


BULLS AND COWS GAME

Importance of Entropy

BY HITIKA GHANANI





DETAILS OF THE PROJECT

What this project does

In this project, I created a working Bulls and Cows game in Python.

The computer chooses a 4 digit number with no repeated digits, and the player tries to guess it. After every guess, the program shows bulls, cows, and updates which numbers are still possible.

How entropy is used

Along with the normal game logic, I added entropy. After each guess, the code calculates the entropy of the remaining possible secret numbers. This shows how much uncertainty remains and how much information the latest guess provides.

Goal of the project

The main idea is to connect this simple guessing game to the concept of entropy in information theory and to see how uncertainty decreases as we collect feedback after each guess.



RULES OF GAME

Basic Rules

The secret number has 4 digits.

All digits are different.

The number cannot start with 0.

The player must follow the same rules when entering guesses.

Bulls and Cows

Bull → a digit is correct and in the correct position.

Cow → a digit is correct but in the wrong position.

Example:

Secret = 5271

Guess = 5213 → 2 bulls, 1 cow

How the game progresses

You enter a guess.

The program shows bulls and cows.

Then it removes all secret numbers that don't match that feedback.

The game continues until you get 4 bulls, which means you found the secret.

WHY ENTROPY MATTERS

Entropy is the key concept

- Entropy tells us how uncertain we are about the secret number.
- At the start of the game, we have no idea which of the 4536 possible numbers is correct so entropy is high.
- As we get feedback (bulls/cows), we remove impossible numbers, and entropy drops.

Why entropy matters here

- Every guess gives us information, and entropy shows exactly how much uncertainty we removed.
- Instead of guessing randomly, entropy gives us a mathematical way to see progress.
- When entropy decreases, it means we are getting closer to the true secret.

How entropy guides the game

- After each guess, we compute:
 - How many possible secrets are left
 - What the new entropy is
 - How much information does the guess provide
- This connects the game directly to Shannon's information theory.

Why entropy is the hero

- It measures progress.
- It shows how "good" or "bad" each guess was.
- It gives us the logic behind the best strategy.
- When entropy becomes zero, the game is fully solved.

WHERE ENTROPY COMES FROM

*Candidate Set: It is a list of all secret numbers that are still possible based on the guesses so far.
At the start, this set contains 4536 valid numbers(4 digits, no repeats, no leading zero).*

Why is this set important

Entropy is calculated from the size of this set.

Large set → high entropy → lots of uncertainty.

Small set → low entropy → we're close to the answer.

How the game logic uses it

After every guess, we calculate bulls and cows.

We remove all numbers that could not produce that feedback.

The candidate set gets smaller.

Entropy is recalculated.

*So basically, the entire game becomes a process of shrinking the candidate set,
and entropy is the way we measure that shrinking.*

CALCULATING ENTROPY

How I calculate entropy in the game

Entropy is based on the size of the remaining candidate set:

$$H = \log_2(|S|)$$

Where S is the list of all possible secret numbers left after a guess.

Why this formula

Shannon's entropy formula measures how many bits of information are needed to identify the secret.

More possibilities \rightarrow more bits \rightarrow higher entropy.

Fewer possibilities \rightarrow fewer bits \rightarrow lower entropy.

Example: At the start $\rightarrow |S| = 4536 \rightarrow H \approx 12.14$ bits

If only 100 possibilities remain:

$$H = \log_2(100) \approx 6.64 \text{ bits}$$

If only 1 possibility remains:

$$H = 0 \text{ bits}$$

Why this matters

Entropy shows exactly how much uncertainty is left.

Watching entropy drop tells us how close we are to solving the secret.

Every guess becomes measurable in terms of information gained.

UPDATING THE CANDIDATE SET

How each guess reduces possibilities

When the player enters a guess, the game compares it to the secret and gives:

- Bulls → correct digit in the right place
- Cows → correct digit but wrong place

Filtering based on feedback

After getting (bulls, cows), we remove every number in the candidate set that does not produce the same feedback.

Formally:

$$S' = \{s \in S \mid \text{bulls_cows}(s, \text{guess}) = (b, c)\}$$

What this achieves

- The candidate set becomes smaller.
- The uncertainty about the secret decreases.
- Entropy drops automatically.

Why this is important

- This step is the “engine” of the entire game.
- Without filtering, entropy wouldn’t change.
- Filtering and entropy is the core logic behind efficiently solving the puzzle.

EXPECTED INFORMATION GAIN

What information gain means

Every guess teaches us something. Information gain measures how much uncertainty we removed after a specific guess.

How I calculate it

Before a guess:

$$H_{\text{before}} = \log_2(|S|)$$

After filtering the candidate set:

$$H_{\text{after}} = \log_2(|S|)$$

So the information gain is:

$$\Delta H = H_{\text{before}} - H_{\text{after}}$$

How to interpret it

High information gain → the guess eliminated a lot of possibilities.

Low information gain → the guess didn't help much.

Zero information gain → no change in uncertainty.

Why this matters

It helps us understand which guesses were actually useful.

It shows how the game becomes more certain turn by turn.

It connects the guessing process directly to Shannon's information theory.

BEST STRATEGY

The main idea

The best strategy is to choose the guess that gives the maximum Expected Information Gain (EIG).

Why this works

High EIG → the guess will, on average, eliminate the most possibilities.

This causes the largest drop in entropy.

The search space becomes smaller much faster.

Fewer turns are needed to find the secret.

What an “optimal” guess does

Splits the candidate set into balanced groups for different feedback patterns.

Avoids guesses that leave uneven or large leftover groups.

Focuses on guesses that teach us the most.

EXPECTED INFORMATION GAIN

How entropy changes during the game

As we keep guessing, the candidate set gets smaller.

This naturally makes entropy decrease.

Here's a simple example to show how uncertainty goes down:

Sample Turns

<i>Turn</i>	<i>Candidate Count</i>	<i>Entropy (bits)</i>	<i>Info Gain</i>
<i>1</i>	<i>4536</i>	<i>12.14</i>	<i>-</i>
<i>2</i>	<i>720</i>	<i>9.49</i>	<i>2.65</i>
<i>3</i>	<i>108</i>	<i>6.75</i>	<i>2.74</i>
<i>4</i>	<i>15</i>	<i>3.90</i>	<i>2.85</i>
<i>5</i>	<i>1</i>	<i>0.00</i>	<i>3.90</i>

What this shows

Entropy steadily goes down as we narrow the search space.

Each informative guess gives a strong “drop” in uncertainty.

Once entropy reaches 0, the secret is completely determined.

The game becomes a process of reducing entropy to zero.

CONCLUSION

What I learned

- Entropy is a powerful way to measure uncertainty in a problem.
- In Bulls and Cows, entropy fits naturally because each guess removes some possibilities.
- Watching entropy drop helps understand how much information each guess gives.

Main takeaways from the project

- The candidate set is the core structure behind the game.
- Entropy = \log_2 of the candidate set size → shows progress in a measurable way.
- Realized information gain tells us how effective each guess actually was.
- Expected information gain helps identify the best possible guess before playing it.

Best strategy

Pick the guess that maximizes expected information gain, because it reduces uncertainty the fastest.

Why this project is meaningful

- It connects programming, logic, and information theory.
- It shows how a simple game can actually illustrate deep concepts like uncertainty and information gain.
- It helped me understand entropy not just as a formula, but as something you can use in a real decision-making process.