A new framework for assessing classification algorithms for mode choice prediction

Tim Hillel[1,2], Michel Bierlaire[2], Mohammed Elshafie[1], and Ying Jin[1]

[1]*University of Cambridge, UK*
[2]*École Polytechnique Fédérale de Lausanne, Switzerland*

1 Introduction

Predicting passenger mode choice is an essential task for transport network simulation and operations management. Machine Learning (ML) classifiers trained on trip diary data are increasingly being investigated as an alternative to Discrete Choice Models (DCMs) for mode choice prediction.

In order to determine the suitability of a particular algorithm for predicting mode choice, it is crucial to be able to reliably quantify its predictive performance for unknown trips. In this paper we propose and test a new framework for performance estimation of classification algorithms.

Firstly, we address three key issues in the literature relating to a) performance metrics for model training, b) sampling methods for model validation, and c) hyperparameter selection for model optimisation. Alternative methods are proposed for each of these issues to form the new framework. The framework is then used to compare the suitability of eight different ML classification algorithms for predicting mode choice. Finally, the implications of the sampling method employed are then investigated experimentally.

2 State of research and proposed framework

There have been several investigations in the literature comparing the suitability of machine learning classifiers for predicting passenger mode choice. For a recent review of the literature, see [1]. Further to the limitations identified in this paper, we identify three linked issues corresponding to three classification stages: 1) model training, 2) model validation, and 3) model optimisation.

2.1 Performance metrics

During *model training*, the expected value of a cost function is minimised over the data. We define a cost function G of the probability distribution generated by the model P for given data $\mathcal{D}$.

$$P^{(*)} = \arg\min_{P} G\left(\mathcal{D}; P\right). \tag{1}$$

G represents an aggregate performance metric which provides a numerical representation of the predictive power of P when used to predict mode choice for new data.

Classification accuracy is predominantly used as the performance metric within the literature. There are a number of issues with the use of accuracy as a performance metric.

Primarily, as the prediction is assigned to the mode with the highest probability, mode choice is treated as a deterministic instead of a stochastic process. As such, it is assumed that mode choice is constant under the same set of conditions and socio-economic characteristics. In reality, passengers have a degree of intra-heterogeneity, and as such their choice is drawn randomly from a probability distribution. This distribution is the *true model*, which we aim to replicate with the classification model. In order to account for this stochastic heterogeneity in simulation, the predicted mode choice must be drawn from a probability distribution. The metric used to assess model performance should therefore represent how well the predicted probability distributions fit the data.

Additionally, accuracy is a poor metric for assessing performance with imbalanced datasets. Trip diaries, where trips by certain modes may occur much more frequently than others, are often highly imbalanced datasets. As such, a trivial classifier can achieve high accuracy by predicting the most likely mode. This is known as the *accuracy paradox*.

Instead, we use the average natural logarithm of the likelihood $\mathcal{L}$ (also known as log-loss or cross entropy loss). For $n = 1, .., N$ trips with corresponding *ground truth* mode choice i_n and feature vector x_n, this is defined as

$$G_{\text{log-loss}} = \mathcal{L} = -\frac{1}{N} \sum_{n=1}^{N} \ln P\left(i_n | x_n\right). \tag{2}$$

where $P\left(i_n | x_n\right)$ is the probability predicted by the model for the ground truth. A smaller log-loss represents a better model fit.

2.2 Sampling methods

During *model validation* the predictive performance of a model is estimated. This is given by the expected value of the cost function evaluated over an unknown set of journeys $\mathcal{U}$

$$\mathbb{E}\left[G\left(\mathcal{U}; P^{(*)}\right)\right]. \tag{3}$$

As $\mathcal{U}$ is unknown, the value of (3) can only be estimated. This is achieved by *validating* the model on data unseen during model training.

In holdout validation the dataset $\mathcal{D}$ is divided into two subsets, a training set $\mathcal{T}$, and a validation set $\mathcal{V}$. The model is trained on T in order to minimise the cost function. From (1)

$$P^{(*)} = \underset{P}{\arg\min}\, G\left(\mathcal{T}; P\right). \tag{4}$$

The predictive performance of the model can be estimated by calculating the performance metric on $\mathcal{V}$

$$\mathbb{E}\left[G\left(\mathcal{U}; P^{(*)}\right)\right] \approx G\left(\mathcal{V}; P^{(*)}\right). \tag{5}$$

In k-fold cross-validation, $\mathcal{D}$ is divided into k approximately equally sized validation folds $\mathcal{V}_i$ with corresponding training sets $\mathcal{T}_i$. These folds allow for k iterations of the model to be trained and validated, for k separate estimates of model performance

$$P_i^{(*)} = \underset{P_i}{\arg\min}\, G\left(\mathcal{T}_i; P_i\right), \tag{6}$$

$$\mathbb{E}\left[G\left(\mathcal{U}; P^{(*)}\right)\right] \approx G\left(\mathcal{V}_i; P_i^{(*)}\right), \tag{7}$$

where $P^{(*)}$ is the model trained on the full dataset $\mathcal{D}$. As each sample $n \in \mathcal{D}$ is predicted exactly one time by the different models P_i, we can define a single performance estimate by calculating G for the combined results of the k models

$$\mathbb{E}\left[G\left(\mathcal{U}; P^{(*)}\right)\right] \approx G\left(\mathcal{V}_1, ..., \mathcal{V}_k; P_1^{(*)}, ..., P_k^{(*)}\right). \tag{8}$$

In order for (5) and (8) to be valid estimates of model performance on new data, the correlation between $\mathcal{T}$ and $\mathcal{V}$ (or $\mathcal{T}_i$ and $\mathcal{V}_i$) must be equivalent to the correlation between $\mathcal{D}$ and $\mathcal{U}$.

Datasets used in the literature are predominantly formed from trip diaries of respondents (individuals and/or households). The respondents are sampled randomly from the population. The predominant approach employed is to assume the trips (elements) in $\mathcal{D}$ are independent and identically distributed (i.i.d.), and therefore to sample trips in $\mathcal{T}$ and $\mathcal{V}$ randomly from $\mathcal{D}$. We refer to this as *trip-wise* sampling.

Whilst the survey respondents are sampled randomly from the population, each respondent provides multiple trips in the trip diary. A simple thought experiment considering an example daily trip itinerary demonstrates that the trips made by the same individual are not distributed i.i.d. For example, return journeys are highly correlated to their corresponding outbound journey, both in terms of the feature vector x_n, and the mode taken y_n.

When sampling trip-wise from $\mathcal{D}$, trips made by the same individual or household can be split across both $\mathcal{T}$ and $\mathcal{V}$. As such, this introduces correlations between $\mathcal{T}$ and $\mathcal{V}$ which will not be present between $\mathcal{T}$ and $\mathcal{U}$. This will result in $G(\mathcal{V}; P^{(*)})$ overestimating the model performance on unknown data $\mathbb{E}\left[G\left(\mathcal{U}; P^{(*)}\right)\right]$.

We propose *household-wise* sampling as an alternative to trip-wise sampling. In this sampling method the trips are sampled grouped by household. This prevents trips made by the same household from appearing in both the training and validation data, and as such ensures no additional correlations are introduced between $\mathcal{T}$ and $\mathcal{V}$.

2.3 Hyperparmeter selection

A trained classification model $P^{(*)}$ is an instance of a classification algorithm $\mathcal{A}$ fitted to training data $\mathcal{T}$. Each algorithm $\mathcal{A}$ has a set of *hyperparameters* λ which control how the model fits to the data during (1) (*model training*).

$$P^{(*)} = \mathcal{A}_\lambda(\mathcal{T}). \tag{9}$$

Model performance is highly dependent on chosen hyperparameter values λ. To enable fair comparison between algorithms, *model optimisation* should be performed to select values of lambda which minimise the cost function of the trained model. We want to select optimal hyperparameter values $\lambda^{(*)}$ so that

$$\lambda^{(*)} = \arg\min_{\lambda} \mathbb{E}\left[G\left(\mathcal{U}; \mathcal{A}_\lambda(\mathcal{T})\right)\right], \tag{10}$$

$$= \arg\min_{\lambda} \mathbb{E}\left[G\left(\mathcal{U}; \arg\min_{P} G\left(\mathcal{T}; P\right)\right)\right]. \tag{11}$$

Optimal hyperparameter value selection (11) contains two optimisation loops. Due to the embedded inner loop, it is difficult to perform this optimisation using gradient decent methods.

The typical approach within the literature is to use grid search over a predefined set of candidate values. Grid search has been shown to perform poorly in practice at finding optimal hyperparameter values [2].

Instead, we propose using Sequential Model-Based Optimisation (SMBO) to select hyperparameter values. This is where hyperparameter values are drawn from pre-defined probability distributions, with subsequent draws learning from previous trials. SMBO has been shown to significantly outperform grid search on real world examples [3].

3 Experimental methodology

We use three years of a dataset of recorded trips alongside details of the alternative trips for each mode (walking, cycling, public transport, and driving). The dataset is derived from the trip diary data from London Travel Demand Survey (see [4] for details). The first two years of data (2012/13 and 2013/14) are used for model optimisation and performance estimation. The final year of data (2014/15) is used as a holdout test set to assess actual model performance for predicting a future year. This represents a realistic transport simulation task.

We compare the performance of eight machine learning classification algorithms:

1. Artificial Neural Network (ANN)

2. Extra Trees (ET)

3. Extreme Gradient Boosting (XGB)

4. K-Nearest Neighbours (KNN)
5. Logistic Regression (LR)
6. Multinomial Naive Bayes (MNB)
7. Random Forest (RF)
8. Support Vector Machine (SVM)

We optimise two sets of model hyperparameters for each algorithm, one using trip-wise sampling, and the other using household-wise sampling. We optimise the hyperparameters using 100 iterations of SMBO, with 10-fold cross validation used to estimate model performance during optimisation.

We use 10-fold cross validation to estimate the performance for each model (two for each algorithm). The validation folds are sampled using the corresponding sampling method to the hyperparameter optimisation.

Finally, we train two models for each algorithm on all of the validation data (2012/13-13/14), one using the trip-wise optimal hyperparameters, and the other using the household-wise hyperparameters. We then evaluate the real world performance of each model on the holdout set (2014/15). The cross-validation estimates of performance are then compared to the real world performance for each model.

4 Results & discussion

Model	**Trip wise**	**Household wise**	**Diff.**
Artificial Neural Network	0.6739	0.6783	0.0044
Extra Trees	0.4782	0.6571	0.1789
Extreme Gradient Boosting	0.4665	**0.6338**	0.1673
K-Nearest Neighbours	0.7403	0.9713	0.2310
Logistic Regression	0.6770	0.6795	0.0025
Multinomial Naïve Bayes	0.8626	0.8638	0.0012
Random Forest	0.4930	0.6578	0.1648
Support Vector Classifier	0.6248	0.6571	0.0322

Table 1: Performance estimates for each algorithm for each sampling method. Given value is log-loss evaluated across 10 validation folds for models optimised using trip-wise and household-wise sampling. Validation folds are generated using the corresponding sampling method to that used in optimisation

Model	**Trip wise**	**Household wise**	**Diff.**
Artificial Neural Network	0.7018	0.6941	-0.0077
Extra Trees	0.6961	0.6791	-0.0170
Extreme Gradient Boosting	0.7259	**0.6510**	-0.0749
K-Nearest Neighbours	1.0171	1.0397	0.0227
Logistic Regression	0.6935	0.6930	-0.0005
Multinomial Naïve Bayes	0.8793	0.8766	-0.0027
Random Forest	0.6832	0.6774	-0.0058
Support Vector Classifier	0.7316	0.6702	-0.0614

Table 2: Test scores for each model optimised for each sampling method. Given value is log-loss evaluated across holdout data for models optimised using trip-wise and household-wise sampling, trained on all validation data (2012/13-13/14) and tested the holdout data (2014/15).

Model	**Trip wise**	**Household wise**
Artificial Neural Network	0.0279	0.0158
Extra Trees	0.2179	0.0220
Extreme Gradient Boosting	0.2594	0.0172
K-Nearest Neighbours	0.2767	0.0684
Logistic Regression	0.0165	0.0135
Multinomial Naïve Bayes	0.0167	0.0128
Random Forest	0.1902	0.0196
Support Vector Classifier	0.1067	0.0132

Table 3: Difference between predicted (cross-validation) performance and test performance for each model

As shown in Table 1, trip-wise sampling consistently predicts higher performance than household-wise sampling. The difference between the estimates is lowest for MNB, LR and ANN models, and greatest for KNN and ensembles of trees classifiers (ET, XGB and RF). XGB shows the highest predicted performance for both trip-wise and household-wise sampling.

Table 2 shows that the XGB household-wise optimised model achieves the highest real world test performance. Except for KNN, where both models performed comparatively poorly, models with hyper-parameters optimised for household wise sampling show consistently higher real world test performance. This demonstrates that the models optimised for trip-wise sampling have overfit to correlations between validation folds which are not present in the real world data.

Table 3 shows that that household-wise sampling cross-validation provides a much more consistent estimate of real-world performance than trip-wise cross-validation. Cross-validation log-loss provides an over-estimate of real-world test performance for all models. However, the predicted log-loss for household-wise sampling cross-validation is consistently much closer to the test value. Additionally, differences between cross-validation score and test score are much more consistent across algorithms for household-wise sampling than trip-wise sampling.

References

[1] J. Hagenauer and M. Helbich. A comparative study of machine learning classifiers for modeling travel mode choice. *Expert Systems with Applications*, 78:273–282, 2017.

[2] J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, 13(Feb):281–305, 2012.

[3] J. Bergstra, B. Komer, C. Eliasmith, D. Yamins, and D. D. Cox. Hyperopt: A Python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, 8(1): 014008, 2015.

[4] T. Hillel, M. Z. E. Elshafie, and Y. Jin. Recreating passenger mode choice-sets for transport simulation. *Smart Infrastructure and Construction*, In press.