Comparison of Naïve Bayes and Ensemble Boosted Trees Applied to Loan Defaults Dataset

Tareq Galala, Sahil Dhingra

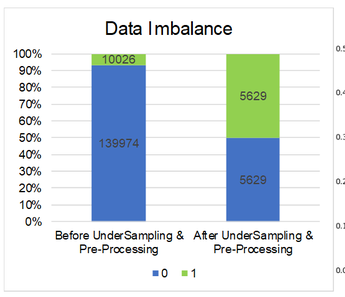
Introduction and Motivation

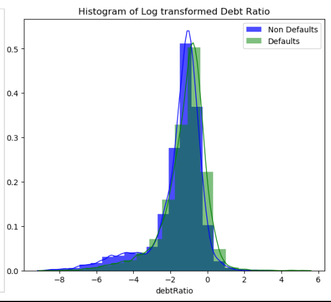
- Primary objective is to compare and contrast the performance of two Machine learning models—Naïve Bayes and Ensemble Boosted Trees.
- We have selected a binary classification problem where we aim to predict whether an individual will default on their loan obligations given a set of features.

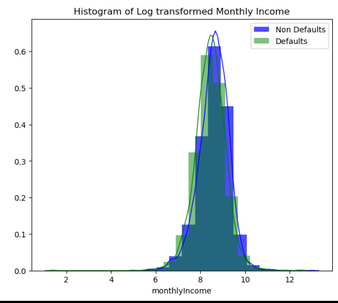
Dataset and Exploratory Analysis

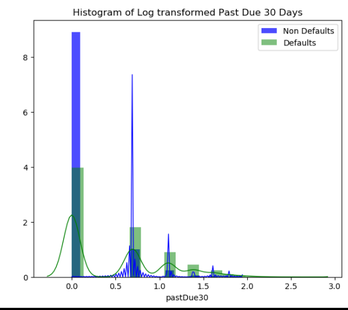
- Dataset: Give Me Some Credit from Kaggle's 2012 competition.
- The dataset contains 10 features, which are all continuous variables and 1 target variable 'Serious Delinquency in the past 2 years', which contains binary values—1 for default and 0 otherwise.
- The dataset contains 150k observations and is highly imbalanced with defaulting observations only approx. 6% of the entire dataset. After down sampling our data, our dataset reduced to 11k observations with equal number of observations in each target class.
- We present the descriptive statistics for defaulted observations and non-defaulted observations in the table. We observe that all features exhibit very high level of Kurtosis implying significant numbers of outliers. All features are also exhibited high skewness levels.
- After pre-processing, both skewness and kurtosis were significantly reduced but did not disappear completely—this is understandable as all features have continuous values and will therefore have significant variability in value range.
- Initially certain features such as loans past due 30 days, past due 60 days and past due 90 days showed high correlation with each other, however, interestingly, the correlation was significantly reduced after pre-processing the data.
- We also present the distribution of a few features after log-normalising the data. We can see that features follow different distributions with significant presence of outliers.

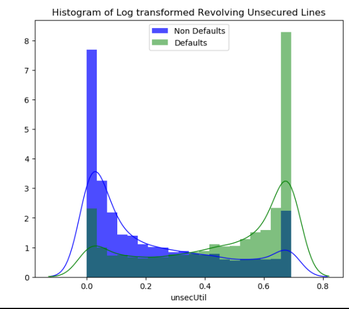
Defaults in the last two years	Count		Age	Debt Ratio	Monthly Income	# of times loans past due 30 days	# of times loans Past Due 60 days	# of times loans Past Due 90 days	# of Credit Lines	Loan Utilisation	# of Real Estate Loans	# of De-pendents
No Default	139970	mean	52.75	357.15	6747.84	0.28	0.13	0.14	8.49	6.17	1.02	0.74
		skewness	0.16	95.27	112.11	31.83	33.46	33.18	1.22	96.57	3.35	1.61
		kurtosis	2.49	13474.50	18635.96	1054.42	1128.62	1115.66	6.09	14083.25	65.67	6.18
Defaulted	10026	mean	45.93	295.12	5630.83	2.39	1.83	2.09	7.88	4.37	0.99	0.95
		skewness	0.46	11.11	16.63	7.92	8.02	7.93	1.19	44.92	4.13	1.27
		kurtosis	3.02	222.71	550.57	64.49	65.60	64.67	6.11	2315.17	43.24	4.24











Summary of choice of Machine Learning Algorithms used

Ensemble Boosted Trees

- This supervised machine learning model combines a large number of decision trees operating collectively as a group or 'Ensemble'. The main principle behind the ensemble methods is that a group of weak learners come together to form a strong learner.
- Boosting algorithms suffer from the over-fitting problem when dealing with very noisy data [1]. To cope with this situation, Friedman et al. suggest the use of LogitBoost, which could greatly reduce training errors and hence yield better generalization. [2]

Pros

- Boosting is combines many weak learners (trees) into a strong classifier.
- Robust to overfitting.

Cons

- Requires more and careful tuning of many hyper-parameters.
- Lack of interpretability.

Naïve Bayes

- Naïve Bayes (NB) is a supervised learning algorithm which performs well on classification tasks. NB is based on Bayes Theorem but 'naively' assumes independence between features given the class variable.
- NB estimates the probabilities of the target variable and each feature given the target variable by estimating their frequencies over the training data. The set of these estimates form the learned hypothesis which is used to classify each observation. [3]

Pros

- It's simple to understand and quick to predict the parameters required for classification.
- Effective on small on of data and can work with missing feature values.

Cons

- NB does not take into account the interactions between various features although this is more problematic for regression tasks than classification tasks.
- The algorithm may not perform well if the data is skewed leading to incorrect Prior value estimates.

Hypothesis Statement

- Both models are expected to have high prediction accuracy, however, empirical evidence suggests Ensemble Boosted Trees have superior prediction accuracy compared to Naïve Bayes. [4]
- The highest performing models in the Kaggle competition used a combination of techniques including Boosted Trees while none of the top performers used Naïve Bayes. [5]
- Therefore, we aim to understand the difference in overall performance of Boosted Trees over Naïve Bayes when taking in to account both performance accuracy and speed.

Training and Evaluation Methodology

- The dataset initially had 150k observations which were reduced to approx. 114k after data pre-processing
- As we have a large number of observations, we chose the under sampling method to deal with the issue of imbalance in our target variable.
- We experimented with ADASYN oversampling but could not proceed with this approach as the computational resource requirements of training our model on over 220k observations made the hyper parameter tuning process very difficult and time-consuming.
- We then reshuffled our data and used the holdout method to allocate 80% data to training and holding 20% for testing and used 10 fold cross validation to train and optimise our models for both algorithms .

Choice of Hyper-parameters and experimental results

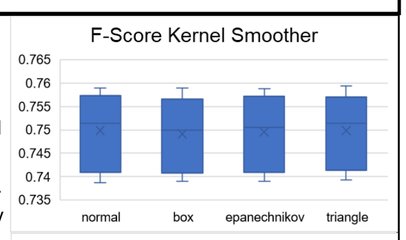
Ensemble Boosted Trees

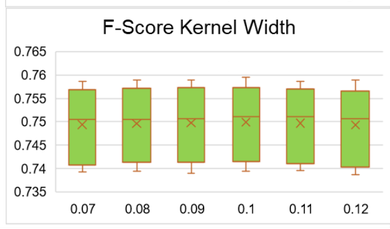
- The optimized method used in our ensemble is LogitBoost, which performed better than AdaBoost, RUSBoost, GentleBoost and Bagging (Random Forest) methods. This boosting algorithm is based on the observation that AdaBoost is in essence fitting an additive logistic regression. [6]
- A grid search was used with 10-fold cross-validation to tune the choice of hyperparameters - number of trees, maximal number of decision splits, minimum observations per leaf, learning rate & the number of bins.
- The optimal number of trees was consistent at 100 trees.
- It is worth noting that the number of bins not only decreased the training time to half but also increased accuracy slightly.
- Our choice of metric decision was F-score and AUC, we picked the highest score in more than 6000 combinations in our grid search, which was computationally expensive and time consuming.
- Learning rate, makes the boosting process more conservative and robust to over-fitting. Although the empirical results of Friedman (2001) have revealed that small value of learning rate < 0.1 leads to better generalization error [7], however, we our finding of 0.5 gave us the best generalization error.

Hyper-parameters	Optimal Value
MaxNumSplits	3
MinLeafSize	13
NumLearningCycles	100
LearnRate	0.5
NumBins	110

Naïve Bayes

- We ran experimental results on four hyperparameters—type of distribution, type of kernel smoother, kernel width and Prior estimates.
- We ran 10 iterations of 10-fold cross-validation in order to eliminate any effect generated by splitting dataset randomly. [8]
- We observed that the Kernel distribution performed much better than Gaussian and specifically the Triangle Kernel Smoother gave best results on average. This is supported by Liu et al who compared the various kernel functions on several UCI datasets. [8]
- We compared model performance on a narrow set of kernel widths (0.08—1.12) after initially investigating a wider range (0.1 - 0.9). Kernel width of 0.1 gave the best results.
- We compared several prior values including the sample prior. In our case we have domain knowledge through under sampling that the prior values are equal and thus the sample prior gave the best results in training. However, we also know that our testing data is unbalanced therefore, equal prior values may not give the best test results.

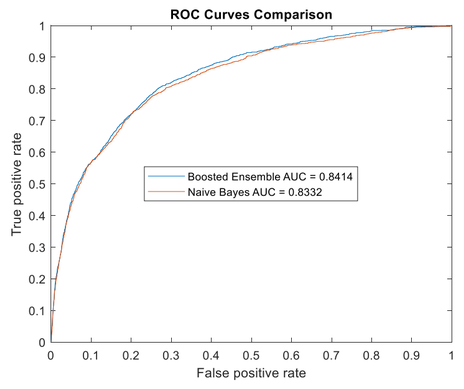


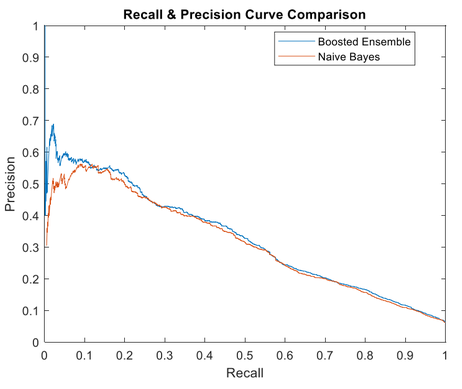


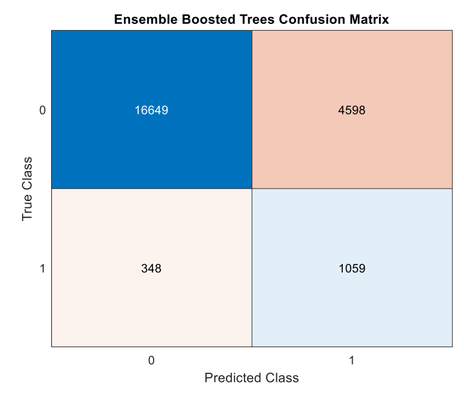
Analysis and Critical Evaluation of Results

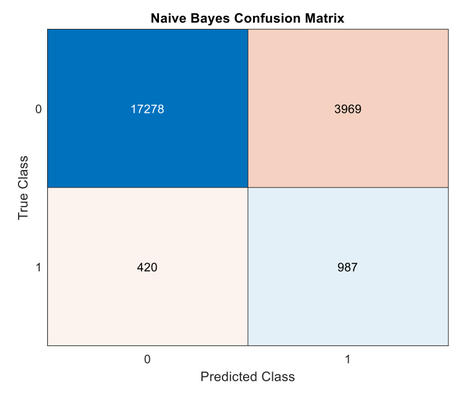
- We present and compare our results on F-Score values to explain the performance of our algorithms rather than just Accuracy. Our objective is to mainly predict defaults, which is measured by the Precision of the algorithm whereas Recall measures the performance of the algorithm in correctly classifying non-defaults. For our dataset it is important to ensure balance between predicting defaults and non defaults and this is measured by the F-Scores.
- We find that the basic Gaussian Naïve Bayes model performed very well at the training stage without any optimization (F-Score 0.7397 and AUC 0.7222), this is one of the strengths of this algorithm. However, we were able to significantly improve the model performance during training to increase F-Score to 0.7571 and AUC to 0.7656 by using kernel distribution and tuning its hyperparameters.
- The basic Classification Tree did not perform well during training (F-Score 0.6843 and AUC of 0.6833) but Ensemble Boosted Trees showed significant improvement in training to increase F-Score to 0.7593 and AUC to 0.7624.
- At the testing stage, we saw a significant reduction in F-Scores for both models. NB model produced F-Score of 0.1992 vs 0.1832 for Ensemble Boosted model. This was due to poor Precision scores implying that our models classified several non-defaults as defaults. We can see from the confusion matrix that False Positives significantly outweighed True Positives.
- The model performance could have been impacted by a number of reasons:
  - Firstly, our data sampling method—under sampling, significantly reduced the size of our dataset which could have reduced the Precision score. Other sampling methods such as SMOTE oversampling methods could have given better results. [9]
  - Secondly, there are model specific factors which may be impacting performance—for the Naïve Bayes model, the performance was impacted by the choice of priors, our training model chose equal prior values which worked well on training data with equal ratio of target classes however it could not perform as well on the imbalanced test data where the prior values were considerably different. It was difficult to understand the reason for the poor Precision performance of the Boosted Tree models and more work needs to be done in future to understand its results better.
- We were surprised by how close the results of both models were especially as we expected Boosted Trees to significantly outperform Naïve Bayes. [4]
- The Ensemble Boosted Model took on average four times longer to run than Naïve Bayes. On an Intel i5-7500 CPU@3.4Ghz with 16GB RAM, the Naïve Bayes model took 0.511 secs to run the test with approx. 22k observations while Boosted Trees took 2.24 secs.
- Therefore, in our experiments, the Naïve Bayes model performed much better than Boosted trees especially after taking into account computational requirements and speed.

Boosted Trees		Measures	Naïve Bayes	
Train	Test		Train	Test
0.7593	0.2941	**F-Score**	0.7594	0.3102
0.7624	0.8408	**AUC**	0.7577	0.8332
0.7695	0.1832	**Precision**	0.7611	0.1992
0.7493	0.7448	**Recall**	0.7577	0.7015
3.5420	2.24	**Time**	0.6148	0.511









Lessons for Future Work

The are a number of areas where we could improve our results:

- Feature selection: Using either backward elimination or forward selection wrapper methods could improve model performance
- Boosting-based feature combination: In this method, at each round, some weak classifiers are built on different feature sets, these classifiers are then combined by weighted voting to create features which are able to produce good results. [10]
- Feature engineering such as net income after expenses were used by winners of the Kaggle competition to improve results. [5]
- Using other methods of dealing with Class imbalance such as resampling by oversampling or creating synthetic minority classes using SMOTE.
- Adding noise to the dataset to test prediction accuracy.
- We could improve Precision by changing the Cost Loss to penalise False Positives more.

References

[1] Sotiris Kotsiantis: Rotation forest with LogitBoost, International Journal of Innovative Computing, Information and Control, Volume 9, Number 3, March 2013.

[2] J. Friedman, T. Hastie and R. Tibshirani, Additive logistic regression: A statistical view of boosting, The Annals of Statistics, vol.28, 2000.

[3] T. M. Mitchell, Machine learning, McGraw Hill, 1997.

[4] Bauer, Eric; Kohavi, Ron: An Empirical Comparison of Voting Classification Algorithms: Bagging, Boosting, and Variants, Machine Learning, Volume 36, Issue 1, 07/1999.

[5] Sharma, Dhruv, Elements of Optimal Predictive Modelling Success in Data Science: An Analysis of Survey Data for the 'Give Me Some Credit' Competition Hosted on Kaggle, March 2, 2013.

[6] McDonald R.A., Hand D.J., Eckley I.A. (2003) An Empirical Comparison of Three Boosting Algorithms on Real Data Sets with Artificial Class Noise. In: Windeatt T., Roli F. (eds) Multiple Classifier Systems. MCS 2003. Lecture Notes in Computer Science, vol 2709. Springer, Berlin, Heidelberg

[7] XiaYufeia, LiuChuanzhea, LiYuYingb, LiuNanaa: A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring, Expert Systems with Applications, Volume 78, 15 July 2017, Pages 225-24.

[8] J. N. K. Liu, Y. He and X. Wang, "Improving kernel incapability by equivalent probability in flexible naïve Bayesian," *2012 IEEE International Conference on Fuzzy Systems*, Brisbane, QLD, 2012, pp. 1-8.

[9] Kaur, Prabhjot and Anjana Gosain. "FF-SMOTE: A Metaheuristic Approach to Combat Class Imbalance in Binary Classification." *Applied Artificial Intelligence* 33 (2019): 420-439.

[10] Xu-Cheng Yin, Chang-Ping Liu: Feature combination using boosting, October 2005.

[11] Radha Vedala, Bandaru Rakesh Kumar, 'An application of Naive Bayes classification for credit scoring in e-lending platform', 2012 International Conference on Data Science & Engineering (ICDSE).

[12] Robert E. Schapire, 'A Brief Introduction to Boosting', 1999 Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence.

[13] Stefan Less manna, Bart Baesensb ,Hsin-VonnSeowd, LynC.Thomasc: Benchmarking state-of-the-art classification algorithms for credit scoring, June 2003, An update of research, 2015.

[14] ChengYeha Che-huiLien: The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients. Expert Systems with Applications, Volume 36, Issue 2, Part 1, March 2009, Pages 2473