Project Details

Dataset Overview

This dataset provides detailed academic and placement-related information for students of a management institute. It is primarily used for analyzing how various academic and personal factors influence a student's chances of securing a job and the salary offered. The additional column named “class” serves as the target variable, classifying each student into two categories: either 0 or 1. Each row in the dataset represents a unique student and captures their educational performance (from secondary school to MBA level), work experience, etc.

Column Descriptions

1. **sl_no**: Serial number — unique identifier for each student. The serial range is 1-216.
2. **gender**: Gender of the student (M for Male, F for Female). There are 65% are Male and 35% are Female.
3. **ssc_p**: Here SSC stand for Secondary School Certificate. The column shows Secondary Education (10th Grade) percentage. Here missing value is 0.
4. **ssc_b**: Board of education for Secondary School, There are Two types of board here, they are: 1) Central, 2) Others.
5. **hsc_p**: Here HSC stand for Higher Secondary Certificate. The column shows Higher Secondary Education (12th Grade) percentage.
6. **hsc_b**: Board of education for Higher Secondary School, There are Two types of board here, they are: 1) Central, 2) Others.
7. **hsc_s**: Specialization stream in Higher Secondary. There are Science, Commerce, Arts which represent three broad fields of study.
8. **degree_p**: Undergraduate degree percentage
9. **degree_t**: Type of undergraduate degree. There are 3 field of degree education, these are Sci&Tech, Comm&Mgmt & Others.
10. **workex**: Work experience before MBA (1.0 = Yes, 0.0 = No).
11. **etest_p**: Percentage in the employability test conducted by the institute.
12. **specialisation**: MBA specialization (Mkt&HR and Mkt&Fin).

13. **mba_p**: MBA percentage.
14. **status**: Placement status those are expressed in two ways Placed and Not Placed
15. **salary**: Salary offered to placed students (in INR); N/A if not placed.
16. **class**: The target column that has two classes: 0 and 1.

Code Segments

Data Loading and Initial Overview

Description:

Reads the CSV file into a dataframe, treating empty strings and "NA" as missing values.

Code:

```
dataset <- read.csv("D:/Study Materials/SPRING_2024-2025/data science/mid_project/Placement_Data_Full_Class - modified.csv",
                    header = TRUE,
                    na.strings = c("", "NA"))
head(dataset)
```

Output:

```
> dataset <- read.csv("D:/Study Materials/SPRING_2024-2025/data science/mid_project/Placement_Data_Full_Class - modified.csv",
+                     header = TRUE,
+                     na.strings = c("", "NA"))
> head(dataset)
  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      0    55.0         Mkt&HR 58.80     Placed 270000
2     2      M 79.33 Central 78.33  Others  Science    77.48  Sci&Tech      1    86.5        Mkt&Fin 66.28     Placed 200000
3     3      M 65.00 Central 68.00 Central     Arts    64.00 Comm&Mgmt      0    75.0        Mkt&Fin 57.80     Placed 250000
4     4      M 56.00 Central 52.00 Central  Science    52.00  Sci&Tech      0    66.0         Mkt&HR 59.43 Not Placed     NA
5     5      M 85.80 Central 73.60 Central Commerce    73.30 Comm&Mgmt      0    96.8        Mkt&Fin 55.50     Placed 425000
6     6      M 55.00  Others 49.80  Others  Science    67.25  Sci&Tech      1    55.0        Mkt&Fin 51.58 Not Placed     NA
  class
1     0
2     0
3     0
4     0
5     1
6     1
```

Summarize Dataset

Description:

Provides a detailed summary of the dataset using skimr.

Code:

```
library(skimr)
skim(dataset)
```

Output:

```
> library(skimr)
> skim(dataset)
── Data Summary ────────────────────────
                           Values
Name                       dataset
Number of rows             216
Number of columns          16
_______________________
Column type frequency:
  character                7
  numeric                  9
________________________
Group variables            None

── Variable type: character ──────────────────────────────────────────────
  skim_variable  n_missing complete_rate min max empty n_unique whitespace
1 gender                 0         1       1   6     0        3          0
2 ssc_b                  0         1       6   7     0        2          0
3 hsc_b                  0         1       6   7     0        2          0
4 hsc_s                  0         1       4   8     0        3          0
5 degree_t               1         0.995   6   9     0        3          0
6 specialisation         0         1       6   7     0        2          0
7 status                 0         1       6  10     0        2          0

── Variable type: numeric ────────────────────────────────────────────────
  skim_variable n_missing complete_rate     mean      sd     p0    p25    p50    p75   p100 hist
1 sl_no                 0         1       108.      62.5      1     54.8   108.   162.    216
2 ssc_p                 0         1        67.0     11.7      0.76  60.4   67     75.2    89.4
3 hsc_p                 0         1        97.1    450.      37     61.0   65     73.0  6680
4 degree_p              0         1        66.3      7.36    50     61     66     72      91
5 workex                1         0.995     0.344    0.476    0      0      0      1       1
6 etest_p               0         1        72.0     13.3     50     60     70.5   83.2    98
7 mba_p                 0         1        62.3      5.82    51.2   58.0   62.0   66.2    77.9
8 salary               67         0.690 288530.   93154.  200000 240000 265000 300000 940000
9 class                 0         1         0.556    0.498    0      0      1      1       1
>
```

Define Column Types

Description:

Separates columns into numeric and categorical types and print them.

Code:

```
numeric_cols <- c("ssc_p", "hsc_p", "degree_p", "etest_p", "salary", "mba_p", "sl_no")
categorical_cols <- c("gender", "ssc_b", "hsc_b", "hsc_s", "degree_t",
                      "specialisation", "status", "class", "workex")
cat("Numeric Columns: \n", numeric_cols,"\n")
cat("Categorical Columns: \n", categorical_cols,"\n")
```

Output:

```
>
> numeric_cols <- c("ssc_p", "hsc_p", "degree_p", "etest_p", "salary", "mba_p", "sl_no")
> categorical_cols <- c("gender", "ssc_b", "hsc_b", "hsc_s", "degree_t",
+                       "specialisation", "status", "class", "workex")
> cat("Numeric Columns: \n", numeric_cols,"\n")
Numeric Columns:
 ssc_p hsc_p degree_p etest_p salary mba_p sl_no
> cat("Categorical Columns: \n", categorical_cols,"\n")
Categorical Columns:
 gender ssc_b hsc_b hsc_s degree_t specialisation status class workex
>
```

Summary Statistics for Columns

Description:

Prints the summary of numeric columns and counts unique values in categorical columns.

Code:

```
print("Summary of Numeric Columns: \n")
for (col in numeric_cols){
  cat("\ncolumn:", col, "\n")
  print(summary(dataset[[col]]))
}

print("unique classes and their length of each attribute of class columns: \n")
for (col in categorical_cols){
  cat("\ncolumn:", col, "\n")
  print(table(dataset[[col]]))
}

```

Output:

```
> print("Summary of Numeric Columns: \n")
[1] "Summary of Numeric Columns: \n"
> for (col in numeric_cols){
+   cat("\ncolumn:", col, "\n")
+   print(summary(dataset[[col]]))
+ }

column: ssc_p
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.76   60.36   67.00   66.95   75.25   89.40

column: hsc_p
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  37.00   60.95   65.00   97.06   73.05 6680.00

column: degree_p
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  50.00   61.00   66.00   66.33   72.00   91.00

column: etest_p
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  50.00   60.00   70.50   72.02   83.25   98.00

column: salary
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
 200000  240000  265000  288530  300000  940000      67

column: mba_p
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  51.21   57.97   61.95   62.26   66.24   77.89

column: sl_no
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   1.00   54.75  108.50  108.50  162.25  216.00
>
```

```
> print("unique classes and their length of each attribute of class columns: \n")
[1] "unique classes and their length of each attribute of class columns: \n"
> for (col in categorical_cols){
+   cat("\ncolumn:", col, "\n")
+   print(table(dataset[[col]]))
+ }

column: gender

     F Female      M
    75      1    140

column: ssc_b

Central  Others
    116     100

column: hsc_b

Central  Others
     84     132

column: hsc_s

    Arts Commerce  Science
      11      114       91

column: degree_t

Comm&Mgmt    Others  Sci&Tech
      144        11        60

column: specialisation

Mkt&Fin  Mkt&HR
    120      96

column: status
```

Check for Duplicates

Description:

Counts and displays the number of duplicate rows.

Code:

```
28  no_of_duplicates <- sum(duplicated(dataset))
29  cat("Number of duplicates row: ", no_of_duplicates, "\n")
30
```

Output:

```
>
> no_of_duplicates <- sum(duplicated(dataset))
> cat("Number of duplicates row: ", no_of_duplicates, "\n")
Number of duplicates row:  0
>
```

Handle Missing Values

Description: Missing values are checked and handled for salary,

workex, and degree_t.

For salary, since the number of missing values matches the number of "Not Placed" students, missing salaries are replaced with 0.

For workex and degree_t, which are categorical features, missing values are filled with their respective most frequent class (mode).

Code:

```
31  cat("Missing Values Before Handling:\n")
32  print(colSums(is.na(dataset)))
33  missing_salary_count <- sum(is.na(dataset$salary))
34  unplaced_count <- sum(dataset$status == "Not Placed", na.rm = TRUE)
35  inconsistent_cases <- dataset[dataset$status == "Not Placed" & !is.na(dataset$salary), ]
36  cat("Total missing salaries:", missing_salary_count, "\n")
37  cat("Total 'Not Placed' students:", unplaced_count, "\n")
38  cat("Number of students that has salary but not placed: ",nrow(inconsistent_cases),"\n")
39  dataset$salary [is.na(dataset$salary)] <- 0
40  workex_mode <- names(which.max(table(dataset$workex)))
41  dataset$workex[is.na(dataset$workex)] <- workex_mode
42  degree_t_mode <- names(which.max(table(dataset$degree_t)))
43  dataset$degree_t[is.na(dataset$degree_t)] <- degree_t_mode
44  cat("\nMissing Values after handling: \n",colSums(is.na(dataset)))
45
```

Output:

```
> cat("ssc_p - Before Cleaning:\n")
ssc_p - Before Cleaning:
> print(summary(dataset$ssc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.76   60.36   67.00   66.95   75.25   89.40
> dataset$ssc_p <- ifelse(dataset$ssc_p < 1, dataset$ssc_p * 100, dataset$ssc_p)
> cat("\nssc_p - After Cleaning:\n")

ssc_p - After Cleaning:
> print(summary(dataset$ssc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  40.89   60.70   67.00   67.30   75.55   89.40
> cat("\nhsc_p - Before Cleaning:\n")

hsc_p - Before Cleaning:
> print(summary(dataset$hsc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  37.00   60.95   65.00   97.06   73.05 6680.00
> dataset$hsc_p <- ifelse(dataset$hsc_p > 100, dataset$hsc_p / 100, dataset$hsc_p)
> cat("\nhsc_p - After Cleaning:\n")

hsc_p - After Cleaning:
> print(summary(dataset$hsc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  37.00   60.95   65.00   66.45   73.00   97.70
>
```

Outliers and Noisy Values

Description:

In the ssc_p column, the value 0.76 appears incorrectly recorded, as a percentage value this low is unrealistic. It is likely meant to be 76. Similarly, in the hsc_p column, the value 6680 is unrealistic for a percentage and was likely intended to be 66.8.

Code:

```
cat("ssc_p - Before Cleaning:\n")
print(summary(dataset$ssc_p))
dataset$ssc_p <- ifelse(dataset$ssc_p < 1, dataset$ssc_p * 100, dataset$ssc_p)
cat("\nssc_p - After Cleaning:\n")
print(summary(dataset$ssc_p))
cat("\nhsc_p - Before Cleaning:\n")
print(summary(dataset$hsc_p))
dataset$hsc_p <- ifelse(dataset$hsc_p > 100, dataset$hsc_p / 100, dataset$hsc_p)
cat("\nhsc_p - After Cleaning:\n")
print(summary(dataset$hsc_p))
```

Output:

```
> cat("ssc_p - Before Cleaning:\n")
ssc_p - Before Cleaning:
> print(summary(dataset$ssc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   0.76   60.36   67.00   66.95   75.25   89.40
> dataset$ssc_p <- ifelse(dataset$ssc_p < 1, dataset$ssc_p * 100, dataset$ssc_p)
> cat("\nssc_p - After Cleaning:\n")

ssc_p - After Cleaning:
> print(summary(dataset$ssc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  40.89   60.70   67.00   67.30   75.55   89.40
> cat("\nhsc_p - Before Cleaning:\n")

hsc_p - Before Cleaning:
> print(summary(dataset$hsc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  37.00   60.95   65.00   97.06   73.05 6680.00
> dataset$hsc_p <- ifelse(dataset$hsc_p > 100, dataset$hsc_p / 100, dataset$hsc_p)
> cat("\nhsc_p - After Cleaning:\n")

hsc_p - After Cleaning:
> print(summary(dataset$hsc_p))
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  37.00   60.95   65.00   66.45   73.00   97.70
```

Invalid Data

Description:

An invalid value "Female" was detected in the gender column and corrected to "F".

Code:

```
cat("Before Validation:", unique(dataset$gender), "\n")
dataset$gender <- ifelse(dataset$gender == "Female", "F", dataset$gender)
cat("After Validation:", unique(dataset$gender), "\n")
```

Output:

```
> cat("Before Validation:", unique(dataset$gender), "\n")
Before Validation: M F Female
> dataset$gender <- ifelse(dataset$gender == "Female", "F", dataset$gender)
> cat("After Validation:", unique(dataset$gender), "\n")
After Validation: M F
```

Filtering Data

Description:

Filters the data by gender and prints the average salary for each gender.

Code:

```
60
61  print("Filtering data to see average salary of males and females: \n")
62  library(dplyr)
63  dataset %>%
64    group_by(gender) %>%
65    summarise(
66      avg_salary = mean(salary)
67    )
68
```

Output:

```
> print("Filtering data to see average salary of males and females: \n")
[1] "Filtering data to see average salary of males and females: \n"
> 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

> dataset %>%
+   group_by(gender) %>%
+   summarise(
+     avg_salary = mean(salary)
+   )
# A tibble: 2 × 2
  gender avg_salary
  <chr>       <dbl>
1 F         168816.
2 M         215436.
```

Categorical to Numerical Conversion

Description:

Converts gender column from categorical to numerical where M is mapped by 1 and F is mapped by 0.

Code:

```
68
69  gender_mapping <- c("M" = 1, "F" = 0)
70  dataset$gender <- gender_mapping[dataset$gender]
71  str(dataset$gender)
72
```

Output:

```
>
> gender_mapping <- c("M" = 1, "F" = 0)
> dataset$gender <- gender_mapping[dataset$gender]
> str(dataset$gender)
 num [1:216] 1 1 1 1 1 1 0 1 1 1 ...
>
```

Normalize Numeric Columns

Description:

Applies Min-Max normalization to numeric columns.

Code:

```
library(dplyr)
min_max_norm <- function(x){
  (x-min(x))/(max(x)-min(x))
}
dataset[numeric_cols] <- lapply(dataset[numeric_cols], min_max_norm)
cat("After Normalizing numerical columns:\n")
print(head(dataset, 5))
```

Output:

```
>
> library(dplyr)
> min_max_norm <- function(x){
+   (x-min(x))/(max(x)-min(x))
+ }
> dataset[numeric_cols] <- lapply(dataset[numeric_cols], min_max_norm)
> cat("After Normalizing numerical columns:\n")
After Normalizing numerical columns:
> print(head(dataset, 5))
        sl_no gender     ssc_p   ssc_b     hsc_p   hsc_b    hsc_s   degree_p  degree_t workex   etest_p specialisation
1 0.000000000      1 0.5382395  Others 0.8896211  Others Commerce 0.19512195  Sci&Tech      0 0.1041667         Mkt&HR
2 0.004651163      1 0.7924139 Central 0.6808896  Others  Science 0.67024390  Sci&Tech      1 0.7604167        Mkt&Fin
3 0.009302326      1 0.4970109 Central 0.5107084 Central     Arts 0.34146341 Comm&Mgmt      0 0.5208333        Mkt&Fin
4 0.013953488      1 0.3114822 Central 0.2471170 Central  Science 0.04878049  Sci&Tech      0 0.3333333         Mkt&HR
5 0.018604651      1 0.9257885 Central 0.6029654 Central Commerce 0.56829268 Comm&Mgmt      0 0.9750000        Mkt&Fin
      mba_p     status    salary class
1 0.2844828     Placed 0.2872340     0
2 0.5648426     Placed 0.2127660     0
3 0.2470015     Placed 0.2659574     0
4 0.3080960 Not Placed 0.0000000     0
5 0.1607946     Placed 0.4521277     1
>
```

Balance the Dataset

Description:

Balances the dataset by undersampling the majority class to match the minority class.

Code:

```
library(dplyr)
class_counts <- dataset %>% count(class)
cat("Before balancing data:\n")
print(class_counts)
minority_class <- class_counts %>% arrange(n) %>% slice(1) %>% pull(class)
majority_class <- class_counts %>% arrange(n) %>% slice(n()) %>% pull(class)
minority_data <- dataset %>% filter(class == minority_class)
majority_data <- dataset %>% filter(class == majority_class) %>% sample_n(nrow(minority_data))
dataset <- bind_rows(minority_data, majority_data)
cat("After balancing data:\n")
print(dataset %>% count(class))

```

Output:

```
> library(dplyr)
> class_counts <- dataset %>% count(class)
> cat("Before balancing data:\n")
Before balancing data:
> print(class_counts)
  class   n
1     0  96
2     1 120
> minority_class <- class_counts %>% arrange(n) %>% slice(1) %>% pull(class)
> majority_class <- class_counts %>% arrange(n) %>% slice(n()) %>% pull(class)
> minority_data <- dataset %>% filter(class == minority_class)
> majority_data <- dataset %>% filter(class == majority_class) %>% sample_n(nrow(minority_data))
> dataset <- bind_rows(minority_data, majority_data)
> cat("After balancing data:\n")
After balancing data:
> print(dataset %>% count(class))
  class  n
1     0 96
2     1 96
>
```

Measure of Central Tendencies

Description:

Calculates and prints mean, median, and mode for two numeric columns- ssc_p and degree_p. Also, mode has been identified for two categorical columns- ssc_b and hsc_b.

Code:

```
 93  cat("ssc_p Central Tendencies\n")
 94  cat("Mean:", mean(dataset$ssc_p), "\n")
 95  cat("Median:", median(dataset$ssc_p), "\n")
 96  cat("Mode:", as.numeric(names(which.max(table(dataset$ssc_p)))), "\n")
 97  cat("degree_p Central Tendencies\n")
 98  cat("Mean:", mean(dataset$degree_p), "\n")
 99  cat("Median:", median(dataset$degree_p), "\n")
100  cat("Mode:", as.numeric(names(which.max(table(dataset$degree_p)))), "\n")
101  cat("ssc_b Mode:", names(which.max(table(dataset$ssc_b))), "\n")
102  cat("hsc_b Mode:", names(which.max(table(dataset$hsc_b))), "\n")
103
```

Output:

```
>
> cat("ssc_p Central Tendencies\n")
ssc_p Central Tendencies
> cat("Mean:", mean(dataset$ssc_p), "\n")
Mean: 67.30199
> cat("Median:", median(dataset$ssc_p), "\n")
Median: 67
> cat("Mode:", as.numeric(names(which.max(table(dataset$ssc_p)))), "\n")
Mode: 62
> cat("degree_p Central Tendencies\n")
degree_p Central Tendencies
> cat("Mean:", mean(dataset$degree_p), "\n")
Mean: 66.33144
> cat("Median:", median(dataset$degree_p), "\n")
Median: 66
> cat("Mode:", as.numeric(names(which.max(table(dataset$degree_p)))), "\n")
Mode: 65
> cat("ssc_b Mode:", names(which.max(table(dataset$ssc_b))), "\n")
ssc_b Mode: Central
> cat("hsc_b Mode:", names(which.max(table(dataset$hsc_b))), "\n")
hsc_b Mode: Others
>
```

Result Interpretation:

ssc_p Central Tendencies:

- The average percentage in SSC is around 67. Students mostly scored around 67%, with the most frequent score being 62%.

degree_p Central Tendencies:

- The average percentage in Degree exams is about 66. Students typically scored near 66%, with 65% being the most common score. **ssc_b Mode:**
- Most students completed their SSC education under the Central board.

hsc_b Mode:

- For HSC, most students studied under boards categorized as Others.

Measure of Spread

Description:

Calculates range, interquartile range (IQR), variance, and standard deviation for ssc_p and hsc_p.

Code:

```
cat("SSC_P Statistics\n")
cat("Range:", paste(range(dataset$ssc_p), collapse = " to "), "\n")
cat("IQR:", IQR(dataset$ssc_p), "\n")
cat("Variance:", var(dataset$ssc_p), "\n")
cat("Standard deviation:", sd(dataset$ssc_p), "\n")
cat("HSC_P Statistics\n")
cat("Range:", paste(range(dataset$hsc_p), collapse = " to "), "\n")
cat("IQR:", IQR(dataset$hsc_p), "\n")
cat("Variance:", var(dataset$hsc_p), "\n")
cat("Standard deviation:", sd(dataset$hsc_p), "\n")

```

Output:

```
> cat("SSC_P Statistics\n")
SSC_P Statistics
> cat("Range:", paste(range(dataset$ssc_p), collapse = " to "), "\n")
Range: 40.89 to 89.4
> cat("IQR:", IQR(dataset$ssc_p), "\n")
IQR: 14.85
> cat("Variance:", var(dataset$ssc_p), "\n")
Variance: 116.6836
> cat("Standard deviation:", sd(dataset$ssc_p), "\n")
Standard deviation: 10.80202
> cat("HSC_P Statistics\n")
HSC_P Statistics
> cat("Range:", paste(range(dataset$hsc_p), collapse = " to "), "\n")
Range: 37 to 97.7
> cat("IQR:", IQR(dataset$hsc_p), "\n")
IQR: 12.05
> cat("Variance:", var(dataset$hsc_p), "\n")
Variance: 121.0203
> cat("Standard deviation:", sd(dataset$hsc_p), "\n")
Standard deviation: 11.00092
>
```

Result Interpretation:

SSC_P Statistics:

- SSC percentages span from about 41% to 89%.
- Middle 50% of SSC scores are spread across 14.85 percentage points.
- There is a moderately high variability in SSC scores.
- On average, SSC scores deviate about 10.8 points from the mean.

HSC_P Statistics:

- HSC percentages range from 37% to about 98%.
- Middle 50% of HSC scores are spread across 12.05 percentage points.
- Variability in HSC scores is slightly higher than in SSC scores.
- On average, HSC scores deviate about 11 points from the mean.

Feature Selection

Description:

Removes the unnecessary column- sl_no from the dataset.

Code:

```
115  library(dplyr)
116  dataset <- dataset %>% select(-sl_no)
117  head(dataset,5)
118
```

Output:

```
> library(dplyr)
> dataset <- dataset %>% select(-sl_no)
> head(dataset,5)
  gender     ssc_p   ssc_b     hsc_p   hsc_b    hsc_s   degree_p  degree_t workex   etest_p specialisation     mba_p
1      1 0.5382395  Others 0.8896211  Others Commerce 0.19512195  Sci&Tech      0 0.1041667         Mkt&HR 0.2844828
2      1 0.7924139 Central 0.6808896  Others  Science 0.67024390  Sci&Tech      1 0.7604167        Mkt&Fin 0.5648426
3      1 0.4970109 Central 0.5107084 Central     Arts 0.34146341 Comm&Mgmt      0 0.5208333        Mkt&Fin 0.2470015
4      1 0.3114822 Central 0.2471170 Central  Science 0.04878049  Sci&Tech      0 0.3333333         Mkt&HR 0.3080960
5      1 0.8474541 Central 0.4448105 Central  Science 0.39024390  Sci&Tech      1 0.3541667        Mkt&Fin 0.4096702
     status    salary class
1    Placed 0.2872340     0
2    Placed 0.2127660     0
3    Placed 0.2659574     0
4 Not Placed 0.0000000     0
5    Placed 0.2680851     0
>
```

Train Test Split

Description:

Randomly split the dataset into 80% training and 20% testing data.

Code:

```
119 set.seed(123)
120 train_indices <- sample(1:nrow(dataset), size = 0.8 * nrow(dataset))
121 train_data <- dataset[train_indices, ]
122 test_data <- dataset[-train_indices, ]
123 cat("Training rows:", nrow(train_data), "\nTesting rows:", nrow(test_data))
```

Output:

```
>
> set.seed(123)
> train_indices <- sample(1:nrow(dataset), size = 0.8 * nrow(dataset))
> train_data <- dataset[train_indices, ]
> test_data <- dataset[-train_indices, ]
> cat("Training rows:", nrow(train_data), "\nTesting rows:", nrow(test_data))
Training rows: 153
Testing rows: 39
>
```