Final Project Report

Code

Hide

```
#NAMES: HAOTIAN SHEN, ENUBI KIM, WEI LIAO, RHIANNON ABRAMS
```

#Loading in the libraries we need.

Hide

```
library(rio)
```

```
Registered S3 method overwritten by 'data.table':
  method           from
  print.data.table
The following rio suggested packages are not installed: ‘arrow’, ‘feather’, ‘fst’, ‘hexView’, ‘p
zfx’, ‘readODS’, ‘rmatio’
Use 'install_formats()' to install them
```

Hide

```
library(kernlab)
library(caret)
```

```
Loading required package: ggplot2

Attaching package: ‘ggplot2’

The following object is masked from ‘package:kernlab’:

    alpha

Loading required package: lattice
```

Hide

```
library(rpart)
library(rpart.plot)
library(imputeTS)
```

```
Registered S3 method overwritten by 'quantmod':
  method            from
  as.zoo.data.frame zoo
```

Hide

```
library(tidyverse)
```

```
Registered S3 methods overwritten by 'dbplyr':
  method         from
  print.tbl_lazy
  print.tbl_sql
── Attaching packages ──────────────────────────────────────────────────
────── tidyverse 1.3.2 ──✓ tibble  3.1.8     ✓ dplyr   1.0.9
✓ tidyr   1.2.0     ✓ stringr 1.4.1
✓ readr   2.1.2     ✓ forcats 0.5.2
✓ purrr   0.3.4     ── Conflicts ──────────────────────────────────────
───────────────────── tidyverse_conflicts() ──
✖ ggplot2::alpha() masks kernlab::alpha()
✖ purrr::cross()   masks kernlab::cross()
✖ dplyr::filter()  masks stats::filter()
✖ dplyr::lag()     masks stats::lag()
✖ purrr::lift()    masks caret::lift()
```

Hide

```
library(ggplot2)
library(arules)
```

```
Loading required package: Matrix

Attaching package: ‘Matrix’

The following objects are masked from ‘package:tidyr’:

    expand, pack, unpack


Attaching package: ‘arules’

The following object is masked from ‘package:dplyr’:

    recode

The following object is masked from ‘package:kernlab’:

    size

The following objects are masked from ‘package:base’:

    abbreviate, write
```

Hide

```
library(ggmap)
```

```
Google's Terms of Service: https://cloud.google.com/maps-platform/terms/.
Please cite ggmap if you use it! See citation("ggmap") for details.
```

#DATA CLEANING ##1.Deal with missing data points

Hide

```
# Download the dataset from url and check for the missing data.
datafile <- "https://intro-datascience.s3.us-east-2.amazonaws.com/HMO_data.csv"
df <- read.csv(datafile)
sum(is.na(df$bmi))
```

```
[1] 78
```

Hide

```
sum(is.na(df$hypertension))
```

```
[1] 80
```

Hide

```
#Comment: There are 78 and 80 missing data points in bmi and hypertension respectively.

df$bmi <- na_interpolation(df$bmi)
df <- df %>% filter(!is.na(hypertension))
```

##2.Inspect the dataset

Hide

```
str(df)
```

```
'data.frame':   7502 obs. of  14 variables:
 $ X              : int  1 2 3 4 5 7 9 10 11 12 ...
 $ age            : int  18 19 27 34 32 47 36 59 24 61 ...
 $ bmi            : num  27.9 33.8 33 22.7 28.9 ...
 $ children       : int  0 1 3 0 0 1 2 0 0 0 ...
 $ smoker         : chr  "yes" "no" "no" "no" ...
 $ location       : chr  "CONNECTICUT" "RHODE ISLAND" "MASSACHUSETTS" "PENNSYLVANIA" ...
 $ location_type  : chr  "Urban" "Urban" "Urban" "Country" ...
 $ education_level: chr  "Bachelor" "Bachelor" "Master" "Master" ...
 $ yearly_physical: chr  "No" "No" "No" "No" ...
 $ exercise       : chr  "Active" "Not-Active" "Active" "Not-Active" ...
 $ married        : chr  "Married" "Married" "Married" "Married" ...
 $ hypertension   : int  0 0 0 1 0 0 0 1 0 0 ...
 $ gender         : chr  "female" "male" "male" "male" ...
 $ cost           : int  1746 602 576 5562 836 3842 1304 9724 201 4492 ...
```

Hide

```
summary(df)
```

```
       X                  age            bmi          children       smoker           locati
on
 Min.   :        1   Min.   :18.00   Min.   :15.96   Min.   :0.000   Length:7502        Length:7
502
 1st Qu.:     5635   1st Qu.:26.00   1st Qu.:26.60   1st Qu.:0.000   Class :character   Class :c
haracter
 Median :    25212   Median :39.00   Median :30.50   Median :1.000   Mode  :character   Mode  :c
haracter
 Mean   :   717291   Mean   :38.92   Mean   :30.79   Mean   :1.108
 3rd Qu.:   119119   3rd Qu.:51.00   3rd Qu.:34.70   3rd Qu.:2.000
 Max.   :131101111   Max.   :66.00   Max.   :53.13   Max.   :5.000
 location_type      education_level    yearly_physical      exercise           married
hypertension
 Length:7502        Length:7502        Length:7502        Length:7502        Length:7502
Min.   :0.0000
 Class :character   Class :character   Class :character   Class :character   Class :character
1st Qu.:0.0000
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character
Median :0.0000

Mean   :0.2005

3rd Qu.:0.0000

Max.   :1.0000
    gender               cost
 Length:7502        Min.   :    2.0
 Class :character   1st Qu.:  966.5
 Mode  :character   Median : 2500.0
                    Mean   : 4049.5
                    3rd Qu.: 4778.8
                    Max.   :55715.0
```

[Comments] We are dealing with a data set with 7502 rows and 14 columns. Cost will be our predictive variables, while the other 12 attributes: age, bmi, number of children, smoker or not, locations, education level, exercise yearly or not, married or not, hypertension or not, gender could be our predictors.

##3.Perform binning and transformation on our variables

Hide

```
#1. age
df_add_age <- df %>% mutate(age_group = case_when(
  df$age < 20 ~ "under 18",
  df$age >= 20 & df$age < 30 ~ "20-29",
  df$age >= 30 & df$age < 40 ~ "30-39",
  df$age >= 40 & df$age < 50 ~ "40-49",
  df$age >= 50 & df$age < 60 ~ "50-59",
  df$age >= 60 ~ 'over 60'
))

#2. bmi
df_add_bmi <- df_add_age %>% mutate(bmi_group = case_when(
  df_add_age$bmi < 18.5 ~ "Underweight",
  df_add_age$bmi >= 18.5 & df_add_age$bmi < 24.9 ~ "Normal Weight",
  df_add_age$bmi >= 24.9 & df_add_age$bmi < 29.9 ~ "Overweight",
  df_add_age$bmi >= 29.9 ~ "Obesity"
))
df_new <- df_add_bmi

# Adding new logical (binary) label of some categorical variables
# 1. Education_level - is_educated (yes, no)
df_add_edu_bin <- df_new %>% mutate(is_educated = case_when(
  df_new$education_level != "No College Degree" ~ "yes",
  TRUE ~ "no"
))


#2. children  - have_child (yes, no)
df_add_child_bin <- df_add_edu_bin %>% mutate(have_child = case_when(
  df_add_edu_bin$children == 0 ~ "no",
  TRUE ~ "yes"
))

df_new <- df_add_child_bin
df_new$hypertension <- ifelse(df_new$hypertension==1, 'yes', 'no')
head(df_new)
```

	X <int>	a.. <int>	bmi <dbl>	children <int>	smo... <chr>	location <chr>	location_type <chr>	education_level <chr>	yearly_physic <chr>
1	1	18	27.900	0	yes	CONNECTICUT	Urban	Bachelor	No
2	2	19	33.770	1	no	RHODE ISLAND	Urban	Bachelor	No
3	3	27	33.000	3	no	MASSACHUSETTS	Urban	Master	No
4	4	34	22.705	0	no	PENNSYLVANIA	Country	Master	No
5	5	32	28.880	0	no	PENNSYLVANIA	Country	PhD	No
6	7	47	33.440	1	no	PENNSYLVANIA	Urban	Bachelor	No

6 rows | 1-10 of 18 columns

##4. Set the boundary for expensive and not expensive

Hide

```
mean(df_new$cost) # Average cost is 4049.5
```

```
[1] 4049.492
```

Hide

```
range(df_new$cost) # (2, 55715)
```

```
[1]     2 55715
```

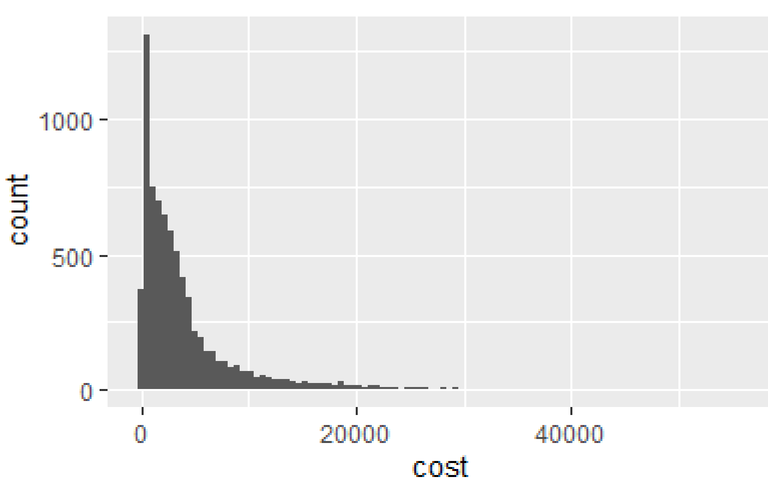
Hide

```
quantile(df_new$cost, probs = 0.8) # 80% people spend less than or equal to 5789.4
```

```
   80%
5789.4
```

Hide

```
ggplot(df_new,aes(x=cost))+geom_histogram(bins=100) #The cost has long-tailed effects on the rig
ht.
```



Hide

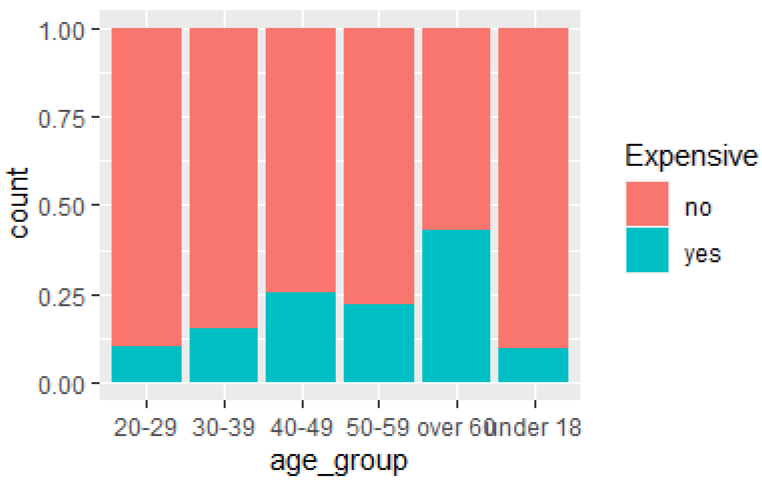
```

#Expensive means a person spends more than 6000 N(included) on his/her health
df_new$Expensive <- ifelse(df_new$cost >= 6000, 'yes', 'no')
```

[Comments] We set the boundary for expensive or not to be 6000. People who were charged more than 6000 dollars will be labeled as “expensive”, while people who paid less will be labeled as “not expensive”.

#DATA VISUALIZATION #1.bar charts

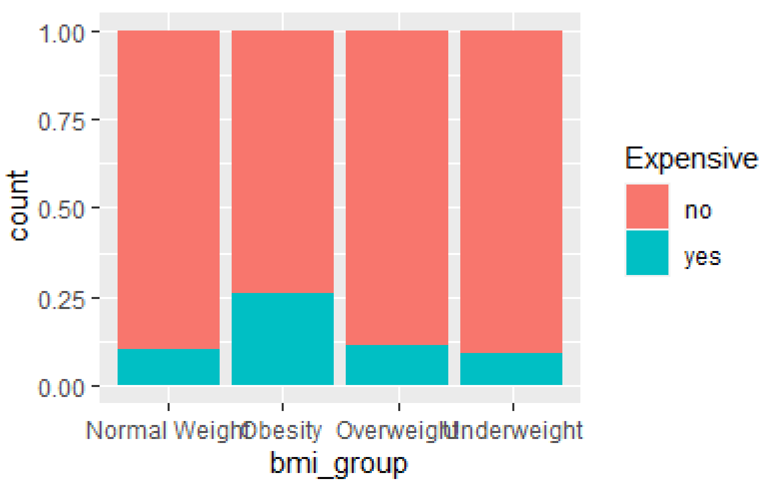
Hide

```
ggplot(df_new, aes(fill= Expensive, x = age_group)) + geom_bar(position = "fill")
```



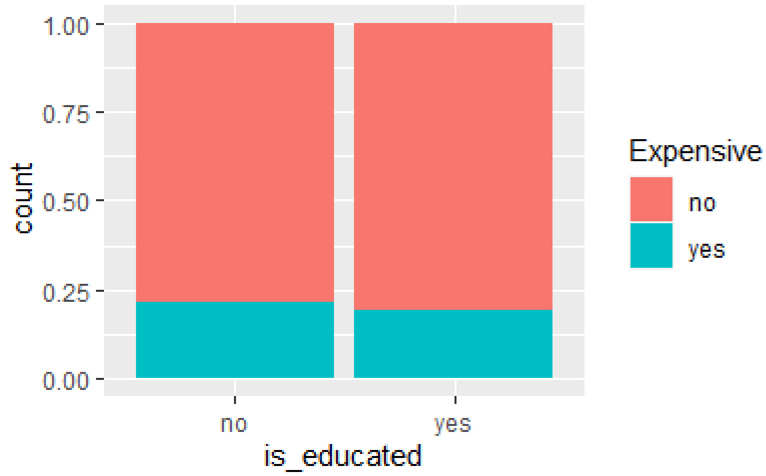
Hide

```
ggplot(df_new, aes(fill= Expensive, x = bmi_group)) + geom_bar(position = "fill")
```



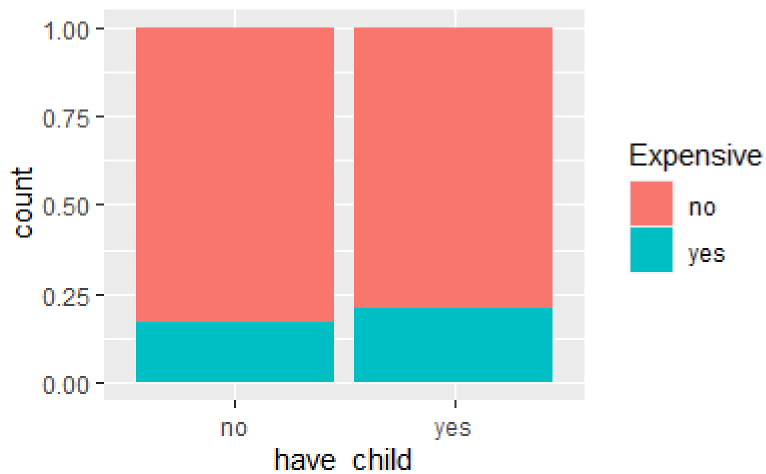
Hide

```
ggplot(df_new, aes(fill= Expensive, x = is_educated)) + geom_bar(position = "fill")
```



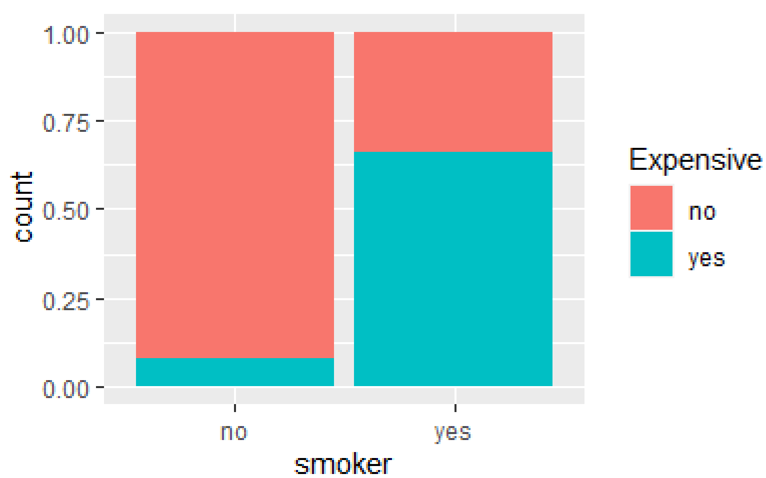
Hide

```
ggplot(df_new, aes(fill= Expensive, x = have_child)) + geom_bar(position = "fill")
```



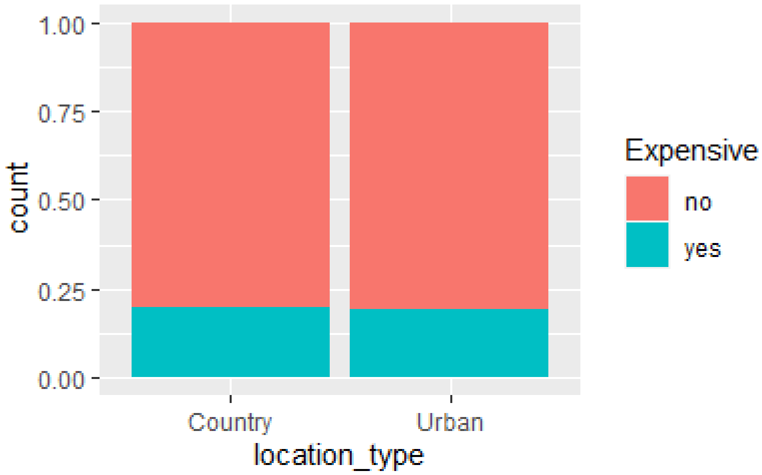
Hide

```
ggplot(df_new, aes(fill= Expensive, x = smoker)) + geom_bar(position = "fill")
```



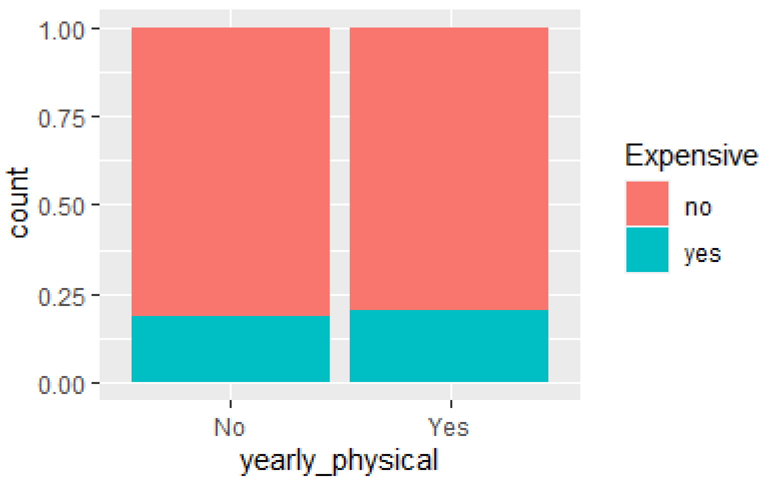
Hide

```
ggplot(df_new, aes(fill= Expensive, x = location_type)) + geom_bar(position = "fill")
```



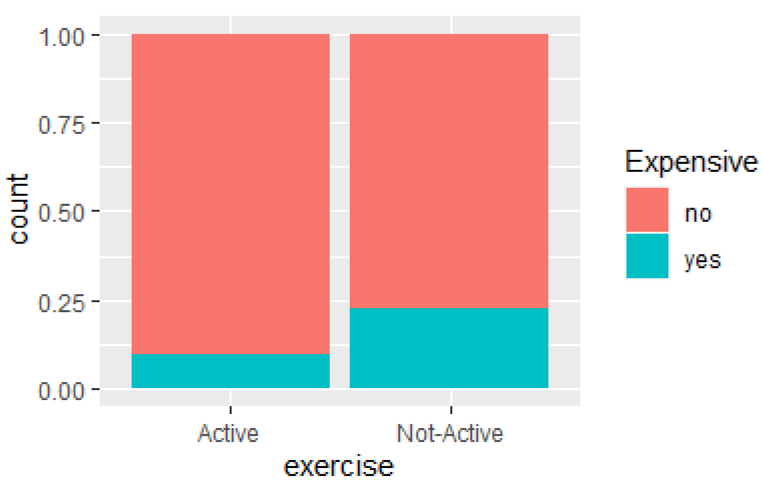
Hide

```
ggplot(df_new, aes(fill= Expensive, x = yearly_physical)) + geom_bar(position = "fill")
```



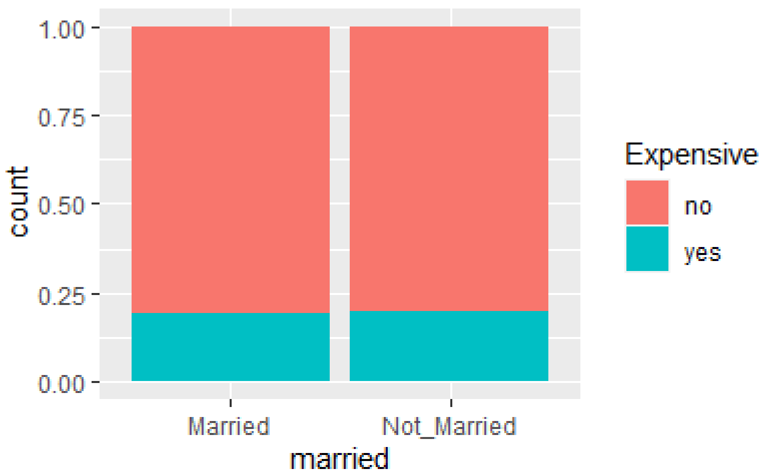
Hide

```
ggplot(df_new, aes(fill= Expensive, x = exercise)) + geom_bar(position = "fill")
```



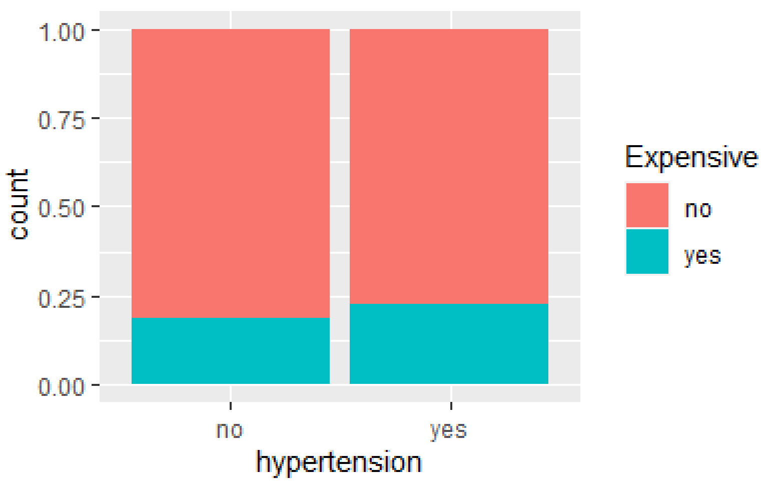
Hide

```
ggplot(df_new, aes(fill= Expensive, x = married)) + geom_bar(position = "fill")
```



Hide

```
ggplot(df_new, aes(fill= Expensive, x = hypertension)) + geom_bar(position = "fill")
```



Hide

```
ggplot(df_new, aes(fill= Expensive, x = gender)) + geom_bar(position = "fill")
```

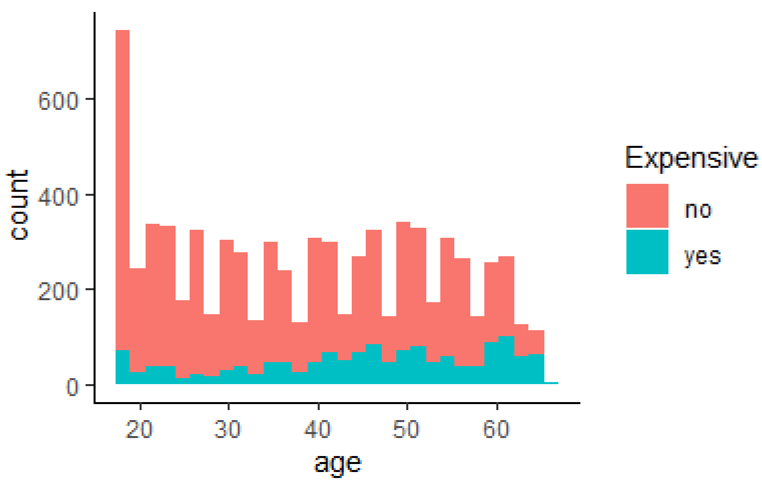
[Comments] All the bar charts demonstrate that the percentages of people paying more than 6000 can vary among different groups.They gave us a general ideas on which attribute might be valid predictor. For example, for people who are smokers, the percentage of people paying more than 6000 is significantly higher vs people who are not smokers. However, the differences in percentage is not that significant for people have children vs doesn't, educated vs not educated, live in country or urban, yearly_physical or not, married or not.

Next, we will look at the attributes independently to get more insights.

1. age

Hide

```
# histogram
ggplot(df_new, aes(age, fill=Expensive)) + geom_histogram() + theme_classic()
```



[Comments] 1. According to the above histogram, the age of most people in the data set is under 20. 2. [Expensive - No] The distribution of this group shows a multimodal shape and a peak in the under-20 categories. 3. [Expensive - Yes] As seen in the green area, we would say that the older a person is, the more he/she will pay for healthcare.

Hide

```
# grouping (age_group ~ number of observation)
# table
age_group <- df_new %>%
  group_by(age_group, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'age_group'. You can override using the `.groups` argument.
```

Hide

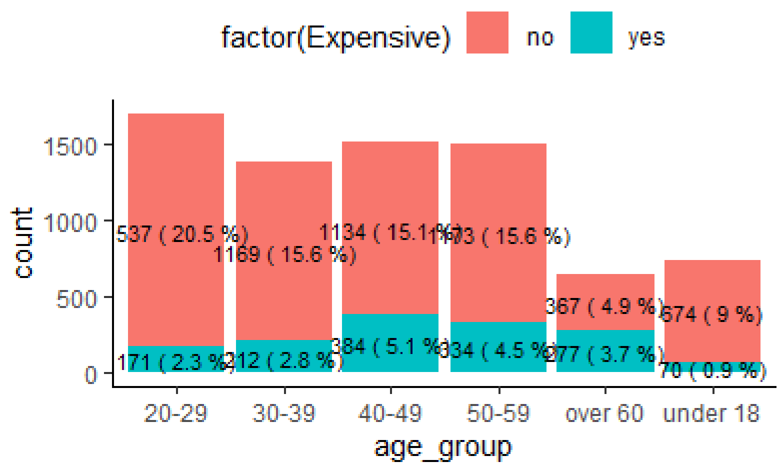
```
colnames(age_group)[3] <- "count"
age_group <- age_group %>% mutate(prop = round(count/7502, 3))
age_group
```

age_group <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
20-29	no	1537	24.23682	8.5493462	2.9239265	0.205
30-39	no	1169	34.37468	8.4964816	2.9148725	0.156
40-49	no	1134	44.47972	8.8182114	2.9695473	0.151
50-59	no	1173	54.24467	8.0194362	2.8318609	0.156
over 60	no	367	61.84741	2.0914072	1.4461698	0.049
under 18	no	674	18.42730	0.2450783	0.4950538	0.090
20-29	yes	171	23.95322	8.7154455	2.9521933	0.023
30-39	yes	212	34.77830	7.6236028	2.7610872	0.028
40-49	yes	384	44.80729	7.8478840	2.8014075	0.051
50-59	yes	334	54.30240	8.6380123	2.9390496	0.045

1-10 of 12 rows Previous **1** 2 Next

Hide

```
# plot
ggplot(age_group, aes(age_group, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



[Comments] The table and bar chart shows the detailed statistical results of two groups (high and low cost) in terms of age. 1. The age group under 50 accounts for more than half of the entire population in the data set. 2. [Expensive - No] Among the people who pay fewer costs for healthcare, the 20-29 age group has the highest proportion of the whole population. 3. [Expensive - Yes] There are most people in the 40-49 age group with the highest healthcare cost.

Hide

```
# grouping (age_group ~ cost)
# table
age_group_cost <- df_new %>%
                    group_by(age_group, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'age_group'. You can override using the `.groups` argument.
```

Hide

```
age_group_cost <- age_group_cost %>% mutate(prop = round(total/30379292 ,3))
age_group_cost
```

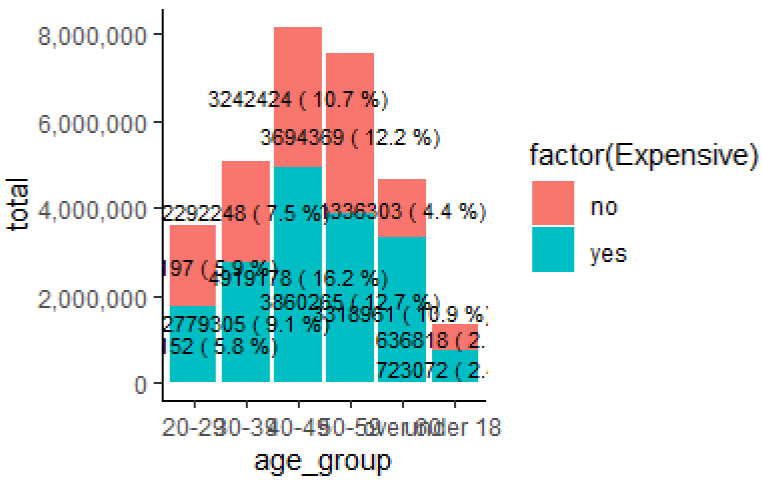
age_group <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
20-29	no	1801197	1171.8913	5954	2	1565567	1251.226	0.059
30-39	no	2292248	1960.8623	5945	8	1579157	1256.645	0.075
40-49	no	3242424	2859.2804	5968	7	1568250	1252.298	0.107
50-59	no	3694369	3149.5047	5986	18	1621382	1273.335	0.122
over 60	no	1336303	3641.1526	5900	34	1743923	1320.577	0.044
under 18	no	636818	944.8338	5938	4	1458238	1207.575	0.021
20-29	yes	1775152	10381.0058	27136	6010	14595061	3820.348	0.058

age_group <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
30-39	yes	2779305	13109.9292	40336	6007	37784598	6146.918	0.091
40-49	yes	4919178	12810.3594	40664	6004	46016943	6783.579	0.162
50-59	yes	3860265	11557.6796	42820	6001	40944549	6398.793	0.127

1-10 of 12 rows Previous 1 2 Next

Hide

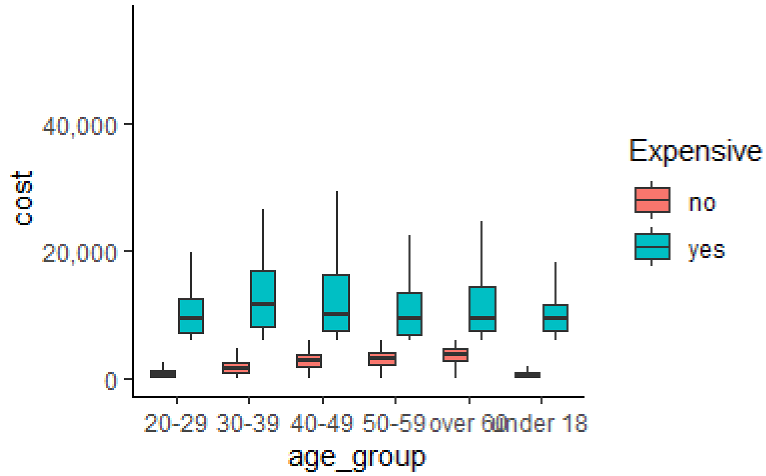
```
# plot
ggplot(age_group_cost, aes(age_group, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] Unlike the above results, as we consider the cost data together, the age group between 40~59 spent a lot of money on their health care. 1) [Overall] The age group 40-49 has the highest total costs and proportion in the entire population. 2) [Expensive - Yes] Among the people with lower healthcare costs, the proportion and total value of the age group 40-49 are the highest with $4,919,178 (16.2%). 3) [Expensive - No] The age group 50-59 pay the highest costs for healthcare services. ($3,694,369 - 12.2%)

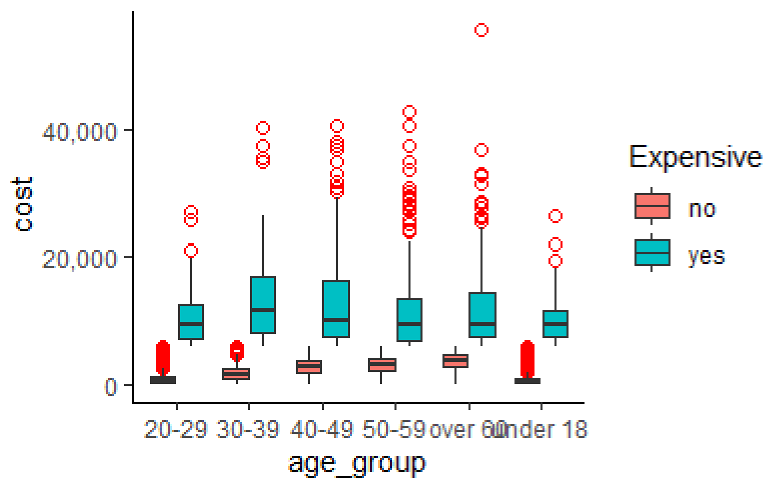
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(age_group, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(age_group, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] The boxplots are made using the age_group and cost columns. 1) As seen in the box plot with the outliers, we can find the outliers of $55,715 on the age group over 60 with the higher healthcare cost. 2) We can also figure out that the age group 50-59 has many outliers in the boxplot with outliers. 3) Without the outliers, the boxplots of each 'Expensive' group show similar shapes. The 'Expensive - yes' group has a more variable range of values than the 'Expensive - no' group in terms of healthcare costs. 4) There are also outliers in the groups: the age group 20-29, 30-39, and under 18 with expensive healthcare cost.

2. bmi

[Comments] 1. The histogram of the overall population in the bmi column shows a normal distribution (a bell shape). 2. [Expensive - No] The distribution with a red color has a bell shape, so we would conclude this is a normal distribution. 3. [Expensive - Yes] As seen in the green area, the shape of this histogram has a right-skewed shape. That means the data would have a higher bmi value than the 'Expensive - yes' group.

Hide

```
# grouping (bmi_group ~ number of observation)
# table
bmi_group <- df_new %>%
  group_by(bmi_group, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'bmi_group'. You can override using the `.groups` argument.
```

Hide

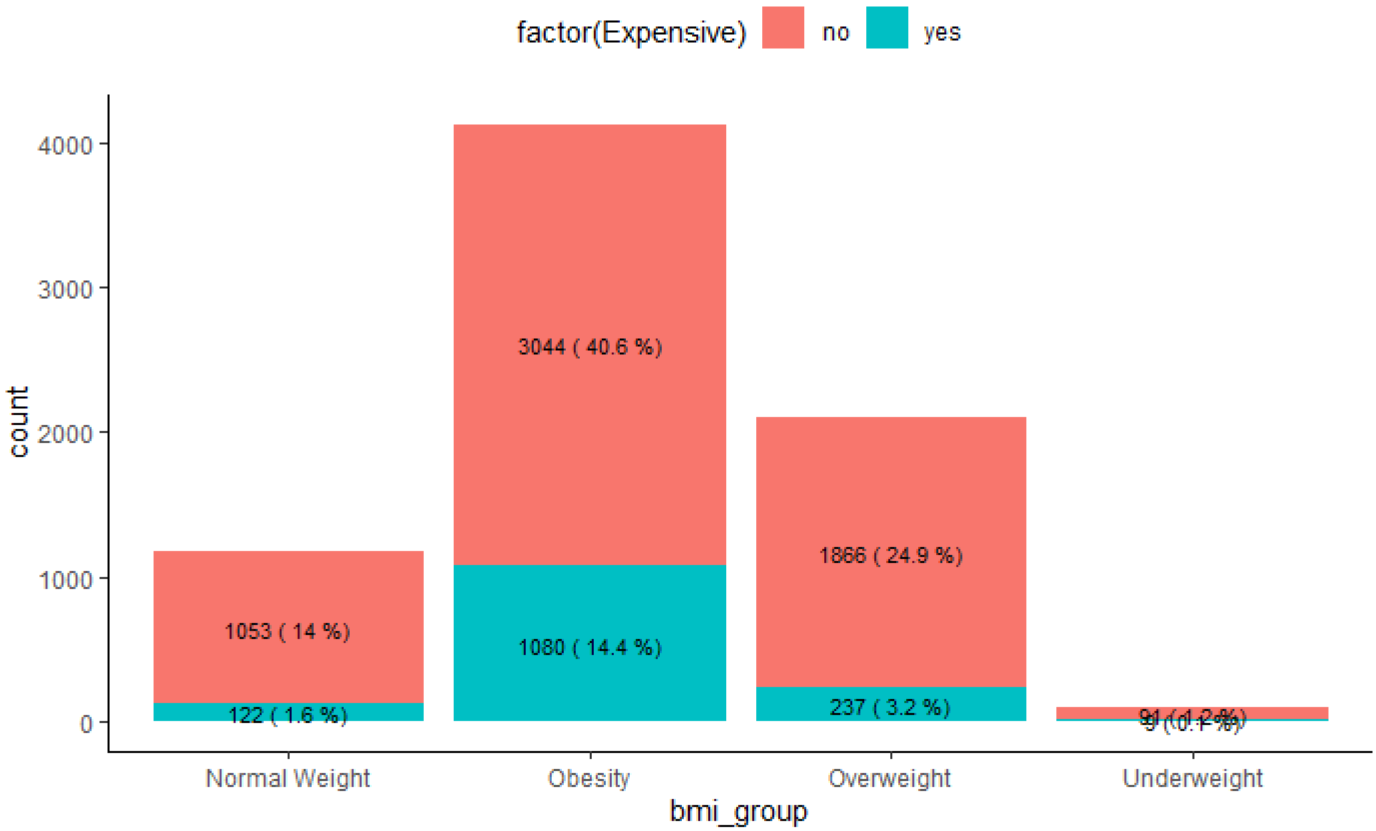
```
colnames(bmi_group)[3] <- "count"
bmi_group <- bmi_group %>% mutate(prop = round(count/7502, 3))
bmi_group
```

bmi_group <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Normal Weight	no	1053	36.03799	190.2267	13.792269	0.140
Obesity	no	3044	38.14947	200.8304	14.171465	0.406
Overweight	no	1866	37.31297	182.3685	13.504388	0.249
Underweight	no	91	32.10989	160.3878	12.664430	0.012
Normal Weight	yes	122	46.12295	100.0922	10.004609	0.016
Obesity	yes	1080	44.53611	204.8106	14.311204	0.144
Overweight	yes	237	47.67089	133.1539	11.539235	0.032
Underweight	yes	9	34.66667	16.7500	4.092676	0.001

8 rows

Hide

```
# plot
ggplot(bmi_group, aes(bmi_group, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



[Comments] The table and bar chart shows the detailed statistical results of two groups (high and low cost) in terms of bmi. 1. [Overall, Expensive - No] In the data set, people who are in 'Obesity' account for most of the population. We would say these people spent fewer costs on their healthcare. 2. [Expensive - Yes] People who are overweight may pay more costs for healthcare because the red area represents that there are 1,866 people (It accounts for 24.9% of the population)

Hide

```
# grouping (bmi_group ~ cost)
# table
bmi_group_cost <- df_new %>%
                    group_by(bmi_group, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'bmi_group'. You can override using the `.groups` argument.
```

Hide

```
bmi_group_cost <- bmi_group_cost %>% mutate(prop = round(total/30379292 ,3))
bmi_group_cost
```

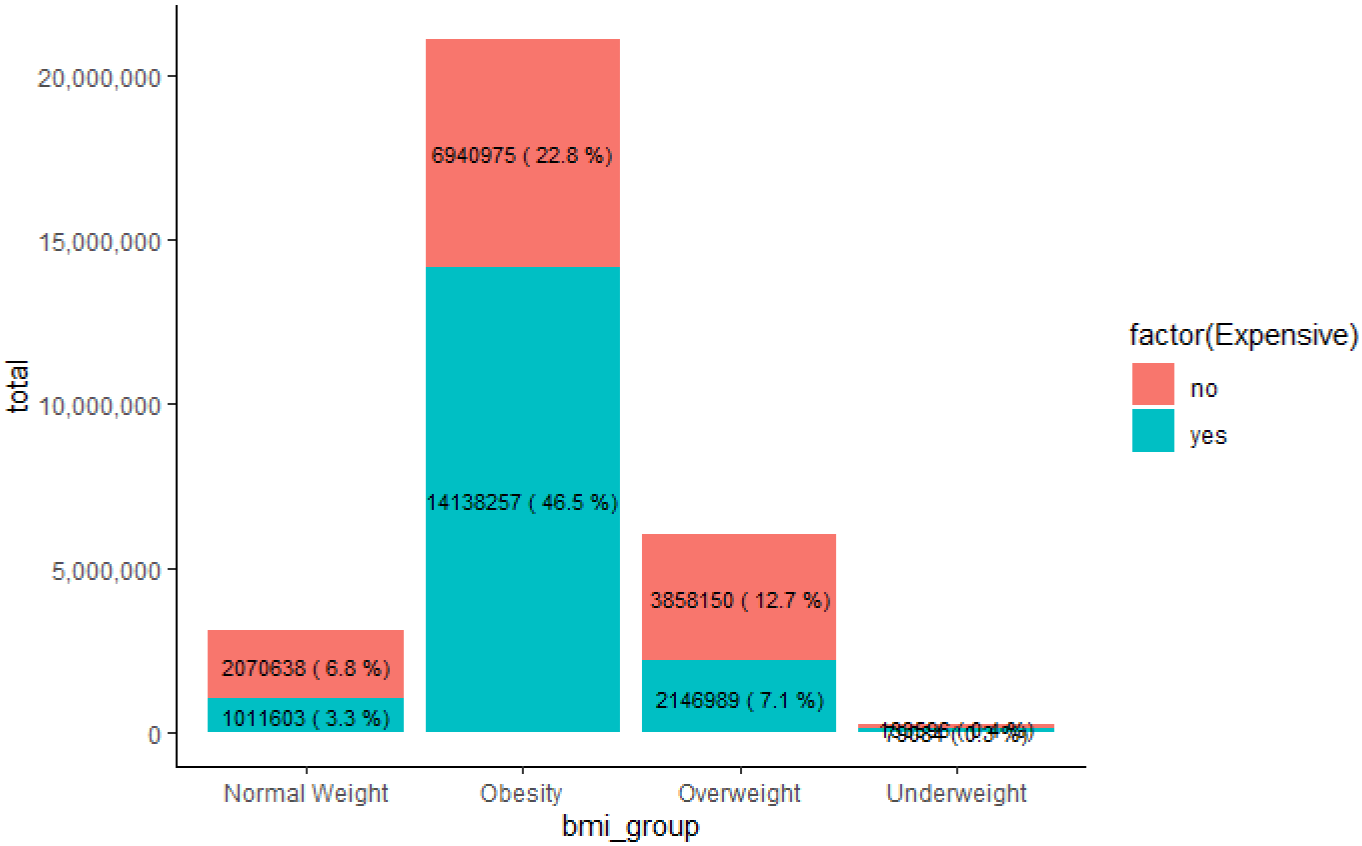
bmi_group <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
Normal Weight	no	2070638	1966.418	5986	2	2352227	1533.697	0.068

bmi_group	Expensive	total	mean	max	min	var	sd	prop
<chr>	<chr>	<int>	<dbl>	<int>	<int>	<dbl>	<dbl>	<dbl>
Obesity	no	6940975	2280.215	5968	5	2586689	1608.319	0.228
Overweight	no	3858150	2067.605	5975	12	2111717	1453.175	0.127
Underweight	no	133596	1468.088	5809	8	1880116	1371.173	0.004
Normal Weight	yes	1011603	8291.828	15360	6001	3993159	1998.289	0.033
Obesity	yes	14138257	13090.979	55715	6004	44180223	6646.820	0.465
Overweight	yes	2146989	9059.025	25738	6003	8475817	2911.326	0.071
Underweight	yes	79084	8787.111	11371	6319	2014046	1419.171	0.003

8 rows

Hide

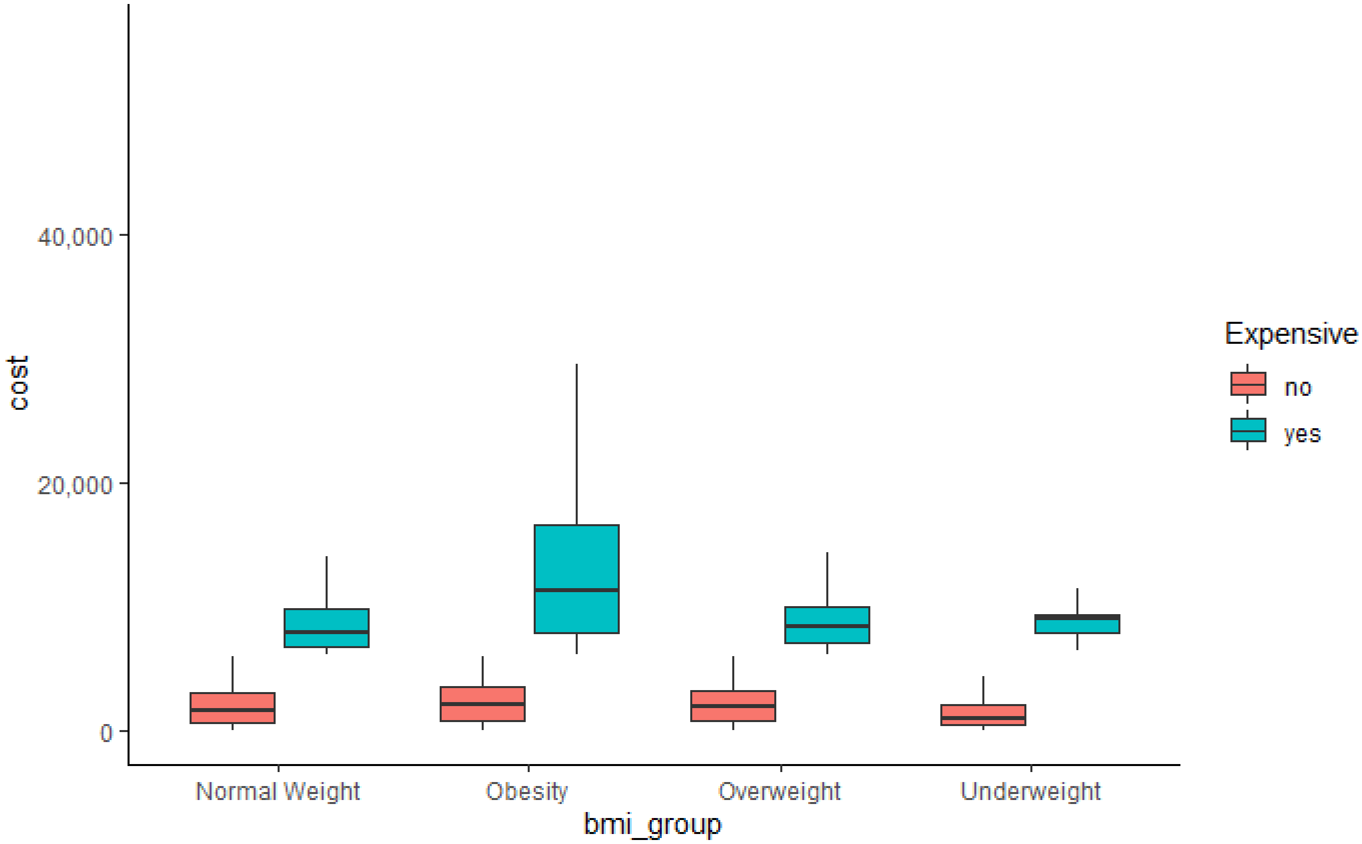
```
# plot
ggplot(bmi_group_cost, aes(bmi_group, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] Unlike the above results, as we consider the cost data together, the people in the Obesity group with both low and high healthcare costs spent a lot of money on healthcare. These groups also have the highest proportion in terms of costs.

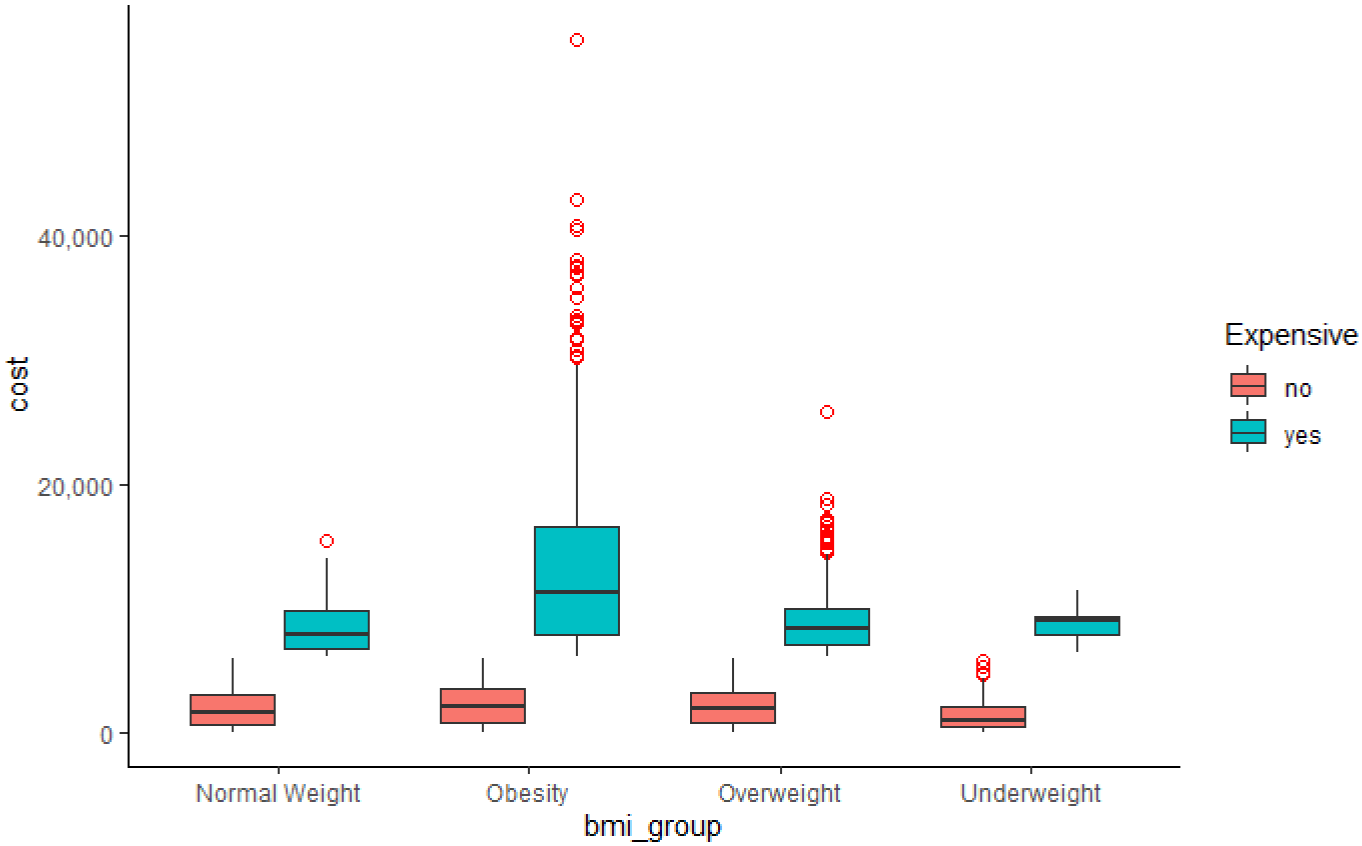
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(bmi_group, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(bmi_group, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] According to the boxplots with and without outliers, the Obesity group has a wide range of healthcare cost data, and there are many outliers in the Expensive-yes category.

3. location_type

Hide

```
# grouping (location_type ~ number of observations)
# table
location_type_group <- df_new %>%
  group_by(location_type, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'location_type'. You can override using the `.groups` argume
nt.
```

Hide

```
colnames(location_type_group)[3] <- "count"
location_type_group <- location_type_group %>% mutate(prop = round(count/7502, 3))
location_type_group
```

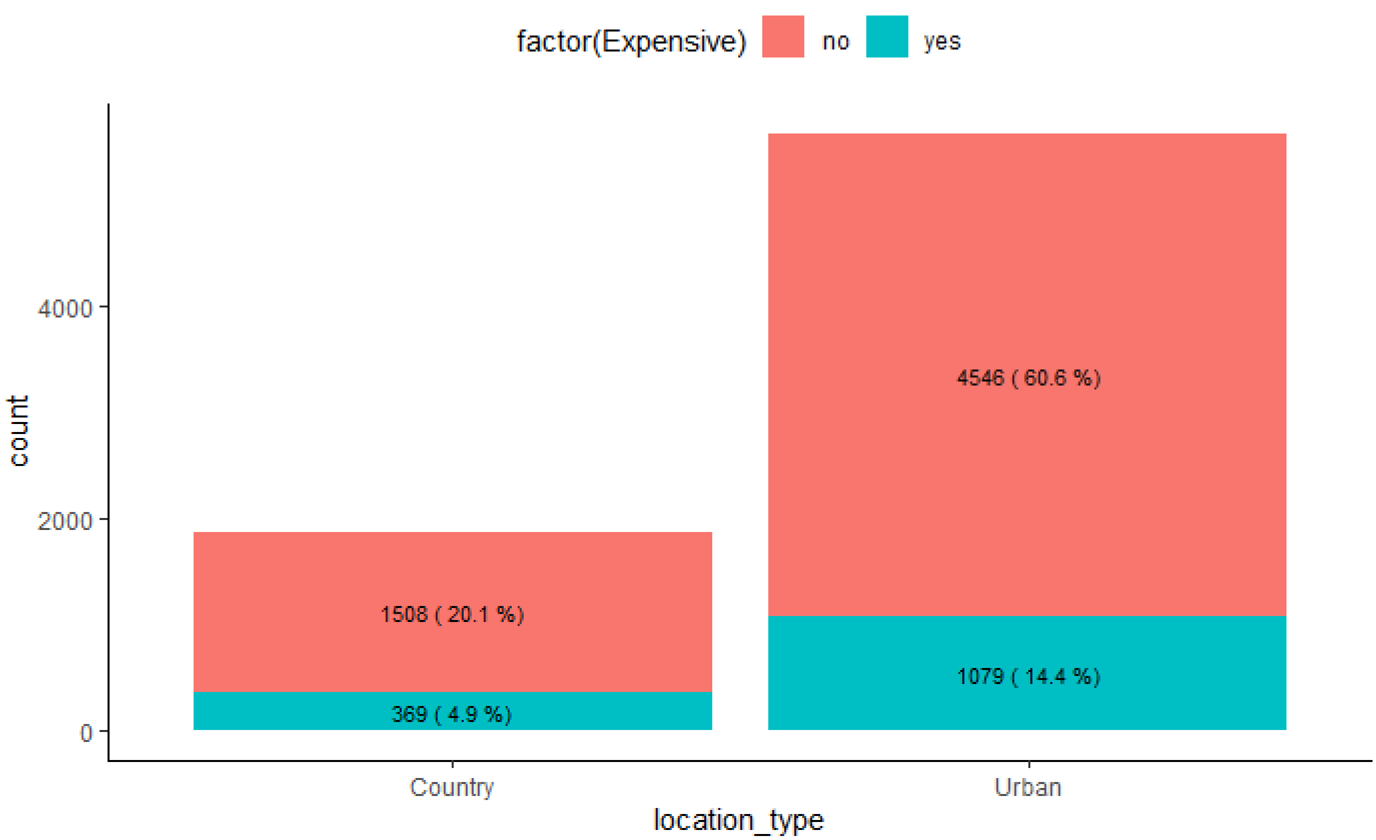
location_type <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Country	no	1508	37.78780	196.5959	14.02127	0.201

location_type	Expensive	count	mean	var	sd	prop
<chr>	<chr>	<int>	<dbl>	<dbl>	<dbl>	<dbl>
Urban	no	4546	37.31610	192.6277	13.87904	0.606
Country	yes	369	45.54472	202.7052	14.23746	0.049
Urban	yes	1079	44.97683	179.0282	13.38014	0.144

4 rows

Hide

```
# plot
ggplot(location_type_group, aes(location_type, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (location_type ~ cost)
# table
location_type_cost <- df_new %>%
                    group_by(location_type, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'location_type'. You can override using the `.groups` argume
nt.
```

Hide

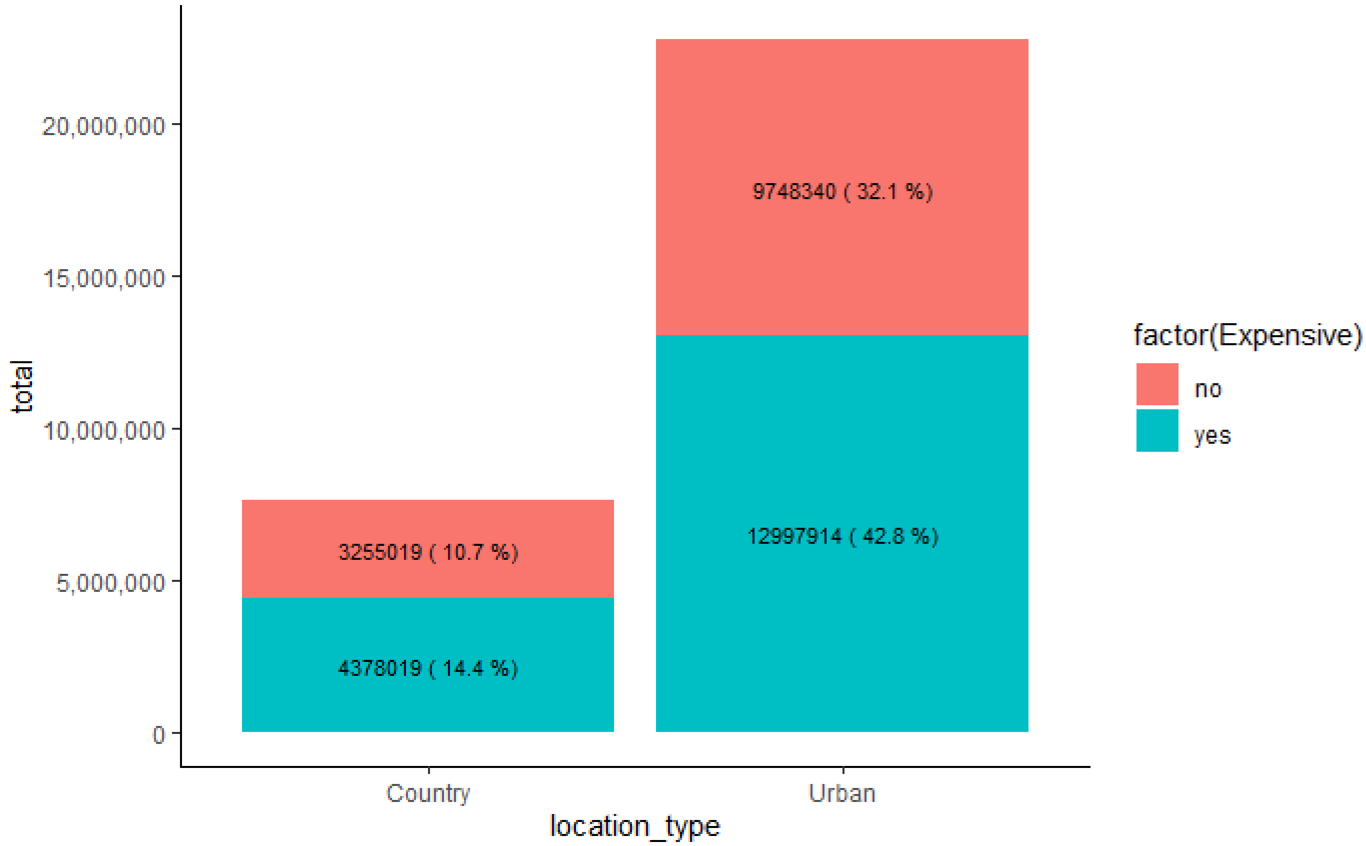
```
location_type_cost <- location_type_cost %>% mutate(prop = round(total/30379292 ,3))
location_type_cost
```

location_type <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
Country	no	3255019	2158.501	5986	4	2412796	1553.318	0.107
Urban	no	9748340	2144.377	5975	2	2411254	1552.821	0.321
Country	yes	4378019	11864.550	40388	6007	33492982	5787.312	0.144
Urban	yes	12997914	12046.259	55715	6001	39834282	6311.441	0.428

4 rows

Hide

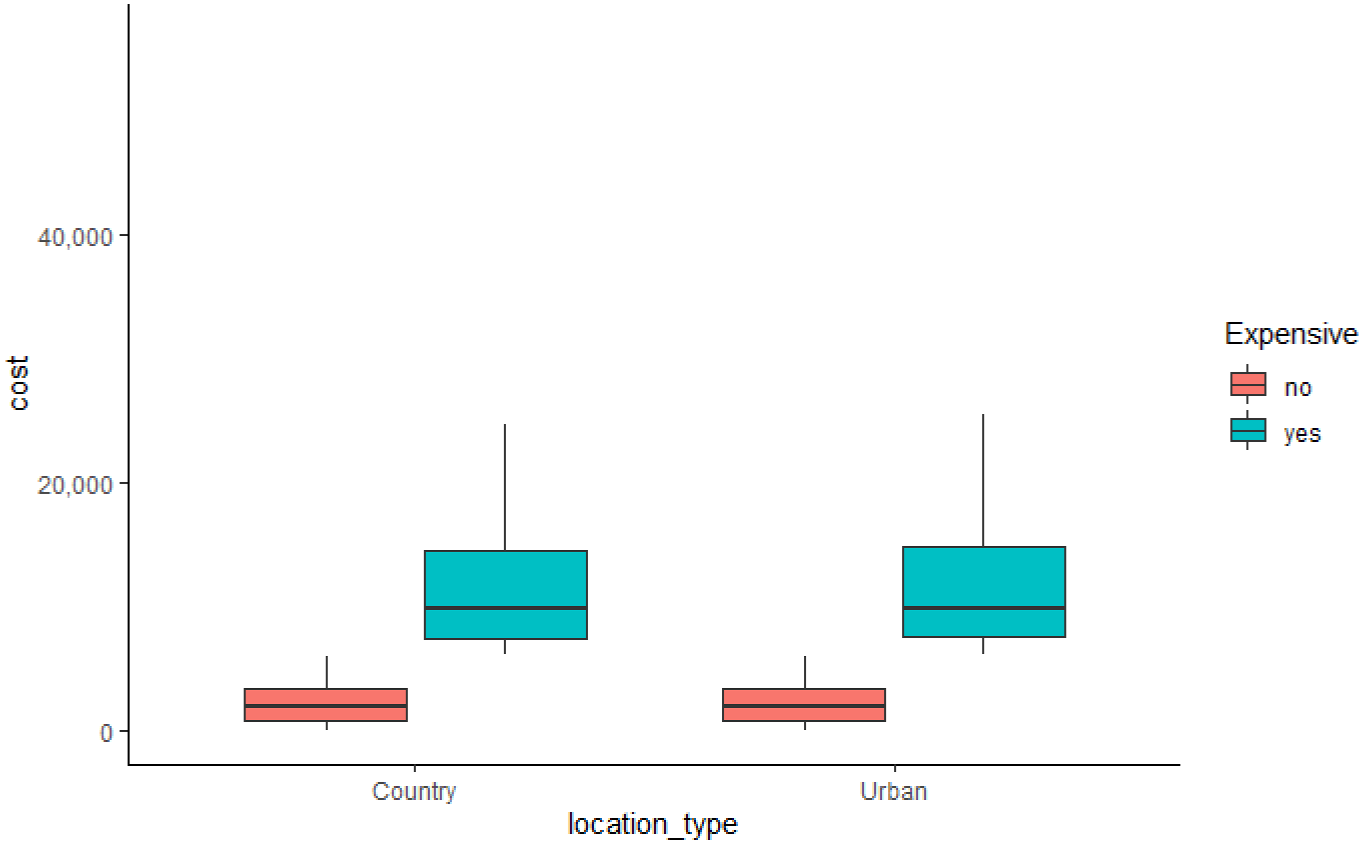
```
# plot
ggplot(location_type_cost, aes(location_type, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The location_type variable has a categorical data type, so we couldn't draw a histogram. The table and bar chart shows the detailed statistical results of two groups (high and low cost) in location_type categories. 1) In the data set, there are more people who live in urban areas in terms of both the number of observations and total healthcare costs. It accounts for almost 73-75% of the population.

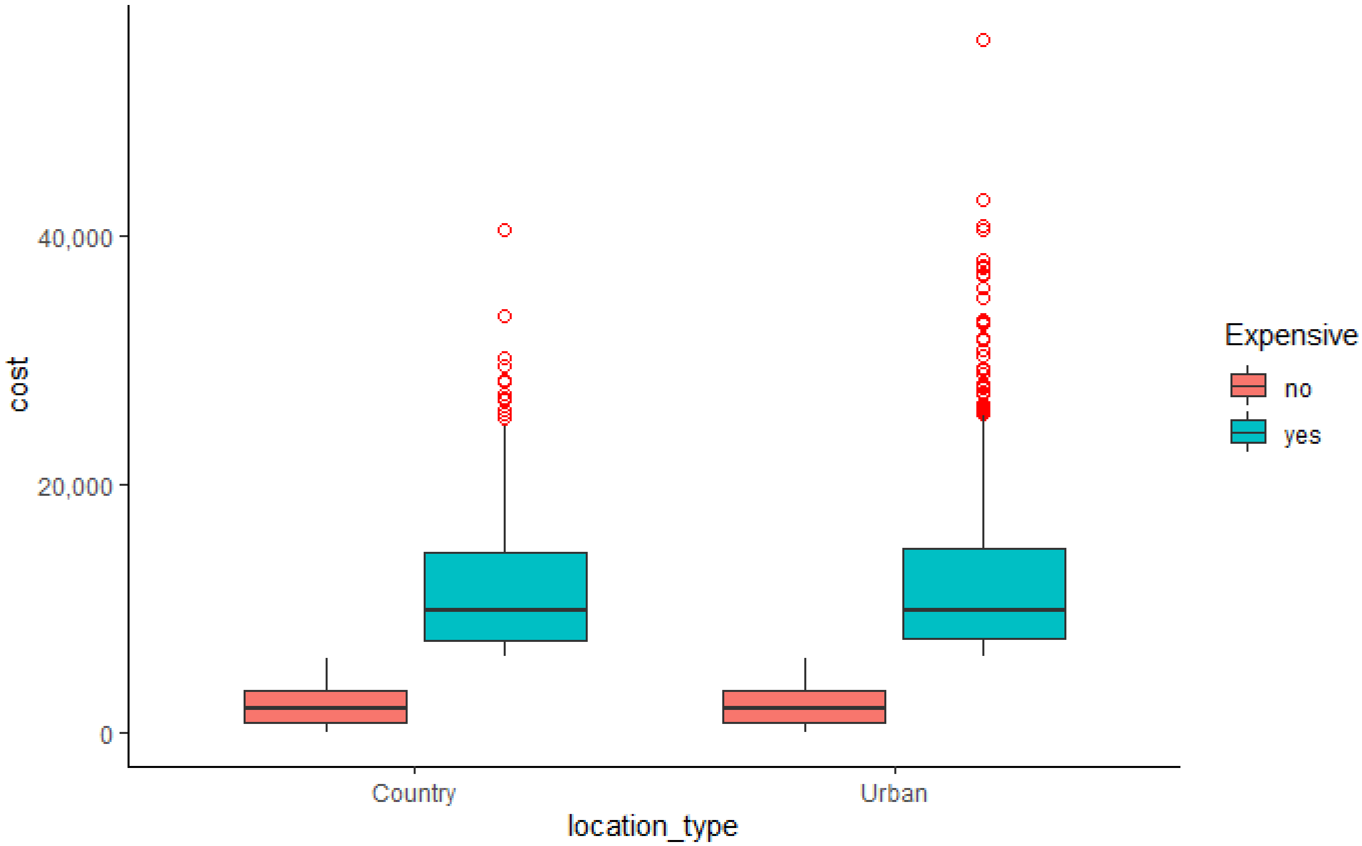
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(location_type, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(location_type, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] The boxplots that made by the location_type and cost columns for the 'Expensive-Yes' group has many outliers. Considering healthcare costs, the boxplots of both country and the urban group have almost similar shapes.

4. exercise

Hide

```
# grouping (exercise ~ numer of observation)
# table
exericse_group <- df_new %>%
  group_by(exercise, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'exercise'. You can override using the `.groups` argument.
```

Hide

```
colnames(exericse_group)[3] <- "count"
exericse_group <- exericse_group %>% mutate(prop = round(count/7502, 3))
exericse_group
```

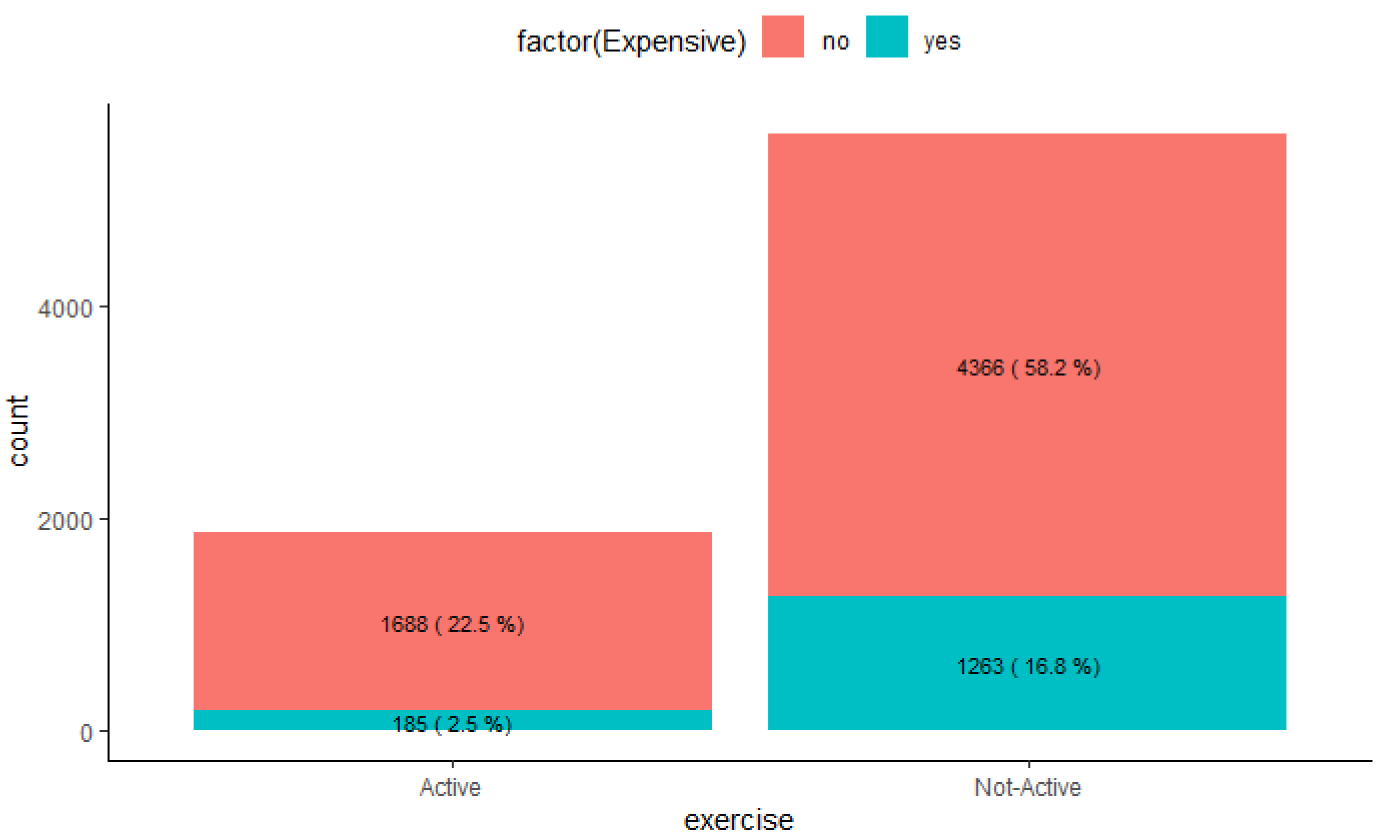
exercise <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Active	no	1688	38.25355	203.4567	14.26383	0.225

exercise <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Not-Active	no	4366	37.11658	189.5097	13.76625	0.582
Active	yes	185	45.20000	175.4000	13.24387	0.025
Not-Active	yes	1263	45.11006	186.5307	13.65762	0.168

4 rows

Hide

```
# plot
ggplot(exericse_group, aes(exercise, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (exercise ~ cost)
# table
exercise_group_cost <- df_new %>%
                    group_by(exercise, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'exercise'. You can override using the `.groups` argument.
```

Hide

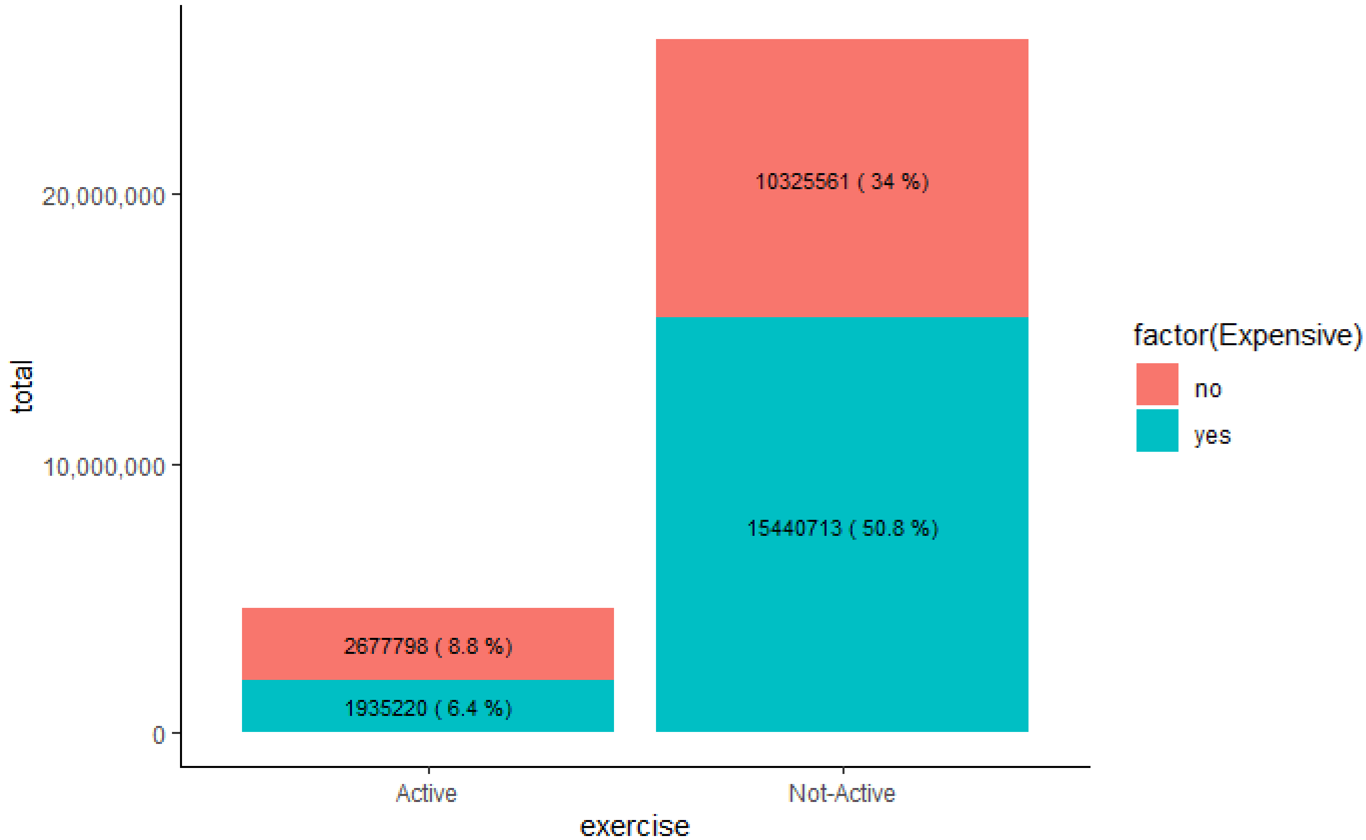
```
exercise_group_cost <- exercise_group_cost %>% mutate(prop = round(total/30379292 ,3))
exercise_group_cost
```

exercise <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
Active	no	2677798	1586.373	5938	2	1855226	1362.067	0.088
Not-Active	no	10325561	2364.993	5986	97	2457658	1567.692	0.340
Active	yes	1935220	10460.649	28219	6035	17120361	4137.676	0.064
Not-Active	yes	15440713	12225.426	55715	6001	40905822	6395.766	0.508

4 rows

Hide

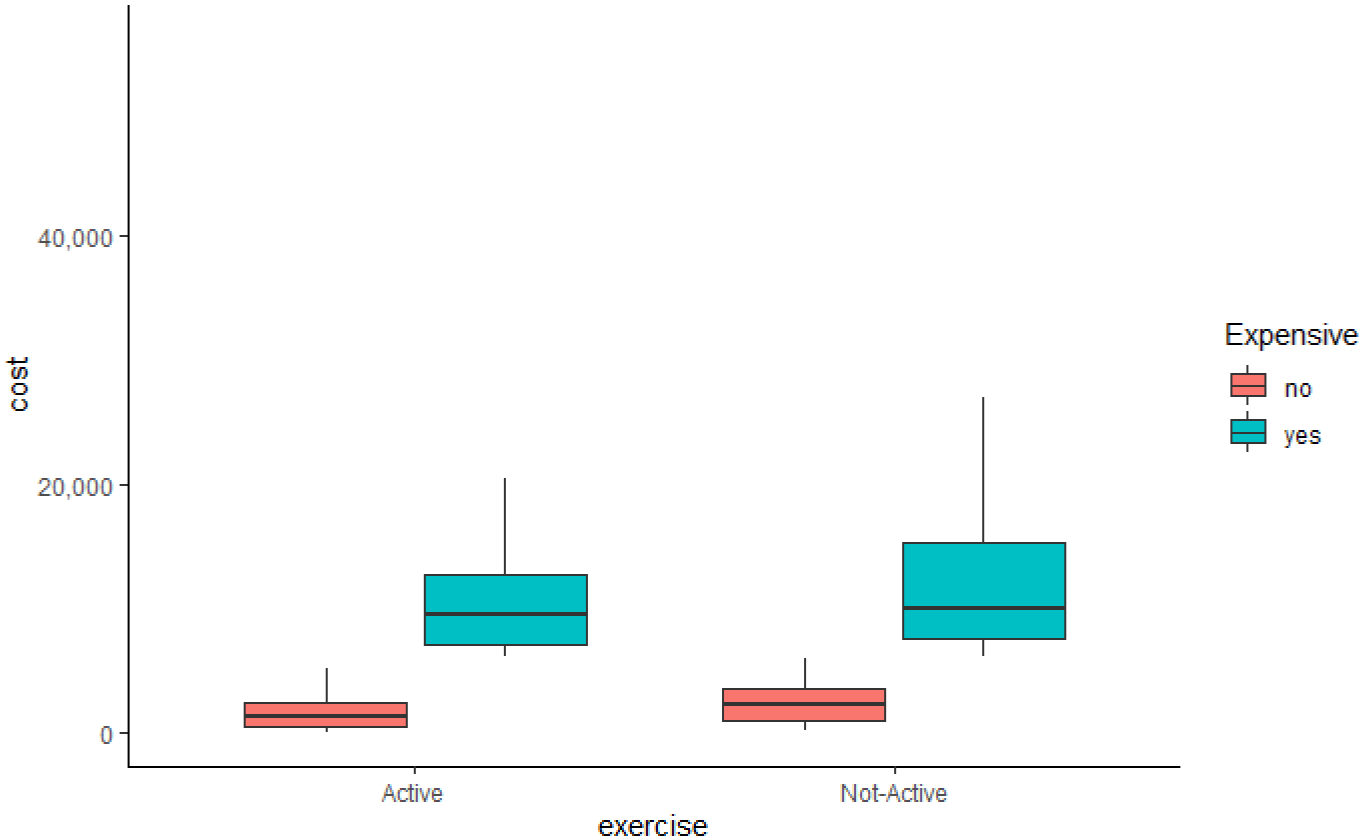
```
# plot
ggplot(exercise_group_cost, aes(exercise, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The exercise variable has a categorical data type, so we couldn't draw a histogram. The table and bar chart shows the detailed statistical results of two groups (high and low cost) in the exercise categories. 1) In the data set, the number of people who exercise regularly is more than the other group without working out. Also, they have a higher healthcare cost. It accounts for almost 84-85% of the population. 2) The interesting point is that even though there are more people who are not active and have a lower healthcare spending, the actual costs of the people who are not active and have a higher healthcare cost are higher than the other group.

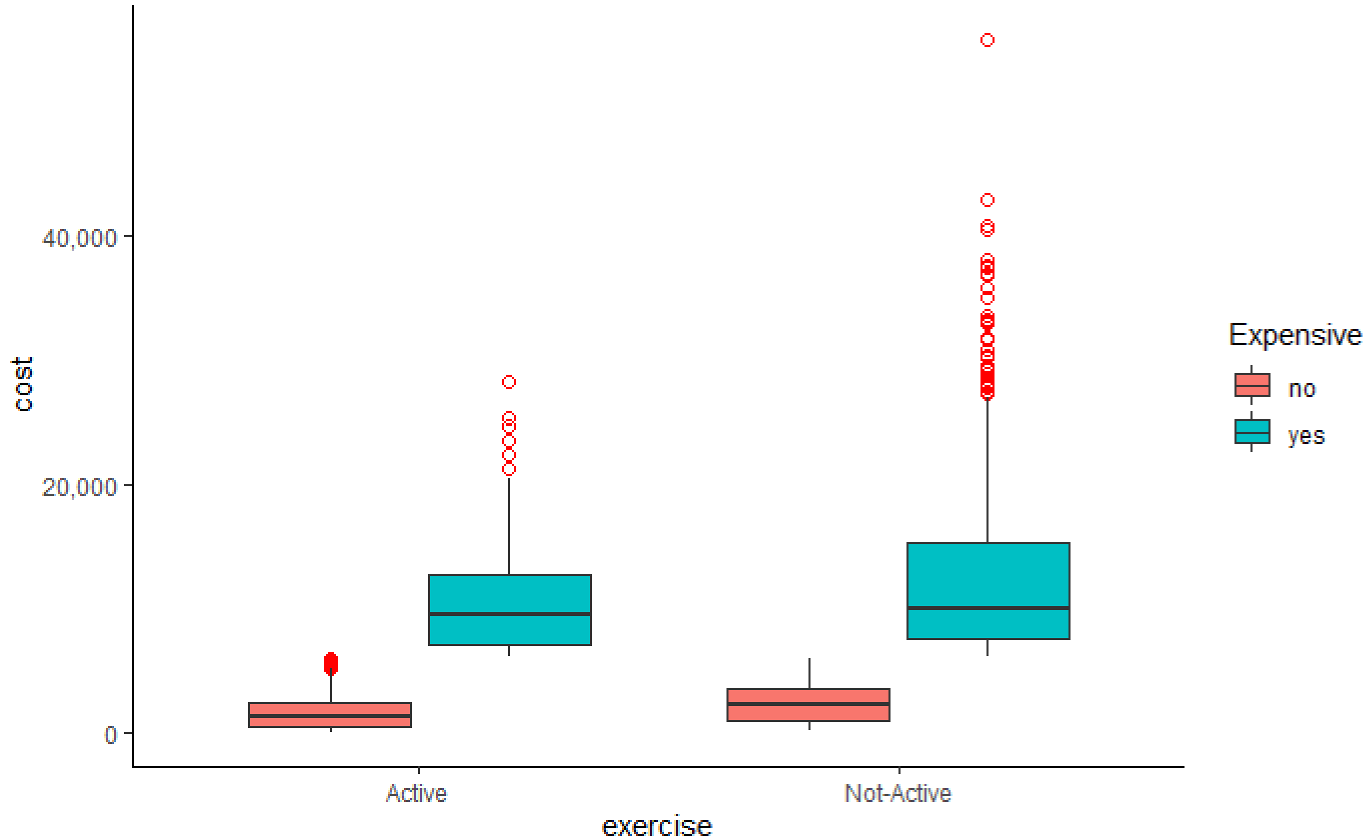
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(exercise, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(exercise, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] There are many outliers in the not-active and expensive - yes group.

5. smoker

Hide

```
# grouping (smoker ~ numer of observation)
# table
smoker_group <- df_new %>%
  group_by(smoker, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'smoker'. You can override using the `.groups` argument.
```

Hide

```
colnames(smoker_group)[3] <- "count"
smoker_group <- smoker_group %>% mutate(prop = round(count/7502, 3))
smoker_group
```

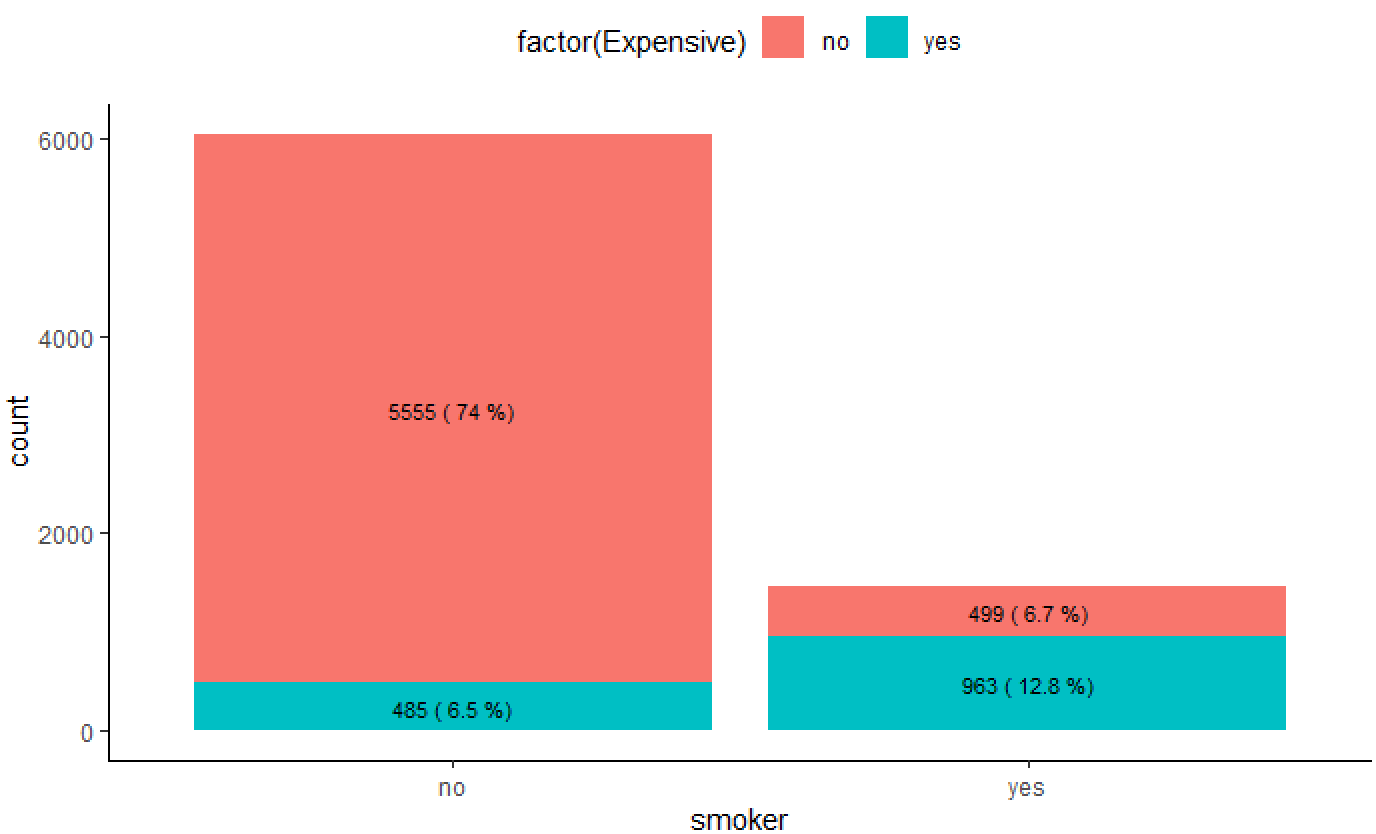
smoker <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
no	no	5555	37.80414	193.3098	13.90359	0.740
yes	no	499	33.30862	178.9528	13.37732	0.067

smoker <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
no	yes	485	50.70928	142.5951	11.94132	0.065
yes	yes	963	42.30737	182.8389	13.52179	0.128

4 rows

Hide

```
# plot
ggplot(smoker_group, aes(smoker, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (smoker ~ cost)
# table
smoker_group_cost <- df_new %>%
                    group_by(smoker, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'smoker'. You can override using the `.groups` argument.
```

Hide

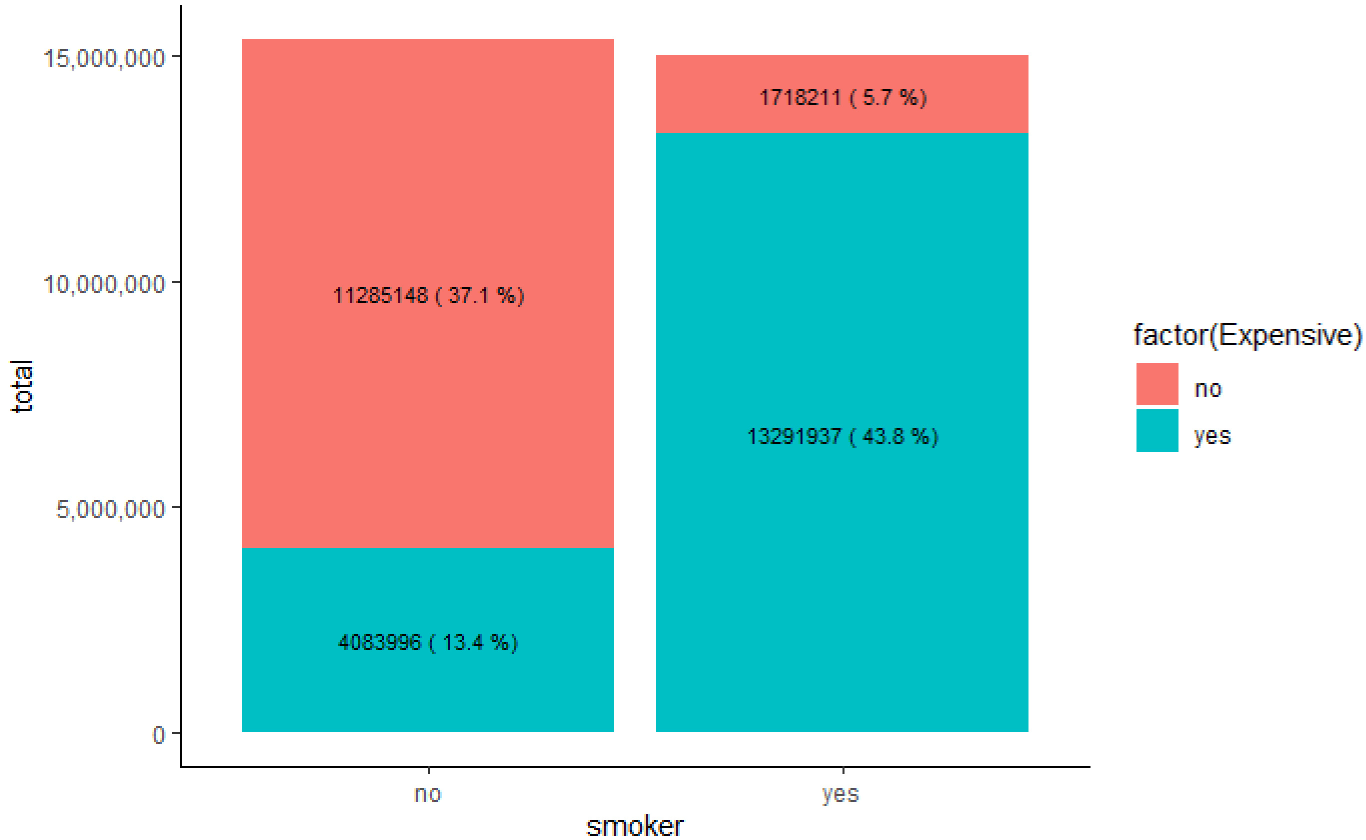
```
smoker_group_cost <- smoker_group_cost %>% mutate(prop = round(total/30379292 ,3))
smoker_group_cost
```

smoker	Expensive	total	mean	max	min	var	sd	prop
<chr>	<chr>	<int>	<dbl>	<int>	<int>	<dbl>	<dbl>	<dbl>
no	no	11285148	2031.530	5968	2	2270763	1506.905	0.371
yes	no	1718211	3443.309	5986	78	2150712	1466.530	0.057
no	yes	4083996	8420.610	31542	6003	6323670	2514.691	0.134
yes	yes	13291937	13802.634	55715	6001	44565594	6675.747	0.438

4 rows

Hide

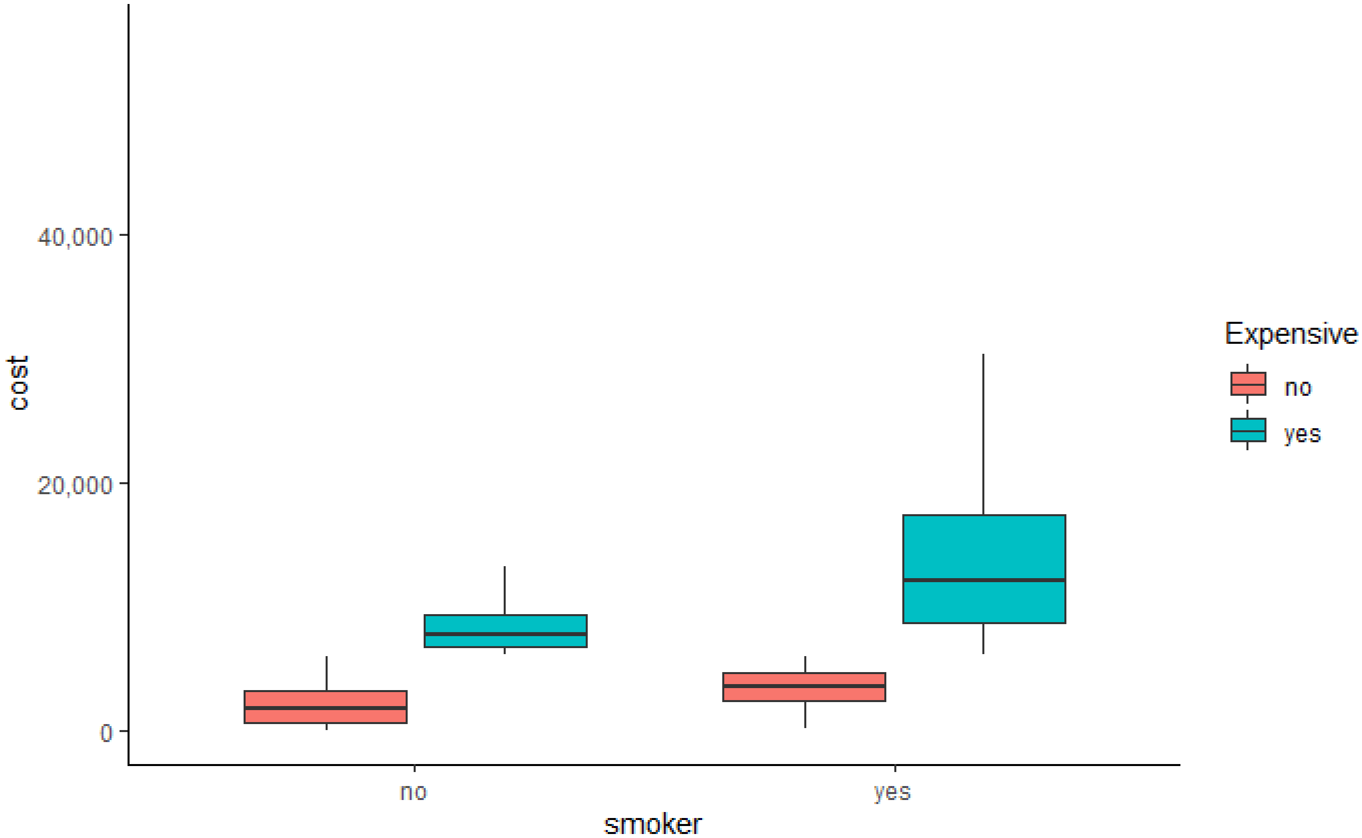
```
# plot
ggplot(smoker_group_cost, aes(smoker, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The smoker variable has a categorical data type, so we couldn't draw a histogram. The table and bar chart shows the detailed statistical results of two groups (high and low cost) in the smoker categories. 1) In the data set, the number of people who smoke is more than the non-smoker group. It accounts for almost 84-85% of the population.

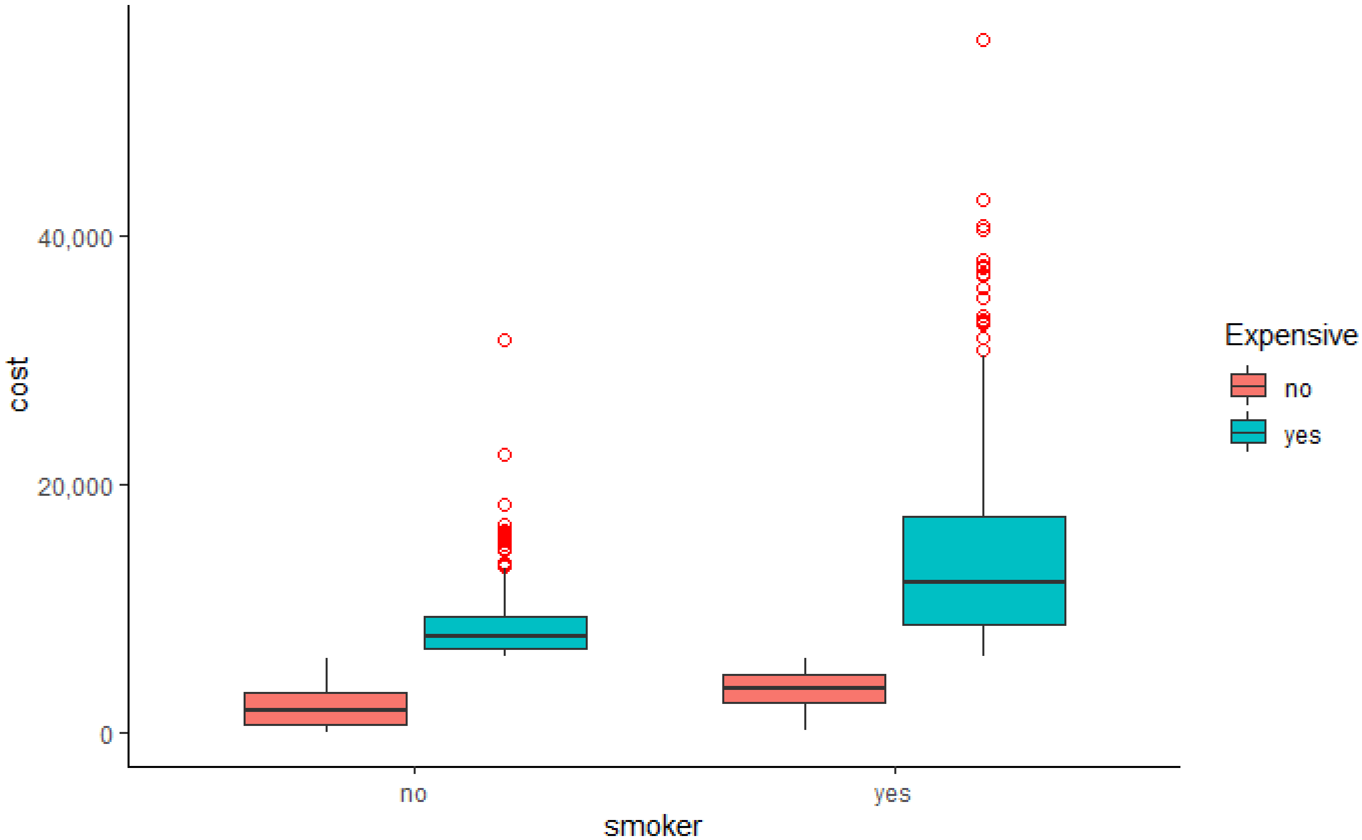
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(smoker, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(smoker, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] There are many outliers in the smoker - yes and expensive - yes group. It also has a variable range of healthcare cost data.

6. yearly_physical

Hide

```
# grouping (yearly_physical ~ numer of observation)
# table
yearly_physical_group <- df_new %>%
  group_by(yearly_physical, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'yearly_physical'. You can override using the `.groups` argu
ment.
```

Hide

```
colnames(yearly_physical_group)[3] <- "count"
yearly_physical_group <- yearly_physical_group %>% mutate(prop = round(count/7502, 3))
yearly_physical_group
```

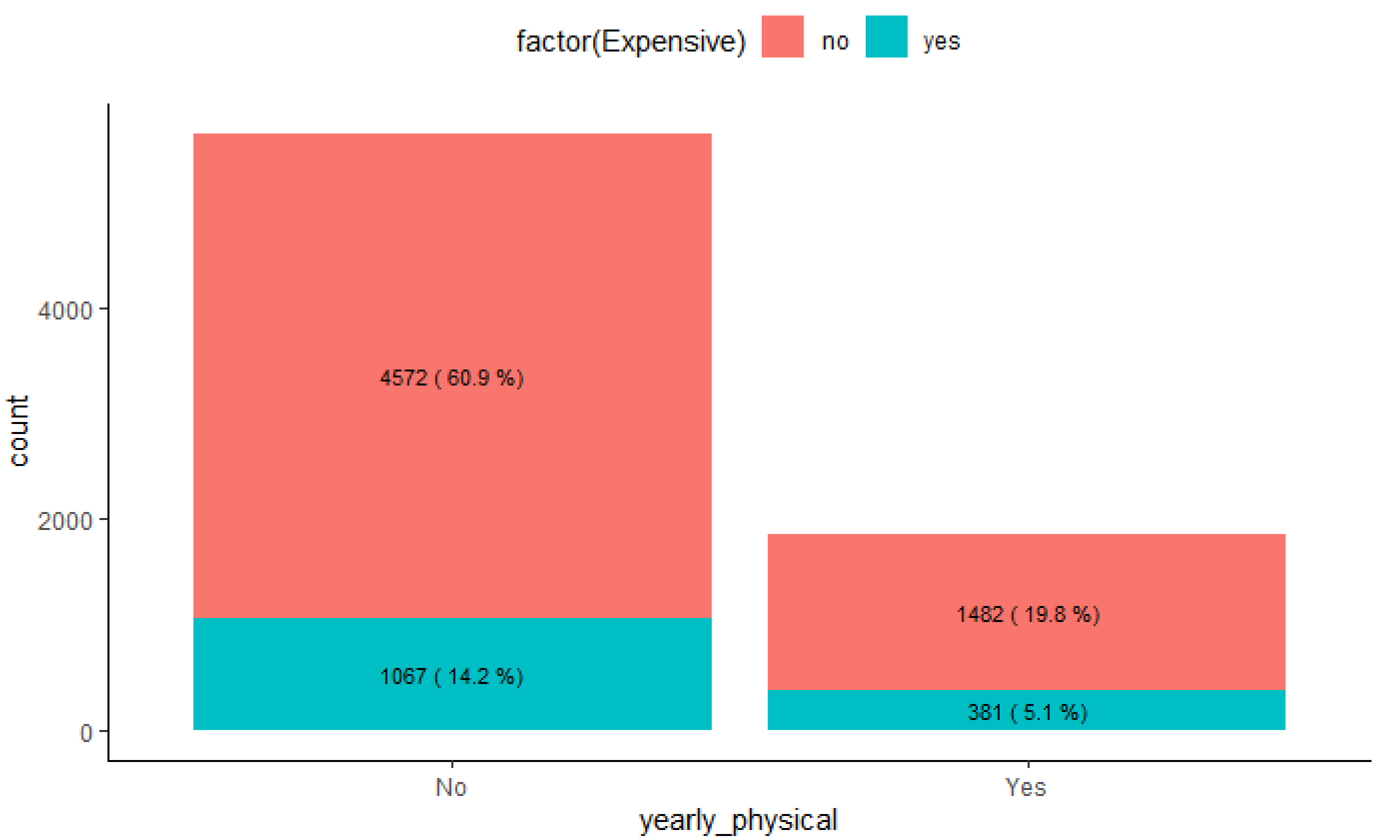
yearly_physical <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
No	no	4572	37.44641	193.0527	13.89434	0.609

yearly_physical <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Yes	no	1482	37.39406	195.5219	13.98291	0.198
No	yes	1067	45.60825	187.4036	13.68954	0.142
Yes	yes	381	43.75853	176.1679	13.27282	0.051

4 rows

Hide

```
# plot
ggplot(yearly_physical_group, aes(yearly_physical, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (yearly_physical ~ cost)
# table
yearly_physical_group_cost <- df_new %>%
                    group_by(yearly_physical, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'yearly_physical'. You can override using the `.groups` argu
ment.
```

Hide

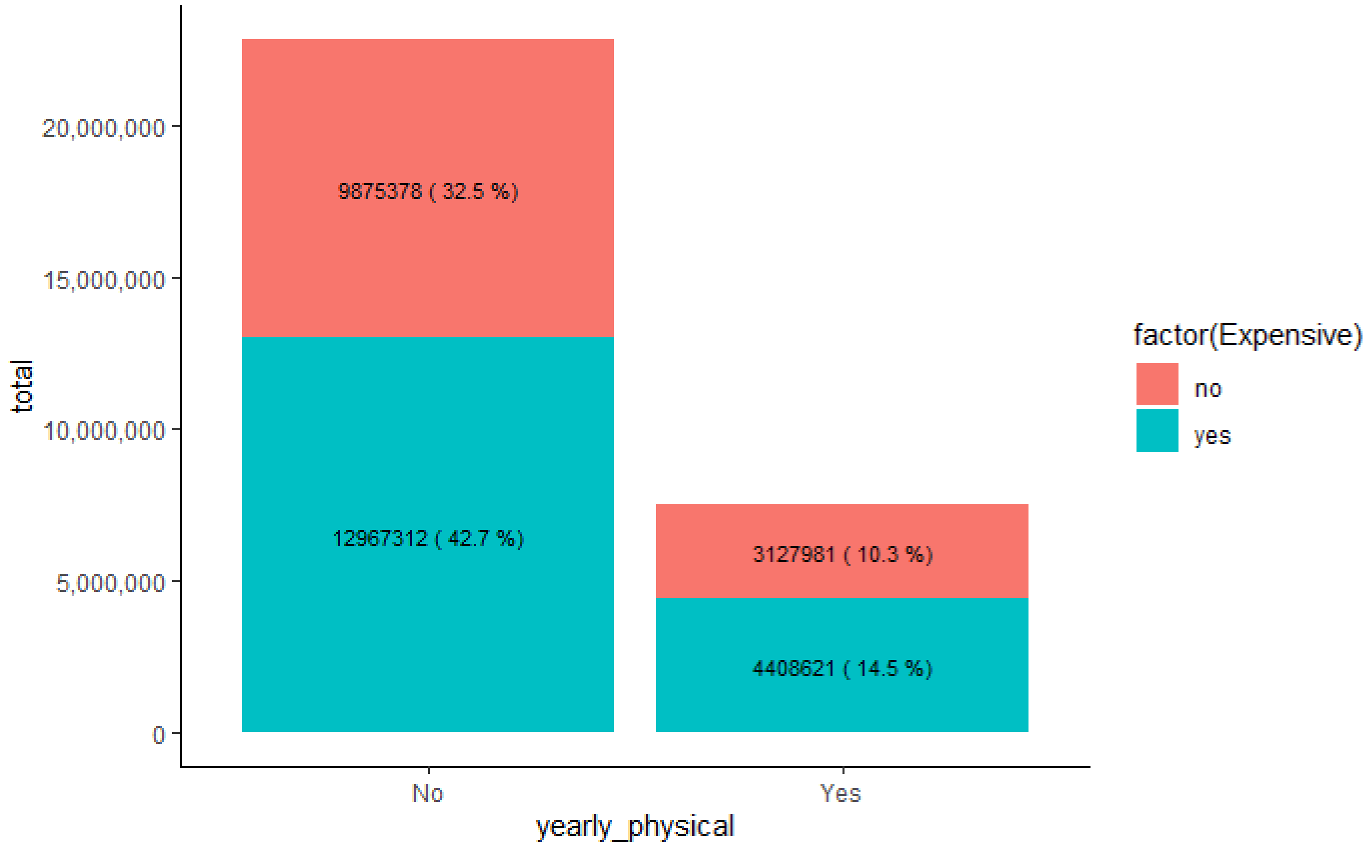
```
yearly_physical_group_cost <- yearly_physical_group_cost %>% mutate(prop = round(total/30379292
,3))
yearly_physical_group_cost
```

yearly_physical <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
No	no	9875378	2159.969	5975	2	2408989	1552.092	0.325
Yes	no	3127981	2110.648	5986	19	2418127	1555.033	0.103
No	yes	12967312	12153.057	55715	6001	41697438	6457.355	0.427
Yes	yes	4408621	11571.184	30334	6003	28240337	5314.164	0.145

4 rows

Hide

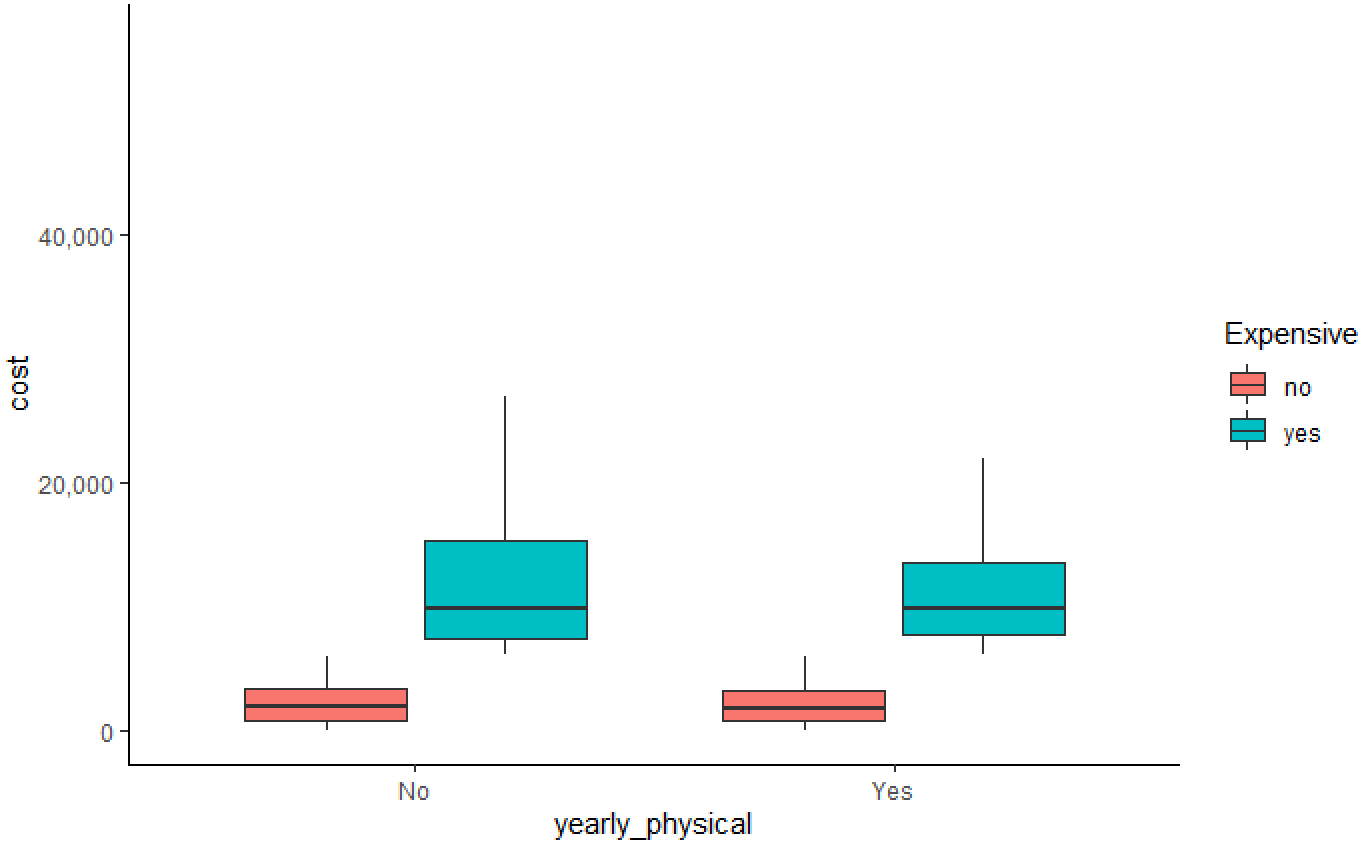
```
# plot
ggplot(yearly_physical_group_cost, aes(yearly_physical, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The table and bar chart shows the detailed statistical results of two groups (high and low cost) in the yearly_physical categories. 1) In the data set, there are more people who if the person had a well visit with their doctor during the year in terms of both the number of observations and total healthcare costs. It accounts for almost 75% of the population. 2) The interesting point is that people who usually didn't see their doctor for a year have a higher healthcare cost.

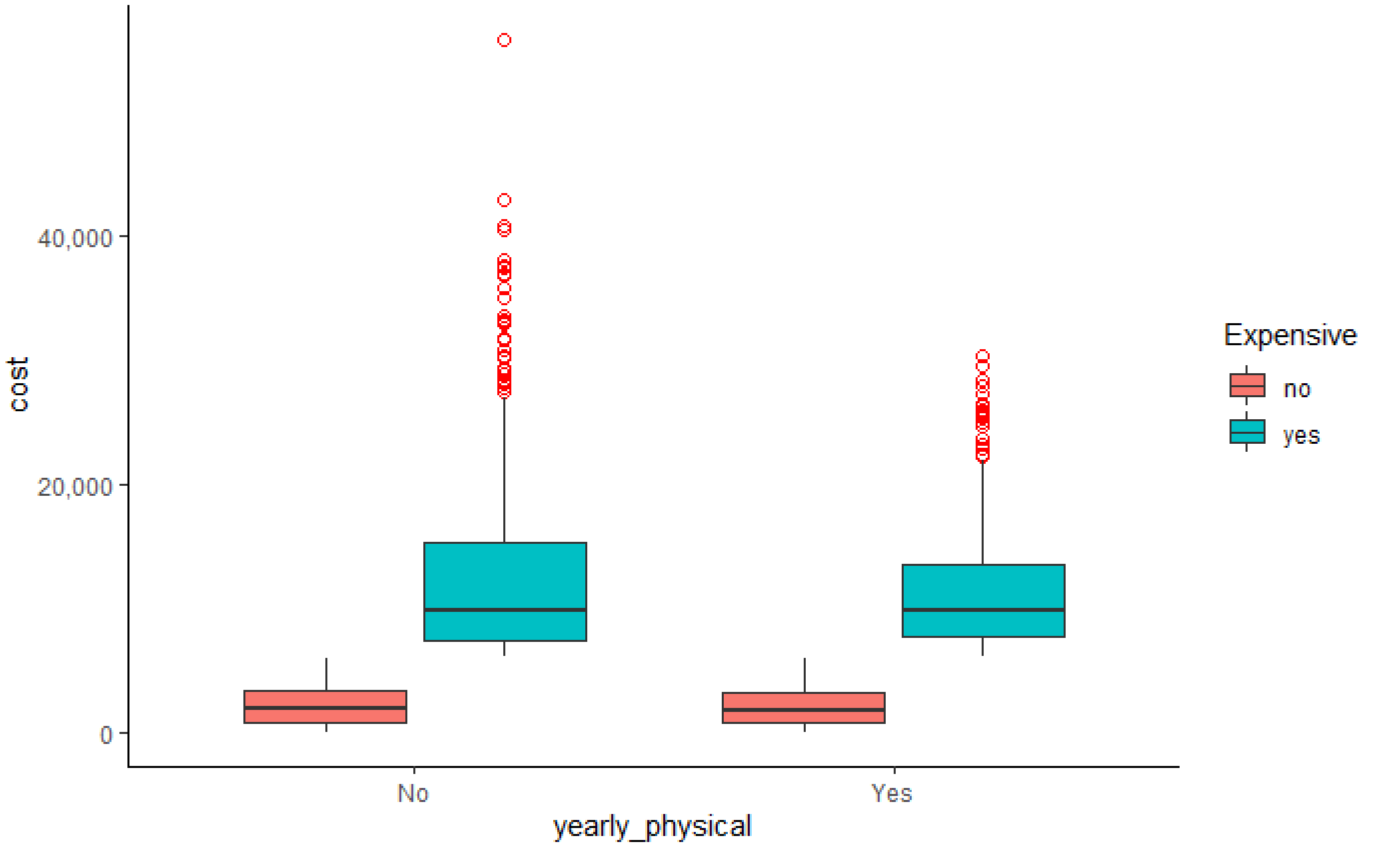
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(yearly_physical, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(yearly_physical, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] Even though the boxplots don't have a wider range of data on healthcare costs, there are many outliers in the expensive-yes group (green boxes).

7. gender

Hide

```
# grouping (gender ~ numer of observation)
# table
gender_group <- df_new %>%
  group_by(gender, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'gender'. You can override using the `.groups` argument.
```

Hide

```
colnames(gender_group)[3] <- "count"
gender_group <- gender_group %>% mutate(prop = round(count/7502, 3))
gender_group
```

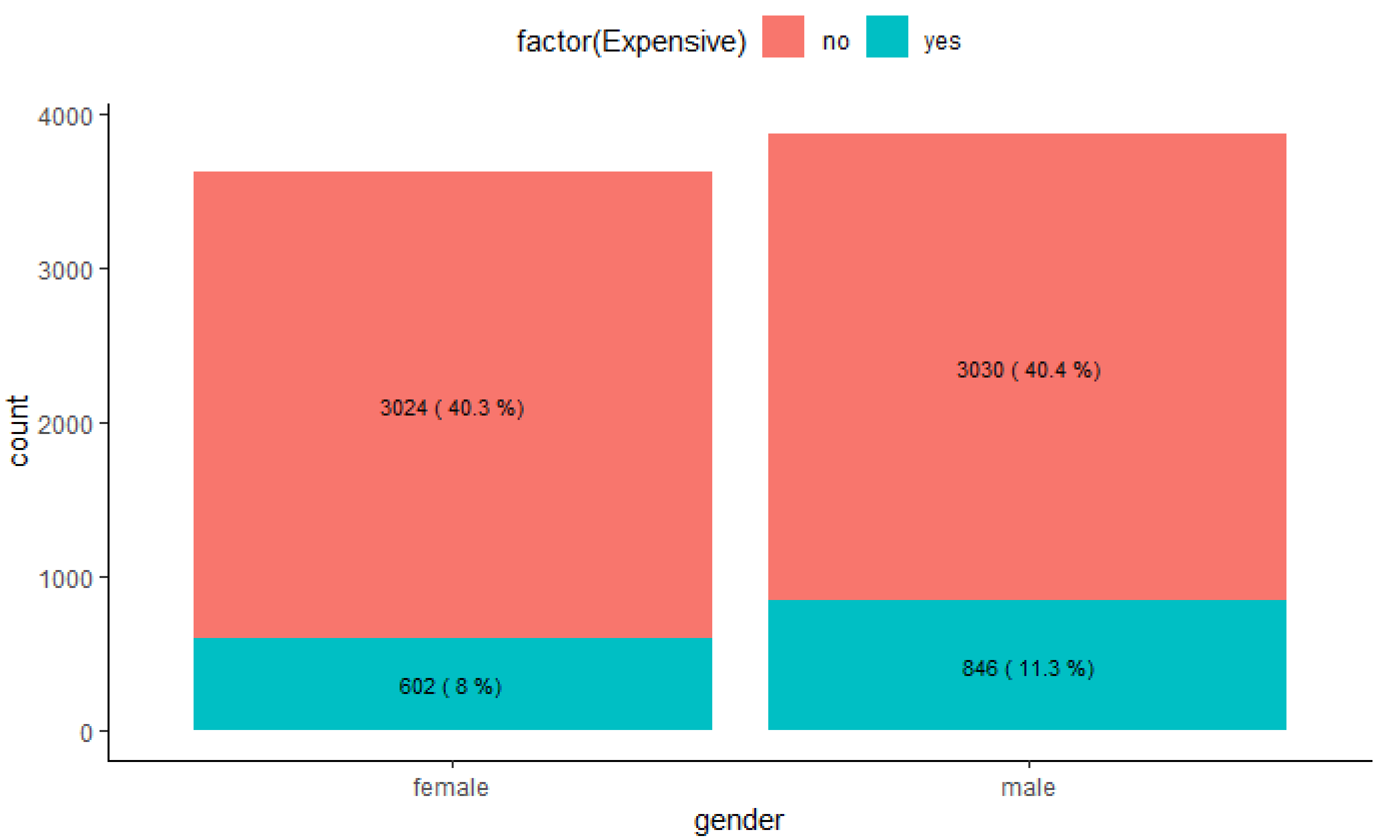
gender <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
female	no	3024	37.68585	197.5711	14.05600	0.403

gender <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
male	no	3030	37.18185	189.6246	13.77042	0.404
female	yes	602	46.70100	183.9238	13.56185	0.080
male	yes	846	43.99764	182.9207	13.52482	0.113

4 rows

Hide

```
# plot
ggplot(gender_group, aes(gender, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (gender ~ cost)
# table
gender_group_cost <- df_new %>%
                    group_by(gender, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'gender'. You can override using the `.groups` argument.
```

Hide

```
gender_group_cost <- gender_group_cost %>% mutate(prop = round(total/30379292 ,3))
gender_group_cost
```

gender <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
female	no	6570059	2172.639	5986	4	2296344	1515.369	0.216
male	no	6433300	2123.201	5968	2	2525557	1589.200	0.212
female	yes	6965779	11571.061	55715	6001	37587393	6130.856	0.229
male	yes	10410154	12305.147	42820	6007	38457152	6201.383	0.343

4 rows

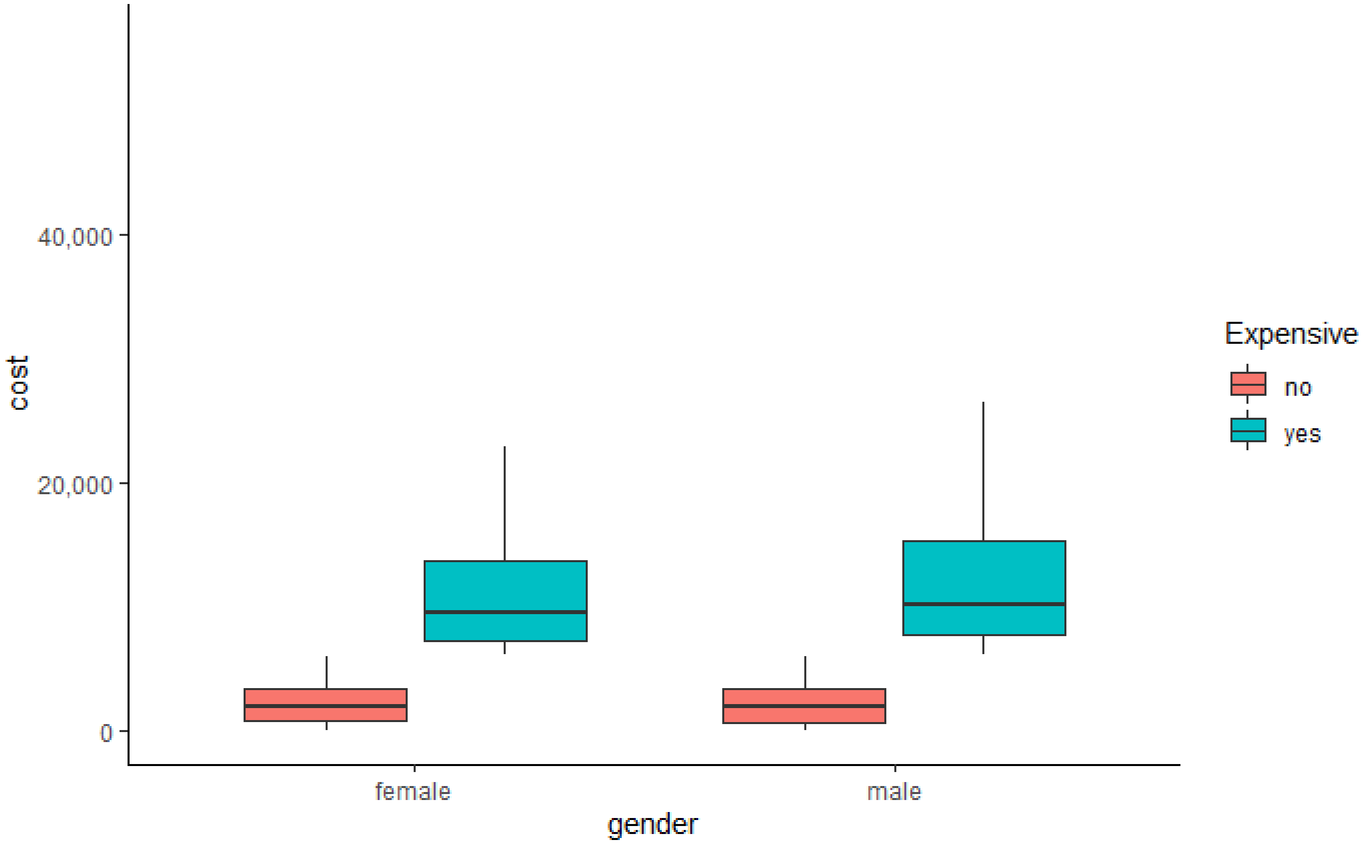
Hide

```
# plot
ggplot(gender_group_cost, aes(gender, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```

[Comments] The table and bar chart shows the detailed statistical results of two groups (high and low cost) in terms of gender. 1. There is no significant difference between the number of observations in female and male groups. 2. However, in terms of healthcare cost, male has a higher healthcare cost than the female.

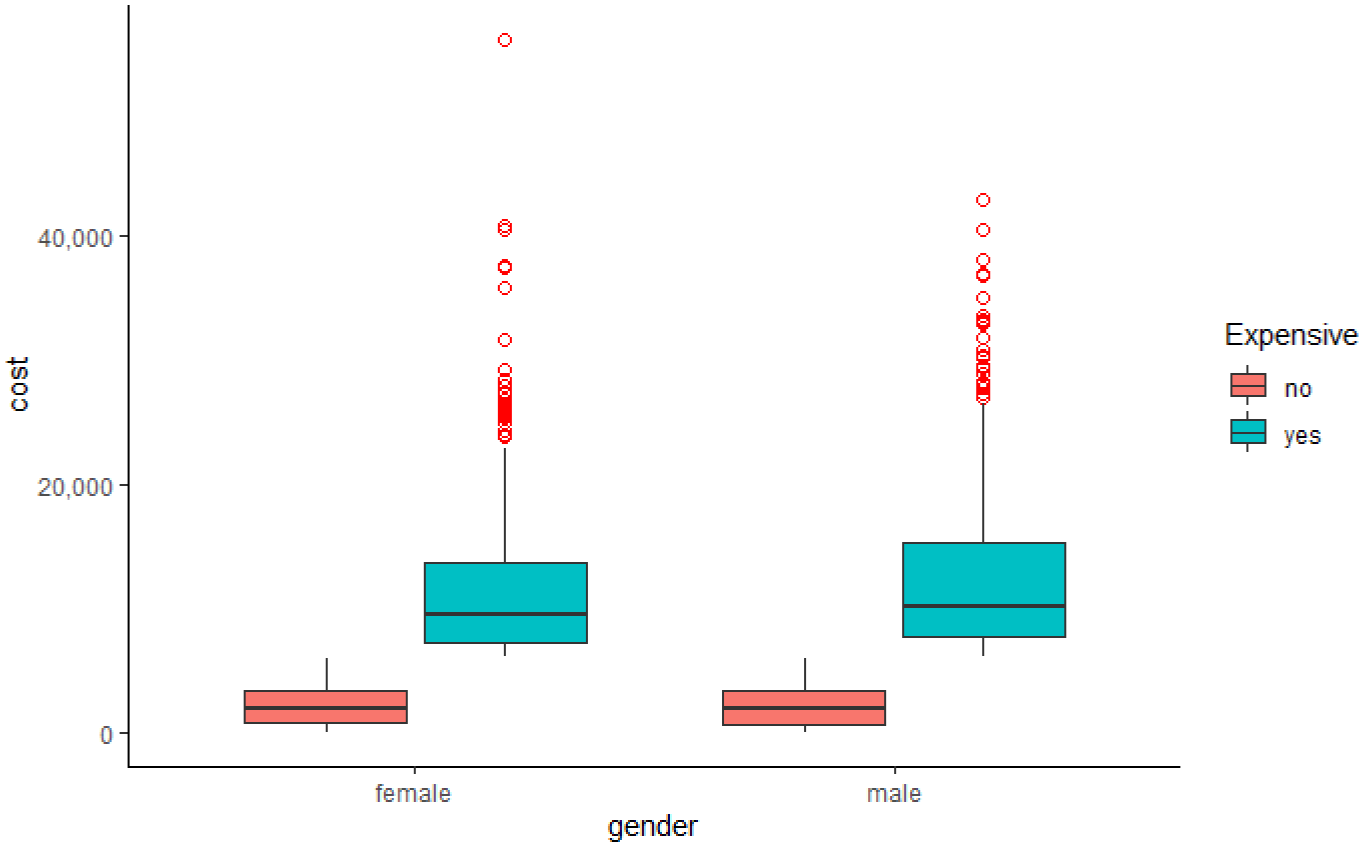
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(gender, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(gender, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] There is no significant difference in both female and male boxplots with healthcare costs.

8. education_level - is_educated

Hide

```
# grouping (education_level ~ numer of observation)
# table
education_level_group <- df_new %>%
  group_by(is_educated, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'is_educated'. You can override using the `.groups` argumen
t.
```

Hide

```
colnames(education_level_group)[3] <- "count"
education_level_group <- education_level_group %>% mutate(prop = round(count/7502, 3))
education_level_group
```

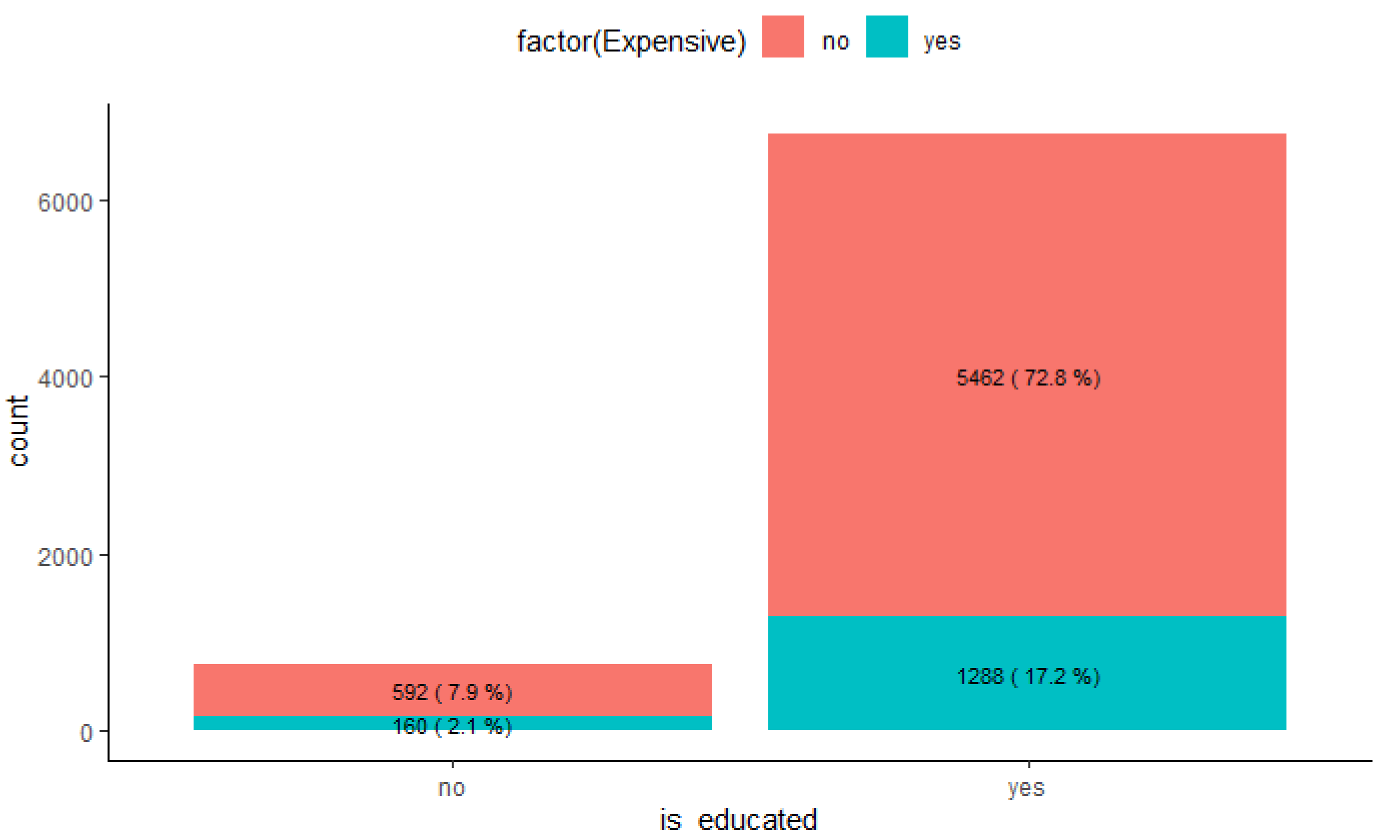
is_educated <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
no	no	592	37.22635	198.7913	14.09934	0.079

is_educated <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
yes	no	5462	37.45606	193.0967	13.89592	0.728
no	yes	160	46.37500	199.7075	14.13179	0.021
yes	yes	1288	44.96584	183.0928	13.53118	0.172

4 rows

Hide

```
# plot
ggplot(education_level_group, aes(is_educated, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (education_level ~ cost)
# table
education_level_group_cost <- df_new %>%
                    group_by(is_educated, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'is_educated'. You can override using the `.groups` argumen
t.
```

Hide

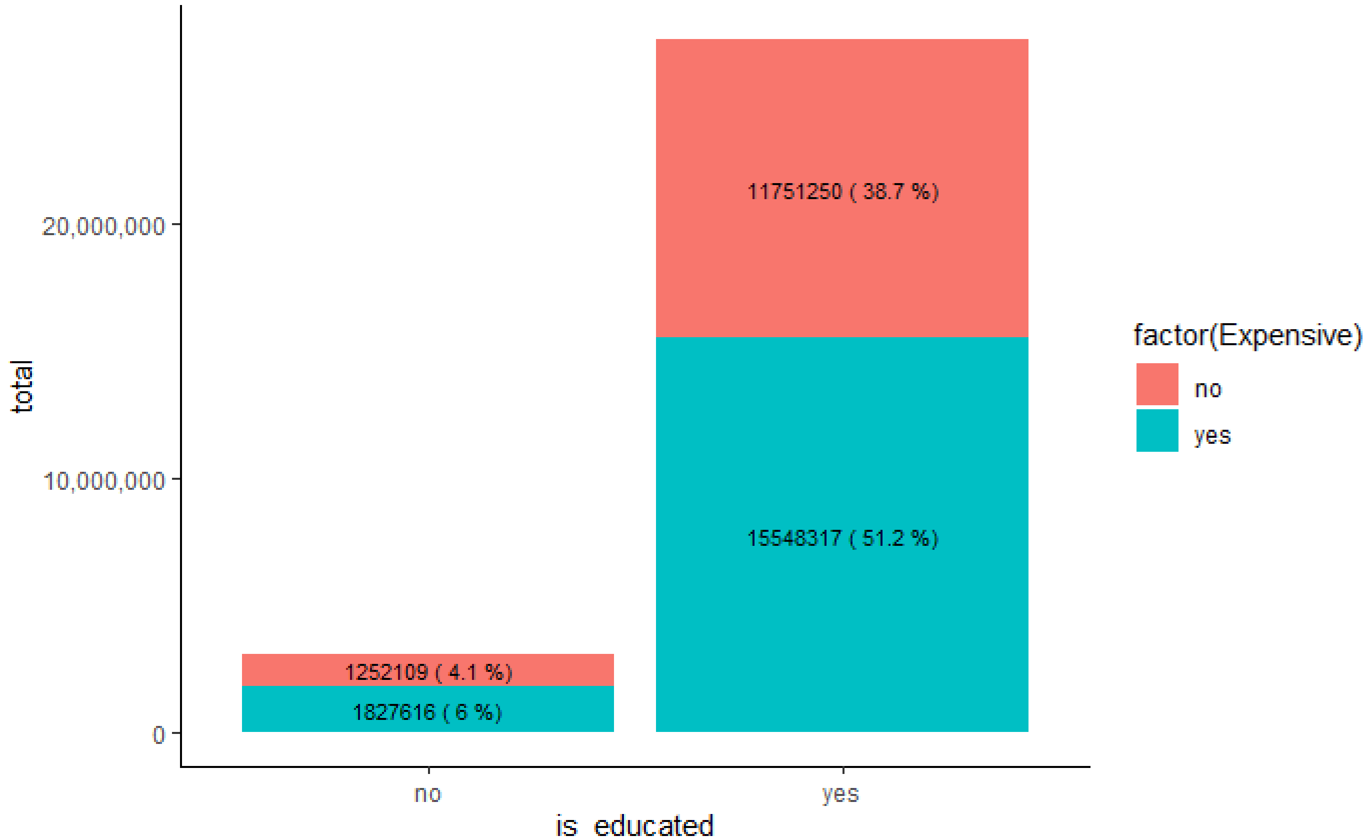
```
education_level_group_cost <- education_level_group_cost %>% mutate(prop = round(total/30379292
,3))
education_level_group_cost
```

is_educated <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
no	no	1252109	2115.049	5952	5	2394189	1547.317	0.041
yes	no	11751250	2151.456	5986	2	2413438	1553.524	0.387
no	yes	1827616	11422.600	42820	6004	38629280	6215.246	0.060
yes	yes	15548317	12071.675	55715	6001	38130410	6174.983	0.512

4 rows

Hide

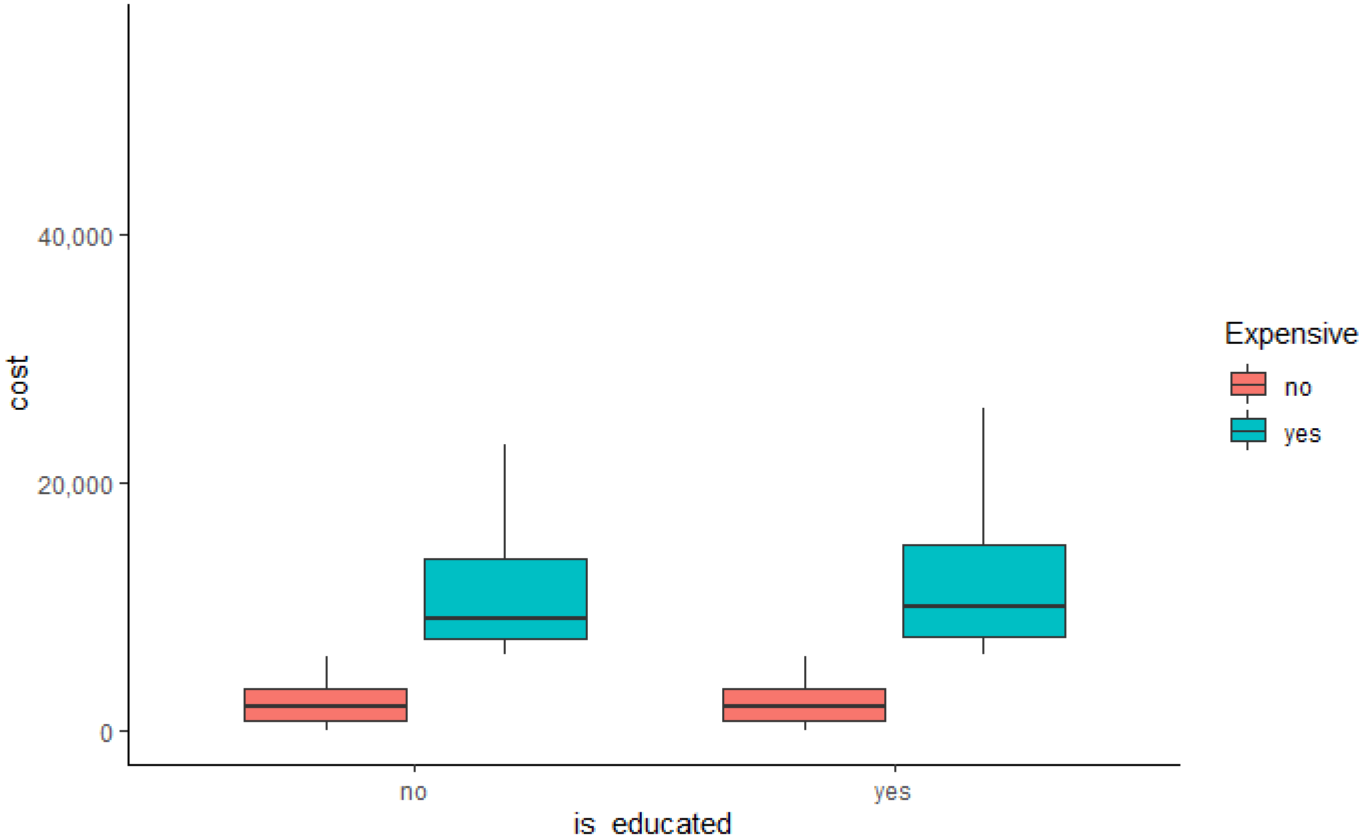
```
# plot
ggplot(education_level_group_cost, aes(is_educated, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The table and bar chart shows the detailed statistical results of two groups (high and low cost) considering whether a person has a college degree or not. The interesting point is that people with a college degree have a higher healthcare cost than other people without an education degree.

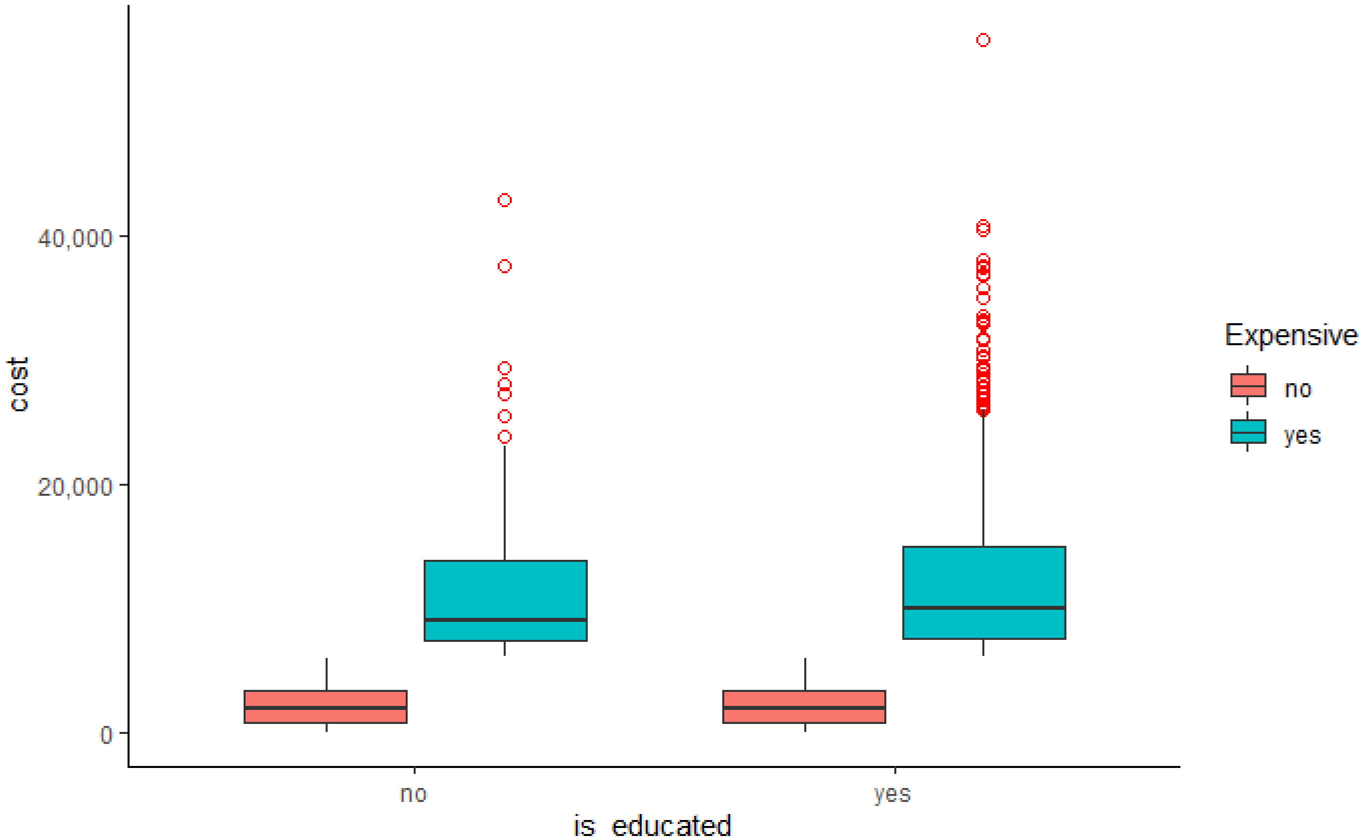
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(is_educated, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(is_educated, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] There are more outliers in the is_educated - yes and expensive - yes group than is_educated - no and expensive - yes group.

9. married

Hide

```
# grouping (married ~ numer of observation)
# table
married_group <- df_new %>%
  group_by(married, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'married'. You can override using the `.groups` argument.
```

Hide

```
colnames(married_group)[3] <- "count"
married_group <- married_group %>% mutate(prop = round(count/7502, 3))
married_group
```

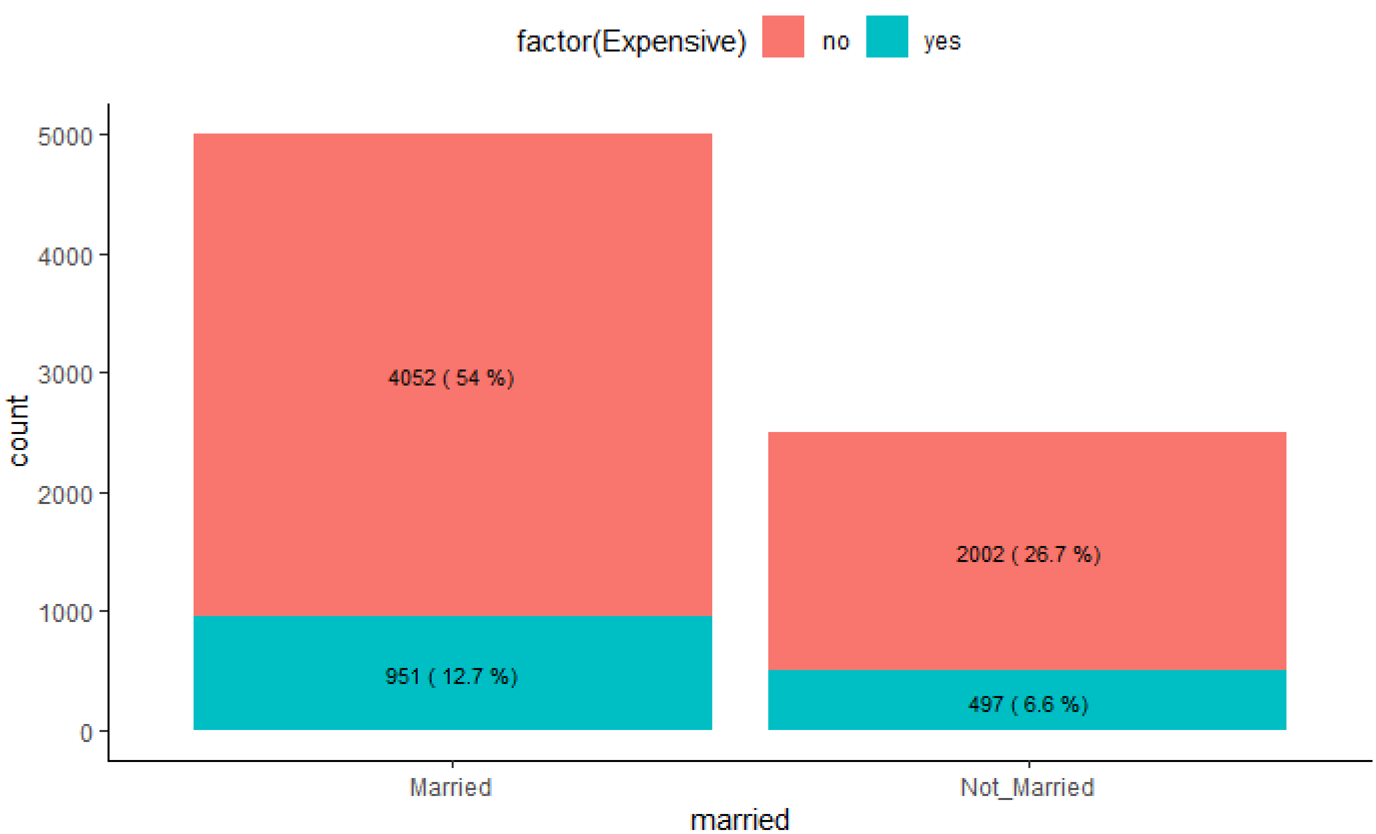
married <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Married	no	4052	37.53702	192.8905	13.88850	0.540

married <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
Not_Married	no	2002	37.22428	195.1446	13.96942	0.267
Married	yes	951	45.08412	185.2224	13.60964	0.127
Not_Married	yes	497	45.19316	184.9021	13.59787	0.066

4 rows

Hide

```
# plot
ggplot(married_group, aes(married, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (married ~ cost)
# table
married_group_cost <- df_new %>%
                    group_by(married, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'married'. You can override using the `.groups` argument.
```

Hide

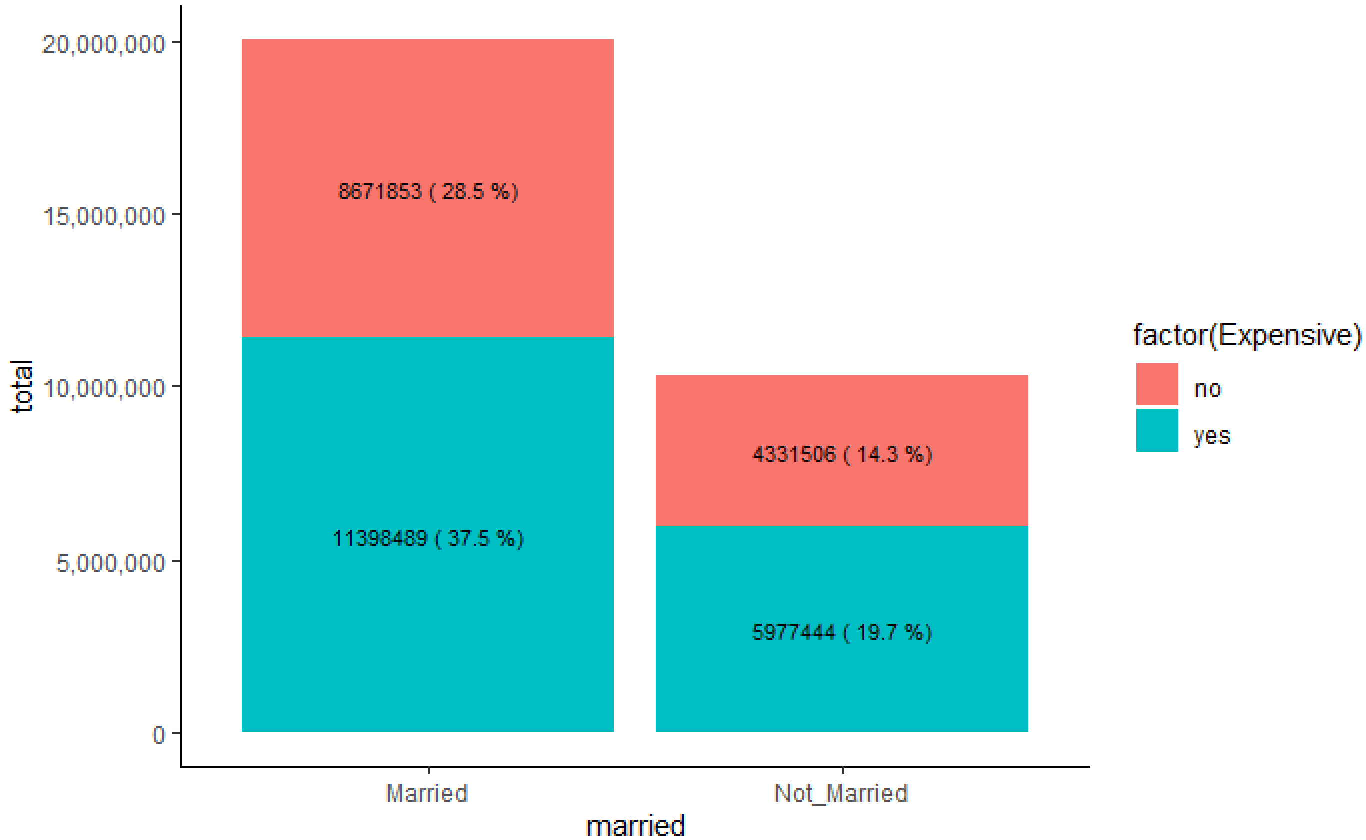
```
married_group_cost <- married_group_cost %>% mutate(prop = round(total/30379292 ,3))
married_group_cost
```

married	Expensive	total	mean	max	min	var	sd	prop
<chr>	<chr>	<int>	<dbl>	<int>	<int>	<dbl>	<dbl>	<dbl>
Married	no	8671853	2140.141	5986	2	2402165	1549.892	0.285
Not_Married	no	4331506	2163.589	5945	5	2430561	1559.026	0.143
Married	yes	11398489	11985.793	42820	6001	36025237	6002.103	0.375
Not_Married	yes	5977444	12027.050	55715	6048	42442181	6514.766	0.197

4 rows

Hide

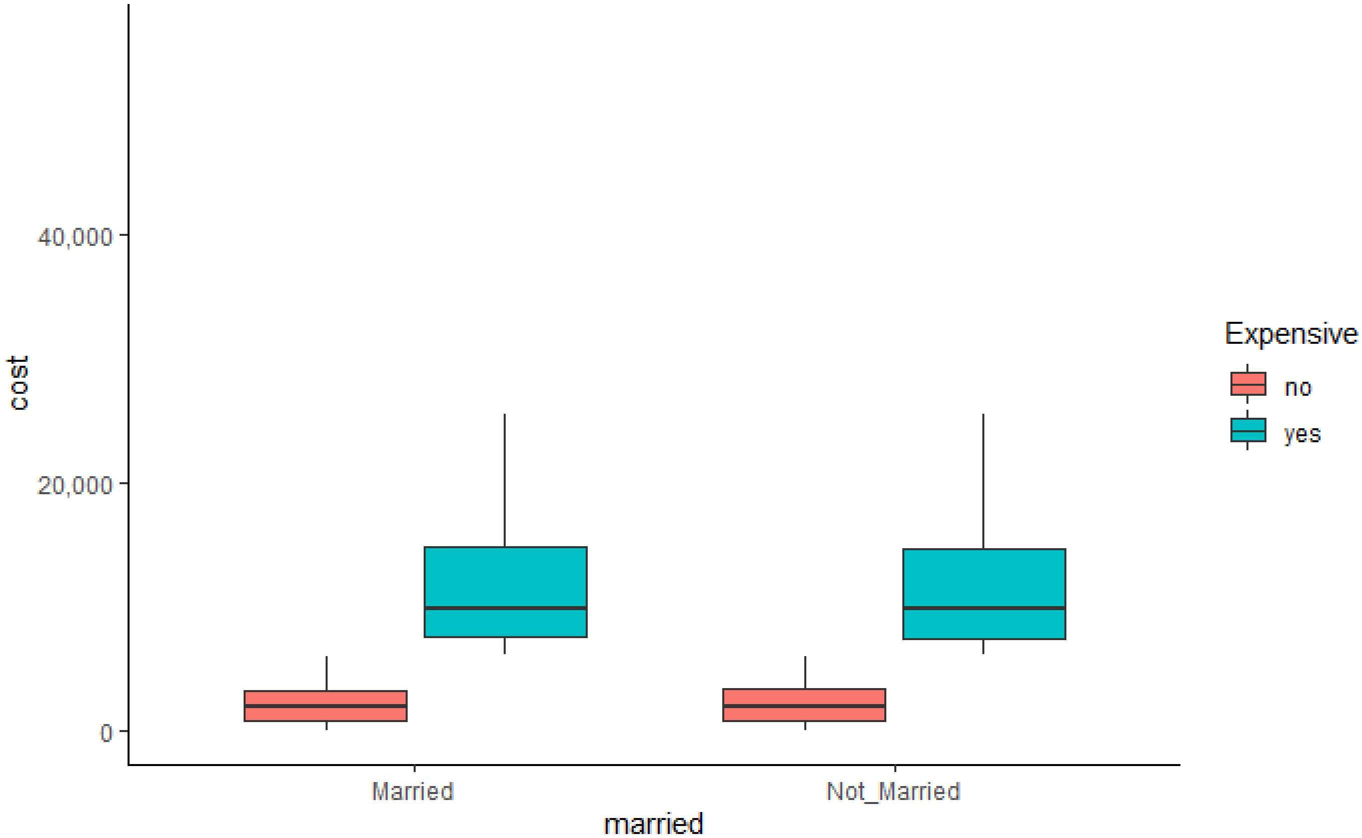
```
# plot
ggplot(married_group_cost, aes(married, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The table and bar chart represents the detailed statistical results of two groups (high and low cost) considering whether a person gets married or not. Both bar plots show that more people are married in the data set, and they have a higher cost than the other people who are not married.

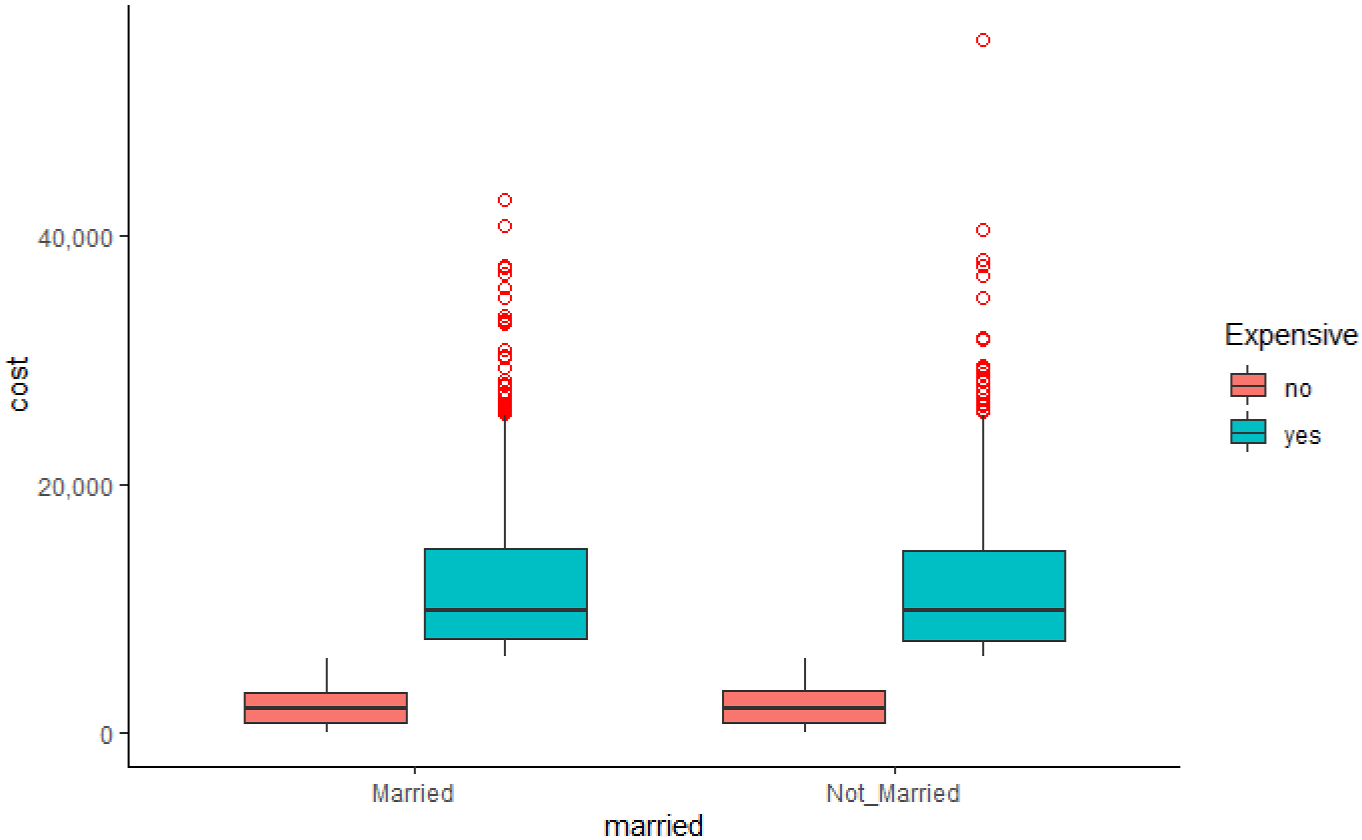
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(married, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(married, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



[Comments] 1) As seen in the box plot with the outliers, we can find the outliers of $55,715 on the not_married group with the higher healthcare cost.

10. number of children - have_child

Hide

```
# grouping (num of children ~ numer of observation)
# table
children_group <- df_new %>%
  group_by(have_child, Expensive) %>%
  summarise(count=n(), mean=mean(age), var=var(age), sd=sd(age)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'have_child'. You can override using the `.groups` argument.
```

Hide

```
colnames(children_group)[3] <- "count"
children_group <- children_group %>% mutate(prop = round(count/7502, 3))
children_group
```

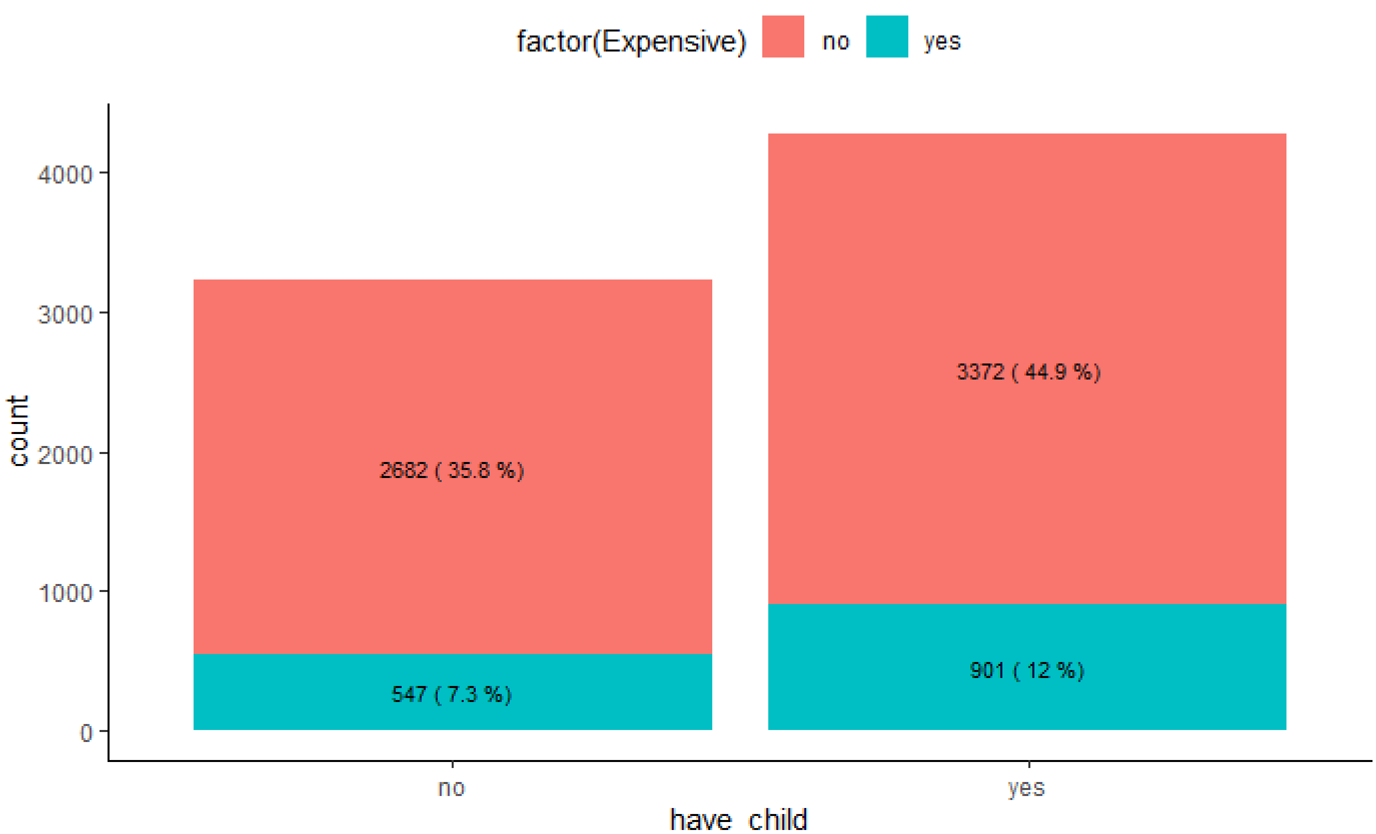
have_child <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
no	no	2682	36.20172	258.2827	16.07118	0.358

have_child <chr>	Expensive <chr>	count <int>	mean <dbl>	var <dbl>	sd <dbl>	prop <dbl>
yes	no	3372	38.41340	140.0925	11.83607	0.449
no	yes	547	45.13711	259.7632	16.11717	0.073
yes	yes	901	45.11210	139.8285	11.82491	0.120

4 rows

Hide

```
# plot
ggplot(children_group, aes(have_child, count, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme_classic() +
  theme(legend.position = "top") +
  geom_text(aes(label=paste(count,"(",prop*100, "%)")), size = 3, position = position_stack(0.
5))
```



Hide

```
# grouping (num of children ~ cost)
# table
children_group_cost <- df_new %>%
                    group_by(have_child, Expensive) %>%
                    summarise(total=sum(cost), mean=mean(cost), max=max(cost), min=min(cost), va
r=var(cost), sd=sd(cost)) %>%
  arrange(Expensive)
```

```
`summarise()` has grouped output by 'have_child'. You can override using the `.groups` argument.
```

Hide

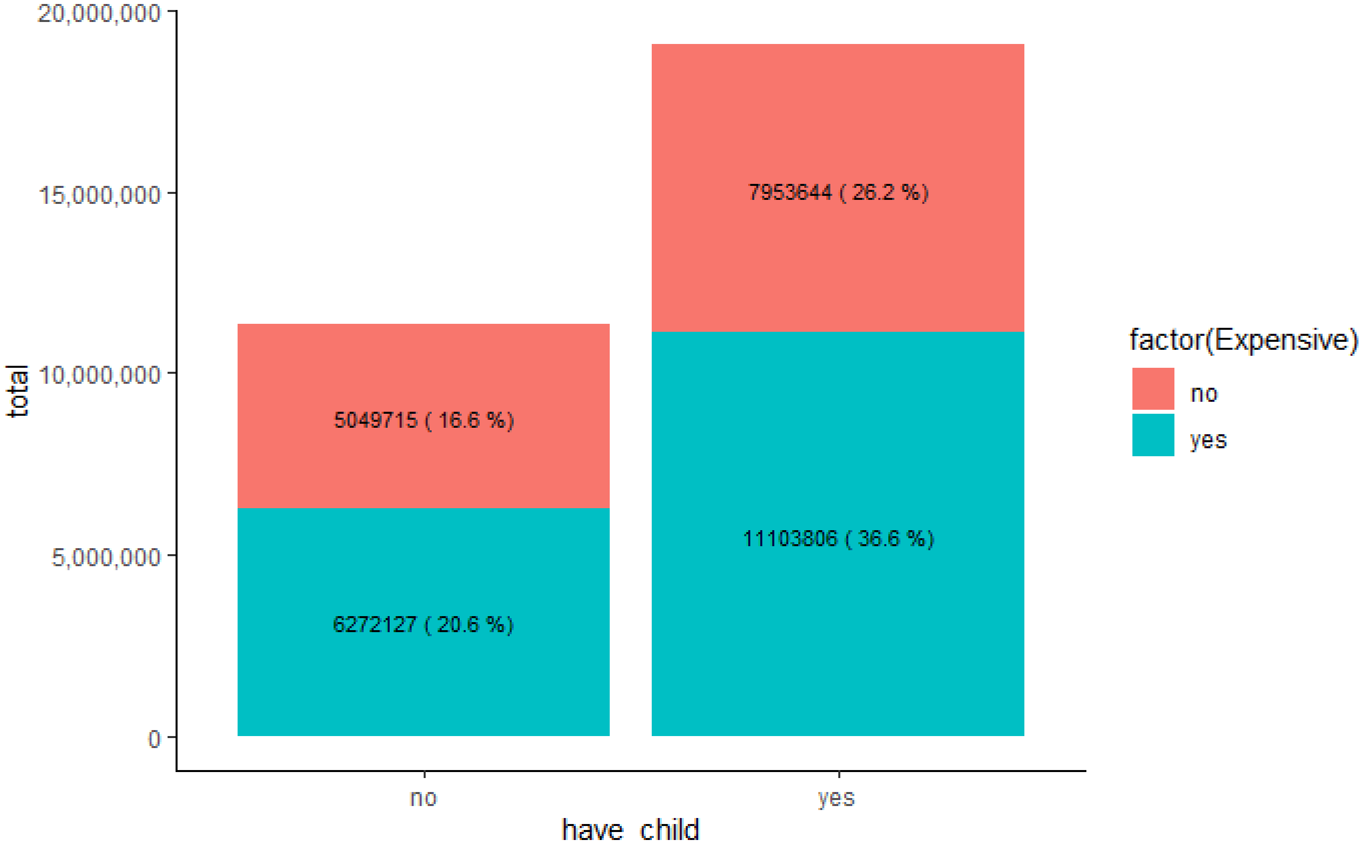
```
children_group_cost <- children_group_cost %>% mutate(prop = round(total/30379292 ,3))
children_group_cost
```

have_child <chr>	Expensive <chr>	total <int>	mean <dbl>	max <int>	min <int>	var <dbl>	sd <dbl>	prop <dbl>
no	no	5049715	1882.817	5965	2	2579601	1606.113	0.166
yes	no	7953644	2358.732	5986	5	2177752	1475.721	0.262
no	yes	6272127	11466.411	40664	6003	33191303	5761.189	0.206
yes	yes	11103806	12323.869	55715	6001	41003505	6403.398	0.366

4 rows

Hide

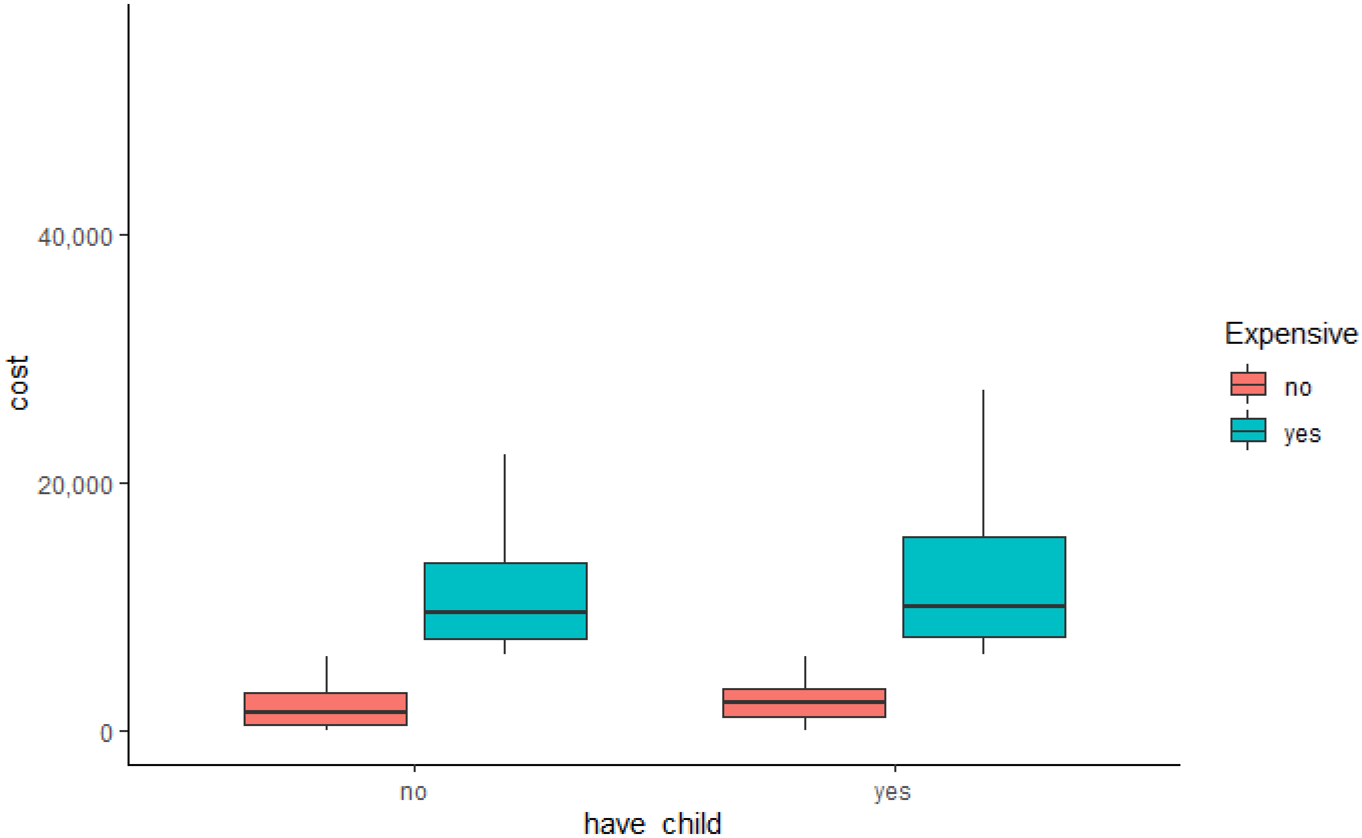
```
# plot
ggplot(children_group_cost, aes(have_child, total, fill=factor(Expensive))) +
  geom_bar(stat="identity", position=position_stack()) +
  theme(legend.position = "top") +
  theme_classic() +
  geom_text(aes(label=paste(round(total, 0),"(",prop*100, "%)")), size = 3, position = position_
stack(0.5)) +
  scale_y_continuous(labels = scales::comma)
```



[Comments] The table and bar chart shows the detailed statistical results of two groups (high and low cost) considering whether a person has a child. Both bar plots show that more people have at least one child and they have a higher cost than the other people who don't have a child.

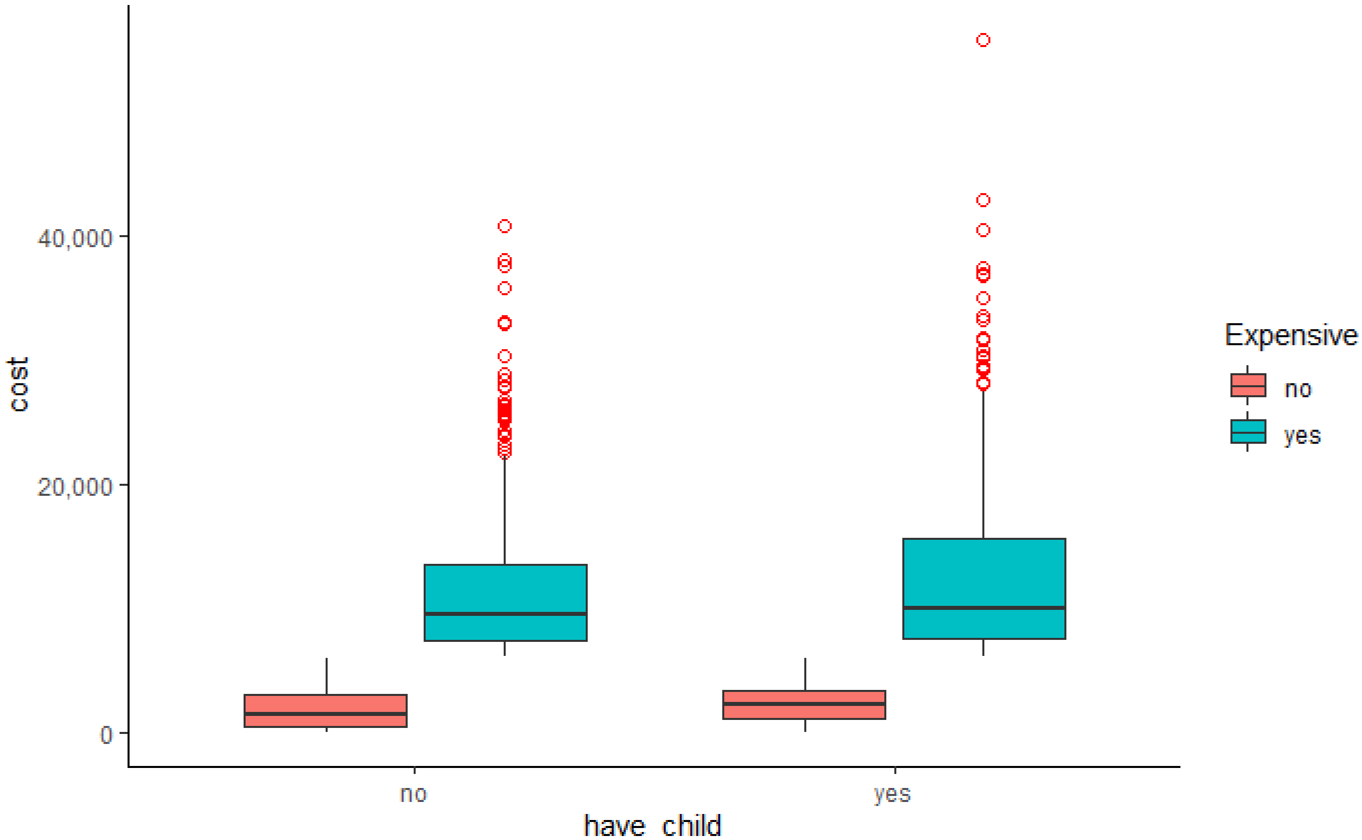
Hide

```
# box plot
# without outlier
ggplot(df_new, aes(have_child, cost, fill=Expensive)) +
  geom_boxplot(outlier.shape=NA) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



Hide

```
# box plot
# with outlier
ggplot(df_new, aes(have_child, cost, fill=Expensive)) +
  geom_boxplot(outlier.colour="red", outlier.shape=1, outlier.size=2) +
  theme_classic() +
  scale_y_continuous(labels=scales::comma)
```



11. mappings

To future investigate the cost in different locations, we created map that summarizes the number of people paying more than 6000.

Hide

```
# Create the US map
states <- map_data("state")
bb <- c(left = min(states$long),
bottom = min(states$lat),
right = max(states$long),
top = max(states$lat)) # set limitations of the map
map <- get_stamenmap(bbox = bb, zoom = 4)
```

```
Source : http://tile.stamen.com/terrain/4/2/5.png
Source : http://tile.stamen.com/terrain/4/3/5.png
Source : http://tile.stamen.com/terrain/4/4/5.png
Source : http://tile.stamen.com/terrain/4/5/5.png
Source : http://tile.stamen.com/terrain/4/2/6.png
Source : http://tile.stamen.com/terrain/4/3/6.png
Source : http://tile.stamen.com/terrain/4/4/6.png
Source : http://tile.stamen.com/terrain/4/5/6.png
```

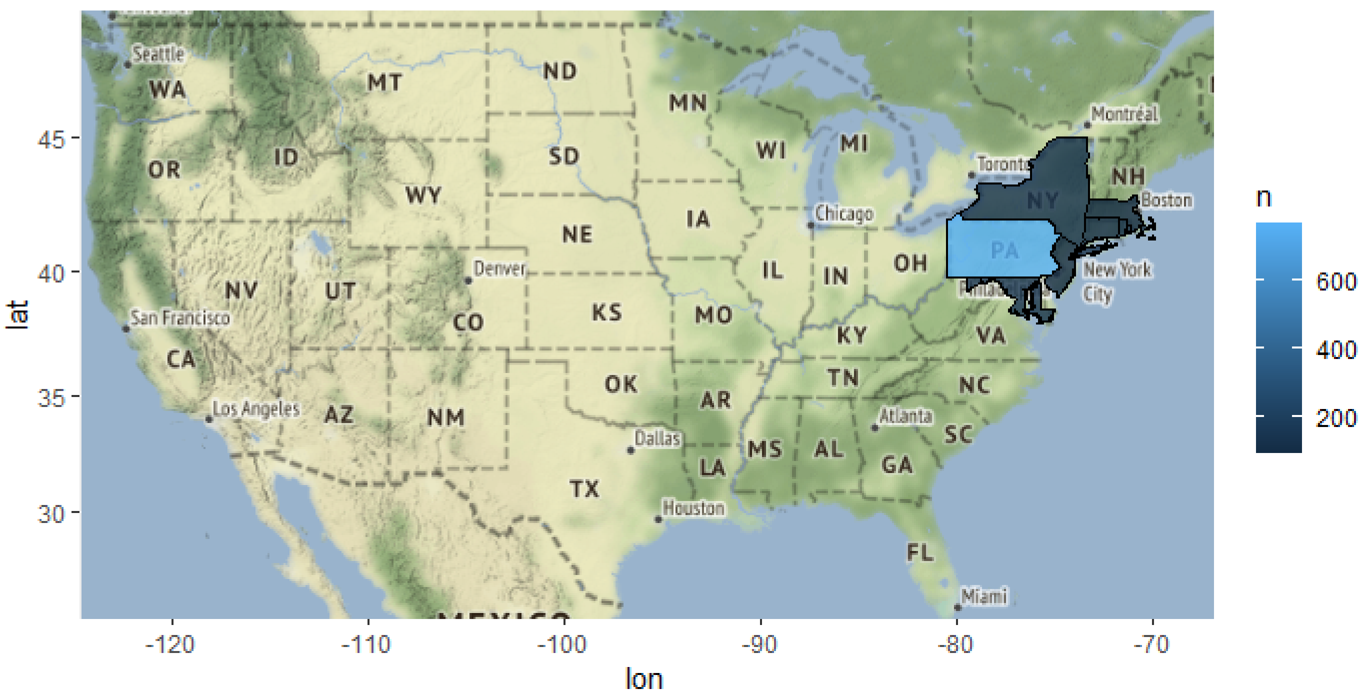
Hide

```
# Show the map of people who are expensive based on their state
df_by_state <- df_new %>% group_by(location,Expensive) %>% summarise(n = n())
```

```
`summarise()` has grouped output by 'location'. You can override using the `.groups` argument.
```

Hide

```
df_by_state$State <- tolower(df_by_state$location)
df_by_state_yes <- filter(df_by_state, Expensive == 'yes')
dfMap <- merge(df_by_state_yes, states, by.x = 'State', by.y = 'region')
dfMap <- dfMap %>% arrange(order)
ggmap(map) + geom_polygon(data = dfMap, color = "black", alpha = 0.8, aes(x = long, y = lat, gro
up = group, fill = n))
```



[Comments] The map shows clearly that PENNSYLVANIA have more people paying more than 6000 on their health.

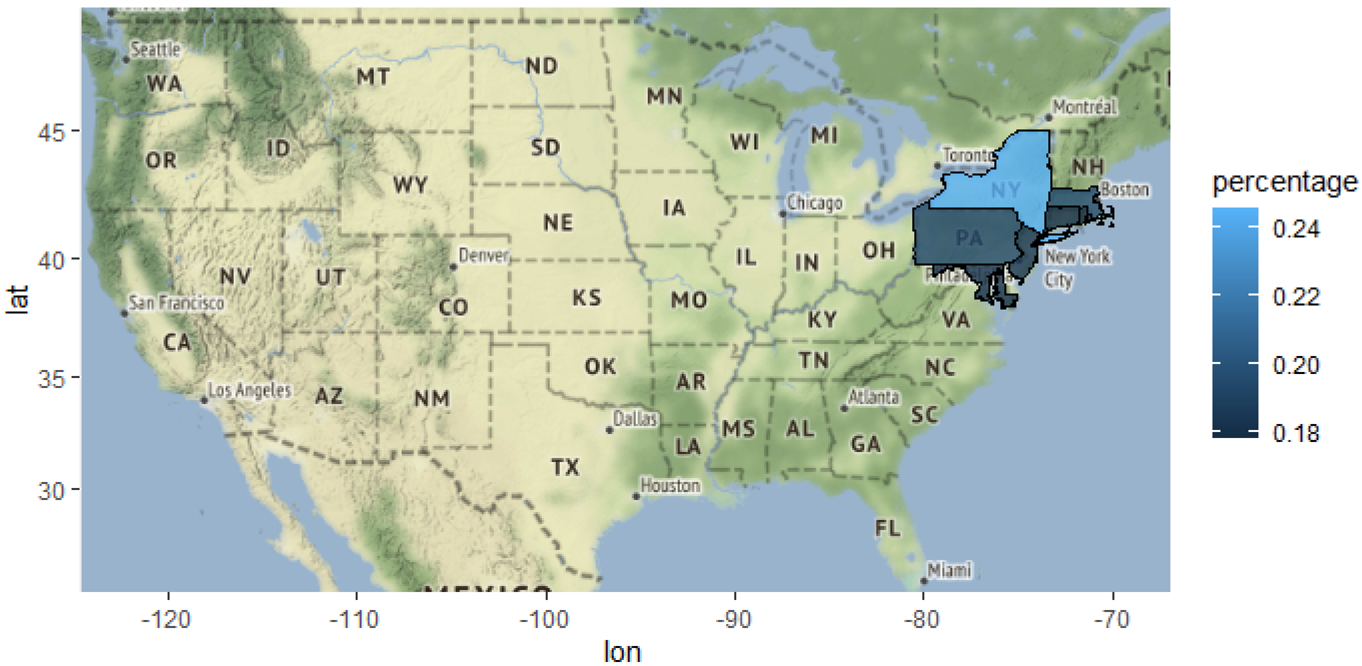
Hide

```
#Show the percentage of people pay more than 6000 in us by state
df_temp <- df_new %>% group_by(location) %>% summarise(n = n())
df_by_state <- df_new %>% group_by(location,Expensive) %>% summarise(n = n())
```

```
`summarise()` has grouped output by 'location'. You can override using the `.groups` argument.
```

Hide

```
df_by_state$State <- tolower(df_by_state$location)
df_by_state_yes <- filter(df_by_state, Expensive == 'yes')
df_by_state_yes$percentage <- df_by_state_yes$n / df_temp$n
dfMap <- merge(df_by_state_yes, states, by.x = 'State', by.y = 'region')
dfMap <- dfMap %>% arrange(order)
ggmap(map) + geom_polygon(data = dfMap, color = "black", alpha = 0.8, aes(x = long, y = lat, gro
up = group, fill = percentage))
```

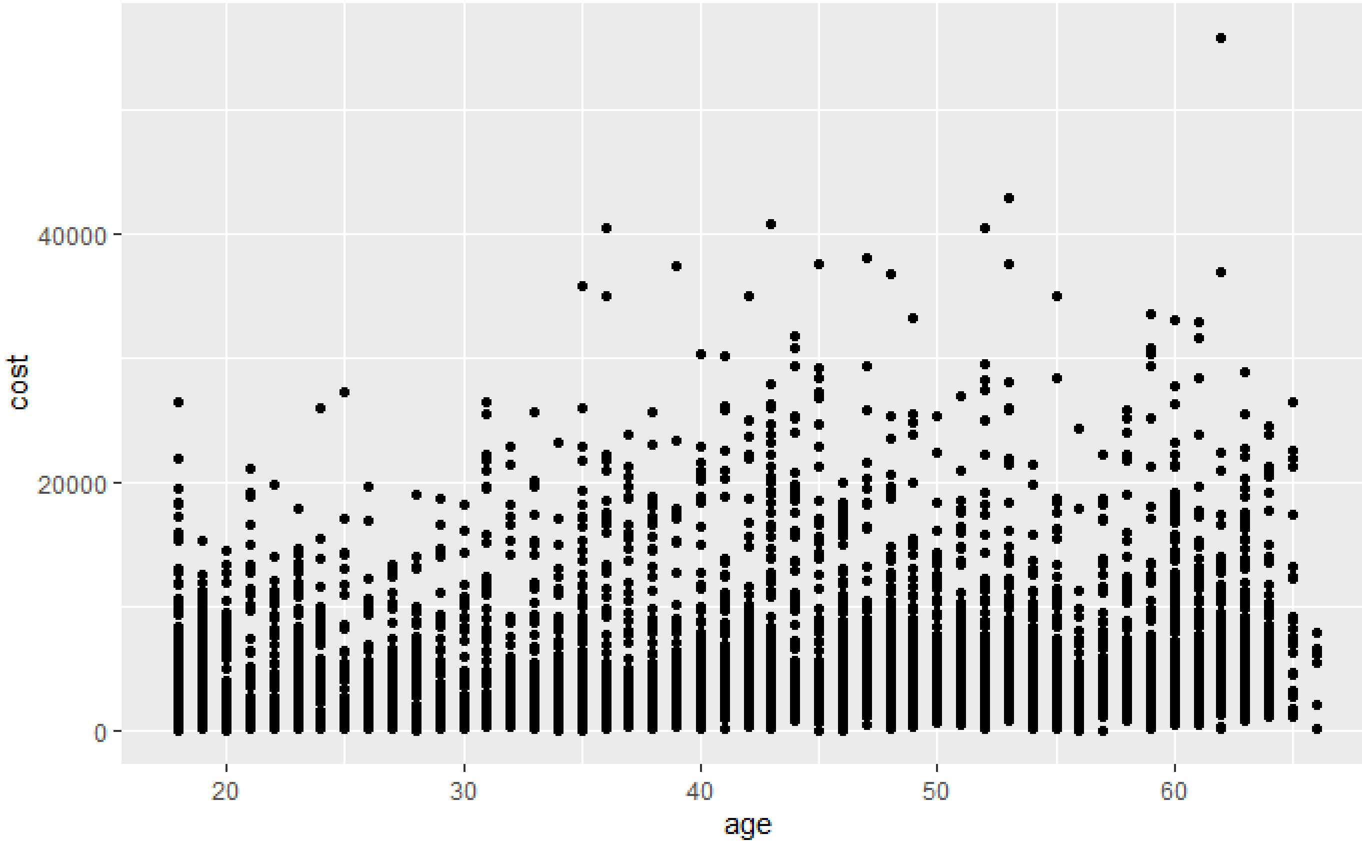


[Comments] The map shows clearly that people who live on New York have higher chances of paying more than 6000 on their health. Both of the maps indicate that which state people live in might make a difference.

#MODEL BUILDING ##1) linear model

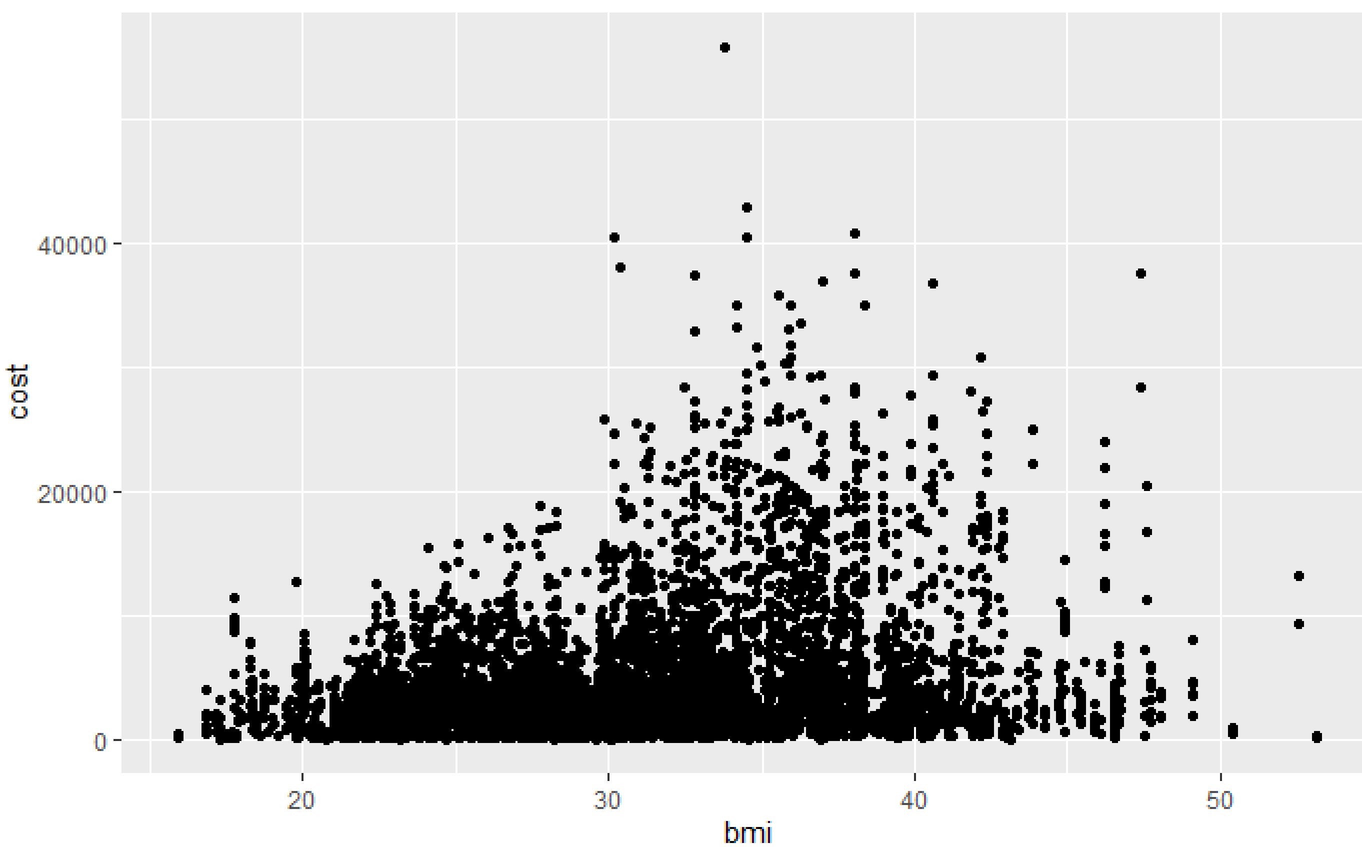
Hide

```
#1)linear model
#Building linear model using numeric predictors
#visualize the relationship between each predictor and cost
ggplot(data=df_new,aes(x=age, y=cost))+geom_point()
```



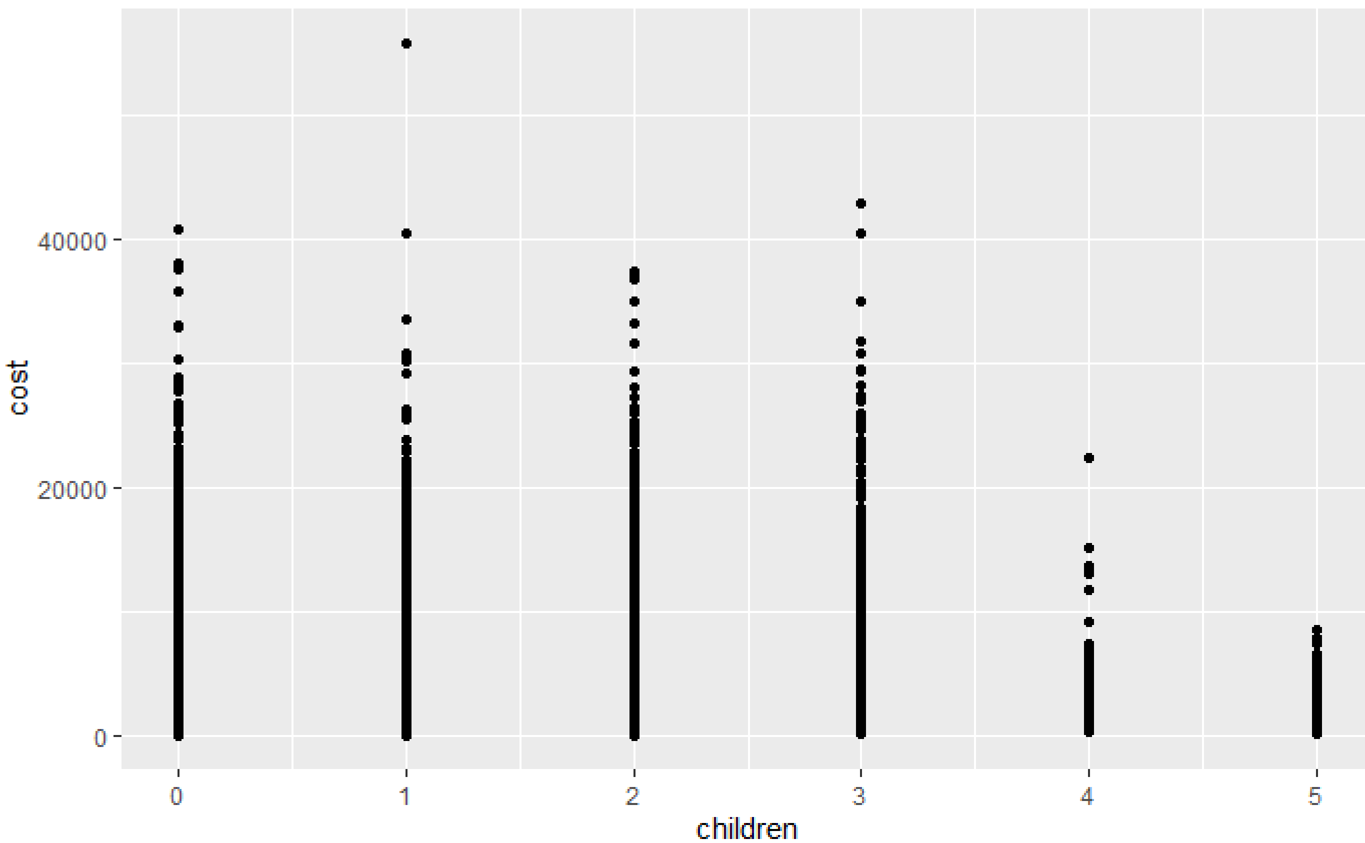
Hide

```
ggplot(data=df_new,aes(x=bmi, y=cost))+geom_point()
```



Hide

```
ggplot(data=df_new,aes(x=children, y=cost))+geom_point()
```



Hide

```
#Build a multiple regression model using age, bmi and number of children
lmOut <- lm(cost~age+bmi+children, data=df)
summary(lmOut)
```

```
Call:
lm(formula = cost ~ age + bmi + children, data = df)

Residuals:
   Min     1Q Median     3Q    Max 
 -7810  -2381  -1278    531  48755 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) -5888.645    302.160 -19.489   <2e-16 ***
age           103.896      3.721  27.920   <2e-16 ***
bmi           180.873      8.807  20.537   <2e-16 ***
children      293.975     43.123   6.817    1e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 4539 on 7498 degrees of freedom
Multiple R-squared:  0.1572,	Adjusted R-squared:  0.1569 
F-statistic: 466.3 on 3 and 7498 DF,  p-value: < 2.2e-16
```

Hide

```
#Comment : Although all of the predictors in this case are significant, the model only explains 15.69% of the dataset, which is quite low.However, we will further use cross validation to test the model's accuracy and sensitivity.

#Divide the data into training and testing dataset for lm
set.seed(1)
trainList <- createDataPartition(y=df$cost, p=.70, list=FALSE)
trainData <- df[trainList,]
testData <- df[-trainList,]
lmOut2 <- lm(cost~age+bmi+children, data=trainData)
lmPred <- predict(lmOut2, newdata=testData)

#getting our confusion matrix for linear model
PredictValues <- as.factor(ifelse(lmPred >= 6000, 'yes', 'no'))
testData$Expensive <- as.factor(ifelse(testData$cost >= 6000, 'yes', 'no'))
confusionMatrix(PredictValues,testData$Expensive)
```

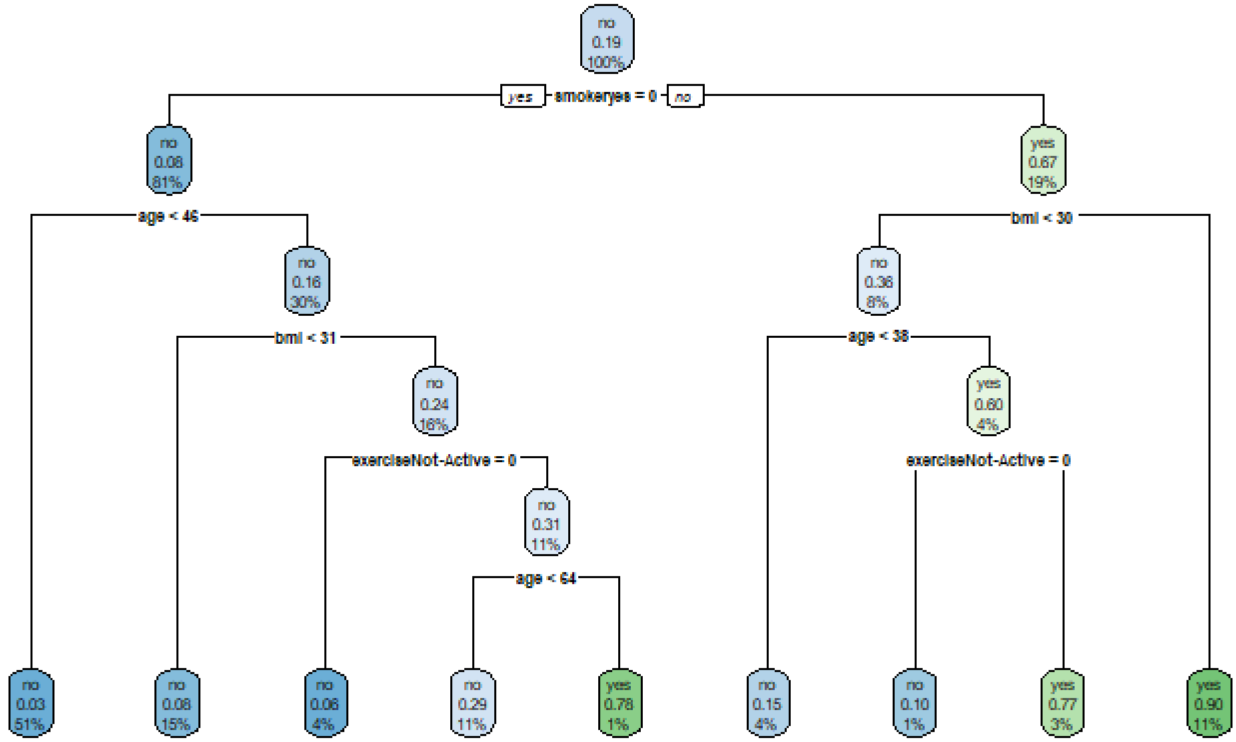
```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1578  290
       yes  227  153

               Accuracy : 0.77
                 95% CI : (0.7521, 0.7873)
    No Information Rate : 0.8029
    P-Value [Acc > NIR] : 0.999947

                  Kappa : 0.2321

 Mcnemar's Test P-Value : 0.006396

            Sensitivity : 0.8742
            Specificity : 0.3454
         Pos Pred Value : 0.8448
         Neg Pred Value : 0.4026
             Prevalence : 0.8029
         Detection Rate : 0.7020
   Detection Prevalence : 0.8310
      Balanced Accuracy : 0.6098

       'Positive' Class : no
```

[Comments] As we can see, the sensitivity here is 0.8705. The accuracy is below No Information Rate. The linear model is not a good model in general.

##2) Decision Tree Model We then turn to more complicated machine learning models.

Hide

```
#Decision tree model 1:
#Use all the predictors(exclude location) to construct a decision tree model
dfX <- data.frame(age = (df_new$age),
                 bmi = (df_new$bmi),
                 education = (df_new$education_level),
                 children = (df_new$children),
                 smoker = (df_new$smoker),
                 location = (df_new$location),
                 location_type = (df_new$location_type),
                 yearly_physical = (df_new$yearly_physical),
                 exercise = (df_new$exercise),
                 married = (df_new$married),
                 hypertension = (df_new$hypertension),
                 gender = (df_new$gender),
                 Expensive = (df_new$Expensive))

#Divide dataframe into train set and test set
set.seed(250)
trainList <- createDataPartition(y=dfX$Expensive, p=0.70, list=FALSE)
trainSet <- dfX[trainList,]
testSet <- dfX[-trainList,]
# Define train control factors, use repeatedcv for 10 times
trctrl <- trainControl(method = "repeatedcv", number = 10)


#Build rpart tree model
tree_model1 <- train(Expensive~., data = trainSet, method = 'rpart', trControl=trctrl, tuneLengt
h = 10)
rpart.plot(tree_model1$finalModel)
```



Hide

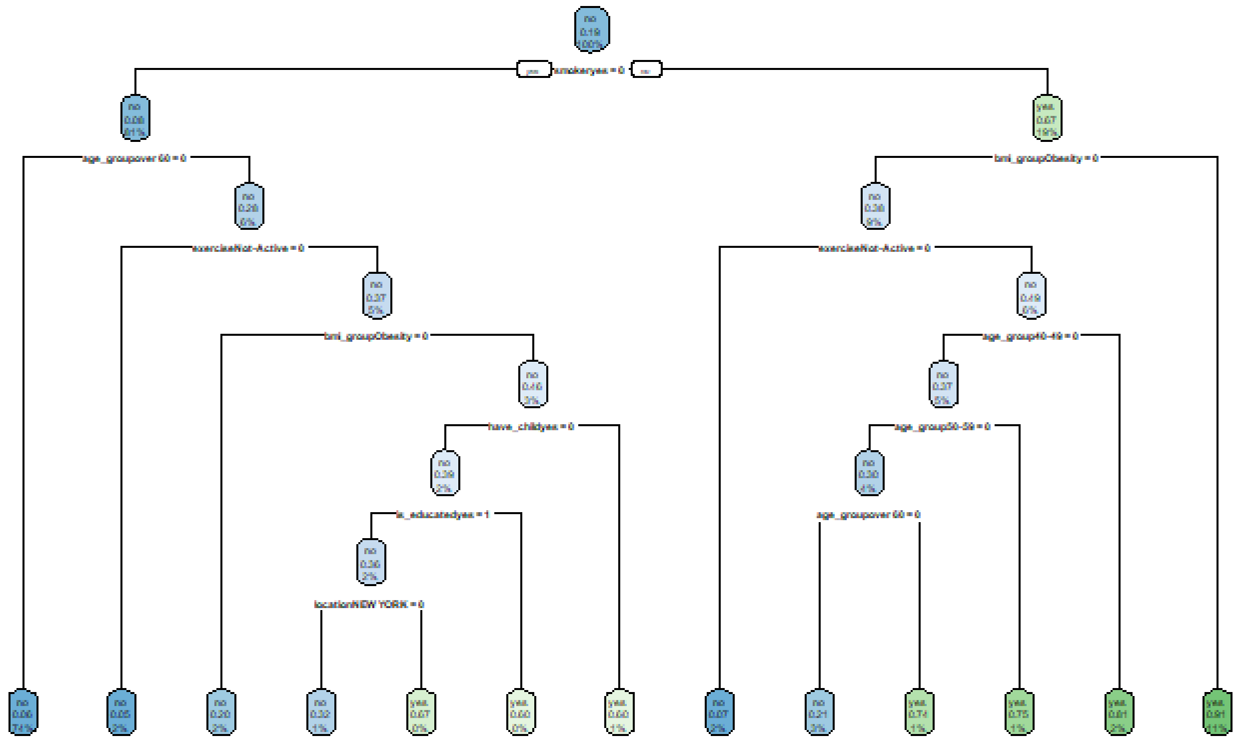
```
#test our tree model 1 on test set:
treePred1 <- predict(tree_model1, newdata = testSet)
confusionMatrix(treePred1, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1764  154
       yes   52  280

               Accuracy : 0.9084
                 95% CI : (0.8958, 0.92)
    No Information Rate : 0.8071
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6771

 Mcnemar's Test P-Value : 1.964e-12

            Sensitivity : 0.9714
            Specificity : 0.6452
         Pos Pred Value : 0.9197
         Neg Pred Value : 0.8434
             Prevalence : 0.8071
         Detection Rate : 0.7840
   Detection Prevalence : 0.8524
      Balanced Accuracy : 0.8083

       'Positive' Class : no
```

[Comments] As we can see, the sensitivity is 0.9714, which has been significantly improved compared with linear model.The accuracy is also higher than No Information Rate. Considering the cost of that putting all predictors into a business model is high, as well as there will be problems in overfitting, we are looking for ways to simplify the model by turning numeric variables into categorical variables and hoping to see the changes in performances.

Hide

```
#Decision tree model 2:
#Turning numeric variables into categorical ones
dfX2 <- data.frame(age_group = as.factor(df_new$age_group),
                 bmi_group = as.factor(df_new$bmi_group),
                 is_educated = as.factor(df_new$is_educated),
                 have_child = as.factor(df_new$have_child),
                 smoker = as.factor(df_new$smoker),
                 location = as.factor(df_new$location),
                 location_type = as.factor(df_new$location_type),
                 yearly_physical = as.factor(df_new$yearly_physical),
                 exercise = as.factor(df_new$exercise),
                 married = as.factor(df_new$married),
                 hypertension = as.factor(df_new$hypertension),
                 gender = as.factor(df_new$gender),
                 Expensive = as.factor(df_new$Expensive))

#Divide dataframe into train set and test set
set.seed(250)
trainList <- createDataPartition(y=dfX2$Expensive, p=0.70, list=FALSE)
trainSet <- dfX2[trainList,]
testSet <- dfX2[-trainList,]
# Define train control factors, use repeatedcv for 10 times
trctrl <- trainControl(method = "repeatedcv", number = 10)


#Build rpart tree model
tree_model2 <- train(Expensive~., data = trainSet, method = 'rpart', trControl=trctrl, tuneLengt
h = 10)
rpart.plot(tree_model2$finalModel)
```



```
#test out tree model 2 on test set
treePred2 <- predict(tree_model2, newdata = testSet)
confusionMatrix(treePred2, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1766  145
       yes   50  289
                                          
               Accuracy : 0.9133          
                 95% CI : (0.9009, 0.9246)
    No Information Rate : 0.8071          
    P-Value [Acc > NIR] : < 2.2e-16       
                                          
                  Kappa : 0.6964          
                                          
 Mcnemar's Test P-Value : 1.679e-11       
                                          
            Sensitivity : 0.9725          
            Specificity : 0.6659          
         Pos Pred Value : 0.9241          
         Neg Pred Value : 0.8525          
             Prevalence : 0.8071          
         Detection Rate : 0.7849          
   Detection Prevalence : 0.8493          
      Balanced Accuracy : 0.8192          
                                          
       'Positive' Class : no              
```

[Comments] The sensitivity rate goes up to 0.9725 when we simplified some of our predictors and the accuracy was significantly improved compared to No Information Rate. Binning all the numeric variables improve the performance of our tree model. Furthermore, we can also rule out some of the predictors that are less important in the tree model to make it more general.

Hide

```
#Decision Tree Model 3: Predictor Selection
varImp(tree_model2)
```

```
rpart variable importance

  only 20 most important variables shown (out of 30)
```

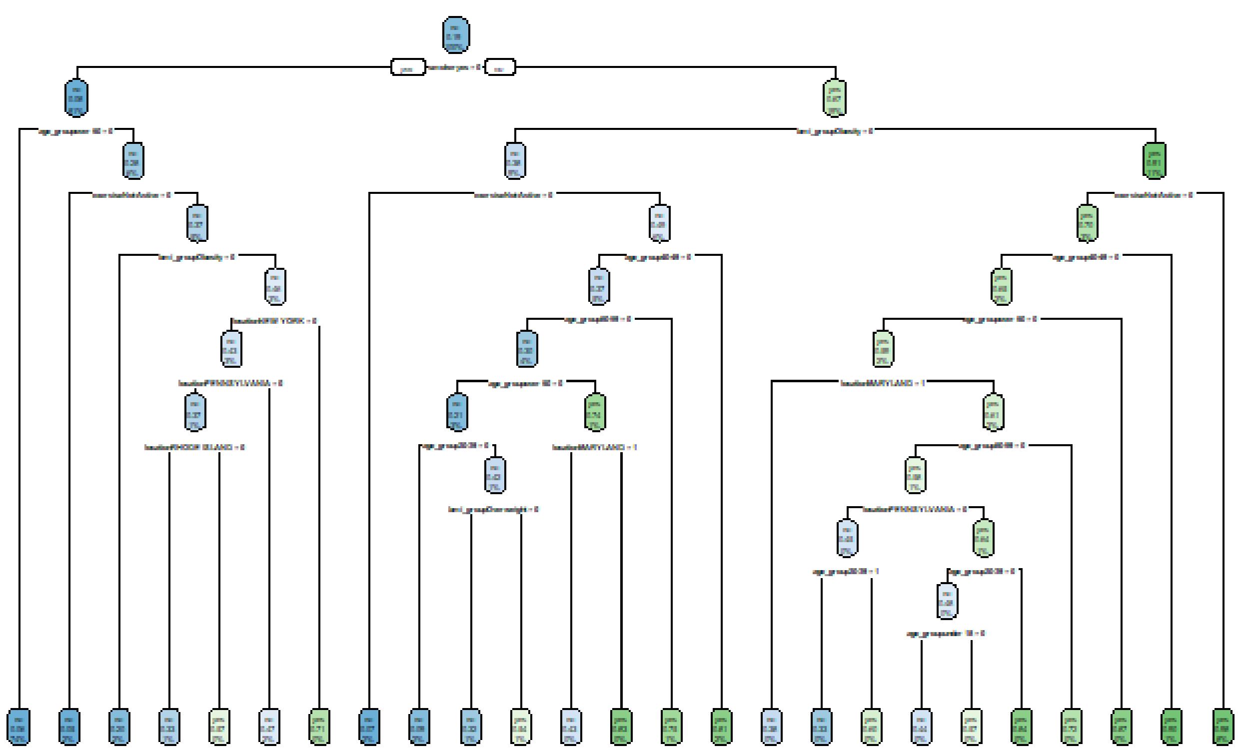
	Overall <dbl>
smokeryes	100.0000000
bmi_groupObesity	40.0633439
exerciseNot-Active	24.1260923
age_groupover 60	19.1034540

	Overall <dbl>
bmi_groupOverweight	15.2111534
age_group40-49	12.0627992
age_groupunder 18	10.2777382
age_group50-59	5.5205105
have_childyes	1.8734008
locationNEW YORK	1.4782605

1-10 of 20 rows Previous 1 2 Next

Hide

```
#We then excluded some of the predictors that are less important according the the result

trainSet <- select(trainSet, -gender, -have_child, -hypertension, -yearly_physical, -location_ty
pe, -married, -is_educated)
tree_model3 <- train(Expensive~., data = trainSet, method = 'rpart', trControl=trctrl, tuneLengt
h = 10)
rpart.plot(tree_model3$finalModel)
```



Hide

```
#test out tree model 3 on test set
treePred3 <- predict(tree_model3, newdata = testSet)
confusionMatrix(treePred3, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1772  158
       yes   44  276

               Accuracy : 0.9102
                 95% CI : (0.8976, 0.9217)
    No Information Rate : 0.8071
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6796

 Mcnemar's Test P-Value : 1.855e-15

            Sensitivity : 0.9758
            Specificity : 0.6359
         Pos Pred Value : 0.9181
         Neg Pred Value : 0.8625
             Prevalence : 0.8071
         Detection Rate : 0.7876
   Detection Prevalence : 0.8578
      Balanced Accuracy : 0.8059

       'Positive' Class : no
```

Hide

```
#Comment: The sensitivity rate goes up to 0.9758 with selected predictors.
```

[Comments] The sensitivity rate goes up to 9758 when we simplified some of our predictors. This is the best performing decision tree model we have so far.

##3)SVM Model Apart from decision tree, support vector machine is also a good machine learning technique in supervised learning. We use the same process with decision trees and compare the performances between each model. First, we included all the predictors as they were without transferring numeric ones into categorical ones.

Hide

```
#SVM Model 1
#Divide dataframe into train set and test set
set.seed(250)
trainList <- createDataPartition(y=dfX$Expensive, p=0.70, list=FALSE)
trainSet <- dfX[trainList,]
testSet <- dfX[-trainList,]
# Define train control factors, use repeatedcv for 10 times
trctrl <- trainControl(method = "repeatedcv", number = 10)
svm_model1 <- train(Expensive~., data = trainSet, method = "svmRadial",trCotrol=trctrl, preProc=
c("center","scale"))
setwd("C:/Users/73457/Desktop/final project GROUP 1")
```

Hide

```
#test out svm model 1 on test data
svmPred1 <- predict(svm_model1, newdata = testSet)
confusionMatrix(svmPred1, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1759  170
       yes   57  264

               Accuracy : 0.8991
                 95% CI : (0.8859, 0.9113)
    No Information Rate : 0.8071
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6403

 Mcnemar's Test P-Value : 1.056e-13

            Sensitivity : 0.9686
            Specificity : 0.6083
         Pos Pred Value : 0.9119
         Neg Pred Value : 0.8224
             Prevalence : 0.8071
         Detection Rate : 0.7818
   Detection Prevalence : 0.8573
      Balanced Accuracy : 0.7885

       'Positive' Class : no

```

[Comments] The sensitivity rate is 0.9686 and the accuracy is 89.91%. However, it’s not better than the best performing decision tree model. We then used binning techniques to see if the performance improved.

Hide

```
#SVM Model 2
#Divide dataframe into train set and test set
set.seed(250)
trainList <- createDataPartition(y=dfX2$Expensive, p=0.70, list=FALSE)
trainSet <- dfX2[trainList,]
testSet <- dfX2[-trainList,]
# Define train control factors, use repeatedcv for 10 times
trctrl <- trainControl(method = "repeatedcv", number = 10)
svm_model2 <- train(Expensive~., data = trainSet, method = "svmRadial",trCotrol=trctrl, preProc=
c("center","scale"))
```

Hide

```
#test out svm model 2 on test data
svmPred2 <- predict(svm_model2, newdata = testSet)
confusionMatrix(svmPred2, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  1773  168
       yes   43  266

               Accuracy : 0.9062
                 95% CI : (0.8934, 0.918)
    No Information Rate : 0.8071
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6617

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.9763
            Specificity : 0.6129
         Pos Pred Value : 0.9134
         Neg Pred Value : 0.8608
             Prevalence : 0.8071
         Detection Rate : 0.7880
   Detection Prevalence : 0.8627
      Balanced Accuracy : 0.7946

       'Positive' Class : no

```

Hide

```
#The sensitivity rate is 0.9763 , which is less than the best performing decision tree model.
```

[Comments] We saw improvements in both sensitivity and accuracy. However, it’s not better than the best performing decision tree model.

#Associate Mining

Hide

```
#We can also use associate mining here to see the importance of each variable.
df_tran <- as(dfX2,"transactions")
rules <- apriori(dfX2, parameter=list(supp=0.05, conf=0.8),
                 control=list(verbose=F),
                 appearance=list(default="lhs",rhs=("Expensive=yes")))
inspect(sort(rules, by="support"))
```

```
     lhs                          rhs                 support confidence   coverage     lift count
[1]  {bmi_group=Obesity,
      smoker=yes}              => {Expensive=yes} 0.09610770  0.9035088 0.10637163 4.681024   721
[2]  {bmi_group=Obesity,
      is_educated=yes,
      smoker=yes}              => {Expensive=yes} 0.08637697  0.9025070 0.09570781 4.675834   648
[3]  {bmi_group=Obesity,
      smoker=yes,
      exercise=Not-Active}     => {Expensive=yes} 0.07677953  0.9762712 0.07864569 5.058002   576
[4]  {bmi_group=Obesity,
      smoker=yes,
      hypertension=no}         => {Expensive=yes} 0.07517995  0.8952381 0.08397761 4.638174   564
[5]  {bmi_group=Obesity,
      smoker=yes,
      yearly_physical=No}      => {Expensive=yes} 0.07251400  0.8976898 0.08077846 4.650876   544
[6]  {bmi_group=Obesity,
      smoker=yes,
      location_type=Urban}     => {Expensive=yes} 0.07144761  0.8903654 0.08024527 4.612929   536
[7]  {bmi_group=Obesity,
      is_educated=yes,
      smoker=yes,
      exercise=Not-Active}     => {Expensive=yes} 0.06878166  0.9735849 0.07064783 5.044084   516
[8]  {bmi_group=Obesity,
      is_educated=yes,
      smoker=yes,
      hypertension=no}         => {Expensive=yes} 0.06704879  0.8918440 0.07517995 4.620589   503
[9]  {bmi_group=Obesity,
      smoker=yes,
      married=Married}         => {Expensive=yes} 0.06544921  0.9059041 0.07224740 4.693434   491
[10] {bmi_group=Obesity,
      is_educated=yes,
      smoker=yes,
      yearly_physical=No}      => {Expensive=yes} 0.06478272  0.8933824 0.07251400 4.628560   486
[11] {bmi_group=Obesity,
      is_educated=yes,
      smoker=yes,
      location_type=Urban}     => {Expensive=yes} 0.06384964  0.8886827 0.07184751 4.604211   479
[12] {bmi_group=Obesity,
      smoker=yes,
      gender=male}             => {Expensive=yes} 0.06238336  0.8897338 0.07011464 4.609657   468
[13] {bmi_group=Obesity,
      smoker=yes,
      exercise=Not-Active,
      hypertension=no}         => {Expensive=yes} 0.06118368  0.9724576 0.06291656 5.038244   459
[14] {bmi_group=Obesity,
      have_child=yes,
      smoker=yes}              => {Expensive=yes} 0.05918422  0.9192547 0.06438283 4.762603   444
[15] {bmi_group=Obesity,
      is_educated=yes,
      smoker=yes,
      married=Married}         => {Expensive=yes} 0.05865103  0.9034908 0.06491602 4.680931   440
[16] {bmi_group=Obesity,
```

```
     smoker=yes,
     yearly_physical=No,
     exercise=Not-Active}   => {Expensive=yes} 0.05851773  0.9799107 0.05971741 5.076858   439
[17] {bmi_group=Obesity,
     smoker=yes,
     location_type=Urban,
     exercise=Not-Active}   => {Expensive=yes} 0.05798454  0.9688196 0.05985071 5.019395   435
[18] {bmi_group=Obesity,
     smoker=yes,
     yearly_physical=No,
     hypertension=no}       => {Expensive=yes} 0.05678486  0.8912134 0.06371634 4.617322   426
[19] {bmi_group=Obesity,
     smoker=yes,
     location_type=Urban,
     hypertension=no}       => {Expensive=yes} 0.05545188  0.8832272 0.06278326 4.575946   416
[20] {bmi_group=Obesity,
     is_educated=yes,
     smoker=yes,
     gender=male}           => {Expensive=yes} 0.05505199  0.8881720 0.06198347 4.601565   413
[21] {bmi_group=Obesity,
     is_educated=yes,
     smoker=yes,
     exercise=Not-Active,
     hypertension=no}       => {Expensive=yes} 0.05438550  0.9691211 0.05611837 5.020958   408
[22] {bmi_group=Obesity,
     is_educated=yes,
     have_child=yes,
     smoker=yes}            => {Expensive=yes} 0.05385231  0.9160998 0.05878432 4.746257   404
[23] {bmi_group=Obesity,
     smoker=yes,
     location_type=Urban,
     yearly_physical=No}    => {Expensive=yes} 0.05371901  0.8837719 0.06078379 4.578769   403
[24] {bmi_group=Obesity,
     smoker=yes,
     exercise=Not-Active,
     married=Married}       => {Expensive=yes} 0.05358571  0.9781022 0.05478539 5.067488   402
[25] {bmi_group=Obesity,
     smoker=yes,
     location=PENNSYLVANIA} => {Expensive=yes} 0.05345241  0.9051919 0.05905092 4.689744   401
[26] {bmi_group=Obesity,
     is_educated=yes,
     smoker=yes,
     yearly_physical=No,
     exercise=Not-Active}   => {Expensive=yes} 0.05198614  0.9774436 0.05318582 5.064076   390
[27] {bmi_group=Obesity,
     is_educated=yes,
     smoker=yes,
     location_type=Urban,
     exercise=Not-Active}   => {Expensive=yes} 0.05171954  0.9651741 0.05358571 5.000509   388
[28] {bmi_group=Obesity,
     smoker=yes,
     married=Married,
```

```
      hypertension=no}       => {Expensive=yes} 0.05091975  0.8967136 0.05678486 4.645819   382
[29] {bmi_group=Obesity,
      smoker=yes,
      exercise=Not-Active,
      gender=male}           => {Expensive=yes} 0.05065316  0.9718670 0.05211943 5.035184   380
```

[Comments] The most supported association here indicated that expensiveness relates to bmi and smoker.

#Further Exploration with Unsupervised Machine Learning Since we manually picked the boundary for determining expensive or not, we now used unsupervised learning and performed k-means clustering to get more insights on cost. According to associate mining and the bar graph, bmi might be a most significant predictor of cost. We used bmi and cost to create clusters.

Hide

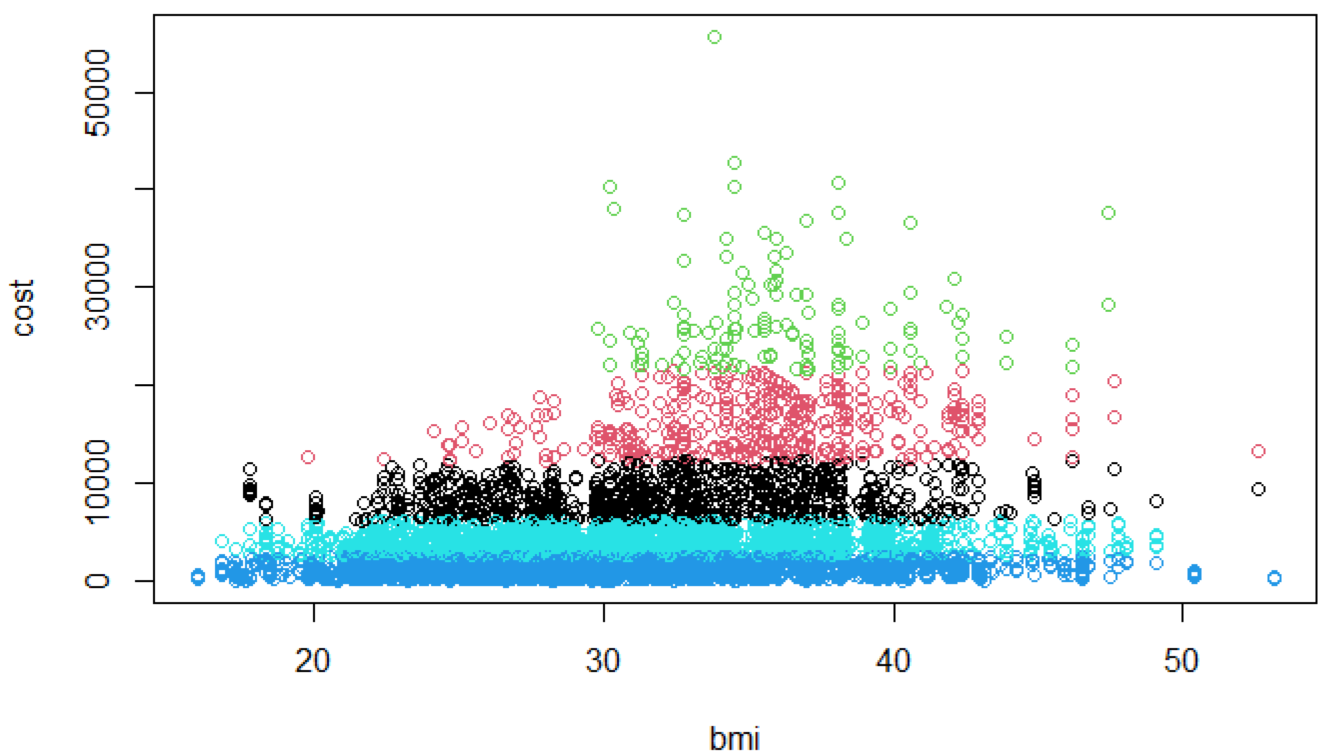
```
df_kmeans <- select(df_new,bmi,cost)
```

Hide

```
set.seed(250)
kmeans_model <- kmeans(df_kmeans,5, iter.max = 10, nstart = 1)
```

Hide

```
plot(df_kmeans, col = kmeans_model$cluster)
```



Hide

```
aggregate(df_kmeans, by=list(cluster=kmeans_model$cluster), mean)
```

cluster <int>	bmi <dbl>	cost <dbl>
1	31.83869	8483.428
2	35.31261	16039.874
3	36.17955	27066.702
4	29.89095	1101.031
5	30.79415	3880.329

5 rows

Hide

```
dd <- cbind(df_kmeans, cluster = kmeans_model$cluster)
lowest_cost_cluster_2 <- dd %>% filter(cluster==2) %>% arrange(by=cost) %>% head(1)
lowest_cost_cluster_2
```

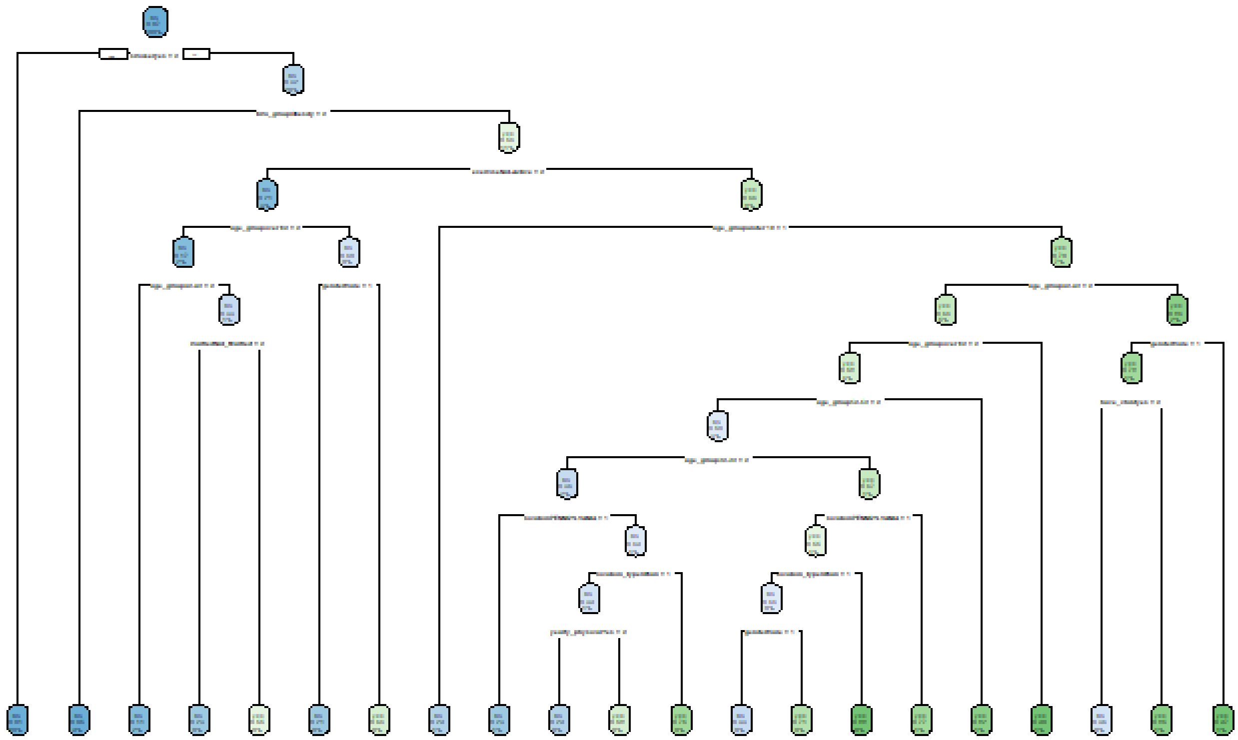
	bmi <dbl>	cost <int>	cluster <int>
1	37.525	12282	2

1 row

[Comments] When we divided the cost into five groups, the lowest cost of the first cluster at the top could be considered as the boundary. We then changed the boundary to 12282 and tested it out with the tree models we have.

Hide

```
#Decision Tree Model 2
df_new$Expensive <- ifelse(df_new$cost >= 12282, 'yes', 'no')

dfX2 <- data.frame(age_group = as.factor(df_new$age_group),
                 bmi_group = as.factor(df_new$bmi_group),
                 is_educated = as.factor(df_new$is_educated),
                 have_child = as.factor(df_new$have_child),
                 smoker = as.factor(df_new$smoker),
                 location = as.factor(df_new$location),
                 location_type = as.factor(df_new$location_type),
                 yearly_physical = as.factor(df_new$yearly_physical),
                 exercise = as.factor(df_new$exercise),
                 married = as.factor(df_new$married),
                 hypertension = as.factor(df_new$hypertension),
                 gender = as.factor(df_new$gender),
                 Expensive = as.factor(df_new$Expensive))

#Divide dataframe into train set and test set
set.seed(250)
trainList <- createDataPartition(y=dfX2$Expensive, p=0.70, list=FALSE)
trainSet <- dfX2[trainList,]
testSet <- dfX2[-trainList,]
# Define train control factors, use repeatedcv for 10 times
trctrl <- trainControl(method = "repeatedcv", number = 10)


#Build rpart tree model
tree_model2 <- train(Expensive~., data = trainSet, method = 'rpart', trControl=trctrl, tuneLengt
h = 10)
rpart.plot(tree_model2$finalModel)
```



Hide

```
#test out tree model 2 on test set
treePred2 <- predict(tree_model2, newdata = testSet)
confusionMatrix(treePred2, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  2072   51
       yes   25  102
                                          
               Accuracy : 0.9662          
                 95% CI : (0.9579, 0.9733)
    No Information Rate : 0.932           
    P-Value [Acc > NIR] : 9.847e-13       
                                          
                  Kappa : 0.7107          
                                          
 Mcnemar's Test P-Value : 0.004135        
                                          
            Sensitivity : 0.9881          
            Specificity : 0.6667          
         Pos Pred Value : 0.9760          
         Neg Pred Value : 0.8031          
             Prevalence : 0.9320          
         Detection Rate : 0.9209          
   Detection Prevalence : 0.9436          
      Balanced Accuracy : 0.8274          
                                          
       'Positive' Class : no              
```

[Comments] After adjusting for the boundary, sensitivity and accuracy have improved for the same tree model 2. We would later go through the predictor selection again to get better performance.

Hide

```
varImp(tree_model2)
```

```
rpart variable importance

  only 20 most important variables shown (out of 32)
```

	Overall <dbl>
smokeryes	100.00000000
bmi_groupObesity	92.77190883
exerciseNot-Active	50.64399763
bmi_groupOverweight	32.59837826
age_groupover 60	26.93864328

	Overall <dbl>
age_groupunder 18	21.71253774
age_group40-49	19.33218443
have_childyes	15.79604425
age_group50-59	11.74637826
location_typeUrban	11.53905654

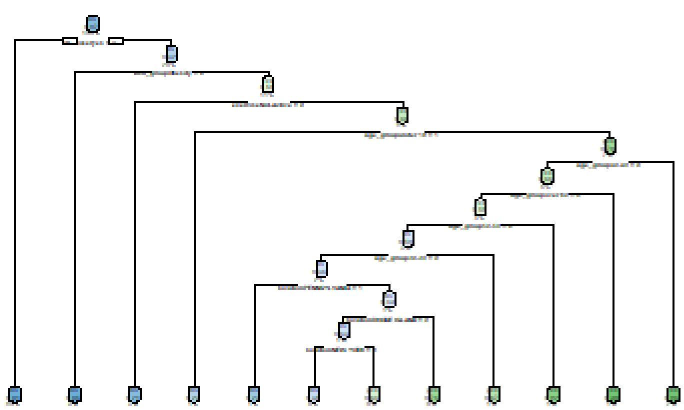
1-10 of 20 rows Previous 1 2 Next

Hide

```
#There are only 20 out of 32 variables are important.We then excluded some of the predictors tha
t are less important.
```

Hide

```
#Decision Tree Model 3
trainSet <- select(trainSet, -gender, -have_child, -hypertension, -yearly_physical, -location_ty
pe, -married, -is_educated)
tree_model3 <- train(Expensive~., data = trainSet, method = 'rpart', trControl=trctrl, tuneLengt
h = 10)
rpart.plot(tree_model3$finalModel)
```



Hide

```
treePred3 <- predict(tree_model3, newdata = testSet)
confusionMatrix(treePred3, as.factor(testSet$Expensive))
```

```
Confusion Matrix and Statistics

          Reference
Prediction   no  yes
       no  2072   46
       yes   25  107

               Accuracy : 0.9684
                 95% CI : (0.9604, 0.9753)
    No Information Rate : 0.932
    P-Value [Acc > NIR] : 2.036e-14

                  Kappa : 0.7341

 Mcnemar's Test P-Value : 0.01762

            Sensitivity : 0.9881
            Specificity : 0.6993
         Pos Pred Value : 0.9783
         Neg Pred Value : 0.8106
             Prevalence : 0.9320
         Detection Rate : 0.9209
   Detection Prevalence : 0.9413
      Balanced Accuracy : 0.8437

       'Positive' Class : no
```

[Comments] The decision tree model 3 returned to significant accuracy of 0.9684 and sensitivity of 0.9881. This is considered our final model with age_group, bmi_group, smoker, location and exercise as the predictors.

#Storing the model for shinny apps

Hide

```
#storing the model
datafile <- "https://intro-datascience.s3.us-east-2.amazonaws.com/HMO_data.csv"
df_raw <- read.csv(datafile)
df_raw$bmi <- na_interpolation(df_raw$bmi)
df_raw <- df_raw %>% filter(!is.na(hypertension))

df_add_age <- df_raw  %>% mutate(age_group = case_when(
  df_raw$age < 20 ~ "under 18",
  df_raw$age >= 20 & df_raw$age < 30 ~ "20-29",
  df_raw$age >= 30 & df_raw$age < 40 ~ "30-39",
  df_raw$age >= 40 & df_raw$age < 50 ~ "40-49",
  df_raw$age >= 50 & df_raw$age < 60 ~ "50-59",
  df_raw$age >= 60 ~ 'over 60'
))

df_add_bmi <- df_add_age %>% mutate(bmi_group = case_when(
  df_add_age$bmi < 18.5 ~ "Underweight",
  df_add_age$bmi >= 18.5 & df_add_age$bmi < 24.9 ~ "Normal Weight",
  df_add_age$bmi >= 24.9 & df_add_age$bmi < 29.9 ~ "Overweight",
  df_add_age$bmi >= 29.9 ~ "Obesity"
))
df_new <- df_add_bmi
df_add_edu_bin <- df_new %>% mutate(is_educated = case_when(
  df_new$education_level != "No College Degree" ~ "yes",
  TRUE ~ "no"
))



df_add_child_bin <- df_add_edu_bin %>% mutate(have_child = case_when(
  df_add_edu_bin$children == 0 ~ "no",
  TRUE ~ "yes"
))

df_new <- df_add_child_bin
df_new$hypertension <- ifelse(df_new$hypertension==1, 'yes', 'no')
df_new$Expensive <- ifelse(df_new$cost >= 12282, 'yes', 'no')

df <- data.frame(age_group = as.factor(df_new$age_group),
                 bmi_group = as.factor(df_new$bmi_group),
                 smoker = as.factor(df_new$smoker),
                 location = as.factor(df_new$location),
                 yearly_physical = as.factor(df_new$yearly_physical),
                 exercise = as.factor(df_new$exercise),
                 Expensive = as.factor(df_new$Expensive))
trctrl <- trainControl(method = "repeatedcv", number = 10)
our_model <- train(Expensive~., data = df, method = 'rpart', trControl=trctrl, tuneLength = 10)
save(our_model, file="our_model.rda")
```