Ranking Applications for Nursery Schools - Relabelling Dataset using Clustering

by Viviane Adohouannon, Kate Alexander, Diana Azbel, Igor Baranov

Abstract The specific problem under consideration is to rank selection of applicants for nursery schools in Ljubljana, Slovenia in the 1980's. Nursery Database was derived from a hierarchical decision model originally developed to rank applications. A classification model has been used to developed a reliable recomendation algorithm to predict if a specific applicant is a suitable candidate to be addmitted into a nursery school.

Introduction

In our

Early schooling such as Nursery school education matters most as it impacts children's long-term development and academic progress. While all children benefit from a high-quality nursery school, family structure, social and financial standing, and proximity to schools may affect the enrollment process.

Nursery dataset (**?**) was derived from a hierarchical decision model originally developed to rank applications for nursery schools. It was used during several years in 1980's when there was excessive enrollment to these schools in Ljubljana, Slovenia, and the rejected applications frequently needed an objective explanation.

Background

In our previous article (**?**) we explored the Nursery dataset and using a classification model to developed our algorithm to predict applicants suitabilty of an admittance within the nursery school system, the evaluation of the model was developed using Random Forest algorithm. Even though the original dataset was clear and did not have missing values, it was unballanced. To overcome this a method called Random Over-Sampling was appplied, which increased the number of instances in the minority class by randomly replicating them in order to present a higher representation of the minority class in the sample. The final model achieved almost 99% accuracy of predictions with overall balanced precision more that 97% accross all the predicted "class" values.

Ljubljana is the capital and largest city of Slovenia. The city with an area of 163.8 square kilometers is situated in the Ljubljana Basin in Central Slovenia, between the Alps and the Karst. Ljubljana is located some 320 kilometers south of Munich, 477 kilometers east of Zurich. In 1981 the population of the city rose to 224,817 inhabitants with approximately 91% of the population speaking Slovene as their primary native language. The second most-spoken language is Bosnian, with Serbo-Croatian being the third most-spoken language (**?**).

During this time according to (Olave et al., 1989) "new housing developments populated by young families, the demand for childrens admission in nursery schools outstrips supply, notwithstanding the fast growth rate of new schools." In this research, conducted in 1989, an application of expert systems for admission procedures in public school systems was presented. The specific problem under consideration was selection of applicants for public nursery schools. The selection was supported by an expert system which evaluates, classifies and ranks applications. The main emphasis was on explanation of the underlying knowledge and solutions, suggested by the system. The system has been developed using DECMAK, an expert system shell for multi-attribute decision making.

In continuation of the search for the practical solution of the problem, second most relevant research was conducted in 1997 (Zupan et al., 1997). It presented another machine learning method that, given a set of training examples, induced a definition of the target concept in terms of a hierarchy of intermediate concepts and their definitions. This effectively decomposed the problem into smaller, less complex problems. The method was inspired by the Boolean function decomposition approach to the design of digital circuits. To cope with high time complexity of inding an optimal decomposition, the authors proposed a suboptimal heuristic algorithm.

It worth to notice that in both cases the researches concentrated on designing and theoretical evaluation of mentioned algorithms without proposing a practical solution that could be deployed in the municipalities and put in use. The reason of this was a relatively high at the time price of

computers that could make predictictions with the models proposed.

Objective

The objective of this article is to provide a reliable and feasible recommendation algorithm to predict if a specific applicant is a suitable candidate to be admitted into a nursery school or if the applicant should stay with their parents. The results of this recommendation may affect the childs engagement within the school, parents involvement in school activities and overall satisfaction of the applicants long-term academic progress.

Plan

To solve the objective a group of four students calling themselves The First Group (T.F.G) from York University School of Continuing Studies, have come together to create an algorithm using the Nursery dataset. Since the target of this problem is a categorical value representing recommendation if a child is suitable for the admittance to a nursery school, a supervised classification algorithm was chosen (Witten et al., 2011).

The main tool used in developing the recomendation algorithm is R (R Core Team, 2012). The R language is widely used among statisticians and data miners for developing statistical software and data analysis.

Data understanding

The dataset (**?**) has 8 attributes and 12960 instances. Creators of the dataset suggested hierarchical model ranks nursery-school applications according to the following concept structure:

```
NURSERY Evaluation of applications for nursery schools
. EMPLOY Employment of parents and child's nursery
. . parents Parents' occupation
. . has_nurs Child's nursery
. STRUCT_FINAN Family structure and financial standings
. . STRUCTURE Family structure
. . . form Form of the family
. . . children Number of children
. . housing Housing conditions
. . finance Financial standing of the family
. SOC_HEALTH Social and health picture of the family
. . social Social conditions
. . health Health conditions
```

Nursery Database contains examples with the structural information removed, i.e., directly relates NURSERY to the eight input attributes: parents, has_nurs, form, children, housing, finance, social, health. Data set attribes presented in the following form:

```
parents: usual, pretentious, great_pret
has_nurs: proper, less_proper, improper, critical, very_crit
form: complete, completed, incomplete, foster
children: 1, 2, 3, more
housing: convenient, less_conv, critical
finance: convenient, inconv
social: non-prob, slightly_prob, problematic
health: recommended, priority, not_recom
```

Target attribute called “class” is a categorical variable having several values that were not revealed in original dataset description and had to be extracted from the data.

Data Preparation

To perform the analysis, certain R libraries were used. The code below was used to load and initialize the libraries. The first line invoking seed function was applied to enforce the repeatability of the calculation results.

```
set.seed(77)
library(readr)
library(dplyr)
library(ggplot2)
library(rpart)
library(rpart.plot)
library(Amelia)
library(rattle)
library(RColorBrewer)
library(caret)
```

The dataset was loaded directly from the dataset site (**?**) using the R statement below. Note that column names were assigned as the online data did not have the header. To pretty-print the head of the dataset xtable (Dahl, 2016) library was used to generate Table 1.

```
nursery_data <- read.csv(
  "http://archive.ics.uci.edu/ml/machine-learning-databases/nursery/nursery.data",
  header = FALSE,
  col.names =
    c("parents","has_nurs","form","children","housing","finance","social","health","class"))
```

Thise are the dimensions of our dataset:

```
dim(nursery_data)

#> [1] 12960     9
```

	parents	has_nurs	form	children	housing	finance	social	health	class
1	usual	proper	complete	1	convenient	convenient	nonprob	recommended	recommend
2	usual	proper	complete	1	convenient	convenient	nonprob	priority	priority
3	usual	proper	complete	1	convenient	convenient	nonprob	not_recom	not_recom
4	usual	proper	complete	1	convenient	convenient	slightly_prob	recommended	recommend
5	usual	proper	complete	1	convenient	convenient	slightly_prob	priority	priority
6	usual	proper	complete	1	convenient	convenient	slightly_prob	not_recom	not_recom
7	usual	proper	complete	1	convenient	convenient	problematic	recommended	priority
8	usual	proper	complete	1	convenient	convenient	problematic	priority	priority
9	usual	proper	complete	1	convenient	convenient	problematic	not_recom	not_recom
10	usual	proper	complete	1	convenient	inconv	nonprob	recommended	very_recom

Table 1: Nursery Data Dataset (head)

Summary of Nursery Data set is extracted by the R summary function, the results are presented below.

```
summary(nursery_data)

#>         parents            has_nurs            form      children
#>  great_pret :4320   critical   :2592   complete  :3240   1   :3240
#>  pretentious:4320   improper   :2592   completed :3240   2   :3240
#>  usual      :4320   less_proper:2592   foster    :3240   3   :3240
#>                     proper     :2592   incomplete:3240   more:3240
#>                     very_crit  :2592
#>        housing           finance               social
#>  convenient:4320   convenient:6480   nonprob      :4320
#>  critical  :4320   inconv    :6480   problematic  :4320
#>  less_conv :4320                     slightly_prob:4320
#>
#>
#>          health            class
#>  not_recom  :4320   not_recom :4320
#>  priority   :4320   priority  :4266
```

```
#>  recommended:4320   recommend :   2
#>                     spec_prior:4044
#>                     very_recom: 328
```

Reordering Dataset Factor Leveks

Very often, especially when plotting data, we need to reorder the levels of a factor because the default order is alphabetical. A direct way of reordering, using standard syntax is as follows. Current levels, need to be corrected to correspont to the dataset description.

```
#> [1] "great_pret"  "pretentious" "usual"

#> [1] "critical"    "improper"    "less_proper" "proper"      "very_crit"

#> [1] "complete"   "completed"  "foster"     "incomplete"

#> [1] "1"    "2"    "3"    "more"

#> [1] "convenient" "critical"   "less_conv"

#> [1] "convenient" "inconv"

#> [1] "nonprob"       "problematic"   "slightly_prob"

#> [1] "not_recom"   "priority"    "recommended"

#> [1] "not_recom"  "priority"   "recommend"  "spec_prior" "very_recom"
```

Correction:

```
nursery_data$parents <- factor(nursery_data$parents,levels(nursery_data$parents)[c(3,2,1)])
nursery_data$has_nurs <- factor(nursery_data$has_nurs,levels(nursery_data$has_nurs)[c(4,3,2,1,5)])
nursery_data$form <- factor(nursery_data$form,levels(nursery_data$form)[c(1,2,4,3)])
nursery_data$children <- factor(nursery_data$children,levels(nursery_data$children)[c(1,2,3,4)])
nursery_data$housing <- factor(nursery_data$housing,levels(nursery_data$housing)[c(1,3,2)])
nursery_data$finance <- factor(nursery_data$finance,levels(nursery_data$finance)[c(1,2)])
nursery_data$social <- factor(nursery_data$social,levels(nursery_data$social)[c(1,3,2)])
nursery_data$health <- factor(nursery_data$health,levels(nursery_data$health)[c(1,3,2)])
nursery_data$class <- factor(nursery_data$class,levels(nursery_data$class)[c(1,3,5,2,4)])
```

Corrected levels, now correspond to the dataset description:

```
#> [1] "usual"       "pretentious" "great_pret"

#> [1] "proper"      "less_proper" "improper"    "critical"    "very_crit"

#> [1] "complete"   "completed"  "incomplete" "foster"

#> [1] "1"    "2"    "3"    "more"

#> [1] "convenient" "less_conv"  "critical"

#> [1] "convenient" "inconv"

#> [1] "nonprob"       "slightly_prob" "problematic"

#> [1] "not_recom"   "recommended" "priority"

#> [1] "not_recom"  "recommend"  "very_recom" "priority"   "spec_prior"
```

Convert to numbers in one step

Return the matrix obtained by converting all the variables in a data frame to numeric mode and then binding them together as the columns of a matrix. Factors and ordered factors are replaced by their internal codes. Logical and factor columns are converted to integers.

```
data <- data.matrix(nursery_data)
head(data)
```

```
#>      parents has_nurs form children housing finance social health class
#> [1,]       1        1    1        1       1       1      1      2     2
#> [2,]       1        1    1        1       1       1      1      3     4
#> [3,]       1        1    1        1       1       1      1      1     1
#> [4,]       1        1    1        1       1       1      2      2     2
#> [5,]       1        1    1        1       1       1      2      3     4
#> [6,]       1        1    1        1       1       1      2      1     1
```

Preparing scaled data

```
#index <- sample(1:nrow(data),round(0.75*nrow(data)))
#index <- createDataPartition(y= data$QLT, p=0.5, list = FALSE)
maxs <- apply(data, 2, max)
mins <- apply(data, 2, min)
scaled <- as.data.frame(scale(data, center = mins, scale = maxs - mins))
#train_ <- scaled[index,]
#test_ <- scaled[-index,]
```

The problem

Distribution of target value in the dataset

The target value class of the wine quality is not equally distributed. The Figure 1 demonstrates the distribution. As we can see, dataset covers mostly medium-quality wines with QLT between 5 and 7 well, low and high quality wines represented poorly.

```
prop.table(table(nursery_data$class))
```

```
#>
#>   not_recom   recommend  very_recom    priority  spec_prior
#> 0.333333333 0.000154321 0.025308642 0.329166667 0.312037037
```

```
ggplot(data = nursery_data, mapping = aes(x = class)) + geom_bar()
```

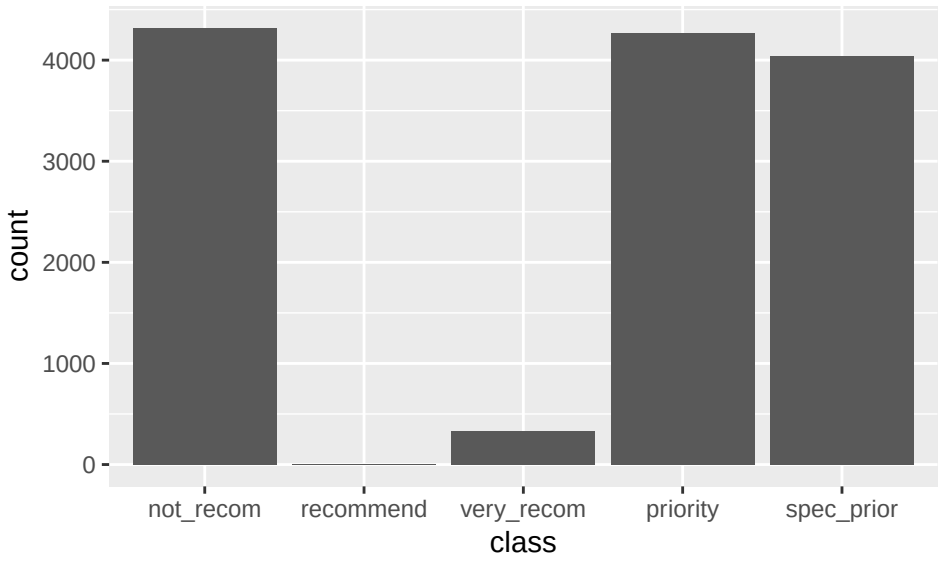


Figure 1: Distribution the Target 'class' Attribute in the Nursery Dataset

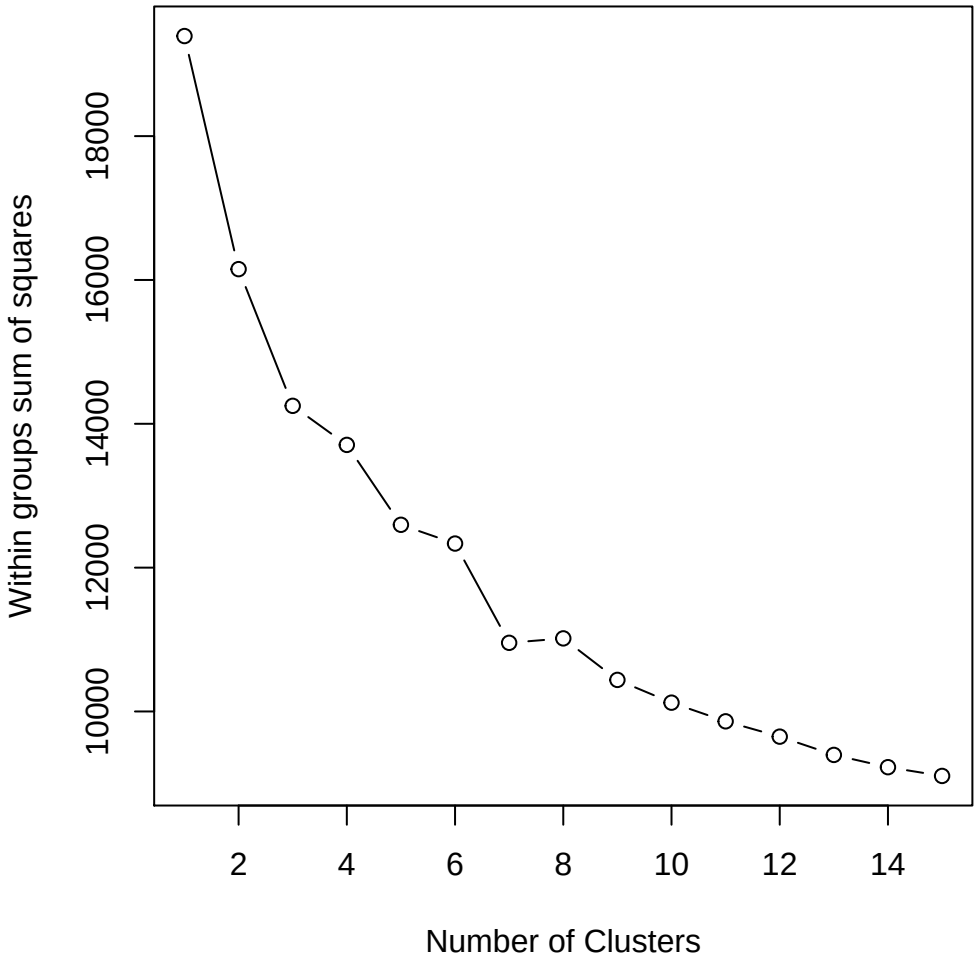


Figure 2: Elbow Criterion Diagram

Clustering

A fundamental question is how to determine the value of the parameter k. If we looks at the percentage of variance explained as a function of the number of clusters: One should choose a number of clusters so that adding another cluster doesn't give much better modeling of the data. More precisely, if one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much information (explain a lot of variance), but at some point the marginal gain will drop, giving an angle in the graph. The number of clusters is chosen at this point, hence the 'elbow criterion'. The diagram presented in Figure 2

```
wssplot <- function(data, nc=15, seed=1234){
  wss <- (nrow(data)-1)*sum(apply(data,2,var))
  for (i in 2:nc){
    set.seed(seed)
    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
  plot(1:nc, wss, type="b", xlab="Number of Clusters",
       ylab="Within groups sum of squares")}

wssplot(scaled, nc=15)
```

Clustering using K-means method

k-means clustering (**?**) is a method of vector quantization, that is popular for cluster analysis in data mining. k-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells (**?**).

The problem is computationally difficult (NP-hard), k-means clustering tends to find clusters of comparable spatial extent, while the expectation-maximization mechanism allows clusters to have different shapes. The algorithm has a loose relationship to the k-nearest neighbor classifier. One can apply the 1-nearest neighbor classifier on the cluster centers obtained by k-means to classify new data into the existing clusters. This is known as nearest centroid classifier or Rocchio algorithm.

```
set.seed(420)
clusters_num =5
k.means.fit <- kmeans(scaled, clusters_num,iter.max = 1000)
attributes(k.means.fit)
```

```
#> $names
#> [1] "cluster"      "centers"      "totss"        "withinss"
#> [5] "tot.withinss" "betweenss"    "size"         "iter"
#> [9] "ifault"
#>
#> $class
#> [1] "kmeans"
```

```
k.means.fit$centers
```

```
#>     parents  has_nurs      form  children   housing finance    social
#> 1 0.5000000 0.5000000 0.5000000 0.5000000 0.5000000       1 0.5000000
#> 2 1.0000000 0.5000000 0.5000000 0.5000000 0.5000000       1 0.5000000
#> 3 0.2500000 0.5000000 0.5000000 0.5000000 0.5000000       1 0.5000000
#> 4 0.5002316 0.5002316 0.5002316 0.5002316 0.5002316       0 0.5001158
#> 5 0.4995375 0.4995375 0.4995375 0.4995375 0.4995375       0 0.4997687
#>         health        class
#> 1 0.0000000000 0.0000000000
#> 2 0.7500000000 0.9399305556
#> 3 0.7500000000 0.8354166667
#> 4 0.7501157943 0.8448355720
#> 5 0.0004625347 0.0002312673
```

```
k.means.fit$size
```

```
#> [1] 2160 1440 2880 4318 2162
```

```
library(cluster)
clusplot(scaled, k.means.fit$cluster, main='',
         color=TRUE, shade=FALSE,
         labels=clusters_num, lines=0)
```

Explain clusters

Explain by 'class'

Let's try to explain clusters by the 'class'. Code below builds a matrix whe columns are cluster numbers and rows are target classes.

```
table(nursery_data$class,k.means.fit$cluster)
```

```
#>
#>                1    2    3    4    5
#>   not_recom  2160    0    0    0 2160
#>   recommend     0    0    0    0    2
#>   very_recom    0    0  110  218    0
#>   priority      0  346 1676 2244    0
#>   spec_prior    0 1094 1094 1856    0
```

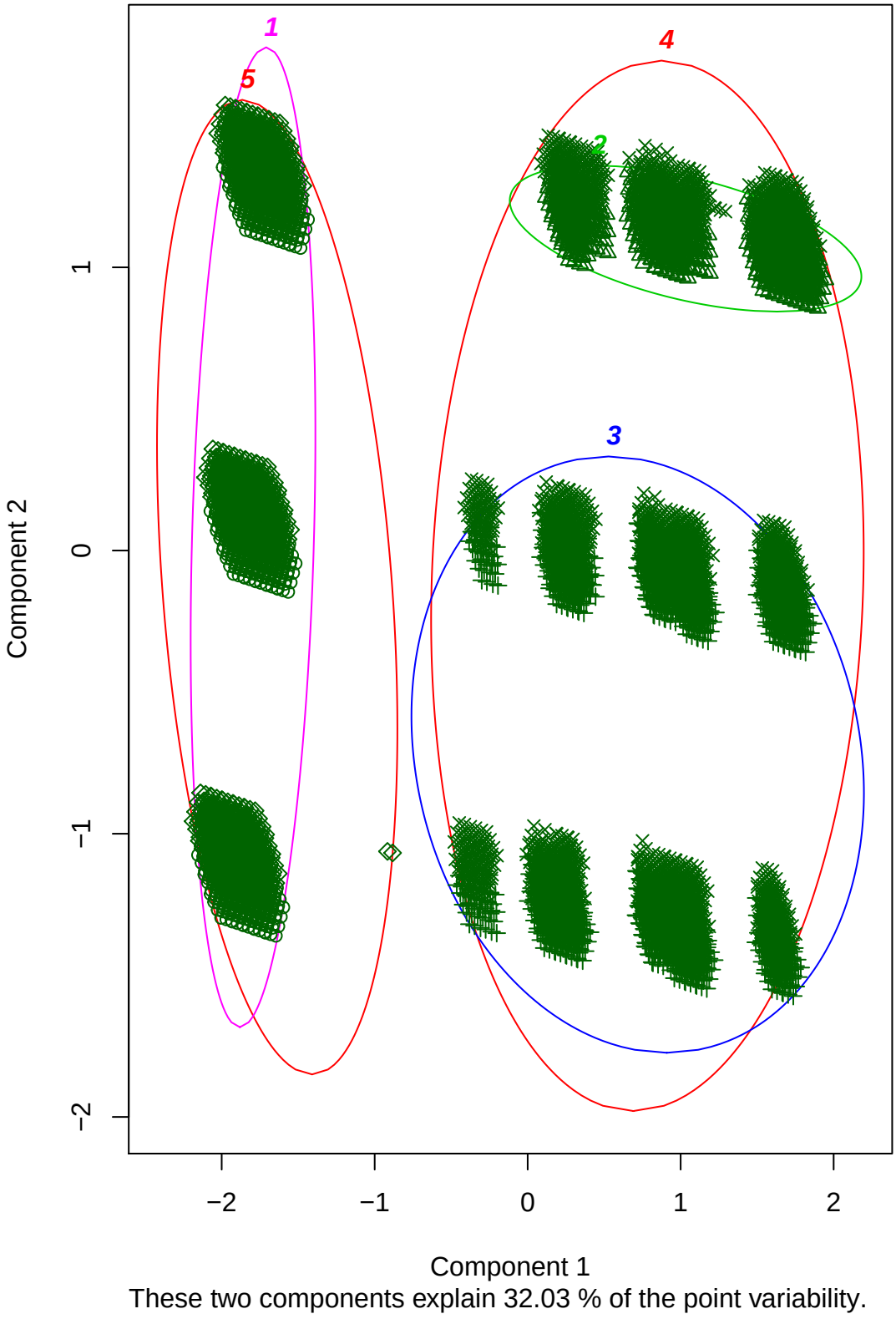


Figure 3: 2D representation of the Cluster solution

Hierarchical Clustering

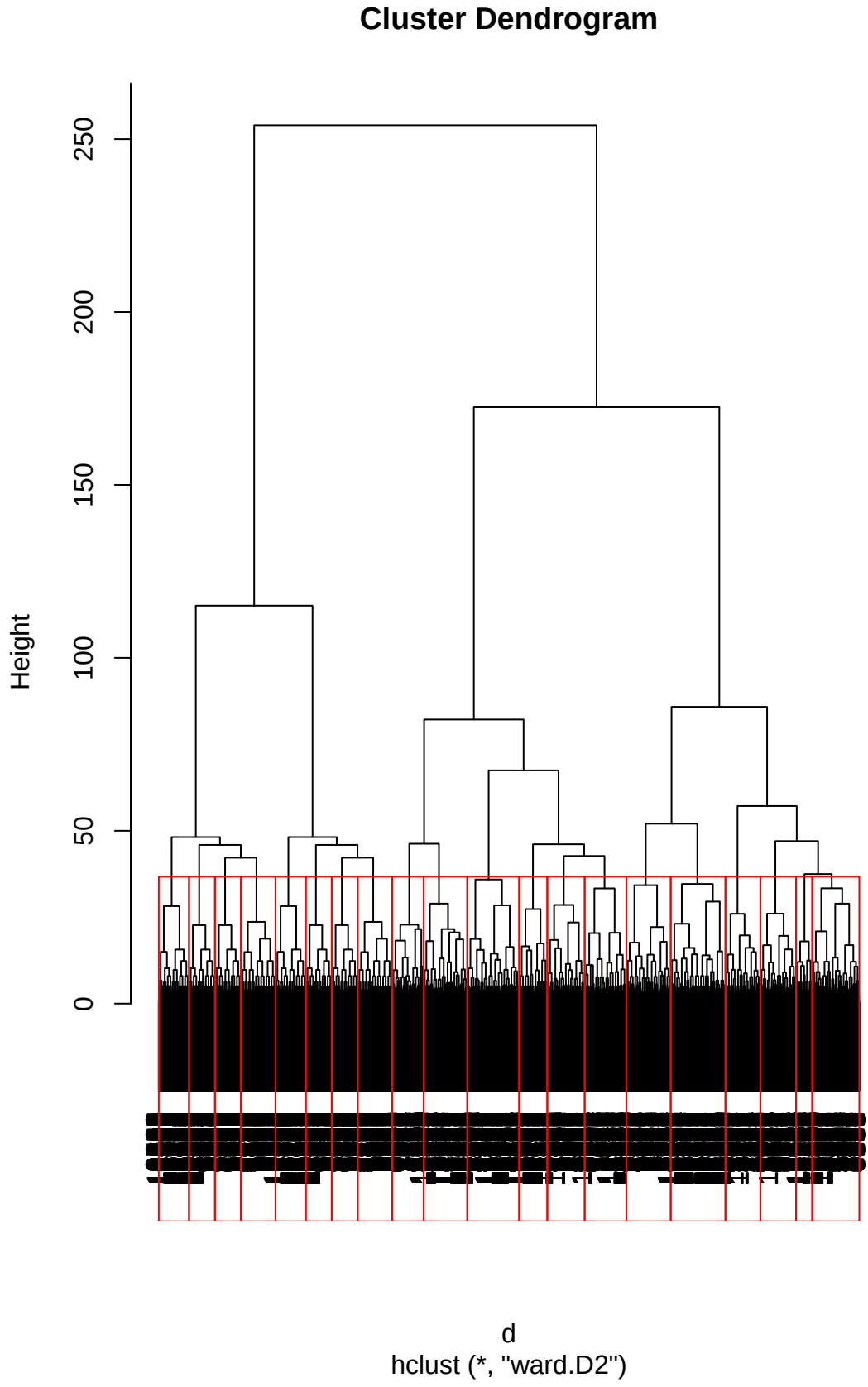
k-means algorithm is not very flexible, and as such is of limited use (except for when vector quantization as above is actually the desired use case). In particular, the parameter k is known to be hard to choose (as discussed above) when not given by external constraints. Another limitation of the algorithm is that it cannot be used with arbitrary distance functions or on non-numerical data.

Hierarchical methods uses a distance matrix as an input for the clustering algorithm. The choice of an appropriate metric will influence the shape of the clusters, as some element may be close to one another according to one distance and farther away according to another. We use the Euclidean distance as an input for the clustering algorithm ward.2D minimum variance criterion minimizes the total within-cluster variance.

```
d <- dist(scaled, method = "manhattan")
H.fit <- hclust(d, method="ward.D2")
```

The clustering output is displayed in a dendrogram **??**.

```
clusters_num = 20
plot(H.fit)
groups <- cutree(H.fit, k=clusters_num)
rect.hclust(H.fit, k=clusters_num, border="red")
```



The clustering performance can be evaluated with the aid of a confusion matrix as follows. Let's look at the groups that have mixed valued of 'class'. Group 1 contains class 'recommend' and also 'very_recom' and 'priority'. Since our idea is relable rows to 'recommend' to increase it's presense, let's check if there is any justification to do this.

```
table(nursery_data$class,groups)
```

```
#>             groups
#>                1    2    3    4    5    6    7    8    9   10   11   12
#>   not_recom    0  640    0  480    0  640    0  480    0  480    0  480
#>   recommend    2    0    0    0    0    0    0    0    0    0    0    0
#>   very_recom 208    0    0    0  100    0    0    0   10    0   10    0
#>   priority   650    0  636    0  416    0  400    0  650    0  562    0
#>   spec_prior  10    0   10    0  176    0  560    0    0    0    8    0
#>             groups
#>               13   14   15   16   17   18   19   20
#>   not_recom    0    0    0    0    0  560    0  560
#>   recommend    0    0    0    0    0    0    0    0
#>   very_recom   0    0    0    0    0    0    0    0
#>   priority   404    0    8  300    0    0  240    0
#>   spec_prior 368  820  800    0 1012    0  280    0
```

Let's find what are the most significant factors that separate group 1 from also mixed group 5. It looks that group has better financial situation but 22% less favorable family structure and 12% better housing situation, with the rest of the attributes beeing close - less that 1%. We could arbitrary decide that it is justified to downgade group 1 grading all the fows to 'recommend' even thogh most of the rows were previously been labeled higher.

```
dif <- colMeans(scaled[groups == 1,]) - colMeans(scaled[groups == 5,])
dif <- dif[order(abs(dif), decreasing = T)]
print(dif)
```

```
#>     finance        form     housing       class     parents    children
#> -1.00000000  0.22487985 -0.12126437 -0.08550262  0.06343100 -0.05783004
#>      health    has_nurs      social
#> -0.03638629 -0.02445020  0.01551724
```

Group 5 has high amount of 'very_recommend' values in addition to 'priority' and some 'special_priority'. Let's find what are the most significant factors that separate group 5 from also mixed similar group 9. Looking at the most influential attribus defining the difference between those groups, it looks that group 5 has less favorable financial situation, but 50% better housing situation and 30% better formal family structure. Again we could arbitrary decide that it is justified to downgrade group 5 down grading all the records in the group as 'very_recommend' even though most of the rows were previously been labeled higher.

```
dif <- colMeans(scaled[groups == 5,]) - colMeans(scaled[groups == 9,])
dif <- dif[order(abs(dif), decreasing = T)]
print(dif)
```

```
#>     finance     housing        form    has_nurs    children      health
#>  1.00000000 -0.56515152 -0.30687219  0.22782011 -0.09110761  0.03878963
#>       class     parents      social
#>  0.03124453  0.01452093  0.00000000
```

Lets fix the groups 1 and 5 by relabelling class according to our conclusions:

```
nursery_data[groups == 1,]$class<- 'recommend'
nursery_data[groups == 5,]$class<- 'very_recom'
```

Lets analyze the result visualizing the distribution of class now. Obviosly the distribution is improved compaired to previous data.

```
prop.table(table(nursery_data$class))
```

```
#>
#>  not_recom  recommend very_recom   priority spec_prior
#> 0.33333333 0.06712963 0.05493827 0.24691358 0.29768519
```

```
ggplot(data = nursery_data, mapping = aes(x = class)) + geom_bar()
```

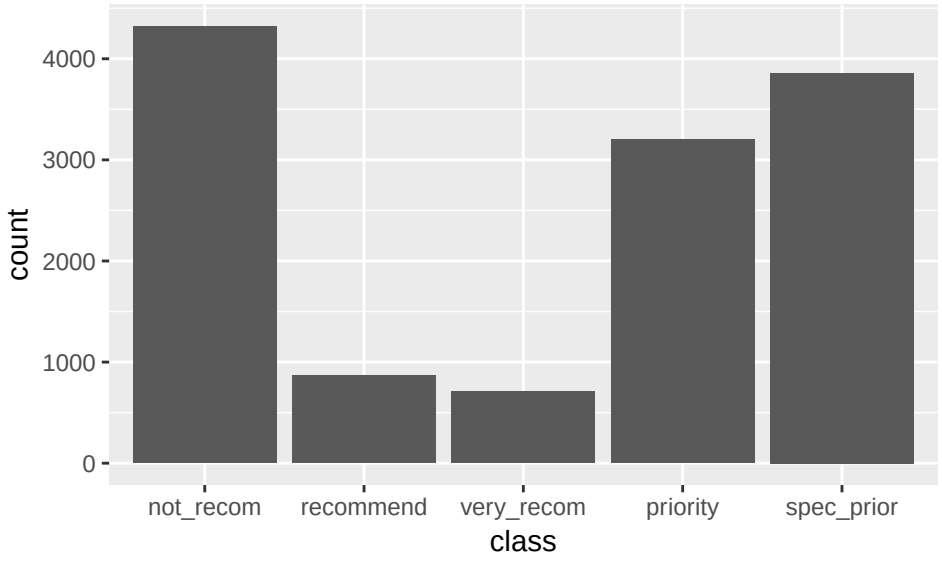


Figure 4: Distribution the Target 'class' Attribute in the Nursery Dataset

Modeling

Random Forest model prediction and evaluation of the original dataset

The Nursery dataset has been split in such a way that train and test sets would have the same distribution of the 'class' attribute. The reason for this stratification strategy is to focus on the priority of an applicants placement in a nursery school rather than an applicants family or social status. We used 75:25 split ratio.

```
train.rows<- createDataPartition(y= nursery_data$class, p=0.75, list = FALSE)
train.data<- nursery_data[train.rows,]
prop.table((table(train.data$class)))
```

```
#>
#>  not_recom  recommend very_recom   priority spec_prior
#> 0.33329904 0.06717416 0.05493262 0.24688818 0.29770600
```

```
test.data<- nursery_data[-train.rows,]
prop.table((table(test.data$class)))
```

```
#>
#>  not_recom  recommend very_recom   priority spec_prior
#> 0.33343625 0.06699599 0.05495523 0.24698981 0.29762272
```

```
library(randomForest)
fitRF2 <- randomForest(
  class~parents+has_nurs+form+children+housing+finance+social+health,
  method="anova",
  data=train.data, importance=TRUE, ntree=1000 )
```

Now let's calculate prediction and it's evaluation using the corrected dataset. As we expected, now all but one of the "very_recom" values of the target "class" attribute were predicted correctly which led to total accuracy of the prediction to 98% with overall balanced precision accross all the predicted "class" values.

```
PredictionRF2 <- predict(fitRF2, test.data)
confMat2 <- table(PredictionRF2,test.data$class)
confMat2
```

```
#>
#> PredictionRF2 not_recom recommend very_recom priority spec_prior
#>    not_recom       1080         0          0        0          0
```

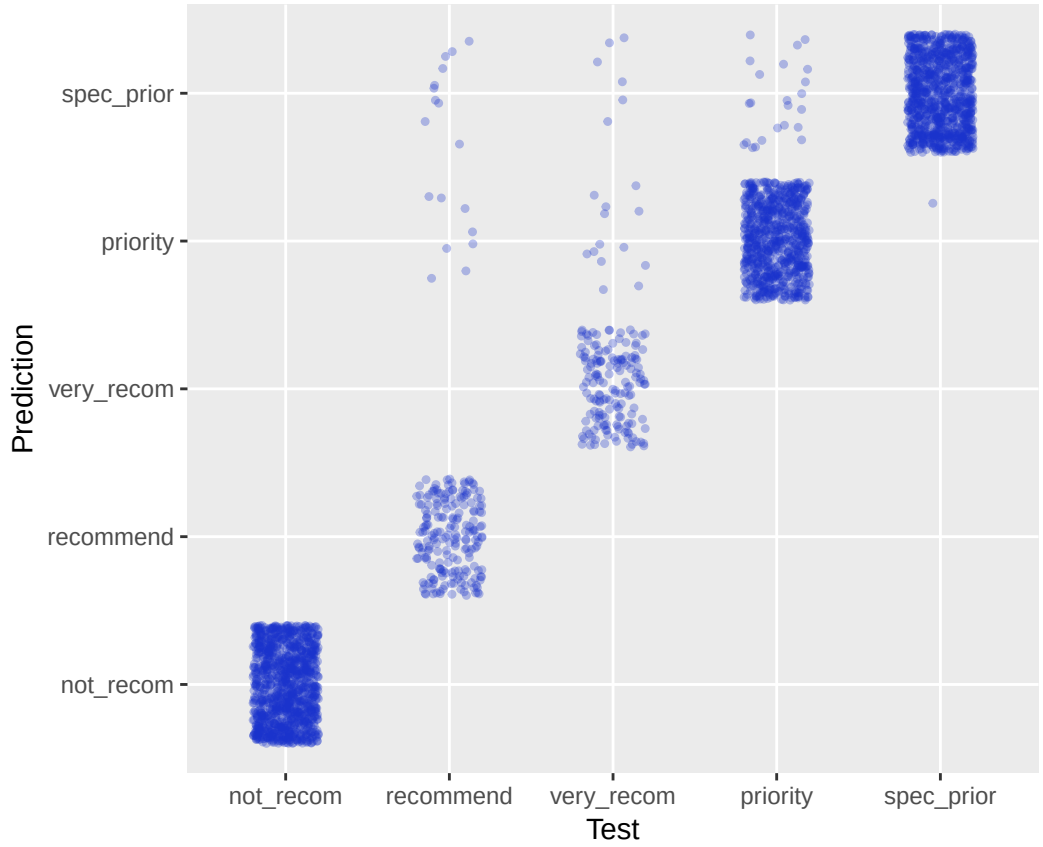


Figure 5: Random Forest Prediction

```
#>    recommend          0       199          0        0          0
#>    very_recom         0         0        159        0          0
#>    priority           0         8         13      777          1
#>    spec_prior         0        10          6       23        963
```

```
accuracy <- sum(diag(confMat2))/sum(confMat2)
cat(sprintf("\nAccuracy=%f", accuracy))
```

```
#>
#> Accuracy=0.981167
```

To visualize the results of the predictions, the code below generates a scatter plot of the Predictor vs Test values (Figure 5). Also a QQ-Plot for Studentized Residuals is generated and presented in Figure **??**.

```
library(ggplot2)
df2 = data.frame(test.data$class, PredictionRF2)
colnames(df2) <- c("Test","Prediction")
ggplot(df2, aes(x = Test, y = Prediction)) +
        geom_jitter(width = 0.2, pch=20, col=rgb(0.1, 0.2, 0.8, 0.3))
```

Conclusion

Through exploring the Nursery dataset and using a classification model to developed our algorithm to predict applicants suitabilty of an admittance within the nursery school system, the evaluation of the model was developed using Random Forest algorithm. Even though the original dataset was clear and did not have missing values, it was unballanced. To overcome this a method called Random Over-Sampling was appplied, which increased the number of instances in the minority class by randomly replicating them in order to present a higher representation of the minority class in the sample. The final model achieved almost 99% accuracy of predictions with overall balanced precision more that 97% accross all the predicted "class" values. The project was a success.

Bibliography

D. B. Dahl. *xtable: Export Tables to LaTeX or HTML*, 2016. URL https://CRAN.R-project.org/package=xtable. R package version 1.8-2. [p3]

M. Olave, V. Rajkovic, and M. Bohanec. An application for admission in public school systems. In *Expert Systems in Public Administration,*, pages 145–160, 1989. [p1]

R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. URL http://www.R-project.org/. ISBN 3-900051-07-0. [p2]

I. H. Witten, E. Frank, and M. A. Hall. *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann series in data management systems. Morgan Kaufmann, Burlington, MA, 3rd ed edition, 2011. ISBN 9780123748560. OCLC: ocn262433473. [p2]

B. Zupan, M. Bohanec, I. Bratko, and J. Demsar. Machine learning by function decomposition. In *ICML*, page 1, 1997. [p1]

Note from the Authors

This file was generated using *The R Journal* style article template, additional information on how to prepare articles for submission is here - Instructions for Authors. The article itself is an executable R Markdown file that could be downloaded from Github with all the necessary artifacts.

Viviane Adohouannon
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/view.php?id=21444

Kate Alexander
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/view.php?id=21524

Diana Azbel
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/view.php?id=20687

Igor Baranov
York University School of Continuing Studies

https://learn.continue.yorku.ca/user/profile.php?id=21219