Get, Clean Data

Gaurav Sood

Spring 2015

Data, Data, Everywhere

Data, Data, Everywhere

– The Famous Five:
  Aural, Visual, Somatic, Gustatory, Olfactory

Data, Data, Everywhere

- The Famous Five:
  Aural, Visual, Somatic, Gustatory, Olfactory
- The Social Famous Five:
  What people (like to) hear, see, sense, smell, taste, . . .

Data, Data, Everywhere

– The Famous Five:
  Aural, Visual, Somatic, Gustatory, Olfactory

– The Social Famous Five:
  What people (like to) hear, see, sense, smell, taste, . . .

– Manifest Data:
  Likes, Ratings, Reviews, Comments, Views, Searches . . .

Data, Data, Everywhere

- The Famous Five:
  Aural, Visual, Somatic, Gustatory, Olfactory
- The Social Famous Five:
  What people (like to) hear, see, sense, smell, taste, . . .
- Manifest Data:
  Likes, Ratings, Reviews, Comments, Views, Searches . . .
- Data about data:
  Location of a tweet, photo, who called whom, . . .

Data, Data, Everywhere

- The Famous Five:
  Aural, Visual, Somatic, Gustatory, Olfactory
- The Social Famous Five:
  What people (like to) hear, see, sense, smell, taste, . . .
- Manifest Data:
  Likes, Ratings, Reviews, Comments, Views, Searches . . .
- Data about data:
  Location of a tweet, photo, who called whom, . . .
- Social data:
  Friend graph, followers, who retweeted, liked,. . .

Data, Data, Everywhere

- The Famous Five:
  Aural, Visual, Somatic, Gustatory, Olfactory
- The Social Famous Five:
  What people (like to) hear, see, sense, smell, taste, . . .
- Manifest Data:
  Likes, Ratings, Reviews, Comments, Views, Searches . . .
- Data about data:
  Location of a tweet, photo, who called whom, . . .
- Social data:
  Friend graph, followers, who retweeted, liked,. . .
- Data about structure:
  Layout of the site, In/out links, . . .

Collecting Digital Data

– Proprietary Data collections
  Lexis-Nexis, comScore . . .

Collecting Digital Data

- Proprietary Data collections
  Lexis-Nexis, comScore . . .
- APIs
  Facebook, NY Times, Twitter, Google, FourSquare, Jstor, Zillow . . .

Collecting Digital Data

- Proprietary Data collections
  Lexis-Nexis, comScore . . .
- APIs
  Facebook, NY Times, Twitter, Google, FourSquare, Jstor, Zillow . . .
- Bulk Downloads
  Wikipedia, data.gov, IMDB, Million Song Database, Google n-grams . . .

Collecting Digital Data

- Proprietary Data collections
  Lexis-Nexis, comScore . . .
- APIs
  Facebook, NY Times, Twitter, Google, FourSquare, Jstor, Zillow . . .
- Bulk Downloads
  Wikipedia, data.gov, IMDB, Million Song Database, Google n-grams . . .
- Scraping

Collecting Digital Data

- Proprietary Data collections
  Lexis-Nexis, comScore . . .
- APIs
  Facebook, NY Times, Twitter, Google, FourSquare, Jstor, Zillow . . .
- Bulk Downloads
  Wikipedia, data.gov, IMDB, Million Song Database, Google n-grams . . .
- Scraping
- Custom Apps
  Build custom apps to observe behavior, get (pay) people to download these apps

Scraping

- To analyze data, we typically need structure.
  For instance, same number of rows for each column.

Scraping

- To analyze data, we typically need structure.
  For instance, same number of rows for each column.
- But found data often with human readable structure.
- Copy and paste, type, to a dataset.

Scraping

- To analyze data, we typically need structure.
  For instance, same number of rows for each column.
- But found data often with human readable structure.
- Copy and paste, type, to a dataset.

Scraping

- To analyze data, we typically need structure.
  For instance, same number of rows for each column.
- But found data often with human readable structure.
- Copy and paste, type, to a dataset.
- But error prone, and not scalable.

Scraping

- To analyze data, we typically need structure.
  For instance, same number of rows for each column.
- But found data often with human readable structure.
- Copy and paste, type, to a dataset.
- But error prone, and not scalable.
- Idea: Find the less accessible structure, automate based on it.

Collecting Found Digital Data

- Software

Collecting Found Digital Data

- Software
  - R - Not the best but will do.

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .
- Some things to keep in mind

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .
- Some things to keep in mind
  - Check if there is an API, or if data are available for download

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .
- Some things to keep in mind
  - Check if there is an API, or if data are available for download
  - Play Nice:

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .
- Some things to keep in mind
  - Check if there is an API, or if data are available for download
  - Play Nice:
    - Scraper may be disallowed in ‘robots.txt’

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .
- Some things to keep in mind
  - Check if there is an API, or if data are available for download
  - Play Nice:
    - Scraper may be disallowed in ‘robots.txt’
    - Build lag between requests. Make lags random.

Collecting Found Digital Data

- Software
  - R - Not the best but will do.
  - Python, Ruby, Perl, Java, . . .
  - 30 Digits, 80 Legs, Grepsr . . .
- Some things to keep in mind
  - Check if there is an API, or if data are available for download
  - Play Nice:
    - Scraper may be disallowed in ‘robots.txt’
    - Build lag between requests. Make lags random.
    - Scrape during off-peak hours

Paper

TELEVISION & CABLE

FACTBOOK®

TV STATIONS 1996

e Authoritative Reference for the Television, Cable & Electronics Industries

Published by Warren Publishing, Inc., 2115 Ward Court, N.W., Washington, D.C. 20037

Stations Volume No. ● 64 ● 1996 Edition

Broadcasting's best friend is home again.

The RCA® broadcast equipment line is now part of Comark Communications Inc.

COMARK

Route 309 & Advance Lane • Colmar, PA 18915
TEL: (215) 822-0777 • FAX: (215) 822-9129

THOMCAST

© 1996 Comark Communications, Inc.

Paper

- Create digital images of paper

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)
- Software

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)
- Software
    - Adobe Pro., etc.

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)
- Software
  - Adobe Pro., etc.
  - Best in class commercial: Abbyy FineReader
    Now has an API

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)
- Software
  - Adobe Pro., etc.
  - Best in class commercial: Abbyy FineReader
    Now has an API
  - Best in class open-source: Tesseract

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)
- Software
  - Adobe Pro., etc.
  - Best in class commercial: Abbyy FineReader
    Now has an API
  - Best in class open-source: Tesseract
- Scrape off recognized characters: pyPdf etc.

Paper

- Create digital images of paper
- Identify colored pixels as characters (OCR)
- Software
  - Adobe Pro., etc.
  - Best in class commercial: Abbyy FineReader
    Now has an API
  - Best in class open-source: Tesseract
- Scrape off recognized characters: pyPdf etc.
- Post-processing

Pictures, Audio, and Video

- Audio (or Video with audio) to text: Dragon Dictates, Google transcription

Pictures, Audio, and Video

- Audio (or Video with audio) to text: Dragon Dictates, Google transcription
- Pictures: recognize color, faces

Pictures, Audio, and Video

- Audio (or Video with audio) to text: Dragon Dictates, Google transcription
- Pictures: recognize color, faces
- Objects in images: Clarifai

Pictures, Audio, and Video

- Audio (or Video with audio) to text: Dragon Dictates, Google transcription
- Pictures: recognize color, faces
- Objects in images: Clarifai
- Scrape closed-captions

Get Others to Work

- Human Computing

Get Others to Work

- Human Computing
- Amazon.com's Mechanical Turk

Get Others to Work

- Human Computing
- Amazon.com's Mechanical Turk
  - Create Human Intensive Tasks (HITs)

Get Others to Work

- Human Computing
- Amazon.com's Mechanical Turk
    - Create Human Intensive Tasks (HITs)
    - Surveys, transcription, translation, . . .

Get Others to Work

- Human Computing
- Amazon.com's Mechanical Turk
    - Create Human Intensive Tasks (HITs)
    - Surveys, transcription, translation, ...
    - You assess the work and pay out

Get Others to Work

- Human Computing
- Amazon.com's Mechanical Turk
  - Create Human Intensive Tasks (HITs)
  - Surveys, transcription, translation, ...
  - You assess the work and pay out
- Odesk, elance, impact sourcing, run your own ads ...

Get Others to Work

- Human Computing
- Amazon.com's Mechanical Turk
  - Create Human Intensive Tasks (HITs)
  - Surveys, transcription, translation, . . .
  - You assess the work and pay out
- Odesk, elance, impact sourcing, run your own ads . . .
- Google – surveys as payment for content

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

```
from BeautifulSoup import BeautifulSoup
```

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

```
from BeautifulSoup import BeautifulSoup

from urllib import urlopen
```

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

```
from BeautifulSoup import BeautifulSoup

from urllib import urlopen


url  = urlopen('http://bit.ly/1D7wKcH').read()
```

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

```
from BeautifulSoup import BeautifulSoup

from urllib import urlopen


url  = urlopen('http://bit.ly/1D7wKcH').read()

soup = BeautifulSoup(url)
```

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

```
from BeautifulSoup import BeautifulSoup

from urllib import urlopen


url  = urlopen('http://bit.ly/1D7wKcH').read()

soup = BeautifulSoup(url)

text = soup.p.contents
```

Scraping one HTML page in Python

Shakespeare's Twelfth Night
Using Beautiful Soup

```
from BeautifulSoup import BeautifulSoup

from urllib import urlopen


url  = urlopen('http://bit.ly/1D7wKcH').read()

soup = BeautifulSoup(url)

text = soup.p.contents

print text
```

Getting text from one pdf in Python

A Political Ad
Using PyPdf

```
import pyPdf
```

Getting text from one pdf in Python

A Political Ad
Using PyPdf

```
import pyPdf


pdf = pyPdf.PdfFileReader(file('path to pdf', 'rb'))
```

Getting text from one pdf in Python

A Political Ad
Using PyPdf

```
import pyPdf


pdf = pyPdf.PdfFileReader(file('path to pdf', ‘rb'))

content = pdf.getPage(0).extractText()
```

Getting text from one pdf in Python

A Political Ad
Using PyPdf

```
import pyPdf


pdf = pyPdf.PdfFileReader(file('path to pdf', 'rb'))

content = pdf.getPage(0).extractText()

print content
```

Scraping many urls/files to structured data

- Loop, exploiting structure of the urls/file paths

Scraping many urls/files to structured data

- Loop, exploiting structure of the urls/file paths
  e.g. ESPN URL

Scraping many urls/files to structured data

- Loop, exploiting structure of the urls/file paths
  e.g. ESPN URL
- Handle errors, if files or urls don't open, what do you do?

Scraping many urls/files to structured data

- Loop, exploiting structure of the urls/file paths
  e.g. ESPN URL
- Handle errors, if files or urls don't open, what do you do?
- To harvest structured data, exploit structure within text

Scraping many urls/files to structured data

- Loop, exploiting structure of the urls/file paths
  e.g. ESPN URL
- Handle errors, if files or urls don't open, what do you do?
- To harvest structured data, exploit structure within text
- Trigger words, html tags, ...

Exception(al) Handling

```
try:
    pdf = pyPdf.PdfFileReader(file(pdfFile, 'rb'))
```

Exception(al) Handling

```
try:
     pdf = pyPdf.PdfFileReader(file(pdfFile, ’rb’))
except Exception, e:
     return ‘Cannot Open: %s with error: %s’ %
(pdfFile, str(e))
```

Inside the page

- Chrome Developer Tools

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
    - Tags begin with < and end with >

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.
  - Mozilla HTML elements

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.
  - Mozilla HTML elements
  - <p> is for paragraph

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.
  - Mozilla HTML elements
  - <p> is for paragraph
  - <a> is for a link

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.
  - Mozilla HTML elements
  - <p> is for paragraph
  - <a> is for a link
  - <ol>, <ul> is for ordered, unordered list; <li> is a bullet

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.
  - Mozilla HTML elements
  - <p> is for paragraph
  - <a> is for a link
  - <ol>, <ul> is for ordered, unordered list; <li> is a bullet
  - tags can have attributes. <a href='http://somesite'></a>

Inside the page

- Chrome Developer Tools
- Quick Tour of HTML
  - Tags begin with < and end with >
  - Tags usually occur in pairs. Some don't (see img). And can be nested.
  - Mozilla HTML elements
  - <p> is for paragraph
  - <a> is for a link
  - <ol>, <ul> is for ordered, unordered list; <li> is a bullet
  - tags can have attributes. <a href='http://somesite'></a>
  - DOM, hierarchical, parent, child:

```
<html>
    <body>
        <p></p>
    </body>
</html>
```

Find Things

Navigate by HTML tags:

```
soup.title, soup.body, soup.body.contents
```

Find Things

Navigate by HTML tags:

```
soup.title, soup.body, soup.body.contents
```

Search HTML tags:

```
soup.find_all('a'), soup.find(id="nav1")
```

Find Things

Navigate by HTML tags:

```
soup.title, soup.body, soup.body.contents
```

Search HTML tags:

```
soup.find_all('a'), soup.find(id="nav1")
```

So to get all the urls in a page:

Find Things

Navigate by HTML tags:

```
soup.title, soup.body, soup.body.contents
```

Search HTML tags:

```
soup.find_all('a'), soup.find(id="nav1")
```

So to get all the urls in a page:

```
for link in soup.find_all('a'):
     print(link.get('href'))
```

Find Things

Navigate by HTML tags:

```
soup.title, soup.body, soup.body.contents
```

Search HTML tags:

```
soup.find_all('a'), soup.find(id="nav1")
```

So to get all the urls in a page:

```
for link in soup.find_all('a'):
    print(link.get('href'))
```

Beautiful Soup Documentation

Data Munging

“Data scientists, according to interviews and expert estimates, spend from 50 percent to 80 percent of their time mired in the mundane labor of collecting and preparing data, before it can be explored for useful information.”

New York Times: For BigData Scientists, ‘Janitor Work’ Is Key Hurdle to Insights

Data Munging

"In our experience, the tasks of exploratory data mining and data cleaning constitute 80% of the effort that determines 80% of the value of the ultimate data."

Dasu and Johnson, Exploratory Data Mining and Data Cleaning

Regular (or Rational) Expressions

- Formal language for specifying text strings

Regular (or Rational) Expressions

- Formal language for specifying text strings
- Stephen Kleene, ‘inventor’ of regular expressions.

Regular (or Rational) Expressions

- Formal language for specifying text strings
- Stephen Kleene, ‘inventor’ of regular expressions.
- Henry Spencer, behind the `regex` library.

Regular (or Rational) Expressions

- Formal language for specifying text strings
- Stephen Kleene, ‘inventor’ of regular expressions.
- Henry Spencer, behind the `regex` library.
- Descend from *finite automata* theory.

Regular (or Rational) Expressions

- Formal language for specifying text strings
- Stephen Kleene, ‘inventor’ of regular expressions.
- Henry Spencer, behind the `regex` library.
- Descend from *finite automata* theory.
- Matching

The most basic regular expression

- String literal

The most basic regular expression

- String literal
- RegexPal.com
- Say you are searching for the word apple – can be uppercase first character, plural, lowercase first character

Disjunction

- Disjunction, Character classes

Disjunction

- Disjunction, Character classes
  - `[]`

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`
  - `[0123456789]  matches any digit`

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`
  - `[0123456789]  matches any digit`
- Ranges

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`
  - `[0123456789]  matches any digit`
- Ranges
  - `[0-9]  matches any digit`

Disjunction

- Disjunction, Character classes
    - `[]`
    - `[aA]pple matches apple and Apple`
    - `[0123456789]  matches any digit`
- Ranges
    - `[0-9]  matches any digit`
    - `[a-z], [[:lower:]]  matches any lowercase`

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`
  - `[0123456789]  matches any digit`
- Ranges
  - `[0-9]  matches any digit`
  - `[a-z], [[:lower:]]  matches any lowercase`
  - `[a-zA-Z], [[:alpha:]]  matches any uppercase`

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`
  - `[0123456789]  matches any digit`
- Ranges
  - `[0-9]  matches any digit`
  - `[a-z], [[:lower:]]  matches any lowercase`
  - `[a-zA-Z], [[:alpha:]]  matches any uppercase`
  - `[a-e1-9]  matches any letter or digit`

Disjunction

- Disjunction, Character classes
  - `[]`
  - `[aA]pple matches apple and Apple`
  - `[0123456789]  matches any digit`
- Ranges
  - `[0-9]  matches any digit`
  - `[a-z], [[:lower:]]  matches any lowercase`
  - `[a-zA-Z], [[:alpha:]]  matches any uppercase`
  - `[a-e1-9]  matches any letter or digit`
  - Hyphen only has a special meaning if used within range.

    [-123]

Disjunction Contd..

- Negation in Disjunction

Disjunction Contd..

- Negation in Disjunction
  - `^ right after the square bracket means a negation`

Disjunction Contd..

- Negation in Disjunction
  - `^` right after the square bracket means a negation
  - `[^A-Z]`

Disjunction Contd..

- Negation in Disjunction
    - `^` right after the square bracket means a negation
    - `[^A-Z]`
    - `[^Aa]` means neither a capital A nor a lowercase a

Disjunction Contd..

- Negation in Disjunction
  - ^ right after the square bracket means a negation
  - [^A-Z]
  - [^Aa] means neither a capital A nor a lowercase a
  - [^e^] means not an e, and not ^

Disjunction Contd..

- Negation in Disjunction
  - `^` right after the square bracket means a negation
  - `[^A-Z]`
  - `[^Aa]` means neither a capital A nor a lowercase a
  - `[^e^]` means not an e, and not ^
- Disjunction for longer strings

Disjunction Contd..

- Negation in Disjunction
  - `^` right after the square bracket means a negation
  - `[^A-Z]`
  - `[^Aa]` means neither a capital A nor a lowercase a
  - `[^e^]` means not an e, and not `^`
- Disjunction for longer strings
  - pipe

Disjunction Contd..

- Negation in Disjunction
  - ^ right after the square bracket means a negation
  - [^A-Z]
  - [^Aa] means neither a capital A nor a lowercase a
  - [^e^] means not an e, and not ^
- Disjunction for longer strings
  - pipe
  - a|b|c = [abc]

Disjunction Contd..

- Negation in Disjunction
  - `^` right after the square bracket means a negation
  - `[^A-Z]`
  - `[^Aa]` means neither a capital A nor a lowercase a
  - `[^e^]` means not an e, and not ^
- Disjunction for longer strings
  - pipe
  - `a|b|c = [abc]`
  - `apple|pie`

Disjunction Contd..

- Negation in Disjunction
  - `^` right after the square bracket means a negation
  - `[^A-Z]`
  - `[^Aa]` means neither a capital A nor a lowercase a
  - `[^e^]` means not an e, and not ^
- Disjunction for longer strings
  - pipe
  - `a|b|c = [abc]`
  - `apple|pie`
  - `[aA]pple|[aA]nd`

Special characters

- ? - previous character is optional: colou?r - color, colour

Special characters

- ? - previous character is optional: colou?r - color, colour
- . matches any character
  e.g. beg.n matches begun, begin, began

Special characters

- ? - previous character is optional: colou?r - color, colour
- . matches any character
  e.g. beg.n matches begun, begin, began
- Kleene Operators - named after Steven Kleene

Special characters

- ? - previous character is optional: colou?r - color, colour
- . matches any character
  e.g. beg.n matches begun, begin, began
- Kleene Operators - named after Steven Kleene
  - * matches 0 or more of the previous characters
    e.g. oo*h will match ooh, oooh, etc.
    (abc)* will match abc, abcabc, etc.

Special characters

- ? - previous character is optional: colou?r - color, colour
- . matches any character
  e.g. beg.n matches begun, begin, began
- Kleene Operators - named after Steven Kleene
  - * matches 0 or more of the previous characters
    e.g. oo*h will match ooh, oooh, etc.
    (abc)* will match abc, abcabc, etc.
  - + matches 1 or more of the previous characters
    e.g. o+h will match ooh, oooh, etc.

Repetition Ranges

- Specific ranges can also be specified

Repetition Ranges

- Specific ranges can also be specified
- `{ }` to specify range for the immediately preceding regex

Repetition Ranges

- Specific ranges can also be specified
- `{ }` to specify range for the immediately preceding regex
- `{n}` means exactly n occurrences

Repetition Ranges

- Specific ranges can also be specified
- `{ }` to specify range for the immediately preceding regex
- `{n}` means exactly n occurrences
- `{n,}` means at least n occurrences

Repetition Ranges

- Specific ranges can also be specified
- `{ }` to specify range for the immediately preceding regex
- `{n}  means exactly n occurrences`
- `{n,} means at least n occurrences`
- `{n,m} means at least n and no more than m occurrences`

Repetition Ranges

- Specific ranges can also be specified
- `{ }` to specify range for the immediately preceding regex
- `{n}` means exactly n occurrences
- `{n,}` means at least n occurrences
- `{n,m}` means at least n and no more than m occurrences
- Example:

```
. {0, } = .*
```

More Regex

- Anchors

More Regex

- Anchors
  - matches the beginning of the line

    e.g.

    `^[A-Z] matches a captial letter at the start of a line.`

More Regex

- Anchors
  - matches the beginning of the line
    e.g.

    `^[A-Z] matches a captial letter at the start of a line.`
  - $ matches the end of the line.

More Regex

- Anchors
  - matches the beginning of the line
    e.g.
    `^[A-Z] matches a captial letter at the start of a line.`
  - $ matches the end of the line.
- `\. means a period`

More Regex

- Anchors
  - matches the beginning of the line

    e.g.

    `^[A-Z] matches a captial letter at the start of a line.`
  - $ matches the end of the line.
- `\. means a period`
- Example: look for the word ‘the’

More Regex

- Anchors
  - matches the beginning of the line
    e.g.
    `^[A-Z] matches a captial letter at the start of a line.`
  - $ matches the end of the line.
- `\. means a period`
- Example: look for the word ‘the’
  - missed capitalization: [tT]he

More Regex

- Anchors
  - matches the beginning of the line

    e.g.

    `^[A-Z] matches a captial letter at the start of a line.`
  - $ matches the end of the line.
- `\. means a period`
- Example: look for the word ‘the’
  - missed capitalization: [tT]he
  - make pattern more precise:

    `[tT]he[^A-Za-z], ^[tT]he[^A-Za-z]`

False Positive and Negatives

- False positives or Type 1 errors - matching things we shouldn't match

False Positive and Negatives

- False positives or Type 1 errors - matching things we shouldn’t match
- False negatives or Type 2 errors - not matching things we should match

False Positive and Negatives

- False positives or Type 1 errors - matching things we shouldn't match
- False negatives or Type 2 errors - not matching things we should match
- Cost attached to false negative and positive

False Positive and Negatives

- False positives or Type 1 errors - matching things we shouldn't match
- False negatives or Type 2 errors - not matching things we should match
- Cost attached to false negative and positive
- Provide some metrics by comparing against good data for a small sample

Edit Distance

- pwned -> owned or pawned?

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?
- Applications

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?
- Applications
  - Spell Correction

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?
- Applications
  - Spell Correction
  - Also comes up in computational biology

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?
- Applications
    - Spell Correction
    - Also comes up in computational biology
    - Machine translation

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?
- Applications
  - Spell Correction
  - Also comes up in computational biology
  - Machine translation
  - Information extraction

Edit Distance

- pwned -> owned or pawned?
- standd -> strand, stand, stood, or sand?
- How similar are two strings?
- Applications
  - Spell Correction
  - Also comes up in computational biology
  - Machine translation
  - Information extraction
  - Speech recognition

Edit Distance

- Typically refers to minimum edit distance

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another
    - Insertion
    - Deletion
    - Substitution
- e.g. two strings: intention, execution

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another
    - Insertion
    - Deletion
    - Substitution
- e.g. two strings: intention, execution
    - align it with second letter

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another
    - Insertion
    - Deletion
    - Substitution
- e.g. two strings: intention, execution
    - align it with second letter
    - d (delete), s (substitute), s, i(nsert), s

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another
  - Insertion
  - Deletion
  - Substitution
- e.g. two strings: intention, execution
  - align it with second letter
  - d (delete), s (substitute), s, i(nsert), s
  - if each operation costs 1, edit distance = 5

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another
  - Insertion
  - Deletion
  - Substitution
- e.g. two strings: intention, execution
  - align it with second letter
  - d (delete), s (substitute), s, i(nsert), s
  - if each operation costs 1, edit distance = 5
  - if substitition cost 2 (levenshtein distance), distance = 8

Edit Distance

- Typically refers to minimum edit distance
- Minimum number of editing operations to convert one string to another
    - Insertion
    - Deletion
    - Substitution
- e.g. two strings: intention, execution
    - align it with second letter
    - d (delete), s (substitute), s, i(nsert), s
    - if each operation costs 1, edit distance = 5
    - if substitition cost 2 (levenshtein distance), distance = 8
- You can implement this at word level so Microsoft Corp. is 1 away from Microsoft.

Text Processing

Text as Data

- Bag of words assumption
  Lose word order

Text as Data

- Bag of words assumption

  Lose word order

- Remove stop words:

  If, and, but, who, what, the, they, their, a, or, . . .

  Be careful: one person's stopword is another's key term.

Text as Data

- Bag of words assumption
  Lose word order

- Remove stop words:
  If, and, but, who, what, the, they, their, a, or, . . .
  Be careful: one person's stopword is another's key term.

- (Same) Word: Stemming and Lemmatization
  Taxing, taxes, taxation, taxable $\rightsquigarrow$ tax

Text as Data

- Bag of words assumption
  Lose word order

- Remove stop words:
  If, and, but, who, what, the, they, their, a, or, . . .
  Be careful: one person's stopword is another's key term.

- (Same) Word: Stemming and Lemmatization
  Taxing, taxes, taxation, taxable $\rightsquigarrow$ tax

- Remove rare words
  $\sim$ .5% to 15%, depending on application

Text as Data

- Bag of words assumption
  Lose word order

- Remove stop words:
  If, and, but, who, what, the, they, their, a, or, . . .
  Be careful: one person's stopword is another's key term.

- (Same) Word: Stemming and Lemmatization
  Taxing, taxes, taxation, taxable $\rightsquigarrow$ tax

- Remove rare words
  $\sim$ .5% to 15%, depending on application

- Convert to lowercase, drop numbers, punctuation, etc.

How?

Using Natural Language Toolkit (`nltk`)

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
```

How?

Using Natural Language Toolkit (`nltk`)

- Lowercase:

  ```
  text = text.lower()
  ```

- Remove stop words:

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
- Remove stop words:
  swords = stopwords.words('english')
```

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
- Remove stop words:
  swords = stopwords.words('english')
  words  = wordpunct_tokenize(text)
```

How?

Using Natural Language Toolkit (`nltk`)

- Lowercase:

```
text = text.lower()
```

- Remove stop words:

```
swords = stopwords.words('english')
words  = wordpunct_tokenize(text)
words  = [w for w in words if w not in swords]
```

How?

Using Natural Language Toolkit (`nltk`)

- Lowercase:

```
text = text.lower()
```

- Remove stop words:

```
swords = stopwords.words('english')
words  = wordpunct_tokenize(text)
words  = [w for w in words if w not in swords]
text = ' '.join(words)
```

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
- Remove stop words:
  swords = stopwords.words('english')
  words  = wordpunct_tokenize(text)
  words  = [w for w in words if w not in swords]
  text = ' '.join(words)
- Stemming:
```

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
- Remove stop words:
  swords = stopwords.words('english')
  words  = wordpunct_tokenize(text)
  words  = [w for w in words if w not in swords]
  text = ' '.join(words)
- Stemming:
  st = EnglishStemmer()
```

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
- Remove stop words:
  swords = stopwords.words('english')
  words  = wordpunct_tokenize(text)
  words  = [w for w in words if w not in swords]
  text = ' '.join(words)
- Stemming:
  st = EnglishStemmer()
  words = wordpunct_tokenize(text)
```

How?

Using Natural Language Toolkit (`nltk`)

- Lowercase:

```
text = text.lower()
```

- Remove stop words:

```
swords = stopwords.words('english')
words  = wordpunct_tokenize(text)
words  = [w for w in words if w not in swords]
text = ' '.join(words)
```

- Stemming:

```
st = EnglishStemmer()
words = wordpunct_tokenize(text)
words = [st.stem(w) for w in words]
```

How?

Using Natural Language Toolkit (`nltk`)

```
- Lowercase:
  text = text.lower()
- Remove stop words:
  swords = stopwords.words('english')
  words  = wordpunct_tokenize(text)
  words  = [w for w in words if w not in swords]
  text = ' '.join(words)
- Stemming:
  st = EnglishStemmer()
  words = wordpunct_tokenize(text)
  words = [st.stem(w) for w in words]
  text = ' '.join(words)
```

To Matrices

To Matrices

- n-grams

```
from nltk import bigrams, trigrams, ngrams
text = word tokenize(text)
text_bi  = bigrams(text)
```