# Image_Filtering

January 29, 2019

```
In [127]: %matplotlib inline
          import numpy as np
          import matplotlib.pyplot as plt
```

We need import **matplotlib** for plotting and **numpy** for array operations.

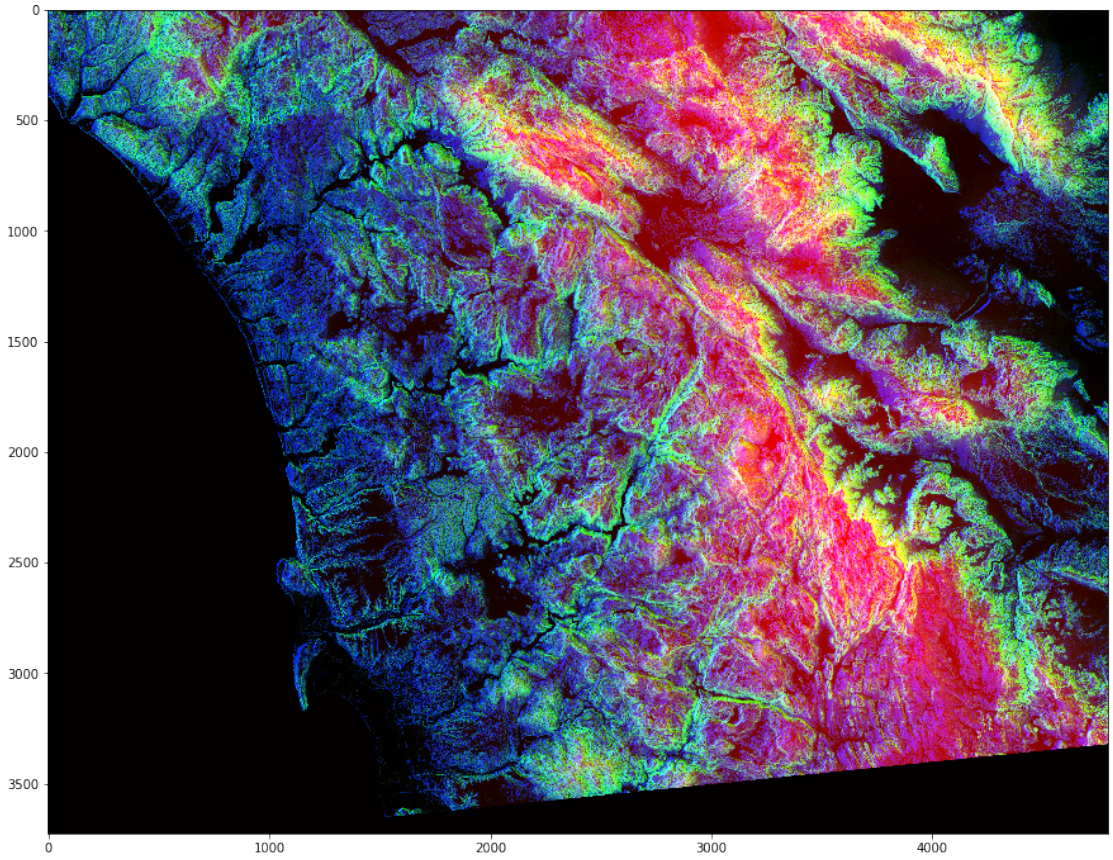
```
In [128]: from skimage import data
          import imageio

          photo_data = imageio.imread('/home/mert/Desktop/Data_Science/Week-3-Numpy/filter.jpg
```

And also we need import **data** from **skimage** and **imageio** for read and manipulate pictures

```
In [129]: plt.figure(figsize=(15,15))
          plt.imshow(photo_data)

Out[129]: <matplotlib.image.AxesImage at 0x7fc20cf3b4e0>
```




We use ***ply.figure()*** for resize image and ***plyt.imshow()*** for showing image on the screen

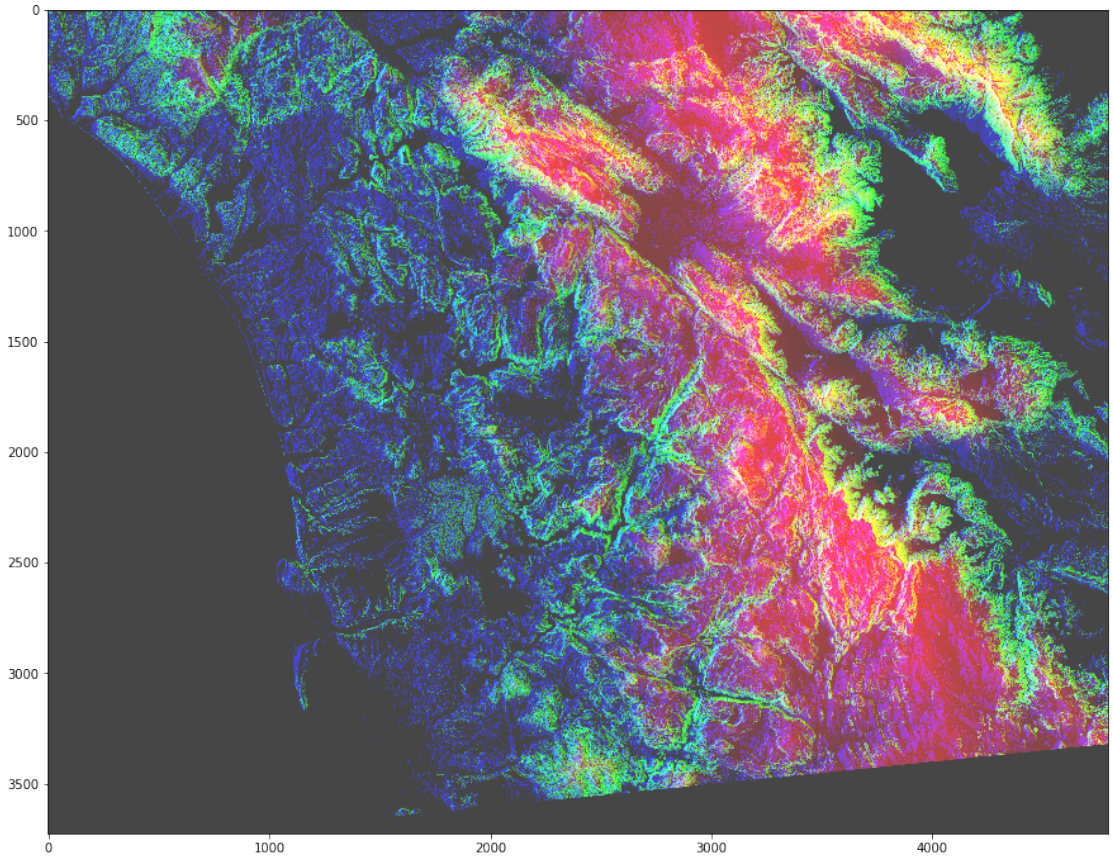
```
In [130]: photo_data = imageio.imread('/home/mert/Desktop/Data_Science/Week-3-Numpy/filter.jpg
          print("Shape of photo_data:", photo_data.shape)
          low_value_filter = photo_data < 100
          print("Shape of low_value_filter:", low_value_filter.shape)
```

```
Shape of photo_data: (3725, 4797, 3)
Shape of low_value_filter: (3725, 4797, 3)
```

Import the picture again and make a ***low_value_filter*** which pixel values (rgb) less than 100

```
In [131]: photo_data[low_value_filter] = 70
          plt.figure(figsize=(15,15))
          plt.imshow(photo_data)
```

```
Out[131]: <matplotlib.image.AxesImage at 0x7fc20ce947b8>
```

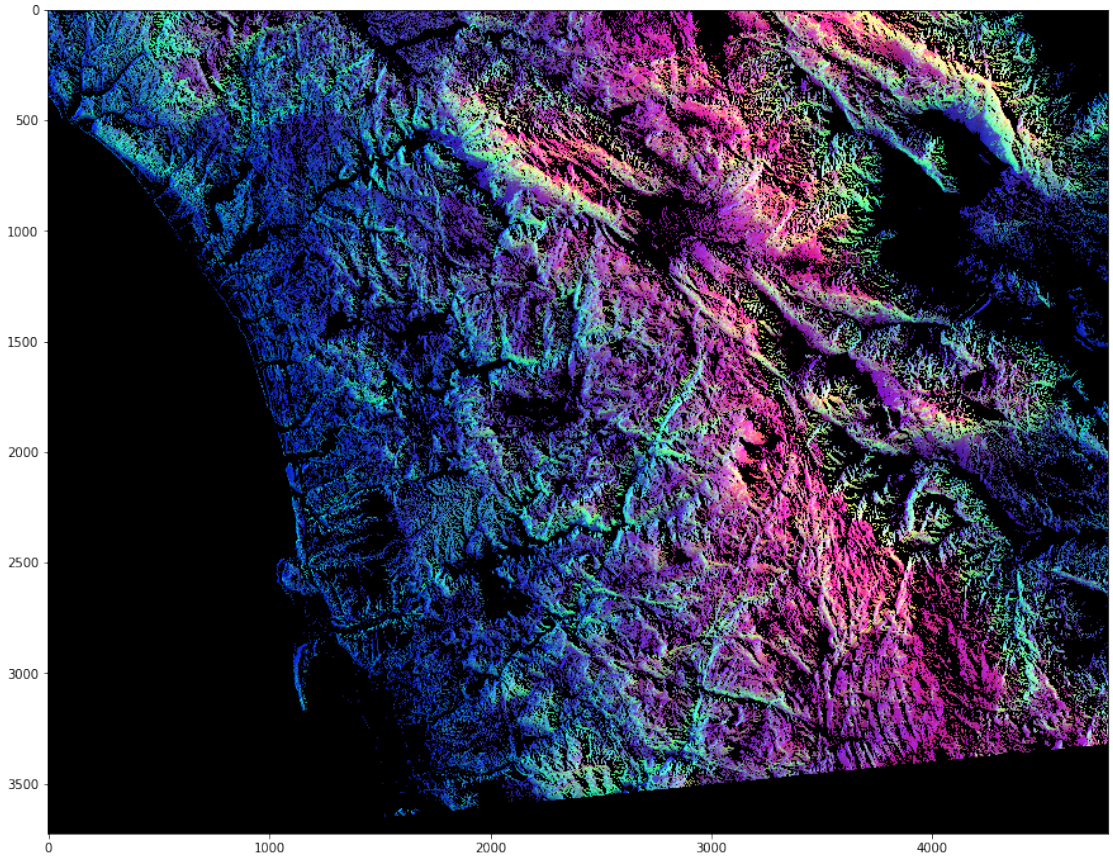



Using ***low_value_filter*** I reassign all the pixel values which are less than 100 to 70.
And we have a brighter image on the screen

```
In [132]: photo_data = imageio.imread('/home/mert/Desktop/Data_Science/Week-3-Numpy/filter.jpg
          blue_mask  = photo_data[:, : ,2] < 100

          photo_data[blue_mask] = 0
          plt.figure(figsize=(15,15))
          plt.imshow(photo_data)

Out[132]: <matplotlib.image.AxesImage at 0x7fc20cc19cc0>
```



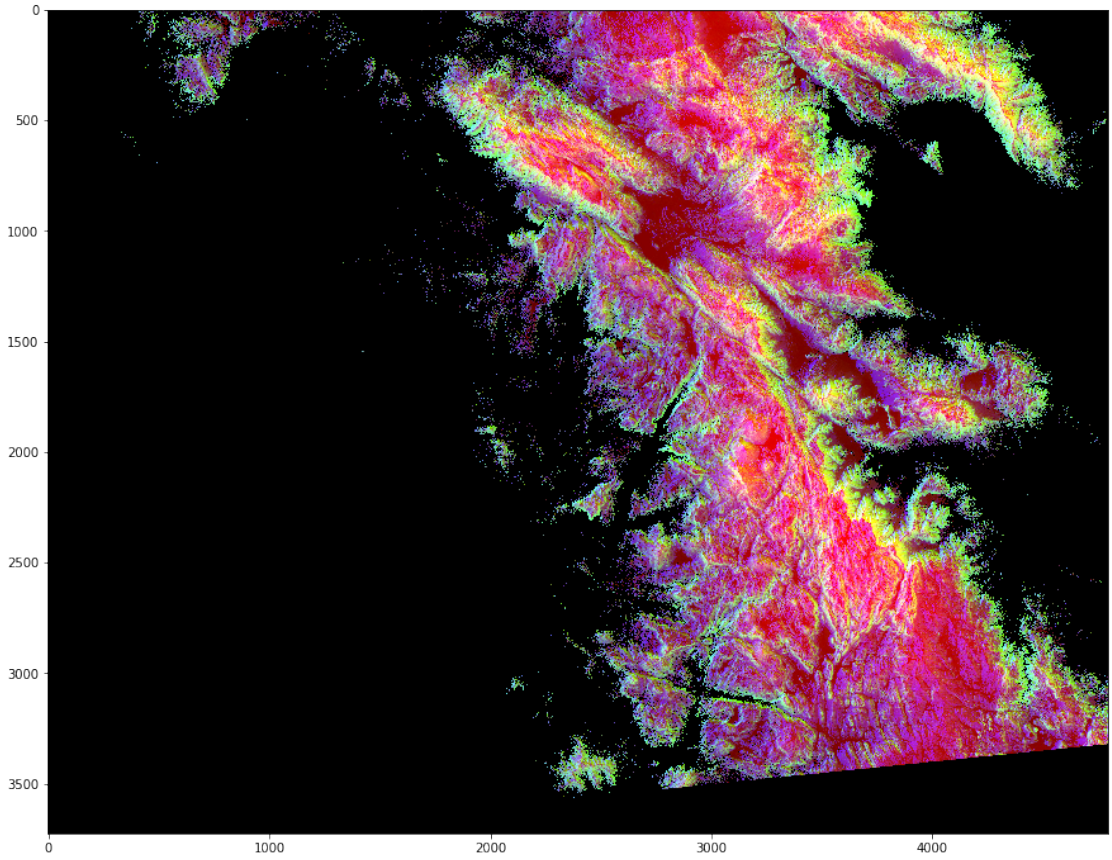

For now I use **blue_mask** which is *photo_data[:, : ,2] < 10*

Here, I pick all the row and columns which are blue pixels **(0:R, 1:G, 2:B)** and less than 100 with using index slicing and I reassign all the blue pixel values which are less than 100 to 0. 0 means all the pixels are being black.

```
In [133]: photo_data = imageio.imread('/home/mert/Desktop/Data_Science/Week-3-Numpy/filter.jpg
          red_mask   = photo_data[:, :,0] < 100

          photo_data[red_mask] = 0
          plt.figure(figsize=(15,15))
          plt.imshow(photo_data)

Out[133]: <matplotlib.image.AxesImage at 0x7fc20cb78e10>
```




For now I use **red_mask** which is *photo_data[:, : ,0] < 100*

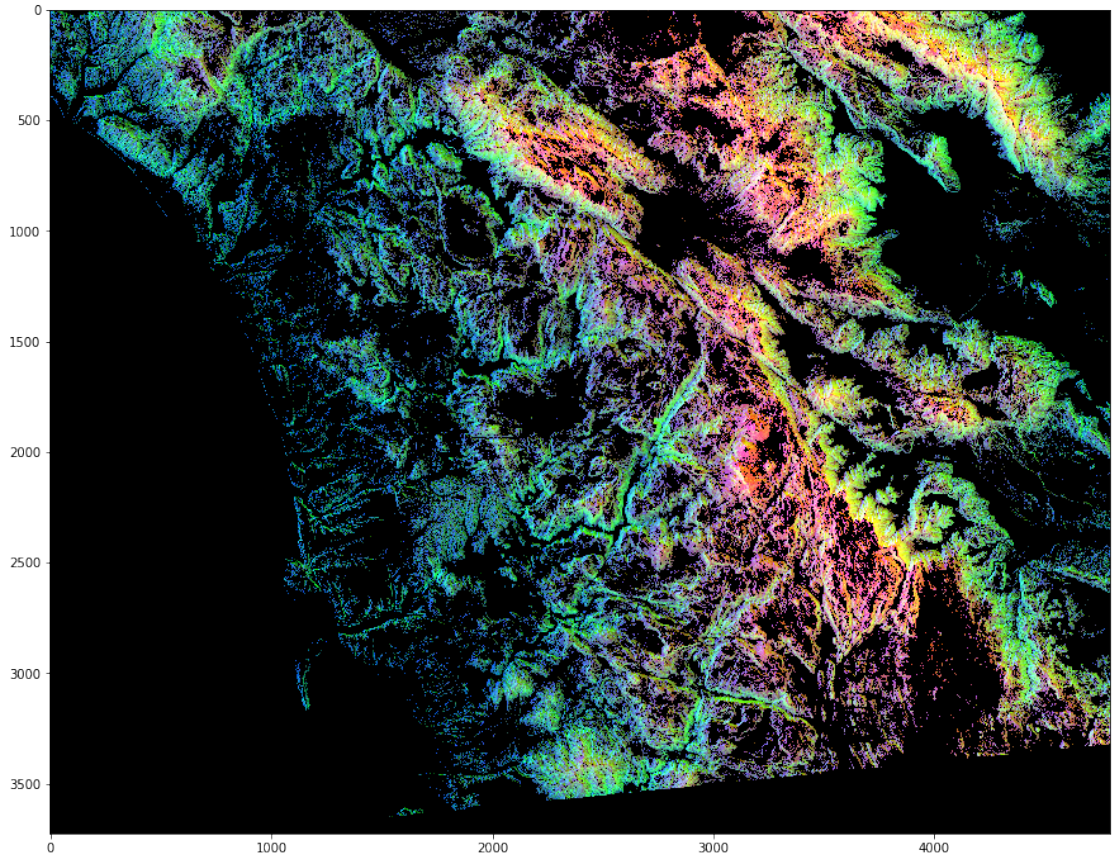
Here, I pick all the row and columns which are red pixels and less than 100 with using index slicing and I reassign all the red pixel values which are less than 100 to 0. 0 means all the pixels are being black.

So we have a **darker** image which *devoid of red*.

```
In [134]: photo_data = imageio.imread('/home/mert/Desktop/Data_Science/Week-3-Numpy/filter.jpg
          green_mask = photo_data[:, : ,1] < 75

          photo_data[green_mask] = 0
          plt.figure(figsize=(15,15))
          plt.imshow(photo_data)

Out[134]: <matplotlib.image.AxesImage at 0x7fc20cb53f28>
```



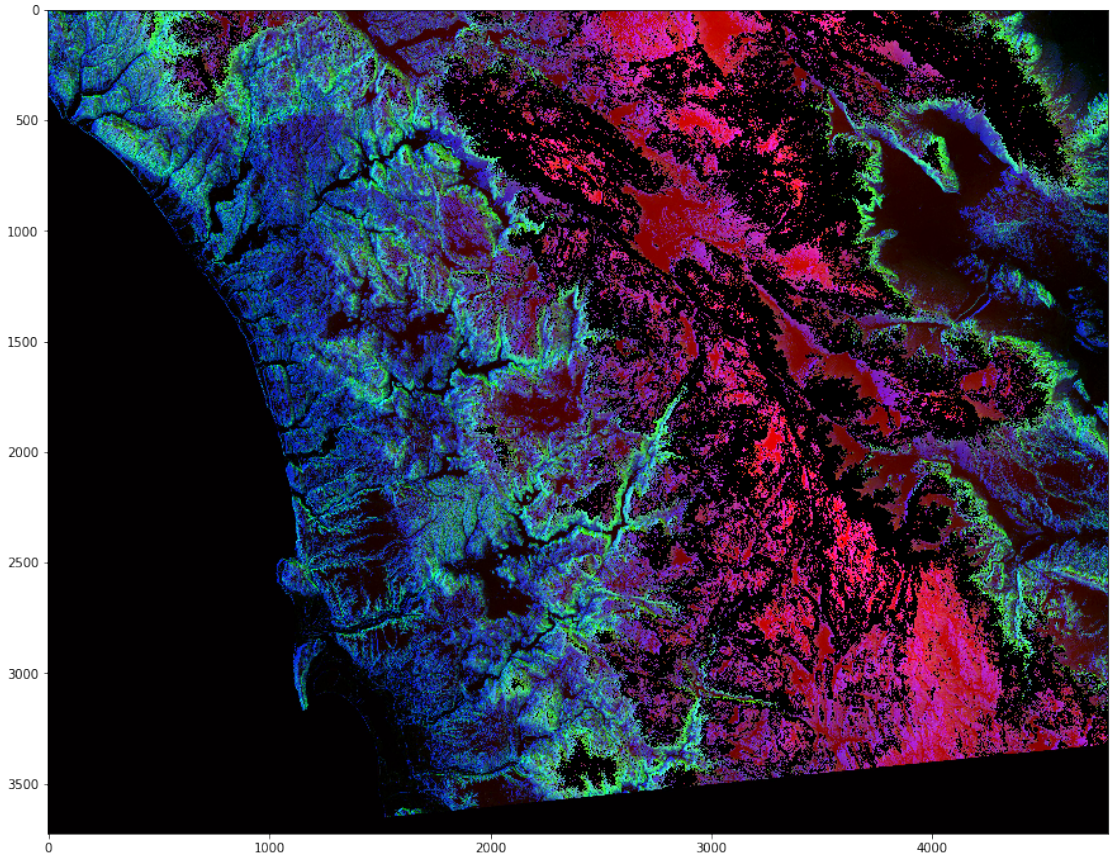

For now I use **green_mask** which is *photo_data[:, : ,1] < 75*

Here, I pick all the row and columns which are green pixels and less than 75 with using index slicing and I reassign all the green pixel values which are less than 75 to 0. 0 means all the pixels are being black.

```
In [135]: photo_data = imageio.imread('/home/mert/Desktop/Data_Science/Week-3-Numpy/filter.jpg
          red_mask   = photo_data[:, : ,0] > 100
          green_mask = photo_data[:, : ,1] > 75
          blue_mask  = photo_data[:, : ,2] < 90

          final_mask = np.logical_and(red_mask, green_mask, blue_mask)
          photo_data[final_mask] = 0
          plt.figure(figsize=(15,15))
          plt.imshow(photo_data)

Out[135]: <matplotlib.image.AxesImage at 0x7fc20cab7780>
```




Here, I merged all the masking features with using ***np.logical_and*** function. And we have a mixed masking picture on the screen.