**Program 2:** Demonstration of normalization technique.

Code:

#include <iostream>

#include <string>

#include <fstream>

#include <vector>

#include<bits/stdc++.h>

#include <algorithm>

#include <math.h>

using namespace std;

int power(float num)

{

int cou=0;

while(num>1){

num /= 10;

cou++;

}

return cou;

}

int main(){

string road,air,temp,line;

vector<float> roadT;

vector<float> airT;

vector<float> roadTM;

vector<float> airTM;

vector<float> roadTZ;

vector<float> airTZ;

vector<float> roadTD;

vector<float> airTD;

int i,powOften\_road,powOften\_air = 0;

float roadTmax,roadTmin,airTmax,airTmin,roadSD,airSD = 0;

ifstream coeff("weather.csv");

//save data in vector formate

if (coeff.is\_open()) //if the file is open

{

//ignore first line

string line;

getline(coeff, line);

while (!coeff.eof())

{

getline(coeff, temp, ',');getline(coeff, temp, ',');

getline(coeff, temp, ',');getline(coeff, temp, ',');

getline(coeff, temp, ',');getline(coeff, temp, ',');

getline(coeff, road, ',');

roadT.push\_back(stof(road));

getline(coeff, air, '\n');

airT.push\_back(stof(air));

i += 1;

}

coeff.close();

cout << "Number of lines: " << i-1 << endl;

}

else{

cout << "Unable to open file"<<endl;

}

//Min-Max normalization

roadTmax = \*max\_element(roadT.begin(), roadT.end());

roadTmin = \*min\_element(roadT.begin(), roadT.end());

for (auto& data : roadT) {

roadTM.push\_back(((data - roadTmin)/(roadTmax-roadTmin))\*(1-0)\*(1));

}

airTmax = \*max\_element(airT.begin(), airT.end());

airTmin = \*min\_element(airT.begin(), airT.end());

for (auto& data : airT) {

airTM.push\_back(((data- airTmin)/(airTmax-airTmin))\*(1-0)\*(1));

}

//decimal

if (abs(roadTmax)>abs(roadTmin)){

powOften\_road = power(abs(roadTmax));

}

else{

powOften\_road = power(abs(roadTmin));

}

if (abs(airTmax)>abs(airTmin)){

powOften\_air = power(abs(airTmax));

}

else{

powOften\_air = power(abs(airTmin));

}

for (auto& data : roadT) {

roadTD.push\_back(data/pow(10,powOften\_road));

}

for (auto& data : airT) {

airTD.push\_back(data/pow(10,powOften\_air));

}

//Z-score

float roadAvg = accumulate( roadT.begin(), roadT.end(), 0.0)/roadT.size();

float airAvg = accumulate( airT.begin(), airT.end(), 0.0)/airT.size();

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

{

roadSD+= pow(roadT[i] - roadAvg, 2);

airSD+= pow(airT[i] - airAvg, 2);

}

roadSD/=roadT.size();

airSD/=airT.size();

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

{

roadTZ.push\_back((roadT[i]-roadAvg)/sqrt(roadSD));

airTZ.push\_back((airT[i]-airAvg)/sqrt(airSD));

}

//give output to file

ifstream inFile;

inFile.open("weather.csv");

ofstream outfile;

outfile.open("Output.csv");

getline(inFile,line);

line=line+",RoadTempMin-max,AirTempMin-Max,RoadTempZ,AirTempZ,RoadTempDec,AirTempDec\n";

outfile<<line;

int k=0;

while(getline(inFile,line))

{

line=line+","+to\_string(roadTM[k])+","+to\_string(airTM[k])+","+to\_string(roadTZ[k])+","+to\_string(airTZ[k])+","+to\_string(roadTD[k])+","+to\_string(airTD[k])+"\n";

k++;

outfile<<line;

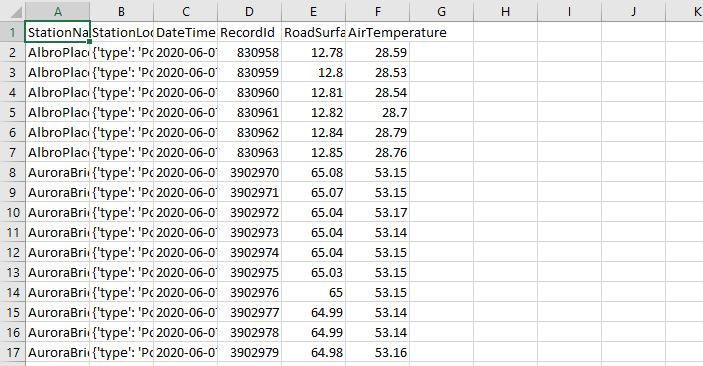
}

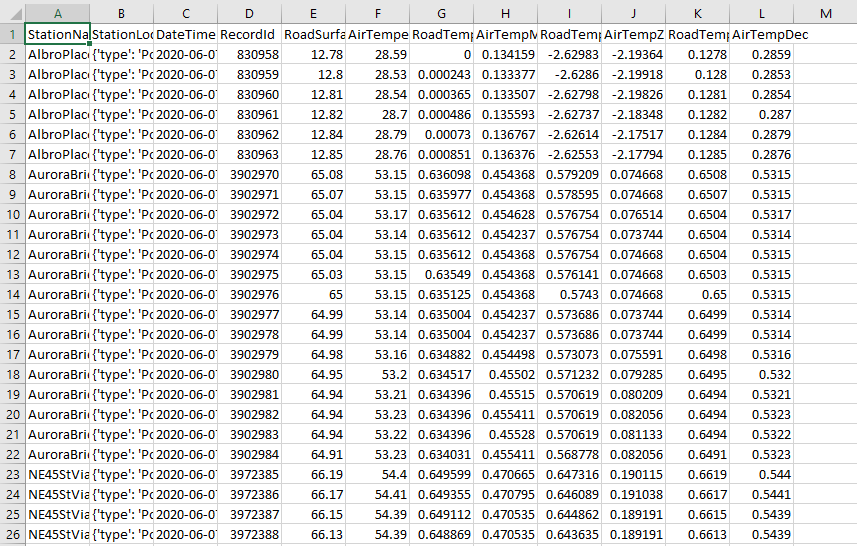
return 0;

}

Output:

Before:



After: