***Real-world problem selection:***

*The problem that I have chosen is the image compression using python. The motivation to choose this as a project is to understand matrix operations and to get a better hold of using numpy arrays in python.*

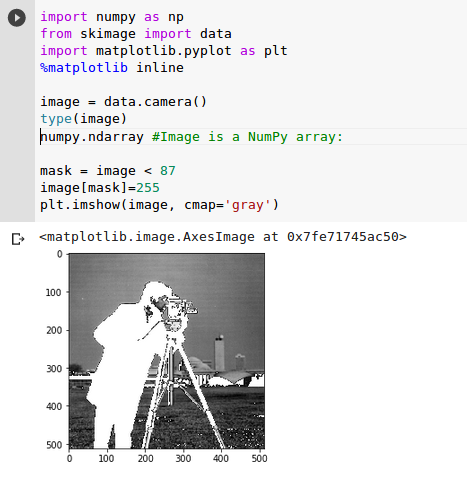
***Progress report:***

*Today's world is full of data, and images form a significant part of this data. However, before they can be used, these digital images must be processed—analyzed and manipulated in order to improve their quality or extract some information that can be put to use.*

Common image processing tasks include displays; basic manipulations like cropping, flipping, rotating, compression. Python is an excellent choice for these types of image processing tasks due to its growing popularity as a scientific programming language and the free availability of many state-of-the-art image processing tools in its ecosystem. It has many support libraries numpy and matplotlib are one of them. In this project these two are used widely:

* Numpy is one of the core libraries in Python programming and provides support for arrays. An image is essentially a standard NumPy array containing pixels of data points. Therefore, by using basic NumPy operations, such as slicing, masking, and fancy indexing, you can modify the pixel values of an image.
* *matplotlib is an open source Python package that works with* *NumPy* *arrays. It implements algorithms and utilities for use in research, education, and industry applications.*

*Demonstration:*



K-means for compression

**K-means clustering**

The algorithm used for image compression is K-means clustering algorithm.It is an optimization algorithm to find ‘k’ clusters in the given set of data points.

**Technique used:**

Initially, it randomly assigns k-cluster centers and then based on distance metric (Euclidean distance) it aims to minimize within cluster sum of squared distance of the data points from the cluster center.

**There are two steps in k-means clustering algorithm:**

1. Assignment step – Each data point is assigned to the cluster whose center is nearest to it.
2. Update step – New means (centroids) are calculated from the data points assigned to the new clusters.

**Value of K used:** 16

**Number of iterations for updating of centroids**: 30

|  |  |
| --- | --- |
| Original image | Compressed Image |
| Size: 113KB | Size: 29.3KB |
| File name: orig\_large | File name: updated\_large |
|  |  |

**Compression Factor:**

If we represent the image with these reduced (16) colors it will be reduced up to 4 times of its original size. The size of our large file was 116,355 bytes and small file was of 30,103 bytes. Size of small size is almost of 4 times of the large file.