Rating Wine Quality Using a Random Forest Classifier

Brian Miller

Overview

Quick intro to random forest models

- Decision Trees
- How to: Build Random Forest
- How to: Use Random Forest

Review of the data

Selecting the best model for our data

Parameter tuning

Evaluating the model/ explore wine results

Decision Trees Explained

Visual representation of a categorical outcome

- The top node explains the most variation in the data
- Not flexible
- In real world applications, only works for small datasets

Loves Soda	Age	Loves Cool As Ice
Yes	7	No
No	12	No
Yes	18	Yes
Yes	35	Yes
Yes	38	Yes
No	50	No
No	83	No

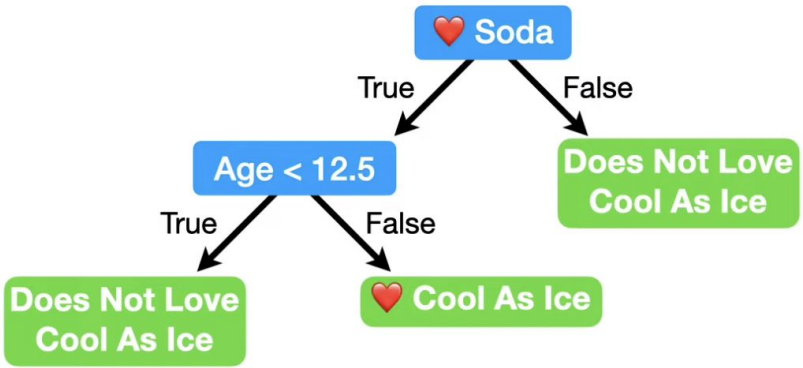
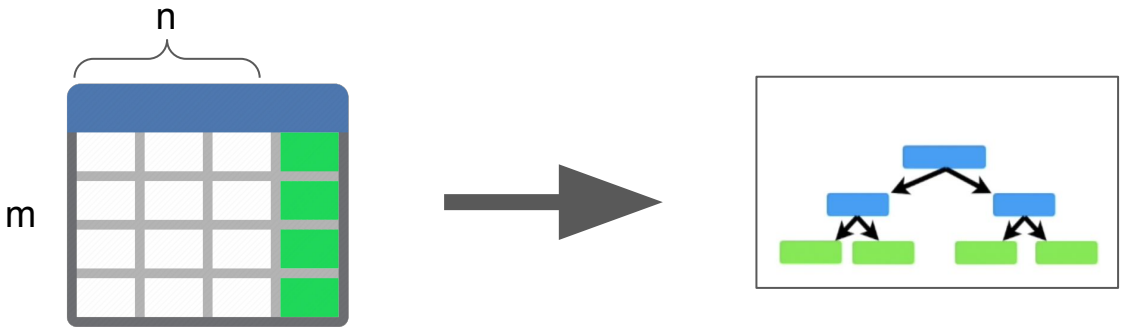


Figure Source: StatQuest

Random Forest Explained: Building your Tree



Step1: Make bootstrapped dataset (same dim as original data)

Step2: Make decision tree

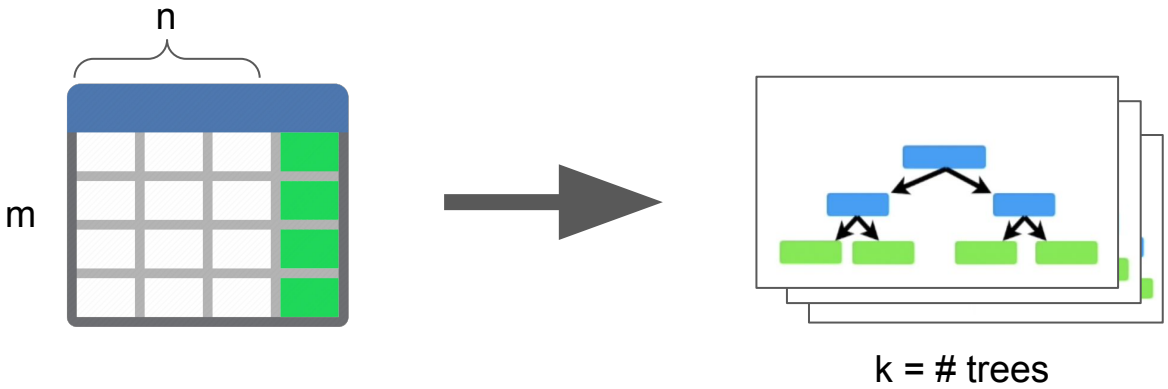
for each branching point:

only consider a random subset of **j** variables at each step

$$j = \sqrt{n}$$

Figure Source: StatQuest

Random Forest Explained: Building your Forest



Step1: Make bootstrapped dataset (same dim as original data)

Step2: Make decision tree

for each branching point:

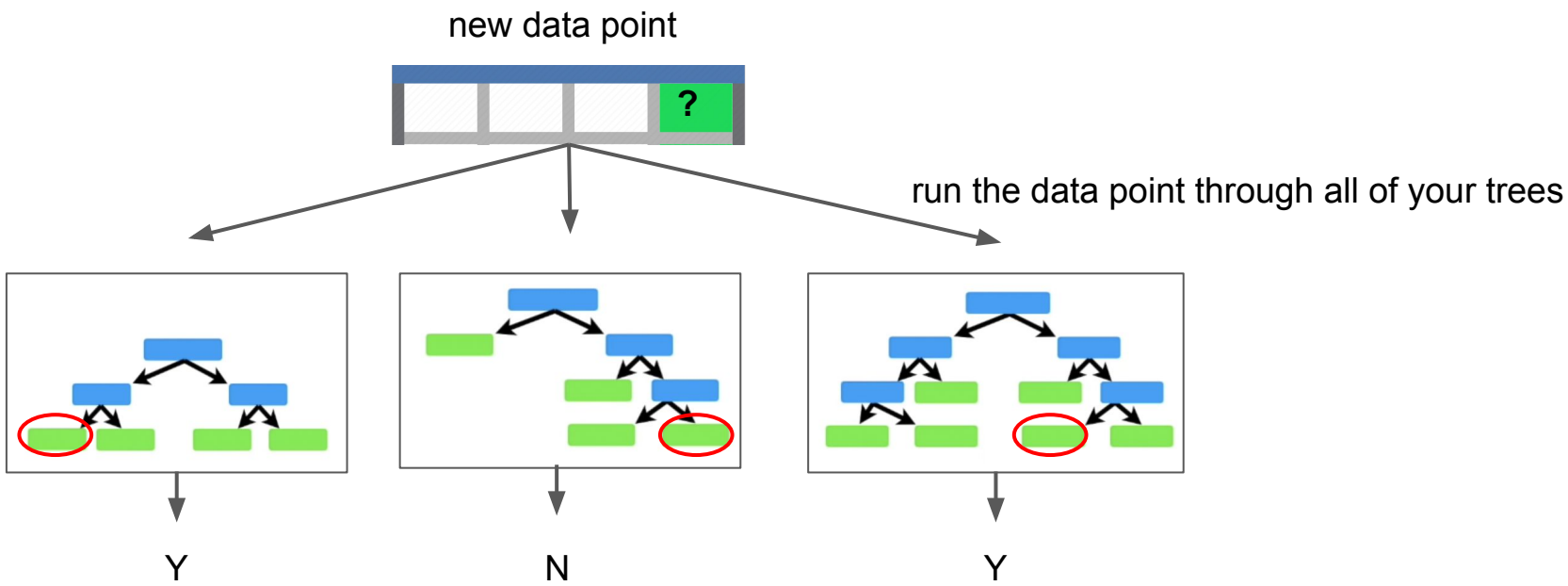
only consider a random subset of **j** variables at each step

repeat **k** times

$$j = \sqrt{n}$$

Figure Source: StatQuest

Random Forest Explained: Using your Random Forest



⅔ trees result in Y, so the outcome is Y (majority vote)

Figure Source: StatQuest

Random Forest Explained: Summary

Flexible classification model that uses many different decision trees

Tree to tree variation comes from:

- Data bootstrapping
- Variable sampling at each branch

Requirements:

No missing values/ NAs

No obvious data outliers

Balanced outcomes (e.g. 50/50 Y/N data)

Main parameters to tune:

k = number of trees

j = number of variables to consider per branch

nodesize = refers to how many observations we want in the terminal nodes (controls tree depth)

Wine Quality Data

Data for 1599 red wine variants from Portugal

- quality scores are the average of three professional wine tasters
- 11 descriptor variables
- No missing/NA data

Sadly there is no grape type, brand data, sale price (data protection legality reasons)

Data Source:

University of California, Irvine

Center for Machine Learning and Intelligent Systems

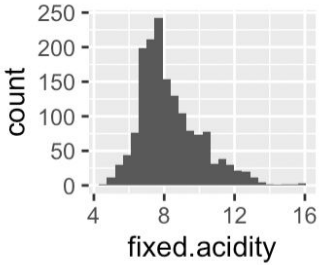
https://archive.ics.uci.edu/ml/datasets/wine+quality

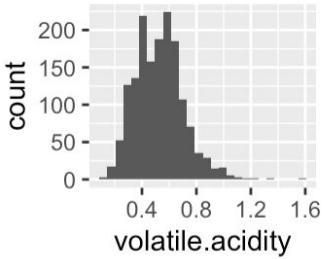
...

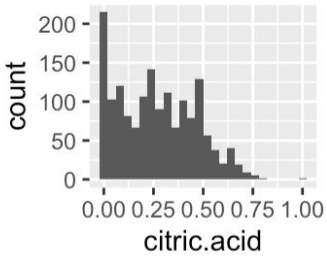
total.sulfur.dioxide	density	pH	sulphates	alcohol	quality
34	0.9978	3.51	0.56	9.4	5
67	0.9968	3.20	0.68	9.8	5
54	0.9970	3.26	0.65	9.8	5
60	0.9980	3.16	0.58	9.8	6
34	0.9978	3.51	0.56	9.4	5
40	0.9978	3.51	0.56	9.4	5
59	0.9964	3.30	0.46	9.4	5
21	0.9946	3.39	0.47	10.0	7
18	0.9968	3.36	0.57	9.5	7

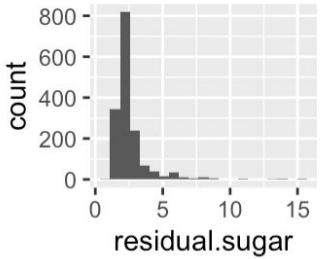
...

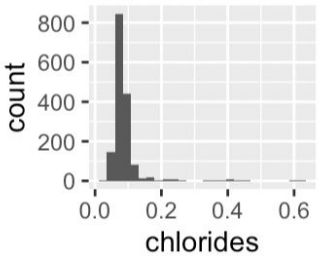
Wine Quality Data

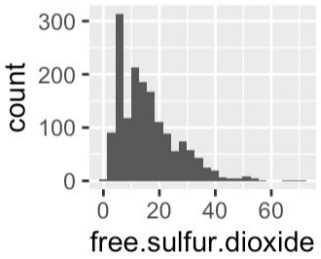


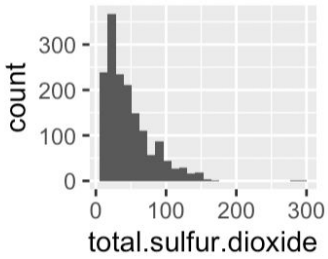


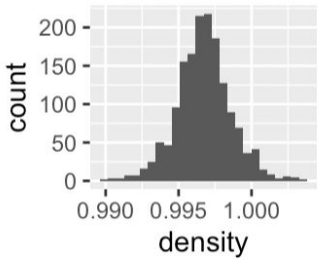


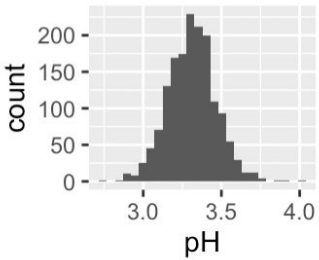


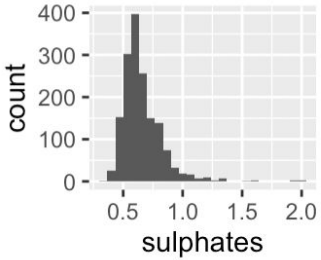


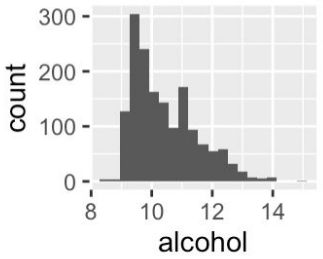


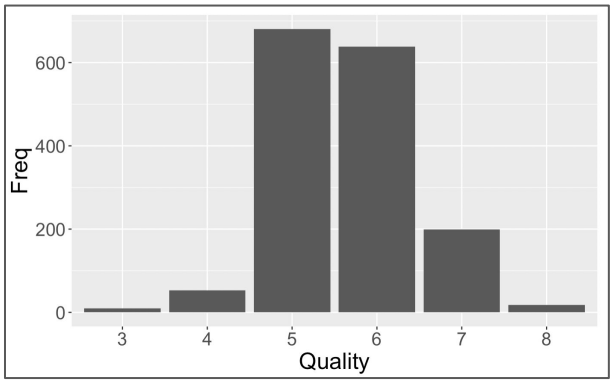












Random Forest Parameter Tuning

j start with $j = \sqrt{n}$ then try + or - 1 → $\sqrt{11} = 3.32$

mtry	OOBError
2	0.1657286
3	0.1701063
4	0.1707317

→ **j** = 2

k {100, 200, 300, … 1000}

nodesize {1,2,3, … 10}

→ try all combinations

→ minimal OOB for <u>all</u> quality score outcomes

→ **k** = 500
nodesize = 2

OOB Error = Out of Bag Error (take data points excluded during bootstrapping, run them through your final model)

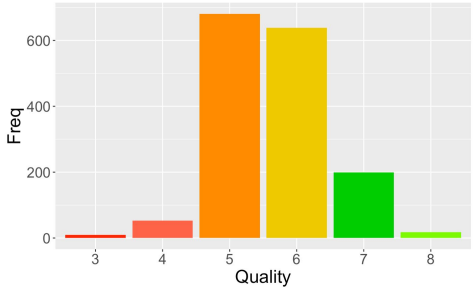
Main parameters to tune:

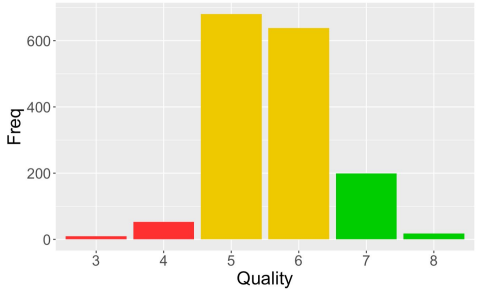
k = number of trees

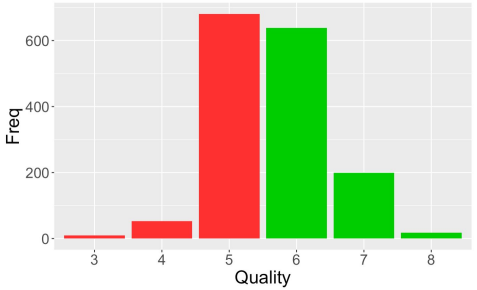
j = number of variables to consider per branch

nodesize = refers to how many observations we want in the terminal nodes (controls tree depth)

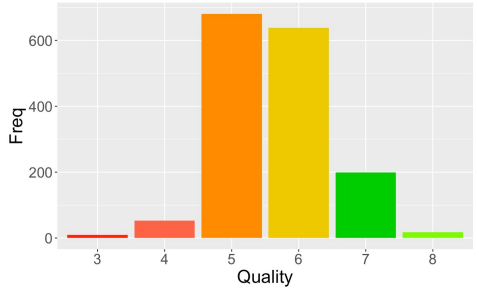
Outcome Binning Affects Model Performance



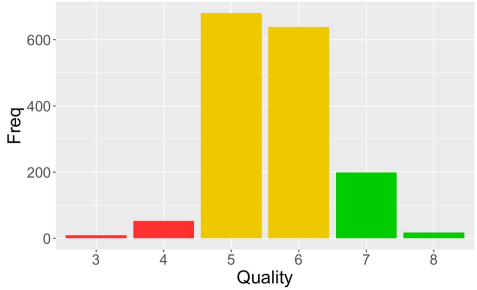




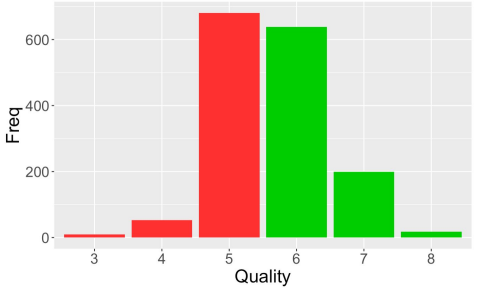
Outcome Binning Affects Model Performance



wine_score	n_total	n_training	n_testing	was_correct	was_incorrect	precision
3	10	8	2	0	2	0.0000000
4	53	48	5	0	5	0.0000000
5	681	538	143	114	29	0.7972028
6	638	520	118	90	28	0.7627119
7	199	149	50	28	22	0.5600000
8	18	16	2	0	2	0.0000000

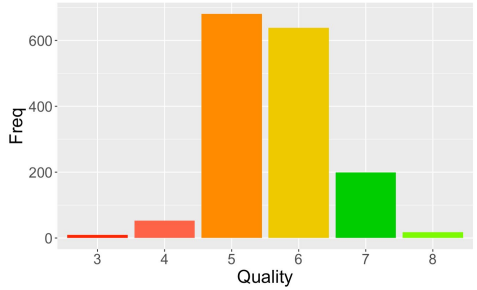


wine_score	n_total	n_training	n_testing	was_correct	was_incorrect	precision
bad(3,4)	63	49	14	0	14	0.0000000
ok(5,6)	1319	1060	259	256	3	0.9884170
good(7,8)	217	170	47	20	27	0.4255319

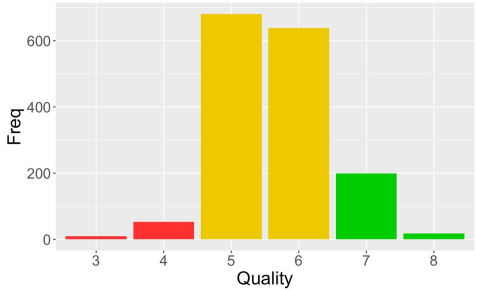


wine_score	n_total	n_training	n_testing	was_correct	was_incorrect	precision
bad(3-5)	744	594	150	122	28	0.8133333
good(6-8)	855	685	170	148	22	0.8705882

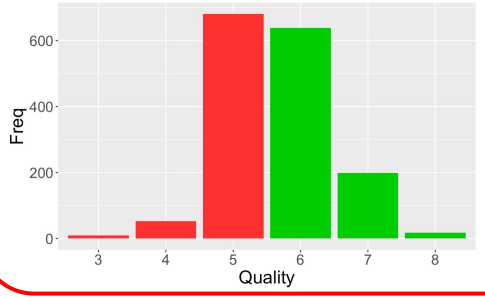
Outcome Binning Affects Model Performance



wine_score	n_total	n_training	n_testing	was_correct	was_incorrect	precision
3	10	8	2	0	2	0.0000000
4	53	48	5	0	5	0.0000000
5	681	538	143	114	29	0.7972028
6	638	520	118	90	28	0.7627119
7	199	149	50	28	22	0.5600000
8	18	16	2	0	2	0.0000000

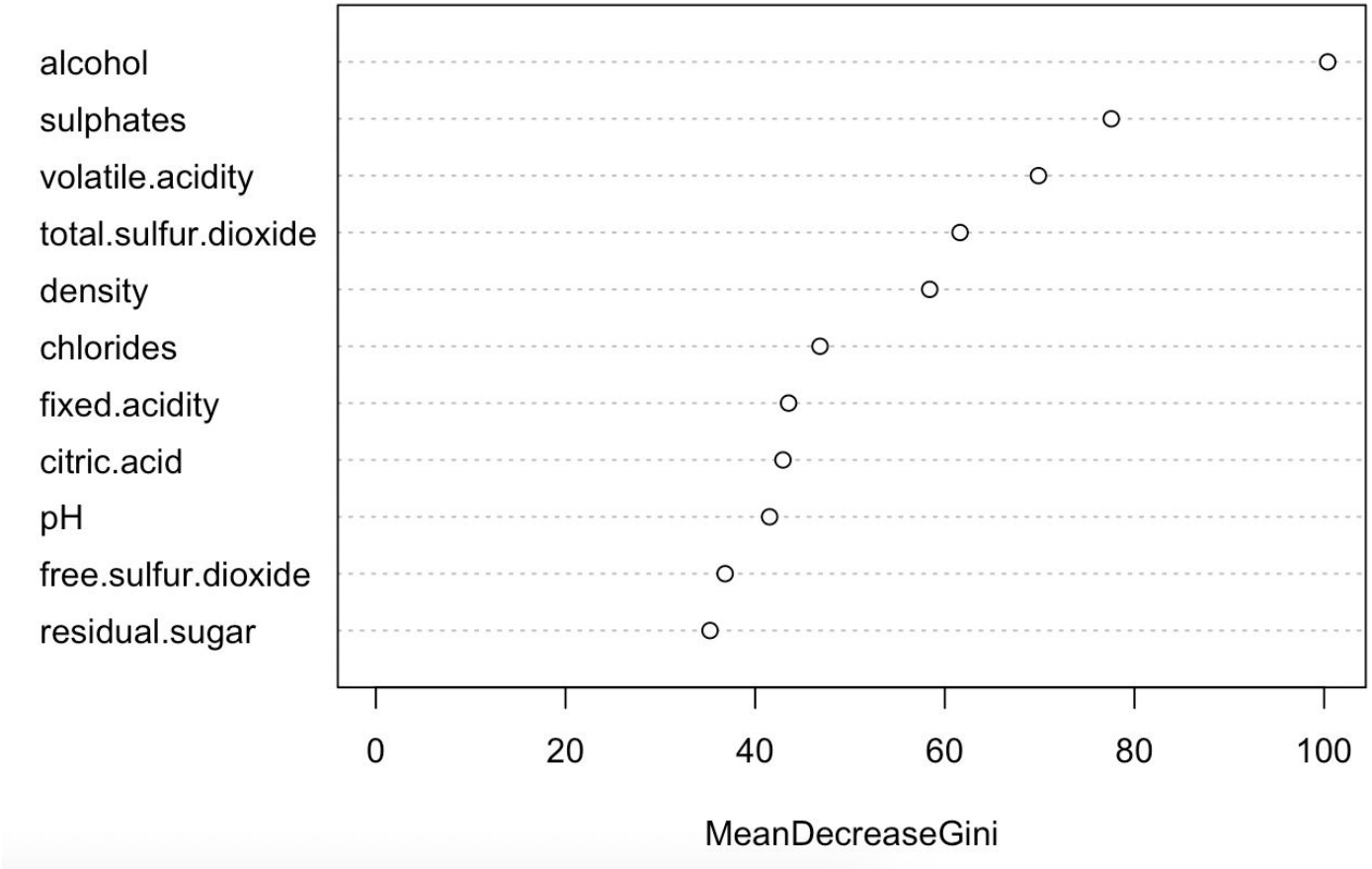


wine_score	n_total	n_training	n_testing	was_correct	was_incorrect	precision
bad(3,4)	63	49	14	0	14	0.0000000
ok(5,6)	1319	1060	259	256	3	0.9884170
good(7,8)	217	170	47	20	27	0.4255319



wine_score	n_total	n_training	n_testing	was_correct	was_incorrect	precision
bad(3-5)	744	594	150	122	28	0.8133333
good(6-8)	855	685	170	148	22	0.8705882

Using Alcohol Content Improves Our Model the Most

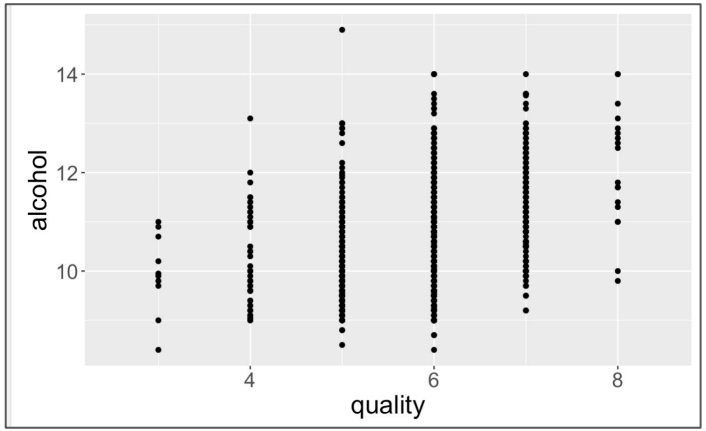


Gini Impurity = A measure of how bad a given tree did at incorporating all outcomes (splitting performance)

alcohol

Directly correlated with higher wine scores

Molecules: Ethanol

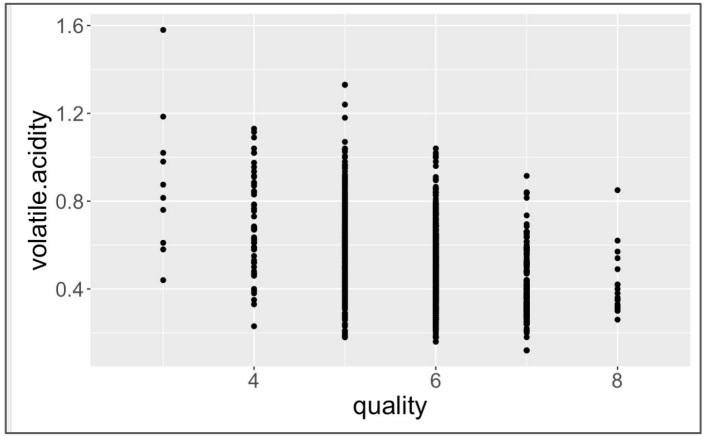


volatile.acidity

Negatively correlated with higher wine scores

Molecules: various acids (acetic, lactic, formic)

tastes and smells like: pungent, vinegar (nail polish remover)



Summary

Random forest model is a classifier that internally uses many different bootstrapped decision trees

Random forests requires: no missing values, balanced outcomes

Best model used a simple binary wine classification (good or bad)

Higher wine scores were associated with more alcohol and less acidity