Homework Assignment 3

Greg Forkutza
Student ID: 400277514

11 November, 2023

Contents

1 Report

In this study, we analyze the Mice Protein Expression data set, sourced from the UCI Machine Learning Repository Clara Higuera [2015]. This data set comprises expression levels of 77 proteins, each measured 15 times in the cerebral cortex of 72 mice, totaling 1080 observations. Among these mice, 38 were of the control genotype and 34 of the trisomic (Down syndrome) genotype. The mice were subjected to treatment injections with either saline (control) or memantine, a drug used for treating moderate to severe Alzheimer's disease Kuns et al. [2023]. Furthermore, a subset of these mice underwent learning stimulation, leading to eight distinct classes derived from this combination of genotype and dual treatment factors.

The data set contains 81,360 individual protein expression measurements, with 1,396 missing values. The distribution pattern of these missing values is illustrated in Figure 2. We excluded 3 observations that exhibited over 50% missing data. The proteins with missing values are listed in Table 1. Additionally, significant multicollinearity was observed among the features. By analyzing the correlation between non-missing and missing columns, we removed one feature from each pair exhibiting a correlation exceeding 0.8.

Outlier detection was performed using isolation forests via the isotree package Cortes, David. [2023], identifying three outliers with scores above 0.56, which were subsequently removed. The remaining missing values in five features were imputed using multiple imputation with classification and regression trees, facilitated by the mice package van Buuren et al. [2023]. The distribution of these features, both pre- and post-imputation, is depicted in Figure 3, demonstrating a satisfactory fit. Ten imputed datasets were generated, and following the application of each clustering method, results were pooled. Reapplication of isolation forests confirmed the absence of new outliers post-imputation. All protein expression features were then scaled and centered for further analysis.

The primary objective is to categorize the data into eight clusters (k=8) based solely on protein expressions, employing a semi-supervised learning approach. By examining the actual class labels, we can assess the distribution of classes within the clusters. Once clusters are identified as corresponding to specific classes, we aim to ascertain the most influential protein expressions defining group membership within each cluster. Subsequently, box plots will be utilized to visualize the distribution of these pivotal proteins, exploring the possibility of certain proteins emerging as primary indicators for membership in particular classes.

K-means clustering was employed to categorize the proteins into $k = 8$ distinct clusters, aligning with the eight predefined classes of mice. This initial model, incorporating 45 variables proved to be complex and challenging to interpret. Consequently, a random forest approach was utilized for feature selection. By aggregating the mean decrease in accuracy and mean decrease in Gini scores, the top ten features were selected, each surpassing a cumulative score threshold of 25. The correlation matrix heat map of these features is presented in Figure 4.

Subsequently, the k-means clustering model was refined using these ten proteins, utilizing the `stats::kmeans()` function. Given the eight distinct classes of mice, the model was structured to identify eight corresponding clusters. Additionally, divisive clustering was performed on the same set of proteins using `cluster::diana()`, as in Maechler et al. [2021], and the resulting dendrogram was pruned to also yield eight clusters. To elucidate the relationship between these clusters and the predefined mouse classes, we examined the frequency of each class within the clusters. Moreover, to understand the central tendency of each protein within each cluster, the centroid expression values for k-means and the mean values for divisive clustering were calculated. These analyses are visually represented in Figure 1. Furthermore, a Principal Component Analysis (PCA) was conducted, reducing the data dimensionality to four principal components, a decision guided by the elbow method illustrated in Figure 6. The class frequency distribution within the PCA-based clusters is depicted in Figure 5.

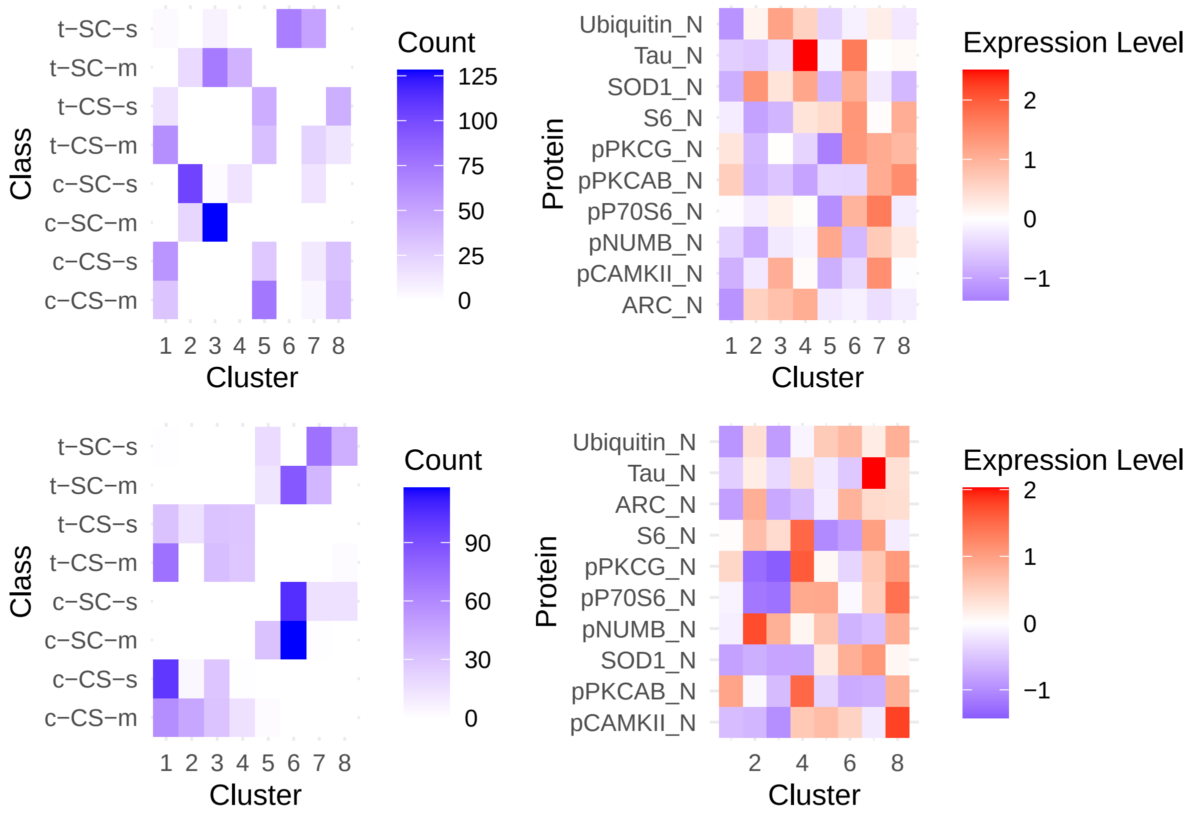

Figure 1: The top row is k-means clustering and the middle row is for divisive clustering. The left column is the frequency count of class per cluster and the right column is the expression mean protein expression level in each cluster.

The silhouette score was calculated for all clustering methods: k-means, divisive clustering, and PCA-enhanced k-means. The average silhouette width was determined for each method, yielding values of 0.21 for divisive clustering, 0.36 for k-means, and 0.43 for PCA-enhanced k-means. The PCA-enhanced k-means approach demonstrated the highest average silhouette width, suggesting a more distinct clustering configuration compared to the other methods. Although PCA-enhanced k-means excelled in cluster separation, it was incapable of identification of specific proteins that are crucial for defining group membership within each cluster. Therefore the standard k-means clustering results were deemed more suitable for further analysis and interpretation.

This analysis, while exploratory in nature, aims to delineate potential patterns in protein expression across the identified clusters. This analysis is guided by visual representations in Figures 1, 7, and 8, each elucidating the distribution of protein expressions within respective clusters.

In Cluster 2, which encompasses 103 of the 145 observations classified as c-SC-S, could represent a "pure-control" baseline, encompassing mice with the control genotype, no exposure to shock learning, and absence of drug treatment. Notably, this cluster exhibits a pronounced expression of SOD1_N, which might be characteristic of this group. Cluster 3 presents a unique composition, consisting of 128 observations classified as c-SC-m and 127 as t-SC-m, with no other classes represented. This pattern suggests that while this cluster does not differentiate between trisomic and control genotypes, it is indicative of mice not subjected to shock learning but treated with Alzheimer's medication. The heightened expression of Ubiquitin_N in this cluster is noteworthy and might imply that high concentrations of this protein are indicative of mice in this specific treatment regime, irrespective of the genotype. Conversely, Cluster 6 is exclusively composed of 71 observations, all from the t-SC-S class. This cluster seems to represent a baseline for trisomic mice who neither received shock learning nor drug treatment. The uniform and elevated expressions of Tau_N, pPKCG_N, S6_N, and SOD1_N, unique to this cluster, suggest these proteins as potential markers for trisomic mice under these specific conditions. In Cluster 4, with a majority of 42 out of 57 observations belonging to the t-SC-m class, we observe a pattern that could be representative of trisomic mice without shock learning but who underwent drug treatment. This cluster is distinguished by a notably high expression of TAU_N and a unique low expression of S6_N, suggesting that this combination of protein expressions might be indicative of trisomic mice in this particular treatment context.

2 Supplementary Material

```r
# Download data set from UCI repo, unpack and convert to csv.
url <- "https://archive.ics.uci.edu/static/public/342/mice+protein+expression.zip"
zip_file <- tempfile(fileext = "mice+protein+expression.zip")
unzip_dir <- tempdir()
download.file(url, zip_file, mode="wb")
unzip(zip_file, exdir = unzip_dir)
data_file <- list.files(unzip_dir, pattern = "\\.xls$", full.names = TRUE)
df <- read_excel(data_file[1])

# Check for missing values
nmiss <- sum(is.na(df))
```

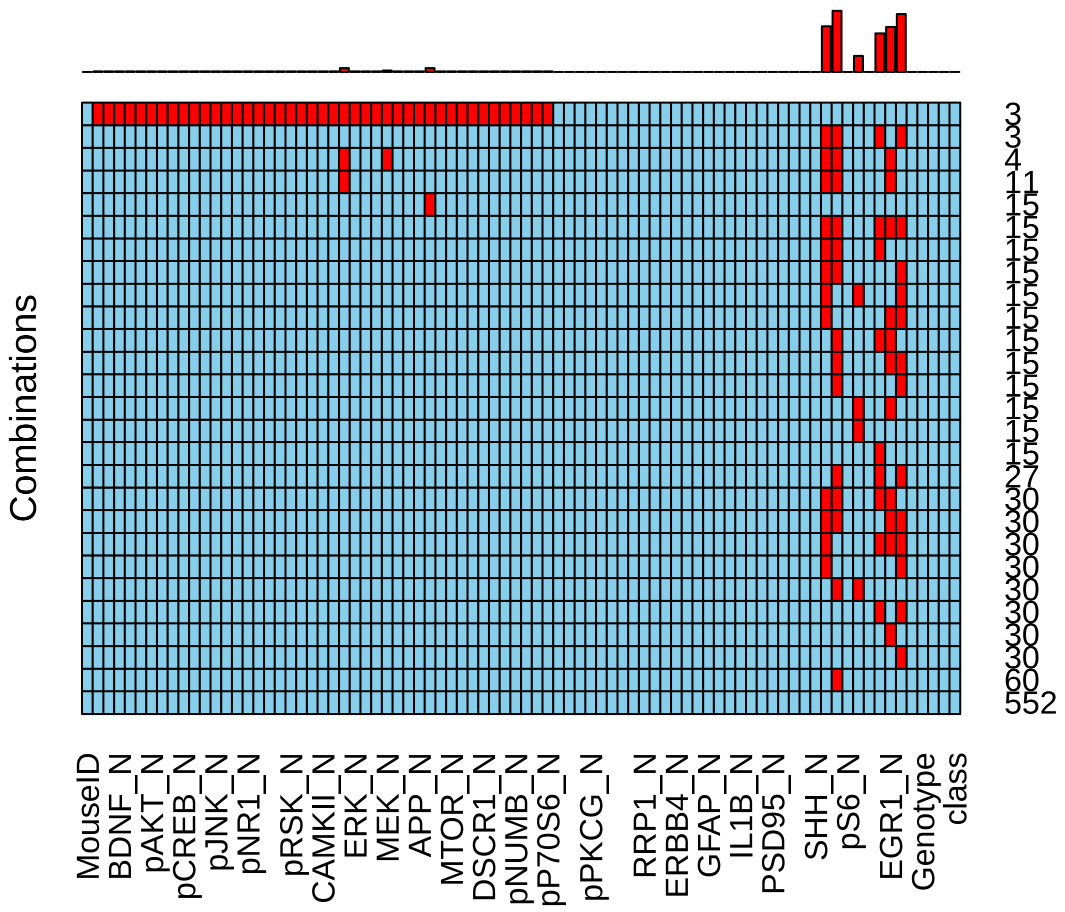

Figure 2: Using `VIM::aggr` to visualize missing data. The x-axis is the frequency for each pattern of missing data in the features seen on the y-axis.

```r
# Calculate the percentage of missing values for each row
missing_percentage <- rowSums(is.na(df)) / ncol(df)

# Identify rows with more than 50% missing values
rows_over_50_percent_missing <- which(missing_percentage > 0.5)

# Display the rows with more than 50% missing values
mis_names <- df[rows_over_50_percent_missing, ]$MouseID
print(mis_names)

# Remove observations where MouseID is in mis_names
df <- df %>%
      filter(!(MouseID %in% mis_names))
```

```r
# Calculate the percentage of missing values for each column
missing_percentage_by_feature <- colSums(is.na(df)) / nrow(df) * 100
kable(tail(sort(missing_percentage_by_feature), n = 9),
      caption = "Proteins with missing values by percentage missing
      (\\#tab:missing-values)",
             format = "latex", booktabs = TRUE) %>%
  kable_styling(latex_options = c("striped", "hold_position"))
```

Table 1: Proteins with missing values by percentage missing

	x
MEK_N	0.6481481
ELK_N	1.6666667
Bcatenin_N	1.6666667
pCFOS_N	6.9444444
H3AcK18_N	16.6666667
EGR1_N	19.4444444
BAD_N	19.7222222
H3MeK4_N	25.0000000
BCL2_N	26.3888889

```r
# Calculate correlation matrix exlcuding non numeric columns
proteins_df <- df[, -c(1, (ncol(df)-6):ncol(df))]
n <- ncol(proteins_df)

# Identify columns with Missing Values
```

```r
cols_with_na <- which(colSums(is.na(proteins_df)) > 0)

# Compute correlation between Non-NA and NA Columns
proteins_df_no_na <- proteins_df %>%
  select(-all_of(cols_with_na))
cor_with_na <- cor(proteins_df_no_na, df[,cols_with_na],
                   use = "pairwise.complete.obs")

# Identify highly correlated features
highly_correlated_features <-
  colnames(cor_with_na)[colSums(cor_with_na > 0.8) > 0]

# Remove highly correlated features
proteins_df <- proteins_df %>%
  select(-all_of(highly_correlated_features))

# Recalculate the correlation matrix
cor_matrix <- cor(proteins_df, use = "pairwise.complete.obs")

# Identify multicollinear variables
multicollinear_vars <- findCorrelation(cor_matrix, cutoff = 0.8,
                                       verbose = FALSE)

# Remove multicollinear variables
proteins_df <- proteins_df %>%
  select(-all_of(multicollinear_vars))
```

```r
# Applying isolation forest
model <- isolation.forest(proteins_df, ntrees = 100, sample_size = 256)
scores <- predict(model, proteins_df, type="score")
proteins_df$anomaly_score = scores
threshold = 0.56
proteins_df$outlier = proteins_df$anomaly_score > threshold
which(proteins_df$outlier == "TRUE")
```

```r
# Multiple Imputation
mice_mod <- mice(proteins_df, m=10,
                 method='cart',
                 maxit = 25,
                 seed = 123,
                 printFlag = FALSE)

# Combine categorical variables with imputed protein data
for(i in 1:10) {
  variable_name <- paste("data_imp", i, sep = "_")
  assign(variable_name, complete(mice_mod, i))
}

first_column <- df[, 1]
last_six_columns <- df[, (ncol(df)-5):ncol(df)]

for(i in 1:10) {
  imputed_dataset <- get(paste("data_imp", i, sep = "_"))
  combined_dataset <- cbind(first_column, imputed_dataset, last_six_columns)
  assign(paste("data_imp", i, sep = "_"), combined_dataset)
}
```

```r
# Scale data for clustering
for (i in 1:10) {
  df_name <- paste0("data_imp_", i)
  df <- get(df_name)
  cols_to_scale <- setdiff(2:ncol(df), c(1, 48:55))
  df[, cols_to_scale] <- scale(df[, cols_to_scale])
  new_df_name <- paste0("data_scaled_", i)
  assign(new_df_name, df, envir = .GlobalEnv)
}
```

```r
set.seed(123)
cluster_vs_class_df <- as.data.frame(as.table(cluster_vs_class))

protein_measurement_columns <-  data_scaled_reduced_1 %>%
  select(all_of(top_features))
```

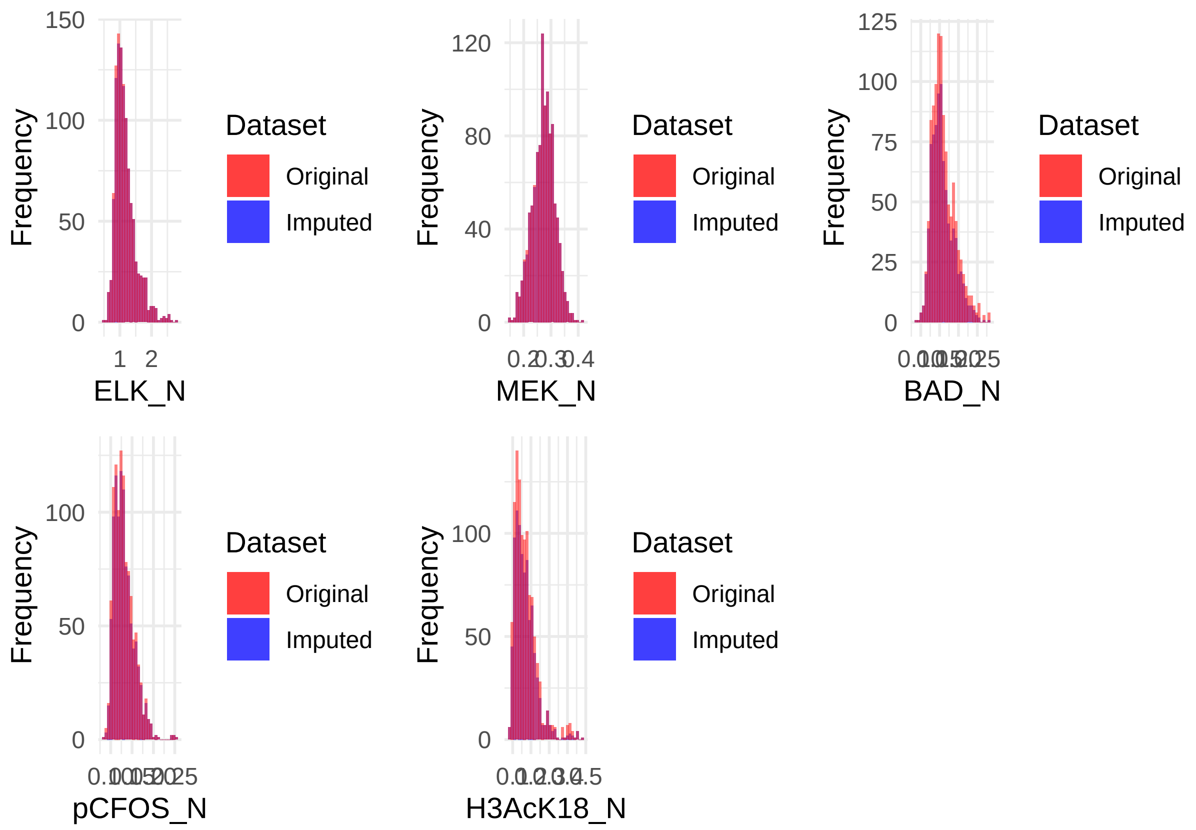

Figure 3: Histograms comparing the distribution of each feature with NA's before and after imputation. This is one of the ten data sets imputed.

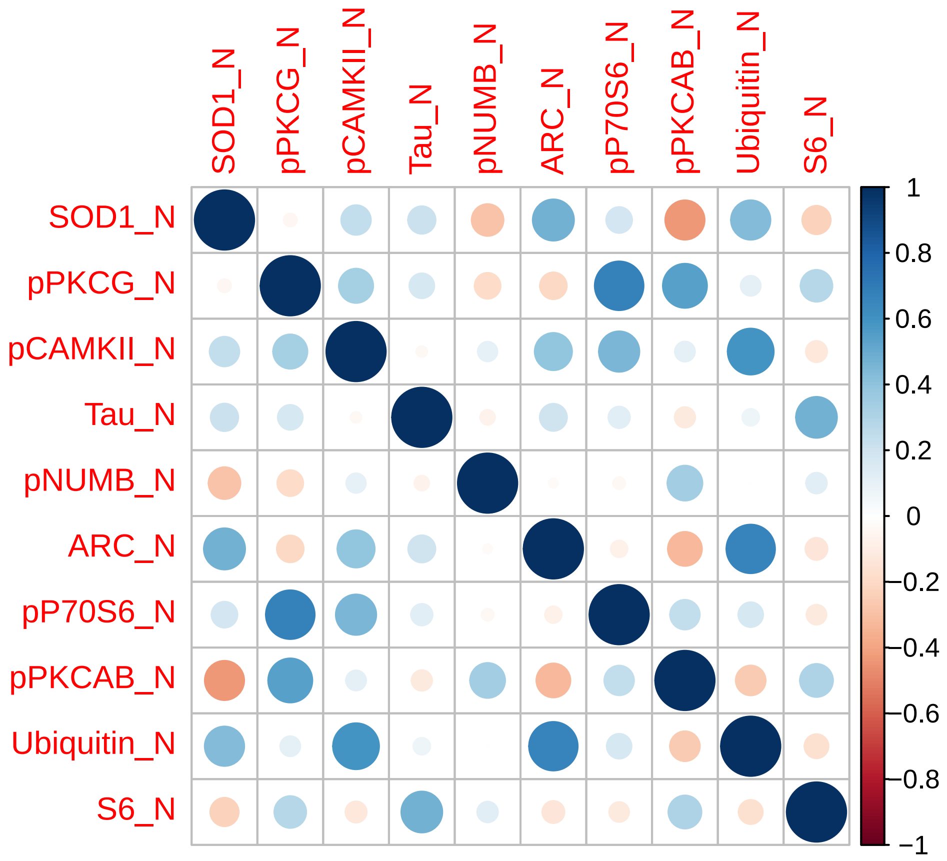

Figure 4: Heat map of correlation matrix for 10 proteins after using random forests for feature selection.

```r
kmeans_result <- kmeans(protein_measurement_columns, centers = 8, nstart = 25)
protein_measurement_columns$Cluster <-kmeans_result$cluster
cluster_vs_class <- table(Cluster = kmeans_result$cluster,
                          Class = data_scaled_reduced_1$class)

data_scaled_1$Cluster.km <- kmeans_result$cluster
cluster_vs_class <- table(Cluster = kmeans_result$cluster,
                          Class = data_scaled_1$class)

cluster_vs_class_df <- as.data.frame(as.table(cluster_vs_class))

centroids <- kmeans_result$centers



# Convert the centroids to a long format for ggplot
centroids_long <- as.data.frame(t(centroids))
centroids_long$Protein <- rownames(centroids_long)
centroids_long <- tidyr::pivot_longer(centroids_long, -Protein,
                                      names_to = "Cluster",
                                      values_to = "Expression")

# heiarchical clustering
divisive_clustering <- diana(protein_measurement_columns[,1:10])
clusters <- cutree(divisive_clustering, k = 8)
data_divisive <- cbind(data_scaled_reduced_1, clusters)
cluster_vs_class <- table(Cluster = data_divisive$clusters,
                          Class = data_scaled_1$class)
cluster_vs_class_df <- as.data.frame(as.table(cluster_vs_class))
clusters <- cutree(divisive_clustering, k = 8)
data_scaled_reduced_1$Cluster.div <- clusters
numeric_columns <- sapply(data_scaled_reduced_1, is.numeric)
cluster_means <- aggregate(data_scaled_reduced_1[,(2:11)],
                        by = list(Cluster = data_scaled_reduced_1$Cluster.div),
                        FUN = mean)
cluster_means_long <- reshape2::melt(cluster_means, id.vars = "Cluster")
```

```r
pca_result <- prcomp(protein_measurement_columns[,(1:10)],
                     center = TRUE, scale. = TRUE)
# PCA and kmeans
num_components <- which(cumsum(pca_result$sdev^2) / sum(pca_result$sdev^2) > 0.8)[1]
data_pca <- as.data.frame(pca_result$x[, 1:num_components])
kmeans_result <- kmeans(data_pca, centers = 8, nstart = 25)
data_pca$Cluster <-kmeans_result$cluster
cluster_vs_class_pca <- table(Cluster = data_pca$Cluster,
                              Class = data_scaled_reduced_1$class)
data_scaled_reduced_1$Cluster.kmpca <- data_pca$Cluster
cluster_vs_class_pca <- table(Cluster = data_scaled_reduced_1$Cluster.kmpca,
                              Class = data_scaled_reduced_1$class)
cluster_vs_class_df <- as.data.frame(as.table(cluster_vs_class_pca))



# Silohuette


sil_km <- silhouette(kmeans_result$cluster, dist(protein_measurement_columns))
sil_div <- silhouette(clusters, dist(protein_measurement_columns[, 1:10]))
sil_pca <- silhouette(kmeans_result_pca$cluster, dist(data_pca))



mean_sil_width_km <- mean(sil_km[, "sil_width"])
mean_sil_width_div <- mean(silhouette(clusters, dist_matrix)[, "sil_width"])
mean_sil_width_pca <- mean(sil_pca[, "sil_width"])
```

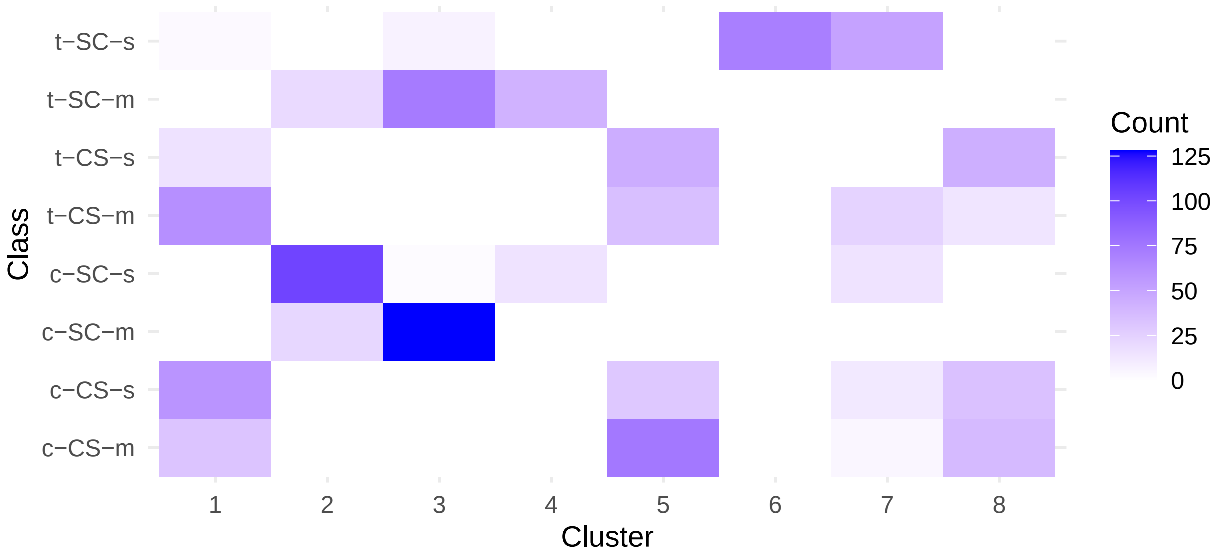

Figure 5: Distribution of classes in clusters by PCA before K-means clustering

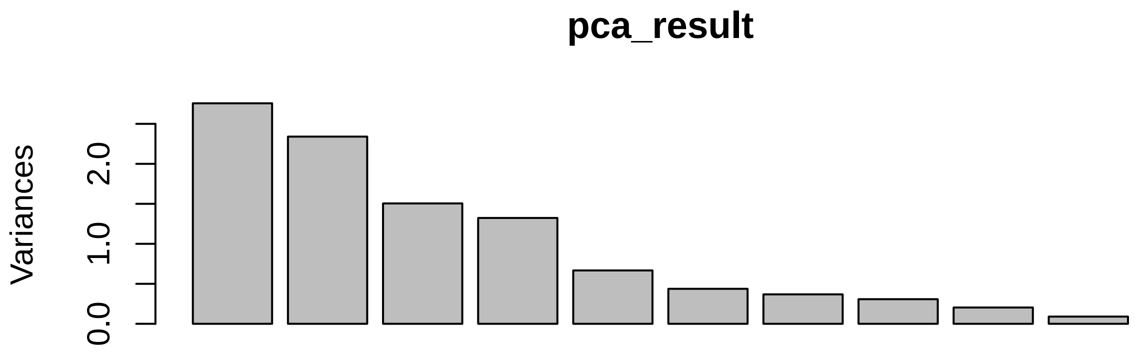

Figure 6: Variance explained per principal component.

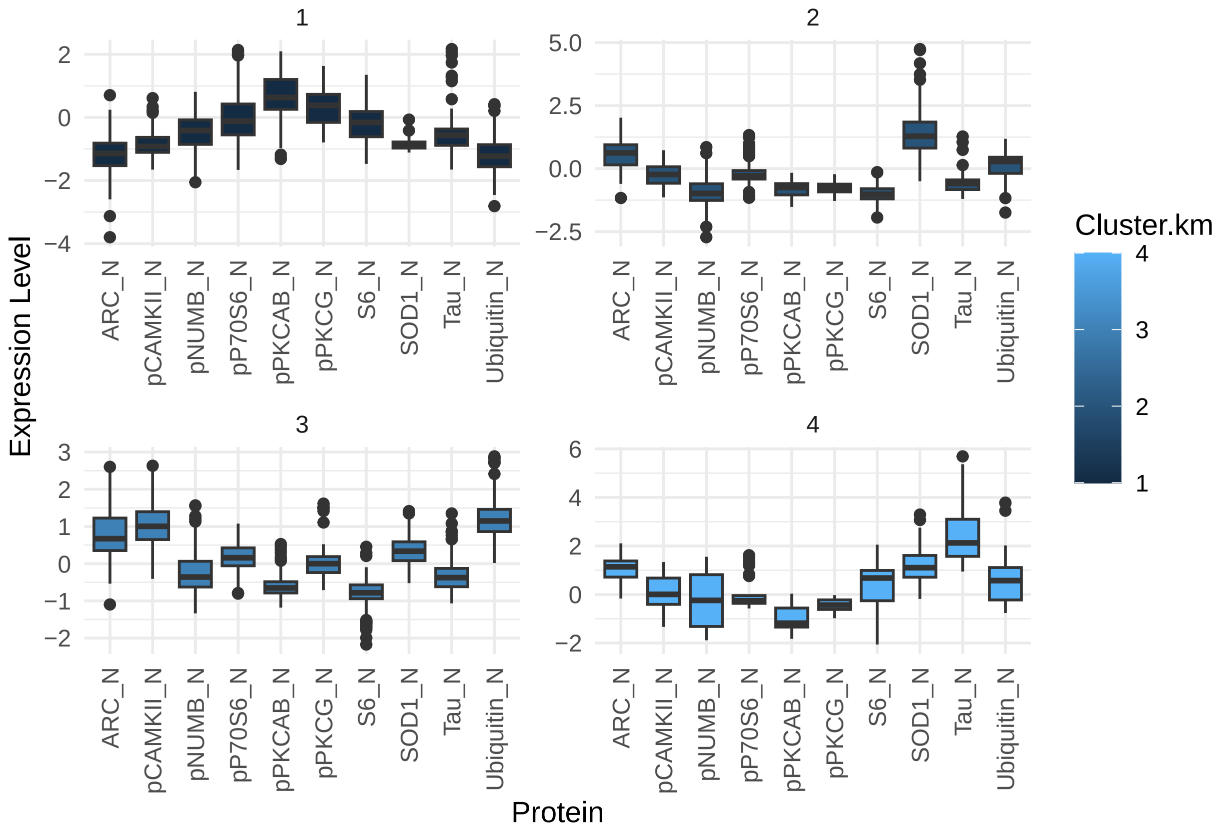

Figure 7: Boxplot of protein expression in clustering by K-means for top 10 features selected by random forest for cluster 1:4.

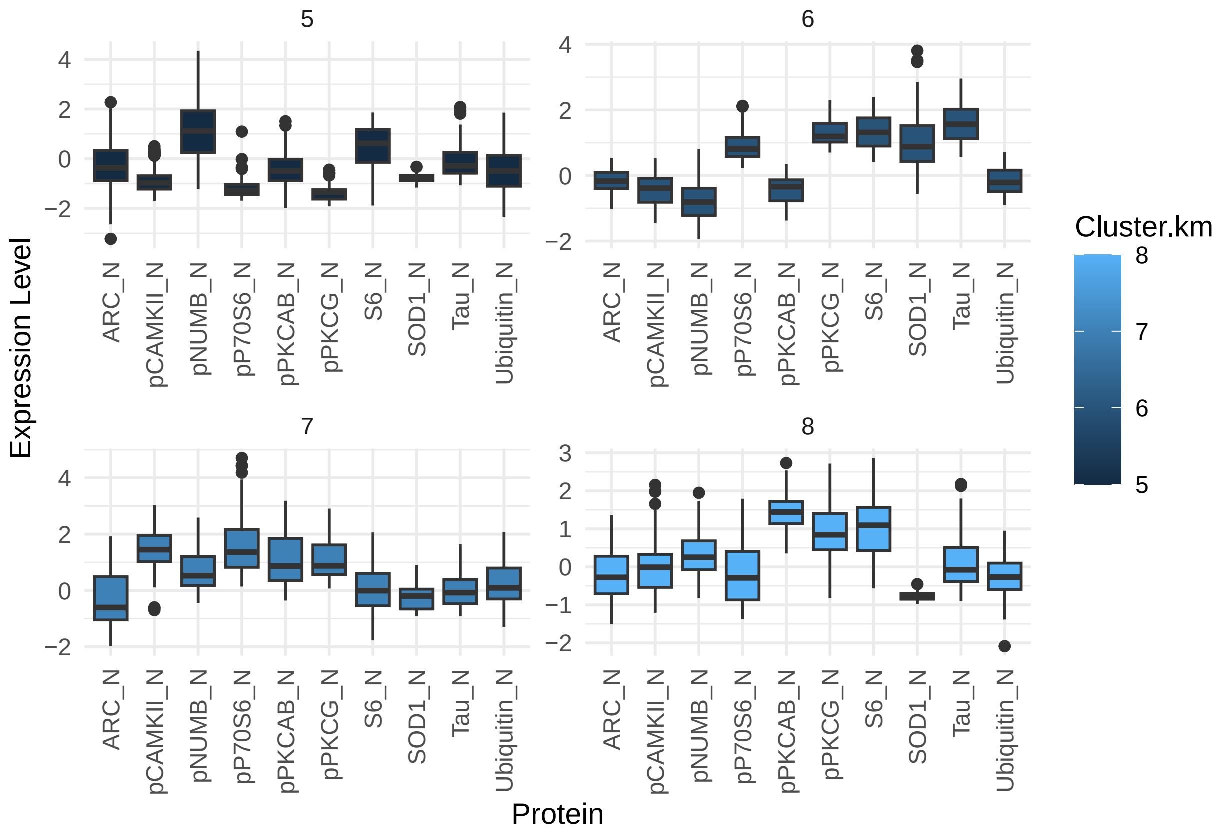

Figure 8: Boxplot of protein expression in clustering by K-means for top 10 features selected by random forest for cluster 5:8.

References

Krzysztof Cios Clara Higuera, Katheleen Gardiner. Mice Protein Expression. UCI Machine Learning Repository, 2015. DOI: https://doi.org/10.24432/C50S3Z.

Cortes, David. *isotree: Isolation-Based Outlier Detection*, 2023. URL https://cran.r-project.org/web/packages/isotree/isotree.pdf. R package version 0.5.22.

B. Kuns, A. Rosani, and D. Varghese. Memantine. In: StatPearls [Internet]. Treasure Island (FL): StatPearls Publishing, 2023. [Updated 2022 Jul 11]. Available from: https://www.ncbi.nlm.nih.gov/books/NBK500025/.

Martin Maechler, Peter Rousseeuw, Anja Struyf, Mia Hubert, and Kurt Hornik. *Cluster: Cluster Analysis Basics and Extensions*, 2021. URL https://CRAN.R-project.org/package=cluster. R package version 2.1.0.

Stef van Buuren, Karin Groothuis-Oudshoorn, Gerko Vink, Rianne Schouten, Alexander Robitzsch, Patrick Rockenschaub, Lisa Doove, Shahab Jolani, Margarita Moreno-Betancur, Ian White, Philipp Gaffert, Florian Meinfelder, Bernie Gray, Vincent Arel-Bundock, Mingyang Cai, Thom Volker, Edoardo Costantini, Caspar van Lissa, and Hanne Oberman. *mice: Multivariate Imputation by Chained Equations*, 6 2023. URL https://cran.r-project.org/web/packages/mice/mice.pdf. R package version 3.16.0.