

# Data Visualisation Lab-5

Bibek Sapkota

## Part-I

### Basic Plot types

Task 1: Importing the library

```
library(ggplot2)
```

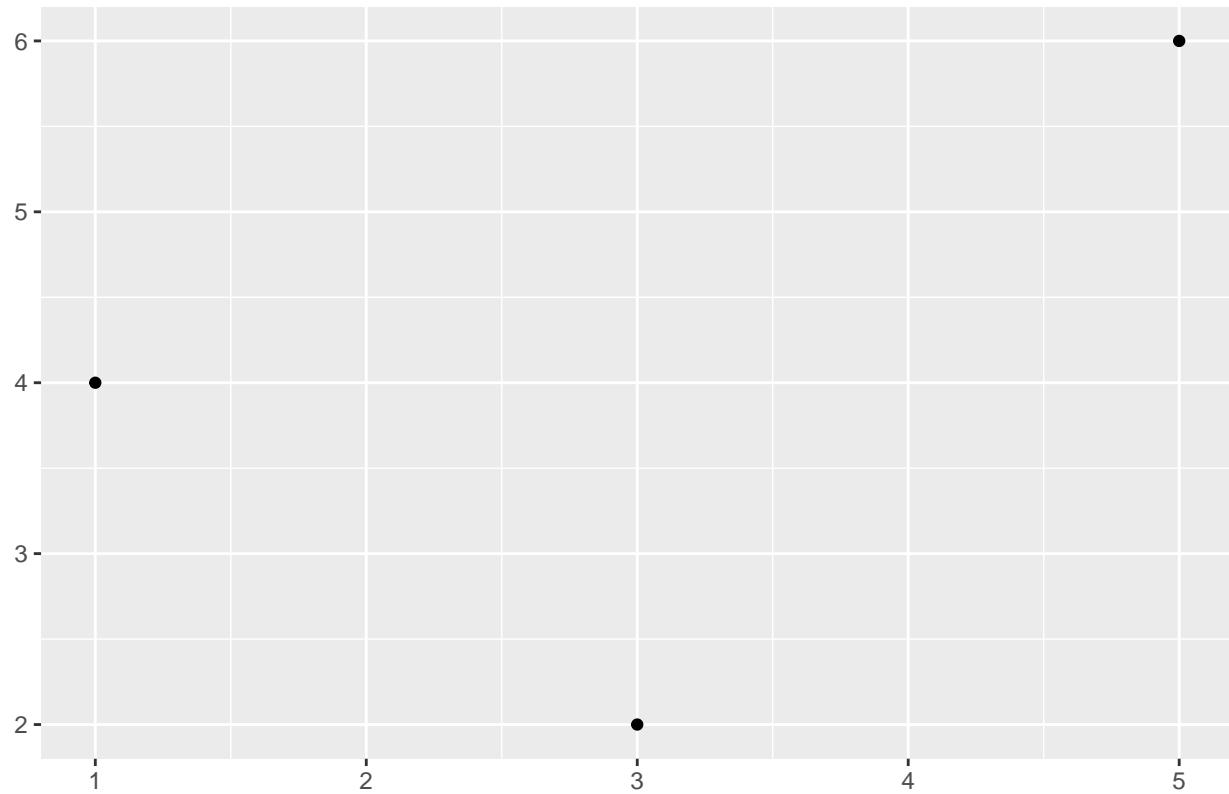
Task 1: Making a basic dataframe

```
df <- data.frame(  
  x=c(3,1,5),  
  y=c(2,4,6),  
  label=c("a","b","c")  
)
```

Task 2: Visualizing the df dataset in point, bar, line, area, path, text, tile and polygon graph

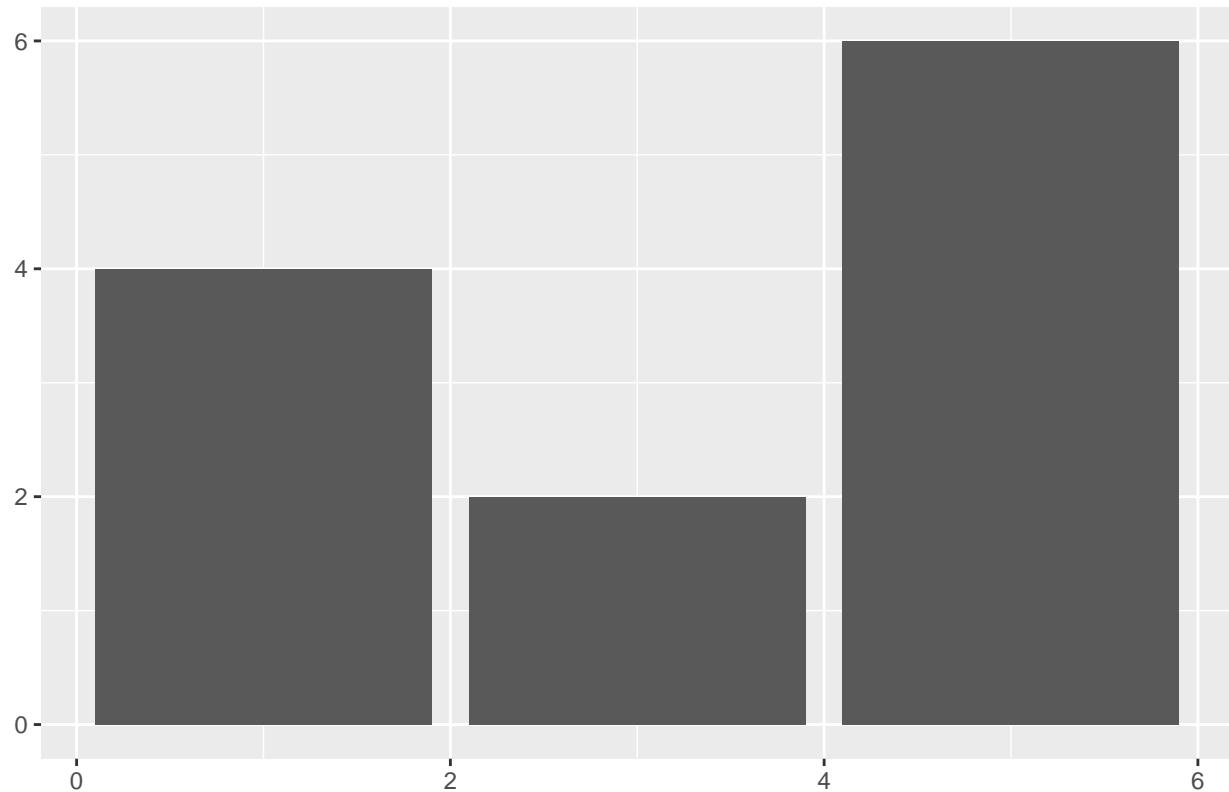
```
p<- ggplot(df,aes(x,y,label=label))+  
  xlab(NULL)+ylab(NULL)  
 p + geom_point() + ggtitle("geom_point")
```

`geom_point`



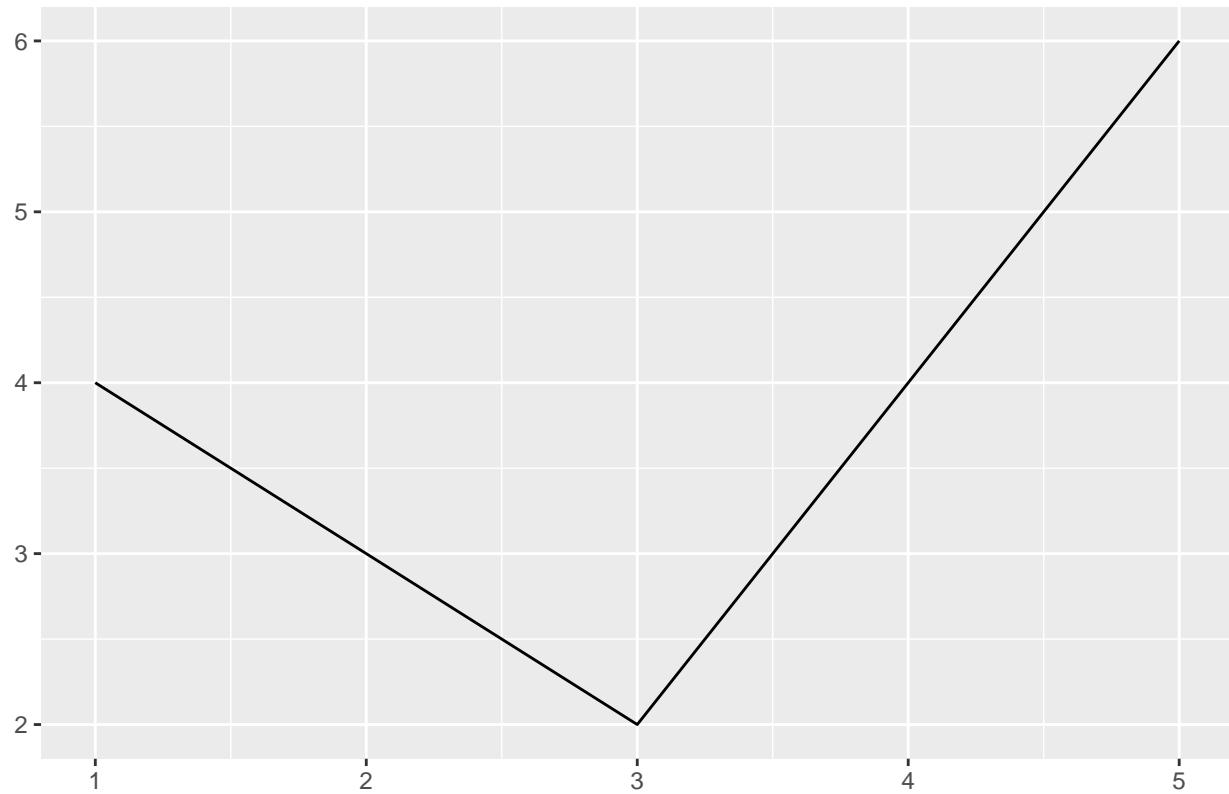
```
p + geom_bar(stat="identity") +  
  ggtitle("geom_bar(stat=\"identity\")")
```

```
geom_bar(stat="identity")
```



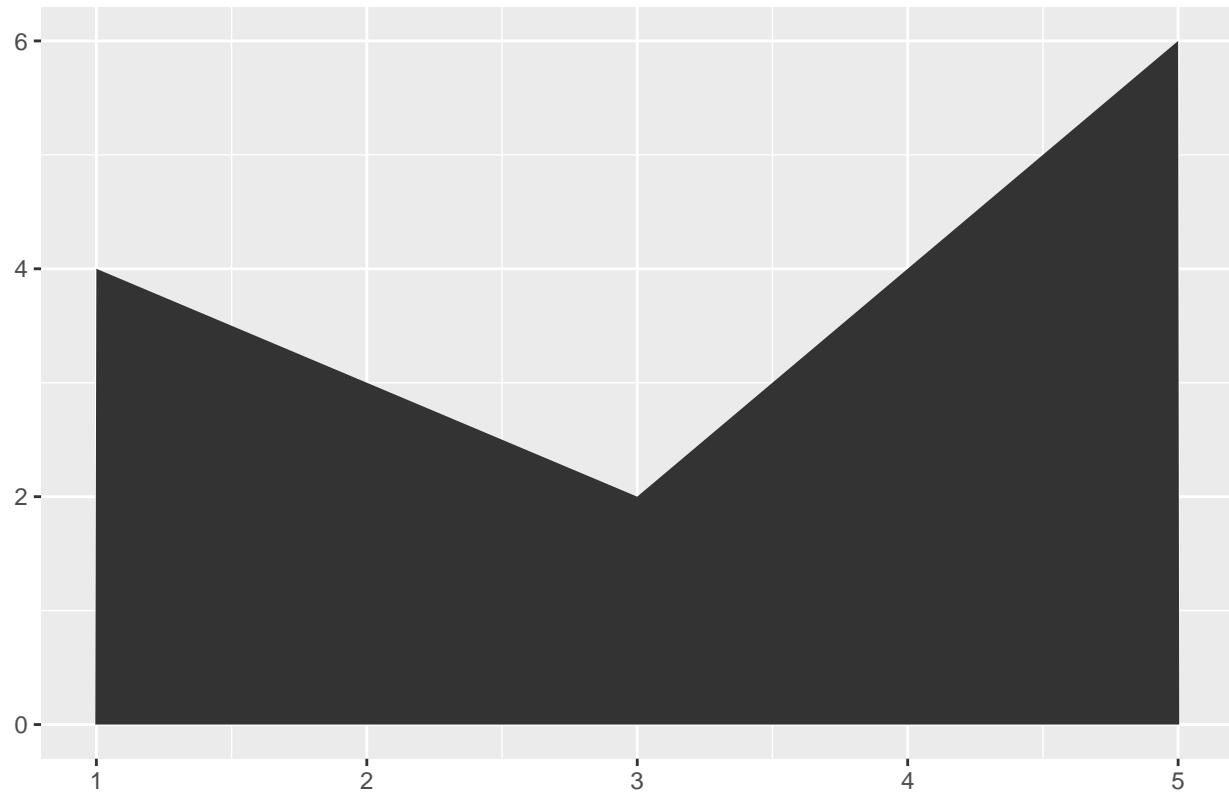
```
p + geom_line() + ggtitle("geom_line")
```

## geom\_line



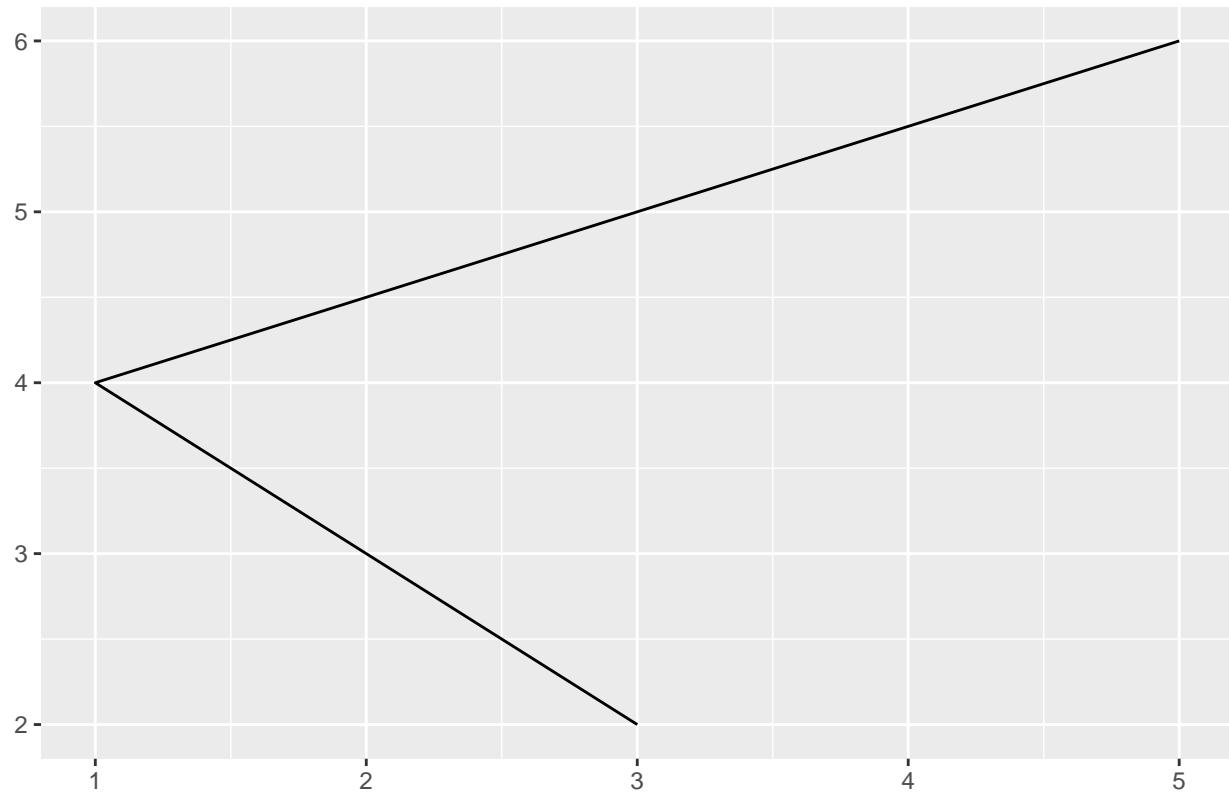
```
p + geom_area() + ggtitle("geom_area")
```

`geom_area`



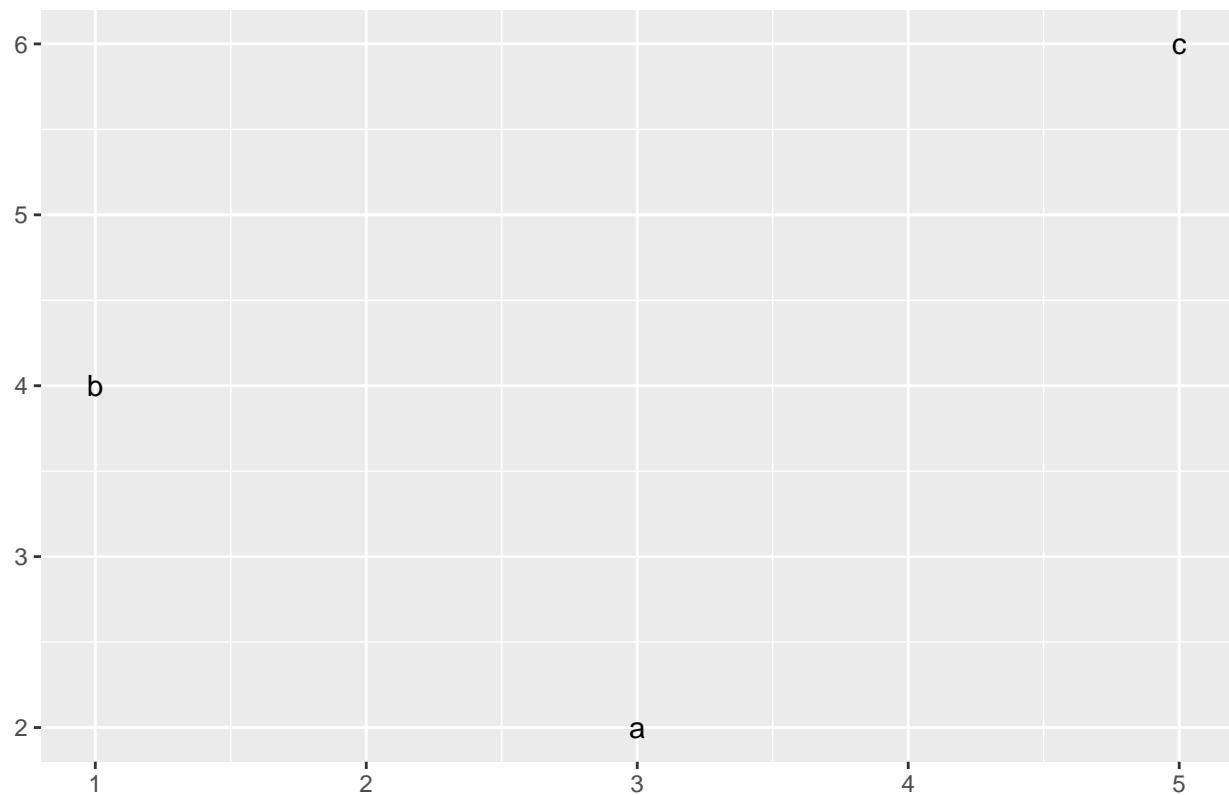
```
p + geom_path() + ggtitle("geom_path")
```

## geom\_path



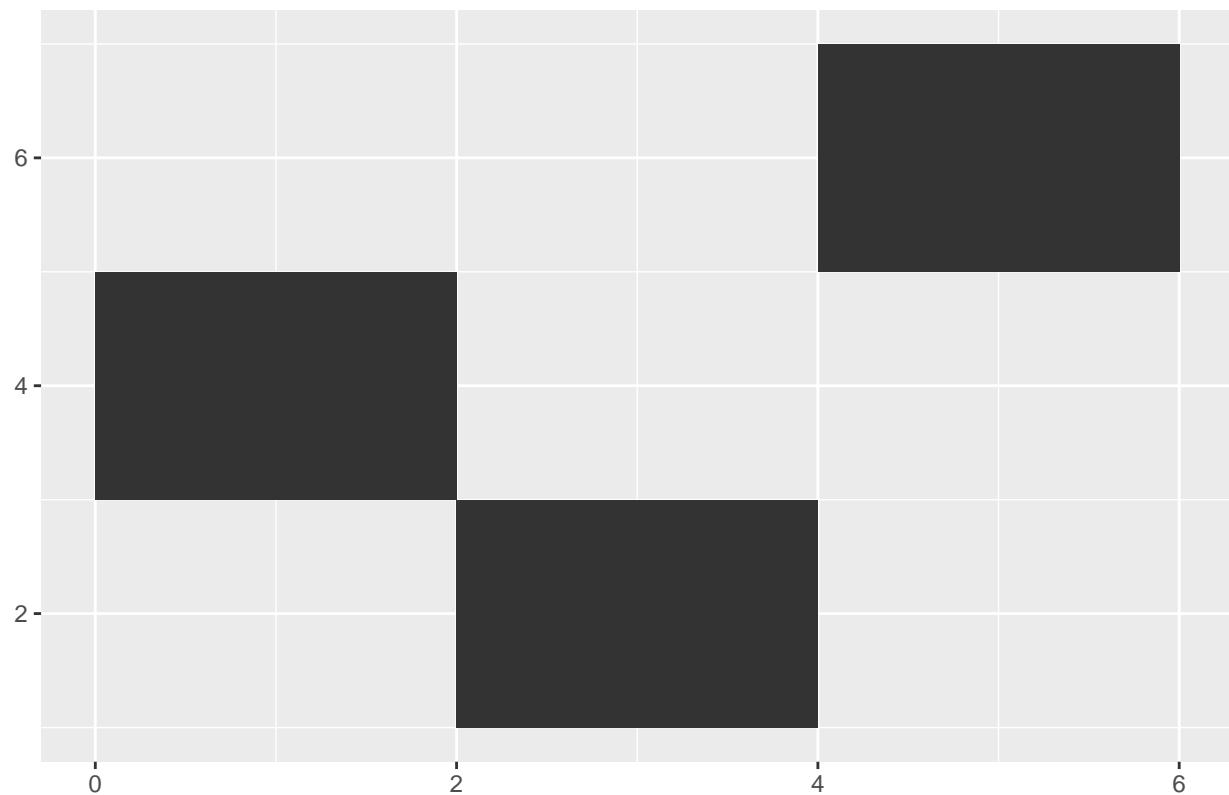
```
p + geom_text() + ggtitle ("geom_text")
```

### geom\_text



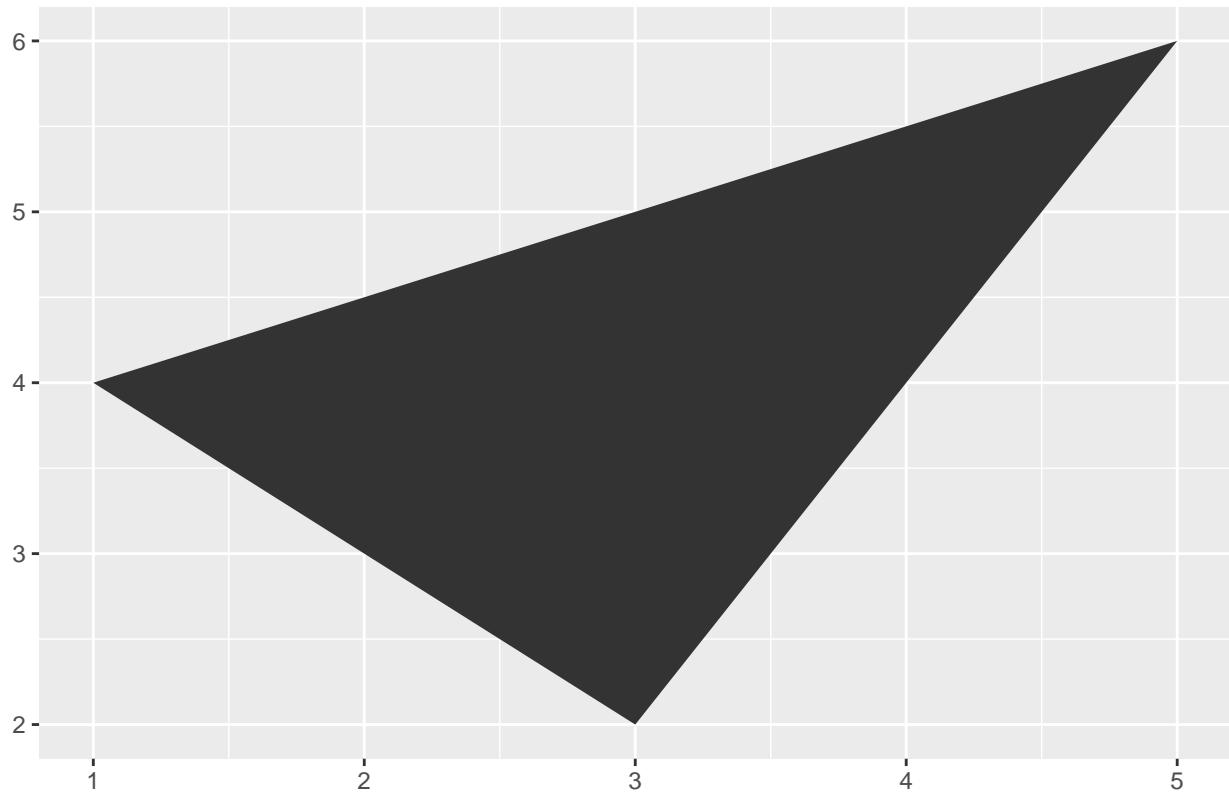
```
p + geom_tile() + ggtitle( "geom_tile")
```

## geom\_tile



```
p + geom_polygon() + ggtitle ("geom_polygon")
```

geom\_polygon

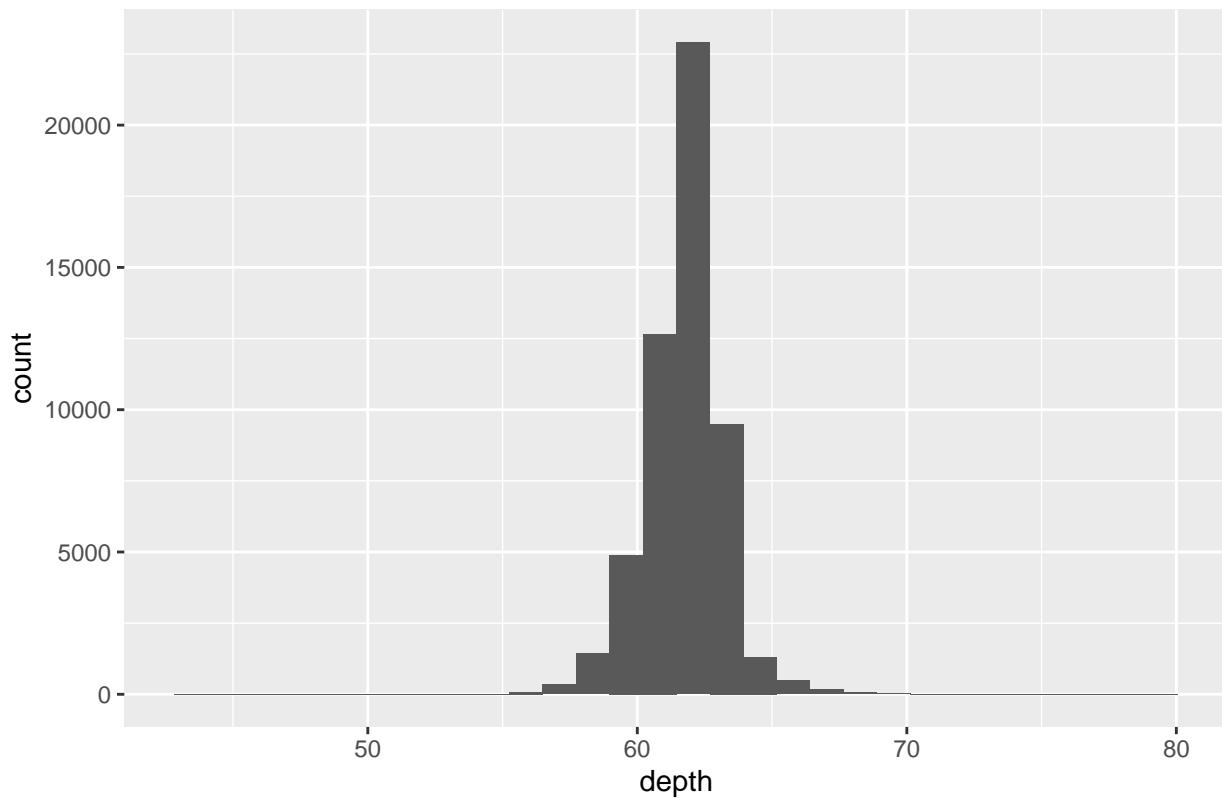


## Displaying Distribution

Task :visualizing distribution of diamond dataset depth in histogram and experimenting with the binwidth to find a revealing view.

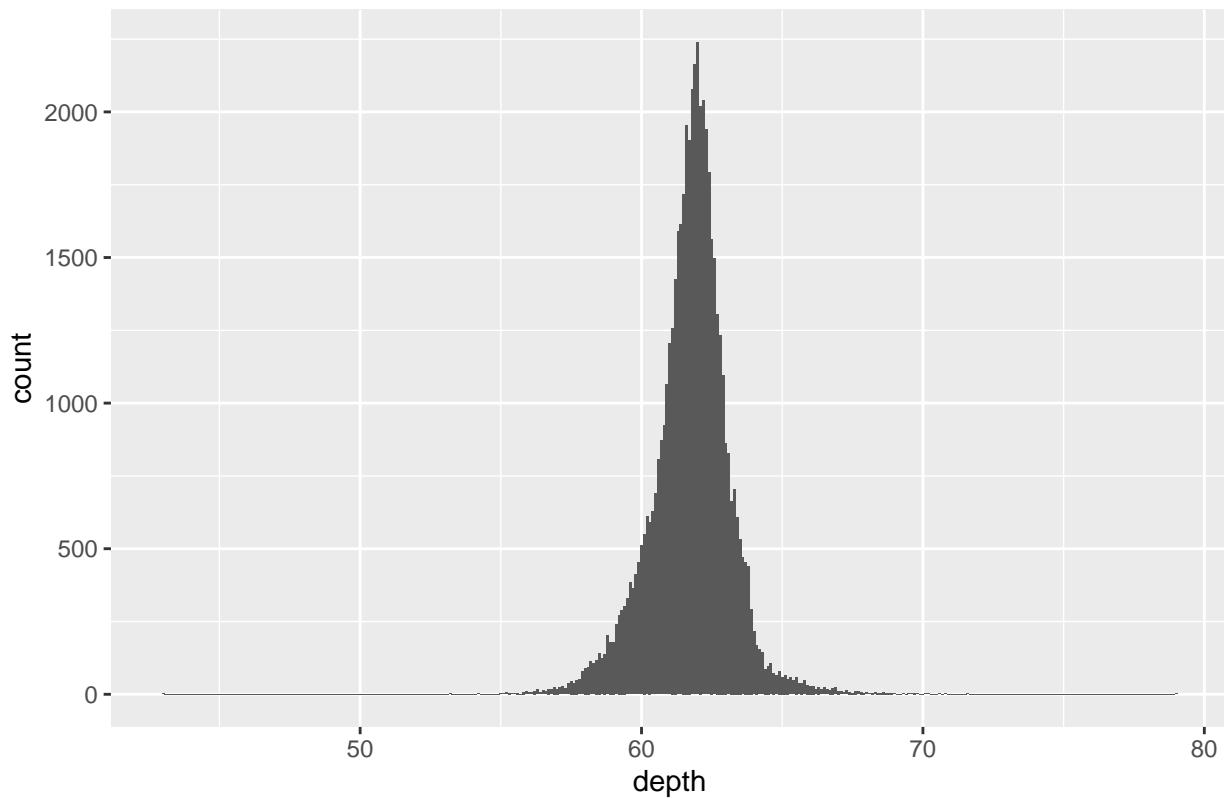
```
data("diamonds")  
  
ggplot(diamonds, aes(x=depth)) +  
  geom_histogram() +  
  ggtitle("Default binwidth")  
  
## 'stat_bin()' using 'bins = 30'. Pick better value with 'binwidth'.
```

## Default binwidth



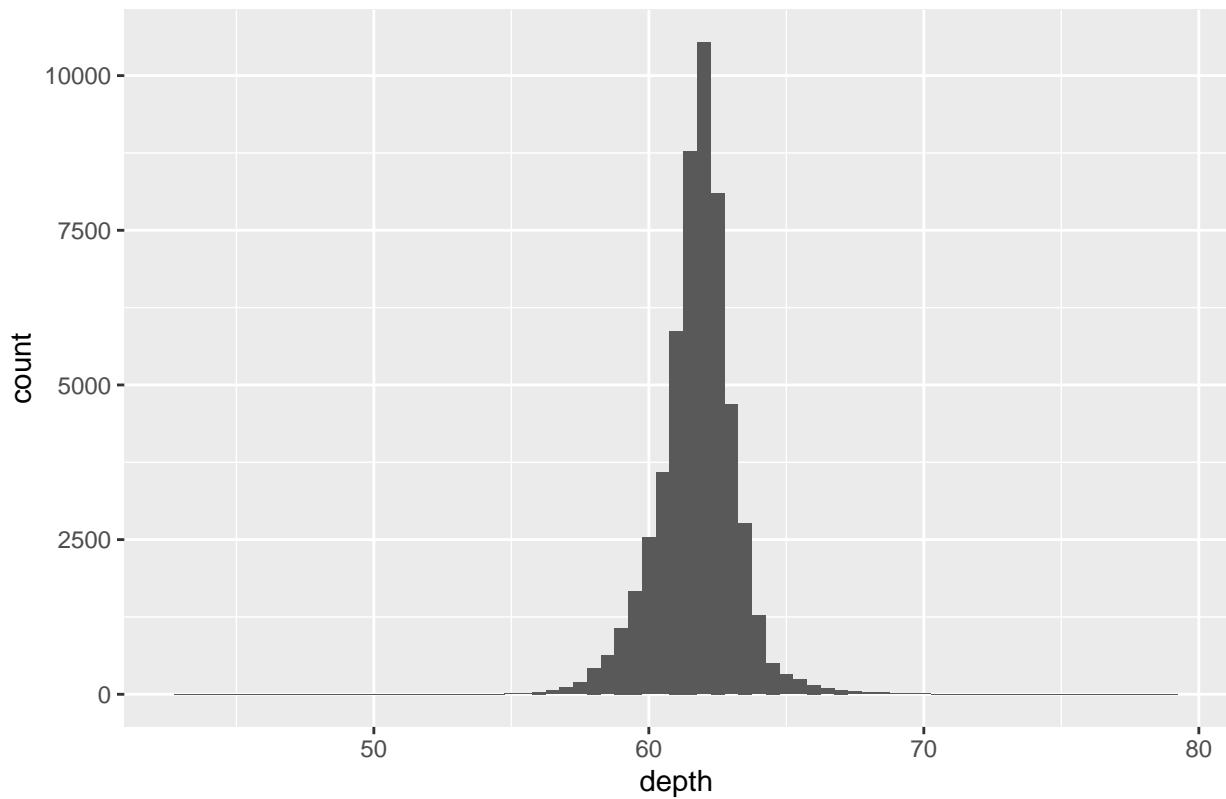
```
ggplot(diamonds, aes(x=depth)) +  
  geom_histogram(binwidth=0.1) +  
  ggtitle("Binwidth = 0.1")
```

**Binwidth = 0.1**

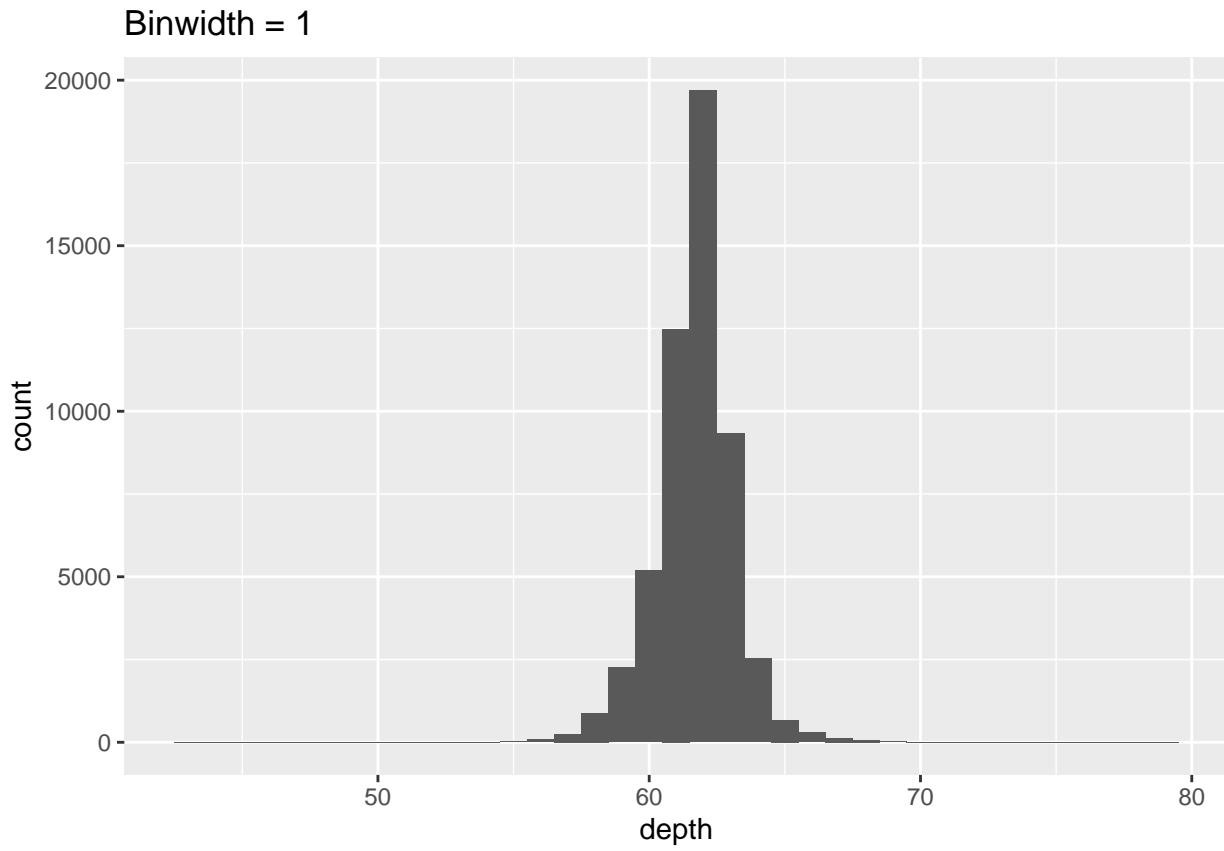


```
ggplot(diamonds, aes(x=depth)) +  
  geom_histogram(binwidth=0.5) +  
  ggtitle("Binwidth = 0.5")
```

**Binwidth = 0.5**



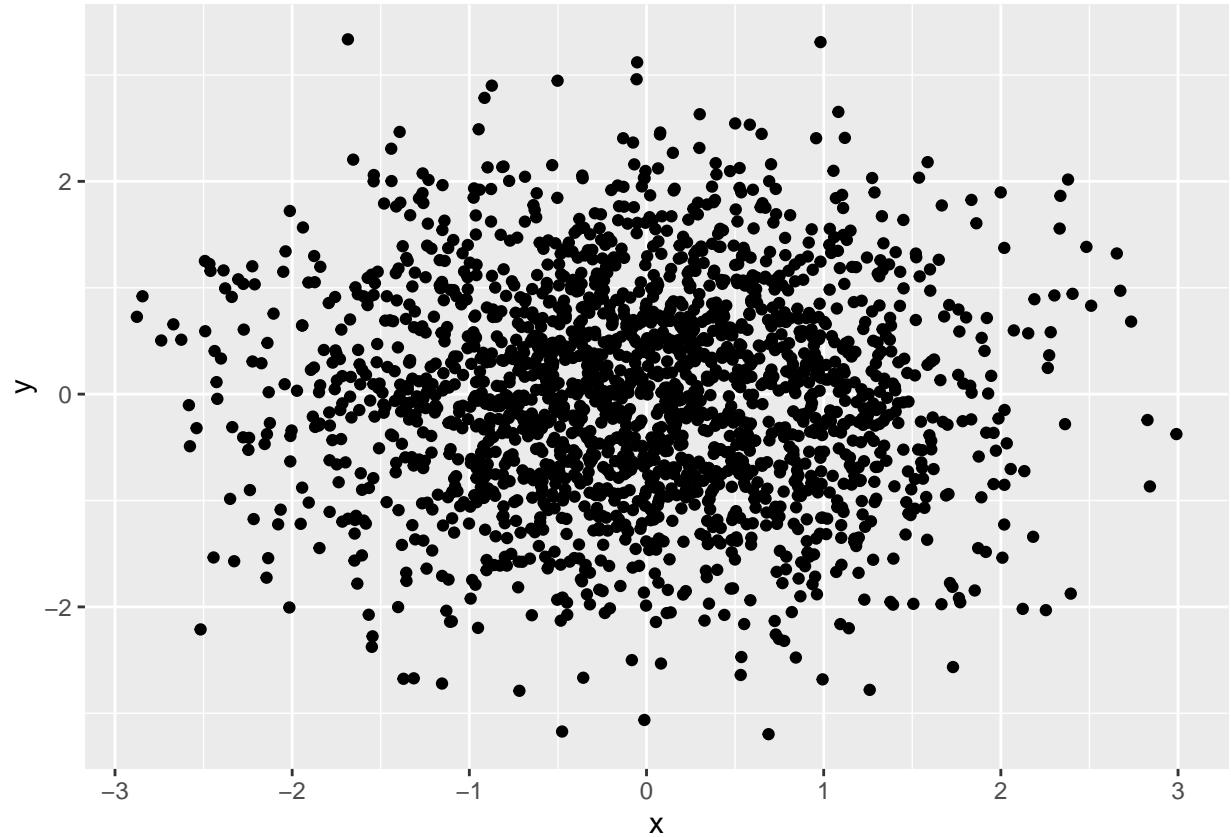
```
ggplot(diamonds, aes(x=depth)) +  
  geom_histogram(binwidth=1) +  
  ggtitle("Binwidth = 1")
```



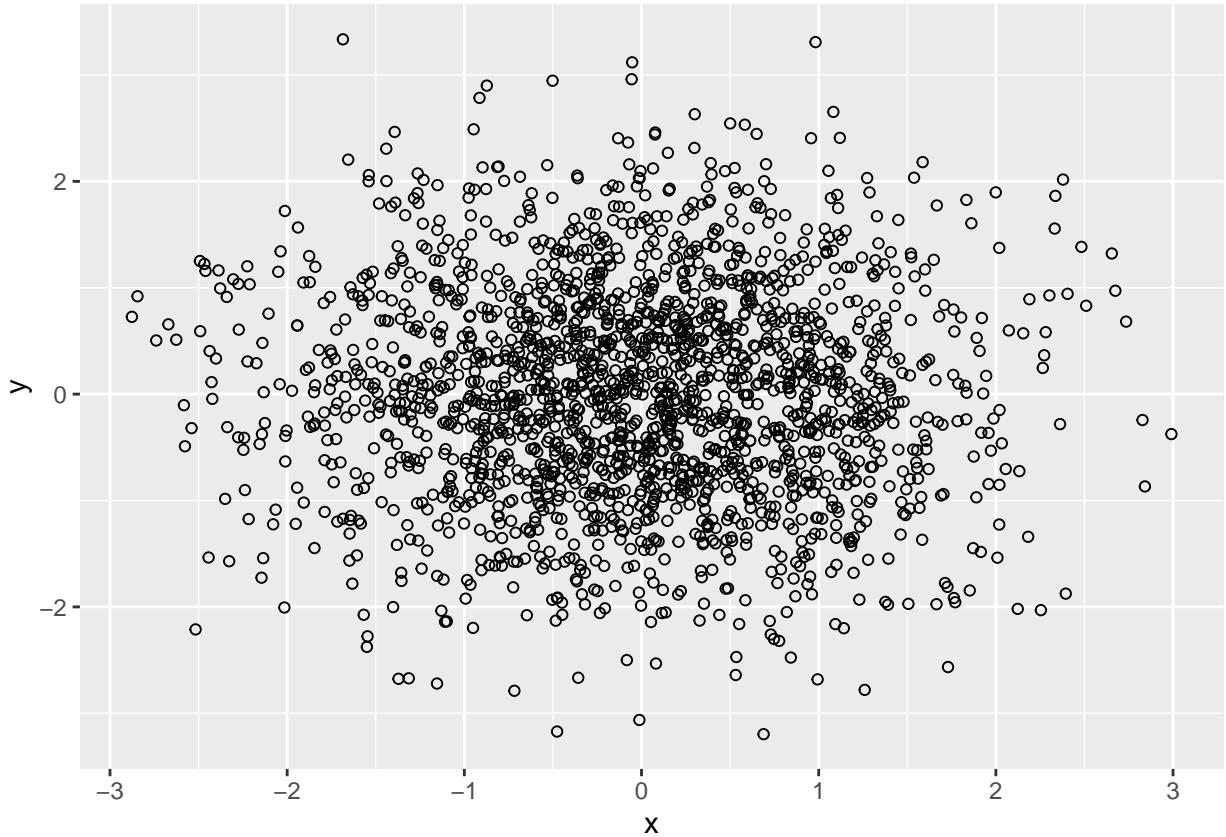
## Dealing with overplotting

Task: Creating a dataset with 2000rows and columnsand then ploting it using geom\_point with different shape to see overplotting of the data.

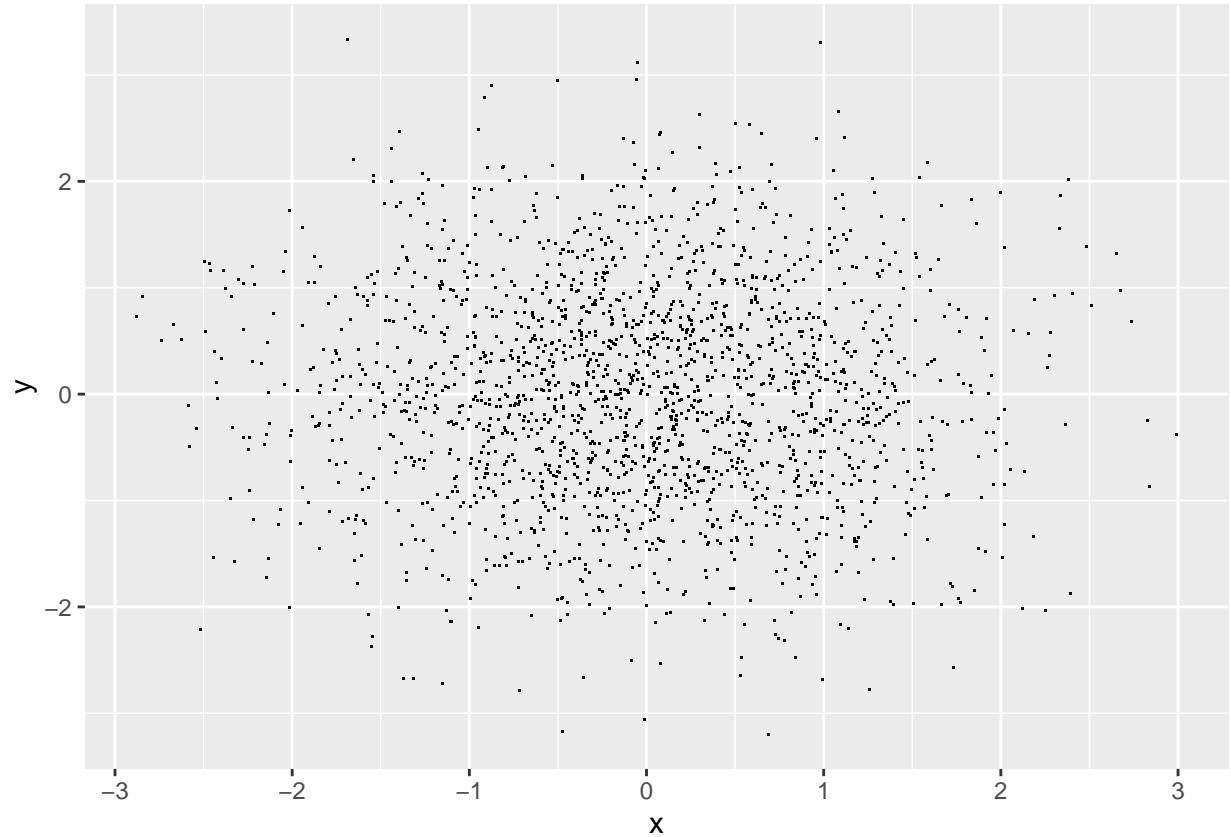
```
df <- data.frame(x = rnorm(2000), y = rnorm(2000))
norm <- ggplot(df, aes(x, y))
norm + geom_point()           # Default pixel sized
```



```
norm + geom_point(shape = 1) # 1 pixel sized
```

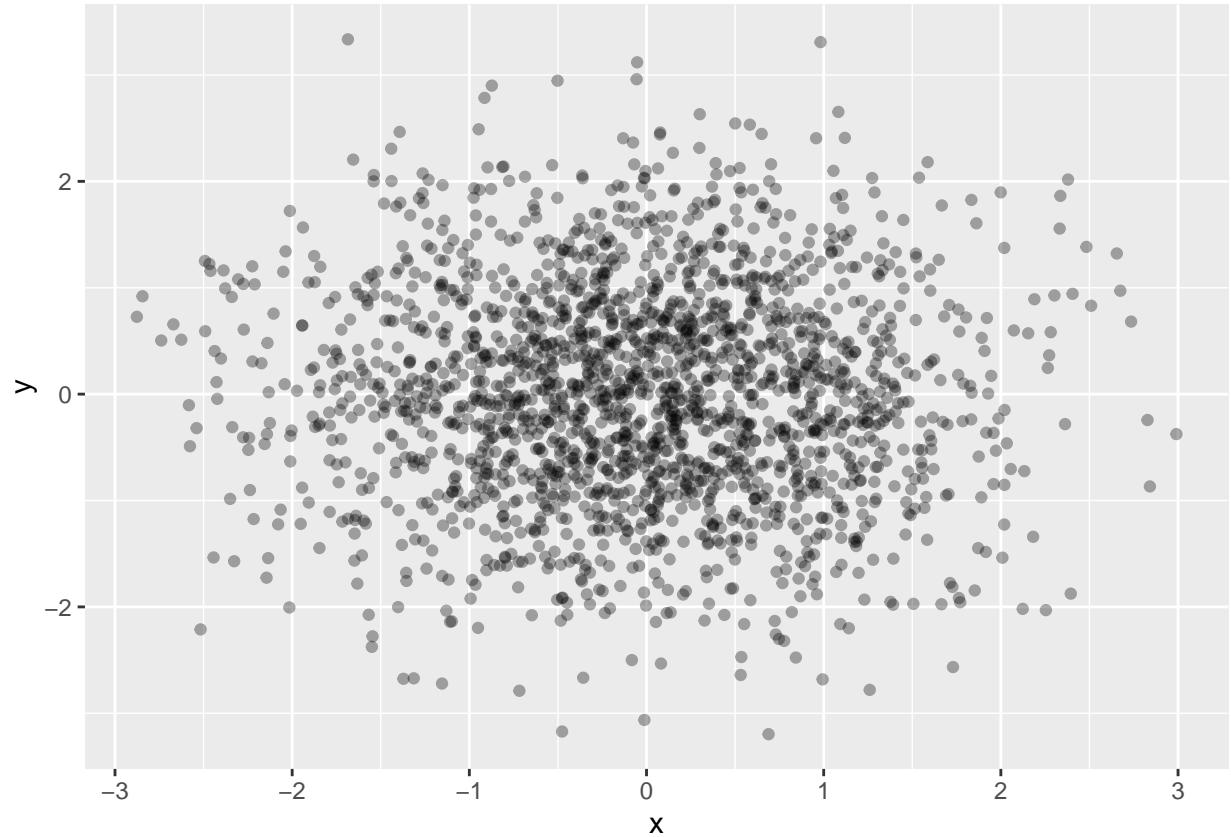


```
norm + geom_point(shape = ".") # Pixel sized
```

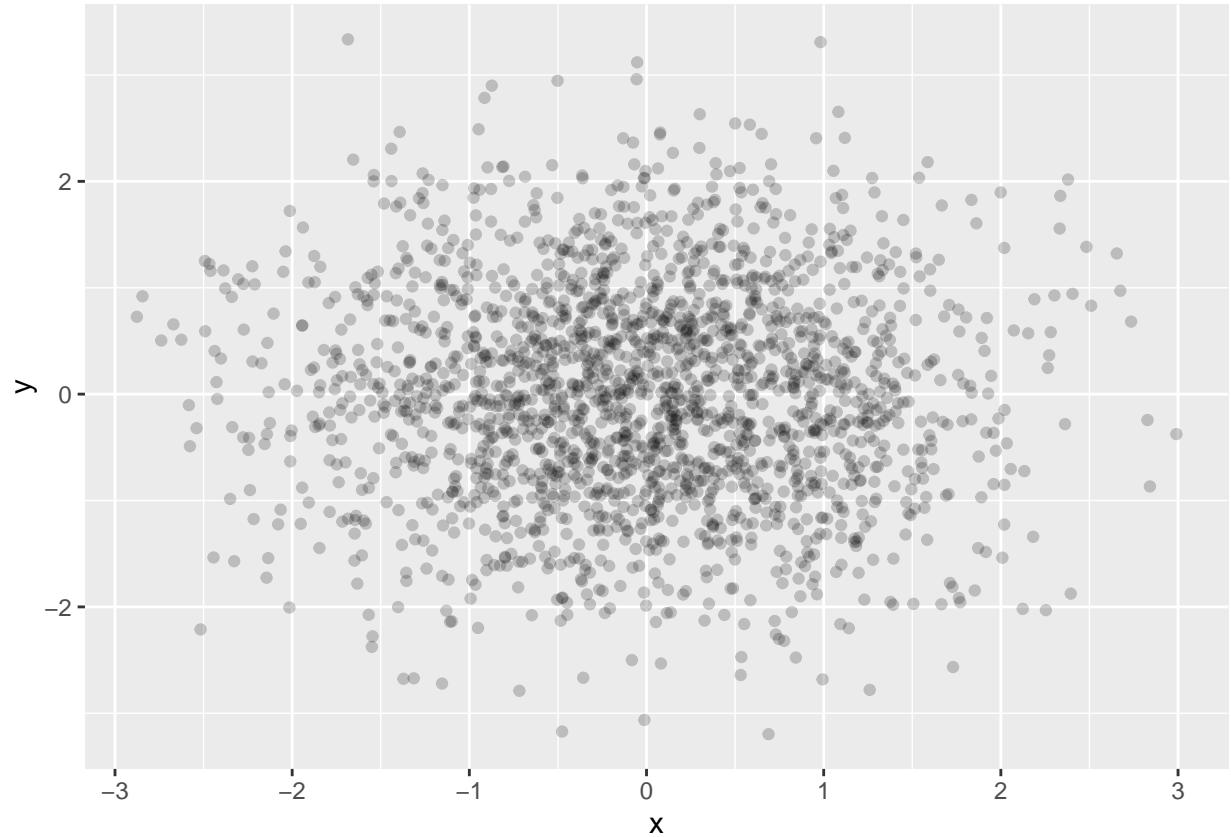


Task: Using alpha blending(transparency) to make the points transparent to see the data more easily for large dataset with overplotting.

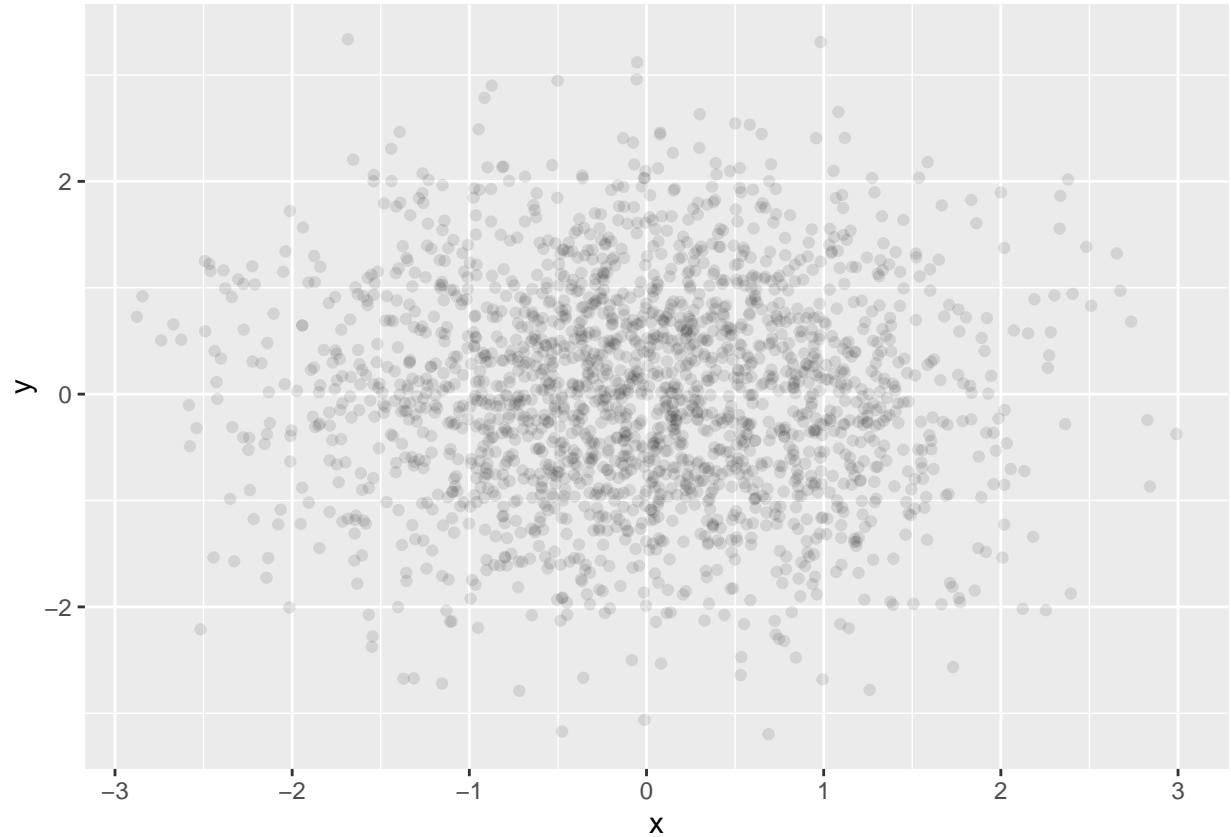
```
norm + geom_point(colour = alpha("black", 1/3))
```



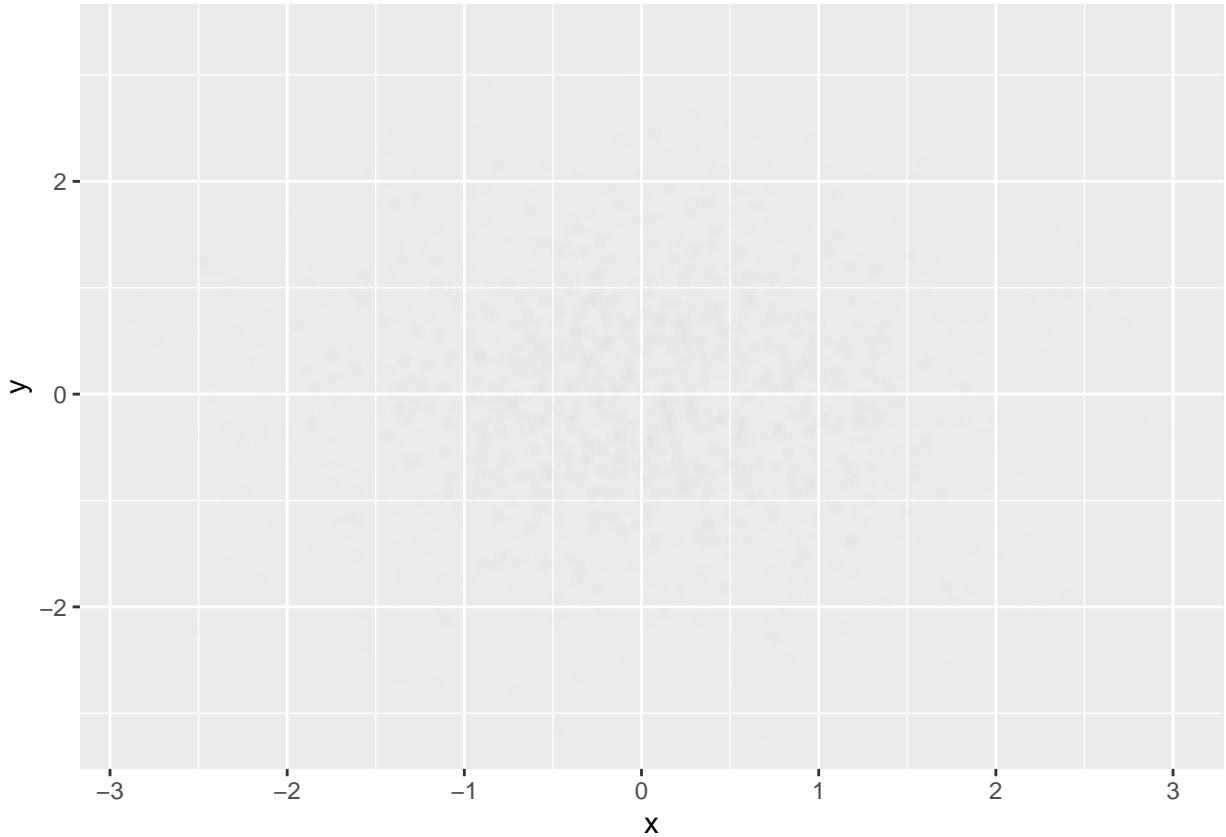
```
norm + geom_point(colour = alpha("black", 1/5))
```



```
norm + geom_point(colour = alpha("black", 1/10))
```

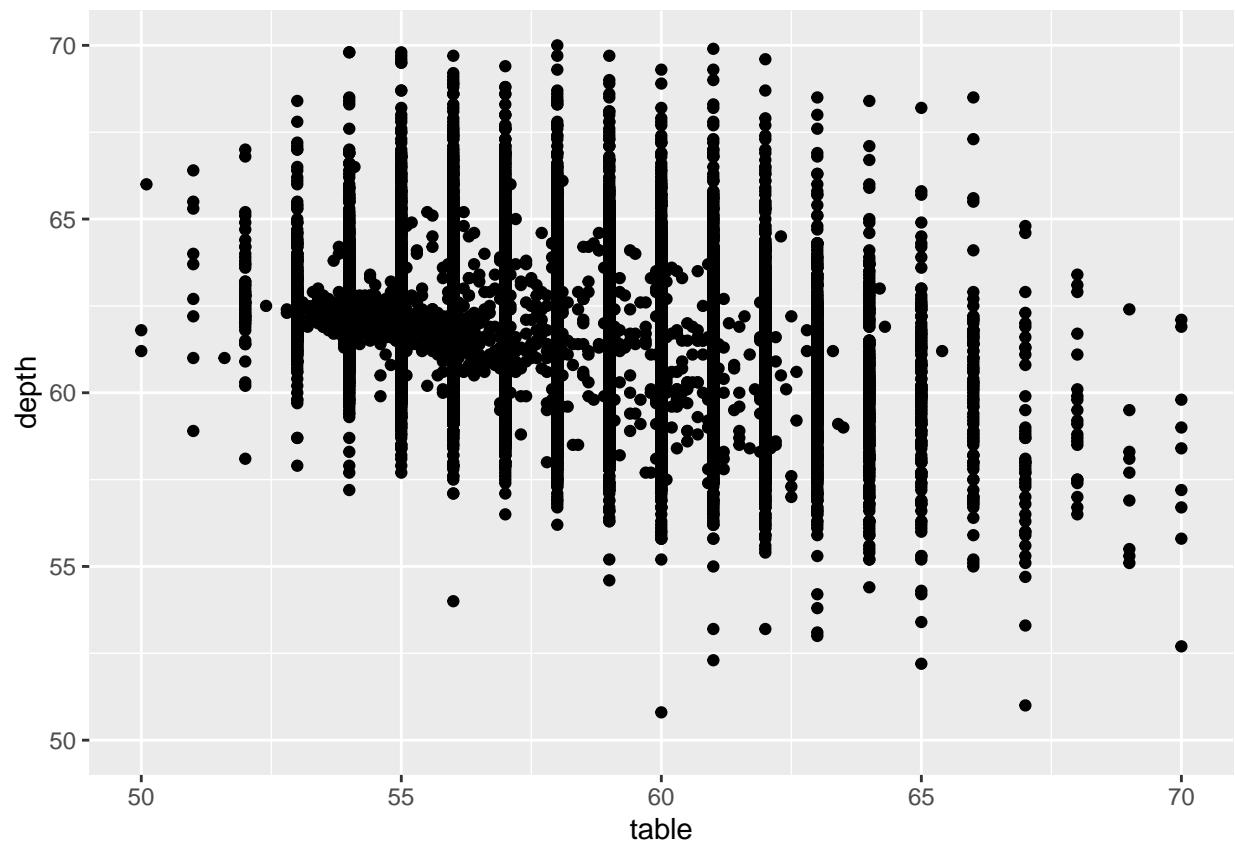


```
norm + geom_point(colour = alpha("black", 1/256)) # you can go till 1/256 transparency which result in
```



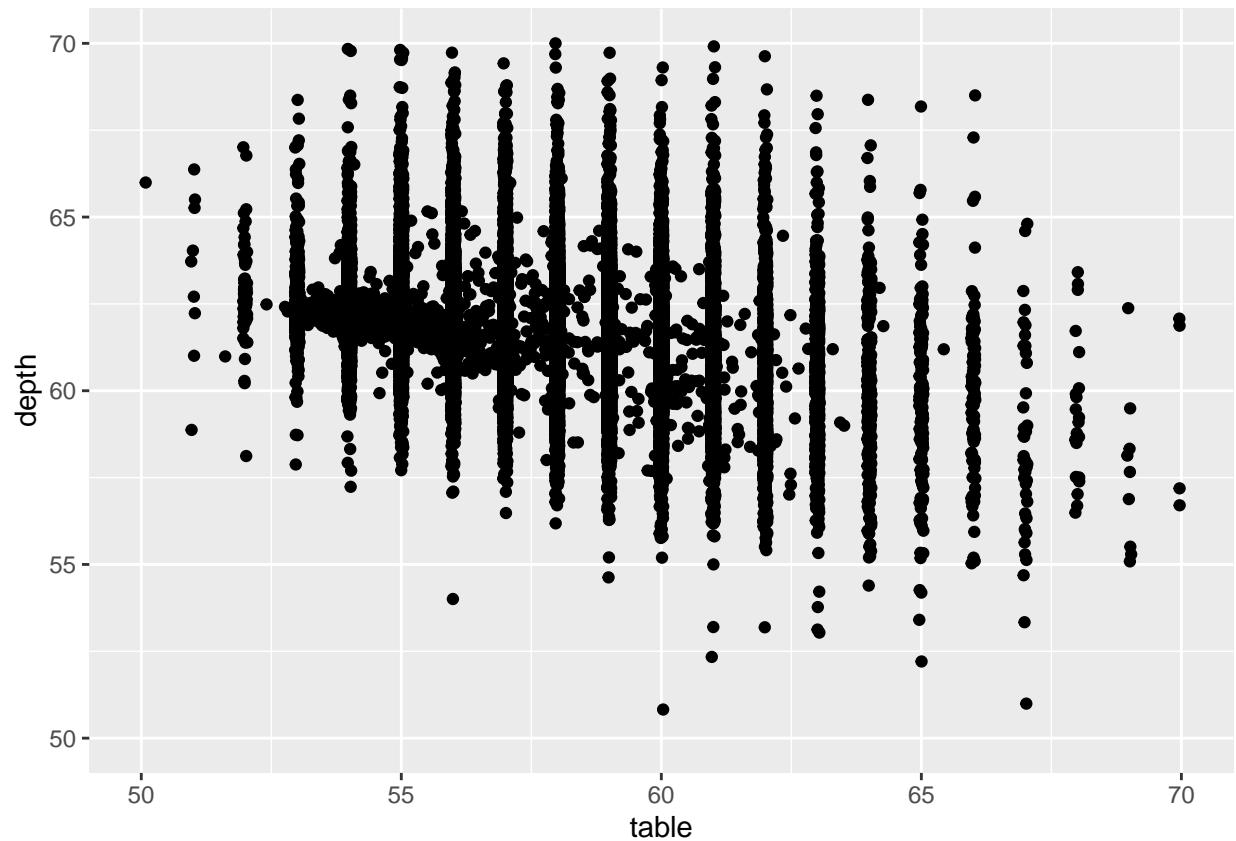
```
td <- ggplot(diamonds, aes(table, depth)) +
  xlim(50, 70) + ylim(50, 70)
td + geom_point()

## Warning: Removed 36 rows containing missing values or values outside the scale range
## ('geom_point()').
```



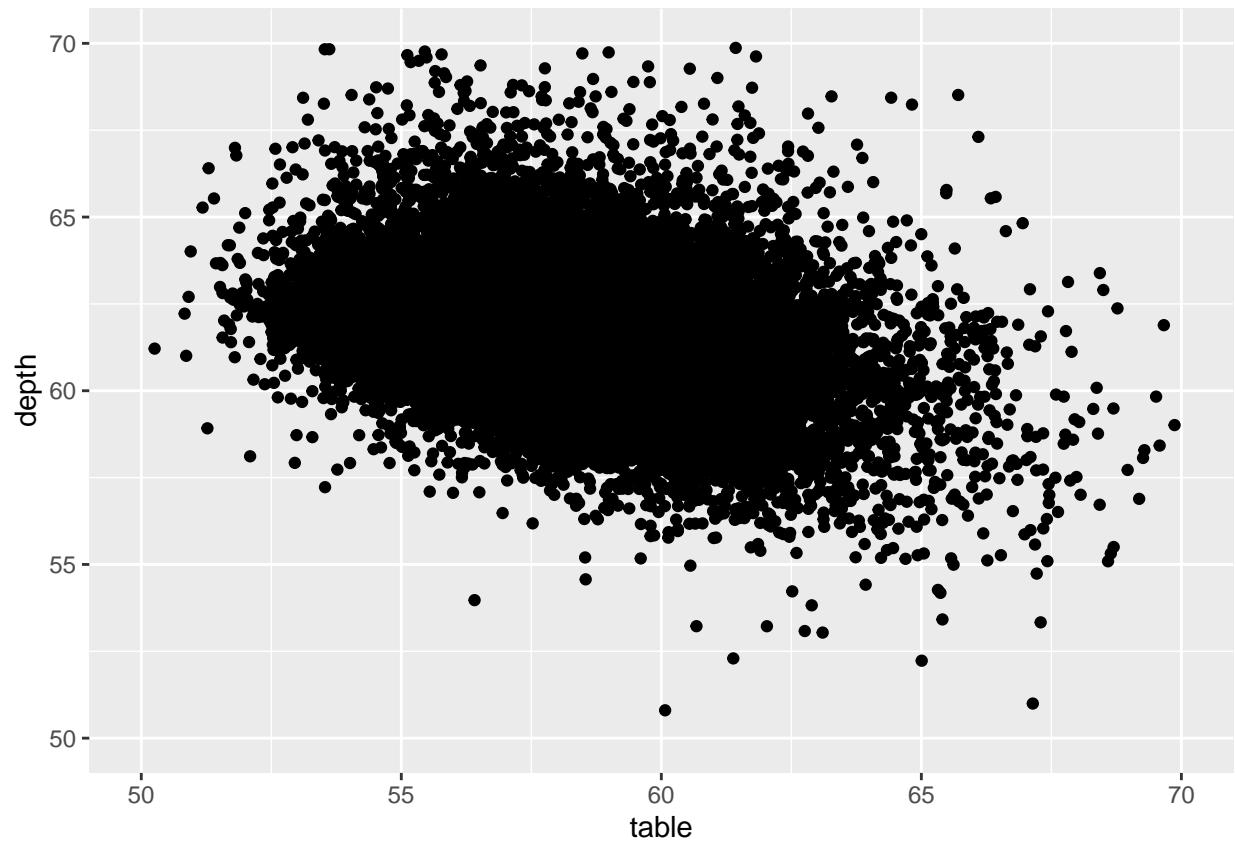
```
td + geom_jitter()
```

```
## Warning: Removed 43 rows containing missing values or values outside the scale range
## ('geom_point()').
```



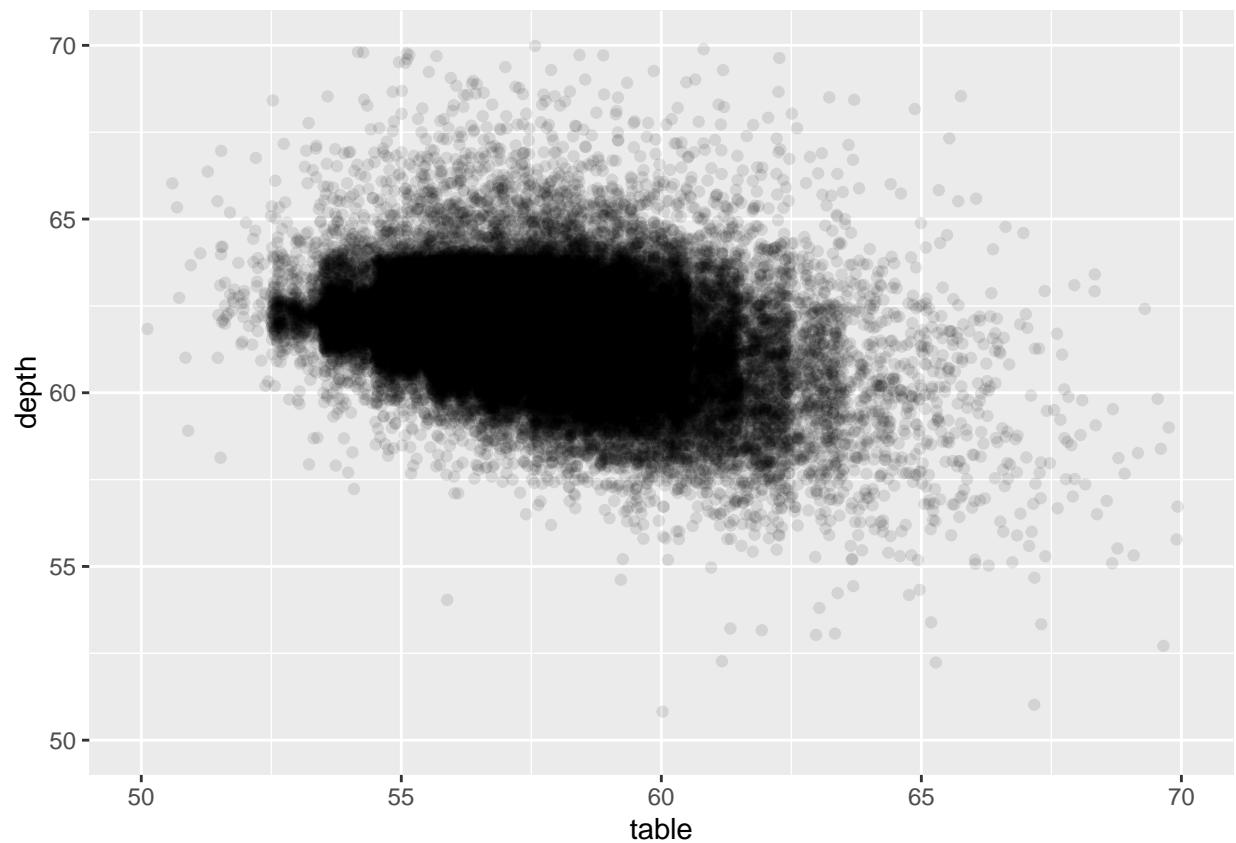
```
jit <- position_jitter(width = 0.5)
td + geom_jitter(position = jit)
```

```
## Warning: Removed 44 rows containing missing values or values outside the scale range
## ('geom_point()').
```



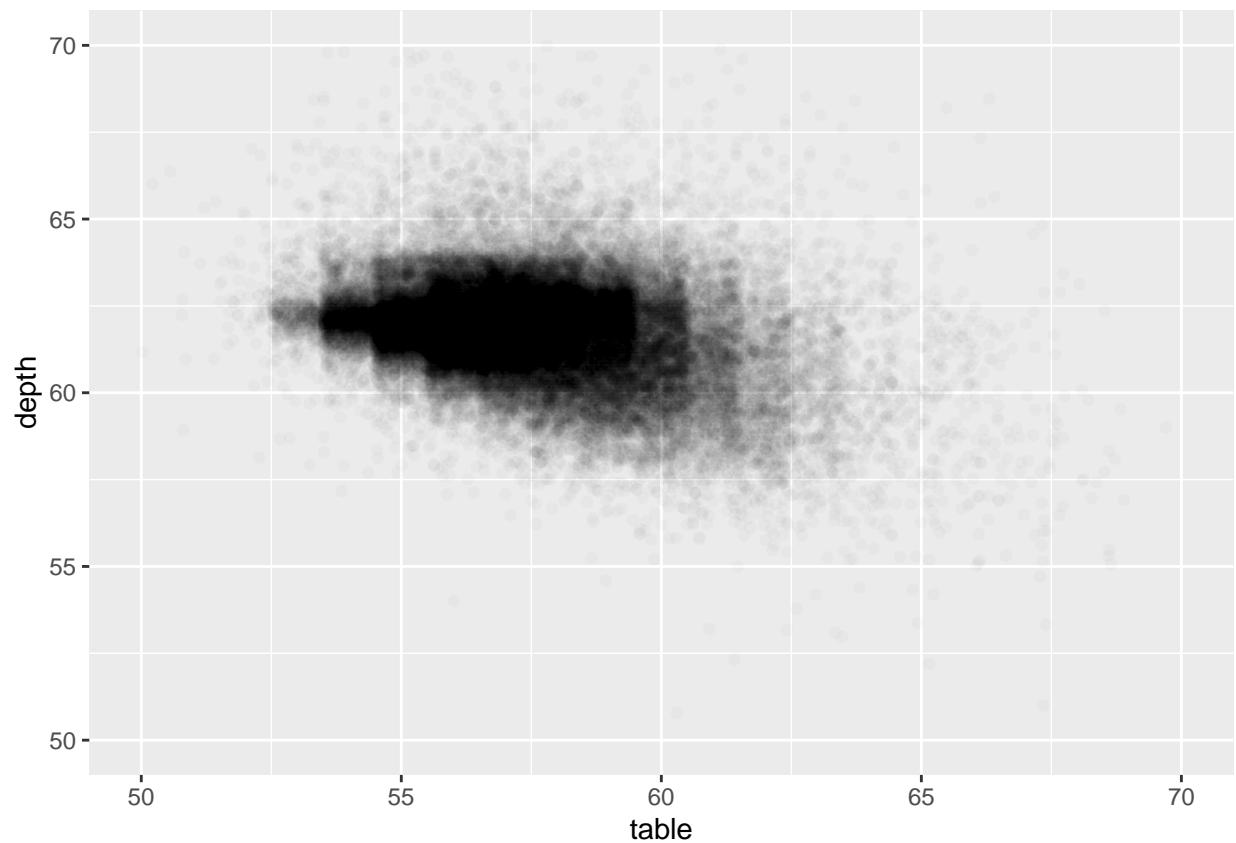
```
td + geom_jitter(position = jit, colour = alpha("black", 1/10))
```

```
## Warning: Removed 40 rows containing missing values or values outside the scale range
## ('geom_point()').
```



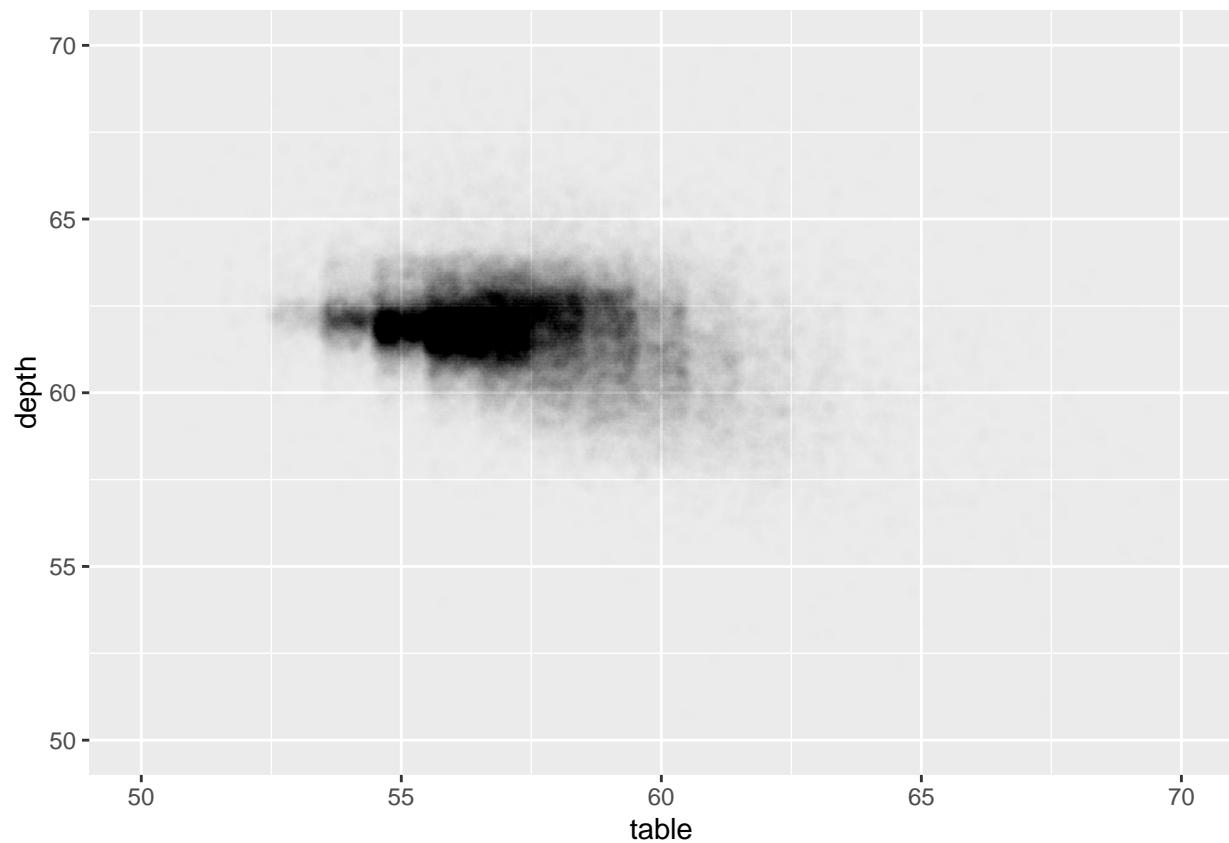
```
td + geom_jitter(position = jit, colour = alpha("black", 1/50))
```

```
## Warning: Removed 45 rows containing missing values or values outside the scale range
##   ('geom_point()').
```



```
td + geom_jitter(position = jit, colour = alpha("black", 1/200))
```

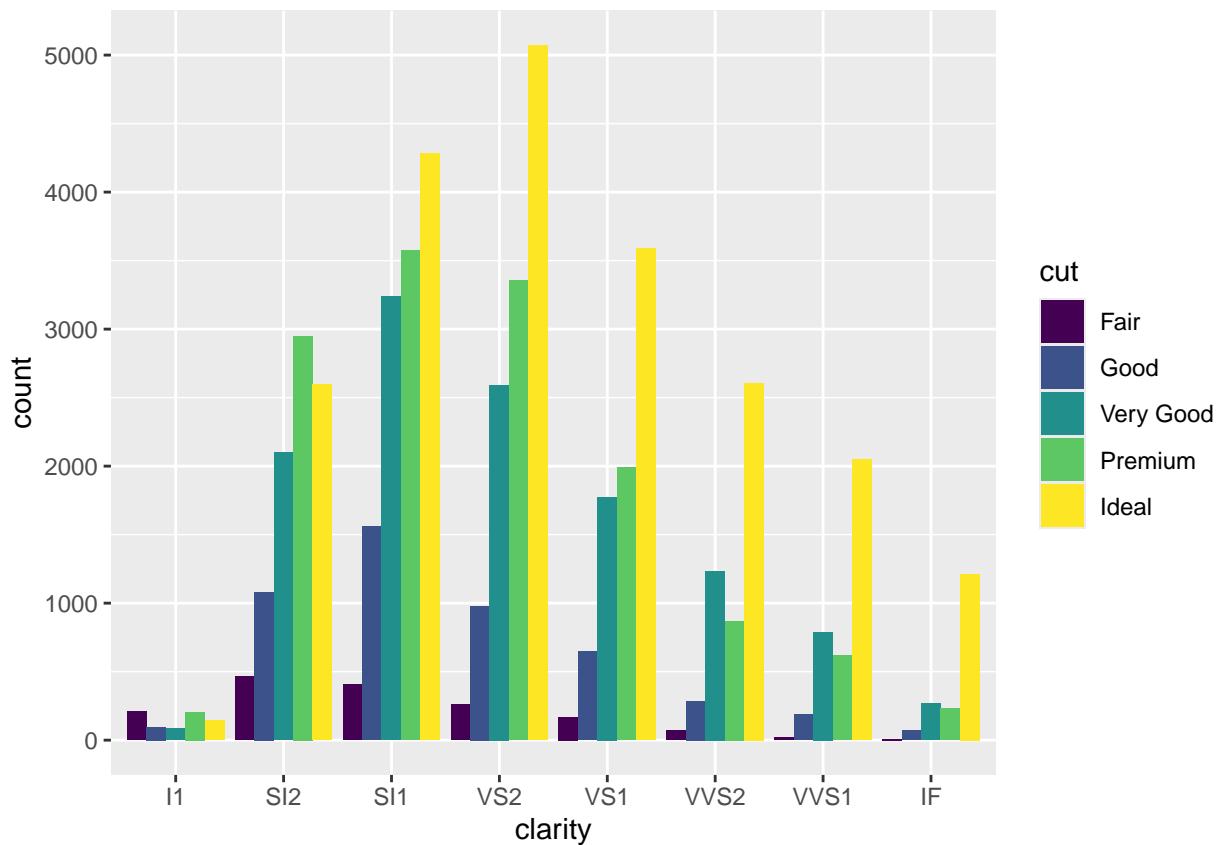
```
## Warning: Removed 42 rows containing missing values or values outside the scale range
## ('geom_point()').
```



## Position adjustments

### Dodge

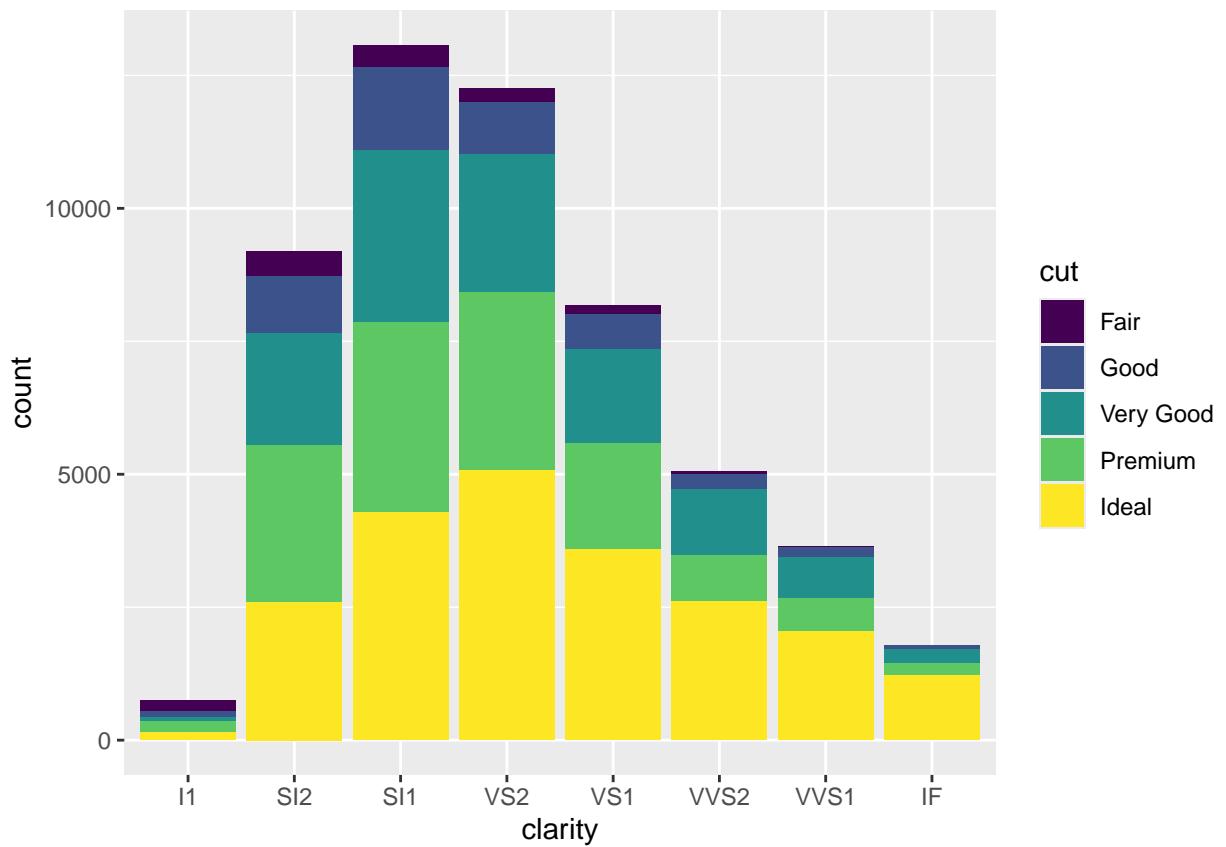
```
depth_dist <- ggplot(diamonds, aes(clarity))
depth_dist + geom_bar(aes(fill = cut), position = "dodge")
```



Interpretation: We use Dodge to adjust position side by side to compare the values

### Stack

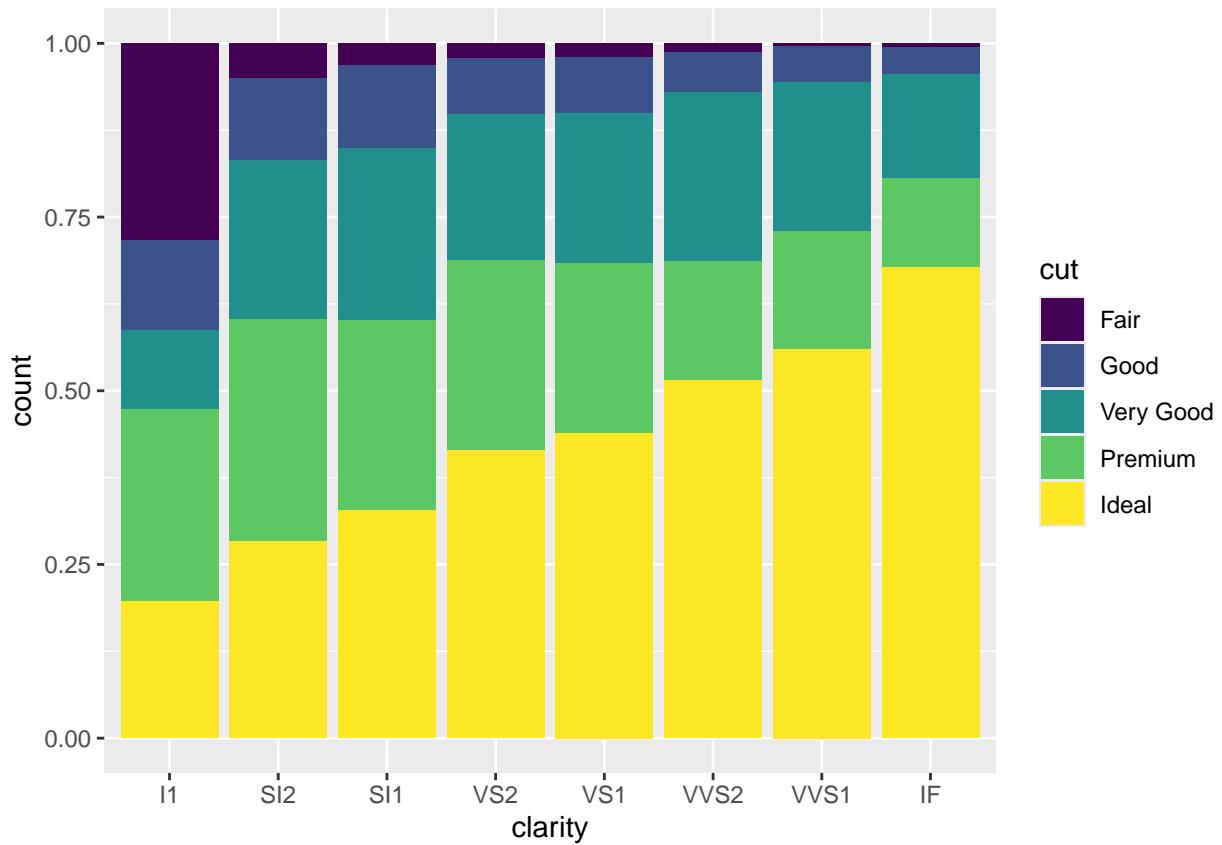
```
depth_dist <- ggplot(diamonds, aes(clarity))
depth_dist + geom_bar(aes(fill = cut), position = "stack")
```



Interpretation: We use Stack to overlap objects on top of one another to see the total height and see which clarity has the highest count in above graph

Fill

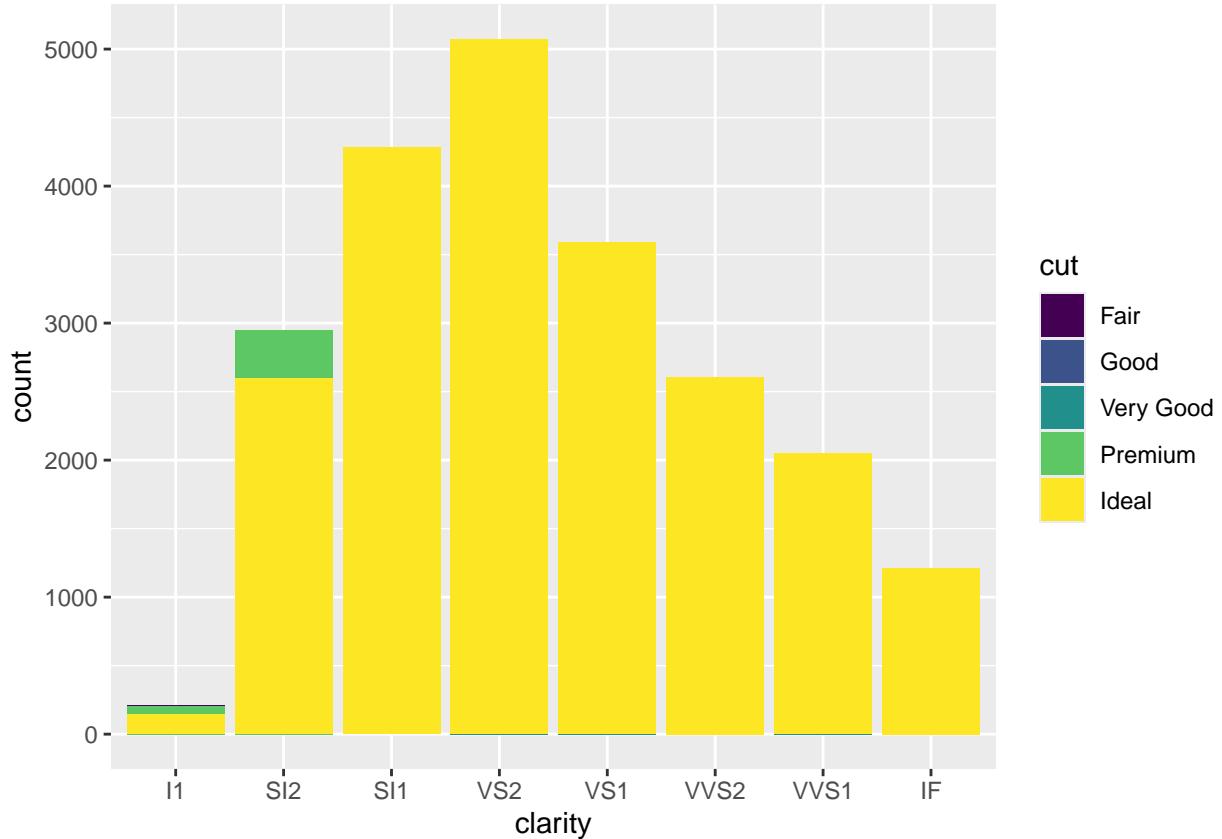
```
depth_dist <- ggplot(diamonds, aes(clarity))
depth_dist + geom_bar(aes(fill = cut), position = "fill")
```



Interpretation: We use Fill to stack for overlapping objects on top of one another and use for comparison of the distribution of cuts within each clarity level in above graph

## Identity

```
depth_dist <- ggplot(diamonds, aes(clarity))
depth_dist + geom_bar(aes(fill = cut), position = "identity")
```



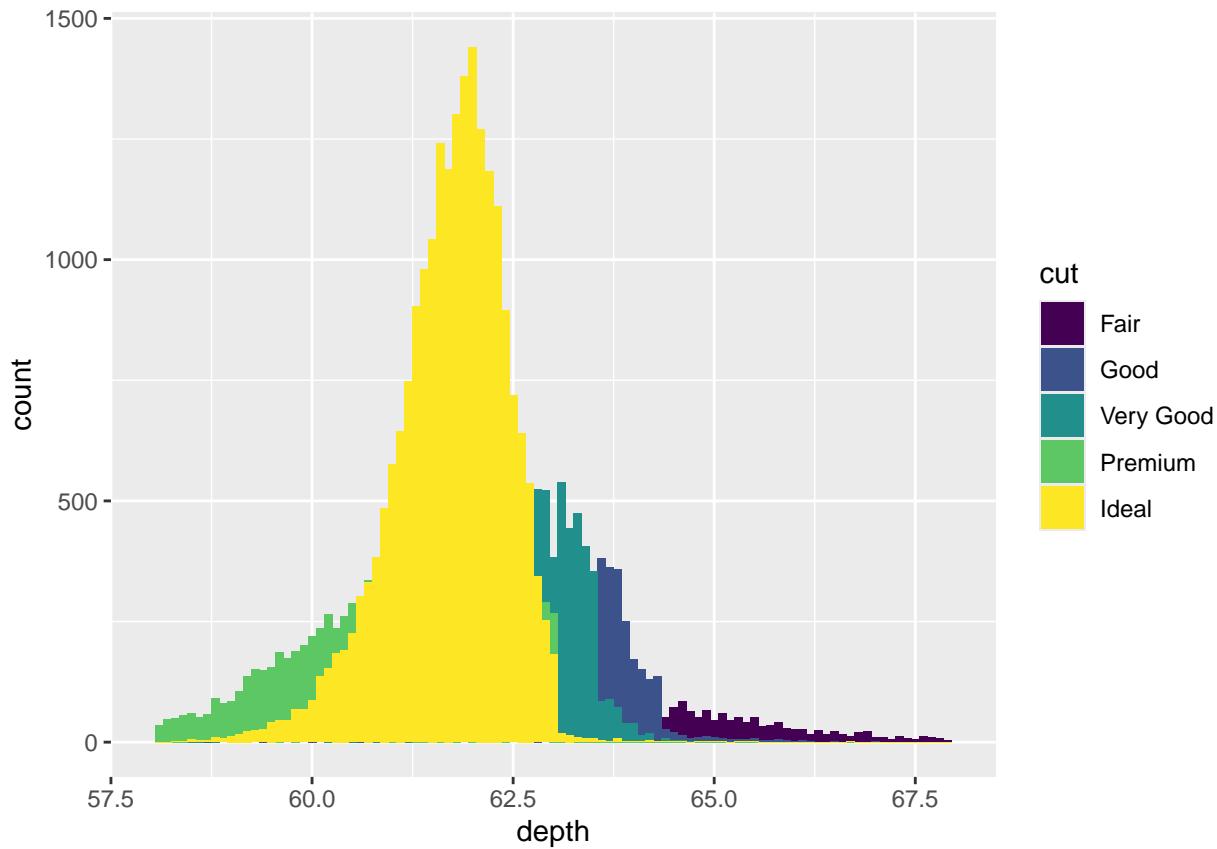
Interpretation: We use Identity will be plotted directly on top of each other for the same x-axis value. In above case all the cut has been plotted in same x-axis but in s12 clarity the the number of premium is more than ideal so it has come up than the ideal bar.

Task: Ploting the same above code but changing it to histogram and its binwidth to 0.1

```
depth_dist <- ggplot(diamonds, aes(depth)) + xlim(58, 68)
depth_dist + geom_histogram(aes(fill = cut), binwidth = 0.1,
  position = "identity")

## Warning: Removed 669 rows containing non-finite outside the scale range
## ('stat_bin()').

## Warning: Removed 10 rows containing missing values or values outside the scale range
## ('geom_bar()').
```

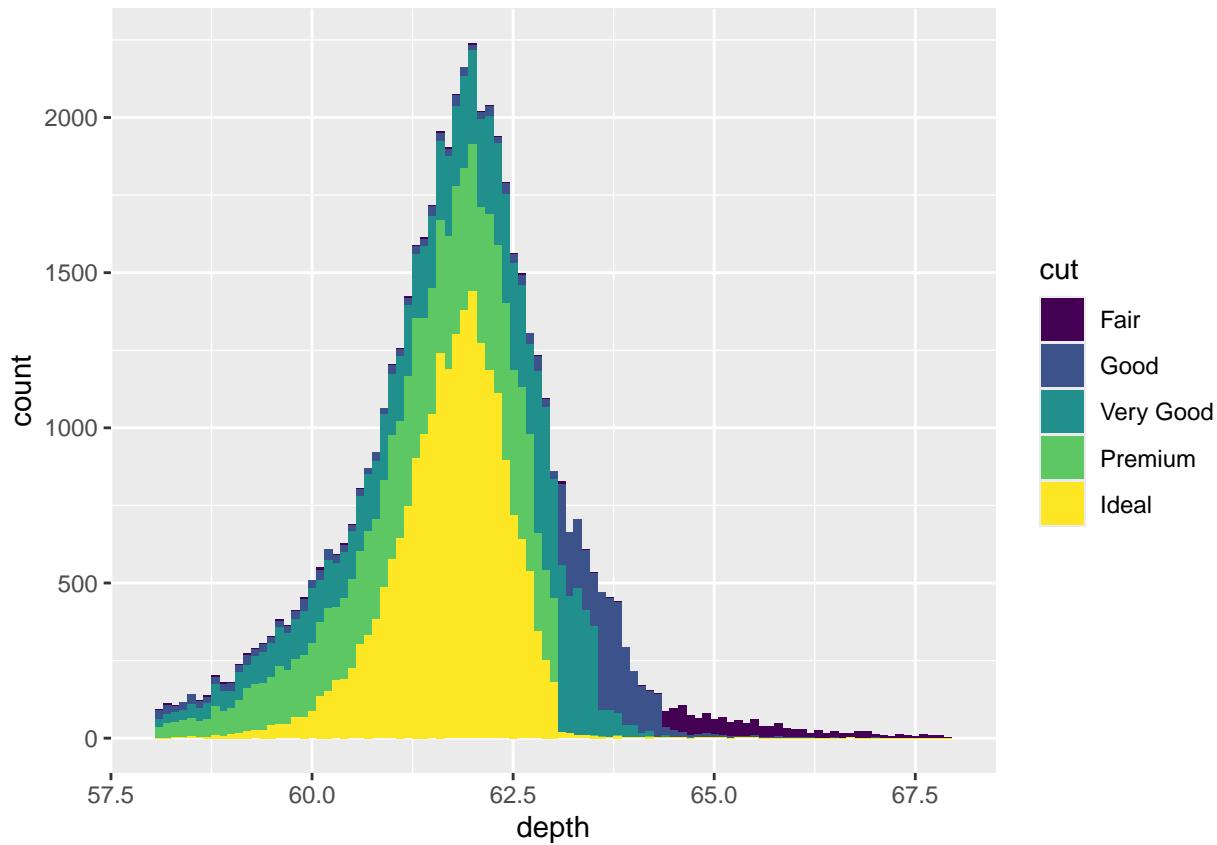


Task: Using the histogram to plot the depth in x and count in y with cut in stack and using binwidth of 0.1

```
depth_dist <- ggplot(diamonds, aes(depth)) + xlim(58, 68)
depth_dist + geom_histogram(aes(fill = cut), binwidth = 0.1,
                           position = "stack")
```

```
## Warning: Removed 669 rows containing non-finite outside the scale range
## ('stat_bin()').

## Warning: Removed 10 rows containing missing values or values outside the scale range
## ('geom_bar()').
```

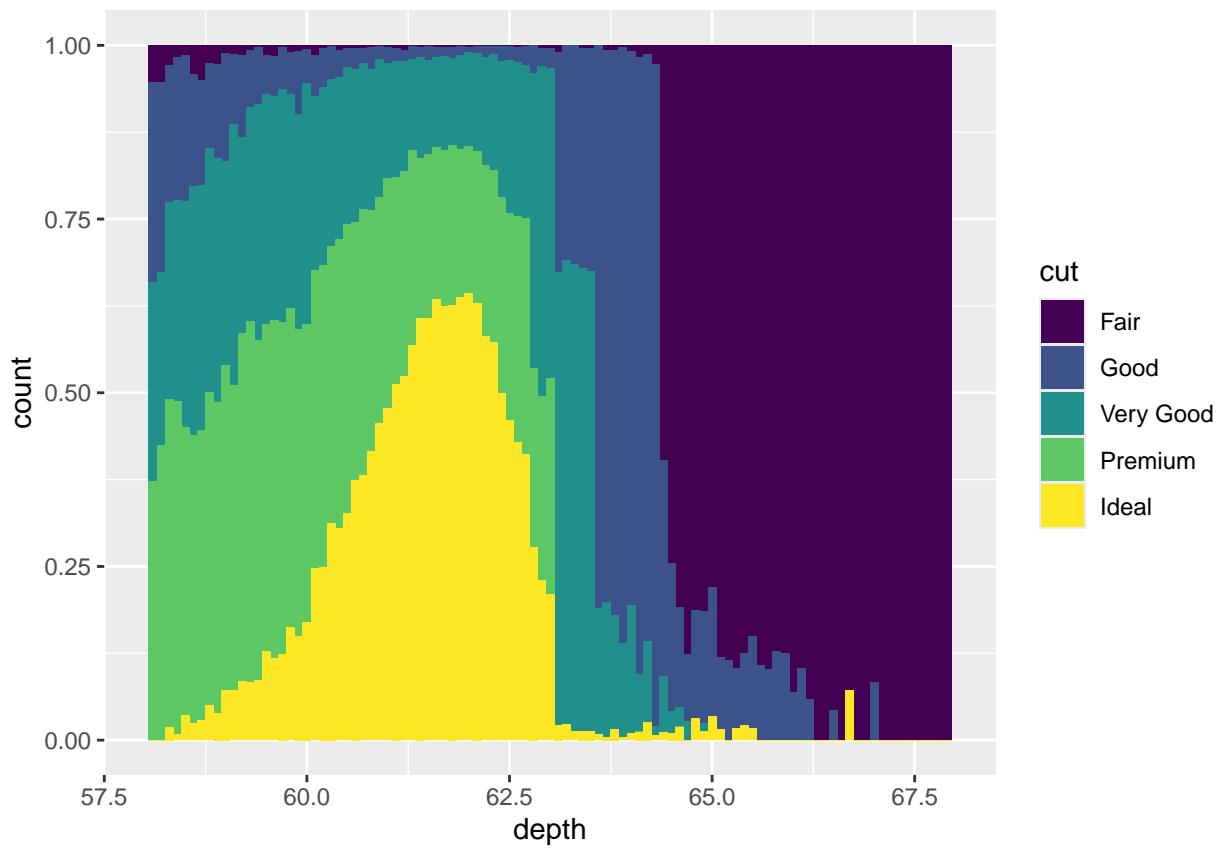


Task: Displaying the histogram with position fill and bindwith to 0.1

```
depth_dist <- ggplot(diamonds, aes(depth)) + xlim(58, 68)
depth_dist + geom_histogram(aes(fill = cut), binwidth = 0.1,
position = "fill")
```

```
## Warning: Removed 669 rows containing non-finite outside the scale range
## ('stat_bin()').
```

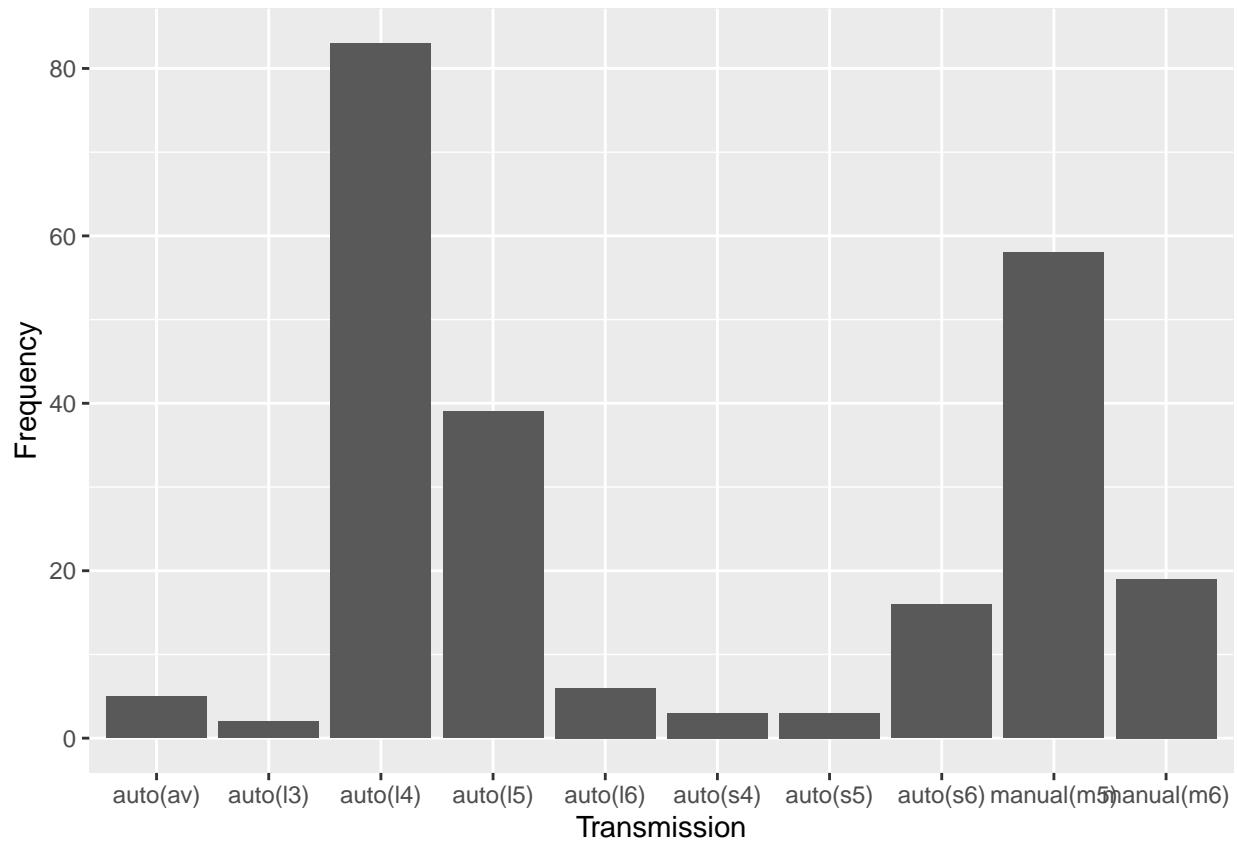
```
## Warning: Removed 10 rows containing missing values or values outside the scale range
## ('geom_bar()').
```



## Overlapping labels

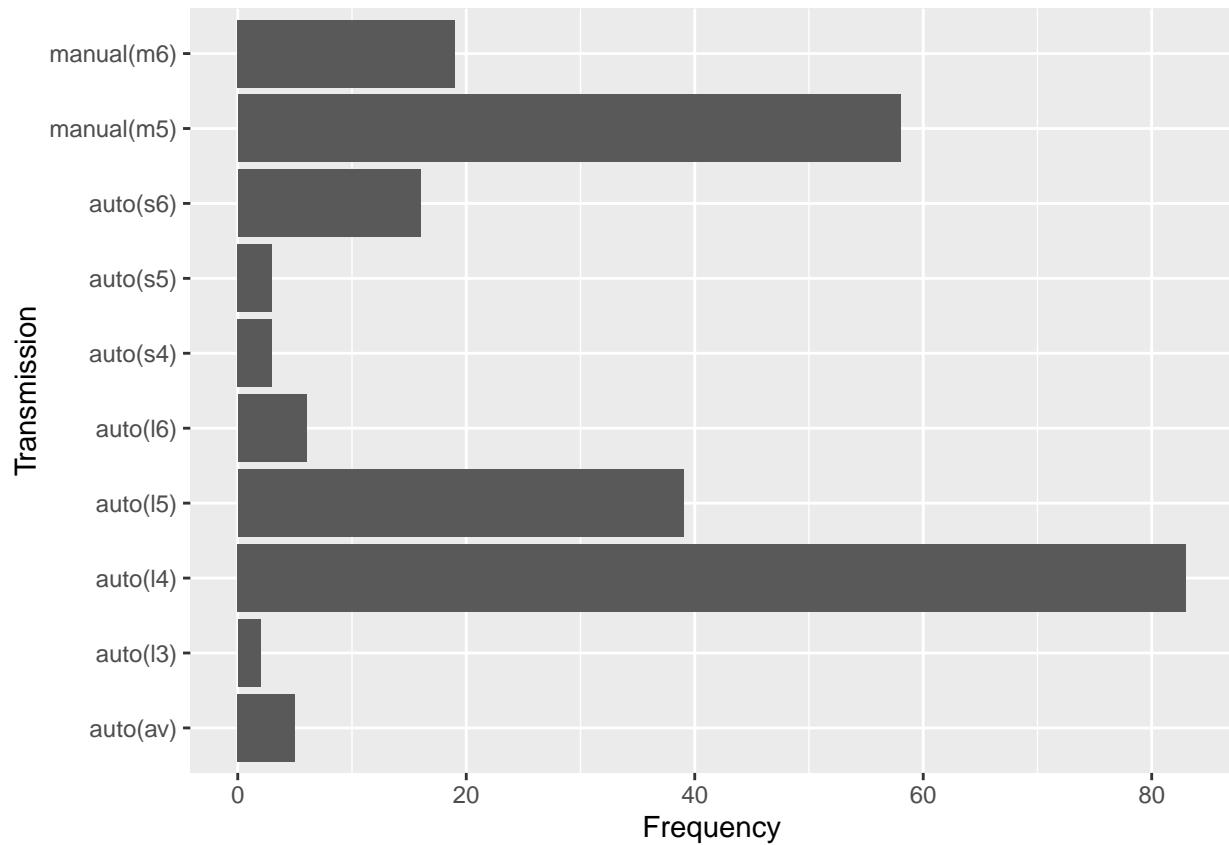
Task: Displaying overlapping labels in categorical variables

```
ggplot(mpg, aes(x = trans)) +  
  geom_bar() +  
  labs(x = "Transmission",  
       y = "Frequency")
```



Task: Flipping the above bar graph to deal with overlapping issue.

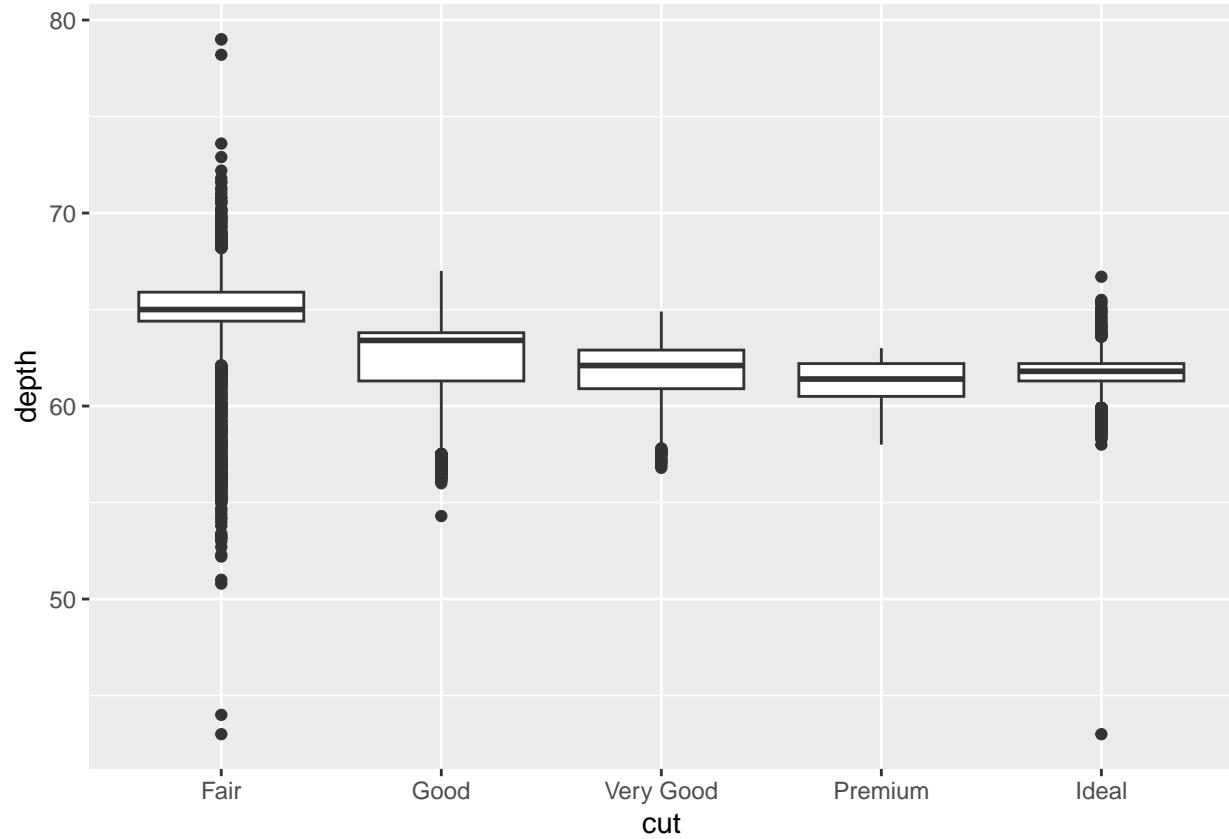
```
# horizontal bar chart
ggplot(mpg, aes(x = trans)) +
  geom_bar() +
  labs(x = "Transmission",
       y = "Frequency") +
  coord_flip()
```



## More Basics plots

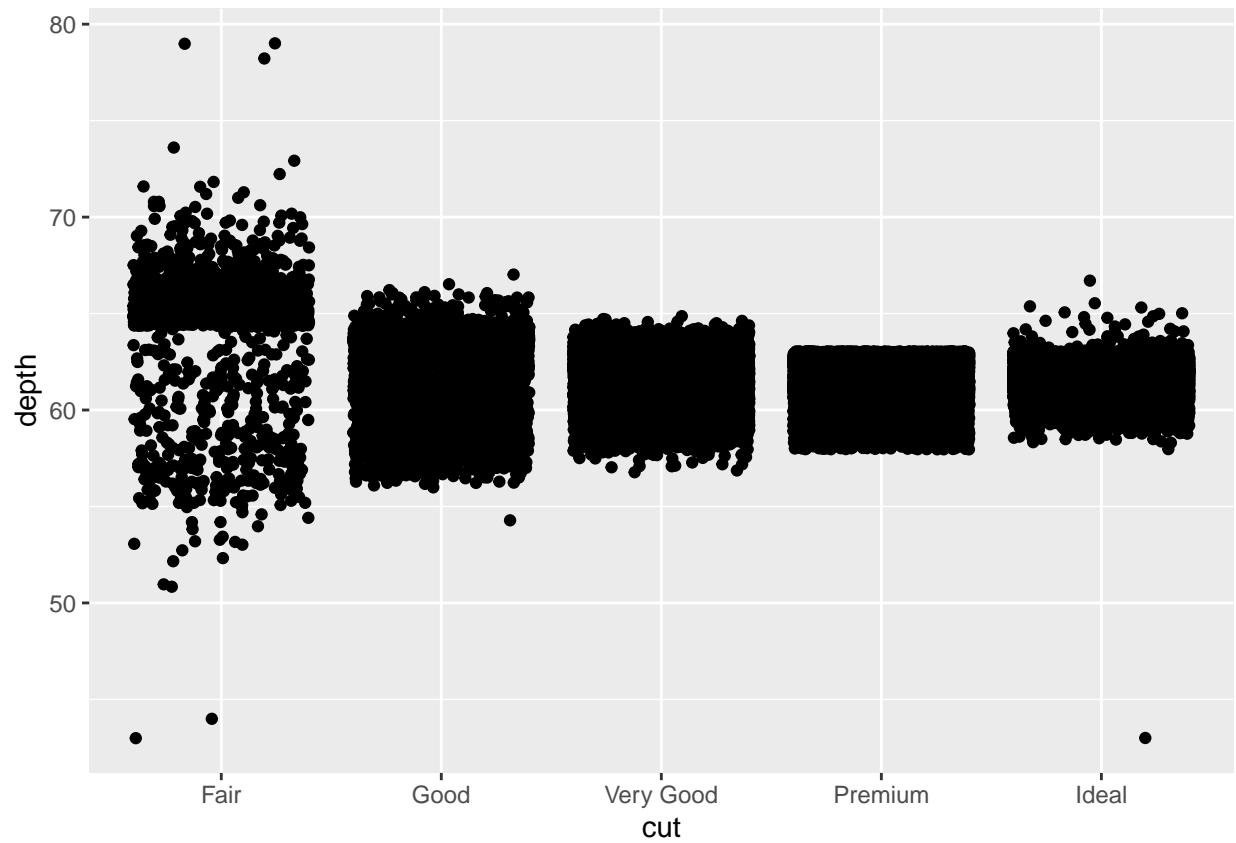
Task: Creating boxplot graph

```
ggplot(diamonds,aes(x = cut, y = depth)) +  
  geom_boxplot()
```



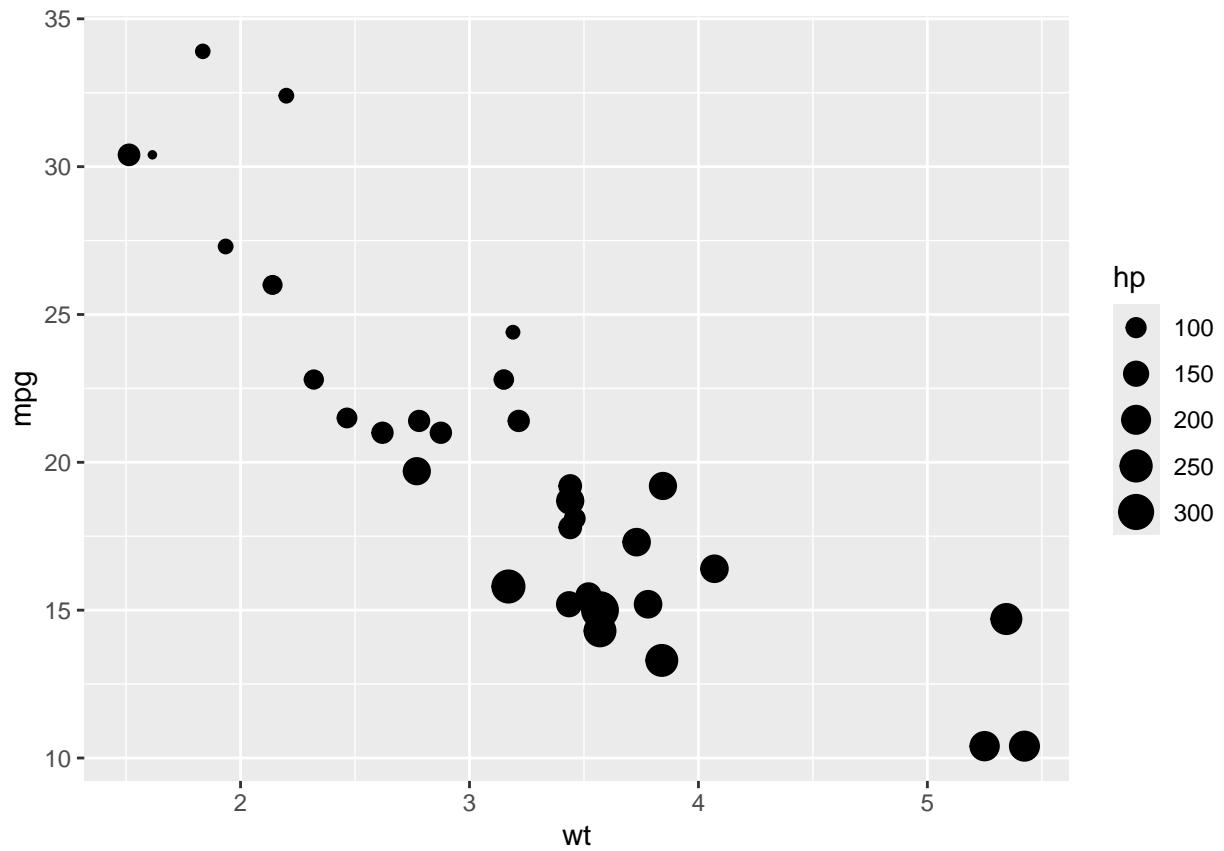
Task: Creating jitter graph

```
ggplot(diamonds,  
       aes(x = cut,  
            y = depth)) +  
  geom_jitter()
```



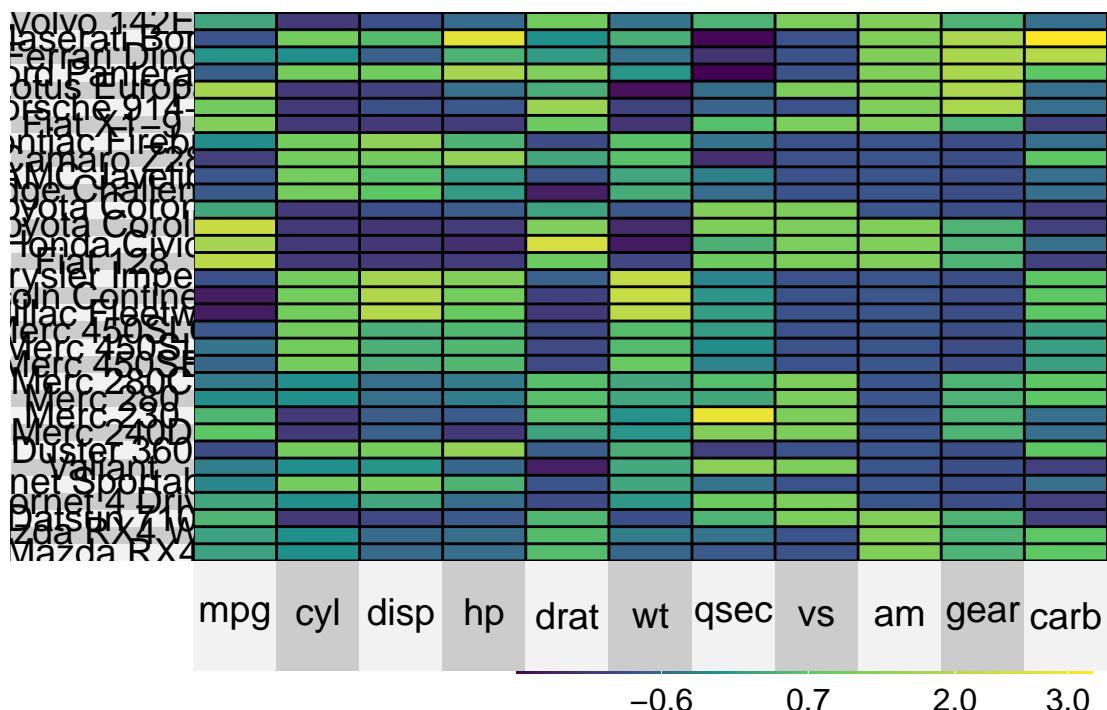
Task: Creating a bubble plot

```
data(mtcars)
ggplot(mtcars,
aes(x = wt, y = mpg, size = hp)) +
geom_point()
```



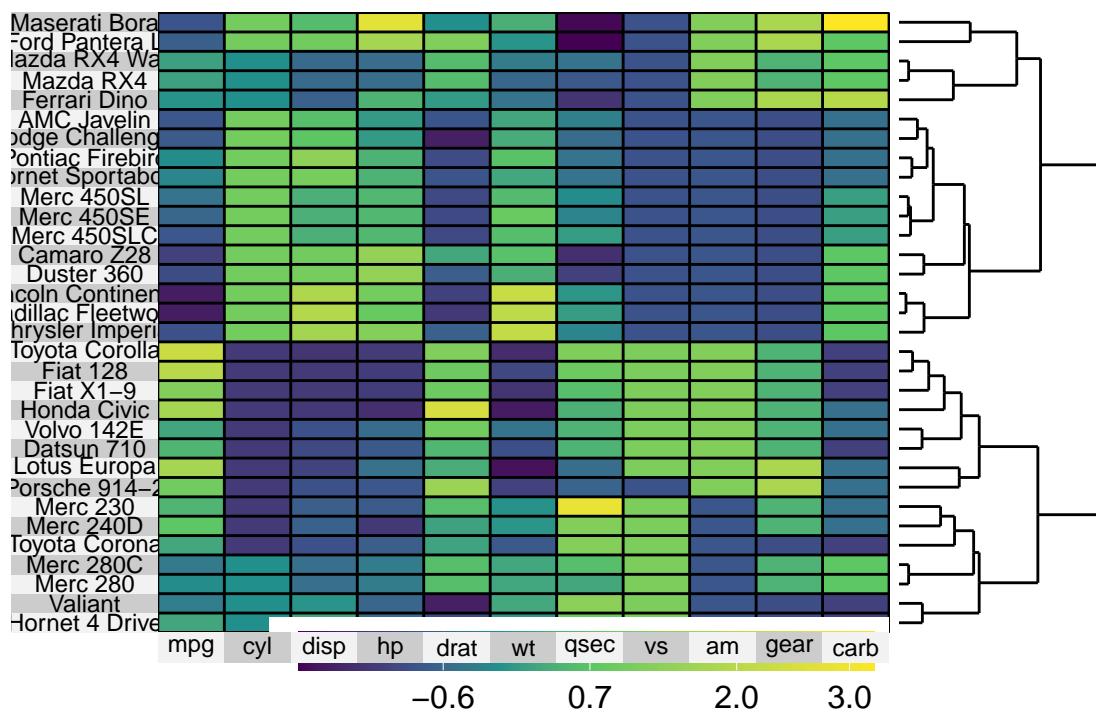
Task: Creating superheat graph

```
library(superheat)
superheat(mtcars, scale = TRUE)
```



Task: Creating sorted heat map

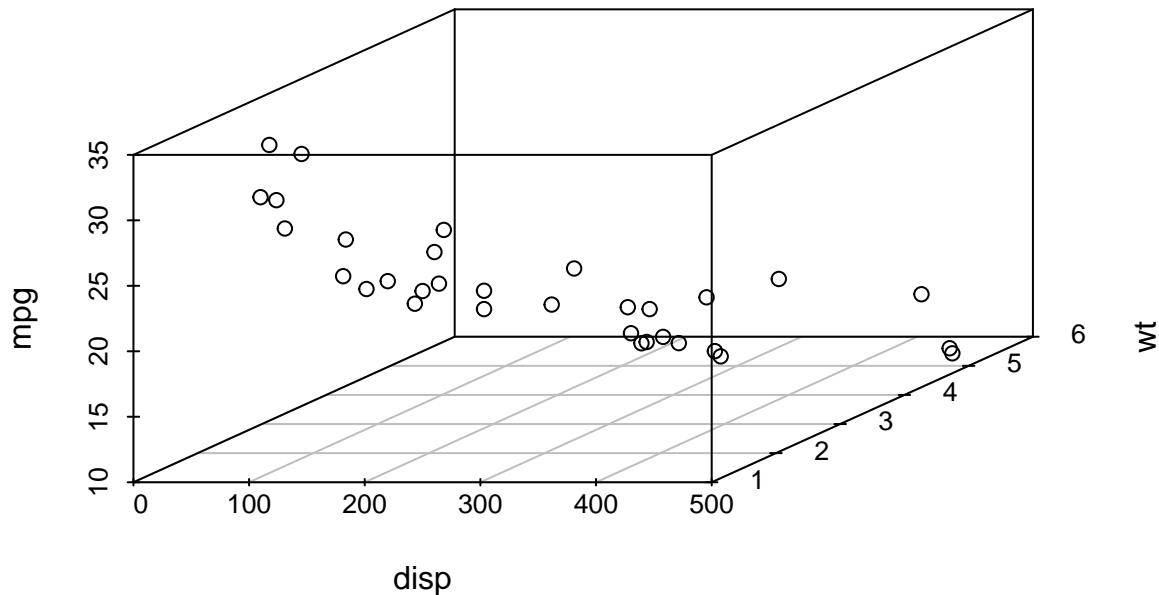
```
superheat(mtcars,
  scale = TRUE,
  left.label.text.size=3,
  bottom.label.text.size=3,
  bottom.label.size = .05,
  row.dendrogram = TRUE )
```



Task: Creating Basic 3-D scatterplot

```
library(scatterplot3d)
with(mtcars, { scatterplot3d(x = disp,
y = wt,
z = mpg,
main="3-D Scatterplot Example 1")
})
```

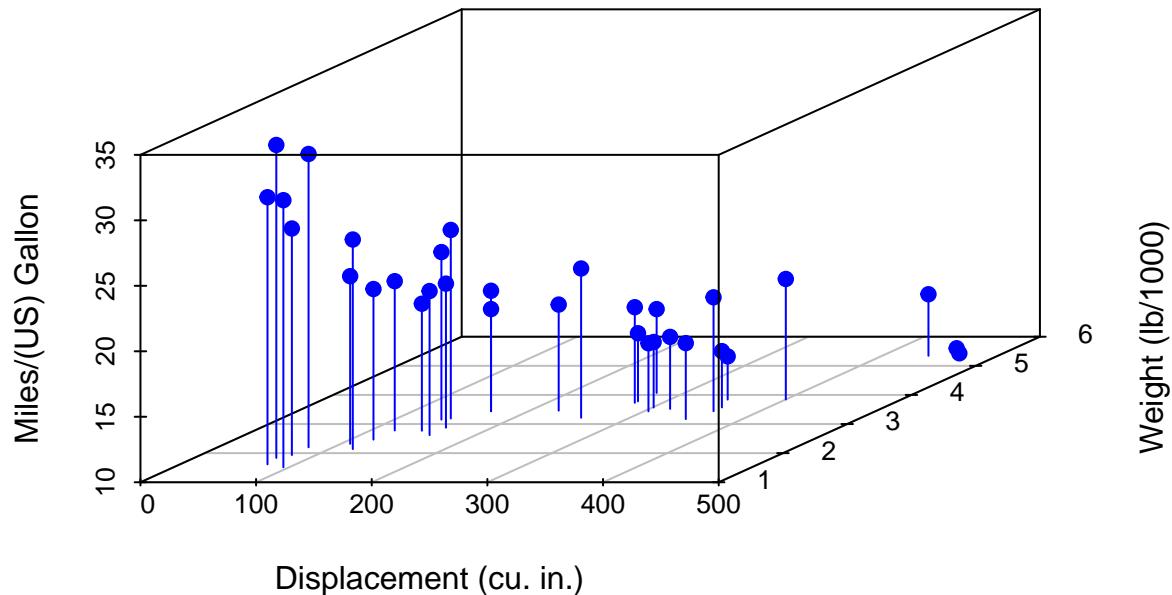
## 3-D Scatterplot Example 1



Task: Creating 3-D scatterplot graph with blue circles and a line to the horizontal plane

```
with(mtcars, {  
  scatterplot3d(x = disp,  
    y = wt,  
    z = mpg,  
    color="blue", pch=19,  
    type = "h",  
    main = "3-D Scatterplot Example 2",  
    xlab = "Displacement (cu. in.)",  
    ylab = "Weight (lb/1000)",  
    zlab = "Miles/(US) Gallon")  
})
```

## 3-D Scatterplot Example 2

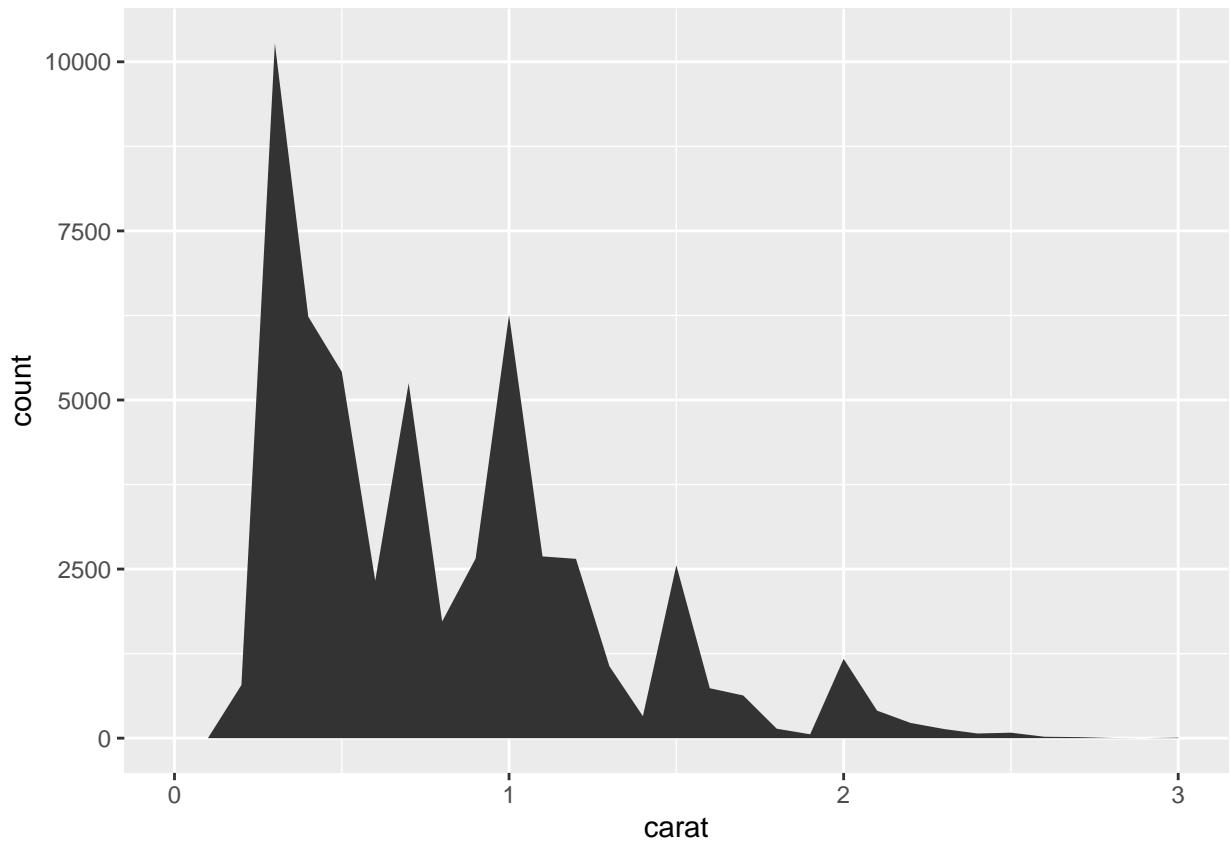


```
d <- ggplot(diamonds, aes(carat)) + xlim(0, 3)
d + stat_bin(aes(ymax = ..count..), binwidth = 0.1, geom = "area")

## Warning in stat_bin(aes(ymax = ..count..), binwidth = 0.1, geom = "area"):
## Ignoring unknown aesthetics: ymax

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

## Warning: Removed 32 rows containing non-finite outside the scale range
## ('stat_bin()').
```



## Part-II