@CopyRight Nitesh Rawal # 👇

####

Image Compression with K-Means Clustering Project is Build Using Python , K-mean and scikit-learn:.:

1. Explain the steps involved in k-means clustering.
2. Apply k-means clustering with scikit-learn and Python to compress images.
3. Create interactive, GUI components in Jupyter notebooks using Jupyter widgets

Structure of project

It is divided into 4 parts:

1. Course Overview: This introductory reading material.
2. Image Compression with K-Means Clustering Project: This is the hands on project that we will work on in Rhyme.
3. Ungraded Quiz: Check your understanding of the concepts learned in the hands on project with this ungraded quiz.
4. Graded Quiz: This is the final assignment that you need to pass in order to successfully complete the course and earn a Course Certificate.

**Project Structure** # 🛠️ **and used technologies**

project on Image Compression with K-Means Clustering is divided into the following tasks:

Task 1: Introduction and Overview

* Introduction to the image compression problem with machine learning.
* See a demo of the final product you will build by the end of this project.
* Introduction to the Rhyme interface.
* Import essential modules and helper functions from [NumPy](https://numpy.org/" \t "_blank), [Matplotlib](https://matplotlib.org/" \t "_blank), [scikit-learn](https://scikit-learn.org/stable/), and [Jupyter Widgets](https://ipywidgets.readthedocs.io/en/latest/" \t "_blank).

Task 2: Data Pre-processing

* Import images from a local directory and store them as numpy arrays.
* Explore the image attributes.
* Normalize the pixel values and unroll the arrays into vectors.

Task 3: Visualizing the Color Space using Point Clouds

* Visualize the set of pixels from the original image as a two 2-D point clouds in color space.

Task 4: Visualizing the K-Means Reduced Color Space

* Understand the math and steps involved in the k-means clustering algorithm.
* Perform k-means clustering with scikit-learn's [MiniBatchKMeans](https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html" \t "_blank) to reduce the number of possible colors in the image from over 16 million to 16.
* Compare and contrast the color space of the original image with that of the k-means compressed image.

Task 5: Creating Interactive Controls with Jupyter Widgets

* Use the [interact](https://ipywidgets.readthedocs.io/en/latest/examples/Using%20Interact.html) function to automatically create UI controls for function arguments.
* Define an argument to control the value of **k**using a slider.
* Define an argument to pick any image from a specified directory.

Task 6: K-means Image Compression with Interactive Controls

* Ensures that k-means image compression is performed only on the slider widget's mouse release events.
* Repurpose the data pre-processing and k-means clustering logic from previous tasks to operate on images of your choice.
* Visualize how the image changes as the number of clusters fed to the k-means algorithm is varied.

.

Image Compression with K-Mean Clustering **#MLproject** **#completed** Unsupervised Technique based on 16 million colors represented as 16 colors to compress the image

That involved k-means clustering with scikit-learn and Python to compress images by Creating interactive, GUI components in Jupyter notebooks using Jupyter widgets

where i have used  essential modules and helper functions from NumPy, Matplotlib, scikit-learn, and Jupyter Widgets.

staring from **#Datapreprocessing** to Import images from a local directory and store them as numpy arrays. then

Visualize the set of pixels from the original image as a two 2-D point clouds in color space. and Perform k-means clustering with scikit-learn's  to reduce the number of possible colors in the image from over 16 million to 16

also Created an Interactive Controls with Jupyter Widgets.















