STAT 479 HW4

Yifan Zhang

Problems

```
#load some packages and libraries
library("dslabs")
library("dplyr")
library("tidyr")
library("ggplot2")
library("naniar")
library("UpSetR")
library("rpart")
library("simputation")
library("readr")
library("tsibble")
library("tsibbledata")
library("lubridate")
library("fpp2")
library("feasts")
library("stringr")
library("grid")
library("gridExtra")
library("ggmap")
library("ggrepel")
library("sf")
library("raster")
library("RStoolbox")
library("scico")
library("ggraph")
library("tidygraph")
library("networkD3")
library("igraph")
library("ggnetwork")
library("superheat")
library("tibble")
library("cluster")
library("tidymodels")
library("viridis")
library("ggpubr")
library("robservable")
library("tidytext")
library("tm")
library("gutenbergr")
library("topicmodels")
library("forcats")
library("embed")
```

```
theme_set(theme_graph())
theme_set(theme_graph())
theme_set(theme_minimal())
theme_set(theme_bw())
```

Polio incidence

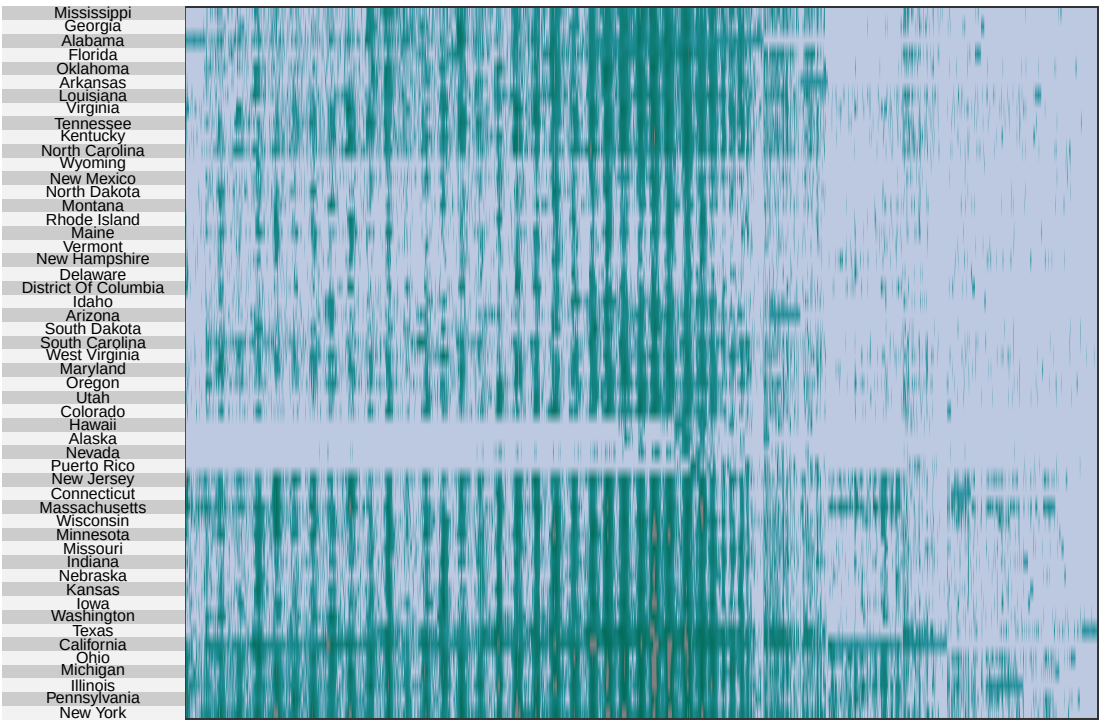
In this problem, we will use a heatmap to visualize a large collection of time series. The data, prepared by the Tycho Project, contain weekly counts of new polio cases in the United States, starting as early as 1912 (for some states).

a. Pivot the raw dataset so that states appear along rows and weeks appear along columns. Fill weeks that don't have any cases with 0's.

```
polio <- read_csv("https://uwmadison.box.com/shared/static/nm7yku4y9q7ylvz5kbxya3ouj2njd0x6.csv")
polio_new <- polio %>%
  pivot_wider(names_from = period_start_date, values_from = cases, values_fill = 0) %>%
  column_to_rownames(var = "state")
```

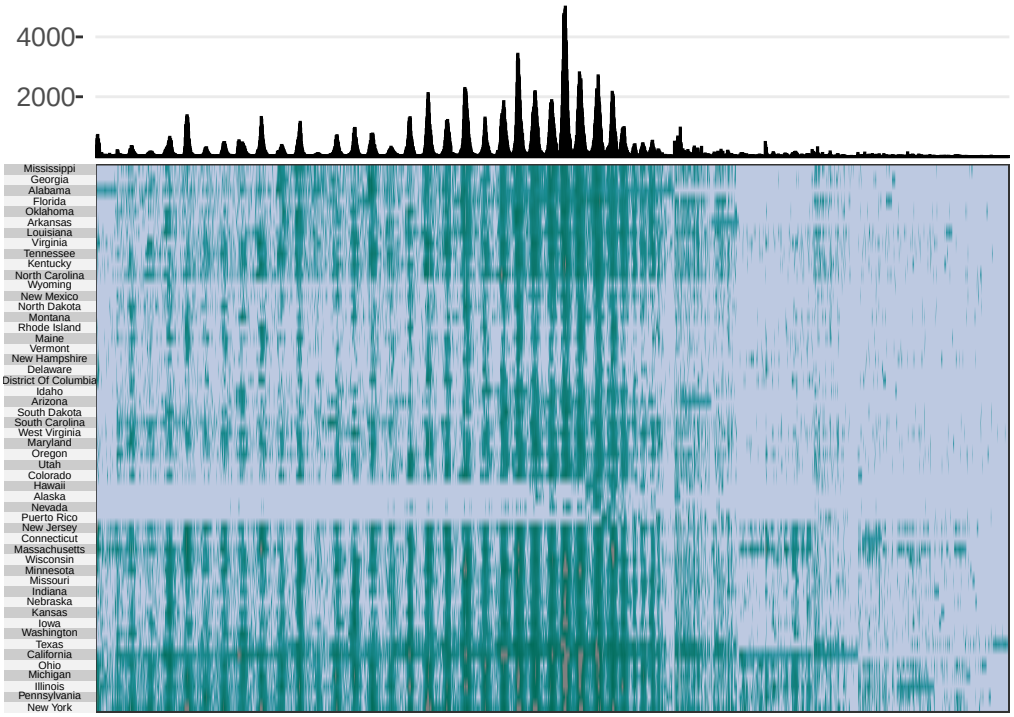
b. Use the `superheat` package to make a heatmap of the data from (a). Have the color of each tile represent `log(1 + cases)`, rather than the raw counts. Reorder the states using a hierarchical clustering by setting `pretty.order.rows = TRUE`.

```
cols <- c('#f6eff7','#bdc9e1','#67a9cf','#1c9099','#016c59')
polio_mat <-log(polio_new +1)
superheat(
  polio_mat,
  left.label.text.size = 2,
  heat.pal = cols,
  heat.lim = c(0, 5),
  pretty.order.rows = TRUE,
  legend = F
)
```



c. Supplement the view from part (b) with a barchart showing the US total incidence during every given week. Interpret the resulting visualization. *Hint: use the* `yt` *argument of superheat.*

```
superheat(
  polio_mat,
  left.label.text.size = 1.5,
  heat.pal = cols,
  heat.lim = c(0, 5),
  pretty.order.rows = TRUE,
  legend = FALSE,
  yt = colSums(polio_new),
  yt.plot.type = "bar",
  yt.bar.col = "black",
  yt.axis.name = ""
)
```



Interpretation: Observing the heat map and the bar plot, the tiles have more density, the corresponding bar will be higher. This is because more density of tiles means the higher number of cases in each state during one week. Therefore, if each state has more cases in that week, the total number of cases in the whole United States will be more, which means the bar will be higher. According to the heatmap in 1c, we find the most cases happen in the middle of this period, which is more than 4000 in the States.

d. Describe types of annotation would improve the informativeness of the plot made in part (c). Also, describe one advantage and one disadvantage of visualizing a collection of time series as a heatmap, instead of as a collection of line plots.

I think which week it is will be a good annotation reflected in the x-axis in the above heatmap, which let us know which exactly week has the most number of cases. I think in this case, visualizing a collection of time series as a heatmap is more clear than a collection of line plots. There are a lot of rows in this data set, if we make line plots, which each line represents one state, there will also be a lot of lines, which may be very messy and difficult for us to know the situation of each state. However, one disadvantage of heatmap is that we cannot identify the exactly number

Silhouette statistics simulation

This problem uses simulation to build intuition about silhouette statistics. The function below simulates a mixture of three Gaussians, with means evenly spaced out between `(start, 0)` and `(end, 0)`. See Figure 2 for an example simulated dataset. We will investigate what happens to the silhouette statistics for each point as the three clusters are made to gradually overlap.

```
library("MASS")
library("tibble")
library("purrr")
```

```r
mixture_data <- function(start = 0, end = 10, K = 3, n = 100) {
  mu <- cbind(seq(start, end, length.out = K), 0)
  map_dfr(1:K, ~ mvrnorm(n, mu[., ], diag(2)) %>% as_tibble())
}

ggplot(mixture_data()) +
  geom_point(aes(x = V1, y = V2)) +
  coord_fixed() +
  labs(x = "x", y = "y") +
  theme(
    panel.background = element_rect(fill = "#f7f7f7"),
    panel.border = element_rect(fill = NA, color = "#0c0c0c", size = 0.6),
    panel.grid.minor = element_blank()
  )
```

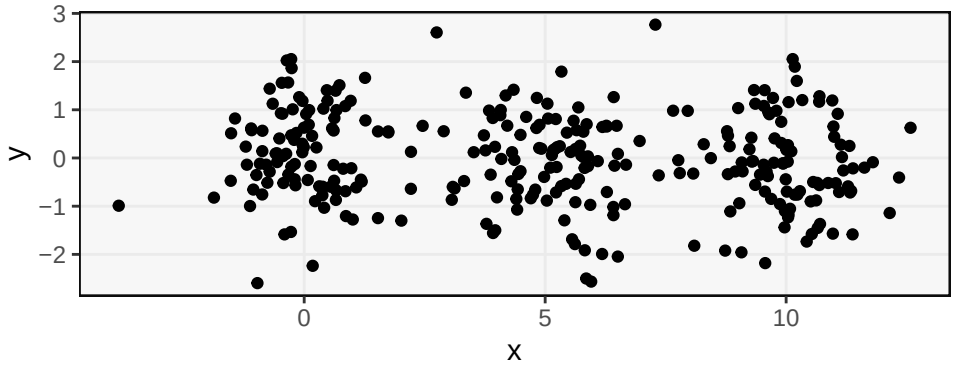


Figure 1: An example simulated dataset from problem 2.

a. Simulate versions of these data where the `end` argument is decreased from 10 to 0.

```r
d1 <- mixture_data(end = 10)
d2 <- mixture_data(end = 8)
d3 <- mixture_data(end = 6)
d4 <- mixture_data(end = 4)
d5 <- mixture_data(end = 2)
d6 <- mixture_data(end = 0)
```

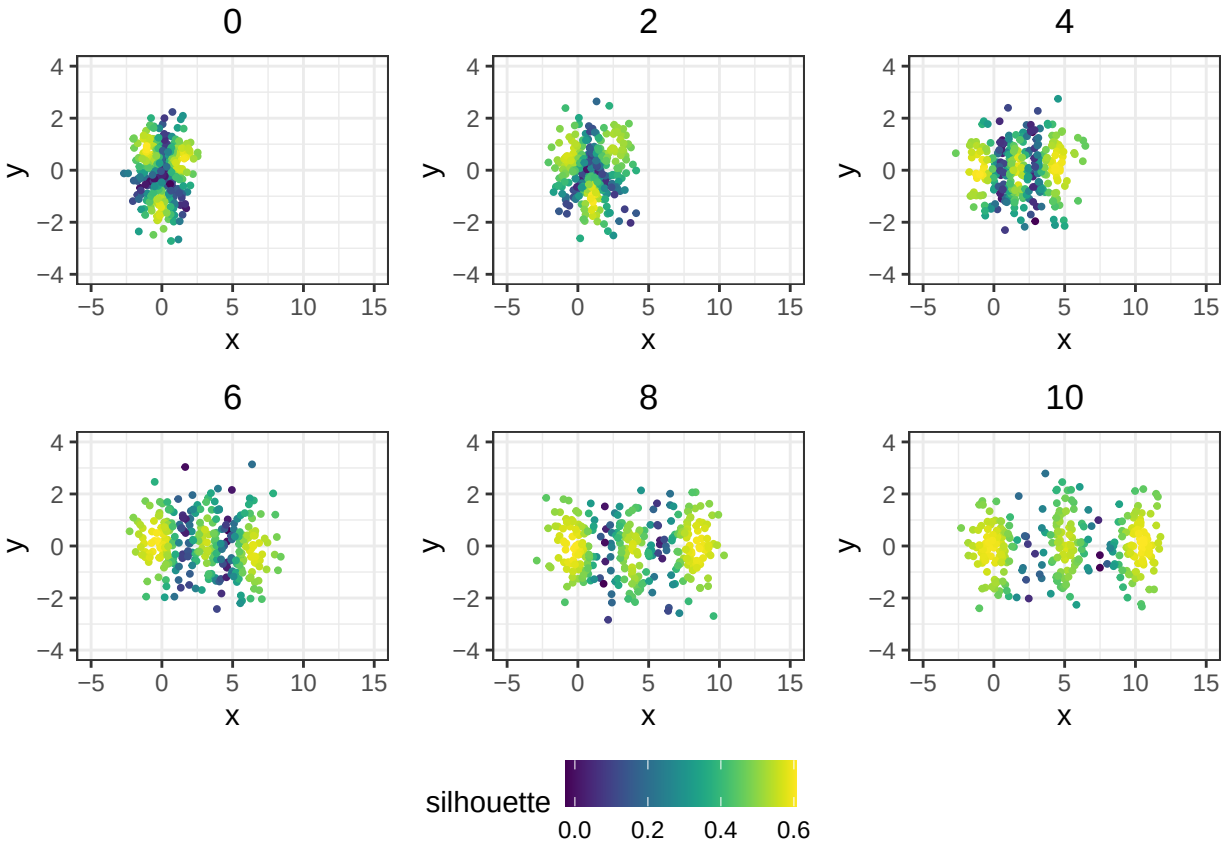
b. Write a function that performs K-means and computes the silhouette statistic given any one of the datasets generated in part (a). Use this function to compute the silhouette statistics for each point in the simulated datasets.

```r
#Write a function
f <- function(x, K) {
  kmeans(x, center = K) %>%
    augment(x) %>% # creates column ".cluster" with cluster label
    mutate(silhouette = silhouette(as.integer(.cluster), dist(x))[, "sil_width"])
}
s1 <- f(d1, 3)
s2 <- f(d2, 3)
s3 <- f(d3, 3)
s4 <- f(d4, 3)
s5 <- f(d5, 3)
s6 <- f(d6, 3)
```

c. Visualize the silhouette statistics from part (b) overlaid onto the simulated data. Discuss the results.

```
p1 <- ggplot(s1,aes(x = V1, y = V2, col = silhouette)) +
  geom_point(size = 0.7) +
  xlim(-5, 15) +
  ylim(-4, 4) +
  scale_color_viridis() +
  labs(
    x = "x",
    y = "y",
    title = "10"
  ) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(hjust = 0.5)
  )

p2 <- ggplot(s2,aes(x = V1, y = V2, col = silhouette)) +
  geom_point(size = 0.7) +
  xlim(-5, 15) +
  ylim(-4, 4) +
  scale_color_viridis() +
  labs(
    x = "x",
    y = "y",
    title = "8"
  ) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(hjust = 0.5)
  )

p3 <- ggplot(s3,aes(x = V1, y = V2, col = silhouette)) +
  geom_point(size = 0.7) +
  xlim(-5, 15) +
  ylim(-4, 4) +
  scale_color_viridis() +
  labs(
    x = "x",
    y = "y",
    title = "6"
  ) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(hjust = 0.5)
  )

p4 <- ggplot(s4,aes(x = V1, y = V2, col = silhouette)) +
  geom_point(size = 0.7) +
  xlim(-5, 15) +
  ylim(-4, 4) +
  scale_color_viridis() +
  labs(
    x = "x",
```

```
    y = "y",
    title = "4"
  ) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(hjust = 0.5)
  )

p5 <- ggplot(s5,aes(x = V1, y = V2, col = silhouette)) +
  geom_point(size = 0.7) +
  xlim(-5, 15) +
  ylim(-4, 4) +
  scale_color_viridis() +
  labs(
    x = "x",
    y = "y",
    title = "2"
  ) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(hjust = 0.5)
  )

p6 <- ggplot(s6,aes(x = V1, y = V2, col = silhouette)) +
  geom_point(size = 0.7) +
  xlim(-5, 15) +
  ylim(-4, 4) +
  scale_color_viridis() +
  labs(
    x = "x",
    y = "y",
    title = "0"
  ) +
  theme(
    legend.position = "bottom",
    plot.title = element_text(hjust = 0.5)
  )

ggarrange(p6, p5, p4, p3, p2, p1, nrow = 2, ncol = 3, common.legend = T, legend = "bottom")
```



Discussion: As the "end" increases, the silhouette statistics of each cluster is larger and larger, because yellow means large silhouette statistics.This means that as the "end" increases. each observation is strongly tied to its assigned cluster, and each cluster is better defined and more isolated to other clusters.

Taxi trips

In this problem, we will use hierarchical clustering to find typical taxi trip trajectories in Porto, Portugal. The data are a subset from the the ECML / PKDD 2015 Challenge – the link provides a complete data dictionary. We have preprocessed it into two formats. The first (`wide`) includes each taxi trip on its own row, with latitude and longitude coordinates along the journey given as separate columns (`x_0, y_0` is the origin of the trip and `x_15`, `y_15` is the destination). The second (`long`) format spreads each point of the journey into a separate row.

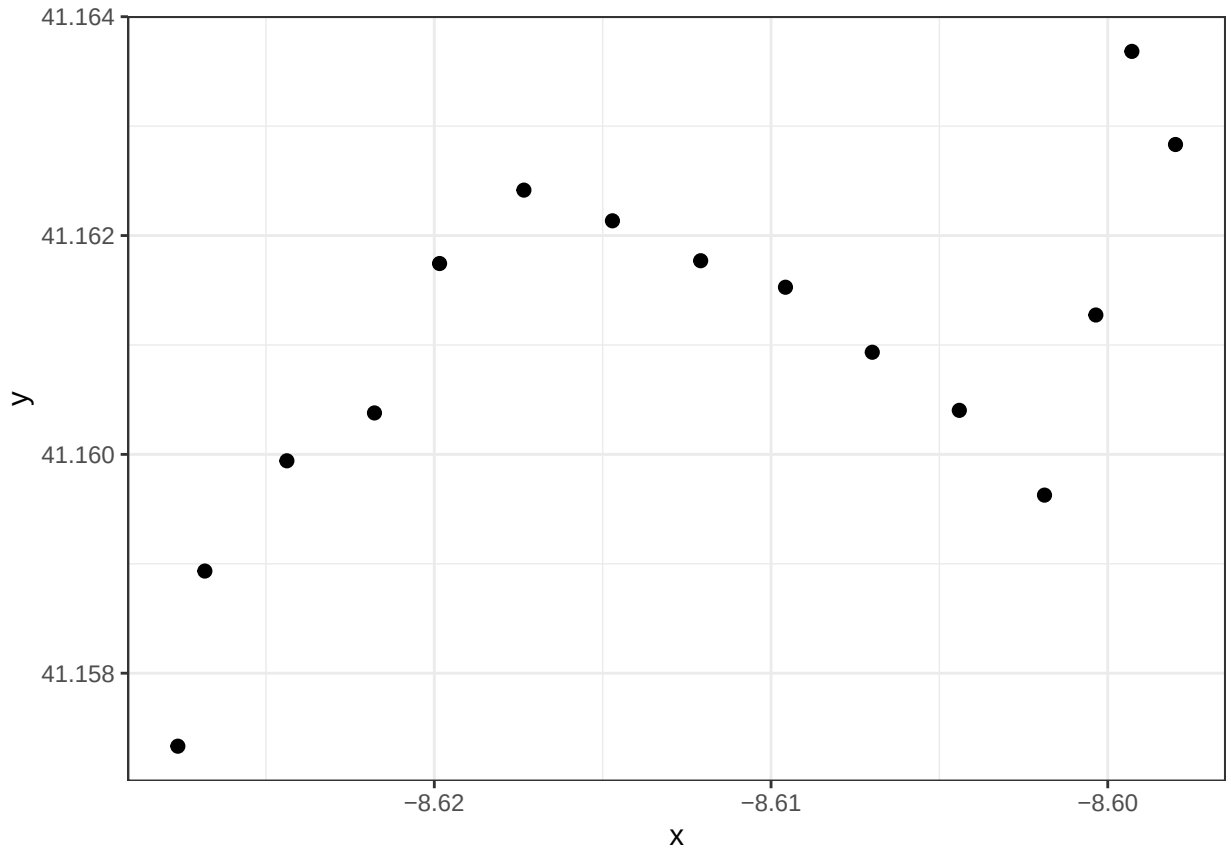
a. Filter the `long` form of the data down to trip `1389986517620000047` and plot its trajectory as a sequence of points.

```
trips <- read_csv("https://uwmadison.box.com/shared/static/098cjaetm8vy0mufq21mc8i9nue2rr2b.csv", col_ty
trips_wide <- read_csv("https://uwmadison.box.com/shared/static/cv0lij4d9gn3s8m2k98t2ue34oz6sbj5.csv",

trips_filter <- trips %>%
  filter(TRIP_ID == "1389986517620000047")

head(trips_filter)
```

```
## # A tibble: 6 x 14
##   TRIP_ID       CALL_TYPE TAXI_ID TIMESTAMP DAY_TYPE trajectory_elem     x     y
##   <chr>         <chr>       <dbl>     <dbl> <chr>              <dbl> <dbl> <dbl>
## 1 138998651762~ B          2.00e7    1.39e9 A                      0 -8.63  41.2
```

```
## 2 138998651762~ B          2.00e7    1.39e9 A                     1 -8.63  41.2
## 3 138998651762~ B          2.00e7    1.39e9 A                     2 -8.62  41.2
## 4 138998651762~ B          2.00e7    1.39e9 A                     3 -8.62  41.2
## 5 138998651762~ B          2.00e7    1.39e9 A                     4 -8.62  41.2
## 6 138998651762~ B          2.00e7    1.39e9 A                     5 -8.62  41.2
## # ... with 6 more variables: n_rows <dbl>, date <dttm>, day_of_week <dbl>,
## #   year <dbl>, month <dbl>, hour <dbl>
```

```
ggplot(trips_filter,aes(x = x, y = y)) +
  geom_point(size = 2) +
  labs(
    x = "x",
    y = "y"
  ) +
  theme(
    legend.position = "bottom"
  )
```



b. We could hierarchically cluster rows in either the `wide` or the `long` format datasets. How would the interpretation of the results differ between the two approaches?

The wide one may be clustered by the trajectory, such as the passed locations of one trip.Trips may be first clustered by y0, then under each cluster of y0, they are clustered by y1... The long one may be clustered by other properties. For example, trips are firstly clustered by Call type, then under each cluster of call type, trips are then clustered by day type, then day of week...

c. Compute a hierarchical clustering of the `wide` format data, using only the columns starting with `x` or `y` as features.

```
D <- trips_wide %>%
  column_to_rownames(var = "TRIP_ID") %>%
  dist()

hclust_result <- hclust(D)
hclust_result
```

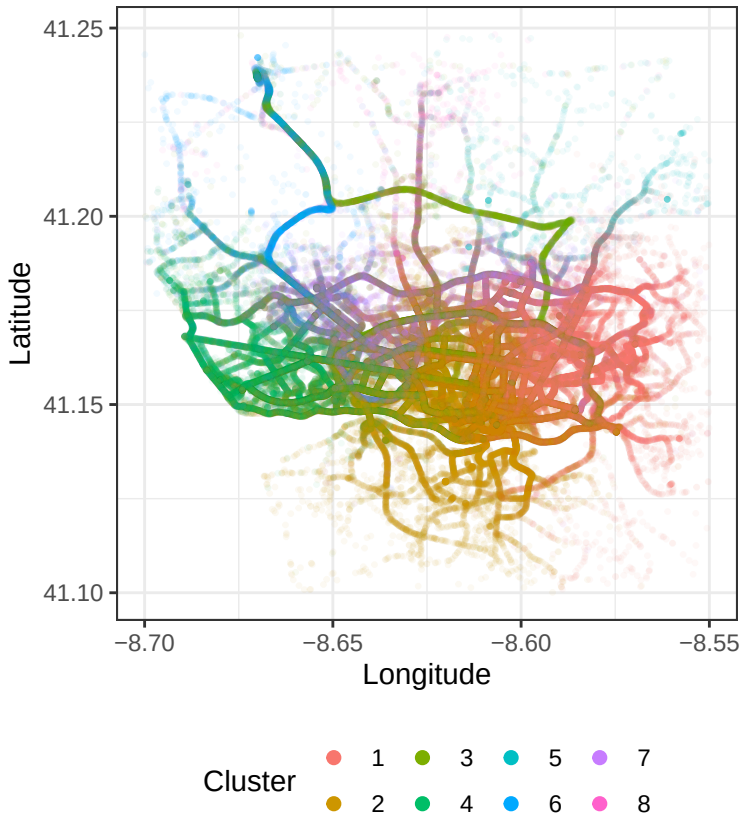
```
##
## Call:
## hclust(d = D)
##
## Cluster method   : complete
## Distance         : euclidean
## Number of objects: 23560
```

d. Cut the hierarchical clustering tree so that 8 clusters are produced. Visualize the trajectories of the taxi trips either colored or faceted by their cluster.

```
cluster_df <- cutree(hclust_result, k = 8) %>%
  tibble("TRIP_ID" = names(.), cluster = as.factor(.))
cluster_df
```

```
## # A tibble: 23,560 x 3
##        . TRIP_ID             cluster
##    <int> <chr>               <fct>
##  1     1 1389986517620000047 1
##  2     2 1382362049620000295 2
##  3     3 1395050501620000440 3
##  4     2 1402055282620000506 2
##  5     4 1389270660620000038 4
##  6     2 1390060716620000675 2
##  7     2 1385469554620000900 2
##  8     4 1374491431620000192 4
##  9     3 1377178428620000329 3
## 10     2 1392222164620000621 2
## # ... with 23,550 more rows
```

```
trips %>%
  left_join(cluster_df, by = "TRIP_ID") %>%
  ggplot(aes(x = x, y = y, col = cluster)) +
  geom_jitter(size = 0.5, alpha = 0.05) +
  labs(
    x = "Longitude",
    y = "Latitude",
    color = "Cluster"
  ) +
  theme(
    legend.position = "bottom"
  ) +
  guides(colour = guide_legend(override.aes = list(size=2, alpha = 1))) +
  coord_fixed()
```



Food nutrients

This problem will use PCA to provide a low-dimensional view of a 14-dimensional nutritional facts dataset. The data were originally curated by the USDA and are regularly used in visualization studies.

```
nutrients <- read_csv("https://uwmadison.box.com/shared/static/nmgouzobq5367aex45pnbzgkhm7sur63.csv")
```

a. Define a tidymodels `recipe` that normalizes all nutrient features and specifies that PCA should be performed.

```
nutrients_rec <- recipe(~.,data = nutrients) %>%
  update_role(id:group_lumped, new_role = "id") %>%
  step_normalize(all_predictors())%>%
  step_pca(all_predictors())

nutrients_prep <- prep(nutrients_rec)
```

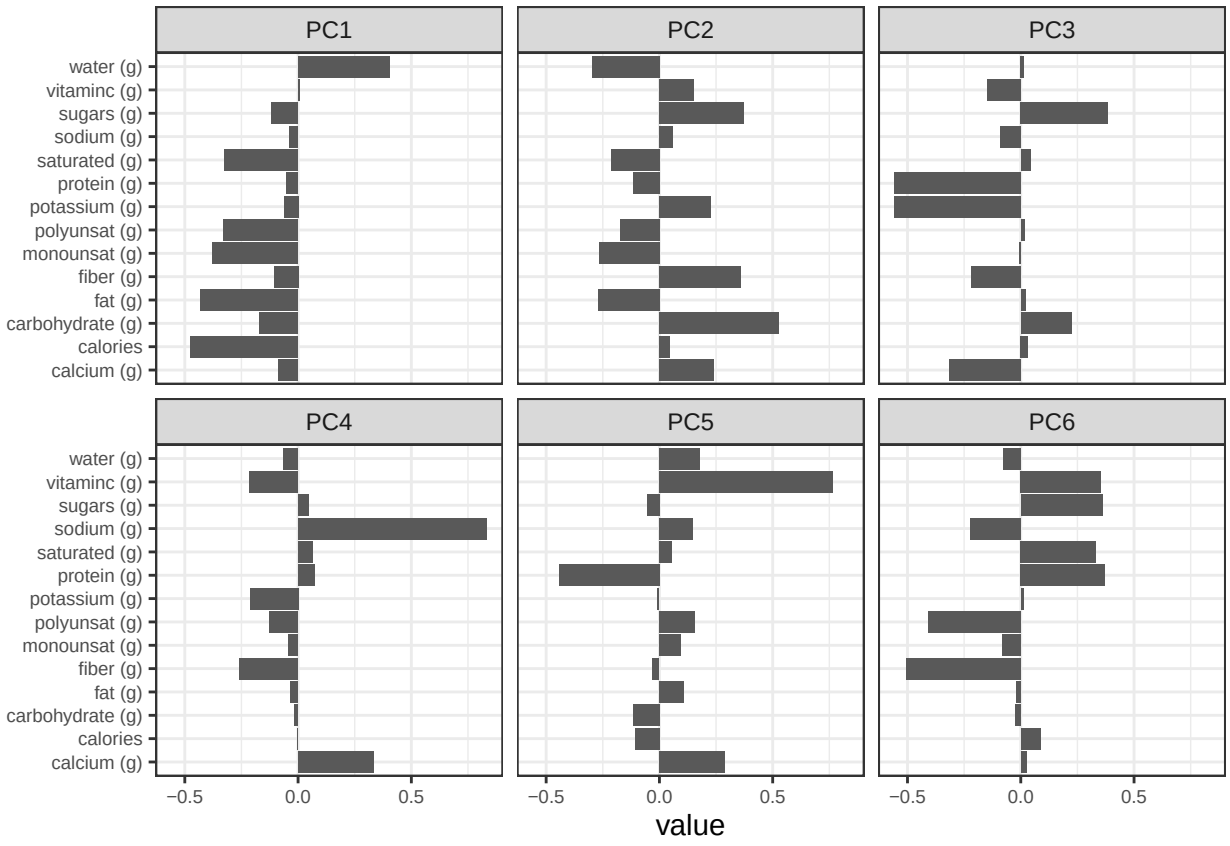
b. Visualize the top 6 principal components. What types of food do you expect to have low or high values for PC1 or PC2?

```
components <- tidy(nutrients_prep, 2) %>%
   filter(component %in% str_c("PC", 1:6))
head(components)
```

```
## # A tibble: 6 x 4
##   terms            value component id
##   <chr>            <dbl> <chr>     <chr>
## 1 protein (g)   -0.0525  PC1       pca_dcw9m
## 2 calcium (g)   -0.0865  PC1       pca_dcw9m
```

```
## 3 sodium (g)    -0.0376  PC1       pca_dcw9m
## 4 fiber (g)     -0.106   PC1       pca_dcw9m
## 5 vitaminc (g)   0.00785 PC1       pca_dcw9m
## 6 potassium (g) -0.0626  PC1       pca_dcw9m
```

```
ggplot(components, aes(value, terms)) +
  geom_col(show.legend = FALSE) +
  facet_wrap(~component, nrow = 2) +
  labs(y = NULL) +
  theme(axis.text = element_text(size = 7))
```



Personally I think types such as beverage and fruit will have high values for PC1, and types such as fast foods, sweets or snacks may have high values for PC2.

c. Compute the average value of PC2 within each category of the `group` column. Give the names of the groups sorted by this average.

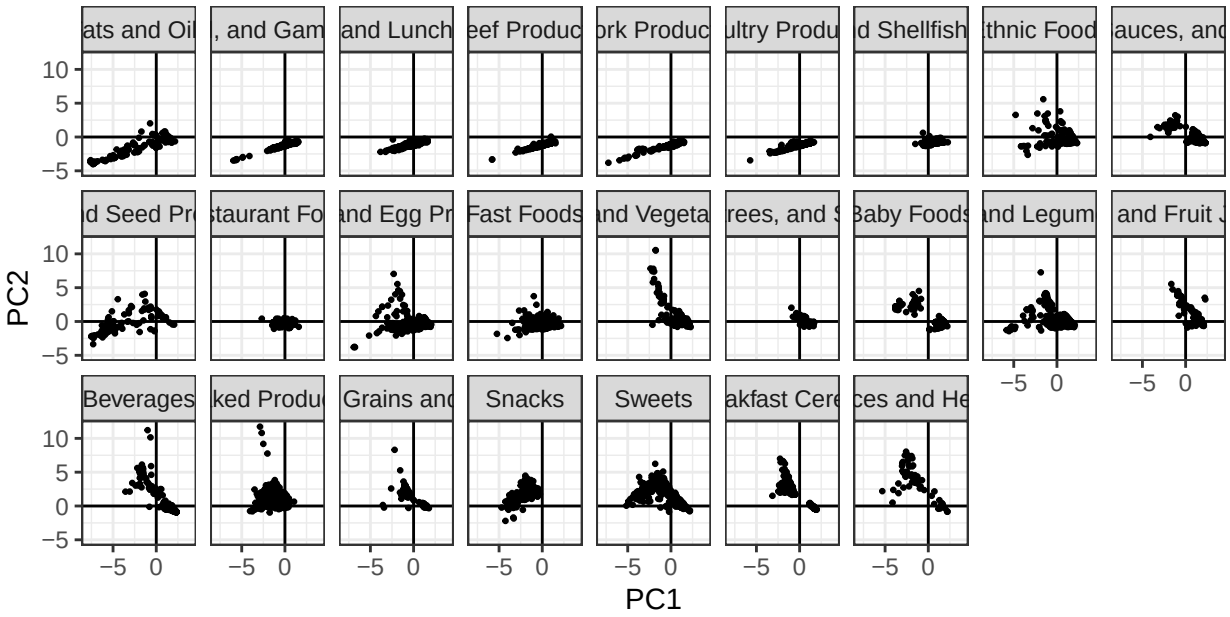
```
average <- juice(nutrients_prep) %>%
  group_by(group) %>%
  summarise(avg = mean(PC2)) %>%
  arrange(avg)
average$group
```

```
##  [1] Fats and Oils                    Lamb, Veal, and Game Products
##  [3] Sausages and Luncheon Meats      Beef Products
##  [5] Pork Products                    Poultry Products
##  [7] Finfish and Shellfish Products   Ethnic Foods
##  [9] Soups, Sauces, and Gravies       Nut and Seed Products
```

```
## [11] Restaurant Foods                    Dairy and Egg Products
## [13] Fast Foods                          Vegetables and Vegetable Products
## [15] Meals, Entrees, and Sidedishes      Baby Foods
## [17] Legumes and Legume Products         Fruits and Fruit Juices
## [19] Beverages                           Baked Products
## [21] Cereal Grains and Pasta             Snacks
## [23] Sweets                              Breakfast Cereals
## [25] Spices and Herbs
## 25 Levels: Baby Foods Baked Products Beef Products ... Vegetables and Vegetable Products
```

d. Visualize the scores of each food item with respect to the first two principal components. Facet the visualization according to the `group` column, and sort the facets according to the results of part (c). How does the result compare with your guess from part (b)?

```
scores <- juice(nutrients_prep)
variances <- tidy(nutrients_prep, 2, type = "variance")
scores$group = factor(scores$group, average$group)

ggplot(scores, aes(PC1, PC2)) +
  xlim(-8, 4) +
  geom_point(size = 0.5) +
  facet_wrap(~group, nrow = 3) +
  coord_fixed(sqrt(variances$value[2] / variances$value[1])) +
  geom_vline(xintercept = 0)+
  geom_hline(yintercept = 0)
```



I think my guess is relativley close to the result except beverages. I expect it to have high values for PC1 but it has higher values for PC2.

Modeling topics in *Pride and Prejudice*

This problem uses LDA to analyze the full text of *Pride and Prejudice*. The code below creates two R objects, `paragraph` and `dtm`. `paragraph` is a data.frame whose rows are paragraphs[1] from the book. `dtm` is a `DocumentTermMatrix` containing word counts across the same paragraphs – the i^{th} row of `dtm` corresponds to the i^{th} row of `paragraph`.

```
library("readr")
library("dplyr")
library("tidytext")

paragraphs <- read_csv("https://uwmadison.box.com/shared/static/pz1lz301ufhbedzsj9iioee77r95xz4v.csv")
dtm <- paragraphs %>%
  unnest_tokens(word, text) %>%
  filter(!word %in% stop_words$word) %>%
  count(paragraph, word) %>%
  cast_dtm(paragraph, word, n)
```

a. Fit an LDA model to `dtm` using 6 topics. Set the seed by using the argument `control = list(seed = 479)` to remove any randomness in the result.

```
paragraph_lda <- LDA(dtm, k = 6, control = list(seed = 479))
```

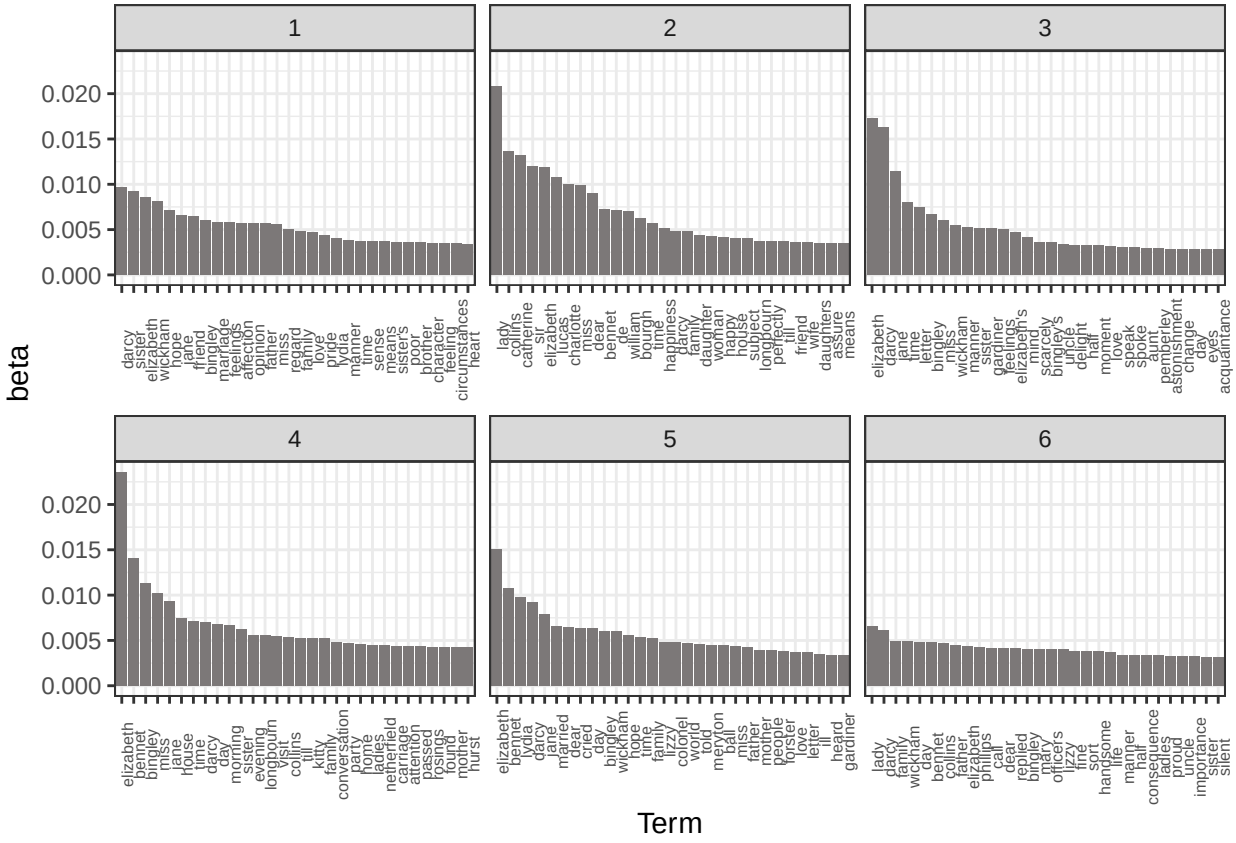
b. Visualize the top 30 words within each of the fitted topics. Specifically, create a faceted bar chart where the heights of the bars correspond to word probabilities and the facets correspond to topics. Reorder the bars so that each topic's top words are displayed in order of decreasing probability.

```
topics <- tidy(paragraph_lda, matrix = "beta")
#memberships <- tidy(chapters_lda, matrix = "gamma")
head(topics)
```

```
## # A tibble: 6 x 3
##   topic term           beta
##   <int> <chr>         <dbl>
## 1     1 considered 2.87e- 3
## 2     2 considered 5.31e-11
## 3     3 considered 1.72e- 4
## 4     4 considered 4.17e- 4
## 5     5 considered 4.29e- 4
## 6     6 considered 3.96e- 8
```

```
top_30_words <- topics %>%
  group_by(topic) %>%
  slice_max(beta, n = 30) %>%
  mutate(term = reorder_within(term, -beta, topic))

ggplot(top_30_words, aes(term, beta)) +
  geom_col(show.legend = F, fill = "#7c7878") +
  scale_x_reordered() +
  facet_wrap(~topic, nrow = 2, scales = "free_x") +
  labs(
    x = "Term"
  )+
  theme(
    axis.text.x = element_text(angle = 90, size = 6)
  )
```

[1] We've filtered very short paragraphs; e.g., from dialogue.



c. Find the paragraph that is the purest representative of Topic 2. That is, if γ_{ik} denotes the weight of topic k in paragraph i, then print out paragraph i^* where $i^* = \arg\max_i \gamma_{i2}$. Verify that the at least a few of the words with high probability for this topic appear. Only copy the first sentence into your solution.

```
memberships <- tidy(paragraph_lda, matrix = "gamma")
gamma <- memberships %>%
  pivot_wider(names_from = topic, values_from = gamma)
which(gamma$`2` == max(gamma$`2`))
```

```
## [1] 347
```

```
paragraphs$text[paragraphs$paragraph == 347]
```

```
## [1] "sir william and lady lucas were speedily applied to for their consent; and it was bestowed with
```

```
#The first sentence:
#sir william and lady lucas were speedily applied to for their consent;
#and it was bestowed with a most joyful alacrity.
```

Single-Cell Genomics

In this problem, we will apply UMAP to a dataset of Peripheral Blood Mononuclear Cells (PBMC) released by 10X Genomics. The first column, `cell_tag`, gives an identifier for each cell in the dataset. All other columns are molecules that were detected in that cell. For example, CD74 is a molecule often found on the surface of T-cells.

a. Define a tidymodels `recipe` that specifies that UMAP should be performed with the parameters `learn_rate = 0.1` and `neighbors = 5`. There is no need to normalize these data, as they have been

normalized in advance using methods tailored to single-cell genomics data.

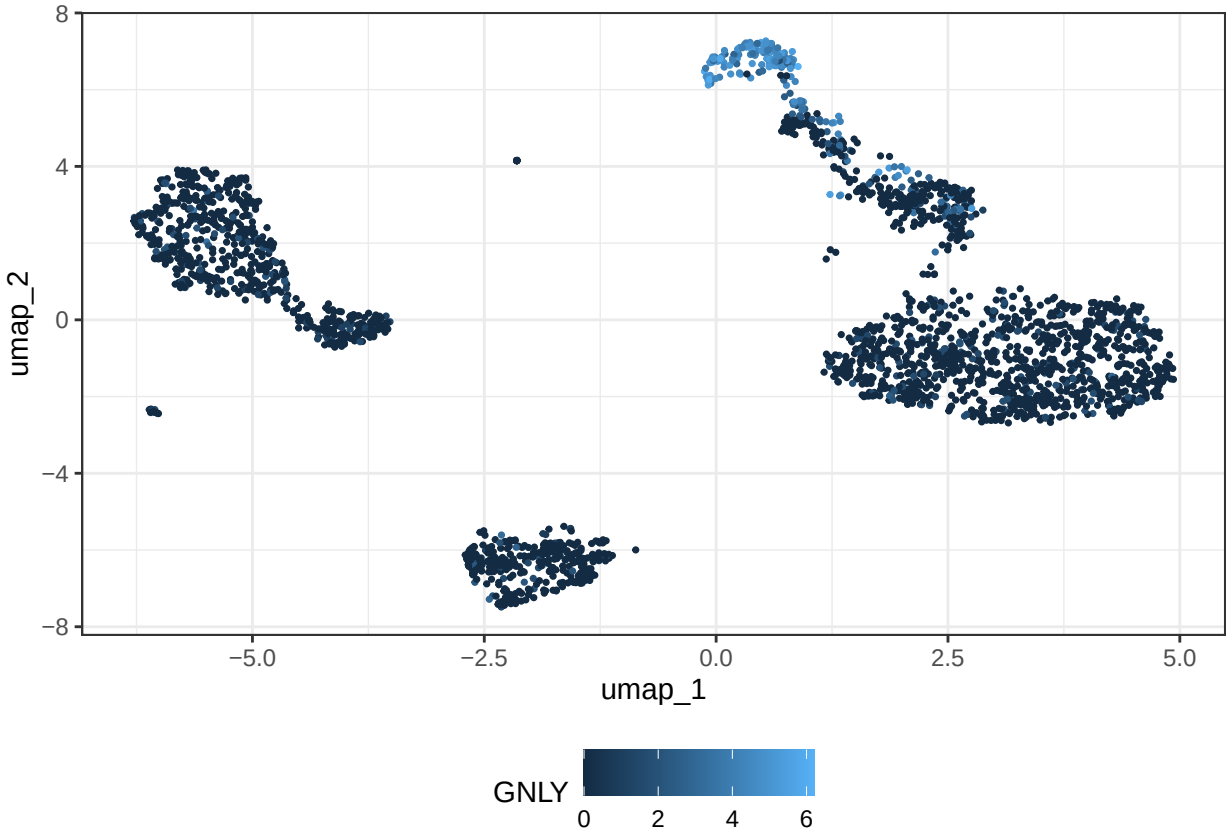
```
pbmc <- read_csv("https://uwmadison.box.com/shared/static/ai539s30rjsw5ke4vxbjrxjaiihq7edk.csv")
pbmc_rec <- recipe(~., data = pbmc) %>%
  update_role(cell_tag, new_role = "id") %>%
  step_umap(all_predictors(), learn_rate = 0.1, neighbors = 5)
pbmc_prep <- prep(pbmc_rec)
```

b. Compute the UMAP embeddings across cells. Color points in by their value of the GNLY molecule.

```
scores <- juice(pbmc_prep) %>%
  left_join(pbmc, by="cell_tag")
head(scores)
```

```
## # A tibble: 6 x 103
##   cell_tag umap_1 umap_2  PPBP S100A9 IGLL5   LYZ  GNLY   FTL   PF4  FTH1 FCER1A
##   <chr>     <dbl>  <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>
## 1 AAACATA~  2.62   2.30   0      0        0  1.64  0     3.92     0  2.23      0
## 2 AAACATT~ -2.32  -6.57   0      0        0  1.96  0     3.06     0  3.57      0
## 3 AAACATT~  1.90  -1.88   0      0        0  2.00  1.43  3.27     0  3.58      0
## 4 AAACCGT~ -4.88   0.910  1.57   3.84     0  4.52  0     5.69     0  5.92      0
## 5 AAACCGT~  0.473  7.21   0      0        0  0     3.45  5.22     0  3.45      0
## 6 AAACGCA~  3.79  -0.785  0      0        0  1.73  0     3.95     0  2.70      0
## # ... with 91 more variables: GNG11 <dbl>, S100A8 <dbl>, HLA.DRA <dbl>,
## #   CD74 <dbl>, CLU <dbl>, GZMB <dbl>, NKG7 <dbl>, C1QA <dbl>, CST3 <dbl>,
## #   CCL4 <dbl>, HLA.DPB1 <dbl>, SDPR <dbl>, C1QB <dbl>, AL928768.3 <dbl>,
## #   TYMS <dbl>, TUBB1 <dbl>, RRM2 <dbl>, GUSB <dbl>, STMN1 <dbl>, MZB1 <dbl>,
## #   C10orf32 <dbl>, GP9 <dbl>, IGJ <dbl>, LYPD2 <dbl>, HAGH <dbl>, TK1 <dbl>,
## #   FUS <dbl>, DSCR3 <dbl>, LYAR <dbl>, SLA <dbl>, SMIM7 <dbl>, KIF5B <dbl>,
## #   HBP1 <dbl>, PPP6C <dbl>, TMX2 <dbl>, HBA1 <dbl>, INTS12 <dbl>,
## #   HLA.DPA1 <dbl>, APOBEC3B <dbl>, KIAA0101 <dbl>, CA2 <dbl>, CCL3 <dbl>,
## #   LILRA4 <dbl>, PRDX1 <dbl>, GIMAP5 <dbl>, HLA.DRB1 <dbl>, CD79A <dbl>,
## #   SERPINF1 <dbl>, MYL9 <dbl>, PRKCB <dbl>, C16orf13 <dbl>, LST1 <dbl>,
## #   MLLT11 <dbl>, BIRC5 <dbl>, TNFRSF17 <dbl>, CCL5 <dbl>, NDUFA12 <dbl>,
## #   THOC7 <dbl>, RABL6 <dbl>, SIVA1 <dbl>, CLDN5 <dbl>, STK38 <dbl>,
## #   SDHB <dbl>, TREML1 <dbl>, UBLCP1 <dbl>, IFI27 <dbl>, SPARC <dbl>,
## #   HIST1H2AC <dbl>, PTGDS <dbl>, C14orf1 <dbl>, RALY <dbl>, IL8 <dbl>,
## #   PRF1 <dbl>, CLEC2B <dbl>, FCGR3A <dbl>, TNFSF10 <dbl>, MED30 <dbl>,
## #   ODC1 <dbl>, STAMBP <dbl>, GMPR <dbl>, GZMH <dbl>, CWC15 <dbl>,
## #   DNASE1L3 <dbl>, TMEM208 <dbl>, FGFBP2 <dbl>, TYROBP <dbl>, CXCL3 <dbl>,
## #   IDH2 <dbl>, ACTB <dbl>, FCER1G <dbl>, NXT2 <dbl>
```

```
ggplot(scores, aes(umap_1, umap_2, col = GNLY)) +
  geom_point(size = 0.6) +
  theme(
    legend.position = "bottom"
  )
```



Feedback

a. How much time did you spend on this homework? 10 hours.
b. Which problem did you find most valuable? The nutrient question. That is related to our daily lives, and I can get some tips to live more healthily from the result.