C2_W1_Assignment

October 30, 2020

1 Assignment 1: Auto Correct

Welcome to the first assignment of Course 2. This assignment will give you a chance to brush up on your python and probability skills. In doing so, you will implement an auto-correct system that is very effective and useful.

1.1 Outline

- Section **??**
  - Section **??**
- Section **??**
  - Section **??**
  - Section **??**
  - Section **??**
- Section **??**
  - Section **??**
  - Section **??**
  - Section **??**
  - Section **??**
- Section **??**
  - Section **??**
  - Section **??**
  - Section **??**
- Section **??**
  - Section **??**
- Section **??**

0. Overview

You use autocorrect every day on your cell phone and computer. In this assignment, you will explore what really goes on behind the scenes. Of course, the model you are about to implement is not identical to the one used in your phone, but it is still quite good.

By completing this assignment you will learn how to:

- Get a word count given a corpus
- Get a word probability in the corpus
- Manipulate strings
- Filter strings
- Implement Minimum edit distance to compare strings and to help find the optimal path for the edits.
- Understand how dynamic programming works

Similar systems are used everywhere. - For example, if you type in the word **"I am lerningg"**, chances are very high that you meant to write **"learning"**, as shown in **Figure 1**.

Figure 1

0.1 Edit Distance

In this assignment, you will implement models that correct words that are 1 and 2 edit distances away. - We say two words are n edit distance away from each other when we need n edits to change one word into another.

An edit could consist of one of the following options:

- Delete (remove a letter): 'hat' => 'at, ha, ht'
- Switch (swap 2 adjacent letters): 'eta' => 'eat, tea,...'
- Replace (change 1 letter to another): 'jat' => 'hat, rat, cat, mat, ...'
- Insert (add a letter): 'te' => 'the, ten, ate, ...'

You will be using the four methods above to implement an Auto-correct. - To do so, you will need to compute probabilities that a certain word is correct given an input.

This auto-correct you are about to implement was first created by Peter Norvig in 2007. - His original article may be a useful reference for this assignment.

The goal of our spell check model is to compute the following probability:

$$P(c|w) = \frac{P(w|c) \times P(c)}{P(w)} \tag{Eqn-1}$$

The equation above is Bayes Rule. - Equation 1 says that the probability of a word being correct $P(c|w) $is equal to the probability of having a certain word w, given that it is correct $P(w|c)$, multiplied by the probability of being correct in general $P(C)$ divided by the probability of that word w appearing $P(w)$ in general. - To compute equation 1, you will first import a data set and then create all the probabilities that you need using that data set.

Part 1: Data Preprocessing

```
In [ ]: import re
        from collections import Counter
        import numpy as np
        import pandas as pd
```

As in any other machine learning task, the first thing you have to do is process your data set. - Many courses load in pre-processed data for you. - However, in the real world, when you build these NLP systems, you load the datasets and process them. - So let's get some real world practice in pre-processing the data!

Your first task is to read in a file called **'shakespeare.txt'** which is found in your file directory. To look at this file you can go to `File ==> Open`.

Exercise 1 Implement the function `process_data` which