Data606-Project: Education & Career Success

Gautham Nagaraj UID: 30273439, Angela Li (30024371), Prichelle Lal (30261130)

2025-06-08

*Introduction:* In today’s competitive job market, understanding the pathways from education to career success is more crucial than ever. Students invest significant time and resources into their academic journey with the hope that strong educational performance will lead to favorable career outcomes.

However, the relationship between education and career achievement is influenced by a multitude of factors, including soft skills, internships, networking, and field of study. This dataset captures the educational backgrounds, skillsets, and career results of 5,000 individuals, providing a rich foundation to explore these connections. By analyzing variables such as GPA, SAT scores, university rankings, certifications, and job offers, we aim to build predictive models and uncover the key drivers of starting salaries, promotions, and overall career satisfaction.

Our goal is to apply sampling, regression estimation, and categorical data analysis to uncover patterns and relationships within the education and career success dataset. Using methods such as stratified sampling, ratio estimation, logistic regression, and classification, we aim to identify key predictors of career outcomes such as job offers, starting salary, and satisfaction. These analyses will allow us to make informed inferences, build predictive models, and ultimately gain a deeper understanding of how educational and experiential factors influence career trajectories.

The dataset was synthetically generated using real-world education and career trends.

education\_data = read.csv('https://raw.githubusercontent.com/Gautham-Nagaraj/Data606-Project/refs/heads/main/education\_career\_success.csv')  
head(education\_data,10)

## Student\_ID Age Gender High\_School\_GPA SAT\_Score University\_Ranking  
## 1 S00001 24 Male 3.58 1052 291  
## 2 S00002 21 Other 2.52 1211 112  
## 3 S00003 28 Female 3.42 1193 715  
## 4 S00004 25 Male 2.43 1497 170  
## 5 S00005 22 Male 2.08 1012 599  
## 6 S00006 24 Male 2.40 1600 631  
## 7 S00007 27 Male 2.36 1011 610  
## 8 S00008 20 Male 2.68 1074 240  
## 9 S00009 24 Male 2.84 1201 337  
## 10 S00010 28 Male 3.02 1415 138  
## University\_GPA Field\_of\_Study Internships\_Completed Projects\_Completed  
## 1 3.96 Arts 3 7  
## 2 3.63 Law 4 7  
## 3 2.63 Medicine 4 8  
## 4 2.81 Computer Science 3 9  
## 5 2.48 Engineering 4 6  
## 6 3.78 Law 2 3  
## 7 3.83 Computer Science 0 1  
## 8 2.84 Computer Science 1 5  
## 9 3.31 Business 2 3  
## 10 2.33 Computer Science 1 5  
## Certifications Soft\_Skills\_Score Networking\_Score Job\_Offers Starting\_Salary  
## 1 2 9 8 5 27200  
## 2 3 8 1 4 25000  
## 3 1 1 9 0 42400  
## 4 1 10 6 1 57400  
## 5 4 10 9 4 47600  
## 6 2 2 2 1 68400  
## 7 3 3 3 2 55500  
## 8 5 5 1 2 38000  
## 9 0 5 5 2 68900  
## 10 3 10 2 0 58900  
## Career\_Satisfaction Years\_to\_Promotion Current\_Job\_Level Work\_Life\_Balance  
## 1 4 5 Entry 7  
## 2 1 1 Mid 7  
## 3 9 3 Entry 7  
## 4 7 5 Mid 5  
## 5 9 5 Entry 2  
## 6 9 2 Entry 8  
## 7 7 4 Mid 3  
## 8 2 3 Entry 3  
## 9 2 2 Entry 2  
## 10 4 2 Senior 2  
## Entrepreneurship  
## 1 No  
## 2 No  
## 3 No  
## 4 No  
## 5 No  
## 6 Yes  
## 7 No  
## 8 No  
## 9 No  
## 10 No

Based on the above table, we need to ensure that categorical variables are handled appropriately when fitting the model. Some of the variables that appear to be continuous but are categorical are:

**University Ranking:** The difference between rank 1 and 2 might not be the same as between rank 999 and 1000 in terms of quality or impact. Thus, it is better to treat this variable as categorical rather than continuous.

As the number of levels will be very high if we factor this variable, it is better to label it as ‘very high ranked’, ‘high ranked’, ‘low rank’.

**Number of Internships Completed(0-4):** While counts can sometimes be treated as continuous if the range is large, here, each number represents a distinct level of internship experience, making it more appropriate as an ordinal categorical variable.

**Projects\_Completed(0-9):** Similar to number of internships, it is more appropriate to treat this as categorical variable.

**Certifications**, **Soft-Skill Score**, **Networking Score** - These variables are also to be treated as categorical due to lower counts.

Now to label the university ranking:

library(dplyr)

##   
## Attaching package: 'dplyr'

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

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

education\_data <- education\_data %>%  
 mutate(  
 University\_Ranking\_Category = case\_when(  
 University\_Ranking >= 1 & University\_Ranking <= 250 ~ "High ranked",  
 University\_Ranking > 250 & University\_Ranking <= 500 ~ "Moderately Ranked",  
 University\_Ranking > 500 & University\_Ranking <= 750 ~ "Low ranked",  
 University\_Ranking > 750 ~ "Very low ranked"  
 )  
 )  
head(education\_data,10)

## Student\_ID Age Gender High\_School\_GPA SAT\_Score University\_Ranking  
## 1 S00001 24 Male 3.58 1052 291  
## 2 S00002 21 Other 2.52 1211 112  
## 3 S00003 28 Female 3.42 1193 715  
## 4 S00004 25 Male 2.43 1497 170  
## 5 S00005 22 Male 2.08 1012 599  
## 6 S00006 24 Male 2.40 1600 631  
## 7 S00007 27 Male 2.36 1011 610  
## 8 S00008 20 Male 2.68 1074 240  
## 9 S00009 24 Male 2.84 1201 337  
## 10 S00010 28 Male 3.02 1415 138  
## University\_GPA Field\_of\_Study Internships\_Completed Projects\_Completed  
## 1 3.96 Arts 3 7  
## 2 3.63 Law 4 7  
## 3 2.63 Medicine 4 8  
## 4 2.81 Computer Science 3 9  
## 5 2.48 Engineering 4 6  
## 6 3.78 Law 2 3  
## 7 3.83 Computer Science 0 1  
## 8 2.84 Computer Science 1 5  
## 9 3.31 Business 2 3  
## 10 2.33 Computer Science 1 5  
## Certifications Soft\_Skills\_Score Networking\_Score Job\_Offers Starting\_Salary  
## 1 2 9 8 5 27200  
## 2 3 8 1 4 25000  
## 3 1 1 9 0 42400  
## 4 1 10 6 1 57400  
## 5 4 10 9 4 47600  
## 6 2 2 2 1 68400  
## 7 3 3 3 2 55500  
## 8 5 5 1 2 38000  
## 9 0 5 5 2 68900  
## 10 3 10 2 0 58900  
## Career\_Satisfaction Years\_to\_Promotion Current\_Job\_Level Work\_Life\_Balance  
## 1 4 5 Entry 7  
## 2 1 1 Mid 7  
## 3 9 3 Entry 7  
## 4 7 5 Mid 5  
## 5 9 5 Entry 2  
## 6 9 2 Entry 8  
## 7 7 4 Mid 3  
## 8 2 3 Entry 3  
## 9 2 2 Entry 2  
## 10 4 2 Senior 2  
## Entrepreneurship University\_Ranking\_Category  
## 1 No Moderately Ranked  
## 2 No High ranked  
## 3 No Low ranked  
## 4 No High ranked  
## 5 No Low ranked  
## 6 Yes Low ranked  
## 7 No Low ranked  
## 8 No High ranked  
## 9 No Moderately Ranked  
## 10 No High ranked

The rest of the variables can be converted to categorical data as shown below:

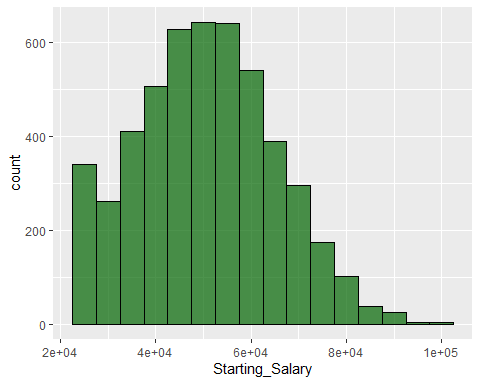
education\_data$Internships\_Completed <- factor(education\_data$Internships\_Completed)  
education\_data$Projects\_Completed <- factor(education\_data$Projects\_Completed)  
education\_data$Certifications <- factor(education\_data$Certifications)  
education\_data$Soft\_Skills\_Score <- factor(education\_data$Soft\_Skills\_Score)  
education\_data$Networking\_Score <- factor(education\_data$Networking\_Score)  
education\_data$Gender <- factor(education\_data$Gender)

We can remove the University\_Ranking column so that it is not used.

education\_data = education\_data[,-6]

Distribution of starting salary:

library(ggplot2)  
ggplot(education\_data, aes(x = Starting\_Salary)) +  
 geom\_histogram(binwidth = 5000, fill = "darkgreen", color = "black", alpha = 0.7)

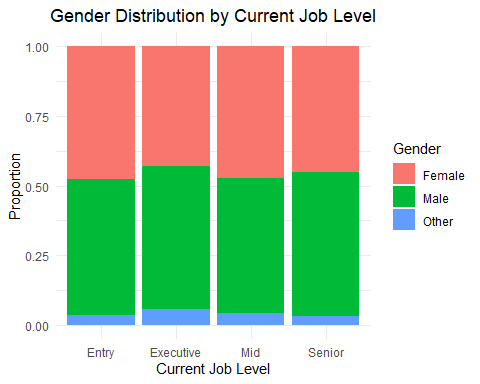


labs(  
 title = "Distribution of Starting Salary",  
 x = "Starting Salary ($)",  
 y = "Count / Density"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))

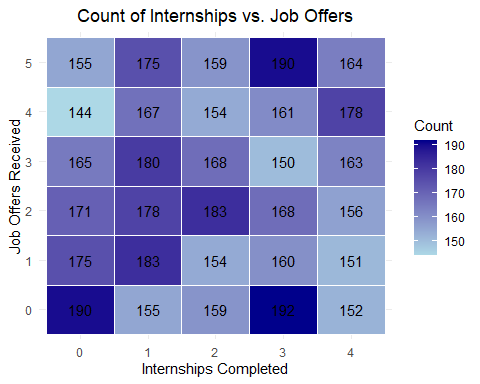
## NULL

Distribution of gender across job levels:

library(ggplot2)  
ggplot(education\_data, aes(x = Current\_Job\_Level, fill = Gender)) +  
 geom\_bar(position = "fill") +  
 labs(  
 title = "Gender Distribution by Current Job Level",  
 x = "Current Job Level",  
 y = "Proportion"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))

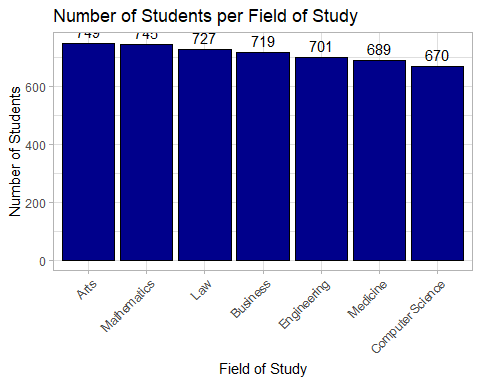
 We can visualize how the number of internships impact the job offers received:

education\_data %>%  
 count(Internships\_Completed, Job\_Offers) %>%   
 ggplot(aes(x = as.factor(Internships\_Completed), y = as.factor(Job\_Offers), fill = n)) +  
 geom\_tile(color = "white") +   
 geom\_text(aes(label = n), color = "black") +   
 scale\_fill\_gradient(low = "lightblue", high = "darkblue") +  
 labs(  
 title = "Count of Internships vs. Job Offers",  
 x = "Internships Completed",  
 y = "Job Offers Received",  
 fill = "Count"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))



Now to display the number of students from each field of study, this could be necessary as a stratified or cluster sampling can be taken based on the filed of study.

library(dplyr)  
library(ggplot2)  
  
field\_of\_study\_counts <- education\_data %>%  
 group\_by(Field\_of\_Study) %>%  
 count(name = "Number\_of\_Students") %>%  
 arrange(desc(Number\_of\_Students))  
  
  
education\_data$Field\_of\_Study <- factor(education\_data$Field\_of\_Study,  
 levels = field\_of\_study\_counts$Field\_of\_Study)  
  
ggplot(data = education\_data, aes(x = Field\_of\_Study)) +  
 geom\_bar(fill = "darkblue", color = "black") +  
 labs(  
 title = "Number of Students per Field of Study",  
 x = "Field of Study",  
 y = "Number of Students"  
 ) +  
 theme\_light() +   
 theme(axis.text.x = element\_text(angle = 45, hjust = 1)) +   
 geom\_text(stat = "count", aes(label = after\_stat(count)), vjust = -0.5)



As the number of students across each field of study is approximately equal, it can be considered as an ideal candidate for cluster sampling. Sampling to be done after a model is fit.

We can now proceed to fit the multiple linear regression model on the dataset and use it to predict the starting salary of a student.

As there are multiple assumptions that are to be cleared for the model, we can choose to switch to classification by changing the starting salary to a categorical variable through labeling.

Using the ordinary least-squares method to fit the model:

colnames(education\_data)

## [1] "Student\_ID" "Age"   
## [3] "Gender" "High\_School\_GPA"   
## [5] "SAT\_Score" "University\_GPA"   
## [7] "Field\_of\_Study" "Internships\_Completed"   
## [9] "Projects\_Completed" "Certifications"   
## [11] "Soft\_Skills\_Score" "Networking\_Score"   
## [13] "Job\_Offers" "Starting\_Salary"   
## [15] "Career\_Satisfaction" "Years\_to\_Promotion"   
## [17] "Current\_Job\_Level" "Work\_Life\_Balance"   
## [19] "Entrepreneurship" "University\_Ranking\_Category"

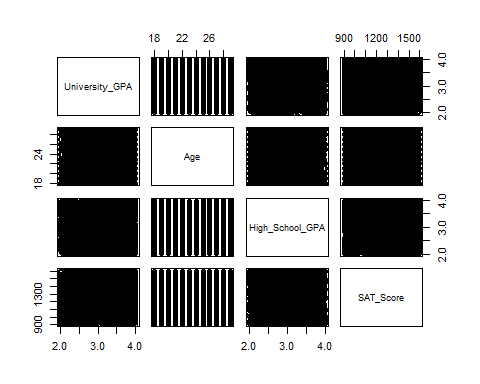
salary\_pred\_full\_model = lm(Starting\_Salary ~ Age+Gender+High\_School\_GPA+SAT\_Score+University\_GPA+Field\_of\_Study+Internships\_Completed+Projects\_Completed+Certifications+Soft\_Skills\_Score+Networking\_Score+University\_Ranking\_Category ,data = education\_data)  
  
summary(salary\_pred\_full\_model)

##   
## Call:  
## lm(formula = Starting\_Salary ~ Age + Gender + High\_School\_GPA +   
## SAT\_Score + University\_GPA + Field\_of\_Study + Internships\_Completed +   
## Projects\_Completed + Certifications + Soft\_Skills\_Score +   
## Networking\_Score + University\_Ranking\_Category, data = education\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -30198 -10577 -273 9890 50362   
##   
## Coefficients:  
## Estimate Std. Error t value  
## (Intercept) 50492.0905 2813.3452 17.947  
## Age 60.1917 59.3476 1.014  
## GenderMale 61.6980 420.1834 0.147  
## GenderOther -687.8222 1094.9583 -0.628  
## High\_School\_GPA -296.1421 358.2188 -0.827  
## SAT\_Score 0.2139 1.0143 0.211  
## University\_GPA 47.3767 357.8969 0.132  
## Field\_of\_StudyMathematics -644.0131 752.8497 -0.855  
## Field\_of\_StudyLaw -1294.2769 757.9416 -1.708  
## Field\_of\_StudyBusiness -1161.8957 761.5724 -1.526  
## Field\_of\_StudyEngineering -973.1841 765.2175 -1.272  
## Field\_of\_StudyMedicine -1161.4509 768.6380 -1.511  
## Field\_of\_StudyComputer Science -678.0277 774.6629 -0.875  
## Internships\_Completed1 -19.9762 645.8117 -0.031  
## Internships\_Completed2 -183.4172 655.1768 -0.280  
## Internships\_Completed3 -164.9211 648.1960 -0.254  
## Internships\_Completed4 943.8292 656.9080 1.437  
## Projects\_Completed1 562.6541 938.4985 0.600  
## Projects\_Completed2 -200.2490 943.3592 -0.212  
## Projects\_Completed3 856.7340 919.4240 0.932  
## Projects\_Completed4 1263.0184 948.3250 1.332  
## Projects\_Completed5 401.4498 943.0525 0.426  
## Projects\_Completed6 1062.3811 930.0437 1.142  
## Projects\_Completed7 -141.6820 947.5466 -0.150  
## Projects\_Completed8 761.1029 938.4340 0.811  
## Projects\_Completed9 1170.5797 919.8914 1.273  
## Certifications1 -241.7171 715.9609 -0.338  
## Certifications2 -6.8432 719.5988 -0.010  
## Certifications3 -1022.2097 717.7293 -1.424  
## Certifications4 -382.0254 713.2115 -0.536  
## Certifications5 -832.3882 719.6549 -1.157  
## Soft\_Skills\_Score2 905.9852 943.5627 0.960  
## Soft\_Skills\_Score3 1584.9334 922.5605 1.718  
## Soft\_Skills\_Score4 1565.4451 944.7033 1.657  
## Soft\_Skills\_Score5 407.5454 926.8212 0.440  
## Soft\_Skills\_Score6 875.1810 926.1701 0.945  
## Soft\_Skills\_Score7 1066.0563 923.2973 1.155  
## Soft\_Skills\_Score8 -792.5577 933.7828 -0.849  
## Soft\_Skills\_Score9 983.5348 921.9703 1.067  
## Soft\_Skills\_Score10 1601.8937 935.5600 1.712  
## Networking\_Score2 -1922.7283 930.5329 -2.066  
## Networking\_Score3 -1722.1845 939.7775 -1.833  
## Networking\_Score4 -2100.0509 936.0527 -2.244  
## Networking\_Score5 -925.6461 925.3021 -1.000  
## Networking\_Score6 -911.9324 912.8331 -0.999  
## Networking\_Score7 -1039.2184 935.5103 -1.111  
## Networking\_Score8 -1234.4905 925.1500 -1.334  
## Networking\_Score9 -1294.3217 926.7356 -1.397  
## Networking\_Score10 -834.3801 934.8819 -0.892  
## University\_Ranking\_CategoryLow ranked -467.1148 589.8739 -0.792  
## University\_Ranking\_CategoryModerately Ranked -336.0557 583.4416 -0.576  
## University\_Ranking\_CategoryVery low ranked 1018.3750 580.9388 1.753  
## Pr(>|t|)   
## (Intercept) <2e-16 \*\*\*  
## Age 0.3105   
## GenderMale 0.8833   
## GenderOther 0.5299   
## High\_School\_GPA 0.4084   
## SAT\_Score 0.8329   
## University\_GPA 0.8947   
## Field\_of\_StudyMathematics 0.3924   
## Field\_of\_StudyLaw 0.0878 .   
## Field\_of\_StudyBusiness 0.1272   
## Field\_of\_StudyEngineering 0.2035   
## Field\_of\_StudyMedicine 0.1308   
## Field\_of\_StudyComputer Science 0.3815   
## Internships\_Completed1 0.9753   
## Internships\_Completed2 0.7795   
## Internships\_Completed3 0.7992   
## Internships\_Completed4 0.1508   
## Projects\_Completed1 0.5488   
## Projects\_Completed2 0.8319   
## Projects\_Completed3 0.3515   
## Projects\_Completed4 0.1830   
## Projects\_Completed5 0.6704   
## Projects\_Completed6 0.2534   
## Projects\_Completed7 0.8811   
## Projects\_Completed8 0.4174   
## Projects\_Completed9 0.2032   
## Certifications1 0.7357   
## Certifications2 0.9924   
## Certifications3 0.1544   
## Certifications4 0.5922   
## Certifications5 0.2475   
## Soft\_Skills\_Score2 0.3370   
## Soft\_Skills\_Score3 0.0859 .   
## Soft\_Skills\_Score4 0.0976 .   
## Soft\_Skills\_Score5 0.6602   
## Soft\_Skills\_Score6 0.3447   
## Soft\_Skills\_Score7 0.2483   
## Soft\_Skills\_Score8 0.3961   
## Soft\_Skills\_Score9 0.2861   
## Soft\_Skills\_Score10 0.0869 .   
## Networking\_Score2 0.0389 \*   
## Networking\_Score3 0.0669 .   
## Networking\_Score4 0.0249 \*   
## Networking\_Score5 0.3172   
## Networking\_Score6 0.3178   
## Networking\_Score7 0.2667   
## Networking\_Score8 0.1821   
## Networking\_Score9 0.1626   
## Networking\_Score10 0.3722   
## University\_Ranking\_CategoryLow ranked 0.4285   
## University\_Ranking\_CategoryModerately Ranked 0.5646   
## University\_Ranking\_CategoryVery low ranked 0.0797 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 14500 on 4948 degrees of freedom  
## Multiple R-squared: 0.009599, Adjusted R-squared: -0.0006095   
## F-statistic: 0.9403 on 51 and 4948 DF, p-value: 0.5952

A negative adjusted R-squared means the model’s predictions are worse than simply using the mean of the dependent variable as a prediction. It suggests that the model doesn’t effectively explain the variation in the outcome variable and that the predictors are not helpful, potentially indicating a poor model.

This could be a result of multicollinearity between the predictors making them not not significant. We can check for multicollinearity between two continuous variables:

pairs(~University\_GPA +Age+High\_School\_GPA+ SAT\_Score , data = education\_data)



There is no pattern that appears between the predictors, we can use the VIF test to confirm if there is multicollinearity.

library(mctest)   
imcdiag(salary\_pred\_full\_model, method = "VIF")

##   
## Call:  
## imcdiag(mod = salary\_pred\_full\_model, method = "VIF")  
##   
##   
## VIF Multicollinearity Diagnostics  
##   
## VIF detection  
## Age 1.0106 0  
## GenderMale 1.0495 0  
## GenderOther 1.0529 0  
## High\_School\_GPA 1.0112 0  
## SAT\_Score 1.0103 0  
## University\_GPA 1.0107 0  
## Field\_of\_StudyMathematics 1.7092 0  
## Field\_of\_StudyLaw 1.6977 0  
## Field\_of\_StudyBusiness 1.6984 0  
## Field\_of\_StudyEngineering 1.6788 0  
## Field\_of\_StudyMedicine 1.6694 0  
## Field\_of\_StudyComputer Science 1.6562 0  
## Internships\_Completed1 1.6318 0  
## Internships\_Completed2 1.6051 0  
## Internships\_Completed3 1.6238 0  
## Internships\_Completed4 1.5972 0  
## Projects\_Completed1 1.8752 0  
## Projects\_Completed2 1.8607 0  
## Projects\_Completed3 1.9431 0  
## Projects\_Completed4 1.8458 0  
## Projects\_Completed5 1.8663 0  
## Projects\_Completed6 1.9170 0  
## Projects\_Completed7 1.8601 0  
## Projects\_Completed8 1.8817 0  
## Projects\_Completed9 1.9451 0  
## Certifications1 1.7089 0  
## Certifications2 1.6935 0  
## Certifications3 1.7076 0  
## Certifications4 1.7134 0  
## Certifications5 1.7036 0  
## Soft\_Skills\_Score2 1.8205 0  
## Soft\_Skills\_Score3 1.8959 0  
## Soft\_Skills\_Score4 1.8214 0  
## Soft\_Skills\_Score5 1.8778 0  
## Soft\_Skills\_Score6 1.8784 0  
## Soft\_Skills\_Score7 1.8861 0  
## Soft\_Skills\_Score8 1.8498 0  
## Soft\_Skills\_Score9 1.8903 0  
## Soft\_Skills\_Score10 1.8435 0  
## Networking\_Score2 1.8501 0  
## Networking\_Score3 1.8229 0  
## Networking\_Score4 1.8320 0  
## Networking\_Score5 1.8684 0  
## Networking\_Score6 1.9216 0  
## Networking\_Score7 1.8399 0  
## Networking\_Score8 1.8743 0  
## Networking\_Score9 1.8612 0  
## Networking\_Score10 1.8408 0  
## University\_Ranking\_CategoryLow ranked 1.5223 0  
## University\_Ranking\_CategoryModerately Ranked 1.5325 0  
## University\_Ranking\_CategoryVery low ranked 1.5358 0  
##   
## NOTE: VIF Method Failed to detect multicollinearity  
##   
##   
## 0 --> COLLINEARITY is not detected by the test  
##   
## ===================================

As some of the factors were insignificant, we can remove them using the step\_wise function of the oslrr package. This method will give us the best fit model based on a few metrics.

library(olsrr)

## Warning: package 'olsrr' was built under R version 4.4.3

##   
## Attaching package: 'olsrr'

## The following object is masked from 'package:datasets':  
##   
## rivers

salary\_pred\_Subsets = ols\_step\_best\_subset(salary\_pred\_full\_model, details=FALSE)

We are only interested in , AIC(Akaike Information Criterion) and Mallow’s cp criterion for this model. We do not choose to use R2 as it does not punish the model for adding more predictors/overfitting.

AdjustedR2=c((salary\_pred\_Subsets$metrics)$adjr)  
cp=c((salary\_pred\_Subsets$metrics)$cp)  
AIC=c((salary\_pred\_Subsets$metrics)$aic)  
cbind(AdjustedR2,cp,AIC)

## AdjustedR2 cp AIC  
## [1,] 5.977466e-04 3.979312 110013.9  
## [2,] 4.196567e-04 13.876697 110023.8  
## [3,] 1.414276e-03 11.931582 110021.8  
## [4,] 8.175907e-04 23.912910 110033.8  
## [5,] 4.712920e-04 31.639101 110041.5  
## [6,] 4.818598e-04 35.591048 110045.4  
## [7,] 1.574866e-04 42.202467 110051.9  
## [8,] 1.574997e-04 43.203169 110052.9  
## [9,] 9.093337e-05 44.533371 110054.3  
## [10,] -2.180239e-04 48.063178 110057.8  
## [11,] -4.108982e-04 50.017523 110059.7  
## [12,] -6.095395e-04 52.000000 110061.7

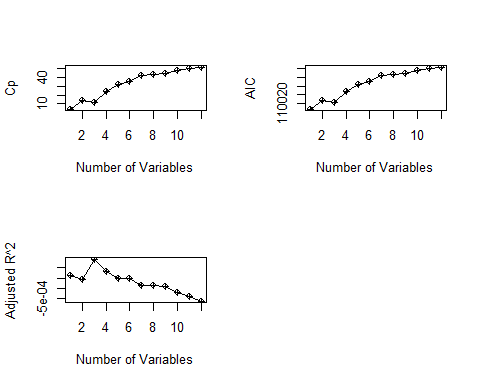
The model with highest adjusted R2 uses only 3 predictors and the cp mallows criterion is slightly lower compared to the other predictors. The AIC is mostly similar across the models but slightly lower for the one with 3 predictors.

salary\_pred\_Subsets$metrics$predictors[3]

## [1] "Soft\_Skills\_Score Networking\_Score University\_Ranking\_Category"

The 3 predictors are soft skills, networking score and university rank

par(mfrow=c(2,2)) # split the plotting panel into a 2 x 2 grid  
plot(cp,type = "o",pch=10, xlab="Number of Variables",ylab= "Cp")  
plot(AIC,type = "o",pch=10, xlab="Number of Variables",ylab= "AIC")  
plot(AdjustedR2,type = "o",pch=10, xlab="Number of Variables",ylab= "Adjusted R^2")



The ideal cp\_mallows criterion is k+2, where k is the number of predictors. For 3 predictors the ideal cp\_mallows should be 5, but it is 11 in this case. Indicating a bias.

salary\_pred\_best\_model = lm(Starting\_Salary ~ Soft\_Skills\_Score+Networking\_Score+University\_Ranking\_Category ,data = education\_data)  
  
summary(salary\_pred\_best\_model)

##   
## Call:  
## lm(formula = Starting\_Salary ~ Soft\_Skills\_Score + Networking\_Score +   
## University\_Ranking\_Category, data = education\_data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -27975 -10488 -319 9831 49795   
##   
## Coefficients:  
## Estimate Std. Error t value  
## (Intercept) 50905.7 993.8 51.221  
## Soft\_Skills\_Score2 804.9 940.6 0.856  
## Soft\_Skills\_Score3 1517.5 918.3 1.652  
## Soft\_Skills\_Score4 1466.6 940.5 1.559  
## Soft\_Skills\_Score5 387.4 923.3 0.420  
## Soft\_Skills\_Score6 848.5 922.8 0.920  
## Soft\_Skills\_Score7 1051.9 920.7 1.142  
## Soft\_Skills\_Score8 -846.2 930.8 -0.909  
## Soft\_Skills\_Score9 995.7 918.6 1.084  
## Soft\_Skills\_Score10 1688.1 932.5 1.810  
## Networking\_Score2 -1941.4 928.4 -2.091  
## Networking\_Score3 -1719.7 936.8 -1.836  
## Networking\_Score4 -2172.9 933.2 -2.328  
## Networking\_Score5 -1014.2 922.0 -1.100  
## Networking\_Score6 -866.7 909.7 -0.953  
## Networking\_Score7 -1043.0 932.1 -1.119  
## Networking\_Score8 -1218.7 922.1 -1.322  
## Networking\_Score9 -1296.0 924.5 -1.402  
## Networking\_Score10 -860.1 932.0 -0.923  
## University\_Ranking\_CategoryLow ranked -438.5 587.1 -0.747  
## University\_Ranking\_CategoryModerately Ranked -324.5 580.9 -0.559  
## University\_Ranking\_CategoryVery low ranked 1017.0 578.6 1.758  
## Pr(>|t|)   
## (Intercept) <2e-16 \*\*\*  
## Soft\_Skills\_Score2 0.3922   
## Soft\_Skills\_Score3 0.0985 .   
## Soft\_Skills\_Score4 0.1190   
## Soft\_Skills\_Score5 0.6748   
## Soft\_Skills\_Score6 0.3579   
## Soft\_Skills\_Score7 0.2533   
## Soft\_Skills\_Score8 0.3633   
## Soft\_Skills\_Score9 0.2785   
## Soft\_Skills\_Score10 0.0703 .   
## Networking\_Score2 0.0366 \*   
## Networking\_Score3 0.0665 .   
## Networking\_Score4 0.0199 \*   
## Networking\_Score5 0.2714   
## Networking\_Score6 0.3408   
## Networking\_Score7 0.2632   
## Networking\_Score8 0.1863   
## Networking\_Score9 0.1610   
## Networking\_Score10 0.3561   
## University\_Ranking\_CategoryLow ranked 0.4552   
## University\_Ranking\_CategoryModerately Ranked 0.5764   
## University\_Ranking\_CategoryVery low ranked 0.0789 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 14480 on 4978 degrees of freedom  
## Multiple R-squared: 0.005609, Adjusted R-squared: 0.001414   
## F-statistic: 1.337 on 21 and 4978 DF, p-value: 0.1386

Based on the adjusted R2 obtained for the best fit model, which is 0.14%,and the cp-mallows criterion it does not make sense to proceed with the multiple linear regression model. So the rest of the assumptions required for mlrm will not be performed.

We can use a regression tree as it requires less assumptions and can be pruned to make it more interpret-able.

Taking a sample of 75% of the data:

library(tree)  
idx=sample(1:nrow(education\_data),0.75\*nrow(education\_data))  
train=education\_data[idx,]  
test=education\_data[-idx,]  
reg.tree.salary<-tree(Starting\_Salary ~ Age+Gender+High\_School\_GPA+SAT\_Score+University\_GPA+Field\_of\_Study+Internships\_Completed+Projects\_Completed+Certifications+Soft\_Skills\_Score+Networking\_Score+University\_Ranking\_Category, train)

## Warning in tree(Starting\_Salary ~ Age + Gender + High\_School\_GPA + SAT\_Score +  
## : NAs introduced by coercion

summary(reg.tree.salary)

##   
## Regression tree:  
## tree(formula = Starting\_Salary ~ Age + Gender + High\_School\_GPA +   
## SAT\_Score + University\_GPA + Field\_of\_Study + Internships\_Completed +   
## Projects\_Completed + Certifications + Soft\_Skills\_Score +   
## Networking\_Score + University\_Ranking\_Category, data = train)  
## Variables actually used in tree construction:  
## character(0)  
## Number of terminal nodes: 1   
## Residual mean deviance: 210600000 = 7.895e+11 / 3749   
## Distribution of residuals:  
## Min. 1st Qu. Median Mean 3rd Qu. Max.   
## -25400.0 -10500.0 -151.4 0.0 9999.0 50200.0

It is not possible to construct a regression tree with a single node, the residual deviance is extremely high, this also indicates that regression cannot be used to predict the starting salary of students.

We can choose to perform classification by converting the salary data into a categorical variable.

education\_data <- education\_data %>%  
 mutate(  
 Starting\_Salary\_Category = case\_when(  
 Starting\_Salary >= 25000 & Starting\_Salary <= 40000 ~ "Low.Salary",  
 Starting\_Salary > 40000 & Starting\_Salary <= 75000 ~ "Median.Salary",  
 Starting\_Salary > 75000 & Starting\_Salary <= 120000 ~ "High.Salary"  
 )  
 )

Remove the continuous variable:

As the dependent variable consists of more than 2 classes it is better to use Linear Discriminant Analysis or Quadratic Discriminant Analysis to predict the outcome.

Both the LDA and QDA require the assumption of normality:

numerical\_cols <- c(  
 "Age", "High\_School\_GPA", "SAT\_Score", "University\_GPA")  
  
shapiro\_results <- list()  
  
cat("--- Shapiro-Wilk Normality Test Results ---\n")

## --- Shapiro-Wilk Normality Test Results ---

for (col\_name in numerical\_cols) {  
   
 data\_vector <- education\_data[[col\_name]]  
  
 data\_vector <- na.omit(data\_vector)  
  
 test\_result <- shapiro.test(data\_vector)  
  
 shapiro\_results[[col\_name]] <- test\_result  
  
 cat(sprintf("\nColumn: '%s'\n", col\_name))  
 cat(sprintf(" Shapiro-Wilk W statistic: %.4f\n", test\_result$statistic))  
 cat(sprintf(" p-value: %.4f\n", test\_result$p.value))  
  
 # Interpret the p-value  
 if (test\_result$p.value < 0.05) {  
 cat(" Conclusion: The data in this column is likely NOT normally distributed (p < 0.05).\n")  
 } else {  
 cat(" Conclusion: The data in this column appears to be normally distributed (p >= 0.05).\n")  
 }  
}

##   
## Column: 'Age'  
## Shapiro-Wilk W statistic: 0.9401  
## p-value: 0.0000  
## Conclusion: The data in this column is likely NOT normally distributed (p < 0.05).  
##   
## Column: 'High\_School\_GPA'  
## Shapiro-Wilk W statistic: 0.9558  
## p-value: 0.0000  
## Conclusion: The data in this column is likely NOT normally distributed (p < 0.05).  
##   
## Column: 'SAT\_Score'  
## Shapiro-Wilk W statistic: 0.9531  
## p-value: 0.0000  
## Conclusion: The data in this column is likely NOT normally distributed (p < 0.05).  
##   
## Column: 'University\_GPA'  
## Shapiro-Wilk W statistic: 0.9559  
## p-value: 0.0000  
## Conclusion: The data in this column is likely NOT normally distributed (p < 0.05).

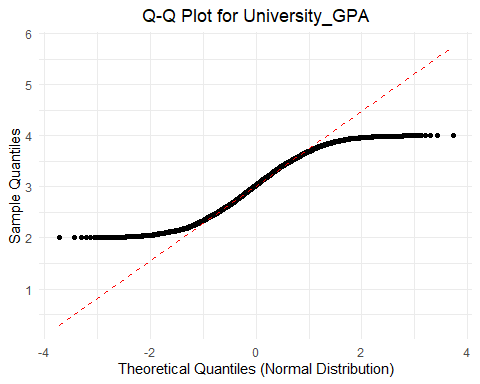
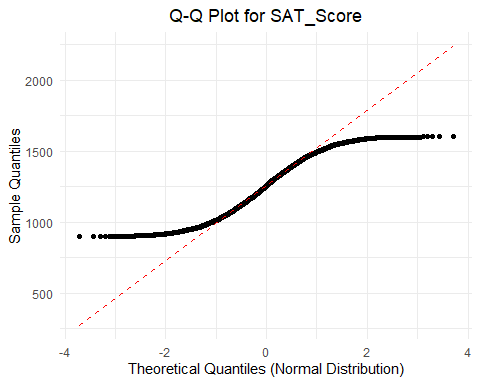
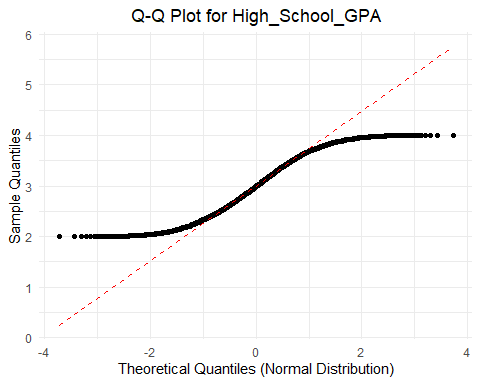
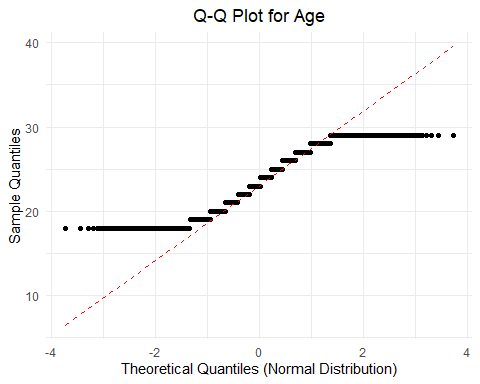
The W statistics which is close to 1 indicates that the variable follows a normal distribution but the p-value states to reject null-hypothesis, which is the data is normally distributed.

The shapiro test is very sensitive and can conclude data is not normal for small deviances. We can use Quantile-Quantile plots to test the results of the test:

qq\_plots\_list <- list()  
  
for (col\_name in numerical\_cols ) {  
 data\_vector <- education\_data[[col\_name]]  
  
  
 p <- ggplot(data.frame(x = data\_vector), aes(sample = x)) +  
 stat\_qq() + # Adds the QQ-plot points  
 stat\_qq\_line(color = "red", linetype = "dashed") +   
 labs(  
 title = paste0("Q-Q Plot for ", col\_name),  
 x = "Theoretical Quantiles (Normal Distribution)",  
 y = "Sample Quantiles"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))  
  
 # Store the plot in the list  
 qq\_plots\_list[[col\_name]] <- p  
}  
  
cat("\n--- Displaying Q-Q Plots ---\n")

##   
## --- Displaying Q-Q Plots ---

for (plot\_name in names(qq\_plots\_list)) {  
 print(qq\_plots\_list[[plot\_name]])  
}



The QQ plots indicate that there are heavier tails and the data is not normally distributed. This could due to outliers, we can check for outliers and remove them. Once the outliers are removed we can check for normality, if this fails we can proceed with box-cox or log-transformations on the data.

education\_data\_cleaned\_iqr <- education\_data  
  
cat("--- Outlier Detection and Removal using IQR Rule ---\n")

## --- Outlier Detection and Removal using IQR Rule ---

for (col\_name in numerical\_cols) {  
 cat(sprintf("\nProcessing column: '%s'\n", col\_name))  
  
 data\_vector <- education\_data[[col\_name]]  
 data\_vector\_no\_na <- na.omit(data\_vector)  
  
   
 Q1 <- quantile(data\_vector\_no\_na, 0.25)  
 Q3 <- quantile(data\_vector\_no\_na, 0.75)  
 IQR\_val <- Q3 - Q1  
  
   
 lower\_bound <- Q1 - 1.5 \* IQR\_val  
 upper\_bound <- Q3 + 1.5 \* IQR\_val  
  
 outlier\_indices <- which(data\_vector < lower\_bound | data\_vector > upper\_bound)  
  
 # Report detected outliers  
 if (length(outlier\_indices) > 0) {  
 cat(sprintf(" Detected %d outliers in '%s'.\n", length(outlier\_indices), col\_name))  
 cat(sprintf(" Outlier values: %s\n", paste(data\_vector[outlier\_indices], collapse = ", ")))  
 cat(sprintf(" Lower bound: %.2f, Upper bound: %.2f\n", lower\_bound, upper\_bound))  
  
 } else {  
 cat(sprintf(" No outliers detected in '%s' using IQR rule.\n", col\_name))  
 }  
}

##   
## Processing column: 'Age'  
## No outliers detected in 'Age' using IQR rule.  
##   
## Processing column: 'High\_School\_GPA'  
## No outliers detected in 'High\_School\_GPA' using IQR rule.  
##   
## Processing column: 'SAT\_Score'  
## No outliers detected in 'SAT\_Score' using IQR rule.  
##   
## Processing column: 'University\_GPA'  
## No outliers detected in 'University\_GPA' using IQR rule.

As there are no outliers detected, we can use the box-cox transformation on the continuous variables to obtain normality.

library(caret)

## Warning: package 'caret' was built under R version 4.4.3

## Loading required package: lattice

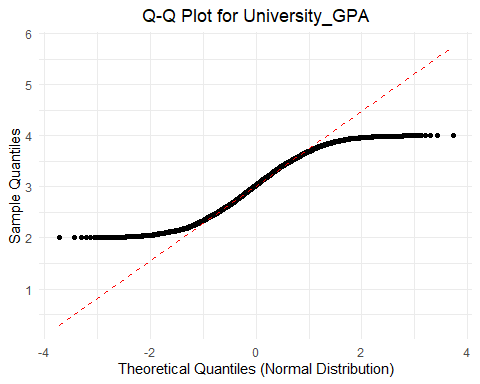
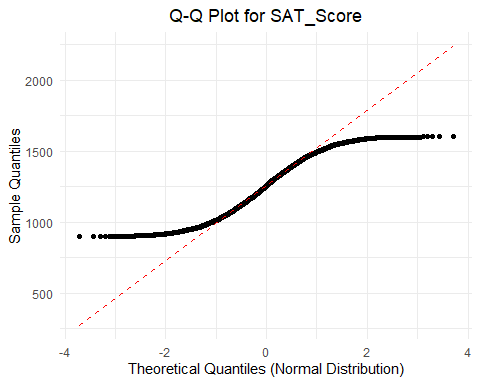
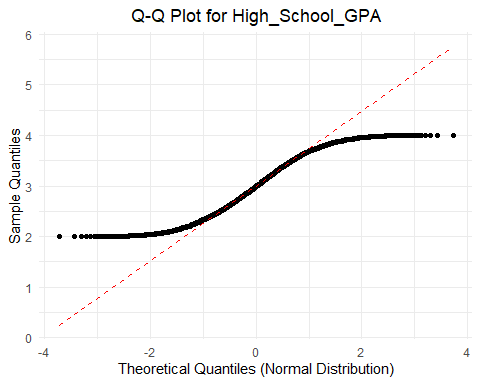
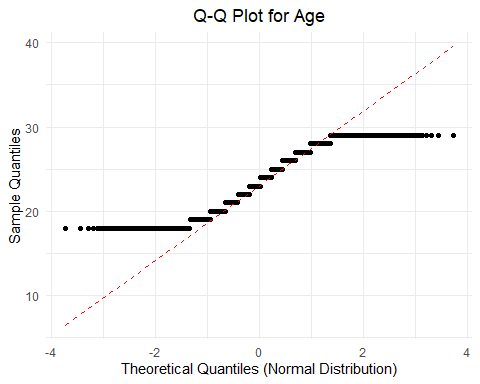
education\_data\_transformed <- education\_data  
for (col\_name in numerical\_cols) {  
 data\_vector <- education\_data\_transformed[[col\_name]]  
 bc\_object <- BoxCoxTrans(data\_vector)  
 lambda\_val <- bc\_object$lambda  
 cat(sprintf(" Optimal lambda for '%s': %.4f\n", col\_name, lambda\_val))  
   
 transformed\_data <- predict(bc\_object, data\_vector)  
 education\_data\_transformed[[paste0(col\_name, "\_BC")]] <- transformed\_data  
}

## Optimal lambda for 'Age': 0.6000  
## Optimal lambda for 'High\_School\_GPA': 0.7000  
## Optimal lambda for 'SAT\_Score': 0.8000  
## Optimal lambda for 'University\_GPA': 0.9000

qq\_plots\_list <- list()  
  
for (col\_name in numerical\_cols ) {  
 data\_vector <- education\_data\_transformed[[col\_name]]  
  
  
 p <- ggplot(data.frame(x = data\_vector), aes(sample = x)) +  
 stat\_qq() + # Adds the QQ-plot points  
 stat\_qq\_line(color = "red", linetype = "dashed") +   
 labs(  
 title = paste0("Q-Q Plot for ", col\_name),  
 x = "Theoretical Quantiles (Normal Distribution)",  
 y = "Sample Quantiles"  
 ) +  
 theme\_minimal() +  
 theme(plot.title = element\_text(hjust = 0.5))  
  
 # Store the plot in the list  
 qq\_plots\_list[[col\_name]] <- p  
}  
  
cat("\n--- Displaying Q-Q Plots for transformed data---\n")

##   
## --- Displaying Q-Q Plots for transformed data---

for (plot\_name in names(qq\_plots\_list)) {  
 print(qq\_plots\_list[[plot\_name]])  
}



Based on the plots it appears the tails are still heavy on the end, indicating kurtosis. However, if we consider the test statistic from the shapiro-Wilk tests, they were closer to 1 indicating the data is approximately normally distributed. While LDA assumes multivariate normality within each class, in practice, it’s often reasonably robust to mild deviations, especially with large sample sizes.

We can consider the data to be approximately normally distributed for this case. Another assumption that needs to be satisfied for LDA is equal variance between the classes/predictors.

We can use Levene’s test for equal variance across the predictors.

The null hypothesis indicates the variances are equal across all groups The alternate hypothesis indicates at least one group variance is different from the others

library(car)

## Loading required package: carData

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

levene\_test\_age <- leveneTest(Age ~ factor(Starting\_Salary\_Category), data = education\_data)  
levene\_test\_age

## Levene's Test for Homogeneity of Variance (center = median)  
## Df F value Pr(>F)  
## group 2 1.4406 0.2369  
## 4997

levene\_test\_SchoolGPA <- leveneTest(High\_School\_GPA ~ factor(Starting\_Salary\_Category), data = education\_data)  
levene\_test\_SchoolGPA

## Levene's Test for Homogeneity of Variance (center = median)  
## Df F value Pr(>F)  
## group 2 1.7073 0.1815  
## 4997

levene\_test\_SAT <- leveneTest(SAT\_Score ~ factor(Starting\_Salary\_Category), data = education\_data)  
levene\_test\_SAT

## Levene's Test for Homogeneity of Variance (center = median)  
## Df F value Pr(>F)  
## group 2 0.3155 0.7295  
## 4997

levene\_test\_UniGPA <- leveneTest(University\_GPA ~ factor(Starting\_Salary\_Category), data = education\_data)  
levene\_test\_UniGPA

## Levene's Test for Homogeneity of Variance (center = median)  
## Df F value Pr(>F)  
## group 2 2.2858 0.1018  
## 4997

The predictor High\_School\_GPA fails the test of equal variances, we choose to remove the predictor or proceed with QDA which does not require the assumption of equal variance:

We can proceed with QDA as the high school GPA can be a significant predictor which would get excluded while performing LDA.

QDA can be performed using a 10 fold cross validation with the help of the caret library. We do not need to use the transformed data as it was unable to transform the data, using this would simply add to complexity in interpreting the model.

library(caret)  
set.seed(42)  
indexs = sample(1:nrow(education\_data),0.75\*nrow(education\_data))  
train\_data = education\_data[indexs,]  
test\_data = education\_data[-indexs,]  
  
qda\_model\_caret <- train(  
 factor(Starting\_Salary\_Category) ~ Age + High\_School\_GPA + SAT\_Score + University\_GPA ,   
 data = train\_data,  
 method = "qda",   
 trControl = trainControl(method = 'cv', number = 10, verboseIter = FALSE, classProbs = TRUE, summaryFunction = defaultSummary),  
 metric = "Accuracy"  
)  
qda\_model\_caret

## Quadratic Discriminant Analysis   
##   
## 3750 samples  
## 4 predictor  
## 3 classes: 'High.Salary', 'Low.Salary', 'Median.Salary'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 3376, 3374, 3374, 3376, 3375, 3376, ...   
## Resampling results:  
##   
## Accuracy Kappa  
## 0.7005342 0

The above output indicates that groups are too small for the folds. We can use fewer folds with the same training split data created before.

qda\_model\_caret\_5fold <- train(  
 factor(Starting\_Salary\_Category) ~ Age + High\_School\_GPA + SAT\_Score + University\_GPA ,   
 data = train\_data,  
 method = "qda",   
 trControl = trainControl(method = 'cv', number = 5, verboseIter = FALSE, classProbs = TRUE, summaryFunction = defaultSummary),  
 metric = "Accuracy"  
)  
qda\_model\_caret\_5fold

## Quadratic Discriminant Analysis   
##   
## 3750 samples  
## 4 predictor  
## 3 classes: 'High.Salary', 'Low.Salary', 'Median.Salary'   
##   
## No pre-processing  
## Resampling: Cross-Validated (5 fold)   
## Summary of sample sizes: 2999, 3000, 3001, 3000, 3000   
## Resampling results:  
##   
## Accuracy Kappa  
## 0.7005338 0

The accuracy is similar for the 10-fold and 5-fold cross validation. We can use the 10-fold cross- validated model.

qda.class<-predict(qda\_model\_caret, test\_data)  
table(qda.class, test\_data$Starting\_Salary\_Category)

##   
## qda.class High.Salary Low.Salary Median.Salary  
## High.Salary 0 0 0  
## Low.Salary 0 0 0  
## Median.Salary 57 311 882

Although the accuracy is 70%, the qda predicted all the salaries to be median salary. We can look at a different approach, which uses a better visualization like classification tress.

train\_data$Starting\_Salary\_Category = factor(train\_data$Starting\_Salary\_Category)

library(klaR)

## Warning: package 'klaR' was built under R version 4.4.3

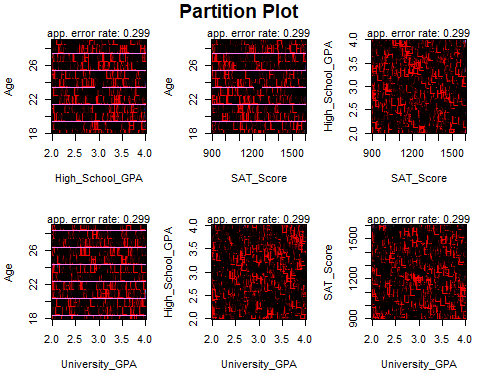
## Loading required package: MASS

##   
## Attaching package: 'MASS'

## The following object is masked from 'package:olsrr':  
##   
## cement

## The following object is masked from 'package:dplyr':  
##   
## select

partimat(Starting\_Salary\_Category ~ Age + High\_School\_GPA + SAT\_Score + University\_GPA,  
 data = train\_data,  
 method = "qda")

 The above plots above are not very clear due to the large amounts of data. We may need to rely on the accuracy score to accurately determine how to classify the outcome variable.

There is a better way to visualize the outcomes, that is using decision trees. Although less accurate it is easier to understand the visual.

classification.salary.tree = tree(Starting\_Salary\_Category ~Age + Gender + High\_School\_GPA + SAT\_Score + University\_GPA + Field\_of\_Study + Internships\_Completed + Projects\_Completed + Certifications + Soft\_Skills\_Score + Networking\_Score, data=train\_data)  
summary(classification.salary.tree)

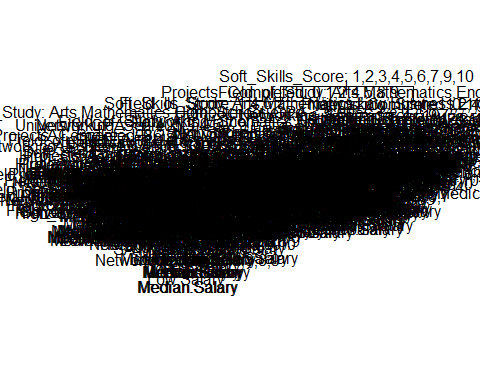
##   
## Classification tree:  
## tree(formula = Starting\_Salary\_Category ~ Age + Gender + High\_School\_GPA +   
## SAT\_Score + University\_GPA + Field\_of\_Study + Internships\_Completed +   
## Projects\_Completed + Certifications + Soft\_Skills\_Score +   
## Networking\_Score, data = train\_data)  
## Variables actually used in tree construction:  
## character(0)  
## Number of terminal nodes: 1   
## Residual mean deviance: 1.495 = 5605 / 3749   
## Misclassification error rate: 0.2995 = 1123 / 3750

The above tree only grew a single node which is not helpful for predictions. We can change few parameters to force the tree to grow more nodes:

classification.salary.tree\_adjusted <- tree(  
 Starting\_Salary\_Category ~ Age + Gender + High\_School\_GPA + SAT\_Score + University\_GPA +  
 Field\_of\_Study + Internships\_Completed + Projects\_Completed + Certifications +  
 Soft\_Skills\_Score + Networking\_Score,  
 data = train\_data,  
 control = tree.control(  
 nobs = nrow(train\_data),  
 mindev = 0.001,  
 mincut = 2   
 )  
)  
summary(classification.salary.tree\_adjusted)

##   
## Classification tree:  
## tree(formula = Starting\_Salary\_Category ~ Age + Gender + High\_School\_GPA +   
## SAT\_Score + University\_GPA + Field\_of\_Study + Internships\_Completed +   
## Projects\_Completed + Certifications + Soft\_Skills\_Score +   
## Networking\_Score, data = train\_data, control = tree.control(nobs = nrow(train\_data),   
## mindev = 0.001, mincut = 2))  
## Number of terminal nodes: 312   
## Residual mean deviance: 0.8246 = 2835 / 3438   
## Misclassification error rate: 0.1907 = 715 / 3750

plot(classification.salary.tree\_adjusted)  
text(classification.salary.tree\_adjusted, pretty = 0)

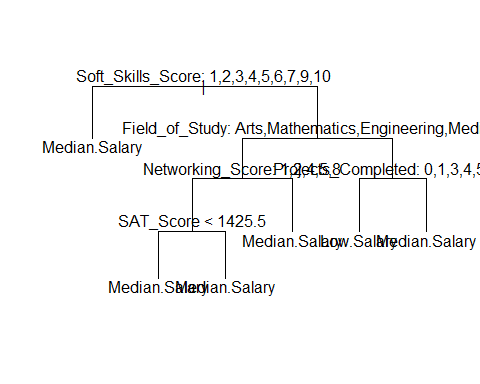
 Now we can prune the tree to obtain a smaller tree which can be understood visually:

#cv.salary <- cv.tree(classification.salary.tree\_adjusted, FUN = prune.misclass, K = 10,  
# control = tree.control(  
# nobs = nrow(train\_data),  
# mindev = 0.001,  
# mincut = 2   
#plot(cv.salary$size,cv.salary$dev,type='b')

It is not possible to prune the tree using cv.tree function as it internally builds a tree with a single node. Using the same parameters explicitly results in errors indicating that there are internal issues on how cv.tree constructs a tree with parameters.

We can simply choose the number of residual nodes based on best visualization

prune.class=prune.tree(classification.salary.tree\_adjusted,best=6)  
plot(prune.class)  
text(prune.class,pretty=0)



summary(prune.class)

##   
## Classification tree:  
## snip.tree(tree = classification.salary.tree\_adjusted, nodes = c(24L,   
## 15L, 25L, 13L, 14L, 2L))  
## Variables actually used in tree construction:  
## [1] "Soft\_Skills\_Score" "Field\_of\_Study" "Networking\_Score"   
## [4] "SAT\_Score" "Projects\_Completed"  
## Number of terminal nodes: 6   
## Residual mean deviance: 1.478 = 5532 / 3744   
## Misclassification error rate: 0.2971 = 1114 / 3750

Although less accurate the complete tree, this is tree is easier to visualize and does not overfit the test dataset. It has the same accuracy as the QDA model.

prune.pred=predict(prune.class,test\_data,type="class")  
table(prune.pred,test\_data$Starting\_Salary\_Category)

##   
## prune.pred High.Salary Low.Salary Median.Salary  
## High.Salary 0 0 0  
## Low.Salary 2 10 25  
## Median.Salary 55 301 857

This indicates that the model did not predict high salary successfully. There were 2 instances where lower salary -> 2 and Median salary -> 55 should have been classified as high salary.

There were 301 Low salary counts which were predicted as median salary and 25 median salary counts which were predicted as low salary.

Now, for the final part of the project. We need to determine what is the best way to obtain samples from the dataset, a Simple random sample, stratified sample and cluster sample can be taken and the population metrics can be compared to see which is more accurate and has a lower standard deviation.

Simple random sample without replacement:

set.seed(2024)  
N = dim(education\_data)[1]  
n = 300  
idx=sample(1:N,size = n, replace = FALSE)  
population\_mean = mean(education\_data$Starting\_Salary)  
population\_SD = sd(education\_data$Starting\_Salary)  
cat("The population mean is: ", population\_mean, "and the standard deviation is: ", population\_SD)

## The population mean is: 50563.54 and the standard deviation is: 14494.96

The Standard error can be calculated as

Standard\_error\_est = population\_SD/sqrt(n) \* sqrt((N-n)/(N-1))  
Standard\_error\_est

## [1] 811.4536

So, the best sampling method would provide a value close to a mean of 50563.54 and SE of 811.4536

library(survey)

## Warning: package 'survey' was built under R version 4.4.3

## Loading required package: grid

## Loading required package: Matrix

## Loading required package: survival

##   
## Attaching package: 'survival'

## The following object is masked from 'package:caret':  
##   
## cluster

##   
## Attaching package: 'survey'

## The following object is masked from 'package:graphics':  
##   
## dotchart

new\_data <- data.frame(education\_data[idx,],pw=rep(N/n,n),fpc=rep(N,n))  
SRS\_svy <- svydesign(id=~0, strata = NULL, weights=~pw, data = new\_data, fpc=~fpc)  
mean\_salary <- svymean(~Starting\_Salary, SRS\_svy)  
mean\_salary

## mean SE  
## Starting\_Salary 52096 850.68

The mean and SE are indeed close to the population statistics, we can check other sampling methods are see if they have the same values.

Stratified sampling:

library(sampling)

## Warning: package 'sampling' was built under R version 4.4.3

##   
## Attaching package: 'sampling'

## The following objects are masked from 'package:survival':  
##   
## cluster, strata

## The following object is masked from 'package:caret':  
##   
## cluster

desired\_sizes\_vector <- c("Arts" = 70,"Mathematics" = 70, "Law" = 70, "Business" = 70,"Engineering" = 70, "Medicine" = 70,  
 "Computer Science" = 70)  
Strata\_idx = sampling::strata(education\_data, stratanames=c("Field\_of\_Study"), size=as.numeric(desired\_sizes\_vector), method="srswor")  
Salary\_strat<-getdata(education\_data,Strata\_idx)  
summary(Salary\_strat$Field\_of\_Study)

## Arts Mathematics Law Business   
## 70 70 70 70   
## Engineering Medicine Computer Science   
## 70 70 70

library(survey)  
Salary\_strat2=data.frame(Salary\_strat, pw=1/Salary\_strat$Prob, fpc=c(rep(749,70),rep(745,70),rep(727,70),rep(719, 70),rep(701,70),rep(689,70),rep(670,70)))  
Strata\_svy<-svydesign(id=~1,strata = ~Field\_of\_Study, weights = ~pw, data = Salary\_strat2, fpc=~fpc)  
Strata\_mean\_salary<-svymean(~Starting\_Salary, Strata\_svy)  
Strata\_mean\_salary

## mean SE  
## Starting\_Salary 51021 645.57

The Simple random sample performed better than the stratified sample in this case. A proportional sampling could improve the results slightly but since the filed of study almost has equal number of counts it would only improve the results slightly.

Cluster Sampling:

all\_clusters <- levels(education\_data$Field\_of\_Study)  
num\_total\_clusters <- length(all\_clusters)  
num\_sampled\_clusters <- 3  
sampled\_cluster\_names <- sample(all\_clusters, size = num\_sampled\_clusters, replace = FALSE)  
cluster\_sample\_data <- education\_data %>%  
 filter(Field\_of\_Study %in% sampled\_cluster\_names)  
  
cluster\_sample\_data$cluster\_id <- as.integer(cluster\_sample\_data$Field\_of\_Study)   
  
cluster\_sample\_data$total\_clusters\_pop <- num\_total\_clusters  
  
cluster\_svy\_design <- svydesign(  
 id = ~cluster\_id,  
 fpc = ~total\_clusters\_pop, # FPC for the clusters themselves  
 data = cluster\_sample\_data  
)  
  
mean\_salary\_cluster <- svymean(~Starting\_Salary, cluster\_svy\_design)  
cat("\nMean Starting\_Salary from Cluster Sample (Survey Package):\n")

##   
## Mean Starting\_Salary from Cluster Sample (Survey Package):

print(mean\_salary\_cluster)

## mean SE  
## Starting\_Salary 50713 280.2

The cluster sampling gives us the best result for estimating the population mean. Indicating that the ‘field of study’ is a suitable column for clustering.

library(lme4)

## Warning: package 'lme4' was built under R version 4.4.3

icc\_model <- lmer(Starting\_Salary ~ (1 | Field\_of\_Study), data = education\_data)

## boundary (singular) fit: see help('isSingular')

variance\_components <- as.data.frame(VarCorr(icc\_model))  
var\_between\_clusters <- variance\_components$vcov[variance\_components$grp == "Field\_of\_Study"]  
var\_within\_clusters <- variance\_components$vcov[variance\_components$grp == "Residual"]  
icc\_score <- var\_between\_clusters / (var\_between\_clusters + var\_within\_clusters)  
icc\_score

## [1] 0

The ICC score is 0 indicating that there is no correlation between the clusters which is a necessary condition to perform cluster sampling. An ICC score of 1 would indicate correlation between clusters indicating that it is not suitable for clustering.

*Summary* To summarize, the regression models were not suitable to predict the starting salary of students. Different methods including multiple linear regression, regression tress all indicated that the dataset is not suitable for regression.

The starting salary was converted to categorical variable through labeling. Different methods of classifications were used including QDA and classification trees. They both had the same mis-classification rate of 70%, but QDA only predicted all the salaries as Median salary, which is not desirable. The classification tree, with the same accuracy, was able to predict a few instances of lower salary but none for higher salary. This is can be considered a slightly better method as it allows us to visualize the results in a better manner.

The sampling was done using 3 methods: Simple Random Sampling Without Replacement, Stratified Sampling, and One-Stage Cluster Sampling. The Cluster sampling had the best results and the estimate was very close to the population parameters. This is to generalize that it is better to take a cluster sampling for student data, especially when collected across different fields of study.