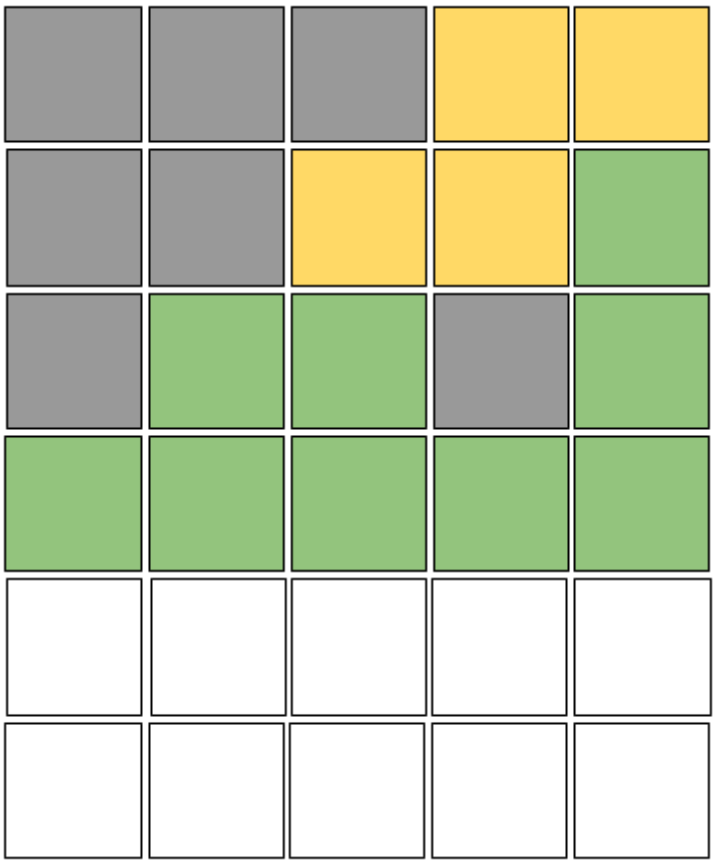

Wordle Guess Block | Image by author

Hi everyone, I'm sure by now you've all seen Wordle, Twitters's most recent obsession.

If you're like me, you struggle with this game. But don't you worry, with the help of search and data analytics, we can become great, and you can post your very own winning screenshot to your Twitter.

What is Wordle

To start things off, let me first explain what Wordle is. Wordle is a simple game where you have six chances at guessing a five-letter word.

After each prediction, the game will give you hints.

A green tile means that you predicted the correct placement of a letter.

A yellow tile means that the letter is in the word, but your prediction had the wrong position.

And lastly, a grey tile means the letter is not in the word. Every day the game's word is reset, giving everyone a chance at guessing the correct word.

Building Console Game

With our newfound understanding of Wordle, let's build a playable version of the game using python 3 for our console. We'll create a custom class containing the game board and the rules. In our class were going to have four functions.

The first function is to check if it's the end of the game. A game is over if the player has successfully guessed the secret word. If the player uses all six guesses, the game is also complete.

The second function of the class determines the outcome of the game. The function returns a tuple. The first element is a binary value representing the outcome of the game. If a player wins, the second element of the tuple is set to the guess the game was won on. If the player loses the second element of the tuple is set to 99.

The third function of the class updates the game board with the player's most recent guess.

And the fourth and final function determines if the player's guess is valid or not. This function helps verify user inputs to ensure the game is played correctly.

Building The Game Assistant

Now let's make the actual Wordle solver. Our word bank starts with 15917 words giving us an initial probability of only 0.0063% that we'll guess the word correctly. This small percentage means shrinking the search space is critical if we want to be able to win the game.

To help increase the probability that will win the game the first thing we'll want to do is exploit the game's rules to shrink our search space. Based on the colours returned by our guesses, we can filter out the grey/black letters from our word bank as we know they aren't in the secret word.

Another filter we can apply to our word bank is selecting words with yellow letters in them since we know all yellow letters are in the secret word. We can also filter out all words with yellow letters in the guessed position as we know the letter will not be in that position.

And the last exploit we'll perform is filtering all green letters in their predicted position as we know this is in the word.

Now that we know how we're going to shrink our search space, we need to determine how we're going to guess the most impactful words. To do this, we'll score each word using the sum of each letter positioning probability.

$$S(w) = \sum_{p=1}^{5} \frac{\sum_{i=1}^{n} [W_p^i = w_p]}{n}$$

Word Score Equation | Image by author

- S = word score
- w = current word
- W = word bank

- n = amount of words in the word bank
- p = current letter position in word

By using the letter positioning probability we are making our best attempt to find green & yellow letters so we can shrink our word bank and increase the probability that we guess the correct word.

If a vowel is not in our know letters we will add a bias to the word score equation to encourage finding a vowel. We do this to exploit the fact that we know every word has a vowel in it. The bias we add to the word score is the percentage of the word that is unique vowels.

$$b(w) = \sum_{p=1}^{6} \frac{[V_p \exists w]}{5}$$

Vowel Bias | Image by author

- b = vowel bias
- w = current word
- V = vowel bank

Lastly, we return the `argmax` of the word score since we know this is the most impactful word.

Wrap Up

And there you have it, we have successfully created our Wordle solver that gives us a high probability that we will win the game.

You can check a full version of the code on my GitHub here.

Thanks for reading.

Reference

- https://www.powerlanguage.co.uk/wordle/