Data Wrangling (Data Preprocessing)

Practical assessment 2

Pragati Patidar and s3858702

Submission Steps:

Required packages

(List of packages which i have installed : readr, xlsx, readxl, foreign, gdata, rvest, dplyr, tidyr, deductive, editrules, validate, Hmisc, forecast, stringr, outliers, knitr and base R)

```
# This is the R chunk for the required packages
library(readr)
library(dplyr)
library(foreign)
library(tidyr)
library(outliers)
library(knitr)
library(rvest)
library(Hmisc)
library(editrules)
library(MVN)
library(stringr)
library(rvest)
library(infotheo)
library(base)
```

Executive Summary

The main purpose of this assignment is reviewing data preprocessing tools and getting used to them. T

- From the https://www.kaggle.com/ (https://www.kaggle.com/) website, i have searched for different data set, there are some requirements like, data set should have different data types, one of data sets should be untidy, should have missing values and can apply mutate function, and transformation, so by keeping in my this conditions.
- First i have searched on https://www.kaggle.com/ (https://www.kaggle.com/), after searching a lot, i got dataset a titanic data according to requirement.It had numeric(double) and Character data type, in character columns, some columns can be converted as factor, so converted those columns into factor. i read whole information about data set and searched on different websites for similar data. But i could not get it.
- I search again on https://www.kaggle.com/ (https://www.kaggle.com/) got two datasets, but i require one untidy dataset, it was hard to find, after searching i got one, i cleaned that dataset using tidyr functions before merging it.

- Within the scope of demonstration the understanding of the preprocessong tools.Firsty , i merged three datasets using right_join(via common key) to form the main data frame named 'Titanic' for this assignment. Then, its structure, attribute are checked and inappropriate data types is converted to the right ones. As the dataset is untidy, applied tidy principles(using pivot_longer function) to make it tidy. Thirdly, it comes the next step which is mutating a new column from the existing variables. Fourthly, the data frame is scanned for any missing values and an relevant method is applied to replace them. Last but not least, all the numeric variables are scanned for any outliers and an appropriate method is used to takle the issue. Finally, transformation is applied to one of the variables to obtain a symmetric distribution.

Data

- Dataset Information: website: "www.kagggle.com"
- About titanic Dataset:There were 2,224 passengers and crew on-bound the ill-fated Titanic when she collided with an iceberg in the North Atlantic, April 14, 1912 and sunk the next morning at 2:20 AM, April 15. Of that total, 1,316 were passengers and 908 were crew. That passenger number is far shy of the total Titanic passenger capacity of 2,435 people.
- In this datasets i have the inforamtion of 891 passengers out of 1316.

  link(URL) of datasets(datasets can be downloaded using these links):

- Hosting website of all three datasets-><https://www.kaggle.com/> (https://www.kaggle.com/)

1. trainNationalitySubset.csv(dataset1)::"https://www.kaggle.com/warrenelder/titanic-passenger- nationalities?select=trainNationalitySubse%22t.csv (https://www.kaggle.com/warrenelder/titanic-passenger-nationalities?select=trainNationalitySubse%22t.csv)

Description of dataset 1:

- i downloaded dataset 1–> **"trainNationalitySubset.csv"** file in my computer saved it into folder, made it is a working directory and imported data through "readr" package.
- This dataset consist of 2 variables:
- *PassengerID-(numeric), (1 to 891 passengers observations).*
- And NAtionality(Character) represent nationality of passengers of titanic.(*1 to 891 observations*)

2.**titanic_train.csv**(dataset2)::"https://www.kaggle.com/tedllh/titanic-train (https://www.kaggle.com/tedllh/titanic-train)" (In both the datasets common key is "PassengerID", so i used left join for merging it with datset 1)

Description of dataset 2:

- The dataset has (1 to 891 observations and 12 variables)
- Variables description:
- PassengerID= unique id of a passenger(1 to 891)
- pclass: A proxy for socio-economic status (SES) 1st = Upper 2nd = Middle 3rd = Lower.
- age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5
- sibsp: The dataset defines family relations in this way… Sibling = brother, sister, stepbrother,
- stepsister Spouse = husband, wife (mistresses and fiancés were ignored)

- parch: The dataset defines family relations in this way… Parent = mother, father Child = daughter, son, stepdaughter, stepson Some children travelled only with a nanny, therefore parch=0 for them.
- ticket - Ticket Number,
- Fare - Passenger Fare
- Cabin - Cabin allotted to passengers(note: In titanic only 371 cabins in total)
- Embarked - Port of Embarkation (C,Q,S) are Cherbourg; = Queenstown, Southampton)

3.**train_data.csv**(dataset3)::https://www.kaggle.com/azeembootwala/titanic?select=train_data.csv (https://www.kaggle.com/azeembootwala/titanic?select=train_data.csv) Description of datasets:

***Description of dataset 3**:

(This dataset has 792 observations and 17 variables)

- Variables description:
- x1: index, i removed it during subsetting, because it is not required.
- PassengerID= unique id of a passenger
- pclass: A proxy for socio-economic status (SES) 1st = Upper 2nd = Middle 3rd = Lower.
- age: Age is fractional if less than 1. If the age is estimated, is it in the form of xx.5
- Title1, title2, title3, title4, are unknowns showing 0 as a value, as it is untidy, i subsetted data frame and made it meaningful with all known variables.
- Fare: fare of a passenger given in fractions.
- Embarked - Port of Embarkation (emb1 = 0/1; emb2= 0/1; emb3= 0/1).(emb1,emb2,emb3 are Cherbourg; = Queenstown, Southampton)
- Downloaded the csv file, saved in working directory.
- With the help of ‘readr’ Package. i imported all three .csv files.3.

```
##Loading other datasets:
#Dataset 1:(having 891 obseravtions and 2 variables)
data1<-read_csv("trainNationalitySubset.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   PassengerId = col_double(),
##   Nationality = col_character()
## )
```

```
#Printing first 6 rows:
print.data.frame(head(data1))
```

```
##   PassengerId                Nationality
## 1           1              CelticEnglish
## 2           2              CelticEnglish
## 3           3 Nordic,Scandinavian,Sweden
## 4           4              CelticEnglish
## 5           5              CelticEnglish
## 6           6              CelticEnglish
```

```
#dataset2: having 891 observations and 12 variables)
data2<- read_csv("titanic_train.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   PassengerId = col_double(),
##   Survived = col_double(),
##   Pclass = col_double(),
##   Name = col_character(),
##   Sex = col_character(),
##   Age = col_double(),
##   SibSp = col_double(),
##   Parch = col_double(),
##   Ticket = col_character(),
##   Fare = col_double(),
##   Cabin = col_character(),
##   Embarked = col_character()
## )
```

```
#Printing first 6 rows:
print.data.frame(head(data2))
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
##                                                 Name    Sex Age SibSp Parch
## 1                             Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                              Heikkinen, Miss. Laina female  26     0     0
## 4        Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                            Allen, Mr. William Henry   male  35     0     0
## 6                                   Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500  <NA>        S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250  <NA>        S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500  <NA>        S
## 6           330877  8.4583  <NA>        Q
```

```
#dataset2: having 792 observations and 17 variables)
data3<-read_csv("train_data.csv")
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   X1 = col_double(),
##   PassengerId = col_double(),
##   Survived = col_double(),
##   Sex = col_double(),
##   Age = col_double(),
##   Fare = col_double(),
##   Pclass_1 = col_double(),
##   Pclass_2 = col_double(),
##   Pclass_3 = col_double(),
##   Family_size = col_double(),
##   Title_1 = col_double(),
##   Title_2 = col_double(),
##   Title_3 = col_double(),
##   Title_4 = col_double(),
##   Emb_1 = col_double(),
##   Emb_2 = col_double(),
##   Emb_3 = col_double()
## )
```

```
print.data.frame(head(data3))
```

```
##   X1 PassengerId Survived Sex    Age       Fare Pclass_1 Pclass_2 Pclass_3
## 1  0           1        0   1 0.2750 0.01415106        0        0        1
## 2  1           2        1   0 0.4750 0.13913574        1        0        0
## 3  2           3        1   0 0.3250 0.01546857        0        0        1
## 4  3           4        1   0 0.4375 0.10364430        1        0        0
## 5  4           5        0   1 0.4375 0.01571255        0        0        1
## 6  5           6        0   1 0.3500 0.01650950        0        0        1
##   Family_size Title_1 Title_2 Title_3 Title_4 Emb_1 Emb_2 Emb_3
## 1         0.1       1       0       0       0     0     0     1
## 2         0.1       1       0       0       0     1     0     0
## 3         0.0       0       0       0       1     0     0     1
## 4         0.1       1       0       0       0     0     0     1
## 5         0.0       1       0       0       0     0     0     1
## 6         0.0       1       0       0       0     0     1     0
```

Tidy and manipulate data 1:

- My 3 third dataset(data3 is untidy): For making it tidy i did following steps:
- sept1: selected important and meaning ful data from dataset.(subsetting throught select function, i can also use subsetting method from module 3, it will give same result)
- step2: Pclass_1,Pclass_2,Pclass_3 are in wider form, i can covert it into longer for making it in tidy form.
- step3: In Pclass_1,Pclass_2,Pclass_3 (0-means passenger did not got particular Class, taken other class whereas 1 means passenger belongs to that particular class).
- Step4: Filtering class whose value is 1(which passenger got which class(among three classes)) step5: For making it more clear i replaced Pclass_1—>1, Pclass_2—>2, Pclass_3—>3). i can also use labels for it . result will be same.
- step5: dropping “class” variable from datasets, it is showing true for pclass(and this not required now because Pclass variable already describing the class of every passenger) so this is unnecessary.(using select function(select all except one)

```
#making the dataset tidy(dataset3):
#The third dataset is untidy, i am selecting meaniful columns and applying tydr principles:
#selecting important and meaning ful columns:(PassengerId(key column in all three dataset), Sex
 and Survived are (key columnsin data2 and data3 )
data3_tidy <- select(data3,"PassengerId","Survived","Pclass_1","Pclass_2","Pclass_3")
#Pclass is looking untidy, by apply gather function i can make it tidy)
data3_tidy <- data3_tidy %>% pivot_longer(names_to = "Pclass", values_to = "class", cols = 3:5)
#Pclass has three classes and in class column 0 and 1 indicating, 0 for False(has no particular
 class) and 1 for true(has class)
#filter values for to avoid duplicay (passenger has which particular class)
data3_tidy <- data3_tidy%>% filter(class==1)
#After applying filter we can see that each passenger has particular class(Pclass_1 or Pclass_2
 or Pclass_3(where class=1 means True))
#To making it more cleaner, i am replacing Pclass_1 =1,Pclass_2 =2,Pclass_3=3)
data3_tidy$Pclass[data3_tidy$Pclass== "Pclass_1"] <- 1
data3_tidy$Pclass[data3_tidy$Pclass== "Pclass_2"] <- 2
data3_tidy$Pclass[data3_tidy$Pclass== "Pclass_3"] <- 3
#after Appying this we can see that Pclass variable has (1,2,3) as passenger class(which passeng
er has got which class, can be identified by Pclass variable)
#printing tidy dataset3:
#Pclass variable already describing the class of every passenger)
data3_tidy<-data3_tidy%>% select( -(class ))
print.data.frame(head(data3_tidy))
```

```
##   PassengerId Survived Pclass
## 1           1        0      3
## 2           2        1      1
## 3           3        1      3
## 4           4        1      1
## 5           5        0      3
## 6           6        0      3
```

```
data3_tidy$Pclass<-as.numeric(data3_tidy$Pclass)
```

- *I applied Right_join for joining three datasets(as i want all rows of dataset1 and matching rows ot other dataset of with all variables(no duplicate values), by using common key ("PassengerId" in data 1 and data2 whereas 3 common keys in dataset 2 and 3("passengerid,Survived,Pclass))which,including all the variables of both datasets. step2*–>This dataset has double as a numeric and character data types, i converted Sex(Character)column into factor("male,female") and other variables also. Variable conversion information is given in next chunk.

```
#Merging data1 and data 2( Two datasets), as it has passengerid as common key:
#Joing two dataset,(data1 and data2 by right join using common key)
titanic1<-right_join(data1,data2, by="PassengerId")
#joing third dataset with newly joined dataset(titanic1):
Titanic<-right_join(data3_tidy,titanic1)
```

```
## Joining, by = c("PassengerId", "Survived", "Pclass")
```

```
colnames(Titanic)
```

```
##  [1] "PassengerId" "Survived"    "Pclass"      "Nationality" "Name"
##  [6] "Sex"         "Age"         "SibSp"       "Parch"       "Ticket"
## [11] "Fare"        "Cabin"       "Embarked"
```

```
#dropping class variable, it is shwoing true for pclass(and this not required now because
#final dataset after merging all three datasets(891 rows and 13 variables):
print.data.frame(head(Titanic))
```

```
##   PassengerId Survived Pclass               Nationality
## 1           1        0      3             CelticEnglish
## 2           2        1      1             CelticEnglish
## 3           3        1      3 Nordic,Scandinavian,Sweden
## 4           4        1      1             CelticEnglish
## 5           5        0      3             CelticEnglish
## 6           6        0      3             CelticEnglish
##                                                 Name    Sex Age SibSp Parch
## 1                            Braund, Mr. Owen Harris   male  22     1     0
## 2 Cumings, Mrs. John Bradley (Florence Briggs Thayer) female  38     1     0
## 3                             Heikkinen, Miss. Laina female  26     0     0
## 4       Futrelle, Mrs. Jacques Heath (Lily May Peel) female  35     1     0
## 5                           Allen, Mr. William Henry   male  35     0     0
## 6                                   Moran, Mr. James   male  NA     0     0
##             Ticket    Fare Cabin Embarked
## 1        A/5 21171  7.2500  <NA>        S
## 2         PC 17599 71.2833   C85        C
## 3 STON/O2. 3101282  7.9250  <NA>        S
## 4           113803 53.1000  C123        S
## 5           373450  8.0500  <NA>        S
## 6           330877  8.4583  <NA>        Q
```

Understand

- Summarize the types of variables and data structures, check the attributes in the data:
- Types of all variables and information: [1] PassengerId (unique id of passenger)=“double” , [2] Survived(0-not survived,1->survived)= “double”,[3] Pclass(1,2,3 classes of passenger,) =“integer”, [4] Name(Name of passenger) = “character”, [5] Sex (gender of passenger)= “integer”, [6] Age(age of passenger) = “double”, [7] SibSp(siblings per parent(passenger)) = “integer”, [8] Parch(parent per children) = “integer”, [9] Ticket(ticket of passenger) = “character”,[10] Fare(fare of passenger) =“double”, [11] Cabin(cabin alloted to passengers) = “character”, [12] Embarked(city of embarked) = “integer”, [13] Nationality(nationality of passenger) = “character”.
- Structure of all variables:
- [1] PassengerId = col_double(), [2] Survived = col_double(), [3] Pclass = col_double(), [4] Name = col_character(), [5] Sex = col_character(), [6] Age = col_double(),[7] SibSp = col_double(), [8] Parch = col_double(), [9] Ticket = col_character(), [10] Fare = col_double(),[11] Cabin = col_character(), [12] Embarked = col_character(), [13] Nationality = col_character().

- Attributes:

1. Column Names:“PassengerId” “Survived” “Pclass” “Nationality” “Name” ,“Sex”, “Age” “SibSp” ,“Parch” , “Ticket” ,“Fare” , “Cabin” ,“Embarked”
2. row.names:[1:891]

- **Data conversions:**
- Sex(Character) converted to Factor(male, female) , labeled as “Male” and “Female” and odered(alphabetical)
- Survived(numeric) converted to Factor(0,1) , labeled as (0->not survived, 1->survived) and ordered.(0<1)
- PClass(numeric) converted to Factor(1,2,3) and labeled as(1,2,3(indicating passenger class 1,2,3)

```
i didn't label because it is already labeled in class(1,2,3),
applied ordered=True.
```

- Embarked(Character) converted to Factor(Q,S,C) , labeled as(“Cherbourg”,“Queenstown”,“Southampton”), and ordered(alphabetical)
- All factor columns are leveled, labeled and ordered.

```
# This is the R chunk for the Understand Section
#Checking Structure of dataset
str(Titanic)
```

```
## tibble[,13] [891 x 13] (S3: tbl_df/tbl/data.frame)
##  $ PassengerId: num [1:891] 1 2 3 4 5 6 7 8 9 10 ...
##  $ Survived   : num [1:891] 0 1 1 1 0 0 0 0 1 1 ...
##  $ Pclass     : num [1:891] 3 1 3 1 3 3 1 3 3 2 ...
##  $ Nationality: chr [1:891] "CelticEnglish" "CelticEnglish" "Nordic,Scandinavian,Sweden" "Cel
ticEnglish" ...
##  $ Name       : chr [1:891] "Braund, Mr. Owen Harris" "Cumings, Mrs. John Bradley (Florence B
riggs Thayer)" "Heikkinen, Miss. Laina" "Futrelle, Mrs. Jacques Heath (Lily May Peel)" ...
##  $ Sex        : chr [1:891] "male" "female" "female" "female" ...
##  $ Age        : num [1:891] 22 38 26 35 35 NA 54 2 27 14 ...
##  $ SibSp      : num [1:891] 1 1 0 1 0 0 0 3 0 1 ...
##  $ Parch      : num [1:891] 0 0 0 0 0 0 0 1 2 0 ...
##  $ Ticket     : chr [1:891] "A/5 21171" "PC 17599" "STON/O2. 3101282" "113803" ...
##  $ Fare       : num [1:891] 7.25 71.28 7.92 53.1 8.05 ...
##  $ Cabin      : chr [1:891] NA "C85" NA "C123" ...
##  $ Embarked   : chr [1:891] "S" "C" "S" "S" ...
```

```
#Checking attributes of dataset
attributes(Titanic)
```

```
## $names
##  [1] "PassengerId" "Survived"    "Pclass"      "Nationality" "Name"
##  [6] "Sex"         "Age"         "SibSp"       "Parch"       "Ticket"
## [11] "Fare"        "Cabin"       "Embarked"
##
## $row.names
##   [1]   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18
##  [19]  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36
##  [37]  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54
##  [55]  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72
##  [73]  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90
##  [91]  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107 108
## [109] 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126
## [127] 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144
## [145] 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162
## [163] 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180
## [181] 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198
## [199] 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216
## [217] 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234
## [235] 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252
## [253] 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270
## [271] 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288
## [289] 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306
## [307] 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324
## [325] 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342
## [343] 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360
## [361] 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378
## [379] 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396
## [397] 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414
## [415] 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432
## [433] 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450
## [451] 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468
## [469] 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486
## [487] 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504
## [505] 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522
## [523] 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540
## [541] 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558
## [559] 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576
## [577] 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594
## [595] 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612
## [613] 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630
## [631] 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648
## [649] 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666
## [667] 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684
## [685] 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702
## [703] 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720
## [721] 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738
## [739] 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756
## [757] 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774
## [775] 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792
## [793] 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810
## [811] 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828
## [829] 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846
```

```
## [847] 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864
## [865] 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882
## [883] 883 884 885 886 887 888 889 890 891
##
## $class
## [1] "tbl_df"     "tbl"        "data.frame"
```

```
#Type of sex variable is character, converting it into factor
Titanic$Sex<-as.factor(Titanic$Sex)
#checking class of sex variable
class(Titanic$Sex)
```

```
## [1] "factor"
```

```
#converting Sex variable into factor(character to factor)
Titanic$Sex<-factor(Titanic$Sex)
#checking levels
levels(Titanic$Sex)
```

```
## [1] "female" "male"
```

```
#ordering Sex variable
Titanic$Sex<-factor(Titanic$Sex,levels=c("female","male"),labels= c("Male", "Female"), ordered=T
RUE)
head(ordered(Titanic$Sex))
```

```
## [1] Female Male   Male   Male   Female Female
## Levels: Male < Female
```

```
#converting Survived variable into factor
Titanic$Survived<-as.factor(Titanic$Survived)
class(Titanic$Survived)
```

```
## [1] "factor"
```

```
#checking levels
levels(Titanic$Survived)
```

```
## [1] "0" "1"
```

```
#ordering Survived variable(numeric to factor)
Titanic$Survived<-factor(Titanic$Survived,levels=c("0","1"),labels= c("not survived", "survived"
), ordered=TRUE)
head(ordered(Titanic$Survived))
```

```
## [1] not survived survived     survived     survived     not survived
## [6] not survived
## Levels: not survived < survived
```

```
#converting Pclass variable into factor(Numeric to factor)
Titanic$Pclass<-as.factor(Titanic$Pclass)
class(Titanic$Pclass)
```

```
## [1] "factor"
```

```
#checking factors and levels of survived variable
Titanic$Pclass<-factor(Titanic$Pclass)
#Checking levels of Pclass variable
levels(Titanic$Pclass)
```

```
## [1] "1" "2" "3"
```

```
#applying order and labels :
Titanic$Pclass<-factor(Titanic$Pclass,levels=c("1","2", "3") ,labels =c("1","2", "3") , ordered=
TRUE)
head(ordered(Titanic$Pclass))
```

```
## [1] 3 1 3 1 3 3
## Levels: 1 < 2 < 3
```

```
#converting Embarked variable into factor(Numeric to factor)
Titanic$Embarked<-as.factor(Titanic$Embarked)
class(Titanic$Embarked)
```

```
## [1] "factor"
```

```
#checking factors and levels of survived variable
Titanic$Embarked<-factor(Titanic$Embarked)
#Checking levels of Embarked variable
levels(Titanic$Embarked)
```

```
## [1] "C" "Q" "S"
```

```
#applying order and labels :
Titanic$Embarked<-factor(Titanic$Embarked,levels=c( "C","Q", "S") ,labels=c("Cherbourg","Queenst
own","Southampton") , ordered=TRUE)
head(ordered(Titanic$Embarked))
```

```
## [1] Southampton Cherbourg   Southampton Southampton Southampton Queenstown
## Levels: Cherbourg < Queenstown < Southampton
```

Tidy & Manipulate Data I

NOTE- I already tidy my dataset before merging it, (dataset3 was untidy, i used pivot longer and other functions to make it tidy). In my dataset Name variable has multiple puncuation, and quite unclear because it combine lastname,title and firstname, i separated them using separate function. * Secondly, parch and sbips variable might be unclear for many. * Using unite function, i combine two variable into one meaningful variable,here variable “Parch” and “Sbips” gives me total family members of a passenger(not including passenger).if values of family size is zero, it means passenger is traveling alone(without family members).

- As In Name variable last name and first name is not ordered, i used select function for ordering these three columns.* After applying these above mention functions,dataset became tidy and meaningful.

```
#In this datset two variables Parch(Parents per children) and sibps(siblings per parents) can be
united in one varible that is family_size(not include passenger(himself/herself))
Titanic<-Titanic%>%unite(Family_size, Parch,SibSp, sep="")
#Checking type of variable
typeof(Titanic$Family_size)
```

```
## [1] "character"
```

```
#checking class of variable
class(Titanic$Family_size)
```

```
## [1] "character"
```

```
#Converting character column into numeric
Titanic$Family_size<-Titanic$Family_size%>%as.numeric(Titanic$Family_size)
#In this dataset Variable Name contain multiple punctuation, making data untidy
Titanic$Name<-gsub("[[:punct:]]", "", Titanic$Name)
#Using separate function of tidyr package to separate Name column into three new variables:
Titanic<-Titanic%>% tidyr::separate(Name, c("lastname", "Title","firstname"), extra = "drop", fi
ll = "right")
#using select function for selecting columns in appropriate order
Titanic<-select(Titanic,PassengerId,Survived,Pclass,Title,firstname,lastname,Sex,Age,Family_siz
e,Ticket,Fare,Embarked,Cabin,Nationality,Cabin)
print.data.frame(head(Titanic))
```

```
##   PassengerId     Survived Pclass Title firstname  lastname    Sex Age
## 1           1 not survived      3    Mr      Owen    Braund Female  22
## 2           2     survived      1   Mrs      John   Cumings   Male  38
## 3           3     survived      3  Miss     Laina Heikkinen   Male  26
## 4           4     survived      1   Mrs   Jacques  Futrelle   Male  35
## 5           5 not survived      3    Mr   William     Allen Female  35
## 6           6 not survived      3    Mr     James     Moran Female  NA
##   Family_size           Ticket    Fare    Embarked Cabin
## 1           1        A/5 21171  7.2500 Southampton  <NA>
## 2           1         PC 17599 71.2833   Cherbourg   C85
## 3           0 STON/O2. 3101282  7.9250 Southampton  <NA>
## 4           1           113803 53.1000 Southampton  C123
## 5           0           373450  8.0500 Southampton  <NA>
## 6           0           330877  8.4583  Queenstown  <NA>
##                 Nationality
## 1              CelticEnglish
## 2              CelticEnglish
## 3 Nordic,Scandinavian,Sweden
## 4              CelticEnglish
## 5              CelticEnglish
## 6              CelticEnglish
```

Tidy & Manipulate Data II

- *I have given 10 percent discount to every passenger for showing mutatingcan apply mutate, the new variable named as "FARE_after_discount" has been created with 10 %discount.(now i have 15 variables in total)*
- *Free_Fare column* Taking no charges from the people who's age is below % years, here 44 people are allowed to travel with zero charges(includes kids), i saved into other variable, for avoiding loosing other data because of filter function(i did not changed this data in main dataframe).

```
# This is the R chunk for the Tidy & Manipulate Data II
#using mutate function for creating new variable named as 10 per discount for female(Fare_after_
discount)
Titanic<-Titanic%>%mutate(FARE_after_discount=Fare-Fare*0.1)
#Taking no charges from the people who's age is below % years, here 44 people are allowed to tra
vel with zero charges(includes kids)
Free_Fare<-Titanic%>%filter(Age<=5)%>%mutate(Free_Fare= Fare*0)
print.data.frame(head(Titanic))
```

```
##   PassengerId     Survived Pclass Title firstname  lastname    Sex Age
## 1           1 not survived      3    Mr      Owen    Braund Female  22
## 2           2     survived      1   Mrs      John   Cumings   Male  38
## 3           3     survived      3  Miss     Laina Heikkinen   Male  26
## 4           4     survived      1   Mrs   Jacques  Futrelle   Male  35
## 5           5 not survived      3    Mr   William     Allen Female  35
## 6           6 not survived      3    Mr     James     Moran Female  NA
##   Family_size           Ticket    Fare    Embarked Cabin
## 1           1        A/5 21171  7.2500 Southampton  <NA>
## 2           1         PC 17599 71.2833   Cherbourg   C85
## 3           0 STON/O2. 3101282  7.9250 Southampton  <NA>
## 4           1           113803 53.1000 Southampton  C123
## 5           0           373450  8.0500 Southampton  <NA>
## 6           0           330877  8.4583  Queenstown  <NA>
##                 Nationality FARE_after_discount
## 1              CelticEnglish             6.52500
## 2              CelticEnglish            64.15497
## 3 Nordic,Scandinavian,Sweden             7.13250
## 4              CelticEnglish            47.79000
## 5              CelticEnglish             7.24500
## 6              CelticEnglish             7.61247
```

Scan I

Scanning the data for missing values, special values and obvious errors (i.e. inconsistencies).

- For scanning missing values, special values, obvious error, i have applied special function for different data types, like numeric, character and factor.
- I have applied is.specialorNa function for scanning misssing values for each data type as well as i applied this function to each variable.
- No missing values found in PassengerId, title, firstname, lastname and Family size.
- I found missing values in “Age” variable, i replaced it with mean, the reason is its a numeric column, only few rows has missing values, and mean can not affect other values and appropriate way to replace it, so i used mean function to deal with it.
- In Fare column some observation has zero values(assuming fare can not be zero and there is no special case or condition for zero fare), i replaced them with na and then applied mean for average calculation of Fare, and imputed those values with mean.
- In FARE_after_discount column some observation has zero values(as this variable is mutated using Fare variable assuming), i replaced them with na and then applied mean for average calculation of Fare, and imputed those values with mean.
- In ‘Embarked’ variable i found 2 missing values, as it is a factor column, so i can not apply mean and median, so iused mode function for replacing missing values(replaced by most occurring value).
- In cabin column it is about 685 missing values, i can remove it using mode function, but when i observed dataset, i didn’t get the base, or can say on which basis cabins are allotted to the passengers, because it is not related to Pclass variable, as every type of class has cabins,as well as it is not related to survived. Moreover, it not related to age of the people, after some research i came to know that, this field is still a area

of research, researchers are working on it(there is no proof of cabins distribution to passengers), but didn't get the key root for this, on what basis cabins are allotted to the people. If i remove it, it will affect my dataset, so i keeping it as it is because cabins are different for each class.

- At last i have checked obvious errors in all numeric columns.No obvious errors found.

```
# This is the R chunk for the Scan I
#checking total number of missing values(NA)in whole dataset
#which(is.na(Titanic))%>%sum()
#Looking for total number of missing values in each variable
colSums(is.na(Titanic))
```

```
##         PassengerId            Survived              Pclass               Title
##                   0                   0                   0                   0
##           firstname            lastname                 Sex                 Age
##                   0                   0                   0                 177
##         Family_size              Ticket                Fare            Embarked
##                   0                   0                   0                   2
##               Cabin         Nationality FARE_after_discount
##                 687                   0                   0
```

```
#Checking every numerical column whether they have infinite or NaN values using a function calle
d is.specialorNA
is.specialorNA <- function(x){
if (is.numeric(x)) (is.infinite(x) | is.nan(x) | is.na(x))
}
#Applying special function in PassengerID variable for finding total of na, nas, or special valu
e
sapply(Titanic$PassengerId, is.specialorNA)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Family_size variable for finding total of na, nas, or special valu
e
sapply(Titanic$Family_size, is.specialorNA)%>%sum()
```

```
## [1] 0
```

```
#Applying special function Age variable for finding total of na, nas, or special value
sapply(Titanic$Age, is.specialorNA)%>%sum()
```

```
## [1] 177
```

```
#Applying special function in Fare variable for finding total of na, nas, or special value
sapply(Titanic$Fare, is.specialorNA)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Fare_after_discount variable for finding total of na, nas, or spec
ial value
sapply(Titanic$FARE_after_discount, is.specialorNA)%>%sum()
```

```
## [1] 0
```

```
# Checking every factor column whether they have infinite or NaN values using a function named i
s.specialorna
is.specialorNA_F <- function(x){
if (is.factor(x)) (is.infinite(x) | is.nan(x) | is.na(x))
}
#Applying special function in Survived variable for finding total of na, nas, or special value
sapply(Titanic$Survived, is.specialorNA_F)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Pclass variable for finding total of na, nas, or special value
sapply(Titanic$Pclass, is.specialorNA_F)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Sex variable for finding total of na, nas, or special value
sapply(Titanic$Sex, is.specialorNA_F)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Embarked variable for finding total of na, nas, or special value
sapply(Titanic$Embarked, is.specialorNA_F)%>%sum()
```

```
## [1] 2
```

```
#Check every Character column whether they have infinite or NaN values using a function called i
s.special
is.specialorNA_C <- function(x){
if (is.character(x)) (is.infinite(x) | is.nan(x) | is.na(x))
}
#Applying special function in title variable for finding total of na, nas, or special value
sapply(Titanic$Title, is.specialorNA_C)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in firstname variable for finding total of na, nas, or special value
sapply(Titanic$firstname, is.specialorNA_C)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in  lastname variable for finding total of na, nas, or special value
sapply(Titanic$lastname, is.specialorNA_C)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Ticket variable for finding total of na, nas, or special value
sapply(Titanic$Ticket, is.specialorNA_C)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in Nationality variable for finding total of na, nas, or special valu
e
sapply(Titanic$Nationality, is.specialorNA_C)%>%sum()
```

```
## [1] 0
```

```
#Applying special function in cabin variable for finding total of na, nas, or special value
sapply(Titanic$Cabin, is.specialorNA_C)%>%sum()
```

```
## [1] 687
```

```
#I found na as a missing values in Age variable , so imputing it with mean:
#Applying special function in age variable for finding total of na, nas, or special value
Titanic$Age[is.na(Titanic$Age)] <- mean(Titanic$Age, na.rm = TRUE)
#varifying after imputing with mean values
which(is.na(Titanic$Age))
```

```
## integer(0)
```

```
#I found 0 as a value as a missing values in Fare column, so dealing with it
Titanic$Fare[Titanic$Fare== "0"] <- NA
#Replacing na in Fare variable by mean
Titanic$Fare[is.na(Titanic$Fare)] <- mean(Titanic$Fare, na.rm = TRUE)
#verifying missing values areputed by mean
which(is.na(Titanic$Fare))
```

```
## integer(0)
```

```
#Replacing na in Fare_after_discount(0-->NA) variable by mean
Titanic$FARE_after_discount[Titanic$FARE_after_discount== "0"] <- NA
Titanic$FARE_after_discount[is.na(Titanic$FARE_after_discount)] <- mean(Titanic$FARE_after_disco
unt, na.rm = TRUE)
#verifying missing values of Fare_after_discount are imputed by mean
which(is.na(Titanic$FARE_after_discount))
```

```
## integer(0)
```

```
#Found missing values in Embarked Variable, i replaced it with mode function
Titanic$Embarked<-Hmisc::impute(Titanic$Embarked, fun= mode)
#Verifiying missing values are replaced or not:
which(is.na(Titanic$Embarked))
```

```
## named integer(0)
```

```
#Looking for obvious errors:
obivious_E<- editset(c("Age>=0", "Age<=100","Fare>=0","Family_size>=0"))
#violated Edits returns a logical array indicating for each row of the data, which rules are vio
lated.
#Applying rules to find out the violation
Violated<- violatedEdits(obivious_E, Titanic)
summary(Violated)
```

```
## No violations detected, 0 checks evaluated to NA
```

```
## NULL
```

Scan II

Scanning the numeric data for outliers:(i used both the Tukey's method and Z_score method to see the difference)

There are five numeric columns in my dataset.

First, "**Tukey's method of outlier detection**" is used to detect outliers. Outliers are detected in [Age, Fare, FAmily_size, Fare_after_discount] varibales. I can not impute them beacuse they are not entery errors, persong can have age (0 to 80), fare is (0 to 550), family *size(0 to 51), Fare* after discount(0 to550)

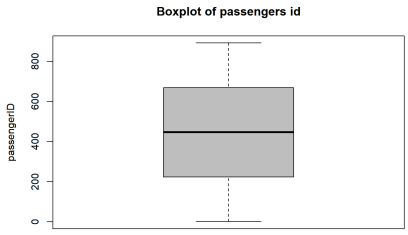
* For confirming the location i used Zscore method also,i checked passengerID variable for outliers using z-score method (is a parametric way of detecting outliers and assumes that the underlying data is normally distributed) I have checked distribution of each numerical variable using histrogram, i found that all the variables are normaly distributed.

NOTE: Because of space i can not show hist() of all variables, by giving range and reason, it could be clear of using univariate Zscore method.(I tried for multivariate but my R studio is giving error, i sent mail regarding this, so i choose to use z_score method for each normally distributed variable)

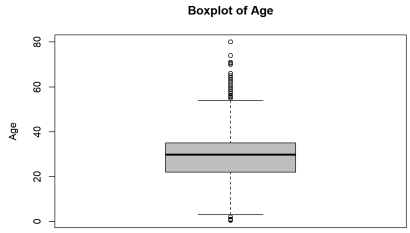
- Range of Passenger id(1 to 891) No missing value.

- Range of Age(1 to 80), missing values are imputed by mean.
- Range of Fare(1 to 550), zero values are imputed by mean.
- Range of Family_size(0 to 51) no missing value.
- Fair_After_discount(0 to 500)
- No outliers are detected in PassengerId variable
- By applying z-score method for detecting outliers in ‘Age’ variable, i found 7 in total, and observed the values carefully, and i found age greater than 70 considered as outliers, but i can not impute them, because a person can have a age of 70 years(because dataset has a range 1 to 80), so actually these are not a data entry errors/typos, 7 passengers are older than others and passengers can have variation in age, that is why it showing their age as outliers.Imputing them will affect the data and it will not be appropriate.
- i used z_score method for detecting outliers in ‘Fare’ column. because it has normal distribution(checked by histogram), i 20 found outliers. but this is not a data entry error/typos, fare can be varied according to class(passenger class 1 has higher fare as compare to class 2 and class 3), in this titanic dataset 3 classes are there so imputing them with mean or any other will not be appropriate.
- i used z_score method for detecting outliers in ‘Fare_after_discount’ column. because it has normal distribution(checked by histogram), i found outliers (20), but this is not a data entry error/typos, fare can be varied according to class(passenger class 1 has higher fare as compare to class 2 and class 3, so discount will also be varied), in this titanic dataset 3 classes are there so imputing them with mean or any other will not be appropriate.
- i used z_score method(standardised score (z-score) of all observations are calculated) for detecting outliers in ‘Family_Size’ column. because it has normal distribution(checked by histogram), i found outlier at one position at 14, after checking that value in dataset it shows family_size 51(as family_size consists of many people including step sister, brother,their husband/wife as well),actually this is not a data entry error/typos,imputing it with mean or median would be wrong.
- I have shown outlier removing process using capping method where i found outliers and saved into variable, changing in main dataframe will be unappropriate as these are not errors.

```
#“Tukey’s method of outlier detection in variable passenger id:
Titanic$PassengerId %>% boxplot(main="Boxplot of passengers id", ylab="passengerID", col = "gre
y")
```

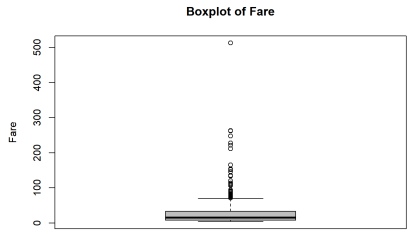



```
#“Tukey’s method of outlier detection in variable Age:
Titanic$Age %>% boxplot(main="Boxplot of Age", ylab="Age", col = "grey")
```

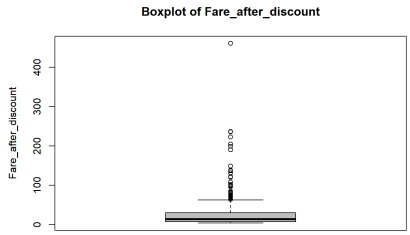



```
#"Tukey's method of outlier detection in variable family_size:
Titanic$Family_size %>% boxplot(main="Boxplot of family_size", ylab="family_size", col = "grey")
```


```
#"Tukey's method of outlier detection in variable Fare:
Titanic$Fare %>% boxplot(main="Boxplot of Fare", ylab="Fare", col = "grey")
```




```
#"Tukey's method of outlier detection in variable Fare_after_discount:
Titanic$FARE_after_discount %>% boxplot(main="Boxplot of Fare_after_discount", ylab="Fare_after_
discount", col = "grey")
```




```
#Using Z_score method on passengerId variable for finding outliers, distribution is normal
z.scores_Id <- Titanic$PassengerId%>%scores(type = "z")
z.scores_Id %>% summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -1.7291 -0.8646  0.0000  0.0000  0.8646  1.7291
```

```
#Looking for the location
which(abs(z.scores_Id) >3 )
```

```
## integer(0)
```

```
#for location(total)
length (which(abs(z.scores_Id) >3 ))
```

```
## [1] 0
```

```
#Using Z_score method on Age variable for finding outliers, distribution is normal
z.scores_Age <- Titanic$Age%>%scores(type = "z")
#Calculating summary
z.scores_Age %>% summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -2.2519 -0.5921  0.0000  0.0000  0.4077  3.8687
```

```
#Looking for the location
which(abs(z.scores_Age) >3 )
```

```
## [1]  97 117 494 631 673 746 852
```

```
#for location(total)
length (which(abs(z.scores_Age) >3 ))
```

```
## [1] 7
```

```
#using Z_score method on fare variable for finding outliers:
z.scores_fare <- Titanic$Fare%>%scores(type = "z")
#Calculating summary
z.scores_fare %>% summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.5805 -0.5015 -0.3566  0.0000  0.0000  9.6856
```

```
#Looking for the location
which(abs(z.scores_fare) >3 )
```

```
##  [1]  28  89 119 259 300 312 342 378 381 439 528 558 680 690 701 717 731 738 743
## [20] 780
```

```
#finding total number of outliers using length
length (which(abs(z.scores_fare) >3 ))
```

```
## [1] 20
```

```
Z.scores_Fam<-Titanic$Family_size%>%scores(type = "z")
Z.scores_Fam%>%summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.5059 -0.5059 -0.5059  0.0000 -0.2727  6.6062
```

```
which(abs(Z.scores_Fam) >3 )
```

```
##  [1]  14  26  87 168 361 438 439 568 611 639 679 737 775 886
```

```
length (which(abs(Z.scores_Fam) >3 ))
```

```
## [1] 14
```

```
#Applying Z_score method on Fare_after_discount variable:
Z.scores_Fd<-Titanic$FARE_after_discount%>%scores(type = "z")
Z.scores_Fd%>%summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
## -0.5805 -0.5015 -0.3566  0.0000  0.0000  9.6856
```

```
which(abs(Z.scores_Fd) >3 )
```

```
##  [1]  28  89 119 259 300 312 342 378 381 439 528 558 680 690 701 717 731 738 743
## [20] 780
```

```
length (which(abs(Z.scores_Fd) >3 ))
```

```
## [1] 20
```

```
#handling outliers:
#For all the univariate outliers, Capping method will be applied to replace them #with the neare
st neighbours that are not outliers.
#Define a function to cap the values outside the limits

cap <- function(x){
    quantiles <- quantile( x, c(.05, 0.25, 0.75, .95 ) )
    x[ x < quantiles[2] - 1.5*IQR(x) ] <- quantiles[1]
    x[ x > quantiles[3] + 1.5*IQR(x) ] <- quantiles[4]
    x
}
#Imputing outliers in Age variable:
Age_capped <-Titanic$Age %>% cap()
#varification if outliers are imputed otr not
Age_capped_zscore<-Age_capped  %>% scores(type = "z")
length(which(abs(Age_capped_zscore) >3))
```

```
## [1] 0
```

```
#Imputing outliers in Family_size variable:
Fam_capped <-Titanic$Family_size %>% cap()
#varification if outliers are imputed otr not
Fam_capped_zscore<-Fam_capped  %>% scores(type = "z")
length(which(abs(Fam_capped_zscore) >3))
```

```
## [1] 0
```

```
#Imputing outliers in Fare variable:
Fare_capped <-Titanic$Fare %>% cap()
#varification if outliers are imputed otr not
Fare_capped_zscore<-Fare_capped  %>% scores(type = "z")
length(which(abs(Fare_capped_zscore) >3))
```

```
## [1] 0
```

```
#Imputing outliers in Fare_after_discout variable:
FAD_capped <-Titanic$FARE_after_discount %>% cap()
#varification if outliers are imputed otr not
FAD_capped_zscore<-FAD_capped  %>% scores(type = "z")
length(which(abs(FAD_capped_zscore) >3))
```

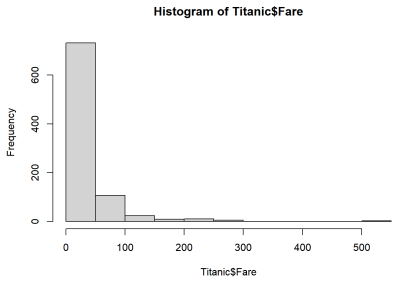
```
## [1] 0
```

Transform

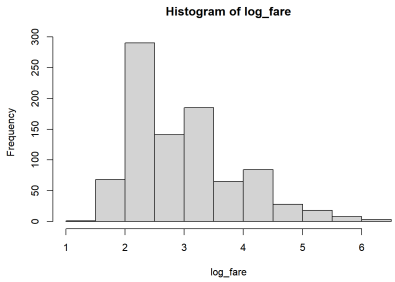
- For normal distribution in “Fare” column i used histogram.

- hist(Fare) is showing right skewness, reducing right skewness in Fair variable, applying log()->loge function for making it more symmetric.(i checked other functions also, but below function giving appropriate result. Pages are limited that is why showing only one more symmetric method)
- Using normalisation for removing right skewness from all numerical variables.

```
#checking distribution of variable Fare in
knitr::opts_chunk$set(fig.width=12, fig.height=8)
hist(Titanic$Fare)
```




```
#reducing right skewness, applying log e function
log_fare<-log(Titanic$Fare)
#plotting histogram after applying log e function
hist(log_fare)
```




```
# The logarithmic transformation has gotten rid of the right-skewness, now it looks normally dis
tributed
# using Data normalisation (z score standardisation)
# Create subset using all variables except State and Crime
z_scale<-Titanic%>%select(PassengerId,Fare,Family_size,FARE_after_discount,Age)
# Apply mean centering
z_normalisation <- scale(z_scale, center = TRUE, scale = TRUE)
head(z_normalisation)
```

```
##      PassengerId       Fare Family_size FARE_after_discount        Age
## [1,]   -1.729137 -0.5151177  -0.3892940          -0.5151177 -0.5921480
## [2,]   -1.725251  0.7781128  -0.3892940           0.7781128  0.6384304
## [3,]   -1.721365 -0.5014852  -0.5058859          -0.5014852 -0.2845034
## [4,]   -1.717480  0.4108789  -0.3892940           0.4108789  0.4076970
## [5,]   -1.713594 -0.4989607  -0.5058859          -0.4989607  0.4076970
## [6,]   -1.709708 -0.4907146  -0.5058859          -0.4907146  0.0000000
```