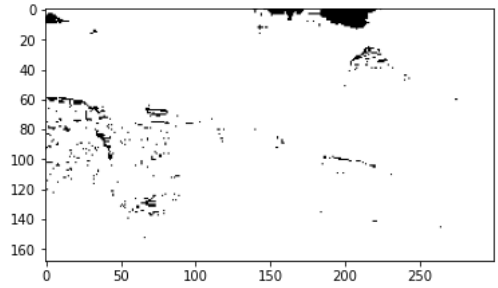
```
from skimage.color import rgb2gray
import numpy as np
import cv2
import matplotlib.pyplot as plt
%matplotlib inline
from scipy import ndimage


image = plt.imread('/content/sky.jpeg')
image.shape
plt.imshow(image)
```

```
<matplotlib.image.AxesImage at 0x7f4f3284e5b0>
```

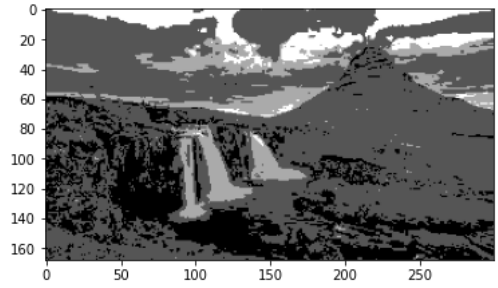
```
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)


gray_r = gray.reshape(gray.shape[0]*gray.shape[1])
for i in range(gray_r.shape[0]):
    if gray_r[i] > gray_r.mean():
        gray_r[i] = 1
    else:
        gray_r[i] = 0
gray = gray_r.reshape(gray.shape[0],gray.shape[1])
plt.imshow(gray, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f4f3454fe80>
```



```
gray = rgb2gray(image)
gray_r = gray.reshape(gray.shape[0]*gray.shape[1])
for i in range(gray_r.shape[0]):
    if gray_r[i] > gray_r.mean():
        gray_r[i] = 3
    elif gray_r[i] > 0.5:
        gray_r[i] = 2
    elif gray_r[i] > 0.25:
        gray_r[i] = 1
    else:
        gray_r[i] = 0
gray = gray_r.reshape(gray.shape[0],gray.shape[1])
plt.imshow(gray, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f4f327588b0>
```



```
image = plt.imread('/content/sky.jpeg')
plt.imshow(image)
```

```
<matplotlib.image.AxesImage at 0x7f4f326b8520>
```

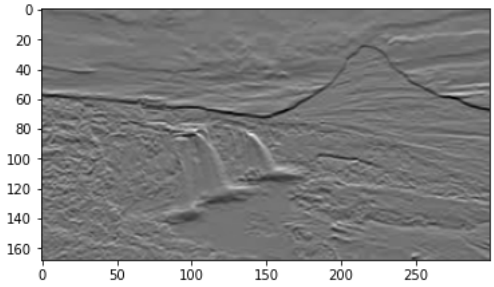
```
# converting to grayscale
gray = rgb2gray(image)

# defining the sobel filters
sobel_horizontal = np.array([np.array([1, 2, 1]), np.array([0, 0, 0]), np.array([-1, -2, -1])])
print(sobel_horizontal, 'is a kernel for detecting horizontal edges')

sobel_vertical = np.array([np.array([-1, 0, 1]), np.array([-2, 0, 2]), np.array([-1, 0, 1])])
print(sobel_vertical, 'is a kernel for detecting vertical edges')
```

```
[[ 1  2  1]
 [ 0  0  0]
 [-1 -2 -1]] is a kernel for detecting horizontal edges
[[-1  0  1]
 [-2  0  2]
 [-1  0  1]] is a kernel for detecting vertical edges
```

```
out_h = ndimage.convolve(gray, sobel_horizontal, mode='reflect')
out_v = ndimage.convolve(gray, sobel_vertical, mode='reflect')
# here mode determines how the input array is extended when the filter overlaps a border.
```

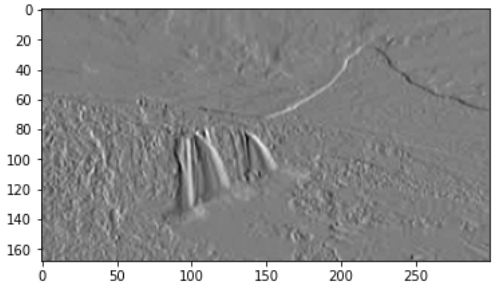
```
plt.imshow(out_h, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f4f3269c4c0>
```



```
plt.imshow(out_v, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f4f325f7430>
```

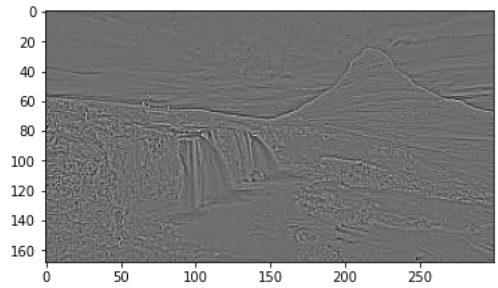


```
kernel_laplace = np.array([np.array([1, 1, 1]), np.array([1, -8, 1]), np.array([1, 1, 1])])
print(kernel_laplace, 'is a laplacian kernel')
```

```
[[ 1  1  1]
 [ 1 -8  1]
 [ 1  1  1]] is a laplacian kernel
```

```
out_l = ndimage.convolve(gray, kernel_laplace, mode='reflect')
plt.imshow(out_l, cmap='gray')
```

```
<matplotlib.image.AxesImage at 0x7f4f325d9550>
```



```
pic = plt.imread('/content/sky.jpeg')/255  # dividing by 255 to bring the pixel values between 0 and 1
print(pic.shape)
plt.imshow(pic)
```

```
(168, 300, 3)
<matplotlib.image.AxesImage at 0x7f4f32515610>
```

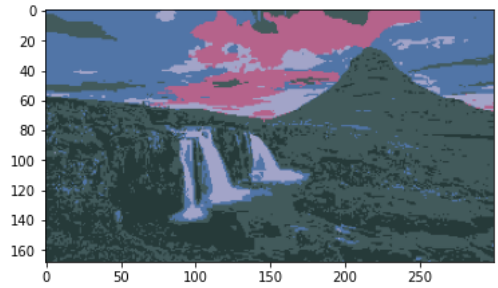
```
pic_n = pic.reshape(pic.shape[0]*pic.shape[1], pic.shape[2])
pic_n.shape
```

```
(50400, 3)
```

```
from sklearn.cluster import KMeans
kmeans = KMeans(n_clusters=5, random_state=0).fit(pic_n)
pic2show = kmeans.cluster_centers_[kmeans.labels_]
```

```
cluster_pic = pic2show.reshape(pic.shape[0], pic.shape[1], pic.shape[2])
plt.imshow(cluster_pic)
```

```
<matplotlib.image.AxesImage at 0x7f4f2ca5b850>
```



✓ 1s completed at 6:37 PM