**EXPLORATORY DATA ANALYSIS**

CIS 678 Machine Learning

Programming Project #1

BY:

Sushmitha Chinna Chenna Reddy Gare

**OBJECTIVE AND INTRODUCTION:**

The Oxford dictionary defines the word “readable”, from which the word “readability” derives, as:

1. Able to be read deciphered; legible: a code, which is readable by a computer readable copy of very old newspapers

(a) Easy or enjoyable to read: a marvelously readable book

For our purposes, readability can be defined as the ease in which a text can be read and understood.

The most common factors in these existing algorithms to calculate the readability are:

• The total amount of words

• The length of sentences

• The amount of words defined as complicated

• The amount syllables, etc . . .

The main task of this project is to find Flesch Index for different text files. This Formula is a simple approach to assess the complexity level of understanding of the reader. It has become a standard readability formula used by many US Government Agencies, including the US Department of Defense. However, primarily, we use the formula to assess the difficulty of a reading passage written in English. If the text is too complicated and hard to understand, people will leave the page and return to the search results. On the other hand, if the text scores a very low Felsch index, users will likely assume that your content isn’t valuable. Both of these things can prevent site visitors from interacting with your content and contribute to a high bounce-rate.

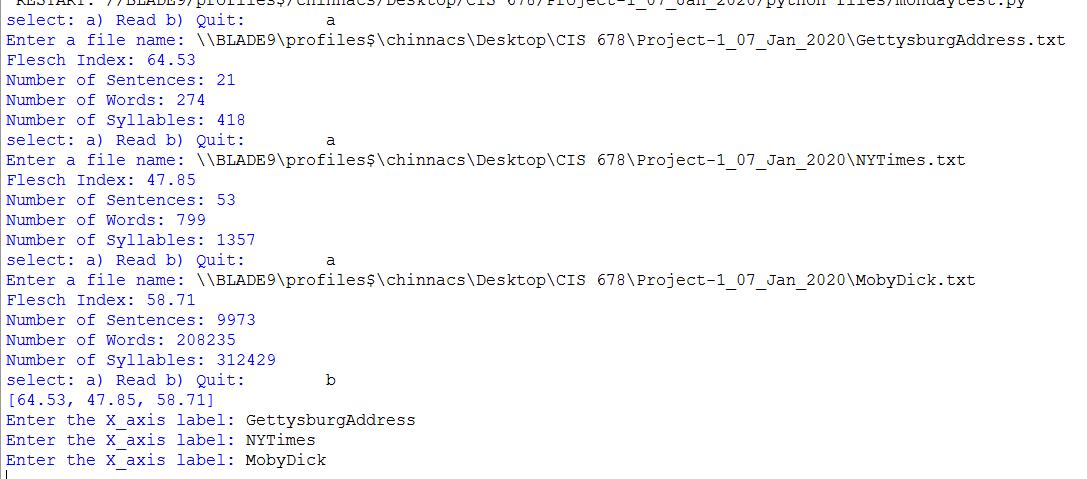
**PROGRAM DEMONSTRATION:**

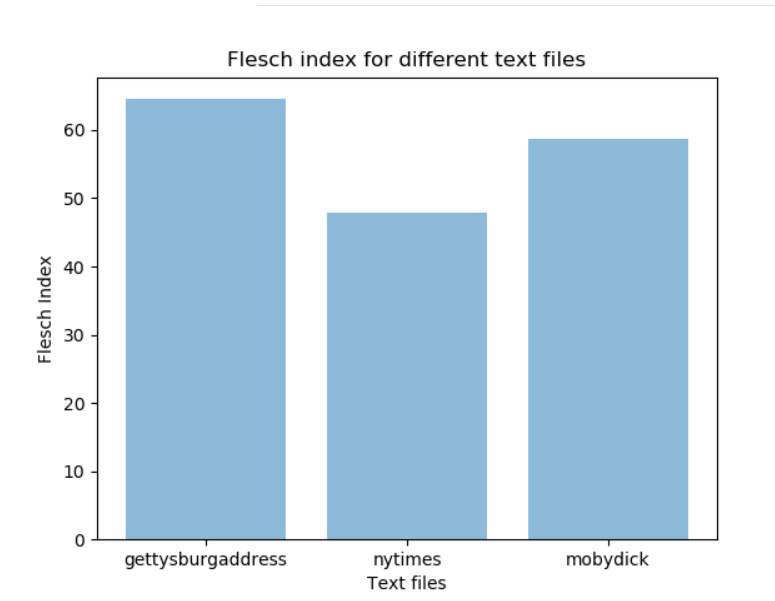
1. Python was the Programming platform of choice used.
2. Initially we have imported data from the external text file, in order to make it more interactive for the user; we have created a dynamic input function to enter for the choice of user file. If suppose the user want to compare between files (Session output are stored in lists), we have included that facility to do so.
3. Followed by processing the data from file as the Flesch Index Formula has certain ambiguities. For instance, punctuation, non-alphabetic characters, colons and semicolons serve as hurdles for the program for calculate certain text characteristics. For these we have employed String library.
4. We have written functions for calculating the Number of syllables, Number of Words, Number of sentences that are the major inputs for calculating the Flesch index and the separate function for Flesch index was written too.
5. For the visualization of the results, matplotlib and numpy were employed.

**ANALYSIS:**

Three text files I.e. Gettysburg Address, NY Times, Moby Dick were imported, Flesch index was calculated for these, and the same was plotted on graph.

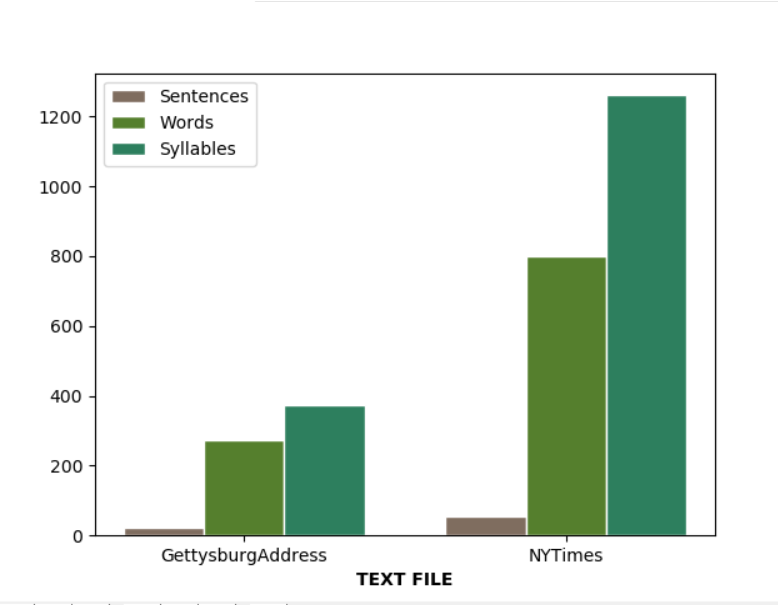
As we can see from the below results and graph the Flesch index for NY Times is lowest and Gettysburg Address text is the highest, in comparison among the three text files.





Target readability score will depend on your intended audience and on the content that is delivered. For example, if we are providing information geared toward elementary school children, we want to keep our content at a lower reading level. On the other hand, if we are writing a university-level piece that involves complicated research and a deep understanding of a particular content area, you can allow your readability score to decrease substantially. Choice of Words, length of sentences directly affect the Flesch index score.

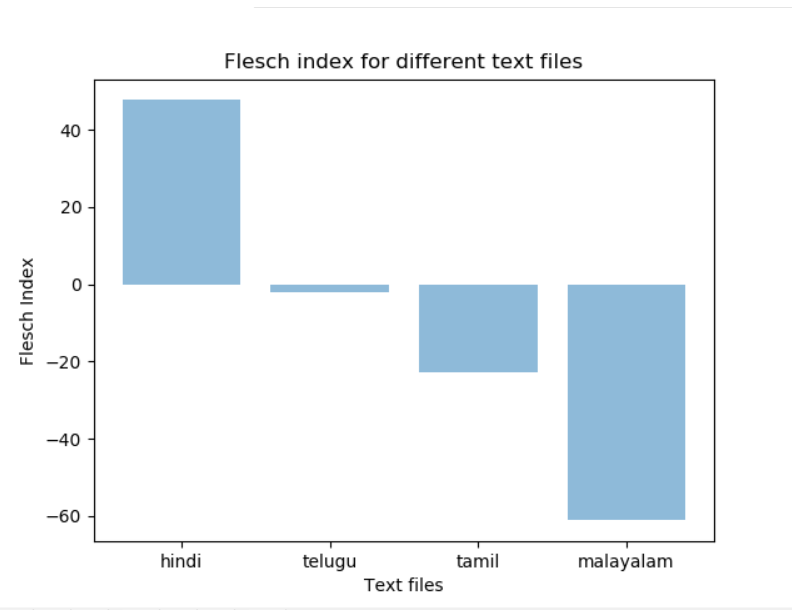
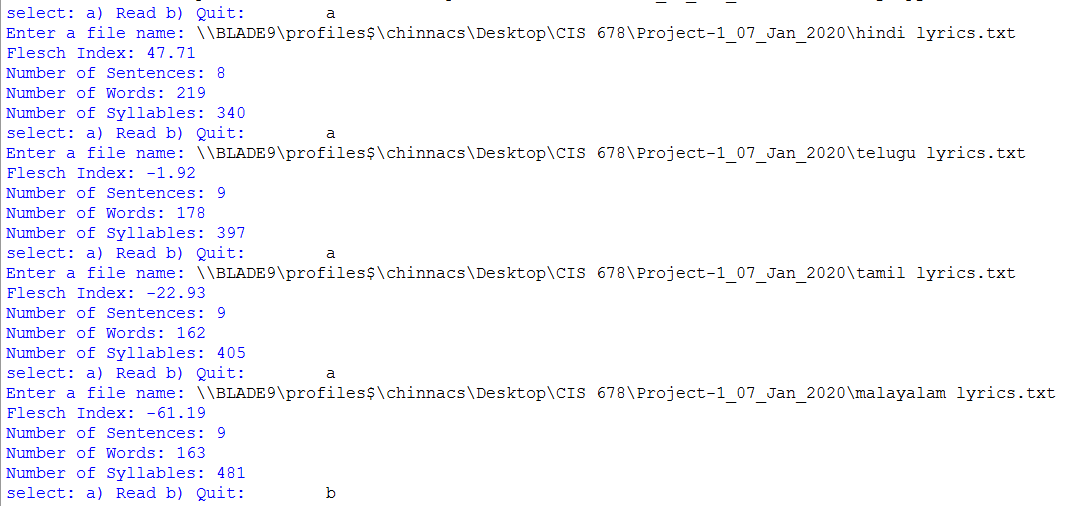
Below is the bar plot of the number of Words, sentences and syllables in Gettysburg and NY times Text, We could not include MobyDick in this because of high values of text characteristics.

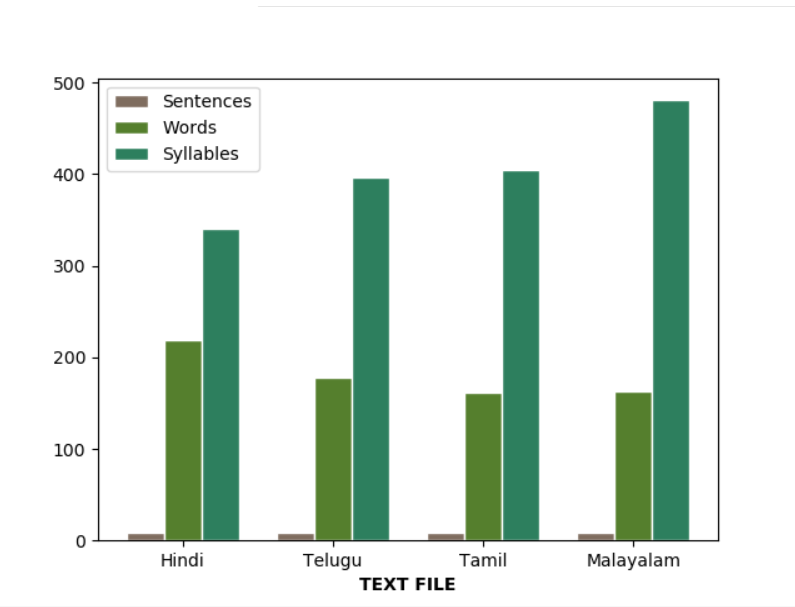


**STATE OF PROBLEM:**

The question examined in this report is how readability algorithm performs when processing texts written in different, not too distantly related, Asian languages. The languages Hindi, Telugu, Tamil and Malayalam were chosen due to their resemblance. Exception Hindi is not related to the other three.

We have chosen the Poetry by a poet who have written the exact same in all the four languages. Below is the Flesch Index comparison graph and other graph, which compares the text characteristics.





**DISCUSSION:**

On calculation of Flesch index for the non-English languages have given a negative Flesch index result for some the languages. We can see the number of sentences were low and we can see that Malayalam language has highest number syllables. Many factors affect the readability score in this we feel the less number of sentences has actually caused the Flesch index to get negative score. Malayalam language having highest number of syllables indicates the complexity of the language and there are some studies, which tell us that the vowels in a language tend to, twist tongue while we speak, which makes this language more complex comparatively with other languages.

**CONCLUSION:**

Evaluating readability will always come with problems, as it is very hard to find a scale, which cover all the existing aspects of it. Having problems collecting suitable data to evaluate will pose a problem to get a trustworthy result.

References:

Code references for graphs: <https://matplotlib.org/>

Content references:

“Speech-Specific Tuning of Neurons in Human Superior Temporal Gyrus”, [Alexander M. Chan](https://www.ncbi.nlm.nih.gov/pubmed/?term=Chan%20AM%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Andrew R. Dykstra](https://www.ncbi.nlm.nih.gov/pubmed/?term=Dykstra%20AR%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Vinay Jayaram](https://www.ncbi.nlm.nih.gov/pubmed/?term=Jayaram%20V%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Matthew K. Leonard](https://www.ncbi.nlm.nih.gov/pubmed/?term=Leonard%20MK%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Katherine E. Travis](https://www.ncbi.nlm.nih.gov/pubmed/?term=Travis%20KE%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Brian Gygi](https://www.ncbi.nlm.nih.gov/pubmed/?term=Gygi%20B%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Janet M. Baker](https://www.ncbi.nlm.nih.gov/pubmed/?term=Baker%20JM%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Emad Eskandar](https://www.ncbi.nlm.nih.gov/pubmed/?term=Eskandar%20E%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Leigh R. Hochberg](https://www.ncbi.nlm.nih.gov/pubmed/?term=Hochberg%20LR%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), [Eric Halgren](https://www.ncbi.nlm.nih.gov/pubmed/?term=Halgren%20E%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), and [Sydney S. Cash](https://www.ncbi.nlm.nih.gov/pubmed/?term=Cash%20SS%5BAuthor%5D&cauthor=true&cauthor_uid=23680841), 2014 Oct; 24(10): 2679–2693.

**CODE:**

import string

import matplotlib.pyplot as plt; plt.rcdefaults()

import matplotlib.pyplot as plt

import numpy as np

## Function to count the Number of sentences

def count\_sentences(text):

# Getting Captial and Small Alphabets through ASCII

y=string.ascii\_uppercase+string.ascii\_lowercase

# conversion String to List

list1=list(y)

# conversion String to List

txt\_list=list(text)

numSentence=0

for i in range (len(txt\_list)):

if (txt\_list[i]=='.' or txt\_list[i]=='?' or txt\_list[i]=='!' ) and (txt\_list[i-1] in list1):

numSentence+=1

return (numSentence)

## Function to count the Number of Words

def count\_words(text):

# Getting Captial and Small Alphabets through ASCII

y=string.ascii\_uppercase+string.ascii\_lowercase

# conversion String to List

list1=list(y)

# conversion String to List

special\_char=list('.?!')

# conversion String to List

txt\_list=list(text)

numWords=0

for i in range (len(txt\_list)):

if (txt\_list[i]==' ') and (txt\_list[i-1] in list1):

numWords+=1

elif (txt\_list[i]==' ') and (txt\_list[i-1] in special\_char)and (txt\_list[i-2] in list1):

numWords+=1

return (numWords)

## Function to count the Number of Syllables

def count\_Syllables(text):

# Getting Captial and Small Alphabets through ASCII

y=string.ascii\_uppercase+string.ascii\_lowercase

# conversion String to Sets

set1=set(y)

# conversion Vowel String to Sets

set2=set('aeiouAEIOU')

# conversion Vowel String to List

vowels=list('aeiouAEIOU')

consonants=list(set1-set2)

special\_char=list("!' '?.")

# conversion String to List

txt\_list=list(text)

count=0

numSyllables=0

for i in range (len(txt\_list)):

if (count==0)and (txt\_list[i] in vowels):

numSyllables+=1

count+=1

elif ((count>0) and (txt\_list[i] in vowels) and (len(txt\_list)>i+1) and (txt\_list[i-1] in consonants)):

numSyllables+=1

count+=1

if (txt\_list[i]=='e' or txt\_list[i]=='E')and (len(txt\_list)>i+1) and (txt\_list[i+1] in special\_char)and (txt\_list[i-1] in consonants):

count-=1

count+=1

if txt\_list[i]==' ':

count=0

return (numSyllables)

## Function to calculate flesch index

def flesch\_index (numSyllables,numWords,numSentences):

x=round(float(numSyllables/numWords),4)

y=round(float(numWords/numSentences),4)

flesch\_idx= ((206.835)-(84.6\*x)-(1.015\*y))

return (round(flesch\_idx,2))

def plotting(y\_axis):

x\_axis=[]

for i in range (len(y\_axis)):

x=str(input('Enter the X\_axis label:\t' )).lower()

x\_axis.append(x)

y\_pos = np.arange(len(x\_axis))

plt.bar(y\_pos,y\_axis,align='center', alpha=0.5)

plt.xticks(y\_pos, x\_axis)

plt.xlabel('Text files')

plt.ylabel('Flesch Index')

plt.title('Flesch index for different text files')

plt.show()

def plotting2(sentences, words, syllables):

# set width of bar

barWidth = 0.25

# set height of bar

bars1 = sentences

bars2 = words

bars3 = syllables

# Set position of bar on X axis

r1 = np.arange(len(bars1))

r2 = [x + barWidth for x in r1]

r3 = [x + barWidth for x in r2]

# Make the plot

plt.bar(r1, bars1, color='#7f6d5f', width=barWidth, edgecolor='white', label='Sentences')

plt.bar(r2, bars2, color='#557f2d', width=barWidth, edgecolor='white', label='Words')

plt.bar(r3, bars3, color='#2d7f5e', width=barWidth, edgecolor='white', label='Syllables')

# Add xticks on the middle of the group bars

plt.xlabel('TEXT FILE', fontweight='bold')

plt.xticks([r + barWidth for r in range(len(bars1))], ['GettysburgAddress', 'NYTimes'])

# Create legend & Show graphic

plt.legend()

plt.show()

def plotting3(sentences, words, syllables):

# set width of bar

barWidth = 0.25

# set height of bar

bars1 = sentences

bars2 = words

bars3 = syllables

# Set position of bar on X axis

r1 = np.arange(len(bars1))

r2 = [x + barWidth for x in r1]

r3 = [x + barWidth for x in r2]

# Make the plot

plt.bar(r1, bars1, color='#7f6d5f', width=barWidth, edgecolor='white', label='Sentences')

plt.bar(r2, bars2, color='#557f2d', width=barWidth, edgecolor='white', label='Words')

plt.bar(r3, bars3, color='#2d7f5e', width=barWidth, edgecolor='white', label='Syllables')

# Add xticks on the middle of the group bars

plt.xlabel('TEXT FILE', fontweight='bold')

plt.xticks([r + barWidth for r in range(len(bars1))], ['Hindi', 'Telugu','Tamil','Malayalam'])

# Create legend & Show graphic

plt.legend()

plt.show()

session\_value\_list=[]

session\_value\_word=[]

session\_value\_sentence=[]

session\_value\_syllable=[]

user\_entry=str(input('select: a) Read b) Quit:\t')).lower()

while(user\_entry!='b'):

# String of characters considered as clean

clean=string.ascii\_uppercase+string.ascii\_lowercase+' '+'.?!-'

#Conversion to list

list1=list(clean)

# Dynamic function to pick any file from the external environment

filename=input('Enter a file name: ')

# Reading the contents of the file

infile=open(filename, encoding='cp437')

data=infile.read()

# Reading all elements in to a list from the file

string\_char=[]

for i in data:

string\_char.append(i)

listdummy=string\_char

# Removal of the tab spaces from list '\n'

listdummy1=[]

for i in range (len(listdummy)):

if listdummy[i]!='\n':

listdummy1.append(listdummy[i])

elif listdummy[i]=='\n':

listdummy1.append(' ')

list2=listdummy1

# Removal of special characters and creating a clean required character list

clean\_string=''

for i in range (len(list2)):

if list2[i]in list1:

clean\_string+=list2[i]

processed\_string=clean\_string

sentences=count\_sentences(processed\_string)

session\_value\_sentence.append(sentences)

words=count\_words(processed\_string)

session\_value\_word.append(words)

syllables=count\_Syllables(processed\_string)

session\_value\_syllable.append(syllables)

g=flesch\_index (syllables,words,sentences)

session\_value\_list.append(g)

print ('Flesch Index: '+str(g))

print ('Number of Sentences: '+str(sentences))

print ('Number of Words: '+str(words))

print ('Number of Syllables: '+str(syllables))

user\_entry=str(input('select: a) Read b) Quit:\t')).lower()

print(session\_value\_list)

plot=plotting(session\_value\_list)

#plot1=plotting2(session\_value\_sentence, session\_value\_word, session\_value\_syllable)

#plot1=plotting3(session\_value\_sentence, session\_value\_word, session\_value\_syllable)