Lab 0 – EDAN20

The first lab of the course language technology (EDAN20) aims at understanding some well evaluated code given by the lecturer. The code formed a spell corrector and this report will focus at analysing each part of the script, which is containing different methods.

def words(text):

return re.findall(r'\w+', text.lower())

The first part of the code creates a list containing all words in the input string.

WORDS = Counter(words(open('big.txt').read()))

WORDS is a counter object containing the number of times each word in a chosen text has been used:

Counter({'the': 79809, 'of': 40024, 'and': 38312, 'to': 28765, 'in': 22023, 'a': 21124, ….

def P(word, N=sum(WORDS.values())):  
 *"Probability of `word`."* return WORDS[word] / N

The definition P above will calculate the probability that a specific word is used by dividing the amount of the times ‘word’ has been used with the sum of all words used.

def correction(word):  
 *"Most probable spelling correction for word."* return max(candidates(word), key=P)  
  
  
def candidates(word):  
 *"Generate possible spelling corrections for word."* return (known([word]) or known(edits1(word)) or known(edits2(word)) or [word])  
  
  
def known(words):  
 *"The subset of `words` that appear in the dictionary of WORDS."* return set(w for w in words if w in WORDS)

The three methods correction, candidates and known will together make sure that the given word is corrected correctly. It does that by first choosing the word that did not need any edit, secondly the words that are only one edit away from the given word or thirdly the words requiring two edits. These lists are created by the methods below. One specific word from one of these lists is then selected – based on which word got highest value from def P.

def known(words):  
 *"The subset of `words` that appear in the dictionary of WORDS."* return set(w for w in words if w in WORDS)  
  
  
def edits1(word):  
 *"All edits that are one edit away from `word`."* letters = 'abcdefghijklmnopqrstuvwxyz'  
 splits = [(word[:i], word[i:]) for i in range(len(word) + 1)]  
 deletes = [L + R[1:] for L, R in splits if R]  
 transposes = [L + R[1] + R[0] + R[2:] for L, R in splits if len(R) > 1]  
 replaces = [L + c + R[1:] for L, R in splits if R for c in letters]  
 inserts = [L + c + R for L, R in splits for c in letters]  
 return set(deletes + transposes + replaces + inserts)  
  
  
def edits2(word):  
 *"All edits that are two edits away from `word`."* return (e2 for e1 in edits1(word) for e2 in edits1(e1))