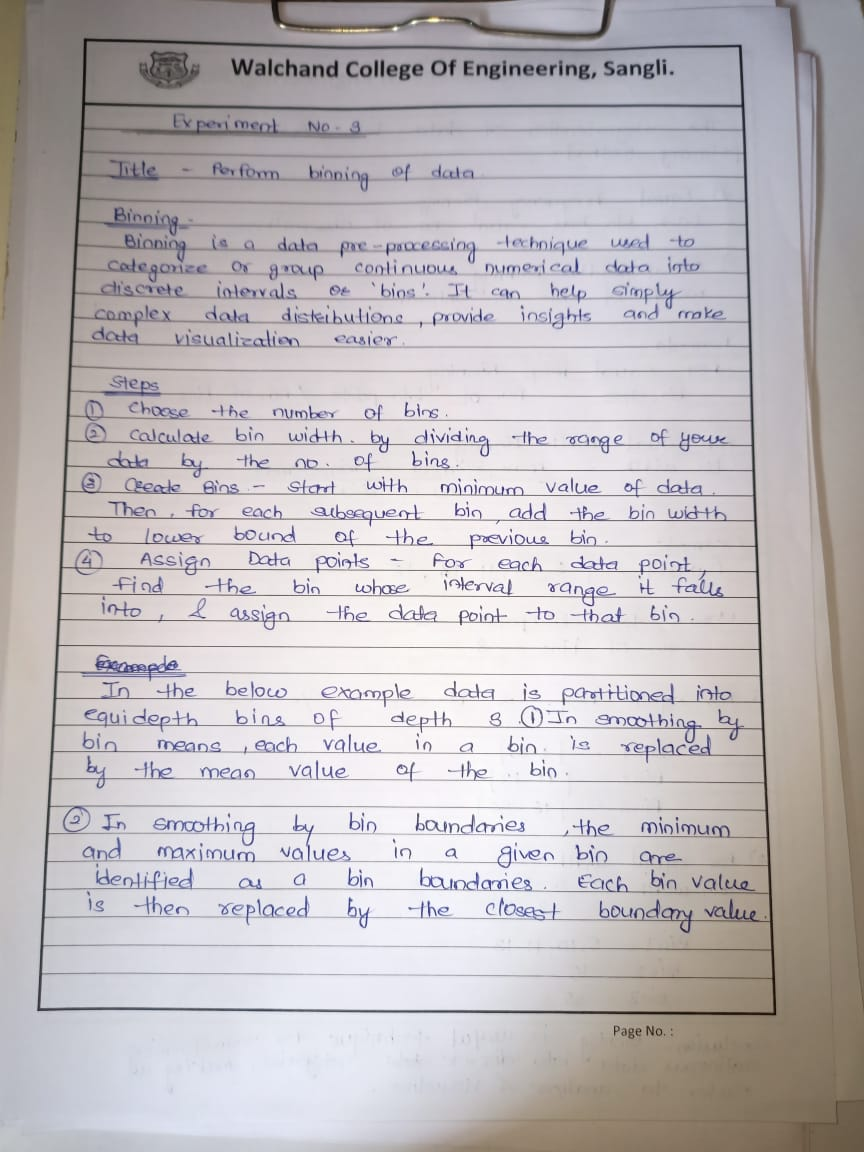
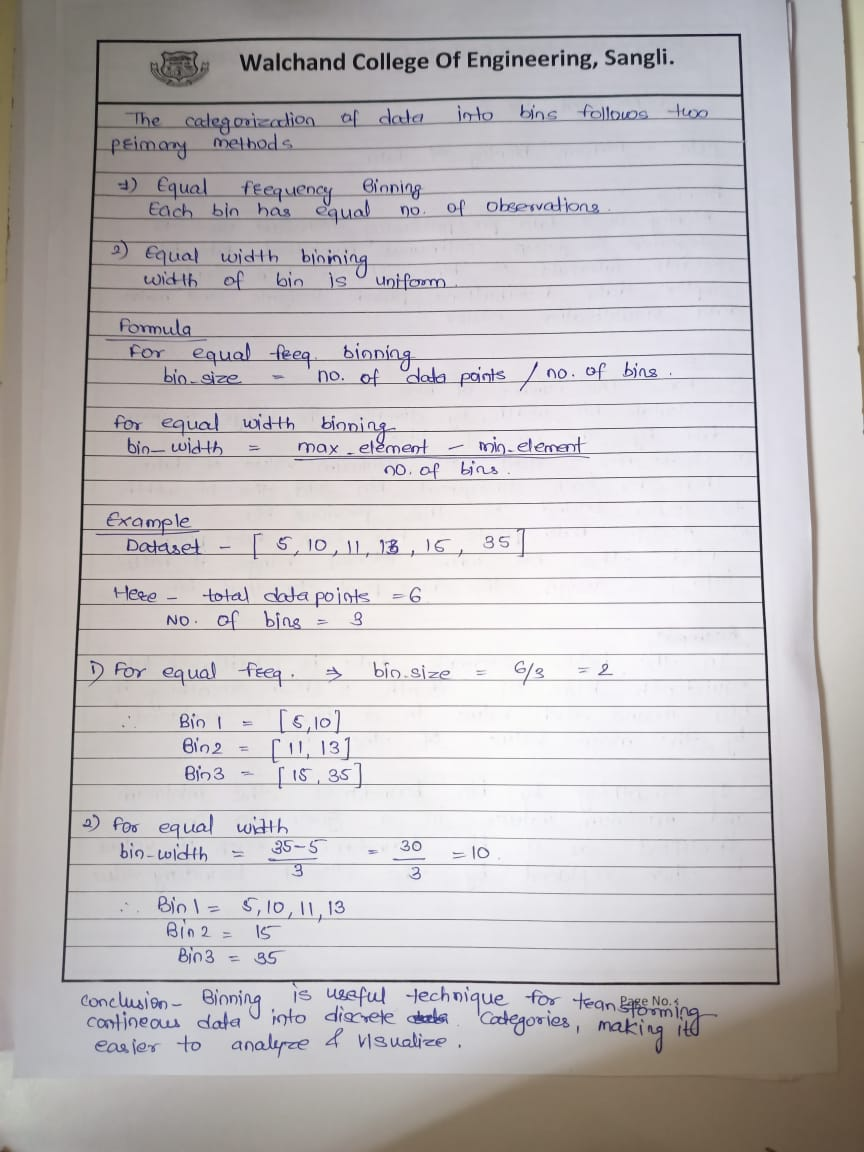
**Assignment 3**

**Title- Perform Binning of data.**





**Code**

#include <iostream>

#include <fstream>

#include <sstream>

#include <vector>

#include <climits>

using namespace std;

// Equal Frequency Binning

vector<vector<int>> equifreq(vector<int> data, int m)

{

  int a = data.size();

  int n = a / m;

  vector<vector<int>> bins;

  for (int i = 0; i < m; i++)

  {

    vector<int> bin;

    for (int j = i \* n; j < (i + 1) \* n; j++)

    {

      if (j >= a)

      {

        break;

      }

      bin.push\_back(data[j]);

    }

    bins.push\_back(bin);

  }

  return bins;

}

// Equal Width Binning

vector<vector<int>> equiwidth(vector<int> data, int m)

{

  int a = data.size();

  int max\_ele = INT\_MIN;

  int min\_ele = INT\_MAX;

  for (int i = 0; i < data.size(); i++)

  {

    max\_ele = max(max\_ele, data[i]);

    min\_ele = min(min\_ele, data[i]);

  }

  int w = (max\_ele - min\_ele) / m;

  int min1 = min\_ele;

  vector<int> arr;

  for (int i = 0; i < m + 1; i++)

  {

    arr.push\_back(min1 + w \* i);

  }

  vector<vector<int>> arri;

  for (int i = 0; i < m; i++)

  {

    vector<int> temp;

    for (int j : data)

    {

      if (j >= arr[i] && j <= arr[i + 1])

      {

        temp.push\_back(j);

      }

    }

    arri.push\_back(temp);

  }

  return arri;

}

// Reading data from CSV

vector<int> readCSV(string filename)

{

  ifstream inputFile(filename);

  vector<int> data;

  string line, value;

  while (getline(inputFile, line))

  {

    stringstream ss(line);

    while (getline(ss, value, ','))

    {

      data.push\_back(stoi(value));

    }

  }

  inputFile.close();

  return data;

}

// Write binning outputs to CSV

void writeCSV(string filename, vector<vector<int>> bins)

{

  ofstream outputFile(filename);

  for (int i = 0; i < bins.size(); i++)

  {

    outputFile << "Bin " << i + 1 << ",";

    for (int num : bins[i])

    {

      outputFile << num << ",";

    }

    outputFile << "\n";

  }

  outputFile.close();

}

int main()

{

  vector<int> data = readCSV("data.csv");

  int m;

  int method;

  cout << "Choose binning method: " << endl;

  cout << "1. Equal Frequency Binning" << endl;

  cout << "2. Equal Width Binning" << endl;

  cout << "\nEnter method number: ";

  cin >> method;

  cout << "\nEnter number of bins: ";

  cin >> m;

  if (method == 1)

  {

    vector<vector<int>> freqBins = equifreq(data, m);

    writeCSV("output\_equifreq.csv", freqBins);

  }

  else if (method == 2)

  {

    vector<vector<int>> widthBins = equiwidth(data, m);

    writeCSV("output\_equiwidth.csv", widthBins);

  }

  else

  {

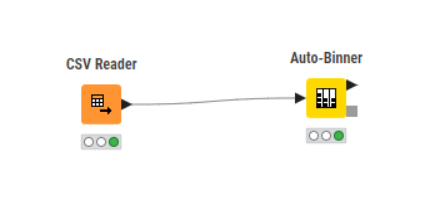
    cout << "Invalid method choice." << endl;

  }

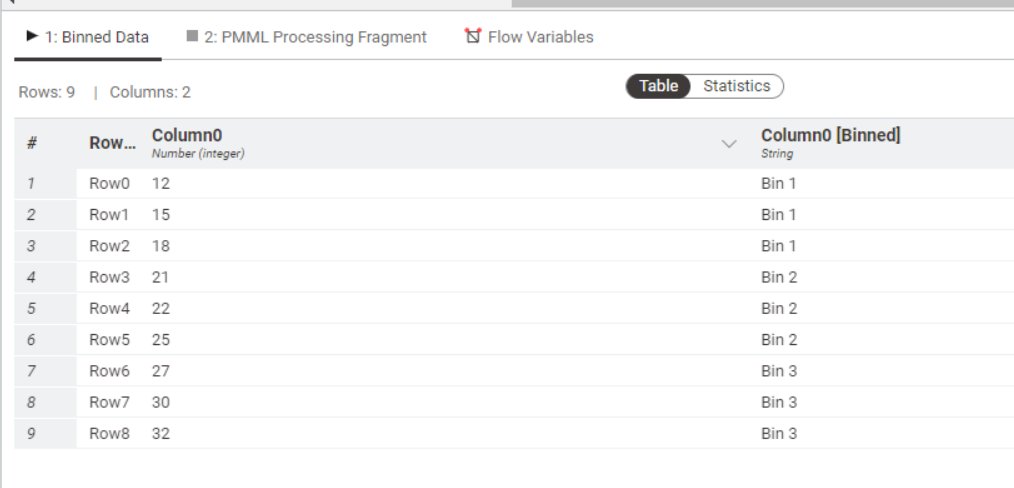
  return 0;

}

**Knime**



**Equi-width binning**



**Equi-frequency binning**

