HW7 - Unsupervised Learning

Adarsh Mathew

3/13/2020

k-Means Clustering by Hand

```
eu_dist <- function(x1, x2, c1, c2){
  out <- sqrt((x1-c1)^2 + (x2-c2)^2)
  return(out)
}

assign_cluster3 <- function(x1, y1, c_clus_mat){
  d_1 <- eu_dist(x1, y1, c_clus_mat["c_1", 1][[1]], c_clus_mat["c_2", 1][[1]])
  d_2 <- eu_dist(x1, y1, c_clus_mat["c_1", 2][[1]], c_clus_mat["c_2", 2][[1]])
  d_3 <- eu_dist(x1, y1, c_clus_mat["c_1", 3][[1]], c_clus_mat["c_2", 3][[1]])

  return(which.min(c(d_1, d_2, d_3)))
}

assign_cluster2 <- function(x1, y1, c_clus_mat){
  d_1 <- eu_dist(x1, y1, c_clus_mat["c_1", 1][[1]], c_clus_mat["c_2", 1][[1]])
  d_2 <- eu_dist(x1, y1, c_clus_mat["c_1", 2][[1]], c_clus_mat["c_2", 2][[1]])

  return(which.min(c(d_1, d_2)))
  }
```

```
kmeans_mini_df <- tibble(input_1 = c(5,8,7,8,3,4,2,3,4,5),
                         input_2 = c(8,6,5,4,3,2,2,8,9,8),
                         cluster_val3 = sample.int(3, 10, replace = TRUE),
                         cluster_val2 = sample.int(2, 10, replace = TRUE))

i = 1

while (i <= 50) {
  clust3_centroids <- kmeans_mini_df %>% group_by(cluster_val3) %>%
    summarize(c_1 = mean(input_1), c_2 = mean(input_2)) %>% t()

  clust2_centroids <- kmeans_mini_df %>% group_by(cluster_val2) %>%
    summarize(c_1 = mean(input_1), c_2 = mean(input_2)) %>% t()

  kmeans_mini_df <- kmeans_mini_df %>% ungroup() %>%
    mutate(cluster_val3 = map2_dbl(input_1, input_2, ~assign_cluster3(.x, .y, clust3_centroids)),
           cluster_val2 = map2_dbl(input_1, input_2, ~assign_cluster2(.x, .y, clust2_centroids)))

  i = i+1
}
```

```
clust3_plot <- kmeans_mini_df %>%
  ggplot() +
  geom_point(aes(x = input_1, y = input_2, colour = as.factor(cluster_val3))) +
  theme_minimal() + ggtitle("k=3") + theme(legend.position = "bottom")

clust2_plot <- kmeans_mini_df %>%
  ggplot() +
  geom_point(aes(x = input_1, y = input_2, colour = as.factor(cluster_val2))) +
  theme_minimal() + ggtitle("k=2") + theme(legend.position = "bottom")

(clust3_plot + clust2_plot) +
  plot_annotation(title = "Final Cluster Assignments after 50 iterations")
```

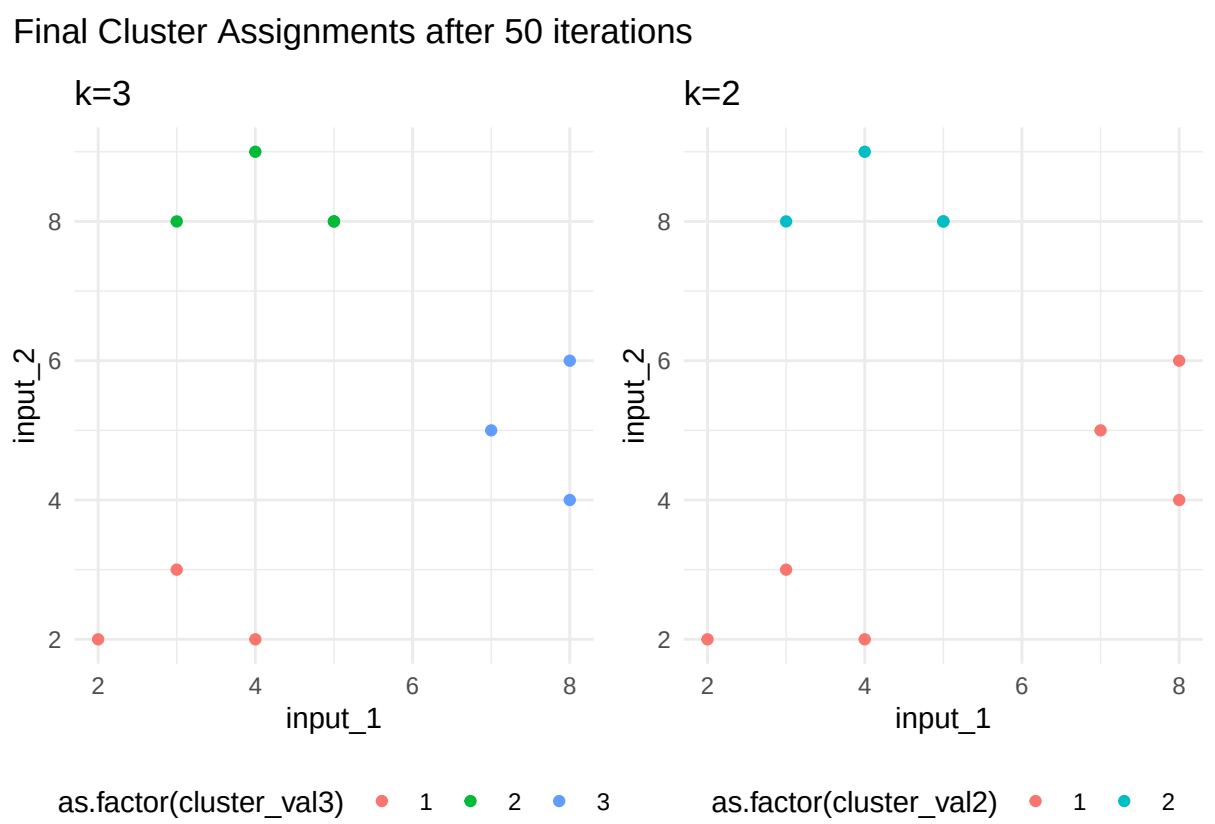


Figure 1: Final Cluster Assignments after 50 iterations (k = 2, 3)

```
clust3_centroids %>% t() %>% as_tibble() %>% select(-cluster_val3) %>% dist()
```

```
##          1        2
## 2 6.047268
## 3 5.374838 4.715518
```

$k = 3$ is a much better choice – there are three distinct groups of points in the plot.

By setting $k = 2$ in the second plot, we force the first cluster (red in $k = 3$) to join the cluster centroid it is closest to – which is cluster 3 (blue).

Application Exercises

Dimension Reduction

```
wiki_scaled <- read_csv("../data/wiki.csv") %>%
  mutate_if(is.numeric, scale)
```

```
pca_wiki <- princomp(wiki_scaled)

pc_plot <- ggbiplot(pca_wiki, alpha = 0.15, varname.abbrev = FALSE) + xlim(-3, 3) + ylim(-3, 3)
pc_plot + theme_minimal() +
  ggtitle("Bi-plot of Variables against top-2 Principal Components",
          "29.2% of variance explained")
```

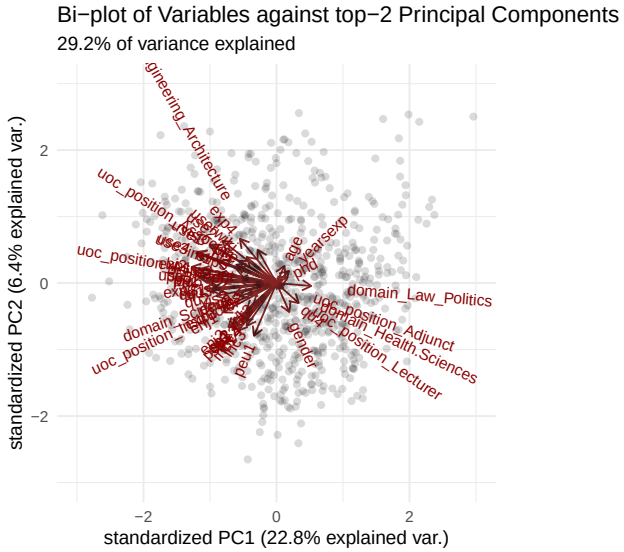


Figure 2: Bi-plot of Variables against top-2 Principal Components

```
detach("package:ggbiplot", unload = TRUE)
```

```
wiki_pcloading <- pca_wiki$loadings[ ,1:ncol(wiki_scaled)] %>% as_tibble(rownames = "vbl_name")
```

```
wiki_pcloading %>% select(1:2) %>%
  arrange(desc(abs(Comp.1))) %>%
  filter(row_number() <= 10)
```

```
## # A tibble: 10 x 2
##    vbl_name Comp.1
##    <chr>     <dbl>
##  1 bi2      -0.231
##  2 bi1      -0.226
##  3 use3     -0.219
##  4 use4     -0.215
##  5 pu3      -0.211
##  6 exp1     -0.209
##  7 use5     -0.207
##  8 exp2     -0.195
##  9 pu1      -0.193
```

```
## 10 pu2      -0.191
```

```
wiki_pcloading %>% select(1, 3) %>%
  arrange(desc(abs(Comp.2))) %>%
  filter(row_number() <= 10)
```

```
## # A tibble: 10 x 2
##    vbl_name Comp.2
##    <chr>     <dbl>
##  1 peu1     -0.272
##  2 inc1     -0.245
##  3 sa3      -0.242
##  4 sa1      -0.230
##  5 exp4      0.228
##  6 enj2     -0.228
##  7 sa2      -0.227
##  8 peu2     -0.222
##  9 inc3     -0.221
## 10 use2      0.219
```

```
pca_wiki_var <- pca_wiki$sdev %>% enframe(name = "Comp", value = "sdev") %>%
  mutate(comp_var = map_dbl(sdev, ~.x^2),
         prop_comp_var = comp_var/sum(comp_var),
         Comp = fct_reorder(Comp, prop_comp_var, .desc = TRUE))

pve_plot <- pca_wiki_var %>%
  ggplot() +
  geom_point(aes(x = Comp, y = prop_comp_var, colour = "PVE"), colour = "blue") +
  theme_minimal() +
  ggtitle("Proportion of Variance")

cve_plot <- pca_wiki_var %>%
  ggplot() +
  geom_point(aes(x = Comp, y = cumsum(prop_comp_var), colour = "CVE"), colour = 'red') +
  theme_minimal() +
  ggtitle("Cumulative Proportion of Variance")

pve_plot / cve_plot
```

```
pca_wiki_var %>% filter(row_number() <=2 ) %>% select(prop_comp_var) %>% sum()
```

```
## [1] 0.291831
```

$PC1$ and $PC2$ explain 29.18 of the cumulative variance.

```
wiki_tsne5 <- Rtsne(wiki_scaled %>% as.matrix(), perplexity = 5)

wiki_tsne5_plot <- wiki_scaled %>%
  mutate(tsne1 = wiki_tsne5$Y[,1],
         tsne2 = wiki_tsne5$Y[,2]) %>%
  ggplot() +
  geom_point(aes(x = tsne1, y = tsne2), alpha = 0.3) +
  labs(x = "First dimension",
       y = "Second dimension") +
  ggtitle('Perplexity = 5') +
  theme_minimal()
```

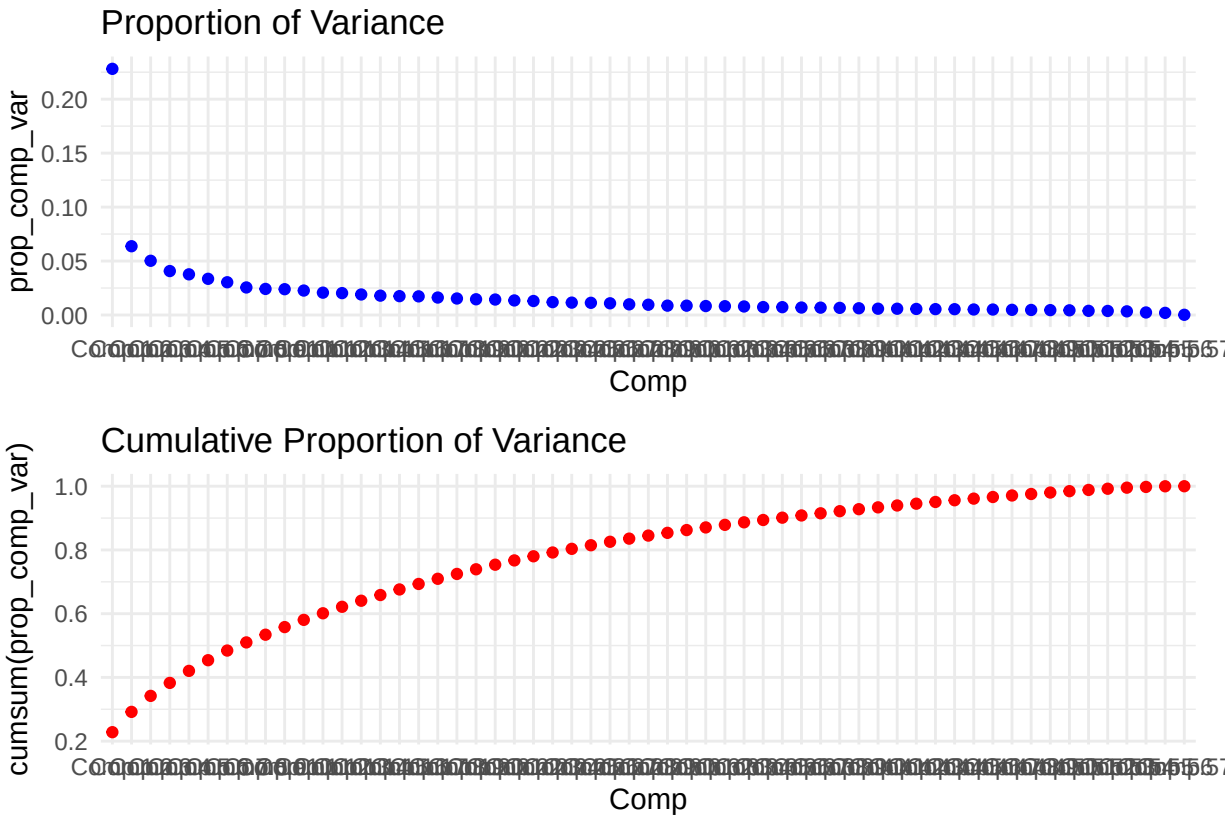


Figure 3: Individual and Cumulative Proportion of Variance explained by top-2 Principal Components

```r
wiki_tsne50 <- Rtsne(wiki_scaled %>% as.matrix(), perplexity = 50)

wiki_tsne50_plot <- wiki_scaled %>%
  mutate(tsne1 = wiki_tsne50$Y[,1],
         tsne2 = wiki_tsne50$Y[,2]) %>%
  ggplot() +
  geom_point(aes(x = tsne1, y = tsne2), alpha = 0.3) +
  labs(x = "First dimension",
       y = "Second dimension") +
  ggtitle('Perplexity = 50') +
  theme_minimal()

wiki_tsne1 <- Rtsne(wiki_scaled %>% as.matrix(), perplexity = 1)

wiki_tsne1_plot <- wiki_scaled %>%
  mutate(tsne1 = wiki_tsne1$Y[,1],
         tsne2 = wiki_tsne1$Y[,2]) %>%
  ggplot() +
  geom_point(aes(x = tsne1, y = tsne2), alpha = 0.3) +
  labs(x = "First dimension",
       y = "Second dimension") +
  ggtitle('Perplexity = 1') +
  theme_minimal()

wiki_tsne10 <- Rtsne(wiki_scaled %>% as.matrix(), perplexity = 10)

wiki_tsne10_plot <- wiki_scaled %>%
  mutate(tsne1 = wiki_tsne10$Y[,1],
         tsne2 = wiki_tsne10$Y[,2]) %>%
  ggplot() +
  geom_point(aes(x = tsne1, y = tsne2), alpha = 0.3) +
  labs(x = "First dimension",
       y = "Second dimension") +
  ggtitle('Perplexity = 10') +
  theme_minimal()


(wiki_tsne5_plot + wiki_tsne50_plot) / (wiki_tsne1_plot + wiki_tsne10_plot) +
  plot_annotation("t-SNE Plots")
```

- `perplexity = 5`: A few isolated clusters at 0°, 60°, 90°, 180°, −115°. The low-value optimizes for local-effects.

- `perplexity = 50`: Two large isolated clusters at ±30°, ±150°. We see a large central blob, as the high value tries to seek out global effects.

- `perplexity = 1`: The dataset is extremely speciated given the low value of perplexity. Pointless.
- `perplexity = 10`: Seems like a good balance between speciation and global behaviour. We see several relatively well-defined clusters all over. This would be my pick.

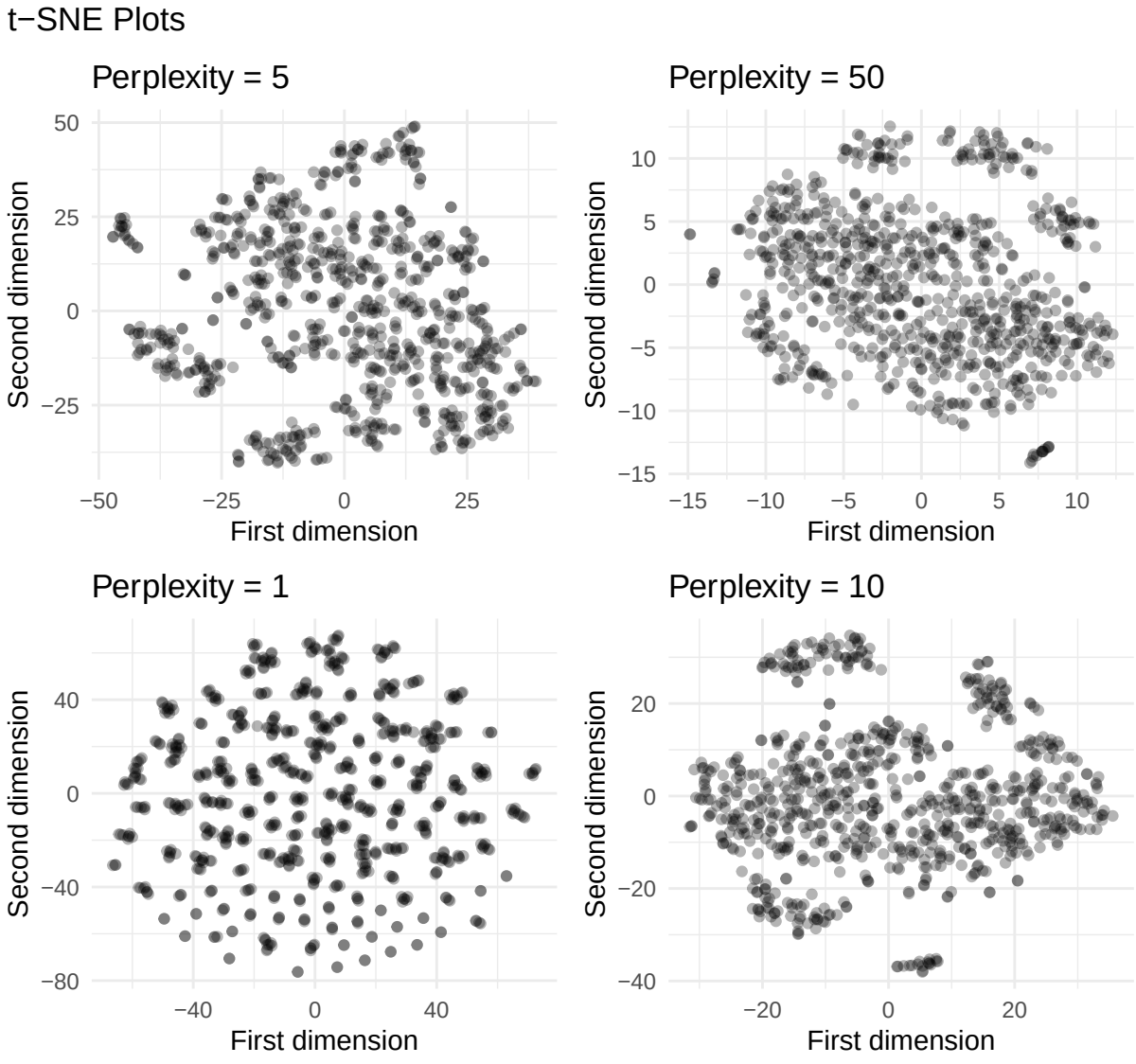


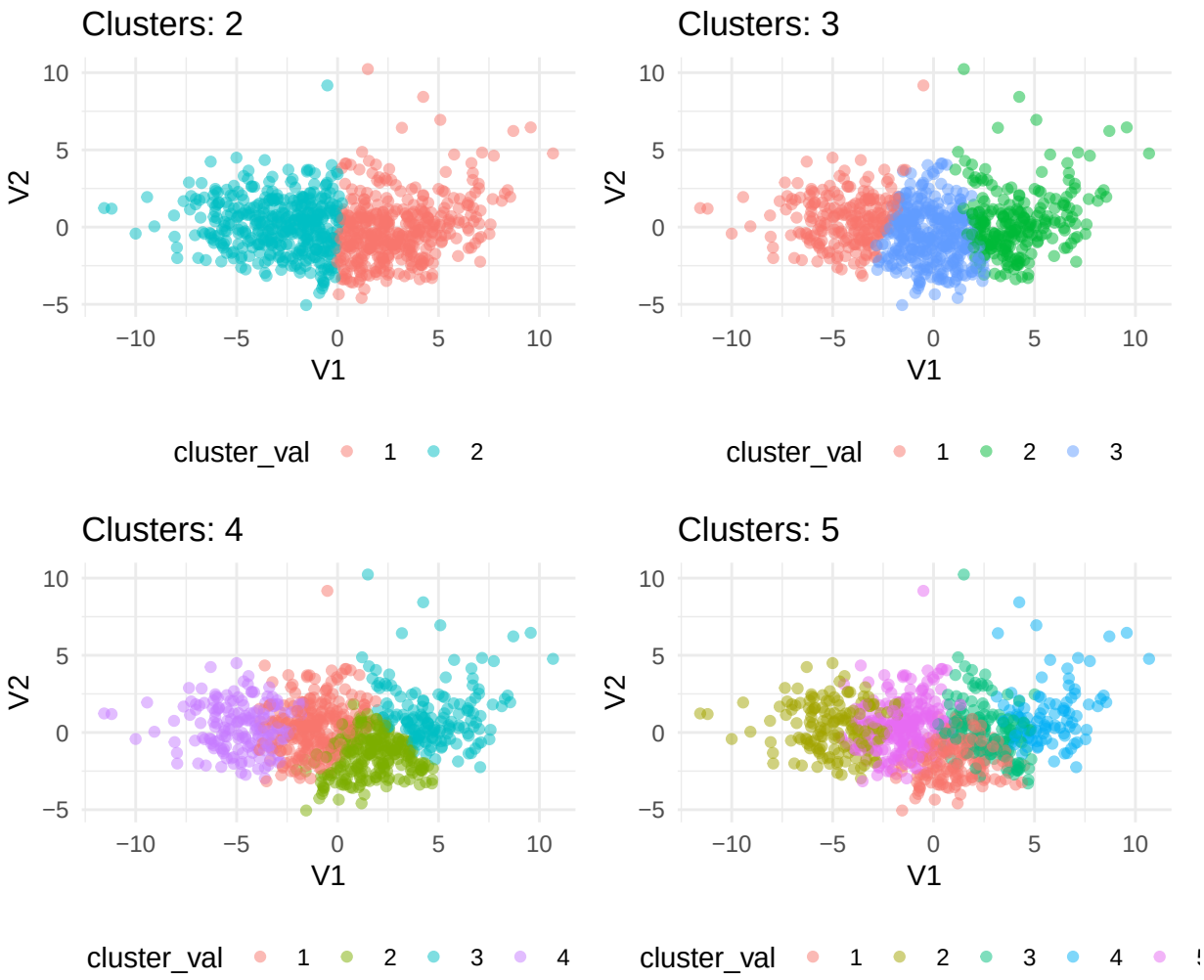
Figure 4: t-SNE plots of Wikipedia data with varying perplexity

Clustering

```
plot_gen <- function(n_clusters, rotation_type){

  if (rotation_type == "pca") {
    out_df <- pca_wiki$scores %>% as_tibble() %>%
      select(Comp.1, Comp.2) %>%
      dplyr::rename(V1 = Comp.1, V2 = Comp.2)
  } else {
    out_df <-wiki_tsne10$Y %>% as_tibble()
    }


  kmeans_model <- kmeans(wiki_scaled, centers = n_clusters, iter.max = 100)

  plot_data <- wiki_scaled %>%
    bind_cols(cluster_val = as.factor(kmeans_model$cluster),
              out_df)

  plot_out <- plot_data %>%
    ggplot() +
    geom_point(aes(x = V1, y = V2, colour = cluster_val), alpha = 0.5) +
    theme_minimal() +
    ggtitle(paste("Clusters: ", as.character(n_clusters), sep = ''))

  return(plot_out)
}
```

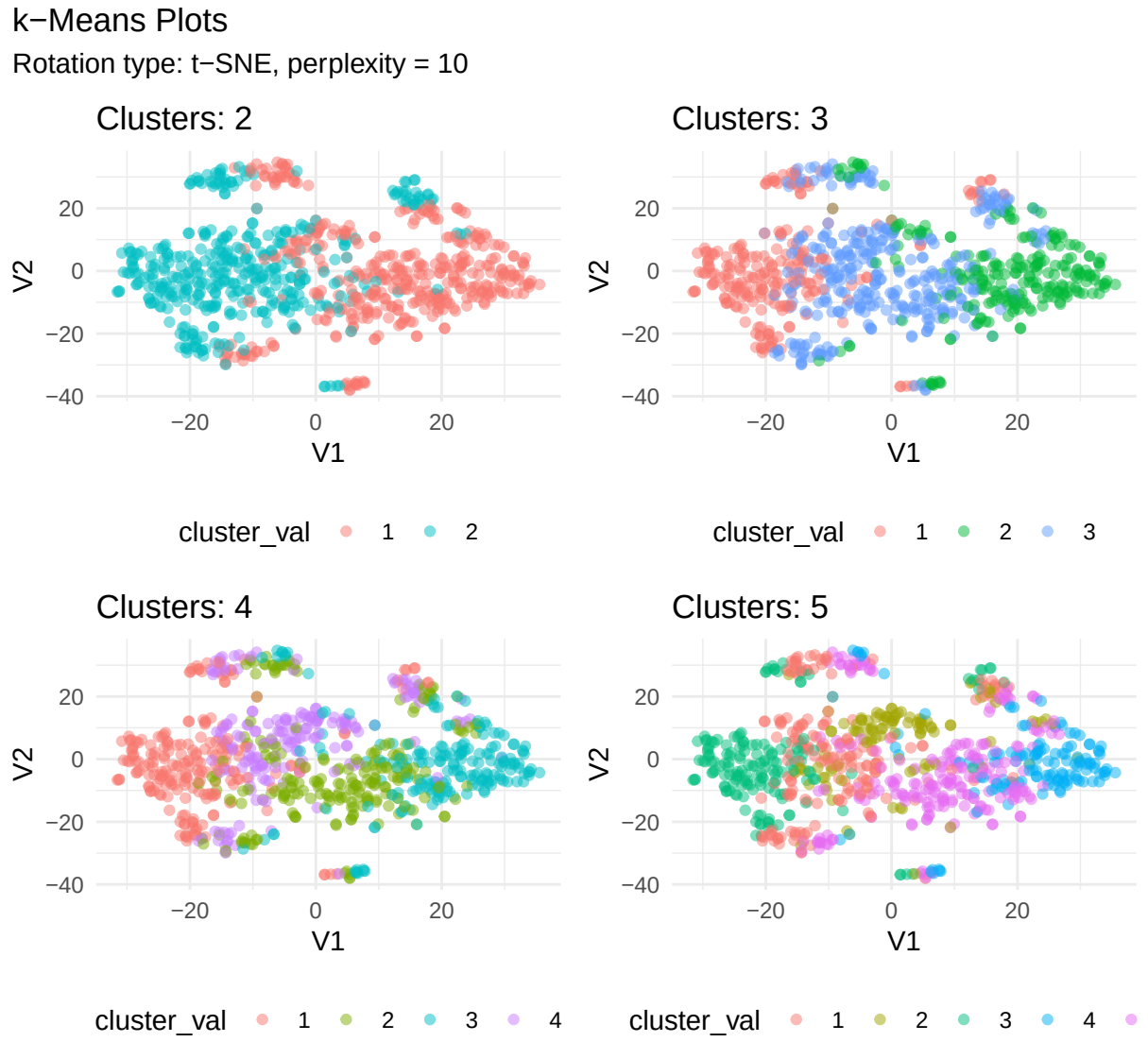
```
kmeans_df <-  tibble(n_clusters = 2:5) %>%
  mutate(pca_plot_obj = map(n_clusters, ~plot_gen(.x, "pca")),
         tsne_plot_obj = map(n_clusters, ~plot_gen(.x, "t-SNE")))
```

```
(kmeans_df$pca_plot_obj[[1]] + kmeans_df$pca_plot_obj[[2]]) / (kmeans_df$pca_plot_obj[[3]] + kmeans_df$
  plot_annotation(title = 'k-Means Plots', subtitle = "Rotation type: PCA") & theme(legend.position = "
```



We notice a clean partition boundary for $k = 2$. We also see good boundaries for $k = 4, 5$, but there is some spatial overlap in these cases. These visual interpretations are obviously a function of the PCA loadings and are not a substantitve substitute for validation statistics.

```
(kmeans_df$tsne_plot_obj[[1]] + kmeans_df$tsne_plot_obj[[2]]) / (kmeans_df$tsne_plot_obj[[3]] + kmeans_
  plot_annotation(title = 'k-Means Plots', subtitle = "Rotation type: t-SNE, perplexity = 10") & theme(
```



The clean boundaries for $k = 2$ persist with the `t-SNE` rotations. We see greater noise creep in for $k = 4, 5$, replicating what we saw in the `PCA` visualizations. The `perplexity` value we've chosen focuses on local behaviour, and yet we're able to distinguish global trends, albeit with some noise.

```
elbow_plot <- fviz_nbclust(wiki_scaled, FUNcluster = kmeans, method = "wss") +
  ggtitle('Elbow plot')

silh_plot <- fviz_nbclust(wiki_scaled, FUNcluster = kmeans, method = "silhouette") +
  ggtitle('Silhouette plot')

gap_plot <- fviz_nbclust(wiki_scaled, FUNcluster = kmeans, method = "gap_stat") +
  ggtitle('Gap-statistic plot')

(elbow_plot / silh_plot / gap_plot) +
  plot_annotation("Cluster Evaluation plots for k-Means")
```

The optimal number of clusters is 2. We see the sharpest fall in WSS for that value, backed up by the other two metrics.

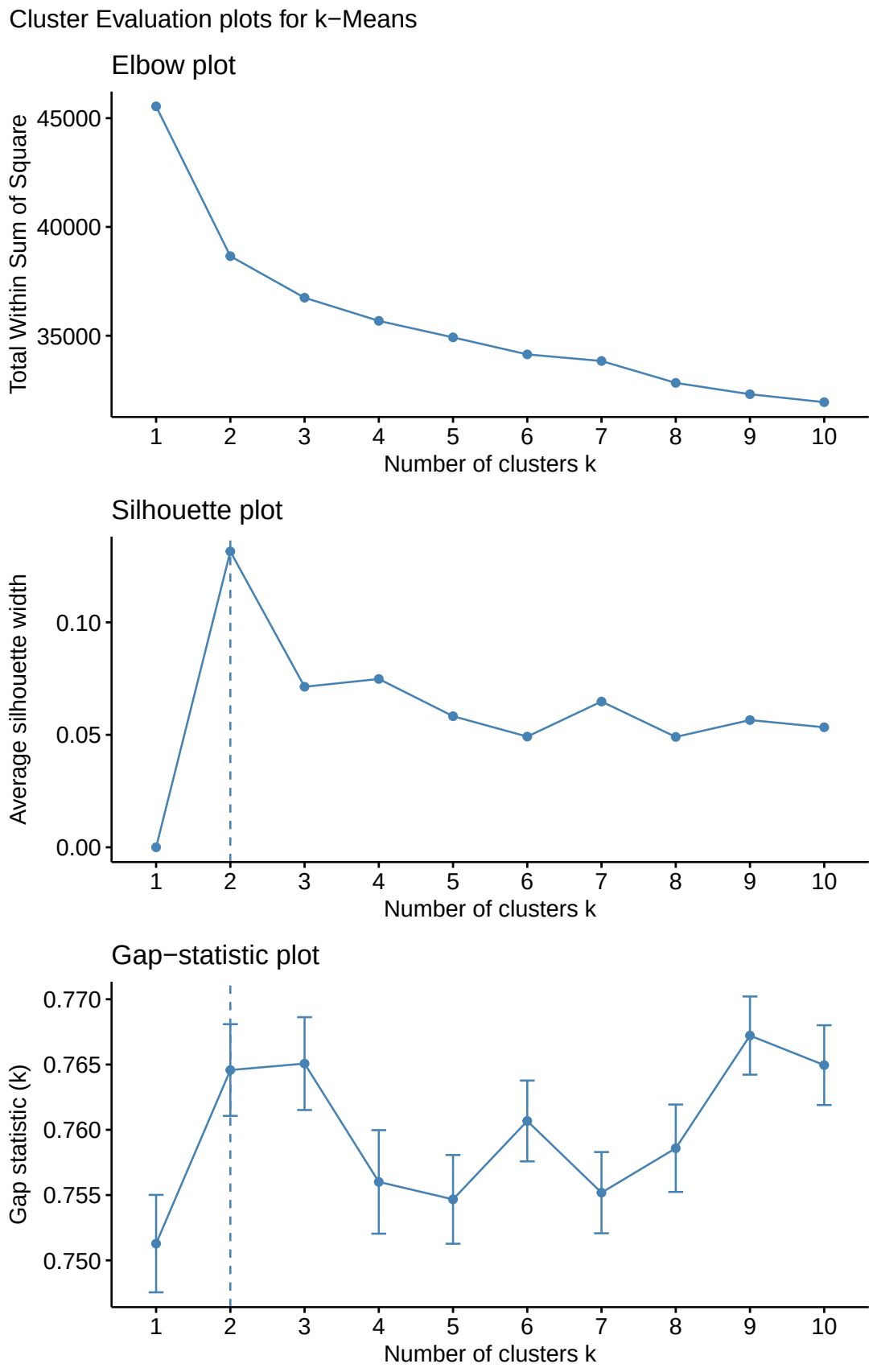


Figure 5: Cluster Validation for k-Means