Working Segmentation Analysis

Matthew

2026-01-21

Overview

Our initial goal for this project was to get acquainted with segmentation techniques (mainly K-means clustering). Our results below show that we are able to cluster foods from The Canadian Nutrient File into rankings ordered by their macro-nutrients (Protein, Carbohydrates, Fatty Acids).

Given our results, this project can grow to incorporate external data sources.

Currently this all that is completed.

Our next steps are to link this data to the CPI. possibly price macro-nutrients.

Cleaning data

In the given documentation we have a file relation chart shown here

Merging all of our datasets into one master file. would give us a hard time. so were going to follow the charts heirchery and merge our data acording to the data they contain.

First we link the CNF "Food Group" data to the CNF "Food Name" Data

```
food <- food_name
food <- merge(food, food_group, by = c("FoodGroupID"))
food <- food[ , !grepl("F$", names(food))]
head(food)
```

Next we merge this data set, with four other datasets seperately.

These will sets be called, Yield, Nutrients, Refuse, Conversion.

It's also important to mention that this is a bilingual data set.

So within this step (considering i don't speak french) I will also be removing french columns.

```
#Nutrients
nutrients <- merge(nutrient_name, nutrient_amount, by = c("NutrientID"))
nutrients <- merge(nutrients, food, by = c("FoodID"))
nutrients <- nutrients[ , !grepl("F$", names(nutrients))]
head(nutrients)

nutrient_wide <- nutrients %>%
  dplyr::select(FoodID, NutrientName, NutrientValue) %>%
  pivot_wider(names_from = NutrientName, values_from = NutrientValue) %>%
```

```
  left_join(food, by = "FoodID")  # to bring in food names/groups

# Optional: remove NA or replace with 0
# nutrient_wide[is.na(nutrient_wide)] <- 0

#Yield
yield <- merge(yield_name, yield_amount, by = c("YieldID"))
yield <- merge(yield, food, by = c("FoodID"))
yield <- yield[ , !grepl("F$", names(yield))]
head(yield)


#Refuse
refuse <- merge(refuse_name, refuse_amount, by = c("RefuseID"))
refuse <- merge(refuse, food, by = c("FoodID"))
refuse <- refuse[ , !grepl("F$", names(refuse))]
head(refuse)


#Conversion
conversion <- merge(measure_name, conversion_factor, by = c("MeasureID"))
conversion <- merge(nutrients, food, by = c("FoodID"))
conversion <- conversion[ , !grepl("F$", names(conversion))]
head(conversion)
```

This step here just removes our raw datasets (as we will not be using them). This cleans up our local environment.

```
df_names <- c("food_group","food_name","food_source","measure_name","conversion_factor","nutrient_name"
rm(list = df_names)

#Ignore
# cnf <- cnf %>%
#   dplyr::select(-MeasureID, -RefuseID, -YieldID, -FoodGroupID, -NutrientID, -FoodCode, -FoodSourceID,
```

Preparing for K-Means Clustering.

From our nutrition dataset we can look at our food entries in terms of their "nutrition profile." by grouping like-foods with one-another. foods similar in nutrition will be grouped in the same group. visa versa for those not.

The following are the selected nutrients we've selected for.

They've been selected for a number of reasons.

1. completeness within the dataset. see here for the completeness of each variable. we ommitted variables when they were found in less than 80% of the data.
2. As the dimensionality in our matrix increases, the usefulness of our results decrease. this is an inherent the geometry behind high-dimensional spheres. The likelihood of clusters forming in nice, evenly-separated spheres is basically zero, not to mention the fact that the volumes of any such sphere becomes increasingly large and hard to deal with.

3. For now let's keep our clustering simple. we'll only measure the three macro-nutrients. (protien, fat, carbs)

```
nutrient_vector <- c("FoodID",
                     "PROTEIN",
                     "FAT (TOTAL LIPIDS)",
                     "CARBOHYDRATE, TOTAL (BY DIFFERENCE)"
                     # "ENERGY (KILOCALORIES)",
                     #"MOISTURE",
                     # "ASH, TOTAL",
                     # "SODIUM",
                     # "CALCIUM",
                     # "IRON"
                     # "FATTY ACIDS, POLYUNSATURATED, 22:6 n-3, DOCOSAHEXAENOIC (DHA)",
                     # "FATTY ACIDS, POLYUNSATURATED, 22:5 n-3, DOCOSAPENTAENOIC (DPA)",
                     # "PHOSPHORUS",
                     # "FOLIC ACID",
                     # "POTASSIUM",
                     #"VITAMIN C",
                     # "CHOLESTEROL",
                     #"MAGNESIUM",
                     #"ZINC",
                     #"FATTY ACIDS, SATURATED, TOTAL",
                     # "FIBRE, TOTAL DIETARY",
                     # "NIACIN (NICOTINIC ACID) PREFORMED",
                     # "TOTAL NIACIN EQUIVALENT",
                     # "RETINOL ACTIVITY EQUIVALENTS",
                     # "RIBOFLAVIN",
                     # "COPPER",
                     # "THIAMIN",
                     # "FATTY ACIDS, POLYUNSATURATED, 18:3 c,c,c n-6, g-LINOLENIC, OCTADECATRIENOIC",
                     # "CAFFEINE",
                     # "FATTY ACIDS, MONOUNSATURATED, TOTAL","FATTY ACIDS, POLYUNSATURATED, TOTAL",
                     # "ALCOHOL",
                     # "THEOBROMINE",
                     #"VITAMIN B-12",
                     #"VITAMIN B-6",
                     # "TOTAL FOLACIN",
                     # "DIETARY FOLATE EQUIVALENTS",
                     # "RETINOL",
                     # "NATURALLY OCCURRING FOLATE",
                     # "FATTY ACIDS, POLYUNSATURATED, 20:3 n-3 EICOSATRIENOIC",
                     # "FATTY ACIDS, POLYUNSATURATED, 20:3 n-6, EICOSATRIENOIC",
                     # "FATTY ACIDS, POLYUNSATURATED, 18:2undifferentiated, LINOLEIC, OCTADECADIENOIC",
                     # "FATTY ACIDS, MONOUNSATURATED, 18:1undifferentiated, OCTADECENOIC","MANGANESE",
                     # "FATTY ACIDS, SATURATED, 16:0, HEXADECANOIC","FATTY ACIDS, SATURATED, 18:0, OCTA
                     # "FATTY ACIDS, TRANS, TOTAL",
                     #"BETA CAROTENE",
                     # "FATTY ACIDS, POLYUNSATURATED, 18:3undifferentiated, LINOLENIC, OCTADECATRIENOIC
                     #"VITAMIN D (INTERNATIONAL UNITS)",
                     # "SELENIUM",
                     # "FATTY ACIDS, SATURATED, 14:0, TETRADECANOIC","FATTY ACIDS, MONOUNSATURATED, 16:
                     # "PANTOTHENIC ACID","FATTY ACIDS, POLYUNSATURATED, 18:3 c,c,c n-3  LINOLENIC, OCT
)
```

```r
#Pivot nutrients so each food is a row, each nutrient a column
nutrient_wide_clean <- nutrient_wide %>%
  dplyr::select(nutrient_vector)

# Scale for clustering
nutrient_matrix <- nutrient_wide_clean %>%
  dplyr::select(where(is.numeric)) %>%
  dplyr::select(-FoodID) %>%
  scale()

colnames(nutrient_matrix)
```

```
## [1] "PROTEIN"                              "FAT (TOTAL LIPIDS)"
## [3] "CARBOHYDRATE, TOTAL (BY DIFFERENCE)"
```

```r
anyNA(nutrient_matrix) # TRUE means there's at least one NA
```

```
## [1] FALSE
```

```r
any(is.infinite(nutrient_matrix))  # TRUE means there's Inf or -Inf
```

```
## [1] FALSE
```

```r
nutrient_matrix_clean <- nutrient_matrix[complete.cases(nutrient_matrix), ]

dim(nutrient_matrix)         # Original size
```

```
## [1] 5690    3
```

```r
dim(nutrient_matrix_clean)   # After removing NAs
```

```
## [1] 5690    3
```

```r
# This allows us to preserve which rows we removed from our data set. this will come in handy later whe
row_mask <- complete.cases(nutrient_matrix)
nutrient_wide_final <- nutrient_wide_clean[row_mask, ]
```

Our Naive Clustering.

Here we run our 3 clusters as a naive test to see what happens.

```r
# K-means clustering
set.seed(5)
k_clusters <- kmeans(nutrient_matrix_clean, centers = 3)

# Add cluster back to data
nutrient_wide_final$Cluster <- factor(k_clusters$cluster)
```

```r
# PCA visualization
pca <- prcomp(nutrient_matrix_clean)
pca_df <- as.data.frame(prcomp(nutrient_matrix_clean)$x)
pca_df$Cluster <- nutrient_wide_final$Cluster
```

This graph can be simply interpreted as.

Each point = one food.

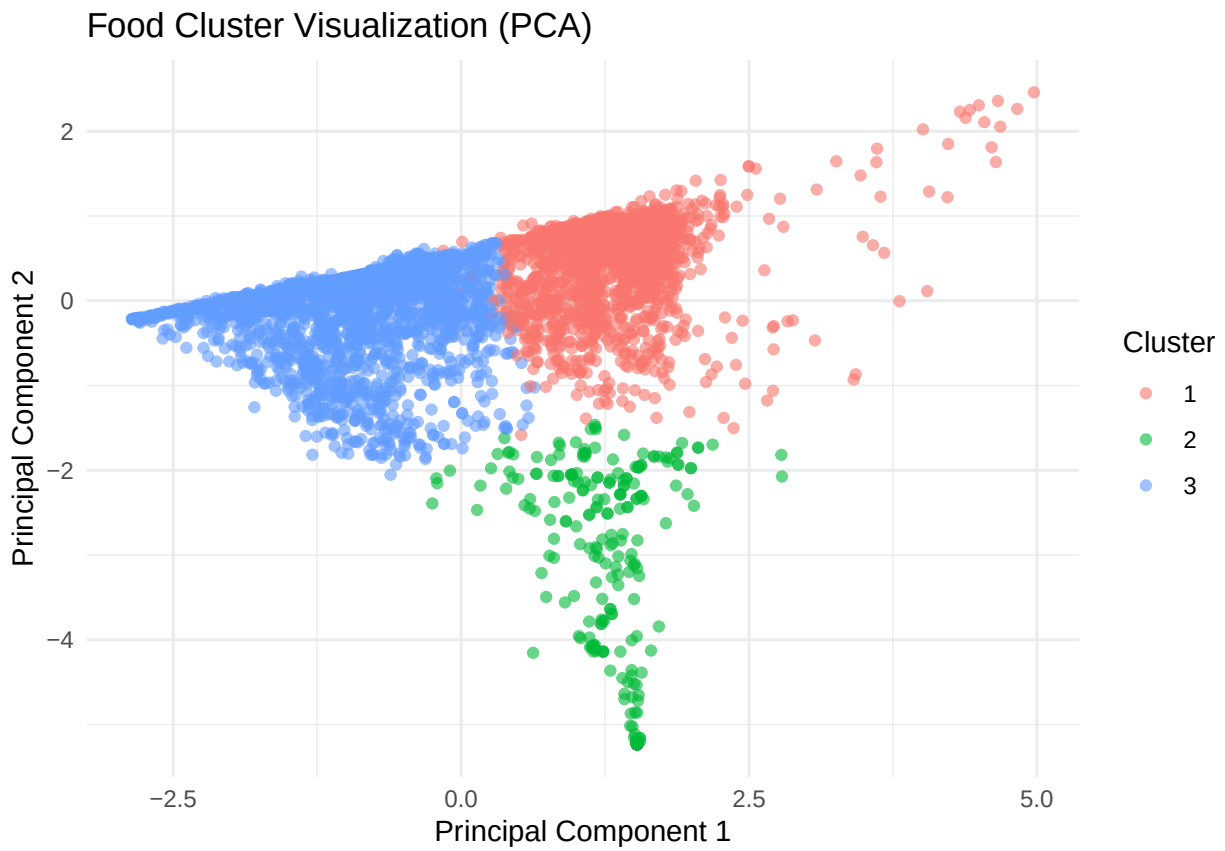
Foods close together = similar nutrient profiles.

Foods far apart = nutritionally different.

PC1 (x-axis) explains the most variance; PC2 the second most.

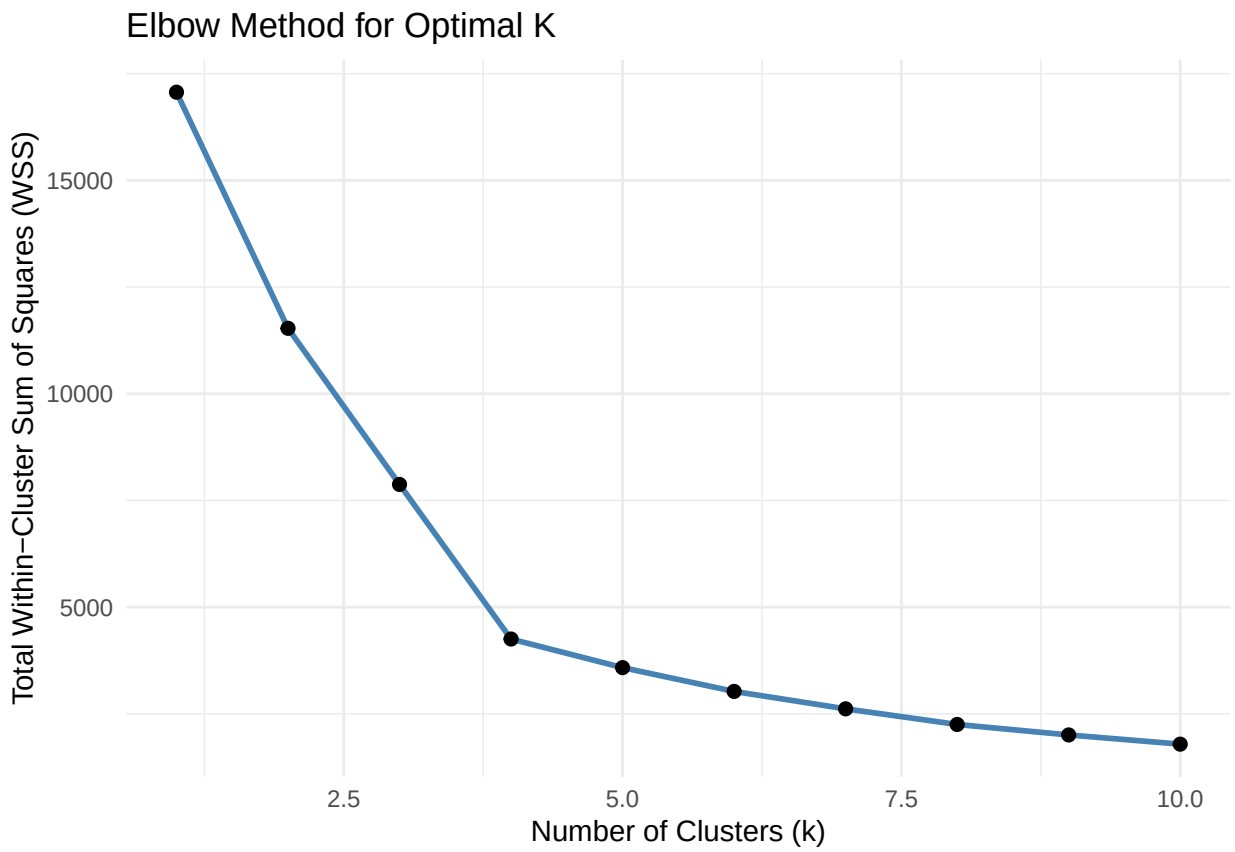
The direction and spread of clusters show how different combinations of nutrients separate the foods.

```r
ggplot(pca_df, aes(PC1, PC2, color = Cluster)) +
  geom_point(alpha = 0.6) +
  theme_minimal() +
  labs(title = "Food Cluster Visualization (PCA)",
       x = "Principal Component 1",
       y = "Principal Component 2")
```



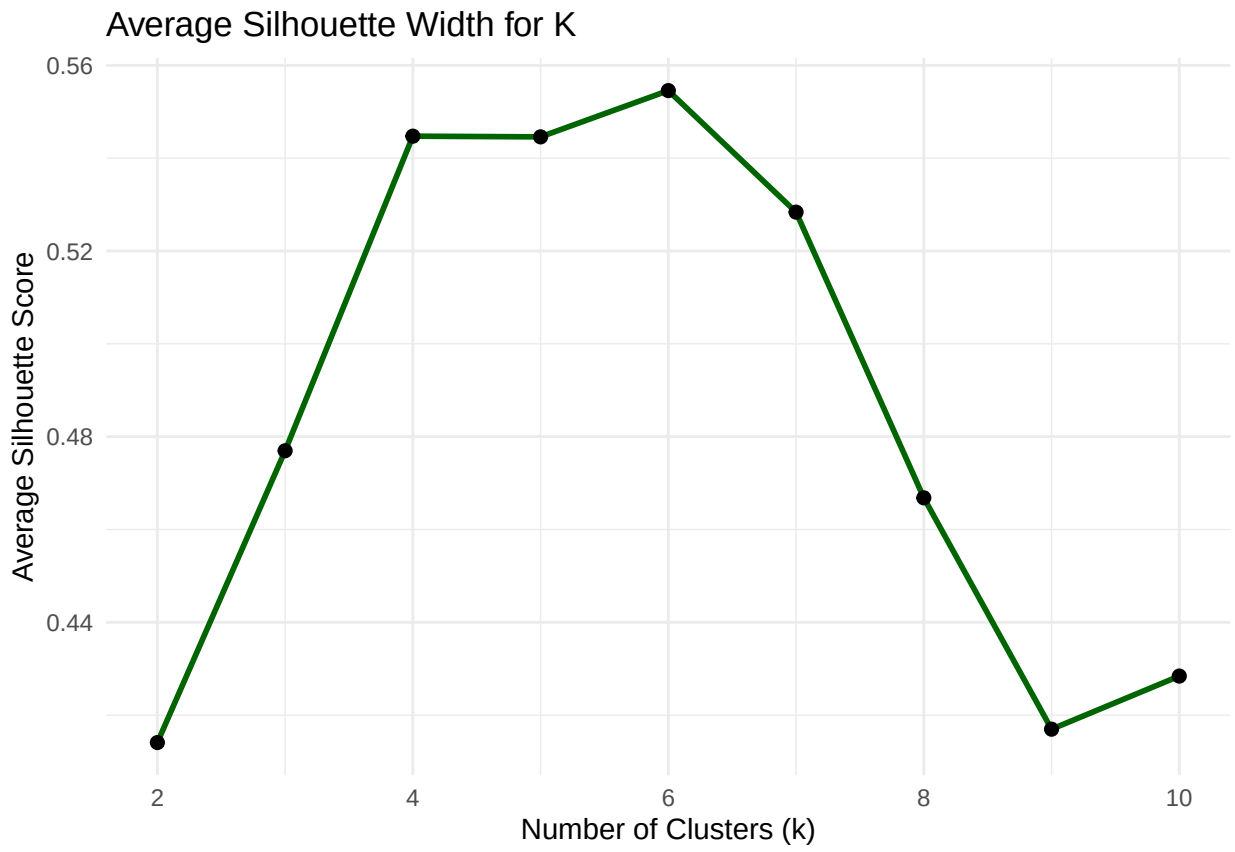
Elbow graph for finding optimal K

```r
# We'll try k from 1 to 10
wss <- numeric(10)

set.seed(42)
for (k in 1:10) {
  kmeans_model <- kmeans(nutrient_matrix_clean, centers = k, nstart = 25)
  wss[k] <- kmeans_model$tot.withinss
}

# Create elbow plot
elbow_df <- data.frame(Clusters = 1:10, WSS = wss)

library(ggplot2)
ggplot(elbow_df, aes(x = Clusters, y = WSS)) +
  geom_line(color = "steelblue", linewidth = 1) +
  geom_point(size = 2) +
  theme_minimal() +
  labs(title = "Elbow Method for Optimal K",
       x = "Number of Clusters (k)",
       y = "Total Within-Cluster Sum of Squares (WSS)")
```



Our elbow graph seems to point to an optimal k at around 3 to 4. let's run another test to see if it's consistent.

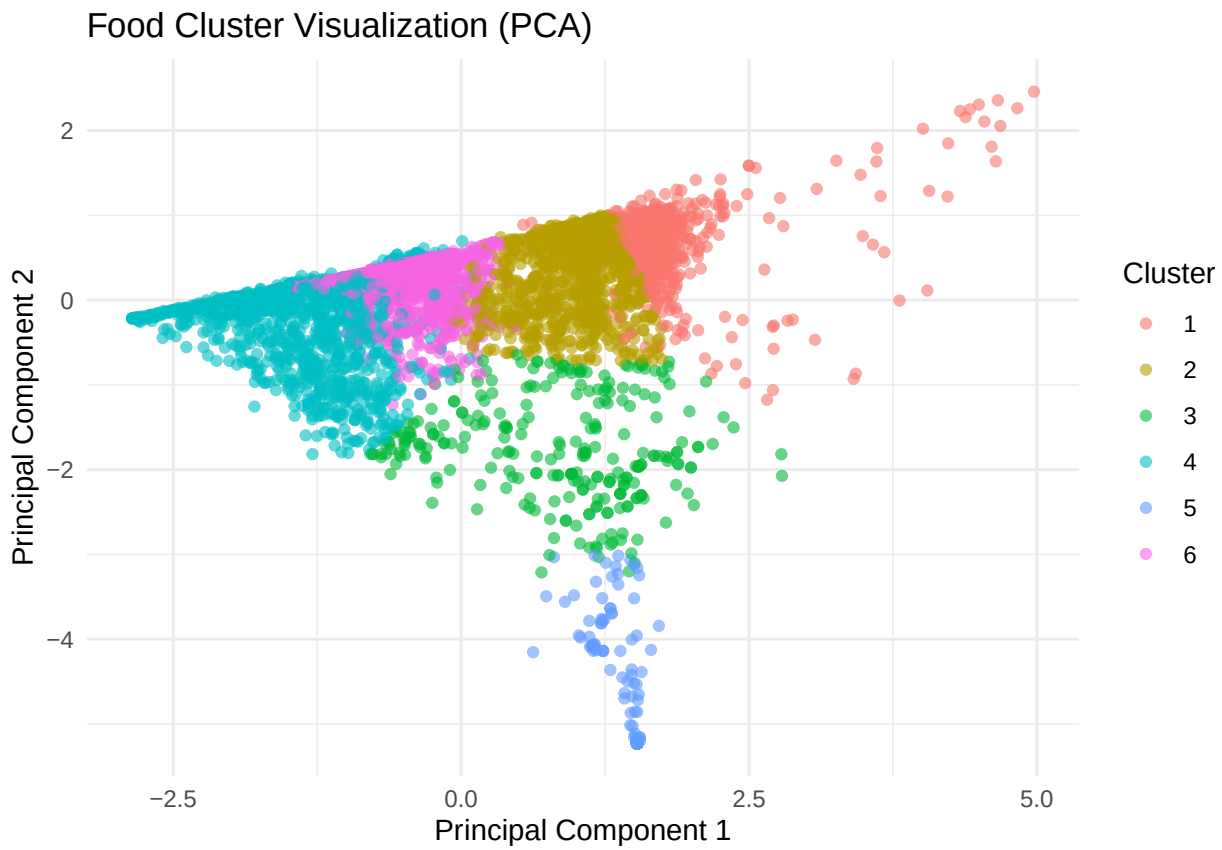
```r
library(cluster)

sil_width <- numeric(9)  # k from 2 to 10

set.seed(450)
for (k in 2:10) {
  km_res <- kmeans(nutrient_matrix_clean, centers = k, nstart = 25)
  ss <- silhouette(km_res$cluster, dist(nutrient_matrix_clean))
  sil_width[k - 1] <- mean(ss[, 3])  # column 3 = silhouette width
}

# Plot silhouette scores
sil_df <- data.frame(Clusters = 2:10, Silhouette = sil_width)

library(ggplot2)
ggplot(sil_df, aes(x = Clusters, y = Silhouette)) +
  geom_line(color = "darkgreen", linewidth = 1) +
  geom_point(size = 2) +
  theme_minimal() +
  labs(title = "Average Silhouette Width for K",
       x = "Number of Clusters (k)",
       y = "Average Silhouette Score")
```



Here we will decide our optimal k by which one maximizes our silhouette score. k = (4,5,6) all seem like valid options.

A Less Naive Clustering.

```r
# K-means clustering
set.seed(750)
k_clusters <- kmeans(nutrient_matrix_clean, centers = 6)

# Add cluster back to data
nutrient_wide_final$Cluster <- factor(k_clusters$cluster)

# PCA visualization
pca <- prcomp(nutrient_matrix_clean)
pca_df <- as.data.frame(prcomp(nutrient_matrix_clean)$x)
pca_df$Cluster <- nutrient_wide_final$Cluster



ggplot(pca_df, aes(PC1, PC2, color = Cluster)) +
  geom_point(alpha = 0.6) +
  theme_minimal() +
  labs(title = "Food Cluster Visualization (PCA)",
       x = "Principal Component 1",
       y = "Principal Component 2")
```



Signifigance of estimated clusters

```
# Identify nutrient columns (excluding ID or non-numeric vars)
nutrient_cols <- nutrient_wide_final %>%
  dplyr::select(where(is.numeric)) %>%
  dplyr::select(-FoodID) %>%
  colnames()

# Run ANOVA for each nutrient
anova_results <- lapply(nutrient_cols, function(col) {
  formula <- as.formula(paste("`", col, "` ~ Cluster", sep = ""))
  fit <- aov(formula, data = nutrient_wide_final)
  summary(fit)[[1]][["Pr(>F)"]][1]  # extract p-value
})

# Combine results into a data frame
f_test_df <- data.frame(
  Nutrient = nutrient_cols,
  P_Value = unlist(anova_results)
)

# Optional: add significance stars
f_test_df <- f_test_df %>%
  mutate(Significant = case_when(
    P_Value < 0.001 ~ "***",
    P_Value < 0.01  ~ "**",
    P_Value < 0.05  ~ "*",
    TRUE ~ ""
  )) %>%
  arrange(P_Value)

print(f_test_df)
```

```
##                              Nutrient P_Value Significant
## 1                             PROTEIN       0         ***
## 2                  FAT (TOTAL LIPIDS)       0         ***
## 3 CARBOHYDRATE, TOTAL (BY DIFFERENCE)       0         ***
```

Visualizing The Difference in Nutrition.

Here we visualize which nutrients were selected in each cluster. From our second graph we can tell how well each nutrient was selected for. I've also gone ahead and put the anova statistics within the graph

Here we can see the composition of each cluster by their respective nutrients. This box plot makes the most sense when comparing a cluster across facets.

For instance, let's draw attention to cluster 1 (in red)

> The first facet shows the average carbohydrate ammount in 100g's for each cluster. We can see that within the first cluster, it contains very litte food with high carbohydrate nutrients.
>
> Moving onto facet 2, we see that this cluster's foods has a low amounts of fats.

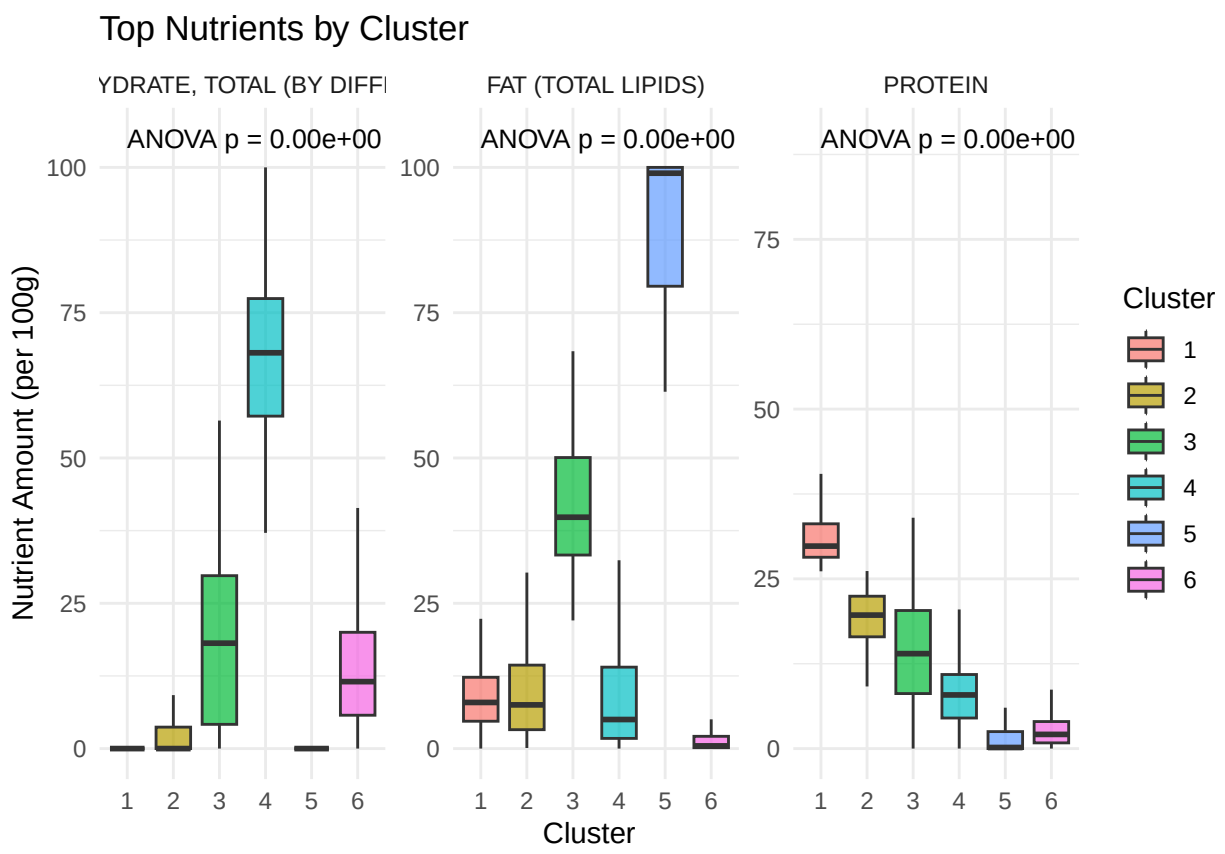
Moving onto facet three, we see that this cluster contains foods high in protien content.

```
nutrient_wide_final %>%
  group_by(Cluster) %>%
  summarise(across(where(is.numeric), ~ round(mean(.x, na.rm = TRUE), 2))) %>%
  arrange(Cluster)
```

```
## # A tibble: 6 x 5
##   Cluster  FoodID PROTEIN `FAT (TOTAL LIPIDS)` CARBOHYDRATE, TOTAL (BY DIFFERE~1
##   <fct>     <dbl>   <dbl>                <dbl>                              <dbl>
## 1 1        47347.   33.0                 9.61                               2.99
## 2 2        73832.   19.3                 9.3                                3.29
## 3 3        77775.   14.3                42.0                               20.3
## 4 4        89072.    8.19                8.61                              67.4
## 5 5        59183.    1.86               89.8                                1.44
## 6 6       140589.    2.87                2.22                              13.5
## # i abbreviated name: 1: `CARBOHYDRATE, TOTAL (BY DIFFERENCE)`
```

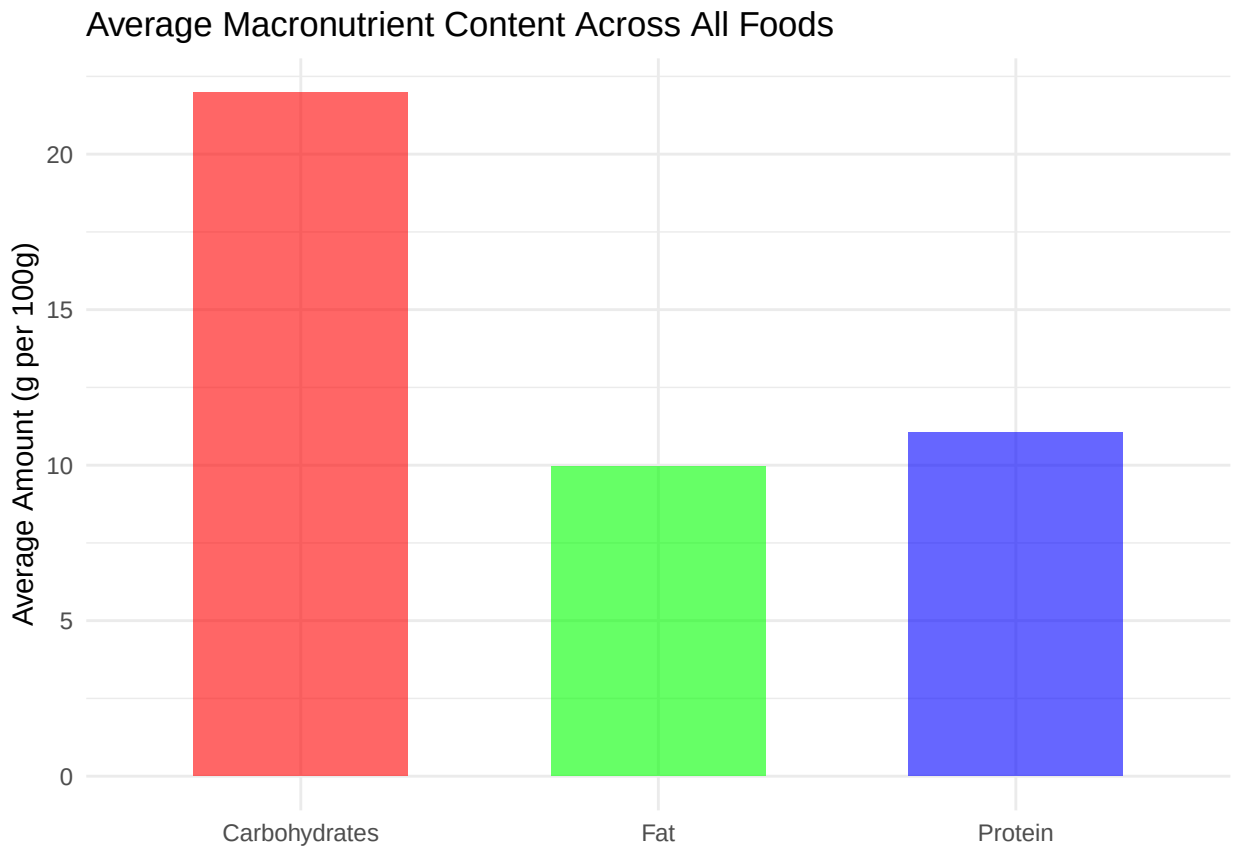
```
top_nutrients <- c("PROTEIN", "FAT (TOTAL LIPIDS)", "CARBOHYDRATE, TOTAL (BY DIFFERENCE)")

# Run ANOVA for each top nutrient
p_vals <- lapply(top_nutrients, function(nutrient) {
  formula <- as.formula(paste0("`", nutrient, "` ~ Cluster"))
  fit <- aov(formula, data = nutrient_wide_final)
  summary(fit)[[1]][["Pr(>F)"]][1]
})

# Convert to data frame
annot_df <- data.frame(
  Nutrient = top_nutrients,
  p_val = formatC(unlist(p_vals), format = "e", digits = 2)
)


# Optional: format label string
annot_df$label <- paste0("ANOVA p = ", annot_df$p_val)

# Prepare long data for plotting
plot_data <- nutrient_wide_final %>%
  pivot_longer(cols = all_of(top_nutrients), names_to = "Nutrient", values_to = "Value")

# Choose appropriate y-position (adjust if needed)
label_y_pos <- plot_data %>%
  group_by(Nutrient) %>%
  summarise(y = max(Value, na.rm = TRUE) * 1.05)

annot_df <- annot_df %>%
  left_join(label_y_pos, by = "Nutrient")

# Final plot
ggplot(plot_data, aes(x = Cluster, y = Value, fill = Cluster)) +
  geom_boxplot(alpha = 0.7, outlier.shape = NA) +
  facet_wrap(~ Nutrient, scales = "free_y") +
```

```
  geom_text(data = annot_df, aes(x = 1, y = y, label = label),
            inherit.aes = FALSE, hjust = 0, size = 3.5) +
  theme_minimal() +
  labs(title = "Top Nutrients by Cluster",
       y = "Nutrient Amount (per 100g)", x = "Cluster") +
  theme(legend.position = "right")
```



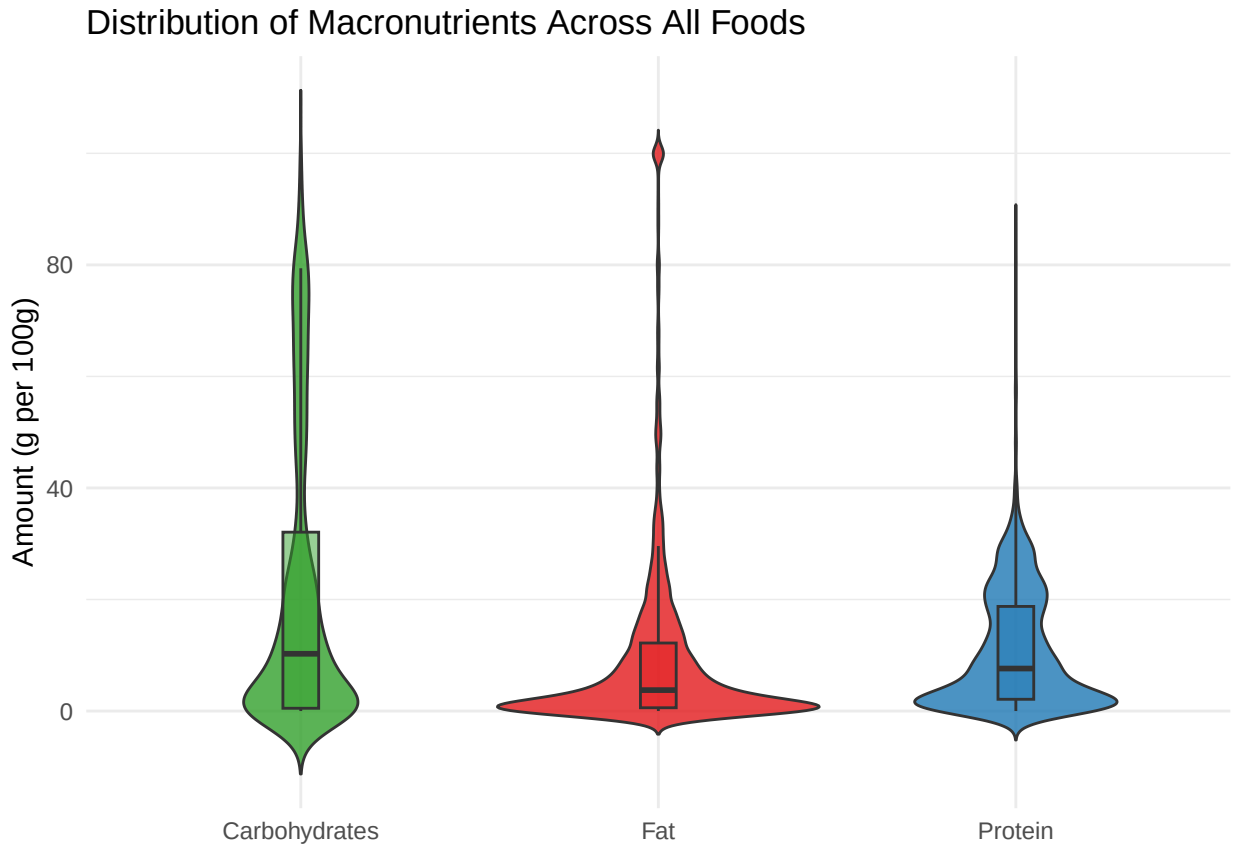

```
macro_avgs <- nutrient_wide_final %>%
  summarise(
    Protein = mean(`PROTEIN`, na.rm = TRUE),
    Carbohydrates = mean(`CARBOHYDRATE, TOTAL (BY DIFFERENCE)`, na.rm = TRUE),
    Fat = mean(`FAT (TOTAL LIPIDS)`, na.rm = TRUE)
  ) %>%
  pivot_longer(cols = everything(), names_to = "Macronutrient", values_to = "Average")


ggplot(macro_avgs, aes(x = Macronutrient, y = Average, fill = Macronutrient)) +
  geom_col(width = 0.6, alpha = 0.6) +
  theme_minimal() +
  scale_fill_manual(values = c(
    "Carbohydrates" = "red",
    "Fat" = "green",
    "Protein" = "blue"
  )) +
  labs(title = "Average Macronutrient Content Across All Foods",
```

```
       y = "Average Amount (g per 100g)", x = NULL) +
  theme(legend.position = "none")
```




```
###

nutrient_means_by_cluster <- nutrient_wide_final %>%
  dplyr::select(-FoodID) %>%
  group_by(Cluster) %>%
  summarise(across(where(is.numeric), ~ round(mean(.x, na.rm = TRUE), 2))) %>%
  arrange(Cluster)



################################################################################


# Prepare long format data from nutrient_wide_final
macro_dist <- nutrient_wide_final %>%
  dplyr::select(`PROTEIN`, `CARBOHYDRATE, TOTAL (BY DIFFERENCE)`, `FAT (TOTAL LIPIDS)`) %>%
  rename(
    Protein = `PROTEIN`,
    Carbohydrates = `CARBOHYDRATE, TOTAL (BY DIFFERENCE)`,
    Fat = `FAT (TOTAL LIPIDS)`
  ) %>%
```

```r
  pivot_longer(cols = everything(), names_to = "Macronutrient", values_to = "Amount")

# Violin plot
ggplot(macro_dist, aes(x = Macronutrient, y = Amount, fill = Macronutrient)) +
  geom_violin(alpha = 0.8, trim = FALSE) +
  geom_boxplot(width = 0.1, outlier.shape = NA, alpha = 0.5) +
  scale_fill_manual(values = c(
    "Protein" = "#1f78b4",
    "Carbohydrates" = "#33a02c",
    "Fat" = "#e31a1c"
  )) +
  theme_minimal() +
  labs(title = "Distribution of Macronutrients Across All Foods",
       y = "Amount (g per 100g)", x = NULL) +
  theme(legend.position = "none")
```



Label/Describe Clusters Meaningfully

This part of the project is one of the more important steps.

Our goal here is to name and label our clusters so they're interpretable.

From our 6 clusters created earlier, we can rank our food in terms of their macro-nutrient composition.

`Very High-Carbs, Very High-Protien, Very High-Fat.`

```
High-Carbs, High-Protien, High-Fat.
Moderate-Carbs,...
Sufficent-Carbs,...
Low-Carbs,...
Very-Low,...
```

```r
# Here we have the means for each macro nutrient from each of our 6 clusters.
cluster_summary <- nutrient_wide_final %>%
  group_by(Cluster) %>%
  summarise(
    Protein = mean(`PROTEIN`, na.rm = TRUE),
    Carbohydrates = mean(`CARBOHYDRATE, TOTAL (BY DIFFERENCE)`, na.rm = TRUE),
    Fat = mean(`FAT (TOTAL LIPIDS)`, na.rm = TRUE),
    .groups = 'drop'
  )

head(cluster_summary)
```

```
## # A tibble: 6 x 4
##   Cluster Protein Carbohydrates   Fat
##   <fct>     <dbl>         <dbl> <dbl>
## 1 1         33.0           2.99  9.61
## 2 2         19.3           3.29  9.30
## 3 3         14.3          20.3  42.0
## 4 4          8.19         67.4   8.61
## 5 5          1.86          1.44 89.8
## 6 6          2.87         13.5   2.22
```

```r
# The centroids of our k-means clusters are "Non-parametric".
# If we would have clustered according to the shape of our distribution, we would already have them nam

# So we will take an ordered approach to our clusters.
# i.e. the cluster with the highest mean protein will be labelled as "Very High"
  # The second highest cluster in protein will be labelled "High". etc
  # The same follows for carbs and fats.


# Note we will have an overlap problem.
# i.e. foods high in fats and protein (i.e. new york striploin.)
# Therefor we want 3 columns. 1 for each macro nutrients.
# each column will contain one of the 6 rankings.


cluster_means <- cluster_summary %>%
  mutate(
    Protein_Rank = rank(-Protein, ties.method = "min"),  # descending (higher = better)
    Carb_Rank = rank(-Carbohydrates, ties.method = "min"),
    Fat_Rank = rank(-Fat, ties.method = "min")
  )



rank_labels <- c("Very High", "High", "Moderate", "Sufficient", "Low", "Very Low")
```

```r
cluster_means <- cluster_means %>%
  mutate(
    Protein_Label = rank_labels[Protein_Rank],
    Carb_Label = rank_labels[Carb_Rank],
    Fat_Label = rank_labels[Fat_Rank]
  )

head(cluster_means)
```

```
## # A tibble: 6 x 10
##   Cluster Protein Carbohydrates   Fat Protein_Rank Carb_Rank Fat_Rank
##   <fct>     <dbl>         <dbl> <dbl>        <int>     <int>    <int>
## 1 1         33.0           2.99  9.61            1         5        3
## 2 2         19.3           3.29  9.30            2         4        4
## 3 3         14.3          20.3  42.0             3         2        2
## 4 4          8.19         67.4   8.61            4         1        5
## 5 5          1.86          1.44 89.8             6         6        1
## 6 6          2.87         13.5   2.22            5         3        6
## # i 3 more variables: Protein_Label <chr>, Carb_Label <chr>, Fat_Label <chr>
```

```r
# From this we have an immediate issue.
# Clusters 3,2, and 5 suffer from "non-uniqueness".
  # Where one cluster shows the same level of nutrients for two or more macro nutrients.
  # i.e cluster 2 contains foods both "Sufficient" in fats and carbs,
      # cluster 3 contains foods both "High" in fats and carbs
      # cluster 5 contains foods both "Very Low" in protein and carbs.
# Let's try a different solution


# cluster_means <- cluster_means %>%
#   arrange(desc(Protein)) %>%
#   mutate(Protein_Rank = dense_rank(desc(Protein))) %>%
#
#   arrange(desc(Carbohydrates)) %>%
#   mutate(Carb_Rank = dense_rank(desc(Carbohydrates))) %>%
#
#   arrange(desc(Fat)) %>%
#   mutate(Fat_Rank = dense_rank(desc(Fat)))

# Does not change a thing.
# We got this far with our k-means process. let's continue with a radar plot for now.


nutrient_wide_final <- nutrient_wide_final %>%
  left_join(cluster_means %>% dplyr::select(Cluster, Protein_Label, Carb_Label, Fat_Label), by = "Cluste
```

The right tool for the right problem.

I'm not quite happy with the incompleteness of our rankings.

As a result of our k-means centroids being chosen randomly (it's a non parametric process) this was bound to happen.

We're abandoning the k-means process. and moving to ranking foods based off their place in within the distribution.

From this we can identify food high in protiens, fats, carbs.

Immediate Next Steps (Analysis & Interpretation)

```
Label/Describe Clusters Meaningfully
Use nutrient averages (already computed) to give each cluster a descriptive label:

    e.g., "High-Protein, Low-Fat Foods", "Carb-Dense Processed Foods", etc.

Add Cluster Counts & Food Examples

    Create a table showing number of foods per cluster

    Include top 3-5 example foods (e.g., by frequency or nutrient levels) per cluster

Radar Chart for Cluster Profiles

    Visualize average nutrient profiles per cluster using radar/spider plots

Explore Micronutrients

    Run a second round of clustering or PCA using key micronutrients (e.g., Iron, Calcium, Sodium, Vita

    Compare cluster behavior when switching nutrient focus

Profile Food Groups Across Clusters

    Tabulate or visualize how CNF's "FoodGroup" categories (e.g., Vegetables, Meats) are distributed ac
```

Visualization Enhancements

```
Heatmap of Nutrient Averages by Cluster

    Cluster rows and/or nutrients visually

    Make nutrient-level comparisons across clusters easy

Silhouette Plot per Food

    Plot silhouette values per food (bar/line), highlighting poorly clustered cases

Parallel Coordinates Plot

    Allows users to track multiple nutrient values per food across clusters (good for web dashboards)
```

Statistical & Machine Learning Extensions

```
Compare K-Means with Hierarchical Clustering or DBSCAN
```

Useful for checking whether a different clustering algorithm yields better structure (esp. for irreg

Build a Simple Classifier (Optional)

Try classifying foods into clusters using decision trees or random forests

Helps understand decision boundaries and important nutrients

Dimensionality Reduction Alternatives

Explore t-SNE or UMAP for visualizing high-dimensional nutrient data

Less immediate long steps.

Incorporate External Data Sources Why? To enrich context, increase real-world relevance, or link to behavior/policy.

Link with price data (e.g., grocery store APIs or Statistics Canada's food CPI)

Compare nutritional value per dollar

Cluster based on nutrient density and affordability

Add environmental impact data

Carbon emissions, land use, water usage of food categories (e.g., from Our World in Data, FAO)

Explore trade-offs between nutrition and sustainability

Link to health outcomes

Pull dietary intake data from CCHS-Nutrition or NHANES (US) and estimate how common these foods are

Hypothesize relationships between food clusters and population health issues (e.g., sodium and hype

Drawbacks to K-means clustering.

1. Sensitive to seed selection.
2. Non-parametric estimation is incredibly useful for prediction. not so much explanation. (which is what were attempting to do right now)
3. interpretability of clusters is difficult when the centroids are not aligned with the underlying distribution of the data.