Clustering Project

ClusteringProject is a data clustering application that uses the K-Means and DBSCAN algorithms. The project allows users to load data from a CSV file, normalize the data, apply clustering algorithms, and visualize the results using the SFML library.

Project Structure

The project consists of several components, each responsible for different functionalities within the application. Below is an overview of the key components and their purposes.

Main Components

main.cpp

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

```
#include <iostream>
#include <vector>
#include "graphs.h"
#include "csv.h"
#include "dbscan.h"
#include "kmeans.h"

void exportClusters(const std::vector<Point>& points, const
std::string& filename) {
    std::ofstream file(filename);
    if (file.is_open()) {
        file << "x,y,cluster\n";
        for (const auto& point : points) {
            file << point.x << "," << point.y << "," << point.cluster
<< "\n";
        }
        file.close();
    } else {
        std::cerr << "Unable to open file for writing: " << filename
<< std::endl;
    }
}
```

```
int main() {
    std::vector<Point> points = readCSV("data.csv");
    normalizePoints(points);
    int k = 3;
    kmeans(points, k, 0.01);
    for (const auto& point : points) {
        std::cout << "x: " << point.x << ", y: " << point.y
                  << ", cluster: " << point.cluster << std::endl;
    }

    drawClusters(points);
    exportClusters(points, "kmeans_data.csv");

    std::vector<Point> points1 = readCSV("data.csv");
    normalizePoints(points1);
    auto [bestEps, bestMinPts] = autoTuneDBSCAN(points1);
    dbscan(points1, bestEps, bestMinPts);
    for (const auto& point : points1) {
        std::cout << "x: " << point.x << ", y: " << point.y
                  << ", cluster: " << point.cluster << std::endl;
    }

    drawClusters(points1);
    exportClusters(points1, "dbscan_data.csv");

    return 0;
}
```

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:

```
mkdir build
cd build
cmake ..
make
```

Run the Application:

```
./ClusteringProject
```

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.

```
cmake_minimum_required(VERSION 3.10)
project(ClusteringProject)

set(CMAKE_CXX_STANDARD 11)

# Path to SFML
set(SFML_DIR "/opt/homebrew/opt/sfml")

find_package(SFML 2.5 COMPONENTS graphics window system REQUIRED)

include_directories(/usr/local/include)

link_directories(/opt/homebrew/opt/sfml)

set(SOURCES
        main.cpp
        csv.cpp
        dbscan.cpp
        graphs.cpp
        kmeans.cpp
        utils.cpp
        utils.h
)

add_executable(ClusteringProject ${SOURCES})
```

```
target_link_libraries(ClusteringProject sfml-graphics sfml-window
sfml-system)
```

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.