Exploratory Data Analysis in R

Dr. Noah Mutai

Exploratory Data Analysis (EDA) is the process of analyzing and visualizing the data to get a better understanding of the data and glean insight from it.

There are various steps involved when doing EDA but the following are the common steps that a data analyst can take when performing EDA.

1. Import the data
2. Clean the data
3. Process the data
4. Visualize the data

This document will show you how to use visualization and transformation to explore your data in a systematic way, a task that statisticians call exploratory data analysis, or EDA for short. EDA is an iterative cycle. You:

1. Generate questions about your data for example;
   (a) What data types am I dealing with?
   (b) What is a typical value?
   (c) What uncertainty surrounds this estimate of typicality?
   (d) What type of variation occurs within my variables?
   (e) Does the data set contain outliers?
   (f) What type of covariation occurs between my variables?
   (g) More specifically, if there are any factors, do they have an effect?
2. Search for answers by visualizing, transforming, and modelling your data.
3. Use what you learn to refine your questions and/or generate new questions.

EDA is not a formal process with a strict set of rules. More than anything, EDA is a state of mind.

During the initial phases of EDA you should feel free to investigate every idea that occurs to you. Some of these ideas will pan out, and some will be dead ends.

1 Import data.

This is the first step in EDA. In this course we will explore several ways of importing different data sets into R.

```
movies <- read.csv("movie.csv")
head(movies)
```

2 Data cleaning

2.1 Check the data

```
headTail(movies)
anyNA(movies)
colSums(is.na(movies))
str(movies)
```

2.2 Rename column names.

```
colnames(movies) <- c("Film", "Genre", "CriticRating", "AudienceRating",
                      "BudgetMillions", "Year")
```

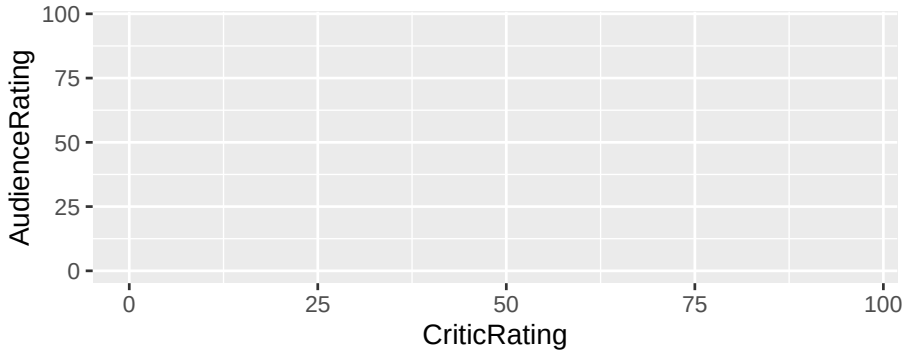
2.3 Convert year to factor

```
# factor(movies$Year)
movies$Year <- factor(movies$Year)
```

3 Visualization

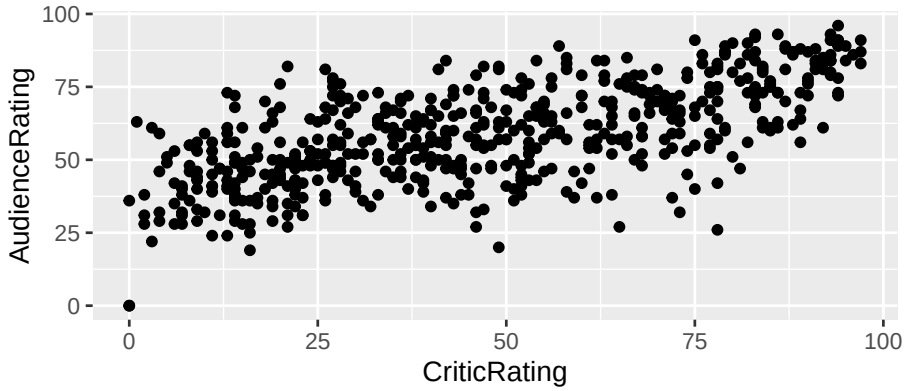
3.1 ggplot2

```
ggplot(data = movies,  aes(x = CriticRating, y = AudienceRating))
```



3.2 Geometry

```
ggplot(data = movies, aes(x = CriticRating, y = AudienceRating)) +
    geom_point()
```

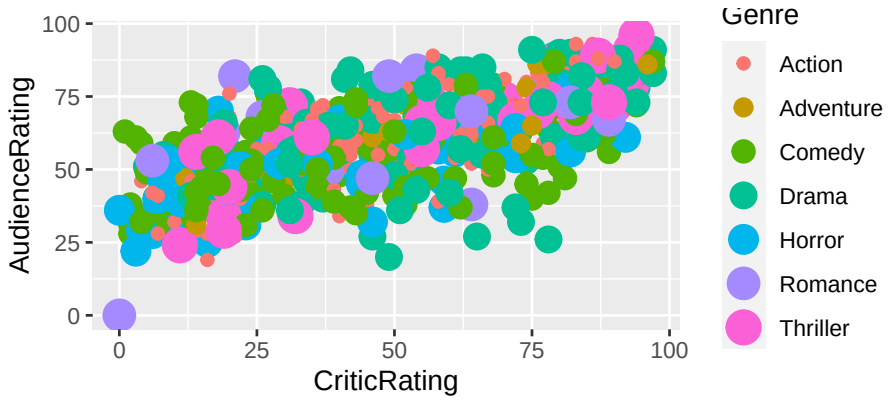


3.3 Add clour

```
ggplot(data = movies, aes(x = CriticRating, y = AudienceRating,
            colour = Genre)) + geom_point()
```

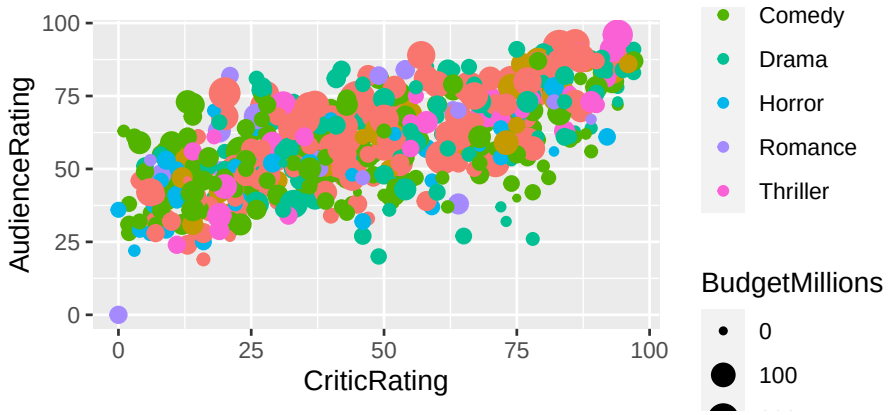
3.4 Add size

```
ggplot(data = movies, aes(x = CriticRating, y = AudienceRating, colour = Genre,
        size = Genre)) + geom_point()
```



3.5 Add size- better way

```
ggplot(data = movies, aes(x = CriticRating, y = AudienceRating, colour = Genre,
        size = BudgetMillions)) + geom_point()
```



3.6 Plotting with layers

- Create an object

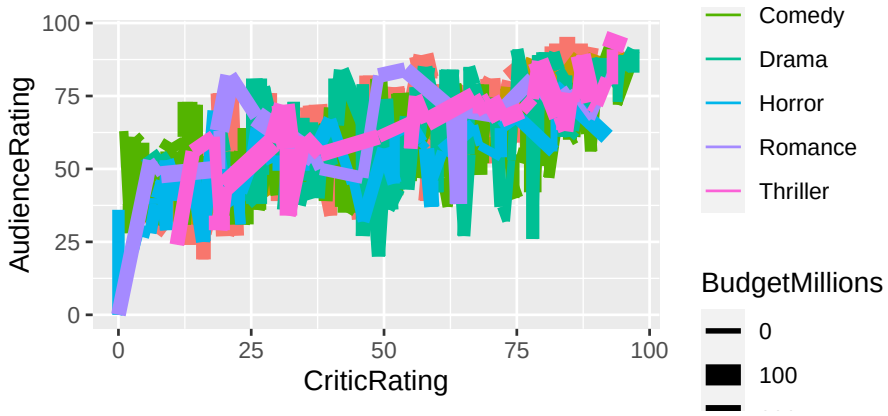
```
p <- ggplot(data = movies, aes(x = CriticRating, y = AudienceRating,
                colour = Genre, size = BudgetMillions))
```

3.7 Points

```
p +
  geom_point()
```

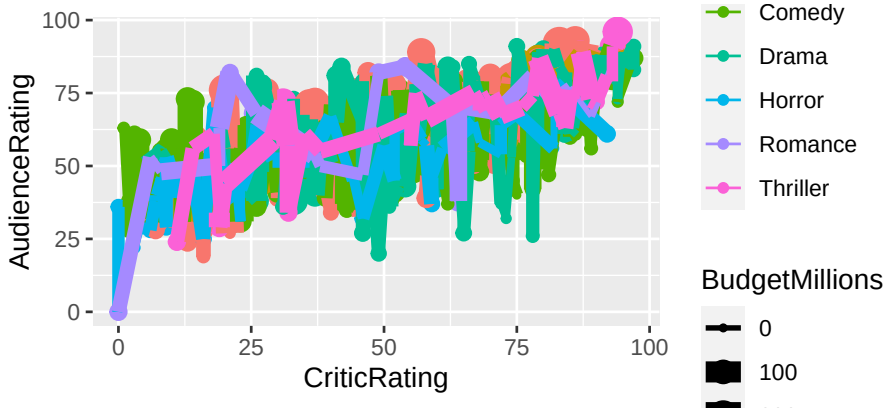
3.8 Lines

```
p +
  geom_line()
```



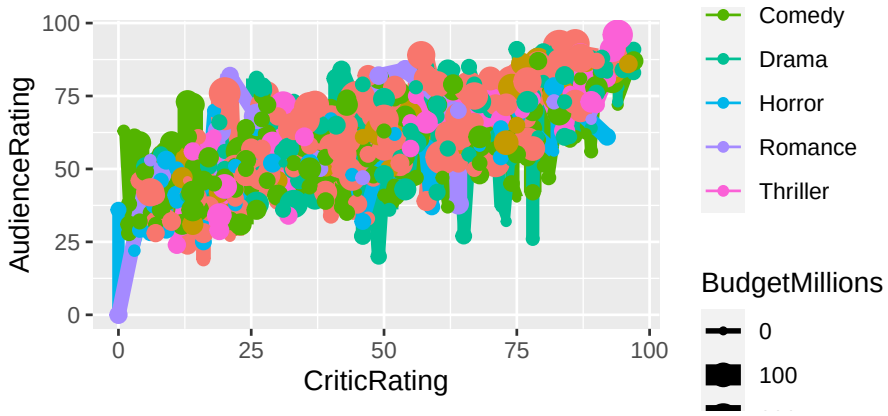
3.9 Multiple layers

```
p +
  geom_point() + geom_line()
```



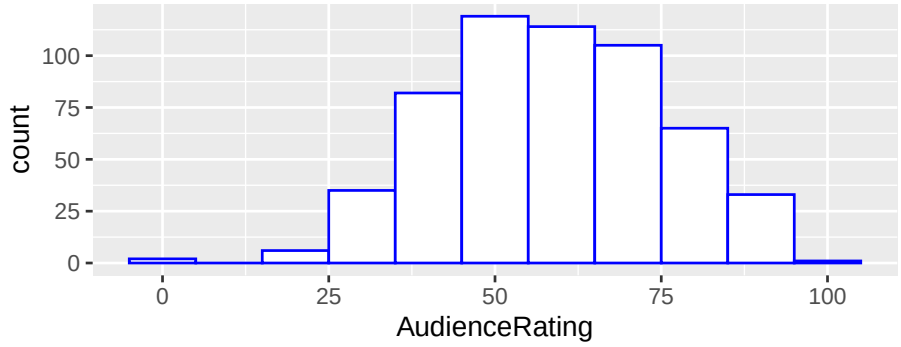
3.10 Lines and points

```
p +
  geom_line() + geom_point()
```



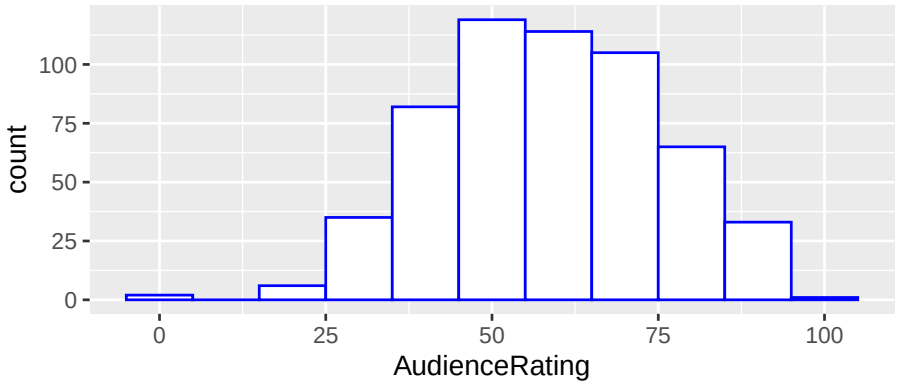
3.11 Starting layer tips

```
t <- ggplot(data = movies, aes(x = AudienceRating))
t + geom_histogram(binwidth = 10,
                   fill = "White",colour = "Blue")
```



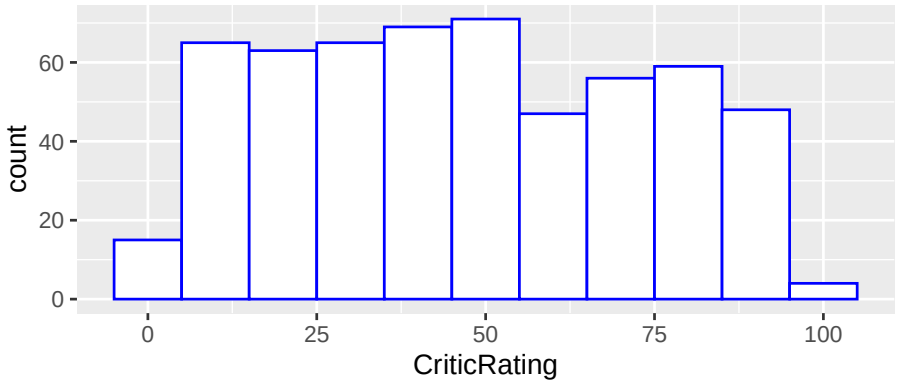
3.12 Another way

```
t <- ggplot(data = movies) ## the same data set.
t + geom_histogram(binwidth = 10, aes(x = AudienceRating),
  fill = "White",colour = "Blue")
```



3.13 4

```
t + geom_histogram(binwidth = 10, aes(x = CriticRating),
  fill = "White",colour = "Blue")
```



- We are not recreating the variable.

3.14 5 skeleton plot

- Use different data set

```
t <- ggplot()
```

3.15 Statistical transformations

? geom_smooth

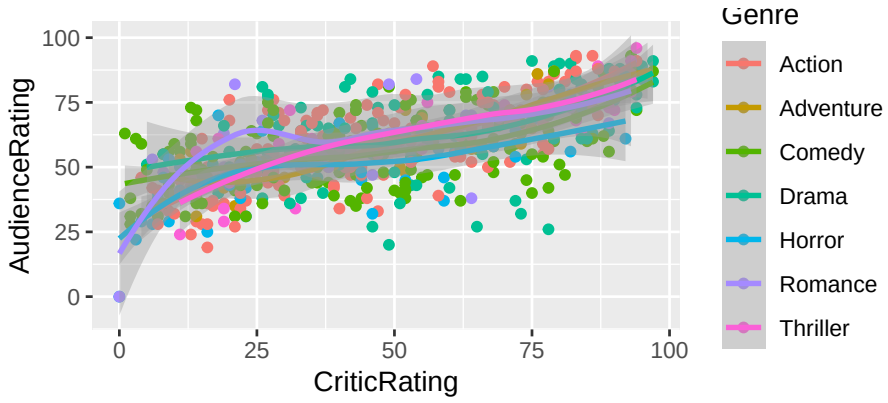
```
u <- ggplot(data = movies, aes(x = CriticRating, y = AudienceRating,
    color = Genre))
```

3.16 ...

```
u + geom_point()
```

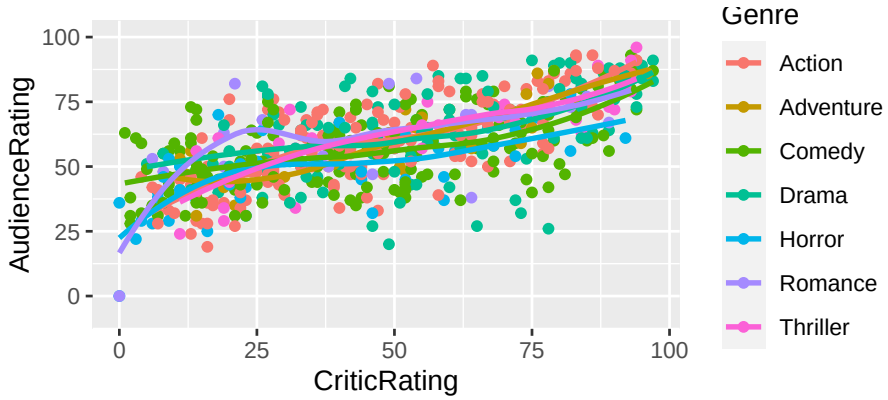
3.17 ....

```
u +
  geom_point() + geom_smooth()
```



3.18 ...

```
u +
  geom_point() +
  geom_smooth(fill = NA)
```

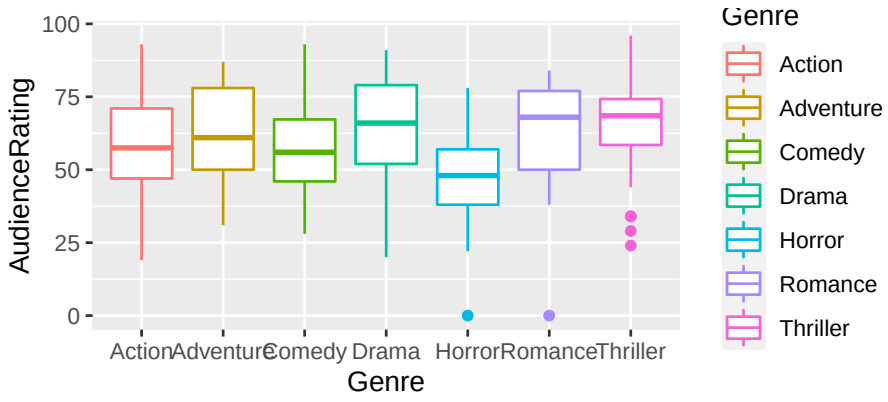


3.19 Boxplots

```
u <- ggplot(data = movies, aes(x = Genre, y = AudienceRating,color = Genre))
```

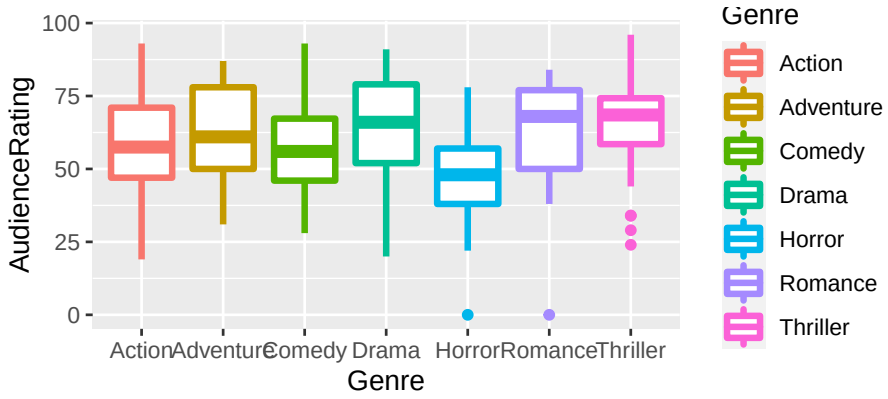
3.20 ...

```
u +
  geom_boxplot()
```



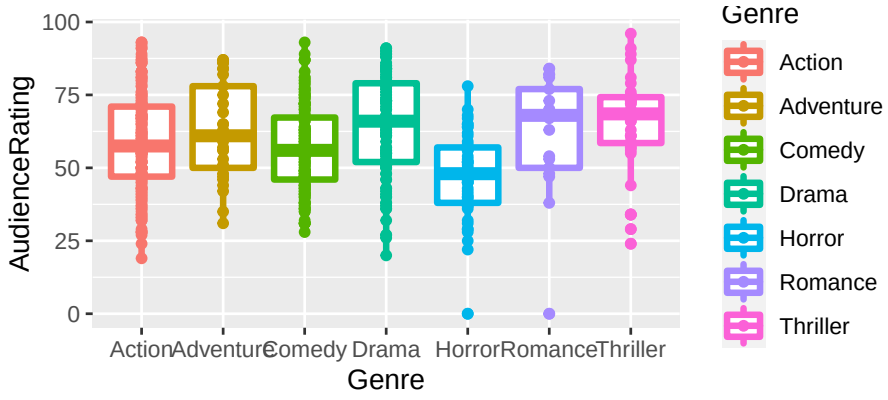
3.21 Size

```
u +
  geom_boxplot(size = 1.2)
```



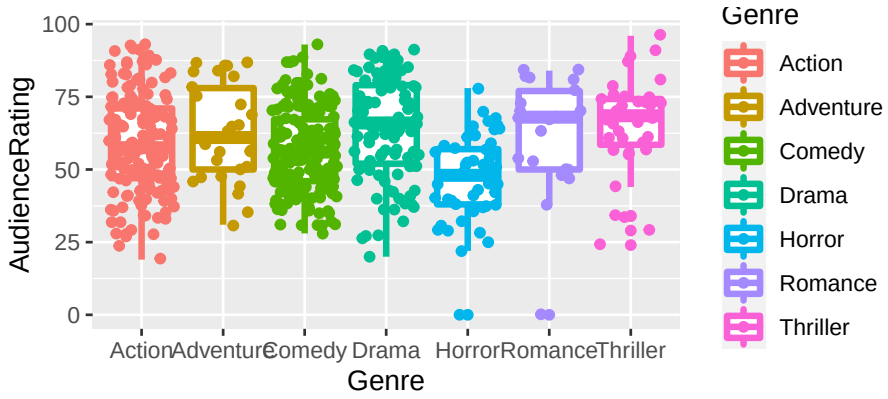
3.22 ...

```
u +
  geom_boxplot(size = 1.2) +
    geom_point()
```



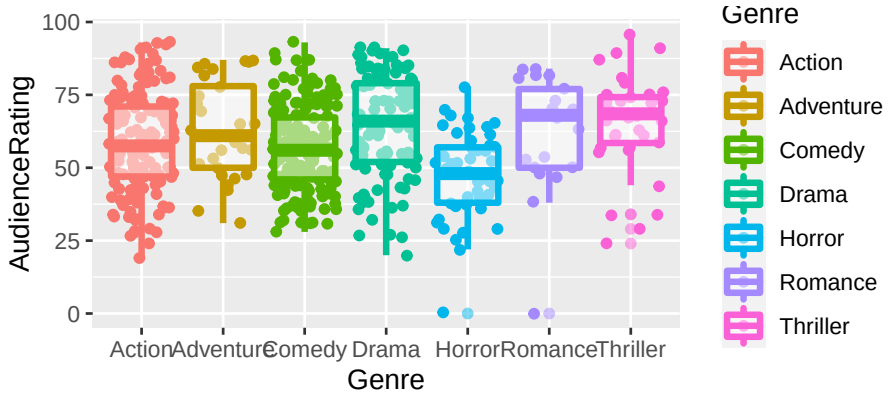
##...tip/hack

```
u +
  geom_boxplot(size = 1.2) +
    geom_jitter()
```



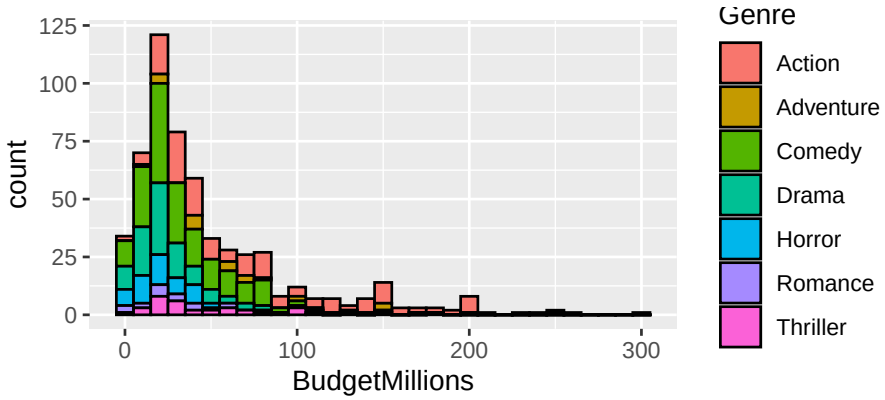
3.23 Another way

```
u +
  geom_jitter() +
    geom_boxplot(size = 1.2, alpha = 0.5)
```



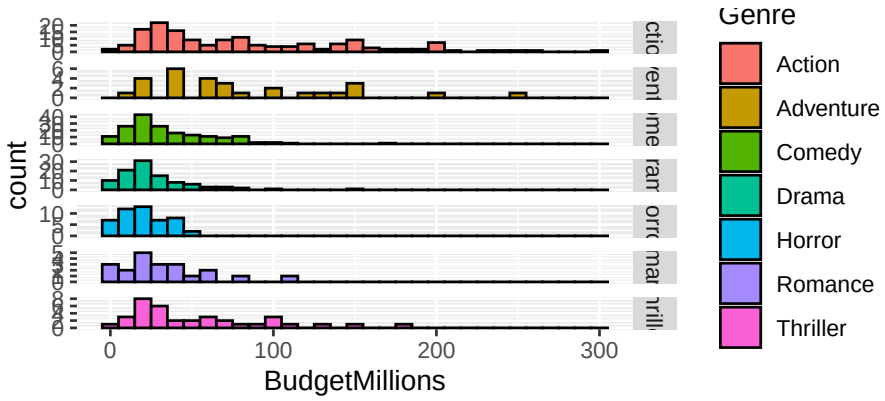
3.24 Facets

```
v <- ggplot(data = movies, aes(x = BudgetMillions))
v + geom_histogram(binwidth = 10, aes(fill = Genre),
                   colour = "Black")
```



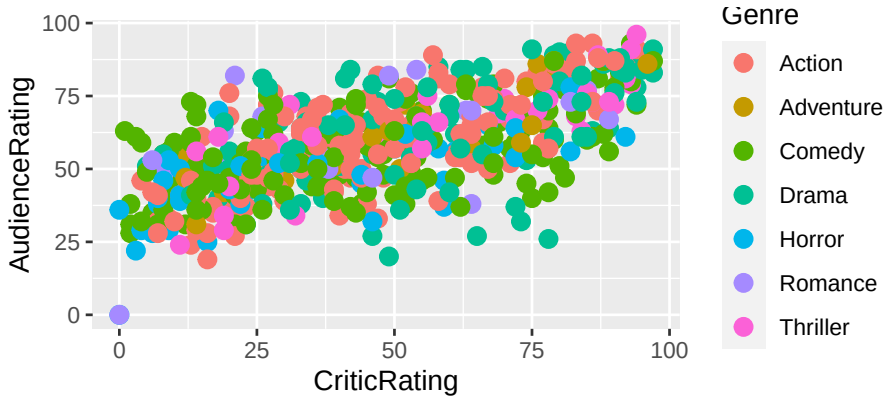
3.25 Facets

```
v + geom_histogram(binwidth = 10, aes(fill = Genre), colour = "Black") +
  facet_grid(Genre ~ ., scales = "free")
```



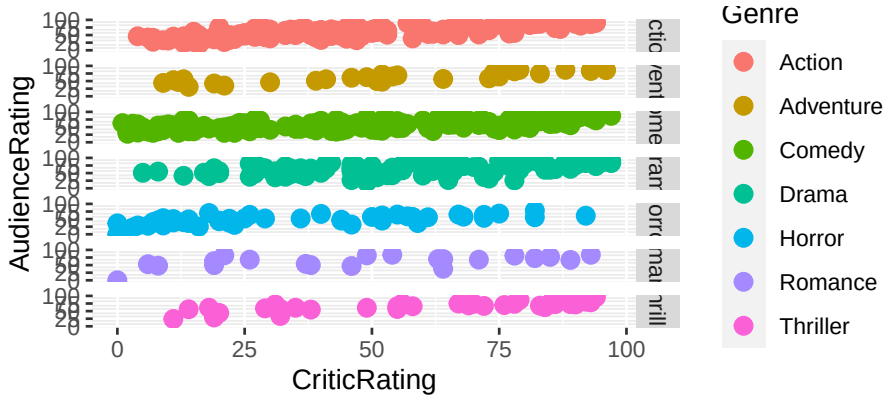
3.26 scatter plots

```
w <- ggplot(data = movies, aes(x = CriticRating, y = AudienceRating, color = Genre))
w + geom_point(size = 3)
```



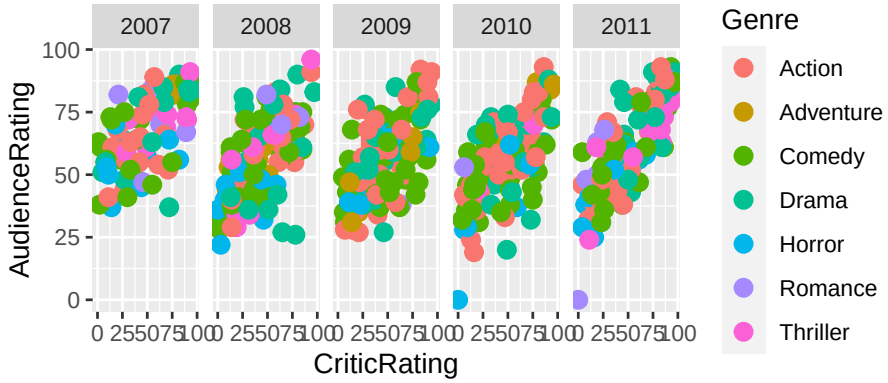
3.27 Facets

```
w +
  geom_point(size = 3) +
  facet_grid(Genre ~ .)
```



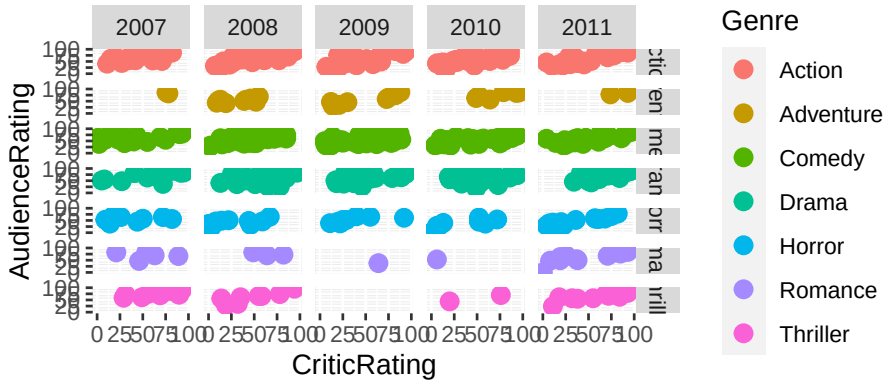
3.28 ...

```
w +
  geom_point(size = 3) +
  facet_grid(. ~ Year)
```



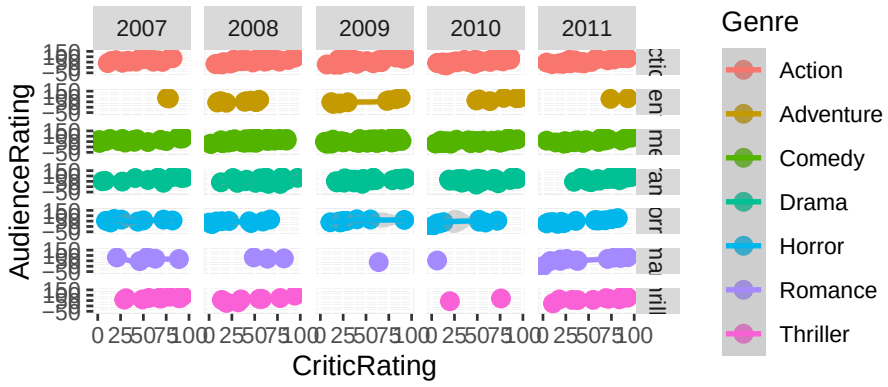
3.29 ...

```
w + geom_point(size = 3) +
  facet_grid(Genre ~ Year)
```



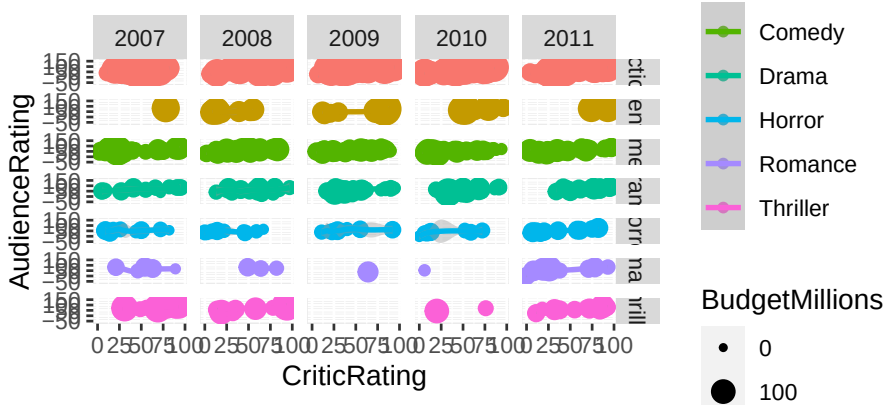
##...

```
w +
  geom_point(size = 3) +
  geom_smooth() +
  facet_grid(Genre ~ Year)
```



3.30 ...

```
w +
  geom_point(aes(size = BudgetMillions)) +
  geom_smooth() +
  facet_grid(Genre ~ Year)
```



3.31 Coordinates

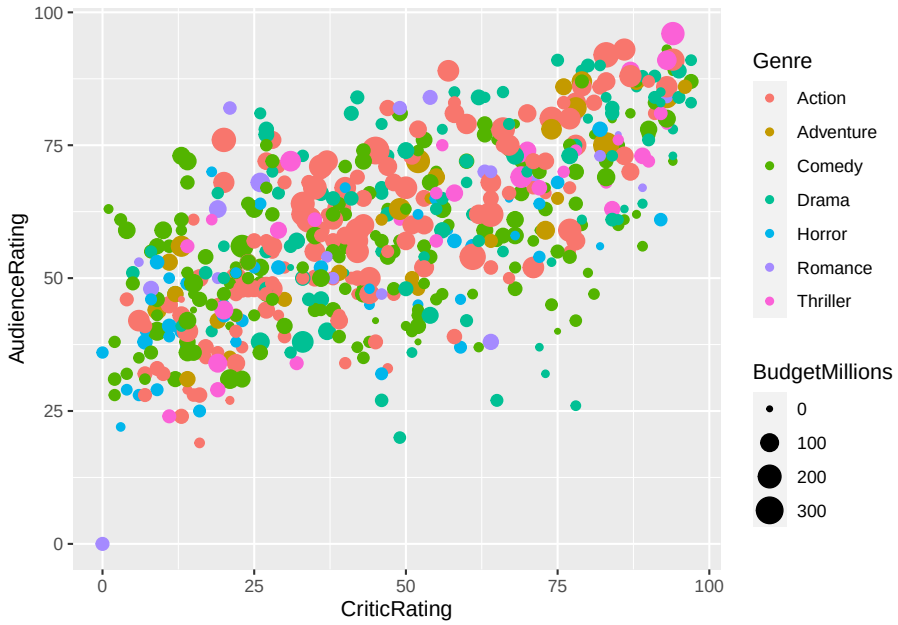
3.31.1 limits

```
m <- ggplot(data = movies,aes(x = CriticRating, y = AudienceRating,
                size = BudgetMillions, color = Genre))
```

3.31.1.1 zoom

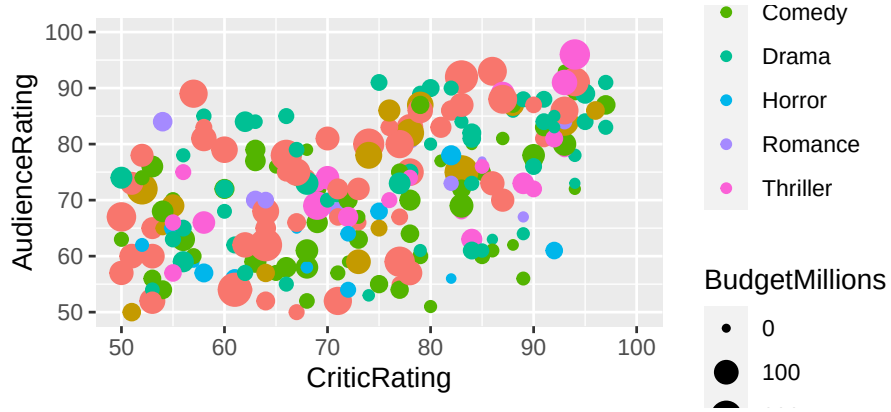
3.32 ...

```
m + geom_point()
```



3.33 ....

```
m + geom_point() + xlim(50, 100) +
  ylim(50, 100)
```

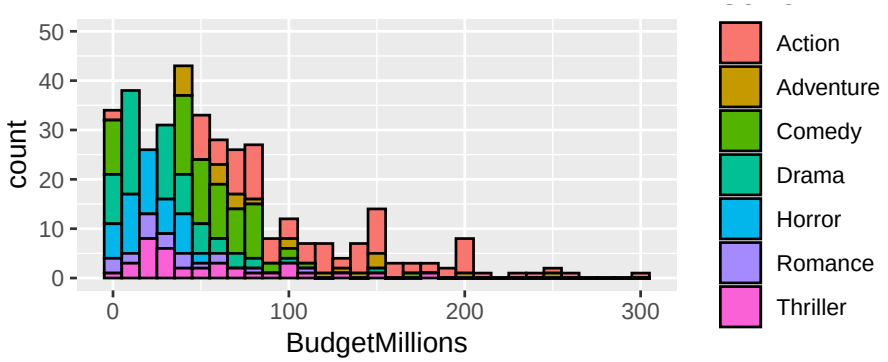


3.34 wont work well always

```
n <- ggplot(data = movies, aes(x = BudgetMillions))
```

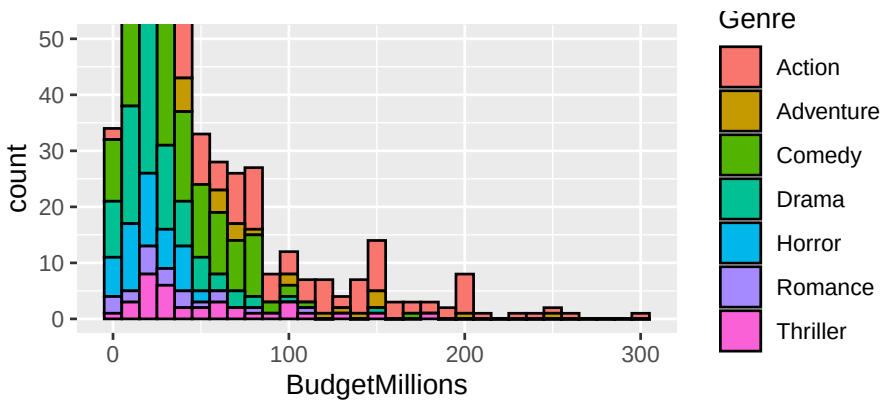
3.35 ...

```
n + geom_histogram(binwidth = 10, aes(fill = Genre),
                   color = "Black") + ylim(0, 50)
```



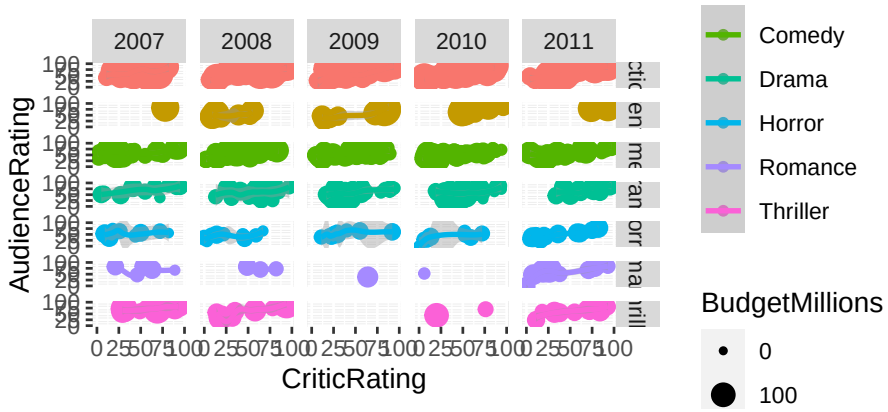
3.36 instead zoom

```
n + geom_histogram(binwidth = 10, aes(fill = Genre), color = "Black") +
  coord_cartesian(ylim = c(0, 50))
```



3.37 improved 1

```
w + geom_point(aes(size = BudgetMillions)) + geom_smooth() +
    facet_grid(Genre ~ Year) + coord_cartesian(ylim = c(0, 100))
```

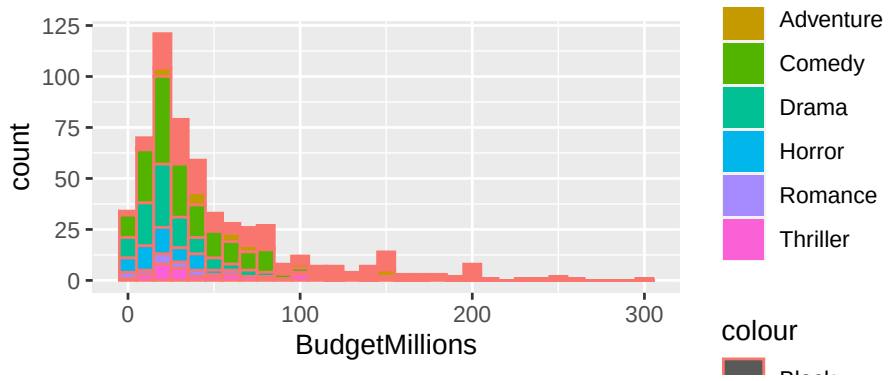


3.38 Themes

```
o <- ggplot(data = movies, aes(x = BudgetMillions))
```

3.39 ...

```
o + geom_histogram(binwidth = 10, aes(fill = Genre,
                        color = "Black"))
```

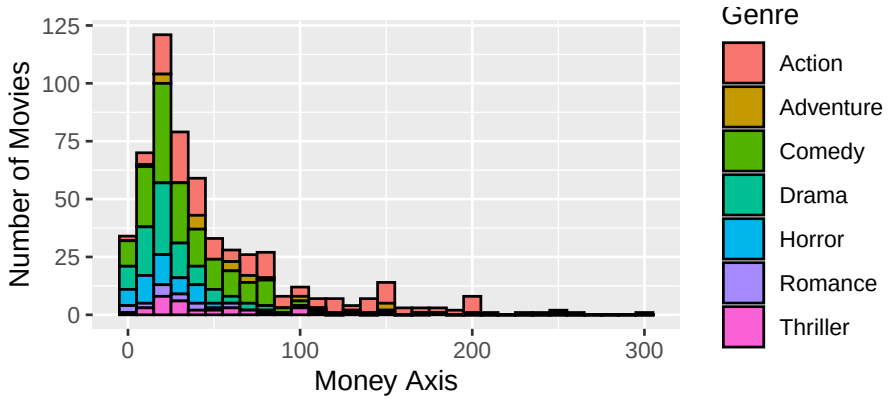


3.40 ...

```
h <- o + geom_histogram(binwidth = 10, aes(fill = Genre),
                        color = "Black")
```

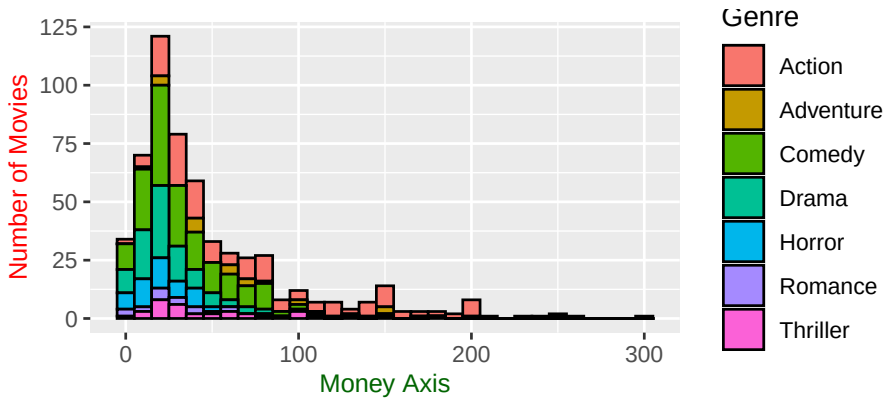
3.41 axes labels

```
h + xlab("Money Axis") + ylab("Number of Movies")
```



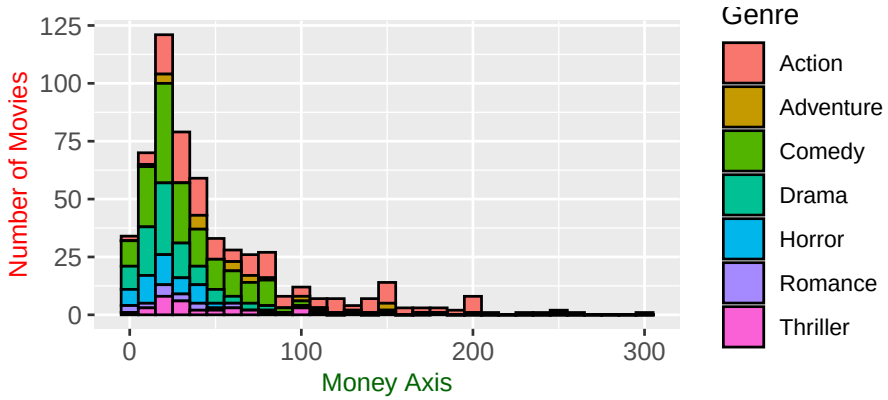
3.42 label formating

```
h + xlab("Money Axis") + ylab("Number of Movies") +
  theme(axis.title.x = element_text(colour = "DarkGreen",
  size = 10), axis.title.y = element_text(color = "Red", size = 10))
```



3.43 tick mark formating

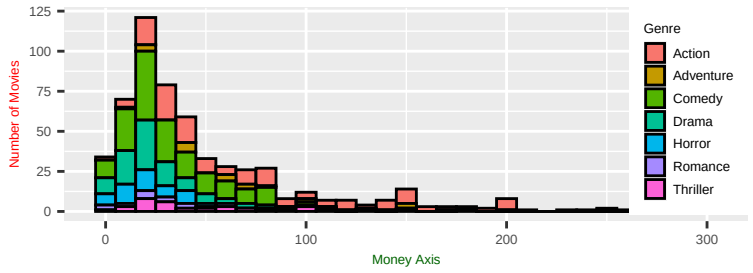
```
h + xlab("Money Axis") +
  ylab("Number of Movies") +
  theme(axis.title.x = element_text(colour = "DarkGreen",
    size = 10),axis.title.y = element_text(color = "Red",
    size = 10),axis.text.x = element_text(size = 10),
    axis.text.y = element_text(size = 10))
```



? theme

3.44 legend formatting

```
h + xlab("Money Axis") + ylab("Number of Movies") +
  theme(legend.key.size = unit(0.3, 'cm'), #change legend key size
        legend.key.height = unit(0.3, 'cm'), #change legend key height
        legend.key.width = unit(0.3, 'cm'), #change legend key width
      axis.title.x = element_text(colour = "DarkGreen",
  size = 5),axis.title.y = element_text(color = "Red",
  size = 5),axis.text.x = element_text(size = 5),
    axis.text.y = element_text(size = 5),
    legend.title = element_text(size = 5),
    legend.text = element_text(size = 5),
legend.position = c(1,1), legend.justification = c(1,1))
```



3.45 tittle

```
h + xlab("Money Axis") + ylab("Number of Movies") +
  ggtitle("Movie Budget Distribution") +
  theme(legend.key.size = unit(0.3, 'cm'), #change legend key size
        legend.key.height = unit(0.3, 'cm'), #change legend key height
```

```
      legend.key.width = unit(0.3, 'cm'), #change legend key width

    axis.title.x = element_text(colour = "DarkGreen",
size = 5),axis.title.y = element_text(colour = "Red",
size = 5),axis.text.x = element_text(size = 5),
  axis.text.y = element_text(size = 5),
  legend.title = element_text(size = 5),
  legend.text = element_text(size = 5),
  legend.position = c(1, 1),
  legend.justification = c(1, 1),
  plot.title = element_text(colour = "DarkBlue", size = 5))
```