Noname manuscript No.
(will be inserted by the editor)

Local Feature Selection with Gaussian Process Regression

Karim Pichara · Alvaro Soto

Received: date / Accepted: date

Abstract Keywords Feature Selection · Local discriminative subspaces · Gaussian Process · Expectation Maximization · Nearest Neighbor · Classification

Most feature selection algorithms determine a global subset of features, where all data instances are projected in order to improve classification accuracy. An attractive alternative solution is to adaptively find a local subset of features for each data instance, such that, the classification of each instance is performed according to its own selective subspace. This paper presents a novel application of Gaussian Processes that improves classification performance by learning discriminative local subsets of features for each instance in a dataset. Gaussian Processes are used to build for each available feature a function that estimates the discriminative power of the feature over all the input space. Using these functions, we are able to determine a discriminative subspace for each possible instance by locally joining the features that present the highest levels of discriminative power. New instances are then classified by using a K-NN classifier that operates in the local subspaces. Experimental results show that by using local discriminative subspaces, we are able to reach higher levels of accuracy than alternative state-of-the-art feature selection approaches.

1 Introduction

Performing classification in high dimensional spaces has strong limitations for most classification models, mainly due to the Curse of Dimensionality problem and the usual high computational load [3] [29]. In case of distance based models is hard to reach high

Pontificia Universidad Católica de Chile
Vicuña Mackenna 4860, Macul, Santiago, Chile
Tel.: +562-3544440
E-mail: kpb@ing.puc.cl

Pontificia Universidad Católica de Chile
Vicuña Mackenna 4860, Macul, Santiago, Chile
Tel.: +562-3544440
E-mail: asoto@ing.puc.cl

levels of accuracy since distance metrics lose part of their discriminative power in high dimensions [3]. In case of probabilistic models that estimate class conditional density functions, the ability to perform a good fit using a training set decreases exponentially with the number of features [5].

Traditional dimensionality reduction techniques, such as feature selection and feature transformation, partially overcome the previous problem [21] [18]. Global feature selection removes irrelevant and redundant dimensions by finding a global subset of features where all the data instances are projected. Feature transformation reduces dimensionality by summarizing the dataset by means of algebraic combinations of dimensions. As a relevant weakness, in both cases all the data instances are project into the same subspace in order to improve classification. Furthermore, in the case of feature transformation, the resulting new features usually do not have a clear interpretation.

In many applications, particularly in high dimensions, projecting all data instances into the same subspace does not produce satisfactory results. For example, in a face recognition application each individual usually has particular visual characteristics, such a particle nose shape, that are more suitable represented by a specific subset of features. In this case, a natural extension of global feature selection is to select a particular local subset of discriminative features for each data instance. Unfortunately, an exhaustive search for suitable local subspaces is generally not possible, particularly in high dimensions, where there are $2^n - 1$ possible subsets of features to characterize each data instance in a n-dimensional space.

In this paper, we present a local feature selection algorithm that for each data instance selects a subset of features, such that by projecting the instance to the resulting subspace, we facilitate its classification. Our model starts by finding for each feature a function that estimates the discriminative power of the feature over all the input space. We estimate these functions using Gaussian Processes (GP) regression [35][26]. To achieve the estimation we propose an iterative optimization method to jointly select the observations included in the regressions and the GP hyperparameters. After learning the GPs for all features, we are able to find a discriminative subspace for each possible input by locally joining the features presenting higher levels of discriminative power.

Accordingly, the main contributions of our work are: i) A new local feature selection algorithm, able to find a particular discriminative subset of features for any position of the input space, ii) A new view that poses the feature selection problem as a regression task using a discriminative score based on GPs, and iii) A new optimization method that uses an iterative strategy to jointly determine the informative observations and the hyperparameters of the GPs.

This paper is organized as follows. Section 2 presents a brief overview of GPs. Section 3 describes related work on global and local feature selection techniques. Section 4 presents the main details of our method. Section 5 shows experimental results. Finally Section 6 presents the main conclusions of this work.

2 Gaussian Processes

A GP is a collection of random variables $V = \{v_1, \ldots, v_n\}$ such that any finite subset of variables from V has a joint Gaussian distribution. A GP is fully specified by a mean function $M(x)$ and a covariance function $K(x_i, x_j)$, becoming a distribution over functions of possible infinite dimensional mean and covariance matrix. Covariance function $K(x_i, x_j)$ typically depends on some hyperparameters that have to be determined. GPs have proven to be a powerful tool for regression, being able to model smooth functions giving uncertainty about estimations [35]. In a standard regression problem, we estimate a function $F(x) : x \in X \rightarrow y \in Y$ given a set of observations $A_y \in Y$ for a set of values $A_x \in X$. In GP regression, the jointly Gaussian distributed variables are any subset of predictions $S_y \subset Y$, with mean $M(S_y)$ and covariance matrix $K_{ij} = K(S_{x_i}, S_{x_j})$ where $S_x \in X$ are the respective pre images of S_y under F. Note that the covariance functions for the output variables S_y depend only on the input variables S_x. The posterior mean and variance of S_y given observations A_y with respective points A_x are [35][26]:

$$\mu_{S_y|S_x,A_x,A_y} = \mu_{S_y} + \Sigma_{S_x A_x} \Sigma^{-1}_{A_x A_x} (A_y - \mu_{A_y}) \tag{1}$$

$$\sigma^2_{S_y|S_x,A_x,A_y} = K(S_x, S_x) - \Sigma_{S_x A_x} \Sigma^{-1}_{A_x A_x} \Sigma^T_{S_x A_x} \tag{2}$$

where $\Sigma_{S_x A_x}$ is a covariance vector with one entry for each $m \in A_x$ with value $K(S_x, m)$. It is important to note that the posterior variance of S_y does not depend on the observations A_y. In other words, the uncertainty about the predicted values for S_y only depends on the points A_x related to the observations A_y.

3 Related Work

An extensive literature presents many algorithms to select global subsets of features for classification [20][17][21][22][11].

In supervised scenarios, common methods are divided in two main types, filters and wrappers [17]. In filter models, feature selection is performed as a preprocessing step, searching for features with some criterion that does not depend on the classifier, ranking the features according to that global criterion. Close to our ideas, many methods find features that are discriminative to classify data points. Guided by this idea, the Relief algorithm [20] selects features on two class classification problems based on calculating weights for each dimension. To calculate the weights, they randomly sample data points and update the weight of each dimension based on the difference between the sampled data point and its nearest elements from the same and different class (nearest hit and nearest miss respectively). That results in high weights for features that are more discriminative for classification, unfortunately this algorithm works only for two class problems and it do not detect redundant features. Some extensions of the Relief algorithm have been proposed [23][36]. The algorithm Relief-F [23] improves the robustness of Relief. They consider the k nearest misses and hits and extend the approach to the multi-class case, also showing better results dealing with noisy data. Regressional Relief-F (R-Relief-F) [36] instead of using information about the class of each data point to estimate nearest hits and misses, they estimate a probability that the predicted class value of two instances are different. They model this probability with a relative distance between the predicted class of two instances. Conditioning each probability on the predicted class value, the weight of each attribute is estimated using Bayes's rule. Relief approaches do not consider cases where some attributes are discriminative for only some instances.

A different filter approach to search for relevant features is to evaluate relations between attributes and the attribute representing the class of the instances. In [14] they perform a best first strategy guided by a score depending on the statistical correlation between the features and the attribute representing the class of instances. In [13] they proposed the inconsistency criterion, that gives a score to features depending on the number of “inconsistencies” or cases where similar elements belong to different classes. In [8] they use an information theoretic criterion to select variables in a text categorization problem, calculating the mutual information between each feature and the class label to estimate a score of relevance. This criterion is suitable for problems where the probabilities needed to calculate the mutual information can be empirically estimated through frequencies of occurrences among data. In [22] they propose an heuristic algorithm that estimates the Markov blanket for each feature in order to allow the elimination of features that do not add information to the class label.

An alternative approach to filters are wrapper methods. In wrapper methods feature selection is guided by a specific classification algorithm, searching for more suitable features that improves the accuracy for that particular classifier [17] [21]. In these models we need to define: (i) How to search the space of all possible subsets of variables; (ii) How to assess the classification accuracy of a classification algorithm to guide the search and halt it; and (iii) Which classifier to use. Some algorithms use genetic approaches to search over the hypothesis spaces. In [30] they propose an algorithm that uses an evolutive search method with leave-one-out cross validation to select subsets of features. They evaluate features in parallel until one feature is better than the others. They use a 1-nearest neighbor classifier as a predictor model. It is common to find add hoc heuristics related to specific classifiers in wrapper approaches, for example in [25] they select features for a Naive Bayes classifier removing variables that introduce dependencies and using leave-one-out cross validation to validate the candidate subsets of attributes. In [32], they search for dependencies among features, creating groups of variables that share dependencies, adding new features represented by these compounds in order to improve a Naive Bayes classifier. In [39] they select features for a Bayesian network classifier. They first select the class label as the initial feature, then iteratively add the variable having the higher increment in accuracy of a Bayes network estimated from the test set. In [40] they select subset of features to use with nearest neighbor classifier, they develop a Monte Carlo technique to choose the subset of prototype training instances and features as well. SVM classifiers also can help the feature selection process [12]. A recursive wrapper feature elimination algorithm is proposed in [12], they use SVM as the underlying classifier to recursively eliminate the features that produce a minimal contribution to the margin of the classifier.

In general wrapper models have high computational cost because they have to run an induction algorithm multiple times, moreover the fact that guiding the feature selection to improve the accuracy of one particular classifier usually increase the risk to suffer overfitting [21].
Due that the heuristics that evaluate the performance of subsets of features are global (in the sense they return scores related to all data points), filter and wrapper approaches are not able to select these subsets for particular data points, requiring different strategies to drive a local subset selection.

A variant of wrappers methods for feature selection are embedded methods. In these methods feature selection is performed within the classification training process, similar to wrapper methods, the selection is linked with a particular classifier, the main difference is that the selection of features and the training of the classifier are linked and dependent processes. This particular characteristic constitutes an advantage in time complexity. Examples of embedded methods are decision trees [34] and boosting methods [9]. Decision tree classification algorithms select as nodes of the trees variables that well discriminate among elements from different classes. First selected variables are placed from root to leafs as an indicator of the discriminative power of each variable. Boosting methods determine ensembles of classifiers, each of these classifiers are experts classifying some areas of the input space. For each of these "weak classifiers" the algorithm select the variables that improves its particular classification. The final decision is taken by the ensemble that combines the decision of each of the weak classifiers. SVM classifiers also have been used as underlying classifiers for embedded approaches [4][31]. In [31] they use a penalization term on the number of selected features to the standard cost function of the SVM, by optimizing the modified cost function features are selected simultaneously to the model construction. ConcaVe feature selection is proposed in [4], they minimizes an approximate zero norm of the standard weights vector in the SVM optimization, using an iterative method called Successive Linearization Algorithm [4]

Distance Metric Learning can also be casted as a feature selection approach, where a distance matrix is learned in order to improve classification. Commonly these methods find linear transformations of the input space in order to minimize distances among elements from the same class and maximizing distances among elements from different classes. Some of these methods performs the linear transformation by computing Mahalanobis distance which calculus depends on a positive semidefinite matrix that can be determined by numerical optimization of a predefined loss function. Other methods perform the linear transformation by finding a matrix that directly reduces the dimensionality of the input space. Relative to distance matrix methods, in [37] they propose a Pseudometric Online Learning Algorithm (POLA). POLA receives at each step pairs of input data points and attempts to learn a Mahalanobis metric M and a scalar threshold b such that similarly labeled inputs are at most a distance of $b - 1$ apart, while differently labeled inputs are at least a distance of $b + 1$ apart. The distance metric M and threshold b are updated after each tuple to correct any violation of the desired relation. Another related distance metric algorithm is Neighborhood Component Analysis (NCA) [10], it computes the expected leave-one-out classification error from a stochastic variant of kNN classification. The stochastic classifier uses a Mahalanobis distance metric parameterized by the linear transformation. The algorithm attempts to estimate the linear transformation that minimizes the expected classification error. Unfortunately many of these algorithms obtain poor results in cases where there are many groups of data points from the same class far from each other (multimodal labeled data). In [41], they propose a Local Fisher Discriminant Analysis technique to deal with multimodal data. The model merges ideas from an unsupervised technique called Locally Preserved Projections (LPP) [16] and Fisher discriminant analysis, with is a discrimination indicator for features that perform poor in presence of multimodal data.

Also related with NCA and POLA, a more recently work is proposed in [42], named Large Margin Nearest Neighbor Classifier (LMNN). They define a safety perimeter for each instance where they expect to have only instances of the same class (target neighbors), any different class elements within the perimeter are impostors. Before learning, a training input has both target neighbors and impostors within its safety perimeter, during learning, impostors are pushed outside, after learning, there exist a finite margin between the perimeter and the impostors. To push out the impostors they define a loss function that consists in two terms, one which acts to pull target neighbors closer together and another which acts to push differently labeled examples further apart. Each of these terms depends on a distance matrix, the goal is to find the matrix that maximizes the loss function. The loss function is expressed in terms of a positive semidefinite distance matrix, which allows to find an optimum value in a closed form. Many other similar methods have been proposed [6][2] [43] [38]. Most of distance metric learning methods also find one distance matrix to be applied over all the instances, unfortunately in some cases the learned matrix is suitable for only a subset of data points.

There are many other kind of approaches that can be viewed as a feature selection techniques. For example in object recognition by image analysis, the method proposed in [33] groups the images taken from a specific view and build a feature space for each view. This idea is extended in [19] with the Locally Linear Discriminant Analysis, where they find linear transformations that map data points to lower dimensional spaces where the inter-class covariance is maximized and the intra-class covariance is minimized. In [15] the Optimal Local Basis method uses a reinforcement learning approach where each state is represented by the last set of features selected for a given point, the action corresponds to the selection of a new feature, and the reward depends on the classification accuracy given the selected pool of features.

As we mentioned before, most of the approaches that attempts to select features do not consider to select a specific set of features for each data point. Our model constitutes a function that is able to determine a set of features for any test data point, this function do not suffer from most of the problems mentioned above, such as redundant features and multimodal data among others.

4 Our Approach

One of the key steps behind our approach is the estimation of a set of functions that evaluate the discriminative power of each feature over the entire input space. Each of these functions are estimated with GP regression using as observations the discriminative score of the feature for some strategically selected data points. We give the details about the calculus of the discriminative score and the strategy of observation selection in next sections. Figure 1 illustrates the process of building functions to estimate the discriminative score of two features using GPs. In this case, for 2 hypothetical features F1 and F3, dimension 1 is discriminative to classify a data instance x_i, while dimension 3 is not. We can observe (black dots) that we add a high score value for the observation relative to x_i on GP1 and a low score value for the observation relative to GP3. Gray dots correspond to previous observations.

Next, we present first the discriminative score used in this work. Then, we describe our application of GPs to estimate the scoring functions related to each feature. Afterwards, we present our technique to locally join these functions in order to achieve local feature selection. Finally, we present a classification scheme that uses our approach.

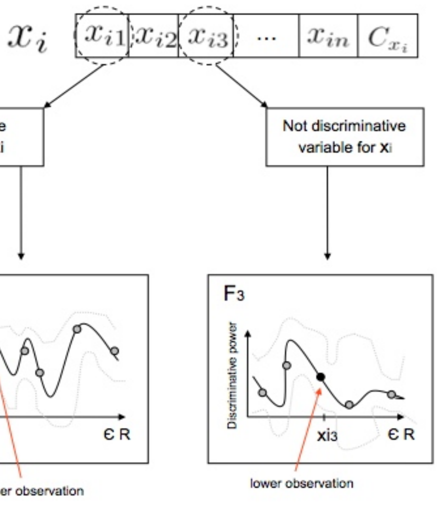


Fig. 1: Example of observations for GPs according to the discriminative score of dimensions 1 and 3 for data instance x_i. Dotted lines represents the GP uncertainty.

4.1 Discriminative Score

This section describes how we calculate the discriminative score of a feature for a given data point, remember this calculus is used to generate the observations needed to estimate the functions described above. In this work, we measure the discriminative power of each variable using a score that resembles the near-hit and near-miss strategy proposed by the Relief feature selection algorithm [20]. Let $x_i = [x_i(1), x_i(2), \ldots, x_i(N), C_{x_i}]$, $i \in [1 \ldots M]$, be a data instance that belongs to class C_{x_i}, and let $j \in [1 \ldots N]$ be a feature. The discriminative score of feature j with respect to x_i is calculated considering how close is x_i to elements of its class and how far is from elements from other classes along dimension j.

Accordingly, the discriminative score $F_j(x_i)$ for dimension j at the input location x_i is given by:

$$F_j(x_i) = \frac{1}{\eta} \sum_{k=1}^{M} z(x_i, x_k) K(d_{ik}^j), \tag{3}$$

where $K(\cdot)$ is a Gaussian kernel given by

$$K(d) = \frac{e^{-d^2/2}}{\sqrt{2\pi}},$$

and

$$d_{ik}^j = \begin{cases} |x_i(j) - x_k(j)| & \text{if } j \text{ is continuous} \\ \delta_{x_i(j)x_k(j)} & \text{if } j \text{ is categorical} \end{cases}$$

also

$$\delta_{x_i(j)x_k(j)} = \begin{cases} 0 & \text{if} \quad x_i(j) = x_k(j) \\ 1 & \text{otherwise} \end{cases}$$

Where $z(x_i, x_k) = -1$ if x_i and x_k belong to the same class and $z(x_i, x_k) = 1$ otherwise. Finally, η is a normalization constant that keeps the discriminative score in the range [0..1]. Note that function $d(\cdot,\cdot)$ can be extended to other types of variables (ordinal,binary, scaled, etc.). With equation (3) we are able to add observations to regressions $\{F_1, F_2, \dots, F_N\}$ at any location of the input space.

4.2 Estimation of regressions for discriminative scores

As we mentioned, we use GPs to estimate the discriminative scoring function associated to each feature. GPs have the property of allowing us not only to estimate a regression, but also to provide information about the level of uncertainty of the estimation in different parts of the input space. This last property is very important to the practical implementation of our method. In general, there are important limitations in the number of observations that is possible to include in a GP regression. Specifically, for m observations in a regression model, the training cost for each GP is $O(m^3)$. Fortunately, using the estimation of the level of uncertainty provided by the GP, we can implement efficient strategies to include observations (evaluations of the discriminative score) in key informative areas of the input space.

A popular solution to select observations for a GP regression is provided in [24]. In this work, they propose an active learning scheme for select locations in a sensor placement problem, where there is a limited number of positions to allocate the sensors. They select points that reduce the entropy of the posterior distribution of the parameters given new observations. Unfortunately, this method does not scale properly with the number of possible observations, therefore is computationally infeasible for our case, where we can potentially observe at any point in the input domain. In fact, for a discretization of a two parameter covariance function in d possible values, the calculation of the expected reduction of entropy for one given point, takes $O(mNd^2)$ for a dataset of m points and N variables.

An additional inconvenience to apply a GP based solution to our case is the estimation of hyperparameters. In effect, in a GP regression problem the hyperparameters of the GP are usually unknown, therefore we also need to select observations to estimate their values.

Accordingly, we need to solve two main problems to perform the GP regressions: i) Determine the set of points to be included as observations on each of the GPs, and ii) Determine the Kernel hyperparameters of each of the GPs. This is a chicken-and-egg problem, because we need the observations to estimate the hyperparameters and vice versa. To solve this problem, we use a strategy that resembles the operation of the Expectation Maximization algorithm [7]. We iterate between two steps, select observations given GP hyperparameters, and find GP hyperparameters given a set of observations. We start the iterations by randomly selecting initial values for the GP hyperparameters. Afterwards, we iterate the following two steps:

4.2.1 Determination of the set of observations to include in the regression given the hyperparameters.

Assuming a known set of hyperparameters for a GP, McKay et al. [27] shows that for a fixed covariance function, we can obtain a suitable estimation of the GP by greedily selecting observations according to the position with highest posterior variance, as given in Equation (2). The advantage is that this strategy requires only one data scan to determine the next point to be observed. Using this strategy we can also determine a suitable number of observations to be included in the regression. This is achieved by calculating the mean squared error (MSE) [35] of the regression in a test set given the discriminative scores (observations) of the selected points so far. The test set is simply a set of points where we calculate the discriminative score using equation 3. We check in our experiments that after being adding a determined number of observations, the MSE remains almost constant. That suggest us to add observations until the MSE do not change with the addition of more observations.

4.2.2 Determination of hyperparameters given observations

Given a set of observations, a suitable approach to determine the kernel hyperparameters of a GP is the maximum likelihood (ML) method. As shown in [35], an efficient strategy to find this maximum is to use a conjugate gradient technique that optimizes the marginal loglikelihood given a set of observations. Specifically, considering a set A_y of n observations relative to a set of data points A_x, the marginal loglikelihood, as a function of the hyperparameters Θ of a GP, is given by:

$$\log p(A_y|A_x,\Theta) = -\frac{1}{2}A_y^t \Sigma_{A_y}^{-1} A_y - \frac{1}{2}\log|\Sigma_{A_y}| - \frac{n}{2}\log 2\pi \tag{4}$$

The first term in Equation (4) corresponds to the data fit, i.e., how the regression fits the observations, the second term is the complexity penalty that depends on the number of inputs and the covariance function, and the third term is a normalization constant. We use the conjungate gradient method to numerically find Θ that maximizes Equation (4).

In this work, we use a covariance based on a squared exponential isotropic function with Gaussian noise, a commonly used covariance function for machine learning problems [35]. This covariance function is given by:

$$k(x_p, x_q) = \theta \exp \frac{-(x_p - x_q)^T P^{-1}(x_p - x_q)}{2} + \sigma_n^2 \delta_{pq}$$

where $P = l^2 I$ and I is the identity matrix, θ is the signal variance, σ_n^2 is the noise variance, l is the characteristic length scale, and δ_{pq} is the Kronecker delta which is one if $p = q$ and zero otherwise. We use the same covariance function and parameters for all the GPs.

Algorithm 1 summarizes the main steps of the strategy used to estimate hyperparameters and to select observations. The algorithm iterates until there are not significant changes in the hyperparameter values. We evaluate this by checking changes between consecutive iterations. We empirically determine a suitable threshold to stop the iterations.

To guarantee convergence of Algorithm 1, we should check if there is an increase of the likelihood function in Equation (4) at each iteration. In case of the step to find new hyperparameters, clearly this is a maximization step, therefore, the log-likelihood increases. In case of the step to select new observations, we can consider that the next selected observation always reduces the variance of the regression. As a consequence, this also reduces the penalization term in Equation (4), which in turns increases the log-likelihood.

Algorithm 1 Strategy to estimate hyperparameters and to select observations

```
Randomly select an initial set of hyperparameters.
while No significant change on hyperparameters do
    Given the hyperparameters select the observations (sec. 4.2.1)
    Given the observations, determine the hyperparameters (sec. 4.2.2).
end while
```

4.3 Construction of local discriminative subspaces

Once we have estimated the GP regression for each feature, we are able to select a discriminative subspace for each possible input. We achieve this by locally selecting the features presenting highest levels of discriminative power. To avoid selecting redundant features, we include features in the set sequentially, avoiding to include features that present high correlation with the features already selected. Formally, let L_{x_i} be the discriminative subspace for x_i, the construction of L_{x_i} is performed by adding the feature with greater discriminative power penalized by its correlation with respect to the features already in L_{x_i}. The discriminative power for variable j at point x_i is denoted by $D^i(j)$ and is calculated by considering the height of the estimation (mean of $F^j(x_i)$) and the uncertainty (variance of $F^j(x_i)$).

Given that $F^j(x_i)$ is itself a Gaussian distribution, we can evaluate $D^i(j)$ calculating how far is $F^j(x_i)$ from a fixed distribution $y^* \sim G(\mu^*, (\sigma^*)^2)$. To calculate the difference between $F^j(x_i)$ and y^* we use the Kullback Leibler (KL) divergence. We manually set low values for μ^* and σ^* (the same among all the GPs). We use y^* just as a reference distribution to obtain a relative value among all the features at the point x_i.

Figure 2 illustrates this idea for a case where there are two features: j and k. In this case, distribution $y^j(x_i)$ is farther from distribution y^* compared to distribution $y^k(x_i)$, indicating that variable j is more discriminative than variable k with respect to input x_i.

Accordingly, the final score of a variable at a specific instance location x_i is given by:

$$Score_{x_i}(j) = [D_{x_i}(j)] - \frac{1}{\eta} \sum_{k=1}^{|L_{x_i}|} |Corr(j, L_{x_i}(k))|,$$

where $D_{x_i}(j)$ is the discriminative power of variable j for x_i and $Corr(j, L_{x_i}(k))$ is the correlation between j and k-th variables in L_{x_i}. Constant η is used to normalize the sum in order to compare scores. The selection process is detailed in the algorithm 2

Note that γ is a parameter that regulates the number of features we select. In this work we set that parameter manually on each dataset. In section 5.6 we show how the results changes with different values of this parameter.

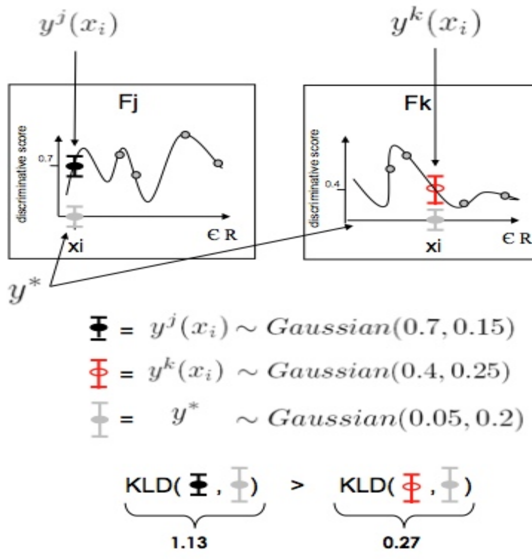

$y^j(x_i)$
$y^k(x_i)$
Fj
Fk
discriminative score
0.7
0.4
xi
∈ R
y^*
$= y^j(x_i) \sim Gaussian(0.7, 0.15)$
$= y^k(x_i) \sim Gaussian(0.4, 0.25)$
$= y^* \sim Gaussian(0.05, 0.2)$
KLD(,) > KLD(,)
1.13
0.27

Fig. 2: In the example, for input x_i variable j is more discriminative than variable k due to its greater KL divergence score with respect to the reference distribution y^*.

Algorithm 2 Selection of the discriminative Subspace for x_i

```
Calculus of the discriminative subspace $L_{x_i}$
$A$ : Let $m$ the feature with highest discriminative power $D_{x_i}(m)$
Initialize $L_{x_i} = \{m\}$
$B$ : Calculate $Scores = \{Score_{x_i}(1), \dots, Score_{x_i}(m-1), \dots, Score_{x_i}(m+1), \dots, Score_{x_i}(N)\}$ for the remaining features
Let $T = Mean(Scores) + \gamma * Std(Scores)$
Let $h$ the feature with highest score ($Score_{x_i}(h)$)
if $Score_{x_i}(h) \geq T$ then
  $L_{x_i} = L_{x_i} \cup h$
  return to $B$
else
  Stop
end if
```

4.4 Classification of new data instances

For a new test instance x_t, we proceed to build its local subspace L_{x_t} by using the method described in Section 4.3. Afterwards, we project all the training instances to L_{x_t}, and we apply a K-NN classifier in this subspace.

4.5 Running time of the method

In the training process, the selection of locations needs the estimation of the posterior variance at each of the possible locations, this takes $O(n^3)$ due the invertion of the covariance matrix in Equation (2). It is important to note that the posterior variance depends mainly on sums of distances between the candidate location and the locations already selected (due to the fact that we are using a squared exponential isotropic covariance function). Moreover, we do not need to calculate the inverse of the covariance matrix in order to select the next locations, given that this matrix is constant when we compare the posterior variance among the candidate locations. This result can be obtained from Equation (2). Suppose we have to choose the next location S_x (whose respective observation is S_y) that maximizes the posterior variance $\sigma^2_{S_y|S_x,A_x,A_y}$. Also, suppose that we already have selected B locations A_x (with respective observations A_y). The posterior variance of S_y can be expressed as:

$$\begin{aligned}\sigma^2_{S_y|S_x,A_x,A_y} &= K(S_x,S_x) - \Sigma_{S_xA_x}\Sigma^{-1}_{A_xA_x}\Sigma^T_{S_xA_x} \\ &= K(S_x,S_x)- \\ &\quad -\textstyle\sum_{i=1}^{B}[K(S_x,A_{x_i})* \\ &\quad *\textstyle\sum_{t=1}^{B}C_{i,t}K(S_x,A_{x_t})],\end{aligned} \tag{5}$$

where $C_{i,t} = \Sigma^{-1}_{A_xA_x}(i,t)$ does not depend on S_x. Using the definition of the squared exponential isotropic covariance function we can write:

$$\begin{aligned}\sigma^2_{S_y|S_x,A_x,A_y} &= K(S_x,S_x) - \textstyle\sum_{i=1}^{B}[K(S_x,A_{x_i})* \\ &\quad *\textstyle\sum_{t=1}^{B}C_{i,t}K(S_x,A_{x_t})] \\ &= \eta- \\ &\quad \textstyle\sum_{i=1}^{B}[\theta e^{\frac{-(S_x-A_{x_i})^T P^{-1}(S_x-A_{x_i})}{2}}* \\ &\quad *\textstyle\sum_{t=1}^{B}C_{i,t}\theta* \\ &\quad *e^{\frac{-(S_x-A_{x_t})^T P^{-1}(S_x-A_{x_t})}{2}}]\end{aligned} \tag{6}$$

The constant η is the sum of constant $K(S_x,S_x)$ and the noise variances. We can see that the instance S_x that maximizes the posterior variance $\sigma^2_{S_y|S_x,A_x,A_y}$ is the instance that also minimizes the right term in Equation (6). This term requires the computation of a sum of differences between the candidate elements in the training set and the locations already selected in A_x. This operation takes $O(n^2)$ instead of the cubic inversion of the covariance matrix $\Sigma_{A_xA_x}$.

Given that the selection process is performed for each of the features, to learn all of the GPs the total cost is $O(mn^2)$ for a dataset with m variables.

At testing time, the estimation of the local discriminative space for a query instance requires the estimation of the respective posterior mean and variance at each of the GPs. This takes $O(mB^3)$ operations for a set of B observations included in the GPs.

Unfortunately, the computational cost calculated above is still expensive for some datasets, this forces us to perform a preprocessing step in order to reduce the number of candidate locations n at each iteration. We achieve this by preprocessing data adding a clustering step using a k-medoids algorithm that provides the cluster centers as the possible candidate locations.

5 Experimental Results

We evaluate our approach using synthetic and real datasets. We compare the accuracy in classification against four global feature selection algorithms and two distance metric learning techniques. 1) Best First with CFS subset evaluator [14], 2) Greedy Stepwise with Consistency Subset Evaluator [13], 3) Relief-F [23], 4) A wrapper approach with a Naive Bayes Classifier [25], and two Distance metric learning algorithms: 1) Local Fisher Discriminant Analysis (LFDA) [41] and 2) Large Margin Nearest Neighboor Classifier (LMNN) [42]. We use a K-NN classifier with the set of features obtained by each of the feature selection algorithms. In our experiments we notice that different values of K do not change substantially the relative results among the algorithms, we choose $K = 8$ in all the experiments. To evaluate accuracy, we use 10-fold cross-validation. We also provide an analysis about the strategic selection of observations, the levels of accuracy respect to the percentage of data used in regressions, and the robustness of our strategy to find GP hyperparameters with respect to different starting values.

5.1 Synthetic Dataset

In this case we generate synthetic data according to the main hypothesis of this work. The generation process starts by generating a random set of candidate subspaces, then each data point is generated from one of the candidate subspaces. Points generated from the same subspace have the same label. Different subspaces can also generate data points with the same label. As underlying distributions we use Gaussian distributions. To add noise we generate some points with uniform distributions. Algorithm 3 shows the main steps of the data generation process:

Algorithm 3 Generation of Synthetic data

```
To generate $n$ instances $x_i \quad i \in [1 \dots n]$ from $c$ different classes in a $d$ dimensional space
Randomly generate a set of $B = \{S_1, S_2, \dots, S_n\}$ subspaces. $|S_i| \leq d$
Generate a normal distribution $G_i \quad i \in [1 \dots d]$ for each variable in subspaces from $B$
Let $B' = B \cup S_\emptyset$.
Associate to each $S_i \in B'$ one class from 1 to $c$. Let $c(S_i)$ be the associated class for $S_i$
for $i = 1$ To $n$ do
  Randomly select one element $S_p$ from $B$‘
  Set the label of $x_i$ equal to $c(S_p)$
  if $S_p = S_\emptyset$ then
    Sample $x_i$ from an uniform distribution in the $d$ dimensional space
  else
    for $j = 1$ To $d$ do
      if $j \in S_p$ then
        Sample $x_i(j)$ from $G(j)$
      else
        Sample $x_i(j)$ from an Uniform distribution
      end if
    end for
  end if
end for
```

In this experiment we generate 500 data points with 50 features, four different classes and six candidate subspaces. Table 1 shows the classification performance for each of the algorithms under test. We can observe that our method outperforms the results of the other six global feature selection algorithms.

Table 1: Mean accuracy and standard deviation for different feature selection models and datasets using 10-Fold Cross-validation.

Method	Accuracy
Gaussian Process	**0.8542 ± 0.0343**
Best First - CFS	0.7625 ± 0.0298
Greedy Stepwise - CSE	0.7479 ± 0.0533
Relief-F	0.7500 ± 0.0529
Wrapper-NB	0.8063 ± 0.0598
LFDA	0.8000 ± 0.0623
LMNN	0.7896 ± 0.0632

5.2 Real Datasets

We use the following real datasets: Breast Cancer Dataset (Digitized images of a fine needle aspirate (FNA) of breast masses [1]), Isolet Dataset (Isolated Letter Speech Recognition [1]), Spectrometer Dataset (Infra-Red Astronomy Satellite Project Database [1]), and X-ray image Dataset (X-ray images from aluminum wheels [28]). See Table 2 for details of the datasets. In the Isolet dataset, we use a preprocesing k_m-medoids clustering with $k_m = 1000$.

Table 2: Details of real datasets.

Name	Instances	Features	Classes	Instances per class
Breast Cancer	569	31	2	357-212
Isolet	7797	617	26	$\approx$ 300 per class
Spectrometer	531	103	5	12-90-273-38-96
X-ray image	1780	336	2	869-911

Table 3 shows the classification performance for each of the algorithms under test. We can observe that our method outperforms the results of the other six feature selection algorithms. We run a T-test to check if the difference in mean and standard deviation on each result is not a random occurrence (Behrens-Fisher problem). We found that in all cases our improvement is statistically significative with a 90% of confidence, except for the X-Ray images in two cases (Best First CFS and LMNN), where our results are significative better with a 80% of confidence and in the Isolet dataset in three cases (Relief-F, Wrapper-NB and LMNN) where our results are significative better with just a 60% of confidence. Note that the algorithm works well despite the unbalanced proportion of classes, as in spectrometer dataset.

Table 3: Mean accuracy and standard deviation for different feature selection models and datasets using 10-Fold Cross-validation.

Method	Breast Cancer	Isolet
GPs	**0.9693 ± 0.0216**	**0.954 ± 0.0357**
Best First - CFS	0.9029 ± 0.0376	0.8726 ± 0.0232
Greedy Stepwise - CSE	0.9124 ± 0.0435	0.8613 ± 0.0324
Relief-F	0.9441 ± 0.0344	0.9531 ± 0.0217
Wrapper-NB	0.9429 ± 0.0479	0.9472 ± 0.0399
LFDA	0.9476 ± 0.0251	0.6734 ± 0.0315
LMNN	0.9476 ± 0.0333	0.9481 ± 0.0289

Method	Spectrometer	X-Ray Images
GP	**0.9062 ± 0.0422**	**0.8627 ± 0.0327**
Best First - CFS	0.8733 ± 0.0427	0.8452 ± 0.0542
Greedy Stepwise - CSE	0.8831 ± 0.0375	0.8294 ± 0.0363
Relief-F	0.8712 ± 0.0527	0.8215 ± 0.0490
Wrapper-NB	0.8694 ± 0.0492	0.8271 ± 0.0405
LFDA	0.8515 ± 0.0334	0.7927 ± 0.1106
LMNN	0.8693 ± 0.0463	0.8483 ± 0.0245

5.3 Analysis of the generalization power of the regression.

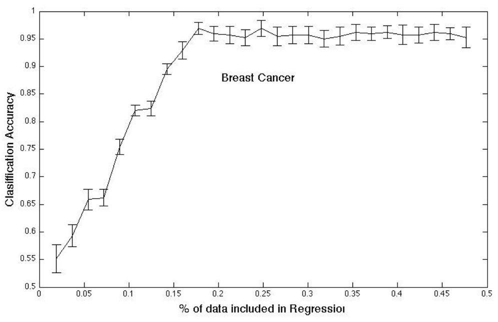
Here we analize the number of observations that our method uses to reach the accuracy values showed in table 3. Figure (3) shows the percent of data used as observations versus classification accuracy. For example in Breast Cancer dataset we use less than 20% of the available data to obtain good classification accuracy. In other cases like x-Ray images dataset, we need less than 70% of the available data to obtain good classification accuracy.

5.4 Analysis of the observation selection strategy

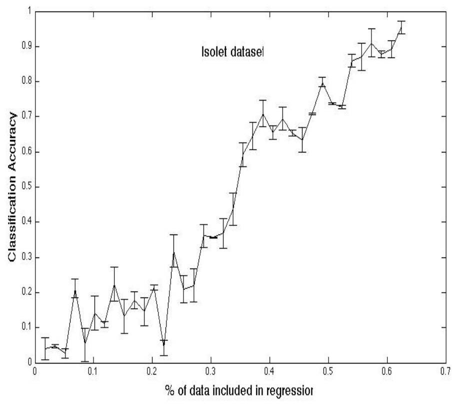
In this section, we evaluate the performance of our strategy to select observations considering the uncertainty of the estimation provided by the GPs. Figure 4 shows the error levels in regression versus the number of observations included. The error corresponds to the MSE between the predicted discriminative score provided by the model and the real discriminative score. Note that using Equation (3) we can directly evalue the real discriminative score. We can appreciate that selecting points considering uncertainty achieves low error levels with less observations included. This is true for all the datasets considered. Given that this analysis can be performed for each of the features available in the datasets, we present the average error among all of them. It is important to note that after adding a certain number of observations, the models overfits the training data, increasing the levels of errors in both cases. This is the main reason to worry about the number of points we should select as observations for GPs.

5.5 Analysis of the strategy to select hyperparameters and observations

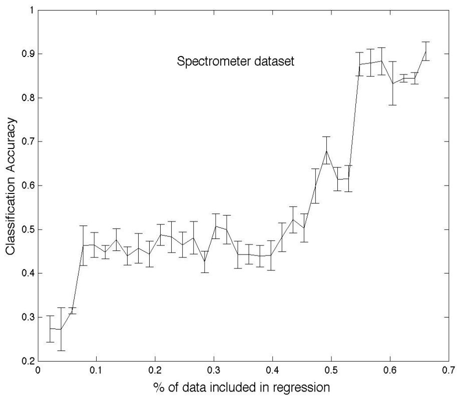
In this section we evaluate the sensibility of our model to different values of the hyper-parameters used to initialize Algorithm 1. To perform this, we use the same synthetic



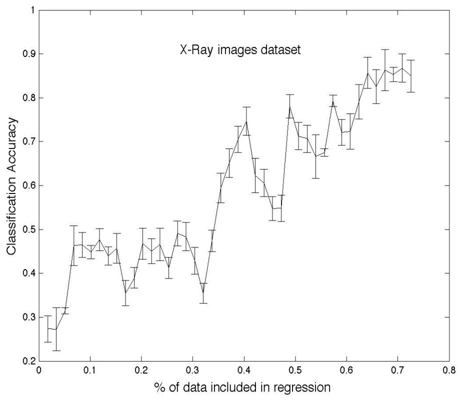
(a)



(b)



(c)



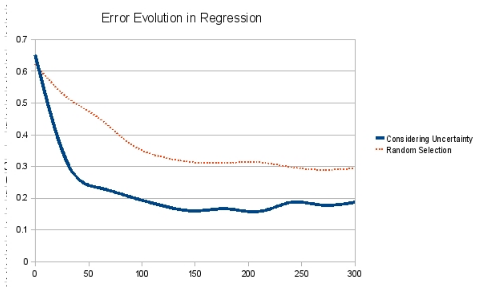
(d)

Fig. 3: Evaluation of the generalization capabilites for each real dataset. We can see that we need just a fraction of the available data as observations to reach suitable accuracy levels.

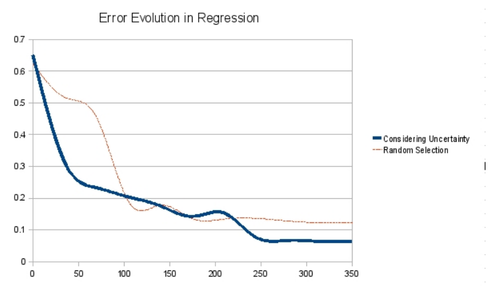
dataset as in 5.1. We run our model with 6 different starting values of the GP hyperparameters. Table 4 shows the results. We can see that the final values of the hyperparameters are slightly different for distinct starting points, however, the final classification results are not significantly affected. Note that the hyperparameters correspond to the characteristic length scale and the signal variance, respectively.

5.6 Accuracy in the detection of discriminative subspaces.

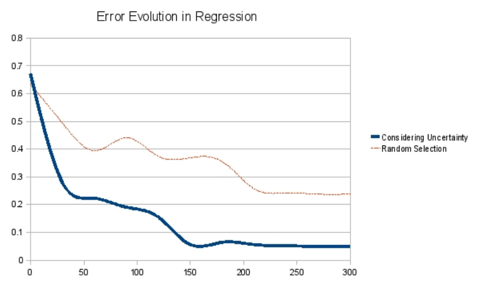
Given that using the synthetic dataset in 5.1 we know exactly the local subspaces from where we generate the data, we can compare that subspace with the ones detected



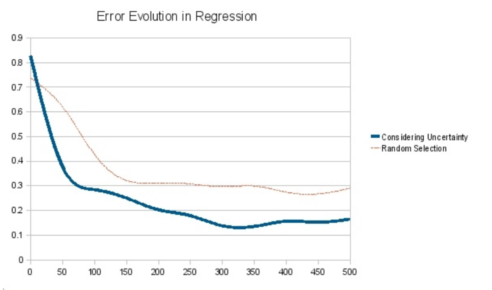
(a) Breast Cancer



(b) Isolet



(c) Spectrometer



(d) X-Ray Wheels

Fig. 4: Comparing levels of mean square error between selecting observations considering uncertainty or select observations randomly. X axis corresponds to the number of observations included in the regression.

Table 4: Classification accuracy for different starting values of the hyperparameters.

starting value	final value	Accuracy
$[0.3; 1.0]$	$[0.913; 0.991]$	0.8496
$[3.0; 1.0]$	$[1.137; 0.993]$	0.8732
$[1.0; 0.3]$	$[0.897; 0.833]$	0.8561
$[1.0; 1.0]$	$[0.972; 0.957]$	0.8691
$[1.0; 3.0]$	$[1.204; 1.093]$	0.8605

by our algorithm. To perform that we determine for each data point two indicators: 1) Recall: how many features from the correct subspaces we detect and 2) Precision: From the subspace we detect, how many features correspond to features included in the correct subspaces. We analyze our results for different values of the parameter γ. Table 5 shows the mean $\pm$ standard deviation values of precision and recall among the test data points. We can see that for higher values of γ the algorithm tends to choose less features on local subspaces, that achieves higher levels of precision but lower levels of recall. If we decrease the value of γ, the algorithm start choosing more features per subspace, that increases the recall levels but decreases the levels of precision.

Table 5: Comparison between the detected subspaces and the real subspaces.

γ	Precision	Recall
2.5	0.925 ± 0.201	0.55 ± 0.293
2	0.863 ± 0.232	0.66 ± 0.204
1.88	0.831 ± 0.239	0.69 ± 0.202
1.81	0.802 ± 0.239	0.71 ± 0.297
1.66	0.613 ± 0.271	0.86 ± 0.211
1.42	0.601 ± 0.243	0.89 ± 0.246
1.25	0.563 ± 0.285	0.94 ± 0.246

6 Discussion

We present a novel method to find local discriminative subspaces to project data instances in order to improve classification results. Our experiments show that the proposed method outperforms several traditional feature selection algorithms. The iterative strategy proposed allow us to approximately solve the well known problems of hyperparameter learning and observation selection. Our analysis also shows that selecting observations considering uncertainty requires less observations to achieve low error levels in regression, that is a desirable situation to reduce computational costs and to avoid overfitting. Our method contributes to eliminate noisy and redundant attributes that deteriorate the results of traditional distance based classifiers. A relevant contribution of our approach is to cast as a regression problem the representation of a score related to the discriminative power of each attribute. This allow us to take advantage of properties of GPs achieving an efficient estimation of the classification relevance of each feature over all the input space. An important advantage of our method is that it is not affected by the presence of unbalanced classes, because if any class is larger that the others, the model just select the most informative instances discarding the ones that do not contribute with extra information. As future work, we are developing new techniques for GP regression in order to deal with higher number of instances without losing accuracy in regression. Another important improvement is to evaluate discriminative power of subsets of features instead of single features, the greedy approach of best first search heuristic loses many subspaces where features becomes discriminative when they are joined with another features.

References

1. Asuncion, A., Newman, D.: UCI Machine Learning Repository (2007). URL http://archive.ics.uci.edu/ml/
2. Bar-Hillel, A., Hertz, T., Shental, N., Weinshall, D.: Learning a mahalanobis metric from equivalence constraints. Journal of Machine Learning Research **6**(1), 937–965 (2006)
3. Bellman, R.: Adaptive control processes - A guided tour. Princeton University Press, Princeton, New Jersey, U.S.A. (1961)
4. Bradley, P., Mangasarian, O.: Feature selection via concave minimization and support vector machines. In: Machine Learning Proceedings of the Fifteenth International Conference (ICML 98), pp. 82–90. Morgan Kaufmann (1998)
5. Chickering, D.: Learning from data: Artificial intelligence and statistics. Learning Bayesian networks is NP-Complete. Springer-Verlag (1996)
6. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similiarty metric discriminatively, with application to face verification. In: Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 349–356 (2005)

7. Dempster, A., Laird, N., Rubin, D.: Maximum likelihood from incomplete data via the em algorithm. Journal of the Royal Statistical Society. Series B (Methodological) **39**(1), 1–38 (1977)
8. Dumais, S., Platt, J., Heckerman, D., Sahami, M.: Inductive learning algorithms and representations for text categorization. In: Proc. of 7th Int. Conf. on Information and knowledge management, CIKM '98., pp. 148–155. ACM (1998)
9. Freund, Y., Schapire, R.: A decision-theoretic generalization of on-line learning and an application to boosting. In: Proc. of the Second European Conference on Computational Learning Theory, pp. 23–37. Springer-Verlag, London, UK (1995)
10. Goldberger, J., Roweis, S., Hinton, G., Salakhutdinov, R.: Neighbourhood components analysis. Advances in Neural Information Processing Systems **17**, 513–520 (2005)
11. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. Journal of Machine Learning (3), 1157–1182 (2003)
12. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. Machine Learning pp. 389–422 (2002)
13. H. Liu, H., Setiono, R.: A probabilistic approach to feature selection. a filter solution. In: Proc. of 13th International Conference on Machine Learning, pp. 319–327 (1996)
14. Hall, M., Smith, L.: Feature subset selection: a correlation based filter approach. In: N. Kasabov, et al. (eds.) Proc. of Fourth Int. Conf. on Neural Information Processing and Intelligent Information Systems, pp. 855–858 (1998)
15. Harandi, M., Ahmadabadi, M., Araabi, B.: Optimal local basis: A reinforcement learning approach for face recognition. Int. Journal of Computer Vision (IJCV) **81**(2), 191–204 (2009)
16. He, X., Niyogi, P.: Locality preserving projections. Advances in Neural Information Processing Systems **16** (2004)
17. John, G., Kohavi, R., Pfleger, K.: Irrelevant features and the subset selection problem. In: Proc. of Int. Conf. on Machine Learning, pp. 121–129. Morgan Kaufmann (1994)
18. Jolliffe, I.: Principal Component Analysis. Springer-Verlag (1986)
19. Kim, T., Kittler, J.: Locally linear discriminant analysis for multimodally distributed classes for face recognition with a single model image. IEEE Transactions on Pattern Analysis and Machine Intelligence **27**(3), 318–327 (2005)
20. Kira, K., Rendell, L.: The feature selection problem: Traditional methods and a new algorithm. In: 10th National Conf. on Artificial Intelligence, pp. 129–134 (1992)
21. Kohavi, R., John, G.: Wrappers for feature subset selection. Artificial Intelligence **97**(1-2), 273–324 (1997)
22. Koller, D., Sahami, M.: Toward optimal feature selection. In: Int. Conf. on Machine Learning, pp. 284–292 (1996)
23. Kononenko, I.: Estimating attributes: Analysis and extensions of relief. In: Proc. of European Conf. on Machine Learning, pp. 171–182 (1994)
24. Krause, A., Guestrin, C.: Nonmyopic active learning of Gaussian processes: An exploration-exploitation approach. In: Proc. of 24th Int. Conf. on Machine Learning (ICML), pp. 518–523 (2007)
25. Langley, P., Sage, S.: Induction of selective Bayesian classifiers. In: Proc. of 10th Conf. on Uncertainty in Artificial Intelligence, pp. 399–406 (1994)
26. Mackay, D.: Introduction to Gaussian processes. In: Book Neural Networks and Machine Learning, Springer-Verlag., pp. 84–92 (1998)
27. McKay, M., Beckman, R., Conover, W.: A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**, 239–245 (1979)
28. Mery, D., Filbert, D.: Automated flaw detection in aluminum castings based on the tracking of potential defects in a radioscopic image sequence. IEEE Transactions on Robotics and Automation **18**(6), 890–901 (2002)
29. Mitchell, T.: Machine Learning. McGraw Hill (1997)
30. Moore, A., Lee, M.: Efficient algorithms for minimizing cross validation error. In: Proc. of 11th Int. Conf. on Machine Learning, pp. 190–198 (1994)
31. Neumann, J., Schnrr, C., Steidl, G.: Combined svm-based feature selection and classification. Machine Learning (61), 129–150 (2005)
32. Pazzani, M.: Searching for dependencies in Bayesian classifiers. In: Proc. of 5th Int. Workshop on Artificial Intelligence and Statistics, pp. 239–248 (1995)
33. Pentland, A., Moghaddam, B., Starner, T.: View-based and modular eigenspaces for face recognition. In: IEEE Int. Conf. on Computer Vision and Pattern Recognition, pp. 84–91 (1994)

34. Quinlan, R.: Induction of decision trees. Machine Learning **1**(1), 81–106 (1986)
35. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. The MIT Press. (2006)
36. Robnik Sikonja, M., Kononenko, I.: An adaptation of relief for attribute estimation in regression. In: Proc. of 14th Int. Conf. on Machine Learning, pp. 296–304 (1997)
37. Shalev-Shwartz, S., Singer, Y., Andrew, N.: In: Proc. of the Twenty First International Conference on Machine Learning, pp. 94–101 (2004)
38. Shental, N., Hertz, T., Weinshall, D., Pavel, M.: Adjustment learning and relevant component analysis. In: Proc. of the Seventh European Conference on Computer Vision, vol. 4, pp. 776–792. Springer-Verlag (2002)
39. Singh, M., Provan, G.: A comparison of induction algorithms for selective and non-selective Bayesian classifiers. In: Proc. of 12th Int. Conf. on Machine Learning, pp. 497–505. Morgan Kaufmann (1995)
40. Skalak, D.: Prototype and feature selection by sampling and random mutation hill climbing algorithms. In: Machine Learning: Proc. of 11th Int. Conf., pp. 293–301. Morgan Kaufmann (1994)
41. Sugiyama, M.: Dimensionality reduction of multimodal labeled data by local fisher discriminant analysis. Journal of Machine Learning Research **8**, 1027–1061 (2007)
42. Weinberger, K., Blitzer, J., Saul, L.: Distance metric learning for large margin nearest neighbor classification. Journal of Machine Learning Research. **10**, 207–244 (2009)
43. Xing, E., Andrew, N., Jordan, M., Russell, S.: Distance metric learning, with application to clustering with side-information. **14**, 521–528 (2002)