R_HW02

Odo Luo

March 15, 2021

Homework: Performance Assessment

Use both R and Python to answer the following questions:

1. Using the census data set, choose a few meaningful categorical features as predictors and Income as target.
2. Create train- and test data using a fixed split (use 1/3 for test set).
3. Fit a k-NN-model and a naive Bayes model. Tune k-NN using 10-times CV.
4. Predict the performances on the test set. Create the confusion matrices and compare the two classifiers in terms of Accuracy, Recall and Precision.
5. Create an ROC-curve for the naive Bayes model. Choose a good threshold, create new predictions using this threshold on the test set a

01 & 02 & 03

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.4     v dplyr   1.0.2
## v tidyr   1.1.2     v stringr 1.4.0
## v readr   1.4.0     v forcats 0.5.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
census <- read.csv("census.csv",header = TRUE) %>%
  as_tibble %>%
  mutate(id=row_number()) %>%
  mutate(income=as.numeric(ifelse(income==">50K",1,ifelse(income=="<=50K",0,NA)))) %>%
```

```r
  mutate_all(function(x) ifelse(x=="?",NA,x)) %>%
  drop_na()

head(census)
```

```
## # A tibble: 6 x 16
##     age workclass fnlwgt education education.num marital.status occupation
##   <int> <chr>      <int> <chr>             <int> <chr>          <chr>
## 1    39 State-gov  77516 Bachelors            13 Never-married  Adm-cleri~
## 2    50 Self-emp~  83311 Bachelors            13 Married-civ-s~ Exec-mana~
## 3    38 Private   215646 HS-grad               9 Divorced       Handlers-~
## 4    53 Private   234721 11th                  7 Married-civ-s~ Handlers-~
## 5    28 Private   338409 Bachelors            13 Married-civ-s~ Prof-spec~
## 6    37 Private   284582 Masters              14 Married-civ-s~ Exec-mana~
## # ... with 9 more variables: relationship <chr>, race <chr>, sex <chr>,
## #   capital.gain <int>, capital.loss <int>, hours.per.week <int>,
## #   native.country <chr>, income <dbl>, id <int>
```

NB

```r
library(sjmisc)
```

```
##
## Attaching package: 'sjmisc'

## The following object is masked from 'package:purrr':
##
##     is_empty

## The following object is masked from 'package:tidyr':
##
##     replace_na

## The following object is masked from 'package:tibble':
##
##     add_case
```

```r
nb <- census %>%
  select(workclass,education,marital.status,occupation,sex,id,income) %>%
  filter( workclass!="Without.pay" && education!="Preschool")
cols <- c("workclass","education","marital.status","occupation","sex","income")
nb[cols] <- lapply(nb[cols], factor)

train <- nb %>%  sample_frac(.70)

test <-  anti_join(nb, train,'id')
train <- train %>% select(-id)
test <- test %>% select(-id)

predictors<- test[1:(length(train)-1)]
target <- test[,ncol(train)]
predictors
```

```
## # A tibble: 9,049 x 5
##    workclass        education marital.status        occupation       sex
##    <fct>            <fct>     <fct>                 <fct>            <fct>
```

```
##  1 State-gov        Bachelors Never-married         Adm-clerical     Male
##  2 Self-emp-not-inc Bachelors Married-civ-spouse    Exec-managerial  Male
##  3 Private          Bachelors Married-civ-spouse    Prof-specialty   Female
##  4 Private          9th       Married-spouse-absent Other-service    Female
##  5 Self-emp-not-inc HS-grad   Married-civ-spouse    Exec-managerial  Male
##  6 Private          Masters   Never-married         Prof-specialty   Female
##  7 Private          Bachelors Married-civ-spouse    Exec-managerial  Male
##  8 Private          Bachelors Never-married         Adm-clerical     Female
##  9 Private          7th-8th   Married-civ-spouse    Transport-moving Male
## 10 Self-emp-not-inc Masters   Divorced              Exec-managerial  Female
## # ... with 9,039 more rows
```

```
model = train(income~ .,train,'naive_bayes',trControl=trainControl(method='cv',number=10))
model
```

```
## Naive Bayes
##
## 21113 samples
##     5 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 19002, 19001, 19002, 19001, 19001, 19002, ...
## Resampling results across tuning parameters:
##
##   usekernel  Accuracy   Kappa
##   FALSE      0.5185899  0.1927434
##    TRUE      0.7497277  0.0000000
##
## Tuning parameter 'laplace' was held constant at a value of 0
## Tuning
##  parameter 'adjust' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were laplace = 0, usekernel = TRUE
##  and adjust = 1.
```

```
result = predict(model,predictors)
head(result)
```

```
## [1] 0 0 0 0 0 0
## Levels: 0 1
```

KNN

```
knn_data <- census %>%
  select(age,education.num,hours.per.week,income,id)

knn_data$income <- as.factor(knn_data$income)
knn_train <- knn_data %>%  sample_frac(.70)

knn_test <-  anti_join(knn_data, knn_train,'id')
knn_train <- knn_train %>% select(-id)
knn_test <- knn_test %>% select(-id)
```

```r
knn_predictors<- knn_test[1:3]
knn_target <- knn_test[,ncol(knn_train)]
```

```r
knn_model = train(income~ ., knn_train,'knn',trControl=trainControl(method='cv',number=10))
knn_model
```

```
## k-Nearest Neighbors
##
## 21113 samples
##     3 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 19002, 19002, 19001, 19002, 19002, 19001, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.7795194  0.3444198
##   7  0.7814611  0.3448299
##   9  0.7848715  0.3525111
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```r
knn_result = predict(knn_model,knn_predictors)
head(knn_result)
```

```
## [1] 0 0 0 0 1 0
## Levels: 0 1
```