Visualization Lab4 Group2

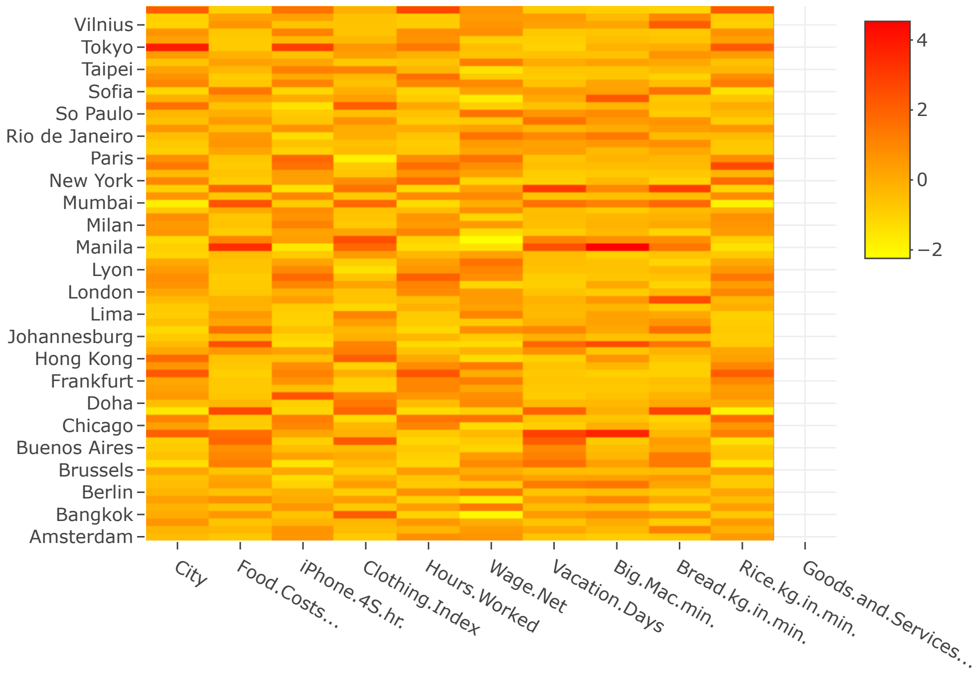
Assignment 1

Task 1

```
dataset<-read.table("prices-and-earnings.txt",sep="\t",header=TRUE,encoding = "UTF-8")
new_dataset<-dataset[,c(1,2,5,6,7,9,10,16,17,18,19)]
```

Task 2

```
a=as.matrix(new_dataset[,2:11])
a=scale(a)
p1<- plot_ly(x=colnames(new_dataset),y=new_dataset$City,z=a,
        type="heatmap",colors=colorRamp(c("yellow","red")))
p1
```

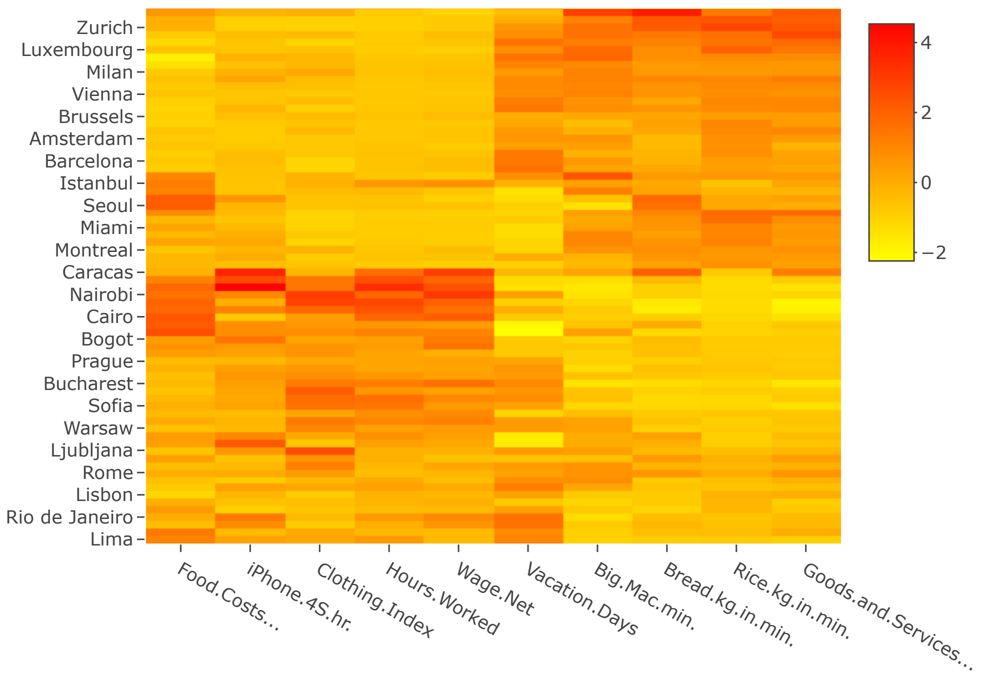


Q1: Is it possible to see clusters,outliers?

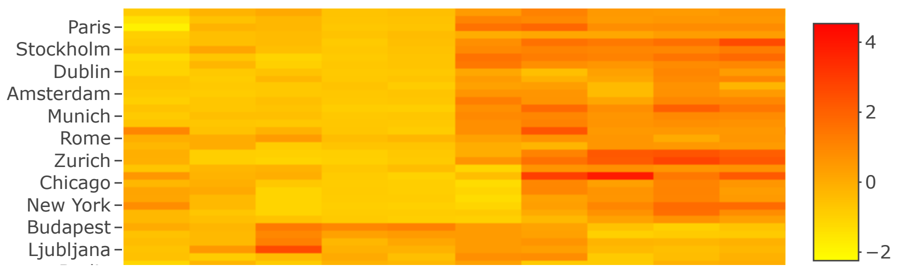
"It's difficult to find the clusters and outliers among cities due to the chaotic permutation of the dataset."

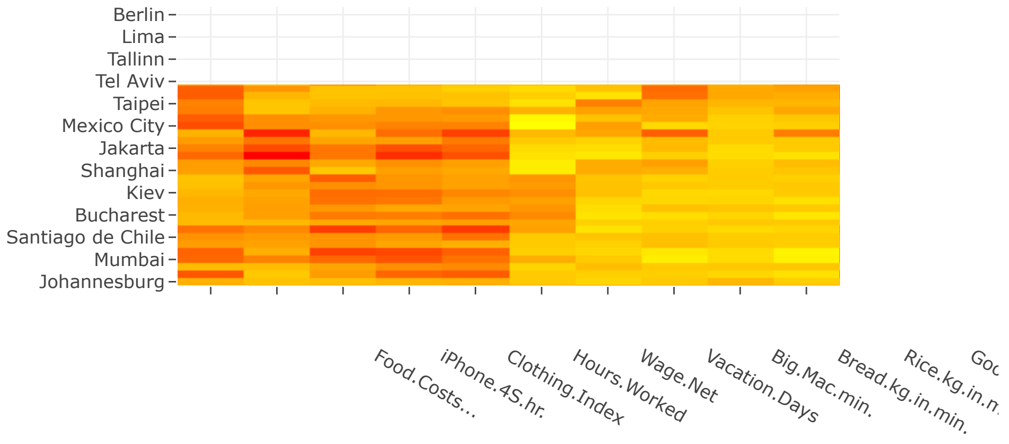
Task 3

```
#--------calculate the euclidean distance--------
euc_row_a_dist=dist(a,method="euclidean")
euc_col_a_dist=dist(t(a),method="euclidean")

euc_order_row=seriate(euc_row_a_dist,method="HC")
euc_order_col=seriate(euc_col_a_dist,method="HC")
euc_order1=get_order(euc_order_row)
euc_order2=get_order(euc_order_col)

reorder_euc_a=a[rev(euc_order1),euc_order2]
reorder_euc_names=new_dataset$City[rev(euc_order1)]

#--------calculate one minus correlation--------
cor_row_a_dist=as.dist(1-cor(t(a)))
cor_col_a_dist=as.dist(1-cor(a))

cor_order_row=seriate(cor_row_a_dist,method="HC")
cor_order_col=seriate(cor_col_a_dist,method="HC")
cor_order1=get_order(cor_order_row)
cor_order2=get_order(cor_order_col)

reorder_cor_a=a[rev(cor_order1),cor_order2]
reorder_cor_names=new_dataset$City[rev(cor_order1)]

#--------plot--------
p2<- plot_ly(x=colnames(new_dataset)[2:11],y=reorder_euc_names,z=reorder_euc_a,
             type="heatmap",colors=colorRamp(c("yellow","red")))

p3<- plot_ly(x=colnames(new_dataset)[2:11],y=reorder_cor_names,z=reorder_cor_a,
             type="heatmap",colors=colorRamp(c("yellow","red")))

p2
```



```
p3
```





Q1: which plot seems to be easier to analyse and why

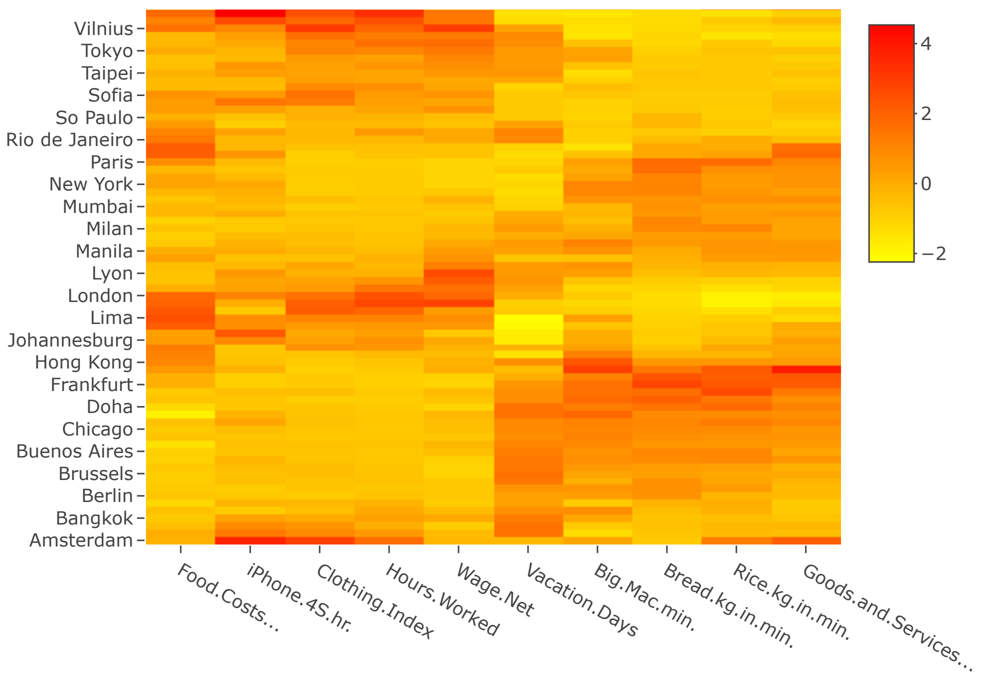
“permuted data using one minus correlation is easier to analyse,the differents clusters are divided in terms of similarity.And the outliners can be clearly identified.”

Q2: Make a detailed analysis of the plot based on Euclidian distance.

“the plot based on Euclidian distance make it easy to identify the similar clusters.the goods and services,bread.kg.in.min,etc of first 35 cities are a cluster and they are close to each other.We can also identify the outliner such as the iPhone.4S.hr. of Manila and Caracas.”

Task 4

```
#Compute a permutation that optimizes Hamiltonian Path Length ,uses Traveling Salesman Problem (TSP) as solver
tsp_order_row=seriate(euc_row_a_dist,method="TSP")
tsp_order_col=seriate(euc_col_a_dist,method="TSP")
tsp_order1=get_order(tsp_order_row)
tsp_order2=get_order(tsp_order_col)
reorder_tsp_a=a[rev(tsp_order1),tsp_order2]

#plot TSP solves
p4<- plot_ly(x=colnames(new_dataset)[2:11],y=new_dataset$City,z=reorder_tsp_a,
           type="heatmap",colors=colorRamp(c("yellow","red")))
p4
```



```
#Compare objective function values such as Hamiltonian Path length and Gradient measure achieved by row permutat
ions of TSP and HC solvers
hamiltonian_path_TSP=criterion(euc_row_a_dist,order=tsp_order_row,method="path_length")
hamiltonian_path_HC=criterion(euc_row_a_dist,order=seriate(euc_row_a_dist,method="HC"),method="path_length")
Gradient_measure_TSP=criterion(euc_row_a_dist,order=tsp_order_row,method="gradient_raw")
Gradient_measure_HC=criterion(euc_row_a_dist,order=seriate(euc_row_a_dist,method="HC"),method="gradient_raw")
result1=cbind(hamiltonian_path_TSP,hamiltonian_path_HC)
result2=cbind(Gradient_measure_TSP,Gradient_measure_HC)
result3=rbind(result1,result2)
colnames(result3)=c("TSP","HC")
rownames(result3)=c("hamiltonian_path","Gradient_measure")

result3
```

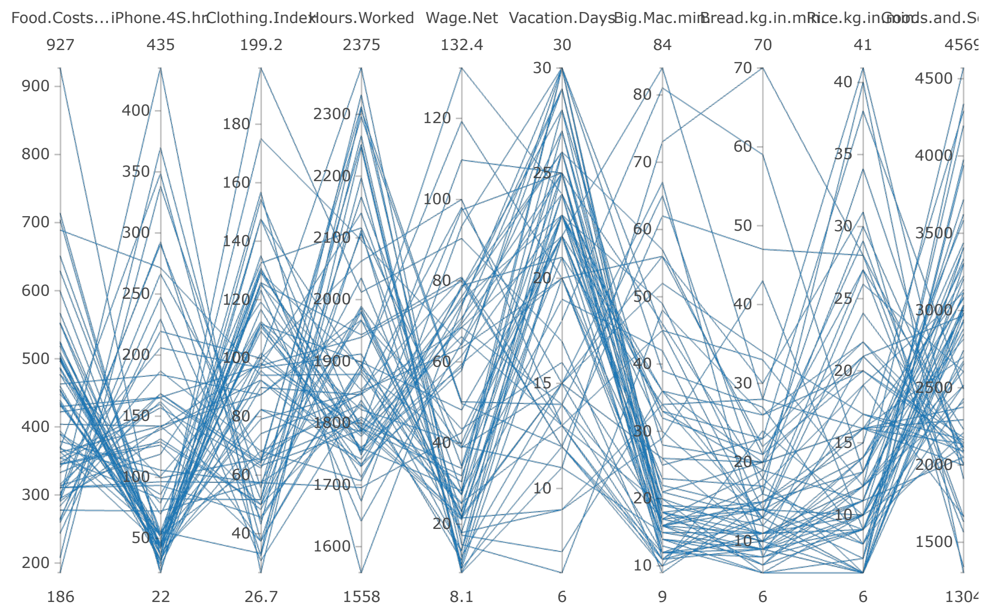
```
##                         TSP        HC
## hamiltonian_path   122.5131   138.9674
## Gradient_measure 18664.0000 30980.0000
```

Q1: Compare the heatmap given by this reordering with the heatmap produced by the HC solver in the previous step – which one seems to be better?

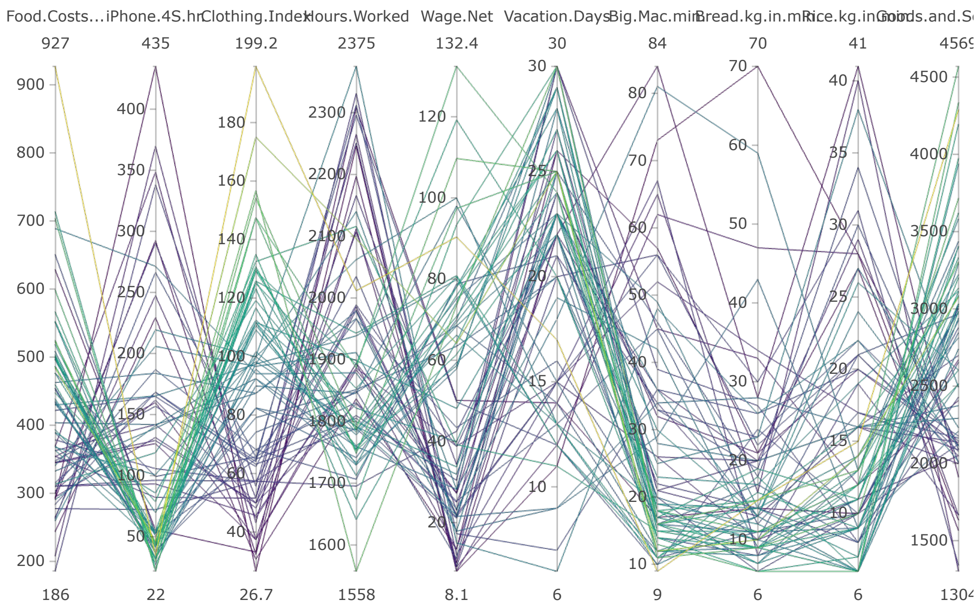
“the heatmap using TSP solves seems to be better.”

Task 5

```
p5<- dataset %>%
  plot_ly(type="parcoords",

        dimensions=list(
          list(label="Food.Costs...",values=~Food.Costs...),
          list(label="iPhone.4S.hr.",values=~iPhone.4S.hr.),
          list(label="Clothing.Index",values=~Clothing.Index),
          list(label="Hours.Worked",values=~Hours.Worked),
          list(label="Wage.Net",values=~Wage.Net),
          list(label="Vacation.Days",values=~Vacation.Days),
          list(label="Big.Mac.min.",values=~Big.Mac.min.),
          list(label="Bread.kg.in.min.",values=~Bread.kg.in.min.),
          list(label="Rice.kg.in.min.",values=~Rice.kg.in.min.),
          list(label="Goods.and.Services...",values=~Goods.and.Services...)
        ))
p5
```



```
p6<- dataset %>%
  plot_ly(type="parcoords",

          line=list(color=~Clothing.Index),
          dimensions=list(
            list(label="Food.Costs...",values=~Food.Costs...),
            list(label="iPhone.4S.hr.",values=~iPhone.4S.hr.),
            list(label="Clothing.Index",values=~Clothing.Index),
            list(label="Hours.Worked",values=~Hours.Worked),
            list(label="Wage.Net",values=~Wage.Net),
            list(label="Vacation.Days",values=~Vacation.Days),
            list(label="Big.Mac.min.",values=~Big.Mac.min.),
            list(label="Bread.kg.in.min.",values=~Bread.kg.in.min.),
            list(label="Rice.kg.in.min.",values=~Rice.kg.in.min.),
            list(label="Goods.and.Services...",values=~Goods.and.Services...)
          ))
p6
```



Q1: which variables are important to define these clusters and what values of these variables are specific to each cluster?

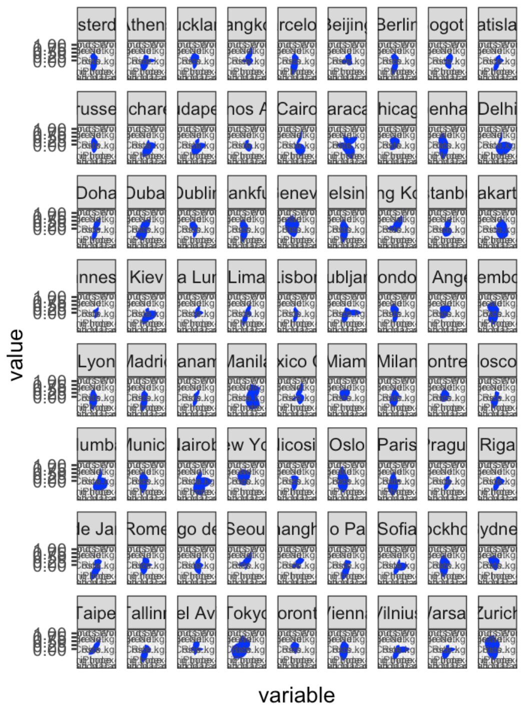
“Food.Cost,Clothing.Index and Wage.Net. In cluster A, the value of Food cost is concentrated in the range of 400-700, the value of Clothing.Index is concentrated in the range of 100-200, and the value of Wage.Net is concentrated in the range of 50-132.4 In cluster B, the value of Food cost is concentrated in the range of 186-400, the value of Clothing.Index is concentrated in the range of 26.7-100, and the value of Wage.Net is concentrated in the range of 8.1-50”

Q2: Can these clusters be interpreted? Find the most prominent outlier and interpret it.

“Cluster A represents some developed cities, so there are better labor security, higher wages and prices, but shorter working hours to buy food and electronic products. Cluster B represents some poorer, backward cities, so they have poor labor security, lower wages and prices, but longer working hours to buy food and electronics. The most prominent outlier is Tokyo(the yellow line in the figure),it is similar to cluster A,but the food cost and the clothing.Index are extremely high,meanwhile the minutes of work needed to buy a Big Mac is pretty low.Maybe Japan produces less grain and cloth,and Big Macs are very cheap there!”

Task 6

```
#Juxtaposed

#Ugly graphics
#stars(reorder_euc_a,key.loc=c(15,2), draw.segments=F, col.stars =rep("Yellow", nrow(a)))

##ggplot2
a1<-as.data.frame(reorder_euc_a)%>% mutate_all(funs(rescale))
a1$name=reorder_euc_names
a2<-a1%>%tidyr::gather(variable, value, -name, factor_key=T)%>%arrange(name)
p7<-a2 %>%
  ggplot(aes(x=variable, y=value, group=name)) +
  geom_polygon(fill="blue") +
  coord_polar() + theme_bw() + facet_wrap(~ name) +
  theme(axis.text.x = element_text(size = 5))
p7
```



Q1: Identify two smaller clusters in your data (choose yourself which ones) and the most distinct outlier.

" Cluster A:Amsterdam,Berlin,Frankfurt,Lisbon. Cluster B:Oslo,Stockholm,Helsinki Outlier:Caracas "

Task 7

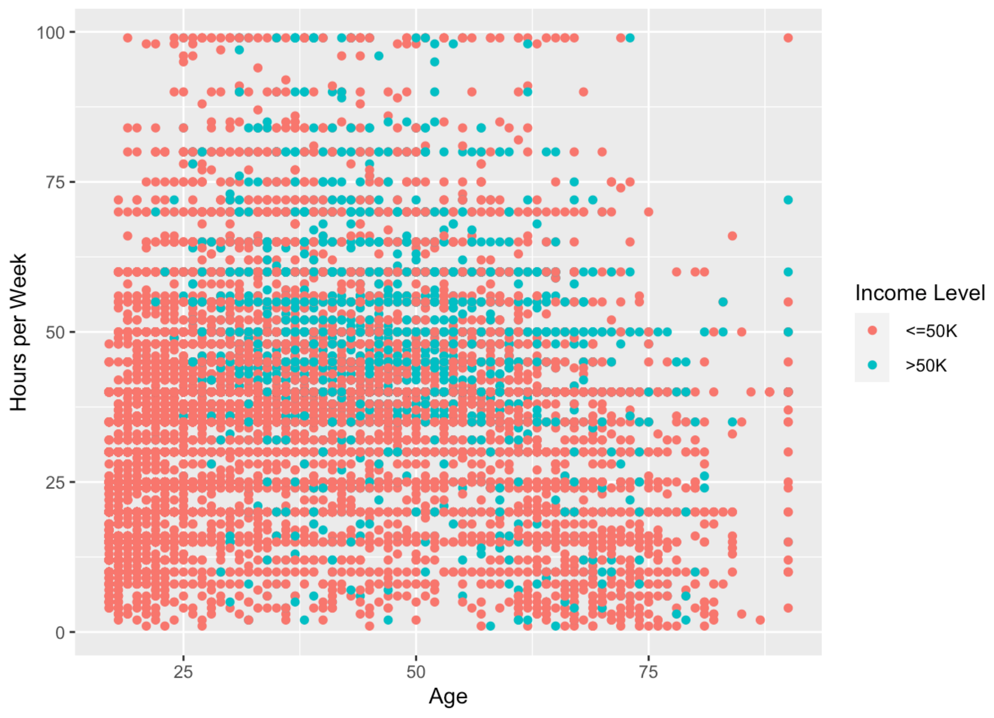
Q1: Which of the tools you have used in this assignment (heatmaps, parallel coordinates or radar charts) was best in analyzing these data? From which perspective? (e.g. efficiency, simplicity,etc.)

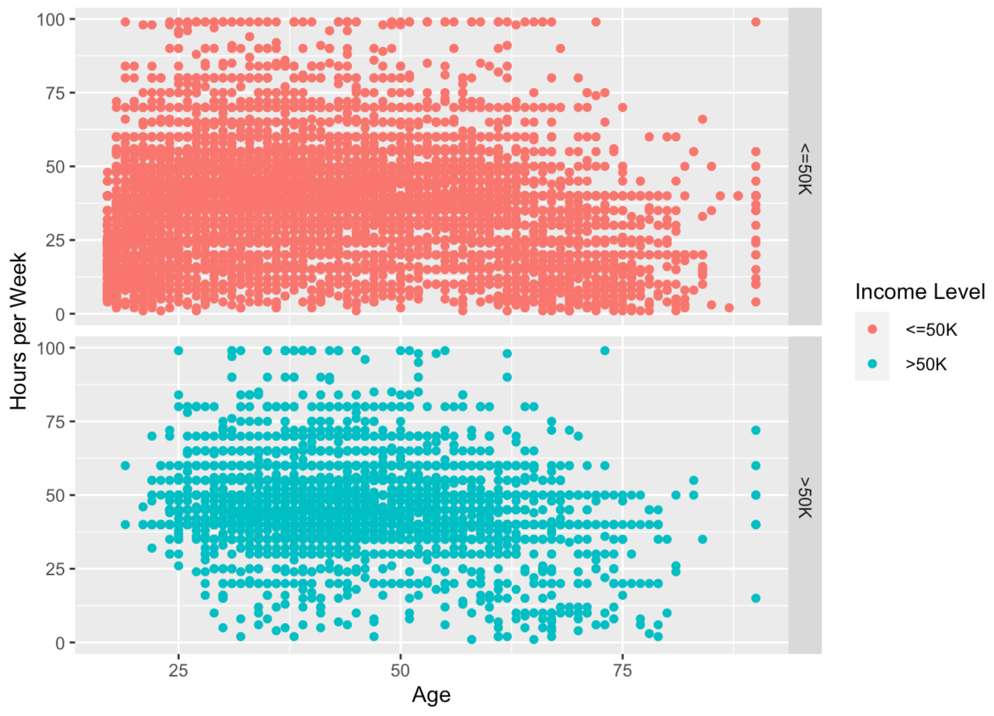
"From efficiency perspective,the radar chart is the best tool in analyse data,but from simplicity perspective,heatmap is the best tool"

Assignment 2

1. Scatter plots

```
df_adult <- read.csv("adult.csv", header = FALSE)

p2_1_1 <- ggplot(df_adult, aes(x = age, y = `hours-per-week`, color = `income-level`)) +
  geom_point() +
  xlab("Age") +
  ylab("Hours per Week") +
  labs(color='Income Level')

p2_1_2 <- p2_1_1 + facet_grid(`income-level`~.)
```





Q1: Why it is problematic to analyze this plot?

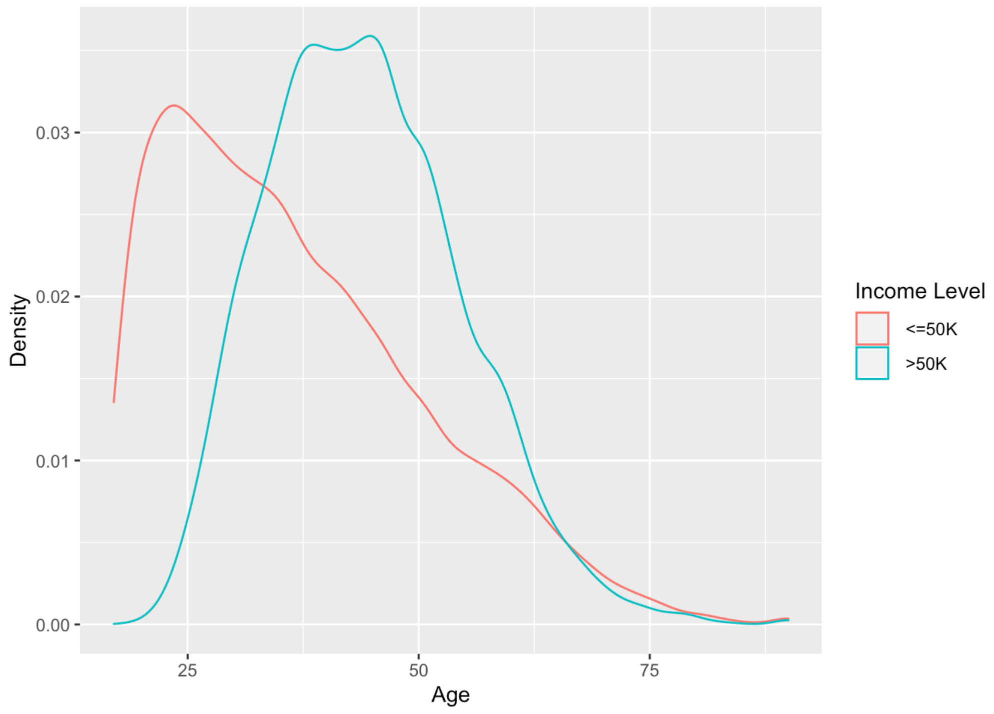
In the first plot, too many points overlap each other. Different colors also cause visual confusion. It's hard to find patterns between groups or inside one group.

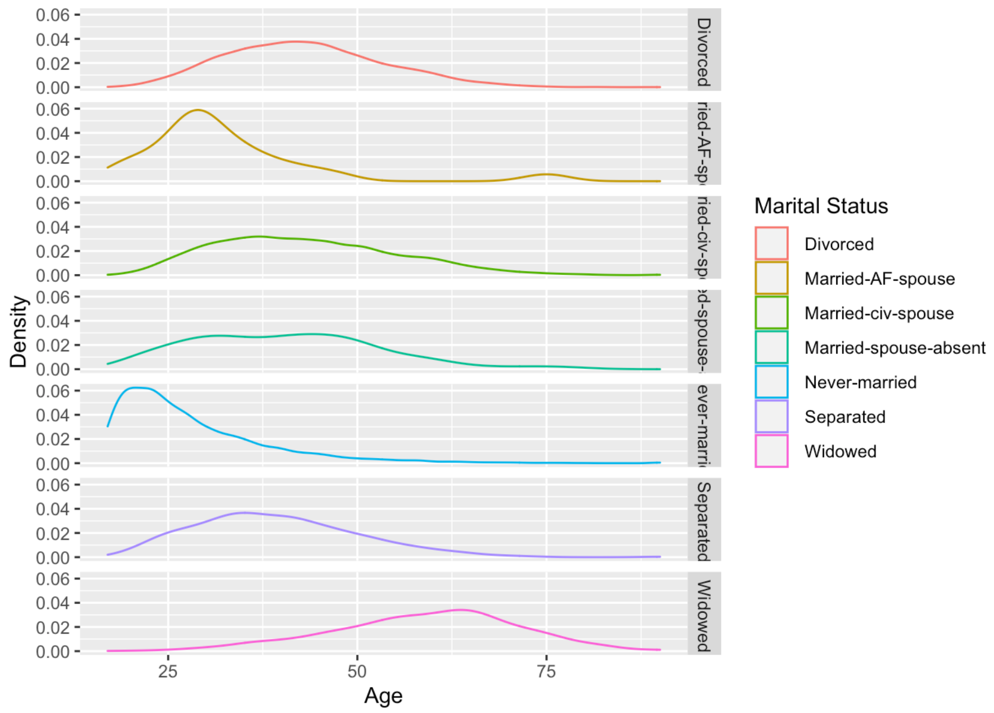
Q2: What new conclusions can you make here?

For the two income levels, they both have highest density in the approximate range of 30 to 60 on the Y-axis. This may indicate that there is no particularly strong correlation between hours per week and income level.

2.

```
p2_2_1 <- ggplot(df_adult, aes(x = age, color = `income-level`)) +
  geom_density() +
  xlab("Age") +
  ylab("Density") +
  labs(color='Income Level')

p2_2_2 <- ggplot(df_adult, aes(x = age, color = `marital-status`)) +
  geom_density() +
  facet_grid(`marital-status`~.) +
  xlab("Age") +
  ylab("Density") +
  labs(color='Marital Status')
```





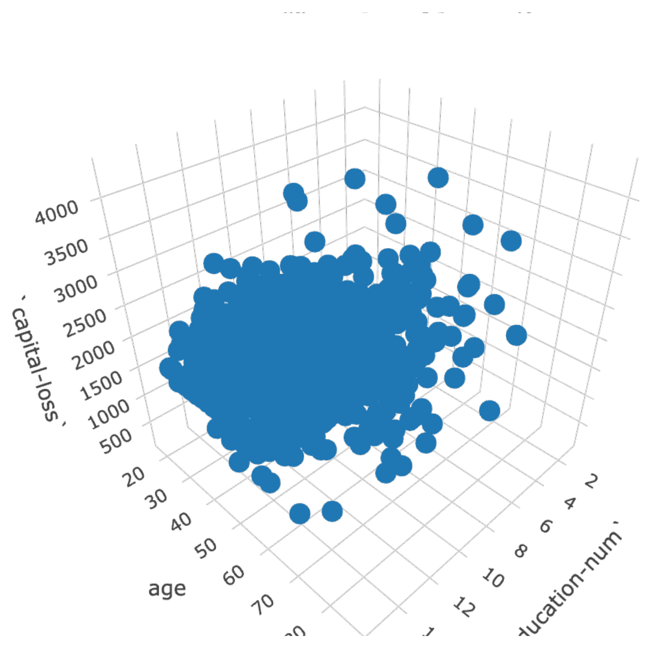
Q1: Analyze these two plots and make conclusions.

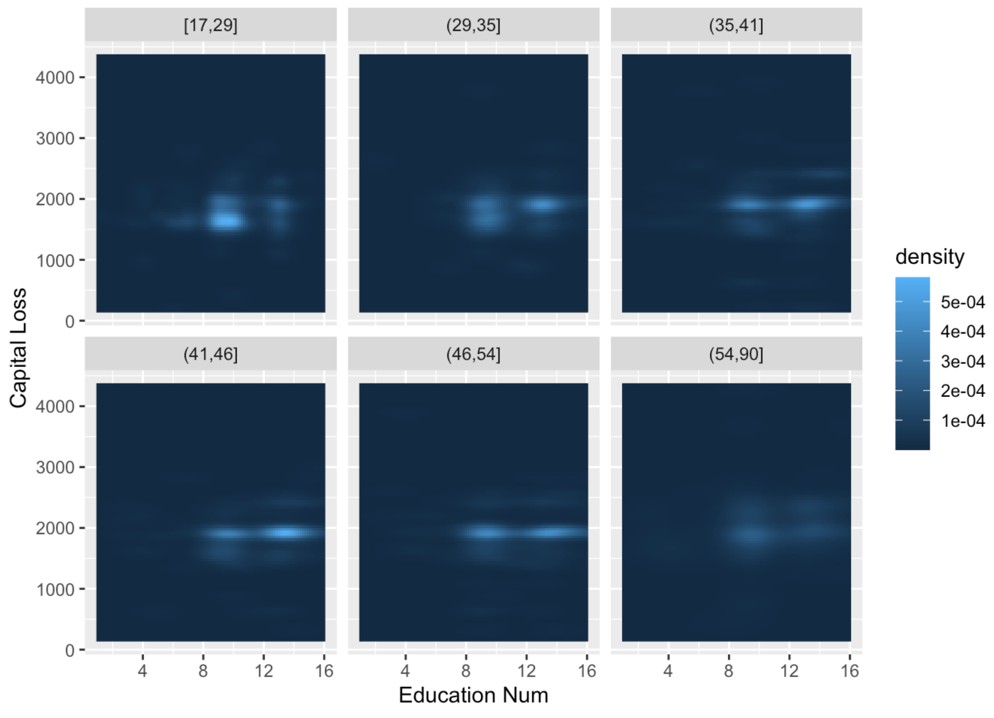
For the first plot. we can see that the mode age is higher in the high-income group. It implies that older people have more money than younger ones. We also can see that the age of low-income group concentrate around 25 and the high-income group concentrate from 35 to 45.

The second plot shows that most never-married people's age are under 25. The people from 30 to 50 years old have similar distribution in various marital status. Most widowed people are around 60 years old. The peak of the married-AF-spouse's curve comes around 30 years old. Such a distribution is in line with our common sense about the marital status for people. And combined with the first plot, income may has some impacts on marital status.

3. Trellis plot

```
df_filted <- df_adult %>%
  filter(.data[["capital-loss"]] != 0)

p2_3_1 <- df_filted %>%
  plot_ly(x = ~`education-num`, y = ~age, z = ~`capital-loss`) %>%
  add_trace(type = "scatter3d", mode = "markers")

group <- cut_number(x = df_filted$age, n = 6)
p2_3_2 <- ggplot(df_filted, aes(x = `education-num`, y = `capital-loss`)) +
  geom_density_2d() +
  stat_density_2d(geom = "raster", mapping = aes(fill=..density..), contour = FALSE) +
  facet_grid(group) +
  facet_wrap(group, ncol = 3) +
  labs(x = "Education Num", y = "Capital Loss")
```





Q1: Why is it difficult to analyze the 3D-Scatter plot?

At the default zoom level, many points cover each other, and after zooming in, it goes to the detail part and we can't find the pattern in global. The transformation from 3D to 2D in our visual system is also unreliable, and may produce misunderstanding.

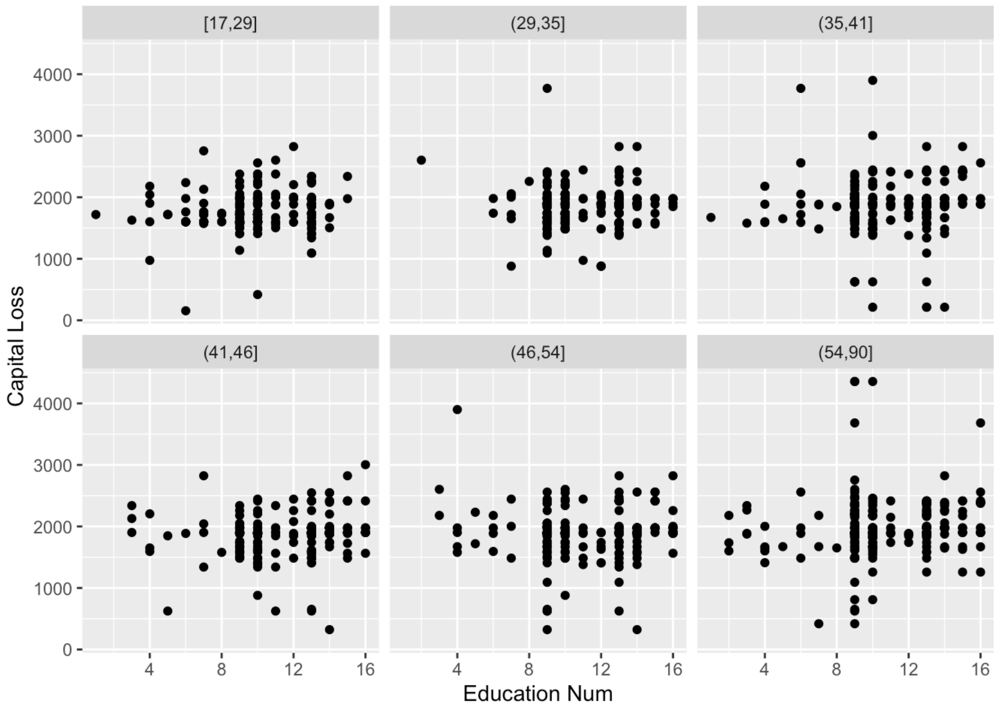
Q2: Analyze the trellis plot

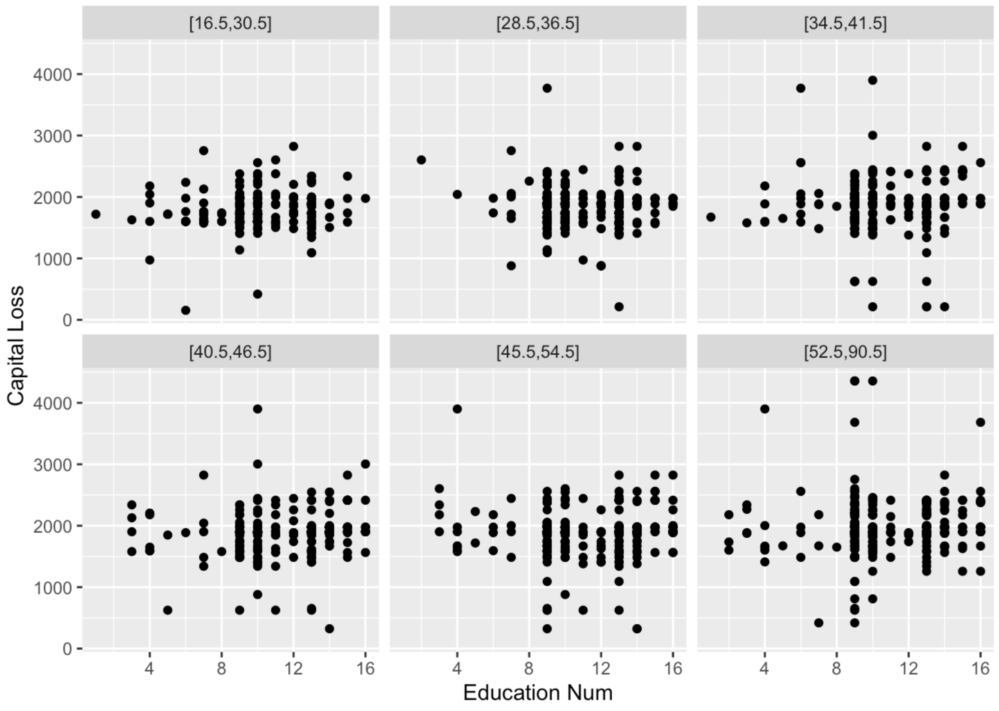
Among the groups, we can see that the youngest group has the most loss density, and the oldest group has the least. Other three groups are in the middle with similar density. It may because that young people lack experience in managing capital. And most capital loss happened in range 8 ~ 16 on x-axis which represent education number. It may because that the people before 12th education (education number is 8) did not know about capital management or did not have enough capital to manage.

Inside the youngest group, the education between 8 and 12 has the most loss density, education number of 12~16 has the second high density. But with the increasing of age, 12~16 caught up, became the most one, except the oldest group.

4. Shingles

```r
p2_4_1 <- df_filted %>%
  ggplot(aes(x = `education-num`, y = `capital-loss`)) +
  geom_point() +
  facet_wrap(group, ncol = 3) +
  labs(x = "Education Num", y = "Capital Loss")

# shingles

agerange <- lattice::equal.count(df_filted$age, number = 6, overlap = 0.1)
levels_array <- levels(agerange)

L <- matrix(unlist(levels(agerange)), ncol = 2, byrow = T)
L1 <- data.frame(Lower = L[,1],Upper = L[,2], Interval = factor(1:nrow(L)))

index <- c()
Class <- c()
for(i in 1:nrow(L)){
  Cl <- paste("[", L1$Lower[i], ",", L1$Upper[i], "]", sep="")
  ind <- which(df_filted$age >= L1$Lower[i] & df_filted$age <= L1$Upper[i])
  index <- c(index,ind)
  Class <- c(Class, rep(Cl, length(ind)))
}

df_shingles <- df_filted[index,]
df_shingles$Class<-as.factor(Class)

p2_4_2 <- df_shingles %>%
  ggplot(aes(x = `education-num`, y = `capital-loss`)) +
  geom_point() +
  facet_wrap(~Class, ncol = 3) +
  labs(x = "Education Num", y = "Capital Loss")
```





Q1: Which advantages and disadvantages you see in using Shingles?

Although using shingles can avoid boundary effects, but we did not find any advantages or disadvantages after using it in this case.

Appendix

Codes For Assignment 1

```r
library(ggplot2)
library(plotly)
library(seriation)

#1
dataset<-read.table("C:/Users/Tsunami Liu/Documents/prices-and-earnings.txt",sep="\t",header=TRUE,encoding = "UT
F-8")
new_dataset<-dataset[,c(1,2,5,6,7,9,10,16,17,18,19)]

#2
a=as.matrix(new_dataset[,2:11])
a=scale(a)
p1<- plot_ly(x=colnames(new_dataset),y=new_dataset$City,z=a,
        type="heatmap",colors=colorRamp(c("yellow","red")))
p1


#3

#--------calculate the euclidean distance--------
euc_row_a_dist=dist(a,method="euclidean")
euc_col_a_dist=dist(t(a),method="euclidean")

euc_order_row=seriate(euc_row_a_dist,method="HC")
euc_order_col=seriate(euc_col_a_dist,method="HC")
euc_order1=get_order(euc_order_row)
euc_order2=get_order(euc_order_col)

reorder_euc_a=a[rev(euc_order1),euc_order2]
reorder_euc_names=new_dataset$City[rev(euc_order1)]

#--------calculate one minus correlation--------
cor_row_a_dist=as.dist(1-cor(t(a)))
cor_col_a_dist=as.dist(1-cor(a))

cor_order_row=seriate(cor_row_a_dist,method="HC")
cor_order_col=seriate(cor_col_a_dist,method="HC")
cor_order1=get_order(cor_order_row)
cor_order2=get_order(cor_order_col)

reorder_cor_a=a[rev(cor_order1),cor_order2]
reorder_cor_names=new_dataset$City[rev(cor_order1)]

#--------plot--------
p2<- plot_ly(x=colnames(new_dataset)[2:11],y=reorder_euc_names,z=reorder_euc_a,
             type="heatmap",colors=colorRamp(c("yellow","red")))

p3<- plot_ly(x=colnames(new_dataset)[2:11],y=reorder_cor_names,z=reorder_cor_a,
             type="heatmap",colors=colorRamp(c("yellow","red")))

p2
p3


#4
#Compute a permutation that optimizes Hamiltonian Path Length ,uses Traveling Salesman Problem (TSP) as solver
tsp_order_row=seriate(euc_row_a_dist,method="TSP")
tsp_order_col=seriate(euc_col_a_dist,method="TSP")
tsp_order1=get_order(tsp_order_row)
tsp_order2=get_order(tsp_order_col)
reorder_tsp_a=a[rev(tsp_order1),tsp_order2]

#plot TSP solves
p4<- plot_ly(x=colnames(new_dataset)[2:11],y=new_dataset$City,z=reorder_tsp_a,
             type="heatmap",colors=colorRamp(c("yellow","red")))
p4


#Compare objective function values such as Hamiltonian Path length and Gradient measure achieved by row permutat
ions of TSP and HC solvers
hamiltonian_path_TSP=criterion(euc_row_a_dist,order=tsp_order_row,method="path_length")
hamiltonian_path_HC=criterion(euc_row_a_dist,order=seriate(euc_row_a_dist,method="HC"),method="path_length")
Gradient_measure_TSP=criterion(euc_row_a_dist,order=tsp_order_row,method="gradient_raw")
```

```r
Gradient_measure_HC=criterion(euc_row_a_dist,order=seriate(euc_row_a_dist,method="HC"),method="gradient_raw")
result1=cbind(hamiltonian_path_TSP,hamiltonian_path_HC)
result2=cbind(Gradient_measure_TSP,Gradient_measure_HC)
result3=rbind(result1,result2)
colnames(result3)=c("TSP","HC")
rownames(result3)=c("hamiltonian_path","Gradient_measure")

result3

#5
library(dplyr)

p5<- dataset %>%
  plot_ly(type="parcoords",

        dimensions=list(
          list(label="Food.Costs...",values=~Food.Costs...),
          list(label="iPhone.4S.hr.",values=~iPhone.4S.hr.),
          list(label="Clothing.Index",values=~Clothing.Index),
          list(label="Hours.Worked",values=~Hours.Worked),
          list(label="Wage.Net",values=~Wage.Net),
          list(label="Vacation.Days",values=~Vacation.Days),
          list(label="Big.Mac.min.",values=~Big.Mac.min.),
          list(label="Bread.kg.in.min.",values=~Bread.kg.in.min.),
          list(label="Rice.kg.in.min.",values=~Rice.kg.in.min.),
          list(label="Goods.and.Services...",values=~Goods.and.Services...)
        ))
p5


p6<- dataset %>%
  plot_ly(type="parcoords",

          line=list(color=~Clothing.Index),
          dimensions=list(
            list(label="Food.Costs...",values=~Food.Costs...),
            list(label="iPhone.4S.hr.",values=~iPhone.4S.hr.),
            list(label="Clothing.Index",values=~Clothing.Index),
            list(label="Hours.Worked",values=~Hours.Worked),
            list(label="Wage.Net",values=~Wage.Net),
            list(label="Vacation.Days",values=~Vacation.Days),
            list(label="Big.Mac.min.",values=~Big.Mac.min.),
            list(label="Bread.kg.in.min.",values=~Bread.kg.in.min.),
            list(label="Rice.kg.in.min.",values=~Rice.kg.in.min.),
            list(label="Goods.and.Services...",values=~Goods.and.Services...)
          ))
p6

#6
#Juxtaposed

#Ugly graphics
#stars(reorder_euc_a,key.loc=c(15,2), draw.segments=F, col.stars =rep("Yellow", nrow(a)))

##ggplot2
library(scales)
a1<-as.data.frame(reorder_euc_a)%>% mutate_all(funs(rescale))
a1$name=reorder_euc_names
a2<-a1%>%tidyr::gather(variable, value, -name, factor_key=T)%>%arrange(name)
p<-a2 %>%
  ggplot(aes(x=variable, y=value, group=name)) +
  geom_polygon(fill="blue") +
  coord_polar() + theme_bw() + facet_wrap(~ name) +
  theme(axis.text.x = element_text(size = 5))
p
```

Codes For Assignment 2

```r
library(ggplot2)
library(plotly)
library(lattice)

# Task 1
df_adult <- read.csv("adult.csv", header = FALSE)
colnames(df_adult) <- c(
  "age",
  "workclass",
  "fnlwgt",
  "education",
  "education-num",
  "marital-status",
  "occupation",
  "relationship",
  "race",
  "sex",
  "capital-gain",
  "capital-loss",
  "hours-per-week",
  "native-country",
  "income-level"
)

p2_1_1 <- ggplot(df_adult, aes(x = age, y = `hours-per-week`, color = `income-level`)) +
  geom_point() +
  xlab("Age") +
  ylab("Hours per Week") +
  labs(color='Income Level')

p2_1_2 <- p2_1_1 + facet_grid(`income-level`~.)

p2_1_1
p2_1_2

# Task 2
p2_2_1 <- ggplot(df_adult, aes(x = age, color = `income-level`)) +
  geom_density() +
  xlab("Age") +
  ylab("Density") +
  labs(color='Income Level')

p2_2_2 <- ggplot(df_adult, aes(x = age, color = `marital-status`)) +
  geom_density() +
  facet_grid(`marital-status`~.) +
  xlab("Age") +
  ylab("Density") +
  labs(color='Marital Status')

p2_2_1
p2_2_2

# Task 3
df_filted <- df_adult %>%
  filter(.data[["capital-loss"]] != 0)

p2_3_1 <- df_filted %>%
  plot_ly(x = ~`education-num`, y = ~age, z = ~`capital-loss`) %>%
  add_trace(type = "scatter3d", mode = "markers")

group <- cut_number(x = df_filted$age, n = 6)
p2_3_2 <- ggplot(df_filted, aes(x = `education-num`, y = `capital-loss`)) +
  geom_density_2d() +
  stat_density_2d(geom = "raster", mapping = aes(fill=..density..), contour = FALSE) +
  facet_grid(group) +
  facet_wrap(group, ncol = 3) +
  labs(x = "Education Num", y = "Capital Loss")

p2_3_1
p2_3_2

# Task 4
p2_4_1 <- df_filted %>%
  ggplot(aes(x = `education-num`, y = `capital-loss`)) +
  geom_point() +
  facet_wrap(group, ncol = 3) +
  labs(x = "Education Num", y = "Capital Loss")

# shingles

agerange <- lattice::equal.count(df_filted$age, number = 6, overlap = 0.1)
levels_array <- levels(agerange)

L <- matrix(unlist(levels(agerange)), ncol = 2, byrow = T)
L1 <- data.frame(Lower = L[,1],Upper = L[,2], Interval = factor(1:nrow(L)))

index <- c()
Class <- c()
for(i in 1:nrow(L)){
  Cl <- paste("[", L1$Lower[i], ",", L1$Upper[i], "]", sep="")
  ind <- which(df_filted$age >= L1$Lower[i] & df_filted$age <= L1$Upper[i])
  index <- c(index,ind)
  Class <- c(Class, rep(Cl, length(ind)))
}

df_shingles <- df_filted[index,]
df_shingles$Class<-as.factor(Class)

p2_4_2 <- df_shingles %>%
  ggplot(aes(x = `education-num`, y = `capital-loss`)) +
  geom_point() +
  facet_wrap(~Class, ncol = 3) +
  labs(x = "Education Num", y = "Capital Loss")

p2_4_1
p2_4_2
```