University of Washington
Department of Statistics
Technical Report No. 645

The "balanced-ness" of classification trees

Steven B. Gillispie
University of Washington, Statistics

October 4, 2017

1 Introduction

A classification tree is a logical tree (in the computer science sense) that encodes an order of instructions used to sort a set of data into n classes. It is a rooted binary (directed-edge) tree with n labeled leaves, one leaf for each of the n classes into which the data are sorted. There are $(2n-3)!!$ possible labeled classification trees, up to isomorphism, each tree representing a different set of encoded instructions. (The double factorial $n!!$ is recursively defined as $n!! \equiv n(n-2)!!$ with $0!! = 1!! = 1$.)

Suppose a supplied data set X to be sorted into n classes has the data evenly divided among the n classes. Attached with X is a feature set F assumed to be capable of producing a correct classification. A classification algorithm will attempt to maximally divide X as evenly as possible using one of the features. Then, repeating this process, each of the two separated halves of X will be evenly divided, on and on, so that the tree representing this process will be very *balanced*. By balanced is meant that the n unique paths from the root of the tree to each of the n leaves will all have the same (or very close to the same) number of edges in them. On the other hand,

if the data set X is very uneven, for example half of the data are in the first class, half of the rest in the second, and so on, then it is likely that the classification algorithm will produce a very unbalanced tree with the first split going entirely into the first leaf, while the second branch of the tree splits off half of its data into the second leaf, and so on. Thus the "balancedness" of a classification tree should be highly correlated with the unevenness of the data distribution among the classes.

Suppose we choose to measure this balance–imbalance of a tree using the variance of its set of n path lengths (in number of edges) from the root to each of its n leaves. Among the $(2n-3)!!$ such trees, each with such a variance, there will be a maximum variance. Therefore we can normalize this set of variances by dividing by its maximum, creating an independent variable over the set $[0,1]$, and then produce a histogram of the number of classification trees occurring at each generated value of normalized variance. What would the resulting probability distribution look like for various values of n? That is the question we will try to answer here.

2 Methods

For any particular n we will have $(2n-3)!!$ labeled classification trees to consider. (This value was determined by Schröder in 1870; its derivation is reproduced in the Appendix at the end of this report.) Each such tree will have an underlying unlabeled rooted binary tree, but no known formula exists for their count. Since it would seem that the best way to generate the probability distributions we seek is to use a computer, we will need an algorithm. The labeled trees can be kept track of efficiently by recording the unlabeled trees along with a count of the number of ways they each can be labeled. For that we need a notation. Let

$$(c; p_1, p_2, \ldots, p_n)$$

represent an unlabeled tree whose n path lengths from the root to each of the n leaves (from left to right) are the $p_1, p_2, \ldots, p_n$ values while the count c represents the number of labeled trees having that path length pattern.

We now propose a recursive algorithm. For $n = 2$ we will have a single tree (up to isomorphism) which will be $(1; 1, 1)$. Suppose for any $n > 2$ we have a list of all unlabeled trees having $2 \leq k < n$ leaves. We first create the new n-trees whose left branch is a single leaf attached to the root (having

edge length 1) while the right branch leads to a node that is the root of a subtree having $n-1$ leaves. There are n ways to label the single leaf, and simultaneously n ways to distribute the remaining $n-1$ labels over the right subtree, so to begin the new list of n-trees we simply multiply all the known trees having $n-1$ leaves (from our recursive list) by n (to represent their n possible re-labelings), add 1 to each of their $p_1, p_2, \ldots, p_{n-1}$ path lengths, and then insert a single edge leaf. As an example, for $n = 3$ we find $(3; 1, 2, 2)$ to be the only possible non-isomorphic addition to our list.

This illustrates the first point we must keep in mind when creating new trees: if we proceed beyond $[n/2]$ (where $[x]$ represents the largest integer less than or equal to x) leaves in our left tree then we are duplicating already counted trees. For $n = 3$ this is equivalent to $\{1, 2\}$ being the same as $\{2, 1\}$. In other words, for $n > 2$ we need to put only $1, 2, \ldots, [n/2]$ leaves into the left branch and then put the remainder into the right branch.

The situation is more complicated for $n = 4$ because then we have the new case of splitting the eventual leaf set into $\{2, 2\}$ as well as $\{1, 3\}$. This brings up the second point to remember: when n is even the $\binom{n}{n/2}$ number of ways to split the leaf numberings double-counts the symmetric splitting of n into two halves. That is, since the tree will be evenly balanced each tree will have an isomorphism using the other half of the numberings. Thus we must divide $\binom{n}{n/2}$ by 2 in this special case.

As a result, for $n = 4$ we first produce $(12; 1, 2, 3, 3)$ using the single leaf additions and then go on to create $(3; 2, 2, 2, 2)$. The first new tree is obtained by multiplying the numberings count of $\binom{4}{1} = 4$ by the count $c = 3$ for the 3-tree $(3; 1, 2, 2)$. The second tree has $\binom{4}{2}/2 = 3$ multiplied by $1 \cdot 1$ from the two 2-tree counts, while each of the 2-trees have 1 added to their path lengths to produce the new $(3; 2, 2, 2, 2)$ tree. As a check we see that $12 + 3 = 15 = 5!! = (2 \cdot 4 - 3)!!$ so that we have accounted for all 15 possible labeled trees.

For $n > 4$ we follow this same pattern, pairing each possible left subtree with each possible right subtree, up through the split of $[n/2]$ leaves in the left subtree. Table 1 shows the result of continuing this process to $n = 8$, with u_n and t_n representing the number of unlabeled and labeled trees, respectively.

To prove our algorithm correct we must establish the two classic issues of existence and uniqueness: do we find every tree, and do we find it only once. Consider the unlabeled trees first. Any such tree can be dismantled first by removing the root and its two descending edges, leaving behind two

n	u_n	t_n	Classification trees
2	1	1	$(1;1,1)$
3	1	3	$(3;1,2,2)$
4	2	15	$(12;1,2,3,3), (3;2,2,2,2)$
5	3	105	$(60;1,2,3,4,4), (15;1,3,3,3,3), (30;2,2,2,3,3)$
6	6	945	$(360;1,2,3,4,5,5), (90;1,2,4,4,4,4), (180;1,3,3,3,4,4),$ $(180;2,2,2,3,4,4), (45;2,2,3,3,3,3), (90;2,3,3,2,3,3)$
7	11	10395	$(2520;1,2,3,4,5,6,6), (630;1,2,3,5,5,5,5),$ $(1260;1,2,4,4,4,5,5), (1260;1,3,3,3,4,5,5),$ $(315;1,3,3,4,4,4,4), (630;1,3,4,4,3,4,4),$ $(1260;2,2,2,3,4,5,5), (315;2,2,2,4,4,4,4),$ $(630;2,2,3,3,3,4,4), (1260;2,3,3,2,3,4,4),$ $(315;2,3,3,3,3,3,3)$
8	24	135135	$(20160;1,2,3,4,5,6,7,7), (5040;1,2,3,4,6,6,6,6),$ $(10080;1,2,3,5,5,5,6,6), (10080;1,2,4,4,4,5,6,6),$ $(2520;1,2,4,4,5,5,5,5), (5040;1,2,4,5,5,4,5,5),$ $(10080;1,3,3,3,4,5,6,6), (2520;1,3,3,3,5,5,5,5),$ $(5040;1,3,3,4,4,4,5,5), (10080;1,3,4,4,3,4,5,5),$ $(2520;1,3,4,4,4,4,4,4), (10080;2,2,2,3,4,5,6,6),$ $(2520;2,2,2,3,5,5,5,5), (5040;2,2,2,4,4,4,5,5),$ $(5040;2,2,3,3,3,4,5,5), (1260;2,2,3,3,4,4,4,4),$ $(2520;2,2,3,4,4,3,4,4), (10080;2,3,3,2,3,4,5,5),$ $(2520;2,3,3,2,4,4,4,4), (5040;2,3,3,3,3,3,4,4),$ $(5040;2,3,4,4,2,3,4,4), (1260;2,3,4,4,3,3,3,3),$ $(1260;3,3,3,3,2,3,4,4), (315;3,3,3,3,3,3,3,3)$

Table 1: The classification trees and their counts for $n = 2$ to 8.

subtrees. These can be recursively dismantled until only a single leaf is left. We can then reverse this process to reassemble the tree. But we see that any such tree will be included by our algorithm: a single leaf is not a relevant classification tree so the first possible tree is the 2-tree $(1; 1, 1)$, which our algorithm began with. Any other tree must have from 1 to $[n/2]$ leaves in the left branch (or be isomorphic to one) and so would be created by the algorithm. Clearly, any labeled tree will also be created as well.

For uniqueness, suppose two labeled trees A and B are identical and both are created by the algorithm. Both have root nodes and identical left and right branching edges, which we can remove. We are then left with four subtrees (left and right branches from both trees) with $A_l = B_l$ and $A_r = B_r$. But this is just a recursive application of the uniqueness problem, so we can continue until we are left with one of the (non-isomorphic) cases $(c; 1, 1)$, $(c; 1, 2, 2)$, or $(c; 2, 2, 2, 2)$ as the combined final left and right subtrees. (The first case only appears when it is the tree we start with.) But the algorithm specifically did not duplicate any of these cases, as we can see by studying the way it started. Therefore we have a contradiction in our assumption that $A = B$ are both produced by the algorithm and the only resolution is that no such case ever occurs.

For the unlabeled trees we see that we can count them much easier because we do not have to keep track of each smaller unlabeled tree but instead only their count. If we define $u_1 = 1$ for convenience then we have

$$u_n = \sum_{i=1}^{[n/2]} u_i u_{n-i}. \tag{1}$$

No special exception when $i = [n/2]$ is needed because that issue is related only to the number of labelings.

3 Results

The sequence of counts of unlabeled classification trees u_n in (1) starts from $n = 2$ as $(1, 1, 2, 3, 6, 11, 24, 47, 103, \ldots)$ and is OEIS (Sloane, 2017) sequence A000992. The sequence of labeled trees t_n is OEIS sequence A001147 and begins at $n = 2$ with $(1, 3, 15, 105, 945, 10395, 135135, 2027025, 34459425, \ldots)$. Clearly, it is the one growing the fastest! Table 2 shows the counts for $n = 2 \ldots 30$ of the number of unlabeled trees and $\log_{10}(2n - 3)!!$, the $\log_{10}$

n	u_n	$\log_{10} t_n$
2	1	0.00
3	1	0.48
4	2	1.18
5	3	2.02
6	6	2.98
7	11	4.02
8	24	5.13
9	47	6.31
10	103	7.54
11	214	8.82
12	481	10.14
13	1030	11.50
14	2337	12.90
15	5131	14.33
16	11813	15.79
17	26329	17.28
18	60958	18.80
19	137821	20.35
20	321690	21.91
21	734428	23.50
22	1721998	25.12
23	3966556	26.75
24	9352353	28.40
25	21683445	30.08
26	51296030	31.77
27	119663812	33.47
28	284198136	35.20
29	666132304	36.94
30	1586230523	38.69

Table 2: The counts u_n of unlabeled trees and $\log_{10}$ counts of labeled trees t_n for $n = 2$ to 30.

Quintile	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$
1	0.285714	0.142857	0.212121	0.142191	0.200000	0.183052
2	0.000000	0.190476	0.121212	0.279720	0.285315	0.293706
3	0.142857	0.190476	0.363636	0.242424	0.179021	0.266557
4	0.000000	0.095238	0.060606	0.149184	0.223776	0.164541
5	0.571429	0.380952	0.242424	0.186480	0.111888	0.092143

Table 3: The probability distribution quintiles of the labeled classification trees for $5 \leq n \leq 10$ using the path length variance as the measure of imbalance.

Quintile	$n = 5$	$n = 6$	$n = 7$	$n = 8$	$n = 9$	$n = 10$
1	0.000000	0.000000	0.030303	0.002331	0.005594	0.005759
2	0.000000	0.142857	0.000000	0.139860	0.093706	0.103250
3	0.285714	0.000000	0.303030	0.279720	0.229371	0.315097
4	0.142857	0.380952	0.363636	0.317016	0.335664	0.332374
5	0.571429	0.476190	0.303030	0.261072	0.335664	0.243521

Table 4: The probability distribution quintiles of the labeled classification trees for $5 \leq n \leq 10$ using the path length standard deviation as the measure of imbalance.

number of labeled trees. The number of unlabeled trees is approximately doubling with each increment of n, though it actually is seen to be increasing slightly faster than that. This number is important to the computer program (Gillispie, 2017) since it dictates how much computer memory is needed to hold the recursive list of tree path lengths.

As to the probability distributions, for class sizes of $n < 5$ they are quite uneven but can be calculated from Table 1 if desired. Otherwise, tables 3, 5, and 7 show the distribution quintiles (for $5 \leq n \leq 10$), deciles (for $11 \leq n \leq 15$), and vigesiles (20-iles) for $16 \leq n \leq 20$. These three tables all use the variance of the n path lengths as their proxy measure for the amount of imbalance in the trees. But the standard deviation of the path lengths is also a legitimate measure, so Tables 4, 6, and 8 show the equivalent results using the standard deviation instead.

Decile	$n = 11$	$n = 12$	$n = 13$	$n = 14$	$n = 15$
1	0.045011	0.041609	0.039863	0.053590	0.053050
2	0.120981	0.146906	0.183182	0.192823	0.224256
3	0.176232	0.227673	0.237294	0.223039	0.236120
4	0.208145	0.188344	0.134608	0.189743	0.182372
5	0.125744	0.108053	0.167990	0.144127	0.118455
6	0.118123	0.137176	0.103532	0.070211	0.094381
7	0.049536	0.047903	0.062150	0.073399	0.049057
8	0.102882	0.054435	0.039382	0.024811	0.019719
9	0.015242	0.026129	0.019691	0.021365	0.017230
10	0.038104	0.021774	0.012307	0.006892	0.005360

Table 5: The probability distribution deciles of the labeled classification trees for $11 \leq n \leq 15$ using the path length variance as the measure of imbalance.

Decile	$n = 11$	$n = 12$	$n = 13$	$n = 14$	$n = 15$
1	0.000000	0.000000	0.000000	0.000038	0.000003
2	0.003334	0.001599	0.002981	0.003457	0.003709
3	0.038819	0.037152	0.031633	0.042321	0.042099
4	0.069540	0.111115	0.111013	0.113889	0.120112
5	0.164801	0.179635	0.179643	0.217224	0.213565
6	0.201000	0.154731	0.228987	0.228036	0.234181
7	0.147178	0.228354	0.203142	0.183712	0.189551
8	0.207668	0.158949	0.142915	0.128965	0.119436
9	0.114313	0.089273	0.072611	0.061683	0.061261
10	0.053346	0.039193	0.027075	0.020676	0.016081

Table 6: The probability distribution deciles of the labeled classification trees for $11 \leq n \leq 15$ using the path length standard deviation as the measure of imbalance.

Vigesile	$n = 16$	$n = 17$	$n = 18$	$n = 19$	$n = 20$
1	0.007846	0.006204	0.007107	0.007054	0.008525
2	0.051661	0.059599	0.064649	0.074693	0.077998
3	0.101387	0.117837	0.124526	0.137138	0.144046
4	0.127591	0.141741	0.143980	0.151326	0.159502
5	0.140274	0.117688	0.142060	0.134005	0.143479
6	0.108260	0.127201	0.115570	0.121266	0.121152
7	0.106745	0.094459	0.096319	0.101290	0.092030
8	0.073527	0.089307	0.083477	0.072123	0.074029
9	0.080874	0.062107	0.060041	0.062729	0.057749
10	0.045669	0.056196	0.050268	0.041921	0.039603
11	0.048402	0.033414	0.035010	0.034530	0.029661
12	0.034750	0.032980	0.026219	0.021030	0.019291
13	0.017454	0.020476	0.020339	0.016420	0.013181
14	0.022181	0.014669	0.010616	0.009272	0.008681
15	0.014365	0.011237	0.008949	0.007345	0.005182
16	0.004859	0.005387	0.004676	0.003516	0.002932
17	0.007816	0.004634	0.002907	0.002178	0.001252
18	0.003380	0.003244	0.001896	0.001406	0.001150
19	0.000845	0.000463	0.000758	0.000412	0.000371
20	0.002112	0.001158	0.000632	0.000343	0.000185

Table 7: The probability distribution vigesiles of the labeled classification trees for $16 \leq n \leq 20$ using the path length variance as the measure of imbalance.

Vigesile	$n = 16$	$n = 17$	$n = 18$	$n = 19$	$n = 20$
1	0.000000	0.000000	0.000000	0.000000	0.000000
2	0.000012	0.000000	0.000001	0.000001	0.000001
3	0.000191	0.000195	0.000285	0.000133	0.000219
4	0.002918	0.002225	0.003086	0.003044	0.003010
5	0.011740	0.012899	0.011707	0.017095	0.015421
6	0.030439	0.033683	0.038422	0.038778	0.049719
7	0.061176	0.065417	0.071903	0.074350	0.084832
8	0.088712	0.092996	0.104216	0.113589	0.112798
9	0.105999	0.122044	0.118644	0.128704	0.134021
10	0.127573	0.113610	0.134058	0.128522	0.133530
11	0.109950	0.130385	0.120181	0.126118	0.125548
12	0.119328	0.112161	0.110657	0.112228	0.104740
13	0.094632	0.094642	0.095150	0.086133	0.084734
14	0.086004	0.083683	0.070686	0.066440	0.063018
15	0.062225	0.051638	0.050647	0.048283	0.041995
16	0.042461	0.040213	0.036053	0.029459	0.024691
17	0.030551	0.023763	0.018885	0.015421	0.013094
18	0.012569	0.012106	0.009794	0.007962	0.006042
19	0.010562	0.006719	0.004234	0.002984	0.002030
20	0.002957	0.001622	0.001390	0.000755	0.000556

Table 8: The probability distribution vigesiles of the labeled classification trees for $16 \leq n \leq 20$ using the path length standard deviation as the measure of imbalance.

n	μ	σ^2	σ	σ/μ
2	0.000000	0.000000	0.000000	$----$
3	0.222222	0.000000	0.000000	0.000000
4	0.550000	0.075625	0.275000	0.500000
5	0.937143	0.253649	0.503636	0.537416
6	1.362434	0.552578	0.743356	0.545609
7	1.814471	0.987275	0.993617	0.547607
8	2.286422	1.570183	1.253070	0.548048
9	2.773858	2.311932	1.520504	0.548155
10	3.273739	3.221727	1.794917	0.548277
11	3.783880	4.307636	2.075485	0.548507
12	4.302654	5.576796	2.361524	0.548853
13	4.828815	7.035578	2.652466	0.549300
14	5.361384	8.689709	2.947831	0.549827
15	5.899578	10.544374	3.247210	0.550414
16	6.442757	12.604291	3.550252	0.551046
17	6.990394	14.873783	3.856654	0.551708
18	7.542047	17.356820	4.166152	0.552390
19	8.097340	20.057073	4.478512	0.553084
20	8.655955	22.977944	4.793531	0.553784

Table 9: The means, variances, standard deviations, and relative standard deviations of the distributions of labeled classification trees shown in Tables 3, 5, and 7 that use the path length variance as their measure of imbalance.

Finally, Tables 9 and 10 show various statistics associated with the distributions in Tables 3 through 8.

4 Discussion

In Table 2 it is clear that the average number of labelings per unlabeled tree is increasing as well, since the counts of labeled trees grow faster than the unlabeled tree counts. Suppose $n = 2^k$ for some integer k, producing a completely balanced tree. At the first division from the root there will be $\frac{1}{2}\binom{n}{n/2}$ possible labelings, at the second split there will be $\frac{1}{2}\binom{n/2}{n/4}$ ways to label each branch (so this number is squared), and so on. This produces,

n	μ	σ^2	σ	σ/μ
2	0.000000	0.000000	0.000000	$----$
3	0.471405	0.000000	0.000000	0.000000
4	0.663325	0.110000	0.331662	0.500000
5	0.920651	0.089544	0.299240	0.325031
6	1.112225	0.125390	0.354104	0.318374
7	1.286728	0.158803	0.398501	0.309701
8	1.447090	0.192351	0.438579	0.303076
9	1.596495	0.225061	0.474405	0.297154
10	1.736145	0.259541	0.509452	0.293439
11	1.867936	0.294694	0.542857	0.290619
12	1.993034	0.330468	0.574864	0.288436
13	2.112342	0.366825	0.605660	0.286725
14	2.226581	0.403721	0.635390	0.285366
15	2.336335	0.441117	0.664166	0.284277
16	2.442086	0.478975	0.692080	0.283397
17	2.544235	0.517262	0.719210	0.282682
18	2.643123	0.555949	0.745620	0.282098
19	2.739038	0.595009	0.771369	0.281620
20	2.832232	0.634418	0.796504	0.281228

Table 10: The means, variances, standard deviations, and relative standard deviations of the distributions of labeled classification trees shown in Tables 4, 6, and 8 that use the path length standard deviation as their measure of imbalance.

after telescoping,

$$c = \prod_{j=1}^{k-1} \left(\frac{1}{2} \binom{n/2^{j-1}}{n/2^j} \right)^{2^{j-1}} = \frac{1}{2^{1+2}} \binom{n}{n/2} \binom{n/2}{n/4}^2 \prod_{j=3}^{k-1} \left(\frac{1}{2} \binom{n/2^{j-1}}{n/2^j} \right)^{2^{j-1}}$$

$$= \frac{1}{2^{4-1}} \frac{n!}{((n/4)!)^4} \prod_{j=3}^{k-1} \left(\frac{1}{2} \binom{n/2^{j-1}}{n/2^j} \right)^{2^{j-1}}$$

$$= \frac{1}{2^{8-1}} \frac{n!}{((n/8)!)^8} \prod_{j=4}^{k-1} \left(\frac{1}{2} \binom{n/2^{j-1}}{n/2^j} \right)^{2^{j-1}} = \ldots = \frac{1}{2^{n/2-1}} \frac{n!}{2^{n/2}} = \frac{n!}{2^{n-1}} \quad (2)$$

as the total count of labelings for this completely balanced tree. When $n \neq 2^k$ for some integer k things will be slightly different though not radically, but that is not the point at the moment. What matters now is to consider a completely unbalanced tree with n leaves. Clearly there will be $n!/2$ ways to label this tree, since each leaf is different in path length from the others except for the longest two which are the same. Comparing this result with the one at (2) we see that the completely unbalanced tree has

$$\frac{n!/2}{n!/2^{n-1}} = 2^{n-2}$$

times more labelings than the completely balanced tree, and that this ratio increases with n. For trees not quite completely unbalanced we will not see as great a number of labelings, but it will still be high. In general, though, we expect to see the number of labelings decrease in rough correlation with the decreasing imbalance among the set of trees.

This has at least two consequences. The first and most obvious is that the number of labels per unlabeled tree is definitely not uniform across the set of unlabeled trees. The second is that, as n increases, the labeled trees at the high end of the balance–imbalance scale should be growing faster than those at the low end; therefore we should see the weight of the distribution move to a higher quantile. Approximately (since not all n have perfectly balanced trees), we should see a doubling of this ratio for every increment of n. But instead Tables 7 and 8 show the mode decreasing; therefore we must conclude that this is caused by a simultaneous shifting of the distribution of the unlabeled trees.

When we look at the algorithm in terms of just the unlabeled trees we see that there is only one completely balanced tree and only one completely

unbalanced one, since the labelings are now ignored. In terms of the combination aspect, the algorithm's essential character is that of mixing and blending, in that these two extreme trees get combined with less extreme ones so that the vast majority of the trees are of this type of blended mixture. Therefore in terms of the unlabeled trees we see that almost all will fall in the middle range of balanced versus unbalanced.

The mean of the path lengths of the perfectly balanced tree when $n = 2^k$ is $\log_2 n$ since they are all the same. For a completely unbalanced tree the mean is $(n(n+1)/2-1)/n \approx n/2$. If we take the geometric mean of these we get $\sqrt{(n \log_2 n)/2} \approx O(\sqrt{n})$. What this means is that, unless n is very small, in the middle ranges of the balance–imbalance scale the average path length will be large enough that we can rearrange the branchings within the tree in an increasing number of ways without changing the average path length. For example, when $n = 8$ the tree with splits

$$\begin{aligned}(8) &\to (2,6) \to ((1,1),(2,4)) \to ((1,1),((1,1),(1,3))) \\ &\to ((1,1),((1,1),(1,(1,2)))) \to ((1,1),((1,1),(1,(1,(1,1)))))\end{aligned}$$

will be represented in our notation as $(c; 2, 2, 3, 3, 3, 4, 5, 5)$, while a different tree split as

$$\begin{aligned}(8) &\to (3,5) \to ((1,2),(1,4)) \to ((1,(1,1)),(1,(1,3))) \\ &\to ((1,(1,1)),(1,(1,(1,2)))) \to ((1,(1,1)),(1,(1,(1,(1,1)))))\end{aligned}$$

has notational form $(c; 2, 3, 3, 2, 3, 4, 5, 5)$. The two trees are clearly different because they differ at their first split yet the set of path lengths to the eight leaves end up the same; therefore they will both have the same mean, variance, and standard deviation. As n becomes larger we see that there will be a bunching up of unlabeled trees in the middle of the balance–imbalance range because that is where the greatest number of blends are possible.

When we then add the labelings back to these trees we see that the sheer number of these blended unlabeled trees in the middle of the range very quickly (in terms of the growth of n) vastly exceeds the number of highly unbalanced labeled trees because, even though the unbalanced trees have much larger numbers of labelings their total counts become overwhelmed due to their tree shape rarity. From this we see that the decreasing value of the mode of the distribution for large n is not due to some phenomenon related to a changing of the distribution but is instead the true distribution shape

reasserting itself after the earlier distortion from the doubling growth rate ratio in the unbalanced tree labelings. Note that this approximate doubling is present for all n but the mixing phenomenon only occurs once the path lengths reach a large enough value for the blending to make a meaningful contribution.

Whether one chooses the variance or the standard deviation as a proxy for the imbalance of the trees is more a matter of individual preference. The square root transformation compresses the upper range of the imbalance values making the highly unbalanced trees look less deviant. For some research purposes this may be more appropriate; for others it may not. But since the two methods differ only in a nonlinear transformation of the independent variable the changes are merely cosmetic. It is true that using the standard deviation shows a more Gaussian-like shape that the Central Limit Theorem predicts we should eventually see, but the main consequence of this is to indicate that we have probably reached the end of any significant changes to the distributions by $n = 20$. No matter which distribution is chosen they both demonstrate the same general tendency of the classification trees to bunch up away from the extreme ends into a small, unimodal band in the middle of the range.

Of course, we could have predicted this from the beginning with only a few minutes of thinking and saved everybody a large amount of trouble! The argument about the bunching up of the unlabeled trees due to the mixing and blending of any rare shape of tree predicts this distribution pattern without doing any calculations at all. It immediately follows that the labeled trees will follow the same pattern because, while the unbalanced trees have many more labelings than the completely balanced ones, neither extreme matters once the middle begins to grow.

Another interesting aspect that appears in Tables 7 and 8 is that the mode seems to be drifting lower. In Table 9 we see statistics for the non-normalized distributions behind Tables 3, 5, and 7 that use the variance as the measure of imbalance. As we expect, the mean increases as n increases; in keeping with the stretching out of the distribution we also see the variance increasing. However, if each data point were being magnified by some constant ρ then the ratio σ/μ should also remain a constant. That we see in the fourth column of Table 9 that this ratio is increasing shows that the peak of the distribution is actually widening.

On the other hand we can look at Table 10 showing the equivalent statistics for the distributions underlying Tables 4, 6, and 8 that use the path

length standard deviation as their measure of imbalance. Again the means and variances are increasing but now σ/μ is decreasing. Which imbalance measure is better still remains a preference of the user but in either case the ratio σ/μ seems to be approaching an asymptotic limit, again supporting the idea that we should not expect many changes for $n > 20$.

But these two new tables still do not explain the drifting lower of the modes of the distributions in Tables 7 and 8. But now we recall that these tables are in quantiles; that is, they have been normalized by dividing by their greatest points, which are represented by the maximally unbalanced trees. Therefore Table 11 shows the variances and standard deviations for these same maximally unbalanced trees. As we should expect, since their path length differences are increasing faster than that of the more blended trees, their imbalance is also growing faster than the average tree as can be seen by comparing the first and fourth columns of Table 11 with the μ values in Tables 9 and 10, respectively. In fact, these comparisons are explicitly calculated in columns two and five of Table 11. And even though the growth rates of all these values are slowing, as suggested above by the growth rate of approximately $O(\sqrt{n})$ and shown in the third and sixth columns of Table 11, the maximal tree at every point still always remains larger than the average tree. As a consequence the maximal quantile slowly drifts lower; therefore the drift in Tables 7 and 8 is seen merely to be a presentation artifact. It remains an open question as to whether this drift will slow to a complete halt or asymptotically continue on toward zero, but it is also clear that if it does so it will only be for values of n far greater than would ever appear in any practical classification situation.

So how might one use this information? We can see that if the distribution of the data among its classes is very balanced or very unbalanced then the number of classification trees that might be expected to fit this data could be relatively small. Thus an algorithm that restricted its searching to such trees might run faster than a general-purpose algorithm. However, it also seems probable that such class distributions are not very likely to occur and that almost all data class distributions will lie somewhere in the middle. This is exactly where most of the trees reside as well, as befits the concept that classification tree-fitting algorithms may produce trees whose balance tends to be correlated with the balance or imbalance among the data class memberships. For these cases, any attempt to restrict the trees under consideration would almost certainly cause more computational difficulty than having no restrictions at all. That is, the probability that any particular tree

	Variance-based			Standard deviation-based		
n	σ^2	μ/σ^2	Change	σ	μ/σ	Change
2	0.000000	−−−−	−−−−	0.000000	−−−−	−−−−
3	0.222222	1.000000	−−−−	0.471405	1.000000	−−−−
4	0.687500	0.800000	0.200000	0.829156	0.800000	0.200000
5	1.360000	0.689076	0.110924	1.166190	0.789452	0.010548
6	2.222222	0.613095	0.075980	1.490712	0.746103	0.043349
7	3.265306	0.555682	0.057414	1.807016	0.712073	0.034030
8	4.484375	0.509864	0.045818	2.117634	0.683352	0.028721
9	5.876543	0.472022	0.037842	2.424158	0.658577	0.024775
10	7.440000	0.440019	0.032003	2.727636	0.636502	0.022075
11	9.173554	0.412477	0.027542	3.028787	0.616727	0.019774
12	11.076389	0.388453	0.024024	3.328121	0.598847	0.017881
13	13.147929	0.367268	0.021185	3.626007	0.582553	0.016293
14	15.387755	0.348419	0.018849	3.922723	0.567611	0.014942
15	17.795556	0.331520	0.016899	4.218478	0.553834	0.013777
16	20.371094	0.316270	0.015250	4.513435	0.541070	0.012763
17	23.114187	0.302429	0.013841	4.807722	0.529198	0.011873
18	26.024691	0.289804	0.012625	5.101440	0.518113	0.011085
19	29.102493	0.278235	0.011568	5.394673	0.507730	0.010383
20	32.347500	0.267593	0.010643	5.687486	0.497976	0.009754

Table 11: The variances, adjusted ratios of the mean over all trees, and ratio changes by n of the maximally unbalanced trees when using the variance as the measure of imbalance, followed by the same for those trees using the standard deviation as the imbalance measure.

being the one that a fitting algorithm might produce would be extremely small, and almost the same as the probability of a huge number of very similar trees. In both cases this reasoning follows from the vast numbers of very similar trees in the middle of the balance–imbalance range.

Acknowledgments

The author would like to thank an unnamed Latin scholar for suggesting the word 'vigesile'.

References

Steven B. Gillispie (2017). The computer program used in preparing this report is available at the Univ. of Washington Library's ResearchWorks archive at http://hdl.handle.net/1773/40333.

Ernst Schröder (1870). Vier combinatorische Probleme, *Zeitschrift für Mathematik und Physik*, **XV**, 5, pp. 361-376.

N. J. A. Sloane, ed. (2017). *The On-Line Encyclopedia of Integer Sequences*, published electronically at https://oeis.org.

Appendix

There are at least three reasons why we might want to reproduce Ernst Schröder's derivation here of the $(2n-3)!!$ count of labeled classification trees with n leaves. First, this is an electronic document so no paper will be wasted; second, Schröder's paper was written in German; and third, it is an elegant calculation worth repeating.

Schröder was actually interested in the number of ways to arrange n items rather than counting classification trees. Obviously, if the items are labeled then there are $n!$ such ways but Schröder went on to impose restrictions. In his third problem he posed the following situation. Suppose, in a collection of n items, we arbitrarily select any two of them and unbreakably pair them together. We then treat the pair as if it were a single item and add it back to the set of unpaired items. This is now the original situation, but with $n-1$ items. The process is then repeated until only one item remains. We see that this will indeed, in reverse, produce any labeled classification tree with n leaves but we see that it also implies these tree objects could be represented via parenthetical objects. For example, the three unlabeled 5-trees could be seen as the three parenthetical groupings $(\bullet,(\bullet,(\bullet,(\bullet,\bullet))))$, $(\bullet,((\bullet,\bullet),(\bullet,\bullet)))$, and $((\bullet,\bullet),(\bullet,(\bullet,\bullet)))$.

The derivation is as follows. Let t_n be the count of labeled classification trees with n leaves. Starting with $n=2$ we know the sequence is $(1,3,15,105,945,\ldots)$. Now we need to convert the algorithm in section 2 into a formula. While we would not normally have a classification tree with a single node we can see that it is theoretically possible, and that there would only be one of them, so that we can set $t_1=1$. This simplifies the first part of the algorithm (where single leaves are added) in that we can say we start by placing $\binom{n}{1}t_1t_{n-1}$ trees into the new t_n set, and then simply add $\binom{n}{k}t_kt_{n-k}$ trees to it for k from 2 to $[n/2]$ except for the complication at the last step when n is even. However, since $\binom{n}{k}=\binom{n}{n-k}$ we see that we can split the first additions in half and reuse them, thus eliminating the special case. That is, we obtain

$$t_n=\frac{1}{2}\sum_{k=1}^{n-1}\binom{n}{k}t_kt_{n-k}.$$

Now we make use of Schröder's first important trick: a change of variables.

Let $t_n = n!\, s_n$. Then we get

$$t_n = n!\, s_n = \frac{1}{2} \sum_{k=1}^{n-1} \frac{n!}{k!\,(n-k)!} k!\, s_k\, (n-k)!\, s_{n-k}$$

which leads to

$$s_n = \frac{1}{2} \sum_{k=1}^{n-1} s_k s_{n-k}. \tag{1}$$

The second important trick is to now use generating functions. A generating function is simply a formal infinite series being used as a "container" to hold a sequence. That is, we define $f(z)$ as

$$f(z) \equiv \sum_{n=0}^{\infty} s_n z^n. \tag{2}$$

But with (2) defined we see that we will also have

$$[f(z)]^2 = s_0^2 + 2s_0 s_1 z + (2s_0 s_2 + s_1^2) z^2 + \sum_{n=3}^{\infty} \sum_{k=0}^{n} s_k s_{n-k} z^n \tag{3}$$

and this is starting to look similar to what we had in (1).

But there are some differences. First, when there are no nodes there are no trees so we need $t_0 = s_0 = 0$. This changes (3) to

$$[f(z)]^2 = \sum_{n=2}^{\infty} \sum_{k=1}^{n-1} s_k s_{n-k} z^n = 2 \sum_{n=2}^{\infty} s_n z^n$$

after making use of (1). But this still isn't quite what we want because it is missing a z^1 term and we know that we must have $s_1 = t_1/1! = 1$. However, this is easily corrected by adding an $s_1 z$ term so we conclude with

$$f(z) = \sum_{n=0}^{\infty} s_n z^n = z + \frac{1}{2}[f(z)]^2. \tag{4}$$

Equation (4) is just the quadratic equation $f^2(z) - 2f(z) + 2z = 0$ with solution

$$f(z) = \frac{2 \pm \sqrt{4 - 8z}}{2} = 1 - (1 - 2z)^{\frac{1}{2}} \tag{5}$$

since (as we will see very soon) the positive square root gives the wrong answer.

The power of using generating functions is based on the ability to recover the coefficients via

$$\left.\frac{d^n f(z)}{dz^n}\right|_{z=0} = \sum_{k=0}^{\infty} s_k \left.\frac{d^n}{dz^n} z^k\right|_{z=0} = n!\, s_n$$

since summands of z^k for $k < n$ are all diffentiated away while those with $k > n$ vanish when $z = 0$. Therefore s_n is just the nth derivative of $f(z)$ evaluated at $z = 0$ and divided by $n!$. But since we are actually interested in $t_n = n!\, s_n$ we obtain

$$t_n = \left.\frac{d^n f(z)}{dz^n}\right|_{z=0}.$$

Setting $z = 0$ in (5) to get the constant zeroth derivative produces $t_0 = 0$ as desired. (The positive square root would have had $t_0 = 2$.) The first derivative of (5) is

$$\frac{df(z)}{dz} = -\frac{1}{2}\,(1-2z)^{-\frac{1}{2}}\,(-2) = (1-2z)^{-\frac{1}{2}}$$

so when $z = 0$ we have $t_1 = 1$, again as desired. Finally we see that, for $n \geq 2$ and the t_n values we are truly interested in,

$$\frac{d^n f(z)}{dz^n} = \left(\prod_{k=2}^{n}(2k-3)\right)(1-2z)^{-\frac{2n-1}{2}}$$

and

$$t_n = (2n-3)!!.$$