DATA605 - Assignment 1

Christian Thieme

8/28/2020

Problem Set 1:

You can think of vectors representing many dimensions of related information. For instance, Netflix might store all the ratings a user gives to movies in a vector. This is clearly a vector of very large dimensions (in the millions) and very sparse as the user might have rated only a few movies. Similarly, Amazon might store the items purchased by a user in a vector, with each slot or dimension representing a unique product and the value of the slot, the number of such items the user bought. One task that is frequently done in these settings is to find similarities between users. And, we can use dot-product between vectors to do just that. As you know, the dot-product is proportional to the length of two vectors and to the angle between them. In fact, the dot-product between two vectors, normalized by their lengths is called as the cosine distance and is frequently used in recommendation engines.

1. Calculate the dot product u:v where u = [0.5; 0.5] and v = [3;-4]:

$$u = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \qquad v = \begin{bmatrix} 3 & -4 \end{bmatrix}$$

$$u \cdot v = \begin{bmatrix} 0.5, 0.5 \end{bmatrix} \begin{bmatrix} 3 \\ -4 \end{bmatrix}$$

$$u \cdot v = (0.5)(3) + (0.5)(-4)$$

$$u \cdot v = 1.5 - 2$$

$$u \cdot v = -0.5$$

2. What are the lengths of u and v? Please note that the mathematical notion of the length of a vector is not the same as a computer science definition.

2. Length of u:

$$\|\vec{u}\|^2 = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$

$$\|\vec{u}\|^2 = (0.5)(0.5) + (0.5)(0.5)$$

$$\|\vec{u}\|^2 = 0.25 + 0.25$$

$$\sqrt{\|\vec{u}\|^2} = \sqrt{0.5}$$

$$\|\vec{u}\| = \sqrt{0.5} \text{ or } 0.7071$$

Length of v:

$$\|\vec{v}\|^2 = \begin{bmatrix} 3 & -4 \end{bmatrix} \begin{bmatrix} 3 \\ -4 \end{bmatrix}$$

$$\|\vec{v}\|^2 = (3)(3) + (-4)(-4)$$

$$\|\vec{v}\|^2 = 9 + 16$$

$$\sqrt{\|\vec{v}\|^2} = \sqrt{25}$$

$$\|\vec{v}\| = 5$$

3. What is the linear combination: 3u - 2v?

$$3u - 2v$$

$$3\begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} - 2\begin{bmatrix} 3 \\ -4 \end{bmatrix} =$$

$$\begin{bmatrix} 1.5 \\ 1.5 \end{bmatrix} - \begin{bmatrix} 6 \\ -8 \end{bmatrix} = \begin{bmatrix} -4.5 \\ 9.5 \end{bmatrix}$$

4. What is the angle between u and v?

$$\vec{u} \cdot \vec{v} = \|\vec{u}\| \|\vec{v}\| \cos\theta$$

using the dot product calculated in problem 1 = -0.5 and the lengths of u and v calculated in problem 2, $\sqrt{0.5}$ and 5 respectively we get:

$$-0.5 = \sqrt{0.5}(5)\cos\theta$$

$$\frac{-0.5}{5\sqrt{0.5}} = \cos\theta$$

take inverse cosine of each side to solve for θ:

$$\arccos\left(\frac{-0.5}{5\sqrt{0.5}}\right) = \arccos(\cos\theta)$$

$$\theta = \arccos\left(\frac{-0.5}{5\sqrt{0.5}}\right)$$

$$\theta = 98.13°$$

Problem Set 2:

Set up a system of equations with 3 variables and 3 constraints and solve for x. Please write a function in R that will take two variables (matrix A & constraint vector b) and solve using elimination. Your function should produce the right answer for the system of equations for any 3-variable, 3-equation system. You don't have to worry about degenerate cases and can safely assume that the function will only be tested with a system of equations that has a solution. Please note that you do have to worry about zero pivots, though. Please note that you should not use the built-in function solve to solve this system or use matrix inverses. The approach that you should employ is to construct an Upper Triangular Matrix and then back-substitute to get the solution. Alternatively, you can augment the matrix A with vector b and jointly apply the Gauss Jordan elimination procedure.

```
library(matlib)
# The below function takes in a matrix (i.e. matrix(c(1,2,-1,1,-1,-2,3,5,4), 3,3)), and a constraint vector (i.e. c(1,2,6))
# and solves the system of equations by constructing an upper triangle matrix and then using back-subsitution to
  get the solutions.
# The function also identifies "zero pivots"

# This function solves a system of equations without using "solve" or the reduced row echelon form functions.

three_by_three_solver <- function(mat, vect) {

  no_constraint_matrix <- mat

  #Creating full matrix out of matrix and constraint vector
  full_matrix <- cbind(no_constraint_matrix, vect)
  print("The matrix is: ")
  print(full_matrix)
   print("--------------------------")

  #reducing matrix to echelon form which is an upper triangle matrix (NOT REDUCED ROW ECHELON FORM!)
  upper_triangle <- echelon(full_matrix, reduced = FALSE)

  print("The upper triangle matrix is: ")
  print(upper_triangle)
  print("--------------------------")

  #checks to see if there are zero pivots, if so, it will print "infinite solutions"
  zero_pivots <- upper_triangle[3,1] + upper_triangle[3,2] + upper_triangle[3,3]

  if(zero_pivots == 0) {

    print("There are infinitely many solutions to the system of equations")

  } else {

     #solving for x, y, and z with back substitution

     z <- upper_triangle[3,4]
     y <- upper_triangle[2,4] - upper_triangle[2,3] * z
     x <- upper_triangle[1,4] - (upper_triangle[1,2] * y) - (upper_triangle[1,3] * z)

     print(paste0("The rounded solutions to the system of equations are: ", round(x,2),", ",round(y,2),", ", rou
nd(z,2)))
  }
}
```

Now let's test our function with the test system provided:

$$\begin{bmatrix} 1 & 1 & 3 \\ 2 & -1 & 5 \\ -1 & -2 & 4 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 6 \end{bmatrix}$$

```
x <- matrix(c(1,2,-1,1,-1,-2,3,5,4), 3,3)
y <- c(1,2,6)

three_by_three_solver(x, y)
```

```
## [1] "The matrix is: "
##                vect
## [1,]  1  1 3    1
## [2,]  2 -1 5    2
## [3,] -1 -2 4    6
## [1] "--------------------------"
## [1] "The upper triangle matrix is: "
##                      vect
## [1,] 1 -0.5  2.5  1.0000000
## [2,] 0  1.0 -2.6 -2.8000000
## [3,] 0  0.0  1.0  0.9545455
## [1] "--------------------------"
## [1] "The rounded solutions to the system of equations are: -1.55, -0.32, 0.95"
```

The function has correctly calculated the rounded answers. Now let's test the function on a system that will have a zero pivot column (infinitely many solutions):

$$\left[\begin{array}{ccc|c} 3 & -3 & 4 & -23 \\ 1 & 2 & -3 & 25 \\ 4 & -1 & 1 & 25 \end{array}\right]$$

```
x <- matrix(c(3,1,4,-3,2,-1,4,-3,1), 3,3)
y <- c(-23,25,25)

three_by_three_solver(x, y)
```

```
## [1] "The matrix is: "
##                vect
## [1,] 3 -3  4  -23
## [2,] 1  2 -3   25
## [3,] 4 -1  1   25
## [1] "--------------------------"
## [1] "The upper triangle matrix is: "
##                        vect
## [1,] 1 -0.25  0.250000 6.250000
## [2,] 0  1.00 -1.444444 8.333333
## [3,] 0  0.00  0.000000 1.000000
## [1] "--------------------------"
## [1] "There are infinitely many solutions to the system of equations"
```

Again the function has correctly identified that the above system of equations will have infinitely many solutions.