A Mathematical Essay on Random Forest

Vignesh Kumar S
Department of Chemical Engineering
Indian Institute of Technology, Madras
Email address: ch18b118@smail.iitm.ac.in

***Abstract*—The purpose of this article is to understand the Random Forest algorithm and the mathematics behind it. In this work, we demonstrate the application of the Random Forest classifier illustrated through Car Evaluation Database. Due to the complex nature and the abstractness involved in classifying a car given its features, it is often difficult to obtain a good solution through manual approaches. Adopting a machine learning technique can help in obtaining insights and analysing different features in a large amount of data comprehensively. We try to establish a nexus between the nature of the car and different features involving its specifications.**

I. Introduction

In this paper, we will study Random Forests, a technique predominantly based on collection of tree-like models of decisions that only contains control statements. It is used to analyze and model either a dichotomous or multiple outcomes in classification setting and could also be extended as a regressor in regression setting. We will use this technique to classify the nature of a car using factors such as safety, capacity, costs etc.. This analysis would aid in obtaining insights among a large dataset and understanding the trend behind the nature of a car and various other factors.

The classification problem attempts to establish a relationship between a categorical target variable with one or more explanatory variable(s). Random Forest consists of Decision Trees which are flow-chart like structure in which each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label. In order to make a prediction for a given observation, we run the sample through the collection of Decision Trees, a.k.a the Forest and typically use the mean (in case of regression) or mode (in case of classification) of all the predictions to make the final prediction.

The dataset used for this problem comprises safety, luggage space, capacity, doors, maintenance and buying costs.. We use Random Forest classifier to learn and predict whether the nature of the car (unaccountable, accountable, good, very good) given these input features.

This work represents the concepts behind Random Forest and evaluation metrics involved. We then use this technique to establish the relationship to predict the class of of a car it may belong to.

II. Random Forest

In the earlier paper, we had discussed about Decision Trees. To recap, they can be applied to both classification and regression problems. In this paper, we'll also look at Random Forest in a classification point of view but the following analysis could easily be extended into regression setting just by predicting the mean among the leaf nodes and using a squared loss function. While decision trees are easier to interpret and are often compared as a white-box model due to the ability we could compare its prediction, by combining many such trees, we could improve the prediction accuracy of a tree with much lower variance but at the expense of some loss in interpretability.

The goal in classification problem is to take an input feature $\boldsymbol{x}$ and assign it to one of the K classes. Ensemble methods like Random Forests consists of predictions averaged over a number of Decision trees. When a new observation comes, it considers the majority vote or average value to determine the final prediction. Even though decision trees are known to overfit and hence a high variance model, averaging the predictions over many trees significantly lowers the variance and improves the nature of predictions. This is the principle behind Random Forest and it hence addresses the shortcomings of Decision Trees while exploiting its powerful nature of tree.

Decision Trees take a top-down, greedy approach with recursive binary splitting. It is important to understand that since Random Forest is a collection of many such trees, it is computationally infeasible to find the global optimum for Random Forest. Hence we take a greedy approach to minimize the cost function.

Generally for a classification setting, accuracy is often deemed as a standard metric that could be used while optimization. But we showed that for a decision tree, metrics such as Gini Index and Entropy are proven to be effective in practice. Random Forests being just a ensemble collection of parallelly trained trees, uses these metrics as well for determining the best split within each tree each step.

'The below plots represents the decision boundary of a Decision Tree and Random Forest classifier (made out of 100 decision trees) obtained on the same synthetic dataset of two features. The decision tree learned is clearly overfit and doesn't seem to generalise well. But the random forest seems to generalise better while not compromising the predictive capability. Thus random forests clearly seems to be a better version compared to a singly overfit decision tree.

Fig. 1: Decision Boundary produced by a decision tree

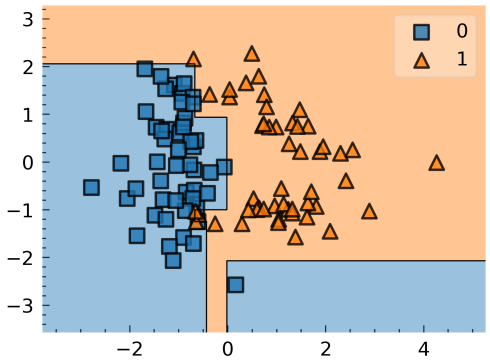
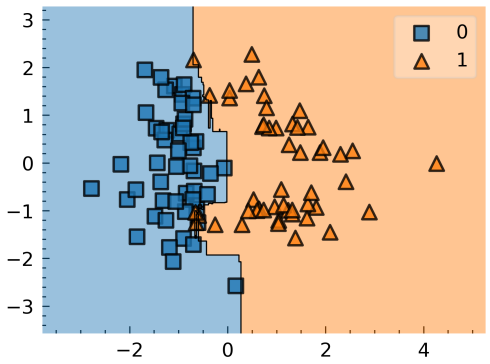


Fig. 2: Decision Boundary produced by a Random Forest



1) Metrics: Gini Index/Impurity:

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$
$$= 1 - \sum_{k=1}^{K} \hat{p}_{mk}^2$$

This is a measure of total variance across the K classes, where $\hat{p}_{mk}$ is the proportion of training observation in the m^{th} region that are from the k^{th} class.

Gini index takes on a small value if all of the $\hat{p}_{mk}$'s are close to zero or one. A zero value implies that the node is pure.

2) Entropy:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} log\, \hat{p}_{mk}$$

Notice the close resemblance with the thermodynamics entropy formulation. Likewise, here too, entropy takes on a small value if the m^{th} node is pure or less random and large values if it is impure.

When building a Decision Tree, either Gini-index or entropy is typically used to evaluate the quality of a split. However, a split produces 2 nodes and in order to calculate the measure, a weighted average of leaf node is taken. The split that results in the most reduction in the measure (Gini index/ Entropy) is chosen as the best split. This process continues till an appropriate stopping criterion is reached or when every training dataset is perfectly classified. For mathematical detail regarding to cost function of a Decision Tree, refer to the previous paper in the series. In this paper, we hence forth, will focus on bagging concepts and ensemble methods related to trees.

A. Bagging

Decision Trees suffer from high variance and can be very non-robust to dataset. A slight change or using a different sample from the dataset could lead to different trees. Bootstrap Aggregation or Bagging is a procedure specifically useful for high capacity classifiers for reducing their variance. Bootstrapping is a technique to obtain data-points. We generally do not have access to multiple training sets and hence we do bootstrapping, by taking repeated samples from the training data sets. Hence, we obtain distinct datasets by repeatedly re-sampling observations from the original dataset.This sampling is done with replacement, meaning the same observation can occur more than once in the bootstrap dataset.

To apply bagging to trees, we simply construct B trees $\hat{f}^1(x), \hat{f}^2(x), .., \hat{f}^B(x)$ using B bootstrapped training sets and average the resulting predictions to obtain a single low variance statistical model. The constructed trees however are grown deep and hence suffer from high variance. But averaging reduces variance and results in a better model. For classification, instead of averaging we take the most frequent predictions. Rest of the approach remains the same. B per se is not a critical parameter as B is just the number of trees we are averaging at. High values of B just implies we are averaging over many trees and hence apart from computational complexity, increasing B doesn't lead to overfit. In practice, B is kept sufficiently large so that the error has settled down.

B. Random Forests

Similar to bagging, we build a number of decision trees on bootstrapped training samples. But when building these Decision Trees, when each time a split is considered, a random sample of m predictors is chosen as split candidates from the full set of p-predictors.This also ensures a single feature is not influencing majority of decisions in all trees. Usually m $\approx \sqrt{p}$ but m is actually a hyper-parameter that could be chosen via cross validation. Prediction from the bagged trees are prone to be correlated. In order to prevent this from happening, we deploy such variation as above. This decorrelates the trees. The variance after averaging reduces in a much better fashion if the predictions from each tree are uncorrelated.

The main advantage of Bagging is that all the trees can be parallelly/ separately trained and hence could be sped up easily. But this kind of approach cannot be readily employed in sequentially/adaptively-trained models also known as boosting.

C. Out of Bag Error (OOB) Estimation

If we take n samples *with replacement* out of n samples, we can show that each bagged trees makes use of approx. $1 - \frac{1}{e}^{th}$ samples, or $approx. \frac{2}{3}^{rd}$ of observations. The remaining $1/3^{rd}$ of observations are left out in each trees. We can predict the response for the i^{th} observation using each of trees in which that observation was OOB or not present. This will yield around $B/3$ predictions for the i^{th} prediction. We can compute a single prediction easily by using an average or a majority vote. The resulting OOB error is a valid estimate of the test error for the bagged/ensemble Random Forest model.

D. Time Complexities

In a decision tree, a split has to be found until a maximum depth d_{epth} has been reached. In the worst case, maximum depth is equal to n, the training set size. The strategy for finding split is to look for each variable to different thresholds. Since there are p such variables, we make np decisions per node. Considering a Random Forest with n_{trees} such trees,

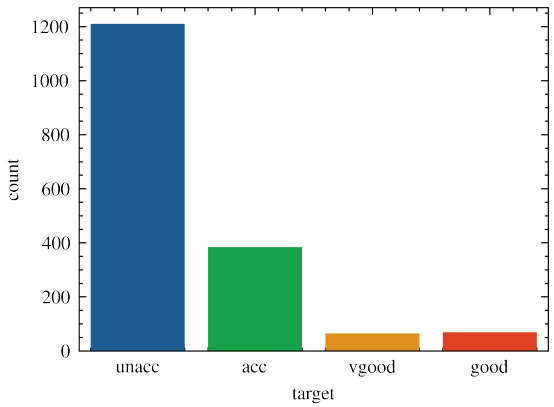
1) Assuming no-feature selection, the upper bound for train time complexity is $O(n_{trees}n^2p)$.
2) If $\sqrt{p}$ random feature selection is done at each node, the upper bound for train time complexity is $O(n_{trees}n^2\sqrt{p})$.

III. The Problem

In this section, We will analyse the Car Evaluation database and try to predict the nature of a car.

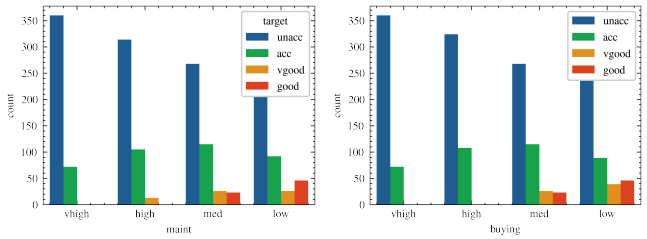
A. Imbalanced Dataset

The dataset predominantly contains class of type 'unacc'. Number of points containing classes 'vgood' and 'good' are considerably less. This is handled by giving appropriate weights to the Random Forest learned from the data.



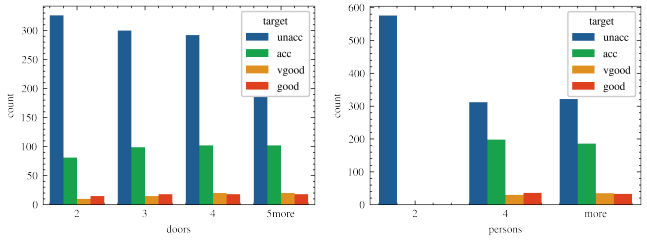
B. Attributes

From plot (a) and (b), We could infer that Most cars falls under unaccountable category. Cars that have high and vhigh buying and maintenance prices are generally unaccountable while the lower priced cars consists of all 4 categories. No good and vgood cars are present with vhigh buying or maintenance.



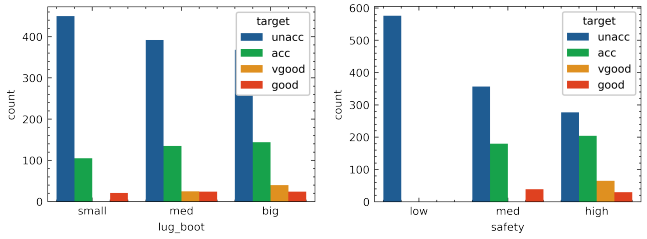
(a) (b)

From plot (c), Almost all categories have similar distributions. Number of doors doesn't seem to be a crucial criteria that could distinguish classes. From plot (d), Cars that have only 2 person capacity are unaccountable. 4 and 4 plus have similar distributions.



(c) (d)

From plot (e), Cars which have small lug boot are never 'vgood'. There has been mixed reviews across cars which have medium to big luggage. However, a trend of bigger space implying higher ratings are seen. From plot (f), Safety seems to play a major role. Every car which have a estimated low safety are unaccountable. No cars are classified as 'vgood' which have medium or low safety. A mixed reviews are seen across cars which were classified as high safety.



(e) (f)

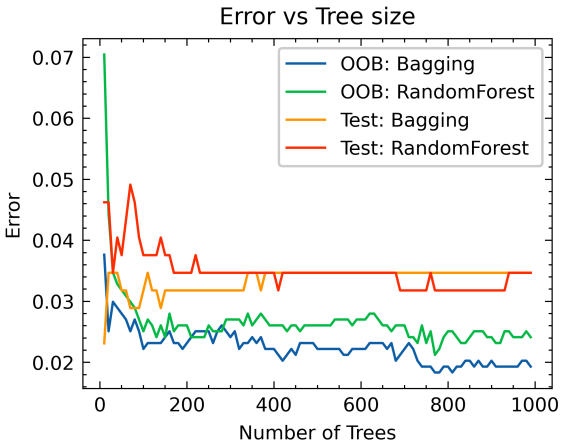
The dataset after checking for missing values is Label Encoded as the features found were categorical of type ordinal. There exists some order in ordinal features which makes label encoding a better choice compared to one-hot encoding.

C. OOB and Test Error of Bagged Model and Random Forest

Bagging and random forest results for the Car Evaluation Data is shown below. The test error (orange and red) is shown as a function of B, the number of bootstrapped training sets used. Random forests were applied with m = $\sqrt{p}$. The green and blue lines show the OOB error, which in this case is

considerably lower but looks like a decent approximation of the test error.

In this case, bagging (without variable selection or $m = p$) seems to perform tiny better than Random Forest ($m \approx \sqrt{p}$) itself. This may be due to the fact the trees are already uncorrelated and doesn't need random variable selection. We'll confirm this using grid-searching through different m's in a k-fold cross validation setting.



D. Model Evaluation

From the error vs tree size plot, 250 trees are sufficient enough where the errors stabilize pretty well. Searching over m (Number of features to consider when looking for best split) from 1 to 6 (total number of features) in 10-fold cross validation setting yields m as 6 (All features). This means for this particular dataset, Random forest without restricted number of features seems to perform marginally better over classifiers with feature selection. This strengthens the reasoning behind Error vs Tree size plot.

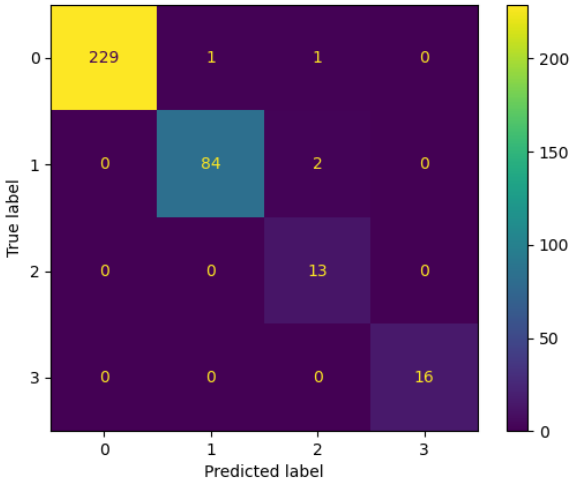
In a classification setting, accuracy may not be the best score to consider especially in a skewed class situation. The training data in our case is imbalanced. Precision is the ratio of correct positive predictions to the total positive predictions of our model. The recall is the ratio of positive instances that are correctly detected by our classifier. The F1 score is simply a harmonic average of these two.

The following metrics are reported for the best classifier on the test set.

Random Forest Classifier				
	Unacc	Acc	Good	Vgood
Precision	1.0	0.988	0.8125	1.0
Recall	0.991	0.976	1.0	1.0
F1-Score	0.995	0.982	0.8965	1.0

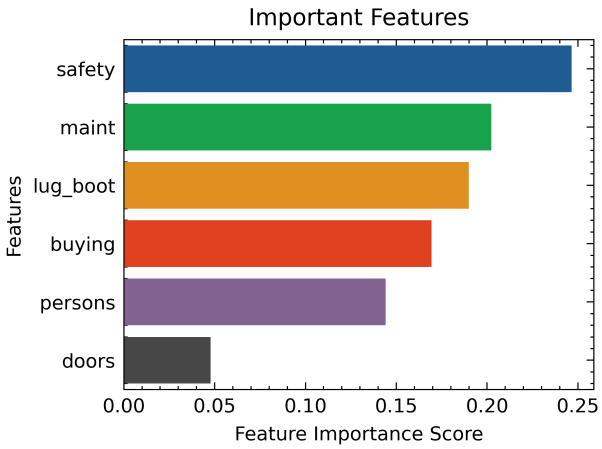
Accuracy = 0.988

The best random forest classifier (got through gridsearch) performs incredibly well over the current dataset. The F1-Score for class 'Good' may be low due to skewed data but apart from that, prediction accuracy seems too good.



The classifier performs extremely well on the test set. Naturally due to imbalanced dataset, the f1-score for class 2(good) is a little less than others but that's seems tolerable considering the skewness. However, the model performance seems too good to be true. This may be due to the fact that the target decision may indeed have arose due to control sequence like data generation. The true nature of the model can only be revealed when deployed in real time and assessing the results. Since the model is a ensemble and is bound to have lesser variance, we will go ahead with this final model unless we get further investigations.

Unlike decision trees, it is no longer possible to view and interpret it as a single tree like structure as this consists of two hundred fifty such trees and is an average over it. This is the trade-off between improved predictions over the expense of interpretability. However a summary of variable importance could be obtained using the Gini index that is decreased due to splits over a given predictor, averaged over all B trees.



The above suggests that safety is the main feature followed by maintenance to predict the nature of the cars. The least

important feature is found to be number of doors. Note that this agrees with the intuition that we obtained through visualization of the dataset.

IV. Conclusions

Based on our analysis, the key takeaways we had from this exercise are the following:

1) Random Forest are powerful learning algorithms that overcome the shortcomings of decision tree by using bagging and averaging techniques.
2) This comes over the expense of interpretability. However still, the model could be fairly interpreted but to a less extent.
3) Safety is the most important feature that is considered for classifying the nature of a car. We also saw how just by visualizing, how a large amounts of insights were obtained which were consistent with the result obtained through the algorithm.

Possible avenues of research could include trying out dimensionality reduction before applying Random Forest, visualizing decision stumps and using boosting methods to compare and contrast Random Forests.

References

[1] An Introduction to Statistical Learning, Gareth James et al. pp.303-323
[2] Aurelian Geron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools and Techniques to Build Intelligent Systems, pp. 230–244
[3] Christopher M. Bishop, Pattern Recognition and Machine Learning