Streaming Feature Selection using Alpha-investing

Jing Zhou
Electrical and Systems Engineering
University of Pennsylvania
Philadelphia, PA 19104
jingzhou@seas.upenn.edu

Dean Foster
Statistics Department
University of Pennsylvania
Philadelphia, PA 19104
foster@wharton.upenn.edu

Robert Stine
Statistics Department
University of Pennsylvania
Philadelphia, PA 19104
stine@wharton.upenn.edu

Lyle Ungar
Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104
ungar@cis.upenn.edu

ABSTRACT

In Streaming Feature Selection (SFS), new features are sequentially considered for addition to a predictive model. When the space of potential features is large, SFS offers many advantages over traditional feature selection methods, which assume that all features are known in advance. Features can be generated dynamically, focusing the search for new features on promising subspaces, and overfitting can be controlled by dynamically adjusting the threshold for adding features to the model. We describe α-investing, an adaptive complexity penalty method for SFS which dynamically adjusts the threshold on the error reduction required for adding a new feature. α-investing gives false discovery rate-style guarantees against overfitting. It differs from standard penalty methods such as AIC, BIC or RIC, which always drastically over- or under-fit in the limit of infinite numbers of non-predictive features. Empirical results show that SFS is competitive with much more compute-intensive feature selection methods such as stepwise regression, and allows feature selection on problems with over a million potential features.

Categories and Subject Descriptors

G.3 [**Probability and Statistics**]: Statistical computing; I.5.2 [**Pattern Recognition**]: Design Methodology—*Feature evaluation and selection*

General Terms

Algorithms

KDD'05, August 21–24, 2005, Chicago, Illinois, USA.

Keywords

Classification, Multiple Regression, Feature Selection, False Discovery Rate

1. INTRODUCTION

In many predictive modeling tasks one has a fixed set of observations from which a vast (or even infinite) set of potentially predictive features can be computed. Of these, often only a small number are expected to be useful in the predictive model. Pairwise interactions and data transformations of an original set of features are frequently important in obtaining superior statistical models, but consequently expand the number of feature candidates while leaving the number of observations constant. For example, in a recent bankruptcy prediction study described below (see Section 6), pairwise interactions between the 365 original candidate features led to a set of over $67,000$ resultant candidate features, of which about 40 proved to be significant. The feature selection problem is to identify and include features from a candidate set with the goal of building a statistical model with minimal out-of-sample (test) error. As the set of potentially predictive features becomes ever larger, careful feature selection to avoid overfitting and to reduce computation time becomes ever more critical.

In this paper, we present *streaming feature selection* (SFS), a class of methods in which features are considered sequentially for addition to a model, and either added to the model or discarded. By modeling the candidate feature set as a dynamically generated stream, we can handle candidate feature sets of unknown, or even infinite size, since not all potential features need to be generated and tested. Enabling selection from a set of features of unknown size is useful in many settings. For example, in statistical relational learning [16, 8, 7], an agent may search over the space of SQL queries to augment the base set of candidate features found in the tables of a relational database. The number of candidate features generated by such a method is limited by the amount of CPU time available to run SQL queries. Generating 100,000 features can easily take 24 CPU hours[19],

while millions of features may be irrelevant due to the large numbers of individual words in text. Another example occurs in the generation of transformations of features already included in the model (e.g. pairwise or cubic interactions). When there are millions or billions of potential features, just generating the entire set of features(e.g. cubic interactions or three-way table merges in SQL) is often intractable. Traditional regularization and feature selection settings assume that all features are pre-computed and presented to a learner before *any* feature selection begins. SFS does not.

We present here both the general SFS approach and a simple algorithm, α-investing, that exploit the streaming feature setting to produce useful models. α-investing is motivated from a desire to control the false discovery rate (FDR) of added features. As described below, FDR measures the percentage of "spurious" features added to the model, i.e., the percentage of features that do not improve accuracy on a hypothetical infinite validation set, and indirectly controls the degree of overfitting, as measured by test error.

SFS with α investing applies to a wide variety of models where p-values or similar measures of feature significance are generated. We evaluate α-investing using linear and logistic regression, where a large variety of selection criteria have been developed and tested previously in traditional feature selection settings. Empirical evaluation shows that, as predicted by theory, α-investing provides superior models under the SFS setting compared to traditional feature selection penalty criteria including AIC [3], BIC [21], and RIC [6, 9].

Although, as noted above, SFS is designed for settings in which the feature set size is unknown, in order to compare it with stepwise regression, we applied SFS in traditional feature selection settings, i.e. those of fixed feature set size. In such settings, we find that SFS provides competitive performance to stepwise regression for smaller feature sets, and offers significant computational savings and higher prediction accuracy when the feature set size becomes moderately large.

2. TRADITIONAL FEATURE SELECTION: A BRIEF REVIEW

Traditional feature selection typically assumes a setting consisting of n observations and a fixed number p of candidate features. The goal is to select the feature subset that will ultimately lead to the best performing predictive model. The size of the search space is therefore 2^p, and identifying the best subset is NP-complete. Many commercial statistical packages offer variants of a greedy method,stepwise feature selection: an iterative procedure in which at each step the best possible feature is selected and added to the model. Stepwise regression thus performs hill climbing in the space of feature subsets. Stepwise selection is terminated when either all candidate features have been added, or none of the remaining features lead to increased expected benefit according to some measure, such as a p-value threshold. We will show that an even greedier search, in which each feature is considered only once (rather than at every step) gives competitive performance. Variants of stepwise selection abound, including forward (adding features deemed helpful) backwards (removing features no longer deemed helpful) and mixed methods that alternate. Our evaluation and discussion will assume a simple forward search.

Table 1: Symbols used throughout the paper and their definitions.

Symbol	Meaning
n	Number of observations
p	Number of candidate features
q	Number of features currently included in a model

Table 2: Different choices for the model complexity penalty F.

Name	Nickname	Penalty
Akaike Information Criterion	AIC	2
Bayesian Information Criterion	BIC	$\log(n)$
Risk Inflation Criterion	RIC	$2\log(p)$

There are many methods for assessing the benefit of adding a feature. Computer scientists tend to use cross-validation, where the training set is divided into several (say k) batches with equal sizes. $k-1$ of the batches are used for training while the remainder batch is used for evaluation. The training procedure is run k times so that the model is evaluated once on each of the batches and performance is averaged. The approach is computationally expensive, requiring k separate retraining steps for each evaluation. A second objection is that when observations are scarce the method does not make good use of the observations. Finally, when many different models are being considered (e.g. different combinations of features), there is a serious danger of over-fitting, when cross-validation is used. One, in effect, is selecting the model to fit the test set.

Penalized likelihood ratio methods [5] for feature selection are preferred to cross-validation by many statisticians, as they have an advantage in not requiring the multiple retraining of the model, and have attractive theoretical properties. Penalized likelihood can be represented as:

$$\text{score} = -2 * \log(\text{likelihood}) + F(n,p) * q \qquad (1)$$

where F is a function designed to penalize model complexity, q represents the number of features currently included in the model at a given point. We continue to denote the total number of candidate features as p and the number of observations in the training set as n. Table 1 contains these definitions which we use throughout the paper. The first term in the equation represents a measure of the in-sample error given the model, while the second is a model complexity penalty.

Using Equation 1, only features that decrease scores get added. In other words, the benefit of adding the feature to the model embodied by the likelihood ratio must surpass the penalty incurred by increasing the model complexity. We focus now on choice of F, where several methods have been developed and applied for regression. Many different functions F have been used, defining different criteria for feature selection. The most widely used of these criteria are the Akaike Information Criterion (AIC), the Bayesian Information Criterion (BIC), and the Risk Inflation Criterion (RIC). Table 2 summarizes the the penalties F used in these methods.

This paper presents SFS using α-investing. A virtually identical method can also be derived using a minimum description length formalism [22], giving what we have called

the information investing criterion (IIC) [23]. For exposition we find it useful to compare the different choices of F as alternative coding cost schemes for use in a minimum description length (MDL) criterion framework. Equation 1 can be viewed as the length of a message encoding a statistical model plus the training set response values given that model. A model encoding scheme for feature selection must identify which features are selected for inclusion in addition to encoding the parameters of the included features. Different choices for F correspond to different coding schemes for the model. In contrast the cost of encoding the training set response values is independent of F.

Generally, encoding schemes work better when they more optimally encode the model: producing the most accurate depiction of the model using the fewest number of bits. AIC corresponds to an encoding of each model parameter in two bits (F=2). BIC's choice of $F = \log(n)$ employs more bits to encode each parameter as the training set size grows larger. Using BIC, each zero parameter (feature not included in the model) is coded with one bit, and each non-zero parameter is coded with $1 + \frac{1}{2}\log(n)$ bits. (All logs are base 2.) Recalling that the log likelihood of the data given a model gives the number of bits to code the model error, leads to the BIC criterion for feature selection: accept a new feature x_i only if the change in log likelihood from adding the feature is greater than $\frac{1}{2}\log(n)$, i.e. if $\log(P(Y|\hat{Y}_i)) - \log(P(Y|\hat{Y}_{-i})) > \frac{1}{2}\log(n)$. BIC is equivalent to a Minimum Description Length (MDL)[20] criterion if the number of features considered, p is much less than the number of observations, n. However, BIC is not a valid code for $p \gg n$.

Both AIC and BIC methods suffer as p grows larger than n, a situation common in SFS settings. We confirm this theory through empirical investigation in Section 6.

RIC corresponds to a penalty of $F = 2 * log(p)$ [12]. Though the criteria is motivated by a minimax argument, following [22] we can view RIC as an encoding scheme where $\log p$ bits encode the index of which feature was added. Such an encoding is most efficient when few of the p candidates enter the model.

RIC can be problematic in SFS settings since RIC requires that we know p in advance, which is often not the case (See Section 3). We are forced to guess a likely p, and when our guess is inaccurate, the method may be too stringent or not stringent enough. By plugging $F = 2\log p$ into Equation 1 and examining the resulting chi-squared hypothesis test, it can be shown that the p-value required to reject the null hypothesis must at least be smaller than $\frac{0.05}{p}$. In other words, RIC may be viewed as a Bonferroni p-value thresholding method. Bonferroni methods are known to be overly stringent [4], a problem exacerbated in SFS applications when p should technically be chosen to be the largest number of features that might be examined. On the other hand, if p is picked to be a lower bound of the number of predictors that might be examined, then it is too small and there is increased risk that some feature will appear by chance to give significant performance improvement.

3. INTERLEAVING FEATURE GENERATION AND TESTING

In streaming feature selection (SFS), candidate features are sequentially presented to the modeling code for potential inclusion in the model. As each feature is presented, a decision is made using an adaptive penalty scheme as to whether or not to include the feature in the model. Each feature need be examined at most once.

The "streaming" view generalizes the process by which features are generated and the order in which they are presented. For instance, one implementation may allow cyclical streams that loop through previously dismissed features, while others may allow features to only be presented to the selection strategy once. In our experiments, we found that using a second pass through the features was not helpful for our real data sets. Most importantly, the stream structure does not assume a fixed number of features, or even that the feature stream be determined in advance. Features can be generated dynamically based on which features have already been added to the model.[1] Note that the theory provided below is independent of the feature generation scheme used. All that is required is a method of generating features, and an estimation package which given a proposed feature for addition to the model returns a p-value for the corresponding coefficient or, more generally, the change in likelihood of the model resulting from adding the feature.

For concreteness, consider a binary classification task to be modeled using logistic regression with feature selection. The data set has n observations and p original predictors. In addition to the p original predictors, there are p^2 pairwise interaction terms formed by multiplying all p^2 pairs of features together (*e.g.* $x_1^2, x_1 \cdot x_2, \ldots$). (Almost half of these features are, of course, redundant with the other half due to symmetry, and so need not be generated and tested.) We refer to the p^2 interaction terms as *generated* predictors, and consider them as members of a more general class of predictors formed from transformations of the original features (square root, log, *etc.*), or combinations of them including, for instance, principle components analysis (PCA) or other unsupervised clustering of the observations. Such strategies are frequently successful in obtaining better predictive models.

In recent years, advances in computing power and data storage technology have lead to novel methods of feature generation. For example, Statistical Relational Learning (SRL) methods often generate tens or hundreds of thousands of potentially predictive features. SRL and related methods "crawl" through a database or other relational structure and generate features by building increasingly complex compound relations [16, 8, 7]. For example, when building a model to predict the journal in which an article will be published, potentially predictive features include the words in the target article itself, the words in the articles cited by the target article, the words in articles that cite articles written by the authors of the target article, and so forth.

Traversing such relational structures can easily generate several billion features, since there are many words, authors, and journals. A hundred billion numbers do not fit easily into memory on most contemporary computers, and so computing all possible features in advance is ill-advised. Even if memory were not an issue, traditional stepwise regression on a hundred billion features is not affordable on most modern computers. Some other strategy is required. A natural alternative is to interleave the generation of features with the assessment of model improvement. Features that are dis-

[1]One cannot use the coefficients of the features that were not added to the model, as this *would* lead to overfitting.

Figure 1: α-investing algorithm

Initialize
$w_0 = W_0$ // initial. prob. of false positives
$model = \{\}$ // initially no features in model
$i = 1$ // index of features
Do forever
$x_i \leftarrow get\_new\_feature()$ // generate next feature
$\alpha_i \leftarrow w_i/2i$
// is p-value of new feature below threshold?
$if(get\_p\text{-}value(x_i, model) < \alpha_i)$
$add\_feature(x_i, model)$ // add x_i to the model
$w_{i+1} \leftarrow w_i + \alpha_\Delta - \alpha_i$
else // otherwise, reject
$w_{i+1} \leftarrow w_i - \alpha_i$ // reduce wealth
$i \leftarrow i + 1$

missed as unhelpful do not burden memory or computation time from that point on.

By interleaving the generation of features with the assessment of model improvement, we may prune the search over features to those which are likely to lead to improvement, and thus make a potentially intractable search tractable. In structural logistic regression and inductive logic programming applications, the approach is to search further in those branches of a refinement graph if at least one of the component terms has proven predictive. In the pairwise interaction term example, as p gets large we see that the number of interactions grows quadratically, creating a similar computational burden to feature selection. In the case of interaction terms we might first perform feature selection on the untransformed p predictors, selecting q examples, and then perform feature selection on the $q{\cdot}p$ interaction terms formed from the selected base predictors. Frequently, $q << p$ and so a great deal of computational time is saved. More importantly, one does not need to penalize complexity for the many interaction terms which are never examined.

4. ALPHA-INVESTING

α-investing controls the false discovery rate by dynamically adjusting a threshold on the p-statistic for a new feature to enter the model. It is easy to implement, and gives provable control of the false discovery rate. The algorithm is shown in figure 1.) The threshold, α_i, corresponds to the probability of including a spurious feature at step i. It is adjusted using the wealth, w_i, which represents the current acceptable number of future false positives. Wealth is increased when a feature is added to the model (presumably correctly, and hence permitting more future false positives without increasing the overall false discovery rate). Wealth is decreased when a feature is not added to the model, in order to save enough wealth to add future features.

More precisely, a feature is added to the model if its p-value is greater than α_i. The p-value is the probability that a feature coefficient could be judged to be non-zero when it is in fact zero. It is computed by using the fact that ΔLoglikelihood is equivalent to t-statistic. The idea of α-investing is to adaptively control the threshold for adding features so that when new (probably predictive) features are added to the model, one "invests" α increasing the wealth, raising the threshold, and allowing a slightly higher future chance of incorrect inclusion of features. We increase wealth by $\alpha_\Delta - \alpha_i$. Note that when α_i is very small, this increase amount is roughly equivalent to α_Δ. Each time a feature is tested and found not to be significant, wealth is "spent", reducing the threshold so as to keep the guarantee of not adding more than a target fraction of spurious features. There are two user-adjustable parameters, α_Δ and W_0, which can be selected to control the false discovery rate; we set both of them to 0.5 in all of the experiments presented in this paper.

α-investing allows us to bound, in expectation, the relative fraction of features incorrectly and correctly added to the model.

THEOREM 1. *Let M_i be the number of correct features included in the model, let N_i be the number of spurious features (those with true coefficient zero) included and w_i be the wealth, all at iteration i, and let $\alpha_\Delta < 1$ be a user selected value. Then if the algorithm in Figure 1 is followed:*

$$E(N_i) < (\alpha_\Delta E(M_i) + W_0)/(1 - \alpha_\Delta)$$

PROOF. The proof relies on the fact that

$$S_i \equiv (1 - \alpha_\Delta)N_i - \alpha_\Delta M_i + w_i$$

is a super-martingale [15]. I.e., S_i is, in expectation, non-increasing in each iteration: $E(S_i) \leq S_{i-1}$. Thus

$$E(S_i) \leq S_0,$$

but since we start out with $N_i = 0$, and $M_i = 0$,

$$E((1 - \alpha_\Delta)N_i - \alpha_\Delta M_i + w_i) \leq W_0$$

Further, since $w_i > 0$ by construction,

$$E((1 - \alpha_\Delta)N_i - \alpha_\Delta M_i) < W_0$$

The proof that S_i is a super-martingale is straightforward by considering the cases when the feature is or is not in the true model and is or is not added to the estimated model. □

Our result, which can be written as

$$E(N_i)/(E(M_i) + W_0/\alpha_\Delta) < \alpha_\Delta/(1 - \alpha_\Delta)$$

or, as many features are added to the model,

$$E(N_i)/E(M_i) < \alpha_\Delta/(1 - \alpha_\Delta)$$

is very similar to a classic false discovery rate, which provides a bound on $E(N_i/M_i)$

The selection of α_i as $w_i/2i$ gives the slowest possible decrease in wealth such that all wealth is used; i.e., so that as many features as possible are included in the model without systematically over-fitting. More formally:

THEOREM 2. *Computing α_i as $w_i/2i$ gives the slowest possible decrease in wealth such that if no feature is added, then*

$$lim_{i \to \infty} w_i = 0$$

PROOF. The proof has two parts. (a) Setting α_i to $w_i/2i^{1+\epsilon}$ would lead to not all of the wealth being used (i.e. not enough features added to the model) and (b) setting α_i to $w_i/2i^{1-\epsilon}$ would lead to the wealth potentially going to zero after a finite number of features was seen, thus prohibiting

any more features from being added, regardless of how significant they were. If no features are added to the model, wealth is $w_i = \Pi_i(1-1/2i)$. Noting that as i becomes large, $\delta_i \equiv 1/2i$ becomes small, and taking the limit as an infinite number of features are considered,

$$w_\infty = \Pi_i(1-\delta_i) = e^{\sum log(1-\delta_i)} = e^{-\sum \delta_i + O(\delta_i^2)}$$

This is zero only if $\sum \delta_i = \infty$, which requires the update rule to decrease as $1/i$ or faster. If wealth were updated more slowly, the corresponding sum $\sum 1/i^{1+\epsilon}$ would be finite and wealth would not go to zero. □

4.1 Guarantees Against Over-fitting

SFS also provides another, much more subtle, guarantee against over-fitting. For the case of "hard" problems, where the coefficients to be estimated are just barely distinguishable above the noise, the cost (increase in out-of-sample error) of adding a "false" feature is comparable to the benefit of adding a true features. One then wants to have a false discover rate of just under fifty percent.

This is a property of using a so-called *testimator*. A testimator tests for significance and then estimates by the usual estimator if it is significant, and estimates by zero otherwise. If a feature has a true coefficient of zero, then when it is falsely included, it will be biased by about $t_\alpha SE$, where t_α is the critical value used for testing significance, and SE is the standard error of the coefficient. On the other hand, the hardest to detect coefficients will have a coefficient of about $t_\alpha SE$. Hence leaving them out will bias their estimated value by about the same amount, namely $t_\alpha SE$. We can thus get optimal test error by adding as many features as possible while not exceeding a specified ratio of false to true features added to the model. This is very similar to controlling the False Discovery Rate (FDR) [4], the number of features incorrectly included in the model divided by the total number of features included in the model, which has become popular in recent years. In the regime that we are working, correctly adding a feature always reduces both the FDR and the out-of-sample error, and incorrectly adding a feature always increases both FDR and error.

5. EVALUATION ON SYNTHETIC DATA

We compared streaming feature selection using α-investing against both streaming and stepwise feature selection using the AIC, BIC and RIC penalties on a battery of synthetic and real data sets.

The base synthetic data set contains 100 observations each of 1,000 features, of which 5 are predictive. We generate the features independently from a normal distribution, $N(0,1)$, with the true model being the sum of five of the predictors plus noise, $N(0, 0.1^2)$. The artificially simple structure of the data (the predictors are uncorrelated and have relatively strong signal) allows us to easily see which feature selection methods are adding spurious features or failing to find features that should be in the model.

The results are presented in Table 3. As expected, AIC massively overfits, always putting in as many features as there are observations. BIC overfits severely, although slightly less badly when streaming is used rather than the less greedy stepwise selection procedure. RIC gives performance comparable to α-investing. As one would also expect, if all of the true features in the model are first in the stream, α-investing does much (three times) better than RIC, while if all of the true features in the model are last, α-investing does much (four times) worse than RIC. In practice, if one is not taking advantage of known structure of the features, one can randomize the feature order to avoid such bad performance.

Stepwise regression gave noticeably better results than streaming feature selection (SFS) for this problem. Using AIC and BIC still resulted in n features being added, but at least all of the correct features were found. RIC gave half the error of its streaming counterpart. However, using standard code from R, the stepwise regression was *much* slower than SFS. Running stepwise regression on data sets with tens of thousands of features, such as the ones presented below, was not possible.

One might hope that adding more spurious features to the end of a feature stream would not severely harm an algorithm's performance.[2] However, AIC and BIC, since their penalty is not a function or p, will add even more spurious features (if they haven't already added a feature for every observation!). RIC (Bonferroni) puts a harsher penalty as p gets large, adding fewer and fewer features. As Table 4 shows, α-investing is clearly the superior method in this case.

6. EVALUATION ON REAL DATA

Table 5 provides a summary of the characteristics of the 10 real data sets that we used. All 10 data sets involve binary classification tasks. The first seven data sets were taken from the UCI repository. The other three contain gene expression data, in which each feature represents a gene expression value for each individual sample (patient with cancer or healthy donor). In the aml data set, samples consist of patients with acute myeloid leukemia and patients with acute lymphoblastic leukemia. The classification task is to identify which patients have which cancer. In the ha and hung data sets, samples consist of healthy donors and patients with either Sezary Syndrome (ha data) or Mycosis Fungoides (hung data). The classification task is to distinguished diseased patients from the controls. Observations which contain missing feature values were deleted.

The baseline accuracy is the accuracy on the whole data set when the majority class is used as the prediction. The feature selection methods were tested on these data sets using ten-fold cross-validation. Since the three biology data sets have large feature sets, we shuffled their features five times (in addition to the cross validations), applied SFS on each feature order, and averaged the five evaluation results.

For each data set, we did two kinds of experiments. The first experiments used only the original feature set. The second interleaved feature selection and generation, initially testing PCA components and the original features, and then generating interaction terms between any of the features which had been selected and any of the original features. In each cross-validation, there were 50 observations in the training set, and the remaining observations were in test set. A small training set size was selected to make sure the problems were difficult enough that the methods gave clearly different results.

[2]One would not, of course, intentionally add features known not to be predictive, but, as described above, there is often a natural ordering on features so that some features such as interactions have a smaller fraction of predictive features.

Table 3: Synthetic data evaluation: AIC and BIC overfit for $p \gg n$. Number of features selected and out-of-sample error, averaged over 10 runs. n = 100 observations, p = 1,000 feature, q = 5 features in data. Synthetic data: $x \sim N(0,1)$, y is linear in x with noise $\sigma^2 = 0.1$. "first" and "last" denote the true features being first or last in the data stream.

stream.	AIC	BIC	RIC	α invest.	α invest. first	α invest. last
features	100	88	4.9	4.7	5.2	3.6
error	41	9	1.0	1.8	0.4	4
stepwise	**AIC**	**BIC**	**RIC**			
features	100	100	5.4	–	–	–
error	2	2	0.5	–	–	–

Table 4: Synthetic data evaluation: Effect of adding spurious features. Average number of features selected and out-of-sample error (10 runs). q = 5 true features, randomly distributed over the first 1,000 features. Otherwise the same model as Table 3. Differences from Table 3 for p=1000 are due to different random samples being used.

p		1,000	10,000	100,000	1,000,000
RIC	**features**	4.5	4.2	3.1	1.9
RIC	**false pos.**	0.1	0.0	0.0	0.0
RIC	**error**	1.0	1.2	1.1	2.8
α invest.	**features**	4.7	4.9	6.1	6.1
α invest.	**false pos.**	0.2	0.6	1.1	1.1
α invest.	**error**	0.9	1.0	1.0	1.0

Table 5: Summary of real data sets. The first seven data sets are from UCI repository and the last three data sets are gene expression data.

		cleve	internet	ionosphere	spam	spect	wdbc	wpbc	aml	ha	hung
features	all (p)	13	1558	34	57	22	30	33	7129	19200	19200
	nominal	7	1555	0	0	22	0	0	0	0	0
	cont.	6	3	34	57	0	30	33	7129	19200	19200
data size		296	2359	351	4601	267	569	194	72	83	57
train size (n)		50	50	50	50	50	50	50	60	70	50
test size		246	2309	301	4551	217	519	144	12	13	7
pca # (if applicable)		3	35	5	7	4	5	5	50	50	50
baseline accuracy		54%	84%	64%	61%	79%	63%	76%	65.3%	71.1%	63%

We used R to implement our evaluation. We were unable to run stepwise regression on the internet data set when interaction terms and PCA were included, and on biology data sets, because the stepwise regression algorithm in R had stack overflow problems on these large feature sets. This problem could no doubt be overcome, but it is indicative of the difficulty of running stepwise regression on large data set. Table 6 shows the evaluation results when only the original features are used; table 7 shows the evaluation results when interaction terms and PCA components are included.

When only the original feature set is used, (Table 6) α-investing has better performance than streaming AIC and BIC only on two of the seven UCI data sets: the internet and wpbc data sets. On the other data sets, for the relatively fewer numbers of features the less stringent penalties do as well as or better than SFS. When interaction terms and PCA components are included (Table 7), α-investing gives better performance than streaming AIC on five data sets, than streaming BIC on three data sets, and than streaming RIC on two data sets. In general, when the feature set size is small, there is no significant difference in the prediction accuracies between α-investing and the other penalties. When the feature set size is larger(i.e., when new features are generated) α-investing begins to show its superiority over the other penalties.

We also compared SFS with stepwise regression using the same penalties on each data set. We find that when the original feature set is used, SFS does not differ significantly from stepwise regression. SFS has better performance than stepwise regression in five cases, and worse performance in four. (Here a "case" is defined as a comparison of SFS and stepwise regression under the same penalty on a data set.) However, when interaction terms and PCA components are included, SFS gives better performance than stepwise regression in nine cases, and stepwise regression has better performance than SFS in only three cases (all on the spam data set). Thus, in our tests, SFS is comparable to stepwise regression on the smaller data sets and superior on the larger ones.

For the three biology data sets (aml, ha and hung), when comparing α-investing with streaming AIC, streaming BIC, and streaming RIC, we find that when the original features are used, RIC gives better performance than α-investing. (Table 8) But when interaction terms and PCA are included (Table 9), RIC is often too conservative to select even only one feature, whereas α-investing has stable performance and higher accuracy than RIC. Note that, regardless of whether or not interaction terms and PCA are included, α-investing always has much higher accuracy than AIC and BIC.

We also tested SFS on a problem of predicting personal bankruptcies[11]. The data set is highly un-balanced, containing 2,244 bankruptcy events and hundreds of thousands of non-bankruptcy observations. The real world loss function for predicting bankruptcy is quite asymmetric: the cost of failing to predict a bankruptcy when one does occur is much higher than the cost of predicting a bankruptcy when none occurs. We call the ratio of these two costs ρ.

We compared Streaming Feature Selection against boosted C4.5, doing 5-fold cross-validation, where each pass of the cross-validation uses 100,000 non-bankruptcies and about one fifth of the bankruptcies. SFS with α-investing was run once, and then the out-of-sample costs were estimated for each cost ratio ρ using the predicted probability of bankruptcy. C4.5 was run separately for each value of ρ.

Table 10: Loss as a function of the loss ratio, ρ, for boosted C4.5 and for SFS with α-investing

ρ	199	99	19	6	4	1
C.45 Loss	132	76	18.6	7.2	5.09	1.45
SFS+alpha Loss	61	41	15.3	6.9	5.02	1.54

Table 10 shows that for low cost ratios, the two methods give very similar results, but at higher cost ratios, SFS with α-investing gives around half the loss of C4.5. Using AIC, one would expect over 1,000 variables to be falsely included in the model, based on the fact that an f-statistic-based penalty of 2 corresponds to a t-statistic of $\sqrt{2}$ which is a wildly generous threshold when considering 67,000 features. BIC also massively overfits, although less severely.

7. DISCUSSION

Recent developments in statistical variable selection take into account the size of the feature space, but only allow for finite, fixed feature spaces, and do not support sequential (or streaming) feature selection. The risk inflation criterion (RIC) produces a model that possesses a type of competitive predictive optimality. RIC chooses a set of features from the potential feature pool so that the loss of the resulting model is within a factor of $\log(p)$ of the loss of the best such model. In essence, RIC behaves like a Bonferroni rule [9]. Each time a predictor is considered, there is a chance that it will enter the model even if it is merely noise. In other words, the tested null hypothesis is that the proposed feature does not improve the prediction of the model. Doing a formal test generates a p-value for this null hypothesis. Suppose we only add this predictor if its p-value is less than α_j when we consider the jth predictor. Then the Bonferroni rule keeps the chance of adding even one extraneous predictor to less than, say, 0.05 by constraining $\sum \alpha_j \leq 0.05$.

Bonferroni methods like RIC are conservative, limiting the ability of a model to add factors that improve its predictive accuracy. The connection of RIC to α-spending rules leads to a more powerful alternative. An α-spending rule is a multiple comparison procedure that bounds its cumulative type 1 error rate at a small level, say 5%. For example, suppose one has to test the p hypotheses $H_1, H_2, \ldots, H_p$. If we test the first using level α_1, the second using level α_2 and so forth with $\sum_j \alpha_j = 0.05$, then we have only a 5% chance of falsely rejecting one of the p hypotheses. If we associate each hypothesis with the claim that a predictor adds to value to a regression, then we are back in the situation of a Bonferroni rule for variable selection. Bonferroni methods and RIC simply fix $\alpha_j = \alpha/p$ for each test.

Alternative multiple comparison procedures control a different property. Rather than control the cumulative α (also known as the family wide error rate), these control the so-called false discovery rate [4]. Control of the false discovery rate at 5% implies that at most 5% of the rejected hypotheses are false positives. In variable selection, this implies that of the included predictors, at most 5% degrade the accuracy of the model. The Benjamini-Hochberg method for controlling the false discovery rate suggests the α-investing rule used in SFS, which keeps the false discovery rate below α: Order the p-values of the independents

Table 6: Real data evaluation: A comparison of penalties in SFS and stepwise regression. The number before ± is the average accuracy on 10 cross-validations; the number immediately after ± is the standard deviation of the average accuracy. The number in parentheses is the average number of features selected by SFS or stepwise regression.

	cleve	internet	ionosphere	spam
stream. (AIC)	77.9±0.9 (6.1)	86.4±0.7 (14.7)	83.2±0.9 (7.9)	80.6±0.7 (9.9)
stream. (BIC)	77.3±1.1 (4.8)	86.8±0.9 (10.2)	80.1±2.5 (4.0)	80.8±0.6 (5.6)
stream. (RIC)	76.0±1.0 (4.2)	90.3±0.2 (3.6)	78.3±2.8 (1.6)	77.8±1.2 (2.5)
stream. (α invest.)	72.2±2.1 (2.7)	90.1±0.1 (3.7)	83.4±0.9 (2.4)	72.6±2.7 (1.7)
step. (AIC)	75.9±1.1 (6.3)	91.0±0.5 (2.9)	79.8±0.9 (7.7)	80.7±0.9 (6.6)
step. (BIC)	76.6±1.0 (4.9)	91.0±0.5 (2.9)	82.7±1.4 (3.0)	81.8±0.8 (5.6)
step. (RIC)	73.7±1.7 (3.4)	88.2±1.0 (0.8)	83.6±1.0 (2.1)	80.1±0.8 (2.2)

	spect	wdbc	wpbc
stream. (AIC)	75.2±1.0 (6.9)	91.5±1.4 (9.6)	69.9±1.9 (4.0)
stream. (BIC)	75.6±0.8 (3.5)	92.4±0.8 (5.5)	71.8±1.5 (2.1)
stream. (RIC)	75.3±1.6 (2.2)	92.4±0.7 (3.4)	74.0±0.8 (1.0)
stream. (α invest.)	74.0±1.6 (2.1)	91.6±0.5 (3.4)	75.1±0.5 (0.9)
step. (AIC)	76.2±1.3 (5.0)	90.2±0.7 (4.1)	68.7±2.2 (4.4)
step. (BIC)	71.2±1.6 (3.1)	90.2±0.7 (4.1)	69.9±1.6 (2.2)
step. (RIC)	69.6±1.8 (2.5)	91.3±0.6 (3.1)	72.2±0.9 (1.2)

Table 7: Real data evaluation: A comparison of penalties in SFS and stepwise regression when interaction terms and PCA features included. This differs from Table 6 in that: (1) we generated PCA components from the original data sets and put them at the front of the feature sets; (2) after the PCA feature "block" and the original feature "block", there is an interaction term "block" in which the interaction terms are generated using the features selected from the first two feature blocks. "NA" means that we were unable to compute the results using the software at hand. See Section 3.

	cleve	internet	ionosphere	spam
stream.+pca+inter. (AIC)	70.1±1.3 (27.5)	82.7±1.7 (37.7)	61.7±3.2 (44)	73.1±2.0 (28.7)
stream.+pca+inter. (BIC)	75.4±0.9 (10.3)	85.5±1.6 (21.3)	83.5±0.9 (8.4)	78.3±1.2 (15.4)
stream.+pca+inter. (RIC)	70.2±1.7 (2.5)	90.5±0.1 (1.5)	74.6±2.5 (0.8)	67.4±2.3 (0.5)
stream.+pca+inter. (α invest.)	74.1±1.2 (4.2)	89.6±0.4 (9.2)	85.2±0.5 (3.3)	67.7±2.4 (0.6)
step.+pca+inter. (AIC)	73.9±1.3 (6.7)	NA	78.7±1.2 (8.1)	79.8±1.1 (6.8)
step.+pca+inter. (BIC)	76.3±0.9 (5.8)	NA	79.5±2.3 (5.2)	81.8±0.8 (5.7)
step.+pca+inter. (RIC)	70.7±1.7 (2.2)	NA	79.8±1.7 (1.1)	77.7±0.9 (1.3)

	spect	wdbc	wpbc
stream.+pca+inter. (AIC)	78.5±1.6 (6.4)	69.2±3.3 (44.6)	66.6±2.3 (18.9)
stream.+pca+inter. (BIC)	80.9±0.4 (2.4)	91.1±1.0 (7.8)	71.1±1.6 (3.9)
stream.+pca+inter. (RIC)	81.6±0.3 (1.8)	93.3±0.6 (1.9)	74.3±0.9 (0.4)
stream.+pca+inter. (α invest.)	81.1±0.4 (1.7)	94.4±0.5 (3.5)	73.7±0.9 (0.8)
step.+pca+inter. (AIC)	78.9±1.2 (3.3)	89.9±0.6 (4.2)	64.7±2.7 (5.9)
step.+pca+inter. (BIC)	78.7±1.5 (2.0)	90.9±0.9 (3.6)	65.4±2.3 (3.7)
step.+pca+inter. (RIC)	81.4±0.3 (1.0)	90.2±0.7 (1.7)	73.6±1.0 (0.4)

Table 8: Gene expression data: A comparison of penalties in SFS, original features only Same format as earlier tables.

	aml	ha	hung
stream. (AIC)	64.8±2.9 (49.0)	62.4±1.7 (49.0)	56.0±4.4 (49.0)
stream. (BIC)	74.6±1.2 (49.0)	67.2±2.2 (49.0)	67.7±1.5 (49.0)
stream. (RIC)	91.0±0.4 (5.4)	77.5±1.2 (2.7)	76.6±2,0 (3.0)
stream. (α invest.)	87.8±1.3 (7.9)	72.4±1.3 (3.2)	75.7±2.8 (3.5)

Table 9: Gene expression data: A comparison of penalties in SFS, interactions and PCA included Same format as earlier tables.

	aml	ha	hung
stream.+pca+inter. (AIC)	78.4±2.2 (49.0)	63.2±1.6 (49)	62.3±3.4 (49.0)
stream.+pca+inter. (BIC)	85.4±1.6 (49.0)	67.2±1.6 (49)	72.6±2.8 (49.0)
stream.+pca+inter. (RIC)	92.7±0.0 (1.2)	70.7±0.1 (0.4)	60.0±0.0 (0.1)
stream.+pca+inter. (α invest.)	93.9±0.3 (9.1)	73.2±0.9 (4.6)	80.9±2.1 (16.2)

tests of $H_1, H_2, \ldots, H_p$ so that $p_1 \leq p_2 \leq \cdots p_p$. Now find the largest p-value for which $p_k \leq \alpha/(p-k)$ and reject all H_i for $i \leq k$. Thus, if the smallest p-value $p_1 \leq \alpha/p$, it is rejected. Rather than compare the second largest p-value to the RIC/Bonferroni threshold α/p, reject H_2 if $p_2 \leq 2\alpha/p$. Our proposed α-investing rule adapts this approach to evaluating an infinite sequence of predictors. There have been many papers that looked at procedures of this sort for use in variable selection from an FDR perspective [2], an empirical Bayesian perspective [13, 17], an information theoretical perspective [10] or simply a data mining perspective [11]. But all of these require knowing the entire list of possible variables ahead of time. Further, most of them assume that the variables are orthogonal and hence tacitly assume that $p < n$. Obviously, the Benjamini-Hochberg method fails as p gets large; it is a batch-oriented procedure.

The α-investing rule of SFS controls a similar characteristic. Framed as a multiple comparison procedure, the α-investing rule implies that, with high probability, no more than α times the number of rejected tests are false positives. That is, the procedure controls a difference rather than a rate. As a sequential feature selector, if one has added, say 20 features to the model, then with high probability (tending to 1 as the number of accepted features grows) no more than 5% (i.e., one) are false positives.

8. SUMMARY

A variety of machine learning algorithms have been developed over the years for online learning where *observations* are sequentially added. Algorithms such as the Streaming Feature Selection (SFS) presented in this paper, which are online in the *features* being used are much less common. For some problems, all predictors are known in advance, and a large fraction of them are predictive. In such cases, regularization or smoothing methods work well and streaming feature selection does not make sense. For other problems, selecting a small number of features gives a much stronger model that trying to smooth across all potential features. (See [1, 14] for a range of feature selection problems and approaches.) For example, in predicting what journal an article will be published in, we find that roughly 10-20 of the 80,000 features we examine are selected [18]. For the problems in citation prediction and bankruptcy prediction that we have looked at, generating potential features (e.g. by querying a database or by computing transformations or combinations of the raw features) takes far more time than the streaming feature selection. Thus, the flexibility that SFS using α-investing provides to dynamically decide which features to generate and add to the feature stream provides potentially large savings in computation.

Empirical tests show that for the smaller data sets where stepwise regression can be done, SFS gives comparable results.For smaller feature sets, any of a number of penalty methods can be used. However, unlike stepwise regression, SFS scales will to large feature sets, and unlike the AIC, BIC and RIC penalties, SFS with α-investing works well for all values of p and n. Key to this guarantee is the use of an α-investing rule which controls the false discovery rate by increasing the threshold on the p-value necessary for adding a variable to the model each time a variable is found significant, and decreasing the threshold each time one is not found to be significant. Given any code which incrementally considers features for addition and calculates their p-value or entropy reduction, SFS using α-investing is extremely easily to implement. For linear and logistic regression, we have found that it can easily handle a million features.

9. ACKNOWLEDGMENTS

We thank **Andrew Schein** for his help in this work and **Malik Yousef** for supplying the gene expression data sets.

10. REFERENCES

[1] Special issue on variable selection. In *Journal of Machine Learning Research (JMLR)*, 2003.

[2] F. Abramovich, Y. Benjamini, D. Donoho, and I. Johnstone. Adapting to unknown sparsity by controlling the false discovery rate. Technical Report 2000–19, Dept. of Statistics, Stanford University, Stanford, CA, 2000.

[3] H. Akaike. Information theory and an extension of the maximum likelihood principle. In B. N. Petrov and F. Csàki, editors, *2nd International Symposium on Information Theory*, pages 261–281, Budapest, 1973. Akad. Kiàdo.

[4] Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society*, Series B(57):289–300, 1995.

[5] P. Bickel and K. Doksum. *Mathematical Statistics*. Prentice Hall, 2001.

[6] D. L. Donoho and I. M. Johnstone. Ideal spatial adaptation by wavelet shrinkage. *Biometrika*, 81:425–455, 1994.

[7] S. Dzeroski and N. Lavrac. *Relational Data Mining*. Springer-Verlag, 2001.

[8] S. Dzeroski, L. D. Raedt, and S. Wrobel. Multi-relational data mining workshop. In *KDD-2003*, 2003.

[9] D. P. Foster and E. I. George. The risk inflation criterion for multiple regression. *Annals of Statistics*, 22:1947–1975, 1994.

[10] D. P. Foster and R. A. Stine. Adaptive variable selection competes with Bayes experts. Submitted for publication, 2004.

[11] D. P. Foster and R. A. Stine. Variable selection in data mining: Building a predictive model for bankruptcy. *Journal of the American Statistical Association (JASA)*, 2004. 303-313.

[12] E. I. George. The variable selection problem. *Journal of the Amer. Statist. Assoc.*, 95:1304–1308, 2000.

[13] E. I. George and D. P. Foster. Calibration and empirical bayes variable selection. *Biometrika*, 87:731–747, 2000.

[14] I. Guyon. Nips 2003 workshop on feature extraction. In *http://clopinet.com/isabelle/Projects/NIPS2003/*, 2003.

[15] J. Jacod and A. Shiryaev. *Limit Theorems for Stochastic Processes*. Springer-Verlag, NY, 2002.

[16] D. Jensen and L. Getoor. *IJCAI Workshop on Learning Statistical Models from Relational Data*. 2003.

[17] I. M. Johnstone and B. W. Silverman. Needles and straw in haystacks: Empirical bayes estimates of

possibly sparse sequences. *Annals of Statistics*, 32:1594–1649, 2004.

[18] A. Popescul and L. H. Ungar. Structural logistic regression for link analysis. In *KDD Workshop on Multi-Relational Data Mining*, 2003.

[19] A. Popescul and L. H. Ungar. Cluster-based concept invention for statistical relational learning. In *Proc. Conference Knowledge Discovery and Data Mining (KDD-2004)*, 2004.

[20] J. Rissanen. Hypothesis selection and testing by the mdl principle. *The Computer Journal*, 42:260–269, 1999.

[21] G. Schwartz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[22] R. A. Stine. Model selection using information theory and the mdl principle. *Submitted for publication*, 2003.

[23] L. H. Ungar, J. Zhou, D. P. Foster, and R. A. Stine. Streaming feature selection using iic. In *AI&STAT'05*, 2005.