Team 5
Yifan Feng: 2671027
Yunxiang Li: 2674844

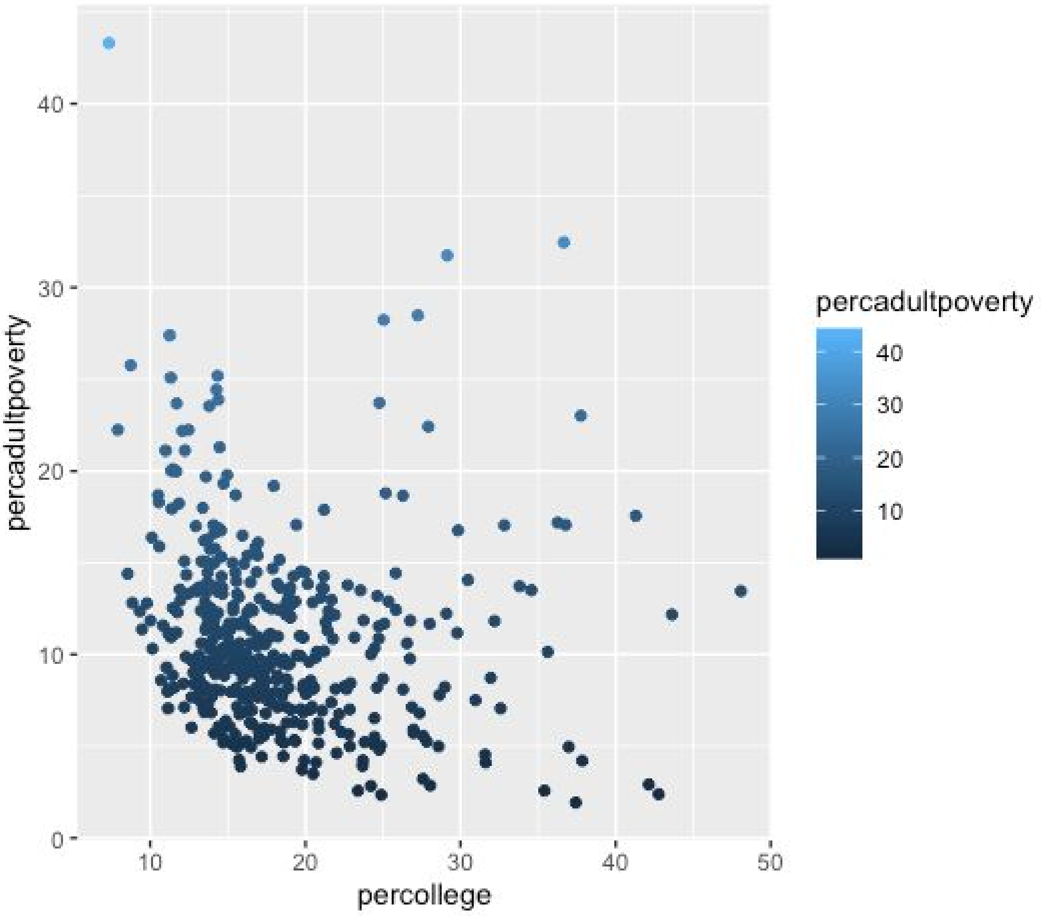
Assignment 1a:

1a)

```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty))
```

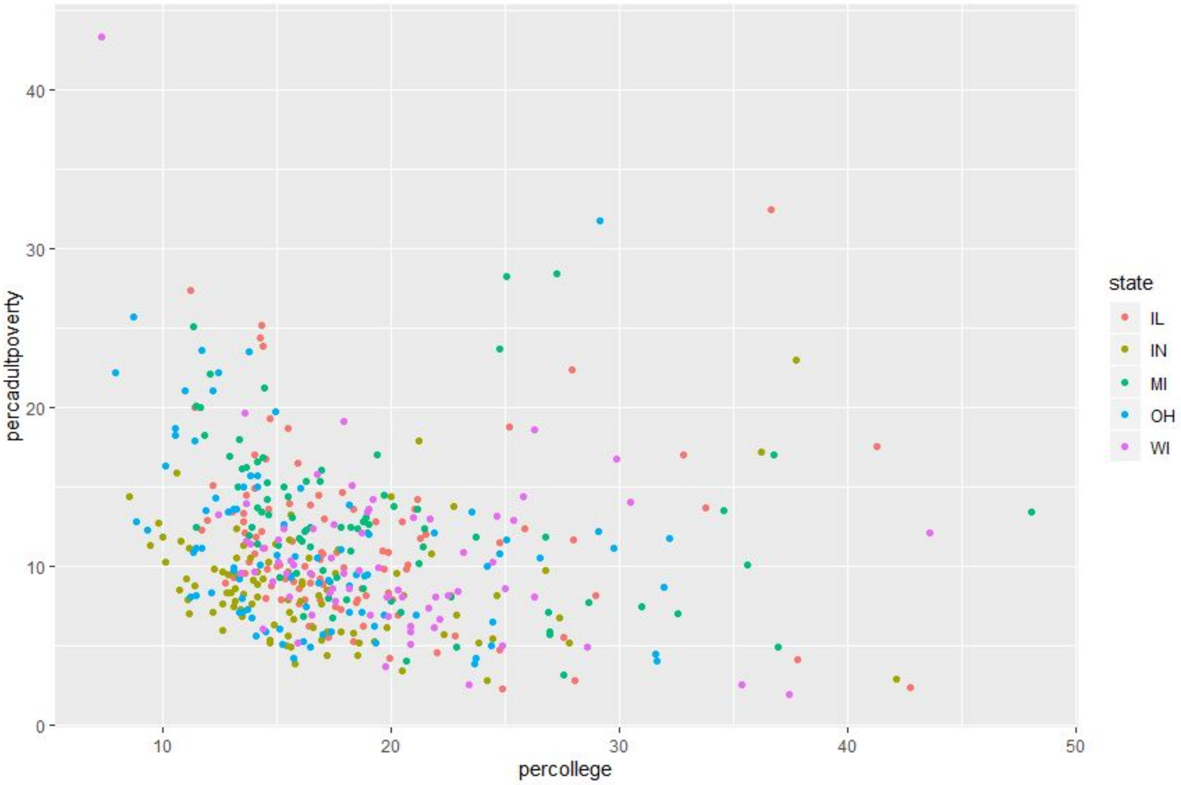
1b)

```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty,color=
percadultpoverty))
```



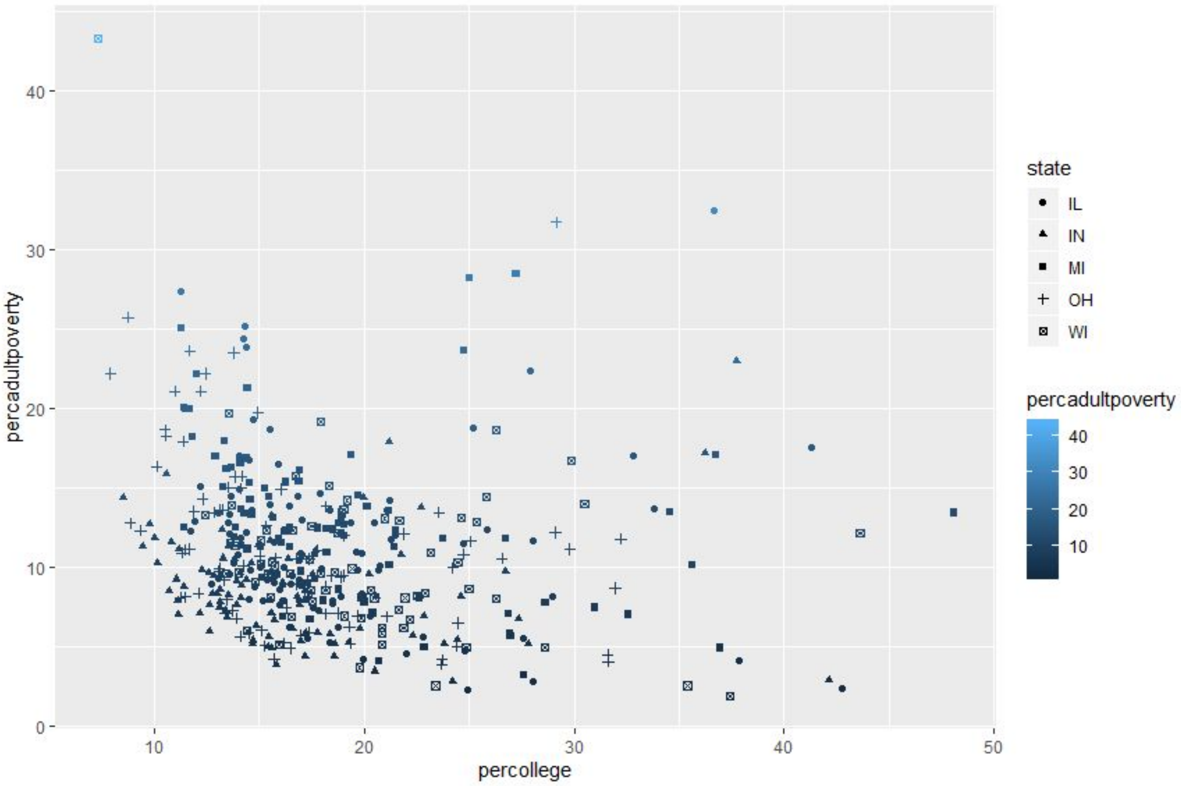
1c)

```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty,color=
state))
```



1d)

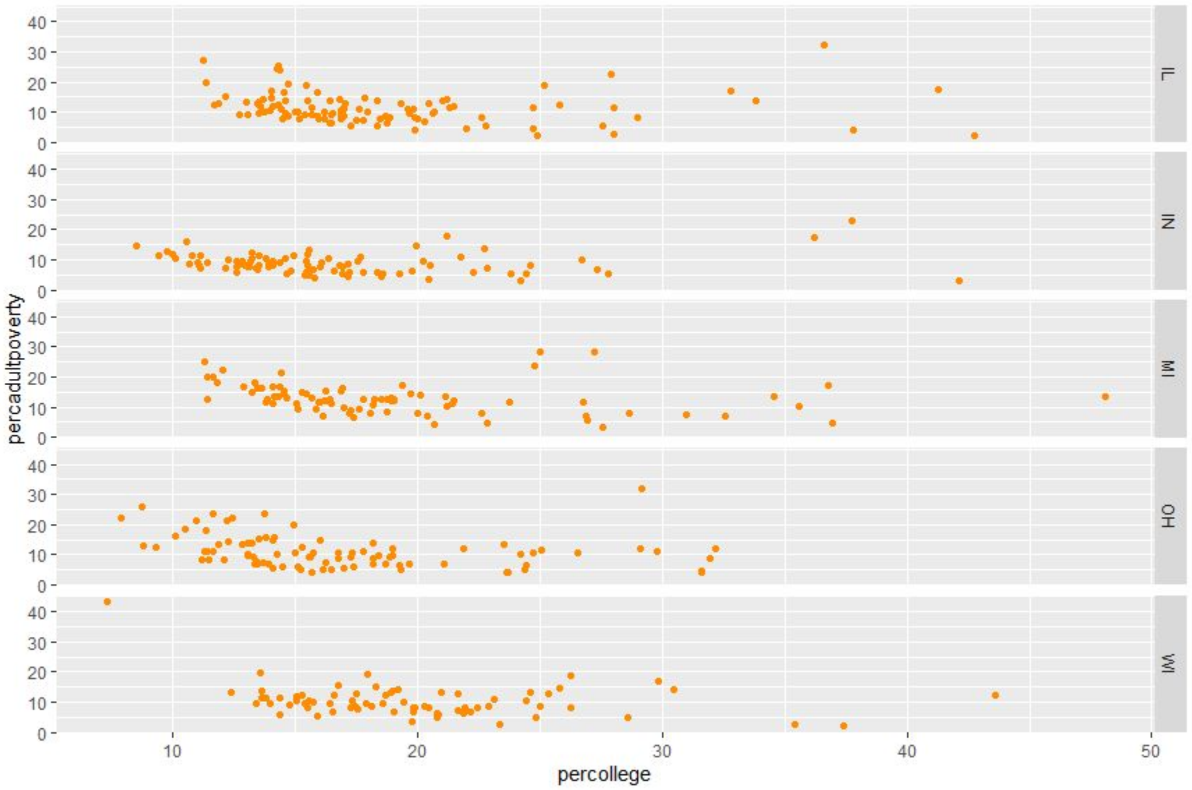
```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty,color=
percadultpoverty,shape=state))
```



1e)

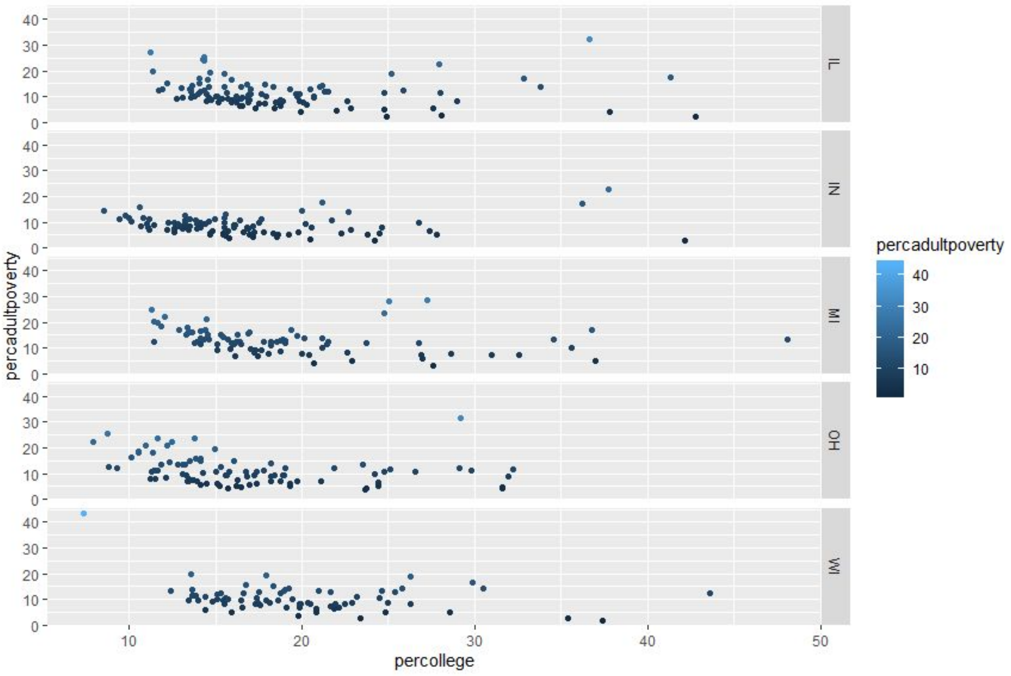
Fixed color

```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty), color= "darkorange") + facet_grid(state~.)
```



Colourised by percadultpoverty

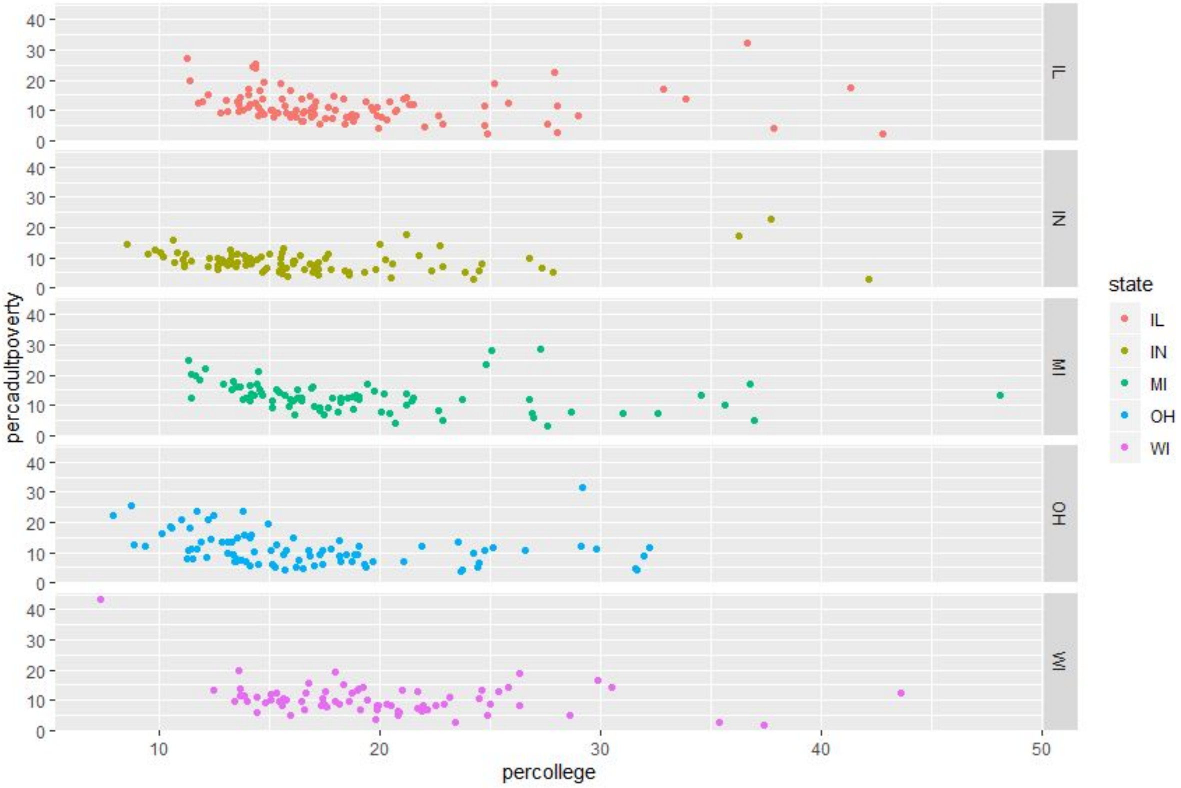
```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty, color= percadultpoverty)) + facet_grid(state~.)
```



Colourised by State

```
> ggplot() + geom_point(data = midwest, aes(x = percollege, y = percadultpoverty, color= state)) + facet_grid(state~.)
```

```
> ggplot(data = midwest, mapping= aes(x = percollege, y = percadultpoverty)) + geom_point(mapping = aes(color = state)) + facet_grid(state~.)
```



Assignment 2:

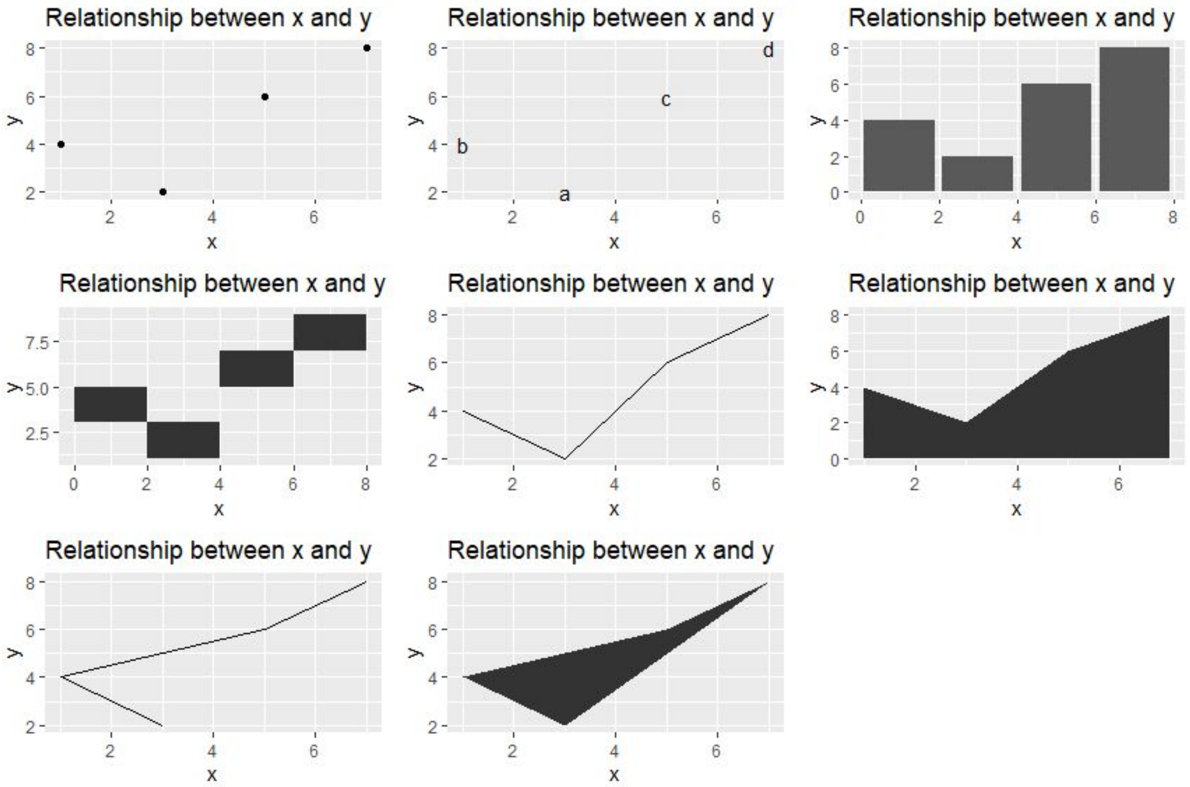
2a)

```
> my_data = data.frame(height = c(177,188,191,167), weight = c(65,75,90,60), label = c("a", "b", "c", "d"))
```

2b)

```
> library(tidyverse)
> my_first_data_frame = data.frame(x = c(3,1,5,7), y = c(2,4,6,8), label = c("a", "b", "c",
"d"))
> p1 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_point()+ggtitle("Relationship
between x and y")
> p2 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_tile()+ggtitle("Relationship
between x and y")
> p3 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_path()+ggtitle("Relationship
between x and y")
> p4 <-
ggplot(my_first_data_frame,aes(x=x,y=y,label=label))+geom_text()+ggtitle("Relationship
between x and y")
> p5 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_line()+ggtitle("Relationship
between x and y")
> p6 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_polygon()+ggtitle("Relationship
between x and y")
> p7 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_col()+ggtitle("Relationship
between x and y")
> p8 <- ggplot(my_first_data_frame,aes(x=x,y=y))+geom_area()+ggtitle("Relationship
between x and y")
> multiplot(p1,p2,p3,p4,p5,p6,p7,p8, cols=3)
```

```
> # Multiple plot function
> #
> # ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects)
> # - cols:   Number of columns in layout
> # - layout: A matrix specifying the layout. If present, 'cols' is ignored.
> #
> # If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE),
> # then plot 1 will go in the upper left, 2 will go in the upper right, and
> # 3 will go all the way across the bottom.
> #
> multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {
+    library(grid)
+
+    # Make a list from the ... arguments and plotlist
+    plots <- c(list(...), plotlist)
+
+    numPlots = length(plots)
+
+    # If layout is NULL, then use 'cols' to determine layout
+   if (is.null(layout)) {
+     # Make the panel
+     # ncol: Number of columns of plots
+     # nrow: Number of rows needed, calculated from # of cols
+     layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
+                      ncol = cols, nrow = ceiling(numPlots/cols))
+   }
+
+   if (numPlots==1) {
+     print(plots[[1]])
+
+   } else {
+     # Set up the page
+     grid.newpage()
+     pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))
+
+     # Make each plot, in the correct location
+     for (i in 1:numPlots) {
+       # Get the i,j matrix positions of the regions that contain this subplot
+       matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))
+
+       print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
+                                       layout.pos.col = matchidx$col))
+     }
+   }
+ }
```



2c)

1. geom_col: Column Chart

A column chart is used to show a comparison among different items, or it can show a comparison of items over time. We can use this format to see the revenue per landing page or customers by close date.

2. geom_line: Line Graph

A line graph reveals trends or progress over time and can be used to show many different categories of data. We use it when you chart a continuous data set.

3. geom_area: Area Chart

An area chart is basically a line chart, but the space between the x-axis and the line is filled with a color or pattern. It is useful for showing part-to-whole relations, such as showing individual sales reps' contribution to total sales for a year. It helps us analyze both overall and individual trend information.

4. geom_point: Scatter Point

The scatterplot is most useful for displaying the relationship between two continuous variables. It can be used to compare one continuous and one categorical variable, or two categorical variables

5. geom_polygon: Polygons

Polygons are very similar to paths (as drawn by geom_path()) except that the start and end points are connected and the inside is coloured by fill. The group aesthetic determines which cases are connected together into a polygon.

6. geom_Tile or heatmap

Heatmap (or heat map) is another way to visualize hierarchical clustering. It's also called a false colored image, where data values are transformed to color scale.

Heat maps allow us to simultaneously visualize clusters of samples and features. First hierarchical clustering is done of both the rows and the columns of the data matrix. The columns/rows of the data matrix are re-ordered according to the hierarchical clustering result, putting similar observations close to each other. The blocks of 'high' and 'low' values are adjacent in the data matrix. Finally, a color scheme is applied for the visualization and the data matrix is displayed. Visualizing the data matrix in this way can help to find the variables that appear to be characteristic for each sample cluster.

7. geom_text: Text

Adding text to a plot is one of the most common forms of annotation. Most plots will not benefit from adding text to every single observation on the plot, but labelling outliers and other important points is very useful.

8. geom_path: Path

Geom_path() connects the observations in the order in which they appear in the data.