**HW #1 REGEX and NORMALIZATION**

**Name: Rudraksh Sugandhi**

**Observation Criteria:**

**Regular Expression Part:**

1. For the file to access the first thing that should be considered to open file in read mode.
2. Lowercase all the input from file so don’t need to add extra regex function expression for upper as well as lower case.
3. For the substitution function there python replace function available but according to problem statement, used the re.sub function.
4. There is multiple expression that came to thought process described in table below:

|  |  |  |
| --- | --- | --- |
| Expression | Explanation | Passed/ Failed test Case |
| '\b\w{1,}our{1,}\b': | word starting with any 1 or more than one character with our in between and end with 1 or more than 1 character | Failed to capture  Neighbour |
| \b\w\*our\w\*\b | Amy word in starting before our and after that | Failed to capture savour |
| \b\w{6,}.\*our.\*\b' | String starting with 6 more character and our in between | Failed |
| (?<!y)our | Negation with which doesn’t contain y as first character | Failed with test case like  “our” |
| **[a-z][a-z]+our** | **Two or more Character in starting followed by our** | **Passed** |
| **([a-z]{1,}(?<!y|h|f|s|c))our** | **One or more Character which not include y,h,f,s andc as starting Character and our at end whole combined in group** | **Passed** |

The Test case word are:

|  |  |  |  |
| --- | --- | --- | --- |
| arbour  ardour  armour  behaviour  British  candour  clamour  colour  demeanour  endeavour | favour  flavour  harbour  honour  humour  labour  neighbour  odour  parlour  our | your  yourself  traour  source  course  hour  four  yourself  fourth  hours | rancour  rigour  rumour  saviour  savour  splendour  tumour  valour  vigour  your |

1. A ‘r’ is used at start of expression as python raw string notation for expression to handle or consider backslashes.
2. For Doctor miss and other have created dictionary to replace.
3. Saving the file into regex.txt file using write mode to write in the file.
4. Calling the Normalize function from same function as we don’t need to call both function again.

**Normalization Part:**

1. After getting the regex file in read mode the first part is lower case.

2. Removing the number as this can affect the dictionary to create later with replacing d+ with nothing.

3. Selecting words or removing the Punctuation like semicolon, double inverted code and other type of exclamation mark as well as hyphen. This will be not useful for parsing the model to train on. As the model only train on words will have no effect of the punctuations.

4. Removing stop words like most repeated words in our dictionary as those are not usefull like “the”,’a’ ‘are’ and etc. This stop word consumes more storage and does not have any impact or increasing in accuracy of model.

5. There are some words in dictionary which contains “\_” in the word before and after which is of no use removed those signs by traversing each word in dictionary.

6. Finally Sorting the dictionary the dictionary in alphabetical order within built sort function

7. Storing the function in the file dictionary.txt.