Compute Derivatives using Numpy

Differentiation is the act of computing a derivative. The derivative of the function y = f (x) of the variable x is a measure of the rate at which the value of the function y changes with respect to the change of the variable x. This is called the derivative of f relative to x.

Performance findings are its rate of rapid change in relation to one of its variables. This is equivalent to finding the slope of a solid line in operation in a particular area.we can find the differentiation of mathematical expressions in the form of variables by using diff() function in SymPy and NumPy packages.

Mathematical Rules for Derivative Calculations:

1) The Law of Power

The law of power says:

$$\boldsymbol{f(x) = x^n \;\rightarrow\; f'(x) = nx^{n-1}}$$

Self-explanatory, if you have listened to a certain calculation section before. If you have not already done so, let us move on to a simple example. Your f (x) function is equal to x to five. Now use the power rule to calculate the output

$$\boldsymbol{f(x) = x^5}$$

$$f'(x) = 5x^{5-1}$$

$$f'(x) = 5x^4$$

2) Product Law

Product law states that if f (x) and g (x) are two separate functions, then the deductions are calculated as the first working times based on the combined and the second times based on the first. That may sound a little confusing when expressed in words, so here is a example:

$$\boldsymbol{F(x) = f(x) \times g(x) \;\rightarrow\; F'(x) = f(x) \times g'(x) + g(x) \times f'(x)}$$

Let's count one example by hand. We have the following:

$$\boldsymbol{F(x) = \left(x^2 + 1\right) \times \cos(x)}$$

$$F'(x) = (x^2 + 1) \times (-\sin(x)) + \cos(x) \times (2x)$$

$$F'(x) = 2xcos(x) - (x^2 + 1)\sin(x)$$

3) Chain Rule

If you decide to go deep into the machine learning algorithms you will see the law of the chain coming from everywhere - gradient dropping, going backwards, naming it. It deals with consolidated operations, for example, f (g (x)) and states that the findings are calculated as the output of the external function repeated by the internal function, and then all are repeated by the appearance of the internal function.

$$\boldsymbol{F(x) = f\big(g(x)\big) \quad \rightarrow \quad F'(x) = f'\big(g(x)\big) \times g'(x)}$$

Here is a example:

$$\boldsymbol{F(x) = \big(x^2 - 3x + 5\big)^3}$$

$$F'(x) = 3 \times (x^2 - 3x + 5)^2 \times (2x - 3)$$

$$F'(x) = (6x - 9)(x^2 - 3x + 5)^2$$

Let us calculate using SymPy and NumPy:

Example 1:
Let us consider we want to find derivatives of function y = x**3 +x**2 + 1 at x=3, if we solve it will get answer 33.

```
from sympy import *
import numpy as np
x = Symbol('x')
y = x**3 +x**2 + 1
yprime = y.diff(x)
yprime
```

Output:

$$3x^2 + 2x$$

```
f = lambdify(x, yprime, 'numpy')
f(3)
```

Output:

33

Example 2:
Let us consider we want to find derivatives of the function y = sin(x) at x=0, if we solve it will get answer 1.0

```
y = sin(x)
yprime = y.diff(x)
yprime
```

Output:

$$\cos(x)$$

```
f = lambdify(x, yprime, 'numpy')
f(0)
```

Output:

1.0

Example 3:

Let us consider we want to find derivatives of the function y = sin(x)*cos(x) at x=0, if we solve it will get answer 1.0

```
y = cos(x)*sin(x)
yprime = y.diff(x)
yprime
```

Output:

$$-\sin^2(x) + \cos^2(x)$$

```
f = lambdify(x, yprime, 'numpy')
f(0)
```

Output:

1.0