

MA 202 PROBABILITY AND STATISTICS

PROJECT REPORT

THE WORDLE SOLVER

Submitted By:

Ayush Yadav	20110033
Kusum	20110100
Mayukh Reddy	20110110
Nidhi Upasani	20110121
Rahul Raj	20110157

INTRODUCTION

Wordle is a game created and developed by a software engineer, named Josh Wardle. It is the first game of its kind in which the player has to determine the correct word having five letters. The player is given six attempts to determine the word correctly. Each day a new word is to be guessed in the Wordle. Daily Wordle is the same for everyone. If the position of a letter is correct then the box color of that letter will change to green. Moreover, if the five lettered word you have guessed contains the letters in the correct word, but they are not in the right position then the boxes containing those letters will be colored orange. Thus, after your entry the boxes colored green will state that the letters are in the correct position and boxes colored orange will state that the letter is present in the correct word but it is not in the correct position. The boxes containing the letters which are not present in the correct word will change to gray color.

The game was initially created by **Mr. Wardle** for himself and his partner to play. Mr. Wardle then introduced the game to his relatives and saw that they enjoyed it a lot. In October 2021, the game was made public and within two months gained huge popularity. On **January 2, 2022** over 300,000 people played this game and within the next ten days the number increased to **1.2 million**. This game is quite innovative and published for the sole purpose of people's fun. The game has become one of the most favorite games in the world.

PROBLEM STATEMENT

Problem solving is an important skill which every person needs in his/her life to succeed. Becoming good at problem solving not only builds confidence but also makes the journey of solving problems more enjoyable and fun. A strategy is the most important tool in problem solving. It not only opens your mind to various possibilities but also makes you think of the easiest and the most efficient way to solve that problem. A well planned strategy which is a result of data analysis, relevant research outcomes, explicit knowledge, rigorous thought process and some amount of experience, is the key to success in any problem solving statement.

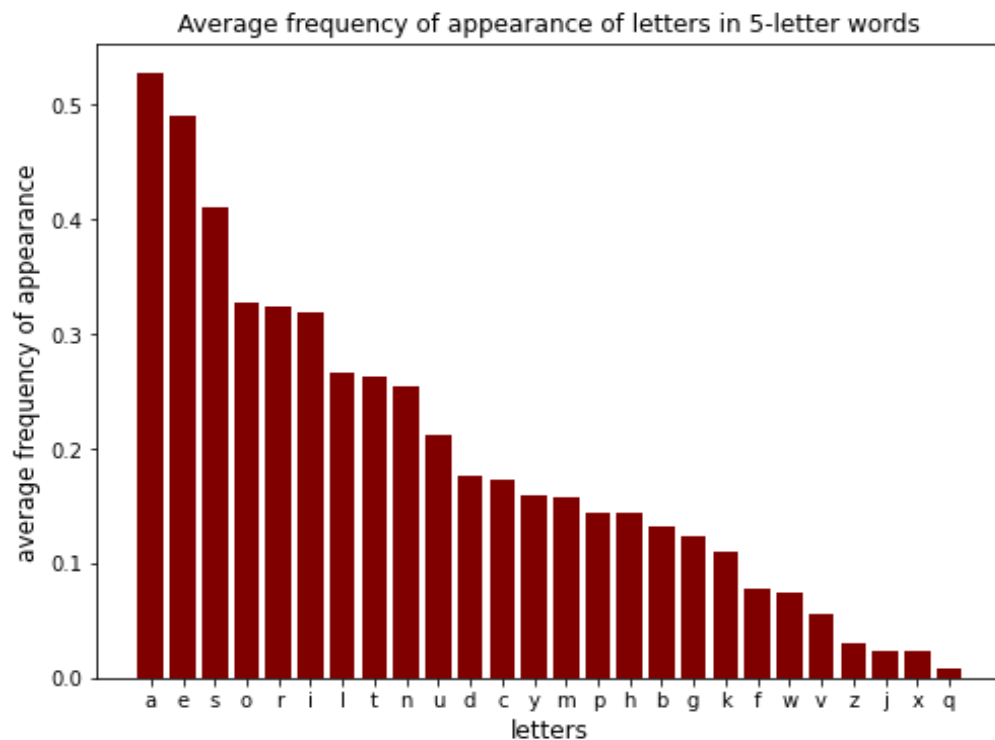
The problem statement for this project is solving the Wordle game with the help of a well thought strategy. We are going to tackle this word problem with the help of a code. The code should have maximum efficiency of being correct in third or fourth guess. We will be using the mathematical topic of probability to solve the problem. Wordle has become a famous daily challenge game and solving it by a code will definitely prove to be very useful. This project serves as a good practice for the understanding of probability.

Some of the concepts that were used while writing this code are listed below:

❖ **Letter frequency -**

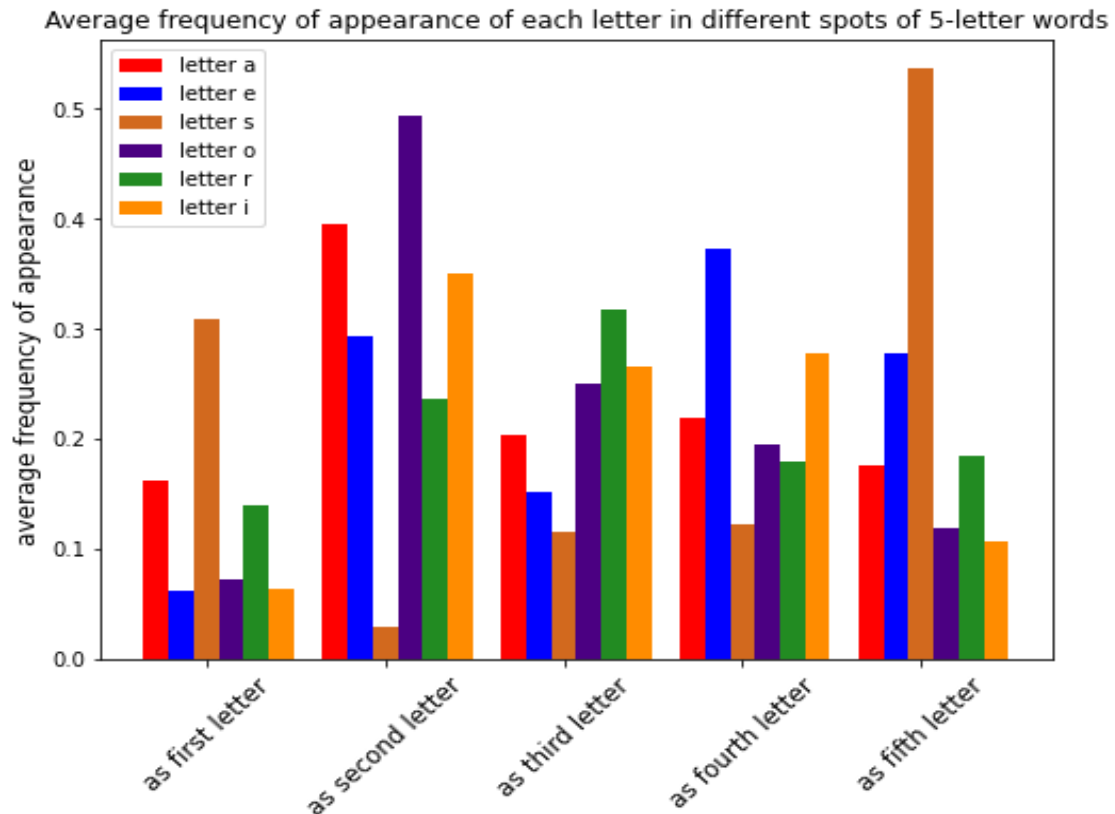
- Letter frequency is the number of times a letter appears in the words of a language. This analysis will help us in forming a strategy to solve the wordle game.

- The letter frequency helps us know which letters are most used and in which places. The frequency of the letters in different places helps to guess a word from a limited number of words as the data array decreases if we find out the correct position of at least one letter.
- Even if we are able to find a letter which exists in the correct word but have not found its correct position, we will have an array of decreased size which will only contain the words having the correct letter in them. In other words our goal is to choose a word which reduces the possible words list to the greatest extent. In this way the size of the array will keep on decreasing in succeeding attempts and we will be able to find the correct word with more efficiency in a lesser number of trials.
- The data of the frequency of a letter with which a five letter word starts is shown below and will come in use while forming a strategy.



<u>Letter</u>	<u>Count</u>	<u>1st Letter</u>	<u>Last Letter</u>
A	1035	182	56
B	292	202	8
C	446	191	29
D	557	113	276
E	1396	70	397
F	228	137	27
G	307	112	41
H	371	75	123
I	686	34	13
J	27	22	0
K	208	19	102
L	697	89	148
M	296	107	37
N	578	40	127
O	758	43	42
P	355	147	48
Q	32	25	0
R	951	108	258
S	705	361	77
T	681	131	232
U	425	25	4
V	154	37	0
W	216	92	18
X	51	0	9
Y	356	5	295
Z	42	3	3

“SLATE”(available in Wordle words Database) → It is the most favorable word possible. Since the frequency of letters “S” , “L”, “A”, “T”, “E” is maximum as shown in the table.



❖ Information Theory :-

- The most important part of information theory is entropy. Entropy is a term which quantifies the amount of uncertainty which is present in the value of a random variable or the end result of some random process
- In the wordle game the goal is to form an algorithm to find a word which gives maximum information about the correct word. In a way it reduces the uncertainty of guessing the correct word. Here we guess the word having the maximum expected value of the gained information. Hence, we are looking for maximizing the entropy.
- In the Wordle game each time we guess there are 35 possible outcomes such as all gray, 4 gray 1 orange, 1 green 1 orange 3 gray, etc. each of these possible outcomes leads to a different list of possible words. So, all we need to do is calculate the expected information on each five-lettered word and write a code to

iteratively compute the expected information for all words and guess the word having maximum entropy, repeating it until we get the correct word.

Strategy Applied in formulation of our code:-

- The algorithm gives a score to each possible word .
- Every letter of the english alphabet has a frequency of occurring at a position i in the five letter word.
- The score given to a position i in a given word is
 - i = character index
 - $\text{maxfreq}(i)$ = maximum possible frequency of a character at the position i
 - $\text{wordfreq}(i)$ = frequency of the character present in the word at the position i
 - $\text{score}[i] = (\text{maxfreq}(i) - \text{wordfreq}(i))$.
- The total score of a word is $\text{score}[1] \times \text{score}[2] \dots \text{score}[5]$.
- The word with the lowest score will be considered the most probable word and will be our guest word in the game .
- Once the suggested word by the algorithm is entered in the game by us and the result is given back by us as input, it checks all the possible words with the given input conditions and suggests another word with the lowest score from the possible words.
- We again enter the suggested word and repeat the process.

Working of our Code

In our code there are 8 functions namely as follows :-

1. **wordleSolver()** :→ This function activates first via function call as shown below.

```
255
256 # calls function named wordleSolver()
257 wordleSolver(possible_words) # function call to initiate the process...!
```

- It receives the database of all 5-letter meaningful words(around 2315) as also present in “Wordle words Database”
- Then it fetches at first the best possible word having letters of high probability of occurrences i.e “SLATE”(as our word opener).
- After getting the result like which position is wrong(gray), partially correct(incorrect position)(yellow), and fully correct letter(correct letter + position)(green).
- It again fetches the best possible collection of words, neglecting those letters which were wrong.
- This process continues till we get the perfect words for winning the Wordle.

2. **Letter Frequency()** :-

- It also receives the words list of 5-letter
- Since, any letter can occupy at 5 different positions i.e at index 0,1,2,3,4.
- So, this function calculates the frequency of all alphabets at these 5 positions.


```

Lenovo@LAPTOP-CSGP8BLS MINGW64 /c/VS.Code/Wordle_Project
$ python -u ""/c/VS.Code/Wordle_Project/freq-letters_list.py"
{'a': [141, 304, 307, 163, 64], 'b': [173, 16, 57, 24, 11], 'c': [198, 40, 56, 152, 31], 'd': [111, 20, 75, 69, 118], 'e': [72, 242, 177, 318, 424], 'f': [136, 8, 25, 35, 26], 'g': [115, 12, 67, 76, 41], 'h': [69, 144, 9, 28, 139], 'i': [34, 202, 266, 158, 11], 'j': [20, 2, 3, 2, 0], 'k': [20, 10, 12, 55, 113], 'l': [88, 201, 112, 162, 156], 'm': [107, 38, 61, 68, 42], 'n': [37, 87, 139, 182, 130], 'o': [41, 279, 244, 132, 58], 'p': [142, 61, 58, 50, 56], 'q': [23, 5, 1, 0, 0], 'r': [105, 267, 163, 152, 212], 's': [366, 16, 80, 171, 36], 't': [149, 77, 111, 139, 253], 'u': [33, 186, 165, 82, 1], 'v': [43, 15, 4, 9, 46, 0], 'w': [83, 44, 26, 25, 17], 'x': [0, 14, 12, 3, 8], 'y': [6, 23, 29, 3, 364], 'z': [3, 2, 11, 20, 4]}

```

→ For example, we can see for letter 'a': [141,304,307,163,64] and etc.

3. wordScore() :-

→ It accepts two arguments: (i) possible_words and (ii) frequency.

→ First we calculate the maximum frequency at each position for calculating the word score.

```

score *= 1 + (max_freq[i]-frequencies[c][i])

```

→ Score of a word was calculated using this method shown.

→ So, more is the score, less is the probability of its occurrence.

→ max_freq[i] - frequencies[c][i] will become minimum when a letter of high frequency comes.

→ Later, a dictionary is used to store words(as keys) and scores(as values) in python programming language.

4. bestWord() :-

→ It finds the best possible word based on the scores of the word we got from the function called **wordScore()** as shown above.

→ Basic analogy of finding the minimum value is used in this function.

5. wordRemover() :-

→ There are 5 checks performed for sorting out the favorable words:

→ **1st check**, we remove all words which had wrong letters in them.

→ Hence, we are left with less number of words this time. In turn our probability has increased many folds.

→ **2nd check**, we store words which is a list of those words which contain letters with correct positions.

- **3rd check**, we store a list of all words where partial letters are not in position as our result.
- **4th check**, we store words which have all good letters(yellow and green) in the list.
- Similarly, **5th check** Stores words in list which has the same frequency of letters as good_letters

Finally, we are left with the best possible collection of words.

6. WrongLetters() :-

- It receives two arguments: result and guess. Guess means the word we entered and the result is the word(5 letter) combination of letters g(green), y(yellow), w(wrong).
- It returns the list of wrong letters.

7. YellowLetters() :-

- It also accepts the same arguments WrongLetters() function.
- It returns the list of partial correct letters(i.e yellow appearing) with an index.

8. GreenLetters() :-

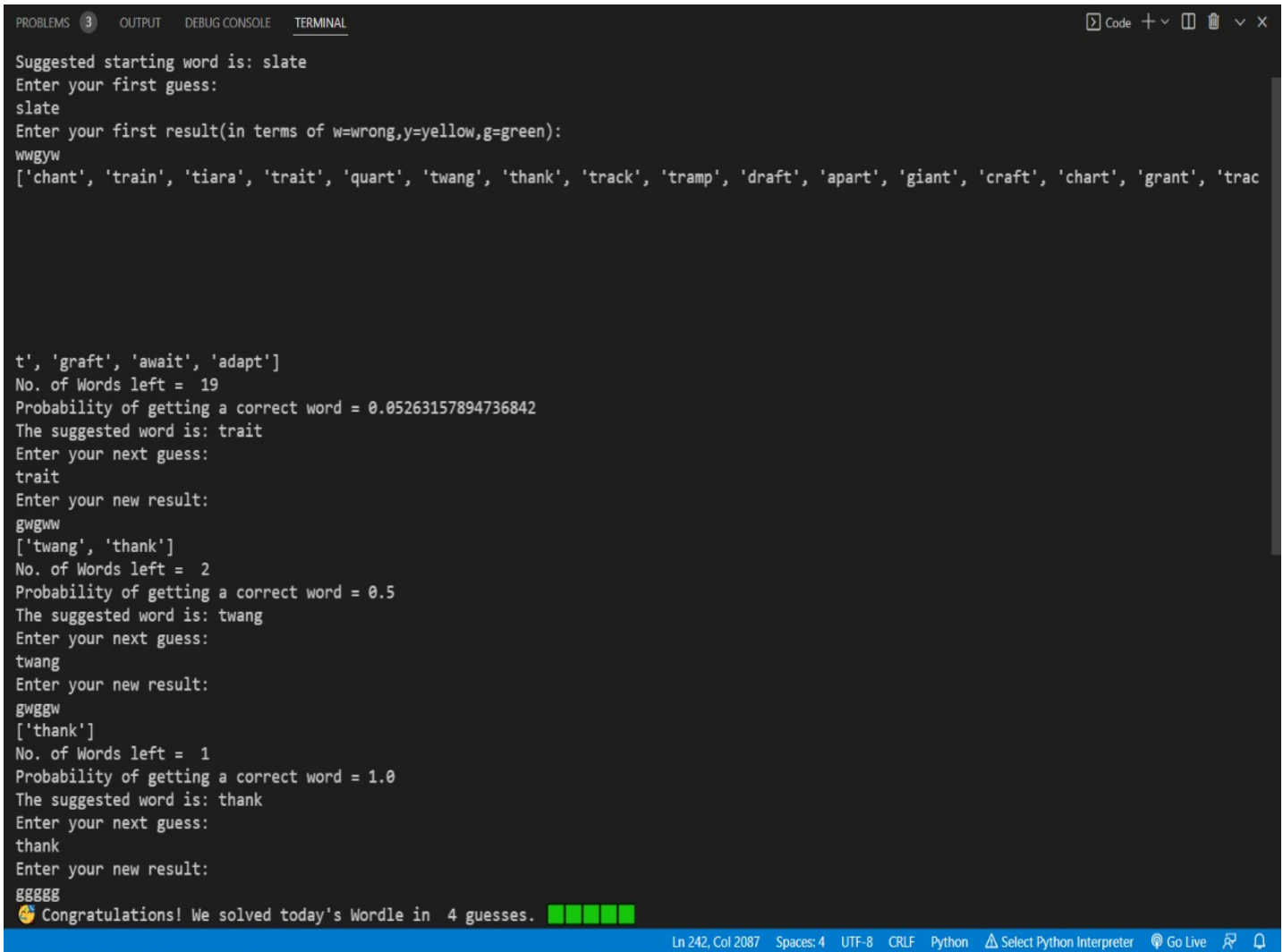
- Similarly, it accepts arguments as above.
- It returns a list of fully correct letters(i.e green appearing) with an index.

Modules imported in code :-

1. **string**(`import string`) → Most of our work is related to string.
2. **matplotlib**(`import matplotlib.pyplot as plt`) → For plotting our bar graphs.
3. **numpy**(`import numpy as np`) → For generating random numbers between 0 to 1.

Working of our code via Example :

Let's suppose at back-end word decided is : **thank**



```

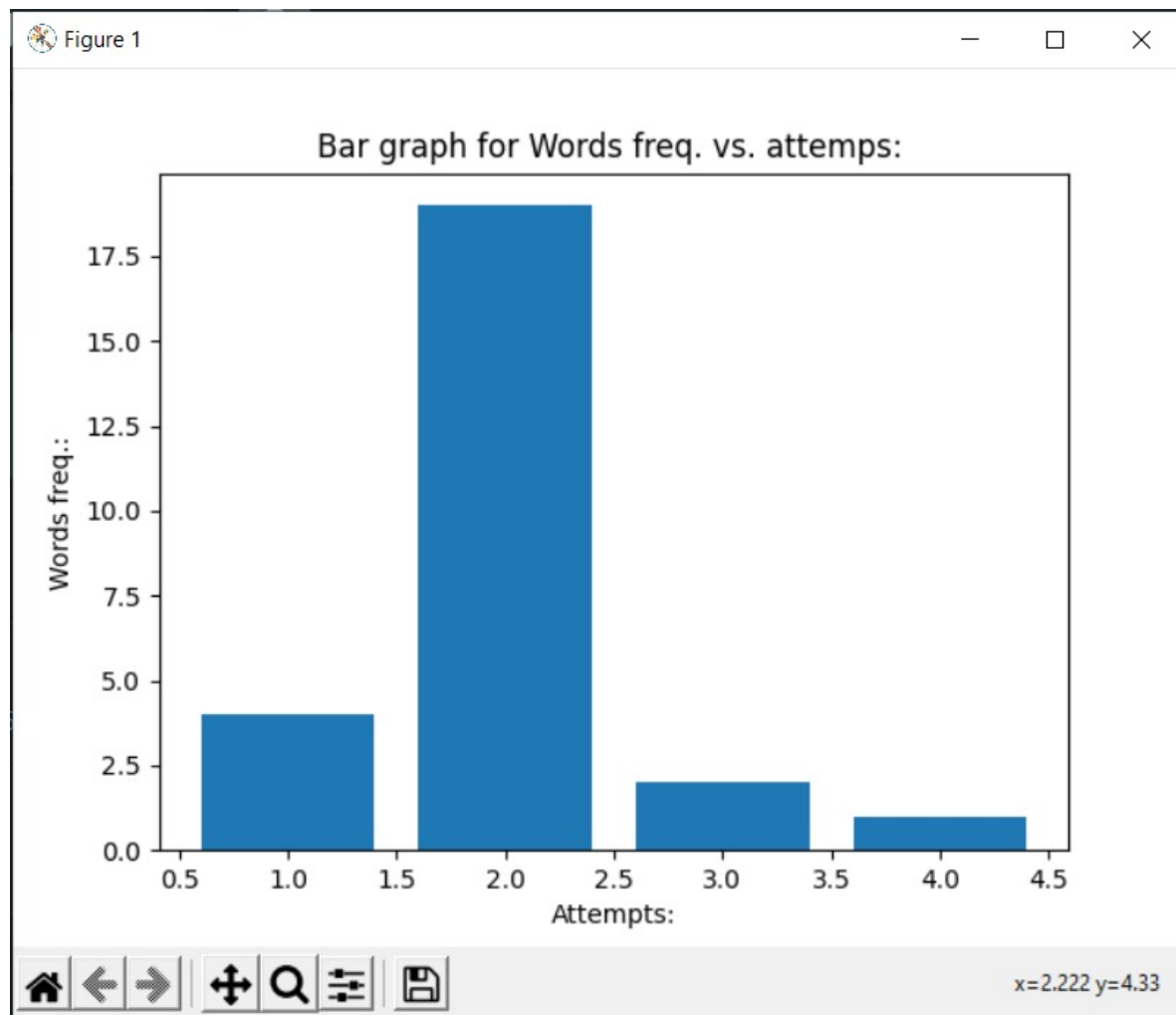
PROBLEMS 3 OUTPUT DEBUG CONSOLE TERMINAL
Suggested starting word is: slate
Enter your first guess:
slate
Enter your first result(in terms of w=wrong,y=yellow,g=green):
wwgyw
['chant', 'train', 'tiara', 'trait', 'quart', 'twang', 'thank', 'track', 'tramp', 'draft', 'apart', 'giant', 'craft', 'chart', 'grant', 'trac
t', 'graft', 'await', 'adapt']
No. of Words left = 19
Probability of getting a correct word = 0.05263157894736842
The suggested word is: trait
Enter your next guess:
trait
Enter your new result:
gwggw
['twang', 'thank']
No. of Words left = 2
Probability of getting a correct word = 0.5
The suggested word is: twang
Enter your next guess:
twang
Enter your new result:
gwggw
['thank']
No. of Words left = 1
Probability of getting a correct word = 1.0
The suggested word is: thank
Enter your next guess:
thank
Enter your new result:
ggggg
🎉 Congratulations! We solved today's Wordle in 4 guesses. ■■■■■
Ln 242, Col 2087 Spaces: 4 UTF-8 CRLF Python Select Python Interpreter Go Live

```

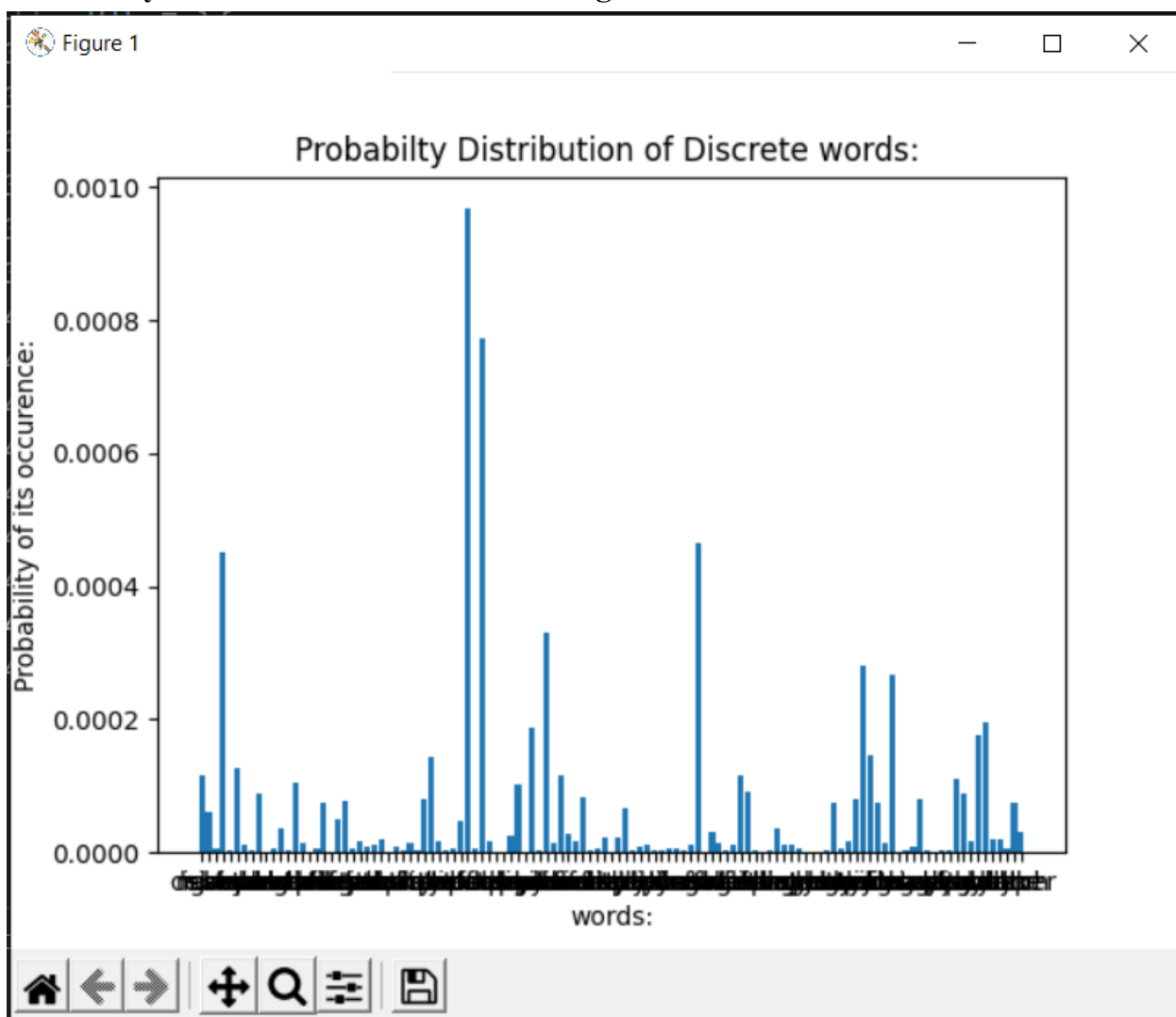
Now after running our code we see as follows in our terminal :-

- It suggests “slate” as our first word.
- After giving this word to the world, it asks for the result.
- Then, it again fetches a list of favorable words(fewer words than the previous 2315).
- It also shows the probability of getting a correct word from the list of words we get at each attempt.
- Hence, Probability increases with each attempt.

Plots of Frequency of words vs Attempts :(for above eg.)



❖ Probability of occurrence of all words using BAR-GRAPH :-



→ This plots we get at the end using the matplotlib module in python.

- Today's(29 March, 2022) Wordle solved within two attempts using our code :-

```
$ python -u ""/c/VS.Code/Wordle_Project/Wordle_Solver_Final.py"
Welcome to the Wordle Solver!!
Suggested starting word is: slate
Enter your first guess:
slate
Enter your first result(in terms of w=wrong,y=yellow,g=green):
gygww
['shall', 'shawl', 'scald', 'snarl', 'snail', 'scalp', 'scaly', 'small']
No. of Words left = 8
Probability of getting a correct word = 0.125
The suggested word is: shall
Enter your next guess:
shall
Enter your new result:
ggggg
🎉 Congratulations! We solved today's Wordle in 2 guesses.
No. of attempts you have taken: [1, 2]
No. of best possible words we chose at each attempt: [4, 8]
```

Ans was → SHALL

ACKNOWLEDGEMENT



We would like to extend our sincere thanks to our mentor for this project, Prof. Madhu Vadali. He gave us the opportunity to do this wonderful project of designing a strategy for the game ‘Wordle’. This project allowed us to do a lot of research and come up with the most efficient solution for the given problem.

All the group members actively participated and contributed to this project. This project helped us all in nurturing patience and teamwork. There are many whom we need to acknowledge and without their coordination this project would not have been completed successfully.

We are really thankful to all those people who supported and helped us in completing our project.

– Thank You !

Citations/Links :-

1.  Solving Wordle using information theory
2.  Oh, wait, actually the best Wordle opener is not “crane”...
3. <https://chidiwilliams.com/post/a-wordle-solver/>
4. [Build a Wordle Solver Using Python 3 | by Ben Bellerose | Towards Data Science](#)