Final project

Natalia Sarabia Vásquez

November 29, 2021

1 Project

Generate chromosomes and set the population matrix

2 Creating the generation

```r
# Number of genes per chromosome: number of  covariates in a linear model
chromosome_length <- 6

# Population size
# The paper recommends fo binary encoding of chromosomes to choose P to satisfy $C \leq P \leq 2C$
population_size <- sample(chromosome_length:(2*chromosome_length), 1, replace=TRUE)

# Generate the genes and set the chromosomes:
create_population <- function(chromosome_length, population_size){
  n <- chromosome_length * population_size
  chromosome <- as.vector(sample(0:1, n, replace=TRUE))
  population <- as.data.frame(matrix(chromosome, nrow = population_size, ncol = chromosome_length))
  names(population) <- dput(paste0('gen_', seq(1,chromosome_length,1)))
  return(population)
}
```

```r
# Function to identify the genes that would be active in each model given a chromosome:
find_genes <- function(chromosome, variables_names){
  variables <- variables_names[grep(1,as.vector(chromosome))]
  return(variables)
}

# Function to construct the formulas given the active genes of a chromosome:
set_formulas <- function(active_genes, name_y){
  formulas <- as.formula(paste(name_y, paste(active_genes, sep = "", collapse = " + "), sep = " ~ "))
  return(formulas)
}

# Function to compute the fitness (AIC) given a formula and a dataset
# By default, it will fit a linear regression. Although, it can receive the parameters
# for a generalized linear model
fitness <- function(formula, data, ...){
   fitness <- AIC(glm(formula = formula, data = data, ...))
   return(fitness)
}
```

```r
# Compute the fitness of an entire generation:
# The result is sort by the fittest individual to the least fit
get_fitness <- function(X, name_y, generation){
  data_names <- names(X)[!names(X) %in% c(name_y)]
  variables <- apply(generation, 1, find_genes, data_names)
  formulas <- lapply(variables, set_formulas, name_y)
  fitness <- lapply(formulas, fitness, X)
  return(fitness)
}
```

Data to try out the code:

```r
my_generation <- create_population(chromosome_length, population_size)
```

```
## c("gen_1", "gen_2", "gen_3", "gen_4", "gen_5", "gen_6")
```

```r
head(my_generation)
```

```
##   gen_1 gen_2 gen_3 gen_4 gen_5 gen_6
## 1     0     0     1     1     0     0
## 2     0     1     0     0     1     0
## 3     0     1     0     0     0     1
## 4     0     0     1     1     1     0
## 5     1     1     0     0     0     0
## 6     0     0     0     0     1     1
```

```r
x <- as.data.frame(matrix(runif(1000,0,1),ncol=10,nrow=10))
names(x) <- letters[1:10]
head(x)
```

```
##           a         b          c          d         e          f          g
## 1 0.5575332 0.7957879 0.76833085 0.36810277 0.5536584 0.06439736 0.23737654
## 2 0.8877217 0.8886941 0.03841423 0.40230083 0.6850230 0.74911808 0.06369718
## 3 0.2794800 0.8331907 0.55118539 0.09989985 0.8286374 0.60226176 0.50655105
## 4 0.1823561 0.5934497 0.18381775 0.90482978 0.8908899 0.02161180 0.47757060
## 5 0.8252538 0.8227338 0.89580447 0.84203561 0.4335668 0.31681218 0.28863843
## 6 0.9448330 0.2296565 0.27517049 0.39019539 0.1851195 0.96425349 0.89116455
##           h          i          j
## 1 0.8695829 0.92932274 0.05207503
## 2 0.1631404 0.36128458 0.95335247
## 3 0.4434900 0.33620484 0.67836215
## 4 0.3791477 0.08718635 0.42049384
## 5 0.9803536 0.38396535 0.64784387
## 6 0.9112647 0.36777312 0.71786743
```

```r
fitness_scores <- unlist(get_fitness(x,"a",my_generation))
head(fitness_scores)
```

```
## [1] 10.02848 12.42595 12.50510 11.96870 11.81733 13.43264
```

```r
my_generation <- cbind(my_generation,fitness_scores) %>%
  arrange(desc(fitness_scores))
head(my_generation)
```

```
##   gen_1 gen_2 gen_3 gen_4 gen_5 gen_6 fitness_scores
## 1     0     1     0     1     1     1       16.22440
## 2     0     0     0     0     1     1       13.43264
## 3     0     0     1     0     1     1       12.89481
## 4     0     1     0     0     0     1       12.50510
## 5     0     1     0     0     1     0       12.42595
## 6     0     0     1     1     1     0       11.96870
```