UNIVERSITY OF ZAGREB
FACULTY OF ELECTRICAL ENGINEERING AND COMPUTING

MASTER THESIS No. 1030

Detecting Complex Diseases Using Genetic Features

Sofia Čolaković

Zagreb, June 2015.

CONTENTS

1. Introduction

Multifactorial diseases are the diseases whose occurrence is influenced by several factors which can be of genetic or environmental origin. The cause of the diseases such as type 2 diabetes, asthma or rheumatoid arthritis is therefore hard to determine. Genetic factors that influence these diseases are not localized only on one chromosome or part of a chromosome. This makes the discovery of factor groups whose interactions influence the disease very hard. Finding these interactions presents an unsolvable task for statistical analysis methods commonly used in this area.

This thesis proposes a solution for rheumatoid arthritis discovery based on mutual information, genetic algorithm and machine learning. Mutual information and genetic algorithm are used for selecting informative genetic markers from a large dataset. Afterwards, a machine learning algorithm is used for classification of the individual as affected or not affected by the disease. A method for deriving an optimal goal function based on genetic programming is introduced and examined.

In the next chapter, biological background of the problem is explained along with a description of used genetic features. In the third chapter follows an overview of similar papers with a focus on one particular paper that had inspired this thesis. Fourth chapter presents the implementation of the solution and used algorithms. In the fifth chapter, results from the experiments are shown and discussed.

2. Biological Background

Since The Human Genome Project ended in 2003 more and more research focuses on links between disease affection and patient's genotype. Discovering genetic causes of a disease can help greatly in early diagnostics and prevention as well as in drug research.

In this thesis, data from Genetic Analysis Workshop 15 (GAW 15) is used. It includes both artificial and real data sets with genome-wide markers. This kind of data is used in genome-wide association studies (GWAS).

GWAS is an approach that uses genetic markers, most usually single-nucleotide polymorphisms (SNPs) to find out what markers are linked to a trait. A trait can be a disease affection status or response to a drug. Markers are sampled from an entire genome so no previous knowledge about what features are informative is introduced. GAW 15 database consists of two types of markers, SNPs and microsatellites, that will be described in detail later in this chapter. Goal of this thesis is to use SNP features to determine what genes are causing rheumatoid arthritis.

2.1. Genome-Wide Association Study (GWAS)

In genome-wide association studies participants are divided in groups based on a trait, e.g. disease affection status or response to a drug. Then, DNA of different groups is compared to find markers that are associated with that trait. Normally there are two groups of participants: cases (people affected by a disease) and controls (people that are not affected). This approach is called phenotype-first because the initial grouping is based on a phenotype feature, e.g. disease affection.

In GWAS, DNA sequences are not compared directly, instead, SNP arrays are used. Environmental factors are also commonly added to the feature array. Factors like age, sex, BMI index or whether the individual is a smoker are important risk factors. Genome-wide association studies test whether one SNP allele is more frequent in cases than in controls. If so, it implies that this SNP is associated with a disease. The region where the associated SNP is located is considered to influence the risk of a

disease.

Prior to GWAS, studies could detect genes responsible for single-gene disorders using genetic linkage, but this wasn't possible for complex diseases. Using genome-wide features enables finding multiple genes that influence the disease and are located all over the genome. Genome-wide association studies became widely used with the advance in DNA sequencing and the establishment of biobanks, repositories of human genetic material. Large collections of DNA samples became easier to obtain.

Typical approach to GWAS includes using odds-ratios. It's a ratio between two odds, odds of the disease for individuals with a specific allele and odds of the disease for individuals without that allele. Allele is considered to be associated with a disease if odds-ratio in cases group is much higher than in controls. Odds are usually calculated using a chi-square test.

Allele odds are often represented in a Manhattan plot with a logarithmic scale. Image 2.1 shows the Manhattan plot of markers responsible for various cardiovascular diseases taken from [11]. The image clearly shows what SNPs are more informative. Also, it can help in determining a threshold for associated alleles.

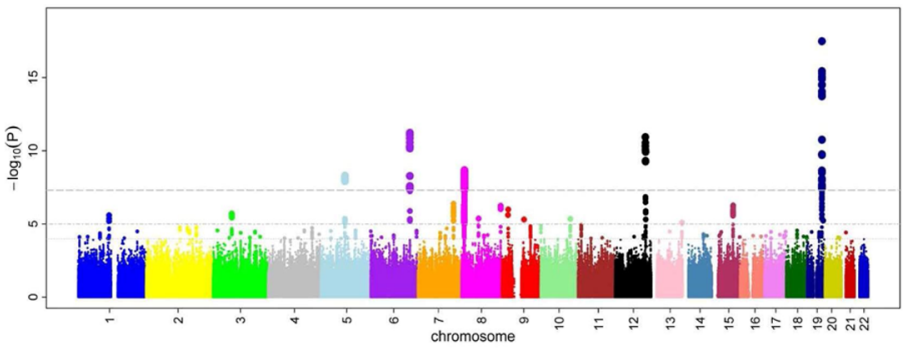

Figure 2.1: Manhattan plot of association between loci and risk of cardiovascular diseases

Various attempts have been made to catalogue all SNPs and diseases they are associated with. The biggest GWA study was the Welcome Trust Case Consortium (WTCC) study in 2007. It included cases for seven different diseases: coronary heart disease, type 1 diabetes, type 2 diabetes, rheumatoid arthritis, Crohn's disease, bipolar disorder, and hypertension. Genes influencing many of these diseases were discovered, including 3 genes influencing rheumatoid arthritis [6].

Although GWA studies discovered many genes that affect the risk of developing the disease, they weren't able to account for most of the disease occurrences. Experiments

done on identical twins estimate heritability at more than 90% for autism and more than 80% for schizophrenia. Also, genetics makes a major contribution to disorders such as obesity, diabetes and heart disease [3]. So far GWA studies aren't able to account for the big majority of heritability. This could imply that more complex models are needed for this kind of a problem.

2.2. Single-Nucleotide Polymorphism (SNP)

Single-Nucleotide Polymorphism (SNP) is a one-base variation in DNA sequence. For example, if DNA sequences AATTCGTA and AGTTCGTA, taken from different individuals differ in only one nucleotide, this nucleotide is an SNP. Variations of that nucleotide, A and G in this case, are called alleles. SNPs have, almost exclusively, only 2 alleles. For this variation to be recognised as an SNP, it has to be present in at least 1% of the population.

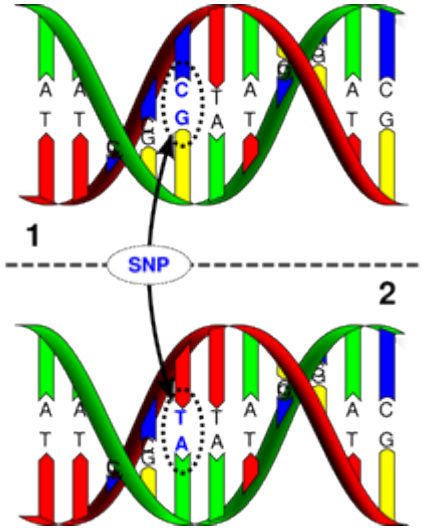

Figure 2.2: Single-Nucleotide Polymorphism (SNP)

SNPs can be found in all genome areas, coding and non-coding gene regions and intergenic regions, but are more frequent in non-coding regions. The reason is that a variety in coding regions tends to be replaced with the more "successful" allele through natural selection. Based on their location SNPs can be categorized as:

1. Linked SNPs - located in intergenic regions. They don't affect protein synthesis, but can be related with a drug response or tendency to develop a disease.

2. Causative SNPs - located in a gene. They affect protein synthesis, tendency

to develop a disease and drug response. Depending on the region the gene is located, there are:

(a) Coding region SNPs - They don't necessarily affect the structure of a protein. This is possible due to redundancy of genetic code inside a gene coding regions. They are categorized as synonymous, which do not affect protein's structure and non-synonymous that do.

(b) Non-coding region SNPs - SNPs which are located in the gene's regulatory region. They influence the amount, timing and placement of a produced protein.

SNPs are responsible for 80% of variations between two individuals [5]. For that reason, they make very good genetic markers. The problem that arises when using SNPs is their number. It's believed that humans have 10 millions SNPs within their genome [1]. In this thesis dataset with 18374 SNP features is used.

Another genetic features that can be used for GWAS are microsatellites. They can predict the density of SNPs in some area.

2.3. Microsatellite

A microsatellite is a repetitive DNA sequence where a small sequence of 2 - 5 base pairs is repeated multiple times. Usually the sequence is repeated 5 - 50 times. Figure 2.3 shows a microsatellite where the sequence "CA" is repeated 6 times.

Figure 2.3: Microsatellite

Microsatellites occur at many places within the genome and have a surprisingly high mutation rate and diversity in the population. Just like SNPs, microsatellites can be found in intergenic regions, as well as in coding and non-coding regions of a gene. The length of a microsatellite from a coding region can influence the length of a protein sequence. Also, the length of a microsatellite from a regulatory part of a gene can affect gene expression. Microsatellites are widely used in DNA profiling and forensic identification. Also, they are good genetic markers and can be used in GWAS. Using

microsatellites in GWAS has some advantages over using SNP markers. Because of the smaller number of microsatellites in the genome, feature arrays become smaller and make algorithm training easier. Nevertheless, SNPs are usually used in genome-wide studies because they are more informative.

3. Related Work

Typical approach to GWA studies is based on statistical analysis. Odds-ratio is a perfect example of it. Although these research have gained some success in discovering disease inductors, they have limited possibilities in predicting connections between genetic factors and the disease. In statistical analysis SNPs are considered one at the time which results in linear analysis [8]. That leads to a model too simple for the discovery of gene to gene interactions that increase the risk of a disease. For this reason, more and more research focuses on stochastic methods like evolutionary and machine learning algorithms. They are able to model more complex relationships between genetic and environmental factors. This thesis combines two stochastic approaches to the problem: feature selection using genetic algorithm (GA) and classification using machine learning (ML). In this chapter, several related papers will be described. They will be grouped into chapters by the type of approach.

3.1. Feature Selection Using GA

The starting point for this thesis was [10]. Its goal was to discover genetic and environmental factors involved in type 2 diabetes and obesity. Features were derived from 3652 SNPs collected from a genome-wide scan and 2 environmental factors. Only genetic data from affected people was taken into consideration.

Approach was similar to the approach in [12]. Genotypes were clustered in groups whose number can vary. Each cluster represents one combination of genetic and environmental factors that influences the disease. From each cluster one rule can be derived, e.g. "genetic factors A and D along with environmental factor E cause the disease". This approach allows modelling of complex relationships between the factors. Risk factors were discovered in 2-phase approach: feature selection using evolutionary algorithm and clustering phase using KMeans algorithm.

In the thesis, features were based on hereditary properties of the genetic factors. For each pair of relatives and each gene, gene similarity score was calculated. If the

similarity score was bigger than expected (e.g. similarity bigger than 0,5 for siblings) corresponding element of binary matrix was equal to 1. This binary matrix along with environmental features composed the feature set the size of 400x1000 + 2 features.

Features were selected using genetic algorithm. GA individual is a binary array of length equal to a feature set. If element corresponding to a feature equals 1 that feature is selected, if the feature equals 0, it's not. Fitness function in GA was based on a support measure that was also used in this paper.

In the paper, sophisticated GA operators were used. They were constructed to encourage smaller feature sets and to ensure that the searched space is as big as possible.

For crossover, Subset Size-Oriented Common Feature Crossover Operator (SSOCF) that specializes in subset selection problems, is used. It tries to keep useful informative blocks of features by producing an offspring that has the same distribution of features as it's parents.

For mutation, biased operator was used that mutates features with different probabilities. 0 is flipped to 1 with the probability of 0,5 and 1 is always flipped to 0. This operator favours small feature sets because the disease is believed to be influenced by a small number of features.

Individuals are selected using tournament selection with a sharing scheme. It boosts a selection chance of individuals in less searched areas of a problem space. Every solution has an appointed parameter that indicates how crowded is it's neighbourhood.

After smaller set of features was selected, individuals were clustered in n clusters using KMeans algorithm. Features that are shared among individuals of a cluster are believed to be one of the combinations responsible for disease affection. Solution was tested on both artificial and real data. The results obtained from artificial dataset are consistent with the official solution but no accuracy can be acquired by this procedure.

3.1.1. Other GA Approaches

The paper [7] proposes different kind of method using GA and support vector regression (SVR). They used the p-value of the Spearman's correlation to group the SNPs. After that, SVR with Pearson universal kernel is used to determine the group with the best performance. After that GA using the wrapper method selects the features one more time trying to eliminate the features that don't contribute to the solution. That means that quality of GA's solution is measured with the output from the SVR classifier.

Another approach based on genetic algorithm was introduced in [12]. Genetic algorithm was used to search SNP groups that are associated with the bipolar disorder. The algorithm is constrained by the structure of a gene interaction network, where each node in the network represents a gene and all SNPs that are located within or near that gene [12]. For fitness function the Chi-square test was used due to it's simplicity and fast calculation. This approach managed to explore the vast majority of a search space, 85-99% from 636169 features had been visited. This is possible thanks to the gene interaction network constraints that dismissed most of solutions as invalid.

3.2. Machine Learning Approach

In recent years, ML approach has gained more popularity in GWAS. Various methods have been proposed, but Support vector machine (SVM) has singled out as the most popular. It's ability to handle data sets of big dimensionality has proven to be very helpful in handling thousands of SNPs. Other ML approaches include Random forests (RF), Multifactor dimensionality reduction (MDR), Naive Bayes and Artificial neural networks (ANN). [8] states that using ML methods on GWA problems should be approached with caution. One method could show excellent results on one problem but fail terribly on the other. In all the cases, adjusting the method is necessary.

3.2.1. Support Vector Machine

Support Vector Machine (SVM) applications in GWAS include studies of various diseases with data sets of different length.

In the paper [13] risk factors for multiple myeloma were researched. SVM with cross-validation was applied on the dataset of 3000 SNPs sampled from an entire genome. Results report 71% accuracy.

[14] applied multiple algorithms on the problem of finding genes that influence chronic hepatitis. SVM scored worse than a decision tree solution with the accuracy of 67.53% while the decision tree scored 72.68%.

Multiple studies used only SNPs that were believed to influence the disease. This approach introduces previous knowledge about the subject which can either contribute to or spoil the result, depending on the accuracy of this information. Paper [4] report the accuracy of 70% using SVM on classification of type 2 diabetes affliction. In the paper dataset of 408 SNPs already associated with diabetes was used.

3.2.2. Random Forest

Random forest (RF) is a robust machine learning technique for both regression and classification. RF is a good choice to use in GWA studies because of it's ability to handle big sets of data and insensitivity on missing data. In paper [9] random forests were applied on 300,000 SNPs dataset. It found responsible genes reported by other studies along with some other unreported genes.

3.2.3. Multifactor Dimensionality Reduction

Multifactor dimensionality reduction (MDR) is a method developed for discovering discrete variable interactions. The algorithm reduces feature dimensionality by replacing two or more variables with one new variable. This method is most commonly used in GWA studies as it can represent high-order gene-gene interactions. In GWAS, a group of two or more SNPs is selected. After that, combined odds-ratios are calculated for all combinations of the selected SNPs. Combinations with the biggest odds-ratios are considered to influence the disease. This method is reported to give good results, but can be complex to compute [8].

4. The Proposed Solution

The goal of this thesis was to propose a solution for detecting gene to gene interactions which increase the risk of rheumatoid arthritis. In this thesis both artificial and real datasets from GAW 15 were used. The data in the artificial dataset consists of 1,800,000 examples of 9,187 SNP arrays. Each SNP consists of 2 alleles, each inherited from one parent. So the size of a feature set is 18,374 elements. The main obstacle for successful application was the dimensionality of a feature vector.

Three-phase solution is proposed. It includes preprocessing using mutual information, feature selection using GA and classification with ML algorithm. In the first phase, most informative features are selected. This lowers the number of features significantly and offers GA the search space it can handle. The next phase decreases the number of features even further, leaving only the ones that are associated with the disease. Then, the ML classifier classifies the individuals as affected or not affected. Classifier's goal is to model the interactions between selected features.

In this chapter the solution is described in detail through these three steps.

4.1. The Data

Features used in this thesis represent SNPs sampled from the whole genome. Each element of a feature vector represents one SNP allele. Since all SNPs have only two alleles, features can be represented with a binary array.

Individuals are labelled by their affection status. Label array is also binary: 0 means that individual is not affected, 1 that it is affected.

4.2. Mutual Information

Mutual information (MI) is a measure commonly used in feature selection and signal processing. It indicates variables' mutual dependence. Mutual information between a

feature (SNP) and a label is calculated as:

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) log \frac{p(x,y)}{p(x) * p(y)} \tag{4.1}$$

x - feature value

y - label value

$p(x,y)$ - joint probability of x being a feature value and y being a label value

$p(x)$ - probability of x being a feature value

$p(y)$ - probability of y being a label value

MI measures the information that is shared between two variables, X and Y. It states how much is known about the value of X if the value of Y is familiar.

In the preprocessing step, MI is calculated between all SNPs and the label variable. SNPs are sorted by the information they share with the label. Only n SNPs with the highest MI score are kept. Various values of n are tested, $n = \{20, 50, 100, 200, 500, 1000, 2000\}$.

This measure is not aware of gene to gene interactions. Nevertheless, it's a crucial step because it decreases the number of features significantly. It is introduced because neither GA nor ML algorithms could cope with the original search space size.

4.3. Genetic Algorithm

Genetic algorithm (GA) is a method of artificial intelligence that mimics natural selection. The population of solutions undergoes selection, crossover and mutation in order to find an individual with the best fitness score. GA achieves good results on optimization and search problems. Also, it can be used in feature selection problems where each individual represents one feature subset.

In this thesis, GA should select only the features that influence rheumatic arthritis. Genetic operators are used to improve feature selection. Pseudocode for algorithm used in this thesis is given in figure 1.

GA is trained with the population of 100 individuals. Mutation factor is set to 0,05. It represents the probability of mutation on one element of individual's array. The algorithm terminates after 50 generations or 10 generations without fitness change.

Algorithm 1 Genetic Algorithm

function GENETICALGORITHM($sizePop, fitnessFunc$)
  $t \leftarrow 0$
  $P \leftarrow createPopulation(sizePop)$
  while $(!terminationCondition)$ **do**
    $newP = []$
    repeat
      $a, b, c \leftarrow select(P)$
      $a, b \leftarrow tournamentSelection(a, b, c)$
      $x \leftarrow crossover(a, b)$
      $x \leftarrow mutation(x)$
      $newP \leftarrow a, b, x$
    until $(populationReplaced)$
    $t \leftarrow t + 1$
    $P \leftarrow newP$
  end while
end function

4.3.1. Individual Representation

In this thesis, GA is used to select most desirable features from the feature vector. Each individual in GA has to represent one solution, i.e. one feature selection. Therefore individual is represented by a binary array of length of a feature vector. If element corresponding to a feature equals 1, the feature is selected. If it equals 0, feature is not selected. When the initial population is created, the probability of an element being equal to 0 is randomly selected, $p \in [0, 1]$. There is p chance that each element of an array will be equal to 0, and $1 - p$ that it will be equal to 1.

4.3.2. Fitness Function

In GA, quality of an individual is estimated by using a fitness function. A fitness function has to be adequate in order to get a suitable solution. This is not always an easy task, especially if filter method for feature selection is used. When using a filter method, feature selection and classification are done in two separate steps. Usually, features are selected by their correlation with the label. This method takes less computer resources and is resistant to overfitting, but can select redundant variables.

Besides the filter method, features can be selected using a wrapper method. Here

feature subset is evaluated using classification output. Wrapper method is able to detect interactions between multiple variables, but uses much more computer resources than filter method and is prone to overfitting.

In this thesis, multiple fitness functions are tested. Fitness functions based on support and entropy are filter methods while fitness based on factor combination is a wrapper method because it includes classification output.

Fitness Based on Support

Support is a measure often used in data mining. It describes the presence of a feature subset in a dataset. It's the ratio between the number of individuals with all features from a subset and the number of individuals with at least one feature from the subset. If we have a set of 2 features: A and B, 3 out of 5 people have A, 3 of 5 have B and 2 of 5 people have both A and B. The support for this feature set will be $\frac{2}{4}$.

The paper [10] proposes a fitness measure based on support for feature selection. The measure applied here is based on it, with a couple of changes that encourage bigger feature subsets. It is calculated as:

$$Fitness = (1 - S) * SF * \frac{\frac{T}{5} - SF}{T} + 2S * SF * \frac{\frac{T}{5} - SF}{T} \tag{4.2}$$

S - support
SF - the number of selected features
T - total number of features

This measure has several goals that are trying to be met. It favours the subsets that have big support in the dataset. Also, encourages big feature subsets by multiplying the fitness with SF, but punishes ones bigger than 20 features. Biologically, the number of features that influence the disease should be small. Nevertheless, the fitness function favours bigger subsets because large subsets with high support are unusual and could be the sign of an association between features. The fitness function favours large subsets with good associations, but punishes if a subset becomes too large [10].

Fitness Based on Entropy

In information theory, entropy is a measure of information contained in an event. It's a measure of uncertainty of an event's outcome. Fitness function uses conditional entropy to measure the connection between a genetic marker and the label variable.

Entropy is calculated as $H = -p*logp$. If $log_2 p$ is used, it represents the number of bits needed for event representation. If a fair coin is tossed, the probability of getting tails is $\frac{1}{2}$ so $H = \frac{1}{2}$. On the other hand, if a coin has two heads, the probability of getting heads is 1. Also $H = 0$, there is no uncertainty about this event.

Conditional entropy is calculated as:

$$H(X|Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) log \frac{p(y)}{p(x,y)} \tag{4.3}$$

$p(x,y)$ - joint probability of x being a feature vector and y being a label value
$p(y)$ - probability of y being a label value

$p(x,y)$ for a feature vector is calculated as:

$$\prod_n p(x_n, y) \tag{4.4}$$

where n represents feature vector element.

$p(y)$ for a feature vector is calculated as:

$$\prod_n p(y_n) \tag{4.5}$$

where n represents feature vector element.

Entropy is always bigger or equal to zero.

$H(X|Y)$ represents the uncertainty of the variable X if Y is known. In feature selection, the goal is to maximize the entropy in order to get as much diverse feature subset as possible.

Fitness Based on Combination of Factors

As presented in the next section, fitness functions based on entropy and support weren't able to provide a better subset of features than the one acquired only by preprocessing. For that reason, new fitness function is introduced. Fitness function based on combination of factors uses factors like classification accuracy, number of features in subset, support and entropy to derive a better feature subset.

Fitness is calculated as:

$$Fitness = a * P + b * f(R) + c * S + d * E \tag{4.6}$$

P - classification accuracy
R - number of features in subset

$f(R)$ - function of R
S - support
E - conditional entropy

Parameters a, b, c and d are factor weights. They reflect the importance of each factor. Their influence on quality of selection is tested in the chapter 6.

4.3.3. The Selection Operator

Selection operator plays an important role in GA. It determines the way how individuals are selected to breed, be eliminated or to be kept in the population. In this thesis, tournament selection is used. It's a method based on running tournaments among individuals. A group of k individuals are randomly selected from the population. Best ones are selected for breeding and the weakest are eliminated.

In this thesis tournament selection is implemented with $k = 3$. Worst individual is eliminated and replaced with an offspring of two remaining individuals. This method ensures elitism. The strategy where the best individual is always carried to the next population.

Figure 4.1: Tournament Selection

4.3.4. Crossover Operator

Crossover operator defines how is an offspring generated from it's parents. In this thesis, five-point crossover is used. Five points on parent chromosomes are selected. A child is then generated from segments between these points. First segment is taken from parent 1, second from parent 2, third from parent 1 and so on. Five-point mutation was selected because of the individual's array length. The tests are conducted on individuals of 100 and elements.

This method is fast and simple, yet it's a good model of genetic recombination with the respect to genetic linkage principle. It states that genes that are closer together in the genome have higher probability of being inherited from the same parent [2].

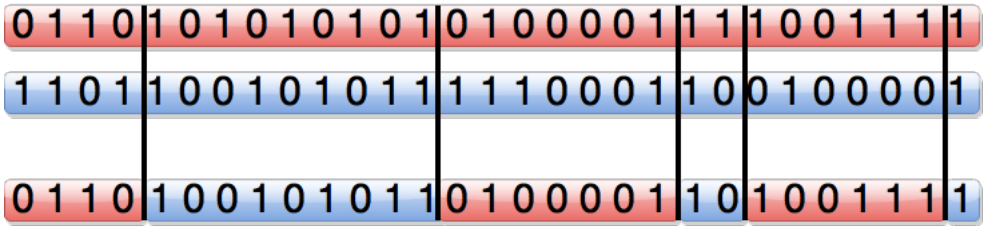

Figure 4.2: GA Crossover

4.3.5. Mutation Operator

In genetic algorithm mutation is often used to maintain genetic diversity in the population. GA without mutation tends to get stuck in a local extreme unable to get out of it. Some individuals are mutated after crossover to prevent the population to become crowded with individuals that are too similar to each other. This thesis proposes the use of point mutation operator. Each element (bit) of an individual's array can be mutated with a certain probability. In this thesis, individual array is binary. So when mutated the bit is flipped, form 1 to 0 or from 0 to 1. The probability of a bit mutation is 0.05. Every 20. bit is going to be flipped.

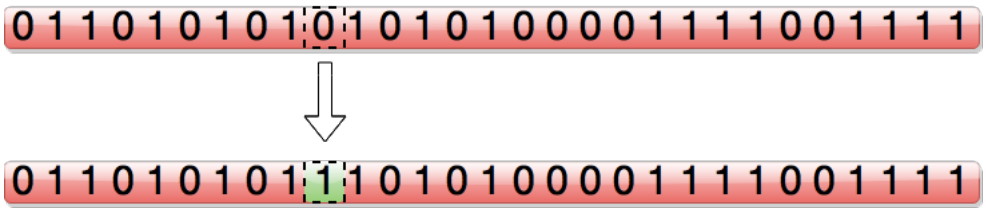

Figure 4.3: GA Mutation

4.4. Genetic Programming

Genetic programming (GP) is a type of evolutionary algorithm that uses operators based on natural selection in order to derive a program that solves a particular problem. Program is represented as a tree-structure where each node represents a mathematical operation, a scaling factor or a free variable. The leaves of a tree-structure always represent variables called terminals in GP. Other nodes are called primitives.

GP has a population of individuals that is improved through generations using crossover and mutation. The quality of an individual is determined by a fitness function. Better individuals have higher chances to be carried to a new generation or reproduced. Just like other evolutionary algorithms, GP also mimics natural selection. The algorithm used for genetic programming is equivalent to the algorithm used in GA, shown in figure 1.

In this thesis, GP is used to derive a fitness function for GA. Variables used in GA fitness function, classification score, support, number of features selected and entropy, are used as terminals. A fitness function in GA is a program represented by a GP individual.

The population of 50 individuals is evolved through generations. In each generation crossover and mutation operators are used to evolve better programs. Individuals are selected for reproduction by tournament selection. Worst individuals are eliminated, but the majority of population gets a chance for reproduction. The algorithm is terminated after 10 generations without improvement or after 50 generations. The maximal tree depth is set to 4 in order to decrease the computational complexity.

4.4.1. Individual Representation

An individual in GP is represented as a tree-structure that consists of nodes. In this solution, a node can be an operation, a constant or a variable. Operations are nodes that have one or two children and determine the mathematical operation on them. Operations used in the solution are addition, subtraction, multiplication, division, cosine and square function.

A constant node is used for scaling. A child of a node is multiplied with a constant ranging from 0 to 1.

A variable node represents a free variable. It doesn't have children, so it is always a leaf of the tree-structure.

A root node is always an operation and leafs are always variables. Other nodes are selected by chance. An example of a GA individual can be seen in figure 4.4.

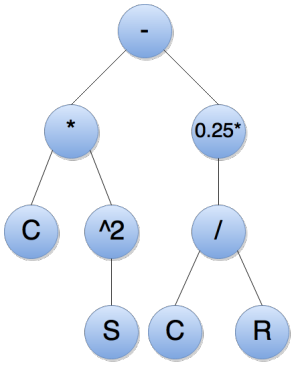

Figure 4.4: GP Individual

This example represents a fitness function:

$$fitness = C * S^2 - 0.25 * \frac{C}{R} \tag{4.7}$$

An individual is evaluated by calling root node's evaluate method with the values of terminals.

4.4.2. Fitness Function

Quality of an individual is evaluated using a genetic algorithm (GA). For every individual, one GA is trained. If the GP program is able to derive a feature subset that has a high classification score and a small number of features, it's considered a good fitness function. The fitness of a GP individual is based on the ability to improve a GA feature selection as well as the classification score. It's calculated as:

$$fitness = P + 0.15 * \frac{T - R}{T} \tag{4.8}$$

P - classification accuracy
R - number of features in subset
T - total number of features

This particular function is used for fitness calculation because it showed good results in tests run with genetic algorithm. The goal is to get the best classification accuracy possible while favouring small subsets of features. The part of the formula favouring small feature subsets is considered less important than classification accuracy and therefore scaled down proportionally.

4.4.3. Selection Operator

Like in GA, tournament selection is used for selecting the individuals that will contribute to the next generation. Tournament selection is a method commonly used in genetic programming. In tournament selection the selective pressure of an algorithm can be adjusted by changing the tournament size and the number of contestants that are going to be eliminated.

In this solution, the population is divided into subsets of three individuals. The worst individual from each subset is discarded while other two are reproduced. The next generation of individuals consists of children created by crossover and their parents. This method ensures elitism, a characteristic where the best individual is always carried to the next population.

4.4.4. Crossover Operator

In GP, crossover is used to combine good individuals in order to get an even better solution. It's conducted by replacing one node of a parent individual's copy with a node from another parent. When replacing the node with another one, the whole subbranch beneath the node is also replaced.

The nodes that are replaced are selected by chance. If a child has the depth bigger than the maximal allowed depth, it is discarded. The procedure is shown in figure 4.5.

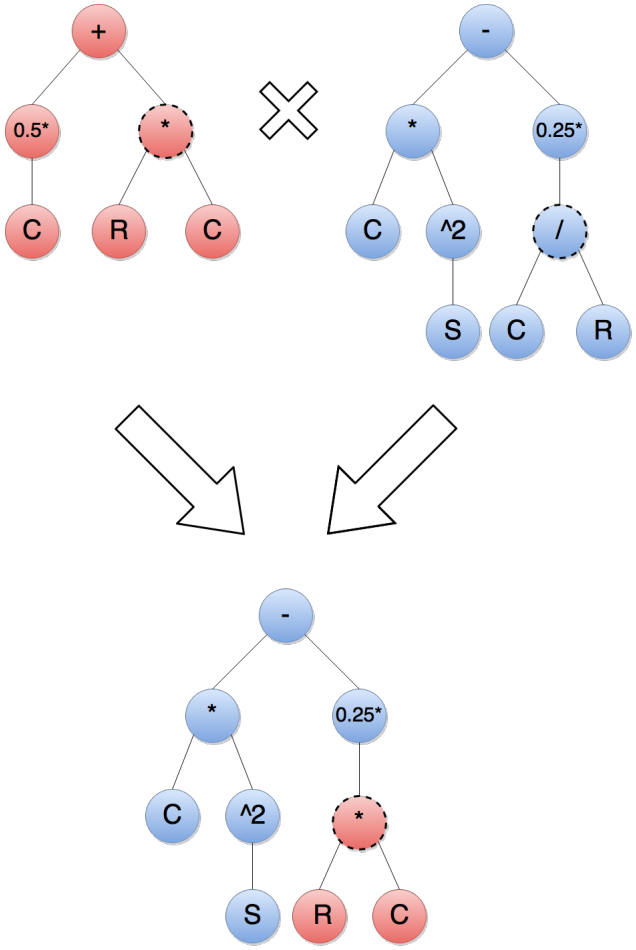

Figure 4.5: GP Crossover

4.4.5. Mutation Operator

A mutation operator is used to prevent the population becoming too uniform. It has the goal of maintaining diversity of solutions in a population.

In this example, an individual is mutated with a 20% chance. One node of an individual is selected randomly and replaced along with the whole branch underneath it. The new branch is created and inserted in its place. The selected node and the old branch are discarded. The procedure is shown in figure 4.6.

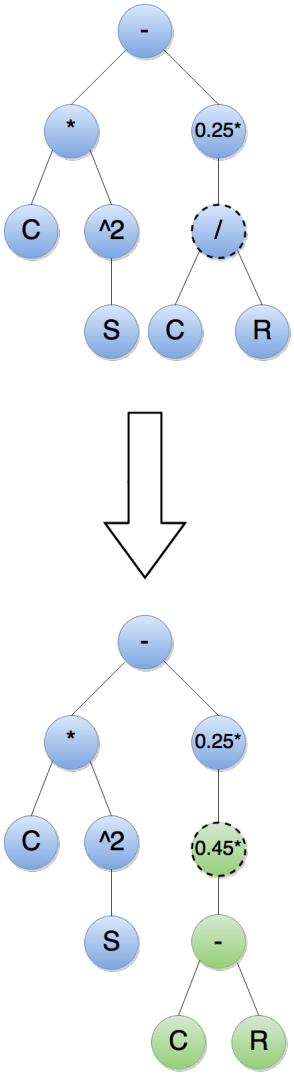

Figure 4.6: GP Mutation

4.5. Classification

Using classification in GWA problems is pretty uncommon. Authors usually only use genetic features from individuals affected by the disease and perform some kind of feature selection. Results are compared to the ones from similar papers and are believed to be correct if they are consistent.

In this thesis, both affected and unaffected individuals are included and classification is used to get a feedback about the quality of a solution. This is crucial for comparing different feature selection techniques done in this thesis. Also, classification score is used in fitness based on combination of factors described in this chapter.

Two machine learning algorithms are used for classification: support vector ma-

chine and KMeans. SVM proved itself to be much more suitable for this task. The majority of evaluations is conducted using a SVM classifier. Data is classified using cross-validation.

4.5.1. Support Vector Machine

Support Vector Machine (SVM) is a supervised learning model used for classification and regression. It's one of the most popular ML methods due to it's efficiency and ability to handle feature sets of big dimensionality. The main idea behind SVM is mapping the linearly inseparable training data into higher-dimensional space where it can be linearly separated.

Apart from linear classification, SVM can also be used for non-linear classification using a kernel trick. In kernel trick, every point from training data is replaced with a non-linear kernel function. Using kernel functions allows modelling of non-linear boundaries between examples.

In this thesis, SVM with Gaussian radial basis function (RBF) kernel is used. The RBF kernel between two samples is calculated as:

$$K(x, x') = exp(\gamma||x - x'||^2) \tag{4.9}$$

where Gamma parameter is defined as:

$$\gamma = -\frac{1}{2 * \sigma^2} \tag{4.10}$$

σ - a free parameter

RBF kernel maps data to an infinite-dimensional space. It has one parameter which defines how much does one training example influences the curve. If γ is too small, model becomes too simple and doesn't model the complexity of an example. If it's too big, model will be prone to overfitting.

SVM with RBF kernel achieves 77,47% accuracy on the test set on the problem of patient classification in this thesis.

4.5.2. KMeans

KMeans is the most commonly used clustering algorithm. Examples are grouped into k groups where parameter k is set in advance. Every group is defined with it's mean value and consists of examples that are closest to it. The algorithm is iterative with alternating two steps, assignment and update. In assignment, an example is assigned

to a cluster whose mean has the smallest Euclidean distance to the example. In the update step, new mean values of cluster elements are calculated. The algorithm is terminated when the solution converges or maximum number of iterations is reached.

In this thesis, examples are grouped into two clusters, affected and not affected. The algorithm didn't achieve good results on the given problem. This shows that interactions between genetic features are too complex for KMeans to handle.

5. Implementation

5.1. Database

The database used in this thesis was originally made for Genetic Analysis Workshop 15 (GAW 15). GAW is a workshop where new methods for genome analysis are presented, compared and discussed. Until recently, statistical analysis was the main subject of the workshop. This is now slowly starting to change in favour of artificial intelligence methods. The database consists of one artificial and multiple real databases collected by several groups from Canada, France and UK.

The artificial database was built to provide a sufficient number of samples for the workshop. It's designed to mimic real data and shows similar results to real database in GAW experiments.

In this thesis, all methods are trained and evaluated on the artificial dataset. It has the advantage of having enough samples and no missing data. The artificial dataset consists of 100 subsets each having approximately 8000 samples. In this solution one of these subsets is used. It consists of 3468 affected and 4532 non-affected individuals.

The database provides additional data about the patients that aren't used in this solution, e.g. age at which the disease was discovered or did the patient smoke. Also, genetic data about patient's family is included so heritability of the disease can be researched.

5.2. Module Description

The solution described in this thesis consists of five modules. The diagram of the solution is presented in figure 5.1.

The first step is parsing the genome data. The parser is used to get an array of genetic features for every patient from the dataset. This step results a list of arrays the size of 18374 features. For each patient a label that indicates affection status is also

provided.

The goal of next three steps is to decrease the number of features while keeping the quality of classification. This makes sense because the number of genetic markers that cause rheumatoid arthritis is low and false positives are not desirable.

The next step is preprocessing where only features with the highest mutual information with the labels are kept. This step proved itself to be crucial because other algorithms used in the thesis responded badly to a feature array that big.

After preprocessing, features are selected one more time in order to get an even smaller feature subset, ideally consisting of only few features. This second phase of feature selection is conducted using a genetic algorithm with different methods of fitness calculation. The choice of a fitness function that encourages the best individuals wasn't the obvious one. Therefore, many methods are tested including the evolution of a fitness function using genetic programming.

The last step is sample classification. The samples are classified into two classes, affected by rheumatoid arthritis and not affected. Two algorithms from scikit-learn library are used for classification, SVM and KMeans. The quality of classification is expressed using accuracy mean. The samples are classified using cross-validation with 5 folds.

Algorithm for mutual information, GA, GP and the parser are implemented as a part of the thesis. The whole solution is written in Python 3.

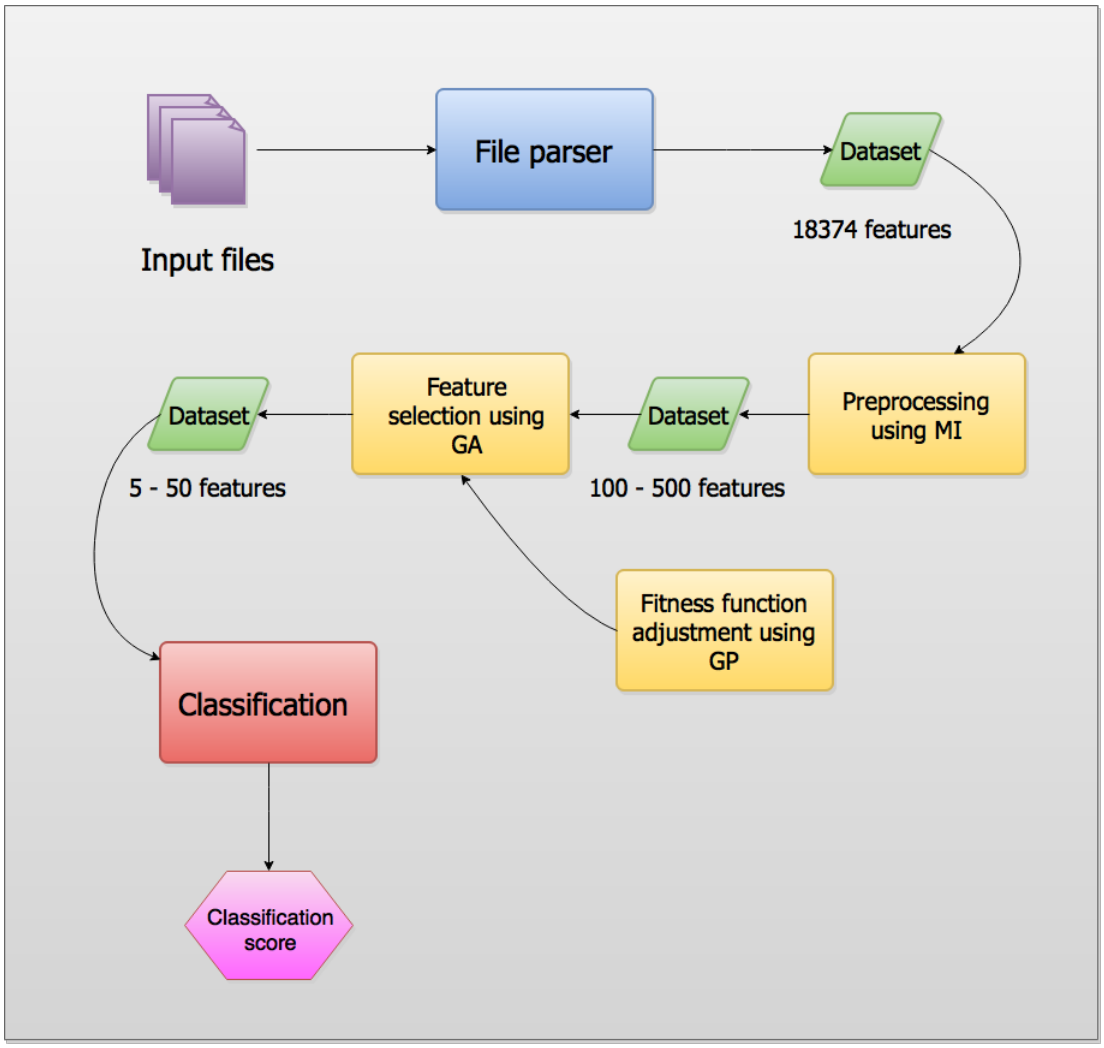

Figure 5.1: Module Diagram

6. Results

In the following chapter previously described methods are tested, discussed and evaluated. In the first section, the influence of preprocessing using mutual information is assessed. Also, two different classification algorithms are compared.

The next section focuses on different methods for fitness evaluation in genetic algorithm. Four different approaches are presented with detail display of relationships between fitness, classification accuracy and number of selected features.

The last section investigates the use of genetic programming in deriving fitness functions used in GA. The quality of fitness functions given from GP is evaluated. The correlation with classification accuracy and number of selected features is investigated.

6.1. Influence of Preprocessing

The graph in figure 6.1 shows classification results on several feature subsets obtained by preprocessing. Subsets are composed of 20, 50, 100, 200, 500, 1000, 2000 and 5000 features with the highest mutual information shared with the label variable. The purpose of this test is to show if feature sets can be safely reduced without loosing relevant data. As shown in table 6.1, the subset of 100 features has only a slightly worse classification accuracy than the subset of 500 features. Smaller feature subsets make the search space smaller and reduce the training of GA significantly. That's the reason why the subset of 100 features is used in further experiments.

The figure 6.1 also shows classification accuracy acquired from a KMeans algorithm. Results are much worse than the ones provided by SVM. KMeans showed not only inability to handle large feature sets, but also poor classification results on datasets of smaller dimensionality. Further testing is therefore done only with an SVM classifier.

feature number	SVM	KMeans
20	0.7746	0.5692
50	0.7746	0.5163
100	0.7748	0.5692
200	0.7760	0.5692
500	0.7765	0.5162
1000	0.7735	0.6696
2000	0.7686	0.4913
5000	0.7180	0.5692

Table 6.1: Preprocessing Results

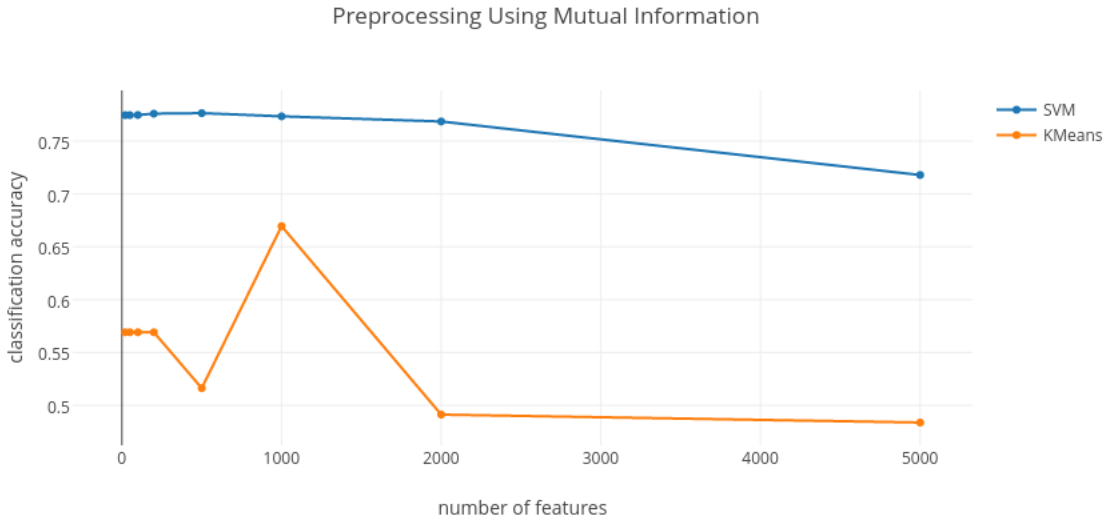

Figure 6.1: Preprocessing Using Mutual Information

6.2. Feature Selection

Features are selected using a genetic algorithm with different fitness functions. Fitness functions based on support, entropy and combination of classification accuracy and the number of features are reviewed in this section. Fitness based on a combination of accuracy and number of features is presented in two variations.

Tests are conducted using GA with the population of 100 individuals. The maximal number of generations is set to 30. The majority of solutions converges very fast, often in the first 10 generations.

6.2.1. Fitness Based on Support

In table 6.2 results from 8 runs of GA with fitness based on support are shown. The table consists of data from the best individuals from each run. Although the number of selected features is pretty low, algorithm isn't able to derive individuals with good classification accuracies. Evolved individuals have accuracy between 57.92% and 62.77%.

The correlation between fitness and accuracy is shown in figure 6.2. It's evident that accuracy declines with the fitness growth. The reason lies in the fact that subsets with more features have lower support. The comparison between support based fitness and other measures can be found in figure 6.6.

run No.	fitness	feature number	accuracy
1	0.2807	7	0.5766
2	0.2829	9	0.5792
3	0.2995	6	0.5798
4	0.2904	4	0.5688
5	0.2685	11	0.5874
6	0.2585	10	0.6278
7	0.2810	6	0.5761
8	0.274	10	0.6185

Table 6.2: Fitness Based on Support

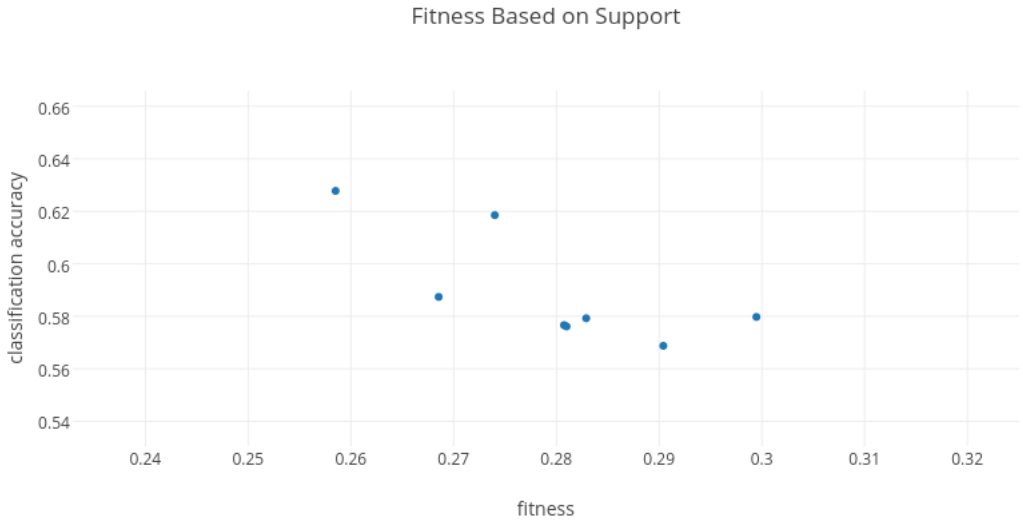

Figure 6.2: Fitness Based on Support

6.2.2. Fitness Based on Entropy

In table 6.3 results from 8 runs of GA with fitness based on entropy are shown. Results show that fitness based on entropy is not suitable for this task. Fitness function favoured very small subsets and produced individuals that select only 1 or 2 features. The individuals achieve classification accuracies between 56.30% and 67.30%. In figure 6.3 a correlation between entropy and accuracy of evolved individuals can be seen. The graph shows no significant relations between these two measures.

run No.	fitness	feature number	accuracy
1	139.7292	2	0.5630
2	163.8607	1	0.5665
3	116.5573	2	0.5680
4	154.2976	1	0.6729
5	163.8607	1	0.5665
6	204.3130	1	0.6229
7	81.7168	2	0.5665
8	141.0565	2	0.5635

Table 6.3: Fitness Based on Entropy

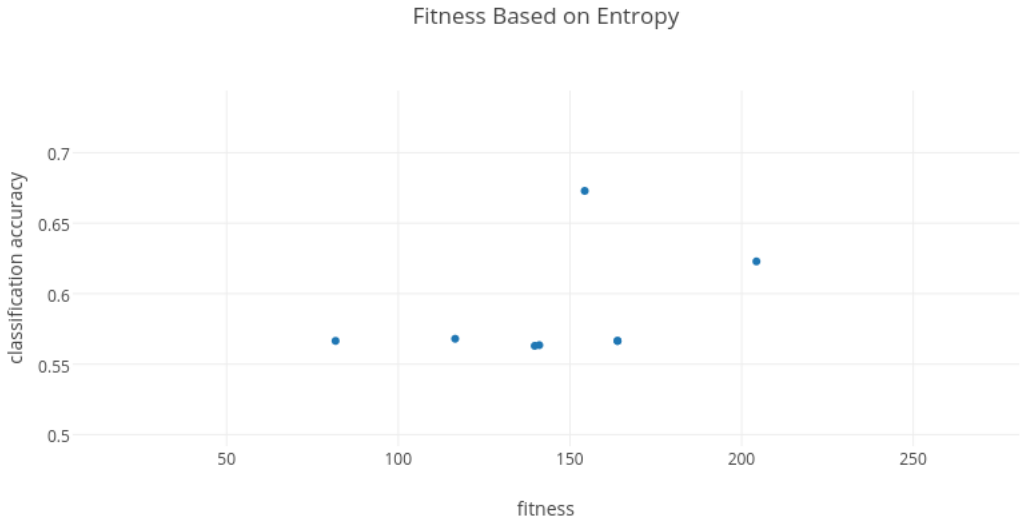

Figure 6.3: Fitness Based on Entropy

6.2.3. Fitness Based on Combination of Factors

In this section, fitness based on factor combination is tested in two variations. In the first one, fitness depends on the inverse of number of features, while in the second one, it depends on the number of features linearly. The first variation is calculated as:

$$Fitness = a * P + b * \frac{1}{R} \tag{6.1}$$

P - classification accuracy
R - number of features in subset
a, b - coefficients

This was the original solution, but it proved itself difficult for adjustment using only linear coefficients. It favours either too small or too big feature subsets. This is evident from graph in figure 6.4.

With a coefficient value $b = 0.175$, two types of solution are evolved, ones with number of features smaller than 5 and the other with number of features bigger than 50. The reason is the non-linear nature of an inverse function. If the algorithm needs to choose between two individuals, containing 1 and 2 selected features. The advantage of choosing the smaller subset is $\Delta fitness = 0.175 - 0.0875 = 0.0875$. If it needs to choose between the subsets of 50 and 80 features, the advantage is much smaller, $\Delta fitness = 0.0035 - 0.00219 = 0.00131$.

The solution is to change the fitness function in order to get a linear dependence to number of selected features. This fitness function is calculated as:

$$Fitness = a * P + b * \frac{T - R}{T} \tag{6.2}$$

P - classification accuracy
R - number of features in subset
T - total number of features
a, b - coefficients

This fitness function is tested with the coefficient value $b = 0.15$. It managed to get smaller feature subsets while keeping the classification accuracy maximal in all tests, as shown in graph 6.5.

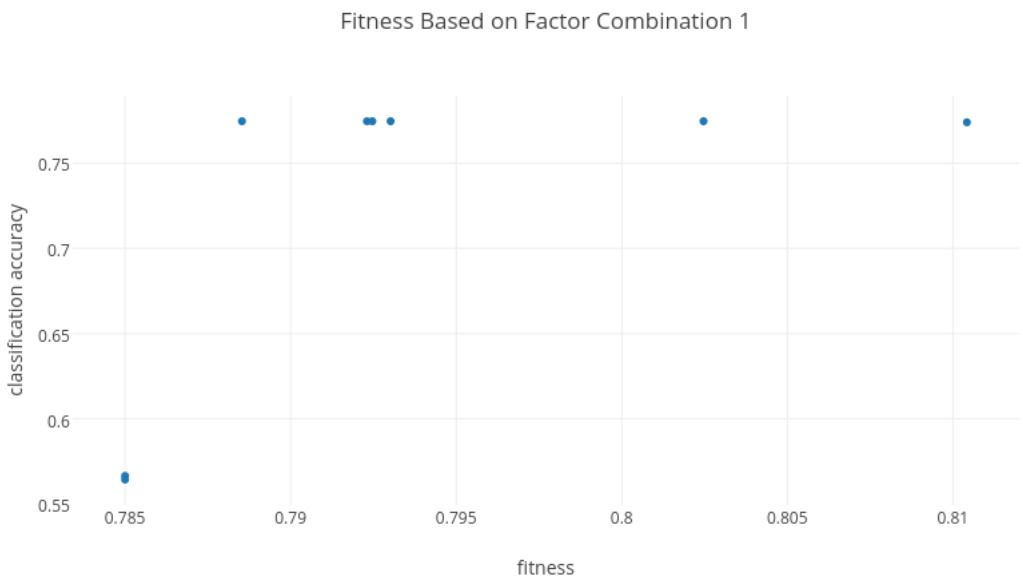

Figure 6.4: Fitness Based on Factor Combination - Variant 1

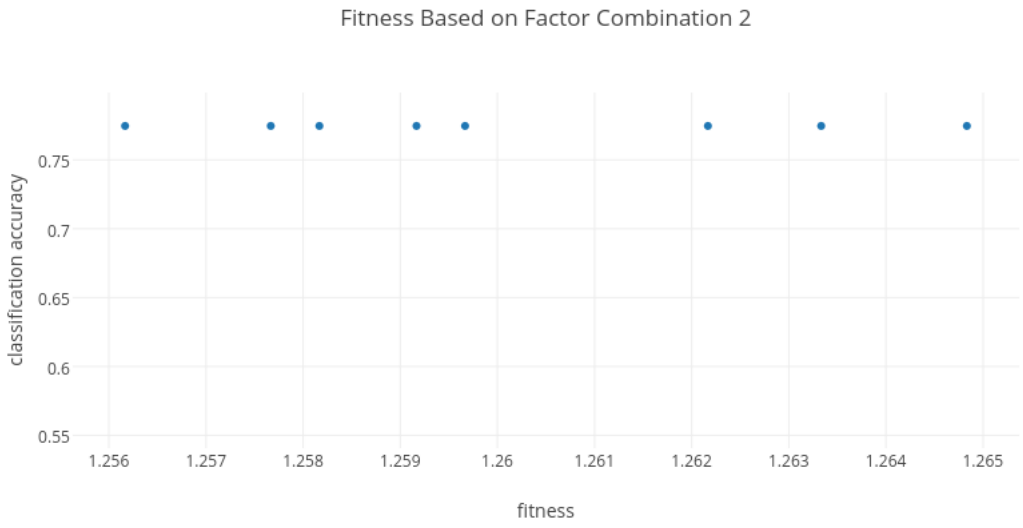

Figure 6.5: Fitness Based on Factor Combination - Variant 2

In this section, only two examples of a fitness function based on factor combination are tested. The possibility of adjusting coefficients and adding new factors opens endless possibilities in deriving new fitness functions. The next section covers this area by using genetic programming for deriving new fitness functions.

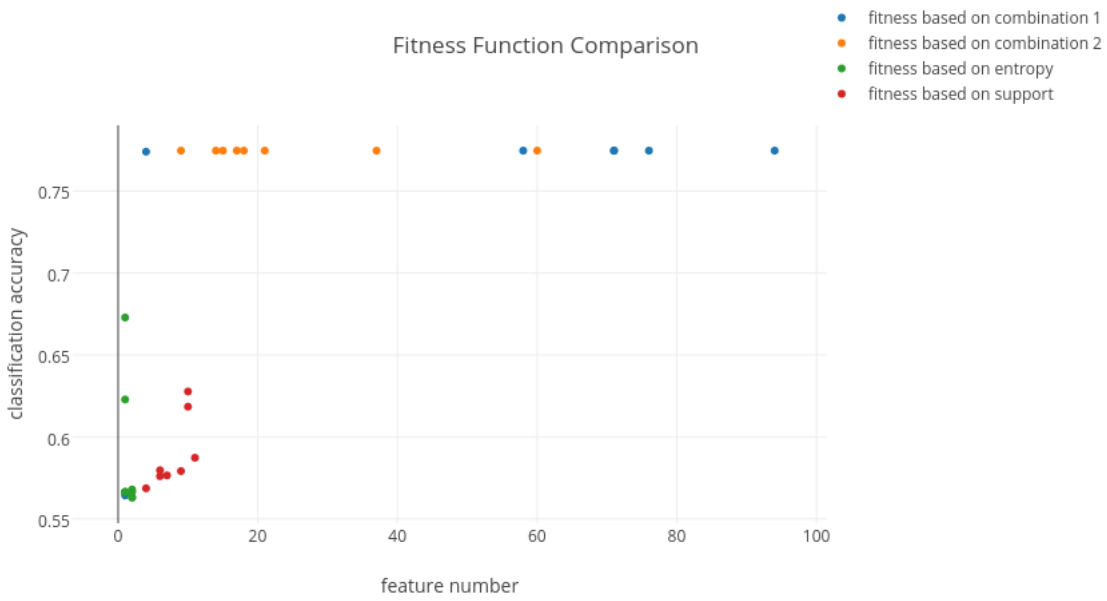

Figure 6.6: Fitness Function Comparison

In figure 6.6 the comparison between fitness measures presented earlier in this section, is provided. It shows the advantage of using classifier output in GA's fitness

function. Almost all examples in which classification accuracy is used, reached the maximum classification score. The fact that classifier used smaller dataset for fitness calculation doesn't affect the quality of solutions. Dataset used for fitness calculation consists of 1000 samples that are separated in two sets before each evaluation. One set is used for training and the other for testing.

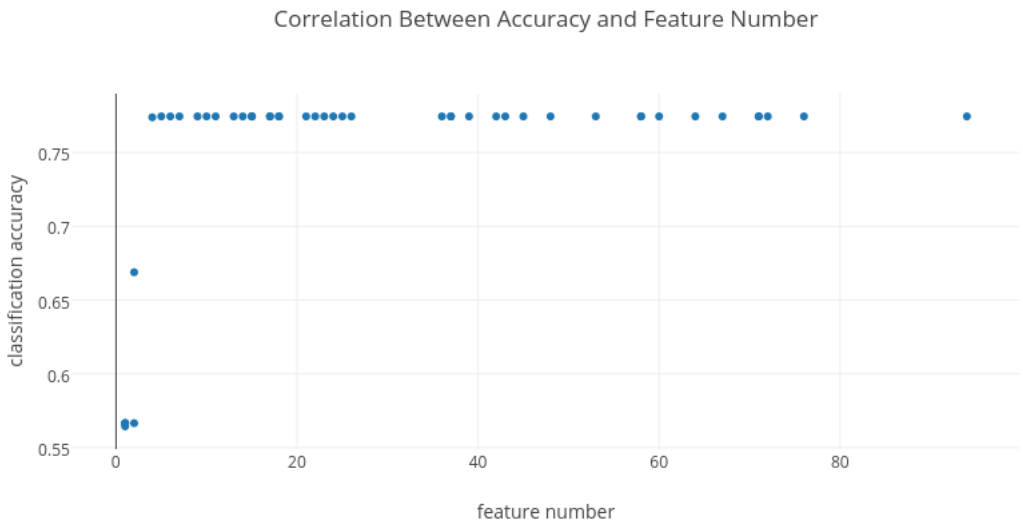

Figure 6.7: Correlation Between Accuracy and Feature Number

One of the most important questions in this thesis is how much can a number of selected features be decreased without losing information about the risk of rheumatoid arthritis. The graph in figure 6.7 shows the dependency between classification score and number of selected features. The graph consists of data from individuals evolved by GA with fitness based on factor combination. Various values of coefficients are used to get data with such broad range of subset sizes. The graph shows that even some small feature subsets manage to get good classification results. The individuals with 4 or more selected features have good classification scores.

6.3. Fitness Function Adjusting

Genetic programming (GP) is used for fitness function generation. It's given the lists of operations and terminals from which it creates fitness functions presented as a tree-structure. In the process of creating a fitness function a node type is selected with equal chances of being an operation, terminal or a constant. A type of an operation

or a terminal is also set randomly. Fitness functions undergo a process of selection, crossover and mutation in order to derive better solutions.

GP is tested with the population of 30 individuals that have the maximal tree depth equal to 4. Experiments presented in this chapter use a small set of terminals containing classification accuracy and a terminal based on the number of selected features. This terminal's value is calculated as $f(R) = \frac{T-R}{T}$ where R represents the number of selected features and T is the total number of features. Entropy and support aren't included because they weren't able to achieve good results so far and increase the training time significantly. Including them in the fitness function could be the subject of further experiments.

The GA used for fitness function evaluation has the population of 30 individuals with the maximal number of generations set to 30.

The table 6.4 shows the results from 8 GP runs. Fitness , classification accuracy, number of selected features and the fitness function of the best GP individual in the run are presented. The classification accuracies and the number of selected features are obtained from an evaluation on a smaller dataset of 1000 individuals separated in two sets. 70% of the dataset is used for training and 30% for testing.

run No.	fitness	feature number	accuracy	fitness function
1	0.901667	10	0.766667	$fitness = P * (f(R)/f(R) + 0.966988 * f(R))$
2	0.901667	10	0.766667	$fitness = 0.603703 * P * 0.975922 * 0.890996 * f(R)$
3	0.882500	5	0.740000	$fitness = f(R) * P$
4	0.906167	7	0.766667	$fitness = (0.092900 * (P + f(R))^2$
5	0.904667	8	0.766667	$fitness = 0.487398 * 0.073442 * (f(R) + P)$
6	0.901667	10	0.766667	$fitness = P^2 + 0.754306 * f(R) + f(R)$
7	0.903167	9	0.766667	$fitness = f(R) * 0.592666 * P$
8	0.897167	13	0.766667	$fitness = f(R) + P$

Table 6.4: GP Evaluation

$f(R) = \frac{T-R}{T}$
P - classification accuracy
R - number of features in subset
T - total number of features

An interesting conclusion can be drawn based on the data from this table. It's evident that many fitness functions include a common form: $P * f(R)$ which hasn't been

considered in this thesis. Further experiments should focus on training a GA with fitness function based on this expression. Also, more experiments using different sets of operations, methods for individual creation and fitness functions should be conducted.

7. Conclusion

Research of genetic markers that influence complex diseases has a great role in disease prevention and the discovery of new medication. A lot of effort has been made in this field, but there is still the need for new research that will confirm current results or find markers missed in previous attempts.

This thesis explores the use of artificial intelligence methods in the discovery of genetic markers linked to the risk of developing rheumatoid arthritis. The solution uses mutual information (MI) preprocessing, genetic algorithm (GA), genetic programming (GP) and support vector machine (SVM) for predicting whether a person is affected with rheumatoid arthritis or not.

The genetic data collected from both affected and non-affected individuals is preprocessed using MI leaving only the 100 most significant markers in the dataset. The reduced dataset then goes through another phase of feature selection where the final set of genetic factors is selected using GA. The selected factors are believed to be linked to the disease. An SVM classifier is then used to predict whether an individual has the risk of developing rheumatoid arthritis or not. The main obstacle in feature selection is deriving a good fitness function. For that reason genetic programming (GP) is introduced to derive fitness functions later used in GA.

This solution achieved good results finding feature subsets that have classification accuracies up to 77.46%. The number of genetic markers used in classifications is drastically reduced. One of the solutions has a feature set containing only 4 genetic factors and achieves the classification accuracy of 77.40%.

BIBLIOGRAPHY

[1] Making SNPs make sense. http://learn.genetics.utah.edu/content/pharma/snips/, . Accessed: 2015-06-08.

[2] Genetic linkage. http://learn.genetics.utah.edu/content/pigeons/geneticlinkage/, . Accessed: 2015-06-08.

[3] Maher B. Personal genomes: The case of the missing heritability. *Nature*, 456 (7218):18–21, 2008.

[4] Ban. Identification of type 2 diabetes-associated combination of snps using support vector machine. *BMC Genetics*, stranica 11:26, 2010.

[5] Bruce Carlson. SNPs—A shortcut to personalized medicine. http://www.genengnews.com/gen-articles/snps-a-shortcut-to-personalized-medicine/2507/, 2008. Accessed: 2015-06-08.

[6] The Wellcome Trust Case Control Consortium. Genome-wide association study of 14,000 cases of seven common diseases and 3,000 shared controls. *Nature*, 447:661–678, 2007.

[7] Condé de Oliveira F., Hasenclever Borges C.C., Nascimento Almeida F., Fonseca e Silva F., da Silva Verneque R., Vinicius GB da Silva M., and Arbex W. SNPs selection using support vector regression and genetic algorithms in gwas. *BMC Genomics*, 15(Suppl 7), 2014.

[8] Paul Godden. Machine learning for genome-wide association studies: A critical review, 2013.

[9] Goldstein. An application of random forests to a genome-wide association dataset: Methodological considerations & new findings. *BMC Genetics*, stranica 11:49, 2010.

[10] L. Jourdan, C. Dhaenens-Flipo, and E. Talbi. *Evolutionary Computation in Bioinformatics*, poglavlje Discovery of genetic and environmental interactions in disease data using evolutionary computation, stranice 297–316. Morgan Kaufmann Publishers, 2003.

[11] Ikram MK., Sim X., and Xueling S. Four novel loci (19q13, 6q24, 12q24, and 5q14) influence the microcirculation in vivo. *PLoS Genetics*, 6(10), 2010.

[12] MA. Mooney, B. Wilmot, Bipolar Genome Study, and SK. McWeeney. The GA and the GWAS: Using genetic algorithms to search for multi-locus associations. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, IEEE, ACM, 10.1109/TCBB.2011.145., 2011.

[13] Waddel N. Predicting cancer susceptibility from single-nucleotide polymorphism data: a case study in multiple myeloma. *BIOKDD '05 Proceedings of the 5th international workshop on Bioinformatics*, stranice 21–28, 2005.

[14] Uhmn S., Kim D-H., Ko Y-W., Cho S., Cheong J., and Kim J. A study on application of single nucleotide polymorphism and machine learning techniques to diagnosis of chronic hepatitis. *Expert Systems*, 26:60–69, 2009.

Detecting Complex Diseases Using Genetic Features

Abstract

This thesis explores the problem of complex disease detection using genetic data. The proposed solution is based on feature selection using genetic algorithm and disease detection using machine learning algorithms. A method for deriving an optimal goal function based on genetic programming is introduced and examined. The proposed solution is tested on a dataset containing data collected from rheumatoid arthritis patients. The results are presented and discussed.

Keywords: disease detection, genetic algorithm, machine learning, SVM, genetic programming

Detekcija oboljenja na temelju genetskih značajki

Sažetak

U ovom radu istažen je problem detekcije složenih bolesti na temelju genetskih značajki. Predloženo rješnje uključuje odabir značajki uz pomoć genetskog algoritma te detekciju bolesti algoritmima strojnog učenja. Predložen je postupak pronalaženja funkcije cilja za odabir značajki korištenjem genetskog programiranja. Rješenje je ispitano na skupu podataka prikupljenom od pacijenata s reumatoidnim artritisom. Rezultati ispitivanja i zaključi priloženi su radu.

Ključne riječi: detekcija bolesti, genetski algoritam, strojno učenje, SVM, genetsko programiranje