CS350 Networks of Software and Developers

Riccardo Romio

2022-12-15

```
## -- Attaching packages ---------------------------------------------------------------------------
## v ggplot2 3.4.0      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------------------------------------------------
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## Loading required package: NLP
##
##
## Attaching package: 'NLP'
##
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
## $newfile
## [1] "crandb.rda saved and loaded. 19096 packages listed between 2008-09-08 and 2023-01-21"
## [2] "0 removed, 0 new, 0 refreshed, 0 uploaded packages."
##
## $oldfile
## [1] "crandb.rda 19096 packages listed between 2008-09-08 and 2023-01-21"
##
## $removed_packages
## character(0)
##
## $new_packages
## character(0)
##
## $uploaded_packages
## character(0)
```

```
taskViews <-  available.views()

#random stuff past this point in the block
taskViewNames <- c()
for(i in seq(length(taskViews))){
  taskViewNames <- append(taskViewNames, unlist(taskViews[i])[1])
}
```

```
#Function for getting description files covering edge cases where there may not be any packages in a TV
getDesc <-  function(tv, coreCheck){
```

```
  temp <- taskViews[[tv]]
  temp2 <- temp %>%
    .$packagelist %>%
    filter(core == coreCheck) %>%
    .$name
  if(length(temp2) == 0){
    return(data.frame(taskView = c(), packages =  c(), description = c()))
  }
  descArr <- c()
  counter <- 0
  for(i in temp2){
    descTemp <- crandb %>%
      filter(Package == i) %>%
      .$Description
    if(length(descTemp) == 0){
      descTemp <- ""
    }
    descArr <- append(descArr, descTemp)
  }
  return(data.frame(taskView = tv, packages = temp2, description = descArr))
}
```

```
#Test that the function above works
MLTVDesc <- getDesc("MachineLearning", FALSE)
```

```
#gathering all desc files
allDescCore <- data.frame(taskView = c(), packages = c(), description = c())
for(i in taskViewNames){
  allDescCore <- rbind(allDescCore, getDesc(i, FALSE))
}
```

```
#Starting the data cleaning process and showing the difference of before and after
vec <- allDescCore$description
vec[[1]]
```

```
## [1] "The functions are designed to calculate the most widely-used county-level variables in agricultu
```

```
corpus <- Corpus(VectorSource(vec)) %>%
  tm_map(content_transformer(tolower)) %>%
  tm_map(removeNumbers) %>%
  tm_map(removePunctuation) %>%
  tm_map(removeWords, stopwords("english")) %>%
  tm_map(stemDocument) %>%
  tm_map(stripWhitespace)

as.character(corpus[[1]])
```

```
## [1] "function design calcul widelyus countylevel variabl agricultur product agriculturalclimat weath
```

```
#Creating the bag of words
dtm <- DocumentTermMatrix(corpus)
dtm <- removeSparseTerms(dtm, 0.99)
finalDataSet <- as.data.frame(as.matrix(dtm))
finalDataSet$taskView <- allDescCore$taskView #inserts class of each observation for the model (perhaps
```

```r
#Initiating cranly data gathering
p_db <- tools::CRAN_package_db()
package_db <- clean_CRAN_db()
package_network <- build_network(package_db)

dependenceTree <- compute_dependence_tree(package_network, package = "PlackettLuce") #problem is how to
```

```r
sample_size <- floor(0.9*nrow(finalDataSet))
set.seed(777)

# randomly split data in r
picked <- sample(seq_len(nrow(finalDataSet)),size = sample_size)
train <- finalDataSet[picked,]
test <- finalDataSet[-picked,]

finalDataSet$taskView <- relevel(factor(finalDataSet$taskView), ref = "Agriculture")
model <- multinom(taskView ~ ., data = train, MaxNWts = 22000)
```

```
## # weights:  21126 (20582 variable)
## initial  value 15253.429712
## iter  10 value 7331.962805
## iter  20 value 5500.675559
## iter  30 value 3465.723921
## iter  40 value 2563.353387
## iter  50 value 2063.831500
## iter  60 value 1682.496333
## iter  70 value 1303.581899
## iter  80 value 1096.324547
## iter  90 value 1002.833163
## iter 100 value 950.907495
## final  value 950.907495
## stopped after 100 iterations
```

```r
modelClassificationTest <- predict(model, newdata = test, "probs")
classifiedList <- colnames(modelClassificationTest)[apply(modelClassificationTest,1,which.max)]
finalClassifiedDf <- data.frame(test$taskView, classifiedList)
counter <- 0
for(i in seq(length(finalClassifiedDf$test.taskView))){

  if(finalClassifiedDf$test.taskView[i] == finalClassifiedDf$classifiedList[i]){
    counter <- counter + 1
  }
}
counter/length(finalClassifiedDf$test.taskView)
```

```
## [1] 0.2687225
```