PROJECT REPORT - COLOR PALETTE EXTRACTOR

Xinwan Wang & Parthivi Varshney
Course : CS6200 - Data Mining
Credits: Prof. Karl, Ni

Introduction

Our project focuses on extracting the color palette of an image using KMeans clustering algorithm. A color palette is a predefined set of colors chosen for a particular project, design, or brand. It usually includes a range of colors that complement each other and are used together to create a cohesive look and feel.

On the other hand, colors are individual hues that can be used on their own or in combination with other colors. They are specific shades on the color spectrum, each with its own unique properties and associations.

The basic idea behind this project is to treat each pixel in the image as a data point with a three-dimensional feature vector representing the pixel's RGB values. Then, the k-means algorithm is applied to cluster the data points into k clusters, where k is a pre-defined parameter. The resulting cluster centers represent the dominant colors in the image.

Background

Our color palette extractor project has several potential benefits, including saving time and effort compared to manual color identification methods and reducing subjectivity in the selection process. For designers and artists, this project can provide quick and accurate identification of the most dominant colors in an image, facilitating the creation of color palettes for new designs. Fashion industry professionals can use the algorithm to identify trending colors for a particular season and incorporate them into their products. In advertising, the algorithm can be used to analyze the theme colors for companies, leading to more effective design of advertisements.

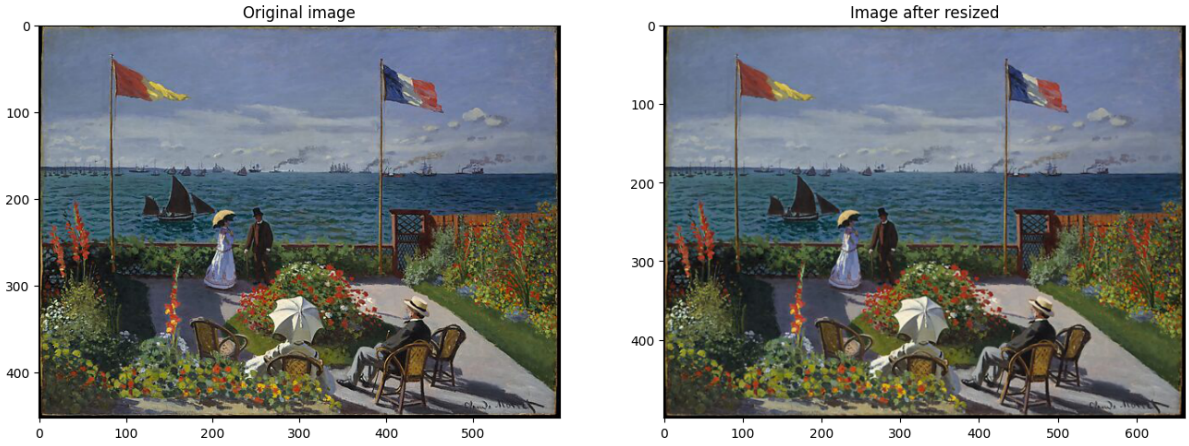
In addition to these benefits, the algorithm can also be used to segment satellite images and identify surface features. Different colors typically belong to different features, which can be converted into groups and identified as various surfaces like water, vegetation, etc.

Overall, developing a project for identifying the most dominant colors in an image is a valuable data mining application that can provide valuable insights into color usage patterns in various industries. This project can improve the efficiency and effectiveness of color identification and facilitate informed decision-making for resource planners, environmentalists, designers, artists, fashion industry professionals, and advertisers.

Image and Data Preprocessing

To ensure our extractor has efficient performance, we decided to do some preprocessing with the input image. We resize the image, reshape it and extract pixel data.

Resizing the input image is to ensure that in the future steps, the process won't be too slow due to the big image and dataset. Here's an example of resizing a painting by French artist Claude Monet:



Our project is set to resize the image to width of 500 pixels, and proportionally resize its height. We are using the cv2.resize() function to resize the original image.

The function first calculates the scaling factor for each dimension (width and height) using the desired output size and the current size of the input image. It then resizes the input image using the calculated scaling factors.

By default, cv2.resize uses a method called bilinear interpolation, which calculates the new pixel value based on the weighted average of the surrounding four pixels.
Since the original image is relatively small, our project enlarges it actually.

Extracting pixel data is to transfer pixels into machine readable data, which is RGB color values from the RGB color model. A color in the RGB color model is described by indicating how much of each of the red, green, and blue is included. The color is expressed as an RGB triplet (r,g,b), each component of which can vary from zero to a defined maximum value of 255. In further steps, we will manipulate each pixel's RGB values.

We imported python packages numpy, pandas, opencv and matplotlib to help us implement preprocessing code in Python.

Approach

- Elbow Method: find the optimal value for k

  One of the most common methods to find the optimal value for k is the Elbow Method. In cluster analysis, the elbow method is a heuristic used in determining the number of clusters in a data set. The method consists of plotting the explained variation as a function of the number of clusters and picking the elbow of the curve as the number of clusters to use. The same method can be used to choose the number of parameters in other data-driven models, such as the number of principal components to describe a data set. The method can be traced to speculation by Robert L. Thorndike in 1953. (Thorndike 1953, 267–276)

  The rationale behind the Elbow method is that as we increase the number of clusters, the within-cluster sum of squares (WCSS) decreases, because each point is assigned to a cluster whose centroid is closer to it. However, after a certain number of clusters, the improvement in WCSS becomes marginal, because adding more clusters does not result in a significant reduction in the WCSS. The optimal number of clusters is therefore the value of k at which the marginal benefit of adding more clusters starts to diminish, which corresponds to the elbow point in the WCSS plot.

  In our project, an elbow plot will be presented as a reference. Users can customize the input k value.

- K-means Clustering: find the k most dominant colors

  K-means clustering is one of the most popular unsupervised machine learning algorithms. It is a clustering algorithm that partitions a dataset into K clusters, where K is a predefined number. The goal of the algorithm is to minimize the sum of the squared distances between each data point and its assigned centroid within the cluster.

  In our project, we utilize k-means clustering to find the most dominant colors in the input image, which are represented by the “centroids”. The algorithm works by assigning each data point to the cluster whose centroid is closest to it. This step is based on the distance metric used, scikit-learn by default uses Euclidean distance.The mean of the data points is then calculated in each cluster to obtain the new centroid.

These steps are repeated until convergence is achieved, which occurs when the assignments of data points to clusters no longer change or when a maximum number of iterations is reached.

We then translate centroids data to visible color palette, regenerate the input image with dominant colors and hex code if needed.

Implementation Specifics

With data of each pixel, we then stepped into the implementation of the color extractor function. We imported sci-kit learn package and leverage KMeans function in it to find k centroids. They are 3-element arrays stored in one numpy array, representing their RGB code. For user convenience, we transfer them into hex color code.

We also calculated the percentage of each dominant color. In our project, we present these colors, the color code and the percentage in graphs with matplotlib, a comprehensive library for creating static, animated, and interactive visualizations in Python.

With all generated data, we are able to regenerate the image. The method is to replace every pixel with the centroid color which it was classified to. Thus the regenerated image is only made of dominant colors that we found previously. We used the opencv library in this step.

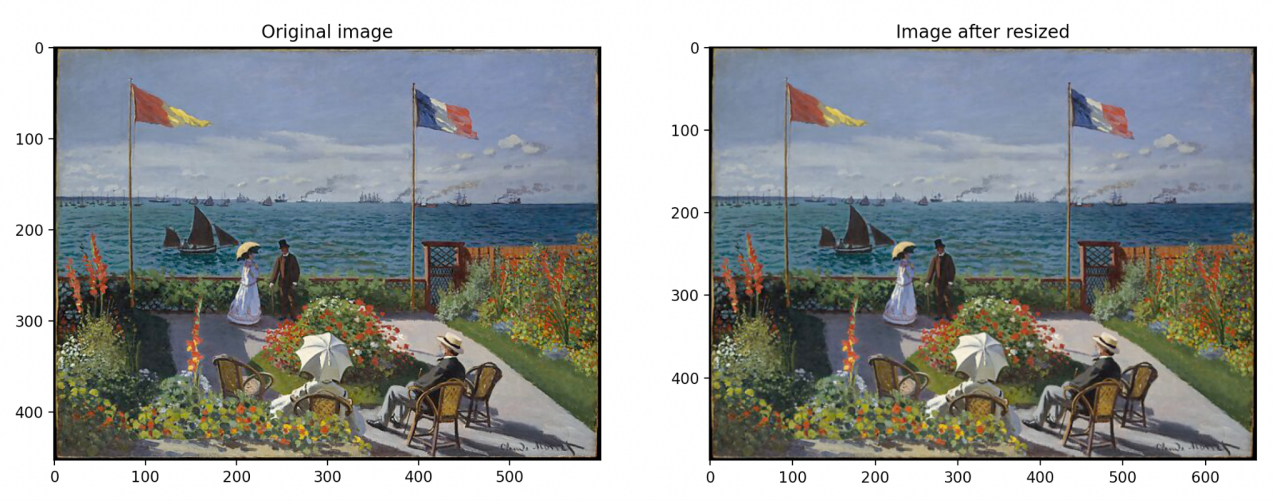
We implemented our project in Google Colab at first. You can find the file in our GitHub. Then we split each part into 3 python .py files stored in the extractor folder for further use.

Contribution

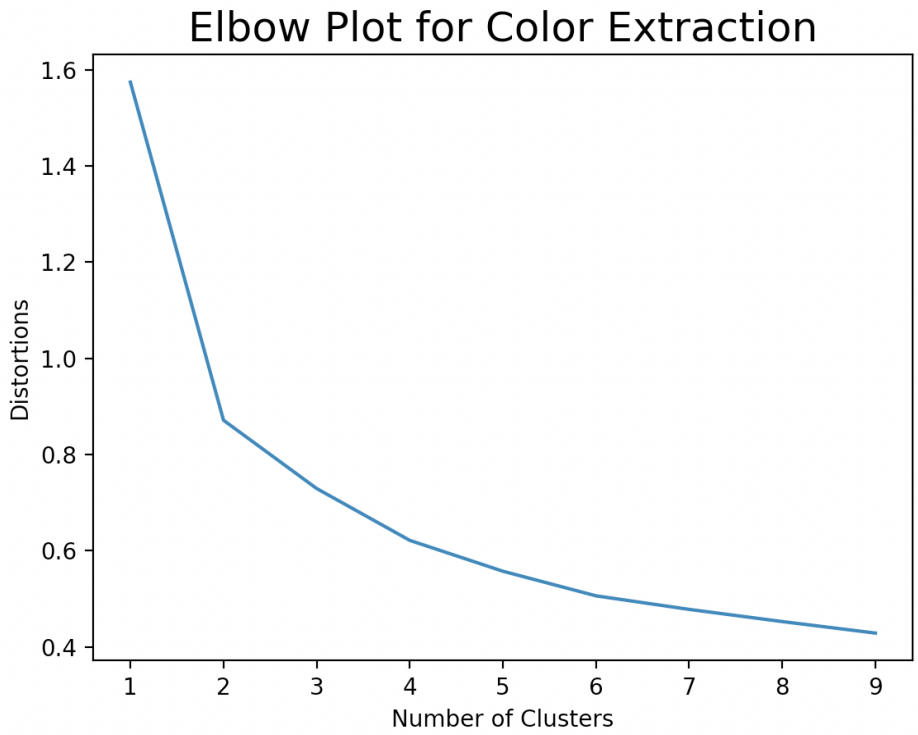
Parthivi Varshney - Image resizing and reshaping, extracting rgb values, plotting resized image, Elbow plot, Converting extracted centroids to hex values and plotting the extracted colors with hex values.
Xinwan Wang - Applied K_mean, Worked on calculating percent of each extracted color, Plotting the percentages, Regenerating new image with extracted color and plotting the regenerated image.

Results

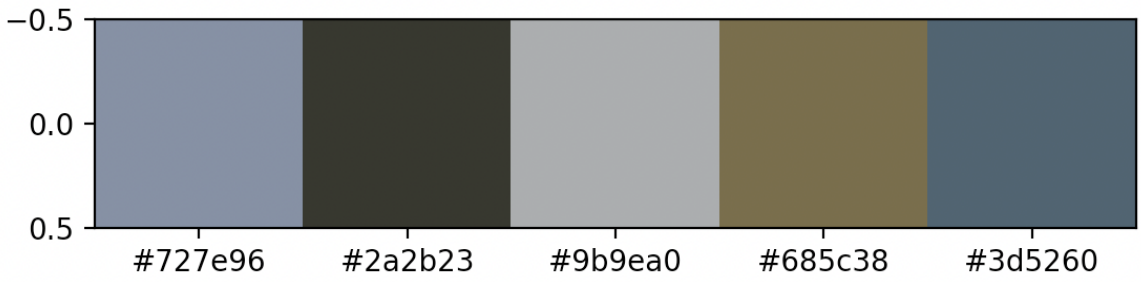
- Resized Image:



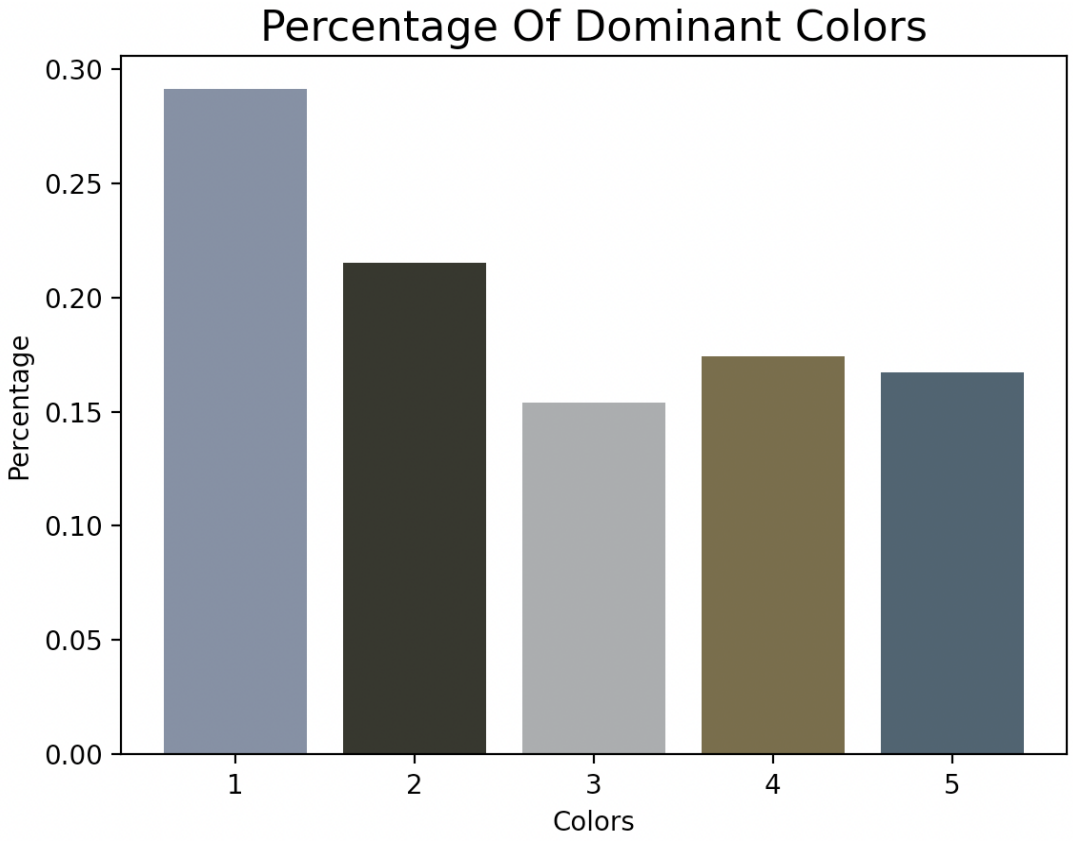
- Elbow Plot:



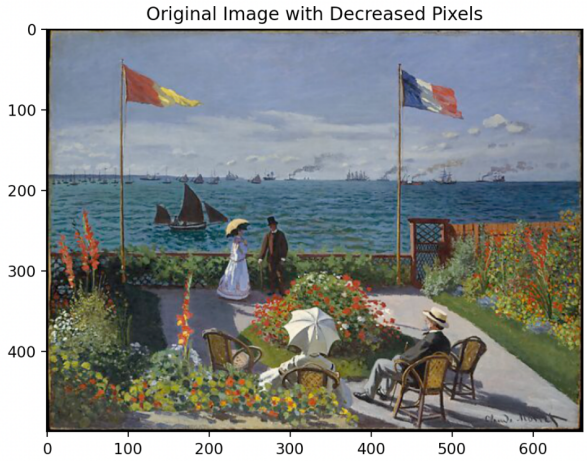
- Extracted colors with hex values:

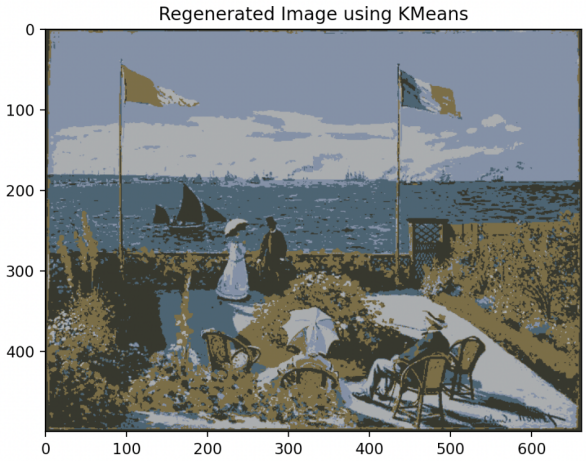


- Extracted colors with their percentage composition:



- Regenerated Image:





Conclusion

In conclusion, the project successfully implemented the K-means algorithm to extract the most dominant colors in an image. The project's implementation was efficient and accurate, and can handle large datasets and complex images. The extracted colors can be used for a variety of applications, such as creating a color palette for graphic design, analyzing color trends in images, or identifying the primary colors in a photograph. We also found that increasing the number of clusters results in a greater resemblance between the generated image and the original image.

Overall, the project provided valuable insights into machine learning and image processing, and demonstrated the power of the K-means algorithm in identifying dominant colors in images. The project can serve as a useful tool for anyone working with visual media and interested in exploring color analysis techniques.

References

Thorndike, Robert L. 1953. *Who Belongs in the Family?* N.p.: Psychometrika.