Avalanche Forecasting: An Ordinal Regression Approach

David Howard Neill Nathan Simonis Ludvig Wärnberg Gerdin

***Abstract*—This study compares three different approaches to the prediction of an ordinal response variable in avalanche risk modelling. The best model is a neural network where the problem is transformed into multi-label classification. It has an estimated MAMSE of 0.182 and would be a valuable tool for forecasters.**

I. Introduction

The number of fatal accidents caused by snow avalanches has been rising in alpine countries and North America for the last 20 years (Choubin et al., 2019). In the period from October 2019 to 2020, 64 people lost their lives in Europe due to avalanches (European Avalanche Warning Services, 2020). Besides, higher average temperatures due to global warming are increasing the frequency and intensity of avalanches (Ballesteros-Cánovas et al., 2018). It is therefore essential that avalanche hazards can be accurately predicted to best protect the populations surrounding the at-risk areas.

Avalanche forecasting is commonly conducted by local, trained observers who specify regional avalanche risk based on an ordinal danger level scale (See European scale European Avalanche Warning Services, 2020). Like other instances of human forecasting, the operations are time-consuming and not economically efficient. Thus, previous research has explored the use of advanced statistical methods for avalanche forecasting.

For instance, Choubin et al. (2019) used Support Vector Machines (SVM) and Multiple Discriminant Analysis (MDA) approaches to predict a hazard risk defined as ”low”, ”medium” and ”high”. They scored well on both methods with an area under the receiver operating characteristics curve (AUC-ROC) of over 90%, yet they selected the MDA as a more effective method.

To the best of our knowledge, no previous study has approached avalanche forecasting as an ordinal regression problem. Thus, this paper describes the data processing and model design to compare such methods in an avalanche forecasting setting. We analyze and compare the results obtained by each method, and conclude with a discussion on methodological considerations and limitations as well as describe possible future research.

II. Methods

The main libraries used are the `scikit-learn` (Pedregosa et al., 2011), `fast.ai` (Howard et al., 2020) and `pytorch` (Paszke et al., 2019) libraries. The full list of packages used are presented in the `requirements.txt` file.

A. Data

The data was recorded at four measurement stations located in the Swiss Alps, each containing measurements at 3-hour resolution. Three of the measurement stations ranged from November 1997 to May 2020, while the fourth ranged from November 2000 to May 2020. Examples of variables include snow height, wind velocity, and ground temperature. Moreover, the data included descriptive variables such as the minimum and maximum altitude of the validity of the forecasts. All non-descriptive variables were continuous.

The dependent variable represents the forecasted danger level of an avalanche for the upcoming day. The variable ranged from 1 to 5, where the former represents the lowest danger level and 5 represents the highest. Danger levels were forecast at 17:00 each day and were valid for 24 hours. Three characteristics of the dependent variable need to be considered when deciding on the approach to modelling and how to evaluate the models. First, the variable is ordered. Second, the classes are imbalanced. The danger levels 1-5 represented 20.8, 47.0, 30.8, 1.3, and 0.1 percent of the classes, respectively, in the concatenated and prepared dataset. Third, due to the inherent ordering of the variable, one may argue that model mistakes should be treated differently. That is, if the true label is 5, the error of predicting a 1 should be treated as more severe than predicting a 4.

B. Data preparation

The data was cleaned of most descriptive variables, variables with near-all missing values were removed and gaps of rows were filled with NAs. Because the number of samples with danger level 5 was small, we converted those to danger level 4. All missing values were forward-filled, limiting the filling to 48 hours. Finally, all variables with variance equal to 0 and nearly identical variables were dropped.

To allow the model to take the relation between measurements at subsequent time points into consideration, we included lagged features. Samples were augmented with the previous seven rows, corresponding to one day of previous data. The rationale for the use of one day's data is to be able to keep as much data as possible for modeling purposes, considering an already limited amount of samples for higher levels of danger. The previous two values of the dependent variable were also used as features. After augmenting the rows with previous measurements, rows with missing values were dropped from the samples. The dates were split into several features to allow the models to utilize temporal information and to consider seasonality. The additional features (date

parameters) were the year, week, day of the month, day of the year, and boolean values representing whether the measurement was made at the start or end of a year, quarter, or month. As we had now significantly increased the dimension of our data, we considered PCA as well (Jolliffe, 2002). We evaluated the models both with and without PCA and the additional date variables. Having prepared the data from each station, we merged the datasets into one. The station code was included as a predictor in the merged dataset.

C. Modelling approach

Four models were compared and evaluated: a baseline model and three candidate models. For the baseline model, we chose to use a multinomial logistic regression model with balanced weights. Because multi-class classification model treats model errors equally, ordinal models should perform accurately with respect the model selection criteria (see section II-D).

The first candidate approach transforms the k class ordinal problem into $k-1$ binary classification problems (Frank & Hall, 2001). The second approach transforms the problem into a multi-label classification problem, which is then modelled using a neural network (NN) (Cheng, 2007). The third candidate approach considers the problem with an extension to the proportional odds model (McCullagh, 1980). Henceforth, we refer to these methods as the Frank et al., Cheng, and OrdLoss approaches, respectively.

1) Frank et al. approach: Given K ordered classes $c_1 \leq c_2 \leq \cdots \leq c_K$, we seek to estimate the probability $P(y_n = c_i)$ that a sample x_n belongs to class c_i. To do this, we train an ensemble of $K-1$ binary classifiers, each of which predicts the probability $P(y_n > c_i)$ that y_n is larger than class c_i for $1 \leq i < K$. To predict the class of an unseen datum, the probabilities of the binary classifiers are combined as follows

$$
\begin{aligned}
P(y_n = c_1) &= 1 - P(y_n > c_1) \\
P(y_n = c_i) &= P(y_n > c_{i-1}) - P(y_n > c_i) \quad \text{for } 1 < i < K \\
P(y_n = c_K) &= Pr(y_n > c_{K-1})
\end{aligned}
$$

The predicted class is chosen to be the one with maximum probability (Frank & Hall, 2001).

The target z^i of the i-th binary classifier was generated from the target y of the original multi-class problem. Specifically, given a datum x_n originally labelled y_n, the new binary label z_n^i is given by

$$
z_n^i = \begin{cases} 1 & \text{if } y_n > c_i \\ 0 & \text{otherwise} \end{cases}
$$

where the ordered classes are denoted $c_1 \leq c_2 \leq \cdots \leq c_K$. This process is repeated for all the $K-1$ binary classifiers.

The binary classifiers are independent of each other, so they need not belong to the same family of models. Thus, the choice of models is a hyper-parameter of this approach. In turn, each of the selected models have their own set of hyper-parameters that must also be tuned. A naive grid search over all combinations of models and their hyper-parameters would be prohibitively expensive. Therefore, we tune the hyper-parameters in a two-step approach. The first stage selects the family of models (e.g. Logistic, SVM, etc.) for the three binary classifiers through a simple grid search. In this stage, the default hyperparameters of each model are used. Whether to pre-process the data with PCA or the date parameters was also explored as part of the first stage. In the second stage, we tune the hyper-parameters of each model independently, one model at a time.

Given there are 4 danger levels, we need to select three binary classifiers in the first stage. We considered the model families of Logistic Regression, Support Vector Machines, Decision Trees, Random Forests, and XGBoost. Readers will be familiar with the first two, and we briefly expand on the latter three.

Decision Trees are tree-shaped models where each node of the tree partitions the data into disjoint sets. The split is determined by a threshold on a feature that minimizes the *impurity*, or dissimilarity, in the target class (Fürnkranz, 2017). Random Forests are ensembles of Decision Trees, where each tree is trained on a random sub-sample of the data and each branch of a tree only considers a subset of the features. The predictions of the different trees is aggregated to construct the ensemble's prediction (Sammut & Webb, 2017). XGBoost is an efficient implementation of *gradient boosted* decision tree ensembles. During training, the data mis-classified by the current ensemble is *boosted*, i.e. weighted more heavily, so new models correct the mistakes of their predecessors (Brown, 2017; T. Chen & Guestrin, 2016).

The three model families selected by the first stage were XGBoost, Logistic Regression, and Decision Trees, in that order. In the second stage, a grid search over a large number of hyper-parameter combinations was run for each model. The selected hyper-parameters are reproduced in Table II.

2) Cheng approach: First, the target classes were encoded to a multi-label target. In the multi-label encoding, output classes were encoded to also "include" the lower-ranked classes. For example, danger level three would be encoded to [1, 1, 1, 0, 0]. The full encoding for the danger levels is the lower triangular matrix with 1s,

$$
T = \begin{bmatrix} 1 \\ 2 \\ \vdots \\ 4 \end{bmatrix} \rightarrow T = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 1 \end{bmatrix}
$$

To transform the predicted probabilities to the original categories, the output targets are cut at the point when the probability is below a pre-defined threshold of 0.5.

The NN architecture included 12 embedding layers, an embedding layer for each variable derived from the date variable and also the station variable. Embeddings are mappings from the discrete, categorical variables to real vectors, which themselves are learned during training (Thomas, 2018). The embedding sizes were chosen to be $\min(600, round(1.6 * n_{cat}^{0.56}))$, where n_{cat} is the number of unique values in the variable. The value is the default setting given in the `fast.ai` library.

The hyperparameters are presented in Table III. PCA was not considered as a hyperparameter for this approach, as no straightforward way of combining PCA and embedding layers were discovered. Each hidden layer are bundled with two components. The first component is a `BatchNorm1d` layer, which normalizes each mini-batch in the gradient descent (Ioffe & Szegedy, 2015), allowing us to speed up training. The second component is a `Linear` layer. As presented in the hyper-parameter configuration, the activation function for each layer except the output layer is the LeakyReLU activation function, with negative slope equal to 0.01.

The network is configured to have as many output nodes as there are danger levels. The output layer has the sigmoid activation function. The loss function is given by the binary cross-entropy (Cheng, 2007),

$$\mathcal{L}_{Cheng} = \frac{1}{k}\sum_{i=1}^{k} y_i log\ f(x_i) + (1-y_i)\ log\ (1-f(x_i))$$

where f is our classifier, (S) is the sample, $f(x_i)$ and y_i represent the classified and true label, respectively.

The loss on the training set and the test set is presented in Figure 2. The final model is picked after the last epoch.

3) Ordloss approach: This approach is a modification of the GLM where we assume that

$$y = \begin{cases} 1 \text{ if } f(X) \leq \theta_1, \\ 2 \text{ if } \theta_1 < f(X) \leq \theta_2, \\ 3 \text{ if } \theta_2 < f(X) \leq \theta_3, \\ \vdots \\ K \text{ if } \theta_{K-1} < f(X) \end{cases}$$

where $\theta_1 < \theta_2 < \cdots < \theta_{K-1}$ are unknown threshold parameters, K is the number of classes and $f(X)$ is a latent variable that is the output of a neural network (Woodbridge, 2002). The conditional distribution of y is described as:

$$\mathbb{P}(y = k|X) = \sigma(\theta_k - f(X)) - \sigma(\theta_{k-1} - f(X))$$

where σ is the link function. In our case, the sigmoid function is used.

The parameters w of the neural network and thresholds θ are estimated by the maximum likelihood with balanced weights ζ. The loss function of each training sample can thus be written:

$$log\mathcal{L}(w, \theta|x_i, y_i) = \sum_{k=1}^{K} \mathbb{I}_{\{y_i=k\}} \zeta_k log[\mathbb{P}(y_i = k|x_i)]$$

In order to preserve the constraint of the increasing order of thresholds we use the projected gradient descent method in order to minimize the negative log-likelihood (Y. Chen & Wainwright, 2015).

The selected hyper-parameters are presented in Table IV. Indicator variables for the dates, station and the PCA were retained. We stop training when the validation error increases (Figure 1).

D. Model Evaluation

To evaluate the performance of the models we compute the per-class precision and recall, as well as the macro-averaged mean squared error (MAMSE). Per-class precision measures the number of correct positive predictions made. Recall measures the number of correct positive predictions from all possible positive predictions. For unbalanced data, these measures provide a better understanding of the effectiveness of classifiers when compared to measures such as accuracy or AUC-ROC that are skewed by the dominant classes (Saito & Rehmsmeier, 2015). The macro-averaged mean absolute error is a measurement for class-imbalanced ordinal regression (Baccianella et al., 2009). We define the MAMSE, inspired by macro-averaged mean absolute error, as

$$\text{MAMSE}(f, (S)) = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{|(S)_j|} \sum_{x_i \in (S)_j} (f(x_i) - y_i)^2$$

where n is the number of classes, (S) is the sample, and $|(S)_j|$ represents the number of rows that contain the true label y_j. By the term $\frac{1}{(S)_j}$, the MAMSE weights the error by taking into account the imbalance among classes. By squaring the deviations from the classification and the true label, rather than calculating the absolute value of the deviations, the metric penalizes larger deviations from the true label with a higher error.

As our methods all have built-in randomness in the training procedure, we rerun fitting $N = 100$ times in order to get an estimate for the mean and confidence interval (CI) for our performance measures. Note that the MAMSE for the baseline model is simply the point estimate.

E. Model Selection

Each of our models includes parameters we need to optimize that influence the quality of predictions. We used randomized cross-validation to estimate the generalization error (Bergstra & Bengio, 2012). To do this, we divided our data into three parts. From 1997 to 2010, we trained 100 random samples of combinations of parameters for each model, then evaluated these parameters on data from 2010 to 2015. We selected the parameters that had the smallest MAMSE on the validation data. The final model was trained from 1997 to 2015 using these parameters and we present the results over the 2015 to 2020 period.

III. Results

TABLE I
MEAN MAMSE AND STANDARD ERROR AT $\alpha = 0.05$.

Benchmark	Frank et al.	Cheng	OrdLoss
0.429	0.404 ± 0.001	**0.182** ± 0.004	0.406 ± 0.006

From Table I, we find that all candidate models performed better than the benchmark model with respect to the MAMSE. The method that performed best was the Cheng approach, whereas the benchmark model scored the lowest.

Table V compares the performance between the baseline and candidate approaches with respect to per-class precision and recall. The Cheng approach scored the highest precision and recall for all danger levels. The OrdLoss scored the lowest for four categories of performance measures. We see that it has difficulty in separating class 3 from class 4 by plotting an estimate of the *pdf* of the latent variable with the thresholds (see Figure 4).

Table VI compares the predicted labels to the actual danger levels 1 and 4. The rationale for showing these classifications are that mis-classifications at this level would cause the most damage; predicting a danger level 1 when actually a 4 would be the most dangerous to humans, whereas predicting a danger level 4 when actually a 1 could be economically costly. The Cheng model was the only model that did not predict a lower danger level than 3 when the actual label was a 4. No model predicted a 1 when actually a 4. On the other end, all models except the OrdLoss model predicted some instances lower than 2 when actually a 1.

IV. Discussion

Deciding on whether an avalanche has occurred is not an easy task, and there is no clear consensus on how the task should be approached (see e.g. Schimmel et al., 2017 and Eckerstorfer et al., 2017 for recent developments). In our context, the predictions were made by trained forecasters, and it is plausible that this serves as a good proxy for avalanche occurrence. However, one clear disadvantage of our data is that the forecasts were not validated, meaning that we have no information on whether it serves as a good proxy or not.

We chose to concatenate the samples with previous values of the variables to take values from previous time points into consideration, as described in (Hyndman & Athanasopoulos, 2018b). The rationale was to be able to utilize the methods and extensive libraries for standard classification. In doing so, we acknowledge the potential issues with using this method, for example, that features may be highly correlated with their lagged versions and thus that lagging may introduce co-linearity issues (Fishman, 2014). However, because the OrdLoss method utilizes PCA, the issue with co-linearity is alleviated. Further, neural networks and XGBoost have been described as able to handle co-linearity well (De Veaux and Ungar, 1994; Herdter Smith, 2019), and Dormann et al., 2013 showed that SVMs were resilient under high co-linearity conditions.

The Frank et al. approach computes the difference of probabilities of the independently trained binary classifiers and therefore runs the risk of returning negative probabilities. While we observe this phenomenon in our data, it is somewhat rare with only 7.1% of all predicted class probabilities being less than zero. Moreover, most of these were very close to zero with few, albeit large, outliers. Since the probabilities for danger levels 1 and 4 depend on only one model, no negative probabilities are observed. This is summarized in figure Figure 3.

Despite these observations, it is not clear that negative probabilities impact predictive performance. By construction, the class probabilities sum to 1, even if some are negative. This means that some classes will have high positive probabilities, which we can interpret as increased certainty of the model. In this view, there is no issue and we can simply consider the class probabilities as "confidence scores" which fall in the range $[-1, 1]$. Finally, we note that the paper outlining this approach does not discuss the issue of negative probabilities (Frank & Hall, 2001).

To map the output of the Cheng approach to the ordinal category, we pick the index that corresponds to the last 1. As described in the original paper, there is a possibility that the model outputs a 1 after a 0, in which the transformation back to the predicted category would fail. One such sample was observed in the test set. To alleviate the problem, one may include a constraint that the labels must be less than or equal to its previous category, however, no such configuration was explored within the scope of this project.

Our data is not stationary, the number of avalanches and even the amount of snow that falls varies greatly from year to year (Gauthier et al., 2017). As a result, our choice to split our data into three to select the best model may be subject to undesirable biases. Another approach would be to perform a walk-forward cross-validation (Hyndman & Athanasopoulos, 2018a). This allows us to assess our models more realistically, as in practice it would be useful to re-train when new data become available.

A. Future research

It would be interesting to review the effectiveness of some of the techniques if the human forecasted danger levels were validated. This would provide data that more accurately captures the relationships between hazard levels and the predictors. Applying infra-sound-based or satellite-based avalanche detection could also be an alternative that provides greater and more accurate data resolution (NORCE, 2019).

Further work is needed to use specific time-series prediction models to better capture the temporal dependencies that exist. This would include more classical econometric models such as VAR-X (Palakeel et al., 2017) or modern recurrent neural network approaches (Shih et al., 2019).

V. Conclusion

This study presents a comparison of machine learning models in avalanche risk prediction. Starting from raw data that we first cleaned then augmented, we trained different models using cross-validation to adjust the hyperparameters. We obtained powerful models that perform better than the baseline method. Several considerations and difficulties were encountered which can be developed in more detail in an attempt to increase performance further.

VI. Acknowledgements

The authors are thankful that Michele Volpi from the Swiss Data Science Center (SDSC) devoted time and effort to guide the project.

References

Baccianella, S., Esuli, A., & Sebastiani, F. (2009). Evaluation measures for ordinal regression. *ISDA 2009 - 9th International Conference on Intelligent Systems Design and Applications*, 283–287. https://doi.org/10.1109/ISDA.2009.230

Ballesteros-Cánovas, J. A., Trappmann, D., Madrigal-González, J., Eckert, N., & Stoffel, M. (2018). Climate warming enhances snow avalanche risk in the Western Himalayas. *Proceedings of the National Academy of Sciences of the United States of America*, *115*(13), 3410–3415. https://doi.org/10.1073/pnas.1716913115

Bergstra, J., & Bengio, Y. (2012). Random search for hyper-parameter optimization. *J. Mach. Learn. Res.*, *13*, 281–305.

Brown, G. (2017). Ensemble learning. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of machine learning and data mining* (pp. 393–402). Boston, MA, Springer US. https://doi.org/10.1007/978-1-4899-7687-1_252

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *CoRR*, *abs/1603.02754*arXiv 1603.02754. http://arxiv.org/abs/1603.02754

Chen, Y., & Wainwright, M. (2015). Fast low-rank estimation by projected gradient descent: General statistical and algorithmic guarantees. *ArXiv*, *abs/1509.03025*.

Cheng, J. (2007). A neural network approach to ordinal regression.

Choubin, B., Borji, M., Mosavi, A., Sajedi-Hosseini, F., Singh, V. P., & Shamshirband, S. (2019). Snow avalanche hazard prediction using machine learning methods. *Journal of Hydrology*, *577*(March). https://doi.org/10.1016/j.jhydrol.2019.123929

De Veaux, R. D., & Ungar, L. H. (1994). Multicollinearity: A tale of two nonparametric regressions, 393–402. https://doi.org/10.1007/978-1-4612-2660-4_40

Dormann, C. F., Elith, J., Bacher, S., Buchmann, C., Carl, G., Carré, G., Marquéz, J. R. G., Gruber, B., Lafourcade, B., Leitão, P. J., Münkemüller, T., McClean, C., Osborne, P. E., Reineking, B., Schröder, B., Skidmore, A. K., Zurell, D., & Lautenbach, S. (2013). Collinearity: A review of methods to deal with it and a simulation study evaluating their performance. *Ecography*, *36*(1), https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1600-0587.2012.07348.x, 27–46. https://doi.org/https://doi.org/10.1111/j.1600-0587.2012.07348.x

Eckerstorfer, M., Malnes, E., & Müller, K. (2017). A complete snow avalanche activity record from a norwegian forecasting region using sentinel-1 satellite-radar data. *Cold Regions Science and Technology*. https://doi.org/10.1016/j.coldregions.2017.08.004

European Avalanche Warning Services. (2020). Avalanche Danger Scale. Retrieved December 13, 2020, from https://www.avalanches.org/standards/avalanche-danger-scale/

Fishman, G. S. (2014). Distributed Lag Models. *Spectral Methods in Econometrics*, 146–180. https://doi.org/10.4159/harvard.9780674334076.c4

Frank, E., & Hall, M. (2001). A simple approach to ordinal classification (L. De Raedt & P. Flach, Eds.). In L. De Raedt & P. Flach (Eds.), *Machine learning: Ecml 2001*, Berlin, Heidelberg, Springer Berlin Heidelberg.

Fürnkranz, J. (2017). Decision tree. In C. Sammut & G. I. Webb (Eds.), *Encyclopedia of machine learning and data mining* (pp. 330–335). Boston, MA, Springer US. https://doi.org/10.1007/978-1-4899-7687-1_66

Gauthier, F., Germain, D., & Hétu, B. (2017). Logistic models as a forecasting tool for snow avalanches in a cold maritime climate: Northern gaspésie, québec, canada. *Natural Hazards*, *89*, 201–232.

Herdter Smith, E. (2019). Using extreme gradient boosting (xgboost) to evaluate the importance of a suite of environmental variables and to predict recruitment of young-of-the-year spotted seatrout in florida. *bioRxiv*, https://www.biorxiv.org/content/early/2019/02/08/543181.full.pdf. https://doi.org/10.1101/543181

Howard, J. Et al. (2020). Fastai. GitHub.

Hyndman, R. J., & Athanasopoulos, G. (2018a). 5.10 Time series cross-validation (2nd ed.), In *Forecasting: Principles and practice* (2nd ed.). Melbourne, Australia, OTexts. https://otexts.com/fpp2/

Hyndman, R. J., & Athanasopoulos, G. (2018b). *Forecasting: Principles and Practice* (2nd ed.). Melbourne, Australia, OTexts. https://otexts.com/fpp2/

Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. *32nd International Conference on Machine Learning, ICML 2015*, *1*arXiv 1502.03167, 448–456.

Jolliffe, I. T. (2002). Principal component analysis.

Kroese, D. P., Taimre, T., & Botev, Z. (2011). Handbook of monte carlo methods.

McCullagh, P. (1980). Regression models for ordinal data. *Journal of the royal statistical society series b-methodological*, *42*, 109–127.

NORCE. (2019). Satellites improve avalanche forecasting [Accessed: 2020-12-16]. https://www.norceresearch.no/en/news/satellitter-bidrar-til-bedre-snoskredvarsling

Palakeel, P., Nambiar, M. N., Vishnu, V., & Deepika, M. (2017). Estimating demand function for gold using the vector auto regression model. *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 1–6.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library (H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett, Eds.). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32*. Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

Saito, T., & Rehmsmeier, M. (2015). The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLoS ONE*, *10*.

Sammut, C., & Webb, G. I. (Eds.). (2017). Random forests. In *Encyclopedia of machine learning and data mining* (pp. 1054–1054). Boston, MA, Springer US. https://doi.org/10.1007/978-1-4899-7687-1_695

Schimmel, A., Hübl, J., Koschuch, R., & Reiweger, I. (2017). Automatic detection of avalanches: evaluation of three different approaches. *Natural Hazards*, *87*(1), 83–102. https://doi.org/10.1007/s11069-017-2754-1

Shih, S.-Y., Sun, F.-K., & Lee, H.-y. (2019). Temporal pattern attention for multivariate time series forecasting. *Machine Learning*, 1–21.

Thomas, R. (2018). An Introduction to Deep Learning for Tabular Data. https://www.fast.ai/2018/04/29/categorical-embeddings/

Woodbridge, J. M. (2002). Econometric analysis of cross section and panel data.

APPENDIX A
FIGURES

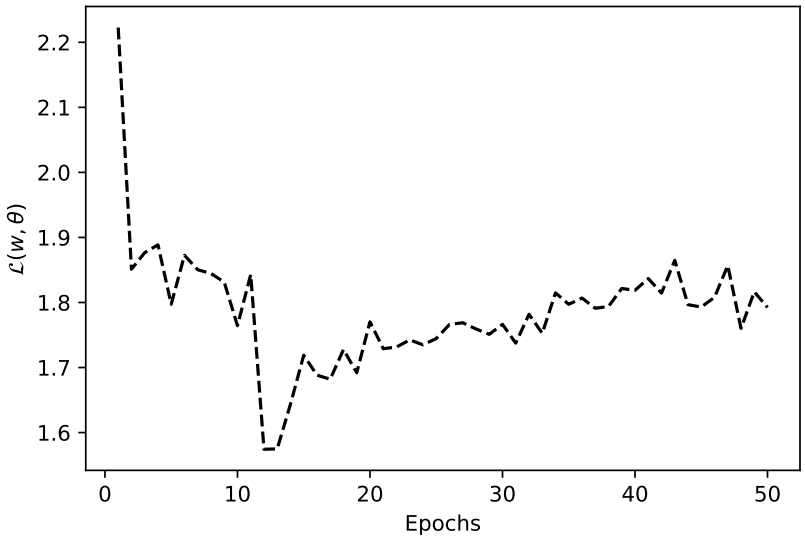

Fig. 1. Validation loss per epoch for OrdLoss approach

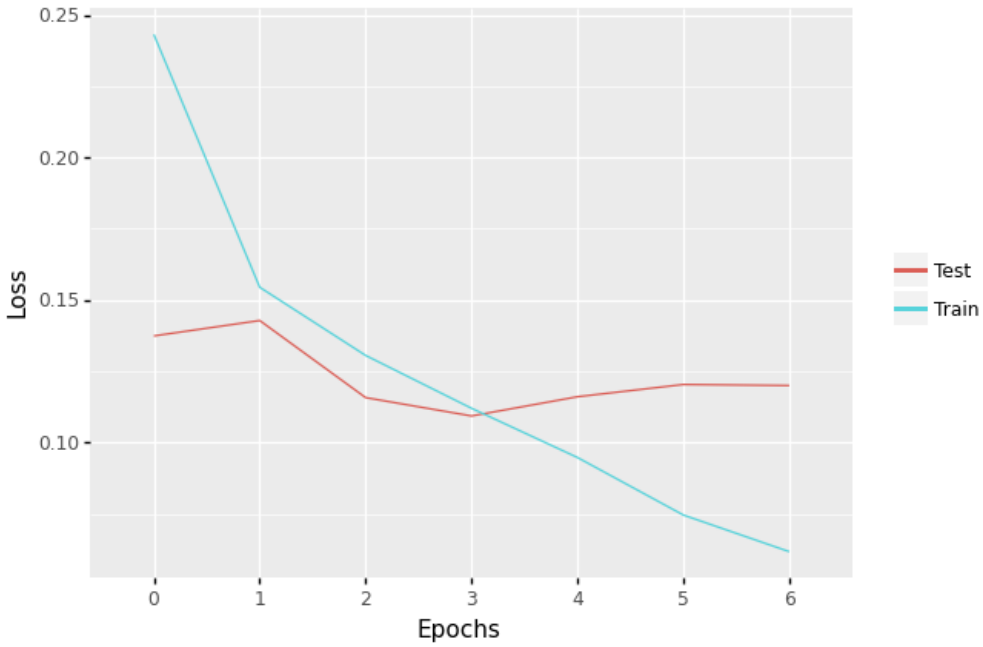

Fig. 2. Training loss against validation loss for the Cheng approach.

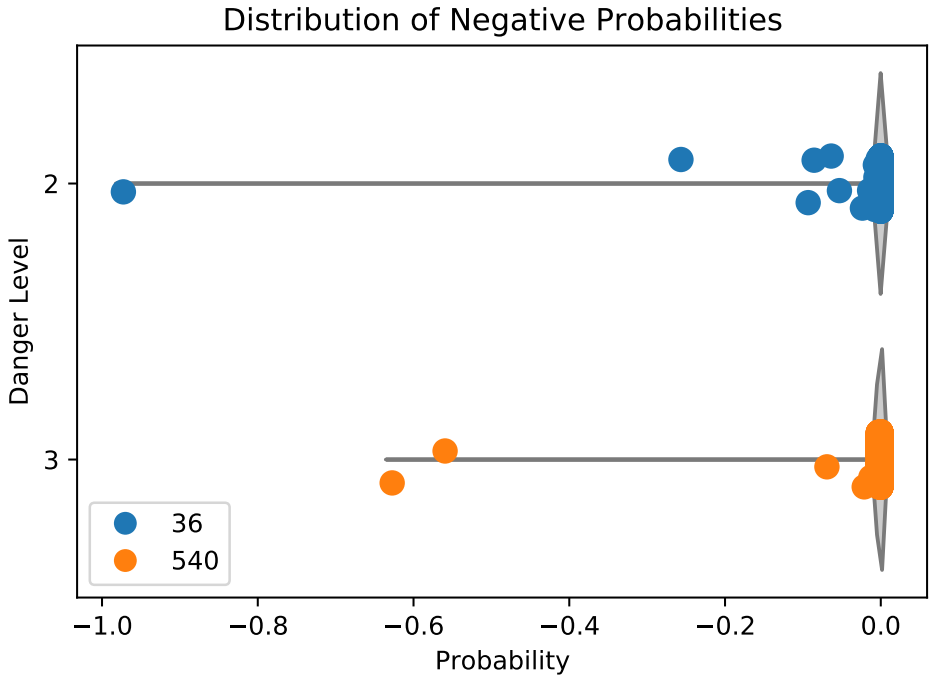

Fig. 3. Distribution of negative probabilities for the Frank et al. approach. Notice that the data is heavily concentrated around 0, with a few large outliers.

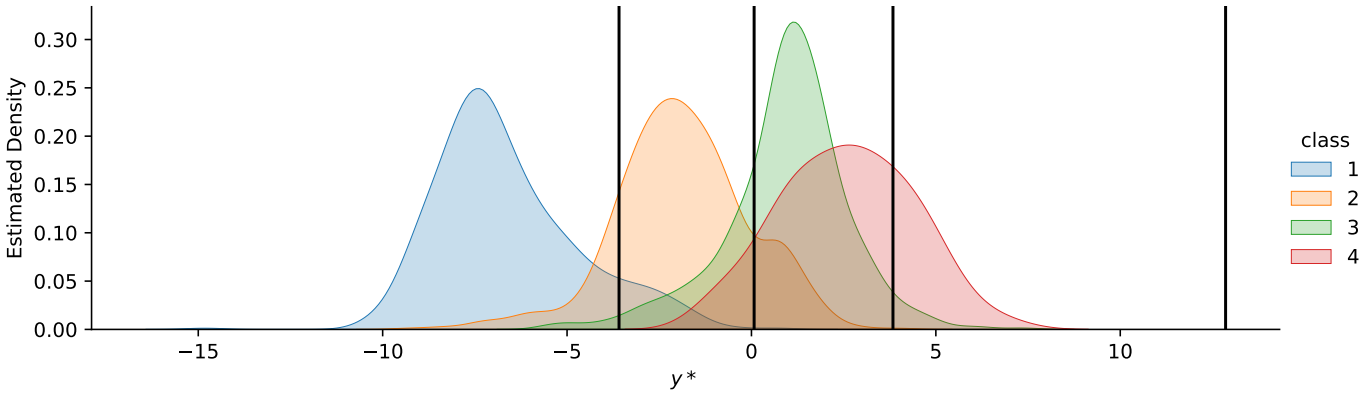

Fig. 4. Kernel Density Estimation plot of the latent variable with fitted thresholds

APPENDIX B
TABLES

TABLE II
SELECTED HYPER-PARAMETERS FOR THE FRANK ET AL. APPROACH.

	Hyper-parameter	Value
Data Processing	PCA	False
	Date parameters	False
XGBoost	Sub-sample Ratio of Features per Tree	1.0
	Learning Rate	0.1
	Min Loss Reduction for Split	0.2
	Max Depth	9
	Weight of Minority Class	1.0
Logistic	Inverse of Regularization	2.0
	Tolerance	0.0001
	Unbalanced Class Weights	Disabled
Decision Tree	Min Samples Required for Leaf	1
	Min Samples Required for Split	8
	Max Depth	6
	Unbalanced Class Weights	Disabled

TABLE III
SELECTED HYPER-PARAMETERS FOR THE CHENG APPROACH. NOTE THAT PCA WAS NOT CONSIDERED FOR THIS APPROACH.

Layers	Nodes $(l^{(1)}, l^{(2)})$	Learning rate	Activation	Batch size	PCA	Date parameters
2	100, 100	0.01	LeakyReLU(0.01)	64	-	True

TABLE IV
SELECTED HYPER-PARAMETERS FOR THE ORDLOSS APPROACH.

Layers	Nodes $(l^{(1)}, l^{(2)})$	Learning rate	Activation	Batch size	PCA	Date parameters
2	5,2	0.01	LeakyReLU(0.01)	2	True	True

TABLE V
MEAN PERFORMANCE MEASURES AND SE AT $\alpha = 0.05$. THE HIGHEST VALUES ARE BOLDED, THE LOWEST ARE IN ITALIC FONT.

Approach	Danger	Precision	Recall
Baseline	1	0.89	0.89
	2	*0.76*	0.80
	3	0.81	*0.77*
	4	*0.16*	0.18
Frank et al.	1	0.937 $\pm$ 0.000	0.883 $\pm$ 0.000
	2	0.771 $\pm$ 0.000	0.823 $\pm$ 0.000
	3	0.779 $\pm$ 0.000	0.784 $\pm$ 0.000
	4	0.292 $\pm$ 0.005	*0.175* $\pm$ 0.004
Cheng	1	**0.950** $\pm$ 0.002	**0.930** $\pm$ 0.003
	2	**0.871** $\pm$ 0.002	**0.873** $\pm$ 0.002
	3	**0.860** $\pm$ 0.002	**0.884** $\pm$ 0.003
	4	**0.717** $\pm$ 0.010	**0.610** $\pm$ 0.013
Ordloss	1	*0.877* $\pm$ 0.007	*0.858* $\pm$ 0.009
	2	0.756 $\pm$ 0.004	*0.711* $\pm$ 0.005
	3	*0.730* $\pm$ 0.005	0.811 $\pm$ 0.005
	4	0.386 $\pm$ 0.012	0.270 $\pm$ 0.012

TABLE VI
SUBSET OF THE CLASSIFICATION MATRICES FOR ACTUAL DANGER LEVEL 1 AND 4.

Approach	Actual label	Predicted label 1	2	3	4
Baseline	1	540	61	2	2
	4	0	4	37	8
Frank et al.	1	534	69	2	0
	4	0	4	35	10
Cheng	1	551	53	1	0
	4	0	0	20	29
Ordloss	1	548	57	0	0
	4	0	6	30	13

APPENDIX C
CONFIDENCE INTERVALS

Starting from the central limit theorem, it is possible to give an asymptotic $1-\alpha$ confidence interval to our estimates (Kroese et al., 2011) given by:

$$\hat{\mathbb{I}}_{\alpha,N} = \left[\hat{\mu}_N - c_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{N}}, \hat{\mu}_N + c_{1-\alpha/2}\frac{\hat{\sigma}}{\sqrt{N}}\right]$$

Where $\hat{\sigma}$ is the sample standard deviation, $c_{1-\alpha/2}$ is the quantile of the student distribution and $\alpha = 0.05$.