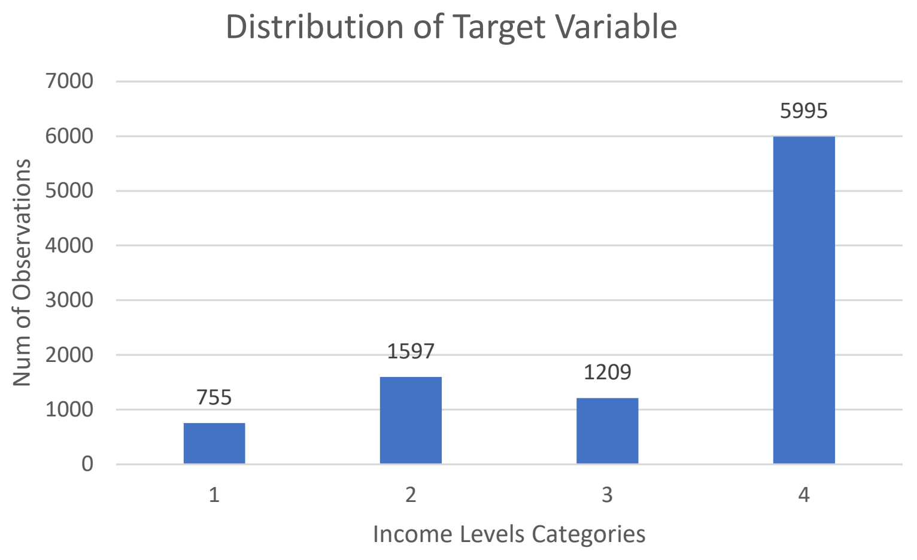
Using Naïve Bayes and Classification Ensembles to predict income levels from the Costa Rica Household Poverty Survey

Description of the problem and motivation

This project aims to compare and contrast the performance of two machine learning algorithms on a supervised, multi class classification problem, aimed at accurately predicting income levels. The Inter-American Development Bank (IDB) uses the Proxy Means Test (PMT) to model a household's observable attributes in order to classify and predict household income levels. IDB established a Kaggle Competition to improve their algorithm; providing labelled data from a household survey in Costa Rica.

Analysis of the data set and basic statistics

- Following initial data cleaning to remove observation IDs, and variables with zero variance, the dataset provided by the Kaggle competition contained 9557 observations and 137 variables.
- Each variable represents responses to survey questions asked to each of the respondents represented by the observations. The high dimensionality of the dataset is noted as a concern.
- The dataset contains both categorical and continuous variables. The categorical data variables are represented using one hot encoding (values of 0 or 1). The categorical variables in the dataset were specifically set to categorical arrays in MATLAB to ensure the MATLAB algorithms modelled these appropriately.
- The target variable is a categorical variable representing income levels on a scale of 1-4 : with 1 = extreme poverty, 2 = moderate poverty, 3 = vulnerable, 4 = non-vulnerable.
- The dataset is highly unbalanced; meaning there were significantly more observations classed as income level 4 than the other targets, as shown in in the chart (left).



Summary of the selected Machine Learning Models

Naïve Bayes

Pros

- Naïve Bayes is a simple, fast and model, shown to have good classification accuracy (5).
- The Naïve Bayes model requires less training data than others to work effectively (6).
- The assumption of independence means that Naïve Bayes can work on data that is both categorical and continuous (3).
- For the same reason, Naïve Bayes can handle high dimensional data well (3, 6).

Cons

- Naïve Bayes assumes that all variables are independent, given the class variable (1). We know that this is not the case with the variables in the Costa Rica Household Survey Dataset. Despite this assumption rarely being true, the performance of Naïve Bayes is considered surprising (1).

Classification Ensembles

Pros

- Random forests ("bagging") faster to train the bagging and boosting methodologies such as Adaboost ("Adaboost M2") and RUSboost (6,7,9).
- Random forests more robust to outliers and noise than boosting methods(7).
- Provides ranking of importance of features that aids interpretability(7).
- Uses out of bag estimates to monitor and provide statistics such as internal estimates of error, strength, correlation and variable importance(7).
- Simple and easily parallelized(7).

Cons

- Outperformed by other Large number of trees can make the algorithm slow and ineffective for real-time predictions(7).
- Less accurate than algorithms that use adaptive reweighting as they build the ensemble(7).
- Relies on law of large numbers to not overfit. If sample size small then may overfit. (7).

Hypothesis

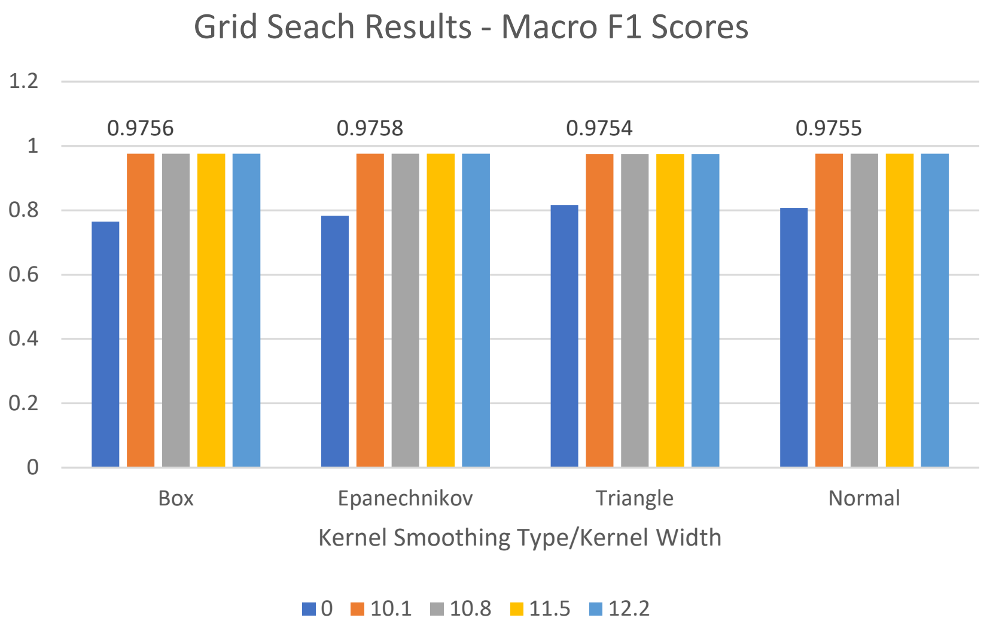
Based on our research on the algorithms, we are anticipating that Naïve Bayes will perform better than random forests, and other variations of classification ensembles, in the task of determining the income level for unseen observations in the Costa Rica Household Survey Dataset. This is due to its ability to work effectively on small datasets, and high dimensional data. Additionally, we are proposing that the high dimensionality of the dataset does not contribute to improve prediction, and that both algorithms, Naïve Bayes and Random classification ensembles, will perform with similar accuracy after the dimensionality is reduce using the principle component analysis method. We also want to show that boosting methods such as random under sampling boosting ("RUSboost" method) and adaptive boosting ("AdaboostM2") will output perform traditional random forests ("bagging"). The adaptive boosting and random under sampling methods are used to help alleviate the cons of random forests.

Description of choice of training and evaluation methodology

- The selected algorithms; Naïve Bayes and set of classification ensemble methods will be used to developed models from the data, and to tune the hyper parameters. Once the best performing hyper parameters have been chosen, the models will be tested on unseen test data.
- The models will be compared using the macro F1 score. F1 scores is combines measures of precision (of the predicted class values, how many were correct) and recall (of the actual classes, how many were correctly identified). For multi class classification, the macro F1 score averages the F1 score of each class. The macro F1 score is a useful measure for assessing the viability of unbalanced datasets.
- Once the best model for each algorithm has been determined, an additional test will be conducted to determine the impact of a principle component analysis. The principle component analysis will aim to reduce the dimensionality of the dataset. The resulting performance of the model on the reduced dataset will be compared to the performance against the full dataset.
- The dataset will be split into 80% training data and 20% test data.

Choice of parameters and experimental results

- In order to establish a baseline for hyperparameter tuning, a basic Naïve Bayes model with default hyperparameter was trained. Using K=5-fold validation, the model resulted in a macro F1 score of 0.9082.
- Following this, a grid search was performed to determine the optimum hyperparameters for the model. The grid search compared variations to the probability density functions, and modifications to the kernel widths.
- The probability density function distributions for categorical variables were set to 'mvmn' (multivariate multinomial distribution). These distributions weren't modified during the grid search.
- The continuous variables were modified; with the grid search testing the density functions for the 'box','epanechnikov', 'triangle', and 'normal' kernel distributions, against the following kernel widths; none/default, 10.1, 10.8, 11.5, 12.2. All tests were validated using K=5-fold validation. The resulting macro F1 scores are charted below.
- Based on the grid search above, a new model was trained on the full set of data, using the highest performing hyperparameters (pdf = 'triangle', kernel width = 10.1). This model was tested on unseen test data. The resulting F1 score was 0.9814.
- This model was then retrained with a dimensionally reduced dataset, following a principle component analysis (PCA). The PCA reduced the dataset from 137 variables to 117. The resulting macro F1 score was 0.9741.
- As an alternative to the PCA; feature selection was tested. A basic decision tree model was developed. Using the 'Predictor Importance' variable (which estimates a value of the importance of each individual predictor, based on summing the changes in the mean squared error after each split in the tree (6)), the number of features was reduced based on the highest predictor value. The model was tested on two sets of data; one containing only the top 50 predictors, and one containing only the top 80 predictors. The resulting model performed much worse that previous models; with macro F1 scores of 0.4514 and 0.4864 respectively.



- For the classification ensemble subset of models we chose to investigate three ensemble algorithms covering random forests ('bag'), adaptive boosting ('AdaboostM2'), and random under sampling boosting ('RUSboost'). The objective was to find the "best" combination of algorithm and their hyperparameter settings.
- We conducted a grid search on our random forest models using k-fold cross validation with 5 folds. The objective function we minimized was cross validation loss; cross-validation and cross-validation loss provides a method and measure "to test the model's ability to predict new data that was not used in estimating it, in order to flag problems like overfitting or selection bias"(8)
- In our search we varied method covering 'bag', 'adaboostm2' and 'RUSboost', the learn rate for boosting algorithms adaboost and RUSboost , the number of trees and the minimum number of observations per leaf for each classification ensemble.
- We found that the best number of learners to use for random forests and AdaboostM2 was 500 learners, and 188 for RUSboost. The best leaf minimum leaf size is 1.
- The best performing algorithm in the experimental stage is "bagging" random forests as specified by Breiman (7).
- We noted that using PCA in our feature selection led to accuracy falls in both F1 macro score, and classification accuracy (correctly identified observations / total observations in train). We expect this to apply to final models.
- We expected Adaboostm2 and RUSboost to have scored higher F1 scores.

Parameter Options	
Parameter	Parameter values
Train Cycles	10, 27, 71, 188, 500
Leaf Size	1,8
Learn Rate	0.001. 0.0056243, 0.031623, 0.17783, 1

Results from applying models to test holdout set

	Max Splits	mdl Training Cyles	Min Leaf Size	Learn Rate	Classification Accuracy	Result F1	PCA F1	Change
Bag	9556	500	1		0.946	0.842	0.6893	-0.1527
AdaBoostM2	20	500	1	1	0.936	0.931	0.425	-0.5058
RUSBoost	20	188	1	1	0.65	0.703	0.6148	-0.887

Experimental Results

	Learning Cyles	Leaf Size	Learn Rate	Classification Accuracy	Model F1 Score	Model F1 Score PCA	F1 Accuracy Change PCA
Bag	500	1		0.873	0.977	0.926	-0.051
AdaBoostM2	500	1	1	0.738	0.641	0.445	-0.1936
RUSBoost	188	1	1	0.645	0.738	0.655	-0.083

Analysis and critical evaluation of results

- The simplicity and effectiveness of the Naive Bayes model held true during tests of the model, as did its ability to cope with high dimensionality.
- The results of the grid search were an improvement on the initial default model; however variation across the grid search was minimal. This was likely due to the fact that only 38 continuous variables of the 137 variables in the dataset were modified during the search, with the remaining categorical variables set to 'mvmn' (multivariate multinomial distribution).
- However, tuning the hyperparameters resulted in a model with a macro F1 score of 0.9758 (an average score based on k fold = 5 validation).
- When the model was applied to unseen test data, the model performed with a similar score to that achieved in the training data validation, suggesting that the model generalized well.
- The process of feature reduction, using PCA, demonstrated that some of the dimensionality of the Naïve Bayes model could be reduced, without a performance loss. Following the PCA process, the number of features was reduced from 137 to 117, with little change to the macro F1 score.
- Feature reduction was anticipated to perform better than it did. Further work would need to be undertaken to understand why this was the case.
- Using the final models to predict the outcome for the train data set the AdaboostM2 algorithm got the highest macro F1 score of the classification ensemble algorithms scoring getting a score of 0.931; this performance is an increase of almost 0.20 from the 0.733 scored in the train data set; this suggests that the model generalizes well. However this could have been the result of the train and test data sets being too small.
- The bagging approach that scored 0.842 F1 on the train set, higher than the 0.977 scored on the train data set. This fall in performance suggests that the model overfits the data. However this may be a result of overfitting the hyperparameters of the model or the data set having too few observations.
- RUSboost scored the lowest with an F1 accuracy of 0.703. However it's fall in performance from the experimental stage is lower than bagging. This result is particularly surprising because the literature suggests that RUSboost should perform better than other tree ensemble methods where the classes are unbalanced (9), as is the case in this project.
- Using PCA on our feature reduction led to big decreases in performance. Further work required to understand why this is the case, and to apply more careful feature reduction.

Lessons learned and future work

- Initial testing of the models confirmed the importance of exploratory data analysis, and data cleaning. For example, correctly setting the one hot encoded categorical variables as categorical arrays in MATLAB was the most effective change made to ensure that the models predicted accurately. Prior to implementing this process, basic models were performing with macro F1 scores of approximately 0.4400, increasing to over 0.8000 after this change.
- The experiment showed that both the Naïve Bayes algorithm and classification ensembles would be suitable choices for the Inter-American Development Bank in developing a model for the Proxy Means test.
- For the classification ensembles we note that we did not optimize all available model hyperparameters for the bagging and RUSboost methods. For example we did not optimize the number of observations or leaners considered in each learner for bagging, not did we optimize the parameter 'RatioToSmallest' that governs the amount to under/oversample from each class. Doing this in the future could lead to better performing methods that could increase performance of bagging and RUSboost methods. Further work here could enhance the performance of classification ensemble methods.
- Both models were able to achieve good macro F1 scores, despite the initial concerns related to the high dimensionality of the dataset, and the unbalanced nature of the target category. Future work could focus on investigating the extent of the impact of these factors on model performance, and the effectiveness of methods to reduce these factors; including feature selection to further reduce the dimensionality, and SMOTE or ADASYN algorithms to balance the dataset.
- Future work to determine a method by which feature selection could perform at levels similar to the full data set has the potential to reduce the amount of data required by the bank to determine household income levels. This could reduce the Bank's operational costs.

References:

1) Zhang, H. (2004). The optimality of naive Bayes. AA, 1(2), 3.
2) Rish, I. (2001, August). An empirical study of the naive Bayes classifier. In IJCAI 2001 workshop on empirical methods in artificial intelligence (Vol. 3, No. 22, pp. 41-46). New York: IBM.
3) Bishop, C. M. (2006). Pattern recognition and machine learning (information science and statistics) springer-verlag new york. Inc. Secaucus, NJ, USA.
4) Zhang, D., Wang, J., & Zhao, X. (2015, September). Estimating the uncertainty of average F1 scores. In Proceedings of the 2015 International Conference on The Theory of Information Retrieval (pp. 317-320). ACM.
5) Langesth, H & Nielsen, T.D. Machine Learning (2006) 63:135.
6) Mathworks. (2018). Matlab Support files. Retrieved November 2018 from www.mathworks.com/help/
7) Breiman, L. Machine Learning (2001) 45:5
8) Cawley, Gavin C.; Talbot, Nicola L. C. (2010). "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation" (PDF). 11. Journal of Machine Learning Research: 2079–2107.
9) Seiffert, C., T. Khoshgoftaar, J. Hulse, and A. Napolitano. RUSBoost: Improving clasification performance when training data is skewed. 19th International Conference on Pattern Recognition, 2008, pp. 1–4.
10) Freund, Y. and R. E. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. J. of Computer and System Sciences, Vol. 55, 1997, pp. 119–139.

Authors: Matthew Stewart and Courtney Irwin