Katholieke
Universiteit
Leuven

MSc in Bioinformatics
Applied Multivariate Statistical Analysis
[I0P16a]

INVESTIGATION OF THE NUTRIENT COMPOSITION OF COMMON GROUPS OF FOOD

The Australian Food Composition Initiative

Antoine Ruzette (r0829308)

Academic year 2021-2022

Contents

1 Introduction and motivations

Awareness around food composition is a major driver for a society living on a healthy diet. In that context, the Australian Food Composition Database was created and is publicly available. It gathers reliable nutrient composition of 1534 foods. We hereby investigate the composition of twenty-four food groups to understand their nutrients patterns. The main idea behind the present project is to pave the way of a systemic food classification based on its contents. Further developments of this project would include the prediction of a health score (i.e. NutriScore) based on the nutrient composition of the food type. Practically, first exploratory section was performed via clustering, PCA and tSNE methods. In a second section, we trained decision tree models to correctly classify the different groups of food. Lastly, we implemented an Artificial Neural Network for classification.

2 Methods

2.1 Extract, Transform and Load (ETL)

Data are accessible at `https://www.foodstandards.gov.au/science/monitoringnutrients/afcd/Pages/downloadableexcelfiles.aspx` in an excel format. The retrieved data set is composed of 1535 food items (classified in 22 food categories) and 252 nutrient information for each food item. We used the data for which all the solids and liquids were measured per 100g to avoid metric conversion. A first step of the ETL process is to subset the feature space in order to select only the meaningful features, while keeping in mind to minimize missing values. To do so, we retrieved all columns having less than 10% missing values and manually picked meaningful features within them. Each food item is classified according to the Australian Health Survey Classification System that attributes a two digit number to food categories (e.g. 11 stands for Non-alcoholic beverages, 12 for Cereals and cereal products, etc.). In the data modeling section, we will train a classifier that predicts the food category of a food item.

Before heading to the next section, namely the exploratory analysis, the data went through a standard scaler.

2.2 Exploratory Analysis

Exploratory analysis aims to find patterns in the data that aren't predicted by the experimenter's current knowledge or pre-conceptions. We hereby investigate typical exploratory methods such as clustering algorithm, PCA and tSNE (multi-dimensional scaling). We will compare them, evaluate their interpretability and performance.

2.2.1 Correlations exploration

Before starting to explore the data set with methods, let's take a sneak peak at correlations between features (see Figure 1). Overall, we observe that multi-collinearity exists between the features. For example, water and energy content are highly negatively correlated (-0.84), protein and nitrogen content are perfectly positively correlated (1.0), phosphorus content is highly correlated to protein and thus nitrogen content (0.68).

Multi-collinearity indicates that a PCA - and features reduction methods - will make perfect sense afterwards.

2.2.2 Clustering methods

We begin our exploratory journey by investigating the clustering patterns in the data set. Typical clustering methods, such as k-means, require the specification of an optimal number of clusters. It is usually a step that requires cross validation to find the optimal number of clusters minimizing the sum distance within the centroids (i.e. wss). Even if we already have a number of clusters i.e. the number of food categories, we will perform a cross validation to confirm that the optimal number of clusters is similar to the number of food categories. Testing for range of clusters from 20 to 25 returned an optimal clusters number of 24. It indicates that the data are indeed suited to be clustered into approximately 22 clusters, that is the number of food categories in the data set. We then apply k-means clustering to the data with 22 centers and plot the result in 2D with the help of a background PCA (please refer to next section for furhter details about PCA).

Figure 2 illustrates the output of the k-means clustering. Due to the load of data, the figure is not that much interpretable. A lot of clusters overlap. We can have a better understanding of the clustering patterns

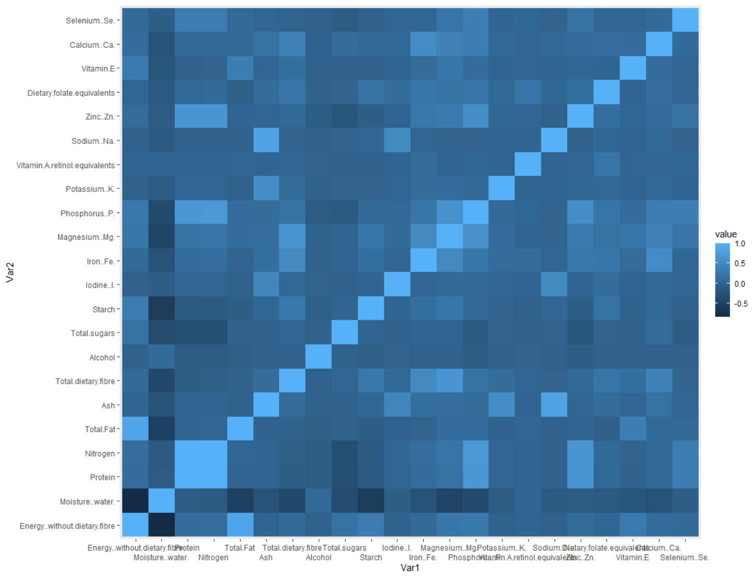


Figure 1: Pearson correlation coefficients heatmap of the features

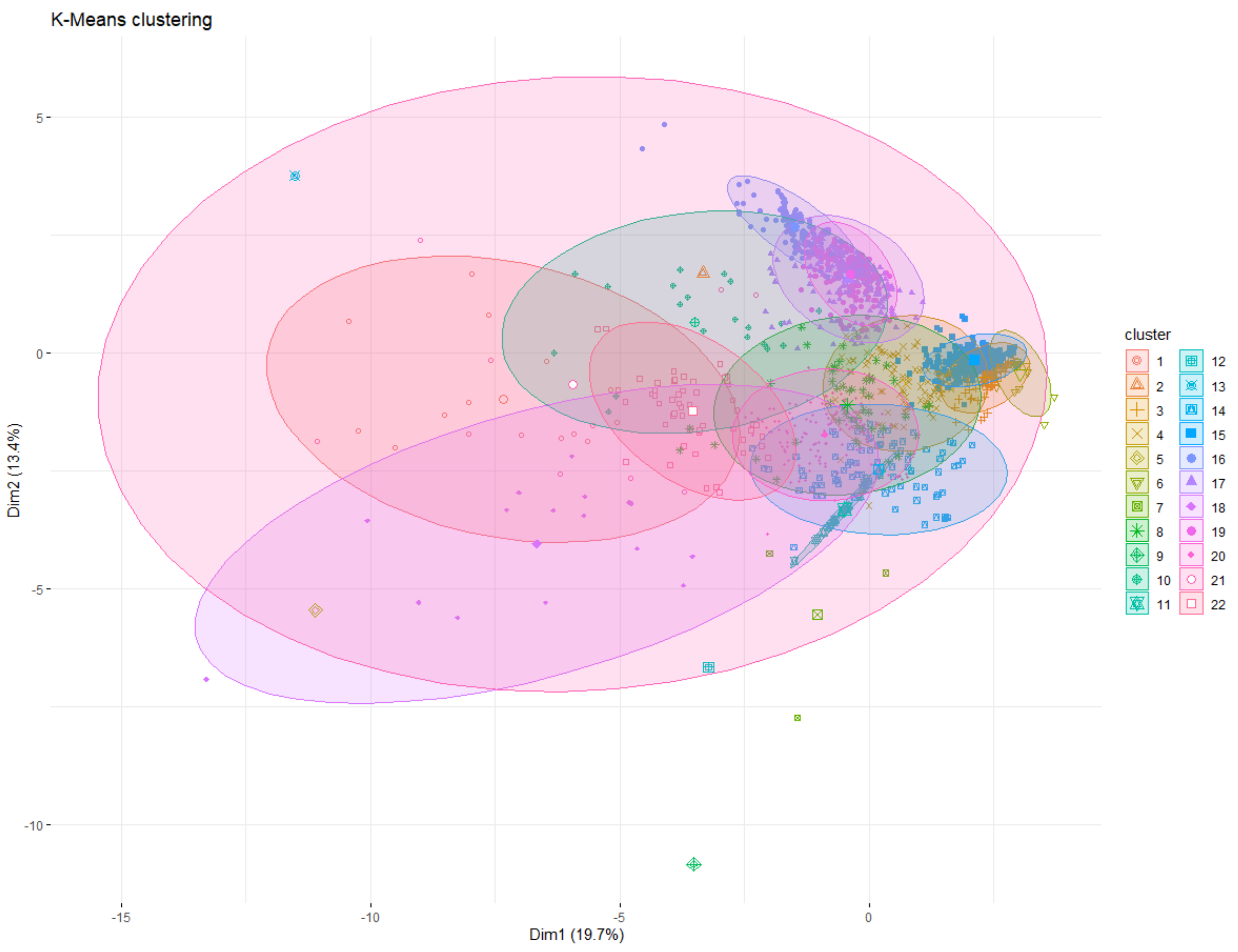
by looking at the table (b), that is a mapping of the food category of each item to its/their corresponding cluster(s). To grasp the table, the conversion between the food category number and name is given by the table (a). Non-alcoholic beverages, fats and oils, seafood products, fruit products, dairy and meat substitutes, vegetables, snack foods, sugar products and confectionery have neat clustering patterns while the others have mixed clustering patterns. For example, cereals and cereal products and cereal-based products obviously share 3 main clusters. Another example is the cluster number 5 that contains food items from non-alcoholic beverages, fruits, milk and vegetables products. The latter relationship between these four food category is instinctive as non-alcoholic beverages are typically juices or dairy product.

Globally, we observe that the data did cluster into food categories, with more or less success depending on the category. From this k-means clustering preliminary analysis, we can conclude that non-random patterns emanate from the data and merit further investigations.

2.2.3 Principal Components Analysis

To continue our exploratory journey in the Australian Food Composition Database, we performed a PCA. To visualize the outcome of the PCA, a biplot was produced (see Figure 3). The PC1 was plotted againsted the PC2 and the vectors of the variables are displayed. Firstly, as the previously performed k-mean clustering used a background PCA, we retrieve the same grouping patterns. Secondly, the variable vectors overlay the observations. The variable vectors seem to gather into two main direction. A first direction, pointing to north-west and south-east, includes Protein, Nitrogen, Zinc, Selenium, Phosphorus and total sugars. A second direction, pointing to south-west and north-east, includes Magnesium, dietary folate equivalents, Calcium, Iron, Potassium, Iodine, moisture, starch and Sodium. PCA is particularly impressive when it comes to visualize linear relationships between variables. It is likely that with 22 variables, non-linear relationships exist. If the data were to showcase such non-linear relationships, PCA may not reveal it. To investigate that, a method called t-SNE is tested in the next section.

Since I was curious about investigating a new method that we did not cover during the classes, another visualization technique for high-dimensional data was applied, namely t-Distributed Stochastic Neighbor Embedding. We still focus on the exploration and try to confirm that our data may contain interesting patterns. When looking at the right plot of Figure 3, we remark that t-SNE does quite a impressive job at visualizing the observations in 2D. The majorly green cluster contains the food category 15, 17 and 18. They represent, respectively, the fish and see food products, egg products and meat products. The majorly blue cluster is made of observations classified as food category 23, 24 and 25. They represent, respectively, savoury sauces and condiments, vegetables, legume and pulse products. Interestingly, we see a greenish cluster nearby the blue grouping, that is the fruit products (food category 16). A third and last major cluster that can be observed is the orange one. It majorly contains food category 12, 13 and 14 which corresponds, respectively, to cereal, cereal-based products, fats and oil. Interestingly, t-SNE distance-based similarity measure reveals that observations from the aforementioned groups are close to the ones of food category 22 and 28, that is seed and



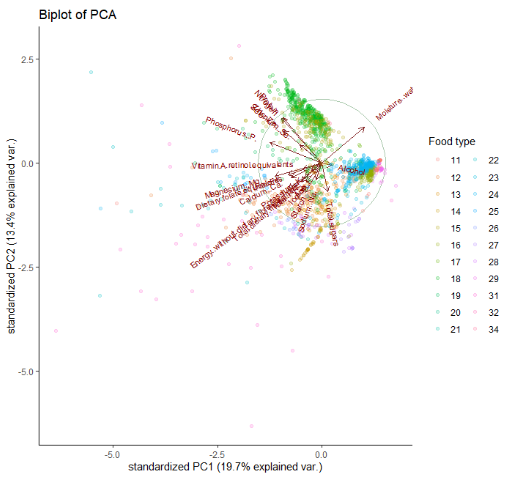
Food number	Food category
11	Non-alcoholic beverages
12	Cereals and cereal products
13	Cereal based products and dishes
14	Fats and oils
15	Fish and seafood products and dishes
16	Fruit products and dishes
17	Egg products and dishes
18	Meat, poultry and game products and dishes
19	Milk products and dishes
20	Dairy and meat substitutes
21	Soup
22	Seed and nut products and dishes
23	Savoury sauces and condiments
24	Vegetable products and dishes
25	Legume and pulse products and dishes
26	Snack foods
27	Sugar products and dishes
28	Confectionery and cereal/nut/fruit/seed bars
29	Alcoholic beverages
30	Special dietary foods
31	Miscellaneous
32	Infant formulae and foods
33	Dietary supplements
34	Reptiles, amphibia and insects

(a) Classification of the food categories

	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	31	32	34
1	1	3	0	0	0	0	0	0	0	0	0	10	0	0	1	0	0	0	0	1	0	0
2	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	17	0	0	0	0	68	0	0	10	0	0	0	8	0	0	0	3	0	0	1	1	0
4	0	24	30	0	4	0	1	3	4	2	0	3	8	24	6	0	0	0	0	2	0	0
5	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	19	1	0	0
7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	3	0	0
8	0	0	1	4	0	0	0	36	3	0	0	6	2	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
10	0	0	0	0	6	0	0	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0
11	0	0	0	29	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
13	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
14	2	0	26	0	0	13	0	0	1	0	0	0	3	0	0	0	16	24	0	0	0	0
15	24	6	2	0	0	43	2	4	9	9	1	5	13	196	5	0	0	0	8	0	0	0
16	0	1	0	0	5	0	0	123	0	0	0	0	0	0	0	0	0	0	0	1	0	0
17	0	0	0	0	99	0	8	71	4	0	0	0	0	0	0	0	0	0	0	0	0	1
18	1	0	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0	0	0	13	0	0
19	0	0	0	0	1	0	0	209	0	3	0	0	0	0	0	0	0	0	0	0	0	2
20	0	93	32	0	0	0	0	0	0	0	5	0	1	4	0	9	0	4	0	1	0	0
21	0	2	0	0	0	0	0	2	0	0	0	0	0	1	0	0	0	0	0	3	0	0
22	2	21	0	0	0	0	0	0	0	0	0	8	0	5	13	1	0	0	0	4	0	0

(b) Cluster assignment table: x entries: food categories and y entries: clusters

Figure 2: k-means clustering - visualization using a background PCA


(a) PCA

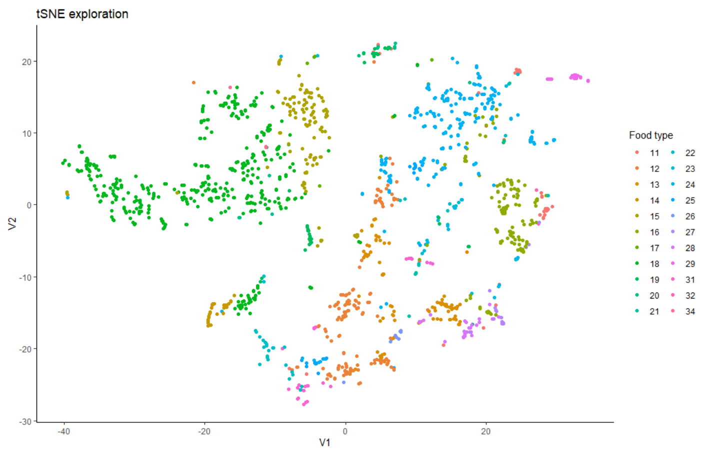

(b) t-SNE

Figure 3: High-dimensional visualization methods: PCA and t-SNE exploration

nut products, confectionery and cereal/nut/fruit/seed bars.

To conclude, t-SNE was able to discriminate three main classes within the 22 food categories, namely meat and see-food products, vegetables and fruit and cereal-based and nutty products.

2.2.4 t-SNE visualization

2.3 Tree-based modelling

Since our exploratory journey comes to an end, it is time to train tree-based models on the nutrients composition data. The modelling methodology is split in three parts: single tree-based modelling, pruning and ensemble method. The data were randomly (seed = 3) split following a 70/30 distribution for train and test set.

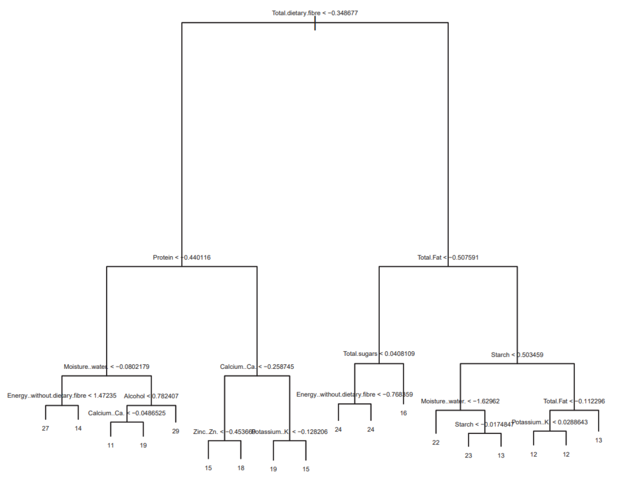
2.3.1 Single-tree based modelling and pruning

We begin by applying a single decision tree model on the data. The outcome is illustrated on Figure 4. That first model yields a train accuracy of 75,61% and a test accuracy of 75.22%. The power of tree is their ability to learn understandable decision rules. We observe that the first split to be made is based on the dietary fiber content. Overall, food containing a high fiber content are cereal-based products, fruits and vegetables products while food containing a low fiber content are meat, see-food, diary products, beverages (alcoholic or not), fats and oil. Furthermore, we can draw some interesting observation:

1. Meat, see-food and diary products tend to have a high-protein content
2. Meat and see-food can be discriminated based on their zinc content
3. Diary products tend to contain more calcium to other non-alcoholic beverages
4. Total fat content discriminates cereal-based and seed products from fruit and vegetables products
5. Fruits products tend to have a higher total sugar content than vegetables products

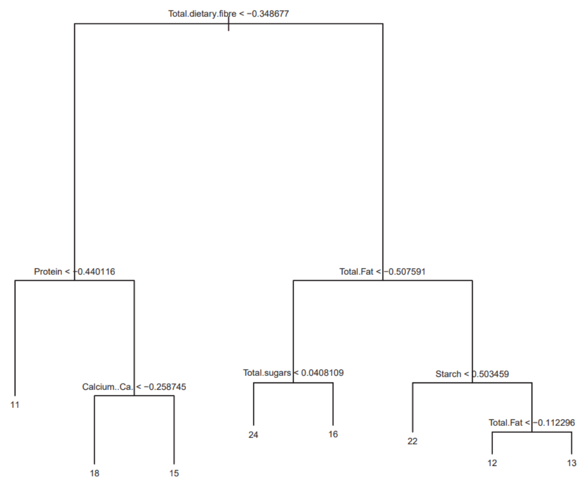
The aforementioned tree contains some decision rule that provide little power to classify instances e.g. Energy..without.dietary.fibre < 0.768359 simply splits a node into similar classes, thus does not bring any additional significant information. To reduce the potential over-fitting of the model, we will prune the branches that are uncritical and redundant to classify instances. Before pruning a tree, we need to decide an optimal value for the depth of the tree using cross-validation. According to its RDocumentation, the cv.tree function determines a nested sequence of subtrees of the supplied tree by recursively "snipping" off the least important splits. A depth of 7 seems a good trade-off. The pruning of the single tree yielded a simplified tree illustrated in Figure 4 (right tree). The cost of this simplification in interpretability is a decrease in predictive accuracy. Indeed, the pruned tree yields a training accuracy of 63.5% and a test accuracy of 65%.

Despite being extremely useful for learning decision rules, single tree-based model tend to not perform very well in term of prediction. The next section is dedicated to ensemble methods, that combine the performance of several weak models into a single strong model.



Total.dietary.fibre < −0.348677
Protein < −0.440116
Total.Fat < −0.507591
Moisture..water. < −0.0802179
Calcium..Ca. < −0.258745
Total.sugars < 0.0408109
Starch < 0.503459
Energy..without.dietary.fibre < 1.47235
Alcohol < 0.782407
Calcium..Ca. < −0.0486525
Zinc..Zn. < −0.453669
Potassium..K. < −0.128206
Energy..without.dietary.fibre < −0.768359
Moisture..water. < −1.62962
Total.Fat < −0.112296
Starch < −0.0174847
Potassium..K. < 0.0288643
27
14
11
19
29
15
18
19
15
24
24
16
22
23
13
12
12
13

(a) Single tree model



(b) Single pruned tree model

Figure 4: Tree-based modelling

2.3.2 Ensemble method: Random Forest Classifier

As explained in the aforementioned section, based on the 'stronger together' concept, we can combine the power of several weak models into a performing one. Using a Random Forest Classifier, we were able to drastically improve the predictive accuracy. The training accuracy is 100% and the test accuracy is 94%.

Aside of being a powerful classifier, random forest enables to perform an importance assessment of variables. The five most important variables according to their mean decrease accuracy are starch, iron, total fat, iodine and calcium. The idea of the latter metric is to directly measure the impact of each feature on accuracy of the model by permuting the values of each feature and measuring how much the permutation decreases the accuracy of the model. It paves the way for variables selection.

3 Conclusion

Throughout the exploration of the Australian Food Composition Initiative, we discovered interesting underlying patterns in the nutrients composition of common food products. After a typical ETL pre-processing step, we started an exploratory analysis with a correlation exploration, a k-mean clustering analysis and two multi-dimensional visualization methods, that is PCA and t-SNE method. That exploratory analysis revealed non-random underlying patterns in the data, showing evidence that classification might apply. Therefore, the data undergone tree-based classification algorithm such as single tree modelling with and without pruning and RandomForestClassifier ensemble method. The single pruned tree model enabled us to grasp interesting decision rules describing the underlying non-random patterns while the ensemble method showed an impressive classification performance with an independent accuracy of 94%.

4 Improvements and follow-up

The project could naturally be enhanced and completed. Indeed, no outliers detection nor removal of almost empty classes was performed. Additionally, a factor analysis seems promising in order to investigate the combination of some variables (e.g. Protein and Nitrogen content highly correlated).

Via the present project, an idea popped in my mind that I plan to deepen, that is the development of a convolutional neural network (image processing) pipeline to calculate a healthy score of common food based on their nutritive labels (i.e. per 100g). It would help the consumer to make a healthy decision for any type of food, even those non NutriScore labelled.

5 R code

```r
#By Antoine Ruzette - r0829308
#Applied Multivariate Statistical Anaysis [I0P16a]
#MSc in Bioinformatics, KU Leuven
#Academic year 2021-2022

#Use of data from the Australian Food Composition Database
#https://www.foodstandards.gov.au/science/monitoringnutrients/afcd/Pages/default.aspx
#to investigate the type of food and its content in nutrients

#load packages
library(naniar)
library(readxl)
library(dplyr)
library(cluster)
library(factoextra)
library(stringr)
library(tree)
library(ISLR)
library(dplyr)
library(ggplot2)
library(NbClust)
library(miscFuncs)
library(readr)
library(reshape2)
library(devtools)
#install_github("vqv/ggbiplot")
library('ggbiplot')
library(Rtsne)
library(randomForest)
library(rpart)

#set your specific environment
setwd("C:/Users/antoi/OneDrive/Documentos/Bioinformatics/M2/M2_Q1/AMSA/")

##1. Extraction, Transformation and Loading
#load the data from excel
nutrient.rawdata = read_excel("Antoine_Ruzette.xlsx",
                              sheet = 'All␣solids␣&␣liquids␣per␣100g')
rawdata = as.data.frame(nutrient.rawdata)
nrow(rawdata)#1535 food items
ncol(rawdata)#252 nutrients information

#load the food categories (keys)
foodCategories = read_delim("FoodCategoriesConversion.txt",
                            delim = "\t", escape_double = FALSE,
                            col_names = FALSE)
#remove all columns have more than 90% missing observations
missing_col = sapply(rawdata, function(x) sum(is.na(x))/1535)
missing_col = as.data.frame(missing_col)
colnames(missing_col)[1] = 'MissingPercentage'
#subset only the column having less than 10% missing data
nonmissing_col = subset(missing_col, missing_col[, 1] < 0.1)
nonmissing_col['colName'] = row.names(nonmissing_col)
nrow(nonmissing_col)
#63 on 252 columns that have less than 10% missing

#remove the highly missing variables
data_sub = rawdata[, nonmissing_col$colName]
```

```
#subsetting the variable space to the variables that we are interested in
all.names = c('Public␣Food␣Key', 'Classification', 'Food␣Name',
              'Energy,␣without␣dietary␣fibre', 'Moisture␣(water)',
              'Protein', 'Nitrogen', 'Total␣Fat', 'Ash', 'Total␣dietary␣fibre',
               'Alcohol', 'Total␣sugars', 'Starch', 'Iodine␣(I)', 'Iron␣(Fe)',
               'Magnesium␣(Mg)', 'Phosphorus␣(P)', 'Potassium␣(K)',
               'Vitamin␣A␣retinol␣equivalents', 'Sodium␣(Na)', 'Zinc␣(Zn)',
               'Dietary␣folate␣equivalents', 'Vitamin␣E', 'Calcium␣(Ca)',
               'Selenium␣(Se)')
features.names = c('Energy,␣without␣dietary␣fibre', 'Moisture␣(water)', 'Protein',
                   'Nitrogen', 'Total␣Fat', 'Ash', 'Total␣dietary␣fibre',
                   'Alcohol', 'Total␣sugars', 'Starch', 'Iodine␣(I)', 'Iron␣(Fe)',
                   'Magnesium␣(Mg)', 'Phosphorus␣(P)', 'Potassium␣(K)',
                   'Vitamin␣A␣retinol␣equivalents', 'Sodium␣(Na)', 'Zinc␣(Zn)',
                   'Dietary␣folate␣equivalents', 'Vitamin␣E', 'Calcium␣(Ca)',
                   'Selenium␣(Se)')
target.name = c('Classification')
data = data_sub[all.names]

#removing remaining missing data, if there still exist some
data = na.omit(data)
sum(is.na(data))#0 missing values
nrow(data)#1534 rows remaining
ncol(data)

data.x = data
X = select(data.x, - c("Public␣Food␣Key", "Classification", "Food␣Name"))
X = lapply(X,  as.numeric)
X = as.data.frame(X)

###2. Exploratory Analysis (correlations, k-means clustering, PCA and tSNE)
##correlation between variables
cormat = round(cor(X), 2)
cormat
melted_cormat <- melt(cormat)
#plot heatmap
ggplot(data = melted_cormat, aes(x=Var1, y=Var2, fill=value)) +
  geom_tile() +
  scale_x_discrete(guide = guide_axis(n.dodge=2))


#scaling data
X = as.data.frame(X)
X.scaled = scale(as.data.frame(X))
X.scaled = as.data.frame(X.scaled)

data.scaled = cbind(data$'Public␣Food␣Key', data$'Classification',
                    data$'Food␣Name', X.scaled)
data = rename(data.scaled, replace = c(`data$"Public␣Food␣Key"` = "FoodKey",
                                        `data$Classification` = 'Classification',
                                        `data$"Food␣Name"` = 'FoodName'))
#create the 22 food categories
data$Classification = str_extract(as.character(data$Classification), "\\d{2}")
data$Classification = as.factor(data$Classification)
length(unique(data$Classification))#22 food categories

#distribution of the observations among classes
distr = aggregate(FoodKey ~ Classification,
          data = data,
          FUN = length)

##Principal Components Analysis
nutrient.pca <- prcomp(data[, c(4:25)], center = TRUE, scale = FALSE)
```

```r
summary(nutrient.pca)#2 PCs explain 33% of the variance, 13 PCs explain 90%

#plot PC1 vs PC2
ggbiplot(nutrient.pca, ellipse = FALSE , groups = data$Classification,
         var.axes = TRUE, varname.size = 3, alpha = 0.2, circle = TRUE) +
  theme_classic() +
  labs(color="Food␣type") +
  ggtitle('Biplot␣of␣PCA')

##tSNE
colors = rainbow(length(unique(data$Classification)))
names(colors) = unique(data$Classification)

tsne = Rtsne(data[, c(4:25)], check_duplicates = FALSE, dims = 2, perplexity=30,
                verbose=TRUE, max_iter = 500)
tsne.Y = as.data.frame(tsne$Y)
#plot t-SNE1 vs t-SNE2
ggplot(tsne.Y, aes(V1, V2, colour = data$Classification)) +
  geom_point() +
  theme_classic() +
  labs(color = 'Food␣type') +
  ggtitle('tSNE␣exploration')

##k-means clustering

#explore the data scatterplot
set.seed(2)
best_clust = NbClust(data = data[, c(4:25)], distance = "euclidean",
                        min.nc = 20, max.nc = 25, method = 'kmeans')
fviz_nbclust(best_clust, ggtheme = theme_minimal())
km = kmeans(data[, c(4:25)], centers = 22, nstart = 25, iter.max = 1000)
#plot clusters (with ellipse) over the data scatterplot
fviz_cluster(object=km, data = data[, c(4:25)], ellipse.type = 'norm',
                geom = 'point', main = 'K-Means␣clustering', ggtheme = theme_minimal())

#table to compare the clustering patterns and the real food categories
data$km_cluster = as.numeric(km$cluster)
assignment_table = table(data$km_cluster, data$Classification)
latextable(foodCategories)
aggregate(cbind(Protein, Nitrogen, Ash, Alcohol) ~ km_cluster, data = data, FUN=mean)

features.names = c("Classification", "Energy..without.dietary.fibre",
                    "Moisture..water.", "Protein", "Nitrogen", "Total.Fat",
                    "Ash", "Total.dietary.fibre", "Alcohol", "Total.sugars",
                    "Starch", "Iodine..I.", "Iron..Fe.", "Magnesium..Mg.",
                    "Phosphorus..P.", "Potassium..K.",
                    "Vitamin.A.retinol.equivalents", "Sodium..Na.",
                    "Zinc..Zn.", "Dietary.folate.equivalents",
                    "Vitamin.E", "Calcium..Ca.", "Selenium..Se.")

##3. Tree based Modeling
#split in train and test sets (approx. 75/25 split)
set.seed(3)
train = data %>% sample_n(1200)
X_train = train[features.names]
y_train = train[target.name]

test = data %>% setdiff(train)
X_test = test[features.names]
y_test = test[target.name]

#SINGLE TREE
tree_nutrient = tree(Classification ~ . , X_train)
```

```
summary(tree_nutrient)

single_pred = predict(tree_nutrient, X_test, type = "class")
mean(single_pred == X_test$Classification)
#train accuracy = 1 - 0.2439 = 0.7561 = 75.6%
#number of terminal nodes = 20 (compared to 22 food categories in the beginning)

#export to pdf file
pdf("SingleTree.pdf", width = 8, height = 7,
    bg = "white", colormodel = "cmyk",
    paper = "A4")
plot(tree_nutrient)
text(tree_nutrient, pretty = 2, cex = 0.45)
dev.off()

#test accuracy
tree_pred = predict(tree_nutrient, X_test, type = "class")
mean(tree_pred == X_test$Classification)
#75.4% of test accuracy

#SINGLE PRUNED TREE
set.seed(4)
cv_nutrient = cv.tree(tree_nutrient, FUN = prune.misclass)
plot(cv_nutrient$size, cv_nutrient$dev, type = "b")
prune_nutrient = prune.misclass(tree_nutrient, best = 7)
summary(prune_nutrient)
#train accuracy = 0.635
#reduced accuracy for the stake of interpretability

#export to pdf file
pdf("PrunedTree.pdf", width = 8, height = 7,
    bg = "white", colormodel = "cmyk",
    paper = "A4")
plot(prune_nutrient)
text(prune_nutrient, pretty = 0, cex = 0.6)
dev.off()

prune_pred = predict(prune_nutrient, X_test, type = "class")
mean(prune_pred == X_test$Classification)
#65% of test accuracy

#RANDOM FOREST
set.seed(5)
rf_nutrient = randomForest(Classification~.,
                           data = X_train,
                           importance = TRUE)
#print summary
rf_nutrient
#OOB accuracy = 90.75%

rf_estimate = predict(rf_nutrient, newdata = X_test)
mean(rf_estimate == X_test$Classification)
#test accuracy = 94%

ggplot() +
  geom_point(aes(x = X_test$Classification, y = rf_estimate)) +
  geom_abline()

#calculate importance of the used variables
importance(rf_nutrient)
#plot importance measure of the used variables: mean decrease accuracy and decrease gini
varImpPlot(rf_nutrient)
```