

Active Learning for Survival Analysis with Incrementally Disclosed Label Information

Klest Dedja^{1,2,*}, Felipe Kenji Nakano^{1,2} and Celine Vens^{1,2,*}

¹KU Leuven - Department of Public Health and Primary Care, Etienne Sabbelaan 53, 8500 Kortrijk, Belgium

²imec - ITEC research group, Etienne Sabbelaan 51, 8500 Kortrijk, Belgium

Abstract

Our study introduces a novel, generalised active learning framework for survival analysis, where we challenge the assumption that the oracle possesses full information about the time-to-event at any stage. Given this generalisation, we allow for querying the same instance multiple times, leading to an approach never employed in survival analysis to our knowledge. A central component of our contribution lies in the modification of the underlying time-to-event prediction model, the Random Survival Forest in our case, enabling it to accommodate partial information on ‘unseen’ data during the active learning phase and thus effectively estimate the conditional risk of a given instance. This adaptation is relevant in scenarios where instances are equipped with partial information. Furthermore, we adapt and introduce new sampling strategies that align with the novel survival analysis framework, thereby ensuring their compatibility with the conditional predictions output by the model. This research offers a comprehensive, novel approach to active learning for survival analysis, setting a solid foundation for further developments in this under-explored intersection of the fields of active learning and survival analysis. Finally, our work not only expands the capabilities of both fields, but also sets the stage for their collaborative use in real-world longitudinal time-to-event studies, where new information typically emerges over time.

1. Introduction


Active Learning (AL) is a field of machine learning that focuses on algorithms that can interactively query an oracle (usually a human expert) to acquire the labels of the most informative unlabelled training examples, with the aim of improving predictive models’ performance in the presence of a pool of unlabelled examples [1], and maximising model performance with the least amount of training labels. These algorithms bypass the need for exhaustive labelling and provide therefore a cost-effective solution to complex problems. AL is especially beneficial when unlabelled data are abundant, but manual labelling is costly or time-consuming. Initially, a base model is trained on the set of labelled data; the model is then used to predict unlabelled data instances and select which ones should be labelled based on an informativeness criterion. These newly labelled instances are then used to update the model, thus iteratively improving its performance.


AL has been broadly employed in regression and in classification tasks, such as image recognition [2] or text classification [3], and these frameworks have been covered by a rich

IAL@ECML-PKDD’23: 7th Intl. Worksh. & Tutorial on Interactive Adaptive Learning, Sep. 22nd, 2023, Torino, Italy

✉ klest.dedja@kuleuven.be (K. Dedja); felipekenji.nakano@kuleuven.be (F. K. Nakano); celine.vens@kuleuven.be (C. Vens)

ORCID 0000-0001-5280-6717 (K. Dedja); 0000-0002-4884-9420 (F. K. Nakano); 0000-0003-0983-256X (C. Vens)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

literature summarised in reviews such as [4, 5]. Other research aspects of AL such as budget constraints [6, 7] and estimation of learning curves [8] have also extensively been analysed. On the other side, little research has been done on applying AL techniques to Survival Analysis tasks.

Survival Analysis is a branch of statistics whose goal is to predict the time until an event occurs. A key challenge tackled within this framework is the so called *censoring* effect, that is when the exact time of the event is not observed, leaving the researcher with partial, or weakly supervised, information. Censoring is a common issue in healthcare applications such as in clinical studies, where patients may drop out of the study before their event occurred (e.g., the occurrence of a disease complication, hospital discharge, or death), or where patients may not have not reached the event by the termination of the study.

Survival Analysis (SA) has only recently been explored in the machine learning community [9] and little work has been done in the intersection between SA and AL. The most notable examples of this are [10], and [11] respectively. The former provides an AL-based survival model which uses (regularised) Cox proportional hazard model [12] in combination with a novel model discriminative gradient-based sampling scheme. Meanwhile, the latter combines a deep learning approach to SA and proposes a novel AL sampling technique for such framework. However, both methods operate under the assumption that the time-to-event labels, once accessed during the query stage, are immutable and cannot be updated in subsequent queries.

This assumption, however, does not always hold in reality as noted in [13], as it does not take into consideration label delay, a common factor to keep in mind when dealing with longitudinal studies. We therefore challenge this assumption and do not assume that full information is revealed by the oracle, we assume instead that an unknown, random amount of information is provided by the oracle, and we open the possibility to query the same instance *multiple* times to obtain more updated information. The relaxation of such assumptions is particularly relevant in real-world applications such as clinical trials, where patients or volunteers can be called multiple times to record the occurrence of a certain event, but for which budget constraints limit the possibility of follow-up.

To align with the assumption, existing SA models need to be adjusted to consider partial information during the querying phase. To this end, we have modified the Random Survival Forest (RSF) algorithm [14] to handle this type of information. Moreover, we have revised two existing AL sampling strategies to fit the SA approach, while also introducing two novel strategies; we ensured the compatibility of these adjustments with the broader proposed framework.

We tested the validity of the proposed framework on several datasets, both real-world and synthetically generated, and showed that relatively simple sampling strategies can outperform random sampling.

1.1. Contributions

To summarise, the main contributions of our paper are the following:

- We propose a generalised framework for AL applied to SA, where we drop the common, underlying assumption that the oracle has access to full information about the time-to-event at any given stage. To our knowledge, this is the first study that includes the possibility of querying the same instance multiple times.

- We adapt the widely used time-to-event RSF model to handle partial information during the querying phase so that it can effectively predict the *conditional* risk of a given instance. This adaptation becomes particularly relevant in the light of the previously listed contribution, where instances with partial information are created.
- We adapt and develop new sampling strategies to be integrated with the novel SA framework, and we also ensure that these strategies are compatible with the conditional predictions made by the framework.
- We evaluate the efficacy of the adapted conditional model in conjunction with various sampling strategies, including both revised existing methods and novel ones, across multiple datasets under our newly proposed framework.

The code and datasets used in this paper can be accessed at: <https://github.com/Klest94/AL-SA-paper-material/>.

2. Background

This section outlines two key areas that underpin our work: SA and AL. Following these, we will discuss related work in a separate subsection.

2.1. Survival Analysis

A SA set-up is able to make unbiased estimates of time-to-event predictions in the presence of censoring. In most of the cases, the censoring leads to a lower estimate of the true time-to-event only, called *right-censoring*. With a right-censoring scenario, there are two possible outcomes: either the true time-to-event is observed, or a censoring event happens before the real event: the instance’s event falls outside of the observation window and the observer has no information on the event time from that moment onward.

Mathematically speaking, the information on the event of interest for an instance x can be represented as a tuple (T, δ) , where T corresponds to the time of the event, and δ corresponds to the observation’s status: $\delta = 1$ stands for the event of interest being effectively observed at time T , while $\delta = 0$ stands for an event being censored as it reaches the end of the observation window at time T .

The goal of a SA study is to estimate such time-to-event at a population or at an individual level given a set of individual information. The typical outcome is a function over time called *survival function* $S(t|x_i)$ which indicates the probability of not experiencing the event (in other words, ‘surviving’, hence the name) until at least time t for individual x_i . In other words:

$$S(t|x_i) = \mathbb{P}(T > t \text{ for instance } x_i) \quad (1)$$

The conditional over the instance x_i is omitted from now on, as the same notation can be used to describe the survival distribution both at an individual and at a population level.

Another useful concept in SA is the *hazard function*, defined as the the conditional probability of an event happening within an infinitesimal time interval Δt , divided by the width of that interval:

$$\lambda(t) = \lim_{\Delta t \rightarrow 0} \frac{\mathbb{P}(t \leq T < t + \Delta t | T \geq t)}{\Delta t \cdot S(t)} = -\frac{S'(t)}{S(t)}. \quad (2)$$

Its integral form $\Lambda(t) = \int^t \lambda(s) ds$, is called *cumulative hazard function* (CHF) and is also widely used.

2.2. Active Learning

The AL framework relies on a iterated cycle of *querying*, *labelling*, and *updating*. Initially, there are two pools of data: the labelled set \mathcal{T} and the unlabelled set \mathcal{U} , and the learning model \mathcal{M} is trained on the labelled pool. Following the initialisation step, the AL framework moves into the querying phase. Here, the learner selects specific instances from the unlabelled pool, based on a pre-defined querying strategy. These instances are labelled by an oracle, usually a human expert, and added to the labelled dataset. The model is then updated or re-trained using the expanded labelled dataset, improving its predictive capabilities, and the cycle repeats, enhancing the model's performance with each iteration. The query strategy plays a key role in the AL framework, as a good sampling strategy can increase the efficiency of the model by reducing the amount of necessary data while simultaneously improving the performance of the model.

Various sampling strategies have been developed to select the most informative samples from the unlabelled pool. The simplest strategy is *uncertainty sampling*, which involves querying the instances about which the current model is most uncertain. This strategy is intuitively appealing because it directly reduces the model's uncertainty [15]. However, it can lead to excessive querying near the decision boundaries, while ignoring potentially informative instances farther away [1]. Another common strategy is *query-by-committee* (QBC), in which several models (the 'committee') are trained on the current labelled set. Each committee member votes on the labels of unlabelled instances, and the instances with the most disagreement are queried [16]. QBC leverages the wisdom of the crowd to mitigate the risk of querying uninformative instances.

Moreover, several hybrid strategies have been proposed, and common approaches consist of strategies that combine the aforementioned ones with a diversity-driven sampling strategy, or that take into account the estimated informativeness of the candidate sample. Finally, several sampling strategies have been designed so to consider the potential influence of instances on the model's future performance, referred to as expected model change [17], expected error reduction [18], and variance reduction [19].

2.3. Related work

As mentioned before, the intersection of AL and SA is a relatively unexplored field, although some studies have emerged recently. The first example is represented by the work of [11], where the authors introduce a novel sampling strategy in AL to iteratively improve a deep learning model. More specifically, the authors propose a Convolutional Neural Network to learn a meaningful feature representation and train a Cox regression model on top of it. As a next step, a sampling strategy based on the expected performance change of the model is proposed for the AL framework. The authors use an external data source as the oracle for labelling the instances during the querying phase. More specifically, the expected time-to event from life tables from an independent data source are used and assumed to be the real time-to event; as a result, fully observed event tuples $(T_i, 1)$ are provided to the queried instances during the labelling phase.

The second example is represented by [10], where an Active Regularised Cox regression framework is introduced, and combines AL with Cox regression models. The authors, starting from the log-likelihood of regularised Cox regressions, derive a framework that makes use of gradient-based sampling strategy to select instances for labelling during the AL process. Unlike [11], there is no assumption that the queried labels include full information on the time-to-event, and tuples of the format (T_i, δ_i) are provided by the oracle upon querying request.

However, neither of these studies takes into consideration the longitudinal nature of time-to-event data. More specifically, these studies do not take into account the fact that in many real-world situations the information of an instance’s label also depends on the time of querying, and that more information can be available at later time points.

3. Framework

To counter the aforementioned limitations, we propose a novel AL framework that includes the possibility to query the same instance multiple times for time to event data. In other words, a given instance in \mathcal{U} can be queried multiple times until all available information is extracted. That is, instances are queried until the real time of the event is observed, with $(T_i, 1)$, or when a terminal censoring factor is encountered, with $(T_i, 0)$.

Our proposed framework represents a more general set-up, expanding in two main directions:

- on the information revealing side: both partial information as well as full information can be revealed by the oracle. In the first setting, the oracle knows that the event has not been reached yet and can provide updated information about the censoring time. In the second setting, the oracle confirms that the event has taken place and provides the event time, or confirms that the instance has been lost to follow-up or has reached the study end, and provides the final censoring time.
- on the inclusion criteria for instances to be part of the pool \mathcal{U} set: both instances with no label information and instances with partial label information are included¹. The former category includes instances for which no information regarding event time is available. The latter includes instances for which the oracle already revealed partial information before, e.g. that the event was not reached yet at a particular point in time.

A visual representation is given in Table 1, where we compare the scope of our framework against other AL + SA set-ups. We refer to our framework as a *multi-query* AL.

Certain modifications are required to implement our suggested framework. Firstly, the label set now comprises triples $(T_i, \delta_i, \epsilon_i)$, where T_i and δ_i represent the previously mentioned time-to-event data, and where the variable ϵ_i shows whether there is additional label information available to be queried from the instance ($\epsilon_i = 0$) or if the instance’s data has been completely ‘exhausted’ ($\epsilon_i = 1$). This additional information is crucial because an instance can now be queried multiple times, allowing it to simultaneously be part of the training set \mathcal{T} and the pool set \mathcal{U} . Consequently, the sampling strategies must consider this new feature, linking each

¹partial label information in the sense that more information can be queried from the oracle in the future. Not to be confused with the partial information in the sense of censoring in a SA setting.

		→ more general	
more general ↓	candidates	Query with fully missing info	Query with missing or partial info
	Full reveal only $\delta = 1$	Nezhad et al. [11]	
	Full reveal $\delta \in \{0, 1\}$	Vinzamuri et.al [10]	
	Partial reveal $\delta \in \{0, 1\}$		Our work

Table 1

Categorisation of current AL and SA combinations. The scope of our paper can be placed in the bottom-right quadrant (broadest scope).

instance with a status ϵ which marks whether additional information can be queried or whether the instance has reached its exhaustion point.

Upon their initial query, query instances are included in the training set \mathcal{T} , and they are only removed from the pool set \mathcal{U} once exhausted. It is important to note that although the model \mathcal{M} is continuously aware (through the oracle) of whether an instance has been exhausted, it remains uninformed about the total data quantity that can be revealed by the oracle in each step.

From the perspective of an instance x_i in the pool set \mathcal{U} , the procedure looks as follows:

- The instance starts with no available information; this equals to being censored at time 0, and the instance label corresponds to the tuple $y_i = (0, 0, 0)$.
- When queried for the first time, the instance enters the training set \mathcal{T} with the corresponding revealed label $(T_{i,1}, \delta_{i,1}, \epsilon_{i,1})$; if $\epsilon_{i,1} = 0$ then $\delta_{i,1} = 0$ automatically.
- In general, when queried for the k -th time, the label $(T_{i,k-1}, 0, 0)$ is updated to $(T_{i,k}, \delta_{i,k}, \epsilon_{i,k})$, and if $\epsilon_{i,k} = 1$ the instance is dropped from \mathcal{U} .

To handle partial information, our proposed framework necessitates adaptations of the underlying model \mathcal{M} . We therefore introduce a conditional adaptation of the Random Survival Forest (RSF) model [14]. The choice of RSF, a time-to-event adaptation of the established Random Forest algorithm by [20], is based on the proven efficacy of RF classifiers in AL scenarios, particularly in handling unbalanced classification tasks [21], coupled with the scarcity of literature concerning applications of RSF in AL scenarios.

Our RSF adaptation updates the predictions initially made by a standard RSF model when faced with partial label information. These updated predictions are then used in querying phase of the AL framework. Further, our framework unlocks the ability of the model \mathcal{M} and the sampling strategy \mathcal{S} to leverage the partial information inherent in the samples that belong to both \mathcal{T} and \mathcal{U} . Specifically, when forming the query set \mathcal{Q} to be submitted to the oracle, the model \mathcal{M} must be capable of estimating survival curves $\hat{S}(t)$ from previously unseen samples in \mathcal{U} , and concurrently estimate curves conditionally, based on the status of a given sample up to time T_i and its corresponding label $y_i = (T_i, 0, 0)$ in \mathcal{T} . This requirement can be fulfilled by

updating the prediction $\hat{S}(t)$ of a standard survival model \mathcal{M} as follows:

$$\tilde{S}(t) = \hat{S}(t|t \geq T_i) = \begin{cases} \hat{S}(t)/\hat{S}(T_i), & \text{for } t \geq T_i \\ 1, & \text{for } t < T_i \end{cases} \quad (3)$$

This method corresponds closely with the approach delineated in [22], where predictions are updated based on the conditional survival function and the definition of conditional probabilities.

In the context of a multi-query framework where it is possible to access partial label information, this assumption proves particularly useful as it allows us to refine our estimate of the time to the event. Specifically, if we know that the event hasn't occurred by time T_i , we can update our estimate based on this knowledge. Similarly, when considering the conditional cumulative hazard function, we obtain the updated estimate as:

$$\tilde{\Lambda}(t) = \hat{\Lambda}(t|t \geq T_i) = \begin{cases} \hat{\Lambda}(t) - \hat{\Lambda}(T_i), & \text{for } t \geq T_i \\ 0, & \text{for } t < T_i \end{cases} \quad (4)$$

It is worth noting that such updates are performed only on queried labels of the form $(T_i, 0, 0)$, as instances that are fully exhausted are dropped from \mathcal{U} and do not need to be updated. In this paper, unless stated otherwise, we will refer to our novel, conditional RSF model as \mathcal{M} .

At each iteration step, a sampling strategy identifies the top B instances from \mathcal{U} , predicted to be the most informative based on a specific heuristic \mathcal{S} . A detailed discussion on this implementation is provided in Section 3.1. The instances that are deemed as the most informative are then queried to the oracle. During the reveal procedure, the oracle discloses some amount of information on the corresponding label. If the instance was previously unseen, it is incorporated into the training set \mathcal{T} with the corresponding label, following a typical AL framework. However, an instance is only removed from the pool set \mathcal{U} once all the available information has been fully disclosed.

The pseudo-code of our multi-query, AL framework for SA is presented in Algorithm 1.

3.1. Sampling strategies

We select a range of sampling strategies, some previously suggested for AL in binary classification tasks and then adapted to align with a SA setting, others that are proposed by our work for the first time. In choosing the sampling strategies, given the exploratory nature of our paper, we favoured strategies that are vastly established, simple to implement, and easy to replicate. We examine their efficacy within the multi-query framework and compare them against their density-weighted counterparts. The specific sampling strategies under comparison in our study include:

Random. Consists of randomly sampling of B instances from the \mathcal{U} at each iteration (also known as 'passive learning'). This strategy serves as a baseline for the next sampling strategies, and can perform reasonably well as noted by [23, 24].

Uncertainty-based. This strategy involves querying instances whose predictions are closest to the population average in the training set: $\bar{Y}_{\mathcal{T}}$. This method is analogous to margin-based

Algorithm 1 Our proposed multi-query AL framework for SA

Require: \mathcal{T}, \mathcal{U} ▷ Initialise training set and pool set
Require: Sampling strategy \mathcal{S} ▷ Define a sampling strategy
Require: N rounds, batch size B ▷ Usually constrained by the budget
1: $\mathcal{M}(\mathcal{T})$ ▷ Fit initial model
2: **for** $j = 1$ **to** N **do**
3: $\mathcal{Q} \leftarrow \mathcal{S}(\mathcal{M}, \mathcal{U})$, with $\mathcal{Q} \subseteq \mathcal{U}$ and $|\mathcal{Q}| = B$ ▷ Select top B instances according to \mathcal{S}
4: Update labels in \mathcal{Q} through the oracle
5: $\mathcal{U}_{new} = \{(x_i, y_i) : y_i \in \mathcal{Q} \cap y_i \notin \mathcal{T}\}$ ▷ Identify unseen (new) samples
6: $\mathcal{U}_{seen} = \{(x_i, y_i) : y_i \in \mathcal{Q} \cap y_i \in \mathcal{T}\}$ ▷ Identify samples used in previous iterations
7: $\mathcal{Q}_{full} = \{(x_i, y_i) : (y_i \in \mathcal{Q}) \cap (\epsilon_i = 1)\}$ ▷ Identify exhausted labels and relative instances
8: Update \mathcal{T} ▷ Update existing labels with the new labels in \mathcal{U}_{seen}
9: Update $\mathcal{T} = \mathcal{T} \cup \mathcal{U}_{new}$ ▷ Add the new, previously unseen labels to \mathcal{T}
10: $\mathcal{M} = \mathcal{M}(\mathcal{T})$ ▷ Retrain model \mathcal{M}
11: $\mathcal{U} = \mathcal{U} \setminus \mathcal{Q}_{full}$ ▷ Drop exhausted instances from pool set
12: **end for**

sampling in classification contexts [1, 25], where instances nearest to the decision boundary are prioritised for querying:

$$x^* = \underset{x}{\operatorname{argmax}} |\mathcal{M}(x) - \bar{Y}_{\mathcal{T}}| := \underset{x}{\operatorname{argmax}} U(x) \quad (5)$$

Given that there is no decision boundary in SA tasks, the first instances to be sampled are those predicted to have a survival rate closest to the population average.

Variance-based. This strategy is a novel approach where instances with the highest standard deviation (or equivalently, variance) in their predicted rank are prioritised for sampling. Uniquely, the standard deviation is calculated among the predicted ranks of individual decision tree learners that make up \mathcal{M} .

Belonging to the Query by Committee (QBC) family, this strategy benefits from directly leveraging the ensemble nature of the Random Survival Forest (RSF) model. The sampling criterion can be expressed as follows:

$$x^* = \underset{x}{\operatorname{argmax}} \sqrt{\operatorname{Var}(r(x))} := \underset{x}{\operatorname{argmax}} V(x) \quad (6)$$

where $r(x) = \{\operatorname{rank}_m(x, \mathcal{T}) \text{ for } m \in \mathcal{M}\}$

Here, $\operatorname{rank}_m(x, \mathcal{T})$ is a function that outputs the rank of the prediction $\hat{y} = m(x)$ of a single learner $m \in \mathcal{M}$, compared to the predictions made by of the ensemble \mathcal{M} on the instances in the training set \mathcal{T} . Note that we are focusing on the standard deviation of the predicted *ranks*, rather than the predicted risk itself. This focus arises from the consideration that in time-to-event scenarios, the relative ranking typically bears more meaningful insights than the absolute predicted risk [12, 26].

Uncertainty+density. Is a hybrid approach that combines the uncertainty-based approach with a density-based score. We compute the uncertainty measure as defined in Equation (5)

and min-max normalise it across the candidates of the pool set to get $\bar{U}(x)$. Next, we add the min-max normalised *abnormality* score $\bar{A}(x)$ of each candidate as computed by an Isolation Forest[27] algorithm, weighted by a factor β . In formulas, the sampling strategy follows the following criterion:

$$x^* = \operatorname{argmax}_x (\bar{U}(x) + \beta \bar{A}(x)) \quad (7)$$

In our experiments we set $\beta = 1$, similarly to other approaches in literature [4, 28].

Variance+density. Is another novel, hybrid approach that combines the variance-based approach introduced above with a density-based score. The computation of the standard deviation in the predicted ranks is computed as in Equation (6), and the correction by normalisation and abnormality score is carried in the same fashion as in the previous approach, leading to $\bar{V}(x)$.

$$x^* = \operatorname{argmax}_x (\bar{V}(x) + \beta \bar{A}(x)), \quad (8)$$

where we again set $\beta = 1$.

4. Evaluation set-up

In order to test the proposed framework, we make use of 69 publicly available time-to-event datasets available in the SurvSet Python library, the NHANES [29] dataset, and two synthetically generated datasets with a partial dependence between the real time-to-event and censoring, and with differing degrees of noise (more details in Appendix B). Out of this initial set, we select for further analysis only the datasets that meet the following criteria:

- The number of samples n is at least 500: there is little labelling cost to be saved with AL with smaller datasets.
- The (5-fold cross validated) performance of the initial training data \mathcal{P}_0 has a concordance index of at least 0.65. Having models with good performance \mathcal{P}_0 is an implicit assumption for most AL frameworks [30].
- The 5-fold cross validated performance obtained by fully labelling the pool set and adding it to the training set is at least $\mathcal{P}_0 + 0.02$. Smaller gains of performance make the AL framework less relevant.

Despite the relatively mild selection criteria (particularly the second), only a few datasets meet these requirements. The datasets that do meet these standards are: rott2, vlbw, grace, NHANES, UnempDur, Framingham, and the two synthetic datasets. More details regarding these datasets are provided in Table 2. Information related to the creation of the synthetic datasets is available in the Appendix.

For the experimental setup, we configure the initial training set \mathcal{T} to be 5% of the total training size. The remaining labels are completely masked to form the pool set \mathcal{U} . We employ a batch-mode AL approach as in [1], we define a fixed batch size B and select the best B queries according to the current querying strategy. Here, B is set to be 1% of the initial pool set size \mathcal{U} .

The decision to use a constant relative batch size stems from our aim to make cross-dataset comparisons more uniform. By maintaining a proportionate batch size across different datasets

Table 2

Overview of the selected datasets, where n is the number of instances and p is the number of covariates.

Dataset	size (n, p)	censoring rate
Simul. data 1	(1000, 10)	38%
Simul. data 2	(1000, 10)	40%
rott2	(2982, 18)	57%
vlbw	(617, 56)	83%
grace	(1000, 7)	68%
NHANES	(9932, 18)	65%
UnempDur	(3241, 12)	39%
Framingham	(4699, 19)	69%

we do not account for diversity within the batch, but we ensure that the AL process is not biased by the absolute size of the dataset. This uniformity allows for more consistent evaluation across datasets of varying sizes, thus providing more meaningful insights when comparing results across datasets in Section 5. The number of iterations is restricted to $N = 200$, in line with [31].

In order to simulate a longitudinal AL scenario, we mimic conditions where the final label $(y_i, \delta_i, 1)$ of an instance in \mathcal{U} is not instantly available to the oracle when queried. Instead, a random portion of the remaining information is provided, ranging from 20% to 100%. Specifically, for an instance currently labelled as $(T_{i,k-1}, 0, 0)$ and with underlying final label being $(T_i, \delta_i, 1)$, the label is updated upon querying as follows:

$$(T_{i,k}, \delta_{i,k}, \epsilon_{i,k}), \text{ where } \begin{cases} T_{i,k} &= T_{i,k-1} + Z \cdot (T_i - T_{i,k-1}) \\ \delta_{i,k} &= \mathbf{1}(Z = 1) \cdot \delta_i \\ \epsilon_{i,k} &= \mathbf{1}(Z = 1) \end{cases} \quad (9)$$

Here, Z is a random variable that with equally likely outcomes in $\{0.2, 0.4, 0.6, 0.8, 1\}$. After a query, the labels are updated in line with the above equation². It is possible, alternatively, to sample the amount of reveal time $(T_{i,k} - T_{i,k-1})$ from a distribution bounded between 0 and T_i .

It is important to underline that the model \mathcal{M} does not have access to the outcomes of the random variable Z nor any expectation about the quantity of information the oracle will disclose upon being queried. The model has only access to the latest value of the labels y_i and to whether more information can be queried (no terminal event or terminal censoring). This represents a scenario where information such as the time since last sampling is not available, but where it is possible to inquire whether the event of interest has (already) happened at query time. For example, it is worth noting that in longitudinal studies such as clinical studies [32], time since the last sampling might be available.

Taking into account the random variable Z , let's consider p as the probability of a specific outcome $Z = z$. The number of repeated observations until the event $Z = z$ is observed is denoted by an integer-valued random variable W , which follows a geometric distribution. The

²The amount of information revealed at the k -th query for instance $x_i \in \mathcal{U}$ is generated in advance and is equal across sampling strategies.

expected value of W is expressed by: y

$$E(W) = \sum_{k=0}^{\infty} k(1-p)^{k-1}p = \frac{p}{1-p} \sum_{k=0}^{\infty} k(1-p)^k = \frac{1}{p} \quad (10)$$

Given that an instance can be queried until the event $Z = 1$ occurs, and given that this happens with probability $p = 0.2$, we conclude that an instance is queried 5 times on average before being dropped from \mathcal{U} . In practice, given the stopping criterion, the average number of queries per instance is lower, but should still follow an exponential behaviour (as confirmed by the results in Appendix A).

5. Results

We compare the proposed sampling strategies on the selected datasets with a standard 5-fold cross validation set-up, where 80% of the data is placed in \mathcal{T} and \mathcal{U} , and 20% serves as a test set to evaluate performance. We record the learning curve across the first 200 rounds and we report the 5-fold average test performance for every learning step. More specifically, we report the average concordance index across the 200 iterations. We also plot the performance obtained by the RSF model in case where all training labels with full information are provided, this serves as an upper benchmark for the framework.

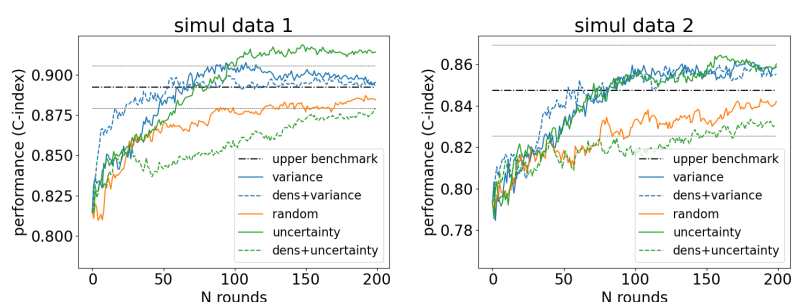


Figure 1: Comparison of the sampling strategies across simulated data. To the left, dataset with lower noise; to the right, dataset with higher noise. The learning curves are also compared against the scenario where all training instances are fully labelled, representing the ‘upper benchmark’. The fluctuations within the curves can be attributed to the inherent randomness of the RSF algorithm.

Figure 1 shows encouraging results from the simulated datasets, where the considered sampling strategies generally outperform random sampling in the proposed multi-query AL framework. In particular, the newly proposed variance+density strategy emerges, as well as the (also newly proposed) variance-based strategy. These strategies, along with the conventional uncertainty-based sampling strategy, require around 75 iterations to reach maximum performance, as opposed to over 200 iterations required for the random sampling and the uncertainty+density sampling strategy.

Additionally, it is noteworthy that, for the first simulated dataset (as shown in Figure 1, left plot), the performance of the uncertainty-based sampling strategy surpasses that of the upper

benchmark. We believe this can be attributed to the introduction of sampling bias, as discussed by [1]. As all instances are progressively exhausted, we anticipate the performance to align with the upper benchmark, mirroring the trends observed with the variance-based strategies between iterations 100 and 200.

Transitioning to real-world data that adheres to the criteria outlined in Section 4, Figure 2 showcases the results. In these datasets, a similar trend emerges with the density-corrected variance-based strategy performing best overall. Even more, random sampling outperforms the density+uncertainty sampling strategy in these instances.

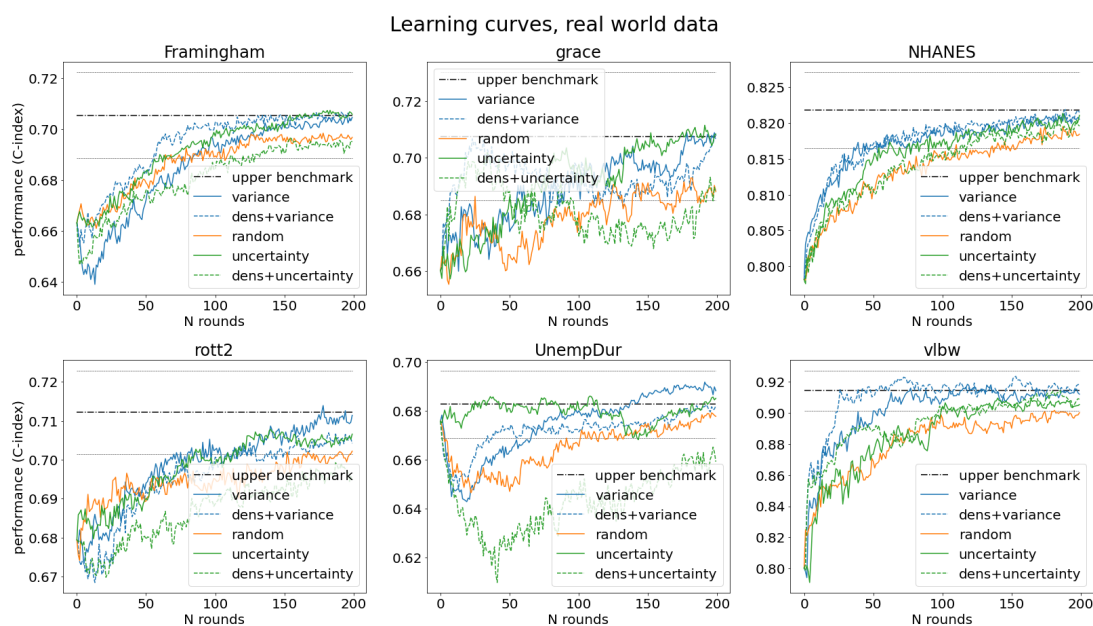


Figure 2: Comparison of the learning curves generated by the sampling strategies, across the selected real-world datasets. The learning curves are also compared against the scenario where all training instances are fully labelled, representing the ‘upper benchmark’. The fluctuations in the learning curves can be attributed to the inherent randomness of the RSF algorithm.

Similarly to the simulated datasets, the density correction leads to more efficient querying for the variance-based strategy, but does not offer any improvement for uncertainty sampling. This could potentially be attributed to instances near the population average not necessarily embodying uncertainty, but rather samples whose risk accurately matched with the population average. It is also worth noting that uncertainty sampling has been reported to perform poorly in tasks other than binary classification [33], a trend which we can somewhat corroborate for SA tasks, especially when it comes to the density-corrected uncertainty sampling.

In order to further investigate these outcomes, we also take a qualitative approach. This involves calculating the area under the curve for each sampling strategy’s performance over time until the last iteration round is reached. The underlying rationale is that more effective sampling strategies will show faster learning and therefore a larger the area under the curve. In practice, this translates to computing the average concordance index for each strategy across

all rounds, providing an insightful comparison on their relative performances. The results are given in the Table 3.

Table 3

Comparison of the considered strategies: average performance (concordance index) and standard deviation across the 5-fold iterations are shown; average and standard deviation across datasets is also reported. The highest average values are in bold.

Dataset	random	uncertainty	variance	density + uncertainty	density + variance
Framingham	0.687 ±0.015	0.692 ±0.013	0.685 ±0.015	0.681 ±0.017	0.694 ±0.017
grace	0.679 ±0.035	0.691 ±0.026	0.689 ±0.032	0.681 ±0.034	0.693 ±0.034
simul data 1	0.869 ±0.019	0.892 ±0.019	0.888 ±0.014	0.857 ±0.024	0.889 ±0.012
simul data 2	0.826 ±0.017	0.843 ±0.016	0.843 ±0.016	0.820 ±0.029	0.845 ±0.019
NHANES	0.814 ±0.006	0.816 ±0.006	0.817 ±0.007	0.814 ±0.005	0.817 ±0.006
rott2	0.695 ±0.008	0.698 ±0.013	0.699 ±0.008	0.687 ±0.013	0.695 ±0.010
UnempDur	0.665 ±0.016	0.680 ±0.013	0.675 ±0.012	0.643 ±0.019	0.672 ±0.016
vlbw	0.881 ±0.028	0.886 ±0.030	0.901 ±0.025	0.893 ±0.020	0.910 ±0.024
average	0.764 ±0.086	0.775 ±0.088	0.775 ±0.091	0.760 ±0.090	0.777 ±0.092

Our original findings are confirmed, with the newly proposed variance based strategies performing well. Among these, the one that incorporates density correction exhibits the best overall performance. The improvements observed in the density-corrected method suggest that even in straightforward cases where $\beta = 1$, incorporating density correction can increase the sampling performance. Interestingly, in the context of uncertainty sampling, the density correction seems to negatively impact performance. This could be attributed to the fact that this correction does not rectify the fundamental issue of the strategy, which has the tendency to sample not only instances that are difficult to predict, but also instances with a risk profile truly close to the population average.

Additionally, a relatively large standard deviation is observed across the folds of each dataset, indicating that the prediction algorithm’s performance is sensitive to the specific data fold in use. This variability in performance is amplified by the relatively low size of the datasets, and we observe larger datasets exhibiting smaller standard deviations.

Finally, we test the statistical significance of the observed average performances across the different sampling strategies and we test them against the performance obtained by fully labelling the training set (indicated as upper benchmark in Figures 1-2). We do so by conducting a post-hoc Friedman-Nemenyi test, setting the significance level to 0.05, as recommended in [34], and having rejected the hypothesis that all methods perform equally well with a p-value $p \approx 2.5 \cdot 10^{-4}$. Results are visualised with a critical difference diagram shown in Figure 3, which connects sampling strategies that are not statistically significantly different by a horizontal line segment.

Despite the limited sample size available for the test, the post-hoc test confirms the aforementioned trends: the newly proposed variance based methods perform reasonably well, and the ‘density+variance’ strategy significantly outperforms random sampling, whereas the ‘density+uncertainty’ approach is relegated to last place.

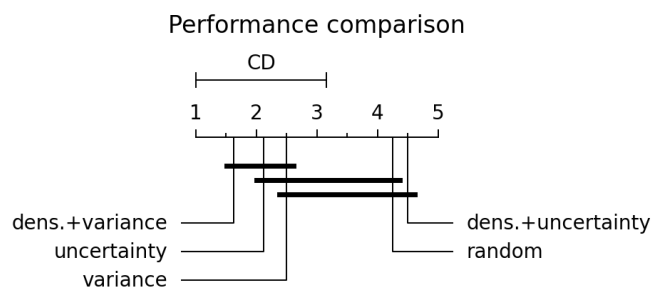


Figure 3: Nemenyi test for comparing sampling strategies over multiple datasets.

To conclude, Figures 1-2 and Table 3 suggest that a performance comparable to the fully labelled dataset is retrieved within 100 iterations, whereas for the current values of $B = 0.01$ and $E(W)$, around $E(W)/B = 500$ iterations would be needed for a standard learning procedure. This validates the idea of performing AL on a SA framework with label information evolving over time in a longitudinal fashion; this is especially relevant given that not all studies have been able to achieve significant gain over random sampling such as in [23, 24].

6. Conclusions and Future work

With this work, we lay the groundwork for a new framework within the field of AL for SA. More specifically, we show that simple AL strategies can be effective in querying labels in a multi-query set-up, where the oracle gradually reveals more information about a time to event. Additionally, our novel sampling strategies based on the variance of the rankings are performing better than uncertainty based strategies adapted from binary classification set-ups. In particular, our density corrected variance based approach ‘density+variance’ is performing best overall.

We also made modifications to the existing Random Survival Forests (RSF) model architecture. This allows RSF to utilise partial information about labels during the prediction phase and it is noteworthy that we assumed the model does not have prior knowledge about the amount of information (partial or full) revealed during the querying phase. We have tested these scenarios on several datasets, proving that simple AL strategies are efficient for this task and outperform random sampling.

Moving forward, our future work will include testing more state-of-the-art sampling strategies based on expected error minimisation [18], hierarchical sampling [35], as well as taking into account batch diversity [7], and we will eventually evaluate such strategies on a larger number of datasets. Another direction of interest lies in the analysis of the added value of the conditional approach within RSF models. Furthermore, we plan to relax the assumptions imposed on the oracle in Section 4. More specifically, we will examine the scenarios where the oracle is aware of the elapsed rounds since the last instance query, or can estimate the amount of updated label information ($T_{i,k} - T_{i,k-1}$) for instances in the pool set.

A. Querying analysis

Here we provide more details on the distribution of the querying across instances. More specifically, we examine two distinct datasets as a form of sanity check. The aim is to spot any peculiarities in the querying pattern and ensure that our process are consistent with theoretical expectations.

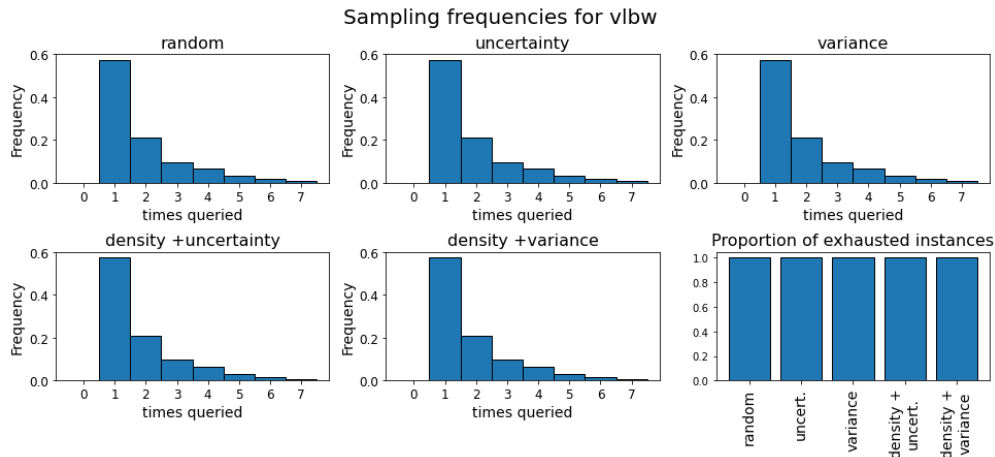


Figure 4: Distribution of the querying across strategies for a small dataset. All instances have been fully exhausted in this case.

We first consider *vlbw* (Figure 4), a relatively small dataset (617 instances), and interestingly, all its instances are fully exhausted within 200 iterations. This is less than expected, as (at most) 500 iterations should be needed instead; the deviation can be explained due the small number of instances under consideration. Finally, the distribution of the sampling across instances shows an exponential decay, which is in line with Equation (10).

We report the same analysis on the substantially larger (4699 instances) Framingham dataset, in Figure 5. Here, not all instances are sampled until exhaustion, which is to be expected, and the truncation effect generated by the stopping criterion can be observed. In particular, we observe that a substantial amount of instances has never been sampled, whereas the instances that have been sampled at least once follow an exponential decay as expected.

Furthermore, once the last iteration is reached, it is worth noting that all considered strategies show higher rates of fully exhausted instances compared to the random strategy (bottom right plot), likely due to a tendency of the sampling strategies to repeatedly sample the most informative instances until all information is revealed.

The above patterns are representative of all considered datasets, with the remaining plots being available in the public repository.

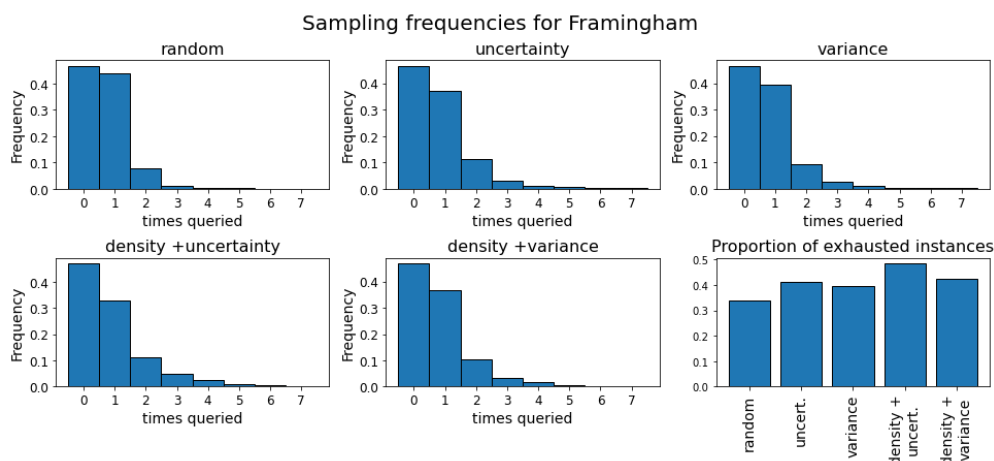


Figure 5: Distribution of the querying across strategies for a large dataset. The truncation of the process can be observed, and not all instances have been fully exhausted (bottom right plot)

B. Synthetic data generation

In this Section, we provide more details on the generative process for the synthetic datasets. The full procedure is available in the shared repository. The generative process consists of two parts: the first one consists in generating the real times to event T_e (without censoring) for each instance; the second part generates the censoring event T_c using some of the previous information to simulate partial dependence. Once both events are generated, we build $T = \min(T_e, T_c)$ and $\delta = \mathbf{1}(T_e \geq T_c)$ as usual.

We start by sampling the covariates for $X = (X_1, \dots, X_{10})$ from a standard normal distribution, and coefficients for $\beta = (\beta_1, \dots, \beta_{10})$ from a uniform distribution in $[-2, 2]$. Next, an interaction effect is added between pairs (X_1, X_2) and (X_3, X_4) by mapping $X_1 \rightarrow X_1 \cdot X_2$ and $X_3 \rightarrow X_3 \cdot X_4$. The dot product $\eta_t = X \cdot \beta$ is computed and scaled within the interval $[-5, 5]$ by means of a sigmoid function. Noise $\epsilon \in \{0.01, 0.3\}$ is added at this point³. More specifically, Gaussian noise of scale ϵ times the standard deviation of the sigmoid transformed η is added, so to control the proportion of signal to noise. Given the (noisy) risk $\tilde{\eta}$, we compute the hazard rate as $h_t = 0.1 \exp(\tilde{\eta})$ and sample the times to event from a Weibull distribution with shape $k = 1.2$ and scale $\lambda = 1/h_t$, where time to event are expected to be larger for smaller values of $\tilde{\eta}$.

To simulate partial conditional censoring we sample 4 random covariates X_i and respective β_i from the previous process, and add 6 new X_i and β_i as independent, external features for censoring. The dot product $\eta_c = X \cdot \beta$, the added noise and the hazard rate h_c are computed as before, and the event times are sampled from a Weibull distribution with parameters $k = 1.1$ and $\lambda = 1/h_c$ respectively.

³simul data 1 has $\epsilon = 0.01$, whereas simul data 2 has $\epsilon = 0.3$

References

- [1] B. Settles, *Active learning*, Synthesis Lectures on Artificial Intelligence and Machine Learning, Springer Cham, 2012. doi:10.1007/978-3-031-01560-1.
- [2] S. Budd, E. C. Robinson, B. Kainz, A survey on active learning and human-in-the-loop deep learning for medical image analysis, *Medical Image Analysis* 71 (2021). doi:10.1016/j.media.2021.102062.
- [3] M. Yuan, H.-T. Lin, J. Boyd-Graber, Cold-start active learning through self-supervised language modeling, *ArXive* (2020). URL: <http://arxiv.org/abs/2010.09535>.
- [4] B. Settles, *Active learning literature survey*, Technical Report, 2009. URL: <https://minds.wisconsin.edu/handle/1793/60660>.
- [5] P. Ren, Y. Xiao, X. Chang, P.-Y. Huang, Z. Li, B. B. Gupta, X. Chen, X. Wang, A survey of deep active learning, *ACM Comput. Surv.* 54 (2021). URL: <https://doi.org/10.1145/3472291>. doi:10.1145/3472291.
- [6] G. Hacohen, A. Dekel, D. Weinshall, Active learning on a budget: Opposite strategies suit high and low budgets, *Proceedings of the 39th International Conference on Machine Learning* (2022). URL: <https://github.com/avihu111/TypiClust>.
- [7] A. L. Chandra, S. V. Desai, C. Devaguptapu, V. N. Balasubramanian, On initial pools for deep active learning, in: *NeurIPS 2020 Workshop on Pre-registration in Machine Learning*, PMLR, 2021, pp. 14–32.
- [8] T. Viering, M. Loog, The shape of learning curves: A review, *IEEE Transactions on Pattern Analysis and Machine Intelligence* 45 (2023) 7799–7819. doi:10.1109/TPAMI.2022.3220744.
- [9] P. Wang, Y. Li, C. K. Reddy, Machine learning for survival analysis: A survey, *ACM Computing Surveys* 51 (2019) 1–39. doi:10.1145/3214306.
- [10] B. Vinzamuri, Y. Li, C. K. Reddy, Active learning based survival regression for censored data, *Proceedings of the 2014 ACM International Conference on Information and Knowledge Management* (2014) 241–250. URL: <http://dx.doi.org/10.1145/2661829.2662065>. doi:10.1145/2661829.2662065.
- [11] M. Z. Nezhad, N. Sadati, K. Yang, D. Zhu, A deep active survival analysis approach for precision treatment recommendations: Application of prostate cancer, *Expert Systems with Applications* 115 (2019) 16–26. doi:10.1016/J.ESWA.2018.07.070.
- [12] D. R. Cox, Regression models and life-tables, *Journal of the Royal Statistical Society: Series B (Methodological)* 34 (1972) 187–202. doi:10.1111/j.2517-6161.1972.tb00899.x.
- [13] H. M. Gomes, J. Read, A. Bifet, J. P. Barddal, J. A. Gama, Machine learning for streaming data: state of the art, challenges, and opportunities, *ACM SIGKDD Explorations Newsletter*, 2019. Available at <https://doi.org/10.1145/3373464.3373470>.
- [14] H. Ishwaran, U. B. Kogalur, E. H. Blackstone, M. S. Lauer, Random survival forests, *Annals of Applied Statistics* 2 (2008) 841–860.
- [15] D. D. Lewis, J. Catlett, Heterogeneous uncertainty sampling for supervised learning, *Machine learning proceedings* (1994) 148–156.
- [16] Y. Freund, H. S. Seung, E. Shamir, N. Tishby, Selective sampling using the query by committee algorithm, *Machine learning* 28 (1997) 133.
- [17] W. Cai, Y. Zhang, J. Zhou, Maximizing expected model change for active learning in regression, in: *2013 IEEE 13th international conference on data mining, IEEE*, 2013, pp.

- 51–60.
- [18] N. Roy, A. McCallum, Toward optimal active learning through monte carlo estimation of error reduction, *Proceedings of the 18th international conference on Machine learning 2* (2001) 441–448.
 - [19] A. Kapoor, E. Horvitz, S. Basu, Selective supervision: Guiding supervised learning with decision-theoretic active learning., in: *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, volume 7, 2007, pp. 877–882.
 - [20] L. Breiman, Random forests, *Machine Learning* 45 (2001) 5–32. URL: <https://link.springer.com/article/10.1023/A:1010933404324>. doi:10.1023/A:1010933404324.
 - [21] N. Bhosle, M. Kokare, Random forest-based active learning for content-based image retrieval, *Int. J. Intelligent Information and Database Systems* 13 (2020) 72–88.
 - [22] S.-H. Jung, H. Y. Lee, S.-C. Chow, Statistical methods for conditional survival analysis, *Journal of Biopharmaceutical Statistics* 28 (2018) 927–938. doi:10.1080/10543406.2017.1405012.
 - [23] K. Tomanek, F. Laws, U. Hahn, H. Schütze, On proper unit selection in active learning: co-selection effects for named entity recognition, in: *Proceedings of the NAACL HLT 2009 Workshop on Active Learning for Natural Language Processing*, 2009, pp. 9–17.
 - [24] B. C. Wallace, K. Small, C. E. Brodley, T. A. Trikalinos, Active learning for biomedical citation screening, in: *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, 2010, pp. 173–182.
 - [25] P. Jain, S. Vijayanarasimhan, K. Grauman, Hashing hyperplane queries to near points with applications to large-scale active learning, *Advances in Neural Information Processing Systems* 23 (2010).
 - [26] F. E. Harrell, R. M. Califf, D. B. Pryor, K. L. Lee, R. A. Rosati, Evaluating the yield of medical tests, *Journal of the American Medical Association* 247 (1982) 2543–2546.
 - [27] F. T. Liu, K. M. Ting, Z.-H. Zhou, Isolation forest, in: *2008 eighth IEEE international conference on data mining, IEEE, 2008*, pp. 413–422.
 - [28] A. McCallum, K. Nigam, et al., Employing em and pool-based active learning for text classification., in: *Proceedings of the 15th international conference on Machine learning*, volume 98, Citeseer, 1998, pp. 350–358.
 - [29] Centers for Disease Control and Prevention (CDC), National Center for Health Statistics (NCHS), National Health and Nutrition Examination Survey Data, NHANES III plan and operations procedures manuals, 1997.
 - [30] P. Felt, E. K. Ringger, K. D. Seppi, K. Heal, R. Haertel, D. Lonsdale, First results in a study evaluating pre-annotation and correction propagation for machine-assisted syriac morphological analysis., in: *8th international conference on Language Resources and Evaluation*, 2012, pp. 878–885.
 - [31] K. Konyushkova, R. Sznitman, P. Fua, Learning active learning from data, *Advances in neural information processing systems* 30 (2017).
 - [32] J. Gabrielsson, D. Weiner, *Pharmacokinetic and pharmacodynamic data analysis: concepts and applications*, CRC press, 2001.
 - [33] A. Holub, P. Perona, M. C. Burl, Entropy-based active learning for object recognition, in: *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, IEEE, 2008, pp. 1–8.

-
- [34] J. Demšar, Statistical comparisons of classifiers over multiple data sets, *The Journal of Machine learning research* 7 (2006) 1–30.
 - [35] S. Dasgupta, D. Hsu, Hierarchical sampling for active learning, in: *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 208–215.