FinalProject-Fall 2023

Ranya Muttaleb, Silvia Morales, Smriti Ale, Nista Shrestha, and Cardin Le

2023-11-29

```r
df <- read_excel("/Users/nistashrestha/Desktop/Fall 2023/ETM540/Project/Project Data Updated.xlsx")
dim(df)
```

```
## [1] 433  12
```

```r
variable_names <- names(df)
print(variable_names)
```

```
##  [1] "StudyID"                         "Award Amount"
##  [3] "Awarded_binary"                  "Level_of_Need"
##  [5] "CummGPA_Award"                   "Class_Standing"
##  [7] "Major"                           "School"
##  [9] "Award_Academic_Year"             "Hardship_AwardACADEMIC_PERIOD"
## [11] "STUDENT_LEVEL_ABBREV"            "Credits_needed"
```

```r
# Choose a year
df<- subset(df, Award_Academic_Year == '2021')
```

```r
# Count occurrences of 'Y' and 'N' for Awarded_binary
count_Y_N <- table(df$Awarded_binary)

# Print the result
print(count_Y_N)
```

```
##
##  N  Y
## 31 40
```

```r
find_missing_values <- function(data) {
  missing_values <- sapply(data, function(x) sum(is.na(x)))
  return(missing_values)
}

missing_values_result <- find_missing_values(df)

print(missing_values_result)
```

```
##                        StudyID                   Award Amount
##                              0                              0
```

```
##                    Awarded_binary                    Level_of_Need
##                                 0                                0
##                      CummGPA_Award                   Class_Standing
##                                 0                                0
##                             Major                           School
##                                10                               10
##                Award_Academic_Year Hardship_AwardACADEMIC_PERIOD
##                                 0                                0
##               STUDENT_LEVEL_ABBREV                   Credits_needed
##                                 0                                0
```

We have 19 missing values for 'Major' and 'School'. They belong to the same rows and are all for non-awarded group of students. Since we already have more awarded students than non-awarded students (364 versus 75), its better to not remove these rows of data.

Instead, we replace the missing values with mode of the each columns.

```r
# Mode function to calculate mode
Mode <- function(x) {
  ux <- unique(x)
  ux[which.max(tabulate(match(x, ux)))]
}

df_filled <- df %>%
  mutate(Major = ifelse(is.na(Major), Mode(Major), Major),
         School = ifelse(is.na(School), Mode(School), School))

missing_values_result <- find_missing_values(df_filled)

print(missing_values_result)
```

```
##                           StudyID                     Award Amount
##                                 0                                0
##                    Awarded_binary                    Level_of_Need
##                                 0                                0
##                      CummGPA_Award                   Class_Standing
##                                 0                                0
##                             Major                           School
##                                10                                0
##                Award_Academic_Year Hardship_AwardACADEMIC_PERIOD
##                                 0                                0
##               STUDENT_LEVEL_ABBREV                   Credits_needed
##                                 0                                0
```

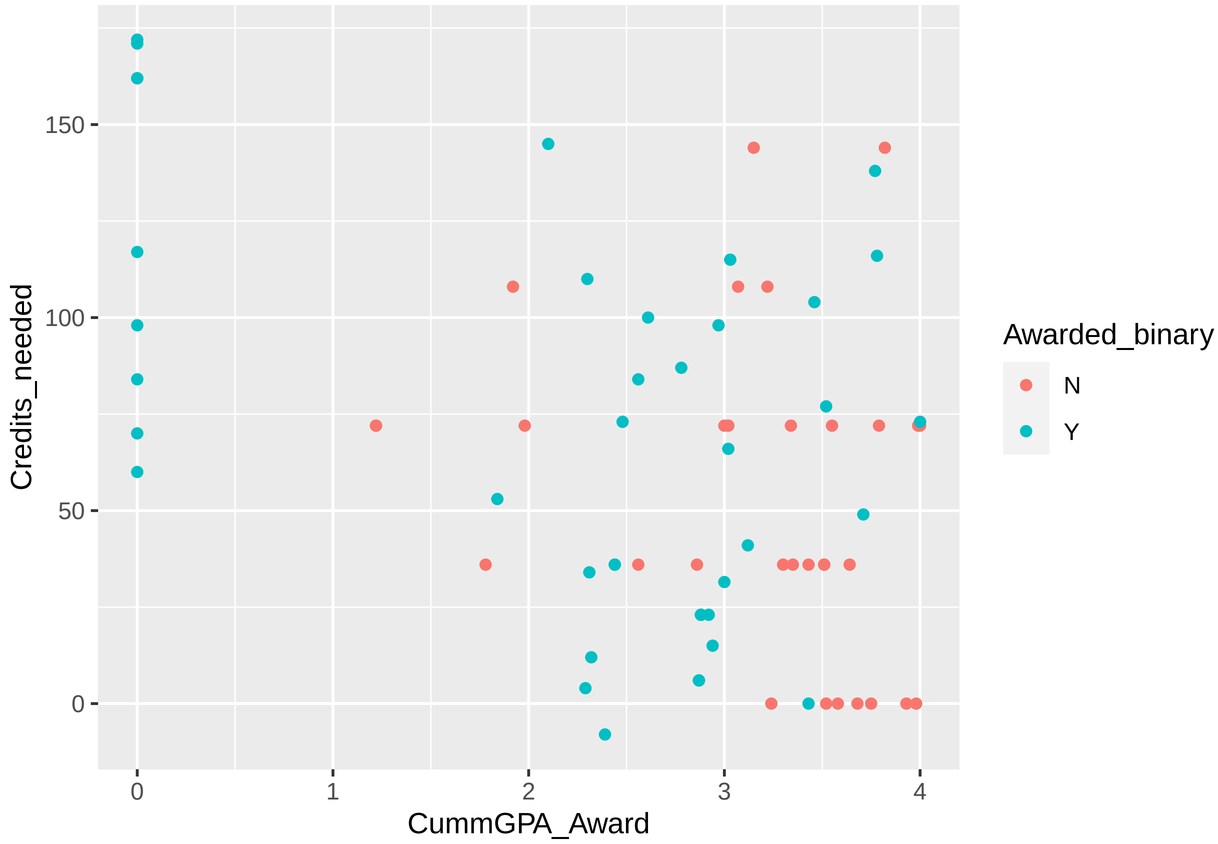
0.1 KMean Clustering

Variables selection

```r
data <- df_filled[c("Awarded_binary", "CummGPA_Award", "Credits_needed")]
dim(data) #dimensions (shape) of the dataframe
```

```
## [1] 71  3
```

```r
data %>% ggplot(aes(CummGPA_Award, Credits_needed, color= Awarded_binary))+
geom_point()
```



Performing kmean analysis separately for **awarded students** and **non-awarded students**.

```r
# Create new dataframes
df_awarded <- subset(data, Awarded_binary == 'Y')
df_non_awarded <- subset(data, Awarded_binary == 'N')
print(dim(df_awarded))
```

```
## [1] 40  3
```

```r
print(dim(df_non_awarded))
```

```
## [1] 31  3
```

Now, removing the numeric column 'Awarded_binary' from the data.

```r
#removing categorical variable Awarded_binary
df_awarded = df_awarded[c("CummGPA_Award", "Credits_needed")]
df_non_awarded = df_non_awarded[c("CummGPA_Award", "Credits_needed")]
print(dim(df_awarded))
```

```
## [1] 40  2
```

```r
print(dim(df_non_awarded))
```

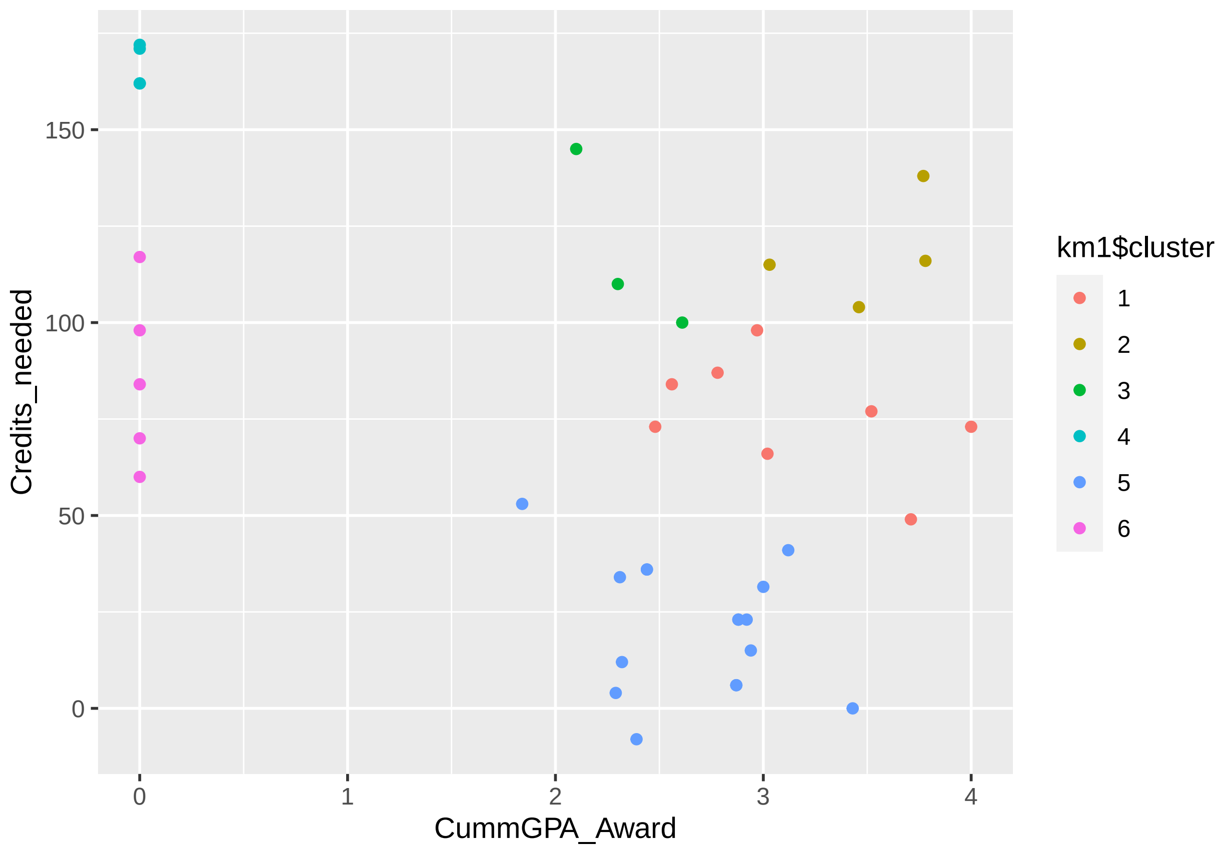
```
## [1] 31  2
```

Kmean for Awarded

```r
scal1 <- scale(df_awarded)
km1 <- kmeans(scal1, centers = 6 , nstart = 5)
km1
```

```
## K-means clustering with 6 clusters of sizes 8, 4, 3, 4, 16, 5
##
## Cluster means:
##   CummGPA_Award Credits_needed
## 1    0.69386842     0.08184323
## 2    0.98545814     0.90513270
## 3    0.08511095     0.90675176
## 4   -1.70791003     1.84742272
## 5    0.35144234    -0.98491164
## 6   -1.70791003     0.27467268
##
## Clustering vector:
##  [1] 5 5 5 5 5 5 1 2 2 5 1 1 3 3 6 5 5 5 1 1 2 5 5 5 1 5 1 5 2 5 3 6 1 4 4 6 4 4
## [39] 6 6
##
## Within cluster sum of squares by cluster:
## [1] 1.84029298 0.44964927 0.49927387 0.03425566 3.07234769 0.77034656
##  (between_SS / total_SS =  91.5 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
km1$cluster <- as.factor(km1$cluster)
df_awarded %>% ggplot(aes(CummGPA_Award, Credits_needed, color= km1$cluster))+
geom_point()
```



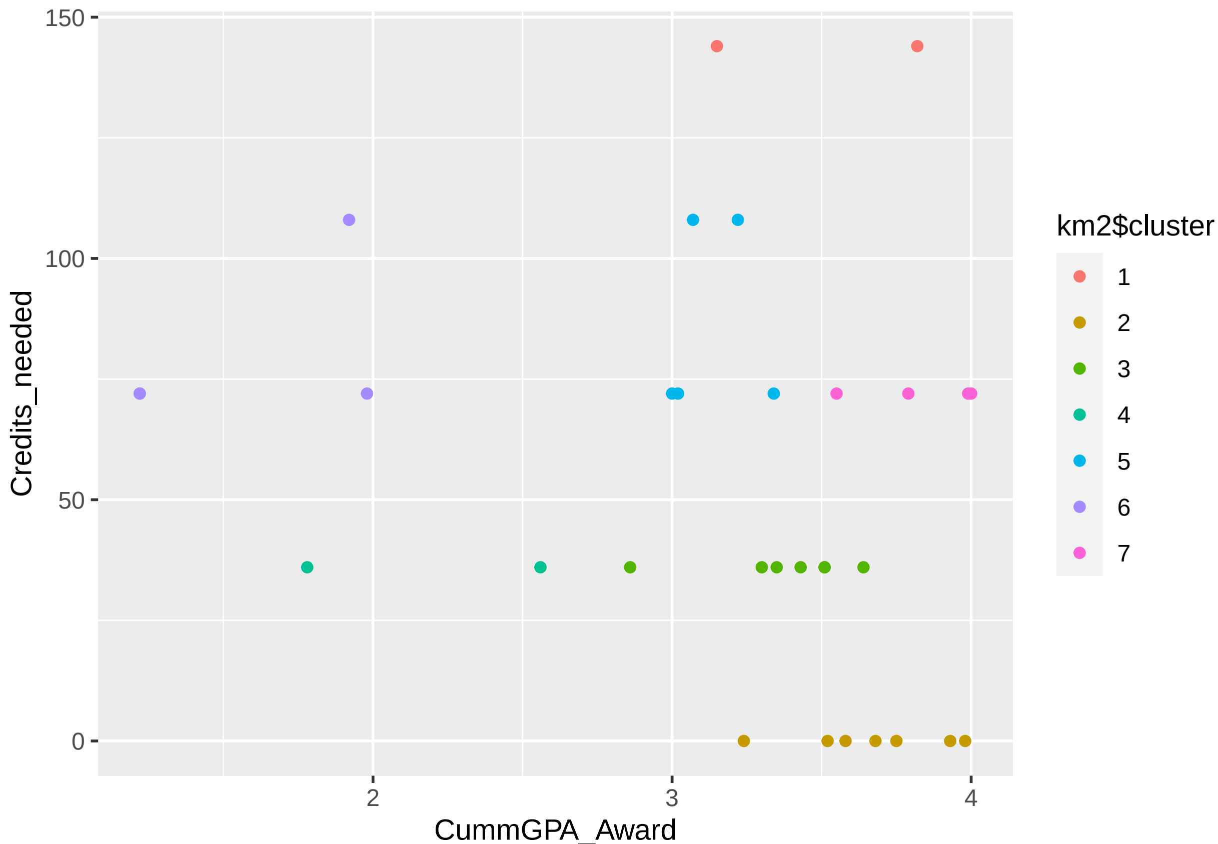
Kmean for Non-Awarded

```
scal2 <- scale(df_non_awarded)
km2 <- kmeans(scal2, centers = 7  , nstart = 5)
km2
```

```
## K-means clustering with 7 clusters of sizes 2, 7, 7, 2, 5, 4, 4
##
## Cluster means:
##   CummGPA_Award Credits_needed
## 1    0.42071899      2.1860950
## 2    0.65718236     -1.2892355
## 3    0.27442454     -0.4204029
## 4   -1.27316839     -0.4204029
## 5   -0.03656619      0.7959628
## 6   -2.02672285      0.6656379
## 7    0.86834322      0.4484297
##
## Clustering vector:
##  [1] 5 3 2 4 2 7 3 5 2 4 3 3 6 5 2 7 2 3 5 3 7 2 6 6 2 3 7 5 1 6 1
##
## Within cluster sum of squares by cluster:
## [1] 0.3724236 0.6399575 0.6323248 0.5047505 1.0465504 1.4533653 0.2231306
##  (between_SS / total_SS =  91.9 %)
##
## Available components:
```

```
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
km2$cluster <- as.factor(km2$cluster)
df_non_awarded %>% ggplot(aes(CummGPA_Award, Credits_needed, color= km2$cluster))+
geom_point()
```



PoolA: Awarded students with GPA between 2-3.5 and Credit-needed under 50 PoolB: Awarded students with GPA between 3-4 and Credit-needed between 100 - 150 PoolC: Non-awarded students with GPA below 2 and Credit-needed between 70-105 PoolD: Non-awarded students with GPA greater than 3.25 and Credit-needed below 5 Average award amount per student for 2021: $35.64 Table summarized below.

Type	PoolA	PoolB	PoolC	PoolD
Average Cumm. GPA	2.75	3.50	1.00	3.75
Average Credit Need	25	125	87	2
No. of students	16	4	4	7
Average Award Amount	570.2	142.55	142.55	249.46

Maximum budget for year 2021 is $262,000.

Objective Function: How to maximize impact of Hardship fund distributed to different student pool.

Maximum Award Amount (X) = 570.2*$Pool_A$ + 142.55*$Pool_B$ + 142.55*$Pool_C$ + 249.46*$Pool_D$

Defining Variables:

- $Pool_A$ = Awarded students with GPA between 2-3.5 and Credit-needed under 50
- $Pool_B$ = Awarded students with GPA between 3-4 and Credit-needed between 100 - 150
- $Pool_C$ = Non-awarded students with GPA below 2 and Credit-needed between 70-105
- $Pool_D$ = Non-awarded students with GPA greater than 3.25 and Credit-needed below 5
- Y_A = 1 if you choose to award PoolA; 0 otherwise
- Y_B = 1 if you choose to award PoolB; 0 otherwise
- Y_C = 1 if you choose to award PoolC; 0 otherwise
- Y_D = 1 if you choose to award PoolD; 0 otherwise
- X = Maximum Award Amount

Constraints:

1. No more than $ 262,000 budget for the year 2021
2. Budget cannot go negative
3. No more than 16 students for PoolA
4. No more than 4 students for PoolB
5. No more than 4 students for PoolC
6. No more than 7 students for PoolD
7. No of students cannot go negative

Optimization model using LATEX:

$$
\begin{aligned}
\text{Max } & 570.2Pool_A + 142.55Pool_B + 142.55Pool_C + 249.46Pool_D \\
\text{s.t.: } & Pool_A \leq 16 \\
& Pool_B \leq 4 \\
& Pool_C \leq 4 \\
& Pool_D \leq 7 \\
& 0 \leq X \leq 262000 \\
& Pool_A,\ Pool_B,\ Pool_C,\ Pool_D \geq 0 \\
& Y_A,\ Y_B,\ Y_C\ Y_D\ \in \{0,1\}
\end{aligned}
$$

Implementing model

```
model1 <- MIPModel() |>
 add_variable(Pool_A, type="integer", lb=0, ub=16)  |>
 add_variable(Pool_B, type="integer", lb=0, ub=4) |>
 add_variable(Pool_C, type="integer", lb=0, ub=4)  |>
 add_variable(Pool_D, type="integer", lb=0, ub=7)  |>
 add_variable(Y_A, type="binary")                 |>
 add_variable(Y_B, type = "binary")               |>
 add_variable(Y_C, type = "binary")               |>
 add_variable(Y_D, type = "binary")               |>
 add_variable(X, type = "continuous")               |>

 set_objective(570.2*Pool_A + 142.55*Pool_B +
                142.55*Pool_C + 249.46*Pool_D, "max")  |>

add_constraint(X <= 226000)

model1_res <- solve_model(model1,
                           with_ROI(solver = "glpk"))
model1_res
```

```
## Status: success
## Objective value: 12009.82
```

```r
model1_summary <-
  cbind(model1_res$objective_value,
        t(as.matrix(model1_res$solution)))
colnames(model1_summary)<-
  c("Obj.Func.Val.", "$Pool_A$","$Pool_B$","$Pool_C$","$Pool_D$",
    "$Y_A$", "$Y_B$", "$Y_C$","$Y_D$", "X")
rownames(model1_summary)<-list("Optimal Award Amount")
kbl (model1_summary, booktabs=T, escape=F,
     caption = "Model 1 Optimal Result") |>
  kable_styling(latex_options = "HOLD_position")
```

Table 2: Model 1 Optimal Result

	Obj.Func.Val.	$Pool_A$	$Pool_B$	$Pool_C$	$Pool_D$	Y_A	Y_B	Y_C	Y_D	X
Optimal Award Amount	12009.82	16	4	4	7	0	0	0	0	226000

```r
model2 <- MIPModel() |>
 add_variable(Pool_A, type="integer", lb=0, ub=300)  |>
 add_variable(Pool_B, type="integer", lb=0, ub=150) |>
 add_variable(Pool_C, type="integer", lb=0, ub=100)  |>
 add_variable(Pool_D, type="integer", lb=0, ub=50)  |>
 add_variable(Y_A, type="binary")                  |>
 add_variable(Y_B, type = "binary")                |>
 add_variable(Y_C, type = "binary")                |>
 add_variable(Y_D, type = "binary")                |>
 add_variable(X, type = "continuous")                |>

 set_objective(570.2*Pool_A + 142.55*Pool_B +
                142.55*Pool_C + 249.46*Pool_D, "max")  |>

add_constraint(X <= 226000)

model2_res <- solve_model(model2,
                          with_ROI(solver = "glpk"))
model2_res
```

```
## Status: success
## Objective value: 219170.5
```

```r
model2_summary <-
  cbind(model2_res$objective_value,
        t(as.matrix(model2_res$solution)))
colnames(model2_summary)<-
  c("Obj.Func.Val.", "$Pool_A$","$Pool_B$","$Pool_C$","$Pool_D$",
    "$Y_A$", "$Y_B$", "$Y_C$","$Y_D$", "X")
rownames(model2_summary)<-list("Optimal Award Amount")
kbl (model2_summary, booktabs=T, escape=F,
```

```r
     caption = "Model 2 Optimal Result") |>
  kable_styling(latex_options = "HOLD_position")
```

Table 3: Model 2 Optimal Result

	Obj.Func.Val.	$Pool_A$	$Pool_B$	$Pool_C$	$Pool_D$	Y_A	Y_B	Y_C	Y_D	X
Optimal Award Amount	219170.5	300	150	100	50	0	0	0	0	226000

```r
model3 <- MIPModel() |>
 add_variable(Pool_A, type="integer", lb=0)  |>
 add_variable(Pool_B, type="integer", lb=0) |>
 add_variable(Pool_C, type="integer", lb=0)  |>
 add_variable(Pool_D, type="integer", lb=0)  |>
 add_variable(Y_A, type="binary")            |>
 add_variable(Y_B, type = "binary")          |>
 add_variable(Y_C, type = "binary")          |>
 add_variable(Y_D, type = "binary")          |>
 add_variable(X, type = "continuous")          |>

 set_objective(570.2*Pool_A + 142.55*Pool_B +
                142.55*Pool_C + 249.46*Pool_D, "max")  |>

add_constraint(X <= 226000)

model3_res <- solve_model(model3,
                           with_ROI(solver = "glpk"))
model3_res
```

```
## Status: error
## Objective value: 0
```

```r
model3_summary <-
  cbind(model3_res$objective_value,
        t(as.matrix(model3_res$solution)))
colnames(model3_summary)<-
  c("Obj.Func.Val.", "$Pool_A$","$Pool_B$","$Pool_C$","$Pool_D$",
    "$Y_A$", "$Y_B$", "$Y_C$","$Y_D$", "X")
rownames(model3_summary)<-list("Optimal Award Amount")
kbl (model3_summary, booktabs=T, escape=F,
     caption = "Model 3 Optimal Result") |>
  kable_styling(latex_options = "HOLD_position")
```

Table 4: Model 3 Optimal Result

	Obj.Func.Val.	$Pool_A$	$Pool_B$	$Pool_C$	$Pool_D$	Y_A	Y_B	Y_C	Y_D	X
Optimal Award Amount	0	0	0	0	0	0	0	0	0	0