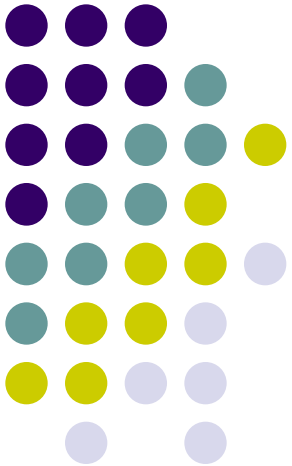
Applied Scaling & Classification Techniques in Political Science

Lecture 8

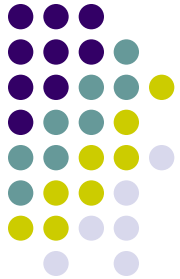
Supervised aggregated classification methods



Reference

- ✓ Grimmer, Justin, and Stewart, Brandon M. (2013). “Text as Data: The Promise and Pitfalls of Automatic Content Analysis Methods for Political Texts”. *Political Analysis*, 21(3): 267-297
- ✓ Ceron, Andrea, Curini, Luigi, Stefano M. Iacus (2016). “ iSA: a fast, scalable and accurate algorithm for sentiment analysis of social media content”, *Information Sciences*, 367–368 (1), 2016, 105–124

Measuring proportions



For many social science applications, only the proportion of documents in a category is needed, not the categories of each individual document

Shifting focus to estimating proportions, p(**C**), can lead to substantial improvements in accuracy - even if the documents are not randomly sampled from the corpus (more on this later)

Measuring proportions

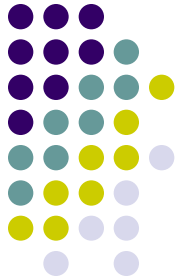
To understand how this new approach actually works, we have to introduce a change in the TDM of a corpus as we discussed up to now

Now we include in the TDM an **indicator** (0/1) of whether a word occurred in a document, rather than **counts** of the words

Post	Cat	Word: nuclear	Word: fear	Word: radiation	Word: pollution	Word: waste	Word: economic
post#1	like	1	0	0	0	0	1

Using this representation, let's define a probability distribution over all possible documents in the corpus p(**W**), where p(W1) in the example above is (1,0,0,0,0,1)

Measuring proportions



The data-generating process for the documents can be written as:

$$p(\boldsymbol{W}) = p(\boldsymbol{W}|\boldsymbol{C}) * p(\boldsymbol{C}),$$

where $p(\boldsymbol{W}|\boldsymbol{C})$ is the distribution of documents in the corpus conditional on categories and $p(\boldsymbol{C})$ is the proportion of documents in each class in the corpus - the quantity of our interest

Solving for $p(\boldsymbol{C})$, we have $p(\boldsymbol{C}) = p(\boldsymbol{W}) * p(\boldsymbol{W}|\boldsymbol{C})^{-1}$

Measuring proportions

$$p(\boldsymbol{C}) = p(\boldsymbol{W}) * p(\boldsymbol{W}|\boldsymbol{C})^{-1}$$

$p(\boldsymbol{W})$ is the distribution of the stems in the whole set (train + test). We have an accurate estimation here!

And what about $p(\boldsymbol{W}|\boldsymbol{C})$? It requires labeled documents - which are unavailable for the test set!

But if we assume that the conditional distributions **are identical in the training and test sets**, then we can estimate $p(\boldsymbol{W}|\boldsymbol{C})$ directly from the training-set

Also this, an accurate estimation (as long as the coders did a good job!)

Measuring proportions

$$p(\boldsymbol{C}) = p(\boldsymbol{W}) * p(\boldsymbol{W}|\boldsymbol{C})^{-1}$$

If $p(\boldsymbol{W})$ is the distribution of the stems in the whole set (train + test), $p(\boldsymbol{C})$ is going to be estimate for both the training-set and the test-set (Of course in this case $p(\boldsymbol{C})$ for the training-set is estimated perfectly!)

If $p(\boldsymbol{W})$ is the distribution of the stems only in the test-set, $p(\boldsymbol{C})$ is going to be estimate for the test-set

Measuring proportions

$$p(\boldsymbol{C}) = p(\boldsymbol{W}) * p(\boldsymbol{W}|\boldsymbol{C})^{-1}$$

Estimatig $p(\boldsymbol{C})$ for both the training-set and the test-set is a problem?

Generally not! You are interested **also** in the $p(\boldsymbol{C})$ included in the training-set and manually codified after all!

One problematic scenario: when # of documents in the training-set come from different days while you want to estimate the test that comes from a specific day. This could be a troublesome situation when the # of documents in the training-set are relatively large compared to the # of documents in the test-set

Still…you know the # of documents in the training-set and in the test-set, you know the % in the training-set and the % in the whole set (training-set+test-set). So you re-estimate the % just for the test-set!

Measuring proportions

Focusing on $p(\boldsymbol{W}|\boldsymbol{C})$ rather than $p(\boldsymbol{C}|\boldsymbol{W})$ as done in machine learning, has two advantages

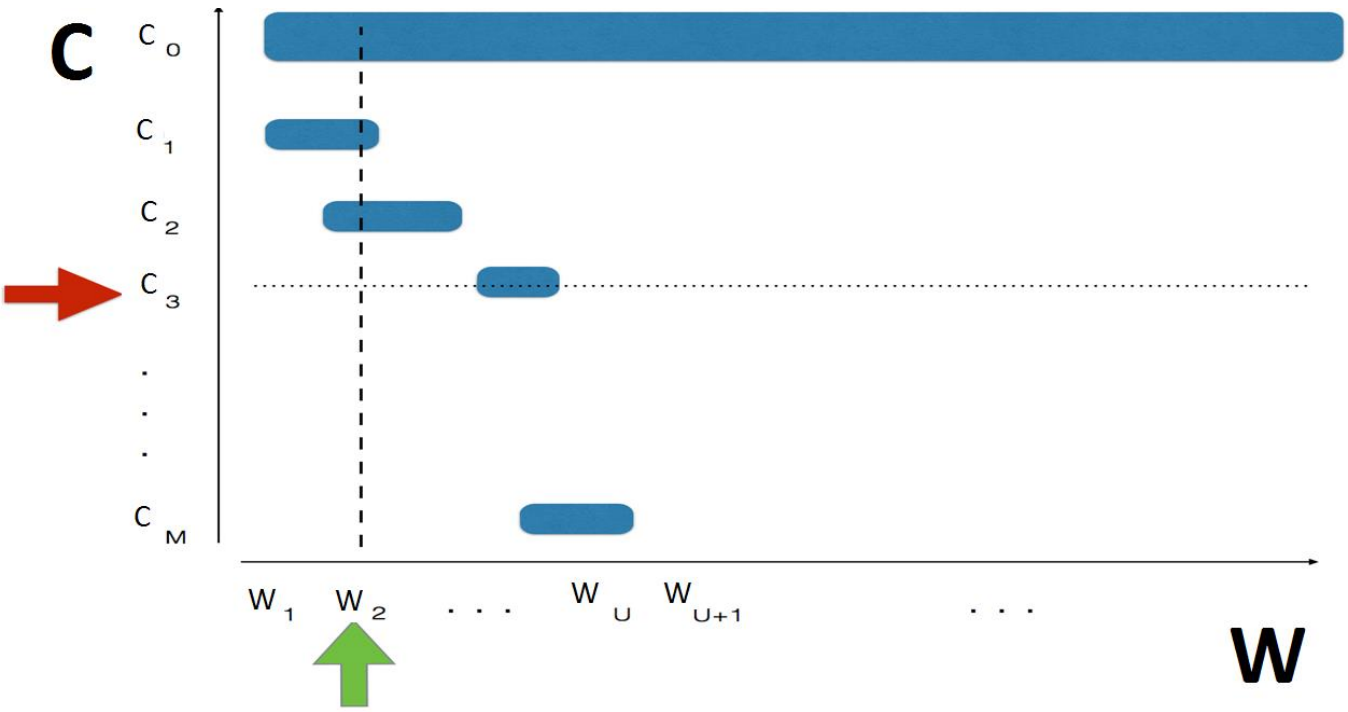
Theoretically: $p(\boldsymbol{W}|\boldsymbol{C})$ means: «given a post that is associated to a given content, which are the sequence of stems effectively employed to express that specific content»?

This makes a lot of sense: you do not start writing and only **afterwards** discover your sentiment toward for example a party. You start with a view, with a “category” in your mind (good, bad, support or not), and then set it out in words

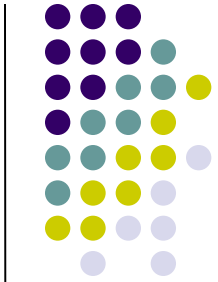
Measuring proportions

Focusing on $p(\boldsymbol{W}|\boldsymbol{C})$ rather than $p(\boldsymbol{C}|\boldsymbol{W})$ as done in machine learning, has two advantages

Empirically: the existence of a category Ci extremely frequent in a training-set can negatively affect $p(\boldsymbol{C}|\boldsymbol{W})$ but not $p(\boldsymbol{W}|\boldsymbol{C})$



Measuring proportions



No statistical property must be satisfied by the training set for this approach to work properly: the training set is **not a representative sample** of the population of texts

However, the language used in the training-set to express some given concept is **assumed** to be the same as in the whole population of posts, i.e. social media users use the same language

✓ Is it a **reasonable assumption**?

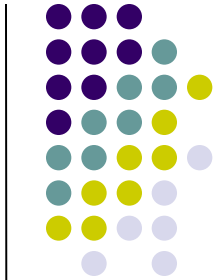
Measuring proportions

After all, in the **Oxford Dictionary** (English) you have **650k terms**

In reality, for any given topic, in the everyday language there is a tendency to use at the maximum between **200 and 500 *stems***

This is what **makes possible** the statistical analysis

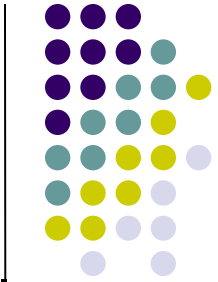
Measuring proportions



Two algorithms available for this type of analysis: ReadMe and iSA (different implementations of the same idea explained above)

ReadMe: to deal with sparsity, ReadMe solves the inverse problem saw above by subsetting of stems and averaging the results (bagging). Possible problems: slow, large variability of the estimates, unstable for large dimension of D, requires further bootstrap for std.err

Measuring proportions



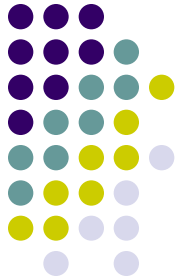
iSA: collapses the vector of stems into one-dimensional entity and solve the inverse problem in fraction of seconds

More in details (but read the Ceron, Curin and Iacus paper!):

- ✓ Each vector of stems, e.g. *sj* = (0, 1, 1, 0, . . . , 0, 1) is transformed into a string-sequence that we denote by *Cj* = "0110 · · · 01"; this is the first level of dimensionality reduction of the problem: from a *N* x *K* matrix to a one-dimensional vector *N* x *1*
- ✓ This sequence of 0's and 1's is further translated into hexadecimal notation such that the sequence '11110010' is recorded as λ = 'F2' or '11100101101' as λ = 'F2D', and so forth. So each text is represented by a label λ of shorter length

Implications: fast, memory saving (dimension reduction), reduced variability of the estimates, stable and scalable, exact std. err. possible

Measuring proportions



Validation when measuring proportions: how to do that?

You cannot use the usual accuracy measures (you do not make individual classifications!!!)

What you can still do is estimating for example the MAE (mean average error) across categories…

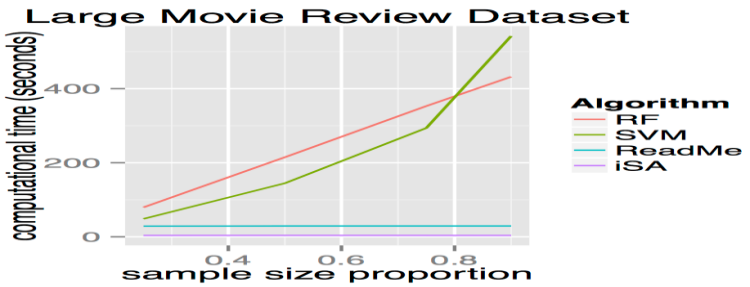
…moreover you can still apply a **cross-validation** procedure on your training-set!

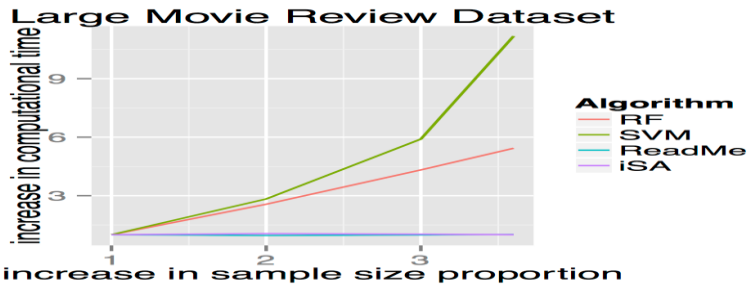
SASA approaches (Supervised Aggregated Sentiment Analysis) always better than ML at the aggregate level!

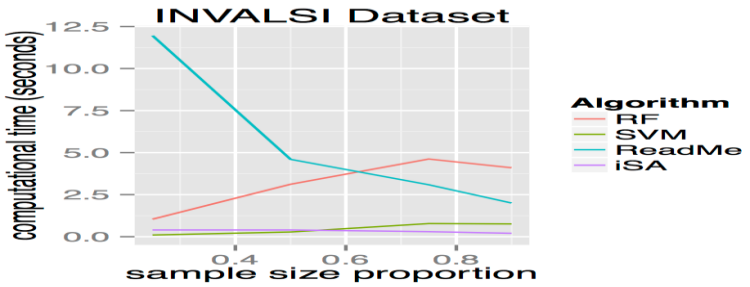
The intuition

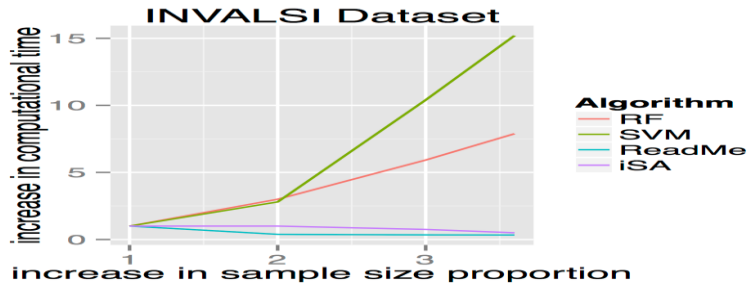
SASA approach does not look for the needle in the haystack. It looks at the shape of the haystack

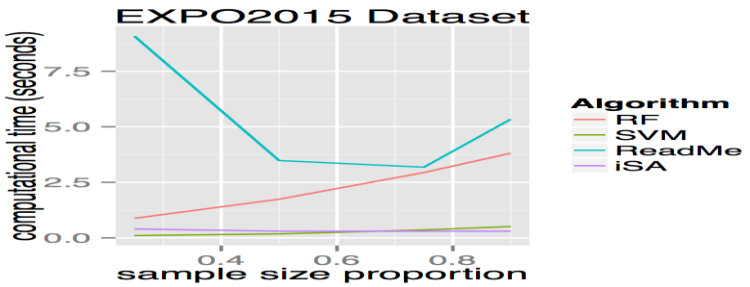
Computational efficiency

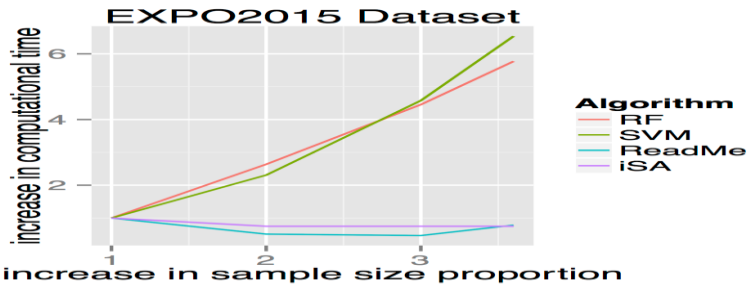




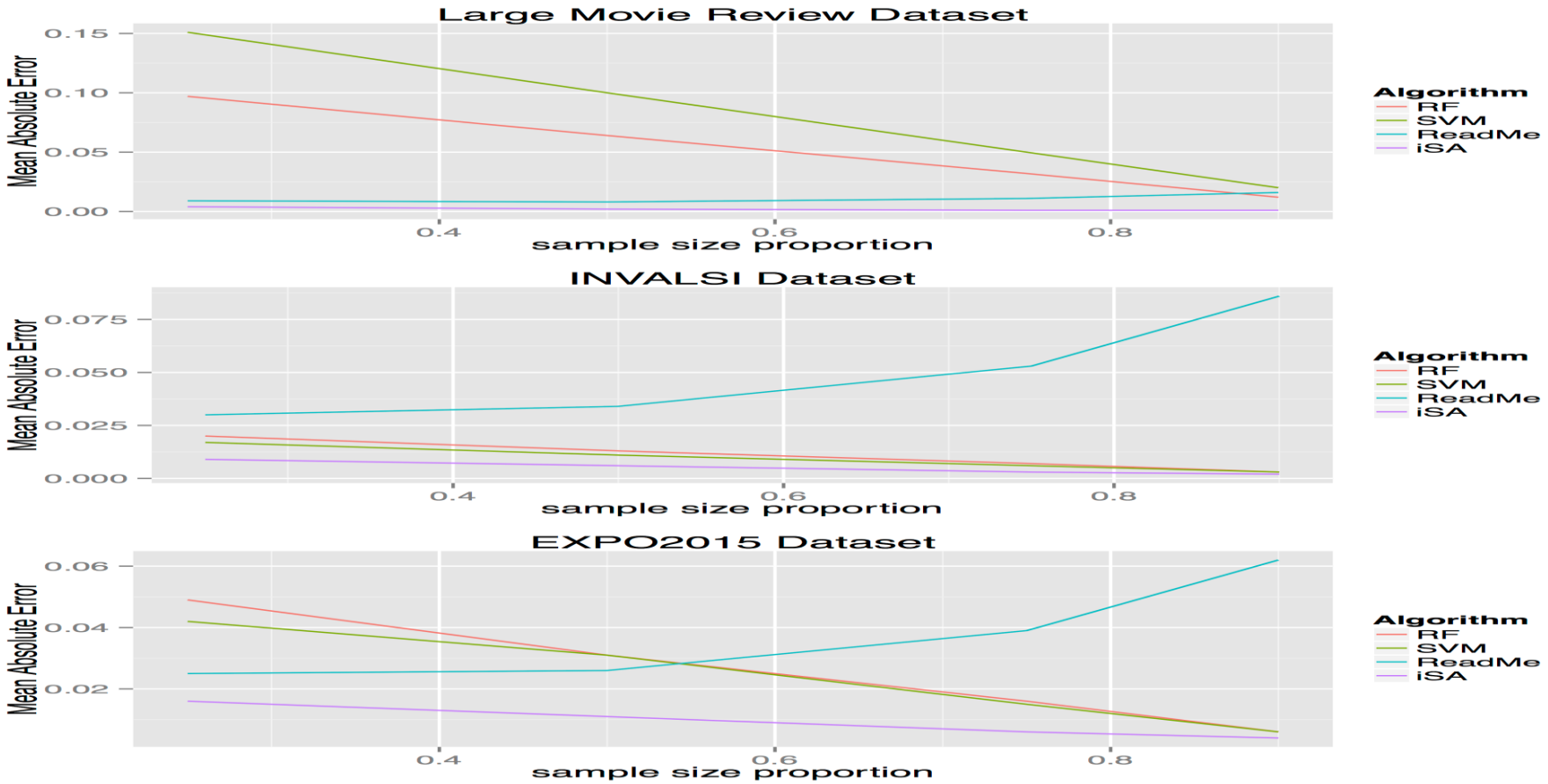




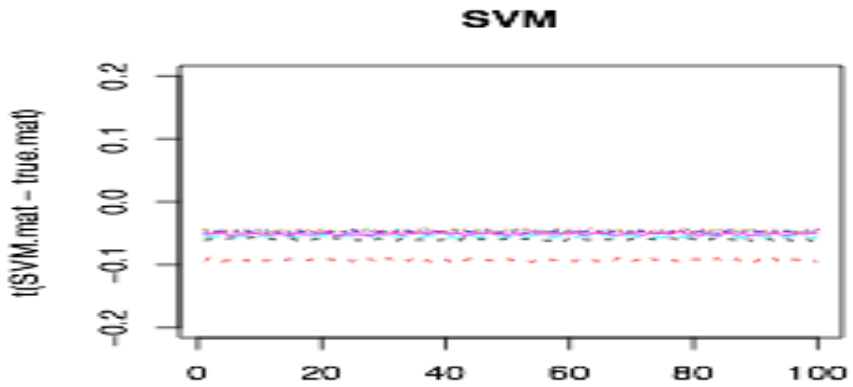


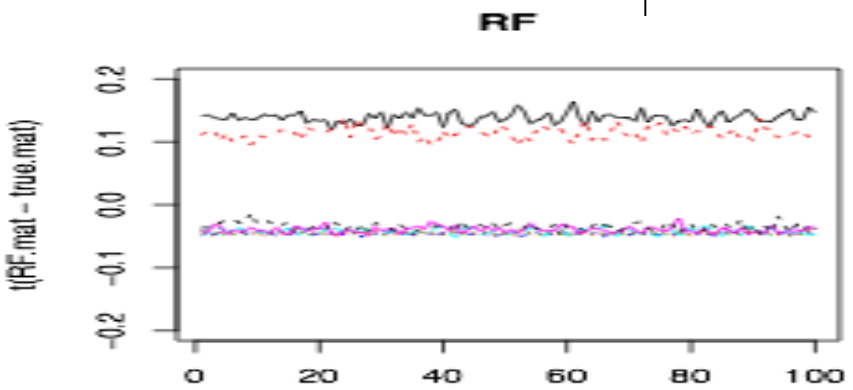


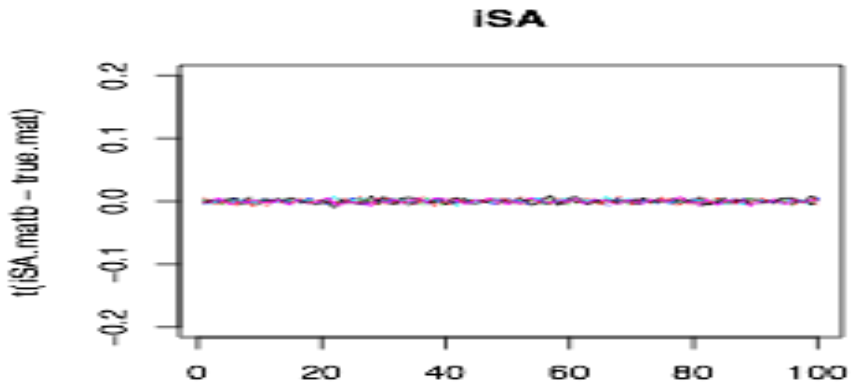
Bias

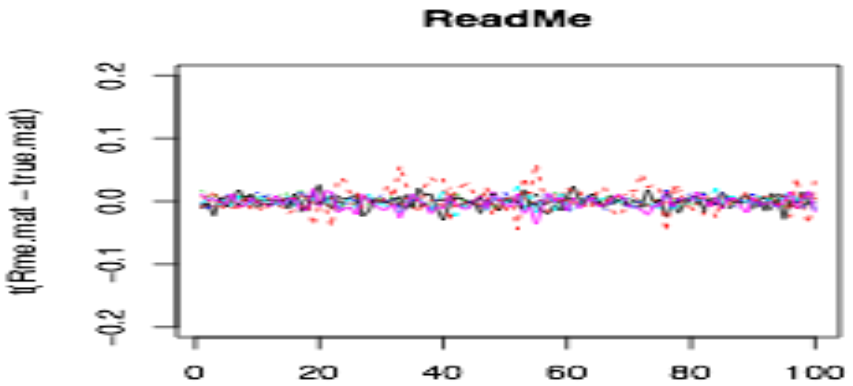

Large Movie Review Dataset
Mean Absolute Error
0.15
0.10
0.05
0.00
0.4
0.6
0.8
sample size proportion
Algorithm
RF
SVM
ReadMe
iSA
INVALSI Dataset
Mean Absolute Error
0.075
0.050
0.025
0.000
0.4
0.6
0.8
sample size proportion
Algorithm
RF
SVM
ReadMe
iSA
EXPO2015 Dataset
Mean Absolute Error
0.06
0.04
0.02
0.4
0.6
0.8
sample size proportion
Algorithm
RF
SVM
ReadMe
iSA

Variability

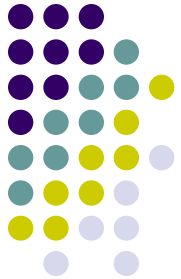








R pakcages to install



```
install.packages("VA", repos= "http://r.iq.harvard.edu",
    type="source")
install.packages("ReadMe", repos= "http://r.iq.harvard.edu",
    type="source")
library(devtools)
install_github(“blogsvoices/iSAX”)
```