Tailored Military Recruitment through Machine Learning Algorithms

Robert Bryce, Ryuichi Ueno, Christopher Mcdonald and Dragos Calitoiu[a]

Director General Military Personnel Research and Analysis, Department of National Defence, 101 Colonel By Drive, Ottawa, Canada

Keywords: Machine Learning, Ensemble Learning, Workforce Analytics, Recruitment.

Abstract: Identifying postal codes with the highest recruiting potential corresponding to the desired profile for a military occupation can be achieved by using the demographics of the population living in that postal code and the location of both the successful and unsuccessful applicants. Selecting N individuals with the highest probability to be enrolled from a population living in untapped postal codes can be done by ranking the postal codes using a machine learning predictive model. Three such models are presented in this paper: a logistic regression, a multi-layer perceptron and a deep neural network. The key contribution of this paper is an algorithm that combines these models, benefiting from the performance of each of them, producing a desired selection of postal codes. This selection can be converted into N prospects living in these areas. A dataset consisting of the applications to the Canadian Armed Forces (CAF) is used to illustrate the methodology proposed.

1 INTRODUCTION

Military recruitment refers to the overall process of attracting and selecting suitable candidates for military occupations. At any given time, a broad range of efforts are deployed in order to continuously improve this process by making it more efficient and more effective, or, in other words, to ensure the best use of the available resources to select the best candidates for the job. Many of these recruiting efforts, such as advertising, or career fairs, are addressed to the general population. Recently, a new line of recruiting efforts became of interest, that of identifying sub-groups within the general population that have higher odds than other sub-groups to contain successful candidates for a particular set of jobs, and then tailor recruiting efforts to these sub-groups. This paper presents a mathematical translation of the process of identifying the most promising sub-groups, i.e., those with the highest potential for prospects that correspond to the desired profile for the occupation(s) of interest. To be more specific, the aim is not to identify individuals per se, but the geographical areas where they live, at the postal code level of granularity. This will allow, for example, for tailored mailing campaigns to the households in a select set of postal codes. The selection of this level of granularity was pragmatic, based on the fact that this information is known for all applicants (whether successful or not), and that a multitude of external data sources exist, which provide rich information about neighbourhoods, also at the postal code level. The approach presented here is based on using the postal code as a primary key (in database terminology) to connect an applicant with the neighbourhood to which he/she belongs which makes possible to augment the applicant data with the demographic attributes of that neighbourhood. With this augmentation realized, it is possible to identify the separation between the profile of the successful applicants (enrollees) and that of non-successful applicants, using only demographics of the population living in the postal code. This separation (a mathematical relationship) can be used to derive the probability to be enrolled for individuals living in previously 'untapped' postal codes (i.e., no previous applicants from that postal code), thus uncovering new promising areas where tailored recruited efforts could be applied successfully. The population in each postal code is known; therefore the number of individuals can be determined. Identification of N individuals with the highest probability to be enrolled, from a population living in untapped postal codes, can be done by ranking the population using a machine learning predictive model derived from the separation described above. Three such models are presented in this paper, along with an algorithm that combines them, benefiting from the strength of each

[a] https://orcid.org/0000-0003-0173-9846

Bryce, R., Ueno, R., Mcdonald, C. and Calitoiu, D.
Tailored Military Recruitment through Machine Learning Algorithms.
DOI: 10.5220/0010506500870092
In *Proceedings of the 2nd International Conference on Deep Learning Theory and Applications (DeLTA 2021)*, pages 87-92
ISBN: 978-989-758-526-5

of the individual models, which is considered the key contribution of this paper.

A brief presentation of the paper follows. We describe developing three Machine Learning (ML) models to predict the probability of applicant's success by postal codes (Section 2.1). We propose a dynamic method for combining models in a manner that satisfies the requirement *N* and considers the performance of each model (Section 2.2).We use a dataset consisting of applications to the Canadian Armed Forces (CAF) to illustrate the methodology we describe, allowing for a concrete discussion of implementation issues (Section 3).

2 MACHINE LEARNING MODELS FOR TAILORED RECRUITMENT

A machine learning algorithm is an algorithm able to learn from data. The concept of learning is used here in the context of improving the algorithm performance, for a given task, by using experience (more data). The task explored in this research is the classification in two categories; the learning algorithm is asked to produce a function $f : \mathbb{R}^n \to \{0, 1\}$. In order to solve this task, we implemented three architectures: a logistic regression, a multi-layer perceptron, and a deep neural network.

The data sets consist of demographics data at the postal code level, namely of 760 attributes (Environics Analytics, 2018), as well as a labeled data set of 59,084 postal codes associated with applications to the CAF in the 2015-18 timeframe. The label is '1' for a successful applicant (enrolled) and '0' for an unsuccessful applicant. Applicants with no final decision were removed from the data set. The labeled set was split into development and validation sets, as will be discussed below.

2.1 Machine Learning Models

Model 1: A logistic regression model (model LR) was trained as a baseline model. A 14 dimensional subset of the 760 attributes was chosen by stepwise selection. Specifically, for stepwise selection a significance level of 0.1 was required to allow a variable into the model and a significance level of 0.01 for a variable to stay in the model. Collinearity was tested for, with the acceptable variance inflation factor set to < 2.5 and the acceptable condition index to < 10. The Hosmer and Lemeshow goodness-of-fit test for the final selected model was used. The specific parameter values used for stepwise selection were based on standard guidance and diagnostics, for example see (Chen et al., 2003).

Table 1: Model performance on the validation set. The percentage of observation and the mean probability are reported for each decile.

Decile	LR		MLP		DNN	
	Obs.	p_mean	Obs.	p_mean	Obs.	p_mean
1	15.8	60.0	16.4	64.9	16.1	58.9
2	12.3	44.4	13.2	45.9	13.0	48.7
3	11.2	42.3	11.4	43.1	12.4	46.4
4	10.8	40.8	10.8	40.9	11.2	44.6
5	10.2	39.6	10.2	39.1	10.0	42.8
6	9.4	38.4	8.8	36.7	9.7	40.7
7	9.0	37.0	8.8	34.1	8.4	38.3
8	8.0	35.3	8.1	30.9	7.7	35.0
9	6.9	32.8	6.6	27.5	6.4	30.6
10	6.4	24.5	5.6	22.7	5.2	24.4

Model 2: A classic feed forward multi-layer perceptron (model MLP) with one hidden layer of five nodes and sigmoidal weights was trained via back-propagation (Rumelhart et al., 1986). The Adam optimizer was used, with the recommended default parameters (Kingma and Ba, 2015). L2 regularization was used to penalize large weight values, with regularization terms of 0.0001. The MLP was trained on the same 14 dimension subset of attributes as the LR model, as accuracy was poor when trained on the full 760 attributes.

Model 3: A Deep Neural Network (model DNN) classifier, with three hidden layers of 30 nodes each and rectified linear unit weights, was also trained with back-propagation. Again, the Adam optimizer was used, with the recommended default parameters (Kingma and Ba, 2015). L2 regularization was used to penalize large weight values, with regularization terms of 0.01. The DNN was trained on the full 760 dimensional attributes.

For the LR model the labeled data set was split into two equal subsets, for development and validation. In contrast, for the neural networks a 80%-20% split was used, due to more parameters existing in these architectures (81 parameters for the MLP and 24,721 for the DNN, versus 15 for the LR model).

The entire validation dataset is scored and a probability is computed from the score, using the standard transformation $p = \frac{e^{\text{score}}}{1+e^{\text{score}}}$, such that each postal code receives a score and a corresponding probability. Next, the validation dataset is sorted in descending order of the probabilities. We build deciles, the first having the highest probability to enroll and the last having the lowest probability. We report for each decile the following numbers: the fraction of enrollments realized on that decile (% Obs.), and the mean of probabilities predicted by the model (Mean Prob.). Table 1 contains these values. The lift is defined as the ratio of enrollments realized in a given segment (decile here) to the average expected under a uniform random distribution; for example, the lift for the LR model on the first decile is $\frac{15.8\%}{10.0\%} = 1.58$. A model is powerful if it can group in the first decile a high percentage of enrollments. Note that the top deciles exhibit enhanced lift for all three models.

Observe that the DNN has limited data per weight, 45,818 observations for 24,721 weights, placing training well outside the classic '10x data to weights' rule of thumb regime (Baum and Haussler, 1989).

2.2 Ensemble of Models

The underlying motivation of this work is to find a means of combining ML models, to benefit from different strengths of each model. We first provide the algorithm for concreteness and then discuss the motivation behind this approach. The algorithm proposed is general, namely it can combine any number of models. The motivation of the algorithm is presented below. There are two main constraints for the design: the target audience N (and a corresponding estimation of B postal codes) and the error within any given model. Predictions for any specific postal code may be in error and predicted orderings may have local 'mixing' and mis-ordering. By asking for an agreement set between several models up to a given number B of postal codes (A1), we place the focus near the 'top' postal codes where we are most interested in good performance. This approach, majority voting, is a common ensemble averaging scheme (Wolpert, 1992), but we note that it does not account for the fact that each model has different performance. By introducing the predictive strength of the models, the algorithm uses the lift in the segment under consideration (observed success in segment/overall observed success) in order to weight rankings (A2). We subtract the normalized lift from one in order to account for the fact that high rankings have a low index value. Note that by focusing on lift and rank order, we do not require a precise calibration of the predicted probabilities. The weighted ranks (A3) will not necessarily be integers; this is not substantial as sorting on non-integers will lead to interpretable ranking.

Inputs:

- The number of postal codes to target, B, estimated from the target audience N.
- The lift profiles for each model.
- Predictions for each model (postal code, p^1, p^2, ... , p^M), where p^i is the vector of the i^{th} ML model prediction of applicants' success, and M is the number of models. (In our setting $M = 3$).

Output:

- Top B ranked postal codes.

Algorithm to Determine the Top B Postal Codes:

- A0: Rank and sort the postal codes for each model, using the p^i vector.
- A1: Find the intersection which contains the top B postal codes, as ranked by each p^i. This 'agreement set' consists of the top S postal codes, with $S \geq B$.
- A2: Determine the lift for each model ℓ_i for the deciles containing the S postal codes. Define normalized weights $w_i = \frac{1}{M-1}(1 - \frac{\ell_i}{\sum_k \ell_k})$, where the sum is over the M models in the ensemble.
- A3: Find the final weighed ranking r_i' for the postal codes in the agreement set, $r_i' = \sum_k r_i^k \cdot w_k$ where i indexes the postal codes and k the models.
- A4: Sort and output the reranked top B postal codes.

3 RESULTS

As discussed in previous section, we trained three ML models (a LR, a MLP, and a DNN model). In the example here, we consider $B = 2000$ postal codes to target; the specific value in practice will be dictated by N. We apply the algorithm to 694,621 unlabelled and 'untapped' (with respect to our labeled data subset) postal codes. The number of untapped postal codes is derived by subtracting the number of postal codes in our labeled data from the total number of postal codes in Canada.

For the selection of B postal codes derived with the algorithm presented above, the number N_f of individuals can be computed. Environics demographic data contains the population per age group for each postal code. If N_f is different than the initial target audience N, an adjustment on B can be applied, increasing or decreasing it.

The lift which characterizes model performance is considered in deciles here, for stability reasons, though percentiles or other more granular breakdowns are potentially useful, given sufficient data. Use of deciles is standard practice in industry as the 'cream of the crop' is the focus. Figure 1 plots the cumulative lift percentage by decile for each model. It was observed that the DNN performs better than the other two models over the majority of deciles, as can be more clearly seen in the inset which shows the residual between the observed lift and the uniform random model with no lift. Further, the relative performance depends on which decile cutoff is considered. This demonstrates the utility of adaptively selecting the lift to match the region we are interested in. For example, if the overlap is identified in the first two deciles the lift corresponding to these two deciles must be used.

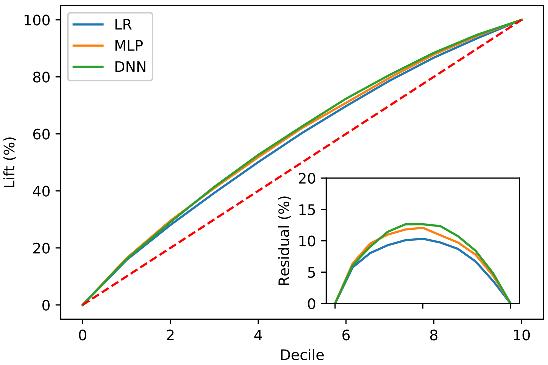

Figure 1: Cumulative lift by decile for each model. The dotted red line is the null case random model; the inset shows the residual between the models and the null case.

Note that logistic regression is a linear model while neural networks can account for nonlinear separation boundaries. With that in mind, the MLP model, with its five nodes, has more similar behaviour to the LR model than the DNN when overlap is considered (see Figure 2) yet still appears to display more capacity to separate than the LR (linear) model (see Figure 1, where performance approaches that of the DNN).

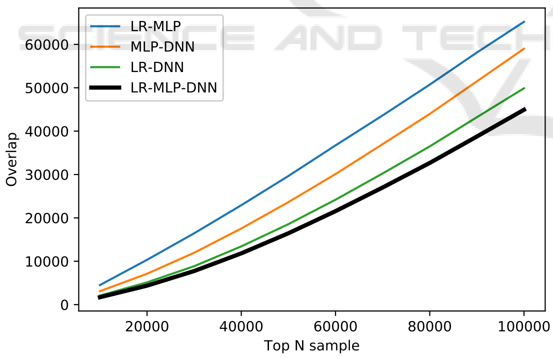

Figure 2: Agreement set size between models.

For illustration of the algorithm, Table 2 shows five postal codes randomly selected from the agreement set, their ranks in the three models, and their reordered ranking; for confidentiality, the postal codes are not presented explicitly, but denoted as (pc1, pc2, ...).

We should emphasize that the approach taken here is not intended to be static, but rather a dynamic learning approach which improves over time. Based on future campaigns using this approach we will update predictions. In particular, as we obtain more data the underlying models and their lifts will be improved.

Table 2: Example postal code rankings and reranking for five untapped postal codes.

P. Code	LR Rank	MLP Rank	DNN Rank	Ens. Rank
pc1	8	15	25	5
pc2	99	97	242	73
pc3	2813	623	875	470
pc4	1338	421	3773	543
pc5	736	3250	10665	1337

4 DISCUSSION

4.1 Weights

We focus on the 'best' predicted B postal codes and this makes the lift in the top percentiles a natural means of estimating a model's performance. We also want to eliminate any postal code that is not near the top across models; to do this, we use the intersection, which may be too conservative, and in which case majority voting may become a viable alternative. Calibration of probabilities (e.g., ensuring absolute accuracy of numbers) is difficult and, in general, there has been little work done in this area. For this reason, we do not use the absolute probabilities in determining performance or reranking, but rather the relative rankings. The resulting algorithm emphasizes performance precisely where we are interested in it (as dictated by the target audience N and the corresponding number of postal codes B).

The weights introduced here linearly reward relative performance of the models; these are adaptively adjusted by using the performance in the segmentation under consideration (e.g., the top S postal code regime). To be more specific, if S covers p deciles, the weights are derived from lifts generated for p deciles. In contrast, equal weighing is widely used in ensemble averaging (Wolpert, 1992), which does not account for different performance of models. We opted for using reward/penalty of relative performance, for adaptive reasons (discounting models which become poor under selection criteria drift; see the next paragraph for more discussion).

It should be noted that by using normalized lift, we impose a non-negative and sum one condition on the weights with values related to relative (predictive) strength, by construction. Alternative weighting (relaxing non-negative or sum one conditions) could potentially be argued as a viable alternative. Early work on linear stacking (averaging models via weights) found that if one forces non-negative weights and optimizes on training data then, empirically, the sum one condition approximately holds in practice; moreover, if non-negativity was not enforced performance was poor (Breiman, 1996). That work further convincingly argued that the sum one constraint will be approximately enforced as long as some low prediction error models exist in the ensemble. Note that in (Dawes, 1979) it was suggested that non-negativity constraints are required: assuming a model's performance is not anti-correlated with the true behaviour the weight should be non-negative in a linear fit. In general, by imposing a sum one condition on the weights we ensure our final model will also be approximately equal to the expectation of the underlying distribution. The non-negative constraint is self-evident as long as models are not in severe error (anti-correlated), and such models are excluded by construction. Selection based on optimality (e.g., relative predictive strength, as measured by lift here) is likely to have a lesser affect, but can be argued on adaptive grounds: such weightings will adaptively adjust to model performance, so, for example, if the selection criteria drift over time, selection of weights via optimization will downweight models that become ineffective and reward models that show improved performance and will therefore reduce model misspecification risk. Moreover, as we use lift, for various values of N, we adaptively reweight to take relative performance into account.

4.2 Implementation Details

The algorithm was implemented in Python 3.6. We found no implementation or computational issues. The LR model was implemented in SAS 9.4 and again no computational difficulties were encountered. In contrast, implementing the neural network models was somewhat problematic on the Windows laptop used (dual core 2.9 GHz i7, 8 GB Random Access Memory (RAM), Windows 7 Enterprise). Python was used with the Scikit-Learn 0.19.2 package (Pedregosa et al., 2011) for training the MLP and the 1.12.0 TensorFlow package (Abadi et al., 2015) for training the DNN. For the DNN model, due to the large dimension of the training data (760 attributes) and number of nodes, our limited computational resources led to slow training and system stability was compromised to the extent that restarts were necessary. ML libraries often target Graphical Processor Units (GPUs) to allow more efficient computations, and the laptop used lacked both GPUs and adequate RAM. Despite the computational loads stressing our machine, training was successful although moving to larger dimensions (more attributes) or training set sizes would be difficult.

4.3 Potential Extensions

We briefly raise a few items of interest for extending the approach taken:

- We do not perform dimension reduction or any other feature engineering, other than the stepwise reduction used for the LR and MLP; such considerations can improve the performance of the underlying models in the ensemble and additional work in this area could be beneficial.
- There are many variations one can make to our algorithm. For example, instead of finding the intersection in step A1, majority voting can be used to find the agreement set. The model accuracy can be used to determine weights in step A2, etc. The crucial aspects are a winnowing of data to keep the 'top' rated postal codes with a voting between models to ensure enough high value data is considered, and the integrated use of a budget and error consideration when selecting and using this subset of data to determine weights. In addition, the algorithm is generic, in that any number of models can be used. If M, the number of models, is large then the agreement set is expected to be too conservative, and the use of majority voting would become increasingly attractive. This is particularly true if we want to use an ensemble of weak learners.
- In Canada, postal codes are categorized into urban and rural regions. Splitting the data into urban and rural regions may be beneficial. If the number of postal codes in rural regions is small, eliminating them from the original data set may improve performance of the model. If the number of postal codes in rural regions is big enough, developing separate urban and rural models can be another option.
- A different direction for research is to explore the saturation and the frequency of the mailings in a fixed period of time. The final selection N can be adjusted considering these aspects.
- It should be noted that the approach explored here is related to stacked generalization (Wolpert, 1992), which is the generic idea of using model outputs (predicted probability of success here) as features to construct a meta-model. Here we are selecting a linear model corresponding to averaging, with weights found by an algorithm that accounts for model error and a finite N, but other meta-models can be used (for example logistic regression is a reasonable approach, as probabilities will be the output, and neural networks or any other suitable machine learning algorithm can be considered).

5 CONCLUSION

The main contribution of this paper is an algorithm that combines three predictive models (a logistic regression, a multi-layer perceptron and a deep neural network) by assigning weights to the ranks produced by each model. The assignation of the weights is done considering the lift of each individual model. The algorithm is not intended to be static, but rather a dynamic learning approach which improves over time: the lift will be updated after each additional recruiting campaign.

We tested the algorithm on $B = 2000$ postal codes and we identified an overlap of 21,554 postal codes in the first decile. The models architectures are considerably different and in this context the overlap is impressive. Using the lift in the first decile of our predictive models we produced a list of ranked postal codes. This algorithm for combining the models will be validated with the data collected in future recruitment campaigns. The same data will be used for adjusting the lift for each individual model in the ensemble.

REFERENCES

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mane, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viegas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). Tensorflow: Large-scale machine learning on heterogeneous systems. https://www.tensorflow.org. Accessed Feb. 20, 2019.

Baum, E. and Haussler, D. (1989). What size net gives valid generalization? *Neural Computation*, 1(1):151–160.

Breiman, L. (1996). Stacked regressions. *Machine Learning*, 24:49–64.

Chen, X., Ender, P., Mitchell, M., and Wells, C. (2003). Regression with SAS. https://stats.idre.ucla.edu/sas/webbooks/reg/. Accessed Feb. 20, 2019.

Dawes, R. M. (1979). The robust beauty of improper linear models in decision making. *American Psychologist*, 34(7):571–582.

Environics Analytics (2018). Demstats database. https://www. environicsanalytics.com. Accessed Feb. 20, 2019.

Kingma, D. and Ba, J. (2015). Adam: A method for stochastic optimizaton. https://arxiv.org/pdf/1412.6980.pdf. Accessed Feb. 20, 2019.

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

Rumelhart, D. E., Hinton, G., and Williams, R. (1986). Learning representations by back-propagating errors. *Nature*, 323(9):533–536.

Wolpert, D. H. (1992). Stacked generalization. *Neural Networks*, 5:241–259.