Values Affirmation Database README

05 October, 2015

This document describes the database compiled as part of the NSF and ROADS sponsored research for the investigation of the values affirmation essays using natural language processing methodology.

Contents of the database

The database consists of four files. The name of the file indicates the content of the file, plus the date it was last edited:

1. **demog9.11.15.csv** - This file features the demographic characteristics of the participants in the studies. As indicated by the number string in the filename, it was last updated on 9/11/2015. Similar dates are appended to the other files.
2. **essays9.30.15.csv** - This file includes the essays and their corresponding attributes.
3. **grades9.11.15.csv** - This file includes grades and other academic outcome measures for participants in the studies
4. **prompts10.5.15.csv** - This file includes characteristics of the prompts for each of the possible interventions.

File format and structure

In order to make these files usable across a variety of software types and system environments, they have been stored as basic csv files. However, using commas to separate the field values in the essays and prompts file will obviously lead to problems, because the content of the fields occasionally contain columns. For that reason, these files are stored as pipe-separated values (a pipe is a |). It's possible to see the difference in structure by opening the files using a basic text-editor (textEdit on mac, notepad, sublimetext, textwrangler, etc)

The format of each file follows the general principles of relational databases. Each file is stored in what is often referred to as a 'long' format, and each file features a specific type of observational unit (i.e. essays, participants, outcomes, or prompts). Within each file, these observational units are organized by the variables, which form columns, and the observations, which form rows.

The variables for each file are listed below.

```
library(printr)
df.essays <- read.csv('../Data/3 CSV Files/essays9.30.15.csv', sep='|')
names(df.essays)
```

Essays

```
## [1] "ID"                  "Study"               "Intervention_number"
## [4] "Essay"               "Condition"           "Intervention_Date"
## [7] "Cohort"
```

```
df.demographics <- read.csv('../Data/3 CSV Files/demog9.11.15.csv')
names(df.demographics)[-6:-8]
```

Participants

```
## [1] "ID"        "Study"     "Cohort"    "Ethnicity" "Gender"
```

```
df.outcomes <- read.csv('../Data/3 CSV Files/grades9.11.15.csv')
names(df.outcomes)[-1]
```

Outcomes

```
## [1] "ID"                "Study"             "Grade"
## [4] "Grade_type"        "est_Grade_date"    "intervention_year"
```

```
library(xlsx)
```

Prompts

```
## Loading required package: rJava
## Loading required package: xlsxjars
```

```
df.prompts <- read.xlsx('../Data/3 CSV Files/prompts10.5.15.xls', sheetIndex = 1)
names(df.prompts)[-1]
```

```
## [1] "Intervention_number" "Condition"           "Intervention_date"
## [4] "Study"               "Cohort"              "Prompt"
## [7] "Values"
```

Note that this file structure makes it very clear what information is contained where. Very little explication of the content of each file is needed here.

All files with the exception of the prompts share a common *key* of the participant ID number. This allows the analyst to join together data from each file using basic join operations. Where studies had duplicate ID numbers, we appended a `.x` to the end of the ID, where x is an integer, incremented each time we encountered a file that had ID numbers that duplicated ID numbers already incorporated into the database.

The prompts file, since the unit of measure is the intervention and not the individual, does not have a specific participant ID associated with each row. Instead, the information from this file can be joined to the others using one of the multiple keys it has in common with the other files. For instance, if the analyst were interested in joining the prompts to the Essays table, one could join by the combination of `Intervention_number`, `Intervention_date`, and `Condition`, which would correctly join

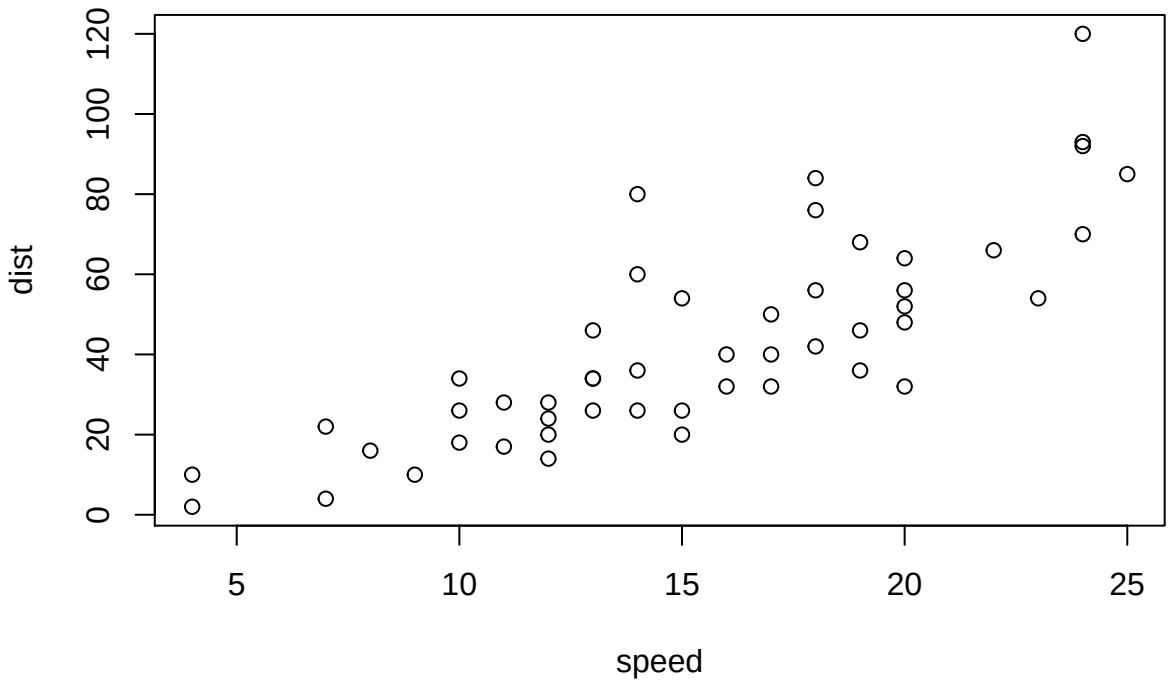
Furthermore, these data are stored in what is sometimes referred to as a 'long' format. This means that there are a minimal number of columns, and each table features measurements on a single element (i.e. essays, participants, outcomes, or prompts). The rows are the individual observations

File contents

I now illustrate the content of each file. For each file, I describe each variable in the file,

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.