Feature Relevance Estimation by Evolving Probabilistic Dependency Networks with Weighted Kernel Machines

A thesis submitted to the
District University Francisco José de Caldas
in fulfilment of the requirements for the degree of
Master of Science in Information and Communications

By

Nestor Andres Rodriguez Gamboa

Advisor

Sergio A. Rojas, PhD.

November 2013

Contents

1. Introduction **1**

1.1. Aim 1

1.2. Problem and Motivation 1

1.3. Contributions 6

1.4. Literature Review 7

1.4.1. Feature Selection Techniques 7

1.4.2. Machine Learning 10

1.4.3. Evolutionary Estimation of Probabilistic Distributions . . . 12

1.4.4. Feature Selection Techniques Using Estimation of Distribution Algorithms 17

2. Proposal **19**

2.1. Why `TILDA` and `Goldenberry` ? 22

3. `TILDA` **23**

4. `Goldenberry` **25**

5. `wKieraII` **26**

6. Conclusion **27**

List of Figures

1.1. Correlation analysis of input variables in relation to overweight conditions of insurance customers. 2
1.2. Using relevant features customers can be easily segmented with a simple linear function. 3
1.3. A feature selection taxonomy. 9
1.4. Transformation from input space to a feature space simplifies the classification task. 11
1.5. Univariate Estimation of Distribution Algorithm flowchart. 14
1.6. Diagram of probability models used in most popular bivariate EDAs. 16
1.7. Diagram of probability models used in most popular multivariate EDAs. 17
1.8. Main components of the FSS-EBNA algorithm. 18
1.9. Main components of the wKIERA algorithm. 18

2.1. The components involved in `wKieraII`. 20
2.2. `wKieraII` flowchart. 21

List of Tables

Acknowledgements

Abstract

This thesis focuses on the problem of estimating relevance of observed variables for a classification task in high dimensional spaces, which is known as feature subset selection or feature relevance determination by the machine learning community. The main goal of the thesis was to design of a novel feature relevance estimation method by combining techniques for estimation of dependency networks with weighted kernel machines[1]. The method, called `wKieraII`, works within a population-based stochastic search framework where relevant (but unknown) variables from a given dataset[2] are found by iteratively evolving a set of relevance estimation candidates. These candidates will consist of parameters of bivariate conditional probability distributions. The distributions could be seen as dependency networks of how variables influence each other. With this information a subset of the most relevant features from the original sample can be selected to perform classification, the suitability of the subset being assessed as its predictive classification accuracy. Weighted kernel classifiers are used for this purpose. The method provides additional information about dependency among such variables as a byproduct of the selection process. The thesis also contributes to the development of `TILDA`, a novel estimation of distribution algorithm, and `Goldenberry`, a supplementary machine learning and evolutionary computation suite for `Orange`.

[1] The topics of feature selection methods, estimation of probability distribution algorithms and kernel machines are briefly reviewed in section 1.4. The interested reader is referred to [?, ?, ?] for details.

[2] In the context of this paper *dataset* means data arranged in a tabular way where columns represent variables and rows represents instances. Other types of formats (such as text, images, graphs, etc.) are not considered.

Resumen

Aca se encontrara el resumen en espaol del Abstract.

Chapter 1

Introduction

1.1. Aim

The main aim of the thesis was to design an algorithm for feature relevance estimation that discovers interactions among variables. We aimed to build upon `wKiera` method a newer version that incorporates a technique for bivariate feature selection within an population-based stochastic algorithm.

1.2. Problem and Motivation

Feature subset selection (FSS) is a machine learning technique for dimensionality reduction in datasets where irrelevant or noisy variables may appear. It is useful in problems where few from a big set of observed variables explain a given pattern in classification, prediction or clustering of data cases (e.g. diagnosis of a disease, market analysis, image segmentation, etc.), and therefore it allows for an expert to focus on data that is really important, reducing the expenditure of

costly or time-consuming experiments.

To illustrate the problem, consider the situation where an insurance company is evaluating the factors that have an impact on people's overweight conditions that lead to claims in illness insurance policies. Upon registration, they collect information of both parents' weight and birthday. They want to know which of these variables are related to the development of the condition. Now let us suppose that one-year time is required by an insurance analyst to process the information of just one variable. Then it would be desirable to discard irrelevant variables in order to save him valuable time. An FSS method could help to select the relevant ones.

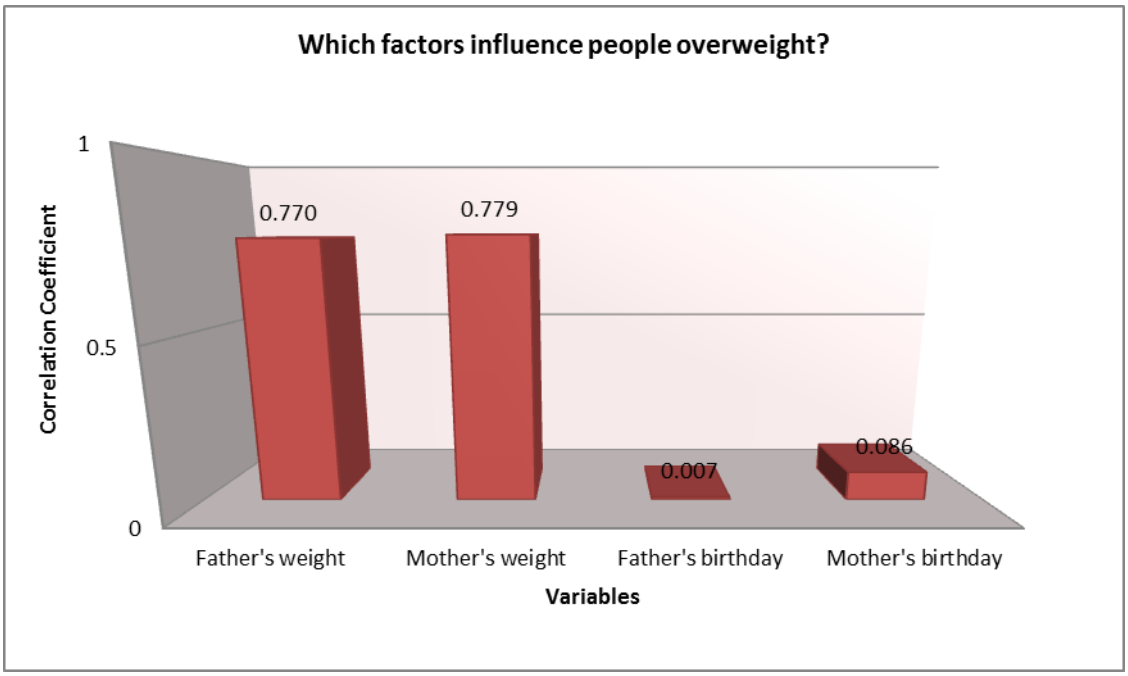

Figure 1.1: Correlation analysis of input variables in relation to overweight conditions of insurance customers (for the fictional example in the text).

Figure 1.1 depicts the result of applying a correlation-based feature selection method on this fictional problem; it is evident that half of the variables, namely

those related to birth dates, are irrelevant and can be ignored for the problem of interest. Thus, dimensionality reduction may provide insights in data mining tasks, yielding a better understanding of objects or phenomena behavior and is useful to speed up data analysis and to improve generalization performance. For instance, in the previous insurance company example, Figure 1.2 shows how sample cases can be now rendered in the 2D space obtained with the relevant variables; they can be easily discriminated with a simple hyperplane. By focusing on those two relevant variables, the analyst would have saved two precious years of working time and the company two equally valuable years of salary payments.

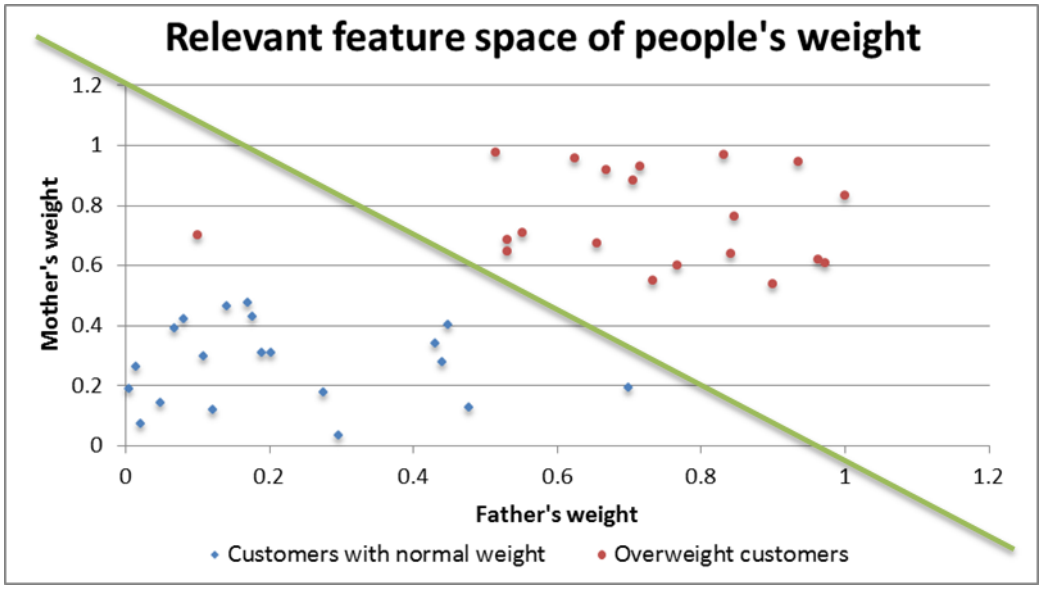

Figure 1.2: Using relevant features customers can be easily segmented with a simple linear function.

Some FSS techniques assume variables are independent. This assumption might not be appropriate for some real world problem since variables may influence others and these interactions are usually hidden. A good example is in the field of bioinformatics where experts analyze proteins relevance in diseases to de-

sign inhibitory vaccines; using state-of-the-art high-throughput technologies (e.g. mass-spectometry [**?**]) they are able to collect large amounts of data about activation of protein mechanisms, but the information about how these interactions occur is unknown or difficult to obtain, not to mention that just a few proteins from a large proteomic spectrum would be related to the activation of the disease.

The independent assumptions (univariate FSS methods, see section **??**) are very popular because of their low computational cost [**?**] but they do not provide additional insights about data relationships. On the other hand, missing information about those dependencies may affect negatively the prediction accuracy of the feature subset. As an alternative approach, multivariate FSS methods search for feature subsets and possible dependency relationships between them at the cost of an increase in computational complexity (recall that FSS is a combinatorial problem known to be NP-Hard [**?**, **?**]). The challenge is then to design novel feature selection methods that take advantage of multivariate power combined with high-accuracy classifiers to obtain improved prediction and explanatory performance.

On the other hand, kernel classifiers have recently emerged as powerful high-accuracy classifiers with robust generalization abilities [**?**]. They use linear functions to perform classification, allowing for non-linear discriminatory borderlines in the input space by means of a kernel function. This function is a mapping of similarity measures in a transformed space where nonlinearities can be solved linearly. Two widely used kernel functions are the RBF kernel and the polynomial kernel (Eq.(1.1) and Eq.(1.2) respectively):

$$K_\sigma(\bar{x}, \bar{z}) = \exp\Big(- \sigma \sum_k (x_k - z_k)^2 \Big) \tag{1.1}$$

and

$$K_d(\bar{x}, \bar{z}) = \langle \bar{x}, \bar{z} \rangle^d, \tag{1.2}$$

where $\bar{x}, \bar{z} \in X$, $X \subseteq \mathbb{R}^D$ represents the input space and D the number of variables or dimensions. The parameter σ and d define the width of the RBF and polynomial kernel respectively. Modified weighted versions of these kernels introduce scale factors $\bar{w} = \{w_1 ... w_\ell\}$ to weight the amount that each dimension contributes to the total computation [**?**]. The weighted RBF and weighted polynomial kernels are then defined as Eq.(1.3) and (1.4):

$$K_\sigma(\bar{x}, \bar{z}) = \exp\Big(- \sigma \sum_k^\ell w_k (x_k - z_k)^2 \Big) \tag{1.3}$$

and

$$K_d(\bar{x}, \bar{z}) = \Big(\sum_k^\ell w_k (x_k \cdot z_k) \Big)^d. \tag{1.4}$$

Some FSS methods have been proposed in connection with kernel functions, for instance the weighted kernels in Eq. (1.3) and (1.4) have been used to define a FSS embedded method (the weigthed kernel iterative estimation of relevance algorithm, wKIERA [**?**]). The idea in this case is to consider the weight vector $\bar{w}$ of the kernel as a relevancy estimator of input variables. The vector is tuned by a univariate estimation of distribution algorithm [**?**] which iteratively refines the distribution by a population-based technique inspired in genetic algorithms (GA) [**?**]. The accuracy of the kernel classifier coupled with the weight vector candidates is taken as the fitness measure of the GA. The authors showed promising results of this algorithm in selecting relevant variables in a number of different classification

tasks, including problems with linear and non-linear hypothesis targets. This FSS method however, is designed on the basis of the independent assumption mentioned above and consequently does not provide the power of explanation of bivariate methods and does not account for information of relationship between the variables. One of the motivations of this thesis was to build upon this method and design a refined version that incorporates techniques for bivariate feature selection within an embedded population-based stochastic algorithm. A Hierarchical Tree was chosen to represent how variables depend and influence each other and a new algorithms, called `wKieraII`, was develop to perform relevance estimation within such framework and to make feasible the approach.

1.3. Contributions

The primary contributions of this thesis are:

- `TILDA` [**?**], a novel continuous *compact* Estimation of Distribution Algorithm (`EDA`) built upon the compact Genetic Algorithm (`cGA`) and the continuous domain Population-Based Incremental Learning algorithm.

- `Goldenberry` [**?**], a supplementary machine learning and evolutionary computation suite of components for `Orange`. This suite provides components in the form of `EDA`(`cGA` [**?**], `UMDA` [**?**], `PBIL` [**?**], `TILDA` [**?**], $\texttt{PBIL}_c$ [**?**], $\texttt{UMDA}_c$ [**?**], `BMDA` [**?**]), Machine Learning (`Kernel Perceptron`, `SVM` adaptation), Feature Selection (`BivariateFeatureSubset`, `WkieraCostFunction`), and a set of utility components (`CostFunctionBuilder` , `KernelBuilder`, `BlackBoxTester`).

- An enhanced version of `BMDA`, able to uses different kind of dependence scor-

ing functions like Mutual Information, Pearson's chi-squared test and the Combined Mutual-Information p-value (SIM [**?**]) method.

- `wKieraII`, an new feature relevance estimation algorithm build upon its previous version `wKiera`. This new version considers the existence bivariate relationships among variables. It uses the `BMDA` for estimating bivariate dependencies and features relevances, and the classification accuracy from a learning algorithm (`Kernel Perceptron` or a `SVM`) as the cost function to assess the goodness of the search space.

1.4. Literature Review

In this section we provide a summarized review of the main elements involved in this thesis, namely feature selection techniques, kernel classification machines, and estimation of distribution algorithms.

1.4.1. Feature Selection Techniques

The problem of feature selection

Nowadays the advances in technologies for data collection (the genome project, particle colliders, internet-based social networks) pose increased challenges for data analysis due to larger sizes and higher dimensionality of the observed samples. It is reasonable however to assume that the collected variables may exhibit redundancy, inconsistency, noisy and irrelevant data and therefore will be difficult to analyze by the human eye. New automated mechanisms are required to identify significant variables for pattern discovery and data mining.

Feature selection is gaining in importance and has recently become an active field of research in disciplines such as knowledge discovery, machine learning, pattern recognition, bioinformatics, geoinformatics, etc. While FSS are techniques for dimensionality reduction, they maintain the original variables compared to other entropy-based, data compression or statistical methods that modify the original data representation instead of providing a subset of significant features with useful information for the domain experts [**?**]. The main goal of these techniques is to obtain a better understanding of the data for visualization purposes or to speed up data analysis.

FSS techniques are helpful in improving prediction models for supervised learning, detecting dimensions of tight data conglomeration in clustering tasks and a deeper understanding of process for data generation. Nonetheless, the design of novel FSS techniques aimed to provide more useful information also incurs in new levels of complexity giving rise to new challenges pertaining to feasible and efficient computational techniques.

In the classification context, feature selection techniques are categorized in three main groups depending on how they interact with a classification model (see Figure 1.3): filters, wrappers and embedded. A detailed description of this categorization is given below (based on the study in [**?**]).

Filter Approach

Methods in this category assess the relevance of variables based on the intrinsic properties of data. The search in feature space is separated from the search in the space of classification hypothesis. In most cases a feature relevance score is calculated (i.e. using mutual-information or data correlation) and low-scoring

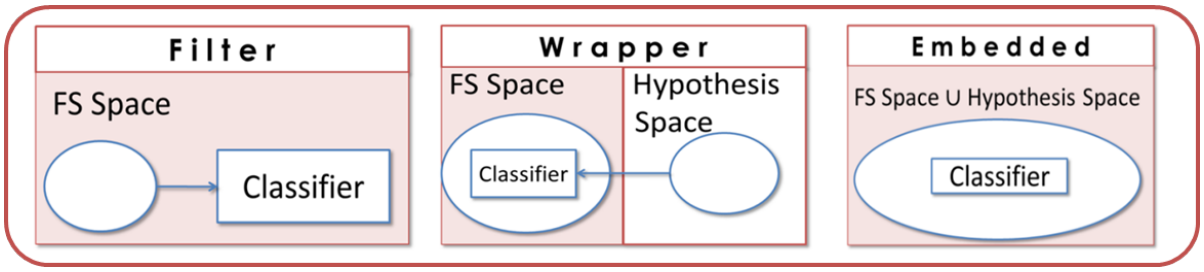

Figure 1.3: A feature selection taxonomy. The feature selector category depends on how the feature and hypothesis space are combined during classifier learning (left and right box taken from [**?**]).

features are removed. These methods have the ability to scale to higher dimensions since they exhibit a low computational cost.

Wrapper Approach

Wrappers establish a symbiotic relationship between the feature subset search and the classification algorithm. The aim of these type of methods is not only to find relevant features but also to find the best suited classification model to those features. However the interaction with the classifier may induce new problems such as a higher risk of overfitting and a higher demand of computational resources.

Embedded Approach

Embedded methods incorporate the feature selection and the classification model construction as a single process. As a result feature and hypothesis spaces are combined providing a richer source of information during the search although incurring as well in additional computational costs.

1.4.2. Machine Learning

Supervised learning aims to find optimal learning algorithms with high generalization and prediction performance to produce classification models of the training set of examples and their associated labels. The prediction accuracy of the resulting classifier is then evaluated with a test set. Among these techniques, linear classifiers are widely used because of their theoretically simplicity and computational efficiency. Application of linear classifiers, in real world problems where nonlinearities arise, has been possible due to the advances in kernel machines [**?**].

Kernel Machines

These algorithms use kernel functions to compute similarity measures of the input samples in a feature space where nonlinearities might be easier to solve by linear classifiers. Figure 1.4 shows an example of feature mapping where in the original space data can only be separated with a nonlinear function but becomes linearly separable if the data is transformed into a feature space. When using the linear classifier the feature space mapping is implicit since the machine only uses inner products of the transformed examples which are computed using kernel function.

A kernel is a function K such that for all $\bar{x}, \bar{z} \in X$

$$K(\bar{x}, \bar{z}) = \langle \phi(\bar{x}), \phi(\bar{z}) \rangle,$$

where ϕ is a mapping from the input space X to an (inner product) feature space. In order to be a kernel, the function must comply with the conditions defined by Mercer's theorem [**?**]. Eq. (1-4) are examples of kernel functions.

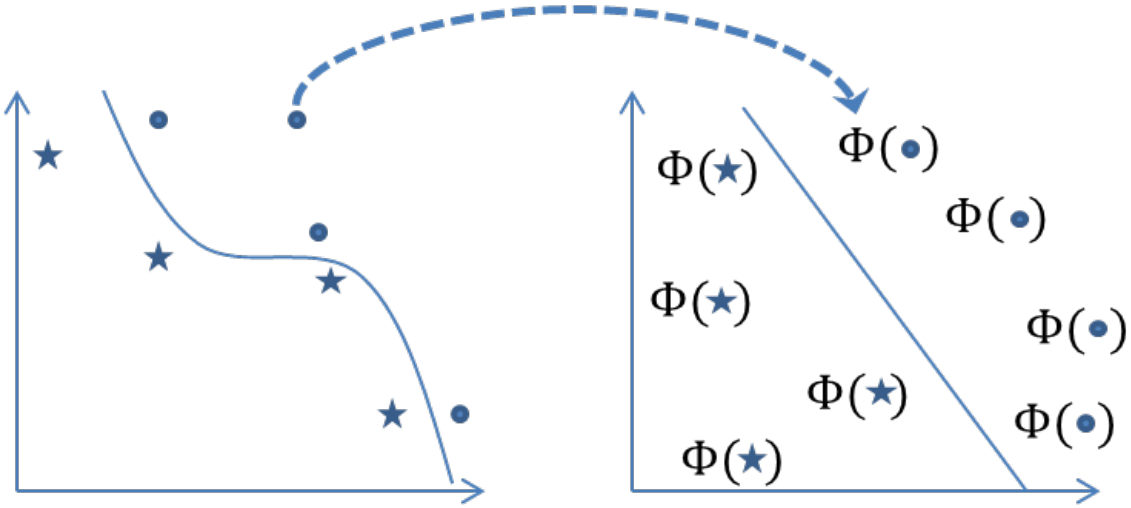

Figure 1.4: Transformation from input space to a feature space simplifies the classification task (taken from [**?**]).

Linear Learning Machines

A linear classifier is a linear function $f(\bar{x})$ where $\bar{x} \in X$ and $\bar{w} \in \mathbb{R}^{\mathcal{D}}$, that can be written as:

$$f(\bar{x}) = \text{sgn}(\langle \bar{w}, \bar{x} \rangle) = \text{sgn}\Big(\sum_i w_i x_i \Big) \tag{1.5}$$

A given sample $\bar{x}$ is assigned to the positive class if $f(\bar{x}) = 0$ or otherwise to the negative class. There are number of training algorithms to learn a classification vector $\bar{w}$, including the perceptron [**?**] and the linear support vector machine [**?**]. For illustration purposes, Algorithm 1 shows the learning procedure of the perceptron.

The kernelized version of the linear classifier allows for classification of non-linearly separable datasets, as mentioned before. The classification function is written consequently as:

$$f(\bar{x}) = \mathrm{sgn}\Big(\sum_k \alpha_k K(\bar{x_k}, \bar{x})\Big), \tag{1.6}$$

where K represents a kernel function and α_k the classifier parameters that can be learned in this case using the kernel perceptron [**?**] or the support vector machine [**?**, **?**].

Algorithm 1 The Perceptron

Requires: Given a linearly separable training set S and learning rate $\eta \in \mathbb{R}^+$

```
w ← 0; k ← 0
repeat
  repeat  i = 1, 2, ... ℓ
    if y_i(⟨w_k, x_i⟩) ≤ 0 then
      w_{k+1} ← w_k + η y_i x_i
      k ← k + 1
    end if
  end repeat
until no mistake made within the for loop
```

Outputs: (w_k) where k is the number of mistakes

1.4.3. Evolutionary Estimation of Probabilistic Distributions

Genetic algorithms (GA) are search stochastic methods inspired in the theory of natural selection of Darwin. The idea is to evolve a population of candidates coding the parameters for the solution of an optimization problem, using genetic operations such as chromosome recombination and mutation [**?**]. A novel technique known as Estimation of Distribution Algorithms (EDAs) has recently emerged motivated by GAs but from a statistical viewpoint. They have proven to be better suited in many applications than canonical GAs [**?**].

The main distinctive aspect of EDAs is that they search for a probabilistic distribution model representing the population of candidates. Instead of using genetic operations, these algorithms are based on well-known statistical techniques to estimate the parameters of a distribution function, and the evolution is guided by sampling the evolving probabilistic model. The complexity of the algorithm lies in the robustness of this probability model and in how it is iteratively re-estimated. The probabilistic models can be as simple as a marginal distribution or as complex as a joint multivariate distribution expressing high order interactions among the observed variables. These categories are briefly described below following the review in [**?**].

Univariate EDAs

Univariate algorithms use a marginal probability model encoded in a probability vector. They are not computational intensive because they assume no interaction between variables. The probability vector is re-estimated using the fittest subset of the population of candidates, as depicted in Figure 1.5. There are three widely used algorithms for this category: Univariate Marginal Distribution Algorithm (UMDA [**?**]), Popupation-based Incremental Learning (PBIL [**?**]) and the Compact Genetic Algorithm (cGA [**?**, **?**]).

UMDA estimates the entire probability vector every iteration. The probability distribution is factorized as a product of independent univariate marginal distributions, which are estimated from marginal frequencies. The probability of the i-th variable being relevant is given by the Eq.(1.7), where S is the subpopulation of N fittest candidates, and δ is the Kronecker delta.

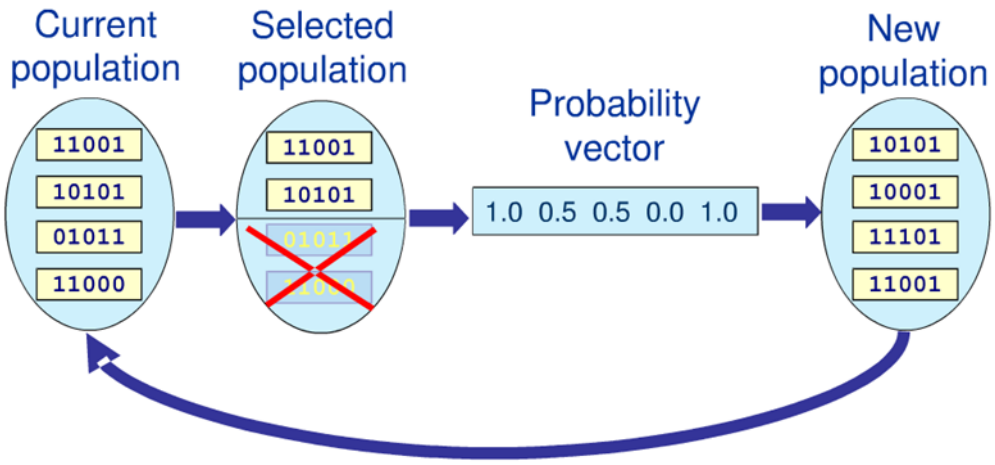

Figure 1.5: Univariate Estimation of Distribution Algorithm flowchart.

$$p(x_i) = \prod_j \frac{\sum_j^N \delta(X_i = x_i|S)}{N} \tag{1.7}$$

PBIL was designed to work in a n-dimensional binary space $\{0,1\}^n$. Unlike UMDA, PBIL does not estimate a new probability vector every iteration t. Instead, fittest candidates are chosen to update the probability vector at a learning rate $\alpha = (0,1]$. The updating rule is shown in Eq.(1.8). Notice that PBIL becomes UMDA when $\alpha = 1$.

$$p_t(x_i) = (1-\alpha)p_{t-1}(x_i) + \alpha\frac{1}{N}\sum_k^N S_{ki} \tag{1.8}$$

Lastly, cGA has become popular for the higher efficiency to solve very large scale problems with millions to billions of variables with a lower computational demand than canonic GA [**?**]. cGA has a low memory consumption because only two candidates per iteration are generated. Both compete and the winner updates the probability vector at a learning rate $1/n$ where n is the population size parameter.

Algorithm 2 shows the pseudocode of the cGA .

Algorithm 2 The Compact Genetic Algorithm

Requires: Population size n and chromosome lenght ℓ
  $p \leftarrow$ initialize a uniform probability vector.
  repeat
    $[a, b] \leftarrow$ generateTwoIndividuals(p)
    $[winner, loser] \leftarrow$compete($a, b$)
    repeat $i = 1, 2, \ldots \ell$
      if $winner[i] \neq loser[i]$ **then**
        if $winner[i] = 1$ **then**
          $p[i] \leftarrow p[i] + \frac{1}{n}$
        else
          $p[i] \leftarrow p[i] - \frac{1}{n}$
        end if
      end if
    end repeat
  until p has converged
Outputs: p represents the final solution

Bivariate EDAs

Bivariate EDAs use joint statistics models of 2-order-dependencies to represent interaction among variables. Most representative algorithms are: the Mutual-Information Maximizing Input Clustering (MIMIC [**?**]), the Combining Optimizers with Mutual Information Trees algorithm (COMMIT [**?**]) and the Bivariate Marginal Distribution Algorithm (BMDA [**?**]). All of them focus their attention in finding hidden pair-wise interactions among variables assuming different interaction's forms as depicted in Figure 1.6. MIMIC was one of the first attempts for a bivariate EDA . MIMIC assumes pair-wise interactions in a chain-like form where each variable is conditioned to the previous one. MIMIC iteratively estimates marginal and conditional probabilities using the promising candidates and

through a greedy algorithms tries to find a optimal chain of variables that better minimizes the cross-entropy. COMMIT behaves similar to MIMIC but instead of using a chain-like model, it tries to find a suitable tree representation of variables. The tree is estimated on each iteration using the promising candidates through a maximum spanning tree algorithm and a graph where variables represent nodes and edges the Mutual Information between nodes. Therefore COMMIT assumes all variables as been connected; whereas BMDA, on the other hand, behaves similar to COMMIT but it assumes that some variables may be independent. BMDA uses a maximum spanning forest algorithm for inferring dependencies from the graph of variables and the *Pearson's chi-squared test* to determine whether edges exists between nodes.

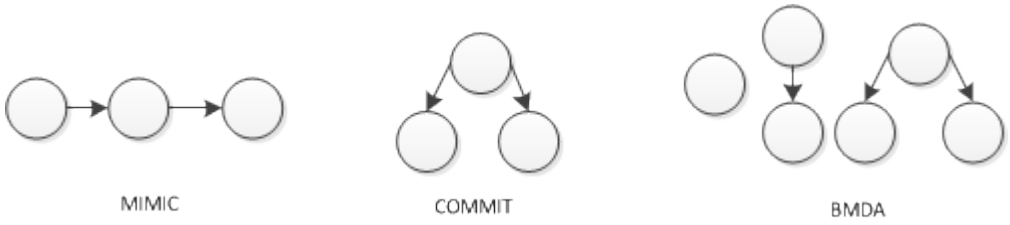

Figure 1.6: Diagram of probability models used in most popular bivariate EDAs.

Multivariate EDAs

Multivariate EDAs use joint statistics models of higher order to represent interaction among variables. These models are usually represented as probabilistic graphical models (see Figure 1.7). As mentioned above, EDA's complexity increases when a higher order of interaction among variables are desired. Factorized Distribution Algorithm (FDA [**?**]), Estimation of Bayesian Networks Algorithm (EBNA [**?**]) and Bayesian optimization algorithm (BOA [**?**]) belong to this cate-

gory. BOA and EBNA both use Bayesian network structures but differs in the score metric they use to select the appropriate network structure. BOA uses Bayesian Dirichlet equivalence score (BDe) while EBNA uses K2+Penalization and Bayesian Information Criterion (BIC). Similarly, FDA uses Boltzmann selection for Boltzman distribution.

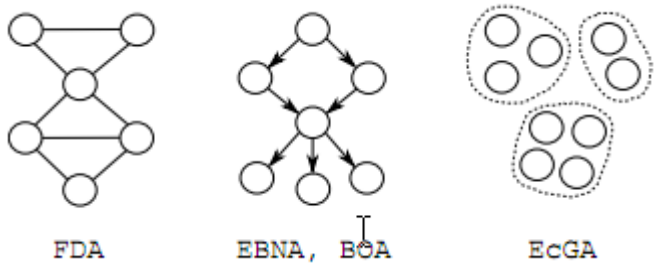

Figure 1.7: Diagram of probability models used in most popular multivariate EDAs(taken from [**?**]).

1.4.4. Feature Selection Techniques Using Estimation of Distribution Algorithms

Estimation of Distribution Algorithms for feature subset selection and feature relevance estimation has been proposed with promising results in large scale domains such as bionformatics [**?**, **?**]. Among these techniques, particularly FSS-EBNA [**?**] and wKIERA [**?**] have been used for selection of relevant feature subsets. FSS-EBNA uses a multivariate estimation of distribution algorithm (EBNA) as the search engine for exploring the feature space and a Naive-Bayes classifier (NB) to predict the class for each instance; Figure 1.8 illustrates the main components for this algorithm.

On the other hand wKIERA uses a univariate estimation of distribution algo-

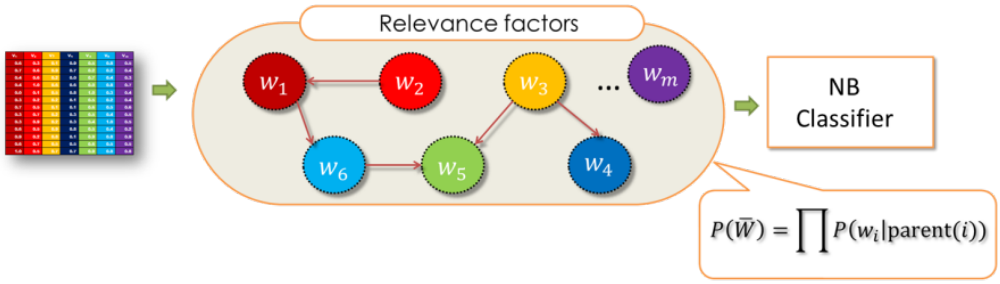

Figure 1.8: Main components of the FSS-EBNA algorithm.

rithm for searching the best suited representation of features relevance and a RBF weighted kernel machine with a perceptron classifier as the learning algorithm; Figure 1.9 illustrates the main components of this algorithm.

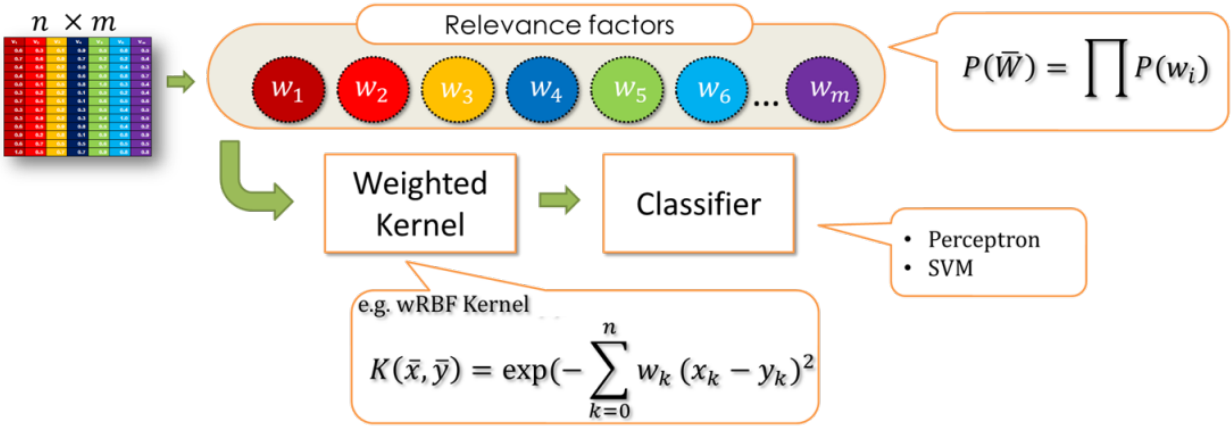

Figure 1.9: Main components of the wKIERA algorithm.

Chapter 2

Proposal

This thesis proposes a new FSS embedded method termed `wKieraII`. Its overall goal is to identify relevant feature subsets and as much as possible information about dependencies that explain significant patterns hidden in the data. This method treats the features estimation and patter discovery as an optimization problem. It uses a `EDA` to explore the search space and the goodness of each candidate is assessed by a learning technique. Each candidate represents, for the `Learner`, a relevance estimation for its feature space. The better the candidate portrays the relevant features in the feature space the better the classification accuracy and fitness score obtained.

The main components of `wKieraII` are depicted in Figure 2.1. The *data* is the sample of observed variables for a given problem; The *Bivariate Bernoulli Distribution* is the probability distribution estimated by the `EDA` algorithm whose parameters are the marginal probabilities $P(\bar{x} = 1)$ and the contingency tables of the pair-wise dependencies from the variables.

The `EDA` is a technique to infer the parameters and structure of the bivariate

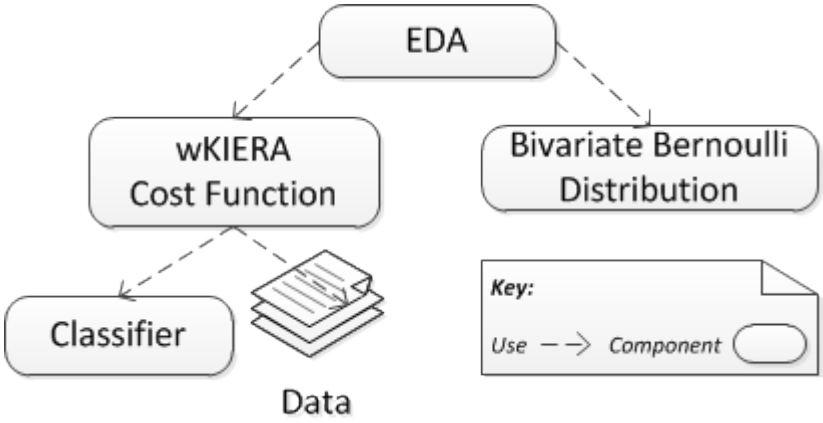

Figure 2.1: The components involved in `wKieraII`.

probability distribution by estimating its parameters iteratively from a pool of promising candidate weight vectors; the distribution and candidates are adjusted within a framework of a stochastic population-based evolutionary algorithm. The `wKiera` *Cost Function* acts as an mediator between the `EDA` and the `Learner`. It takes each candidate $\bar{w}$ from the `EDA` as a relevance estimator of the `Learner`'s feature space. If the `Learner` is a *Weighted Kernel Classifier* the weight vector $\bar{w}$ becomes the weights for the *Weighted Kernel*. Therefore the fitness of each candidate $\bar{w}$ is the classification accuracy measurement from the cross-validation.

A general flowchart of `wKieraII` is shown in Figure 2.2. There, the *Bivariate EDA* generates the candidates to explore the search space. Each candidate is assessed by the `Learner` and top ranked are selected by the `EDA`to reestimate the bivariate distribution. This flow is repeated until the probability distribution converge or a finalization criterion is met. As a result the `EDA`provides the best solution found with a weigh vector $\bar{w}$ representing the variables relevances and the dependencies as a hierarchical forest. The pseudo-code is summarized in Algorithm

3.

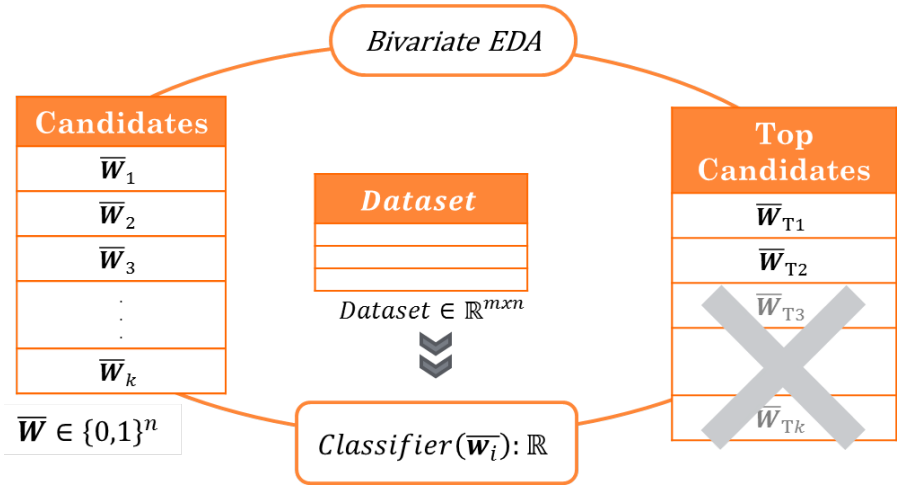

Figure 2.2: `wKieraII` flowchart.

Algorithm 3 Pseudocode of the method described in this proposal

Requires: Given a dataset $\mathcal{D}$, a weighted kernel κ_ω and a learner $\mathcal{A}$

Let β represents a bivariate bernoulli probability distribution initialized with an independent joint distribution. Let $\mathcal{B}$ represents the best solution.

repeat

$\bar{\Omega} \leftarrow$ Sample k candidates from β

repeat $\boldsymbol{\omega}_j \in \Omega$

Cross Validation: $cv_j \leftarrow \mathcal{A}(\mathcal{D}, \kappa_{\omega_j})$

Classification Accuracy: $ca_j \leftarrow \text{CA}(cv_j)$

end repeat

$\bar{\Omega}' \leftarrow \text{get\_top\_ranked}(\bar{\Omega}, \bar{ca})$

Update best solution if needed: $\mathcal{B} \leftarrow update(\mathcal{B}, \bar{\Omega}')$

Re-estimate distribution: $\beta \leftarrow \text{re\_estimate}(\bar{\Omega}')$

until β has converged or maximum number of evaluations reached

return Distribution β and best solution found $\mathcal{B}$

2.1. Why TILDA and Goldenberry ?

During the exploration and development phases of wKieraII two important contributions were achieved: TILDA and Goldenberry. On one hand, TILDA was the result of the analysis performed on EDAs algorithm. We identify cGA as a memory efficient algorithm with promising results on problems of millions of variables. However, it was only conceived to work on a discrete domain $[0, 1]$. Therefore we mixed the cGA's memory-efficient capabilities and the well-known PBIL_c continuous domain EDA in a new algorithm named *Tiny Incremental Learning Density Estimation Algorithm* (TILDA) and its arithmetic-coding version @TILDA(a detailed explanation can be found in chapter 3). On the other hand, after an exploration of machine learning frameworks and tools in order to build upon them wKieraII, we found Orange: An open-source component-based software framework, featuring visual and scripting interfaces for many machine learning algorithms. By design Orange provides clear extensibility points named *widgets and add-ons*, to integrate as plug-ins new developed algorithms. Orangedid not provided tools for optimization nor for EDAs, one of the main components needed for wKieraII. Therefore we introduced Goldenberry, an Orange add-on with visual components for stochastic search-based optimization, feature selection and machine learning. Its main purpose was to provide an user-friendly workbench for building and testing wKieraIIupon its versatile visual front-end of and the powerful reuse and glue principles of component-based software development. Architecture of the toolbox and implementation details are given in chapter 4, including description and working examples for all its components.

Chapter 3

TILDA

TILDA is novel method that is able to compactly solve regular and noisy versions of these problems with minimal memory requirements, regardless of problem or population size. This feature allows the algorithm to be run in a conventional desktop machine upon the compact Genetic Algorithm (cGA) and the continuous domain Population-Based Incremental Learning algorithm ($\texttt{PBIL}_c$) together with its arithmetic-coding version. TILDA iteratively by means of 2-way tournaments (like cGA) estimates a Gaussian distribution with the fittest candidate whose parameters are $(\mu_i, \sigma_i)_{i=1}^{\ell}$ for both the mean and standard deviation of each input variable as depicted in algorithm 4. The Gaussian distribution for each input variable are updated incrementally with a fraction of each tournament's winner, as the formula in Equation (3.1) and (3.2) indicate for μ_i and σ_i respectively, where x_i^t is the i-th component of the candidate $\mathbf{x}_t$, n the number of candidates and μ_i

and σ_i are initialized with 0.

$$\begin{aligned} \mu &= \mu + \tfrac{1}{n} x_i^t && (3.1) \\ \sigma_i^2 &= \sigma_i^2 - \mu_i^2 && (3.2) \\ t &= 1, \ldots, n. \end{aligned}$$

Algorithm 4 TILDA

Requires: $\ell > 0$, fitness function $f(\cdot)$
Outputs: best candidate $\boldsymbol{\beta}$

$\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = \mathbf{1}, \boldsymbol{\beta} = \mathbf{0}, \gamma = 0.5, \epsilon = 0.01$
repeat until convergence ending citeria not met
  $\tilde{\boldsymbol{\mu}} = \tilde{\boldsymbol{\Sigma}} = \mathbf{0}$
  repeat n **times**
    $\{\mathbf{x}', \mathbf{x}''\} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
    $\hat{\mathbf{x}} = \mathsf{argmax}(f(\mathbf{x}'), f(\mathbf{x}''))$
    $\boldsymbol{\beta} = \mathsf{argmax}(f(\boldsymbol{\beta}), f(\hat{\mathbf{x}}))$
    $\tilde{\boldsymbol{\mu}} = \tilde{\boldsymbol{\mu}} + \frac{1}{n}(\hat{x}_1, \hat{x}_2, \ldots, \hat{x}_\ell)$
    $\tilde{\boldsymbol{\Sigma}} = \tilde{\boldsymbol{\Sigma}} + \frac{1}{n}(\hat{x}_1^2, \hat{x}_2^2, \ldots, \hat{x}_\ell^2)$
  end repeat
  $\boldsymbol{\mu} = \boldsymbol{\mu} - \gamma(\boldsymbol{\mu} - \frac{1}{2}(\boldsymbol{\beta} + \tilde{\boldsymbol{\mu}}))$
  $\boldsymbol{\Sigma} = \boldsymbol{\Sigma} - \gamma(\boldsymbol{\Sigma} - (\tilde{\boldsymbol{\Sigma}} - (\tilde{\mu}_0^2, \tilde{\mu}_1^2, \ldots, \tilde{\mu}_\ell^2)) + \epsilon$
end repeat until convergence

Chapter 4

Goldenberry

Chapter 5

wKieraII

Chapter 6

Conclusion

Bibliography