Comparison between Co-training and Self-training for single-target regression in data streams using AMRules

Ricardo Sousa[1] and João Gama[1,2]

[1] LIAAD/INESC TEC, Universidade do Porto, Portugal
rtsousa@inesctec.pt
[2] Faculdade de Economia, Universidade do Porto, Portugal
jgama@fep.up.pt

Abstract. A comparison between co-training and self-training method for single-target regression based on multiples learners is performed. Data streaming systems can create a significant amount of unlabeled data which is caused by label assignment impossibility, high cost of labeling or labeling long duration tasks. In supervised learning, this data is wasted. In order to take advantaged from unlabeled data, semi-supervised approaches such as Co-training and Self-training have been created to benefit from input information that is contained in unlabeled data. However, these approaches have been applied to classification and batch training scenarios.
Due to these facts, this paper presents a comparison between Co-training and Self-learning methods for single-target regression in data streams. Rules learning is used in this context since this methodology enables to explore the input information.
The experimental evaluation consisted of a comparison between the real standard scenario where all unlabeled data is rejected and scenarios where unlabeled data is used to improve the regression model.
Results show evidences of better performance in terms of error reduction and in high level of unlabeled examples in the stream. Despite this fact, the improvements are not expressive.

1 Introduction

Prediction represents an essential task in data streams contexts that depend on accurate predictions for decision making or planning [1]. In these contexts, large quantities of data is not labeled due to label assignment impossibility, high cost of label assignment or long time tasks. Frequently, sensitive data requires label omission [4].

The main areas where unlabeled data occurs are Engineering Systems (video object detection) [7], Physics (weather forecasting and ecological models) [8], Biology (model of cellular processes) [9] and Economy/Finance (stock price forecasting) [1]. In most of these areas, data from streams are obtained and processed in real time [4].

Semi-supervised Learning (SSL) methodology have been suggested to use input information from unlabeled data for more accurate predictions [4]. Only in unlabeled examples abundance cases, this methodology may be useful [11]. In fact, the unlabeled examples convey information related to the variety or to the range of the inputs values. These values ranges may create constrains to the models and them more precise. As negative characteristic, this methodology may introduce errors and lead to less accurate predictions [11, 12].

More formally, $\mathbf{X} = \{X_1, ..., X_j, ..., X_J\} \in \mathbb{R}^J$ represents a vector of input random variables and $\mathbf{Y}$ represents a scalar random variable, with a joint probability distribution $P(\mathbf{X}, \mathbf{Y})$. The vectors $\mathbf{x}_i = (x_{i,1}, ..., x_{i,j}, ..., x_{i,J}) \in \mathbb{R}^J$ and y_i, where $i \in \{0, 1, 2, ...\}$, represent realizations of $\mathbf{X}$ and $\mathbf{Y}$, respectively. A stream is defined as the sequence of examples $\mathbf{e}_i = (\mathbf{x}_i, y_i)$ represented as $\mathcal{S} = \{(\mathbf{x}_0, y_0), (\mathbf{x}_1, y_1), ..., (\mathbf{x}_i, y_i), ...\}$. Label absence is represented by $y_i = \emptyset$. The objective of SSL is to use examples $(\mathbf{x}_i, \emptyset)$ to enhance the regression model $y_i \leftarrow f(\mathbf{x}_i)$ and reduce the error of prediction for both labeled and unlabeled examples.

SSL methods work on batch mode and are applied to classification. The imediate adaption to regression is not possible [11]. Co-training is a SSL approach that uses more than one different models. The model diversity is created by through different inputs, different regression methods or different parametrization [4].

The training stage produces an artificial label for the unlabeled example from the regressors predictions of the same example, according to a criterion (e.g., mean of all predictions) [11]. Posteriorly, this artificially labeled example is used in the training of the regressors. The prediction stage yield a final prediction from the regressors predictions of the example, according to a similar criterion as in training.

Self-training can be seen as a particular case of Co-training where just one model is trained. This method uses its own predictions to artificially label the unlabeled examples and use this examples in the respective training.

It worths to say that the active learning can also be easily introduced in these methods. However, once used in the training, the example contribution cannot be removed from the model.

This work focus on the comparison between Self-training method and Co-training method which uses several models that learn with each other for online, single-target regression. Despite expecting better prediction results from Co-training, it is important to find how much the results are superior to Self-training. In fact, Co-training is more computationaly expensive then Self-training. This work also may pave the way for the extension to online multi-target regression using the Random Adaptive Model Rules (Random AMRules) algorithm in future works [15, 16].

This document is structured as follows. In Section 2, the fundamentals of SSL for Self-training and Co-training are briefly revised. Section 3 describes the modifications of the Co-training to online learning and regression using ensembles of rule models. Section 4 describes the evaluation method. The results are discussed in Section 5 and the main conclusions are remarked in Section 6.

2 Related Work

This section explains some concepts used in the development of Co-training and Self-training methods. As general pattern, Co-training involves the training of two or more different models in some aspects (e.g., different inputs, different regressors, different parametrization, different examples ...). The labeled examples are processed as a supervised processure. The unlabeled examples are artificially labeled and processed as a supervised procedure. The artificial label is essentially a prediction (Self-training) or a processed prediction derived from a combination of predictions of complementary learners (Co-training). The learners are considered to predict reliably (confidence driven method).

Co-training methods follow these assumptions: consensus, complementary, sufficiency, compatibility and conditional independence. Self-training only considers sufficiency and compatibility. Note that these assumptions can be applied for both batch and online(incremental) methods.

- **Consensus** assumption states that the more similar the learner predictions are, the more reliable the artificial label is [18].
- **Complementarity** assumption states the learners contain different information and can learn from each other [18].
- **Sufficiency** assumption states that each regressor should be sufficiently consistent (e.g., by enough number of attributes) to build a model.
- **Compatibility** assumption implies that the predictions of different models present the same probabilistic distribution.
- **Conditional independence** assumption gives the chance of at least one learner can produce a more accurate prediction.This prediction can be used to teach the other learners. [19].
- **Conditional independence** assumption considers that the learning process of each Despite being very important for Co-training, the independence assumption is very restrictive. Therefore, related but less restrictive assumptions were considered.
- **Weak dependence** assumption tolerates a small dependence level between inputs which lead to positive results. This assumption overcomes the restrictive characteristic of Conditional independence [20].
- **Large diversity** assumption considers that using different algorithms or the same algorithms but with different parametrization lead to independent models [21].

Concerning the drawbacks, the inaccuracy of the artificially labeled examples introduce error into the models and it is the main cause of model degradation. Moreover, the artificially labeled examples may not carry the information to the regressor leading to unnecessary operations [11]. Different strategies to artificially label or criteria to discard non-beneficial artificially labeled examples may be present in some Self-training and Co-training variants. The prediction stage generally combines the predictions of the models according to a pre-defined criterion to produce the final prediction [11].

3 Online Co-training and Self-training Regression

This section provides the description of a developed a Co-training method and also a Self-training method through the presentation of the main adaptations to the online and regression context. Here, the description was focused in the Co-training algorithm. A small description of the underlying algorithm regressor Random AMRules (ensemble rules based method) is also presented.

The new method that is being proposed divides the inputs variables of the example into two groups randomly which is defined in the initial stage. Here, weak dependence is assumed since no independence information between pairs of attributes is available. The complementarity assumption is also used since each produced model contains information that other does not contain.

The two groups are forced to share a randomly selected inputs by a pre-defined overlap percentage. Two Random AMRules complementary regressors are used to produce artificial labels through prediction for the unlabeled example. The initial models are obtained previously in a training stage using a dataset portion. The size of dataset portion should be sufficient to produce a consistent model. Here, the inputs overlapping increase the number of attributes in each model and contribute for the sufficiency assumption.

A score that reflects the benefit or confidence of artificially labeled example is calculated for the decision of being accepted for training. The score is the relative difference (RD) compared to the maximum of absolute values of the output found in the stream y_{max}. Here, the consensus assumption is used. Equation 1 defines de relative difference.

$$RD = \frac{|\hat{y}_i^1 - \hat{y}_i^2|}{y_{max}} \tag{1}$$

If the score is lower than a pre-defined threshold, the predictions are used to train the complementary regressor. Otherwise, the artificially labeled example is rejected. The consensus assumption is used in this step. If the example is labeled, this example is used to compute the mean error for each regressor. Next, the example is used for all regressors training. Here, the compatibility assumption is used since both models are trained with the same output. Algorithm 1 explains the training procedure of the proposed method.

Prediction is performed by combining the regressor predictions through prediction weighting. The weights are computed by inverting the values of the respective error produced by labeled examples in the training stage since the higher the error is, the less the artificial example benefits the model. In other words, this strategy gives more credit to the regressor that produces less errors. Algorithm 2 shows the steps of label prediction.

The Random AMRules regressor was employed to train the models and to produce the artificial labels for the unlabeled examples [16]. Random AMRules is a multi-target algorithm (predicts several outputs for the same example) that is based on rule learning which can be calibrated to work on single-target mode [3].

In essence, Random Rules is an ensemble based algorithm that uses bagging to create diversity and uses AMRules algorithm as a regressor. AMRules

Algorithm 1 Training algorithm of the proposed method

Initialization:
  $\alpha - Overlap\ percentage$
  $s - Score\ Threshold$
  Random input allocation and overlapping
  into the two groups using α
Input: *Example* $(\mathbf{x}_i, y_i) \in \mathcal{S}$
Output: *Updated Models*
Method:
Divide $\mathbf{x}_i$ *into* $\mathbf{x}_i^1$ *and* $\mathbf{x}_i^2$
if $(y_i = \emptyset)$ **then**
    $\hat{y}_i^1 = PredictModel1(\mathbf{x}_i^1)$
    $\hat{y}_i^2 = PredictModel2(\mathbf{x}_i^2)$
    if $(|\hat{y}_i^1 - \hat{y}_i^2|/y_{max} < s)$ **then**
        $TrainModel1((\mathbf{x}_i^1, \hat{y}_i^2))$
        $TrainModel2((\mathbf{x}_i^2, \hat{y}_i^1))$
else
    $\bar{e}_1 = Update\ the\ mean\ error\ of\ Model1(\hat{y}_i^1, y_i)$
    $\bar{e}_2 = Update\ the\ mean\ error\ of\ Model2(\hat{y}_i^2, y_i)$
    $TrainModel1((\mathbf{x}_i^1, y_i))$
    $TrainModel2((\mathbf{x}_i^2, y_i))$

Algorithm 2 Prediction algorithm of the proposed method

Input: *Example* $(\mathbf{x}_i, y_i) \in \mathcal{S}$
Output: *Example prediction* $\hat{y}_i$
Method:
Divide $\mathbf{x}_i$ *into* $\mathbf{x}_i^1$ *and* $\mathbf{x}_i^2$
$\hat{y}_i^1 = PredictModel1(\mathbf{x}_i^1)$
$\hat{y}_i^2 = PredictModel2(\mathbf{x}_i^2)$
$w_1 = \bar{e}_2/(\bar{e}_1 + \bar{e}_2))$
$w_2 = \bar{e}_1/(\bar{e}_1 + \bar{e}_2)$
$\hat{y}_i = w_1 * \hat{y}_i^1 + w_2 * \hat{y}_i^2$

divides the input space in order to train local model in each partition. AMRules partionates the input space and creates local models for each partition. The local models are trained using a single layer perceptron. Its main advantages are models simplicity, low computational cost and low error rates [3].

Modularity is one of the main advantages. In fact, this method allows the train of models for local input sections limited by the rule that are more precise. This algorithm also resorts to anomaly detection to avoid data outliers damage. Moreover, change detection on the stream is also employed by this method in order to avoid the influence of old information on the current predictions. The ensembles of AMRules can benefit the prediction by creating multiple and diverse regressor models by pruning the input partitions. The multiple regressor predictions create more possibilities to find a more accurate value. The ensembles also lead to a more stable final prediction and data change resilience.

The Self-training method basically consists of one learner that artificially labels the incoming example and uses directly in the training (without a rejection criteria). This method is very simple compared to Co-training, since only one model is trained and it doesnt present a condition for training acceptance. In terms of complexity, the Co-training presents the double complexity in required memory and computing power, when compared to the Self-Learning. In fact, the Co-training method basically trains two modesl while Self-training just trains one.

4 The Evaluation Method

The evaluation method and the material used in the experiments are described in this section. Real-world and artificial datasets were used to evaluate the proposed algorithm through a data stream simulation. A portion of 30% of the first examples of the stream were used for a initial consistent model training and the remaining 70% were used in the testing.

In order to produce an unlabeled examples in the test stage, a binary Bernoulli random process with a probability p was used to assign an example as labeled or unlabeled. In case of unlabeled assignment, the true output value is hidden from the algorithm. The p probabilities of unlabeled examples occurrence were 50%, 80%, 90%, 95% and 99%.

For the Co-training method, the score threshold values for algorithm calibration were 1×10^{-4}, 5×10^{-4}, 0.001, 0.005, 0.01, 0.05, 0.1, 0.5 and 1. These values of score threshold are justified by the possibility of algorithm behaviour observation in multiple scales of this parameter. The overlap percentages assume the following values: 0%, 10%, 30%, 50%, 70% and 90%. The evaluation was performed in Prequential mode where in example arrival, the label prediction is performed first and then the example is used in the training [25]. Each Random AMRules regressor consists of ten regressors ensemble. This value was determined in a validation step were no significant improvement was observed above 10 regressors.

In these experiments, five real world and four artificial datasets were used to simulate the data stream. The real world datasets were House8L (Housing Data Set), House16L (Housing Data Set), CASP (Physicochemical Properties of Protein Tertiary Structure Data Set), California, blogDataTrain and the artificial datasets were 2dplanes, fried, elevators and ailerons. These datasets contain a single-target regression problem and are available at UCI repository [26].

Table 2 an shows the features of the real world and artificial data sets used in the method evaluation.

The performance measure used in these experiments was the mean relative error (MRE). The MRE is used as an intermediate measure to quantify the prediction precision of each test scenario for both labeled and unlabeled examples. The MRE Reduction (MRER) was measured by using the relative difference (in percentage) between the reference scenario (no unlabeled examples used) MRE_0 and the case with the parametrization that lead to the lowest error MRE_{lowest}

Table 1. Real world datasets description

Dataset	# Examples	# Inputs
House8L	22784	8
House16H	22784	16
calHousing	20640	7
CASP	45730	9
blogDataTrain	52472	281

Table 2. Artificial datasets description

Dataset	# Examples	# Inputs
2dplanes	40768	10
fried	40768	10
ailerons	13750	41
elevators	8752	18

(includes the reference case MRE_0). Equation 2 defines the MRER performance measure.

$$MRER = \frac{|MRE_0 - MRE_{lowest}|}{MRE_0}.100 \quad (\%) \tag{2}$$

If the reference case yields the lowest error, then the MRER is zero, which means that the algorithm is not useful for that particular scenario.

Massive Online Analysis (MOA) platform was used to accommodate the proposed algorithm [27]. This platform contains Machine Learning and Data Mining algorithms for data streams processing and was developed in JAVA programming language.

5 Results

In this section, the evaluation results for Co-training and Self-training are presented and discussed.

5.1 Co-training results

For each combination of overlap percentage and score threshold, the experiments were performed in 10 runs due to the fact that the inputs are selected randomly. This procedure is important to obtain more consistent values. The results also include the presentation of the MRER for each dataset and the unlabeled examples percentage simulation.

In the experiments was registered, for the particular case of overlap of 50% and score threshold of 0.001, the use of 9.1% of the unlabeled examples in the training lead to reduction of 3.85% of the MRE in average for the House16H dataset. In general, it was also observed that the overlapping decrease the MRE.

The failure in some scenario is explained by the fact of many unlabeled examples lead to model degradation and the artificial labels were very inaccurate (the curves of unlabeled examples scenarios are above the reference curve). This fact indicates that features of the datasets such as inputs variables distributions may dictate the performance.

This methods are prone to error propagation through the model. The error propagation through the model lead to worst predictions in the artificial labeling. This effect leads to a cycle that reinforce the error on each unlabeled example processing. In fact, the more unlabeled examples arrive the higher is the error.

Table 3 provides the MRER values of the experiments on real world datasets for each chosen unlabeled examples probabilities, for Co-training method.

Table 3. MRER (%) for real world datasets

Datasets	Unlabeled examples probabilities				
	50%	80%	90%	95%	99%
House8L	2,23	3,21	2,77	0,00	0,00
House16H	3,85	1,93	0,32	0,00	0,00
calHousing	2,37	2,02	0,75	0,01	0,00
CASP	0,80	1,65	0,00	0,00	0,00
blogDataTrain	1,17	0,40	0,37	0,00	0,00

Table 3 suggests that the proposed algorithm seems to improve the performance for most part of the scenarios. The Co-training method can produce error reduction in higher percentage of unlabeled examples than the Self-training method. Despite this fact, the MRER are in general relatively small but superior than the Self-training method.

Table 6 provides the MRER value for real artificial datasets in similar way as the real world datasets presented in Table 5.

Table 4. MRER (%) for artificial datasets

Datasets	Unlabeled examples probabilities				
	50%	80%	90%	95%	99%
2dplanes	2,39	0,90	0,75	0,00	0,00
fried	3,55	3,35	1,71	0,00	0,00
ailerons	2,67	1,79	0,01	0,95	0,00
elevators	1,35	1,11	0,71	0,00	0,00

The results on artificial datasets reinforce the same conclusions that were obtained from real world datasets. The MRER is similarly small.

The results show that for 99% of unlabeled examples probability, the method does not produce beneficial artificial labels. This high level of unlabeled examples in the stream represents an extreme scenario where the model is training almost with artificially labeled examples and the high error propagation can frequently occur.

5.2 Self-training results

Table 5 provides the MRER values of the experiments on real world datasets for each chosen unlabeled examples probabilities, for the Self-training method.

When MRER assumes the zero value, a combination of overlap percentage and score threshold values that improves the model was not found and the reference scenario presents the lower MRE.

Table 5. MRER (%) for real world datasets

Datasets	Unlabeled examples probabilities				
	50%	80%	90%	95%	99%
House8L	0,57	0,01	0,00	0,00	0,00
House16H	0,23	0,00	0,00	0,00	0,00
calHousing	0,02	0,00	0,00	0,00	0,00
CASP	0,00	0,00	0,00	0,00	0,00
blogDataTrain	0,44	0,00	0,00	0,00	0,00

Table 5 suggests that the proposed algorithm seems to improve the performance for few scenarios. In fact, the algorithm fails in high probabilities of unlabeled examples. Inclusively, there is one dataset that didnt produce any favourable result. In successful cases, the MRER are in general relatively small.

Table 6 provides the MRER value for real artificial datasets in similar way as the real world datasets presented in Table 5.

Table 6. MRER (%) for artificial datasets

Datasets	Unlabeled examples probabilities				
	50%	80%	90%	95%	99%
2dplanes	1,12	0,00	0,00	0,00	0,00
fried	0,17	0,00	0,00	0,00	0,00
ailerons	0,81	0,00	0,00	0,00	0,00
elevators	0,22	0,00	0,00	0,00	0,00

The results on artificial datasets also support the view that the more elevated the unlabeled probability is, the less is the benefit of the unlabeled examples. The MRER is similarly small and there are very few successfull cases.

These results show that Self-training is limited by the percentage of unlabeled in the stream. For unlabeled examples higher than 50 %, the Self-training does not produce any error reduction. This limitation is explained by the fact that the artificially labeled examples produce high errors which does not garantee compability of the predictions.

6 Conclusion

This paper addresses a comparison of an online Co-training and Self-training algorithm for single-target regression based on ensembles of rule models. This work is the base for the development of multi-target regression methodology capable of using unlabeled examples information for model improving.

The results support that Co-training approach which uses the Random AMRules method reduces the error with the appropriate parameters calibration. The main contribution was the overlapping and the consensus measure strategies that contribute to increase diversity and model consistency in a online co-training scenario. The comparison between Co-training and Self-training reveal that Co-training can in fact lead to higher error reductions that the Self-training. In addiction, Co-training can produce error reduction in higher level of unlabeled examples in the stream.

In fact, the MRER is positive when an amount of unlabeled examples are used in the training in most evaluation combinations. Despite this fact, the model benefit is still relatively small and the performance is highly dependent of a good parametrization tuning (score threshold and overlap percentage). In addition, the amount of unlabeled examples is relatively small to obtain some model improvement.

Considering future work, this work will be extended to multi-target regression. The fact that very few unlabeled examples can lead to some improvement may suggest the study of the conditions that lead to this improvement. To increase the method validity, future works will include a higher number of real world datasets with higher amount of examples. Datasets with particular features such drifts presence are also in view.

7 Acknowledgements

This work is financed under the project "NORTE-01-0145-FEDER-000020" which was funded by the North Portugal Regional Operational Programme (NORTE 2020), under the PORTUGAL 2020 Partnership Agreement, and through the European Regional Development Fund (ERDF).

References

1. Adebiyi A. Ariyo, Adewumi O. Adewumi, and Charles K. Ayo. Stock price prediction using the arima model. In *Proceedings of the 2014 UKSim-AMSS 16th International Conference on Computer Modelling and Simulation*, UKSIM '14, pages 106–112, Washington, DC, USA, 2014. IEEE Computer Society.
2. Changsheng Li, Weishan Dong, Qingshan Liu, and Xin Zhang. MORES: online incremental multiple-output regression for data streams. *CoRR*, abs/1412.5732, 2014.
3. João Duarte and João Gama. Multi-Target Regression from High-Speed Data Streams with Adaptive Model Rules. In *IEEE conference on Data Science and Advanced Analytics*, 2015.
4. Zhi hua Zhou, Senior Member, and Ming Li. Semi-supervised regression with cotraining style algorithms. *IEEE Transactions on Knowledge and Data Engineering*, page 2007.
5. Nitesh V. Chawla and Grigoris Karakoulas. Learning from labeled and unlabeled data: An empirical study across techniques and domains. *J. Artif. Int. Res.*, 23(1):331–366, March 2005.
6. Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010.
7. Chuck Rosenberg, Martial Hebert, and Henry Schneiderman. Semi-supervised self-training of object detection models. In *Proceedings of the Seventh IEEE Workshops on Application of Computer Vision (WACV/MOTION'05) - Volume 1 - Volume 01*, WACV-MOTION '05, pages 29–36, Washington, DC, USA, 2005. IEEE Computer Society.
8. Zaid Chalabi Punam Mangtani Masahiro Hashizume Chisato Imai, Ben Armstrong. Article: Time series regression model for infectious disease and weather. *International Journal of Environment Research*, (142):319–327, June 2015.
9. Huseyin Sekerc Volkan Uslana. Article: Quantitative prediction of peptide binding afnity by using hybrid fuzzy support vector regression. *Applied Soft Computing*, (43):210–221, January 2016.
10. Jurica Levatić, Michelangelo Ceci, Dragi Kocev, and Sašo Džeroski. *Semi-supervised Learning for Multi-target Regression*, pages 3–18. Springer International Publishing, Cham, 2015.
11. Pilsung Kang, Dongil Kim, and Sungzoon Cho. Semi-supervised support vector regression based on self-training with label uncertainty: An application to virtual metrology in semiconductor manufacturing. *Expert Syst. Appl.*, 51:85–106, 2016.
12. Andrew B. Goldberg, Xiaojin Zhu, Alex Furger, and Jun-Ming Xu. Oasis: Online active semi-supervised learning. In *Proceedings of the Twenty-Fifth AAAI Conference on Artificial Intelligence*, AAAI'11, pages 362–367. AAAI Press, 2011.
13. Ian H. Witten, Eibe Frank, and Mark A. Hall. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 3rd edition, 2011.
14. Amparo Albalate and Wolfgang Minker. *Semi-Supervised Classification Using Prior Word Clustering*, pages 91–125. John WileySons, Inc., 2013.
15. Ezilda Almeida, Petr Kosina, and João Gama. Random rules from data streams. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, SAC '13, pages 813–814, New York, NY, USA, 2013. ACM.
16. João Duarte and João Gama. Ensembles of adaptive model rules from high-speed data streams. In *Proceedings of the 3rd International Conference on Big Data,*

Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications - Volume 36, BIGMINE'14, pages 198–213. JMLR.org, 2014.
17. Monica Bianchini, Marco Maggini, and Lakhmi C. Jain. *Handbook on Neural Information Processing*. Springer Publishing Company, Incorporated, 2013.
18. Chang Xu, Dacheng Tao, and Chao Xu. A survey on multi-view learning. *CoRR*, abs/1304.5634, 2013.
19. Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, COLT' 98, pages 92–100, New York, NY, USA, 1998. ACM.
20. Steven P. Abney. Bootstrapping. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA.*, pages 360–367, 2002.
21. Sally Goldman and Yan Zhou. Enhancing Supervised Learning with Unlabeled Data. In *Proc. 17th International Conf. on Machine Learning*, pages 327–334. Morgan Kaufmann, San Francisco, CA, 2000.
22. Mohamed Farouk Abdel Hady, Friedhelm Schwenker, and Günther Palm. Semi-supervised learning for regression with co-training by committee. In *Proceedings of the 19th International Conference on Artificial Neural Networks: Part I*, ICANN '09, pages 121–130, Berlin, Heidelberg, 2009. Springer-Verlag.
23. Ulf Brefeld, Thomas Gärtner, Tobias Scheffer, and Stefan Wrobel. Efficient co-regularised least squares regression. In *Proceedings of the 23rd international conference on Machine learning*, ICML '06, pages 137–144, New York, NY, USA, 2006. ACM.
24. Zhi-Hua Zhou and Ming Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Trans. on Knowl. and Data Eng.*, 17(11):1529–1541, November 2005.
25. João Gama, Raquel Sebastião, and Pedro Pereira Rodrigues. On evaluating stream learning algorithms. *Machine Learning*, 90(3):317–346, 2013.
26. K. Bache and M. Lichman. UCI machine learning repository, 2013.
27. Albert Bifet, Geoff Holmes, Richard Kirkby, and Bernhard Pfahringer. Moa: Massive online analysis. *J. Mach. Learn. Res.*, 11:1601–1604, August 2010.