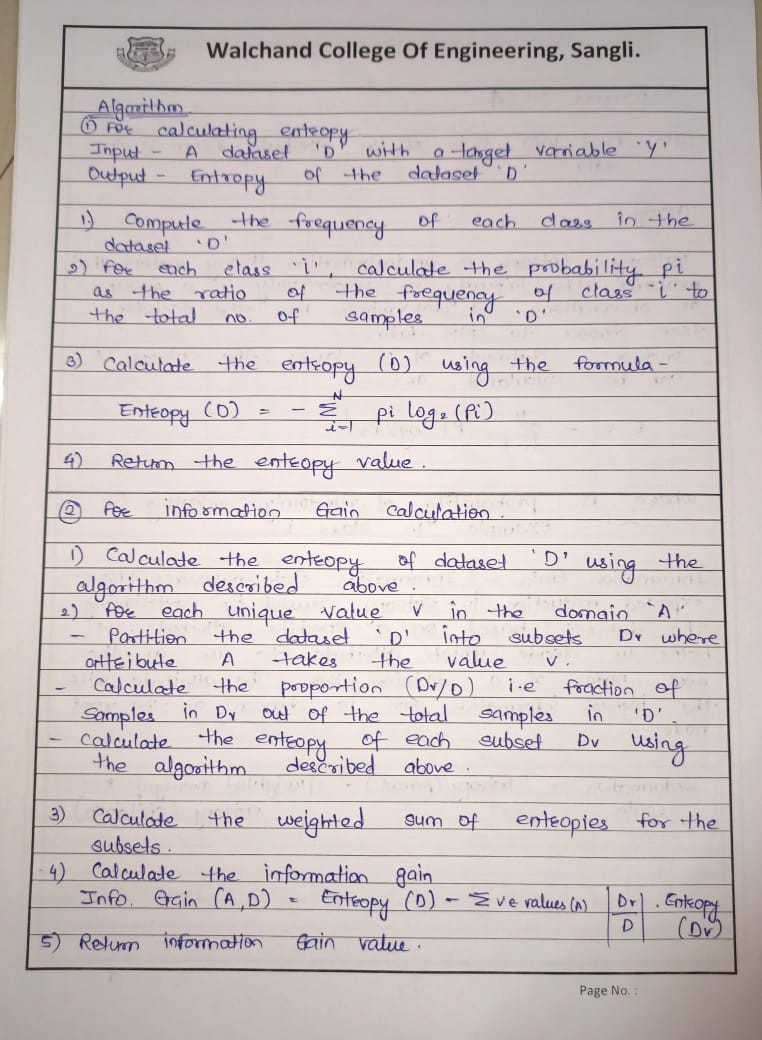
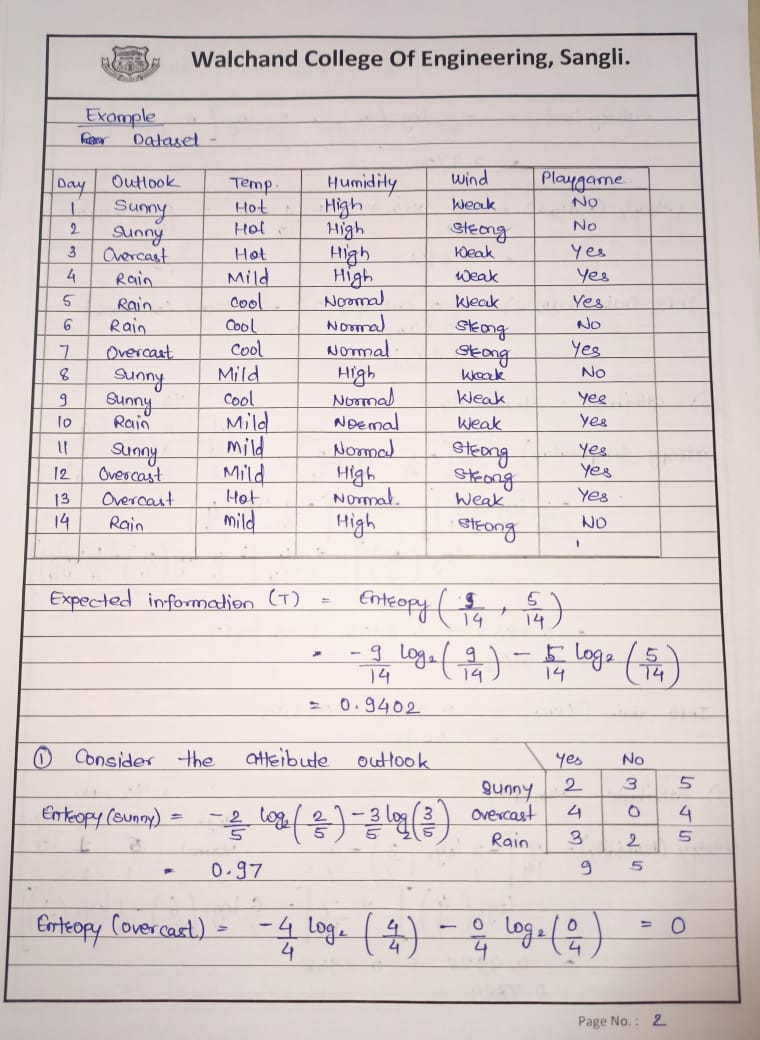
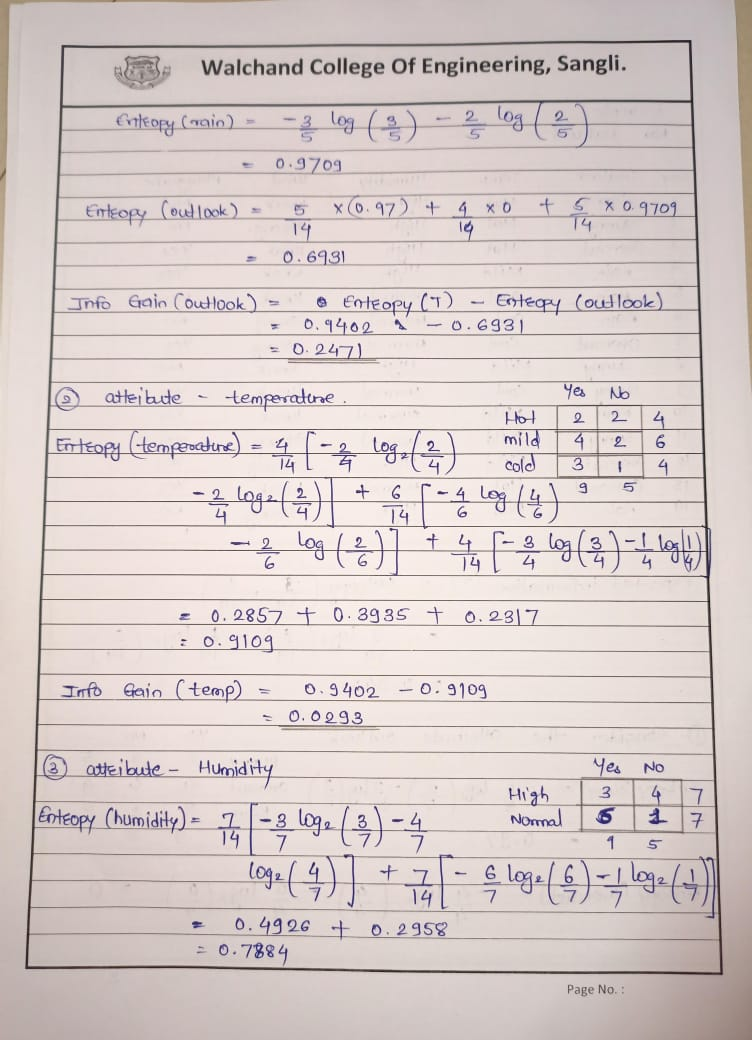
**Assignment 4**

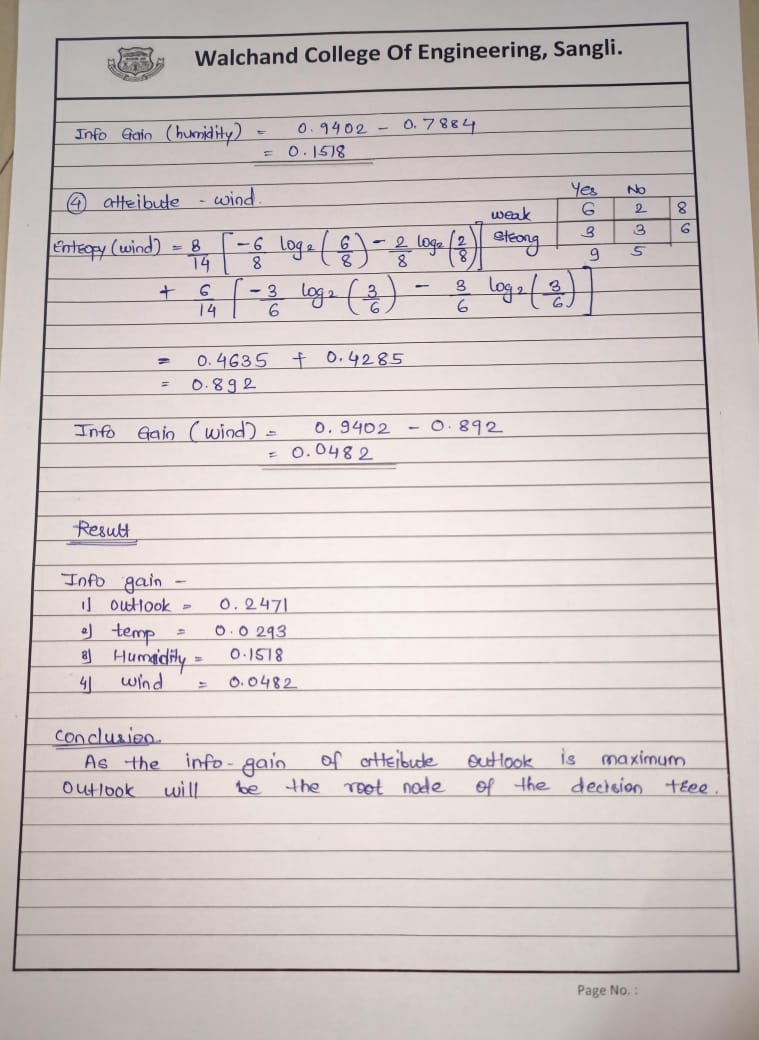
**Title- Find the info gain of an attribute from given data**

****

****

****

****

****

**Code-**

#include<iostream>

#include<fstream>

#include<sstream>

#include<map>

#include<cmath>

using namespace std;

//function to calculate entropy

double calculateEntropy(double positive, double negative){

    double total = positive + negative;

    double entropy = 0.0;

    if(total > 0){

        double positiveProbability = positive/total;

        double negativeProbability = negative/total;

        if(positiveProbability > 0){

            entropy -= positiveProbability \* log2(positiveProbability);

        }

        if(negativeProbability > 0){

            entropy -= negativeProbability \* log2(negativeProbability);

        }

    }

    return entropy;

}

//function to calculate info gain

double computeInformationGain( map<string,int>&parentCounts, map<string,map<string,int> >childCounts){

        double positiveParent = parentCounts["Yes"];

        double negativeParent = parentCounts["No"];

        double totalParent = positiveParent + negativeParent;

        double parentEntropy = calculateEntropy(positiveParent,negativeParent);

        cout << "Parent Entropy: " << parentEntropy << "\n";

        double childEntropy = 0;

        for(auto it = childCounts.begin(); it != childCounts.end(); ++it){

            string childName = it -> first;

            double positiveChild = it->second["Yes"];

            double negativeChild = it->second["No"];

            double totalChild = positiveChild + negativeChild;

            double childEntropyPart = calculateEntropy(positiveChild, negativeChild);

            childEntropy += (totalChild / totalParent) \* childEntropyPart;

        }

        cout << "Weighted Child Entropy: " << childEntropy << "\n";

        double informationGain =   parentEntropy - childEntropy;

    cout << "Information Gain: " << informationGain << "\n";

    return informationGain;

}

int main(){

    ifstream in("info-gain.csv");

    string line, day, outlook, temp,humidity,wind, playgame,value;

    map<string,int>parentCounts;

    map<string,map<string,int> >childCounts;

    if(!in.is\_open()){

        cerr<<"Error in opening the input file"<<endl;

        return -1;

    }

    int i = 0;

    string childname;

    int choice;

    while(getline(in,line)){

        stringstream str(line);

        getline(str, day, ',');

        getline(str, outlook, ',');

        getline(str, temp, ',');

        getline(str, humidity, ',');

        getline(str, wind, ',');

        getline(str, playgame, '.');

        if(i == 0){

            i++;

            cout<<"Enter the column for which you want to calculate info gain";

            cout<<"Press:"<<endl;

            cout<<"1-Outlook"<<endl;

            cout<<"2-temp"<<endl;

            cout<<"3-humidity"<<endl;

            cout<<"4-wind"<<endl;

            cin >> choice;

            continue;

        }

        switch(choice){

            case 1:

                childname = outlook;

                break;

            case 2:

                childname = temp;

                break;

            case 3:

                childname = humidity;

                break;

            case 4:

                childname = wind;

                break;

            default:

                childname = value;

                break;

        }

        parentCounts[playgame]++;  //it stores total yes and no of of playgame column

        childCounts[childname][playgame]++;  //it stores the yes and no values for a specific column

    }

    double informationGain = computeInformationGain(parentCounts,childCounts);

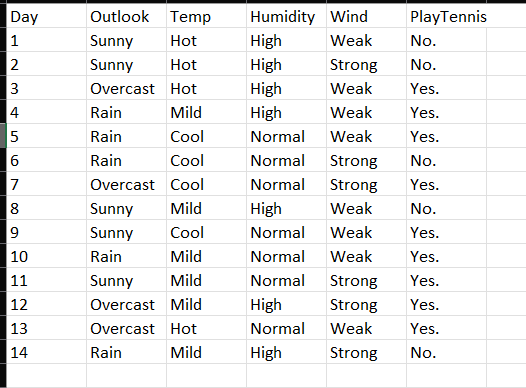
    cout << "Overall Information Gain: " << informationGain << "\n";

    in.close();

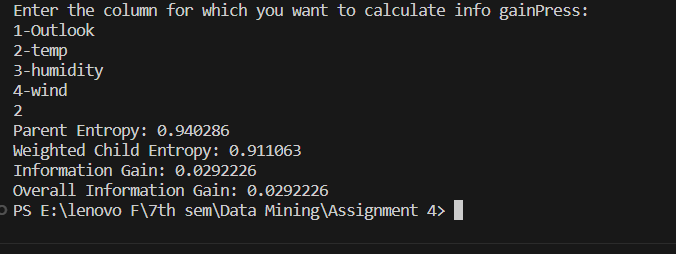
    return 0;

}

**Input dataset**

****

**Output**

****

**Kinme-**