# Online Feature Selection for Mining Big Data

Steven C.H. Hoi[*], Jialei Wang[*], Peilin Zhao[*], Rong Jin[†]
[*]School of Computer Engineering, Nanyang Technological University, Singapore
[†]Department of Computer Science and Engineering, Michigan State University, USA
{chhoi, JL.Wang, zhao0106}@ntu.edu.sg, rongjin@msu.edu

## ABSTRACT

Most studies of online learning require accessing all the attributes/features of training instances. Such a classical setting is not always appropriate for real-world applications when data instances are of high dimensionality or the access to it is expensive to acquire the full set of attributes/features. To address this limitation, we investigate the problem of **Online Feature Selection** (OFS) in which the online learner is only allowed to maintain a classifier involved a small and fixed number of features. The key challenge of Online Feature Selection is how to make accurate prediction using a small and fixed number of active features. This is in contrast to the classical setup of online learning where all the features are active and can be used for prediction. We address this challenge by studying sparsity regularization and truncation techniques. Specifically, we present an effective algorithm to solve the problem, give the theoretical analysis, and evaluate the empirical performance of the proposed algorithms for online feature selection on several public datasets. We also demonstrate the application of our online feature selection technique to tackle real-world problems of big data mining, which is significantly more scalable than some well-known batch feature selection algorithms. The encouraging results of our experiments validate the efficacy and efficiency of the proposed techniques for large-scale applications.

## Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications-data mining; I.2.6 [**Artificial Intelligence**]: Learning

## General Terms

Algorithms, Experimentation

## Keywords

Feature Selection, Online Learning, Classification

## 1. INTRODUCTION

Feature selection is an important topic in data mining and machine learning and has been studied for many years in literature [10, 19, 26, 31]. The objective of feature selection is to select a subset of relevant features for building effective prediction models. By removing irrelevant and redundant features, feature selection can improve the performance of prediction models by alleviating the effect of the curse of dimensionality, enhancing the generalization performance, speeding up the learning process, and improving the model interpretability. Feature selection has found applications in many domains, especially for the problems involved high dimensional data.

Despite being studied extensively, most existing studies on feature selection often assume the feature selection task is conducted in an off-line learning fashion and all the features of training instances are given a priori. Such assumptions may not always hold for some real-world applications where training examples may arrive in an online manner or it is expensive to collect the full information of training data. For example, in an online spam email detection system, training data usually arrive sequentially, making it difficult to deploy a regular batch feature selection technique in a timely, efficient, and scalable manner. Another example is feature selection in bioinformatics, where acquiring the entire set of features/attributes for every instance is expensive due to the high cost of conducting wet lab experiments.

Unlike the existing studies on feature selection, in this paper, we investigate the problem of *Online Feature Selection* (OFS) that aims to solve the feature selection problem by an online learning approach. The goal of online feature selection is to develop online classifiers that involve only a small and fixed number of features. In order to mine big data in real-world applications, we must be able to efficiently identify a fixed number of relevant features for building accurate prediction models in the online learning process. In this paper, we propose algorithms to solve the OFS task, analyze the theoretical property, and validate their empirical performance by extensive experiments. Finally, we apply our technique to solve applications with large-scale data sets.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 presents the problem and the proposed algorithms as well as the theoretical analysis. Section 4 discusses our empirical studies and Section 5 concludes this work.

## 2. RELATED WORK

Our work is closely related to the studies of online learning

*BigMine'12,* August 12, 2012 Beijing, China.

and feature selection in machine learning and data mining. Below we review some important related work in both areas.

For online learning studies, many techniques have been proposed for solving different tasks in literature [30, 6, 39, 20, 41, 25]. The most well-known method is the Perceptron algorithm [30, 16], which updates the model by adding a new example with some constant weight into the current set of support vectors when the incoming example is misclassified. Moreover, a lot of new online learning algorithms have been developed recently, in which many of them usually follow the criterion of maximum margin learning principle [17, 21, 6, 40, 34]. For example, the Passive-Aggressive algorithm [6] proposes to update a classifier that is near to the previous function while suffering less loss on the current instance. The PA algorithm is limited in that it only exploits the first order information, which has been addressed by the recently proposed confidence weighted online learning algorithms that exploit the second order information [14, 7, 8]. Despite the extensive investigation, most studies of online learning exploit the full features. In contrast, we consider an online learning problem where the number of active features is fixed, a significantly more challenging problem than the conventional setup of online learning.

Feature Selection (FS) has been actively studied. The goal of FS is to select the most relevant features from the whole feature space in order to improve the prediction performance of the predictors.Various FS methods have been proposed. Based on the selection criterion choice, these methods can be roughly divided into three categories: *Filter methods*, *Wrapper methods*, and *Embedded methods* approaches. Filter methods [38, 9, 1] relies on the characteristics of the data such as correlation, distance and information, without involving any learning algorithm; wrapper methods [22] require one predetermined learning algorithm for evaluating the performance of selected features set. Generally, wrapper methods will search the features suitable for the predetermined learning algorithm to improve the performance, but will be more computationally expensive; while Embedded methods [2, 5, 37] aim to integrate the feature selection process into the model training process, which are faster than the wrapper methods and still provide suitable feature subset for the learning algorithm, but those resultant features may be not suitable for other learning algorithms.

It is important to distinguish online feature selection, the problem addressed in this work, from Online Feature Selection or Streaming Feature Selection studied in [29, 18, 36]. In these works, features are assumed to arrive one at a time while all the training instances are available before the learning process starts. Their goal is to select a subset of features and return an appropriate model at each time step given the features observed so far. In contrast, we focus on online learning where training instances arrive sequentially.

The proposed work is closely related to sparse online learning [15, 24], whose goal is to learn a sparse linear classifier. Our work differs from these studies in that we impose a hard constraint on the number of non-zero elements in classifier $\mathbf{w}$, while all the studies of sparse online learning only have soft constraints on the sparsity of the classifier. Our work is also related to budget online learning [3, 11, 27, 42] where the number of support vectors is bounded by a predefined number. A common strategy behind many budget online learning algorithms is to remove the “oldest” support vector when the maximum number of support vectors is reached. This simple strategy however is not applicable to online feature selection. Finally, our work is closely related to the online learning algorithm that aims to learn a classifier that performs as well as the best subset of experts [35], which is in contrast to most online learning work on prediction with expert advice that only compares to the best expert in the ensemble [4]. Unlike the work [35] where only positive weights are assigned to individual experts, in this study, the weights assigned to individual features can be both negative and positive, making it more flexible.

# 3. ONLINE FEATURE SELECTION

## 3.1 Problem Setting

In this paper, we consider the problem of online feature selection for binary classification. Let $\{(\mathbf{x}_t, y_t) | \; t = 1, \ldots, T\}$ be a sequence of input patterns received over the trials, where each $\mathbf{x}_t \in \mathbb{R}^d$ is a vector of $d$ dimension and $y_t \in \{-1, +1\}$. In our study, we assume that $d$ is a large number and for computational efficiency we need to select a relatively small number of features for linear classification. More specifically, in each trial $t$, the learner presents a classifier $\mathbf{w}_t \in \mathbb{R}^d$ that will be used to classify instance $\mathbf{x}_t$ by a linear function $\mathrm{sgn}(\mathbf{w}_t^\top \mathbf{x}_t)$. Instead of using all the features for classification, we require the classifier $\mathbf{w}_t$ to have at most $B$ non-zero elements, i.e.,

$$\|\mathbf{w}_t\|_0 \leq B$$

where $B > 0$ is a predefined constant, and consequently at most $B$ features of $\mathbf{x}_t$ will be used for classification. We refer to this problem as online feature selection. Our goal is to design an effective strategy for online feature selection that is able to make a small number of mistakes. Throughout the paper, we assume $\|\mathbf{x}_t\|_2 \leq 1, t = 1, \ldots, T$.

## 3.2 Algorithms

In this problem, we assume the learner is provided with the full inputs of every training instance (i.e. $\mathbf{x}_1, \ldots, \mathbf{x}_T$)

### *3.2.1 A Simple Truncation Approach*

A straightforward approach to online feature selection is to modify the Perceptron algorithm by applying truncation. Specifically, In the $t$-th trial, when being asked to make prediction, we will truncate the classifier $\mathbf{w}_t$ by setting everything but the $B$ largest (absolute value) elements in $\mathbf{w}_t$ to be zero. This truncated classifier, denoted by $\mathbf{w}_t^B$, is then used to classify the received instance $\mathbf{x}_t$. Similar to the Perceptron algorithm, when the instance is misclassified, we will update the classifier by adding the vector $y_t \mathbf{x}_t$ where $(\mathbf{x}_t, y_t)$ is the misclassified training example. Algorithm 1 shows the steps of this approach.

Unfortunately, this simple approach does not work: it can not guarantee a small number of mistakes. To see this, consider the case where the input pattern $\mathbf{x}$ can only take two possible patterns, either $\mathbf{x}_a$ or $\mathbf{x}_b$. For $\mathbf{x}_a$, we set its first $B$ elements to be one and the remaining elements to be zero. For $\mathbf{x}_b$, we set its first $B$ elements to be zero and the remaining elements to be ones. An instance $\mathbf{x}$ is assigned to the positive class (i.e., $y = +1$) when it is $\mathbf{x}_a$, and assigned to the negative class (i.e., $y = -1$) when it is $\mathbf{x}_b$. Let $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_{2T}, y_{2T})$ be a sequence of $2T$ examples, with $\mathbf{x}_{2k+1} = \mathbf{x}_a, y_{2k+1} = 1, k = 0, \ldots, T-1$ and

**Algorithm 1** Modified Perceptron by Truncation for OFS

```
Input
    • $B$: the number of selected features
Initialization
    • $\mathbf{w}_1 = 0$
for $t = 1, 2, \dots, T$ do
    Receive $\mathbf{x}_t$
    Make prediction $\mathrm{sgn}(\mathbf{x}_t^\top \mathbf{w}_t)$
    Receive $y_t$
    if $y_t \mathbf{x}_t^\top \mathbf{w}_t \le 0$ then
        $\widehat{\mathbf{w}}_{t+1} = \mathbf{w}_t + y_t \mathbf{x}_t$
        $\mathbf{w}_{t+1} = Truncate(\widehat{\mathbf{w}}_{t+1}, B)$
    else
        $\mathbf{w}_{t+1} = \mathbf{w}_t$
    end if
end for
```

**Algorithm 2** $\mathbf{w} = Truncate(\widehat{\mathbf{w}}, B)$

```
if $\|\widehat{\mathbf{w}}\|_0 > B$ then
    $\mathbf{w} = \widehat{\mathbf{w}}^B$ where $\widehat{\mathbf{w}}^B$ is $\widehat{\mathbf{w}}$ with everything but the $B$
    largest elements set to zero.
else
    $\mathbf{w} = \widehat{\mathbf{w}}$
end if
```

$\mathbf{x}_{2k} = \mathbf{x}_b, y_{2k} = -1, k = 1, \dots, T$. It is clear that Algorithm 1 will always make the mistake while a simple classifier that uses only two attributes (i.e., the first feature and the $B+1$ feature) will make almost no mistakes.

### 3.2.2 A Sparse Projection Approach

One reason for the failure of Algorithm 1 is that although it selects the $B$ largest elements for prediction, it does not guarantee that the numerical values for the unselected attributes are sufficiently small, which could potentially lead to many classification mistakes. We can avoid this problem by exploring the sparsity property of $L1$ norm, given in the following proposition from [13].

PROPOSITION 1. *For $q > 1$ and $\mathbf{x} \in \mathbb{R}^d$, we have*

$$\|\mathbf{x} - \mathbf{x}^m\|_q \le \xi_q \|\mathbf{x}\|_1 (m+1)^{1/q-1}, m = 1, \dots, d$$

*where $\xi_q$ is a constant depending only on $q$ and $\mathbf{x}^m$ stands for the vector $\mathbf{x}$ with everything but the $m$ largest elements set to 0.*

This proposition indicates that when a vector $\mathbf{x}$ lies in a $L1$ ball, most of its numerical values are concentrated in its largest elements, and therefore removing the smallest elements will result in a very small change to the original vector measured by the $L_q$ norm. Thus, we will enforce the classifier to be restricted to a $L1$ ball, i.e.,

$$\Delta_R = \{\mathbf{w} \in \mathbb{R}^d : \|\mathbf{w}\|_1 \le R\} \qquad (1)$$

Based on this idea, we present a new approach in Algorithm 3 for Online Feature Selection (OFS). The online learner maintains a linear classifier $\mathbf{w}_t$ that has at most $B$ non-zero elements. When a training instance $(\mathbf{x}_t, y_t)$ is misclassified, the classifier is updated by online gradient descent and then projected; since after projection, the norm of the classifier could be bound, which is inspired by the Pegasos algorithm [32]. If the resulting classifier $\widehat{\mathbf{w}}_{t+1}$ has more than $B$ non-zero elements, we will simply keep the $B$ elements in $\widehat{\mathbf{w}}_{t+1}$ with the largest absolute weights.

**Algorithm 3** OFS via Sparse Projection. (**OFS**)

```
Input
    • $\lambda$: regularization parameter
    • $\eta$: step size
    • $B$: the number of selected features
Initialization
    • $\mathbf{w}_1 = 0$
for $t = 1, 2, \dots, T$ do
    Receive $\mathbf{x}_t$
    Make prediction $\mathrm{sgn}(\mathbf{w}_t^\top \mathbf{x}_t)$
    Receive $y_t$
    if $y_t \mathbf{w}_t^\top \mathbf{x}_t \le 1$ then
        $\widetilde{\mathbf{w}}_{t+1} = (1 - \lambda\eta)\mathbf{w}_t + \eta y_t \mathbf{x}_t$
        $\widehat{\mathbf{w}}_{t+1} = \min\{1, \frac{\frac{1}{\sqrt{\lambda}}}{\|\widetilde{\mathbf{w}}_{t+1}\|_2}\}\widetilde{\mathbf{w}}_{t+1}$
        $\mathbf{w}_{t+1} = Truncate(\widehat{\mathbf{w}}_{t+1}, B)$
    else
        $\mathbf{w}_{t+1} = (1 - \lambda\eta)\mathbf{w}_t$
    end if
end for
```

The following theorem gives the mistake bound of Algorithm 3.

THEOREM 1. *Let $\ell(z)$ be a convex loss function decreasing in z, with $|\ell'(1)| \ge G$ and $\ell(0) = 1$. After running Algorithm 3 over a sequence of training examples $(\mathbf{x}_1, y_1), \dots, (\mathbf{x}_T, y_T)$ with $\|\mathbf{x}_t\|_2 \le 1$ and $\mathbf{x}_t \in \mathbb{R}^d, t \in [T]$, we have the following the bound for the number of mistakes $M$ made by Algorithm 3*

$$M \le \frac{1}{\Omega}\left\{\min_{\mathbf{w} \in \Delta_{\sqrt{d}/\lambda}} \|\mathbf{w}\|_2^2 + 2\sum_{t=1}^{T} \ell(y_t \mathbf{w}^\top \mathbf{x}_t)\right\}$$

$$\Omega = 2\eta - \eta^2 G^2 - \frac{4\xi_2 d}{\sqrt{B+1}\lambda} - \frac{\xi_2^2 d}{\lambda(B+1)}$$

PROOF. At trial $t$ for any classifier $\mathbf{w} \in \Delta_{\sqrt{d/\lambda}}$,

$$\begin{aligned}
\Delta_t &= \|\mathbf{w} - \mathbf{w}_t\|_2^2 - \|\mathbf{w} - \mathbf{w}_{t+1}\|_2^2 \\
&= \|\mathbf{w} - \mathbf{w}_t\|_2^2 - \|\mathbf{w} - \widehat{\mathbf{w}}_{t+1} + \widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\|_2^2 \\
&= \|\mathbf{w} - \mathbf{w}_t\|_2^2 - \|\mathbf{w} - \widehat{\mathbf{w}}_{t+1}\|_2^2 \\
&\quad -2\langle \mathbf{w} - \widehat{\mathbf{w}}_{t+1}, \widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\rangle_2 - \|\widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\|_2^2
\end{aligned}$$

Denote $\widetilde{\mathbf{w}}_{t+1} = (1 - \lambda\eta)\mathbf{w}_t + \eta y_t \mathbf{x}_t$. Then according to Generalized pythagorean inequality [4],

$$\|\mathbf{w} - \widetilde{\mathbf{w}}_{t+1}\|_2^2 \ge \|\mathbf{w} - \widehat{\mathbf{w}}_{t+1}\|_2^2 + \|\widehat{\mathbf{w}}_{t+1} - \widetilde{\mathbf{w}}_{t+1}\|_2^2$$

As a result, we have

$$\begin{aligned}
&\|\mathbf{w} - \mathbf{w}_t\|_2^2 - \|\mathbf{w} - \widehat{\mathbf{w}}_{t+1}\|_2^2 \ge \|\mathbf{w} - \mathbf{w}_t\|_2^2 - \|\mathbf{w} - \widetilde{\mathbf{w}}_{t+1}\|_2^2 \\
&\ge 2\eta\langle \nabla\ell(y_t \mathbf{w}_t^\top \mathbf{x}_t), \mathbf{w} - \mathbf{w}_t\rangle - \eta^2\|\nabla_t\|^2 \\
&\ge 2\eta(\ell(y_t \mathbf{w}_t^\top \mathbf{x}_t) - \ell(y_t \mathbf{w}^\top \mathbf{x}_t)) - \eta^2 G^2
\end{aligned}$$

When the number of non-zero elements in $\widehat{\mathbf{w}}_{t+1}$ is more than $B$, we will generate $\mathbf{w}_{t+1}$ by only keeping the $B$ largest elements in $\widehat{\mathbf{w}}_{t+1}$. Since $\|\widehat{\mathbf{w}}_{t+1}\|_2 \le \frac{1}{\sqrt{\lambda}}$, then$\|\widehat{\mathbf{w}}_{t+1}\|_1 \le \frac{\sqrt{d}}{\sqrt{\lambda}}$. Using Proposition 1, we have

$$\begin{aligned}
&\langle \mathbf{w} - \widehat{\mathbf{w}}_{t+1}, \widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\rangle_2 \le \|\mathbf{w} - \widehat{\mathbf{w}}_{t+1}\|_2 \|\widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\|_2 \\
&\le \|\mathbf{w} - \widehat{\mathbf{w}}_{t+1}\|_1 \|\widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\|_2 \le \frac{2\xi_2 d}{\sqrt{B+1}\lambda}
\end{aligned}$$

and

$$\|\widehat{\mathbf{w}}_{t+1} - \mathbf{w}_{t+1}\|_2^2 \leq \frac{\xi_2^2 d}{\lambda(B+1)}$$

We thus have

$$\Delta_t \geq 2\eta(\ell(\mathbf{w}_t) - \ell(\mathbf{w})) - \eta^2 G^2 - \frac{4\xi_2 d}{\sqrt{B+1}\lambda} - \frac{\xi_2^2 d}{\lambda(B+1)}$$

We complete the proof by adding up the inequalities of all trials and using the fact $\ell(y_t \mathbf{w}_t^\top \mathbf{x}_t) \geq 1$ when $y_t \mathbf{w}_t^\top \mathbf{x}_t \leq 0$. □

# 4. EXPERIMENTAL RESULTS

In this section, we conduct an extensive set of experiments to evaluate the performance of the proposed online feature selection algorithms. We will evaluate the online predictive performance of the OFS task on several benchmark datasets from UCI machine learning repository, and then demonstrate the applications of the proposed online feature selection technique for real-world applications by comparing the proposed OFS technique with some well-known and successful batch feature selection technique in literature [28]. Finally, we also examine the scalability of the proposed technique for mining large-scale data sets.

## 4.1 Experiment I: Evaluation of Online Learning Performance on UCI Data Sets

### *4.1.1 Experimental Testbed on UCI Datasets*

To extensively examine the performance, we test the algorithms on a number of public datasets from web machine learning repositories. All of the datasets can be downloaded either from LIBSVM website [1] or UCI machine learning repository [2]. Table 1 shows a list of six binary datasets used in our experiments, which were chosen randomly.

**Table 1: List of UCI machine learning datasets used in our experiments.**

| Dataset | Number | Dimension |
|---|---|---|
| magic04 | 19020 | 10 |
| svmguide3 | 1243 | 21 |
| german | 1000 | 24 |
| splice | 3175 | 60 |
| spambase | 4601 | 57 |
| a8a | 32561 | 123 |

### *4.1.2 Experimental Setup and Compared Algorithms*

We compare the proposed OFS algorithm with two baselines: (i) the modified perceptron by simple truncation in Algorithm 1, denoted as "$PE_{trun}$", and (ii) a randomized feature selection algorithm, which randomly selects a fixed number of active features in an online learning task, denoted "RAND" for short.

To make a fair comparison, all algorithms adopt the same experimental settings. We set the number of selected features as $round(0.1 * dimensionality)$ for every dataset, the regularization parameter $\lambda$ to 0.01, and the learning rate $\eta$ to 0.2. All parameters were chosen by the same approach. After that, all the experiments were conducted over 20 random permutations for each dataset. All the experimental results were reported by averaging over these 20 runs.

### *4.1.3 Evaluation of Online Predictive Performance*

Table 8 summarizes the online predictive performance of the compared algorithms with a fixed fraction of selected features (10% of all dimensions) on the datasets.

**Table 2: Evaluation of the average number of mistakes by three algorithms on the six datasets.**

| Algorithm | svmguide3 | german | magic04 |
|---|---|---|---|
| RAND | 567.6 ± 17.3 | 472.4 ± 11.1 | 8689.8 ±58.9 |
| PE$_{trun}$ | 512.2 ± 32.6 | 489.6 ± 29.8 | 8153.1 ±79.3 |
| OFS | **400.9 ± 66.8** | **432.8 ± 13.6** | **6023.4 ±1342.3** |
| Algorithm | splice | spambase | a8a |
| RAND | 1517.0 ±25.7 | 1827.7 ±45.2 | 15610.7 ±78.8 |
| PE$_{trun}$ | 1039.9 ±35.0 | 1294.8± 66.3 | 14086.8 ±300.4 |
| OFS | **735.4 ±68.3** | **913.1 ± 157.8** | **9424.4 ±2545.8** |

Several observations can be drawn from the results. First of all, we found that among all the compared algorithms, the RAND algorithm has the highest mistake rate for all the cases. This shows that it is important to learn the active features in an OFS task. Second, we found that the simple "$PE_{trun}$" algorithm can outperform the RAND algorithm considerably, which indicates the algorithm is effective to choose informative features for online learning tasks. Finally, among the three algorithms, we found that the OFS algorithm achieved the smallest mistake rate, which is significantly smaller than the two algorithms. This shows that the proposed algorithm is able to considerably boost the performance of the simple "$PE_{trun}$" approach.

To further examine the online predictive performance, Figure 1 shows the details of online average mistake rates varying accord the entire OFS process on the three randomly chosen datasets (similar observations can be found on the other three datasets, we simply omit them due to space limitation). Similar to the previous observations, we can see that the proposed OFS algorithm consistently surpassed the other two algorithms for all the situations. This again verifies the efficacy of the proposed OFS algorithm.

Finally, Figure 2 further shows the details of the online performance of the compared online feature selection algorithms with varied fractions of selected features. The proposed OFS algorithm outperform the other two baselines for most cases. This encouraging result further verifies the efficacy of the proposed technique.

## 4.2 Experiment II: Online vs. Batch Comparison for Real-world Applications

In this experiment, we apply the proposed OFS technique to tackle feature selection tasks of real-world applications in computer vision and Bioinformatics.

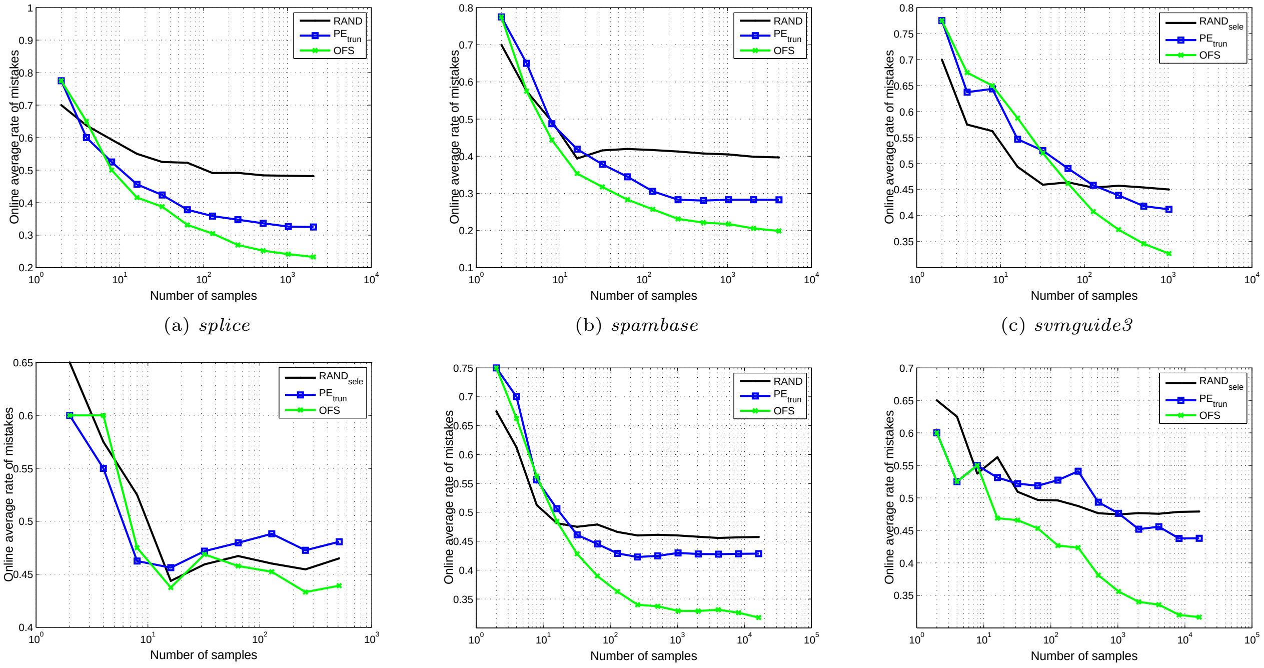
### *4.2.1 Experimental Datasets and Setup*

The first application is to solve feature selection for image classification. We adopt the CIFAR-10 image dataset [23] [3],

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/`

[2] `http://www.ics.uci.edu/~mlearn/MLRepository.html`

[3] `http://www.cs.toronto.edu/~kriz/cifar.html`



(a) *splice*

(b) *spambase*

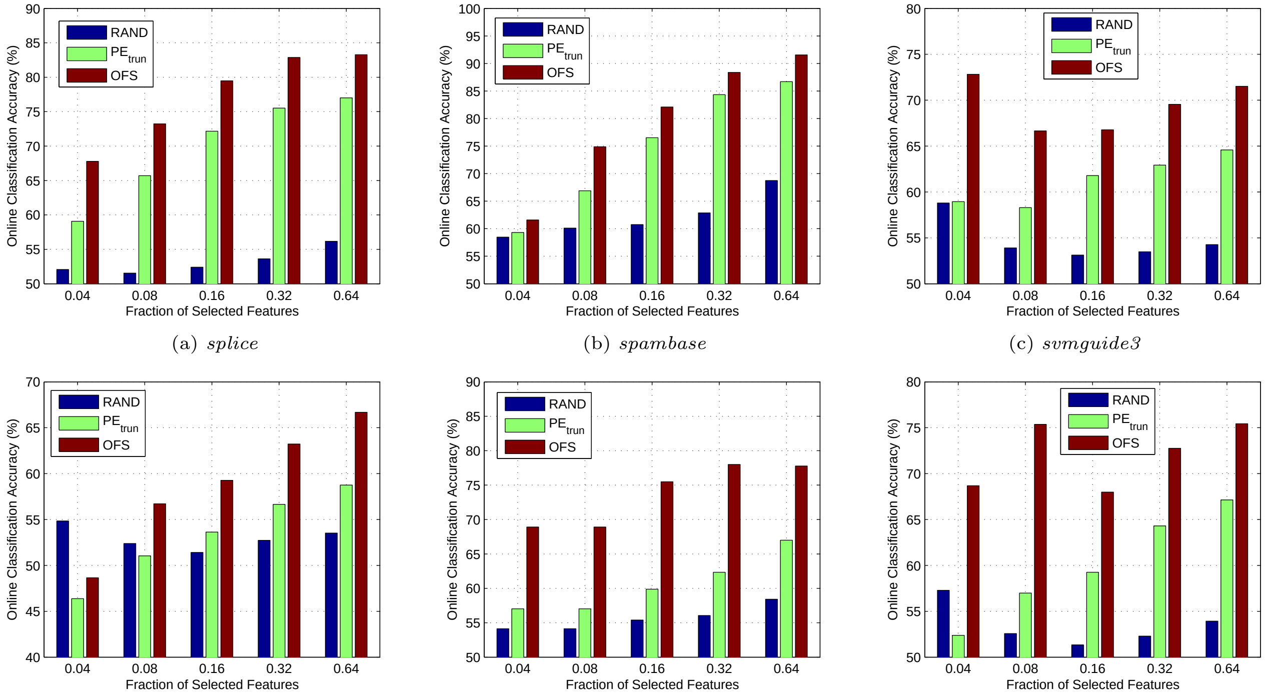
(c) *svmguide3*

(d) *german*

(e) *magic04*

(f) *a8a*

**Figure 1: Performance evaluation of online feature selection in the online learning process.**



(a) *splice*

(b) *spambase*

(c) *svmguide3*

(d) *german*

(e) *magic04*

(f) *a8a*

**Figure 2: Online Classification Accuracy with various fractions of selected features.**

which consists of 10 classes of images, which were a subset of the well-known 80-million images. In this experiment, we randomly choose two classes "airplane" and "bird" to a binary classification task. In our dataset, the CIFAR-10 dataset consists of a total of 3,992 images and each image is represented by a 3073-dimensional feature vector.

**Table 3: Evaluation of online mistake rates on CIFAR-10. $\rho$ is the fraction of selected features.**

| $\rho$ | RAND | PE$_{trun}$ | OFS |
|---|---|---|---|
| 0.01 | 0.471 ±0.006 | 0.367 ± 0.009 | **0.323 ± 0.020** |
| 0.02 | 0.466 ±0.006 | 0.340 ±0.006 | **0.309 ±0.017** |
| 0.04 | 0.466 ±0.006 | 0.321 ±0.006 | **0.288 ±0.012** |
| 0.08 | 0.464 ±0.008 | 0.306 ±0.003 | **0.267 ±0.008** |
| 0.16 | 0.461 ±0.006 | 0.295 ±0.004 | **0.252 ±0.004** |
| 0.32 | 0.455 ±0.009 | 0.291 ±0.005 | **0.238 ±0.003** |
| 0.64 | 0.436 ±0.008 | 0.288 ±0.005 | **0.226 ± 0.001** |

**Table 4: Evaluation of online mistake rates on "Colon". $\rho$ is the fraction of selected features.**

| $\rho$ | RAND | PE$_{trun}$ | OFS |
|---|---|---|---|
| 0.01 | 0.496 ±0.061 | 0.406 ±0.060 | **0.362 ± 0.068** |
| 0.02 | 0.485 ±0.075 | 0.391 ±0.048 | **0.325 ±0.053** |
| 0.04 | 0.512 ±0.050 | 0.366 ±0.045 | **0.337 ±0.066** |
| 0.08 | 0.496 ±0.056 | 0.361 ±0.046 | **0.334 ±0.053** |
| 0.16 | 0.506 ±0.075 | 0.375 ±0.052 | **0.341 ±0.055** |
| 0.32 | 0.508 ±0.060 | 0.372 ±0.035 | **0.347 ±0.055** |
| 0.64 | 0.457 ±0.046 | 0.391 ±0.040 | **0.350 ±0.059** |

The second application is to solve feature selection of microarray gene expression data in bioinformatics. We adopt the Colon dataset, which is a microarray gene expression data of tumor and normal colon tissues [33] [4]. The dataset has 62 samples and each sample contains 2000 gene features.

The parameter settings are the same as the previous section. All the experiments were conducted over 20 random permutations for each of the two datasets. All the results were reported by averaging over these 20 runs.

### 4.2.2 *Evaluation of Online Prediction Performance*

The experimental result is shown in Table 3 and Table 4. Note that the average mistake rates of the original online algorithm with full set of features on CIFAR-10 and Colon datasets are 0.2266±0.0030 and 0.3548±0.0541, respectively.

Several observation could be drawn. Firstly, in both datasets our OFS algorithms performs significantly better than the random feature selection method and the simple truncation approach, which demonstrate the effectiveness of our algorithms; Secondly, in CIFAR-10 dataset, our OFS algorithm performs better as the fraction of features selected increase, and obtain the same performance with OGD when the fraction reach 64%, in Colon dataset, the performance, as the fraction of features used increase, first improve then drop, and achieves the best performance when selecting 2% of the features, which is superior to using all the features, this may be because there exists some noise features in Colon dataset which may affect the performance.

### 4.2.3 *Online vs. Batch Feature Selection Methods*

All the above experiments are conducted in an online learning setting. It will also be useful to compare the proposed algorithm against some existing batch feature selection method. To this purpose, we compare our OFS algorithm with a state-of-the-art batch feature selection method: minimum Redundancy Maximum Relevance Feature Selection (mRMR) [28, 12].

We divide the datasets equally into two parts: the first part is used by running feature selection algorithms (OFS and mRMR), and the second part is used to test the performance of the feature selection algorithms. To examine the efficacy of the selected features invariant to different classifiers, we adopt two types of widely used classifiers: (i) Online gradient descent (OGD) which is an online learning classifier, and (ii) K-nearest neighbor classifier (KNN), which is a batch learning classifier. In this experiment, we simply fix $K = 5$ for the parameter $K$ in the KNN classifier. We evaluate the performance in terms of both the classification error rates and the computational time efficiency of the two different feature selection algorithms. The experimental results were shown in Table 5 and Table 6, respectively, and the time efficiency comparison was shown in Figure 4.

From the experimental results, we can see that the proposed OFS algorithm in general outperforms mRMR for most cases in terms of classification efficacy for both different classifiers. In terms of time efficiency and scalability, we observe that the OFS algorithm has a significant advantage over the batch feature selection algorithm, especially when the number of features to be selected is large. For example, when choosing 32% of features on the CIFAR-10 dataset, the mRMR algorithm spent about 1045.83 seconds for learning, while the proposed OFS algorithm took only 1.08 seconds, which is almost 1000 times faster.

## 4.3 Experiment III: Evaluation on Big Data

In this section, we evaluate the performance of the proposed algorithms for mining big data sets, in which each of these data sets contains at least 100,000 instances. The statistics of these data sets are shown in table 7.

**Table 7: List of big datasets used in our experiments.**

| Dataset | Number | Dimension |
|---|---|---|
| KDDCUP08 | 102294 | 117 |
| ijcnn1 | 141691 | 22 |
| codrna | 271617 | 8 |
| covtype | 581012 | 54 |

Table 8 shows the experimental results of the average numbers of mistakes and the time costs by three different algorithms, in which the experiments were implemented in matlab and run in a regular PC with a Dual-core CPU. From the results, it is clear to see that the proposed OFS algorithm significantly outperforms the other two baselines. In addition, by examining the time costs, we found that all the three online algorithms are very efficient, which generally require only a few minutes in learning the feature selection on these large-scale data sets.

## 5. CONCLUSIONS

In this paper, we investigated a new research problem of **Online Feature Selection** (OFS), which aims to select a fixed number of features for prediction by an online learning

[4] `http://genomics-pubs.princeton.edu/oncology/affydata/index.html`

**Table 5: Evaluation of the classification error rates with different classifiers on Colon.**

| OGD classifier | 0.005 | 0.01 | 0.02 | 0.04 | 0.08 | 0.16 | 0.32 |
|---|---|---|---|---|---|---|---|
| mRMR | 0.464 ± 0.146 | **0.400 ± 0.055** | 0.361 ± 0.067 | 0.358 ± 0.075 | 0.387 ± 0.076 | 0.393 ± 0.083 | 0.380 ± 0.058 |
| OFS | **0.441 ± 0.088** | 0.409 ± 0.127 | **0.348 ± 0.062** | **0.306 ± 0.065** | **0.345 ± 0.057** | **0.341 ± 0.066** | **0.332 ± 0.050** |
| KNN classifier | 0.005 | 0.01 | 0.02 | 0.04 | 0.08 | 0.16 | 0.32 |
| mRMR | 0.377 ± 0.066 | 0.341 ± 0.094 | 0.251 ± 0.042 | 0.280 ± 0.088 | 0.271 ± 0.055 | 0.319 ± 0.068 | 0.325 ± 0.110 |
| OFS | **0.345 ± 0.097** | **0.335 ± 0.107** | **0.212 ± 0.097** | **0.180 ± 0.120** | **0.222 ± 0.068** | **0.290 ± 0.068** | **0.287 ± 0.105** |

**Table 6: Evaluation of the classification error rates with different classifiers on CIFAR-10.**

| OGD classifier | 0.005 | 0.01 | 0.02 | 0.04 | 0.08 | 0.16 | 0.32 |
|---|---|---|---|---|---|---|---|
| mRMR | 0.437 ± 0.018 | 0.428 ± 0.009 | 0.427 ± 0.012 | 0.424 ± 0.012 | 0.364 ± 0.012 | 0.359 ± 0.017 | 0.342 ± 0.013 |
| OFS | **0.341 ± 0.028** | **0.318 ± 0.021** | **0.309 ± 0.018** | **0.295 ± 0.015** | **0.270 ± 0.012** | **0.254 ± 0.010** | **0.241 ± 0.008** |
| KNN classifier | 0.005 | 0.01 | 0.02 | 0.04 | 0.08 | 0.16 | 0.32 |
| mRMR | 0.325 ± 0.009 | 0.327 ± 0.014 | 0.327 ± 0.015 | 0.323 ± 0.011 | **0.238 ± 0.006** | 0.264 ± 0.013 | 0.256 ± 0.010 |
| OFS | **0.293 ± 0.013** | **0.273 ± 0.012** | **0.269 ± 0.012** | **0.259 ± 0.014** | 0.257 ± 0.010 | **0.253 ± 0.017** | **0.255 ± 0.007** |

**Table 8: Evaluation of the average number of mistakes by three algorithms on the big data sets and their time costs (as shown inside the parentheses).**

| Algorithm | KDDCUP08 | ijcnn1 |
|---|---|---|
| RAND | 50718.4 ± 210.2(20.8s) | 74778.4 ± 159.9(28.2s) |
| PE$_{trun}$ | 48714.4 ± 2654.8(16.8s) | 62282.6 ± 188.7(21.3s) |
| OFS | **31155.6 ± 2733.1**(44.3s) | **40512.8 ± 257.0**(51.7s) |
| Algorithm | codrna | covtype |
| RAND | 128709.0 ± 284.2(114.0s) | 277394.5 ±127.0(349.5s) |
| PE$_{trun}$ | 119838.4 ±165.8(93.4s) | 198661.6± 125.6(195.0s) |
| OFS | **78685.8 ±42.8**(183.4s) | **148223.1 ± 193.0**(601.5s) |

fashion. We presented a novel OFS algorithm to solve the learning task, and offered theoretical analysis on the mistake bound of the proposed OFS algorithm. We extensively examined their empirical performance on both regular UCI data datasets and large-scale datasets. We also compared the proposed online feature selection technique with a regular state-of-the-art batch feature selection algorithm for solving real-world applications: image classification in computer vision and microarray gene expression analysis in bioinformatics. Encouraging results show the proposed algorithms are fairly effective for feature selection tasks of online applications, and significantly more efficient and scalable than some state-of-the-art batch feature selection technique.

## Acknowledgements

This work was in part supported by MOE tier 1 grant (RG33/11) and Microsoft Research grant (M4060936).

## 6. REFERENCES

[1] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *Journal of Machine Learning Research*, 3:1183–1208, 2003.

[2] J. Bi, K. P. Bennett, M. J. Embrechts, C. M. Breneman, and M. Song. Dimensionality reduction via sparse support vector machines. *Journal of Machine Learning Research*, 3:1229–1243, 2003.

[3] G. Cavallanti, N. Cesa-Bianchi, and C. Gentile. Tracking the best hyperplane with a simple budget perceptron. *Machine Learning*, 69(2-3):143–167, 2007.

[4] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006.

[5] A. B. Chan, N. Vasconcelos, and G. R. G. Lanckriet. Direct convex relaxations of sparse svm. In *ICML*, pages 145–153, 2007.

[6] K. Crammer, O. Dekel, J. Keshet, S. Shalev-Shwartz, and Y. Singer. Online passive-aggressive algorithms. *J. Mach. Learn. Res. (JMLR)*, 7:551–585, 2006.

[7] K. Crammer, M. Dredze, and F. Pereira. Exact convex confidence-weighted learning. In *NIPS*, pages 345–352, 2008.

[8] K. Crammer, A. Kulesza, and M. Dredze. Adaptive regularization of weight vectors. In *NIPS*, pages 414–422, 2009.

[9] M. Dash and V. Gopalkrishnan. Distance based feature selection for clustering microarray data. In *DASFAA*, pages 512–519, 2008.

[10] M. Dash and H. Liu. Feature selection for classification. *Intell. Data Anal.*, 1(1-4):131–156, 1997.

[11] O. Dekel, S. Shalev-Shwartz, and Y. Singer. The forgetron: A kernel-based perceptron on a budget. *SIAM J. Comput.*, 37(5):1342–1372, 2008.

[12] C. H. Q. Ding and H. Peng. Minimum redundancy feature selection from microarray gene expression data. *J. Bioinformatics and Computational Biology*, 3(2):185–206, 2005.

[13] D. Donoho. Compressed sensing. *IEEE Trans. on Information Theory*, 52(4):1289 – 1306, 2006.

[14] M. Dredze, K. Crammer, and F. Pereira. Confidence-weighted linear classification. In *ICML*, pages 264–271, 2008.

[15] J. Duchi and Y. Singer. Efficient online and batch learning using forward backward splitting. *J. Mach. Learn. Res.*, 10:2899–2934, 2009.

[16] Y. Freund and R. E. Schapire. Large margin classification using the perceptron algorithm. *Machine Learning*, 37(3):277–296, 1999.

[17] C. Gentile. A new approximate maximal margin classification algorithm. *J. Mach. Learn. Res. (JMLR)*, 2:213–242, 2001.

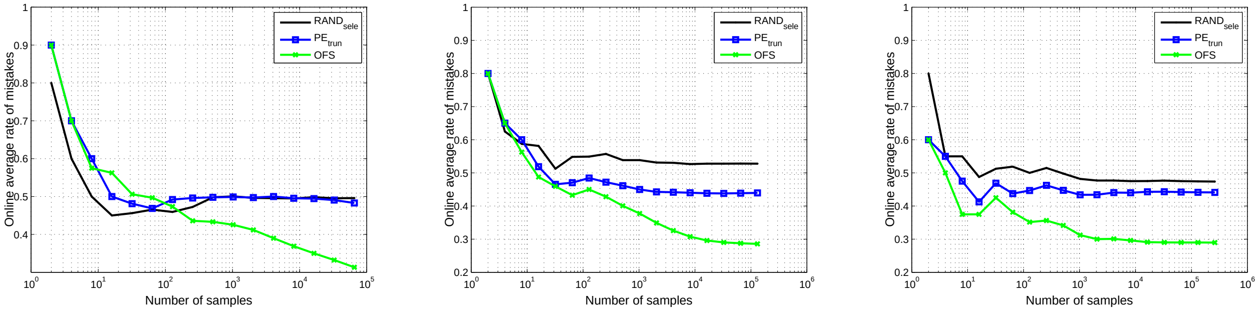
[18] K. A. Glocer, D. Eads, and J. Theiler. Online feature selection for pixel classification. In *ICML*, pages 249–256, 2005.

[19] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[20] R. Jin, S. C. H. Hoi, and T. Yang. Online multiple kernel learning: Algorithms and mistake bounds. In *ALT*, pages 390–404, 2010.

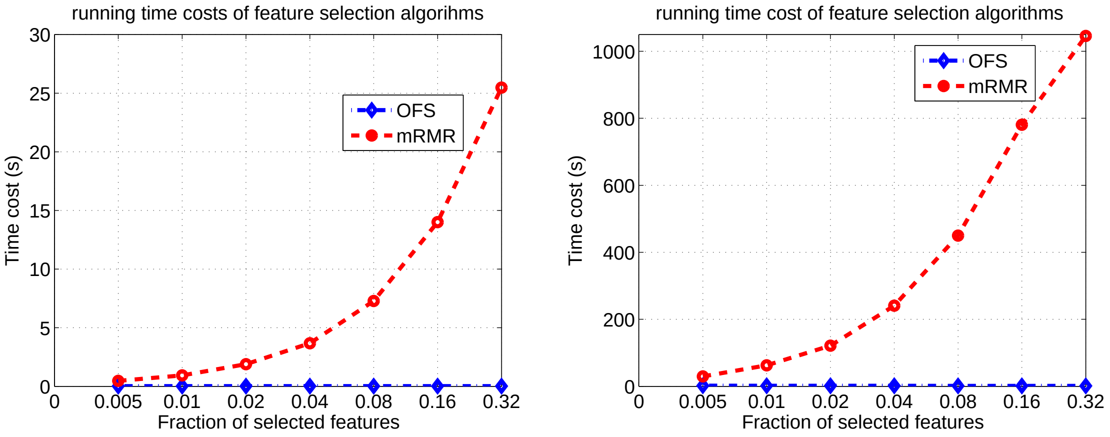
[21] J. Kivinen, A. J. Smola, and R. C. Williamson. Online learning with kernels. In *Advances in Neural Information Processing Systems*, pages 785–792, 2001.

[22] R. Kohavi and G. H. John. Wrappers for feature subset



(a) *KDDCUP08* (b) *ijcnn1* (c) *codrna*

**Figure 3: Performance evaluation of online feature selection on big data sets.**



(a) Colon (b) CIFAR-10

**Figure 4: Evaluation of time efficiency: online feature selection (OFS) v.s. batch feature selection (mRMR).**

selection. *Artif. Intell.*, 97(1-2):273–324, 1997.

[23] A. Krizhevsky. Learning multiple layers of features from tiny images. *Technical Report, University of Toronto*, 2009.

[24] J. Langford, L. Li, and T. Zhang. Sparse online learning via truncated gradient. *J. Mach. Learn. Res.*, 10:777–801, 2009.

[25] G. Li, K. Chang, S. C. H. Hoi, W. Liu, and R. Jain. Collaborative online learning of user generated content. In *CIKM*, pages 285–290, 2011.

[26] H. Liu and L. Yu. Toward integrating feature selection algorithms for classification and clustering. *IEEE Trans. Knowl. Data Eng.*, 17(4):491–502, 2005.

[27] F. Orabona, J. Keshet, and B. Caputo. The projectron: a bounded kernel-based perceptron. In *ICML*, pages 720–727, 2008.

[28] H. Peng, F. Long, and C. H. Q. Ding. Feature selection based on mutual information: Criteria of max-dependency, max-relevance, and min-redundancy. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(8):1226–1238, 2005.

[29] S. Perkins and J. Theiler. Online feature selection using grafting. In *ICML*, pages 592–599, 2003.

[30] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407, 1958.

[31] Y. Saeys, I. Inza, and P. Larrañaga. A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19):2507–2517, 2007.

[32] S. Shalev-Shwartz, Y. Singer, and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *ICML*, pages 807–814, 2007.

[33] U.Alon, N.Barkai, D.A.Notterman, K.Gish, S.Ybarra, D.Mack, and A.J.Levine. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. *Proceedings of the National Academy of Sciences*, pages 6745–6750, 1999.

[34] J. Wang, P. Zhao, and S. C. H. Hoi. Exact soft confidence-weighted learning. In *ICML*, 2012.

[35] M. K. Warmuth and D. Kuzmin. Randomized pca algorithms with regret bounds that are logarithmic in the dimension. In *Advances in Neural Information Processing Systems 19 (NIPS 06)*, 2006.

[36] X. Wu, K. Yu, H. Wang, and W. Ding. Online streaming feature selection. In *ICML*, pages 1159–1166, 2010.

[37] Z. Xu, R. Jin, J. Ye, M. R. Lyu, and I. King. Non-monotonic feature selection. In *ICML*, page 144, 2009.

[38] L. Yu and H. Liu. Feature selection for high-dimensional data: A fast correlation-based filter solution. In *ICML*, pages 856–863, 2003.

[39] P. Zhao and S. C. H. Hoi. Otl: A framework of online transfer learning. In *ICML*, pages 1231–1238, 2010.

[40] P. Zhao, S. C. H. Hoi, and R. Jin. Double updating online learning. *Journal of Machine Learning Research*, 12:1587–1615, 2011.

[41] P. Zhao, S. C. H. Hoi, R. Jin, and T. Yang. Online auc maximization. In *ICML*, pages 233–240, 2011.

[42] P. Zhao, J. Wang, P. Wu, R. Jin, and S. C. H. Hoi. Fast bounded online gradient descent algorithms for scalable kernel-based online learning. In *ICML*, 2012.