Image Labelling

Artificial Intelligence

Departament de Ciències de la Computació

Universitat Autònoma de Barcelona

Authors:

George-Vlad Manolache 1718986

Petru Balan 1719379

1. Introduction

In this project, we tackle the task of image labeling, essential for applications like image search engines and object recognition in computer vision. Our approach combines two methods: K-means for color classification and K-nearest neighbors (KNN) for shape classification.

K-means Algorithm

K-means is an unsupervised learning algorithm that clusters data into K distinct groups based on similarity. We use K-means to identify and classify the colors in clothing images by minimizing the within-cluster sum of squares (WCD).

KNN Algorithm

K-nearest neighbors (KNN) is a supervised learning algorithm used for classification. It labels images by comparing them with a set of pre-labeled training images and assigning the label of the closest match based on a defined distance metric. KNN is effective for recognizing shapes in clothing items.

Objective

The goal of this practicum is to develop and evaluate a combined image labeling system that accurately assigns color and shape labels to clothing images. By integrating K-means and KNN, we aim to improve the accuracy and efficiency of the labeling process.

In this final part, we focus on evaluating our methods through qualitative and quantitative analyses and introduce enhancements to improve the performance of K-means and KNN. This practicum provides valuable insights into image labeling techniques and practical skills in applying machine learning algorithms to real-world problems, aiming to achieve high labeling accuracy for computer vision applications.

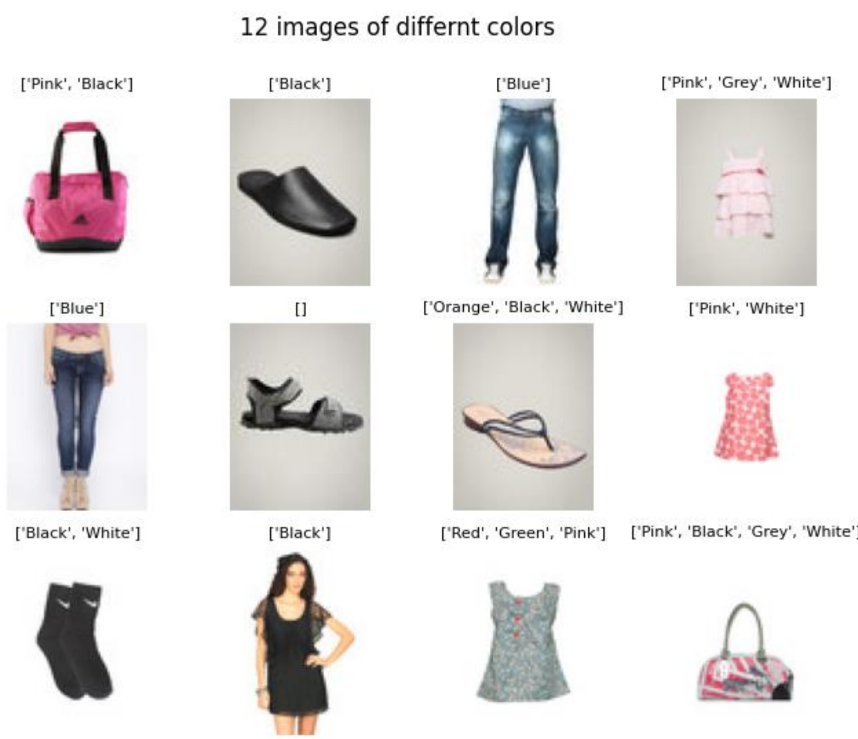
2 Implemented Methods of Analysis

2.1. Qualitative Analysis Methods

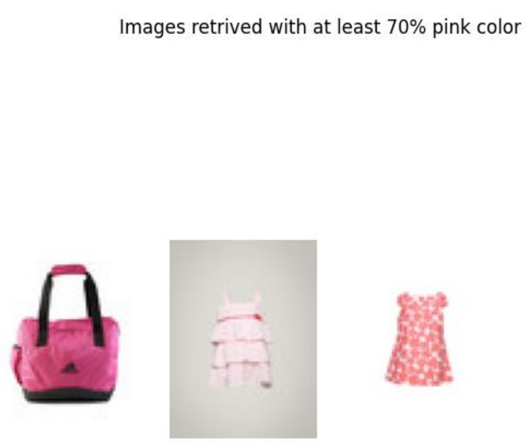
To qualitatively evaluate the performance of our K-means and KNN classifiers, we implemented several functions to visually inspect the results of our image labeling.

Retrieval-by-Color

- **Functionality**: This function retrieves images based on color tags assigned by the K-means algorithm. Given a query color, it returns all images that contain this color. A percentage of the query color can also be specified.
- **Visualizations and Results**: We performed searches for various colors (e.g., "pink", "blue") and obtained the following results:

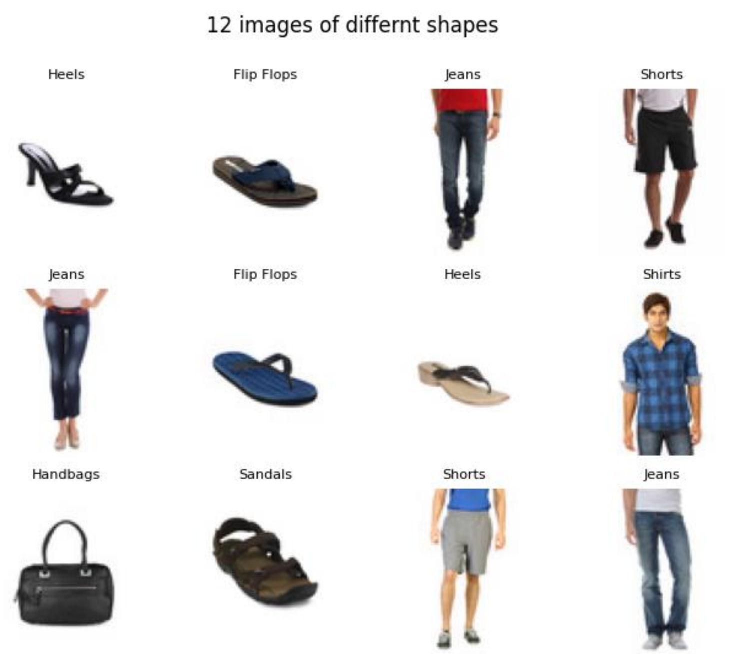


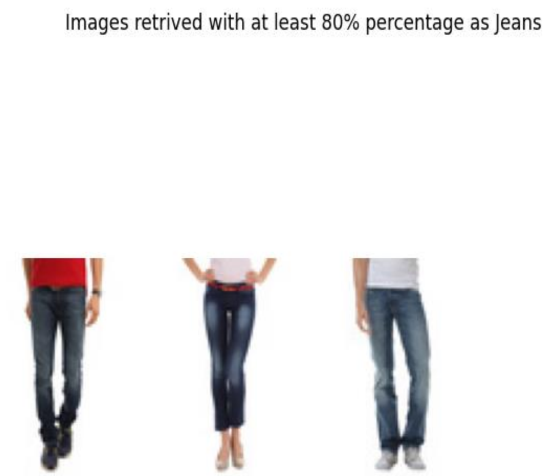
(Obs: It seems to be an annotation mistake in the data since label 6 has no color)

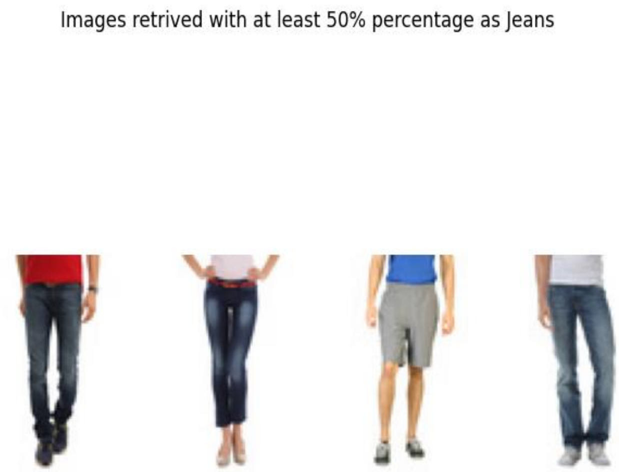


Retrieval-by-Shape

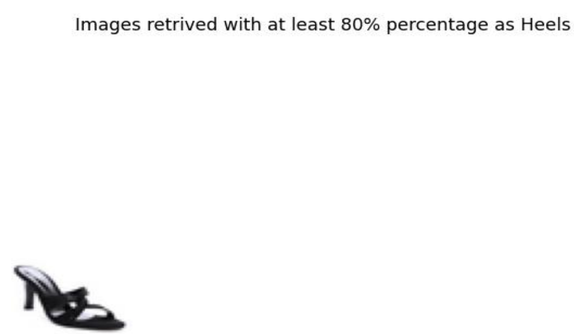
- **Functionality**: This function retrieves images based on shape tags assigned by the KNN algorithm. Given a query shape (e.g., "shirt", "pants"), it returns all images that match this shape. A percentage of the query shape can also be specified.
- **Visualizations and Results**: We performed searches for various colors (e.g., "pink", "blue") and obtained the following results:







(Obs: We can see that if we select a low percentage of retrieval our function will retrieve this pair of shorts as jeans, because the baseline model predicted this label jeans, with a confidence of 57% as jeans and 43% as shorts, misclassifying it in reality)

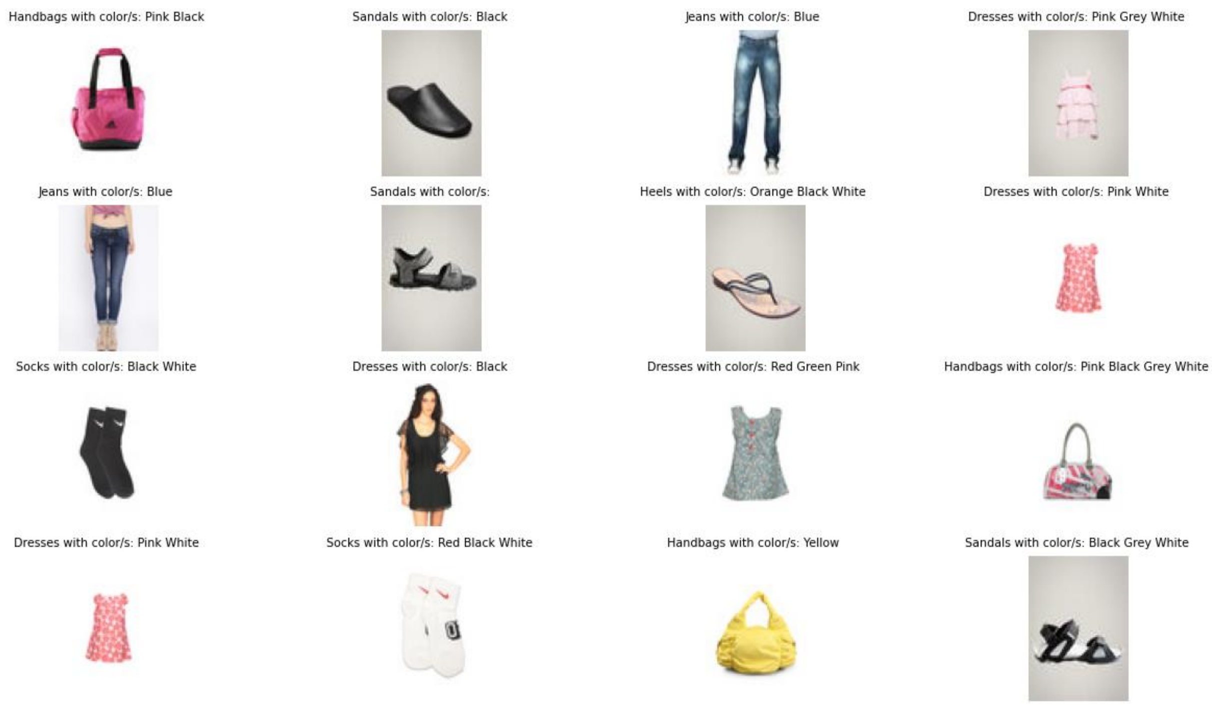


Here we can see that if we select a percentage of retrieval that is too high, our function will retrieve only one pair of Heels, even if our model made the right prediction about the second item, with a confidence of 43% as Heels, 28,5% as Flip Flops and 28,5% as Sandals.

Retrieval-Combined

- **Functionality**: This function combines both color and shape tags to retrieve images. For example, a query for “Yellow” and one for “Handbags” returns images labeled as both "Yellow" and "Handbags". A percentage of the query color and shape can also be specified.
- **Visualizations and Results**: The combined retrieval results are presented, showing images that match both the color and shape queries:

16 images of differnt shapes and colors



Images retrieved for Black Dresses

Images retrieved for Yellow Handbags

2.2. Quantitative Analysis Methods

To quantitatively evaluate our classifiers, we implemented several functions to measure their performance using statistical metrics.

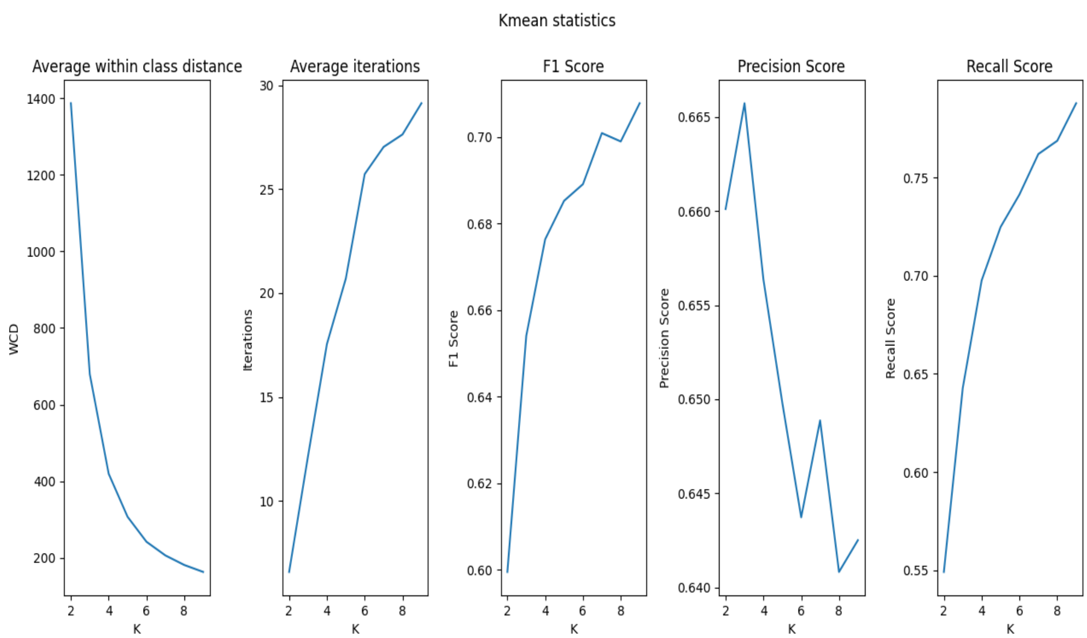
K-means Statistics

- **Functionality**: This function evaluates the K-means algorithm with the standard initialization over a range of K values (from 2 to K-max). It computes metrics such as the within-cluster distance (WCD) and the number of iterations required to converge.
- **Results**: We plotted WCD and number of iterations versus K to determine the optimal number of clusters and iterations required to converge.
- **Visualizations**: Graphs showing WCD and no. Iterations for convergence for each K value help us understand the performance of the K-means algorithm.

Get-Color-Accuracy

- **Functionality**: This function assesses the performance of the K-means color classifier with the standard initialization by comparing the predicted color labels with the Ground-Truth labels.
- **Metrics**: We measured the precision, recall and F1 score, taking into account multiple color labels per image.

Visualizations: Same as WCD and no. Iterations, we plotted graphs representing the chosen metrics according to each value of K.

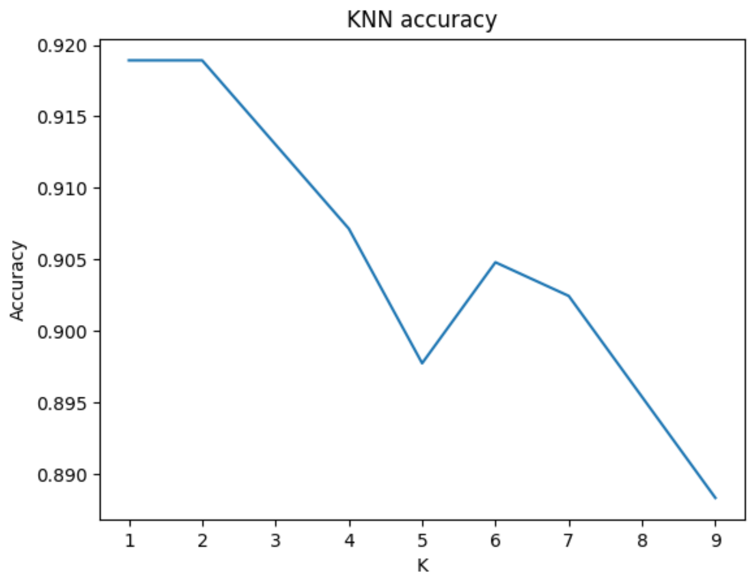


We can observe that the F1 score is 70.8%, which is decent for a simple and fast approach like K-means.

Get-Shape-Accuracy

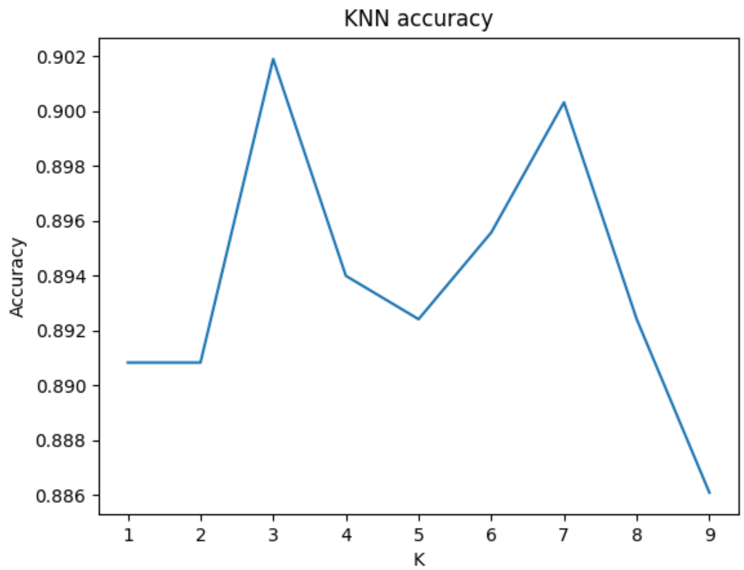
- **Functionality**: This function calculates the accuracy of the KNN shape classifier by comparing the predicted labels with the Ground-Truth labels.
- **Metrics and Results**: We computed the percentage of correctly classified shapes and plotted the results to analyze the classifier's performance

First, we obtained the following accuracy for different values of K from 1 to 10:



The accuracy is decreasing as the number of neighbors is increasing, which is something unusual. We found out that this is happening because 25% of the provided test set is also in the train set.

We decided to drop the images from the test set that are also in the train set and calculate the accuracy again. This is the result we obtained:



We can observe that the accuracy for this simple KNN baseline model is very high: 90% for K=3.

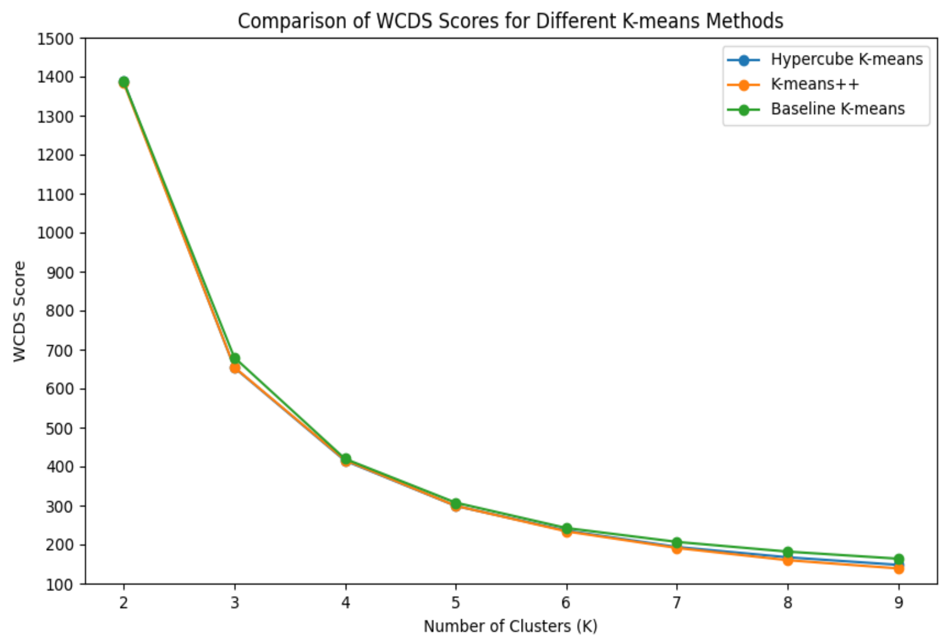
3. Improvements on K-means and KNN

3.1. K-means Improvements

Initialization Methods

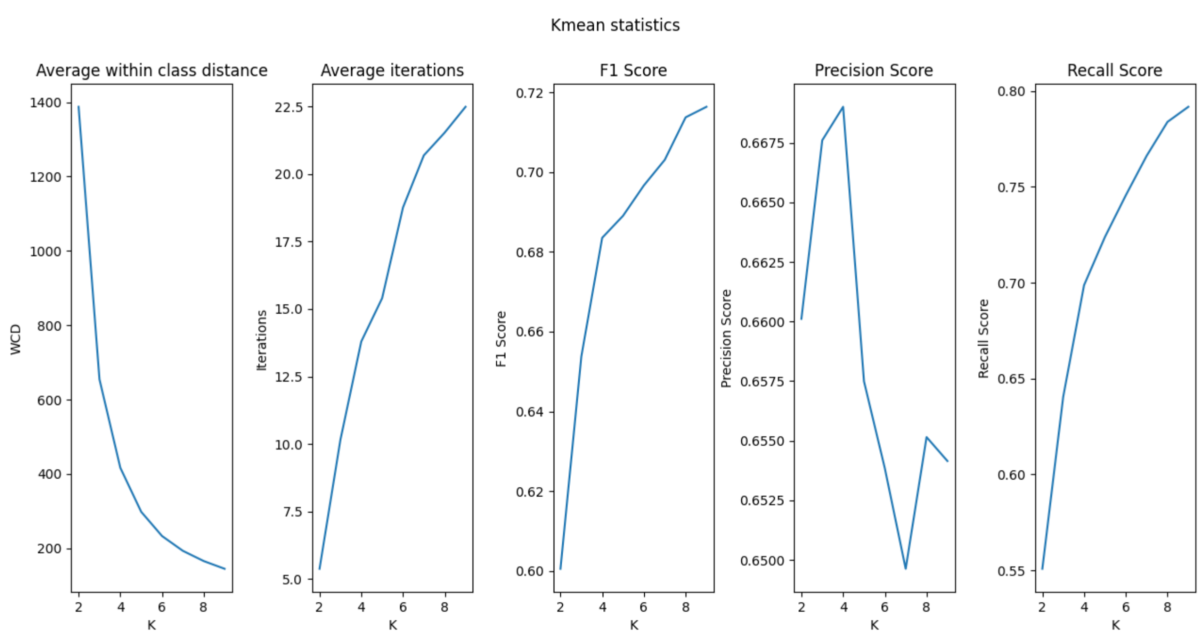
- **Improvement**: We implemented two different initialization methods for the K-means algorithm to enhance the quality of the initial centroids, leading to better clustering results and faster convergence.
    - **Hypercube Initialization**: Centroids are initialized within a hypercube that surrounds the data, ensuring a diverse spread of initial centroids.
    - **K-means++ Initialization**: This method selects initial centroids that are far apart from each other, reducing the chances of poor clustering.
- **Results and Conclusion**: Both Hypercube and K-means++ initialization methods resulted in lower within-cluster distance (WCD), improved F1 score with 1%e and faster convergence in terms of number of iterations compared to basic initialization. K-means++ performed better than Hypercube in terms of WCD and convergence, the F1 score remains basically the same with little minor changes.

- **Visualizations:** We plotted a graph representing the comparison between those three initialization methods in terms of WCD and graphs for individually results for K-means++ and Hypercube:

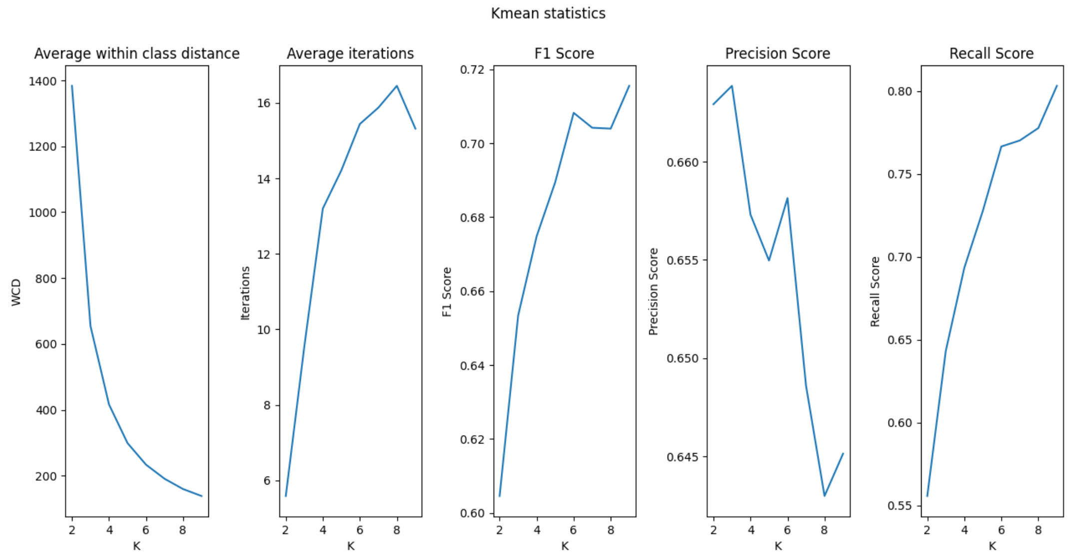


Obs: K-means performs best, followed by Hypercube at short distance

Hypercube results:



K-means++ results:

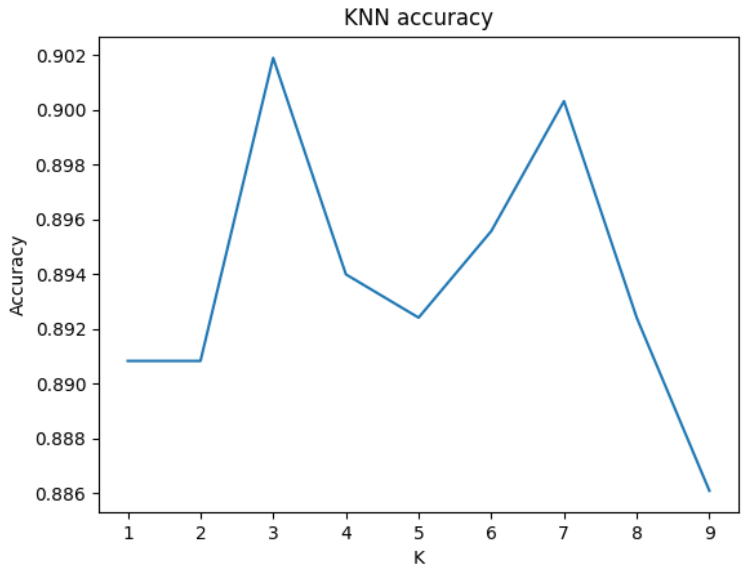


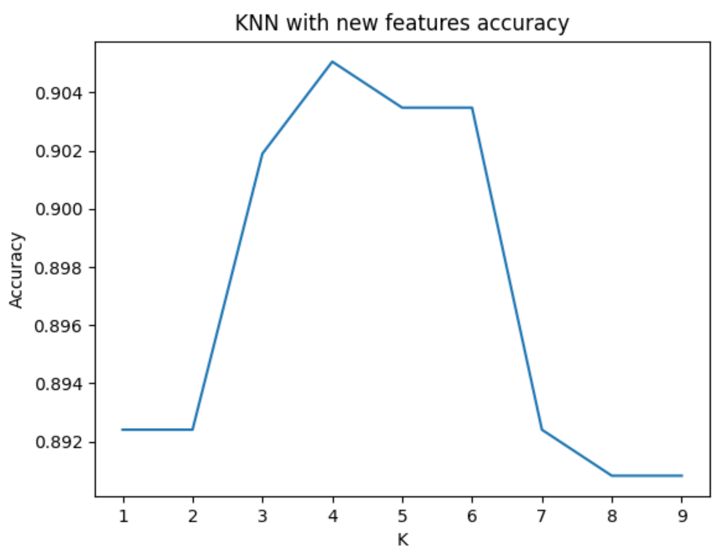
Obs: K-means++ is the fastest in terms of convergence: only 16 iterations as upper bound compared to 22.5 for Hypercube and 29 for baseline

3.2. KNN Improvements

Feature Space Modifications

- **Improvement**: We explored different feature spaces to improve the KNN shape classifier:
    - **Reduced Image Size**: Reducing the image resolution, making the algorithm faster.
    - **Custom Features**: Using features such as the mean pixel value, pixel variance, upper and lower pixel values to capture more relevant shape information.
- **Results**: Reducing the image size to a manageable resolution maintained classification accuracy while significantly improving computational efficiency. Custom features also provided a slight improvement in accuracy for higher values of K, but the upper bound for accuracy improved just a little compared to the baseline model.
- **Visualization:** Below are presented the plotted graphs for accuracy:

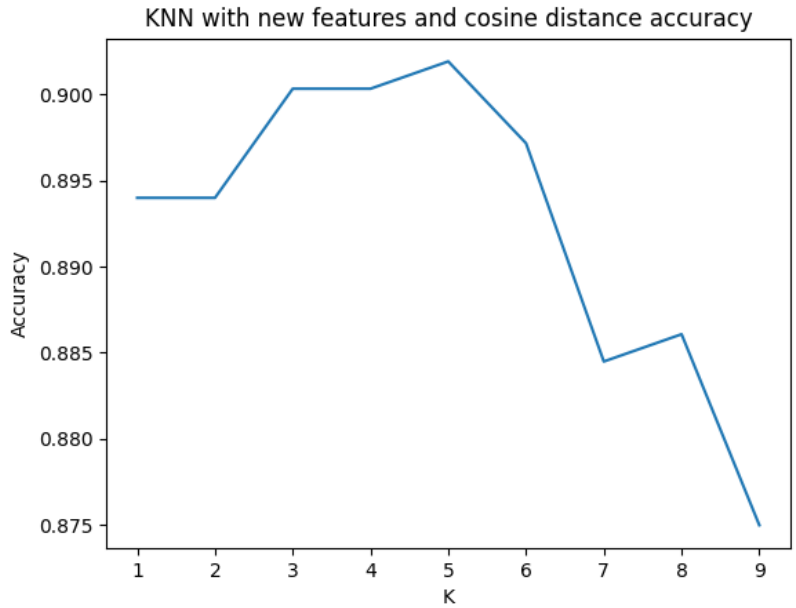




Distance Metrics

- **Improvement**: We experimented with different distance metrics for the KNN classifier:
    - **Euclidean Distance**: Standard metric used initially.
    - **Cosine Similarity**: Measures the cosine of the angle between two vectors.

- **Results**: Cosine Similarity provided a minor downgrade in accuracy, meaning that the Euclidean distance remained the most effective metric for our dataset
- **Visualizations:** Below we present the accuracy for Cosine Similarity:



Conclusion of improvements:

These enhancements to the K-means and KNN methods led to improvements in the performance and efficiency of our image labeling system, even if in some cases the results after improvements brought just small differences compared to the initial ones, they are still notable. But in case of convergence time in case of K-means++ for example, the improvements can be clearly seen. Overall, by refining initialization methods, modifying feature spaces, and experimenting with distance metrics, we achieved more accurate and efficient labeling of clothing images.

4. Final Conclusions

In this project, we addressed the image labeling problem by combining two simple but effective algorithms: K-means for color classification and KNN for shape classification. Our system effectively labels clothing images based on both color and shape.

Qualitative analysis methods, including retrieval-by-color, retrieval-by-shape, and retrieval-combined, provided visual insights into the performance of our classifiers. Quantitative analysis methods, such as K-means statistics, shape accuracy, and color accuracy, offered concrete metrics to evaluate our algorithms, revealing significant performance improvements.

Key enhancements to the K-means algorithm, such as Hypercube and K-means++ initialization methods, led to better clustering results and faster convergence.

For the KNN classifier, modifying the feature space and experimenting with different distance metrics improved classification accuracy and computational efficiency. Reducing image resolution and using custom features captured more relevant shape information

Overall, our enhancements to K-means and KNN resulted in significant improvements in accuracy and efficiency. The successful combination of these algorithms enabled us to create an effective image labeling system with potential applications in fashion e-commerce and automated inventory management.