1. **Problem Description**

Given a string s, return whether s is a **valid number**.  
  
For example, all the following are valid numbers: "2", "0089", "-0.1", "+3.14", "4.", "-.9", "2e10", "-90E3", "3e+7", "+6e-1", "53.5e93", "-123.456e789", while the following are not valid numbers: "abc", "1a", "1e", "e3", "99e2.5", "--6", "-+3", "95a54e53".

Formally, a **valid number** is defined using one of the following definitions:

1. An **integer number** followed by an **optional exponent**.
2. A **decimal number** followed by an **optional exponent**.

An **integer number** is defined with an **optional sign** '-' or '+' followed by **digits**.

A **decimal number** is defined with an **optional sign** '-' or '+' followed by one of the following definitions:

1. **Digits** followed by a **dot** '.'.
2. **Digits** followed by a **dot** '.' followed by **digits**.
3. A **dot** '.' followed by **digits**.

An **exponent** is defined with an **exponent notation** 'e' or 'E' followed by an **integer number**.

The **digits** are defined as one or more digits.

**Example 1:**

**Input:** s = "0"

**Output:** true

**Example 2:**

**Input:** s = "e"

**Output:** false

**Example 3:**

**Input:** s = "."

**Output:** false

**Constraints:**

* 1 <= s.length <= 20
* s consists of only English letters (both uppercase and lowercase), digits (0-9), plus '+', minus '-', or dot '.'.

1. **Hints:**

* Using state changing to solve the problem
* Iterate through the string number and change state by the current token.
* List all of the available cases to create a state dict, for example:
  + The first token of the string should be a sign (+ or -) or a digits or a ‘.’ or blank
  + Then based on the first token we can know that the second token should be a digit (if the first is sign or blank or ‘.’), we move to that state for available third token
  + …
* Break if the current token of the string is not in the available token of current state