SCOTT JACKSON, MICHAEL HANKIN, ZACH KATTAWAR, ERYN LI

SCALAR

OVERVIEW

▸ Objective / Motivations

▸ What is ScalaR

▸ Design / Implementation

 ▸ Typing

 ▸ RVectors

 ▸ DataFrames

▸ Opportunities for Expansion

OBJECTIVE AND MOTIVATION

▸ Create a convenient language for statistical analysis in Scala

▸ Lack a native structure for working with large datasets

▸ Offer an easy way for users to visualize their data

WHAT IS SCALAR

‣ A subset of the R programming language that uses a similar syntax

  ‣ All variable assignment is done with the "<--" operator

  ‣ c(...) can be used to create new vectors

‣ Uses DataFrame and vector objects as wrappers for all variables

  ‣ Allows reading in of datasets from external files

  ‣ Easy to query and perform operations on

‣ Gives the user a way to visualize their data in various ways

  ‣ Scatter Plots

  ‣ Histograms

TYPING

- Present four types for representing data

- Numeric

  - Represents Numbers

  - For simplicity, everything is a Double

- Logical

  - Booleans

- Character

  - "Hello World"

- NA

  - Absence of type

- Important concept for statistics

```scala
abstract class Type {
  type T
  def getType: String
  def storedValue: T
  override def toString: String = storedValue.toString
}

class Logical(value: Boolean) extends Type {
  type T = Boolean
  def storedValue: Boolean = value
  def getType: String = "Logical"
}

class Numeric(value: Double) extends Type {
  type T = Double
  def storedValue: Double = value
  def getType: String = "Numeric"

  def ==(that: Numeric): Boolean = {
    return this.storedValue == that.storedValue
  }
}

class Character(value: String) extends Type {
  type T = String
  def storedValue: String = value
  def getType: String = "Character"
}

class NAType extends Type {
  type T = String
  var curType: String = "Logical"
  def storedValue: String = "NA"
  def getType: String = return curType
  def setType(t: String) = t match {
    case "Logical" => curType = "Logical"
    case "Numeric" => curType = "Numeric"
    case "Character" => curType = "Character"
  }
  override def toString: String = "NA"
}
```

VECTORS

- A vector is a one dimensional collection of values all of which are the same type

- Type coercion among elements within a single vector

    - Logical -> Numeric -> Character

- Supports operations to be performed on the data

    - +

    - –

    - Sum

    - Mean

    - Standard Deviation

- Missing data within a vector will take on the type NA

DATAFRAMES

▸ Column based storage of vectors with a specified schema it follows

  ▸ `[column1, column2, column3]` , `[String, Numeric, Numeric]`

▸ Simple queries may be performed on the Dataframe

  ▸ `"Column1 > 10"`

▸ Allows for a few types of graphical representation (WISP)

FURTHER GOALS

▸ Add user-defined functions

▸ Implement ability to execute more complex queries ▸

THANK YOU!

scalar