Supplementary data

Working principle of Artificial Neural Network (ANN)

In Artificial Neural Network, each neuron works in three steps:
First we will start by making a weighted average of the inputs, then we multiply by their corresponding weights.

$$\sum (a_i \times w_i)$$

Then we will add what we called a bias b. It is a specific value for each neuron.

$$\sum (a_i \times w_i) + b$$

Finally we have a result that we can send to our activation function σ.

$$\sigma \left(\sum (a_i \times w_i) + b \right)$$

in matrix notation:

$$a^{(1)} = \sigma \left(W a^{(0)} + B \right)$$

$a^{(0)}$ represents the first layer, W the weights relating the first layer to the first neurons of the second layer. $a^{(1)}$ the vector result of second neural layer. B is the vector which contains the bias' value for every neuron. We apply this formula to all others layers (Figure 4a).

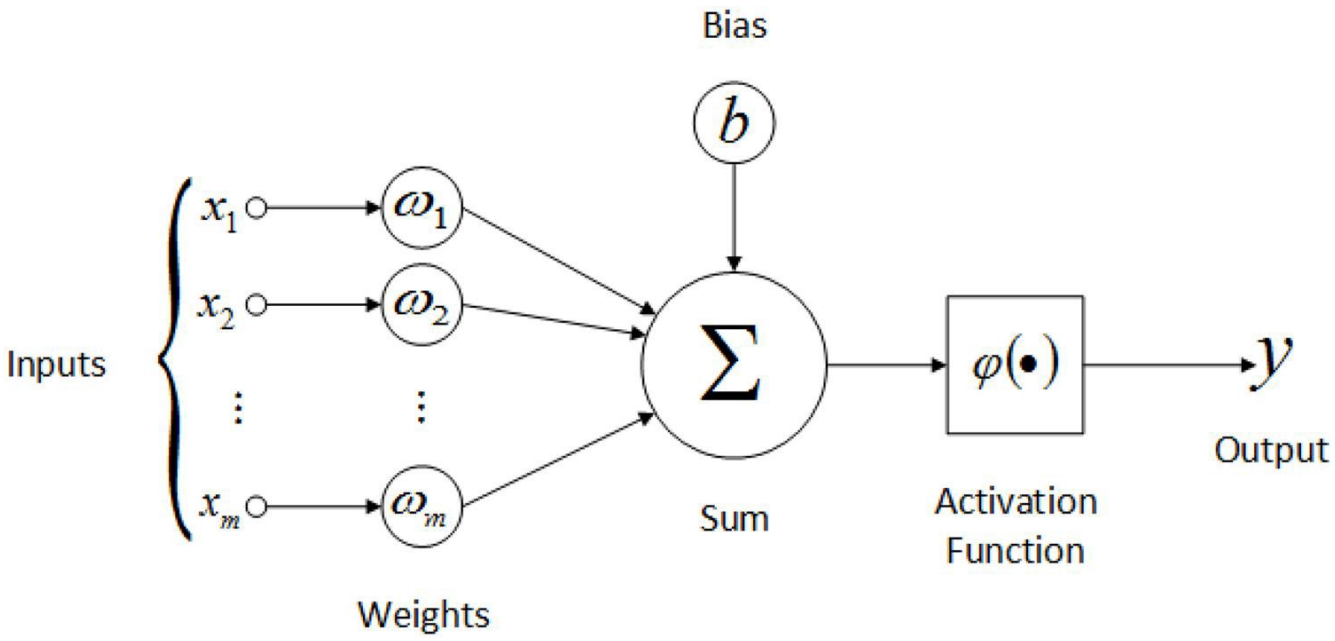

Figure 4a. Artificial Neural Network (ANN)

The bigger the given value in input will be, the more the output value will be close to 1. On the other hand, the more the input value will be negative, the more the exit value will approach 0.It is this result which tells us how our neural network is activated. At 0, we will say the neurons is inactive, at 1 it will be fully activated and can have all of the intermediate states. In order to increase control on this exit value we bring the bias b. Bias is the threshold where we will consider our neuron activated. For example, if our sigmoid function returns a 50% activated neuron when it receives 0 in input, if we add a bias to our equation, the neuron will be activated more easily. This value allows us to shift the activation function. We will add the bias b to input value of our function. This value will be adjusted at the same time as all of our weights during the network training phase.

After this calculation, we end up with our exit value. This process where our value are propagated until bringing a result is called feedforward. Information of our initial variant will propagate through hidden layers based on weight values.

We finally have a result in the output layer. According to our model, we have 2 output neurons. Each of them are the representation of a number between 0 to 1 (NO or YES).The degree of activation of each of these neurons represents the percentage chance that the variant passed as input is pathogenic -> 1 (YES) or 0 (NO) (according to our training dataset).

For example, variant chr12:25398280/ KRAS / 24.41% / 1962 / c.35G>T / p.Gly12Val, the best possible answer would be to have all of neurons = NO (=0) shut off except the one representing the YES (=1). At the beginning, our network is trying to find the best solution

for the first time (first epoch:is when an entire dataset is passed forward and backward through the neural network only once) all of its output neurons will be more or less activated, it predicted randomly because he had no information about the result we were expected. To improve the result, we have to correct the parameters of our neural network.
The only way to change the behaviour of our network is to vary its weights. The learning process consist in finding the best possible weights to have a correct answer. For this, we compare the given result and the expected result. We will end up with a value called loss. The higher the loss, the more our network is far from the correct answer. Thanks to the loss, we are able to deduct which weights have participated the most on our incorrect answer. Then we have to adjust the weights in order to minimize the loss and get closer to the correct answer.
On the next epoch, we retry with another variant, the network will be wrong again but it is closer to the expected answer.
Loss value is a representation of our error rate on this network for each of its output neurons. Then amongst our parameters, we will have to identify which combination of each of these weights will allow us to obtain the smallest possible loss. To test all these values, we will use the algorithm of gradient descent (optimizer). Gradient descent uses derivatives of a function. As we know the derivative function allows to know the tangent's gradient of the curve for a given point. According to that tangent, we will be able to determine how to change the entry weight to get closer to the result that we are looking for. If the derivative is positive, we will have to lower down our weight and vice versa. We can also deduce that the more important our derivative is, the more we will be far from the right result, we will have to increase our input, recalculate the derivative and restart those steps again and again until we reach a nil derivative which means we reached the minimum loss, consequently the best possible result. Gradient descent has a parameter called learning rate.
Initially the steps are bigger that means the learning rate is higher and as the point goes down the learning rate becomes smaller by the shorter size of steps. The backpropagation system corresponds to the action of correcting the weights that led to the error, doing it layer by layer using the gradient descent.
We only have to duplicate these fitforward and back propagation steps again and again until we get the optimal result.
We repeat these steps thousands of times with lot of somatic variants and we adjust the weights each times. After a lot of epochs the network learned how to recognize onco-somatic variants.
Progressively, the network will be able to have its own representation of each variant entered in the inputs. After the training process, we can test the network on mutation variants never seen before. If the learning is correctly done, the network gives us our expected answer. All these calculation processes are automated thanks to NeuralNet package.

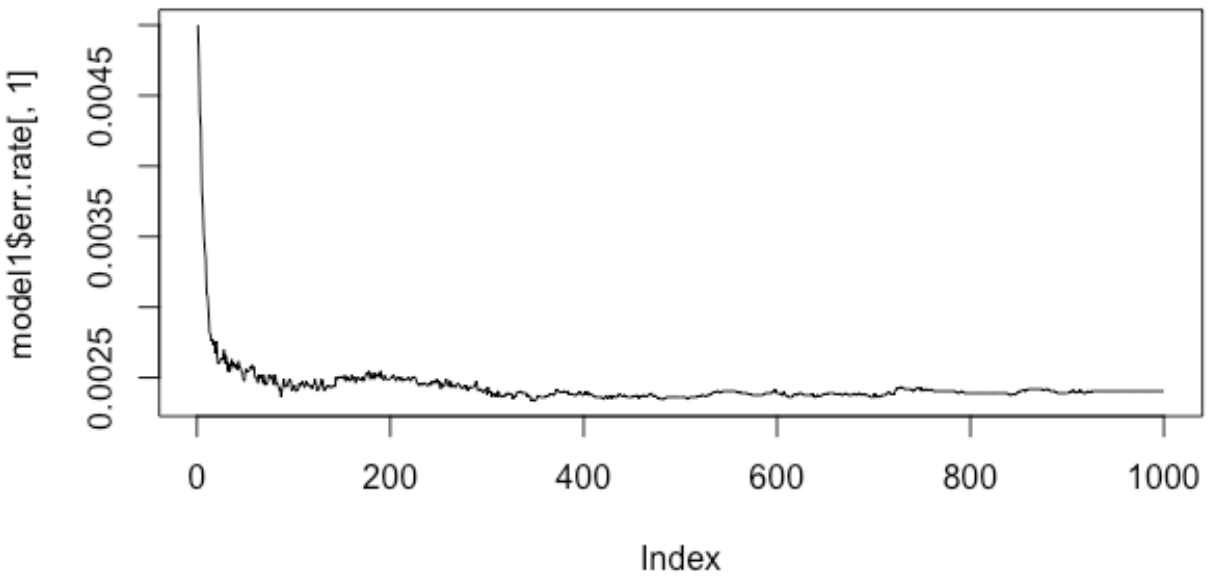

Figure 1a. The plot shows that our best error rate was stabilized for ntree = 500. All values before 500 oscillate too much and after it's more machine time consuming

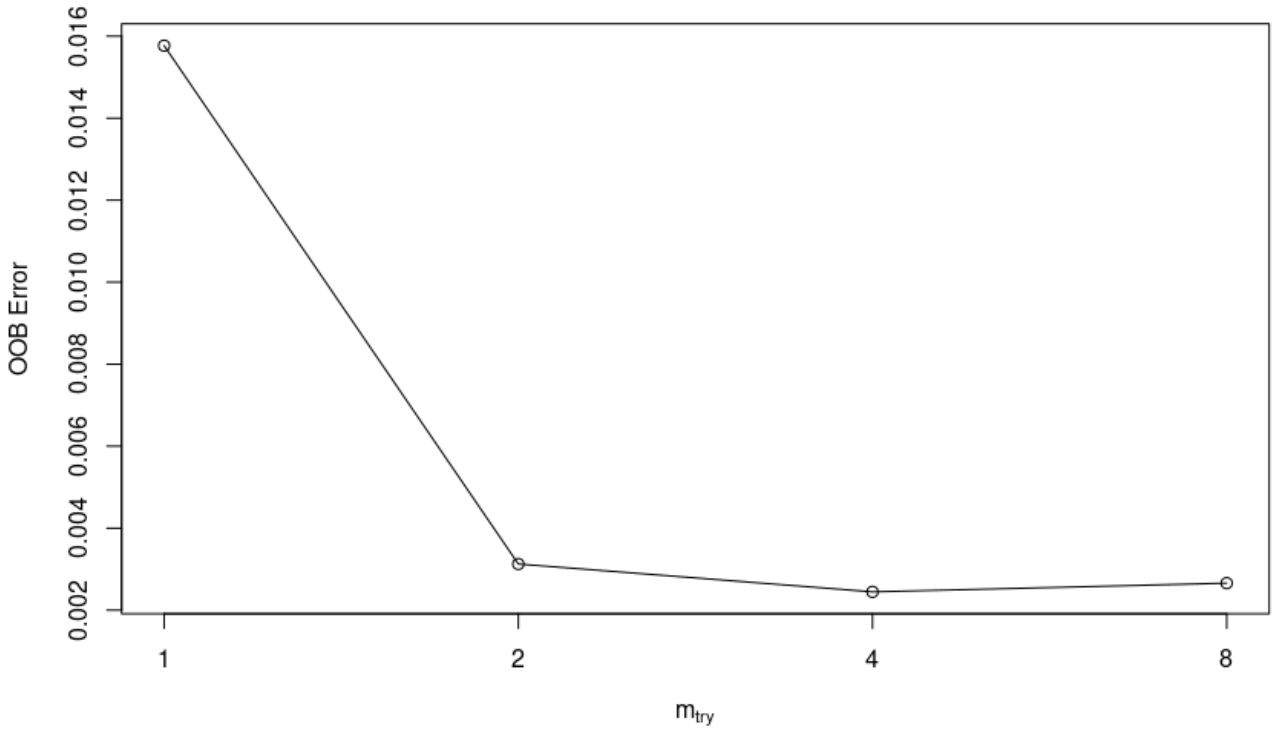

Figure 2a. Our best OOB was found with mtry = 4

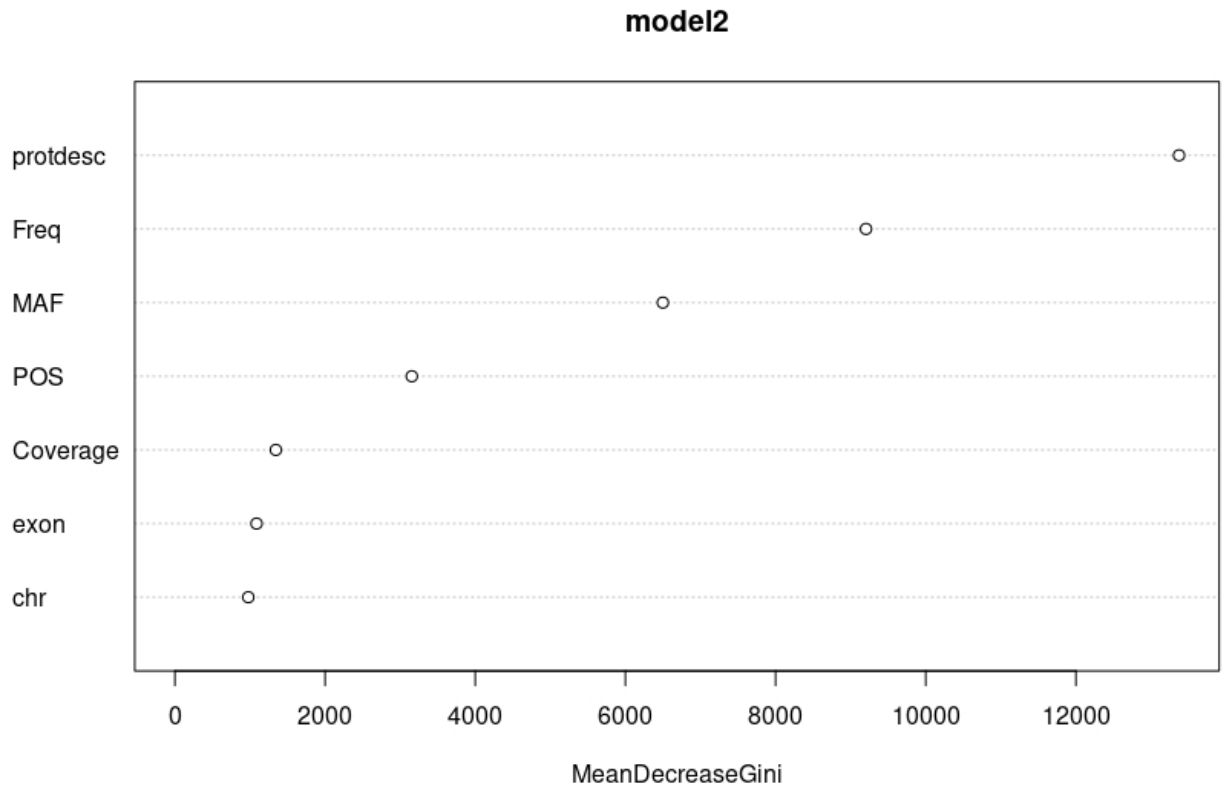

Figure 3a. Features importance

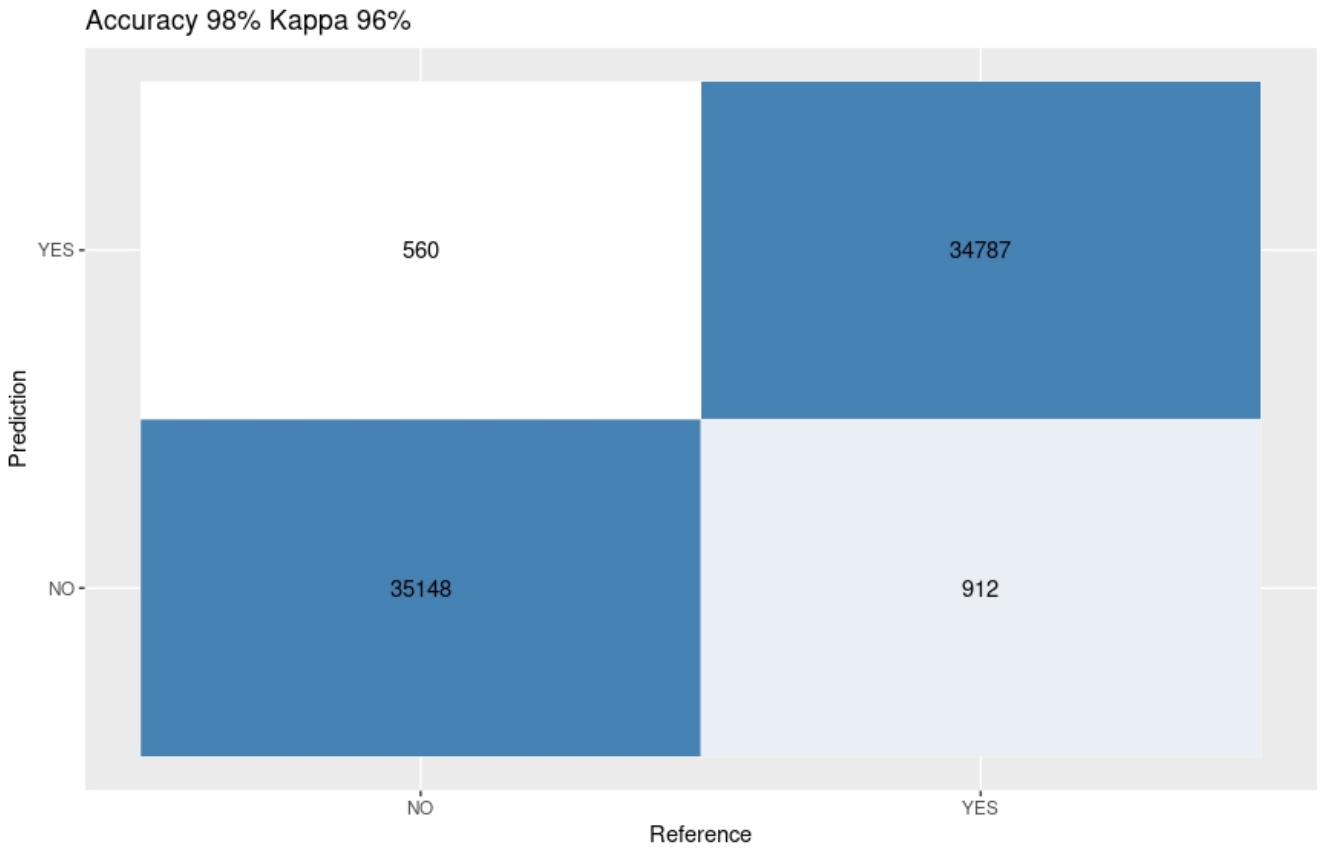

Figure 5a. Confusion Matrix on Test Set with ANN (L1:5 / L2: 0 / L3: 5)

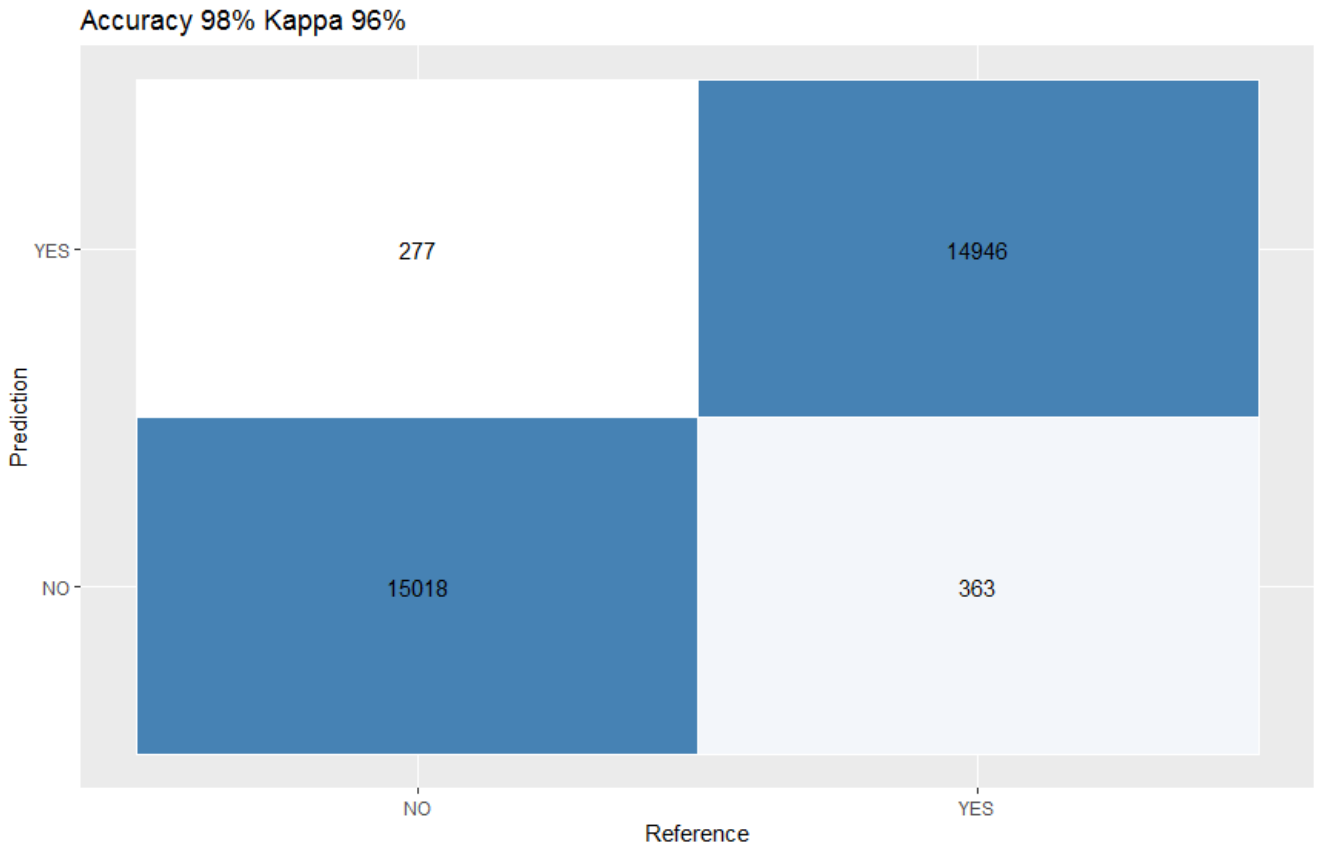

Figure 6a. Confusion Matrix on Validation Set with ANN (L1:5 / L2: 0 / L3:5)

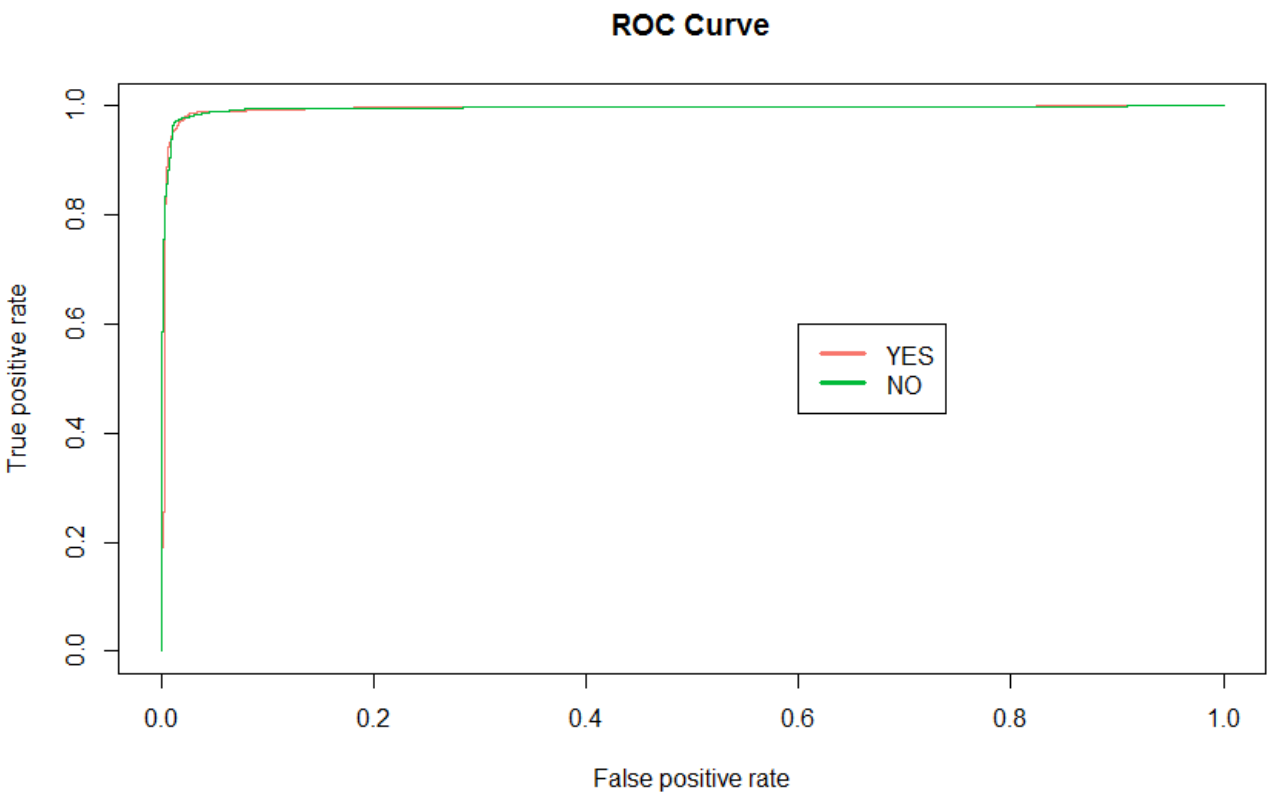

Figure 7a. ROC Curve for our ANN

id	isMut	VarClass	MLRF	Location	Exon	Coding	ProtDesc	MAF	Coverage	%Frequency	TP53-IARC	ClinVar	Locus	PHRED_Q
1	☐		NO	DDR2:exonic:NM_006182.2	13	c.1509C>T	p.(=)	0.0	2000	51.30			chr1:162741818	41
2	☐		YES	DDR2:exonic:NM_006182.2	14	c.1750G>A	p.Gly584Arg		2000	24.30			chr1:162743280	36
4	☐		NO	ERBB4:intronic:NM_005235.2	0	c.421+58A>G	p.?	0.355	2000	58.15			chr2:212812097	41
5	☐		YES	PIK3CA:exonic:NM_006218.3	10	c.1636C>A	p.Gln546Lys		1979	27.39		Likely pathogenic,Pathogenic	chr3:178936094	38
6	☐		NO	FGFR3:exonic:NM_000142.4	14	c.1953G>A	p.(=)	0.044 (ref)	879	99.66			chr4:1807894	42
7	☐		NO	PDGFRA:exonic:NM_006206.4	12	c.1701A>G	p.(=)	0.042 (ref)	2000	99.60		Benign	chr4:55141055	45
8	☐		NO	EGFR:intronic:NM_005228.3	0	c.1498+22A>T	p.?	0.227 (ref)	1999	10.91			chr7:55228053	29
9	☐		NO	EGFR:intronic:NM_005228.3	0	c.1498+29G>A	p.?		2000	13.35			chr7:55228060	31
10	☐		NO	EGFR-AS1:exonic_nc:NR_047551.	20	c.2361G>A	p.(=)	0.433	2000	2.20		Benign,Likely benign	chr7:55249063	8
11	☐		NO	MET:exonic:NM_001127500.2	2	c.534C>T	p.(=)	0.088	1987	38.10		Benign	chr7:116339672	39
12	☐		NO	MET:exonic:NM_001127500.2	20	c.3912C>T	p.(=)	0.352	1989	62.95		Benign	chr7:116435768	42
13	☐		NO	RET:exonic:NM_020975.4	11	c.2071G>A	p.Gly691Ser	0.169	872	46.56		Benign,Likely benign	chr10:43610119	36
14	☐		NO	RET:intronic:NM_020975.4	0	c.2136+49C>T	p.?		843	26.33			chr10:43610233	32
15	☐		NO	RET:exonic:NM_020975.4	13	c.2307G>T	p.(=)	0.287 (ref)	1998	48.30		Benign	chr10:43613843	40
16	☐		NO	RET:intronic:NM_020975.4	0	c.2608-24G>A	p.?	0.173	1167	4.11		Benign	chr10:43615505	18
17	☐		NO	RET:exonic:NM_020975.4	15	c.2712C>G	p.(=)	0.172	1166	42.88		Benign,Likely benign	chr10:43615633	37
18	☐		YES	RET:exonic:NM_020975.4	16	c.2767C>T	p.Leu923Phe		1959	24.60			chr10:43617430	36
20	☐		YES	TP53:exonic:NM_000546.5	8	c.848G>C	p.Arg283Pro		1918	27.22	deleterious		chr17:7577070	38
21	☐		YES	TP53:exonic:NM_000546.5	7	c.741_742delCCinsTT	p.Arg248Trp		1942	31.67			chr17:7577528	38
22	☐		NO	TP53:exonic:NM_000546.5	7	c.741C>T	p.(=)		2000	31.10	NA	Likely benign	chr17:7577540	39

Figure a. Example of NGS analysis result with Ion Interface and our MLRF (Machine Learning Random Forest) result column

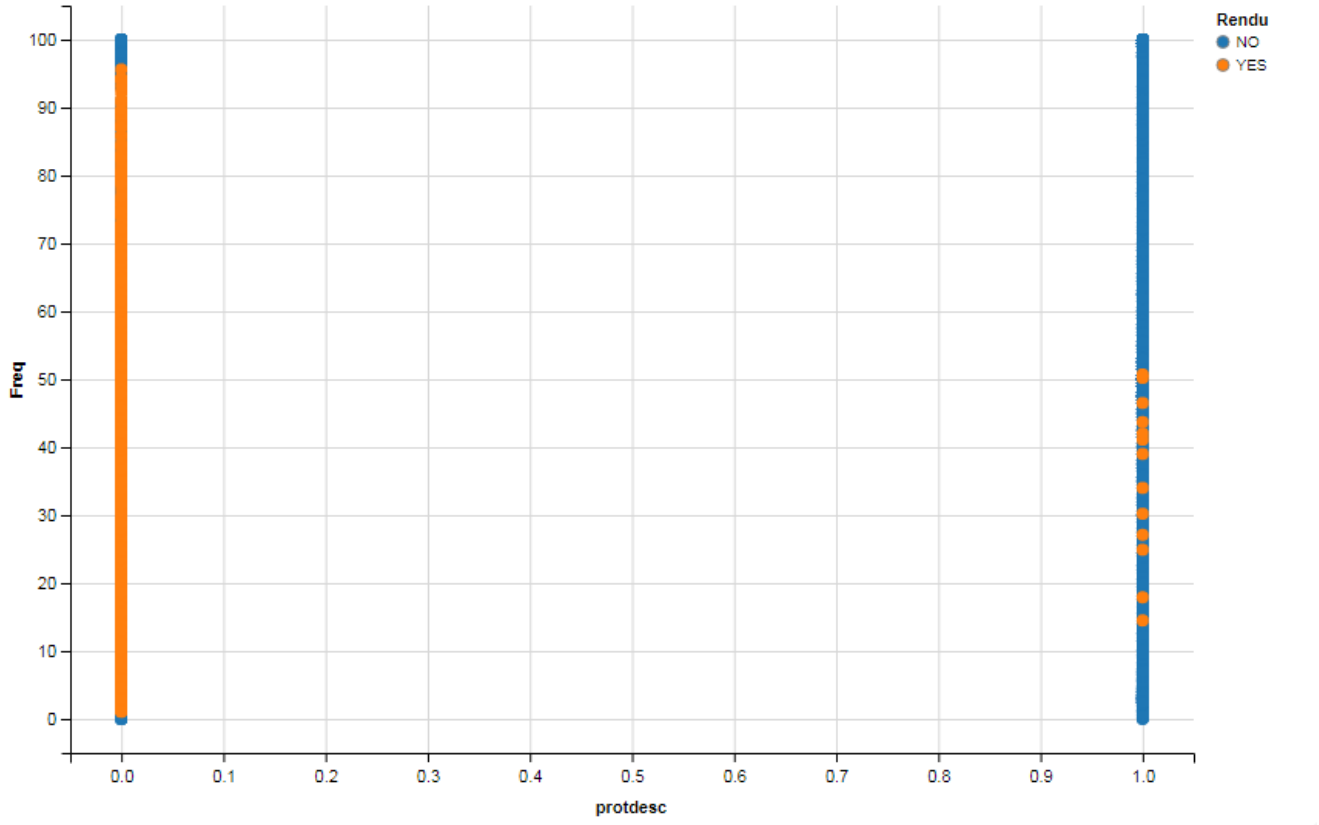

Figure b. Variant Allele Frequency (Freq) in function of Acid Amino Change (protdesc). A strong proportion of variants were flagged as pathogenic (labelled as YES) by the Biologist when the Acid Amino Change is different from silent (protdesc $\neq 1$)

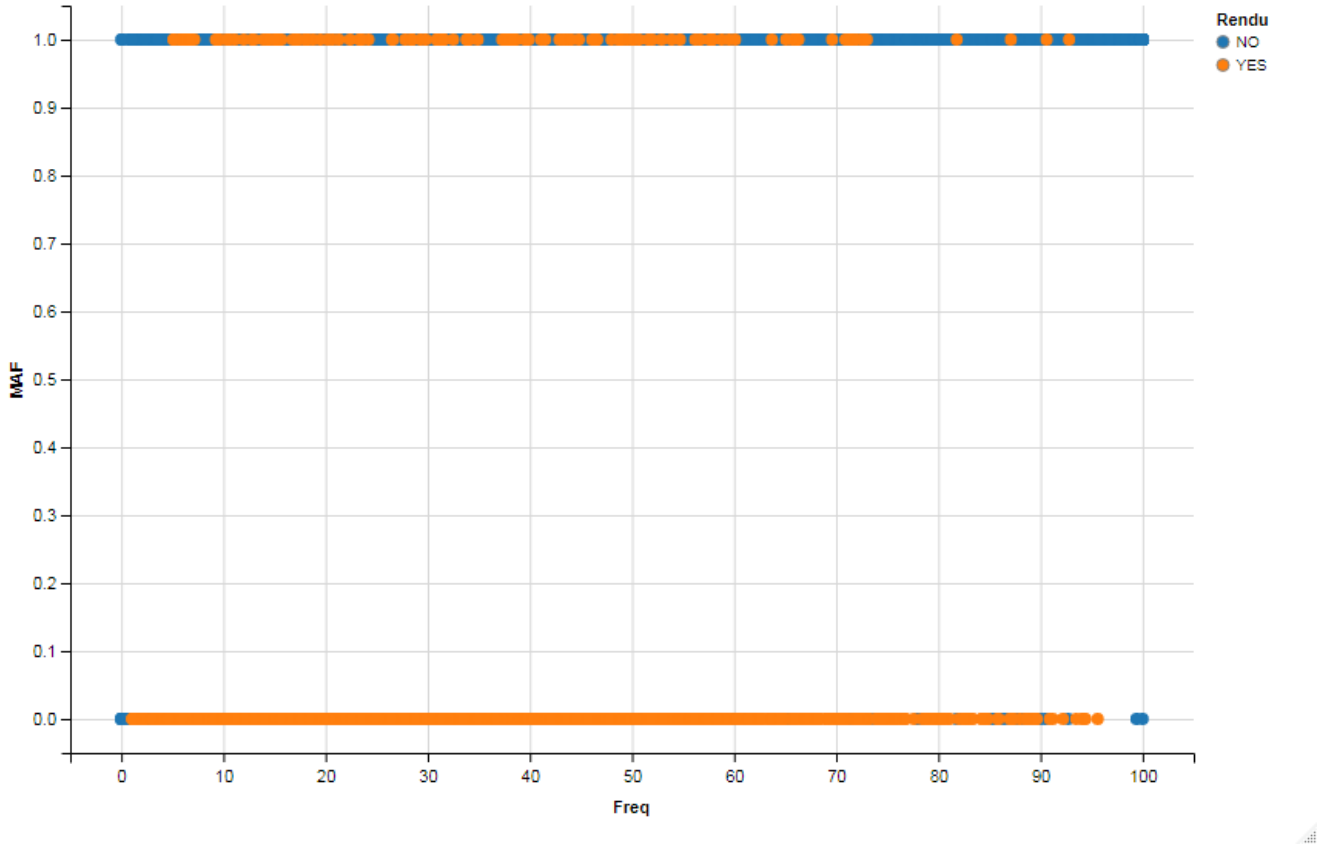

Figure c. Minor Allele Frequency (MAF) in function of Variant Allele Frequency (Freq), we can observe more variants were flagged as benign when a MAF is present on a range of Frequency [0;100%]

Genes	Coding	Locus	Type	Exon	VAF	MAF	Coverage	Protdesc	Biologist Decision	MLRF	MLRF proba	ANN	ANN Prob
KRAS	c.34_35delGGinsTT	chr12:25398280	MNV	2	43.19	0	3941	0	YES	YES	1.000	YES	0.998280525
KRAS	c.34G>T	chr12:25398280	SNV	2	25.47	0	3957	0	YES	YES	1.000	YES	0.998280525
ERBB4	c.955G>T	chr2:212578302	SNV	8	12.98	0	4000	0	YES	YES	1.000	YES	0.996720195
TP53	c.592G>T	chr17:7578256	SNV	6	50.43	0	3881	0	YES	YES	1.000	YES	0.998146176
KRAS	c.35G>C	chr12:25398280	SNV	2	10.19	0	3965	0	YES	YES	1.000	YES	0.998280466
EGFR	c.2240_2254delTAAGAGAAGCAACAT	chr7:55242467	INDEL	19	7.58	0	3877	0	YES	YES	1.000	YES	0.964137554
TP53	c.701A>G	chr17:7577578	SNV	7	56.12	0	3981	0	YES	YES	1.000	YES	0.997943997
SMAD4	c.1587dup	chr18:48604764	INDEL	12	30.09	0	2888	0	YES	YES	1.000	YES	0.984070003
NRAS	c.182A>G	chr1:115256528	SNV	3	48.26	0	3997	0	YES	YES	1.000	YES	0.996720195
POLE	c.1348G>A	chr12:133250172	SNV	13	53.33	0	1999	0	YES	YES	1.000	YES	0.994729102
EGFR	c.2308_2309insCCAGCGTGG	chr7:55248998	INDEL	20	64.28	0	1982	0	YES	YES	1.000	NO	0.815051854
TP53	c.749C>T	chr17:7577528	SNV	7	49.13	0	1964	0	YES	YES	1.000	YES	0.983990073
MET	c.2942-19TCTTTCTCTCTGTTTTAAGA>T	chr7:116411884	INDEL	14	6.14	0	1943	1	YES	NO	0.557	YES	0.995143354
TERT	c.1-124C>T	chr5:1295228	SNV	0	49.90	0	1950	1	YES	YES	1.000	YES	0.998228788
FBXW7	c.1513C>T	chr4:153247289	SNV	10	17.08	0	1997	0	YES	YES	0.998	YES	0.996720195
KRAS	c.436G>A	chr12:25378562	SNV	4	27.70	0	2000	0	YES	YES	1.000	YES	0.998292804
NRAS	c.182A>T	chr1:115256528	SNV	3	69.93	0	1992	0	YES	YES	1.000	YES	0.996720195
ERBB4	c.421+58A>G	chr2:212812097	SNV	0	40.56	1	3994	1	NO	NO	1.000	NO	0.575408518
FGFR3	c.1953G>A	chr4:1807894	SNV	14	99.51	1	2060	1	NO	NO	1.000	NO	0.817090154
MET	c.1124A>G	chr7:116340262	SNV	2	38.65	1	3997	0	NO	NO	1.000	NO	0.575388730
HRAS	c.81T>C	chr11:534242	SNV	2	39.30	1	257	1	NO	NO	1.000	NO	0.996798575
EGFR	c.1498+22A>T	chr7:55228053	SNV	0	99.95	1	3989	1	NO	NO	0.982	NO	0.993650138
TP53	c.215C>G	chr17:7579472	SNV	4	49.46	1	3999	0	NO	NO	1.000	YES	0.983942688
MET	c.3313G>T	chr7:116415165	SNV	15	48.68	0	3991	0	YES	YES	1.000	YES	0.996720195

Table a. Random Forest (MLRF) decision with probability versus Artificial Neural Network (ANN) decision on NGS variants. Green color means in accordance with Biologist decision and in red not in accordance.

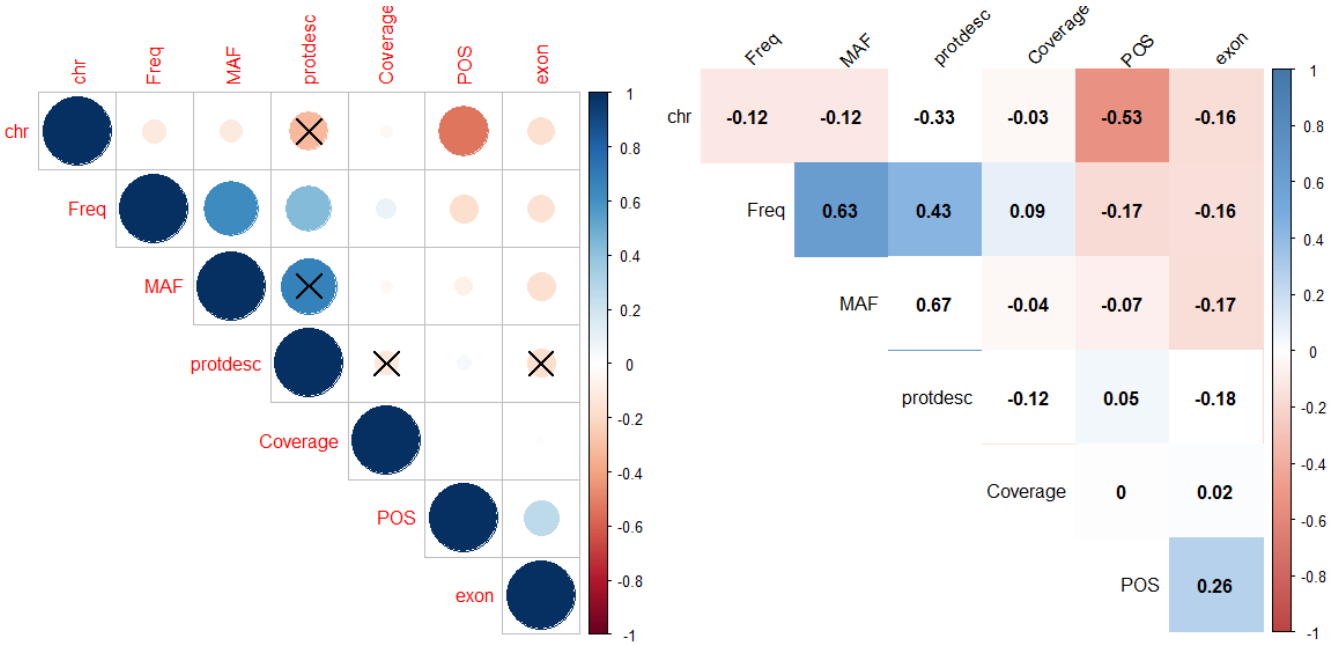

Figure d. Correlation Features. **Correlation** with **p-value** > 0,01 are considered as insignificant. In this case the **correlation coefficient values** are leaved blank or crosses are added.

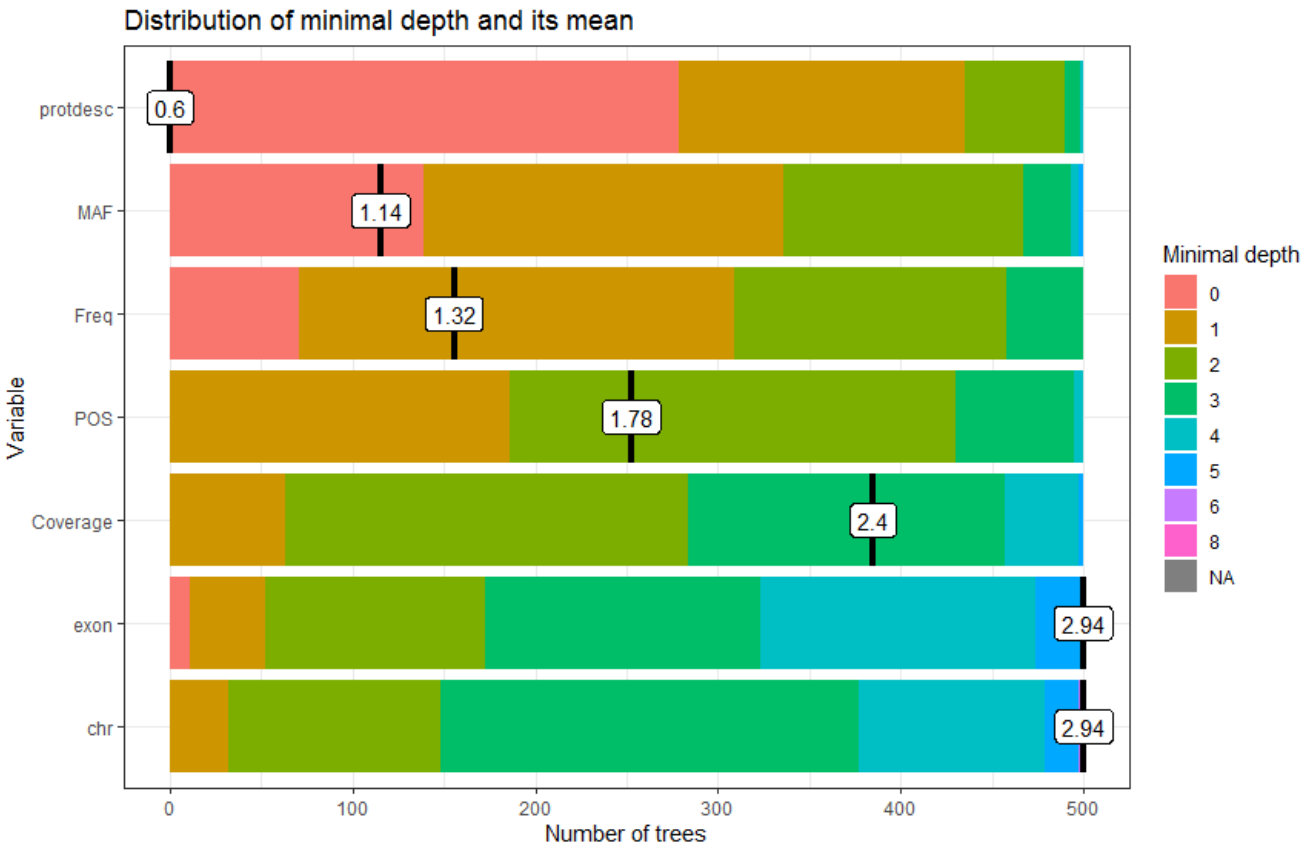

Figure e. The smaller the mean minimal depth, the more important the variable is and the higher up the y-axis the variable will be. The rainbow gradient reveals the min and max minimal depth for each variable. The bigger the proportion of minimal depth zero (red blocks), the more frequent the variable is used for splitting trees. The range of the x-axis is from zero to the maximum number of trees for the feature.