Placement Wizard Recruitment Made Easier Using Statistics

Mr. Bhushan Jadhav[1], Mr. Om B Borkar[2], Mr. Anurag S Surve[3], Mr. Kartik Wategaonkar[4], Mr. Yash D Yeola[5], Mr. Shantaram Dhum[6]

[1,2,3,4,5]Students, Sir Parashurambhau College, Pune.
[6]Department of Statistics, Sir Parashurambhau College, Pune.

ABSTRACT

The recruitment process for jobs is often challenging and time consuming. It entails multiple stages, including screening resumes, conducting interviews, and assessing candidates through tests. Recently, machine learning algorithms have gained popularity in the recruitment process as they can predict the likelihood of a candidate's placement based on various factors. In this study, we aimed to develop and compare three machine learning algorithms, including LR, DT, NB, KNN & PCA for predicting the placement status of students in a job recruitment process.

INTRODUCTION

Placement prediction is a crucial task in the field of human resource management, as it plays a significant role in determining the right candidate for the right job. Traditional methods of placement prediction often involve human judgment, which can be subjective and time-consuming. With the advancements in machine learning, it is now possible to automate this process and make it more accurate and efficient. Machine learning algorithms can analyse large amounts of data and extract patterns and relationships between various factors that contribute to a successful placement. These factors can include a candidate's academic qualifications, work experience, skills, and other relevant factors. By leveraging machine learning, organizations can improve the accuracy of their placement predictions and reduce the time and effort required for the hiring process. In this project we will also discuss the challenges and limitations of using machine learning for placement prediction and provide recommendations for future research in this area. Overall, this project aims to provide a comprehensive overview of the current state of placement prediction using machine learning and its potential for improving the hiring process.

ABOUT THE DATASET

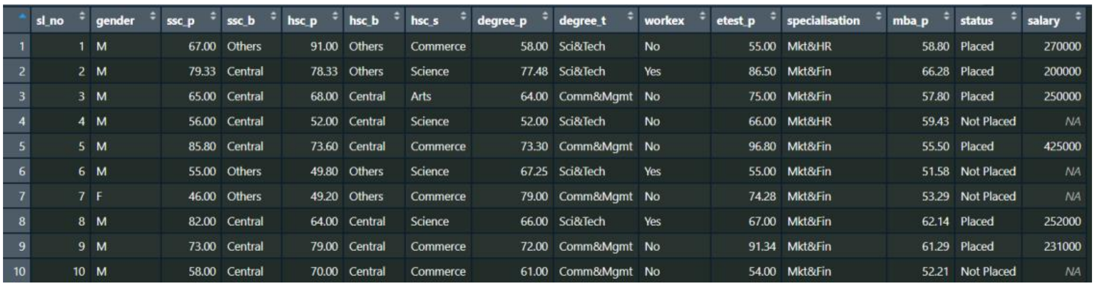
In order to build and compute the various Machine Learning algorithms, we used a data of 215 students involved in the recruitment process of a management college.

We found a data taken from a reputed management institute in India. The data has various variables telling us about any given candidate’s educational and professional background. The variables in the dataset are:

1. Gender – Male(M) or Female(F)
2. Ssc_p – 10th marks in percentage
3. Ssc_b – Board under which 10th exam was given

4. Hsc_p – 12th marks in percentage
5. Hsc_b – Board under which 12th exam was given
6. Hsc_s – Stream of study for 11th and 12th (Science/Commerce/Arts)
7. Degree_p – Marks earned in Undergraduate Degree (%)
8. Degree_t – Trend chosen for UG degree (Science&Tech./Commerce&Management)
9. Workex – Whether the candidate has any previous full time paid work experience before the recruitment process.
10. Etest_p – Score in the MBA entrance test. (%)
11. Specialisation – Area of Specialization in MBA (Mkt&HR / Mkt&Fin)
12. Mba_p – Marks earned in MBA PG Degree (%)
13. Status – Whether the candidate got placed during the recruitment process or not
14. Salary – Salaries (in Rupees) of the placed candidates. “NA” for candidates who were not placed.

Here is a sample of the Dataset used for this project:



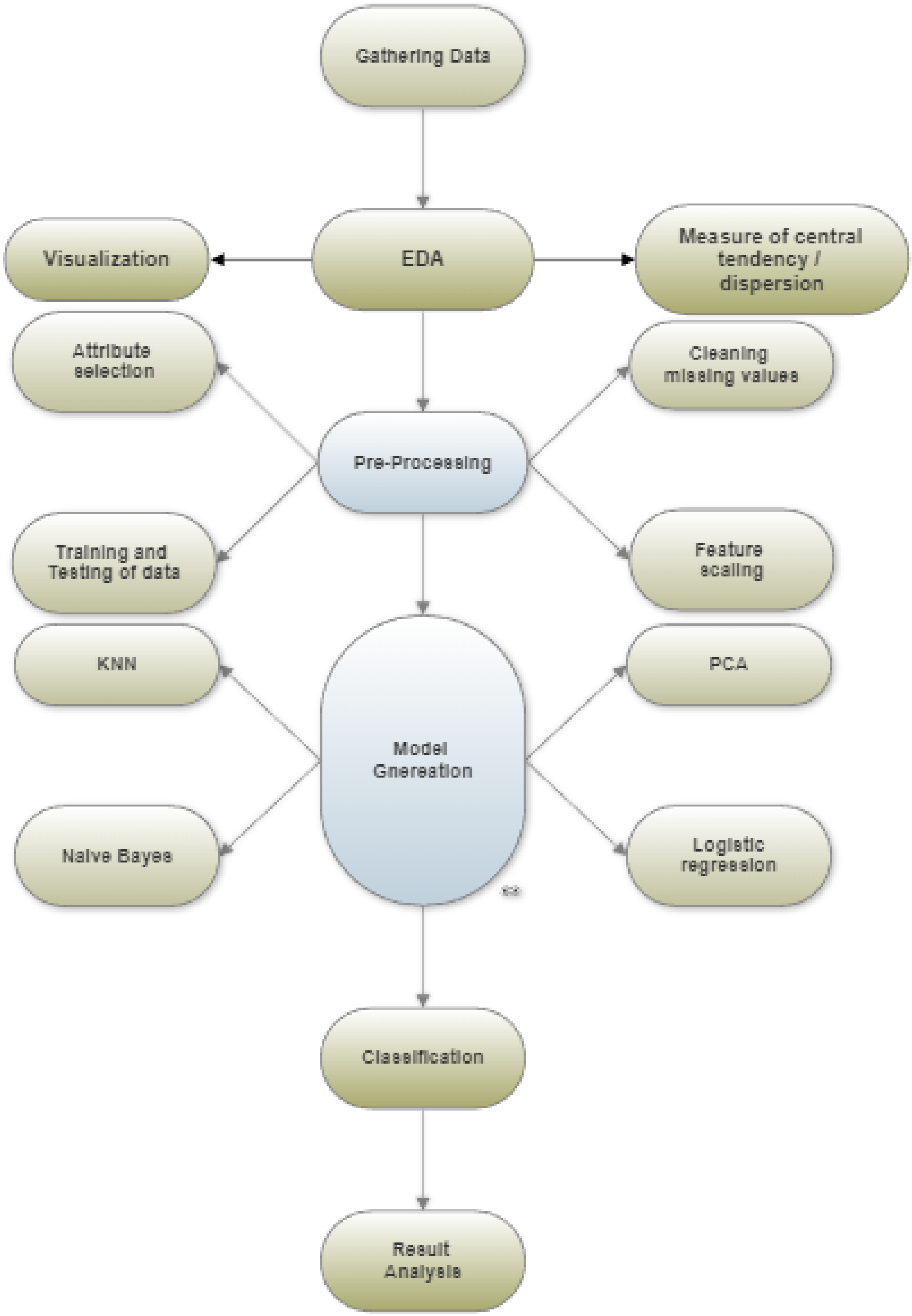
	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
1	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.00	Mkt&HR	58.80	Placed	270000
2	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.50	Mkt&Fin	66.28	Placed	200000
3	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.00	Mkt&Fin	57.80	Placed	250000
4	4	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No	66.00	Mkt&HR	59.43	Not Placed	NA
5	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.80	Mkt&Fin	55.50	Placed	425000
6	6	M	55.00	Others	49.80	Others	Science	67.25	Sci&Tech	Yes	55.00	Mkt&Fin	51.58	Not Placed	NA
7	7	F	46.00	Others	49.20	Others	Commerce	79.00	Comm&Mgmt	No	74.28	Mkt&Fin	53.29	Not Placed	NA
8	8	M	82.00	Central	64.00	Central	Science	66.00	Sci&Tech	Yes	67.00	Mkt&Fin	62.14	Placed	252000
9	9	M	73.00	Central	79.00	Central	Commerce	72.00	Comm&Mgmt	No	91.34	Mkt&Fin	61.29	Placed	231000
10	10	M	58.00	Central	70.00	Central	Commerce	61.00	Comm&Mgmt	No	54.00	Mkt&Fin	52.21	Not Placed	NA

OBJECTIVES

1. To check whether there is any grouping structure in the placed and un-placed candidates
2. To predict placement status based on given variables.
3. To build the classification model for future use.

PROPOSED METHODOLOGY

The overall approach for achieving our objectives is illustrated in the following flowchart:



STATISTICAL TOOLS USED

1. Performing principal component analysis to control dimensions of data.
2. Fitting of logistic regression model to given data.
3. Fitting “k-nearest neighbours” ml model to data.
4. Fitting “naïve bayes” ml model.
5. Fitting Decision Tree Classifier Model.

PROCEDURE

1. CLEANING THE DATA

Data cleaning, also known as data cleansing or data pre-processing, is the process of identifying and correcting or removing errors, inconsistencies, missing values, and irrelevant or noisy data from a dataset. It is an essential step in data analysis and machine learning tasks to ensure the accuracy, reliability, and quality of the data.

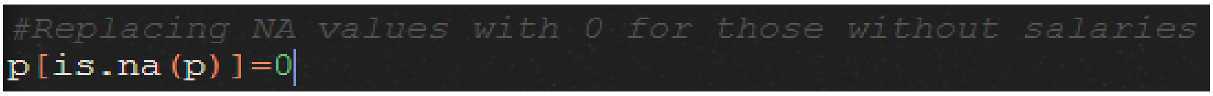
Data cleaning involves several operations, which can vary depending on the characteristics and requirements of the dataset. Here are some common data cleaning tasks:

1. Handling Missing Values
2. Removing Duplicates
3. Correcting Inconsistent Values
4. Handling Outliers
5. Standardizing and Normalizing Data
6. Dealing with Inconsistent or Incomplete Data Structures
7. Handling Irrelevant or Noisy Data

These are just a few examples of common data cleaning tasks. The specific steps and techniques employed in data cleaning will depend on the nature of the data, the analysis goals, and the domain knowledge of the analyst.

In the dataset that we are considering, initially there were several other areas of specialization such as Digital Marketing, Business Analytics, Operations Management, etc. But we decided to consider only 2 Specializations viz a viz. “Marketing and HR” and “Marketing and Finance” in order to simplify the computations and formulations.

There were several NA values in the dataset hence we had to omit those values whenever required. for ex. - Salaries of Unplaced Students were initially “NA” but we replaced salaries as “0” so that our data become computable without changing the meaning of the data.



```
#Replacing NA values with 0 for those without salaries
p[is.na(p)]=0
```

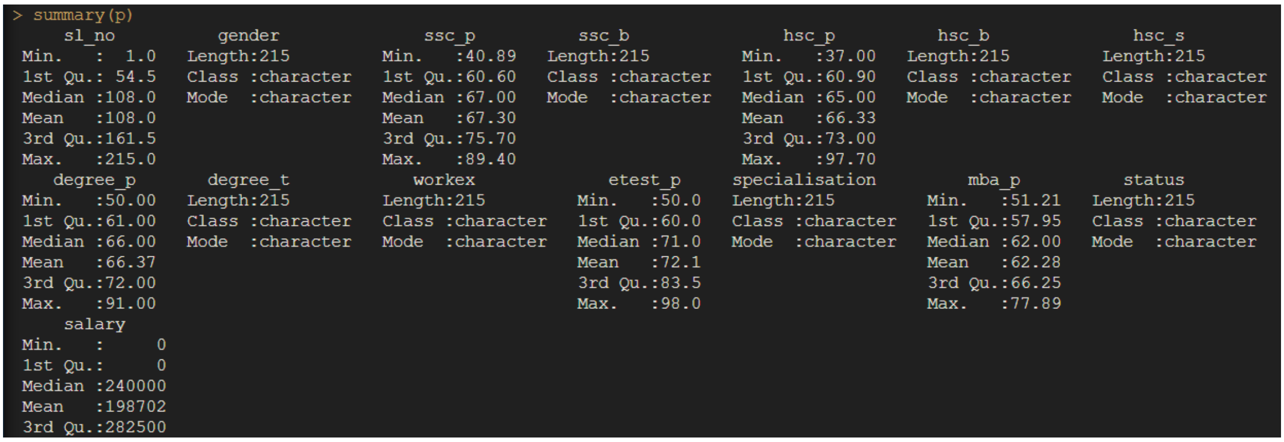
2. EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is a crucial step in the data analysis process that involves examining and summarizing the main characteristics of a dataset. It aims to gain insights, detect patterns, and uncover relationships within the data. EDA techniques include data visualization, summary statistics, and data transformation. By exploring the data, we can understand the distribution, identify outliers, assess missing values, and determine the appropriate pre-processing steps. EDA helps in formulating hypotheses, selecting modelling techniques, and making informed decisions. It serves as a foundation for further statistical analysis, model building, and drawing meaningful conclusions from the data.

❖ Summary statistics

It provides a concise summary of the key features of a dataset, helping to understand its distribution and properties. Measures like mean, median, and mode provide insights into the central tendency of the data, while measures of dispersion such as standard deviation and range indicate the variability and spread. Skewness and kurtosis describe the shape of the distribution. Minimum and maximum values highlight the range of the data. Quartiles offer information about the data's spread and can help detect outliers. Overall, summary statistics provide a quick overview of the dataset's characteristics, enabling initial insights and informing subsequent analysis and decision-making processes.

The summary() function provides a summary of the central tendency, dispersion, and distribution of each variable in the dataset. The output will include the minimum, 1st quartile, median, mean, 3rd quartile, and maximum values for numeric variables. For factor or character variables, it will show the frequency counts of each unique value.



```
> summary(p)
     sl_no          gender              ssc_p          ssc_b               hsc_p          hsc_b              hsc_s
 Min.   :  1.0   Length:215         Min.   :40.89   Length:215         Min.   :37.00   Length:215         Length:215
 1st Qu.: 54.5   Class :character   1st Qu.:60.60   Class :character   1st Qu.:60.90   Class :character   Class :character
 Median :108.0   Mode  :character   Median :67.00   Mode  :character   Median :65.00   Mode  :character   Mode  :character
 Mean   :108.0                      Mean   :67.30                      Mean   :66.33
 3rd Qu.:161.5                      3rd Qu.:75.70                      3rd Qu.:73.00
 Max.   :215.0                      Max.   :89.40                      Max.   :97.70
    degree_p       degree_t            workex             etest_p     specialisation         mba_p          status
 Min.   :50.00   Length:215         Length:215         Min.   :50.0   Length:215         Min.   :51.21   Length:215
 1st Qu.:61.00   Class :character   Class :character   1st Qu.:60.0   Class :character   1st Qu.:57.95   Class :character
 Median :66.00   Mode  :character   Mode  :character   Median :71.0   Mode  :character   Median :62.00   Mode  :character
 Mean   :66.37                                         Mean   :72.1                      Mean   :62.28
 3rd Qu.:72.00                                         3rd Qu.:83.5                      3rd Qu.:66.25
 Max.   :91.00                                         Max.   :98.0                      Max.   :77.89
     salary
 Min.   :     0
 1st Qu.:     0
 Median :240000
 Mean   :198702
 3rd Qu.:282500
```

- We can see maximum variance in the “entrance test percentage” hence we can say that although the mean is observed at 72.
- Measures of Symmetry

	SSC percentage	**HSC percentage**	**Degree percentage**	**Entrance test percentage**	**MBA percentage**
Skewness	-0.131722	0.1624952	0.2432051	0.2803347	0.3113837
Kurtosis	2.378748	3.412572	3.023161	1.908794	2.512376

	SSC percentage	**HSC percentage**	**Degree percentage**	**Entrance test percentage**	**MBA percentage**
Mean	67.30340	66.33316	66.37019	72.10056	62.27819
Variance	117.2284	118.7557	54.1511	176.251	34.02838
Standard Deviation	10.82721	10.89751	7.358743	13.27596	5.833385
Range	40.89- 89.40	37.0- 97.7	50 -91	50 -98	51.21- 77.89

- If skewness is -0.131, it means that the distribution is slightly negatively skewed. This indicates that the tail of the distribution is longer on the left-hand side and that the majority of the data values are clustered towards the right-hand side of the distribution. while all the remaining others are slightly positively skewed.

- HSC percentage and Degree percentage has kurtosis >3 from which we can infer that the distribution is leptokurtic. And for others the distribution is platykurtic since kurtosis <3.

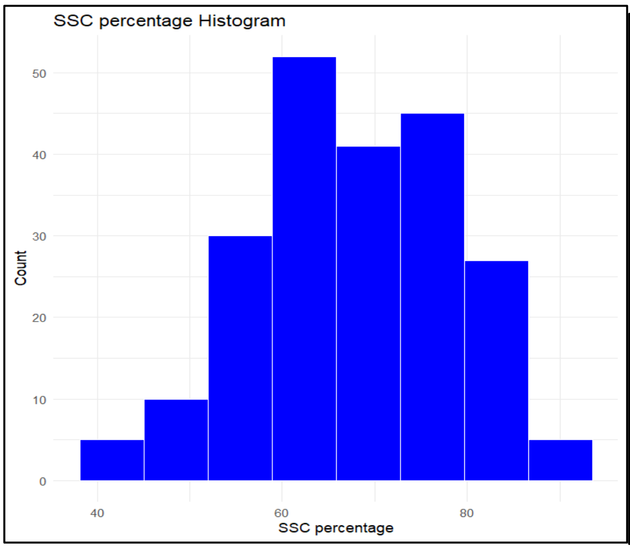
❖ Measures of Dispersion

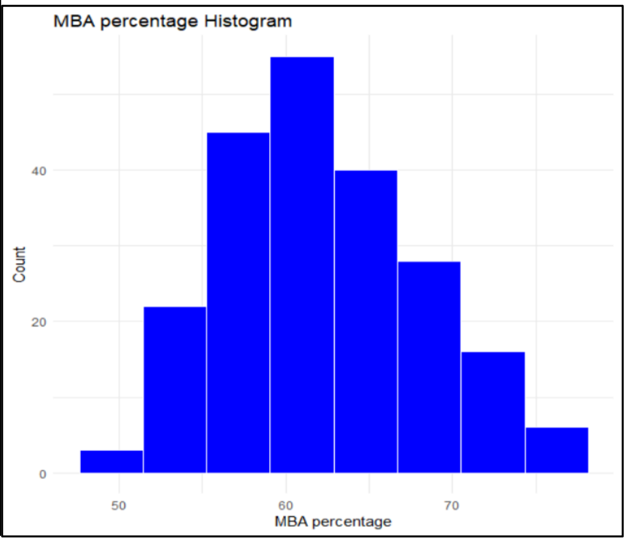
	SSC percentage	HSC percentage	Degree percentage	Entrance test percentage	MBA percentage
Inter quartile range	15.10	12.10	11.00	23.50	8.31
Coefficient of variation	0.16087161	0.16428448	0.11087423	0.18413112	0.09366658
Mean absolute deviation	10.67472	8.89560	7.56126	16.30860	6.22692

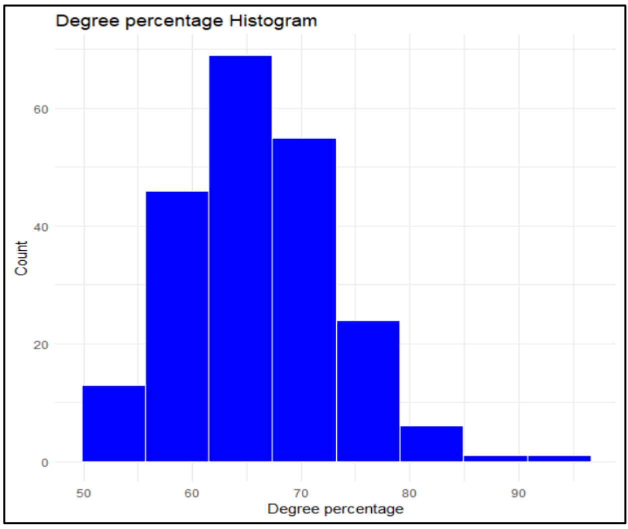
The observing coefficient of variation we compare the variances of all the academic marks. Here the variance of "SSC percentage" and "HSC percentage" are almost same while the variance of "Entrance test" is maximum.

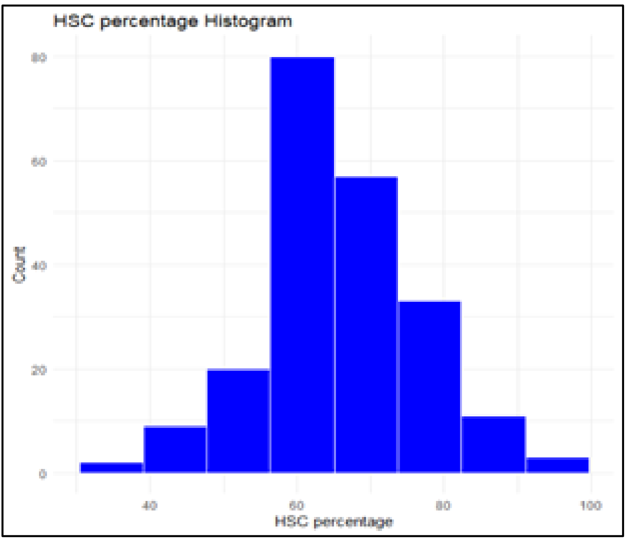
Here a low coefficient of variation (less than 10%) indicates that the variable has relatively low variability or dispersion with respect to the mean.

❖ Data Visualisation:

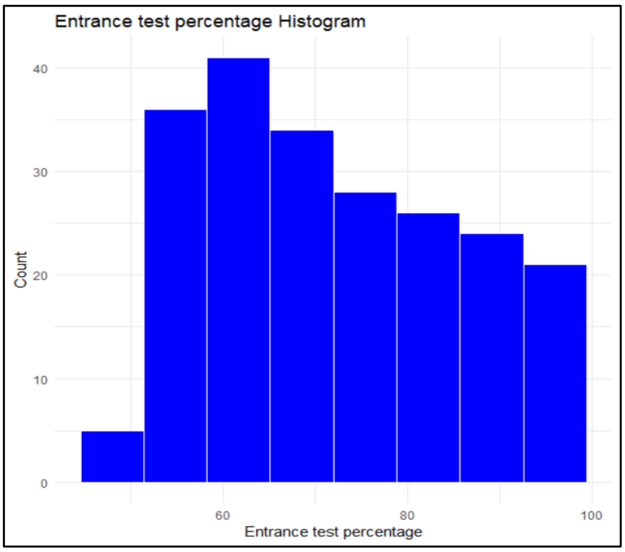




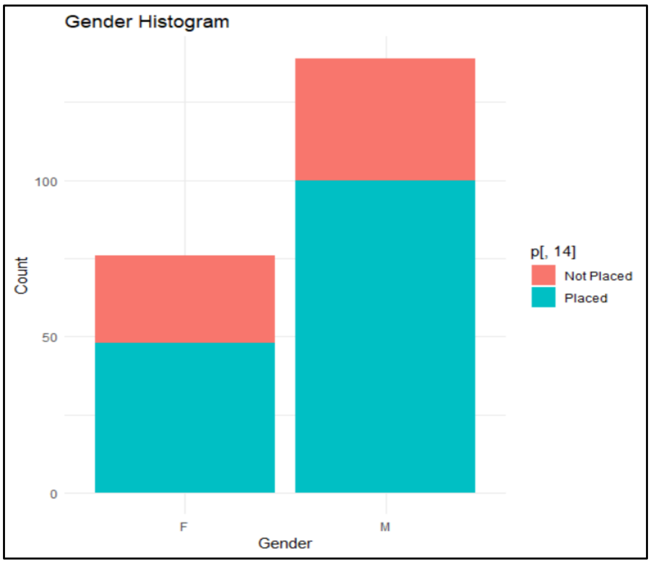




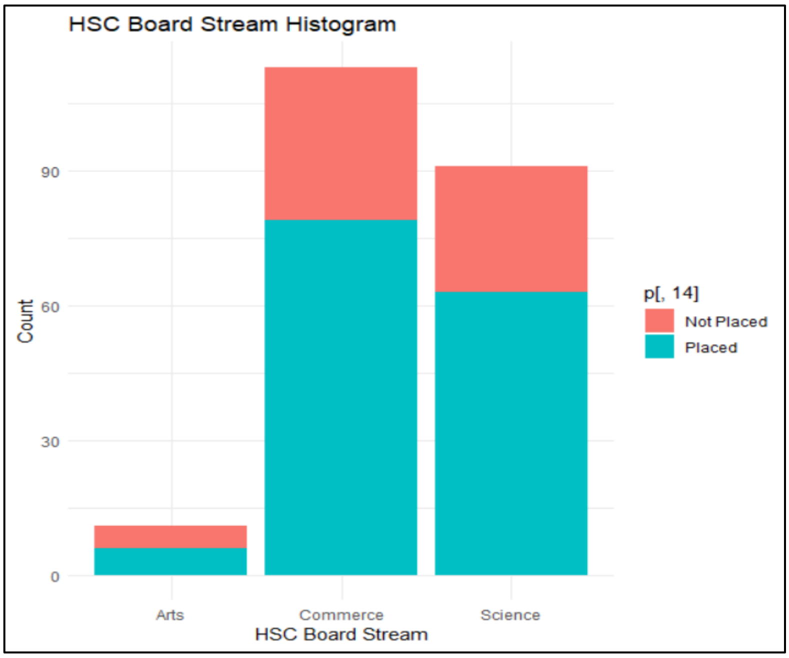
- The histogram of SSC percentages appears to be normally distributed as the shape of the histogram seems symmetric and bell-shaped.
- Also, peak (mode) is always seen in almost middlemost observations for most of the histograms.
- Hence, it can be interpreted that majority of the population scores about an average of 60 percentages. This trend is being seen in all of the 4 histograms.



- The above graph shows histogram of entrance exam percentage. It can be seen that the graph is not symmetric, rather it is positively skewed. In all the earlier graphs, the frequency peaked at 60-70% and then the frequencies dropped drastically in higher percentages. But in this case, there are quite a few students in high percentages.

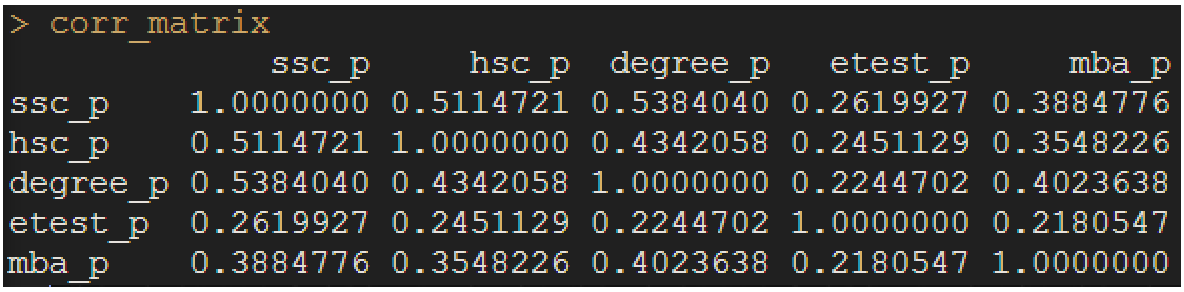


- No. of male students = 139
- No. of female students =76
- No. of male placed =100 and no. of unplaced males=39
- No. of females placed =48 and no. of unplaced females = 28
- The sample proportion of males getting placed (71.9%) is slightly higher than no. of females getting placed (63.1%).
- Also, overall proportion of placements is 68.3 %

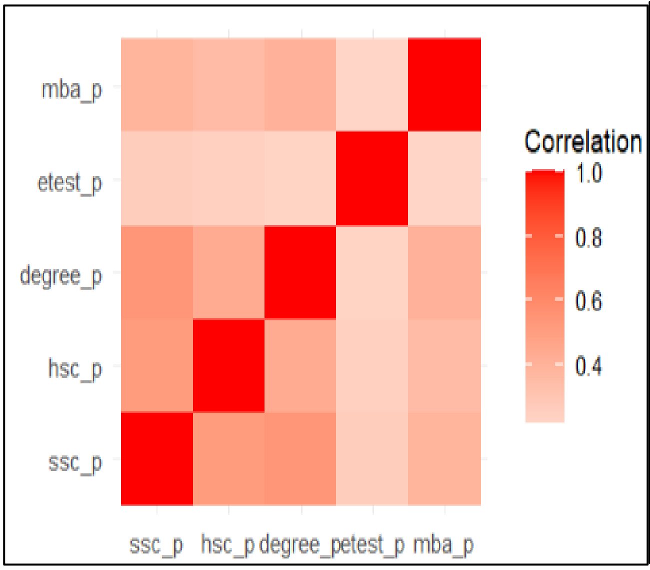


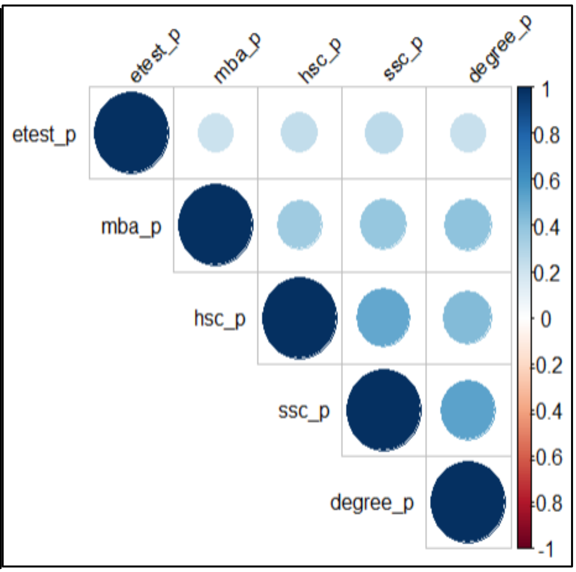
- The above histogram shows the HSC board stream vs status of placements.
- It can be seen that proportion of placements of arts is almost 50 percent ,we cannot conclude any statements because we have very low amount of data of arts stream students.
- From the above graph we can say that there are candidates from all streams opting for MBA.

❖ **Correlation Analysis:**



```
> corr_matrix
             ssc_p     hsc_p  degree_p   etest_p     mba_p
ssc_p    1.0000000 0.5114721 0.5384040 0.2619927 0.3884776
hsc_p    0.5114721 1.0000000 0.4342058 0.2451129 0.3548226
degree_p 0.5384040 0.4342058 1.0000000 0.2244702 0.4023638
etest_p  0.2619927 0.2451129 0.2244702 1.0000000 0.2180547
mba_p    0.3884776 0.3548226 0.4023638 0.2180547 1.0000000
```





- Here, the Correlation matrix shows the correlation between all the numeric variables that is percentages of marks in various exams.
- We know, whenr lies between [-0.3, 0.3], the observations are uncorrelated or have negligible correlation. This generally occurs due to randomness in the data
- There is considerable correlation among all the academic exams whereas it is observed that there is negligible correlation between academic exams and entrance tests.

3. MACHINE LEARNING MODELS

ML model building is the process of creating a computer program that can make predictions or provide insights based on patterns it learns from data.

We first choose an algorithmi.e.,select a method or technique (algorithm) that suits the problem we want to solve. Then, prepare the data: Organize and clean the data, making sure it's in a format that the algorithm can understand. After that, wetrain the model byfeeding the algorithm with labelled data, allowing it to learn patterns and relationships between the input (features) and the output (labels). Then we evaluate the model by assessing how well the model performs by testing it on a separate set of data that it hasn't seen before.

Similarly, we may practice unsupervised learning as well wherein the goal is to discover patterns, structures, or relationships in data without labelled examples.

The ultimate goal of ML Model building is to create a reliable model that can make accurate predictions or provide useful insights when given new, unseen data.

Supervised machine learning models are trained on labelled data, where the inputs and corresponding outputs are provided, enabling the model to learn patterns and make predictions or classifications. In contrast, unsupervised machine learning models work with unlabelled data and aim to discover inherent structures, relationships, or clusters within the data without any predefined target outputs.

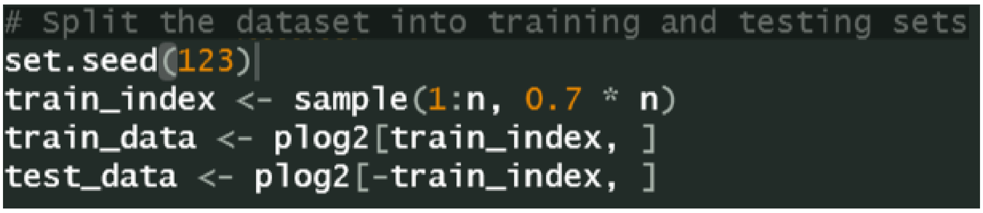
The unsupervised ML model that we use in this project is :

i. PRINCIPAL COMPONENT ANALYSIS

Similarly, the different supervised ML models that we used are as follows:

1. LOGISTIC REGRESSION MODEL
2. NAÏVE BAYES ALGORITHM
3. K-NEAREST NEIGHBOURS ALGORITHM
4. DECISION TREE ALGORITHM

Training and testing data: To prepare the dataset for analysis, we split it into two parts: a training set and a testing set. This was accomplished by creating separate datasets for training and testing. During the training phase, the machine learning model will attempt to comprehend the various correlations present in the training dataset, and the accuracy of the predictions will be assessed. To split the dataset, we utilized an 70-30 ratio, where 70% of the data was reserved for training, and the remaining 30% was used for testing purpose.



```
# Split the dataset into training and testing sets
set.seed(123)
train_index <- sample(1:n, 0.7 * n)
train_data <- plog2[train_index, ]
test_data <- plog2[-train_index, ]
```

PRINCIPAL COMPONENT ANALYSIS

Principal component analysis (PCA) is a statistical technique used to identify patterns in data by reducing the number of variables while retaining the essential information. It does this by finding the principal components, which are linear combinations of the original variables that capture the most variance in the data. PCA is often used for data visualization, dimensionality reduction, and feature extraction. PCA can help in several ways, including:

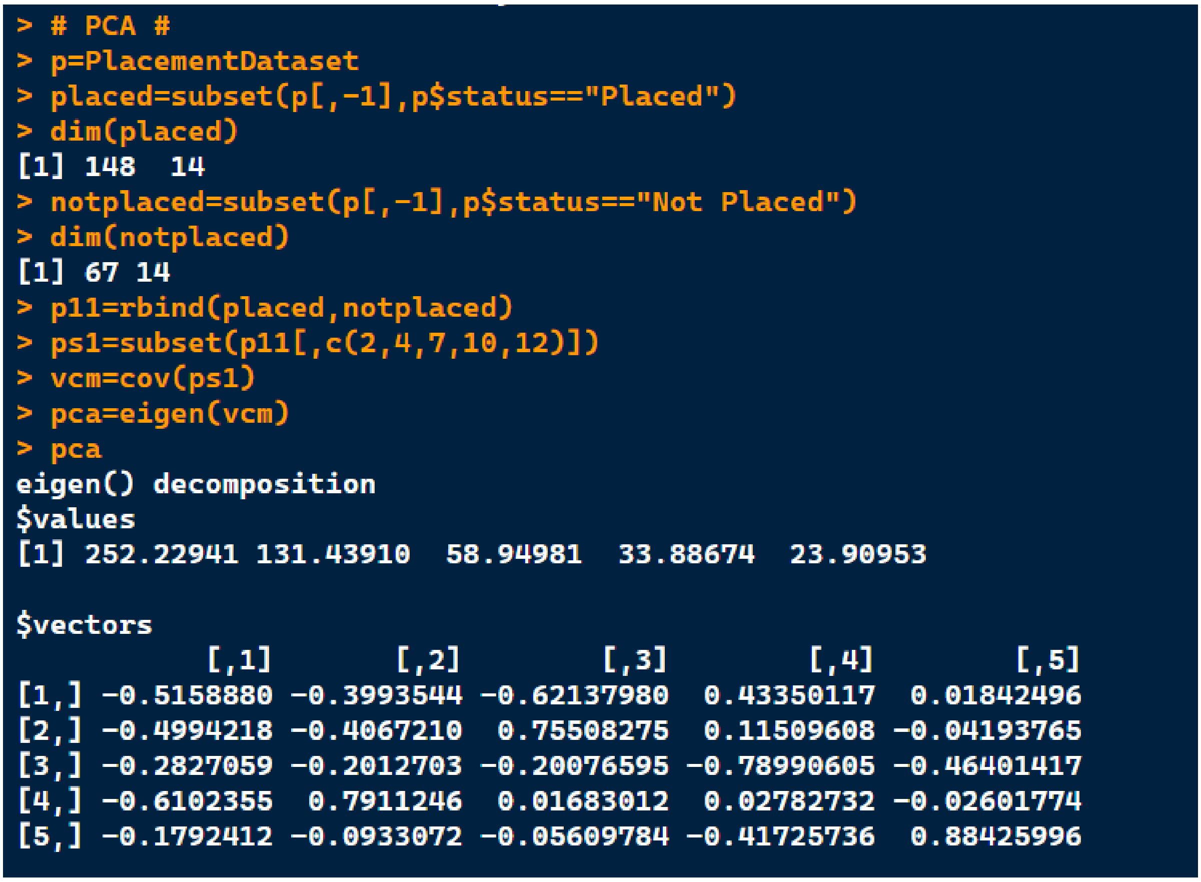
a) Reducing the dimensions of data.

b) Classifying and grouping the data.

c) To detect and avoid multicollinearity.

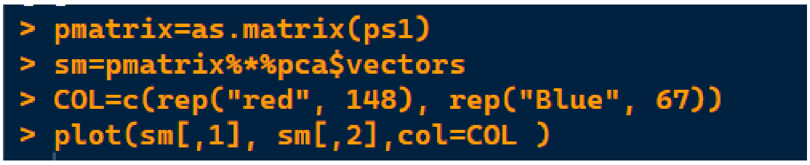
Here, we need to know whether there is grouping structure between the data of placed students and un-placed students

So, now we have data containing all numeric variables (Percentages), where 1st 148 observations are of place students and remaining of unplaced students.

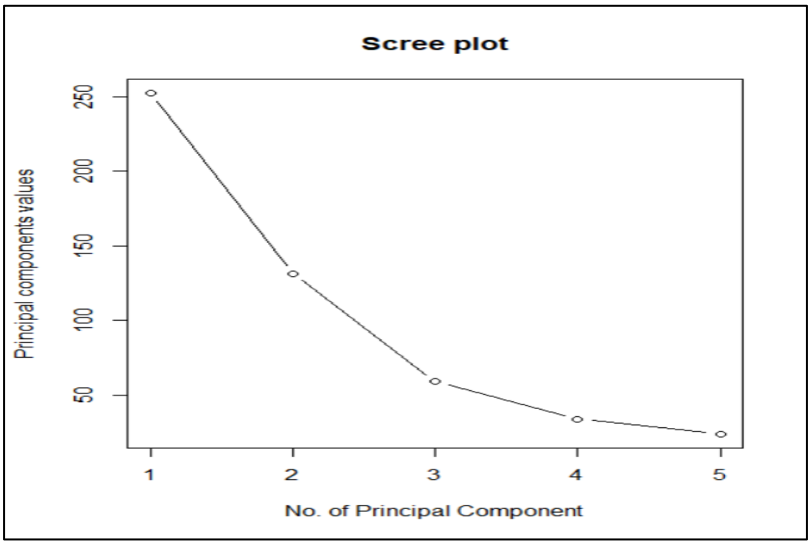


```
> # PCA #
> p=PlacementDataset
> placed=subset(p[,-1],p$status=="Placed")
> dim(placed)
[1] 148  14
> notplaced=subset(p[,-1],p$status=="Not Placed")
> dim(notplaced)
[1] 67 14
> p11=rbind(placed,notplaced)
> ps1=subset(p11[,c(2,4,7,10,12)])
> vcm=cov(ps1)
> pca=eigen(vcm)
> pca
eigen() decomposition
$values
[1] 252.22941 131.43910  58.94981  33.88674  23.90953

$vectors
           [,1]       [,2]        [,3]        [,4]        [,5]
[1,] -0.5158880 -0.3993544 -0.62137980  0.43350117  0.01842496
[2,] -0.4994218 -0.4067210  0.75508275  0.11509608 -0.04193765
[3,] -0.2827059 -0.2012703 -0.20076595 -0.78990605 -0.46401417
[4,] -0.6102355  0.7911246  0.01683012  0.02782732 -0.02601774
[5,] -0.1792412 -0.0933072 -0.05609784 -0.41725736  0.88425996
```

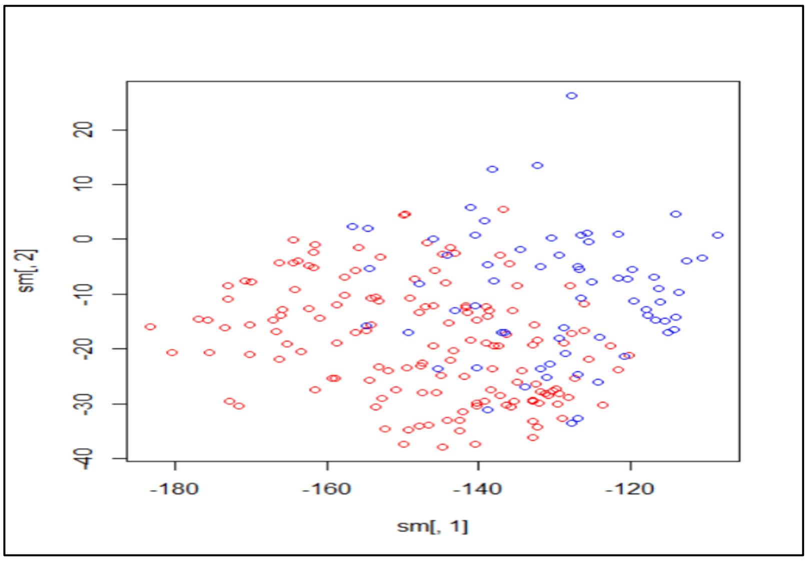
Now, we draw scree plot to check point at which elbow appears. This elbow point helps us to identify number of principal components. Here, the elbow is at k=3, hence we consider the first 2 principal components.



```
> pmatrix=as.matrix(ps1)
> sm=pmatrix%*%pca$vectors
> COL=c(rep("red", 148), rep("Blue", 67))
> plot(sm[,1], sm[,2],col=COL )
```



Here, we get to know that themost of the variation is given by first 2 principal components. now by converting the dataset into matrix and multiplying by Eigen vectors, we get all the linear combinations i.e., all the principal component values



The 2 clusters overlap hence we can say that we fail to differentiate the data into distinct clusters. Hence there is no grouping structure between recruited students and non-recruited students.

Here, we use 5 numeric variables viz a viz ssc_p, hsc_p, degree_p, etest_p, mba_p, mostly related to scores in academic exams. We can say that scores in the academic exams do not help us to classify students into 2 groups Placed and Not Placed.

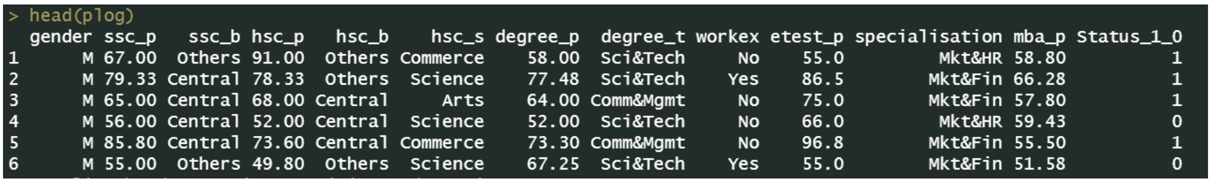
LOGISTIC REGRESSION MODEL

Logistic regression is a statistical method used for binary classification problems. It uses a mathematical function to model the probability of a binary outcome. The model assumes a linear relationship between the input variables and the log-odds of the positive outcome.

The model is used to make predictions on new data by computing the logit and applying the mathematical function to obtain the probability of the positive outcome.

$$\hat{p} = \frac{\exp(b_0 + b_1X_1 + b_2X_2 + \ldots + b_pX_p)}{1 + \exp(b_0 + b_1X_1 + b_2X_2 + \ldots + b_pX_p)}$$

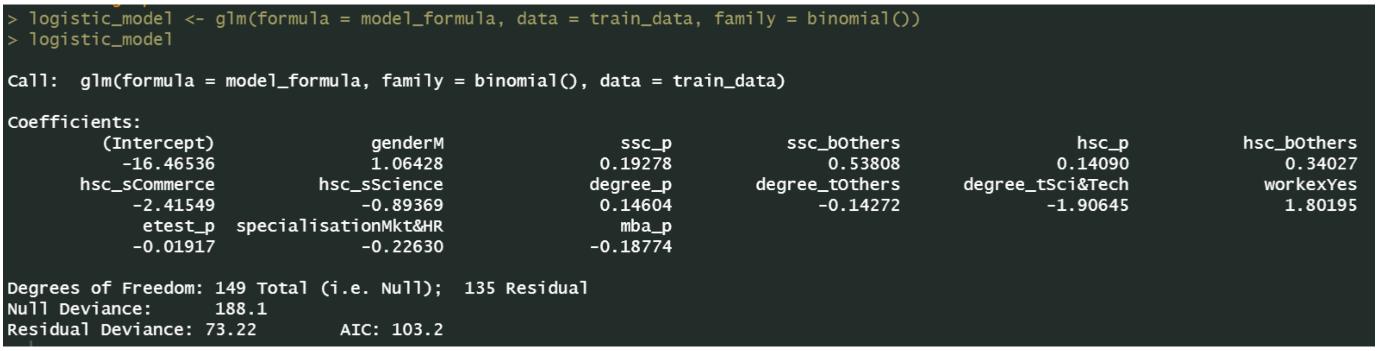
In order to fit the logistic regression model to our data, we prepare the data first. We remove the salary and serial number columns as of now since we do not wish to study it. Then, we transform the “Status” column to a numeric format by assigning value “1” for “Placed” and value “0” for “Not Placed”.



```
> head(plog)
  gender ssc_p   ssc_b hsc_p   hsc_b    hsc_s degree_p  degree_t workex etest_p specialisation mba_p Status_1_0
1      M 67.00  Others 91.00  Others Commerce    58.00  Sci&Tech     No    55.0         Mkt&HR 58.80          1
2      M 79.33 Central 78.33  Others  Science    77.48  Sci&Tech    Yes    86.5        Mkt&Fin 66.28          1
3      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt     No    75.0        Mkt&Fin 57.80          1
4      M 56.00 Central 52.00 Central  Science    52.00  Sci&Tech     No    66.0         Mkt&HR 59.43          0
5      M 85.80 Central 73.60 Central Commerce    73.30 Comm&Mgmt     No    96.8        Mkt&Fin 55.50          1
6      M 55.00  Others 49.80  Others  Science    67.25  Sci&Tech    Yes    55.0        Mkt&Fin 51.58          0
```

We define the logistic regression model using the formula interface. We define all required variables beforehand.

Now, we fit the LOGISTIC REGRESSION MODEL by using the pre-decided Training dataset. We use the "glm" command in R which is a direct command to fit generalized linear models.

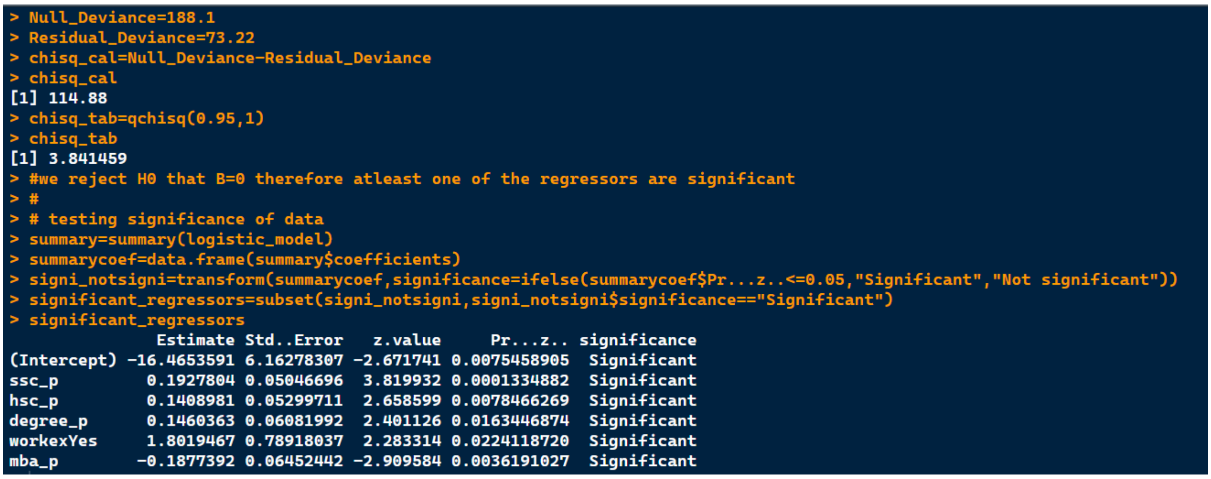


```
> logistic_model <- glm(formula = model_formula, data = train_data, family = binomial())
> logistic_model

Call:  glm(formula = model_formula, family = binomial(), data = train_data)

Coefficients:
        (Intercept)              genderM                ssc_p          ssc_bOthers                hsc_p          hsc_bOthers  
          -16.46536              1.06428              0.19278              0.53808              0.14090              0.34027  
       hsc_sCommerce          hsc_sScience             degree_p       degree_tOthers     degree_tSci&Tech            workexYes  
           -2.41549             -0.89369              0.14604             -0.14272             -1.90645              1.80195  
            etest_p  specialisationMkt&HR                mba_p  
           -0.01917             -0.22630             -0.18774  

Degrees of Freedom: 149 Total (i.e. Null);  135 Residual
Null Deviance:      188.1 
Residual Deviance: 73.22        AIC: 103.2
```

Testing of hypothesis using chi-square test for significance of regressors (at least one of the regressors is significant):

H0 : All the regressors are insignificant

H1 : At least one of the regressor is significant.



```
> Null_Deviance=188.1
> Residual_Deviance=73.22
> chisq_cal=Null_Deviance-Residual_Deviance
> chisq_cal
[1] 114.88
> chisq_tab=qchisq(0.95,1)
> chisq_tab
[1] 3.841459
> #we reject H0 that B=0 therefore atleast one of the regressors are significant
> #
> # testing significance of data
> summary=summary(logistic_model)
> summarycoef=data.frame(summary$coefficients)
> signi_notsigni=transform(summarycoef,significance=ifelse(summarycoef$Pr...z..<=0.05,"Significant","Not significant"))
> significant_regressors=subset(signi_notsigni,signi_notsigni$significance=="Significant")
> significant_regressors
              Estimate Std..Error   z.value     Pr...z.. significance
(Intercept) -16.4653591 6.16278307 -2.671741 0.0075458905  Significant
ssc_p         0.1927804 0.05046696  3.819932 0.0001334882  Significant
hsc_p         0.1408981 0.05299711  2.658599 0.0078466269  Significant
degree_p      0.1460363 0.06081992  2.401126 0.0163446874  Significant
workexYes     1.8019467 0.78918037  2.283314 0.0224118720  Significant
mba_p        -0.1877392 0.06452442 -2.909584 0.0036191027  Significant
```

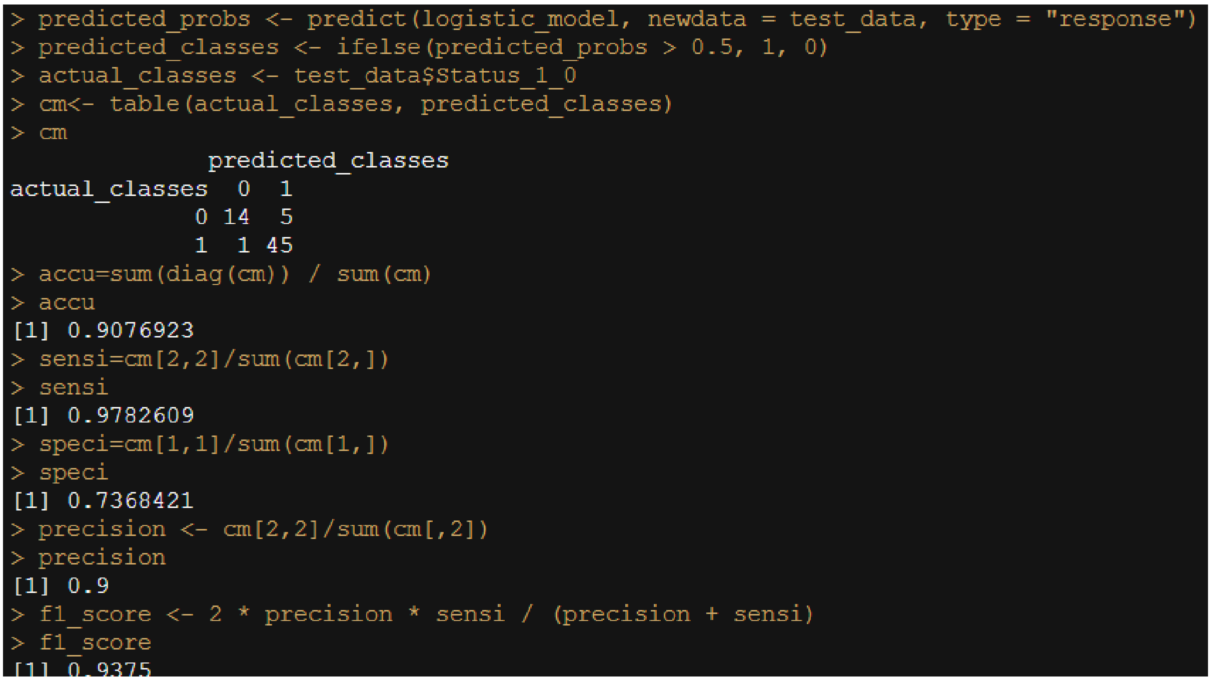
Here, H0 gets rejected, since G=null deviance – residual deviance exceeds chi-square with 12 degrees of freedom at 5 percent level of significance.

Since we found out that at least one of the regressors is significant, we now need to check which of them is significant; we observe that the p-value is less than 0.05.

We conclude that the regressors "ssc_b", "hsc_p", "degree_p", "workex", "mba_p" are significant for the logistic model.

Now we need to evaluate the performance of the model by applying the formulated model on our TESTING DATA. We get the confusion matrix of the test which gives us the True Positives, True Negatives, False Negatives, False Positives of the test.

We use these 4 values to calculate the Accuracy, Specificity, Sensitivity, Precision and F1 Score of our model.



```
> predicted_probs <- predict(logistic_model, newdata = test_data, type = "response")
> predicted_classes <- ifelse(predicted_probs > 0.5, 1, 0)
> actual_classes <- test_data$Status_1_0
> cm<- table(actual_classes, predicted_classes)
> cm
              predicted_classes
actual_classes  0  1
             0 14  5
             1  1 45
> accu=sum(diag(cm)) / sum(cm)
> accu
[1] 0.9076923
> sensi=cm[2,2]/sum(cm[2,])
> sensi
[1] 0.9782609
> speci=cm[1,1]/sum(cm[1,])
> speci
[1] 0.7368421
> precision <- cm[2,2]/sum(cm[,2])
> precision
[1] 0.9
> f1_score <- 2 * precision * sensi / (precision + sensi)
> f1_score
[1] 0.9375
```

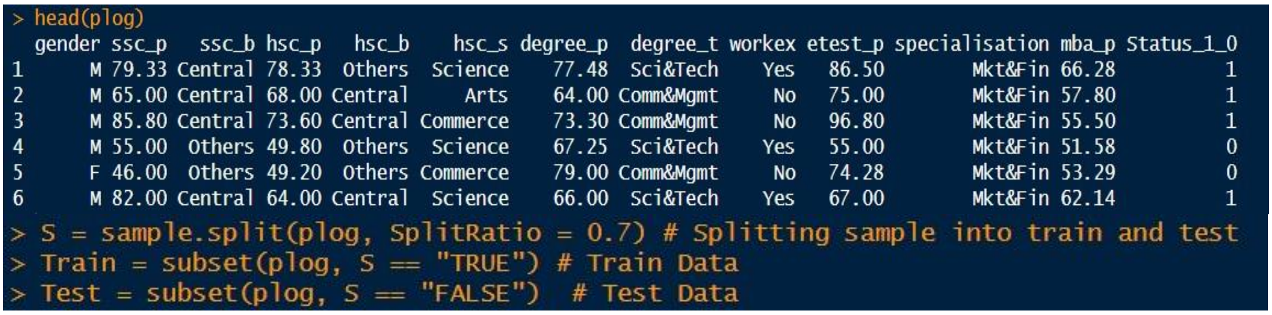
We can see that our formulated Logistic Regression Model is approximately 90.76% accurate.

NAÏVE BAYES ALGORITHM

Naïve Bayes is a machine learning algorithm used for classification problems. It calculates the probability of each class given a set of input features using Bayes' theorem. It assumes that the features are independent of each other, which simplifies the calculations. The algorithm works by calculating the prior probability of each class and the likelihood of each feature given each class. It then combines these probabilities to compute the posterior probability of each class given the input features. The class with the highest posterior probability is then chosen as the predicted class.

To perform the Naïve Bayes algorithm, we first remove the unnecessary columns(Sr. No. and Salary). After cleaning the data, we convert the Status column in 0 (for Not Placed) and 1 (for Placed).

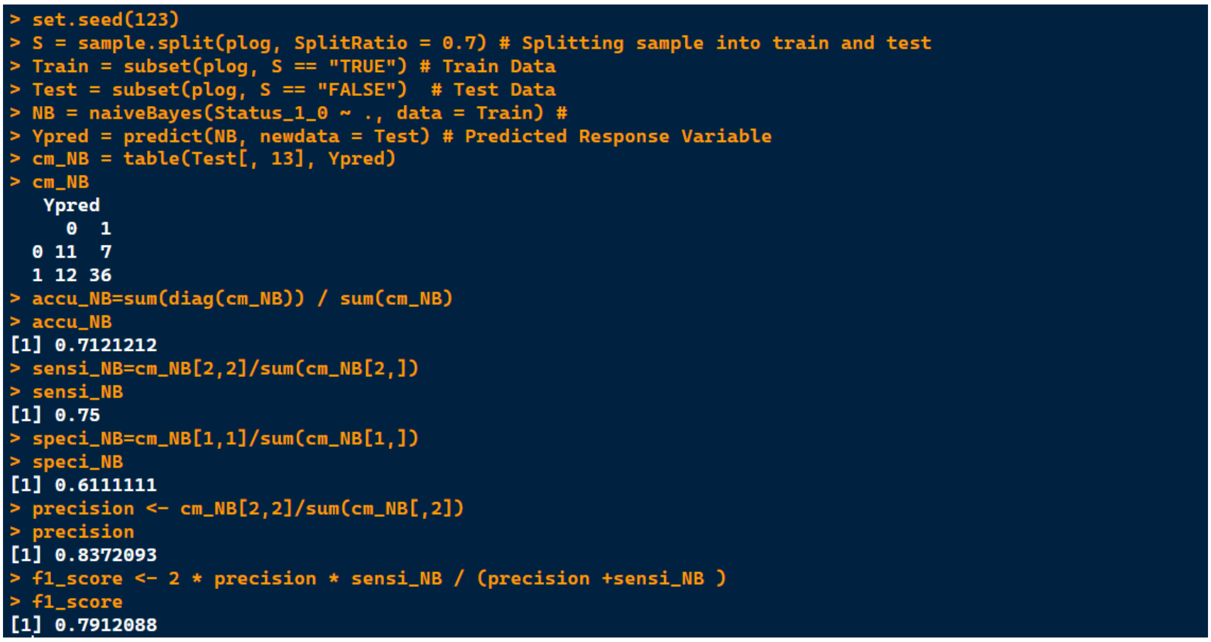
This is the cleaned data:



```
> head(plog)
  gender ssc_p   ssc_b hsc_p   hsc_b    hsc_s degree_p  degree_t workex etest_p specialisation mba_p Status_1_0
1      M 79.33 Central 78.33  Others  Science    77.48  Sci&Tech    Yes   86.50        Mkt&Fin 66.28          1
2      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt     No   75.00        Mkt&Fin 57.80          1
3      M 85.80 Central 73.60 Central Commerce    73.30 Comm&Mgmt     No   96.80        Mkt&Fin 55.50          1
4      M 55.00  Others 49.80  Others  Science    67.25  Sci&Tech    Yes   55.00        Mkt&Fin 51.58          0
5      F 46.00  Others 49.20  Others Commerce    79.00 Comm&Mgmt     No   74.28        Mkt&Fin 53.29          0
6      M 82.00 Central 64.00 Central  Science    66.00  Sci&Tech    Yes   67.00        Mkt&Fin 62.14          1
> S = sample.split(plog, SplitRatio = 0.7) # Splitting sample into train and test
> Train = subset(plog, S == "TRUE") # Train Data
> Test = subset(plog, S == "FALSE")  # Test Data
```

Command for fitting the model is naiveBayes().

After formulating the model, we test the model for its fitness.
The Accuracy, Sensitivity and Specificity of the model is calculated on the basis of confusion matrix.



```
> set.seed(123)
> S = sample.split(plog, SplitRatio = 0.7) # Splitting sample into train and test
> Train = subset(plog, S == "TRUE") # Train Data
> Test = subset(plog, S == "FALSE")  # Test Data
> NB = naiveBayes(Status_1_0 ~ ., data = Train) #
> Ypred = predict(NB, newdata = Test) # Predicted Response Variable
> cm_NB = table(Test[, 13], Ypred)
> cm_NB
   Ypred
     0  1
  0 11  7
  1 12 36
> accu_NB=sum(diag(cm_NB)) / sum(cm_NB)
> accu_NB
[1] 0.7121212
> sensi_NB=cm_NB[2,2]/sum(cm_NB[2,])
> sensi_NB
[1] 0.75
> speci_NB=cm_NB[1,1]/sum(cm_NB[1,])
> speci_NB
[1] 0.6111111
> precision <- cm_NB[2,2]/sum(cm_NB[,2])
> precision
[1] 0.8372093
> f1_score <- 2 * precision * sensi_NB / (precision +sensi_NB )
> f1_score
[1] 0.7912088
```

Naïve BayesAlgorithm is 71.21% accurate for the given data.

K-NEAREST NEIGHBOURS (KNN) ALGORITHM

In a single sentence, nearest neighbours’classifiers are defined by their characteristic of classifying unlabelled examples by assigning them the class of the most similar labelled examples. Despite the simplicity of this idea, nearest neighbours’ methods are extremely powerful. They have been used successfully for several complicated prediction projects.

The k-NN algorithm is a non-parametric method used for classification tasks, where the class membership of a new observation is determined by the majority vote of its k nearest neighbours in the feature space.

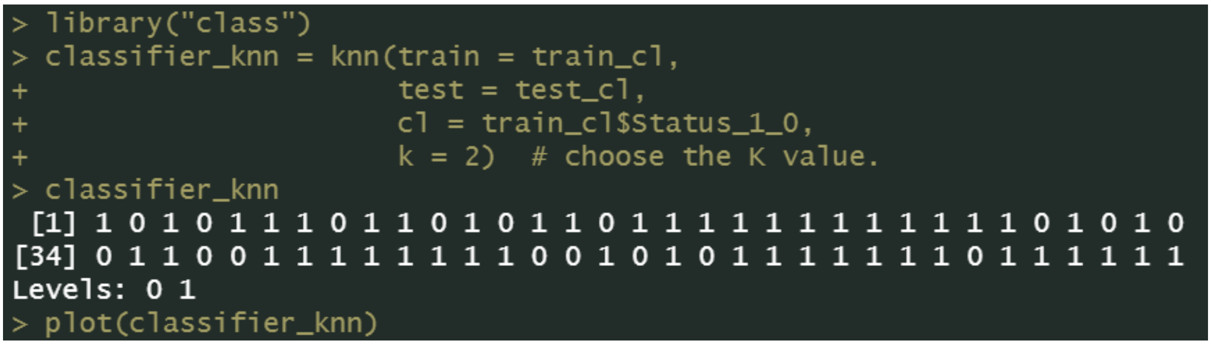
KNN Algorithm:

- The KNN algorithm begins with a training dataset made up of examples that are classified into several categories, as labelled by a nominal variable.
- Assume that we have a test dataset containing unlabelled examples that otherwise have the same features as the training data.
- For each record in the test dataset, KNN identifies k records in the training data that are the "nearest" in similarity, where k is an integer specified in advance.
- The unlabelled test instance is assigned the class of the majority of the k nearest neighbours.

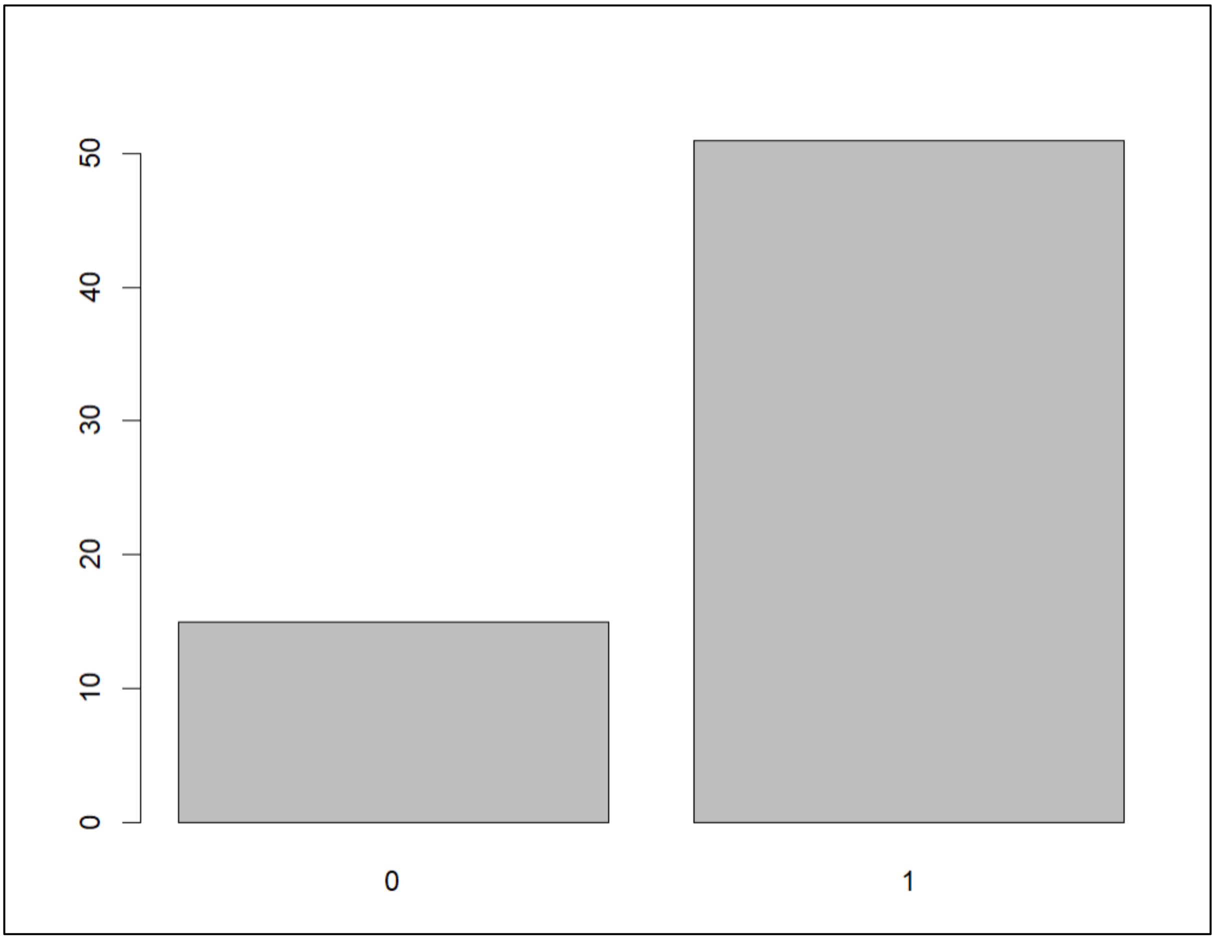
FITTING OF KNN ALGORITHM:

We consider column numbers 2,4,7,10,12,13 only for this model as we wish to work only of the scores of students and their impact on the placement status (1/0).

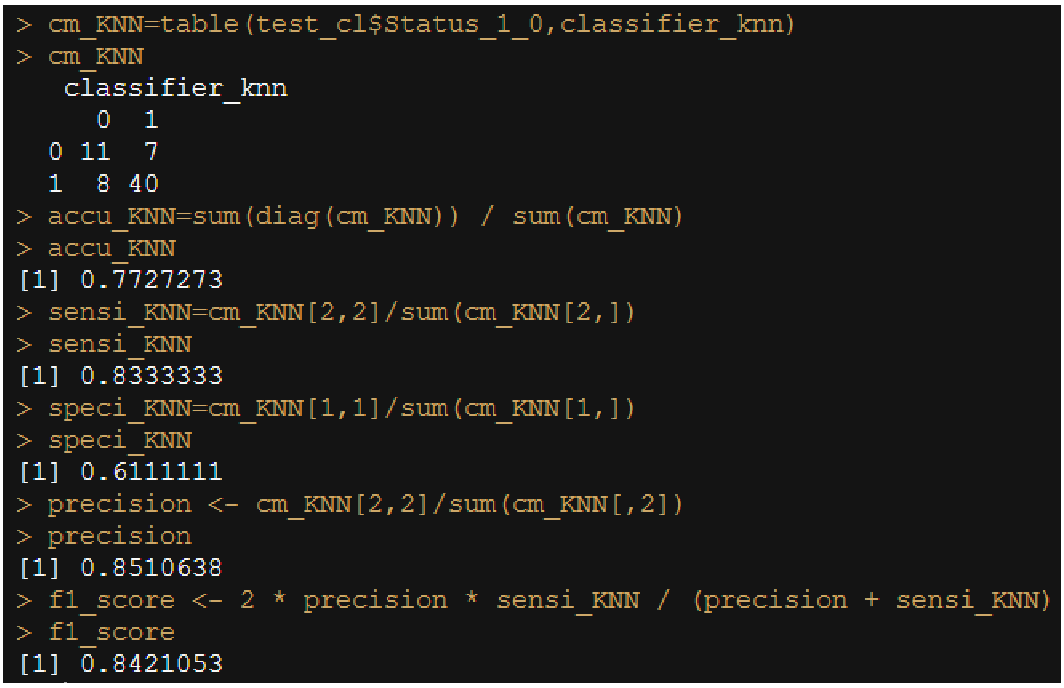
We now Fit the KNN Model to our training dataset using the “knn” command in R. This is a direct command in R to fit the KNN model to any data.



```
> library("class")
> classifier_knn = knn(train = train_cl,
+                      test = test_cl,
+                      cl = train_cl$Status_1_0,
+                      k = 2)  # choose the K value.
> classifier_knn
 [1] 1 0 1 0 1 1 1 0 1 1 0 1 0 1 1 0 1 1 1 1 1 1 1 1 1 1 1 1 0 1 0 1 0
[34] 0 1 1 0 0 1 1 1 1 1 1 1 1 0 0 1 0 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1
Levels: 0 1
> plot(classifier_knn)
```



We can see that based on the learnings from the training data, our model has classified the observations in testing data into “1” and “0”. Now we need to see how accurately this has been done. In order to do that, we find the confusion matrix of the test and from the values of the confusion matrix; we calculate the Accuracy, Specificity, Sensitivity, Precision and F1 Score



```
> cm_KNN=table(test_cl$Status_1_0,classifier_knn)
> cm_KNN
   classifier_knn
     0  1
  0 11  7
  1  8 40
> accu_KNN=sum(diag(cm_KNN)) / sum(cm_KNN)
> accu_KNN
[1] 0.7727273
> sensi_KNN=cm_KNN[2,2]/sum(cm_KNN[2,])
> sensi_KNN
[1] 0.8333333
> speci_KNN=cm_KNN[1,1]/sum(cm_KNN[1,])
> speci_KNN
[1] 0.6111111
> precision <- cm_KNN[2,2]/sum(cm_KNN[,2])
> precision
[1] 0.8510638
> f1_score <- 2 * precision * sensi_KNN / (precision + sensi_KNN)
> f1_score
[1] 0.8421053
```

We observe that our KNN algorithm has an accuracy of about 77.27%.

DECISION TREE

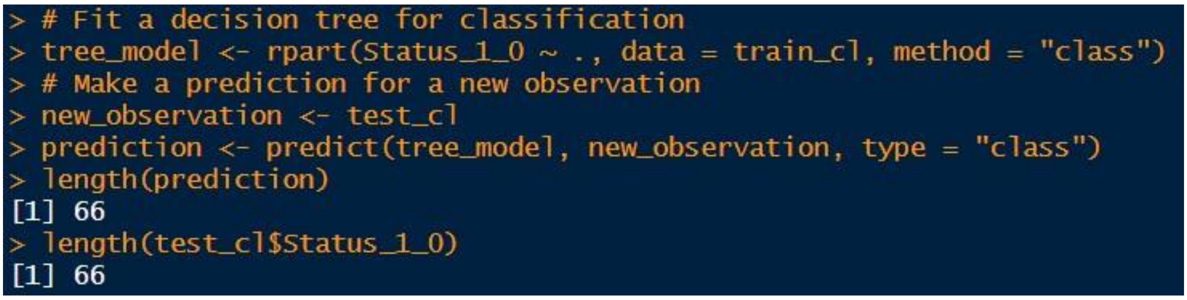
A decision tree is a supervised learning algorithm that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

Decision trees are commonly used in machine learning to solve classification and regression problems. They are also used in decision analysis, where they are known as influence diagrams. For classification problems, the decision tree is used to predict the class of a new data point. For regression problems, the decision tree is used to predict the value of a continuous variable.

To make a prediction, the algorithm starts at the root of the tree and follows the branches down to the leaf that corresponds to the values of the features for the data point being predicted. The outcome at the leaf is then used as the prediction.

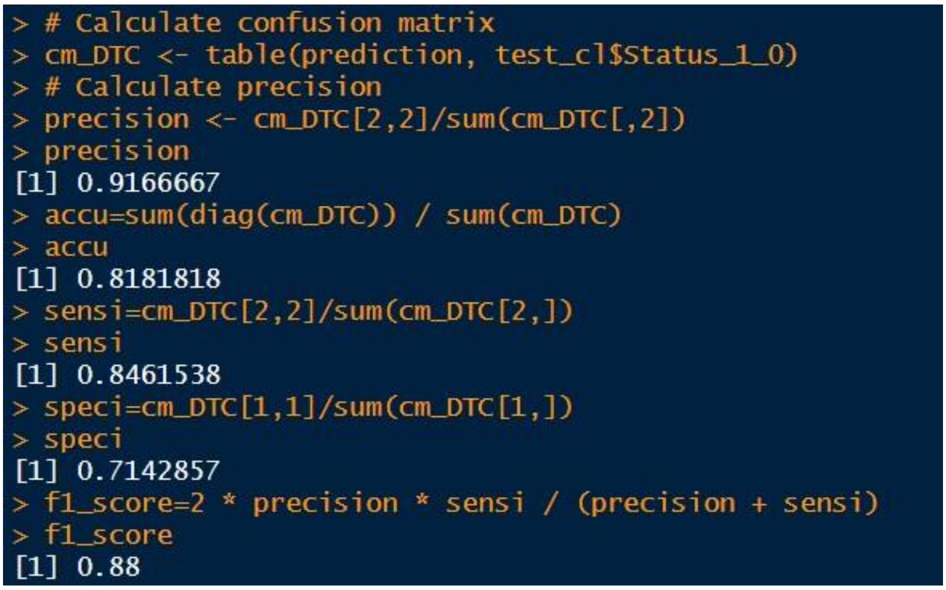
Decision tree algorithms are relatively easy to understand and interpret, and they can be used to solve a wide variety of problems. However, they can also be sensitive to over fitting, which is when the algorithm learns the training data too well and is not able to generalize to new data.

Following is the Decision tree classification for the data, the data here used is the training data which has been obtained earlier. From this data, only classifier variables are considered.



```
> # Fit a decision tree for classification
> tree_model <- rpart(Status_1_0 ~ ., data = train_cl, method = "class")
> # Make a prediction for a new observation
> new_observation <- test_cl
> prediction <- predict(tree_model, new_observation, type = "class")
> length(prediction)
[1] 66
> length(test_cl$Status_1_0)
[1] 66
```

Now, after considering the data, we fit the Decision Tree Classifier Algorithm.

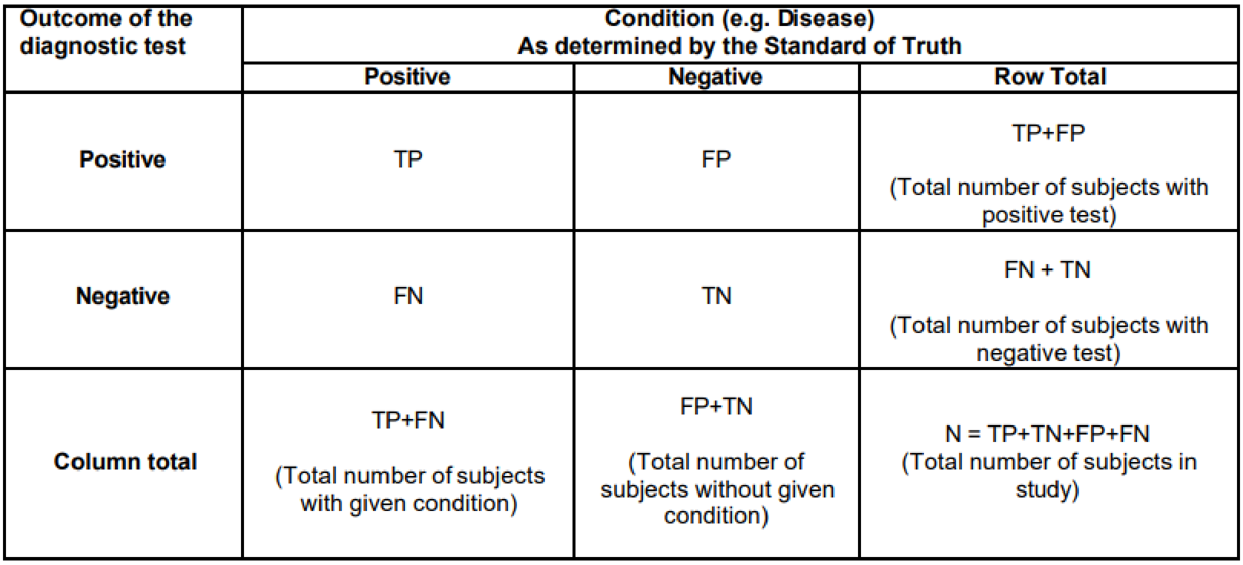


```
> # Calculate confusion matrix
> cm_DTC <- table(prediction, test_cl$Status_1_0)
> # Calculate precision
> precision <- cm_DTC[2,2]/sum(cm_DTC[,2])
> precision
[1] 0.9166667
> accu=sum(diag(cm_DTC)) / sum(cm_DTC)
> accu
[1] 0.8181818
> sensi=cm_DTC[2,2]/sum(cm_DTC[2,])
> sensi
[1] 0.8461538
> speci=cm_DTC[1,1]/sum(cm_DTC[1,])
> speci
[1] 0.7142857
> f1_score=2 * precision * sensi / (precision + sensi)
> f1_score
[1] 0.88
```

After fitting, we check the fitness of the model by obtaining confusion matrix.
Accuracy of Decision Tree Classifier is 81.81 %

COMPARISON OF MODELS

Here the algorithms used by us for classification of prediction are LR, KNN, NB and DT. The classification and comparison of algorithms is done on the basis of Accuracy, Recall, F1 and Precision.



Outcome of the diagnostic test	Condition (e.g. Disease) As determined by the Standard of Truth		
	Positive	Negative	Row Total
Positive	TP	FP	TP+FP (Total number of subjects with positive test)
Negative	FN	TN	FN + TN (Total number of subjects with negative test)
Column total	TP+FN (Total number of subjects with given condition)	FP+TN (Total number of subjects without given condition)	N = TP+TN+FP+FN (Total number of subjects in study)

There are several terms that are commonly used along with the description of sensitivity, specificity and accuracy. They are true positive (TP), true negative (TN), false negative (FN), and false positive (FP).

Sensitivity: In machine learning, sensitivity, also known as recall or true positive rate, measures the proportion of actual positive instances that are correctly identified by the model, indicating its ability to minimize false negatives.

Precision: Precision in machine learning measures the proportion of correctly predicted positive instances out of all instances predicted as positive, highlighting the model's ability to minimize false positives.

Specificity: In machine learning, specificity measures the proportion of actual negative instances that are correctly identified by the model, indicating its ability to minimize false positives and correctly identify true negative instances.

Accuracy: In machine learning, accuracy is a metric that measures the overall correctness of the model's predictions by calculating the ratio of correctly classified instances to the total number of instances in the dataset.

F1 Score: In machine learning, the F1 score is a metric that combines precision and recall (sensitivity) into a single value. It provides a balanced measure of a model's performance by considering both the true positive rate and the positive prediction accuracy.

All these measures are described in terms of TP, TN, FN and FP as follows:

- Sensitivity = TP/(TP + FN)
- Specificity = TN/(TN + FP)
- Accuracy = (TN + TP)/(TN+TP+FN+FP)
- Precision = TP / (TP + FP)
- F1 score = 2*Precision*Sensitivity/(Precision + Sensitivity)

INFERENCE

After computing various machine learning models and testing them for their efficiencies, we get the following output to compare them with each other:

Model	Accuracy	Sensitivity	Specificity	Precision	F1 Score
Logistic Regression	90.76%	97.82%	73.68%	90%	93.75%
Naïve Bayes	71.21%	75%	61.11%	83.72%	79.12%
K-Nearest Neighbours	77.27%	88.88%	52.38%	85.11%	81.22%
Decision Tree Classifier	81.81%	84.61%	71.42%	91.66%	88%

We hereby infer that amongst all the formulated ML models, Logistic Regression Prediction Model has the highest accuracy of 90.76%.

CONCLUSION

From the above table of inferences, we can declare that our Logistic Regression Model is the best model for handling and prediction of data for placement purposes. This model is meant to assist any recruiter to find the ideal candidates for their jobs based on the candidates' educational and professional qualifications.

In general, if the relationship between the input features and the output variable is complex and non-linear, a random forest may perform better than logistic regression. However, if the relationship is simple and linear, like in this case, logistic regression may be sufficient and more interpretable. For our dataset, Logistic Regression works the best because the sample size is small and data is linearly separable. Logistic Regression model performs well when data size is small and decision tree works better when data set is large.

In future it can be extended to create job profile selector which help interviewers reduce any sort of subjectivity or control rush of students at campus drives and shortlist best candidates directly. That can save a lot more time and money for the company.

Machine learning models are a powerful tool that can be used to make predictions from data. However, it is important to remember that machine learning models are only as good as the data that they are trained on. It is also important to carefully consider the limitations of machine learning models, such as their potential to overfit or underfit the data.

There are some obvious drawbacks to this process of using ML assistance such as inability to judge inter-personal skills and other human values of the candidate. Still, this model can help us cut-down the much tedious part if the recruitment process and give us an option to interview and screen the shortlisted candidates in the end.

In conclusion, we would like to say that although Artificial Intelligence cannot completely replace the current way of doing things, it can certainly assist us in increasing efficiency or reducing efforts and this project is a very good proof of the same.

REFERENCES

1. Sane,M.S.(2017).*Regression Analysis*.Nirali Publication
2. Gupta,S.C.Kapoor,V.K.(1989). *Fundamentals of Mathematical Statistics*.Purohit Publication
3. https://en.wikipedia.org/wiki/Principal_component_analysis
4. https://www.ibm.com/topics/naive-bayes#:~:text=The%20Na%C3%AFve%20Bayes%20classifier%20is,a%20given%20class%20or%20category.
5. https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm
6. https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm