Imbalanced cost in regression problems

Constanze Bayer

Matrikelnummer 515326

Humboldt-Universität zu Berlin

Wirtschaftswissenschaftliche Fakultät

Fachbereich Wirtschaftsinformatik

Abstract

Economic applications of regression methods often involve asymmetric cost for different kind of prediction errors. Underpredicting is costlier than overprediction in most contexts. In standard regression performance measurements, this is not considered. Besides re-sampling and cost-sensitive predictors, a third approach to handle this is explained and evaluated: the average misprediction cost. These are optimized by using an adjustment added post-hoc to the base regression outcome. In this paper, the algorithm has been used on a bank data set from Europe. It can be shown that costs associated with misprediction can be reduced significantly without changing the regression method and data. Also the best regression method can be chosen – based on cost and not statistical conventions.

Keywords

Asymmetric cost, Cost-sensitive regression, Imbalanced cost, Post-hoc tuning

Introduction

Regressions are used in practical contexts very often for prediction, may it be the future selling price for a house, a stock exchange price or a loan default. But standard performance measures assume symmetric loss functions concerning prediction errors. But in many practical economic contexts, asymmetric costs arise: It makes a difference for a real estate agent whether he predicts the selling price for a house too low or too high, banks need to predict as accurate as possible how much loss reserves they have to keep considering the amount of credit (and risk) they give to customers. Is the estimation to high and too much capital is kept, this means that investing opportunities are wasted. This is annoying – but it is much worse for the financial institution if the prediction is too low: The reserves will not suffice. Legal guidelines cannot be followed, this as well as a severe crisis can even jeopardize the financial institution's future. Consequences are exemplified in Zhao, Sinha & Bansal (2011).

To meet the problem of asymmetric or imbalanced cost, two main approaches have been made: either re-sampling the data or incorporating the asymmetry in the regression. This paper will evaluate a method presented by Bansal, Sinha & Zhao (2008). They are the first to use a post-hoc tuning approach. For this, a new measurement was introduced: the average misprediction cost. By adding an adjustment measure to the outcome of a standard (cost-insensitive) regression method, the costs associated with errors in the prediction are minimised. The authors test their algorithm on a data set from the US. I want to replicate their findings with another data set from the same problem domain: expected loan losses. By this, the validity of the so called cost-sensitive regression will be strengthened.

This paper proceeds as follows: In the next section, a brief literature review will be given to summarize findings in the field of regression tuning. Then the methodology will be explained in detail, followed by a section on the data set used. Thereafter, the empirical results will be summarized, followed by a discussion. The last section concludes and names fields of possible future research.

Literature review

In the field of prediction two major tasks occur: On the one side there is classification. In the problem domain of credit and banking, this would be to forecast whether a new customer applying for a loan is creditworthy or not. On the other hand, there is regression: By how much will a customer default or how much earnings can be expected from a given loan?

Imbalanced costs occur in both tasks: It is much costlier for a bank to give a loan to bad customer then to decline a loan to a good customer. In regression tasks, the numerical value of the default would be of interest. Here the costs for the bank are asymmetric as well: Underpredicting, that is estimating a smaller default than realized, is costlier to a financial institution than overprediction.

Much of the research on imbalanced cost has been done on classification. As this paper focusses on regression problems, I refer f.e. to Brown & Mues (2012) for more detailed discussion of possible strategies for classification tasks.

Two main approaches have been made to incorporate the asymmetry of cost in the regression process: The first is to re-sample the data. This can be done both for classification and regression tasks. Branco, Torgo & Ribeiro (2015) summarize several techniques to use this. In general, the special cases of interests will be oversampled, or the number of "normal" cases will be decreased. I will not discuss these approaches in detail as this paper focusses on the aspect of imbalanced cost and not the imbalanced data distribution.

The second approach is to use asymmetric loss functions in the estimating process. In statistical theory, loss functions are symmetric. Applied to a practical domain, this would mean that the cost of an error depends only on the magnitude. But reality is complex: In practical economic contexts asymmetric loss function were acknowledged in several domains: May it be house selling prices, credit loan charge offs or stock markets.

Several authors tried to incorporate this by deriving estimators for asymmetric cost functions[1]. Based on their shape on both sides of 0, they are named Lin-Lin (both sides linear), Lin-Ex (linear-exponentially) and Quad-Quad (both quadratic). All of these functions have in common that the slope on "one side" of the function is steeper than on the other side. This is due to the fact that underprediction is usually more expensive than overprediction which is taken into account by several authors (f.es. Granger 1969, Varian 1974).

Granger (1969) offers the first findings about non-symmetric cost functions. He finds that by adding a bias factor to the prediction using a linear non-symmetric cost function, one can approximate any non-symmetric cost function very good.

Very widely used was the work of Varian (1975). He derived an optimal estimator for the so called Lin-Ex loss function. The established function is approximately linear on the side of

[1] The words cost function and loss function are both used in the literature and mean basically the same.

underprediction and approximately exponential on the overprediction side. Zellner (1986) proofs that given the precision of this incorporation of asymmetry makes its use favourable against other measures in practical problems. Cain & Janssen (1995) apply these findings on the problem domain of real estate market and the prediction of selling prices. Using linear and quadratic loss function as well as a Lin-Ex loss function, they computed adjustment measures. In their basic papers, Christoffersen & Diebold (1994, 1996) followed two different approaches: Their goal was to find an optimal predictor for a general loss function that is not further specified. First (1994) they presented a method to find an approximate predictor for an exact loss function. Then (1996) they turned things upside down - presenting an accurate predictor for a loss function that is approximated. The approximation is done by a piecewise linear loss function.

These early insights to asymmetric loss functions have been used in the data mining literature to incorporate asymmetry in the training of new computational possibilities: Crone, Lessmann & Stahlbock (2005) address the issue of asymmetric costs in regression problem by training a neural network directly with an asymmetric cost function. In their paper, they use a Lin-Lin cost function. They find that concerning time series analysis in inventory management, their method of training the NN on an asymmetric linear loss function outperforms the standard approach to use statistical error measures.

Czajkowski, Czerwonka & Kretowski, (2015) use the target variable average misprediction cost as developed by Bansal et al. (2008). By this, they want to make the Global Model Tree as explained in Hedderich & Sachs (2018) cost-sensitive: Their concept implements the cost-sensitivity in the fitness function that evaluates a population of trees and genetic operators. The authors emphasize the importance of their findings as financial institutions can reduce their costs significantly using the authors method.

The already mentioned authors Bansal et al. (2008) try to develop a third approach to incorporate imbalanced cost besides re-sampling and changing the regression: They present a post-learning adjustment to the cost-insensitive prediction. The authors claim that they are the first to suggest such an algorithm that changes the regression outcome. In their paper, they use an asymmetric linear cost function. In a second paper, they develop their model further (Zhao et al., 2011). to include more complex loss functions. This is advantageous in contexts where cost ratios (under- vs. overestimation) might change but not the underlying pattern. Furthermore, adjusting the output is a relatively simple to implement compared to altering regression methods to increment asymmetric cost functions. It is easy to be built up on top on existing models. This may be of advantage especially in a practical context.

Besides these methods, Torgo & Ribeiro (2007) provide a new approach to implement cost-sensitivity into the analysis. Instead of focussing only on one type of cost (costs of over- and underprediction), they introduce the relevance function that can involve several cost types. This takes into account, that in many data sets, not only over - and underprediction is a problem, but that even a bigger error might be less of a problem if it occurs on "one side" of the prediction. In terms of loan charge off forecasting translated: It is less problematic if a system predicts a bigger loss than realized but it is a problem if it predicts a gain. The absolute error might be the same. But the second scenario might be far more relevant to the bank. So besides the loss function, a relevance function comes into consideration, especially to predict extreme values. For this, the authors introduce the concept of utility, as a balance of benefits from a prediction and associated costs. Depending on the problem domain, more emphasize may be given to the one or the other. In later papers, the authors established new metrics based on this concept of utility to evaluate various regression models, depending on the practical needs and context.

Methodology

Background

Due to the asymmetric cost function of misprediction errors, statistical error measures are inappropriate to evaluate the outcome of a regression in a cost-sensitive context. Mean Square Error and Absolute Error assume symmetry in the loss function. But in many business contexts, such as banking and loan default forecasting, the error costs are different for underprediction and overprediction. It is costlier to become short on loan loss reserve - it might risk the well-being of the bank itself. A bank could become bankrupt and/ or violate loan reserve requirements made by the financial authorities. Overprediction is costly mostly in terms of opportunity costs of declined loans and less investment opportunities. Nevertheless, regression methods are used quite commonly, also in banking, as they are easy to understand and wide spread. For Bansal et al. (2008), this is the starting point: Instead of pre-tuning the data or the regression method, they build the adjustment on top. This could be advantageous in a practical application as you do not have to redesign the whole prediction process.

Methodological approach

Bansal et al. (2008) introduce a new performance measure corresponding to the average misclassification cost in classification problems: the average misprediction cost (AMC). The outcome of a standard regression method is adjusted: the numerical predicted values increased or decreased, depending on the direction of error, over- or underprediction. The authors name this procedure "Cost Sensitive Regression".

Bansal et al. (2008) implement the adjustment with the three base regression methods: Linear Regression (LR), M5 Model tree (M5) and back propagation neural network (NN).

Linear regression is used to quantify a correlation between a target variable and their explanatory variables. The predicted value is thereby expressed by a linear combination of weighted attributes. The weights are computed mostly to minimize the sum of the squared differences as explained in Witten, Frank, & Hall (2011).

Model trees are in general a normal decision tree that delivers a linear regression function at a node instead of a value. They are used (together with regression trees) to predict continuous numeric variables. Most model tree algorithms minimize some sort of variation within the leaf. The M5 model tree method uses the standard deviation to maximize the error reduction as a splitting criterion. Again, further explanations can found in Witten et al. (2011).

The basics of neural networks working on back propagation is explained in Runkler (2015)[2]. Neural networks consist of a starting node and an end node, with a number of hidden layers in between. The learning process consists of two phases. The forward-propagating phase, after random initialisation of the model, takes the input data and calculates the expected output through the initialized regression function in the hidden layers. The loss function then computes the loss of precision by replacing the actual by the predicted value. In the second phase, the so called back-propagation, the way is gone back from the output through the neural network using the derivative of the function in the layer. By stepwise adjusting the hidden layer function, the network learns the weights to best fit the predicted values to the input.

The algorithm developed by Bansal et al. (2008) takes the outcome of the regression, that is the prediction for the target value of the test set, as the function f that is adjusted by δ (lower case delta). This will produce the adjusted function f' as in equation (1).

(1) $f' = f + \delta$

δ will be calculated from the training set and used on the test set.

[2] A very comprehensive explanation is written down here: https://medium.com/datathings/neural-networks-and-backpropagation-explained-in-a-simple-way-f540a3611f5e

The AMC is declared by θ (lower case theta). θ will be evaluated by the test set. The AMC depends on the error e made, that is in general

(2) $e = f-y$

with y being the actual value of the target variable in the test set. This error is standardized: In the paper the authors use the Aggregate gross book value of total loans. This is made to make sure that a single big misprediction do not bias the outcome.

Depending on the direction of the standardized error, this error will produce costs, represented by a cost function C(e). This cost function is asymmetric as costs do not only depend on magnitude but also on direction of error. This represented in a simple cost function with different cost ratios. Underprediction will be "punished" by a factor f.e. 20. This means that c+ equals the error (factor 1), c- is punished by 20*e (or other factors). For zero error, the function is zero. A graphical example of this simple Lin-lin cost function can be seen in Figure 1.

Bansal et al. (2008) consider in their paper only losses, that is negative values for the actual outcome. Doing the same on my data set would lead to shrinking the data set size to 2305 instances. Furthermore, not every actual error is predicted as one: Sometimes even a predicted gain turns out to be a loss. To take this into account, I had to adjust the error calculation. Simple measures like error (e) = predicted (p) – actual (a) (like in equation (2)) did not work as over- and underpredicted errors might both be smaller than zero, depending on the values. It follows that:

Overprediction occurred if $p>0$ and $p>a$ as well as $p<0$ and $p<a$.

Underprediction occurred accordingly if $p>0$ and $p<a$ as well as $p<0$ and $p>a$.

The absolute values of error have been assigned to the left for underprediction by multiplying the absolute value with -1. The result of this error calculation can be seen in Figure 1 where underprediction is punished far more than overprediction.

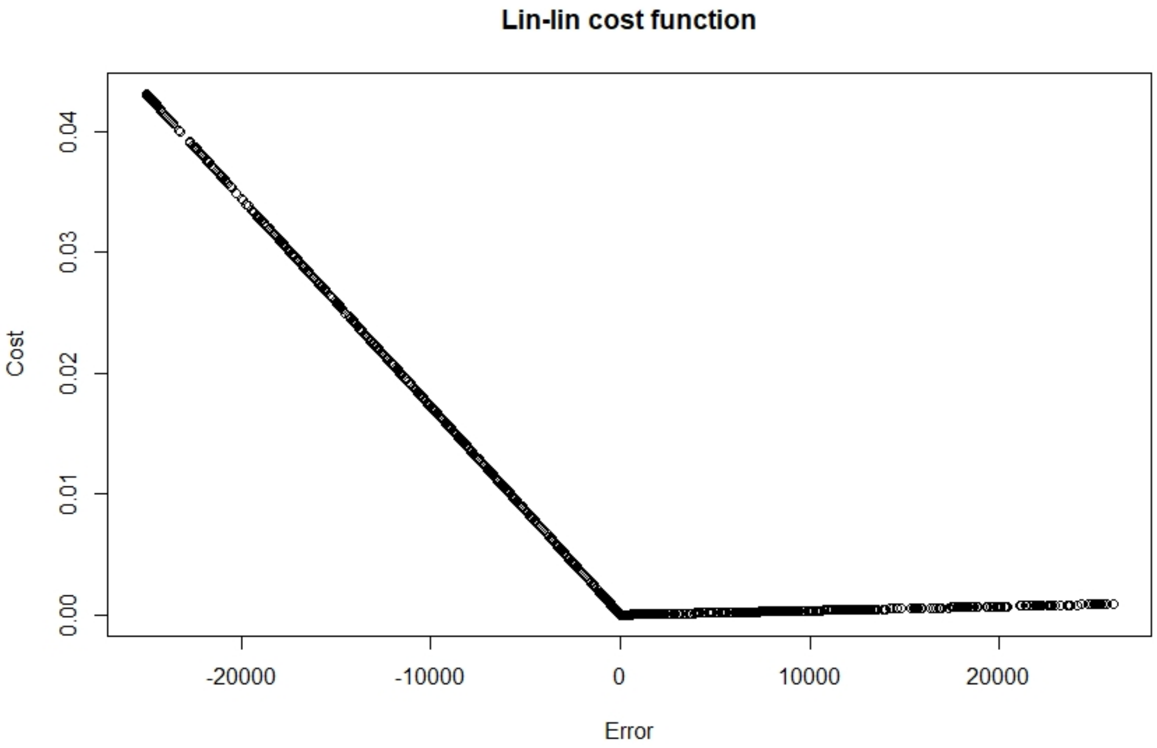


Figure 1 Lin-lin cost function based on LR prediction, cost ratio 1:50

After the calculation of errors, the cost for each data point is computed. The mean of these costs of the normal regression would be the average misprediction cost (AMC) with zero adjustment (δ=0).

(3) $\theta(\delta) \ = 1/N \sum_{i=1}^{N} C(f(xi) + \delta - yi)$

Then the task is to add a δ different from 0. The AMC for an adjusted f' would be minimal and certainly smaller than for the original regression outcome. Equation (3) will be optimized. The range of δ would be in between zero adjustment and such an amount until all predictions become overpredictions.

Important part of the methodology is the search for the best δ that minimises the average misprediction cost. The authors propose a so called hill-climbing algorithm. This is similar to a hiker on a foggy mountain that cannot see the path to the top. He moves forwards, backwards or sideways and moves wherever it is going uphill. If he cannot move up anywhere he must be on the top. The search works similar[3]: From a starting point, δ is increased (decreased) in small steps and costs are compared. The adjustment with the lower AMC is kept. The size of the steps in one direction is always increased to speed up the procedure. When the next AMC are bigger than the previous ones, the steps are made smaller again and the search starts again, but not from the original point but the last smaller AMC producing δ. The search ends when there is no chance to find a better δ, the adjusted f' is returned as well as the optimizing δ.

In the paper by Bansal et al. (2008), the algorithm is coded as a function that takes the regression method, the training data set, the cost function and a preferred precision of adjustment above 0.

My code works as follows: The data set is loaded. The predicted and the actual data are defined. This step will be done three times in the end, each for every regression method LR, NN, M5. Errors are calculated, as the absolute error divided by the sum of loans. This variable is not available in the given data set. As an approximation, I use the total sum of profits, losses in their absolute value. It is possible (and realistic) that the original value of the loan was bigger.

The cost function is defined. This is done very roughly as it depends on the domain. Here, I will go with Bansal et al. (2008) and define a general cost ratio as function. Underprediction will be punished by the factor 10, 20, 50 and 100. Symmetric costs, that is a punishment factor of 1 is included for completeness. Costs for every data point are calculated. The function for the AMC is defined as the mean of those costs.

In the following section, the hill climbing is described in more detail. It was also coded but not used as existing computing power did not suffice. Instead, a standard optimization was used, under the assumption that the AMC function can be differentiated twice.

The hill climbing start with the AMCs for the case without adjustment. This serves as benchmark. Then, the search for the minimizing δ starts. The minimum precision of adjustment is defined, p >0. Bansal et al. (2008) use p=0.01 as the first δ. The adjusted and the unadjusted AMC are compared. The outcome defines the direction of movement as it can be seen in Figure 2: +1 means we are moving towards the optimum from the left, -1 from the right. 0 means no movement.

[3] It is possible that the found maximum (or minimum) is a local one. But Bansal et al. (2008) assume that the cost function is convex and therefore θ(δ) is convex. Every found local minimum must be a global one.

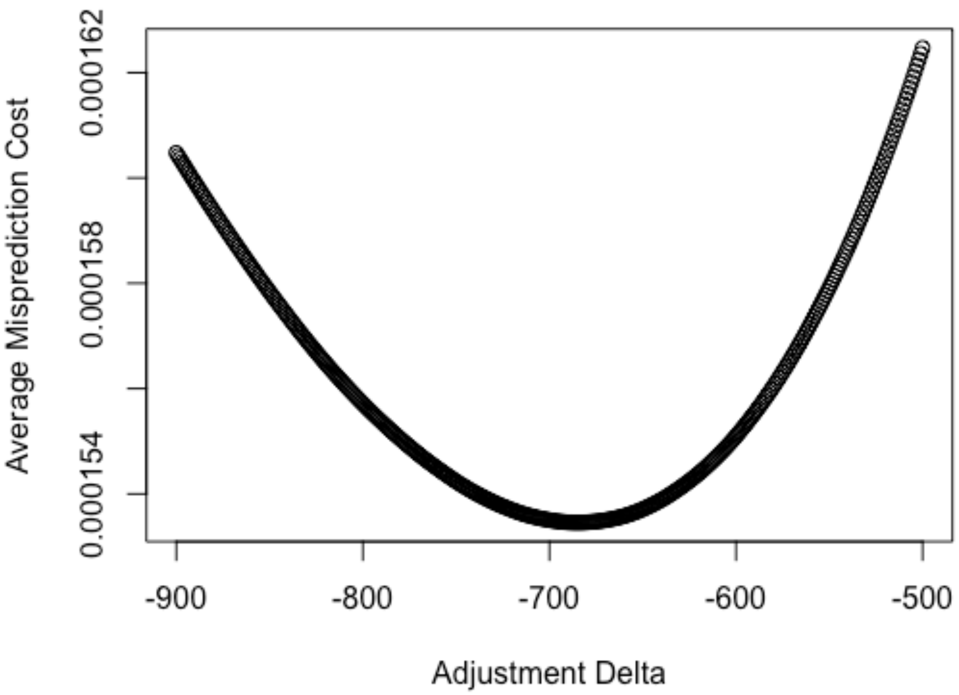


Figure 2 Average Misprediction Cost as a function of the Amount of Adjustment, based on the NN regression, cost ratio 1:50

Then the search itself starts. Zero adjustment is again the starting point, delta previous. Delta is set to the previous delta.

In a loop, the search steps are done. The size of the step, the stride s, was set to 1 and always doubled in the next execution of the loop. Delta is increased (or decreased if the direction of improvement is -1) by the product of s and p in the direction given. As s is doubled for every loop, the amount of δ rises very fast. This increase of δ is done until the AMC of the new δ is bigger than the AMC of the old δ. Then the old δ is kept and search starts from there again in smaller steps. The authors defined that the breaking point for this search is s<=2 which means no further improvement can be achieved. This means that within one loop execution no improvement achieved, another step of the size of p would already be too much. As a result, a δ* is returned which can be added to the actual values and returns f'*.

Implementation

For this paper, I separated regression and adjustment. Therefor, the regression results will be loaded only. I used RapidMiner Version 8.2 to perform the base regression, extended with the Weka Toolkit available via RapidMiner Marketplace. I kept the default settings for the regression methods. They are the same as in the Weka toolkit that was used by Bansal et al. (2008). For the adjustment, I used R, version 3.5 with the stats package.

Data

I use the "Give me some credit" data set of the 2011 Kaggle competition. It shares information about a bank, presumably seated in Europe, and their customers credit behaviour. The bank's exact location and name is not shared as well as time when the data has been collected. All together, there are 37,638 data points. In table 1, the data variables will be explained briefly as provided by Kaggle.

Variable Name	Description	Type
BAD	No Description by Kaggle, Status of Customer: GOOD/ BAD	BAD/ GOOD
RevolvingUtilizationOfUnsecuredLines	Total balance on credit cards and personal lines of credit except real estate and no installment debt like car loans divided by the sum of credit limits	percentage
age	Age of borrower in years	integer
NumberOfTime30-59DaysPastDueNotWorse	Number of times borrower has been 30-59 days past due but no worse in the last 2 years.	integer
DebtRatio	Monthly debt payments, alimony,living costs divided by monthy gross income	percentage
MonthlyIncome	Monthly income	real
NumberOfOpenCreditLinesAndLoans	Number of Open loans (installment like car loan or mortgage) and Lines of credit (e.g. credit cards)	integer
NumberRealEstateLoansOrLines	Number of mortgage and real estate loans including home equity lines of credit	integer
NumberOfDependents	Number of dependents in family excluding themselves (spouse, children etc.)	integer
PROFIT	No Description by Kaggle, real values of negative or positive profit made from loans	real

Table 1 Variable names and definitions

There are no descriptions for BAD and PROFIT. These are my assumptions looking at descriptive statistics: PROFIT has a minimum value of -25,000. More than ten percent of the negative values in PROFIT have this level. It seems reasonable that this is used as a default value if the complete or big part of the loan defaults. The positive values for PROFIT range between 545 and 1023.8 and represent most likely profits from loan giving from interest payments. The median of PROFIT and the median of only strictly positive PROFIT are similar with 852.02 and 866.6 respectively. As measuring unit, I assume Euros.

BAD is treated as a label for risk evaluation: Every customer that defaults his loan will be given the status BAD. Every client with associated positive profit will receive GOOD.

For the regression, I used "PROFIT" as a target variable. As it is highly correlated with the variable "BAD", "BAD" will not be used for the regression. The status BAD or GOOD will be given after someone pays back his loan or defaults. It is therefore reasonable to assume that it will not be available for new data on a new client.

Concerning the target variable, the data set is highly imbalanced: Only 2,539 out of the 37,638 entries (6.75 %) are labelled as BAD which goes along with negative or zero PROFIT.

None of the data has missing values. Except for BAD and PROFIT, all variables are standardized (given in the data set).

The data set has been split in a training and test data set. The division factor was 0.5 so that each subset consists of 17819 data points. This factor has been chosen as Bansal et al. (2008) use the data collected from one quarter to predict the next and from one year (four quarters) to predict the next four. Accordingly, I use one half of the data to predict the PROFIT in the second half.

Empirical results

Table 2 presents the results of the tuned and untuned LR, NN and M5 models. The cost figures show the mean costs for each regression method and tuning (with or without) combination. The means were computed by averaging the cost across all five cost ratios used (1:1, 1:10, 1:20, 1:50 and 1:100). NN turned out to be the best performer in all scenarios, tuned or untuned. In every scenario, using whichever base regression method, the average costs were decreased.

	Costs without tuning		**Costs with tuning**	
Method	**Mean**	**SD****	**Mean**	**SD**
LR	98.69	106.67	69.49	75.17
NN	31.41	31.82	11.98	9.15
M5	80.01	85.84	61.04	64.44
*Cost values are on a scale of 10^-5				
**SD = Standard Deviation				

Table 2 Effects of tuning across all cost ratios

To test for significance, a 3x2 factorial analysis of variance (ANOVA) was made. AMC was used as a dependent variable and the method (with three levels) and tuning (with two levels) as factors. Both the main effects were significant at a p= 1% or 5% level: method p =0.00769, tuning p=0.02122. Interaction effects were not computed due to missing degrees of freedom.

Pairwise comparisons were made using the Tukey's Honest Significance Test. Concerning the method, differences between NN-LR (p=0.0078) and NN-M5 (p=0.0126) were significant at the p=1% or 5% level. Only M5-LR was not (p=0.14). The tuned models performed significantly better than the untuned ones at the 5% level (p=0.0211).

The data for table 2 were aggregated from the values in table 3: Here costs are shown for each cost ratio using a base regression method and a tuning condition. Different from Bansal et al. (2008), I computed a tuned and untuned value for cost ratio 1:1. There is no reason why the tuning should not be useful for a symmetric instead of asymmetric cost function.

For all cost ratios, costs go down. NN performs best for all combinations of cost ratio and tuning mode. For M5 and LR, it depends on the cost ratio. The higher the punishment for underprediction, the better M5 performs. To test for significance of these results, ANOVA tests were made. Again, no interaction effects could be tested. Except for the 1:1 cost ratio, all effects are significant as can be seen in table 4.

	Cost Ratio									
	1:1		10:1		20:1		50:1		100:1	
Method	Tuned	Untuned	Tuned	Untuned	Tuned	Untuned	Tuned	Untuned	Tuned	Untuned
LR	3.64*	5.14	20.5	29.07	39.1	55.67	95.2	135.42	189	268.14
NN	3.34	3.51	6.26	10.65	8.64	18.57	15.35	42.35	26.31	81.98
M5	4.4	4.76	19.03	24.00	35.16	45.38	83.25	109.52	163.38	216.41

*Cost values are on a scale of 10^-5

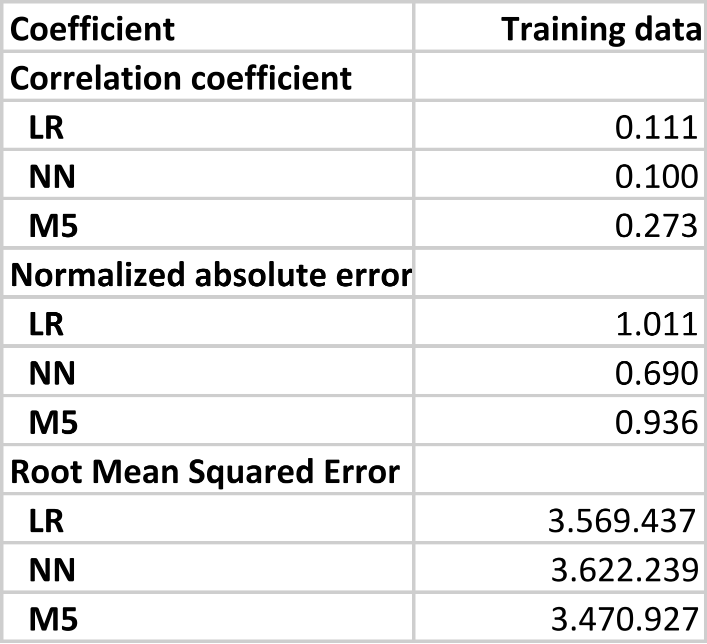
Table 3 Effects of tuning on costs

	P-Values	
Cost Ratio	**Tuning**	**Method**
1:1	0.245	0.252
1:10	0.0447	0.0169
1:20	0.0299	0.0110
1:50	0.0299	0.0110
1:100	0.01712	0.00612

Table 4 P-Values for different cost ratios

Again, the higher the cost ratio, the higher the significance of cost reduction is.

These results are compared to the cost-insensitive performance measures for the base regression methods as reported by RapidMiner. They are summarized in table 5.



Coefficient	**Training data**
Correlation coefficient	
LR	0.111
NN	0.100
M5	0.273
Normalized absolute error	
LR	1.011
NN	0.690
M5	0.936
Root Mean Squared Error	
LR	3.569.437
NN	3.622.239
M5	3.470.927

Table 5 Conventional performance measures for base regression methods

It is important to keep in mind, that these performance measures do not incorporate cost asymmetry. Furthermore, the picture is not homogenous. In respect to the correlation coefficient, NN performs worst and M5 best. In respect to Normalized absolute error NN is the best chosen method, M5 is second and LR worst. The Root Mean Squared Error makes M5 best.

Discussion

The empirical results show that a significant reduction of average misprediction cost could be achieved by using the adjustment method proposed by Bansal et al. (2008). For every base regression method (LR, NN, M5) used, a cost reduction could be found. In this particular case, NN would be most useful regression method combined with tuning as it caused the lowest overall misprediction cost. As it performed best for two cost ratios, M5 would be the second choice ahead of LR. This is in contrast to the findings of Bansal et al. (2008): They found the best effects for M5, followed by NN and LR being the worst method. So overall, I can confirm the overall cost reduction achieved by the algorithm. The underlying regression method has to be chosen according to the problem domain and the available data sets. Focussing only on losses (in this case, negative profits) might change the picture.

Compared to standard performance measures, the results are not as ambiguous as in Bansal et al. (2008). Considering all the three standard measures, M5 would most probably be the best choice as it performs best on average. This method performs best in two out three measurements. Bansal et al. (2008) found a different "winner" for every coefficient. As one can see from table 2 (using the 1:1 cost ratio as these measures do not consider asymmetry in cost), this would lead to higher costs on average. This is proof to the claim of the authors that statistical measurements are of limited use in economic problem domains and application.

Reasons for these differences to the findings of Bansal et al. (2008) are most likely to be found in the data. As I do not know details about the time span covered in the data, it is possible that older data mislead the regression. In a practical context this will not be problematic, as meta data are available. The best fitting regression method and cost ratio can be chosen. I can emphasize the necessity to careful choose both depending on the context.

Incorporating further costs, such as computing time and amount for bigger data sets, might change the picture again: Bansal et al. (2008) mention that M5 is less computing intensive than NN. This might be an interesting follow up research.

Conclusions

In this paper, I evaluated a metric established by Bansal et al. (2008) to incorporate asymmetry in a cost function. The authors propose the new measure: average misprediction cost to evaluate the outcome of a regression to predict an economic variable. By this, they want to replace statistical standard measures – at least in practical application contexts. For this, the outcome of a base regression, linear regression (LR), neural networks (NN) and M5 model tree (M5), was optimized by an adjustment of δ.

My findings confirm that this metric yields a significant reduction in the average costs associated with misprediction. NN performed as the best base regression, compared to M5 model tree and LR, linear regression. Those regressions stayed untouched which makes the metric easy to understand and to build "on top" of an existing prediction process. This might make the algorithm easy to use in practical contexts.

References

Bansal, G., Sinha, A., Zhao, H. (2008): Tuning Data Mining Methods for Cost-Sensitive Regression. A Study in Loan Charge-Off Forecasting. *J. Manage. Inf. Syst.* 25, 3. S. 315-336.

Branco, P., Torgo, L., Ribeiro, R. P. (2015): A Survey of Predictive Modelling under Imbalanced Distributions. *CoRR, abs/1505.01658*.

Brown, I, Mues, C. (2012): An experimental comparison of classification algorithms for imbalanced credit scoring data sets. *Expert Systems with Applications (39, 3)*. S. 3446-3453.

Cain, M. and Janssen, C. (1995): Real estate price prediction under asymmetric loss. *Annals of the Institute of Statistical Mathematics, 47(3)*. S. 401–414.

Christoffersen, P.F. & Diebold, F.X. (1994): Optimal Prediction under Asymmetric Loss. *National Bureau of Economic Research Technical Working paper, 167*, Cambridge, Massachusetts.

Christoffersen, P.F. & Diebold, F.X. (1996): Further results on forecasting and model selection under asymmetric loss. *Journal of Applied Econometrics, 11*, S. 561–572.

Crone, S. F., Lessmann, S., Stahlbock, R. (2005): Utility based data mining for time series analysis. Cost-sensitive learning for neural network predictors. *Proceedings of the 1st* *international workshop on Utility-based data mining* (UBDM '05). ACM, New York, NY, USA, S. 59-68.

Czajkowski, M., Czerwonka, M., Kretowski, M. (2015): Cost-sensitive Global Model Trees applied to loan charge-off forecasting. *Decision Support Systems, Volume 74 (C),* S. 57-66.

Granger, C. (1969): Prediction with a Generalized Cost of Error Function. *OR, 20*(2), S. 199-207.

Hedderich, J., Sachs, L. (2018): *Angewandte Statistik. Methodensammlung mit R* (16., überarbeitete und erweiterte Auflage). Berlin: Springer Spektrum.

Quinlan, J.R. (1992): Learning with Continuous Classes. *Proceedings of Australian Joint Conference on Artificial Intelligence*, Hobart 16-18 November 1992, 343-348.

Runkler, T. (2015): *Data Mining. Modelle und Algorithmen intelligenter Datenanalyse* (2. aktualisierte Auflage). Wiesbaden: Springer Vieweg.

Torgo L., Ribeiro R. (2007): Utility-Based Regression. In: Kok J.N., Koronacki J., Lopez de Mantaras R., Matwin S., Mladenič D., Skowron A. (eds): *Knowledge Discovery in Databases: PKDD 2007. 11th European Conference on Principles and Practice of Knowledge Discovery in Databases, Warsaw, Poland, September 17-21, 2007, Proceedings. Lecture Notes in Computer Science, vol 4702.* S. 597-604. Berlin, Heidelberg: Springer.

Varian, H. (1974): A Bayesian approach to real estate assessment. In: Feinberg, S.E. & Zellner, A. (eds.), *Studies in Bayesian Econometrics and Statistics in Honor of LJ. Savage*, S. 195–208. Amsterdam: North-Holland.

Witten, I. H., Frank, E., & Hall, M. A. (2011): *Data mining. Practical machine learning tools and techniques*. Burlington, MA: Morgan Kaufmann.

Zhao, H., Sinha, A.P., Bansal, G. (2011): An extended tuning method for cost-sensitive regression and forecasting. *Decision Support Systems, Volume 51(3)*, S. 372-383.

Acknowledgement

I want to thank Huimin Zhao for providing the Java source code for the extended tuning method for cost sensitive regression. This was a great starting point and reference for my own coding.