Lab block 2

johmo870, nisra674

2024-12-07

Statement of Contribution

- Nisal Amashan(nisra674) - Assignment 1, Assignment 3, Report
- John Möller (johmo870) - Assignment 2, Assignment 4, Report

Assignment 2

Data import and cleaning

The categories specified in the data compared to the instructions differed. Here's a summary of the differences:

- There are a number of names without dashes in the instructions but that have dashes in the csv, those are: blue-collar, self-employed.
- A number of columns had unknown as a category, specifically those between marital to to contact and poutcome.
- poutcome had 'other' in csv which is assumed in this report to be the same as 'nonexistent' given this category doesn't come up.
- Day of week column does not exist in csv.
- There exists a column called "day" which is not specified in the instruction. It is presumed in the report that this is the numbered day of the month.
- Not previously contacted in pdays is encoded as -1 in the csv as opposed to 999 as it is described in the instructions.

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ---------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```
df_raw <- read_delim(file = "data/bank-full.csv", delim = ";")
```

```
## Rows: 45211 Columns: 17
## -- Column specification --------------------------------------------------------
## Delimiter: ";"
## chr (10): job, marital, education, default, housing, loan, contact, month, p...
## dbl  (7): age, balance, day, duration, campaign, pdays, previous
##
```

```
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
df <- df_raw
df$job <- factor(df$job, levels = c('admin.','blue-collar','entrepreneur','housemaid','management','reti
df$marital <- factor(df$marital, levels = c('divorced','married','single','unknown'))
df$education <- factor(df$education, levels = c('primary', 'secondary', 'tertiary', 'unknown'))
df$default <- factor(df$default, levels = c('no', 'yes', 'unknown'))
df$housing <- factor(df$housing, levels = c('no', 'yes', 'unknown'))
df$loan <- factor(df$loan, levels = c('no', 'yes', 'unknown'))
df$contact <- factor(df$contact, levels = c('cellular', 'telephone', 'unknown'))
df$month <- factor(df$month, levels = c('jan','feb', 'mar', 'apr', 'may', 'jun', 'jul', 'aug', 'sep', '
#df$day <- factor(df$day, levels = c('mon', 'tue', 'wed', 'thu', 'fri', 'unknown'))
df$poutcome <- factor(df$poutcome, levels = c('failure', 'other', 'success', 'unknown'))
df$y <- factor(df$y, levels = c('yes', 'no'))
colSums(is.na(df))
```

```
##       age       job   marital education   default   balance   housing      loan
##         0         0         0         0         0         0         0         0
##   contact       day     month  duration  campaign     pdays  previous  poutcome
##         0         0         0         0         0         0         0         0
##         y
##         0
```

Task 1

Let's remove duration

```
data <- select(df, -duration)
names(data)
```

```
##  [1] "age"       "job"       "marital"   "education" "default"   "balance"
##  [7] "housing"   "loan"      "contact"   "day"       "month"     "campaign"
## [13] "pdays"     "previous"  "poutcome"  "y"
```

Let's divide the data as specified in lecture 2a.

```
n=dim(data)[1]
set.seed(12345)
id=sample(1:n, floor(n*0.4))
train=data[id,]
id1=setdiff(1:n, id)
set.seed(12345)
id2=sample(id1, floor(n*0.3))
valid=data[id2,]
id3=setdiff(id1,id2)
test=data[id3,]
```

Task 2

Importing randomForest library

```
library(tree)
```

Task 2a Decision Tree with default settings

```
tree_default <- tree(y ~., data = train)
summary(tree_default)
```

```
##
## Classification tree:
## tree(formula = y ~ ., data = train)
## Variables actually used in tree construction:
## [1] "poutcome" "month"    "contact"  "housing"
## Number of terminal nodes:  6
## Residual mean deviance:  0.6022 = 10890 / 18080
## Misclassification error rate: 0.1048 = 1896 / 18084
```

Task 2b Decision Tree with smallest allowed node size equal to 7000

```
tree_node_7000 <- tree(y ~., data = train, control = tree.control(nobs = nrow(train), minsize = 7000))
summary(tree_node_7000)
```

```
##
## Classification tree:
## tree(formula = y ~ ., data = train, control = tree.control(nobs = nrow(train),
##     minsize = 7000))
## Variables actually used in tree construction:
## [1] "poutcome" "month"    "contact"
## Number of terminal nodes:  5
## Residual mean deviance:  0.6097 = 11020 / 18080
## Misclassification error rate: 0.1048 = 1896 / 18084
```

Task 2c Decision Tree with minimum deviance 0.0005

```
tree_deviance_0005 <- tree(y ~., data = train, control = tree.control(nobs = nrow(train), mindev = 0.000
summary(tree_deviance_0005)
```

```
##
## Classification tree:
## tree(formula = y ~ ., data = train, control = tree.control(nobs = nrow(train),
##     mindev = 5e-04))
## Variables actually used in tree construction:
##  [1] "poutcome"  "pdays"     "job"       "month"     "previous"  "day"
##  [7] "education" "contact"   "balance"   "age"       "marital"   "housing"
## [13] "campaign"
## Number of terminal nodes:  122
## Residual mean deviance:  0.5213 = 9363 / 17960
## Misclassification error rate: 0.09362 = 1693 / 18084
```

Task 2 discussion

Among the three models, Model 3 (minimum deviance 0.0005) is the best. It achieves the lowest residual mean deviance (0.5213) and misclassification error rate (0.09362), indicating it fits the data better and generalizes well despite having 122 terminal nodes. This suggests that the model, while more complex, captures subtle patterns in the data, improving performance.

Model 1 (default settings), with 6 terminal nodes, offers a simpler tree but has a higher error rate (0.1048) and deviance (0.6022). It strikes a balance between simplicity and performance but doesn't perform as well

as Model 3.

Model 2 (minimum node size 7000), with 5 terminal nodes, results in the same misclassification error rate as the default model (0.1048) but with a slightly higher deviance (0.6097). Restricting the tree's complexity by setting a large minimum node size didn't improve performance and likely prevented the model from capturing important data patterns.

In summary, Model 3 provides the best trade-off between fitting the data and minimizing error, while Models 1 and 2 either simplify the tree too much or do not capture enough complexity to improve performance.