# UniLeiden at LeQua2024: Evaluating Continuous Sweep and Comparison Using Underlying Classifiers

Kevin Kloos

University of Leiden, NL

**Abstract.** We compared the continuous sweep quantifier with median sweep, SLD, and DyS using Task T1 of the LeQua2024 competition data. We fitted 100 different underlying support vector machines and evaluated all quantifiers on all models. Continuous sweep is outperformed by SLD and DyS if a well-performing underlying classifier exists. Using worse quantifiers, continuous sweep and median sweep appeared to be more stable than DyS and SLD. We compared these findings with existing results in quantification literature.

## 1 Introduction

Quantification Learning is a task that is focused on predicting the prevalence of a data set rather than individually labelling each observation [1, 2]. In earlier years, quantification has been a side product of classification, but over the last decades, quantification learning has been developed to a standalone field with sophisticated tasks and methods. One of those tasks are the LeQua competitions which are designed to evaluate and compare quantifiers with each other. The LeQua competitions contains various tasks regarding quantification learning [3]. In this paper, we focus on task T1 of the LeQua2024 competition. This task concerns binary quantifiers from which the data is affected by prior-probability shift.

In this paper, we compare the continuous sweep quantifier with median sweep, SLD, and DyS. All quantifiers use a support vector machine with a radial basis function as an underlying classifier. We not only investigate the raw performance of the quantifiers, but also investigate the relationship with the underlying classifiers. We finalize the paper with a discussion about the competition and a generalization to other studies.

## 2 Methods

In this study, we compare our four quantifiers. These four quantifiers need an underlying classifier to compute probabilities for each observation. In this section, we explain how we computed the underlying classifier and elaborate on the

four quantifiers. Extra emphasis is put on the Continuous Sweep quantifier, because the author of this paper is one of the developers of Continuous Sweep and it is the most unknown quantifier out of the four. Moreover, we submitted the prevalences extracted from Continuous Sweep to the LeQua2024 competition.

**Underlying classifier** The underlying classifier of all quantifiers is a support vector machine using a radial basis function. Using the kernlab package supported by the Tidymodels library [4], we can tune this support vector machine with two hyperparameters: the cost (C) and the radial basis function sigma ($\sigma$). Moreover, the data is preprocessed by normalizing all predictor variables. A max entropy grid is used to find 100 pairs of hyperparameters that optimally cover the hyperparameterspace. We therefore fit 100 SVMs with different hyperparameters. Using the SVM, we can predict the probabilties of observations that belong to the positive class. With 5-fold cross validation, we also estimate the probabilties of all observations in the training data, which will be used to fit the quantifiers.

**Median Sweep (MS)** Median sweep is an ensemble quantifier in the group *Classify, Count, and Correct* [5]. First, we use the SVM to compute probabilities of all observations in the test data. Accordingly, we compute an adjusted count estimate for every probability that occurs in the test data. The adjusted count estimates are computed using true and false positive rate estimated by the cross validated probabilities from the training data. Consequently, we discard unreliable estimates, that is, every adjusted count estimate where the difference between the true and false positive rates are smaller than 0.25. Finally, we compute the median of the remaining adjusted count estimates as our median sweep prevalence estimate.

**Continuous Sweep (CS)** Continuous sweep is a quantifier that is similar to median sweep [6]. More details of Continuous Sweep can be found in [6], but we provide a short explanation of the method. Continuous sweep and median sweep are different in two characteristics. First, continuous sweep uses continuous functions to estimate the true and false positive rates. In this task, we used the kernel cumulative density function from R's *ks* package to compute the true and false positive rates with the cross-validated training data. The continuous functions of the true and false positive functions enabled us to integrate the adjusted count estimates with respect to the estimated probabilities. Therefore, the second change is that we compute the mean of all adjusted count estimates using integration, instead of computing the median. Moreover, we were able to improve the procedure of discarding unreliable adjusted count estimates. We derived derivations for the bias and variance of continuous sweep which enabled us to optimize the value of the minimal difference between true and false positive rates (i.e., $p^{\Delta}$) that an adjusted count estimate is determined as "reliable". The reliable prevalence estimates are located between $\theta_l$ and $\theta_r$. The final continuous

sweep estimate is the mean area under the curve of the adjusted count function between $\theta_l$ and $\theta_r$.

**SLD** The SLD algorithm is a expectation maximization approach to the quantification task [7]. The prevalence of the training data is used as a starting point to update the test prevalence iteratively by an optimal Bayes classifier until convergence is reached. The stopping criterium is a prevalence difference of 0.0001 between two updates or a maximum of 1000 iterations.

**DyS** The DyS algorithm is algorithm that matches histograms to find the prevalence of a test set [8]. First, the cross validated probabilities of the training set are used to construct histograms of the probabilities in the positive and negative class. Moreover, the probabilities of the test set are used to make a histogram for the test data. Using a tenary search, we aim to find the optimal mixture between the positive and negative class probabilities to match the test histogram as good as possible. The measure we used to describe the difference between histogram is the hellinger distance. Moreover, default values of 8 bins are used to construct the histograms and a the stopping criterium is a prevalence difference of 0.0001 between two updates.

## 3 Results

We fitted 100 different SVM-RBF models with different hyperparameters using a max entropy grid. Out of these 100 models, 76 converged. We compared the performance of the four quantifiers with 1) their best MAE overall, and 2) using the model with the lowest ROC-AUC. From Table 1, we see that SLD has the lowest MAE out of the four models in the two scenarios. Moreover, the differences between the minimum MAE with the MAE of the model with the lowest ROC-AUC are small across the quantifiers.

We also compared the performance of the classifiers against their quantification performance, both within as between quantifiers. For all 76 converged models, we evaluated the ROC-AUC of the classifier against the cross-validated probabilities from the training data and we computed the MAE of all four quantifiers on each of the models. Two models had a ROC lower than 0.8, which means that those models perform very bad compared to the other models. Those two models are excluded from the illustrations. In Figure 1, we see that the best classifiers are embraced by all quantifiers. SLD and DyS perform better than CS and MS using the best classifiers. However, these two quantfiers are less stable if the classifiers perform worse. For a ROC-AUC around $[0.90 - 0.92]$, SLD had a remarkable drop in performance. Moreover, out of the 76 converged models, SLD had a MAE lower than 0.05 for 22 models, whereas DyS, MS, and CS had a MAE lower than 0.05 for 47, 45, and 44 respectively. For classifier with a low ROC-AUC, CS and MS outperformed SLD and DyS, but the MAE's respectively are fairly high compared to the good classifiers. Moreover, we extract from Figure 1 that DyS performed either good or very bad for worse classifiers.

**Table 1.** The MAE of four quantifiers evaluated on 1000 development sets. At the top, the lowest MAE of every quantifier across all models. At the bottom, the MAE of every quantifier based on the largest ROC-AUC. The columns of the table denote the quantifier, the model number, the cost hyperparameter (C), the rbf sigma hyperparameter ($\sigma$), the ROC-AUC metric of the model, and the MAE of the quantifier given the classifier.

| Quantifier | Model no. | C | $\sigma$ | ROC-AUC | MAE |
|---|---|---|---|---|---|
| *Continuous Sweep* | 77 | 4.436 | $3.28 \times 10^{-4}$ | 0.9177 | *0.0241* |
| Median Sweep | 74 | 0.236 | $3.68 \times 10^{-3}$ | 0.9099 | 0.0267 |
| SLD | 77 | 4.436 | $3.28 \times 10^{-4}$ | 0.9177 | **0.0205** |
| DyS | 77 | 4.436 | $3.28 \times 10^{-4}$ | 0.9177 | 0.0223 |
| *Continuous Sweep* | 78 | 25.1 | $4.45 \times 10^{-4}$ | 0.9198 | *0.0246* |
| Median Sweep | 78 | 25.1 | $4.45 \times 10^{-4}$ | 0.9198 | 0.0291 |
| SLD | 78 | 25.1 | $4.45 \times 10^{-4}$ | 0.9198 | **0.0209** |
| DyS | 78 | 25.1 | $4.45 \times 10^{-4}$ | 0.9198 | 0.0224 |

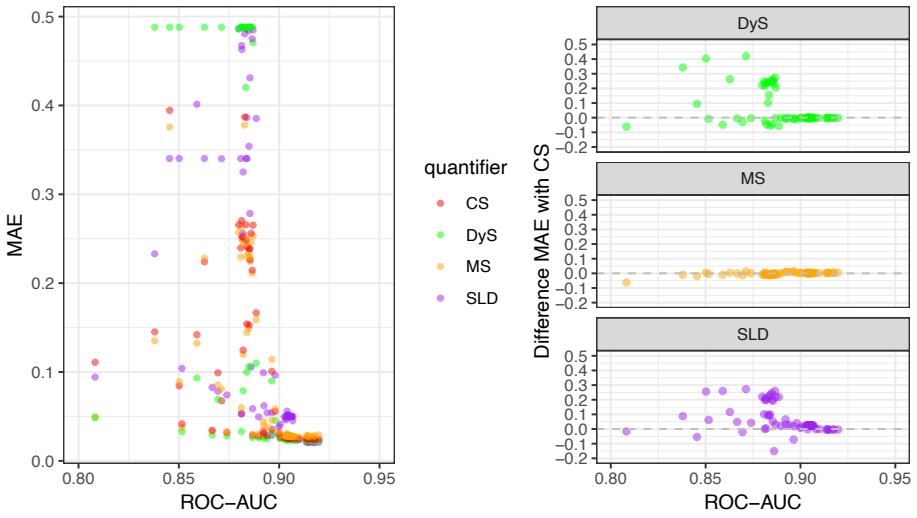


**Fig. 1.** Plot that compares the MAE of he four quantifiers against the ROC-AUC. On the left, we compare the MAE against the ROC-AUC in general for all models. On the left, we compare the difference between the MAE of the displayed quantifier against Continuous Sweep, where a value higher than zero indicates a higher MAE for the respective quantifier.

## 4 Discussion

Our quantifier, Continuous Sweep, underperforms against good baseline quantifiers like SLD and DyS in the LeQua2024 competition. If a good classifier can be obtained from the training data, SLD and DyS performed well. Similar results have been found in the previous LeQua2022 competition, where DyS and SLD were good baseline classifiers [3]. If the underlying classifier performs worse, the ensemble methods such as Continuous Sweep and Median Sweep had a lower performance drop than, and outperformed, SLD and DyS. These findings could be translated to the elaborative comparison of [9], where median sweep generally outperformed SLD and DyS for datasets with a binary target variable. These datasets used were usually much smaller than the datasets of the LeQua competitions. Whereas the LeQua competition contained a training set of 5000 observations with 1000 development sets containing 250 observations, the datasets used in [9] were smaller baseline UCI or Kaggle datasets only splitted into a training and a test set.

In future research, it could be interesting to investigate the performance of quantifiers when it is difficult to construct a good (probabilistic) classifier. Moreover, it could be interesting to investigate whether some decision rules could be developed to choose a suitable quantifier based on the characteristics of an underlying classifier. For example, if one has an excellent classifier, we could safely apply SLD, whereas a worse classifier might be better used by a quantifier like continuous sweep. Last, it might also be interesting to investigate the performance of direct learners when it is difficult to construct a good classifier.

## References

1. González P, Castaño A, Chawla NV, del Coz JJ. A Review on Quantification Learning. ACM Computing Surveys. 2017;50(5):74:1-74:40.
2. Esuli A, Fabris A, Moreo A, Sebastiani F. Learning to Quantify. 1st ed. The Information Retrieval Series. Cham: Springer Nature; 2023.
3. Esuli A, Moreo A, Sebastiani F, Sperduti G. A Detailed Overview of LeQua@ CLEF 2022: Learning to Quantify; 2022. Available from: `https://api.semanticscholar.org/CorpusID:251471941`.
4. Kuhn M, Wickham H. Tidymodels: a collection of packages for modeling and machine learning using tidyverse principles; 2020. Available from: `https://www.tidymodels.org`.
5. Forman G. Quantifying Counts and Costs via Classification. Data Mining and Knowledge Discovery. 2008;17(2):164-206.
6. Kloos K, Karch JD, Meertens QA, de Rooij M. Continuous Sweep: an improved, binary quantifier; 2023. Available from: `https://arxiv.org/abs/2308.08387`.
7. Saerens M, Latinne P, Decaestecker C. Adjusting the Outputs of a Classifier to New a Priori Probabilities: A Simple Procedure. Neural Computation. 2002;14(1):21-41.
8. Maletzke A, dos Reis D, Cherman E, Batista G. DyS: A Framework for Mixture Models in Quantification. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 33; 2019. p. 4552-60.
9. Schumacher T, Strohmaier M, Lemmerich F. A Comparative Evaluation of Quantification Methods; 2023.