Fertility Data

Sanjay

2023-07-08

#Leading a long and healthy life is a priority as humans. Especially in this day and age, we need to keep track of the countless factors that could be having impacts on our well being and longevity. This data set presents numerous factors that one could experience in a lifetime that may have extreme effects on ones health. Thus, I wanted to how much each of these variables play a role in ones health, and see if I can use them optimize a long and safe life.

#This dataset includes 10 variables: Season, Age, Childish diseases, accident or serious trauma, surgical intervention, high fevers in the last year, frequency of alcohol consumption, smoking habits, hours spent sitting per day, and the condition of each respondents semen. Childish diseases signifies whether or not the individual had a disease as a child, accident or serious trauma corresponds to whether or not the individual has suffered a recent accident or had serious trauma. All of these variables are general indicators of health for men in this case.

#Using the data, I want to see which variables have impacts on other variables. I am curious to see which factors have the most impact on semen condition. I also want to investigate which factors influence ones lifestyle habits. Lastly, I want to investigate the ways to predict whether an individual will experience trauma, and the effects of this trauma.

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.1.3
```

```
## Warning: package 'ggplot2' was built under R version 4.1.3
```

```
## Warning: package 'tibble' was built under R version 4.1.3
```

```
## Warning: package 'tidyr' was built under R version 4.1.3
```

```
## Warning: package 'readr' was built under R version 4.1.3
```

```
## Warning: package 'purrr' was built under R version 4.1.3
```

```
## Warning: package 'dplyr' was built under R version 4.1.3
```

```
## Warning: package 'stringr' was built under R version 4.1.3
```

```
## Warning: package 'forcats' was built under R version 4.1.3
```

```
## Warning: package 'lubridate' was built under R version 4.1.3
## -- Attaching core tidyverse packages ----------------------- tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
library(modelr)
```

```
## Warning: package 'modelr' was built under R version 4.1.3
```

```
library(broom)
```

```
## Warning: package 'broom' was built under R version 4.1.3
```

```
##
## Attaching package: 'broom'
##
## The following object is masked from 'package:modelr':
##
##     bootstrap
```

```
library(purrr)
library(Metrics)
```

```
## Warning: package 'Metrics' was built under R version 4.1.3
```

```
##
## Attaching package: 'Metrics'
##
## The following objects are masked from 'package:modelr':
##
##     mae, mape, mse, rmse
```

```
library(dplyr)
library(ggplot2)
library(Hmisc)
```

```
## Warning: package 'Hmisc' was built under R version 4.1.3
```

```
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:dplyr':
##
##     src, summarize
```

```
##
## The following objects are masked from 'package:base':
##
##     format.pval, units
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(plotly)
```

```
##
## Attaching package: 'plotly'
##
## The following object is masked from 'package:Hmisc':
##
##     subplot
##
## The following object is masked from 'package:ggplot2':
##
##     last_plot
##
## The following object is masked from 'package:stats':
##
##     filter
##
## The following object is masked from 'package:graphics':
##
##     layout
```

```
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 4.1.3
```

```
##
## Attaching package: 'reshape2'
##
## The following object is masked from 'package:tidyr':
##
##     smiths
```

```
library(leaps)
```

```
## Warning: package 'leaps' was built under R version 4.1.3
```

```
library(margins)
```

```
## Warning: package 'margins' was built under R version 4.1.3
```

```
library(yardstick)
```

```
## Warning: package ’yardstick’ was built under R version 4.1.3
```

```
## For binary classification, the first factor level is assumed to be the event.
## Use the argument ‘event_level = "second"‘ to alter this as needed.
##
## Attaching package: ’yardstick’
##
## The following objects are masked from ’package:Metrics’:
##
##     accuracy, mae, mape, mase, precision, recall, rmse, smape
##
## The following objects are masked from ’package:modelr’:
##
##     mae, mape, rmse
##
## The following object is masked from ’package:readr’:
##
##     spec
```

```
library(randomForest)
```

```
## Warning: package ’randomForest’ was built under R version 4.1.3
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: ’randomForest’
##
## The following object is masked from ’package:dplyr’:
##
##     combine
##
## The following object is masked from ’package:ggplot2’:
##
##     margin
```

```
library(caret)
```

```
## Warning: package ’caret’ was built under R version 4.1.3
```

```
## Loading required package: lattice
##
## Attaching package: ’caret’
##
## The following objects are masked from ’package:yardstick’:
##
##     precision, recall, sensitivity, specificity
##
## The following objects are masked from ’package:Metrics’:
```

```
##
##     precision, recall
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(MASS)
```

```
## Warning: package 'MASS' was built under R version 4.1.3
```

```
##
## Attaching package: 'MASS'
##
## The following object is masked from 'package:plotly':
##
##     select
##
## The following object is masked from 'package:dplyr':
##
##     select
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 4.1.3
```

```
## Loading required package: carData
##
## Attaching package: 'car'
##
## The following object is masked from 'package:dplyr':
##
##     recode
##
## The following object is masked from 'package:purrr':
##
##     some
```

```
library(naivebayes)
```

```
## Warning: package 'naivebayes' was built under R version 4.1.3
```

```
## naivebayes 0.9.7 loaded
```

```
library(bestglm)
```

```
## Warning: package 'bestglm' was built under R version 4.1.3
```

```
library(pROC)
```

```
## Warning: package ’pROC’ was built under R version 4.1.3

## Type ’citation("pROC")’ for a citation.
##
## Attaching package: ’pROC’
##
## The following object is masked from ’package:Metrics’:
##
##     auc
##
## The following objects are masked from ’package:stats’:
##
##     cov, smooth, var
```

```
fert = read_csv('fertility.csv')
```

```
## Rows: 100 Columns: 10
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (8): Season, Childish diseases, Accident or serious trauma, Surgical int...
## dbl (2): Age, Number of hours spent sitting per day
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
head(fert, 10)
```

```
## # A tibble: 10 x 10
##    Season   Age ‘Childish diseases‘ ‘Accident or serious trauma‘
##    <chr>  <dbl> <chr>               <chr>
##  1 spring    30 no                  yes
##  2 spring    35 yes                 no
##  3 spring    27 yes                 no
##  4 spring    32 no                  yes
##  5 spring    30 yes                 yes
##  6 spring    30 yes                 no
##  7 spring    30 no                  no
##  8 spring    36 yes                 yes
##  9 fall      30 no                  no
## 10 fall      29 yes                 no
## # i 6 more variables: ‘Surgical intervention‘ <chr>,
## #   ‘High fevers in the last year‘ <chr>,
## #   ‘Frequency of alcohol consumption‘ <chr>, ‘Smoking habit‘ <chr>,
## #   ‘Number of hours spent sitting per day‘ <dbl>, Diagnosis <chr>
```

#data cleaning

```r
dim(fert)
```

```
## [1] 100  10
```

```r
columns = c(colnames(fert))
columns
```

```
##  [1] "Season"
##  [2] "Age"
##  [3] "Childish diseases"
##  [4] "Accident or serious trauma"
##  [5] "Surgical intervention"
##  [6] "High fevers in the last year"
##  [7] "Frequency of alcohol consumption"
##  [8] "Smoking habit"
##  [9] "Number of hours spent sitting per day"
## [10] "Diagnosis"
```

```r
#Find unique values for each variable.
for (i in columns) {
 uniquevals = unique(fert[i])
 print(uniquevals)
}
```

```
## # A tibble: 4 x 1
##   Season
##   <chr>
## 1 spring
## 2 fall
## 3 winter
## 4 summer
## # A tibble: 10 x 1
##       Age
##     <dbl>
##  1    30
##  2    35
##  3    27
##  4    32
##  5    36
##  6    29
##  7    33
##  8    28
##  9    31
## 10    34
## # A tibble: 2 x 1
##   ‘Childish diseases‘
##   <chr>
## 1 no
## 2 yes
## # A tibble: 2 x 1
##   ‘Accident or serious trauma‘
##   <chr>
## 1 yes
## 2 no
## # A tibble: 2 x 1
##   ‘Surgical intervention‘
##   <chr>
## 1 yes
## 2 no
## # A tibble: 3 x 1
##   ‘High fevers in the last year‘
##   <chr>
## 1 more than 3 months ago
## 2 less than 3 months ago
## 3 no
## # A tibble: 5 x 1
##   ‘Frequency of alcohol consumption‘
##   <chr>
## 1 once a week
## 2 hardly ever or never
## 3 several times a week
## 4 several times a day
## 5 every day
## # A tibble: 3 x 1
##   ‘Smoking habit‘
##   <chr>
## 1 occasional
## 2 daily
## 3 never
## # A tibble: 14 x 1
##    ‘Number of hours spent sitting per day‘
##                                       <dbl>
##  1                                       16
##  2                                        6
##  3                                        9
##  4                                        7
##  5                                        8
##  6                                        5
##  7                                        2
##  8                                       11
##  9                                        3
## 10                                      342
## 11                                       14
## 12                                       18
## 13                                       10
## 14                                        1
## # A tibble: 2 x 1
##   Diagnosis
##   <chr>
## 1 Normal
## 2 Altered
```

```
#rename columns
fert1 = fert %>%
  rename('child_disease'= 'Childish diseases' , 'trauma' = 'Accident or serious trauma' , 'surgery' = '
fert1
```

```
## # A tibble: 100 x 10
##    Season   Age child_disease trauma surgery fever           alcohol_consumption
##    <chr>  <dbl> <chr>         <chr>  <chr>   <chr>           <chr>
##  1 spring    30 no            yes    yes     more than 3 mo~ once a week
##  2 spring    35 yes           no     yes     more than 3 mo~ once a week
##  3 spring    27 yes           no     no      more than 3 mo~ hardly ever or nev~
##  4 spring    32 no            yes    yes     more than 3 mo~ hardly ever or nev~
##  5 spring    30 yes           yes    no      more than 3 mo~ once a week
##  6 spring    30 yes           no     yes     more than 3 mo~ once a week
##  7 spring    30 no            no     no      less than 3 mo~ once a week
##  8 spring    36 yes           yes    yes     more than 3 mo~ several times a we~
##  9 fall      30 no            no     yes     more than 3 mo~ once a week
## 10 fall      29 yes           no     no      more than 3 mo~ hardly ever or nev~
## # i 90 more rows
## # i 3 more variables: smoking_frequency <chr>, hours_sitting_daily <dbl>,
## #   Diagnosis <chr>
```

```
#convert categorical data into factors
fert1$Season = as.factor(fert1$Season)
fert1$fever = as.factor(fert1$fever)
fert1$alcohol_consumption = as.factor(fert1$alcohol_consumption)
fert1$smoking_frequency = as.factor(fert1$smoking_frequency)

str(fert1)
```

```
## spc_tbl_ [100 x 10] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Season             : Factor w/ 4 levels "fall","spring",..: 2 2 2 2 2 2 2 2 1 1 ...
##  $ Age                : num [1:100] 30 35 27 32 30 30 30 36 30 29 ...
##  $ child_disease      : chr [1:100] "no" "yes" "yes" "no" ...
##  $ trauma             : chr [1:100] "yes" "no" "no" "yes" ...
##  $ surgery            : chr [1:100] "yes" "yes" "no" "yes" ...
##  $ fever              : Factor w/ 3 levels "less than 3 months ago",..: 2 2 2 2 2 2 1 2 2 2 ...
##  $ alcohol_consumption: Factor w/ 5 levels "every day","hardly ever or never",..: 3 3 2 2 3 3 3 5 3 2
##  $ smoking_frequency  : Factor w/ 3 levels "daily","never",..: 3 1 2 2 2 3 2 2 2 2 ...
##  $ hours_sitting_daily: num [1:100] 16 6 9 7 9 9 8 7 5 5 ...
##  $ Diagnosis          : chr [1:100] "Normal" "Altered" "Normal" "Normal" ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Season = col_character(),
##   ..   Age = col_double(),
##   ..   ‘Childish diseases‘ = col_character(),
##   ..   ‘Accident or serious trauma‘ = col_character(),
##   ..   ‘Surgical intervention‘ = col_character(),
##   ..   ‘High fevers in the last year‘ = col_character(),
##   ..   ‘Frequency of alcohol consumption‘ = col_character(),
##   ..   ‘Smoking habit‘ = col_character(),
##   ..   ‘Number of hours spent sitting per day‘ = col_double(),
##   ..   Diagnosis = col_character()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
#convert binary data into numeric
for (i in 1:100) {
  if (fert1$child_disease[i] == 'no') {
    fert1$child_disease[i] = '0'

  } else {
    fert1$child_disease[i] = '1'
  }
}

for (i in 1:100) {
  if (fert1$trauma[i] == 'no') {
    fert1$trauma[i] = '0'

  } else {
    fert1$trauma[i] = '1'
  }
}

for (i in 1:100) {
  if (fert1$surgery[i] == 'no') {
    fert1$surgery[i] = '0'

  } else {
    fert1$surgery[i] = '1'
  }

}

for (i in 1:100) {
  if (fert1$Diagnosis[i] == 'Normal') {
    fert1$Diagnosis[i] = '0'

  } else {
    fert1$Diagnosis[i] = '1'
  }
}

fert1$child_disease = as.numeric(fert1$child_disease)
fert1$trauma = as.numeric(fert1$trauma)
fert1$surgery = as.numeric(fert1$surgery)
fert1$Diagnosis = as.numeric(fert1$Diagnosis)

#All binary response variables were recoded as 1 being yes, and 0 being no. Diagnosis was recoded as 0

fert1
```

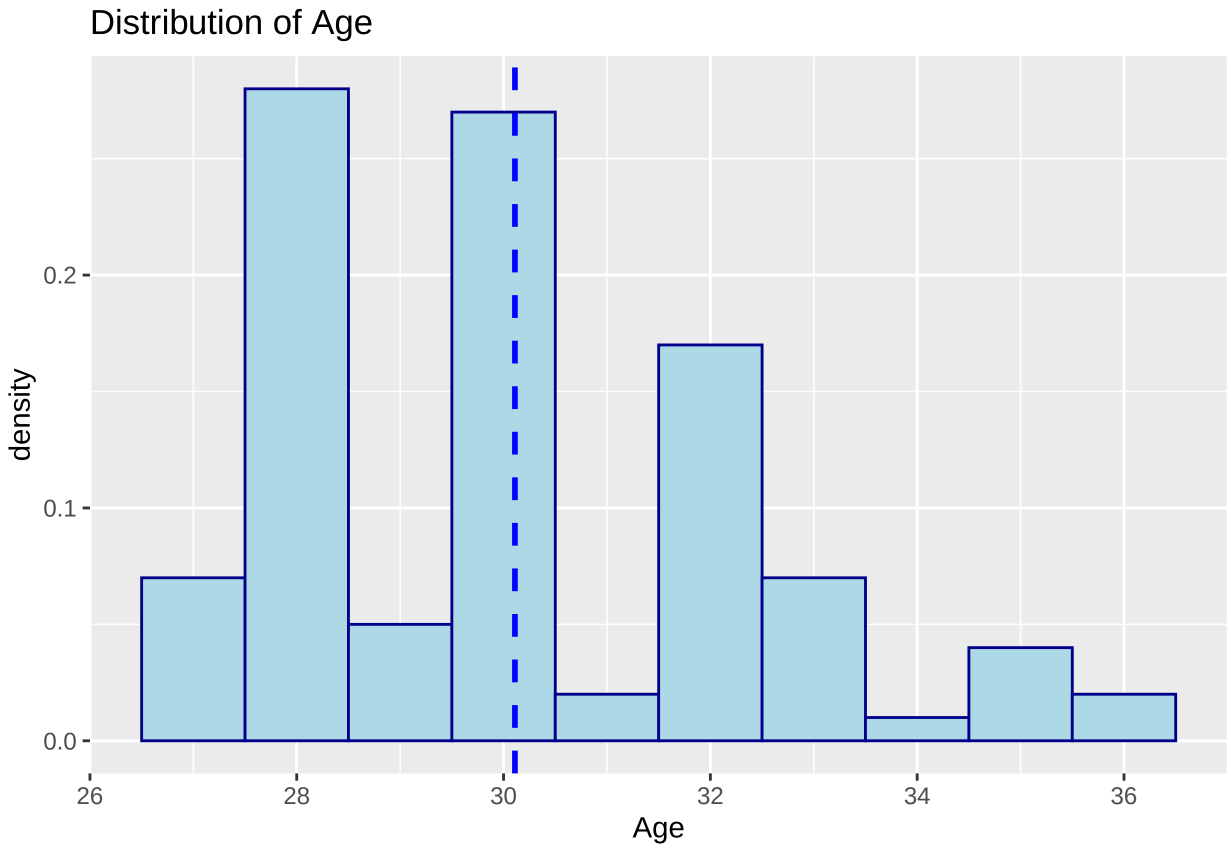
```
## # A tibble: 100 x 10
##    Season   Age child_disease trauma surgery fever           alcohol_consumption
##    <fct>  <dbl>         <dbl>  <dbl>   <dbl> <fct>           <fct>
##  1 spring    30             0      1       1 more than 3 mo~ once a week
##  2 spring    35             1      0       1 more than 3 mo~ once a week
##  3 spring    27             1      0       0 more than 3 mo~ hardly ever or nev~
##  4 spring    32             0      1       1 more than 3 mo~ hardly ever or nev~
##  5 spring    30             1      1       0 more than 3 mo~ once a week
##  6 spring    30             1      0       1 more than 3 mo~ once a week
##  7 spring    30             0      0       0 less than 3 mo~ once a week
##  8 spring    36             1      1       1 more than 3 mo~ several times a we~
##  9 fall      30             0      0       1 more than 3 mo~ once a week
## 10 fall      29             1      0       0 more than 3 mo~ hardly ever or nev~
## # i 90 more rows
## # i 3 more variables: smoking_frequency <fct>, hours_sitting_daily <dbl>,
## #   Diagnosis <dbl>
```

#Visualizing distributions of quantitative variables.

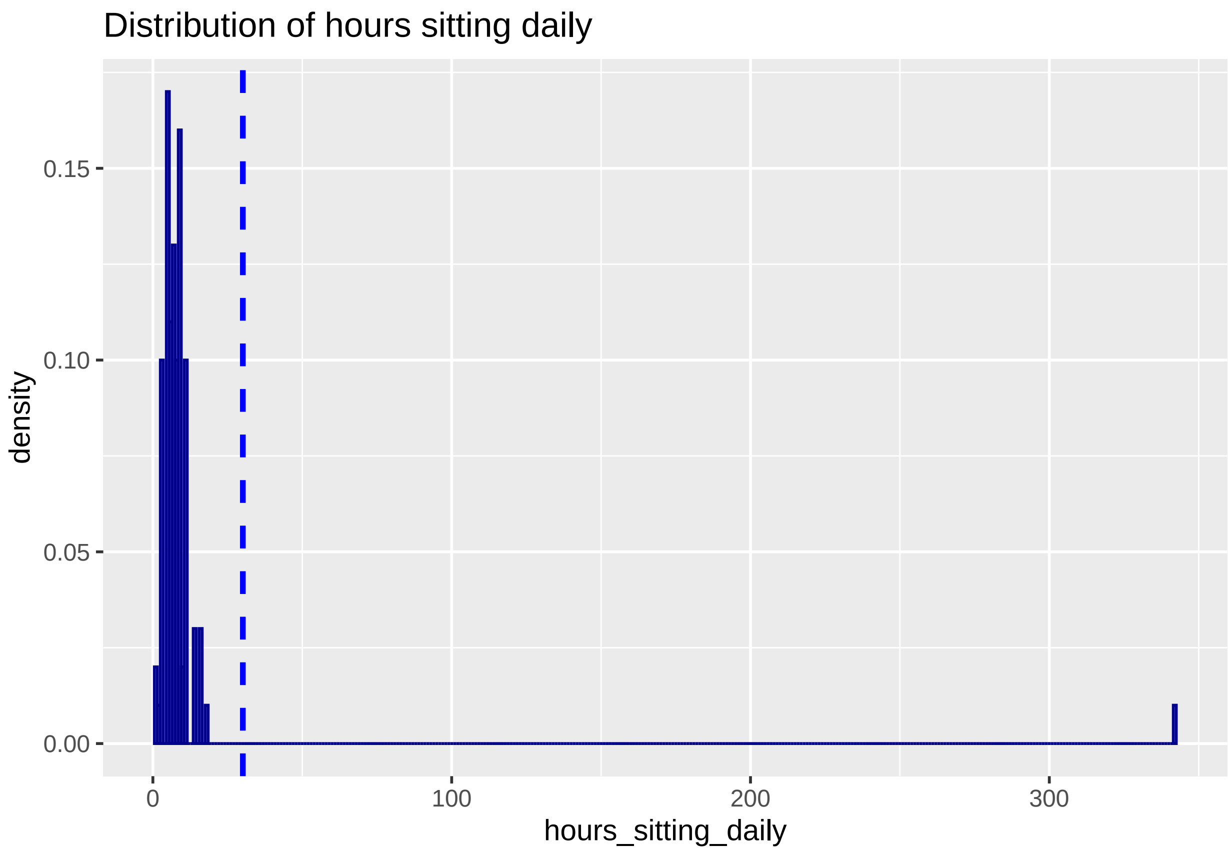
```
ggplot(fert1, aes(x=Age)) + geom_histogram(aes(y=..density..), colour="darkblue", fill="lightblue" , bi
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

```
## Warning: The dot-dot notation (`..density..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(density)` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```

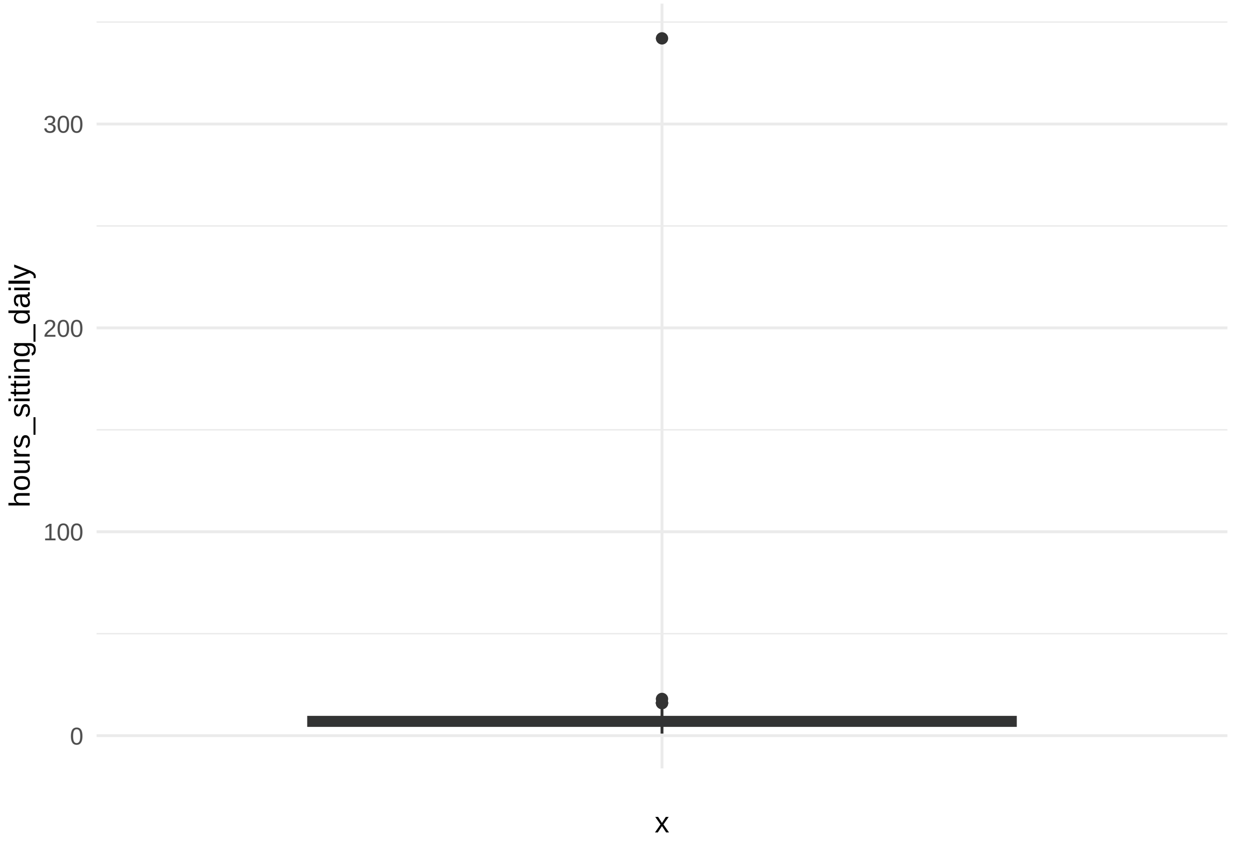


```
ggplot(fert1, aes(x=hours_sitting_daily)) + geom_histogram(aes(y=..density..), colour="darkblue", fill=
```



```
#Based on this histogram there seems to be a massive outlier.
```

```
#Checking outlier with a boxplot
ggplot(fert1) +
  aes(x = "", y = hours_sitting_daily) +
  geom_boxplot(fill = "#0c4c8a") +
  theme_minimal()
```



```
#Based on the boxplot this point is a massive outlier, as it lies far outside the interquartile range.
```

```
fert1 %>% filter(hours_sitting_daily >300)
```

```
## # A tibble: 1 x 10
##   Season   Age child_disease trauma surgery fever            alcohol_consumption
##   <fct>  <dbl>         <dbl>  <dbl>   <dbl> <fct>            <fct>
## 1 spring    30             1      0       1 more than 3 mon~ once a week
## # i 3 more variables: smoking_frequency <fct>, hours_sitting_daily <dbl>,
## #   Diagnosis <dbl>
```

```
#Identifying the outlier.
```

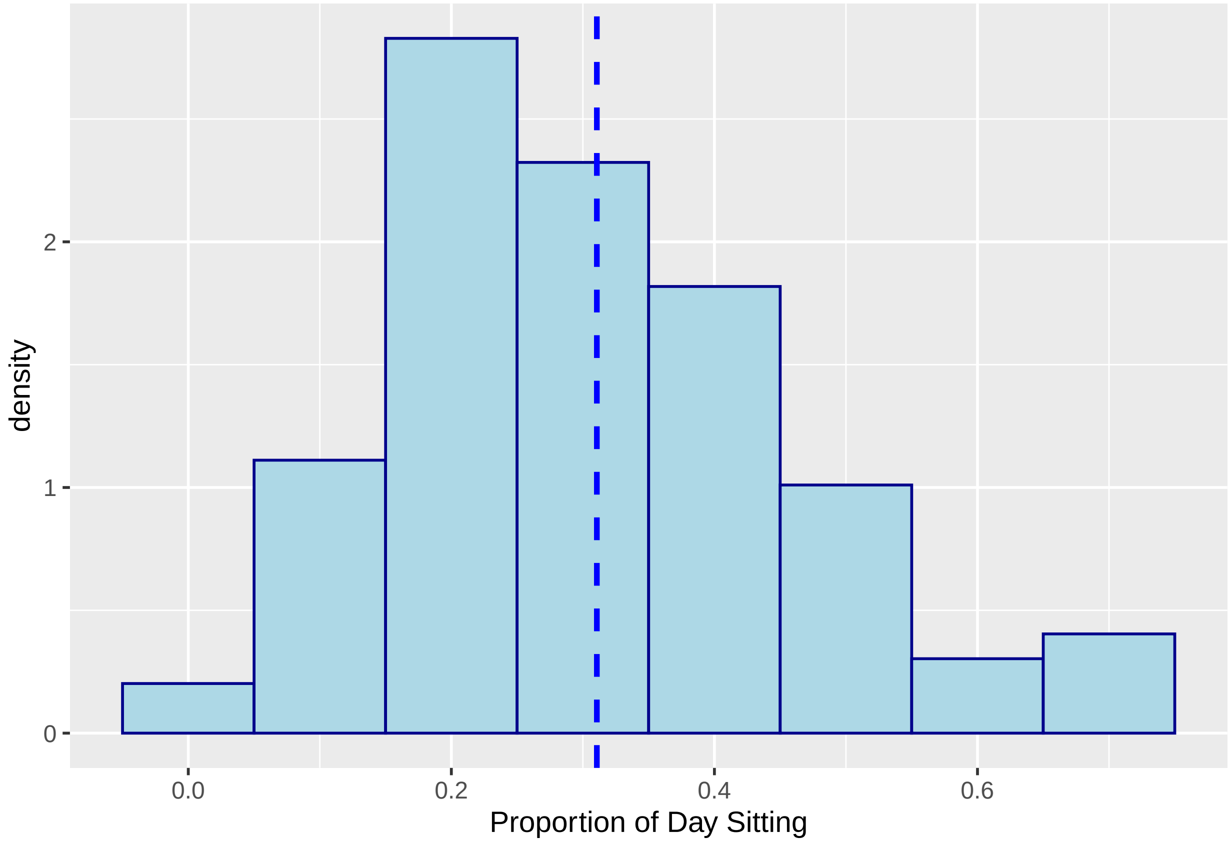
```
#Dropping the outlier.
fert2 = subset(fert1,hours_sitting_daily < 300 )
fert2
```

```
## # A tibble: 99 x 10
##    Season   Age child_disease trauma surgery fever           alcohol_consumption
##    <fct>  <dbl>         <dbl>  <dbl>   <dbl> <fct>           <fct>
##  1 spring    30             0      1       1 more than 3 mo~ once a week
##  2 spring    35             1      0       1 more than 3 mo~ once a week
##  3 spring    27             1      0       0 more than 3 mo~ hardly ever or nev~
##  4 spring    32             0      1       1 more than 3 mo~ hardly ever or nev~
##  5 spring    30             1      1       0 more than 3 mo~ once a week
##  6 spring    30             1      0       1 more than 3 mo~ once a week
##  7 spring    30             0      0       0 less than 3 mo~ once a week
##  8 spring    36             1      1       1 more than 3 mo~ several times a we~
##  9 fall      30             0      0       1 more than 3 mo~ once a week
## 10 fall      29             1      0       0 more than 3 mo~ hardly ever or nev~
## # i 89 more rows
## # i 3 more variables: smoking_frequency <fct>, hours_sitting_daily <dbl>,
## #   Diagnosis <dbl>
```

```
for (i in 1:length(fert2$hours_sitting_daily)) {
  fert2$hours_sitting_daily[i] = (fert2$hours_sitting_daily[i])/24

}

fert3 = fert2 %>% rename("prop_day_sitting" = 'hours_sitting_daily')

fert3
```

```
## # A tibble: 99 x 10
##    Season   Age child_disease trauma surgery fever           alcohol_consumption
##    <fct>  <dbl>         <dbl>  <dbl>   <dbl> <fct>           <fct>
##  1 spring    30             0      1       1 more than 3 mo~ once a week
##  2 spring    35             1      0       1 more than 3 mo~ once a week
##  3 spring    27             1      0       0 more than 3 mo~ hardly ever or nev~
##  4 spring    32             0      1       1 more than 3 mo~ hardly ever or nev~
##  5 spring    30             1      1       0 more than 3 mo~ once a week
##  6 spring    30             1      0       1 more than 3 mo~ once a week
##  7 spring    30             0      0       0 less than 3 mo~ once a week
##  8 spring    36             1      1       1 more than 3 mo~ several times a we~
##  9 fall      30             0      0       1 more than 3 mo~ once a week
## 10 fall      29             1      0       0 more than 3 mo~ hardly ever or nev~
## # i 89 more rows
## # i 3 more variables: smoking_frequency <fct>, prop_day_sitting <dbl>,
## #   Diagnosis <dbl>
```

```
#Recoding hour_sitting_daily to proportion of day sitting to make it easier to work with.

ggplot(fert3, aes(x=prop_day_sitting)) + geom_histogram(aes(y=..density..), colour="darkblue", fill="li
```



```
summary(fert3)
```

```
##     Season        Age        child_disease        trauma          surgery      
##  fall  :31   Min.   :27.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000  
##  spring:36   1st Qu.:28.00   1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:0.0000  
##  summer: 4   Median :30.00   Median :1.0000   Median :0.0000   Median :1.0000  
##  winter:28   Mean   :30.11   Mean   :0.8687   Mean   :0.4444   Mean   :0.5051  
##              3rd Qu.:32.00   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000  
##              Max.   :36.00   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000  
##                     fever              alcohol_consumption  smoking_frequency
##  less than 3 months ago: 9   every day           : 1       daily     :21     
##  more than 3 months ago:62   hardly ever or never:40       never     :55     
##  no                    :28   once a week         :38       occasional:23     
##                              several times a day : 1                         
##                              several times a week:19                         
##                                                                              
##  prop_day_sitting    Diagnosis     
##  Min.   :0.04167   Min.   :0.0000  
##  1st Qu.:0.20833   1st Qu.:0.0000  
##  Median :0.29167   Median :0.0000  
##  Mean   :0.31061   Mean   :0.1212  
##  3rd Qu.:0.37500   3rd Qu.:0.0000  
##  Max.   :0.75000   Max.   :1.0000
```

```
#since there is only 1 response for alcohol consumption several times a day and every day, I am going t
```

```
fert4 = fert3 %>%
 mutate(alcohol_consumption = fct_collapse(alcohol_consumption,
 'several times a week or more' = c("several times a day", "every day", 'several times a week')))
```

```
lifestyle.score = function(smoking,alcohol,sitting,fever){
  score = c(rep(0,99))
  i = 1

  while (i <= 99) {
    if (smoking[i] == 'daily') {
      score[i] = score[i] + 1
    } else if (smoking[i] == 'occasional') {
      score[i] = score[i] + 2
    } else if (smoking[i] == 'never') {
      score[i] = score[i] + 3}

     if (alcohol[i] == 'daily') {
      score[i] = score[i] + 1
    } else if (alcohol[i] == 'once a week') {
      score[i] = score[i] + 2
    } else if (alcohol[i] == 'hardly ever or never') {
      score[i] = score[i] + 3}

     if (sitting[i] < 0.25) {
      score[i] = score[i] + 3
    } else if (between(sitting[i], .25,.50)) {
      score[i] = score[i] + 2
    } else if (sitting[i] > 0.50) {
      score[i] = score[i] + 1}

     if (fever[i] == 'less than 3 months ago') {
      score[i] = score[i] + 0
    } else {
      score[i] = score[i] + 1}
    i = i + 1
    }
  return (score)
}


fert4$lifestyle = lifestyle.score(fert4$smoking_frequency, fert4$alcohol_consumption, fert4$prop_day_si


summary(fert4$lifestyle)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.000   6.000   8.000   7.465   9.000  10.000
```

#Lifestyle score was created using the variables: frequency of alcohol consumption, smoking frequency, recent fever, and proportion of day sitting. A higher lifestyle score is better and vice versa. This might be an interesting variable to predict, to identify risk factors or the effects of a poor lifestyle.

```
summary(fert4)
```

```
##     Season        Age        child_disease        trauma          surgery
##  fall  :31   Min.   :27.00   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  spring:36   1st Qu.:28.00   1st Qu.:1.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  summer: 4   Median :30.00   Median :1.0000   Median :0.0000   Median :1.0000
##  winter:28   Mean   :30.11   Mean   :0.8687   Mean   :0.4444   Mean   :0.5051
##              3rd Qu.:32.00   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##              Max.   :36.00   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##                     fever                      alcohol_consumption
##  less than 3 months ago: 9   several times a week or more:21
##  more than 3 months ago:62   hardly ever or never        :40
##  no                    :28   once a week                 :38
##
##
##
##   smoking_frequency prop_day_sitting    Diagnosis        lifestyle
##  daily     :21      Min.   :0.04167   Min.   :0.0000   Min.   : 4.000
##  never     :55      1st Qu.:0.20833   1st Qu.:0.0000   1st Qu.: 6.000
##  occasional:23      Median :0.29167   Median :0.0000   Median : 8.000
##                     Mean   :0.31061   Mean   :0.1212   Mean   : 7.465
##                     3rd Qu.:0.37500   3rd Qu.:0.0000   3rd Qu.: 9.000
##                     Max.   :0.75000   Max.   :1.0000   Max.   :10.000
```

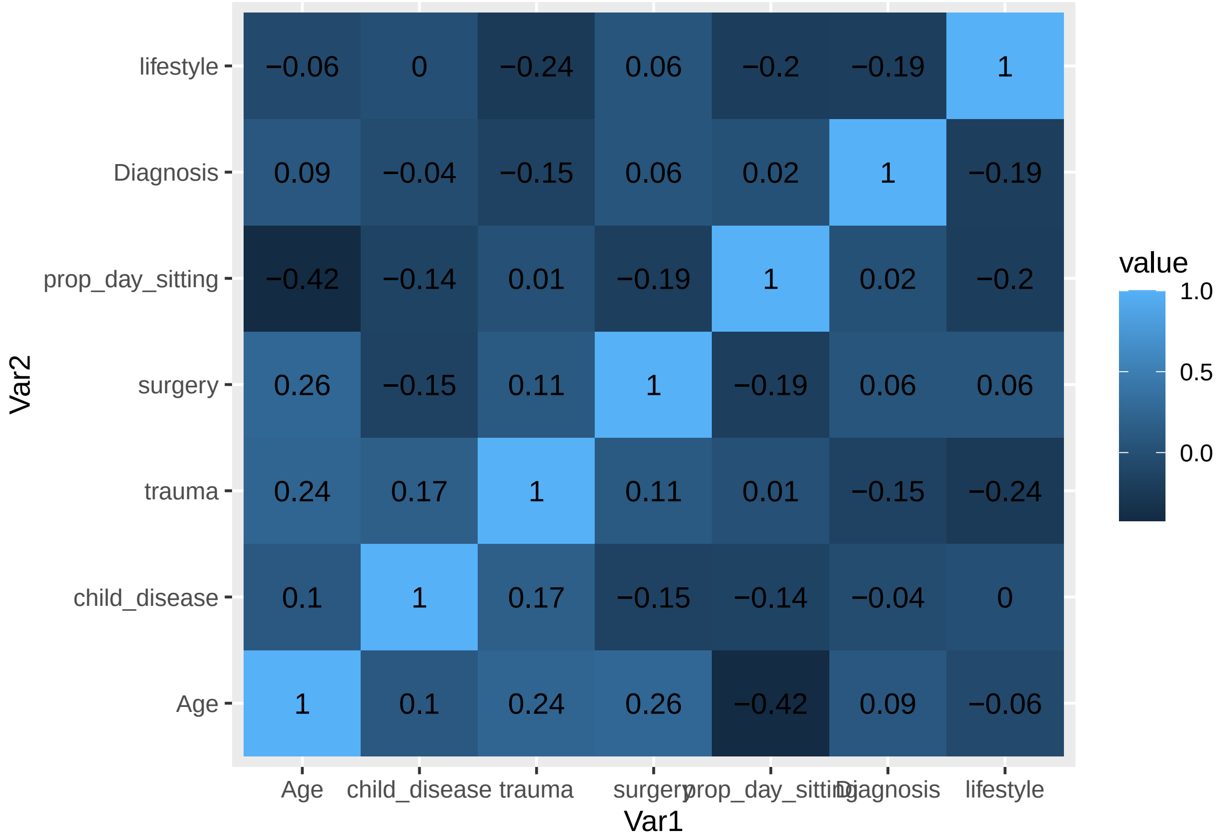
#Visualizing relationships between variables

```
#correlations between numeric variables including diagnosis.
fert_num = fert4 %>% select_if(is.numeric)
correlations = round(cor(fert_num),2)
correlations
```

```
##                      Age child_disease trauma surgery prop_day_sitting Diagnosis
## Age               1.00          0.10   0.24    0.26            -0.42      0.09
## child_disease     0.10          1.00   0.17   -0.15            -0.14     -0.04
## trauma            0.24          0.17   1.00    0.11             0.01     -0.15
## surgery           0.26         -0.15   0.11    1.00            -0.19      0.06
## prop_day_sitting -0.42         -0.14   0.01   -0.19             1.00      0.02
## Diagnosis         0.09         -0.04  -0.15    0.06             0.02      1.00
## lifestyle        -0.06          0.00  -0.24    0.06            -0.20     -0.19
##                  lifestyle
## Age                  -0.06
## child_disease         0.00
## trauma               -0.24
## surgery               0.06
## prop_day_sitting     -0.20
## Diagnosis            -0.19
## lifestyle             1.00
```

```
melted_corr_mat <- melt(correlations)
# head(melted_corr_mat)

# plotting the correlation heatmap
```

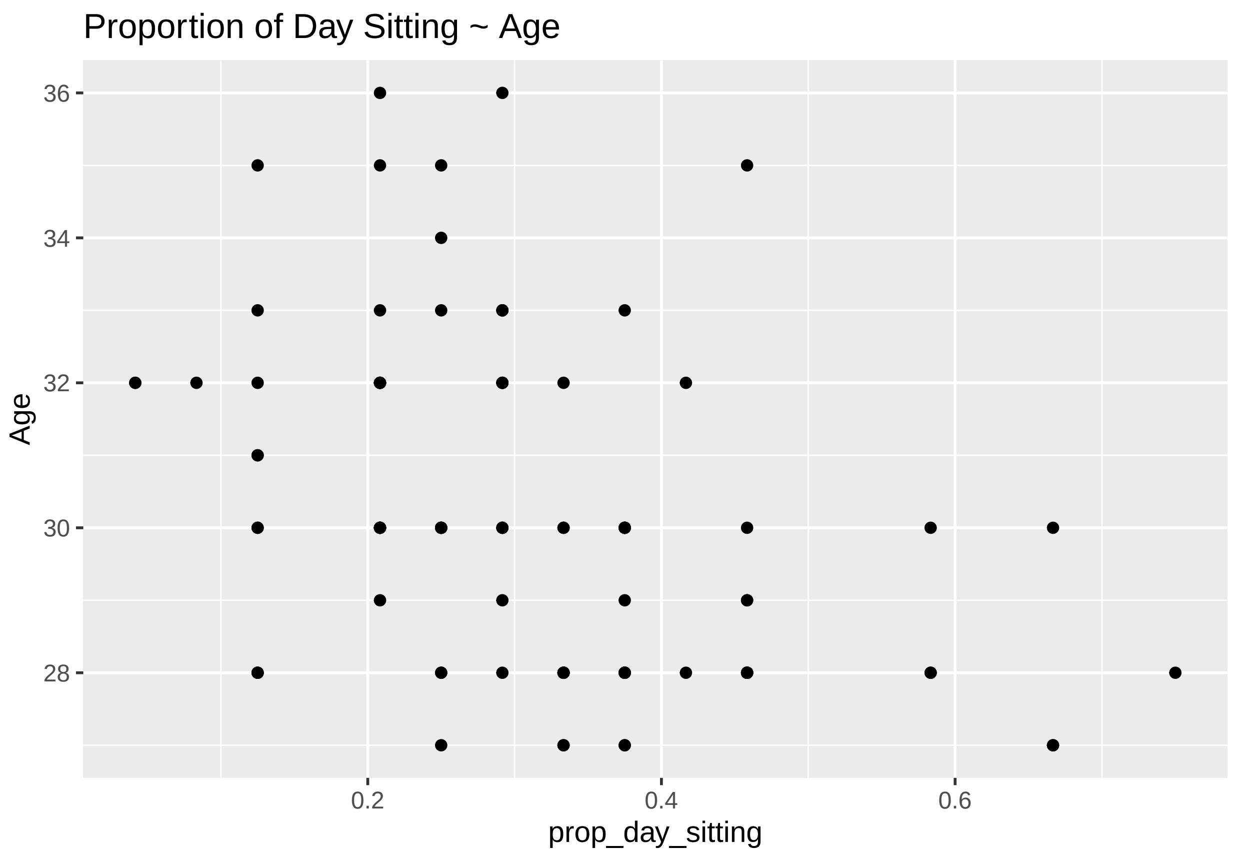
```
library(ggplot2)
ggplot(data = melted_corr_mat, aes(x=Var1, y=Var2,
                                   fill=value)) +
geom_tile() + geom_text(aes(Var2, Var1, label = value),
          color = "black", size = 4)
```



```
# No variables seem to have a very strong linear correlation based on the heat map apart from proportio
```

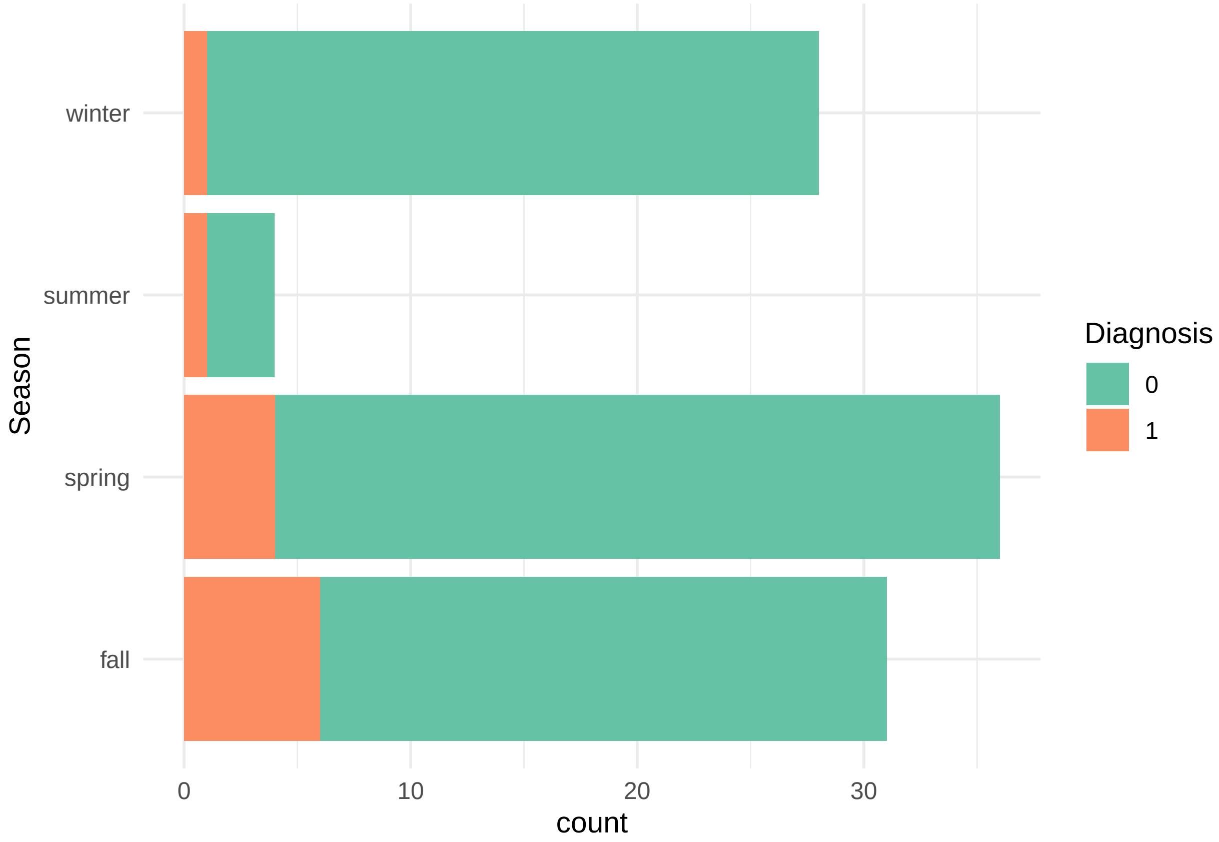
#Diagnosis is not very strongly correlated with many other variables, which I will further investigate through plots. Proportion of day sitting and age have the strongest correlation with each other out of the data. Trauma and lifestyle, as well as Diagnosis and lifestyle also have a mdoerate correlation compared to the other variables. This might be worth investigating further.

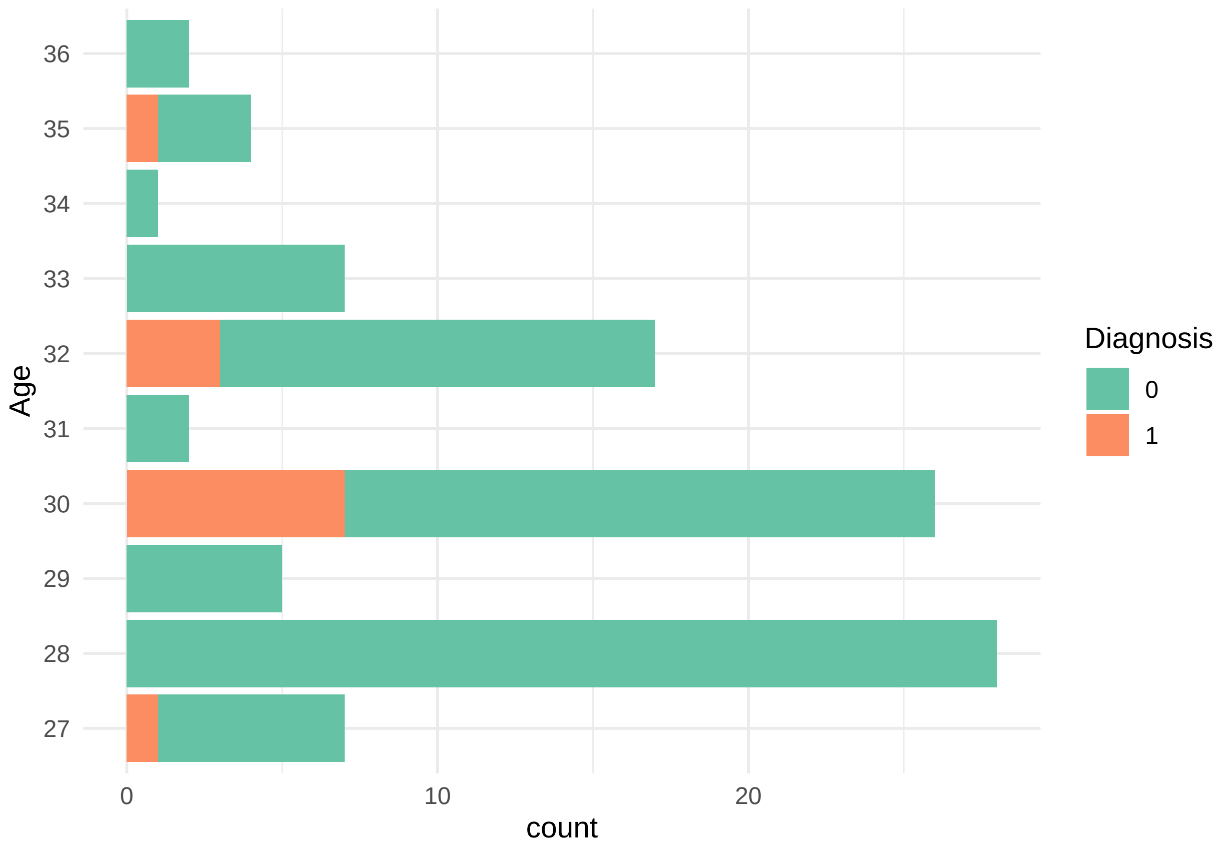
```
#Visualizing Proportion of day sitting versus age.
ggplot(fert4) + geom_point(aes(x = prop_day_sitting, y = Age)) + ggtitle("Proportion of Day Sitting ~ Ag
```

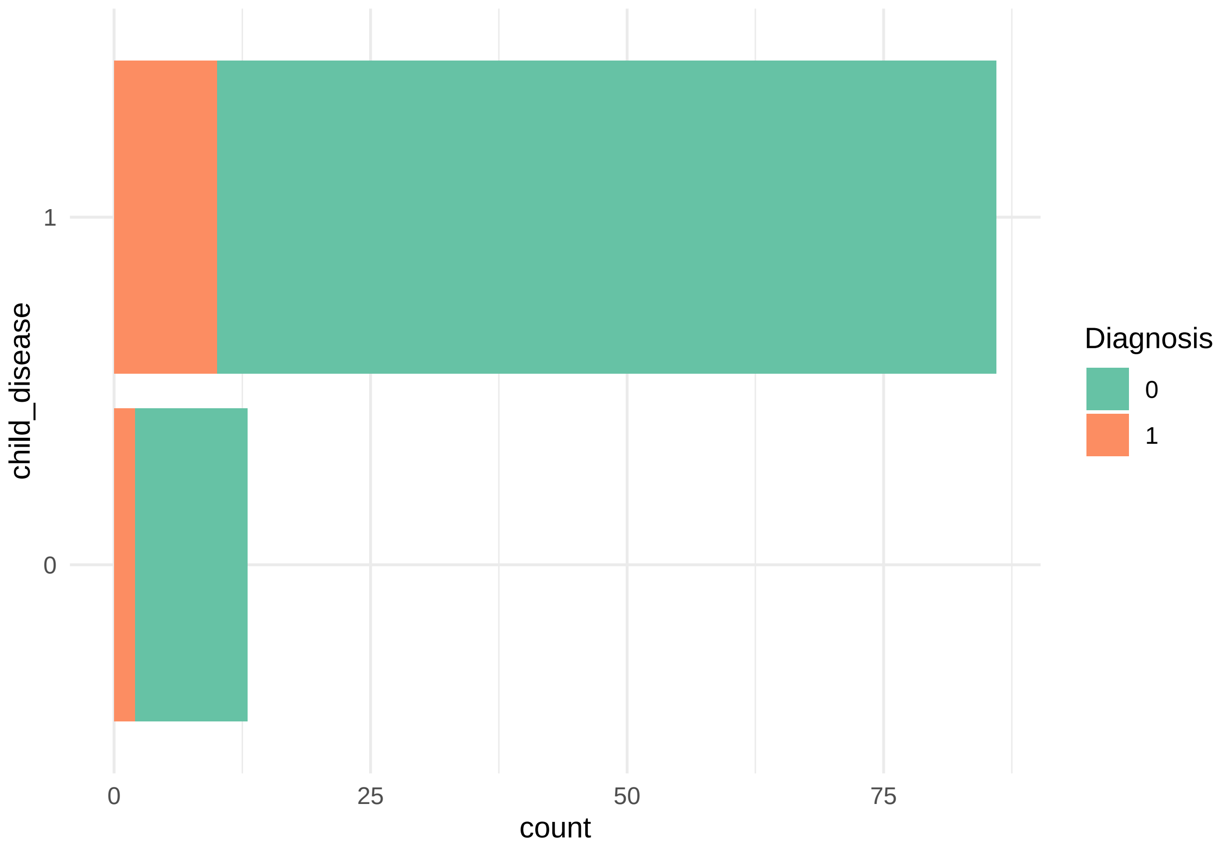


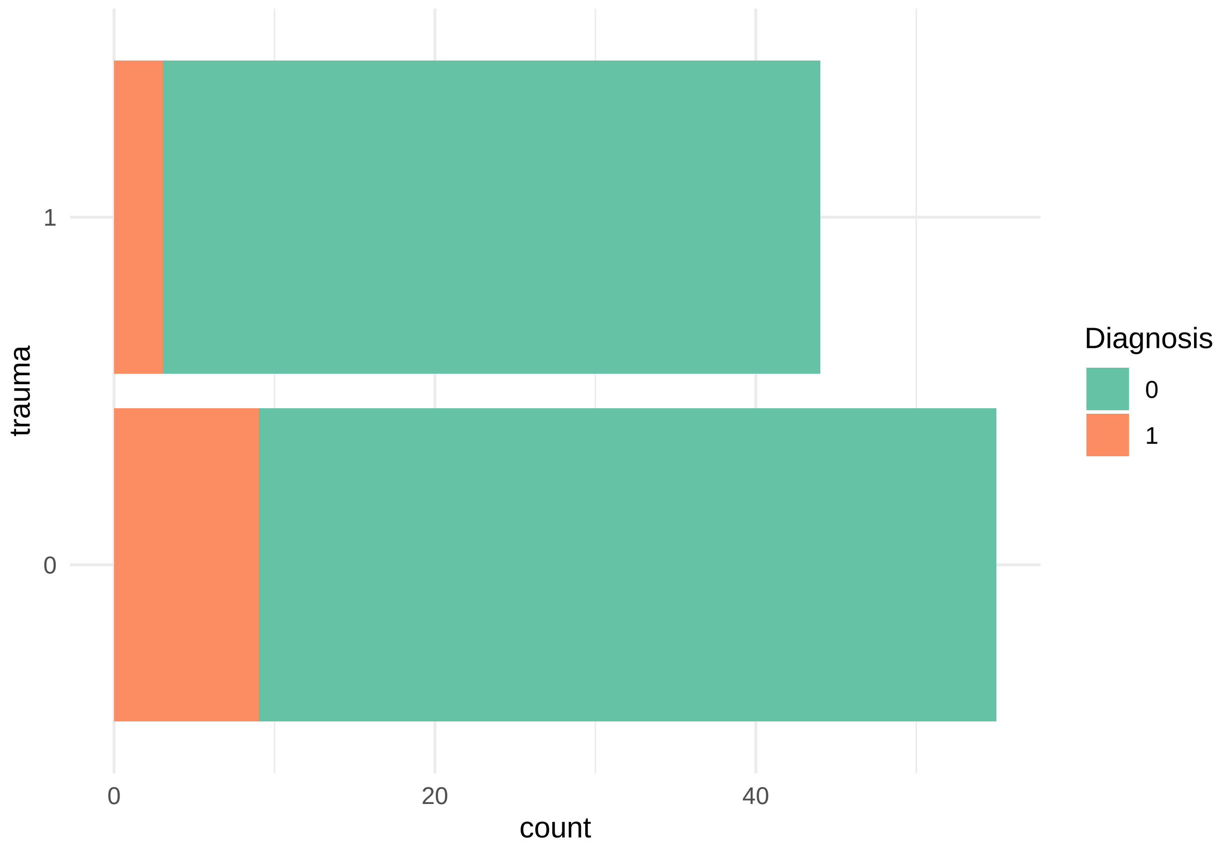
#However, apart from the higher values for proportion of day sitting, the distribution of proportion of day sitting with age seems to be very random. It does not seem to be correlated in any predictable way, so I will not be looking into this relationship further.

```
#Relationships between all variables and diagnosis.
fert4cols = c(colnames(fert4))
for (i in fert4cols){
  plt = ggplot(fert4,
         aes(x = factor(.data[[i]]),
             fill = factor(Diagnosis)))+
    geom_bar(position = "stack") +
    scale_fill_brewer(palette = "Set2") +
    labs(y = "count",
         fill = "Diagnosis",
         x = i)+
    theme_minimal() + coord_flip()
  print(plt)
}
```

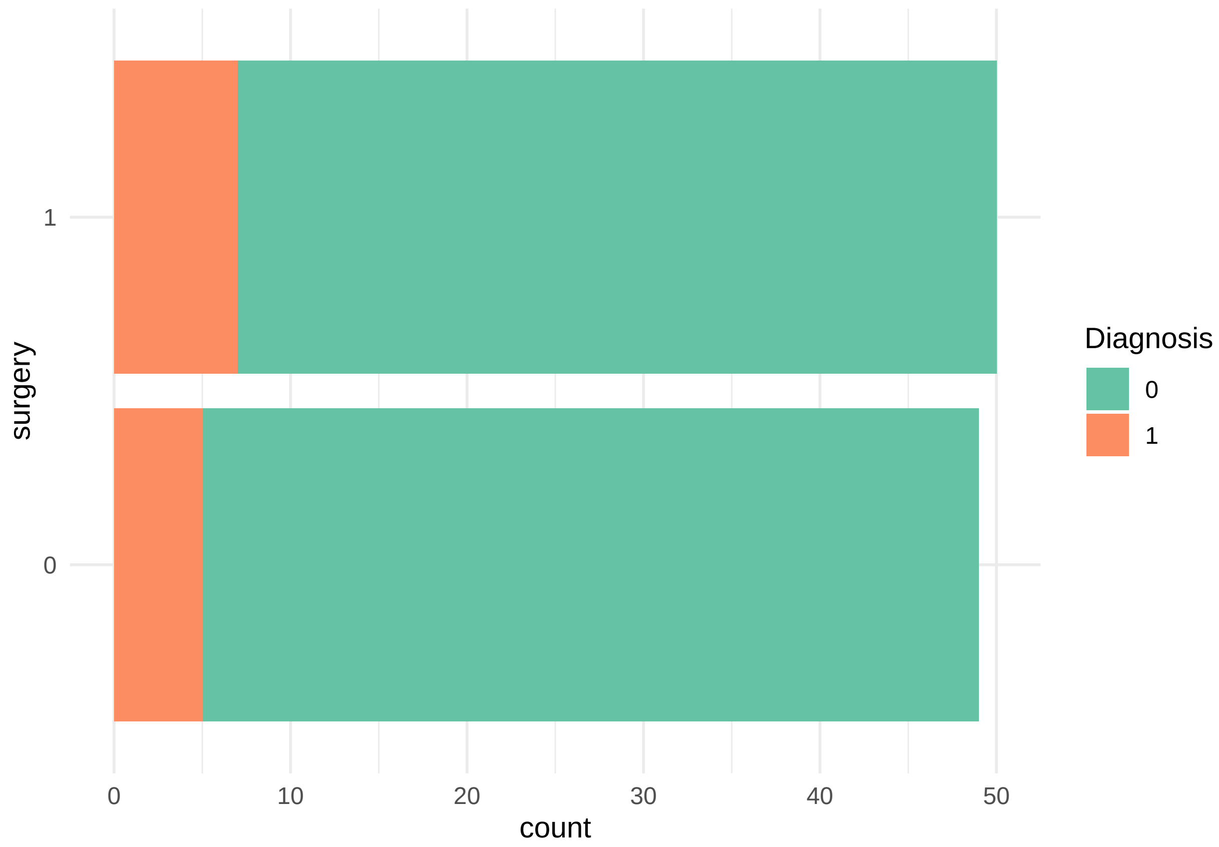

winter
summer
spring
fall
Season
0
10
20
30
count
Diagnosis
0
1


36
35
34
33
32
31
30
29
28
27
Age
0
10
20
count
Diagnosis
🟩 0
🟧 1

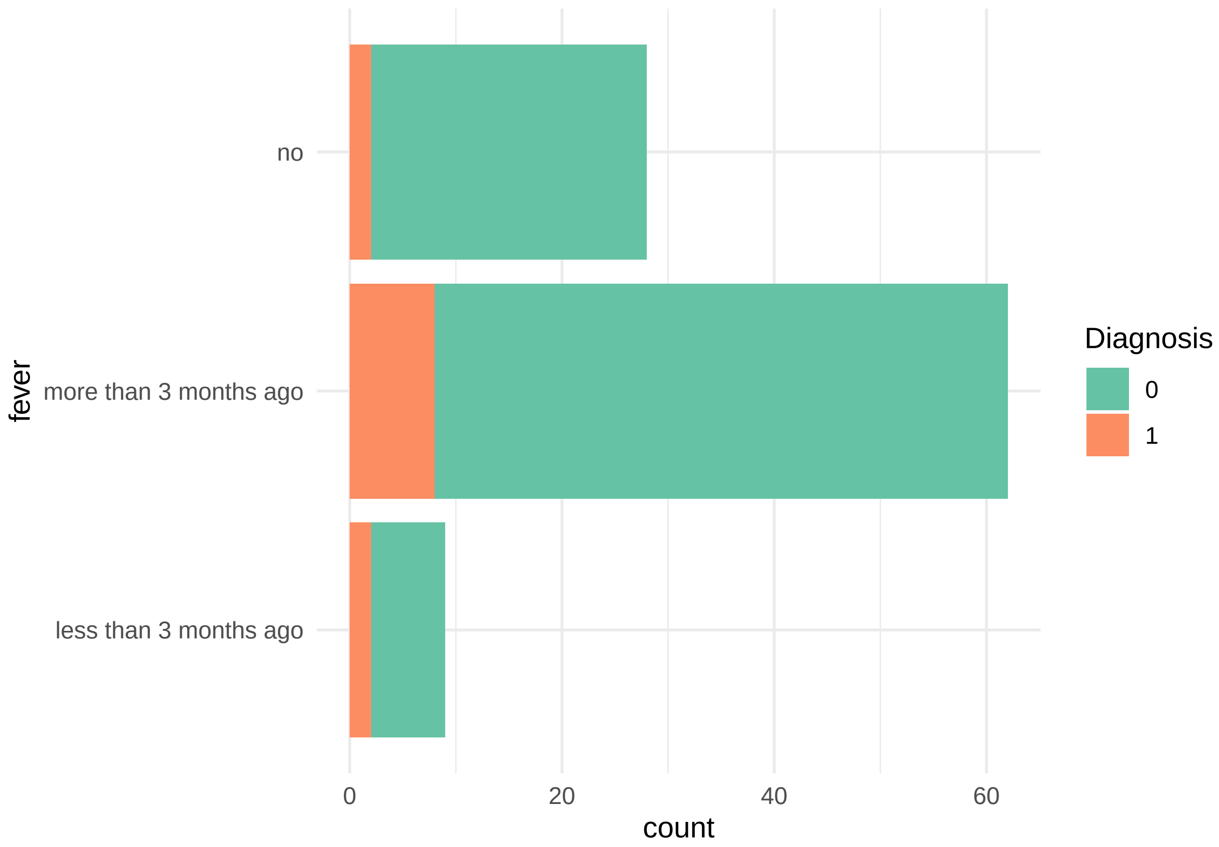

child_disease
1
0
count
0
25
50
75
Diagnosis
0
1

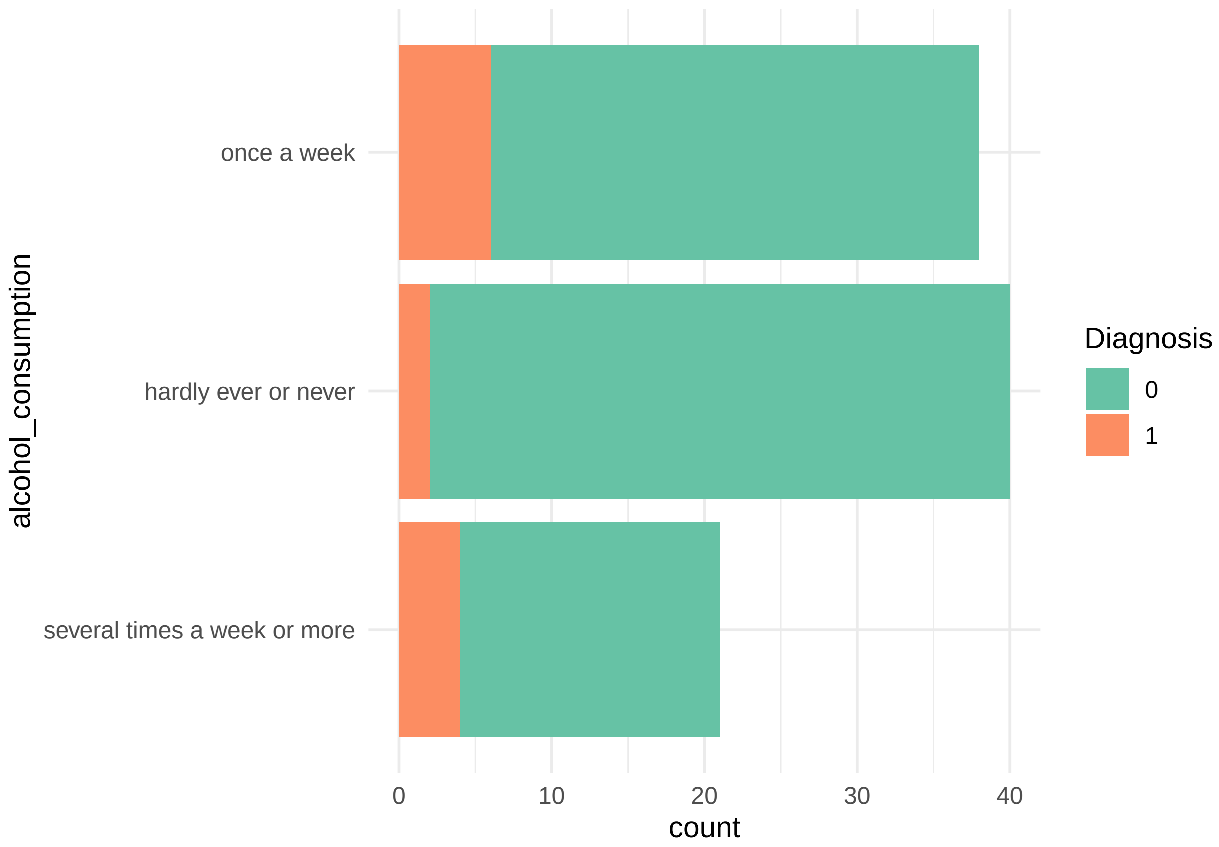


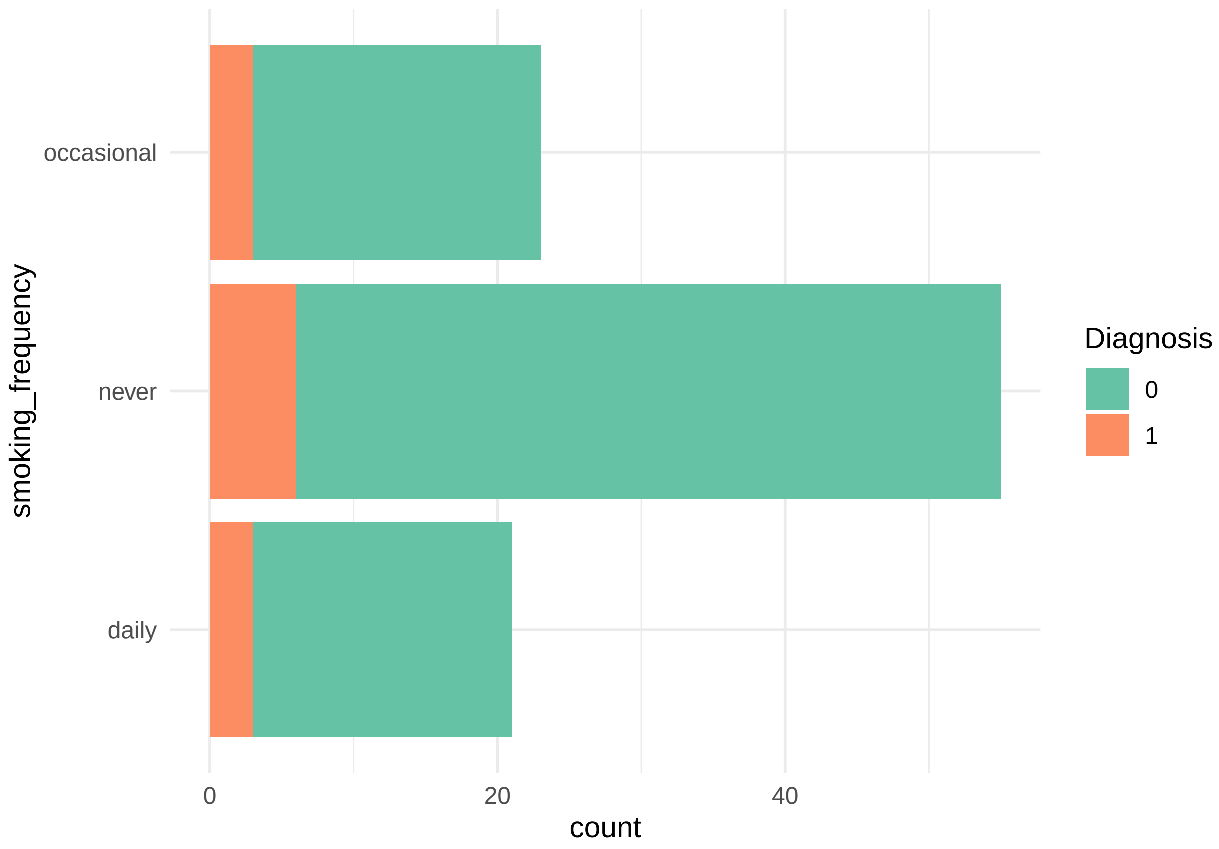
1
0
trauma
0
20
40
count
Diagnosis
🟩 0
🟥 1

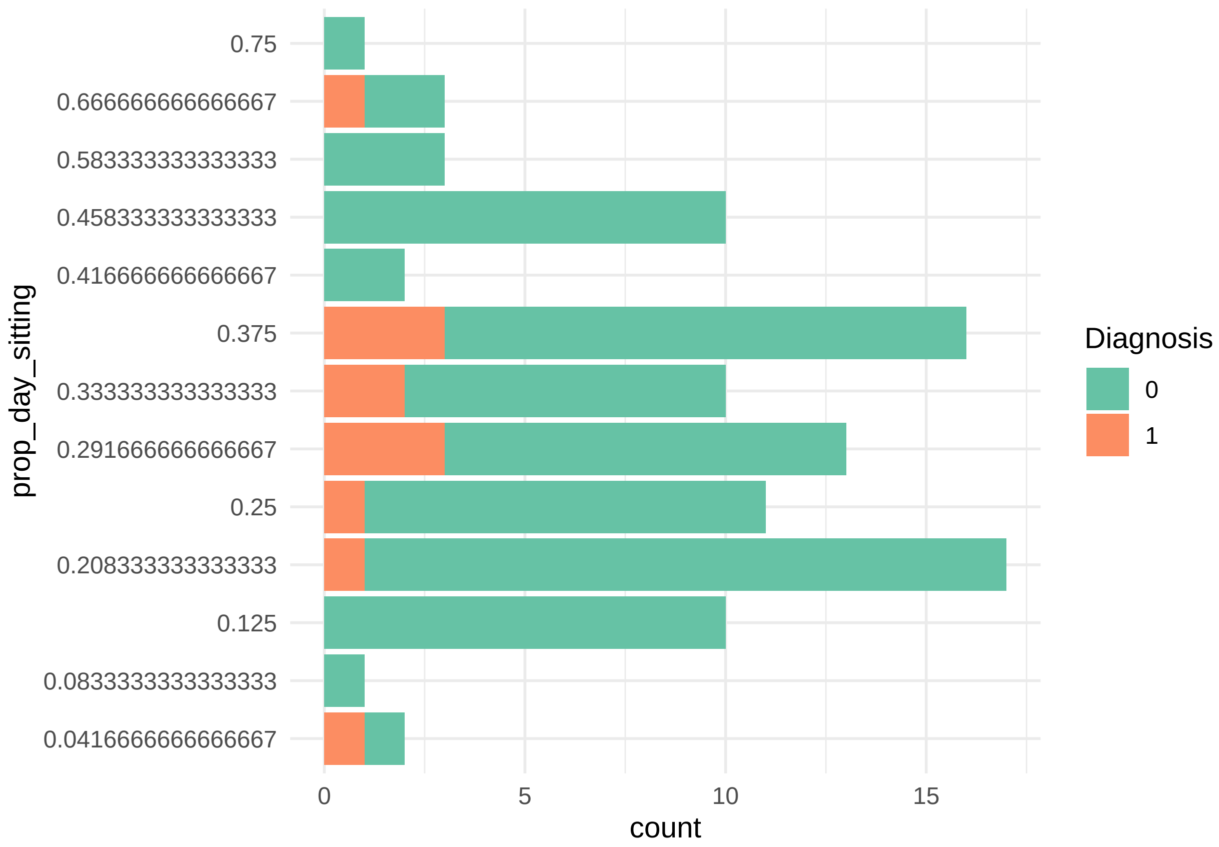


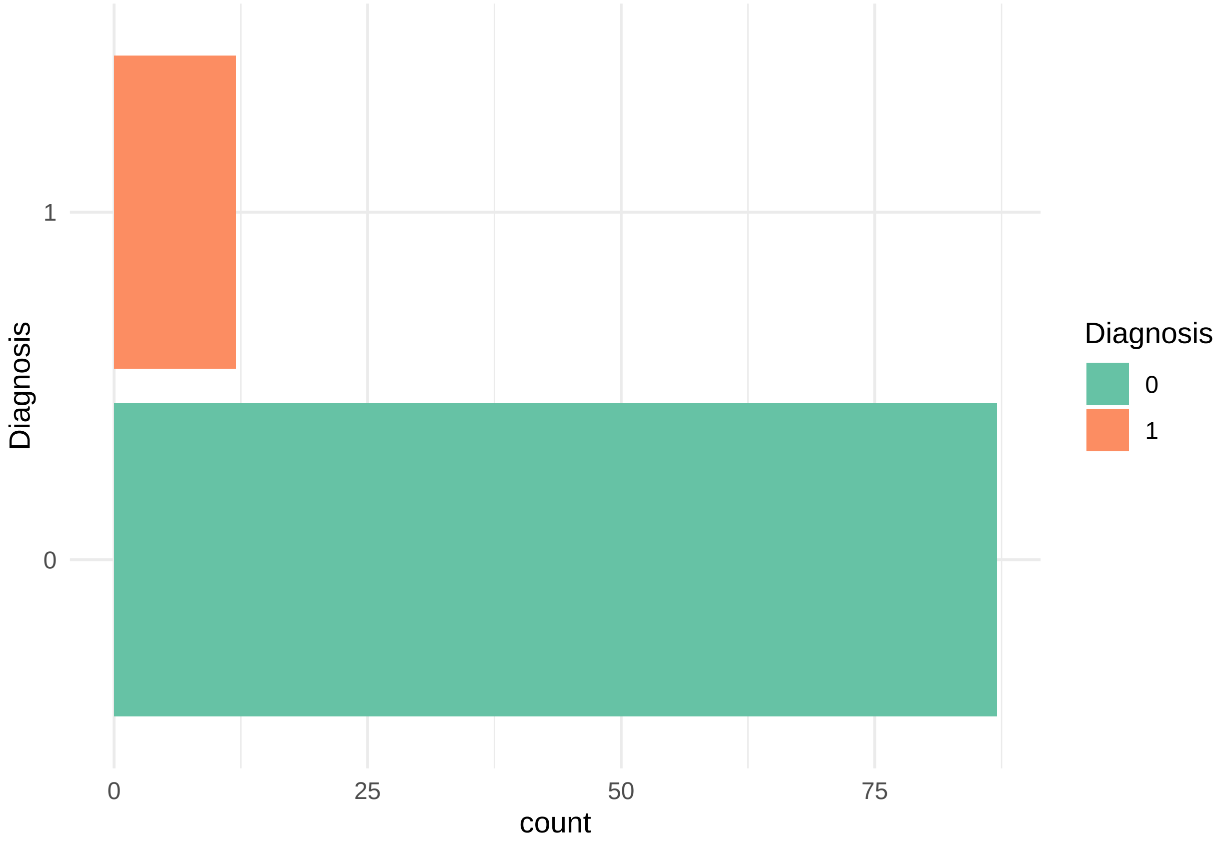
surgery
1
0
0
10
20
30
40
50
count
Diagnosis
0
1

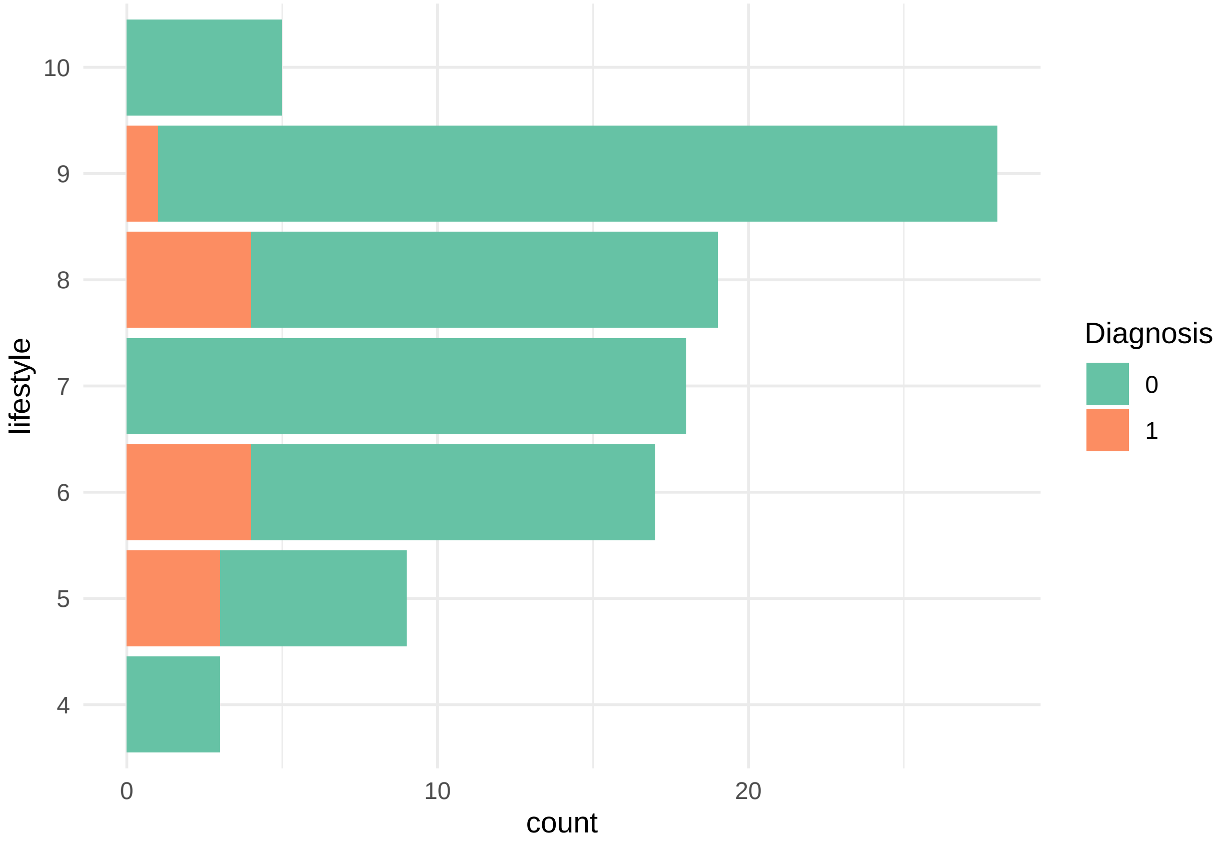

fever
no
more than 3 months ago
less than 3 months ago
0
20
40
60
count
Diagnosis
0
1


alcohol_consumption
once a week
hardly ever or never
several times a week or more
0
10
20
30
40
count
Diagnosis
0
1


occasional
never
daily
smoking_frequency
0
20
40
count
Diagnosis
0
1


0.75
0.666666666666667
0.583333333333333
0.458333333333333
0.416666666666667
0.375
0.333333333333333
0.291666666666667
0.25
0.208333333333333
0.125
0.0833333333333333
0.0416666666666667
prop_day_sitting
0
5
10
15
count
Diagnosis
0
1


1
0
Diagnosis
0
25
50
75
count
Diagnosis
🟩 0
🟥 1

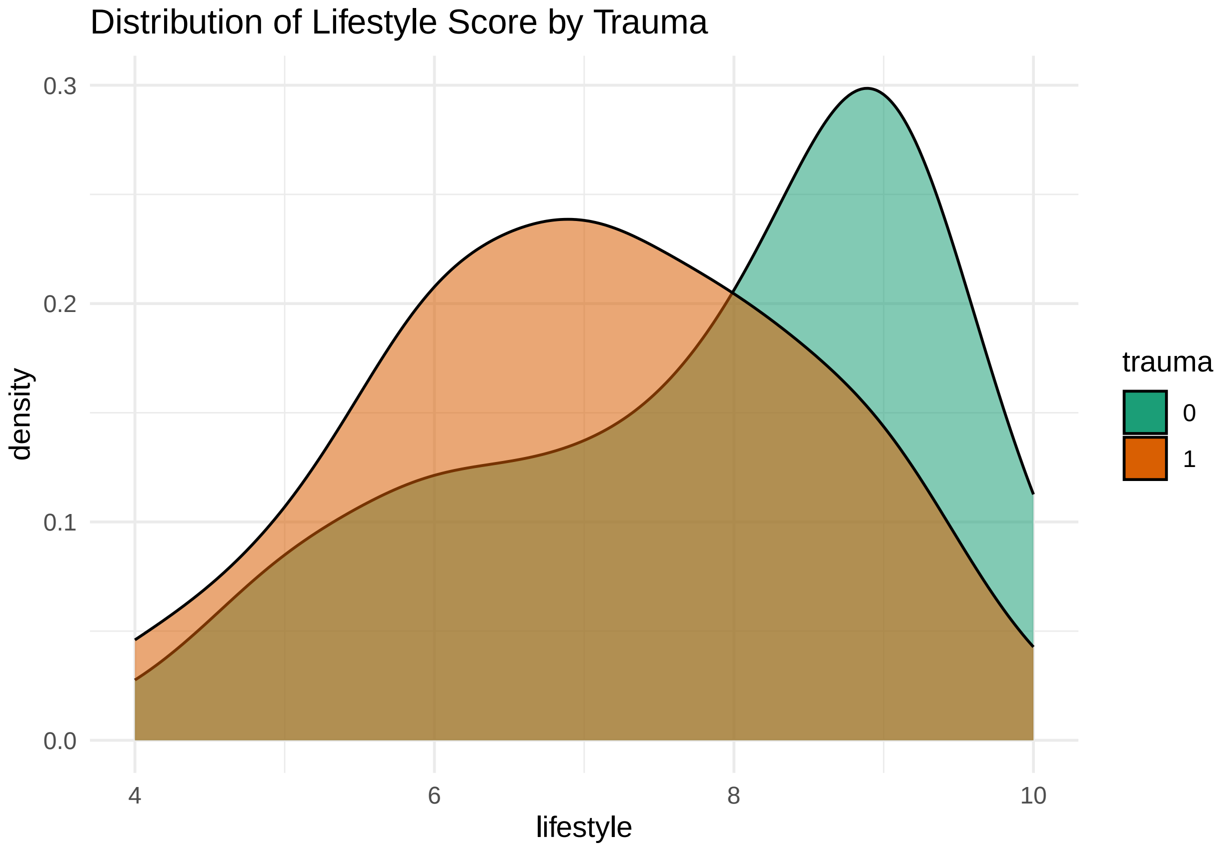


#Based on these plots, it seems as though most variables have very little correlation to diagnosis. However, some variables, such as age, season, lifestyle, trauma, and childish diseases may have a slight relationship with diagnosis.

#From the previous correlation plot, many variables seemed to have a relationship with trauma and lifestyle. I will further investigate these variables.

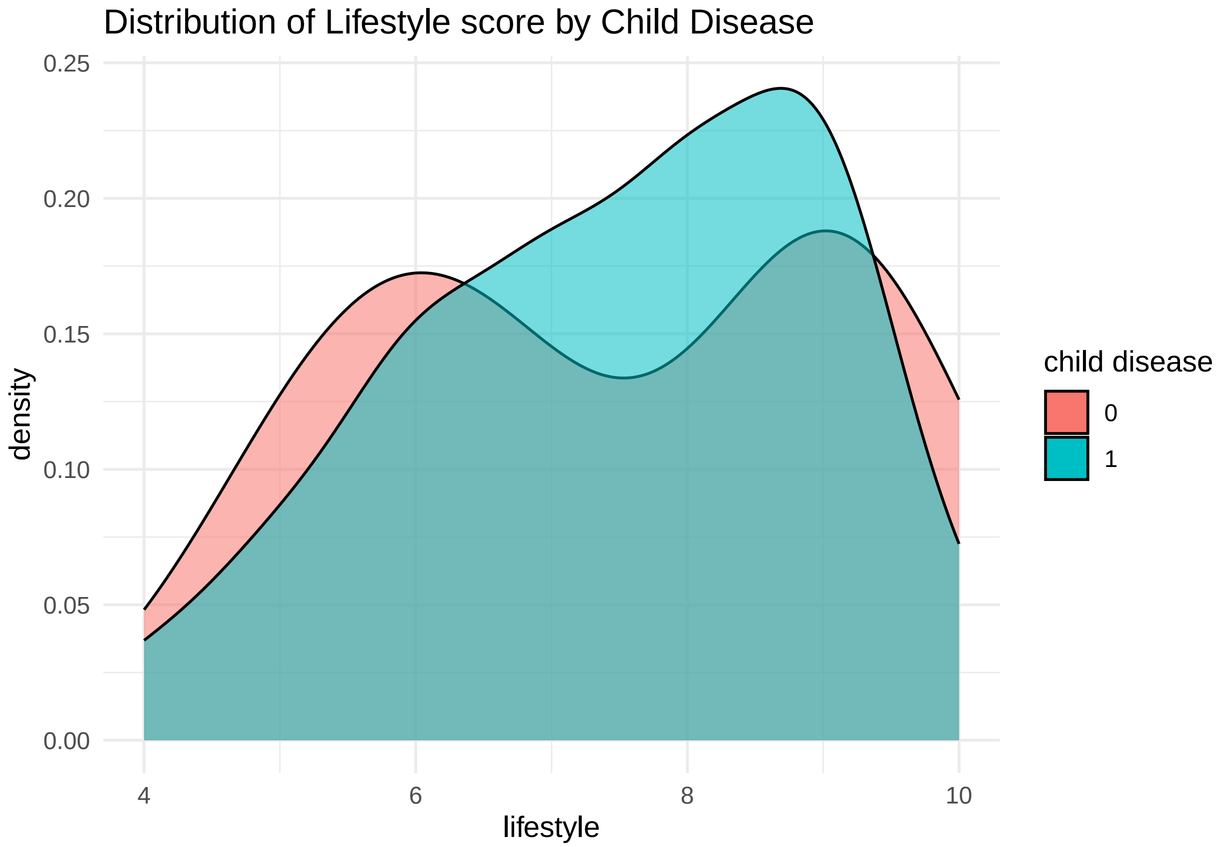
```
ggplot(fert4) + geom_density(aes(x=lifestyle, fill=factor(trauma), alpha = 0.4))  + scale_fill_brewer(pa
```

```
## Warning: The `<scale>` argument of `guides()` cannot be `FALSE`. Use "none" instead as
## of ggplot2 3.3.4.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



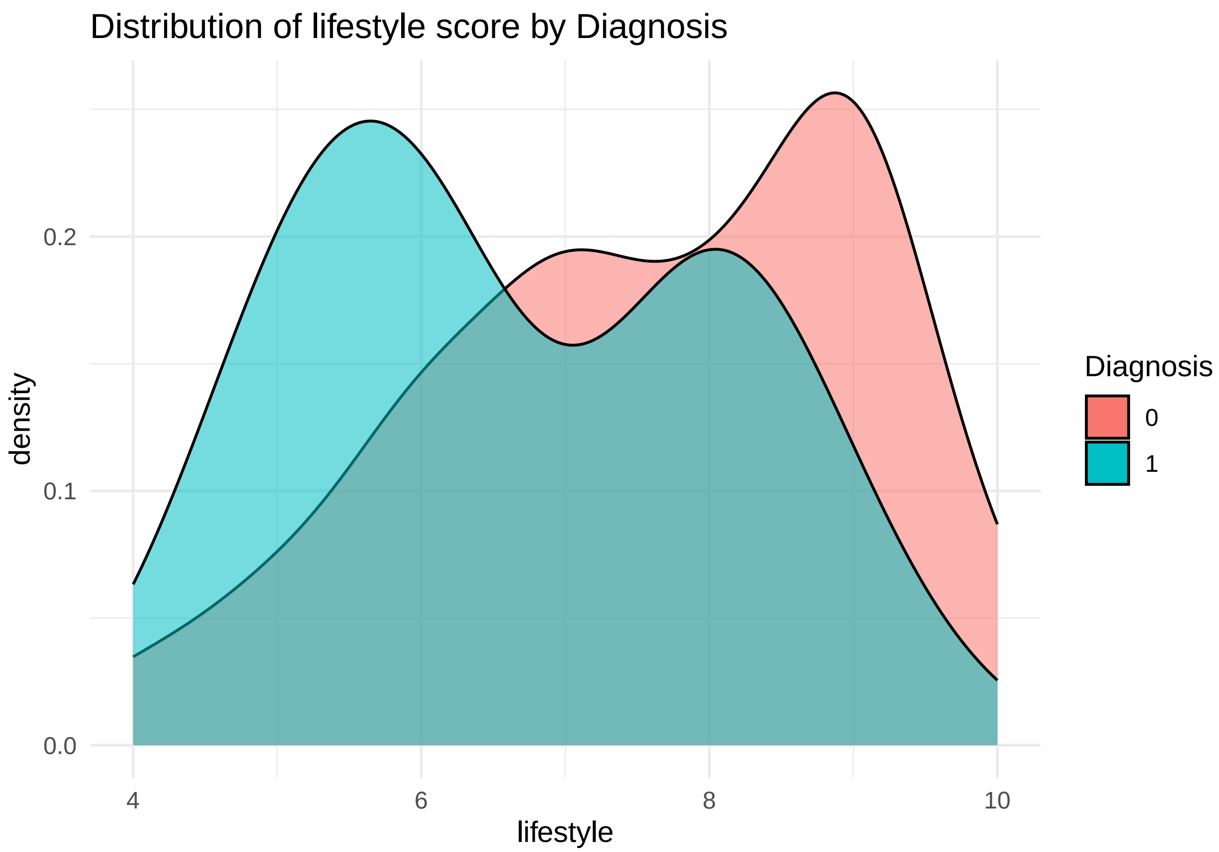
#Individuals without serious trauma tend to have better lifestyle scores. This begs the question: does having previous accidents and trauma cause ones lifestyle to become worse? Or does ones lifestyle being worse cause people to experience trauma? Either way this relationship could be useful.

```
ggplot(fert4) + geom_density(aes(x=lifestyle, fill=factor(child_disease), alpha = .3))  + scale_color_bi
```



#There are more indivduals who had a disease as a child that have better lifestyle scores. This makes sense, as those who had conditions as a child might be more inclined to lead healthier lifestyles later in life.

```
ggplot(fert4) + geom_density(aes(x=lifestyle, fill=factor(Diagnosis), alpha = 0.3)) + scale_color_brewe
```



#There are significantly more individuals who have lower lifestyle scores that also have altered semen.

#Based on these plots, I want to use the dataset to create a model that can predict lifestyle score.

```
fullmod = lm(lifestyle ~ Season + Age + child_disease + Diagnosis + surgery + trauma , data = fert4 )
summary(fullmod)
```

```
##
## Call:
## lm(formula = lifestyle ~ Season + Age + child_disease + Diagnosis +
##     surgery + trauma, data = fert4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9636 -0.9477  0.1584  1.1361  2.8190
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)    8.12154    2.13473   3.804 0.000259 ***
## Seasonspring  -0.27662    0.38186  -0.724 0.470705
## Seasonsummer   0.74699    0.81273   0.919 0.360493
## Seasonwinter  -0.17451    0.40350  -0.432 0.666418
## Age           -0.01548    0.07287  -0.212 0.832219
## child_disease  0.27561    0.46856   0.588 0.557875
## Diagnosis     -1.19701    0.47801  -2.504 0.014077 *
## surgery        0.41683    0.31990   1.303 0.195897
## trauma        -0.84542    0.33423  -2.529 0.013165 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.493 on 90 degrees of freedom
## Multiple R-squared:  0.1448, Adjusted R-squared:  0.06883
## F-statistic: 1.905 on 8 and 90 DF,  p-value: 0.06871
```

#From the summary output of the full model, it seems as though only Diagnosis and trauma have statistically significant impacts on lifestyle score. To test this, I am going to build up the model through stepwise selection.

```
#Finding best subsets of variables to predict lifestyle score.
all = regsubsets(lifestyle ~ Season + Age + child_disease + Diagnosis + surgery + trauma, data = fert4)
summary(all)
```

```
## Subset selection object
## Call: regsubsets.formula(lifestyle ~ Season + Age + child_disease +
##     Diagnosis + surgery + trauma, data = fert4)
## 8 Variables  (and intercept)
##               Forced in Forced out
## Seasonspring      FALSE      FALSE
## Seasonsummer      FALSE      FALSE
## Seasonwinter      FALSE      FALSE
## Age               FALSE      FALSE
## child_disease     FALSE      FALSE
## Diagnosis         FALSE      FALSE
## surgery           FALSE      FALSE
## trauma            FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          Seasonspring Seasonsummer Seasonwinter Age child_disease Diagnosis
## 1  ( 1 ) " "          " "          " "          " " " "           " "
## 2  ( 1 ) " "          " "          " "          " " " "           "*"
## 3  ( 1 ) " "          " "          " "          " " " "           "*"
## 4  ( 1 ) " "          "*"          " "          " " " "           "*"
## 5  ( 1 ) "*"          "*"          " "          " " " "           "*"
## 6  ( 1 ) "*"          "*"          " "          " " "*"           "*"
## 7  ( 1 ) "*"          "*"          "*"          " " "*"           "*"
## 8  ( 1 ) "*"          "*"          "*"          "*" "*"           "*"
##          surgery trauma
## 1  ( 1 ) " "     "*"
## 2  ( 1 ) " "     "*"
## 3  ( 1 ) "*"     "*"
## 4  ( 1 ) "*"     "*"
## 5  ( 1 ) "*"     "*"
## 6  ( 1 ) "*"     "*"
## 7  ( 1 ) "*"     "*"
## 8  ( 1 ) "*"     "*"
```

#The best subset of size two includes Diagnosis and trauma.

```
#Training the linear regression using backwards elimination, which works back from a full model and fin
set.seed(123)
train.control <- trainControl(method = "cv", number = 10)
```

```r
step.model <- train( lifestyle ~ Season + Age + child_disease + Diagnosis + surgery + trauma , data = fe
                    method = "leapBackward",
                    tuneGrid = data.frame(nvmax = 1:6),
                    trControl = train.control
                    )
step.model$results
```

```
##   nvmax     RMSE   Rsquared      MAE    RMSESD RsquaredSD     MAESD
## 1     1 1.523043 0.07102439 1.299944 0.2099110 0.09242585 0.1565423
## 2     2 1.454140 0.14976125 1.216145 0.2312315 0.14574768 0.1736640
## 3     3 1.496846 0.12221782 1.266159 0.2655524 0.14150919 0.2075383
## 4     4 1.481089 0.12390689 1.265645 0.2693159 0.14485707 0.2112807
## 5     5 1.491433 0.12152562 1.265024 0.2790923 0.12459635 0.2155624
## 6     6 1.496656 0.11496247 1.265076 0.2678444 0.11816920 0.1895125
```

```r
#Best model based on the previous parameters.
step.model$bestTune
```

```
##   nvmax
## 2     2
```

```r
summary(step.model$finalModel)
```

```
## Subset selection object
## 8 Variables  (and intercept)
##                Forced in Forced out
## Seasonspring      FALSE      FALSE
## Seasonsummer      FALSE      FALSE
## Seasonwinter      FALSE      FALSE
## Age               FALSE      FALSE
## child_disease     FALSE      FALSE
## Diagnosis         FALSE      FALSE
## surgery           FALSE      FALSE
## trauma            FALSE      FALSE
## 1 subsets of each size up to 2
## Selection Algorithm: backward
##          Seasonspring Seasonsummer Seasonwinter Age child_disease Diagnosis
## 1  ( 1 ) " "          " "          " "          " " " "           " "
## 2  ( 1 ) " "          " "          " "          " " " "           "*"
##          surgery trauma
## 1  ( 1 ) " "     "*"
## 2  ( 1 ) " "     "*"
```

```r
#In addition to the stepwise built model, I also conducted a K folds cross validation. Based on both of
mod3 = lm(lifestyle ~ trauma  + Diagnosis + I(trauma *surgery)  , data = fert4)
summary(mod3)
```

```
##
## Call:
## lm(formula = lifestyle ~ trauma + Diagnosis + I(trauma * surgery),
##     data = fert4)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9617 -0.9617  0.0383  1.0383  2.6000
##
## Coefficients:
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)           7.9617     0.2114  37.667  < 2e-16 ***
## trauma               -1.2267     0.3896  -3.149  0.00219 **
## Diagnosis            -0.9884     0.4617  -2.141  0.03486 *
## I(trauma * surgery)   0.6650     0.4515   1.473  0.14409
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.464 on 95 degrees of freedom
## Multiple R-squared:  0.1322, Adjusted R-squared:  0.1048
## F-statistic: 4.824 on 3 and 95 DF,  p-value: 0.003604
```

```
rmse = sqrt(mean(mod3$residuals^2))
rmse
```

```
## [1] 1.434109
```

```
#The interaction increased the r squared and also lowered the rmse. Thus, I think it is a good addition
```

```
#To check for multicollinearity in the model. None of the variables have an associated VIF score above
vif(mod3)
```

```
##              trauma           Diagnosis I(trauma * surgery)
##            1.731074            1.048924            1.777334
```

```
#Splitting the data into train and test sets.
n <- nrow(fert4)
n_train <- round(0.95  * n)
set.seed(123)
train_indices <- sample(1:n, n_train)
train <- fert4[train_indices, ]
test <- fert4[-train_indices, ]
test
```

```
## # A tibble: 5 x 11
##   Season   Age child_disease trauma surgery fever            alcohol_consumption
##   <fct>  <dbl>         <dbl>  <dbl>   <dbl> <fct>            <fct>
## 1 spring    30             0      1       1 more than 3 mon~ once a week
## 2 fall      32             1      1       0 no               several times a we~
## 3 fall      28             1      1       1 more than 3 mon~ hardly ever or nev~
## 4 spring    28             1      1       0 more than 3 mon~ once a week
## 5 winter    28             1      0       0 no               once a week
## # i 4 more variables: smoking_frequency <fct>, prop_day_sitting <dbl>,
## #   Diagnosis <dbl>, lifestyle <dbl>
```

```
paste("train sample size: ", dim(train)[1])
```

```
## [1] "train sample size:  94"
```

```
paste("test sample size: ", dim(test)[1])
```

```
## [1] "test sample size:  5"
```

```
#Due to the low number of observations I am going to train the model on 95% of the data to prioritize t
pred <- predict(mod3, test, type="response")

act_pred <- data.frame(observed = test$lifestyle, predicted =
                      round(pred))
act_pred
```

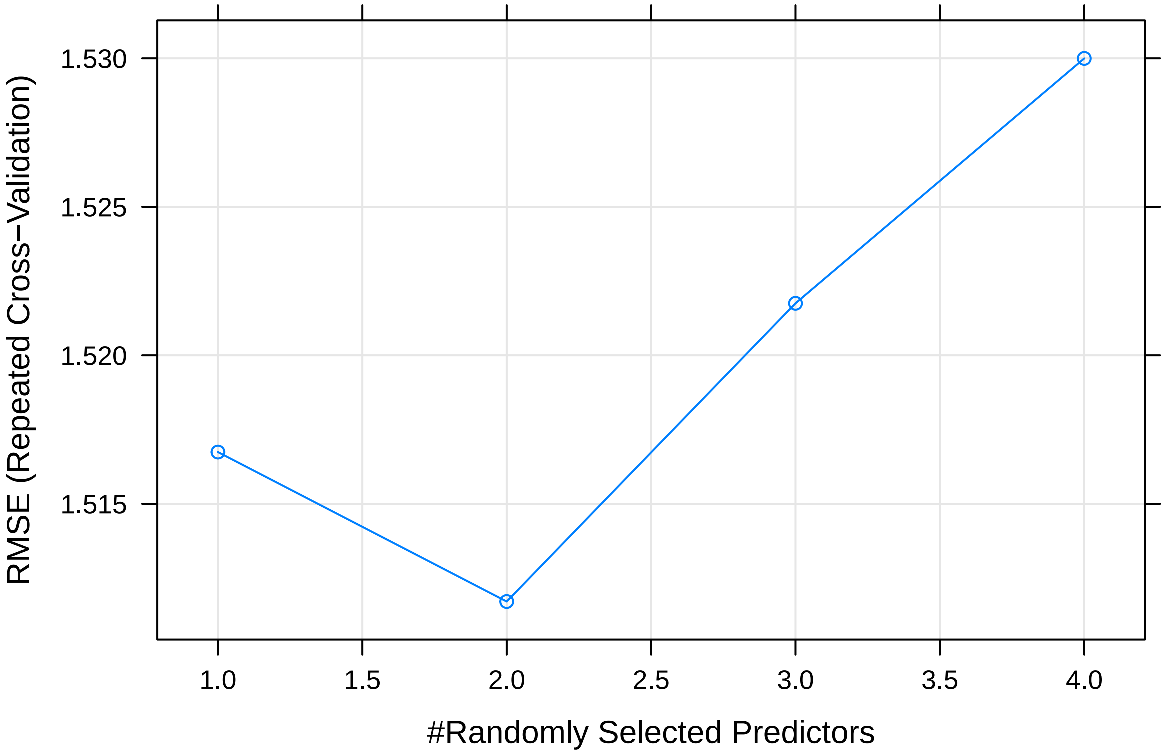
```
##   observed predicted
## 1        6         7
## 2        6         6
## 3        9         7
## 4        7         7
## 5        8         8
```

#The best linear model is about 60% accurate. This is decent with such a low number of observations and 10 unique lifestyle values. However, one of the two incorrect predictions is only 1 score off, which may be close enough.The total off for the predictions was 3.

```
#Starting to train a random forest model. I am finding the optimal mtry value.
control <- trainControl(method="repeatedcv", number=10, repeats=3, search="grid")
set.seed(7)
tunegrid <- expand.grid(.mtry=c(1:4))
rf_gridsearch <- train(lifestyle ~ Diagnosis + trauma + surgery + Season , data=train, method="rf", tun
print(rf_gridsearch)
```

```
## Random Forest
##
## 94 samples
##  4 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 84, 84, 83, 85, 84, 85, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE      Rsquared   MAE
##   1     1.516743  0.1877401  1.309620
##   2     1.511710  0.1698332  1.299869
##   3     1.521751  0.1509558  1.299269
##   4     1.529998  0.1457734  1.296966
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 2.
```

```
plot(rf_gridsearch)
```



#Adding surgery and season also do not have an impact on the random forest model.

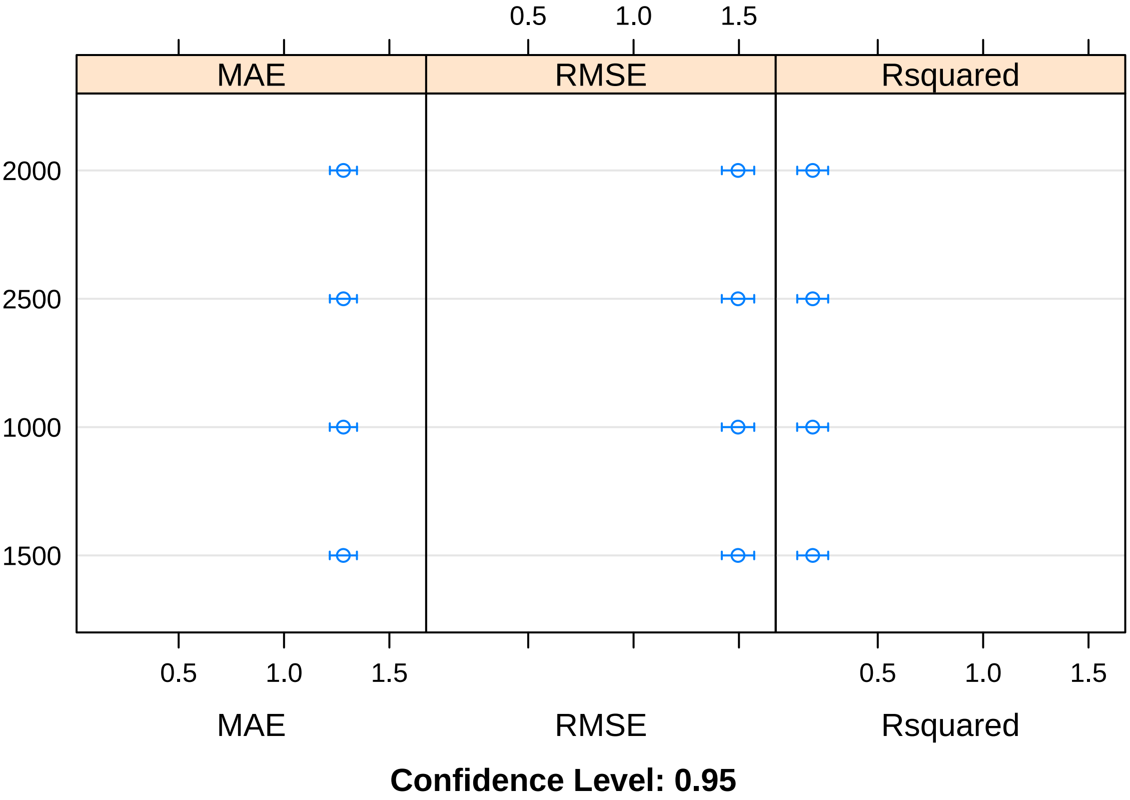
```
#The best mtry value, or number of variables to try at each split is 2.
tunegrid1 <- expand.grid(.mtry=c(2))

rf_gridsearch <- train(lifestyle ~ Diagnosis + trauma, data=train, method="rf", tuneGrid=tunegrid1, trC
rf
```

```
## function (n, df1, df2, ncp)
## {
##     if (missing(ncp))
##         .Call(C_rf, n, df1, df2)
##     else (rchisq(n, df1, ncp = ncp)/df1)/(rchisq(n, df2)/df2)
## }
## <bytecode: 0x0000000030f9a1e0>
## <environment: namespace:stats>
```

```
control <- trainControl(method="repeatedcv", number=10, repeats=3, search="grid")
tunegrid <- expand.grid(.mtry=c(1))
modellist <- list()
for (ntree in c(1000, 1500, 2000, 2500)) {
 set.seed(7)
 fit <- train(lifestyle ~ Diagnosis + trauma , data=train, method="rf", tuneGrid=tunegrid, trControl=co
 key <- toString(ntree)
 modellist[[key]] <- fit
}
# compare results
results <- resamples(modellist)
summary(results)
```

```
##
## Call:
## summary.resamples(object = results)
##
## Models: 1000, 1500, 2000, 2500
## Number of resamples: 30
##
## MAE
##           Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA’s
## 1000 0.9578177 1.198593 1.279849 1.281808 1.386814 1.622264    0
## 1500 0.9561001 1.199662 1.281226 1.281429 1.387481 1.617776    0
## 2000 0.9569566 1.199618 1.283878 1.281876 1.385484 1.618585    0
## 2500 0.9570264 1.201019 1.284574 1.281819 1.385232 1.619321    0
##
## RMSE
##          Min.  1st Qu.   Median     Mean  3rd Qu.     Max. NA’s
## 1000 1.015536 1.390305 1.529799 1.495552 1.627963 1.936217    0
## 1500 1.013935 1.387594 1.529266 1.495459 1.627395 1.935798    0
## 2000 1.013932 1.387807 1.529404 1.495607 1.626521 1.934924    0
## 2500 1.013831 1.389729 1.529149 1.495456 1.625803 1.935245    0
##
## Rsquared
##             Min.    1st Qu.    Median      Mean   3rd Qu.      Max. NA’s
## 1000 0.001050230 0.02759170 0.1587090 0.1910324 0.3020058 0.6648212    0
## 1500 0.001087132 0.02761043 0.1550575 0.1913153 0.3017886 0.6648102    0
## 2000 0.001093504 0.02746843 0.1581429 0.1911636 0.3009730 0.6648123    0
## 2500 0.001061682 0.02677284 0.1592527 0.1912417 0.3011854 0.6648068    0
```

```
dotplot(results)
```



#MAE, RMSE, and R-Squared based on different n-tree values. They do not seem to vary much.

```
control <- trainControl(method="repeatedcv", number=10, repeats=3, search="grid")
tunegrid <- expand.grid(.mtry=c(2))
finalrf <- train(lifestyle ~ Diagnosis + trauma + surgery + Season, data=train, method="rf", tuneGrid=tu
```

```
pred1 <- predict(finalrf, test, type="raw")

act_pred <- data.frame(observed = test$lifestyle, predicted =
                       round(pred1))
act_pred
```

```
##   observed predicted
## 1        6         7
## 2        6         7
## 3        9         7
## 4        7         7
## 5        8         8
```

#The best tuned random forest model is only 40% accurate on the test data, so the linear model is more accurate. The total score off for this model was 4. The linear model seems to predict better.

#Next, I want to build a model to predict if an individual has had trauma or not. I am going to train the model on 90% of the data since there are more responses for each unique value of trauma. This way I can test the model on more data.

```r
n <- nrow(fert4)
n_train1 <- round(0.90  * n)
set.seed(123)
train_indices1 <- sample(1:n, n_train1)
train1 <- fert4[train_indices1, ]
test1 <- fert4[-train_indices1, ]
test
```

```
## # A tibble: 5 x 11
##   Season   Age child_disease trauma surgery fever            alcohol_consumption
##   <fct>  <dbl>         <dbl>  <dbl>   <dbl> <fct>            <fct>
## 1 spring    30             0      1       1 more than 3 mon~ once a week
## 2 fall      32             1      1       0 no               several times a we~
## 3 fall      28             1      1       1 more than 3 mon~ hardly ever or nev~
## 4 spring    28             1      1       0 more than 3 mon~ once a week
## 5 winter    28             1      0       0 no               once a week
## # i 4 more variables: smoking_frequency <fct>, prop_day_sitting <dbl>,
## #   Diagnosis <dbl>, lifestyle <dbl>
```

```r
paste("train sample size: ", dim(train1)[1])
```

```
## [1] "train sample size:  89"
```

```r
paste("test sample size: ", dim(test1)[1])
```

```
## [1] "test sample size:  10"
```

```r
logit1 <- glm(trauma ~ Diagnosis  + lifestyle + Season + surgery + child_disease, data = train, family =
summary(logit1)
```

```
##
## Call:
## glm(formula = trauma ~ Diagnosis + lifestyle + Season + surgery +
##     child_disease, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7555  -0.8664  -0.4835   0.9592   1.8918
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.6227     1.5204   0.410   0.6821
## Diagnosis       -1.8016     0.8930  -2.018   0.0436 *
## lifestyle       -0.3811     0.1635  -2.331   0.0198 *
## Seasonspring     1.2474     0.5982   2.085   0.0370 *
## Seasonsummer     0.2695     1.2872   0.209   0.8342
## Seasonwinter     0.1040     0.6296   0.165   0.8688
## surgery          0.7238     0.4855   1.491   0.1360
## child_disease    1.3737     0.8484   1.619   0.1054
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```

```
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 128.22  on 93  degrees of freedom
## Residual deviance: 105.32  on 86  degrees of freedom
## AIC: 121.32
##
## Number of Fisher Scoring iterations: 4
```

#Based on the summary output for the binomial logistic regression, it seems as though only lifestyle, diagnosis, and the spring season seem to have a statistically significant impact on trauma. Therefore, I am going to fit a second logistic model using only these values.

```
logit2<- glm(trauma ~ Diagnosis  + lifestyle + Season , data = train, family = "binomial")
summary(logit2)
```

```
##
## Call:
## glm(formula = trauma ~ Diagnosis + lifestyle + Season, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5384  -0.9180  -0.5808   1.1360   1.7775
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)    2.0286     1.2602   1.610   0.1074
## Diagnosis     -1.7945     0.8851  -2.027   0.0426 *
## lifestyle     -0.3723     0.1590  -2.342   0.0192 *
## Seasonspring   1.3952     0.5791   2.409   0.0160 *
## Seasonsummer   0.2571     1.2783   0.201   0.8406
## Seasonwinter   0.3035     0.6046   0.502   0.6157
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 128.22  on 93  degrees of freedom
## Residual deviance: 110.30  on 88  degrees of freedom
## AIC: 122.3
##
## Number of Fisher Scoring iterations: 4
```

#I am curious if using the combined lifestyle score makes a difference in the models accuracy compared to using each part of lifestyle score: alcohol consumption, smoking frequency, etc.

```
#Model with the components of lifestyle score.
logit3 = glm(trauma ~ Diagnosis  + smoking_frequency+ alcohol_consumption + prop_day_sitting + fever+
```

#To test this I am going to build another model using these variables, and perform a drop-in-deviance test to determine if the model is better with more terms.

```
drop_in_dev <- anova(logit2, logit3, test = "Chisq")
drop_in_dev
```

```
## Analysis of Deviance Table
##
## Model 1: trauma ~ Diagnosis + lifestyle + Season
## Model 2: trauma ~ Diagnosis + smoking_frequency + alcohol_consumption +
##     prop_day_sitting + fever + Season
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1        88     110.30
## 2        82     107.07  6   3.2251   0.7801
```

#Due to the large p-value we accept the null hypothesis, which means that there is not a significant difference between the full and nested models. Thus, lifestyle is acceptable to use.

#I am going to use the bestglm function to test for the best combinations of variables.

```
fert4i = data.frame(fert4[, c(1,2,3,5,10,11,4)])
fert4i
```

```
##    Season Age child_disease surgery Diagnosis lifestyle trauma
## 1  spring  30             0       1         0         6      1
## 2  spring  35             1       1         1         6      0
## 3  spring  27             1       0         0         9      0
## 4  spring  32             0       1         0         9      1
## 5  spring  30             1       0         1         8      1
## 6  spring  30             1       1         0         7      0
## 7  spring  30             0       0         0         7      0
## 8  spring  36             1       1         0         6      1
## 9    fall  30             0       1         0         9      0
## 10   fall  29             1       0         0        10      0
## 11   fall  30             1       0         0         6      1
## 12   fall  32             1       1         0         6      1
## 13   fall  32             1       1         0         7      1
## 14   fall  33             1       0         0         9      0
## 15   fall  35             1       1         0         7      1
## 16   fall  33             1       0         0         7      1
## 17   fall  30             1       1         0         9      0
## 18   fall  30             1       1         1         9      0
## 19   fall  32             1       1         0         8      1
## 20   fall  30             1       0         1         6      0
## 21   fall  30             0       1         0         9      0
## 22   fall  32             1       0         0         6      0
## 23   fall  30             1       0         0         9      1
## 24   fall  30             1       1         1         8      0
## 25   fall  28             1       1         0         9      0
## 26   fall  30             1       0         0        10      0
## 27   fall  30             1       1         1         6      0
## 28   fall  32             1       0         1         6      1
## 29   fall  28             0       1         0        10      0
## 30   fall  30             0       1         1         5      0
## 31   fall  29             1       1         0         9      0
## 32   fall  28             1       0         0         9      0
## 33   fall  30             0       0         0         9      0
## 34   fall  28             1       1         0         7      1
## 35   fall  28             1       1         0         9      1
## 36 winter  32             1       0         0         6      1
## 37 winter  32             1       1         0        10      0
## 38 winter  28             1       1         0         9      0
## 39 winter  30             0       1         1         5      0
## 40 winter  30             1       0         0         9      0
## 41 winter  28             1       1         0         6      1
## 42 winter  28             1       0         0         6      1
## 43 winter  28             1       1         0         5      0
## 44 winter  28             1       0         0         9      0
## 45 winter  28             1       0         0         8      1
## 46 winter  28             1       0         0         8      0
## 47 spring  28             1       0         0         9      0
## 48 spring  31             1       0         0         5      1
## 49 spring  30             1       1         0         8      1
## 50 spring  32             1       1         0         7      1
## 51 spring  28             1       0         0         8      1
## 52 spring  28             1       0         0         7      1
## 53 spring  28             1       1         0         7      1
## 54 spring  29             1       1         0         9      0
## 55 spring  28             1       1         0         7      0
## 56 spring  28             1       0         0         6      1
## 57 spring  30             1       1         0         7      1
## 58 spring  28             1       0         0         4      1
## 59   fall  28             0       0         0         6      0
## 60   fall  28             0       0         0         6      0
## 61 winter  30             1       0         0         8      0
## 62 winter  29             1       1         0         6      1
## 63 winter  28             1       0         0         9      0
## 64 winter  28             1       0         0         8      0
## 65 spring  28             0       1         0         9      0
## 66 spring  27             1       0         0         5      1
## 67 spring  27             1       0         0         9      0
## 68 spring  27             1       0         0         7      0
## 69 spring  27             1       1         0         7      0
## 70 spring  27             1       0         1         5      1
## 71 summer  30             1       0         0         9      0
## 72   fall  28             1       0         0         5      0
## 73 winter  27             1       0         0         8      0
## 74 winter  28             1       0         0         8      0
## 75 winter  32             1       1         0         8      0
## 76 winter  32             1       1         0         5      0
## 77 winter  31             1       1         0         9      1
## 78 winter  28             1       0         0         8      1
## 79 winter  36             1       1         0         6      0
## 80 spring  35             1       0         0         9      1
## 81 winter  33             1       1         0         8      1
## 82 spring  35             1       0         0         7      0
## 83 spring  33             1       1         0        10      1
## 84 spring  32             1       0         1         8      0
## 85 spring  34             1       0         0         4      1
## 86 spring  32             1       1         0         5      1
## 87 spring  32             1       1         0         7      1
## 88 spring  33             1       1         0         9      1
## 89 spring  33             1       1         0         7      1
## 90 spring  33             1       1         0         8      1
## 91 summer  32             1       0         0         8      0
## 92 summer  32             1       0         0         8      1
## 93 summer  32             1       1         1         8      0
## 94   fall  28             1       0         0         4      0
## 95 winter  30             1       0         0         9      0
## 96 winter  29             1       0         0         7      0
## 97 winter  30             1       1         0         9      1
## 98 winter  30             1       1         0         9      0
## 99 winter  30             0       1         0         7      1
```

```
best = bestglm(fert4i, family = binomial, IC = "AIC", method = "exhaustive")
```

```
## Morgan-Tatar search since family is non-gaussian.
```

```
## Note: factors present with more than 2 levels.
```

```
best$BestModels
```

```
##   Season  Age child_disease surgery Diagnosis lifestyle Criterion
## 1   TRUE TRUE          TRUE   FALSE      TRUE      TRUE  125.0506
## 2   TRUE TRUE         FALSE   FALSE      TRUE      TRUE  125.1486
## 3  FALSE TRUE          TRUE   FALSE      TRUE      TRUE  125.5182
## 4  FALSE TRUE         FALSE   FALSE      TRUE      TRUE  125.6153
## 5   TRUE TRUE          TRUE    TRUE      TRUE      TRUE  126.2325
```

```
logit3.5 = glm(trauma ~ Diagnosis  + lifestyle + Age + Season , data = train, family = "binomial")
summary(logit3.5)
```

```
##
## Call:
## glm(formula = trauma ~ Diagnosis + lifestyle + Age + Season,
##     family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1122  -0.8722  -0.5083   0.9445   1.8707
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.81396    3.46824  -1.676   0.0937 .
## Diagnosis    -1.96820    0.92734  -2.122   0.0338 *
## lifestyle    -0.38825    0.16552  -2.346   0.0190 *
## Age           0.26386    0.11063   2.385   0.0171 *
## Seasonspring  1.41349    0.60262   2.346   0.0190 *
## Seasonsummer -0.06477    1.30425  -0.050   0.9604
## Seasonwinter  0.41771    0.62856   0.665   0.5063
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 128.22  on 93  degrees of freedom
## Residual deviance: 104.00  on 87  degrees of freedom
## AIC: 118
##
## Number of Fisher Scoring iterations: 4
```

```
#Final logistic model. Removing age improves model. Can see through summary, increases significance of

logit4 =  glm(trauma ~ Diagnosis  + lifestyle + Age  , data = train, family = "binomial")
summary(logit4)
```

```
##
## Call:
## glm(formula = trauma ~ Diagnosis + lifestyle + Age, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0684  -0.8696  -0.6267   1.0065   2.1875
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.0387     3.2908  -1.531  0.12573
## Diagnosis    -1.9233     0.8751  -2.198  0.02796 *
## lifestyle    -0.4040     0.1558  -2.593  0.00952 **
## Age           0.2632     0.1063   2.476  0.01330 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 128.22  on 93  degrees of freedom
## Residual deviance: 110.76  on 90  degrees of freedom
## AIC: 118.76
##
## Number of Fisher Scoring iterations: 4
```

```
#Interaction terms of Diagnosis and lifestyle, child_disease and diagnosis, and age and lifestyle, do n
```

#Since a logistic model outputs the log odds for each variable, I will present this instead as average marginal effect.

```
effects_logit_dia = margins(logit4)
summary(effects_logit_dia)
```

```
##     factor     AME     SE       z      p   lower   upper
##        Age  0.0531 0.0188  2.8278 0.0047  0.0163  0.0900
##  Diagnosis -0.3882 0.1608 -2.4142 0.0158 -0.7035 -0.0730
##  lifestyle -0.0816 0.0271 -3.0097 0.0026 -0.1347 -0.0284
```

#We can interpret this result as an increase in each respective unit for each indepedent variables increases the chance of having trauma by the AME. For intstance, for every unit change in lifestyle there is a -7% chance of a change in trauma. We can also see that in our logistic model, Diagnosis and the spring season have the largest impact on presence of trauma.

```
pred1 <- predict(logit4, test1, type="response")


act_pred <- data.frame(observed = test1$trauma, predicted =
                      round(pred1))
act_pred
```

```
##    observed predicted
## 1         1         1
## 2         0         0
## 3         1         1
## 4         1         0
## 5         1         0
## 6         0         0
## 7         1         0
## 8         0         0
## 9         1         0
## 10        0         0
```

#The best logistic model is 60% accurate.

```
pred1. = round(pred1)
pred1.factor = as.factor(pred1.)

confusionMatrix(factor(test1$trauma), pred1.factor,  mode = "everything")
```

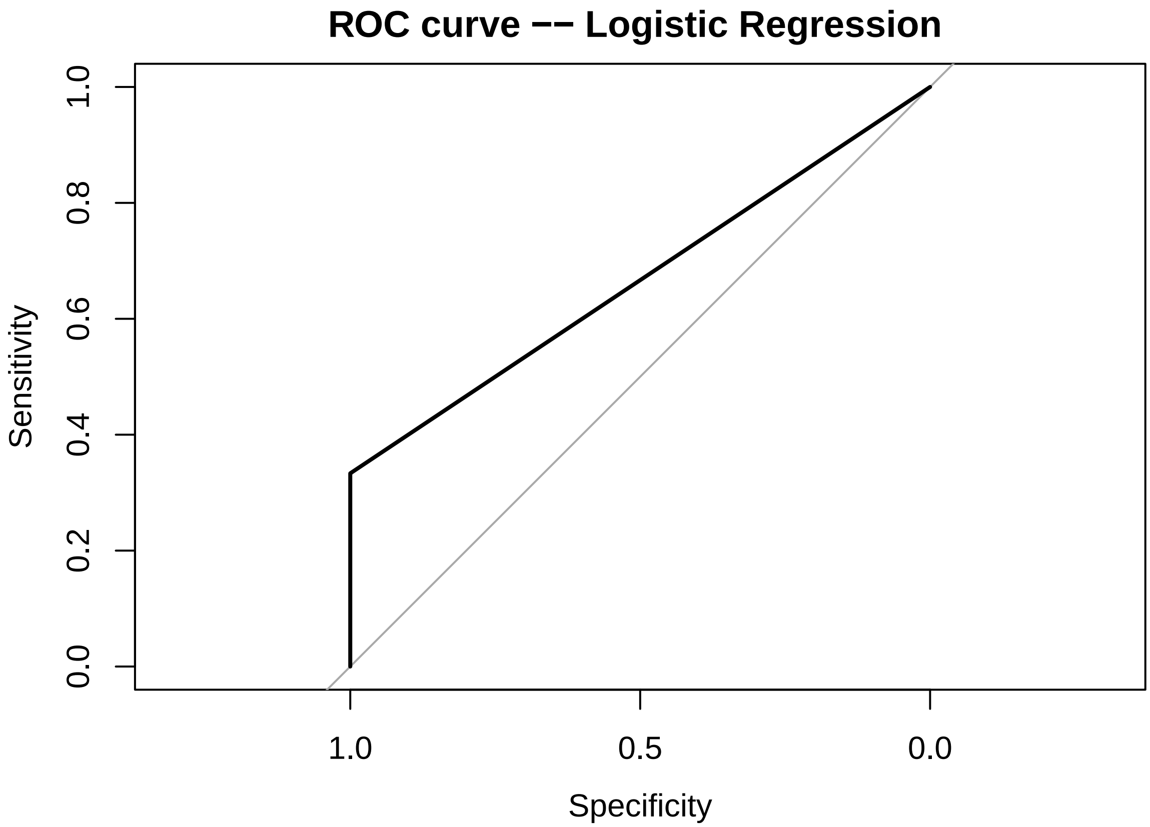
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 4 0
##          1 4 2
##
##                Accuracy : 0.6
##                  95% CI : (0.2624, 0.8784)
##     No Information Rate : 0.8
##     P-Value [Acc > NIR] : 0.9672
##
##                   Kappa : 0.2857
##
##  Mcnemar’s Test P-Value : 0.1336
##
##             Sensitivity : 0.5000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 0.3333
##               Precision : 1.0000
##                  Recall : 0.5000
##                       F1 : 0.6667
##               Prevalence : 0.8000
##           Detection Rate : 0.4000
##     Detection Prevalence : 0.4000
##        Balanced Accuracy : 0.7500
##
##         'Positive' Class : 0
##
```

```
roc_score=roc(test1$trauma, pred1.) #AUC score
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(roc_score ,main ="ROC curve -- Logistic Regression ")
```



#From the confusion matrix, we can see that the model predicted all negative cases correctly. However, positive predictions were not great.

#To model trauma, I am also going to utilize the K-Nearest-Neighbors algorithm for classification.

```
trainControl <- trainControl(method="repeatedcv", number=10, repeats=3)
metric = 'Accuracy'
```

```
fit.knn <- train(factor(trauma) ~ Diagnosis  + lifestyle + Season , data=train1, method="knn" , metric
knn.k1 <- fit.knn$bestTune
print(fit.knn)
```

```
## k-Nearest Neighbors
##
## 89 samples
##  3 predictor
##  2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 81, 80, 80, 80, 80, 80, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.6144444  0.1955550
##   7  0.5982407  0.1419722
##   9  0.5978704  0.1530006
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

```
test_pred <- predict(fit.knn, newdata = test1)
test_pred
```

```
##  [1] 1 1 1 0 0 0 1 1 1 0
## Levels: 0 1
```

```
confusionMatrix(factor(test1$trauma),test_pred, mode = "everything")
```

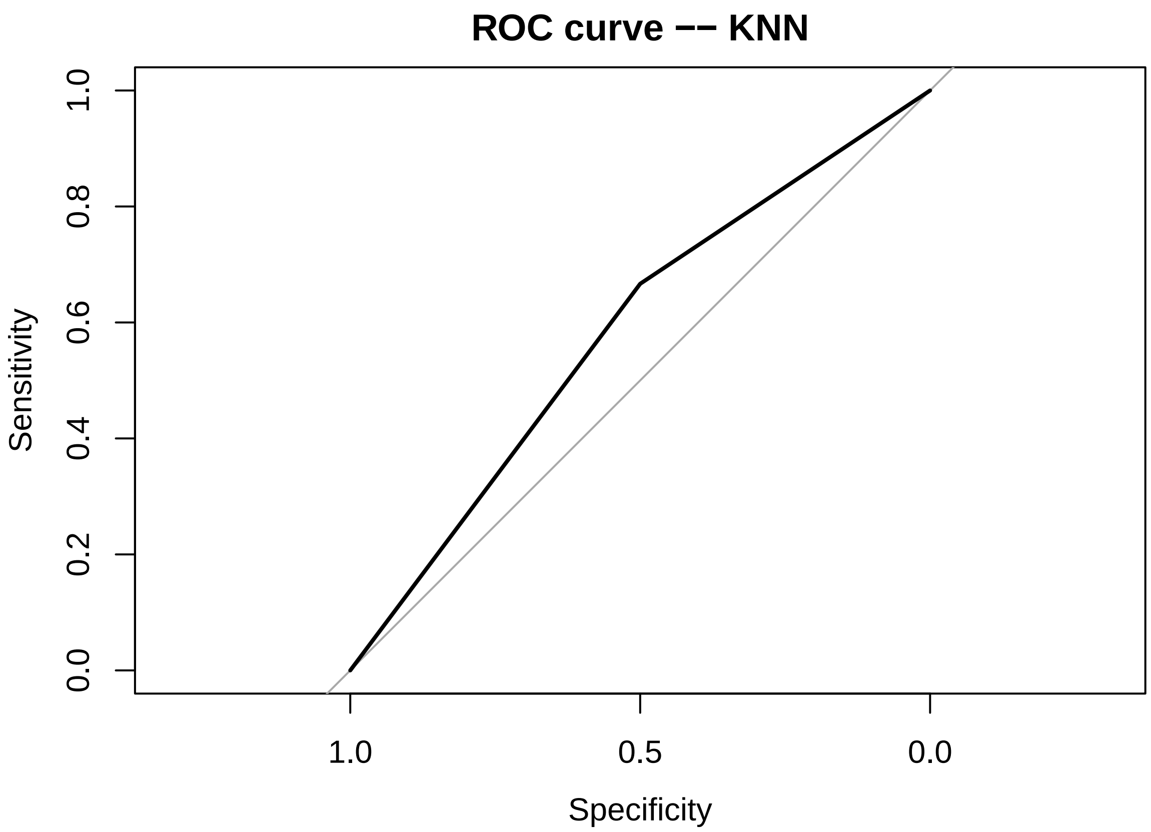
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 2 2
##          1 2 4
##
##                Accuracy : 0.6
##                  95% CI : (0.2624, 0.8784)
##     No Information Rate : 0.6
##     P-Value [Acc > NIR] : 0.6331
##
##                   Kappa : 0.1667
##
##  Mcnemar's Test P-Value : 1.0000
##
##             Sensitivity : 0.5000
##             Specificity : 0.6667
##          Pos Pred Value : 0.5000
##          Neg Pred Value : 0.6667
##                Precision : 0.5000
##                   Recall : 0.5000
##                       F1 : 0.5000
##               Prevalence : 0.4000
##           Detection Rate : 0.2000
##     Detection Prevalence : 0.4000
##        Balanced Accuracy : 0.5833
##
##         'Positive' Class : 0
##
```

```
roc_score=roc(test1$trauma, as.numeric(test_pred)) #AUC score
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(roc_score ,main ="ROC curve -- KNN")
```



```
pred1knn <- predict(fit.knn, test1, type="raw")


act_pred <- data.frame(observed = test1$trauma, predicted =
                         pred1knn)
```

```
act_pred
```

```
##    observed predicted
## 1         1         1
## 2         0         1
## 3         1         1
## 4         1         0
## 5         1         0
## 6         0         0
## 7         1         0
## 8         0         1
## 9         1         1
## 10        0         0
```

#Using a knn approach, I achieved 70% accuracy in predicting presence of trauma in the test set, as compared to 60% accuracy through the logistic model. We can see this in the previous table. However, the negative prediction rate was not as great as the logistic model, but the positive prediction rate was better.

#Lastly, I am going to train another random forest model.

```
control1 <- trainControl(method="repeatedcv", number=10, repeats=3, search="grid")
set.seed(7)
rf1 <- train(factor(trauma) ~ Diagnosis  + lifestyle + Season + child_disease + surgery, data=train, me
rf1
```

```
## Random Forest
##
## 94 samples
##  5 predictor
##  2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 84, 84, 84, 85, 84, 85, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.6648148  0.3077165
##   4     0.6959259  0.3724376
##   7     0.6503704  0.2802822
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 4.
```

```
test_predrf <- predict(rf1, newdata = test1)
test_predrf
```

```
##  [1] 1 0 0 0 0 0 0 1 1 0
## Levels: 0 1
```

```
confusionMatrix(factor(test1$trauma), test_predrf,  mode = "everything")
```

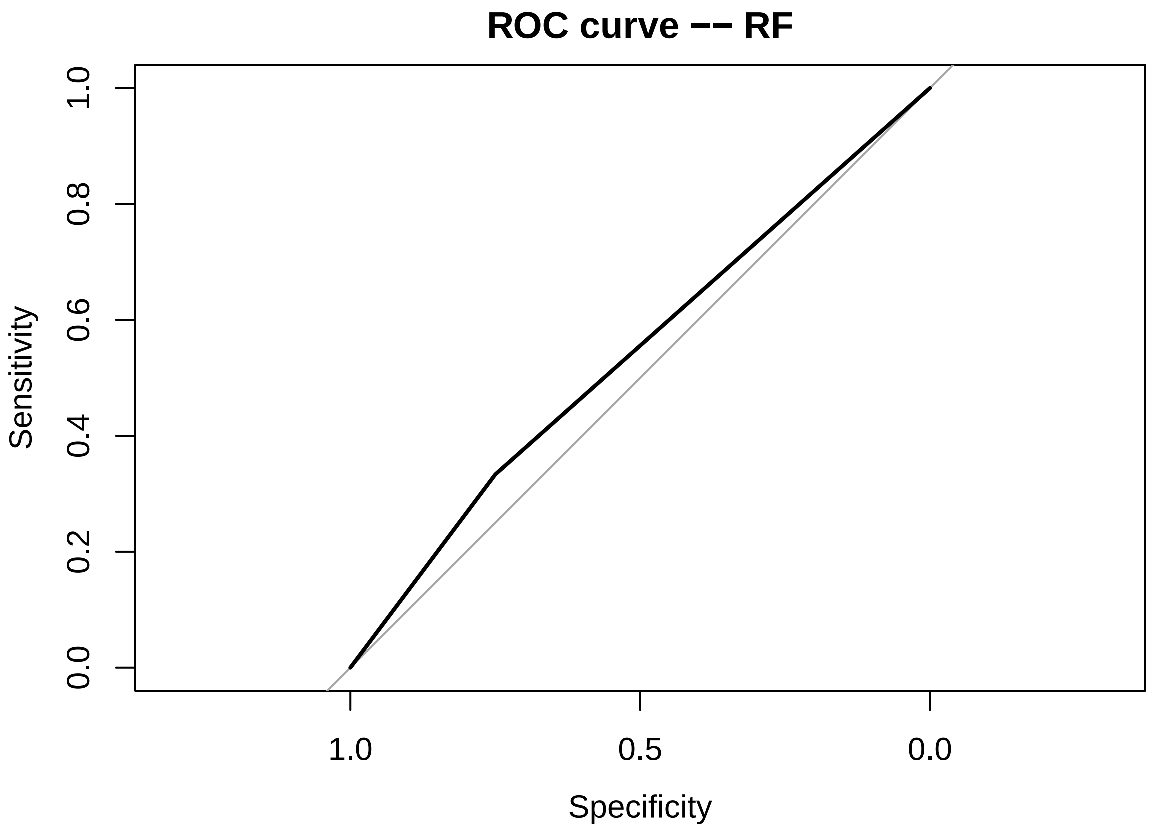
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 3 1
##          1 4 2
##
##                Accuracy : 0.5
##                  95% CI : (0.1871, 0.8129)
##     No Information Rate : 0.7
##     P-Value [Acc > NIR] : 0.9527
##
##                   Kappa : 0.0741
##
##  Mcnemar's Test P-Value : 0.3711
##
##             Sensitivity : 0.4286
##             Specificity : 0.6667
##          Pos Pred Value : 0.7500
##          Neg Pred Value : 0.3333
##               Precision : 0.7500
##                  Recall : 0.4286
##                      F1 : 0.5455
##              Prevalence : 0.7000
##          Detection Rate : 0.3000
##    Detection Prevalence : 0.4000
##       Balanced Accuracy : 0.5476
##
##        'Positive' Class : 0
##
```

```
roc_score=roc(test1$trauma, as.numeric(test_predrf)) #AUC score
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
plot(roc_score ,main ="ROC curve -- RF")
```



```
pred1 <- predict(rf1, test1, type="raw")


act_pred <- data.frame(observed = test1$trauma, predicted =
                      pred1)
act_pred
```

```
##    observed predicted
## 1         1         1
## 2         0         0
## 3         1         0
## 4         1         0
## 5         1         0
## 6         0         0
## 7         1         0
## 8         0         1
## 9         1         1
## 10        0         0
```

#The random forest model is 50% accurate. It also misclassified positive cases much more than the other models.

#Thus, the KNN model is the best at predicting trauma.

#In this analysis, I began by cleaning, restructuring, and preparing the data for modeling. To do this, I converted categorical data into binary responses and factor variables. Then, I combined variables into lifestyle score, to make modeling easier. I also transformed hours sitting per day into proportion of the day sitting. After I finished setting the data up, I began to explore it. I first wanted to see which variables were correlated with diagnosis. To do this, I created bar plots divided up by diagnosis. From these graphs I learned that there were only a few variables that were correlated with diagnosis. This could be due to the low number of responses in the data set, and the even smaller number of altered responses for semen.

#With this information I wanted to see what other relationships I could discover. After more exploration I discovered that lifestyle score had several interesting properties. Lifestyle score and trauma seemed to be inversely correlated: that is more individuals with trauma generally had lower lifestyle scores than those without trauma. Another interesting finding was that there were more individuals with a child disease who had lifestyle scores on the higher end. However, there were more individuals without a child disease that had extreme high or low values of lifestyle. Lastly, there were more individuals with altered semen who also had lower lifestyle scores, and vice versa. From these visualizations, I began to see that lifestyle score, diagnosis, trauma, and child disease all seemed to be very important to the quality of ones health and safety.

#Despite there being such a low number of observations and 10 unique lifestyle scores, I wanted to see how accurately I could predict lifestyle score. This could be extremely important as it could help identify risk factors for things such as drug and alcohol abuse, as well the impacts of good and bad lifestyles. I wanted to use to approaches: a linear regression model and a random forest model. To build my linear model, I first created a full model to see which variables had significant impacts on lifestyle score. From this I determined that Diagnosis and trauma were statistically significant. From here I chose to use a backwards selection approach to build my model. This method also chose Diagnosis and trauma as the best subset. The next best subset included surgery. This made me think: surger and trauma seemed to logically go hand in hand. Therefore, I decided to include them as an interaction term. This final model(lifestyle ~ trauma + Diagnosis + I(trauma *surgery)), provided the best r-squared with the lowest root mean squared error. Next, I trained my random forest model. I used various methods to fine tune it(optimal mtry and ntry values). Testing both models on the test data, the linear regression model was more accurate, with 60% of its predictions being completely accurate, as compared to the random forests 40%. I think it is worth mentioning that the 2 incorrect predictions were 6 and 9, where the actual values were 7 and 7 respectively. Thus, the predictions were only 1 and 2 off. I think that this model can atleast predict whether an individual has a good or bad lifestyle. The random forest had the same issue, however, it only predicted 2 of the 5 values exactly.

#Next I aimed to model whether an individual has experience serious trauma in their life. As this is a binary response variable I first started developing a logistic model. I used the same approach as the linear model and found that lifestyle, diagnosis, and interestingly the spring season were statistically significant in predicting the probability of an individual having trauma. Next, I wanted to see if the model was different if I simply used lifestyle score rather than the variables that I used to create lifestyle score. To do this I performed a drop in deviance test, and found that there was no significant different between the two. From there, I used to bestglm function to build the best logistic model based on AIC. I discovered that age also could be important in modeling trauma, so I decided to include it in the model. I also tested various interactions terms and tranformations on age and other variables, and found that none of these improved the model. So my final logistic model was: trauma ~ Diagnosis + lifestyle + Season + Age. Testing this model on the test data, I achieved 60% accuracy. I wanted to train a KNN model, as they tend to do well on smaller sample sizes. Using K-fold cross validation, I model trauma on Diagnosis, lifestyle, and Season. Interestingly, including age in the knn model significantly decreased its efficacy. The variables I chose to include in this model created the most acccurate knn model. The confusion matrix I created based off the knn model predicted 60% accuracy, however testing the model on the test set, it surpassed expectations. It was 80% accurate. With such a low number of observations this is very. I trained a random forest model as well in the same manner as before, but it was not as effective as the knn model. The KNN model seemed to perform the best by comparing the actual and predicted values, however, based on the confusion matrixes I produced for each model, the logistic model had the best balanced accuracy and f1 score. Thus, based on which parameter I could've chosen to pursue, a different model may have been better suited.

#After exploring and modeling this dataset, I hope that I was able to show how important it is to lead as healthy a lifestyle as possible. Conditions involving serious trauma, drug abuse, and altered semen can be extremely impactful on a persons life. In order to reduce the odds that you may be a victim of such things, people should lead as healthy a lifestyle as possible. Even things that may seem inconsequential, such as recreational drug use, fevers, how much you sit, and whether or not you were sick as a child, can have much more serious consequences.