---

title: “Data Visualization in R”
subtitle: ‘Mahedi Hasan, PhD Candidate in Statistical Science, WSU’
author: “Email: mdmahedi.hasan@wsu.edu”
date: “February 27, 2023”
output:
pdf_document: default
word_document: default

Data Visualization

Common Graphical Techniques

- Bar Chart
- Pie Chart
- Histogram
- Line Chart
- Scatter Plot etc.

R Package and its Installation

- A Package is basically the compilation of a set of codes, data and instructions
- R has more than 17,000 packages (ref: r-cran) by now and adding…
- Packages are developed by the R users
- Different packages compiled different methods and has uses accordingly
- I assume by now you know how to install an R package, however; there are multiple ways to do that and one easy way is to write: `install.packages("Name of the package")` on R console or script and run.

Recall some Learning

- R is case sensitive
- R is an interpreted language

How to get the Data

- let’s get introduced to the data set that we will be using today! For convenience, we are using data available in R under a package called `ISLR`. However, we all know that we can do the same analysis and graphing using the data from other sources as well, including our computer hard drive.

To begin, first, we need to install the `ISLR` package to get access to it. I assume you have already installed the package.

- now, running the following code, we will see a couple of datasets stored under this `ISLR` package. After installing the package, you will have access to any of the data sets just by typing `data("name of the dataset")`.
- one of the data sets under this package is called, ‘`Credit` and we will work with this data set.

Installing the required package and get the dataset

```
#install.packages("ISLR")

library(ISLR)
```

- After installing any package in R, you should comment on this line (by giving a # at the beginning) because you need to install the package just once, but you will call the package (by library function) every time you work fresh with that package.

Let see what are dataset available in ISLR package

```
data(package = "ISLR")
```

Calling a dataset into R

```
data("Credit")
```

To see the list of the variables in the dataset

```
names(Credit)
```

```
##  [1] "ID"        "Income"    "Limit"     "Rating"    "Cards"     "Age"
##  [7] "Education" "Gender"    "Student"   "Married"   "Ethnicity" "Balance"
```

To see the types of the variables

```
str(Credit)
```

```
## 'data.frame':    400 obs. of  12 variables:
##  $ ID       : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Income   : num  14.9 106 104.6 148.9 55.9 ...
##  $ Limit    : int  3606 6645 7075 9504 4897 8047 3388 7114 3300 6819 ...
##  $ Rating   : int  283 483 514 681 357 569 259 512 266 491 ...
##  $ Cards    : int  2 3 4 3 2 4 2 2 5 3 ...
##  $ Age      : int  34 82 71 36 68 77 37 87 66 41 ...
##  $ Education: int  11 15 11 11 16 10 12 9 13 19 ...
##  $ Gender   : Factor w/ 2 levels " Male","Female": 1 2 1 2 1 1 2 1 2 2 ...
##  $ Student  : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 1 1 1 1 2 ...
##  $ Married  : Factor w/ 2 levels "No","Yes": 2 2 1 1 2 1 1 1 1 2 ...
##  $ Ethnicity: Factor w/ 3 levels "African American",..: 3 2 2 2 3 3 1 2 3 1 ...
##  $ Balance  : int  333 903 580 964 331 1151 203 872 279 1350 ...
```

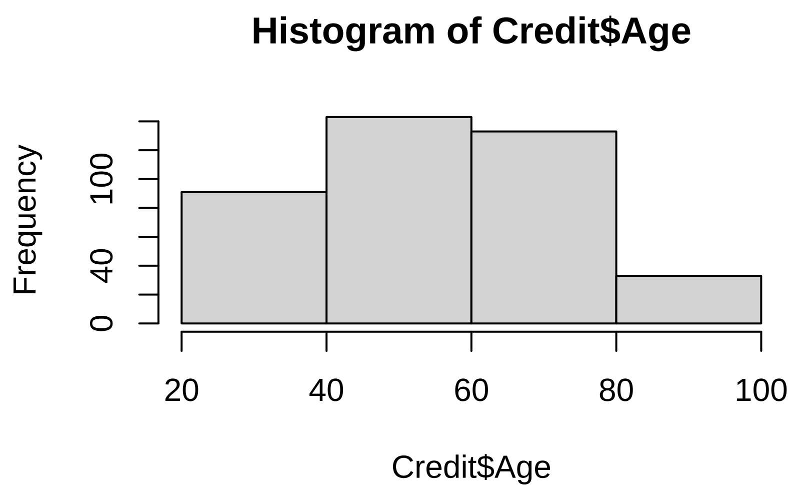
```
So, by now we are familier about the dataset and its variables!
```

Lets first create some common graphs in basic R. We will then move on to our main tool/approach of graphing.

Histogram

- Commonly used to show the `frequency distribution` of data
- Applicable to the numerical data, e.g. Age, Income, Temperature etc.

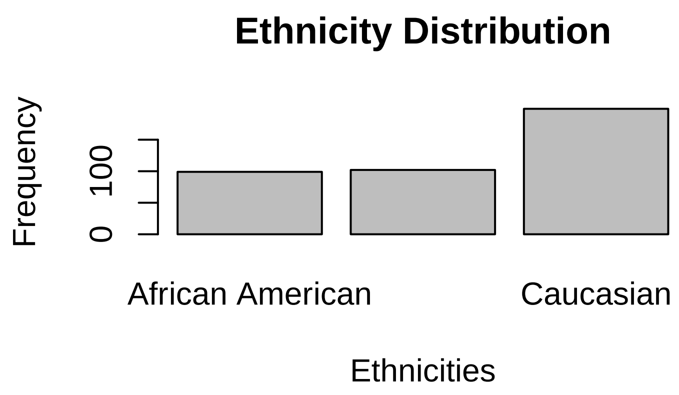
```r
hist(Credit$Age, breaks = 4)
```



Bar Chart

- Bar chart is useful to show the frequency information of categorical variable by the bars
- Commonly used to the variable such as gender, ethnicity, groups etc.

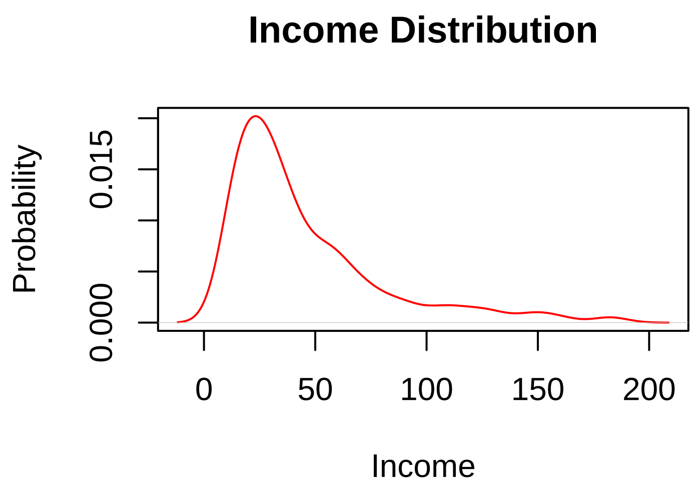
```r
counts = table(Credit$Ethnicity)
barplot(counts, main = "Ethnicity Distribution", xlab = "Ethnicities", ylab = "Frequency" )
```



Density Plot

- Useful to represent the distribution of a numerical variable.
- It shows the probability density function of the variable.

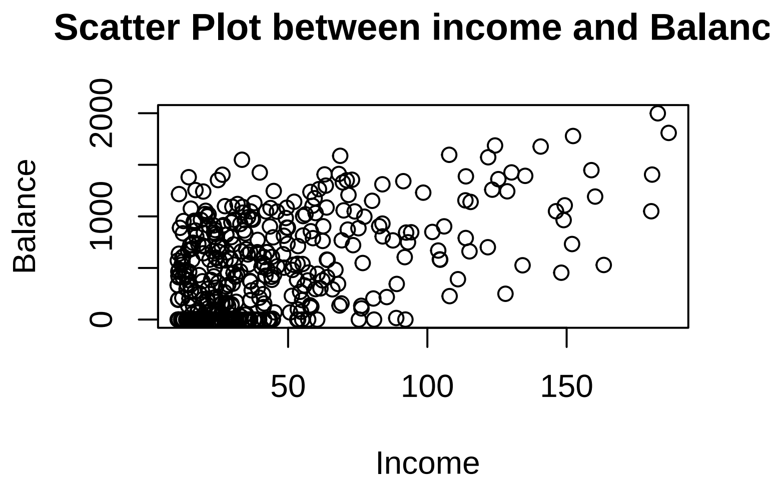
```
d = density(Credit$Income)
plot(d, col= "red", main = "Income Distribution", xlab = "Income", ylab = "Probability")
```



Scatter Plot

- It shows the graphical relationship between two numerical variables (e.g. Income vs Age)
- Horizontal (x) axis represents one variable, and the vertical axis (y) represents the other variable.

```
plot(Credit$Income, Credit$Balance, main = "Scatter Plot between income and Balance", xlab = " Income",
```



We have seen how to create the common graphs in the basic version of R. Now we will see how we can create those graphs in R using a very popular R package called `ggplot2`

A Few Facts about ggplot2 R package

- One of the most popular R package for creating graph
- Hadley Wickham created ggplot2 in 2005
- ggplot2 follows the `grammar of graphics`
- Any plot is composed of the data and mapping, and the data's variable that are mapped to aesthetic attributes depends on the following parameters:
  - A Layer
  - Scales
  - Coord
  - Facet
  - Theme
- All these attributes are addressed very addroitly in ggplot2

Installing the "ggplot2" R package

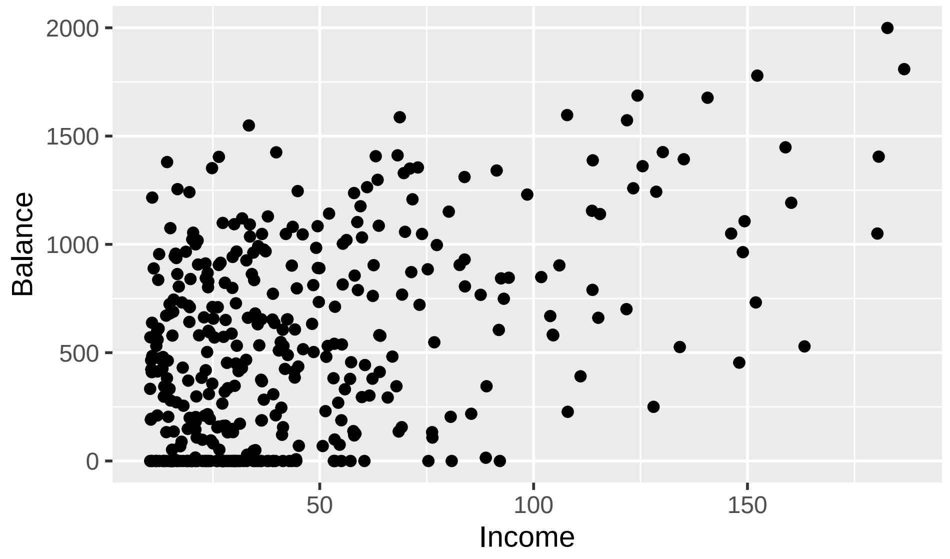
```
#install.packages("ggplot2")
library(ggplot2)
```

We plan to work with one graph in detail to follow the same procedure for other charts as well. Let's work with a scatter plot. We will be using the same data set name, `Credit` under the `ISLR` package.

Lets make a basic Scatter Plot using ggplot2

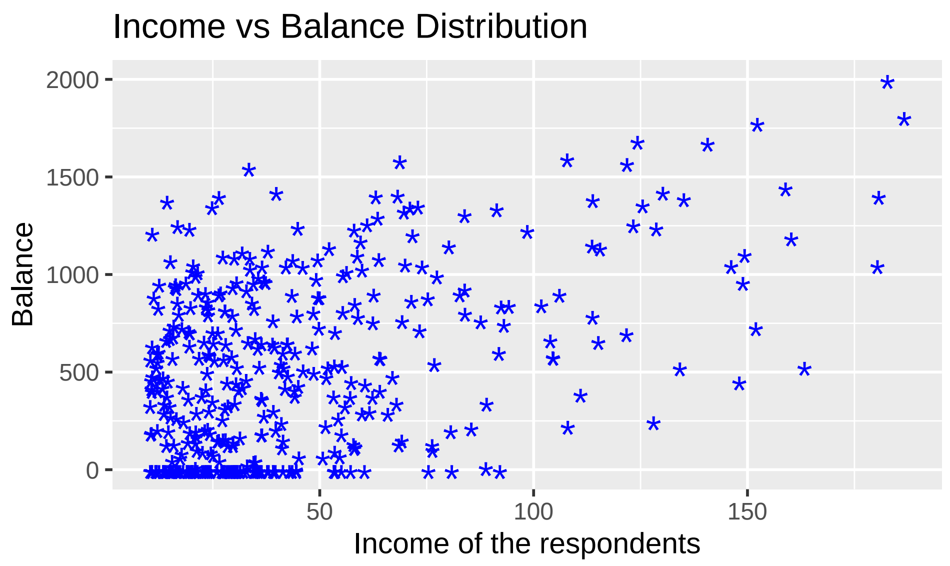
- `data` is the place where we put the name of dataset we are working on, (e.g. here, `Credit`)
- `aes` in ggplot means `something you can see`, e.g. x & y asis.
- ‘`geom_point` is used to create the scatter plots

```
ggplot(data = Credit, aes(x = Income, y = Balance))+
  geom_point()
```



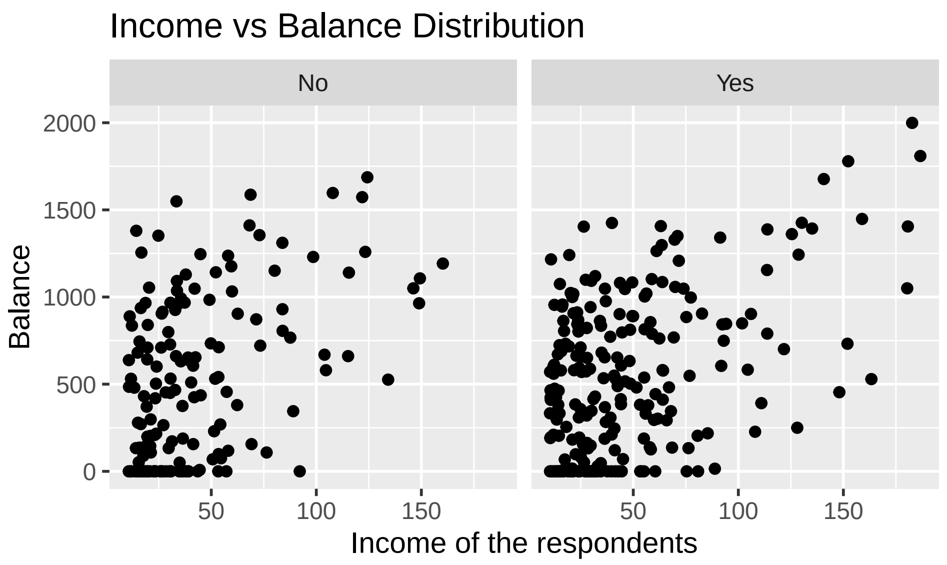
Lets label the graph and do some experiments with different parameters

```
ggplot(data = Credit, aes(x = Income, y = Balance))+
  geom_point(size = 5, color = "blue", shape = "*")+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")
```



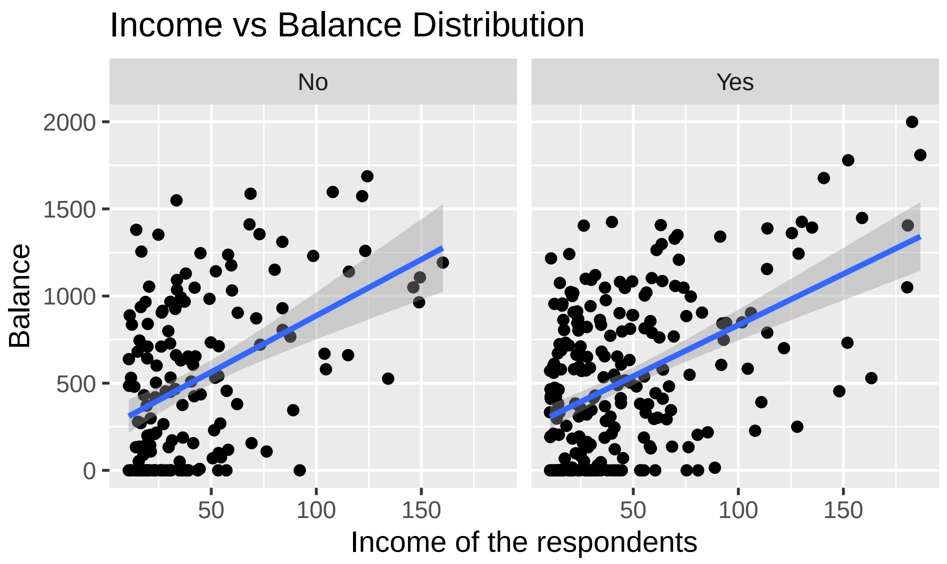
Lets now add a facet on the graph

```
ggplot(data = Credit, aes(x = Income, y = Balance))+
  geom_point()+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")+
  facet_grid(.~Married)
```



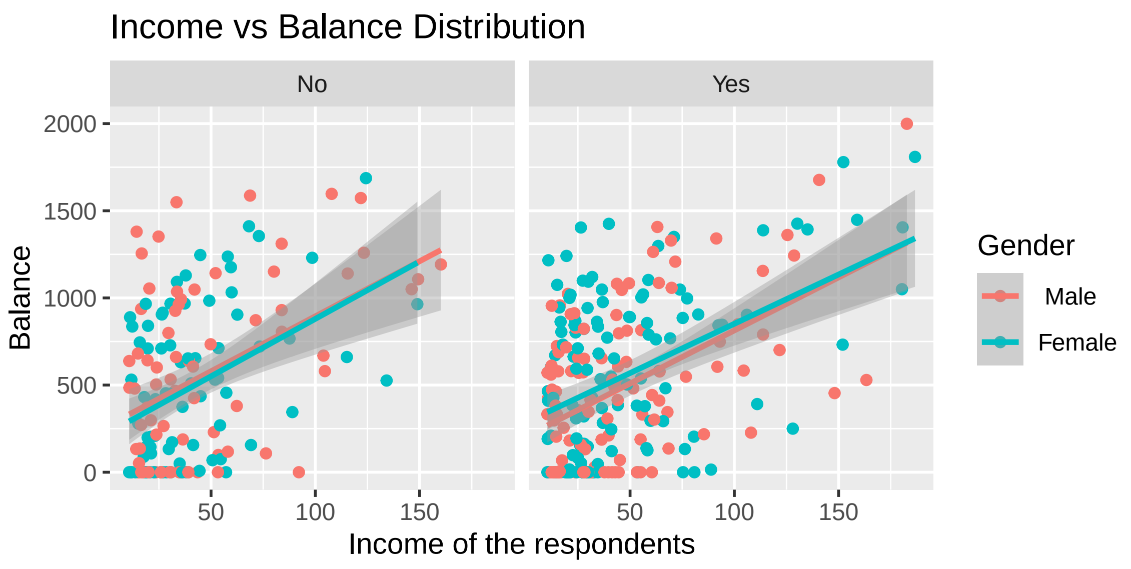
Can we fit a linear line on the scatter plots?

```
ggplot(data = Credit, aes(x = Income, y = Balance))+
  geom_point()+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")+
  facet_grid(.~Married) +
  geom_smooth(method = "lm")
```



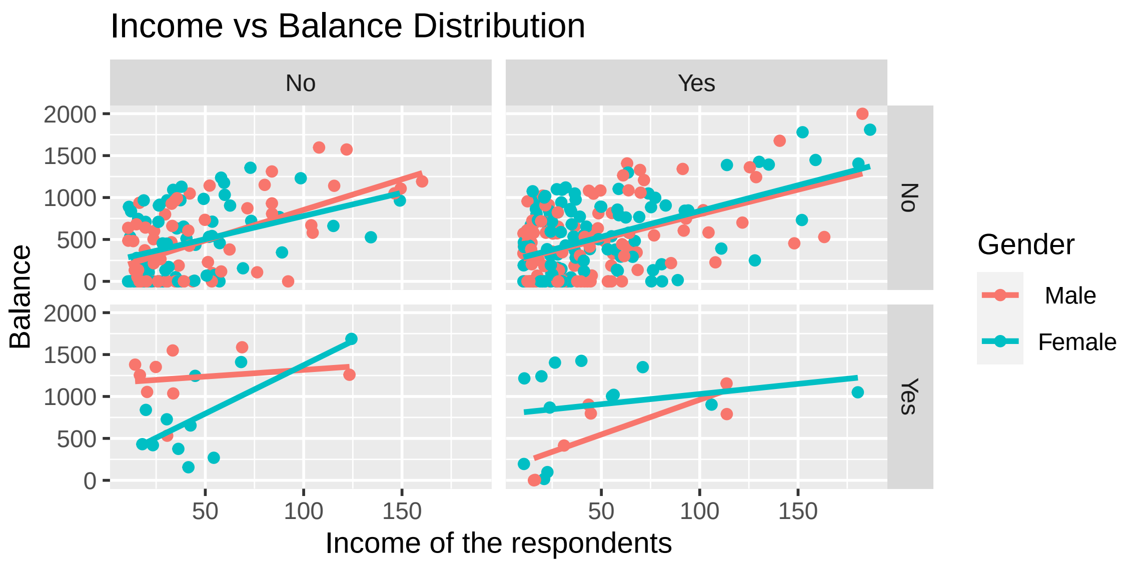
Can we also see the relationship by a third variable, let say Gender?

```r
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")+
  facet_grid(.~Married) +
  geom_smooth(method = "lm")
```



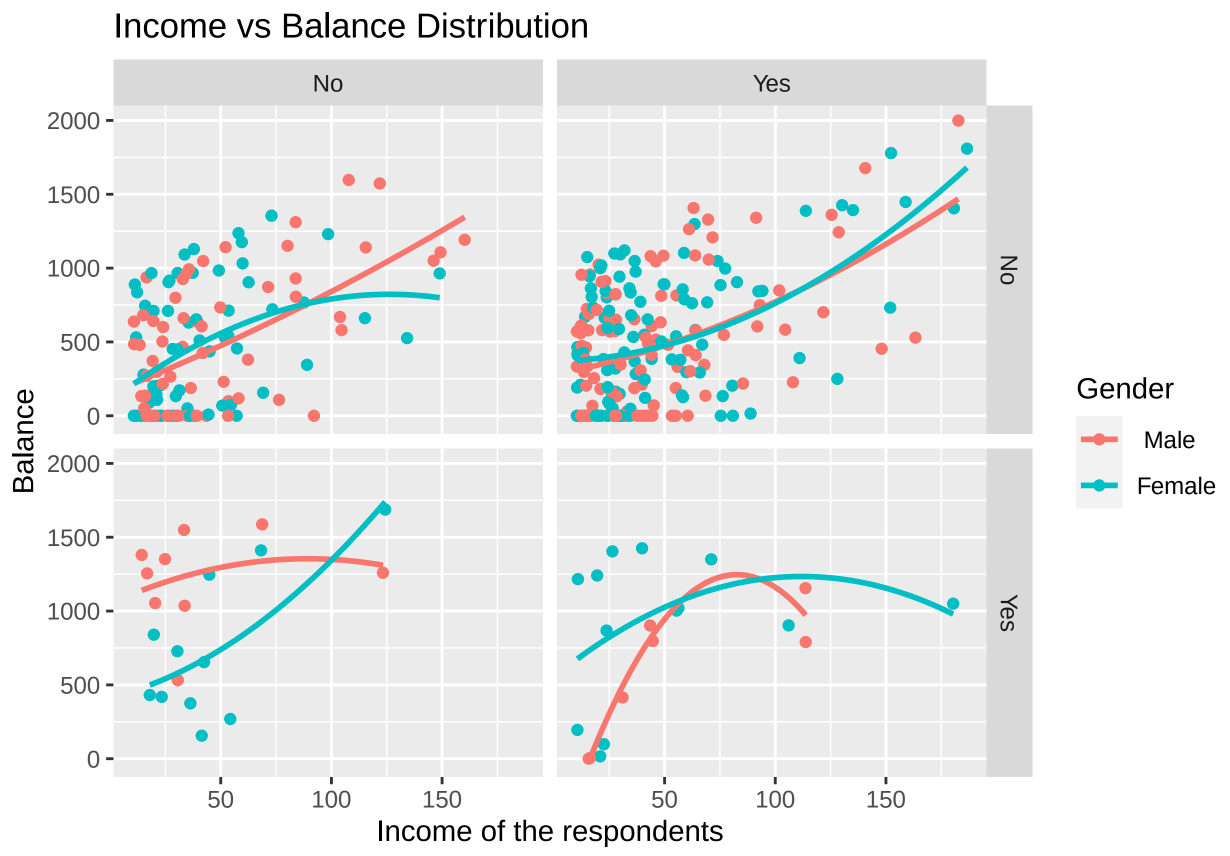
Lets add another factor in this relationship and see what happens!

```r
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")+
  facet_grid(Student~Married) +
  geom_smooth(method = "lm", se= F)
```



How about introducing polynomials in the relationship but be careful about the polynomial order!!!

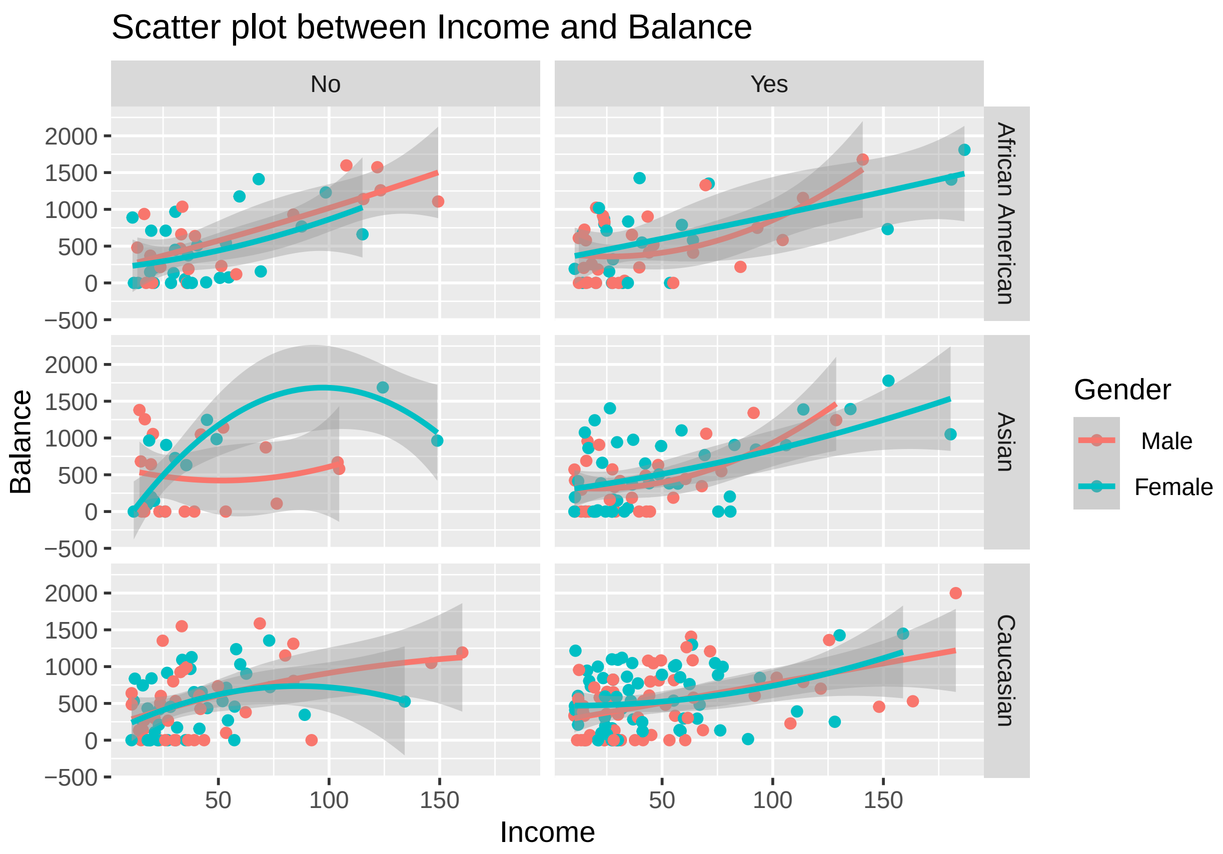
```
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")+
  facet_grid(Student~Married) +
  geom_smooth(method = "lm", formula = y ~ x + (poly(x, 2)-1), se = F)
```



Another Way to do Polynomial regression

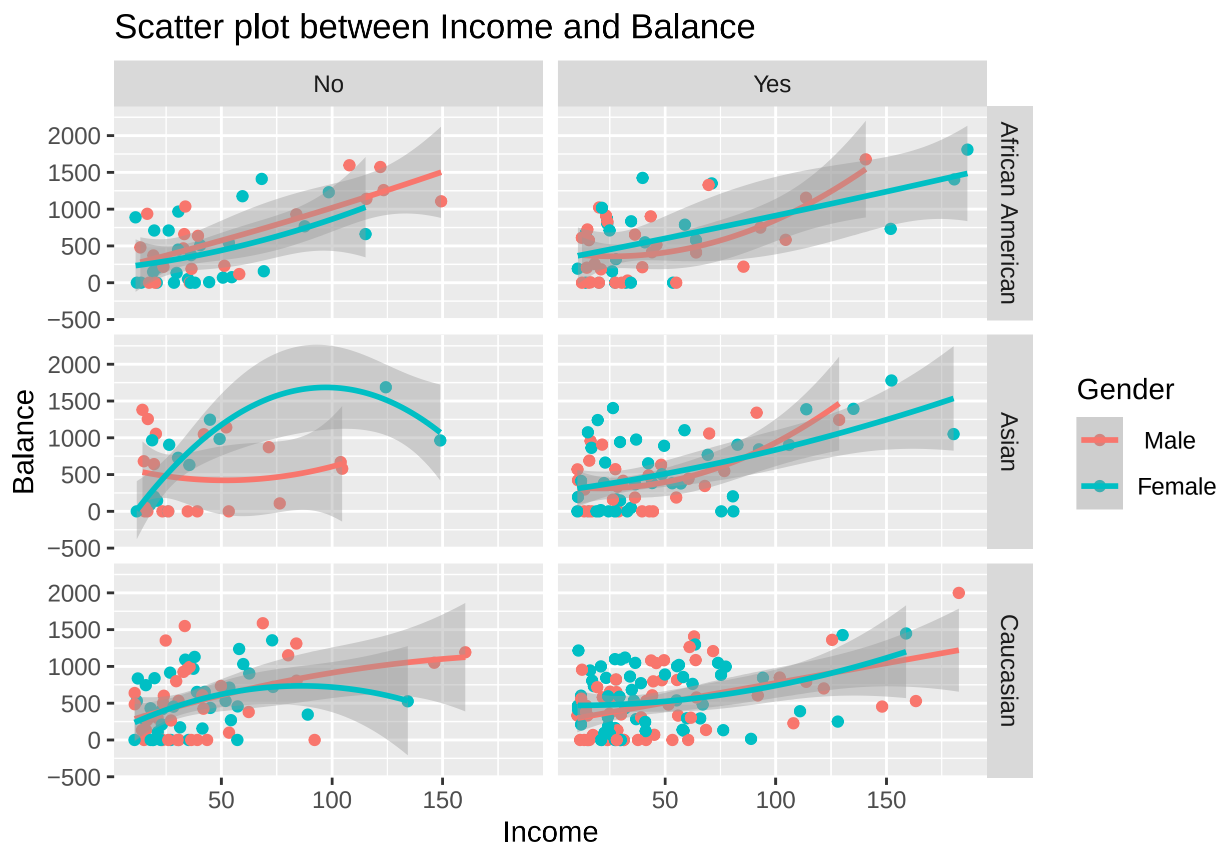
```r
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Scatter plot between Income and Balance", x = "Income", y = "Balance")+
  facet_grid(Ethnicity~ Married)+
  geom_smooth(method = "lm", formula = y ~ x + I(x^2), size = 1)
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
```



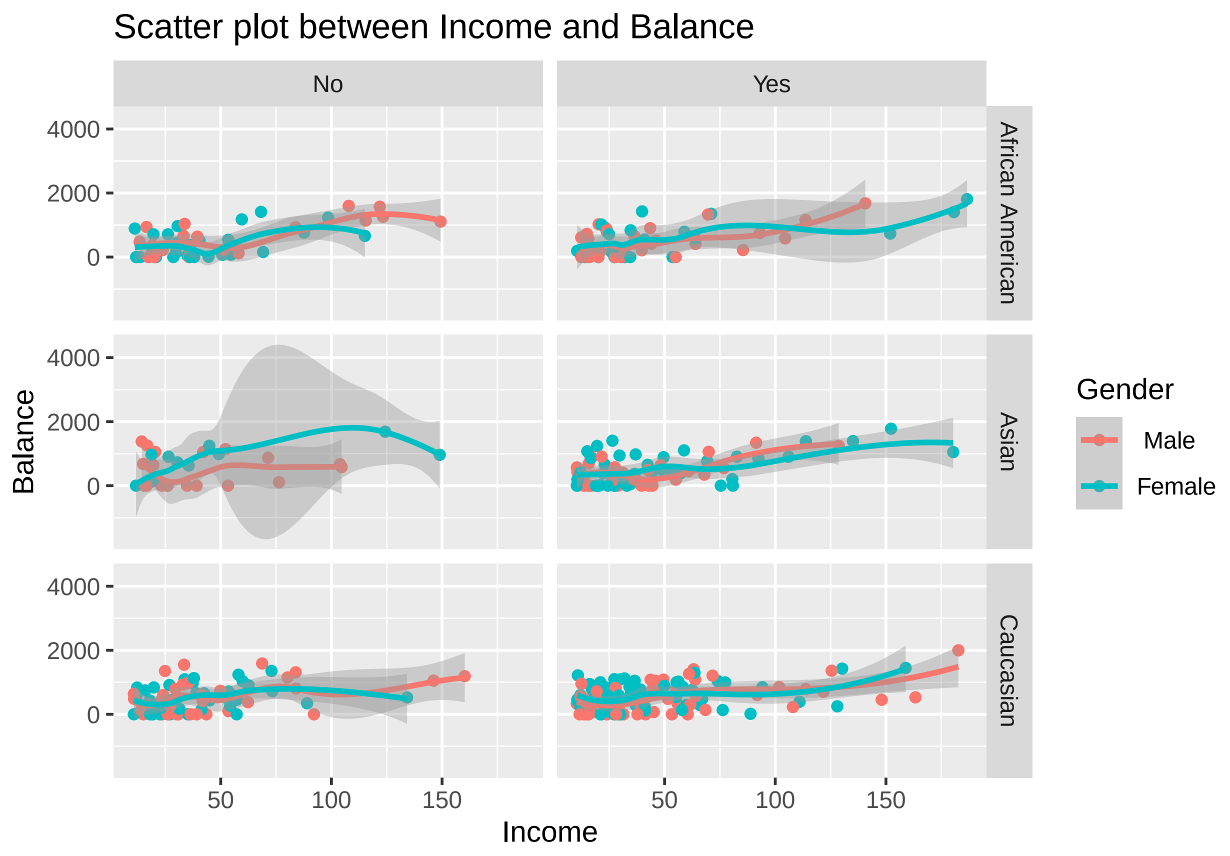
Polynoial regression using poly function

```
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Scatter plot between Income and Balance", x = "Income", y = "Balance")+
  facet_grid(Ethnicity~ Married)+
  geom_smooth(method = "lm", formula = y ~  poly(x, 2), size = 1)
```



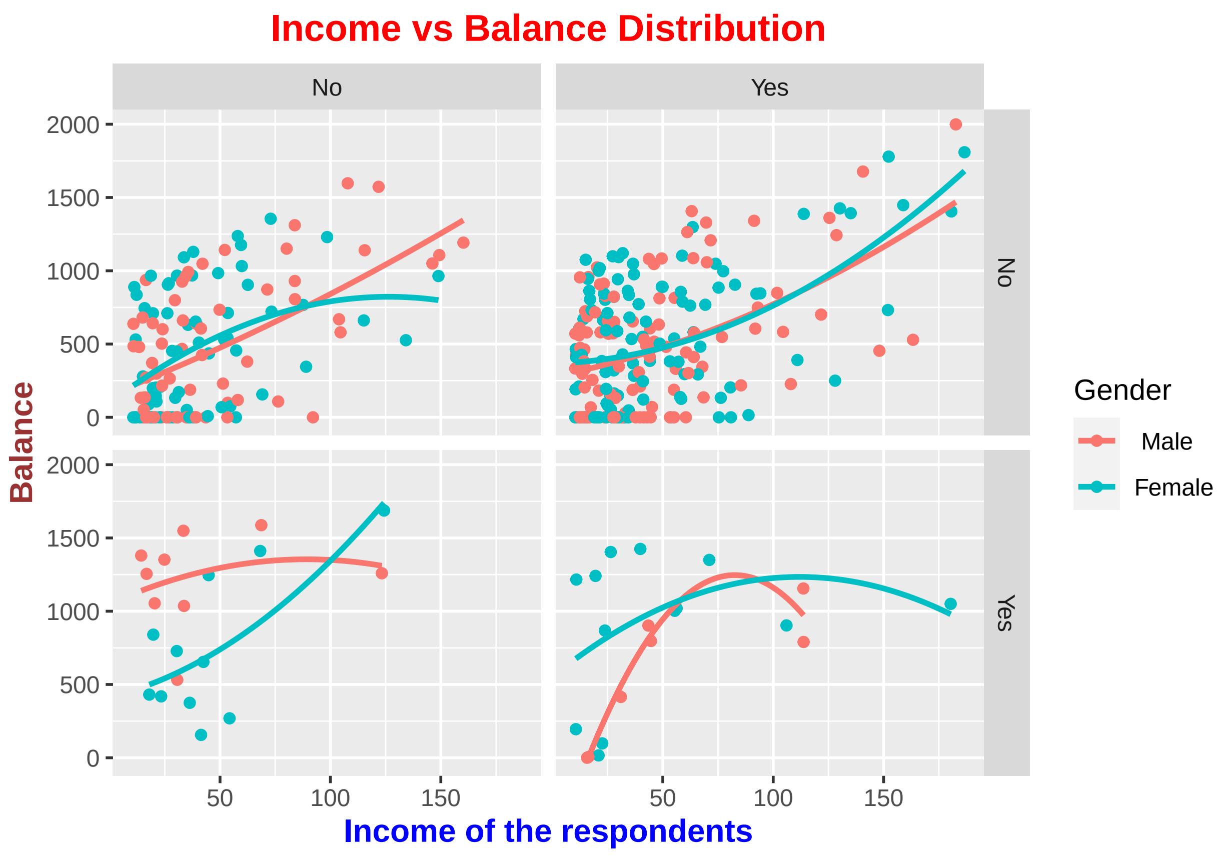
Loess regression

```
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Scatter plot between Income and Balance", x = "Income", y = "Balance")+
  facet_grid(Ethnicity~ Married)+
  geom_smooth(method = "loess", formula = y~x, size = 1)
```



Finally, lets work with some theme on the graph

```
ggplot(data = Credit, aes(x = Income, y = Balance, color = Gender))+
  geom_point()+
  labs(title = "Income vs Balance Distribution", x = "Income of the respondents", y = "Balance")+
  facet_grid(Student~Married) +
  geom_smooth(method = "lm", formula = y ~ x + (poly(x, 2)-1), se = F)+
  theme(
    plot.title = element_text(color="red", size=14, face="bold", hjust = 0.5),
    axis.title.x = element_text(color="blue", size=12, face="bold"),
    axis.title.y = element_text(color="#993333", size=12, face="bold")
  )
```

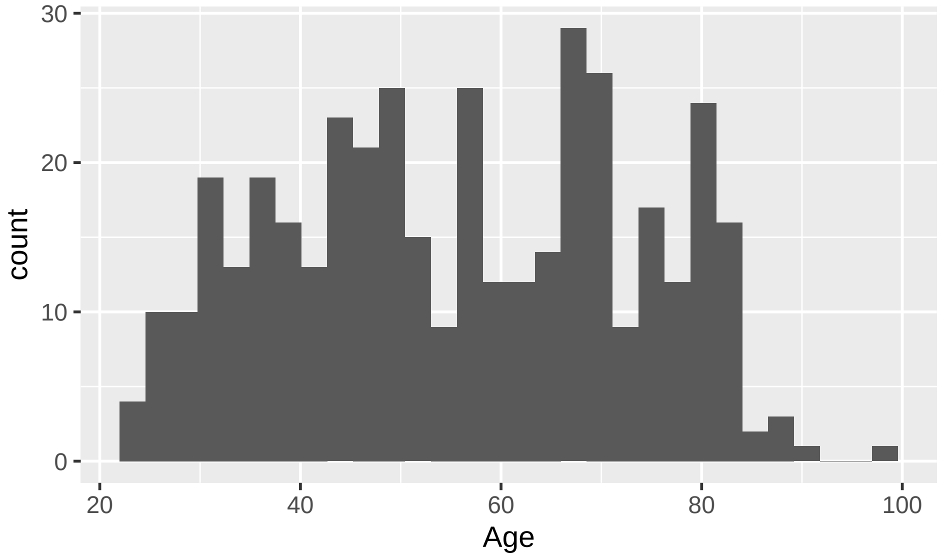


Few Others Common Graphs

Histogram 1

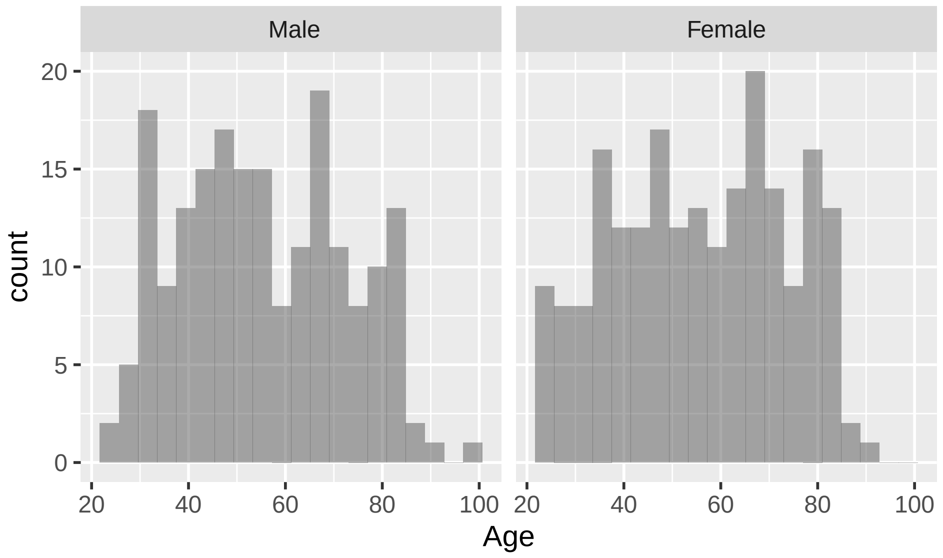
```r
ggplot(data = Credit, aes(x = Age)) +
  geom_histogram()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



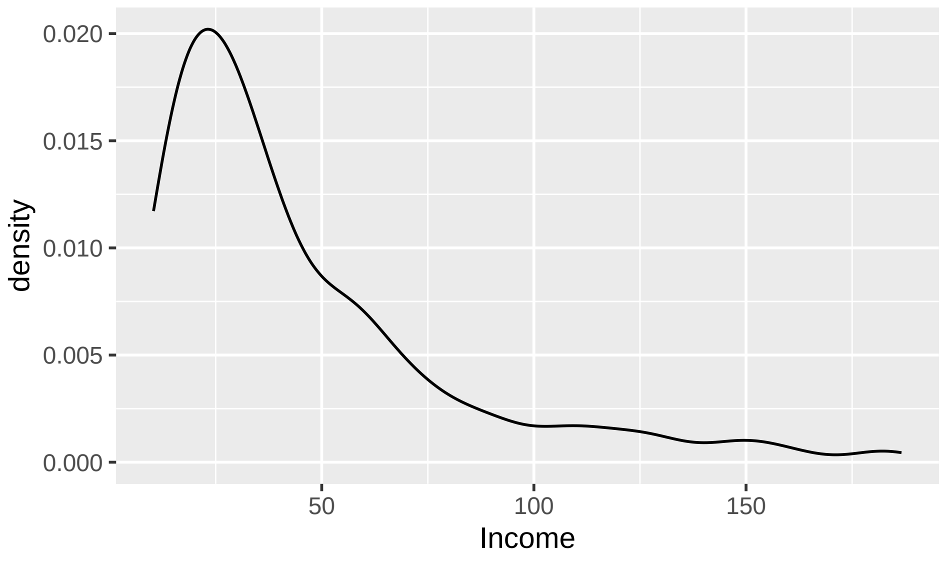
Histogram 2

```r
ggplot(data = Credit, aes(x = Age)) +
  geom_histogram(bins = 20, alpha = 0.5)+
  facet_grid(.~Gender)
```



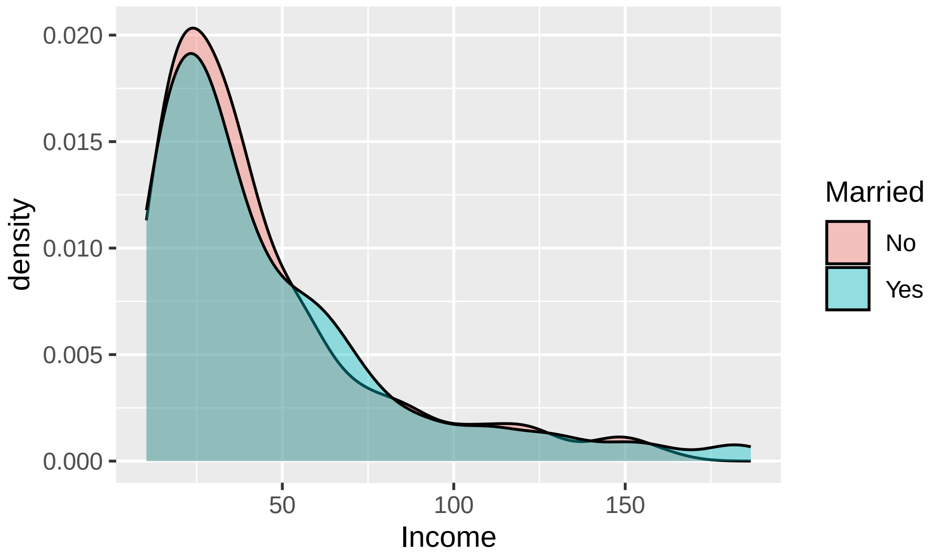
Density Plot1

```r
data("Credit")
ggplot(data = Credit, aes(x = Income)) +
  geom_density()
```



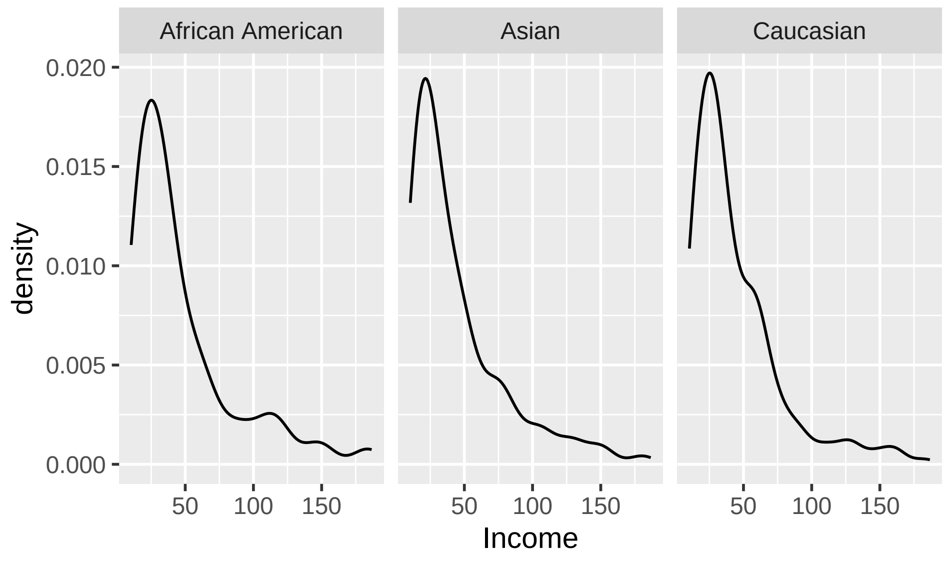
Density Plot2

```r
ggplot(data = Credit, aes(x = Income, fill = Married)) +
  geom_density(alpha = 0.4)
```



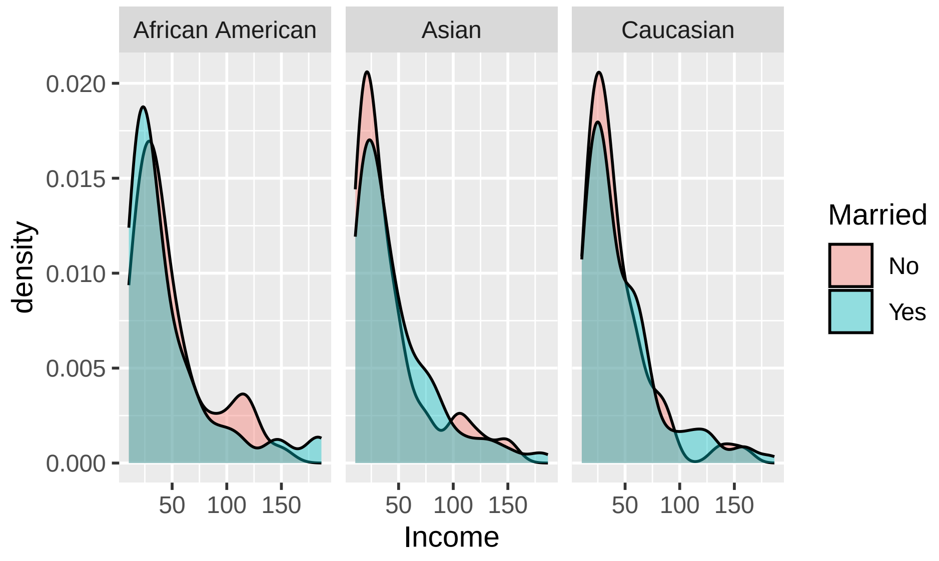
Density Plot3

```
ggplot(data = Credit, aes(x = Income)) +
  geom_density()+
  facet_grid(.~ Ethnicity)
```



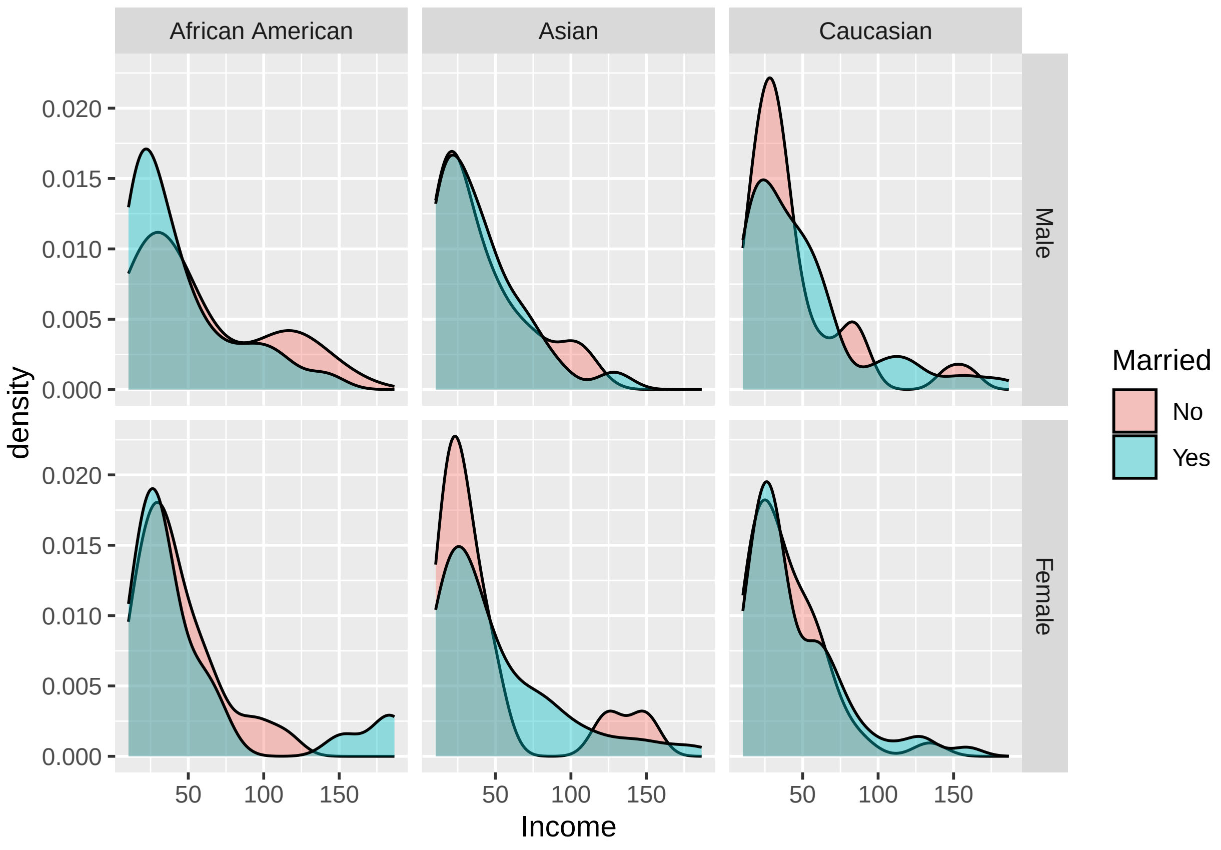
Density Plot4

```
ggplot(data = Credit, aes(x = Income, fill = Married)) +
  geom_density(alpha = 0.4)+
  facet_grid(.~ Ethnicity)
```



Density Plot5

```
ggplot(data = Credit, aes(x = Income, fill = Married)) +
  geom_density(alpha = 0.4)+
  facet_grid(Gender~ Ethnicity)
```



Data Analysis

We are done with the first part of this session; the graphing with ggplot2. Now lets move on to the data analysis part. We will be dealing with `Simple Linear Regression (SLR) model` and `Multiple Linear Regression (MLR) model` in this part.

Our objective here is to:

- Briefly introduce the concept of simple and multiple linear regression models and their mathematical form
- How we can write the SLR and MLR model in R and get the results from the model
- How to interpret the model findings
- Some diagnostics of the model

NB: There are four assumptions associated with a linear regression model:

1. Linearity: The relationship between X and the mean of Y is linear.
2. Homoscedasticity: The variance of residual is the same for any value of X.
3. Independence: Observations are independent of each other.
4. Normality: For any fixed value of X, Y is normally distributed.

Simple Linear Regression Model

A simple linear regression model(i.e just with one predictor) can be written as:

$$y_i = \beta_0 + \beta_1 x_1 + \epsilon_i$$

Multiple Linear Regression Model

A multiple linear regression model with n independent variables can be written as:

$$y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n + \epsilon_i$$

Here, y is our dependent variable and $x's$ are the independent variables.

Specification of the model

- Dependent varialbe (response): Balance
- Independent variables (predictors): Income, Age, Education etc.

Data

In this section we will also use the same dataset, named, `Credit` under ‘`ISLR` package as we have used before for creating graphs.

For this case again, lets recall the name of the variables that we have in our dataset.

```r
data("Credit")
names(Credit)
```

```
##  [1] "ID"        "Income"    "Limit"     "Rating"    "Cards"     "Age"
##  [7] "Education" "Gender"    "Student"   "Married"   "Ethnicity" "Balance"
```

```r
#head(Credit)
```

Descriptive Statistics

Before we dive into the regression modelling in R, lets get some descriptive statistics from our dataset.

```r
summary(Credit)
```

```
##        ID            Income           Limit           Rating     
##  Min.   :  1.0   Min.   : 10.35   Min.   :  855   Min.   : 93.0  
##  1st Qu.:100.8   1st Qu.: 21.01   1st Qu.: 3088   1st Qu.:247.2  
##  Median :200.5   Median : 33.12   Median : 4622   Median :344.0  
##  Mean   :200.5   Mean   : 45.22   Mean   : 4736   Mean   :354.9  
##  3rd Qu.:300.2   3rd Qu.: 57.47   3rd Qu.: 5873   3rd Qu.:437.2  
##  Max.   :400.0   Max.   :186.63   Max.   :13913   Max.   :982.0  
##      Cards            Age          Education        Gender    Student  
##  Min.   :1.000   Min.   :23.00   Min.   : 5.00    Male :193   No :360  
##  1st Qu.:2.000   1st Qu.:41.75   1st Qu.:11.00   Female:207   Yes: 40  
##  Median :3.000   Median :56.00   Median :14.00                         
##  Mean   :2.958   Mean   :55.67   Mean   :13.45                         
##  3rd Qu.:4.000   3rd Qu.:70.00   3rd Qu.:16.00                         
##  Max.   :9.000   Max.   :98.00   Max.   :20.00                         
##  Married              Ethnicity      Balance       
##  No :155   African American: 99   Min.   :   0.00  
##  Yes:245   Asian           :102   1st Qu.:  68.75  
##            Caucasian       :199   Median : 459.50  
##                                   Mean   : 520.01  
##                                   3rd Qu.: 863.00  
##                                   Max.   :1999.00  
```

Correlation Analysis

Correlation analysis to know how some variables are related.

```r
data_corr = Credit[, c(2:7, 12)]
names(data_corr)
```

```
## [1] "Income"    "Limit"     "Rating"    "Cards"     "Age"       "Education"
## [7] "Balance"
```

```r
# Now get the correlation coefficients
round(cor(data_corr), 3)
```

```
##           Income  Limit Rating  Cards   Age Education Balance
## Income     1.000  0.792  0.791 -0.018 0.175    -0.028   0.464
## Limit      0.792  1.000  0.997  0.010 0.101    -0.024   0.862
## Rating     0.791  0.997  1.000  0.053 0.103    -0.030   0.864
## Cards     -0.018  0.010  0.053  1.000 0.043    -0.051   0.086
## Age        0.175  0.101  0.103  0.043 1.000     0.004   0.002
## Education -0.028 -0.024 -0.030 -0.051 0.004     1.000  -0.008
## Balance    0.464  0.862  0.864  0.086 0.002    -0.008   1.000
```

Fit the model in R

```r
data("Credit")
# Fit simple linear regression model
fit1 = lm(Balance ~ Income, data = Credit)
summary(fit1)
```

```
##
## Call:
## lm(formula = Balance ~ Income, data = Credit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -803.64 -348.99  -54.42  331.75 1100.25
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 246.5148    33.1993   7.425  6.9e-13 ***
## Income        6.0484     0.5794  10.440  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 407.9 on 398 degrees of freedom
## Multiple R-squared:  0.215,  Adjusted R-squared:  0.213
## F-statistic:   109 on 1 and 398 DF,  p-value: < 2.2e-16
```

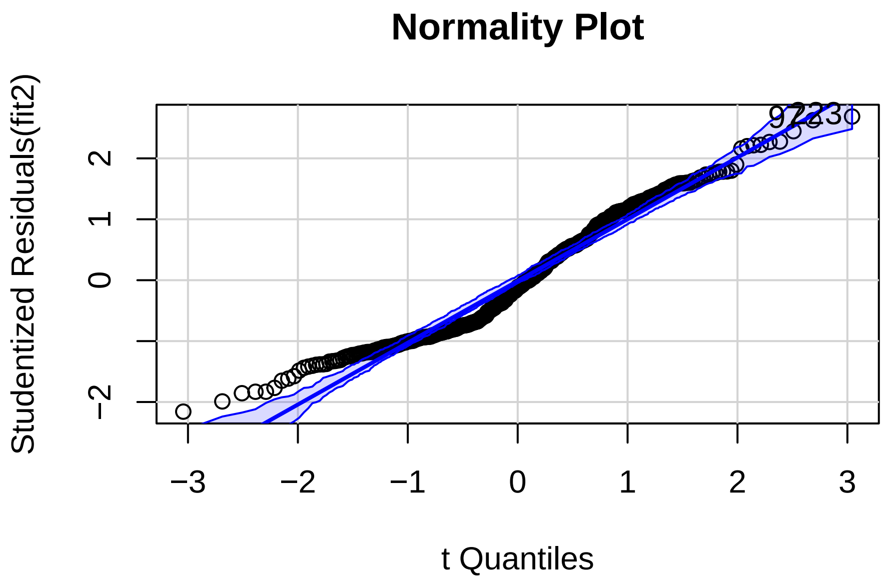
```r
data("Credit")
# Fit multiple linear regression model
fit2 = lm(Balance ~ Income + Age + Education, data = Credit)
summary(fit2)
```

```
##
## Call:
## lm(formula = Balance ~ Income + Age + Education, data = Credit)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -867.14 -343.14  -49.44  316.55 1080.56
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 348.8115   112.6895   3.095  0.00211 **
## Income        6.2380     0.5877  10.614  < 2e-16 ***
## Age          -2.1863     1.2004  -1.821  0.06930 .
## Education     0.8058     6.5254   0.123  0.90179
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 407.2 on 396 degrees of freedom
## Multiple R-squared:  0.2215, Adjusted R-squared:  0.2156
## F-statistic: 37.56 on 3 and 396 DF,  p-value: < 2.2e-16
```

```r
# data("Credit")
# # Fit multiple linear regression model
# fit3 = lm(Balance ~ ., data = Credit)
# summary(fit3)
```

qq plot

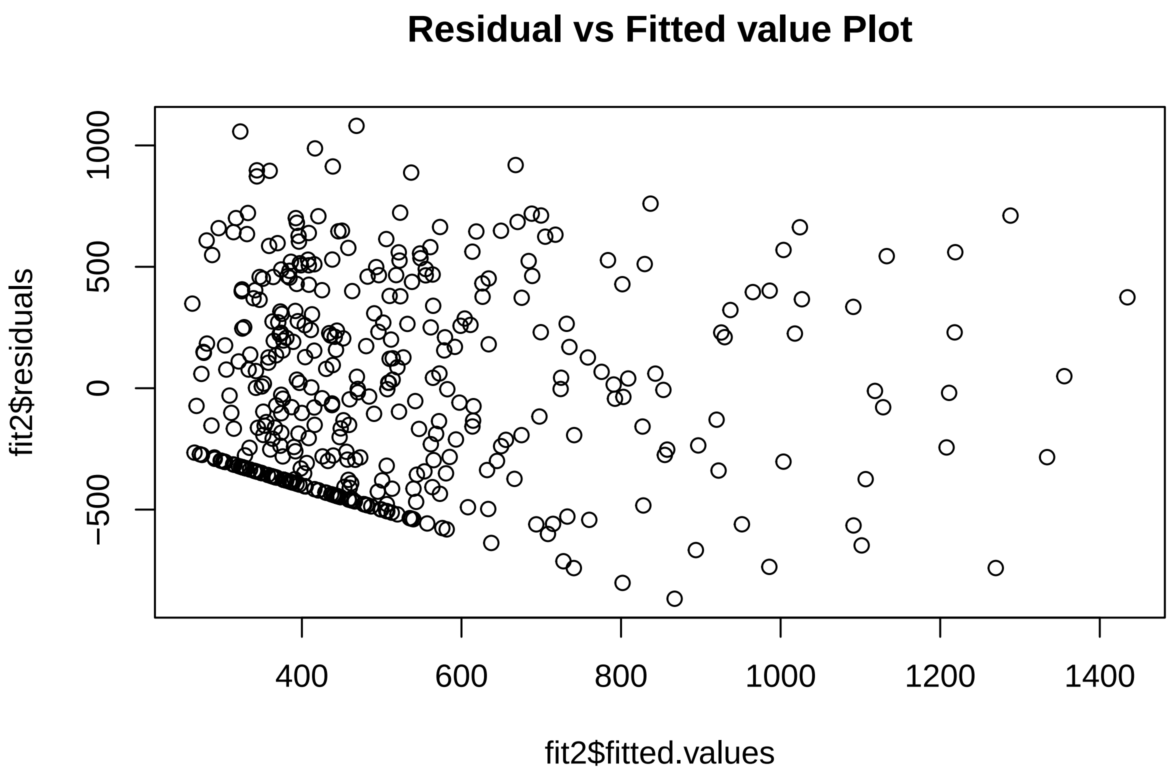
```r
library(car)
qqPlot(fit2, main = "Normality Plot")
```



```
## [1]  97 223
```

Residual vs Fitted value Plot

```r
plot(fit2$fitted.values, fit2$residuals,main = "Residual vs Fitted value Plot")
```



Resources for further Learning

- For Graphing in R
  - https://www.r-graph-gallery.com
- For ggplot2
  - https://www.r-graph-gallery.com/ggplot2-package.html
- Few Uefull Books
  - Easy R Programing for Beginners- Your Step-by-Step Guide to Learning R Programing by Felix Alvaro
  - R Programming for Beginners: An Introduction to Learn R Programming with Tutorials and Hands-On Examples by Nathan Metzler.
  - R for everyone: Advanced Analytics and Graphics by Jared P. Lander