschoolSatisfactionGood 1.2180606 0.68932978  2.1650191
## StudentAbsenceDaysUnder-7   0.5939898 0.32553522  1.0799350
```

Interpretation:

Since we have many independent variables we will be having a look at just three of them for interpretation purpose.

1. The gender of the student being Male increases the odds of the student being in Low or Middle or High level grades combined by 1.326 than Female

2. When the nnumber of raised hands go up by 1 unit, the odds of moving from Low level to Middle or High levels are multiplied by 0.987

3. For every parent that views the announcement the odds of the student moving from the Low scoring bracket to Medium or High level increases by 1.008

Let us now go ahead and predict the Class variable using the above model

```
predictedClass <- predict(m, test)  # predict the classes directly
head(predictedClass)
```

```
## [1] H H M M M M
## Levels: H L M
```

Confusion Matrix:

```
table(test$Class, predictedClass)
```

```
##    predictedClass
##      H  L  M
##   H 32  0 24
##   L  3  0 47
##   M 31  0 53
```

Miscassification Error:

```
mean(as.character(test$Class) != as.character(predictedClass))
```

```
## [1] 0.5526316
```

ARTIFICIAL NEURAL NETWORK

As we have manipulated the data for performing Ordinal Logistic Regression, let us now read the data again and perform train-test split for the Multilayer Perceptron modelling.

```
data <- read.csv("Students Academic Performance.csv", header = TRUE)
train_index <- createDataPartition(data$Class, p = 0.6, list = FALSE)
train <- data[train_index, ]
test <- data[-train_index, ]
```

Data Preprocessing:

```
rec_obj <- recipe(Class ~ ., data = train) %>%
  step_dummy(all_nominal(), -all_outcomes()) %>%
   step_center(all_predictors(), -all_outcomes()) %>%
  step_scale(all_predictors(), -all_outcomes()) %>%
  prep(data = train)
rec_obj
```

```
## Data Recipe
##
## Inputs:
##
##       role #variables
##    outcome          1
##  predictor         16
##
## Training data contained 290 data points and no missing data.
##
## Operations:
##
## Dummy variables from gender, NationalITy, PlaceofBirth, StageID, ... [trai
ned]
## Centering for raisedhands, ... [trained]
## Scaling for raisedhands, ... [trained]
```

Splitting the dependent and the independent variables in the training and testing datasets

```
x_train_tbl <- as.matrix(bake(rec_obj, new_data = train) %>% select(-Class))
x_test_tbl <- as.matrix(bake(rec_obj, new_data = test) %>% select(-Class))
```

```
y_train_vec <- ifelse(pull(train, Class) == "H", 2,ifelse(pull(train, Class)
== "M", 1,0))
y_test_vec <-  ifelse(pull(test, Class) == "H", 2,ifelse(pull(test, Class) ==
"M", 1,0))
str(x_train_tbl)
```

```
##  num [1:290, 1:60] -1.027 -0.145 -0.374 0.116 -1.125 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:60] "raisedhands" "VisITedResources" "AnnouncementsView" "D
iscussion" ...
```

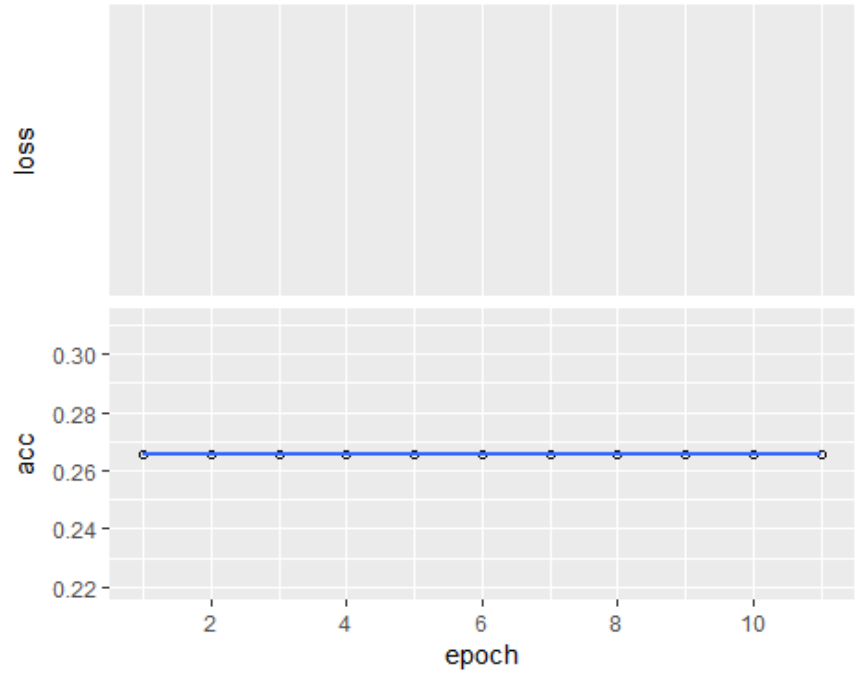
```
dim(x_train_tbl)
```

```
## [1] 290  60
```

```
library(keras)
use_python("/usr/local/bin/python3")
model_keras <- keras_model_sequential()

model_keras %>%
  layer_flatten(input_shape = ncol(rec_obj)) %>%
  layer_dense(units = 128, activation = 'relu') %>%
  layer_dense(units = 10, activation = 'softmax')

model_keras %>% compile(
  optimizer = 'adam',
  loss = 'sparse_categorical_crossentropy',
  metrics = c('accuracy')
)

set.seed(100)
fit <- model_keras %>% fit(x_train_tbl, y_train_vec, epochs = 11)
```

Plot training & validation accuracy values

```
plot(fit)
```



We can see from the aboev plot that the accuracy saturates around 84% as the loss continues to drop. (*Kindly run the rmd file for a proper plot*)

Predicting the Class:

```
class_pred <- model_keras %>% predict_classes(x_test_tbl)
class_pred[1:20]
```

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

NAIVE BAYES CLASSIFIER

Since, this is a classification problem let us try few other algorithms straight out of the box.

```
model_nb <- train(Class ~., data = train, method = "nb",
                  trControl = trainControl(method = "repeatedcv",
                                           number = 6, repeats = 5),
                  tuneGrid = expand.grid(fL = 1, usekernel = T, adjust = 1))

model_nb
```

```
## Naive Bayes
##
## 290 samples
##  16 predictor
##   3 classes: 'H', 'L', 'M'
##
## No pre-processing
## Resampling: Cross-Validated (6 fold, repeated 5 times)
## Summary of sample sizes: 241, 242, 241, 243, 242, 241, ...
```

```
## Resampling results:
##
##   Accuracy   Kappa
##   0.4379724  0
##
## Tuning parameter 'fL' was held constant at a value of 1
## Tuning
##  parameter 'usekernel' was held constant at a value of TRUE
##
## Tuning parameter 'adjust' was held constant at a value of 1
```

As we can see from the summry, that the model is having just 45% accuracy with a kappa score of 0.0284.

Let us now go ahead and predict the Class variable using the above model and evaluate it through confusion matrix.

```
predict_nb <- predict(model_nb, newdata = test, type = "raw")
head(predict_nb)
```

```
## [1] M M M M M M
## Levels: H L M
```

```
# Confusion Matrix
confusionMatrix(predict_nb, test$Class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  H  L  M
##          H  0  0  0
##          L  0  0  0
##          M 56 50 84
##
## Overall Statistics
##
##                Accuracy : 0.4421
##                  95% CI : (0.3703, 0.5158)
##     No Information Rate : 0.4421
##     P-Value [Acc > NIR] : 0.528
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: H Class: L Class: M
## Sensitivity            0.0000   0.0000   1.0000
## Specificity            1.0000   1.0000   0.0000
## Pos Pred Value            NaN      NaN   0.4421
```

```
## Neg Pred Value         0.7053   0.7368      NaN
## Prevalence             0.2947   0.2632   0.4421
## Detection Rate         0.0000   0.0000   0.4421
## Detection Prevalence   0.0000   0.0000   1.0000
## Balanced Accuracy      0.5000   0.5000   0.5000
```

DECISION TREE

Let us now implement the decision tree algorithm using the training dataset and see if it out performs Naive Bayes.

```
model_dt <- train(Class ~ ., data = train, method = "rpart",
                        metric = "Accuracy",
                        tuneLength = 10,
                        trControl = trainControl(method = "repeatedcv", number
= 10, repeats = 3))
```

Siimilar to the previous steps, we will go ahead and predict the Class variable using the decision tree model.

```
predict_dt <- predict(model_dt, newdata = test, type = "raw")
head(predict_dt)
```

```
## [1] M L L M M M
## Levels: H L M
```

```
# Confusion Matrix
confusionMatrix(predict_dt, test$Class)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  H  L  M
##          H 37  0 18
##          L  0 42 10
##          M 19  8 56
##
## Overall Statistics
##
##                Accuracy : 0.7105
##                  95% CI : (0.6405, 0.7739)
##     No Information Rate : 0.4421
##     P-Value [Acc > NIR] : 6.671e-14
##
##                   Kappa : 0.5543
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: H Class: L Class: M
```

```
## Sensitivity            0.6607   0.8400   0.6667
## Specificity            0.8657   0.9286   0.7453
## Pos Pred Value         0.6727   0.8077   0.6747
## Neg Pred Value         0.8593   0.9420   0.7383
## Prevalence             0.2947   0.2632   0.4421
## Detection Rate         0.1947   0.2211   0.2947
## Detection Prevalence   0.2895   0.2737   0.4368
## Balanced Accuracy      0.7632   0.8843   0.7060
```

As we can see, the Decision Tree algorithm performs really weell and we can see the same in the accuracy of 69.47%.

Let us plot above model so that we can have a better idea about the conditions used for partitioning the data.

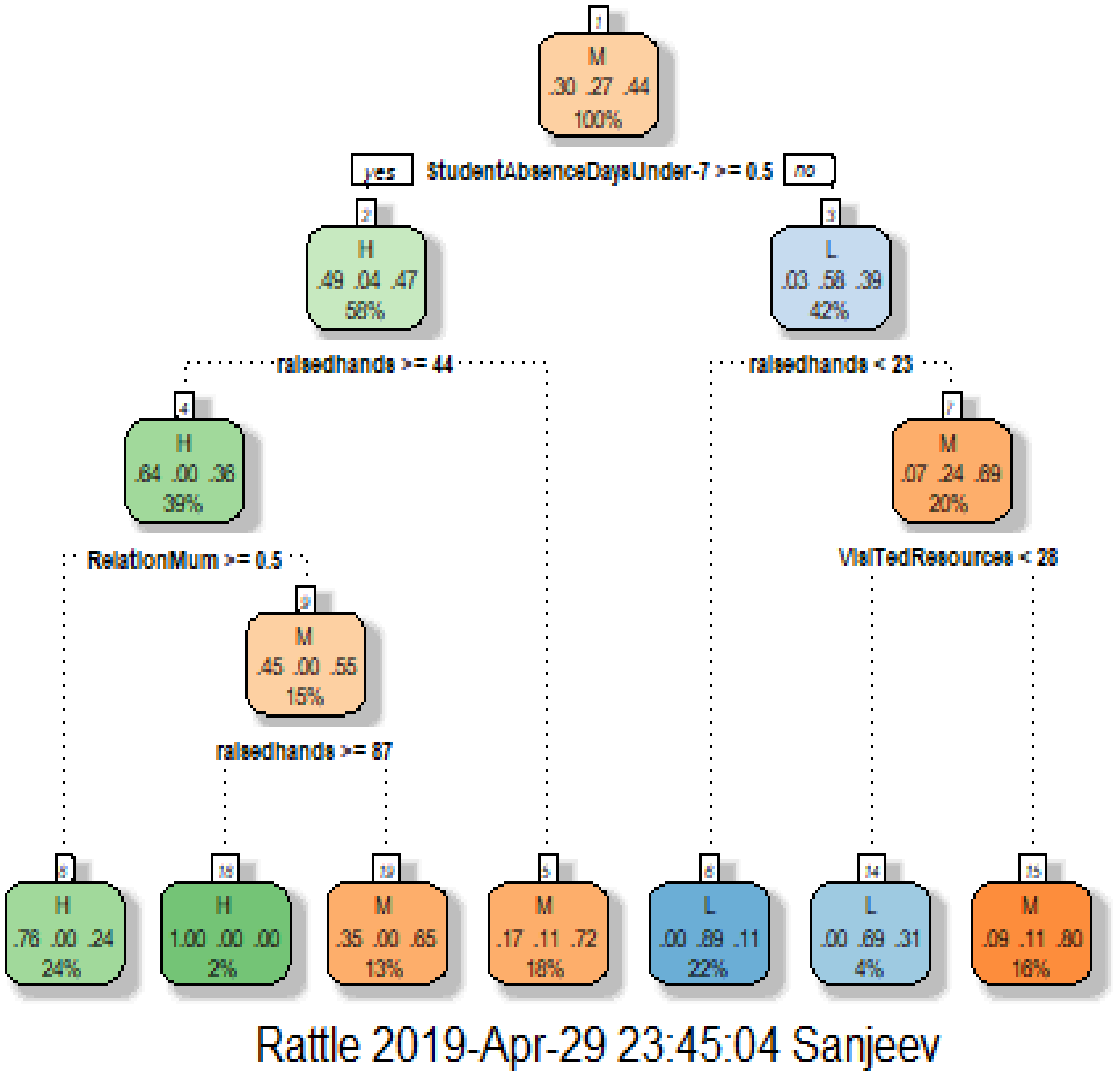
```
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
##
## Attaching package: 'rattle'
```

```
## The following object is masked from 'package:randomForest':
##
##     importance
```

```
fancyRpartPlot(model_dt$finalModel)
```

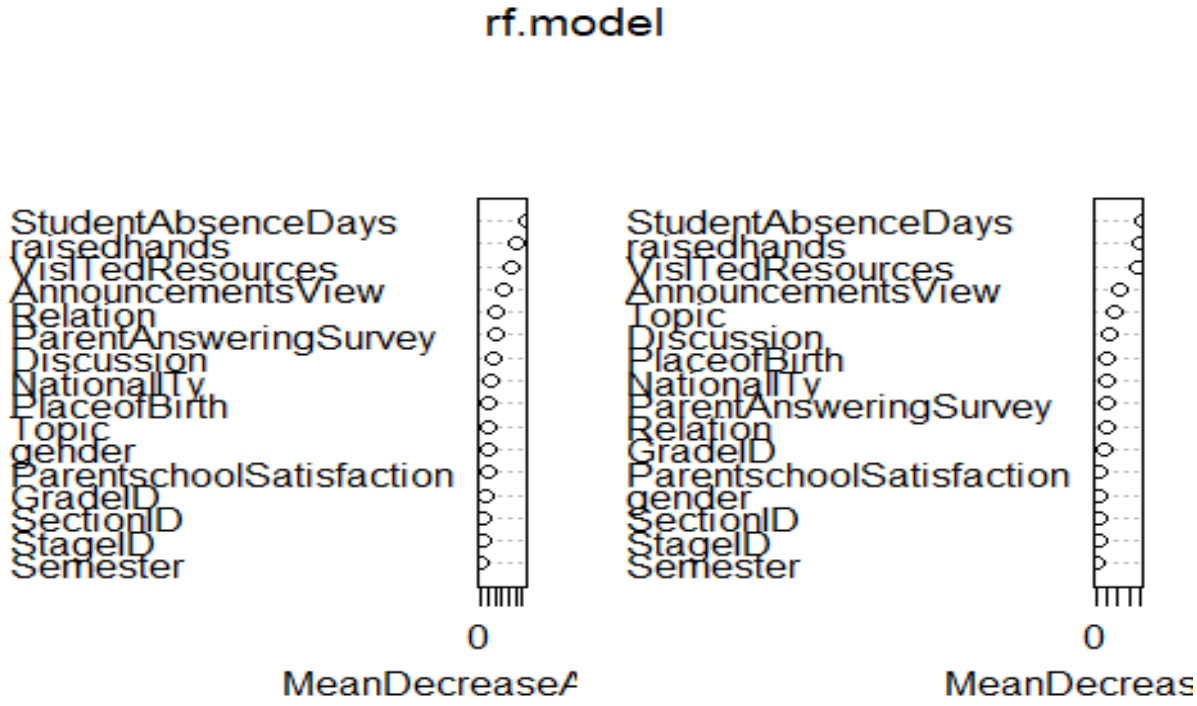


RANDOM FOREST

```
set.seed(10)

rf.model <- randomForest(Class ~ ., data = train, importance = TRUE,
                         ntree = 2000, nodesize = 20)

rf.predict <- predict(rf.model, test)
confusionMatrix(test$Class, rf.predict)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  H  L  M
##          H 40  0 16
##          L  0 42  8
```

```
##          M 11  5 68
##
## Overall Statistics
##
##                Accuracy : 0.7895
##                  95% CI : (0.7246, 0.8451)
##     No Information Rate : 0.4842
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6719
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: H Class: L Class: M
## Sensitivity            0.7843   0.8936   0.7391
## Specificity            0.8849   0.9441   0.8367
## Pos Pred Value         0.7143   0.8400   0.8095
## Neg Pred Value         0.9179   0.9643   0.7736
## Prevalence             0.2684   0.2474   0.4842
## Detection Rate         0.2105   0.2211   0.3579
## Detection Prevalence   0.2947   0.2632   0.4421
## Balanced Accuracy      0.8346   0.9188   0.7879
```

```
varImpPlot(rf.model)
```



With an accuracy of 76.32%, Random Forest still lags behind the Multilayer Perceptron model.

Conclusion

Based on the accuracy measures, the Multilayer Perceptron outperforms all the other models with an accuracy over 80%.