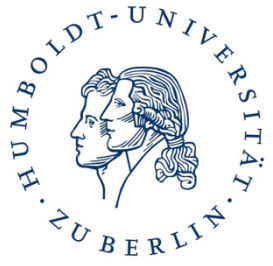

ACTIVE LEARNING IN CREDIT SCORING

Master Thesis

for acquiring the degree of
Master of Science (M.Sc.) in Information Systems

at the School of Business and Economics
of Humboldt-Universität zu Berlin

submitted by
Lukas Kolbe
Student no. 554673

Examiner
Prof. Dr. Stefan Lessmann

Berlin, 29.07.2022

Outline

List of Tables

List of Algorithms

List of Figures

List of Abbreviations

AL ..... Active Learning
AUC ..... Area Under the Receiver Operating Curve
BMDR ..... Discriminative and Representative Queries for Batch Mode Active Learning
BS ..... Brier Score
CORS ..... Core-Set – Greedy Kcenter
CPL ..... Cost Per Loan
CV ..... Cross-Validation
DENS ..... Graph Density
DW ..... Weighted Density
ECPL ..... External Cost Per Loan
EER ..... Expected-Error-Reduction
EMP ..... Expected Maximum Profit
ERM ..... empirical risk minimization
FN ..... False Negative
FNR ..... False Negative Rate
FP ..... False Positive
HM ..... H-measure
i.i.d. ..... Independent and Identically Distributed
ICPL ..... Internal Cost Per Loan
IFRS9 ..... International Financial Reporting Standard 9
KNN ..... K-Nearest-Neighbor Algorithm
LAL ..... Learning by Active Learning
LR ..... Logistic Regression
PCC ..... Percentage Correctly Classified
QBC ..... Query-by-Committee
QUIRE ..... Querying Informative and Representative Examples
RND ..... Random Sampling
SMOTE ..... Synthetic Minority Over-sampling Technique
SPAL ..... Self-Paced Active Learning
TN ..... True Negative
TP ..... True Positive
UNC ..... Uncertainty Sampling

1. Introduction

Credit Scoring is concerned with the data driven acceptance or rejection of loan applications. It relies on data from past credit contracts, where actual repayment behaviour of borrowers can be compared against the model's initial assessment and prediction. Naturally, such data only provides risk labels (i.e., GOOD or BAD) for instances where credit was granted (Hand, 2005). Any rejected instance cannot be used for supervised learning, as no ground truth is available. This introduces sample selection bias into the data (Banasik, et al., 2003): A feedback loop begins and over time, the population of borrowers in the data might become unrepresentative of the loan applicant population from which they are selected. While a complete mitigation of such biases might be unfeasible, looking at the data through a too narrow lens inhibits the model's ability to capture trends and shifts in population behaviour (Pavlidis, et al., 2012). Ultimately, this hinders performance (Verstraeten & Van den Poel, 2005). Intuitively, the least biased data would be gained by granting credit to a representative sample of applicants, including those that any existing model deems risky. The problem: accepting bad risks can lead to significant cost (Hand & Henley, 1997; Lessmann, et al., 2015).

While the issue of sample selection bias has been discussed extensively in the financial literature, one promising procedure has so far been neglected: Through smart querying of rejected instances for labelling (i.e., granting credit), Active Learning (AL) could help balance the trade-off between data quality and cost of data acquisition, fulfilling two goals: maximizing generalizability while keeping cost low. However, though AL has seen a lot of attention in the machine learning literature (Settles, 2009; Fu, et al., 2012) and deep learning literature (Ren, et al., 2022), applications in Credit Scoring have been underrepresented.

This thesis contributes to closing this gap. It sets out to answer multiple research questions: Can AL algorithms contribute to improving predictive power of Credit Scoring models? If so, under which circumstances? Which (group of) model(s) performs best? Can AL help reduce misclassification cost? It provides a concise survey of the Credit Scoring and Active Learning literature (sections 2 and 3) before benchmarking the performance of multiple Active Learning models by means of a thorough experiment across several real-world Credit Scoring datasets[1]. The

[1] All codes and datasets used for this thesis can be found in the appendant online repository at https://github.com/lukekolbe/AL-in-CreditScoring/

main part of this thesis paper focuses on the experiment design (section 4) and the subsequent, thorough analysis of the promising empirical results (section 5). It concludes with a critical discussion (section 6) and a brief conclusion (section 7).

2. Credit Scoring

The field of Credit Scoring is concerned with building the most effective scorecard models for credit risk default prediction. For their decision to grant loans or not, commercial lenders rely heavily on data, which is processed through decision heuristics and predictive models (Petrides, et al., 2020). The data is crucial in informing the binary classification problem of predicting whether a debtor will pay back the loan (GOOD) or default (BAD). Having a scorecard that gives accurate predictions serves both the lenders by protecting their assets as well as the debtors by avoiding overcommitment (Hand & Henley, 1997). The credit industry in general is an important economic factor for businesses and individuals alike (Lessmann, et al., 2015). As such, Credit Scoring is subject to extensive regulation by the International Financial Reporting Standard 9 (IFRS9) and the Basel II/III accords (Verbraken, et al., 2014; Verbeke, et al., 2017; Pavlidis, et al., 2012). Hand & Henley (1997) give an introduction into the field, outlining the challenges and streams of research, introducing classification methods, and suggesting performance measures.

2.1. Selection Bias in Credit Scoring

Credit Scoring suffers from one important flaw: selection bias (Banasik, et al., 2003; Hand & Adams, 2014). While lenders may have data on a large pool of credit applicants, the outcome (i.e., whether a loan is a good or a bad risk) is only ever known for accepted loan applications. As such, the scorecard model selects its own future training data while simultaneously trying to minimize the occurrence of one class (BAD). Hence, the goals of maximizing profit on one hand and assuring high quality (i.e., representativeness) of the training data on the other are opposed to one another: From a profit perspective, a large imbalance between the two classes is preferable – the bank ideally wants to only hand out good loans. From a data and modelling perspective though, this imbalance is problematic if one considers the fact that Credit Scoring is never a singular event, but rather a process that is applied over longer periods of time to an influx of applicants, whose data in turn is used for future predictions.

In short: The better a scorecard, the more pronounced the class imbalance in the data will become over time. This inherent flaw can lead to a deterioration of predictive power (Pavlidis, et al., 2012), which is accelerated through demographic changes to the population of loan applicants, or changes

to the economic environment. Such changes to the distribution of explanatory variables are called population drift (Pavlidis, et al., 2012; Hofer & Krempl, 2013; Hand & Adams, 2014; Hand & Henley, 1993). These issues are exacerbated through verification latency (Marrs, et al., 2010), i.e., the prediction refers to an outcome that lies far in the future, hence there is considerable delay before the label is revealed.

This is particularly true for Credit Scoring applications, where the default status of a loan is uncertain for long periods (Hofer & Krempl, 2013). Selection bias, population drift and verification latency all are mechanisms that lead to covariate shift: the training data represents a different distribution than the test data (Hand & Adams, 2014). This violates the independent and identically distributed (i.i.d.) assumption that is prevalent in most machine learning applications, but often violated in practice (Farquhar, et al., 2021).

In the process, a once well-calibrated scorecard may become increasingly biased over time and lose predictive power in the process. This translates to potential losses or forgone profits for lenders (Banasik, et al., 2003; Hand, 2005). Authors who use data consisting of randomly accepted loans confirm that sample selection bias does have a modest impact on classification performance (Banasik, et al., 2003; Verstraeten & Van den Poel, 2005).

2.2. Literature on Classifier Selection

There are several streams in the Credit Scoring literature. One is concerned with increasing the predictive power of scorecards by comparing or improving classification models. This research field takes the constraints of self-selection and class imbalance in the data as facts that are to be worked with and strives to maximize predictive power regardless of their presence. Extensive research has been undertaken to find the best-performing classification models across several Credit Scoring datasets and performance metrics. Baesens, et al. (2003) conduct a comparison of single-model classification algorithms for Credit Scoring. Their thoroughly tested results show that while complex models such as Neural Networks and Support Vector Machines perform best, simpler models such as Logistic Regression achieve very good results, too.

Lessmann, et al. (2015) build on this, comparing conventional single-model classification techniques to more elaborate ensemble techniques employing multiple models at once. In their extensive and statistically confirmed benchmarking study they find the best-performing model to be a complex heterogeneous ensemble of several classifiers.

Their research has been further extended upon by other authors who investigate new methods such as genetic algorithms and fuzzy logic (Louzada, et al., 2016), or merge machine learning and econometric methodology for better interpretability of the results (Dumitrescu, et al., 2022), or consider techniques that optimize cost directly (Petrides, et al., 2020).

2.3. Literature on Bias Mitigation

A second stream of the Credit Scoring literature is concerned with the mitigation of the biases and imbalances inherent in the data. Reject inference tries to overcome the sample selection bias using additional data on rejected instances (for which the explanatory variables are known, but not their label). As is the case with this experiment, such data is not always available. Many reject inference approaches come with another caveat: they rely on the assumption that accepts can be used to represent rejects (Joanes, 1994). While early approaches suffered from prohibitively strict assumptions (Hand & Henley, 1993) and yielded mixed results (Banasik, et al., 2003), recent research has achieved greater success by employing, e.g., self-learning (Kozodoi, et al., 2020) or deep-learning techniques for reject inference (Mancisidor, et al., 2020).

(Re-)sampling techniques try to mitigate imbalance-related issues by changing the composition of the training data. These techniques have received a lot of attention in the machine learning literature and have also been applied to Credit Scoring: Over-sampling the minority class and under-sampling the majority class to balance out the data are well-researched and have shown to be effective in certain cases (García, et al., 2012; Marqués, et al., 2013). Marqués et al. also show that the Synthetic Minority Over-sampling Technique (SMOTE) – which uses existing minority class instances to generate artificial instances to balance out the classes – can be a useful tool in Credit Scoring under imbalance constraints. Bahnsen et al. (2014) on the other hand find SMOTE to perform worse than conventional over- and under-sampling. Pavlidis, et al., (2012) suggest adaptive filtering to accommodate population drift. Their method gradually reduces the overall influence that old training data has on the model building process compared to more recent data.

Reviewing re-sampling strategies in a Credit Scoring context, Crone & Finley (2012) suggest Active Learning as an alternative method to augment imbalanced training data through intelligent instance sampling. Despite their suggestion, no further research has since combined Credit Scoring and Active Learning, which is motivation for this thesis to explore AL in a Credit Scoring setting. Section 3 will give a conceptual overview of AL and outline recent research in the field.

3. Active Learning

The main principle of any supervised machine learning task is to train a learner on data for which the ground truth label of each instance is known. Ideally, the trained learner is then capable of making better-than-random predictions on unlabelled, unseen data of the same structure. The emergence of AL as a research field within machine learning is motivated by the fact that manually labelling training data takes time and effort, and may require expertise – all of which lead to annotation cost (Cohn, et al., 1994; Settles, 2009; Fu, et al., 2012). In many cases, this annotation cost is instance-dependent instead of static (Settles, et al., 2008) – which makes the trade-off between informativeness and labelling cost an even more interesting field of research. Applications that often rely on data manually labelled data include computer vision and natural language processing (Settles, 2009).

AL is concerned with using the available training data most effectively, thus keeping labelling cost to a minimum. The way in which this is achieved is usually by giving the learning algorithm, or a proprietary AL-algorithm, some degree of control over the input data on which the model is trained. The hypothesis is that through this mechanism, AL achieves better generalization compared to a random sample of the same size (Cohn, et al., 1994). AL can help streamline the labelling process by selecting only the most useful instances for labelling. This avoids redundancies in the data and shifts focus to those instances that help improve performance the most. AL-models can be sorted into several types of models (Fu, et al., 2012; Zhan, et al., 2021), depending on their utility function for instance selection (Settles, 2009): AL-models based on *uncertainty* (informativeness) criteria, AL-models considering data *diversity* (representativeness), and *combined* AL-models that apply both uncertainty and diversity criteria. Each of these types is represented in this study, the selected models being introduced in section 4.3.

Yang & Loog (2018) benchmark AL-models that are built for use with Logistic Regression (LR) on multiple classification problems. They choose LR because of its prevalence in the industry as well as it being the classifier of choice in a host of AL literature. The reason for that is the fact that LR puts out posterior class probabilities, which are exploited in many AL utility metrics. The authors find that AL-models outperform a random sampling of instances in many cases but cannot find a single model that does so consistently across all 47 datasets in their benchmark.

Zhan et al. (2021) compare well-proven AL models as well as newly emerged techniques. They find that AL performance is highly dependent on the type of learning problem

(e.g., binary vs. multi-class classification) and that simpler, uncertainty-based models often outperform more advanced approaches that combine several uncertainty and diversity criteria.

Attenberg & Provost (2011) state that annotation cost may not merely be a function of time and effort. Labels might instead be revealed only after costly incentives, experiments, or interventions. This is also the case for Credit Scoring, where data is not labelled by a human expert. To label any instance, a loan must be granted, and some time must pass before the lender will know if a loan is bad or good. They remark that real-world implementations of AL are scarce and criticise the lack of standards and best-practices to help navigate the vast landscape of AL-models. Difficult decisions include the choice of the base-learner and AL-model when no labelled data is initially available, as well as budgetary considerations when allocating annotation resources between training and validation data. They agree with Settles (2009), who suggests using AL to fine-tune models that can rely on an existing pool of labelled data, or when some specifications are dictated ex ante for domain-specific reasons. Attenberg & Provost (2011) name Credit Scoring as an example, where regulatory requirements make Logistic Regression a popular model choice.

Farquhar et al. (2021) investigate bias in active learning settings. Apart from overfitting bias, which is an issue in any supervised learning application, AL introduces statistical bias to the data by constructing a sample that is non-representative of the population from which it was selected. An interesting finding in their paper is that when overfitting bias and statistical bias have different signs, they may cancel each other out under some circumstances.

Similarly, Richards, et al. (2011) show promising results when employing AL to mitigate sample bias in astronomical image classification. These findings are particularly interesting for the Credit Scoring domain, since any training data is inherently biased toward the majority class. Thus, AL may be a useful tool to mitigate bias in Credit Scoring as well.

Motivated by these findings and suggestions, the premise of the experiment described in section 4 is to investigate whether it is beneficial to augment the training data by having AL-models select instances to grant credit to from the pool of rejected credit applications. By partially shifting the focus of instance selection to increased generalizability instead of strictly selecting instances based on profitability criteria, the bias inherent in the credit data may be offset to some extent. The ratio by which instances are selected via AL will determine the strength of the effect.

In short: the experiment examines whether an increase in short-term risk (selecting risky instances) can be offset by an increase predictive performance through an AL-driven reduction of self-selection bias.

4. Experiment Design

This thesis submits an AL-experiment that, over several generations, selects instances from a pool of rejected credit applications in a binary classification problem. Consequently, this AL-experiment is of the pool-based type (Fu, et al., 2012; Zhan, et al., 2021; Settles, 2009).

After splitting and rescaling the data, the selection cycle begins from a warm start, meaning a sample of labelled instances to first train the classifier on is drawn at the beginning[2]. This provides a common baseline across models and avoids the cold-start problem (Attenberg & Provost, 2011), a performance-inhibiting cycle of bad selections made by a badly trained model. Also, a warm start is closer to practical applications (Konyushkova, et al., 2017), where it can be safely assumed that there is some historic loan data upon which the credit scorecard is built.

Thus, in the first period there is a small initial labelled set $L_0 = \{(x_1, y_1), ..., (x_m , y_m)\}$ and a large unlabelled data pool $U = \{x_1,...,x_n\}$, where each instance $x_i \in \mathbb{R}^d$ is a d-dimensional feature vector and $y_i \in \{0, 1\}$ is the binary class label of x_i. Over several iterations, applicants are randomly drawn from U, scored by the classifier, and accepted or rejected based on their predicted probability of default (lowest to highest). The overall number of accepted instances depends on the tuned classification threshold (cut-off).

From the pool of rejected applicants, further instances are then selected by the respective AL-model: In each generation of each round, the active learner selects a batch r from a set of rejected instances R and queries their labels (i.e., accepts the loans). L and U are then updated, and the LR classifier is re-trained on L. A prediction on the unseen test data is made, and result scores for all relevant metrics are computed. Over several periods, the pool of labelled data L grows through this mixture of score- and AL-selected data. The distribution of the labelled data increasingly diverges from that of the initially labelled data (the warm-start sample). The process terminates after a pre-determined number of periods (p=10). Once the loop has run over all folds, the average performance in each metric is computed across all rounds.

Below, Algorithm A contains the complete logical sequence. Several different specifications of this experiment are run in parallel and compared. These experimental parameters and several critical assumptions are explained in section 4.1.

[2] In the context of this experiment, labelling an instance means granting a loan, an unlabelled instance has either not yet been evaluated by the scorecard, or it has been rejected.

Algorithm A: Active Learning Loop

models: pool of AL and baseline models	AL: active learner
D: available data (full dataset)	LR: tuned LR classifier
K: number of cross-validation folds (rounds)	LR_p: fitted classifier at end of p
k: current fold (round)	A: applicant sample in period p
D_{K-1}: training data (K-1 folds)	$\hat{\tau}$: optimal threshold
D_k: test data (fold k)	a: accepted applicants, a $\subset A$
P: total number of periods	R: rejected applicants, R $\subset A$
p: current period (generation)	r: AL-selected sub-sample of R; $r \subset R$
U_p^k: unlabelled training data (at k, p)	s_p: vector of result scores at end of p
L_p^k: labelled training data (at k, p)	S^k: matrix of result scores at fold k
	$\bar{S}_{AL}$: average scores per model across folds

input: models, D, K, P, p=0, LR

randomly split **D** into **K** folds
for AL in models:
  for each k in K=10:
    while p = 0:
      split training data $\mathbf{D_{K-1}}$ into $\mathbf{L_0^k} \cap \mathbf{U_0^k}$ [a]
      train robust scaler on $\mathbf{L_0^k}$; transform all data **D** [b]
      train classifier **LR** on $\mathbf{L_0^k} \longrightarrow \mathbf{LR_0}$
      make initial prediction on $\mathbf{D_k}$, collect and store $\mathbf{s_0}$ (performance baseline)
      p=**p**+1
    while $\mathbf{1 \leq p \leq P}$**:**
      draw applicant sample $\mathbf{A} \subset \mathbf{U_{p-1}^k}$, set $\mathbf{U_p^k} = \mathbf{U_{p-1}^k} \setminus \mathbf{A}$
      make prediction with $\mathbf{LR_{p-1}}$ on **A**; rank predicted scores (ascending)
      within boundaries set by $\hat{\boldsymbol{\tau}}$, accept best ranking $\mathbf{a} \subset \mathbf{A}$ instances [c]
      set $\mathbf{R} = \mathbf{A} \setminus \mathbf{a}$
      pick $\mathbf{r} \subset \mathbf{R}$ with **AL**, discard **R**
      add {**a**, **r**} to labelled data: $\mathbf{L_p^k} = \mathbf{L_{p-1}^k} \cup \{\mathbf{a}, \mathbf{r}\}$
      re-train **LR** on $\mathbf{L_p^k} \longrightarrow \mathbf{LR_p}$
      make prediction on $\mathbf{D_k}$, compute results $\mathbf{s_p}$
      return $\mathbf{U_p^k}$, $\mathbf{L_p^k}$, $\mathbf{LR_p}$, $\mathbf{s_p}$, **p**=**p**+1
    from all $\mathbf{s_p}$ ($p \in \{0, 1, \ldots, P\}$) construct matrix of shape (P × metrics) $\longrightarrow \mathbf{S^k}$
  end for
  compute element-wise averages for each period and metric across all $\mathbf{S^k}$ [d]:
  $\bar{\mathbf{S}}_{\mathbf{AL}} = \frac{\mathbf{1}}{\mathbf{K}} \sum_{\mathbf{k}}^{\mathbf{K}} \mathbf{S^k}$
end for

[a] Splits are random, shuffled.
[b] Any train, test, labelled and unlabelled data are managed via their indices in the full data. Any operation on D (such as scaling) can therefore be applied to all data, even after splitting.
[c] see Figure 4-1 for how the threshold and AL-ratios work together, and Algorithm C for the determination of $\hat{\tau}$
[d] Summation here means dot-wise addition of K result matrices, followed by dot-wise division by constant K.

4.1. Experimental Parameters

Rounds & Generations. To run the experiment, the data is split into five folds, each serving as the test set once while the remaining four folds are used as training data. This means there are five rounds (iterations) per combination of AL-model and dataset. Within each round, there are 10 periods (generations) in which applicants are selected and added to a growing set of labelled instances. Due to the nature of an AL-experiment, where some data is selected and some is discarded, not all training data is ever used to train the model. The overall number of accepted instances depends on the size of the dataset and the classification threshold (Verbeke, et al., 2017), see section 4.8.

Sample Sizes. In the first generation of each round, all AL-models start off with the same labelled sample. This sets a common baseline for the subsequent comparison. The size of the initial sample is 10% of the available training data, or 500 instances at maximum. The initial sample size is capped at 500 instances because if the initial sample becomes very large, the number of instances that are sampled in each period must increase accordingly to have an impact on the model. Otherwise, the contributions of AL-selections would be dominated by the large initial set, making differences between models difficult to discern. Also, since some AL-models solve complex problems, querying large samples each round would be computationally expensive.

Acceptance ratios. Most AL applications inherently trade-off between labelling cost and increased performance through AL-selected data. A good model maximizes the positive impact gained through a limited number of queries. This is true for the experiment at hand, where different ratios of score-based versus AL-based selections are applied. The selections in each period are in part based on the scores predicted by the classifier: the best-ranking samples are selected; the rest are rejected. Further instances are then selected via AL. Since there is no indication in the literature which ratio is best used (Yang & Loog, 2018), several ratios by which instances are selected via AL relative to the score-based selections are tested in the experiment. In total, five AL-ratios $r \in \{0.1, 0.2, ...,0.5\}$ are employed and benchmarked against one another. The respective AL-ratio of each model is indicated by the suffix *r01, ..., r05*.

Importantly, the total number of accepted instances depends on the classification threshold. In case of the SCORE model, it is desirable to only accept instances whom the model deems GOOD. Specifying a fixed number of instances to accept does not make sense in that regard, since it might force the model to keep instances it expects to be risky (or the number of instances would have to be very small). As illustrated in Figure 4-1, this has implications on the selection process:

The number of instances to be kept is determined by applying the threshold to the initial prediction on the applicant sample of each period. The baseline SCORE model accepts all instances that fall below the threshold. When an AL-model is applied, the same number of instances is kept, the difference being that the selection is only in part done via the predicted score. The other part is instead selected by the AL-model. The AL-ratio governs how the shares are distributed between score-based and AL-based selections. Generally, score-based selections are carried out first, then the AL-model selects its share from the population of instances that were rejected.

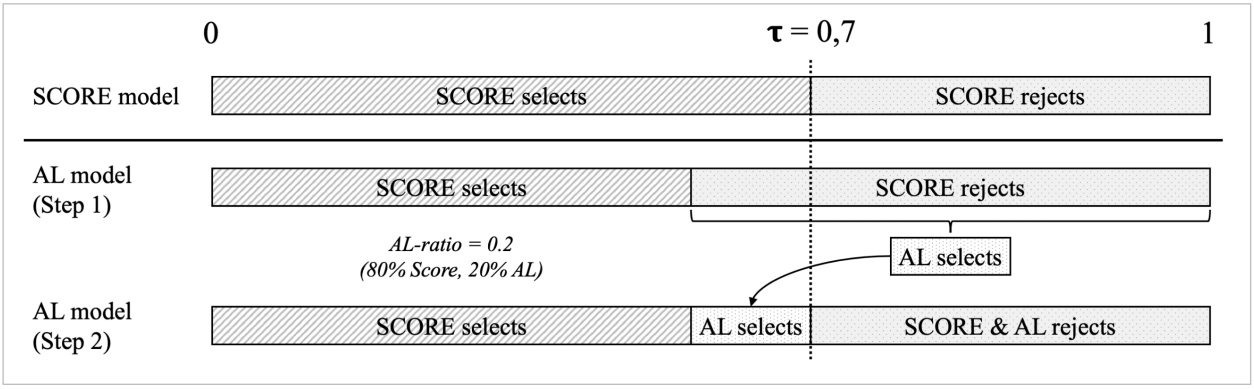

Figure 4-1: Selection of Instances Dependent on Threshold and AL-Ratio

Sample Weights. A second variable model parameter is the optional setting of sample weights in the LR classifier. With sample weights, each AL-selected instance is weighted higher than score-based selections when training the classifier. The magnitude of the weight is inversely proportional to the AL-ratio, e.g., at an AL:score ratio of 1:9, the weights are set 9:1. If the AL-ratio increases, the sample weights decrease accordingly. When 50% of selections are made by AL, all labelled instances in the training data are weighted the same. This balances the overall impact that AL-selected instances have on the model parameterization.

Intuitively, this seems sensible: if AL-selections are costly, their number should be kept small. Setting higher sample weights for any AL-selected instance in the training data maximizes the attention that the learner pays to such instances. It was confirmed in preliminary experiments that setting sample weights as described did increase the overall performance of AL algorithms on some datasets, motivating the decision to add sample weights as an option to the benchmark. The respective weight option of each model is indicated by the suffix *wT* or *wF*. All possible combinations of acceptance ratios and sample weights are run, meaning that results are collected for 10 versions of each AL-model.

4.2. Main Assumptions

Certain assumptions are made for this experiment. First, it is assumed that labels (i.e., a risk being GOOD or BAD) are available at the end of each period. This translates to very short credit periods, which are likely not realistic. In practice, the terms of a loan are individually agreed upon, so the credit period would be different between datasets and instances. Second, it is assumed that loss in case of default is 100%. In real life applications, a loan can default at any time, usually resulting in a partial loss only (Bahnsen, et al., 2014; Petrides, et al., 2020). This also would also have severe implications on misclassification cost. Lacking information on credit periods and time-of-default in the data, the above assumptions seem reasonable.

Adjacent to the assumption of 100% loss at default, it is also assumed that misclassification cost (when predicting a good loan as bad or a bad loan as good) is inversely proportional to the prior default rate in the data (i.e., the less frequent defaults are in a dataset, the more costly their misclassification becomes). The number of accepted applicants in each period is dependent on the threshold, which in turn depends on the dataset and its misclassification cost. This choice is well motivated by the literature, as detailed in section 4.7.3.

Further, it is assumed that a fixed number of applicants is presented in each period. While in practice a constant stream of credit applications is more realistic, making AL-selections from a stream of applicants requires a threshold for the utility metric of each AL model (uncertainty/diversity). Building such thresholds requires expert domain knowledge and extensive tuning and is out of the scope of this experiment.

4.3. Selection of AL-models

All AL-models in this experiment are deployed as implemented in the public ALiPy package (Tang, et al., 2019), which is used in several other publications (Kumar & Gupta, 2020; Zhan, et al., 2021). The models are grouped into three families as introduced in section 3. Table A-1 in the Appendix summarizes all AL-models, including their tuneable model parameters.

Uncertainty-based models

- *Uncertainty Sampling (UNC)* (Lewis & Gale, 1994) chooses those instances with the most uncertainty in their predicted labels. Variants utilize different uncertainty measures (entropy, least confidence, margin, distance-to-boundary). For each dataset, the best uncertainty measure is selected via parameter optimization.

- *Query-by-Committee (QBC)* (Seung, et al., 1992) deploys multiple learners with differing hypotheses and selects the instance with the highest divergence in predictions between the learners. It improves upon uncertainty sampling, which might be biased toward the learning algorithm supplied. Implemented here is the query-by-bagging approach (Abe & Mamitsuka, 1998).
- *Expected-Error-Reduction (EER)* (Roy & McCallum, 2001) queries those instances that are expected to minimize the generalization error most. The expected loss for each possible label *y* is weighted by the posterior probability obtained from the current classifier.
- *Learning by Active Learning (LAL)* (Konyushkova, et al., 2017) considers properties of both the trained classifier as well as the data to predict the potential error reduction via a random forest regression model. This approach should lead to a model that automatically adapts to class imbalances and other domain-specific factors.

Diversity-based models

- *Random sampling (RND)*, while not a model-based query strategy, is added to the diversity category. For this approach, a portion of rejected instances are randomly accepted ex post. The idea is that a random sample is least biased and may thus contribute to the reduction of classification bias. The possibility is frequently mentioned (Hand & Henley, 1993; Verstraeten & Van den Poel, 2005; Ditrich, 2015), along with the increased cost that such sampling brings. While random classification is often used as a worst-case baseline in many experiments, randomly selecting a sub-sample may be a viable strategy in the Credit Scoring domain. The approach requires no model or training.
- *Core-Set – Greedy KCenter (CORS)* (Sener & Savarese, 2018) greedily selects the sample that minimizes the maximum distance of any point to its cluster-centre, resulting in samples which are representative of the population they were selected from.
- *Graph Density (DENS)* (Ebert, et al., 2012) uses the K-Nearest-Neighbor (KNN) algorithm to build a graph-structure from which highly connected nodes are identified. Data points are ranked by how well they are connected within the graph structure, and the most representative are selected.

Combined models

- *Querying Informative and Representative Examples (QUIRE)* (Huang, et al., 2010) uses the prediction on the labelled data to measure informativeness, and the prediction accuracy based on the unlabelled data to measure representativeness.

- *Weighted Density (DW)* (Settles, 2008) weighs the informativeness of an unlabelled instance by its similarity to other unlabelled instances, hence making the query more robust against outliers.
- *Discriminative and Representative Queries for Batch Mode Active Learning (BMDR)* (Wang & Ye, 2015) generalizes the empirical risk minimization principle (ERM) to Active Learning, where samples in the training data are not i.i.d. It queries the most informative samples while preserving the source distribution as much as possible.
- *Self-Paced Active Learning (SPAL)* (Tang & Huang, 2019) queries a batch of informative and representative examples by minimizing an elaborate objective function, which ensures that easy (less complex) instances are queried first.

4.4. Datasets and Data Processing

The Credit Scoring literature generally suffers from a lack of data. A comprehensible reason is that credit institutes keep their data to themselves, both protecting their clients and not giving away business secrets. The few available datasets are often small and incomplete (Petrides, et al., 2020; Lessmann, et al., 2015). As established before, most Credit Scoring datasets suffer from sample selection bias. Banasik et al. (2003) use data where each credit application was granted (i.e., data free of selection bias) but do not make it available to the scientific community.

Table 4-1: Data summary

Name	Cases	Variables[a]	Default rate	CV-folds[b]	Source
Australian	690	42	.44	10	http://archive.ics.uci.edu/ml/datasets/statlog+(australian+credit+approval)
German	1,000	61	.30	10	https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)
Thomas	1,225	28	.26	10	(Thomas, et al., 2002)
Bene 1	3,123	18	.33	5	(Baesens, et al., 2003)
HMEQ	5,960	20	.20	5	http://www.creditriskanalytics.net/datasets-private.html
Bene 2	7,190	28	.30	5	(Baesens, et al., 2003)
UK	30,000	51	.04	3	(Baesens, et al., 2003)
Lendingclub	43,344	114	.07	3	https://www.kaggle.com/wendykan/lending-club-loan-data
PAKDD	50,000	373	.26	3	https://www.kdnuggets.com/2010/03/f-pakdd-2010-data-mining-competition.html
GMSC	150,000	68	.07	3	https://www.kaggle.com/c/GiveMeSomeCredit

[a] Data is processed; numbers of features differ from original datasets.
[b] CV-folds which are used for parameter tuning. The experiment itself is run in a 5-fold split (5 rounds of each model configuration are run on each dataset).

The experiment is run on a total of 10 different datasets (see Table 4-1). Most have also been used in other recent Credit Scoring literature (Lessmann, et al., 2015; Kozodoi, et al., 2019; Kozodoi, et al., 2022). Data processing is kept to a minimum. Redundant or otherwise useless features (e.g., ID-columns) are removed, as are columns with many missing values (> 0.33), high correlation ($r > 0.95$), or very small variance. For the remaining features, missing values are imputed (means for numerical, modes for categorical variables). All categorical variables are one-hot encoded. No other feature selection is in place. Outliers are not removed from the data, instead, the data is centred and re-scaled to the interquartile range using the initial warm-start sample of each round. This scaling procedure increases robustness against outliers and is generally suggested when regularized logistic regression is applied (Hastie, et al., 2009). Following Lessmann et al. (2015), no treatment of class imbalance is undertaken. Since imbalance is a natural occurrence in most Credit Scoring data, and since AL is concerned with the augmentation of data, it is interesting to see how well AL-models operate under the constraints of class imbalance without interference.

4.5. Classifier Choice

For this experiment, Logistic Regression (LR) is chosen as the base classifier. There are several reasons for this. Firstly, LR is commonly used in the industry, mainly because of its easy deployment and good interpretability of its coefficients, which often is a regulatory requirement (Hand, 2005; Verbeke, et al., 2017; Dumitrescu, et al., 2022).

Secondly, it is well established that, since LR directly optimizes log-loss, it delivers well-calibrated predictions (Kuhn & Johnson, 2013). In addition, there is evidence that LR performs well with small sample sizes compared to more complex models, and that it is less sensitive to class imbalance compared to other classifiers (Crone & Finlay, 2012). This is important for the current experiment, which uses small, imbalanced samples that only grow over iterations. LR is also fast and does not require extensive tuning, given that it has few meta-parameters. While more complex models such as ensembles perform better in their experiment, Lessmann et al. (2015) find base LR to be the second-best individual classifier in their benchmark. Only Artificial Neural Networks score higher in that category.

Lastly, LR is often used in combination with AL-models, as many techniques exploit its posterior class probabilities (Yang & Loog, 2018). Hence, choosing LR as the classifier ensures good compatibility with the AL-models it is paired with.

In this experiment, LR is used with additional parameters that are tuned for best performance. With the tuning scheme described in the following section, regularization type and strength can be selected, effectively deciding whether Ridge or Lasso regression are performed. Yang & Loog (2018) also use regularized LR for their AL experiment.

4.6. Parameter Optimization

It is generally agreed upon that k-fold cross validation (CV) is the ideal way to resample data (Hastie, et al., 2009). It is used to generate robust validation data without wasting any instances, which is particularly important for small datasets. As such, some form of CV is generally used in any supervised machine learning task where the model has tuneable parameters (Kuhn & Johnson, 2013). Attenberg & Provost (2011) state that CV can be difficult to apply to AL, since any CV scheme relies on there being labelled data in the first place which can be split into folds. Since there is data available in this experiment, this is not considered an issue. There is precedent of authors not applying any dataset-specific pre-processing or parameter tuning in AL benchmarking experiments, e.g., (Zhan, et al., 2021). Nevertheless, the experiment at hand does apply several steps of parameter-optimization of both the LR-classifier as well as the AL-models.

An ideal parameter tuning scheme would optimize model parameters using nested cross-validation. The main argument for nested cross-validation despite its complexity and computational demand is that it produces unbiased parameters and performance estimates (Stone, 1974; Wainer & Cawley, 2021). In contrast to this established practice, Wainer & Cawley (2021) find that for many practical applications even a single (‘flat’) k-fold CV scheme is sufficient to perform parameter optimization and performance estimation. They state that the bias from not having truly unseen test data will rarely lead to significantly different results compared to nested cross-validation.

Motivated by this finding, a different, non-nested approach is applied in this experiment. As laid out in section 4.1, contrary to most machine learning applications, this study employs both classification and AL-models in conjunction. Instead of modelling a classifier–data relation, it models a classifier–data–AL relation, which also makes parameter tuning more complex, not only by drastically increasing the total number of possible parameters. Another level of complexity is introduced by having not a single iteration of training and prediction within each fold (round), but instead multiple iterations of instance selection, model building and prediction.

Consequently, parameter optimization and performance estimation are handled in two separate k-fold cross-validation loops. In the first loop, the classifier and AL-models are tuned sequentially (see Algorithm B), which further reduces complexity. In the second loop, the AL-algorithm is run, and performance is estimated (see Algorithm A)[3]. Also, since this benchmarking experiment focusses on AL-models and not on the classifier, it well serves the performance comparison if variables outside the AL-models are kept constant, the classifier being one of them. To avoid data leakage, it is ensured that the same data is used between both tuning loops.

Before tuning begins, the data is split into folds. For datasets with fewer than 7,500 instances, the whole dataset is used. For larger datasets, the size is capped at that threshold and a stratified sample of the maximum size is drawn. Through this, the size of the training sample is kept close to the conditions in the subsequent benchmarking runs. Before tuning begins, the data is scaled via the robust scaler. In the first step of the tuning procedure, the LR classifier is optimized on this data using Grid Search on a 10-fold CV scheme. Tuneable parameters and choices are *penalty* {l1, l2}; *regularization strength* {0.001, 0.01, 0.1, 1, 10, 100}; *maximum iterations* {50, 100, 250, 500}; *stopping criteria tolerance* {0.001, 0.0001, 0.00001}.

Then, separately, the AL-models are tuned, using the parameter grids specified in Table A-1, and the previously tuned classifier. For this second step, 50% of the available data are provided as labelled instances. From the data that remains unlabelled, the AL-model then selects 50% to be added to the labelled data. This results in a labelled training sample with 50% randomly selected instances and 25% AL-selected instances.

After the selection process, the model is fit on the labelled data and a prediction on the test-fold is made. From the predicted class probabilities, the Area Under the Receiver Operating Curve (AUC) (see section 4.7.1) score is computed. The optimal parameters of both the classifier and the AL-models are determined by the highest AUC score. While not without faults (Hand, 2009) the AUC is chosen for its prevalence in binary classification in general (Kuhn & Johnson, 2013) and the Credit Scoring literature in particular (Verbeke, et al., 2017). A second important reason is the fact that the AUC is threshold-independent, i.e., it does not require a tuned threshold to be applied to predicted class probabilities (see section 4.7 for further elaboration on thresholds). As detailed in the next section, the threshold underlies some assumptions regarding misclassification cost.

[3] Algorithms are numbered by their appearance in the paper, not by the order in which they are run.

These assumptions can fit one dataset well but might be off on a different dataset. Hence, using a threshold-dependent performance metric for optimization would exacerbate any deficiencies the threshold might have. Consequently, optimizing parameters based on AUC – independently of any threshold – seems sensible. Also, the optimization of parameters and threshold can then be performed separately, which reduces complexity.

Algorithm B: K-Fold Cross Validation for Parameter Optimization

models: pool of AL and baseline models	$\Theta_{LR/AL}$: parameter grid for LR or AL models
data: full dataset	$\theta_{LR/AL}$: (optimal) parameters for LR or AL models
K: number of cross-validation folds	LR: binary LR classifier
D: available data	LR_θ: classifier with parameters θ
D_{K-1}: training data (K-1 folds)	LR_t: tuned LR classifier (optimal parameters)
D_k: test data (fold k)	AL: active learner
U: unlabelled training data	a: sample queried by AL (50% of U)
L: labelled training data	

input: data, Θ, LR, models, K

if data > 7,500 instances: draw stratified random sample **D** ⊂ **data**
else: D = data
pass **D** to algorithms 2a and 2b

2a: classifier tuning

```
for each k in K = 10:
    for each θ in ΘLR:
        split Dk-1 50/50: L∪ U = Dk-1
        fit robust scaler on L
        transform (re-scale) D

        train LRθ on L
        predict on test data Dk (kth fold)
        compute performance score
        return score, θ
    end for

    save θLR with best average score
end for
```

2b: AL-model tuning

```
for AL in models:
    for each k in K = 10:
        for each θ in ΘAL:
            split Dk-1 50/50: L ∪ U = Dk-1
            fit robust scaler on L
            transform (re-scale) D
            from U select a via AL, add to L = U ∪ a
            train LR with parameters θLR on L
            predict on test data Dk (kth fold)
            compute performance score
            return score, θ
        end for
    end for
    save θAL with best average score
end for
```

[a] Splits are random, shuffled. All parameters are tuned on the same data to avoid leakage.
[b] Any train, test, labelled and unlabelled data are managed via their indices in the full data. Any operation on D (such as scaling) can therefore be applied to all data, even after splitting.

Some experimental parameters that are not directly passed as arguments to the classifier or AL-model, i.e., AL-ratio or sample weights (see section 4.1), are not tuned. Instead, the experiment loop is run for each of these parameters, and results are collected and

compared for all possible combinations. As noted in Table A-1, there are several exemptions in model tuning: the EER model has no tuneable parameters. The QUIRE model does allow tuning but is generally very slow to run and was found to be unstable for certain parameter combinations. The authors themselves use default parameters for the 'rbf' kernel function, with no other parameter tuning applied (Huang, et al., 2010). For computational reasons and because of the precedent set by the authors, QUIRE is not tuned in this experiment. Lastly, the BMDR and SPAL models, both of which use quadratic computing, experience convergence issues on the Thomas dataset. Hence, they could not be tuned on this data.

4.7. Measuring Performance: Threshold and Metrics

Measuring performance of a scorecard is a very important aspect of the modelling process in Credit Scoring. Misclassification can be very costly, and cost can differ widely depending on which class is wrongly predicted. This makes the selection of suitable performance metrics very important, both in research as well as in practical applications. This paper considers several metrics, covering both empirical as well as economic approaches.

Table 4-2: Confusion Matrix

	Predicted class	
Actual class	0:=GOOD	1:=BAD
0:=GOOD	True Negative (**TN**)	False Positive (**FP**)
1:=BAD	False Negative (**FN**)	True Positive (**TN**)

Most metrics are based on the confusion matrix of binary classification, which knows four outcomes: true negative (TN), false positive (FP, type I error), false negative (FN, type II error), true positive (TP)[4]. As in many classification problems, Credit Scoring assigns instances to classes based on whether their predicted class probabilities lie above or below a certain threshold

[4] There is potential for misunderstanding when referring to misclassification error in credit scoring using the type I, type II nomenclature. This comes from the fact that the positive class of interest 1:=BAD is the undesirable outcome, i.e., a loan default. If one assumes that the null hypotheses in a credit scoring context is that an instance is good (the majority outcome in most cases), the rejection of said hypotheses means predicting 1 (loan is BAD). A false rejection of the hypotheses means a loan is wrongfully declined (false positive; type I error), while a false acceptance of the hypotheses means a bad loan is accepted (false negative; type II error).

(Hand, 2005; Verbeke, et al., 2017), i.e., the threshold determines above which probability score between 0 and 1 an instance is sorted into the BAD class. In most cases, this threshold requires tuning, as setting the threshold arbitrarily at 0.5 may lead to sub-optimal results, especially when dealing with class imbalance and class-dependent misclassification cost (Kuhn & Johnson, 2013).

The metrics described below are separated into threshold-independent and threshold-dependent metrics. Cost metrics for their importance are detailed in a separate section, albeit belonging to the group of threshold-dependent metrics in a strict sense.

4.7.1. Threshold-Independent Metrics

Metrics in this category are directly based on predicted class probabilities. Their benefit is that they are less influenced by other modelling decisions, such as the criteria on which the threshold is chosen. As is the case in this experiment, the threshold can be subject to many assumptions. In that way, the threshold-independent metrics can be seen as more neutral.

The *Area Under the Receiver Operating Curve* (AUC) assess the discriminatory ability of the scorecard. It is a popular and widespread performance metric for binary classification (Hand, 2009; Lessmann, et al., 2015; Verbeke, et al., 2017). It plots the true positive rate against the false positive rate across all possible thresholds and can be used to determine the (AUC-)optimal threshold. A benefit of the AUC is that it is insensitive to class imbalance (Fawcett, 2006). In case of Credit Scoring, it represents the probability that a randomly chosen bad risk gets a higher prediction score than a randomly chosen good risk (Hanley & McNeil, 1982; Verstraeten & Van den Poel, 2005). One drawback of the AUC is that is assumes equal misclassification cost for false positives and false negatives. In many cases, this assumption is violated (Verbeke, et al., 2017).

A second drawback of the AUC is that, while it assumes equal misclassification cost between classes, the distributions of these costs often differ between classifiers. Hand (2009) argues that this is nonsensical, as misclassification cost is a property of the data and the problem domain, not of the classifier. As such, the AUC is not ideal for benchmarking performance, because models are compared on different scales. He instead proposes the *H-measure* (HM), which considers the misclassification cost of different classes in its computation and fixes the cost distribution in an invariable manner across classifiers. It thus provides better comparability than the AUC. This makes it a well-suited performance metric for this experiment, where classes are imbalanced and misclassification cost is class dependent. The H-measure has since seen wide use in the Credit Scoring literature (Pavlidis, et al., 2012; Lessmann, et al., 2015; Xia, et al., 2017).

The *Brier Score* (BS) is a metric that reflects model calibration by measuring the divergence of probability estimates from the observed labels. As such, it is a cost function similar to the mean square error where smaller values relate to a better calibrated classifier. While not being entirely robust against class imbalance (Wallace & Dahabreh, 2014), the Brier score has been a popular performance metric in Credit Scoring experiments of recent years (Lessmann, et al., 2015; Ala'raj & Maysam, 2016; Xia, et al., 2017; Dumitrescu, et al., 2022).

4.7.2. Threshold-Dependent Metrics

Metrics in this category require discretization of the continuous probability scores estimated by the classifier. Hence, these metrics can be computed only after a threshold has been applied. They measure model performance with respect to a single reference point (i.e., the threshold).

The *Percentage Correctly Classified* (PCC), or alternatively, Accuracy, is a well-known and wide-spread performance metric in classification. It is intuitive and easily implemented, albeit not without its faults, the main criticism being that it is highly influenced by class imbalance. For a very imbalanced dataset, even a naïve classifier that always guesses the majority class would receive a high accuracy score. Lessman et al. (2015) argue that this criticism does not consider the impact of the threshold, which, if properly calibrated, will mitigate the class imbalance problem to some extent.

Hand (2005) suggests the ratio of accepted instances (i.e., instances scoring below the threshold) who turn out to be bad risks as a basic but appropriate performance metric for Credit Scoring. This metric, here referred to as the *False Negative Rate* (FNR), also depends on the threshold and incorporates the fact that misclassifying bad risks as good (predicting 0 when the true label is 1, i.e., the type II error) is the more costly error in Credit Scoring. Through this, it is a metric that can deliver an approximation of economic impact to the extent that the threshold incorporates such information.

4.7.3. Cost Model

Since this experiment also investigates the economic consequences of AL in Credit Scoring, it puts special emphasis on the cost model. While the cost metrics included in the empirical analysis are all threshold-dependent, their importance makes it sensible to dedicate a separate section towards their elaboration.

Minimizing misclassification error and minimizing cost can be two very different objectives (Viaene & Dedene, 2005; Elkan, 2001). For a precise measurement of misclassification cost,

a cost matrix is required. These are not easily derived from the data and oftentimes encompass very complex economic reasoning (Verbeke, et al., 2017). In practical applications, the cost of default would be highly dependent on the individual terms of a loan. Bahnsen et al. (2014) and Petrides et al. (2020) study cost-sensitive Credit Scoring by using an instance-dependent cost model. The latter suggest using class-dependent cost in case a model cannot handle instance-dependent cost, or when such information is not available.

Few datasets are published with a cost matrix, most come without. An exception is the German Credit dataset also used in this experiment. It comes with the suggestion to set the misclassification cost of a false positive $C_{(FP)} = 1$ and that of a false negative $C_{(FN)} = 5$, while any true positives or true negatives generate zero cost. This approach of using a fixed FP:FN cost ratio of 1:5 is adopted by several authors (West, 2000; Abdou & Pointon, 2011). Lessmann et al. (2015) keep FP cost fixed to 1 while varying the magnitude of FN cost. The argument for including FP cost in the cost matrix stems from the idea of cost of missed opportunity – when rejecting a good risk, the lender forgoes potential future profits (Baesens, et al., 2003; Nayak & Turvey, 1997).

There is some disagreement on whether focussing on the FP:FN cost ratio is the correct way of portraying misclassification cost. Elkan (2001) argues that if an instance is predicted to be bad, it is always rejected. Hence the actual cashflow generated from this instance is always the same, regardless of true outcome. According to his reasoning it is not sensible to attribute different cost to FPs and TPs. While Elkan's approach might apply for isolated single-round Credit Scoring applications, the case is different for this experiment with its multiple rounds of AL-selections. In such a setup, where any instance can be selected through AL despite having a prediction score above the threshold, it is important to differentiate between the cost of false positives and true positives in the cost matrix. FP cost also serve an important purpose when tuning the threshold, as they make selecting no instance at all (i.e., a very low threshold) costly for the lender. In that sense, FP cost moderate the threshold tuning process and help finding a balance between accepting too few and too many applicants.

Verbraken et al. (2014) introduce a different measure, the Expected Maximum Profit (EMP), which computes cost and derives a recommended threshold from specific cost-benefit distributions. Cost is computed against a baseline where all instances are accepted. Hence, in the EMP, the cost of false-negative classifications is of no concern since the baseline already incurs all said cost. This marks a departure from prior examples where FN misclassifications are considered the costliest.

The EMP is used in several recent studies on Credit Scoring (Devos, et al., 2018; Kozodoi, et al., 2019; Kozodoi, et al., 2022), but was ultimately not chosen for this experiment. The reason lies in the missing flexibility of the EMP. The EMP cost measure is only true for the specific threshold provided by the model. While in this experiment the threshold does determine the total number of instances to be selected, the selections are not performed strictly according to the threshold (see Figure 4-1). Hence, it would be difficult if not impossible to compute exact EMP values for the AL-moderated labelling process. As such, the EMP could only be used as a performance metric for the test-prediction, not as a measure of labelling cost.

Comparing the methods for inferring misclassification cost suggested in this section, a modified variant of the fixed FP:FN cost ratio is applied. Instead of arbitrarily setting this ratio to 1:5, misclassification cost is set inversely proportional to the class distribution in the data. This means that the rarer an outcome, the more costly it is to misclassify it, which is true for many imbalanced classification tasks and applies to Credit Scoring, too. This method of determining misclassification cost is popular among scholars (Japkowicz & Stephen, 2002; Verbeke, et al., 2017; Sheng & Ling, 2006). When no problem-specific information on misclassification cost is available, Hand & Anagnostopoulos (2014) suggest applying the same principle to determine the severity-ratio used to compute Betas for the H-measure.

Table 4-3: Cost Matrix

	Predicted class	
Actual class	GOOD	BAD
GOOD	$C_{TN} = 0$	$C_{FP} = 1$
BAD	$C_{FN} = (1-\bar{y}) / \bar{y}$	$C_{TP} = 0$

$\bar{y}$ = mean value of y ⟶ prior probability of a risk being BAD

An important trait of the experiment is that it knows two dimensions of cost: *internal* and *external* cost. Differentiating between these dimensions is useful because it underlines how different sampling strategies can affect the cost incurred during sample building. As described in detail in section 4, there are several cycles of applicant selection through a mix of score-ranking and AL. Each of these selections can potentially entail misclassification cost: Selecting an instance is equivalent to granting a loan, which bears the chance of default. This type of cost is here referred

to as *internal cost* or labelling cost. It can also be described as training cost – the misclassification errors that are made during the AL-supported compilation of the training data.

The second type, *external cost* occurs when a prediction is made on unseen data: In each period, the model trained on all available labelled data is used to make a prediction on the test data of the respective fold. This is standard procedure to see how well a model generalizes on unseen data. Here, the same principal as above applies, each predicted probability that lies below the classification threshold is considered an accepted loan, each score above the threshold is a reject. When comparing these predictions to the ground truth via the confusion matrix, the cost matrix can be applied, and the misclassification cost be computed.

For this, the concept of *Cost Per Loan* (CPL) is introduced as a metric to compare cost for different models across multiple datasets. The metric is computed by multiplying each element of the confusion matrix by the element in cost matrix at the same position, i.e., the number of cases times the cost. To receive the cost per loan, the sum is divided by the total number of instances *n*:

$$\text{cpl} = \frac{\text{FN} \times \text{C}_{\text{FN}} + \text{FP} \times \text{C}_{\text{FP}}}{\text{n}}$$

For the overall performance benchmark, it is necessary to build the average cost per loan across folds *k* and periods *p*:

$$\overline{\text{CPL}} = \frac{1}{\text{k}} \sum_{\text{k=1}}^{\text{k}} \frac{1}{\text{p}} \sum_{\text{p=1}}^{\text{p}} \text{cpl}_{\text{p,k}}$$

Cost is expressed as cost per loan instead of absolute cost because the experiment design does not allow control of the total number of instances kept/queried across all models, resulting in vastly different scales. Hence, displaying cost per loan ensures comparability between models across datasets. Following the differentiation between internal labelling cost and external prediction cost, the CPL is computed twice, resulting in the *Average External Cost per Loan* (ECPL) and the internal equivalent (ICPL). Both metrics are represented in the results (see Table 5-1). One must note that the ICPL is independent of the threshold and the performance of the classification model. Instead, it only depends on the AL-model and its selections. This also means that it is not directly affected by the respective sample-weight configuration. Having measured the two cost components, one can examine whether incurring higher labelling cost through AL-selection is offset by a reduction of external cost. In other words: does adding costly AL-selected instances to the training data sufficiently improve any future predictions.

4.8. Threshold Optimization

There are examples in the literature where the classification threshold is optimized within the nested cross-validation scheme (Zhong, et al., 2020). Sheng & Ling use this approach for their empirical thresholding method, where the optimal threshold is the one that minimizes cost most effectively across multiple cross-validation folds (2006). This study applies the same optimization principle but embeds it in a separate (non-nested) CV-loop described in Algorithm C. This means thresholds are not optimized for each individual combination of AL-model and performance measure. This may be sub-optimal for achieving maximum performance in each metric but is grounded in the complexity of the experiment design (see section 4.6), which made the combined implementation prohibitively difficult.

Algorithm C: K-Fold Cross Validation for Threshold Optimization

data: full dataset	T: list of thresholds
K: number of cross-validation folds	τ: current threshold
D: available data	$\hat{\tau}$: optimal threshold
D_{K-1}: training data (K-1 folds)	P: predicted probabilities
D_k: test data (fold k)	P_τ: predicted classes (τ applied to P)
LR: tuned LR classifier	M_k^τ: misclassification cost at threshold τ (in k^{th} fold)
C: cost matrix	$\overline{M}$: average misclassification cost

input: data, C, T, LR

1[a]: split data into K folds
2: **for each k in K=10:**
3: fit robust scaler on $\mathbf{D_{K-1}}$
4[b]: transform (re-scale) **D**
5: train **LR** on $\mathbf{D_{K-1}}$
6: predict on test data $\mathbf{D_k}$ (k^{th} fold), store **P**
7: **for each τ in T:**
8: convert **P** to classes [0,1] via $\boldsymbol{\tau}$ to receive $\mathbf{P_\tau}$
9: via **C**, $\mathbf{P_\tau}$ compute misclassification cost $\mathbf{M_k^\tau}$
10: return $\mathbf{M_k^\tau}$
11: **end for**
12: **end for**
13: compute average cost $\overline{\mathbf{M_\tau}}$ for each threshold across folds
14: select optimal threshold that minimizes cost: $\hat{\boldsymbol{\tau}} = \underset{\tau \in T}{\mathbf{argmin}}\, \overline{\mathbf{M_\tau}}$

[a] Splits are random, shuffled.
[b] Any train, test, labelled and unlabelled data are managed via their indices in the full data. Any operation on D (such as scaling) can therefore be applied to all data, even after splitting.

Further, the choice of threshold has large implications, as it affects the total number of instances that is selected in each round (see Figure 4-1). An advantage of tuning the threshold once for each dataset and not specifically for each data–classifier–AL combination is that the acceptance rate stays roughly the same across specifications (sample weight and AL-ratio setups), which is beneficial for the model comparison in this experiment. Lastly, in real-world applications that truly optimize cost, a cost-optimal threshold would likely be used to make loan choices, even if further performance metrics other than cost are monitored.

As in Algorithm B, the classification threshold is tuned via 10-fold CV for each combination of dataset and AL-model. In each iteration, one CV-fold is used as test data and the remaining folds are used for training. First, the robust scaler is fitted, and all data is transformed. Then, the classifier is fitted on the data using the tuned parameters as collected by Algorithm B. The probabilities from a prediction on the test fold are then fed into a loop which applies each candidate threshold $\tau \in \{0, 0.001, \ldots, 0.999, 1\}$ and computes misclassification cost based on the dataset-specific cost matrix (see Table 4-3). For each threshold, the total cost is gathered and then averaged across iterations of the tuning loop. Using the average cost of each τ, the optimal threshold with minimal average misclassification cost is selected.

5. Empirical Results

The testing procedure and presentation of the main results is structured in accordance with Lessmann et al. (2015). For the final comparison, the best-performing version of each AL-model is determined ex post based on the lowest average rank across all seven performance metrics and benchmarked against one another. The best version of each model is chosen from 10 variants: 2 weight-options $w \in$ [wF, wT] × 5 AL-ratios $r \in [0.1, \ldots, 0.5]$. With 11 AL-models, this equates to 110 combinations. The experiment is run for 5 rounds (CV folds) on each combination of dataset (10), AL-model (11), and model parameters (10). Not counting the 10 periods of applicant-selection within each round, this results in a total of 5500 runs of the experiment, plus 50 runs of the baseline SCORE model (see below).

The results from the experiment are recorded in Table 5-1. For each metric, the table shows the average rank of each model and the Finner-adjusted p-values of the post-hoc test (García, et al., 2012). The last row shows the Friedman test-statistic and the respective p-values (Demsar, 2006) with Iman-Davenport correction to avoid overly conservative values (García, et al., 2010).

It must be noted that for the AUC, FNR and ECPL metrics, the Friedmann test statistics are not significant on a 5% level. In case of the AUC, the Finner test is still able to locate several models that rank significantly worse compared to the best-ranking QBC model. The last two columns show the average ranks across all metrics in the table and a high score. These are not statistically relevant but help summarize the overall results.

The benchmark comparison serves two main purposes. Most importantly, it affirms the assumption that AL can improve performance in Credit Scoring applications. For this, the models are compared to the baseline scenario of only selecting applicants via their prediction score, i.e., only keeping instances which score below the classification threshold. This SCORE model would be considered the standard approach in the credit industry. The results show evidence that training classifiers using AL-curated data improves performance compared to the baseline across several evaluation metrics.

The best model in the benchmark is the Query-By-Committee (QBC) model with sample weights enabled (*wT*) and an AL-ratio of 0.4 (*r04*). It ranks best in three out of seven metrics. (AUC, HM, BS), in all of which it significantly outranks the baseline SCORE model. Further, it ranks second best in the PCC and ECPL metrics, the distance to the best models being insignificant in both cases.

The good performance of the QBC model is interesting, as it is a rather conventional AL-model of the *uncertainty* type that is less complex than recent approaches such as, e.g., BMDR or SPAL. It seems that the construction of the model as an ensemble of bagging algorithms is efficient in mitigating an important issue that other models in its category suffer from: Usually, a model that selects instances based on how uncertain the classifier is about their true class is inherently biased toward said classifier. The AL-model's selections will change whenever the classifier changes. This can lead to behaviour similar to the cold-start problem (Attenberg & Provost, 2011), i.e., a badly fit classifier leads to bad AL-selections. QBC overcomes this by deploying an ensemble of models, each with a different hypothesis, and having them cast a vote on which instance to select. The point where the ensemble disagrees most is then chosen. This regulates how decisions are formed and reduces bias toward the (LR) learner, which results in good overall performance that ranks favourably compared to other models. QBC being the best-performing model in the benchmark confirms results by Zhan et al. (2021), who find simpler, *uncertainty*-based models to often outperform more complex models.

Table 5-1: Empirical Comparison of the Best Models Across Datasets and Metrics

Query model	AUC	HM	BS	PCC	FNR	ECPL	ICPL	Avg. Rank	High Score
SCORE	7.8 (.038)	7.6 (.042)	10.1 (.000)	8.2 (.016)	5.2 (.710)	6.5 (.325)	**1.4** (/)	6.69	9.0
QBC (wT, r04)	**3.6** (/)	**3.7** (/)	**3.0** (/)	4.4 (.535)	6.6 (.413)	4.7 (.852)	9.7 (.000)	5.10	1.0
RND (wT, r04)	5.2 (.321)	4.4 (.664)	4.1 (.528)	4.9 (.380)	7.7 (.186)	**4.4** (/)	9.3 (.000)	5.71	2.0
EER (wT, r04)	6.2 (.144)	4.6 (.612)	3.1 (.951)	6.0 (.144)	6.1 (.450)	6.5 (.325)	11.4 (.000)	6.27	3.0
DENS (wT, r04)	6.5 (.112)	5.1 (.448)	5.1 (.230)	5.5 (.230)	8.0 (.178)	6.2 (.325)	7.9 (.000)	6.33	4.0
QUIRE (wF, r01)	5.7 (.230)	6.9 (*.085*)	8.3 (.003)	7.8 (.023)	5.3 (.699)	7.0 (.300)	4.1 (.103)	6.44	5.0
DW (wF, r02)	6.6 (.112)	6.3 (.144)	5.9 (.111)	6.4 (*.097*)	6.8 (.406)	7.3 (.300)	6.5 (.003)	6.54	6.0
BMDR (wT, r04)	6.8 (.101)	6.5 (.127)	5.3 (.205)	**3.4** (/)	8.8 (*.097*)	6.4 (.325)	9.0 (.000)	6.60	7.0
SPAL (wF, r01)	5.3 (.316)	7.3 (*.055*)	8.7 (.001)	7.6 (.025)	6.5 (.413)	7.0 (.300)	4.2 (.100)	6.66	8.0
UNC (wF, r01)	8.2 (.038)	8.5 (.026)	7.0 (.024)	7.5 (.025)	6.4 (.413)	6.1 (.325)	5.6 (.014)	7.04	10.0
LAL (wF, r01)	7.8 (.038)	8.6 (.026)	7.9 (.005)	8.8 (.009)	**4.6** (/)	8.5 (.115)	4.9 (.041)	7.30	11.0
CORS (wF, r01)	8.3 (.038)	8.5 (.026)	9.5 (.000)	7.5 (.025)	6.0 (.450)	7.4 (.300)	4.0 (.107)	7.31	12.0
Friedman $\chi 2$	1.626 .103	2.624 .006	7.624 .000	2.483 .009	1.153 .330	0.966 .482	18.966 .000		

Model parameters in paratheses indicate sample weight (w) and AL-ratio (r) specification. Bold face indicates the best classifier (lowest average rank) per performance measure. Values in brackets give the Finner-adjusted p-value (García, et al., 2010) corresponding to a pairwise comparison of this row's classifier to the best classifier (per performance measure). An underscore indicates that p-values are significant at the 5-percent level. Italic p-values are significant at a 10-percent level. The Friedman test is employed to verify that at least two classifiers perform significantly different (Demsar, 2006) in the respective metric. The last row shows the corresponding $\chi 2$ and p-values. Following García et al., (2010), the Iman Davenport correction is applied to the $\chi 2$ statistic before p-values are computed.

The second-best model according to the high score is the random model (RND). Usually, having a random choice model ranking high is a bad sign, but in a Credit Scoring application it is a viable choice to mix random selections into the training data (Ditrich, 2015). After all, they are non-biased and hence representative of the debtor population. It is arguably the fairest of all approaches. If one considers that the fit of an AL-model might be highly dependent on the dataset and classifier, it makes even more sense that the random option ranks high. After all, its choices are completely independent of any fit considerations. This is similar to findings by Yang & Loog (2018), who achieve good results with random sampling and state that no single AL-model would consistently outperform the random model in their experiments.

This is supported by the results, in which the RND model is never significantly outperformed by any AL-model, and even achieves the highest rank (albeit insignificantly) in the ECPL metric. The hypotheses that a model trained on an AL-curated pool of training data will outperform a model trained on data with an equally sized random sample added (Cohn, et al., 1994) is not supported by these results. In contrast, the baseline SCORE model achieves an average rank of 6.7 and only scores highest in one metric, the labelling cost expressed as internal cost per loan (ICPL). The good performance in that particular metric is no surprise – the score model strictly selects instances where it predicts a low chance of default, whereas AL-models apply different criteria in their selection process with not regard to cost. This leads to potentially costly selections, hence the dominant performance of the SCORE model. On the other hand, the mediocre performance in all other metrics, being significantly outranked in four out of seven metrics, is an interesting finding. It verifies that strictly selecting instances by minimizing the predicted probability of default inhibits performance.

Table 5-1 points towards some important secondary results, the most relevant of which are examined in the next sections. Below, the effects of AL-ratio, sample weights, and model type are drawn and analysed. The analysis does not focus on the best models as selected for the comparison table, but instead draws general conclusions on how different modelling parameters affect the overall performance of models. It is to note that the analysis of such results and their compact, concise graphical representation require a high degree of aggregation across datasets and performance metrics, which blurs any local effects (early vs. late periods, dataset specific results). Also, because of the different scales of the performance measures, most graphics will show relative

performance via aggregated rankings instead of raw scores[5]. The analysis of the best models' performance across generations (periods) is deferred to the appendix, where the development of rankings (APPENDIX B: Average Ranks of Best Models Over Periods) and raw scores (APPENDIX C: Raw Scores Over Periods in Selected Metrics) in are drawn and explained.

5.1. Effects of different AL-ratios

Because there is little information in the literature about how to implement AL in the Credit Scoring domain, it is interesting to see how different model specifications affect the results. One parameter of interest is the AL-ratio. The four highest ranking AL-models all use increased sample weights for AL-selected instances, and an AL-ratio of 0.4. Most lower scoring models use uniform sample weights and select fewer instances via AL ($\leq$ 20%). An exception to this rule is the BMDR model, which selects a high share of accepted instances and applies moderate weights, but only ranks seventh overall. Considering its low performance in the ICPL metric due to the high AL-ratio, and the mediocre performance in other metrics, BMDR can be considered the least efficient model in this comparison. Five models use the lowest possible AL-ratio, and only one (DW) selects 20% of instances via AL. This polarizing tendency to either have high (0.4) or very low (0.1) AL-ratios, with only one exception, is interesting and to some degree counterintuitive. A likely reason is the presence of the ICPL among the selection criteria when choosing the best models. The ICPL penalizes high AL-ratios because AL-selections are costly. Whenever an AL-model only moderately increases performance through its selections, the gains (i.e., improved rankings) in classification performance will not offset the increased cost at higher AL-ratios. In that sense, the ICPL metrics provides an indirect threshold, making the cost of high AL-ratios only worthwhile if ranks in the other metrics improve drastically. Hence, for low-performing models, the lowest AL-ratio is the preferable option.

The fact that the best four models all select at high ratios demonstrates that an AL-augmented instance selection process can indeed improve performance and generalizability. If AL-selections were generally detrimental to classification performance, one would expect the best scoring configuration of each model to be one with a very low AL-ratio.

[5] The rankings are built using all 111 possible model specifications (110 AL model versions plus 1 SCORE baseline). For some graphics, the rankings are re-scaled to a range of 1-12 for a more intuitive understanding.

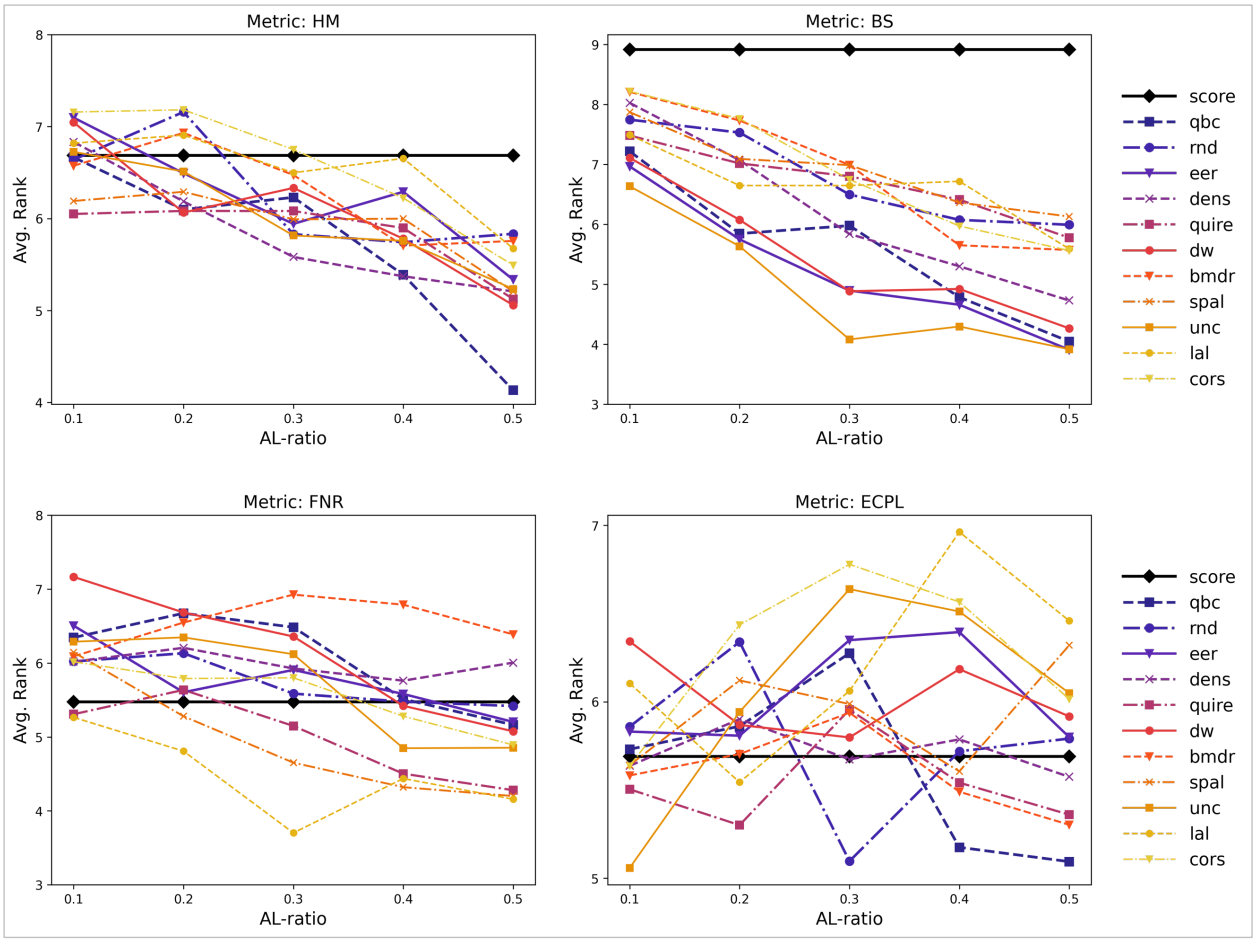

Figure 5-1: Average Relative Ranks Across AL-Ratios (Weights Off)

Further support for this can be found in Figure 5-1, which shows how changes to the AL-ratio influence rankings. Four key performance metrics are selected. The HM as an improved alternative to the AUC that considers misclassification cost, the BS as a measure to express model calibration, the FNR as a simple metric to approximate economic impact, and the ECPL as the metric introduced in this paper, which is used to tune the threshold. For each metric, ranks are first averaged per dataset. The ranks are then aggregated and averaged across datasets. Naturally, low ranks are more desirable. To isolate the effects of the AL-ratio, only result data from the unweighted (*wF*) configuration is used. This avoids interaction effects between the ratio and the weight factor, which would cloud the analysis. The baseline SCORE model is not affected by changes to the AL-ratio and hence stays constant in each plot. Because it shows only 50% of possible model configurations, the plot is not suited to identify the best model versions in a particular metric. Instead, it illustrates the general effects of setting different AL-ratios.

For the HM, BS, and FNR plots, the curves of most AL models show a downward slope indicating that the models improve in the ranking when more instances are selected via AL. The SCORE model shows bad performance throughout in the BS. Even at low AL-ratios, the classifier seems

to improve in general calibration when a (small) share of instances is selected via AL. The QBC model shows the steepest slope in the HM category and generally ranks favourably in the BS category, regardless of ratio. In the FNR metric, QBC is only able to outrank the SCORE model slightly with very high AL-ratios, being outperformed by other AL-models at many points. For the HM, BS, FNR metrics, increasing the ratio by which instances are selected via AL tends to have a positive impact across AL-models– at least in the unweighted scenario viewed here. While all models seem to benefit from higher AL-ratios, having the ICPL as a selection criterion for the best models likely acts as a moderating counterweight.

For the ECPL metric, the plot shows a less clear picture. The QBC model visibly separates itself from the rest at higher ratios ($r > 0.4$). The UNC model starts off with very low ranks, which get worse when AL-ratios increase, only to then converge toward the mid-field when the ratio reaches 50% of selected instances. The curves are rather volatile, an exception being the relatively constant DENS model, and the range of the y-axis of the plot indicates that the rankings are generally very close together, resulting in many shifts in ranks. This points towards either a very similar performance of all models, or large differences in how models perform between datasets. A factor for this could be that the dynamic cost matrix fits some datasets better than others, possibly resulting in either well or badly tuned thresholds, depending on dataset. A badly fitting threshold would of course influence predictive performance of all models.

5.2. Effects of Sample Weights

The impact of sample weights is rather nuanced and must be carefully interpreted. In contrast to the effects of the AL-ratios analysed above, the effects of sample-weights cannot be examined in isolation. One must recall that sample weights are set inversely proportional to the AL-ratio, hence there will always be interaction effects between the weights and the respective AL-ratio. The weights are very high whenever a low share of instances is selected via AL.

Since the top four models all apply sample weights, one can conclude that these models achieve a higher average performance with weights enabled. Importantly, since the AL-ratio of these models is rather high at 0.4, the weights for AL-selected instances are only moderately increased. The effects are exemplified in Figure 5-2, which shows how the average ranks of the best models change when weights are either enabled or disabled. For the graphic, ranks are built across all datasets and metrics.

When sample weights are disabled, the average ranks are rather close between models. This seems intuitive: for AL-models where selections are not very useful, or even detrimental, the optimal model version would be one with a low AL-ratio. For such models, most instances are selected via the prediction score. The population of accepted instances, i.e., the training data, would then be similar to the SCORE baseline, resulting in similar performance. Naturally, the baseline model itself is not affected by changes to the sample weights, since only those instances not selected via their prediction score are weighted differently. The SCORE model does not supply any such instances.

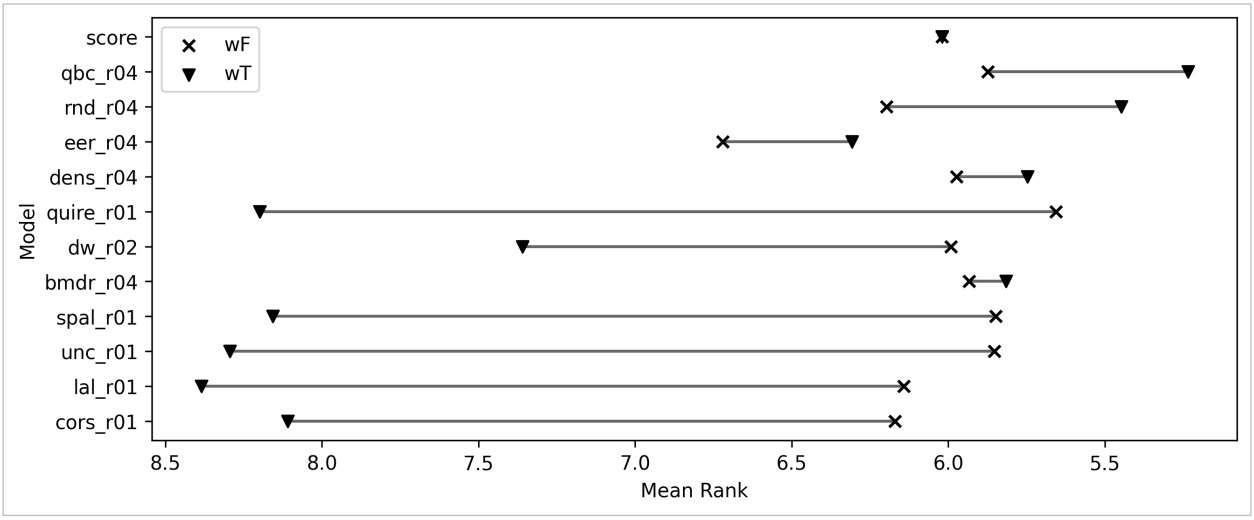

Figure 5-2: Average Ranks per Weight-Option (Best Models)

When weights are enabled, the rankings diverge drastically. While the top-ranking models of the overall benchmark respond positively with a moderate increase with weights enabled, most lower scoring models show drastic losses. On closer inspection, a clear pattern emerges: the worst-affected models all have low AL-ratios. An intuitive interpretation would be that the instances queried by these AL-models do not improve overall classification performance much, e.g., because the models query outliers or otherwise unusual instances at a high rate. Hence the best version of these models is one with a small share of AL-selections in the first place. By enabling weights, the non-helpful selections are further magnified when training the classifier, potentially biasing its predictions in a detrimental fashion.

However, Figure 5-1 shows that all AL-models improve the classifier's performance through their selections, albeit to a varying extent. So, it seems less likely that the weight-induced decline seen in Figure 5-2 merely stems from increasing the weights of ill-fitting instance samples. At this point, due to the interaction effects of weights and sampling ratio, Figure 5-2 makes it difficult to

judge whether the strong negative effects of sample weights at low ratios are caused by the bad quality of the models' selections, or the magnitude of the weights that are applied to them.

Figure 5-3 helps break down the effects further. Omitting AL-ratios 0.1 (extremely high weights) and 0.5 (uniform weights), the graphic shows how enabling sample weights influences model ranks at different AL-ratios, i.e., different weights.

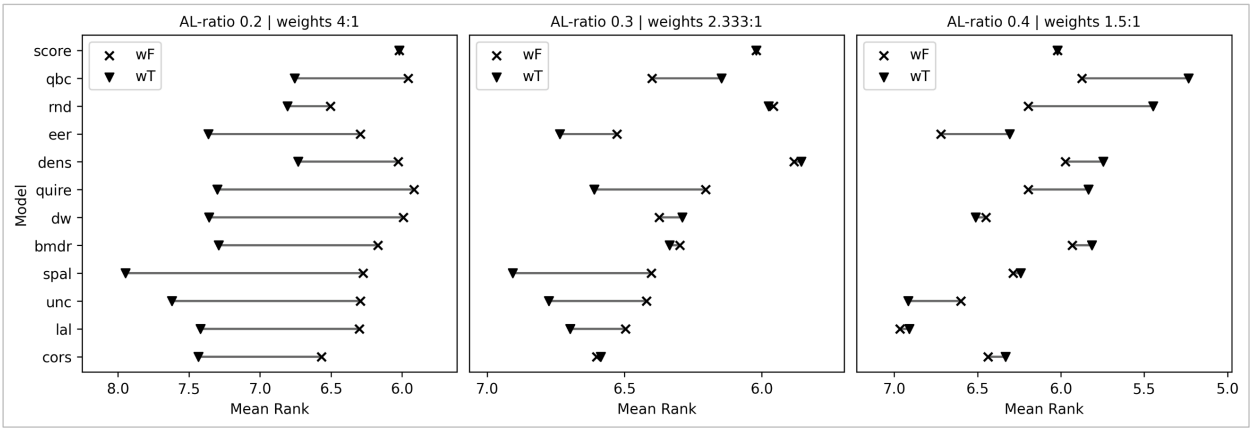

Figure 5-3: Influence of Sample Weights on Average Ranks at Different AL-Ratios

While all models drop in rankings at high weights, the effect lessens at AL-ratio 0.3, where some models are less affected or even gain slightly in ranking. At AL-ratio 0.4, which translates to sample weights of 1.5, most models either improve in performance, or are unaffected. Only the UNC model still has lower average ranks even with low sample weights. The positive effects of moderate sample weights are generally higher for the top-ranking models. The analysis allows for the conclusion that moderately increasing sample-weights for AL-selected instances can be useful for some model configurations. Since weights are not constant across AL-ratios in the experiment, it is difficult to derive a clear recommendation regarding the optimal weight setting, though it is clear that very high sample weights are not preferable in any case, for any model.

5.3. Comparing Performance of Different Model-Types

Section 3 defines three types/families of AL-models. To see if there are distinct performance differences between model families, Figure 5-4 presents multiple box and scatter plots, each representing the performance ranks of all models within the respective model family[6].

[6] Ranks are presented on the original scale from 1 to 111, since there are 10 versions of each AL-model (11 models × 5 AL-ratios × 2 weight options), plus the SCORE baseline model.

For most metrics, the SCORE model ranks within the interquartile range across model families, so not considerably different from most models. It ranks notably worse than average for the BS metric, and notably better for the ICPL. The former result is congruent with Figure 5-1 – the SCORE model performs consistently bad in the Brier Score. The dominant performance of the SCORE model in terms of ICPL is not a surprise, considering that it only strives to select the lowest-risk instances, while any AL-model is more likely to query bad ones.

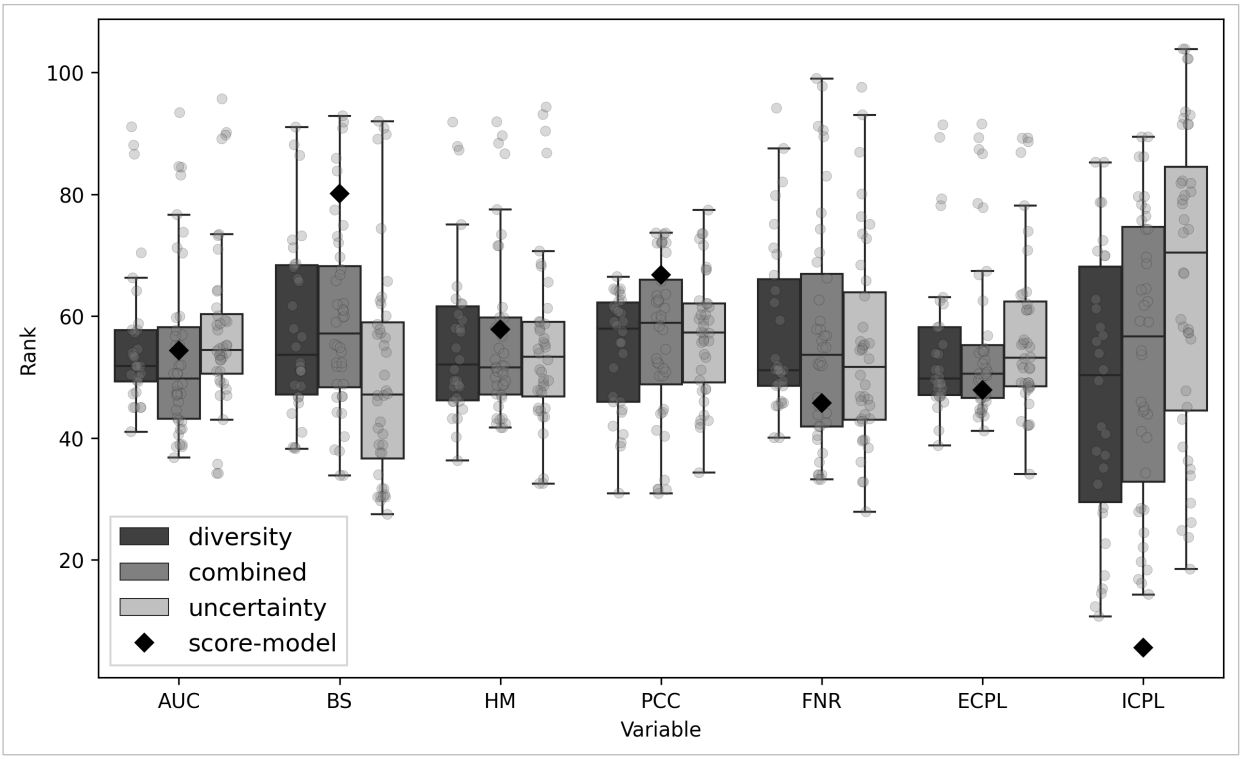

Figure 5-4: Boxplots of Model-Ranks for Each Metric, Grouped by AL-Model Type

Interestingly, the BS and ICPL are also the metrics for which a difference in performance can be observed between the three AL-model types. The uncertainty-based models as a group achieve better ranks for the Brier Score, and worse ranks in terms of ICPL. This is logical if one considers that in highly imbalanced applications such as Credit Scoring, a classifier will be most uncertain about rare instances (i.e., the minority BAD class). Thus, querying such instances is the priority for AL-models with uncertainty-focused utility functions. Consequently, selecting rare instances will improve calibration of the classifier, as it trains on more-balanced data. This will reflect in the BS. On the other hand, since querying bad instances is of course costly, the labelling cost represented by the ICPL will be very high. In contrast, models in the diversity/representativeness group will strive to preserve the natural distribution of the data, hence they will not predominantly select rare (bad) instances, which keeps labelling cost lower in comparison.

5.4. Economic considerations

The analysis of economic implications is footed in the assumption that two types of cost incur in the experiment: the cost of labelling (selecting) instances, and the cost of predicting on a test set of unseen instances. The general premise of the experiment is to investigate whether increased labelling cost through AL are offset by a reduction of future cost through better predictions.

As laid out in section 4, the queries of different AL-models lead to differences in the data used for training the classifier. Since it can be asserted that any instances in the labelled data (i.e., all granted loans) effectively have the predicted class 0 (i.e., GOOD), and since the true class of these instances is known, the cost matrix can be applied to the set of labelled data via the confusion matrix. This makes it interesting to compare the labelling cost (ICPL) between AL-models, i.e., the cost of deploying an AL-model, and to relate it to the external cost (ECPL). To get results that are on a comparable scale across datasets, ICPL and ECPL are computed by dividing the total cost from the respective confusion matrix by the number of instances. Figure 5-5 shows the results from the comparison of both cost dimensions.

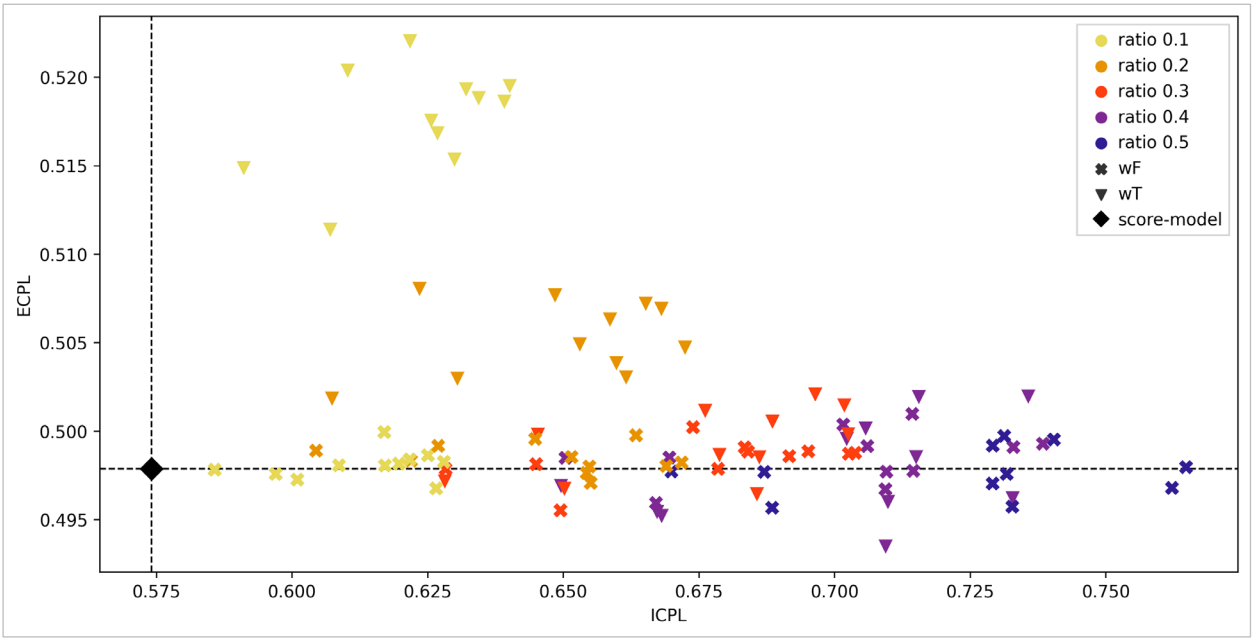

Figure 5-5: Scatter Plot of ICPL and ECPL; Grouped by Ratio and Weight Class

Instead of ranks, the plot shows raw scores (cost per loan). Each marker represents one model's performance in both categories. Markers are coloured according to the respective AL-ratio and shaped according to the weight option. The score model is shown in a separate styling. The figure reiterates the dominance of the SCORE model in the ICPL dimension. Also, it is clearly visible how the prediction performance in terms of ECPL deteriorates when very high sample weights are applied.

Further, the graphic shows that several models achieve lower ECPL than the SCORE model, meaning that these models do in fact reduce the expected cost of any future prediction. It is important to note that these gains always come at the price of increased labelling cost. No model dominates the SCORE model in both dimensions. Generally, the margins in the ECPL dimension are much slimmer, meaning that models are closer together. Consequently, a small decrease in external cost is often connected to a substantial increase in labelling cost. This underlines how important it is to navigate the trade-off between both dimensions. Even at low margins, a better performance in the ECPL may still outweigh increased ICPL labelling cost when one considers the size of the labelled sample versus the size of the unlabelled test data on which the ECPL is based.

Deriving practical recommendations from these results is challenging. The question of breaking even when deciding for an AL-model is influenced by the number of samples that are queried. Further, the analysis is constrained by the assumptions on which the cost matrix is constructed. The true cost structure of a dataset might be very different from the one applied in this experiment. Lastly, having a blind spot at the low ratios/low weights combination excludes some potentially efficient model specifications.

6. Discussion

Being one of the earliest studies – if not the first – to apply AL-methodology to the Credit Scoring domain, this thesis breaks new ground in many areas. The experiment is modelled closely after the real-life Credit Scoring practice by running over multiple generations, gradually adding AL-curated samples to the training data and repeatedly re-training the classifier to predict on a periodical influx of new credit applicant samples. The development of a complex parameter optimization scheme accentuates the challenges of having a triangular relationship between dataset, classifier, and AL-model.

New contributions to the literature include the structured application of different AL-ratios and dynamic sample weights, whose effects are measured and compared to offer a point of reference for future research and practical applications. Without any indication in the literature, a multitude of different model specifications are run and compared, leading to the identification of potential trends and best-practices.

The paper also models the concepts of internal and external misclassification cost, both based on the cost matrix, that serve as main categories by which economic impact can be measured when

applying AL to Credit Scoring. The classification threshold is set to directly optimize the external cost. For the computation of costs, a dynamic cost matrix is used, which sets the cost of a false negative inversely proportional to the BAD rate in the respective dataset. Both the structure of the cost matrix as well as the threshold tuning process are well-grounded in the literature. Results from the benchmark comparison undergo thorough statistical testing according to established best-practices, and the significant results from the benchmark and the subsequent analysis underline the value of AL-selections when maximizing the performance of credit scorecards.

Ultimately, there are several factors limiting the scope of this experiment. First, the nature of the data itself and the standard procedure of data shuffling may aid the SCORE model and mask some of its weaknesses. Namely, there is no population drift in the data, which, as established in section 2.1, would potentially lead to attrition of performance when strictly accepting instances via their predicted probability of default. In real-world applications, any credit scorecard would face such changes to the population over time. Not having such effects in the data reduces the value that AL-models can provide.

A second constraint can be found in the parameter optimization scheme. While each AL-model is tuned for each dataset, as is the LR classifier, the parameter optimization is data-centric and does not consider the interaction between AL-models and the classifier. Especially for models which directly base their queries on information gained from the classifier, this can be a problem. Also, the processes of parameter optimization, threshold tuning, and model selection should ideally be based on the same performance indicator. Mainly due to complexity constraints, model parameters are tuned to optimize AUC performance, while the optimal threshold is selected via a cost criterion, the latter being the more desirable approach from a practical perspective. AL-models in the experiment are selected by their average rankings across seven performance metrics. In practice, the most important selection criteria would likely be cost-based.

On the other hand, there are constraints to the cost model as well. Lacking cost matrices specific to each dataset, the cost is set inversely proportional to the ratio of BAD instances in the data. This approach is supported by the literature (Verbeke, et al., 2017), but ultimately it remains unclear how well the cost model represents each dataset. This affects confidence in the ICPL and ECPL cost measures and limits the economic scope of the experiment, especially because in the main comparison, the ranks in the ECPL metric are never significantly different from another. Instead, the rankings in this category seem rather volatile between different AL-ratios and datasets, as shown in Figure 5-1. Another caveat is the fact that misclassification cost in any practical Credit

Scoring application will likely be instance-dependent due to highly individual credit amounts and terms. Due to the nature of the data, this is not an option here.

To maintain comparability between models, fixed AL-ratios are used. Further, the overall number of instances accepted by either their prediction score or by AL is determined by the acceptance threshold (see section 4.7). While it is conceivable that institutions would grant credits until a fixed budget is exhausted (e.g., when granting micro loans, or in public banking), the more likely scenario is to only grant credit when a risk is deemed good. Therefore, having a classification threshold that is tuned to minimize some cost or error function is common practice. For AL to be an effective tool in real-life applications, it would be ideal to have a utility threshold in place. Similar to the classification threshold, only instances that exceed the utility threshold of the AL-model would be selected, thus ensuring that all selections actually improve the model. Such an approach would keep cost low. The difficulty lies of course in determining such a threshold, which is an interesting field of research for future work in the field.

Lacking precedent in the literature, sample weights are dynamically set in the experiment. While it becomes clear that moderately increasing sample weights of AL-selected instances can be beneficial, and that high sample weights are problematic, it would be interesting in a next step to see how well models with a low AL-ratio and low to moderate sample weights do perform. These combinations are not present in the current experiment, which leaves a blind spot in the analysis. Low ratios are the most promising, selecting fewer instances via AL reduces labelling cost. Also, a smaller batch ensures that the most promising instances are kept, while a high AL-ratio might keep instances that are only moderately useful for the classifier to learn from.

Having the experiment run over multiple generations provides limited additional insight. A benefit is having more data points, which may lead to smoother, more robust results. But the analysis of local or timely effects ultimately exceeds the scope of this experiment: Having many model specifications, datasets, and generations of data requires a lot of aggregation of results, through which some effects are lost. Also, since no population drift is contained in the data, the evolution of performance over periods becomes less interesting.

Lastly, the good performance of the random selection approach compared to the SCORE model combined with the fact that cost reductions are generally not easily attained, puts into perspective the expected benefits versus the effort required to deploy an effective AL-pipeline in Credit Scoring.

7. Conclusion

This thesis paper describes in detail an elaborate experiment that, as a first, deploys Active Learning to contribute to solving the well-established Credit Scoring problem. In a round-based experiment, 11 AL-models representing three different model types select a varying number of instances from the rejected applicant-population to be granted a loan. The AL-selected instances are either weighted inversely proportional to their share in the total training population, or not. Due to these variations in weight and share, a total of 110 different model-parameter configurations are run and benchmarked against one another, and against the baseline SCORE model which strictly selects instances based on their predicted probability of default.

The experiment delivers promising results. Across multiple well-established performance metrics, AL-supported models frequently outperform a conventional SCORE model by a significant margin. The core hypothesis that Active Learning can improve the predictive performance of a credit scorecard is supported by the results. Many of the models that select a share of instances via alternative criteria apart from their probability of default perform decidedly better than the conventional SCORE model in the Brier Score, H-measure, and PCC metrics, indicating that the quality of predictions is improved via an AL-augmented training sample.

The results also show that it does not require a complex model with high computational requirements to achieve good results. The leading QBC model is rather simple and quick compared to others, and the second-best random selection approach (RND) does not require an AL-model at all. The good performance of the random approach indicates that, indeed, the SCORE model is inhibited by the well-known selection bias, which even random selections can alleviate to some extent. Adding random selections to the training data keeps its distribution closer to the applicant population and hence improves predictive ability of the classifier.

However, the competitive results from the RND model also indicate that not all models are worth implementing and optimizing, since the random model requires neither. The complexity of the experiment required some compromises regarding model optimization. Also, there remains a gap to be bridged in terms of economic impact, which is the most crucial metric when analysing credit scorecard performance (Lessmann, et al., 2015). While some models achieve a reduction of cost when predicting on unseen data, there is often a disproportionately larger increase in labelling cost, driven by costly selections by the AL-models. This is especially true for model versions that select a relatively large share of instances via AL.

8. References

Abdou, H. A. & Pointon, J., 2011. Credit scoring, statistical techniques and evaluation criteria: a review of the literature. *Intelligent systems in accounting, finance and management,* 18(2-3), pp. 59-88.

Abe, N. & Mamitsuka, H., 1998. *Query Learning Strategies Using Boosting and Bagging.* Madison, WI, USA, ICML.

Ala'raj, M. & Maysam, F. A., 2016. Classifiers consensus system approach for credit scoring. *Knowledge-Based Systems,* Volume 104, pp. 89-105.

Attenberg, J. & Provost, F., 2011. Inactive Learning? Difficulties Employing Active Learning in Practice. *ACM SIGKDD Explorations Newsletter,* 12(2), pp. 36-41.

Baesens, B. et al., 2003. Benchmarking state-of-the-art classification algorithms for credit scoring. *Journal of the Operational Research Society,* 54(6), pp. 627-635.

Bahnsen, A. C., Aouada, D. & Ottersten, B., 2014. *Example-Dependent Cost-Sensitive Logistic Regression for Credit Scoring.* s.l., IEEE.

Banasik, J., Crook, J. & Thomas, L., 2003. Sample Selection Bias in Credit Scoring Models. *The Journal of the Operational Research Society,* August, 54(8), pp. 822-832.

Cohn, D., Atlas, L. & Ladner, R., 1994. Improving Generalization with Active Learning. *Machine Learning,* Volume 12, pp. 201-221.

Crone, S. F. & Finlay, S., 2012. Instance sampling in credit scoring: An empirical study of sample size and balancing. *International Journal of Forecasting,* Volume 28, pp. 224-238.

Demsar, J., 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *Journal of Machine Learning Research,* Volume 7, p. 1–30.

Devos, A. et al., 2018. Profit Maximizing Logistic Regression Modeling for Credit Scoring. *Proceedings of the IEEE Data Science Workshop (DSW).*

Ditrich, J., 2015. *Selection Bias Reduction in Credit Scoring Models.* Prague, The 9th International Days of Statistics and Economics.

Dumitrescu, E., Hué, S., Hurlin, C. & Tokpavi, S., 2022. Machine learning for credit scoring: Improving logistic regression with non-linear decision-tree effects. *European Journal of Operational Research,* 297(3), pp. 1178-1192.

Ebert, S., Fritz, M. & Schiele, B., 2012. *A reinforced active learning formulation for object class recognition.* Providence, RI, USA, IEEE, pp. 3626-3633.

Elkan, C., 2001. The Foundations of Cost-Sensitive Learning Article. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence.*

Farquhar, S., Gal, Y. & Rainforth, T., 2021. *On Statistical Bias In Active Learning: How and When To Fix It.* s.l., preprint.

Fawcett, T., 2006. An introduction to ROC analysis. *Pattern recognition letters,* 27(8), pp. 861-874.

Fu, Y., Zhu, X. & Li, B., 2012. *A survey on instance selection for active learning.* London, Springer.

García, S., Fernández, A., Luengo, J. & Herrera, F., 2010. Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. *Information Sciences,* Volume 180, p. 2044–2064.

García, V., Marqués, A. & Sánchez, J., 2012. *Improving Risk Predictions by Preprocessing Imbalanced Credit Data.* Berlin, Heidelberg, Springer.

Hand, D. J., 2005. Good practice in retail credit scorecard assessment. *Journal of the Operational Research Society,* 56(9), pp. 1109-1117.

Hand, D. J., 2009. Measuring classifier performance: a coherent alternative to the area under the ROC curve. *Machine Learning,* Volume 77, pp. 103-123.

Hand, D. J. & Adams, N. M., 2014. Selection bias in credit scorecard evaluation. *Journal of the Operational Research Society,* 65(3), pp. 408-415.

Hand, D. J. & Anagnostopoulos, C., 2014. A better Beta for the H measure of classification performance. *Pattern Recognition Letters,* Volume 40, pp. 41-46.

Hand, D. J. & Henley, W. E., 1993. Can reject inference ever work?. *IMA Journal of Mathematics Applied in Business & Industry,* Volume 5, pp. 45-55.

Hand, D. J. & Henley, W. E., 1997. Statistical Classification Methods in Consumer Credit Scoring: A Review. *Journal of the Royal Statistical Society. Series A (Statistics in Society),* 160(3), pp. 523-541.

Hanley, J. A. & McNeil, B. J., 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. *Radiology,* Volume 143, pp. 29-36.

Hastie, T., Tibshirani, R. & Friedman, J., 2009. *The Elements of Statistical Learning.* 2 ed. New York: Springer.

Hofer, V. & Krempl, G., 2013. Drift mining in data: A framework for addressing drift in classification. *Computational Statistics and Data Analysis,* Volume 57, pp. 377-391.

Huang, S.-J., Jin, R. & Zhou, Z.-H., 2010. *Active Learning by Querying Informative and Representative Examples.* s.l., Advances in Neural Information Processing Systems 23 (NIPS 2010), pp. 1936-1949.

Japkowicz, N. & Stephen, S., 2002. The Class Imbalance Problem: A Systematic Study. *Intelligent data analysis,* 6(5), pp. 429-449.

Joanes, D. N., 1994. Reject inference applied to logistic regression for credit scoring. *Journal of Mathematics Applied in Business & Industry,* Volume 5, pp. 35-43.

Konyushkova, K., Sznitman, R. & Fua, P., 2017. *Learning Active Learning from Data.* Long Beach, CA, USA, s.n., pp. 4228-4238.

Kozodoi, N., Jacob, J. & Lessmann, S., 2022. Fairness in credit scoring: Assessment, implementation and profit implications. *European Journal of Operational Research,* 297(3), pp. 1083-1094.

Kozodoi, N. et al., 2020. *Shallow Self-Learning for Reject Inference in Credit Scoring.* Cham, Springer, pp. 516-532.

Kozodoi, N., Lessmann, S., Papakonstantinou, K. & Baesens, B., 2019. A Multi-Objective Approach for Profit-Driven Feature Selection in Credit Scoring. *Decision support systems,* Volume 120, pp. 106-117.

Kuhn, M. & Johnson, K., 2013. *Applied predictive modeling.* 26 ed. New York: Springer.

Kumar, P. & Gupta, A., 2020. Active Learning Query Strategies for Classification, Regression, and Clustering: A Survey. *Journal of Computer Science and Technology,* 35(4), pp. 913-945.

Lessmann, S., Baesens, B., Seow, H.-V. & Thomas, L. C., 2015. Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research,* Issue 247, pp. 124-136.

Lewis, D. & Gale, W., 1994. A sequential algorithm for training text classifiers. *Proceedings of the 17th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval,* pp. 3-12.

Louzada, F., Ara, A. & Fernandes, G. B., 2016. Classification methods applied to credit scoring: A systematic review and overall comparison. *Surveys in Operations Research and Management Science,* 21(2), pp. 117-134.

Mancisidor, R. A., Kampffmeyer, M., Aas, K. & Jenssen, R., 2020. Deep generative models for reject inference in credit scoring. *Knowledge-Based Systems,* Volume 196.

Marqués, A. I., García, V. & Sánchez, J. S., 2013. On the suitability of resampling techniques for the class imbalance problem in credit scoring. *Journal of the Operational Research Society,* 64(7), pp. 1060-1070.

Marrs, G., Hickey, R. & Black, M., 2010. *The impact of latency on online classification learning with concept drift.* Berlin, Heidelberg, Springer, pp. 459-469.

Michie, D., Spiegelhalter, D. J. & Taylor, C. C., 1994. *Machine learning, neural and statistical classification.* s.l.:Ellis Horwood Ltd.

Nayak, G. N. & Turvey, C. G., 1997. Credit risk assessment and the opportunity costs of loan misclassification. *Canadian Journal of Agricultural Economics/Revue canadienne d'agroeconomie,* 45(3), pp. 285-299.

Pavlidis, N. G., Tasoulis, D. K., Adams, N. M. & Hand, D. J., 2012. Adaptive consumer credit classification. *Journal of the Operational Research Society,* 63(12), pp. 1645-1654.

Petrides, G. et al., 2020. Cost-sensitive learning for profit-driven credit scoring. *Journal of the Operational Research Society.*

Ren, P. et al., 2022. A survey of deep active learning. *ACM Computing Surveys,* 54(9), pp. 1-40.

Richards, J. et al., 2011. Active learning to overcome sample selection bias: Application to photometric variable star classification. *The Astrophysical Journal,* 744(2), pp. 192-211.

Roy, N. & McCallum, A., 2001. Toward optimal active learning through sampling estimation of error reduction. *Proceedings of the 18th International Conference on Machine Learning,* pp. 441-448.

Sener, O. & Savarese, S., 2018. Active Learning for Convolutional Neural Networks: A Core-Set Approach. *International Conference on Learning Representations (ICLR).*

Settles, B., 2008. *Curious machines: Active learning with structured instances.* Madison, WI, USA: Dissertation. University of Wisconsin, 2008.

Settles, B., 2009. *Active Learning Literature Survey,* Madison, Wisconsin, USA: University of Wisconsin-Madison Department of Computer Sciences.

Settles, B., Craven, M. & Friedland, L., 2008. Active Learning with Real Annotation Costs. *Proceedings of the NIPS Workshop on Cost-Sensitive Learning,* Volume Vol. 1.

Seung, H. S., Opper, M. & Sompolinsky, H., 1992. *Query by committee.* PA, USA, ACM, pp. 287-294.

Sheng, V. & Ling, C., 2006. Thresholding for Making Classifiers Cost-sensitive. *Proceedings of the National Conference on Artificial Intelligence,* Volume 6, pp. 476-81.

Stone, M., 1974. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society,* Volume Series B (Methodological), pp. 111-147.

Tang, Y. & Huang, S., 2019. Self-paced active learning: Query the right thing at the right time. *Proceedings of the AAAI Conference on Artificial Intelligence,* Volume 33, pp. 5117-5124.

Tang, Y., Li, G. & Huang, S., 2019. *ALiPy: Active Learning in Python.* s.l.:s.n.

Thomas, L., Edelman, D. & Crook, J., 2002. *Credit Scoring and Its Applications.* s.l.:s.n.

Verbeke, W., Baesens, B. & Bravo, C., 2017. *Profit-Driven Business Analytics: A Practitioner's Guide to Transforming Big Data into Added Value.* s.l.:John Wiley & Sons.

Verbraken, T., Bravo, C., Weber, R. & Baesens, B., 2014. Development and application of consumer credit scoring models using profit-based classification measures. *European Journal of Operational Research,* Volume 238, pp. 505-513.

Verstraeten, G. & Van den Poel, D., 2005. The Impact of Sample Bias on Consumer Credit Scoring Performance and Profitability. *Journal of the Operational Research Society,* Volume 56, p. 981–992.

Viaene, S. & Dedene, G., 2005. Cost-sensitive learning and decision making revisited. *European Journal of Operational Research,* Volume 166, pp. 212-220.

Wainer, J. & Cawley, G., 2021. Nested cross-validation when selecting classifiers is overzealous for most practical applications. *Expert Systems with Applications,* Volume 182.

Wallace, B. C. & Dahabreh, I. J., 2014. Improving class probability estimates for imbalanced data. *Kowledge Information Systems,* Volume 41, pp. 33-52.

Wang, Z. & Ye, J., 2015. Querying discriminative and representative samples for batch mode active learning. *ACM Transactions on Knowledge Discovery from Data (TKDD),* 9(3), pp. 1-23.

West, D., 2000. Neural network credit scoring models. *Computers and Operations Research,* 27(12), p. 1131–1152.

Xia, Y., Liu, C., Li, Y. & Liu, N., 2017. A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring. *Expert Systems with Applications,* 78(225-241).

Yang, Y. & Loog, M., 2018. A Benchmark and Comparison of Active Learning for Logistic Regression. *Pattern Recognition,* Volume 83, pp. 401-415.

Zhan, X., Liu, H., Li, Q. & Chan, A. B., 2021. A Comparative Survey: Benchmarking for Pool-based Active Learning. *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21).*

Zhong, Y., He, J. & Chalise, P., 2020. Nested and Repeated Cross Validation for Classification Model With High-dimensional Data. *Revista Colombiana de Estadística,* 43(1), pp. 103-125.

APPENDIX A: Overview of AL-models

Table A-1: AL-Models by Group, with Model Parameters

Model	Short	Source	Type	Parameters (subject to tuning)	Notes
Uncertainty	UNC	(Lewis & Gale, 1994)	Uncertainty	uncertainty measure: ['entropy', 'least_confident', 'margin', 'distance_to_boundary']	
Query by Committee (Bagging)	QBC	(Seung, et al., 1992; Abe & Mamitsuka, 1998)	Uncertainty	disagreement: [vote_entropy, KL_divergence]	
Expected Error Reduction	EER	(Roy & McCallum, 2001)	Uncertainty	–	No tunable parameters
Learning Active Learning	LAL	(Konyushkova, et al., 2017)	Uncertainty	# estimators random forest 1 (rf1): [16, 32, 64] # estimators rf2: [32, 64, 128], depth rf2: [5, 10, 20], maximum features rf2: [5, 6 ,7]	
Random	RND	(Hand & Henley, 1993)	Diversity	–	No model required
Core-Set (Greedy Kcenter)	CORS	(Sener & Savarese, 2018)	Diversity	distance metric: ['cityblock', 'cosine', 'euclidean']	
Graph Density	DENS	(Ebert, et al., 2012)	Diversity	distance metric: ['canberra', 'jaccard', 'cosine', 'hamming']	
Querying Informative and Representative Examples	QUIRE	(Huang, et al., 2010)	Combined	–	Due to stability issues, the model is run with default parameters.
Density Weighted	DW	(Settles, 2008)	Combined	uncertainty measure: ['least_confident', 'margin', 'entropy'], distance metric: ['cityblock', 'cosine', 'euclidean'], β (tradeoff parameter): [0.5, 1, 2]	
Discriminative and Representative Queries for Batch Mode Active Learning	BMDR	(Wang & Ye, 2015)	Combined	β (of Maximum Mean Discrepancy – MMD): [0.01, 0.1, 1], γ (L2 parameter): [0.01, 0.1, 1]	Due to stability issues, BMDR could not be tuned on the Thomas data.
Self-Paced Active Learning	SPAL	(Tang & Huang, 2019)	Combined	μ (of Maximum Mean Discrepancy – MMD): [0.01, 0.1, 1], γ (L2 regularizer parameter): [0.01, 0.1, 1], $\lambda_{initial}$ (of self-paced learning – SPL): [0.01, 0.1, 1] , λ_{pace} (when updating): [0.001, 0.01, 0.1]	Due to stability issues, SPAL could not be tuned on the Thomas data.

APPENDIX B: Average Ranks of Best Models Over Periods

Figure B-1 shows how the rankings from Table 5-1 evolve when spread out over periods[7]. Periodical effects across datasets seem to be very individual in general and rather volatile in some cases. Two exceptions in terms of volatility are the BS and ICPL metrics, which show spread out rankings with little shifts. In case of the ICPL metric this is because the metric is solely based on the selections of the AL-model and as such independent of the classifier predictions. Instead, the selection patterns of each AL-model determine the labelling cost incurred. Unsurprisingly, models with high AL-ratios incur higher labelling cost relative to models with lower ratios. Among those models with high AL-ratios, there are distinct differences in the cost they incur. The EER model at its optimal specification incurs the highest labelling cost at a wide margin. The overall best ranking QBC model also makes costly choices on average. Interestingly, the plots for the ICPL and BS show opposite effects. The models with the highest labelling costs are also the ones with the best average ranks in the Brier Score. A logical explanation is that in order to achieve good calibration of the model (BS), a larger number of costly queries must be gathered.

Of interest in the other metrics are any curves having visible slopes, indicating an improvement or decline in performance relative to other models as the generations progress. The SCORE model shows generally low and progressively declining performance in the BS metric. A slightly upward slope indicating worse rankings over time can also be seen in the FNR and ECPL metrics. The SCORE model's ability to correctly predict BAD cases in the test data seems to deteriorate as periods progress, increasing cost. In terms of AUC, the SCORE model achieves a small increase in relative performance, showing a slightly negative slope in its rankings curve. For other metrics, no clear slope is visible in case of the score model. The overall best QBC model on the other hand shows constant improvement in the FNR and, consequently, the ECPL metrics over periods, while showing a slight decline in PCC compared to other models. This is an indication that the model gradually learns to better predict BAD cases in the data at the expense of correctly predicting GOOD cases. In other words, as periods progress the model errs toward predicting BAD, which reduces costly false negatives (FN) and increases false positives (FP). The performance in the PCC metric, only measuring how many predictions are correct, suffers from this increase in FPs. This makes sense, since there are far more GOOD cases in the data, i.e., the FPs will grow at a faster

[7] Since the graphic displays rankings on the y axis, low y values are desirable. Also, ranks are relative, so for a model to rank worse it suffices if it just does not improve as much over periods as other models.

pace when a model increasingly predicts the positive (BAD) class. The development of the QBC model in the FNR is particularly impressive, starting off with the worst average rank and ending at the best rank in the last period. This also puts into perspective the QBC's mediocre performance in terms of FNR the main comparison table, which, being averaged over periods, will mask such positive developments. The overall impression when looking at the ECPL plot is that rankings are especially volatile. The behaviour is similar to Figure 5-1, where it is difficult to discern any general trends of how different AL-ratios influence ECPL performance. Instead, model and data-specific effects seem to play a large role.

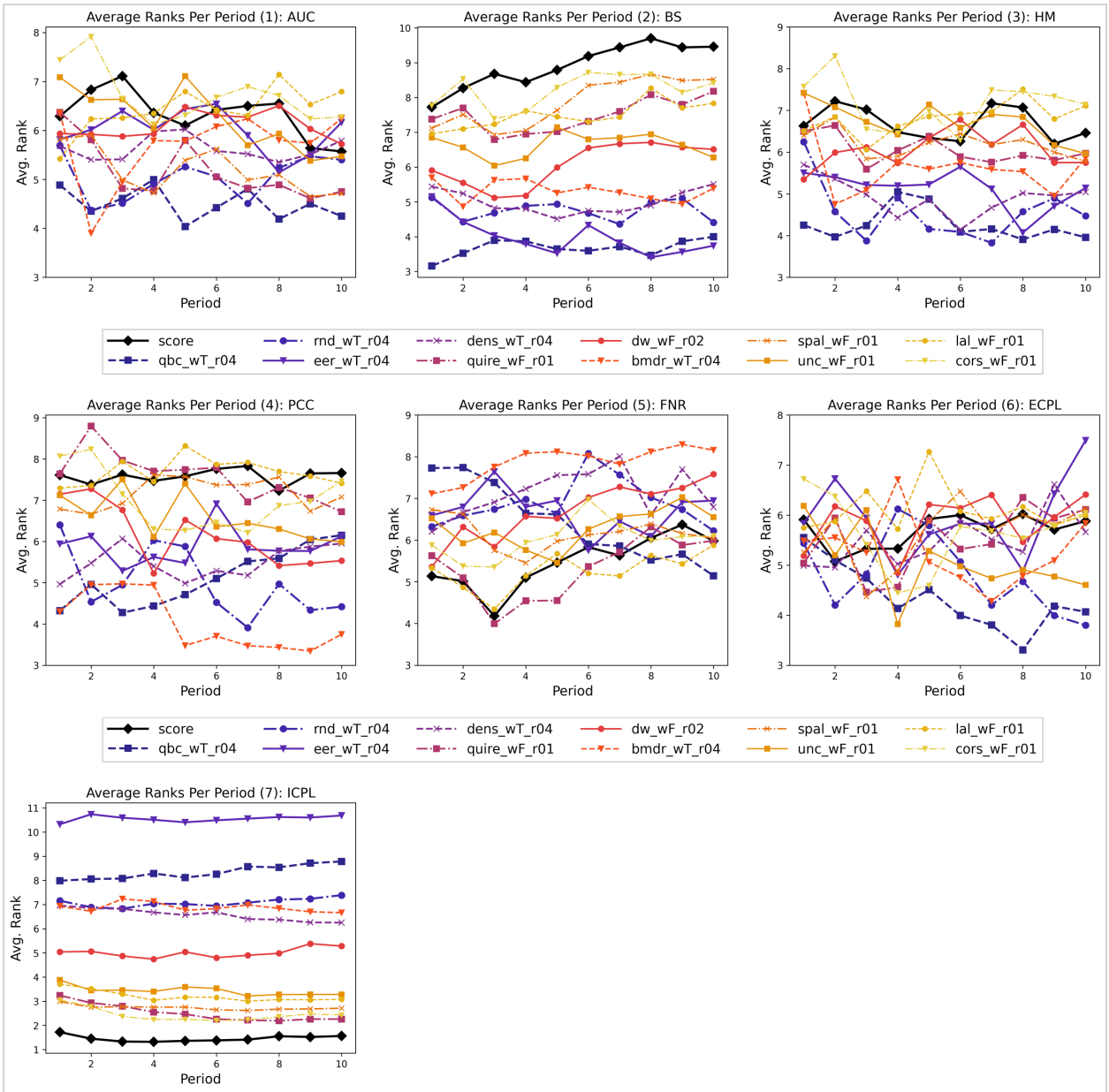

Figure B-1: Performance-Rankings Over Generations

APPENDIX C: Raw Scores Over Periods in Selected Metrics

Graphics in this section show how the raw performance of the best models as selected for Table 5-1 evolves in a specific metric as the generations (periods) progress. Plots are shown separately for each dataset. The metrics selected for display are the HM, BS, and ECPL. In terms of the HM (Figure C-1), the SCORE model performs notably bad on the GMSC and Lendingclub datasets. For the German and HMEQ datasets, its performance curve flattens quicker than other models, indicating that the model's instance selections are comparably less helpful. The QBC model performs strongly on the Australian, GMSC, Lendingclub, and UK datasets. On the other hand, it is decidedly outperformed on the Bene1 data. The DENS model achieves good performance quickly in the German, HMEQ, and Lendingclub data, before the other models catch up. Most models show unusual behaviour on the Thomas data, which might indicate that the LR classifier is generally not a good fit for this dataset.

In the BS dimension (Figure C-2), the QBC model exhibits strong performance once again, generally achieving very low (good) scores from early rounds onwards. The only exception to this rule is again the Bene1 dataset, where the QBC model does not seem to be a good fit across metrics. The score model on the other hand is consistently outperformed, its slope being noticeably flatter on many datasets, meaning the training data selected by the SCORE model helps the classifier less in terms of calibration and prediction quality. Consequently, as periods progress, the model trained on SCORE selected data improves at a lower rate. This effect is most notable on the GMSC, UK, and PAKDD and Thomas datasets. A curious case is the HMEQ dataset, where some models decline in performance over periods.

Lastly, Figure C-3 shows the development of the ECPL scores of each model on each dataset. As before, all models gradually increase their performance on most datasets. The DENS model once again displays good ranks quickly on several datasets before its curve flattens toward later rounds. The QBC model scores well on the GMSC, Australian, and Bene2 data, and poorly on the Bene1 data.

Generally, the analysis of performance across periods seems rather specific to the dataset. Models with competitive performance on one dataset can be outranked significantly on others. This demonstrates that it is important to carefully select the AL-model that best fits the current data. Ultimately, having the experiment run over multiple periods provides limited additional insight for this thesis, which is more concerned with general trends.

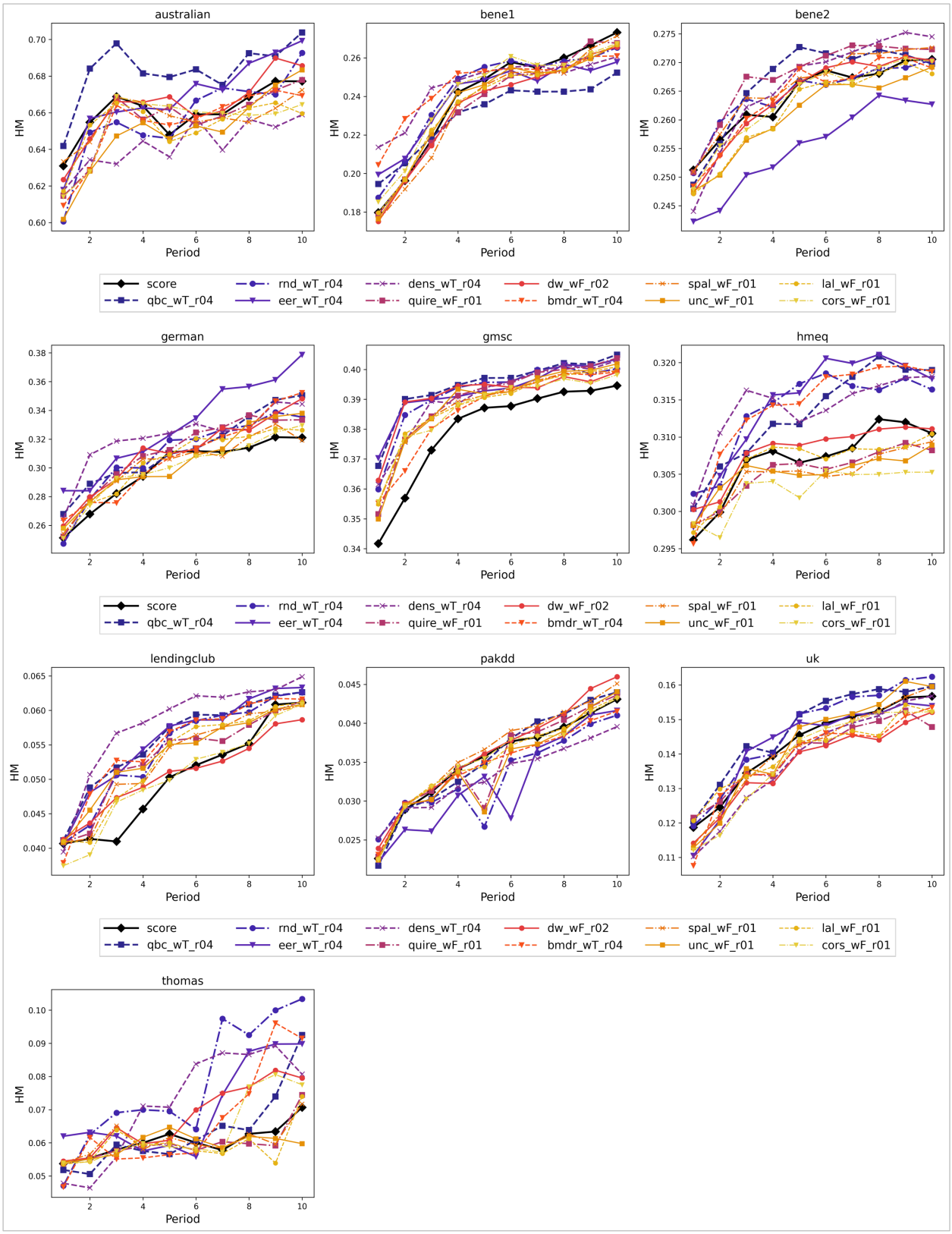

Figure C-1: H-Measure Performance Over Periods, per Dataset

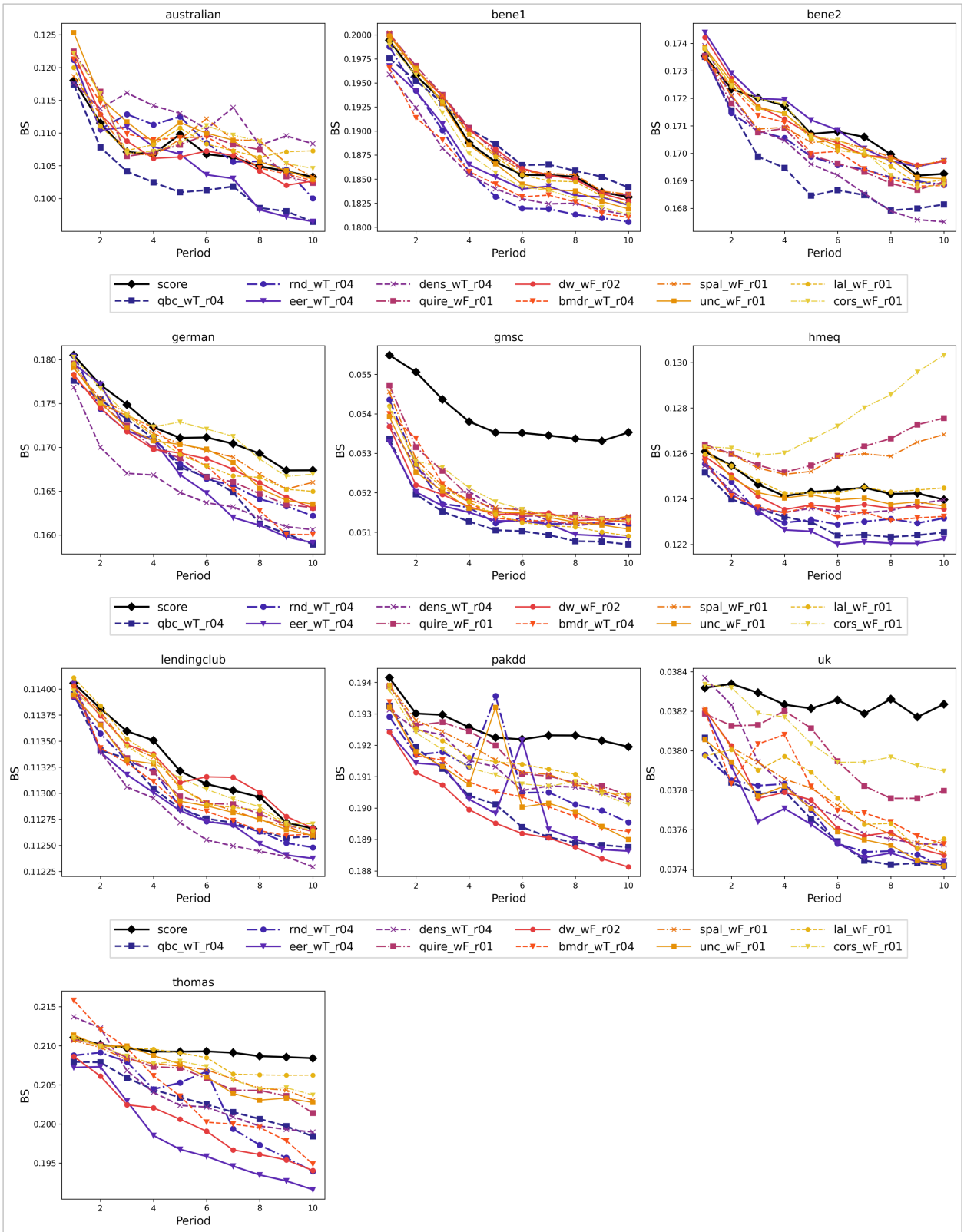

Figure C-2: Brier Score Performance Over Periods, per Dataset

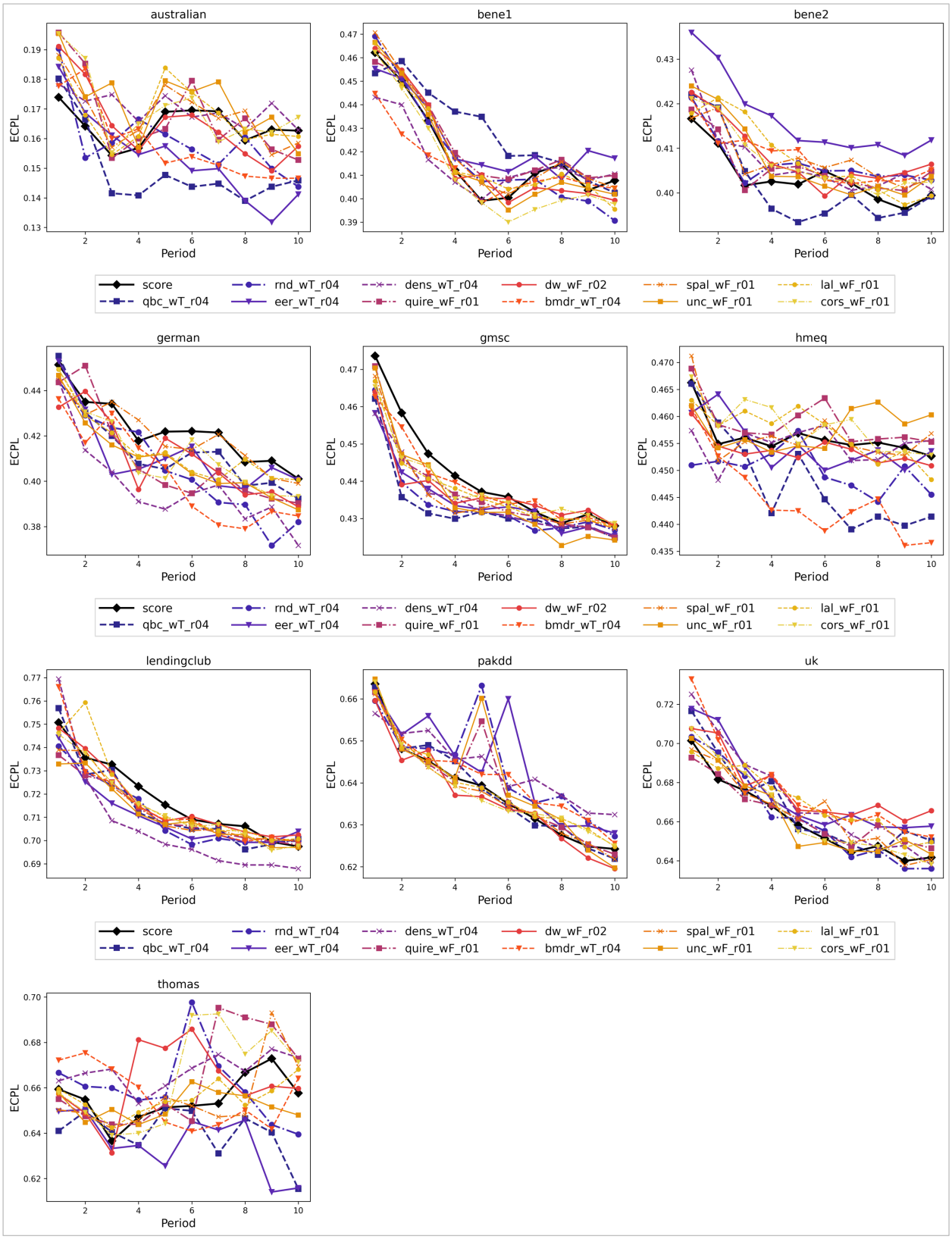

Figure C-3: ECPL Performance Over Periods, per Dataset

Declaration of Academic Honesty

Hiermit erkläre ich, Lukas Kolbe, dass ich die vorliegende Arbeit noch nicht für andere Prüfungen eingereicht habe. Ich habe die Arbeit selbständig verfasst. Sämtliche Quellen einschließlich Internetquellen, die ich unverändert oder abgewandelt wiedergegeben habe, insbesondere Quellen für Texte, Grafiken, Tabellen und Bilder, habe ich als solche kenntlich gemacht. Ich bin mir darüber bewusst, dass bei Verstößen gegen diese Grundsätze ein Verfahren wegen Täuschungsversuchs bzw. Täuschung eingeleitet wird.

I, Lukas Kolbe, hereby declare that I have not previously submitted the present work for other examinations. I wrote this work independently. All sources, including sources from the Internet, that I have reproduced in either an unaltered or modified form (particularly sources for texts, graphs, tables, and images), have been acknowledged by me as such. I understand that violations of these principles will result in proceedings regarding deception or attempted deception.

Lukas Kolbe

Berlin, 29.07.2022