hw4Clustering

2024-11-03

HW 4 - DSC 441

Problem 1:

For this problem, you will tune and apply kNN and compare it to other classifiers. We will use the wine quality data, which has a number of measurements about chemical components in wine, plus a quality rating. There are separate files for red and white wines, so the first step is some data preparation.

- a: Load the two provided wine quality data sets and prepare them by (1) ensuring that all the variables have the right type (e.g., what is numeric vs. factor), (2) adding a type column to each that indicates if it is red or white wine and (2) merging the two tables together into one table (hint: try full_join()). You now have one table that contains the data on red and white wine, with a column that tells if the wine was from the red or white set (the type column you made).

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(caret)
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```r
# load two datasets
red_wine <- read_csv2("/Users/danielkim/Downloads/winequality-red.csv")
```

```
## i Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.

## Warning: One or more parsing issues, call `problems()` on your data frame for details,
## e.g.:
##   dat <- vroom(...)
##   problems(dat)

## Rows: 1599 Columns: 12
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (5): volatile acidity, citric acid, chlorides, density, sulphates
## dbl (2): total sulfur dioxide, quality
## num (5): fixed acidity, residual sugar, free sulfur dioxide, pH, alcohol
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
white_wine <- read_csv2("/Users/danielkim/Downloads/winequality-white.csv")
```

```
## i Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
## Rows: 4898 Columns: 12-- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (6): volatile acidity, citric acid, residual sugar, chlorides, density, ...
## dbl (1): quality
## num (5): fixed acidity, free sulfur dioxide, total sulfur dioxide, pH, alcohol
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
# type column added to both data sets (red = 1, white = 0)
red_wine$type <- as.factor(1)
white_wine$type <- as.factor(0)

# convert data type for residual sugar in white wine table to data type for residual sugar in red wine
white_wine <- white_wine %>% mutate(`residual sugar` = as.double(`residual sugar`))
head(red_wine)
```

```
## # A tibble: 6 x 13
```

```
##   `fixed acidity` `volatile acidity` `citric acid` `residual sugar` chlorides
##             <dbl> <chr>              <chr>                     <dbl> <chr>
## 1              74 0.7                0                            19 0.076
## 2              78 0.88               0                            26 0.098
## 3              78 0.76               0.04                         23 0.092
## 4             112 0.28               0.56                         19 0.075
## 5              74 0.7                0                            19 0.076
## 6              74 0.66               0                            18 0.075
## # i 8 more variables: `free sulfur dioxide` <dbl>,
## #   `total sulfur dioxide` <dbl>, density <chr>, pH <dbl>, sulphates <chr>,
## #   alcohol <dbl>, quality <dbl>, type <fct>
```

```
head(white_wine)
```

```
## # A tibble: 6 x 13
##   `fixed acidity` `volatile acidity` `citric acid` `residual sugar` chlorides
##             <dbl> <chr>              <chr>                     <dbl> <chr>
## 1               7 0.27               0.36                       20.7 0.045
## 2              63 0.3                0.34                        1.6 0.049
## 3              81 0.28               0.4                         6.9 0.05
## 4              72 0.23               0.32                        8.5 0.058
## 5              72 0.23               0.32                        8.5 0.058
## 6              81 0.28               0.4                         6.9 0.05
## # i 8 more variables: `free sulfur dioxide` <dbl>,
## #   `total sulfur dioxide` <dbl>, density <chr>, pH <dbl>, sulphates <chr>,
## #   alcohol <dbl>, quality <dbl>, type <fct>
```

```
# merge the two tables together
wine <- full_join(red_wine, white_wine)
```

```
## Joining with `by = join_by(`fixed acidity`, `volatile acidity`, `citric acid`,
## `residual sugar`, chlorides, `free sulfur dioxide`, `total sulfur dioxide`,
## density, pH, sulphates, alcohol, quality, type)`
```

```
summary(wine) # NA row in total sulfur dioxide column
```

```
##  fixed acidity    volatile acidity   citric acid        residual sugar
##  Min.   :  5.00   Length:6497        Length:6497        Min.   :  0.60
##  1st Qu.: 62.00   Class :character   Class :character   1st Qu.:  2.00
##  Median : 68.00   Mode  :character   Mode  :character   Median :  7.40
##  Mean   : 65.51                                         Mean   : 11.39
##  3rd Qu.: 76.00                                         3rd Qu.: 15.25
##  Max.   :715.00                                         Max.   :655.00
##
##   chlorides         free sulfur dioxide total sulfur dioxide   density
##  Length:6497        Min.   :   1.00     Min.   :   6.0       Length:6497
##  Class :character   1st Qu.:  17.00     1st Qu.:  77.0       Class :character
##  Mode  :character   Median :  29.00     Median : 118.0       Mode  :character
##                     Mean   :  34.99     Mean   : 123.5
##                     3rd Qu.:  41.00     3rd Qu.: 156.0
##                     Max.   :1465.00     Max.   :3665.0
##                                         NA's   :2
```

```
##        pH         sulphates            alcohol             quality
##  Min.   :  3.0   Length:6497        Min.   :8.000e+00   Min.   :3.000
##  1st Qu.:306.0   Class :character   1st Qu.:9.300e+01   1st Qu.:5.000
##  Median :318.0   Mode  :character   Median :1.010e+02   Median :6.000
##  Mean   :289.5                      Mean   :1.733e+12   Mean   :5.818
##  3rd Qu.:331.0                      3rd Qu.:1.120e+02   3rd Qu.:6.000
##  Max.   :401.0                      Max.   :9.733e+14   Max.   :9.000
##
##  type
##  1:1599
##  0:4898
##
##
##
##
##
```

```
sum(is.na(wine)) # 2 rows with NA column in entire data set
```

```
## [1] 2
```

```
na_rows <- wine[is.na(wine$`total sulfur dioxide`),] # select NA rows
na_rows
```

```
## # A tibble: 2 x 13
##   `fixed acidity` `volatile acidity` `citric acid` `residual sugar` chlorides
##             <dbl> <chr>              <chr>                    <dbl> <chr>
## 1              66 0.63               0                           43 0.093
## 2              66 0.63               0                           43 0.093
## # i 8 more variables: `free sulfur dioxide` <dbl>,
## #   `total sulfur dioxide` <dbl>, density <chr>, pH <dbl>, sulphates <chr>,
## #   alcohol <dbl>, quality <dbl>, type <fct>
```

```
# drop NA rows
wine <- wine %>% drop_na()
dim(wine)
```

```
## [1] 6495   13
```

```
# convert remaining character columns into double type
wine <- wine %>% mutate(across(where(is.character), as.double))
summary(wine)
```

```
##  fixed acidity    volatile acidity  citric acid     residual sugar
##  Min.   :  5.00   Min.   :0.0800   Min.   :0.0000   Min.   :  0.60
##  1st Qu.: 62.00   1st Qu.:0.2300   1st Qu.:0.2500   1st Qu.:  2.00
##  Median : 68.00   Median :0.2900   Median :0.3100   Median :  7.40
##  Mean   : 65.51   Mean   :0.3396   Mean   :0.3187   Mean   : 11.38
##  3rd Qu.: 76.00   3rd Qu.:0.4000   3rd Qu.:0.3900   3rd Qu.: 15.20
##  Max.   :715.00   Max.   :1.5800   Max.   :1.6600   Max.   :655.00
##    chlorides       free sulfur dioxide total sulfur dioxide    density
```

```
##  Min.   :0.00900   Min.   :   1.00     Min.   :   6.0       Min.   :0.9871
##  1st Qu.:0.03800   1st Qu.:  17.00     1st Qu.:  77.0       1st Qu.:0.9923
##  Median :0.04700   Median :  29.00     Median : 118.0       Median :0.9949
##  Mean   :0.05602   Mean   :  34.99     Mean   : 123.5       Mean   :0.9947
##  3rd Qu.:0.06500   3rd Qu.:  41.00     3rd Qu.: 156.0       3rd Qu.:0.9970
##  Max.   :0.61100   Max.   :1465.00     Max.   :3665.0       Max.   :1.0390
##        pH          sulphates         alcohol             quality      type
##  Min.   : 3.0   Min.   :0.2200   Min.   :8.000e+00   Min.   :3.000   1:1597
##  1st Qu.:306.0   1st Qu.:0.4300   1st Qu.:9.300e+01   1st Qu.:5.000   0:4898
##  Median :318.0   Median :0.5100   Median :1.010e+02   Median :6.000
##  Mean   :289.6   Mean   :0.5313   Mean   :1.733e+12   Mean   :5.819
##  3rd Qu.:331.0   3rd Qu.:0.6000   3rd Qu.:1.120e+02   3rd Qu.:6.000
##  Max.   :401.0   Max.   :2.0000   Max.   :9.733e+14   Max.   :9.000
```

```
# column names w/o underscore or space
colnames(wine) <- make.names(colnames(wine))
```

- b: Use PCA to create a projection of the data to 2D and show a scatter plot with color showing the wine type.

```
# no near zero variance predictors
nzv <- nearZeroVar(wine)
length(nzv)
```

```
## [1] 0
```

```
# remove dependent/categorical variable
pca.wine <- wine %>% select(-c('type'))
head(pca.wine)
```

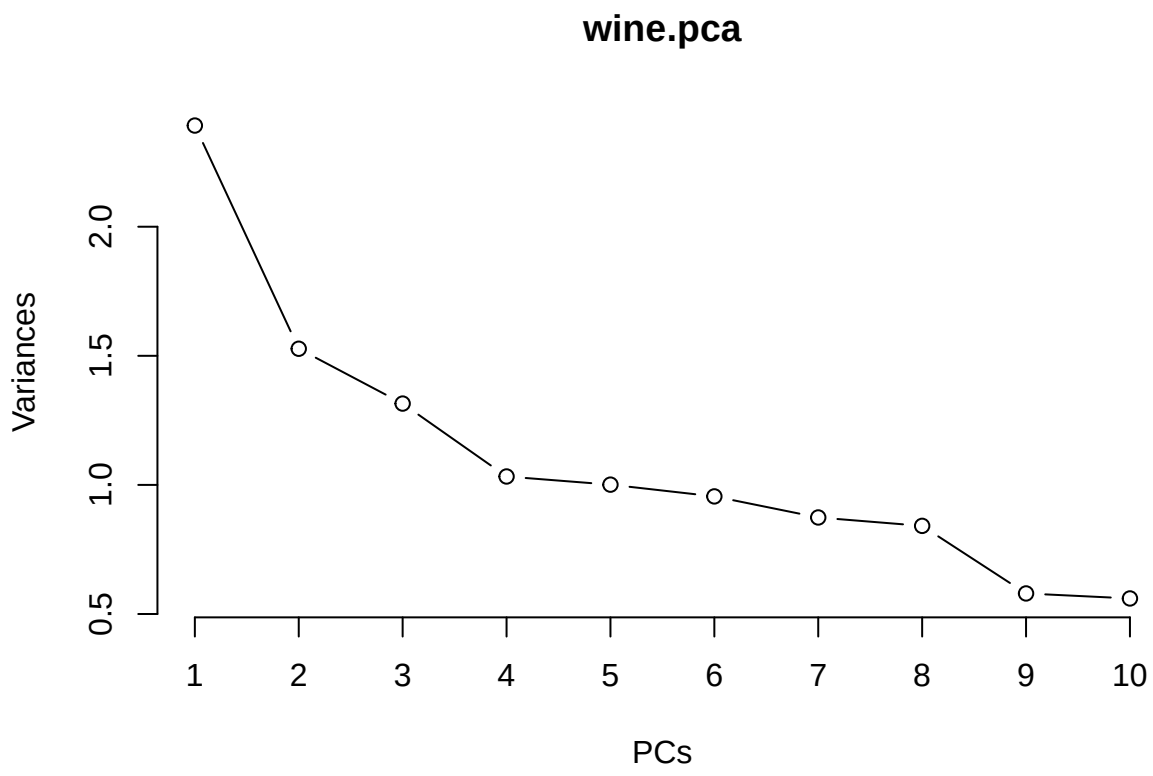
```
## # A tibble: 6 x 12
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
##           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
## 1            74             0.7         0                 19     0.076
## 2            78             0.88        0                 26     0.098
## 3            78             0.76        0.04              23     0.092
## 4           112             0.28        0.56              19     0.075
## 5            74             0.7         0                 19     0.076
## 6            74             0.66        0                 18     0.075
## # i 7 more variables: free.sulfur.dioxide <dbl>, total.sulfur.dioxide <dbl>,
## #   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>
```

```
# pca with scaled/normalized variables
wine.pca <- prcomp(pca.wine, scale. = TRUE)
summary(wine.pca)
```

```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Standard deviation     1.5465 1.2359 1.1467 1.01617 1.00049 0.97749 0.93480
## Proportion of Variance 0.1993 0.1273 0.1096 0.08605 0.08341 0.07962 0.07282
## Cumulative Proportion  0.1993 0.3266 0.4362 0.52223 0.60564 0.68526 0.75808
```

```
##                             PC8    PC9    PC10    PC11    PC12
## Standard deviation     0.91724 0.7613 0.74849 0.70998 0.64636
## Proportion of Variance 0.07011 0.0483 0.04669 0.04201 0.03482
## Cumulative Proportion  0.82819 0.8765 0.92318 0.96518 1.00000
```

```
# screeplot
screeplot(wine.pca, type = "l") + title(xlab = "PCs")
```

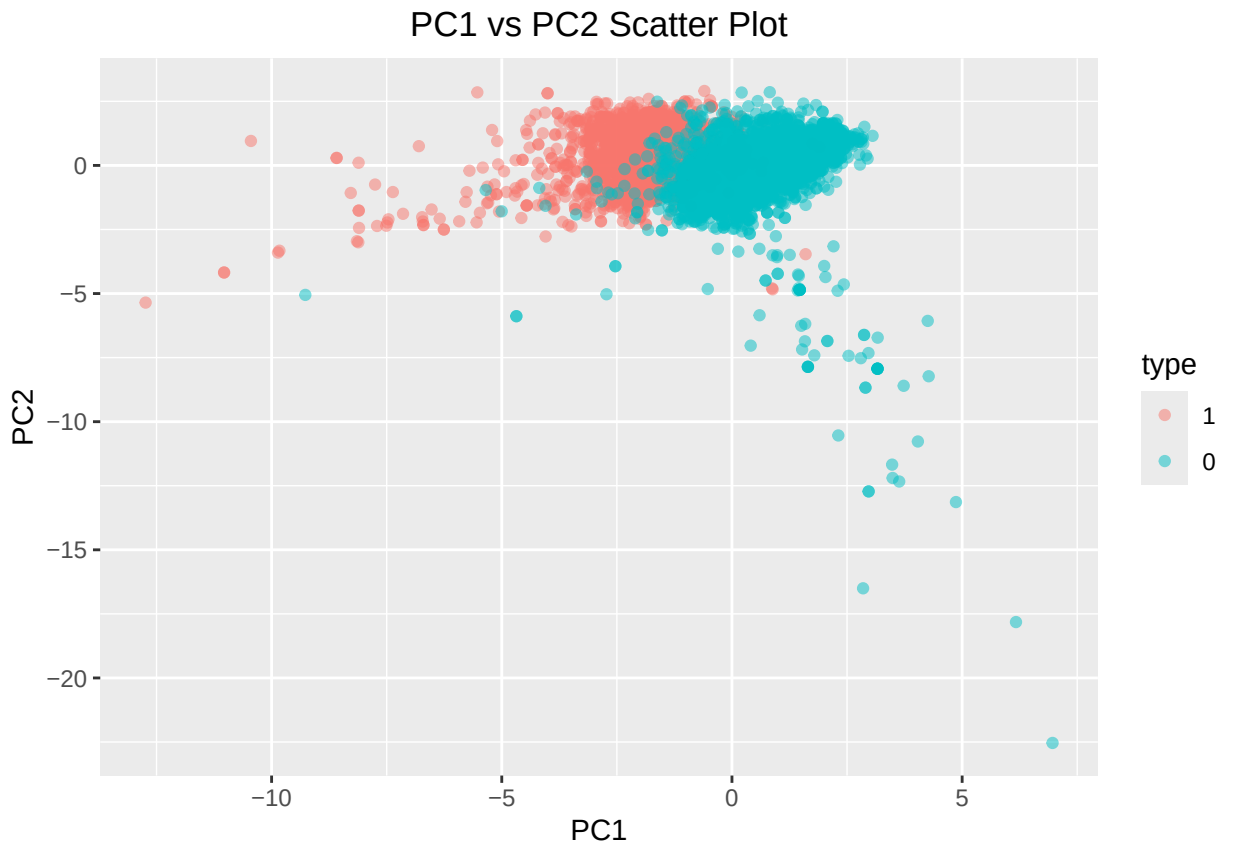



```
## integer(0)
```

```
preProc <- preProcess(pca.wine, method = 'pca', pcaComp = 4)
wine.pc <- predict(preProc, pca.wine)
wine.pc$type <- wine$type
head(wine.pc)
```

```
##         PC1       PC2       PC3        PC4 type
## 1 -2.512936  1.563726 -1.773875 -0.2397890    1
## 2 -3.439306  1.213848 -1.595881  1.6955042    1
## 3 -3.014736  1.330993 -1.531453 -0.3522434    1
## 4 -1.058775 -1.279207  1.839968  0.1715793    1
## 5 -2.512936  1.563726 -1.773875 -0.2397890    1
## 6 -2.363470  1.469454 -1.730991 -0.2308262    1
```

```
# scatter plot with PCs 1 and 2 by type (1 = red wine, 0 = white wine)
ggplot(wine.pc, aes(PC1, PC2, color = type)) + geom_point(alpha = 0.5) + labs(title = "PC1 vs PC2 Scatt
```



- c: We are going to try kNN, SVM and decision trees on this data. Based on the 'shape' of the data in the visualization from (b), which do you think will do best and why?

  Based on the 'shape' of the data presented in the scatter plot from the previous question, the kNN will perform best at classification of new data points. The issue with SVM is the granularity of data points at the potential boundaries. There would be too many support vectors along the boundary that it would be difficult to find an optimal hyperplane, even with margin leniency. Additionally, decision trees on this data would produce an overfitted model given the large dataset and non-linear relationship between the components. As a greedy algorithm, the large dataset would produce a computationally expensive model. kNN would perform best with the non-linear data, large data set, and class imbalance.

- d: Use kNN (tune k), use decision trees (basic rpart method is fine), and SVM (tune C) to predict type from the rest of the variables. Compare the accuracy values – is this what you expected? Can you explain it?

  Note: you will need to fix the columns names for rpart because it is not able to handle the underscores. This code will do the trick (assuming you called your data wine_quality): **colnames(wine_quality) <- make.names(colnames(wine_quality))**

```
library(caret)
library(e1071)

train_control <- trainControl(method = 'cv', number = 10)
```

```r
preproc <- c('center', 'scale')
grid <- expand.grid(C = 10^seq(-5, 1, 0.5))
# Fit the SVM model
svm <- train(type ~., data = wine, method = "svmLinear", trControl = train_control, preProc = preproc, 
# Evaluate fit
svm
```

```
## Support Vector Machines with Linear Kernel
##
## 6495 samples
##   12 predictor
##    2 classes: '1', '0'
##
## Pre-processing: centered (12), scaled (12)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5846, 5847, 5846, 5845, 5845, 5846, ...
## Resampling results across tuning parameters:
##
##   C             Accuracy   Kappa
##   1.000000e-05  0.7541189  0.00000000
##   3.162278e-05  0.7550436  0.00564587
##   1.000000e-04  0.7770612  0.13741799
##   3.162278e-04  0.9408804  0.82932043
##   1.000000e-03  0.9736736  0.92824170
##   3.162278e-03  0.9829100  0.95373660
##   1.000000e-02  0.9867602  0.96417957
##   3.162278e-02  0.9889157  0.97002888
##   1.000000e-01  0.9896859  0.97212088
##   3.162278e-01  0.9896861  0.97209633
##   1.000000e+00  0.9898397  0.97251871
##   3.162278e+00  0.9901479  0.97335175
##   1.000000e+01  0.9896861  0.97209470
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was C = 3.162278.
```

```r
library(caret)
library(lattice)

train_control <- trainControl(method = 'cv', number = 10)

model_tree <- train(type ~., data = wine, method = 'rpart', trControl = train_control)
model_tree
```

```
## CART
##
## 6495 samples
##   12 predictor
##    2 classes: '1', '0'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5845, 5845, 5845, 5845, 5845, 5846, ...
```

```
## Resampling results across tuning parameters:
##
##   cp          Accuracy   Kappa
##   0.0676268  0.9619716  0.8971783
##   0.1383845  0.9371798  0.8266901
##   0.7013150  0.8708347  0.5429505
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was cp = 0.0676268.
```

```
set.seed(123)
train_control <- trainControl(method="cv", number = 10)

# tune and fit the model with 10-fold cross validation,
# standardization, and our specialized tune grid
knn_fit <- train(type ~ .,
                 data = wine,
                 method = 'knn',
                 trControl = train_control,
                 preProcess = c('center', 'scale'),
                 tuneLength = 10)

# Printing trained model provides report
knn_fit
```

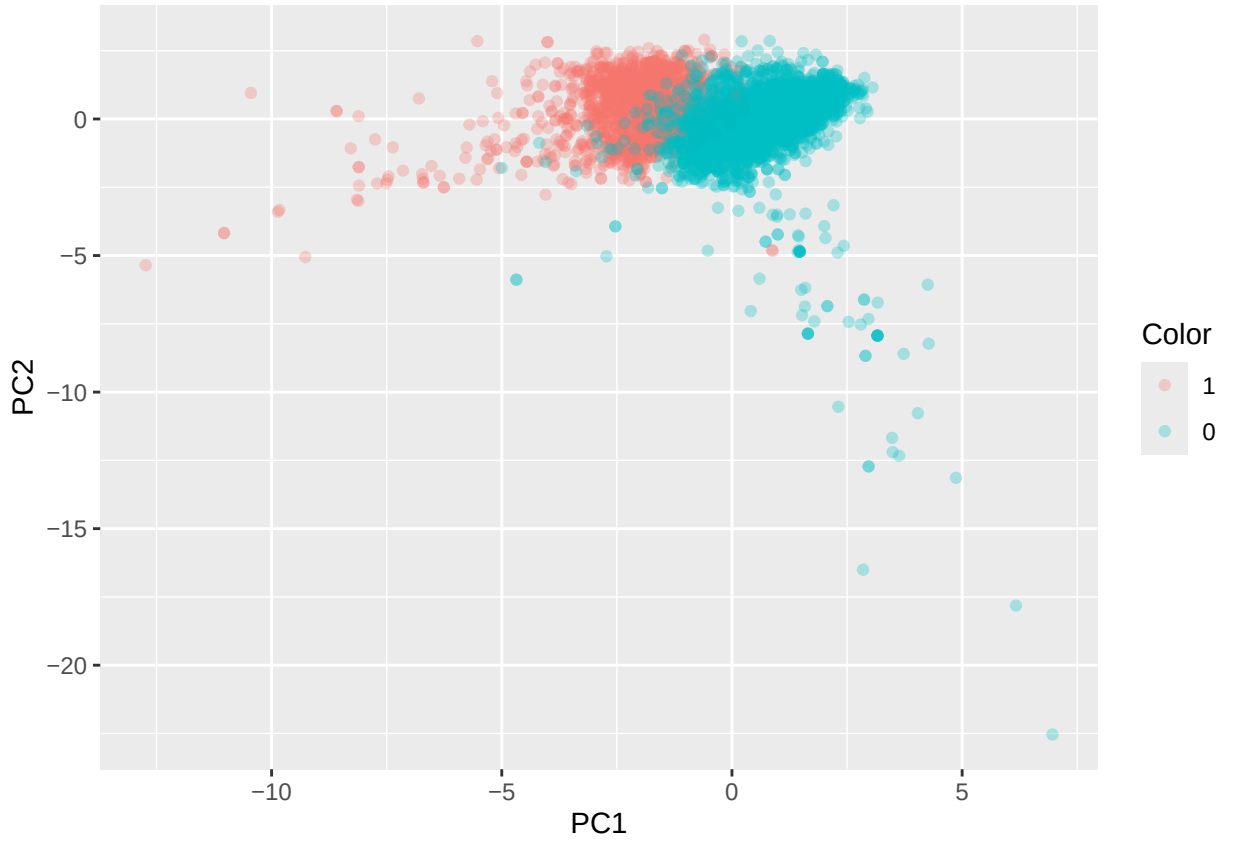
```
## k-Nearest Neighbors
##
## 6495 samples
##   12 predictor
##    2 classes: '1', '0'
##
## Pre-processing: centered (12), scaled (12)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 5846, 5845, 5845, 5845, 5845, 5846, ...
## Resampling results across tuning parameters:
##
##   k   Accuracy   Kappa
##    5  0.9869129  0.9646849
##    7  0.9859877  0.9621770
##    9  0.9847567  0.9588710
##   11  0.9846021  0.9584848
##   13  0.9836788  0.9560168
##   15  0.9827555  0.9534946
##   17  0.9829088  0.9538978
##   19  0.9819851  0.9513851
##   21  0.9812151  0.9492611
##   23  0.9807531  0.9480297
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 5.
```

SVM generated a model with C = 1 that had an accuracy of 0.9903 and a kappa of 0.9738. The decision tree model with cp = 0.0676 had an accuracy of 0.9567 and a kappa of 0.8826. Finally, kNN generated a

model with k = 5, having an accuracy of 0.9869 and a kappa of 0.9646. Given these accuracy levels, the SVM generated the most accurate model, followed closely by kNN model. This was not what was expected; however, the decision tree model was least accurate, which was expected. The accuracy of SMV tells us that there is in fact a hyperplane for the data set that can adequately classify the wine better than the kNN model can.

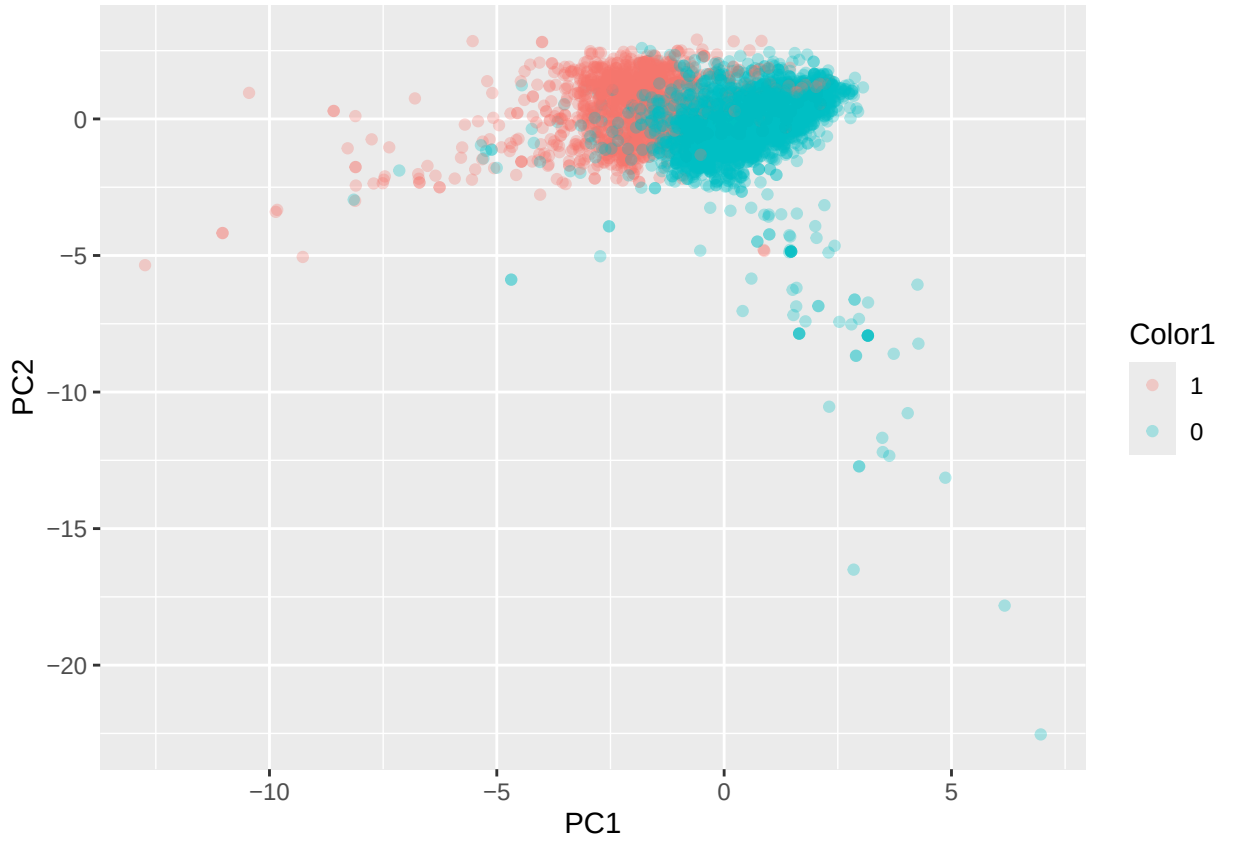
- e: Use the same already computed PCA again to show a scatter plot of the data and to visualize the labels for kNN, decision tree and SVM. Note that you do not need to recreate the PCA projection, you have already done this in 1b. Here, you just make a new visualization for each classifier using its labels for color (same points but change the color). Map the color results to the classifier, that is use the "predict" function to predict the class of your data, add it to your data frame and use it as a color. This is done for KNN in the tutorial, it should be similar for the others. Consider and explain the differences in how these classifiers performed.

```
# Save as PCs as dataframe
rotated_data = as.data.frame(wine.pca$x)

svm_predict <- predict(svm, wine)
# Add labels from model prediction as a reference
rotated_data$Color <- svm_predict
# Plot and color by labels
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = Color)) + geom_point(alpha = 0.3)
```

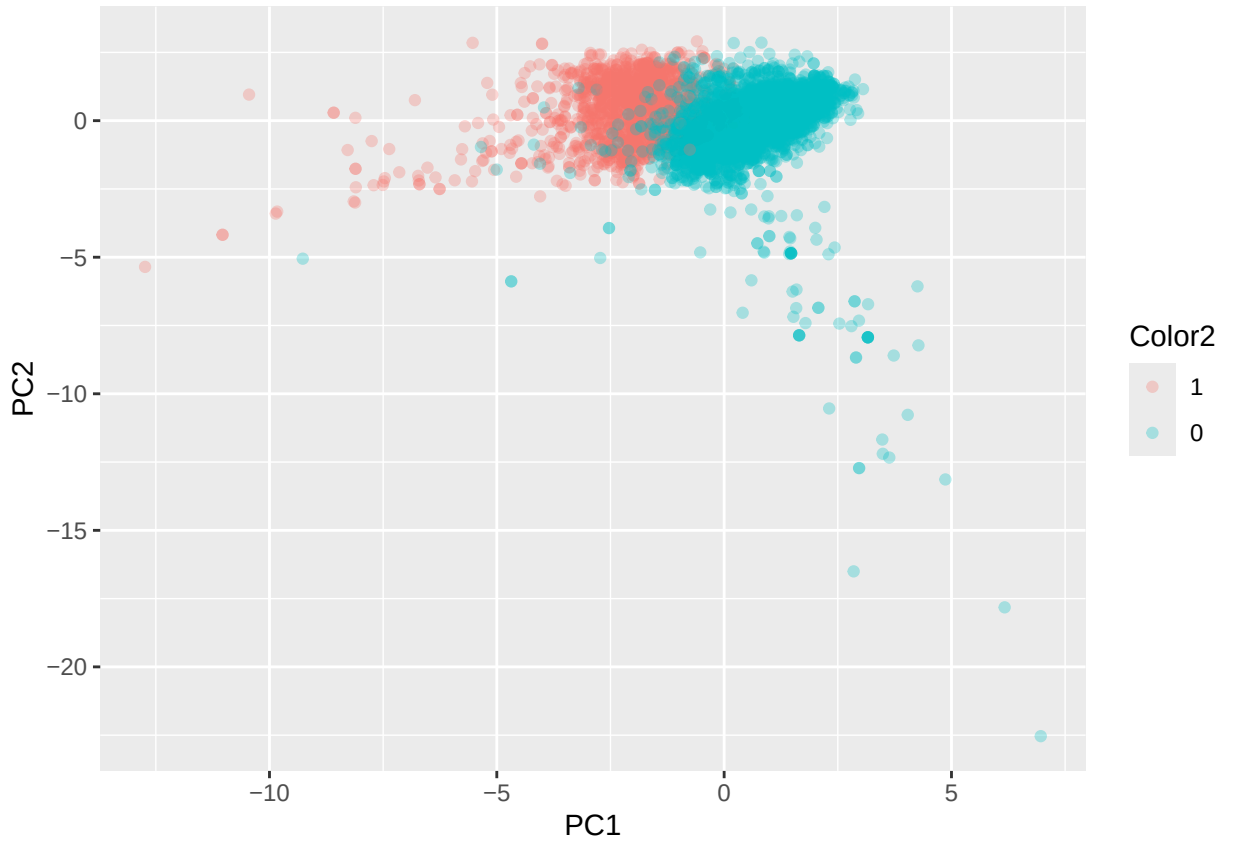



```
pred_tree <- predict(model_tree, wine)
# Add labels from model prediction as a reference
```

```
rotated_data$Color1 <- pred_tree
# Plot and color by labels
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = Color1)) + geom_point(alpha = 0.3)
```




```
knn_predict <- predict(knn_fit, wine)
# Add labels from model prediction as a reference
rotated_data$Color2 <- knn_predict
# Plot and color by labels
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = Color2)) + geom_point(alpha = 0.3)
```



As seen in the scatter plot of the data using the PCs for each classifier, we see the general performance for all the classifiers were similar. There was minimal change to the scatter plot from one classifier to another. The subtle differences among the scatter plot was primarily at the data points along the boundary. The slight difference in performance was likely attributed to the different classifications of data points along the boundary. These differences were expected given the varying algorithm each classifier uses to classify data. The nuanced differences in data classification was confirmed by the accuracy levels from each classifier; all the classifiers had a similar accuracy level.

Problem 2:

In this question we will use the Sacramento data, which covers available housing in the region of that city. The variables include numerical information about the size of the housing and its price, as well as categorical information like zip code (there are a large but limited number in the area), and the type of unit (condo vs house (coded as residential)).

- a: Load the data from the tidyverse library with the data("Sacramento") command and you should have a variable Sacramento. Because we have categorical, convert them to dummy variables.

```
library(tidyverse)
data(Sacramento)
sacramento <- select(Sacramento, everything())
head(sacramento)
```

```
##         city    zip beds baths sqft        type price latitude longitude
```

```
## 1 SACRAMENTO z95838    2     1  836 Residential 59222 38.63191 -121.4349
## 2 SACRAMENTO z95823    3     1 1167 Residential 68212 38.47890 -121.4310
## 3 SACRAMENTO z95815    2     1  796 Residential 68880 38.61830 -121.4438
## 4 SACRAMENTO z95815    2     1  852 Residential 69307 38.61684 -121.4391
## 5 SACRAMENTO z95824    2     1  797 Residential 81900 38.51947 -121.4358
## 6 SACRAMENTO z95841    3     1 1122       Condo 89921 38.66260 -121.3278
```

```
summary(sacramento)
```

```
##              city          zip           beds            baths
##  SACRAMENTO    :438   z95823 : 61   Min.   :1.000   Min.   :1.000
##  ELK_GROVE     :114   z95828 : 45   1st Qu.:3.000   1st Qu.:2.000
##  ROSEVILLE     : 48   z95758 : 44   Median :3.000   Median :2.000
##  CITRUS_HEIGHTS: 35   z95835 : 37   Mean   :3.276   Mean   :2.053
##  ANTELOPE      : 33   z95838 : 37   3rd Qu.:4.000   3rd Qu.:2.000
##  RANCHO_CORDOVA: 28   z95757 : 36   Max.   :8.000   Max.   :5.000
##  (Other)       :236   (Other):672
##       sqft                type         price            latitude
##  Min.   : 484   Condo       : 53   Min.   : 30000   Min.   :38.24
##  1st Qu.:1167   Multi_Family: 13   1st Qu.:156000   1st Qu.:38.48
##  Median :1470   Residential :866   Median :220000   Median :38.62
##  Mean   :1680                      Mean   :246662   Mean   :38.59
##  3rd Qu.:1954                      3rd Qu.:305000   3rd Qu.:38.69
##  Max.   :4878                      Max.   :884790   Max.   :39.02
##
##    longitude
##  Min.   :-121.6
##  1st Qu.:-121.4
##  Median :-121.4
##  Mean   :-121.4
##  3rd Qu.:-121.3
##  Max.   :-120.6
##
```

```
# dummy variable for each non-classifier, categorical variables (zip, city)
dummy <- dummyVars(type ~ ., data = sacramento)
# Transform dummy variables into dataframe
dummies <- as.data.frame(predict(dummy, newdata = sacramento))
```

```
## Warning in model.frame.default(Terms, newdata, na.action = na.action, xlev =
## object$lvls): variable 'type' is not a factor
```

```
head(dummies)
```

```
##   city.ANTELOPE city.AUBURN city.CAMERON_PARK city.CARMICHAEL
## 1             0           0                 0               0
## 2             0           0                 0               0
## 3             0           0                 0               0
## 4             0           0                 0               0
## 5             0           0                 0               0
## 6             0           0                 0               0
##   city.CITRUS_HEIGHTS city.COOL city.DIAMOND_SPRINGS city.EL_DORADO
```

```
## 1                   0         0                   0              0
## 2                   0         0                   0              0
## 3                   0         0                   0              0
## 4                   0         0                   0              0
## 5                   0         0                   0              0
## 6                   0         0                   0              0
##   city.EL_DORADO_HILLS city.ELK_GROVE city.ELVERTA city.FAIR_OAKS city.FOLSOM
## 1                    0              0            0              0           0
## 2                    0              0            0              0           0
## 3                    0              0            0              0           0
## 4                    0              0            0              0           0
## 5                    0              0            0              0           0
## 6                    0              0            0              0           0
##   city.FORESTHILL city.GALT city.GARDEN_VALLEY city.GOLD_RIVER city.GRANITE_BAY
## 1               0         0                  0              0                0
## 2               0         0                  0              0                0
## 3               0         0                  0              0                0
## 4               0         0                  0              0                0
## 5               0         0                  0              0                0
## 6               0         0                  0              0                0
##   city.GREENWOOD city.LINCOLN city.LOOMIS city.MATHER city.MEADOW_VISTA
## 1              0            0           0           0                0
## 2              0            0           0           0                0
## 3              0            0           0           0                0
## 4              0            0           0           0                0
## 5              0            0           0           0                0
## 6              0            0           0           0                0
##   city.NORTH_HIGHLANDS city.ORANGEVALE city.PENRYN city.PLACERVILLE
## 1                    0              0           0                0
## 2                    0              0           0                0
## 3                    0              0           0                0
## 4                    0              0           0                0
## 5                    0              0           0                0
## 6                    0              0           0                0
##   city.POLLOCK_PINES city.RANCHO_CORDOVA city.RANCHO_MURIETA city.RIO_LINDA
## 1                  0                   0                   0             0
## 2                  0                   0                   0             0
## 3                  0                   0                   0             0
## 4                  0                   0                   0             0
## 5                  0                   0                   0             0
## 6                  0                   0                   0             0
##   city.ROCKLIN city.ROSEVILLE city.SACRAMENTO city.WALNUT_GROVE
## 1            0              0               1                 0
## 2            0              0               1                 0
## 3            0              0               1                 0
## 4            0              0               1                 0
## 5            0              0               1                 0
## 6            0              0               1                 0
##   city.WEST_SACRAMENTO city.WILTON zip.z95603 zip.z95608 zip.z95610 zip.z95614
## 1                    0           0          0          0          0          0
## 2                    0           0          0          0          0          0
## 3                    0           0          0          0          0          0
## 4                    0           0          0          0          0          0
## 5                    0           0          0          0          0          0
```

```
## 6                   0           0           0           0           0           0
##   zip.z95619 zip.z95621 zip.z95623 zip.z95624 zip.z95626 zip.z95628 zip.z95630
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95631 zip.z95632 zip.z95633 zip.z95635 zip.z95648 zip.z95650 zip.z95655
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95660 zip.z95661 zip.z95662 zip.z95663 zip.z95667 zip.z95670 zip.z95673
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95677 zip.z95678 zip.z95682 zip.z95683 zip.z95690 zip.z95691 zip.z95693
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95722 zip.z95726 zip.z95742 zip.z95746 zip.z95747 zip.z95757 zip.z95758
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95762 zip.z95765 zip.z95811 zip.z95814 zip.z95815 zip.z95816 zip.z95817
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          1          0          0
## 4          0          0          0          0          1          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95818 zip.z95819 zip.z95820 zip.z95821 zip.z95822 zip.z95823 zip.z95824
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          1          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          1
## 6          0          0          0          0          0          0          0
##   zip.z95825 zip.z95826 zip.z95827 zip.z95828 zip.z95829 zip.z95831 zip.z95832
## 1          0          0          0          0          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
```

```
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          0          0          0
##   zip.z95833 zip.z95834 zip.z95835 zip.z95838 zip.z95841 zip.z95842 zip.z95843
## 1          0          0          0          1          0          0          0
## 2          0          0          0          0          0          0          0
## 3          0          0          0          0          0          0          0
## 4          0          0          0          0          0          0          0
## 5          0          0          0          0          0          0          0
## 6          0          0          0          0          1          0          0
##   zip.z95864 beds baths sqft price latitude longitude
## 1          0    2     1  836 59222 38.63191 -121.4349
## 2          0    3     1 1167 68212 38.47890 -121.4310
## 3          0    2     1  796 68880 38.61830 -121.4438
## 4          0    2     1  852 69307 38.61684 -121.4391
## 5          0    2     1  797 81900 38.51947 -121.4358
## 6          0    3     1 1122 89921 38.66260 -121.3278
```

```
dummies$type <- as.factor(sacramento$type)
```

- b: With kNN, because of the high dimensionality, which might be a good choice for the distance function?

  With the high dimensionality of the sacramento data set, the best choice for the distance function with kNN would be the Manhattan distance particularly as the data set has sparse data with all the newly created dummy variables. Not only will it be computationally efficient, the Manhattan distance will be robust to noise and outliers.

- c: Use kNN to classify this data with type as the label. Tune the choice of k plus the type of distance function. Report your results – what values for these parameters were tried, which were chosen, and how did they perform with accuracy?

```
library(kknn)
```

```
##
## Attaching package: 'kknn'
```

```
## The following object is masked from 'package:caret':
##
##     contr.dummy
```

```
set.seed(124)
# tuneGrid with the tuning parameters
# test a range of k values 3 to 7
# regular and cosine-based distance functions
# 1 - Manhattan, 2 - Euclidean
tuneGrid <- expand.grid(kmax = 3:7, kernel = c("rectangular", "cos"), distance = 1:2)
train_control <- trainControl(method = 'cv', number = 10)

# tune and fit the model with 10-fold cross validation, standardization, and our specialized tune grid
kknn_fit <- train(type ~., data = dummies, method = 'kknn', trControl = train_control, preProcess = c('
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
```

```
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.DIAMOND_SPRINGS,
## city.FORESTHILL, city.GREENWOOD, city.PENRYN, zip.z95619, zip.z95631,
## zip.z95635, zip.z95663
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
```

```
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MEADOW_VISTA, zip.z95722
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
```

```
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.COOL, city.GARDEN_VALLEY,
## city.GRANITE_BAY, zip.z95614, zip.z95633, zip.z95746
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
```

```
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.MATHER, zip.z95655
```

```
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
## Warning in preProcess.default(thresh = 0.95, k = 5, freqCut = 19, uniqueCut =
## 10, : These variables have zero variances: city.WALNUT_GROVE, zip.z95690
```

```
# Printing trained model provides report
kknn_fit
```

```
## k-Nearest Neighbors
```

```
##
## 932 samples
## 111 predictors
##   3 classes: 'Condo', 'Multi_Family', 'Residential'
##
## Pre-processing: centered (111), scaled (111)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 838, 839, 839, 838, 839, 839, ...
## Resampling results across tuning parameters:
##
##   kmax  kernel       distance  Accuracy   Kappa
##   3     rectangular  1         0.9292486  0.3285250
##   3     rectangular  2         0.9292255  0.3678307
##   3     cos          1         0.9452976  0.5114555
##   3     cos          2         0.9474250  0.5280127
##   4     rectangular  1         0.9292486  0.3285250
##   4     rectangular  2         0.9292255  0.3678307
##   4     cos          1         0.9485122  0.5213096
##   4     cos          2         0.9474250  0.5280127
##   5     rectangular  1         0.9292486  0.3285250
##   5     rectangular  2         0.9292255  0.3678307
##   5     cos          1         0.9485122  0.5213096
##   5     cos          2         0.9474250  0.5280127
##   6     rectangular  1         0.9292486  0.3285250
##   6     rectangular  2         0.9292255  0.3678307
##   6     cos          1         0.9485122  0.5213096
##   6     cos          2         0.9474250  0.5280127
##   7     rectangular  1         0.9292486  0.3285250
##   7     rectangular  2         0.9292255  0.3678307
##   7     cos          1         0.9485122  0.5213096
##   7     cos          2         0.9474250  0.5280127
##
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were kmax = 7, distance = 1 and kernel
##  = cos.
```

The kmax values that were tried were 3 - 7. The kernels tested were 'cos' and 'rectangular'. The distance tested were 1 (Manhattan) and 2 (Euclidean). The final values used for the model were kmax = 7, distance = 1 and kernel = cos. With these selected parameter values, the kNN model had an accuracy of 0.9485.

Problem 3:

In this problem we will continue with the wine quality data from Problem 1, but this time we will use clustering. Do not forget to remove the type variable before clustering because that would be cheating by using the label to perform clustering.

```
head(wine)
```

```
## # A tibble: 6 x 13
##   fixed.acidity volatile.acidity citric.acid residual.sugar chlorides
##           <dbl>            <dbl>       <dbl>          <dbl>     <dbl>
## 1            74             0.7         0                19     0.076
```

```
## 2            78             0.88        0                26     0.098
## 3            78             0.76        0.04             23     0.092
## 4           112             0.28        0.56             19     0.075
## 5            74             0.7         0                19     0.076
## 6            74             0.66        0                18     0.075
## # i 8 more variables: free.sulfur.dioxide <dbl>, total.sulfur.dioxide <dbl>,
## #   density <dbl>, pH <dbl>, sulphates <dbl>, alcohol <dbl>, quality <dbl>,
## #   type <fct>
```

```
wine_class <- wine$type
wine_variables <- wine %>% select(-c(type))
summary(wine_variables)
```

```
##  fixed.acidity    volatile.acidity  citric.acid     residual.sugar
##  Min.   :  5.00   Min.   :0.0800   Min.   :0.0000   Min.   :  0.60
##  1st Qu.: 62.00   1st Qu.:0.2300   1st Qu.:0.2500   1st Qu.:  2.00
##  Median : 68.00   Median :0.2900   Median :0.3100   Median :  7.40
##  Mean   : 65.51   Mean   :0.3396   Mean   :0.3187   Mean   : 11.38
##  3rd Qu.: 76.00   3rd Qu.:0.4000   3rd Qu.:0.3900   3rd Qu.: 15.20
##  Max.   :715.00   Max.   :1.5800   Max.   :1.6600   Max.   :655.00
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :0.00900   Min.   :   1.00     Min.   :   6.0       Min.   :0.9871
##  1st Qu.:0.03800   1st Qu.:  17.00     1st Qu.:  77.0       1st Qu.:0.9923
##  Median :0.04700   Median :  29.00     Median : 118.0       Median :0.9949
##  Mean   :0.05602   Mean   :  34.99     Mean   : 123.5       Mean   :0.9947
##  3rd Qu.:0.06500   3rd Qu.:  41.00     3rd Qu.: 156.0       3rd Qu.:0.9970
##  Max.   :0.61100   Max.   :1465.00     Max.   :3665.0       Max.   :1.0390
##        pH          sulphates         alcohol             quality
##  Min.   :  3.0   Min.   :0.2200   Min.   :8.000e+00   Min.   :3.000
##  1st Qu.:306.0   1st Qu.:0.4300   1st Qu.:9.300e+01   1st Qu.:5.000
##  Median :318.0   Median :0.5100   Median :1.010e+02   Median :6.000
##  Mean   :289.6   Mean   :0.5313   Mean   :1.733e+12   Mean   :5.819
##  3rd Qu.:331.0   3rd Qu.:0.6000   3rd Qu.:1.120e+02   3rd Qu.:6.000
##  Max.   :401.0   Max.   :2.0000   Max.   :9.733e+14   Max.   :9.000
```

```
# Center scale to standardize the data as clustering relies o n distances and dissimilarities
preproc <- preProcess(wine_variables, method=c("center", "scale"))
# Fit data based on preprocessing
wine_variables <- predict(preproc, wine_variables)

summary(wine_variables)
```

```
##  fixed.acidity      volatile.acidity   citric.acid       residual.sugar
##  Min.   :-2.09044   Min.   :-1.5772   Min.   :-2.19463   Min.   :-0.4661
##  1st Qu.:-0.12112   1st Qu.:-0.6658   1st Qu.:-0.47325   1st Qu.:-0.4056
##  Median : 0.08617   Median :-0.3012   Median :-0.06012   Median :-0.1722
##  Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.36257   3rd Qu.: 0.3671   3rd Qu.: 0.49072   3rd Qu.: 0.1649
##  Max.   :22.43971   Max.   : 7.5368   Max.   : 9.23531   Max.   :27.8179
##    chlorides       free.sulfur.dioxide total.sulfur.dioxide    density
##  Min.   :-1.3422   Min.   :-0.5826     Min.   :-0.83046     Min.   :-2.52955
##  1st Qu.:-0.5144   1st Qu.:-0.3083     1st Qu.:-0.32866     1st Qu.:-0.78569
##  Median :-0.2575   Median :-0.1026     Median :-0.03888     Median : 0.06457
```

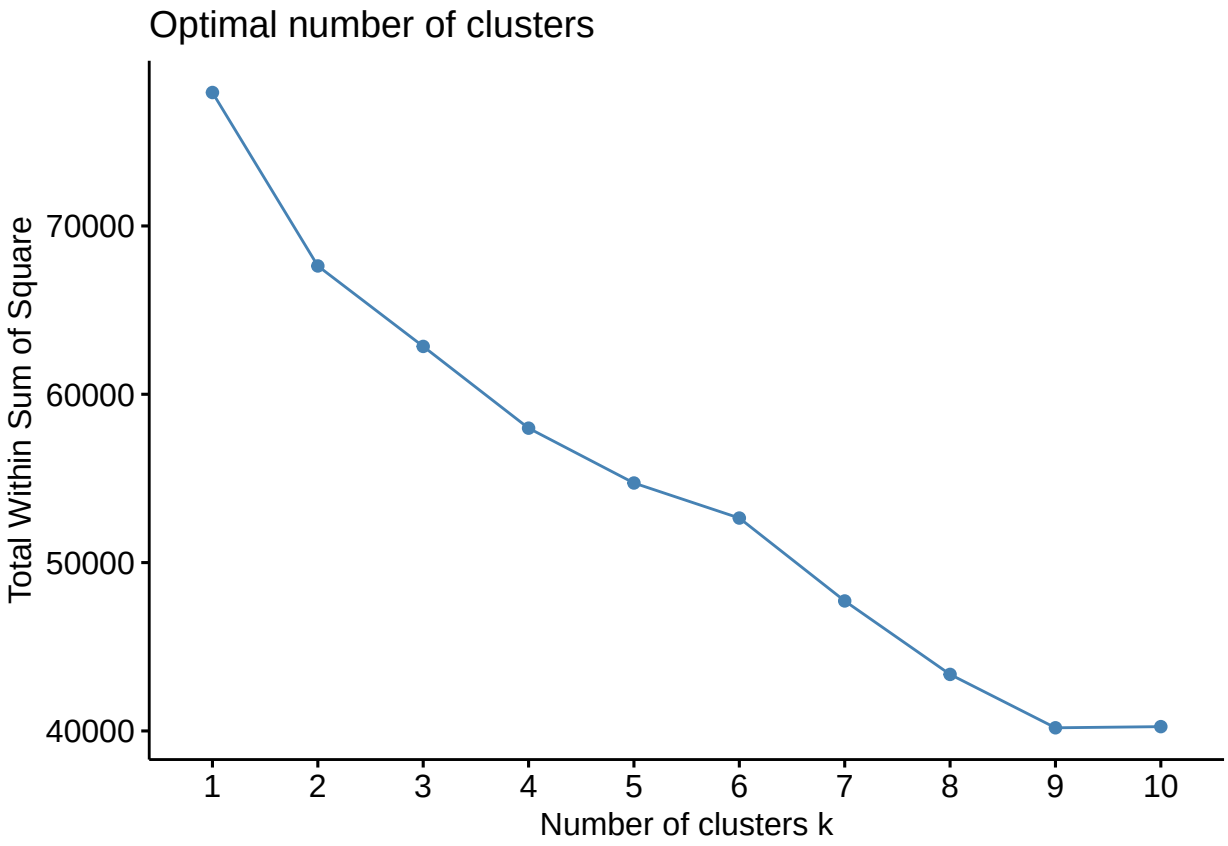
```
##  Mean   : 0.0000   Mean   : 0.0000     Mean   : 0.00000     Mean   : 0.00000
##  3rd Qu.: 0.2563   3rd Qu.: 0.1031     3rd Qu.: 0.22969     3rd Qu.: 0.76478
##  Max.   :15.8416   Max.   :24.5138     Max.   :25.03014     Max.   :14.76567
##        pH            sulphates          alcohol            quality
##  Min.   :-3.0771   Min.   :-2.0917   Min.   :-0.05031   Min.   :-3.2277
##  1st Qu.: 0.1761   1st Qu.:-0.6806   1st Qu.:-0.05031   1st Qu.:-0.9374
##  Median : 0.3049   Median :-0.1431   Median :-0.05031   Median : 0.2077
##  Mean   : 0.0000   Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 0.4445   3rd Qu.: 0.4617   3rd Qu.:-0.05031   3rd Qu.: 0.2077
##  Max.   : 1.1961   Max.   : 9.8689   Max.   :28.20372   Max.   : 3.6430
```

- a: Use k-means to cluster the data. Show your usage of silhouette and the elbow method to pick the best number of clusters. Make sure it is using multiple restarts.

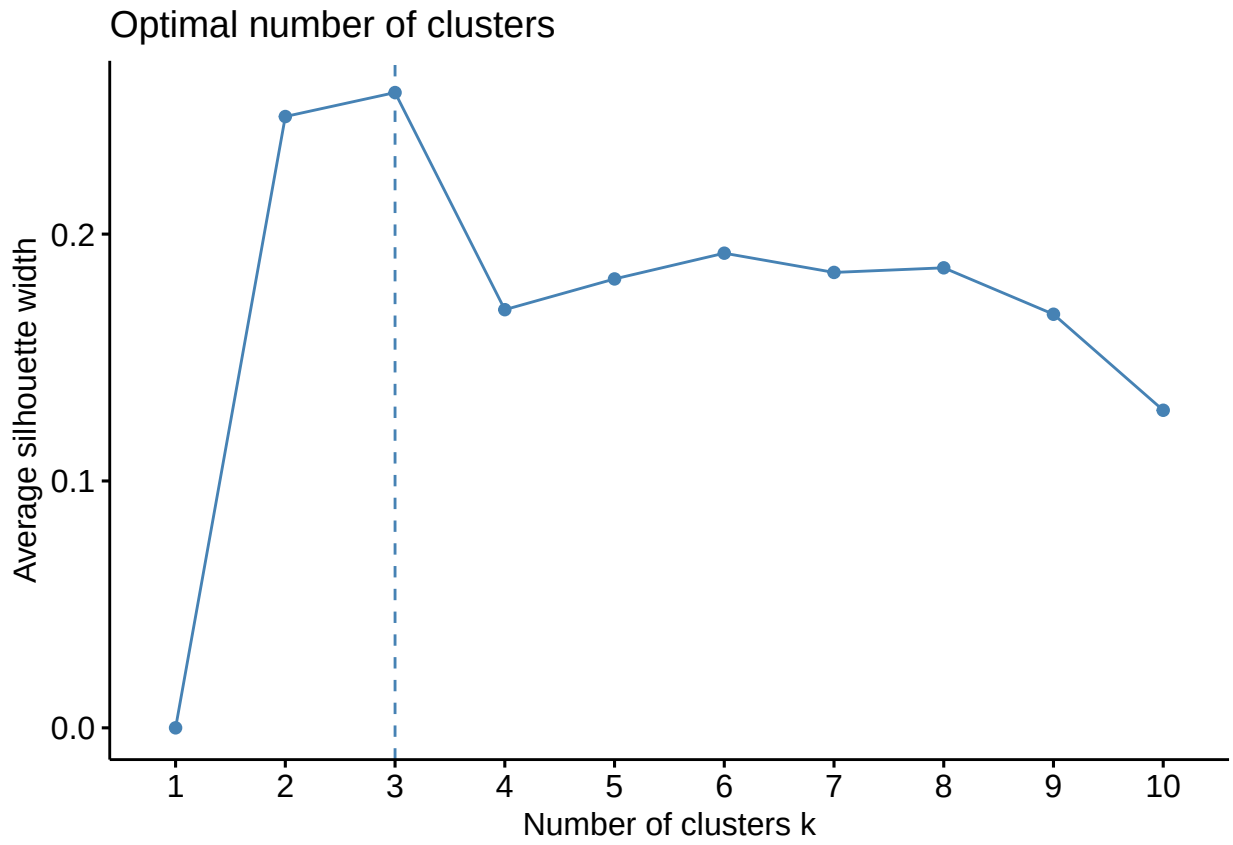
```
library(stats)
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
# Set seed
set.seed(125)

# Elbow
fviz_nbclust(wine_variables, kmeans, method = "wss")
```



```
# Silhouette
fviz_nbclust(wine_variables, kmeans, method = "silhouette")
```




```
# Fit the data
wine_kmeans <- kmeans(wine_variables, centers = 6, nstart = 25)
# Display the kmeans object information
wine_kmeans
```

```
## K-means clustering with 6 clusters of sizes 2196, 925, 65, 2044, 644, 621
##
## Cluster means:
##   fixed.acidity volatile.acidity citric.acid residual.sugar   chlorides
## 1  -0.014642307       -0.3495483  0.20282142   -0.106807271 -0.18012380
## 2  -0.009854666        1.7843078 -1.30196725    0.458789536  0.65551212
## 3   0.129759848       -0.5007976  0.25237642   -0.002261294 -0.01074202
## 4  -0.269209025       -0.4558000  0.03053054   -0.314818481 -0.48989257
## 5  -0.076701973       -0.2230589  0.01921337   -0.086492967 -0.12563475
## 6   1.018510467        0.3622872  1.07526822    0.820460349  1.40442927
##   free.sulfur.dioxide total.sulfur.dioxide     density         pH   sulphates
## 1          0.05370493          0.200622227  0.39744226  0.3018761 -0.29809124
## 2         -0.30871438         -0.475833737  0.49254335  0.3586381  0.36948702
## 3          7.67028038          5.959611316  0.53060362  0.1296770 -0.01179155
## 4         -0.05698307         -0.044709946 -0.94961853  0.3373465 -0.26734264
## 5         -0.05388620         -0.009252076 -0.03322794 -2.7995040 -0.12429800
## 6         -0.28948043         -0.467712520  0.96544933  0.1775446  1.51384153
```

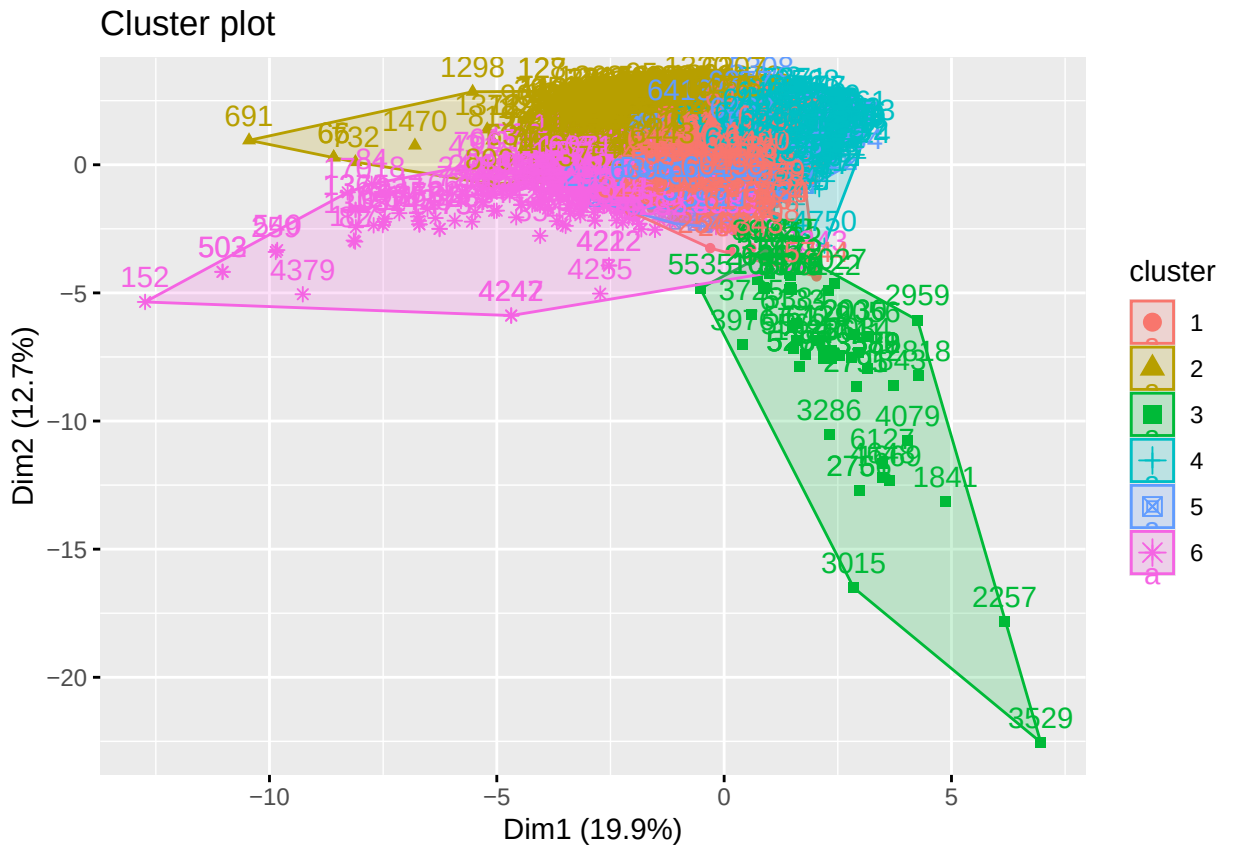
```
##        alcohol     quality
## 1  0.020108233 -0.54059991
## 2  0.016416228 -0.54127922
## 3 -0.050311824  0.29577666
## 4 -0.016291365  0.77632849
## 5  0.004108379  0.05654908
## 6 -0.040931832  0.07307915
##
## Clustering vector:
##    [1] 2 2 2 6 2 2 5 2 2 6 2 6 2 6 6 6 5 6 2 6 1 6 6 2 2 2 2 6 2 2 2 2 2 2 1 2 2
##   [38] 6 2 6 6 2 6 6 2 5 2 6 2 1 2 5 2 5 2 2 6 2 2 1 6 6 2 2 2 2 2 2 6 2 2 2 2 2
##   [75] 6 6 6 2 2 2 2 6 1 6 6 2 6 2 6 2 2 6 6 2 2 2 2 2 2 5 5 2 5 2 2 2 6 2 6 6 2
##  [112] 2 2 6 2 6 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 5 2 2 2 1 2 2 2 2 2 2 2 6 2 6
##  [149] 2 6 1 6 2 2 6 6 6 6 2 2 2 2 2 2 2 6 2 2 2 6 2 2 2 2 1 2 1 6 2 2 2 6 2 2 2
##  [186] 6 1 2 5 5 2 1 2 2 2 2 2 6 2 2 6 6 2 1 1 6 6 2 2 6 5 2 6 5 2 6 2 2 2 5 1 2
##  [223] 2 2 2 2 6 2 2 2 2 2 2 6 2 2 2 2 2 2 2 2 6 6 2 6 6 2 2 5 2 2 6 2 6 2 2 2 6 2 6
##  [260] 6 1 2 2 2 6 5 2 6 2 6 2 6 6 2 2 2 2 6 6 6 6 6 2 6 2 2 5 2 2 6 2 6 2 2 6 6
##  [297] 2 2 2 2 2 2 6 2 2 5 6 2 6 6 1 6 2 2 5 6 6 5 2 6 2 6 2 2 6 2 2 6 6 6 6 6 6 2
##  [334] 2 2 6 6 6 6 6 6 6 6 6 6 6 2 2 6 6 2 6 2 2 2 6 3 2 6 6 6 6 2 6 6 6 6 6 6 6 6 6
##  [371] 2 6 6 2 6 6 6 6 6 6 6 6 6 6 2 2 2 2 6 6 2 6 1 6 6 6 2 6 6 5 2 1 6 6 2 6 6
##  [408] 6 6 6 1 6 2 6 6 6 6 2 6 2 6 2 2 6 2 2 2 2 2 2 6 6 2 5 6 6 6 2 6 6 2 6 6 6 4
##  [445] 2 2 6 6 5 6 6 6 2 6 5 6 6 2 6 6 6 2 6 6 6 6 6 6 6 2 6 5 6 6 6 2 6 6 2 2 6
##  [482] 6 6 6 6 2 2 2 6 6 2 6 6 2 2 6 2 6 6 2 2 6 6 6 6 6 6 6 2 6 6 2 6 6 6 6 6 6
##  [519] 6 6 6 6 6 6 6 2 6 6 6 2 6 6 6 6 2 6 2 2 6 6 2 6 2 2 6 6 2 6 6 6 2 6 6 2 6
##  [556] 6 6 6 6 5 6 1 1 6 5 6 6 6 6 2 6 2 6 5 6 6 2 5 6 6 6 6 6 4 6 2 6 2 4 6 1 4
##  [593] 1 6 2 6 6 6 5 6 2 5 2 5 2 2 6 6 6 2 6 6 2 6 6 5 5 6 6 6 2 2 2 6 2 2 2 2 2
##  [630] 2 2 6 2 2 1 2 2 2 2 6 6 6 6 6 6 2 2 2 6 2 6 2 6 6 1 2 6 6 2 2 2 2 2 6 6 2
##  [667] 6 6 2 6 2 5 2 5 6 6 6 2 2 1 6 2 6 2 2 2 2 2 5 6 2 2 6 2 2 5 2 2 2 6 2 2 2
##  [704] 6 2 2 2 2 2 2 6 2 2 2 1 2 2 1 2 2 2 2 6 2 6 2 2 2 2 2 2 6 2 2 2 2 2 2 2 5 2
##  [741] 2 2 2 6 6 2 5 5 2 2 2 2 2 2 6 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 6 6 2 2
##  [778] 2 2 5 2 2 2 2 2 1 1 2 2 2 6 2 2 2 6 6 5 6 6 6 2 2 2 2 2 4 4 4 2 2 5 6 6 2
##  [815] 6 6 2 6 2 2 2 4 2 2 2 2 5 2 2 2 2 2 6 6 2 2 4 4 6 2 6 2 6 2 6 2 2 5 2 2 6
##  [852] 6 6 6 6 2 6 5 6 5 2 2 1 2 2 2 2 5 5 2 2 2 1 4 6 6 5 2 2 2 2 2 6 2 2 2 2 6
##  [889] 2 2 1 2 6 2 2 2 6 2 6 2 5 2 2 2 2 2 2 2 2 4 6 6 1 6 4 6 2 6 5 2 6 5 2 6 2
##  [926] 6 6 2 2 5 2 2 1 2 2 6 6 2 6 2 6 4 6 6 6 6 6 6 6 6 6 6 6 5 6 6 6 6 2 2 6 2
##  [963] 2 6 6 6 4 2 5 2 6 6 6 6 5 1 1 2 4 6 6 2 6 6 6 2 4 1 2 1 2 1 2 1 1 2 2 2 2
## [1000] 2 6 6 6 4 1 4 6 6 6 6 4 1 2 2 2 6 6 4 4 2 6 6 2 6 2 2 1 2 1 2 2 2 2 5 5 6
## [1037] 4 2 6 6 2 2 6 6 4 2 2 2 6 6 2 6 2 6 2 2 6 2 6 6 6 6 5 6 2 2 5 6 6 2 6 2 5
## [1074] 2 2 6 6 6 6 3 1 3 1 6 1 2 6 1 6 6 4 1 2 6 2 6 2 5 6 5 6 2 2 2 2 2 1 6 2 6
## [1111] 2 4 6 1 4 2 2 2 5 2 5 2 2 6 2 4 4 2 2 6 2 1 6 2 5 4 6 6 2 2 6 6 5 1 1 6 6
## [1148] 1 6 1 5 2 2 6 2 2 6 4 6 6 6 6 6 2 2 6 6 6 1 2 5 2 6 1 1 2 2 2 2 6 6 6 6 2
## [1185] 2 4 2 4 2 2 6 2 6 2 2 2 2 2 1 2 2 4 6 2 5 5 5 6 5 6 2 2 2 6 5 6 2 6 6 5 6
## [1222] 6 2 6 6 2 2 2 4 5 6 2 5 1 2 5 2 2 2 2 5 6 4 2 6 2 2 2 6 5 5 2 2 2 2 2 5 2
## [1259] 2 2 6 2 6 2 6 2 2 6 1 5 5 4 5 5 2 2 1 2 2 4 2 2 2 2 2 6 4 2 6 6 2 2 2 2 2
## [1296] 2 2 2 2 2 5 1 2 5 2 2 2 2 5 2 2 5 5 2 2 6 2 6 2 2 1 6 2 2 2 2 2 2 2 2 2 2
## [1333] 2 2 2 2 2 2 2 2 2 2 6 2 2 2 2 2 2 2 2 2 2 1 1 5 2 6 2 2 6 2 2 2 2 6 2 2 6
## [1370] 6 6 2 2 2 2 2 2 2 2 2 5 5 2 2 2 2 2 5 2 2 2 5 2 2 2 2 2 2 2 2 6 6 2 6 6 2
## [1407] 6 2 2 6 6 6 5 2 5 6 2 2 2 2 2 2 6 6 1 5 2 6 2 1 2 2 6 6 6 2 2 2 4 2 2 2 2
## [1444] 2 2 2 2 4 4 6 2 2 6 2 5 2 1 4 2 2 2 2 2 2 5 2 5 2 2 2 6 2 6 2 6 2 2 6 2 6
## [1481] 6 1 2 2 2 2 2 2 2 4 2 2 2 4 2 2 2 2 2 2 2 2 6 6 2 2 6 6 1 2 2 2 2 2 2 6 2 6
## [1518] 2 2 2 6 2 2 2 2 5 2 6 2 2 2 2 1 2 2 5 5 2 6 2 5 2 6 6 2 2 2 5 4 2 2 2 2 2 2
## [1555] 2 2 6 2 2 2 2 2 2 2 2 6 2 2 2 6 2 2 2 6 6 5 2 2 2 1 2 2 2 6 5 6 2 2 2 2 2 2
## [1592] 2 2 2 2 2 1 5 5 1 1 1 1 1 5 5 1 4 1 1 4 6 4 4 4 4 1 4 4 4 2 1 1 4 1 4 5 5
## [1629] 4 4 4 1 1 1 4 1 1 1 5 1 4 4 4 1 1 1 1 4 4 4 4 1 4 1 5 4 4 1 5 4 4 4 2 4 1
```

```
## [1666] 4 1 5 3 1 1 4 1 4 4 1 1 4 1 1 1 1 1 1 1 1 1 1 4 4 4 1 1 4 1 1 5 5 1 1 5
## [1703] 5 5 1 1 1 1 1 1 1 1 2 4 4 1 1 2 1 1 1 4 4 1 4 5 4 1 4 1 1 1 1 4 1 4 4 5 1
## [1740] 4 1 4 5 4 2 4 4 4 1 4 1 6 1 1 4 4 4 4 1 1 1 1 1 1 4 1 1 4 4 1 4 1 4 1 1 2
## [1777] 1 1 1 1 1 1 1 1 1 4 1 1 1 4 1 6 6 1 1 5 1 1 5 5 1 1 1 1 1 2 4 1 4 1 1 5 1
## [1814] 1 1 1 1 1 5 5 4 4 1 5 1 5 1 2 1 1 1 1 1 1 1 4 1 5 1 4 3 1 3 4 4 4 1 1 1 1
## [1851] 1 4 4 4 4 4 1 4 1 1 1 5 1 1 2 2 2 1 2 1 1 1 1 1 1 1 4 4 5 1 1 1 1 1 1 1 1
## [1888] 1 1 1 4 1 4 1 4 1 4 2 4 2 4 5 5 5 5 5 4 5 4 1 4 1 1 4 1 4 4 4 1 4 1 1 1 1
## [1925] 4 1 1 4 5 4 1 1 4 4 1 1 4 4 5 1 1 4 1 4 5 1 1 4 4 4 4 3 1 1 4 5 5 1 1 5 1
## [1962] 4 5 4 4 4 1 4 4 2 5 4 4 4 4 4 4 4 4 5 1 5 4 4 4 3 1 1 1 5 2 4 1 1 1 4 1 5 4
## [1999] 1 1 1 4 1 4 1 4 1 1 5 1 1 1 5 4 5 4 4 4 1 1 1 4 1 1 1 1 5 1 1 3 2 4 4 3 4
## [2036] 3 2 4 5 4 1 5 1 1 1 1 4 2 1 4 4 4 1 5 1 3 1 1 1 1 4 1 1 1 4 1 1 5 1 4 4 4
## [2073] 1 4 4 2 4 1 4 1 1 6 4 4 5 4 1 4 4 1 5 4 4 1 4 1 1 1 1 5 1 1 1 2 4 4 4 4 4
## [2110] 4 4 4 4 1 5 4 1 1 1 4 4 4 1 1 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 4 1 1 4 1 5 4
## [2147] 1 4 4 5 4 5 4 6 4 1 4 4 1 1 4 5 1 1 1 4 1 1 4 1 4 4 4 4 4 4 4 4 1 1 1 1 4 1
## [2184] 1 4 4 4 4 1 4 5 4 4 4 4 5 4 4 4 4 6 4 4 4 4 4 4 1 1 1 1 4 4 1 5 5 1 4 4 4 4 1 1 4
## [2221] 1 2 4 2 1 1 4 1 1 4 1 4 4 4 4 5 5 5 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 4 4 1 1 3
## [2258] 1 4 5 1 1 1 4 1 4 1 1 4 4 4 4 1 1 1 4 4 4 4 1 1 1 6 1 1 5 6 1 4 1 4 1 1 1 1 4
## [2295] 1 1 1 1 4 1 4 4 1 1 1 1 1 1 5 5 1 4 4 1 1 4 4 1 5 4 4 4 1 4 1 1 2 1 1 4 1
## [2332] 1 1 1 1 4 1 1 4 4 1 1 6 1 4 5 4 4 1 1 1 1 4 1 5 5 5 5 5 5 2 1 4 1 4 4 4 4
## [2369] 4 4 5 5 4 4 1 6 6 2 1 6 6 1 1 1 1 4 1 1 4 5 4 1 1 1 1 4 4 1 1 1 1 1 1 1 1
## [2406] 1 1 1 4 4 1 1 1 1 5 1 5 4 2 5 1 1 1 1 4 4 4 1 1 4 4 1 4 4 4 4 4 4 4 1 1 4 5
## [2443] 5 1 1 1 4 4 5 4 1 4 4 5 5 1 1 4 6 4 4 1 4 4 6 6 4 1 1 4 1 1 5 4 4 1 4 5 1
## [2480] 4 4 1 5 4 4 4 4 4 5 4 1 4 4 4 4 1 4 4 1 4 1 1 5 4 1 1 4 1 1 1 1 1 1 1 1 4 4 4
## [2517] 4 4 1 1 1 4 1 2 4 1 1 4 1 1 1 1 1 1 1 1 1 1 1 4 4 4 4 4 1 1 2 1 1 1 4 1 1 1
## [2554] 1 4 4 1 1 4 1 4 1 1 4 1 4 4 4 4 1 4 4 1 1 1 4 1 4 4 5 4 1 4 1 1 1 4 4 1 1
## [2591] 1 1 4 4 1 1 4 4 1 1 4 4 4 4 2 4 4 4 4 1 1 1 1 6 4 4 1 4 4 4 1 6 4 4 2 4 1
## [2628] 1 1 3 3 6 1 1 1 4 1 2 1 1 1 4 1 1 4 1 4 5 1 1 1 4 1 4 2 4 5 4 1 1 5 1 1 1
## [2665] 5 4 4 5 1 4 1 1 4 1 1 1 1 4 1 1 1 4 1 4 4 1 4 1 1 4 5 1 4 1 4 1 4 1 4 1 1
## [2702] 1 4 4 5 4 1 5 4 1 1 2 4 1 1 1 1 1 4 4 1 1 5 4 4 1 4 4 5 1 1 1 1 5 4 1 1 4
## [2739] 1 1 1 1 1 1 1 5 1 1 1 2 5 1 1 1 1 3 1 1 1 4 3 4 1 1 4 1 1 1 1 4 1 1 5 1 1
## [2776] 1 1 1 1 1 4 4 1 1 4 4 5 4 4 1 3 1 1 4 3 1 5 5 5 5 4 4 1 1 5 4 4 4 1 4 4 4
## [2813] 1 4 6 4 4 3 4 4 1 4 4 4 5 4 5 1 4 4 1 4 1 4 1 4 1 1 5 4 4 1 1 4 4 4 1 4 1
## [2850] 4 4 1 1 1 1 1 1 1 1 1 3 1 4 4 1 1 1 1 1 5 1 1 4 1 1 4 4 4 4 1 4 4 4 4 4 4
## [2887] 4 4 1 5 5 5 1 1 4 5 1 1 1 5 1 1 4 4 1 1 1 1 4 1 1 1 5 4 1 4 1 4 4 4 1 4 5
## [2924] 1 4 4 1 1 5 1 4 4 1 4 1 1 1 4 1 1 1 4 4 4 4 4 1 4 4 1 1 1 1 1 1 4 4 1 3 4
## [2961] 1 1 4 4 1 1 5 1 5 6 6 4 4 4 4 4 5 4 4 2 4 4 1 4 4 5 4 1 4 4 4 4 4 5 1 1 4
## [2998] 1 1 5 4 5 1 4 4 1 4 4 4 4 1 4 2 4 3 1 4 5 4 1 1 1 4 4 1 5 5 5 5 5 4 1 1 6
## [3035] 5 1 1 1 1 1 4 1 4 4 4 4 5 1 1 1 4 1 5 4 4 4 1 1 1 4 4 4 1 5 4 1 1 4 1 1 4
## [3072] 1 1 2 4 4 1 1 4 1 1 1 4 5 1 1 1 4 1 1 4 4 4 2 4 4 4 4 4 1 1 4 4 4 4 5 1 1 1 1
## [3109] 1 4 1 4 4 1 1 4 1 1 1 1 1 1 1 4 1 4 1 1 5 4 4 6 4 1 1 4 4 6 1 4 5 1 1 1 1
## [3146] 4 4 1 1 4 4 1 4 1 1 1 1 4 1 5 4 1 1 1 1 1 1 1 1 1 1 6 4 1 1 1 1 4 4 1 1
## [3183] 1 1 1 1 4 4 1 4 1 4 1 2 1 6 6 4 1 4 4 1 4 4 4 1 1 4 1 4 4 4 1 5 1 1 1 4 4 1
## [3220] 1 1 4 1 1 1 4 1 1 4 4 5 1 4 1 4 6 5 1 1 1 1 1 4 1 4 5 1 4 6 1 1 1 1 1 1 1
## [3257] 1 1 1 4 1 1 4 1 4 4 4 4 5 5 5 5 5 1 5 4 1 1 1 1 1 1 3 1 1 3 3 4 1 1 4 1 1 4
## [3294] 1 4 4 4 4 4 1 1 1 1 4 4 1 2 1 1 1 1 5 1 4 4 5 1 4 4 5 1 4 4 1 4 1 1 5 4 1 2
## [3331] 4 1 4 1 1 1 1 1 4 1 1 5 1 1 1 1 5 1 4 1 4 4 1 1 1 5 2 1 4 4 5 1 1 1 1 1 4
## [3368] 1 4 1 1 4 1 1 4 4 4 1 1 1 1 4 1 4 4 1 4 1 1 5 1 1 1 1 4 1 1 1 1 1 1 1 1 1
## [3405] 6 1 6 4 1 4 4 5 4 4 2 4 4 4 1 5 1 1 1 1 1 1 4 4 4 4 4 5 6 1 4 1 1 1 4 1 6
## [3442] 4 1 1 1 6 4 5 4 5 1 4 1 2 1 1 1 1 1 4 1 4 6 1 5 4 4 4 4 1 1 1 1 4 4 1 1 1
## [3479] 1 1 1 5 1 5 1 1 1 1 1 5 5 1 1 5 5 1 4 1 1 1 1 1 5 1 4 4 1 1 4 4 1 5 4 1 4
## [3516] 1 1 3 1 1 1 1 6 6 1 4 1 1 3 2 5 4 1 1 4 1 1 1 1 1 1 1 1 5 1 4 1 1 2 4 1 1
## [3553] 1 1 1 1 1 1 5 1 1 1 1 1 4 4 4 1 4 1 1 1 1 1 1 4 1 3 3 3 3 3 5 5 1 1 4 1 3
## [3590] 5 4 1 6 1 6 6 1 1 1 4 4 1 1 6 4 1 4 4 1 1 1 1 4 5 1 4 1 4 4 1 4 6 6 6 1 5
## [3627] 1 1 1 4 1 1 1 5 1 1 1 1 1 4 4 2 1 1 1 1 4 1 5 5 1 1 4 5 4 1 1 1 4 1 5 5 1
```

```
## [3664] 1 1 4 4 1 1 1 1 5 5 5 1 4 1 5 1 1 1 1 1 1 1 1 1 1 2 1 1 5 1 1 1 1 1 4 1
## [3701] 1 1 1 1 1 1 1 5 5 5 1 1 5 1 1 4 5 1 4 1 1 4 5 1 3 2 1 1 1 1 4 4 1 1 1 1
## [3738] 1 1 5 5 1 5 1 5 1 4 4 1 5 1 2 1 1 4 4 5 1 4 2 4 1 1 1 3 3 3 3 3 1 1 3 3 1
## [3775] 4 1 4 5 4 1 1 1 4 6 1 1 1 4 1 1 1 1 1 5 4 1 1 1 4 1 1 1 1 5 4 1 4 4 4 4 4
## [3812] 1 1 1 4 1 4 4 1 4 1 4 1 1 1 4 1 4 1 1 1 1 1 1 1 5 1 1 1 1 1 1 4 1 5 4 1 6
## [3849] 4 1 1 4 1 1 6 1 6 1 1 4 1 1 1 1 1 1 6 4 1 5 1 4 1 1 4 1 6 4 4 1 1 1 1 2 2
## [3886] 1 4 4 5 4 4 4 4 1 4 1 4 4 4 4 1 1 1 1 4 1 1 1 4 1 4 1 4 4 5 4 1 1 4 5 2 4 4 1
## [3923] 4 1 4 1 1 1 1 1 4 1 1 1 1 1 4 1 1 4 5 4 1 1 5 4 6 1 1 4 4 4 4 5 5 1 2 1 1
## [3960] 4 1 3 1 1 1 1 1 4 1 1 1 5 4 5 1 3 1 1 5 4 4 4 1 1 1 1 4 4 1 4 1 1 1 1 4 4
## [3997] 4 1 5 1 4 1 1 4 4 5 1 1 1 1 1 2 1 1 2 1 1 4 1 2 4 2 4 4 1 4 1 1 1 1 1 1 1
## [4034] 1 4 1 1 2 4 5 3 5 3 1 1 4 4 2 2 4 4 5 1 1 1 1 1 1 1 1 4 1 1 1 5 4 1 5 4 1
## [4071] 1 4 2 1 4 1 1 1 3 1 1 1 5 1 4 4 1 6 1 1 1 1 1 1 1 1 4 1 1 1 1 1 5 1 4 1 1
## [4108] 1 1 4 1 4 4 1 1 4 1 1 4 4 1 1 4 4 4 4 1 4 1 1 1 4 4 1 1 1 1 1 4 1 1 4 4 1
## [4145] 1 5 1 1 5 1 4 4 4 5 4 4 5 4 1 4 5 4 1 4 1 1 1 1 4 1 1 1 1 1 1 5 5 5 5 5 5
## [4182] 5 5 5 4 5 5 1 4 5 1 4 1 1 4 1 1 4 1 4 4 4 5 4 1 1 5 1 1 4 1 6 4 4 4 4 5 1
## [4219] 1 4 4 6 1 4 4 1 2 5 4 1 5 6 5 5 6 4 4 4 1 1 1 6 4 5 1 1 6 1 2 4 1 5 1 1 6
## [4256] 4 4 1 4 4 4 4 4 1 5 5 5 5 5 4 5 4 4 4 4 4 4 1 5 4 1 5 1 4 4 4 4 1 1 5 4 5 4 5 5
## [4293] 4 1 4 4 4 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 5 1 1 5 4 4 4 6 4 4 4 1 4 4 1 4 1 1
## [4330] 1 4 1 1 4 4 1 4 1 1 4 4 4 4 4 5 4 5 4 4 4 4 4 4 1 1 1 4 1 1 1 1 4 1 4 4 1 4
## [4367] 1 1 5 4 4 4 4 4 4 4 4 4 5 6 1 1 1 1 4 1 1 1 1 1 5 4 1 4 4 1 4 4 1 5 5 4 4 4
## [4404] 1 1 5 4 4 1 4 4 4 4 1 4 1 4 6 5 1 4 1 4 1 4 4 1 1 1 4 4 4 4 1 1 5 4 4 4 5
## [4441] 5 5 4 5 5 1 6 1 1 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 4 4 4 4 4 4 4 1 1 4 4
## [4478] 1 4 5 4 4 4 4 1 1 1 4 4 4 1 4 4 4 1 1 1 4 1 5 4 1 4 4 5 5 4 4 4 1 1 4 1 4 4 4
## [4515] 5 4 1 1 1 4 4 1 5 5 4 4 1 4 5 1 4 5 1 5 5 4 4 4 4 1 1 4 4 4 4 4 4 1 5 1 1 1 1
## [4552] 4 1 5 4 1 4 4 4 4 1 4 1 4 1 1 4 4 4 4 1 4 1 1 4 4 4 1 1 4 4 4 4 1 4 4 4 4 4 4 4 4 4
## [4589] 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 5 4 1 1 4 1 4 4 4 4 1 1 2 1 1 4 1 5
## [4626] 5 4 4 1 1 1 1 4 1 1 4 1 5 1 1 6 1 4 4 1 4 4 1 3 1 4 1 5 4 4 1 1 5 1 1 4 1 1
## [4663] 1 1 1 4 4 4 5 4 4 1 4 1 1 4 1 5 1 4 4 4 4 1 4 1 4 4 4 4 1 4 4 4 4 4 4 2 4 4 4 4
## [4700] 4 4 1 5 4 4 4 1 5 5 4 1 1 4 4 4 4 4 4 4 1 1 4 4 4 1 1 4 1 4 5 4 1 4 4 4 4 1 5 4 4
## [4737] 1 1 4 4 5 1 1 1 1 5 1 5 4 1 4 4 4 4 4 1 4 4 5 4 1 4 4 4 4 4 4 4 4 4 4 4 4 4 5 1
## [4774] 4 5 5 5 4 4 4 1 4 4 4 4 1 4 4 4 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 4 4 4 4 1 1 1 1 4
## [4811] 4 4 4 4 1 1 5 4 4 5 4 1 4 4 4 4 1 1 4 1 4 1 4 4 4 4 4 4 4 1 5 4 5 4 5 4 1 4 4 4 4 4
## [4848] 4 1 4 5 1 5 5 5 5 5 5 4 5 4 4 4 1 4 4 4 4 1 5 4 4 4 4 1 2 4 4 4 1 4 5 4 4 4 1 4 4 4
## [4885] 4 1 1 1 4 1 4 4 4 1 1 1 4 5 1 4 4 4 4 4 4 4 5 5 4 4 4 4 4 5 1 4 4 4 4 4 5 1 4 4 4 4
## [4922] 4 4 1 4 5 4 1 1 4 4 4 4 5 5 5 1 4 4 4 4 1 5 5 1 1 4 1 1 4 4 4 4 4 4 4 4 1 4 4 4 4 4
## [4959] 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 1 4 5 4 1 4 5 4 4 4 1 4 4 4 1 4 4 4 5 4 4 4 4 5 5 5 4
## [4996] 1 4 1 1 1 1 1 4 4 4 4 1 1 1 4 4 4 1 1 4 4 4 5 2 5 4 1 4 4 4 1 4 1 1 1 4 4 4 1 1 1 1 4 4
## [5033] 4 5 1 1 1 4 4 1 5 4 1 4 1 4 4 4 4 4 4 1 4 2 5 4 4 4 4 1 1 4 4 4 4 1 4 4 4 1 4 4 1 4 1 4 1
## [5070] 5 5 4 1 4 4 4 1 4 5 1 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 1 4 5 6 4 4 4 4 1 1 4 4 4 4 4 4 1
## [5107] 1 4 4 4 4 4 4 4 4 4 4 1 4 1 4 5 1 4 4 4 4 1 2 4 1 1 5 1 1 1 5 1 4 4 4 4 4 5 4 4 5
## [5144] 5 5 4 4 4 4 1 1 1 4 4 4 4 4 1 1 4 2 4 2 4 4 5 4 4 4 4 4 2 5 4 4 4 4 1 4 5 4 1 4 5
## [5181] 4 4 4 4 4 1 1 1 4 1 4 4 4 4 4 5 1 4 5 5 4 1 4 4 4 1 1 4 1 1 1 4 4 4 1 1 4 5 4 4 4 1
## [5218] 4 4 4 4 1 4 5 4 1 1 1 4 4 5 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 5 5 4 1 1 4 1 1 4 4
## [5255] 4 4 5 4 5 2 4 4 5 4 4 1 4 1 4 4 4 4 1 4 4 4 4 5 4 3 4 4 4 3 4 3 1 1 4 4 5 4 4 4
## [5292] 5 4 4 3 4 1 5 1 1 1 1 1 4 4 1 1 5 1 1 1 1 1 1 4 1 1 4 1 1 4 4 4 4 4 4 4 1 1
## [5329] 4 1 4 1 4 1 6 4 4 4 4 4 4 4 4 4 4 1 5 4 4 4 1 1 1 4 4 4 1 1 1 4 4 4 4 1 4 4 4 1 1 1
## [5366] 1 1 1 1 1 4 1 4 1 4 4 5 1 4 4 4 1 5 4 3 1 1 1 1 1 1 1 1 5 1 1 1 5 1 4 1 4 1
## [5403] 4 1 1 4 4 4 1 1 1 5 1 4 1 4 4 4 4 5 1 4 1 4 4 4 4 4 4 4 4 1 1 1 1 1 5 5 1 1 4 4
## [5440] 4 5 4 1 4 1 6 1 4 4 4 5 4 4 4 4 4 4 1 1 5 4 4 4 4 5 1 1 5 1 4 1 1 1 4 5 4 1
## [5477] 2 5 5 1 4 5 4 4 4 5 1 4 1 1 1 4 4 1 1 4 1 1 2 4 5 4 4 4 4 4 4 4 4 6 4 4 4 4
## [5514] 4 1 4 5 4 1 4 1 1 1 1 1 1 1 1 4 4 4 4 1 4 1 1 3 4 4 1 1 4 4 4 4 4 4 4 1 1 1 1 4
## [5551] 4 1 4 1 1 4 4 1 1 4 4 3 5 4 1 1 4 4 4 4 6 1 1 4 4 4 4 4 4 1 4 4 4 4 4 5 5 1 5 5
## [5588] 1 1 5 4 4 5 4 4 6 6 6 5 1 4 4 4 4 4 5 1 5 5 4 6 1 1 1 5 4 1 1 1 1 5 1 4 4 1
## [5625] 1 5 4 4 1 1 4 1 4 1 1 1 2 1 1 5 5 5 5 1 5 5 4 5 4 1 5 1 4 1 4 5 4 2 2 4 1
```

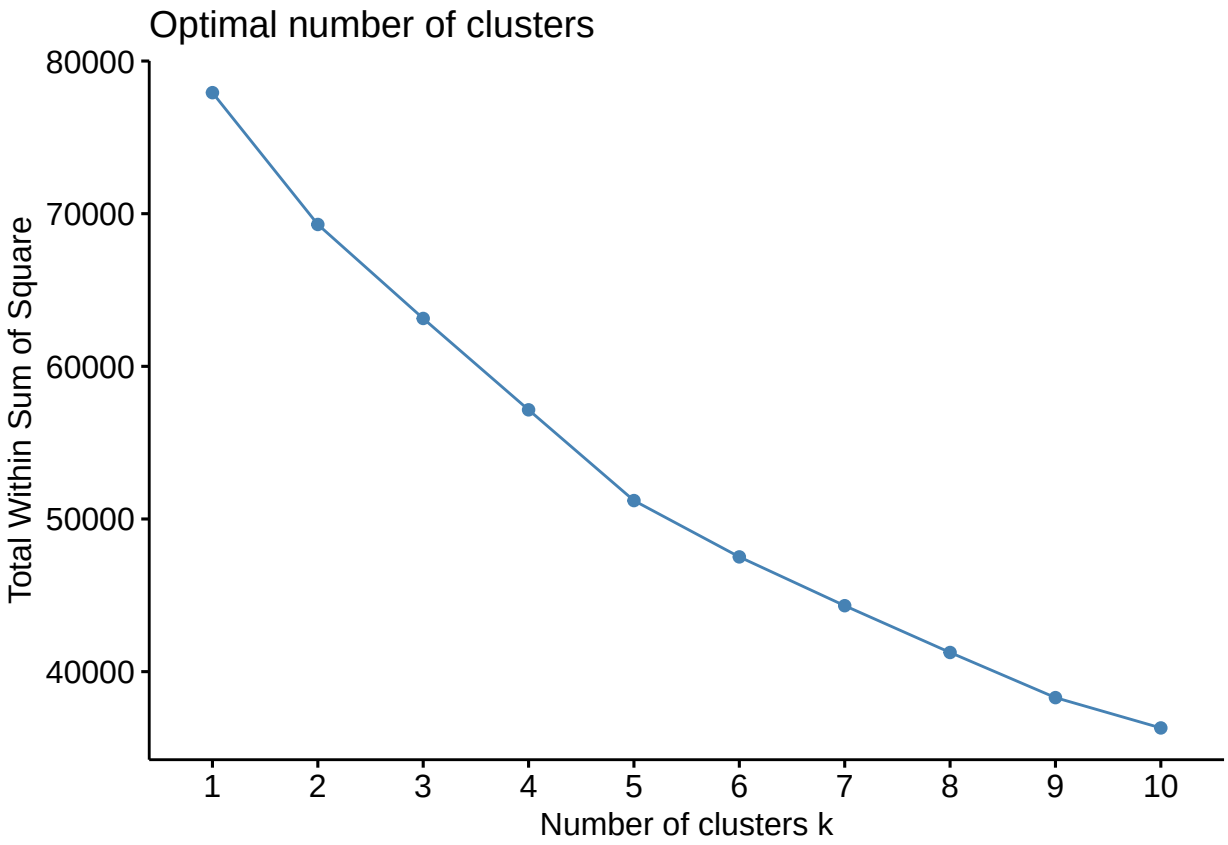
```
## [5662] 4 4 4 1 4 1 1 4 3 1 1 3 1 5 5 4 5 3 1 1 4 4 4 4 4 1 1 4 4 1 4 1 1 4 1 4 1
## [5699] 1 1 4 4 4 1 1 4 4 1 4 4 4 4 4 1 1 4 4 1 1 1 5 5 1 1 1 1 4 4 1 1 4 4 1 4 1
## [5736] 4 5 1 1 4 1 1 1 1 5 1 5 1 5 1 1 1 1 1 5 5 5 5 5 5 1 1 4 4 4 4 4 5 4 4 6 1
## [5773] 5 5 4 4 5 1 4 1 4 4 5 1 4 4 1 5 1 4 4 4 4 4 1 1 4 4 4 4 5 1 4 1 4 4 4 4 1
## [5810] 1 1 1 1 1 1 4 1 4 4 1 1 4 1 4 4 1 4 4 4 4 4 4 1 1 1 1 4 1 5 4 4 1 1 4 6 4
## [5847] 4 5 5 5 2 1 1 1 4 4 4 4 4 4 2 4 4 4 4 4 1 1 1 1 1 1 1 4 1 1 1 1 5 5 4 4 4
## [5884] 4 1 4 1 5 1 4 1 5 1 4 4 4 5 5 1 1 4 4 4 1 4 4 4 1 4 4 4 5 4 2 5 5 4 1 5 4
## [5921] 4 5 1 1 1 1 1 1 1 4 4 4 4 4 4 4 4 4 4 1 4 4 6 4 6 4 1 1 4 4 1 4 4 4 1 4 1 1
## [5958] 1 4 4 1 4 1 4 1 1 4 1 1 4 4 1 4 1 4 1 1 4 4 5 4 1 1 4 4 1 1 1 1 1 1 1 5 1
## [5995] 1 1 5 1 6 4 1 1 1 4 4 1 4 1 4 4 4 4 1 1 4 1 1 1 1 4 4 1 1 1 1 5 3 1 4 4 1
## [6032] 4 4 4 4 1 2 1 4 1 2 4 4 4 4 1 1 1 1 1 4 5 1 1 1 4 1 5 4 4 4 1 1 1 5 4 4 4
## [6069] 4 4 6 5 4 4 1 1 2 5 1 4 1 2 4 4 4 4 4 4 5 4 4 1 4 1 4 5 1 4 5 2 4 2 1 1 4 5
## [6106] 5 1 4 4 4 4 1 4 4 5 4 1 1 1 4 1 3 1 1 5 4 3 3 3 4 1 4 3 1 1 4 1 4 4 1 1 4
## [6143] 4 4 4 1 1 5 4 4 4 4 1 1 1 4 4 4 4 4 5 4 1 3 1 4 4 5 4 5 5 4 4 4 1 4 1 1 4
## [6180] 5 1 4 4 4 4 1 4 4 1 4 5 4 1 4 2 5 5 5 5 1 4 4 4 1 1 5 2 4 1 1 1 5 4 1 4 4
## [6217] 2 4 4 4 4 4 4 1 1 4 4 4 4 4 4 1 4 4 1 1 4 1 1 4 4 4 1 4 4 4 4 4 2 2 2 4 1 5 4 4
## [6254] 4 4 4 4 4 4 1 4 4 4 4 4 1 4 4 4 1 1 1 1 4 4 4 4 1 4 1 1 2 4 4 4 4 4 1 2 1 1 1 1 1 1
## [6291] 1 1 4 4 4 6 5 5 2 2 4 5 1 4 5 4 4 5 5 1 4 4 1 4 4 4 1 4 1 4 1 1 1 4 4 1 1
## [6328] 1 4 4 4 4 4 4 4 4 4 4 4 1 4 4 4 4 1 1 1 1 1 1 4 1 4 4 4 1 4 4 4 4 4 4 4 1 4 4 4 4 4
## [6365] 5 1 1 1 1 1 4 1 4 4 4 4 1 5 1 1 1 1 1 4 4 4 4 1 4 4 4 1 2 1 1 4 4 4 1 4 4 4 4 4 4
## [6402] 1 4 1 4 1 4 1 4 4 5 4 5 1 5 4 4 5 4 4 5 1 4 4 5 1 5 4 4 1 4 4 4 4 1 4 4 1 4
## [6439] 4 4 1 4 2 4 4 4 4 5 4 5 5 4 5 5 5 1 1 4 4 4 4 4 5 5 4 4 4 4 4 1 4 4 1 4 4 4 4 4 2
## [6476] 2 1 1 4 1 4 1 1 4 4 4 1 1 4 4 4 1 4 1 4 4 4
##
## Within cluster sum of squares by cluster:
## [1] 11821.630  7879.669  5449.194  6911.112  4252.546 12209.414
##  (between_SS / total_SS =  37.7 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```r
# Display cluster plot
fviz_cluster(wine_kmeans, data = wine_variables)
```

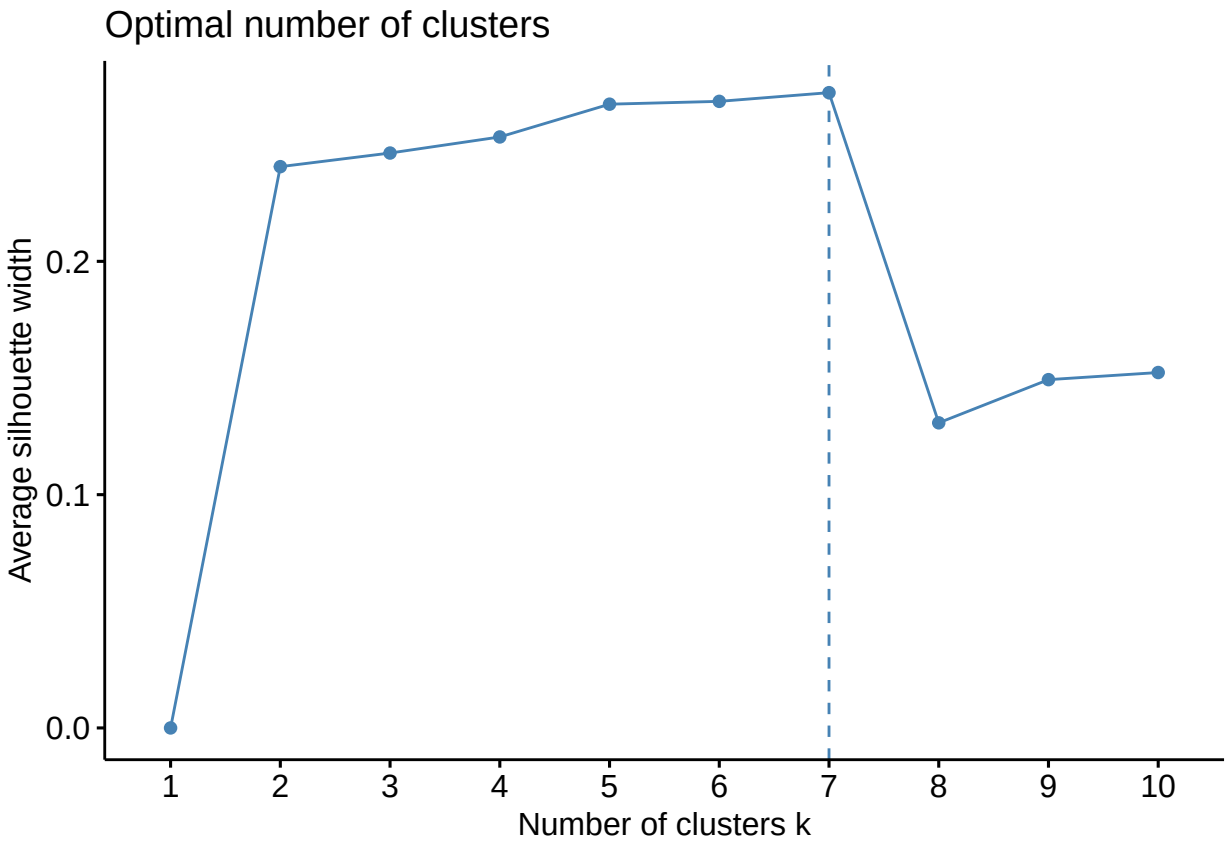


- b: Use hierarchical agglomerative clustering (HAC) to cluster the data. Try at least 2 distance functions and at least 2 linkage functions (cluster distance functions), for a total of 4 parameter combinations. For each parameter combination, perform the clustering.

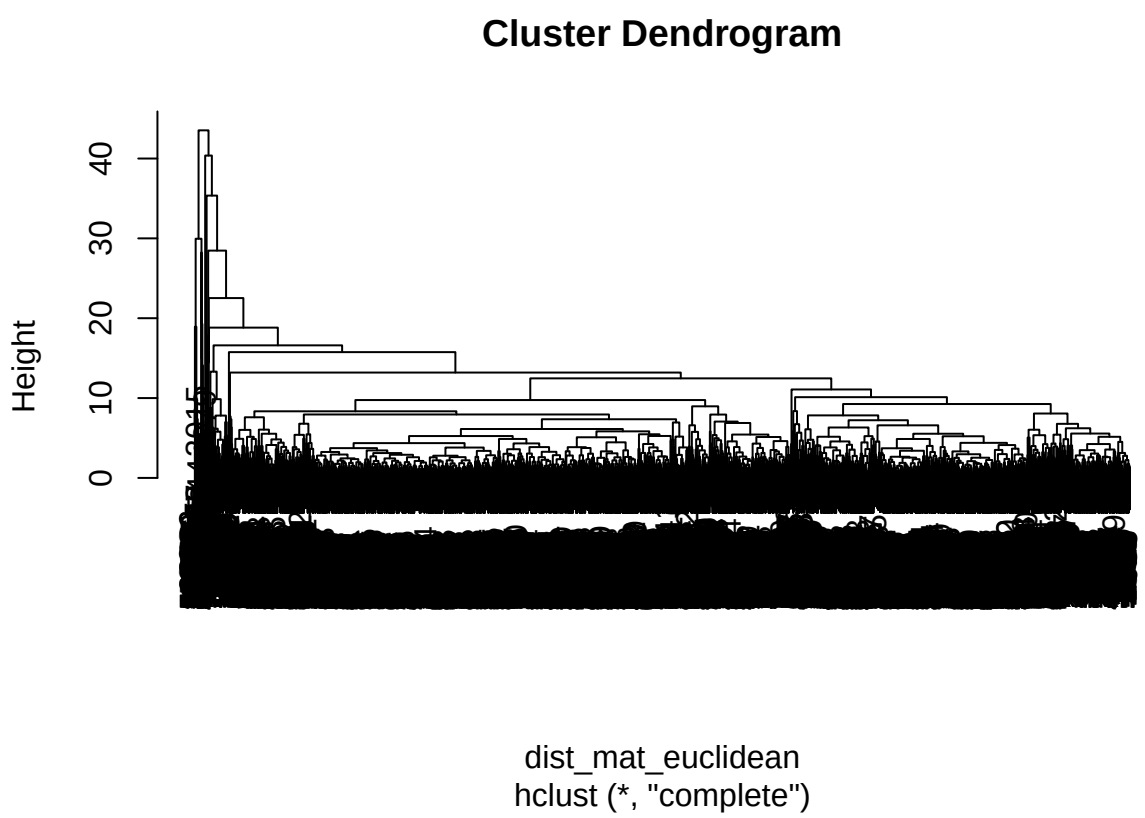
```
library(stats)
library(factoextra)

fviz_nbclust(wine_variables, FUN = hcut, method = "wss")
```



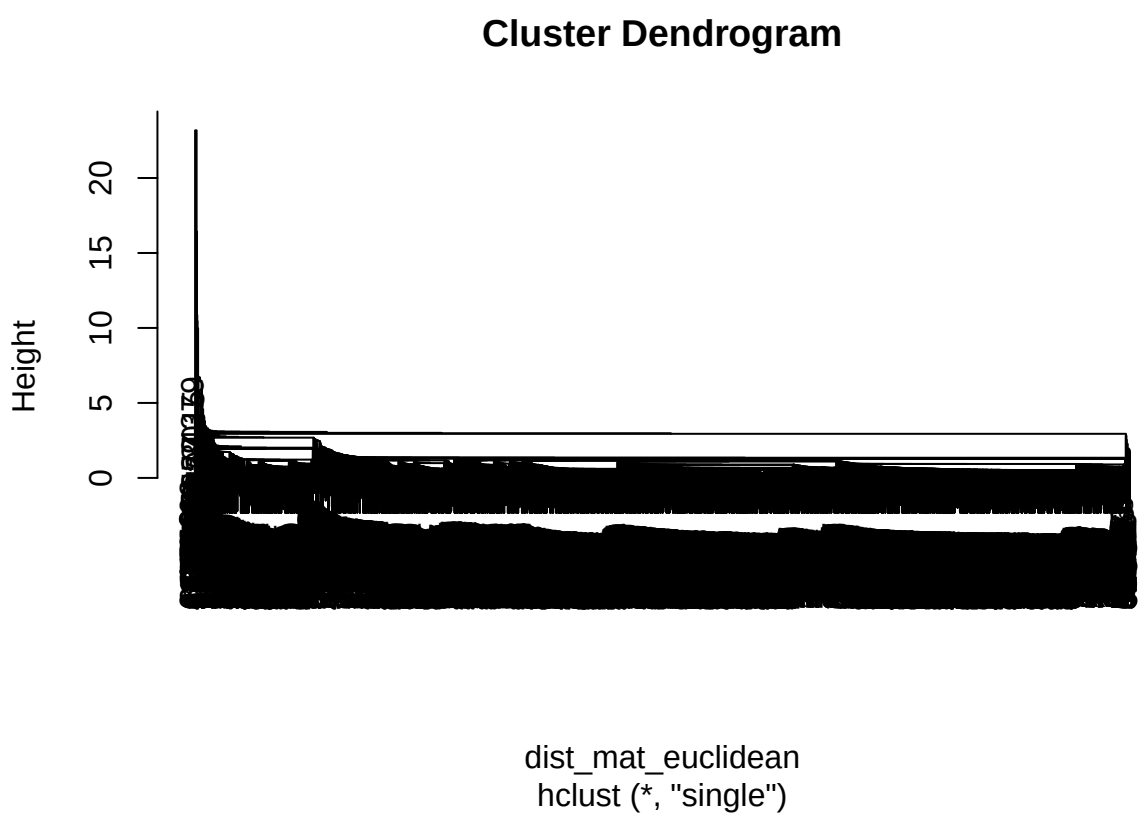

```
fviz_nbclust(wine_variables, FUN = hcut, method = "silhouette")
```



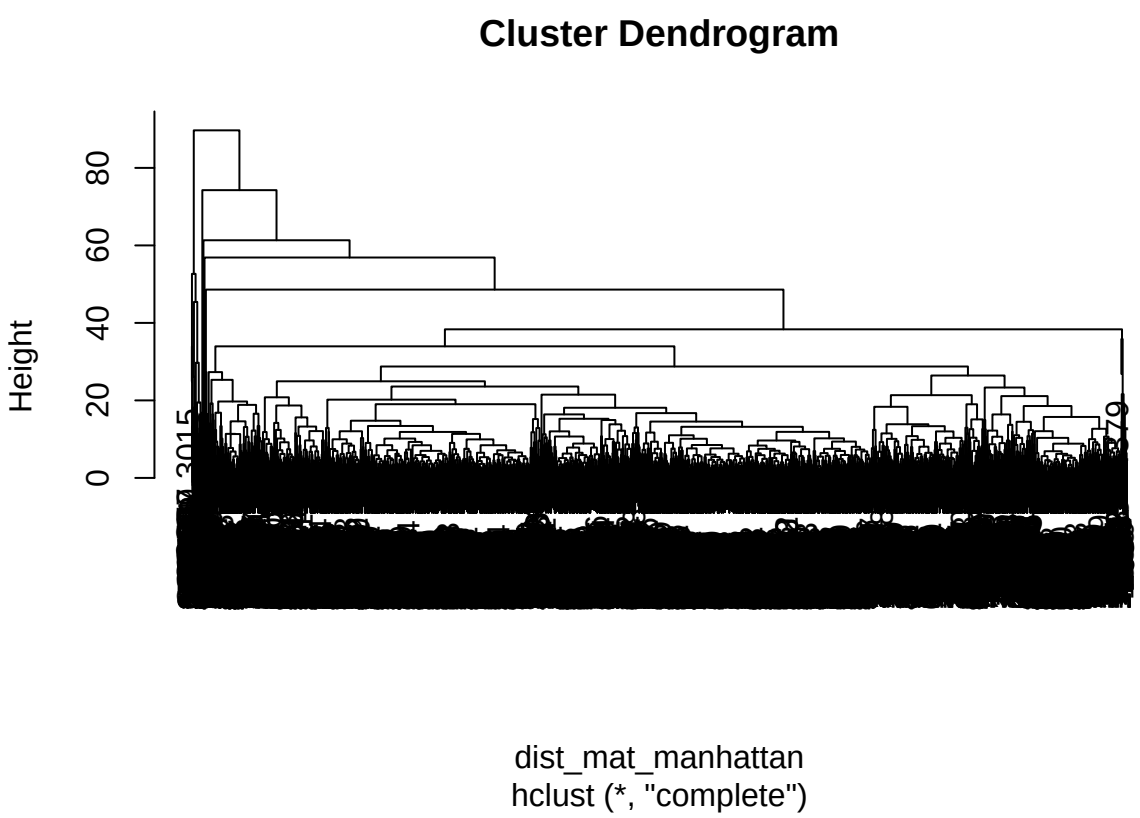

```
# Calculate euclidean distances
dist_mat_euclidean <- dist(wine_variables, method = 'euclidean')
# Calculate manhattan distances
dist_mat_manhattan <- dist(wine_variables, method = 'manhattan')

# Determine assembly/agglomeration method and run hclust (average uses mean)
hfit1 <- hclust(dist_mat_euclidean, method = 'complete')
hfit2 <- hclust(dist_mat_euclidean, method = 'single')

hfit3 <- hclust(dist_mat_manhattan, method = 'complete')
hfit4 <- hclust(dist_mat_manhattan, method = 'single')

# dendogram
plot(hfit1)
```


Cluster Dendrogram
Height
0
10
20
30
40
dist_mat_euclidean
hclust (*, "complete")

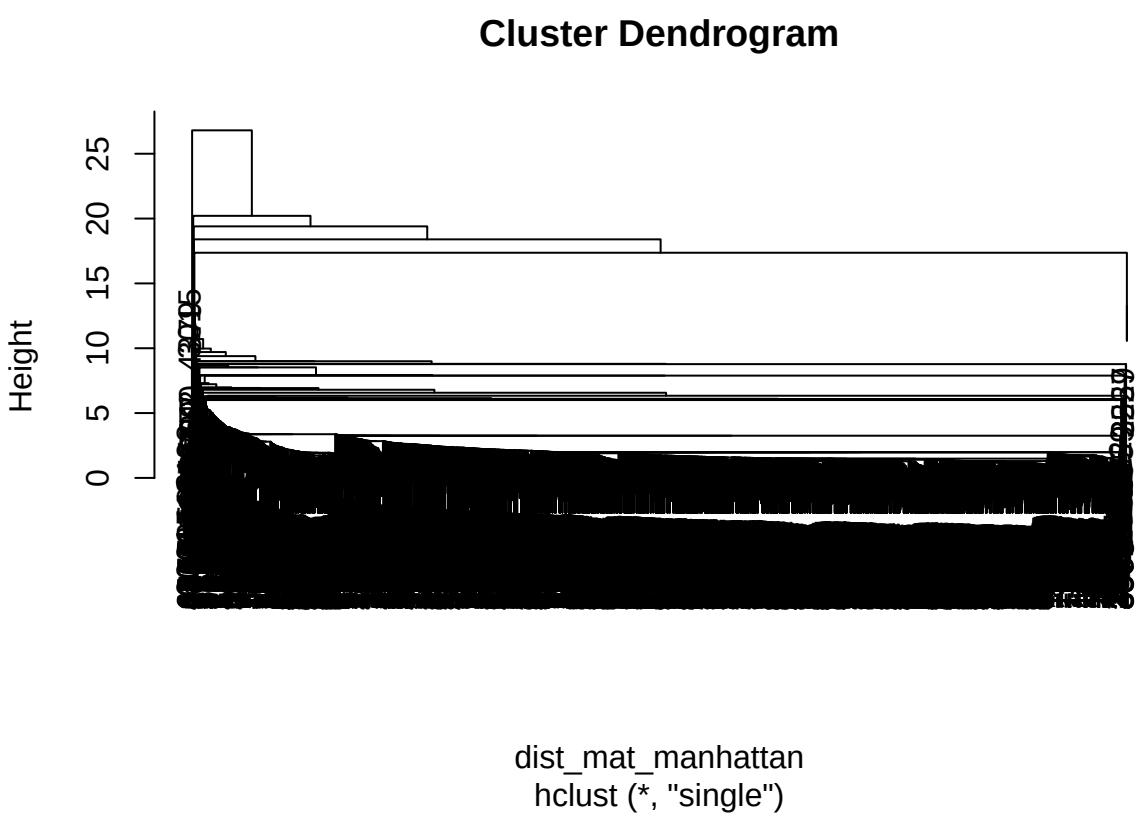
```
plot(hfit2)
```


Cluster Dendrogram
Height
0
5
10
15
20
dist_mat_euclidean
hclust (*, "single")

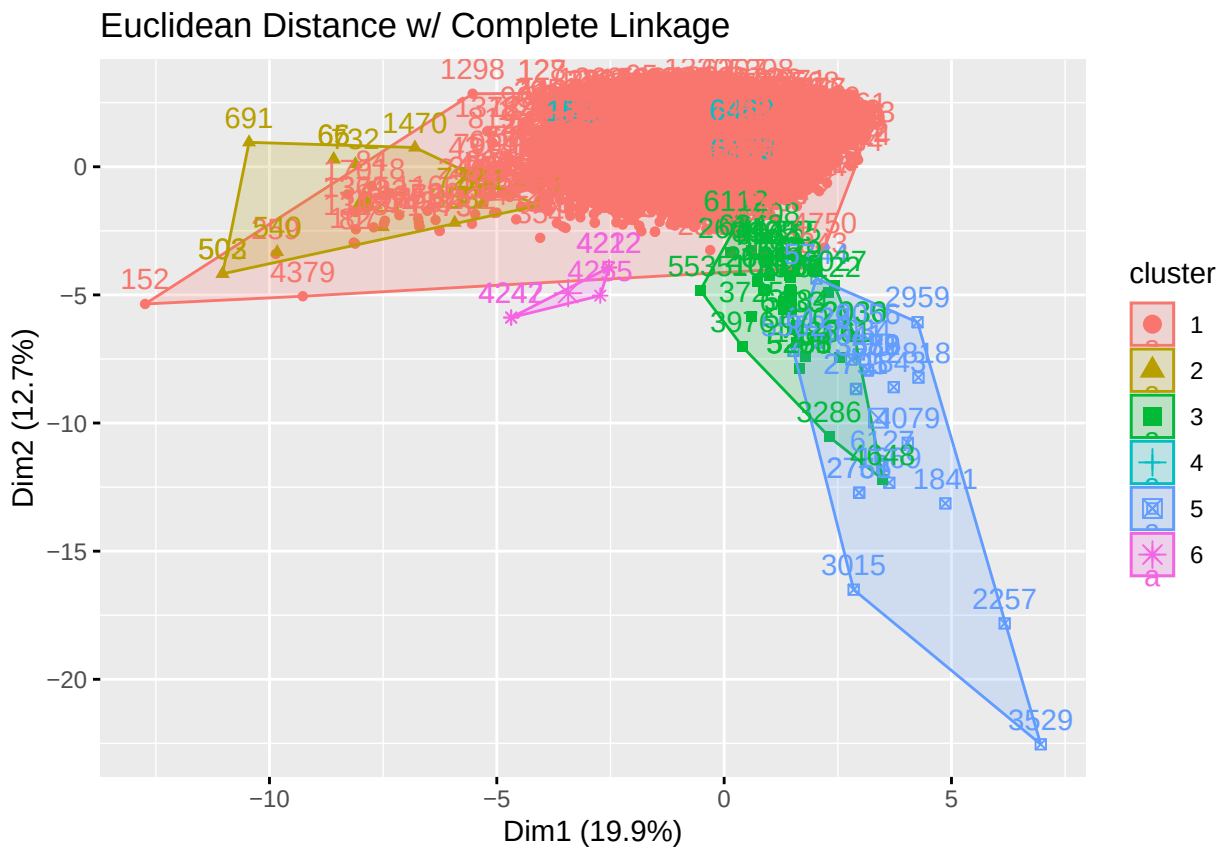
```
plot(hfit3)
```


Cluster Dendrogram
Height
0
20
40
60
80
dist_mat_manhattan
hclust (*, "complete")

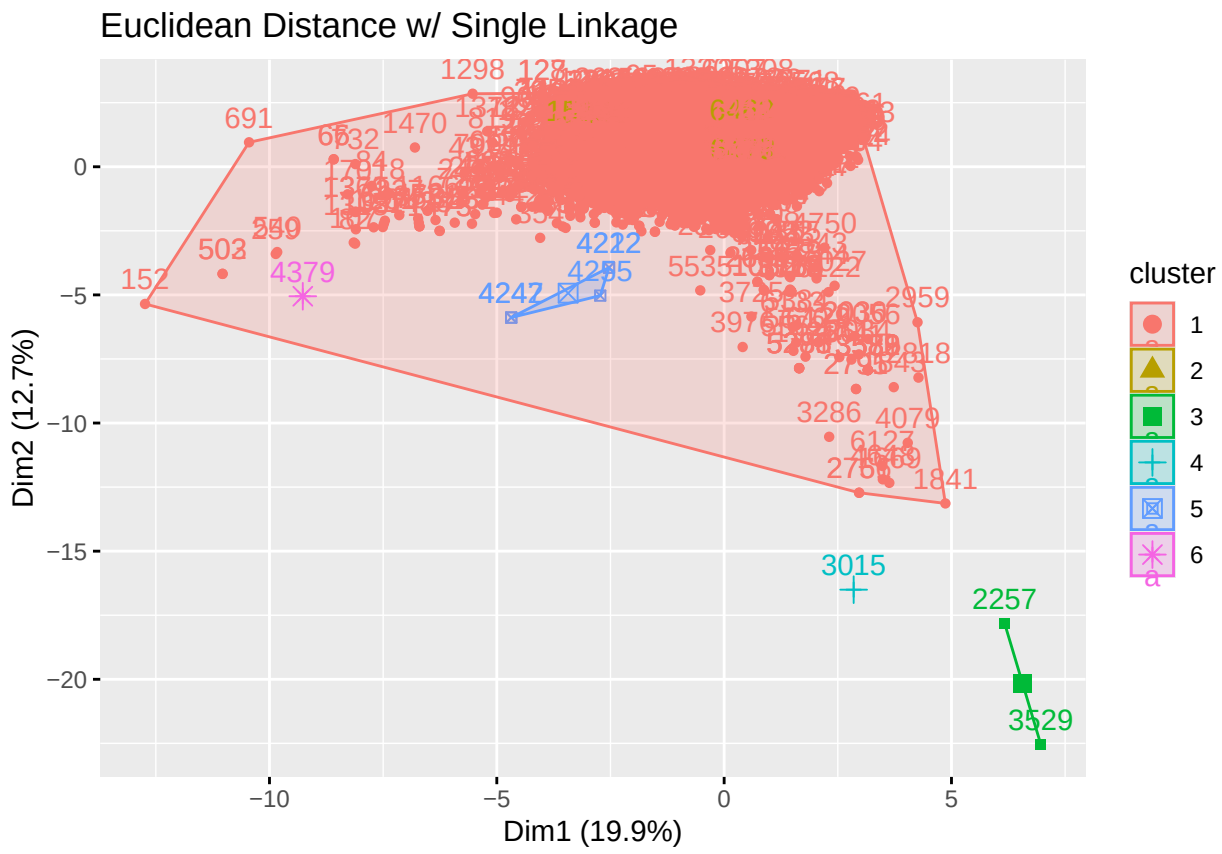
```
plot(hfit4)
```



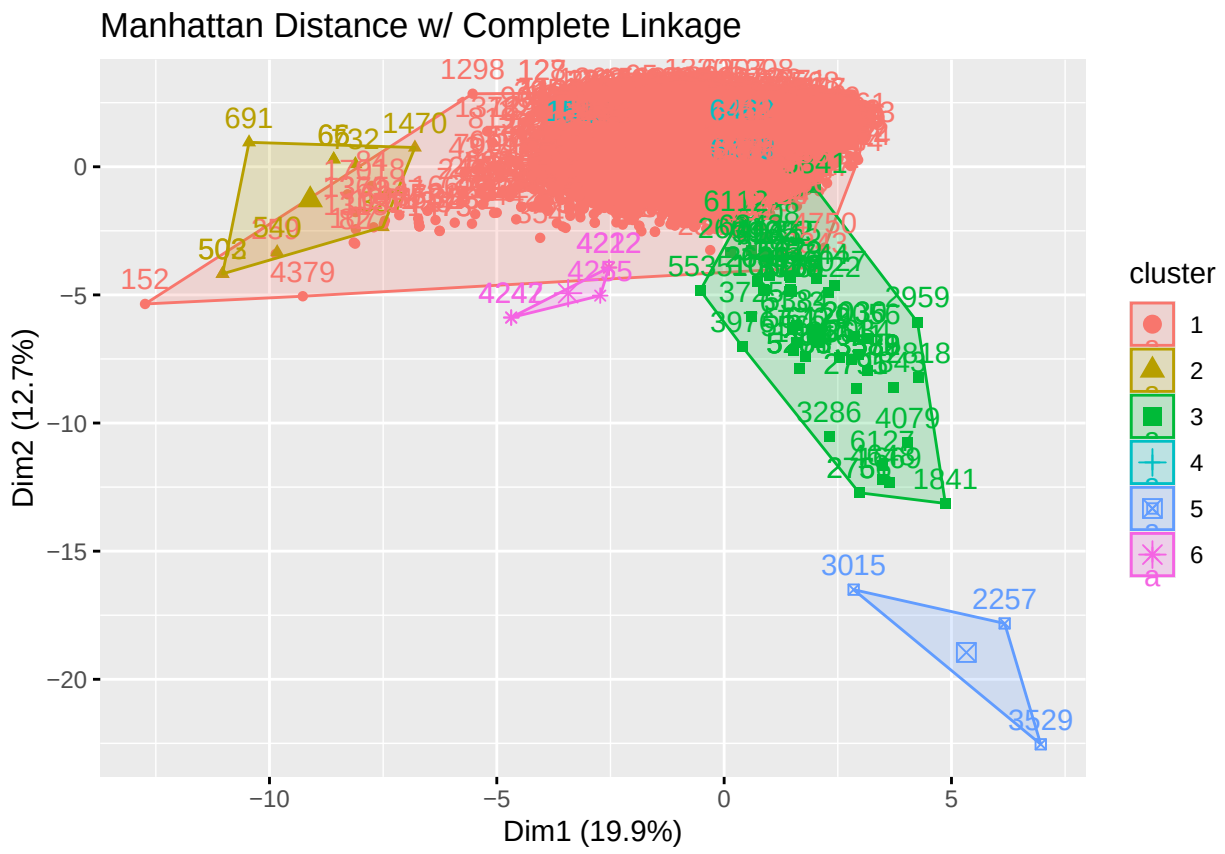

```r
# Build the new model
h1 <- cutree(hfit1, k=6)
h2 <- cutree(hfit2, k=6)
h3 <- cutree(hfit3, k=6)
h4 <- cutree(hfit4, k=6)

# Visualize cluster HAC
fviz_cluster(list(data = wine_variables, cluster = h1)) + labs(title = "Euclidean Distance w/ Complete l
```



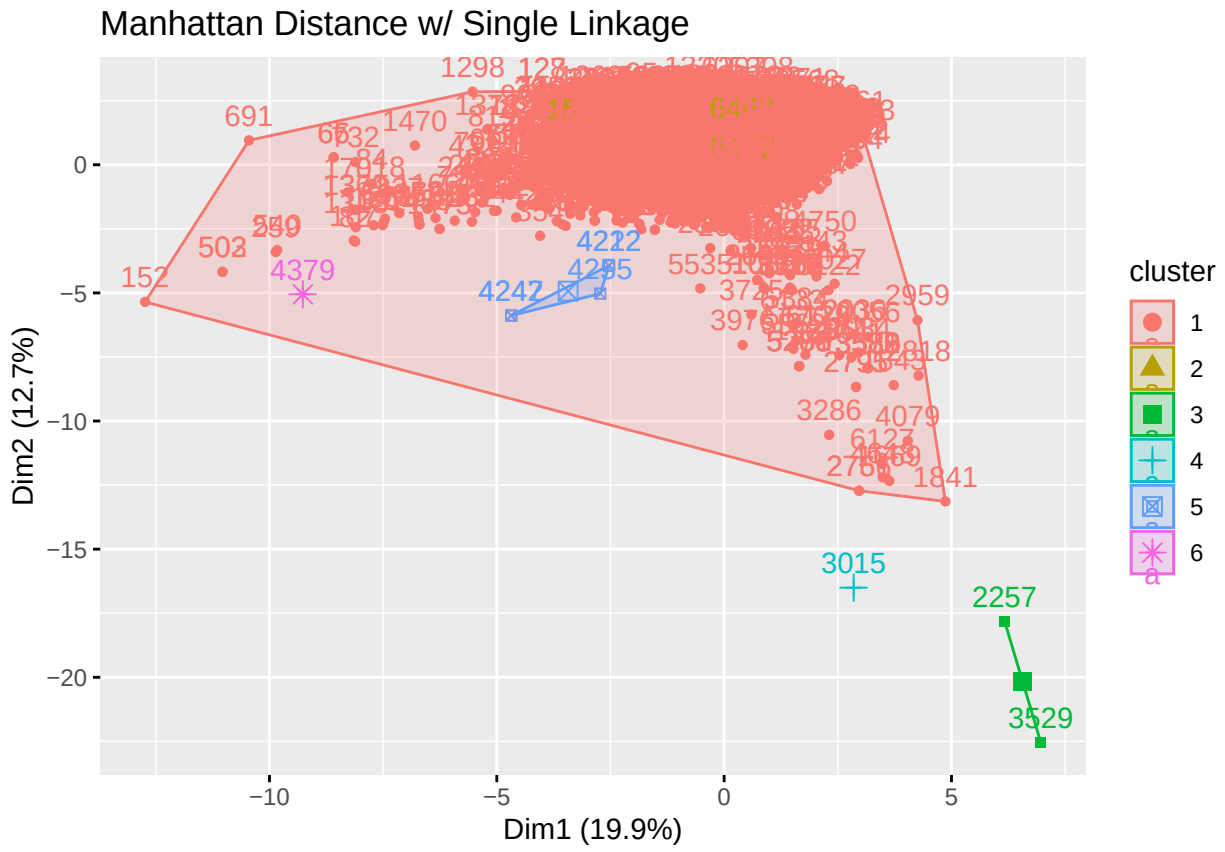

```
fviz_cluster(list(data = wine_variables, cluster = h2)) + labs(title = "Euclidean Distance w/ Single Li
```




```
fviz_cluster(list(data = wine_variables, cluster = h3)) + labs(title = "Manhattan Distance w/ Complete L
```




```
fviz_cluster(list(data = wine_variables, cluster = h4)) + labs(title = "Manhattan Distance w/ Single Li
```



- c: Compare the k-means and HAC clustering by creating a cross tabulation between their labels.

```
wine_results <- data.frame(Status = wine_class, HAC = h3, Kmeans = wine_kmeans$cluster)

# Crosstab for HAC
wine_results %>% group_by(HAC) %>% select(HAC, Status) %>% table()
```

```
##    Status
## HAC    1    0
##   1 1583 4819
##   2    9    0
##   3    3   65
##   4    2    6
##   5    0    3
##   6    0    5
```

```
# Crosstab for K Means
wine_results %>% group_by(Kmeans) %>% select(Kmeans, Status) %>% table()
```

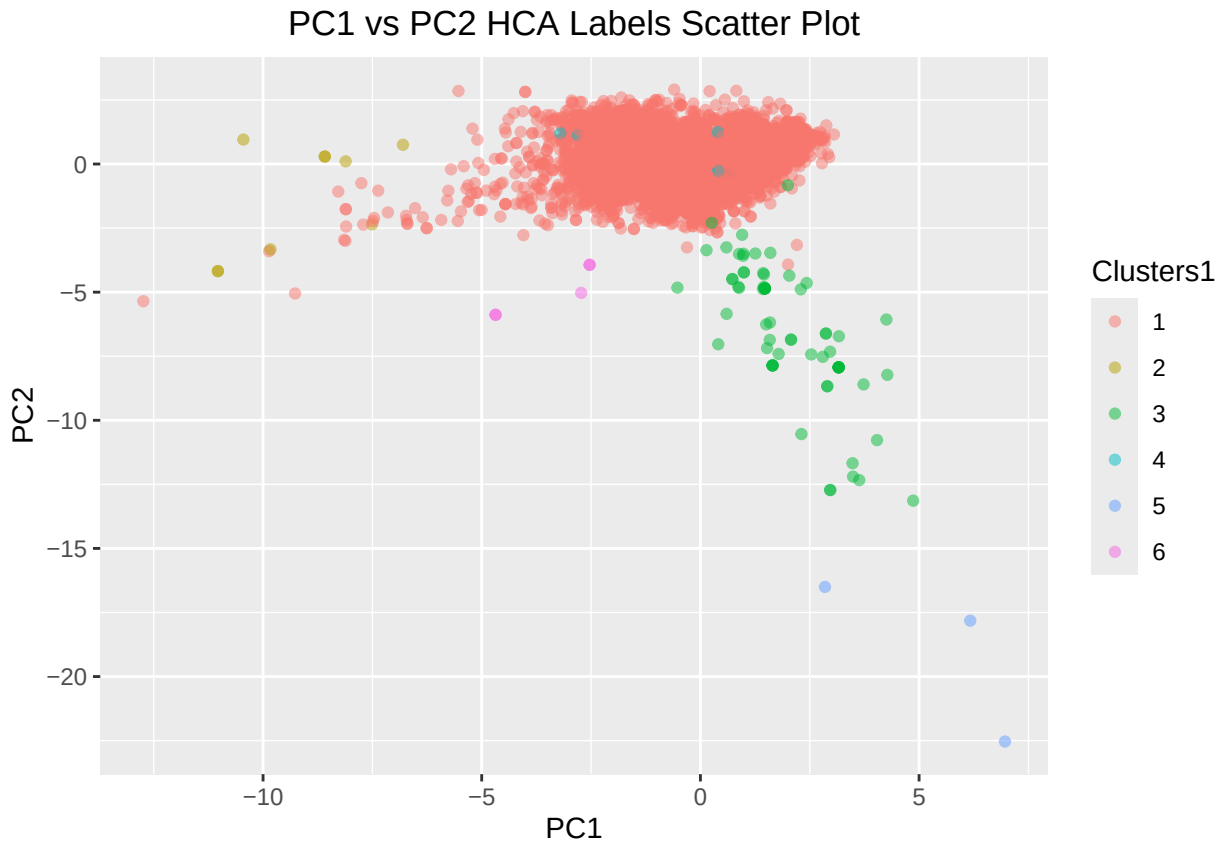
```
##       Status
## Kmeans    1    0
##      1   77 2119
##      2  821  104
##      3    3   62
```

```
##       4   46 1998
##       5  111  533
##       6  539   82
```

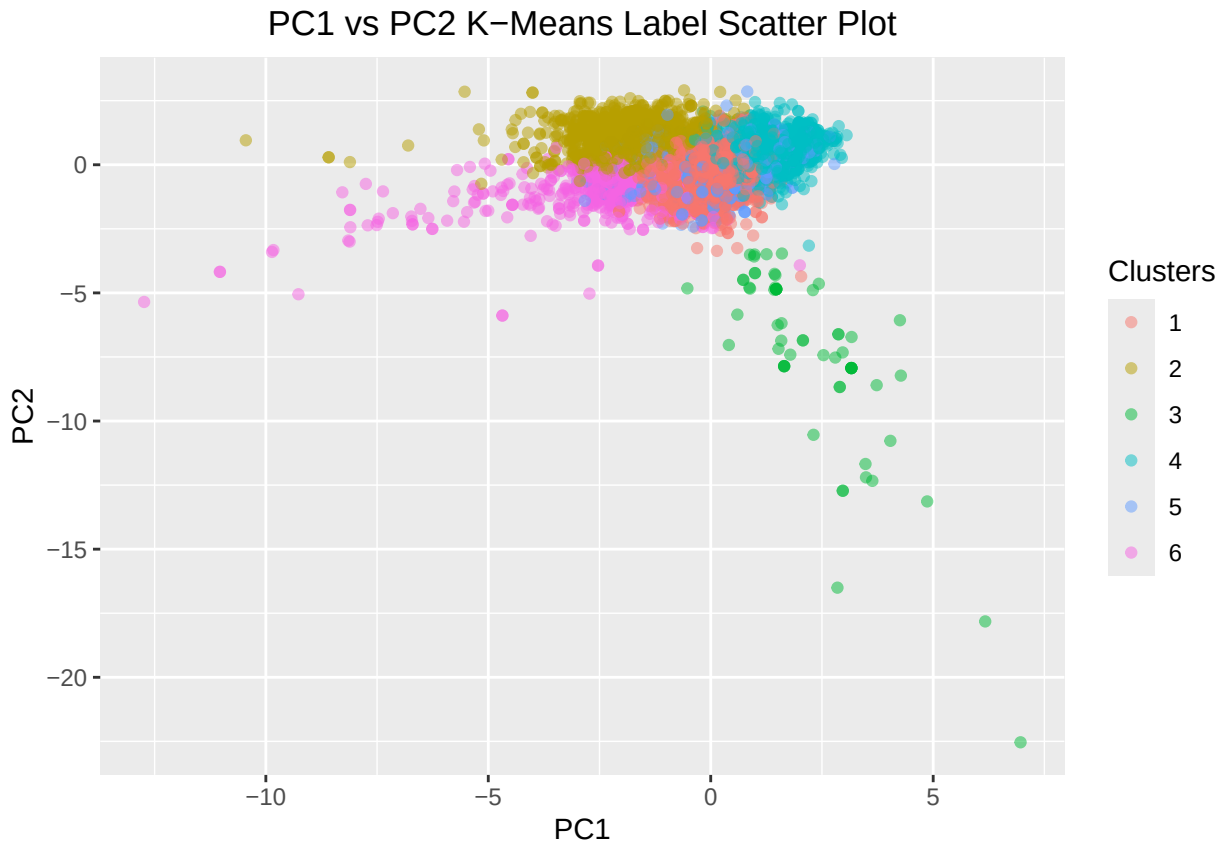
K-means seemed to get relatively better clusters on the data according to both cross tabulations. HAC placed most of the data in the first cluster while struggling to cluster red wine from white wine. Most of the observations were grouped into 1 cluster. K-means on the other hand grouped observation into multiple clusters, creating more nuanced clusters.

- d: For comparison – use PCA to visualize the data in a scatter plot. Create 3 separate plots: use the color of the points to show (1) the type label, (2) the k-means cluster labels and (3) the HAC cluster labels.

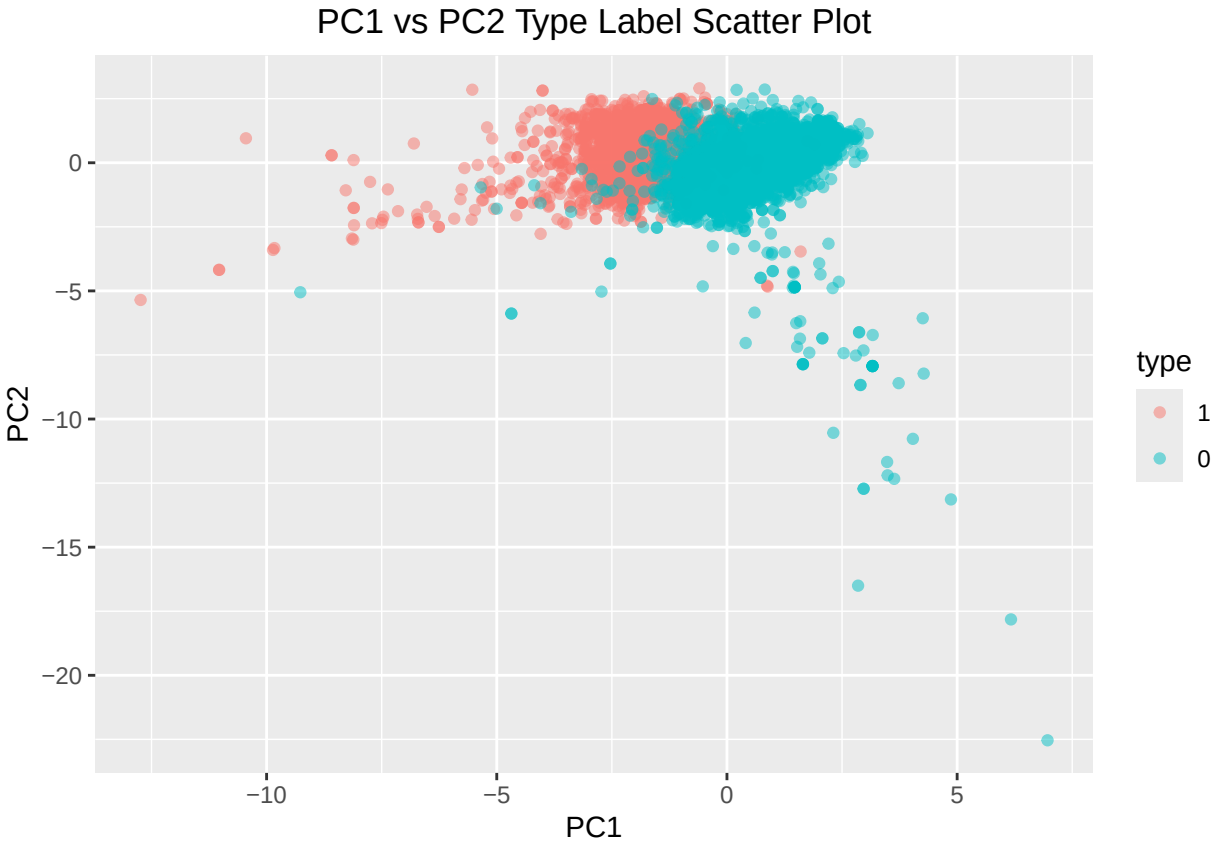
```
# Assign clusters as a new column HCA
rotated_data$Clusters1 = as.factor(h3)
# Plot and color by labels
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = Clusters1)) + geom_point(alpha = 0.5) + labs(ti
```




```
# Assign clusters as a new column k-means
rotated_data$Clusters = as.factor(wine_kmeans$cluster)
# Plot and color by labels
ggplot(data = rotated_data, aes(x = PC1, y = PC2, col = Clusters)) + geom_point(alpha = 0.5) + labs(titl
```




```
# scatter plot with PCs 1 and 2 by type (1 = red wine, 0 = white wine)
ggplot(wine.pc, aes(PC1, PC2, color = type)) + geom_point(alpha = 0.5) + labs(title = "PC1 vs PC2 Type
```



- e: Consider the results of C and D and explain the differences between the clustering results in terms of how the algorithms work.

  In k-means clustering, the algorithm arbitrarily chooses k data points as initial centroids, calculating the distance between each data point to other centroids. Once the distance is calculated, the data point is assigned to its closest centroid. After these new clusters are formed, the new centroid is calculated for the new cluster, using different statistical values to determine the new centroid value. These steps are repeated until there no significant change from the iteration or a stopping criterion was met. In HAC, each data point is considered its own cluster. First, the distance is calculated from one cluster to every other cluster. The cluster then links with the closest cluster, creating a larger, new cluster. There are varying linkage methods available, like how there are varying ways to calculate the new centroids in the k-means algorithm. New distances are measured with the new clusters and the remaining clusters. Again, the clusters are merged based on the linkage methods and this process repeats until all clusters have been merged into one, when the HAC can be represented as a dendogram. This dendogram is cut to determine the level, or number of clusters, desired.

Problem 4:

Back to the Starwars data from a previous assignment! Remember that the variable that lists the actual names and the variables that are actually lists will be a problem, so remove them (name, films, vehicles, starships). Make sure to double check the types of the variables, i.e., that they are numerical or factors as you expect.

```r
data(starwars)
star_data <- select(starwars, -c("films", "vehicles", "starships", "name"))
head(star_data)
```

```
## # A tibble: 6 x 10
##   height  mass hair_color skin_color eye_color birth_year sex   gender homeworld
##    <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>  <chr>
## 1    172    77 blond      fair       blue            19   male  mascu~ Tatooine
## 2    167    75 <NA>       gold       yellow         112   none  mascu~ Tatooine
## 3     96    32 <NA>       white, bl~ red             33   none  mascu~ Naboo
## 4    202   136 none       white      yellow          41.9 male  mascu~ Tatooine
## 5    150    49 brown      light      brown           19   fema~ femin~ Alderaan
## 6    178   120 brown, gr~ light      blue            52   male  mascu~ Tatooine
## # i 1 more variable: species <chr>
```

```r
summary(star_data)
```

```
##      height           mass          hair_color         skin_color
##  Min.   : 66.0   Min.   :  15.00   Length:87          Length:87
##  1st Qu.:167.0   1st Qu.:  55.60   Class :character   Class :character
##  Median :180.0   Median :  79.00   Mode  :character   Mode  :character
##  Mean   :174.6   Mean   :  97.31
##  3rd Qu.:191.0   3rd Qu.:  84.50
##  Max.   :264.0   Max.   :1358.00
##  NA's   :6       NA's   :28
##   eye_color           birth_year         sex              gender
##  Length:87          Min.   :  8.00   Length:87          Length:87
##  Class :character   1st Qu.: 35.00   Class :character   Class :character
##  Mode  :character   Median : 52.00   Mode  :character   Mode  :character
##                     Mean   : 87.57
##                     3rd Qu.: 72.00
##                     Max.   :896.00
##                     NA's   :44
##   homeworld          species
##  Length:87          Length:87
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
##
```

```r
# remove rows with missing values
star_data <- drop_na(star_data)
dim(star_data)
```

```
## [1] 29 10
```

```r
summary(star_data)
```

```
##      height           mass         hair_color         skin_color
```

```
##  Min.   : 88.0   Min.   : 20.00   Length:29          Length:29
##  1st Qu.:172.0   1st Qu.: 75.00   Class :character   Class :character
##  Median :180.0   Median : 79.00   Mode  :character   Mode  :character
##  Mean   :178.7   Mean   : 77.77
##  3rd Qu.:188.0   3rd Qu.: 83.00
##  Max.   :228.0   Max.   :136.00
##   eye_color           birth_year         sex               gender
##  Length:29          Min.   :  8.00   Length:29          Length:29
##  Class :character   1st Qu.: 31.00   Class :character   Class :character
##  Mode  :character   Median : 46.00   Mode  :character   Mode  :character
##                     Mean   : 51.29
##                     3rd Qu.: 57.00
##                     Max.   :200.00
##   homeworld           species
##  Length:29          Length:29
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##
```

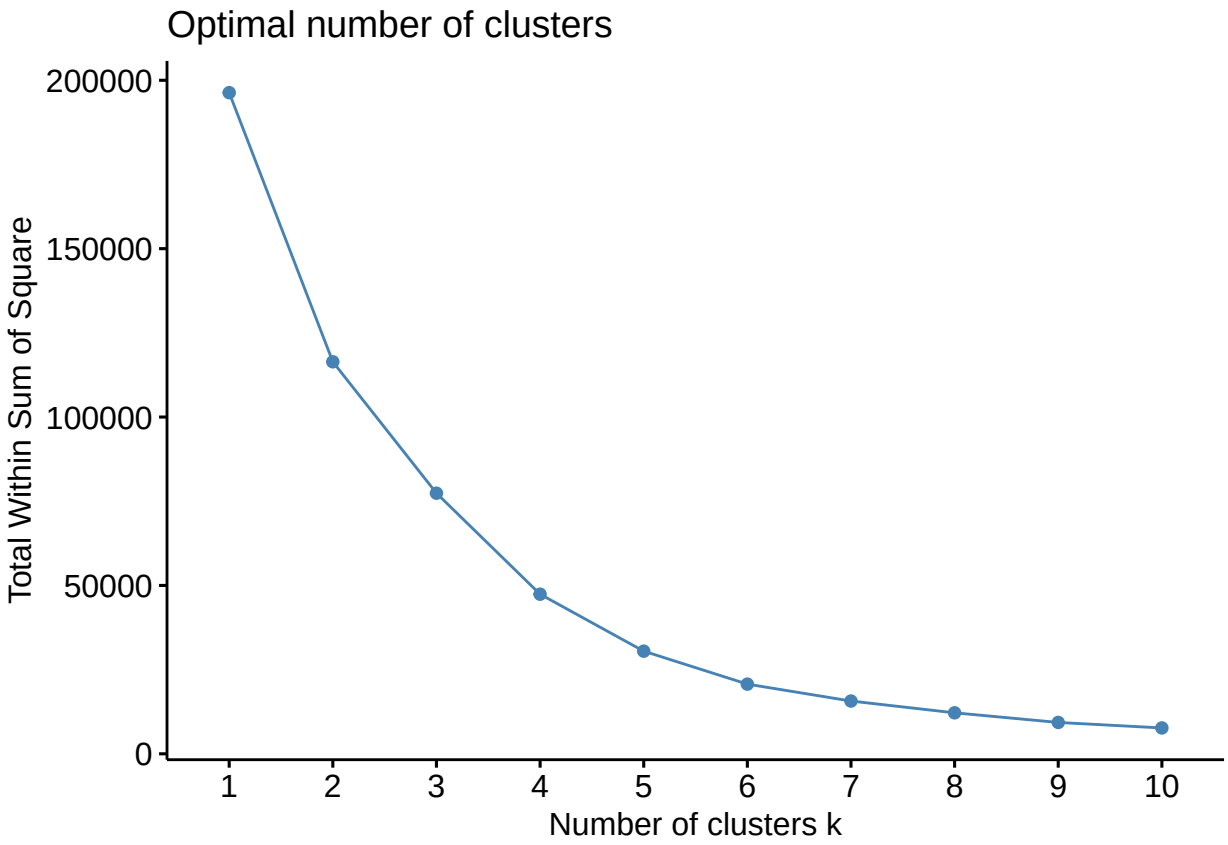
```
star_data <- star_data %>% mutate(across(where(is.character), as.factor))
star_class <- star_data$gender
star_variables <- star_data %>% select(-c(gender))
```

- a: Use hierarchical agglomerative clustering to cluster the Starwars data. This time we can leave the categorical variables in place, because we will use the gower metric from daisy in the cluster library to get the distances. Use average linkage. Determine the best number of clusters.

```
# Load library
library(cluster)
# Elbow
fviz_nbclust(star_variables, FUN = hcut, method = "wss")
```

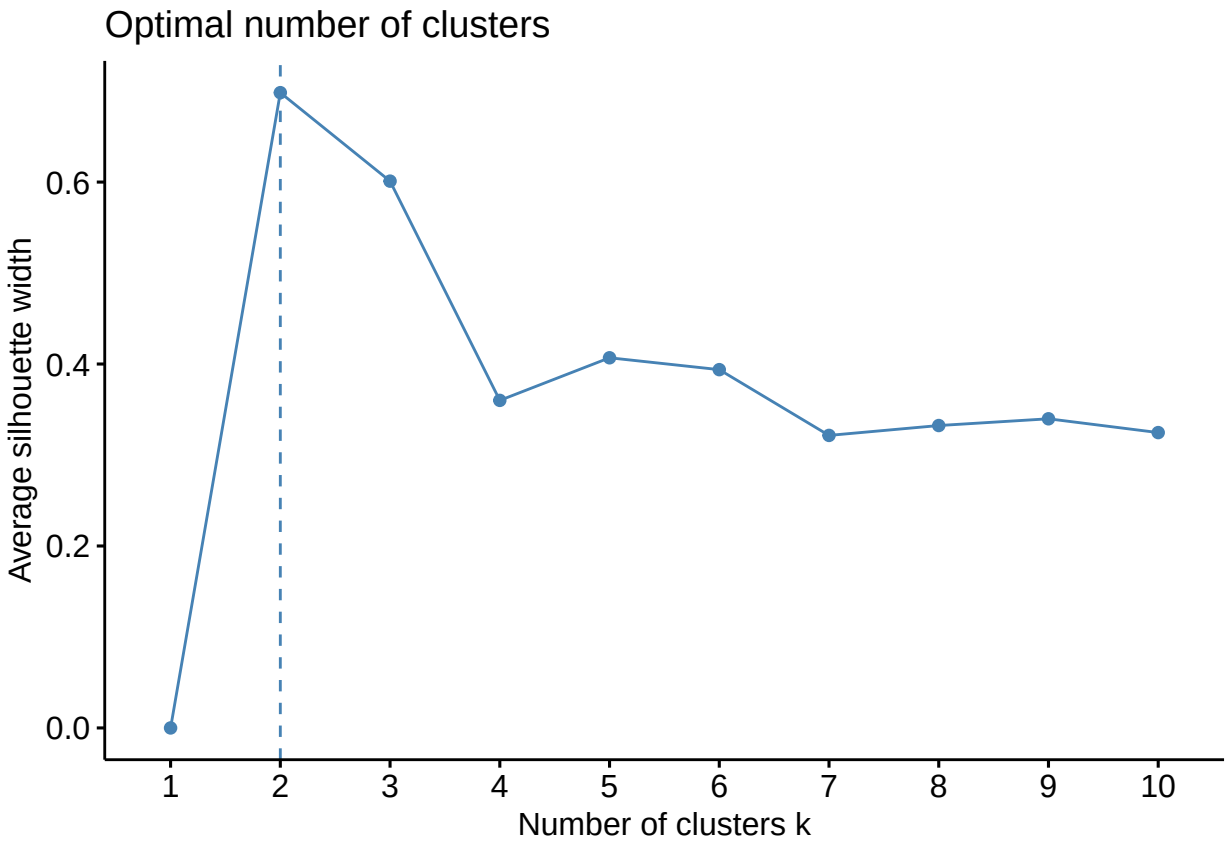
```
## Warning in stats::dist(x): NAs introduced by coercion
```

```
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
## Warning in stats::dist(x, method = method, ...): NAs introduced by coercion
```




```
# Silhouette
fviz_nbclust(star_variables, FUN = hcut, method = "silhouette")
```

```
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
## Warning in stats::dist(x): NAs introduced by coercion
```




```
# Pass data frame directly with metric = gower to determine distance for mixed variables
cat_dist_mat <- daisy(star_variables, metric = "gower")
# Result is a dissimilarity matrix
cat_dist_mat
```

```
## Dissimilarities :
##             1          2          3          4          5          6          7
## 2  0.42690858
## 3  0.59983580 0.69341104
## 4  0.28726909 0.37355153 0.55377155
## 5  0.35700830 0.53519003 0.27523490 0.27853665
## 6  0.35166199 0.40857992 0.39594223 0.27687694 0.36954992
## 7  0.36326058 0.52556930 0.62976305 0.49170031 0.57675036 0.47104030
## 8  0.03265571 0.39425287 0.63249150 0.27048640 0.36315938 0.34766038 0.35353864
## 9  0.62715916 0.69067204 0.78255052 0.57743797 0.61842672 0.71994162 0.70834303
## 10 0.34834314 0.52300989 0.39262338 0.49765611 0.47155515 0.34243922 0.35399790
## 11 0.33607804 0.53844959 0.48829479 0.50992121 0.46537470 0.46320300 0.36369048
## 12 0.48440579 0.44036530 0.63279112 0.51125821 0.57977844 0.49694741 0.47035156
## 13 0.35044671 0.52090631 0.50583807 0.50031444 0.58187631 0.23211806 0.35003335
## 14 0.62399995 0.48242252 0.77939131 0.57236305 0.72637863 0.60567130 0.59870234
## 15 0.45727285 0.52519129 0.50155309 0.49666290 0.57817004 0.23582432 0.46537470
## 16 0.34804666 0.41219525 0.50343801 0.27255576 0.35088830 0.35199496 0.46348978
## 17 0.58038337 0.51319188 0.73577472 0.59894921 0.68970649 0.56873233 0.57214924
## 18 0.68318566 0.77676091 0.52779431 0.74823253 0.69191810 0.59040321 0.71311291
## 19 0.61214936 0.65858506 0.15834531 0.52531130 0.26740957 0.38500844 0.59495359
## 20 0.60423680 0.52210107 0.73855535 0.62156541 0.70278405 0.59931810 0.58009658
```

```
## 21 0.58106470 0.41540407 0.73645605 0.59740809 0.68344337 0.58309729 0.56572079
## 22 0.70928378 0.65571947 0.60030731 0.73124202 0.58560881 0.71230159 0.69612240
## 23 0.49451913 0.41167186 0.53879938 0.49843779 0.59689781 0.36507937 0.46459189
## 24 0.62322512 0.52833624 0.77861647 0.63097519 0.72560379 0.60872788 0.59329787
## 25 0.57286364 0.52071161 0.72825499 0.61916735 0.70417750 0.56451263 0.58344565
## 26 0.59963567 0.77781740 0.60089457 0.62648810 0.47278610 0.61217729 0.69669255
## 27 0.59833231 0.77871311 0.58136460 0.63907362 0.47323538 0.61087393 0.71506511
## 28 0.40090597 0.54000724 0.55629732 0.52359857 0.61439575 0.39024015 0.37097872
## 29 0.48228939 0.52806827 0.52656963 0.49578658 0.58466806 0.25131705 0.45236214
##             8          9         10         11         12         13         14
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9  0.59450346
## 10 0.35097924 0.61214936
## 11 0.36641894 0.62758906 0.12655081
## 12 0.49055687 0.70531495 0.48784152 0.48166108
## 13 0.34887566 0.72115689 0.22777407 0.35087661 0.48705158
## 14 0.59134425 0.67174158 0.60899015 0.62442985 0.62460945 0.60688657
## 15 0.46427175 0.72544187 0.33782955 0.45770275 0.48334531 0.22803976 0.61117155
## 16 0.35127571 0.61244584 0.45400018 0.45958767 0.47828578 0.45474280 0.49817552
## 17 0.56338345 0.71344246 0.56537356 0.58081326 0.59488175 0.56803189 0.48806103
## 18 0.71584137 0.75478927 0.47597325 0.57164466 0.71614099 0.58918793 0.75163006
## 19 0.59766551 0.74297916 0.38066445 0.50146814 0.50591817 0.48622372 0.73981995
## 20 0.58499105 0.71066183 0.59497412 0.60466669 0.49106117 0.58942227 0.49480421
## 21 0.57670675 0.71276113 0.57399140 0.58149459 0.46940567 0.57664973 0.48853711
## 22 0.70591104 0.84890987 0.70319570 0.59860256 0.71184889 0.70240575 0.62352844
## 23 0.46186343 0.68819558 0.36839822 0.49494903 0.47313789 0.36629464 0.48480489
## 24 0.59440083 0.67060071 0.60821531 0.62365501 0.36804757 0.60611174 0.61416769
## 25 0.57090318 0.72096219 0.56595569 0.57329354 0.60935276 0.56674563 0.49558076
## 26 0.60578675 0.73721150 0.71418252 0.70800208 0.69856322 0.60228146 0.73983933
## 27 0.60668246 0.75674147 0.71287915 0.70669871 0.71809319 0.60097809 0.75358226
## 28 0.37591310 0.67069764 0.27478505 0.40133586 0.47906176 0.27268148 0.61790218
## 29 0.46714873 0.70042533 0.35808418 0.48271928 0.46785258 0.24295378 0.60120131
##            15         16         17         18         19         20         21
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16 0.44950397
```

```
## 17 0.56755496 0.45455893
## 18 0.58490296 0.69789899 0.70801348
## 19 0.49204126 0.49015634 0.70992691 0.56736567
## 20 0.59523981 0.48224377 0.36868101 0.71079411 0.58787288
## 21 0.57141090 0.45524026 0.45880941 0.70869481 0.71275771 0.47567848
## 22 0.70028679 0.58729075 0.58801371 0.79476829 0.57184706 0.58269248 0.58535509
## 23 0.25946850 0.36869470 0.46969132 0.62214924 0.49922801 0.47960910 0.46900999
## 24 0.61039671 0.60851179 0.60031301 0.75085523 0.73904511 0.59561668 0.48660486
## 25 0.57045190 0.46440031 0.46466258 0.70049375 0.71646141 0.37045361 0.47328042
## 26 0.59857519 0.59351567 0.71011164 0.79535555 0.58513273 0.70016078 0.69574667
## 27 0.59727183 0.59221230 0.70996568 0.77582558 0.57889641 0.71274631 0.71064701
## 28 0.38807756 0.49730375 0.60404751 0.63964719 0.51672596 0.60028165 0.59761905
## 29 0.24723876 0.46757606 0.57623552 0.60991949 0.49017287 0.58642698 0.56980706
##            22         23         24         25         26         27         28
## 2
## 3
## 4
## 5
## 6
## 7
## 8
## 9
## 10
## 11
## 12
## 13
## 14
## 15
## 16
## 17
## 18
## 19
## 20
## 21
## 22
## 23 0.60515873
## 24 0.73386471 0.57698185
## 25 0.60248472 0.47721105 0.60591703
## 26 0.56884122 0.71568259 0.73327746 0.72458265
## 27 0.57449827 0.73521255 0.75280743 0.72327928 0.01952997
## 28 0.73376779 0.35849412 0.45611544 0.60582011 0.73318053 0.75271050
## 29 0.70404009 0.34556308 0.58538018 0.58594463 0.59234173 0.61187169 0.36306103
##
## Metric :  mixed ;  Types = I, I, N, N, N, I, N, N, N
## Number of objects : 29
```

```
# Determine assembly/agglomeration method and run hclust (average uses mean)
hfit <- hclust(cat_dist_mat, method = 'average')
hfit
```

```
##
## Call:
## hclust(d = cat_dist_mat, method = "average")
##
```

```
## Cluster method   : average
## Number of objects: 29
```

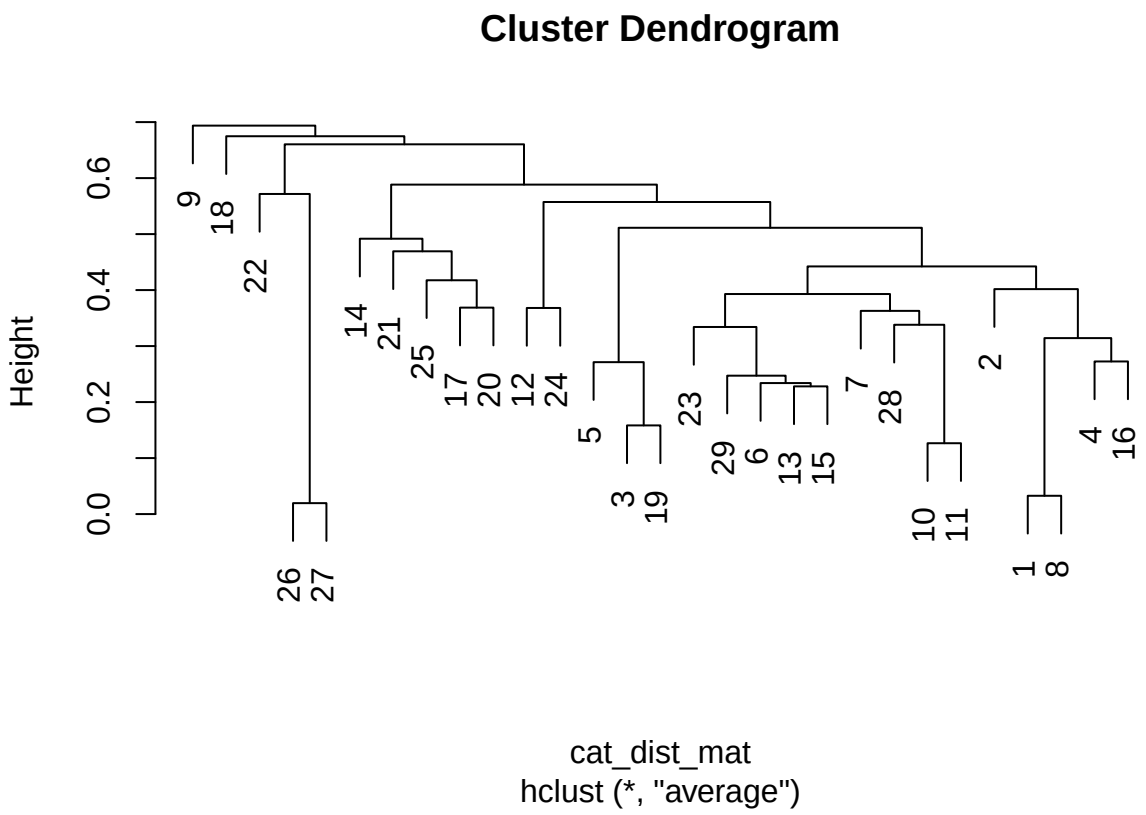
```
# Build tree with 3 clusters
h <- cutree(hfit, k=3)
h
```

```
##  [1] 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1
```

The best number of clusters is 3.

- b: Produce the dendogram for (a). How might an anomaly show up in a dendogram? Do you see a Starwars character who does not seem to fit in easily? What is the advantage of considering anomalies this way as opposed to looking for unusual values relative to the mean and standard deviations, as we considered earlier in the course? Disadvantages?

```
plot(hfit)
```



An anamoly may show up in a dendogram in several ways. The most obvious way to detect anamolies, or outliers, would be to identify any unusual clusters that emerge at certain distances and deviating from the expected pattern of clustering. This could indicate an outlier or an anamoly that doesn't fit well with the remaining data. Another way to stating this would be to identify a cluster that remains isolated at high distances. Given the ways to detect anamolies in a dendogram and based on the dendogram generated, observations 26 and 27 do not seem to fit in easily. These two observations are in close proximity to each other and easily forms a cluster, however, in reference to the other clusters, this cluster is quite distanced from

the nearest cluster. Therefore, I'd state these characters do not seem to fit easily. A dendrogram provides a hierarchical representation of the data, showing how observations are related to each other. This context is lost when focusing solely on mean and standard deviations. Anomalies in a dendrogram can reveal patterns and relationships that might not be apparent when examining individual values because of the inherent hierarchial nature of dendrograms, allowing for the identification of clusters and sub-clusters. This can reveal nested patterns and structures in the data that might not be apparent when examining individual values. The disadvantage of using dendograms, which are only generated after HAC has been performed only after which a distance matrix is caculated, is the computational requirements. As the distance to all points from all other points are required to cluster objects using HAC, there are more computations required to generate a dendogram as opposed to calulating the mean and standard deviation of a single attribute.

- c: Use dummy variables to make this data fully numeric and then use k-means to cluster. Choose the best number of clusters.

```
# dummy variable for each non-classifier, categorical variables
dummy_star <- dummyVars(gender ~ ., data = star_data)

# Transform dummy variables into dataframe
dummies_star <- as.data.frame(predict(dummy_star, newdata = star_data))
```

```
## Warning in model.frame.default(Terms, newdata, na.action = na.action, xlev =
## object$lvls): variable 'gender' is not a factor
```

```
head(dummies_star)
```

```
##   height mass hair_color.auburn, white hair_color.black hair_color.blond
## 1    172   77                       0                0                1
## 2    202  136                       0                0                0
## 3    150   49                       0                0                0
## 4    178  120                       0                0                0
## 5    165   75                       0                0                0
## 6    183   84                       0                1                0
##   hair_color.brown hair_color.brown, grey hair_color.grey hair_color.none
## 1                0                      0               0               0
## 2                0                      0               0               1
## 3                1                      0               0               0
## 4                0                      1               0               0
## 5                1                      0               0               0
## 6                0                      0               0               0
##   hair_color.white skin_color.blue skin_color.brown skin_color.brown mottle
## 1                0               0                0                       0
## 2                0               0                0                       0
## 3                0               0                0                       0
## 4                0               0                0                       0
## 5                0               0                0                       0
## 6                0               0                0                       0
##   skin_color.dark skin_color.fair skin_color.green skin_color.light
## 1               0               1                0                0
## 2               0               0                0                0
## 3               0               0                0                1
## 4               0               0                0                1
## 5               0               0                0                1
```

```
## 6               0              0               0               1
##   skin_color.orange skin_color.pale skin_color.red skin_color.tan
## 1                 0              0              0              0
## 2                 0              0              0              0
## 3                 0              0              0              0
## 4                 0              0              0              0
## 5                 0              0              0              0
## 6                 0              0              0              0
##   skin_color.unknown skin_color.white skin_color.yellow eye_color.black
## 1                  0                0                 0               0
## 2                  0                1                 0               0
## 3                  0                0                 0               0
## 4                  0                0                 0               0
## 5                  0                0                 0               0
## 6                  0                0                 0               0
##   eye_color.blue eye_color.blue-gray eye_color.brown eye_color.hazel
## 1              1                   0               0              0
## 2              0                   0               0              0
## 3              0                   0               1              0
## 4              1                   0               0              0
## 5              1                   0               0              0
## 6              0                   0               1              0
##   eye_color.orange eye_color.red eye_color.yellow birth_year sex.female
## 1                0             0                0       19.0          0
## 2                0             0                1       41.9          0
## 3                0             0                0       19.0          1
## 4                0             0                0       52.0          0
## 5                0             0                0       47.0          1
## 6                0             0                0       24.0          0
##   sex.male homeworld.Alderaan homeworld.Bespin homeworld.Cerea
## 1        1                  0                0               0
## 2        1                  0                0               0
## 3        0                  1                0               0
## 4        1                  0                0               0
## 5        0                  0                0               0
## 6        1                  0                0               0
##   homeworld.Concord Dawn homeworld.Corellia homeworld.Dathomir homeworld.Dorin
## 1                      0                 0                 0               0
## 2                      0                 0                 0               0
## 3                      0                 0                 0               0
## 4                      0                 0                 0               0
## 5                      0                 0                 0               0
## 6                      0                 0                 0               0
##   homeworld.Endor homeworld.Haruun Kal homeworld.Kamino homeworld.Kashyyyk
## 1               0                    0                0                 0
## 2               0                    0                0                 0
## 3               0                    0                0                 0
## 4               0                    0                0                 0
## 5               0                    0                0                 0
## 6               0                    0                0                 0
##   homeworld.Mirial homeworld.Mon Cala homeworld.Naboo homeworld.Ryloth
## 1                0                  0               0               0
## 2                0                  0               0               0
## 3                0                  0               0               0
```

```
## 4                0                  0               0                0
## 5                0                  0               0                0
## 6                0                  0               0                0
##   homeworld.Serenno homeworld.Socorro homeworld.Stewjon homeworld.Tatooine
## 1                 0                 0                 0                 1
## 2                 0                 0                 0                 1
## 3                 0                 0                 0                 0
## 4                 0                 0                 0                 1
## 5                 0                 0                 0                 1
## 6                 0                 0                 0                 1
##   homeworld.Trandosha species.Cerean species.Ewok species.Gungan species.Human
## 1                   0              0             0              0             1
## 2                   0              0             0              0             1
## 3                   0              0             0              0             1
## 4                   0              0             0              0             1
## 5                   0              0             0              0             1
## 6                   0              0             0              0             1
##   species.Kel Dor species.Mirialan species.Mon Calamari species.Trandoshan
## 1               0                0                    0                  0
## 2               0                0                    0                  0
## 3               0                0                    0                  0
## 4               0                0                    0                  0
## 5               0                0                    0                  0
## 6               0                0                    0                  0
##   species.Twi'lek species.Wookiee species.Zabrak
## 1               0              0              0
## 2               0              0              0
## 3               0              0              0
## 4               0              0              0
## 5               0              0              0
## 6               0              0              0
```

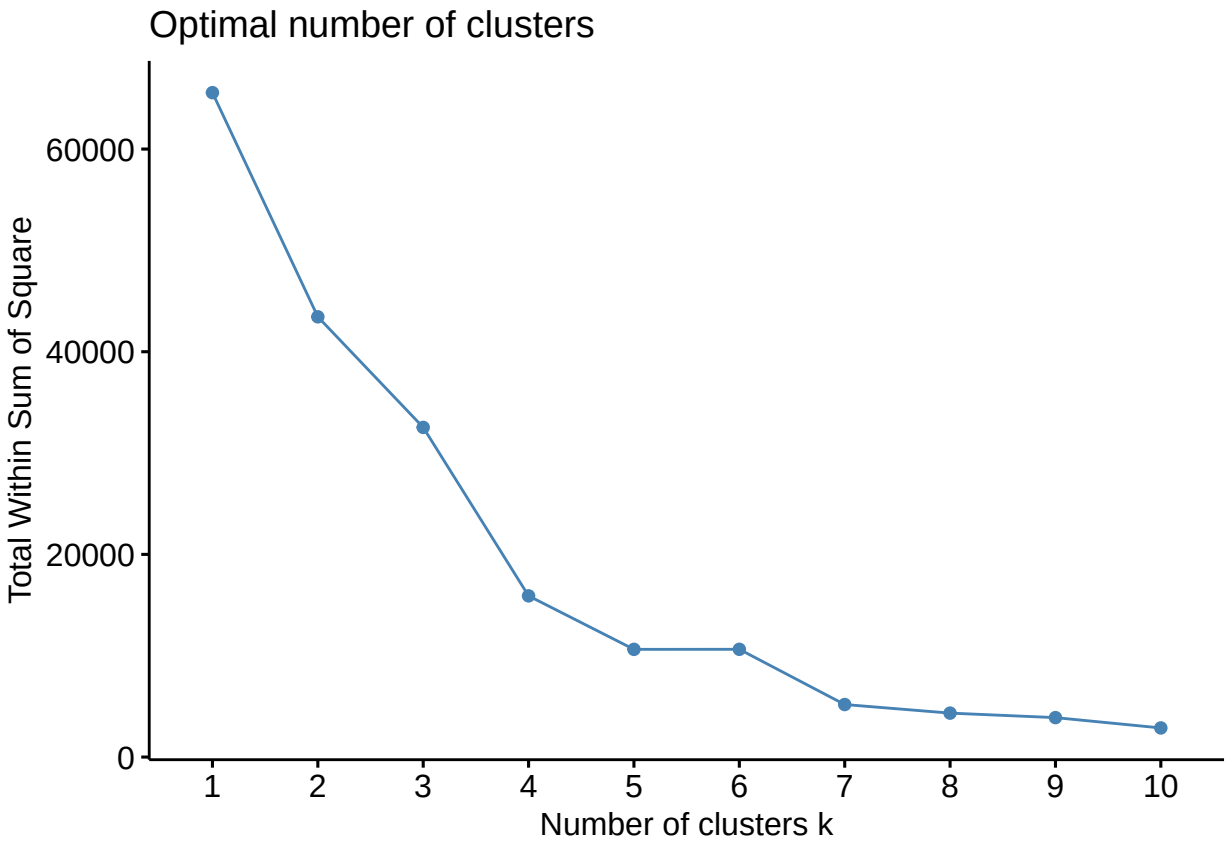
```
# Add target variable
dummies_star$gender <- as.factor(star_data$gender)
head(dummies_star)
```

```
##   height mass hair_color.auburn, white hair_color.black hair_color.blond
## 1    172   77                       0                0                1
## 2    202  136                       0                0                0
## 3    150   49                       0                0                0
## 4    178  120                       0                0                0
## 5    165   75                       0                0                0
## 6    183   84                       0                1                0
##   hair_color.brown hair_color.brown, grey hair_color.grey hair_color.none
## 1                0                      0               0               0
## 2                0                      0               0               1
## 3                1                      0               0               0
## 4                0                      1               0               0
## 5                1                      0               0               0
## 6                0                      0               0               0
##   hair_color.white skin_color.blue skin_color.brown skin_color.brown mottle
## 1                0               0                0                       0
## 2                0               0                0                       0
## 3                0               0                0                       0
```

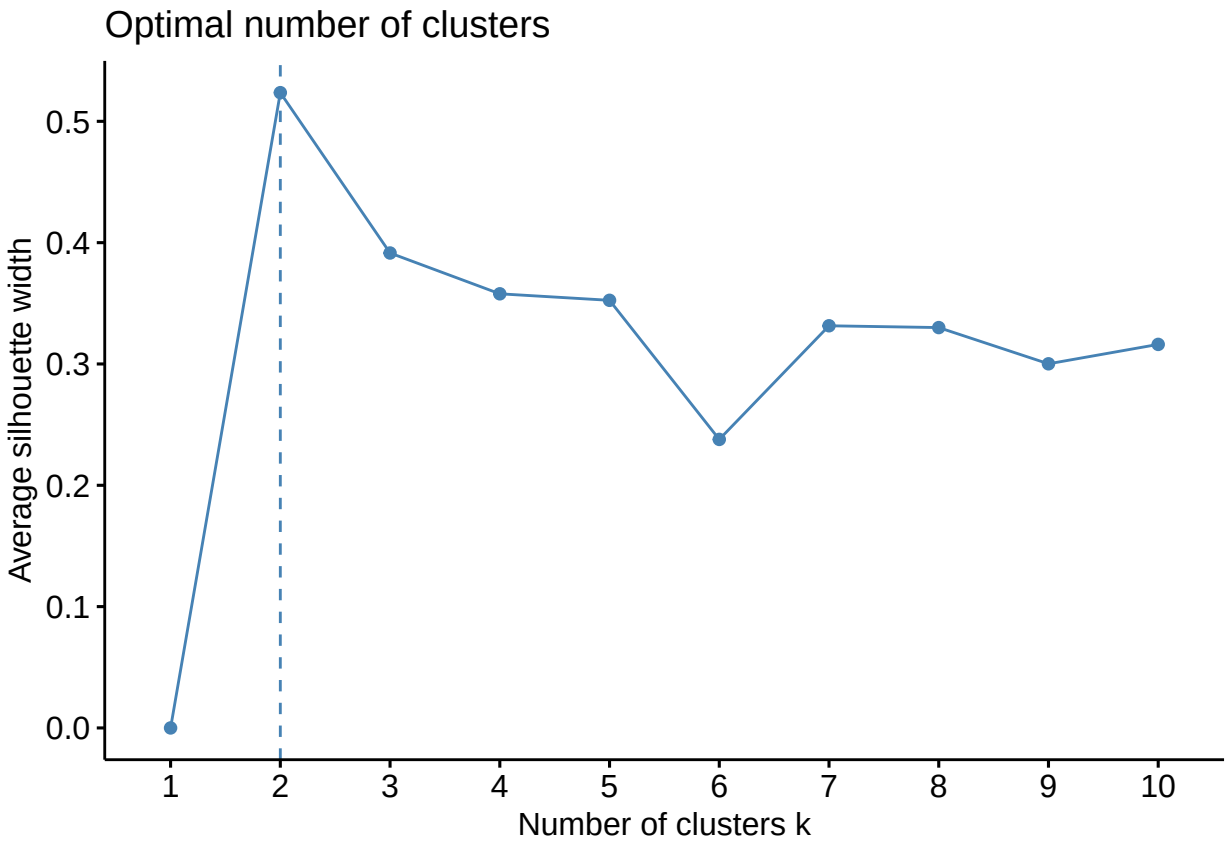
```
## 4                0              0               0                      0
## 5                0              0               0                      0
## 6                0              0               0                      0
##   skin_color.dark skin_color.fair skin_color.green skin_color.light
## 1               0              1                0                0
## 2               0              0                0                0
## 3               0              0                0                1
## 4               0              0                0                1
## 5               0              0                0                1
## 6               0              0                0                1
##   skin_color.orange skin_color.pale skin_color.red skin_color.tan
## 1                 0              0              0              0
## 2                 0              0              0              0
## 3                 0              0              0              0
## 4                 0              0              0              0
## 5                 0              0              0              0
## 6                 0              0              0              0
##   skin_color.unknown skin_color.white skin_color.yellow eye_color.black
## 1                  0                0                 0               0
## 2                  0                1                 0               0
## 3                  0                0                 0               0
## 4                  0                0                 0               0
## 5                  0                0                 0               0
## 6                  0                0                 0               0
##   eye_color.blue eye_color.blue-gray eye_color.brown eye_color.hazel
## 1              1                   0               0              0
## 2              0                   0               0              0
## 3              0                   0               1              0
## 4              1                   0               0              0
## 5              1                   0               0              0
## 6              0                   0               1              0
##   eye_color.orange eye_color.red eye_color.yellow birth_year sex.female
## 1                0             0                0       19.0          0
## 2                0             0                1       41.9          0
## 3                0             0                0       19.0          1
## 4                0             0                0       52.0          0
## 5                0             0                0       47.0          1
## 6                0             0                0       24.0          0
##   sex.male homeworld.Alderaan homeworld.Bespin homeworld.Cerea
## 1        1                  0                0               0
## 2        1                  0                0               0
## 3        0                  1                0               0
## 4        1                  0                0               0
## 5        0                  0                0               0
## 6        1                  0                0               0
##   homeworld.Concord Dawn homeworld.Corellia homeworld.Dathomir homeworld.Dorin
## 1                      0                 0                 0               0
## 2                      0                 0                 0               0
## 3                      0                 0                 0               0
## 4                      0                 0                 0               0
## 5                      0                 0                 0               0
## 6                      0                 0                 0               0
##   homeworld.Endor homeworld.Haruun Kal homeworld.Kamino homeworld.Kashyyyk
## 1               0                    0                0                 0
```

```
## 2                0                    0                0                    0
## 3                0                    0                0                    0
## 4                0                    0                0                    0
## 5                0                    0                0                    0
## 6                0                    0                0                    0
##   homeworld.Mirial homeworld.Mon Cala homeworld.Naboo homeworld.Ryloth
## 1                0                   0                0                 0
## 2                0                   0                0                 0
## 3                0                   0                0                 0
## 4                0                   0                0                 0
## 5                0                   0                0                 0
## 6                0                   0                0                 0
##   homeworld.Serenno homeworld.Socorro homeworld.Stewjon homeworld.Tatooine
## 1                 0                 0                0                  1
## 2                 0                 0                0                  1
## 3                 0                 0                0                  0
## 4                 0                 0                0                  1
## 5                 0                 0                0                  1
## 6                 0                 0                0                  1
##   homeworld.Trandosha species.Cerean species.Ewok species.Gungan species.Human
## 1                   0              0            0              0             1
## 2                   0              0            0              0             1
## 3                   0              0            0              0             1
## 4                   0              0            0              0             1
## 5                   0              0            0              0             1
## 6                   0              0            0              0             1
##   species.Kel Dor species.Mirialan species.Mon Calamari species.Trandoshan
## 1               0                0                    0                  0
## 2               0                0                    0                  0
## 3               0                0                    0                  0
## 4               0                0                    0                  0
## 5               0                0                    0                  0
## 6               0                0                    0                  0
##   species.Twi'lek species.Wookiee species.Zabrak    gender
## 1               0              0              0 masculine
## 2               0              0              0 masculine
## 3               0              0              0  feminine
## 4               0              0              0 masculine
## 5               0              0              0  feminine
## 6               0              0              0 masculine
```

```
dummies_star_variables <- dummies_star %>% select(-c(gender))
dummies_star_target <- dummies_star$gender
# Set seed
set.seed(123)

# Elbow
fviz_nbclust(dummies_star_variables, kmeans, method = "wss")
```




```
# Silhouette
fviz_nbclust(dummies_star_variables, kmeans, method = "silhouette")
```




```
# Fit the data
star_kmeans <- kmeans(dummies_star_variables, centers = 3, nstart = 25)
# Display the kmeans object information
star_kmeans
```

```
## K-means clustering with 3 clusters of sizes 1, 2, 26
##
## Cluster means:
##     height      mass hair_color.auburn, white hair_color.black hair_color.blond
## 1 228.0000 112.00000               0.00000000        0.0000000       0.00000000
## 2 119.0000  34.50000               0.00000000        0.0000000       0.00000000
## 3 181.3462  79.78462               0.03846154        0.2307692       0.07692308
##   hair_color.brown hair_color.brown, grey hair_color.grey hair_color.none
## 1        1.0000000             0.00000000      0.00000000       0.0000000
## 2        1.0000000             0.00000000      0.00000000       0.0000000
## 3        0.1538462             0.03846154      0.03846154       0.3461538
##   hair_color.white skin_color.blue skin_color.brown skin_color.brown mottle
## 1       0.00000000      0.00000000              0.0              0.00000000
## 2       0.00000000      0.00000000              0.5              0.00000000
## 3       0.07692308      0.03846154              0.0              0.03846154
##   skin_color.dark skin_color.fair skin_color.green skin_color.light
## 1      0.00000000       0.0000000       0.00000000        0.0000000
## 2      0.00000000       0.0000000       0.00000000        0.5000000
## 3      0.07692308       0.2692308       0.03846154        0.1923077
##   skin_color.orange skin_color.pale skin_color.red skin_color.tan
## 1        0.00000000      0.00000000     0.00000000     0.00000000
```

```
## 2        0.00000000      0.00000000     0.00000000     0.00000000
## 3        0.07692308      0.07692308     0.03846154     0.03846154
##   skin_color.unknown skin_color.white skin_color.yellow eye_color.black
## 1                  1       0.00000000        0.00000000      0.00000000
## 2                  0       0.00000000        0.00000000      0.00000000
## 3                  0       0.03846154        0.07692308      0.03846154
##   eye_color.blue eye_color.blue-gray eye_color.brown eye_color.hazel
## 1      1.0000000          0.00000000       0.0000000      0.00000000
## 2      0.0000000          0.00000000       1.0000000      0.00000000
## 3      0.2692308          0.03846154       0.3076923      0.07692308
##   eye_color.orange eye_color.red eye_color.yellow birth_year sex.female
## 1       0.00000000    0.00000000        0.0000000  200.00000  0.0000000
## 2       0.00000000    0.00000000        0.0000000   13.50000  0.5000000
## 3       0.07692308    0.03846154        0.1538462   48.47308  0.1923077
##    sex.male homeworld.Alderaan homeworld.Bespin homeworld.Cerea
## 1 1.0000000                0.0       0.00000000      0.00000000
## 2 0.5000000                0.5       0.00000000      0.00000000
## 3 0.8076923                0.0       0.03846154      0.03846154
##   homeworld.Concord Dawn homeworld.Corellia homeworld.Dathomir homeworld.Dorin
## 1             0.00000000         0.00000000         0.00000000      0.00000000
## 2             0.00000000         0.00000000         0.00000000      0.00000000
## 3             0.03846154         0.07692308         0.03846154      0.03846154
##   homeworld.Endor homeworld.Haruun Kal homeworld.Kamino homeworld.Kashyyyk
## 1             0.0           0.00000000       0.00000000                  1
## 2             0.5           0.00000000       0.00000000                  0
## 3             0.0           0.03846154       0.03846154                  0
##   homeworld.Mirial homeworld.Mon Cala homeworld.Naboo homeworld.Ryloth
## 1       0.00000000         0.00000000       0.0000000       0.00000000
## 2       0.00000000         0.00000000       0.0000000       0.00000000
## 3       0.07692308         0.03846154       0.1153846       0.03846154
##   homeworld.Serenno homeworld.Socorro homeworld.Stewjon homeworld.Tatooine
## 1        0.00000000        0.00000000        0.00000000          0.0000000
## 2        0.00000000        0.00000000        0.00000000          0.0000000
## 3        0.03846154        0.03846154        0.03846154          0.2307692
##   homeworld.Trandosha species.Cerean species.Ewok species.Gungan species.Human
## 1          0.00000000     0.00000000          0.0     0.00000000     0.0000000
## 2          0.00000000     0.00000000          0.5     0.00000000     0.5000000
## 3          0.03846154     0.03846154          0.0     0.03846154     0.6538462
##   species.Kel Dor species.Mirialan species.Mon Calamari species.Trandoshan
## 1      0.00000000       0.00000000           0.00000000         0.00000000
## 2      0.00000000       0.00000000           0.00000000         0.00000000
## 3      0.03846154       0.07692308           0.03846154         0.03846154
##   species.Twi'lek species.Wookiee species.Zabrak
## 1      0.00000000               1     0.00000000
## 2      0.00000000               0     0.00000000
## 3      0.03846154               0     0.03846154
##
## Clustering vector:
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
##  3  3  2  3  3  3  3  3  1  3  3  3  3  3  3  3  3  2  3  3  3  3  3  3  3  3
## 27 28 29
##  3  3  3
##
## Within cluster sum of squares by cluster:
```

```
## [1]     0.00  2407.00 23209.59
##  (between_SS / total_SS =  60.9 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss"
## [6] "betweenss"    "size"         "iter"         "ifault"
```

```
# Display cluster plot
fviz_cluster(star_kmeans, data = dummies_star_variables)
```

- d: Compare the HAC and k-means clustering with a cross tabulation.

```
star_result <- data.frame(Status = star_data$gender, HAC = h, Kmeans = star_kmeans$cluster)

# Crosstab for HAC
star_result %>% group_by(HAC) %>% select(HAC, Status) %>% table()
```

```
##    Status
## HAC feminine masculine
##   1        6        21
##   2        0         1
##   3        0         1
```

```r
# Crosstab for K Means
star_result %>% group_by(Kmeans) %>% select(Kmeans, Status) %>% table()
```

```
##       Status
## Kmeans feminine masculine
##      1        0         1
##      2        1         1
##      3        5        21
```

Compared to the HAC, k-means clustering performed similarly, likely due to the small sample size of 29. After all the NA rows were removed from the data set, the sample size dropped to 29. However, based on the limited information in the cross tabulation, the k-means clustering performed marginally better than HAC. K-means was able to cluster one more observation into a different cluster than HAC did.