Data Wrangling (Data Preprocessing)

Practical assessment 1

Snigdha Mathur (s4017572)

Setup

```r
# Load the necessary packages required to reproduce the report. For example:

library(kableExtra)
library(magrittr)
```

Student names, numbers and percentage of contributions

Table 1: Group information

Student name	Student number	Percentage of contribution
Snigdha Mathur	S4017572	50
Astha Bathla	S3999096	50

Read/Import Data

```
accidents <- read.csv("ACCIDENT.csv")
head(accidents)
```

```
##    ACCIDENT_NO ACCIDENT_DATE ACCIDENT_TIME ACCIDENT_TYPE
## 1 T20120000012    2012-01-01      02:00:00             1
## 2 T20120000043    2012-01-01      00:45:00             1
## 3 T20120000044    2012-01-01      16:25:00             1
## 4 T20120000046    2012-01-01      16:25:00             2
## 5 T20120000060    2012-01-01      19:40:00             6
## 6 T20120027702    2012-01-01      13:20:00             1
##                 ACCIDENT_TYPE_DESC DAY_OF_WEEK DAY_WEEK_DESC DCA_CODE
## 1            Collision with vehicle           1        Sunday      110
## 2            Collision with vehicle           1        Sunday      116
## 3            Collision with vehicle           1        Sunday      120
## 4                 Struck Pedestrian           1        Sunday      102
## 5 Vehicle overturned (no collision)           1        Sunday      184
## 6            Collision with vehicle           1        Sunday      131
##                                   DCA_DESC LIGHT_CONDITION NODE_ID
## 1          CROSS TRAFFIC(INTERSECTIONS ONLY)               3   41780
## 2             LEFT NEAR (INTERSECTIONS ONLY)               5   43910
## 3                   HEAD ON (NOT OVERTAKING)               1   42677
## 4 FAR SIDE. PED HIT BY VEHICLE FROM THE LEFT               1   47545
## 5    OUT OF CONTROL ON CARRIAGEWAY (ON BEND)               1  248602
## 6                                  LEFT REAR               1   39704
##   NO_OF_VEHICLES NO_PERSONS_KILLED NO_PERSONS_INJ_2 NO_PERSONS_INJ_3
## 1              2                 0                1                0
## 2              2                 0                2                0
## 3              2                 0                1                0
## 4              1                 0                0                1
## 5              1                 0                1                0
## 6              2                 0                0                1
##   NO_PERSONS_NOT_INJ NO_PERSONS POLICE_ATTEND ROAD_GEOMETRY ROAD_GEOMETRY_DESC
## 1                  2          3             1             1 Cross intersection
## 2                  1          3             1             2     T intersection
## 3                  1          2             1             2     T intersection
## 4                  1          2             1             2     T intersection
## 5                  0          1             1             2     T intersection
## 6                  2          3             2             1 Cross intersection
##   SEVERITY SPEED_ZONE
## 1        2         80
## 2        2         80
## 3        2         60
## 4        3         60
## 5        2         60
## 6        3         60
```

- The ACCIDENTS.csv file is imported using the read.csv function. The function read.csv is a basic r function and hence does not require any particular library to be installed.
- The csv file can also be imported using read_csv function, which requires the readr library to be installed in the Rstudio.
- The variable 'accidents' stores the CSV file

- The first 6 values of the accident.csv are displayed using head function.

Data Description

- The dataset has been taken from the official website of Victorian Government, Australia.
- URL https://discover.data.vic.gov.au/dataset
- The data "ACCIDENT.csv" has been sourced from Victoria Police records combined with injury information from hospitals. This endorsed data provides particulars of road accidents that transpired in the state of Victoria.
- The datset provides details regarding the number of accidents befallen over a period of 11 years, starting from January 2012 till August 2023. It comprehends information like severity of an accident , place of an accident, count of injured people, type of crash and other essential factors.
- The dataset contains 22 columns describing the accidents.

Variable Description

Using Victorian Government's Victoria Road Crash metadata, variable description is as follows:

1. ACCIDENT_NO : The accident number is the primary key of the accident record and indicates an accident's ID.
2. ACCIDENTDATE : Stores date of an accident in dd/mm/yyyy format.
3. ACCIDENTTIME : Records time of an accident in hh:mm:ss, a 24 hour format.
4. ACCIDENT_TYPE : This field indicates the type of accident on the basis of "what happened" and groups the accidents into 9 categories.
5. ACCIDENT_TYPE_DESCRIPTION : This field is relative to the ACCIDENT_TYPE field mentioned above while offering a detailed description of what eventuated during the accident. These descriptions are segregated into 9 following categories: 1 Collision with vehicle 2 Struck pedestrian 3 Struck animal 4 Collision with a fixed object 5 Collision with some other object 6 Vehicle overturned (no collision) 7 Fall from or in moving vehicle 8 No collision and no object struck 9 Other accident
6. DAY_OF_WEEK: Number depicting the day of the week when the accident occured. Range 1- 7.
7. DAY_OF_WEEK_DESCRIPTION : States the day of week (Monday - Sunday) on which the accident took place..
8. DCA_CODE: DCA stands for "Definitions for classifying accidents". This attribute has the code for accident categorization. DCA_CODE is not null attribute.
9. DCA_CODE_DESCRIPTION: This variable provides the classification of the accident and is complimentary to the DCA_CODE. There are around 650 plus accident descriptions available, each bestowing detailed accounts of the occurrences of an accident.
10. LIGHT_CONDITION: Refers to the level of brightness at the time of accident. The levels are identified between a range of 1 to 9. 1 Day 2 Dusk/dawn 3 Dark street lights on 4 Dark street lights off 5 Dark no street lights 6 Dark street lights 7 Dark street lights partial working 8 No street lights 9 Unknown
11. NODE_ID: This is an identifier of the location where the accident occurred, representing an intersection or junction point on a road network.
12. NO_OF_VEHICLES : Count of vehciles incorportated in an accident.
13. NO_PERSONS_KILLED : Count of people killed in an accident.
14. NO_PERSONS_INJ_2 : Count of people who faced a serious injury in an accident.
15. NO_PERSONS_INJ_3: Count of people who met any other form of injury.

16. NO_PERSONS_NOT_INJ: Count of people with no injuries.
17. NO_PERSONS : Total count of people involved in the accident.
18. POLICE_ATTEND: This attribute depicts whether police visited the place of accident, where 1, 2 and 9 signifies yes, no and unknown values respectively.
19. ROAD_GEOMETRY: The code for road layout (based of official records) whereever an accident transpired.
20. ROAD_GEOMETRY_DESCRIPTION : Description of the road geometry where an accident happened.
21. SEVERITY: Indicates the level of seriousness of an accident.
22. SPEED_ZONE : The speed zone of the area where an accident occured.

Inspect and Understand

```
dim(accidents)
```

```
## [1] 165892     22
```

- The dim function stands for dimension. It retrieves the total number of rows and columns in a dataset.
- The output shows that the accident dataset has 1,65,892 rows and 22 columns

```
colnames(accidents)
```

```
##  [1] "ACCIDENT_NO"        "ACCIDENT_DATE"      "ACCIDENT_TIME"
##  [4] "ACCIDENT_TYPE"      "ACCIDENT_TYPE_DESC" "DAY_OF_WEEK"
##  [7] "DAY_WEEK_DESC"      "DCA_CODE"           "DCA_DESC"
## [10] "LIGHT_CONDITION"    "NODE_ID"            "NO_OF_VEHICLES"
## [13] "NO_PERSONS_KILLED"  "NO_PERSONS_INJ_2"   "NO_PERSONS_INJ_3"
## [16] "NO_PERSONS_NOT_INJ" "NO_PERSONS"         "POLICE_ATTEND"
## [19] "ROAD_GEOMETRY"      "ROAD_GEOMETRY_DESC" "SEVERITY"
## [22] "SPEED_ZONE"
```

- The colnames() functions prints the names of all columns in a dataset.
- Based on the results obtained, we can say that accident dataset has 22 columns describing the details of a particular accident that occured on Victoria streets

```
str(accidents)
```

```
## 'data.frame':    165892 obs. of  22 variables:
##  $ ACCIDENT_NO       : chr  "T20120000012" "T20120000043" "T20120000044" "T20120000046" ...
##  $ ACCIDENT_DATE     : chr  "2012-01-01" "2012-01-01" "2012-01-01" "2012-01-01" ...
##  $ ACCIDENT_TIME     : chr  "02:00:00" "00:45:00" "16:25:00" "16:25:00" ...
##  $ ACCIDENT_TYPE     : int  1 1 1 2 6 1 4 1 1 4 ...
##  $ ACCIDENT_TYPE_DESC: chr  "Collision with vehicle" "Collision with vehicle" "Collision with vehicl
##  $ DAY_OF_WEEK       : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ DAY_WEEK_DESC     : chr  "Sunday" "Sunday" "Sunday" "Sunday" ...
##  $ DCA_CODE          : int  110 116 120 102 184 131 183 130 113 181 ...
##  $ DCA_DESC          : chr  "CROSS TRAFFIC(INTERSECTIONS ONLY)" "LEFT NEAR (INTERSECTIONS ONLY)" "HEA
##  $ LIGHT_CONDITION   : int  3 5 1 1 1 1 1 1 1 5 ...
##  $ NODE_ID           : int  41780 43910 42677 47545 248602 39704 249133 256115 39192 248603 ...
##  $ NO_OF_VEHICLES    : int  2 2 2 1 1 2 1 2 2 1 ...
##  $ NO_PERSONS_KILLED : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ NO_PERSONS_INJ_2  : int  1 2 1 0 1 0 2 1 1 0 ...
##  $ NO_PERSONS_INJ_3  : int  0 0 0 1 0 1 0 0 0 2 ...
##  $ NO_PERSONS_NOT_INJ: int  2 1 1 1 0 2 0 1 2 0 ...
##  $ NO_PERSONS        : int  3 3 2 2 1 3 2 2 3 2 ...
##  $ POLICE_ATTEND     : int  1 1 1 1 1 2 1 2 1 1 ...
##  $ ROAD_GEOMETRY     : int  1 2 2 2 2 1 2 5 2 5 ...
##  $ ROAD_GEOMETRY_DESC: chr  "Cross intersection" "T intersection" "T intersection" "T intersection"
##  $ SEVERITY          : int  2 2 2 3 2 3 2 2 2 3 ...
##  $ SPEED_ZONE        : int  80 80 60 60 60 60 100 70 60 100 ...
```

- The str() function provides the glimpse of the datset by displaying the data types of the columns along with values present in the data.
- The csv file contains
  - 6 Character variables : ACCIDENT_NO, ACCIDENT_TYPE_DESC, DAY_WEEK_DESC , DCA_DESC, ROAD_GEOMETRY_DESC and SPEED_ZONE
  - 1 Date format variable - ACCIDENT_DATE
  - 1 Time format variables : ACCIDENT_TIME
  - 14 Integer variables : ACCIDENT_TYPE , DAY_OF_WEEK, DCA_CODE, LIGHT_CONDITION, NODE_ID, NO_OF_VEHICLES, NO_PERSONS_KILLED, NO_PERSONS_INJ_2, NO_PERSONS_INJ_3, NO_PERSONS_NOT_INJ, NO_PERSONS, POLICE_ATTEND, ROAD_GEOMETRY, SEVERITY
- In this datset, ACCIDENT_TYPE, DAY_OF_WEEK, DCA_CODE, LIGHT_CONDITION, ROAD_GEOMETRY, POLICE_ATTEND and SEVERITY are identified as categorical variables.
- The columns ACCIDENT_TYPE, DAY_OF_WEEK, DCA_CODE and ROAD_GEOMETRY have a description available in ACCIDENT_TYPE_DESCRIPTION, DAY_OF_WEEK_DESCRIPTION, DCA_CODE_DESCRIPTION, ROAD_GEOMETRY_DESCRIPTION respectively.
- Hence we check and assign levels to columns - LIGHT_CONDITION, POLICE_ATTEND and SEVERITY

```r
levels(accidents$SEVERITY)
```

```
## NULL
```

- levels() checks the levels present in the categorical variable in the dataset.
- The output states that the variable SEVERITY has not been assigned levels in the dataset.

```r
accidents$SEVERITY <- factor( accidents$SEVERITY , levels = c(4,3,2,1) ,
                              labels = c("Non Injury", "Other Injury Accident",
                                         "Serious Injury accident","Fatal Accident"))
```

- It was identified that SEVERITY is a categorical variable
- The SEVERITY column is being converted into a factor variable with 4 levels and labels
  - Value 4 corresponds to the accident resulting in a Non Injury
  - Value 3 corresponds to the accident resulting in Other Injury Accident
  - Value 2 corresponds to the accident resulting in Serious Injury Accident
  - Value 1 corresponds to teh accident resulting in a Fatal Accident

```r
levels(accidents$SEVERITY)
```

```
## [1] "Non Injury"              "Other Injury Accident"
## [3] "Serious Injury accident" "Fatal Accident"
```

- Checking the levels again of Severity column, we see that now it has 4 distinct levels

```r
levels(accidents$LIGHT_CONDITION)
```

```
## NULL
```

- levels() checks the levels present in the categorical variable in the dataset.
- The output states that the variable LIGHT_CONDITION has not been assigned levels in the dataset.

```r
accidents$LIGHT_CONDITION <- factor( accidents$LIGHT_CONDITION , levels = c(1,2,3,4,5,6,7,8,9) ,
                               labels = c("Day", "Dusk/dawn", "Dark street lights on", "Dark street ligh
```

- It was identified that LIGHT_CONDITION is a categorical variable
- The LIGHT_CONDITION column is being converted into a factor variable with 9 levels and labels
  - Level of brightness 1 refers to daylight
  - Level of brightness 2 refers to Dusk/Dawn
  - Level of brightness 3 refers to Dark street lights on
  - Level of brightness 4 refers to Dark street lights off
  - Level of brightness 5 refers to ark no street lights
  - Level of brightness 6 refers to Dark street lights
  - Level of brightness 7 refers to Dark street lights partial working
  - Level of brightness 8 refers to Do street lights
  - Level of brightness 9 refers to Unknown

```r
levels(accidents$LIGHT_CONDITION)
```

```
## [1] "Day"                                "Dusk/dawn"
## [3] "Dark street lights on"              "Dark street lights off"
## [5] "Dark no street lights "             "Dark street lights "
## [7] "Dark street lights partial working" "No street lights"
## [9] "Unknown"
```

- Checking the levels again of Lights_Conditon column, we see that now it has 9 distinct levels

```r
levels(accidents$POLICE_ATTEND)
```

```
## NULL
```

- levels() checks the levels present in the categorical variable in the dataset.

- The output states that the variable POLICE_ATTEND has not been assigned levels in the dataset.

```r
accidents$POLICE_ATTEND <- factor( accidents$POLICE_ATTEND , levels = c(1,2,9) ,
                              labels = c("Attended", "Didn't Attend", "Unknown"))
```

- It was identified that POLICE_ATTEND is a categorical variable
- The POLICE_ATTEND column is being converted into a factor variable with 3 levels and labels
  - 1 refers to "Attended"

  - 2 refers to "Didn't Attend"

  - 9 refers to "Unknown"

```r
levels(accidents$POLICE_ATTEND)
```

```
## [1] "Attended"      "Didn't Attend" "Unknown"
```

- Checking the levels again of POLICE_ATTEND column, we see that now it has 3 distinct levels

Subsetting

```r
new_accidents <- accidents[1:10,]
dim(new_accidents)
```

```
## [1] 10 22
```

- A new variable 'new_accidents' is declared which stores the subset of the accident dataset
- accidents[1:10,] signifies that the new_ accidents variable will store the rows 1 to 10 with all columns

- Using dim function, we see that the new_accident data frame has 10 rows and 22 columns

```r
matt <- data.matrix(new_accidents)
head(matt)
```

```
##   ACCIDENT_NO ACCIDENT_DATE ACCIDENT_TIME ACCIDENT_TYPE ACCIDENT_TYPE_DESC
## 1           1             1             2             1                  2
## 2           2             1             1             1                  2
## 3           3             1             5             1                  2
## 4           4             1             5             2                  3
## 5           6             1             8             6                  4
## 6          10             1             3             1                  2
##   DAY_OF_WEEK DAY_WEEK_DESC DCA_CODE DCA_DESC LIGHT_CONDITION NODE_ID
## 1           1             1      110        1               3   41780
## 2           1             1      116        4               5   43910
## 3           1             1      120        3               1   42677
## 4           1             1      102        2               1   47545
## 5           1             1      184        8               1  248602
## 6           1             1      131        5               1   39704
##   NO_OF_VEHICLES NO_PERSONS_KILLED NO_PERSONS_INJ_2 NO_PERSONS_INJ_3
## 1              2                 0                1                0
## 2              2                 0                2                0
## 3              2                 0                1                0
## 4              1                 0                0                1
## 5              1                 0                1                0
## 6              2                 0                0                1
##   NO_PERSONS_NOT_INJ NO_PERSONS POLICE_ATTEND ROAD_GEOMETRY ROAD_GEOMETRY_DESC
## 1                  2          3             1             1                  1
## 2                  1          3             1             2                  3
## 3                  1          2             1             2                  3
## 4                  1          2             1             2                  3
## 5                  0          1             1             2                  3
## 6                  2          3             2             1                  1
##   SEVERITY SPEED_ZONE
## 1        3         80
## 2        3         80
## 3        3         60
## 4        2         60
## 5        3         60
## 6        2         60
```

- The data.matrix function is used to create a matrix in RStudio
- A new matrix, 'matt', is declared which stores the data of the new_accidents dataframe.
- head(matt) displayes the top 6 rows of the matrix matt

```r
str(matt)
```

```
##  int [1:10, 1:22] 1 2 3 4 6 10 5 9 7 8 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : chr [1:10] "1" "2" "3" "4" ...
##   ..$ : chr [1:22] "ACCIDENT_NO" "ACCIDENT_DATE" "ACCIDENT_TIME" "ACCIDENT_TYPE" ...
```

- Str() function provides a concise summary of the structure of an matrix, including its type, dimensions and the first few elements

- The output shows that the matrix matt is an integer matrix. This is because 14 variables out of 22 variable list consist of integer values.

- The character variables are converted into integer as RStudio automatically converts it to integer format to maintain a uniform data type across all columns of the matrix. Hence the structure of the matrix is integer besides presence of charecter variables in the dataset.

Create a new Data Frame

```
PurchaseID <- c(1:10)
PurchaseAmount <- c (">10k", "5k - 10k", "<5k" , "5k - 10k", "<5k", "<5k",">10k", ">10k", "5k - 10k", "

df <- data.frame (PurchaseID,PurchaseAmount)
head(df)
```

```
##   PurchaseID PurchaseAmount
## 1          1           >10k
## 2          2       5k - 10k
## 3          3            <5k
## 4          4       5k - 10k
## 5          5            <5k
## 6          6            <5k
```

- For creating a new dataframe with two variables (Numeric and Categorical) and 10 observations, we define 2 variables, PurchaseID and PurhaseAmount.
- The PurchaseID is a integer variables which stores the purchase/ transaction ID of the shopping
- The PurchaseAmount is a categorical variables storing the amount spent in purchasing. The variable has 3 levels - <5k , 5k - 10k , >10k
- Both PurchaseID and PurchaseAmount have 10 values stored in them
- data.frame() function creates a data frame 'df' which stores PurchaseID and PurchaseAmount
- head() prints the top 6 rows in the dataframe

```
df$PurchaseAmount <- factor(df$PurchaseAmount, levels = c("<5k", "5k - 10k", ">10k"), ordered = TRUE)
levels(df$PurchaseAmount)
```

```
## [1] "<5k"      "5k - 10k" ">10k"
```

- Since PurchaseAmount is a categorical variable, we factorise the variable and assign levels in a Ordered= TRUE manner.
- levels() prints the levels of PurchaseAmount in an ordered manner.

```
str(df)
```

```
## 'data.frame':    10 obs. of  2 variables:
##  $ PurchaseID    : int  1 2 3 4 5 6 7 8 9 10
##  $ PurchaseAmount: Ord.factor w/ 3 levels "<5k"<"5k - 10k"<..: 3 2 1 2 1 1 3 3 2 3
```

- The str() shows the structure of the dataframe.
- The dataframe df has 10 rows , 2 columns.
  - The variable PurchaseID is a integer variables with values 1,2,3,4,5,6,7,8,9,10
  - The variable PurchaseAmount is a ordered factor with 10 values

```
PurchaseQuantity <- seq(from = 100,  to = 190, by = 10)
PurchaseQuantity
```

```
##  [1] 100 110 120 130 140 150 160 170 180 190
```

- A new vector PurchaseQuantity is declared with numeric values. PurchaseQuantity stores the quantity of the products bought.

- For assigning values to PurchaseQuantity, we use seq() function.
- The seq() assigns values to the vector in a sequential manner.
- The parameters passed in seq(from = 100, to = 190, by = 10) are :
  - from = 100: Initializes the starting value of the sequence to 100.
  - to = 190: Initializes the ending value of the sequence to 190.
  - by = 10: Indicates the increment between consecutive numbers in the sequence, which is set to 10..
- By printing PurchaseQuatity, we can see that PurchaseQuantity vector stores 10 integer values starting from 100 with an increment of 10

```r
df2 <- cbind(df, PurchaseQuantity)
df2
```

```
##    PurchaseID PurchaseAmount PurchaseQuantity
## 1           1           >10k              100
## 2           2       5k - 10k              110
## 3           3            <5k              120
## 4           4       5k - 10k              130
## 5           5            <5k              140
## 6           6            <5k              150
## 7           7           >10k              160
## 8           8           >10k              170
## 9           9       5k - 10k              180
## 10         10           >10k              190
```

- cbind() function is used to combine 2 or more data frames by column binding. The code cbind(df, PurchaseQuantity) combines the df data frame to Purchasequantity in a column binding
- The result of the cbind() is stored in a new data frame 'df2'
- Upon printing df2, we can see that df2 now has 3 variables - PurchaseID, PurchaseAmount and PurchaseQuantity

References

Victorian Government(n.d.) *Victoria Road Crash Data*, Victorian Government website, accessed 19th March 2024. https://discover.data.vic.gov.au/dataset