Autism Detection Using R

Rajeshwar Reddy Konkisa

18/Sept/2021

```
library(plyr)
library(readr)
```

```
## Warning: package 'readr' was built under R version 4.0.4
```

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 4.0.4
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
## Loading required package: ggplot2
```

About this dataset

What is Autism

Autism, or autism spectrum disorder (ASD), refers to a broad range of conditions characterized by challenges with social skills, repetitive behaviors, speech and nonverbal communication.

Causes and Challenges It is mostly influenced by a combination of genetic and environmental factors. Because autism is a spectrum disorder, each person with autism has a distinct set of strengths and challenges.

The ways in which people with autism learn, think and problem-solve can range from highly skilled to severely challenged. Research has made clear that high quality early intervention can improve learning, communication and social skills, as well as underlying brain development. Yet the diagnostic process can take several years.

The Role of Machine Learning This dataset is composed of survey results for more than 700 people who filled an app form. There are labels portraying whether the person received a diagnosis of autism, allowing machine learning models to predict the likelihood of having autism, therefore allowing healthcare professionals prioritize their resources.

```
data <- read.csv(file.choose())
head(data)
```

```
##   X A1_Score A2_Score A3_Score A4_Score A5_Score A6_Score A7_Score A8_Score
## 1 0        1        1        1        1        0        0        1        1
## 2 1        1        1        0        1        0        0        0        1
## 3 2        1        1        0        1        1        0        1        1
## 4 3        1        1        0        1        0        0        1        1
## 5 4        1        0        0        0        0        0        0        1
## 6 5        1        1        1        1        1        0        1        1
##   A9_Score A10_Score age gender ethnicity jundice austim contry_of_res
## 1        0         0   9      0        10       0      0            64
## 2        0         1   7      1         4       0      1            13
## 3        1         1  10      1         4       1      1            56
## 4        0         1  18      0        10       0      1            64
## 5        0         0  23      0         0       0      0            22
## 6        1         1  19      1         6       1      0            64
##   used_app_before result age_desc relation Class.ASD
## 1               0      6        0        5         0
## 2               0      5        0        5         0
## 3               0      8        0        3         1
## 4               0      6        0        5         0
## 5               0      2        0        0         0
## 6               0      9        0        5         1
```

Lets now check the statistical aspects of our data.

```
summary(data)
```

```
##        X            A1_Score         A2_Score         A3_Score
##  Min.   :  0.0   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:175.8   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :351.5   Median :1.0000   Median :0.0000   Median :0.0000
##  Mean   :351.5   Mean   :0.7216   Mean   :0.4531   Mean   :0.4574
##  3rd Qu.:527.2   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :703.0   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##     A4_Score         A5_Score         A6_Score         A7_Score
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.4957   Mean   :0.4986   Mean   :0.2841   Mean   :0.4176
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
```

```
##     A8_Score         A9_Score        A10_Score           age
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 0.0
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 4.0
##  Median :1.0000   Median :0.0000   Median :1.0000   Median :10.0
##  Mean   :0.6491   Mean   :0.3239   Mean   :0.5739   Mean   :12.2
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:18.0
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :45.0
##      gender         ethnicity         jundice           austim
##  Min.   :0.0000   Min.   : 0.000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.: 1.000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :1.0000   Median : 5.000   Median :0.00000   Median :0.0000
##  Mean   :0.5213   Mean   : 5.305   Mean   :0.09801   Mean   :0.1293
##  3rd Qu.:1.0000   3rd Qu.:10.000   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :11.000   Max.   :1.00000   Max.   :1.0000
##  contry_of_res   used_app_before       result          age_desc    relation
##  Min.   : 0.00   Min.   :0.00000   Min.   : 0.000   Min.   :0   Min.   :0.000
##  1st Qu.:29.00   1st Qu.:0.00000   1st Qu.: 3.000   1st Qu.:0   1st Qu.:4.000
##  Median :43.00   Median :0.00000   Median : 4.000   Median :0   Median :5.000
##  Mean   :44.05   Mean   :0.01705   Mean   : 4.875   Mean   :0   Mean   :4.099
##  3rd Qu.:63.00   3rd Qu.:0.00000   3rd Qu.: 7.000   3rd Qu.:0   3rd Qu.:5.000
##  Max.   :66.00   Max.   :1.00000   Max.   :10.000   Max.   :0   Max.   :5.000
##    Class.ASD
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2685
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

Lets now Check for any missing values in the dataset

```
sum(is.na(data))
```

```
## [1] 0
```

This Data Set has no null values. So we need not perform any imputation. Lets Plot the values to see their distributions. It will help us to understand the distribution of the data.

Plotting country of res vs autism

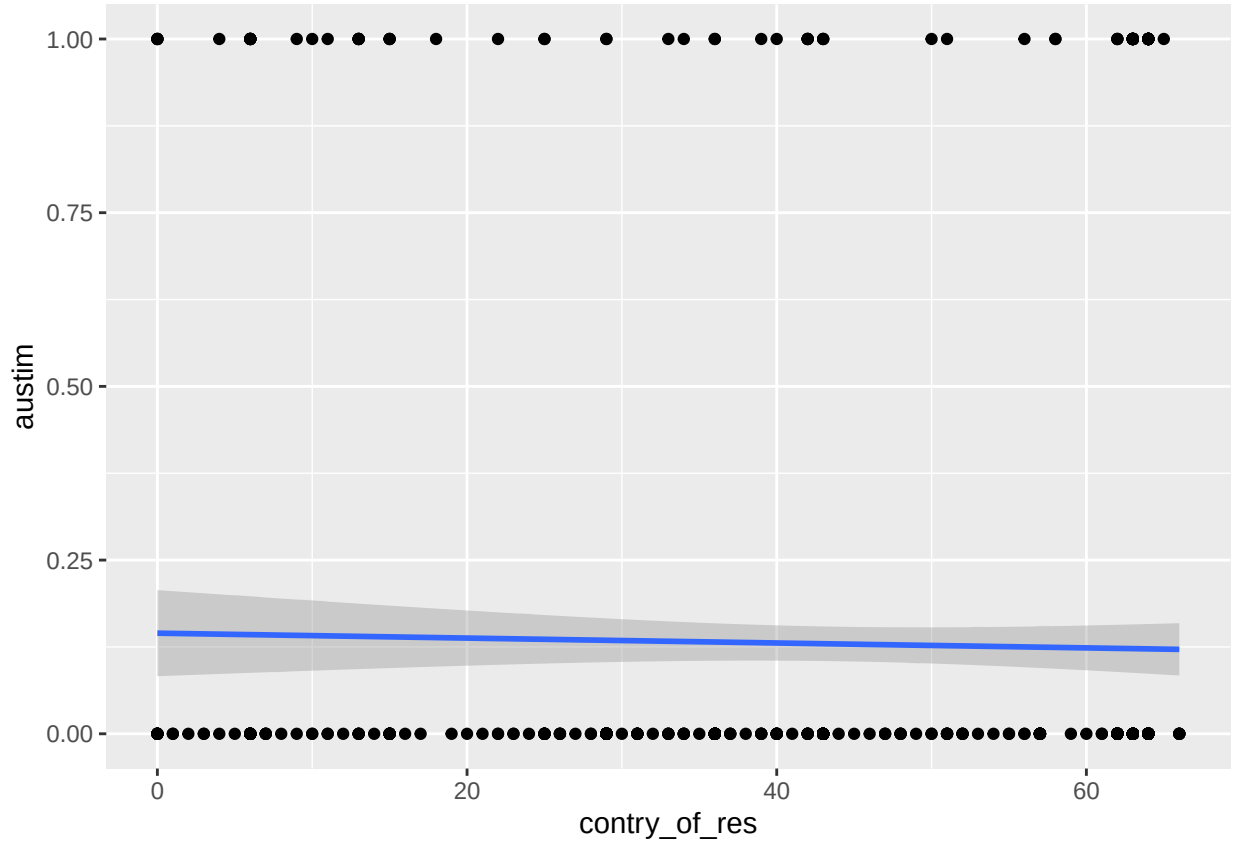
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=austim)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting country of res vs result

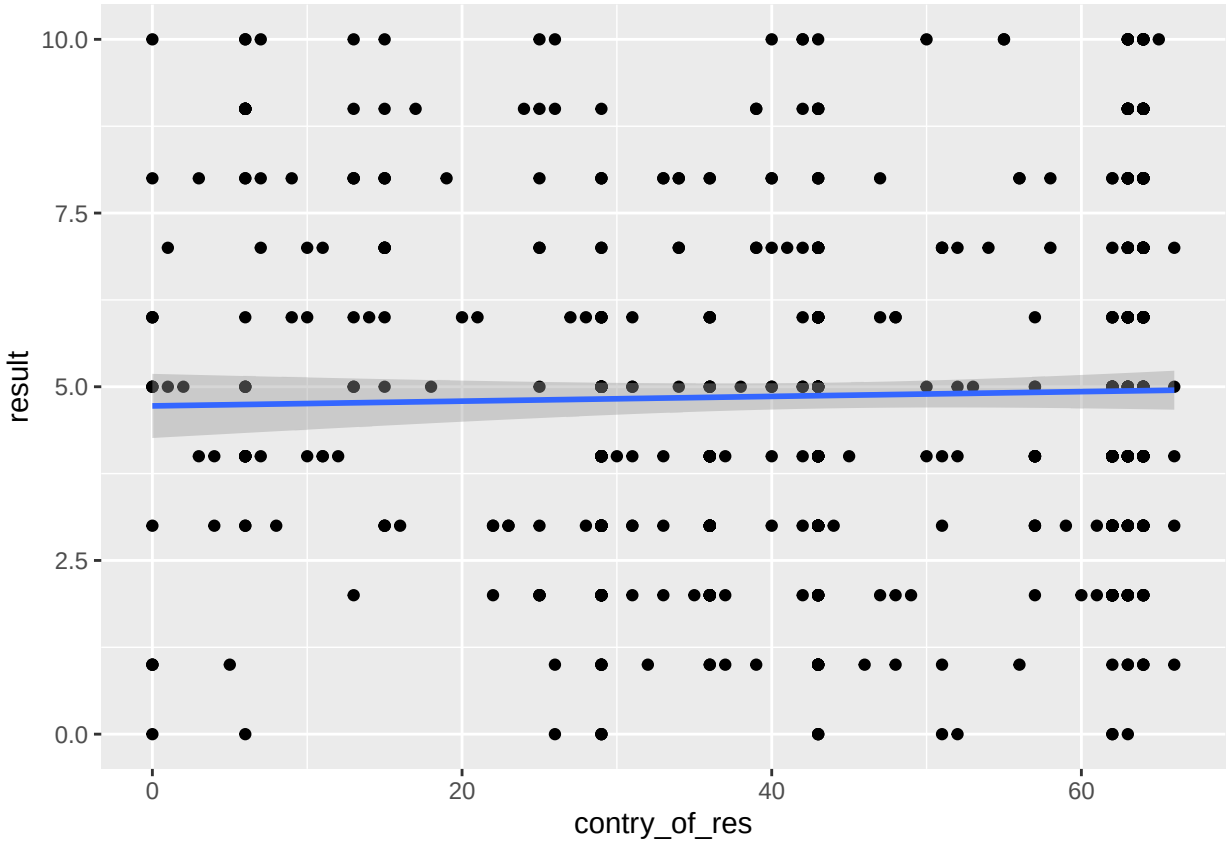
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=result)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting country of res vs jaundice

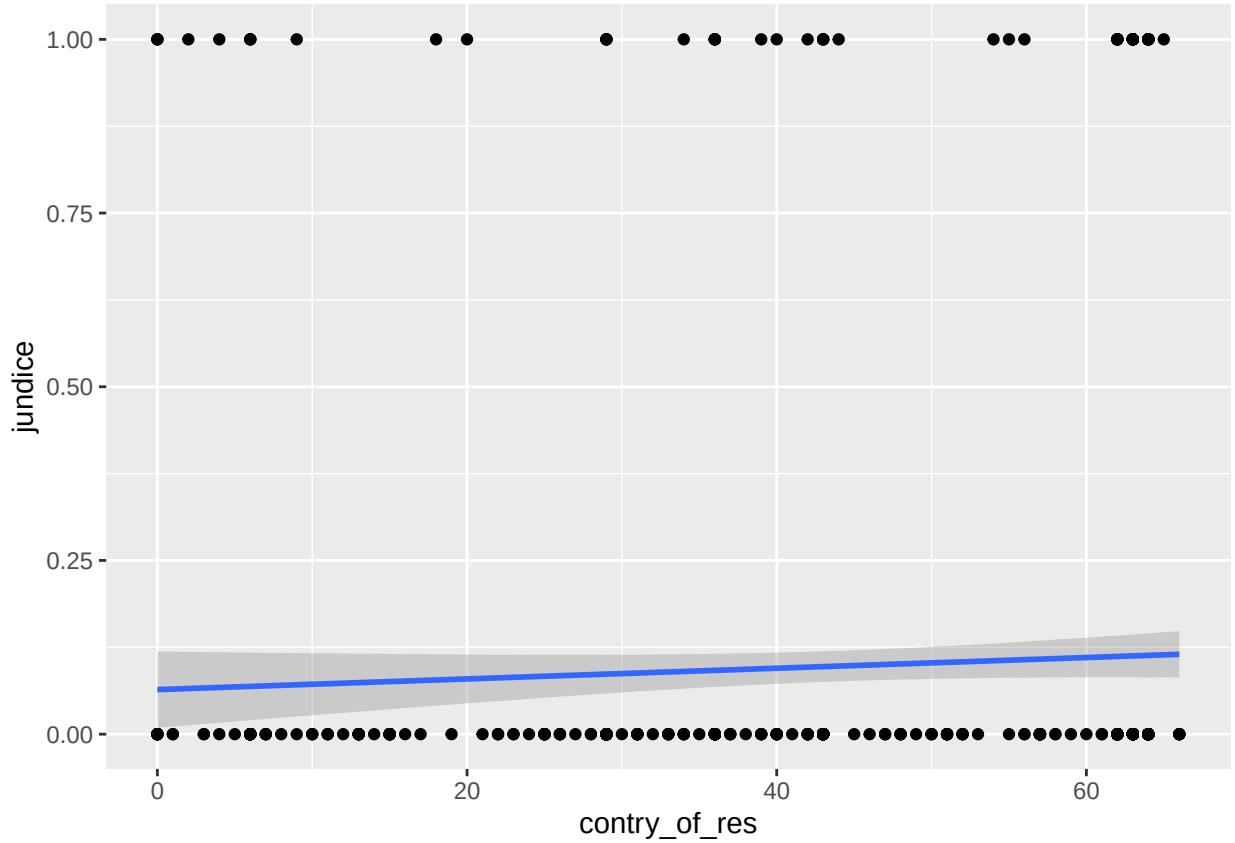
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=jundice)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting country of res vs ethniciry

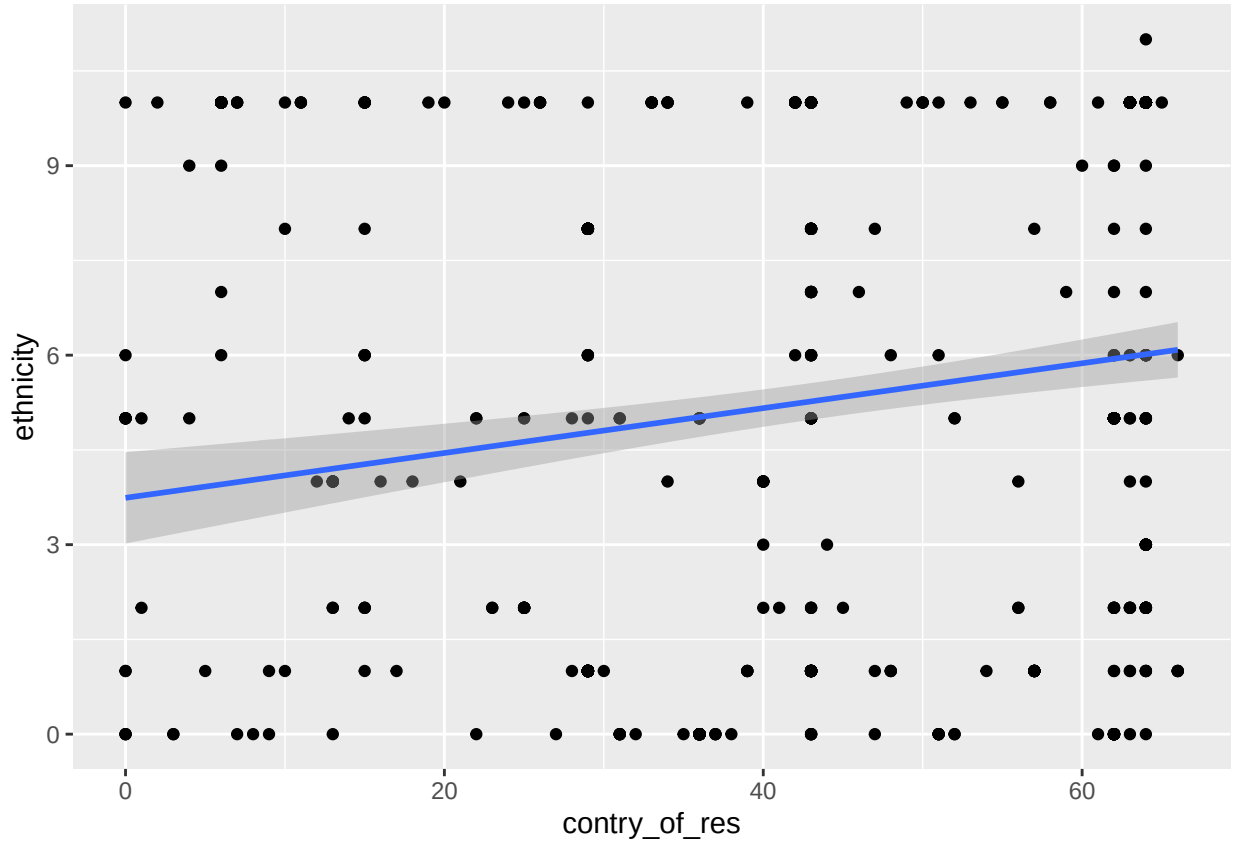
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=ethnicity)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting country of res vs age

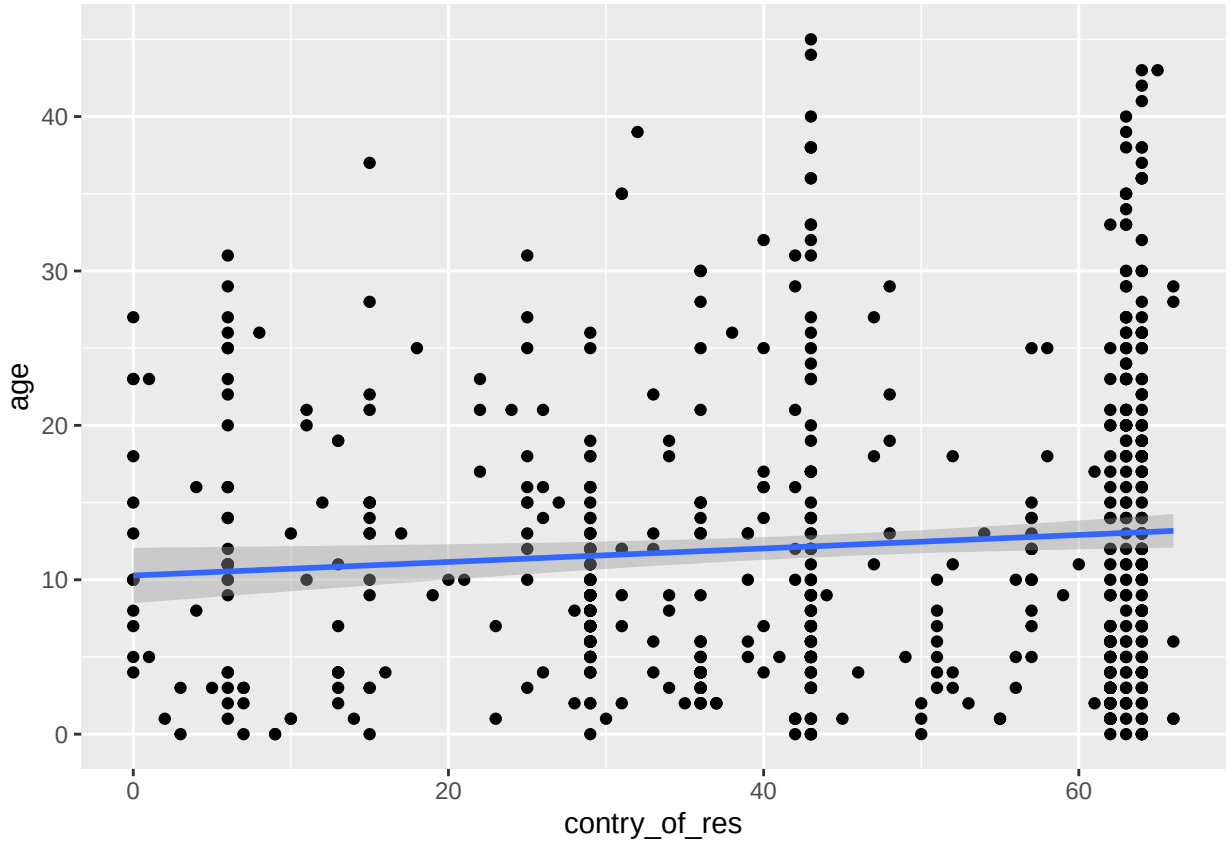
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=age)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting country of res vs gender

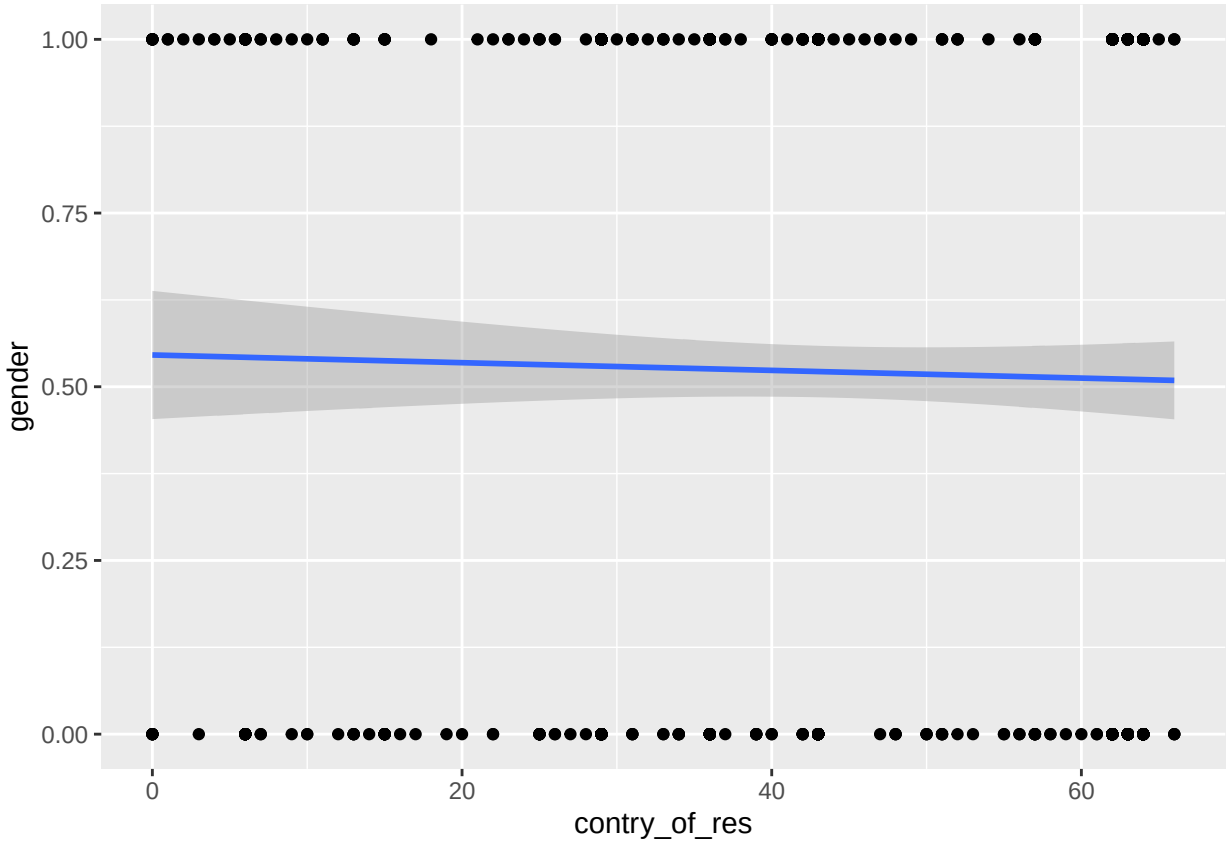
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=gender)) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting gender vs Class.ASD

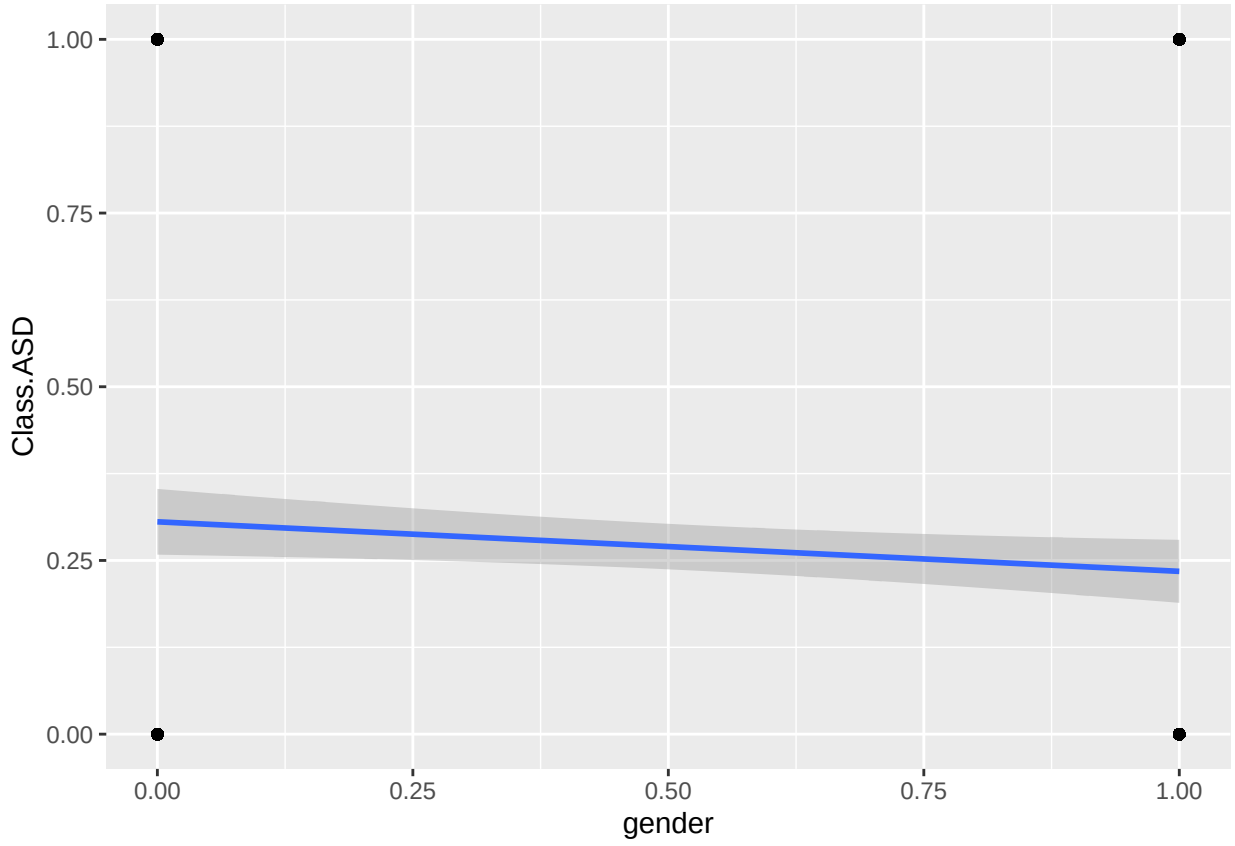
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=gender, y=Class.ASD )) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting age vs Class.ASD

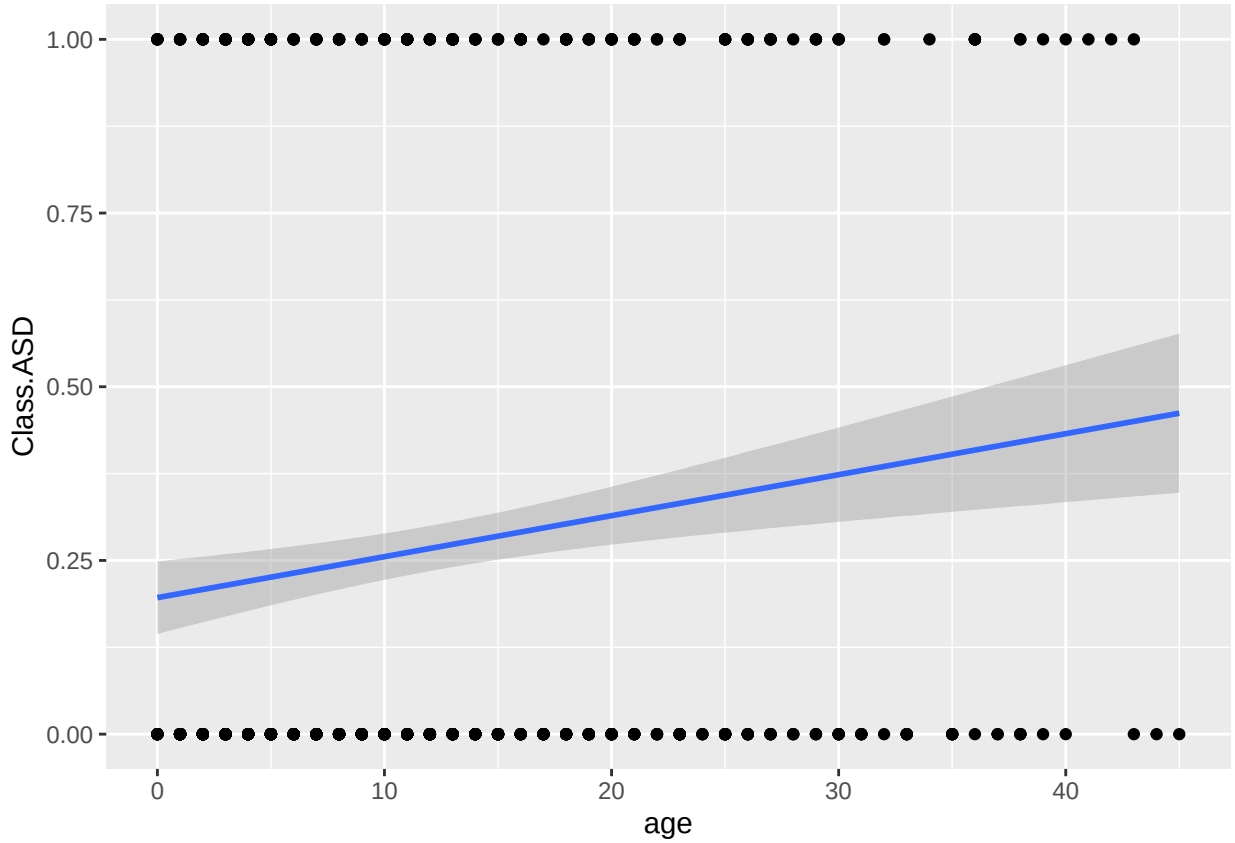
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=age, y=Class.ASD )) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting ethnicity vs Class.ASD

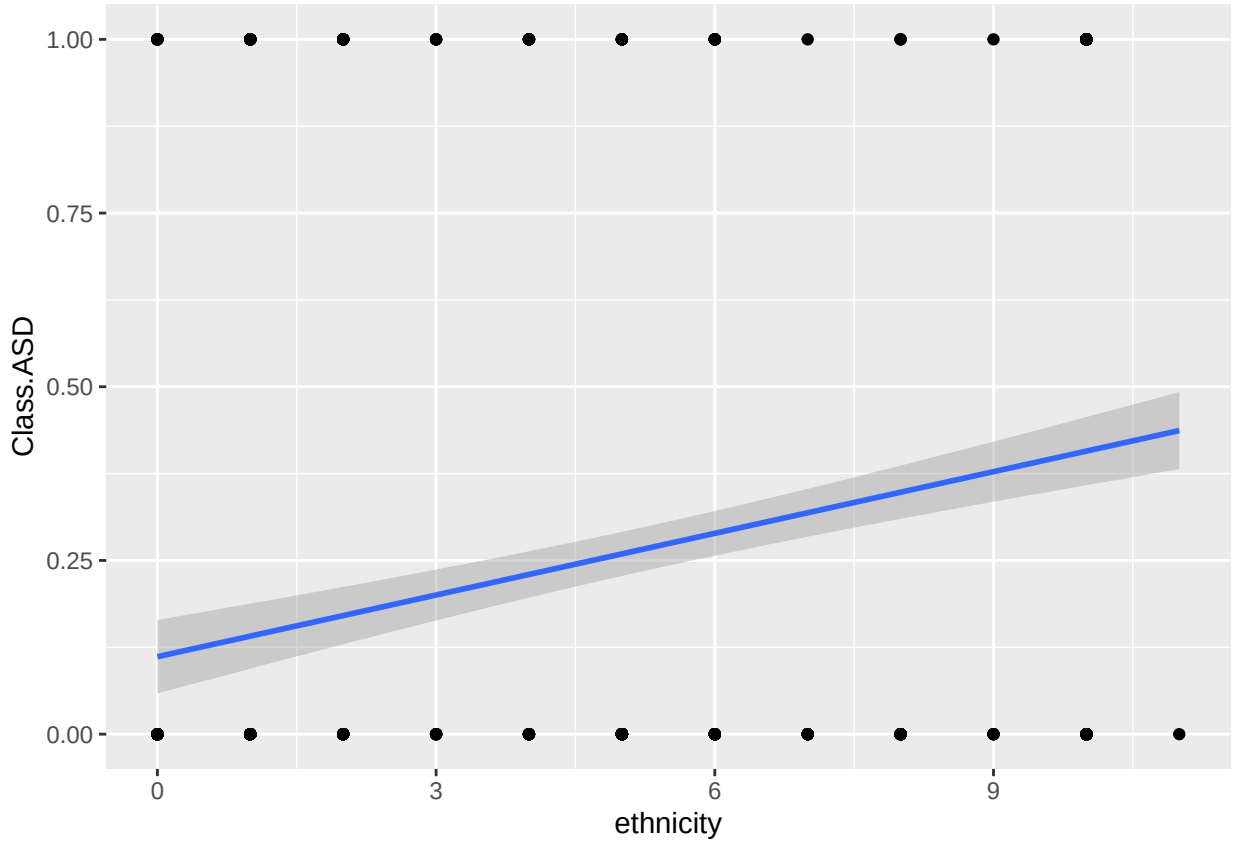
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=ethnicity, y=Class.ASD )) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting contry_of_res vs Class.ASD

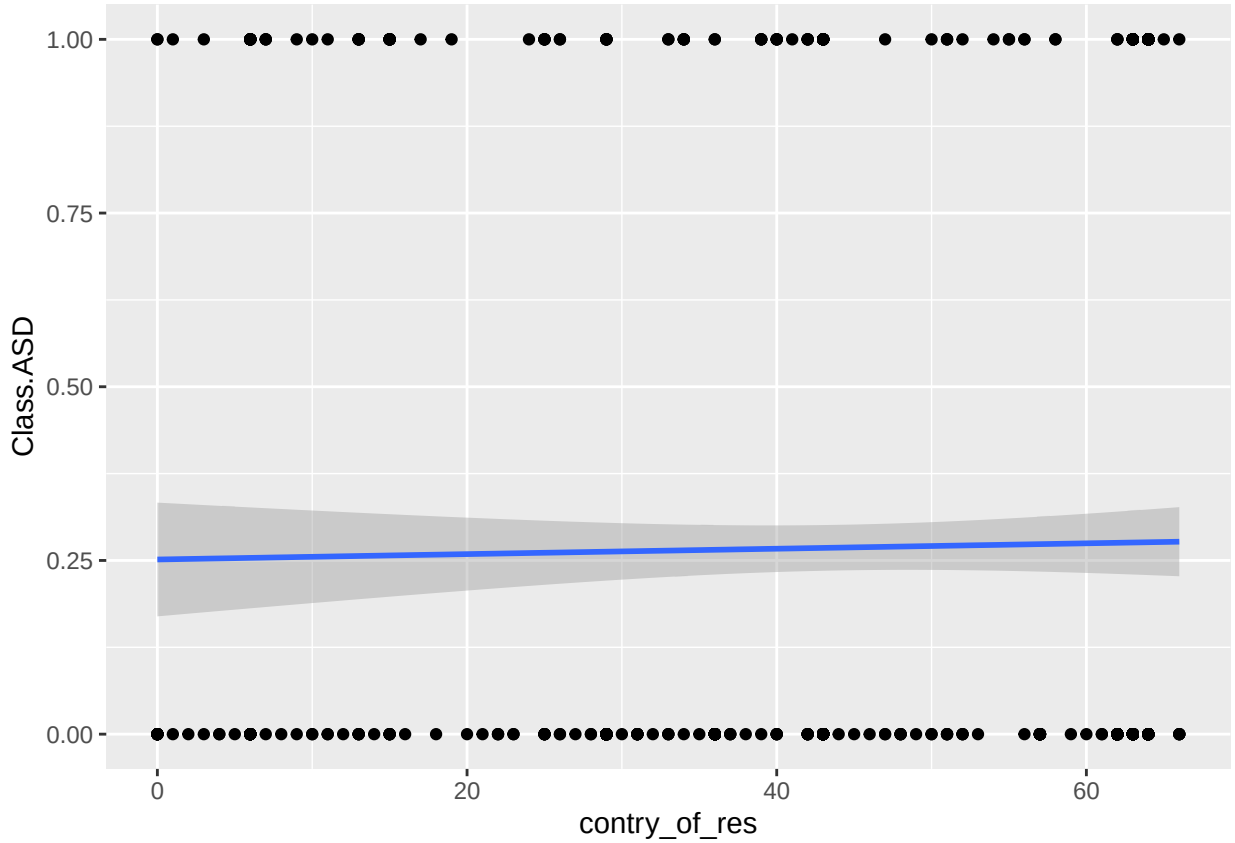
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=contry_of_res, y=Class.ASD )) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Plotting jaundice vs Class.ASD

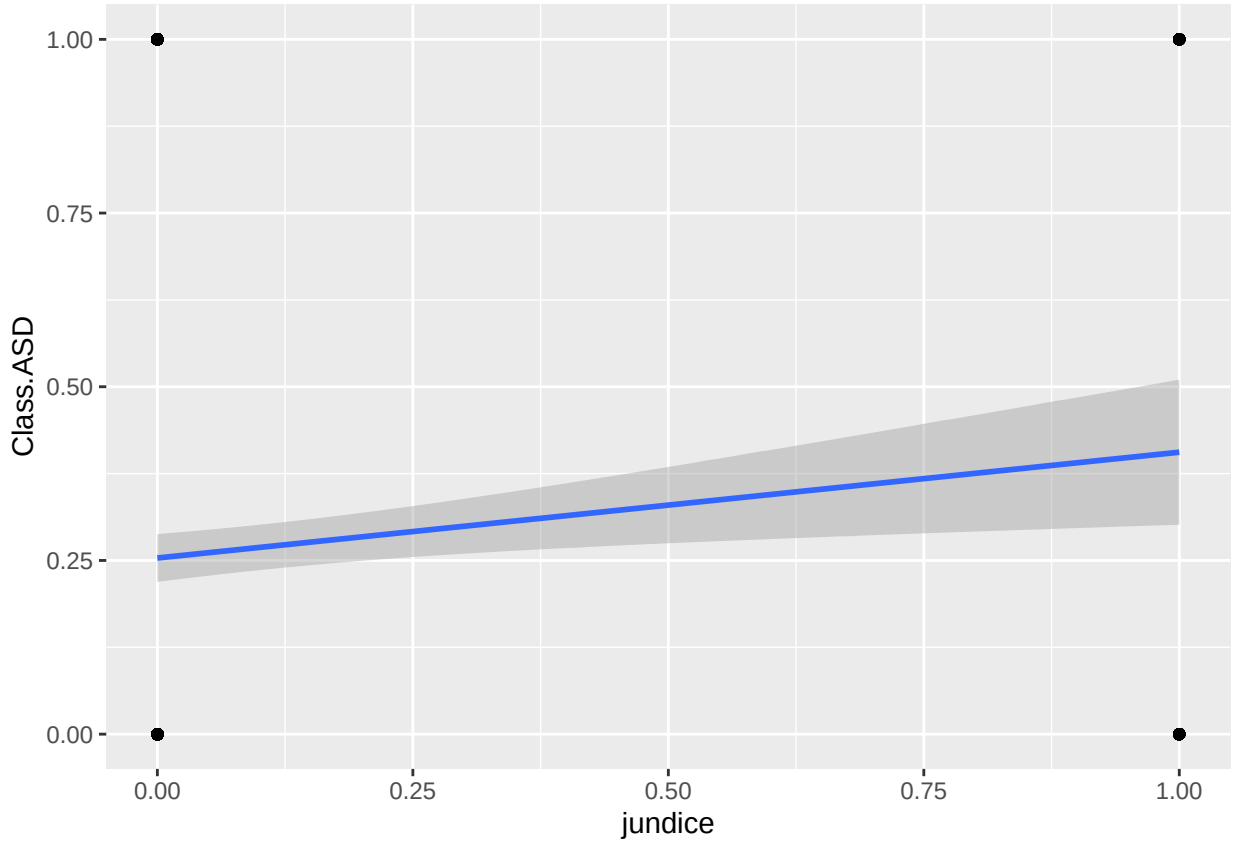
```
library(ggplot2)

geom_smooth(method="auto", se=TRUE, fullrange=FALSE, level=0.95)
```

```
## geom_smooth: na.rm = FALSE, orientation = NA, se = TRUE
## stat_smooth: na.rm = FALSE, orientation = NA, se = TRUE, fullrange = FALSE, level = 0.95, method = au
## position_identity
```

```
# Add the regression line
ggplot(data, aes(x=jundice, y=Class.ASD )) +
  geom_point()+
  geom_smooth(method=lm)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Heat map to check correlation

Compute the correlation matrix Correlation matrix can be created using the R function cor() :

```
cormat <- round(cor(data),2)
```

```
## Warning in cor(data): the standard deviation is zero
```

```
head(cormat)
```

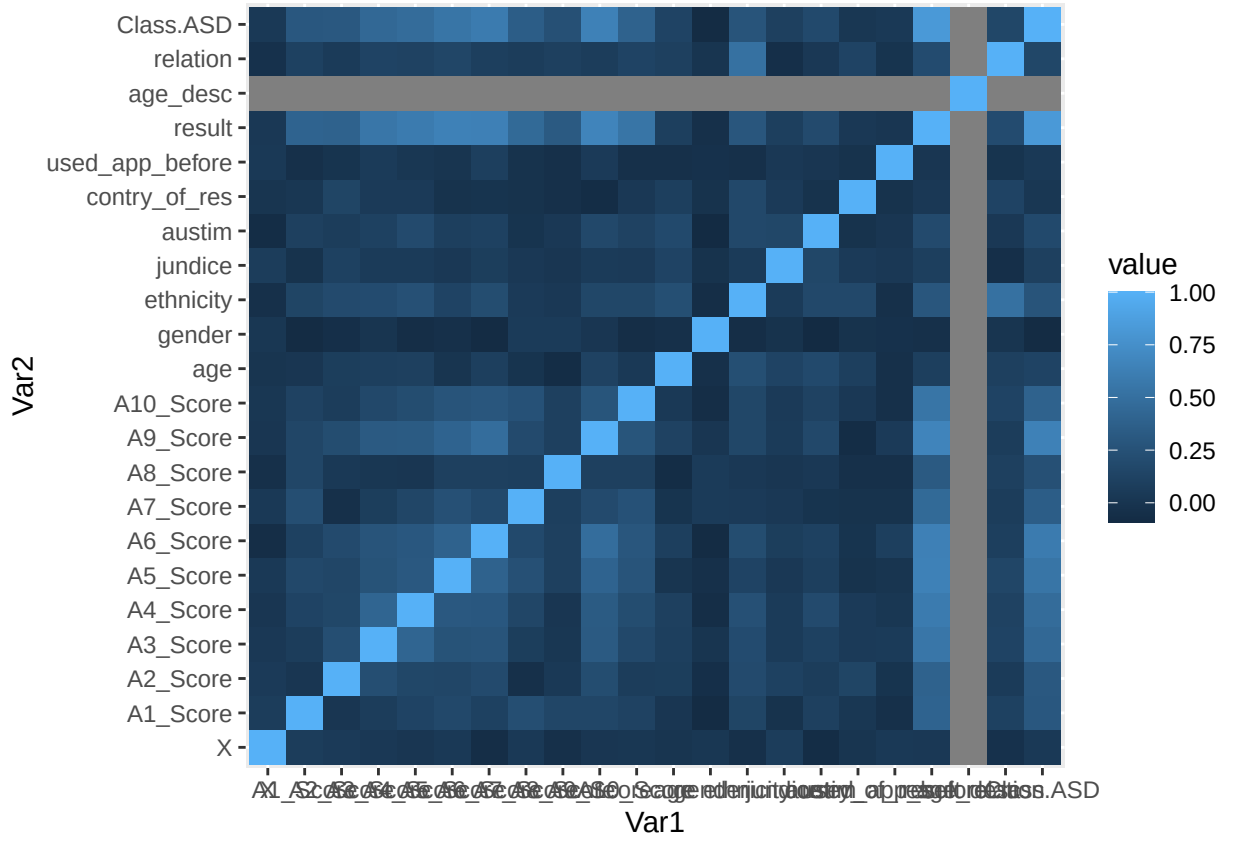
```
##             X A1_Score A2_Score A3_Score A4_Score A5_Score A6_Score A7_Score
## X        1.00     0.07     0.05     0.03     0.01     0.04    -0.06     0.04
## A1_Score 0.07     1.00     0.01     0.07     0.13     0.17     0.11     0.22
## A2_Score 0.05     0.01     1.00     0.22     0.16     0.15     0.19    -0.04
## A3_Score 0.03     0.07     0.22     1.00     0.41     0.26     0.27     0.08
## A4_Score 0.01     0.13     0.16     0.41     1.00     0.31     0.30     0.15
## A5_Score 0.04     0.17     0.15     0.26     0.31     1.00     0.39     0.24
##          A8_Score A9_Score A10_Score  age gender ethnicity jundice austim
## X           -0.04     0.01      0.02 0.00   0.02     -0.04    0.07  -0.07
## A1_Score     0.15     0.15      0.12 0.01  -0.08      0.14   -0.02   0.10
## A2_Score     0.04     0.21      0.07 0.08  -0.05      0.19    0.11   0.07
## A3_Score     0.02     0.32      0.17 0.09   0.00      0.20    0.06   0.11
## A4_Score     0.01     0.33      0.21 0.10  -0.06      0.24    0.06   0.19
## A5_Score     0.10     0.40      0.27 0.00  -0.04      0.13    0.03   0.09
##          contry_of_res used_app_before result age_desc relation Class.ASD
## X                 0.00            0.04   0.03       NA    -0.03      0.04
```

```
## A1_Score          0.02          -0.04   0.40       NA     0.11       0.30
## A2_Score          0.14          -0.01   0.39       NA     0.06       0.31
## A3_Score          0.05           0.06   0.55       NA     0.13       0.44
## A4_Score          0.05           0.02   0.59       NA     0.12       0.47
## A5_Score         -0.02           0.00   0.64       NA     0.15       0.54
```

```
library(reshape2)
melted_cormat <- melt(cormat)
head(melted_cormat)
```

```
##       Var1 Var2 value
## 1        X    X  1.00
## 2 A1_Score    X  0.07
## 3 A2_Score    X  0.05
## 4 A3_Score    X  0.03
## 5 A4_Score    X  0.01
## 6 A5_Score    X  0.04
```

```
library(ggplot2)
ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile()
```




```
# Get lower triangle of the correlation matrix
  get_lower_tri<-function(cormat){
```

```
  cormat[upper.tri(cormat)] <- NA
  return(cormat)
}
# Get upper triangle of the correlation matrix
get_upper_tri <- function(cormat){
  cormat[lower.tri(cormat)]<- NA
  return(cormat)
}
```

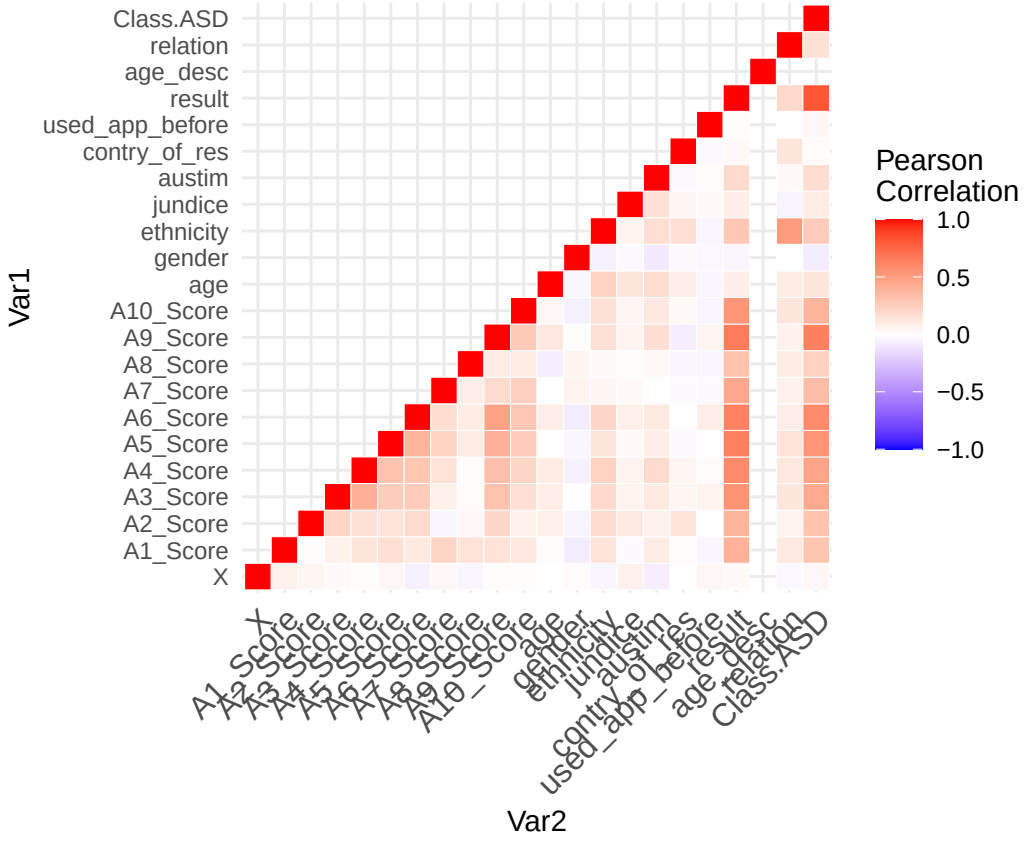
```
upper_tri <- get_upper_tri(cormat)
upper_tri
```

```
##                  X A1_Score A2_Score A3_Score A4_Score A5_Score A6_Score
## X                1     0.07     0.05     0.03     0.01     0.04    -0.06
## A1_Score        NA     1.00     0.01     0.07     0.13     0.17     0.11
## A2_Score        NA       NA     1.00     0.22     0.16     0.15     0.19
## A3_Score        NA       NA       NA     1.00     0.41     0.26     0.27
## A4_Score        NA       NA       NA       NA     1.00     0.31     0.30
## A5_Score        NA       NA       NA       NA       NA     1.00     0.39
## A6_Score        NA       NA       NA       NA       NA       NA     1.00
## A7_Score        NA       NA       NA       NA       NA       NA       NA
## A8_Score        NA       NA       NA       NA       NA       NA       NA
## A9_Score        NA       NA       NA       NA       NA       NA       NA
## A10_Score       NA       NA       NA       NA       NA       NA       NA
## age             NA       NA       NA       NA       NA       NA       NA
## gender          NA       NA       NA       NA       NA       NA       NA
## ethnicity       NA       NA       NA       NA       NA       NA       NA
## jundice         NA       NA       NA       NA       NA       NA       NA
## austim          NA       NA       NA       NA       NA       NA       NA
## contry_of_res   NA       NA       NA       NA       NA       NA       NA
## used_app_before NA       NA       NA       NA       NA       NA       NA
## result          NA       NA       NA       NA       NA       NA       NA
## age_desc        NA       NA       NA       NA       NA       NA       NA
## relation        NA       NA       NA       NA       NA       NA       NA
## Class.ASD       NA       NA       NA       NA       NA       NA       NA
##                 A7_Score A8_Score A9_Score A10_Score   age gender ethnicity
## X                   0.04    -0.04     0.01      0.02  0.00   0.02     -0.04
## A1_Score            0.22     0.15     0.15      0.12  0.01  -0.08      0.14
## A2_Score           -0.04     0.04     0.21      0.07  0.08  -0.05      0.19
## A3_Score            0.08     0.02     0.32      0.17  0.09   0.00      0.20
## A4_Score            0.15     0.01     0.33      0.21  0.10  -0.06      0.24
## A5_Score            0.24     0.10     0.40      0.27  0.00  -0.04      0.13
## A6_Score            0.18     0.10     0.48      0.29  0.09  -0.08      0.21
## A7_Score            1.00     0.09     0.19      0.25 -0.01   0.06      0.05
## A8_Score              NA     1.00     0.10      0.10 -0.07   0.06      0.03
## A9_Score              NA       NA     1.00      0.28  0.12   0.01      0.16
## A10_Score             NA       NA       NA      1.00  0.04  -0.06      0.16
## age                   NA       NA       NA        NA  1.00  -0.04      0.23
## gender                NA       NA       NA        NA    NA   1.00     -0.06
## ethnicity             NA       NA       NA        NA    NA     NA      1.00
## jundice               NA       NA       NA        NA    NA     NA        NA
## austim                NA       NA       NA        NA    NA     NA        NA
## contry_of_res         NA       NA       NA        NA    NA     NA        NA
```

```
## used_app_before       NA       NA       NA        NA    NA     NA        NA
## result                NA       NA       NA        NA    NA     NA        NA
## age_desc              NA       NA       NA        NA    NA     NA        NA
## relation              NA       NA       NA        NA    NA     NA        NA
## Class.ASD             NA       NA       NA        NA    NA     NA        NA
##                  jundice austim contry_of_res used_app_before result age_desc
## X                   0.07  -0.07          0.00            0.04   0.03       NA
## A1_Score           -0.02   0.10          0.02           -0.04   0.40       NA
## A2_Score            0.11   0.07          0.14           -0.01   0.39       NA
## A3_Score            0.06   0.11          0.05            0.06   0.55       NA
## A4_Score            0.06   0.19          0.05            0.02   0.59       NA
## A5_Score            0.03   0.09         -0.02            0.00   0.64       NA
## A6_Score            0.08   0.11         -0.01            0.09   0.63       NA
## A7_Score            0.03  -0.01         -0.02           -0.02   0.45       NA
## A8_Score            0.01   0.03         -0.04           -0.04   0.32       NA
## A9_Score            0.06   0.17         -0.07            0.05   0.66       NA
## A10_Score           0.05   0.12          0.03           -0.04   0.54       NA
## age                 0.13   0.18          0.09           -0.04   0.09       NA
## gender             -0.02  -0.09         -0.02           -0.03  -0.04       NA
## ethnicity           0.06   0.17          0.17           -0.04   0.29       NA
## jundice             1.00   0.16          0.05            0.03   0.09       NA
## austim                NA   1.00         -0.02            0.01   0.19       NA
## contry_of_res         NA     NA          1.00           -0.02   0.03       NA
## used_app_before       NA     NA            NA            1.00   0.01       NA
## result                NA     NA            NA              NA   1.00       NA
## age_desc              NA     NA            NA              NA     NA        1
## relation              NA     NA            NA              NA     NA       NA
## Class.ASD             NA     NA            NA              NA     NA       NA
##                  relation Class.ASD
## X                   -0.03      0.04
## A1_Score             0.11      0.30
## A2_Score             0.06      0.31
## A3_Score             0.13      0.44
## A4_Score             0.12      0.47
## A5_Score             0.15      0.54
## A6_Score             0.09      0.59
## A7_Score             0.07      0.35
## A8_Score             0.10      0.24
## A9_Score             0.07      0.64
## A10_Score            0.13      0.39
## age                  0.10      0.13
## gender               0.00     -0.08
## ethnicity            0.51      0.27
## jundice             -0.05      0.10
## austim               0.03      0.18
## contry_of_res        0.13      0.02
## used_app_before     -0.01      0.04
## result               0.20      0.82
## age_desc               NA        NA
## relation             1.00      0.16
## Class.ASD              NA      1.00
```

```r
# Melt the correlation matrix
library(reshape2)
```

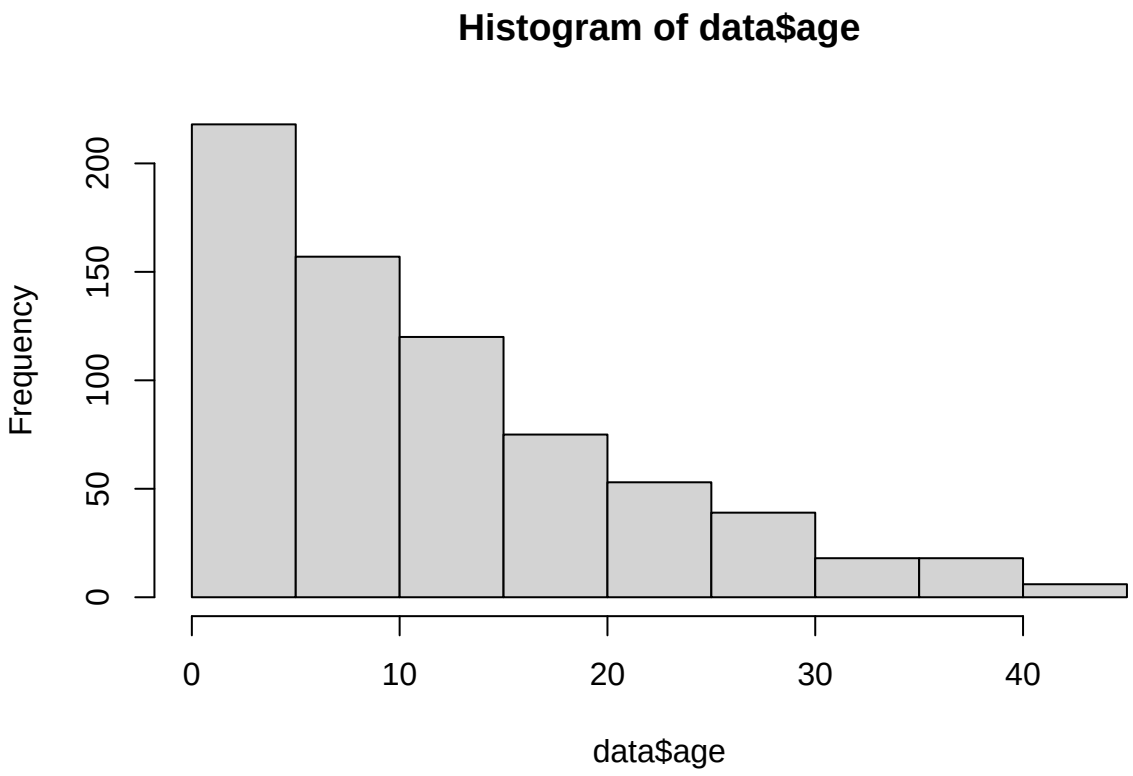
```
melted_cormat <- melt(upper_tri, na.rm = TRUE)
# Heatmap
library(ggplot2)
ggplot(data = melted_cormat, aes(Var2, Var1, fill = value))+
 geom_tile(color = "white")+
 scale_fill_gradient2(low = "blue", high = "red", mid = "white",
   midpoint = 0, limit = c(-1,1), space = "Lab",
   name="Pearson\nCorrelation") +
  theme_minimal()+
 theme(axis.text.x = element_text(angle = 45, vjust = 1,
    size = 12, hjust = 1))+
 coord_fixed()
```



Age Histogram

To see the distribution of age

```
hist(data$age)
```



Logistic Regression

Logistic regression are the most common models used with binary outcomes.Events are coded as binary variables with a value of 1 representing the occurrence of a target outcome, and a value of zero representing its absence. OLS can also model binary variables using linear probability models.

Now we will split the data into Train & test Set

```
library(caret)
library(dplyr)

index <-  createDataPartition(data$Class.ASD,p= .8, times=1, list=F)

train <-  data[index,]
test <- data[-index,]
```

Fitting a logistic regression model:

We will use logistic regression model on the training set which is 80% of the total data.

```
logistic<- glm(Class.ASD~., data=train, family='binomial')
```

```
## Warning: glm.fit: algorithm did not converge
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```r
summary(logistic)
```

```
##
## Call:
## glm(formula = Class.ASD ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## -3.147e-05  -2.110e-08  -2.110e-08   2.110e-08   3.063e-05
##
## Coefficients: (2 not defined because of singularities)
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)     -2.886e+02  5.454e+04  -0.005    0.996
## X                1.156e-03  2.076e+01   0.000    1.000
## A1_Score         4.407e+01  1.371e+04   0.003    0.997
## A2_Score         4.402e+01  1.105e+04   0.004    0.997
## A3_Score         4.408e+01  1.132e+04   0.004    0.997
## A4_Score         4.435e+01  1.186e+04   0.004    0.997
## A5_Score         4.443e+01  1.261e+04   0.004    0.997
## A6_Score         4.415e+01  1.195e+04   0.004    0.997
## A7_Score         4.438e+01  1.149e+04   0.004    0.997
## A8_Score         4.411e+01  1.260e+04   0.004    0.997
## A9_Score         4.430e+01  1.124e+04   0.004    0.997
## A10_Score        4.375e+01  1.210e+04   0.004    0.997
## age             -3.696e-03  5.095e+02   0.000    1.000
## gender          -2.874e-01  9.034e+03   0.000    1.000
## ethnicity       -2.413e-02  1.331e+03   0.000    1.000
## jundice         -1.475e-03  1.418e+04   0.000    1.000
## austim           3.481e-01  1.325e+04   0.000    1.000
## contry_of_res    5.635e-03  2.465e+02   0.000    1.000
## used_app_before  4.205e+01  1.232e+05   0.000    1.000
## result                 NA         NA      NA       NA
## age_desc               NA         NA      NA       NA
## relation         2.175e-01  3.871e+03   0.000    1.000
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6.3867e+02  on 563  degrees of freedom
## Residual deviance: 4.6969e-08  on 544  degrees of freedom
## AIC: 40
##
## Number of Fisher Scoring iterations: 25
```

Pedicting on the Dataset

```r
pred= predict(logistic, test, type='response')
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
pred
```

```
##            1            5            6            9           18           19
## 1.917506e-10 2.220446e-16 1.000000e+00 1.041436e-10 2.220446e-16 2.220446e-16
##           21           23           27           34           36           39
## 2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00 2.220446e-16 1.000000e+00
##           46           47           53           54           57           65
## 1.000000e+00 2.220446e-16 2.220446e-16 1.000000e+00 1.000000e+00 1.000000e+00
##           71           77           92           95           96          102
## 2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00 2.220446e-16 2.220446e-16
##          104          110          116          120          126          133
## 2.220446e-16 2.220446e-16 1.000000e+00 1.000000e+00 1.000000e+00 2.220446e-16
##          140          146          147          148          156          159
## 2.220446e-16 1.000000e+00 1.000000e+00 1.000000e+00 2.220446e-16 2.220446e-16
##          161          164          166          169          171          177
## 2.220446e-16 2.220446e-16 2.220446e-16 2.841435e-10 1.429846e-10 1.000000e+00
##          181          184          189          195          198          203
## 2.220446e-16 2.220446e-16 1.000000e+00 2.220446e-16 1.000000e+00 1.000000e+00
##          206          215          216          219          225          228
## 1.000000e+00 2.220446e-16 1.000000e+00 1.000000e+00 1.000000e+00 2.220446e-16
##          229          230          231          233          236          247
## 2.220446e-16 1.000000e+00 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16
##          249          255          263          264          289          297
## 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00
##          300          307          319          321          329          330
## 1.000000e+00 1.000000e+00 1.335244e-10 2.220446e-16 2.220446e-16 2.220446e-16
##          332          344          348          352          354          357
## 1.000000e+00 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16
##          360          373          374          385          390          398
## 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16
##          402          406          407          410          417          432
## 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00
##          436          439          441          442          444          448
## 2.220446e-16 2.220446e-16 2.556178e-10 2.220446e-16 1.000000e+00 2.220446e-16
##          450          456          466          471          476          485
## 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16
##          488          497          499          505          513          518
## 1.000000e+00 2.220446e-16 1.000000e+00 1.000000e+00 2.220446e-16 1.000000e+00
##          523          528          538          542          545          557
## 1.000000e+00 2.220446e-16 2.220446e-16 2.220446e-16 1.000000e+00 2.220446e-16
##          558          560          566          568          571          580
## 2.220446e-16 2.997553e-10 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16
##          587          592          594          595          611          615
## 1.000000e+00 3.985352e-10 2.220446e-16 2.220446e-16 2.220446e-16 2.220446e-16
##          638          640          646          670          671          673
## 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00 1.000000e+00
##          676          685          688          693          694          697
## 2.220446e-16 2.220446e-16 1.000000e+00 1.000000e+00 2.220446e-16 1.000000e+00
##          699          701
## 1.000000e+00 2.220446e-16
```

```
pred = ifelse(pred > 0.40, 1, 0)
pred= as.factor(pred)
```

Checking Accuracy with Confusion Matrix

```
matrix = confusionMatrix(pred,as.factor(test$Class.ASD))
matrix
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 94  0
##          1  0 46
##
##                Accuracy : 1
##                  95% CI : (0.974, 1)
##     No Information Rate : 0.6714
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.6714
##          Detection Rate : 0.6714
##    Detection Prevalence : 0.6714
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : 0
##
```

Accuracy is 100 percent while we use Logistic regression

We will now try the classification using Random Forest Classifiers

Random Forest Classifiers

Random forest is a machine learning algorithm that uses a collection of decision trees providing more flexibility, accuracy, and ease of access in the output. This algorithm dominates over decision trees algorithm as decision trees provide poor accuracy as compared to the random forest algorithm. In simple words, the random forest approach increases the performance of decision trees. It is one of the best algorithm as it can use both classification and regression techniques. Being a supervised learning algorithm, random forest uses the bagging method in decision trees and as a result, increases the accuracy of the learning model.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.3
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'

## The following object is masked from 'package:ggplot2':
##
##     margin

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
require(caTools)
```

```
## Loading required package: caTools

## Warning: package 'caTools' was built under R version 4.0.4
```

```
dim(data)
```

```
## [1] 704  22
```

```
dim(train)
```

```
## [1] 564  22
```

```
dim(test)
```

```
## [1] 140  22
```

```
summary(data)
```

```
##        X            A1_Score         A2_Score         A3_Score
##  Min.   :  0.0   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:175.8   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :351.5   Median :1.0000   Median :0.0000   Median :0.0000
##  Mean   :351.5   Mean   :0.7216   Mean   :0.4531   Mean   :0.4574
##  3rd Qu.:527.2   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :703.0   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##     A4_Score         A5_Score         A6_Score         A7_Score
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000
##  Median :0.0000   Median :0.0000   Median :0.0000   Median :0.0000
##  Mean   :0.4957   Mean   :0.4986   Mean   :0.2841   Mean   :0.4176
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :1.0000
##     A8_Score         A9_Score        A10_Score           age
##  Min.   :0.0000   Min.   :0.0000   Min.   :0.0000   Min.   : 0.0
##  1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.:0.0000   1st Qu.: 4.0
##  Median :1.0000   Median :0.0000   Median :1.0000   Median :10.0
##  Mean   :0.6491   Mean   :0.3239   Mean   :0.5739   Mean   :12.2
```

```
##  3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:1.0000   3rd Qu.:18.0
##  Max.   :1.0000   Max.   :1.0000   Max.   :1.0000   Max.   :45.0
##      gender         ethnicity         jundice            austim
##  Min.   :0.0000   Min.   : 0.000   Min.   :0.00000   Min.   :0.0000
##  1st Qu.:0.0000   1st Qu.: 1.000   1st Qu.:0.00000   1st Qu.:0.0000
##  Median :1.0000   Median : 5.000   Median :0.00000   Median :0.0000
##  Mean   :0.5213   Mean   : 5.305   Mean   :0.09801   Mean   :0.1293
##  3rd Qu.:1.0000   3rd Qu.:10.000   3rd Qu.:0.00000   3rd Qu.:0.0000
##  Max.   :1.0000   Max.   :11.000   Max.   :1.00000   Max.   :1.0000
##  contry_of_res   used_app_before       result          age_desc    relation
##  Min.   : 0.00   Min.   :0.00000   Min.   : 0.000   Min.   :0   Min.   :0.000
##  1st Qu.:29.00   1st Qu.:0.00000   1st Qu.: 3.000   1st Qu.:0   1st Qu.:4.000
##  Median :43.00   Median :0.00000   Median : 4.000   Median :0   Median :5.000
##  Mean   :44.05   Mean   :0.01705   Mean   : 4.875   Mean   :0   Mean   :4.099
##  3rd Qu.:63.00   3rd Qu.:0.00000   3rd Qu.: 7.000   3rd Qu.:0   3rd Qu.:5.000
##  Max.   :66.00   Max.   :1.00000   Max.   :10.000   Max.   :0   Max.   :5.000
##    Class.ASD
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.2685
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

```
library(randomForest)
library(mlbench)
```

```
## Warning: package 'mlbench' was built under R version 4.0.5
```

```
library(caret)

data$Class.ASD =as.factor(data$Class.ASD)

dataset <- data
x <- data[,1:8]
y <- data["Class.ASD"]
```

```
# Create model with default paramters
control <- trainControl(method="repeatedcv", number=10, repeats=3)
seed <- 7
#metric <- "Accuracy"
set.seed(seed)
mtry <- sqrt(ncol(x))
tunegrid <- expand.grid(.mtry=mtry)
rf_default <- train(Class.ASD~., data=data, method="rf", tuneGrid=tunegrid, trControl=control)
print(rf_default)
```

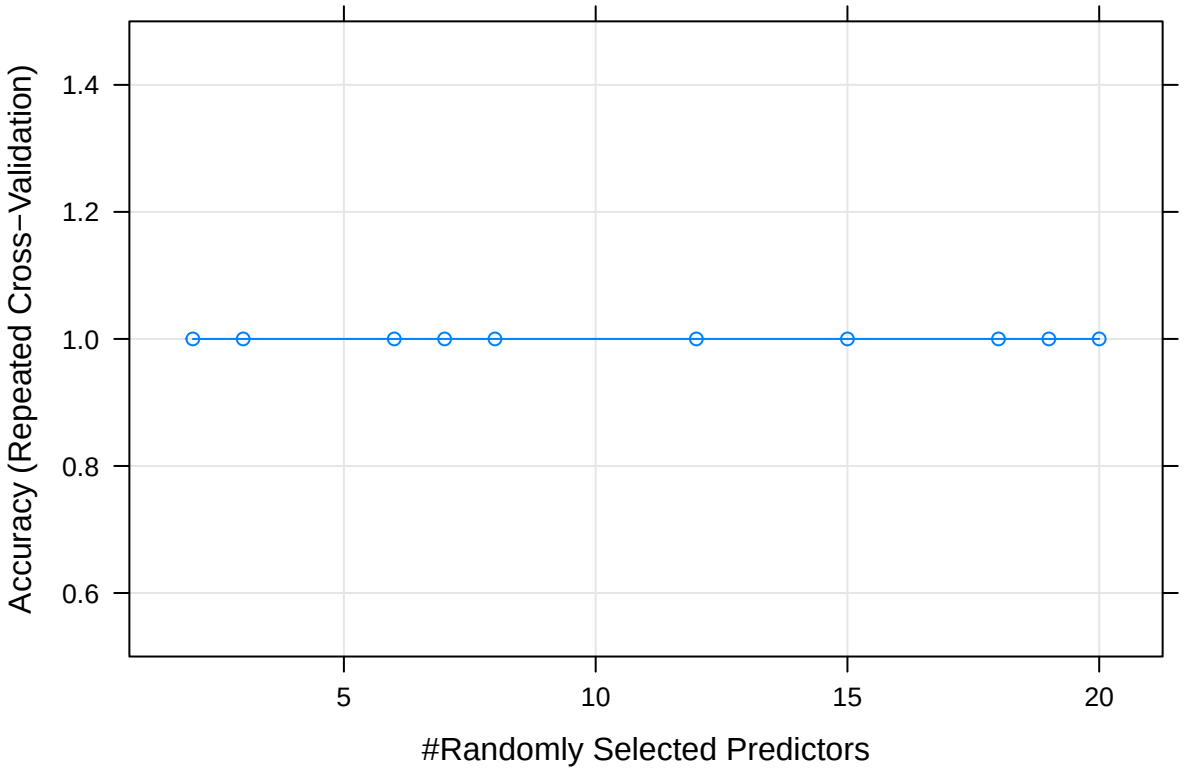
```
## Random Forest
##
## 704 samples
##  21 predictor
##   2 classes: '0', '1'
```

```
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 633, 633, 633, 634, 633, 634, ...
## Resampling results:
##
##   Accuracy  Kappa
##   1         1
##
## Tuning parameter 'mtry' was held constant at a value of 2.828427
```

```
# Random Search
control <- trainControl(method="repeatedcv", number=10, repeats=3, search="random")
set.seed(seed)
mtry <- sqrt(ncol(x))
rf_random <- train(Class.ASD~., data=dataset, method="rf", tuneLength=15, trControl=control)
print(rf_random)
```

```
## Random Forest
##
## 704 samples
##  21 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 633, 633, 633, 634, 633, 634, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy  Kappa
##    2    1         1
##    3    1         1
##    6    1         1
##    7    1         1
##    8    1         1
##   12    1         1
##   15    1         1
##   18    1         1
##   19    1         1
##   20    1         1
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
plot(rf_random)
```


Accuracy (Repeated Cross−Validation)
1.4
1.2
1.0
0.8
0.6
5
10
15
20
#Randomly Selected Predictors