King Saud University

College of Computer and Information Sciences

Information Technology department

IT 326: Data Mining

Course Project

Water quality

Project Report: Data Mining techniques

<table>
<tr><td>Group#:</td><td colspan="2">4</td></tr>
<tr><td>Section#:</td><td colspan="2">52847</td></tr>
<tr><td rowspan="5">Group Members</td><td>Name</td><td>ID</td></tr>
<tr><td>Jumanah aldawsari</td><td></td></tr>
<tr><td>Lama alshaya</td><td></td></tr>
<tr><td>Nouf alsadhan</td><td></td></tr>
<tr><td>Aljawharah alzamil</td><td></td></tr>
</table>

[Pick the date]

1 Data Mining Technique

For our dataset, we will use both classification and clustering.

We applied classification technique(supervised) because the class label (is_safe) is provided in our dataset which indicates whether water is safe or not by predicting the amount of elements(barium,ammonia..etc) for each liter of water.

We will divide our dataset to training and testing data by applying decision tree. We will use the training data set to construct the classification model, and the test data set to determine the accuracy of the classification model so we can predict the new data class labels accurately.

In order to apply the clustering technique(unsupervised), the class label (is_safe) must be removed from our dataset.

We used the K-means technique, which that each cluster is represented by the center of the cluster.

K-means assign each object to the cluster with the nearest center point based on euclidean distance.

The used packages for both techniques: party – caret – factoextra – cluster - NbClust

The used methods: set.seed() – sample() – ctree() – table() – predict() – print() – plot()

confusionMatrix() – nrow() – subset() – scale() – kmeans() – fviz_cluster() – silhouette() – fviz_sillhouete() – fviz_NbClust() – lab()

2 Evaluation and Comparison

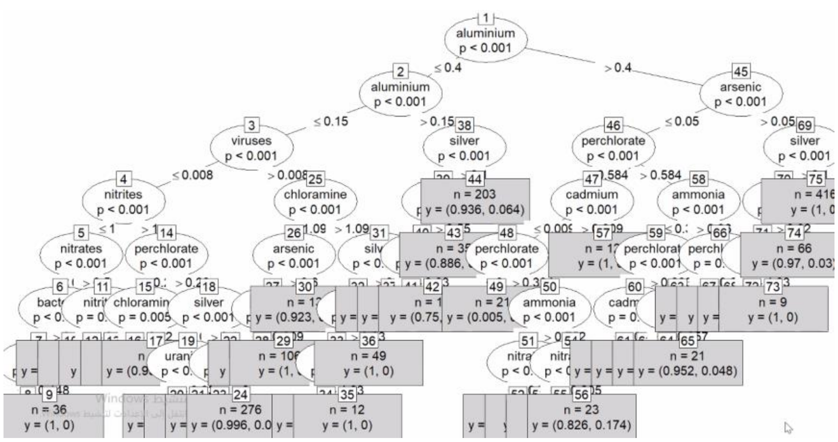
Classification:

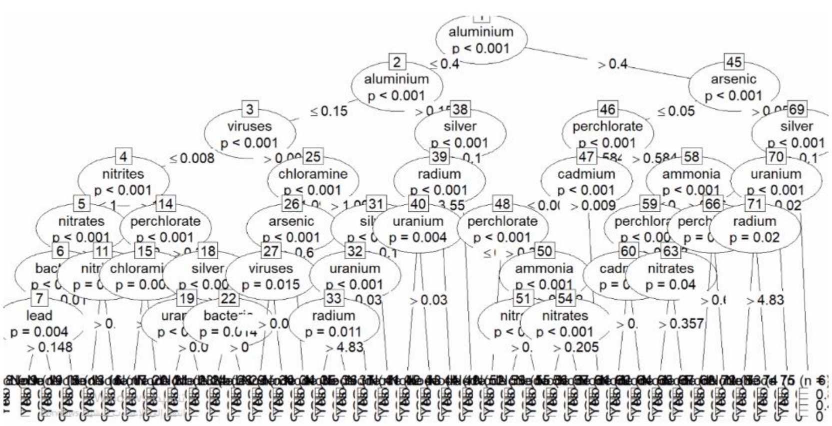
<table>
<tr><td>Mining task</td><td>Comparison Criteria</td></tr>
<tr><td>Classification</td><td>

Decision Tree #1: 70% training 30% testing
Decision Tree #2: 50% training 50% testing
Decision Tree #3: 80% training 20% testing

<table>
<tr><td></td><td>#1: 70% training 30% testing</td><td>#2: 50% training 50% testing</td><td>#3: 80% training 20% testing</td></tr>
<tr><td>Accuracy</td><td>95.50791%</td><td>96.25799%</td><td>96%</td></tr>
<tr><td>precision</td><td>96.02%</td><td>96.77%</td><td>96.35%</td></tr>
<tr><td>sensitivity</td><td>99.02%</td><td>99.07%</td><td>99.23%</td></tr>
<tr><td>specificity</td><td>69.26%</td><td>74.74%</td><td>71.61%</td></tr>
<tr><td>Preferred partition?</td><td>✗</td><td>✓</td><td>✗</td></tr>
</table>
</td></tr>
</table>

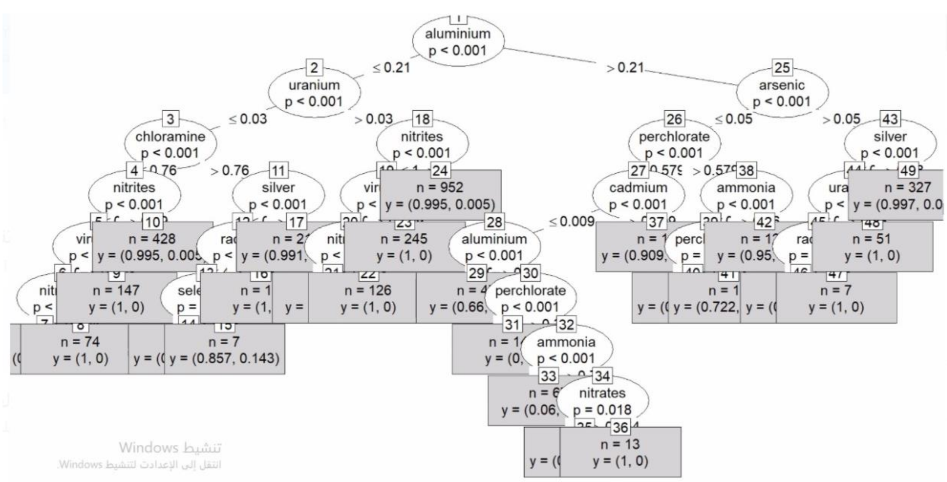
Decision Tree #1: 70%-30%

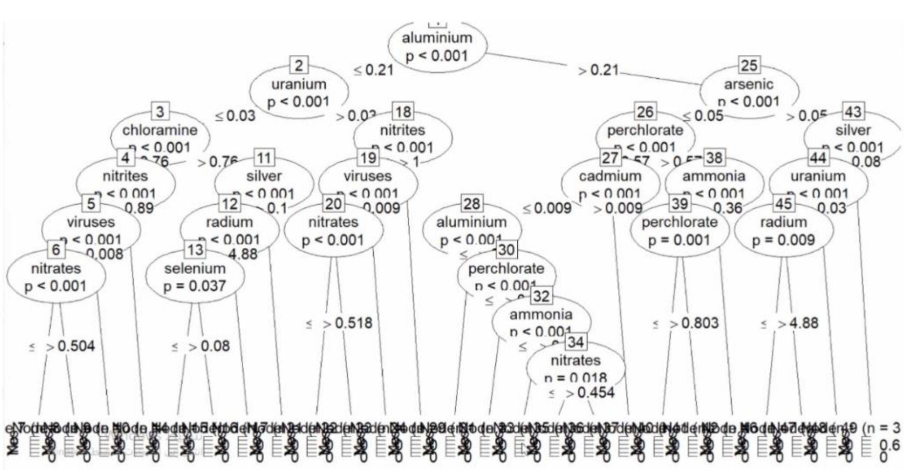




```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1711   71
       Yes   17  160

               Accuracy : 0.9551
                 95% CI : (0.9449, 0.9638)
    No Information Rate : 0.8821
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7597

 Mcnemar's Test P-Value : 1.606e-08

            Sensitivity : 0.9902
            Specificity : 0.6926
         Pos Pred Value : 0.9602
         Neg Pred Value : 0.9040
             Prevalence : 0.8821
         Detection Rate : 0.8734
   Detection Prevalence : 0.9096
      Balanced Accuracy : 0.8414

       'Positive' Class : No
```

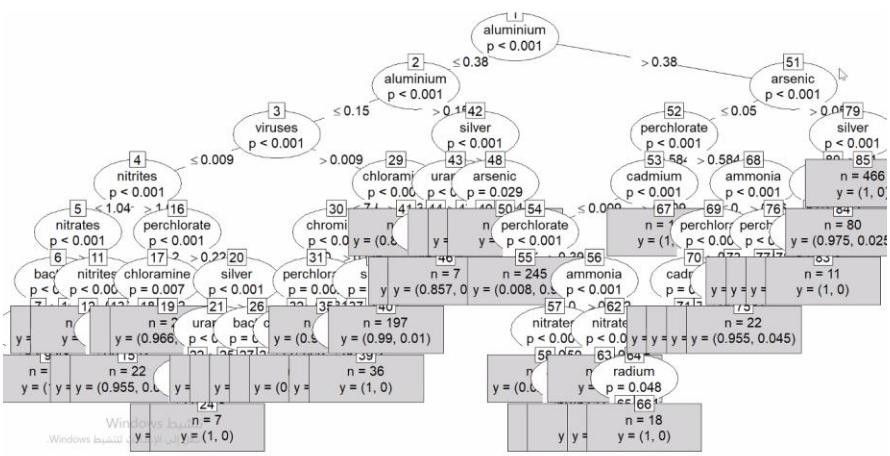

Decision Tree #2: 50%-50%

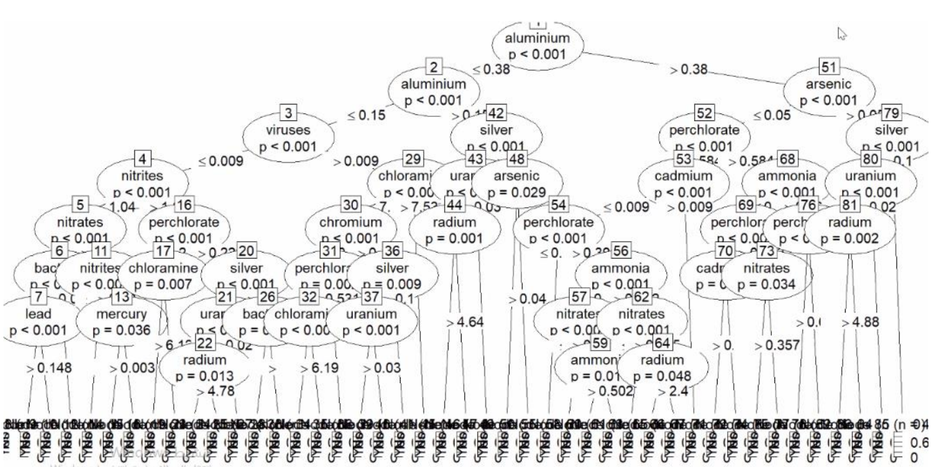




```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  2880   96
       Yes   27  284

               Accuracy : 0.9626
                 95% CI : (0.9555, 0.9688)
    No Information Rate : 0.8844
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.8013

 Mcnemar's Test P-Value : 8.713e-10

            Sensitivity : 0.9907
            Specificity : 0.7474
         Pos Pred Value : 0.9677
         Neg Pred Value : 0.9132
             Prevalence : 0.8844
         Detection Rate : 0.8762
   Detection Prevalence : 0.9054
      Balanced Accuracy : 0.8690

       'Positive' Class : No
```


Decision Tree #3: 80%-20%





```
Confusion Matrix and Statistics

          Reference
Prediction   No  Yes
       No  1161   44
       Yes    9  111

               Accuracy : 0.96
                 95% CI : (0.948, 0.9699)
    No Information Rate : 0.883
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7854

 Mcnemar's Test P-Value : 3.008e-06

            Sensitivity : 0.9923
            Specificity : 0.7161
         Pos Pred Value : 0.9635
         Neg Pred Value : 0.9250
             Prevalence : 0.8830
         Detection Rate : 0.8762
   Detection Prevalence : 0.9094
      Balanced Accuracy : 0.8542

       'Positive' Class : No
```

Clustering:

Now we will apply clustering to our dataset after removing the class label attribute

```
##Removing the class label for clustring technique
waterQuality<- subset( waterQuality, select = -is_safe )
```

Dataset after removing class label(is_safe)

Class label was here

```
waterQuality    6624 obs. of 20 variables
```

	aluminium	ammonia	arsenic	barium	cadmium	chloramine	chromium	copper	flouride	bacteria
1	1.65	0.305917753	0.040	2.85	0.007	0.35	0.83	0.17	0.05	0.20
2	2.32	0.709796055	0.010	3.31	0.002	5.28	0.68	0.66	0.90	0.65
3	1.01	0.471079906	0.040	0.58	0.008	4.24	0.53	0.02	0.99	0.05
4	1.36	0.381143430	0.040	2.96	0.001	7.23	0.03	1.66	1.08	0.71
5	0.92	0.815780675	0.030	0.20	0.006	2.67	0.69	0.57	0.61	0.13
6	0.94	0.486125042	0.030	2.88	0.003	0.80	0.43	1.38	0.11	0.67
8	3.93	0.666666667	0.040	0.66	0.001	6.22	0.10	1.86	0.86	0.16
9	0.60	0.824139084	0.010	0.71	0.005	3.14	0.77	1.45	0.98	0.35
10	0.22	0.562688064	0.020	1.37	0.007	6.40	0.49	0.82	1.24	0.83
11	3.27	0.122701438	0.001	2.69	0.005	5.75	0.15	0.60	1.29	0.04
12	1.35	0.736542962	0.040	0.84	0.002	0.10	0.76	0.17	0.58	0.52
13	1.88	0.646272150	0.020	2.78	0.008	0.05	0.42	1.00	0.09	0.91
14	4.93	0.804078903	0.040	3.05	0.008	0.70	0.51	1.35	1.07	0.70
16	0.61	0.082915413	0.030	0.59	0.002	1.94	0.77	1.54	0.62	0.23
17	3.47	0.531929121	0.020	0.06	0.001	5.29	0.47	1.08	1.43	0.89
19	4.88	0.903042461	0.020	0.36	0.001	1.21	0.68	0.71	0.99	0.75
21	0.68	0.637245069	0.001	0.04	0.006	4.57	0.20	1.18	1.00	0.92
22	1.15	0.273821464	0.020	0.97	0.007	3.47	0.65	1.51	1.46	0.58
23	0.27	0.359077232	0.020	0.55	0.001	3.74	0.12	1.77	0.43	0.80
24	4.32	0.692410565	0.030	2.60	0.008	7.24	0.61	1.23	1.44	0.56
26	3.31	0.740220662	0.030	0.46	0.001	7.22	0.73	1.05	1.00	0.25

	viruses	lead	nitrates	nitrites	mercury	perchlorate	radium	selenium	silver	uranium
1	0.000	0.054	0.811206461	1.13	0.007	0.63032226	6.78	0.08	0.34	0.02
2	0.650	0.100	0.100959112	1.93	0.003	0.53865420	3.21	0.08	0.27	0.05
3	0.003	0.078	0.714285714	1.11	0.006	0.83953916	7.07	0.07	0.44	0.01
4	0.710	0.016	0.070671378	1.29	0.004	0.15227918	1.72	0.02	0.45	0.05
5	0.001	0.117	0.339727410	1.11	0.003	0.28218400	2.41	0.02	0.06	0.02
6	0.670	0.135	0.491670873	1.89	0.006	0.45366505	5.42	0.08	0.19	0.02
8	0.005	0.197	0.688541141	1.81	0.001	0.89079980	7.24	0.08	0.08	0.07
9	0.002	0.167	0.739525492	1.84	0.004	0.39121723	4.99	0.08	0.25	0.08
10	0.830	0.109	0.241292277	1.46	0.010	0.50793121	0.08	0.03	0.31	0.01
11	0.008	0.145	0.427057042	1.25	0.006	0.92502922	7.80	0.05	0.33	0.06
12	0.520	0.011	0.928319031	1.49	0.009	0.35932543	1.30	0.08	0.48	0.08
13	0.910	0.103	0.220090863	1.95	0.006	0.36934380	1.97	0.03	0.06	0.05
14	0.700	0.101	0.058051489	1.11	0.008	0.44748706	5.58	0.09	0.38	0.03
16	0.001	0.017	0.099949520	1.08	0.007	0.18634163	0.98	0.01	0.47	0.03
17	0.890	0.080	0.095911156	1.20	0.008	0.00300551	6.89	0.06	0.12	0.08
19	0.750	0.071	0.015143867	1.22	0.002	0.94673568	1.00	0.00	0.41	0.05
21	0.920	0.086	0.477031802	1.41	0.007	0.36383370	3.05	0.03	0.13	0.08
22	0.580	0.061	0.451792024	1.50	0.004	0.24378026	1.74	0.03	0.01	0.06
23	0.800	0.114	0.640080767	1.18	0.008	0.57839372	0.90	0.02	0.16	0.06
24	0.560	0.012	0.475012620	1.74	0.004	0.60494239	3.22	0.07	0.18	0.08
26	0.007	0.109	0.096415952	1.07	0.001	0.65787277	0.49	0.04	0.47	0.05

Showing 1 to 21 of 6,624 entries, 20 total columns

Clustering

- K=2
- K=3
- K=5

	K=2	K=3	K=5

Silhouette width for each cluster	`cluster size ave.sil.width`
	`1 1 3168 0.07`
	`2 2 3456 0.26`
		`cluster size ave.sil.width`
		`1 1 1072 0.11`
		`2 2 2951 0.05`
		`3 3 2601 0.13`
			`cluster size ave.sil.width`
			`1 1 918 0.08`
			`2 2 1855 0.07`
			`3 3 1121 0.04`
			`4 4 1174 0.08`
			`5 5 1556 0.13`

Silhouette width for all clusters	0.17	0.09	0.08
Visualization	Figure 1	Figure 2	Figure 3
Preferred partition?	✓	✗	✗

Figure 1


Cluster plot
Dim2 (8.4%)
Dim1 (20.5%)
-4
-2
0
2
cluster
1
2

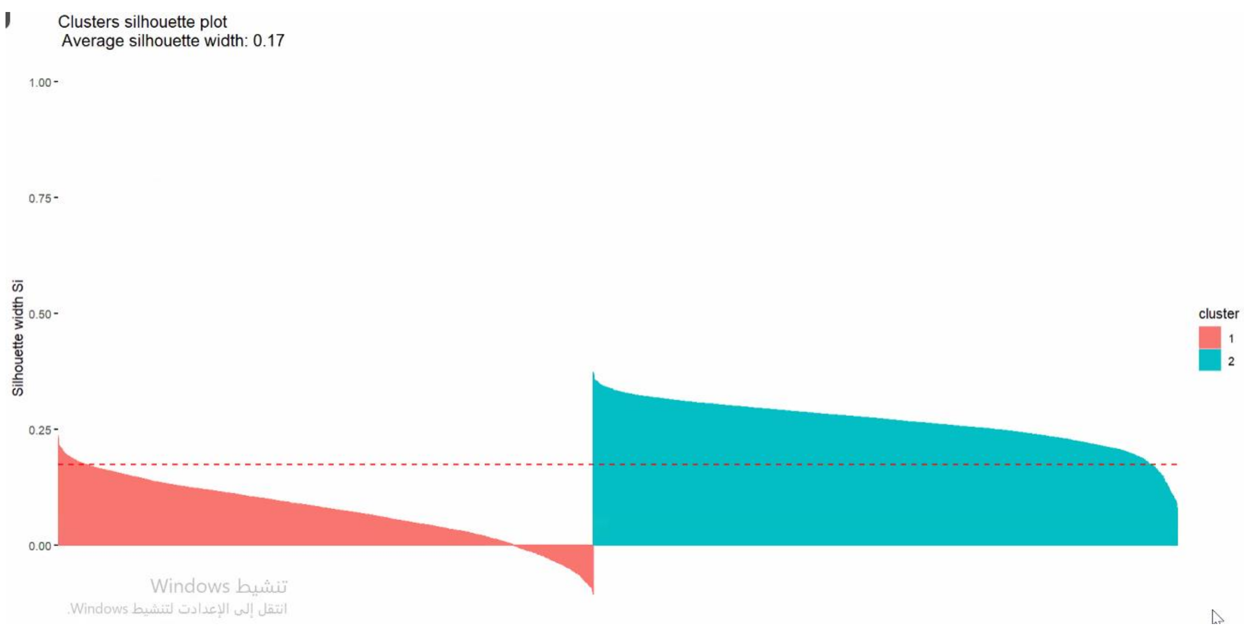
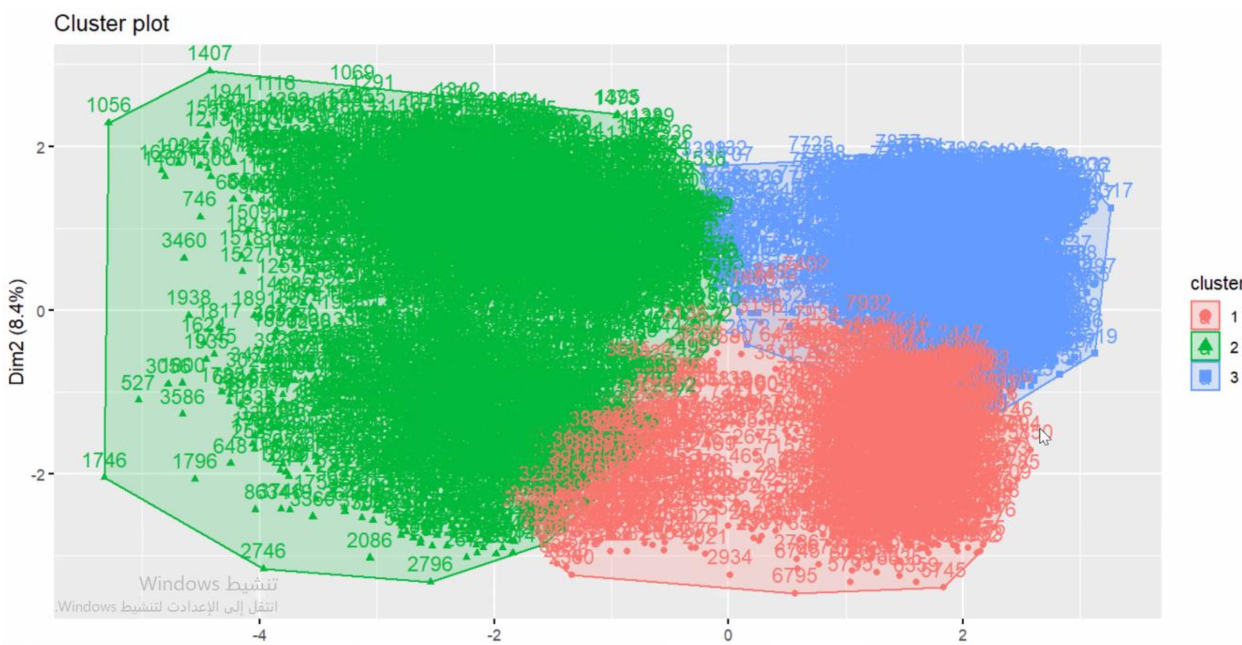

Clusters silhouette plot
Average silhouette width: 0.17
Silhouette width Si
1.00
0.75
0.50
0.25
0.00
cluster
1
2

Figure 2


Cluster plot
Dim2 (8.4%)
cluster
1
2
3
-4
-2
0
2

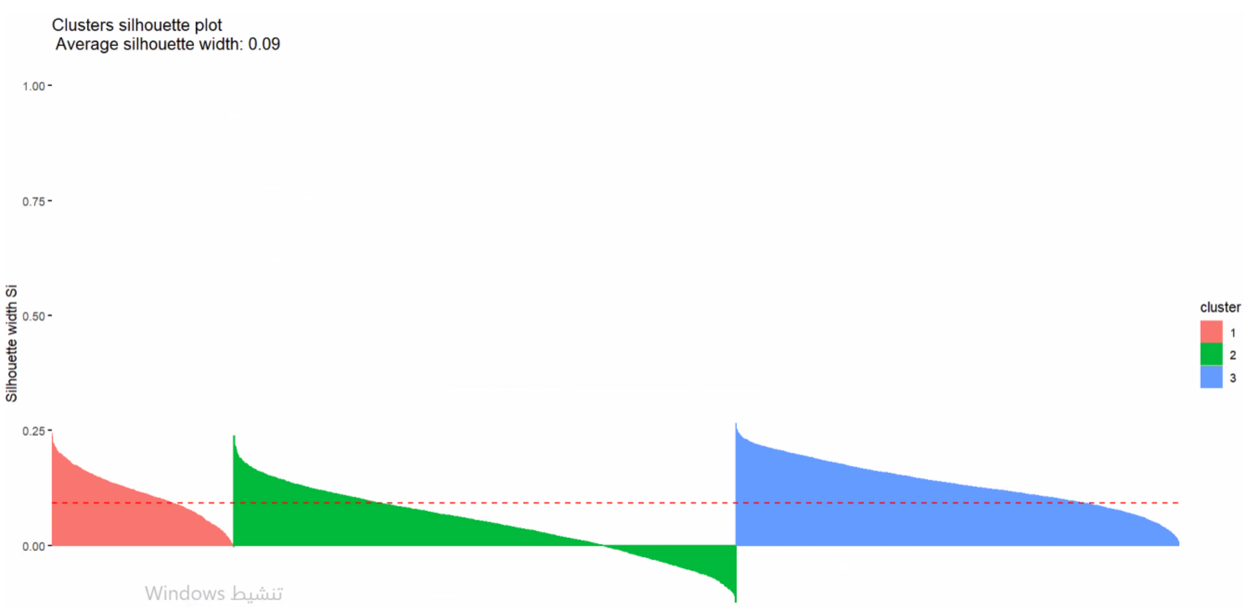
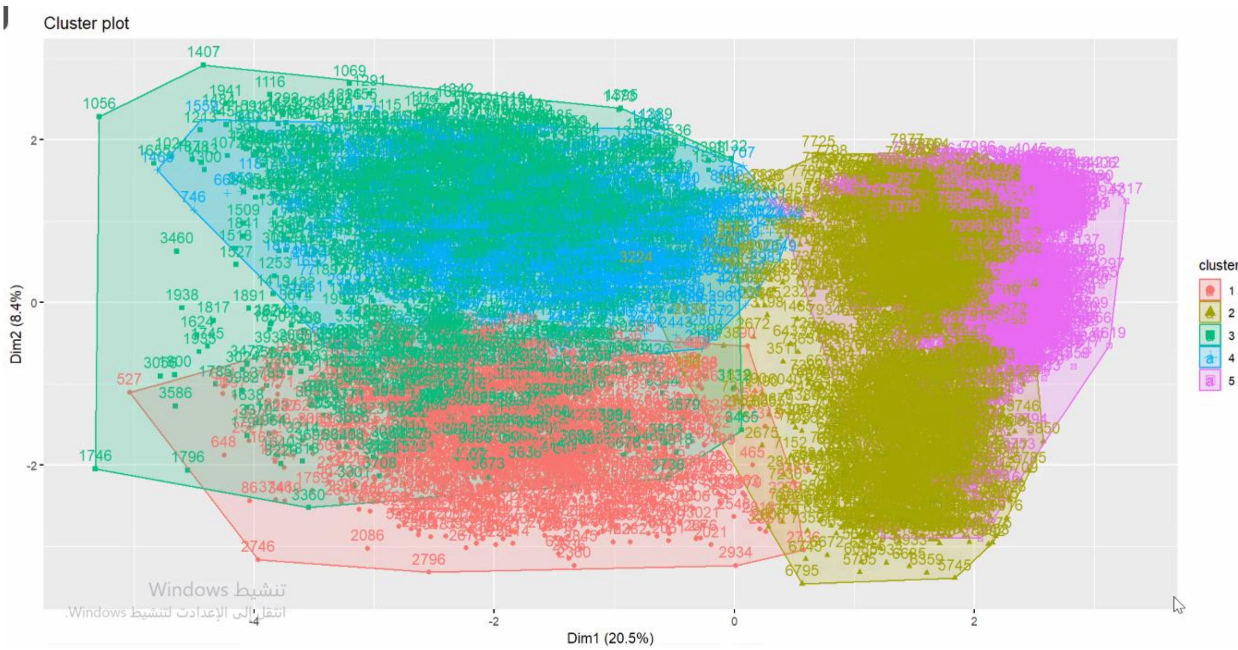
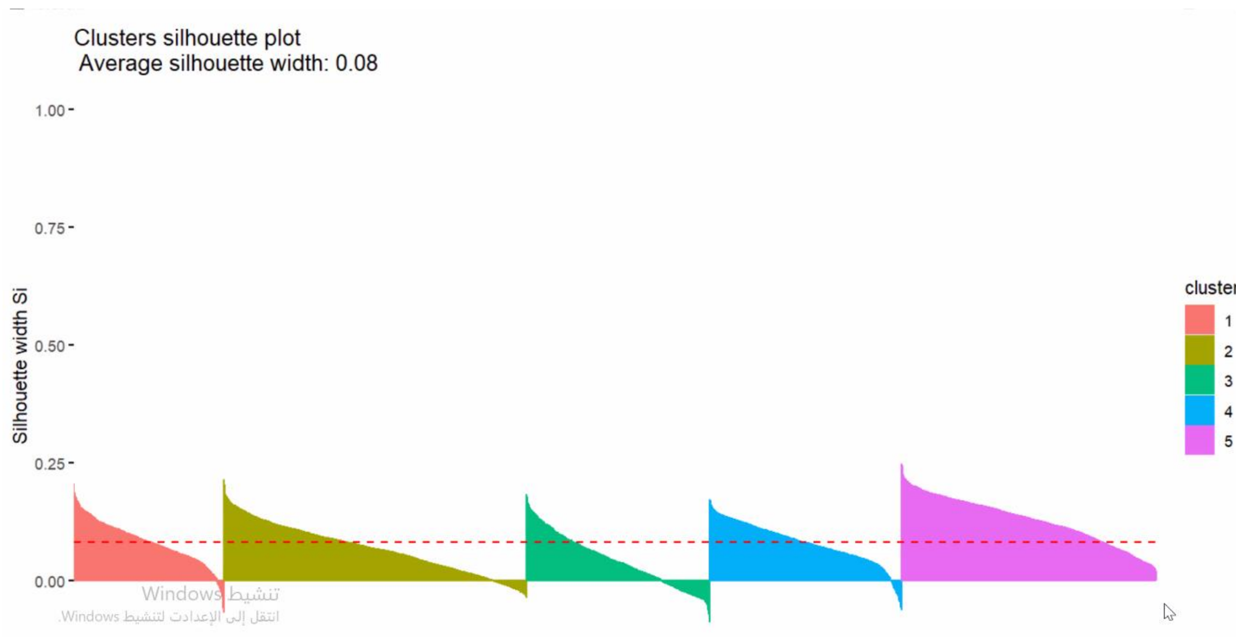
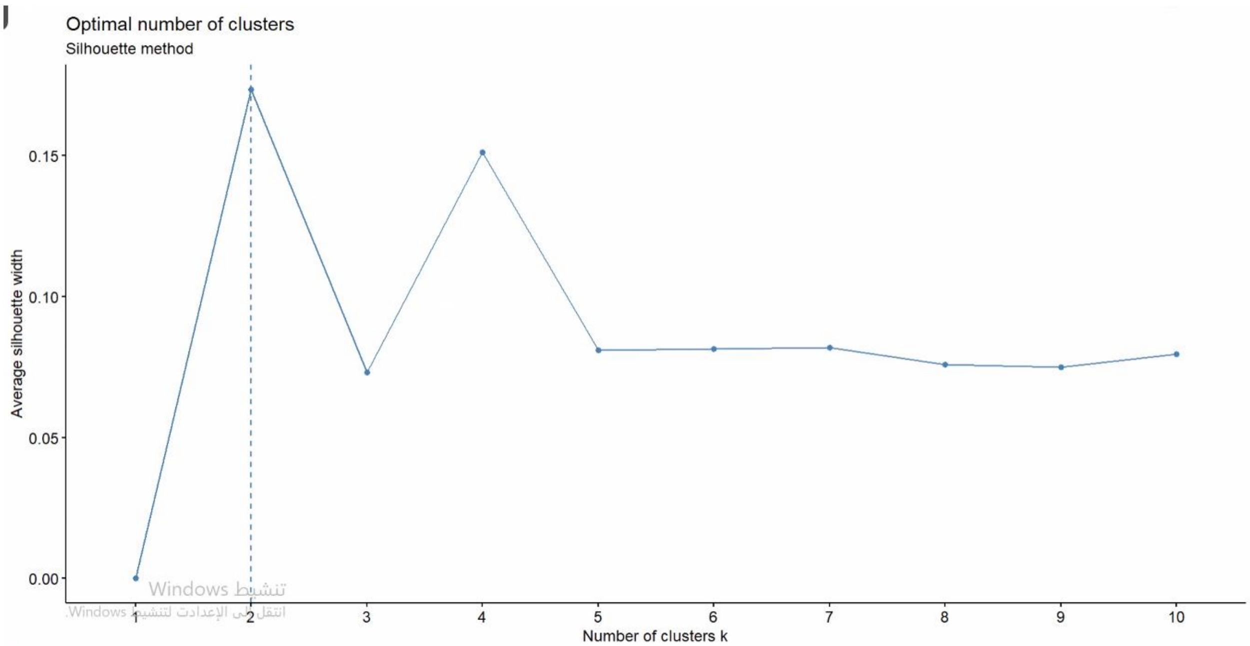

Clusters silhouette plot
Average silhouette width: 0.09
Silhouette width Si
1.00
0.75
0.50
0.25
0.00
cluster
1
2
3

Figure 3


Cluster plot
Dim1 (20.5%)
Dim2 (8.4%)
cluster
1
2
3
4
5
تنشيط Windows
انتقل إلى الإعدادات لتنشيط Windows.


Clusters silhouette plot
Average silhouette width: 0.08
Silhouette width Si
1.00
0.75
0.50
0.25
0.00
cluster
1
2
3
4
5
تنشيط Windows
انتقل إلى الإعدادات لتنشيط Windows.



3 Findings

After studying the water quality dataset, from determining what each attribute do and how the attribute will effect each other, we apply some preprocessing methods such as cleaning, transformation to prepare our dataset for data mining process.

For classification we studied different cases by dividing our dataset by using Ctree method and then we came up with these results:

-70% training 30% testing , Accuracy=95.50791%

-50% training 50% testing , Accuracy=96.25799%

-80% training 20% testing , Accuracy=96%

We noticed that almost all accuracies are the same, but these results lead to determine the best model for classification technique which is (50%,50%) because it has the highest accuracy which means the class label (is_safe) is affected by all attributes.

(ammonia , barium , arsenic…etc) , that means the evaluation model we considered to be the best classify most of the tuples that are covered by the rule and it correctly classified by class label (is_safe), also this model lead to determine the correct class_label for each object faster than the others . after analyzing the decision trees we noticed that (80%,20%) and (70%,30%) first splitting point was aluminum and the second level of the tree include aluminum and we think that what makes both two cases accuracies lower than (50%,50%), in (50%,50%) case the second level exchanged the aluminum with uranium.

We noticed that some of our dataset attributes has no affect in the decision tree and has not appeared as a decision node to determine the leaf node (decision) , and these attribute are barium , chromium , copper , fluoride , bacteria , lead and mercury .

As a result, we think analyzing and studying the decision tree is interesting for individuals because they can use this tree to determine that the water they use and drink is safe or not, and for companies to end up selling water that is good for people and the environment. We can also extract some rules from this tree such as:

if the aluminum > 0.21 and the arsenic > 0.05 and the silver < 0.08 and the uranium > 0.03 then n=51 and y=(1,0)

For clustering we studied different cases by changing the number of clusters k and using k-means method and then we came up with these results:

-k=2 , Silhouette width for all clusters: 0.17

-k=3 , Silhouette width for all clusters: 0.09

-k=5 , Silhouette width for all clusters:0.08

These results lead to determine the best model for clustering technique is k=2 because it has the highest Silhouette width (0.17) for all clusters, (0.07) for the first cluster and (0.26) for the second cluster, since the value is approaching 1 ,that means the objects within a cluster are closer to each other than to the objects in the other cluster.

After analyzing the plots, we configure what supports the quality of k=2, that the clusters are not overlapping in figure 1 above, unlike k=3 and k=5, the clusters are overlapping pointedly, in k=3, we can see that cluster 1,3 and cluster 1,2 are intertwined with each other in figure 2 above, as well as in k=5, we can see for example that cluster 2,5 and cluster 3,4 and other clusters are highly overlapped in figure 3 above, which can make k=5 the worst case in choosing number of clusters.

To capitalize, what makes k=3 and k=5 not in consideration is that because of the overlapping that result in inability when observing where each object belong to the right cluster, but k=2 the Silhouette width is 0.17 which is optimal because k-means method consider the number approach to 1 is better than the others, and what increased our confidence with k=2 is that it is not overlapping, to enhance our analyzing we validated which number of clusters is the best using fviz_nbclus() method , that came up with 2 clusters.

4 Code

```
#Data minning project
dataset=read.csv('/Users/admin/Desktop/WATER QUALITY/waterQuality.txt')

nrow(waterQuality)
ncol(waterQuality)
sum(is.na(waterQuality))
summary(waterQuality)
waterQuality$ammonia<-as.numeric(waterQuality$ammonia)

boxplot(waterQuality$aluminium , data=waterQuality)
boxplot(waterQuality$ammonia , data=waterQuality)
boxplot(waterQuality$arsenic , data=waterQuality)
boxplot(waterQuality$cadmium , data=waterQuality)
boxplot(waterQuality$chloramine,data=waterQulaity)
boxplot(waterQuality$Chromium,data=waterQulaity)
boxplot(waterQuality$Copper,data=waterQulaity)
boxplot(waterQuality$flouride,data=waterQulaity)
boxplot(waterQuality$bacteria,data=waterQulaity)
hist(waterQuality$barium)
pie(table(waterQuality$is_safe))
plot(waterQuality$lead,waterQuality$ammonia , main="Scatterplot", xlab="Lead", ylab="Ammonia")

is.na(waterQuality&ammonia)
waterQuality$ammonia=ifelse(is.na(waterQuality$ammonia),ave(waterQuality$ammonia,FUN=function(x) mean(x,na.rm=TRUE))

#install.packeges("outliers")
library(outliers)
OutlierUran=outlier(waterQuality$uranium,logical=TRUE)
sum(OutlierUran)
Find_outlier=which(OutlierUran==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierAl=outlier(waterQuality$aluminium,logical=TRUE)
sum(OutlierAl)
Find_outlier=which(OutlierAl==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierArs=outlier(waterQuality$arsenic,logical=TRUE)
sum(OutlierArs)
Find_outlier=which(OutlierArs==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierBar=outlier(waterQuality$barium,logical=TRUE)
sum(OutlierBar)
Find_outlier=which(OutlierBar==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierCad=outlier(waterQuality$cadmium,logical=TRUE)
sum(OutlierCad)
Find_outlier=which(OutlierCad==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierChlo=outlier(waterQuality$chloramine,logical=TRUE)
sum(OutlierChlo)
Find_outlier=which(OutlierChlo==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierChrom=outlier(waterQuality$chromium,logical=TRUE)
sum(OutlierChrom)
Find_outlier=which(OutlierChrom==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierCo=outlier(waterQuality$copper,logical=TRUE)
sum(OutlierCo)
Find_outlier=which(OutlierCo==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierFl=outlier(waterQuality$flouride,logical=TRUE)
sum(OutlierFl)
Find_outlier=which(OutlierFl==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierBa=outlier(waterQuality$bacteria,logical=TRUE)
sum(OutlierBa)
Find_outlier=which(OutlierBa==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierVi=outlier(waterQuality$viruses,logical=TRUE)
sum(OutlierVi)
Find_outlier=which(OutlierVi==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierLe=outlier(waterQuality$lead,logical=TRUE)
sum(OutlierLe)
Find_outlier=which(OutlierLe==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierNi1=outlier(waterQuality$nitrates,logical=TRUE)
sum(OutlierNi1)
Find_outlier=which(OutlierNi1==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierNi2=outlier(waterQuality$nitrites,logical=TRUE)
sum(OutlierNi2)
Find_outlier=which(OutlierNi2==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierMe=outlier(waterQuality$mercury,logical=TRUE)
sum(OutlierMe)
Find_outlier=which(OutlierMe==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierPe=outlier(waterQuality$perchlorate,logical=TRUE)
```

```r
OutlierPe=outlier(waterQuality$perchlorate,logical=TRUE)
sum(OutlierPe)
Find_outlier=which(OutlierPe==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierRa=outlier(waterQuality$radium,logical=TRUE)
sum(OutlierRa)
Find_outlier=which(OutlierRa==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierSe=outlier(waterQuality$selenium,logical=TRUE)
sum(OutlierSe)
Find_outlier=which(OutlierSe==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

OutlierSi=outlier(waterQuality$silver,logical=TRUE)
sum(OutlierSi)
Find_outlier=which(OutlierSi==TRUE,arr.ind=TRUE)
waterQuality= waterQuality[-Find_outlier,]

#Encoding:
waterQuality$is_safe = factor(waterQuality$is_safe,levels = c("0","1"), labels = c("No","Yes"))

##Normlize amonia
normlize<- function(x){
  return((x-min(x)) / (max(x)-min(x)))
}
waterQuality$ammonia<-normlize(waterQuality$ammonia)
##Normlize perchlorate
normlize<- function(x){
  return((x-min(x)) / (max(x)-min(x)))
}
waterQuality$perchlorate<-normlize(waterQuality$perchlorate)
##Normlize nitrates
normlize<- function(x){
  return((x-min(x)) / (max(x)-min(x)))
}
waterQuality$nitrates<-normlize(waterQuality$nitrates)


View(waterQuality)

waterQuality <- na.omit(waterQuality)
pie(table(waterQuality$is_safe))

##Classification/30,70
set.seed(1234)
firstP <- sample(2, nrow(waterQuality), replace=TRUE, prob=c(0.7, 0.3))
trainData <- waterQuality[firstP==1,]
testData <- waterQuality[firstP==2,]
install.packages('party')
library(party)
myFormula <- is_safe ~ aluminium + ammonia + arsenic + barium + cadmium + chloramine+ chromium + copper + flouride +

waterQuality_ctree <- ctree(myFormula, data=trainData)
table(predict(waterQuality_ctree), trainData$is_safe)
print(waterQuality_ctree)
plot(waterQuality_ctree,type="simple")
plot(waterQuality_ctree)

testPred <- predict(waterQuality_ctree, newdata = testData)

#Evaluate the model
#Create the confusion matrix
table(testPred, testData$is_safe)

install.packages('caret')
library(caret)
results <- confusionMatrix(testPred, testData$is_safe)
acc <- results$overall["Accuracy"]*100
acc
results

##Classifation/50,50
set.seed(1234)
firstP <- sample(2, nrow(waterQuality), replace=TRUE, prob=c(0.5, 0.5))
trainData <- waterQuality[firstP==1,]
testData <- waterQuality[firstP==2,]

myFormula <- is_safe ~ aluminium + ammonia + arsenic + barium + cadmium + chloramine+ chromium + copper + flouride +

waterQuality_ctree <- ctree(myFormula, data=trainData)
table(predict(waterQuality_ctree), trainData$is_safe)
print(waterQuality_ctree)
plot(waterQuality_ctree,type="simple")
plot(waterQuality_ctree)

testPred <- predict(waterQuality_ctree, newdata = testData)

#Evaluate the model
#Create the confusion matrix
table(testPred, testData$is_safe)

results <- confusionMatrix(testPred, testData$is_safe)
acc <- results$overall["Accuracy"]*100
acc
results

##Classifation/80,20
```

```r
##Classifation/80,20
set.seed(1234)
firstP <- sample(2, nrow(waterQuality), replace=TRUE, prob=c(0.8, 0.2))
trainData <- waterQuality[firstP==1,]
testData <- waterQuality[firstP==2,]

myFormula <- is_safe ~ aluminium + ammonia + arsenic + barium + cadmium + chloramine+ chromium + copper + flouride +

waterQuality_ctree <- ctree(myFormula, data=trainData)
table(predict(waterQuality_ctree), trainData$is_safe)
print(waterQuality_ctree)
plot(waterQuality_ctree,type="simple")
plot(waterQuality_ctree)

testPred <- predict(waterQuality_ctree, newdata = testData)

#Evaluate the model
#Create the confusion matrix
table(testPred, testData$is_safe)

results <- confusionMatrix(testPred, testData$is_safe)
acc <- results$overall["Accuracy"]*100
acc
results
```

```r
##Removing the class label for clustring technique
waterQuality<- subset( waterQuality, select = -is_safe )

# k-means clustering
set.seed(8953)

waterQuality <- scale(waterQuality)
#First clustring K=2:
kmeans.result1 <- kmeans(waterQuality, 2)
kmeans.result1

install.packages("factoextra")
library(factoextra)
fviz_cluster(kmeans.result1, data = waterQuality)

###Cluster Validation
install.packages("cluster")
library(cluster)
#average for each cluster
avg_sil <- silhouette(kmeans.result1$cluster,dist(waterQuality))
fviz_silhouette(avg_sil)

############################
#Second clustring K=3:
kmeans.result2 <- kmeans(waterQuality,3)
kmeans.result2
```

```r
fviz_cluster(kmeans.result2, data = waterQuality)

###Cluster Validation

#average for each cluster
avg_sil <- silhouette(kmeans.result2$cluster,dist(waterQuality))
fviz_silhouette(avg_sil)

############################
#Third clustring K=5:
kmeans.result3 <- kmeans(waterQuality,5)
kmeans.result3


fviz_cluster(kmeans.result3, data = waterQuality)

###Cluster Validation

#average for each cluster
avg_sil <- silhouette(kmeans.result3$cluster,dist(waterQuality))
fviz_silhouette(avg_sil)

install.packages("NbClust")
library(NbClust)
fviz_nbclust(waterQuality, kmeans, method = "silhouette")+labs(subtitle = "Silhouette method")
```