Master Analysis

Marcin Kosinski

25.10.2015

```
extractSurvival <- function(cohorts){

  survivalData <- list()
  for(i in cohorts){
    get(paste0(i, ".clinical"), envir = .GlobalEnv) %>%
                select(patient.bcr_patient_barcode,
                            patient.vital_status,
                            patient.days_to_last_followup,
                            patient.days_to_death ) %>%
                mutate(bcr_patient_barcode = toupper(patient.bcr_patient_barcode),
                       patient.vital_status = ifelse(patient.vital_status %>%
                                                       as.character() =="dead",1,0),
                   barcode = patient.bcr_patient_barcode %>%
                                 as.character(),
                    times = ifelse( !is.na(patient.days_to_last_followup),
                         patient.days_to_last_followup %>%
                           as.character() %>%
                           as.numeric(),
                  patient.days_to_death %>%
                           as.character() %>%
                           as.numeric() )
                        ) %>%
   filter(!is.na(times)) -> survivalData[[i]]


  }
  do.call(rbind,survivalData) %>%
    select(bcr_patient_barcode, patient.vital_status, times) %>%
    unique

}


extractMutations <- function(cohorts, prc){
  mutationsData <- list()
  for(i in cohorts){
    get(paste0(i, ".mutations"), envir = .GlobalEnv) %>%
      select(Hugo_Symbol, bcr_patient_barcode) %>%
      filter(nchar(bcr_patient_barcode)==15) %>%
      filter(substr(bcr_patient_barcode, 14, 15)=="01") %>%
      unique -> mutationsData[[i]]
  }
  do.call(rbind,mutationsData) %>% unique -> mutationsData

  mutationsData %>%
    group_by(Hugo_Symbol) %>%
    summarise(count = n()) %>%
```

```
    arrange(desc(count)) %>%
    mutate(count_prc = count/length(unique(mutationsData$bcr_patient_barcode))) %>%
    filter_(paste0("count_prc > ",prc)) %>%
    select(Hugo_Symbol) %>%
    unlist -> topGenes

  mutationsData %>%
    filter(Hugo_Symbol %in% topGenes) -> mutationsData_top


  mutationsData_top %>%
    dplyr::group_by(bcr_patient_barcode) %>%
    dplyr::summarise(count = n()) %>%
    group_by(count) %>%
    summarise(total = n()) %>%
    arrange(desc(count))
#
#   mutationsData_top %>%
#     spread(Hugo_Symbol, bcr_patient_barcode) -> mutationsData_top_sp

  as.data.table(mutationsData_top) -> mutationsData_top_DT
  dcast.data.table(mutationsData_top_DT, bcr_patient_barcode ~ Hugo_Symbol , fill = 0) %>%
    as.data.frame -> mutationsData_top_dcasted

  mutationsData_top_dcasted[,-1][mutationsData_top_dcasted[,-1] != "0"] <- 1

  mutationsData_top_dcasted -> result
  names(result) <- gsub(names(result),pattern = "-",  replacement = "")
  result
}


extractCohortIntersection <- function(){

  data(package = "RTCGA.mutations")$results[,3] %>%
    gsub(".mutations", "", x = .) -> mutations_data
  data(package = "RTCGA.clinical")$results[,3] %>%
    gsub(".clinical", "", x = .) -> clinical_data

  intersect(mutations_data, clinical_data)
}

prepareCoxDataSplit <- function(mutationsData, survivalData, groups, seed = 4561){
  mutationsData %>%
  mutate(bcr_patient_barcode = substr(bcr_patient_barcode,1,12)) %>%
  left_join(survivalData,
            by = "bcr_patient_barcode") -> coxData

  coxData <- coxData[, -c(1,2)]

  coxData %>%
    filter(times > 0) %>%
    filter(!is.na(times)) -> coxData
```

```
  apply(coxData[,-c(1092, 1093)], MARGIN = 2,function(x){
    as.numeric(as.character(x))
  }) -> coxData[,-c(1092, 1093)]

  set.seed(seed)
  sample(groups, replace = TRUE, size = 6085) -> groups
  split(coxData, groups) #coxData_split
}

prepareForumlaSGD <- function(coxData){
  as.formula(paste("Surv(times, patient.vital_status) ~ ",
                   paste(names(coxData[[1]])[-c(1092, 1093)],
                         collapse="+"), collapse = ""))
}


full_cox_loglik_matrix <- function(beta, x, censored){
  order(x$times) -> order2
  x[order2, ] -> xORD
  censored[order2] -> censORD
  sum(censORD*(beta%*%x[, -which(names(x)=='times')] -
                       log(cumsum(exp(beta1*rev(x1) + beta2*rev(x2))))))
}



library(dplyr)
```

```
Warning: package 'dplyr' was built under R version 3.2.3
```

```

Attaching package: 'dplyr'

The following objects are masked from 'package:stats':

    filter, lag

The following objects are masked from 'package:base':

    intersect, setdiff, setequal, union
```

```
library(RTCGA.clinical)
```

```
Loading required package: RTCGA
Loading required package: knitr
Welcome to the RTCGA (version: 1.1.11).
```

```
library(RTCGA.mutations)
library(data.table)
```

```
Attaching package: 'data.table'

The following objects are masked from 'package:dplyr':

    between, last
```

```
library(coxphSGD)
```

```
Loading required package: survival
```

```
library(archivist)
```

```
Welcome to archivist (version: 1.9.3.1).
```

```
#setLocalRepo(getwd())
# createEmptyLocalRepo(getwd(), default = TRUE)
# alink('e8f55d0bc8c17d6c4c663f871866c0ec', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(survivalData)
# alink('8aa74b9f156f087944defb48347e0d3e', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(mutationsData)
# alink('4644873a37d69da344d5db8647389415', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(coxData_split)
# alink('aa0d32d2f32d39197e65ff632fdd600e', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(formulaSGD)
# alink('0642f7e1c2a1fbea36d7d0ac518b9c8d', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(testCox) # 3eebc99bd231b16a3ea4dbeec9ab5edb
# alink('3eebc99bd231b16a3ea4dbeec9ab5edb', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(trainCox) # 1a06bef4a60a237bb65ca3e2f3f23515
# alink('1a06bef4a60a237bb65ca3e2f3f23515', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(model_1_over_t) # "446ac4dcb7d65bf39057bb341b296f1a"
# alink('446ac4dcb7d65bf39057bb341b296f1a', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(model_1_over_50sqrt_t) #"044ad9f336ac4626ee779f8468dc6a4a"
# alink('044ad9f336ac4626ee779f8468dc6a4a', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
# saveToRepo(model_1_over_100sqrt_t) # "47de266ea701af9f81d90b9e204250f2"
# alink('47de266ea701af9f81d90b9e204250f2', repo = 'coxphSGD', user = 'MarcinKosinski', format = 'latex
```

Do analizy badającej wpływ występowania mutacji genów na czas przeżycia wykorzystano dane kliniczne i dane o występujących u pacjentów mutacjach genetycznych. Starano się wykorzystać dane ze wszystkich 38 dostępnych kohort nowotworowych z badania *The Cancer Genome Atlas* (TCGA), jednak nie dla wszystkich kohort umieszczono w badaniu dane o mutacjach. Częśc wspólną nazw dla kohort zawierających zarówno dane kliniczne oraz dane o mutacjach wygenerowaną dzięki wywołaniu

```
(extractCohortIntersection() -> cohorts)
```

```
 [1] "ACC"      "BLCA"     "BRCA"     "CESC"     "CHOL"     "COAD"
 [7] "COADREAD" "DLBC"     "ESCA"     "GBM"      "GBMLGG"   "HNSC"
[13] "KICH"     "KIPAN"    "KIRC"     "KIRP"     "LAML"     "LGG"
[19] "LIHC"     "LUAD"     "LUSC"     "OV"       "PAAD"     "PCPG"
[25] "PRAD"     "READ"     "SARC"     "SKCM"     "STAD"     "STES"
[31] "TGCT"     "THCA"     "UCEC"     "UCS"      "UVM"
```

archivist::aread('MarcinKosinski/coxphSGD/e8f55d0bc8c17d6c4c663f871866c0ec')

Następnie dla tak otrzymanych 35 kohort nowotworowych uzyskano dane o statusie pacjenta (śmierć bądź cenzurowanie) oraz jego czasie spędzonym pod obseracją dzięki funkcji

```
head(extractSurvival(cohorts) -> survivalData)
```

```
      bcr_patient_barcode patient.vital_status times
ACC.1        TCGA-OR-A5J1                    1  1355
ACC.2        TCGA-OR-A5J2                    1  1677
ACC.3        TCGA-OR-A5J3                    0  1942
ACC.4        TCGA-OR-A5J4                    1   423
ACC.5        TCGA-OR-A5J5                    1   365
ACC.6        TCGA-OR-A5J6                    0  2428
```

archivist::aread('MarcinKosinski/coxphSGD/8aa74b9f156f087944defb48347e0d3e')

Dane o mutacjach występujących wśród tkanek nowotworowych kolejnych pacjentów uzyskano za pomocą

```
extractMutations(cohorts, 0.02) -> mutationsData
```

```
Using 'bcr_patient_barcode' as value column. Use 'value.var' to override
```

```
mutationsData[1:8, c(1,4,56,100,207,801)]
```

```
  bcr_patient_barcode A2ML1 ALMS1 ATP2B2 CNTNAP4 PLEC
1     TCGA-02-0003-01     0     1      0       0    0
2     TCGA-02-0033-01     0     0      0       0    0
3     TCGA-02-0047-01     0     0      0       0    0
4     TCGA-02-0055-01     0     0      0       0    0
5     TCGA-02-2470-01     0     0      0       0    0
6     TCGA-02-2483-01     0     0      1       0    0
7     TCGA-02-2485-01     0     0      0       0    0
8     TCGA-02-2486-01     0     0      0       0    0
```

archivist::aread('MarcinKosinski/coxphSGD/4644873a37d69da344d5db8647389415')

gdzie wybrano jedynie te geny, których mutacja dotyczyła co najmniej 2 % pacjentów mających zarówno dane kliniczne jak i dane o występujących mutacjach w genach.

Dla tak otrzymanych dwóch zbiorów danych połączono dla pacjentów informacje kliniczne z informacjami o mutacjach dzięki przypisanym do pacjentów i ich próbek kodów `bcr_patient_barcode`, by ostatecznie podzielić zbiór pacjentów na 100 losowo utworzonych grup.

```
set.seed(4561)
prepareCoxDataSplit(mutationsData,survivalData, groups = 100) -> coxData_split
head(coxData_split[[1]][c(1,10), c(210,302,356,898,911,1092:1093)])
```

```
     COL14A1 DOCK9 FASN SEMA5A SHPRH patient.vital_status times
81         0     0    0      0     0                    1     7
1068       1     0    0      1     0                    1  1171
```

archivist::aread('MarcinKosinski/coxphSGD/aa0d32d2f32d39197e65ff632fdd600e')

Niezbędną formułę modelu potrzebną do sprezycowania, które geny (a pozostało ich 1091) należy uwzględnić w modelu uzyskano dzięki pomocniczej funkcji

```
prepareForumlaSGD(coxData_split) -> formulaSGD
```

archivist::aread('MarcinKosinski/coxphSGD/064277e1c2a1fbea36d7d0ac518b9c8d')

Ostatecznie dla 6085 pacjentów, którzy posiadali informacje o występujących mutacjach, oraz dla których odnotowano komplet i poprawność danych klinicznych dotyczących statusu i obserwowanego czasu przeżycia wyliczono współczynniki modelu proporcjonalnych hazardów Coxa z wykorzystaniem stochastycznego spadku gradientu do estymacji. Model dopasowano wielokrotnie z różnymi ciągami odpowiadającymi za długość kroku algorytmu, dodatkowo badano różną ilość epok w algorytmie. Dla tak powstałych kilku modeli wybrano ten, który dla swoich współczynników dawał największą wartość funkcji częściowej log-wiarogodności dla niewykorzystanej do uczenia próbki, zawierającej 2 ostatnie zaobserwowane podzbiory obserwacji. Dla każdego z ciągów $1/t, 1/50 * \sqrt(t), 100/5 * \sqrt(100)$ odpowiadających długościom kroków w algorytmie wyznaczono współczynniki modelu dla 5 epok, dzięki czemu możliwe było rozważanie postępu danego wariantu algorytmu również po 1, 2, 3 czy 4 epokach.

```
coxData_split[99:100] -> testCox
coxData_split[1:98] -> trainCox
coxphSGD(formulaSGD, data = trainCox, max.iter = 490) -> model_1_over_t
coxphSGD(formulaSGD, data = trainCox, max.iter = 490,
         learningRates = function(t){1/(50*sqrt(t))}) -> model_1_over_50sqrt_t
coxphSGD(formulaSGD, data = trainCox, max.iter = 490,
         learningRates = function(t){1/(100*sqrt(t))}) -> model_1_over_100sqrt_t
```

Niemożliwe było sprawdzenie założeń modelu dotyczących proporcjonalności hazardu, gdyż zakładano napływającą postać danych (stąd podział danych na 100 grup). Dla takiej postaci pojawiania się danych ciężko także mówić o jakiejkolwiek diagnostyce poprawności dopasowania modelu i dokładności otrzymanych wpsółczynników. Nie stworzono teorii pozwalającej badać istotność statystyczną otrzymanych współczynników w modelu, jednak założono, że współczynniki dostatecznie odległe od 0 można uznać za istotnie wpływające na czas życia pacjenta. Współczynniki dodatnie oznaczają zwiększenie hazardu pacjenta posiadającego mutację w danym genie w stosunku do pacjentów nie posiadających mutacji w danym genie. Współczynniki ujemne oznaczają zmniejszenie hazardu pacjenta posiadającego mutację w danym genie w stosunku do pacjentów nie posiadających mutacji w danym genie. Wzrost proporcji hazardu można otrzymać dla danego genu poprzez obłożenie współczynnika funkcją wykładniczą o wykładniku e.

Wyniki estymacji dla genów zawierających największe co do modułu współczynniki można znaleźć w Tabeli 1.