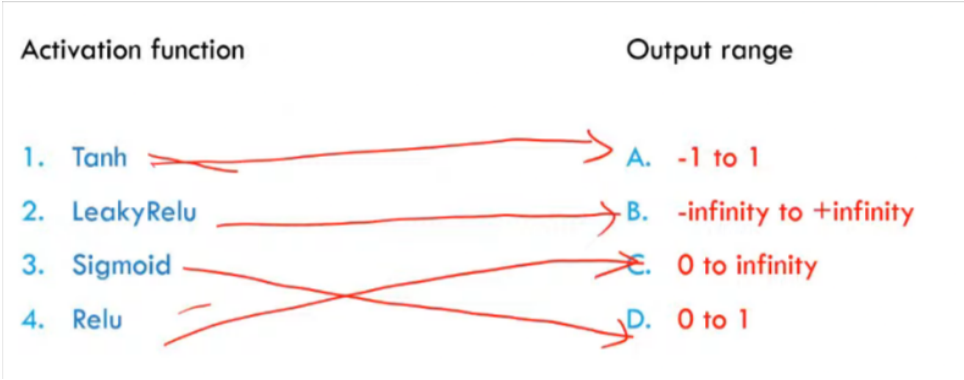
Session 9

Thursday, April 06, 2023 11:24 PM



The Rectified Linear Unit (ReLU) activation function returns the input value if it is positive, and zero otherwise. Mathematically, the ReLU activation function can be defined as:

f(x) = max(0, x)

where x represents the input to the function. The output of the ReLU activation function is always non-negative. If the input is negative, the function outputs 0. If the input is positive, the function outputs the same value as the input. In other words, the ReLU activation function "turns on" or activates neurons that receive positive inputs, while "turning off" neurons that receive negative inputs.

The tanh activation function, also known as the hyperbolic tangent activation function, maps any input value to a range between -1 and 1. The output of the tanh function is computed using the formula:

tanh(x) = (e^x - e^-x) / (e^x + e^-x)

where x is the input value. For example, if the input is 0, the output is 0. If the input is positive, the output will be closer to 1, and if the input is negative, the output will be closer to -1.

The Leaky ReLU (Rectified Linear Unit) activation function is a variant of the standard ReLU activation function that allows for a small, non-zero gradient when the input is negative. The output of the Leaky ReLU function is defined as:

f(x) = { x, if x > 0, ax, if x <= 0 }

where x is the input value and a is a small constant representing the slope of the function when x is negative.

For example, if a = 0.01 and the input is -1, the output of the Leaky ReLU function would be -0.01 (-1 multiplied by the slope). If the input is positive, the output will be equal to the input value.

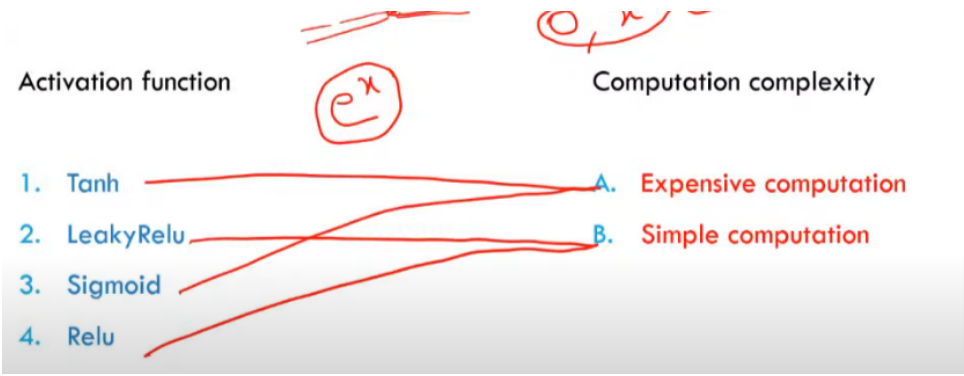
The sigmoid activation function is a mathematical function that maps any input value to a value between 0 and 1. The output of the sigmoid function is given by the formula:

f(x) = 1 / (1 + e^-x)

where x is the input value.

For small positive inputs, the output of the sigmoid function approaches 1, while for large negative inputs, the output approaches 0. When the input is 0, the output of the sigmoid function is exactly 0.5.

The sigmoid function was commonly used as an activation function in neural networks in the past, but it has fallen out of favor for some applications due to its tendency to saturate when the input is very large or very small, which can make training difficult.



Identify correct sequence of a SQL statement

A. SELECT -> FROM -> GROUP BY -> WHERE -> HAVING
B. SELECT -> GROUP BY -> FROM -> WHERE -> HAVING
C. SELECT -> GROUP BY -> FROM -> HAVING -> WHERE
D. SELECT -> FROM-> WHERE -> GROUB BY -> HAVING

Merge below two dictionaries

dict1 = {'a' : 1, 'b' : 2, 'c' : 3}
dict2 = {'d' : 4, 'e' : 5, 'f' : 6}

> Bigdict

Method 1: Using the `update()` method

```
pythonCopy Code
# create two dictionaries dict1 = {'a': 1, 'b': 2} dict2 = {'c': 3, 'd': 4} # merge the dictionaries
dict1.update(dict2) # print the merged dictionary print(dict1)
```

Output:

```
Copy Code
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

In this method, we first create two dictionaries `dict1` and `dict2`. We then use the `update()` method on `dict1` to merge the contents of `dict2` into `dict1`. The result is `dict1` containing all the keys and values from both original dictionaries.

Method 2: Using the `**` operator

```
pythonCopy Code
# create two dictionaries dict1 = {'a': 1, 'b': 2} dict2 = {'c': 3, 'd': 4} # merge the dictionaries
merged_dict = {dict1, dict2} # print the merged dictionary print(merged_dict)
```

Output:

```
Copy Code
{'a': 1, 'b': 2, 'c': 3, 'd': 4}
```

In this method, we use the `**` operator to merge the contents of the two dictionaries into a new dictionary called `merged_dict`. This creates a new dictionary that contains all the keys and values from both original dictionaries.

What is the difference of arithmetic mean and geometric mean of these numbers

1,2,8,16

1. 1.75
2. 2.75
3. 3.75
4. 4.75

To calculate the arithmetic mean of a set of numbers, you add up all the numbers and then divide by the total number of numbers. To calculate the geometric mean of a set of numbers, you multiply all the numbers together and then take the nth root of the result, where n is the total number of numbers.

So, to find the arithmetic mean and geometric mean of the numbers 1, 2, 8, and 16:

Arithmetic mean:

(1 + 2 + 8 + 16) / 4 = 27 / 4 = 6.75

The arithmetic mean of these numbers is 6.75.

Geometric mean:

√(1 × 2 × 8 × 16) = √(256) = 16/4

The geometric mean of these numbers is 4.

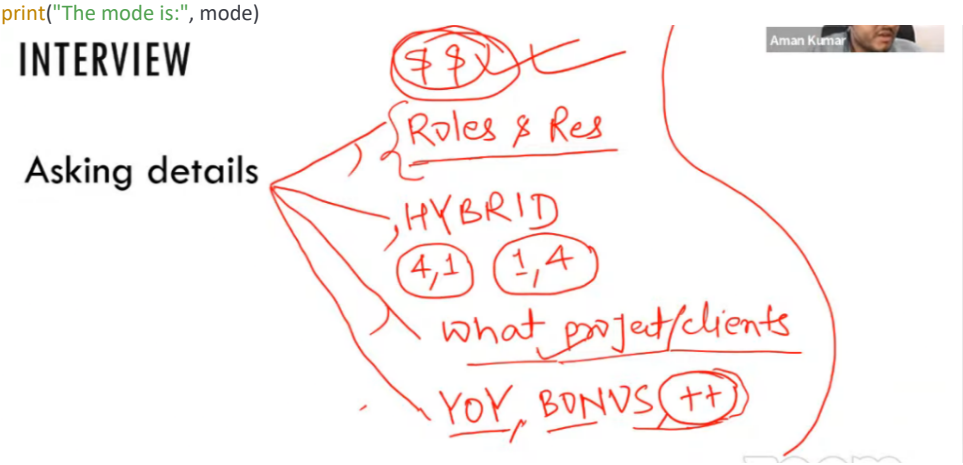
I trained a fraud detection model, I am trying to predict "fraud cases". Choose the correct statement on test set

1. Precision is the percentage of "non fraud" from "predicted frauds"
2. Precision is percentage of "fraud" from "predicted fraud"
3. Recall is percentage of "fraud" from "predicted non fraud"
4. Recall is percentage of "predicted fraud" from "fraud"

Find most frequent number from this set using python

numlist = [8,4,8,2,2,5,8,0,3,5,2,5,8,9,3,8]

```
# loop through the numbers and update the frequency count in the dictionary
for num in numbers:
if num in freq_dict:
freq_dict[num] += 1
else: freq_dict[num] = 1
# find the key with the highest value (i.e., the mode)
mode = max(freq_dict, key=freq_dict.get)
# print the result
print("The mode is:", mode)
```



Is there any hiring platform in UK ?