THE EFFECT OF HISTORY ON MODELING SYSTEMS' PERFORMANCE: THE PROBLEM OF THE DEMANDING LORD

George Giannakopoulos and Themis Palpanas

October 2010

Technical Report # DISI-10-052

The Effect of History on Modeling Systems' Performance: The Problem of the Demanding Lord

George Giannakopoulos*, Themis Palpanas*
*Department of Information Engineering and Computer Science,
University of Trento,
Via Sommarive, 14 I-38123 POVO, Italy
Email: {ggianna,themis}@disi.unitn.it

***Abstract*—In several concept attainment systems, ranging from recommendation systems to information filtering, a sliding window of learning instances has been used in the learning process to allow the learner to follow concepts that change over time. However, no analytic study has been performed on the relation between the size of the sliding window and the performance of a learning system. In this work, we present such an analytic model that describes the effect of the sliding window size on the prediction performance of a learning system based on iterative feedback. Using a signal-to-noise approach to model the learning ability of the underlying machine learning algorithms, we can provide good estimates of the average performance of a modeling system independently of the supervised machine learning algorithm employed. We experimentally validate the effectiveness of the proposed methodology with detailed experiments using synthetic and real datasets, and a variety of learning algorithms, including Support Vector Machines, Naive Bayes, Nearest Neighbor and Decision Trees. The results validate the analysis and indicate very good estimation performance in different settings.**

***Keywords*-concept drift; user modeling; adaptive learning;**

I. Introduction

In the literature it has been argued [1] that machine learning methods are not sufficient to perform such tasks as user-modeling. This statement expresses the inability of several classifiers to detect changing context (e.g., a user's preferences), that causes changes in the interest-indicative concept that needs to be targeted by a classification algorithm. It is exactly this implied change of context, its effects on classification, and the evolution of learning methods in order to tackle the change that have been studied in the machine learning community as the problem of "concept drift", which takes into account any concept that changes over time. Drifting concepts appear in a variety of settings in the real world, e.g., the state of a free market or the traits of the most viewed movie. Even though there have been a number of methodologies to either track or react to concept drift, there has been little *analytic* work on the connection between the performance of learning and the learning window, i.e., the number of recent instances a learner should remember to keep track of the drifting concept. Most existing works rely on experimental results and heuristic rules to determine the window parameter and optimize the learners' performance.

In contrast, we propose a systematic approach that allows us to estimate the average performance of learning algorithms for a range of learning window sizes and concept drift frequencies, within a learning task in the presence of concept drift. This approach implies a way to optimize the window size for incremental learning tasks and provides the basis for further analytic study of the connection between average performance of an incremental learning system and the noise in the training set. The questions we answer with this study are the following.

- How can we model the expected performance of learning algorithms based on knowledge of the characteristics of the abrupt concept drift (also termed "concept shift"), such as the period of occurrence of these drifts?
- How can we estimate the performance of a learner for different window sizes and concept change periods, regardless of the underlying learning algorithm?

To answer these questions, we focus on the functional relation between the window size and the average performance of a learning system. We formulate the problem under the label "the problem of the demanding lord", emphasizing on its user modeling aspect. Then, we propose a generic methodology that allows to a-priori estimate a function of the window size that computes average performance of a learning system in the presence of periodic concept shifts. In the experimental section (Section V) we also provide promising results for non-periodic shift in real-world data.

To realize this methodology we show that one can approximately express the average performance of a system as a function of signal-to-noise ratio, referring to the training instances in the memory window. Then, for a given period of abrupt concept drift, we analytically express the signal-to-noise ratio at each moment in time as a function of the selected learning window size.

In summary, we make the following contributions. We offer a formulation and analytic solution of the problem of estimating the average performance in learning systems in the presence of abrupt concept drift (concept shift). We describe a methodology to approximately estimate — and consequently easily optimize — the performance of a learning algorithm as a function of signal-to-noise in the training set, regardless of the learning algorithm idiosyncrasies. To the best of our knowledge, this is the first systematic approach for estimating the average performance of learning algorithms in this setting.

In the following sections, we present the related work (Section II) on concept drift and we formulate the problem we face within this work (Section III). We then describe the proposed analytic methodology that describes the average performance of a learning algorithm as a function of its memory window size. Then, we experimentally validate the analysis (Section V) and we conclude with a discussion on our findings (Section VI).

II. Related Work

There have been several studies with different assumptions on the speed or type of drift. The drift can be gradual (termed ”drift”), or instantaneous (termed ”abrupt drift” or ”shift”). It can also be caused by a real change of context (real drift) or by the change on the distribution of arriving instances of the — otherwise fixed distribution — target class (virtual drift). The systems and learning algorithms then react in different ways to the drift, sometimes using fixed window sizes, adaptive window sizes or simply no window. The no window case means that either the algorithm uses all the available information (full-memory) or only the last given information (no-memory). In the following paragraphs we elaborate on the literature, taking into account these aforementioned distinctions, where applicable.

In an early work, the problem of “concept attainment” in the presence of noise was indicated and studied in the STAGGER system [2]. The system approximated a (boolean expression) concept based on examples through weighted symbolic characterizations. The reaction to concept drift was a backtracking methodology that allows changing the current description of the target concept to account for the drift. From that time on a multitude of systems have appeared facing the problem of change in a target learned concept, many of them in the incremental learning domain, which has been studied since the late 80’s [3]. In [4] a full-memory incremental-learning speed-efficient system is presented, aiming to find concept descriptions that are both characteristic (wide coverage) and discriminative (high precision). This work, does not aim to provide a new algorithm for concept attainment, but supports concept attainment processes, allowing optimization of the learning window size regardless of underlying learning algorithm.

A focused study of the mistake rate of a learning algorithm that updates its estimate based on the most recent examples [5] identifies bounds for this rate, based on the number of recent examples. In this work, the adaptation to concept drift over time is termed “incremental tracking”. In [6], another heavily theoretic study, the authors study the problem of tracking a subset of a domain (called the target) which changes gradually over time, under the assumption that the drift occurs slowly. The work connects the VC-dimension (d) of the class of possible targets to the difficulty of attaining the target concept, finally indicating the sampling rate that makes a concept drift trackable. Drift has also appeared as a function of time [7], where a parameter indicates the speed of the drift. Our work is the first to study the learning process in the presence of concept drift from the aspect of signal-to-noise in the training set *analytically*.

In later works, we find approaches where either hard-coded thresholds are used [8] based on trial-and-error, or the window is adjusted whenever a shift is detected [9]. In [7], in order to deal with the drift the authors describe a heuristic algorithm (Window Adjustment Heuristic) that adjusts window size, but they state that the algorithm requires optimization, as its performance affects the entire system strongly. A heuristic approach to deal with concept drift is also described in [10], where we can also find a study of fixed and adaptive windows. Gradual forgetting [11] has also been used, by weighting observations based on the distance of their iteration (time) of appearence to the current system iteration. Another approach uses small sequences (batches) of statistically significant size to estimate, using a statistical test, the performance of the classifier over running data [12]. On a drift detection event, a window-size optimization procedure using the Golden Section algorithm [13] is performed. An alternative window-size optimization approach based on the “optimal switch point of two Gaussian classes” [14] uses the analytics expression of classifier error-rate, to calculate the optimal window size based on a “maximum likelihood”-estimated critical point of drift occurence in the case of frequent abrupt change.

Recently, researchers have also used “local windows” in sub-parts of models, as in [15] where an incremental decision tree uses local sub-concept adaptive window sizes. The local window size is adjusted based on a local performance measure, based on instantaneous performance of the sub-concept classifier. Another approach uses multiple competing windows of different sizes [16], that try to tackle the problem of differentiating noise from virtual drift from actual concept drift.

Other approaches use a window differently or not at all. In [17] we find a distinction between short- and long-term changes in user interests for information filtering, in a feedback-based system. However, no window is used; instead, for short-term updating, interest weights are updated on every iteration, which means that prompt shifts are actually not tackled. The long-term interests are calculated as the average of all interest feedback given by the user per topic. In [18], the proposed system learns based on extreme examples and batch learning. Ensemble based approaches exist, such as the case of boosting [19] which uses only last batches of instances to determine dynamically the training instances per iteration. The base classifiers are also continuously reweighted and updated. In [20] a

Concept Drift Committee (CDC) of decision trees vote for the current classification of instances and each decision tree classifier remains used until its voting record efficiency is reduced below a given threshold. Another approach for high-speed learning in data streams, namely the CVFDT algorithm [21], builds upon the VFDT decision tree learner, updating statistics of subtrees and the set of used subtrees based on a recent window of examples. The window size is static, but the authors indicate that it would be important to dynamically adjust the size. In [22] two classifiers are used, one with full memory and one with partial memory (fixed memory w), in a *paired learning* approach. The full-memory learner is used to learn the current concept, while a reactive learner is used to define key time-points, where the full-memory learner is to be reset and start learning again. In [23], an EM algorithm is used to assign weights to ensemble classifiers, which are created and disposed with the passing of time, in order to adapt to concept drift.

In this work, we provide an analytic framework that allows the a-priori estimation of an optimal window size for the case of periodic concept shifts, overcoming the heuristic or algorithm-specific approaches of the literature. This simplified methodology means to provide the basis for efficient optimization of window sizes in online learning. A major contribution, other than the analysis itself, is based on the proposal of a signal-to-noise function (see Section IV-A) describing the connection between noisy input and the performance of a learning algorithm. The estimation of this function allows one to optimize the window size without explicit knowledge of the learning algorithm, based on an estimation step that captures the behavior of any learning algorithm in the presence of noise.

The methodology presented in the next paragraphs holds for classification tasks. The learning algorithms supported include the supervised learning algorithms, e.g., Naive Bayes, SVMs, Decision Trees, or Nearest Neighbor (see the experimental section, Section V for the related experiments), because no restrictive assumptions are made for the details of the learners.

III. Problem Formulation

We call the problem we analyze the "problem of the demanding lord". The idea is that there is a demanding lord that requires a meal every day from his good servant. The servant tries to estimate a classification of the meals his lord likes, based on his reactions to previous meals. Each day the servant offers a set of meals and gets the full set of reactions from the lord as feedback. The lord, however, changes his preferences randomly. We want to determine how many of the lord's latest answers the servant needs to remember, in order to offer the lord the most satisfactory meals on average over time. It is important to note that we allow the servant to have his own way of learning based on his lord's answers.

The analogy to the actual user modeling problem is the following. The user $\mathbb{W}$ is the demanding lord. The user modeling system $\mathbb{H}$ is the servant. A day identified by its number $d, d \in \mathbb{N}^*$, counted from the beginning of the servant's arrival, is a system iteration. The meal G is the information that has to be evaluated by the user model. The preference A of the lord to a meal is the feedback to the system, concerning a given meal. We consider this to be a value taken from a set $\mathbb{A}$. The way the servant learns, or training policy $\mathbb{P}$, is the machine learning algorithm or methodology used by the user modeling system. The number r of the lord's reactions, which the servant remembers when training, which we call *memory window size*. Since the servant remembers the *last* r reactions, this memory window is sliding over time. The period of shift T_s is the time of days (iterations) that (are expected to) pass between two consequtive interest shifts. In the case of random shifts, T_s can be approximated by the expected value of days between two consecutive interest shifts.

We can differentiate servants from their policy of learning $\mathbb{P}$ and by the number of reactions r they take into account. *The finite-memory servant* remembers the last r reactions only. *The all-remembering servant* remembers all his lord's reactions. The all-remembering servant is a special case of the finite-memory servant one with $r \to \infty$. Therefore, a servant can be described as the pair $\mathbb{H} \equiv < \mathbb{P}, r >$. The lord can be described based on the probability distribution $p(d)$ of an occuring shift, over the days elapsed from the last shift: $\mathbb{W} \equiv < p(d) >$.

We make some assumptions that facilitate the representation of the problem:

- The lord $\mathbb{W}$ periodically changes his interests through what we call an *interest shift*, or simply *shift*. This implies that:
  $p(d) = 1$, if $d = kT_s, k \in \mathbb{N}^*$ else $p(d) = 0$.
- A shift is radical, so that no information is valid concerning reactions on the previous sets of meals. This makes sure that we know which part of the information we have is useful, based on the knowledge of the last interest shift that has occurred.

For a given day d and a set of offered meals $\mathbb{G}_d = \{G_1, G_2, G_3, ..., G_n\}, n > 0$ the set of the lord's reactions on that day is $\mathbb{A}_d = \{A_1, A_2, ..., A_n\}$ containing the reactions mapped to each one of the n meals.

In the following elaboration we refer to Figure 2 to visualize the described states. In Figure 1 we provide the explanation-legend of the corresponding symbols. Each given day d_c, the servant $\mathbb{H}$ uses the r last feedback sets (see Figure 2) $\mathbb{A}_{d_c-r}, \mathbb{A}_{d_c-r+1}, ..., \mathbb{A}_{d_c-1}$ to learn, using his training policy $\mathbb{P}$, to estimate meals. We call this set of feedback sets the training set $\mathbb{T}$ of the servant. In a given point it time d_c the servant is trained using only valid information (the white circles in Figure 2), if within the last r days, no shift has occurred. Otherwise, if a shift occurred

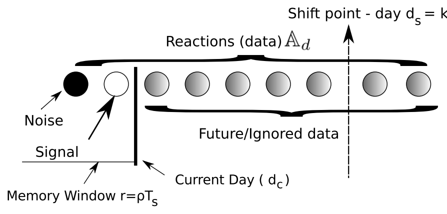
Shift point - day $d_s = kT_s$
Reactions (data) $\mathbb{A}_d$
Noise
Signal
Future/Ignored data
Current Day (d_c)
Memory Window $r=\rho T_s$

Figure 1. Legend, explaining the symbols used in the following illustrations.

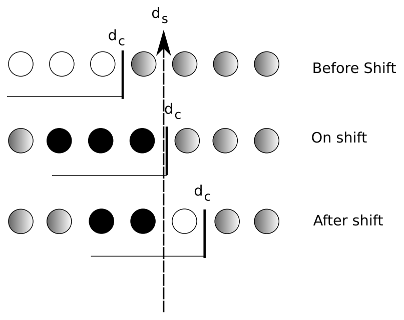

Figure 2. The validity of training data over time. When the current day is after an interest shift, all data before the interest shift become invalid (i.e., noise).

on day d_s, before the current day d_c, $d_c - d_s \leq r$, then the servant has some no longer valid feedback set $\mathbb{N} \subset \mathbb{T}$ (noise, shown as black circles in Figure 2) and some valid $\mathbb{S} \subset \mathbb{T}$ (signal), and $\mathbb{T} = \mathbb{S} \cup \mathbb{N}$. The period of the shift will be noted as T_s, i.e., every shift happens exactly T_s days after the previous one. The first interest shift happens on day $d = T_s$. We start with this assumption of periodicity, to facilitate the formulation of the problem. Later, we verify whether the results of our analysis are also valid for random shift frequency.

If $|\cdot|$ is the operator of the size of a training (sub)set, then we let $S = |\mathbb{S}|$ and $N = |\mathbb{N}|$ represent the signal magnitude and the noise magnitude of a training set $\mathbb{T}$. We also allow $\mathbb{S} = \emptyset \Rightarrow S = 0, \mathbb{N} = \mathbb{T} \Rightarrow N = r$, when all the training set is not valid any longer because a shift has just occurred. Correspondingly, $\mathbb{N} = \emptyset \Rightarrow N = 0, \mathbb{S} = \mathbb{T} \Rightarrow S = r$, when no change has occurred within the last r days (fore more intuition on why this is the case see Figure 2).

Given the above definitions, we define the signal to noise ratio Z of a given moment in time, as:

$$Z = log\frac{S}{N} \simeq log'S - log'N \tag{1}$$

where $log'x = log(1+x)$ returns for a given x the natural logarithm of $x+1$, to return a value also for $x = 0$.

Let us consider that the servant training set size r is a ratio ρ of the shift period T_s:

$$r = \rho T_s \tag{2}$$

We call this ratio ρ, i.e., the memory window-to-shift period ratio, *characteristic ratio* of a given servant $\mathbb{H}$. For a given servant, $\mathbb{H}$ and a given lord $\mathbb{W}$ we support that the servant's average performance on predicting the preferences of the lord is a function $f(\mathbb{W}, \mathbb{H}, \rho)$ or, equivalently, $f(p(d), \mathbb{P}, \rho)$. This means that we consider the performance to be a function of the shift probability distribution, the learning algorithm and the characteristic ratio.

IV. Analysis of Window Size Effect on Performance in a Concept Drift Setting

We perform an analysis of the effect of the memory (window) size of a learner on its performance, in the presence of concept drift. The analysis is based on the estimation of a function connecting signal-to-noise in the training set to the expected performance of a given learner. The estimation process is described in Section IV-A. Then, given the estimated function, an analysis of the mathematic relation between the window size of a learner and the signal-to-noise ratio for every iteration of a modeling system with periodic concept shifts allows the estimation of the average performance of the learning system over time.

The described analysis can be used, in conjunction with concept shift detection methods (e.g., [24]) or various related shift indicators (see [10] for an overview of indicators) to optimize the ρ parameter, for a given user and learning algorithm.

A. The Characteristic Transfer Function of a Learning Algorithm

To describe the $\mathbb{P}$ component of the predictive function $f(p(d), \mathbb{P}, \rho)$, we consider that each learning algorithm is described by a function which indicates the impact of signal-to-noise ratio in the training set to the performance of the algorithm. Given that an algorithm has a minimum performance of $\underline{m}$ and a maximum performance of $\overline{M}$ for a given domain, then we argue that the function that describes the *average* performance $\overline{f}$ as a function of the signal-to-noise ratio Z, is of the form:

$$\overline{f}(Z) = \underline{m} + (\overline{M} - \underline{m})\frac{1}{1 + b \times exp(-c \times Z)} \tag{3}$$

where $exp(x) = e^x$ and the constants $b \in \mathbb{R}, c \in \mathbb{R}$ are parameters of the *sigmoid* function. We call the $\overline{f}$ function the *characteristic transfer function (CTF)* of the learning algorithm. In the case where $\underline{m} = 0, \overline{M} = 1$, Equation 3 takes the form

$$\overline{f}_N(Z) = \frac{1}{1 + b \times exp(-c \times Z)} \tag{4}$$

which we call the *normal characteristic transfer function (NCTF)* and it represents a normalized version of the CTF.

The intuition behind the sigmoid is based on the fact that a (non-trivial) learning algorithm starts to perform well after a certain ratio of good to bad examples has been observed. From that moment on, the performance of the algorithm constantly improves as the ratio is improved, until the point where the best performance is reached. Then, no matter how much the ratio of good to bad examples increases, there is little change, because the algorithm cannot do much better, due to its generalization property. We consider the CTF to be characteristic of an algorithm for a given dataset. We expect that the sigmoid can be estimated from training sets of varying Z and, then, it can be used as a known function for the given algorithm. We illustrate this property of the function in the experimental section (Section V).

In existing literature there have been works that estimate aspects of a learning algorithms' performance, e.g., based on the interaction between design (training) and test instance sets [25], or based on the relation between number of training samples and features to performance [26]. Other works measure the generalization ability, e.g., of Support Vector Machines [27] or neural network classifiers [28]. However, these approaches do not deal directly with the presence of noise in the original data and they do not offer a straight-forward way for estimating the performance as a function of the signal-to-noise ratio.

We emphasize that we do not make any specific assumption for the underlying distribution of training instances in the concept space. The performance estimation methodology we use exploits experimental results to approximate performance, in the place of using robust analytic estimation of error and of noise effect. If, however, a functional representation of the relation between signal-to-noise and performance can be obtained by analytic means, then our estimated function can be simply replaced with the analytically obtained one, without any further consequence to the overall methodology of average performance estimation, described in the following sections, as long as the monotonicity assumption remains in force.

B. Average Performance as a Function of Memory Window to Shift Period Ratio

Given the estimation of the CTF, which describes the $\mathbb{P}$ component of the servant, we need to find the relation between ρ and $\overline{f}(Z)$. We examine the case of the short-memory servant, where $\rho \leq 1$. In this study, we focus on the case where window is smaller than the shift period, i.e., the lord is not expected to have an interest shift too often, and we omit the discussion of $\rho > 1$ due to space limitations.

1) The Short-memory Servant ($\rho \leq 1$): For the case where $\rho \leq 1$, we can calculate the signal to noise ratio, studying different key iteration intervals as follows:

- In the beginning $d = 0$, $S = 0$, $N = 0$.
- In the interval $0 < d < T_s$, $S = min(d, r)$, $N = 0$. This happens because the maximum number of training instances are r i.e.,

$$S + N = r \Rightarrow N = r - S \quad (5)$$

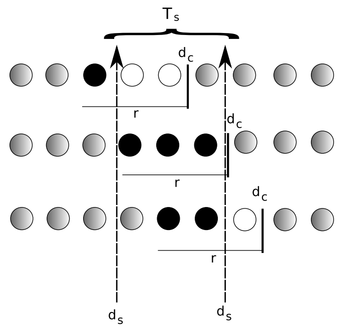

Figure 3. Three consequtive days of a short-memory servant. $\rho = 1, T_s = 3$.

Since everything we know so far is suddenly useless, i.e., noise, if $[a]_b$ is the integer part of the division $\frac{a}{b}$ (the modulo operator), it stands that:

$$d \in \{d' | \, [d']_{T_s} = 0\} \Rightarrow S = 0, N = r \quad (6)$$

- The sum of training instances are r at most; also, everything that is not signal, is necessarily noise. Thus, while $d \in \{d' | d' > T_s, 0 < [d']_{T_s} < T_s\}$ it stands that

$$S = min([d]_{T_s}, r), N = r - S \quad (7)$$

The function $min()$ is the minimum function.

In Figure 3 we illustrate the case where the partial memory ratio $\rho = 1$, and $T_s = 3$.

Since we are interested on the overall average performance, we will only take into account equations 6, 7. From the above we deduce that S, N are actually $S(d), N(d)$ functions of the current day. Every day d the Z is: $Z(d) = log'(S(d)) - log'(r - S(d)) = log'(S(d)) - log'(\rho T_s - S(d))$. The expected value $\overline{Z}$ of Z is:

$$\overline{Z} = E(Z) = \sum_i Z_i q(Z_i) \quad (8)$$

where $q(Z_i)$ is the probability of occurence of Z_i and $Z_i, 0 \leq i < r = \rho T_s, i \in \mathbb{N}*$ indicates the possible values of Z.

The possible values of Z, that appear after the first shift, are exactly $r = \rho T_s$ in number, since S is an ρT_s bounded function of $d(modT_s)$ (see Eq. 7). Their values are:

$$Z_i = log'(i) - log'(\rho T_s - i), 0 \leq i < \rho T_s$$

In the first shift, the values Z_i^1 are also exactly $r = \rho T_s$ with $Z_i^1 = log'(i) - log'(0), 0 \leq i < \rho T_s$ because no noise is present. Over the time of several days (i.e., many iterations), the probability $q(Z_i^1)$ of the Z_i^1 values tends to zero, because there is only one occurrence of the value, regardless of the current day. So for day d, the probability $q(Z_i^1) = \frac{1}{d}$, so when $d \to \infty \Rightarrow q(Z_i^1) \to 0$. In the following analysis, we therefore ignore th Z_i^1 values of Z.

A day d can be modeled related to the last shift that occured. So, if d is k_1 days after the last shift, which occurred on day $k_0 T_s$, then we can express d as $d = k_0 T_s + k_1$. Then, for a chosen, arbitrarily big $k_0 >> 0$, $k_0 \simeq k_0 + 1$

$$d = k_0 T_s + k_1, k_0 >> 0, 0 \leq k_1 < \rho T_s$$

i.e., after many shifts, all the values $Z_i, i \neq \rho T_s - 1$ have a probability of approximately

$$q(Z_i) = \frac{k_0}{k_0 T_s} \Rightarrow$$

$$q(Z_i) \simeq \frac{1}{T_s} \tag{9}$$

since they appear k_0, or $k_0 + 1$ times (depending on k_1) in $k_0 T_s$ days. For the value $Z_{\rho T_s - 1} \equiv Z_{max} = log'(\rho T_s) - log' 0$ (which is a *constant*) — i.e., the maximum value of Z — the probability of occurence of Z_{max} is what is left from the probability mass, if one subtracts the probabilities of other values:

$$q(Z_{max}) = 1 - \sum_{i=0}^{\rho T_s - 2} q(Z_i) \Rightarrow q(Z_{max}) = 1 - \rho + \frac{1}{T_s} \tag{10}$$

Therefore, Equation 8, gives

$$\overline{Z} = \frac{1}{T_s} \sum_{i=0}^{\rho T_s - 2} Z_i - (\rho - \frac{1}{T_s} - 1) Z_{max}$$

Setting $R_{tot} = \sum_{i=0}^{\rho T_s - 1} Z_i = \sum_{i=0}^{\rho T_s - 2} Z_i + Z_{max}$ we get:

$$\overline{Z} = \frac{R_{tot}}{T_s} + (1 - \rho) Z_{max} \tag{11}$$

Using the same methodology, we calculate the performance of the system, using Equation 3. Given the fact that performance f is a strictly monotonic function of Z, the unique values of f are exactly the same in number, as the Z possible values. In addition, each value $f(Z_i)$ holds the same probability of appearence as Z_i. This means that, similarly to Equation 8, the expected value of the performance of the *system* is:

$$\overline{f} = E(f(Z)) = \sum_{i=0}^{\rho T_s - 1} f_i q(f_i) \tag{12}$$

where f_i is the i-th possible distinct value of $f(Z)$, with $q(f_i) \equiv Z_i$ and, given Equation 3: $f_i = \underline{m} + (\overline{M} - \underline{m}) \frac{1}{1 + b \times exp(-c \times Z_i)}$. Thus, for $\overline{f}$, we have the following equation.

$$\overline{f} = \sum_{i=0}^{\rho T_s - 1} \left(\left(\underline{m} + (\overline{M} - \underline{m}) \frac{1}{1 + b \times exp(-c \times Z_i)} \right) q(f_i) \right)$$

Performing associatively multiplication, and taking into account the fact that $\sum_{i=0}^{\rho T_s - 1} q(f_i) = 1 \Rightarrow \underline{m} \sum_{i=0}^{\rho T_s - 1} q(f_i) = \underline{m}$, and using Equations 9, 10 and 4, gives:

$$\overline{f} = \underline{m} + (\overline{M} - \underline{m}) \left(\frac{1}{T_s} \sum_{i=0}^{\rho T_s - 2} \overline{f}_N(Z_i) + (1 - \rho + \frac{1}{T_s}) \overline{f}_N(Z_{max}) \right) \tag{13}$$

Corollary 4.1: Maximization of the average performance $\overline{f}$ of a user-modeling system with a specific training policy (learning algorithm), given a user with periodic interest shifts, can be achieved by merely changing the parameter ρ $(0 < \rho \leq 1)$.

2) Optimization of the Performance: We have argued that the optimization of a system, consists of the following process steps: estimation of the CTF, detection of the interest shift period and optimization of the ρ parameter.

For the optimization process there are two important considerations. The first is that estimated CTF must be as collinear as possible to the true function of performance (an example of estimated and real CTF is illustrated in the following section, in Figure 5). If the collinearity is strong, then the optimization process will give near-optimal results. The second is that, if there is a requirement for a maximally *exact* estimation of the average performance, we need not only a collinear CTF but a CTF with minimum error, as error can be defined by e.g., the absolute difference between the estimated and actual value of performance for all possible Z values.

If the above process stands, then all problems that can be expressed through the problem of the demanding lord can have an a-priori estimation of performance for any given algorithm, with the effort of estimating the characteristic transfer function of the learning algorithm and the calculation of the expected shift period of the user, which of course are non-trivial tasks.

V. Experimental Evaluation

In order to validate the effectiveness of the proposed approach, we perform a set of experiments on different tasks. The purpose of the experiments is twofold:

- First, we want to check whether estimation of the characteristic transfer function (CTF) of an algorithm can be achieved. To do this, for each dataset we perform a 5-fold cross-validation over 10 folds of the experiment on the dataset: the training set is used to estimate the CTF, which is in turn validated on the test set. To validate the model we use correlation tests, where a high correlation value with statistical significance will indicate a useful estimator.
- Second, we want to determine whether the analytic estimator of signal-to-noise converges to the actual average signal-to-noise of the system in the random shifts case. If this is true, then the analytic estimator

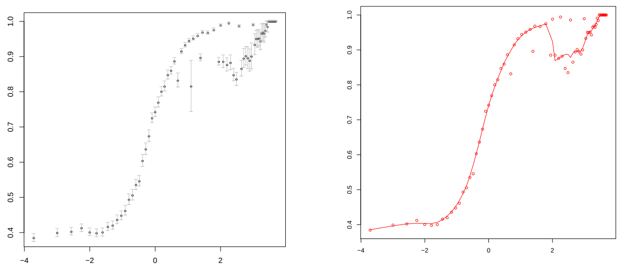

Figure 4. Mean Performance (vertical axis) per signal-to-noise ratio (horizontal axis): Means plot with confidence bars (left) and LOWESS regression plot (right).

can be used for online optimization of ρ, recalculated after every detected shift.

In every case, we describe the evaluation methodology for the learning algorithm, we perform the modeling of the CTF for two algorithms and we validate the CTF estimator concerning the mean performance estimation. Then, we provide the correlation between the estimation and the real average performance of the system.

Before we proceed with the evaluation, we give an example of a Signal-to-Noise to Mean Performance graph — drawn from a single test case) — in Figure 4. The left graph indicates the average performance per signal to noise, including also confidence interval bars within 95% confidence level. Due to the fact that a signal-to-noise value may appear more or less times, there are estimations of average performance that are quite uncertain. Thus, to view the overall tendency of average performance we also use LOWESS regression [29] (with a span parameter of 0.03) to indicate a better estimate of average performance per signal-to-noise value (right graph of Figure 4). The figure illustrates that a sigmoid underlying function possibly relates Z to performance $\overline{f}$. In our experiments, we will try to search for a good-enoigh sigmoid that describes the average performance based on signal-to-noise. For the LOWESS regression calculation we only use Z values that have enough support, i.e., have appeared enough times to have a standard error below 0.05 for the estimation of their mean performance.

A. Description of Datasets

The datasets we report in this study are the following (we experimented with other datasets that yielded similar results, which we omit for brevity).

1) Boolean Concept Dataset: The first dataset we use is based on the STAGGER method evaluation dataset [2], which is traditionally used in the concept drift domain. The experiment is performed as follows. The problem is a problem of classifying 100 randomly generated instances per iteration as either belonging to the current iteration's target concept or not. The instances are objects described based on three dimensions: size, which can be small, medium or large; color, which is either red, green, or blue; shape, which is either square, circular, or rectangular. In the beginning of the task, the set of training instances is generated randomly from the possible variations of objects. Each instance is labeled according to the following rules:

- In iterations 1–40 only *small, red* objects are considered to belong to the target concept.
- In iterations 41–80 objects that are either *green or circular* are considered to belong to the target concept.
- In iterations 81–120 objects that are either *medium or large* are considered to belong to the target concept.

After the first 120 iterations the labelling is repeated circularly.

In every iteration i the training algorithm has access to the first i training instances. On that same iteration, the testing is performed on 100 randomly generated instances, which are labeled according to the current target concept. That is, for the *test data*, on every iteration the labeling is coherent with the current target concept.

2) Real World Dataset: The German Market Dataset: The real world dataset we use is based on the Ifo Business Survey [30], which describes some aspects of the business climate in Germany. The dataset contains one record per month from January, 1991 to February, 2010 (230 records). Each record contains six questionnaire-derived measurements representing the *balance* values and the *index* values of the business situation, the business outlook and the business climate. The balance value of the current business situation is the difference in percentage of the responses "good" and "poor" in the questionnaires; the balance value of the business outlook (expectations) is the difference in percentage shares of the responses "more favorable" and "more unfavorable". The business climate is a transformed (i.e., geometric) mean of the balances of the business situation and the business expectations. The data also includes corresponding index values, normalizing the balances to the average of the year 2000 (which is mapped to the index value 100). Based on the signs of the balances the study classifies the state of the market as "boom", "downswing", "recession", "upswing" (also see Table I).

	Situation Negative	*Situation Positive*
Expectations Positive	Upswing	Boom
Expectations Negative	Recession	Downswing

Table I
THE STATE OF THE MARKET AS RELATED TO BUSINESS SITUATION AND BUSINESS EXPECTATIONS BALANCE SIGNS.

B. Experimental Results

We now provide the experimental results for two tasks: sigmoid estimation and performance estimation over time.

1) Searching for a Good Sigmoid: Given an equation of sigmoid type (see Equation 3), we need to identify the parameters that best describe a set of observed data points. In our case, the data points are measured performance values for given signal-to-noise ratios.

The search in the parameter space is performed by a genetic algorithm [31], searching for an approximate good set of parameters. In our case the fitness function is based on the Kolmogorov-Smirnov goodness-of-fit D statistic [32]. The Kolmogorov-Smirnov test statistic D is expected to have a low value if two sets of samples from distributions are more likely to originate from the same underlying distribution. In our case the two compared distributions are the actual and estimated values of performance, corresponding to the possible Z values.

To determine whether the sigmoid estimation is indeed a good estimator even given unseen values of signal-to-noise we perform a five-fold cross-validation of our sigmoid estimation process. In every fold we use $\frac{4}{5}$ of the data points as the training set and the rest as the test set. The training set is used to determine the CTF and the test set to determine the collinearity (through a Pearson test) between the estimation and the real values of the performance with the given CTF. A high collinearity value indicates that the CTF is a good estimator and can be used for optimization, where collinearity between the estimated and true values are of concern. The fact, however, that we search for a maximally similar function, through the use of the Kolmogorov-Smirnov test, allows even a good approximation of the actual average performance value.

Discussion on the CTF: Observing the data in these runs, we identified an important aspect of the learners, illustrated in Figure 5: sometimes the sigmoid is shifted to the left or to the right. This may indicate that a learner is quite robust to noise, or quite sensitive to noise, since for the same value of Z the performance is different. In other words, the CTF may be better expressed by the form:

$$\overline{f}(Z) = \underline{m} + (\overline{M} - \underline{m})\frac{1}{1 + b \times exp(-c \times (Z - d))} \quad (14)$$

which adds a parameter d that models the horizontal position shift. Thus, in the following experiments we have added the d parameter, which improved the CTF estimation.

In Tables II, III we illustrate the Pearson correlation values indicating how collinear the performance values from the estimated sigmoid CTFs are to the actual values. In order to elaborate on the whole set of results over all folds, we provide the quantiles of the collinearity values, as well as the mean value. The results on the STAGGER dataset, shown in Table II, indicate the consistently high correlation of the estimation to the actual data points.

The market dataset presented a problem for the evaluation, due to the fact that 5-fold cross-validation was not possible. This was caused by the relatively small number of samples for different values of Z, which caused some folds to lack in testing data (remember that the weakly supported Z values were ignored).

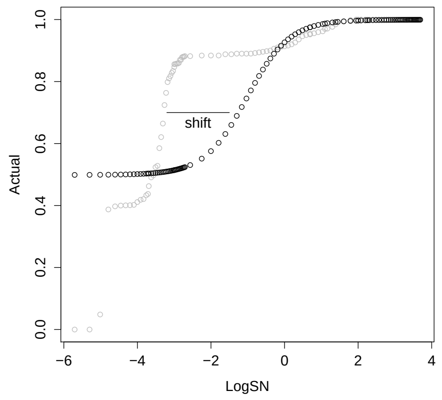

Figure 5. The problem of shift in the Sigmoid. Gray indicates the true data points, black the estimated.

Naive Bayes

Setting T_s	ρ	Min	1st Q.	Median	**Mean**	3rd Q.	Max
40	0.50	0.89	0.94	0.95	**0.94**	0.95	0.96
40	0.85	0.87	0.88	0.89	**0.90**	0.89	0.95
40	1.00	0.61	0.70	0.74	**0.76**	0.79	0.96

J48 Decision Tree

Setting T_s	ρ	Min	1st Q.	Median	**Mean**	3rd Q.	Max
40	0.50	0.83	0.86	0.96	**0.92**	0.96	0.97
40	0.85	0.84	0.87	0.88	**0.89**	0.91	0.95
40	1.00	0.66	0.68	0.74	**0.77**	0.81	0.96

(Correlation Quantiles)

Table II
STAGGER DATASET: CORRELATION IN 10-FOLD CROSS-VALIDATION BETWEEN PERFORMANCE THROUGH CTF ESTIMATION AND ACTUAL PERFORMANCE VALUES. ALL CORRELATIONS ARE WITH A P-VALUE BELOW 0.1.

To avoid this pitfall the evaluation was performed as follows. In every fold we do a random sampling over the Z-Performance value pairs appearing in the dataset, until we construct a training set of 50% the size of the original dataset. We do the same to create an equally big testing test. Then, we estimate the CTF using the training set and measure the correlation of the estimated performance for the Z values in the test set to the actual performance in the test set.

Table III illustrates the very promising performance of estimation for the CTF on the dataset. We should note, however, that in the worst cases the correlation is not statistically significant (p-value $>$ 0.05), even though it is rather strong. This indicates that there is still room for improvement in the estimation of the CTF. This apparent deficiency occurs mostly in cases where there is a sudden fall in performance below a certain Z value, which ”breaks” the

SVM

Setting		**Correlation Quantiles**					
$\overline{T_s}$	$\rho(r)$	Min	1st Q.	Median	**Mean**	3rd Q.	Max
8.46	0.59 (5)	0.39 (0.39)	0.72	0.80	**0.76**	0.90	0.95
8.64	1.04 (9)	0.47 (0.20)	0.58 (0.08)	0.84	**0.77**	0.93	0.97

NN

Setting		**Correlation Quantiles**					
$\overline{T_s}$	$\rho(r)$	Min	1st Q.	Median	**Mean**	3rd Q.	Max
8.46	0.59 (5)	0.63 (0.13)	0.84	0.89	**0.86**	0.94	0.99
8.64	1.04 (9)	0.56 (0.12)	0.81	0.89	**0.84**	0.94	0.98

Table III
GERMAN INDUSTRY DATASET: CORRELATION (5-FOLD VALIDATION) BETWEEN PERFORMANCE THROUGH CTF ESTIMATION AND ACTUAL PERFORMANCE VALUES FOR DIFFERENT Z VALUES. IN PARENTHESES THE P-VALUE, IF P-VALUE ≥ 0.05.

sigmoid function (e.g., see in Figure 5 the three data points with $Z < 4.5$ at the lower left corner of the graph). However, in most cases the collinearity between the estimated CTF and the real CTF is very strong, which allows its use in the prediction of the average performance.

2) Estimation of Performance over Time: In this section we study whether the estimated and real average performance of a *demanding lord system* (DLS), converge over time. We perform an estimation of the sigmoid as indicated before and we keep the best performing estimation function (in terms of collinearity to the actual performance). Using this CTF we follow a learning system over time recalculating on every iteration the average period of the shift, the estimated performance and the actual performance, based on the feedback.

Given the above information we plot graphs indicating the relation between iteration and absolute difference between the estimation and the actual value (absolute error). We call this absolute difference the *delta* of performance. To determine whether this delta is reduced over time, indicating convergence, we measure the Spearman and the Pearson correlations, indicating rank and linear correlation correspondingly between iteration and delta. If their values are negative (ideally strongly negative), this would indicate that *the absolute error reduces over time*. In the case of the real dataset, where the shift period changes over time, the system takes into account on every iteration the *average* period of the shifts.

We perform 10-fold validation, also increasing the number of iterations during which we examine the system, for the synthetic dataset. We present graphs for the different ρ values ($\rho < 1$, $\rho = 1$) for each dataset: Boolean Concept dataset (exhibiting periodic shifts), Figure 6 ; Business Climate dataset (aperiodic shifts), Figure 7. The vertical axis of the figures corresponds to the delta, while the horizontal axis to the iteration number.

We observe (refer to Table IV) that the performance estimation converges (negative values for correlation) well within the statistical significance level of 99%. We note that the Spearman correlation indicates rank-based correlation, while Pearson correlation indicates linear correlation. Correspondingly, Spearman correlation is negative, if for later iterations the delta tends to be lower and Pearson correlation if is reduced linearly to the iteration number. It is very interesting that the convergence also happens in the random shift dataset (Table IV, Business Climate dataset), which indicates that the estimator is robust. It appears that the estimation error falls to levels below 5% rather quickly (few hundreds of iterations), which indicates that the average performance can be estimated quite early, allowing for the corresponding optimization of the memory window. We note that the difference in average performance estimation accuracy in different cases is mainly related to how good approximation the estimated CTF is of the actual CTF. For example, in the case of $ratio = 1$ the estimated CTF is a worse approximation of the actual CTF, than in the case of

Synthetic dataset: Boolean Concept Dataset - 6000 Iterations

Setting		Spearman	Pearson
Bayes	40-20	-0.3022561	-0.272744
Bayes	40-40	-0.9611965	-0.482552

Real dataset: Business Climate, 212 Iterations

Setting		Spearman	Pearson
SVM	Random-5	-0.1777558	-0.3815536
SVM	Random-9	-0.3168430	-0.4193718

Table IV
CORRELATION OF ITERATION NUMBER AND DELTA (10-FOLD VALIDATION). P-VALUE OF TESTS < 0.05.

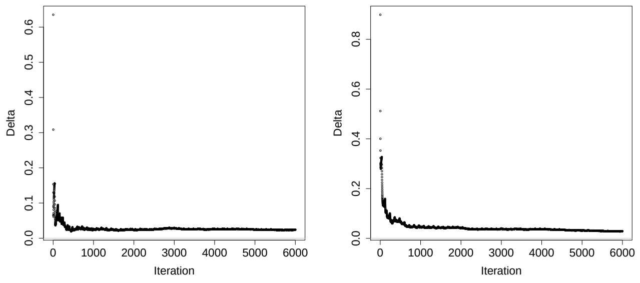

Figure 6. Boolean Concept Dataset: Convergence (10-fold validation) of the average delta. Left to right: $\rho < 1; \rho = 1$.

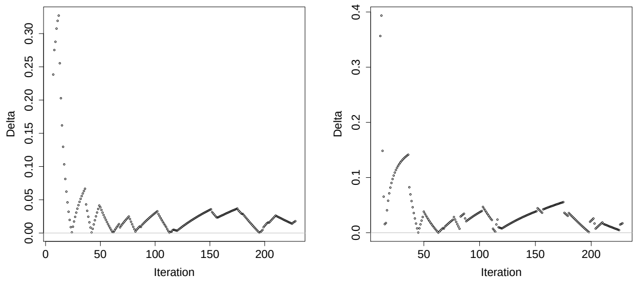

Figure 7. Business Climate Dataset: Convergence (10-fold validation) of the average delta. Left to right: $\rho < 1; \rho = 1$.

$\rho < 1$. This causes different delta values for the two cases (see Figure 6). Given the analysis of the connection between the CTF and the performance, we expect the performance estimation error to be bounded by a function of the error in the estimation of the CTF, but we do not elaborate on this expectation due to space limitations.

VI. Conclusions

In this study, we have shown that we can estimate a characteristic transfer function, connecting signal-to-noise in the training set of a learning algorithm to the average performance of the algorithm. Given this CTF, we can use a closed-form estimation function for the average performance of a learning system (after a sufficiently long observation time) in the presence of concept shifts. Even in cases where not all the assumptions of our closed-form estimator stand, such as the case where we know on average the period of change, the system provides a good estimate of performance as a function of the "memory window"-to-"shift period" ratio, i.e., the characteristic ratio. This means that, on a partial memory online learning system, we can estimate a good memory window for a given series of instances. We can also closely predict the average performance, if the CTF estimation is relatively accurate and our memory window is not bigger than the average concept shift period.

The analysis we have performed offers a basis for studying learning algorithms from a signal-to-noise-response perspective. For a given dataset, calculating a good CTF allows for the estimation of performance, without exhaustive experiments. Furthermore, in cases where we know, for example, the distribution of signal-to-noise we expect to find in a dataset, we can estimate the average performance of a system by finding a weighted average of the CTF values.

References

[1] G. I. Webb, M. J. Pazzani, and D. Billsus, "Machine learning for user modeling," *User Modeling and User-Adapted Interaction*, vol. 11, no. 1, pp. 19–29, Mar. 2001.

[2] J. Schlimmer and R. Granger, "Incremental learning from noisy data," *Machine learning*, vol. 1, no. 3, pp. 317–354, 1986.

[3] D. Angluin, "Queries and concept learning," *Machine learning*, vol. 2, no. 4, pp. 319–342, 1988.

[4] R. Reinke and R. Michalski, "Incremental learning of concept descriptions: A method and experimental results," *Machine intelligence*, vol. 11, pp. 263–288, 1988.

[5] A. Kuh, T. Petsche, and R. Rivest, "Learning time-varying concepts," in *Proceedings of the 1990 conference on Advances in neural information processing systems 3*. Morgan Kaufmann Publishers Inc., 1990, p. 189.

[6] D. P. Helmbold and P. M. Long, "Tracking drifting concepts by minimizing disagreements," *Machine Learning*, vol. 14, no. 1, pp. 27–45, 1994. [Online]. Available: http://dx.doi.org/10.1007/BF00993161

[7] G. Widmer and M. Kubat, "Learning in the presence of concept drift and hidden contexts," *Machine learning*, vol. 23, no. 1, pp. 69–101, 1996.

[8] T. Mitchell, R. Caruana, D. Freitag, J. McDermott, and D. Zabowski, "Experience with a learning personal assistant," *Communications of the ACM*, vol. 37, no. 7, pp. 80–91, 1994.

[9] B. I. Crabtree and S. Soltysiak, "Identifying and tracking changing interests," *International Journal on Digital Libraries*, vol. 2, no. 1, pp. 38–53, 1998.

[10] R. Klinkenberg and I. Renz, "Adaptive information filtering: Learning in the presence of concept drifts," *Learning for Text Categorization*, pp. 33–40, 1998.

[11] I. Koychev and I. Schwab, "Adaptation to drifting user's interests," in *Proc. of ECML2000 Workshop: Machine Learning in New Information Age*. Citeseer, 2000, pp. 39–45.

[12] I. Koychev and R. Lothian, "Tracking drifting concepts by time window optimisation," in *Proceedings of AI-2005, the Twenty-fifth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*. Citeseer, 2005, pp. 46–59.

[13] J. Kiefer, "Sequential minimax search for a maximum," *Proceedings of the American Mathematical Society*, vol. 4, no. 3, pp. 502–506, 1953.

[14] L. Kuncheva and I. Žliobaitė, "On the window size for classification in changing environments," *Intelligent Data Analysis*, vol. 13, no. 6, pp. 861–872, 2009.

[15] M. Núñez, R. Fidalgo, and R. Morales, "Learning in environments with unknown dynamics: Towards more robust concept learners," *The Journal of Machine Learning Research*, vol. 8, pp. 2595–2628, 2007.

[16] M. Lazarescu, S. Venkatesh, and H. Bui, "Using multiple windows to track concept drift," *Intelligent Data Analysis*, vol. 8, no. 1, pp. 29–59, 2004.

[17] D. Widyantoro, T. Ioerger, and J. Yen, "An adaptive algorithm for learning changes in user interests," in *Proceedings of the eighth international conference on Information and knowledge management*. ACM New York, NY, USA, 1999, pp. 405–412.

[18] M. Maloof and R. Michalski, "Selecting examples for partial memory learning," *Machine Learning*, vol. 41, no. 1, pp. 27–52, 1999.

[19] M. Scholz and R. Klinkenberg, "Boosting classifiers for drifting concepts," *Intelligent Data Analysis*, vol. 11, no. 1, pp. 3–28, 2007.

[20] K. Stanley, "Learning concept drift with a committee of decision trees," *Computer Science Department, University of Texas-Austin*, 2001.

[21] G. Hulten, L. Spencer, and P. Domingos, "Mining time-changing data streams," in *Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM New York, NY, USA, 2001, pp. 97–106.

[22] S. Bach and M. Maloof, "Paired learners for concept drift," in *Data Mining, 2008. ICDM '08. Eighth IEEE International Conference on*, 2008, pp. 23–32. [Online]. Available: 10.1109/ICDM.2008.119

[23] F. Chu, Y. Wang, and C. Zaniolo, "An adaptive learning approach for noisy data streams," in *Data Mining, 2004. ICDM '04. Fourth IEEE International Conference on*, 2004, pp. 351–354.

[24] J. Gama, P. Medas, G. Castillo, and P. Rodrigues, "Learning with drift detection," *Lecture Notes in Computer Science*, pp. 286–295, 2004.

[25] K. Fukunaga and R. Hayes, "Estimation of classifier performance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 10, pp. 1087–1101, 1989.

[26] ——, "Effects of sample size in classifier design," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, no. 8, pp. 873–885, 1989.

[27] T. Joachims, "Estimating the generalization performance of an SVM efficiently," in *MACHINE LEARNING-INTERNATIONAL WORKSHOP THEN CONFERENCE-*. Citeseer, 2000, pp. 431–438.

[28] M. Musavi, K. Chan, D. Hummels, and K. Kalantri, "On the generalization ability of neural network classifiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 6, pp. 659–663, 1994.

[29] W. S. Cleveland, "LOWESS: a program for smoothing scatterplots by robust locally weighted regression," *American Statistician*, p. 5454, 1981.

[30] Ifo Institute, "Ifo business survey - the ifo business climate for germany since january 1991," 2010, http://www.cesifo-group.de/portal/page/portal/ifoHome/a-winfo/d6zeitreihen/15reihen/_reihenkt.

[31] D. Goldberg, *Genetic Algorithms in Search and Optimization*. Addison-wesley, 1989.

[32] F. Massey Jr, "The Kolmogorov-Smirnov test for goodness of fit," *Journal of the American Statistical Association*, vol. 46, no. 253, pp. 68–78, 1951.