Tree Methods, Random Forests and Bagging

Mihai Croicu[1]

[1]Department of Peace and Conflict Research
Uppsala University

2019-03-30

Generalized Linear Models and the curse of linearity/1

1. OLS:
   - $$Y \sim N(\mu, \sigma), \mu = \beta X$$
2. Logistic:
   - $$Y \sim Bernoulli(\mu), \ln\left(\frac{\mu}{1-\mu}\right) = \beta X$$
3. Poisson
   - $$Y \sim Poisson(\mu), ln(\mu) = \beta X$$

Independent variables always contribute a linear component to the model (*the systematic component*). What differs (making the models non-linear) is a transformation of that component to fit the distributional assumptions (stochastic component) of the response variable (*the link function* - $g(\mu)$).

The curse of linearity/2

- Each independent variable adds information independently to the model. The total contribution to the model of the IVs is always the euclidean/linear sum of all the predictors.
- Each independent variable has a linear relationship to a (link) function (or transformation) of the dependent variable.

Reasonable in the social sciences? Not frequently, at least not according to theories! Examples:

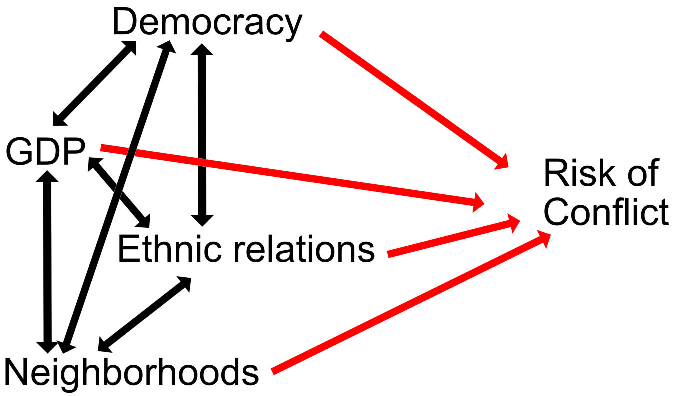
- Highly repressive countries and highly democratic countries experience little civil conflict; it is in the middle where most conflict is. (Hegre, 2001; Jones and Lupu, 2018)
- There is no effect of climate anomalies on the risk of conflict in general; the only increase in risk happens where climate anomalies meet extremely poor, unstable societies, with very low societal performance (high infant mortality) (Von Uexkull et al., 2016)

The curse of linearity/3

- Transform IVs so that the transformation has a more linear relationship to $g(\mu)$ - e.g. $\textit{logarithms} - \textit{log}(X), \textit{polynomials} - X + X^2 + X^3$
- Use interaction effects when the effect of one predictor on the dependent variable depends on the level of another predictor.
- Choose a more complex link function (e.g. log-linear models, negative-binomial models).
- Split samples where you expect the relationship to exist in one category of data, but not in the other one.

The curse of linearity/3

But the real life is much more complex. Interactions are much more complex, and relations are rarely taking a functional form!



Can we do better?

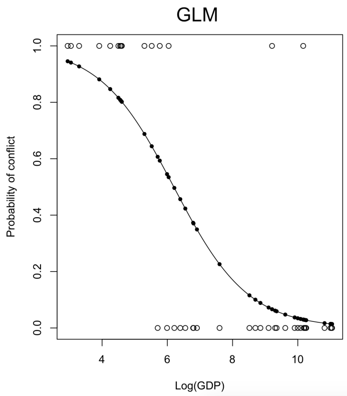
Perhaps! Let's look at how humans make decisions!
Is it an orange or a grapefruit?

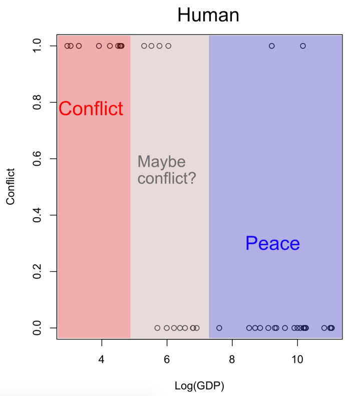
Human decision making/1

Is it an orange or a grapefruit?

- Orange color?
  - Smooth skin? $\Longrightarrow$ Grapefruit!
  - Bumpy skin? $\Longrightarrow$ Orange!
- Dark orange color?
  - Smooth skin? $\Longrightarrow$ Grapefruit!
  - Bumpy skin? $\Longrightarrow$ Orange!
- Yellow color? $\Longrightarrow$ Grapefruit!

Human decision making/2





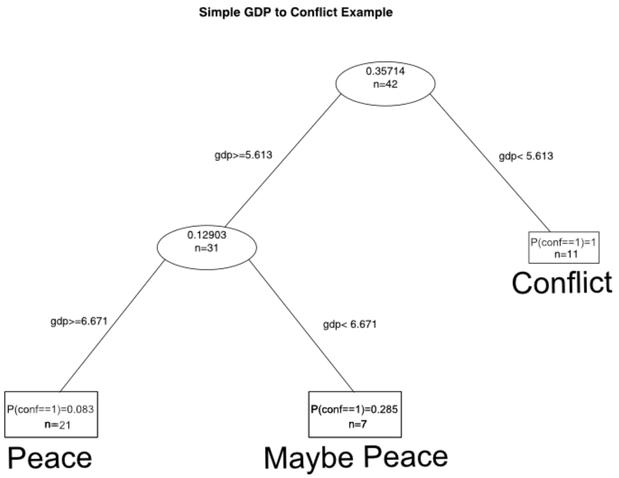
Decision trees

We split the data repeatedly. *How?*

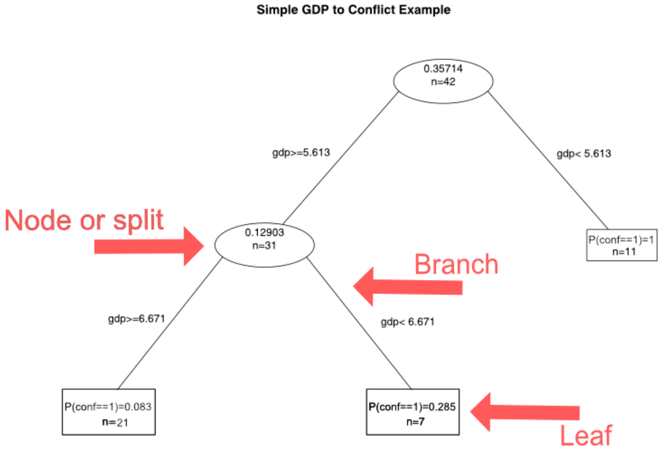
1. Take one predictor at a time.
2. See which predictor **splits the data best** into the categories of the dependent variable (e.g. which single predictor allows you to best discern observations of conflict conflict from those of peace).
3. Split the data at the value of the predictor where the categories are **best separated**. This will result into a bucket for each category. Each bucket will have observations more likely to correspond to the respective category (in our example, one bucket for peace, one bucket for conflict, each containing more likely observations for each of the categories).
4. For each bucket, go back to step 1. and repeat, only using data in bucket, until the data is **fully separated** into categories.

Algorithm works for binomial responses (1/0), multinomial responses (categorical data) and continuous data (think of it as infinite categories).

Decision trees/2



Decision trees/3



Decision trees/4

How do we know when to split? Classification and Regression trees (CART):

- Minimize Gini Impurity (G) across all branches:

$$G = 1 - \sum_{i=1}^{C} p_i^2$$

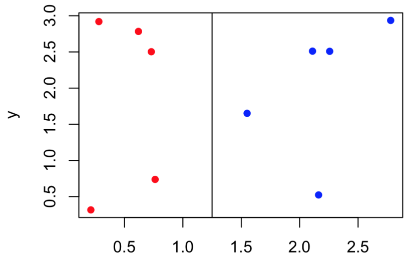
  where p_i is the proportion of the points (in the data) to be put in the correct class given the split, with C as the number of classes.

- Create a weighted sum of G for each branch weighted by the number of points assigned to each branch B; minimize that sum.

$$min(\sum_{i=1}^{B} G_B * \frac{P \in B}{P})$$

  This is tried by the algorithm at each potential split of each predictor $x_i \in X$!

Decision trees/5



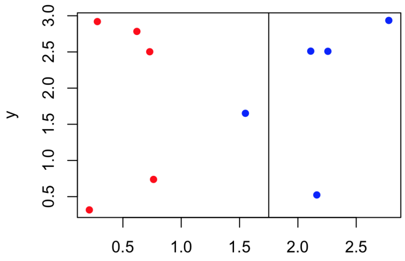
split x=1.25

$G_{LEFT}=1-1^2$ $G_{RIGHT}=1-1^2$

$G_{LEFT}=0$ $G_{RIGHT}=0$

$G_{SPLIT}=0*(5/10)+0*(5/10)$

$G_{SPLIT}= 0$



split x=1.75

$G_{LEFT}=1-(5/6)^2$ $G_{RIGHT}=1-1^2$

$G_{LEFT}=0.306$ $G_{RIGHT}=0$

$G_{SPLIT}=0.306*(6/10)+0*(4/10)$

$G_{SPLIT}= 0.184$

Decision trees/6

There are other algorithms:

- Iterative Dichotomiser (ID3, C4.5, C5.0) - Maximize information gain (IG):

$$IG = -\sum_{i=1}^{C} p_i * log_2(p_i)$$

- Conditional Inference Trees - test partial correlation of each $x_i \in X$ to Y for null hypothesis (no correlation between x_y and Y) and split on smallest p value.[1]

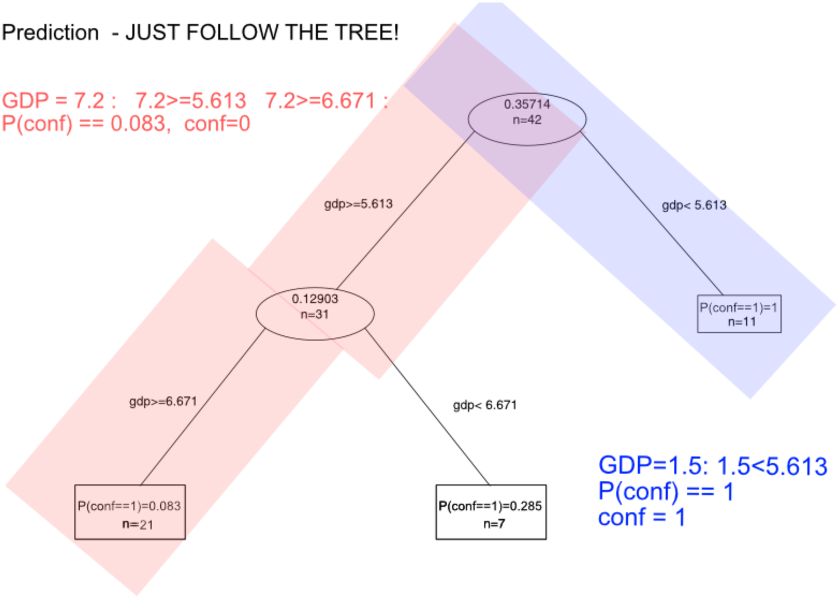
CART is most widely used.

[1]yes, we are talking of *THAT* p-value.

Decision trees/7 - Forecast

Prediction - JUST FOLLOW THE TREE!

GDP = 7.2 : 7.2>=5.613 7.2>=6.671 :
P(conf) == 0.083, conf=0



GDP=1.5: 1.5<5.613
P(conf) == 1
conf = 1

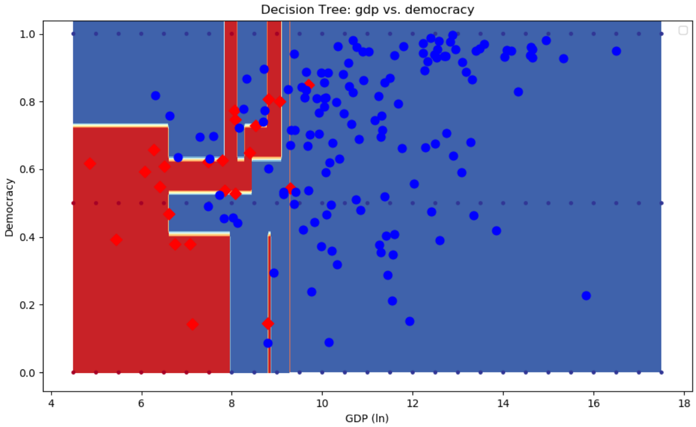
Decision trees/7 - Evaluation

- No distributional assumptions;
- No parametric form for error;

Thus, only out-of-sample goodness-of-fit statistics.

- Techniques: Cross-validation, Forecasts into the future;
- Continuous outcome (regression): Mean Absolute Error (MAE), (Root) Mean Squared Error (RMSE) etc.
- Binary/discrete outcome (classification): ROC-curves, PR-curves, AUC, Thresholds etc. etc.

Decision trees/8 - Real Data



Decision trees/9 - Real Data

Extreme overfitting!

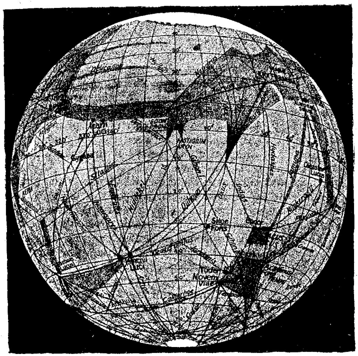
- GDP $= 9.753$ and democracy $= 0.2 \implies$ conflict
- GDP $= 9.752$ or 9.755 and democracy $= 0.2$? $\implies$ peace

Remember: *Overfitting* is extracting a pattern from the data where there isn't one in reality.

- Trees have low bias - they detect patterns in the data; but very high variance - each new point will change the shape of the tree.

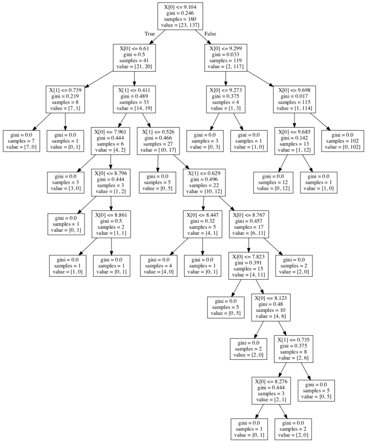
Decision trees/10 - Humans again!

Well, humans make the same mistake. Our brains are built to see patterns. Even where there aren't any. Schiaparelli and Lowell (1906) saw patterns from random lines on Mars, interpreting them to be a civilization!



Decision trees/11 - What happens?

We split until there's nothing left to split (Gini=0)...



X[0] <= 9.104 gini = 0.246 samples = 160 value = [23, 137]
True
False
X[0] <= 6.61 gini = 0.5 samples = 41 value = [21, 20]
X[0] <= 9.299 gini = 0.033 samples = 119 value = [2, 117]
X[1] <= 0.739 gini = 0.219 samples = 8 value = [7, 1]
X[1] <= 0.411 gini = 0.489 samples = 33 value = [14, 19]
X[0] <= 9.273 gini = 0.375 samples = 4 value = [1, 3]
X[0] <= 9.698 gini = 0.017 samples = 115 value = [1, 114]
gini = 0.0 samples = 7 value = [7, 0]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= 7.961 gini = 0.444 samples = 6 value = [4, 2]
X[1] <= 0.526 gini = 0.466 samples = 27 value = [10, 17]
gini = 0.0 samples = 3 value = [0, 3]
gini = 0.0 samples = 1 value = [1, 0]
X[0] <= 9.685 gini = 0.142 samples = 13 value = [1, 12]
gini = 0.0 samples = 102 value = [0, 102]
gini = 0.0 samples = 3 value = [3, 0]
X[0] <= 8.796 gini = 0.444 samples = 3 value = [1, 2]
gini = 0.0 samples = 5 value = [0, 5]
X[1] <= 0.629 gini = 0.496 samples = 22 value = [10, 12]
gini = 0.0 samples = 12 value = [0, 12]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= 8.861 gini = 0.5 samples = 2 value = [1, 1]
X[0] <= 8.447 gini = 0.32 samples = 5 value = [4, 1]
X[0] <= 8.767 gini = 0.457 samples = 17 value = [6, 11]
gini = 0.0 samples = 1 value = [1, 0]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 4 value = [4, 0]
gini = 0.0 samples = 1 value = [0, 1]
X[0] <= 7.823 gini = 0.391 samples = 15 value = [4, 11]
gini = 0.0 samples = 2 value = [2, 0]
gini = 0.0 samples = 5 value = [0, 5]
X[0] <= 8.123 gini = 0.48 samples = 10 value = [4, 6]
gini = 0.0 samples = 2 value = [2, 0]
X[1] <= 0.735 gini = 0.375 samples = 8 value = [2, 6]
X[0] <= 8.276 gini = 0.444 samples = 3 value = [2, 1]
gini = 0.0 samples = 5 value = [0, 5]
gini = 0.0 samples = 1 value = [0, 1]
gini = 0.0 samples = 2 value = [2, 0]

Decision trees/12 - What can we do?

- Limit when a split can happen:
  - limiting the depth of the tree;
  - forcing the tree to have at least a minimum amount of observations before splitting;
  - forcing the leaves to have at least minimum amount of observations;
- Pruning the trees - eliminating those splits where out-of-sample cross-validation indicates no improvement in predictive performance.
- Or **something MUCH smarter** - and inspired by humans: random forests.

Random Forests/1 - What can we do?

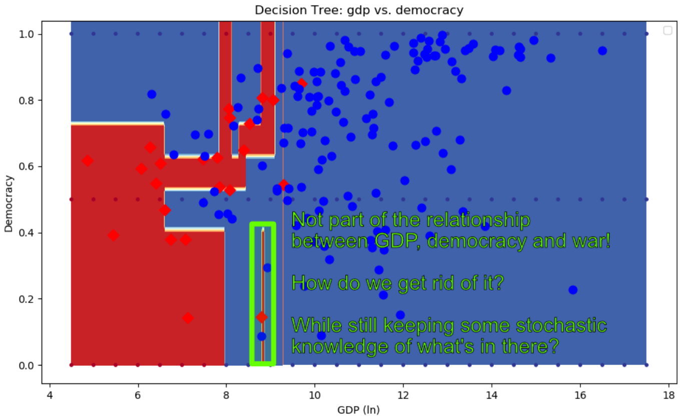
Back to humans - crowd wisdom is better than individual wisdom.
Intuition: If 20 people see the same thing, looking from different sides, that must mean something is there!

How do we do "crowd wisdom"? We vote! Works well enough – we elect our government in that way, and it works better (*less worse*) than anything else we tried! (Churchill, 1947).

Random Forests/2 - Intuition

- We want to identify the pattern (the **generally existent, population level** relation between the predictors and the dependent variable)
- We want to get rid of the noise that may over-fit the data (a pattern that's identified **because the model fits the sample data much too closely**).
- Do we know of a way?

Do we know of a way?



Not part of the relationship between GDP, democracy and war!

How do we get rid of it?

While still keeping some stochastic knowledge of what's in there?

Yes, we do!

And we know it - we've all done Monte Carlos (MC)!

- Bootstrap the data!.
- Then, train a tree on each bootstrapped sample. This way, the pesky observation will only be seen by **SOME** trees. Others will be oblivious to it.
- Have all the trees predict new data individually. Those that saw the pesky point will say "something's there". Those that didn't will say "nothing's there".

How many times? Same as with MC: hundreds, thousands of times!

What do we do with all these trees?

We have them vote in a (free and fair) election!

- Each tree's prediction counts as one vote. They all cast their vote.
- Tally all the votes; if most classifiers see "conflict" by looking at various re-sampled subsets, it's clearly part of the general relationship.
- If only those seeing the "wonky" point will vote "conflict", then it's just that point that's misbehaving (an outlier).

And this works why?

Because each tree only sees a limited amount of data, each tree has access to the following proportion of data (P_e) from the original dataset, given n observations:

$$P_e = 1 - (1 - \frac{1}{n})^n$$

P_e converges to 63% (at $n = 50$ or above).
Effect - reduction of variance (and no effect on bias; but remember trees are unbiased).

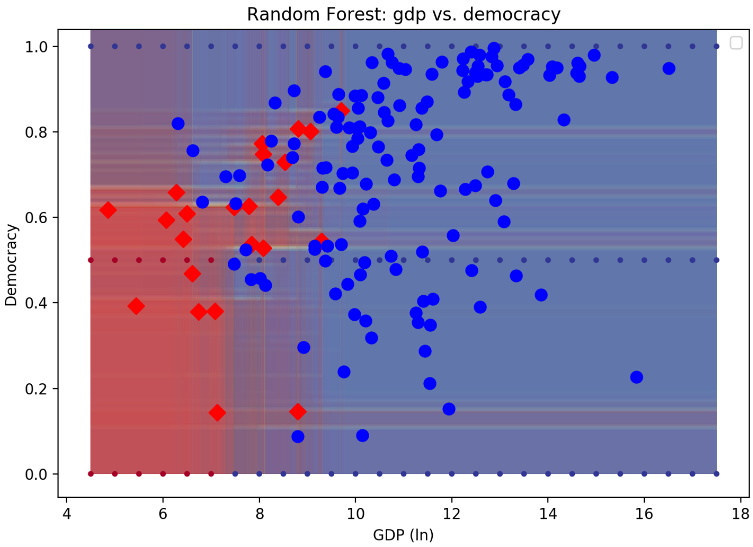
And some terminology!

- Training multiple models and having each have a say in contributing to the final result in an "election" is called creating an **ensemble**.
- The election needs not be fair! Or free! We have some where smarter estimators get more votes than dumber ones - e.g. Ensemble Bayesian Model Averaging (EBMA).
- When we do **ensembles** of identical models where the only difference between them is that they are trained on bootstrapped samples, this is called a **bootstrapped aggregation** (ensemble). Or, to keep it short, we call this **bagging.**

The final twist - Random Forests

- When looking at which predictors to attempt to split, only take a sample of them.
- If there are M predictors in X, look only at a random sample of size $m = \sqrt{M}$ from them for each split.
- E.g. If you are predicting conflict as a function of GDP, literacy, democracy and repression, at each split only consider two of them randomly (e.g. GDP and literacy).
- Imagine having an uncorrelated, noisy predictor, and a highly correlated, noisy predictor. What happens here?

And it looks like this (100 trees)!



Advantages:

- No parametric assumptions what-so-ever. You don't even need to linearize data! If trees can grow enough, they can fit any function.
- No distributional assumptions what-so-ever - remember data is bootstrapped for every tree, meaning you don't need to assume much.
- Fully interactive - each split in each tree can be a different variable - e.g. "if GDP ¿ 10000 AND infant mortality ¡ 0.002 AND democracy ¿ 0.9 then conflict = 0

Advantages:

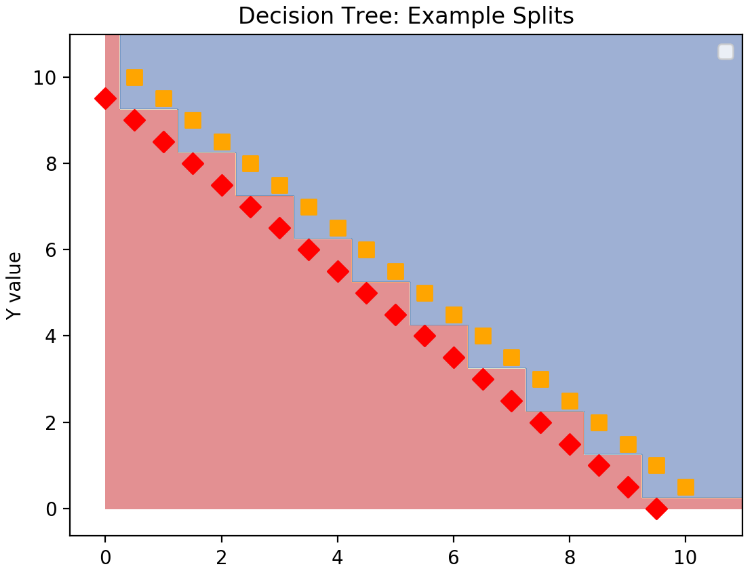
- Theoretically proven as almost impossible to over-fit, with either increase in number of trees or increase in number of features (even irrelevant features)(Breiman, 2001).[2].

[2] Yes, you can throw every variable in, but computational complexity increases rapidly

Disadvantages:

- Since it is a series of splits it will always discretize data and predictions. Will perform worse than a GLM when data comes from a GLM-like DGP!
- Needs at least for predictor to work (since it samples the predictors). No maximum limit to number of predictors.
- Is quite data hungry - remember each split in each tree reduces the data available by (approximately) half. $n > 100$ for classification or $n > 500$ for regression needed.
- Slower by orders of magnitude as opposed to a GLM.

Disadvantages:



Model hyper-parameters:

Unlike most regression models, there are three hyper-parameters that need to be specified by the modeller. These are the most important:

- *Number of trees (ntrees)* : how many trees should be included in the ensemble? Rule of thumb - greater is generally better - 100 is a reasonable minimum.
- *Nodesize* : minimum number of observations that can exist on the terminal leaf of a tree. A lower nodesize means a larger-grown tree, with more splits. Default is 1 (fully grown trees). You can also control this by specifying how many terminal nodes you allow at most in a tree.
- The larger the number of trees is, the more the ensemble can compensate for overfitting of individual trees. Thus, a smaller nodesize should require a larger number of trees.
- The larger a tree can grow (the lower the nodesize), the more complicated representations (functional forms) it can find between any x_i and y.

Model hyper-parameters/2:

How to find the best hyper-parameters?

- 1. Create lists of potential (test) hyper-parameters values separated by a reasonable step size. For example, values for ntrees of $(50, 100, 150, 200, 250)$ and nodesizes of $(1, 2, 5, 10)$.
- 2. Create a matrix containing all the combinations of the above lists.
- 3. For each combination above (e.g. ntree=50 and nodesize=1) do a cross-validation training and testing of a random forest. This is usually done using a $k = 5$ (split the data in 5, use 4 parts for training, one for test). Evaluate using a metric of choice (AUC, ROC, Precision, Recall etc.).
- 4. Choose the combination that produces the best metric.

This procedure is called `GridSearch`, and implemented in `R` by `caret` and in `Python` by `GridSearchCV`.

Making predictions:

Continuous response (regression problem): Given B tree estimators f_b, each trained on a bootstrapped sample $b \in B$, the predicted value of the ensemble given new data x' is:

$$\hat{y}' = \frac{1}{B} \sum_{b=1}^{B} f_b(x')$$

Making predictions:

For binary or categorical data - i.e. classification problems, the vote is computed as such. First, the share of votes ($\hat{v}'$) is computed, using the following formula, where $f_b(x')$ returns 1 if the tree b predicts the element x belongs to a class (e.g. "conflict") and 0 otherwise.

$$\hat{v}' = \frac{1}{B} \sum_{b=1}^{B} f_b(x')$$

A cutoff C is chosen so that:

$$y' = \begin{cases} 1 & \text{if } \hat{v}' \geq C \\ 0 & \text{if } \hat{v}' < C \end{cases}$$

Choices for C can either be natural, majority decision (0.5) or in order to maximize a performance metric (such as precision or recall). Seen this before? Yup - lecture 4!

Making predictions:

For binary data - i.e. classification problems, predicted pseudo-probabilities can be computed in two ways:

- 1. The proportion of votes each classifier makes for a class (i.e. if 25% of the trees predict "conflict", the predicted pseudo-probability of conflict of the bagged ensemble is 25%). If we have two trees, one predicting a probability of conflict of 75% and another one predicting 55%, and the cutoff is set to 50% the total probability would be, in this case, 100% (two classifiers voting "1"). In R the `randomForest` package does this.

$$P(y' = 1) = \frac{1}{B} \sum_{b=1}^{B} f_b(x')$$

Making predictions:

- 2. The mean of all predicted probabilities from each trees tree. If we have two trees, one predicting a probability of conflict of 75% and another one predicting 55%, the total probability would be 65%. In `R` the `ranger` package does this; in `Python scikit-learn`, this is the default behavior:

$$P(y' = 1) = \frac{1}{B}\sum_{b=1}^{B} P(y' = 1 | f_b, x')$$

Prediction Uncertainty:

For each predicted value, a standard deviation σ can be computed by computing the sum of squared differences between the overall random forest prediction and each tree's individual prediction:

$$\sigma = \sqrt{\frac{\sum_{b=1}^{B}(f_b(x') - \hat{y}')^2}{B-1}}.$$

Semi-in-sample evaluation: Out-of-bag (oob)

- Remember that only 63% of the data is used by each tree in the forest.
- For each observation x_i, take all trees that do not contain x_i and use only those to make a prediction $\hat{y}_i$. Compare that to the actual value of y_i, and calculate the error using a metric of preference (binary outcome : AUROC, AUPR, precision, recall...; continuous outcome : MSE, RMSE, MAE etc.).
- Fast to compute, but will under-estimate predictive power of the model.
- No useful fully in-sample evaluation - remember there are neither any distributional assumption for Y nor any expected functional forms - so we cannot compute p-values.

Out-of-sample Evaluation

- Random forests can be evaluated in the same way as any other model. The model takes a vector of predictors for each observation (x_i) and produces a prediction $\hat{y}_i$ that can be compared to an actual y.
- This can be done by using cross-validation. Split the dataset into k fractions. Train on $k-1$ fractions and evaluate on 1 fraction. Change the evaluation (test) fraction k times, until all the data has been in the test fraction at least once.
- Or this can be done by using forecasting for time-series data - Split the dataset into two partitions - one preceding the other temporally (using 80% for train and 20% for test). Train on the train partition, and forecast on the test partition.
- Evaluate using a metric of choice (binary outcome : AUROC, AUPR, precision, recall...; continuous outcome : MSE, RMSE, MAE etc.).

Variable Importance Scores

- Computes the relative importance of a variable in the overall forest.
- For a variable j, permute the values in the dataset randomly (the idea is randomly shuffling values will reduce j's expected predictive power to 0).
- Using the already trained trees, calculate new predictions (with j predictive power destroyed), and evaluate against normal predictions (with j predictive power intact).
- The higher the increase in error, the more important the variable j is.
- This is usually done o.o.b.

Variable Importance Scores/2

Since each variable is permuted "on its own", and trees are not retrained, the variable importance score always assumes each variable is independent of all the other variables in the model.
If not independent (e.g. colinear), dropping/permuting a variable would allow other variables containing the same information to replace the "lost" variable when fitting the model from scratch.
N.B. Colinearity and interactivity do not harm random forest's predictive power, but can harm your interpretation of

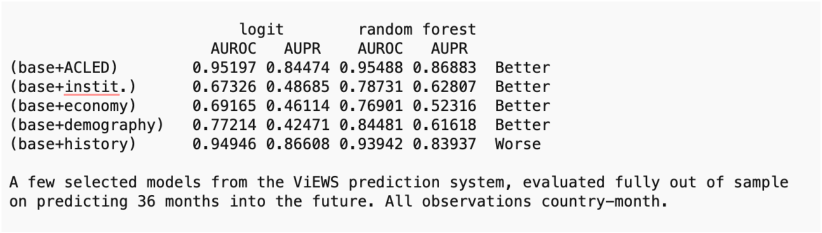
Partial dependency plot

Same as marginal effects / quantities of interest for GLMs:

$$\hat{f}_{x_S}(x_S) = \frac{1}{B}\sum_{i=1}^{B}\hat{f}(x_S, x_C^{(i)})$$

Holding the predicted values for a vector C of independent covariates at a constant value (usually the mean or median), compute the predicted values for the whole range of the variable of interest S. Interpretation is the same as in the case of GLM, as are assumptions of independence.

Application - Predicting conflict - ViEWS

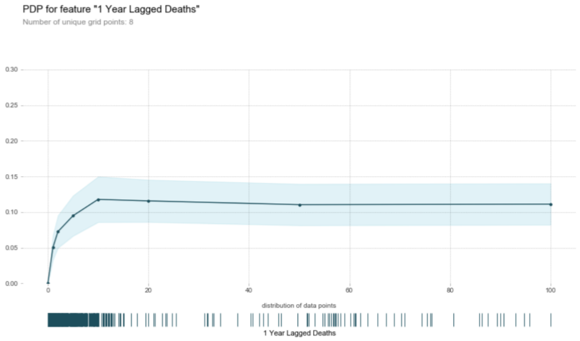


```
                          logit          random forest
                       AUROC    AUPR     AUROC    AUPR
(base+ACLED)         0.95197 0.84474 0.95488 0.86883  Better
(base+instit.)       0.67326 0.48685 0.78731 0.62807  Better
(base+economy)       0.69165 0.46114 0.76901 0.52316  Better
(base+demography)    0.77214 0.42471 0.84481 0.61618  Better
(base+history)       0.94946 0.86608 0.93942 0.83937  Worse
```

A few selected models from the ViEWS prediction system, evaluated fully out of sample on predicting 36 months into the future. All observations country–month.

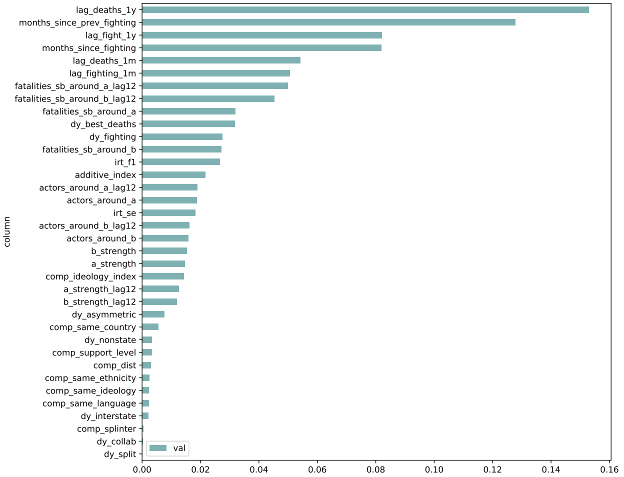
Predicting conflict, 36-months ahead of time, trained on monthly country data starting in 1989. Comparison between logit models using various combinations of predictors and random forests with the same number of predictors (obs=59240, predictors=278). (Hegre et al., 2019)

An example of a PDP from ViEWS



A partial dependency plot showing the relationship between fatalities (1-year lag) and conflict in the next year. Observe that the relationship is almost lograithmical - this functional form was detected by the random forest, not fed into the model by hand!

An example of a VIS plot from ViEWS



Variable importance scores from a small RF model in the (not yet released) actor model set of ViEWS.

Bibliography I

Breiman, Leo. 2001. "Random forests." *Machine learning* 45(1):5–32.

Hegre, Håvard. 2001. "Toward a democratic civil peace? Democracy, political change, and civil war, 1816–1992." *American political science review* 95(1):33–48.

Hegre, Håvard, Marie Allansson, Matthias Basedau, Michael Colaresi, Mihai Croicu, Hanne Fjelde, Frederick Hoyles, Lisa Hultman, Stina Högbladh, Remco Jansen et al. 2019. "ViEWS: A political violence early-warning system." *Journal of Peace Research* p. 0022343319823860.

Jones, Zachary M and Yonatan Lupu. 2018. "Is There More Violence in the Middle?" *American Journal of Political Science* 62(3):652–667.

Von Uexkull, Nina, Mihai Croicu, Hanne Fjelde and Halvard Buhaug. 2016. "Civil conflict sensitivity to growing-season drought." *Proceedings of the National Academy of Sciences* 113(44):12391–12396.