About Random Forest and imbalanced datasets

Matteo Sani

January 10, 2022

Contents

1 Classification Trees

Supervised learning algorithm introduced by grajski1986classification in which the general idea is to recursively split the covariate space into homogeneous and non-overlapping partitions within each the prediction of the response is constant. Suppose to have a sample $X_1, .., X_n$ and each sample point is a p-variate random variable $X_{i1}, .., X_{ip}$ with $i = 1, .., n$. In the initial setting exists only a single region R which is equivalent to the covariate space χ. The goal is to find a splitting point, defined as a couple (j, s_1) where $j = 1, .., p$ is one of the p covariates and s_1 is one of its possible values in the sample, that partition the covariate space into two non-overlapping regions. Mathematically speaking:

$$(j, s_1) : R_1 = \{(X_1, ...X_p) \in \chi : X_j \leq s_1\}, \qquad R_2 = \{(X_1, ...X_p) \in \chi : X_j > s_1\} \tag{1}$$

The best splitting point is defined as that point that provides the best improvement of an impurity measure ϕ defined as a measure of the degree of heterogeneity in the class distribution. Possible choices of ϕ are:

Gini Index

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

Entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \tag{3}$$

where $\hat{p}_{mk}$ is the proportion of observations that are from the k-th class in the m-th region. Looking at the Gini index, we notice that it can be considered as a sum of Bernoulli variances over the k classes; it's easy to see that the minimum value is observed when the $\hat{p}_{mk}$'s have extreme values (i.e. when the partition is mainly represented of observations belonging to one class). Once the splitting has occurred, a proportion p_1 of the observations is sent to the first region, p_2 to the second and the impurity measure for the two regions ϕ_{R_1},ϕ_{R_2} is recorded. Defined ϕ_R the overall impurity measure, the change in impurity is given by:

$$\Delta i = \phi_R - p_1\phi_{R_1} - p_2\phi_{R_2} \tag{4}$$

so the best splitting point is obtained via minimization:

$$\min_{j,s1} \left[p_1\phi_{R_1} + p_2\phi_{R_2} \right] \tag{5}$$

The procedure is applied recursively on both the partitions created until some termination criterion is satisfied; generally we consider a minimum number of observations within each region (or leaf node) or a maximum depth of the tree. This is done since a very depth tree or a tree in which each leaf node contains only few observations, is more likely to overfit our data providing lack in generalization which will lead in poor predictive performance. Once all the partitions are being defined, the prediction of the response is done by taking the most occurring class in that partition.

2 Ensemble Methods

Decision trees are very simply to apply and to communicate to people: they represent graphically how people make decisions! The cost of simplicity is a lower predictive accuracy when compared to other classification algorithms due to the fact that they more likely will overfit our data. However the performance can be drastically improved with **ensemble methods**, defined as approaches that combines many *weak learners* defined as such since they provide poor predictions on their own. Ensemble methods are not properly related to decision trees, since the concept is a general idea. As a drawback, these methods are no longer easily interpretable and the decision path is much less transparent respect to a DT. Furthermore, in RF is not possible to depict the role of the variables in explaining or causing; it's only possible to measure the impact of each predictor into the predictive process (*variables importance*).The following are two of the ensemble models which use decision tree as weak learner.

2.1 Bagging

Bagging, short for Bootstrap aggregation, is a technique which combines many classification trees in order to improve the prediction accuracy. In the following we present the algorithm of bagging.

- Given a training set, a number B of new sets are obtained via Bootstrap i.e by resampling with replacement the given set.
- For each Bootstrap sample a very deep and not pruned classification tree is fitted: it will have low bias (performance on the training set) and high variance.
- Define $\hat{f}^{*b}(x)$ the prediction for a new observation x from a single tree. The bagging prediction $\hat{f}_{bag}(x)$ is given by the *majority vote rule*.

$$\hat{f}_{bag}(x) = \arg\max_k \sum_{b=1}^{B} I_{\{\hat{f}^{*b}(x)=k)\}} \tag{6}$$

Out Of Bag Test Error Estimation Bagging provides also a good estimate of the test error; in particular, it can been shown that on average each tree is fitted on the $\frac{2}{3}$ of the entire sample: the remaining $\frac{1}{3}$ are called Out Of Bag (OOB) observations; so given a single observation i, there will be about $\frac{B}{3}$ trees for which it is an OOB. Using them for making the prediction and taking the majority vote, the estimate error is computed. Repeating this for all the observations in the sample one can compute the total estimate error and so provide an estimate of the test error.

2.2 Random Forest

breiman2001random

The amount of variance reduction in bagging is reduced if the trees are highly correlated; this phenomena occurs when the trees share similar structures. Random Forest is a bagging procedure, with the

only difference that each tree is forced to consider only a subset m of predictors randomly chosen; in classification tasks generally $m = \sqrt{p}$. This provides a simple tool to decorrelate the trees and improve the predictive accuracy. The idea beyond is that if there's a variable with an high impact on the predictions, the trees more likely will make the first split over that variable, and so the tree structures will be similar; by randomly choosing a subset of variables we give chances also to the less important variables!

3 Learning algorithms with imbalanced data

Considering a binary classification problem (but is also true for multi class classification), a dataset is said *imbalanced* when it contains lots of samples belonging to one class which is called *majority class* and few samples in the other, *minority class*. The problem here is that many classification algorithms assumes that there is an approximately equal number of samples in each class; this is also the reason for which *accuracy* is the most used metric to evaluate the predictive performances. In imbalanced data framework, each Bootstrap sample more likely will contain mainly observations in the *majority class* and so the algorithm will learn quite perfectly how to predict them, but with poor predictive performance on the *minority class*.
One way to solve this issue is to use **Resampling techniques** i.e. techniques which aim is to define a new training set by sampling from the imbalanced one; these are divided into:

- Oversampling
- Undersampling
- Hybrid

A lot of literature and techniques exist on the topic, but we will focus only on three of them, one for each category.

3.1 Random Under Sampling (RUS)

This technique is extremely easy: it simply randomly remove observations from the majority class until the set is balanced. It can be applied in presence of a great number of observations. However, when the dataset is highly imbalanced, there will remain too few samples to train the model; furthermore, we loose a relevant amount of data, which is never a good idea.

3.2 SMOTE

Introduced by chawla2002smote, SMOTE is a data augmentation technique which generates new synthetic samples by resampling the minority class until the new training set is balanced. Oversampling techniques such as ROS (Random Oversampling) are criticized since they could lead to overfitting. However this is not the case of SMOTE because instead of randomly copy existing sample, it generates new ones by taking into account the nearest neighbours.

Taking one sample X we compute the *k-nearest neighbours* $T_1, .., T_k$ where k is the number of neighbours that must be set a priori (generally 5). We then take only $N < K$ of them where N depends on the amount of oversampling desired and for each one T_i, compute the difference between its feature vector and the feature vector of the minority sample X. This difference is then multiplied by a random number $\delta \in (0, 1)$ and then added to the feature vector of X. The new synthetic samples X_i are obtained as follows.

$$X_i = X + \delta(T_i - X) \quad i = 1, .., N \tag{7}$$

The amount of new synthetic samples depends on the imbalance degree and so to the ratio between number of minority samples and majority ones. For example, if the ratio between minority and majority samples is 0.5, we will consider all the minority samples and for each of them we will consider only 2 of the k neighbours and create 2 new synthetic samples as described above.

3.3 SMOTE + RUS

In the original paper, it has been demonstrated that SMOTE provide better performances when combined with Random Undersampling. The idea is to firstly oversample the minority class until a desired ratio is achieved, then undersample the majority class in order to balance the data. In this way we fix the issues induced by RUS.

References

[1] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning*, vol. 112. Springer, 2013.

[2] G. Breiman Leo, L. Kamil A, G. V. Di Prisco, and W. J. Freeman, "Classification of eeg spatial patterns with a tree-structured methodology: Cart," *IEEE transactions on biomedical engineering*, no. 12, pp. 1076–1086, 1986.

[3] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[4] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority oversampling technique," *Journal of artificial intelligence research*, vol. 16, pp. 321–357, 2002.

[5] A. Fernández, S. Garcia, F. Herrera, and N. V. Chawla, "Smote for learning from imbalanced data: progress and challenges, marking the 15-year anniversary," *Journal of artificial intelligence research*, vol. 61, pp. 863–905, 2018.

[6] X. Tan, S. Su, Z. Huang, X. Guo, Z. Zuo, X. Sun, and L. Li, "Wireless sensor networks intrusion detection based on smote and the random forest algorithm," *Sensors*, vol. 19, no. 1, p. 203, 2019.