Big Data tools

Introduction to Regular Expression

by Leotis Buchanan(LeotisBuchanan@gmail.com)

12 May 2015

Motivation

Consider these problems:

1. **Searching problem**

   Find the first 3 twitter handles mentioned in a tweet in the twitter data.

2. **Data cleaning** :

   Remove all records from a petabyte of data that have invalid zip codes

By the end of this presentation you will know how to solve these problems.

What are Regular expressions?

Regular expressions allows you to formally describe a set of strings based on common characteristics shared by each string in the set.

what are strings?

From the computer perspective a string is any series of characters that are contained within single or double quotes:

examples:

1. "123"
2. "My name is Leotis"
3. 'https://ryerson.ca'

Example of reqular expressions

1. Regular expression for matching and email

   ```
   /^([a-z0-9_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/
   ```

   This pattern will match : **myname@ryerson.ca** and any other string having the same pattern.

2. Regular expression for matching a url

   ```
   /^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/
   ```

   This will match: http://my.ryerson.ca

Basic Building blocks of Regualar Expressions

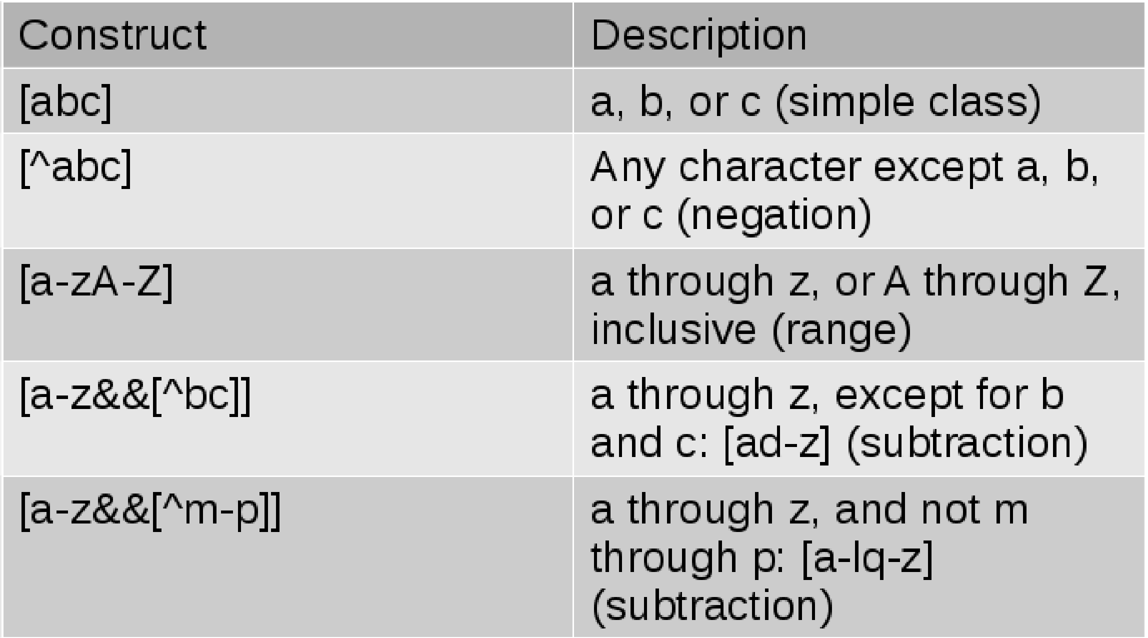
String Literals

The simpliest regular expression you can write, is the actual string you want to match. So if given a sentence "Big data tools are great", the regular expression **'data'** will match the word **data** exactly.

Character Classes

Character classes allow us to create regular expressions that match single characters. Later we will combine them to match a group of characters(a.k.a a string)

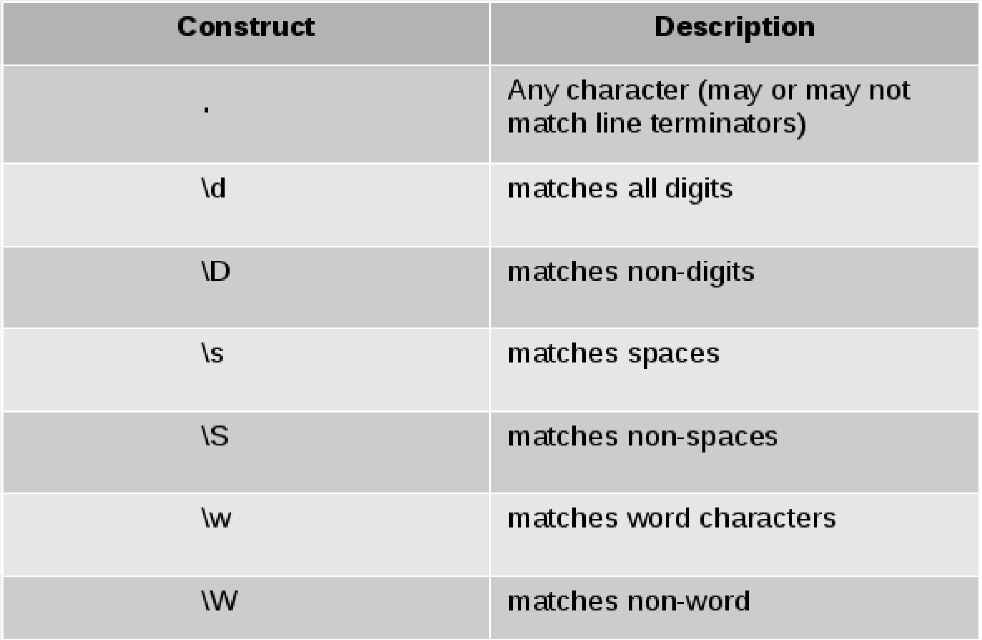
The table below shows the standard character classes.



Construct	Description
[abc]	a, b, or c (simple class)
[^abc]	Any character except a, b, or c (negation)
[a-zA-Z]	a through z, or A through Z, inclusive (range)
[a-z&&[^bc]]	a through z, except for b and c: [ad-z] (subtraction)
[a-z&&[^m-p]]	a through z, and not m through p: [a-lq-z] (subtraction)

Predefined Character Classes

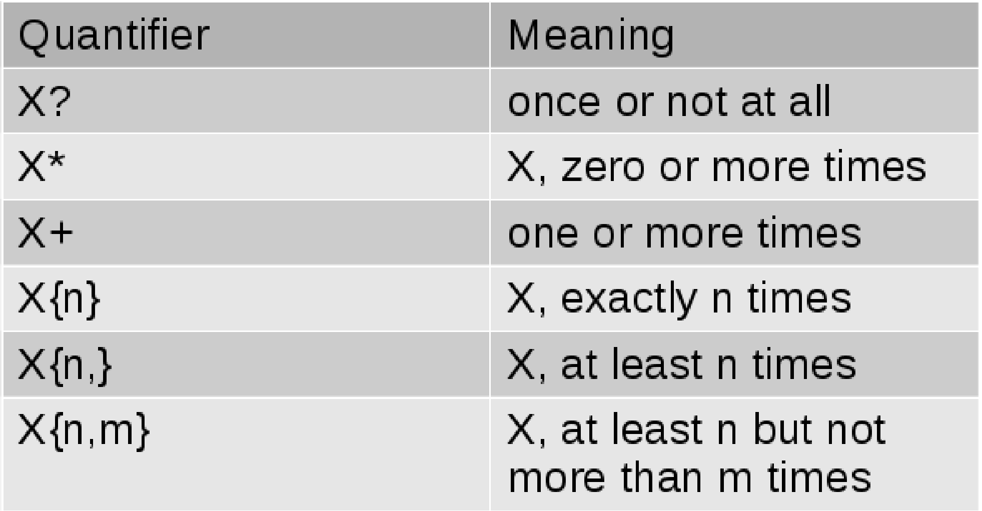
These are shorthand for commonly used regualar expressions.



Construct	Description
.	Any character (may or may not match line terminators)
\d	matches all digits
\D	matches non-digits
\s	matches spaces
\S	matches non-spaces
\w	matches word characters
\W	matches non-word

Quantifiers

Quantifiers allow us to specify the number of occurrences to match against.



Quantifier	Meaning
X?	once or not at all
X*	X, zero or more times
X+	one or more times
X{n}	X, exactly n times
X{n,}	X, at least n times
X{n,m}	X, at least n but not more than m times

for example a+ will match the following character sequence a , aaa, aaaaa etc

Capturing Groups and Character Classes with Quantifiers

It is often required to capture groups of characters, for instance:

647-879-3456

Quantfiers will only allow you to capture a single character.

To specify that you want to capture a group , you surround the characters with **()** i.e

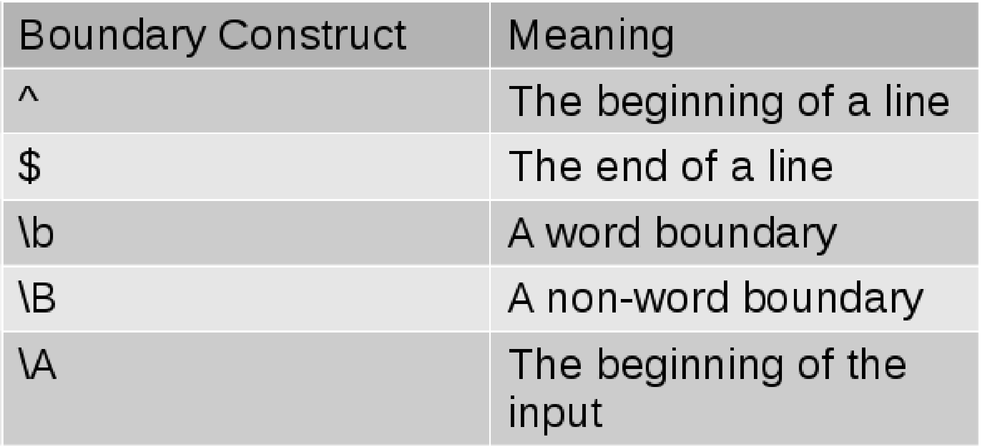
(647-879-3456)

This will now return the match 647-879-3456

what will **(647-879-3456){3}** match?

Boundary Matchers

It is sometimes required to control where match should occur. You can specify this by using boundary matchers. The following tables shows a list of all the applicable boundary matcher constructs.



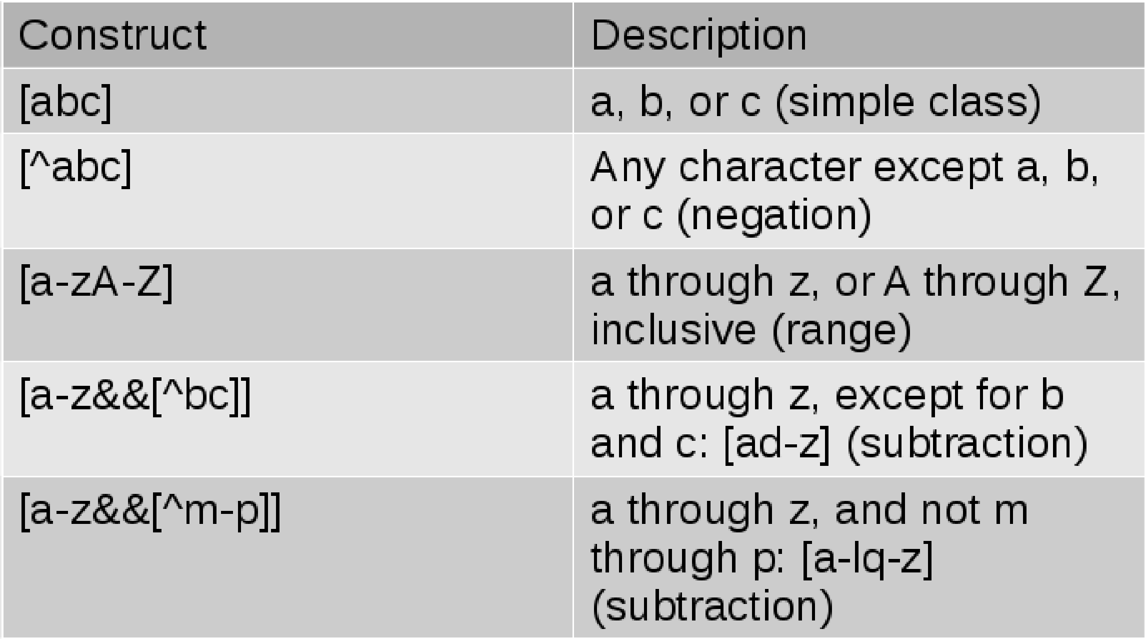
Boundary Construct	Meaning
^	The beginning of a line
$	The end of a line
\b	A word boundary
\B	A non-word boundary
\A	The beginning of the input

Example : The following literal regex will be matched only at the the start of the line:

^Big

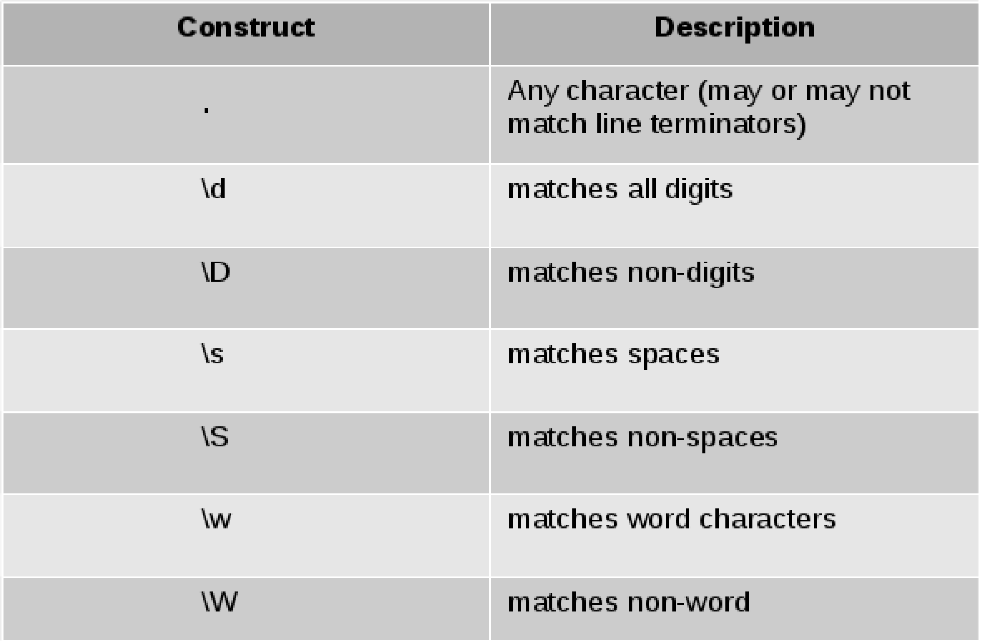
Cheat sheet

Character Classes



Construct	Description
[abc]	a, b, or c (simple class)
[^abc]	Any character except a, b, or c (negation)
[a-zA-Z]	a through z, or A through Z, inclusive (range)
[a-z&&[^bc]]	a through z, except for b and c: [ad-z] (subtraction)
[a-z&&[^m-p]]	a through z, and not m through p: [a-lq-z] (subtraction)

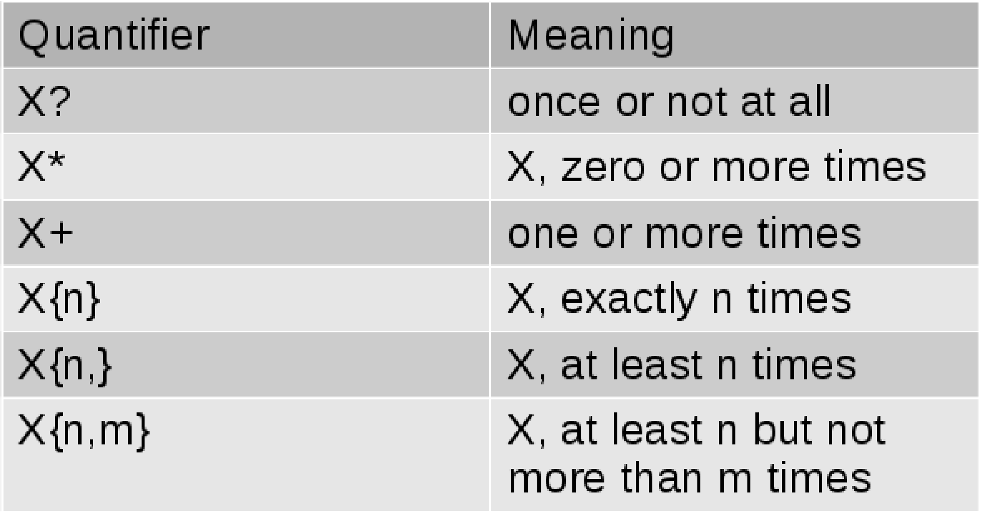
Predefined Character Classes



Construct	Description
.	Any character (may or may not match line terminators)
\d	matches all digits
\D	matches non-digits
\s	matches spaces
\S	matches non-spaces
\w	matches word characters
\W	matches non-word

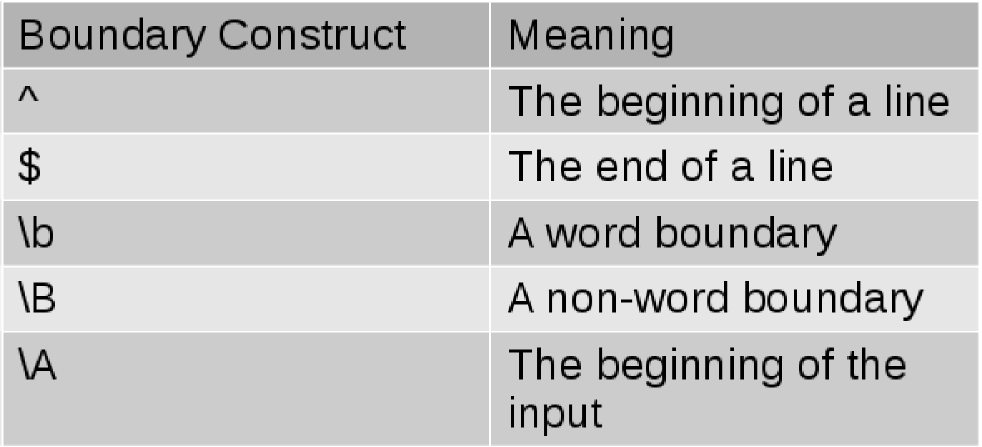
Cheat sheet

Quantifiers



Quantifier	Meaning
X?	once or not at all
X*	X, zero or more times
X+	one or more times
X{n}	X, exactly n times
X{n,}	X, at least n times
X{n,m}	X, at least n but not more than m times

Boundary Matchers



Boundary Construct	Meaning
^	The beginning of a line
$	The end of a line
\b	A word boundary
\B	A non-word boundary
\A	The beginning of the input

Now lets practice

1. Write a regular expression that matches all valid Canadian Postal code.
2. In your own words explain what each components of the following regex do:

   ```
   /^(https?:\/\/)?([\da-z\.-]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/
   ```