Technical Report

IST 687: Introduction to Data Science – Final Project

Healthcare Cost Analysis for a sample U.S. population

Nakul Rattan, Aarti Mehra, Himanshu Mangal, Nandita Pathardikar

1. Abstract

This project aims to analyze a given real-life data set including people who have paid their healthcare expense. This study investigates the data to decide whether the cost paid by a person is expensive or cheap. With this project, we are trying to determine the factors that affect the difference in costs for different individuals, and why some people pay more than others. It uses various data analysis techniques, including modeling and visualizations. We have also used correlation to find what are the factors that affect the cost of healthcare strongly. We conclude that people falling in some specific age categories be suggested to quit smoking in order to lower their healthcare expense.

1.1 Project Goals

Using various data modeling and visualization technique, we will try to answer the listed questions about our data:

- Predict people who will spend a lot of money on health care next year (i.e., which people will have high healthcare costs).
- Provide actionable insight to the HMO, in terms of how to lower their total health care costs, by providing a specific recommendation on how to lower health care costs.

2. Introduction

We have a data set that contains details of 7,582 distinct individuals, who at some points have availed and paid for a healthcare service. In this section, we summarize what we explored about the data.

2.1 Dataset

The available data set has a record of 7,582 individuals based on 14 different parameters. In the latter part of our discussion, we have analyzed this data in various combinations of parameter consideration to better understand the affect that these parameters have on the degree of expense for people.

2.2 Dataset Variables

Listed below are the parameters / variables for each data entry point:

- **X**: Integer, Unique identified for each person
- **age**: Integer, The age of the person (at the end of the year).
- **location**: Categorical, the name of the state (in the United States) where the person lived (at the end of the year)
- **location_type**: Categorical, a description of the environment where the person lived (urban or country).
- **exercise**: Categorical, “Not-Active” if the person did not exercise regularly during the year, “Active” if the person did exercise regularly during the year.
- **smoker**: Categorical, “yes” if the person smoked during the past year, “no” if the person didn’t smoke during the year.
- **bmi**: Integer, the body mass index of the person. The body mass index (BMI) is a measure that uses your height and weight to work out if your weight is healthy.
- **yearly_physical**: Categorical, “yes” if the person had a well visit (yearly physical) with their doctor during the year. “no” if the person did not have a well visit with their doctor.
- **Hypertension**: “0” if the person did not have hypertension.

- **gender**: Categorical, the gender of the person
- **education_level**: Categorical, the amount of college education ("No College Degree", "Bachelor", "Master", "PhD")
- **married**: Categorical, describing if the person is “Married” or “Not_Married”
- **num_children**: Integer, Number of children
- **cost**: Integer, the total cost of health care for that person, during the past year.

3. Data Summarization

In this section, we describe methods that we followed for loading and manipulating the data, as well as justify new variable(s) that we created as we explored and aggregated the data. With every step, there is also a snippet of the code included in this section.

3.1 Data Loading

In the initial exploration of data, we loaded the data from a given csv file into R objects. This allowed us to view individual records and locate interesting observations and attributes in the data. The **tidyverse** library used here helps us to swiftly convert between different data objects. As in our case, we have converted a csv format to R objects.

```
```{r}
library(tidyverse)

datafile <- "https://intro-datascience.s3.us-east-2.amazonaws.com/HMO_data.csv"
data <- read_csv(datafile)

```
```

3.2 Data Cleaning

In order to work with the data to yield accurate results, we have cleaned our data off all the null values. To do this, first we check for null values in all our available variables in the R dataframe.

```
# Checking for null values.

```{r}
sum(is.na(data$X))
sum(is.na(data$age))
sum(is.na(data$bmi))
sum(is.na(data$children))
sum(is.na(data$smoker))
sum(is.na(data$location))
sum(is.na(data$location_type))
sum(is.na(data$education_level))
sum(is.na(data$yearly_physical))
sum(is.na(data$exercise))
sum(is.na(data$married))
sum(is.na(data$hypertension))
sum(is.na(data$gender))
sum(is.na(data$cost))
```
```

Upon checking for null values, we observed that 2 of the given variables in the data frame had some null values. To eliminate the risk of inaccuracy in our results, we have then used interpolation to create new estimated data points between know data points. We basically replace the null values in our data with an estimated new value. To be able to do this, we have used the **imputeTS** library. This provides us with functions that help with handling the missing values in our data by the method of imputation using various algorithms and tools.

```
46
47 # Interpolate the null values.
48
49 ```{r}
50 library(imputeTS)
51 data$bmi <- na_interpolation(data$bmi)
52 data$hypertension <- na_interpolation(data$hypertension)
53 ```
```

3.3 Data Creation

To determine whether the cost of healthcare is expensive or not for a person, we have created a new variable in our data.

- **expensive**: Categorical Boolean, TRUE for expensive, FALSE for not expensive. This value is calculated based on the median of all the cost. In our data, anything more than this value is an expensive cost, otherwise, it is not.

```
55 # Adding the *expensive* column in the dataframe.
56
57 ```{r}
58 data = data %>%
59   mutate(expensive = if_else(cost<=median(cost)*2,FALSE, TRUE))
60
61 # Define empty columns
62 empty_cols <- c('ageCategory')
63
64 # Add empty columns
65 data[ , empty_cols] <- NA
66
67 data$ageCategory <- as.factor(ifelse(data$age<20, 'children',
68                       ifelse(data$age>=20 & data$age<=34, 'young_adults',
69                       ifelse(data$age>=35 & data$age<=54, 'middle_aged-adults', 'older_adults'))))
70 summary(data)
71 ```
```

median(x, na.rm = FALSE, ...)

4. Data Modeling

In this section, we discuss our modeling strategies that we implemented and applied on our data set. These strategies include linear regression, data visualization using different techniques, correlation and predictive models.

4.1 Regression Analysis

Regression Analysis is a powerful predictive analysis model for statistical operations that allows us to examine the relationship between two or more variables in a data set. There are quite a few regression analysis methods that are used, but the underlining use of these methods include the study of the influence of one or more independent variables on one dependent variable. In our data set, we try to examine the influence of all the available variables on the variable **expensive.**

```
78 # We can now run a linear regression model on the dataframe. This would also help us identify significant predictors.
79
80 ```{r}
81 model <- lm(expensive~.-X-cost,data=data)
82 summary(model)
83 ```
```

From a statistical point of view, the best way to determine whether a variable is a good predictor of the dependent variable, is to check for its p-value in the regression model. The p-value of a variable allows us to decide whether a predictor is statistically significant or not.

In our linear regression model, we observe that p-values for the variables **age, bmi, children, smokeryes, locationNEWYORK, yearly_physicalYes, exerciseNot-Active and hypertension** have a value less than 0.05. Whenever the value of p-value for a variable in a regression model is less than 0.05, it is considered a good predictor of the dependent variable. Therefore, all the variables listed above will help us predict whether the cost of healthcare was expensive or not, in a much better way than the variables that are not good predictors, i.e., the variables that have a p-value greater than 0.05.

Another important observation in our regression model deals with the value of adjusted R-squared. For our model, this value is **0.4305.** This means that 43.13% change in the expenditure on healthcare will be explained by the changes in the predictors of our dependent variables. Simply said, 43.05% of times, we can predict how expensive the

healthcare is going to be for the people based on the changes in all the predictors in our model.

```
Call:
lm(formula = expensive ~ . - X - cost, data = data)

Residuals:
     Min       1Q   Median       3Q      Max
-0.93379 -0.19133 -0.05534  0.12191  1.13975

Coefficients:
                                Estimate Std. Error t value Pr(>|t|)
(Intercept)                    -0.6894907  0.0412408 -16.719  < 2e-16 ***
age                             0.0067340  0.0007288   9.239  < 2e-16 ***
bmi                             0.0125321  0.0006217  20.157  < 2e-16 ***
children                        0.0127107  0.0031522   4.032 5.58e-05 ***
smokeryes                       0.5960293  0.0093539  63.720  < 2e-16 ***
locationMARYLAND               -0.0043513  0.0175377  -0.248   0.8041
locationMASSACHUSETTS          -0.0097604  0.0197890  -0.493   0.6219
locationNEW JERSEY              0.0104758  0.0194075   0.540   0.5894
locationNEW YORK                0.0372544  0.0189316   1.968   0.0491 *
locationPENNSYLVANIA            0.0001304  0.0139646   0.009   0.9926
locationRHODE ISLAND           -0.0093912  0.0177803  -0.528   0.5974
location_typeUrban             -0.0055211  0.0085249  -0.648   0.5172
education_levelMaster          -0.0023634  0.0094910  -0.249   0.8034
education_levelNo College Degree  0.0116034  0.0125992   0.921   0.3571
education_levelPhD             -0.0080916  0.0129593  -0.624   0.5324
yearly_physicalYes              0.0196374  0.0085500   2.297   0.0217 *
exerciseNot-Active              0.1614532  0.0085426  18.900  < 2e-16 ***
marriedNot_Married              0.0073042  0.0078408   0.932   0.3516
hypertension                    0.0381210  0.0092524   4.120 3.83e-05 ***
gendermale                      0.0109769  0.0074362   1.476   0.1399
ageCategoryolder_adults         0.0286707  0.0152431   1.881   0.0600 .
ageCategoryyoung_adults         0.0111014  0.0168535   0.659   0.5101
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

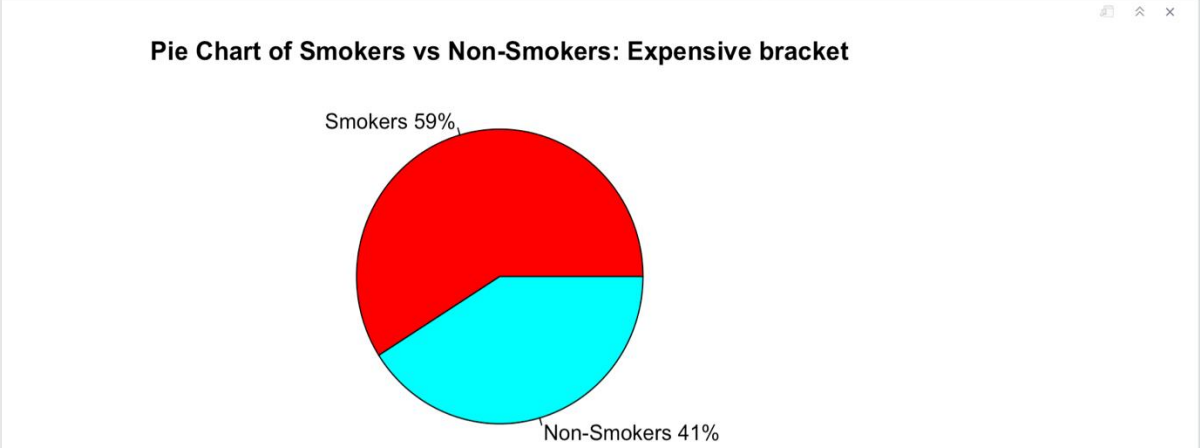
Residual standard error: 0.3212 on 7560 degrees of freedom
Multiple R-squared:  0.4321,    Adjusted R-squared:  0.4305
F-statistic: 273.9 on 21 and 7560 DF,  p-value: < 2.2e-16
```

4.2 Data Visualization

In this part of the section, we will look at some visualization techniques to better understand the patterns and trends in our data. We will be using different visualizations based on different aspects of our data. Since, smoking is a very good predictor (concluded in 4.1 regression analysis) of the expense, we have divided our data based on this factor. To do this, we have first created two groups – expensive and not expensive.

```
84  # Dividing the group into expensive and not expensive
85
86  ```{r}
87  dfExpensive <- data %>% filter(expensive==TRUE)
88  #dfExpensive
89
90  dfNotExpensive <- data %>% filter(expensive==FALSE)
91  #dfNotExpensive
92  ```
```

4.2.1 Technique 1: Pie Chart

To verify our reasoning, we have calculated the total percentage of smokers who fall in the expensive group. By doing this, we see that 58% of people who have an expensive cost to pay for their healthcare are smokers.

```
# Checking the percentage of smokers in the expensive group

```{r}
smokersExpensive <- sum(if_else(dfExpensive$smoker=="yes",1,0))
#1063 smokers out of 1792 people in the expensive set
ratioSmokersExpensive <- smokersExpensive/nrow(dfExpensive)

```

Now, we try to visualize the conclusion from our last step by using a pie chart. This shows the total percent of smokers and non-smokers that fall under the expensive bracket of the healthcare cost.

```
# Visualizing the above data in a pie chart

```{r}
slices <- c(ratioSmokersExpensive*100, (1-ratioSmokersExpensive)*100)
lbls <- c("Smokers","Non-Smokers")
pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie(slices,labels = lbls, col=rainbow(length(lbls)),
    main="Pie Chart of Smokers vs Non-Smokers: Expensive bracket")

```
```



Similarly, we also have visualized the percent of smokers and non-smokers that fall under the not expensive category. By doing so, we have observed that 93% of the people who
```

have a low expense on healthcare are non-smokers, and only 7% of such people are smokers.

```r
# Checking the percentage of smokers in the not expensive group

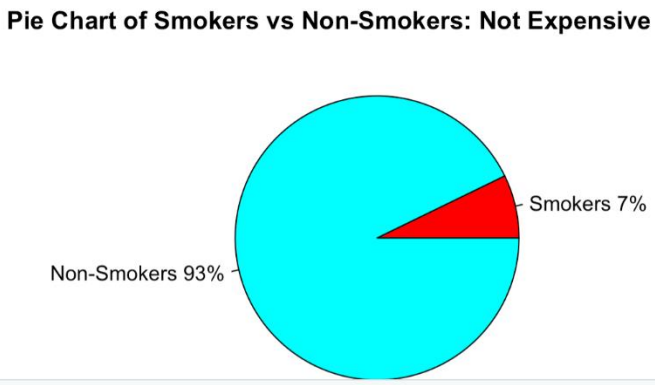
```{r}
smokersNotExpensive <- sum(if_else(dfNotExpensive$smoker=="yes",1,0))
ratioSmokersNotExpensive <- smokersNotExpensive/nrow(dfNotExpensive)

#ratioSmokersNotExpensive

slices <- c(ratioSmokersNotExpensive*100, (1-ratioSmokersNotExpensive)*100)
lbls <- c("Smokers","Non-Smokers")
pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie(slices,labels = lbls, col=rainbow(length(lbls)),
    main="Pie Chart of Smokers vs Non-Smokers: Not Expensive bracket")

# 7% of people in the not expensive group are smokers.

```
```



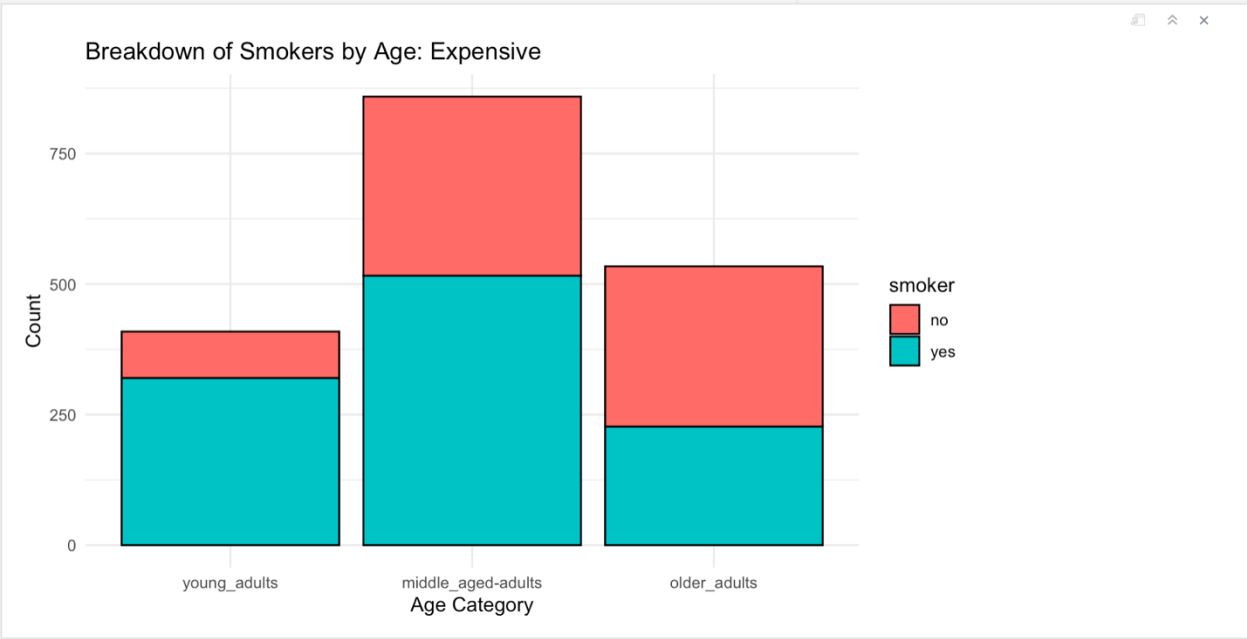
4.2.2 Technique 2: Stacked bar graph

To explore and understand the data set in a much better way, next we have used stacked bar graph as our visualization technique to segregate the data based on age groups and smoking activities. We have divided the people in three age groups here – young_adults, middle_aged-adults and older adults. To check the ratio of count of people un these age groups and whether they smoke or not based on their expenses, we have first created a bar graph that represents the data related to people who fall under the expensive bracket.

```r
```{r}
library(ggplot2)
smokersByAge <- dfExpensive %>% select(smoker,ageCategory) %>% group_by(ageCategory,smoker)
%>%summarise(total_count=n(),.groups = 'drop')

ggplot(smokersByAge,aes(x=ageCategory,y=total_count,fill=smoker)) + geom_bar(stat="identity",color="black") + theme_minimal()
+ xlab("Age Category") + ylab("Count") + ggtitle("Breakdown of Smokers by Age: Expensive") +
scale_x_discrete(limits=c("young_adults", "middle_aged-adults", "older_adults"))
```
```

In this case, we observed that more than 70% of young adults who pay high expense of healthcare are smokers, and for middle aged adults that ratio is a little over 50%. Whereas, for older adults, less than 50% smoke.



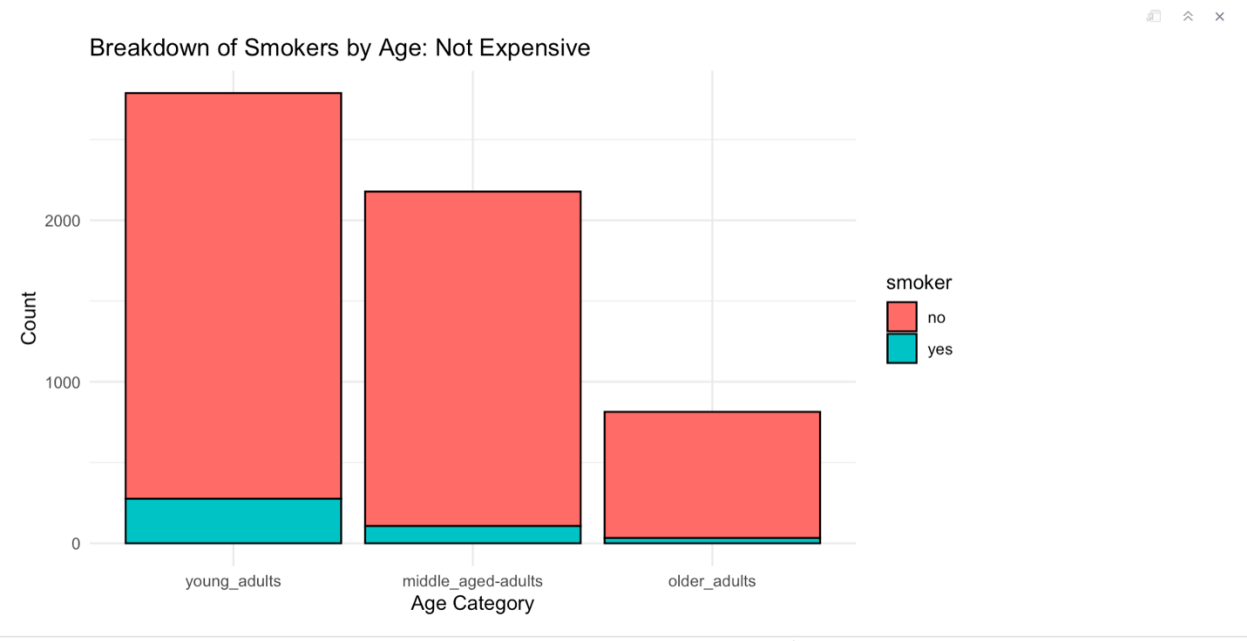
Up next, we checked the same for the non-expensive bracket.

```
smokersByAge <- dfNotExpensive %>% select(smoker,ageCategory) %>% group_by(ageCategory,smoker)
%>%summarise(total_count=n(),.groups = 'drop')

ggplot(smokersByAge,aes(x=ageCategory,y=total_count,fill=smoker)) + geom_bar(stat="identity",color="black") + theme_minimal()
+ xlab("Age Category") + ylab("Count") + ggtitle("Breakdown of Smokers by Age: Not Expensive") +
scale_x_discrete(limits=c("young_adults", "middle_aged-adults", "older_adults"))

```

Here, we observed that in all three categories the ration of people who smoke is very, very less than the ones who do.



4.2.3 Technique 3: Maps

We wanted to visualize our data in a form of map so that we can understand the demographics of the people that are in our survey. To do so, we used following libraries that enable us to plot out data on a US map by state – **maps, ggmap and mapproj**.

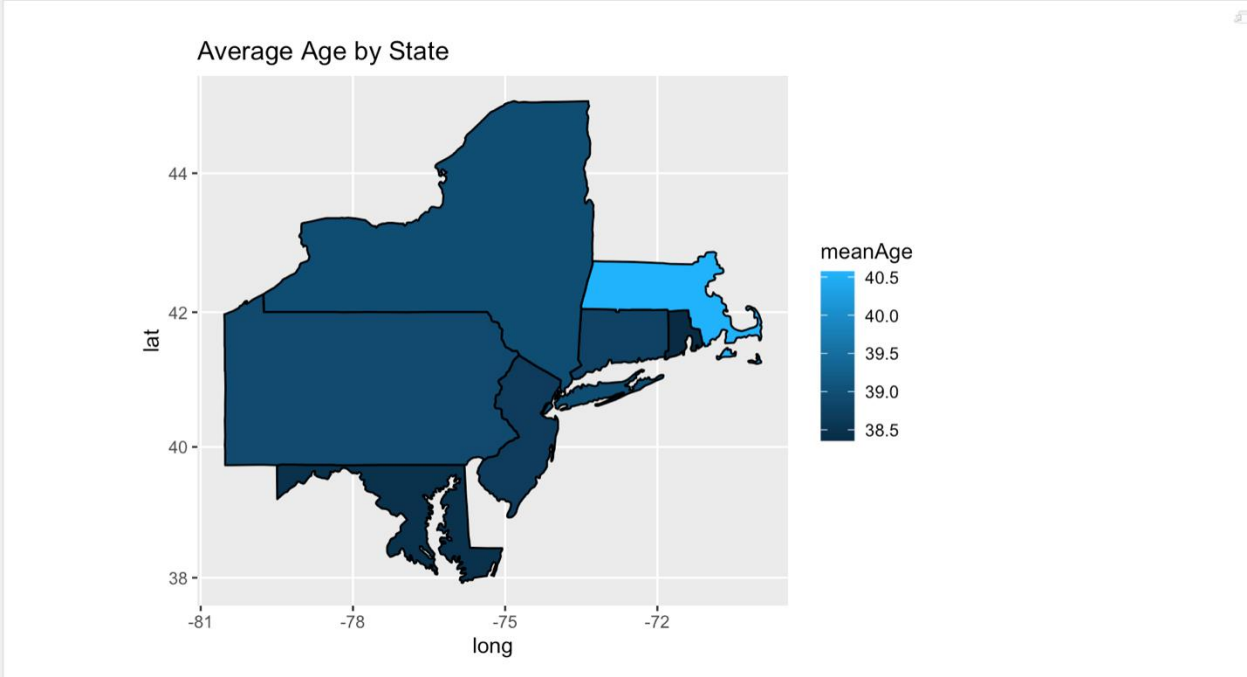
```r
```{r}
#unique(data$location)
us <- map_data("state")
data$location <- tolower(data$location)

dfSimple <- data %>% group_by(location) %>% summarise(meanAge=mean(age))

dfMerged <- merge(dfSimple, us, by.x="location",by.y = "region")
dfMerged <- dfMerged %>% arrange(order)
map <- ggplot(dfMerged)
map <- map + aes(x=long, y=lat, group=group,label=location,fill=meanAge) +  geom_polygon(color="black")
map <- map + expand_limits(x=dfMerged$long, y=dfMerged$lat)
map <- map + coord_map() + ggtitle("Average Age by State")
map

```
```

We have plotted our map based on the average age of the people from the same state. By doing so, we understood that the state of Massachusetts (upper right corner) has the highest average age of people, suggesting that possibly most of the older adults come from MA. While, Maryland and Rhode Island seem to have the lowest average rates, suggesting that possibly most of the middle aged and young adults belong to these two states.



4.2.4 Technique 4: Violin Plot

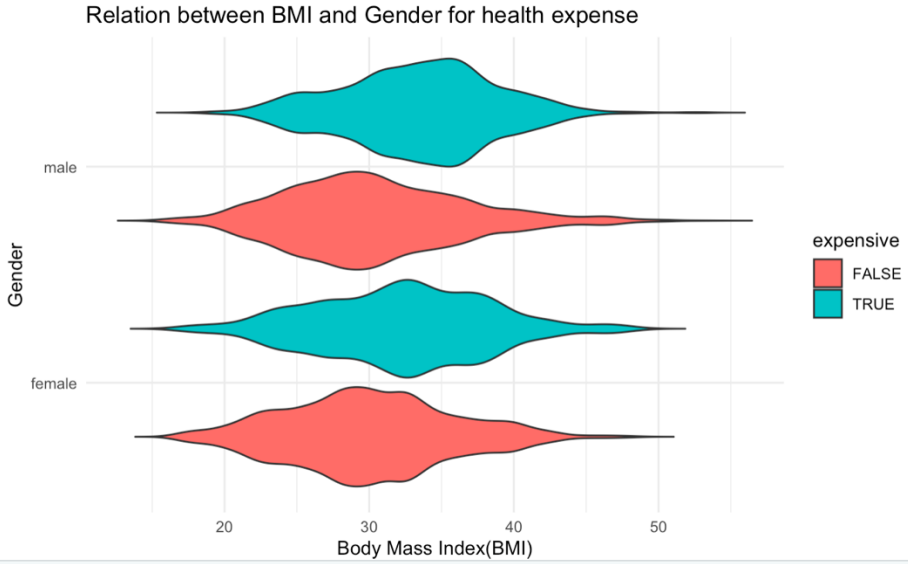
To dig deeper into exploring the patterns and trends of our data, we have visualized using violin-plot. In this visualization, we have used **bmi** on the x-axis and **gender** on the y-axis. To do this, we have used the library **ggplot2.** This allows us to use a system that creates graphics. For this, we provide the data, tell ggplot2 how to map variables to aesthetics (simply means, something you can see), what graphical measures to use and ggplot2 provides us the output based on our requirements. Violin plot is one of the many graphical representation techniques that ggplot2 allows us to use and conclude on our findings.

From the visualization, we can conclude that expensive males and females have high bmi as compared to their non-expensive counterparts. Higher bmi suggests that the individuals falling under this category have a unhealthy lifestyle, and that can be very well used to conclude why would they more for their healthcare as compared to the individuals who have a low bmi.

```
#adding violin-plot of bmi wrt gender
library(ggplot2)
vioPlot <- ggplot(data, aes(x=gender, y=bmi, fill=expensive)) +
  geom_violin(width=1,trim=FALSE) + theme_minimal() + coord_flip() + xlab("Gender") + ylab("Body Mass Index(BMI)") +
  ggtitle("Relation between BMI and Gender for health expense")

vioPlot

# From the visualization, we can see that expensive males and females have high bmi as compared to their non-expensive
counterparts
```



4.2.5 Technique 5: Histogram

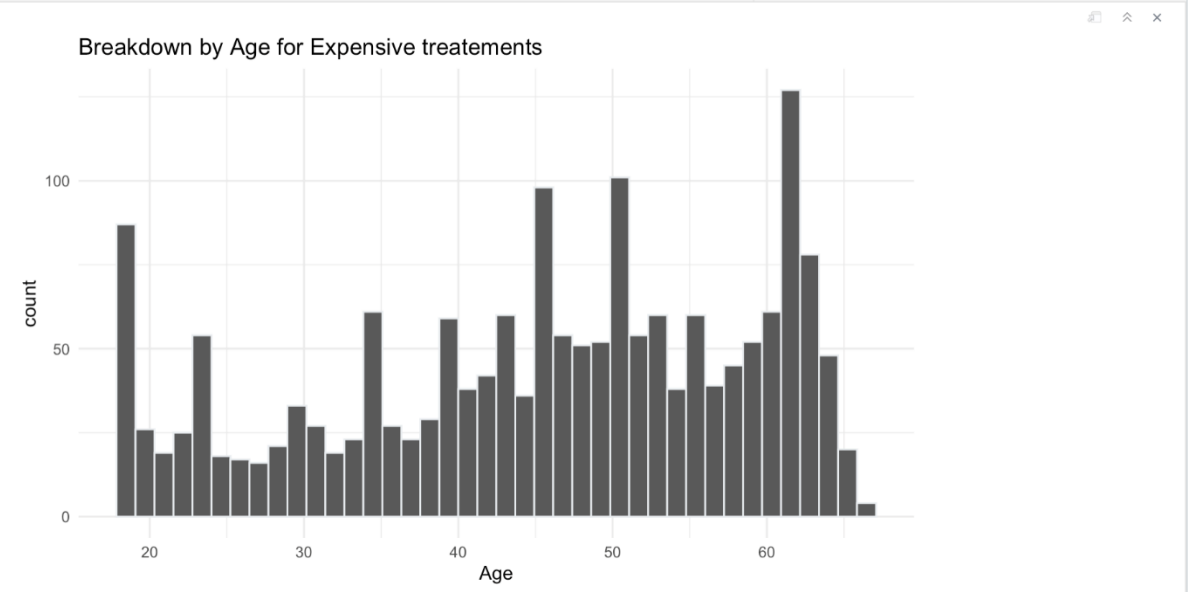
The next method for visualization used for out data is histogram. A histogram is majorly used when need to show frequency distributions in a dataset. For our data, we have used the frequency distributions based on the **age** of the people. This allows us to narrow down the age groups that fall under the expensive category of healthcare cost. It helps us understand which age group is more susceptible to illness than others.

From our visualization, we observe that there are three age ranges who pay comparatively a higher cost for healthcare than other age ranges. This is easily concluded by the spikes observed in the histogram. Majority of the people who pay a high cost fall under either of these three age ranges – below 20 years, between 45-50 and early 60s. Since, **age** was also a good predictor of the expense on healthcare, we can reply on this observation for our conclusion and suggestions.

```
# histogram plot of age for expensive data set
histPlot <-
  ggplot(dfExpensive,aes(x=age)) +
    geom_histogram( color="#e9ecef", position = 'identity', bins=40) + theme_minimal() + xlab("Age") +  ggtitle("Breakdown by Age for Expensive treatements")

histPlot

# from the plot, expensive individuals are mostly the ones with age less than 20, age greater than 60, and also with age around 45-50.
```

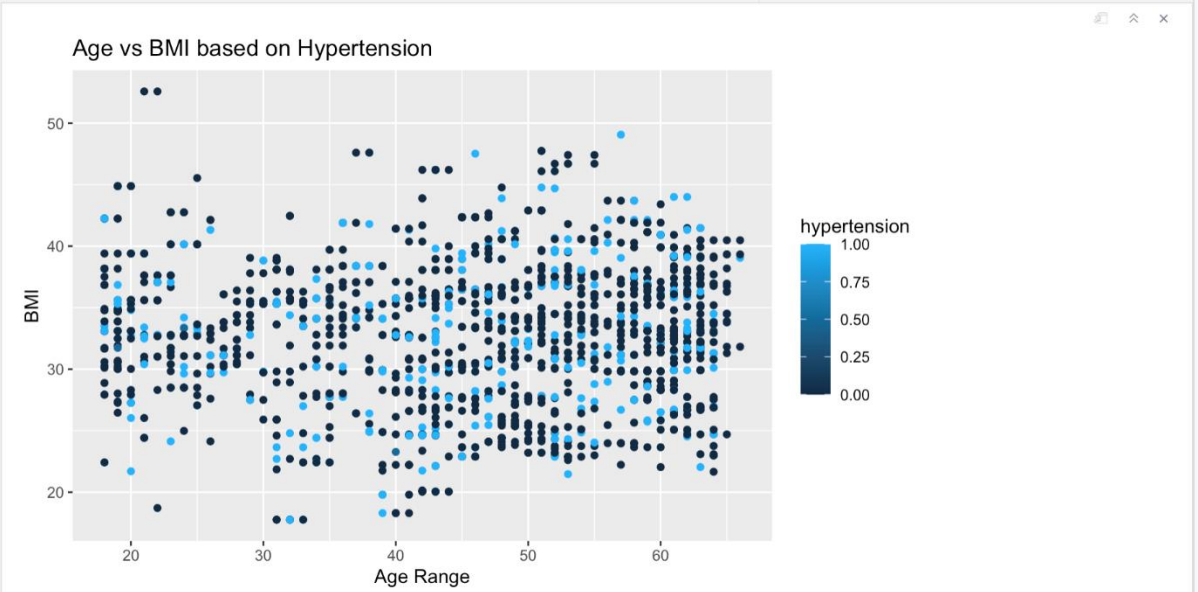


4.2.6 Technique 6: Scatter Plot

Our sixth technique of visualization is a scatter plot. It is a set of points plotted in a 2-dimensional setting with an x-axis and a y-axis. In our data, we have plotted **age** vs **bmi** based on the **hypertension** of individuals.

From the plot, it is observed that individuals with age greater than 60 and a high bmi pay a higher cost for healthcare, than the ones who have a low bmi. And we also see, that not a lot of people suffer from hypertension in our data set.

```r
# Scatter plot of age vs bmi based on hypertension

library(ggplot2)
scatterPlot <- ggplot(dfExpensive, aes(x=age, y=bmi, color=hypertension)) + geom_point() +
  ylab("BMI") + xlab("Age Range") + ggtitle("Age vs BMI based on Hypertension")
scatterPlot

# from the plot, it can be seen that individuals with age greater than 60 and a high bmi are expensive, though their levels of hypertension differ, most individuals do not suffer from hypertension
```

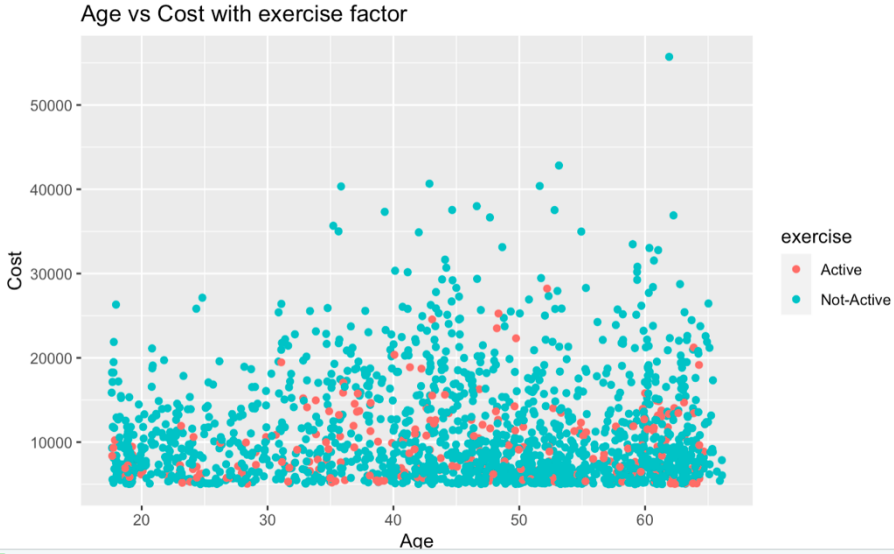


4.2.6 Technique 7: Jitter Plot

Our final technique of visualization is a jitter plot. This is a similar way to that of a scatter plot. The only difference in a jitter plot is that it helps visualize the relationship between a measurement variable and a categorical variable.

In our case, we have taken **age** as a measurement variable vs **cost**, with respect to the variable **exercise**, which is a categorical variable with two distinct values “active” and “not active”. rom the plot, we can infer that most individuals pay between $10000 and $15000, with most of them being over the age of 45 and also, there are few individuals with active workout routine.

```r
# jitter plot of age vs cost with excercise

library(ggplot2)
jitterPlot <- ggplot(dfExpensive, aes(x=age, y= cost, color= exercise)) +
        geom_jitter() + xlab("Age") + ylab("Cost") + ggtitle("Age vs Cost with exercise factor")

jitterPlot

# from the plot, we can infer that most individuals pay between $10000 and $15000, with most of them being over the age of 45 and also, there are few individuals with active workout routine.
```



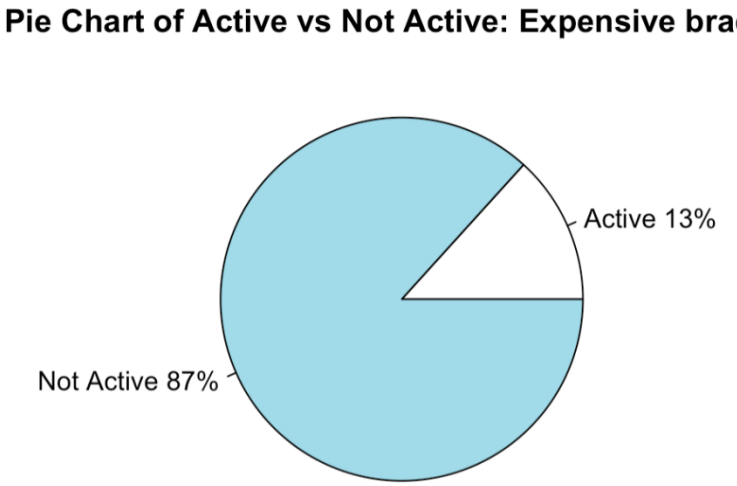
4.3 Correlation Analysis

Correlation analysis is a statistical method used to measure the strength of the linear relationship between two variables and compute their association. In our data set we have tried to compute the correlation between activity and cost. By doing so, we have arrived on two results.

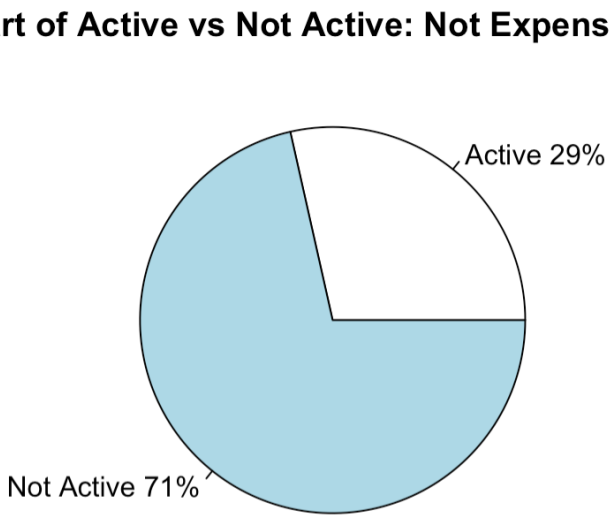
1. 13% people that fall under the expensive cost bracket have exercise in their routine.
2. 87% people that fall under expensive cost bracket do not have exercise in their routine.

```r
# Finding the correlation between activity and cost
```

```r
exerciseExpensive <- sum(if_else(dfExpensive$exercise=="Active",1,0))
exerciseExpensive
nrow(dfExpensive)

ratioExerciseExpensive <- exerciseExpensive/nrow(dfExpensive)
ratioExerciseExpensive
# 13.2% people that are active are in the expensive group

exerciseNotExpensive <- sum(if_else(dfNotExpensive$exercise=="Active",1,0))
exerciseNotExpensive
nrow(dfNotExpensive)

ratioExerciseNotExpensive <- exerciseNotExpensive/nrow(dfNotExpensive)
ratioExerciseNotExpensive
# 28.5% of people who are active are in the not expensive group.
```



To get more clarity on our data, we performed the same correlation on the non-expensive bracket of people.

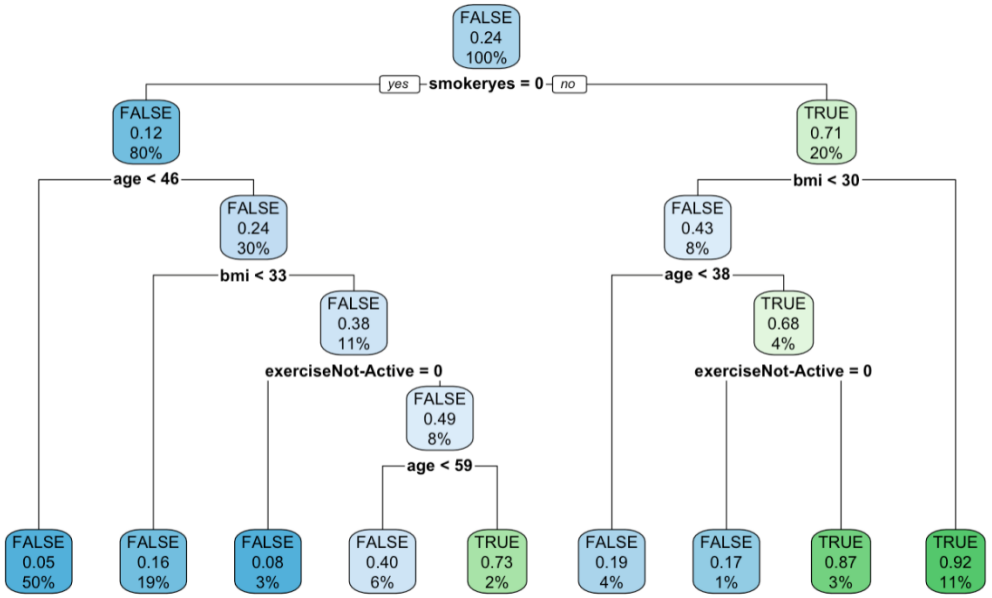
```r
```{r}
exerciseNotExpensive <- sum(if_else(dfNotExpensive$exercise=="Active",1,0))

ratioExerciseNotExpensive <- exerciseNotExpensive/nrow(dfNotExpensive)
# 28.5% of people who are active are in the not expensive group.

slices <- c(ratioExerciseNotExpensive*100, (1-ratioExerciseNotExpensive)*100)
lbls <- c("Active","Not Active")
pct <- round(slices/sum(slices)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie(slices,labels = lbls,
    main="Pie Chart of Active vs Not Active: Not Expensive bracket")
```
```

Stated below are the conclusions that we drew from this analysis.

1. 29% people that fall under the not expensive cost bracket have exercise in their routine.
2. 71% people that fall under not expensive cost bracket do not have exercise in their routine.



4.4 Predictive Modeling

In this section, we will discuss about predictive modeling. Predictive modeling is a data analysis technique that allows us to predict future behavior based on the past behavior. Once we have successfully built a model, we use the results to make predictions about the future. For our data set, we have included 4 libraries to perform predictive modeling. These are:

1. Rpart: Recursive partitioning and regression trees – used for building classification and regression trees.
2. E1071: a package for R programming that provides functions for statistic and probabilistic algorithms
3. Kernlab: kernel-based machine learning lab
4. Caret: classification and regression training

To run predictive models on the data, we first read in the test data provided to us.

```
213 # Reading in the test data
214
215 ```{r}
216 testData <- read_csv("~/Downloads/HMO_TEST_data_sample.csv")
217 testDataPrediction <- read_csv("~/Downloads/HMO_TEST_data_sample_solution.csv")
218 TEST_PRED = as.factor(testDataPrediction$expensive)
219 ```
```

4.4.1 SVM (Support Vector Machine) Modeling

Support vector machines (SVM) are supervised learning models with associated learning algorithms that analyze data used for classification and regression analysis. It is mostly used in classification problems. svm is used to train a support vector machine. It can be

used to carry out general regression and classification (of nu and epsilon-type), as well as density-estimation. A formula interface is provided.

The createDataPartition() function was used to randomly select 70% of the data with respect to high costs and assigning it to trainList. Next, the trainSet and testSet were created to be used to train and test our model. This model was not efficient, so we tested other models.

```
220
221 # SVM Model
222
223 ```{r}
224 data$expensive <- as.factor(data$expensive)
225
226 trainList <- createDataPartition(y=data$expensive,p=0.7,list=FALSE)
227 trainSet <- data[trainList,]
228 testSet <- data[-trainList,]
229
230 svmModel <- train(expensive~age+bmi+children+smoker+exercise+hypertension,data=trainSet,method="svmRadial",trControl=trainCon
    trol(method="none"),preProcess=c("center","scale"))
231 svmModel
232
233 predictValues <- predict(svmModel,testSet)
234
235 confusionMatrix(predictValues, testSet$expensive)
236
237 ```
```

4.4.2 KSVM Modeling

The KSVM model was trained against the trainset training data, while using the age, bmi, children, smoker, exercise and hypertension variable. Based on the output of the confusion matrix, the accuracy and sensitivity of the model were not promising.

```
239 # KSVM
240
241 ```{r}
242 svm.model<-ksvm(data=trainSet,expensive~age+bmi+children+smoker+exercise+hypertension)
243 svm.model
244 predictValues <- predict(svm.model,testSet)
245 confusionMatrix(predictValues,testSet$expensive)
246 ```
```

4.4.3 SVM model with KFold

The SVM model with Kfold was trained against the trainset. The training control method was set to “repeatedcv”, the number of folds was set to 10 and the repeats to 5. This model was predicted with the testSet. The output of the confusion matrix was high with an accuracy of 88.3% and sensitivity of 97%,

```
# SVM model with KFold #88.3% Accuracy, 97% sensitivity

```{r}
trctrl <- trainControl(method="repeatedcv",number=10,repeats=5)
svm.model.kfold <- train(expensive~age+bmi+children+smoker+exercise+hypertension,data=trainSet,method="svmRadial",trControl=t
rctrl,preProcess=c("center","scale"))
svm.model.kfold
predictValues <- predict(svm.model.kfold,testSet)
confusionMatrix(predictValues,testSet$expensive)
```
```

4.4.4 Model with Rpart

The last model we tested was the Rpart model. This model type, unlike the SVM, provides a more intuitive explanation of how the model works which were helpful when generating the actionable insights. The below code was written to generate the model for the high costs.

The createDataPartition() function was used to randomly select 70% of the data with respect to high costs and assigning it to trainList. Next, the trainSet and testSet were created to be used to train and test our model. The model is being trained with method ”rpart” while using data set = “trainSet”. The training method is set to trctrl.

On training this model and testing the testSet against it, it gave an accuracy of 88.92% and 97% sensitivity.

The SVM model with Kfold and Rpart gave us approximately similar results, but since the Rpart model has a slight high accuracy, it was considered the best.

```
#Model with rpart - 88.92% accuracy, 97% sensitivity

```{r}
model.rpart <- train(expensive~age+bmi+children+smoker+exercise+hypertension,method="rpart",data=trainSet,trControl=trctrl,tu
neLength=50)
predictValues <- predict(model.rpart,testSet)
confusionMatrix(predictValues,testSet$expensive)

model.rpart

library(rpart.plot)
rpart.plot(model.rpart$finalModel)
```
```


FALSE
0.24
100%
yes
smokeryes = 0
no
FALSE
0.12
80%
TRUE
0.71
20%
age < 46
bmi < 30
FALSE
0.24
30%
FALSE
0.43
8%
bmi < 33
age < 38
FALSE
0.38
11%
TRUE
0.68
4%
exerciseNot-Active = 0
exerciseNot-Active = 0
FALSE
0.49
8%
age < 59
FALSE
0.05
50%
FALSE
0.16
19%
FALSE
0.08
3%
FALSE
0.40
6%
TRUE
0.73
2%
FALSE
0.19
4%
FALSE
0.17
1%
TRUE
0.87
3%
TRUE
0.92
11%

5. Shiny App

In this section, we discuss how we built the shiny app and which model we used for predicting results of the uploaded data files.

5.1 User Interface Code and App

On analyzing of various machine learning models, we finalized with Unsupervised Rpart model and trained data set. For an interactive web application, we imported shiny package in R library. Shiny apps are contained in a single script with extensions '. R'. It has three components UI for user interface, server for output section, shinyApp function to combine UI and server. We installed libraries shiny, shinythemes and created user interface. Below snippet is for user interface that we used.

```
ui <- fluidPage(theme=shinytheme("yeti"),

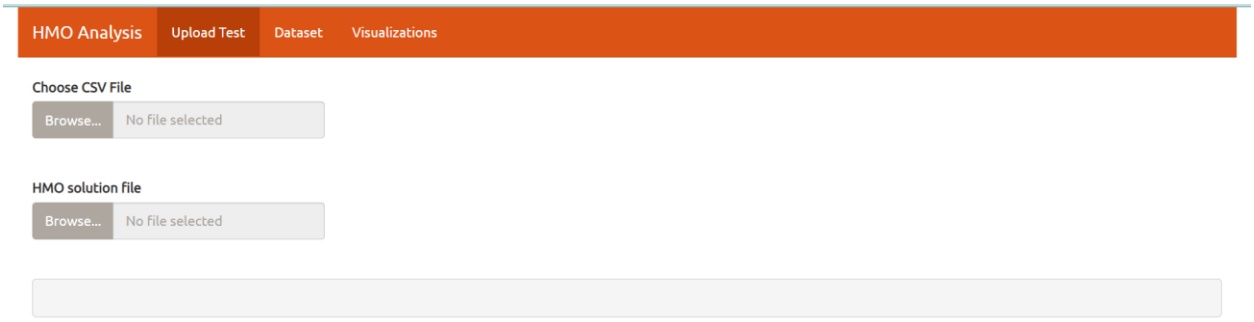
                navbarPage(
                  "HMO Analysis",
                  tabPanel("Upload Test",
                           fileInput(inputId = "upload", "Choose CSV File",accept = ".csv"),
                           fileInput("upload_Solution", label="HMO solution file",accept = c(".csv")),
                           #get a number (how much of the dataframe to show)
                           verbatimTextOutput("txt_results", placeholder = TRUE)),

                  tabPanel("Dataset",
                           numericInput("n", "Number of Rows", value = 5, min = 1, step = 1),
                           #a place to output a table (i.e., a dataframe)
                           tableOutput("headForDF")),
                  tabPanel("Visualisations",
                          # plotOutput("distPlot",height = "400px"),
                           plotOutput("sp_age",height = "400px"),
                           plotOutput("sp_bmi",height = "400px"),
                           plotOutput("sp_children_cost",height = "400px"),
                           plotOutput("box_cost",height = "400px"),
                           plotOutput("map_plot",height = "400px"),
                          plotOutput("plot6",height = "400px"),
                          plotOutput("plot7",height = "400px"),
                          plotOutput("plot8",height = "400px"),
                          plotOutput("plot9",height = "400px"),
                          plotOutput("plot10",height = "400px"),
                          plotOutput("plot11",height = "400px"),
                          #plotOutput("plot12",height = "400px"),


                           theme = shinytheme("yeti")
                  ),
                ),

)
```

In this interface the menu section is about the data where user can input file (only csv), then result section predicted results bases on trained model, confusion matrix of trained model, sensitivity for user data, and for visualizations we used map, bar plots, scatter plot and histograms.



5.2 Server Side

After UI, we developed code for server side which mainly uses to output for input in UI, and results are displayed on respective actions. At the end we call our shinyAPP function to combine UI, sever.

```
# Define server logic required to draw a histogram
server <- function(input, output,session) {
  #require an input file, then read a CSV file
  getTestData <- reactive({
    req(input$upload)
    read_csv(input$upload$name)
  })
  #require an the actual values for the prediction (i.e. solution file)
  getSolutionData <- reactive({
    req(input$upload_Solution)
    read_csv(input$upload_Solution$name)
  })
  #show the output of the model
  output$txt_results <- renderPrint({
    #load the data
    dataset <- getTestData()
    dataset_solution <- getSolutionData()
    #load and use the model on the new data
    use_model_to_predict(dataset, dataset_solution)
  })

  #show a few lines of the dataframe
  output$headForDF <- renderTable({
    df <- getTestData()
    head(df, input$n)
  })


  #Graph plot 1:
  smokersExpensive <- sum(if_else(dfExpensive$smoker=="yes",1,0))
  #1063 smokers out of 1792 people in the expensive set
  ratioSmokersExpensive <- smokersExpensive/nrow(dfExpensive)
  slices <- c(ratioSmokersExpensive*100, (1-ratioSmokersExpensive)*100)
  lbls <- c("Smokers","Non-Smokers")
  pct <- round(slices/sum(slices)*100)
  lbls <- paste(lbls, pct) # add percents to labels
  lbls <- paste(lbls,"%",sep="") # ad % to labels
  # pie(slices,labels = lbls, col=rainbow(length(lbls)),
  #     main="Pie Chart of Smokers vs Non-Smokers: Expensive bracket")
  #


  #Scatter plot Age Vs Cost
  output$sp_age <- renderPlot(
```

6. Results, conclusions and suggestions

On analyzing and studying the data, we came across trends that made quite the sense for higher cost of healthcare for some people, while a low cost for others. We saw how smoking affected the cost. For people who are smokers, the cost was higher. Majority of the non-smokers fell under the not expensive cost category.

We also saw that age was also a deciding factor for having a higher cost of healthcare. 3 age groups (refer 4.2.3) were more dominant in this case than others.

As a solution to this, we suggest running awareness campaigns and broadcasting the findings to the public. With the information that we just figured out, we can try to run campaigns and awareness focusing on the highlighted age groups in section 4.2.3 and make sure to convey the message that people falling under these age categories be suggested to quit smoking.