Value of Information Based on Decision Robustness

Suming Chen and **Arthur Choi** and **Adnan Darwiche**
Computer Science Department
University of California, Los Angeles
{suming,aychoi,darwiche}@cs.ucla.edu

Abstract

There are many criteria for measuring the *value of information (VOI)*, each based on a different principle that is usually suitable for specific applications. We propose a new criterion for measuring the value of information, which values information that leads to robust decisions (i.e., ones that are unlikely to change due to new information). We also introduce an algorithm for Naive Bayes networks that selects features with maximal VOI under the new criterion. We discuss the application of the new criterion to classification tasks, showing how it can be used to tradeoff the budget, allotted for acquiring information, with the classification accuracy. In particular, we show empirically that the new criterion can reduce the expended budget significantly while reducing the classification accuracy only slightly. We also show empirically that the new criterion leads to decisions that are much more robust than those based on traditional VOI criteria, such as information gain and classification loss. This make the new criterion particularly suitable for certain decision making applications.

Introduction

Consider a probabilistic graphical model with some features $F_1, \ldots, F_n$ that can be observed at a cost. Consider also a decision variable D in the model, whose posterior distribution will be used to make a decision. Given a budget on observations, a classical problem is that of finding a set of features $\mathbf{G}$ from $F_1, \ldots, F_n$, where the *cost* of observing $\mathbf{G}$ is within budget, and the *value* obtained from observing $\mathbf{G}$ is maximal. This problem appears in a variety of applications, including active sensing (Greiner, Grove, and Roth 2002; Gao and Koller 2011), medical diagnosis (Yu et al. 2009), fault identification (Bellala et al. 2013), knowledge assessment (Munie and Shoham 2008; Millán et al. 2013), and computer vision (Ahmad and Yu 2013).

There are different criteria for measuring the VOI, each based on a different principle that is suitable for specific applications. Most of these criteria can be formulated in terms of the *expected reward* of observing features $\mathbf{G}$ (Krause and Guestrin 2009). This includes, for example, the popular criterion based on information gain; see (Lu and Przytula 2006; Zhang and Ji 2010; Yu et al. 2009; Gao and Koller 2011). Here, the reward function is entropy and the goal is to select features that reduce the expected entropy of the decision variable D. Another criterion is the one based on classification loss (Gao and Koller 2011), which is used in the context of threshold-based classification. Some additional criteria are proposed in (Krause and Guestrin 2009), which are also formulated in terms of expected reward.

In this paper, we propose a new criterion for VOI in the context of threshold-based decisions. The proposed criterion is based on a new principle, *decision robustness,* which values features that, when observed, would lead to a decision that is maximally robust against further observations.

Our proposed VOI criterion can also be formulated in terms of expected reward, where the reward function is a recently proposed probabilistic query, called the *Same-Decision Probability* (SDP) (Darwiche and Choi 2010). The SDP quantifies the stability of a threshold-based decision and is defined as the probability that a current decision would stay the same, had we observed further information.

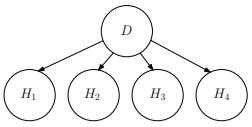
Our second contribution is an algorithm, `MaxDR`, that optimally selects features using the new criterion, given a budget. We restrict our attention to Naive Bayes networks, as computing the SDP is PP^{PP}-complete for Bayesian networks (Choi, Xue, and Darwiche 2012) but only NP-hard for Naive Bayes networks (Chen, Choi, and Darwiche 2013).[1] The proposed algorithm, `MaxDR`, is based on a combination of techniques used to solve the knapsack problem as well as a branch-and-bound search algorithm.

Our third contribution is in exploring the application of `MaxDR` to classification and decision making tasks. For the former, we show that `MaxDR` can be used to tradeoff the ex-

[1] First, observe the following relationship between complexity classes:

$$\text{NP} \subseteq \text{PP} \subseteq \text{NP}^{\text{PP}} \subseteq \text{PP}^{\text{PP}}$$

Note that MPE is the prototypical NP-complete problem for probabilistic inference (Shimony 1994). MAP (Park and Darwiche 2004) is the prototypical NP^{PP}-complete problem. SDP was shown to be PP^{PP}-complete, where PP^{PP} can be thought of as a counting variant of the class NP^{PP}, hence highly intractable. Naive Bayes networks are typically tractable for basic queries, yet remain NP-hard for MAP (De Campos 2011), and PP-complete for computing the value of information (Krause and Guestrin 2009). SDP has also been shown to be an NP-hard problem in Naive Bayes (Chen, Choi, and Darwiche 2013).



D	$Pr(D)$
$+$	0.50
$-$	0.50

Figure 1: A Naive Bayes network.

D	H_1	$Pr(H_1 \mid D)$
$+$	$+$	0.508
$+$	$-$	0.482
$-$	$+$	0.016
$-$	$-$	0.984

D	H_3	$Pr(H_3 \mid D)$
$+$	$+$	0.61
$+$	$-$	0.39
$-$	$+$	0.39
$-$	$-$	0.61

D	H_2	$Pr(H_2 \mid D)$
$+$	$+$	0.80
$+$	$-$	0.20
$-$	$+$	0.20
$-$	$-$	0.80

D	H_4	$Pr(H_4 \mid D)$
$+$	$+$	0.543
$+$	$-$	0.456
$-$	$+$	0.456
$-$	$-$	0.543

Figure 2: Additional CPTs for network in Figure 1.

pended budget with classification accuracy (MaxDR can reduce the expended budget significantly, while reducing the classification accuracy only slightly). For decision making tasks, we show that MaxDR leads to much more robust decisions, under a limited budget, compared to some traditional VOI criteria. This makes MaxDR particularly suitable to applications where the decision maker wishes to reduce the liability of their decisions against unknown information.

This paper is structured as follows. We start by providing some background, and follow by introducing the new VOI criterion. The corresponding algorithm, MaxDR, is then presented along with its applications and empirical results.

Background

We use standard notation for variables and their instantiations, where variables are denoted by upper case letters (e.g. X) and their instantiations by lower case letters (e.g. x). Sets of variables are then denoted by bold upper case letters (e.g. $\mathbf{X}$) and their instantiations by bold lower case letters (e.g. $\mathbf{x}$). We use $\mathbf{E}$ to denote the set of currently observed features in a Naive Bayes network, and $\mathbf{H}$ to denote the set of unobserved features. The decision variable, which we assume is binary, is denoted by D with states $+$ and $-$.

Value of Information. Consider the Naive Bayes network in Figure 1, which will serve as a running example. Here, D is a decision variable and H_1, H_2, H_3, H_4 are tests that bear on this decision. The decision threshold is 0.80, leading to a positive decision if $Pr(D{=}+ \mid \mathbf{e}) \geq 0.8$.[2]

Suppose that tests H_1 and H_2 have a cost of \$4 each, whereas tests H_3 and H_4 have a cost of \$1 each. We are interested in the following problem: given a budget, which features should we observe? Feature selection is typically based on maximizing the *value of information (VOI)*. A common VOI criterion is information gain. The information gain for the different features are as follows: H_1: 0.2708, H_2: 0.2781, H_3: 0.0350, H_4: 0.0054. Suppose that we have a budget of \$5 to expend. Using information gain as the reward criterion, features H_2 and H_3 would be the optimal features to observe as they maximize information gain (with a combined gain of 0.2815) while being within budget. Note that myopic (greedy) feature selection (Yu et al. 2009; Gao and Koller 2011) agrees with non-myopic (optimal) feature selection (Krause and Guestrin 2009; Bilgic and Getoor 2011) in this example.

We next review the recently introduced notion of the *Same-Decision Probability (SDP)* (Darwiche and Choi 2010; Choi, Xue, and Darwiche 2012; Chen, Choi, and Darwiche 2012) and then justify the usage of the SDP as a reward function for VOI.

Same-Decision Probability. Consider our running example and suppose that no features have been observed. We then have $Pr(D{=}+) = 0.5$, leading to the decision (classification) $D{=}-$. The question we ask now is: What is the probability that this decision will stay the same after we observe features $\mathbf{H} = \{H_1, \ldots, H_4\}$? This probability is the *Same-Decision Probability* (SDP).

Intuitively, we can compute the SDP by considering every possible instantiation $\mathbf{h}$ of features $\mathbf{H}$, computing the decision under each instantiation, and then adding up the probability of each instantiation $\mathbf{h}$ that leads to the same decision (i.e., $Pr(d \mid \mathbf{h}) < 0.8$). The process of enumerating instantiations is shown in Table 1, leading to an SDP of 0.738. Hence, the probability of reaching a different decision after observing features $\mathbf{H} = H_1, \ldots, H_4$ is only 0.262.

Assuming that features $\mathbf{E}$ have been already observed to $\mathbf{e}$ and that $\mathbf{H}$ are the unobserved features, we have:

$$SDP(d, \mathbf{H}, \mathbf{e}, T) = \sum_{\mathbf{h}} [Pr(d \mid \mathbf{h}, \mathbf{e}) \geq T] Pr(\mathbf{h} \mid \mathbf{e}). \quad (1)$$

Here, $[\alpha]$ is an indicator function equal to 1 if α is true and equal to 0 otherwise. In the next section, we propose using the SDP as a reward function when computing the VOI.

Maximizing Decision Robustness

Using the SDP as a reward criterion for VOI, we would observe feature H_1 instead of both H_2 and H_3 (which would be selected by information gain). In particular, if we observe $H_1{=}+$, the decision made will not change regardless of what the other features are observed to be — since $Pr(D{=}+ \mid H_1{=}+, h_2, h_3, h_4)$ will always be greater than 0.8. Technically, this means that the SDP after observing $H_1{=}+$ is 1.0. Similarly, if we observe $H_1{=}-$, the SDP will be 1.0 as well. As such, the *expected SDP* is 1.0 after observing H_1. This means that *whatever decision* we make after observing H_1, that decision stays the same *after* observing H_2, H_3 and H_4. By a similar calculation, the expected SDP of observing both H_2 and H_3 is 0.697. This shows that maximizing information gain does not necessarily maximize decision robustness. This is an extreme example, which is meant to highlight the proposed criterion. More generally, however, we seek to identify a set of features that

[2] This threshold is often used in educational and medical diagnosis (VanLehn and Niu 2001; Cantarel et al. 2014).

H_1	H_2	H_3	H_4	$Pr(\mathbf{h} \mid \mathbf{e})$	$Pr(d \mid \mathbf{h}, \mathbf{e})$
$+$	$+$	$+$	$+$	0.0676	**0.995**
$+$	$+$	$+$	$-$	0.0430	**0.989**
$+$	$+$	$-$	$+$	0.0570	**0.994**
$+$	$+$	$-$	$-$	0.0366	**0.985**
$+$	$-$	$+$	$+$	0.0179	**0.937**
$+$	$-$	$+$	$-$	0.0125	**0.858**
$+$	$-$	$-$	$+$	0.0155	**0.913**
$+$	$-$	$-$	$-$	0.0111	**0.810**
$-$	$+$	$+$	$+$	0.0828	0.788
$-$	$+$	$+$	$-$	0.0690	0.602
$-$	$+$	$-$	$+$	0.0757	0.725
$-$	$+$	$-$	$-$	0.0676	0.517
$-$	$-$	$+$	$+$	0.0863	0.189
$-$	$-$	$+$	$-$	0.1202	0.086
$-$	$-$	$-$	$+$	0.0969	0.141
$-$	$-$	$-$	$-$	0.1393	0.062

Table 1: Scenarios $\mathbf{h}$ for the network in Figure 1. The cases where $Pr(d \mid \mathbf{h}, \mathbf{e}) \geq 0.8$ are bolded. Note that whether $H_1{=}+$ or $H_1{=}-$ is entirely indicative of whether $Pr(d \mid \mathbf{h}, \mathbf{e}) \geq 0.8$.

maximize the expected SDP, even though the expected SDP may not necessarily be 1.0.

We now formally define the expected SDP. Suppose that we have already observed features $\mathbf{E}$ to $\mathbf{e}$, and let $\mathbf{G}$ be a subset of the hidden features $\mathbf{H}$. Our goal is to define a new measure for deciding the worth of observing $\mathbf{G}$. Let $D(\mathbf{ge})$ be the decision made assuming that features $\mathbf{G}$ are observed to the values $\mathbf{g}$. The **expected SDP (E-SDP)** is:

$$\mathcal{E}(D, \mathbf{G}, \mathbf{H}, \mathbf{e}, T) = \sum_{\mathbf{g}} SDP(D(\mathbf{ge}), \mathbf{H} \setminus \mathbf{G}, \mathbf{ge}, T) Pr(\mathbf{g}|\mathbf{e}). \quad (2)$$

The E-SDP of features $\mathbf{G}$ can be thought of as measuring the similarity between the decision made after observing $\mathbf{G} \subseteq \mathbf{H}$ and the one made after observing all of $\mathbf{H}$ (Chen, Choi, and Darwiche 2014). If the E-SDP of a set $\mathbf{G}$ is high, we know that after observing $\mathbf{G}$, there is little chance that observing $\mathbf{H} \setminus \mathbf{G}$ will change our decision. This means that observing $\mathbf{G}$ will lead to high *decision robustness*. We are thus interested in finding the set of features that maximize the E-SDP. We refer to this criterion as *maximizing decision robustness (DR)*. It should be clear that choosing features based on DR amounts to maximizing an expected reward, where $SDP(D(\mathbf{ge}), \mathbf{H} \setminus \mathbf{G}, \mathbf{ge}, T)$ is the reward function. We next provide an algorithm for identifying the subset $\mathbf{G}$ of hidden features that maximizes decision robustness.

MaxDR: Maximizing Decision Robustness

Given a Naive Bayes network $\mathcal{N}$ with class variable D, threshold T, observation $\mathbf{e}$, unobserved features $\mathbf{H}$, a cost C_i for each individual feature H_i, and a total budget B, our algorithm `MaxDR` will output a set of features $\mathbf{G}^\star \subseteq \mathbf{H}$ where

- the cost of observing $\mathbf{G}^\star$ is within the budget B:

$$\Big(\sum_{H_i \in \mathbf{G}^\star} C_i \Big) \leq B,$$

- the E-SDP of observing $\mathbf{G}$ is maximal:

$$\mathbf{G}^\star = \arg\max_{\mathbf{G} \subseteq \mathbf{H}} \mathcal{E}(D, \mathbf{G}, \mathbf{H}, \mathbf{e}, T).$$

Intuitively, observing the set of features identified by `MaxDR` will best render the remaining features redundant. Note the similarity between our problem and the 0-1 **knapsack problem:** we have a set of items, each with an associated cost and profit, and a budget B, where our goal is to maximize the profit while keeping the total cost within budget B. The knapsack problem is difficult: the decision problem is NP-complete (Kellerer, Pferschy, and Pisinger 2004). There is a key difference, however, between our problem and the typical knapsack problem: computing the total profit of a set for the knapsack problem can be done in linear time, while computing the total profit of a set for our problem involves computing E-SDP, which is NP-hard even in Naive Bayes networks.[3]

We will first show how to compute E-SDP, and then show how to solve the knapsack component of the problem.

Computing the Expected SDP. Computing the E-SDP of $\mathbf{G} \subseteq \mathbf{H}$ involves the test $Pr(d \mid \mathbf{h}, \mathbf{e}) =_T Pr(d \mid \mathbf{g}, \mathbf{e})$. That is, we test whether both posterior probabilities are on the "same side" of threshold T. We find it more convenient to implement this test in the *log-odds* domain, where

$$\log O(d \mid \mathbf{h}, \mathbf{e}) = \log \frac{Pr(d \mid \mathbf{h}, \mathbf{e})}{Pr(\overline{d} \mid \mathbf{h}, \mathbf{e})}.$$

We then define the *log-odds threshold* as

$$\lambda = \log \frac{T}{1 - T},$$

and subsequently we have the log-odds test:

$$\log O(d \mid \mathbf{h}, \mathbf{e}) =_\lambda \log O(d \mid \mathbf{g}, \mathbf{e}).$$

In a Naive Bayes network we have D as the class variable and $\mathbf{H} \cup \mathbf{E}$ as the leaf variables. The posterior log-odds after observing an arbitrary *partial instantiation* $\mathbf{q} = \{h_1, \ldots, h_j\}$ of variables $\mathbf{Q} \subseteq \mathbf{H}$ can be written as

$$\log O(d \mid \mathbf{q}, \mathbf{e}) = \log O(d \mid \mathbf{e}) + \sum_{i=1}^{j} w_{h_i},$$

where w_{h_i} is the *weight of evidence* h_i and defined as:

$$w_{h_i} = \log \frac{Pr(h_i \mid d)}{Pr(h_i \mid \overline{d})}.$$

The weight of evidence w_{h_i} is then the contribution of evidence h_i to the quantity $\log O(d \mid \mathbf{q}, \mathbf{e})$ (Chan and Darwiche 2003). Note that all weights can be computed in time and space linear in $|\mathbf{H}|$ (for Naive Bayes networks). For our example network, the weights of evidence $\{w_{h_i}, w_{\overline{h}_i}\}$ are shown in Table 2.

[3] Computing the SDP in Naive Bayes networks is NP-hard (Chen, Choi, and Darwiche 2013). Computing E-SDP is NP-hard as well since computing the SDP is just a special case of computing the E-SDP where $\mathbf{G} = \{\}$.

i	1	2	3	4
w_{h_i}	5.00	2.00	0.65	0.25
$w_{\overline{h}_i}$	1.00	-2.00	-0.65	-0.25

Table 2: Weights of evidence.

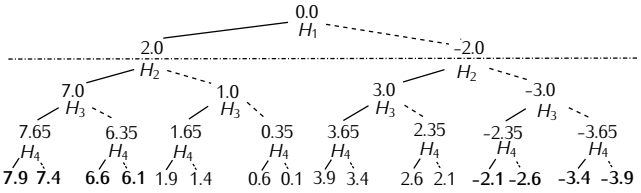


Figure 3: This figure depicts a two-tiered search tree to compute E-SDP with $\mathbf{G} = \{H_1\}$ and $\lambda = 2.0$ for the network shown in Figure 1. A solid line indicates $+$ and a dashed line indicates $-$. Note the horizontal line dividing $\mathbf{G}$ and the remaining features. The quantity $\log O(d \mid \mathbf{q}, \mathbf{e})$ is displayed next to each node $\mathbf{q}$ in the tree. Leaf nodes where $\log O(d \mid \mathbf{h}, \mathbf{e}) =_\lambda \log O(d \mid \mathbf{g}, \mathbf{e})$ are bolded.

For convenience, we also define the quantities UB and LB that respectively represent the upper and lower bounds on the total weight of evidence of observing variables $\mathbf{Q}$:

$$UB(w_{\mathbf{Q}}) = \sum_{H_i \in \mathbf{Q}} \max_{h_i} w_{h_i}$$

$$LB(w_{\mathbf{Q}}) = \sum_{H_i \in \mathbf{Q}} \min_{h_i} w_{h_i}$$

One brute-force method to compute the E-SDP is similar to the brute force method discussed in (Chen, Choi, and Darwiche 2013) for computing the SDP, where we first initialize the total SDP to 0, then enumerate the instantiations of $\mathbf{H}$ into a search tree, then check whether $\log O(d \mid \mathbf{h}, \mathbf{e}) =_\lambda \log O(d \mid \mathbf{e})$, and if so, add $Pr(\mathbf{h} \mid \mathbf{e})$ to the total SDP.

Our proposed algorithm, `CalcESDP`, utilizes a two-tiered search tree. The first tier of the search tree includes $\mathbf{G}$, whereas the second tier includes $\mathbf{H} \setminus \mathbf{G}$. An example two-tiered search tree can be seen in Figure 3. To reduce the number of nodes that must be explored, we use a branch-and-bound approach similar to (Chen, Choi, and Darwiche 2013) to detect when the children of an intermediate node can be pruned. In addition, our algorithm uses a novel *dynamic* variable order that leads to significant improvements to the amount of pruning. The pseudocode of `CalcESDP` is shown in Algorithm 1. When applied to the previous example, the algorithm will only explore the tree in Figure 4.

Note that the dynamic ordering heuristic (Line 20) will select the variable that upon observation, can maximize the margin between the current posterior log-odds and the log threshold. For instance, in our running example, if $\log O(d \mid \mathbf{e}) = -2$, then H_1 would be selected, as the observation of $H_1 = -$ would result in a minimal posterior log-odds of -4. Whereas if $\log O(d \mid \mathbf{e}) = 1$, then H_2 would be selected as the observation of $H_2 = +$ would result in a maximal posterior log-odds of 6. This allows us to first explore certain parts of the search tree where we can aggressively prune. Note that we keep track of the upper bound on the final value of the E-SDP. This quantity, as well as our dynamic ordering heuristic, will be further discussed in the next section.

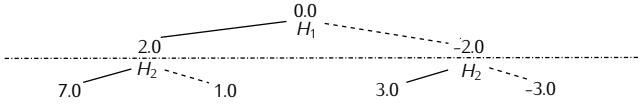


Figure 4: The pruned search tree when computing E-SDP for $\mathbf{G} = \{H_1\}$.

Algorithm 1: CalcESDP Computing E-SDP in a Naive Bayes network with class variable D, evidence $\mathbf{e}$, and attributes $\mathbf{H} = \{H_1, \ldots, H_n\}$. Note $w_{\mathbf{q}} = \sum_{h_i \in \mathbf{q}} w_{h_i}$.

input:
$\mathcal{N}$: Naive Bayes network
$\mathbf{G}$: a subset of $\mathbf{H}$
λ: log-odds threshold

output: Expected Same-Decision Probability p

main:

```
p ← 0.0                                   a lower-bound on E-SDP
m ← 1.0                                   an upper-bound on E-SDP
q ← {}                                    initial instantiation (empty)
DFS_E_SDP(q)
return p                                  p = m on return
```

```
1:  procedure DFS_E_SDP(q)
2:      t ← log O(d | e) + w_q            current log-odds
3:      if |Q| ≤ |G|                      enumerating top-half of tree
4:          if t + UB(w_{H\Q}) < λ or λ ≤ t + LB(w_{H\Q})
5:                                        decision is now fixed
6:              p ← p + Pr(q | e)
7:              return
8:      else                              searching bottom-half of tree
9:          if t + UB(w_{H\Q}) < λ or λ ≤ t + LB(w_{H\Q})
10:                                       decision is now fixed
11:             if t =_λ log O(d | e) + w_g    same decision
12:                 p ← p + Pr(q | e)
13:             else                      different decision
14:                 m ← m − Pr(q | e)
15:             return
16:     if |Q| < |H|                      continue traversing tree
17:         H_i ← NEXT_VAR(q, t)          pick next variable
18:         for each value h_i of attribute H_i do
19:             DFS_E_SDP(q, h_i)         recurse
20: procedure NEXT_VAR(q, t)
21:     C ← H \ Q                         candidate variables
22:     if |Q| < |G|                      if enumerating top half
23:         C ← C \ H                     remove bottom half
24:     return arg max_{H_i ∈ C} |(t + w_{h_i ∈ H_i} − λ)|
```

Finding Valid Sets of Features. Feature selection algorithms usually exploit submodularity and/or monotonicity (Krause and Guestrin 2005; 2009; Golovin and Krause 2011). However, as the expected SDP is neither mono-

Algorithm 2: FindCands Finding candidate subsets to compute E-SDP over. A subset $[T, F, F, T]$ represents a subset that includes features H_1 and H_4. For a candidate set to be valid, the total profit must $\geq V$ while the total cost $\leq B$.

input:
$\mathbf{H}$: $\{H_1, \ldots, H_n\}$, by decreasing efficiency
V: target threshold (see Equations 3 & 4)
B: budget

output: All subsets of features where total profit exceeds V

main:

$p \leftarrow 0.0$ *initial profit*
$\mathbf{q} \leftarrow [F, \ldots, F]_n$ *initially, no features selected*
$d \leftarrow 0$ *initial depth*
$b \leftarrow B$ *initial budget*
$candidates \leftarrow [\,]$ *list of candidate feature sets*
DFS_KS($\mathbf{q}$, p, d, b)
return $candidates$

procedure DFS_KS($\mathbf{q}$, p, d, b)
  if $p + UBP(\mathbf{H}, d, b) < V$ *no profit potential*
    return
  else if $b < 0$ *budget exceeded*
    return
  else if $p \geq V$ *enough profit potential*
    add $\mathbf{q}$ to $candidates$
  DFS_KS($\mathbf{q}$, p, $d + 1$, b)
  set feature $H_d = T$ in $\mathbf{q}$ *include* H_d
  DFS_KS($\mathbf{q}$, $p + \mathsf{profit}(H_d)$, $d + 1$, $b - \mathsf{cost}(H_d)$)

tonic, nor submodular (shown in the introductory example), a novel approach must be devised. A brute-force computation enumerates all $2^{|\mathbf{H}|}$ sets of features and then computes E-SDP for each set that respects the budget.

We can greatly improve upon this method using a key observation: if observing variables $\mathbf{G} \subseteq \mathbf{H}$ cannot change the original decision, then the E-SDP of observing variables $\mathbf{G}$ is *the same* as the original SDP with respect to variables $\mathbf{H}$ (without the expectation). Hence, we need only compute this "expected" SDP once, and focus only on those subsets $\mathbf{G}$ that are not so vacuous.

More precisely, if our decision is initially below the threshold, $\log O(d \mid \mathbf{e}) < \lambda$, then we only need to consider sets of features $\mathbf{G}$ that are able to cross the threshold: $\lambda \leq \log O(d \mid \mathbf{e}) + UB(w_{\mathbf{G}})$. Similarly, if our decision is initially above the threshold, i.e., $\lambda \geq \log O(d \mid \mathbf{e})$, then we only need to consider features $\mathbf{G}$ that satisfy $\log O(d \mid \mathbf{e}) + LB(w_{\mathbf{G}}) \leq \lambda$. Hence, to obtain a non-trivial expected SDP problem, we must select enough features in $\mathbf{G}$ so that:

$$UB(w_{\mathbf{G}}) = \sum_{H_i \in \mathbf{G}} \max_{h_i} w_{h_i} \geq \lambda - \log O(d \mid \mathbf{e}), \quad (3)$$

if our decision is initially below the threshold, or else if the decision is initially above the threshold:

$$LB(w_{\mathbf{G}}) = \sum_{H_i \in \mathbf{G}} \min_{h_i} w_{h_i} \leq \lambda - \log O(d \mid \mathbf{e}). \quad (4)$$

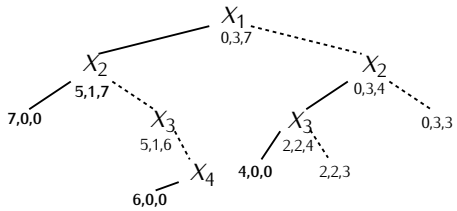


Figure 5: A pruned inclusion/exclusion tree for the knapsack problem instance shown in Table 3. Note that a left branch corresponds to the *inclusion* of an item whereas a right branch corresponds to the *exclusion* of an item. At every node there is a tuple of 3 items: the profit so far, the budget remaining, and an upper bound for the remaining profit (UBP). Due to pruning, we thus avoid traversing the full inclusion/exclusion tree.

We can find these sets by solving the following variant of the knapsack problem:

Definition 1. *Consider n items X_1 to X_n, where each item has cost $\mathsf{cost}(X_i)$ and profit $\mathsf{profit}(X_i)$. Is there a subset of items with total cost at most B and total profit at least V?*

Here, our cost function is the same cost as in our feature selection, but our profit function is either $\mathsf{profit}(H_i) = \max_{h_i} w_{h_i}$ or $\mathsf{profit}(H_i) = \min_{h_i} w_{h_i}$ (depending on the initial decision). We then want subsets of items (features) that yield enough profit for our decision to cross its threshold, and hence yield a non-trivial expected SDP problem. Our proposed algorithm, `FindCands`, uses a branch-and-bound approach similar to (Pisinger 1995; Zäpfel, Braune, and Bögl 2010), and builds an inclusion/exclusion tree, as done by (Korf 2009), in order to find all possible solutions.

In order to increase the amount of pruning that can be done, we build the inclusion/exclusion tree of items by *efficiency*, defined as the ratio $\frac{\mathsf{profit}(X_i)}{\mathsf{cost}(X_i)}$. After the items are sorted, and as we traverse the tree, we keep track of (1) all costs accrued (by the features currently selected), (2) and the remaining profit potential that is available (that we could get by selecting from the remaining features). We backtrack if we find that (1) the budget has been surpassed, or (2) the profit potential is too low, which is based on an efficiently computable upper bound on our profit, UBP — we add all items as allowed by the budget in order of decreasing efficiency. At the point where the next item's cost exceeds the remaining budget, we take, according to the remaining budget, the fraction of the item's profit, which yields an upper bound on the maximum obtainable profit (Kellerer, Pferschy, and Pisinger 2004; Zäpfel, Braune, and Bögl 2010).

The pseudocode for `FindCands` can be found in Algorithm 2. In Figure 5, we provide an example of the inclusion/exclusion tree after using `FindCands` for the following knapsack problem instance found in Table 3. The candidate knapsack items include X_1, X_2, X_3, X_4, and their respective $\{\texttt{profit}, \texttt{cost}\}$ values can be found in Table 3. The final algorithm `MaxDR` to find the feature set $\mathbf{G}^\star$ with highest expected SDP is: use `FindCands` to generate valid sets, and for each candidate set $\mathbf{G}$, use `CalcESDP` to com-

pute the expected SDP. Note that `CalcESDP` maintains an upper bound for the current value of the expected SDP — if that value falls below the highest expected SDP of some previous candidate set, we can abort the computation and continue our traversal through the inclusion/exclusion tree. This is where the previously introduced dynamic ordering heuristic is especially useful, as it allows us to swiftly detect when the upper bound falls below some threshold.

	X_1	X_2	X_3	X_4
Profit	5.0	2.0	2.0	1.0
Cost	2.0	1.0	2.0	1.0

Table 3: A knapsack problem instance with budget = 3 and profit = 4.

Complexity analysis

Let n be the number of features in the network, where c is the number of states of a feature variable. In the best-case scenario, we can detect that the E-SDP will never change regardless of what instantiation of features is observed — in this trivial scenario, no search needs to be performed. However, for the worst-case time complexity, we may need to traverse all of the $O(2^n)$ possible candidate subsets of features and take $O(c^n)$ time to compute the E-SDP of each subset. Therefore, the worst-case time complexity is $O(c^n 2^n)$.

Applications and Experimental Results

We now discuss the application of `MaxDR` to classification and decision making tasks. We performed experiments on Naive Bayes networks from a variety of sources: UCI Machine Learning Repository (Bache and Lichman 2013), BFC (`http://www.berkeleyfreeclinic.org/`) and CRESST (`http://www.cse.ucla.edu/`). Each network has an associated dataset containing labelled examples, which we used in some of the experiments.

Classification. Consider the problem of classification with Naive Bayesian networks. Let D be the class variable and let $\mathbf{H} = H_1, \ldots, H_n$ be the features. In a classical setting, one classifies by observing all the available features. That is, one computes $Pr(D|H_1, \ldots, H_n)$ and then chooses a class depending on a given threshold T. If $Pr(d|h_1, \ldots, h_n) \geq T$, one classifies the example $h_1, \ldots, h_n$ positively; otherwise, one classifies it negatively.

Suppose now that we wish to classify based on only a subset of the features, due to feature cost. Our approach involves using `MaxDR` to find the smallest subset $\mathbf{G}$ of features where the expected SDP of observing $\mathbf{G}$ crosses a certain level L. In other words, we want the subset $\mathbf{G}$ that satisfies $\arg\min_{\mathbf{G}} |\mathbf{G}|$ and $\mathcal{E}(D, \mathbf{G}, \mathbf{H}, \mathbf{E}, T) \geq L$. Note here that we are not maximizing the expected SDP, but only ensuring that it passes a given threshold L. We do this so we can tradeoff the number of selected features with the classification accuracy. The larger L is, the more features `MaxDR` will select, and the better the classification accuracy. The smaller L is, the fewer features that `MaxDR` will select, but at the expense of less classification accuracy. As we shall see next, one can choose L so as to obtain a very interesting balance: a relatively small number of features can be selected while ensuring a very small penalty in classification accuracy.

We experimented with thresholds L ranging from $[0.70, 0.75, 0.80, 0.85, 0.90, 0.95]$. For each L, and each network, `MaxDR` found the smallest subset $\mathbf{G}$ that satisfies the above conditions. We then computed the classification accuracy based on observing only $\mathbf{G}$ versus observing all features $\mathbf{H}$. We classified each example in each dataset based on $\mathbf{G}$ and $\mathbf{H}$, and then computed for each the proportion of correctly labeled instances to total number of instances.

Table 4 displays the percentage of features ($\frac{|\mathbf{G}|}{|\mathbf{H}|}$) that `MaxDR` selected for different levels L of decision robustness, across all networks. We also list the average percentage of selected features and the average difference of classification accuracy (accuracy of $\mathbf{H}$ - accuracy of $\mathbf{G}$) for each threshold L. The table reveals an interesting tradeoff between the number of features selected and the classification accuracy, which can be controlled by varying the decision robustness threshold L. For example, for $L = 0.95$, we select about 60% of the features on average, while incurring only a 0.64% reduction in classification accuracy. This clearly shows that many features are redundant given other features. Moreover, `MaxDR` is able to identify those redundant features, by excluding them from the selected features.

Decision Making. We now discuss another application of `MaxDR` to decision making. Given a Naive Bayes network with decision variable D, features $\mathbf{H} = H_1, \ldots, H_n$, and budget B, we wish to make a decision that is within budget, and that is maximally robust against additional observations. For example, if we can only observe three features, H_i, H_j and H_k, we wish to select them such that when observed, there is a low probability of reaching a different decision if features $\mathbf{H} \setminus \{H_i, H_j, H_k\}$ were later observed. The goal here is to minimize our liability against what we chose not to observe. Contrary to the previous application, we have no data here. Therefore, the notion of classification accuracy is not meaningful in this setting, and the quality of a decision is measured solely based on its robustness.

For each Naive Bayes network, budget B and decision threshold T, we used `MaxDR` to select features $\mathbf{G}$ and considered the expected SDP of decisions based on observing $\mathbf{G}$ (`MaxDR` computes the expected SDP as a side effect). For each network, the budget was set to $1/3$ the number of features, with decision thresholds in $[0.1, 0.2, \ldots, 0.8, 0.9]$.

Figure 6 depicts a sample result for one of the considered networks.[4] We compare against two baseline feature selection algorithms: non-myopic information gain (`NM-IG`) and non-myopic, generalized classification loss (`NM-DT`).[5] That is, we select features $\mathbf{G}$ using each of these feature selection criteria and then measure the robustness of decisions based

[4] Additional results are given in the full version of our paper at `http://reasoning.cs.ucla.edu/`.

[5] Suppose we are making decisions based on the following utilities of positive and negative decisions: $U_p = Pr(d|\mathbf{e})$ and $U_n = Pr(\bar{d}|\mathbf{e})T/(1-T)$. This corresponds to a threshold-based decision for arbitrary T. `NM-DT` selects features using the reward function $\max(U_p, U_n)$. When $T = 1/2$, we get classification loss.

Network	bupa	pima	ident	anatomy	heart	voting	hepatitis	nav		
# features	6	8	9	12	13	16	19	20		
Threshold	% features selected								avg % selected	avg diff C.A.
0.75	33.3	12.5	11.1	8.3	15.4	6.2	5.3	30.0	15.2	0.0518
0.80	66.7	25.0	11.1	16.6	23.1	6.2	5.3	30.0	23.0	0.0191
0.85	66.7	37.5	22.2	16.6	30.8	12.5	15.8	30.0	29.0	0.0162
0.90	100	50.0	33.3	25.0	38.5	12.5	26.3	45.0	41.3	0.0109
0.95	100	87.5	55.5	41.6	61.5	18.7	47.4	60.0	59.1	0.0064
0.98	100	100	88.8	66.6	76.9	31.2	68.4	75.0	75.9	0.0048
0.99	100	100	88.8	83.3	92.3	43.7	89.5	80.0	84.7	0.0042
1.00	100	100	100	100	100	100	100	100	100	0.0000

Table 4: This table shows 1) the percentage of features selected by MaxDR for different levels of decision robustness 2) the average percentage of features selected and 3) the average difference of classification accuracy.

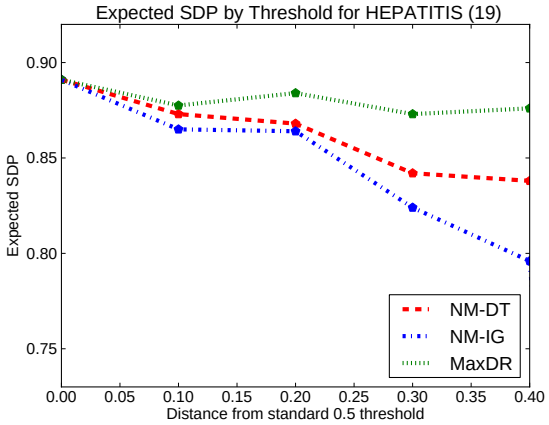


Figure 6: Decision robustness.

Network	NM-DT (s)	NM-IG (s)	MaxDR (s)
bupa	0.028	0.021	0.032
pima	0.137	0.124	0.044
ident	0.129	0.119	0.147
anatomy	0.431	0.419	0.424
heart	0.704	0.682	3.230
voting	28.146	22.734	4.612
hepatitis	284.873	276.143	147.887
nav	342.767	328.874	121.715

Figure 7: Running time.

on observing the selected features.[6] Our results show a number of patterns. First, the proposed criterion selects features leading to the most robust decisions. Moreover, the difference between MaxDR and the other criteria becomes more pronounced for extreme thresholds and for large networks. These results show that MaxDR, as a feature selection algorithm, is very suitable for applications in which the decision maker wishes to minimize their liability against information that they chose not to observe.

Running time. Assuming n binary features, a budget of m, and a cost of 1 per feature, NM-IG and NM-DT can be implemented in $O(\binom{n}{m}2^m)$ time. In particular, one only needs to consider candidate features $\mathbf{G}$ of size m due to the monotonicity of corresponding objective functions (i.e., $\mathbf{G}$ will dominate all its strict subsets). For each candidate $\mathbf{G}$, one needs to compute an expectation over $\mathbf{G}$, which takes $O(2^m)$ time. In fact, we know of no algorithm that does better than $\Theta(\binom{n}{m}2^m)$ for these criteria. MaxDR has a worst-case time complexity of $O(2^{n+m})$. In this case, we need to consider all 2^m candidates $\mathbf{G}$ with size $\leq m$ since the SDP is not strictly monotonic. We also need $O(2^n)$ time to compute the expected SDP by enumerating all feature states. This is much worse than the running time for NM-IG and NM-DT. However, Figure 7 shows that MaxDR does significantly better than this on average and also does better than both NM-IG and NM-DT, especially for larger networks. This is due to the knapsack and branch-and-bound pruning techniques employed by MaxDR.

Conclusion

We introduced a new criterion for measuring the VOI, which favors features that lead to robust decisions. We also proposed an algorithm, MaxDR, that optimally selects features in Naive Bayes networks based on the new criterion. We demonstrated the application of MaxDR to classification and decision making tasks. In the first application, we showed empirically that MaxDR can be used to significantly reduce the cost of observing features while minimally affecting the classification accuracy. In the second application, we showed empirically that MaxDR is more suitable than traditional VOI criteria when the goal is to make stable decisions given a limited budget.

Acknowledgments. This work has been partially supported by ONR grant #N00014-12-1-0423, NSF grant #IIS-1118122, and a Google Research Award. In addition, we would like to thank Brian Milch, Russell Greiner, Guy Van den Broeck, and Richard Korf for comments and suggestions that were crucial for shaping this paper.

[6] We also experimented with the minimum Redundancy Maximum Relevance (mRMR) feature selection criterion (Ding and Peng 2003). We found the criterion to be trivial for Naive Bayes structures as it always prefers sets of features over their supersets.

References

Ahmad, S., and Yu, A. J. 2013. Active sensing as Bayes-optimal sequential decision making. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI-13)*, 12–21.

Bache, K., and Lichman, M. 2013. UCI machine learning repository.

Bellala, G.; Stanley, J.; Bhavnani, S. K.; and Scott, C. 2013. A rank-based approach to active diagnosis. *IEEE Trans. Pattern Anal. Mach. Intell.* 35(9):2078–2090.

Bilgic, M., and Getoor, L. 2011. Value of information lattice: Exploiting probabilistic independence for effective feature subset acquisition. *Journal of Artificial Intelligence Research (JAIR)* 41:69–95.

Cantarel, B. L.; Weaver, D.; McNeill, N.; Zhang, J.; Mackey, A. J.; and Reese, J. 2014. Baysic: a Bayesian method for combining sets of genome variants with improved specificity and sensitivity. *BMC bioinformatics* 15(1):104.

Chan, H., and Darwiche, A. 2003. Reasoning about Bayesian network classifiers. In *Proceedings of the 19th Conference in Uncertainty in Artificial Intelligence*, 107–115.

Chen, S.; Choi, A.; and Darwiche, A. 2012. The Same-Decision Probability: A new tool for decision making. In *Proceedings of the Sixth European Workshop on Probabilistic Graphical Models*, 51–58.

Chen, S.; Choi, A.; and Darwiche, A. 2013. An exact algorithm for computing the Same-Decision Probability. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence*, 2525–2531.

Chen, S.; Choi, A.; and Darwiche, A. 2014. Algorithms and applications for the Same-Decision Probability. *Journal of Artificial Intelligence Research (JAIR)* 49:601–633.

Choi, A.; Xue, Y.; and Darwiche, A. 2012. Same-Decision Probability: A confidence measure for threshold-based decisions. *International Journal of Approximate Reasoning (IJAR)* 2:1415–1428.

Darwiche, A., and Choi, A. 2010. Same-Decision Probability: A confidence measure for threshold-based decisions under noisy sensors. In *Proceedings of the Fifth European Workshop on Probabilistic Graphical Models*, 113–120.

De Campos, C. P. 2011. New complexity results for MAP in Bayesian networks. In *Proceedings of the Twenty-Second IJCAI*, 2100–2106. AAAI Press.

Ding, C., and Peng, H. 2003. Minimum redundancy feature selection from microarray gene expression data. In *Proceedings of the IEEE Computer Society Conference on Bioinformatics*, CSB '03, 523–. Washington, DC, USA: IEEE Computer Society.

Gao, T., and Koller, D. 2011. Active classification based on value of classifier. In *Advances in Neural Information Processing Systems (NIPS 2011)*.

Golovin, D., and Krause, A. 2011. Adaptive submodularity: Theory and applications in active learning and stochastic optimization. *Journal of Artificial Intelligence Research (JAIR)* 42:427–486.

Greiner, R.; Grove, A. J.; and Roth, D. 2002. Learning cost-sensitive active classifiers. *Artificial Intelligence* 139(2):137–174.

Kellerer, H.; Pferschy, U.; and Pisinger, D. 2004. *Knapsack problems*. Springer.

Korf, R. 2009. Multi-way number partitioning. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence*, 538–543.

Krause, A., and Guestrin, C. 2005. Near-optimal nonmyopic value of information in graphical models. In *21st Conference on Uncertainty in Artificial Intelligence*, 324–331.

Krause, A., and Guestrin, C. 2009. Optimal value of information in graphical models. *Journal of Artificial Intelligence Research (JAIR)* 35:557–591.

Lu, T.-C., and Przytula, K. W. 2006. Focusing strategies for multiple fault diagnosis. In *Proceedings of the 19th International FLAIRS Conference*, 842–847.

Millán, E.; Descalco, L.; Castillo, G.; Oliveira, P.; and Diogo, S. 2013. Using Bayesian networks to improve knowledge assessment. *Computers & Education* 60(1):436–447.

Munie, M., and Shoham, Y. 2008. Optimal testing of structured knowledge. In *Proceedings of the 23rd National Conference on Artificial intelligence*, 1069–1074.

Park, J. D., and Darwiche, A. 2004. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research (JAIR)* 21:101–133.

Pisinger, D. 1995. *Algorithms for Knapsack Problems*. Ph.D. Dissertation, University of Copenhagen.

Shimony, S. E. 1994. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence* 68(2):399–410.

VanLehn, K., and Niu, Z. 2001. Bayesian student modeling, user interfaces and feedback: A sensitivity analysis. *International Journal of Artificial Intelligence in Education* 12(2):154–184.

Yu, S.; Krishnapuram, B.; Rosales, R.; and Rao, R. B. 2009. Active sensing. In *International Conference on Artificial Intelligence and Statistics*, 639–646.

Zäpfel, G.; Braune, R.; and Bögl, M. 2010. *Metaheuristic Search Concepts: A Tutorial with Applications to Production and Logistics*. Springer-Verlag Berlin Heidelberg.

Zhang, Y., and Ji, Q. 2010. Efficient sensor selection for active information fusion. *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 40(3):719–728.