Code ▾

Dmitrii Obideiko

Part 3 - Classification

Hide

```
# Load data
library(readr)
data <- read_csv("bank-full.csv")
```

```
Rows: 45211 Columns: 17── Column specification ──────────────────────────────────────────
──────────────────────────────────────────────────────────────────
Delimiter: ","
chr (10): job, marital, education, default, housing, loan, contact, month, poutcome, y
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.
```

Hide

```
# Display infor about data
str(data)
```

```
spc_tbl_ [45,211 × 17] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
 $ age      : num [1:45211] 58 44 33 47 33 35 28 42 58 43 ...
 $ job      : chr [1:45211] "management" "technician" "entrepreneur" "blue-collar" ...
 $ marital  : chr [1:45211] "married" "single" "married" "married" ...
 $ education: chr [1:45211] "tertiary" "secondary" "secondary" "unknown" ...
 $ default  : chr [1:45211] "no" "no" "no" "no" ...
 $ balance  : num [1:45211] 2143 29 2 1506 1 ...
 $ housing  : chr [1:45211] "yes" "yes" "yes" "yes" ...
 $ loan     : chr [1:45211] "no" "no" "yes" "no" ...
 $ contact  : chr [1:45211] "unknown" "unknown" "unknown" "unknown" ...
 $ day      : num [1:45211] 5 5 5 5 5 5 5 5 5 5 ...
 $ month    : chr [1:45211] "may" "may" "may" "may" ...
 $ duration : num [1:45211] 261 151 76 92 198 139 217 380 50 55 ...
 $ campaign : num [1:45211] 1 1 1 1 1 1 1 1 1 1 ...
 $ pdays    : num [1:45211] -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
 $ previous : num [1:45211] 0 0 0 0 0 0 0 0 0 0 ...
 $ poutcome : chr [1:45211] "unknown" "unknown" "unknown" "unknown" ...
 $ y        : chr [1:45211] "no" "no" "no" "no" ...
 - attr(*, "spec")=
  .. cols(
  ..   age = col_double(),
  ..   job = col_character(),
  ..   marital = col_character(),
  ..   education = col_character(),
  ..   default = col_character(),
  ..   balance = col_double(),
  ..   housing = col_character(),
  ..   loan = col_character(),
  ..   contact = col_character(),
  ..   day = col_double(),
  ..   month = col_character(),
  ..   duration = col_double(),
  ..   campaign = col_double(),
  ..   pdays = col_double(),
  ..   previous = col_double(),
  ..   poutcome = col_character(),
  ..   y = col_character()
  .. )
 - attr(*, "problems")=<externalptr>
```

Hide

```
# Create a new variable that keeps track of the number of people who subscribed
data$subscribed <-ifelse(data$y == 'yes', 1, 0)
# set seed and seperate our data set into 80% train data and 20% test data
set.seed(1234)
# find train index
i <- sample(1:nrow(data), nrow(bank.full)*0.8, replace=FALSE)
trainData <- data[i,]
testData <- data[-i,]
# Convert categorical variables into factors
testData$age <- factor(testData$age)
testData$age <- as.numeric(as.character(testData$age))
testData$job <- factor(testData$job)
testData$marital <- factor(testData$marital)
```

Hide

```
# Explore the training data statistically and graphically
str(testData)
```

```
tibble [9,043 × 18] (S3: tbl_df/tbl/data.frame)
 $ age       : num [1:9043] 33 47 33 56 51 57 37 44 54 58 ...
 $ job       : Factor w/ 12 levels "admin.","blue-collar",..: 3 2 12 5 5 10 1 8 6 6 ...
 $ marital   : Factor w/ 3 levels "divorced","married",..: 2 2 3 2 2 1 3 1 2 2 ...
 $ education : chr [1:9043] "secondary" "unknown" "unknown" "tertiary" ...
 $ default   : chr [1:9043] "no" "no" "no" "no" ...
 $ balance   : num [1:9043] 2 1506 1 779 10635 ...
 $ housing   : chr [1:9043] "yes" "yes" "no" "yes" ...
 $ loan      : chr [1:9043] "yes" "no" "no" "no" ...
 $ contact   : chr [1:9043] "unknown" "unknown" "unknown" "unknown" ...
 $ day       : num [1:9043] 5 5 5 5 5 5 5 5 5 5 ...
 $ month     : chr [1:9043] "may" "may" "may" "may" ...
 $ duration  : num [1:9043] 76 92 198 164 336 ...
 $ campaign  : num [1:9043] 1 1 1 1 1 1 1 1 1 1 ...
 $ pdays     : num [1:9043] -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
 $ previous  : num [1:9043] 0 0 0 0 0 0 0 0 0 0 ...
 $ poutcome  : chr [1:9043] "unknown" "unknown" "unknown" "unknown" ...
 $ y         : chr [1:9043] "no" "no" "no" "no" ...
 $ subscribed: num [1:9043] 0 0 0 0 0 0 0 0 0 0 ...
```

Hide

```
head(testData)
```

a…	job	marital	education	default	balance	housing	loan	contact	d…
<dbl>	<fctr>	<fctr>	<chr>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>
33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5
47	blue-collar	married	unknown	no	1506	yes	no	unknown	5
33	unknown	single	unknown	no	1	no	no	unknown	5

a...	job	marital	education	default	balance	housing	loan	contact	d...
<dbl>	<fctr>	<fctr>	<chr>	<chr>	<dbl>	<chr>	<chr>	<chr>	<dbl>
56	management	married	tertiary	no	779	yes	no	unknown	5
51	management	married	tertiary	no	10635	yes	no	unknown	5
57	technician	divorced	secondary	no	63	yes	no	unknown	5

6 rows | 1-10 of 18 columns

Build a Logistic Regression Model

Hide

```
install.packages("caret")
```

```
trying URL 'https://cran.rstudio.com/bin/macosx/big-sur-arm64/contrib/4.2/caret_6.0-94.t
gz'
Content type 'application/x-gzip' length 3578948 bytes (3.4 MB)
==================================================
downloaded 3.4 MB
```

```


The downloaded binary packages are in
    /var/folders/3k/gg_sdc9j771f9x853_g4f9wm0000gn/T//Rtmp987rJX/downloaded_packages
```

Hide

```
library(caret)
```

```
Loading required package: ggplot2
Loading required package: lattice

Attaching package: ‘caret’

The following object is masked from ‘package:kknn’:

    contr.dummy
```

Hide

```
library(class)
glm1 <- glm(subscribed ~ job + marital + age, data = trainData, family = binomial)
summary(glm1)
```

```

Call:
glm(formula = subscribed ~ job + marital + age, family = binomial, 
    data = trainData)

Deviance Residuals: 
    Min       1Q   Median       3Q      Max  
-0.9473  -0.5252  -0.4597  -0.3870   2.4086  

Coefficients: (2 not defined because of singularities)
                  Estimate Std. Error z value Pr(>|z|)    
(Intercept)      -2.143721   0.222100  -9.652  < 2e-16 ***
jobadmin.         0.081308   0.211100   0.385 0.700115    
jobblue-collar   -0.441382   0.210218  -2.100 0.035760 *  
jobentrepreneur  -0.322369   0.231801  -1.391 0.164311    
jobhousemaid     -0.194728   0.232230  -0.839 0.401741    
jobmanagement     0.191368   0.208314   0.919 0.358278    
jobretired        0.738191   0.214289   3.445 0.000571 ***
jobself-employed  0.023772   0.223466   0.106 0.915283    
jobservices      -0.253335   0.214634  -1.180 0.237875    
jobstudent        1.007652   0.224105   4.496 6.91e-06 ***
jobtechnician    -0.076600   0.209846  -0.365 0.715091    
jobunemployed     0.293950   0.223370   1.316 0.188181    
jobunknown              NA         NA      NA       NA    
maritaldivorced  -0.342205   0.060108  -5.693 1.25e-08 ***
maritalmarried   -0.470623   0.041368 -11.377  < 2e-16 ***
maritalsingle           NA         NA      NA       NA    
age               0.010081   0.001957   5.152 2.58e-07 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 26180  on 36167  degrees of freedom
Residual deviance: 25461  on 36153  degrees of freedom
AIC: 25491

Number of Fisher Scoring iterations: 5
```

Hide

```
probs <- predict(glm1, newdata=testData, type="response")
```

```
Warning: prediction from a rank-deficient fit may be misleading
```

Hide

```
pred <- ifelse(probs>0.5, 1, 0)
acc <- mean(pred==testData$subscribed)
print(paste("accuracy = ", acc))
```

```
[1] "accuracy =  0.885104500718788"
```

Analysts:

We got a fairly high accuracy which indicates that the logical regression model works fairly well if the person subscribed base on their age, job occupation, and martial status. The algorithm works by creating a mathematical model that finds the relation between input features to the binary outcome. During training, the algorithm changes the parameters of the model for better results and once the model is fully trained it’s compared to a decision threshold in order to make binary classification decision.

Build a kNN Model

Hide

```
install.packages("kknn")
```

```
trying URL 'https://cran.rstudio.com/bin/macosx/big-sur-arm64/contrib/4.2/kknn_1.3.1.tg
z'
Content type 'application/x-gzip' length 319886 bytes (312 KB)
==================================================
downloaded 312 KB
```

```


The downloaded binary packages are in
	/var/folders/3k/gg_sdc9j771f9x853_g4f9wm0000gn/T//Rtmp987rJX/downloaded_packages
```

Hide

```
library(kknn)



for (i in seq(1, 39, 2)) {
  # Train kNN model - consider 3 nearest neighbors to classify each test point
  model <- kknn(subscribed ~ age + job + marital, train=trainData, test = testData, k=i)
  # Make predictions using the test data set
  predictData <- data.frame(age = testData$age, job = testData$job, marital = testData$m
arital)
  # Make predictions using the test data set
  predictions <- predict(model, newdata = predictData)
  # Calculate the accuracy
  accuracy <- mean(predictions == testData$subscribed)
  # Print the accuracy
  cat("Accuracy:", accuracy, " K:", i, "\n")
}
```

```
Accuracy: 0.7937631  K: 1
Accuracy: 0.6282207  K: 3
Accuracy: 0.5069114  K: 5
Accuracy: 0.4254119  K: 7
Accuracy: 0.3379409  K: 9
Accuracy: 0.2760146  K: 11
Accuracy: 0.2251465  K: 13
Accuracy: 0.1746102  K: 15
Accuracy: 0.1534889  K: 17
Accuracy: 0.1339157  K: 19
Accuracy: 0.1016256  K: 21
Accuracy: 0.08625456  K: 23
Accuracy: 0.06446976  K: 25
Accuracy: 0.0551808  K: 27
Accuracy: 0.04710826  K: 29
Accuracy: 0.04456486  K: 31
Accuracy: 0.04191087  K: 33
Accuracy: 0.03638173  K: 35
Accuracy: 0.03140551  K: 37
Accuracy: 0.02665045  K: 39
```

Analysis

Looks like the algorithm performs best when we look for the closest data point in the training set, with an accuracy of 0.79, which is fairly high. The way the algorithm works is it calculates the value of the new data based on the closest data points in the data set. During training, each instance is represent by a vector of features that describe the properties of the instance. After that a new instance is presented to the algorithm and then the similarity score is computed for finding the distance between two instances. Choosing the k value for the model is crucial to avoid under-fitting or over-fitting of the model.

Decision Trees

Hide

```
# Load the rpart package
library(rpart)
# Train the decision tree model using the trainData dataset
model <- rpart(subscribed ~ age + job + marital, data = trainData, method = "class")
# Print the model summary
summary(model)
```

```
Call:
rpart(formula = subscribed ~ age + job + marital, data = trainData,
    method = "class")
  n= 36168

  CP nsplit rel error xerror xstd
1  0      0         1      0    0

Node number 1: 36168 observations
  predicted class=0  expected loss=0.1175072  P(node) =1
    class counts: 31918  4250
   probabilities: 0.882 0.118
```

Hide

```
# Make predictions on the testData dataset
predictions <- predict(model, newdata = testData, type = "class")
# Calculate the accuracy of the model
accuracy <- mean(predictions == testData$subscribed)
cat("Accuracy:", accuracy, "\n")
```

```
Accuracy: 0.8851045
```

Analysis

We got a fairly high accuracy with the decision trees model, which is roughly 0.89. The way the algorithm works, it selects the best features to split the data at each node. The entire process continues until it meets some criteria that was set for stooping the algorithm.

Conclusion

From what it looks like, the logistic regression and decision trees performed the best. The kNN model performed well but not as good as the other two. The major reason why Logistic regression and the trees model performed so well is because we are mostly working with categorical data (martial status, job occupation). Those two algorithms especially especially perform well with binary classification tasks. The kNN model can still work with categorical data but perhaps the reason why it didn’t work as well is because the data was not similar enough for it to make better predictions.