Data Cleaning of FIFA-2021-Dataset Using R

Background

This challenge was organized by @ChinosoPromise and @VicSomadina for data newbies, intermediate, and pro data analysts to test their data cleaning knowledge or to learn how to clean dirty and messy data as the case may be. The uncleaned data was gotten from Kaggle

About the dataset

The dataset contains information on players that partook in FIFA 2021 football. There are 18979 rows and 77 columns in the datasets.

Data Cleaning Process

The notable cleaning performed on the dataset is described below

The CSV file was loaded into RStudio using the below code

```
#load the libraries
library(tidyverse)
library("scales")
#load the data
new_data <- read.csv("C:/Users/Raufr/OneDrive/Desktop/R/Data Cleaning with R/fifa21 raw data v2.csv")
view(new_data)
```

Club Column

This column contains white and trailing spaces so I used the gsub() function to remove white space

```
# Removing white spaces from club column
new_data$Club<- gsub("\\s+", "", new_data$Club)
```

Contract column

Contains information about two separate years. The column was used to create a "contract type" column to show if a player is on contract, on loan, or Free.

```
#Create a Contract_type to indicate if players are on contract, on loan or free
new_data <- new_data %>%
  mutate(Contract_type = case_when(
    grepl("~",Contract) ~ "Contract",
    grepl("L", Contract) ~ "On Loan",
    grepl("F",Contract) ~ "Free") )
```

Output:

Contract_type
Contract
Contract
Contract
Contract
Contract

The contract column was also split into "Contract Start", and "Contract_ End" columns to capture the start and existing date of players that are on contract.

```
#split contract column into Contract_Start and Contract_End Columns
new_data <- separate(new_data,Contract, into = c('Contract_Start','Contract_End'),sep = "~")
view(new_data)
```

Output:

Contract_Start	Contract_End
2004	2021
2018	2022
2014	2023
2015	2023
2017	2022
2014	2023

Height column

This column contains height measured in cm, feet, and inches and we need to adopt just a unit of measurement which is centimeters, code was written to convert values that are in inches to cm, and values in feet were converted to cm as well. Cleaning this column was a bit tough because some rows contain values both in inches and feet which need to be converted separately before adding them together. Values that are in inches were multiplied by 30.48, while values in feet were multiplied by 2.54 and both numbers were added together.

Below is the snippet of the code written to achieve this.

```
#cleaning Height column
v <<- 1
for (val in new_data$Height) {
  if (grepl("cm",val)) {
    val <- gsub("cm","", val)
    new_data$Height[v] = val
  }
  else if (grepl("'",val)){
    val <- gsub("'","", val)
    val <- gsub('"',"",val)

    if (as.numeric(val) < 100){
      wholenumber <- floor(as.numeric(val)/10)
      decimal <- as.numeric(val) - floor(as.numeric(val)/10) * 10
      val <- wholenumber * 30.48 + (decimal) * 2.54
    }
    else
    {
      wholenumber <- floor(as.numeric(val)/100)
      decimal <- as.numeric(val) - floor(as.numeric(val)/100) * 100
      val <- wholenumber * 30.48 + (decimal) * 2.54
    }
    new_data$Height[v] = val
  }
  else{}
  v <- v+1
}
new_data$Height <- as.numeric(new_data$Height)  #convert height to number
```

The data type was also changed to numeric

Output:

```
> print(new_data$Height)
   [1] 170.00 187.00 188.00 181.00 175.00 184.00 175.00 191.00 178.00 187.00 193.00 175.00 185.00 199.00 193.00
  [16] 185.00 184.00 173.00 170.00 168.00 176.00 177.00 188.00 188.00 193.00 187.00 175.00 183.00 176.00 180.00
  [31] 180.00 173.00 178.00 189.00 179.00 188.00 183.00 181.00 185.00 187.00 189.00 187.00 195.00 180.00 172.00
  [46] 182.00 188.00 185.00 186.00 192.00 173.00 191.00 165.00 191.00 179.00 194.00 191.00 183.00 173.00 167.00
  [61] 170.00 182.00 191.00 191.00 176.00 188.00 189.00 188.00 186.00 196.00 175.00 184.00 181.00 186.00 183.00
  [76] 179.00 175.00 180.00 182.00 181.00 180.00 163.00 186.00 183.00 176.00 190.00 191.00 180.00 174.00 183.00
  [91] 181.00 191.00 190.00 169.00 183.00 187.00 175.00 178.00 180.00 183.00 185.00 190.00 185.00 181.00 174.00
 [106] 194.00 181.00 179.00 171.00 171.00 195.00 170.00 184.00 190.00 189.00 184.00 185.00 170.00 172.00 188.00
```

Weight Column

The column is similar to the height but less technical because the values in this column contain just two units, kg, and lbs. The values in lbs were converted to kg by multiplying them with 0.454 and unwanted characters were removed. The data type was also changed to numeric

```
#cleaning weight column
v <<- 1
for (val in new_data$weight) {

  if (grepl("kg",val)){
    val <- gsub("kg","", val)
    new_data$weight[v] = val
  }
  else if (grepl("lbs",val)){
    val <- gsub("lbs","", val)
    new_data$weight[v] = as.numeric(val) * 0.454
  }
  else{}
  v <- v+1
}
new_data$weight <- as.numeric(new_data$weight)  #convert weight column to numeric
```

Output:

```
print(new_data$weight)
  [1]  72.000  83.000  87.000  70.000  68.000  80.000  71.000  91.000  73.000  85.000  92.000  69.000  84.000
 [14]  96.000  92.000  81.000  82.000  70.000  69.000  70.000  73.000  75.000  86.000  89.000  92.000  89.000
 [27]  74.000  76.000  73.000  76.000  69.000  64.000  64.000  85.000  69.000  78.000  78.000  76.000  80.000
 [40]  85.000  76.000  80.000  90.000  69.000  66.000  83.000  82.000  85.000  75.000  82.000  73.000  81.000
 [53]  60.000  84.000  74.000  85.000  94.000  79.000  67.000  68.000  68.000  80.000  82.000  76.000  78.000
 [66]  83.000  89.000  80.000  70.000  90.000  75.000  75.000  65.000  81.000  75.000  67.000  70.000  76.000
```

The Value, Wage, and Released Clause columns

The columns contain special characters (€, K, M)

Value	Wage	Release.Clause
€103.5M	€560K	€138.4M
€63M	€220K	€75.9M
€120M	€125K	€159.4M
€129M	€370K	€161M
€132M	€270K	€166.5M
€111M	€240K	€132M

Approach**:** The special characters were removed and multiplied by the values with K by “1000” and M by “1,000,000”. The values were converted to dollars by multiplying them by 1.06 (using the £ to $ rate). Datatypes were changed to numeric

```
#clean the Value Column
new_data$value <- gsub("€","", new_data$value)
v <<- 1
for (val in new_data$value) {

  if (grepl("M",val)){
    val <- gsub("M","", val)
    new_data$value[v] = as.numeric(val) * 1000000
  }
  else if (grepl("K",val)){
    val <- gsub("K","", val)
    new_data$value[v] = as.numeric(val) * 1000
  }
  else{}
  v <- v+1
}
# convert from euros to dollars
new_data$value <-as.numeric(new_data$value)
new_data$value <- new_data$value * 1.06       #official rate as at March 16 2023
```

The code above shows the cleaning of the Value column and it was the same approach used for cleaning the Wage and Released Clause columns.

Below is the snippet of the three columns after cleaning

Value	Wage	Release.Clause
109710000	593600	146704000
66780000	233200	80454000
127200000	132500	168964000
136740000	392200	170660000
139920000	286200	176490000
117660000	254400	139920000

WF, SM, and IR columns

These columns contain the ratings of the player's weak foot, Skill Moves, and Injury ratings on a scale of 1 to 5. The columns consist of special characters (★).

W.F	SM	A.W	D.W	IR
4 ★	4★	Medium	Low	5 ★
4 ★	5★	High	Low	5 ★
3 ★	1★	Medium	Medium	3 ★
5 ★	4★	High	High	4 ★
5 ★	5★	High	Medium	5 ★
4 ★	4★	High	Medium	4 ★
3 ★	4★	High	Medium	3 ★
3 ★	1★	Medium	Medium	3 ★
4 ★	5★	High	Low	3 ★

Approach: The ★ was removed and the data type of each column was changed to an integer.

```
#remove special characters from W.F.SM, and IR columns
new_data$W.F <- gsub("★","",new_data$W.F)
new_data$W.F <- gsub(" ","",new_data$W.F)
new_data$SM <- gsub("★","",new_data$SM)
new_data$SM <- gsub(" ","",new_data$SM)
new_data$IR <- gsub("★","",new_data$IR)
new_data$IR <- gsub(" ","",new_data$IR)
#Convert W.F, SM,and IR columns to integer
new_data$W.F <-as.integer(new_data$W.F)
new_data$SM <-as.integer(new_data$SM)
new_data$IR <-as.integer(new_data$IR)
```

Output:

W.F	SM	A.W	D.W	IR
4	4	Medium	Low	5
4	5	High	Low	5
3	1	Medium	Medium	3
5	4	High	High	4
5	5	High	Medium	5
4	4	High	Medium	4
3	4	High	Medium	3
3	1	Medium	Medium	3

Hits Column

There are null values and a special character K in this column.

The null values were removed while values that contain K were multiplied by “1000” and the data type was converted to numeric.

```
#cleaning Hits column
v <<- 1
for (val in new_data$Hits) {

  if (grepl("K",val)){
    val <- gsub("K","", val)
    new_data$Hits[v] = val = as.numeric(val) * 1000
  }
  else{}
  v <- v+1
}
new_data$Hits <-as.numeric(new_data$Hits)   #convert Hit Column to number
unique(new_data$Hits)
new_data<- new_data %>% #replacing null values in Hits column with 0
  mutate(Hits=if_else(is.na(Hits), 0, Hits))
```

BOV, X.OVA, and POT columns

The values of these columns are integers but they need to be represented as percentages.

Approach: The column was divided by 100 and the “Percent” function was used to format the values as percentages.

```
#Divide BOV, OVA, and POT columns by 100 to change the columns to numeric
#then convert to percentage(%)
new_data$POT<- new_data$POT/100
new_data$X.OVA<- new_data$X.OVA/100
new_data$BOV<- new_data$BOV/100
#convert them to percentage using the "percent" function
new_data$POT<- percent(new_data$POT, accuracy=1)
new_data$X.OVA<- percent(new_data$X.OVA, accuracy=1)
new_data$BOV<- percent(new_data$BOV, accuracy=1)
```

Output:

X.OVA	POT
93%	93%
92%	92%
91%	93%
91%	91%
91%	91%
91%	91%

Inconsistent Column names

There are inconsistencies in some column names such as LongName, X.OVA, and others. These columns were renamed to follow the naming convention.

```
#rename columns
new_data<- new_data %>%
  rename(Full_Name=Long_Name, Overall_Rating=X.OVA, Potential=POT, Height_cm= Height, Weight_kg=Weight,
         Best_Overall=BOV,Injury_Rating= IR, Pass_Accuracy=PAS, Shooting_Attribute=SHO,Pace=PAC)
```

Finally, the cleaned dataset was saved as a CSV file and exported.

Conclusion

Indeed, it was a real challenge that requires critical thinking and a lot of research, especially with the use of R. But in all, the knowledge and experience gained during this challenge cannot be quantified.

I am happy for participating in this challenge and I am looking forward to taking part in the Data Visualization soon.