Project 1

Yash Prashant Gokhale

College of Professional Studies, Northeastern University

ALY6000: Introduction to Analytics

Prof. Richard He

September 25, 2023

Introduction

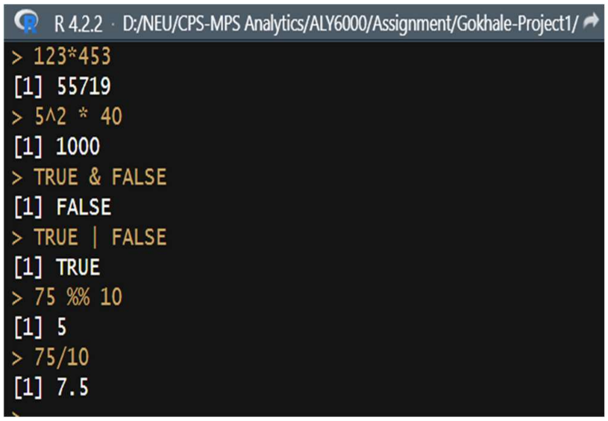
This project acts as an introductory session to the R language and at the same time gives an idea as to how statistics can be seamlessly performed through the language. Vectors and Data Frames are the data types that have been stressed throughout the project.

Various handy, in-built functions such as “c”, “seq”,” rep”, statistical functions like sum, mean, median, max, min are used. Accessing data from a csv file using “data_csv()” function and then using functions like head(),name(), select(),arrange(),filter(),slice(),mutate() are implemented so as to have a clearer understanding of what the data frame is, which is almost always the very first step of EDA.

Function like the ggplot() is used to visualize a part of data so as to convey the story behind the data graphically.

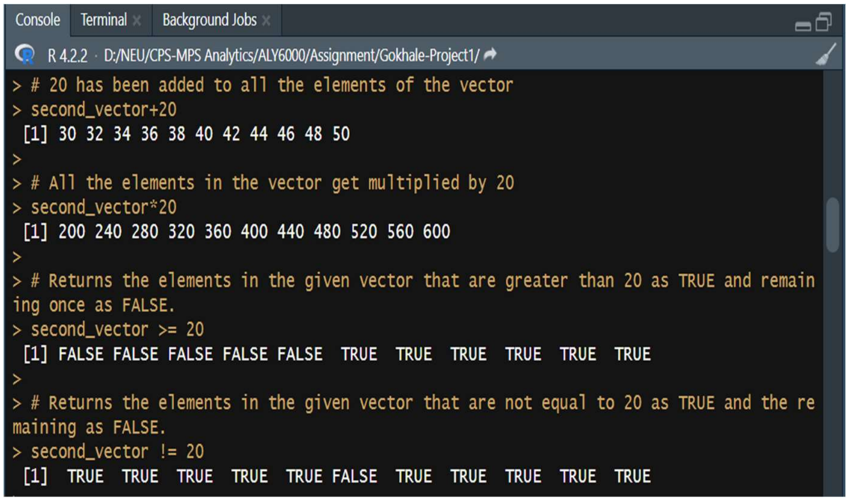
Key Findings

(1) Along with the use of sophisticated statistical functions, R can also be used to compute simple calculations as shown below.



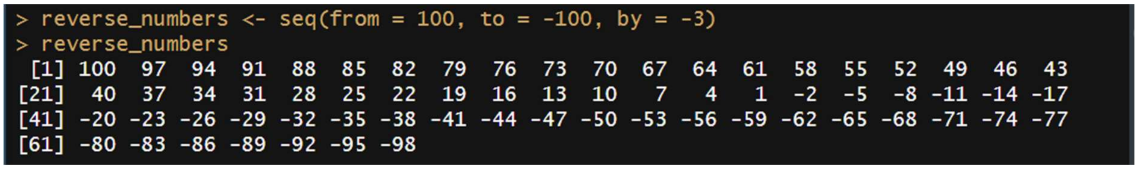
```
R 4.2.2 · D:/NEU/CPS-MPS Analytics/ALY6000/Assignment/Gokhale-Project1/
> 123*453
[1] 55719
> 5^2 * 40
[1] 1000
> TRUE & FALSE
[1] FALSE
> TRUE | FALSE
[1] TRUE
> 75 %% 10
[1] 5
> 75/10
[1] 7.5
>
```

(2) The code snippet below shows as to how the Broadcasting works in R.
The first line adds 20 to all the elements of the vector ‘second_vector’ while the next line multiplies all the elements of the ‘second_vector’ by 20.
The next two lines compare all the elements of the vector ‘second_vector’ with 20 and accordingly return a Boolean vector with TRUE corresponding to 1 and FALSE corresponding to 0.
These lines in no way mutate or change the original vector ‘second_vector’.


Console
Terminal
Background Jobs
R 4.2.2 · D:/NEU/CPS-MPS Analytics/ALY6000/Assignment/Gokhale-Project1/

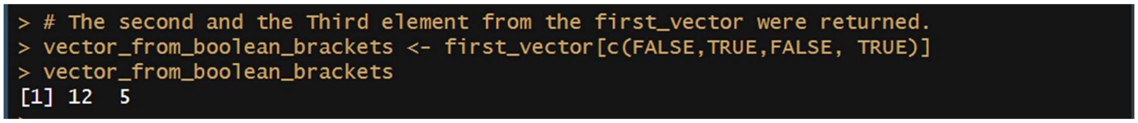
```
> # 20 has been added to all the elements of the vector
> second_vector+20
 [1] 30 32 34 36 38 40 42 44 46 48 50
>
> # All the elements in the vector get multiplied by 20
> second_vector*20
 [1] 200 240 280 320 360 400 440 480 520 560 600
>
> # Returns the elements in the given vector that are greater than 20 as TRUE and remain
ing once as FALSE.
> second_vector >= 20
 [1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
>
> # Returns the elements in the given vector that are not equal to 20 as TRUE and the re
maining as FALSE.
> second_vector != 20
 [1]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE
```

(3) seq() function is used to generate a sequence of elements for a vector. The parameters like 'from' and 'to' act as starting and ending points for the sequence generation while 'by' can be used to provide an equal increment/decrement.



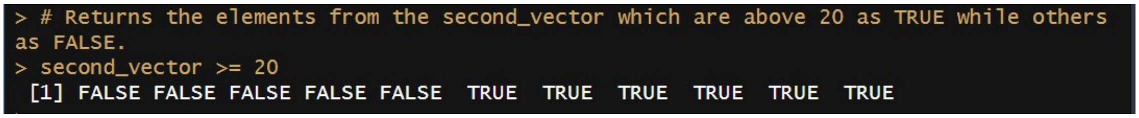
```
> reverse_numbers <- seq(from = 100, to = -100, by = -3)
> reverse_numbers
 [1] 100  97  94  91  88  85  82  79  76  73  70  67  64  61  58  55  52  49  46  43
[21]  40  37  34  31  28  25  22  19  16  13  10   7   4   1  -2  -5  -8 -11 -14 -17
[41] -20 -23 -26 -29 -32 -35 -38 -41 -44 -47 -50 -53 -56 -59 -62 -65 -68 -71 -74 -77
[61] -80 -83 -86 -89 -92 -95 -98
```

(4) A Boolean vector is created using the combine function and 'first_vector' as it's parameter. The corresponding 'TRUE' values are returned.



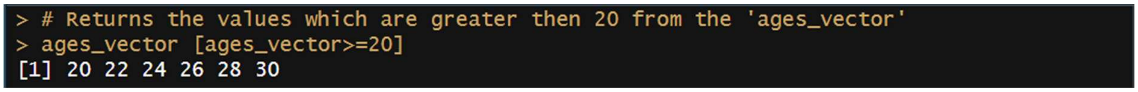
```
> # The second and the Third element from the first_vector were returned.
> vector_from_boolean_brackets <- first_vector[c(FALSE,TRUE,FALSE, TRUE)]
> vector_from_boolean_brackets
[1] 12  5
```

(5) "second_vector >= 20"returns a Boolean vector which states 'TRUE' for the values that are above 20 and 'FALSE' for the values less than 20.



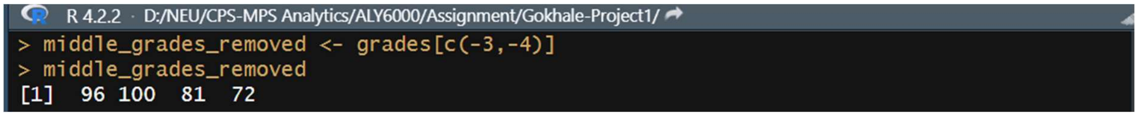
```
> # Returns the elements from the second_vector which are above 20 as TRUE while others
as FALSE.
> second_vector >= 20
 [1] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

(6) 'ages_vector' returns the values that are above 20 using Booleans as a mode of filtrations.



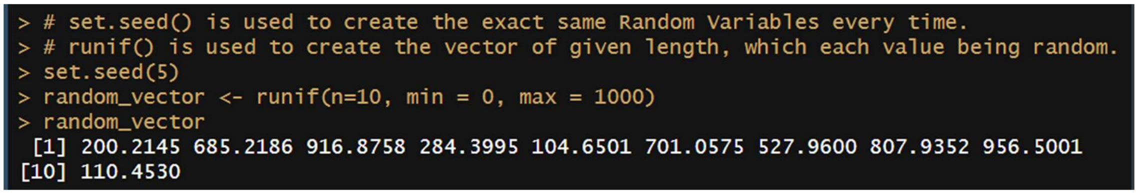
```
> # Returns the values which are greater then 20 from the 'ages_vector'
> ages_vector [ages_vector>=20]
[1] 20 22 24 26 28 30
```

(7) By passing negative indexes to the combine function, we can drop the corresponding elements from the vector as displayed below.


R 4.2.2 · D:/NEU/CPS-MPS Analytics/ALY6000/Assignment/Gokhale-Project1/

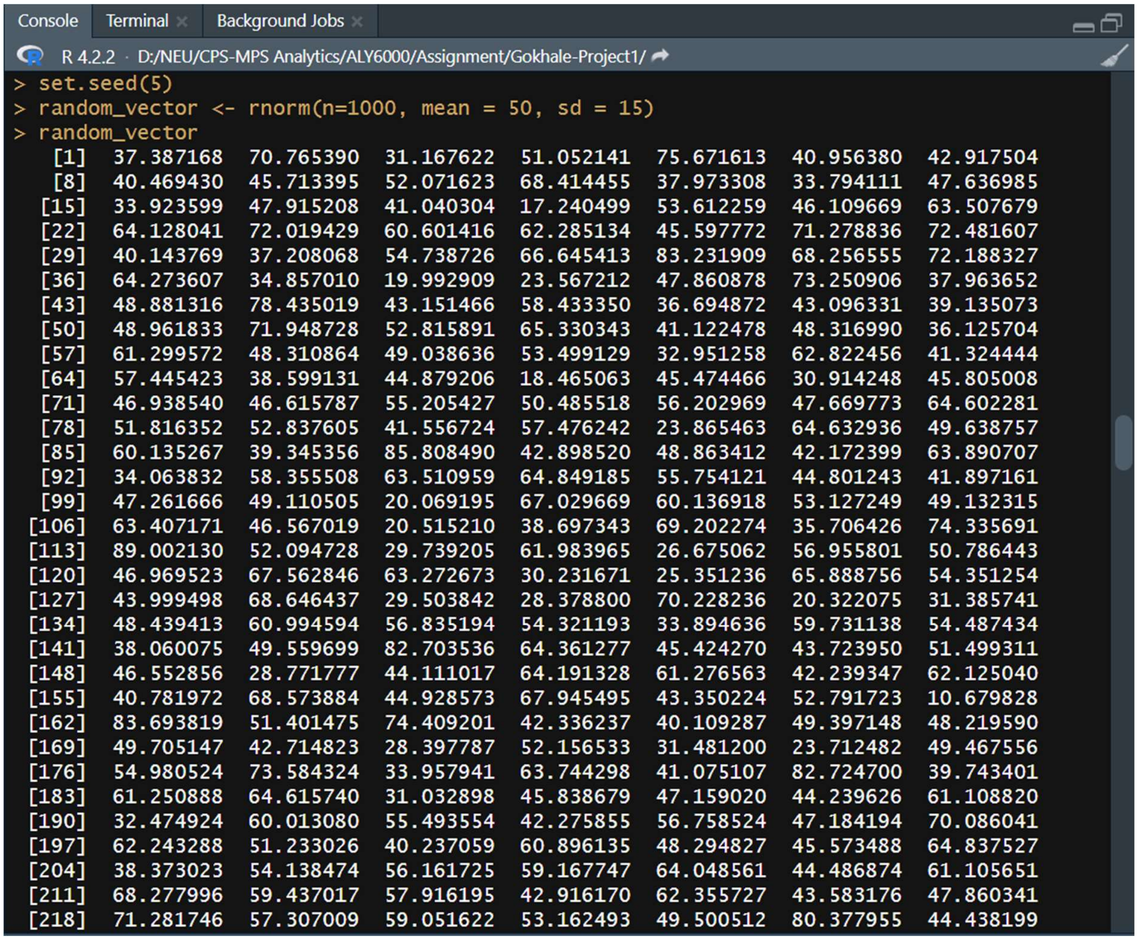
```
> middle_grades_removed <- grades[c(-3,-4)]
> middle_grades_removed
[1]  96 100  81  72
```

(8) set.seed() is used to create the exact same random variables every time while the runif() is used to create a random variables with the arguments as minimum(starting point), maximum(end point) and the actual length of the vector.



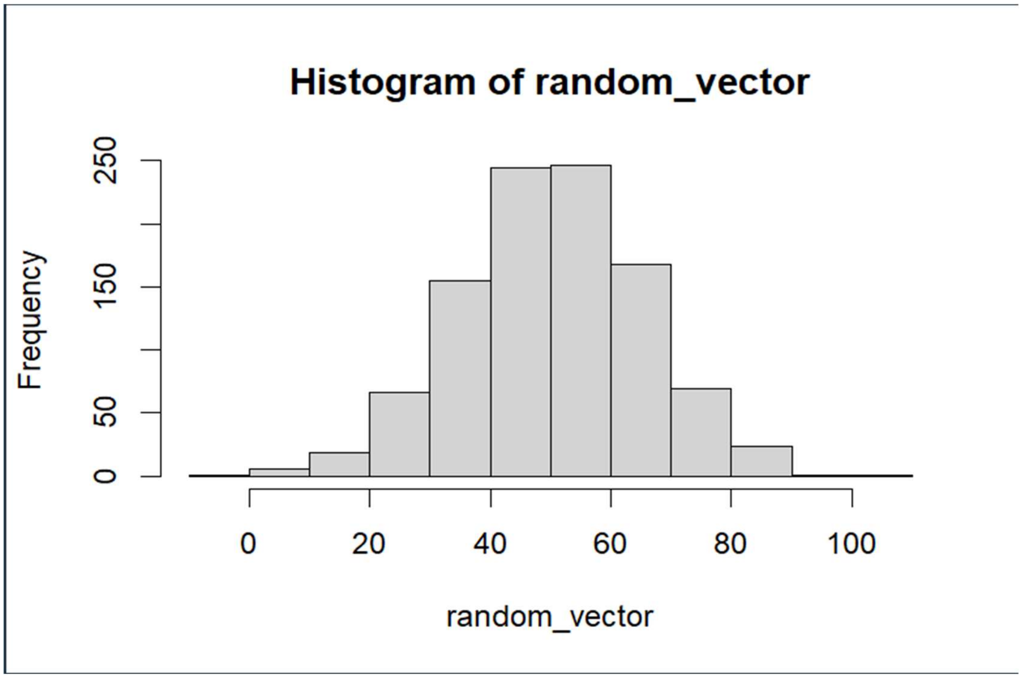
```
> # set.seed() is used to create the exact same Random Variables every time.
> # runif() is used to create the vector of given length, which each value being random.
> set.seed(5)
> random_vector <- runif(n=10, min = 0, max = 1000)
> random_vector
 [1] 200.2145 685.2186 916.8758 284.3995 104.6501 701.0575 527.9600 807.9352 956.5001
[10] 110.4530
```

(9) The below code shows how to create random variables using normal distribution. Here a 1000 values with a mean value of 50 and the standard deviation of 15 are sampled out from a Gaussian Distribution using the “rnorm” function.

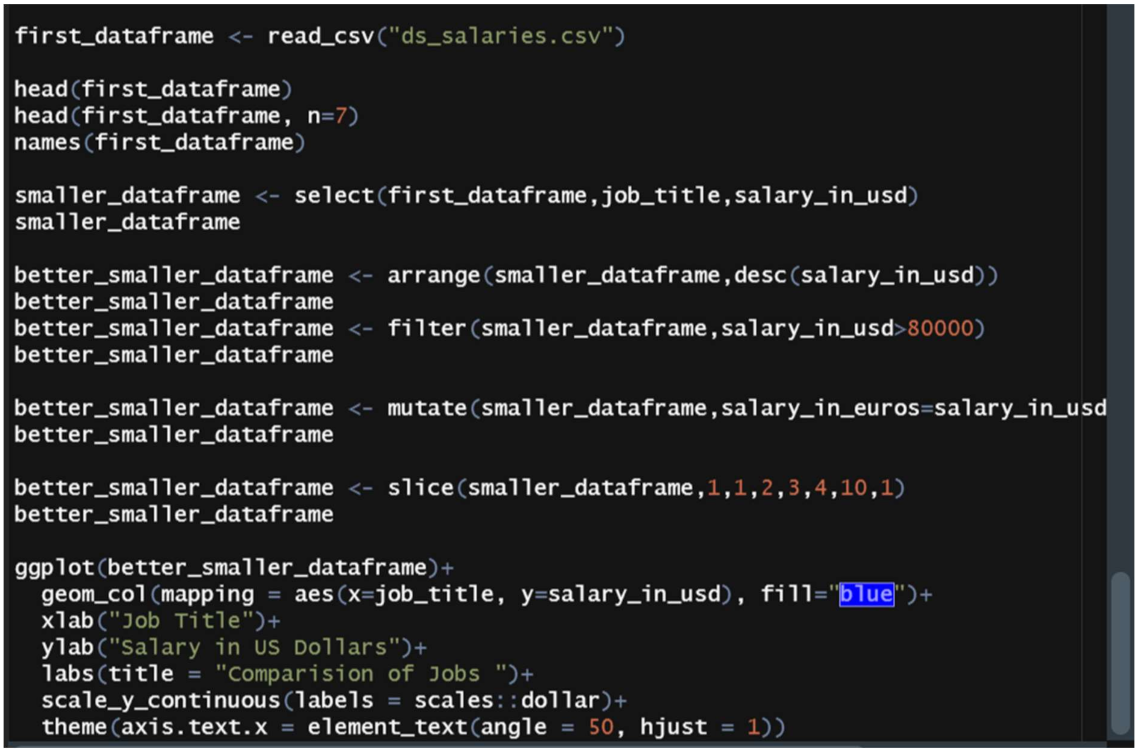

Console
Terminal
Background Jobs
R 4.2.2 · D:/NEU/CPS-MPS Analytics/ALY6000/Assignment/Gokhale-Project1/

```
> set.seed(5)
> random_vector <- rnorm(n=1000, mean = 50, sd = 15)
> random_vector
   [1]  37.387168  70.765390  31.167622  51.052141  75.671613  40.956380  42.917504
   [8]  40.469430  45.713395  52.071623  68.414455  37.973308  33.794111  47.636985
  [15]  33.923599  47.915208  41.040304  17.240499  53.612259  46.109669  63.507679
  [22]  64.128041  72.019429  60.601416  62.285134  45.597772  71.278836  72.481607
  [29]  40.143769  37.208068  54.738726  66.645413  83.231909  68.256555  72.188327
  [36]  64.273607  34.857010  19.992909  23.567212  47.860878  73.250906  37.963652
  [43]  48.881316  78.435019  43.151466  58.433350  36.694872  43.096331  39.135073
  [50]  48.961833  71.948728  52.815891  65.330343  41.122478  48.316990  36.125704
  [57]  61.299572  48.310864  49.038636  53.499129  32.951258  62.822456  41.324444
  [64]  57.445423  38.599131  44.879206  18.465063  45.474466  30.914248  45.805008
  [71]  46.938540  46.615787  55.205427  50.485518  56.202969  47.669773  64.602281
  [78]  51.816352  52.837605  41.556724  57.476242  23.865463  64.632936  49.638757
  [85]  60.135267  39.345356  85.808490  42.898520  48.863412  42.172399  63.890707
  [92]  34.063832  58.355508  63.510959  64.849185  55.754121  44.801243  41.897161
  [99]  47.261666  49.110505  20.069195  67.029669  60.136918  53.127249  49.132315
 [106]  63.407171  46.567019  20.515210  38.697343  69.202274  35.706426  74.335691
 [113]  89.002130  52.094728  29.739205  61.983965  26.675062  56.955801  50.786443
 [120]  46.969523  67.562846  63.272673  30.231671  25.351236  65.888756  54.351254
 [127]  43.999498  68.646437  29.503842  28.378800  70.228236  20.322075  31.385741
 [134]  48.439413  60.994594  56.835194  54.321193  33.894636  59.731138  54.487434
 [141]  38.060075  49.559699  82.703536  64.361277  45.424270  43.723950  51.499311
 [148]  46.552856  28.771777  44.111017  64.191328  61.276563  42.239347  62.125040
 [155]  40.781972  68.573884  44.928573  67.945495  43.350224  52.791723  10.679828
 [162]  83.693819  51.401475  74.409201  42.336237  40.109287  49.397148  48.219590
 [169]  49.705147  42.714823  28.397787  52.156533  31.481200  23.712482  49.467556
 [176]  54.980524  73.584324  33.957941  63.744298  41.075107  82.724700  39.743401
 [183]  61.250888  64.615740  31.032898  45.838679  47.159020  44.239626  61.108820
 [190]  32.474924  60.013080  55.493554  42.275855  56.758524  47.184194  70.086041
 [197]  62.243288  51.233026  40.237059  60.896135  48.294827  45.573488  64.837527
 [204]  38.373023  54.138474  56.161725  59.167747  64.048561  44.486874  61.105651
 [211]  68.277996  59.437017  57.916195  42.916170  62.355727  43.583176  47.860341
 [218]  71.281746  57.307009  59.051622  53.162493  49.500512  80.377955  44.438199
```

(10)R can be extensively used to create engaging visualizations, like histogram as shown below. The hist function is passed with "random_vector" to plot the histogram with the 1000 samples with the mean of 50 and SD of 15.

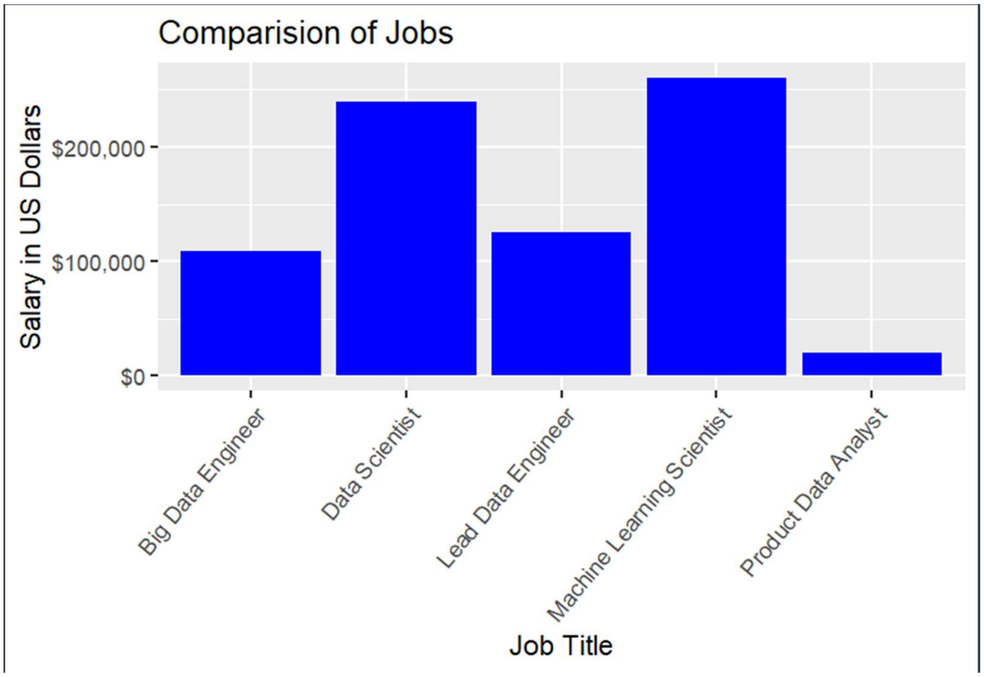


(11)The following lines of R code prove as to how flexible and versatile the R language is when it comes to analyzing the data frames. The readily available functions like head, select, arrange, filter, mutate and slice are extremely useful to get an overview as to how the data is structured and do the data conversions to gain insights out of it.



```
first_dataframe <- read_csv("ds_salaries.csv")

head(first_dataframe)
head(first_dataframe, n=7)
names(first_dataframe)

smaller_dataframe <- select(first_dataframe,job_title,salary_in_usd)
smaller_dataframe

better_smaller_dataframe <- arrange(smaller_dataframe,desc(salary_in_usd))
better_smaller_dataframe
better_smaller_dataframe <- filter(smaller_dataframe,salary_in_usd>80000)
better_smaller_dataframe

better_smaller_dataframe <- mutate(smaller_dataframe,salary_in_euros=salary_in_usd
better_smaller_dataframe

better_smaller_dataframe <- slice(smaller_dataframe,1,1,2,3,4,10,1)
better_smaller_dataframe

ggplot(better_smaller_dataframe)+
  geom_col(mapping = aes(x=job_title, y=salary_in_usd), fill="blue")+
  xlab("Job Title")+
  ylab("Salary in US Dollars")+
  labs(title = "Comparision of Jobs ")+
  scale_y_continuous(labels = scales::dollar)+
  theme(axis.text.x = element_text(angle = 50, hjust = 1))
```

(12) As mentioned previously R has extensive functions and ability to create visualization which truly convey insights in a compelling graphical manner. ggplot() from tidyverse package is extremely useful to create visualizations from the dataset or data frames. Various parameters like data-frame, labels, title, legend, color can be passed to create graphs. In the assignment a bar graph showing the "Comparison of the Job" is visualized with x-axis being "Job Title" while the y-axis being "Salary in USD"



Conclusion

The assignment acted has a good exercise to gently introduce the R scripting language. It was quite good to learn and understand functions of R which have tremendous potential to not just to EDA but also carry out statistical analysis. R language also has a plethora of functions to visualize the data in a compelling manner. Overall, the assignment successfully helped me understand the basic functionality of R.

References

https://www.rdocumentation.org/packages/pacman/versions/0.5.1

https://www.rdocumentation.org/packages/ggplot2/versions/0.9.0/topics/ggplot

https://www.rdocumentation.org/packages/tidyverse/versions/2.0.0