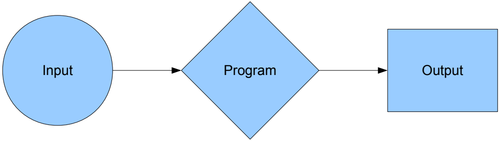
Why data mining?

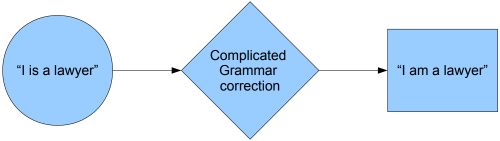
Edo Liberty

Old programing paradigm



- The input is small and the program can store/read it many times
- There is a lot of domain intelligence built into the program

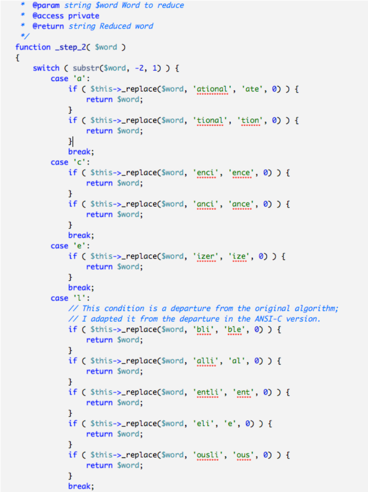
Old programing paradigm



- A short sentence is given to a grammar correction software.
- Programers and linguists produced code which is highly specialized.

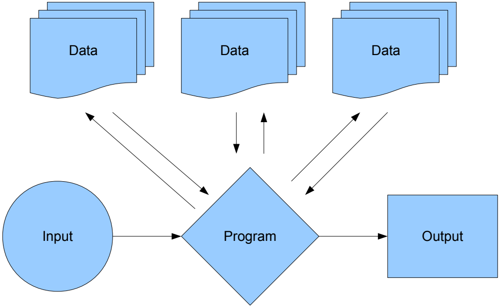
Old programing paradigm

Part of a stemming module (tiny fraction of the whole process)



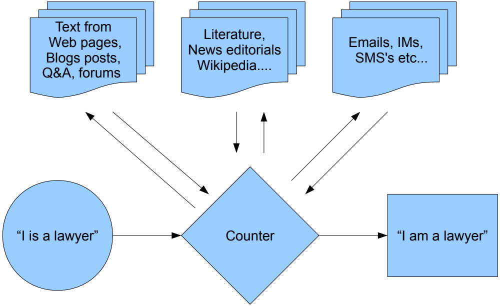
```
 *  @param string $word Word to reduce
 *  @access private
 *  @return string Reduced word
 */
function _step_2( $word )
{
    switch ( substr($word, -2, 1) ) {
        case 'a':
            if ( $this->_replace($word, 'ational', 'ate', 0) ) {
                return $word;
            }
            if ( $this->_replace($word, 'tional', 'tion', 0) ) {
                return $word;
            }
            break;
        case 'c':
            if ( $this->_replace($word, 'enci', 'ence', 0) ) {
                return $word;
            }
            if ( $this->_replace($word, 'anci', 'ance', 0) ) {
                return $word;
            }
            break;
        case 'e':
            if ( $this->_replace($word, 'izer', 'ize', 0) ) {
                return $word;
            }
            break;
        case 'l':
            // This condition is a departure from the original algorithm;
            // I adapted it from the departure in the ANSI-C version.
            if ( $this->_replace($word, 'bli', 'ble', 0) ) {
                return $word;
            }
            if ( $this->_replace($word, 'alli', 'al', 0) ) {
                return $word;
            }
            if ( $this->_replace($word, 'entli', 'ent', 0) ) {
                return $word;
            }
            if ( $this->_replace($word, 'eli', 'e', 0) ) {
                return $word;
            }
            if ( $this->_replace($word, 'ousli', 'ous', 0) ) {
                return $word;
            }
            break;
```

New programing paradigm



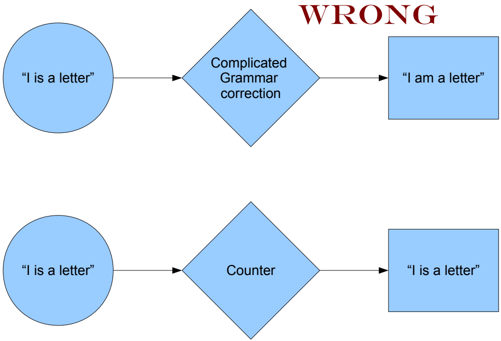
- There is a huge (virtually infinite) amount of data
- The “brain” is the data and not the program

New programing paradigm



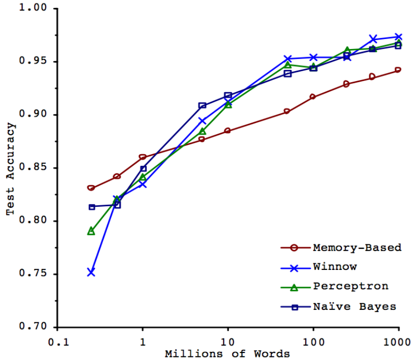
- “I is a lawyer” appeared 800,000 times usually like “i) is a lawyer ...” or “George I. is a lawyer” etc.
- “I am a lawyer” appeared as is 1,200,000 in respected sources.

New programing paradigm



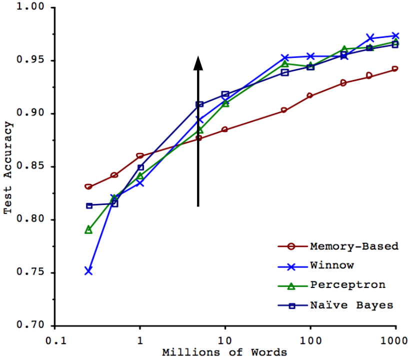
- “I is a letter” appeared 5,000,000 times
- “I am a letter” appeared only 200,000 times

New programing paradigm



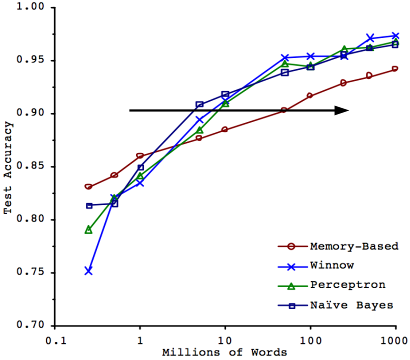
Michele Banko, Eric Brill: Scaling to Very Very Large Corpora for Natural Language Disambiguation.

New programing paradigm



Clearly, some algorithms perform better than others.

New programing paradigm



But, having more data is sometime more important than the algorithm...

Other examples where the data is “everything”

- Ranking / sorting search results
- Web advertising
- Text and image search
- Recommendation engines
- Fighting web abuse (spam, malware etc...)
- Spelling, Suggesting
- many many more...

So, you want more data?

Careful what you wish for!

- King James **Bible** - **1.4 MB**
- Only text on **WIkipedia** - **6.1 GB** (1GB = 1000MB)
- All ***.gov domain** on the web **1 TB** (1TB = 1000GB)
- Incoming **daily emails**[1] to Yahoo! **1-10 PB** (1PB = 1000GB)
- Size of the **internet**[2] **10 Exa-byte** (1EB = 1000 PB)

“Simply counting” doesn’t sound so easy any more....

[1] This number depends on whether you count spam, forwards, attachments and so on. The number I give is a conservative (and intentionally obfuscated) estimate of the amount of new raw text.

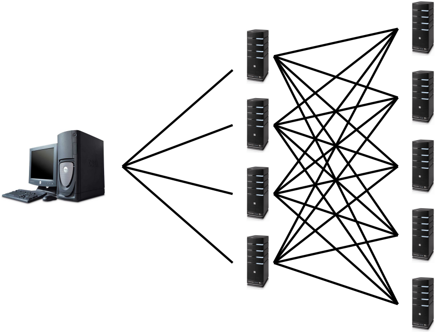
[2] The size of the internet is not really known, and it is even unclear what the word “size” means exactly in this context. However, 10EB is, to the best of my knowledge, a conservative estimate of the amount of text in publicly accessible static p

New paradigms for dealing with data

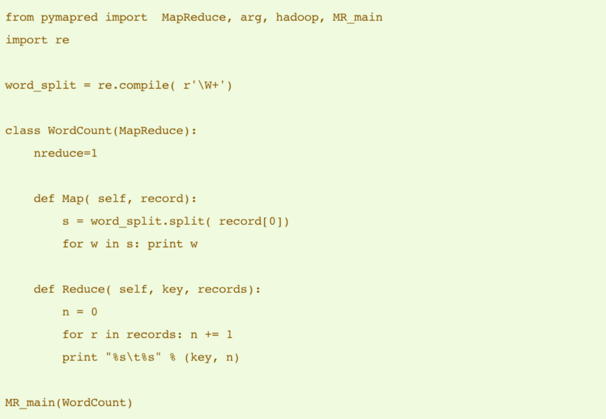
So, we need data mining...

- Specialized data structures, for example: bloom filters, tries and search indexes.
- New algorithmic tools like sampling, hashing, and sketching
- Streaming online algorithms, e.g., online linear regression, online classification
- Massively distributed computation like map-reduce or message passing.

Massively distributed systems (map-reduce)

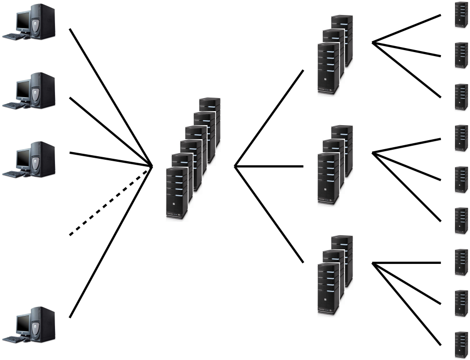


Massively distributed systems (map-reduce)



```
from pymapred import  MapReduce, arg, hadoop, MR_main
import re

word_split = re.compile( r'\W+')

class WordCount(MapReduce):
    nreduce=1

    def Map( self, record):
        s = word_split.split( record[0])
        for w in s: print w

    def Reduce( self, key, records):
        n = 0
        for r in records: n += 1
        print "%s\t%s" % (key, n)

MR_main(WordCount)
```

Massively distributed systems (searching)



What is the goal of this class?

It turns out the a rather small set of simple tools give us many and wonderful abilities. Most of these are randomized in nature:

- Sampling
- Hashing
- Streaming
- Estimating
- Embedding
- and Indexing and mapping

The goal of this class to make you as comfortable with these ideas as possible by learning many different manifestations of them.

What do we do today?

1. Intro presentation (this one)
2. Recup of basic probability, Markov's and Chebyshev's inequalities
3. Example problem; estimating set sizes