Ranking Applications for Nursery Schools - Relabeling 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. In this report the dataset was analyzed and the target value imbalance was corrected using clustering. Data was partially relabeled to correct underrepresented values of the target.

Introduction

Nursery dataset (UCI Nursery Data Set) 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.

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.

Background

In our previous report (TFG Lab1 Classification Problem, 2018) we explored the Nursery dataset and using a classification model to developed our algorithm to predict applicants suitability 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 unbalanced. 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 (Wikipedia, Ljubljana, 2018).

During this time according to (Olave et al., 1989) "new housing developments populated by young families, the demand for children's 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. Another research was made later in 1997 (Zupan et al., 1997), slightly improving the original algorithm. Both researches were modeling the original dataset without any attempts to challege it's correctness.

It was thought, after presenting our previous report, that the labeling of the Nursery dataset was affected by human bios, resulting in most labels to be given as extreme - only 'not recommend' or 'highly recommend' suggestions were made. The middle values are almost not presented. It is very important for the algorithm to be more objective, since the proper recommendation would help produce much more reasonable results and help properly distribute such a limited resource as a nursing school placement.

Objective

The objective of this report is to provide a reliable and feasible recommendation algorithm to correct, the discovered previously, imbalance of the dataset "class" target value. 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 relabel the Nursery dataset. The idea was to use clustering methods, for instance (*k*-means clustering, 2018), to group observation in such a way that would allow us to make a conscious decision of the dataset relabeling.

The main tool used in developing was 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 (UCI Nursery Data Set) 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 the original dataset description and had to be extracted from the data.

Data Preparation

Data loading and summary

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 (UCI Nursery Data Set) 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"))
```

	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
```

Fixing Dataset Factor Levels

Very often, especially when plotting data and applying clustering algorithms, we need to reorder the levels of a factor because the default order is alphabetical. Current levels need to be corrected to correspond to the dataset description.

Current levels

```
#> [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 of levels

A direct way of reordering, using standard syntax is as follows:

```
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)])
```

Fixed levels

Here are the corrected levels, now they 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 order to use clustering algorithms, the data should be transformed from categorical to numeric and normalized. First we obtain the numbers matrix 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 (Table 2).

	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
7	1	1	1	1	1	1	3	2	4
8	1	1	1	1	1	1	3	3	4
9	1	1	1	1	1	1	3	1	1
10	1	1	1	1	1	2	1	2	3
11	1	1	1	1	1	2	1	3	4
12	1	1	1	1	1	2	1	1	1
13	1	1	1	1	1	2	2	2	3
14	1	1	1	1	1	2	2	3	4
15	1	1	1	1	1	2	2	1	1

Table 2: `Nursery Data Dataset in numeric format (head)`

Preparing scaled data

In order to use clustering algorithms, the data should be normalized. The following code performs the scaling to 0:1 range and prints the head of the datatset (Table 3):

```
maxs <- apply(data, 2, max)
mins <- apply(data, 2, min)
scaled <- as.data.frame(scale(data, center = mins, scale = maxs - mins))
```

	parents	has_nurs	form	children	housing	finance	social	health	class
1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.25
2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.75
3	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
4	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.50	0.25
5	0.00	0.00	0.00	0.00	0.00	0.00	0.50	1.00	0.75
6	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00
7	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.50	0.75
8	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00	0.75
9	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
10	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.50	0.50
11	0.00	0.00	0.00	0.00	0.00	1.00	0.00	1.00	0.75
12	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00
13	0.00	0.00	0.00	0.00	0.00	1.00	0.50	0.50	0.50
14	0.00	0.00	0.00	0.00	0.00	1.00	0.50	1.00	0.75
15	0.00	0.00	0.00	0.00	0.00	1.00	0.50	0.00	0.00

Table 3: `Scaled Nursery Data Dataset in numeric format (head)`

Data Relabeling

The problem analysis

The target value class of the the nursery dataset is not equally distributed (Table 4). The Figure 1 demonstrates the distribution graphically. As we can see, dataset covers mostly extreme values, middle values are represented poorly.

not_recom	0.33
recommend	0.00
very_recom	0.03
priority	0.33
spec_prior	0.31

Table 4: `Nursery Data Dataset in numeric format (head)`

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

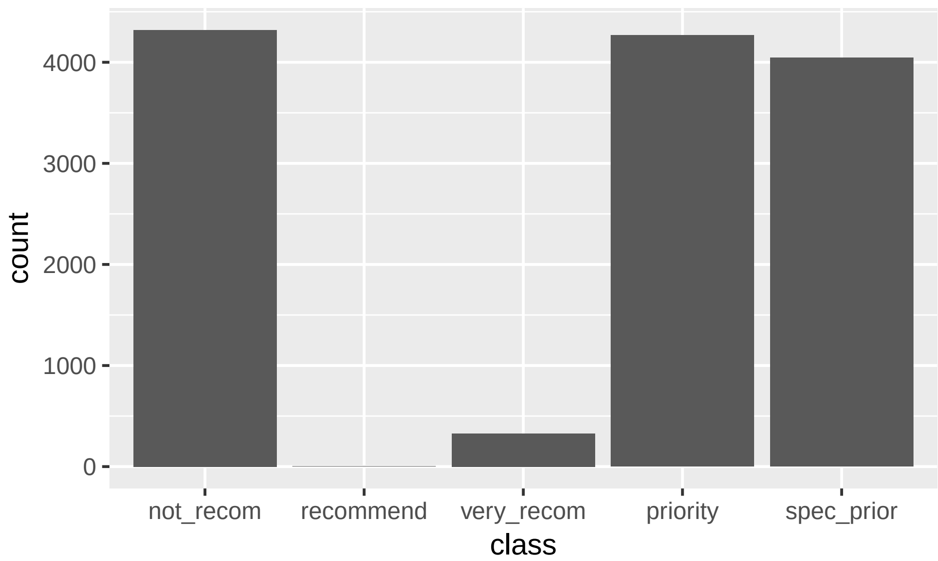

Figure 1: Distribution the Target 'class' Attribute in the Nursery Dataset

It was thought, that the labeling of the Nursery dataset was affected by human bios, resulting in most labels to be given as extreme - mostly 'not _recom', 'priority' and 'special_priority' suggestions were made.

One of the ways to solve this issue is to use cluster analysis, for instance K-Means method (*k*-means clustering, 2018), to group observation in such a way that would allow us to make a conscious decision of the dataset relabeling.

In centroid-based clustering, clusters are represented by a central vector, which may not necessarily be a member of the dataset. When the number of clusters is fixed to k, k-means clustering gives a formal definition as an optimization problem: find the k cluster centers and assign the objects to the nearest cluster center, such that the squared distances from the cluster are minimized.

Figure 2: Elbow Criterion Diagram

Preparation for cluster analysis

First we need to determine number of clusters. Looking at the percentage of variance explained as: a function of the number of clusters, we should choose a number of clusters in order to ensure that too much modeling of the data is not given. More precisely, if one plots the percentage of variance explained by the clusters against the number of clusters, the first clusters will add much more information (explains 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. This method is called the 'elbow criterion'. The diagram presented in Figure 2 demonstrates the 'elbow' curve. From this diagram we decided to use five (5) clusters in our analysis.

```
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=7)
```

Clustering using K-means method

k-means clustering (*k*-means clustering, 2018) is a method of vector quantization, which 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 (noa, 1908).

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.

Resulting cluster centers are presented in Table 5.

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

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

	parents	has_nurs	form	children	housing	finance	social	health	class
1	0.50	0.50	0.50	0.50	0.50	1.00	0.50	0.00	0.00
2	1.00	0.50	0.50	0.50	0.50	1.00	0.50	0.75	0.94
3	0.25	0.50	0.50	0.50	0.50	1.00	0.50	0.75	0.84
4	0.50	0.50	0.50	0.50	0.50	0.00	0.50	0.75	0.84
5	0.50	0.50	0.50	0.50	0.50	0.00	0.50	0.00	0.00

Table 5: K-means Resulting Cluster Centers

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

Analysis of clusters by ‘class’ attribute

Let’s try to explain clusters by the ‘class’. Code below builds a matrix where columns are cluster numbers and rows are target classes. Results of that presented in Table 6. Cluster 1 contains only rows with ‘not_recom’ class. Class 5 contain only rows with ‘not_recom’ and both ‘recommend’ rows presented in the dataset. Clusters 2,3, and 4 have the rest of the ‘class’ values. It is not possible to increase groups ‘recommed’ and ‘highly_recom’ by relabeling other rows. It looks that k-means can’t help us in solving of the problem since the clusters obviously not defined the ‘class’ attribute.

	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

Table 6: K-means Resulting Cluster Centers

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.

Another approach, called connectivity-based clustering, also known as hierarchical clustering, is based on the core idea of objects being more related to nearby objects than to objects farther away. These algorithms connect “objects” to form “clusters” based on their distance. A cluster can be described largely by the maximum distance needed to connect parts of the cluster. At different distances, different clusters will form, which can be represented using a dendrogram which explains where the common name “hierarchical clustering” comes from: these algorithms do not provide a single partitioning of the data set, but instead provide an extensive hierarchy of clusters that merge with each other at certain distances. In a dendrogram, the y-axis marks the distance at which the clusters merge, while the objects are placed along the x-axis such that the clusters don’t mix.

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.

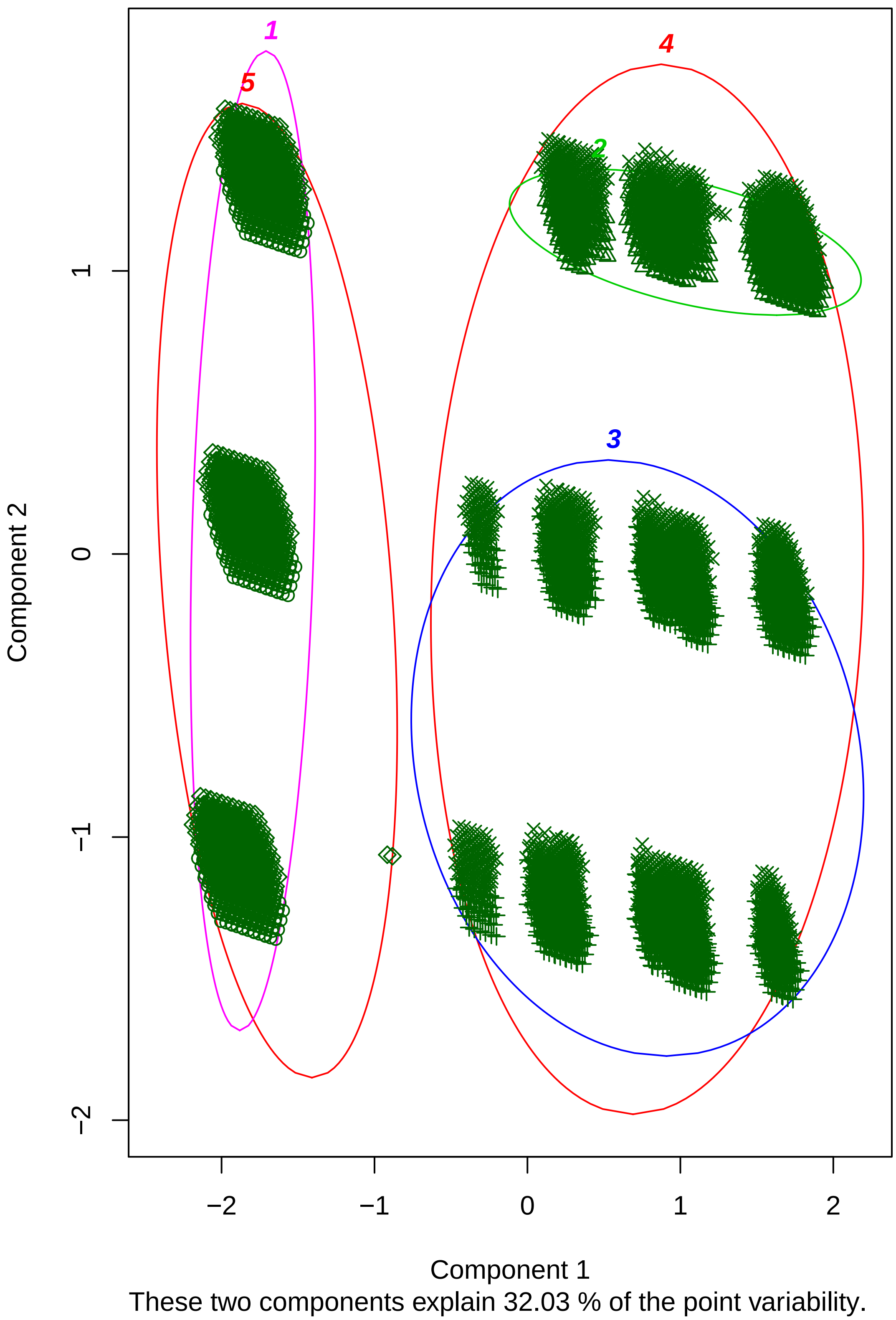

Figure 3: 2D representation of the Cluster solution

We use the Manhattan distance as an input for the clustering algorithm ward. 2D minimum variance criterion minimizes the total within-cluster variance. The reason for that is that we are essentially analyzing a categorical dataset where even scaled data has grouped values, not like in a naturally numeric dataset.

After several tries we decided to set a number of clusters to twenty (20) to make them small but allowing us reliably separate records by 'class' attribute. The code performing hierarchical clustering presented below. Visualization of clustering is displayed in a dendrogram in Figure 4.

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

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

Analysis of Hierarchical Clustering results

The clustering performance can be evaluated with the aid of a confusion matrix as shown in Table 7. 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 relabel rows to 'recommend' to increase its presense, let's check if there is any justification to do this.

	1	2	3	4	5	6	7	8	9	10
not_recom	0	640	0	480	0	640	0	480	0	480
recommend	2	0	0	0	0	0	0	0	0	0
very_recom	208	0	0	0	100	0	0	0	10	0
priority	650	0	636	0	416	0	400	0	650	0
spec_prior	10	0	10	0	176	0	560	0	0	0

	11	12	13	14	15	16	17	18	19	20
not_recom	0	480	0	0	0	0	0	560	0	560
recommend	0	0	0	0	0	0	0	0	0	0
very_recom	10	0	0	0	0	0	0	0	0	0
priority	562	0	404	0	8	300	0	0	240	0
spec_prior	8	0	368	820	800	0	1012	0	280	0

Table 7: `Cluster group centers`

Let's find what are the most significant factors that separate group 1 from also mixed group 5. Code below calculates the different results presented in Table 8. It looks this 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 downgrade group 1 grading all the rows to 'recommend' even though most of the rows were previously been labeled higher.

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

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. Results of the calculations presented in Table 9.

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

Looking at the most influential attributes defining the difference between those groups, it appears that group 5 has less favorable financial situation, but 50% better housing situation and 30% better formal family structure. Again we could arbitrarily 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.

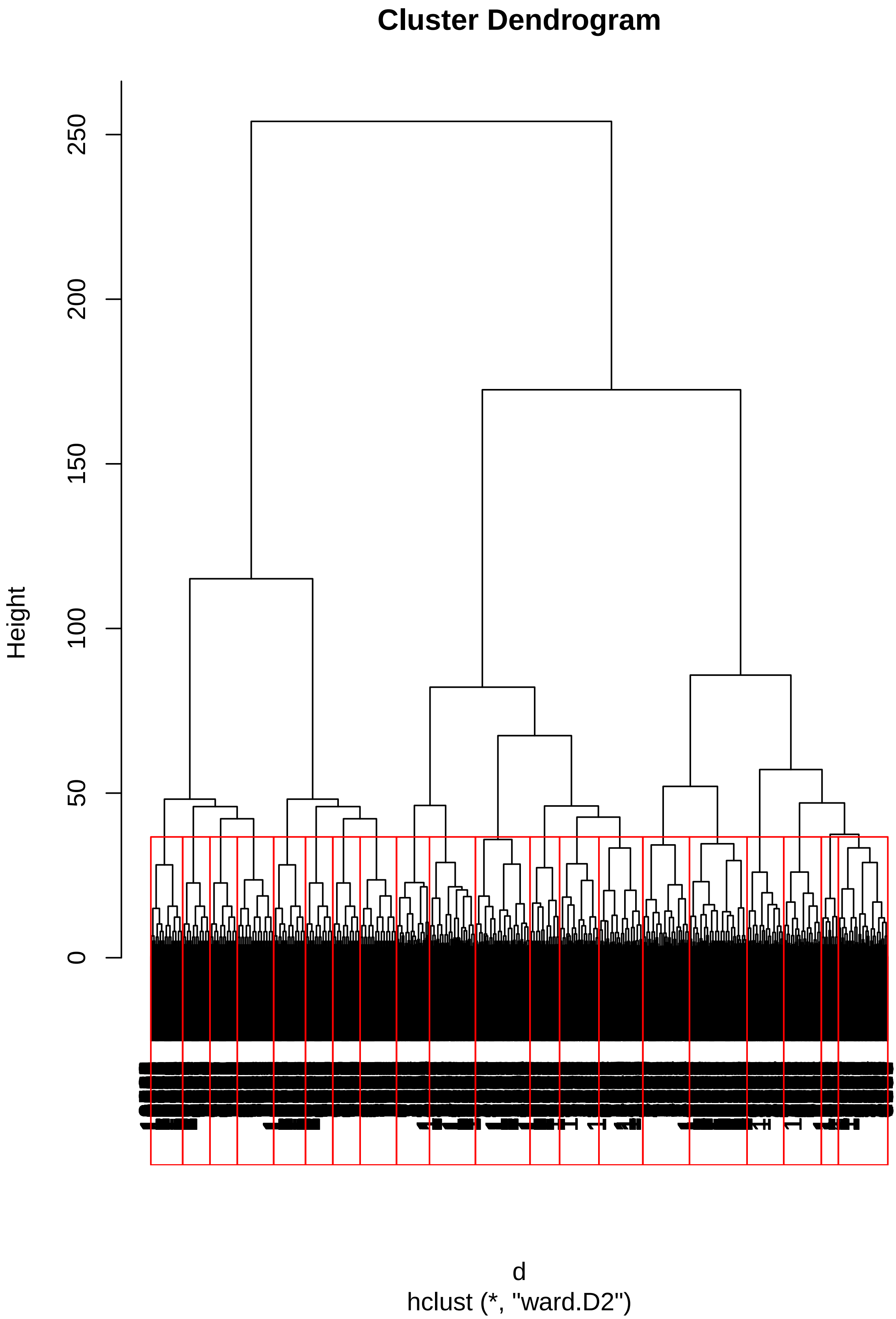

Figure 4: Visual Presentation of a Cluster Denrogram

	Difference
finance	-1.00
form	0.22
housing	-0.12
class	-0.09
parents	0.06
children	-0.06
health	-0.04
has_nurs	-0.02
social	0.02

Table 8: Difference between Clusters 1 and 5

	Difference
finance	1.00
housing	-0.57
form	-0.31
has_nurs	0.23
children	-0.09
health	0.04
class	0.03
parents	0.01
social	0.00

Table 9: Difference between Clusters 5 and 9

Relableling of the Nursery dataset

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

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

Lets analyze the results visualizing the distribution of class now. Obviously the distribution is improved compaired to previous data (Table 10).

not_recom	0.33
recommend	0.07
very_recom	0.05
priority	0.25
spec_prior	0.30

Table 10: Distribution the target 'class' after relabelling

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

Modeling of the Relabeled 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 (Tables 11 and 12). The reason for this stratification strategy is to focus on the priority of an applicants placement in a nursery school rather than an applicant's 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,]


test.data<- nursery_data[-train.rows,]
```

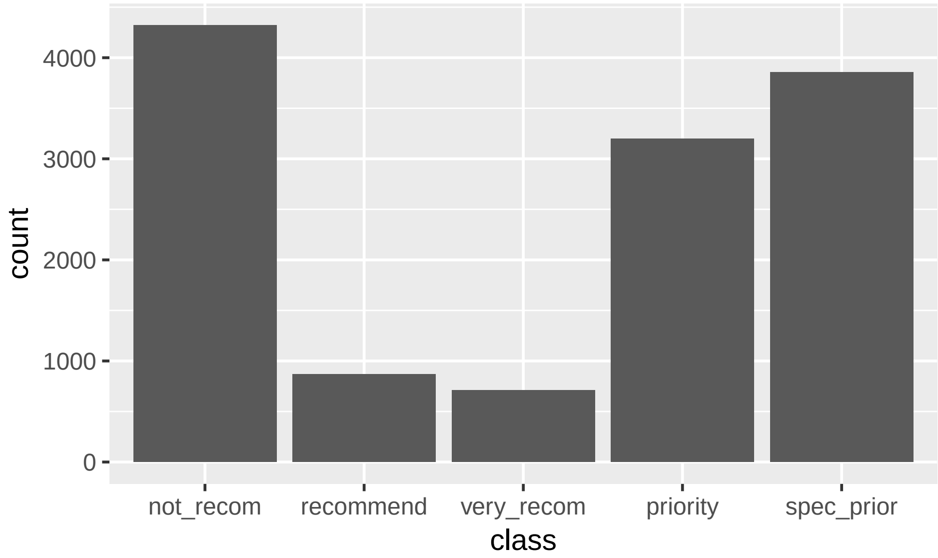

Figure 5: Distribution the Target 'class' Attribute in the Nursery Dataset

not_recom	0.33
recommend	0.07
very_recom	0.05
priority	0.25
spec_prior	0.30

Table 11: `Distribution the target 'class' in the train dataset`

```
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. The code below calculates the predictions, the results are presented in Table 13. As we expected, 90% of all the "recommend" and "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 across all the predicted "class" values. We noticed that this results is similar to accuracy of the original dataset in our previous report (TFG Lab1 Classification Problem, 2018), but the current model has very good precision in the 'recommend' and 'highly_recom'. The dataset correction allowed the RF algorithm to perform well without performing any "tricks" to fix imbalanced datasets like suggested here (Imbalanced Classification Problems, 2017).

```
PredictionRF2 <- predict(fitRF2, test.data)
confMat2 <- table(PredictionRF2,test.data$class)
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 6).

```
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))
```

not_recom	0.33
recommend	0.07
very_recom	0.05
priority	0.25
spec_prior	0.30

Table 12: Distribution the target 'class' in the test dataset

	not_recom	recommend	very_recom	priority	spec_prior
not_recom	1080	0	0	0	0
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

Table 13: Random Forest Pledictor Confusion Matrix

Conclusion

Through exploring the Nursery dataset and using clustering analysis we were able to fix the dataset target value 'class' imbalance. Data was partially relabeled to correct underrepresented values of the target. As the result, 90% of all the "recommend" and "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 across all of the predicted "class" values. Modeling the corrected dataset, we achieved results that are similar to accuracy of the original dataset in our previous report (TFG Lab1 Classification Problem, 2018), but the current model has very good precision in the 'recommend' and 'highly_recom'. The dataset correction allowed the RF algorithm to perform well without performing any "tricks" to fix imbalanced datasets. The project was a success.

Bibliography

Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs. *Journal für die reine und angewandte Mathematik (Crelle's Journal)*, 1908(134), 1908. ISSN 0075-4102, 1435-5345. doi: 10.1515/crll.1908.134.198. URL `https://www.degruyter.com/view/j/crll.1908.issue-134/crll.1908.134.198/crll.1908.134.198.xml`. [p7]

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]

Imbalanced Classification Problems. How to handle Imbalanced Classification Problems in machine learning?, Mar. 2017. URL `https://www.analyticsvidhya.com/blog/2017/03/imbalanced-classification-problem/`. [p13]

k-means clustering. *k*-means clustering, Aug. 2018. URL `https://en.wikipedia.org/w/index.php?title=K-means_clustering&oldid=854136563`. Page Version ID: 854136563. [p2, 6, 7]

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]

TFG Lab1 Classification Problem. Scda1010-lab1: csda1010suma18 - lab exercise 1: classification problem, July 2018. URL `https://github.com/ivbsoftware/scda1010-lab1`. original-date: 2018-07-10T02:16:41Z. [p1, 13, 14]

UCI Nursery Data Set. Uci machine learning repository: nursery data set. URL `http://archive.ics.uci.edu/ml/datasets/Nursery`. [p1, 2, 3]

Wikipedia, Ljubljana. Ljubljana, July 2018. URL `https://en.wikipedia.org/w/index.php?title=Ljubljana&oldid=851232760`. Page Version ID: 851232760. [p1]

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

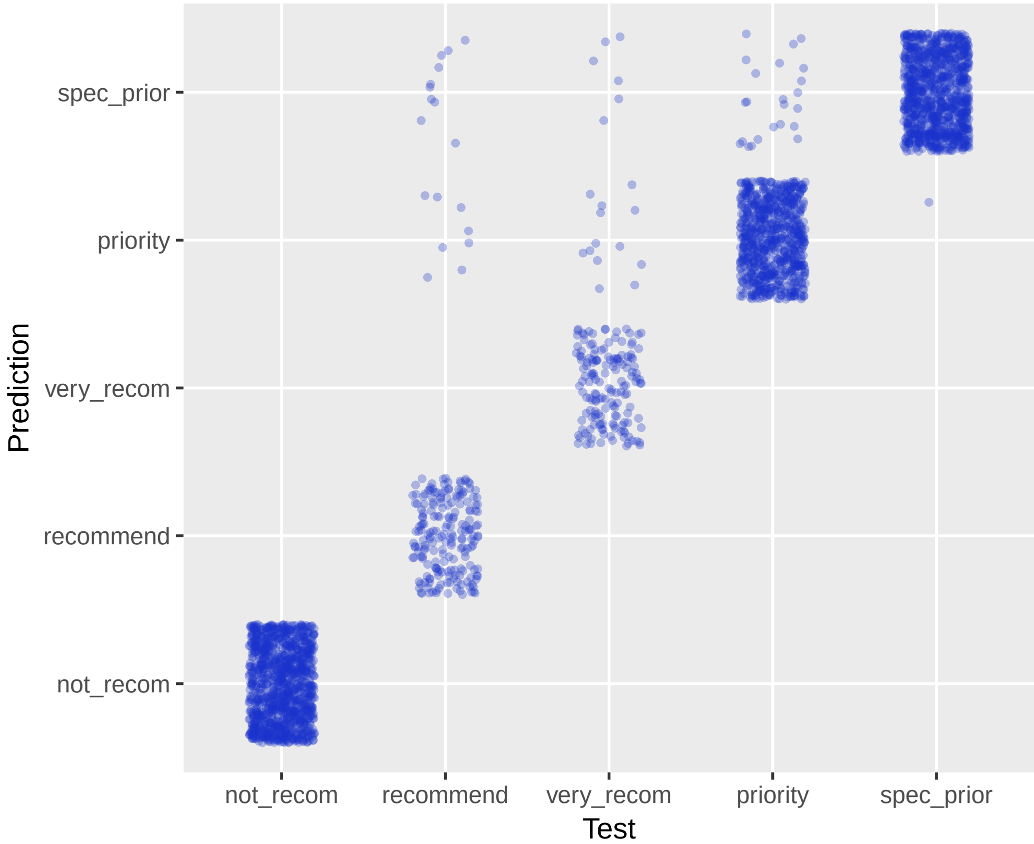

Figure 6: Random Forest Prediction Scatter Plot

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