Experiment no. 3

Name: Sonali Dattatray Kaingade

PRN: 21620002

Title: To perform Binning of data

Code:

```
#include <iostream>

#include <fstream>

#include <vector>

#include<algorithm>

#include <climits>

#include<cmath>

using namespace std;

//equal frequency

vector<vector<int>> equi_frequency(vector<int> data,double m)

{


   double n=data.size();

   double ele=ceil(n/m);


   vector<vector<int>> totalbins;

   for(int i=0;i<m;i++)

   {

      vector<int> bin;

      for(int j=i*ele;j<(i+1)*ele;j++)
```

```
        {
            if(j>=n)
            {
                break;
            }
            bin.push_back(data[j]);
        }
        totalbins.push_back(bin);
    }
    return totalbins;
}

//equal width
vector<vector<int>>equi_width(vector<int> data,int m)
{
    int n=data.size();

    int min_ele=INT_MAX;
    int max_ele=INT_MIN;

    for(int i=0;i<data.size();i++)
    {
        min_ele= min(min_ele,data[i]);
        max_ele= max(max_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 k:data)
        {
            if(k>=arr[i] && k<=arr[i+1])
            {
                temp.push_back(k);
            }
        }
        arri.push_back(temp);
    }
return arri;
}
```

```
// 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()
{

  ifstream inputf("input.csv");
  vector<int> data;
  int val;
  while(inputf>>val)
  {
       data.push_back(val);

  }

  sort(data.begin(),data.end());

   int method,m;

 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 = equi_frequency(data, m);

  writeCSV("output_equi_frequency.csv", freqBins);

 }

 else if (method == 2)

 {

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

  writeCSV("output_equi_width.csv", widthBins);

 }

 else

 {

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

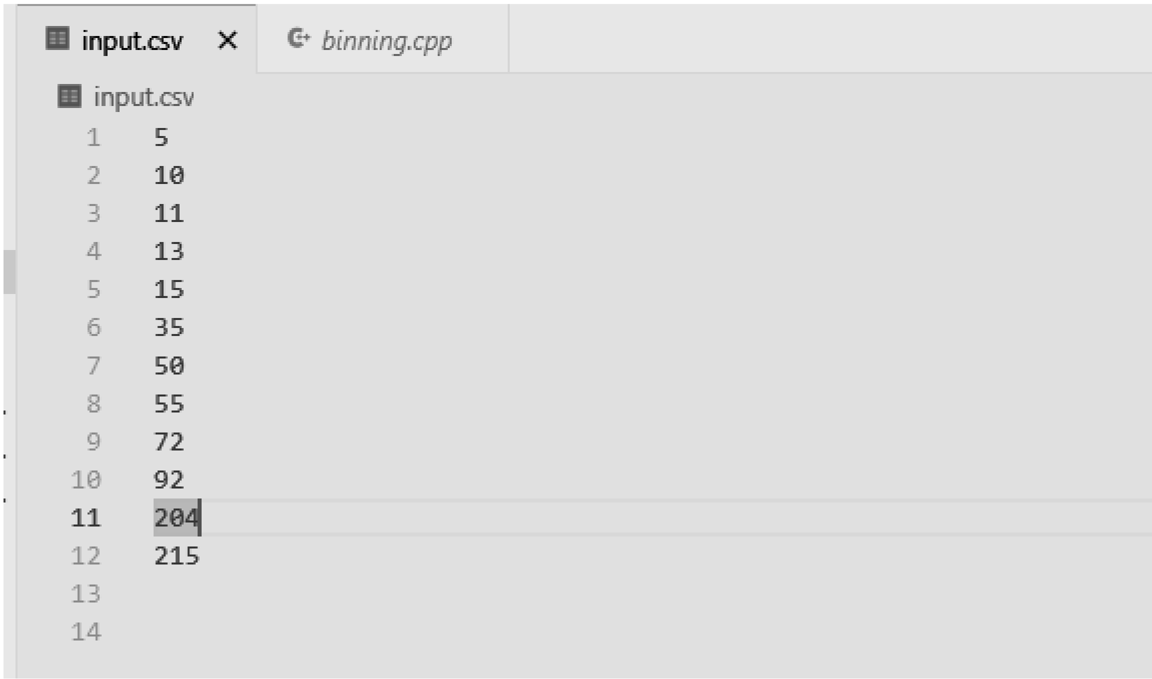
```
 }

   return 0;

}
```

Output:

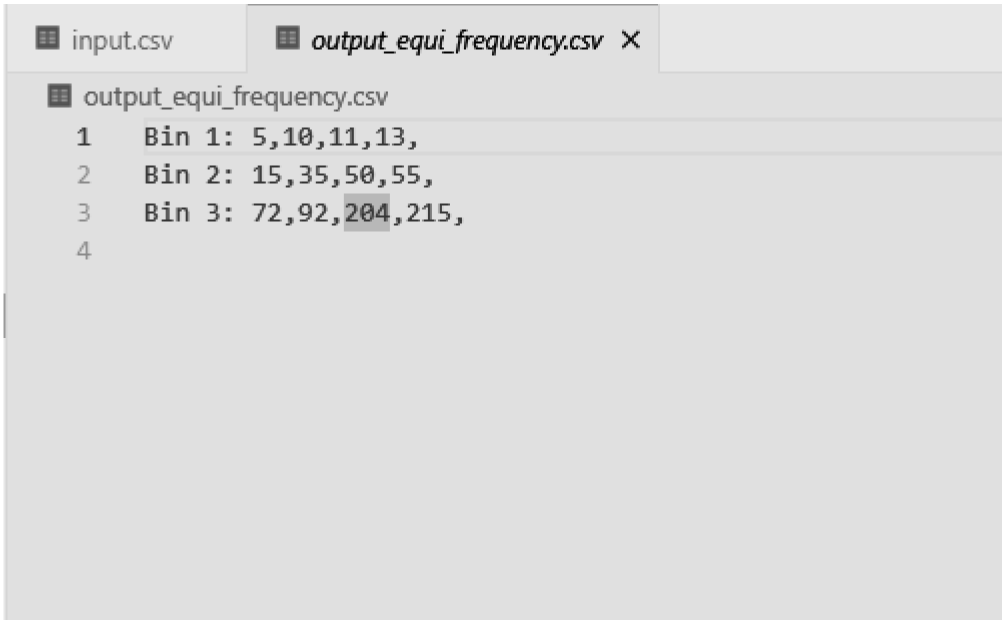
Input.csv



```
5
10
11
13
15
35
50
55
72
92
204
215
```

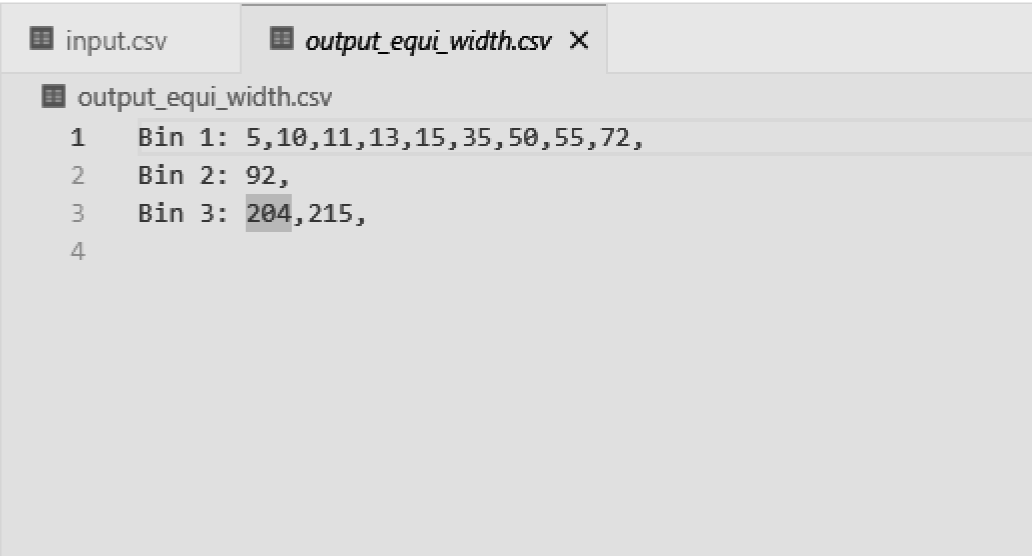
Output.csv

1.Equal frequency



```
Bin 1: 5,10,11,13,
Bin 2: 15,35,50,55,
Bin 3: 72,92,204,215,
```

2. Equal width

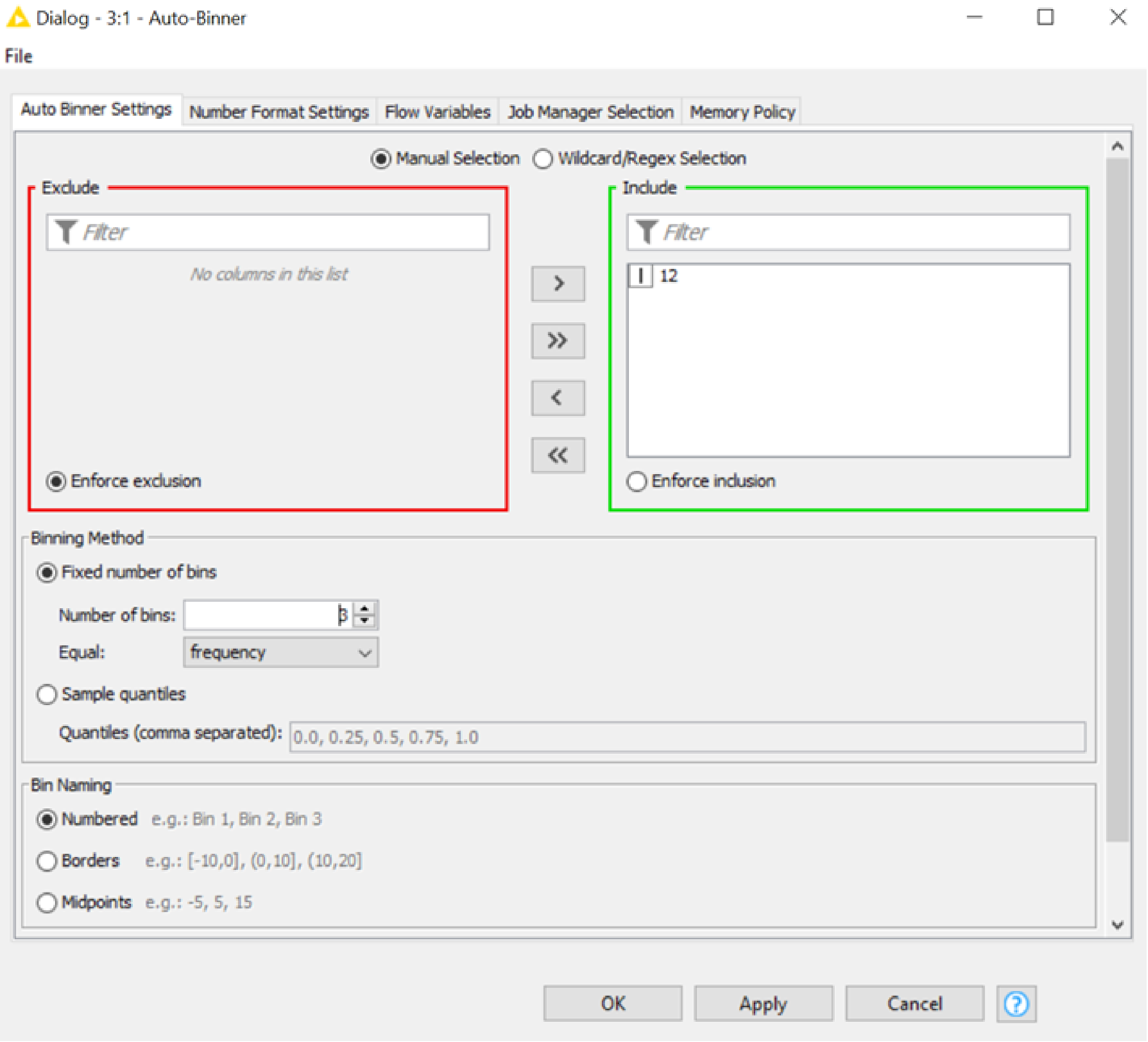


```
Bin 1: 5,10,11,13,15,35,50,55,72,
Bin 2: 92,
Bin 3: 204,215,
```

Knime:

Binning methods:

1. Equi- Frequency:



Dialog - 3:1 - Auto-Binner

File

Auto Binner Settings | Number Format Settings | Flow Variables | Job Manager Selection | Memory Policy

● Manual Selection ○ Wildcard/Regex Selection

Exclude

Filter

No columns in this list

● Enforce exclusion

Include

Filter

I 12

○ Enforce inclusion

Binning Method

● Fixed number of bins

Number of bins: 3

Equal: frequency

○ Sample quantiles

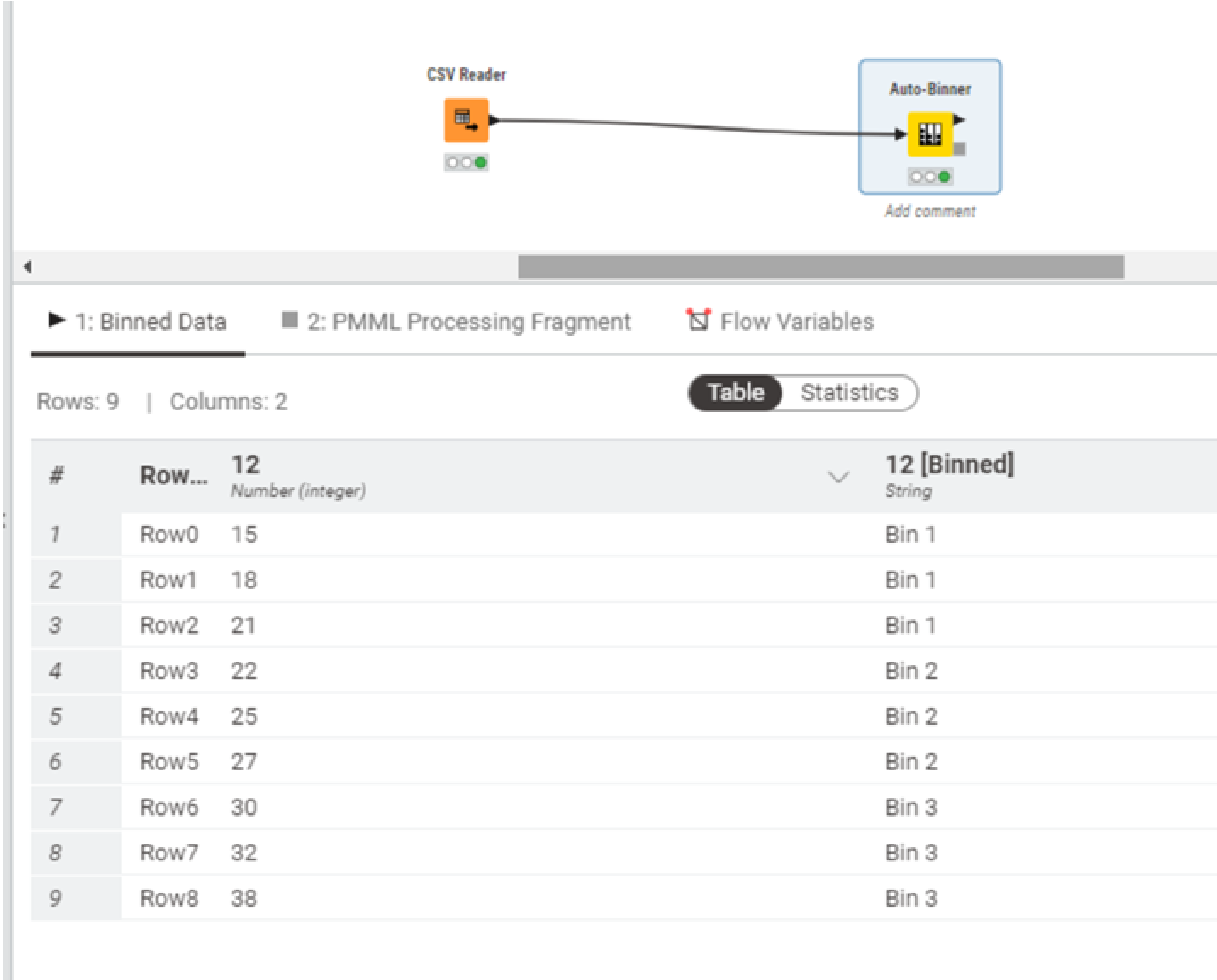
Quantiles (comma separated): 0.0, 0.25, 0.5, 0.75, 1.0

Bin Naming

● Numbered e.g.: Bin 1, Bin 2, Bin 3

○ Borders e.g.: [-10,0], (0,10], (10,20]

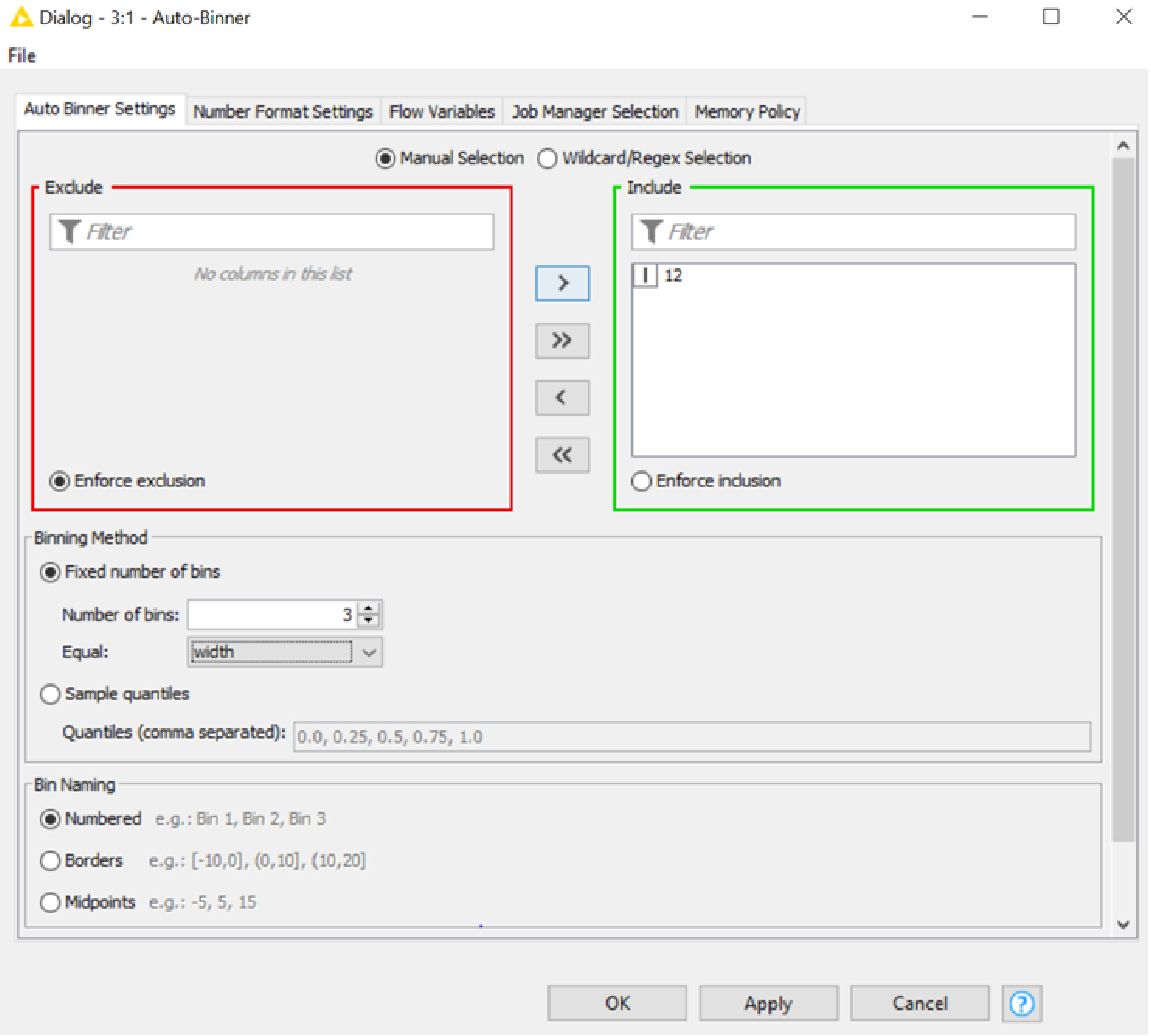
○ Midpoints e.g.: -5, 5, 15



Rows: 9 | Columns: 2

#	Row...	12 Number (integer)	12 [Binned] String
1	Row0	15	Bin 1
2	Row1	18	Bin 1
3	Row2	21	Bin 1
4	Row3	22	Bin 2
5	Row4	25	Bin 2
6	Row5	27	Bin 2
7	Row6	30	Bin 3
8	Row7	32	Bin 3
9	Row8	38	Bin 3

2. Equi width:



Dialog - 3:1 - Auto-Binner

File

Auto Binner Settings | Number Format Settings | Flow Variables | Job Manager Selection | Memory Policy

● Manual Selection ○ Wildcard/Regex Selection

Exclude

Filter

No columns in this list

● Enforce exclusion

Include

Filter

I 12

○ Enforce inclusion

Binning Method

● Fixed number of bins

Number of bins: 3

Equal: width

○ Sample quantiles

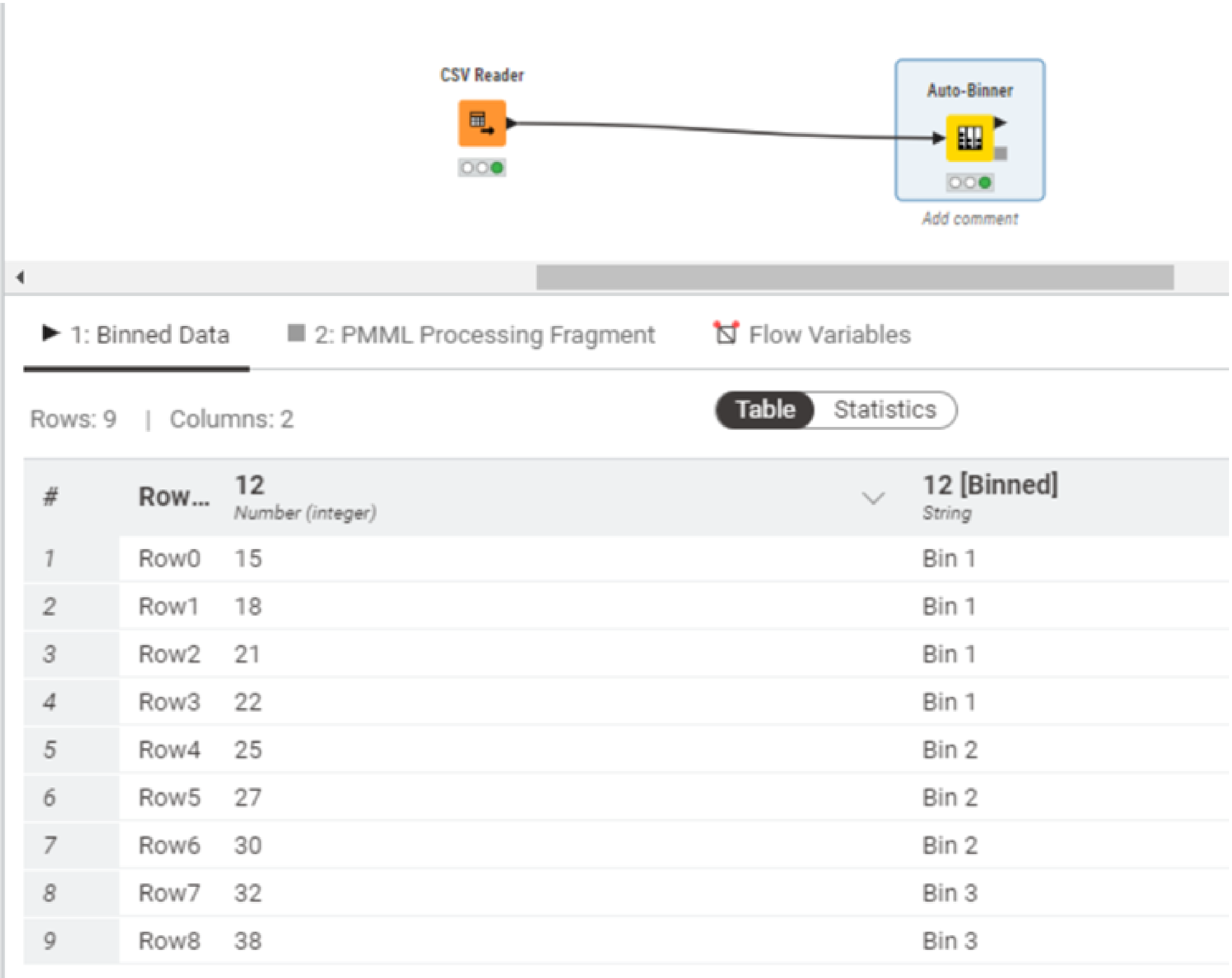
Quantiles (comma separated): 0.0, 0.25, 0.5, 0.75, 1.0

Bin Naming

● Numbered e.g.: Bin 1, Bin 2, Bin 3

○ Borders e.g.: [-10,0], (0,10], (10,20]

○ Midpoints e.g.: -5, 5, 15



Rows: 9 | Columns: 2

#	Row...	12 *Number (integer)*	12 [Binned] *String*
1	Row0	15	Bin 1
2	Row1	18	Bin 1
3	Row2	21	Bin 1
4	Row3	22	Bin 1
5	Row4	25	Bin 2
6	Row5	27	Bin 2
7	Row6	30	Bin 2
8	Row7	32	Bin 3
9	Row8	38	Bin 3