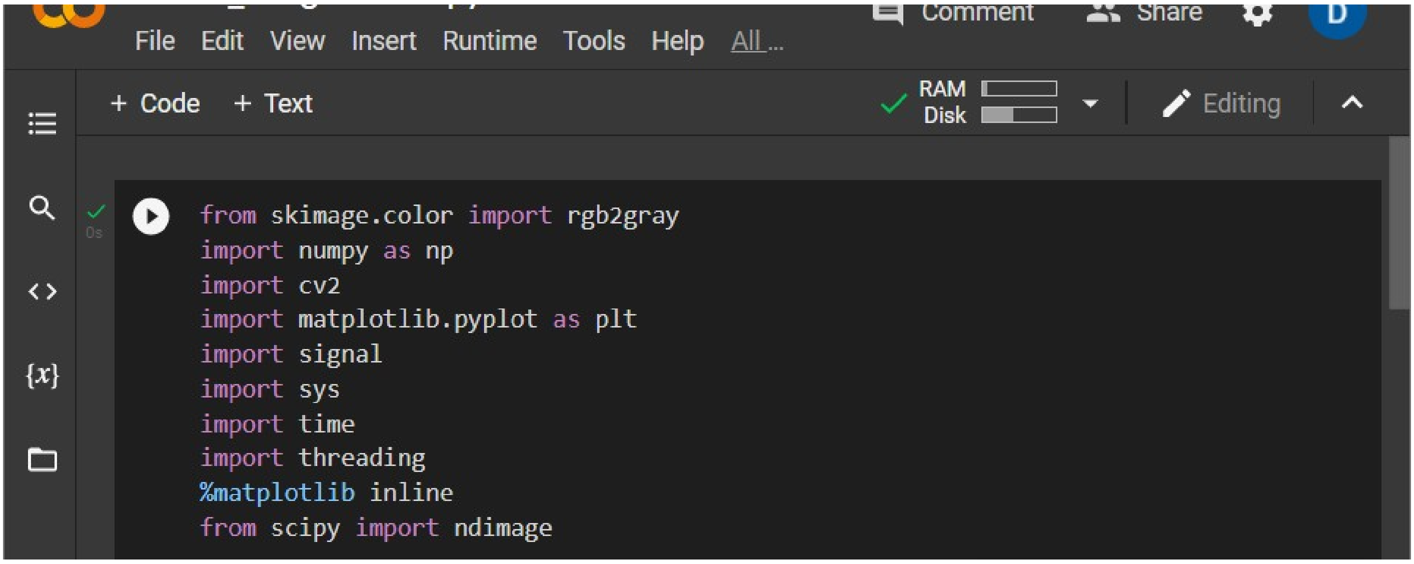
Implementation of image segmentation algorithm

Code and Screenshot

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

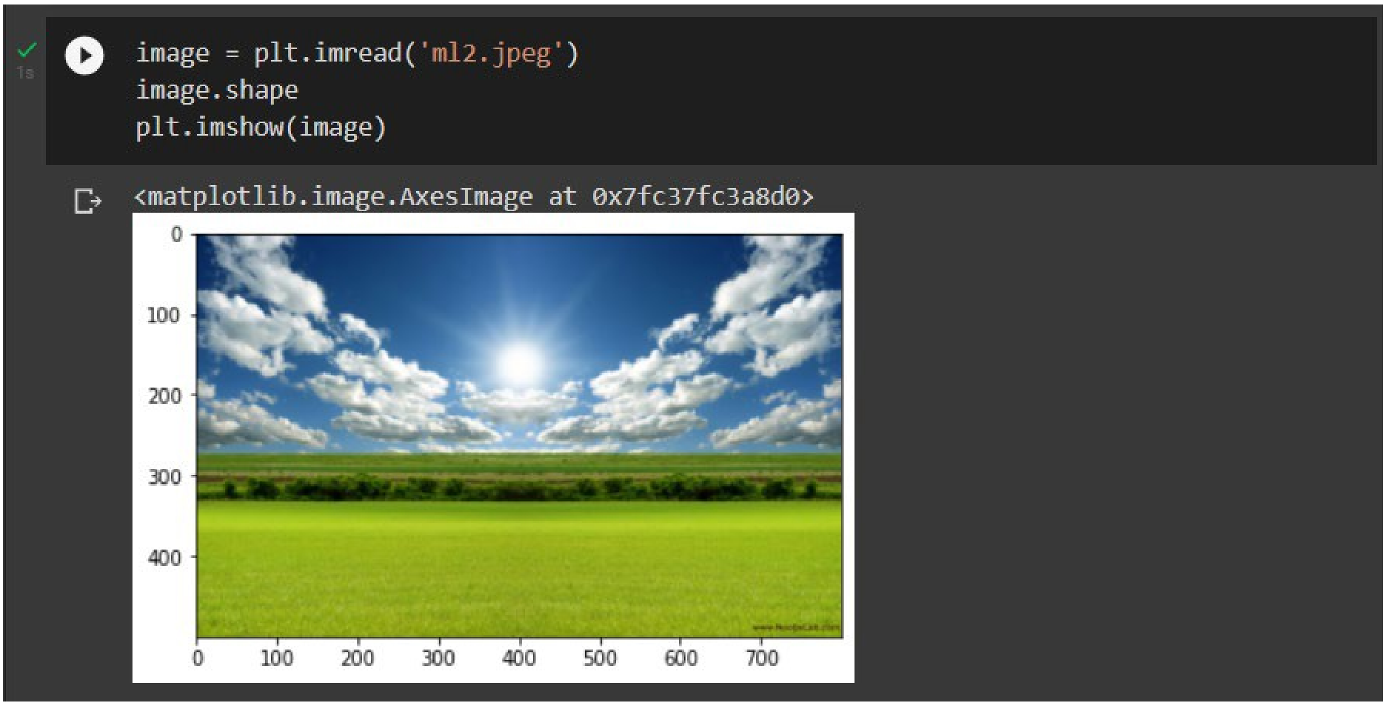


```
from skimage.color import rgb2gray
import numpy as np
import cv2
import matplotlib.pyplot as plt
import signal
import sys
import time
import threading
%matplotlib inline
from scipy import ndimage
```

image = plt.imread('ml2.jpeg')

image.shape

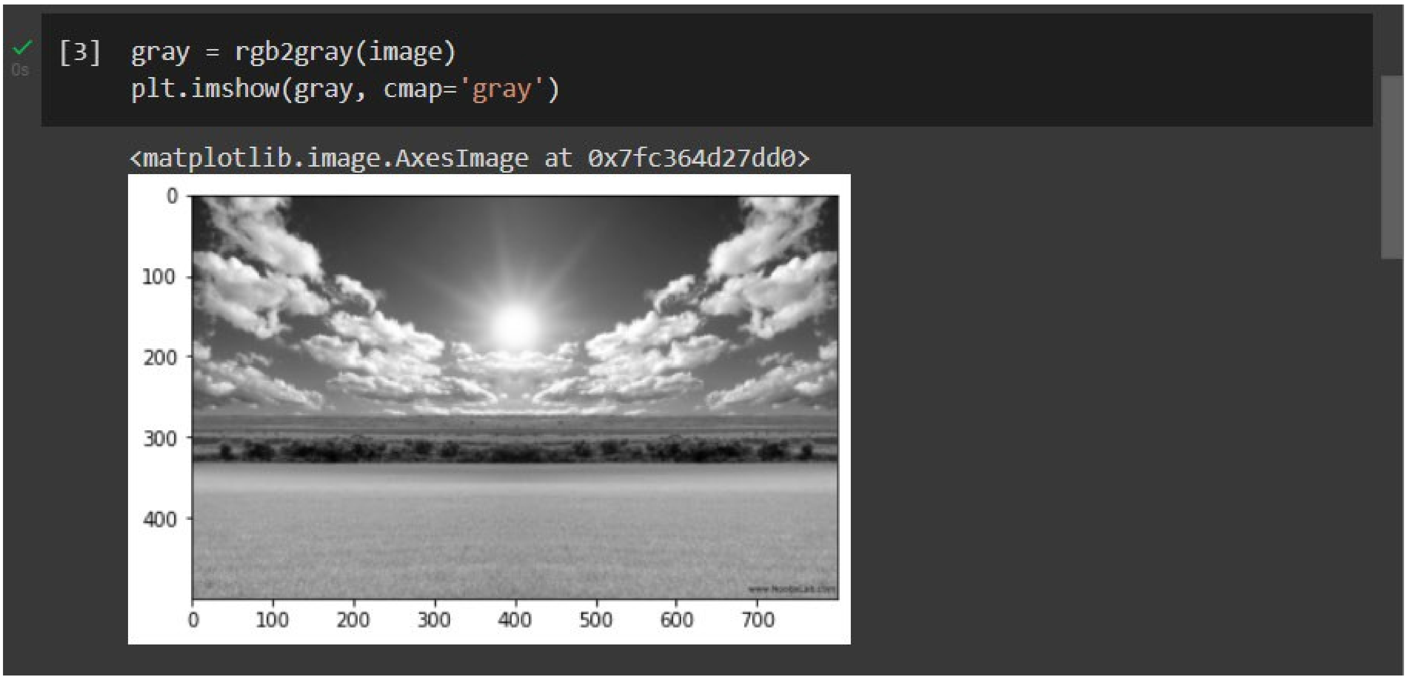
plt.imshow(image)



```
image = plt.imread('ml2.jpeg')
image.shape
plt.imshow(image)
```

```
<matplotlib.image.AxesImage at 0x7fc37fc3a8d0>
```

```
gray = rgb2gray(image)
plt.imshow(gray,  cmap='gray')
```

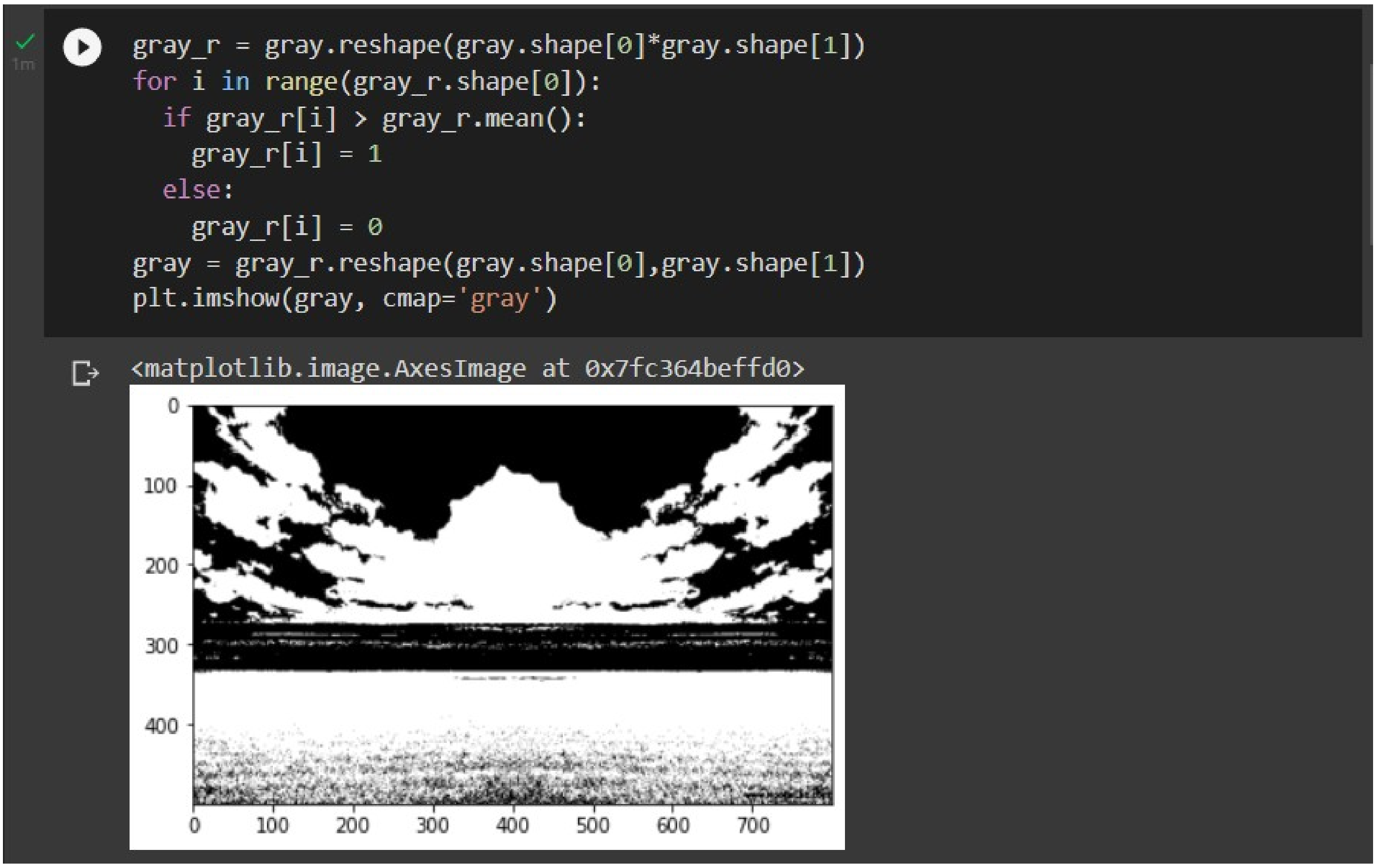


```
[3] gray = rgb2gray(image)
    plt.imshow(gray, cmap='gray')

<matplotlib.image.AxesImage at 0x7fc364d27dd0>
```

```
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
```

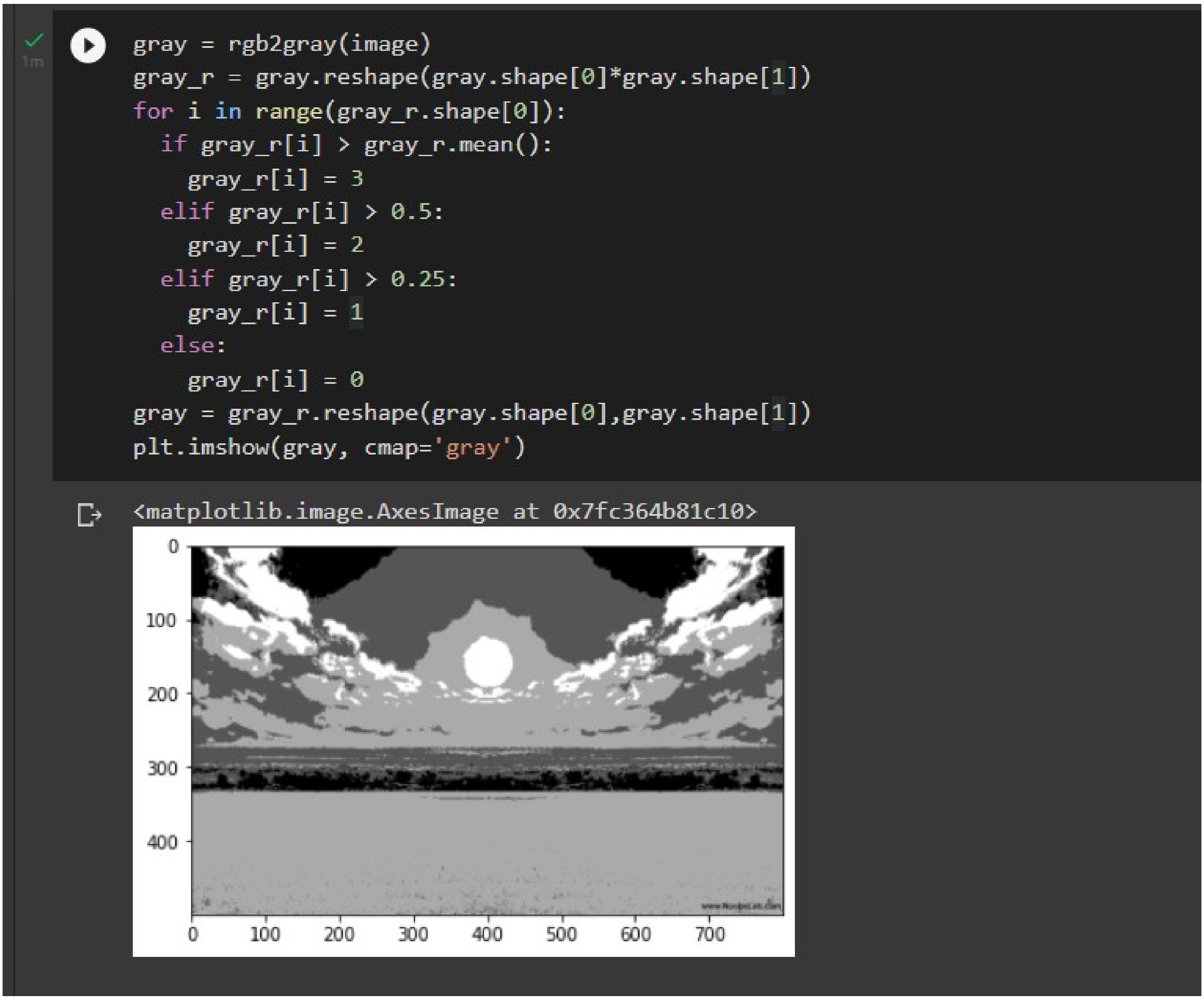
```
gray = gray_r.reshape(gray.shape[0],gray.shape[1])
plt.imshow(gray, cmap='gray')
```



```
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 0x7fc364beffd0>
```

```
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')
```



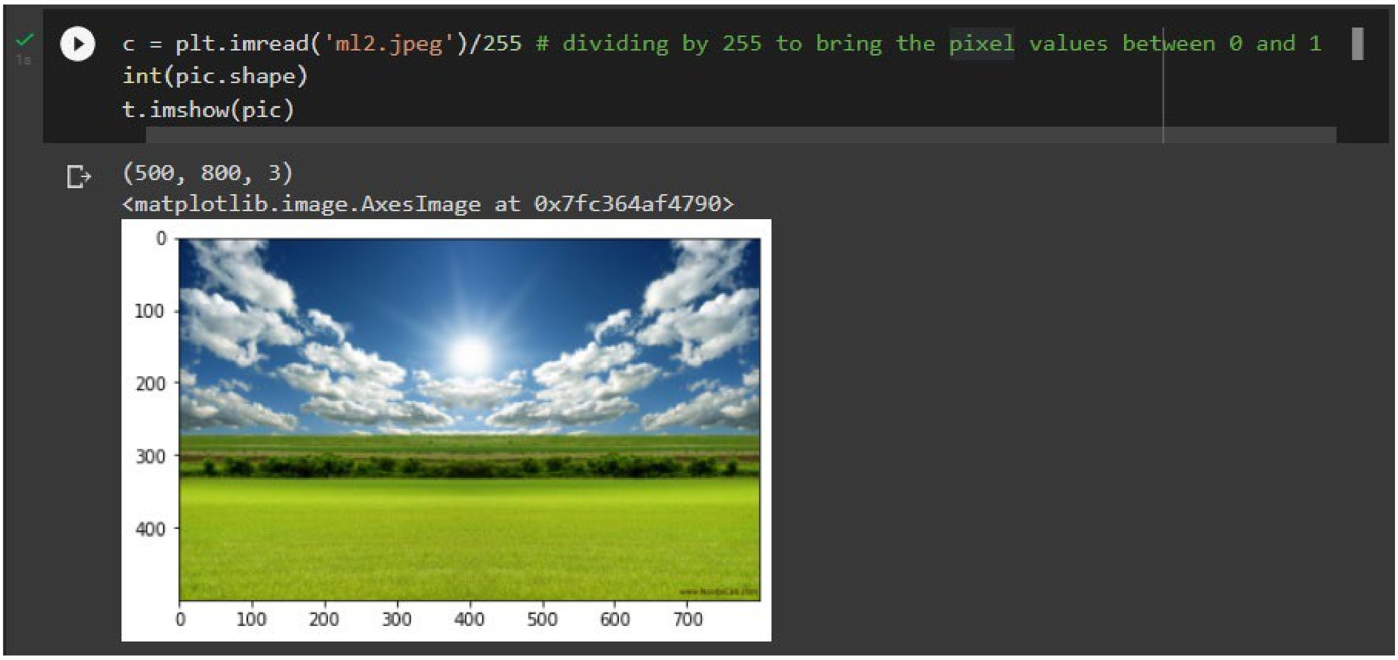
```
gray = rgb2gray(image)
gray_r = gray.reshape(gray.shape[0]*gray.shape[1])
for i in range(gray_r.shape[0]):
  if gray_r[i] > gray_r.mean():
    gray_r[i] = 3
  elif gray_r[i] > 0.5:
    gray_r[i] = 2
  elif gray_r[i] > 0.25:
    gray_r[i] = 1
  else:
    gray_r[i] = 0
gray = gray_r.reshape(gray.shape[0],gray.shape[1])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

```
<matplotlib.image.AxesImage at 0x7fc364b81c10>
```

pic = plt.imread('ml2.jpeg')/255 # dividing by 255 to bring the pixel values between 0 and 1

print(pic.shape)

plt.imshow(pic)

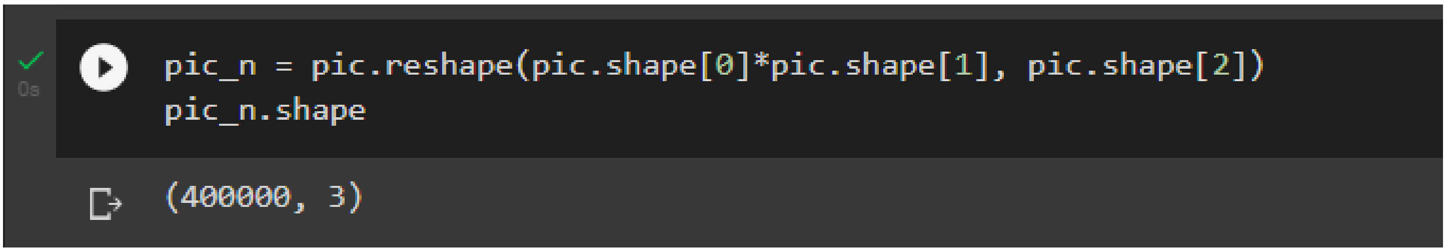


```
c = plt.imread('ml2.jpeg')/255 # dividing by 255 to bring the pixel values between 0 and 1
int(pic.shape)
t.imshow(pic)
```

```
(500, 800, 3)
<matplotlib.image.AxesImage at 0x7fc364af4790>
```

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

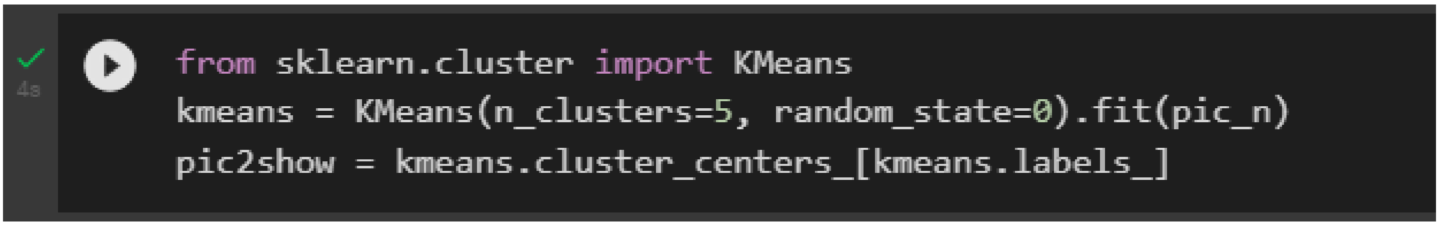
pic_n.shape



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

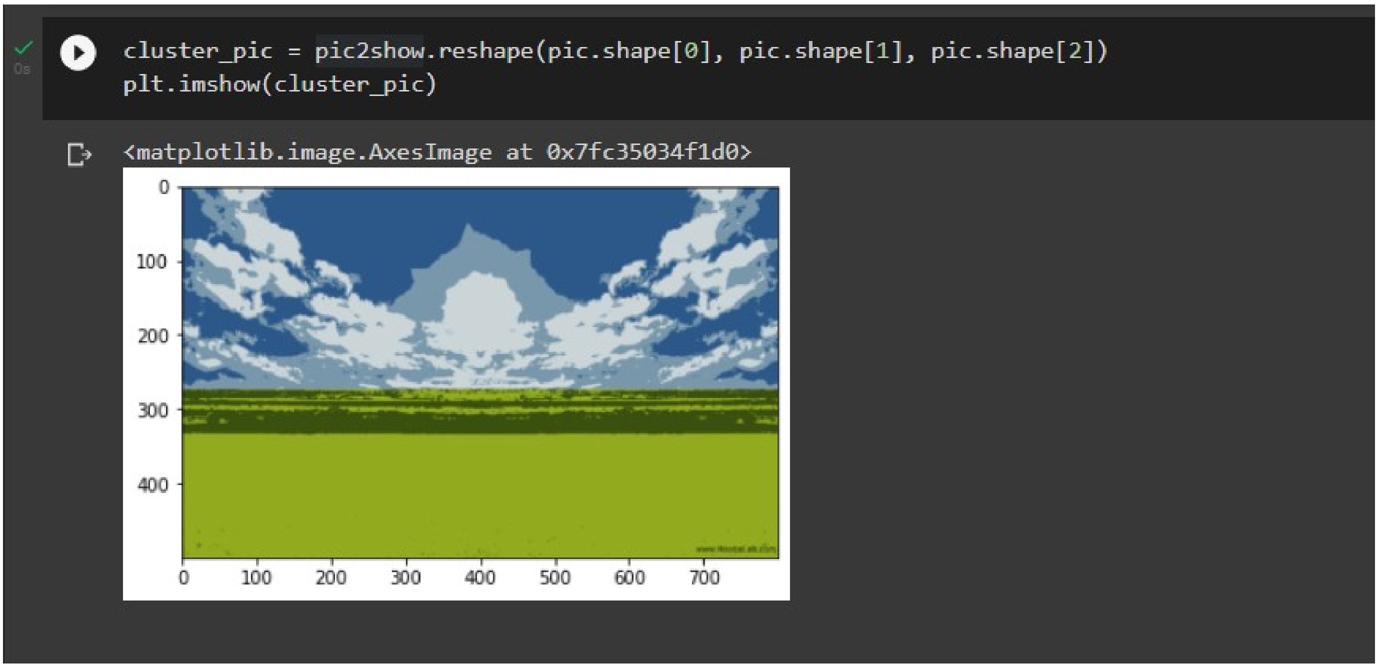
```
(400000, 3)
```

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



```
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)
```



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

```
<matplotlib.image.AxesImage at 0x7fc35034f1d0>
```