

CS4375.003

Simon Kim

Sxk190106

1. Code and output

<C++ code>

```
#include <iostream>
```

```
#include <sstream>
```

```
#include <fstream>
```

```
#include <iomanip>
```

```
#include <vector>
```

```
#include <string.h>
```

```
#include <algorithm>
```

```
#include <numeric>
```

```
#include <iterator>
```

```
using namespace std;
```

```
double getSum(vector<double> vect) {
```

```
    //get sum
```

```
    double sum = accumulate(vect.begin(), vect.end(), 0.0);
```

```
    return sum;
```

```
}
```

```
double getAverage(vector<double> vect) {
```

```
    // get average
```

```
    double sum = getSum(vect);
```

```
    double average = sum / vect.size();
```

```
    return average;
}
```

```
void print_stats(vector<double> vect) {
    //sort
    sort(vect.begin(), vect.end());

    //sum
    double sum = getSum(vect);
    cout << "sum: " << sum << endl;

    //mean
    double mean = getAverage(vect);
    cout << "mean: " << mean << endl;

    //median
    auto n = vect.size();
    double median = ((vect[n / 2] + vect[(n - 1) / 2]) / 2.0);
    cout << "median: " << median << endl;

    //range(min max)
    double min = *min_element(vect.begin(), vect.end());
    double max = *max_element(vect.begin(), vect.end());
    cout << "range: " << min << " " << max << endl;

}
```

```
double getStandardDeviation(vector<double>vect){
```

```

//get standard deviation

double mean = getAverage(vect);

double deviation = 0.0;
double variance = 0.0;

int j = vect.size();
for (int i = 0; i < j; i++)
{
    deviation = vect[i] - mean;
    variance += deviation * deviation;
}

double sd = sqrt(variance);
return sd;
}

double covar(vector<double> rm, vector<double> medv){
    //covariation

    //get average for rm
    double mean1 = getAverage(rm);

    //get average for medv
    double mean2 = getAverage(medv);

```

```

//get covariance
double num = 0.0;
int j = rm.size();
for (int i = 0; i < j; i++)
{
    num += (rm[i] - mean1) * (medv[i] - mean2);
}
double covar = num / rm.size();

return covar;

}

double cor(vector<double> rm, vector<double> medv){
    //correlation

    //get average for rm
    double mean1 = getAverage(rm);

    //get average for medv
    double mean2 = getAverage(medv);

    //get standard deviation for rm
    double sd1 = getStandardDeviation(rm);

    //get standard deviation for medv

```

```

double sd2 = getStandardDeviation(medv);

//calculate pcc from covar
double pcc = covar(rm, medv) / (sd1 * sd2);

return pcc;
}

int main() {
    ifstream inFS;
    string line;
    string rm_in, medv_in;
    const int MAX_LEN = 1000;
    vector<double> rm(MAX_LEN);
    vector<double> medv(MAX_LEN);

    //Try to open file
    cout << "Opening file Boston.csv" << endl;

    inFS.open("Boston.csv");
    if (!inFS.is_open()) {
        cout << "Could not open file Boston.csv." << endl;
        return 1; //1 indicates error
    }

    // Can now use inFS stream like cin stream

```

```
// Boston.csv should contain two doubles

cout << "Reading line 1" << endl;
getline(inFS, line);

// echo heading
cout << "heading: " << line << endl;

int numObservations = 0;
while (inFS.good()) {

    getline(inFS, rm_in, ',');
    getline(inFS, medv_in, '\n');
    rm.at(numObservations) = stof(rm_in);
    medv.at(numObservations) = stof(medv_in);

    numObservations++;

}

rm.resize(numObservations);
medv.resize(numObservations);

cout << "new length" << rm.size() << endl;
cout << "Closing file Boston.csv." << endl;
inFS.close(); // Done with file, so close it

cout << "Number of records: " << numObservations << endl;
```

```

cout << "\nStats for rm" << endl;
print_stats(rm);

cout << "\nStats for medv" << endl;
print_stats(medv);

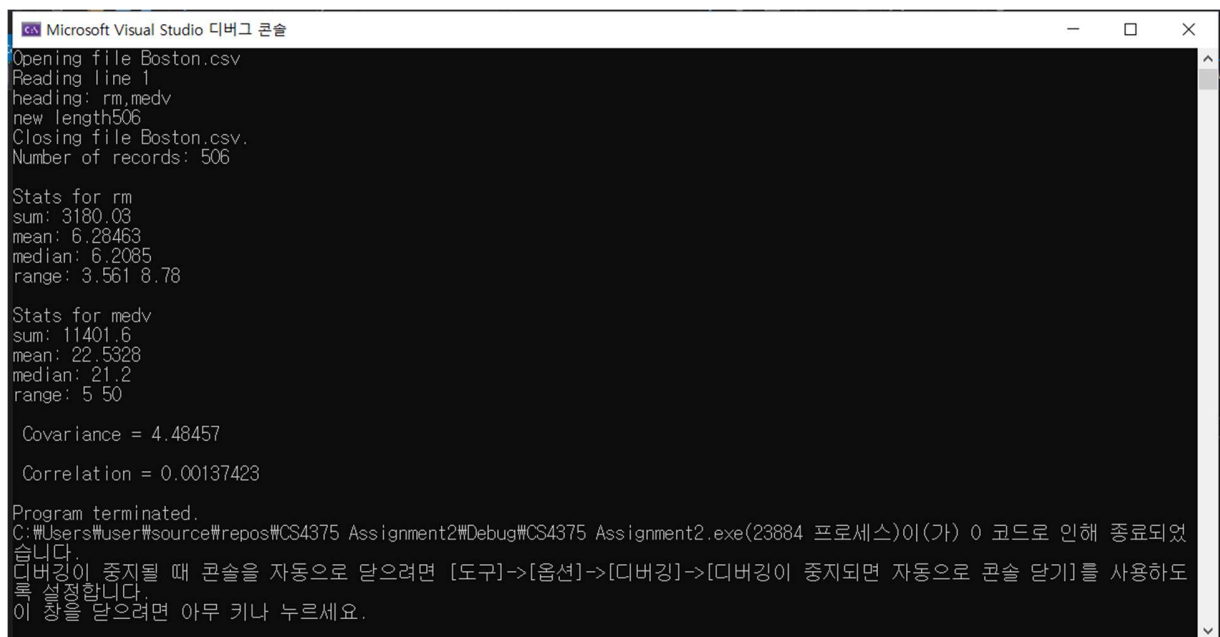
cout << "\n Covariance = " << covar(rm, medv) << endl;
cout << "\n Correlation = " << cor(rm, medv) << endl;

cout << "\nProgram terminated.";
return 0;

}

```

<output>



```

Microsoft Visual Studio 디버그 콘솔
Opening file Boston.csv
Reading line 1
heading: rm,medv
new length506
Closing file Boston.csv.
Number of records: 506

Stats for rm
sum: 3180.03
mean: 6.28463
median: 6.2085
range: 3.561 8.78

Stats for medv
sum: 11401.6
mean: 22.5328
median: 21.2
range: 5 50

Covariance = 4.48457

Correlation = 0.00137423

Program terminated.
C:\Users\user\source\repos\CS4375 Assignment2\Debug\CS4375 Assignment2.exe(23884 프로세스)이(가) 0 코드로 인해 종료되었
습니다.
디버깅이 중지될 때 콘솔을 자동으로 닫으려면 [도구]->[옵션]->[디버깅]->[디버깅이 중지되면 자동으로 콘솔 닫기]를 사용하도
록 설정합니다.
이 창을 닫으려면 아무 키나 누르세요.

```

2. R vs C++

C++ is much faster than R when we run the program. However, R is easier to use, since R already have those functions to calculate sum, mean, range, correlation and covariance. Therefore, if a person isn't familiar to use C/C++, he will be best to use R, not C++.

3. Statistical measures and Machine Learning

The descriptive statistical measures I used in this task are the mean, the median, and the variance. All are data used to determine the tendency of data. Regression is a method of predicting new data through data trends, and when this method is applied to machine learning, supervised learning can be performed by predicting new data from existing data.

4. Covariance and correlation

Covariance and correlation is about how 2 data is correlated to each other. The results obtained through machine learning should be explained through the relationship between prediction and actual value. Covariance and PCC are values that describe the degree of correlation between the two values. With this value, we can quantify the explanatory power of the model. Therefore, Covariance and PCC will be used as indicators to evaluate the accuracy of machine learning using regression.