# INFORMATICS INSTITUTE OF TECHNOLOGY

**In Collaboration with**

# UNIVERSITY OF WESTMINSTER (UOW)

BEng/BEng.(Hons) in Software Engineering

Final year Project 2014/2015

**Prototype Report**

For

Project Title: **Identify Inherited Diseases based on DNA (IIDDNA)**

By

Iddamalgodage Don Lahiru Manohara - 2010070

Supervised By: Mr. Achala Chathuranga Aponso

**…………………………..** **…..………………..**

Signature of Supervisor Signature of Student

# Table of Contents

# 1 Disease Prediction Application

Disease prediction application is main feature of the IIDDNA project. The application is capable of predicting possibility of having inherited diseases on given dataset. The dataset is including disease related information. Weka dada mining tool and library was used to build the application.

The problem was considered as a Binary classification problem. The patient has inherited disease classifies as positive and native is patient has not inherited disease. In disease prediction application is considering different binary classification algorithms.

## 1.1 Application Knowledge Flow

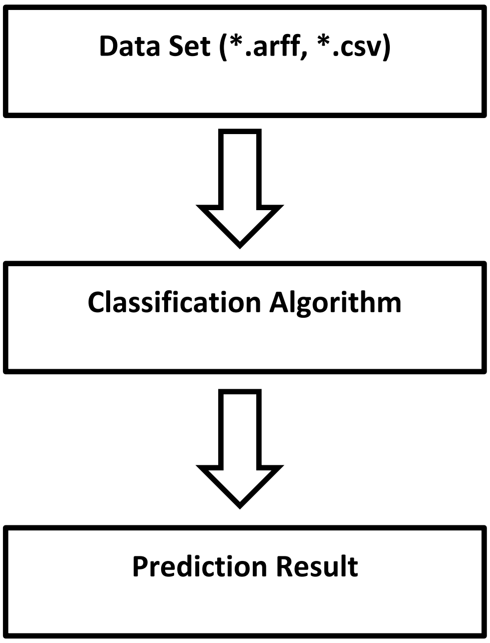


Figure 1 Application knowledge flow

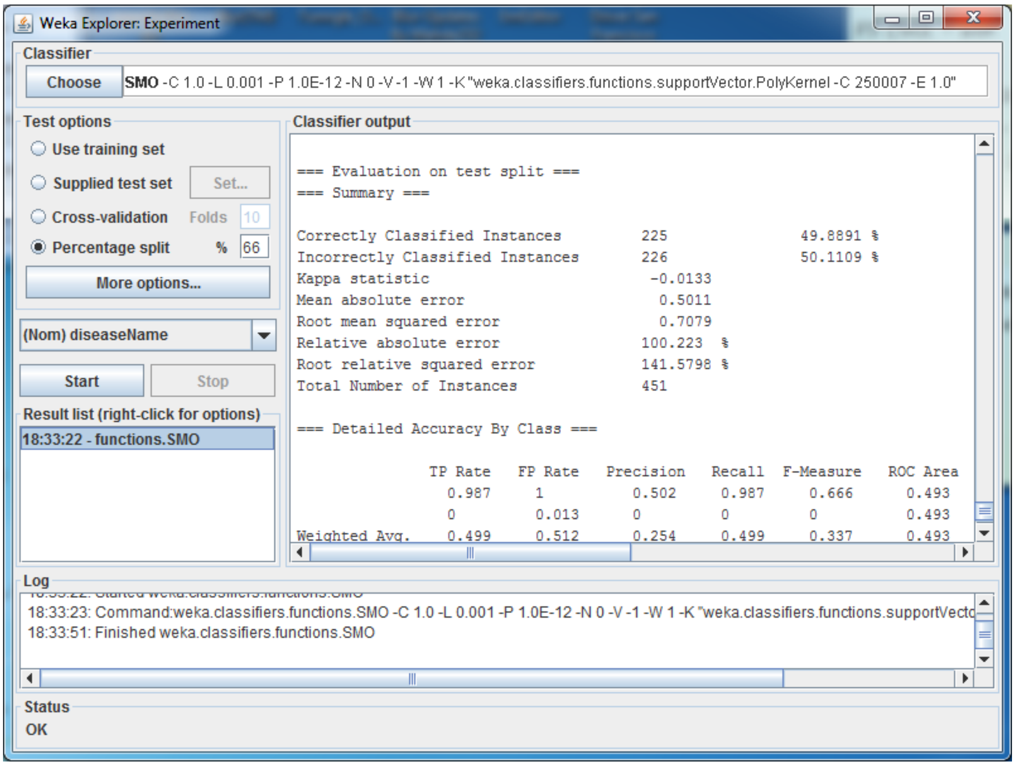
1. **Data Set**
   In classification model is considering two attributes, there are geneSymbol and diseaseName. diseaseName is the class attribute, class attribute has positive and negative data related to inherited diseases. The data set prepared and preprocess according to medical perspective.

## 2. Classification Algorithm

Weka is providing many prebuilt set of algorithms. According to different prediction algorithms and compering their result SMO algorithm (Support Vector Machine) is displayed good accuracy.

## 3. Prediction Result

The prediction result is displaying in following screen shot.



SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -C 250007 -E 1.0"

○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

(Nom) diseaseName

18:33:22 - functions.SMO

```
=== Evaluation on test split ===
=== Summary ===

Correctly Classified Instances         225               49.8891 %
Incorrectly Classified Instances       226               50.1109 %
Kappa statistic                         -0.0133
Mean absolute error                      0.5011
Root mean squared error                  0.7079
Relative absolute error                100.223  %
Root relative squared error            141.5798 %
Total Number of Instances              451

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area
                 0.987     1         0.502     0.987     0.666      0.493
                 0         0.013     0         0         0          0.493
Weighted Avg.    0.499     0.512     0.254     0.499     0.337      0.493
```

```
18:33:22: Started weka.classifiers.functions.SMO
18:33:23: Command:weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVecto
18:33:51: Finished weka.classifiers.functions.SMO
```

Status
OK

Figure 2 Prediction results

# 2 Disease Genes Prediction Application

Disease genes prediction application is a subprogram of the IIDDNA project. This application was build using Genetic Algorithm based software framework calls JGAP (Java Genetic Algorithm Package). Application is explaining the concept of genes association of the particular disease. In genetic inherited diseases causes many genes interaction and other conditions. The problem is difficult to find closely related set of the genes in disease.

The solution is genetic algorithm based search method to find out optimal set of genes related to particular disease.

## 2.1 Process of the Application

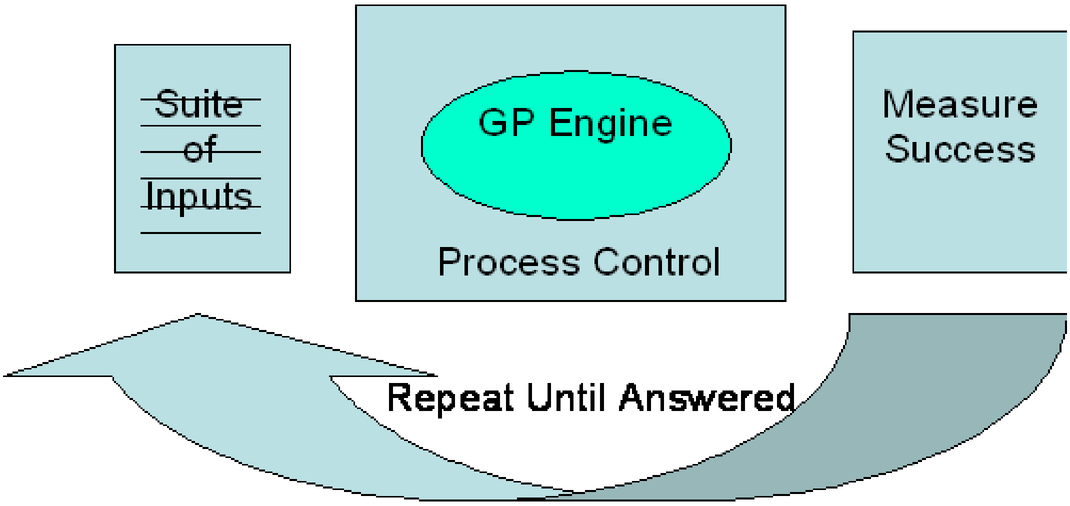


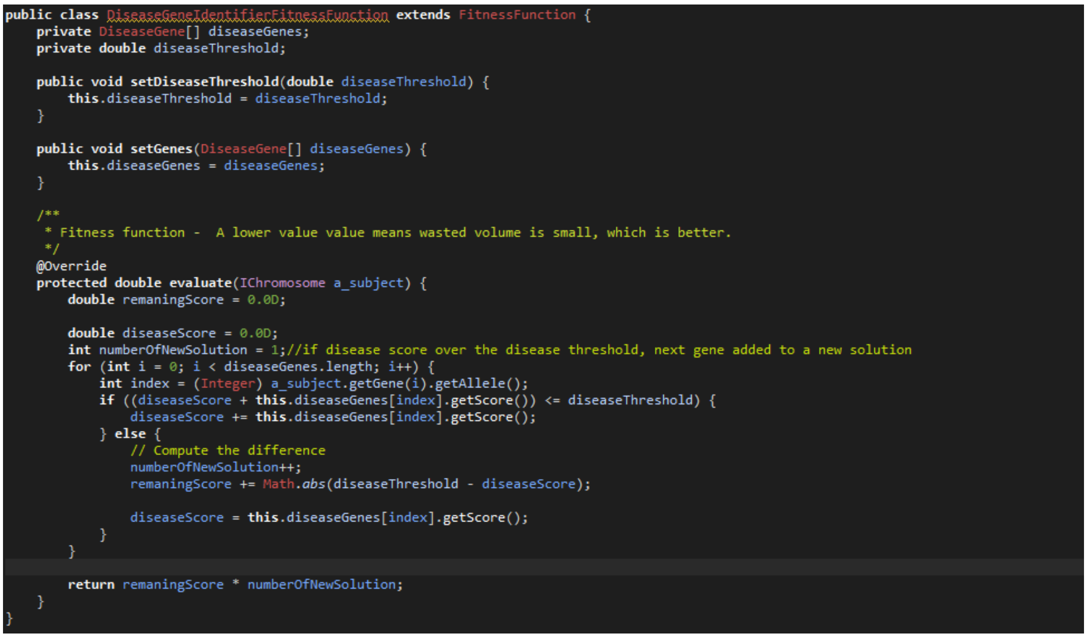
Figure 3 Process of the genetic algorithm

1. **Suite of Inputs**

   The application is getting inputs as genes and score related to particular disease. Score is displaying the how one of the gene important to the disease. The data set input from the excel file.

## 2. GP Engine

The following code is displaying the fitness function of the genetic algorithms related to disease gene identification.

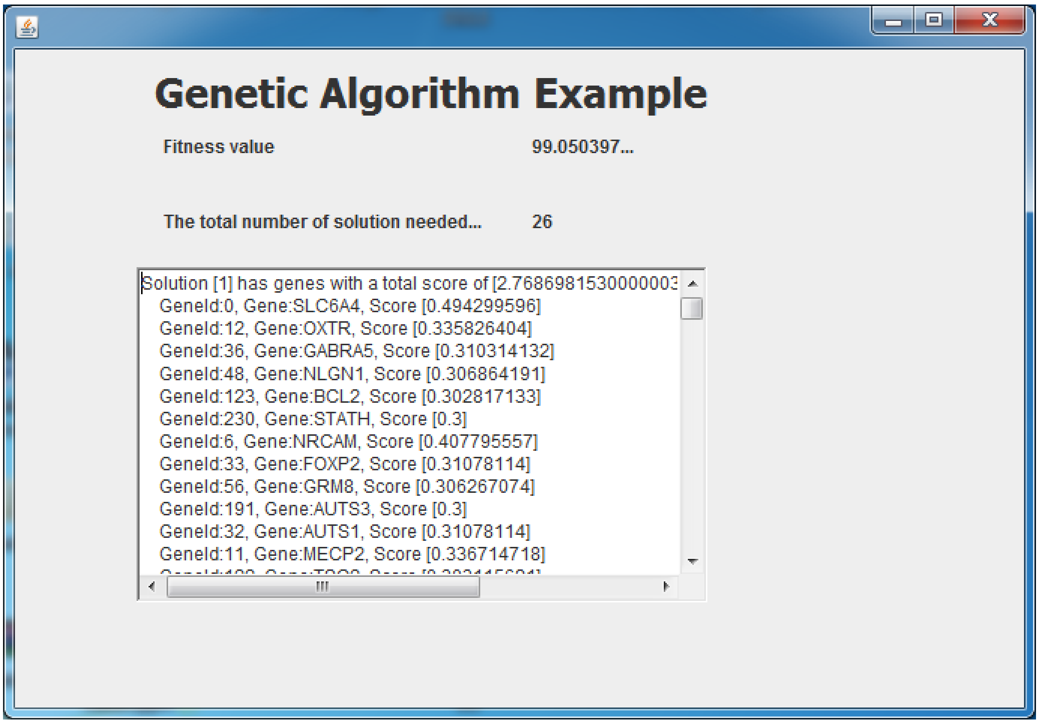


```java
public class DiseaseGeneIdentifierFitnessFunction extends FitnessFunction {
    private DiseaseGene[] diseaseGenes;
    private double diseaseThreshold;

    public void setDiseaseThreshold(double diseaseThreshold) {
        this.diseaseThreshold = diseaseThreshold;
    }

    public void setGenes(DiseaseGene[] diseaseGenes) {
        this.diseaseGenes = diseaseGenes;
    }

    /**
     * Fitness function -  A lower value value means wasted volume is small, which is better.
     */
    @Override
    protected double evaluate(IChromosome a_subject) {
        double remaningScore = 0.0D;

        double diseaseScore = 0.0D;
        int numberOfNewSolution = 1;//if disease score over the disease threshold, next gene added to a new solution
        for (int i = 0; i < diseaseGenes.length; i++) {
            int index = (Integer) a_subject.getGene(i).getAllele();
            if ((diseaseScore + this.diseaseGenes[index].getScore()) <= diseaseThreshold) {
                diseaseScore += this.diseaseGenes[index].getScore();
            } else {
                // Compute the difference
                numberOfNewSolution++;
                remaningScore += Math.abs(diseaseThreshold - diseaseScore);

                diseaseScore = this.diseaseGenes[index].getScore();
            }
        }

        return remaningScore * numberOfNewSolution;
    }
}
```

Figure 4 Fitness function code snippet

## 3. Measure Success

Measuring success and out putting the highest fitness values included solutions. The following screen shot is displaying the final result.



Genetic Algorithm Example

Fitness value 99.050397...

The total number of solution needed... 26

```
Solution [1] has genes with a total score of [2.76869815300000003
   GeneId:0, Gene:SLC6A4, Score [0.494299596]
   GeneId:12, Gene:OXTR, Score [0.335826404]
   GeneId:36, Gene:GABRA5, Score [0.310314132]
   GeneId:48, Gene:NLGN1, Score [0.306864191]
   GeneId:123, Gene:BCL2, Score [0.302817133]
   GeneId:230, Gene:STATH, Score [0.3]
   GeneId:6, Gene:NRCAM, Score [0.407795557]
   GeneId:33, Gene:FOXP2, Score [0.31078114]
   GeneId:56, Gene:GRM8, Score [0.306267074]
   GeneId:191, Gene:AUTS3, Score [0.3]
   GeneId:32, Gene:AUTS1, Score [0.31078114]
   GeneId:11, Gene:MECP2, Score [0.336714718]
```

Figure 5 Disease gene identification application

# 3 Problem Encountered

- The main problem was encountered when finding the data set for predict the disease.
- Data preprocessing and preparing was a difficult part after finding the correct data set.
- Selecting correct prediction algorithm was also difficult and some algorithms did not display the good results.
- Writing the fitness function for find out correct genes set was difficult and there was no clear documentation available for that.