Principles of Machine Learning: Exercise 1

Alina Pollehn (3197257), Julian Litz (3362592), Manuel Hinz (3334548)
Name 4-6
Name 7-9
Name 10-11

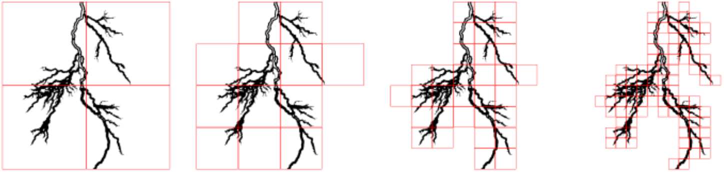
06.11.2023

test

Fractal dimensions

1. Binarize the image
2. Partition the image into 2^l boxes for $l = 1, \ldots, L-2$,
3. Calculate the fractal dimension using linear regression

$$D \cdot \log\left(\frac{1}{s_l}\right) + b = \log(n_l)$$



$s_l = \frac{1}{2}, n_l = 4$ $s_l = \frac{1}{4}, n_l = 12$ $s_l = \frac{1}{8}, n_l = 32$ $s_l = \frac{1}{16}, n_l = 94$ ...

Implementation: Box counting

```
def box_counting(img):
    w, h = img.shape
    n_ls = []
    # l runs from 1 to 9-2 =7
    for l in range(1,8):
        s_l = 1/2**l
        # get box sizes
        box_sizeW = s_l * w
        box_sizeH = s_l * h
        n_l=0
        #each l has 2**l boxes
        for box_w in range(0,(2**l)):
            for box_h in range(0,(2**l)):
                #check if any value in the box is equal 1.
                #If so increment n_l by one
                if (np.any(img[int(box_w * box_sizeW):int((box_w+1) * box_sizeW),\
                               int(box_h * box_sizeH): int((box_h+1) * box_sizeH)]\
                           ==1)):
```

Implementation: Calculating the fractal dimension

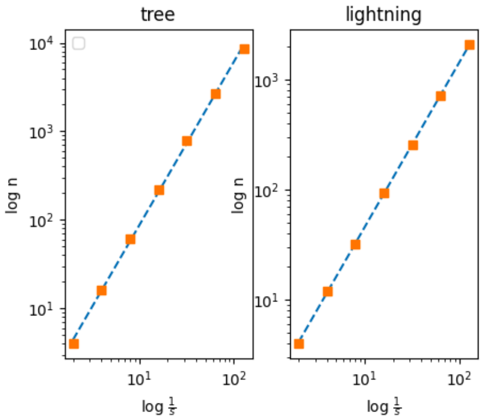
```
def slope(n_l):
    inverted_s_l=[2**l for l in range(1,8)]
    matX=np.vander(np.log(inverted_s_l),2,increasing=True)
    b,D=la.lstsq(matX, np.log(n_l),rcond=None)[0]
    return b,D
```

where np.vander generates the Vandermonde matrix

$$V = \begin{bmatrix} 1 & \log(2^1)^1 \\ 1 & \log(2^2)^1 \\ \vdots & \vdots \\ 1 & \log(2^{L-2})^1 \end{bmatrix}$$

Results

1. Fractal dimension
   - Tree: ≈ 1.846
   - Lighting: ≈ 1.493
2. The fractal dimension of the image "lighting.png" is higher



Learnings

1. Fractal dimensions
2. Fractal dimensions as a least squares problem using box counting
3. Application of least squares to a wider class of problems