

challenge-3

Naiyu Jiang

Question 1: Create a dichotomous feature for each identity question.

```
# Load libraries
library(tidyverse)
library(amerika)
library(tictoc)
library(kohonen)
set.seed(1234)

# read in the 2016 ANES data
anes <- read_csv("anes_2016.csv")
# select the fourteen survey questions on salient social issues
anes_short <- anes %>%
  select(vaccine, autism, birthright_b, forceblack, forcewhite,
         stopblack, stopwhite, freetrade, aa3,
         warndo, finwell, childcare, healthspend,
         minwage, amer_ident, race_ident) %>%
  mutate(strong_amer_ident = case_when(amer_ident == 1 ~ 1,
                                       amer_ident == 2 ~ 1, TRUE ~ 0),
         strong_race_ident = case_when(race_ident == 1 ~ 1,
                                       race_ident == 2 ~ 1, TRUE ~ 0))

# take a look at the dataset
skimr::skim(anes_short)
```

Table 1: Data summary

Name	anes_short
Number of rows	1200
Number of columns	18
Column type frequency:	
numeric	18
Group variables	None

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
vaccine	0	1	2.30	1.74	1	1	1	3	7	
autism	0	1	4.48	1.56	1	3	5	6	6	
birthright_b	0	1	6.58	2.83	1	4	7	9	9	
forceblack	0	1	3.59	1.05	1	3	3	5	5	
forcewhite	0	1	2.68	0.86	1	2	3	3	5	

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
stopblack	0	1	3.66	1.03	1	3	4	5	5	
stopwhite	0	1	2.69	0.94	1	2	3	3	8	
freetrade	0	1	3.90	1.62	1	3	4	5	7	
aa3	0	1	6.71	2.67	1	4	7	9	9	
warmdo	0	1	3.20	2.05	1	1	3	4	8	
finwell	0	1	4.59	1.73	1	3	5	6	8	
childcare	0	1	3.38	1.79	1	2	3	4	7	
healthspend	0	1	3.26	1.92	1	2	3	4	8	
minwage	0	1	1.60	0.91	1	1	1	2	8	
amer_ident	0	1	2.14	1.25	1	1	2	3	5	
race_ident	0	1	2.80	1.47	1	1	3	4	5	
strong_amer_ident	0	1	0.66	0.47	0	0	1	1	1	
strong_race_ident	0	1	0.46	0.50	0	0	0	1	1	

According to the descriptive statistics given by the summary table above, we can read that there are responses with overly large scales than expected embedded in `birthright_b`, `stopwhite`, `aa3`, `warmdo`, `finwell`, `healthspend`, `minwage`. There are two options for these incorrect values, which are either removing these anomalies, or imputing them with appropriate values. If dropping all these responses with abnormally large scales, the size of the dataset will shrink drastically from 1,200 to 314. This is not considered a good practice as too much information loss is incurred. Therefore, I opt for the alternative method, imputation.

I assume that the inputs from the participants are trustworthy and responses with overly large scales are just careless errors made by respondents who wanted to tick the largest value. Therefore, I will impute those wrong values with the upper bound of their respective scale.

```
# preprocess and clean the 14 variables
anes_short <- anes_short %>%
  mutate(birthright_b = replace(birthright_b, birthright_b > 7, 7),
         stopwhite = replace(stopwhite, stopwhite > 5, 5),
         aa3 = replace(aa3, aa3 > 7, 7),
         warmdo = replace(warmdo, warmdo > 7, 7),
         finwell = replace(finwell, finwell > 7, 7),
         healthspend = replace(healthspend, healthspend > 7, 7),
         minwage = replace(minwage, minwage > 4, 4)) %>%
  as.data.frame()
```

Question 2: Build a self-organizing map based on only the above-listed social questions.

We need to first scale the 14 variables for further analysis and build a search grid. As for the hyperparameter tuning for the self-organizing map, I select the tuning parameters as `alpha` (learning rate), `radius` (neighborhood size), and `rlen` (#iterations), and set the internal validation criterion to be the mean distance to closest nodes to capture the tightness and stability of the clustering around the neurons. By iteratively employing the grid-search logic, I manage to find the best combination of hyperparameters at the minimal mean distance to the closest nodes.

```
# scale variables
anes_short$strong_amer_ident <- as.factor(anes_short$strong_amer_ident)
anes_short$strong_race_ident <- as.factor(anes_short$strong_race_ident)
anes_short[1:14] <- lapply(anes_short[1:14], as.numeric)
anes_scaled <- anes_short %>%
  dplyr::select(vaccine:minwage) %>%
  scale()
```

```

# create the structure of the output layer;
# specify the dimensions of the grid to search
search_grid <- somgrid(xdim = 10,
                      ydim = 10,
                      topo = "rectangular",
                      neighbourhood.fct = "gaussian")

# initialize the gridsearch table to store all combinations of the hyperparameters.
gridsearch_som <- tibble()
# Here, we use the for loop because we want to select the best hyperparameters
{
  tic()
  for (alpha in c(0.1,0.3,0.5)){
    for (radius in c(0.5,1,1.5)){
      for (rlen in c(300,500,700)){
        som_fit <- som(anes_scaled,
                      grid = search_grid,
                      # we set the lower bound of the learning rate to be
                      # 1/100 of the original one
                      alpha = c(alpha, alpha/100),
                      radius = radius, # neighborhood size
                      rlen = rlen, # number of epochs (# sees the data)
                      dist.fcts = "euclidean",
                      mode = "batch")
        gridsearch_som <- rbind(gridsearch_som,
                                c(alpha, radius, rlen, min(som_fit$changes)))
      }
    }
  }
  toc()
}## ~3mins

## 59.021 sec elapsed
names(gridsearch_som) <- c("initial_alpha", "initial_radius", "iterations",
                          "mean_distance_to_closest_nodes")
# Let's take a look at the best combination and
# the corresponding mean distance to closest nodes
# initial alpha = 0.3, initial radius = 1, number of iterations = 500,
# and the smallest mean distance is around 0.0161
gridsearch_som[which.min(gridsearch_som$mean_distance_to_closest_nodes),]

##      initial_alpha initial_radius iterations mean_distance_to_closest_nodes
## 23              0.5              1         500              0.01606905

# refit the SOM using the best combination of hyperparameters
{
  tic()

  best_alpha <- gridsearch_som[which.min(gridsearch_som$mean_distance_to_closest_nodes),
                                "initial_alpha"]
  best_radius <- gridsearch_som[which.min(gridsearch_som$mean_distance_to_closest_nodes),
                                "initial_radius"]
  best_rlen <- gridsearch_som[which.min(gridsearch_som$mean_distance_to_closest_nodes),
                              "iterations"]
}

```

```
som_fit <- som(anes_scaled,
  grid = search_grid,
  alpha = c(best_alpha, best_alpha/100),
  radius = best_radius,
  rlen = best_rlen,
  dist.fcts = "euclidean",
  mode = "batch")

toc()
}
```

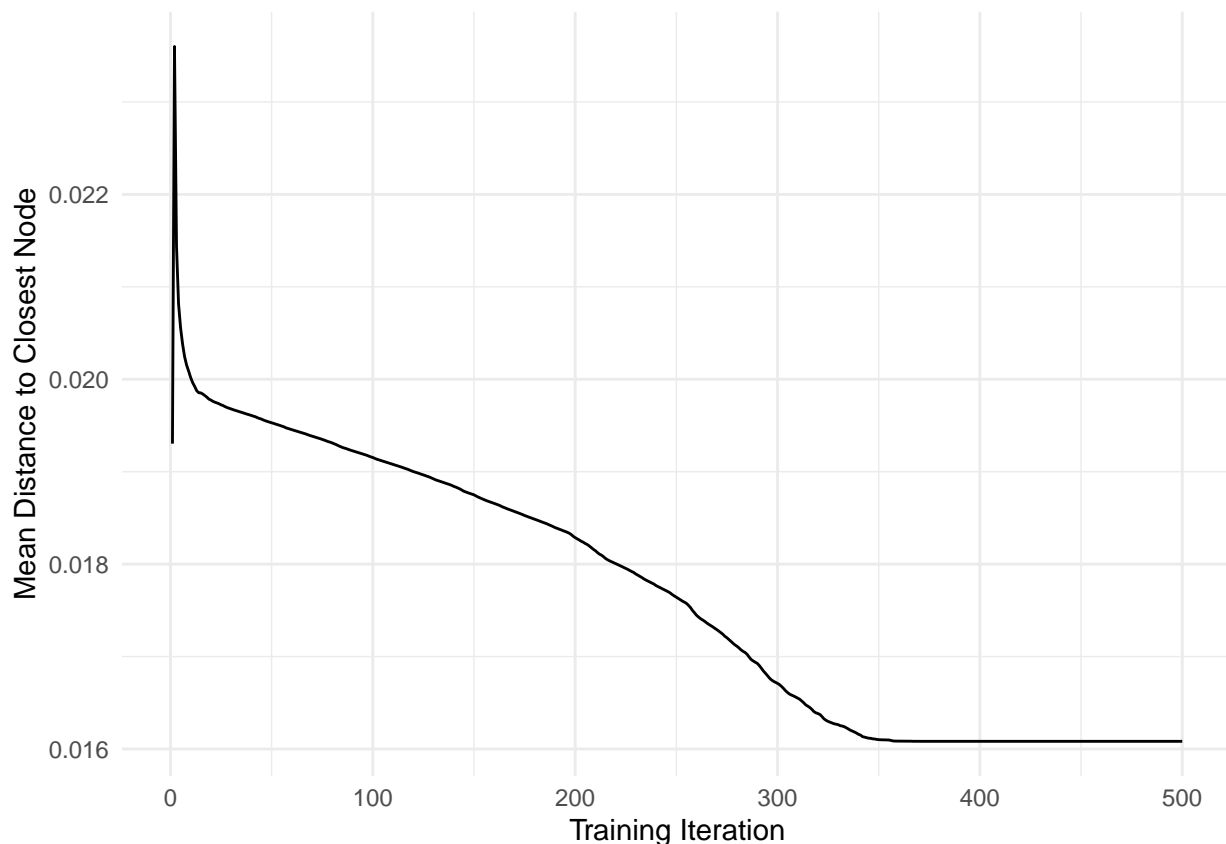
2.185 sec elapsed

Question 3: Diagnose the output of your self-organizing map *visually*.

(1) Plot the training process.

The graph explains how the mean distance to the closest unit changes with increase in the iteration level.

```
# plot the training process
som_fit$changes %>%
  as_tibble() %>%
  mutate(changes = V1,
    iteration = seq(1:length(changes))) %>%
  ggplot(aes(iteration, changes)) +
  geom_line() +
  labs(x = "Training Iteration", y = "Mean Distance to Closest Node") +
  theme_minimal()
```

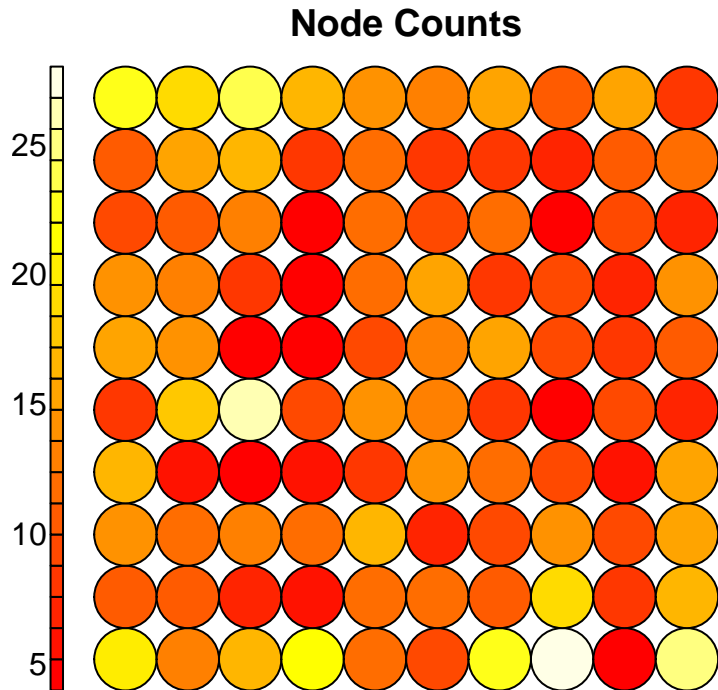


We can observe that the learning rate decreases with each iteration.

(2) Node counts.

The Kohonen package allows us to visualize the count of how many samples are mapped to each node on the map. This metric can be used as a measure of map quality – ideally, the sample distribution is relatively uniform. Large values in some map areas suggest that a larger map would be beneficial.

```
# plot node counts
plot(som_fit, type="count", main="Node Counts")
```

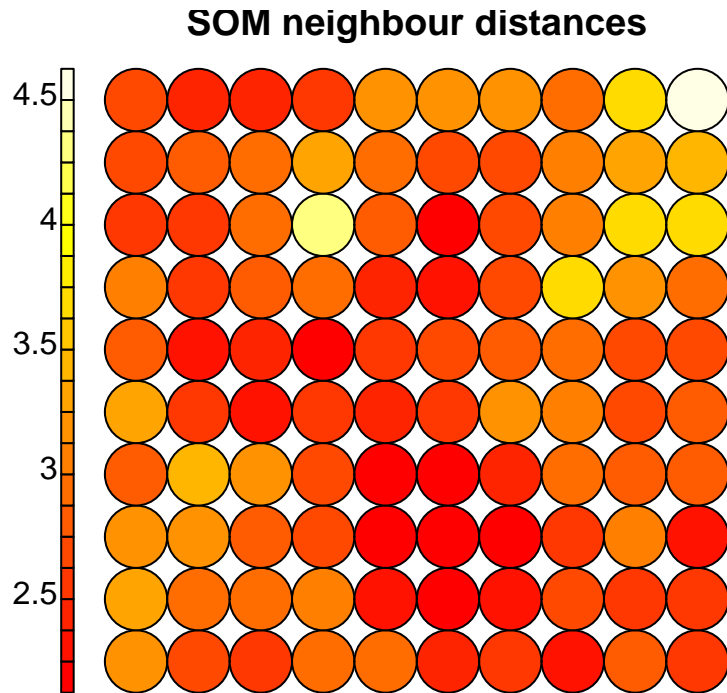


Nodes that colored by red mean nodes that have the least number of input values. If the color nodes are brighter indicating the nodes have a lot of input values. From this model, we can see that a lot of nodes have small number of input values.

(3) Neighbor distance.

Often referred to as the “U-Matrix”, this visualization is of the distance between each node and its neighbors. Typically viewed with a grayscale palette, areas of low neighbor distance indicate groups of nodes that are similar. Areas with large distances indicate the nodes are much more dissimilar – and indicate natural boundaries between node clusters. The U-Matrix can be used to identify clusters within the SOM map.

```
# plot neighbor distance
plot(som_fit, type="dist.neighbours", main = "SOM neighbour distances")
```



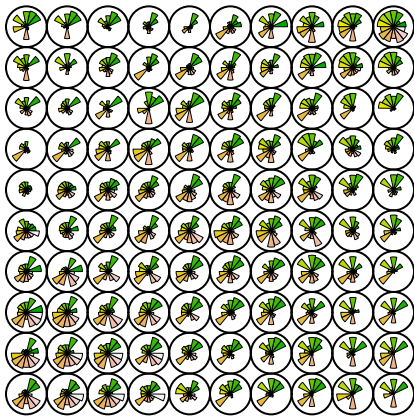
Nodes that have darker colors mean that the nodes have a vector input that is closer, whereas nodes that have lighter colors mean that the nodes have vector inputs that are farther apart. From this model, we can see that a lot of nodes have a vector of input that is closer.















(4) Codes / Weight vectors

The node weight vectors, or “codes”, are made up of normalized values of the original variables used to generate the SOM. Each node’s weight vector is representative / similar to the samples mapped to that node. By visualizing the weight vectors across the map, we can observe patterns in the distribution of samples and variables. The default visualization of the weight vectors is a “fan diagram”, where individual fan representations of the magnitude of each variable in the weight vector are shown for each node.

```
plot(som_fit, type="codes")
```

Codes plot

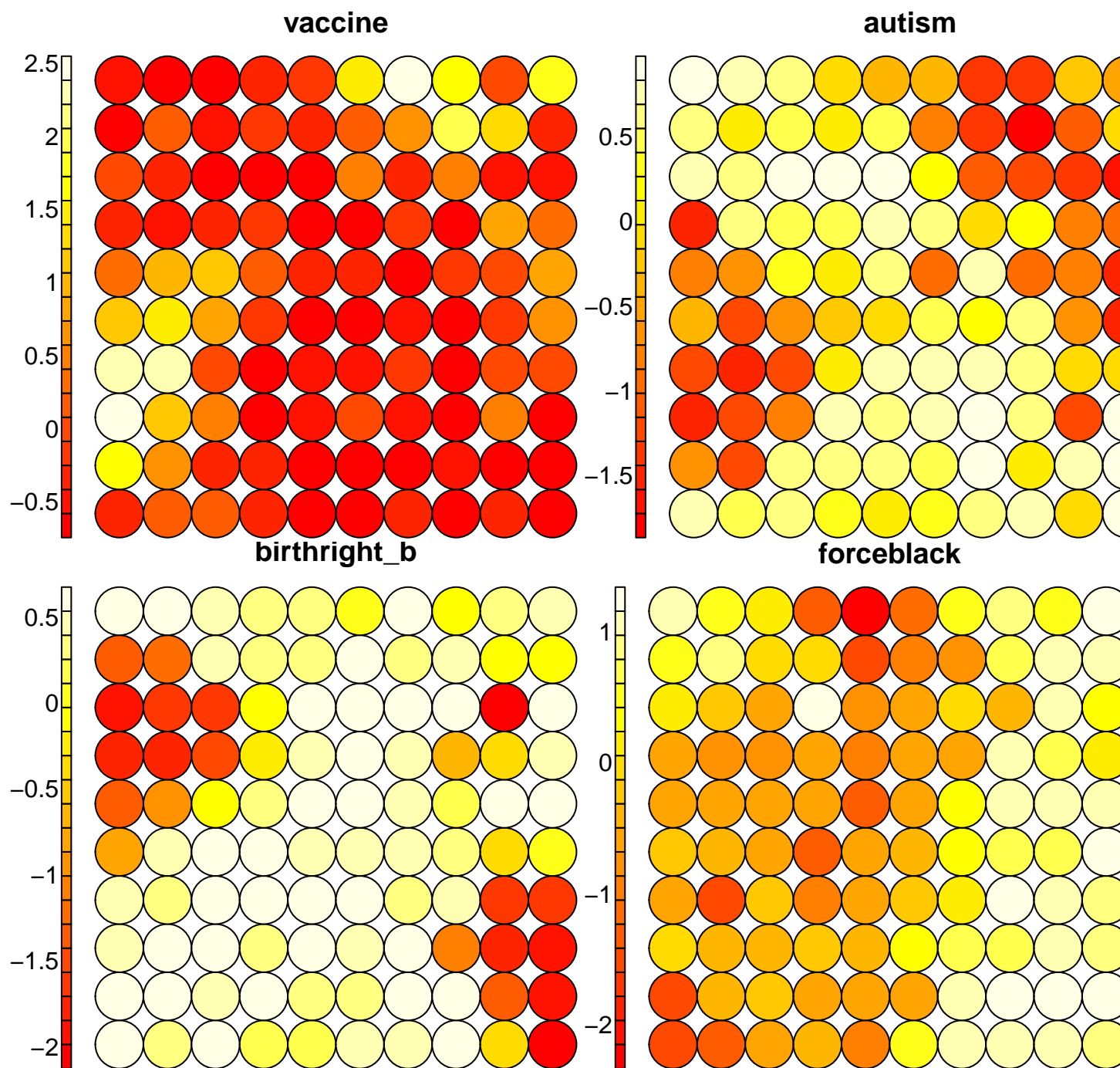


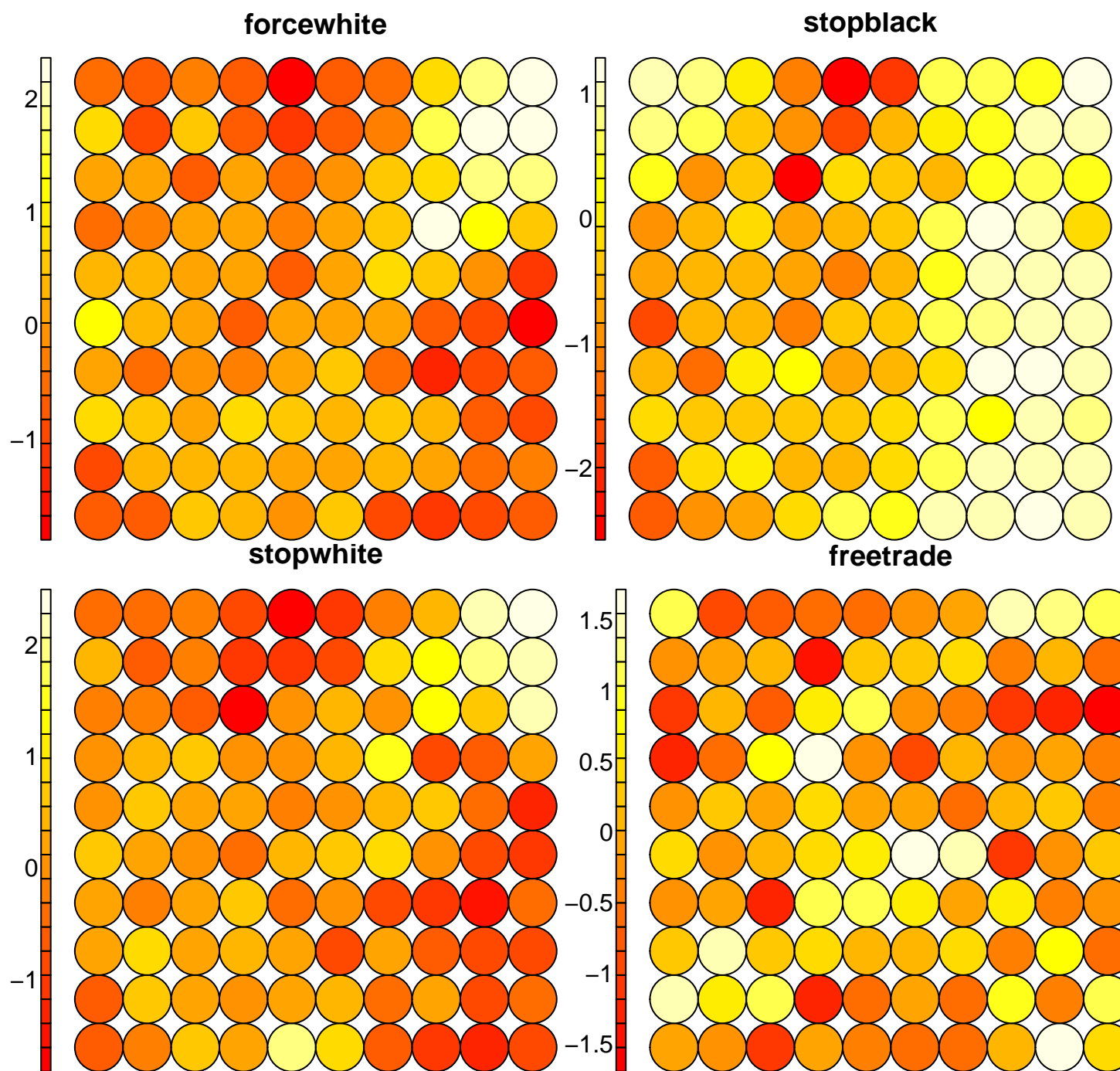
	vaccine		freetrade
	autism		aa3
	birthright_b		warmdo
	forceblack		finwell
	forcewhite		childcare
	stopblack		healthspend
	stopwhite		minu

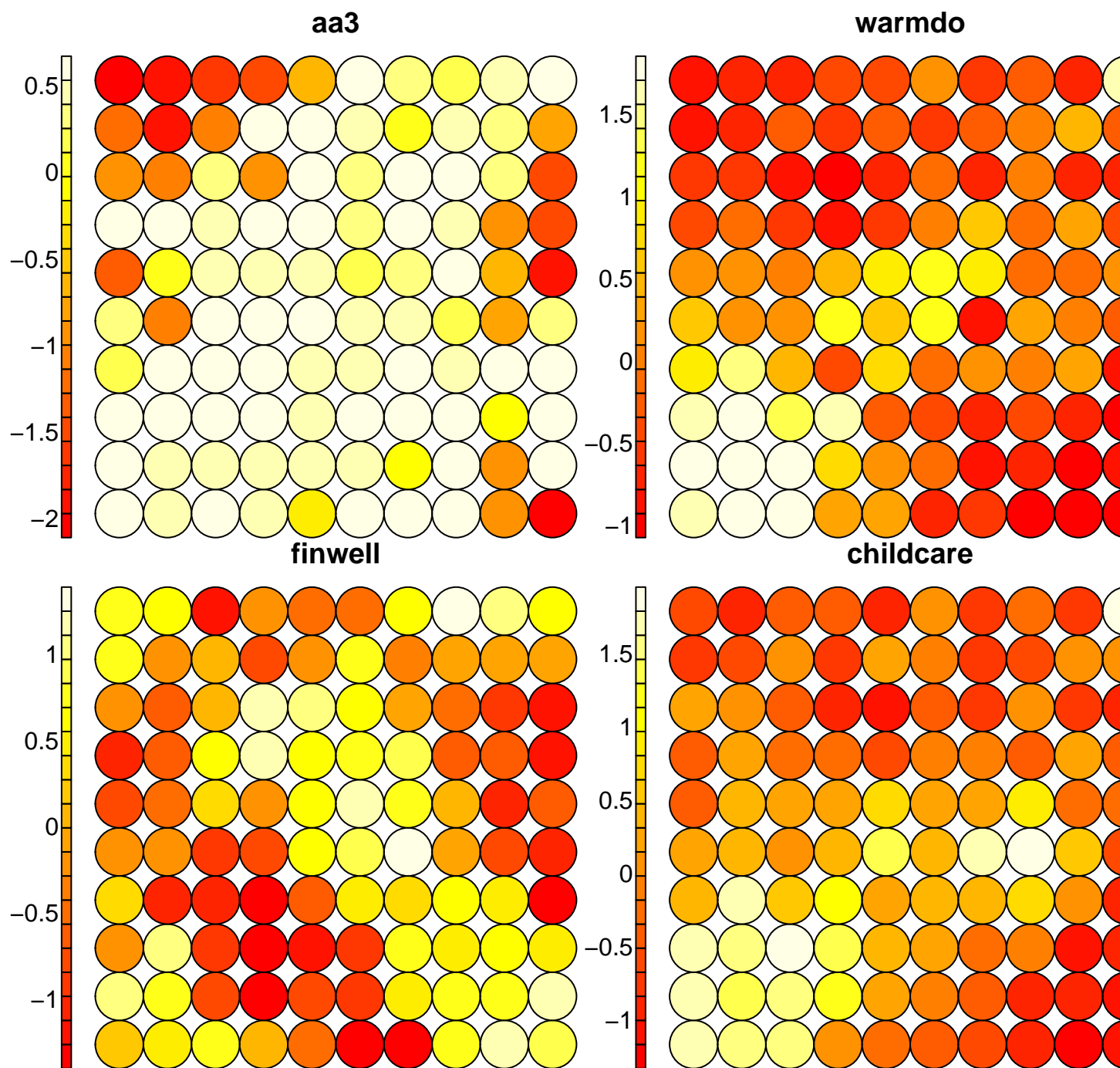
Here we can see that the node will have neighbors that have similar characteristics to it.

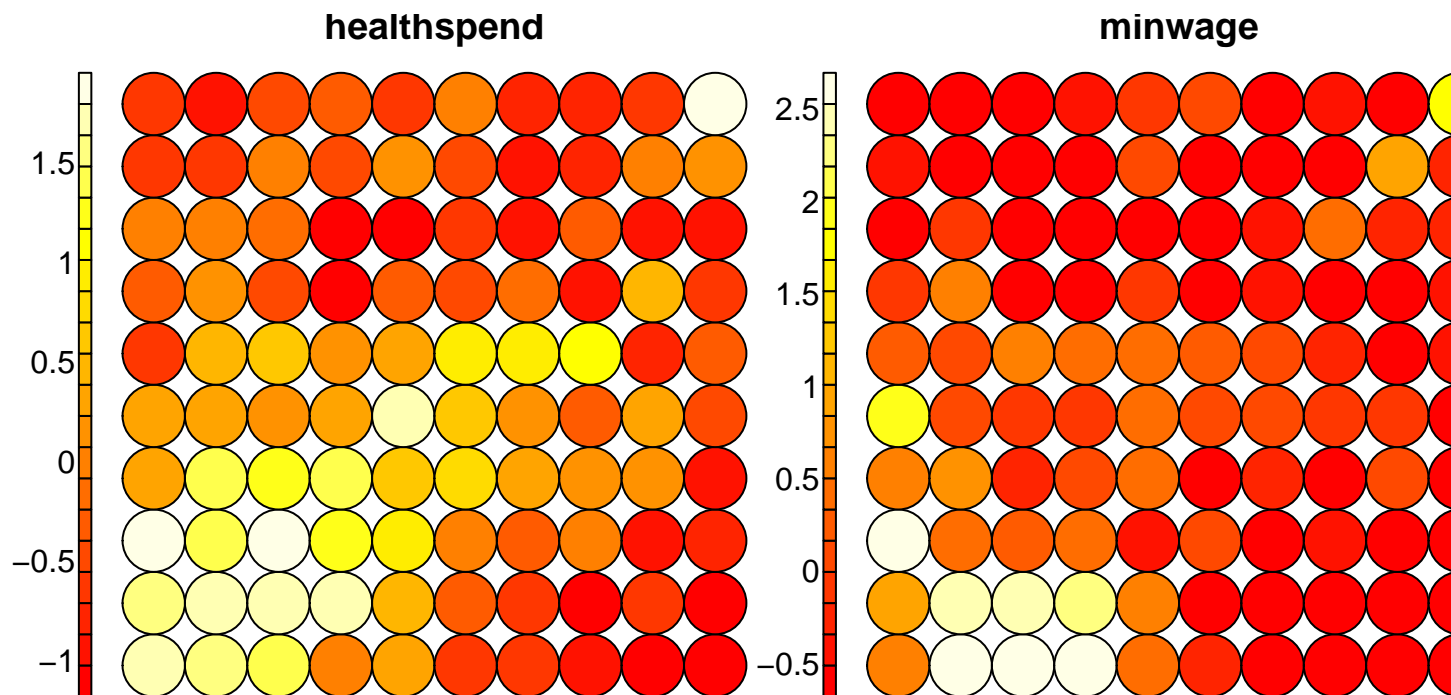
(5) Heatmaps

```
heatmap.som <- function(model){
  for (i in 1:14) {
    plot(model, type = "property", property = getCodes(model)[,i],
         main = colnames(getCodes(model))[i])
  }
}
heatmap.som(som_fit)
```









From the heatmap that is formed, we can know which nodes have the characteristics of each variable whose value is high and the value is low.

Summary: In this question, I offer necessary explanations for each kind of plots. From various visualizations generated by the SOM, we can observe that, when considering the 14 features together, there is no particular dichotomous pattern as we assumed in the clusters.

Question 4: Color the output layer (“mapping”) by your dichotomous feature for weak/strong American identity.

```
point_colors <- c(amerika_palettes$Republican[2],
                 amerika_palettes$Democrat[2])

neuron_colors <- c(amerika_palettes$Republican[3],
                  amerika_palettes$Democrat[3])
```

Now, let's start with a k-means algorithm to the codes data, searching for two clusters given the dichotomous weak/strong American identity feature.

```
## k-means
kmeans_clusters <- som_fit$codes[[1]] %>%
  kmeans(., centers = 2)

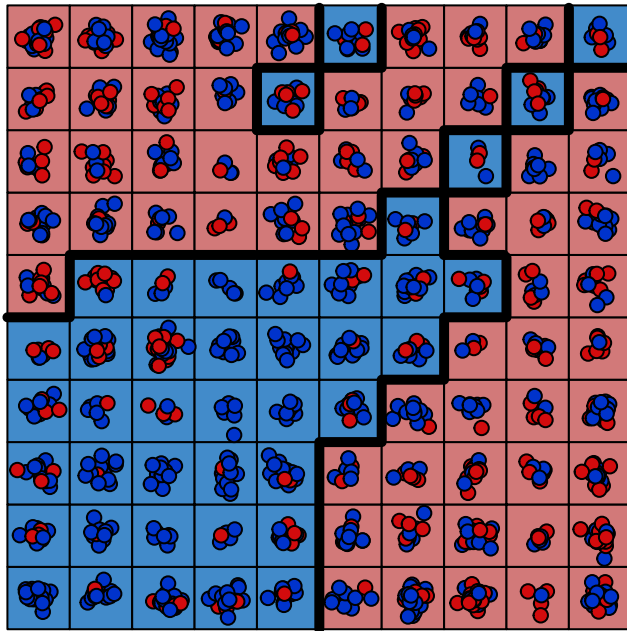
# Then, derive cluster labels (1 or 2) for the clusters.
class_assign_km <- map_dbl(kmeans_clusters$cluster, ~{
  if(. == 1) 2
  else 1
})
```

Finally, plot accordingly varying color by American identity. The expectation is that the majority of observations (points) should correspond to the color of the nodes (background color) found from the given clustering algorithm. If points are scattered and not clearly coordinating, then this would suggest there is

not clear separation in the output layer.

```
plot(som_fit,
     type = "mapping",
     pch = 21,
     bg = point_colors[as.factor(anes_short$strong_amer_ident)],
     shape = "straight",
     bgcol = neuron_colors[as.integer(class_assign_km)],
     main = "2 clusters via k-means");
add.cluster.boundaries(x = som_fit, clustering = class_assign_km,
                      lwd = 5, lty = 5)
```

2 clusters via k-means



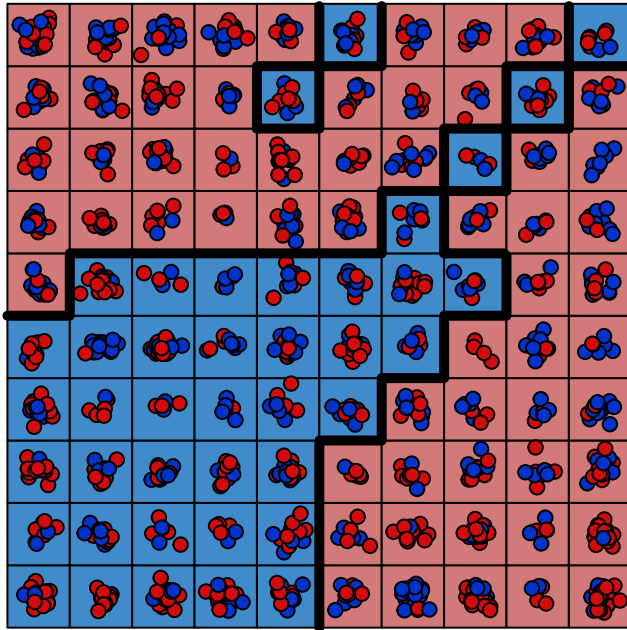
It's hard to observe similarities belonging to each color region. There are some but overall unclear structures from this graph. We can see that for each region with the same background color, the composition of American identity is ununiform and disorganized. If there does exist any difference in American identity considering all the 14 questions together, we would observe a uniform composition for each region. Therefore, from a holistic perspective with all the 14 questions incorporated, it is hard to say that we capture any obvious structures on American identity, at least visually, according to the SOM.

Question 5: Color the output layer (“mapping”) by your dichotomous feature for weak/strong race identity.

Plot accordingly varying color by race identity.

```
plot(som_fit,
     type = "mapping",
     pch = 21,
     bg = point_colors[as.factor(anes_short$strong_race_ident)],
     shape = "straight",
     bgcol = neuron_colors[as.integer(class_assign_km)],
     main = "2 clusters via k-means");
add.cluster.boundaries(x = som_fit, clustering = class_assign_km,
                      lwd = 5, lty = 5)
```

2 clusters via k-means



There is no clear structure from this graph. We can see that for each region with the same background color, the composition of race identity is ununiform and disorganized. If there does exist any race identity difference considering all the 14 questions together, we would observe a uniform composition for each region. Therefore, from a holistic perspective with all the 14 questions incorporated, it is hard to say that we capture any obvious differences on race identity, at least visually, according to the SOM. Points are scattered and not clearly coordinating, then this would suggest there is not clear separation in the output layer. Among 14 questions, there are some relevant to people's sense of their racial identities, such as **forceblack**, **stopblack**, but other questions are quite irrelevant to racial identities. I guess that is the reason why we are unable to observe clear separations from the output layer.

Question 6: Offer a few concluding thoughts comparing the patterns in responses to social questions across these two conceptualizations of identity.

I feel that the structure is a bit clearer in grouping along respondents with similar senses of American identity than that of race identity because points are more scattered and coordinated. I guess people with same sense of American identity may have more similar perceptions about their lives in this country in relative to race identity. However, both of these two visualizations are not clearly organized and therefore cannot reflect the true structures that we want to see. Overall, there is no obvious pattern on grouping along respondents with similar senses of American/ race identity in these 14 questions. The reason I guess is that 14 questions are too many, especially when we aggregate them together to assess the grouping. Maybe we can observe clearer grouping along American or race identity if we only look at a smaller set of questions. Fourteen questions are too complicated and irrelevant to find a pattern.