Health Management Organization

Project Report

IST 687: Introduction to Data Science

Group 3: **Jainish Savaliya(SUID: 225038319)**
Priyanka Seth(SUID:730444482)
Manasi Rathi(SUID:618023107)
Maitreyi Ahire(SUID:830350745)

Table of Content

(1)Introduction

- As Consultants for the health management organization, the main objective is to predict future healthcare costs and provide actionable insight to the HMO to lower healthcare costs by specific recommendations on how to do that.

- **Getting a better understanding of the problem:**
At first, we noted all the essential points and considered all the variables, and tried to decide what approach and models would be the best to get more valuable insights. After that, we decided the business questions needed to be asked to identify the key factors and appropriate actions.

(2) Business Questions Addressed

- What is the number of the total people we doing the analysis of?
- What are the factors that affect the cost of health care the for those people?
- What are the outliers?
- What relations of those factors affect the level of the cost the most?
- What actions should be taken to balance out the healthcare cost?

(3)Data Acquisition, Cleansing, Transformation, and Munging

- After acquiring the relevant data, the most important phase is data cleaning and preparation. The data may contain missing values and outliers. As a result, we must clean the data and choose just important attributes. We also did exploratory data analysis, which involved using some fundamental methods such as summary(), table(), View(), head(), and so on, as well as more complex plots, to better comprehend the data. where several fundamental methods such as summary(), str(), View(), head(), and others were utilized advanced charts for data comprehension

Step 1: Gathering the relevant data:

We used the given link to get the data set:

https://intro-datascience.s3.us-east-2.amazonaws.com/HMO_data.csv

```
       X                  age             bmi           children         smoker         
 Min.   :        1   Min.   :18.00   Min.   :15.96   Min.   :0.000   Length:7582       
 1st Qu.:     5635   1st Qu.:26.00   1st Qu.:26.60   1st Qu.:0.000   Class :character  
 Median :    24916   Median :39.00   Median :30.50   Median :1.000   Mode  :character  
 Mean   :   712602   Mean   :38.89   Mean   :30.80   Mean   :1.109                     
 3rd Qu.:   118486   3rd Qu.:51.00   3rd Qu.:34.77   3rd Qu.:2.000                     
 Max.   :131101111   Max.   :66.00   Max.   :53.13   Max.   :5.000                     
                                     NA's   :78                                        
   location         location_type      education_level    yearly_physical      exercise        
 Length:7582        Length:7582        Length:7582        Length:7582        Length:7582       
 Class :character   Class :character   Class :character   Class :character   Class :character  
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character  
```

```
   married           hypertension        gender               cost      
 Length:7582        Min.   :0.0000   Length:7582        Min.   :    2  
 Class :character   1st Qu.:0.0000   Class :character   1st Qu.:  970  
 Mode  :character   Median :0.0000   Mode  :character   Median : 2500  
                    Mean   :0.2005                      Mean   : 4043  
                    3rd Qu.:0.0000                      3rd Qu.: 4775  
                    Max.   :1.0000                      Max.   :55715  
                    NA's   :80                                         
```

Step 2: installed all the packages

- After getting the data sets we decided what are the next steps going to be during our project so we library all the necessary files in advance.

library(dplyr)
library(ggplot2)
library(tidyverse)
library(rsample)
library(caret)
library(kernlab)
library(e1071)
library(arules)
library(arulesViz)
library(imputeTS)
library(rio)

library(rpart)
library(rpart.plot)
library(shiny)
library(shinydashboard)

Step 3: Cleaning the data frame

So in this step first, we checked if any null values were available in the data frame and later on we tried to remove them via the interpolation process.

```
# cleaning the dataframe

# Checking for NA values in all columns
colSums(is.na(data))
anyNA(data)
# Removing NA values
data$bmi<- na_interpolation(data$bmi)
data$hypertension <- na_interpolation(data$hypertension)
```

```
              X             age             bmi        children          smoker        location 
              0               0              78               0               0               0 
  location_type education_level yearly_physical        exercise         married    hypertension 
              0               0               0               0               0              80 
         gender            cost 
              0               0 
[1] TRUE
```

Step 4: defining the expensive variable

- We sorted and considered if the cost variable is higher that the 75 percentile we considered it expensive using quantiles.

```
#Checking the quantile of cost to define the expensive variable
quantile(data$cost, probs = c(0.75))
data$expensive <- data$cost>4775
# replacing TRUE with 1 and FALSE with 0
data <- data %>% mutate( expensive = str_replace_all( string = expensive, pattern = "TRUE", "1"))
data <- data %>% mutate( expensive = str_replace_all( string = expensive, pattern = "FALSE", "0"))
head(data)
```

A tibble: 6 x 15

X <dbl>	age <dbl>	bmi <dbl>	children <dbl>	smoker <chr>	location <chr>	location_type <chr>	education_level <chr>	yearly_physical <chr>
1	18	27.900	0	yes	CONNECTICUT	Urban	Bachelor	No
2	19	33.770	1	no	RHODE ISLAND	Urban	Bachelor	No
3	27	33.000	3	no	MASSACHUSETTS	Urban	Master	No
4	34	22.705	0	no	PENNSYLVANIA	Country	Master	No
5	32	28.880	0	no	PENNSYLVANIA	Country	PhD	No
7	47	33.440	1	no	PENNSYLVANIA	Urban	Bachelor	No

6 rows | 1-9 of 15 columns

- Then we divided expensive and inexpensive people into 2 subsets and stored them into different variables

```{r}
#dividing expensive and inexpensive people into 2 subsets
expensivePeople <- subset(data,expensive=="TRUE")
inexpensivePeople <- subset(data,expensive=="FALSE")
head(expensivePeople)
head(inexpensivePeople)

smokerPeople <- subset(data,smoker=="yes")
head(smokerPeople)
```

tbl_df 6 x 15 | tbl_df 6 x 15 | tbl_df 6 x 15

A tibble: 6 × 15

X <dbl>	age <dbl>	bmi <dbl>	children <dbl>	smoker <chr>	location <chr>	location_type <chr>	education_level <chr>	yearly_physical <chr>
1	18	27.90	0	yes	CONNECTICUT	Urban	Bachelor	No
12	61	26.29	0	yes	CONNECTICUT	Urban	No College Degree	No
15	26	42.13	0	yes	PENNSYLVANIA	Urban	Bachelor	No
20	31	35.30	0	yes	PENNSYLVANIA	Urban	PhD	No
24	32	31.92	1	yes	NEW JERSEY	Urban	No College Degree	Yes
30	31	36.30	2	yes	PENNSYLVANIA	Urban	Bachelor	No

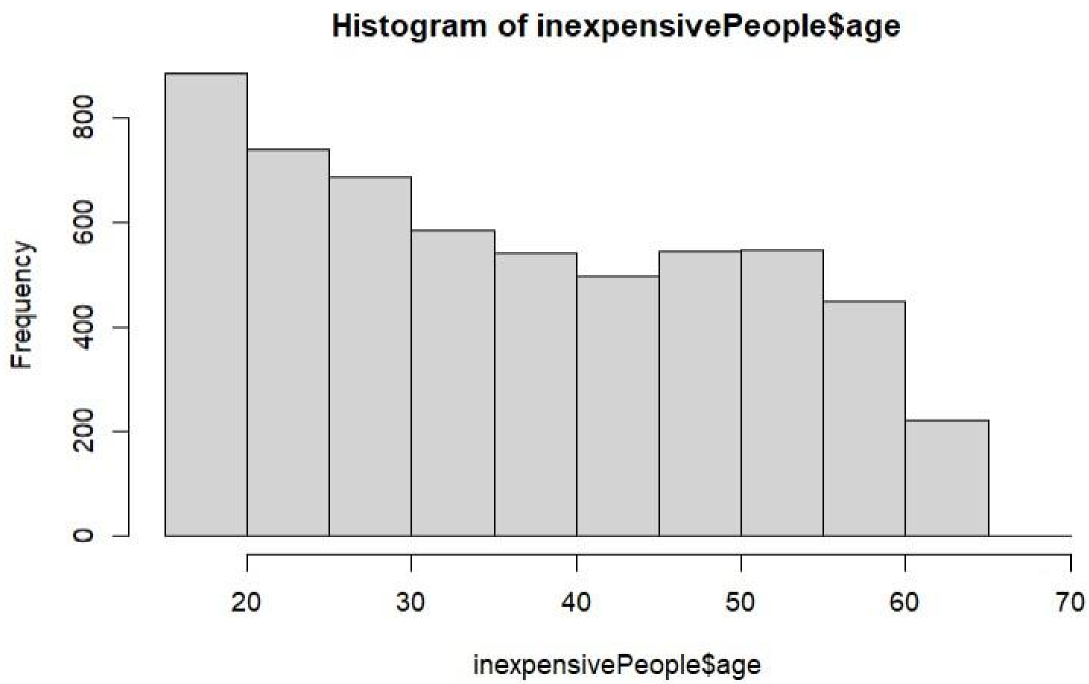
6 rows | 1-9 of 15 columns

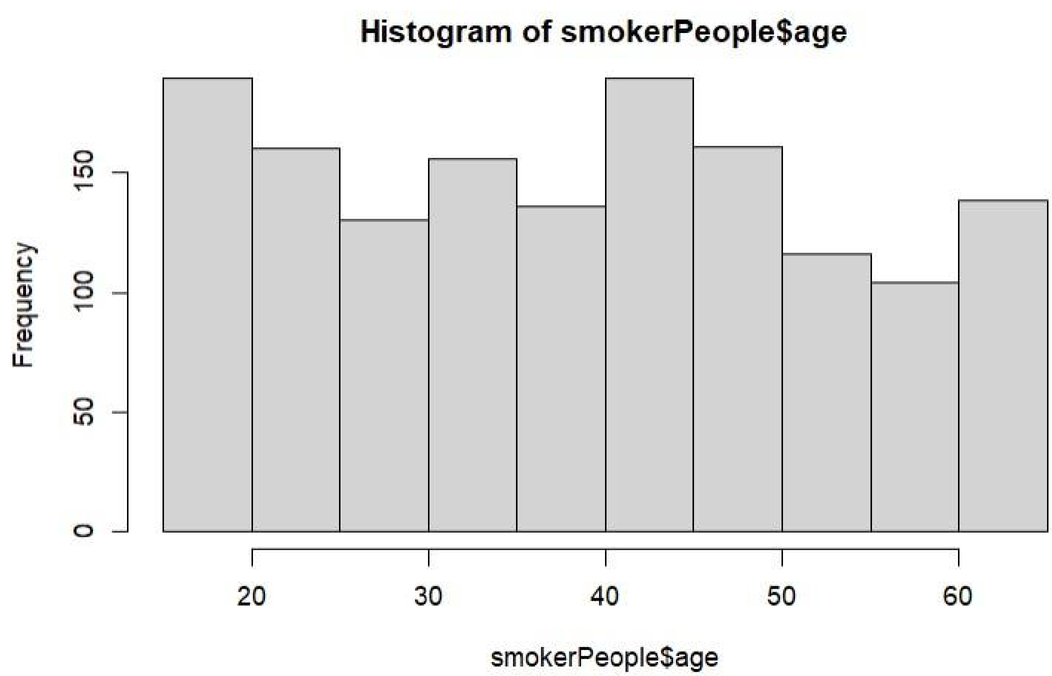
(4)<u>Descriptive statistics and Visualization</u>

- For Visualization, we used different variables like age, BMI, hypertension, and expense which can be providing the first broad meaningful information.

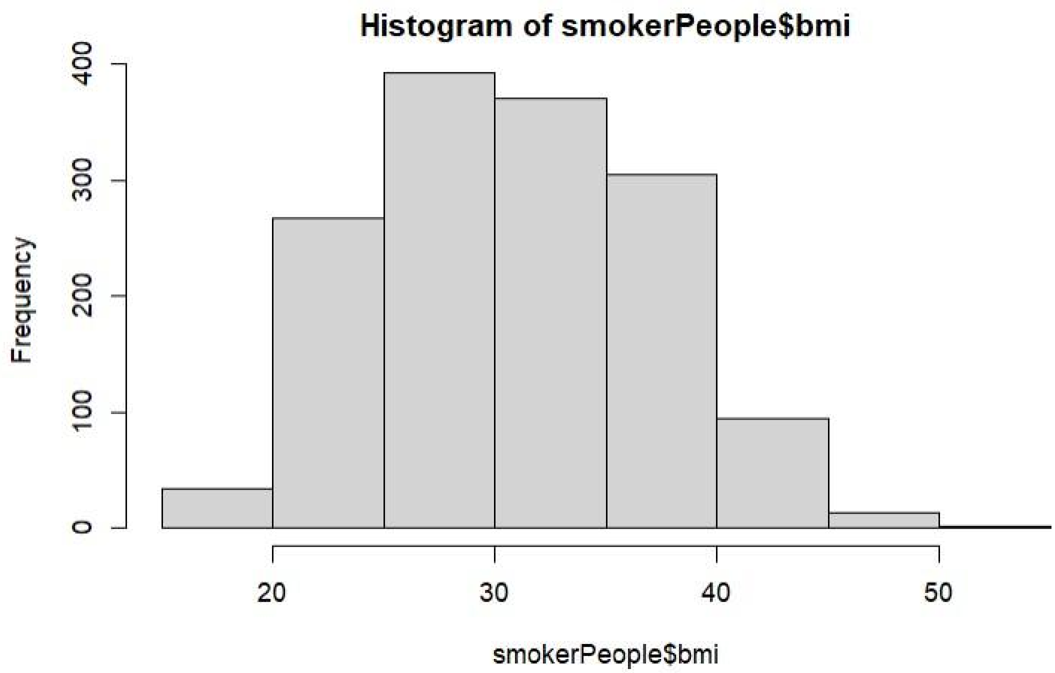
  #Visualizations: Histograms
  hist(expensivePeople$age)
  hist(inexpensivePeople$age)
  hist(smokerPeople$age)
  hist(smokerPeople$bmi)
  hist(as.numeric(smokerPeople$expensive))
  hist(as.numeric(smokerPeople$hypertension))
  hist(as.numeric(inexpensivePeople$hypertension))
- The histogram summarizes discrete or continuous data taken on an interval scale. It is frequently used to highlight the key characteristics of data distribution in a handy format.

- We found out that age people's age increases the expenses also increase which can be observed from the histograms below:

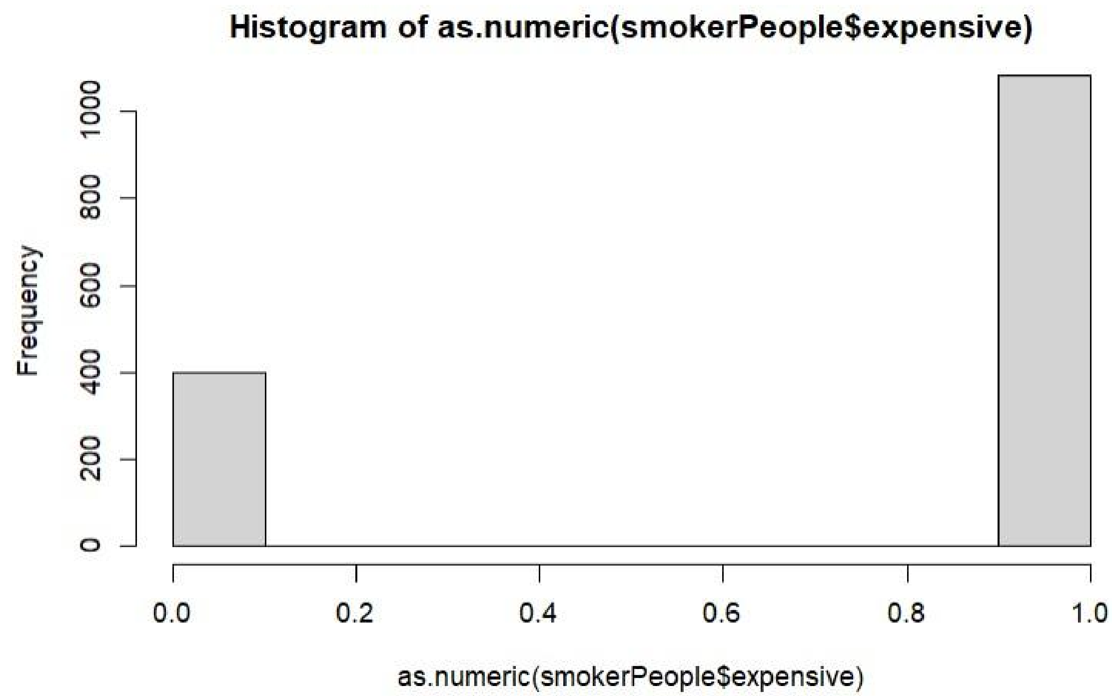

Histogram of inexpensivePeople$age
Frequency
0
200
400
600
800
20
30
40
50
60
70
inexpensivePeople$age

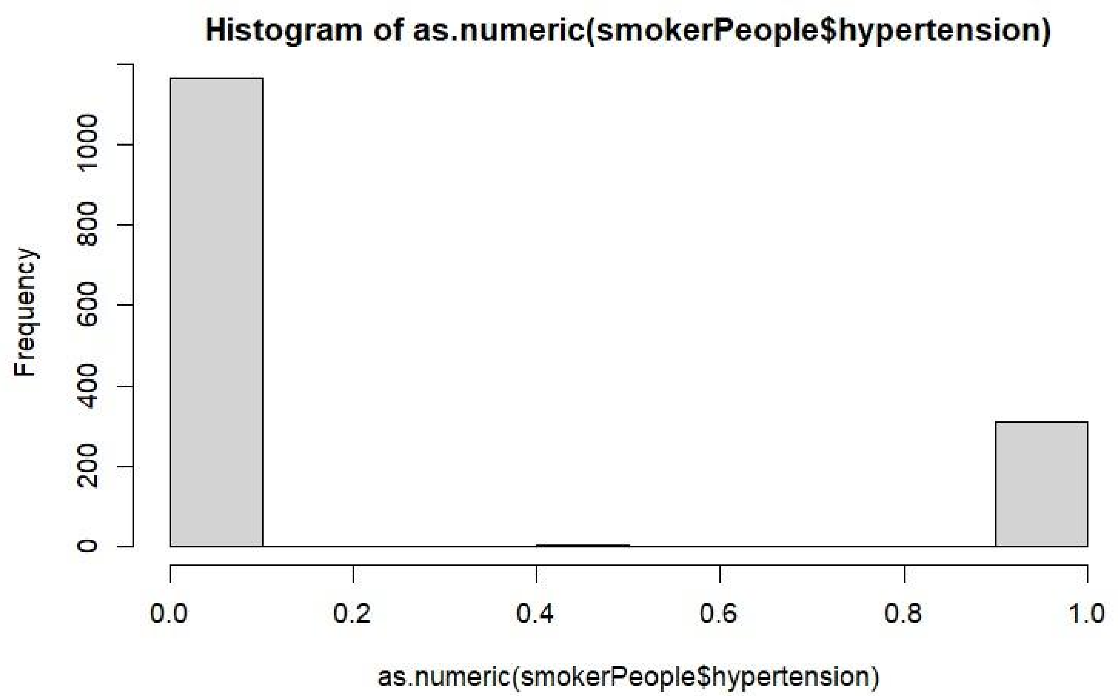

Histogram of smokerPeople$age
Frequency
0
50
100
150
20
30
40
50
60
smokerPeople$age

- From the histogram of the smoker's BMI with respect to their age we can observe below that most of the frequency of the people with the highest BMI is between 25 to 30



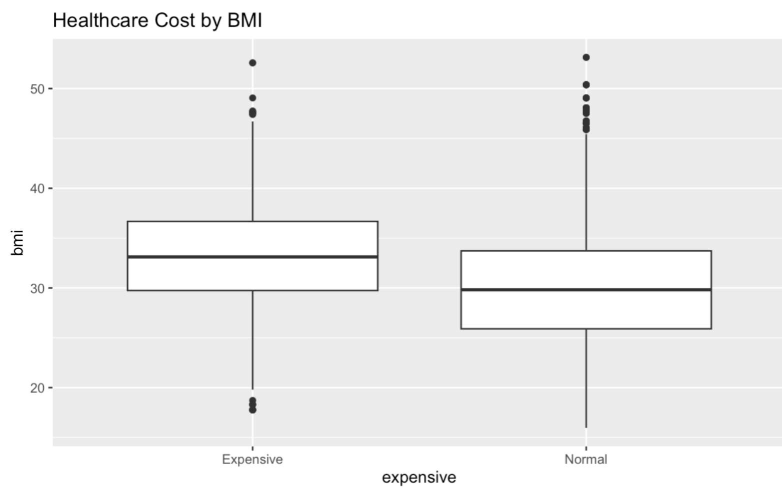
- From the below histogram we observed that people who smoke have a higher tendency to become an expensive patient as the frequency is 1000.

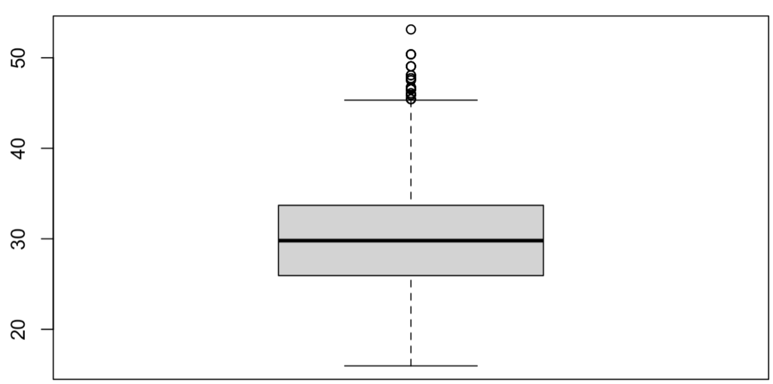

Histogram of as.numeric(smokerPeople$expensive)
Frequency
0
200
400
600
800
1000
0.0
0.2
0.4
0.6
0.8
1.0
as.numeric(smokerPeople$expensive)


Histogram of as.numeric(smokerPeople$hypertension)
Frequency
0
200
400
600
800
1000
0.0
0.2
0.4
0.6
0.8
1.0
as.numeric(smokerPeople$hypertension)

#Boxplot for cost, BMI, age, hypertension
boxplot(data$age)
boxplot(expensivePeople$age)
boxplot(inexpensivePeople$bmi)
boxplot(data$age)
boxplot(data$hypertension)

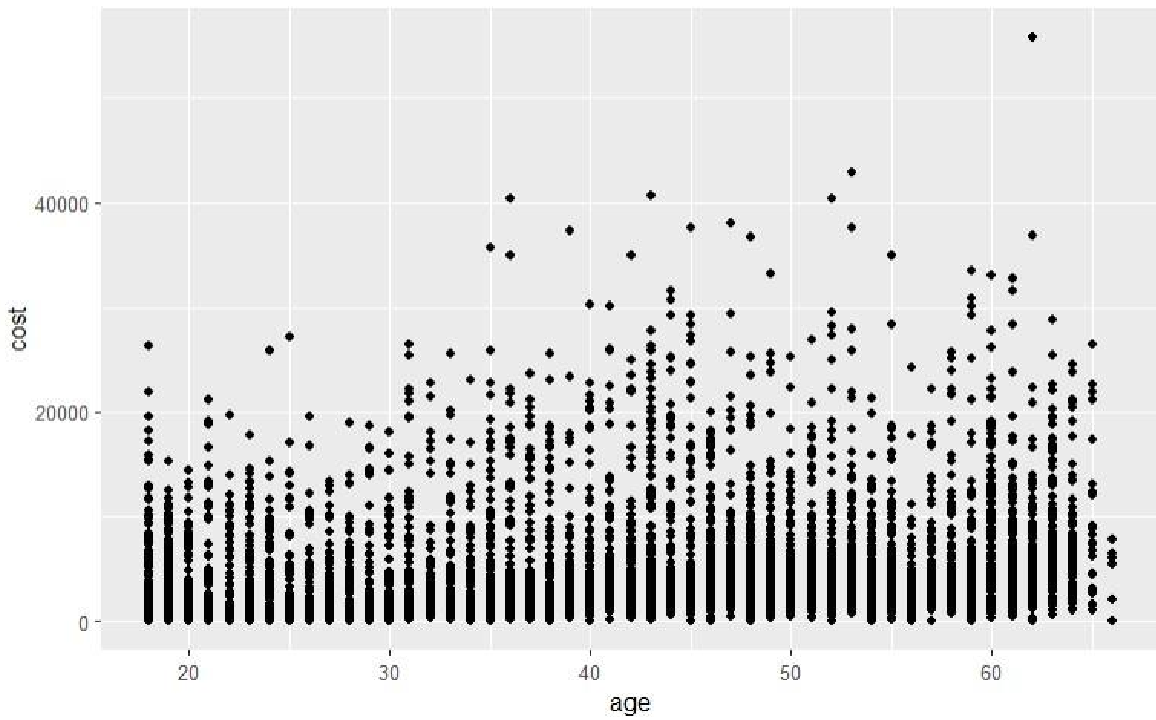
- From the below boxplot we observed that expensive people have an average age of 45 whereas normal people are around 35 years old.
- From the below box plot we observed that expensive people have BMI between 35 to 40. whereas normal people and inexpensive people have it between 30 to 33.



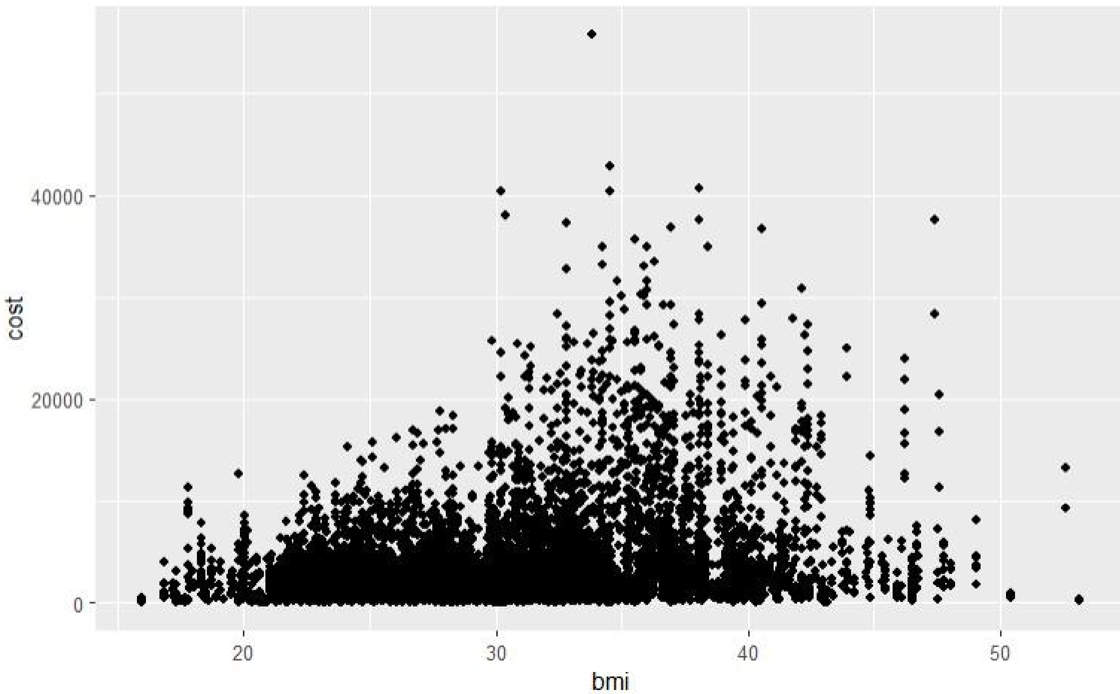


#Scatterplot for Age Vs Cost
agecost <- ggplot(data,aes(x=age, y=cost)) + geom_point()
agecost

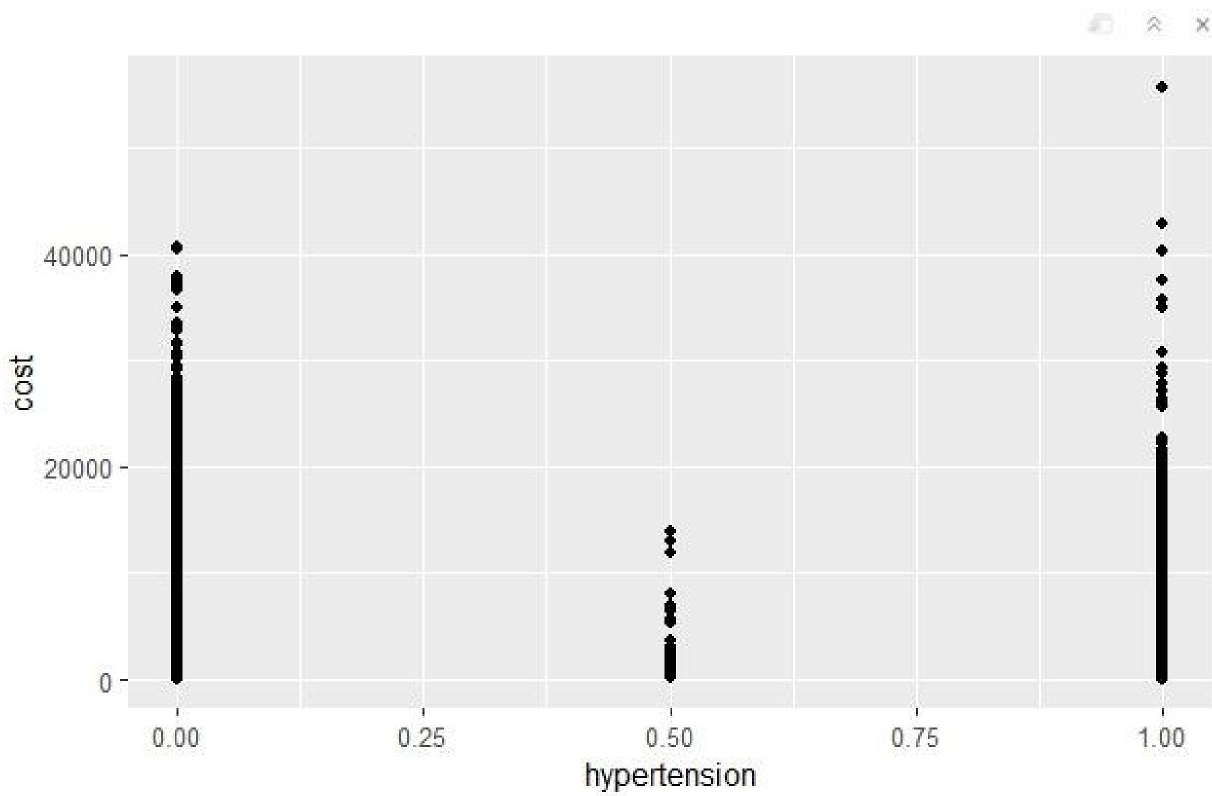
- From the below scatterplot for Age Vs Cost, we can observe that the scatterplot is populated highly from at 12000 USD for people around 65 years old.



#Scatterplot for Bmi Vs Cost
bmicost <- ggplot(data,aes(x=bmi, y=cost)) + geom_point()
Bmicost

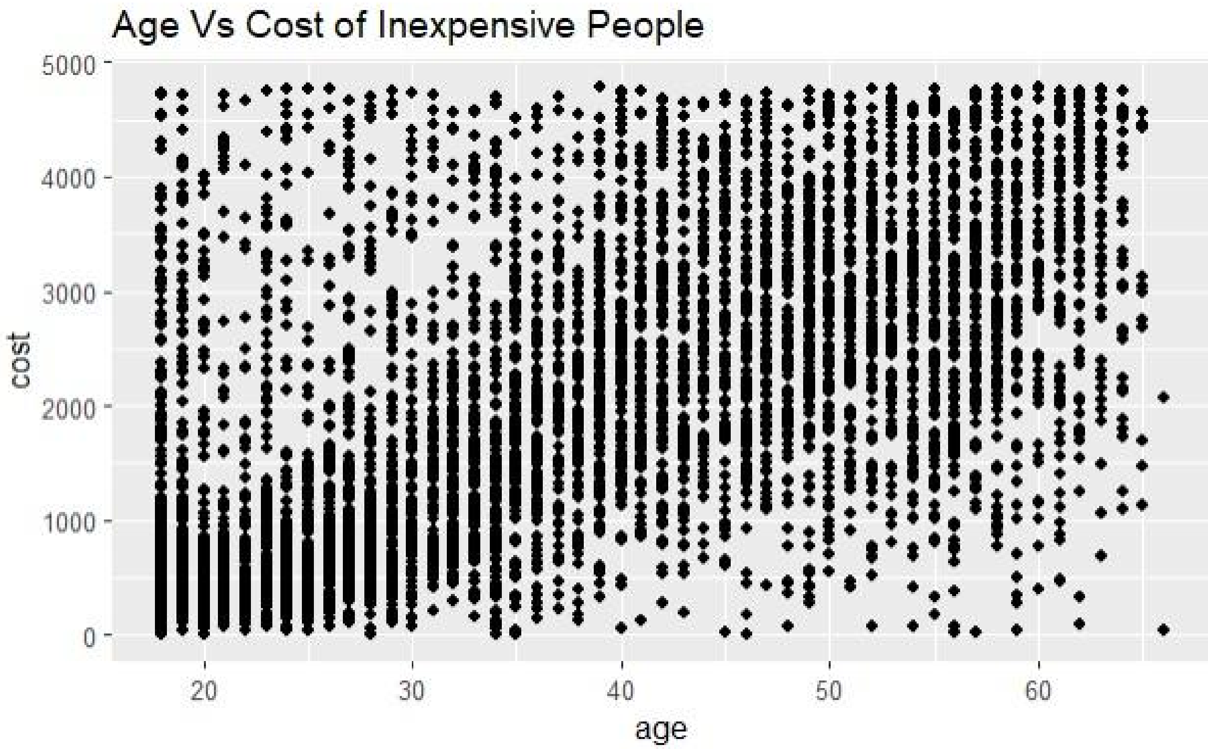


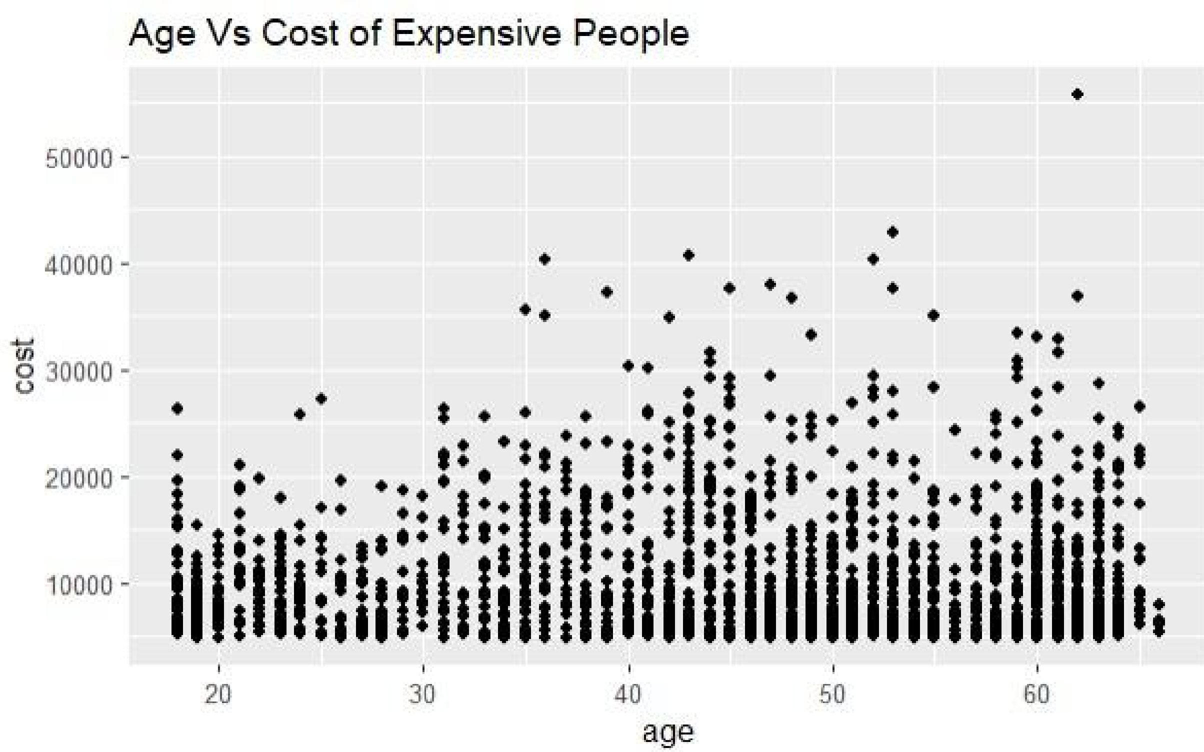
#Scatterplot for Hypertension Vs Cost
hypertensioncost <- ggplot(data,aes(x=hypertension, y=cost)) + geom_point()
hypertension cost



- here from the graph we can observe that hypertension can also be the factor of high-cost

```
#scatterplot for Age vs Expensive and Inexpensive people
ggplot(inexpensivePeople, aes(x=age,y=cost))+geom_point() + ggtitle("Age Vs Cost of Inexpensive People")
ggplot(expensivePeople, aes(x=age,y=cost))+geom_point() + ggtitle("Age Vs Cost of Expensive People")
```



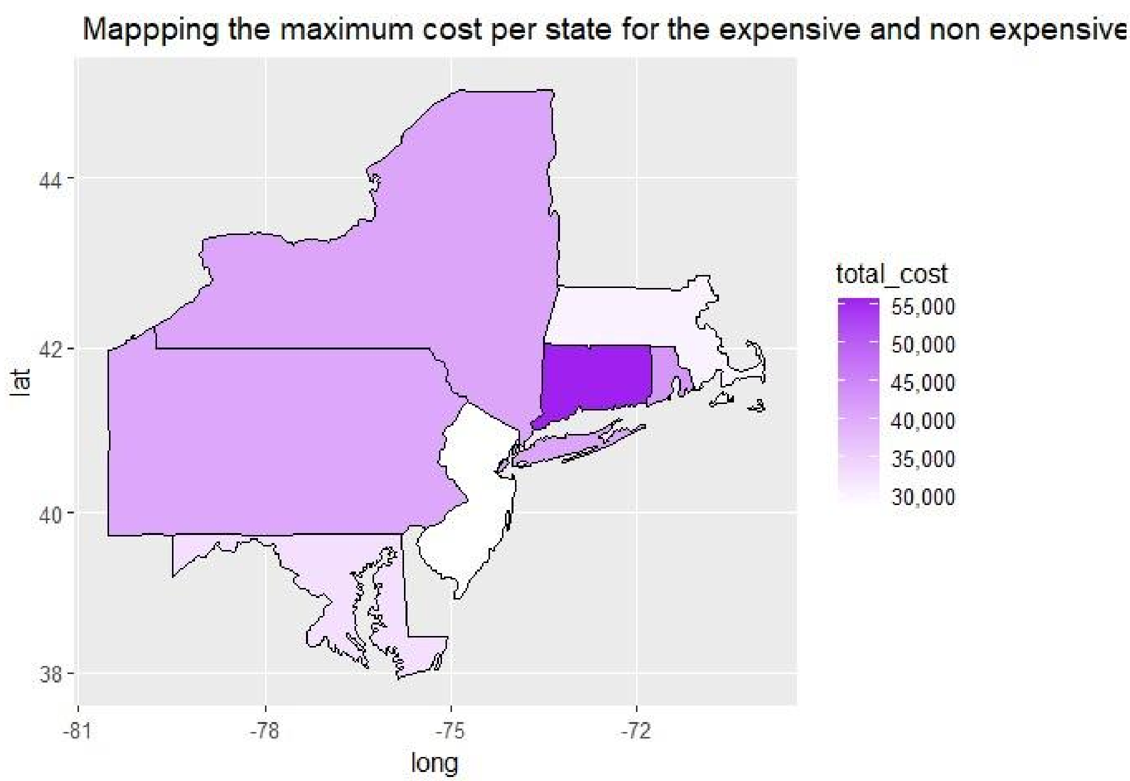



```
#Maps(Cost based on location)
dfAgg <- data %>% group_by(location) %>% summarise(total_cost = max(cost))
dfAgg$state <- tolower(dfAgg$location)
us <- map_data("state")
```

```
us$state <- us$region
mergedNew <- merge(dfAgg,us,on = "state")
mergedNew <- mergedNew[order(mergedNew$order),]
map <- ggplot(mergedNew) + geom_polygon(aes(x = long, y = lat, group = group,fill = total_cost), color =
"black")
map + scale_fill_continuous(low = "white", high = "purple", name = "total_cost", label = scales::comma) +
coord_map() +ggtitle("Mapping the maximum cost per state for the expensive and nonexpensive people")
```

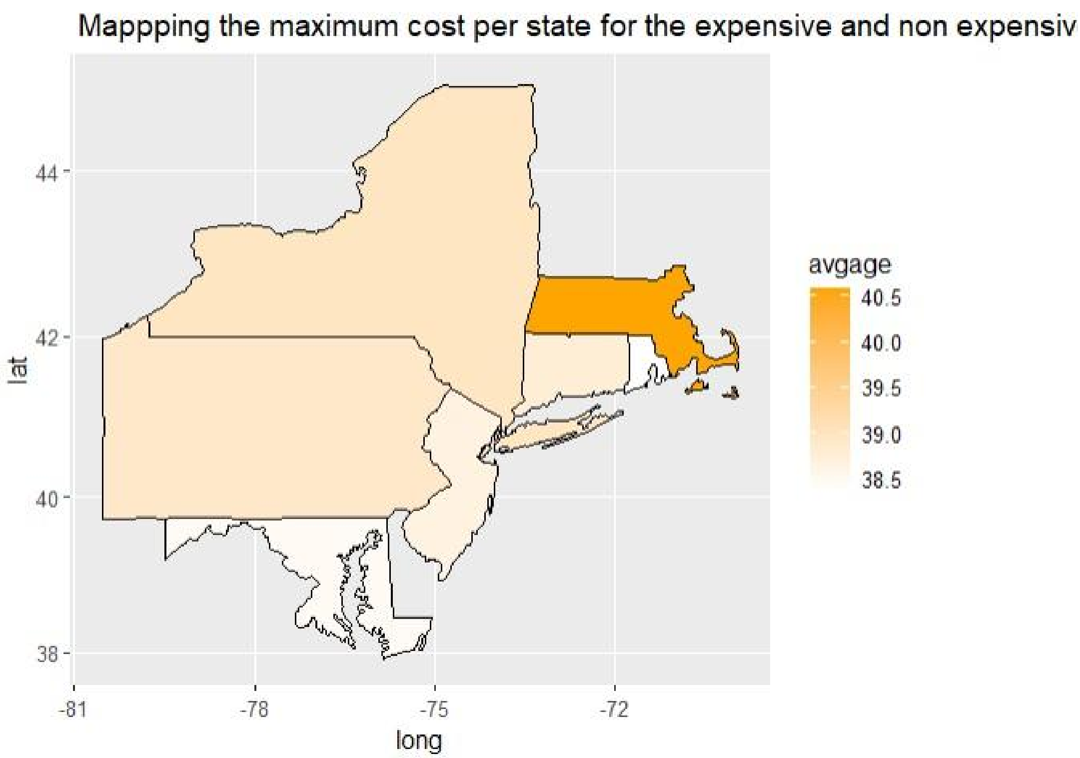
Geographic Findings

- As we can see three different illustrations of heat maps we found the most expensive state, most expensive sate by age and most expensive state based on the average number of the smokers in the state. So in the end we found out that the most expensive overall is Connecticut, by age its Massachusetts, and by the average number of smokers it is new york.




```
#Maps(Avg age based on location)
dfAgg <- data %>% group_by(location) %>% summarise(avgage = mean(age))
dfAgg$state <- tolower(dfAgg$location)
us <- map_data("state")
us$state <- us$region
mergedNew <- merge(dfAgg,us,on = "state")
mergedNew <- mergedNew[order(mergedNew$order),]
map <- ggplot(mergedNew) + geom_polygon(aes(x = long, y = lat, group = group,fill = avgage), color =
"black")
```

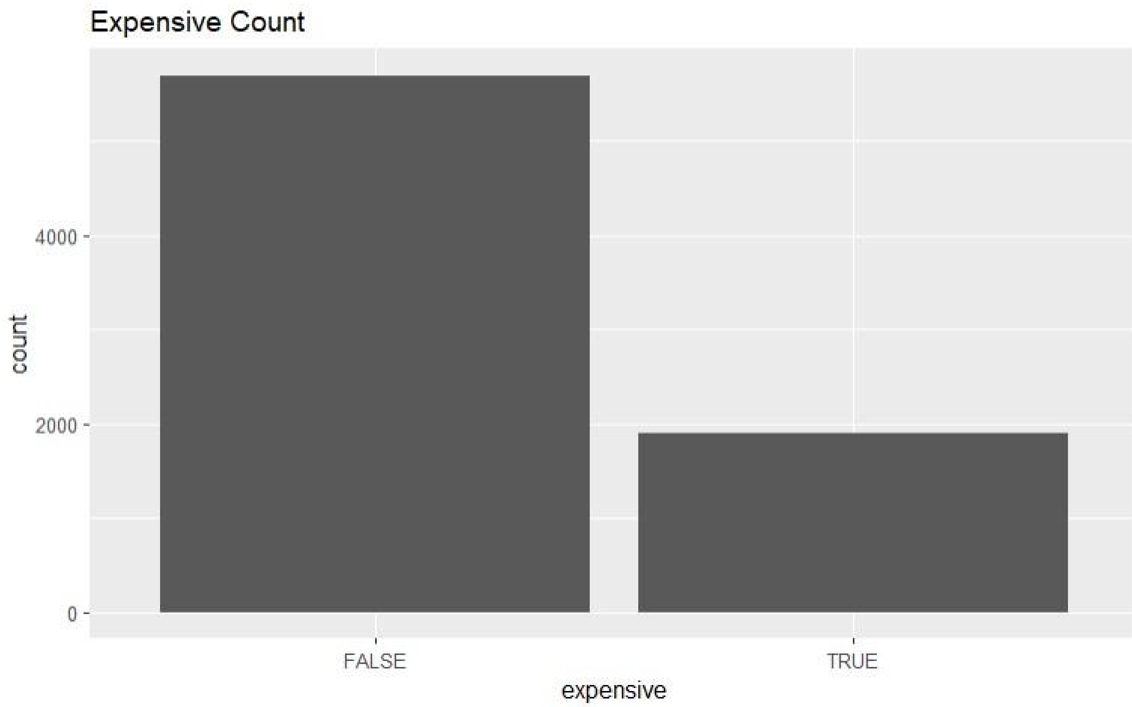
map + scale_fill_continuous(low = "white", high = "orange", name = "avgage", label = scales::comma) + coord_map() +ggtitle("Mapping the maximum cost per state for the expensive and nonexpensive people")




```{r}
#barplot for expensive count
expensivePlot <- ggplot(data,aes(x=expensive)) + geom_bar() + ggtitle("Expensive Count")
expensivePlot
```

```{r}
#Maps(Avg age based on location)
data <- data %>% mutate( smoker = str_replace_all( string = smoker, pattern = "yes", "1"))
data <- data %>% mutate( smoker = str_replace_all( string = smoker, pattern = "no", "0"))
data$smoker <- as.numeric(data$smoker)
dfAgg <- data %>% group_by(location) %>% summarise(avgsmokers = mean(smoker))
dfAgg$state <- tolower(dfAgg$location)
us <- map_data("state")
us$state <- us$region
mergedNew <- merge(dfAgg,us,on = "state")
mergedNew <- mergedNew[order(mergedNew$order),]
map <- ggplot(mergedNew) + geom_polygon(aes(x = long, y = lat, group = group,fill = avgsmokers), color = "black")
map + scale_fill_continuous(low = "white", high = "blue", name = "avgsmokers", label = scales::comma) + coord_map() +ggtitle(" Mapping the average smokers per state ")
```

- After all this Visualization we could find out that overall there are more inexpensive people than the expensive one in total.




```
#creating a new data frame
hmoData <- data.frame(age = data$age,
                      bmi = data$bmi,
                      smoker= data$smoker,
                      yearly_physical= data$yearly_physical,
                      exercise =data$exercise,
                      hypertension = data$hypertension,
                      expensive=as.factor(data$expensive))

# replacing TRUE with 1 and FALSE with 0
hmoData <- hmoData %>% mutate( expensive = str_replace_all( string = expensive, pattern = "TRUE", "1"))
hmoData <- hmoData %>% mutate( expensive = str_replace_all( string = expensive, pattern = "FALSE", "0"))
hmoData$expensive <- as.factor(hmoData$expensive)
str(hmoData)
```

- So, after observing the current situation and the variables with a high impact we used different models to predict future costs by using the prediction model. And for that, we divided the available data into train and test data sets.

```
'data.frame':   7582 obs. of  7 variables:
 $ age            : num  18 19 27 34 32 47 36 59 24 61 ...
 $ bmi            : num  27.9 33.8 33 22.7 28.9 ...
 $ smoker         : chr  "yes" "no" "no" "no" ...
 $ yearly_physical: chr  "No" "No" "No" "No" ...
 $ exercise       : chr  "Active" "Not-Active" "Active" "Not-Active" ...
 $ hypertension   : num  0 0 0 1 0 0 0 1 0 0 ...
 $ expensive      : Factor w/ 2 levels "0","1": 1 1 1 2 1 1 1 2 1 1 ...
```

(5)Use of modeling techniques & Visualizations

```
# Building SVM model
set.seed(123)
ksvm_model <- ksvm(data= trainSetS, expensive~.,C=5, CV=3, prob.model= TRUE)
svmPred<- predict(ksvm_model,newdata= testSetS, type= "response")
head(svmPred)
str(svmPred)

# Checking accuracy of ksvm model using confusion matrix
confusionMatrix(svmPred,as.factor(testSetS$expensive))
```

```
# Checking accuracy of ksvm model using confusion matrix
confusionMatrix(svmPred,as.factor(testSetS$expensive))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1099  145
##          1   38  234
##
##                Accuracy : 0.8793
##                  95% CI : (0.8618, 0.8953)
##     No Information Rate : 0.75
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6447
##
##  Mcnemar's Test P-Value : 4.661e-15
##
##             Sensitivity : 0.9666
##             Specificity : 0.6174
##          Pos Pred Value : 0.8834
##          Neg Pred Value : 0.8603
##              Prevalence : 0.7500
##          Detection Rate : 0.7249
##    Detection Prevalence : 0.8206
##       Balanced Accuracy : 0.7920
##
##        'Positive' Class : 0
##
```

- The Fist model we used was the SVM model and build the confusion matrix to see the TN, TP, FN, and FP observations as shown below. In this model, we found an accuracy of 87.93% and a sensitivity of 96.66%

- The second model we used was the tree model and did the same thing as SVM and we found it 86.87 % accurate which is less accurate than the SVM model but has higher sensitivity than SVM model i.e. has asensitivity of 98.24%.

```
# Building a tree model
rpart_model <- rpart(expensive ~ age+bmi+children+smoker+hypertension+exercise+yearly_physical, data = trainSet, method = "class")
rpartPred <- predict(rpart_model, newdata= testSet, type= "class")
# str(rpart_model)
# str(as.factor(testSet$expensive))
# head(rpartPred)
confusionMatrix(rpartPred, as.factor(testSet$expensive))
```
```

```r
# Building a tree model
rpart_model <- rpart(expensive ~ age+bmi+children+smoker+hypertension+exercise+yearly_physical, d
rpartPred <- predict(rpart_model, newdata= testSet, type= "class")
# str(rpart_model)
# str(as.factor(testSet$expensive))
# head(rpartPred)
confusionMatrix(rpartPred, as.factor(testSet$expensive))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1117  179
##      TRUE     20  200
```



```
##
##                Accuracy : 0.8687
##                  95% CI : (0.8507, 0.8853)
##     No Information Rate : 0.75
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.593
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9824
##             Specificity : 0.5277
##          Pos Pred Value : 0.8619
##          Neg Pred Value : 0.9091
##              Prevalence : 0.7500
##          Detection Rate : 0.7368
##    Detection Prevalence : 0.8549
##       Balanced Accuracy : 0.7551
##
##        'Positive' Class : FALSE
##
```
```

- Lastly we went ahead with the multiple variable linear models which gave us the second highest accuracy so we used the model with more sensitivity which is rpart means the tree model to check the frequency and build the shiny app to predict the future cost by observing the given dataset.

```
Call:
lm(formula = expensive ~ age + bmi + children + smoker + hypertension +
    exercise + yearly_physical, data = trainSet)

Residuals:
     Min       1Q   Median       3Q      Max
-0.94650 -0.20381 -0.05809  0.12000  1.13955

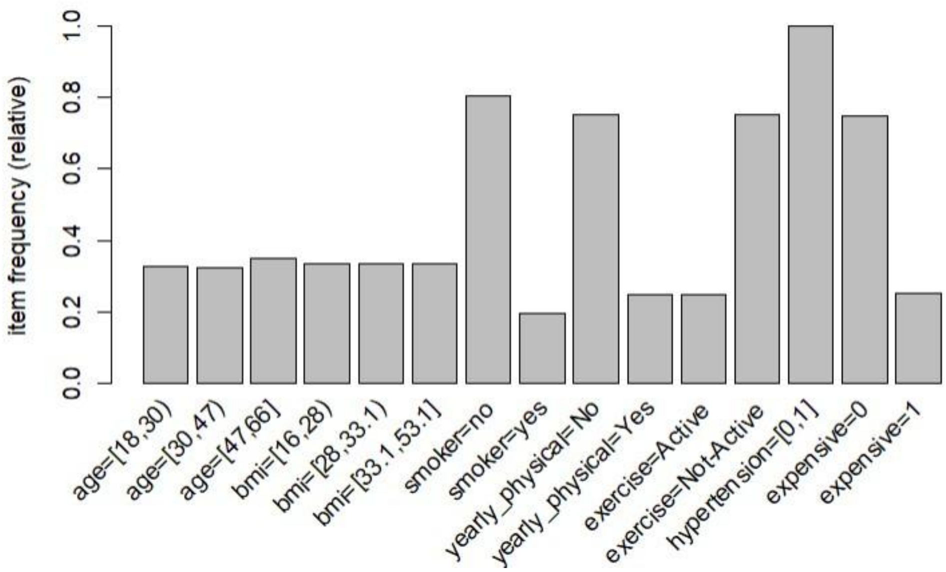
Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)        -0.6705137  0.0256627 -26.128  < 2e-16 ***
age                 0.0073023  0.0002982  24.487  < 2e-16 ***
bmi                 0.0121343  0.0007061  17.184  < 2e-16 ***
children            0.0105168  0.0034620   3.038  0.00239 **
smokeryes           0.6054765  0.0106940  56.618  < 2e-16 ***
hypertension        0.0303565  0.0104096   2.916  0.00356 **
exerciseNot-Active  0.1660275  0.0097583  17.014  < 2e-16 ***
yearly_physicalYes  0.0156726  0.0096845   1.618  0.10565
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3275 on 6058 degrees of freedom
Multiple R-squared:  0.4284,    Adjusted R-squared:  0.4278
F-statistic: 648.7 on 7 and 6058 DF,  p-value: < 2.2e-16
```

Apriori Algorithm: For association rule mining, the Apriori algorithm is used to find frequent item sets in a dataset. Apriori is named after the fact that it makes use of prior knowledge of common itemset properties. We have used an iterative approach or level-wise search to find k+1 itemsets using k-frequent itemsets. We have converted it into a sparse transaction matrix by defining rules.

```
# unsupervised : Apriori algorithm
# coverting to sparse transaction matrix
dataX <- hmoData
dataX<-as(dataX,'transactions')

itemFrequency(dataX)
itemFrequencyPlot(dataX)
```



```
# defining rules
ruleset <- apriori(dataX,
parameter=list(supp=0.040, conf=0.71),
control=list(verbose=F),
appearance=list(default="lhs",rhs=("expensive=1")))

summary(ruleset)


# parameter=list(supp=0.040, conf=0.9) 10 values
```


```
set of 28 rules

rule length distribution (lhs + rhs):sizes
 2  3  4  5  6 
 1  7 12  7  1 

   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
      2       3       4       4       5       6 

summary of quality measures:
    support          confidence        coverage            lift           count       
 Min.   :0.04181   Min.   :0.7169   Min.   :0.04181   Min.   :2.869   Min.   : 317.0  
 1st Qu.:0.04366   1st Qu.:0.7937   1st Qu.:0.05368   1st Qu.:3.176   1st Qu.: 331.0  
 Median :0.05421   Median :0.8558   Median :0.06047   Median :3.424   Median : 411.0  
 Mean   :0.06840   Mean   :0.8620   Mean   :0.08284   Mean   :3.449   Mean   : 518.6  
 3rd Qu.:0.08784   3rd Qu.:0.9538   3rd Qu.:0.10921   3rd Qu.:3.816   3rd Qu.: 666.0  
 Max.   :0.14244   Max.   :1.0000   Max.   :0.19507   Max.   :4.001   Max.   :1080.0  
```

inspect(ruleset)

```
     lhs                          rhs              support confidence   coverage     lift count
[1]  {smoker=yes}              => {expensive=1} 0.14244263  0.7302231 0.19506726 2.921663  1080
[2]  {age=[30,47),
      smoker=yes}              => {expensive=1} 0.05684516  0.7937385 0.07161699 3.175792   431
[3]  {bmi=[28,33.1),
      smoker=yes}              => {expensive=1} 0.04365603  0.7644342 0.05710894 3.058544   331
[4]  {bmi=[33.1,53.1],
      smoker=yes}              => {expensive=1} 0.06779214  0.9553903 0.07095753 3.822570   514
[5]  {age=[47,66],
      smoker=yes}              => {expensive=1} 0.05420733  0.8491736 0.06383540 3.397590   411
[6]  {smoker=yes,
      exercise=Not-Active}     => {expensive=1} 0.11685571  0.8158379 0.14323398 3.264213   886
[7]  {smoker=yes,
      yearly_physical=No}      => {expensive=1} 0.10656819  0.7169476 0.14864152 2.868547   808
[8]  {smoker=yes,
      hypertension=[0,1]}      => {expensive=1} 0.14244263  0.7302231 0.19506726 2.921663  1080
[9]  {age=[30,47),
      smoker=yes,
      exercise=Not-Active}     => {expensive=1} 0.04682142  0.8722359 0.05367977 3.489864   355
[10] {age=[30,47),
      smoker=yes,
      hypertension=[0,1]}      => {expensive=1} 0.05684516  0.7937385 0.07161699 3.175792   431
[11] {bmi=[28,33.1),
      smoker=yes,
      hypertension=[0,1]}      => {expensive=1} 0.04365603  0.7644342 0.05710894 3.058544   331
[12] {bmi=[33.1,53.1],
      smoker=yes,
      exercise=Not-Active}     => {expensive=1} 0.05420733  1.0000000 0.05420733 4.001055   411
[13] {bmi=[33.1,53.1],
      smoker=yes,
      yearly_physical=No}      => {expensive=1} 0.05209707  0.9495192 0.05486679 3.799079   395
```

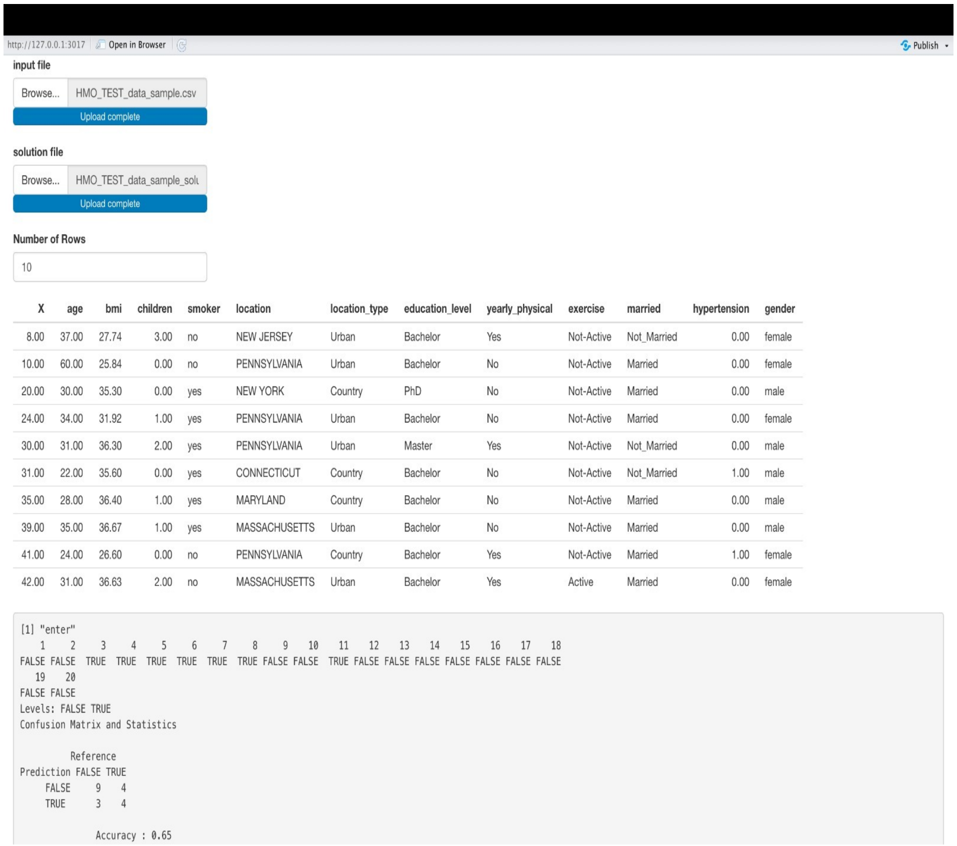
```{r}
# shinny app
best_model2 <- rpart_model
saveRDS(best_model2,file="/Users/maitreyiahire/Documents/DS Project/best_model2.rds")
readRDS(file="/Users/maitreyiahire/Documents/DS Project/best_model2.rds")
```

```
library(shiny)
library(caret)
library(kernlab)
library(e1071)
library(tidyverse)
ui <- fluidPage (
  #Obtain the data
  fileInput("upload", label="Insert input file", accept = c(".csv")),
  #Get the real data.
  fileInput("upload_Solution", label="Insert solution file", accept = c(".csv")),
  #Obtain a number
  numericInput("n", "Number of Rows", value = 10, min = 1, step = 1),

  tableOutput("headForDF"),

  verbatimTextOutput("txt_results", placeholder = TRUE)
)

server <- function(input, output, session) {

  use_model_to_predict <- function(df, df_solution){

    my_model <- readRDS("/Users/maitreyiahire/Documents/DS Project/best_model2.rds")

    print('enter')
    P <- predict(my_model, df, type = "class")

    print(P)

    confusionMatrix(P, as.factor(df_solution$expensive))
  }

  getTestData <- reactive({
    req(input$upload)
    read_csv(input$upload$name)
  })
```

```
  getSolutionData <- reactive({
    req(input$upload_Solution)
    read_csv(input$upload_Solution$name)
  })
  output$txt_results <- renderPrint({

    dataset <- getTestData()
    dataset_solution <- getSolutionData()

    use_model_to_predict(dataset, dataset_solution)
  })

  output$headForDF <- renderTable({
    df <- getTestData()
    head(df, input$n)
  })
}
shinyApp(ui, server)
```

- During the Shiny app process, we first read the CSV file and came up with the number and decided how much data frame to show then located a place to show the output.

- After that, we loaded the prediction model and computed a confusion matrix and saw how the data performed and tried with different data sets, and Finally showed the few lines of the data frame as you can see in the shiny app results below we reached 65 % accuracy in the different set of data.



X	age	bmi	children	smoker	location	location_type	education_level	yearly_physical	exercise	married	hypertension	gender
8.00	37.00	27.74	3.00	no	NEW JERSEY	Urban	Bachelor	Yes	Not-Active	Not_Married	0.00	female
10.00	60.00	25.84	0.00	no	PENNSYLVANIA	Urban	Bachelor	No	Not-Active	Married	0.00	female
20.00	30.00	35.30	0.00	yes	NEW YORK	Country	PhD	No	Not-Active	Married	0.00	male
24.00	34.00	31.92	1.00	yes	PENNSYLVANIA	Urban	Bachelor	No	Not-Active	Married	0.00	female
30.00	31.00	36.30	2.00	yes	PENNSYLVANIA	Urban	Master	Yes	Not-Active	Not_Married	0.00	male
31.00	22.00	35.60	0.00	yes	CONNECTICUT	Country	Bachelor	No	Not-Active	Not_Married	1.00	male
35.00	28.00	36.40	1.00	yes	MARYLAND	Country	Bachelor	No	Not-Active	Married	0.00	male
39.00	35.00	36.67	1.00	yes	MASSACHUSETTS	Urban	Bachelor	No	Not-Active	Married	0.00	male
41.00	24.00	26.60	0.00	no	PENNSYLVANIA	Country	Bachelor	Yes	Not-Active	Married	1.00	female
42.00	31.00	36.63	2.00	no	MASSACHUSETTS	Urban	Bachelor	Yes	Active	Married	0.00	female

```
[1] "enter"
    1     2     3     4     5     6     7     8     9    10    11    12    13    14    15    16    17    18 
FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE 
   19    20 
FALSE FALSE 
Levels: FALSE TRUE
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE     9    4
     TRUE      3    4
                                          
               Accuracy : 0.65
```

http://127.0.0.1:3017 Open in Browser Publish

20.00	30.00	35.30	0.00	yes	NEW YORK	Country	PhD	No	Not-Active	Married	0.00	male
24.00	34.00	31.92	1.00	yes	PENNSYLVANIA	Urban	Bachelor	No	Not-Active	Married	0.00	female
30.00	31.00	36.30	2.00	yes	PENNSYLVANIA	Urban	Master	Yes	Not-Active	Not_Married	0.00	male
31.00	22.00	35.60	0.00	yes	CONNECTICUT	Country	Bachelor	No	Not-Active	Not_Married	1.00	male
35.00	28.00	36.40	1.00	yes	MARYLAND	Country	Bachelor	No	Not-Active	Married	0.00	male
39.00	35.00	36.67	1.00	yes	MASSACHUSETTS	Urban	Bachelor	No	Not-Active	Married	0.00	male
41.00	24.00	26.60	0.00	no	PENNSYLVANIA	Country	Bachelor	Yes	Not-Active	Married	1.00	female
42.00	31.00	36.63	2.00	no	MASSACHUSETTS	Urban	Bachelor	Yes	Active	Married	0.00	female

```
[1] "enter"
    1     2     3     4     5     6     7     8     9    10    11    12    13    14    15    16    17    18 
FALSE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE 
   19    20 
FALSE FALSE 
Levels: FALSE TRUE
Confusion Matrix and Statistics

          Reference
Prediction FALSE TRUE
     FALSE     9    4
     TRUE      3    4
                                          
               Accuracy : 0.65            
                 95% CI : (0.4078, 0.8461)
    No Information Rate : 0.6             
    P-Value [Acc > NIR] : 0.4159          
                                          
                  Kappa : 0.2553          
                                          
 Mcnemar's Test P-Value : 1.0000          
                                          
            Sensitivity : 0.7500          
            Specificity : 0.5000          
         Pos Pred Value : 0.6923          
         Neg Pred Value : 0.5714          
             Prevalence : 0.6000          
         Detection Rate : 0.4500          
   Detection Prevalence : 0.6500          
      Balanced Accuracy : 0.6250          
                                          
       'Positive' Class : FALSE           
```

(6)Actionable Insights / Overall interpretation of results

- Based on our analyses, we recommend three things to the HMO, with an emphasis on those who smoke.
- Smokers should pay higher premiums. Retrospective of age, charging smokers a greater premium would help offset the high cost of healthcare because healthcare expenditures are significantly higher for smokers than non-smokers.
- The premium ought to be significantly higher if you smoke in New York. The only state where healthcare costs were significantly higher than in another state was New York.
- We don't need to charge more for someone with a high BMI if we already charge older folks with higher premiums due to their age. This is so because aging affects BMI more so than smoking does.