AMERICAN INTERNATIONAL UNIVERSITY-BANGLADESH

Faculty of Science and Technology

Department of Computer Science

Course Name: Introduction to Data Science

Final term Project

Section: B

Prepared by-

Name	ID
Mrinmoy Das	20-43856-2
Helen Chora Chowdhury	20-43996-2
Limia Sadina Sathi	20-43851-2

Submitted to:

TOHEDUL ISLAM
Assistant Professor
Faculty of Science & Technology
American International University-Bangladesh

Dataset Source: https://www.kaggle.com/datasets/uciml/adult-census-income

About the dataset: The Adult Census Income dataset comprises demographic and socioeconomic information about individuals, featuring columns such as age, work class, education, marital status, occupation, relationship, race, sex, capital gains, capital losses, hours per week worked, native country, and income. Each entry represents an individual's characteristics, including employment details, education level, marital status, and more. The dataset is often used in machine learning tasks, where the goal is to predict whether an individual's income exceeds $50,000 based on the provided features. It serves as a benchmark for binary classification problems, with "income" being the target variable indicating whether the person earns more than $50,000 or not.

Import the data set as csv and print the data set:

```
data<- read.csv("D:/adult.csv",header = TRUE,sep = ",")
data
```

Output:

	age	workclass	fnlwgt	education	education.num	marital.status	occupation	relationship	race	sex
1	90	?	77053	HS-grad	9	Widowed	?	Not-in-family	White	Female
2	82	Private	132870	HS-grad	9	Widowed	Exec-managerial	Not-in-family	White	Female
3	66	?	186061	Some-college	10	Widowed	?	Unmarried	Black	Female
4	54	Private	140359	7th-8th	4	Divorced	Machine-op-inspct	Unmarried	White	Female
5	41	Private	264663	Some-college	10	Separated	Prof-specialty	Own-child	White	Female
6	34	Private	216864	HS-grad	9	Divorced	Other-service	Unmarried	White	Female
7	38	Private	150601	10th	6	Separated	Adm-clerical	Unmarried	White	Male
8	74	State-gov	88638	Doctorate	16	Never-married	Prof-specialty	Other-relative	White	Female
9	68	Federal-gov	422013	HS-grad	9	Divorced	Prof-specialty	Not-in-family	White	Female
10	41	Private	70037	Some-college	10	Never-married	Craft-repair	Unmarried	White	Male
11	45	Private	172274	Doctorate	16	Divorced	Prof-specialty	Unmarried	Black	Female
12	38	Self-emp-not-inc	164526	Prof-school	15	Never-married	Prof-specialty	Not-in-family	White	Male
13	52	Private	129177	Bachelors	13	Widowed	Other-service	Not-in-family	White	Female
14	32	Private	136204	Masters	14	Separated	Exec-managerial	Not-in-family	White	Male
15	51	?	172175	Doctorate	16	Never-married	?	Not-in-family	White	Male
16	46	Private	45363	Prof-school	15	Divorced	Prof-specialty	Not-in-family	White	Male
17	45	Private	172822	11th	7	Divorced	Transport-moving	Not-in-family	White	Male
18	57	Private	317847	Masters	14	Divorced	Exec-managerial	Not-in-family	White	Male
19	22	Private	119592	Assoc-acdm	12	Never-married	Handlers-cleaners	Not-in-family	Black	Male
20	34	Private	203034	Bachelors	13	Separated	Sales	Not-in-family	White	Male
21	37	Private	188774	Bachelors	13	Never-married	Exec-managerial	Not-in-family	White	Male
22	29	Private	77009	11th	7	Separated	Sales	Not-in-family	White	Female
23	61	Private	29059	HS-grad	9	Divorced	Sales	Unmarried	White	Female
24	51	Private	153870	Some-college	10	Married-civ-spouse	Transport-moving	Husband	White	Male
25	61	?	135285	HS-grad	9	Married-civ-spouse	?	Husband	White	Male
26	21	Private	34310	Assoc-voc	11	Married-civ-spouse	Craft-repair	Husband	White	Male
27	33	Private	228696	1st-4th	2	Married-civ-spouse	Craft-repair	Not-in-family	White	Male
28	49	Private	122066	5th-6th	3	Married-civ-spouse	Other-service	Husband	White	Male
29	37	Self-emp-inc	107164	10th	6	Never-married	Transport-moving	Not-in-family	White	Male
30	38	Private	175360	10th	6	Never-married	Prof-specialty	Not-in-family	White	Male
31	23	Private	44064	Some-college	10	Separated	Other-service	Not-in-family	White	Male
32	59	Self-emp-inc	107287	10th	6	Widowed	Exec-managerial	Unmarried	White	Female
33	52	Private	198863	Prof-school	15	Divorced	Exec-managerial	Not-in-family	White	Male
34	51	Private	123011	Bachelors	13	Divorced	Exec-managerial	Not-in-family	White	Male
35	60	Self-emp-not-inc	205246	HS-grad	9	Never-married	Exec-managerial	Not-in-family	Black	Male

Description: Here is the code of importing the dataset as csv file. It is the output of the dataset which is imported in RStudio.

Structure of Data:

Code:

```
str(data)
```

Output:

```
'data.frame':   32561 obs. of  15 variables:
 $ age           : int  90 82 66 54 41 34 38 74 68 41 ...
 $ workclass     : chr  "?" "Private" "?" "Private" ...
 $ fnlwgt        : int  77053 132870 186061 140359 264663 216864 150601 88638 422013 70037 ...
 $ education     : chr  "HS-grad" "HS-grad" "Some-college" "7th-8th" ...
 $ education.num : int  9 9 10 4 10 9 6 16 9 10 ...
 $ marital.status: chr  "Widowed" "Widowed" "Widowed" "Divorced" ...
 $ occupation    : chr  "?" "Exec-managerial" "?" "Machine-op-inspct" ...
 $ relationship  : chr  "Not-in-family" "Not-in-family" "Unmarried" "Unmarried" ...
 $ race          : chr  "White" "White" "Black" "White" ...
 $ sex           : chr  "Female" "Female" "Female" "Female" ...
 $ capital.gain  : int  0 0 0 0 0 0 0 0 0 0 ...
 $ capital.loss  : int  4356 4356 4356 3900 3900 3770 3770 3683 3683 3004 ...
 $ hours.per.week: int  40 18 40 40 40 45 40 20 40 60 ...
 $ native.country: chr  "United-States" "United-States" "United-States" "United-States" ...
 $ income        : chr  "<=50K" "<=50K" "<=50K" "<=50K" ...
```

Description: The dataset comprises 32,561 observations with 15 variables in an R data frame format. Each observation represents an individual, featuring information such as age, workclass, education, marital status, occupation, relationship, race, sex, capital gains, capital losses, hours per week worked, native country, and income. The dataset includes integers for age, education numerical representation, final weight, capital gains, capital losses, and hours per week. Workclass, education, marital status, occupation, relationship, race, sex, native country, and income are represented as character vectors. The dataset is characterized by missing values denoted as "?" in certain categorical variables like workclass and occupation. The goal of the dataset is often to predict whether an individual's income is above or below $50,000, with the income variable denoted as a character vector ("<50K" or "<=50K").

Descriptive Statistics Using summary () Function:

Code:

```
summary(data)
```

Output:

```
      age          workclass            fnlwgt          education          education.num   marital.status      occupation
 Min.   :17.00   Length:32561       Min.   :  12285   Length:32561       Min.   : 1.00   Length:32561       Length:32561
 1st Qu.:28.00   Class :character   1st Qu.: 117827   Class :character   1st Qu.: 9.00   Class :character   Class :character
 Median :37.00   Mode  :character   Median : 178356   Mode  :character   Median :10.00   Mode  :character   Mode  :character
 Mean   :38.58                      Mean   : 189778                      Mean   :10.08
 3rd Qu.:48.00                      3rd Qu.: 237051                      3rd Qu.:12.00
 Max.   :90.00                      Max.   :1484705                      Max.   :16.00
 relationship          race              sex             capital.gain    capital.loss    hours.per.week  native.country
 Length:32561       Length:32561       Length:32561       Min.   :    0   Min.   :   0.0   Min.   : 1.00   Length:32561
 Class :character   Class :character   Class :character   1st Qu.:    0   1st Qu.:   0.0   1st Qu.:40.00   Class :character
 Mode  :character   Mode  :character   Mode  :character   Median :    0   Median :   0.0   Median :40.00   Mode  :character
                                                          Mean   : 1078   Mean   :  87.3   Mean   :40.44
                                                          3rd Qu.:    0   3rd Qu.:   0.0   3rd Qu.:45.00
                                                          Max.   :99999   Max.   :4356.0   Max.   :99.00

   income
 Length:32561
 Class :character
 Mode  :character
```

Description: Here using descriptive statistic, and the summary() function. In the output min, max, median, and mean are shown for every column.

Dropping Columns:

Code:

```
data <- data[ , -which(names(data) %in% c("education.num", "fnlwgt"))]
data
```

Output:

	age	workclass	education	marital.status	occupation	relationship	race	sex	capital.gain
1	90	?	HS-grad	Widowed	?	Not-in-family	White	Female	0
2	82	Private	HS-grad	Widowed	Exec-managerial	Not-in-family	White	Female	0
3	66	?	Some-college	Widowed	?	Unmarried	Black	Female	0
4	54	Private	7th-8th	Divorced	Machine-op-inspct	Unmarried	White	Female	0
5	41	Private	Some-college	Separated	Prof-specialty	Own-child	White	Female	0
6	34	Private	HS-grad	Divorced	Other-service	Unmarried	White	Female	0
7	38	Private	10th	Separated	Adm-clerical	Unmarried	White	Male	0
8	74	State-gov	Doctorate	Never-married	Prof-specialty	Other-relative	White	Female	0
9	68	Federal-gov	HS-grad	Divorced	Prof-specialty	Not-in-family	White	Female	0
10	41	Private	Some-college	Never-married	Craft-repair	Unmarried	White	Male	0
11	45	Private	Doctorate	Divorced	Prof-specialty	Unmarried	Black	Female	0
12	38	Self-emp-not-inc	Prof-school	Never-married	Prof-specialty	Not-in-family	White	Male	0
13	52	Private	Bachelors	Widowed	Other-service	Not-in-family	White	Female	0
14	32	Private	Masters	Separated	Exec-managerial	Not-in-family	White	Male	0
15	51	?	Doctorate	Never-married	?	Not-in-family	White	Male	0

Description: In this code, the dataframe is being modified by dropping two columns: "education.num" and "fnlwgt." The reason for dropping these columns is mentioned as "this data is not useful for this model." Specifically, in the context of the "adult census income" prediction model, it appears that the features "education.num" and "fnlwgt" are considered irrelevant or unnecessary for the model's training or analysis. Removing these columns helps streamline the dataset and potentially improve the model's performance by eliminating extraneous or less informative features.

Minimum & Maximum age

Code:

```
min_age <- min(data$age)
max_age <- max(data$age)
cat("Minimum Age:", min_age, "\n")
cat("Maximum Age:", max_age, "\n")
```

Output:

```
> cat("Minimum Age:", min_age, "\n")
Minimum Age: 17
> cat("Maximum Age:", max_age, "\n")
Maximum Age: 90
```

Description: The dataset's age range spans from a minimum of 17 years to a maximum of 90 years. These values provide insights into the range of ages present in our dataset.

Converting numeric to categorical data for Age

Code:

```
data$age <- case_when(
data$age <= 17 ~ "Adolescence",
data$age <= 24 ~ "Young Adult",
data$age <= 64 ~ "Adult",
data$age >= 65 ~ "Senior")
View(data)
```

Output:

	age	workclass	education	marital.status	occupation	relationship	race
1	Senior	?	HS-grad	Widowed	?	Not-in-family	White
2	Senior	Private	HS-grad	Widowed	Exec-managerial	Not-in-family	White
3	Senior	?	Some-college	Widowed	?	Unmarried	Black
4	Senior	Private	7th-8th	Divorced	Machine-op-inspct	Unmarried	White
5	Senior	Private	Some-college	Separated	Prof-specialty	Own-child	White
6	Senior	Private	HS-grad	Divorced	Other-service	Unmarried	White
7	Senior	Private	10th	Separated	Adm-clerical	Unmarried	White
8	Senior	State-gov	Doctorate	Never-married	Prof-specialty	Other-relative	White
9	Senior	Federal-gov	HS-grad	Divorced	Prof-specialty	Not-in-family	White
10	Senior	Private	Some-college	Never-married	Craft-repair	Unmarried	White

Description: In this code, a categorical variable "age" is created in the dataframe based on the values in the existing "age" column. The case_when function is used to assign categories to different age ranges. The categories include "Adolescence" for ages 17 and below, "Young Adult" for ages 18 to 24, "Adult" for ages 25 to 64, and "Senior" for ages 65 and above. This transformation allows for a more interpretable representation of age groups in the dataset, which can be useful for analysis and modeling tasks. The resulting modified dataframe can be viewed using the View(data) function.

Minimum & Maximum for working hour

Code:

```
filtered_data <- data[is.finite(data$hours.per.week), ]
min_hour <- min(filtered_data$hours.per.week)
max_hour <- max(filtered_data$hours.per.week)
cat("Minimum Hour:", min_hour, "\n")
cat("Maximum Hour:", max_hour, "\n")
```

Output:

```
> cat("Minimum Hour:", min_hour, "\n")
Minimum Hour: 1
> cat("Maximum Hour:", max_hour, "\n")
Maximum Hour: 99
```

Description: In this code, the cat function is used to print the minimum and maximum values of a variable named "hour." The minimum value is displayed with the label "Minimum Hour" and the maximum value with the label "Maximum Hour." In this case, the minimum hour is printed as "1" and the maximum hour as "99."

Converting numeric to categorical data for hours per week

Code:

```
data$hours.per.week <- case_when(
data$hours.per.week <= 20 ~ "Medium",
data$hours.per.week <= 40 ~ "Good",
data$hours.per.week <= 60 ~ "active",
data$hours.per.week <= 99 ~ "Hard worker")
View(data)
```

Output:

capital.loss	hours.per.week	native.country
4356	Good	United-States
4356	Medium	United-States
4356	Good	United-States
3900	Good	United-States
3900	Good	United-States
3770	active	United-States
3770	Good	United-States
3683	Medium	United-States
3683	Good	United-States
3004	active	?
3004	Good	United-States

Description: In this code, a new categorical variable "hours.per.week" is created in the data dataframe based on the values in the existing "hours.per.week" column. The categories include "Medium," "Good," "Active," and "Hard worker," representing different ranges of weekly working hours. This transformation simplifies the representation of working hours for analysis or modeling purposes.

Result for hour per week:

```
table(data$hours.per.week)

  active        Good Hard worker      Medium
    8471       20052        1110        2928
```

Description: This is the result of the hour per week people work.

Minimum & Maximum for capital gain

Code:

```
min_cgain <- min(data$capital.gain)
max_cgain <- max(data$capital.gain)
cat("Minimum Capital gain:", min_cgain, "\n")
cat("Maximum Capital gain:", max_cgain, "\n")
```

Output:

```
> cat("Minimum Capital gain:", min_cgain, "\n")
Minimum Capital gain: 0
> cat("Maximum Capital gain:", max_cgain, "\n")
Maximum Capital gain: 99999
```

Description: In this code, the cat function is used to print the minimum and maximum values of a variable named "Capital gain." The minimum value is displayed as "0," and the maximum value is displayed as "99999."

Converting numeric to categorical data for capital gain

Code:

```
data$capital.gain <- case_when(
data$capital.gain <= 0 ~ "No Capital Gain",
data$capital.gain <= 5000 ~ "Low",
data$capital.gain <= 20000 ~ "Moderate",
data$capital.gain <= 50000 ~ "High",
data$capital.gain <= 99999 ~ "very High")
View(data)
```

Output:

capital.gain	capital.loss	hours.per.week
No Capital Gain	very High	Good
No Capital Gain	very High	Medium
No Capital Gain	very High	Good
No Capital Gain	very High	Good
No Capital Gain	very High	Good
No Capital Gain	very High	active
No Capital Gain	very High	Good
No Capital Gain	very High	Medium
No Capital Gain	very High	Good
No Capital Gain	very High	active
No Capital Gain	very High	Good

Description: In this code, a categorical variable "capital.gain". The case_when function is used to categorize individuals into different groups based on their capital gains. The categories include "No Capital Gain," "Low," "Moderate," "High," and "Very High." This transformation provides a more meaningful representation of the levels of capital gains for analysis and interpretation. The resulting modified dataframe can be viewed using the View(data) function.

Result for capital gain:

```
table(data$capital.gain)
```

```
  High            Low       Moderate No Capital Gain       very High
    94           1064           1395          29849            159
```

Description: This is the result of capital gain.

Maximum and Minimum for capital loss

Code:

```
min_closs <- min(data$capital.loss)
max_closs <- max(data$capital.loss)
cat("Minimum Capital loss:", min_closs, "\n")
cat("Maximum Capital loss:", max_closs, "\n")
```

Output:

```
> cat("Minimum Capital loss:", min_closs, "\n")
Minimum Capital loss: High
> cat("Maximum Capital loss:", max_closs, "\n")
Maximum Capital loss: very High
```

Description: In this code, the cat function is used to print the minimum and maximum values of a variable named "Capital loss." The minimum capital loss is displayed as "High," and the maximum capital loss is displayed as "very High." Typically, the output should represent numeric values rather than categories.

Converting numeric to categorical data for capital Loss

Code:

```
data$capital.loss <- case_when(
data$capital.loss <= 0 ~ "No Capital Loss",
data$capital.loss <= 1000 ~ "Low",
data$capital.loss <= 2000 ~ "Moderate",
data$capital.loss <= 3000 ~ "High",
data$capital.loss <= 4356 ~ "very High",)
View(data)
```

Output:

capital.gain	capital.loss
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High
No Capital Gain	very High

Description: In this code, a new categorical variable "capital.loss" is created in the dataframe based on the values in the existing "capital.loss" column. The case_when function is used to categorize individuals into different groups based on their capital losses. The categories include "No Capital Loss," "Low," "Moderate," "High," and "Very High." This transformation provides a

meaningful representation of the levels of capital losses for analysis and interpretation. The resulting modified dataframe can be viewed using the view () function.

Missing value

Code:

```
data[data == '?'] <- NA
data
```

Output:

	age	workclass	education	marital.status	occupation	relationship
1	90	<NA>	HS-grad	Widowed	<NA>	Not-in-family
2	82	Private	HS-grad	Widowed	Exec-managerial	Not-in-family
3	66	<NA>	Some-college	Widowed	<NA>	Unmarried
4	54	Private	7th-8th	Divorced	Machine-op-inspct	Unmarried
5	41	Private	Some-college	Separated	Prof-specialty	Own-child
6	34	Private	HS-grad	Divorced	Other-service	Unmarried
7	38	Private	10th	Separated	Adm-clerical	Unmarried
8	74	State-gov	Doctorate	Never-married	Prof-specialty	Other-relative
9	68	Federal-gov	HS-grad	Divorced	Prof-specialty	Not-in-family
10	41	Private	Some-college	Never-married	Craft-repair	Unmarried

Description: Missing values are replacing NA.

Finding missing values

Code:

```
colSums(is.na(data))
```

Output:

```
        age     workclass     education marital.status    occupation  relationship          race           sex  capital.gain
          0          1836             0              0          1843             0             0             0             0
capital.loss hours.per.week native.country        income
          0              0           583             0
```

Description: The colSums(is.na(data)) results indicate the number of missing values in each column of the data dataframe. Columns such as "workclass," "occupation," and "native.country" have notable missing data that may require attention or imputation.

Delete the rows with missing values

Code:

```
removedata<- na.omit(data)
removedata
mydata<-removedata
colSums(is.na(mydata))
View(mydata)
```

Output:

```
          age     workclass     education marital.status    occupation  relationship          race           sex  capital.gain
            0             0             0              0             0             0             0             0             0
capital.loss hours.per.week native.country        income
            0             0              0             0
```

age	workclass	education	marital.status	occupation	relationship	race	sex	capital.gain	capital.loss
82	Private	HS-grad	Widowed	Exec-managerial	Not-in-family	White	Female	No Capital Gain	very High
54	Private	7th-8th	Divorced	Machine-op-inspct	Unmarried	White	Female	No Capital Gain	very High
41	Private	Some-college	Separated	Prof-specialty	Own-child	White	Female	No Capital Gain	very High
34	Private	HS-grad	Divorced	Other-service	Unmarried	White	Female	No Capital Gain	very High
38	Private	10th	Separated	Adm-clerical	Unmarried	White	Male	No Capital Gain	very High
74	State-gov	Doctorate	Never-married	Prof-specialty	Other-relative	White	Female	No Capital Gain	very High
68	Federal-gov	HS-grad	Divorced	Prof-specialty	Not-in-family	White	Female	No Capital Gain	very High
45	Private	Doctorate	Divorced	Prof-specialty	Unmarried	Black	Female	No Capital Gain	very High
38	Self-emp-not-inc	Prof-school	Never-married	Prof-specialty	Not-in-family	White	Male	No Capital Gain	High
52	Private	Bachelors	Widowed	Other-service	Not-in-family	White	Female	No Capital Gain	High

Description: The colSums(is.na(mydata)) results for the modified mydata dataframe indicate that there are no missing values in any of the columns. This suggests that the operation "removedata" successfully handled or eliminated missing values from the dataset, resulting in a complete dataset across all columns.

Pearson's Chi-squared test

Code:

```
target <- as.factor(data$income)
features <- data[, -which(names(data) == 'income')]
chi_squared <- vector("numeric", length = ncol(features))
for (i in 1:ncol(features)) {
  contingency_table <- table(features[, i], target)
  chi_squared[i] <- chisq.test(contingency_table)$statistic
}
sorted_features <- names(features)[order(chi_squared, decreasing = TRUE)]
cat("Chi-Squared values for each feature:\n")
print(chi_squared)
cat("\nSorted features based on Chi-Squared values:\n")
print(sorted_features)
```

Output:

```
Chi-Squared values for each feature:
> print(chi_squared)
 [1] 1770.5795  804.1575 4070.3816 6061.7480 3687.6207 6233.8405  304.2414 1415.2864 3829.6172  629.2364 1964.6588  317.7367

Sorted features based on Chi-Squared values:
> print(sorted_features)
 [1] "relationship"   "marital.status" "education"      "capital.gain"   "occupation"     "hours.per.week" "age"            "sex"            "workclass"
[10] "capital.loss"   "native.country" "race"
```

Description: The R code conducts a chi-squared test for each feature in relation to the 'income' target variable, yielding chi-squared values. Subsequently, the features are sorted based on these

values in descending order. And 'relationship', 'marital.status', and 'education' exhibit the highest chi-squared values, indicating their stronger association with income levels in the dataset.

Dataset Describe

Code:

```
total_rows <- nrow(mydata)
total_columns <- ncol(mydata)
cat("Total Rows:", total_rows, "\n")
cat("Total Columns:", total_columns, "\n")
```

Output:

```
> cat("Total Rows:", total_rows, "\n")
Total Rows: 30162
> cat("Total Columns:", total_columns, "\n")
Total Columns: 13
```

Description: The provided code prints the total number of rows and columns in the dataset. In this instance, there are 30,162 rows and 13 columns.

```
glimpse(mydata)
table(mydata$income)
```

```
Rows: 30,162
Columns: 13
$ age            <chr> "Senior", "Adult", "Adult", "Adult", "Adult", "Senior", "Senior", "Adult", "Adult",
$ workclass      <chr> "Private", "Private", "Private", "Private", "Private", "State-gov", "Federal-gov", "
$ education      <chr> "HS-grad", "7th-8th", "Some-college", "HS-grad", "10th", "Doctorate", "HS-grad", "Do
$ marital.status <chr> "Widowed", "Divorced", "Separated", "Divorced", "Separated", "Never-married", "Divor
$ occupation     <chr> "Exec-managerial", "Machine-op-inspct", "Prof-specialty", "Other-service", "Adm-cler
$ relationship   <chr> "Not-in-family", "Unmarried", "Own-child", "Unmarried", "Unmarried", "Other-relative
$ race           <chr> "White", "White", "White", "White", "White", "White", "White", "Black", "White", "Wh
$ sex            <chr> "Female", "Female", "Female", "Female", "Male", "Female", "Female", "Female", "Male"
$ capital.gain   <chr> "No Capital Gain", "No Capital Gain", "No Capital Gain", "No Capital Gain", "No Capi
$ capital.loss   <chr> "very High", "very High", "very High", "very High", "very High", "very High", "very
$ hours.per.week <chr> "Medium", "Good", "Good", "active", "Good", "Medium", "Good", "Good", "active", "Med
$ native.country <chr> "United-States", "United-States", "United-States", "United-States", "United-States",
$ income         <chr> "<=50K", "<=50K", "<=50K", "<=50K", "<=50K", ">50K", "<=50K", ">50K", ">50K", ">50K"
> table(mydata$income)

<=50K  >50K
22654  7508
```

Description: The glimpse(mydata) function provides a concise summary of the structure and content of the 'mydata' showing all the value in now categorical. The table(mydata$income) function creates a frequency table, showing the distribution of values in the 'income' column of the 'mydata'.

Balancing Dataset:

Code:

```
over=ovun.sample(income~.,data = mydata,method="over")
over=over$data
table(over$income)
```

Output:

```
<=50K  >50K
22654 22556
```

Description: The dataset is imbalanced. For balancing the dataset, the ovun.sample function from the ROSE package in R to perform oversampling on the 'income' variable based on the other variables in the mydata. The oversampled data is stored in the 'over' variable. The subsequent table(over$income) command provides a frequency table of the 'income' variable in the oversampled dataset, showing the distribution of values after the oversampling process. Here 22654 people have income <=50k and 7508 people have income >50k so the dataset is imbalanced. For balance, those whose income is >50 are added with duplicate data This type of oversampling is commonly used to address imbalances in binary classification problems.

K Fold Cross Validation

Code:

```
over$income <- as.factor(over$income)
over$workclass <- as.factor(over$workclass)
over$education <- as.factor(over$education)
over$marital.status <- as.factor(over$marital.status)
over$occupation <- as.factor(over$occupation)
over$relationship <- as.factor(over$relationship)
over$race <- as.factor(over$race)
over$sex <- as.factor(over$sex)
over$capital.gain <- as.factor(over$capital.gain)
over$capital.loss <- as.factor(over$capital.loss)
over$hours.per.week <- as.factor(over$hours.per.week)
over$native.country <- as.factor(over$native.country)

set.seed(123)
train_control <- trainControl(method = "cv",
                              number = 10)
model <- train(income~., data = over,
               trControl = train_control,
               method = "naive_bayes")

print(model)
```

Output:

```
Naive Bayes

45210 samples
   12 predictor
    2 classes: '<=50K', '>50K'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 40689, 40688, 40689, 40689, 40690, 40690, ...
Resampling results across tuning parameters:

  usekernel  Accuracy   Kappa
  FALSE      0.7205263  0.4415657
   TRUE      0.6572446  0.3155051

Tuning parameter 'laplace' was held constant at a value of 0
Tuning parameter 'adjust' was held constant at a value of 1
Accuracy was used to select the optimal model using the largest value.
The final values used for the model were laplace = 0, usekernel = FALSE and adjust = 1.
```

Description: The code snippet conducts a naive Bayes classification model on the oversampled 'over' dataset, aiming to predict the 'income' variable based on various features. The model is assessed using 10-fold cross-validation, and two variations are explored: one with a kernel density estimate disabled (usekernel = FALSE) and one with it enabled (usekernel = TRUE). The resulting accuracy and kappa statistics are compared, with the model without the kernel generally achieving higher accuracy. The final selected model parameters are reported, showing that laplace smoothing is disabled (laplace = 0), and no adjustments are made (adjust = 1). The naive Bayes model is trained on a dataset with 45,210 samples and 12 predictors, categorizing individuals into two income classes: '<=50K' and '>50K'.

Splitting data into train and test data:

Code:

```r
split <- sample.split(over, SplitRatio = 0.7)
train_cl <- subset(over, split == "TRUE")
test_cl <- subset(over, split == "FALSE")
set.seed(120)
classifier_cl <- naiveBayes(income ~ ., data = train_cl)
classifier_cl
y_pred <- predict(classifier_cl, newdata = test_cl)
cm <- table(test_cl$income, y_pred)
cm
confusionMatrix(cm)
recall <- cm[2, 2] / sum(cm[2, ])
precision <- cm[2, 2] / sum(cm[, 2])
f_measure <- 2 * (precision * recall) / (precision + recall)
cat("Recall:", recall, "\n")
cat("Precision:", precision, "\n")
cat("F-measure:", f_measure, "\n")
```

Output:

```
Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
    <=50K      >50K
0.5010691 0.4989309

Conditional probabilities:
       age
Y       Adolescence       Adult      Senior Young Adult
  <=50K 0.013758093 0.755812242 0.032739847 0.197689818
  >50K  0.000000000 0.965346535 0.027190779 0.007462687
```

```
> confusionMatrix(cm)
Confusion Matrix and Statistics

       y_pred
        <=50K >50K
 <=50K  5173 1796
 >50K    926 6069

               Accuracy : 0.8051
                 95% CI : (0.7984, 0.8116)
    No Information Rate : 0.5632
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.61

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.8482
            Specificity : 0.7716
         Pos Pred Value : 0.7423
         Neg Pred Value : 0.8676
             Prevalence : 0.4368
         Detection Rate : 0.3705
   Detection Prevalence : 0.4991
      Balanced Accuracy : 0.8099

       'Positive' Class : <=50K
```

```
> cat("Recall:", recall, "\n")
Recall: 0.8676197
> cat("Precision:", precision, "\n")
Precision: 0.7716465
> cat("F-measure:", f_measure, "\n")
F-measure: 0.8168237
```

Description: The dataset is split into training and testing sets using the sample.split function. A Naive Bayes classifier is then trained on the training set (train_cl). The model is applied to the test set (test_cl), and a confusion matrix (cm) is generated to evaluate the classification performance. The code calculates metrics such as recall, precision, and the F1 score (f_measure) to assess the model's effectiveness in predicting the '>50K' income class. The final printed output includes the recall, precision, and F1 measure values, providing insights into the classifier's performance on the test set.