# Clustering Project

Clustering program needs to cluster the 2D data and show the abilities and principles of work of the main clusterization algorithms such as K-Means and DBSCAN. This program use simple 2D dataset to show beginners the main principles of work and implementation of the basic algorithms and show that data sciense is not hard and can be realized on C++.

**Project Structure**

The project consists of several components: reading dataset, normalization of data, clustering by k-means, processing the results in terminal, visualizing them and exporting in file with clustered data. Then It doing the same for the second algorithm DBSCAN but with addition as tuning hyperparameters such as epsilon and minPts.

## Main Components

**main.cpp**

The entry point of the application. It initializes the application, loads data, normalizes it, performs clustering, and visualizes the results.

**csv.h and csv.cpp**

Header and source files for handling CSV file operations and normalizing data.

- **readCSV:** Reads data from a CSV file into a vector of **Point** objects.
- **normalizePoints:** Normalizes the coordinates of points to a scale of 0 to 1.

**dbscan.h and dbscan.cpp**

Header and source files for the DBSCAN clustering algorithm.

- **dbscan:** Applies the DBSCAN algorithm to a set of points.
- **autoTuneDBSCAN:** Automatically tunes the parameters for DBSCAN for optimal clustering.
- **silhouetteScore**: Calculates the silhouette score to evaluate clustering quality.

**graphs.h and graphs.cpp**

Header and source files for visualizing the clusters using the SFML library.

- **drawClusters:** Draws the clusters on a graphical window, including axes, grid, and labels.

**kmeans.h and kmeans.cpp**

Header and source files for the K-Means clustering algorithm.

- **kmeans**: Applies the K-Means algorithm to a set of points.
- **initializeCentroids:** Initializes the centroids for K-Means.
- **assignClusters**: Assigns points to the nearest centroid.
- **updateCentroids:** Updates the centroids based on the assigned points.
- **checkConvergence:** Checks if the centroids have converged.

**utils.h and utils.cpp**

Utility files containing helper functions and common operations used throughout the project.

## Usage

### Running the Application

To run the application, build the project using CMake and a compatible C++ compiler. Ensure that all dependencies are installed, including the SFML library.

**Clone the Repository:**

git clone <repository_url>
cd ClusteringProject

**Build the Project:**

**Run the Application:**

Dependencies

Ensure that the following dependencies are installed:

- SFML library (version 2.5 or later)

## Installation

**CMakeLists.txt**

The configuration file for building the project with CMake.

## Notes

- Ensure that the CSV files have the correct format and column names to avoid errors during data loading and processing.

- The application uses the SFML library for visualization, which must be correctly installed and linked.

## Future Enhancements

- Add more clustering algorithms for comparison.
- Implement user-friendly GUI for easier interaction.
- Enhance the visualization with more features and customization options.

This documentation provides an overview of the ClusteringProject application, its structure, and usage. For more detailed information, refer to the source code and inline comments.