Ranking Applications for Nursery Schools Classification Problem

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

Introduction

Malala Yousafzai once said, Let us remember: One book, one pen, one child, and one teacher can change the world. Education is important because it gives people the skills of basic literacy and numeracy, as well as the ability to communicate, complete tasks and work with others.

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 (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.

Background

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 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 (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 original dataset description and had to be extracted from the 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(42)
library(ggplot2)
library(rpart)
library(rpart.plot)
library(rattle)
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.

```
library(readr)
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
```

Taking a quick look at the summary it is noticed that target attribute 'class' is presented in the dataset by five values that appear logically in the order of recommendation strength:

- not_recom
- recommend
- very_recom
- priority
- special_prior

Note that 'recommend' value is presented in the dataset by only two rows. Let's look at the distribution of values. The frequency of the segment "recommend" is low (0.00015%) and will not be have effect on the results of prediction algorithms.

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

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

The following code that uses graphical R ggplot library (Wickham, 2009) gererates a graphical presentation of this distribution.

```
ggplot(nursery_data, aes(x=class)) +
  geom_bar(aes(y= (..count..)/sum(..count..)),color="blue",fill=rgb(0.2,0,0.5)) +
  theme(legend.position = "none") +
  scale_y_continuous(labels=scales::percent) +
  labs(x = "class",y="total")
```

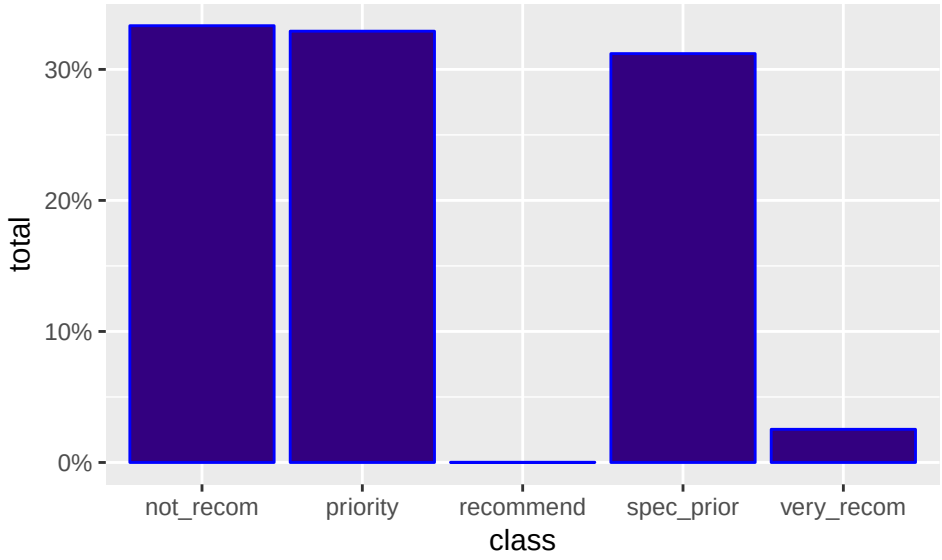

Figure 1: Distribution of Class attribute

Figure 1 shows the targeted attribute with five segments. From the data snapshot, 33% of nursery school applications ended not being recommended. As expected, 'recommend' value has almost no visible presentation. It is worth noticing that presentation of 'very_recom' value is also very low in the dataset and almost ten times lower than other values. This might be an issue that could affect the accuracy of the model and most likely will have to be taken care of later. We can also conclude that there is no missing information and dataset is in good standing to be used.

The following code rendered to (Figure 2) shows distribution of class attribute depending on parents attribute. From this graph, it clearly shows that 'priority' and 'recommendation'

are given to applications from 'pretentious' parents while recommending kids with usual parents to stay home.

```
ggplot(nursery_data,aes(x=class, fill=parents))+
  geom_bar(position="dodge")
```

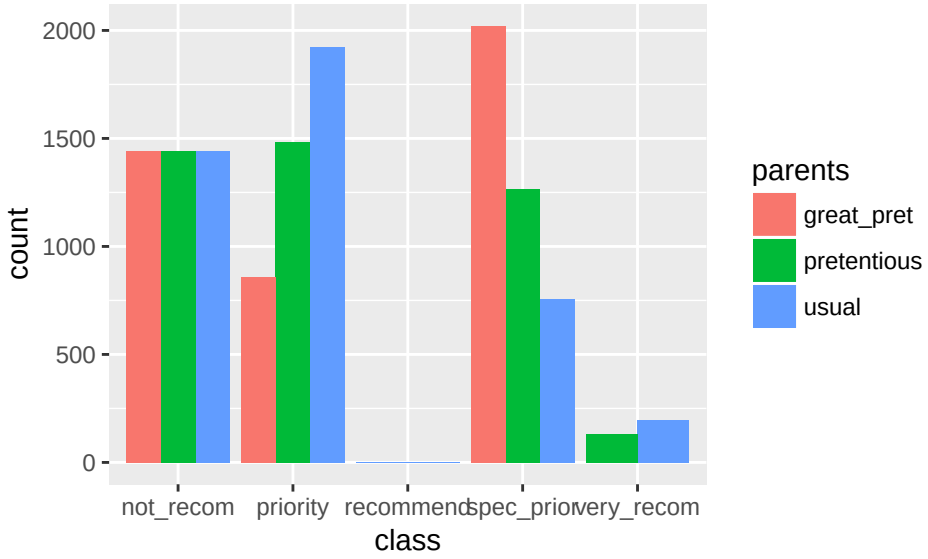

Figure 2: Distribution of Class attribute depending on parents

The following code rendered to (Figure 3) shows distribution of class attribute depending on child 'health' attribute. Being healthy is most important and relevant for parents to have their nursery application being recommended. Parents having strong and good social situation are more likely to have their application recommended than the ones who do not.

```
nursery_data$health <- as.factor(nursery_data$health)
ggplot(data = nursery_data, mapping = aes(x = class, fill = health)) +
  geom_bar()
```

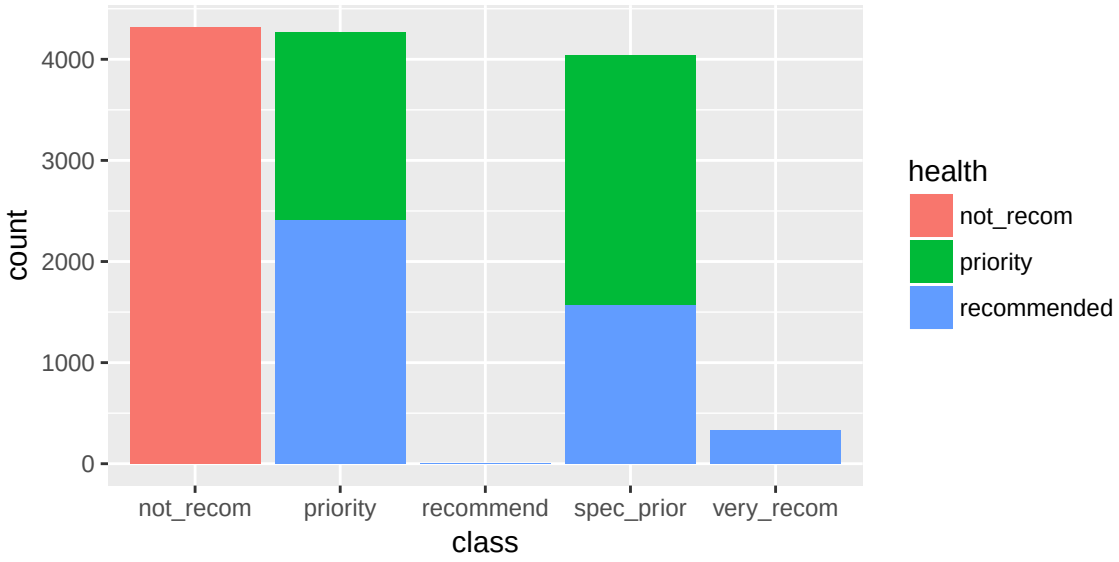

Figure 3: Distribution of Class attribute depending on health

To analyze distribution of class attribute depending on 'social' and 'parents' attributes the following code renders (Figure 4).

```
ggplot(nursery_data, aes(class, fill=social)) +
  geom_bar(aes(y = (..count..)/sum(..count..)), alpha=0.9) +
  facet_wrap(~parents) +
  scale_fill_brewer(palette = "Dark2", direction = -1) +
  scale_y_continuous(labels=scales:::percent, breaks=seq(0,0.4,0.05)) +
  ylab("Percentage") +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

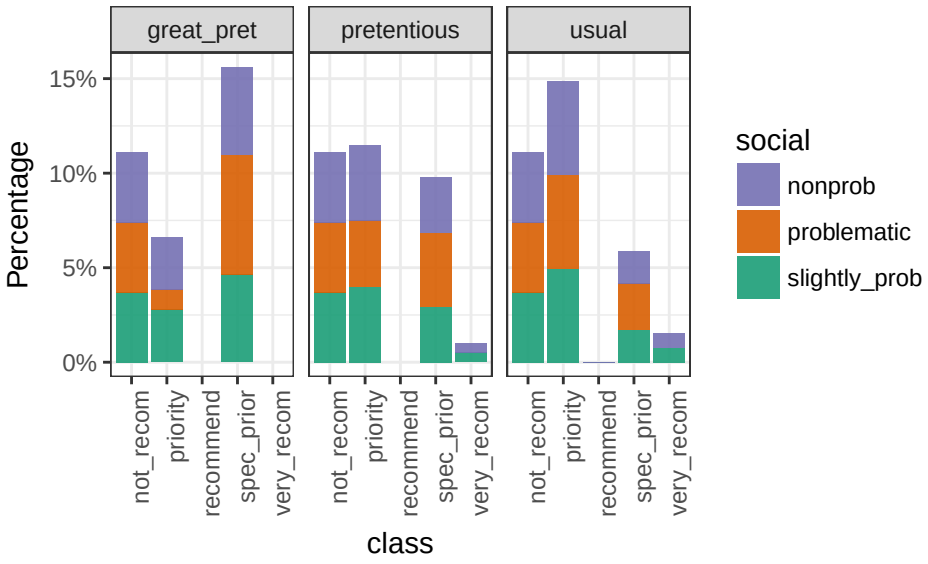

Figure 4: Distribution of Class attribute depending on income

The following code (Figure 5) renders distribution of 'class' attribute depending on 'has_nurs' and 'form' attribute. Parents in segments "proper" and "less_proper" appear to have similar distribution and their application taken as priority.

```
ggplot(nursery_data, aes(class, fill=form)) +
  geom_bar(aes(y = (..count..)/sum(..count..)), alpha=0.9) +
  facet_wrap(~has_nurs) +
  scale_fill_brewer(palette = "Dark2", direction = -1) +
  scale_y_continuous(labels=scales:::percent, breaks=seq(0,0.4,0.05)) +
  ylab("Percentage") +
  theme_bw() +
  theme(plot.title = element_text(hjust = 0.5)) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

To analyze the interdependance of different attributes and their influence on the target "class" attribute does not provide as a clear picture of the importance of those attributes. The exclusion of the 'health' attribute seems to have a visible correlation with the target. The more sophisticated method should be applied to evaluate the attribute importance.

Modeling

Splitting the dataset into train and test

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.

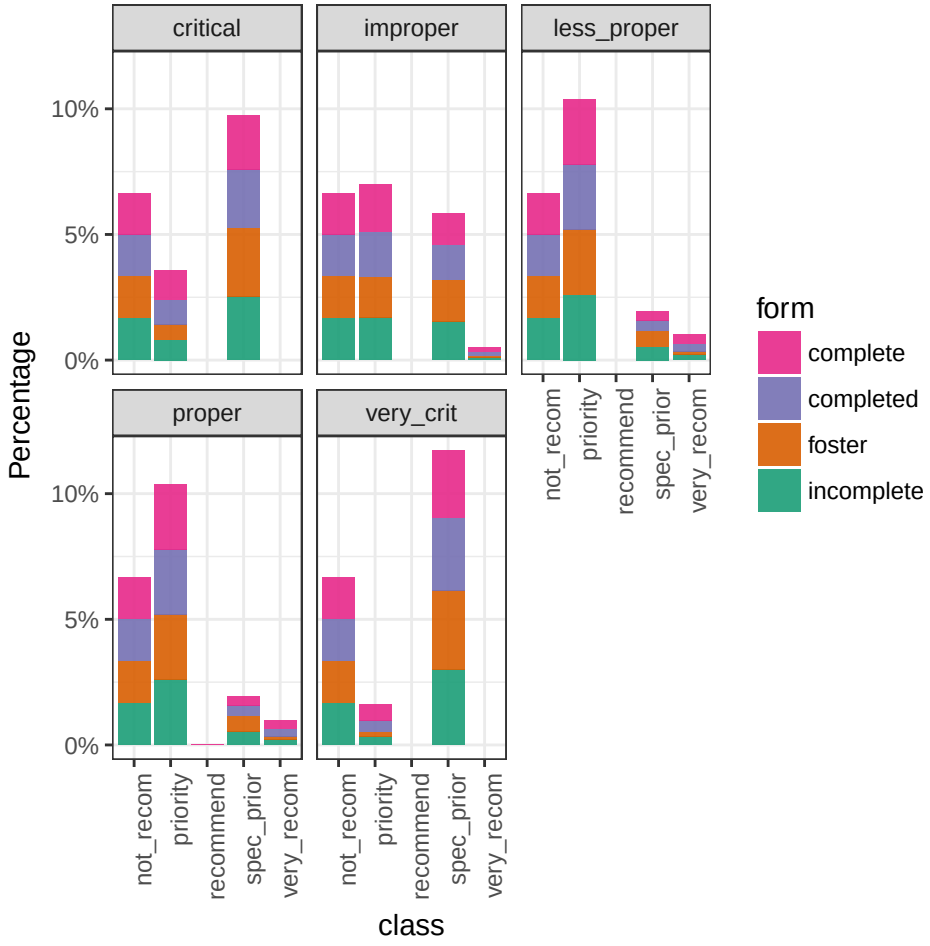

Figure 5: Distribution of Class attribute depending on nursery

```
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     priority    recommend   spec_prior   very_recom
#> 0.3332990433 0.3291842403 0.0002057402 0.3120049378 0.0253060385

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

#>
#>  not_recom   priority  recommend spec_prior very_recom
#> 0.33343625 0.32911392 0.00000000 0.31213337 0.02531646
```

The code below (Figure 6) renders distribution of 'class' attribute depending on 'nursery' attribute. Not suprisingly, the test set did not get any of rows with "recommend" class attribute.

```
ggplot(test.data, aes(x=as.factor(class))) +
  geom_bar(aes(y = (..count..)/sum(..count..)),width=0.4,
  color="red", fill=rgb(0.9,1,0.7) )+theme(legend.position = "none") +
  labs(x = "class",y="total")+scale_y_continuous(labels=scales:::percent)
```

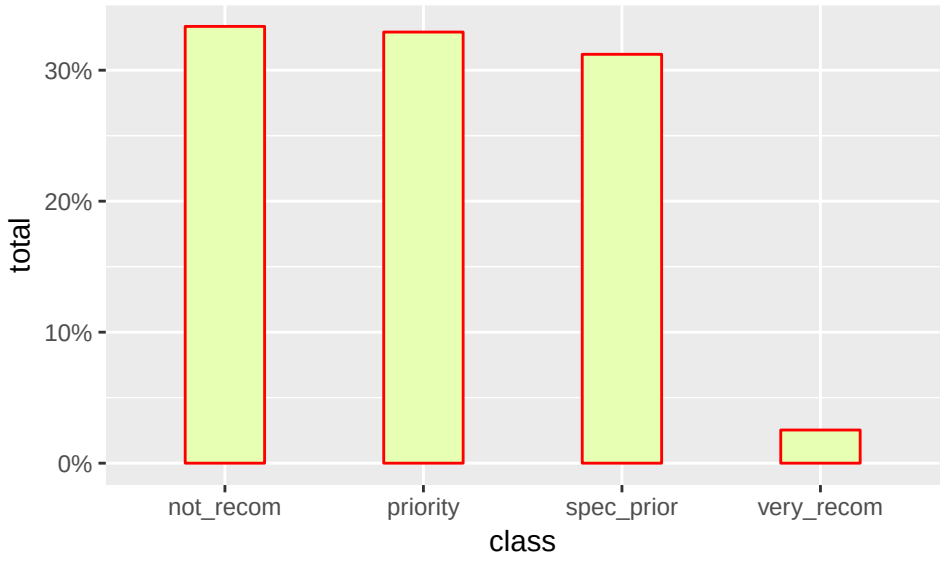

Figure 6: Distribution of Class attribute depending on nursery

Decision Tree model fit

As a first step we have used a Decision Tree model (Therneau and Atkinson, 2018). Though this model is known to be computationally fast but not very precise, we have use all default parameters and all attributes of the train dataset in the resulting Decision Tree presented in Figure 7.

```
fitdt <- rpart(
  as.factor(class)~parents+has_nurs+form+children+housing+finance+social+health,
  method="class", data=train.data)
fancyRpartPlot(fitdt, main="", sub="")
```

Decision Tree model evaluation

This Decision Tree favours the following attributes in order of their importance for the prediction of the target attribute: health, has_nurs, parents. It does not consider the rest of the attributes as important. Let's apply the model to the test set and evaluate accuracy of the predictions.

```
dtPrediction <- predict(fitdt, test.data, type = "class")

confMat <- table(dtPrediction,test.data$class)
confMat

#>
#> dtPrediction not_recom priority recommend spec_prior very_recom
#>   not_recom       1080        0         0          0          0
#>   priority           0      836         0         96         82
#>   recommend          0        0         0          0          0
#>   spec_prior         0      230         0        915          0
#>   very_recom         0        0         0          0          0

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

#>
#> Accuracy=0.874035
```

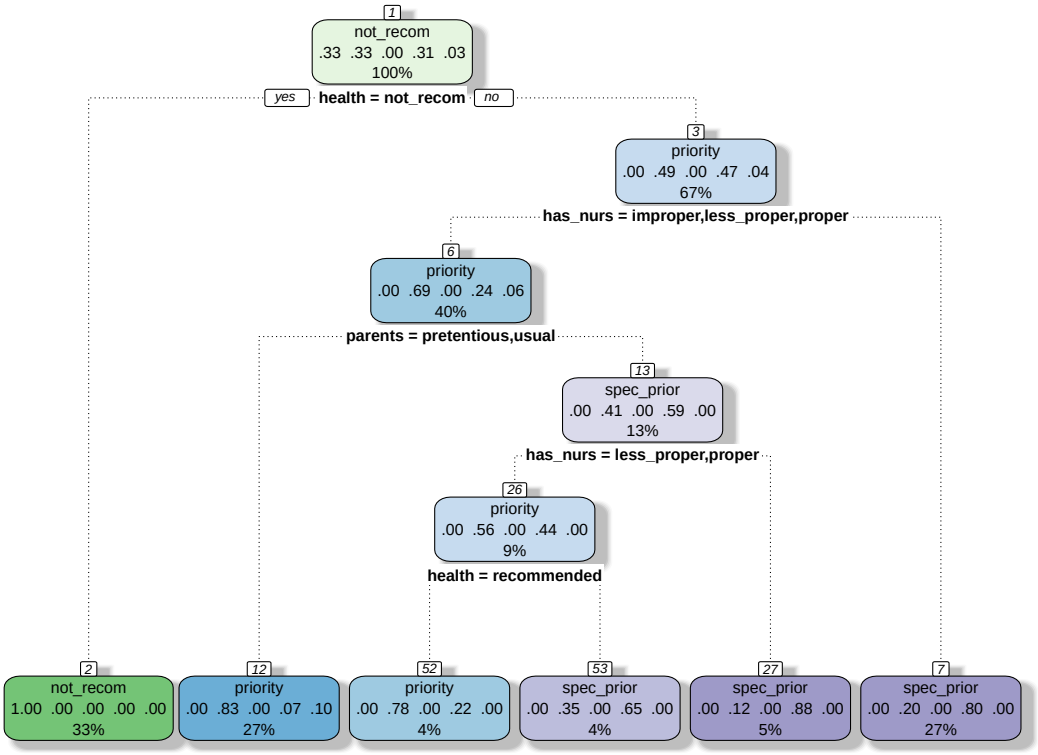

Figure 7: Decision tree diagram

As we can see, the models predicted all "not_recom" values correctly, wrongly predicted all "very_recom" values and had partial success predicting "priority" and "special_priority" values. The overall accuracy is 87%, but the model is not precise and not usable.

Random Forest model fit

Next step is to use more advanced but more computationally demanding Random Forest method (Liaw and Wiener, 2002):

```
library(randomForest)
fitRF1 <- randomForest(
  as.factor(class)~parents+has_nurs+form+children+housing+finance+social+health,
  data=train.data, importance=TRUE, ntree=1000)

varImpPlot(fitRF1, main="")
```

Importance of the dataset attributes for the prediction of the "class" attribute shown in Figure 8. Analyzing this chart we conclude that the proper order of attributes importance for the prediction of the target attribute is: health, has_nurs, parents, housing, social, children. The last two attributes are less important. This contradicts the original analysis (UCI Nursery Data Set) where it is assumed that parents and finance are the most important attributes and health is the least. The only way to solve this dilema is to calculate accuracy of our model.

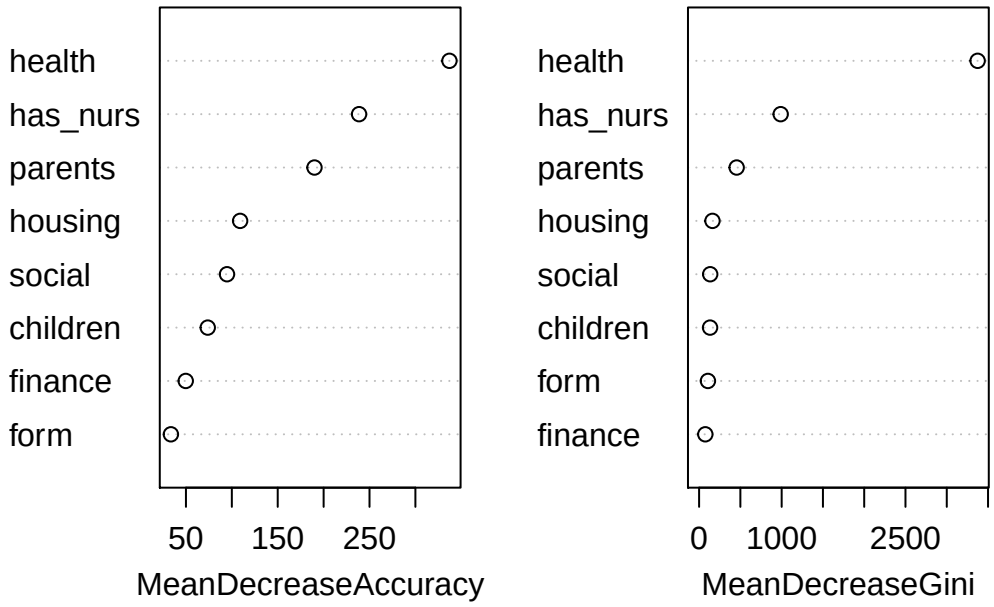

Figure 8: Importance of the dataset attributes for the prediction of the 'class' attribute

Random Forest model prediction and evaluation

```
PredictionRF1 <- predict(fitRF1, test.data)

confMat <- table(PredictionRF1,test.data$class)
confMat

#>
#> PredictionRF1 not_recom priority recommend spec_prior very_recom
#>    not_recom       1080        0         0          0          0
#>    priority           0     1046         0         14         37
#>    recommend          0        0         0          0          0
#>    spec_prior         0       20         0        997          0
#>    very_recom         0        0         0          0         45

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

#>
#> Accuracy=0.978080
```

The models predicted "not_recom" values correctly, "priority" and "special_priority" values were predicted with about 90% accuracy. As expected, "very_recom" value was predicted poorly - with almost 40% error. The overall accuracy is much better at 97.8%, but the model is still not precise and not usable.

Imballanced Classification Correction

To improve the precision of the model we need to take care of the imballanced distribution of rows with target value "very_recom". There are several techniques to correct the problem of low accuracy in regards to predicting "very_recom" value of target attribute "class" (Imbalanced Classification Problems, 2017). One particular method that we decided to employ is method called Random Over-Sampling, which increases 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. This method leads to no information loss, but increases the likelihood of overfitting since it replicates the minority class events. To decrease the latter, we will apply this technique only to train data. As you can see from the output of the code below, the portion of "very_recom" value of the "class" attribute in new train2.data dataset is now about 30% comparing to previous 7% of the rest of the values (excluding "recommend" of course).

```
vr <- train.data[train.data[,"class"] == "very_recom", ]
train2.data <- train.data
for (i in 1:3) {
  train2.data <- rbind(train2.data, vr)
}
prop.table((table(train2.data$class)))
```

```
#>
#>    not_recom     priority    recommend   spec_prior   very_recom
#> 0.3097810498 0.3059565924 0.0001912229 0.2899894827 0.0940816522
```

Corrected Random Forest model prediction and evaluation

```
library(randomForest)
fitRF2 <- randomForest(
  as.factor(class)~parents+has_nurs+form+children+housing+finance+social+health,
  data=train2.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 99% with overall balanced precision more that 97% accross all the predicted "class" values.

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

```
#>
#> PredictionRF2 not_recom priority recommend spec_prior very_recom
#>    not_recom       1080        0         0          0          0
#>    priority           0     1036         0         15          1
#>    recommend          0        0         0          0          0
#>    spec_prior         0       28         0        996          0
#>    very_recom         0        2         0          0         81
```

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

```
#>
#> Accuracy=0.985798
```

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]

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/. [p11]

A. Liaw and M. Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002. URL https://CRAN.R-project.org/doc/Rnews/. [p9]

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]

T. Therneau and B. Atkinson. *rpart: Recursive Partitioning and Regression Trees*, 2018. URL https://CRAN.R-project.org/package=rpart. R package version 4.1-13. [p8]

UCI Nursery Data Set. Uci machine learning repository: nursery data set. URL http://archive.ics.uci.edu/ml/datasets/Nursery. [p1, 2, 3, 9]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York, 2009. ISBN 978-0-387-98140-6. URL http://ggplot2.org. [p4]

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

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