**R for Data Science**

R is a language and environment for statistical computing and graphics.

R provides a wide variety of statistical (linear and nonlinear modelling, classical statistical tests, time-series analysis, classification, clustering, …) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity.

One of R’s strengths is the ease with which well-designed publication-quality plots can be produced, including mathematical symbols and formulae where needed.

R is available as Free Software under the terms of the Free Software Foundation’s GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.

The R environment

R is an integrated suite of software facilities for data manipulation, calculation and graphical display. It includes

* an effective data handling and storage facility,
* a suite of operators for calculations on arrays, in particular matrices,
* graphical facilities for data analysis and display either on-screen or on hardcopy, and
* For computationally-intensive tasks, C, C++ and Fortran code can be linked and called at run time. Advanced users can write C code to manipulate R objects directly.
* R has its own LaTeX-like documentation format, which is used to supply comprehensive documentation, both on-line in a number of formats and in hardcopy.

**Creating a Local R Repository**

**Creating a Repository for R libraries like a copy of CRAN online repository for R in an offline environment is possible through miniCRAN.**

**miniCRAN**

A package has dependencies which further have dependencies of their own.

miniCRAN makes it possible to create an internally consistent repository consisting of selected packages from CRAN-like repositories. The user specifies a set of desired packages, and 'miniCRAN' recursively reads the dependency tree for these packages, then downloads only this subset. The user can then install packages from this repository directly, rather than from CRAN. This is useful in production settings, e.g. server behind a firewall, or remote locations with slow (or zero) Internet access.

In this way, we can achieve a CRAN like R repository with only the packages we actually require

**Implementation:**

1. Create a VM (Ubuntu 18.04) in Vcenter
2. Install base -r (Programming Language)
3. Install Rstudio (GUI for R)

We now have a system with base R installed.

1. Install miniCRAN R package from the internet (Only if you are interested in setting up an R repository)

In the process, one might have to install ubuntu packages (over which certain R packages work)

1. Setting up a local R Repo using miniCRAN can be done by following the official Guide/ Specially created guide for the organization.

<LINK1>

<LINK2>

1. Install Apache2 for hosting the R packages
2. Once the repository is ready, host it in the apache server (Move the Entire R repository from the local system into the var/www/html/ path of apache2
3. Allow the IP of the VM (hosting the R repo) to be accessible by other VMs.
4. Install the packages into local R Environment by following the official documentation.
5. Include the installed packages and run the R code

Natural Language Processing (NLP)

**Natural language processing** (**NLP**) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data. The goal is a computer capable of "understanding" the contents of documents, including the contextual nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves.

Special use of Tesseract -OCR

Tesseract is an optical character recognition engine for various operating systems.

OCR extracts text from images and documents without a text layer and outputs the document into a new searchable text file, PDF, or most other popular formats

Natural Language Processing With R

R can be a good choice if you intend to quantify your language data for NLP purposes.

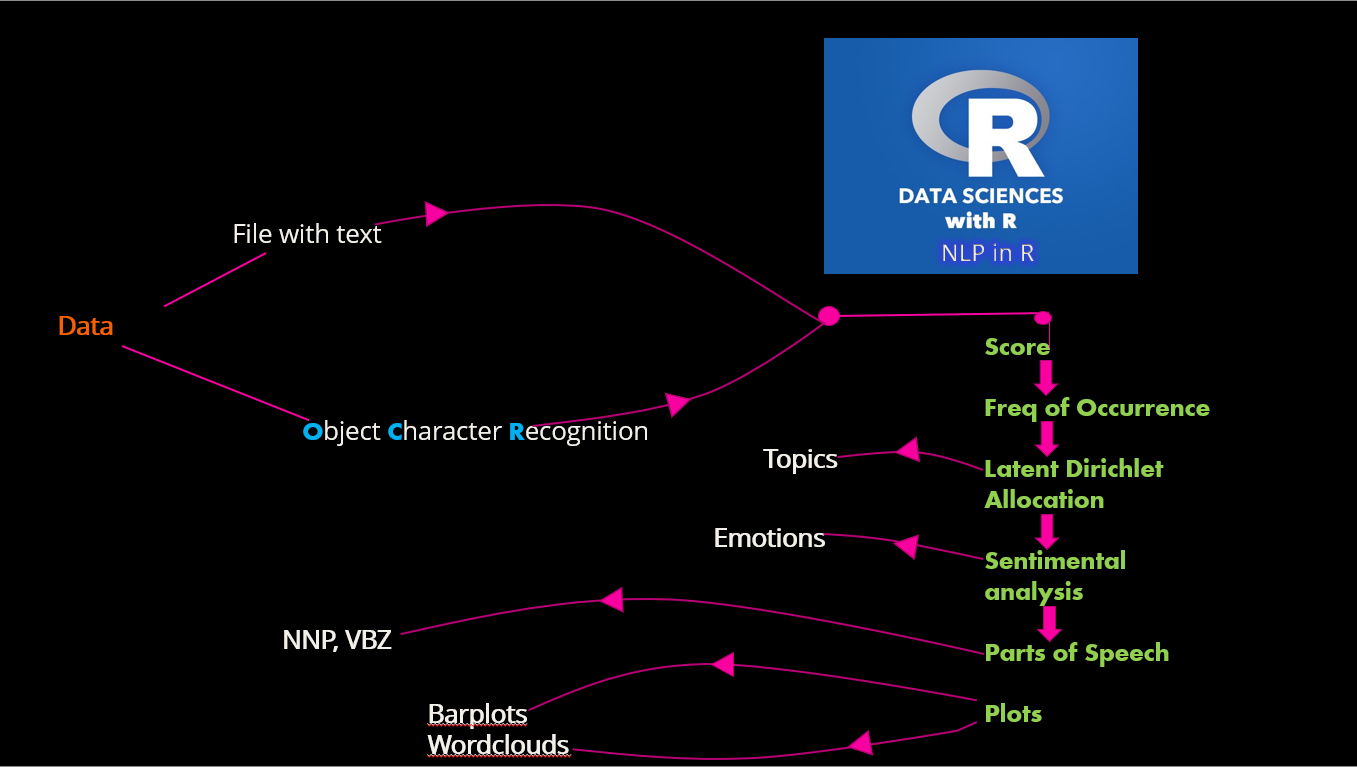
Things executed in the project – ‘NLP in R’

1. Read the data using tesseract OCR
   1. Input can a any image file format / PDF of Images
   2. Measure and visualize the accuracy of the tesseract OCR by writing a custom scoring snippet
2. The script offers
   1. Data Cleaning methods
   2. Freq of Occurrence
   3. Latent Dirichlet Allocation (Topic Modelling)
   4. Stemming/Lemma
   5. Sentimental Analysis
   6. N-grams
   7. Parts of speech Tagging & Named Entity Recognition

All the visualization of the results are done using

1. Barplots
2. Histograms
3. Wordcloud/Wordcloud2
4. R Markdown to get final HTML/PDF based output of the snippet along with the intermediate outputs

Finally, the code used for final presentation offered the following features



Extra features that were also executed

1. Hunspell for explicit Spell-error correction
2. correlation - co occurrence of key-words
3. N-grams

Further Efforts

Added ML/AI based r packages in the offline R repository

Include R packages for Digital Signal Processing/Audio processing (on Audio Files)

Speech to Text conversion Package (Support of Google)

Listed out Py based Packages to carry out NLP in Py

//////Custom script for R package Installation

Miscellaneous tasks/efforts

1. Developing the web page for the repository ‘Central Repository for data Science’ which will guide the end user on ‘how to work in the offline R environment’
2. Moving the VM from C to B to A
3. Changing the FQDN of a
4. Exploring Spark ( a Hadoop enhancement to MapReduce)
5. Working with DSA for sca

Tesseract, OCR, Changing FQDN, Sparks, Moving from C to B to A, DSA, Web page for CRDS