Online Streaming Feature Selection

Abstract
In the paper, we consider an interesting and challenging problem, online streaming feature selection, in which the size of the feature set is unknown, and not all features are available from learning while leaving the number of observations constant. In this problem, the candidate features arrive one at a time, and the learner's task is to select a “best so far” set of features from streaming features. The standard feature selection methods cannot perform well in this scenario. Thus, we present a novel framework based on feature relevance. Under the framework, a promising alternative method, online streaming feature selection (OSFS, for short), is presented to online select strongly relevant and non-redundant features. In addition to OSFS, a faster Fast-OSFS algorithm is proposed to further improve the selection efficiency. Experiment results show that our methods achieve both more compactness and better accuracy than existing streaming feature selection algorithms on various datasets.

1. Introduction
Feature selection for predictive modeling has received considerable attention during the last three decades both in statistics and in machine learning. A great variety of feature selection algorithms have been developed and proven to be effective in improving predictive accuracy for classification (Kohavi 1997; Guyon 2003; Dhillon 2008, Yu 2009). The standard feature selection methods assume that all candidate features are available and presented to a learner before any feature selection methods begin.

In this paper, another interesting scenario is taken into account where the candidate feature set size is unknown, or even infinite instead of all of the candidate features being known in advance. In this problem, the candidate features are generated dynamically and arrive one at a time while the number of observations is left constant. The scenario is called streaming feature selection. The task of streaming feature selection is to select a minimal yet good set of features from the features generated so far. Streaming feature selection has the practical use in many settings. For example, when we have potentially hundreds of thousands of features and the computational cost is expensive in generating those features, we cannot afford to wait until all features have been generated before learning begins so that it could be far more preferable to generate each new feature dynamically and to online evaluate it. Another example is that the feature set has an infinite size. One way of managing the situation is to generate features, one at a time in a random order, and to select a “best so far” set of features.

Although many standard feature selection algorithms are effective in selecting a subset of predictive features for various classification problems, they are not necessarily reliable to deal with the scenario above. Therefore, in this paper, we present a novel framework for streaming feature selection which aims to solve the task. Our framework is motivated by the definitions of feature relevance and feature redundancy and is clearly different from existing work for streaming feature selection. Our work gives a new definition of redundant features, and presents a novel framework for streaming feature selection based on feature relevance which consists of two parts: (1) online selecting relevant features one by one; (2) dynamically removing the redundant features from the set of features selected so far.

2. Related Work
For many years, feature selection as an effective means when dealing with large dimensionality with many irrelevant features, has been generally viewed as a problem of searching for an optimal subset of features guided by some evaluation measures. Various feature selection methods can be broadly classified into three categories: filter, wrapper, and embedded methods. The filter method is independent of a classifier, instead applying statistical criteria to first select features and then build the classifier with the best features (Dash 2003), while the wrapper method uses heuristic search in the space of all possible variable subsets using a classifier of choice to assess each subset (Kohavi 1997). In addition, the embedded method attempts to simultaneously maximize classification performance while minimizing the number of features used (Tibshirani 1996).

All work discussed above assumes that all candidate features are available from the beginning and pays little
attention to candidate feature sets of unknown, or even infinite size, that is, the problem of streaming feature selection.

Two major lines of research efforts have recently studied the problem. Perkins et al first proposed a grafting algorithm based on a stagewise gradient descent approach for streaming feature selection (Perkins 2003). However, in fact, grafting requires all candidate features in advance to determine the value of the tuning parameter $\lambda$ using cross-validation before learning. Thus, grafting is a quasi-streaming feature selection method. Zhou et al presented two algorithms based on streamwise regression, information-investing and alpha-investing for streaming feature selection (Zhou 2005; Zhou 2006). Since information-investing gave extremely similar results with alpha-investing and alpha-investing was emphasized in their work, we focus on alpha-investing in this paper. Alpha-investing uses a p-value to determine whether a new feature is added to the model or not, and a linear regression to evaluate the current model. But it needs some prior knowledge about the structure of the space of potential features. Alpha-investing needs to provide an ordering on features to generate features and put the potentially useful features at the head of the streaming features. In such settings, it is efficient. On the contrary Alpha-investing might not provide good performance on the original streaming features, as in the real world, we cannot get enough prior information about the structure of the candidate features, and we can always acquire the original feature stream. Thus, more efforts are needed in order to manage the original feature stream without any transformations of the original features in advance.

Therefore, our work takes a paradigm shift from the above research efforts and proposes a novel framework which is clearly different from previous work on streaming feature selection. Under the framework, a novel online streaming feature selection algorithm (OSFS for short) is presented in this paper. In addition to OSFS, a faster Fast-OSFS algorithm is proposed to further improve the selection efficiency.

3. A Novel Framework For Streaming Feature Selection

In the section, we first review notions of feature relevance. Then we redefine feature redundancy and propose a novel framework for streaming feature selection based on feature relevance.

3.1 Notations and Definitions

John and Koller et al. proposed a classification of input features $X$ with respect to their relevance to the target $T$ in terms of conditional independence (Kohavi 1997; Koller 1996). They classified features into three disjoint categories, namely, strongly relevant, weakly relevant and irrelevant features. In the following definitions, let $V$ be a full set of features, $X_i$ denote the $i$th input features, and $X_{\bar{i}}$ represent all input features excluding $X_i$.

**Definition 1 (Conditional independence)** In a feature set $V$, Two features $X$ and $Y$ are conditionally independent given the set of features $Z$, if and only if

$$P(X|Y,Z) = P(X|Z), \text{ denoted as } \text{Ind}(X,Y|Z).$$

For notational convenience we will denote conditional dependence as $\text{Dep}(X,Y|Z)$.

**Definition 2 (Strong relevance)** A feature $X_i$ is strongly relevant to the target $T$ if

$$P(T|X_i) \neq P(T|X_{\bar{i}},X_i)$$

**Definition 3 (Week relevance)** A feature $X_i$ is weakly relevant to the target $T$ if it is not strongly relevant and

$$\exists S \subseteq X_i: P(T|S) \neq P(T|S,X_i)$$

**Definition 4 (Irrelevance)** A feature $X_i$ is irrelevant to the target $T$ if it is neither strongly nor weakly relevant, that is, if

$$\forall S \subseteq X_i: P(T|S) = P(T|S,X_i)$$

Yu and Tuv et al. further studied the feature relevance and pointed out that the weakly relevant feature set could be classified into redundant features and non-redundant features. They gave a definition of feature redundancy based on a Markov blanket criterion (Yu 2004; Eugene Tuv 2009).

**Definition 5 (Markov blanket)** Given a feature $X_i$, assuming $M_i \in V \setminus X_i \notin M_i$, $M_i$ is said to be a Markov blanket for $X_i$, if and only if

$$P(V-M_i \setminus \{X_i\}, T|X_i, M_i) = P(V-M_i \setminus \{X_i\}, T|M_i)$$

**Definition 6 (Redundant feature-1)** Let $V$ be the current set of features. A feature is redundant and hence should be removed from $V$, if and only if it is weakly relevant and has a Markov blanket $M_i$ within $V$.

According to the definition of redundant features, we further study the relation of redundancy between a target feature $T$ and a feature $X$, and give our definition of a redundant feature using conditional independence as follows.

**Definition 7 (Redundant feature-2)** Given a Markov blanket of the target feature $T$, denoted as $MB(T)$, and a feature $X \in MB(T)$, $X$ is said to be redundant to $T$, if and only if

$$\exists S \subseteq MB(T): P(T|X, S) = P(T|S)$$

(1)

Proof: $\Rightarrow$ According to definition 5 and the term $X \in MB(T)$, $X$ is relevant to $T$. Because $X$ is redundant to $T$, there must exist a subset $S \subseteq MB(T)$ which subsumes all of the information that $X$ has about $T$. That is to say, $X$
and T are conditional independence given the subset S. Then we get formula (1).

\[ \iff \text{According to formula (1) and the term } X \in MB(T), \text{ the subset } S \text{ carries the information that } X \text{ has about } T. \text{ Thus, } X \text{ is redundant to } T. \square \]

3.2 A Novel Framework for Streaming Feature Selection

Based on the definitions above, an entire feature set is divided into four basic disjoint parts: (1) irrelevant features, (2) redundant features (part of weakly relevant features), (3) weakly relevant but non-redundant features, and (4) strongly relevant features. An optimal subset essentially contains non-redundant and strongly relevant features.

Since the global information of all candidate features is unknown and the features are generated continuously, it is difficult to find all of strongly relevant and non-redundant features from streaming features. Our task is to design an efficient yet effective way to find an optimal subset from streaming features. Since searching for an optimal subset based on the definitions of feature relevance and redundancy is combinatorial in nature, we can achieve this goal through a novel framework based on online feature relevance and redundancy analysis (shown in Figure 1).

![Figure 1 A novel framework for streaming feature selection](image)

The framework is composed of two steps: first, online relevance analysis determines a new feature with respect to its relevance to the target T and removes irrelevant ones; and second, online redundancy analysis determines and eliminates redundant features from the features selected so far. The two steps are alternated till some stopping criteria are satisfied.

4. ALGORITHMS AND ANALYSIS

Under the framework in Section 3, two novel algorithms are presented for streaming features selection, OSFS and Fast-OSFS, and an analysis is given in this section.

4.1 An Online Streaming Feature Selection Algorithm

The pseudo-code of our online streaming feature selection (OSFS for short) algorithm is shown in Figure 2.

![Figure 2. The OSFS algorithm](image)

**The OSFS algorithm**

<table>
<thead>
<tr>
<th>Input: class label C, feature stream X</th>
<th>Output: BCF (the best candidate features so far)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. BCF = {}</td>
<td>13. <strong>online redundancy analysis</strong></td>
</tr>
<tr>
<td>2. (i = 1)</td>
<td>14. if (\text{added})</td>
</tr>
<tr>
<td>3. <strong>repeat</strong></td>
<td>15. for each feature (Y \subseteq \text{BCF})</td>
</tr>
<tr>
<td>4. <strong>online relevance analysis</strong></td>
<td>16. if (\exists Y \subseteq \text{BCF}, \text{ s.t. } \text{ind}(Y; C</td>
</tr>
<tr>
<td>5. (\text{added} = 0)</td>
<td>17. <strong>remove redundant feature</strong></td>
</tr>
<tr>
<td>6. <strong>generating new features</strong></td>
<td>18. (\text{BCF} = \text{BCF} \cup Y)</td>
</tr>
<tr>
<td>7. (X \rightarrow \text{get next feature()})</td>
<td>19. <strong>endif</strong></td>
</tr>
<tr>
<td>8. if (\text{Dep}(X; C; \emptyset))</td>
<td>20. <strong>endfor</strong></td>
</tr>
<tr>
<td>9. <strong>add relevant feature (X_i) to BCF</strong></td>
<td>21. <strong>endif</strong></td>
</tr>
<tr>
<td>10. (\text{BCF} = \text{BCF} \cup X_i)</td>
<td>22. (i = i + 1)</td>
</tr>
<tr>
<td>11. <strong>added</strong> = 1</td>
<td>23. until the stopping criteria satisfied</td>
</tr>
<tr>
<td>12. <strong>endif</strong></td>
<td>24. output BCF</td>
</tr>
</tbody>
</table>

OSFS fines an optimal subset using a two-phase scheme: online relevance analysis (from step 4 to step 12) and online redundancy analysis (from step 14 to step 21). In the relevance analysis phase, OSFS discovers strongly or weakly features and adds them into BCF. When a new feature arrives, OSFS online assesses whether it is irrelevant to the class label C; if so, it is discarded, otherwise it is added to BCF.

If a new feature enters BCF, the redundancy analysis phase is performed. In this phase, OSFS dynamically eliminates redundant features in the subset of features selected so far. If there exists a subset within BCF to make Y and C conditionally independent, Y is removed from BCF. OSFS alternates the two phases till some stopping criteria are satisfied.

4.2 An Analysis for the OSFS Algorithm

With the OSFS algorithm above, under the assumption that all statistical independence tests are reliable, let us have an analysis about its performance in theory.

Firstly, we can analyze its performance on a small data set with hundreds of thousands of features. When the data set is so small in size so as to make most of the conditional independence tests unreliable, OSFS might fail. There are two lines in the pseudo-code for conditional independence tests. One is at line 8, and the other is at line 16. OSFS doesn’t fail at line 8, since the conditioning set is an empty set.

But OSFS might fail at line 16, when the conditioning set \(S\) exponentially grows. However, in the relevance analysis phase, only strongly or weakly relevant features are admitted into BCF. In the redundancy analysis phase OSFS dynamically evaluates each feature within BCF and removes redundant features from BCF when a feature enters into BCF. With many irrelevant and redundant features, BCF keeps as minimal as possible so that
Online Streaming Feature Selection: A Submission to ICML 2010

conditioning on all subsets of BCF is feasible. Thus, OSFS can deal with a dataset with a small sample-to-variable ratio.

Secondly, we can analyze whether the OSFS algorithm can discover all strongly relevant and some non-redundant features or not.

According to Definition 4, if a feature X is irrelevant to C, X must be discarded in the relevance analysis phase. Thus, from Definitions 2 and 3, all strongly and some weakly relevant features will enter BCF at line 10. According to Definition 7, if X is a strongly relevant feature, there doesn’t find a subset $S \subseteq BCF$ to satisfy the term $\text{Ind}(X, C \mid S)$. X cannot be removed in any phase. Thus, OSFS can find all strongly relevant features.

For a redundant feature, we consider two situations as follows. One situation is that the set of streaming features in size is unknown, but finite. According to definition 3, some weakly relevant features, including redundant features and non-redundant features, will enter BCF in the relevance analysis phase. Therefore, OSFS needs to remove redundant features from those weakly relevant features. In the redundancy analysis phase, based on Definition 7, OSFS searches a subset S for each feature within BCF to make it redundant to C. For example, if the term $\text{Dep}(X, C \mid \phi)$ is satisfied at line 8, X will be added into BCF as a relevant feature at line 10. Now assuming that X is a redundant to C, as the time goes on, the subset $S \subseteq BCF$ must be found in the redundancy analysis phase, and satisfies $\text{Ind}(X, C \mid S)$ according to Definition 7. Then X is removed from BCF at line 18.

The other situation is that if the set of streaming features in size is infinite, OSFS could fail to remove X at line 13 at time t. Because the size of streaming features is infinite and a feature is generated randomly, OSFS doesn’t know when S can be found within BCF. Thus, we don’t know when OSFS can remove X at line 18. But, in theory, since OSFS can find all strongly relevant features, if X is a redundant feature within BCF, there must exist an S to satisfy $\text{Ind}(X, C \mid S)$.

Therefore, OSFS can discover all strongly relevant features and some non-redundant features.

Finally, the time complexity is analyzed. The complexity of OSFS depends on the number of independent tests. At time t, assuming V features arriving, then the worst-case complexity is $O(|V||BCF|^k|BCF|)$ where k is the maximum allowable size that a conditioning set may grow. Assuming $SF \subseteq V, |SF| \ll |V|$ where SF contains all of strongly relevant features, then the average is $O(|SF||BCF|^k|BCF|)$ at time t.

4.3 The Fast-OSFS Algorithm

According to our analysis, the most time-consuming part of OSFS is the redundancy analysis phase. When a new feature enters BCF, the redundancy analysis phase will re-examine each feature of BCF with respect to its relevance to C. Therefore, in order to further improve the selection efficiency, Fast-OSFS is designed Figure 3.

Figure 3 The fast-OSFS algorithm

The key difference between Fast-OSFS and OSFS is that the Fast-OSFS algorithm divides the redundancy analysis phase into two phases, inner-redundancy analysis and outer-redundancy analysis. Fast-OSFS only alternates the relevance analysis and the inner-redundancy analysis phase. In the inner-redundancy analysis phase Fast-OSFS only re-examines the feature just added into BCF, whereas the outer-redundancy analysis phase re-examines each feature of BCF only when the process of generating a feature is stopped. The worst-case complexity is $O(|V|^k|BCF|^k|BCF|)$ and the average is $O(|SF|^k|BCF|^k|BCF|)$ at time t. Thus, Fast-OSFS is more efficient than OSFS.

5. Experimental Results

In this section, in order to have a comprehensive comparison of existing streaming feature selection methods on various data sets with our methods, we apply these algorithms in traditional feature selection settings, that is, those of fixed features, but the features arrive one at a time in a random order to simulate the situation of streaming features.

Our data sets include UCI benchmark data, gene expression, NIPS 2003 feature challenge data sets, and the WCCI 2006 and WCCI 2008 performance prediction challenges. We used three classifiers: Knn, J48 and Randomforest (the Spider classification package is available on the web (Spider toolbox 2010) and selected the best accuracy as the result. The experiments were conducted on a computer with Windows XP, 2.6GHz CPU and 2GB memory.
Grafting and Alpha-investing were performed using their authors' original implementation. The tuning parameter $\lambda$ for Grafting was selected using cross-validation and the parameters of Alpha-investing use default settings, $W_0=0.5$ and $s_\alpha=0.5$. The conditional independence tests in our implementation are $G^2$ tests and the parameter alpha is the statistical significance level.

5.1 Results on UCI Benchmark Data Sets

All together 8 data sets in the traditional form are selected from the UCI Machine Learning Repository. These data sets, including their problem type and numbers of features and instances, are shown in Table 1. For UCI data sets we use their original test data, and for the other three data sets we use 1-fold cross validation.

Table 1. Summary of UCI benchmark data sets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Domain</th>
<th>Features</th>
<th>Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPECT</td>
<td>MEDICINE</td>
<td>22</td>
<td>267</td>
</tr>
<tr>
<td>SPECTF</td>
<td>MEDICINE</td>
<td>44</td>
<td>267</td>
</tr>
<tr>
<td>WDBC</td>
<td>MEDICINE</td>
<td>30</td>
<td>569</td>
</tr>
<tr>
<td>IONOSPHERE</td>
<td>RADAR DATA</td>
<td>34</td>
<td>351</td>
</tr>
<tr>
<td>SPAMBASE</td>
<td>SPAM E-MAIL</td>
<td>57</td>
<td>4601</td>
</tr>
<tr>
<td>INFANT</td>
<td>MEDICINE</td>
<td>86</td>
<td>5337</td>
</tr>
<tr>
<td>BANKRUPTCY</td>
<td>FINANCIAL</td>
<td>147</td>
<td>7063</td>
</tr>
<tr>
<td>SYLVIA</td>
<td>ECOLOGY</td>
<td>216</td>
<td>14374</td>
</tr>
</tbody>
</table>

Two measurements for solving the feature selection problem are compactness (the proportion of selected features) and predictive accuracy (%). A maximally compact method which cannot achieve the optimal predictive accuracy doesn’t solve our feature selection problem. Therefore, Figure 4 reports the compactness and predictive accuracy by four algorithms where the value of alpha is up to 0.01. The best possible mark for each graph is at the upper left corner, which selects the fewest features to the best accuracy.

According to Figure 4, we have an analysis as follows. (1) Our methods vs the Alpha-investing algorithm. On 7 out of 8 data sets, our methods achieve both more compact and higher accurate results than Alpha-investing where Alpha-investing selects almost 80 percent of features on spambase data and all features on wdbc data. On the bankruptcy dataset, although they only get a little lower accuracy than Alpha-investing, our methods achieve more compact results.

(2) Our methods vs the Grafting algorithm. Our methods outperform Grafting on 6 out of 8 data sets on both compactness and accuracy. On bankruptcy and ionosphere data sets our methods are competitive with Grafting.

(3) Grafting vs Alpha-investing. Grafting is more compact than Alpha-investing on all data sets, and its accuracy is higher on a half of the datasets.

We can conclude that our methods have achieved both more compactness and higher accuracy than the two state-of-the-art algorithms.

From Figure 5, with alpha up to 0.05, our methods also outperform on 7 out of 8 datasets on both compactness and accuracy than Alpha-investing. On wdbc data, Alpha-investing has the best accuracy than the other methods, but it selected all features.

Compared with Grafting, our methods outperform on 4 out of 8 datasets on both compactness and accuracy. On the ionosphere dataset, Grafting achieves more compact and higher accuracy than the rest. On the remaining three datasets, our methods are competitive with Grafting.

Finally, we give an analysis of the performance of our two methods with different values of alpha, as shown in Figures 6 and 7. From Figure 6 to Figure 7, when alpha is up to 0.05, our two methods tend to select more features, but the performance of the two methods is different. OSFS has a little degrade while Fast-OSFS improves a little. In sum, when alpha equal to 0.01 and when alpha is up to 0.05, our experimental results reveal that the performance of our two methods is similar.

5.2 Results on Challenge Data Sets

On UCI data, we use datasets with no more than 300 features to simulate the situation of streaming features. In this section, we further assess our methods on 10 public challenge data sets with tens of thousands of features, as shown in Table 2.

Table 2. Summary of challenge datasets

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Domain</th>
<th>Features</th>
<th>Instances</th>
</tr>
</thead>
<tbody>
<tr>
<td>LYMHPOMA</td>
<td>GENE</td>
<td>7399</td>
<td>227</td>
</tr>
<tr>
<td>OVARIAN CANCER</td>
<td>PROTEOMICS</td>
<td>2190</td>
<td>216</td>
</tr>
<tr>
<td>BREAST CANCER</td>
<td>GENE</td>
<td>17816</td>
<td>286</td>
</tr>
<tr>
<td>HIVA</td>
<td>DRUG</td>
<td>1617</td>
<td>4229</td>
</tr>
<tr>
<td>NOVA</td>
<td>TEXT</td>
<td>16969</td>
<td>1929</td>
</tr>
<tr>
<td>MANELON</td>
<td>SYNTHETIC</td>
<td>500</td>
<td>2000</td>
</tr>
<tr>
<td>ARCNE</td>
<td>CLINICAL</td>
<td>10000</td>
<td>100</td>
</tr>
<tr>
<td>DEXTER</td>
<td>TEXT</td>
<td>20000</td>
<td>300</td>
</tr>
<tr>
<td>DOROTHHEA</td>
<td>DRUG</td>
<td>100000</td>
<td>800</td>
</tr>
<tr>
<td>SID00</td>
<td>GENOMICS</td>
<td>4932</td>
<td>12768</td>
</tr>
</tbody>
</table>
Online Streaming Feature Selection: A Submission to ICML 2010

Figure 4 The compactness and predictive accuracy of four algorithms (alpha=0.01)

Figure 5 The compactness and predictive accuracy of four algorithms (alpha=0.05)

Figure 6 The performance of OSFS with different values of alpha

Figure 7 The performance of Fast-OSFS with different values of alpha
Ovarian cancer and Breast-cancer data are bio-medical datasets (Conrads 2004; Wang 2005). Hiva, Nova and Sido0 datasets are from the WCCI 2006 and WCCI 2008 Performance Prediction Challenges, respectively. The other datasets are from the NIPS 2003 feature selection challenge. Ovarian cancer, Breast-cancer and Lymphoma used 10-fold cross validations; Sido0 data set used 1-fold cross validations; and the NIPS 2003 challenge data sets used their original training and validation sets.

![Figure 8](image)

With alpha equal to 0.01, Figure 8 gives the compactness and predictive accuracy (%) of the four algorithms on 10 challenge datasets. Our methods achieve both more compactness and higher accuracy than Alpha-investing, on 8 out of the 10 datasets. On the hiva dataset, our methods select fewer features and their accuracy is competitive with Alpha-investing. On the ovarian-cancer dataset, Alpha-investing selects more features to achieve the best accuracy. But on the Dexter dataset, Alpha-investing failed to select any features.

Compared with Grafting, Our methods have both better compactness and higher accuracy on 7 out of the 10 datasets. On the nova dataset, OSFS and Fast-OSFS achieve much higher accuracy, up to 0.926 and 0.966, respectively than Grafting with the accuracy up to 0.846, although they selected a little more features than Grafting. On ovarian-cancer and dexter data, OSFS selects many fewer features than Grafting, and its accuracy is also competitive with Grafting. Moreover, Grafting failed to select any features on dorothea and breast-cancer data because of out of memory.

On those challenge data sets, we can see that our methods also outperform Grafting and Alpha-investing on most of these data sets.

### 5.3 An Analysis of Time Complexity

Since the Grafting and Alpha-investing code used in the experiments was implemented in Matlab and our methods were written in the C language, a direct time comparison between them and our algorithms was not performed.

Although we had a theoretical analysis of time complexity for OSFS and Fast-OSFS above, a summary of the running time results of the execution of OSFS and Fast-OSFS is also given in Figure 9. The time reported is the normalized time which is the running time of OSFS for a data set divided by the corresponding running time of Fast-OSFS. Thus, a greater normalized running time than one implies that OSFS is slower than Fast-OSFS on the same learning task.

On the UCI data sets, the speed of Fast-OSFS is at least twice faster than that of OSFS. Since the running time of Fast-OSFS and OSFS is less than one second on most data sets, we only report the running time longer than ten seconds on five data sets in Figure 9.

![Figure 9](image)

On the challenge data sets, the selected features of Fast-OSFS are competitive with OSFS, and get a higher accuracy than OSFS on most of the datasets and are much faster than OSFS on all datasets.
5.4 Discussion

In this section, we further analyze four algorithms. Firstly, although Grafting is compared with our methods, it is a quasi-streaming feature selection algorithm. So we don’t further discuss it.

Secondly, in our experiments, our methods are more compact than Alpha-investing on all 18 datasets, and their accuracy is higher on most of these datasets. Therefore, our framework can manage streaming feature selection better than Alpha-investing on the original streaming features.

Thirdly, on all datasets, with regard to the compactness, Fast-OSFS is competitive with OSFS. As to the predictive accuracy, on UCI data sets with small features, OSFS is a little better than Fast-OSFS while on challenge datasets with tens of thousands of features, Fast-OSFS is superior to OSFS. Regarding the speed, Fast-OSFS is faster than OSFS. Thus, we can conclude that Fast-OSFS is more effective and efficient than OSFS on data sets with millions of potential features.

Finally, without any prior knowledge of the structure of the candidate features, our framework can deal with streaming features well. As discussed above, with prior knowledge, Alpha-investing can achieve good performance. If we have some prior knowledge, our framework can also deal with the task well. For example, with domain knowledge, features can be sorted so that we can place potentially more informative features earlier in the streaming features, making our methods easier to find strongly relevant and useful features. If strongly relevant features can be found earlier, the corresponding redundant features can be earlier eliminated in the online redundancy analysis step of our framework. Fast-OSFS will especially benefit from this sorting.

6. Conclusions

The novel features that distinguish our work from existing approaches are threefold: (1) our work further studies feature relevance and redefines feature redundancy so that the feature redundancy between a feature and a target is expressed explicitly; (2) based on feature relevance, we proposed a novel framework to manage the original streaming feature selection; and (3) two novel online streaming feature selection algorithms were presented in the paper. Compared with existing streaming feature selection methods, our methods have demonstrated their efficiency and effectiveness in supervised learning in domains where a dataset contains many irrelevant and/or redundant features.

Significant efforts are still needed to extend our framework. In our experiments, we stimulated the feature set with unknown, but a finite size. With an infinite size, how to dynamically assess the predictive accuracy, when reaching a certain threshold, and at this time, how to stop our algorithms are our future work.

Acknowledgements

This section is omitted for double blind reviewing.

References


Steven Loscalzo, Lei Yu et al. Consensus Group Based Stable Feature Selection. KDD 2009


Jing Zhou and Dean Foster. Streaming Feature Selection using Alpha-investing. KDD 2005


