Looper Churn Rate Analysis

Youssef Agour

2023-10-30

Introductions

Abstract

This analysis aims to pinpoint the stage at which individuals in the Pursuit fellowship program are most susceptible to exiting, referred to as "looping out." We are trying to use historical data to make actionable insights on reducing this churn rate. The primary objectives include determining the overall looper churn rate, categorizing characteristics defining the 'looper profile,' and understanding the causative factors leading to fellows exiting the program. [redacted] The desired outcome is to identify a separation timing marked as 'Completed,' indicating successful program completion, and thereby, maximizing the number of fellows who successfully finish the fellowship. This analysis provides valuable insights for optimizing the fellowship program and enhancing participant retention.

About Pursuit

Pursuit, a one of a kind social impact organization established in 2011 and headquartered in New York City, is dedicated to bridging the opportunity gap in America. Focused on training individuals with the greatest need and potential, Pursuit aims to propel individuals from underrepresented communities into tech, fostering career advancement and nurturing the emergence of future tech leaders. On average, Pursuit Fellows witness a significant boost in their annual income from 18,000 to nearly 90,000, securing positions at esteemed companies such as Citi, Uber, Peloton, and Blackstone. The Pursuit Fellowship journey begins with a rigorous and supportive year of training, followed by three years of dedicated career development post-employment in the form of Pursuit Commit. Pursuit collaborates directly with companies to secure long-term hiring commitments, contributing to sustained growth and retention. Employing a blended finance model, Pursuit ensures the sustainability and scalability of the Pursuit Fellowship by leveraging both philanthropic capital and impact investor funding. Dismantling barriers and facilitating connections between talent and opportunities, Pursuit empowers its Fellows to be faces of meaningful change.

Analysis Goals

1. Determine the overall looper churn rate in the Pursuit Fellowship program.
2. Categorize characteristics defining the 'looper profile.'
3. Understand the causative factors leading to fellows exiting the program.

Our Data

[redacted]

[redacted]It records everyone at Pursuit who has ever looped out of the program. This includes and is limited to: Fellows currently looped out, regardless of if they are on their first enrollment; Fellows who were looped out but eventually withdrew; Fellows who looped at one point and completed the program in 2 or more enrollments; Fellows who looped at a certain point in time and are currently enrolled at Pursuit. [redacted] The reason for this was due to the way data had existed in our internal stores. Aggregating the data set was done separately for a period before the start of this document to ensure cross referential integrity from archived stores, as well as with original data owners. The data was formatted and cleaned external to this document in Google sheets.

Methodology

In this analysis the leveraging of descriptive, predictive and causal models were used to cross reference performance between different statistical methods and to understand what best suits the structure of our data. Following the collection of basic descriptive statistics, decision tree and random forest modeling were conducted primarily to begin as a reference for the accuracy of classification. After understanding the outtakes of those models, a PCA (Principal component analysis) was conducted in order to help reduce the dimensionality of our data and understand what of our many attributes had the most pull in determining clustering done by K-Means algorithms, to group our historical data. From this a causal logistic regression was used in order to quantify that causation as well as make further predictions, where we then finally conducted a Naive Bayes model to further again produce what probability certain attributes contributed to whether or not a fellow looped. Naive Bayes was used due to the large amount of leveled attributes in our data, and the seeming lack of correlations found across them, this was the logical choice over something overly complex for this project such as a neural network, where a small set like ours would surely over fit the model. In the end the Naive Bayes was used and leveraged heavily in this process due to its usefulness with limited data sets, its assumption that all independent variables have an equal effect on the response variable, and its ease of deployment. A combination of each of these findings in the order I have mentioned them helped to define the structure of this report and each model helped us understand different data trends and see important things to look at.

The Analysis

Loading Packages

```
library(tidyverse) #for data functions
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.3     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(magrittr)
```

```
##
## Attaching package: 'magrittr'
##
## The following object is masked from 'package:purrr':
##
##     set_names
##
## The following object is masked from 'package:tidyr':
##
##     extract
```

Disclaimers

All fellows in this data are records of the latest enrollment of everyone at Pursuit who has ever looped, fellows who have 'Latest_Status' marked as 'Enrolled' are the subjects of this analysis' predictions. Any Fellow marked with 0 loops has looped out and not yet looped back into the program. This also indicates they are on their first enrollment

The original data-set as it was extracted can be found here:

```r
OG_Data <- read.csv('Original Data.csv')
# I tend to like seeing the data set after I run something as it helps me create more intimacy with it,
# View(OG_Data)
```

Further Notable Data Changes/Modifications

- Columns with Yes or No responses have been formatted to binary outputs. For Yes or No columns specifically, a lack of response was assumed to be 'No'.
- Our age Data was inconsistent in 2 records, these have been filled in with the average age value of 34.
- If separation timing was left blank in the original data it was either due to the fellow completing the program or the data being unavailable. Those who have completed the program have listed separations as 'Completed' and those with missing values have been omitted in the following data sets. There were less than 10 omissions.
- Columns 'Fellow + Class', 'TBC', 'Application_Borough___c (from Fellow)' have been omitted either due to lack of data or not being used in the analysis.
- Column 'Season' has been added, for more details about categorizations, refer to the data dictionary.

Data Dictionary

Attribute	Definition	Notes
SF_Enroll_ID	ID as listed on Sales force	
Fellow	Fellow Name	
Latest_Class	Class apart of in cohort	
Cohort	Latest Cohort Enrollment	
Latest_Status	Enrollment Status at the time of data collection	
Separation_Timing	Which Module did the fellow Leave Pursuit out of	Pre-Orientation; Module 1-6; Completed

Attribute	Definition	Notes
ReapplyRec	Was the Fellow Recommended to Reapply to the Program?	1-Yes; 0-No
Schedule	What schedule did the fellow take on?	Daytime; N+W
Enrollments	Number of times enrolled in the program	
Times_Looped	Number of Times looped back in	Enrollments - 1; If = 0, first-time looper on the first enrollment
Most_Recent_Grade	Grade most recently recorded while in Core	Percentages formatted as decimals
Age	Age	
Gender	Gender	Man; Woman; Other
Income	Categorization of Income	No Income; 12,500 or Below; 25,000 or Below; 50,000 or Below
Ethnicity	Fellow's indicated Ethnicity	African American/Black; Hispanic or Latino; Middle Eastern; Asian; White/Caucasian; Two or More Races; Other
Education	Highest level of Education Attained	DNF-HS (Did not finish HS); HS/GED; TS(Trade School); Associates; Bachelor's; Masters
Immigrant	Whether or not fellow is an Immigrant	1-Yes; 0-No, Assumed to be no if left blank
Dependents	Whether or not fellow has dependents	1-Yes; 0-No, Assumed to be no if left blank
Is_a_Dependent	Whether or not fellow is a dependent	1-Yes; 0-No, Assumed to be no if left blank
Core_Start	Year Started Core	
Cycle	Season started core	Winter(Nov-Jan); Spring(Feb-Apr); Summer(May-July); Fall(Aug-Oct)
Public_Assistance	Whether or not fellow is receiving public assistance	1-Yes; 0-No, Assumed to be no if left blank
Emp_Status	Recorded status of employment for the fellow during the program	PT(Employed Part-Time), FT(Employed Full-Time), JS("Not employed, but looking for work" (Job Search)), UE(Not employed and not looking for work (Unemployed)), SE (Self Employed)

Loading & Formatting Our Data

```
#The data set which or model will be fully trained on
Full <- read.csv('Full_Set.csv')
```

```
#Pre-split of the Full_Set in order to judge viability of models
Train <- read.csv('Training_Set.csv')
Test <- read.csv('Test_Set.csv')

#The set with Separation_Timing blank i.e. also includes fellows who are enrolled
TBD <- read.csv('Prediction_Set.csv')
```

Ensure the data has loaded in properly and get familiar with column names, regular format of values etc.

```
head(TBD)
```

```
##                 NA          0  Daytime           3            2
## 2                NA          0  Daytime           3            2
## 3                NA          0      N+W           3            2
## 4                NA          0      N+W           2            1
## 5                NA          0      N+W           2            1
## 6                NA          0      N+W           2            1
##   Most_Recent_Grade Age Gender           Income              Ethnicity
## 1              0.80  30    Man $12,500 or Below African American/Black
## 2              0.74  30  Woman $50,000 or Below     Hispanic or Latine
## 3              0.73  46  Woman $50,000 or Below         Middle Eastern
## 4              0.85  31  Woman $50,000 or Below African American/Black
## 5              0.74  25  Woman $25,000 or Below      Two or More Races
## 6              0.73  36  Woman        No Income      Two or More Races
##     Education Immigrant Dependents Is_a_Dependent Core_Start  Cycle
## 1      HS/GED         0          0              0       2023 Spring
## 2      HS/GED         0          0              0       2021 Winter
## 3      DNF-HS         1          0              0       2021 Winter
## 4      HS/GED         0          0              0       2022   Fall
## 5          TS         0          0              0       2022 Summer
## 6 Associate's         1          1              0       2022 Winter
##   Public_Assistance Emp_Status
## 1                 0         JS
## 2                 0         SE
## 3                 1         SE
## 4                 1         PT
## 5                 0         PT
## 6                 1         JS
```

```
# View(TBD)

#lets see how well the data imported over and if we need to fix column values
str(TBD)
```

```
## 'data.frame':    39 obs. of  23 variables:
```

```
##  $ Schedule          : chr  "N+W" "Daytime" "N+W" "N+W" ...
##  $ Enrollments       : int  2 3 3 2 2 2 3 2 3 4 ...
##  $ Times_Looped      : int  1 2 2 1 1 1 2 1 2 3 ...
##  $ Most_Recent_Grade: num  0.8 0.74 0.73 0.85 0.74 0.73 0.75 0.8 0.76 0.93 ...
##  $ Age               : int  30 30 46 31 25 36 36 48 34 45 ...
##  $ Gender            : chr  "Man" "Woman" "Woman" "Woman" ...
##  $ Income            : chr  "$12,500 or Below" "$50,000 or Below" "$50,000 or Below" "$50,000 or Belo
##  $ Ethnicity         : chr  "African American/Black" "Hispanic or Latine" "Middle Eastern" "African A
##  $ Education         : chr  "HS/GED" "HS/GED" "DNF-HS" "HS/GED" ...
##  $ Immigrant         : int  0 0 1 0 0 1 0 0 0 0 ...
##  $ Dependents        : int  0 0 0 0 0 1 1 0 0 1 ...
##  $ Is_a_Dependent    : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Core_Start        : int  2023 2021 2021 2022 2022 2022 2022 2022 2021 2021 ...
##  $ Cycle             : chr  "Spring" "Winter" "Winter" "Fall" ...
##  $ Public_Assistance: int  0 0 1 1 0 1 0 0 1 1 ...
##  $ Emp_Status        : chr  "JS" "SE" "SE" "PT" ...
```

```
#Data is not arranged in factors and some variables should be changed from numbers etc.

Full[Full == "Trade Degree"] <- "TS"
Train[Train == "Trade Degree"] <- "TS"
TBD[TBD == "Hispanic or Latino/a"] <- "Hispanic or Latine"

# Noticed that there were stray values that didn't fit into how I wanted my variables to factor
```

Because all of our CSV files have been formatted the same way, I will apply all structural changes across all 4 tables.

```
Full$Latest_Class <- factor(Full$Latest_Class, levels = c("6.2", "6.3", "6.4",  "7.1",  "7.2",  "8.1",
Full$Cohort <- factor(Full$Cohort, levels = c(6, 7, 8, 9, 10))
Full$Core_Start <- factor(Full$Core_Start, levels = c(2018, 2019, 2020, 2021, 2022, 2023))
#Realized later on the above 3 should be factorized as well, latest class is a category, not a variable
Full$Latest_Status <- factor(Full$Latest_Status, levels = c("Withdrawn", "Looped Back", "Completed"))
Full$Separation_Timing <- factor(Full$Separation_Timing, levels = c("Pre-Orientation", "Module 1", "Modu
Full$ReapplyRec <- factor(Full$ReapplyRec)
Full$Schedule <- factor(Full$Schedule, levels = c("Daytime", "N+W"))
Full$Gender <- factor(Full$Gender, levels = c("Man", "Woman", "Other"))
Full$Income <- factor(Full$Income, levels = c("No Income", "$12,500 or Below", "$25,000 or Below", "$50
Full$Ethnicity <- factor(Full$Ethnicity)
Full$Education <- factor(Full$Education, levels = c("DNF-HS", "HS/GED", "TS", "Associate's", "Bachelor'
Full$Immigrant <- factor(Full$Immigrant)
Full$Dependents <- factor(Full$Dependents)
Full$Is_a_Dependent <- factor(Full$Is_a_Dependent)
Full$Cycle <- factor(Full$Cycle, levels = c("Winter", "Spring", "Summer", "Fall"))
Full$Public_Assistance <- factor(Full$Public_Assistance)
Full$Emp_Status <- factor(Full$Emp_Status)
```

Checking to see if the restructure worked:

```
str(Full)
```

```
##  $ Schedule          : Factor w/ 2 levels "Daytime","N+W": 1 1 2 1 1 2 1 2 1 1 ...
##  $ Enrollments       : int  2 1 1 1 2 2 1 1 4 2 ...
##  $ Times_Looped      : int  1 0 0 0 1 1 0 0 3 1 ...
##  $ Most_Recent_Grade: num  0.83 0.76 0.87 0.97 0.98 0.71 0.95 0.78 0.99 0.94 ...
##  $ Age               : int  24 26 25 25 46 33 33 25 36 32 ...
##  $ Gender            : Factor w/ 3 levels "Man","Woman",..: 1 2 2 1 1 2 2 1 2 2 ...
##  $ Income            : Factor w/ 4 levels "No Income","$12,500 or Below",..: 1 1 3 3 3 1 1 4 4 1 ...
##  $ Ethnicity         : Factor w/ 6 levels "African American/Black",..: 2 1 5 5 3 3 1 3 3 5 ...
##  $ Education         : Factor w/ 6 levels "DNF-HS","HS/GED",..: 2 2 2 1 4 2 4 2 4 2 ...
##  $ Immigrant         : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
##  $ Dependents        : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ Is_a_Dependent    : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Core_Start        : Factor w/ 6 levels "2018","2019",..: 4 6 6 5 3 5 6 5 4 4 ...
##  $ Cycle             : Factor w/ 4 levels "Winter","Spring",..: 3 2 2 4 1 3 3 4 1 1 ...
##  $ Public_Assistance: Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 2 1 2 ...
##  $ Emp_Status        : Factor w/ 5 levels "FT","JS","PT",..: 2 2 1 1 1 2 2 1 1 2 ...
```

```
# View(Full)
```

Applying to the rest of our tables:

```
#For the Training Set
Train$Latest_Class <- factor(Train$Latest_Class, levels = c("6.2", "6.3",   "6.4",  "7.1",  "7.2",  "8.
Train$Cohort <- factor(Train$Cohort, levels = c(6, 7, 8, 9, 10))
Train$Core_Start <- factor(Train$Core_Start, levels = c(2018, 2019, 2020, 2021, 2022, 2023))
Train$Latest_Status <- factor(Train$Latest_Status, levels = c("Withdrawn", "Looped Back", "Completed"))
Train$Separation_Timing <- factor(Train$Separation_Timing, levels = c("Pre-Orientation", "Module 1", "M
Train$ReapplyRec <- factor(Train$ReapplyRec)
Train$Schedule <- factor(Train$Schedule, levels = c("Daytime", "N+W"))
Train$Gender <- factor(Train$Gender)
Train$Income <- factor(Train$Income, levels = c("No Income", "$12,500 or Below", "$25,000 or Below", "$
Train$Ethnicity <- factor(Train$Ethnicity)
Train$Education <- factor(Train$Education, levels = c("DNF-HS", "HS/GED", "TS", "Associate's", "Bachelo
Train$Immigrant <- factor(Train$Immigrant)
Train$Dependents <- factor(Train$Dependents)
Train$Is_a_Dependent <- factor(Train$Is_a_Dependent)
Train$Cycle <- factor(Train$Cycle, levels = c("Winter", "Spring", "Summer", "Fall"))
Train$Public_Assistance <- factor(Train$Public_Assistance)
Train$Emp_Status <- factor(Train$Emp_Status)

#For the Test Set
```

```
Test$Latest_Class <- factor(Test$Latest_Class, levels = c("6.2", "6.3", "6.4",  "7.1",  "7.2",  "8.1",
Test$Cohort <- factor(Test$Cohort, levels = c(6, 7, 8, 9, 10))
Test$Core_Start <- factor(Test$Core_Start, levels = c(2018, 2019, 2020, 2021, 2022, 2023))
Test$Latest_Status <- factor(Test$Latest_Status, levels = c("Withdrawn", "Looped Back", "Completed"))
Test$Separation_Timing <- factor(Test$Separation_Timing, levels = c("Pre-Orientation", "Module 1", "Modu
Test$ReapplyRec <- factor(Test$ReapplyRec)
Test$Schedule <- factor(Test$Schedule, levels = c("Daytime", "N+W"))
Test$Gender <- factor(Test$Gender)
Test$Income <- factor(Test$Income, levels = c("No Income", "$12,500 or Below", "$25,000 or Below", "$50
Test$Ethnicity <- factor(Test$Ethnicity)
Test$Education <- factor(Test$Education, levels = c("DNF-HS", "HS/GED", "TS", "Associate's", "Bachelor'
Test$Immigrant <- factor(Test$Immigrant)
Test$Dependents <- factor(Test$Dependents)
Test$Is_a_Dependent <- factor(Test$Is_a_Dependent)
Test$Cycle <- factor(Test$Cycle, levels = c("Winter", "Spring", "Summer", "Fall"))
Test$Public_Assistance <- factor(Test$Public_Assistance)
Test$Emp_Status <- factor(Test$Emp_Status)

#For our Prediction Set, where separation is 'To be Determined'
TBD$Latest_Class <- factor(TBD$Latest_Class, levels = c("6.2", "6.3",   "6.4",  "7.1",  "7.2",  "8.1",
TBD$Cohort <- factor(TBD$Cohort, levels = c(6, 7, 8, 9, 10))
TBD$Core_Start <- factor(TBD$Core_Start, levels = c(2018, 2019, 2020, 2021, 2022, 2023))
TBD$Latest_Status <- factor(TBD$Latest_Status, levels = c("Withdrawn", "Looped Back", "Completed"))
TBD$Separation_Timing <- factor(TBD$Separation_Timing, levels = c("Pre-Orientation", "Module 1", "Modul
TBD$ReapplyRec <- factor(TBD$ReapplyRec)
TBD$Schedule <- factor(TBD$Schedule, levels = c("Daytime", "N+W"))
TBD$Gender <- factor(TBD$Gender)
TBD$Income <- factor(TBD$Income, levels = c("No Income", "$12,500 or Below", "$25,000 or Below", "$50,0
TBD$Ethnicity <- factor(TBD$Ethnicity)
TBD$Education <- factor(TBD$Education, levels = c("DNF-HS", "HS/GED", "TS", "Associate's", "Bachelor's"
TBD$Immigrant <- factor(TBD$Immigrant)
TBD$Dependents <- factor(TBD$Dependents)
TBD$Is_a_Dependent <- factor(TBD$Is_a_Dependent)
TBD$Cycle <- factor(TBD$Cycle, levels = c("Winter", "Spring", "Summer", "Fall"))
TBD$Public_Assistance <- factor(TBD$Public_Assistance)
TBD$Emp_Status <- factor(TBD$Emp_Status)
```

Check that it worked.

```
unique(TBD$Latest_Class)
```

```
## [1] 10.4 10.1 9.6  10.2 10.3 9.5
## 19 Levels: 6.2 6.3 6.4 7.1 7.2 8.1 8.2 8.3 8.4 9.1 9.2 9.3 9.4 9.5 9.6 ... 10.4
```

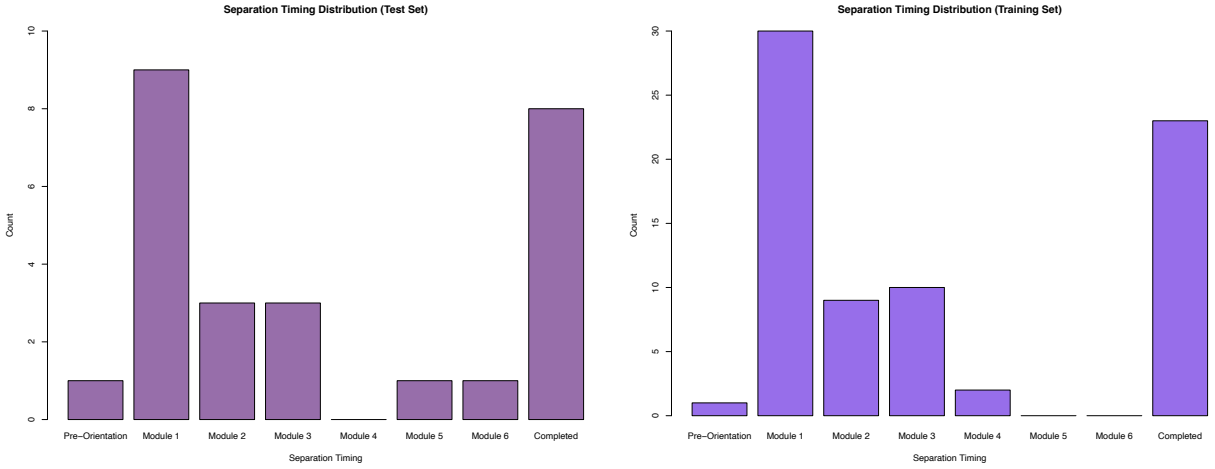
```
# View(TBD)
# Please note 'Latest_Status' is N/A for this set as 'Enrolled' was not included as a factor. This does
```

Proof of proper apportionment between Training & Test Sets

While the data was manually split for the purpose of this task, the below is to demonstrate the reliability of the split. The shape of the two charts between distribution of the Separation timing variable reflects

proportionality across the data, and this was to ensure the model was not over fit to deliver a bias towards certain modules.

```
par(mfrow = c(1, 2))

#Counting each level of Separation_Timing in the Test Set
Test_Table <- table(Test$Separation_Timing)
barplot(Test_Table, main = "Separation Timing Distribution (Test Set)", xlab = "Separation Timing", ylab
        ylim=c(0,10))

#Counting each level of Separation_Timing in the Training Set
Train_Table <- table(Train$Separation_Timing)
barplot(Train_Table, main = "Separation Timing Distribution (Training Set)", xlab = "Separation Timing"
```




```
par(mfrow = c(1, 1))
```

Descriptive Statistics

Below are descriptive Statistics on our Data set as a collective whole, for this purpose, we will join the Full & TBD dataset together to view in one place all records of Pursuit loopers at this point in time.

```
Joined <- rbind(Full, TBD)
All_loopers <- Joined[order(Joined$Fellow),]
All <- All_loopers[, -c(1, 2)]
# View(All)
str(All)
```

```
## 'data.frame':    140 obs. of  21 variables:
##  $ Latest_Class     : Factor w/ 19 levels "6.2","6.3","6.4",..: 17 15 17 12 16 15 17 12 6 15 ...
##  $ Cohort           : Factor w/ 5 levels "6","7","8","9",..: 5 4 5 4 5 4 5 4 3 4 ...
##  $ Latest_Status    : Factor w/ 3 levels "Withdrawn","Looped Back",..: NA NA NA 2 2 NA 2 2 3 2 ...
##  $ Separation_Timing: Factor w/ 8 levels "Pre-Orientation",..: NA NA NA 4 2 NA 3 2 8 4 ...
##  $ ReapplyRec       : Factor w/ 2 levels "0","1": 1 1 1 2 2 1 2 1 1 2 ...
##  $ Schedule         : Factor w/ 2 levels "Daytime","N+W": 2 2 2 1 1 2 2 1 1 2 ...
##  $ Enrollments      : int  2 2 2 2 1 3 1 1 2 2 ...
```

```
##  $ Times_Looped     : int  1 1 1 1 0 2 0 0 1 1 ...
##  $ Most_Recent_Grade: num  0.8 0.91 0.72 0.83 0.76 0.73 0.87 0.97 0.98 0.71 ...
##  $ Age              : int  48 29 51 24 26 46 25 25 46 33 ...
##  $ Gender           : Factor w/ 3 levels "Man","Woman",..: 1 1 2 1 2 2 2 1 1 2 ...
##  $ Income           : Factor w/ 4 levels "No Income","$12,500 or Below",..: 2 4 4 1 1 4 3 3 3 1 ...
##  $ Ethnicity        : Factor w/ 7 levels "African American/Black",..: 3 1 1 2 1 7 5 5 3 3 ...
##  $ Education        : Factor w/ 6 levels "DNF-HS","HS/GED",..: 5 2 2 2 2 1 2 1 4 2 ...
##  $ Immigrant        : Factor w/ 2 levels "0","1": 1 1 2 1 1 2 1 1 2 1 ...
##  $ Dependents       : Factor w/ 2 levels "0","1": 1 1 2 1 1 1 1 1 1 2 ...
##  $ Is_a_Dependent   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Core_Start       : Factor w/ 6 levels "2018","2019",..: 5 5 4 4 6 4 6 5 3 5 ...
##  $ Cycle            : Factor w/ 4 levels "Winter","Spring",..: 4 4 1 3 2 1 2 4 1 3 ...
##  $ Public_Assistance: Factor w/ 2 levels "0","1": 1 1 2 1 1 2 1 2 1 1 ...
##  $ Emp_Status       : Factor w/ 5 levels "FT","JS","PT",..: 3 1 1 2 2 4 1 1 1 2 ...
```

Notable Observations - Average Age: 34.12 - Average Program Grade: 84.1% - Modal Separation Module: Module 1 - Est. 41.4% of loopers are recommended to reapply - Although slight, more enrollments in N+W as well as Female fellows that are loopers - Max enrollments: 4, Max Loops: 3 - Surprising over 20% of loopers have either a Bachelor's or Master's - Generally based off the summary statistics alone, it is very unlikely a fellow loops out of the program after Module 3

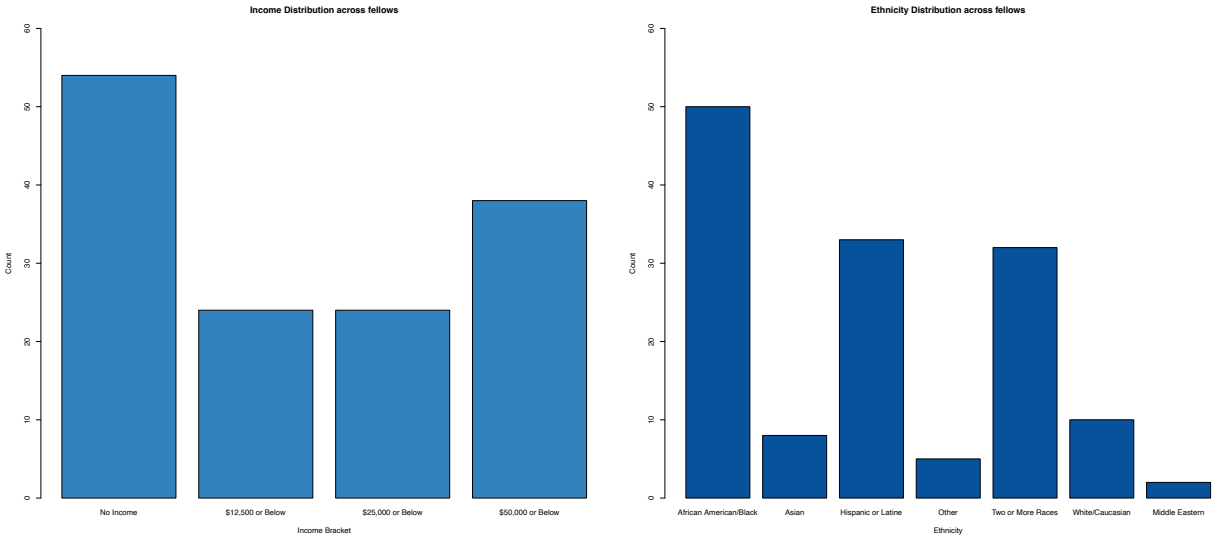
```
summary(All)
```

```
##   Latest_Class Cohort      Latest_Status       Separation_Timing ReapplyRec
##  10.2   :19    6 : 9   Withdrawn  :12    Module 1       :39      0:82
##  10.1   :16    7 : 3   Looped Back:58    Completed      :31      1:58
##  9.3    :14    8 :14   Completed  :31    Module 3       :13
##  9.6    :13    9 :64   NA's       :39    Module 2       :12
##  9.4    :12    10:50                     Pre-Orientation: 2
##  10.3   :11                              (Other)        : 4
##  (Other):55                              NA's           :39
##     Schedule   Enrollments     Times_Looped    Most_Recent_Grade
##  Daytime:62   Min.   :1.000   Min.   :0.0000   Min.   :0.700
##  N+W    :78   1st Qu.:1.000   1st Qu.:0.0000   1st Qu.:0.760
##               Median :2.000   Median :1.0000   Median :0.835
##               Mean   :1.893   Mean   :0.8929   Mean   :0.841
##               3rd Qu.:2.000   3rd Qu.:1.0000   3rd Qu.:0.920
##               Max.   :4.000   Max.   :3.0000   Max.   :1.000
##
##       Age          Gender                Income                     Ethnicity
##  Min.   :20.00   Man  :61   No Income       :54   African American/Black:50
##  1st Qu.:28.00   Woman:73   $12,500 or Below:24   Asian                  : 8
##  Median :33.00   Other: 6   $25,000 or Below:24   Hispanic or Latine     :33
##  Mean   :34.12              $50,000 or Below:38   Other                  : 5
##  3rd Qu.:38.00                                    Two or More Races      :32
##  Max.   :58.00                                    White/Caucasian        :10
##                                                   Middle Eastern         : 2
##        Education  Immigrant Dependents Is_a_Dependent Core_Start    Cycle
##  DNF-HS     : 3   0:113     0:110      0:118          2018: 3    Winter:53
##  HS/GED     :90   1: 27     1: 30      1: 22          2019:11    Spring:13
##  TS         : 5                                     2020: 9    Summer:42
##  Associate's:15                                     2021:34    Fall  :32
##  Bachelor's :25                                     2022:66
```

```
##  Master's   : 2                                        2023:17
##
##  Public_Assistance Emp_Status
##  0:72              FT:26
##  1:68              JS:67
##                    PT:22
##                    SE:17
##                    UE: 8
##
##
```

The majority of our fellows are either African American/Black, Hispanic/Latine, or Two or More Races. This category was defined for the purpose of this analysis to account for respondents who filled multiple racers in their application. The respondents who make up this portion overwhelmingly were still minorities. 100% of our looped fellows have incomes lower than $50,000 in a calendar year, with most listing no income whatsoever during their application year.

```
par(mfrow = c(1, 2))

#Visualizing Income
Income <- table(All$Income)
barplot(Income, main = "Income Distribution across fellows", xlab = "Income Bracket", ylab = "Count", c
        ylim=c(0,60))

#Visualizing Ethnicity
Ethnicity <- table(All$Ethnicity)
barplot(Ethnicity, main = "Ethnicity Distribution across fellows", xlab = "Ethnicity", ylab = "Count",
        ylim=c(0,60))
```

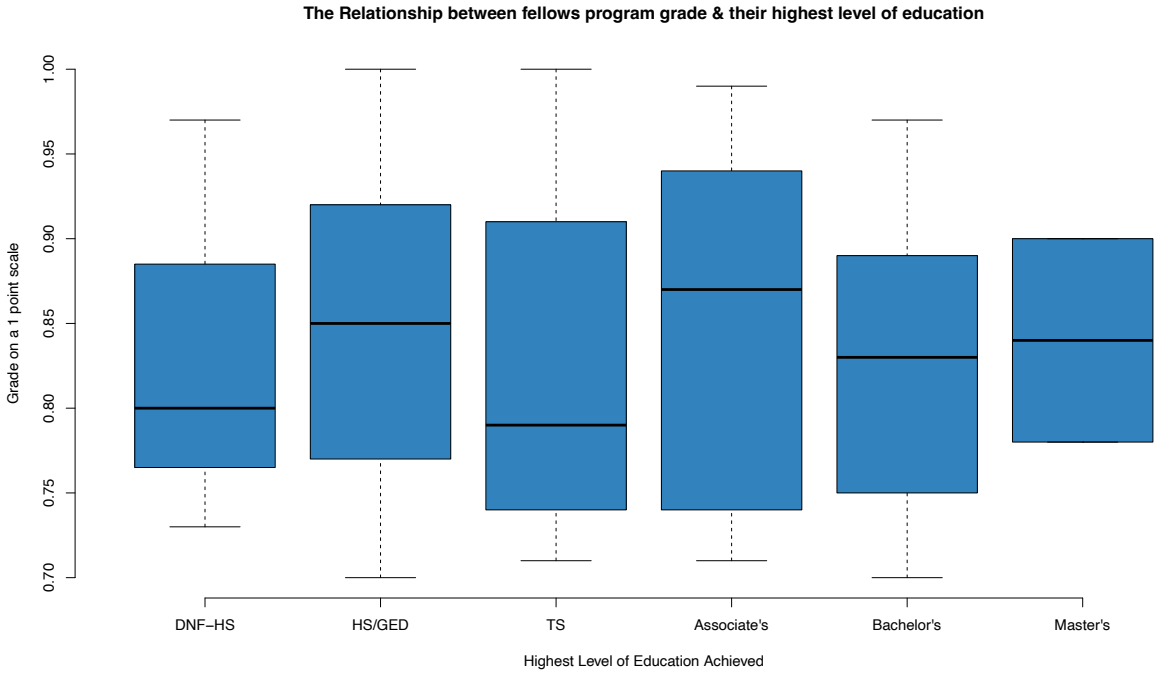



```
par(mfrow = c(1, 1))
```

Surprisingly, although showing great variation, on a median level of performance, fellows who held a High School diploma as their highest level of attained education, performed better than those with a Bachelor's Degree. Despite performing better than fellows who held a Master's Degree, the data present on fellows who

hold a Master's degree is highly minimal, and so this factor should not be weighed as heavily when judging our data. We want to focus on fellows between no high school degree, up to the attainment of a bachelor's.

```
plot(All$Education, All$Most_Recent_Grade, type="p", xlab="Highest Level of Education Achieved", ylab="
```



By surprise, there is no real correlation between age and the most recent grade attained by fellows. A good example of this is the two older fellows in the higher end of the 50s range. These two fellows representing the two oldest in the sample population, displaying a great range in their recorded grades, proves the lack of importance of Age as it impacts program grades.

```
# a scatter plot of age against grades including income & times looped
ggplot(data = All, aes(Age, Most_Recent_Grade, color = Schedule)) +
  geom_point() + ggtitle("The Relationship Between Age & Performance") + labs(y = "Grade on a 1.0 Scale
       color = "Schedule")
```


```
install.packages("scales")
```

```
##
## The downloaded binary packages are in
##  /var/folders/bs/_h1fsr8d6_g67rh2_kt902nm0000gn/T//RtmpmX5tHa/downloaded_packages
```

```
library(scales)
```

```
## Warning: package 'scales' was built under R version 4.3.1
```

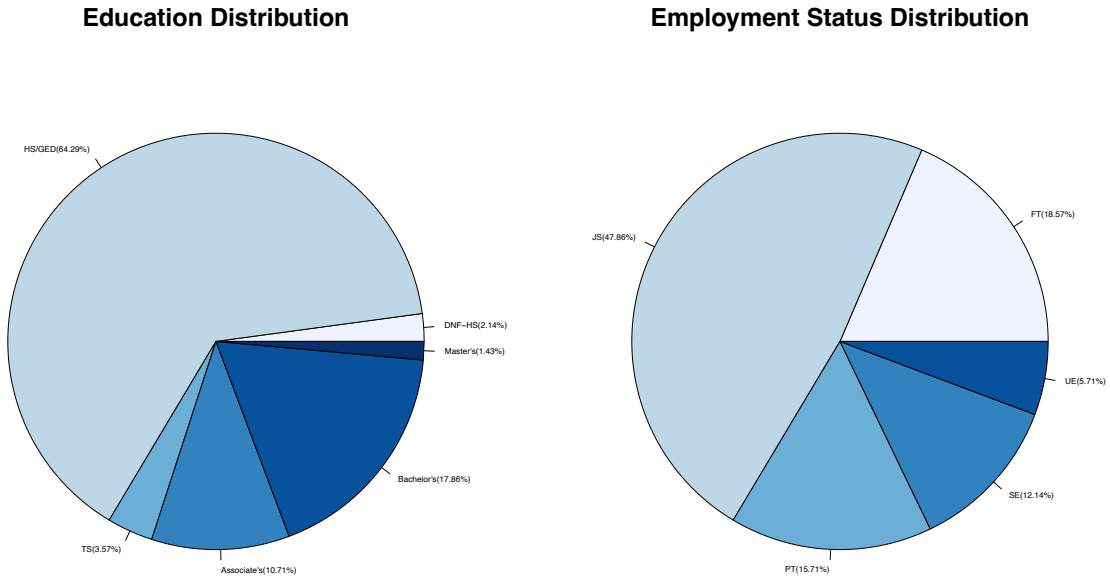
```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

The above package can be used for creating pie charts. Below are a few pie charts visualizing the general demographic information of our looper sample. Some core observations to note here, there are almost as many fellows who have dependents proportionally as there are fellows that are dependents themselves. This can be linked towards the income distribution that we see showing that about a third of fellows have no income, another third make above 0 but less than 25,000 annually, and the final third make between 25,000 and 50,000 annually. Over three quarters of our fellows do not have a Bachelor's Degree, with the overwhelming majority only holding a High School Diploma. More than half of our loopers receive public assistance, and more than 80% come from minority backgrounds.

```r
par(mfrow = c(1, 2))

# Gotta make it pretty
tonal_palette <- c("#eff3ff", "#bdd7e7", "#6baed6", "#3182bd", "#08519c", "#08306b")

# create a pie chart with percentages function! make pretty pretty
create_pie_chart <- function(data, title, colors) {
  table_data <- table(data)
  percentage_data <- (table_data / sum(table_data)) * 100
  labels <- paste(names(table_data), "(", round(percentage_data, 2), "%)", sep="")
  pie(table_data, labels = labels, col = colors, main = title, cex.main = 3)
}

# Create pie charts with percentages for different variables
create_pie_chart(All$Education, "Education Distribution", tonal_palette)
create_pie_chart(All$Emp_Status, "Employment Status Distribution", tonal_palette)
```




```r
par(mfrow = c(1, 1))
```

Predictive Modelling

After evaluating the findings our our descriptive statistics, we have an idea of the structure of our data set, and understand more about how it operates, and have identified basic characteristics of our sample population. In this next section, we start to jump into predictive models.

```r
# View(Full)
colnames(Full)
```

```
##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-8
```

```
library(rpart)
```

```
## Warning: package 'rpart' was built under R version 4.3.1
```

```
library(rattle)
```

```
## Loading required package: bitops

##
## Attaching package: 'bitops'

## The following object is masked from 'package:Matrix':
##
##     %&%

## Rattle: A free graphical interface for data science with R.
## Version 5.5.1 Copyright (c) 2006-2021 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.

##
## Attaching package: 'rattle'

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

```
library(rpart.plot)
library(RColorBrewer)
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 4.3.1

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

Random Forest + Decision Tree Modelling

```r
# Important!: When factor levels are removed by subsetting, you must reset levels:
levels(Train$Separation_Timing)
```

```
## [1] "Pre-Orientation" "Module 1"        "Module 2"        "Module 3"
## [5] "Module 4"        "Module 5"        "Module 6"        "Completed"
```

```r
Train$Separation_Timing <- factor(Train$Separation_Timing)
levels(Test$Separation_Timing)
```

```
## [1] "Pre-Orientation" "Module 1"        "Module 2"        "Module 3"
## [5] "Module 4"        "Module 5"        "Module 6"        "Completed"
```

```r
Test$Separation_Timing <- factor(Test$Separation_Timing)
```

```r
# Separate the dependent variable for the training set
Train_y <- Train$Separation_Timing
# Define the list of predictor variables
train_attributes <- c('Cohort', 'Latest_Class', 'ReapplyRec', 'Schedule', 'Enrollments', 'Times_Looped'
# Select columns corresponding to features for the training data
Train_X <- Train[, train_attributes]

# Separate the dependent variable for the test set
val_y <- Test$Separation_Timing
# Define the list of predictor variables for the test data
test_attributes <- c('Cohort', 'Latest_Class', 'ReapplyRec', 'Schedule', 'Enrollments', 'Times_Looped',
# Select columns corresponding to features for the test data
val_X <- Test[, test_attributes]

# Define a random forest model for multi-class classification with empty classes allowed
rf_model <- randomForest(Train_X, Train_y, ntree = 100, mtry = sqrt(length(train_attributes)), allowemp

# Make predictions on the validation set using the model
rf_val_predictions <- predict(rf_model, val_X)

# Convert factor variables to numeric
rf_val_predictions <- as.numeric(rf_val_predictions)
val_y <- as.numeric(val_y)

# Calculate the mean absolute error (MAE) for the validation set
rf_val_mae <- mean(abs(rf_val_predictions - val_y))

# Print the validation MAE
cat("Validation MAE for Random Forest Model: ", format(rf_val_mae, big.mark = ','), "\n")
```

```
## Validation MAE for Random Forest Model:  1.192308
```

```r
# Create a new Decision Tree model for the training set
dt_model_on_train <- rpart(Train_y ~ ., data = cbind(Train_X, Train_y), method = "class")

# Confusion Matrix (training data)
```

```r
conf.matrix_train <- table(Train$Separation_Timing, predict(dt_model_on_train, type = "class"))
rownames(conf.matrix_train) <- paste("Actual", rownames(conf.matrix_train), sep = ":")
colnames(conf.matrix_train) <- paste("Pred", colnames(conf.matrix_train), sep = ":")
print(conf.matrix_train)
```

```
##
##                          Pred:Pre-Orientation Pred:Module 1 Pred:Module 2
##   Actual:Pre-Orientation                    0             0             0
##   Actual:Module 1                           0            24             0
##   Actual:Module 2                           0             6             0
##   Actual:Module 3                           0             4             0
##   Actual:Module 4                           0             1             0
##   Actual:Completed                          0             1             0
##
##                          Pred:Module 3 Pred:Module 4 Pred:Completed
##   Actual:Pre-Orientation             0             0              1
##   Actual:Module 1                    1             0              5
##   Actual:Module 2                    2             0              1
##   Actual:Module 3                    4             0              2
##   Actual:Module 4                    1             0              0
##   Actual:Completed                   0             0             22
```

```r
# Scoring on the test set
val1_tree <- predict(dt_model_on_train, newdata = val_X, type = "prob")

# Storing Model Performance Scores
roc_val_tree <- pROC::roc(val_y, val1_tree[, 2])
```

```
## Warning in roc.default(val_y, val1_tree[, 2]): 'response' has more than two
## levels. Consider setting 'levels' explicitly or using 'multiclass.roc' instead
```

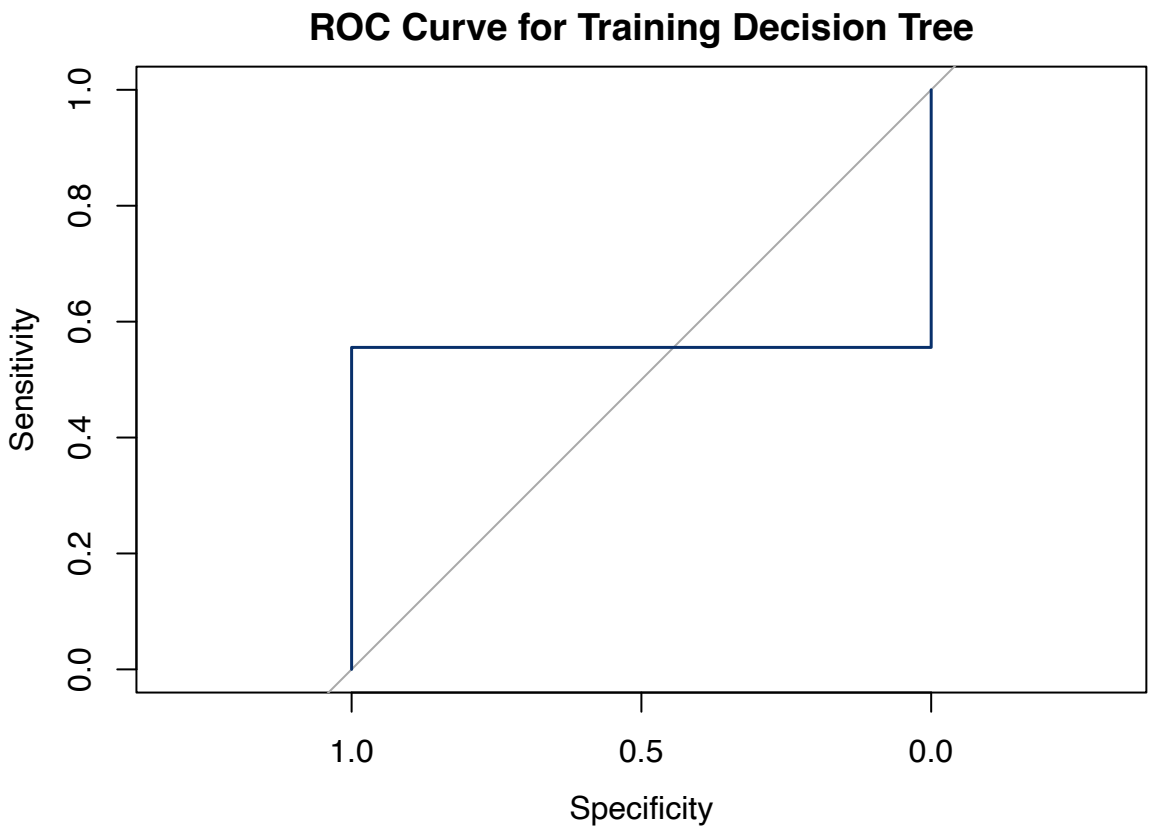
```
## Setting levels: control = 1, case = 2
```

```
## Setting direction: controls < cases
```

```r
# Calculating Area under Curve (AUC)
auc_val_tree <- pROC::auc(roc_val_tree)
cat("AUC:", auc_val_tree, "\n")
```

```
## AUC: 0.5555556
```

```r
# Plotting ROC curve
plot(roc_val_tree, col = "#08306b", lwd = 1.5, main = "ROC Curve for Training Decision Tree")
```




```
# Calculating KS statistics
ks_tree <- max(roc_val_tree$sensitivities - (1 - roc_val_tree$specificities))
cat("KS Statistic:", ks_tree, "\n")
```

```
## KS Statistic: 0.5555556
```

Random Forest - Interpreting Our Margin of Error

A MAE of 1.230769 in the context of the fellowship program with 8 stages (Assuming the inclusion of Pre-Orientation & Completion) where the stages represent when fellows leave the fellowship, indicates that, on average, the model's predictions are off by approximately 1.2 modules with respect to when fellows actually leave/loop out of the fellowship.

An MAE of 1.2 means that, on average, the model's predictions for the timing of when fellows leave or loop are off by approximately 1.2 modules. For example, if the model predicts that a fellow will leave after "Module 3," they may actually loop/leave after "Module 4," on average.

This level of error is quite significant for our purposes, however, the size of the data set, and also the continuity of the program in real life play a role here. For example, in our prediction data, at the time of writing there are quite a few members of our 9.5 & 9.6 cohorts in it. This means also that at the time of writing 9.5 & 9.6 are in their capstone module, which historically has showed quite low loop levels. Hence, if the models we generate say that a fellow will loop in Module 1 for example, even without the capstone module (Module 6) concluding, we know this will be a prediction made in error.

However, this data is still useful in terms of profiling the likely hood of looping/exiting the program for these fellows as it can help map the 'ideal looper' or give us a profile to focus on in admitting fellows to

the program. Further study will need to be conducted in order to cross reference this data as the cohorts included of the enrolled cohorts in this analysis progress through the program.

Decision Tree - Interpreting Our Validation Metrics

After conducting the decision tree we can see it is not suitable at all for the data we have at hand, ideally I do believe that from here a different algorithm should definitely be used. The AUC (Area Under the ROC Curve) serves as a measure of a classification model's discriminatory power by measuring the area under the graphical representation of the trade-off between sensitivity (true positive rate - someone who is an actual looper, predicted to loop) and specificity (true negative rate - someone who is not a looper predicted to complete core).

A model achieving perfect discrimination has an AUC of 1, while one performing no better than random chance has an AUC of 0.5. In our case, an AUC of 0.5555556, close to 0.5, indicative of performance not significantly better than random chance. The KS statistic, measuring the maximum vertical distance between cumulative distributions of positive and negative classes, similarly indicates a moderate level of discrimination with a value of 0.5555556.

Higher KS values are targeted as they signify better separation between positive and negative classes in classification problems. With this decision tree both through display of the curve and the hovering of both the AUC and KS stats around 0.5, our model essentially only has a 50/50 chance of producing the prediction we want.

This is not suitable, and making predictions off this would bear insignificant results. Ultimately, the simplicity of a decision tree, combined with our limited data set, essentially makes for this not being the optimal choice for making inferences.

Modelling the Full Data

Please give this chunk time to run.

```
# Separate the dependent variable for the Full set
Full_y <- Full$Separation_Timing
# Define the list of predictor variables
full_attributes <- c('Cohort', 'Latest_Class', 'ReapplyRec', 'Schedule', 'Enrollments', 'Times_Looped',
# Select columns corresponding to features for the training data
Full_X <- Full[, full_attributes]

# Create a new Random Forest model
rf_model_on_full_data <- randomForest(Full_X, Full_y, ntree = 1000, mtry = sqrt(length(full_attributes))

# Create a new Decision Tree Model
dt_model_on_full_data <- rpart(Full_y ~ ., data = cbind(Full_X, Full_y), method = "class")
```

Lets View our Random Forest Model.

```
rf_model_on_full_data
```

```
##
## Call:
##  randomForest(x = Full_X, y = Full_y, ntree = 1000, mtry = sqrt(length(full_attributes)),      allow
##                Type of random forest: classification
##                      Number of trees: 1000
```

```
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 41.58%
## Confusion matrix:
##                 Pre-Orientation Module 1 Module 2 Module 3 Module 4 Module 5
## Pre-Orientation               0        0        0        0        0        0
## Module 1                      0       29        2        4        0        0
## Module 2                      0        9        0        2        0        0
## Module 3                      0       10        0        0        0        0
## Module 4                      0        2        0        0        0        0
## Module 5                      0        1        0        0        0        0
## Module 6                      0        1        0        0        0        0
## Completed                     0        1        0        0        0        0
##                 Module 6 Completed class.error
## Pre-Orientation        0         2  1.00000000
## Module 1               0         4  0.25641026
## Module 2               0         1  1.00000000
## Module 3               0         3  1.00000000
## Module 4               0         0  1.00000000
## Module 5               0         0  1.00000000
## Module 6               0         0  1.00000000
## Completed              0        30  0.03225806
```

Lets View our Decision Tree Model.

```
dt_model_on_full_data
```

```
## n= 101
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 101 62 Module 1 (0.02 0.39 0.12 0.13 0.02 0.0099 0.0099 0.31)
##    2) ReapplyRec=1 58 24 Module 1 (0 0.59 0.19 0.17 0.034 0 0 0.017)
##      4) Latest_Class=9.4,10.1,10.2,10.3 30  7 Module 1 (0 0.77 0.2 0.033 0 0 0 0) *
##      5) Latest_Class=6.4,8.4,9.2,9.3,9.5,9.6 28 17 Module 1 (0 0.39 0.18 0.32 0.071 0 0 0.036)
##       10) Emp_Status=JS,PT 18  9 Module 1 (0 0.5 0 0.39 0.11 0 0 0) *
##       11) Emp_Status=FT,SE,UE 10  5 Module 2 (0 0.2 0.5 0.2 0 0 0 0.1) *
##    3) ReapplyRec=0 43 13 Completed (0.047 0.12 0.023 0.07 0 0.023 0.023 0.7) *
```

```
# Will return a table of the cross-validated error rate for the tree at different levels
printcp(dt_model_on_full_data)
```

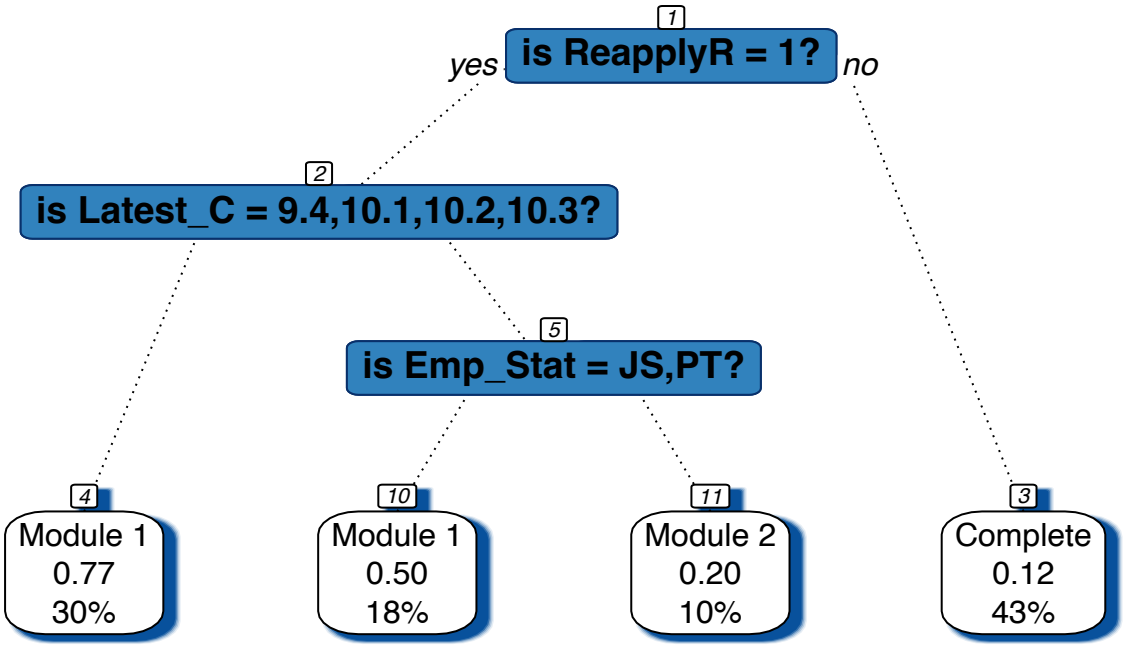
```
##
## Classification tree:
## rpart(formula = Full_y ~ ., data = cbind(Full_X, Full_y), method = "class")
##
## Variables actually used in tree construction:
## [1] Emp_Status   Latest_Class ReapplyRec
##
## Root node error: 62/101 = 0.61386
##
## n= 101
```

```
##
##         CP nsplit rel error  xerror     xstd
## 1 0.403226      0   1.00000 1.00000 0.078918
## 2 0.024194      1   0.59677 0.59677 0.078098
## 3 0.010000      3   0.54839 0.70968 0.080373
```

```r
# X axis is the complexity parameter, y axis shows the cross validated error rate, tried to kind of mak
prp(dt_model_on_full_data, main="How Our Decision Tree Makes Decisions",
    extra=106,
    nn=TRUE,
    fallen.leaves=TRUE,
    branch=.5,
    faclen=0,
    trace=1,
    shadow.col="#08519c",
    branch.lty=3,
    split.cex=1.2,
    split.prefix="is ",
    split.suffix="?",
    split.box.col="#3182bd",
    split.border.col="#08306b",
    split.round=.5)
```

```
## Warning: extra=106 but the response has 8 levels (only the 2nd level is
## displayed)
```

```
## cex 1   xlim c(0, 1)   ylim c(0, 1)
```



Interpreting Our Random Forest

Our Classification used to predict the class labels of the fellows who are currently enrolled has some flaws, but also shows great strength. The initial thing which seems to stand out the most is the number of times the Model believes it was able to correctly predict itself in terms of Module 1. According to the confusion matrix *refer to the above*, Module 1 accounts for 29 classifications in this set and seems to be responsible for most of our false positives. However, due to the class.error column of the other variables, some of the the other modules pretty much having guaranteed error, it confirms what was outlines towards the start of this analysis, that there is simply a shortage of data for loopers within the Module 4-6 or even Module 3-6 range.

However, one thing which did come out of this analysis as beneficial was the very low error rate of 0.06 for Completed looper fellows. Due to the size of this data set, I believe this result is highly statistically significant. What this could indicate is this model could be re-utilized to run a binary classification on whether or not someone will be a looper based on the entire Pursuit population.

While I worry that the results of this model come as a result of over fitting the data, (as there was a clear proportional bias towards either completion of the program or looping in module 1) the low class error for predicting completion gives me hope for this model's viability. This is especially meaningful as the OOB estimate of error rate at 41.58% is quite high overall, so to extract this finding is without a doubt special.

Interpreting Our Decision Tree

Root Node (Node 1): - The root node represents the entire dataset (n=101). - The most common class is "Module 1" with a probability of 0.39.

Decision Based on ReapplyRec: If ReapplyRec is 1 (Node 2), it further splits the data: - Most observations (24 out of 58) fall into "Module 1" with a probability of 0.59. - Another group (34 out of 58) is further split based on the Latest_Class variable. - If Latest_Class is in the range 9.4 to 10.3 (Node 4), the majority (7 out of 30) belongs to "Module 1" with a high probability of 0.77. - If Latest_Class is in the range 6.4 to 9.6 (Node 5), further splits occur based on the Emp_Status variable. - If Emp_Status is JS or PT (Node 10), the majority (9 out of 18) belongs to "Module 1" with a probability of 0.5. - If Emp_Status is FT, SE, or UE (Node 11), the majority (5 out of 10) belongs to "Module 2" with a probability of 0.5. - If ReapplyRec is 0 (Node 3), the majority (30 out of 43) belongs to "Completed" with a high probability of 0.7.

I have decided to keep the Decision tree visualization here as a means of communication to non-technical stakeholders how a tree-type model makes decisions. While I do not plan to make any further tests or investigations into the decision tree results due to its poor performance with this dataset, I think it will be useful in helping to explain the premise of a Random Forest, and displaying visually why our Random Forest makes the predictions it makes.

(*If these numbers look different than what you see in the R chunk output, follow to next section*)

```
rf_model_on_full_data$err.rate
```

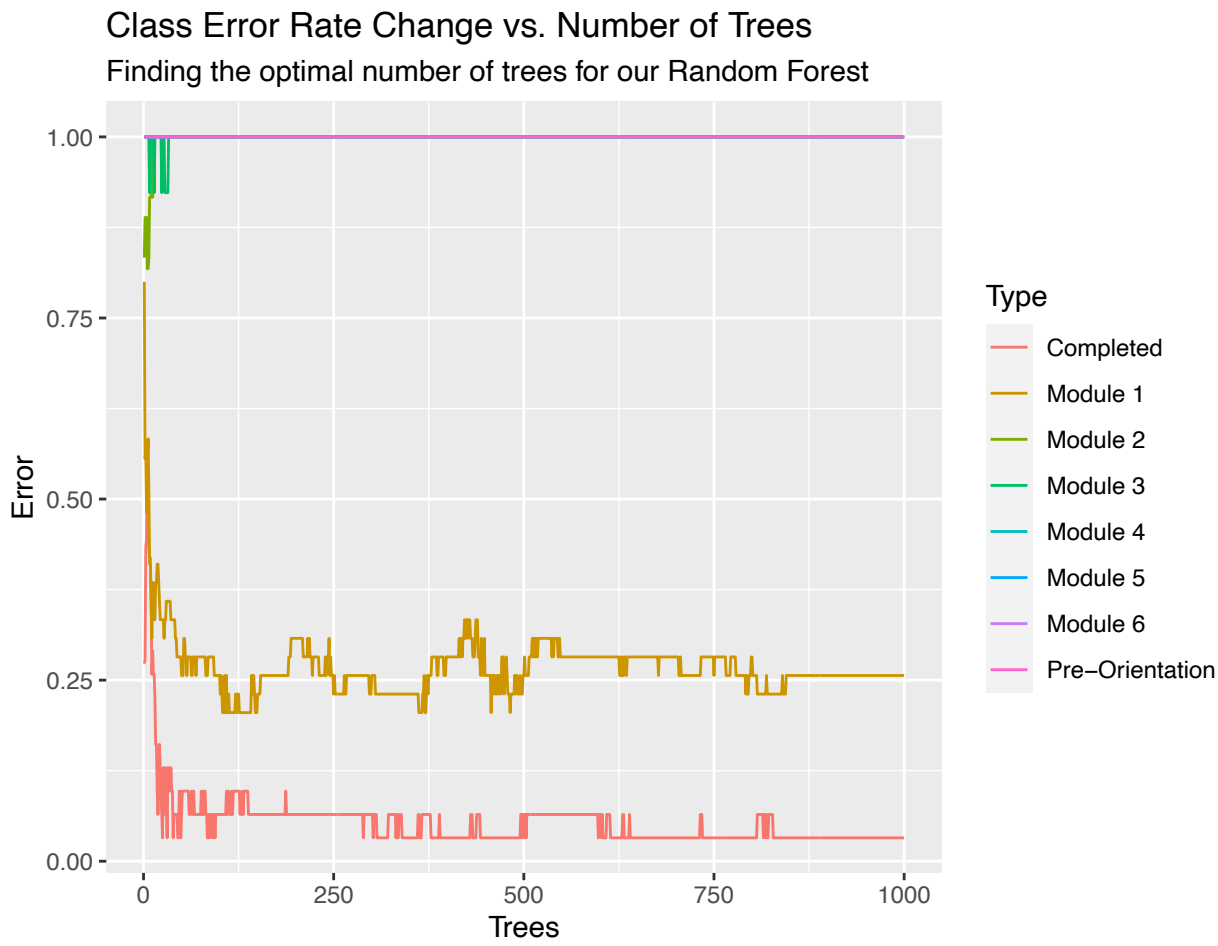
```
##               OOB Pre-Orientation  Module 1  Module 2  Module 3 Module 4
##    [1,] 0.7027027               1 0.8000000 0.8333333 1.0000000        1
##    [2,] 0.6140351               1 0.5555556 0.8888889 1.0000000        1
##    [3,] 0.6666667               1 0.5652174 0.8888889 1.0000000        1
##    [4,] 0.6250000               1 0.4814815 0.8888889 1.0000000        1
##    [5,] 0.6781609               1 0.5483871 0.8181818 1.0000000        1
##    [6,] 0.6666667               1 0.5833333 0.8181818 1.0000000        1
##    [7,] 0.6224490               1 0.5135135 0.8333333 1.0000000        1
##    [8,] 0.5742574               1 0.4102564 0.9166667 0.9230769        1
##    [9,] 0.5841584               1 0.4102564 0.9166667 1.0000000        1
##   [10,] 0.5445545               1 0.3589744 0.9166667 1.0000000        1
##   [11,] 0.5049505               1 0.3076923 1.0000000 1.0000000        1
##   [12,] 0.5346535               1 0.3846154 0.9166667 1.0000000        1
##   [13,] 0.5346535               1 0.3846154 1.0000000 1.0000000        1
##   [14,] 0.5049505               1 0.3333333 1.0000000 0.9230769        1
##   [15,] 0.5049505               1 0.3333333 1.0000000 1.0000000        1
##   [16,] 0.5049505               1 0.3846154 1.0000000 1.0000000        1
##   [17,] 0.5049505               1 0.3846154 1.0000000 1.0000000        1
##   [18,] 0.4950495               1 0.4102564 1.0000000 1.0000000        1
##   [19,] 0.4851485               1 0.4102564 1.0000000 1.0000000        1
##   [20,] 0.4851485               1 0.3846154 1.0000000 1.0000000        1
##   [21,] 0.4950495               1 0.3589744 1.0000000 1.0000000        1
##   [22,] 0.4752475               1 0.3333333 1.0000000 1.0000000        1
##   [23,] 0.4653465               1 0.3333333 1.0000000 1.0000000        1
##   [24,] 0.4455446               1 0.3333333 1.0000000 0.9230769        1
##   [25,] 0.4455446               1 0.3333333 1.0000000 1.0000000        1
##   [26,] 0.4752475               1 0.3333333 1.0000000 1.0000000        1
##   [27,] 0.4455446               1 0.3076923 1.0000000 1.0000000        1
##   [28,] 0.4653465               1 0.3333333 1.0000000 0.9230769        1
##   [29,] 0.4653465               1 0.3333333 1.0000000 0.9230769        1
##   [30,] 0.4653465               1 0.3589744 1.0000000 0.9230769        1
##   [31,] 0.4455446               1 0.3589744 1.0000000 0.9230769        1
##   [32,] 0.4752475               1 0.3589744 1.0000000 0.9230769        1
##   [33,] 0.4851485               1 0.3589744 1.0000000 1.0000000        1
##   [34,] 0.4851485               1 0.3589744 1.0000000 1.0000000        1
##   [35,] 0.4752475               1 0.3589744 1.0000000 1.0000000        1
```

```
##  [979,]        1        1 0.03225806
##  [980,]        1        1 0.03225806
##  [981,]        1        1 0.03225806
##  [982,]        1        1 0.03225806
##  [983,]        1        1 0.03225806
##  [984,]        1        1 0.03225806
##  [985,]        1        1 0.03225806
##  [986,]        1        1 0.03225806
##  [987,]        1        1 0.03225806
##  [988,]        1        1 0.03225806
##  [989,]        1        1 0.03225806
##  [990,]        1        1 0.03225806
##  [991,]        1        1 0.03225806
##  [992,]        1        1 0.03225806
##  [993,]        1        1 0.03225806
##  [994,]        1        1 0.03225806
##  [995,]        1        1 0.03225806
##  [996,]        1        1 0.03225806
##  [997,]        1        1 0.03225806
##  [998,]        1        1 0.03225806
##  [999,]        1        1 0.03225806
## [1000,]        1        1 0.03225806
```

The above shows that 41% of the time the model predicted the wrong module, however, this is largely in part to the 100% error probability for the prediction of Modules 3-6 which is without a doubt due to lack of data. Before closing the coffin on this however, it would be good to try and see if we could find a way to make this model more accurate by increasing the number of trees. See the below.

```
oob.error.data <- data.frame(
  Trees=rep(1:nrow(rf_model_on_full_data$err.rate), times=8),
  Type=rep(c("Pre-Orientation", "Module 1", "Module 2", "Module 3", "Module 4", "Module 5", "Module 6",
  Error=c(rf_model_on_full_data$err.rate[,"Pre-Orientation"],
          rf_model_on_full_data$err.rate[,"Module 1"],
          rf_model_on_full_data$err.rate[,"Module 2"],
          rf_model_on_full_data$err.rate[,"Module 3"],
          rf_model_on_full_data$err.rate[,"Module 4"],
          rf_model_on_full_data$err.rate[,"Module 5"],
          rf_model_on_full_data$err.rate[,"Module 6"],
          rf_model_on_full_data$err.rate[,"Completed"]))

ggplot(data=oob.error.data, aes(x=Trees, y=Error)) +
  geom_line(aes(color=Type)) + ggtitle("Class Error Rate Change vs. Number of Trees") + labs(subtitle =
```

```
## Warning: Removed 5 rows containing missing values (‘geom_line()‘).
```



After observing that there were still many fluctuations in our Random Forest as it applies to the number of trees, I figured that playing to achieve the best number of trees would not hurt.

"People tend to forget that play is serious." ~ David Hockney

Overall, after some trial and error, I have settled towards 1000 trees to be ideal for our trained model as the above plot shows that around the 500 mark, both our most significant classes begin to get stable. Although the overall error rate of the model increased slightly, our error for predicting fellows who will complete the program continues to support the belief that re-purposing this model for binary classification on a bigger set would bear great fruit.

```
oob.values <- vector(length=10)
for(i in 1:10) {
  temp_rf <- randomForest(Full_X, Full_y, ntree = 1000, mtry = i)
  oob.values[i] <- temp_rf$err.rate[nrow(temp_rf$err.rate),1]
}
# essentially creating a for loop to run through different numbers of variables at each step to try and

oob.values
```

```
##  [1] 0.3564356 0.3861386 0.3960396 0.4059406 0.4257426 0.4356436 0.4257426
##  [8] 0.4653465 0.4455446 0.4356436
```

What is interesting from the output above is that it recommends using only 1 variable to try at each tree split as the 1st value which corresponds to the lowest error rate in this return represents this. Let us try

running a model with only 1 variable tried at each tree split. This is called the out of bag error rate. The out-of-bag (OOB) error rate is a measure of the performance of a random forest model. It is a valuable tool for estimating how well the model will generalize to unseen data, without the need for a separate validation data set.

```
rf_002_on_full_data <- randomForest(Full_X, Full_y, ntree = 1000, mtry = 1, allowempty = TRUE)
rf_002_on_full_data
```

```
##
## Call:
##  randomForest(x = Full_X, y = Full_y, ntree = 1000, mtry = 1,      allowempty = TRUE)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 1
##
##         OOB estimate of  error rate: 35.64%
## Confusion matrix:
##                 Pre-Orientation Module 1 Module 2 Module 3 Module 4 Module 5
## Pre-Orientation               0        0        0        0        0        0
## Module 1                      0       34        0        0        0        0
## Module 2                      0       10        0        0        0        0
## Module 3                      0        9        0        0        0        0
## Module 4                      0        2        0        0        0        0
## Module 5                      0        1        0        0        0        0
## Module 6                      0        1        0        0        0        0
## Completed                     0        0        0        0        0        0
##                 Module 6 Completed class.error
## Pre-Orientation        0         2   1.0000000
## Module 1               0         5   0.1282051
## Module 2               0         2   1.0000000
## Module 3               0         4   1.0000000
## Module 4               0         0   1.0000000
## Module 5               0         0   1.0000000
## Module 6               0         0   1.0000000
## Completed              0        31   0.0000000
```

Random Forests are Great but can ALWAYS be Better

From the above we see that with the ideal number of attributes being tested at each split being assigned at 1 and the number of trees being upped to 1000, we have a significantly more accurate model than the one we first produced. Still terribly over fitted (due to data availability), but none the less, is much more accurate at predicting Module 1 loopers as well as those who will go onto complete the program at the lowest error rates we have observed thus far.

From here we will predict and assign the classes that result from the trained model. However, we will continue to use the initial Random Forest that was created as this still has the best base point of output that matches the shape of our original data.

```
# Make predictions on the new data
predictions <- predict(rf_model_on_full_data, new_data=TBD)
TBD$Separation_Timing <- predictions[1:39]
```

Saving the prediction data to a new structured CSV file.

```
write.csv(TBD, "rfmodel_looper_predictions.csv", row.names = FALSE)
```

```
Predictions_11_1_23 <- read.csv("rfmodel_looper_predictions.csv")
# View(Predictions_11_1_23)
head(Predictions_11_1_23)
```

```
##   Separation_Timing ReapplyRec Schedule Enrollments Times_Looped
## 1          Module 1          0      N+W           2            1
## 2          Module 1          0  Daytime           3            2
## 3          Module 1          0      N+W           3            2
## 4         Completed          0      N+W           2            1
## 5         Completed          0      N+W           2            1
## 6          Module 1          0      N+W           2            1
##   Most_Recent_Grade Age Gender           Income              Ethnicity
## 1              0.80  30    Man $12,500 or Below African American/Black
## 2              0.74  30  Woman $50,000 or Below     Hispanic or Latine
## 3              0.73  46  Woman $50,000 or Below         Middle Eastern
## 4              0.85  31  Woman $50,000 or Below African American/Black
## 5              0.74  25  Woman $25,000 or Below      Two or More Races
## 6              0.73  36  Woman        No Income      Two or More Races
##     Education Immigrant Dependents Is_a_Dependent Core_Start  Cycle
## 1      HS/GED         0          0              0       2023 Spring
## 2      HS/GED         0          0              0       2021 Winter
## 3      DNF-HS         1          0              0       2021 Winter
## 4      HS/GED         0          0              0       2022   Fall
## 5          TS         0          0              0       2022 Summer
## 6 Associate's         1          1              0       2022 Winter
##   Public_Assistance Emp_Status
## 1                 0         JS
## 2                 0         SE
## 3                 1         SE
## 4                 1         PT
## 5                 0         PT
## 6                 1         JS
```

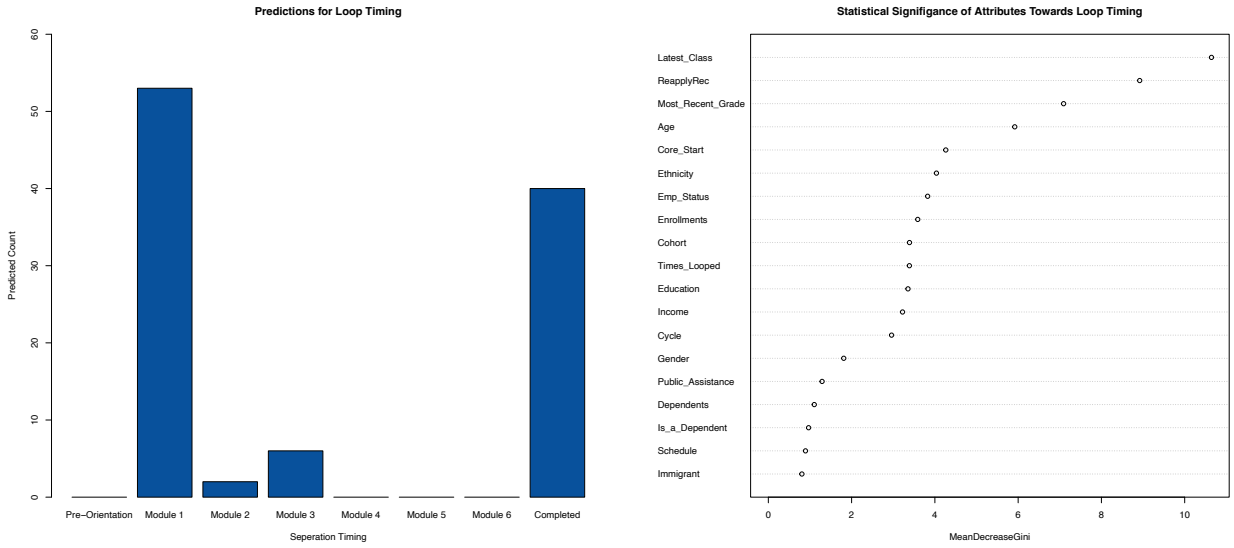
The above set will be used as predictive data at the conclusion at each of these cohorts to analyse whether or not our predictions were accurate to our purposes. Below is a visualization of our Random Forest predictions, as well as a graphic to help us identify the most important factors in this analysis that contributed to our results.

Engineering Our Predictions Set for later use in case we need it.

```
Predictions_11_1_23$Latest_Class <- factor(Predictions_11_1_23$Latest_Class, levels = c("6.2", "6.3", "
Predictions_11_1_23$Cohort <- factor(Predictions_11_1_23$Cohort, levels = c(6, 7, 8, 9, 10))
Predictions_11_1_23$Core_Start <- factor(Predictions_11_1_23$Core_Start, levels = c(2018, 2019, 2020, 2
Predictions_11_1_23$Latest_Status <- factor(Predictions_11_1_23$Latest_Status, levels = c("Withdrawn",
```

```
Predictions_11_1_23$Separation_Timing <- factor(Predictions_11_1_23$Separation_Timing, levels = c("Pre-
Predictions_11_1_23$ReapplyRec <- factor(Predictions_11_1_23$ReapplyRec)
Predictions_11_1_23$Schedule <- factor(Predictions_11_1_23$Schedule, levels = c("Daytime", "N+W"))
Predictions_11_1_23$Gender <- factor(Predictions_11_1_23$Gender, levels = c("Man", "Woman", "Other"))
Predictions_11_1_23$Income <- factor(Predictions_11_1_23$Income, levels = c("No Income", "$12,500 or Be
Predictions_11_1_23$Ethnicity <- factor(Predictions_11_1_23$Ethnicity)
Predictions_11_1_23$Education <- factor(Predictions_11_1_23$Education, levels = c("DNF-HS", "HS/GED", "
Predictions_11_1_23$Immigrant <- factor(Predictions_11_1_23$Immigrant)
Predictions_11_1_23$Dependents <- factor(Predictions_11_1_23$Dependents)
Predictions_11_1_23$Is_a_Dependent <- factor(Predictions_11_1_23$Is_a_Dependent)
Predictions_11_1_23$Cycle <- factor(Predictions_11_1_23$Cycle, levels = c("Winter", "Spring", "Summer",
Predictions_11_1_23$Public_Assistance <- factor(Predictions_11_1_23$Public_Assistance)
Predictions_11_1_23$Emp_Status <- factor(Predictions_11_1_23$Emp_Status)
```

```
par(mfrow = c(1,2))

#Visualizing our predictions and understanding the structure of our trees and what attribute had the mo
plot(predictions, xlab="Seperation Timing", ylab="Predicted Count", main="Predictions for Loop Timing",

# This is to help us determine what attribute had the most statistical significance on our model
varImpPlot(rf_model_on_full_data, main="Statistical Signifigance of Attributes Towards Loop Timing")
```



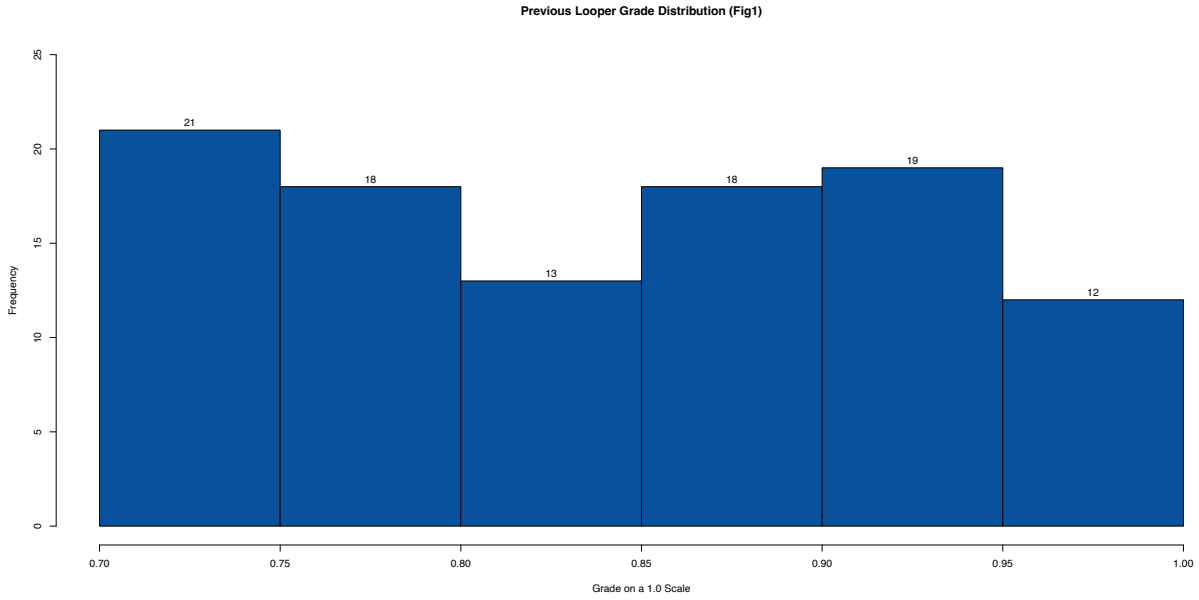

```
par(mfrow = c(1,1))
```

Refer back to chunk 9. Our Random Forest proportionally matches the distribution shape that we expected from both our training and test set. Although we have already established that the need for a bigger data set has been established in our model, what is interesting is what the model deemed the most significant as a result of its findings. "MeanDecreaseGini" in a Random Forest model measures the average decrease in Gini index achieved by using a specific predictor. The Gini Index measures how often a randomly chosen element in the dataset would be misclassified if it were randomly assigned to a class. It is a valuable tool for assessing the significance of predictors in classification tasks and can guide feature selection and model interpretation. Oddly enough the model deemed 'Latest Class' to have the highest impact on predicting separation timing. This makes me hypothesize a more in depth analysis of instructors may need to be conducted, or that simply put, some classes coming into the program are much stronger than others. However, the 'Schedule' attribute

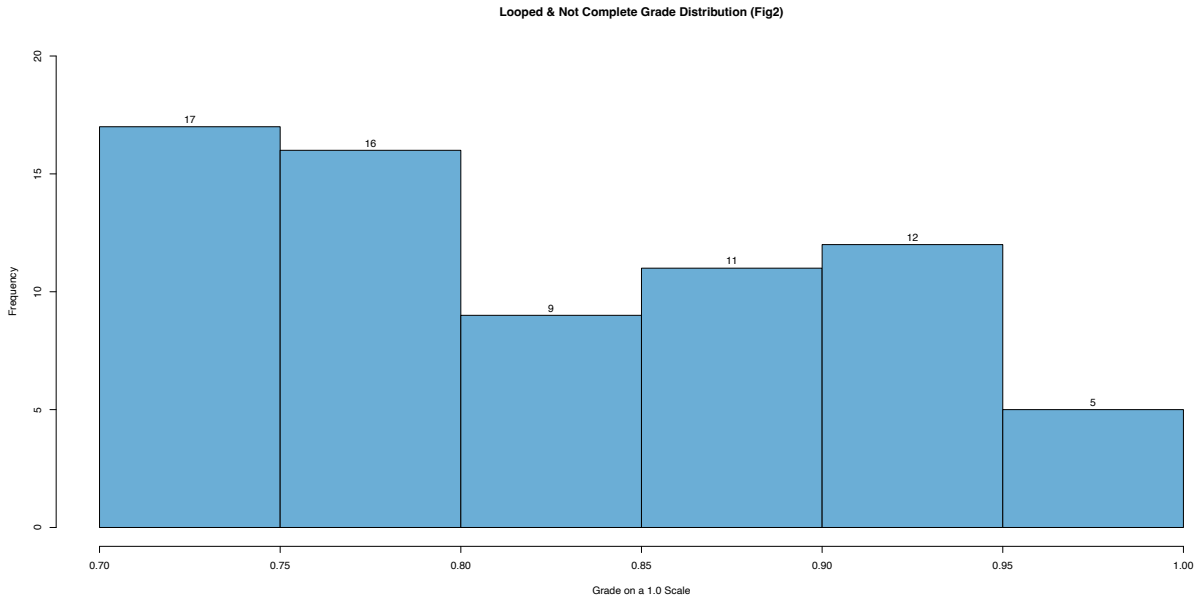
being so low, contradicts this as schedule and 'Latest Class' are directly related. For each class we have daytime and N+W sections. For example 10.1 is a daytime class in the 10.0 cohort and 10.2 in the N+W class. Classes are admitted on a paired basis. The next set after that would be the 10.3 daytime class and the 10.4 daytime class. Because the model does not know this it is interesting to see how we can extrapolate some extra insights from this.

```
# Lets look at both the 'Full' set & the set we predicted on, however, excluding the people who categor
# Creating new data frames where separation timing is listed as anything other than "Completed"
Full_OnlyLoop <- Full[Full$'Separation_Timing' != "Completed", ]
Predi_OnlyLoop <- Predictions_11_1_23[Predictions_11_1_23$'Separation_Timing' != "Completed",]
# View(Full_OnlyLoop)
# View(Predi_OnlyLoop)
```

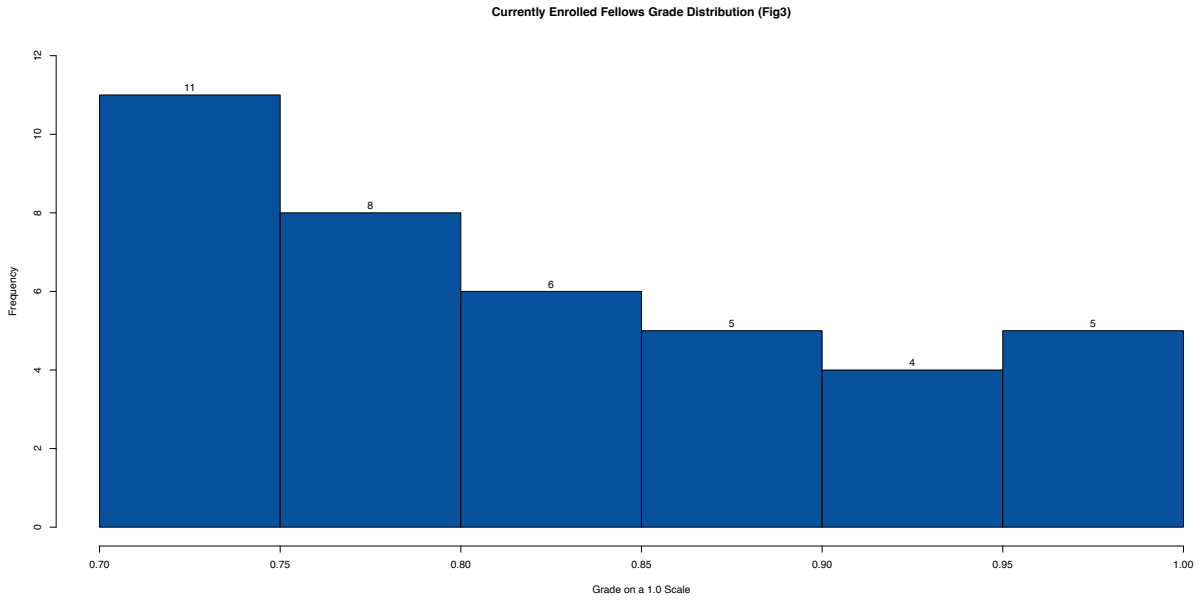
```
# Histogram on Full set we trained our model on
hist(Full$Most_Recent_Grade, main="Previous Looper Grade Distribution (Fig1)", ylim=c(0,25), col="#0851
```



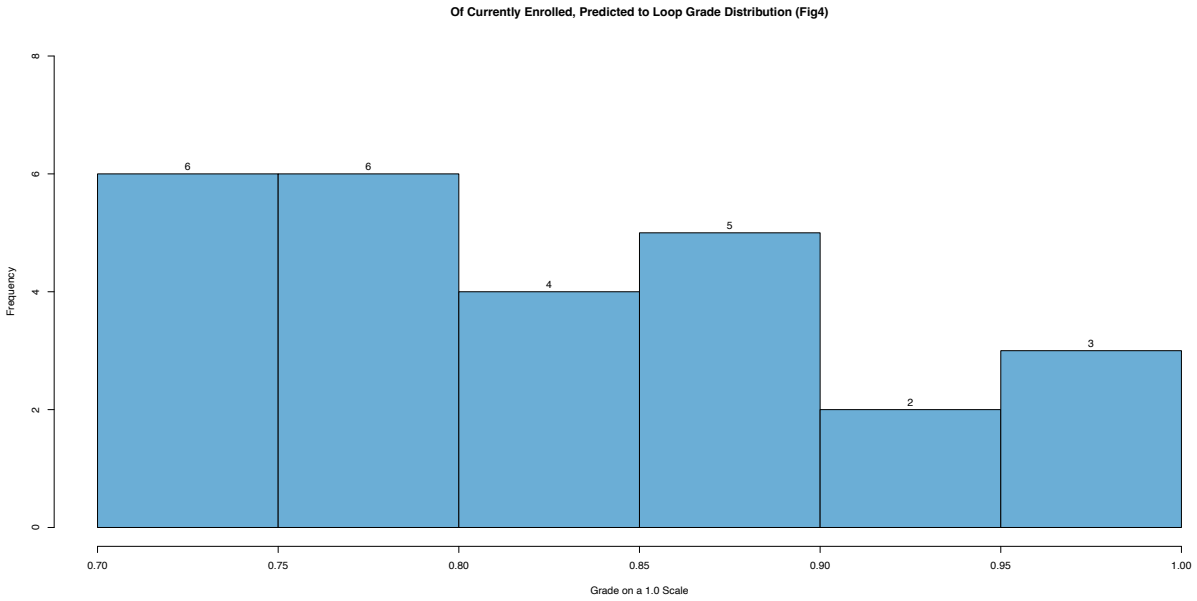

```
# Histogram on Full set - EXCLUDING those who completed the program - we trained our model on
hist(Full_OnlyLoop$Most_Recent_Grade, main="Looped & Not Complete Grade Distribution (Fig2)", ylim=c(0,
```




```r
# Histogram on our Predicted set that our model outputted
hist(Predictions_11_1_23$Most_Recent_Grade, main="Currently Enrolled Fellows Grade Distribution (Fig3)"
```



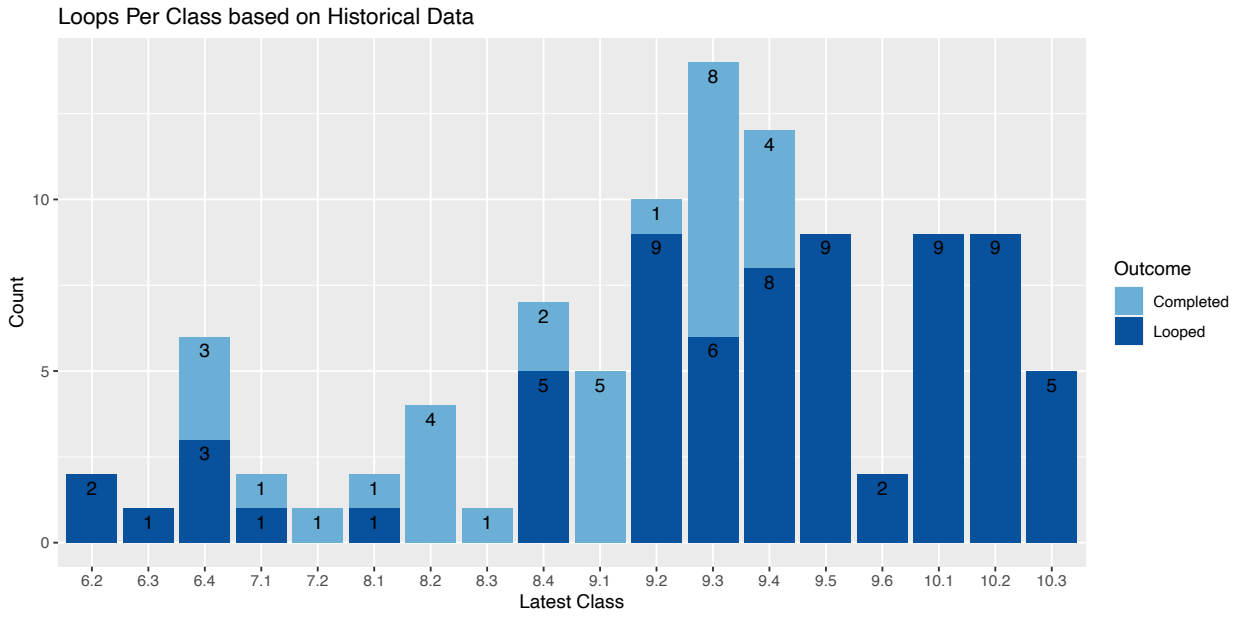

```r
# Histogram on our Predicted set - EXCLUDING those who completed the program - that our model outputted
hist(Predi_OnlyLoop$Most_Recent_Grade, main="Of Currently Enrolled, Predicted to Loop Grade Distributio
```



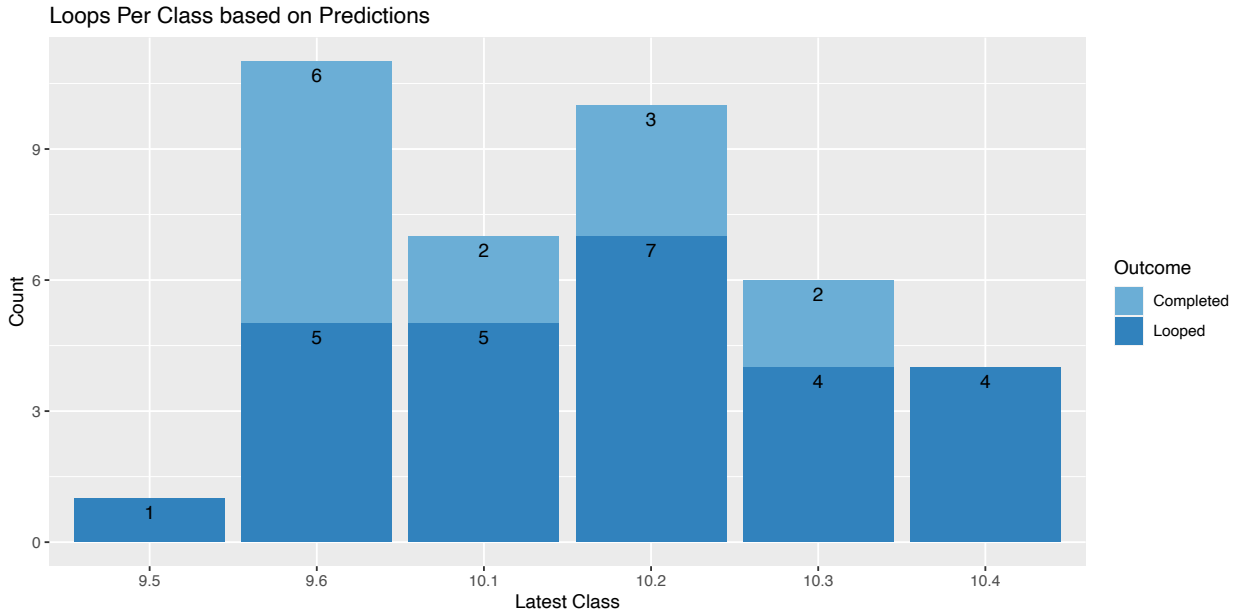
The first thing I wanted to understand was Grade significance. On the left side for both plots, we show the distribution of grades for the new data frames we constructed to show how grades are spread for fellows who were not marked as "Completed" in both the 'Full' data set and our predicted loopers. While the sample size again is too small to have the data truly representative of a positively skewed distribution, we can observe ever so slightly in both data sets that loopers generally are prone to operating at a grade level of below 85%. This difference is less apparent with the 'Currently Enrolled' distribution. While it is generally not good practice to leave gaps in our histogram like I have, I left them there in order to display the true spotty nature of the grades. For example, if you look to the higher grade end of the prediction set (Fig3), of our currently enrolled fellows, we can estimate that 5-6 fellows that are currently enrolled represent the .95 to 1.0 grade range. Of them, 3 are predicted to loop (Fig4). This is an area which will be covered in a future analysis but serves as food for thought at the moment.

```
# Create a new variable for grouping to make a stacked bar Chart
Full$Separation_Timing_Grouped <- ifelse(Full$Separation_Timing == "Completed", "Completed", "Looped")

# Plot the stacked bar chart
ggplot(Full, aes(fill = Separation_Timing_Grouped, x = Latest_Class)) +
  geom_bar(position = "stack") +
  geom_text(aes(label = ..count..), stat = "count", position = "stack", vjust = 1.5) +
  labs(title = "Loops Per Class based on Historical Data",
       x = "Latest Class",
       y = "Count",
       fill = "Outcome") +
  scale_fill_manual(values = c("Looped" = "#08519c", "Completed" = "#6baed6"))
```

```
## Warning: The dot-dot notation (‘..count..‘) was deprecated in ggplot2 3.4.0.
## i Please use ‘after_stat(count)‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```




```
# Lets do the same thing for our predictions
Predictions_11_1_23$Separation_Timing_Grouped_P <- ifelse(Predictions_11_1_23$Separation_Timing == "Comp

# Plot the stacked bar chart
ggplot(Predictions_11_1_23, aes(fill = Separation_Timing_Grouped_P, x = Latest_Class)) +
  geom_bar(position = "stack") +
  geom_text(aes(label = ..count..), stat = "count", position = "stack", vjust = 1.5) +
  labs(title = "Loops Per Class based on Predictions",
       x = "Latest Class",
       y = "Count",
       fill = "Outcome") +
  scale_fill_manual(values = c("Looped" = "#3182bd", "Completed" = "#6baed6"))
```



```r
colnames(All)
```

```
##  [1] "Latest_Class"      "Cohort"            "Latest_Status"
##  [4] "Separation_Timing" "ReapplyRec"        "Schedule"
##  [7] "Enrollments"       "Times_Looped"      "Most_Recent_Grade"
## [10] "Age"               "Gender"            "Income"
## [13] "Ethnicity"         "Education"         "Immigrant"
## [16] "Dependents"        "Is_a_Dependent"    "Core_Start"
## [19] "Cycle"             "Public_Assistance" "Emp_Status"
```

Interesting findings here support my hypothesis about the effect of a N+W schedule on fellow performance. Based on the prediction bar chart, there are a total of 10 fellows currently enrolled in the 10.2 N+W class and of those 10, 7 are predicted to loop. This indicates a 70% chance of looping, for a previous looper in this class. We cannot generalize this to the data set however. Lets calculate the chance of looping based on our current data and the predictions for those in a daytime vs N+W class.

```r
# Creating counts for solely daytime classes
dayclass_counts <- table(Full$Latest_Class)
# Extract the counts for specific classes
counts_to_sum01 <- dayclass_counts[c('6.3', '7.1', '8.1', '8.3', '9.1', '9.3', '9.5', '10.1', '10.3')]
# Sum the counts for the specific classes
total_count01 = sum(counts_to_sum01)

# Do the same for our loopers
dayclass_counts_loop <- table(Full_OnlyLoop$Latest_Class)
counts_to_sum02 <- dayclass_counts_loop[c('6.3', '7.1', '8.1', '8.3', '9.1', '9.3', '9.5', '10.1', '10.3
total_count02 = sum(counts_to_sum02)

# Calculating the percentage
DAYhist_chance_to_loop <- round((total_count02/total_count01) * 100, 1)
message <- paste("The historical probability of a daytime schedule fellow looping based off data from la
cat(message, "\n")
```

```
## The historical probability of a daytime schedule fellow looping based off data from latest enrollmen
```

```r
# Creating counts for solely N+W classes
nightclass_counts <- table(Full$Latest_Class)
# Extract the counts for specific classes
counts_to_sum03 <- nightclass_counts[c('6.2', '6.4', '7.2', '8.2', '8.4', '9.2', '9.4', '9.6', '10.2',
# Sum the counts for the specific classes
total_count03 = sum(counts_to_sum03)

# Do the same for our loopers
nightclass_counts_loop <- table(Full_OnlyLoop$Latest_Class)
counts_to_sum04 <- nightclass_counts_loop[c('6.2', '6.4', '7.2', '8.2', '8.4', '9.2', '9.4', '9.6', '10
total_count04 = sum(counts_to_sum04)

# Calculating the percentage
NWhist_chance_to_loop <- round((total_count04/total_count03) * 100, 1)
message <- paste("The historical probability of a N+W schedule fellow looping based off data from latest
cat(message, "\n")
```

The historical probability of a N+W schedule fellow looping based off data from latest enrollments is

```
table(Predictions_11_1_23$Latest_Class)
```

```
##
##  6.2  6.3  6.4  7.1  7.2  8.1  8.2  8.3  8.4  9.1  9.2  9.3  9.4  9.5  9.6 10.1
##    0    0    0    0    0    0    0    0    0    0    0    0    0    1   11    7
## 10.2 10.3 10.4
##   10    6    4
```

```
table(Predi_OnlyLoop$Latest_Class)
```

```
##
##  6.2  6.3  6.4  7.1  7.2  8.1  8.2  8.3  8.4  9.1  9.2  9.3  9.4  9.5  9.6 10.1
##    0    0    0    0    0    0    0    0    0    0    0    0    0    1    5    5
## 10.2 10.3 10.4
##    7    4    4
```

```
# Creating counts for solely daytime classes
dayclass_counts01 <- table(Predictions_11_1_23$Latest_Class)
# Extract the counts for specific classes
counts_to_sum05 <- dayclass_counts01[c('9.5', '10.1', '10.3')]
# Sum the counts for the specific classes
total_count05 = sum(counts_to_sum05)

# Do the same for our loopers
dayclass_counts_loop01 <- table(Predi_OnlyLoop$Latest_Class)
counts_to_sum06 <- dayclass_counts_loop01[c('9.5', '10.1', '10.3')]
total_count06 = sum(counts_to_sum06)

# Calculating the percentage
DAYpred_chance_to_loop <- round((total_count06/total_count05) * 100, 1)
message <- paste("The predicted probability of a daytime schedule fellow looping based off data from la
cat(message, "\n")
```

The predicted probability of a daytime schedule fellow looping based off data from latest enrollments

```
# Creating counts for solely N+W classes
nightclass_counts01 <- table(Predictions_11_1_23$Latest_Class)
# Extract the counts for specific classes
counts_to_sum07 <- nightclass_counts01[c('9.6', '10.2', '10.4')]
# Sum the counts for the specific classes
total_count07 = sum(counts_to_sum07)

# Do the same for our loopers
nightclass_counts_loop01 <- table(Predi_OnlyLoop$Latest_Class)
counts_to_sum08 <- nightclass_counts_loop01[c('9.6', '10.2', '10.4')]
total_count08 = sum(counts_to_sum08)

# Calculating the percentage
NWpred_chance_to_loop <- round((total_count08/total_count07) * 100, 1)
message <- paste("The predicted probability of a N+W schedule fellow looping based off data from latest
cat(message, "\n")
```

```
## The predicted probability of a N+W schedule fellow looping based off data from latest enrollments is
```

Final Random Forest Outtakes

The historical probability of a daytime schedule fellow looping based off data from latest enrollments is 66.7 % The historical probability of a N+W schedule fellow looping based off data from latest enrollments is 71.7 % The predicted probability of a daytime schedule fellow looping based off data from latest enrollments is 71.4 % The predicted probability of a N+W schedule fellow looping based off data from latest enrollments is 68 %

Generally, we can say that the probability of looping from within out sample ranges from 66.7% to 71.4%. The model was better suited than I had first imagined, and did indeed properly discern that schedule did not have a heavy impact on the results of our RF model. the ranges of difference between the schedules as shown in both the historical and predicted stat are too close to call this very impactful. *Refer to Class Counts post-RF model chunk.* Looking at this count of each class, noticeable things like the fact that the 5 fellows had previously looped that were past of 9.1 all completed the program, vs 5 of the 7 8.4 fellows who looped have not completed program show why the attribute was assigned such high significance.

Recommendations - This analysis could include data on instructors as well. Curriculum changes could also be recorded here to understand the different situation of each class. - Overall one significant outcome of this is the discovery that there is an over 60% of fellows who have looped before of looping once again. This begs to ask the question, are the resources that are put into these fellows worth continuing the loop program, if they are 2 out of 3 times likely going to loop? Financial data would have to be drawn here. - Recommendation to Reapply was not tested for as upon analysis of the prediction set, all fellows were listed as 0 or "No". To avoid invalid insights, this value was not investigated upon, and will need further pre-processing for the larger set.

K Means Modelling

For my initial two PCA Component Analyses of the K-Means below, understanding the data clustering done by the model was achieved by utilizing the generated columns used to make the ML model and understanding the relationships between the clusters in Tableau Prep.

```
str(Full)
```

```
## 'data.frame':    101 obs. of  24 variables:
##  $ SF_Enroll_ID             : chr  "a158Z000007WzcUQAS" "a158Z00000A7ufnQAB" "a158Z00000A7ufsQAB" "a
##  $ Fellow                   : chr  "Alex Tan" "Alexandria Brinson" "Ana Torres" "Andre Ortiz" ...
##  $ Latest_Class             : Factor w/ 19 levels "6.2","6.3","6.4",..: 12 16 17 12 6 15 18 13 14 12
##  $ Cohort                   : Factor w/ 5 levels "6","7","8","9",..: 4 5 5 4 3 4 5 4 4 4 ...
##  $ Latest_Status            : Factor w/ 3 levels "Withdrawn","Looped Back",..: 2 2 2 2 3 2 2 2 2 1 .
##  $ Separation_Timing        : Factor w/ 8 levels "Pre-Orientation",..: 4 2 3 2 8 4 2 2 3 2 ...
##  $ ReapplyRec               : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 2 2 2 2 ...
##  $ Schedule                 : Factor w/ 2 levels "Daytime","N+W": 1 1 2 1 1 2 1 2 1 1 ...
##  $ Enrollments              : int  2 1 1 1 2 2 1 1 4 2 ...
##  $ Times_Looped             : int  1 0 0 0 1 1 0 0 3 1 ...
##  $ Most_Recent_Grade        : num  0.83 0.76 0.87 0.97 0.98 0.71 0.95 0.78 0.99 0.94 ...
##  $ Age                      : int  24 26 25 25 46 33 33 25 36 32 ...
##  $ Gender                   : Factor w/ 3 levels "Man","Woman",..: 1 2 2 1 1 2 2 1 2 2 ...
##  $ Income                   : Factor w/ 4 levels "No Income","$12,500 or Below",..: 1 1 3 3 3 1 1 4 4
##  $ Ethnicity                : Factor w/ 6 levels "African American/Black",..: 2 1 5 5 3 3 1 3 3 5 ..
##  $ Education                : Factor w/ 6 levels "DNF-HS","HS/GED",..: 2 2 2 1 4 2 4 2 4 2 ...
##  $ Immigrant                : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
```

```
##  $ Dependents                : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ Is_a_Dependent            : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Core_Start                : Factor w/ 6 levels "2018","2019",..: 4 6 6 5 3 5 6 5 4 4 ...
##  $ Cycle                     : Factor w/ 4 levels "Winter","Spring",..: 3 2 2 4 1 3 3 4 1 1 ...
##  $ Public_Assistance         : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 2 1 2 ...
##  $ Emp_Status                : Factor w/ 5 levels "FT","JS","PT",..: 2 2 1 1 1 2 2 1 1 2 ...
##  $ Separation_Timing_Grouped: chr  "Looped" "Looped" "Looped" "Looped" ...
```

```
library(cluster)
```

```
##
## Attaching package: 'cluster'
```

```
## The following object is masked from 'package:maps':
##
##     votes.repub
```

```
library(ggplot2)
install.packages("factoextra")
```

```
##
## The downloaded binary packages are in
##  /var/folders/bs/_h1fsr8d6_g67rh2_kt902nm0000gn/T//RtmpmX5tHa/downloaded_packages
```

```
library(factoextra)  # For PCA visualization
```

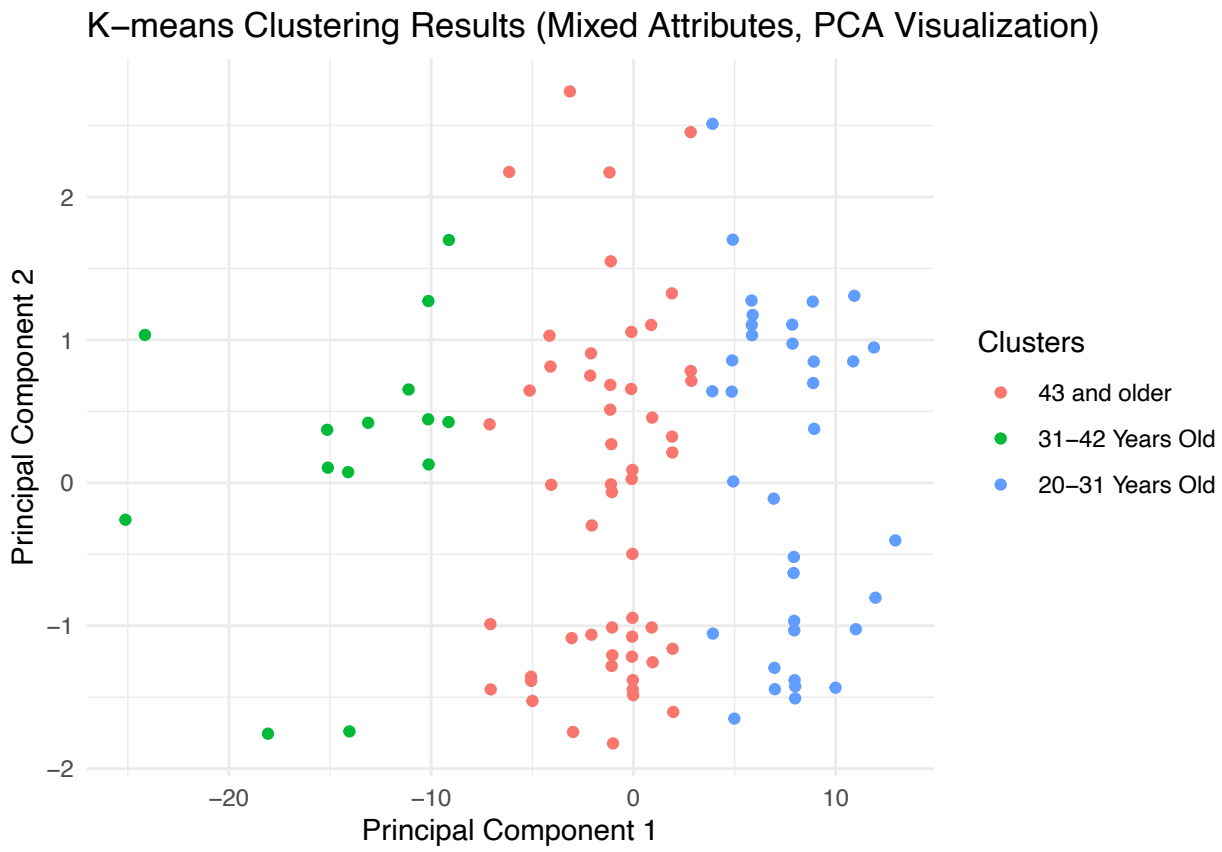
```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

Why Principal Component Analysis?

Principal Component Analysis (PCA) is a widely used statistical technique for dimensionality reduction and feature extraction in data analysis and machine learning. It is primarily employed to reduce the number of variables in a data set while preserving essential features, especially beneficial for handling high-dimensional data and enhancing computational efficiency. PCA facilitates visualization by transforming high-dimensional data into a lower-dimensional space, aiding exploratory data analysis and revealing the underlying structure. With our data set that is loaded with so many features, after observing the ineffectiveness of Random Forests to predict on some of our intermediary modules, I have come to understand that the length of our data, combined with its dimensionality may have played a role in that. Additionally, it helps in noise reduction by focusing on principal components that capture significant variability, thereby filtering out irrelevant information and highlighting crucial patterns in the data. Ideally from this I would like to find specific variables to focus on as we dig later into causation for looping.

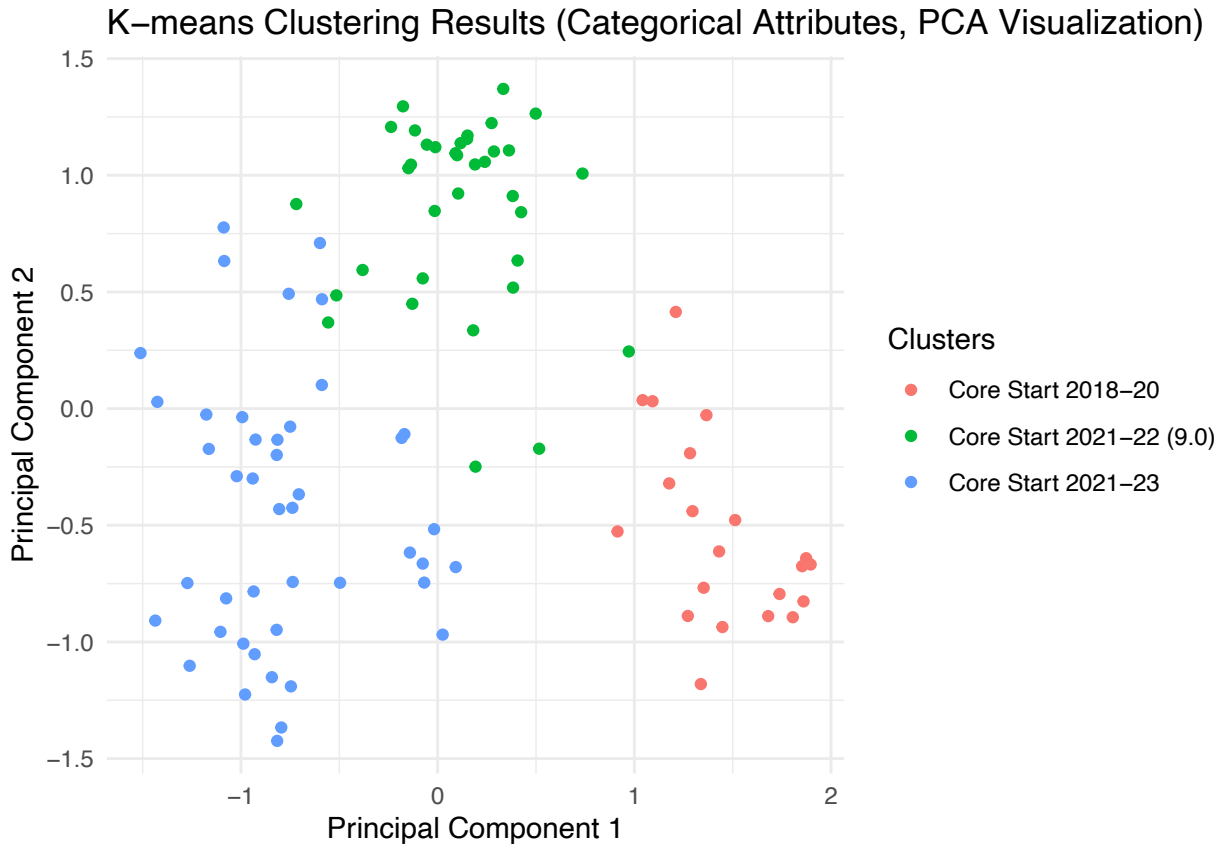
```
# Select all the variables for clustering (categorical and numeric)
mixed_data <- Full[, c('Cohort', 'Latest_Class', 'ReapplyRec', 'Schedule', 'Separation_Timing', 'Gender

# Convert categorical columns to dummy variables (one-hot encoding)
dummy_data_m<- model.matrix(~., data = mixed_data[, c('Cohort', 'Latest_Class', 'ReapplyRec', 'Schedule

# Add the numeric variables to the dummy variables
dummy_data_mi <- cbind(dummy_data_m, mixed_data[, c('Enrollments', 'Times_Looped', 'Most_Recent_Grade',
```

```r
# Perform K-means clustering on the mixed data
k <- 3
kmeans_result_mi <- kmeans(dummy_data_mi, centers = k)

# Apply PCA to reduce the dimensions of the mixed data
pca_result_mi <- prcomp(dummy_data_mi)

# Add the cluster assignment to the Full data frame
Full$Cluster_Mixed <- as.factor(kmeans_result_mi$cluster)
levels(Full$Cluster_Mixed) <- c("43 and older", "31-42 Years Old", "20-31 Years Old")

# Create a scatterplot of the PCA components, color-coded by cluster
ggplot(data = as.data.frame(pca_result_mi$x), aes(x = PC1, y = PC2, color = Full$Cluster_Mixed)) +
  geom_point() +
  labs(title = "K-means Clustering Results (Mixed Attributes, PCA Visualization)",
       x = "Principal Component 1",
       y = "Principal Component 2",
       color = "Clusters") +
  theme_minimal()
```



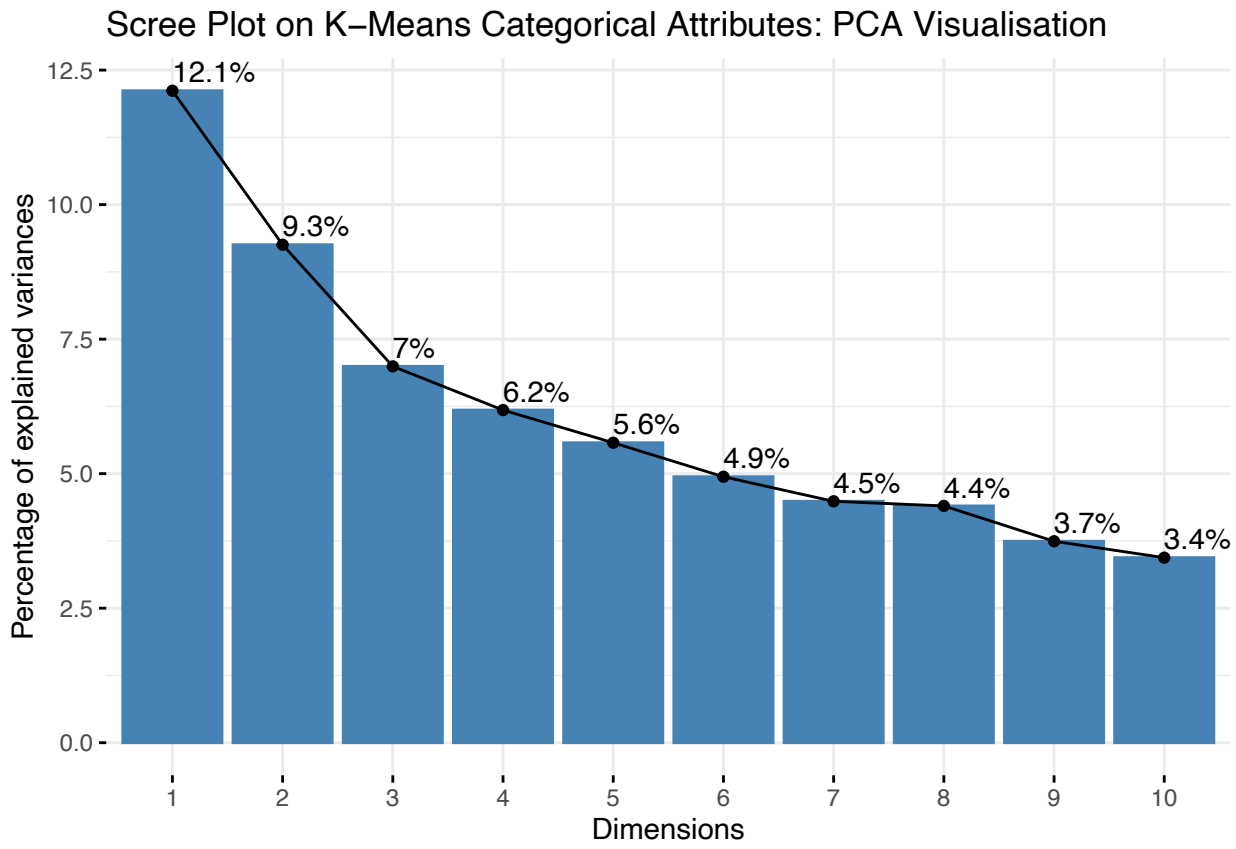

```r
# Select only the categorical variables for clustering
categorical_cols <- Full[, c('Cohort', 'Latest_Class', 'ReapplyRec', 'Schedule', 'Separation_Timing', '

# Convert categorical columns to dummy variables
dummy_data_c <- model.matrix(~., data = categorical_cols)
```

```r
# Perform K-means clustering on the categorical data
k <- 3
kmeans_result_c <- kmeans(dummy_data_c, centers = k)

# Apply PCA to reduce the dimensions of the dummy variables
pca_result_c <- prcomp(dummy_data_c)

# Add the cluster assignment to the Full data frame
Full$Cluster_Categorical <- as.factor(kmeans_result_c$cluster)
levels(Full$Cluster_Categorical) <- c("Core Start 2018-20", # Single Time Loopers who went on to comple
                                      "Core Start 2021-22 (9.0)", # Only those marked as 'Looped Back' 
                                      "Core Start 2021-23" # Only 9.0 Fellows
                                      )

# Create a scatterplot of the PCA components, color-coded by cluster
ggplot(data = as.data.frame(pca_result_c$x), aes(x = PC1, y = PC2, color = Full$Cluster_Categorical)) +
  geom_point() +
  labs(title = "K-means Clustering Results (Categorical Attributes, PCA Visualization)",
       x = "Principal Component 1",
       y = "Principal Component 2",
       color = "Clusters") +
  theme_minimal() +
  guides(color = guide_legend(label.position = "right", ncol = 1))
```

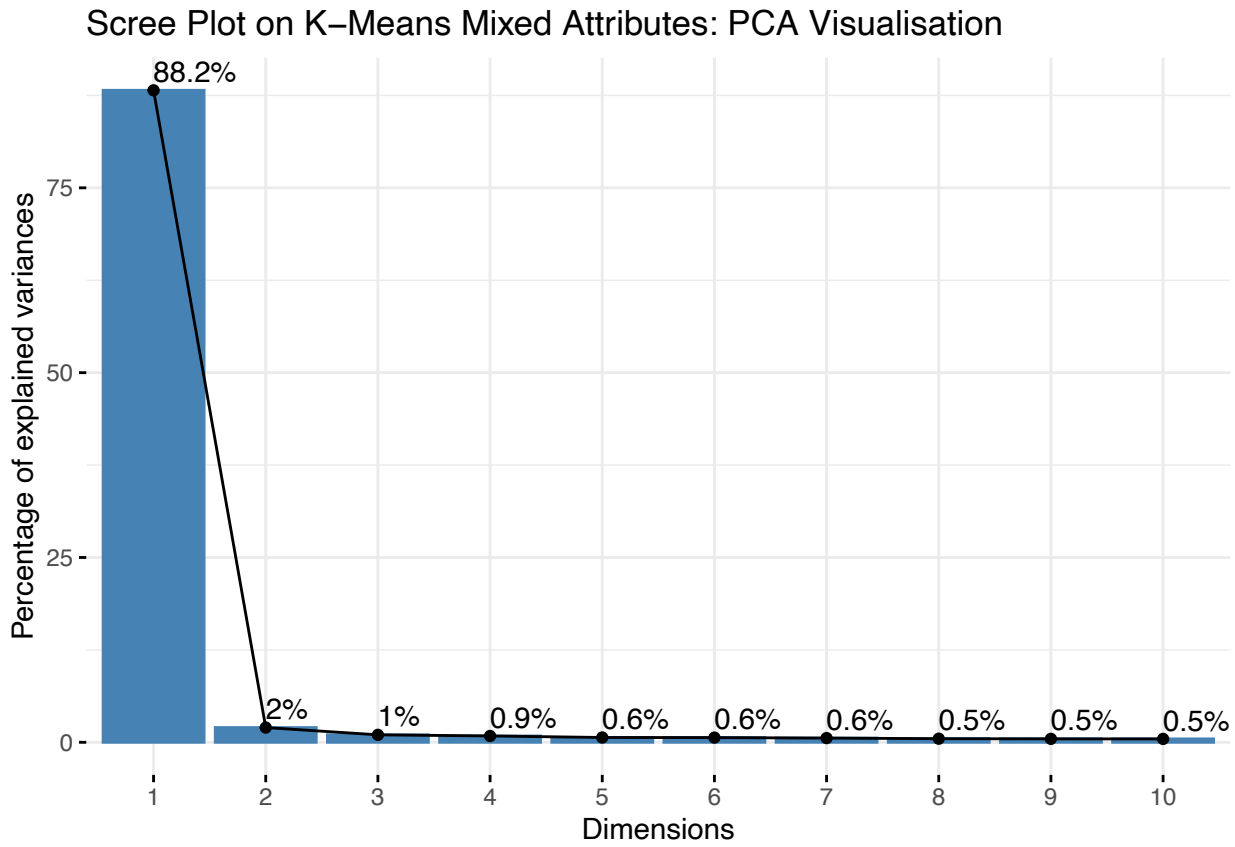


Visualization of the principal components

```r
# The first approach of the list is the scree plot. It is used to visualize the importance of each prin

# categorical attribute only PCA
fviz_eig(pca_result_c, addlabels = TRUE, main="Scree Plot on K-Means Categorical Attributes: PCA Visual
```

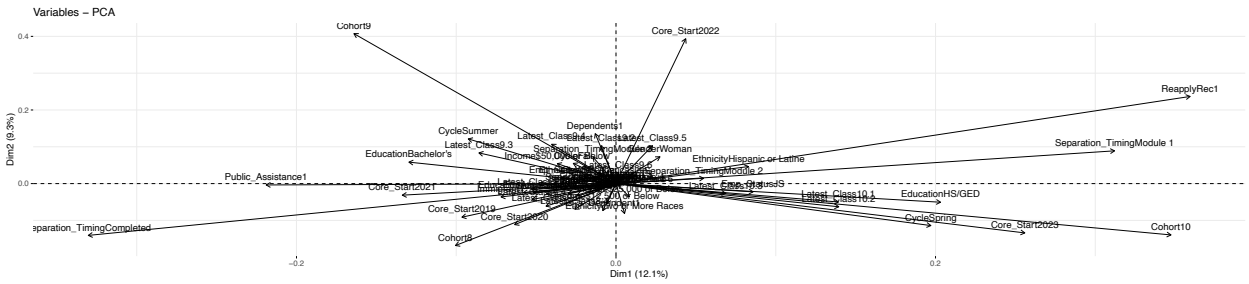



```r
# mixed attribute PCA
fviz_eig(pca_result_mi, addlabels = TRUE, main="Scree Plot on K-Means Mixed Attributes: PCA Visualisati
```

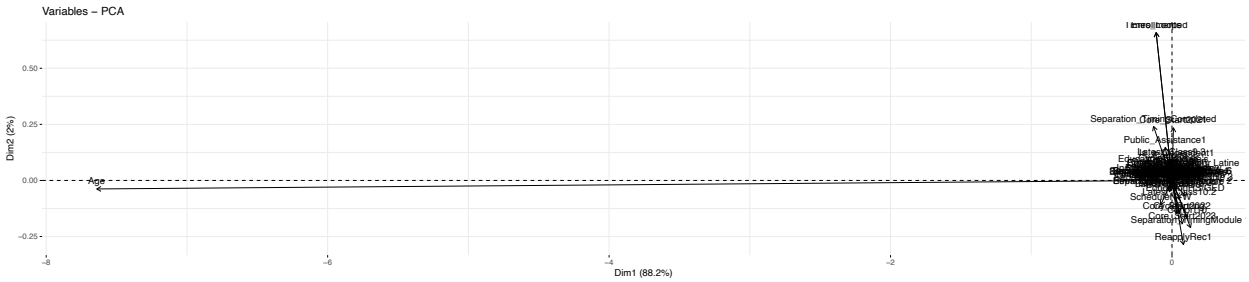


These plots shows the eigenvalues in a downward curve, in both, from highest to lowest. The first 10 components of the categorical only plot can be considered to be the most significant since they contain almost 60% of the total information of the data. For the mixed variable plot, the first component accounts for 88.2% which shows it is one of the most impactful principle components.

```
# With the biplot, it is possible to visualize the similarities and dissimilarities between the samples

# categorical attribute only PCA
fviz_pca_var(pca_result_c, col.var = "black")
```

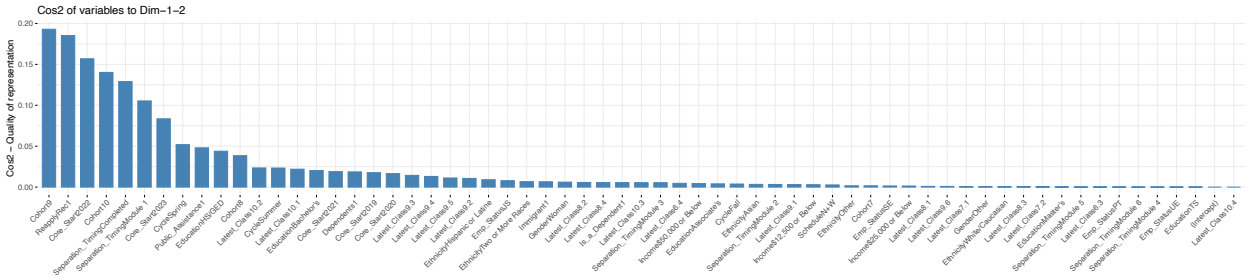



```
# mixed attribute PCA
fviz_pca_var(pca_result_mi, col.var = "black")
```

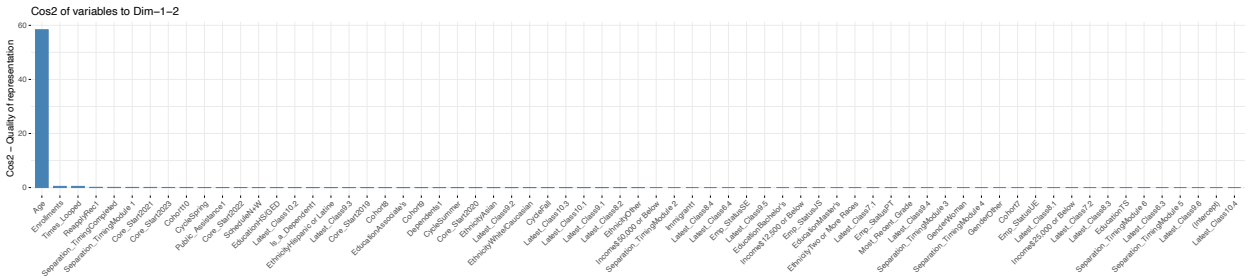


From the above, all the variables that are grouped together are positively correlated to each other, and that is the case for instance for Core_Start 2023 are correlated shown by their proximity to each other, which stands true as all 10.0 cohorts were only launched in 2023. We can observe a direct correlation between fellows with a HS/GED education and those who are unemployed but looking for a job (JS). The distance from the center represents each variable's impact as a principal component.

```
# The goal of the third visualization is to determine how much each variable is represented in a given

# A low value means that the variable is not perfectly represented by that component.
# A high value, on the other hand, means a good representation of the variable on that component.

# categorical attribute only PCA
fviz_cos2(pca_result_c, choice = "var", axes = 1:2)
```

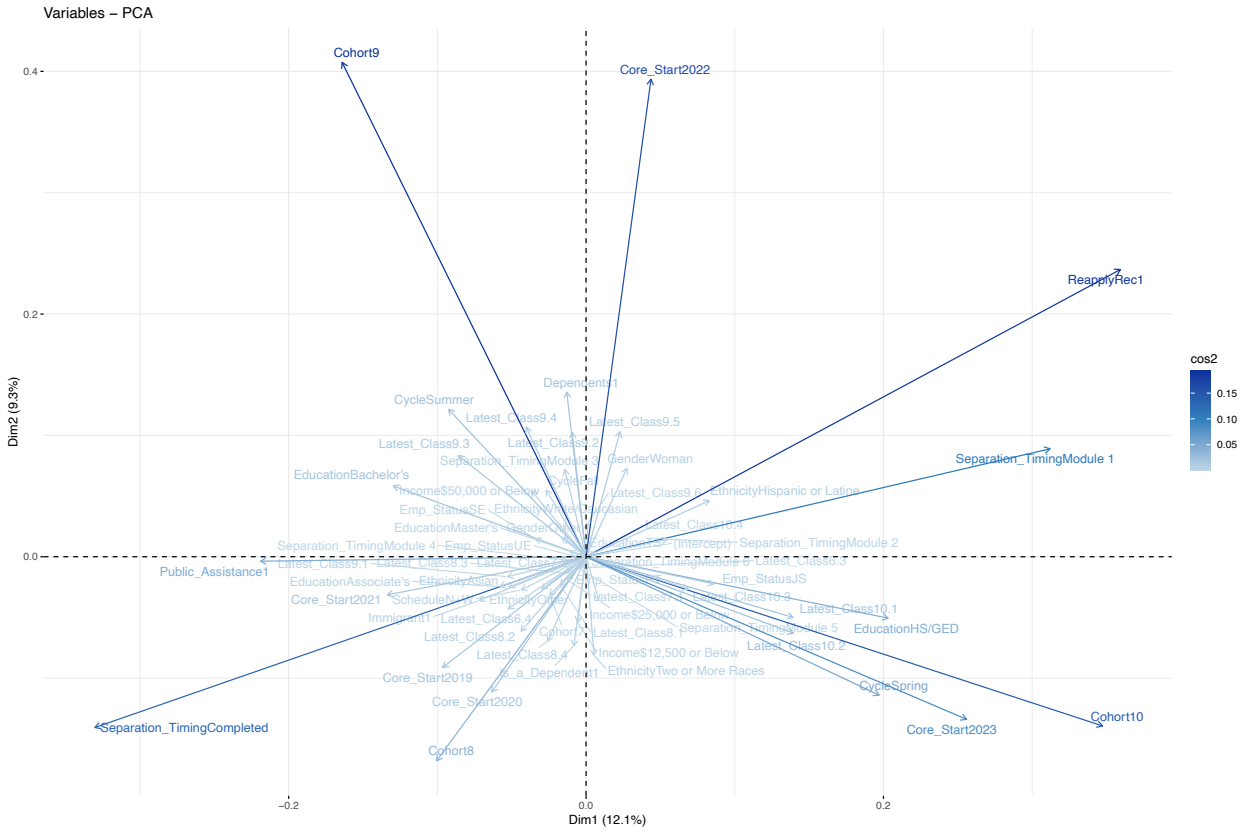



```
# mixed attribute PCA
fviz_cos2(pca_result_mi, choice = "var", axes = 1:2)
```

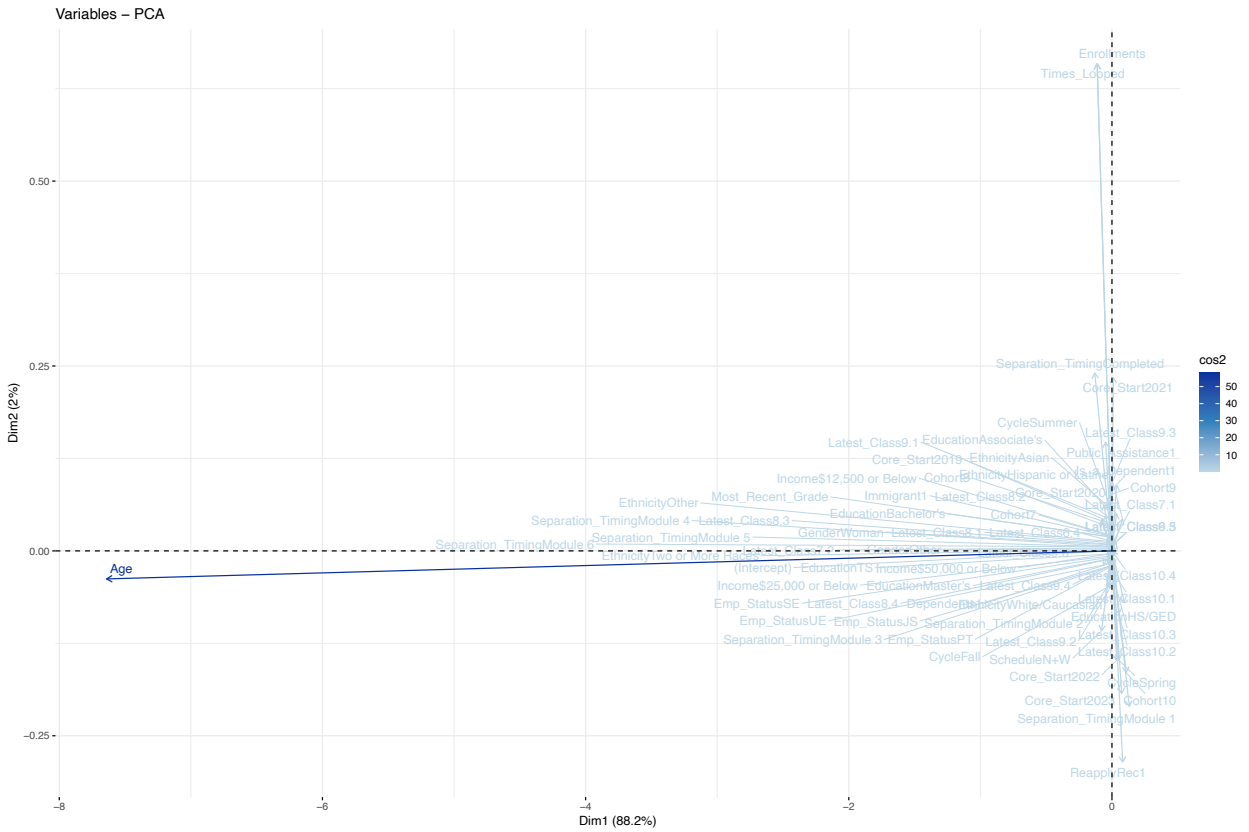


From the above, a majority of our clustered looper population when only evaluating categorical attributes were best represented by those in cohort 9.0 and those who were recommended to reapply to the program after looping. This means they were the highest contributors to principal components 1 & 2. This plot is combined with the Bi Plot below to allow us to view representation & effect on principal components, as well as explore any possible previously undiscovered correlations.

```
# Combining the square cosine and BiPlot visualisations

# categorical attribute only PCA
fviz_pca_var(pca_result_c, col.var = "cos2",
             gradient.cols = c("#bdd7e7", "#3182bd", "#08319c"),
             repel = TRUE)
```




```
# mixed attribute PCA
fviz_pca_var(pca_result_mi, col.var = "cos2",
             gradient.cols = c("#bdd7e7", "#3182bd", "#08319c"),
             repel = TRUE)
```




```
#eff3ff", "#bdd7e7", "#6baed6", "#3182bd", "#08519c", "#08306b
```

K Means Review - Categorical Attributes Only Cluster

Our data set as shown by the Biplots is far too categorical to draw out any significant conclusions. Modeled outside of R to understand the clusters using Tableau prep, from what could be observed, the most important factor in assigning the clusters was Core Start date. What this could mean, considering the previous significance of Latest_Class, is that the model has over fitted based on the distribution of fellow start time. Our data is largely skewed, as shown by the biplot on the categorical cluster, representing mostly 9.0 fellows as data prior 8.0 was very scarce, and poorly recorded. It is for this reason that it is hard to draw any meaningful conclusions out of our K-Means PCA cluster using only categorical variables. In addition to this, our factor variables failed to draw out anything significant enough in cos2 value that could help us identify possible attributes for causal modeling.

K Means Review - Mixed Attributes Cluster

From this biplot we can overwhelmingly observe the impact Age has had on this algorithm's clustering. The direction of Age as well as its Cos2 value makes it in comparison to the other attributes the most important principal component. We can see through the negative direction both on the PCA cluster and the biplot that Age is inversely proportional to Times_Looped, suggesting that older fellows are actually the ones who are less susceptible to looping out of core. Once again however, our data is still very dimensional. The purpose of PCA component analysis is to reduce data's dimensionality, and the model's inability to find any meaningful attributes in our data set (excluding numeric) is a testament to the noise in our set. It is for this reason that I believe any further scatter models would be redundant from here. Running a supervised K-NN when we already understanding data's noise will significantly skew results would be a waste. From here it is clearly a must to explore Age.

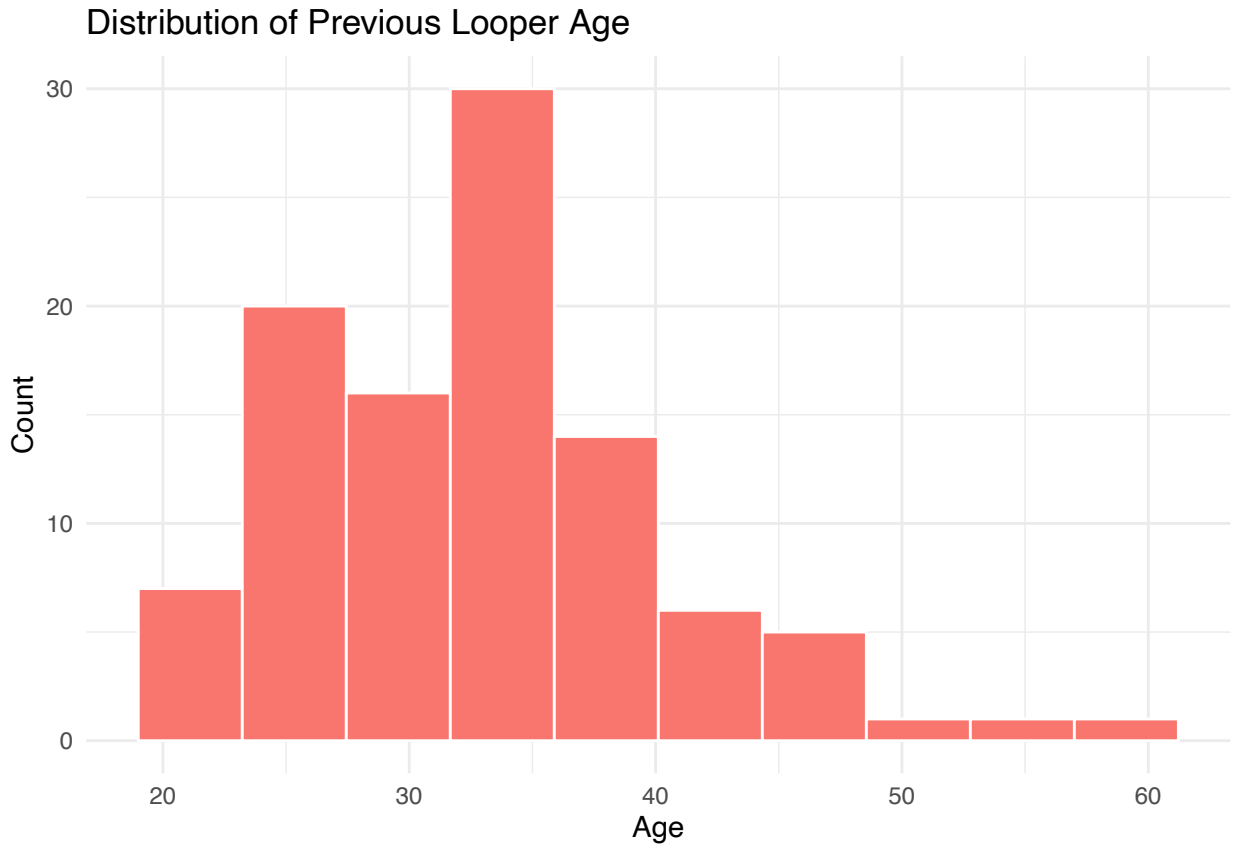
Okay, that was a lot. What do we know?

Lets run some more descriptive stats to understand the some of the outtakes of earlier.

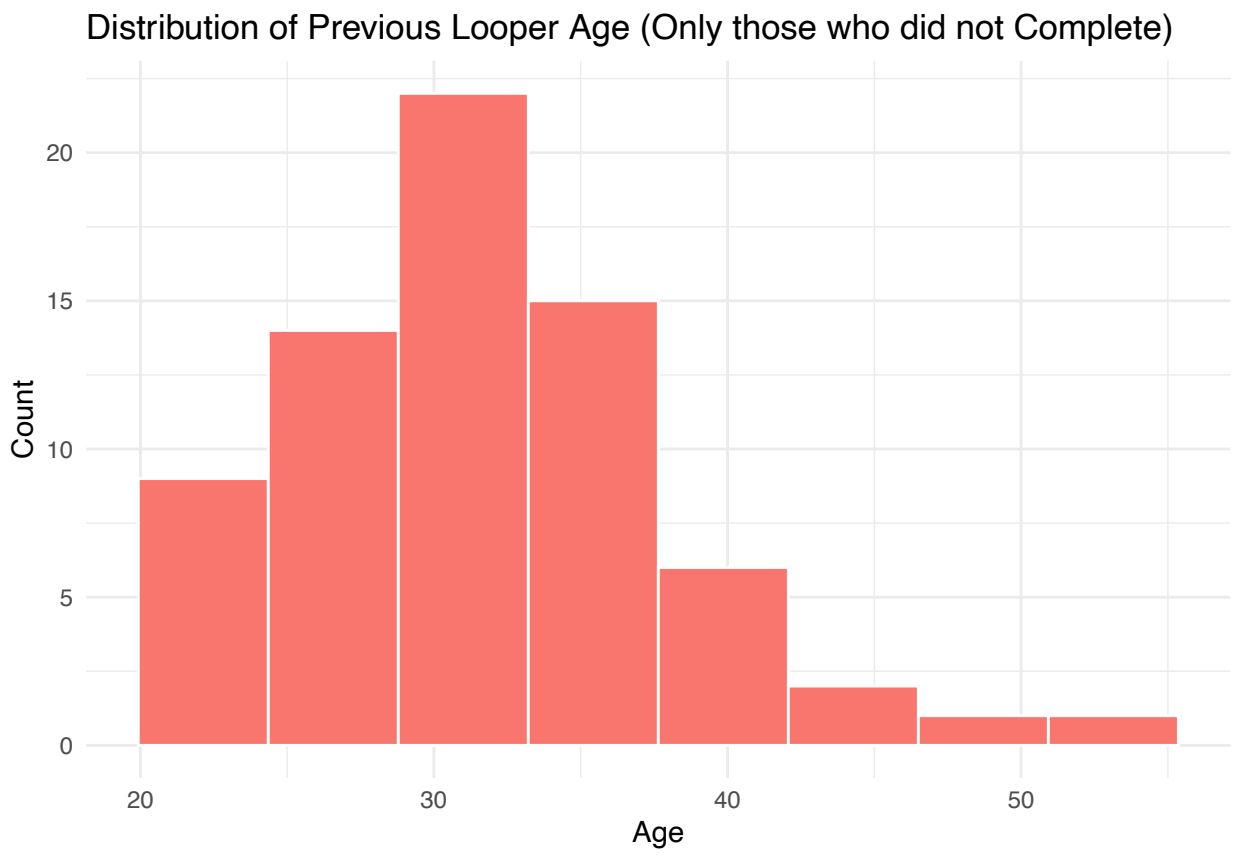
```
xcomp <- (length(Predictions_11_1_23$Separation_Timing)) - (length(Predi_OnlyLoop$Separation_Timing)) #
xloopers_and_complete <- length(Predictions_11_1_23$Separation_Timing) #39 Predictions on currently enr
will_loop <- round(((xcomp/xloopers_and_complete) * 100), 2)

x2comp <- (length(Full$Separation_Timing)) - (length(Full_OnlyLoop$Separation_Timing)) #70 predicted to
x2loopers_and_complete <- length(Full$Separation_Timing) #101 fellows that were used as training refere
will_loop2 <- round(((x2comp/x2loopers_and_complete) * 100), 2)

compavg <- paste("The historical probability of a looper completeing the program is ", will_loop2,"%, wh
cat(compavg, "\n")
```

```
## The historical probability of a looper completeing the program is  30.69 %, while the predicted proba
```

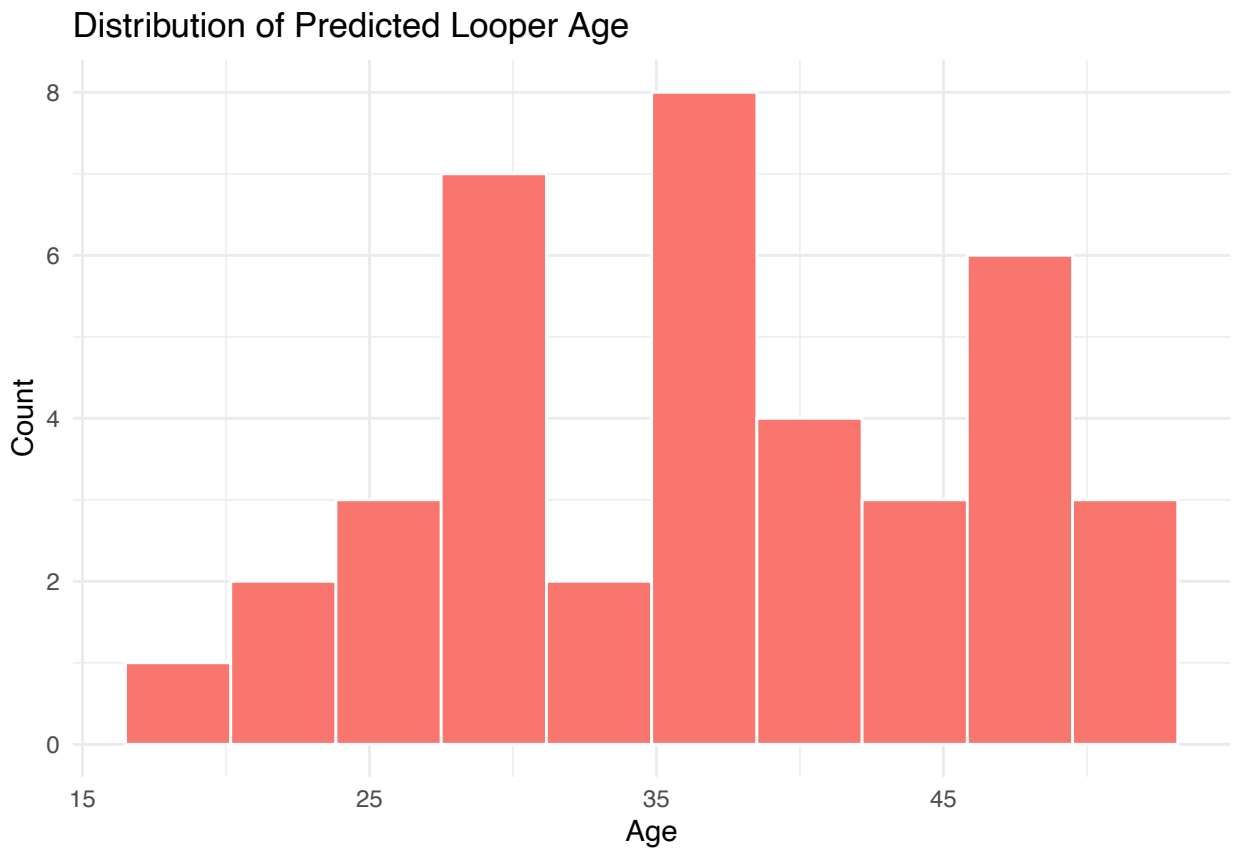
```
ggplot(data = Full, aes(x = Age, fill = "blue")) +
  geom_histogram(color = "white", bins = 10) +
  labs(title = "Distribution of Previous Looper Age",
       x = "Age",
       y = "Count") +
  theme_minimal() +
  theme(legend.position = "none")
```



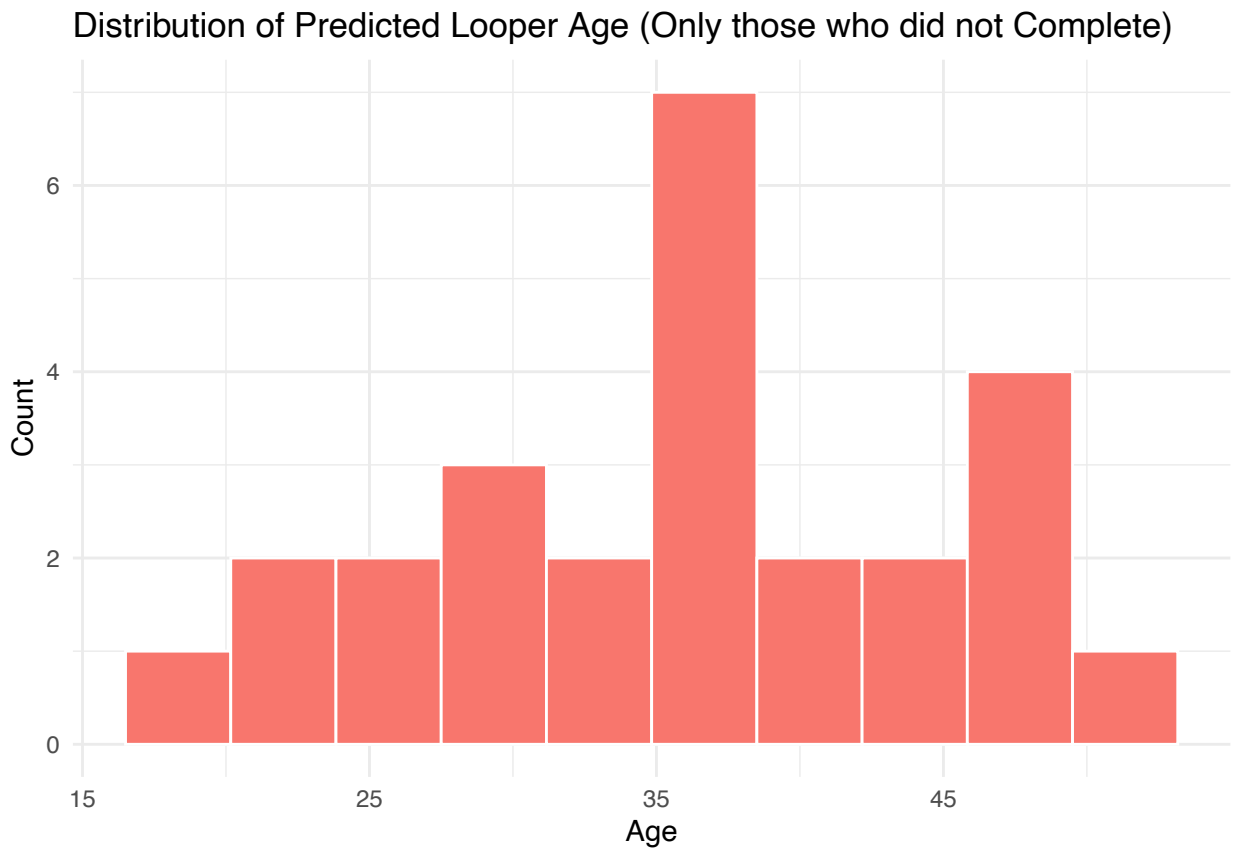
```
ggplot(data = Full_OnlyLoop, aes(x = Age, fill = "blue")) +
  geom_histogram(color = "white", bins = 8) +
  labs(title = "Distribution of Previous Looper Age (Only those who did not Complete)",
       x = "Age",
       y = "Count") +
  theme_minimal() +
  theme(legend.position = "none")
```




```
ggplot(data = Predictions_11_1_23, aes(x = Age, fill = "blue")) +
  geom_histogram(color = "white", bins = 10) +
  labs(title = "Distribution of Predicted Looper Age",
       x = "Age",
       y = "Count") +
  theme_minimal() +
  theme(legend.position = "none")
```




```
ggplot(data = Predi_OnlyLoop, aes(x = Age, fill = "blue")) +
  geom_histogram(color = "white", bins = 10) +
  labs(title = "Distribution of Predicted Looper Age (Only those who did not Complete)",
       x = "Age",
       y = "Count") +
  theme_minimal() +
  theme(legend.position = "none")
```




```
# making a data frame for all previous looper who have completed the program
Full_OnlyComp <- Full[Full$'Separation_Timing' == "Completed", ]
# making a data frame for all previous looper who have been predicted to complete the program
Predi_OnlyComp <- Predictions_11_1_23[Predictions_11_1_23$'Separation_Timing' == "Completed",]

cat("The median age of those predicted to loop from our data was", median(Predi_OnlyLoop$Age),
    "- younger than the median age of those predicted to complete the program at",
    median(Predi_OnlyComp$Age), "years old. This is also reflected in our historical data, with the medi
```

```
## The median age of those predicted to loop from our data was 36 - younger than the median age of thos
```

Lets Jump back in – Logistic Regression

As stated earlier, where the initial random forest fell short in delivering for accuracy, it opened the potential to explore another area and that was the binary likelihood of a fellow being a looper or not. Through this we hope to identify what sort of fellow profile constitutes a looper, and dig in.

Lets reformat our original data so that we can predict on Separation_Timing to either give us “Completed” or “Looped”

```
# Create a copy of the Full Data as LM_Full to conduct a binomial linear regression
V2_Full <- Full

# Set all levels to "Looped"
```

```r
V2_Full$Separation_Timing <- "Looped"

# Identify rows where Separation_Timing is "Completed" and set those to "Completed"
V2_Full$Separation_Timing[Full$Separation_Timing == "Completed"] <- "Completed"

# Convert Separation_Timing to a factor with levels "Completed" and "Looped"
V2_Full$Separation_Timing <- factor(V2_Full$Separation_Timing, levels = c("Completed", "Looped"))
levels(V2_Full$Separation_Timing)
```

```
## [1] "Completed" "Looped"
```

```r
# Dropping Rows created through PCA and other analyses that we do not need for the next part
V2_Full <- select(V2_Full, -c("SF_Enroll_ID", "Fellow", "Separation_Timing_Grouped", "Cluster_Mixed", "
str(V2_Full)
```

```
## 'data.frame':    101 obs. of  21 variables:
##  $ Latest_Class     : Factor w/ 19 levels "6.2","6.3","6.4",..: 12 16 17 12 6 15 18 13 14 12 ...
##  $ Cohort           : Factor w/ 5 levels "6","7","8","9",..: 4 5 5 4 3 4 5 4 4 4 ...
##  $ Latest_Status    : Factor w/ 3 levels "Withdrawn","Looped Back",..: 2 2 2 2 3 2 2 2 2 1 ...
##  $ Separation_Timing: Factor w/ 2 levels "Completed","Looped": 2 2 2 2 1 2 2 2 2 2 ...
##  $ ReapplyRec       : Factor w/ 2 levels "0","1": 2 2 2 1 1 2 2 2 2 2 ...
##  $ Schedule         : Factor w/ 2 levels "Daytime","N+W": 1 1 2 1 1 2 1 2 1 1 ...
##  $ Enrollments      : int  2 1 1 1 2 2 1 1 4 2 ...
##  $ Times_Looped     : int  1 0 0 0 1 1 0 0 3 1 ...
##  $ Most_Recent_Grade: num  0.83 0.76 0.87 0.97 0.98 0.71 0.95 0.78 0.99 0.94 ...
##  $ Age              : int  24 26 25 25 46 33 33 25 36 32 ...
##  $ Gender           : Factor w/ 3 levels "Man","Woman",..: 1 2 2 1 1 2 2 1 2 2 ...
##  $ Income           : Factor w/ 4 levels "No Income","$12,500 or Below",..: 1 1 3 3 3 1 1 4 4 1 ...
##  $ Ethnicity        : Factor w/ 6 levels "African American/Black",..: 2 1 5 5 3 3 1 3 3 5 ...
##  $ Education        : Factor w/ 6 levels "DNF-HS","HS/GED",..: 2 2 2 1 4 2 4 2 4 2 ...
##  $ Immigrant        : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...
##  $ Dependents       : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 1 1 1 1 ...
##  $ Is_a_Dependent   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Core_Start       : Factor w/ 6 levels "2018","2019",..: 4 6 6 5 3 5 6 5 4 4 ...
##  $ Cycle            : Factor w/ 4 levels "Winter","Spring",..: 3 2 2 4 1 3 3 4 1 1 ...
##  $ Public_Assistance: Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 2 1 2 ...
##  $ Emp_Status       : Factor w/ 5 levels "FT","JS","PT",..: 2 2 1 1 1 2 2 1 1 2 ...
```

For our 'Full' data set I have turned the dependent variable of Separation_Timing into a binary classification for the purpose of conducting a logistic regression. The previous feature engineering exercised on our dataset will be used for this model for the purpose of consistency between findings.

```r
logistic01 <- glm(Separation_Timing ~ Age, data = V2_Full, family = "binomial") # Using GLM to get deta
summary(logistic01)
```

```
##
## Call:
## glm(formula = Separation_Timing ~ Age, family = "binomial", data = V2_Full)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  3.51565    1.05788   3.323  0.00089 ***
```

```
## Age         -0.08034    0.03030  -2.651  0.00802 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 124.56  on 100  degrees of freedom
## Residual deviance: 116.80  on  99  degrees of freedom
## AIC: 120.8
##
## Number of Fisher Scoring iterations: 4
```

Interestingly, from the results of this intial model, based on estimated coefficients of Intercept (3.51565) and Age (-0.08034) we get a similar result from what we have drawn from our earlier predictions. These coefficients are used to calculate the log-odds of the event. In this case, an increase in "Age" is associated with a increase in the log-odds of "Separation_Timing" being "Completed". This supports our earlier prediction that actually younger fellows are more susceptible to not completing the program vs older fellows. However, we should consider the exclusion in this circumstance of all the other attributes.

Even though our model is showing strong enough significance codes for both attributes, and both have p-values well below 0.05 testifying for this, we also want decent effect sizes. This is something to consider as we beign to set up the model with more attributes.

Deviance: Null deviance: 124.56 on 100 degrees of freedom. This represents the deviance when no predictors are included in the model. Residual deviance: 116.80 on 99 degrees of freedom. This represents the deviance when the predictors (in this case, "Age") are included.

AIC (Akaike Information Criterion): The AIC value is 120.8. A lower AIC indicates a better-fitting model. It penalizes models with more parameters, balancing goodness of fit with model complexity.

```r
# logistic02 <- glm(Separation_Timing ~ ., data = V2_Full, family = "binomial")
# Uncommenting the above will give the results of the logistic regression on our Full Set of data which

logistic02 <- glm(Separation_Timing ~ Age + Enrollments + Most_Recent_Grade, data = V2_Full, family = "
summary(logistic02)
```

```
##
## Call:
## glm(formula = Separation_Timing ~ Age + Enrollments + Most_Recent_Grade,
##     family = "binomial", data = V2_Full)
##
## Coefficients:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       10.50323    2.80212   3.748 0.000178 ***
## Age               -0.06522    0.03238  -2.014 0.044019 *
## Enrollments       -0.99423    0.36818  -2.700 0.006926 **
## Most_Recent_Grade -6.60930    2.95511  -2.237 0.025314 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 124.56  on 100  degrees of freedom
## Residual deviance: 100.78  on  97  degrees of freedom
## AIC: 108.78
```

```
##
## Number of Fisher Scoring iterations: 4
```

```
# Coefficients from the logistic regression
coefficients <- c(-0.06522, -0.99423, -6.60930)

# Names of the variables
variable_names <- c("Age", "Enrollments", "Most_Recent_Grade")

# Calculate percentage change for each variable
percentage_change <- 100 * (1 - exp(coefficients))

# Create a data frame for better presentation
interpretation_df <- data.frame(Variable = variable_names, Percentage_Change = percentage_change)

# Print the interpretation
print(interpretation_df)
```

```
##            Variable Percentage_Change
## 1               Age          6.313867
## 2       Enrollments         62.999176
## 3 Most_Recent_Grade         99.865222
```

From what I have found, it seems that the most sensical combination of attributes that deliver a solid model, as well as a lower AIC value and residual deviance, is the combination of Age, Enrollments, and Most_Recent_Grade. Understanding why the full set of attributes bore such poor results is something we will dig into in our summary section.

Overall with this model we see solid p-values all lying below 0.05, which for a data set of this size, says the significance of these attributes put together. All of them have negative correlations with fellow completion. The coefficient estimates that can be seen with Grade having the strongest at -6.6 means that as each of these values rise, the chance of a fellow looping out of the program decreases. Essentially the above shows a table that represents 'for each 1 unit increase of *'Variable'*, the chance of completion increases by *'Percentage_Change'%*.' Or: 'For each 1 unit change in *'Variable'*, the chance of looping reduces by *'Percentage_Change'%*.'

While these values are quite extraneous for Enrollments and Most Recent Grade, on a logical front, they make sense. A looped out fellow only realistic has a chance of completion assuming they re-enroll in the program, and loigcally, one would assume fellows with higher grades, will be less likely to leave.

What is the most interesting here is the model tells us with each 1 year increase in age, the chance's of looped fellow completion drop by about 6% This is by far one of the most statistically sound predictions we have made thus far, and holds true with previous findings related to age.

```
ll_log01 <- logistic01$null.deviance/-2 # log-likelihood of a looper from Age mode by dividing null dev
ll_log02 <- logistic02$deviance/-2 # log-likelihood of a looper from multi variable model by diving res

(ll_log01 - ll_log02)/ll_log01 # Psuedo R^2, can be reffered to as overall effect size
```

```
## [1] 0.1908895
```

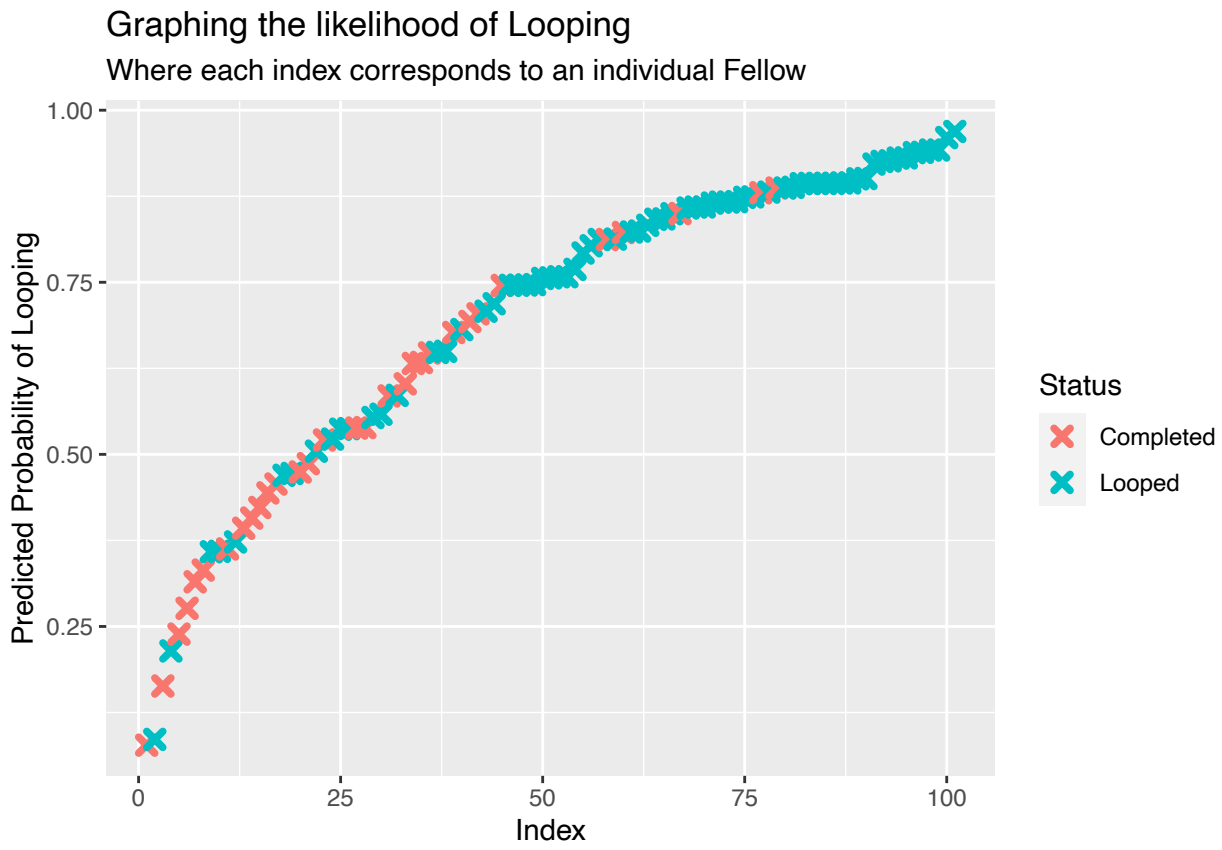
```
1 - pchisq(2*(ll_log02-ll_log01), df=(length(logistic02$coefficients)-1)) # generating a p-value to eva
```

```
## [1] 2.780907e-05
```

The first values in the above R output is our pseudo R^2 which represents how much of the variability in our response variable - separation timing, could be explained by the variables we chose. In our case 0.19 indicates our model explains about 19% of the variability in the response variable (Separation_Timing) based on the included predictors (Age, Enrollments, Most_Recent_Grade).

The second value is our chi-square value. The smaller the p-value associated with the chi-square test, the better the model fits the data. In our case, the p-value is extremely small (0.0005023981), indicating that our model with predictors is a significantly better fit than a null model without predictors.

```
# Creating a new data frame with the stored probabilities
predicted_data <- data.frame(prob_of_loop = logistic02$fitted.values, Separation_Timing=V2_Full$Separati

# Sort from low to high
predicted_data <- predicted_data[order(predicted_data$prob_of_loop, decreasing=FALSE),]

# Add column that has the rank of each sample
predicted_data$rank <- 1:nrow(predicted_data)

# Plotting the logistic regression
ggplot(data = predicted_data, aes(x=rank, y=prob_of_loop)) +
  geom_point(aes(color=Separation_Timing), alpha = 1, shape = 4, stroke = 2) +
  xlab("Index") +
  ylab("Predicted Probability of Looping") +
  labs(title = "Graphing the likelihood of Looping", color = "Status", subtitle = "Where each index cor
```



```r
# Finding the exact probabilities
predicted_data %>%
  group_by(Separation_Timing) %>%
  summarize(avg_probability_of_looping = mean(prob_of_loop))
```

```
## # A tibble: 2 x 2
##   Separation_Timing avg_probability_of_looping
##   <fct>                                 <dbl>
## 1 Completed                             0.536
## 2 Looped                                0.763
```

Logistic Regression Outtakes

From both our visualization, and our model's predictive output, the statistical significance of the attributes showed as across the board quite solid predictions were made in relation to the data's actual counterparts. The predicted values also give us some more insight into reinforcing some of the things we have gathered so far. In the above R output, our Completed fellows still had an average loop probability of 0.53. This means that even if a fellow loops back into the program after having been looped out, the highest chance they have at completion is going to be 47%. As for our Looped fellows their chance of looping overall was 0.76, making their highest chance at completion 24%.

In the context of our data set, as these Looped fellows are also marked in our set in the 'Latest_Staus' column as "Looped Back", we can contextualize this and say that for fellows who are currently looped out of the program, only 1 in 4 will go on to complete the program. Once enrolled, a looper has just under a 50% chance of completion. As a flat average between our two calculated values, any individual past their first enrollment only has a 35.5% chance of completing the program.

Naive Bayes Modeling

```r
install.packages("naivebayes")
```

```
##
## The downloaded binary packages are in
##  /var/folders/bs/_h1fsr8d6_g67rh2_kt902nm0000gn/T//RtmpmX5tHa/downloaded_packages
```

```r
library(naivebayes)
```

```
## naivebayes 0.9.7 loaded
```

```r
library(psych)
```

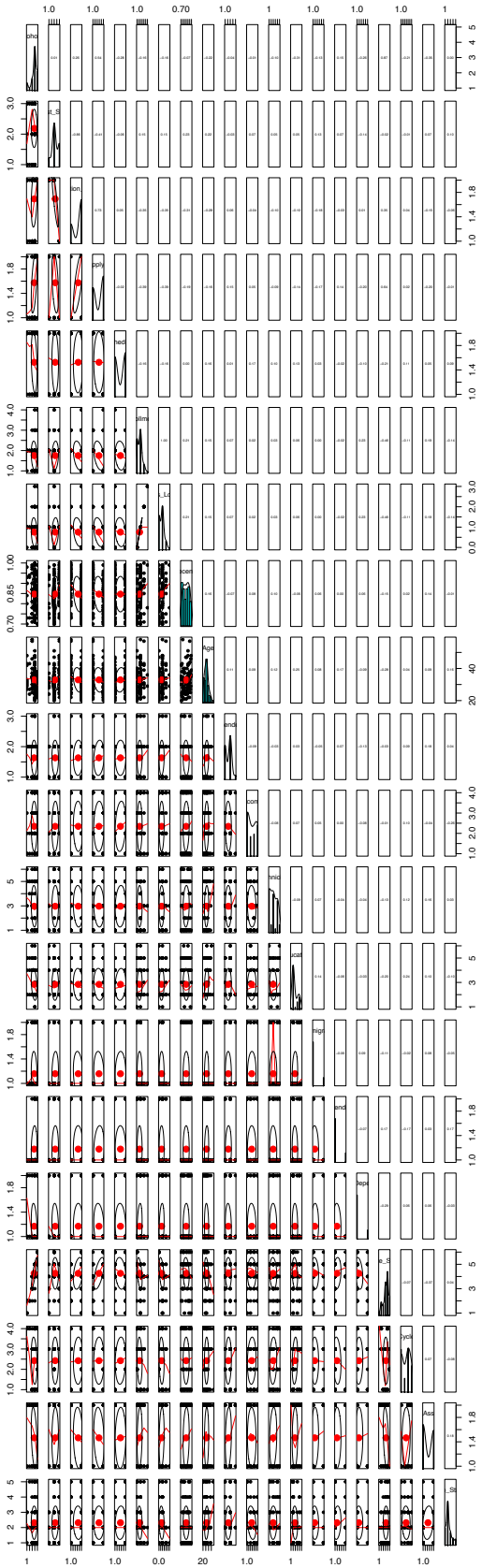
```
##
## Attaching package: 'psych'
```

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

```
## The following objects are masked from 'package:scales':
##
##     alpha, rescale

## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(caTools)
library(e1071)
```

```
pairs.panels(V2_Full[-1]) # One of the best lines of R code ever
```



```r
V3_Full <- select(V2_Full, -Core_Start, -Latest_Status, -Cohort, -ReapplyRec) # Removing due to possibl
```

Naive Bayes modeling takes the assumption that none of our independent variables are not highly correlated and that each of them have an equal impact on determining our response variable. In our data set two variables have a significant enough correlation to be excluded from our Naive Bayes model. As displayed, 'Core_Start' and 'cohort' both are correlated at 0.87 which makes sense as only certain cohorts launched during different years.

```r
set.seed(101) # Random number generation for dataset
sample = sample.split(V3_Full$Separation_Timing, SplitRatio = 0.75)
train = subset(V3_Full, sample == TRUE)
test = subset(V3_Full, sample == FALSE)

nB_model <- naiveBayes(Separation_Timing ~ ., data = train) # Training the model
pred <- predict(nB_model, test, type = "raw") # Predictions on Test Set
```

```r
nB_model
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
## Completed    Looped
## 0.3066667 0.6933333
##
## Conditional probabilities:
##            Latest_Class
## Y                   6.2        6.3        6.4        7.1        7.2        8.1
##   Completed 0.00000000 0.00000000 0.08695652 0.04347826 0.04347826 0.04347826
##   Looped    0.03846154 0.01923077 0.05769231 0.01923077 0.00000000 0.01923077
##            Latest_Class
## Y                   8.2        8.3        8.4        9.1        9.2        9.3
##   Completed 0.13043478 0.00000000 0.00000000 0.17391304 0.04347826 0.30434783
##   Looped    0.00000000 0.00000000 0.07692308 0.00000000 0.15384615 0.09615385
##            Latest_Class
## Y                   9.4        9.5        9.6       10.1       10.2       10.3
##   Completed 0.13043478 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##   Looped    0.07692308 0.11538462 0.01923077 0.07692308 0.15384615 0.07692308
##            Latest_Class
## Y                  10.4
##   Completed 0.00000000
##   Looped    0.00000000
##
##            Schedule
## Y             Daytime       N+W
##   Completed 0.5652174 0.4347826
##   Looped    0.4230769 0.5769231
##
```

```
##            Enrollments
## Y               [,1]      [,2]
##   Completed 2.130435 0.3443502
##   Looped    1.557692 0.7518203
##
##            Times_Looped
## Y                [,1]      [,2]
##   Completed 1.1304348 0.3443502
##   Looped    0.5576923 0.7518203
##
##            Most_Recent_Grade
## Y                [,1]       [,2]
##   Completed 0.8952174 0.07873255
##   Looped    0.8240385 0.08805657
##
##            Age
## Y               [,1]     [,2]
##   Completed 35.95652 8.720291
##   Looped    32.15385 6.365916
##
##            Gender
## Y                  Man      Woman      Other
##   Completed 0.43478261 0.47826087 0.08695652
##   Looped    0.40384615 0.55769231 0.03846154
##
##            Income
## Y           No Income $12,500 or Below $25,000 or Below $50,000 or Below
##   Completed 0.3043478        0.1739130        0.2173913        0.3043478
##   Looped    0.4038462        0.1538462        0.1923077        0.2500000
##
##            Ethnicity
## Y           African American/Black      Asian Hispanic or Latine      Other
##   Completed             0.21739130 0.17391304         0.17391304 0.08695652
##   Looped                0.36538462 0.01923077         0.32692308 0.00000000
##            Ethnicity
## Y           Two or More Races White/Caucasian
##   Completed        0.30434783      0.04347826
##   Looped           0.23076923      0.05769231
##
##            Education
## Y               DNF-HS     HS/GED         TS Associate's Bachelor's   Master's
##   Completed 0.00000000 0.65217391 0.00000000  0.13043478 0.21739130 0.00000000
##   Looped    0.01923077 0.67307692 0.01923077  0.09615385 0.17307692 0.01923077
##
##            Immigrant
## Y                  0         1
##   Completed 0.7391304 0.2608696
##   Looped    0.8846154 0.1153846
##
##            Dependents
## Y                  0         1
##   Completed 0.7391304 0.2608696
##   Looped    0.8269231 0.1730769
##
```

```
##            Is_a_Dependent
## Y                     0         1
##   Completed 0.8260870 0.1739130
##   Looped    0.8461538 0.1538462
##
##            Cycle
## Y              Winter    Spring    Summer      Fall
##   Completed 0.3478261 0.0000000 0.4782609 0.1739130
##   Looped    0.3076923 0.1538462 0.3076923 0.2307692
##
##            Public_Assistance
## Y                     0         1
##   Completed 0.3913043 0.6086957
##   Looped    0.5769231 0.4230769
##
##            Emp_Status
## Y                    FT         JS         PT         SE         UE
##   Completed 0.26086957 0.34782609 0.17391304 0.13043478 0.08695652
##   Looped    0.23076923 0.53846154 0.13461538 0.03846154 0.05769231
```

```
head(cbind(pred, test), 26)
```

```
##        Completed      Looped Latest_Class Separation_Timing Schedule
## 12  1.046053e-04 0.999895395          9.4            Looped      N+W
## 14  9.710749e-01 0.028925086          8.3         Completed  Daytime
## 17  8.147425e-02 0.918525750          9.3         Completed  Daytime
## 18  1.557908e-07 0.999999844          9.5            Looped  Daytime
## 21  6.928008e-02 0.930719921          9.6            Looped      N+W
## 23  9.943128e-01 0.005687155          8.4         Completed      N+W
## 27  5.214036e-09 0.999999995         10.1            Looped  Daytime
## 34  1.793910e-02 0.982060899          9.5            Looped  Daytime
## 35  2.217480e-07 0.999999778          9.5            Looped  Daytime
## 41  9.945420e-01 0.005458000          8.2         Completed      N+W
## 47  4.248179e-05 0.999957518          9.4            Looped      N+W
## 49  8.845788e-01 0.115421180          9.3            Looped  Daytime
## 52  1.800898e-03 0.998199102          8.4         Completed      N+W
## 55  3.107023e-10 1.000000000         10.2            Looped      N+W
## 61  6.000608e-01 0.399939209          9.4         Completed      N+W
## 64  8.042739e-12 1.000000000         10.1            Looped  Daytime
## 65  1.402188e-01 0.859781223          9.2            Looped      N+W
## 77  8.149195e-01 0.185080549          9.4            Looped      N+W
## 80  9.971931e-01 0.002806850          9.1         Completed  Daytime
## 84  9.835421e-01 0.016457852          6.4         Completed      N+W
## 87  1.407439e-05 0.999985926          9.4            Looped      N+W
## 88  1.013319e-03 0.998986681          8.4            Looped      N+W
## 93  1.961333e-02 0.980386669         10.1            Looped  Daytime
## 94  5.950916e-04 0.999404908         10.1            Looped  Daytime
## 99  4.081496e-07 0.999999592         10.3            Looped  Daytime
## 101 2.832231e-11 1.000000000         10.1            Looped  Daytime
##     Enrollments Times_Looped Most_Recent_Grade Age Gender           Income
## 12            1            0              0.78  34    Man $50,000 or Below
## 14            2            1              0.90  37    Man $12,500 or Below
## 17            3            2              1.00  39    Man $12,500 or Below
## 18            1            0              0.79  34  Woman        No Income
```

```
## 21           2           1             0.95  34  Woman $50,000 or Below
## 23           2           1             0.75  58    Man $50,000 or Below
## 27           1           0             0.99  36    Man $50,000 or Below
## 34           2           1             0.79  34  Other $25,000 or Below
## 35           1           0             0.85  21  Woman        No Income
## 41           2           1             0.87  34    Man        No Income
## 47           1           0             0.82  33  Woman $50,000 or Below
## 49           2           1             0.88  24    Man $25,000 or Below
## 52           2           1             0.73  30  Woman $50,000 or Below
## 55           1           0             0.92  25  Woman        No Income
## 61           2           1             0.94  22  Woman $12,500 or Below
## 64           4           3             0.78  29  Woman        No Income
## 65           2           1             0.80  21    Man        No Income
## 77           2           1             0.94  37  Woman $25,000 or Below
## 80           2           1             0.83  24    Man        No Income
## 84           2           1             0.90  48  Woman        No Income
## 87           1           0             0.79  32  Woman        No Income
## 88           2           1             0.78  34  Woman $12,500 or Below
## 93           2           1             0.87  28  Woman $25,000 or Below
## 94           3           2             0.75  22    Man $50,000 or Below
## 99           1           0             0.91  22    Man        No Income
## 101          1           0             0.79  33  Woman $12,500 or Below
##                  Ethnicity   Education Immigrant Dependents Is_a_Dependent
## 12         White/Caucasian  Bachelor's         0          0              0
## 14       Two or More Races Associate's         1          0              0
## 17      Hispanic or Latine          TS         0          0              0
## 18  African American/Black      HS/GED         0          1              0
## 21  African American/Black Associate's         0          0              0
## 23                   Other      HS/GED         0          0              0
## 27      Hispanic or Latine      HS/GED         0          0              0
## 34       Two or More Races      HS/GED         0          0              0
## 35      Hispanic or Latine      HS/GED         1          0              0
## 41       Two or More Races      HS/GED         0          0              0
## 47       Two or More Races  Bachelor's         0          0              0
## 49  African American/Black      HS/GED         0          0              1
## 52  African American/Black Associate's         0          0              0
## 55      Hispanic or Latine      HS/GED         0          0              0
## 61  African American/Black      HS/GED         0          0              0
## 64      Hispanic or Latine      HS/GED         0          0              1
## 65                   Asian      DNF-HS         0          0              0
## 77      Hispanic or Latine      HS/GED         0          1              0
## 80  African American/Black  Bachelor's         0          0              0
## 84         White/Caucasian  Bachelor's         1          0              1
## 87  African American/Black      HS/GED         0          0              0
## 88  African American/Black    Master's         1          0              0
## 93       Two or More Races      HS/GED         0          0              0
## 94      Hispanic or Latine      HS/GED         0          1              1
## 99      Hispanic or Latine      HS/GED         0          0              1
## 101 African American/Black          TS         0          0              0
##      Cycle Public_Assistance Emp_Status
## 12    Fall                 1         SE
## 14  Summer                 1         JS
## 17  Summer                 1         JS
## 18  Winter                 1         JS
```

```
## 21    Fall                0         PT
## 23  Winter                0         SE
## 27  Spring                0         JS
## 34    Fall                1         JS
## 35  Winter                0         JS
## 41  Winter                1         JS
## 47    Fall                1         JS
## 49  Summer                0         PT
## 52  Winter                0         FT
## 55  Spring                0         JS
## 61  Summer                0         JS
## 64  Winter                0         JS
## 65  Winter                1         UE
## 77  Winter                0         PT
## 80  Winter                0         SE
## 84  Summer                1         JS
## 87    Fall                1         UE
## 88  Winter                1         PT
## 93  Summer                0         JS
## 94  Summer                0         FT
## 99  Summer                0         JS
## 101 Spring                0         JS
```

```r
# Extract probabilities for the "Looped" class
pred_probs_looped <- pred[, "Looped"]

# Convert probabilities to class predictions (0 or 1)
predicted <- as.numeric(pred_probs_looped > 0.5)

# Ensure predicted has the same length as the actual outcome
predicted <- predicted[1:length(test$Separation_Timing)]

# Create a confusion matrix
conf_matrix <- table(predicted, test$Separation_Timing)

# Calculate misclassification rate (Test)
misclassification_rate <- 1 - sum(diag(conf_matrix)) / sum(conf_matrix)
print("Misclassification Rate for Test Set:")
```

```
## [1] "Misclassification Rate for Test Set:"
```

```r
print(misclassification_rate)
```

```
## [1] 0.1538462
```

```r
# Make predictions on the training set
pred_train <- predict(nB_model, train, type = "raw")

# Extract probabilities for the "Looped" class
pred_probs_looped_train <- pred_train[, "Looped"]

# Convert probabilities to class predictions (0 or 1)
```

```r
predicted_train <- as.numeric(pred_probs_looped_train > 0.5)

# Ensure predicted_train has the same length as the actual outcome
predicted_train <- predicted_train[1:length(train$Separation_Timing)]

# Create a confusion matrix for the training set
conf_matrix_train <- table(predicted_train, train$Separation_Timing)

# Calculate misclassifications rate (Train)
misclassification_rate_train <- 1 - sum(diag(conf_matrix_train)) / sum(conf_matrix_train)
print("Misclassification Rate for Training Set:")
```

```
## [1] "Misclassification Rate for Training Set:"
```

```r
print(misclassification_rate_train)
```

```
## [1] 0.1066667
```

Naive Bayes Training Model - Looking Pretty Good!

From our Training model we are gathering insights that display, and reinforce what we have been able to cover so far. Overall the probability of a looper completing the program is 30% post their first enrollment. We also see that the average age of completion is nearly 36 years of age where as the average age of a looper is 32 years of age. This reinforces our previous findings that loopers, contrary to logical thinking, are actually younger individuals.

There is clear bias towards certain classes as some classes exclusively have only probabilities of fellows either looping or completing - however, we understand why this is from previous visualizations as some classes' loopers simply did not finish, even if that class had only a single looper.

There is an interesting insight drawn from the Schedule confusion Matrix that suggests fellows who go on to complete core are coming from the Daytime class with a 56% probability, and fellows who go onto loop are coming from N+W classes with a 57% probability.

I would like to trust this model due to its high propensity of impressive predictions shown by the head displayed predicting with nearly 90% accuracy. This is further reflected in the 15-10% misclassification Rates between the training and test set. While this is optimistic, lets try this on our full data set and then make predictions on the set with the generated separation timings from the Random Forest we had made previously.

Naive Bayes-Full Data & Predictions

```r
nB_model01 <- naiveBayes(Separation_Timing ~ ., data = V3_Full)
nB_model01
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
```

```
## A-priori probabilities:
## Y
## Completed    Looped
## 0.3069307 0.6930693
##
## Conditional probabilities:
##            Latest_Class
## Y                   6.2        6.3        6.4        7.1        7.2        8.1
##   Completed 0.00000000 0.00000000 0.09677419 0.03225806 0.03225806 0.03225806
##   Looped    0.02857143 0.01428571 0.04285714 0.01428571 0.00000000 0.01428571
##            Latest_Class
## Y                   8.2        8.3        8.4        9.1        9.2        9.3
##   Completed 0.12903226 0.03225806 0.06451613 0.16129032 0.03225806 0.25806452
##   Looped    0.00000000 0.00000000 0.07142857 0.00000000 0.12857143 0.08571429
##            Latest_Class
## Y                   9.4        9.5        9.6       10.1       10.2       10.3
##   Completed 0.12903226 0.00000000 0.00000000 0.00000000 0.00000000 0.00000000
##   Looped    0.11428571 0.12857143 0.02857143 0.12857143 0.12857143 0.07142857
##            Latest_Class
## Y                  10.4
##   Completed 0.00000000
##   Looped    0.00000000
##
##            Schedule
## Y             Daytime       N+W
##   Completed 0.5161290 0.4838710
##   Looped    0.4571429 0.5428571
##
##            Enrollments
## Y               [,1]      [,2]
##   Completed 2.129032 0.3407771
##   Looped    1.585714 0.7707113
##
##            Times_Looped
## Y                 [,1]      [,2]
##   Completed 1.1290323 0.3407771
##   Looped    0.5857143 0.7707113
##
##            Most_Recent_Grade
## Y                  [,1]       [,2]
##   Completed 0.8874194 0.08181148
##   Looped    0.8290000 0.08423827
##
##            Age
## Y               [,1]     [,2]
##   Completed 36.09677 9.470497
##   Looped    31.50000 6.247898
##
##            Gender
## Y                   Man      Woman      Other
##   Completed 0.48387097 0.45161290 0.06451613
##   Looped    0.38571429 0.57142857 0.04285714
##
##            Income
```

```
## Y           No Income $12,500 or Below $25,000 or Below $50,000 or Below
##   Completed 0.3225806        0.2258065        0.1612903        0.2903226
##   Looped    0.4000000        0.1428571        0.2000000        0.2571429
##
##            Ethnicity
## Y           African American/Black      Asian Hispanic or Latine      Other
##   Completed             0.25806452 0.12903226         0.16129032 0.09677419
##   Looped                0.35714286 0.02857143         0.34285714 0.00000000
##            Ethnicity
## Y           Two or More Races White/Caucasian
##   Completed        0.29032258      0.06451613
##   Looped           0.21428571      0.05714286
##
##            Education
## Y               DNF-HS     HS/GED         TS Associate's Bachelor's   Master's
##   Completed 0.00000000 0.58064516 0.03225806  0.16129032 0.22580645 0.00000000
##   Looped    0.02857143 0.67142857 0.02857143  0.08571429 0.15714286 0.02857143
##
##            Immigrant
## Y                   0         1
##   Completed 0.7419355 0.2580645
##   Looped    0.8857143 0.1142857
##
##            Dependents
## Y                   0         1
##   Completed 0.8064516 0.1935484
##   Looped    0.8285714 0.1714286
##
##            Is_a_Dependent
## Y                   0         1
##   Completed 0.8387097 0.1612903
##   Looped    0.8285714 0.1714286
##
##            Cycle
## Y              Winter    Spring    Summer      Fall
##   Completed 0.3870968 0.0000000 0.4838710 0.1290323
##   Looped    0.3142857 0.1571429 0.2857143 0.2428571
##
##            Public_Assistance
## Y                   0         1
##   Completed 0.4193548 0.5806452
##   Looped    0.5857143 0.4142857
##
##            Emp_Status
## Y                  FT         JS         PT         SE         UE
##   Completed 0.22580645 0.41935484 0.12903226 0.16129032 0.06451613
##   Looped    0.18571429 0.54285714 0.15714286 0.04285714 0.07142857
```

```r
# Make predictions on the full data
pred_full <- predict(nB_model01, V3_Full, type = "raw")

# Extract probabilities for the "Looped" class
pred_probs_looped_full <- pred_full[, "Looped"]
```

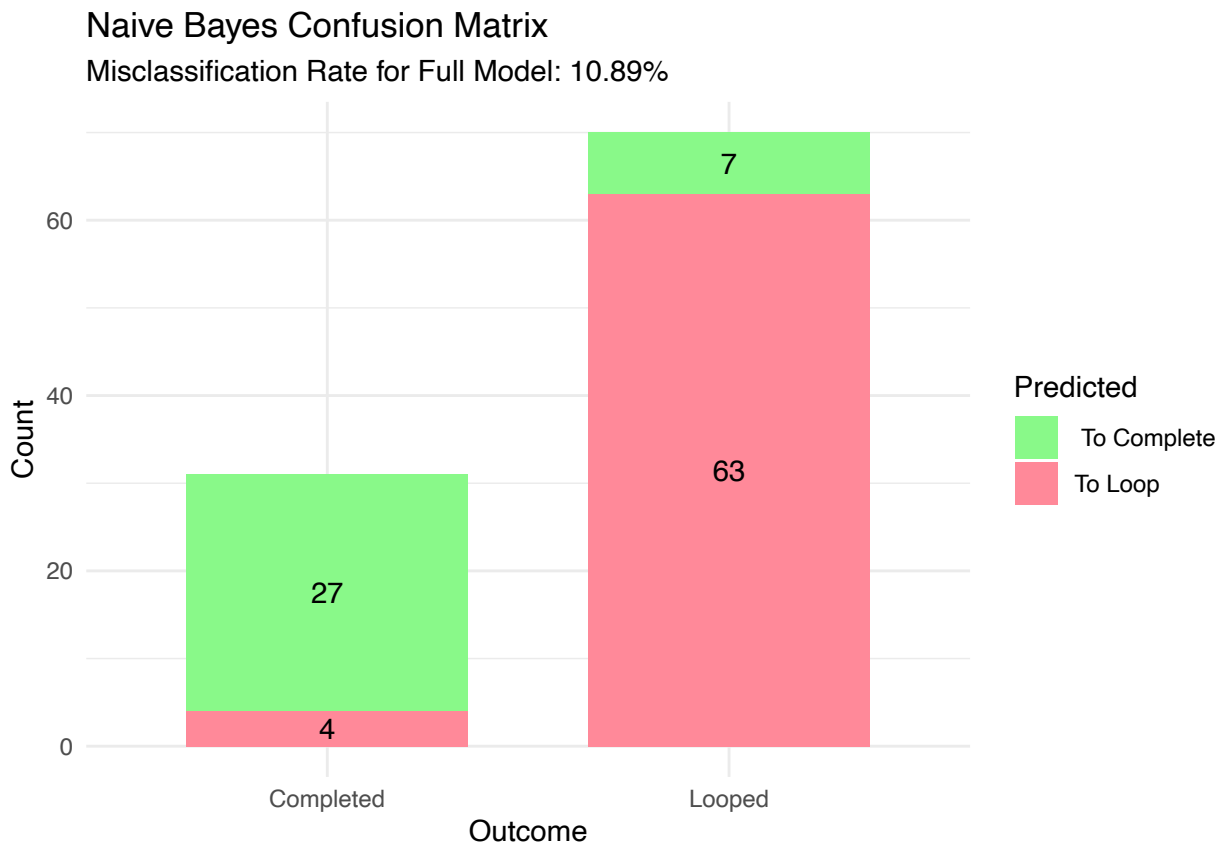
```r
# Convert probabilities to class predictions (0 or 1)
predicted_full <- as.numeric(pred_probs_looped_full > 0.5)

# Ensure predicted_full has the same length as the actual outcome
predicted_full <- predicted_full[1:length(V3_Full$Separation_Timing)]

# Create a confusion matrix for the full data
conf_matrix_full <- table(predicted_full, V3_Full$Separation_Timing)

# Calculate misclassification rate for the full model
misclassification_rate_full <- 1 - sum(diag(conf_matrix_full)) / sum(conf_matrix_full)

# Print the misclassification rate for the full model
print("Misclassification Rate for Full Model:")
```

```
## [1] "Misclassification Rate for Full Model:"
```

```r
print(misclassification_rate_full)
```

```
## [1] 0.1089109
```

```r
# Convert the confusion matrix to a data frame
conf_matrix_df <- as.data.frame(as.table(conf_matrix_full))

# Rename the columns for clarity
colnames(conf_matrix_df) <- c("Predicted", "Actual", "Count")

# Create the ggplot
ggplot(conf_matrix_df, aes(x = Actual, y = Count, fill = Predicted)) +
  geom_col(position = "stack", width = 0.7) +
  labs(title = "Naive Bayes Confusion Matrix", x = "Outcome", y = "Count", subtitle = "Misclassificatio
  theme_minimal() +
  geom_text(aes(label = Count), position = position_stack(vjust = 0.5), show.legend = FALSE) + scale_fi
    values = c("0" = "#89F989", "1" = "#FF8999"),
    breaks = c("0", "1"),
    labels = c(" To Complete", "To Loop")
  )
```



Naive Bayes Full Set Model - Consistency in Outcomes Our Full Naive Bayes Model looks to provide promising results with only an estimated 11% misclassification rate, this is by far our most solid model, and looks to provide the best insights with regards to our data structure. It reflects what has been previously found of the data set. The prominence of schedule while existing here is slight as previously observed, but still evident. In this model their is significant evidence suggesting that Women are more likely to loop than men. With this model there is a 57% chance of a woman looping as opposed to only a 38% chance for men, something we haven't been able to see with our other models.

Due to the noise in our data set as we have touched upon through the PCA analysis, in addition to this model's low misclassification rate, I will take this results as statistically significant to our purpose of prediction. The noise in our data set can be quelled with the Naive Bayes model's assumption that all independent variables have the same effect rate on our response variable. Because of this, I believe it is fair to say in our data set with such low correlations between most of our variables that we can attribute a higher chance of looping assuming an individual checks all of the following boxes:

-Woman -N+W Schedule -Under 35 years of age

Using the generated output of our previous 'Predictions_11_1_23' Set, Lets match the Bayes Probabilities from the model trained on a full data set to cross verify predictions between models.

```
# Removing columns that could alter predictions as they were previously excluded
V3_Pred <- select(Predictions_11_1_23, -SF_Enroll_ID, -Cohort, -Latest_Status, -Separation_Timing, -Rea

# Renaming
V3_Pred <- V3_Pred %>% rename(Separation_Timing = Separation_Timing_Grouped_P)

# Reordering
V3_Pred <- V3_Pred[, c("Fellow", "Latest_Class", "Separation_Timing", "Schedule", "Enrollments", "Times
```

```r
# Factorizing "V3_Pred" column in the "Separation_Timing" data frame
V3_Pred$Separation_Timing <- factor(V3_Pred$Separation_Timing, levels = c("Completed", "Looped"))

# Printing the structure of V3_Pred
str(V3_Pred)
```

```
##  $ Times_Looped     : int  1 2 2 1 1 1 2 1 2 3 ...
##  $ Most_Recent_Grade: num  0.8 0.74 0.73 0.85 0.74 0.73 0.75 0.8 0.76 0.93 ...
##  $ Age              : int  30 30 46 31 25 36 36 48 34 45 ...
##  $ Gender           : Factor w/ 3 levels "Man","Woman",..: 1 2 2 2 2 2 2 1 2 2 ...
##  $ Income           : Factor w/ 4 levels "No Income","$12,500 or Below",..: 2 4 4 4 3 1 3 2 3 2 ...
##  $ Ethnicity        : Factor w/ 7 levels "African American/Black",..: 1 3 4 1 6 6 1 3 1 1 ...
##  $ Education        : Factor w/ 6 levels "DNF-HS","HS/GED",..: 2 2 1 2 3 4 4 5 2 2 ...
##  $ Immigrant        : Factor w/ 2 levels "0","1": 1 1 2 1 1 2 1 1 1 1 ...
##  $ Dependents       : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 1 2 ...
##  $ Is_a_Dependent   : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Core_Start       : Factor w/ 6 levels "2018","2019",..: 6 4 4 5 5 5 5 5 4 4 ...
##  $ Cycle            : Factor w/ 4 levels "Winter","Spring",..: 2 1 1 4 3 1 3 4 3 1 ...
##  $ Public_Assistance: Factor w/ 2 levels "0","1": 1 1 2 2 1 2 1 1 2 2 ...
##  $ Emp_Status       : Factor w/ 5 levels "FT","JS","PT",..: 2 4 4 3 3 2 2 3 1 2 ...
```

```r
# View(V3_Pred)
```

```r
# Making predictions
pred_11.26.23 <- predict(nB_model01, V3_Pred, type = "raw")

# Saving Our CSV with Loop Predictions
looper_probabilities <- cbind(pred_11.26.23, V3_Pred)
write.csv(looper_probabilities, "Final_Predictions.csv", row.names = FALSE)

# A combined file showing Predictions on whether or not a fellow separates, with their estimated probab
Final_Predictions <- read.csv("Final_Predictions.csv")
head(Final_Predictions)
```

```
##   Schedule Enrollments Times_Looped Most_Recent_Grade Age Gender
## 1      N+W           2            1              0.80  30    Man
## 2  Daytime           3            2              0.74  30  Woman
## 3      N+W           3            2              0.73  46  Woman
## 4      N+W           2            1              0.85  31  Woman
```

```
## 5      N+W           2            1              0.74  25  Woman
## 6      N+W           2            1              0.73  36  Woman
##              Income              Ethnicity   Education Immigrant Dependents
## 1 $12,500 or Below African American/Black      HS/GED         0          0
## 2 $50,000 or Below     Hispanic or Latine      HS/GED         0          0
## 3 $50,000 or Below         Middle Eastern      DNF-HS         1          0
## 4 $50,000 or Below African American/Black      HS/GED         0          0
## 5 $25,000 or Below      Two or More Races          TS         0          0
## 6        No Income      Two or More Races Associate's         1          1
##   Is_a_Dependent Core_Start  Cycle Public_Assistance Emp_Status
## 1              0       2023 Spring                 0         JS
## 2              0       2021 Winter                 0         SE
## 3              0       2021 Winter                 1         SE
## 4              0       2022   Fall                 1         PT
## 5              0       2022 Summer                 0         PT
## 6              0       2022 Winter                 1         JS
```

The above output, similar to the original prediction CSV, should be cross validated with the actual churn results at a certain point in the future. This will inform our decision making understanding what statistical models will best fit this problem.

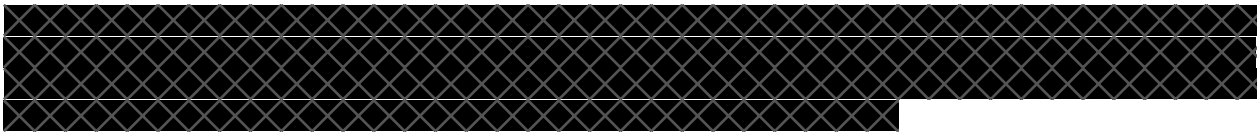
Summary

What do we now know?

- Identify Overall Looper Churn Rate: The chance of a fellow completing the program after having previously looped is **30%**, once re-enrolled, fellow has about a **47%** chance of completion, and of those looped out **25%** will go on to complete core at some future point.
- Categorically identify the characteristics that describe our 'looper profile': Based largely on Bayesian statistics, an 'ideal looper' is *a woman in a nights and weekend class, under the age of 30.*
- Understand the characteristic causation of what causes loopers to exit the program: *Age, Most Recent Grade, and Number of Enrollments* were the largest causal factors in determining the probability of an individual looping. It was these characteristics that drove the discovery of a 57% chance of a Woman looping out over Men at 38%, and a 54% chance of a looper being from a N+W class as opposed to the daytime class at 45%. Age was the most tangible of these causal factors as **with each 1 unit increase an Age, the chances a fellow loops out reduces by 6%**, there is a positive correlation between Age and Probability of Completion.

Reflections & Further Research

As mentioned before, this set is a very small with a large amount of factored attributes. In essence, our data was very detailed but lacked the size in some cases to provide truly meaningful insights, delivering predictions for when during Core fellows would loop. Hence many of our more solid conclusions have been drawn from modeling done after we turned our response variable into binary, understanding who completes core and who loops. From this we have come up with a reasonable profile to understand a target audience to direct more focused support.

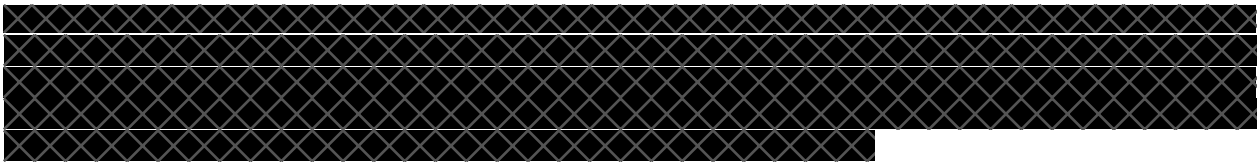


To improve the results of our analysis, a bigger set should be prepared for a training model. With the variability, a set of more than 500 would be suitable for a second stage analysis. This can be achieved by following up this analysis with a replicated process examining fellows on each enrollment. This set includes only any historical looper's latest enrollment, which is why we had fellows who are currently enrolled. As mentioned before, the predictions of both the Naive Bayes model and the Random Forest Classifer should be evaluated with actual results at a later date, to determine the better method of achieving accurate results with a larger set.

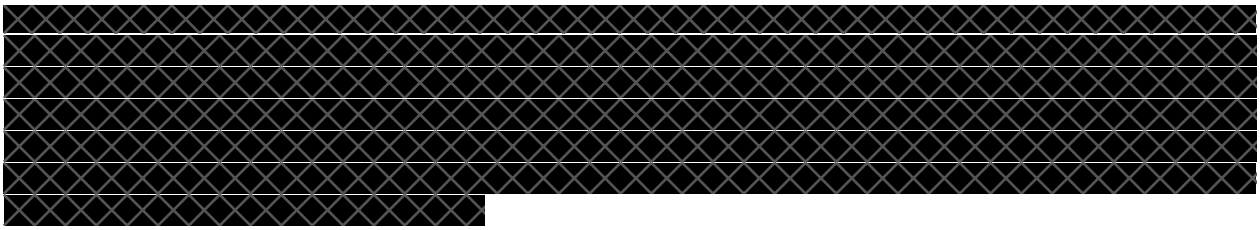
One of the reasons why I believe so much noise existed in our data was the already inevitable skew of our distribution. Pursuit is a program which bases its admission criteria based on need. When we already know that most of our fellows are coming from minority backgrounds, unemployed, making less that $45k annually etc. It is hard to see these demographic features making any meaningful impact on modeling. However, their inclusion is still important, but it is something to note about the dimensionality of this data set. Perhaps a later analysis will utilize reduced column set, with more continuous variables, but that is to be seen. In future the inclusion of module grades, and instructors will help in this analysis. Our loopers coming from specific classes in higher proportion of others could be due to instructor changes, curriculum edits and a slew of other things not included in this data set. Individual module performance related to loops would be helpful in terms of modifying curriculum content to better facilitate fellow success.

Recommendations

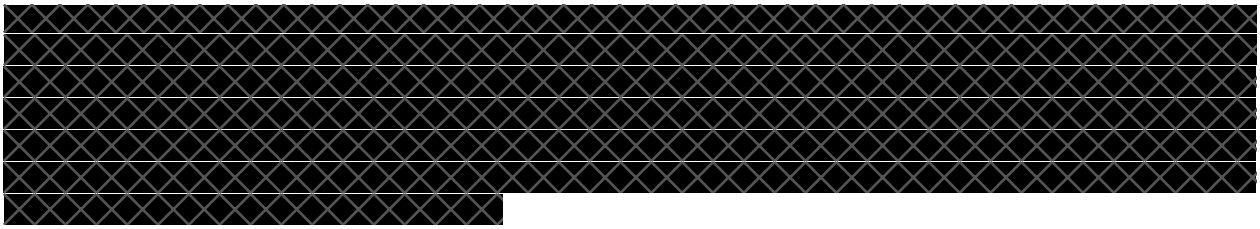
Data Restructure



Core Evaluation of Module 1 Content



Data Documentation & Process



[redacted]

[redacted]

[redacted]

[redacted]

```
cat[redacted]ncome from a single fellow's [redacted]to:", full_co
```

[redacted]

- [redacted]

[redacted]

[redacted]