Porter Stemmer

THE ALGORITHM

- A *consonant* in a word is a letter other than A, E, I, O or U.
- If a letter is not a consonant it is a *vowel*.

These may all be represented by the single form
[C]VCVC ... [V]

where the square brackets denote arbitrary presence of their contents.

Using (VC)m to denote VC repeated m times, this may again be written as
$[C](VC)^{m}[V]$.

- m will be called the *measure* of any word or word part when represented in this form. The case m = 0 covers the null word. Here are some examples:

- m=0 TR, EE, TREE, Y, BY.
- m=1 TROUBLE, OATS, TREES, IVY.
- m=2 TROUBLES, PRIVATE, OATEN, ORRERY.

- The rules for removing a suffix will be given in the form

(condition) S1 -> S2

This means that if a word ends with the suffix S1, and the stem before S1 satisfies the given condition, S1 is replaced by S2. The condition is usually given in terms of m, e.g.

(m > 1) EMENT ->

Here S1 is ‘EMENT’ and S2 is null. This would map REPLACEMENT to REPLAC, since REPLAC is a word part for which m = 2.

- The ‘condition’ part may also contain the following:
- *S -the stem ends with S (and similarly for the other letters).
- *v* -the stem contains a vowel.
- *d -the stem ends with a double consonant (e.g. -TT, -SS).
- *o -the stem ends cvc, where the second c is not W, X or Y (e.g. -WIL, -HOP).

- In a set of rules written beneath each other, only one is obeyed, and this will be the one with the longest matching S1 for the given word. For example, with

SSES	->	SS
IES	->	I
SS	->	SS
S	->	

here the conditions are all null) CARESSES maps to CARESS since SSES is the longest match for S1. Equally CARESS maps to CARESS (S1=‘SS’) and CARES to CARE (S1=‘S’).

- Step 1a

SSES	->	SS	caresses	->	caress
IES	->	I	ponies	->	poni
			ties	->	ti
SS	->	SS	caress	->	caress
S	->		cats	->	cat

Step 1b

(m>0) EED	->	EE	feed	->	feed
			agreed	->	agree
(*v*) ED	->		plastered	->	plaster
			bled	->	bled
(*v*) ING	->		motoring	->	motor
			sing	->	sing

If the second or third of the rules in Step 1b is successful, the following is done:

T	->	ATE	conflat(ed)	->	conflate
BL	->	BLE	troubl(ed)	->	trouble
IZ	->	IZE	siz(ed)	->	size
(*d and not (*L or *S or *Z))	->	single letter	hopp(ing)	->	hop
			tann(ed)	->	tan
			fall(ing)	->	fall
			hiss(ing)	->	hiss
			fizz(ed)	->	fizz
(m=1 and *o)	->	E	fail(ing)	->	fail
			fil(ing)	->	file

- Step 1: deals with plurals and past participles. The subsequent steps are much more straightforward.
- Step : 2 PTO

(m>0) ATIONAL	->	ATE	relational	->	relate
(m>0) TIONAL	->	TION	conditional	->	condition
			rational	->	rational
(m>0) ENCI	->	ENCE	valenci	->	valence
(m>0) ANCI	->	ANCE	hesitanci	->	hesitance
(m>0) IZER	->	IZE	digitizer	->	digitize
(m>0) ABLI	->	ABLE	conformabli	->	conformable
(m>0) ALLI	->	AL	radicalli	->	radical
(m>0) ENTLI	->	ENT	differentli	->	different
(m>0) ELI	->	E	vileli	->	vile
(m>0) OUSLI	->	OUS	analogousli	->	analogous
(m>0) IZATION	->	IZE	vietnamization	->	vietnamize
(m>0) ATION	->	ATE	predication	->	predicate
(m>0) ATOR	->	ATE	operator	->	operate
(m>0) ALISM	->	AL	feudalism	->	feudal
(m>0) IVENESS	->	IVE	decisiveness	->	decisive
(m>0) FULNESS	->	FUL	hopefulness	->	hopeful
(m>0) OUSNESS	->	OUS	callousness	->	callous
(m>0) ALITI	->	AL	formaliti	->	formal
(m>0) IVITI	->	IVE	sensitiviti	->	sensitive
(m>0) BILITI	->	BLE	sensibiliti	->	sensible

Step 3

m>0) ICATE	->	IC	triplicate	->	triplic
(m>0) ATIVE	->		formative	->	form
(m>0) ALIZE	->	AL	formalize	->	formal
(m>0) ICITI	->	IC	electriciti	->	electric
(m>0) ICAL	->	IC	electrical	->	electric
(m>0) FUL	->		hopeful	->	hope
(m>0) NESS	->		goodness	->	good

Step 4

(m>1) AL	->	revival	->	reviv
(m>1) ANCE	->	allowance	->	allow
(m>1) ENCE	->	inference	->	infer
(m>1) ER	->	airliner	->	airlin
(m>1) IC	->	gyroscopic	->	gyroscop
(m>1) ABLE	->	adjustable	->	adjust
(m>1) IBLE	->	defensible	->	defens
(m>1) ANT	->	irritant	->	irrit
(m>1) EMENT	->	replacement	->	replac
(m>1) MENT	->	adjustment	->	adjust
(m>1) ENT	->	dependent	->	depend
(m>1 and (*S or *T)) ION	->	adoption	->	adopt
(m>1) OU	->	homologou	->	homolog
(m>1) ISM	->	communism	->	commun
(m>1) ATE	->	activate	->	activ
(m>1) ITI	->	angulariti	->	angular

Step 5a

(m>1) E	->		probate	->	probat
			rate	->	rate
(m=1 and not *o) E	->		cease	->	ceas