Jointly Informative Feature Selection

Leonidas Lefakis
Idiap Research Institute
Martigny, Switzerland
leonidas.lefakis@idiap.ch

François Fleuret
Idiap Research Institute
Martigny, Switzerland
francois.fleuret@idiap.ch

Abstract

We propose several novel criteria for the selection of groups of jointly informative continuous features in the context of classification. Our approach is based on combining a Gaussian modeling of the feature responses, with derived upper bounds on their mutual information with the class label and their joint entropy.

We further propose specific algorithmic implementations of these criteria which reduce the computational complexity of the algorithms by up to two-orders of magnitude, making these strategies tractable in practice.

Experiments on multiple computer-vision data-bases, and using several types of classifiers, show that this class of methods outperforms state-of-the-art baselines, both in terms of speed and classification accuracy.

1 Introduction

It is often desirable, given a training set in $\mathbb{R}^D$ – labeled or not – to reduce its dimensionality by either extracting [13] or selecting [9] a number of features $d \ll D$, which carry "as much information as possible". The motivation behind this dimensionality reduction can be either to control over-fitting by reducing the capacity of the classifier space, or to improve computational requirements by reducing the optimization domain. It can also be used alone as a tool to facilitate the understanding or the graphical representation of high-dimension data.

In the present work we focus on the selection of features, which can be divided into two large families of methods. The first group, *filters*, are predictor-agnostic as they do not optimize the selection of features for a specific prediction method. They are usually based on classical statistics or information theory tools, and the methods we propose belongs to this category. The second group, *wrappers* choose features to optimize the performance of a certain predictor. They usually require the retraining of the predictor at each step of a greedy search, and hence are typically computationally more expensive than filters; furthermore the selected features are tailored to a specific predictor and often do not work well with another type.

In the following we present a filter approach to feature selection in the context of classification tasks based on the maximization of the mutual information between the selected variables and the class. The use of mutual information as a criterion for feature selection has been extensively studied in the literature, and can be easily motivated in the context of classification by Fano's inequality, which lower bounds the probability of incorrect classification P_E by $P_E \geq \frac{H(C)-I(X;Y)-1}{\log|C|}$.

An important issue however that arises in this context, is that of the joint "informativeness" of the selected features. Though wrappers by their very nature tend to select features which are jointly informative, in the case of filters this issue is only partially addressed due to its computational complexity. It is often assumed [17, 8, 11, 15] that selecting features which are jointly informative with the class is too expensive computationally and such methods typically compromise by relying on the mutual information of individual features and the class, as well as the mutual information between pairs of selected features.

We argue however, that rather than compromising on the joint behavior of the selected features, it is preferable to accept a compromise on the density model, which will allow to analyze this joint behavior in an efficient manner.

In the case of continuous features for classification, and if we aim at taking into account the joint behavior of

Appearing in Proceedings of the 17th International Conference on Artificial Intelligence and Statistics (AISTATS) 2014, Reykjavik, Iceland. JMLR: W&CP volume 33. Copyright 2014 by the authors.

features, a Gaussian model is a very natural choice. This unfortunately leads to a technical difficulty: If such a model is used for the conditional distributions of the features given the class, the non-conditioned distribution is a mixture of Gaussians, and its entropy has no simple analytical form. There exists extensive literature on this problem [12], but most of the existing approximations are too computationally intensive in the context at hand, which requires the estimation of the mutual information of a very large number of subsets of features with the class to predict.

Instead we derive two upper-bounds on the Mutual Information – and similarly two bounds on the joint entropy – and maximize these bounds instead of the true information estimate. Also, we propose several algorithmic procedures to update covariance matrices that drastically reduce the computational cost during the optimization of the selected feature set.

2 Related works

It can be easily seen, when selecting d features from a pool of D candidates, that it does not suffice to select features independently informative with respect to the class. It is also important that these features exhibit low redundancy between them: *joint* informativeness is at the core of feature selection.

As mentioned in the introduction, wrappers, due to their very nature, address this issue by creating subsets of features that perform well when combined with a specific predictor. In [10], the authors propose to iteratively train a *SVM*, removing at each iteration the features with the smallest weights. In [7] Adaboost is employed in connection with decision stumps to perform feature selection.

Other wrapper methods impose sparsity on the resulting predictor, thus implicitly performing feature selection. Examples are [3] which uses a Laplacian prior to perform sparse logistic regression, and [19] which casts an l_0 regularized *SVM* as a mixed integer programming problem, while in [1] perform feature selection imposing sparsity via a l_1-norm regularizer. Such wrappers, that train predictors only once, tend to be much faster, however like other wrappers they tend not to generalize well across predictors.

In an approach that share similarities with the algorithm proposed here, forward regression [6] iteratively augments a subset of features to build a linear regressor which is near-optimal in a least-squared error sense.

In the context of filters, the simplest methods are those that calculate statistics on the individual features and then rank these features based on these values, keeping the d features of highest rank. Examples of such statistics are Fisher score, mutual information between the feature and the class (aka information gain), etc. Though quick to compute, such approaches typically result in large redundancy and sub-optimal performance.

The RelieFF algorithm [18] looks at individual features, assigning a score by randomly selecting samples and calculating for that feature and for each sample the difference in distance between the random sample and the nearest sample of the same class, dubbed "nearest hit", and the random sample and the nearest sample of a different class, dubbed "nearest miss". Despite looking at features in isolation, it has been shown to perform well in practice.

In [17] the authors attempt to address the issue of redundancy by adding features to their pool of maximum relevance and minimum redundancy, i.e. features with high mutual information with the class and low mutual information with the features already in the pool d, thus selecting features that are not pairwise redundant.

The *FCBF* [15] algorithm uses symmetrical uncertainty $\frac{I(X;Y)}{H(X)+X(Y)}$ as a quantitative criterion and adds features to the pool based on a novel concept of predominant correlation, namely that the feature is more highly correlated with the class than any of the features already in the pool. *CFS* [11] similarly combines symmetric uncertainty, with Pearson's correlation to add features exhibiting low correlation with the features already in the pool.

Another approach to addressing redundancy uses the concept of a Markov Blanket [16]. The Markov blanket of a variable X is defined as the set of variables S such that X is independent of the remaining variables $D \setminus S \cup X$ given the values of the variables in S. Based on this concept, the authors in [8] select features that have high mutual information with the class when conditioned on one of the features already in the pool. The resulting algorithm is suitable only for binary data, here however we explicitly address the problem of feature selection in a continuous domain.

Perhaps most closely related, at least conceptually, to the work presented here is that of [20] which similarly attempts to find features that are jointly informative by resorting to a Gaussian modeling. In that work however the aim is feature extraction and the mutual information is used as an objective to guide a gradient ascent algorithm.

Finally, we note a family of feature selection algorithms, which have become very popular in recent years, based on spectral clustering. For example [14] selects features based on their influence on the affinity graph Laplacian, and [21] analyzes the spectrum of the Laplacian matrix.

Table 1: Notation

$F = \{X_1, X_2, \dots, X_D\}$	the set of candidate features
X_j	a single feature
Y	the class label
S	a subset of F
S_{n-1}	in Forward Feature Selection, the features selected up until iteration n
Σ_S	the covariance matrix of the features in S
Σ_{jS}	the covariance vector of feature X_j and the features in S
σ^2_{ij}	the covariance of features X_j and X_i
σ^2_i	the variance of feature i
Σ^y_S	the variance of the features in S conditioned on $Y = y$
$\sigma^2_{j\|S}$	the variance of feature X_j conditioned on the value of the features in S

3 Feature selection criteria

3.1 Mutual Information and Gaussian model

The mutual information between two variables is a standard way of measuring the amount of information they share, that is how much of their respective randomness is common.

Given a continuous variable X and a finite variable Y, their mutual information is defined as

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) && (1)\\ &= H(X) - \sum_y H(X|Y=y)P(Y=y). && (2) \end{aligned}$$

Using a Gaussian density model for continuous variables is a natural strategy, due in part to the simplicity of its parametrization, and to its ability to capture the joint behavior of its components. Moreover, the entropy of a n-dimensional multivariate Gaussian $X \sim \mathcal{N}(\mu, \Sigma)$ has a simple and direct expression, namely

$$H(X) = \frac{1}{2}\log(|\Sigma|) + \frac{n}{2}(\log 2\pi + 1). \qquad (3)$$

3.2 Bounds on the Mutual Information and the Entropy

Estimating the mutual information as defined in equation (2) requires to estimate the entropy of both the conditional distributions $X|Y=y$ for any y, and that of X itself. If we model the former with Gaussian distributions, the latter is a mixture of Gaussian distributions, the entropy of which has no simple analytic form.

We propose to mitigate this problem by deriving upper bounds with tractable forms.

Let $f_y, y = 1, \dots, C$ denote Gaussian densities on $\mathbb{R}^D$, p_y a distribution on $\{1, \dots, C\}$, and f^* the Gaussian approximation of the joint law

$$f = \sum_y f_y p_y,$$

that is the Gaussian density of same expectation and covariance matrix as the mixture. Let Y be a random variable of distribution p_y and X a continuous random variable of conditional distribution $\mu_{X|Y=y} = f_y$.

3.2.1 Gaussian compromise bound

As a first bound, we propose to use the entropy of $H(f^*)$ as an approximation of $H(f)$. While combining it with Gaussian models of the conditioned densities is not consistent, estimating the mutual information with it still has all the important properties one desires for continuous feature selection:

- It normalizes with respect to any affine transformation of the features, since such a transformation changes by the same amount all the densities in (2).
- It captures the information content of individual features, since similarly adding a non-informative feature would change by the same amount all the terms of (2).
- It accounts for redundancy, since linearly dependent features would induce a small determinant of the covariance matrix, and a small mutual information.

However, this approximation suffers from a core weakness, namely that the entropy of f^* can become arbitrarily larger than the entropy of $\sum_i p_i f_i$, which leads to degenerated cases where families of features looks "infinitely informative". To mitigate this effect, we propose to upper-bound the approximation of $H(\sum_i p_i f_i)$ with the maximum value it can reach, which corresponds to having the distributions f_i "far apart".

More precisely, we have by definition

$$I(X;Y) = H\left(\sum_y f_y p_y\right) - \sum_y H(f_y) p_y. \qquad (4)$$

Since f^* is a Gaussian density, it has the highest entropy for its variance, and we have $H(f^*) \geq H(\sum_y f_y p_y)$, hence

$$I(X;Y) \leq H(f^*) - \sum_y H(f_y) p_y. \qquad (5)$$

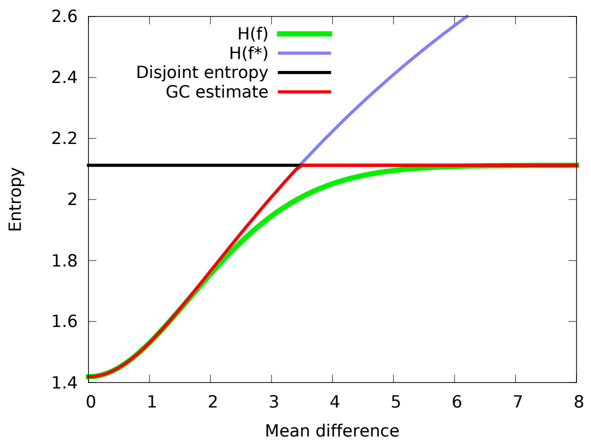

Figure 1: This graph shows several estimates of the entropy of a mixture of two 1D Gaussian densities of variance 1, as a function of the difference between their means. The green curve is the real value of the entropy, estimated numerically. The blue curve is the entropy of the Gaussian fitted on the mixture. The black line stands for the limit entropy when the two components are far apart. The red curve finally is the Gaussian compromise estimate described in § 3.2.1, which is the minimum of the blue and black values.

We know that the mutual information between two variables is upper-bounded by the entropy of each, hence

$$I(X;Y) \leq H(Y) = -\sum_y p_y \log p_y, \tag{6}$$

from which we get

$$I(X;Y) \leq \sum_y (H(f_y) - \log p_y) p_y - \sum_y H(f_y) p_y \tag{7}$$

Taking the min of inequalities (5) and (7), we get our "Gaussian Compromise" bound

$$\begin{aligned} I(X;Y) \quad \leq \quad & \sum_y \min(H(f^*), H(f_y) - \log p_y) p_y \\ & - \sum_y H(f_y) p_y. \end{aligned} \tag{8}$$

From this bound and Equation (2), we can derive an upper bound on $H(X)$ as well. Figure 1 illustrates the behavior of this bound in the case of two 1D Gaussians.

3.2.2 KL-based bound

We derive here a more general bound in the case of a distribution f which is a mixture of two distribution $f = p_1 f_1 + p_2 f_2$. In this case we have:

$$\begin{aligned} H(f) = -\int_{-\infty}^{\infty} & p_1 f_1(u) \log(p_1 f_1(u) + p_2 f_2(u)) \\ & + p_2 f_2(u) \log(p_1 f_1(u) + p_2 f_2(u)) du \end{aligned}$$

Working with the first term in the above integral we have:

$$\begin{aligned} & -\int_{-\infty}^{\infty} p_1 f_1(u) \log(p_1 f_1(u) + p_2 f_2(u)) du \\ = & -\int_{-\infty}^{\infty} p_1 f_1(u) \log\left(1 + \frac{p_2 f_2(u)}{p_1 f_1(u)}\right) du \\ & -\int_{-\infty}^{\infty} p_1 f_1(u) log(p_1 f_1(u)) du \\ = c & -\int_{-\infty}^{\infty} p_1 f_1(u) \log\left(1 + \frac{p_2 f_2(u)}{p_1 f_1(u)}\right) du + p_1 H(f_1(u)) \\ \leq c & -\int_{-\infty}^{\infty} p_1 f_1(u) \log\left(\frac{p_2 f_2(u)}{p_1 f_1(u)}\right) du + p_1 H(f_1(u)) \\ & = p_1 D_{KL}(f_1(u) \parallel f_2(u)) + p_1 H(f_1(u)) + c', \end{aligned}$$

where c, c' are constants related to the mixture coefficients and the inequality comes from the fact that $\log(1+x) \geq \log(x)$. Similarly for the second term we have:

$$\begin{aligned} & -\int_{-\infty}^{\infty} p_2 f_2(u) \log(p_1 f_1(u) + p_2 f_2(u)) du \\ & \leq p_2 D_{KL}(f_2(u) \parallel f_1(u)) + p_2 H(f_2(u)) + c'' \end{aligned}$$

Based on this we have:

$$\begin{aligned} H(f) \leq & p_2 D_{KL}(f_2(u) \parallel f_1(u)) + p_2 H(f_2(u)) \\ & + p_1 D_{KL}(f_1(u) \parallel f_2(u)) + p_1 H(f_1(u)) + c''', \end{aligned}$$

and by extension:

$$\begin{aligned} I(X;Y) \leq & p_2 D_{KL}(f_2(u) \parallel f_1(u)) \\ & + p_1 D_{KL}(f_1(u) \parallel f_2(u)) + c'''. \end{aligned}$$

In the case where $f_1 = N(\mu_1, \Sigma_1)$ and $f_2 = N(\mu_2, \Sigma_2)$ are both multivariate Gaussian distributions of dimensionality D, we have that:

$$\begin{aligned} D_{KL}(f_1 \parallel f_2) = & \frac{1}{2}\left(\mathrm{Tr}\left({\Sigma_2}^{-1}\Sigma_1\right) - \ln\frac{|\Sigma_1|}{|\Sigma_2|} - D\right) \\ & + \frac{1}{2}(\mu_2 - \mu_1)^T {\Sigma_2}^{-1} (\mu_2 - \mu_1) \end{aligned}$$

In the case of a binary classification problem, we can directly work with the above quantity for our mixture of two Gaussians. In the case where $|Y| > 2$, we consider the resulting $|Y|$ one-against-all binary classification problems and attempt to maximize the average of the upper bounds of the $|Y|$ mutual information values.

More specifically, for each class y we consider the following mixture model:

$$f = p_y f_y + (1 - p_y) f_{Y \setminus y}$$

where the $f_{Y \setminus y}$ is the conditional distribution of $X|Y \neq y$. We then calculate the upper bound of the mutual information for all the possible mixtures f, one for each y.

4 Computational Implementation

Computing the optimal set of features S of size N under either of the derived upper bound criteria, is computationally intractable as there are $\frac{F!}{N!(F-N)!}$ candidate sets. Instead we settle for a greedy algorithm which attempts to approximate the optimal set S by adding one feature at a time.

Specifically, in order to find a good approximation of the optimization of the upper bound criteria presented in § 3, we make use of forward selection, a standard greedy optimization method. It iteratively builds a sequence of sets S_n with $n = 1, \dots, N$ of increasing size, each set S_n built by adding one feature $X_{j(n)}$ to the previous one S_{n-1}. Thus at a given iteration n, the greedy forward selection algorithm calculates for every candidate feature $X_j \in F \setminus S_{n-1}$, the mutual information, or entropy accordingly, between the set $S'_n = S_{n-1} \cup \{X_j\}$ and the label Y. It then creates the set S_n by adding that feature which leads to the largest value of the optimization criterion.

4.1 Complexity of the Gaussian Compromise

Though forward selection leads to a computationally tractable feature selection algorithm, it remains nonetheless very expensive.

In the case of the Gaussian compromise approach, at each iteration n and for each feature X_j not in S_{n-1}, forward selection requires the estimation of $I(S_{n-1} \cup \{X_j\}; Y)$, which in turn requires the estimation of $|Y|+1$ determinants of size $n \times n$. A naive approach would be to calculate these determinants from scratch, incurring a cubic cost of $O(n^3)$ for the calculation of each determinant and $O(|Y||F \setminus S_{n-1}|n^3)$ per iteration.

4.1.1 Quadratic in n

In order to speed up this process, we note that $\Sigma_{S'_n}$ differs from $\Sigma_{S_{n-1}}$ by the addition of a row and a column

$$\Sigma_{S'_n} = \begin{bmatrix} \Sigma_{S_{n-1}} & \Sigma_{jS_{n-1}} \\ \Sigma^T_{jS_{n-1}} & 1 \end{bmatrix} \tag{9}$$

Where $\Sigma_{jS_{n-1}}$ is the vector of covariances between the features in S_{n-1} and X_j. Thus $\Sigma_{S'_n}$ is the result of two rank-one updates to the augmented matrix

$$\begin{bmatrix} \Sigma_{S_{n-1}} & 0_{n-1} \\ 0^T_{n-1} & \sigma^2_j \end{bmatrix}, \tag{10}$$

namely one rank-one update corresponding to changing the last row, and one rank-one update corresponding to changing to last column.

Consequently we can efficiently calculate the determinant $|(\Sigma_{S'_n})|$ by applying, twice, the matrix determinant lemma.

The cost of this calculation is obviously $O(n^2)$. Thus we can efficiently add one feature to a pool of $|S| = n$ features previously selected, by incurring a cost of $O(|Y||F \setminus S_{n-1}|n^2)$.

4.1.2 Linear in n

We can further speed-up the proposed algorithm by a factor of n, if we are willing to incur a slightly higher memory load. We first note that $I(X, Z; Y) = I(Z; Y) + I(X; Y|Z)$, which in the context of our forward selection algorithm translates to

$$I(S'_n; Y) = I(S_{n-1}; Y) + I(X_j; Y|S_{n-1}). \tag{11}$$

Of course the first term in the above expression is common for all candidate features X_j, meaning that finding the feature X_j that maximizes $I(S'_n; Y)$ is equivalent to finding the feature that maximizes $I(X_j; Y|S_{n-1})$. If $\sigma^2_{j|S_{n-1}}$ denotes the variance of feature j given the features in S_{n-1} and $\sigma^{y^2}_{j|S_{n-1}}$ the variance conditioned on the features in S_{n-1} and the class $Y = y$, we have

$$\underset{X \in F \setminus S_{n-1}}{\operatorname{argmax}} \; I(X_j; Y|S_{n-1}) = \underset{X \in F \setminus S_{n-1}}{\operatorname{argmax}} \; \log(\sigma^2_{j|S_{n-1}}) - \sum_y P(Y = y) \log(\sigma^{y^2}_{j|S_{n-1}})$$

where we have exploited the fact that the conditional variance $\sigma_{j|S_{n-1}}$ is independent of the specific values of the features in S_{n-1} and thus the integrations of the entropies over the conditioned values is straightforward. That is

$$\int_{\mathbb{R}^{|S_{n-1}|}} H(X_j|S_{n-1} = s)\mu_{S_{n-1}}(s)ds = \frac{1}{2}\log((\sigma^2_{j|S_{n-1}})) + \frac{1}{2}(\log 2\pi + 1)$$

Under the Gaussian assumption, we have

$$\sigma^2_{X_j|S_{n-1}} = \sigma^2_{X_j} - \Sigma^T_{jS_{n-1}} \Sigma^{-1}_{S_{n-1}} \Sigma_{jS_{n-1}}. \tag{12}$$

Calculating the above value for a candidate feature X_j incurs a cost of $O(n^2)$ resulting in an algorithm of similar complexity as the one presented in the previous section. However, as mentioned, we can improve on this complexity with an increase in memory requirements. To be specific, we consider the case of calculating $\sigma^2_{j|S_{n-1}}$ where $S_{n-1} = S_{n-2} \cup X_i$. We have

$$\Sigma^T_{jS_{n-1}} \Sigma^{-1}_{S_{n-1}} \Sigma_{jS_{n-1}} = \begin{bmatrix} \Sigma^T_{jS_{n-2}} \sigma^2_{ji} \end{bmatrix} \Sigma^{-1}_{S_{n-1}} \begin{bmatrix} \Sigma_{jS_{n-2}} \\ \sigma^2_{ij} \end{bmatrix} \tag{13}$$

By applying the Sherman-Morrison formulas twice to update $\Sigma^{-1}_{S_{n-2}}$ to $\Sigma^{-1}_{S_{n-1}}$, we obtain an update formula of the form

$$\Sigma^{-1}_{S_{n-1}} = \left[\begin{array}{cc} \Sigma^{-1}_{S_{n-2}} & w \\ z^T & c \end{array} \right] + uv^T \qquad (14)$$

where the vectors u, v, w, z, and the scalar c are computed prior to each iteration n with a cost of $O(n^2)$. From 14 and 13 we have

$$\begin{aligned} \sigma^2_{j|S_{n-1}} &= \sigma^2_j - \Sigma^T_{jS_{n-1}} \left(\left[\begin{array}{cc} \Sigma^{-1}_{S_{n-2}} & w \\ z^T & c \end{array} \right] + uv^T \right) \Sigma_{jS_{n-1}} \\ &= \sigma^2_j - \left(\Sigma^T_{jS_{n-2}} \Sigma^{-1}_{S_{n-2}} + \sigma^2_{ji} z^T \right) \Sigma_{jS_{n-2}} \\ &\quad - \Sigma^T_{jS_{n-1}} \left[\begin{array}{c} w \\ c \end{array} \right] \sigma^2_{ji} \\ &\quad - \left(\Sigma^T_{jS_{n-1}} u \right) \left(v^T \Sigma_{jS_{n-1}} \right) \qquad (15) \end{aligned}$$

In equation (15), the third term is simply an inner product and can be calculated in $O(n)$, while the fourth term can be also calculated using inner products, similarly in $O(n)$. The main computational cost is incurred in calculating the term $\Sigma^T_{jS_{n-2}} \Sigma^{-1}_{S_{n-2}} \Sigma_{jS_{n-2}}$, that is in computing the vector $\Sigma^T_{jS_{n-2}} \Sigma^{-1}_{S_{n-2}}$. Given this vector, the second term is simply an inner product and can also be calculated in $O(n)$. The size $n-2$ vector $\Sigma^T_{jS_{n-2}} \Sigma^{-1}_{S_{n-2}}$ however has already been calculated $\forall X_j \in F \setminus S_{n-2}$ when calculating their scores during the previous iteration $n-1$. Thus if are willing to accept memory requirements by $O((F-n)n)$ we can store this vector for each candidate feature and simply incur a cost of $O(|Y||F \setminus S_{n-1}|n)$ per iteration.

4.2 Complexity of the KL-based Algorithms

In the case of the KL-based algorithms, similarly with above, a naive implementation would incur a cost of $O(|Y||F \setminus S_{n-1}|n^3)$.

Working with the upper bound value:

$$\begin{aligned} & p_2 D_{KL}(f_2(u) \parallel f_1(u)) + p_2 H(f_2(u)) \\ & + p_1 D_{KL}(f_1(u) \parallel f_2(u)) + p_1 H(f_1(u)) + c''' \end{aligned}$$

we see that the entropy values $H(f_y(u))$ can be computed efficiently as in the previous subsection. What remains is to efficiently compute the Kullback-Leibler divergences for each of the $|Y|$ binary classification problems. That is to say $\forall y, Y \setminus y$ the value:

$$\frac{1}{2} \left(\mathrm{Tr}\left({\Sigma_2}^{-1} \Sigma_1 \right) - \ln \frac{|\Sigma_1|}{|\Sigma_2|} + (\mu_2 - \mu_1)^T {\Sigma_2}^{-1} (\mu_2 - \mu_1) \right) \qquad (16)$$

where $\Sigma_1 = \Sigma^y_{S'_n}, \Sigma_2 = \Sigma^{Y \setminus y}_{S'_n}$ and $\mu_1 = \mu^y_{S'_n}, \mu_2 = \mu^{Y \setminus y}_{S'_n}$.

From equation 11, it can be seen that $|\Sigma_{S'_n}| = \sigma^2_{j|S_{n-1}} |\Sigma_{S_{n-1}}|$ and thus the second term above can be computed efficiently in time $O(n)$. For the first and third terms however we need to calculate the matrix ${\Sigma_2}^{-1}$ which incurs a cost of $O(n^2)$ (using Sherman-Morrison). We note that it is possible, by storing values, to efficiently compute the term $(\mu_2 - \mu_1)^T {\Sigma_2}^{-1} (\mu_2 - \mu_1)$ in $O(n)$ once ${\Sigma_2}^{-1}$ is calculated. Finally the first term, i.e. the trace, also incurs a cost of $O(n^2)$ as we only need to compute the values of the product matrix along the main diagonal. Thus the overall complexity of the KL-based approaches is $O(|Y||F \setminus S_{n-1}|n^2)$, i.e. quadratic in n.

Unfortunately in this case it is not possible to exploit equation 11 to obtain a $O(n)$ algorithms as this requires integrating over the conditional mean $\mu_{j|S_{n-1}}$ whose value, unlike the conditional variance, depends on the value of the conditioned variables in S_{n-1}.

5 Experiments and discussion

5.1 Data-sets

In order to experimentally demonstrate the merits of the various proposed information theoretic feature selection algorithms, we conducted experiments on three popular computer vision datasets.

CIFAR-10 contains images of size 32×32, of 10 distinct classes depicting vehicles and animals. The training data consists of 5,000 images of each class. We pre-process the data as in [4] using code provided by the authors. The original pool F of features consists of 2,048 candidates.

INRIA is a pedestrian detection dataset. There are 12,180 training images of size 64×128 of pedestrians and background images. We use 3,780 HoG features that have been shown to perform well in practice [5].

STL-10 consists of images of size 96×96 belonging to 10 classes, each represented by 500 training images. As for CIFAR we pre-process the data as in [4], resulting in a pool F of 4,096 features.

5.2 Baselines

We compare the proposed feature selection method against a number of baselines. The **Fisher**, **T-test**,χ^2, and **InfoGain** methods all compute statistics on individual features. In particular **InfoGain** calculates the mutual information of the individual features to the class, without taking into account there joint informativeness. As such, its comparison with our approaches is a very good indicator of the merit of joint informativeness and its effect on classification performance.

	FCBF	MRMR	SBMLR	Spec. Clus.	CFS	CMTF	Relieff	**GC.E**	**GC.MI**	**GKL.E**	**GKL.MI**
CIFAR	621	901	1449	1379	4262	394	1652	79	**20**	486	483
STL	68	207	1002	367	409	208	2089	23	**5**	887	856
INRIA	247	579	88	1072	2516	459	2413	**43**	**43**	135	131

Table 2: Cost in cputime of running the more sophisticated feature selection algorithms in order to select 100 features on the three datasets. We highlight in bold the fastest algorithm for each dataset.

As noted in section 2, the **FCBF** [15] and **CFS** [11] baselines employ symmetric uncertainty criterion and check for pairwise redundancy of features. Similarly **MRMR** [17], uses mutual information to select features that have high relevance to the class while having low mutual information with the other selected features, thus checking for pairwise informativeness. Again comparison with the proposed methods proves the importance of going beyond pairwise redundancy.

The **RelieFF** [18] baseline looks at the nearest neighbors of random samples along the individual features. In order to compare with spectral clustering approaches we show results for [21], marked as **Spec. Clus.** in the tables, which we found to outperform [22] in practice. Finally, we also show results for two wrapper method, namely **SBMLR** [3], which employs a logistic regression predictor, and **CMTF** [1] which uses a sparsity inducing l_1-norm.

We compare against the four methods proposed here, namely maximizing the entropy (**GC.E**) or the mutual information (**GC.MI**) under the Gaussian compromise bound, and maximizing the KL-based entropy (**GKL.E**) and mutual information (**GKL.MI**). In the case of the **GC** methods, when an iteration is reached where all candidate features attain the upper bound, we halt feature selection and randomly select the remaining features.

5.3 Results

In tables 3, 4, and 5, we show experimental results for the three datasets. In order to show the general applicability of the proposed methods, we combined the selected features with four different classifiers: AdaBoost with classification stumps, linear SVM, RBF kernel SVM, and quadratic discriminant analysis (QDA). We show results for several numbers of selected features $\{10, 25, 50, 100\}$.

In each of these tables, for each dataset and classifier we highlight the best three performing methods in bold, while underlining the best performing method. As can be seen from these tables, **GC.MI** and **GKL.E** consistently rank amongst the top three methods, with **GC.MI** ranking in the top three 35 out of 48 times and first 10 times, and **GKL.E** 33 out of 48 times and first 8 times. The only other comparable method is the wrapper method **SBMLR** ranks in the top three 22 out of 48 times and first 14 times.

Furthermore as can be seen in table 2, the running time of the proposed methods is very competitive with respect to the more complex of the remaining feature selection algorithms. The Gaussian Compromise algorithms are especially fast as they are two orders of magnitude faster then practically all other methods. We especially note that the **SBMLR** method which performs comparably in terms of accuracy is very slow when compared to **GC.MI**.

The computation times provided were obtained with C++ implementations of the proposed methods. The **MRMR** is also implemented in C++, while the **Spect. Clust.** and **CMTF** baselines are implemented in Matlab, as both these algorithms mainly use matrix algebra we believe these timings to be indicative. The remaining algorithms were implemented in Java, as noted in [2] these implementations are competitive in speed with C++ implementations.

6 Conclusion

We have presented a family of novel information theoretic algorithms for the selection of continuous features in the context of classification. They rely on modeling the conditional joint distributions of the features given the class to predict and maximizing upper bounds on the information theoretic measures we seek to maximize. These models exhibit the main properties we expect for characterizing "good" groups of continuous features, and are shown experimentally to be competitive with current state-of-the-art methods.

To reduce the computational cost of a forward-selection implementation of these criteria, we have proposed efficient implementations for both approaches, so that they are competitive with other state-of-the-art methods. We have shown that with the appropriate modeling and with careful implementation, it is in fact possible to develop algorithms, which are able to exploit joint feature informativeness in practice.

Acknowledgements

This work was supported by the Hasler Foundation (www.haslerstiftung.ch) through the MASH2 project.

	AdaBoost				SVMLinear				SVMRBF				QDA			
Method	10	25	50	100	10	25	50	100	10	25	50	100	10	25	50	100
Fisher	86.90	89.83	90.38	91.45	92.55	**93.73**	94.03	94.68	92.44	93.55	93.38	92.97	87.41	88.63	89.17	91.31
FCBF	90.87	**94.02**	**95.44**	94.67	**94.14**	**96.03**	**96.03**	96.03	88.29	**93.91**	92.60	**95.66**	**89.95**	**94.00**	**94.00**	94.00
MRMR	87.42	87.30	87.57	87.50	86.03	86.08	86.77	86.72	82.32	80.01	80.01	80.01	79.23	80.02	81.10	81.47
χ^2	**92.81**	93.11	93.94	94.91	**92.94**	93.27	93.50	94.61	**92.78**	93.16	93.02	93.25	87.85	88.20	89.30	91.75
SBMLR	86.40	87.50	88.04	88.06	85.92	87.95	88.57	88.64	82.82	86.05	87.39	87.14	76.30	80.36	81.00	81.49
tTest	85.01	88.41	88.84	91.70	80.01	87.21	87.64	89.23	80.01	87.00	87.11	87.32	76.16	82.50	82.85	85.23
InfoGain	92.58	93.29	93.96	94.93	92.35	93.08	93.75	94.68	92.28	92.71	93.01	93.38	**87.99**	88.08	89.49	91.77
Spec. Clus.	**92.78**	93.69	93.92	94.83	**92.67**	93.57	93.64	94.44	**92.67**	93.09	92.85	93.29	**87.99**	88.26	89.07	91.26
RelieFF	91.79	**95.44**	95.83	96.43	90.99	**95.04**	**95.97**	**96.36**	90.62	**94.56**	**95.05**	95.20	83.38	**92.14**	93.16	94.24
CFS	89.69	92.60	**96.41**	**97.69**	88.64	91.68	96.11	**97.53**	88.34	91.31	**95.44**	**97.14**	83.79	88.31	**94.00**	**96.66**
CMTF	80.01	83.72	92.55	95.58	79.09	80.29	89.49	93.01	80.01	83.72	92.55	93.68	61.04	72.31	89.23	92.67
GC.E	89.54	90.09	94.30	95.81	87.73	87.67	91.96	93.13	87.73	87.67	91.96	93.13	85.06	86.31	91.22	94.10
GC.MI	**95.04**	**95.87**	**96.68**	**97.30**	89.76	93.09	**95.71**	**96.45**	**94.26**	**94.17**	**94.44**	**95.76**	**92.14**	**94.08**	**95.07**	**96.31**
GKL.E	89.92	91.84	94.14	**96.63**	85.31	89.46	92.05	**96.36**	86.01	88.94	92.79	95.43	79.30	86.21	90.83	**95.74**
GKL.MI	92.18	93.09	95.21	96.15	85.66	90.99	92.14	95.16	91.03	91.91	93.36	93.98	85.09	88.03	91.72	94.84

Table 3: Test accuracy on the **INRIA** dataset for a different number of selected features $\{10, 25, 50, 100\}$ when combined with the four different classifiers

	AdaBoost				SVMLinear				SVMRBF				QDA			
Method	10	25	50	100	10	25	50	100	10	25	50	100	10	25	50	100
Fisher	29.23	36.96	42.07	49.06	25.19	33.53	39.47	48.12	29.11	39.22	46.05	54.68	25.41	33.31	39.67	47.53
FCBF	**37.77**	44.42	51.15	54.83	**33.65**	42.02	47.77	54.97	**40.48**	**51.15**	57.73	64.26	**35.02**	**43.97**	52.32	**58.99**
MRMR	30.86	35.42	37.56	40.85	27.94	33.79	37.78	43.63	32.75	38.77	40.63	46.75	29.13	34.07	36.36	42.39
χ^2	28.13	35.54	43.68	49.46	21.77	32.06	40.65	48.58	27.16	38.23	47.60	54.70	21.81	31.85	39.39	47.75
SBMLR	34.87	**45.08**	**52.17**	**56.70**	30.43	**42.60**	**51.41**	**56.81**	36.06	49.83	**60.32**	**64.97**	31.71	43.46	**53.31**	**58.86**
tTest	25.74	31.30	36.57	43.16	25.69	32.56	40.17	45.12	28.68	35.75	41.89	49.13	26.34	33.39	39.16	45.33
InfoGain	29.01	35.90	40.20	48.34	24.79	32.32	37.98	47.37	29.21	38.68	43.92	53.94	22.38	31.61	37.65	46.47
Spec. Clus.	19.90	25.13	33.18	40.44	17.19	23.14	32.78	42.62	22.89	30.92	40.41	49.75	17.97	24.80	34.99	44.25
RelieFF	28.13	34.64	40.85	47.70	24.56	30.60	38.17	46.51	29.49	37.08	45.39	53.96	24.61	29.67	38.20	47.18
CFS	33.50	38.96	44.58	54.22	31.49	36.46	42.17	51.70	35.50	43.74	50.98	61.01	31.50	36.18	42.93	52.92
CMTF	21.79	31.98	39.43	45.23	21.10	31.64	40.39	47.71	23.9	36.74	45.51	52.86	20.61	31.98	41.04	48.60
GC.E	32.45	42.54	50.15	55.06	28.76	41.14	48.70	55.16	35.29	**51.12**	**60.34**	**65.76**	31.01	**44.10**	**53.21**	58.41
GC. MI	**36.47**	**44.55**	**51.44**	**55.39**	**34.02**	**42.14**	**49.16**	**55.07**	**39.57**	49.91	57.79	64.32	**33.68**	43.02	51.84	**58.43**
GKL.E	**37.51**	**46.41**	**52.11**	**56.41**	**32.39**	**43.26**	**50.12**	**55.02**	**39.84**	**52.80**	**60.94**	**65.64**	**34.06**	**44.21**	**53.29**	57.98
GKL. MI	33.71	40.04	47.17	51.12	28.67	34.65	43.30	48.69	34.49	43.09	51.48	56.54	29.06	35.10	42.50	46.24

Table 4: Test accuracy on the **CIFAR** dataset for a different number of selected features $\{10, 25, 50, 100\}$ when combined with the four different classifiers

	AdaBoost				SVMLinear				SVMRBF				QDA			
Method	10	25	50	100	10	25	50	100	10	25	50	100	10	25	50	100
Fisher	31.86	35.78	39.72	41.81	26.09	30.79	34.63	38.02	34.71	40.13	43.87	45.77	34.73	39.91	44.24	48.35
FCBF	33.25	38.05	39.87	42.81	31.74	34.85	38.11	40.66	**38.86**	43.35	46.06	47.20	**37.44**	41.89	45.70	48.89
MRMR	31.57	34.34	36.55	37.49	28.26	29.73	31.16	33.12	34.20	37.28	38.32	38.87	33.94	37.30	41.35	43.33
χ^2	29.61	36.88	39.39	41.89	22.61	31.82	34.29	37.96	32.53	41.27	43.22	44.88	32.71	40.45	43.64	47.25
SBMLR	**34.22**	**41.26**	**44.65**	**47.15**	**32.29**	38.26	43.29	47.15	**39.97**	**47.21**	**51.05**	**53.58**	36.89	**45.26**	**49.58**	**51.65**
tTest	31.74	34.75	39.31	42.34	26.72	29.95	36.23	39.14	34.30	38.73	44.30	45.90	33.92	40.09	45.05	47.63
InfoGain	31.13	36.60	38.62	42.03	27.17	31.82	33.70	37.84	35.57	41.23	42.92	45.12	34.17	40.94	44.51	47.81
Spec. Clus.	19.06	26.30	33.52	38.51	18.91	26.55	32.65	38.24	24.80	32.91	40.11	43.70	25.39	35.45	44.39	49.68
RelieFF	33.91	37.46	42.79	45.22	29.16	32.40	38.05	42.94	38.22	42.36	47.27	50.35	**37.57**	42.59	**47.60**	50.92
CFS	30.75	38.40	41.85	44.39	28.63	34.45	38.54	41.88	35.32	42.72	47.46	49.82	34.06	42.29	**48.45**	**51.09**
CMTF	28.70	33.55	34.71	36.86	27.61	34.81	38.99	42.32	31.80	36.94	38.06	39.65	29.39	36.26	40.32	43.44
GC.E	31.86	37.41	42.19	**46.99**	31.20	37.60	43.31	**49.75**	36.16	42.64	45.37	47.79	33.76	**43.04**	47.51	51.02
GC.MI	**36.50**	**40.79**	**43.82**	44.39	**32.50**	**39.75**	**44.15**	**48.88**	37.25	**45.51**	**49.58**	**52.36**	35.31	42.24	47.21	49.88
GKL.E	**34.76**	**39.71**	**43.49**	**46.46**	**33.44**	**38.62**	**44.27**	**50.54**	**39.67**	**46.31**	**50.06**	**52.89**	**37.39**	**43.30**	47.39	**51.82**
GKL.MI	33.00	38.80	42.13	43.58	32.16	**39.35**	**44.87**	47.96	35.95	41.65	45.27	45.86	33.56	40.02	45.07	46.44

Table 5: Test accuracy on the **STL** dataset for a different number of selected features $\{10, 25, 50, 100\}$ when combined with the four different classifiers

References

[1] Andreas Argyriou, Theodoros Evgeniou, and Massimiliano Pontil. Convex multi-task feature learning. *Machine Learning*, 73(3):243–272, 2008.

[2] Remco R. Bouckaert, Eibe Frank, Mark A. Hall, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. Weka—experiences with a java open-source project. *Journal Machine Learning Research (JMLR)*, 11:2533–2541, 2010.

[3] Gavin C. Cawley, Nicola L. C. Talbot, and Mark Girolami. Sparse multinomial logistic regression via bayesian l1 regularisation. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 209–216, 2006.

[4] A. Coates and A. Ng. The importance of encoding versus training with sparse coding and vector quantization. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 921–928, 2011.

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *Proceedings of Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893, 2005.

[6] A. Das and D. Kempe. Submodular meets spectral: Greedy algorithms for subset selection, sparse approximation and dictionary selection. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1057–1064, 2011.

[7] S. Das. Filters, wrappers and a boosting-based hybrid for feature selection. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 74–81, 2001.

[8] F. Fleuret. Fast binary feature selection with conditional mutual information. *Journal of Machine Learning Research (JMLR)*, 5:1531–1555, 2004.

[9] Isabelle Guyon and André Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research (JMLR)*, 3:1157–1182, 2003.

[10] Isabelle Guyon, Jason Weston, Stephen Barnhill, and Vladimir Vapnik. Gene selection for cancer classification using support vector machines. *Machine Learning*, 46(1-3):389–422, March 2002.

[11] Mark A. Hall and Lloyd A. Smith. Feature selection for machine learning: Comparing a correlation-based filter approach to the wrapper. In *Proceedings of the International Florida Artificial Intelligence Research Society Conference*, pages 235–239, 1999.

[12] J.R. Hershey and P.A. Olsen. Approximating the kullback leibler divergence between gaussian mixture models. In *Proceedings of ICASSP*, volume 4, pages IV–317–IV–320, 2007.

[13] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[14] Yi Jiang and Jiangtao Ren. Eigenvector sensitive feature selection for spectral clustering. In *Proceedings of European Conference on Machine Learning (ECML)*, pages 114–129, 2011.

[15] H. Liu and L. Yu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 856–863, 2003.

[16] Dimitris Margaritis. Toward provably correct feature selection in arbitrary domains. In *Proceedings of Neural Information Processing Systems (NIPS)*, pages 1240–1248, 2009.

[17] F. Ding C. Peng, H. Long. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 27(8):1226–1238, 2005.

[18] Marko Robnik-Šikonja and Igor Kononenko. Theoretical and empirical analysis of relieff and rrelieff. *Machine Learning*, 53(1-2):23–69, 2003.

[19] Mingkui Tan, Li Wang, and Ivor W. Tsang. Learning sparse svm for feature selection on very high dimensional datasets. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1047–1054, 2010.

[20] Kari Torkkola. Feature extraction by non-parametric mutual information maximization. *Journal of Machine Learning Research (JMLR)*, 3:1415–1438, 2003.

[21] Lior Wolf and Amnon Shashua. Feature selection for unsupervised and supervised inference: The emergence of sparsity in a weight-based approach. *Journal of Machine Learning Research (JMLR)*, 6:1855–1887, 2005.

[22] Zheng Zhao and Huan Liu. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of International Conference on Machine Learning (ICML)*, pages 1151–1157, 2007.