JSC370 Assignment 2

Michael J. Moon

2020-02-26

Prepare data

Read data

```
> library(vroom)
> # read rna fpkm from gzip file using vroom::vroom to
> # quickly read from multiple files as a tibble
> fpkms <- vroom(Sys.glob("gdc_download*/*/*.gz"), delim = "\t",
+   col_names = c("gene", "rna"), id = "filename")
> # read link data
> links <- read.table("gdc_sample_sheet.2020-01-30.tsv", sep = "\t",
+   header = TRUE)
> # read clinical data
> clinicals <- read.table("clinical.tsv", sep = "\t", header = TRUE)
```

Transform FPKMs

```
> library(tidyverse)
> # it's also a good idea to standardize the rna's before
> # log-transform, but I won't do it here
> fpkms$logrna <- log10(fpkms$rna + 1)
> logrna <- fpkms %>% pivot_wider(id_cols = filename, values_from = logrna,
+   names_from = gene) %>% mutate(case_id = str_split(filename,
+   "/", simplify = TRUE)[, 3])
```

Join data

```
> # Join links data and clinical data
> df <- clinicals %>% select(submitter_id, vital_status, days_to_death,
+   days_to_last_follow_up, gender, age_at_index, tumor_stage) %>%
+   distinct() %>% inner_join(links %>% select(File.Name,
+   Case.ID), by = c(submitter_id = "Case.ID"))
```

Clustering

We will follow the tSNE + HDBSCAN (hierarchical density-based spatial clustering of applications with noise) as outlined in a blog post in Towards Data Science[1]. The blog post also has a comparison between different combinations of dimension reduction methods and clustering methods for gene expressison data if you are interested.

First, we need to specify the minimum cluster size for the HDBSCAN algorithm. We will pick the parameter based on a score function. We define the score function as a fraction of cells with low confidence assignment to a cluster based on 5% confidence threshold. The objective is to pick the number of clusters with the minimum score. That is, we want to minimize the number of cases where we are unsure about their cluster assignments. Since tSNE is a stochastic method, We repeat clustering for each number of clusters multiple times (`N_inter=25`).

```
> library(Rtsne)
> library(dbscan)
> # use parallel processing to speed up the execution
> library(foreach)
> library(doParallel)
> registerDoParallel(cores = 4)
> N_iter <- 25
> N_pt <- 10
> N_cells <- dim(logrna)[1]
> score <- vector(length = N_pt - 2)
> expr <- logrna %>% select(-c(filename, case_id))
> # drop all 0 columns
> expr <- expr %>% select(names(expr)[sapply(expr, function(x) !all(x ==
+   0))])
> expr <- as.matrix(expr)
> # evaluate for 3 to N_pt number of clusters
> for (i in 3:N_pt) {
+   # repeat N_iter times in parallel and get mean score
+   score_iter <- foreach(1:N_iter, .combine = c) %dopar%
+     {
+       tsne_iter <- Rtsne(expr, max_iter = 10000)
+       res <- hdbscan(tsne_iter$Y, minPts = i)
+       score_iter_temp <- sum(res$membership_prob <
+         0.05)/N_cells
+       return(score_iter_temp)
+     }
+   score[i - 2] <- mean(score_iter, na.rm = TRUE)
+ }
> # this cell takes a long time...save the resulting
> # object to avoid running it again
> save(logrna, df, score, N_pt, file = "save_point.RData")
```

```
> load("save_point.RData")
```

Check the `.Rmd` file for configuring Figure blocks that generate captions and labels in LaTeX. In text body, the labels are referenced with `Figure \ref{fig:<label-in-block>}`.

- `fig.cap` (string) Figure caption;

[1] Oskolkov, N. (2019). How to cluster in High Dimensions. Towards Data Science. https://towardsdatascience.com/how-to-cluster-in-high-dimensions-4ef693bacc6

- `fig.subcap` (array) Captions for subfigures if you are placing more than one plot; `\usepackage(subfig)` in the document head is required;
- `fig.ncol` (integer) Number of columns per row when placing subfigures;
- `fig.align` (string) Figure alignment; use “center”;
- `out.width`, `fig.width`, `out.height`, `fig.height` (string or integer) Figure dimension; refer to this blog[2] for details; try different combinations until you are satisfied with the overall size, font size, etc. of the plot displayed on pdf;

```
> # get the clustering size with the minimum score
> opt_num <- which.min(score) + 2  # started from 3
```

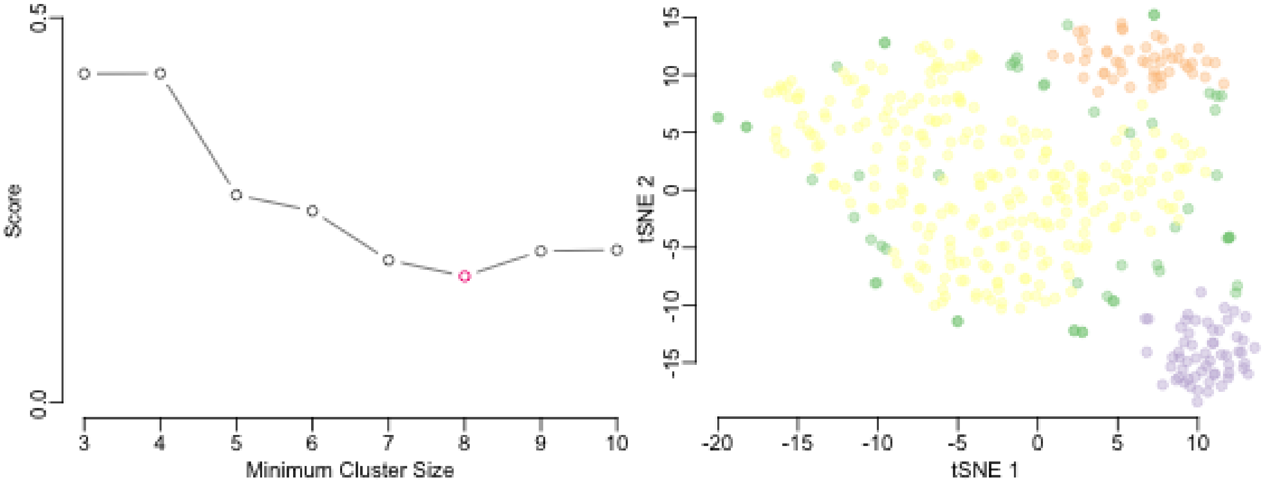
```
> # perform the final tSNE reduction and HDBSCAN will
> # iterate N_tsne times and pick the best tSNE result
> # based on KL divergence
> N_tsne <- 50
> tsne_out <- list(length = N_tsne)
> KL <- vector(length = N_tsne)
> for (k in 1:N_tsne) {
+   tsne_out[[k]] <- Rtsne(expr, max_iter = 10000)
+   # extract KL divergence of the last iteration
+   KL[k] <- tail(tsne_out[[k]]$itercosts, 1)
+ }
> opt_tsne <- tsne_out[[which.min(KL)]]$Y
> res_opt <- hdbscan(opt_tsne, minPts = opt_num)
> # this cell takes a long time...save the resulting
> # object to avoid running it again
> save(opt_tsne, res_opt, file = "save_point2.RData")
```

```
> load("save_point2.RData")
```

```
> # plot the scores
> plot(score ~ seq(from = 3, to = N_pt, by = 1), type = "b",
+   xlab = "Minimum Cluster Size", ylab = "Score", col = ifelse(score ==
+     min(score), COLORS[6], COLORS[8]), axes = FALSE,
+   xlim = c(3, 10), ylim = c(0, 0.5))
> axis(1, at = seq(3, 10))
> axis(2, at = c(0, 0.5))
> # plot the final clusters from HDBSCAN
> plot(opt_tsne, pch = 19, col = alpha(COLORS[res_opt$cluster +
+   1], 0.4), xlab = "tSNE 1", ylab = "tSNE 2", axes = FALSE)
> axis(1)
> axis(2)
```

Figure 1 (a) shows the resulting scores for each minimum cluster sizes from 3 to 10. Minimum clustering size of 8 achieved the minimum score. Figure 1 (b) shows the resulting clusters using minimum clustering size of 8 for HDBSCAN.

[2] https://sebastiansauer.github.io/figure_sizing_knitr/



(a) Scores

(b) Clusters

Figure 1: (a) Scores for minimum cluster sizes from 3 to 10. Minimum cluster size of 8 (○) achieves the minimum score. (b) Clusters based on the optimal minimum cluster size with Cluster 1 (●), Cluster 2 (●), Cluster 3 (●), and Cluster 4 (●).

Survival analysis

Join data and inspect

```
> # join case_id and clusters
> clusters <- data.frame(
+   case_id = as.character(logrna$case_id),
+   rna_cluster = res_opt$cluster + 1, # start from 1
+   stringsAsFactors = FALSE
+ )
> # 1. join clinical data and clusters
> # 2. remove rows with no vital status
> # 3. code survival event and number of days
> # 4. drop rows with missing number of days
> # 5. fix other data types
> df_final <- clusters %>%
+   inner_join(df %>% mutate(File.Name = as.character(File.Name)),
+              by=c('case_id'='File.Name')) %>%
+   filter(vital_status != 'Not Reported') %>%
+   mutate(
+     outcome = if_else(vital_status == 'Alive', 0 ,1),
+     days = if_else(
+       vital_status == 'Alive',
+       # using as.numeric only will extract the factor order not the value
+       as.numeric(as.character(days_to_last_follow_up)),
+       as.numeric(as.character(days_to_death)))
+   ) %>%
+   drop_na(days) %>%
+   mutate(
+     age=as.numeric(as.character(age_at_index))
+   )
```

Check the `.Rmd` file for configuring tables blocks that generate captions and labels in LaTeX using `knitr::kable`. In text body, the labels are referenced with `Table \ref{tab:<label-in-block>}`.

Table **??** and Figure 2 show the number of cases for gender and tumor stage respectively by cluster. Cases in tumor stages III and beyond are grouped as the number of patients in these later stages are small.

```
> gender_table <- table(df_final$gender, df_final$rna_cluster)
> row.names(gender_table) <- str_to_upper(row.names(gender_table))
> colnames(gender_table) <- sapply(colnames(gender_table),
+   function(x) paste("Cluster", x))
> knitr::kable(gender_table, caption = "Count of patient records by gender and cluster.",
+   digits = 2, booktabs = TRUE, escape = FALSE) %>% kable_styling(latex_options = c("hold_position"))
```

```
> stage_table <- table(df_final$tumor_stage, df_final$rna_cluster)
> row.names(stage_table) <- str_to_upper(row.names(stage_table))
> colnames(stage_table) <- sapply(colnames(stage_table), function(x) paste("Cluster",
+   x))
> par(mar = c(4, 8, 4, 2))
> barplot(t(stage_table[10:2, ]), col = COLORS, border = NA,
+   hori = TRUE, axes = FALSE, las = 2)
> axis(1, las = 0)
> axis(2, tick = FALSE, line = NA, lty = 0, labels = FALSE)
```

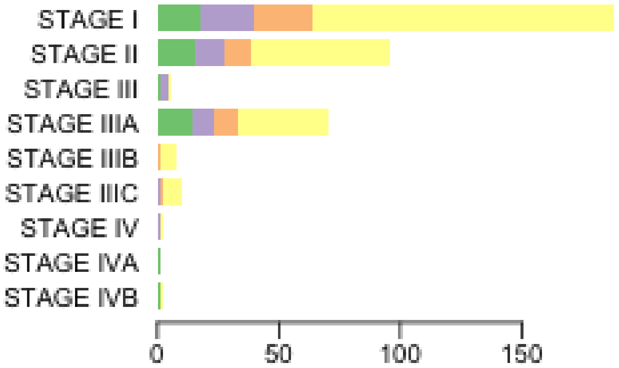


Figure 2: Counts of patients by tumor stages and clusters - Cluster 1 (🟩), Cluster 2 (🟪), Cluster 3 (🟧), and Cluster 4 (🟨). Patients without reported tumor stages (n=32) are not displayed

```
> # group tumor stages
> df_surv <- df_final %>% select(outcome, days, age, rna_cluster,
+   gender, tumor_stage) %>% mutate(tumor_stage = if_else(str_starts(as.character(tumor_stage),
+   "stage i((ii)|(v))"), "stage iii+", as.character(tumor_stage)))
```

Survival rates vs. clusters

The Kaplan-Meier cuves in Figure 3 show different survival rates for the four clusters identified above. In order to test whether the differences are statistically significant, we will use the log-rank test.

```
> library(survival, quietly = TRUE)
> # KM plots
> km_cluster <- survfit(Surv(days, outcome) ~ factor(rna_cluster),
+   data = df_surv)
> plot(km_cluster, col = COLORS, axes = FALSE, xlab = "Day",
+   ylab = "Survival", ylim = c(0, 1))
> grid()
> axis(1)
> axis(2, at = c(0, 0.5, 1))
```

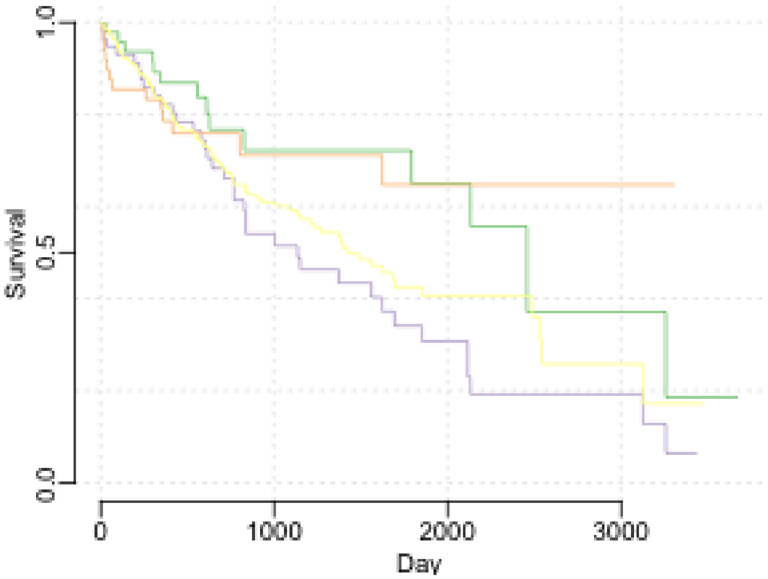


Figure 3: Kaplan-Meier curves for each cluster - Cluster 1 (🟩), Cluster 2 (🟪), Cluster 3 (🟧), and Cluster 4 (🟨).

```
> lrank_cluster <- survdiff(Surv(days, outcome) ~ factor(rna_cluster),
+   data = df_surv, rho = 0  # log-rank test
+ )
> lrank_tab <- data.frame(N = as.vector(lrank_cluster$n),
+   Observed = lrank_cluster$obs, Expected = lrank_cluster$exp,
+   row.names = sapply(sort(unique(df_surv$rna_cluster)),
+     function(x) paste("Cluster", x)))
> # calculate p value
> lrank_pval <- pchisq(lrank_cluster$chisq, df = length(lrank_cluster$n) -
+   1, lower.tail = FALSE)
> knitr::kable(lrank_tab, row.names = TRUE, col.names = c("Number of Patients",
+   "Observed", "Expected"), caption = paste0("Observed and expected numbers of deaths for each cluster
+   round(lrank_cluster$chisq, 2), " with p-value of ",
+   round(lrank_pval, 3), "."), digits = 2, booktabs = TRUE,
+   escape = FALSE) %>% kable_styling(latex_options = c("hold_position"))
```

Table 1 shows the observed and expected numbers of deaths for each cluster.The numbers of observed deaths are smaller than the expected numbers for Cluster 1 and Cluster 3 while Cluster 2 and Cluster 4 display the opposite result. You can also see from Figure 3 that patients in Cluster 1 and Cluster 3 have higher survival rates in general. The p-value of the test is 0.057 indicating that the differences in the survival rates between clusters aren't significant at 5% significance level. Although we fail to reject the null hypothesis that different clusters share a common survival curve at 5% significance level, we note the p-value is quite close to 5%.

```
> cox_cluster <- coxph(Surv(days, outcome) ~ factor(rna_cluster),
+   data = df_surv)
```

Table 1: Observed and expected numbers of deaths for each cluster if their survival curves were identical. From the log-rank test, the chisquare statistic is 7.53 with p-value of 0.057.

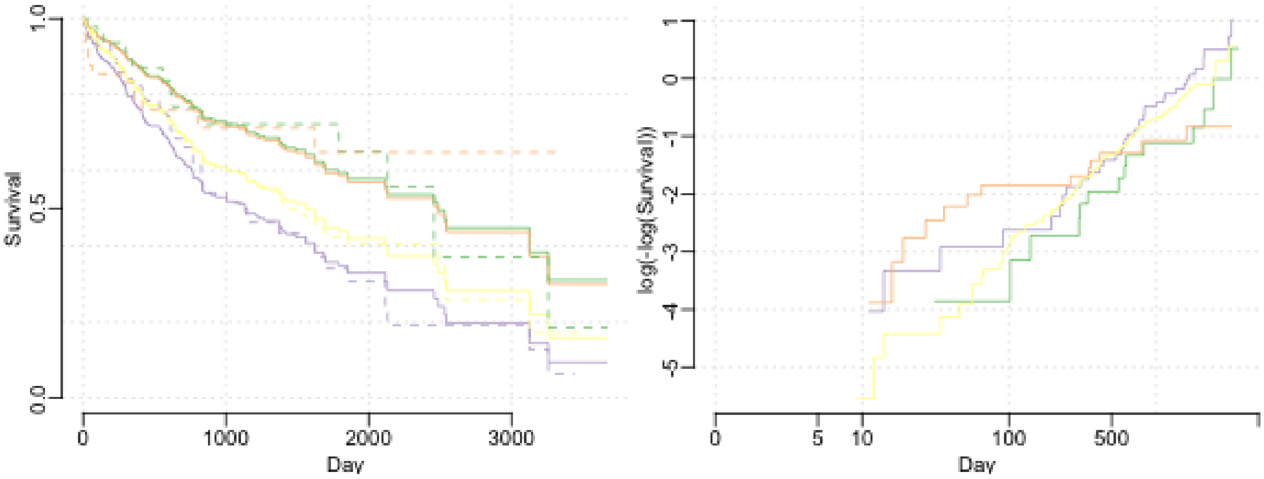
	Number of Patients	Observed	Expected
Cluster 1	51	14	21.12
Cluster 2	57	36	27.05
Cluster 3	49	13	19.00
Cluster 4	259	101	96.83

```
> ci_cluster <- summary(cox_cluster)$conf.int
> row.names(ci_cluster) <- sapply(sort(unique(df_surv$rna_cluster)),
+   function(x) paste("Cluster", x))[-1]
> knitr::kable(ci_cluster[, -2], row.names = TRUE, col.names = c("Hazard Ratio",
+   "Lower CI", "Upper CI"), caption = "Estimated hazard ratios of clusters with respect to Cluster 1 ar
+   digits = 2, booktabs = TRUE, escape = FALSE) %>% kable_styling(latex_options = c("hold_position"))
```

Table 2: Estimated hazard ratios of clusters with respect to Cluster 1 and their 95% confidence bounds.

	Hazard Ratio	Lower CI	Upper CI
Cluster 2	2.02	1.09	3.75
Cluster 3	1.03	0.48	2.19
Cluster 4	1.58	0.90	2.77

We can also fit a proportional hazard model and check whether the clusters have significant effect on the hazard function. Table 2 shows the estimated hazard ratios for each cluster with respect to Cluster 1. The confidence intervals indicate that Cluster 2 has a significant effect at 5% significance level. Patients in Cluster 2 have twice of the risk compared to those in Cluster 1.



(a) Fitted (▬) vs. observed (╌).

(b) Log-log plot.

Figure 4: Diagnosis plots for the proportional hazard model with clusters - Cluster 1 (🟩), Cluster 2 (🟪), Cluster 3 (🟧), and Cluster 4 (🟨).

The proportional hazard model assumes hazard ratios are constant over time. In order to diagnose the model assumptions, we can use various graphical approaches. In Figure 4 (a), the fitted curve for Cluster 3 is inconsistent with the observed cuve while other fitted curves follow the observed curves in general. Parallel lines in a log-log plot would suggest the proportional hazard assumption. We can observe the Cluster 3 curve in Figure 4 (b) moving in a different direction while the other three lines are parallel.

Effect of covariates

In order to analyse whether different tumor stages change the effect of the clusters on survival, we will fit another Cox model with tumor size variable added. (I won't repeat it for gender.) The estimated hazard ratios and thier 95% confidence intervals are shown in Figure 5. The estimates for interaction terms aren't significant.

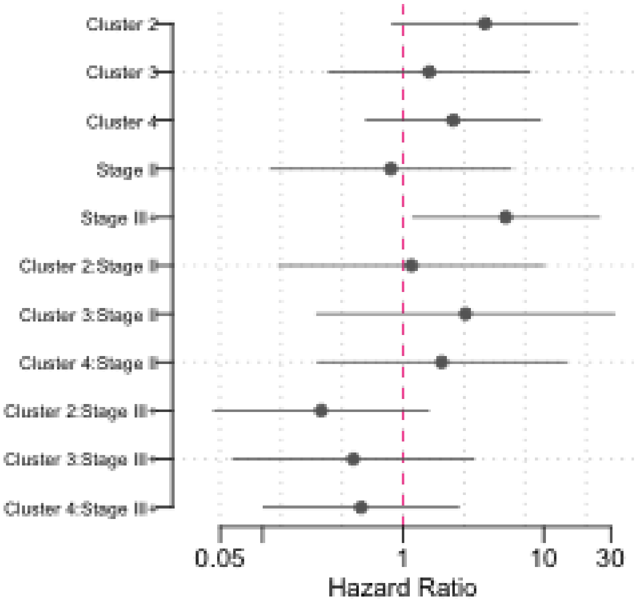


Figure 5: Estimated hazard ratios (●) and 95% confidence intervals (▬) with respect to Cluster 1 in Stage I.