Experiment no. 2

Name: Sonali Dattatray Kaingade

PRN: 21620002

Title: To perform normalization of data (Min-max and z-score).

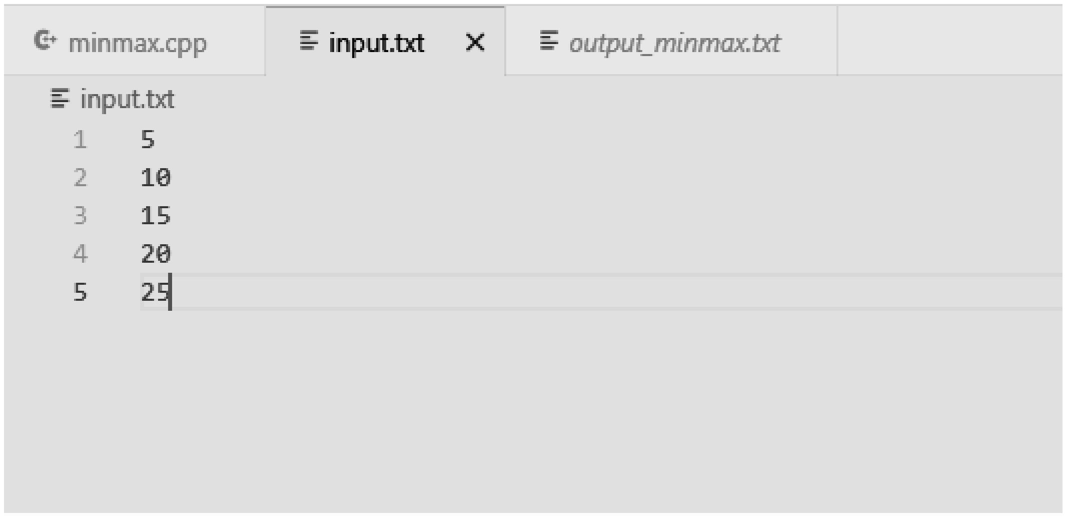
1. Min max normalization

Code:

```
#include <iostream>

#include <fstream>

#include <vector>

#include <algorithm>


using namespace std;


double min_max_scaling(double x, double x_min, double x_max, double x_newmin, double x_newmax)
{

   return ((x - x_min) / (x_max - x_min)) * (x_newmax - x_newmin) + x_newmin;

}


int main() {

   ifstream input_file("input.txt");

   ofstream output_file("output_minmax.txt");


   vector<double> data;

   double value;


   while (input_file >> value) {
        data.push_back(value);
    }


    double x_min = *min_element(data.begin(), data.end());
    double x_max = *max_element(data.begin(), data.end());


    double x_newmin = 1.0; // New minimum value for scaled data
    double x_newmax = 10.0; // New maximum value for scaled data


    for (const double &x : data) {
        double normalized_value = min_max_scaling(x, x_min, x_max, x_newmin, x_newmax);
        output_file << normalized_value << endl;
    }


    input_file.close();
    output_file.close();





    cout << "output is generated in output_minmax file" << endl;


    return 0;
}
```

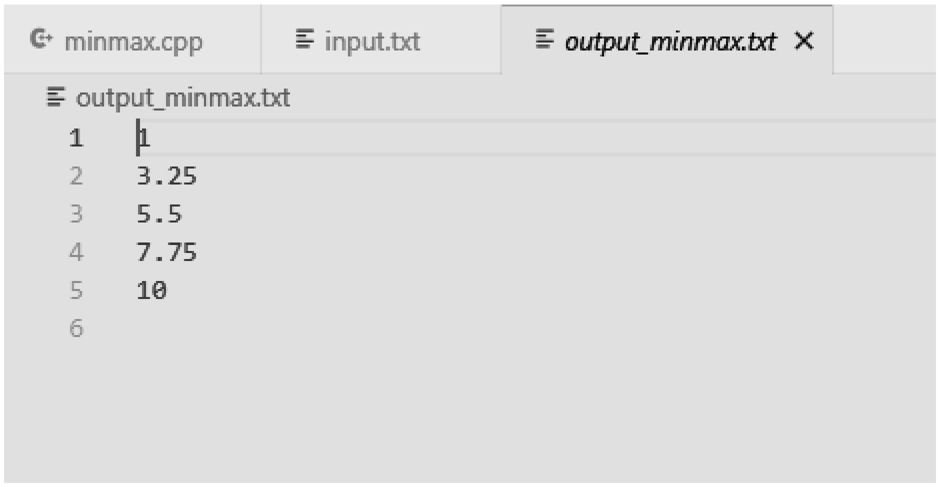
Output:

input.txt



```
5
10
15
20
25
```

Output_minmax.txt



```
1
3.25
5.5
7.75
10

```

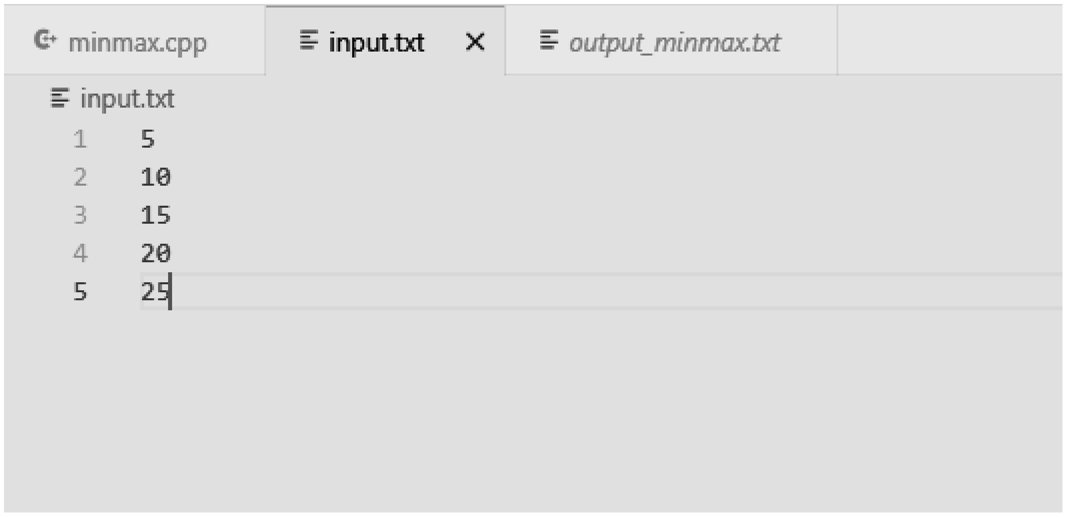
2. Z-score normalization

Code:

```
#include <iostream>

#include <fstream>

#include <vector>

#include <algorithm>

#include <cmath>


using namespace std;


double z_score(double x, double mean, double std_dev) {

    return (x - mean) / std_dev;

}


int main() {

    ifstream input_file("input.txt");

    ofstream output_file("output_zscore.txt");


    vector<double> data;

    double value;


    while (input_file >> value) {

        data.push_back(value);

    }
    double sum = 0.0;
    for (const double &x : data) {
        sum += x;
    }
    double mean = sum / data.size();

    double squared_diff_sum = 0.0;
    for (const double &x : data) {
        squared_diff_sum += pow(x - mean, 2);
    }
    double std_dev = sqrt(squared_diff_sum / data.size());

    for (const double &x : data) {
        double normalized_value = z_score(x, mean, std_dev);
        output_file << normalized_value << endl;
    }

    input_file.close();
    output_file.close();

    cout << "output is generated in output_zscore file" << endl;

    return 0;
}
```

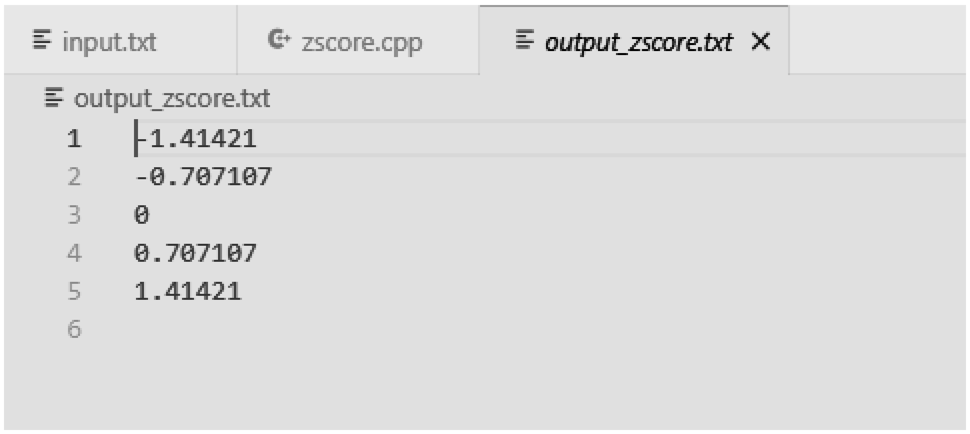
Output:

input.txt:



```
5
10
15
20
25
```

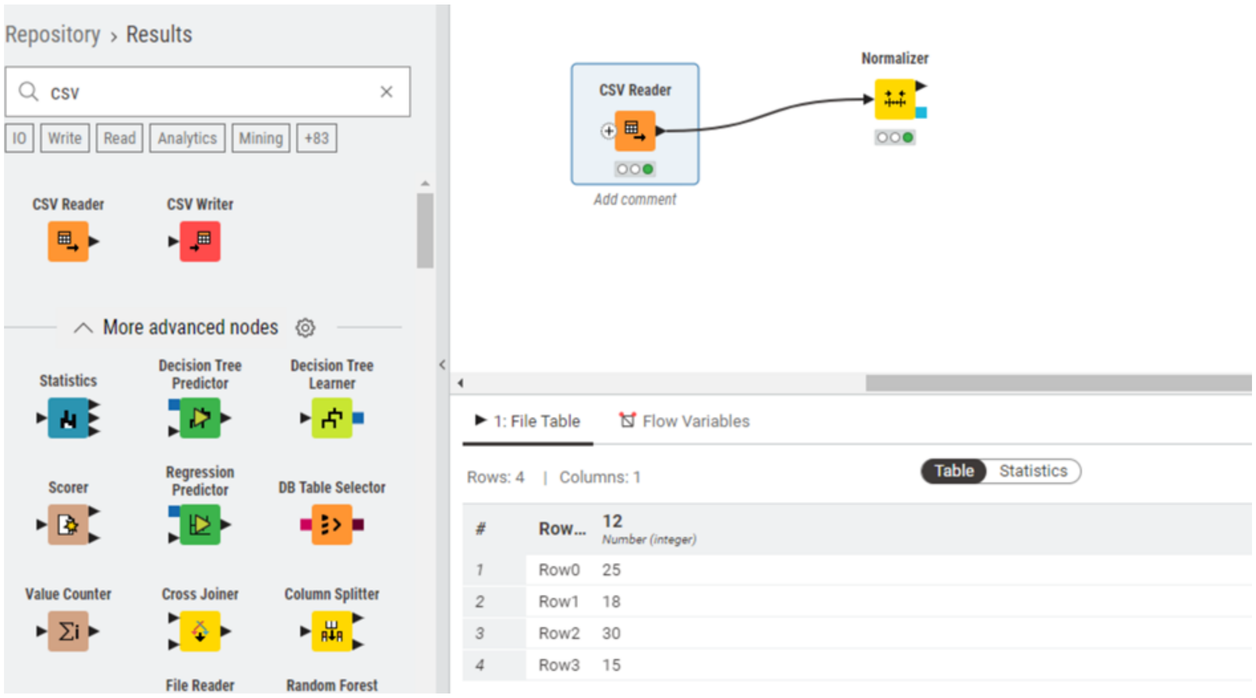
Output_zscore.txt:



```
-1.41421
-0.707107
0
0.707107
1.41421

```

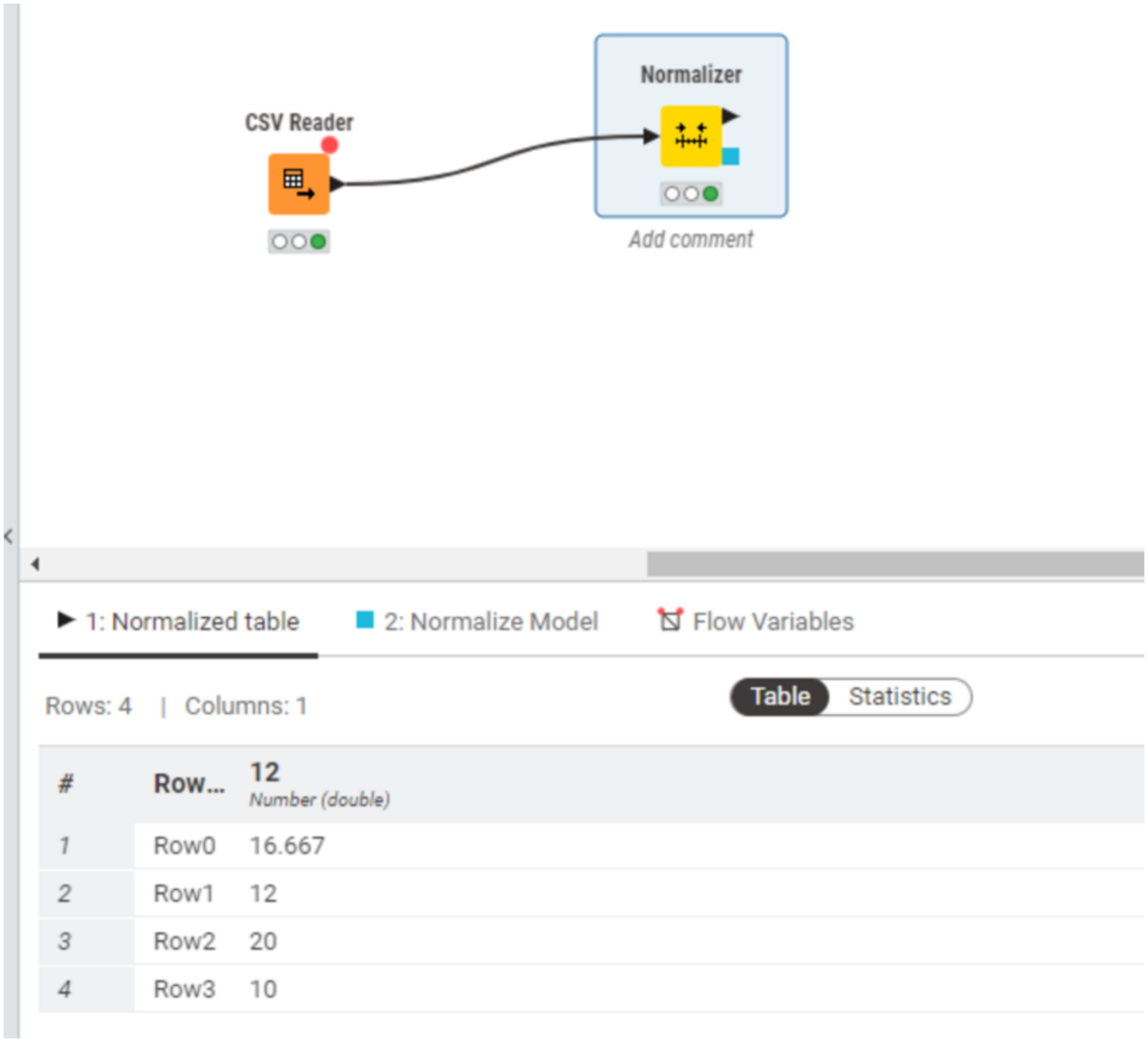
Knime:

Min-Max Normalization:



Rows: 4 | Columns: 1

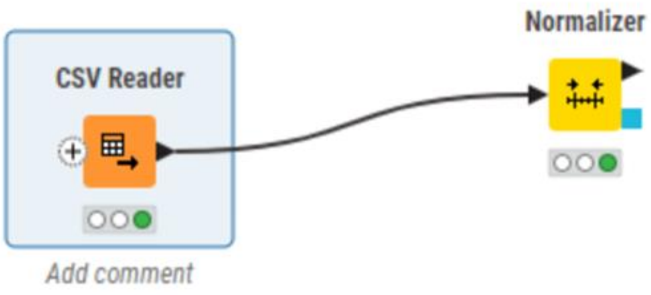
#	Row...	12 Number (integer)
1	Row0	25
2	Row1	18
3	Row2	30
4	Row3	15

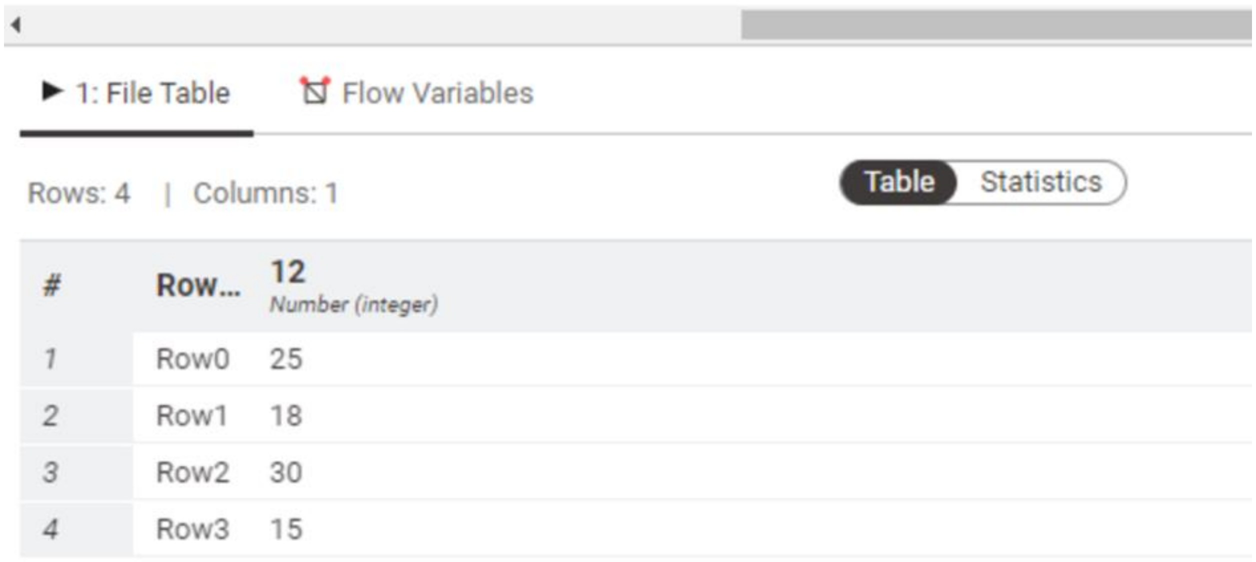


Rows: 4 | Columns: 1

#	Row...	12 Number (double)
1	Row0	16.667
2	Row1	12
3	Row2	20
4	Row3	10

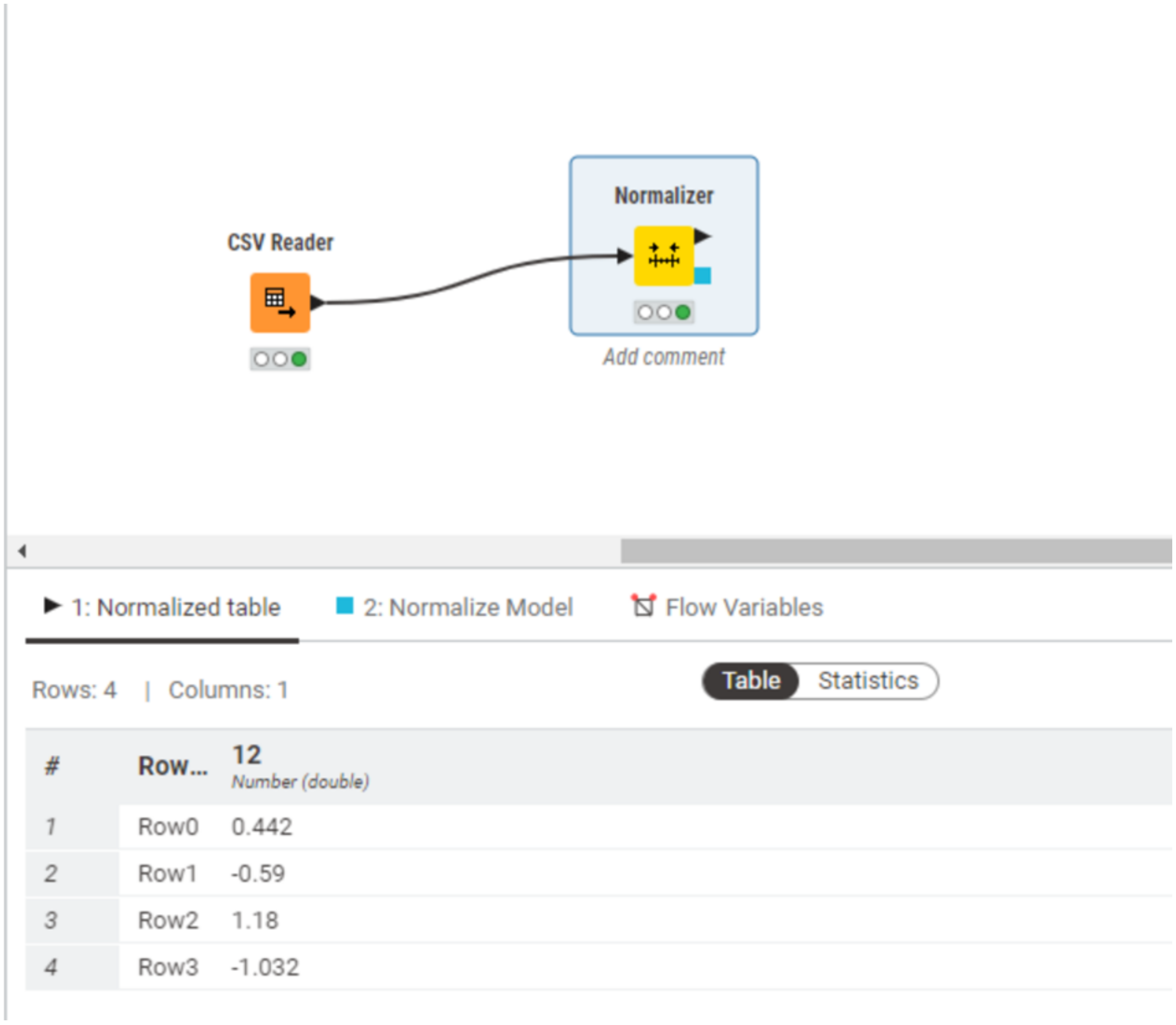
Z - Score Normalization:





Rows: 4 | Columns: 1

#	Row...	12 Number (integer)
1	Row0	25
2	Row1	18
3	Row2	30
4	Row3	15



Rows: 4 | Columns: 1

#	Row...	12 Number (double)
1	Row0	0.442
2	Row1	-0.59
3	Row2	1.18
4	Row3	-1.032