Data Dive in R

Chloe Israel

Data Description

My data set is called Student Performance Factors, which I found on Kaggle (kaggle.com). It provides a comprehensive overview of various factors that affect student performance during exams. It has 6,607 rows and 20 variables. Of the variables, 10 are character variables, 7 are numerical variables, and 3 are Boolean variables. The variables represent a different factor that influences a student's success during exams. These include but are not limited to study habits, student demographics, and school quality. The students' exam scores are also provided as a metric to measure their performance. I did not make any changes to the data set. The data set is saved in the file 'StudentPerformanceFactors.csv'.

Variables

Name	Data Type	Description
Hours_Studied	integer	Number of hours the student spends studying per week.
Attendance	integer	Percentage of classes the student attended.
Parental_Involvement	string	Level of parental involvement in the student's education (Low/Medium/High).
Access_to_Resources	string	Availability of educational resources to the student (Low/Medium/High).
Extracurricular_Activities	Boolean	Indicates whether the student participates in extracurriculars.
Sleep_Hours	integer	Average number of hours of sleep the student gets per night.
Previous_Scores	integer	Student's scores from past exams.
Motivation_Level	string	Student's motivation level (Low/Medium/High).
Internet_Access	Boolean	Indicates whether the student can access the internet.
Tutoring_Sessions	integer	Number of tutoring sessions the student attended per month.
Family_Income	string	Student's family income level (Low/Medium/High).
Teacher_Quality	string	Quality of the teachers at the student's school (Low/Medium/High).
School_Type	string	Type of school the student attends (Public/Private)
Peer_Influence	string	Influence of peers on the student's academic performance (Positive/Negative/Neutral).
Physical_Activity	integer	Average number of hours of physical activity the student gets per week.
Learning_Disabilities	Boolean	Indicates whether the student has a learning disability.
Parental_Education_Level	string	Highest level of education of the student's parents (High School/College/Postgraduate).
Distance_from_Home	string	Distance from the student's home to their school (Near/Moderate/Far).
Gender	string	Student's gender (Male, Female).
Exam_Score	integer	Student's final exam score.

Loading Required Packages

```
## List of packages used for this data dive.
packages = c("dplyr", "tibble")


## Function to install and load a package
package.check <- lapply(packages, FUN = function(x) {
    if (!require(x, character.only = TRUE)) {
        install.packages(x, dependencies = TRUE)
        library(x, character.only = TRUE)
    }
})
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
## Loading required package: tibble
```

```
# Load the dplyr and tibble packages to help with data manipulation
# and
library(dplyr)
library(tibble)
```

Loading the Data

```
# Load our data into the spf data frame
spf <- read.csv("data/StudentPerformanceFactors.csv")
```

```
# spf column names
colnames(spf)
```

```
##  [1] "Hours_Studied"              "Attendance"
##  [3] "Parental_Involvement"       "Access_to_Resources"
##  [5] "Extracurricular_Activities" "Sleep_Hours"
##  [7] "Previous_Scores"            "Motivation_Level"
##  [9] "Internet_Access"            "Tutoring_Sessions"
```

```
## [11] "Family_Income"              "Teacher_Quality"
## [13] "School_Type"                "Peer_Influence"
## [15] "Physical_Activity"          "Learning_Disabilities"
## [17] "Parental_Education_Level"   "Distance_from_Home"
## [19] "Gender"                     "Exam_Score"
```

```
# spf dimensions
dim(spf)
```

```
## [1] 6607   20
```

```
# First 10 rows of spf, formatted as a tibble to make it more
# readable
as_tibble(head(spf, 10))
```

```
## # A tibble: 10 x 20
##    Hours_Studied Attendance Parental_Involvement Access_to_Resources
##            <int>      <int> <chr>                <chr>
##  1            23         84 Low                  High
##  2            19         64 Low                  Medium
##  3            24         98 Medium               Medium
##  4            29         89 Low                  Medium
##  5            19         92 Medium               Medium
##  6            19         88 Medium               Medium
##  7            29         84 Medium               Low
##  8            25         78 Low                  High
##  9            17         94 Medium               High
## 10            23         98 Medium               Medium
## # i 16 more variables: Extracurricular_Activities <chr>, Sleep_Hours <int>,
## #   Previous_Scores <int>, Motivation_Level <chr>, Internet_Access <chr>,
## #   Tutoring_Sessions <int>, Family_Income <chr>, Teacher_Quality <chr>,
## #   School_Type <chr>, Peer_Influence <chr>, Physical_Activity <int>,
## #   Learning_Disabilities <chr>, Parental_Education_Level <chr>,
## #   Distance_from_Home <chr>, Gender <chr>, Exam_Score <int>
```

Data Check

Strange or Unexpected Values

Currently, the 3 Boolean variables "Extracurricular_Activities", "Internet_Access", and "Learning_Disabilities" hold the values Yes/No instead of the standard Boolean values TRUE/FALSE. To make it easier to work with these variables later and to keep with Boolean conventions, I will convert the values to TRUE/FALSE.

```
# Copy spf into the dat data frame for modifications.
dat <- spf

# Use a for loop to iterate through each column, checking for 'Yes'
# or 'No'.  If 'Yes', change it to True. If 'No', change it to False.

# 'Extracurricular_Activities' column
```

```
for (i in 1:length(dat$Extracurricular_Activities)) {

    if (dat$Extracurricular_Activities[i] == "Yes") {
        dat$Extracurricular_Activities[i] <- TRUE
    } else {
        dat$Extracurricular_Activities[i] <- FALSE
    }
}

# 'Internet_Access' column
for (i in 1:length(dat$Internet_Access)) {

    if (dat$Internet_Access[i] == "Yes") {
        dat$Internet_Access[i] <- TRUE
    } else {
        dat$Internet_Access[i] <- FALSE
    }
}

# 'Learning_Disabilities' column
for (i in 1:length(dat$Learning_Disabilities)) {

    if (dat$Learning_Disabilities[i] == "Yes") {
        dat$Learning_Disabilities[i] <- TRUE
    } else {
        dat$Learning_Disabilities[i] <- FALSE
    }
}

# Now, each column should have the standard TRUE/FALSE
head(dat[c("Extracurricular_Activities", "Internet_Access", "Learning_Disabilities")],
    5)
```

```
##   Extracurricular_Activities Internet_Access Learning_Disabilities
## 1                      FALSE            TRUE                 FALSE
## 2                      FALSE            TRUE                 FALSE
## 3                       TRUE            TRUE                 FALSE
## 4                       TRUE            TRUE                 FALSE
## 5                       TRUE            TRUE                 FALSE
```

Create Useful Categories

Most of the variables in spf already provide information that can be easily used. The only addition I will make is a new variable "Average_Score", which will take the average of the two variables "Previous_Scores" and "Exam_Score". This is to make it easier to compare the other variable related to student performance to a metric.

```
# Use basic arithmetic to take the average of the two columns
# 'Previous_Scores' and 'Exam_Score'. Use integer division so the
# scores are consistent with the other columns. Store it in the new
# column 'Average_Score'.
dat$Average_Score <- (dat$Previous_Scores + dat$Exam_Score)%/%2
```

```r
# Now dat has a new column 'Average_Score'
head(dat[c("Average_Score")], 5)
```

```
##   Average_Score
## 1            70
## 2            60
## 3            82
## 4            84
## 5            67
```

Data Moves

For my analysis, I would like to focus on factors that give some insight into a student's level of effort and their socioeconomic status. I think these two categories impact a student's school performance the most.

To start, the variables I will be exploring are: "Attendance", "Internet_Access", "Hours_Studied", "Family_Income", and "School_Type".

Calculating

In the section above, I used calculating to create a new column "Average_Score". I did this so I could have a single metric to use to measure student performance instead of two. This will make it easier for me to analyze my data and draw conclusions later. "Average_Score" will also be important for my later exploration of other variables.

Filtering

I will filter "Attendance" based on whether students have attended an above average percentage of classes or below average. This is to help gauge a student's effort in class, with higher attendance indicating a higher effort.

```r
# First, determine the average percentage of attendance
attAvg <- mean(dat$Attendance)
print(paste("The average pecentage of classes students attend is ", round(attAvg),
    "%", sep = ""))
```

```
## [1] "The average pecentage of classes students attend is 80%"
```

```r
# Then, use the subset function filter the Attendance column in dat
# based on attAvg

# Students that are at or above the threshold go in the attSat data
# frame
attSat <- subset(dat, dat$Attendance >= attAvg)

# Students that are below the threshold go in the attUnsat data frame
attUnsat <- subset(dat, dat$Attendance < attAvg)

# Check by displaying the first 3 rows of the new, filtered data
```

```r
# frames
head(attSat["Attendance"], 3)
```

```
##   Attendance
## 1         84
## 3         98
## 4         89
```

```r
head(attUnsat["Attendance"], 3)
```

```
##    Attendance
## 2          64
## 8          78
## 15         78
```

I will filter "Internet_Access" based on its TRUE or FALSE value. This is to help gauge a student's socioeconomic status, as students who do not have access to the internet at home tend to be of a lower economic standing than those who do.

```r
# Use the subset function filter the Internet_Access column of dat
# based on a row's TRUE/FALSE value

# Students who do have internet access go in the hasInt data frame
hasWeb <- subset(dat, dat$Internet_Access == TRUE)

# Students who do not have internet access go in the noInt data frame
noWeb <- subset(dat, dat$Internet_Access == FALSE)

# Check by displaying the first 3 rows of the new, filtered data
# frames
head(hasWeb["Internet_Access"], 3)
```

```
##   Internet_Access
## 1            TRUE
## 2            TRUE
## 3            TRUE
```

```r
head(noWeb["Internet_Access"], 3)
```

```
##    Internet_Access
## 11           FALSE
## 57           FALSE
## 65           FALSE
```

I will filter "Hours_Studied" based on whether students have studied an above average number of hours or below average. This is to help gauge a student's effort in class, with higher hours indicating a higher effort.

```r
# First, determine the average hours students spend studying
studyAvg <- mean(dat$Hours_Studied)
print(paste("The average number of hours students spend studying is", round(studyAvg),
    "hours"))
```

```
## [1] "The average number of hours students spend studying is 20 hours"
```

```r
# Then, use the subset function filter the Hours_Studied column in
# dat based on studyAvg

# Students that are at or above the threshold go in the studySat data
# frame
studySat <- subset(dat, dat$Hours_Studied >= studyAvg)

# Students that are below the threshold go in the studyUnsat data
# frame
studyUnsat <- subset(dat, dat$Hours_Studied < studyAvg)

# Check by displaying the first 3 rows of the new, filtered data
# frames
head(studySat["Hours_Studied"], 3)
```

```
##   Hours_Studied
## 1            23
## 3            24
## 4            29
```

```r
head(studyUnsat["Hours_Studied"], 3)
```

```
##   Hours_Studied
## 2            19
## 5            19
## 6            19
```

I will filter "Family_Income" based on its value of "Low","Medium, or"High". This is to help gauge a student's socioeconomic status, with each income category corresponding to an economic standing.

```r
# Use the subset function filter the Family_Income column of dat
# based on a row's 'Low', 'Medium', or 'High' value

# Students who's family income is 'Low' go into the lowInc data frame
lowInc <- subset(dat, dat$Family_Income == "Low")

# Students who's family income is 'Medium' go into the medInc data
# frame
medInc <- subset(dat, dat$Family_Income == "Medium")

# Students who's family income is 'Medium' go into the highInc data
# frame
highInc <- subset(dat, dat$Family_Income == "High")

# Check by displaying the first 3 rows of the new, filtered data
# frames
head(lowInc["Family_Income"], 3)
```

```
##    Family_Income
## 1            Low
## 7            Low
## 12           Low
```

```r
head(medInc["Family_Income"], 3)
```

```
##   Family_Income
## 2        Medium
## 3        Medium
## 4        Medium
```

```r
head(highInc["Family_Income"], 3)
```

```
##    Family_Income
## 8           High
## 10          High
## 17          High
```

```r
# Since this variable is categorical, I'll also check by summing the
# three data frames to ensure they add to the total number of
# observations (6607).  This is to catch any rows that might not fit
# into our defined values for any reason.
sum(nrow(lowInc), nrow(medInc), nrow(highInc))
```

```
## [1] 6607
```

I will filter "School_Type" based on its value of "Public" or "Private". This is to help gauge a student's socioeconomic status, as students who go to a private school tend to be of a higher economic standing than those at a public school.

```r
# Use the subset function filter the School_Type column of dat based
# on a row's 'Public' or 'Private' value

# Students who go to public school go in the pub data frame
pub <- subset(dat, dat$School_Type == "Public")

# Students who go to private school go in the priv data frame
priv <- subset(dat, dat$School_Type == "Private")

# Check by displaying the first 3 rows of the new, filtered data
# frames
head(pub["School_Type"], 3)
```

```
##   School_Type
## 1      Public
## 2      Public
## 3      Public
```

```r
head(priv["School_Type"], 3)
```

```
##    School_Type
## 7      Private
## 9      Private
## 11     Private
```

```r
# Since this variable is categorical, I'll also check by summing the
# two data frames to ensure they add to the total number of
# observations (6607).  This is to catch any rows that might not fit
# into our defined values for any reason.
sum(nrow(pub), nrow(priv))
```

```
## [1] 6607
```

Summarizing

I want summarize the average exam scores of students who have average or above attendance and those who have below average attendance. This is to help understand how much class attendance impacts student performance.

```r
# Calculate the mean of the Average_Score column for the attSat dat
# frame and the attUnsat data frame.
attSatScore <- mean(attSat$Average_Score)
attUnsatScore <- mean(attUnsat$Average_Score)

# Show results
print(paste("Students who attended ", round(attAvg), "% or more classes ",
    "had an average exam score of ", round(attSatScore), sep = ""))
```

```
## [1] "Students who attended 80% or more classes had an average exam score of 72"
```

```r
print(paste("Students who attended less than ", round(attAvg), "% of classes ",
    "had an average exam score of ", round(attUnsatScore), sep = ""))
```

```
## [1] "Students who attended less than 80% of classes had an average exam score of 70"
```

Next, I'll summarize the average exam scores of students who have internet access and those who don't. This is to help understand how much at home access to the internet impacts student performance, which could have further implications about a student's socioeconomic status.

```r
# Calculate the mean of the Average_Score column for the hasWeb dat
# frame and the noWeb data frame.
hasWebScore <- mean(hasWeb$Average_Score)
noWebScore <- mean(noWeb$Average_Score)

# Show results
print(paste("Students who could access the internet at home had ", "an average exam score of ",
    round(hasWebScore), sep = ""))
```

```
## [1] "Students who could access the internet at home had an average exam score of 71"
```

```r
print(paste("Students who could not access the internet at home had ",
    "an average exam score of ", round(noWebScore), sep = ""))
```

```
## [1] "Students who could not access the internet at home had an average exam score of 70"
```

Next, I'll summarize the average exam scores of students who study for an average or above number of hours and those who study a below average number of hours. This is to help understand how much studying impacts student performance.

```
# Calculate the mean of the Average_Score column for the studySat dat
# frame and the studyUnsat data frame.
studySatScore <- mean(studySat$Average_Score)
studyUnsatScore <- mean(studyUnsat$Average_Score)

# Show results
print(paste("Students who studied for ", round(studyAvg), " or more hours had an average ",
    "exam score of ", round(studySatScore), sep = ""))
```

```
## [1] "Students who studied for 20 or more hours had an average exam score of 72"
```

```
print(paste("Students who studied for less than ", round(studyAvg), " hours had an average ",
    "exam score of ", round(studyUnsatScore), sep = ""))
```

```
## [1] "Students who studied for less than 20 hours had an average exam score of 70"
```

Next, I'll summarize the average exam scores of students based on their family's income level. This is to help understand directly how much a student's socioeconomic status impacts their performance.

```
# Calculate the mean of the Average_Score column for the lowInc dat
# frame, the medInc data frame, and the highInc data frame.
lowIncScore <- mean(lowInc$Average_Score)
medIncScore <- mean(medInc$Average_Score)
highIncScore <- mean(highInc$Average_Score)

# Show results (Rounded to two decimal points due to the small score
# difference).
print(paste("Students from low-income families had an average exam score of",
    round(lowIncScore, 2)))
```

```
## [1] "Students from low-income families had an average exam score of 70.77"
```

```
print(paste("Students from medium-income families had an average exam score of",
    round(medIncScore, 2)))
```

```
## [1] "Students from medium-income families had an average exam score of 71"
```

```
print(paste("Students from high-income families had an average exam score of",
    round(highIncScore, 2)))
```

```
## [1] "Students from high-income families had an average exam score of 70.97"
```

Finally, I'll summarize the average exam scores of students based on their school type. This is also to help understand directly how much a student's socioeconomic status impacts their performance.

```r
# Calculate the mean of the Average_Score column for the priv dat
# frame and the pub data frame.
privScore <- mean(priv$Average_Score)
pubScore <- mean(pub$Average_Score)

# Show results (Rounded to two decimal points due to the small score
# difference).
print(paste("Students who attended private school had an average", "exam score of",
    round(privScore, 2)))
```

```
## [1] "Students who attended private school had an average exam score of 70.78"
```

```r
print(paste("Students who attended public school had an average", "exam score of",
    round(pubScore, 2)))
```

```
## [1] "Students who attended public school had an average exam score of 70.95"
```

Grouping

Because the above results from summarizing were so close together, I think I need to explore other factors to help get a better idea of what impacts student performance and how much. I will explore one more different variable, "Tutoring Sessions". This is to help understand how much effort outside of class impacts student performance. Instead of filtering and summarizing this variable, I will use grouping. This is because "Tutoring Sessions" has a narrow range of values, so it will be easier to group and display than some of the other numerical variables.

```r
# Use the pipe operator to pass dat through different functions.
# First, group dat by the unique number of Tutoring_Sessions Then,
# use summarise() to calculate and display Average_Score for each
# group Round the scores to two decimal places in case any have close
# values Then, arrange the results in descending order by
# Average_Score to better visualize how tutoring impacts performance.
dat %>%
    group_by(Tutoring_Sessions) %>%
    summarise(Average_Score = round(mean(Average_Score), 2)) %>%
    arrange(desc(Average_Score))
```

```
## # A tibble: 9 x 2
##   Tutoring_Sessions Average_Score
##               <int>         <dbl>
## 1                 6          76.9
## 2                 7          74.9
## 3                 8          73
## 4                 5          72.0
## 5                 2          71.2
## 6                 4          71.0
## 7                 0          70.8
## 8                 3          70.8
## 9                 1          70.6
```