Evaluating random forests for estimation of parameter importance and interaction

1. Configuration ranking as dependent variable

In this example we use the ranking as the dependent variable when training.

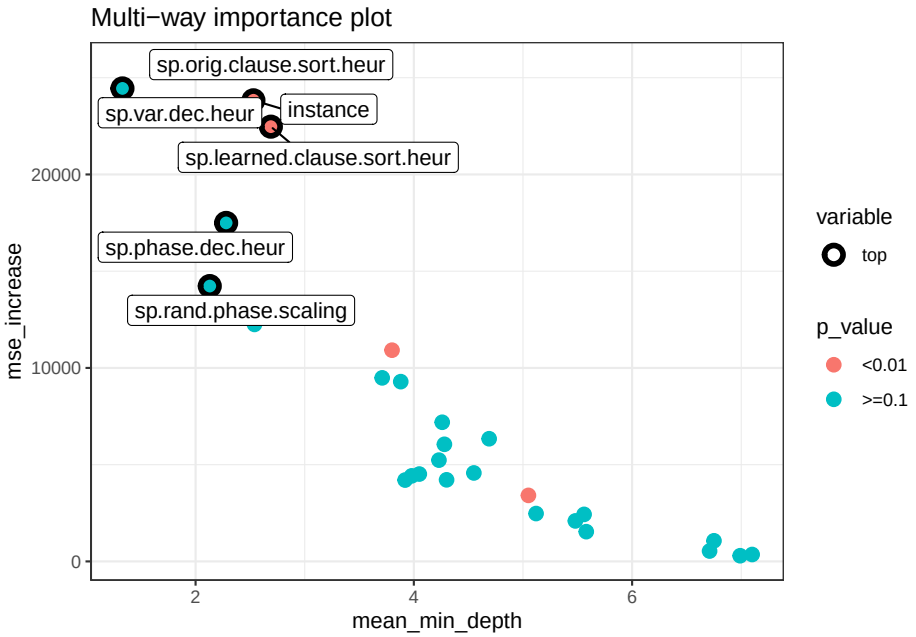
```
load("model_data/model-spear-cat-ranking.Rdata")
importance_frame = model$importance_frame
important_parameters = model$important_parameters
kable(importance_frame[order(importance_frame[,"mean_min_depth"]),])  %>%  kable_styling(latex_options=
```

	variable	mean_min_depth	no_of_nodes	mse_increase	node_purity_increase	no_of_trees	times_a_root	p_value
27	sp.var.dec.heur	1.33	9106	24446.9698	121600402	100	26	1.0000000
13	sp.orig.clause.sort.heur	1.88	20230	25566.5698	124046970	100	10	0.0000000
16	sp.rand.phase.scaling	2.13	9409	14231.3811	51548281	100	27	1.0000000
14	sp.phase.dec.heur	2.28	8900	17495.6841	51337713	100	19	1.0000000
2	instance	2.53	108775	23832.6824	398603132	100	0	0.0000000
21	sp.res.order.heur	2.54	8839	12248.5564	62438101	100	1	1.0000000
8	sp.learned.clause.sort.heur	2.69	21136	22470.0548	124228616	100	2	0.0000000
17	sp.rand.var.dec.freq	3.71	8305	9486.7698	37215467	100	1	1.0000000
7	sp.first.restart	3.80	13525	10916.7249	52661142	100	0	0.0000000
18	sp.rand.var.dec.scaling	3.88	9857	9291.0568	38044288	100	1	1.0000000
12	sp.max.res.runs	3.92	5624	4204.1246	20405182	100	1	1.0000000
11	sp.max.res.lit.inc	3.98	4950	4420.9002	18318719	100	2	1.0000000
19	sp.res.cutoff.cls	4.05	4986	4515.6858	17774115	100	2	1.0000000
23	sp.restart.inc	4.23	9594	5235.0097	27340577	100	0	1.0000000
9	sp.learned.clauses.inc	4.26	9792	7192.0556	31240203	100	1	1.0000000
10	sp.learned.size.factor	4.28	9977	6052.5487	29597571	100	1	1.0000000
15	sp.rand.phase.dec.freq	4.30	11633	4216.2544	31114884	100	0	0.9910404
5	sp.clause.del.heur	4.55	4852	4571.8672	15828230	100	4	1.0000000
26	sp.var.activity.inc	4.69	4909	6340.2837	16771791	100	2	1.0000000
1	dummy	5.05	16714	3411.9532	28154146	100	0	0.0000000
20	sp.res.cutoff.lits	5.12	5321	2473.7645	13184566	100	0	1.0000000
4	sp.clause.decay	5.48	5647	2092.2866	13619317	100	0	1.0000000
28	sp.variable.decay	5.56	6200	2429.7683	13261453	100	0	1.0000000
3	sp.clause.activity.inc	5.58	4795	1537.5368	10962504	100	0	1.0000000
22	sp.resolution	6.71	2797	536.8824	3855853	100	0	1.0000000
25	sp.use.pure.literal.rule	6.75	3437	1065.5794	5808692	100	0	1.0000000
6	sp.clause.inversion	6.99	1547	293.6258	3501961	100	0	1.0000000
24	sp.update.dec.queue	7.10	1929	357.0880	3368373	100	0	1.0000000

We can plot some measures using the randomForestExplainer package. In this case, since we are not predicting performance analizing the importance of parameters not including the instance would be an error.

```
#no_of_nodes
plot_multi_way_importance(importance_frame, size_measure = "p_value", y_measure="mse_increase", x_measu
```

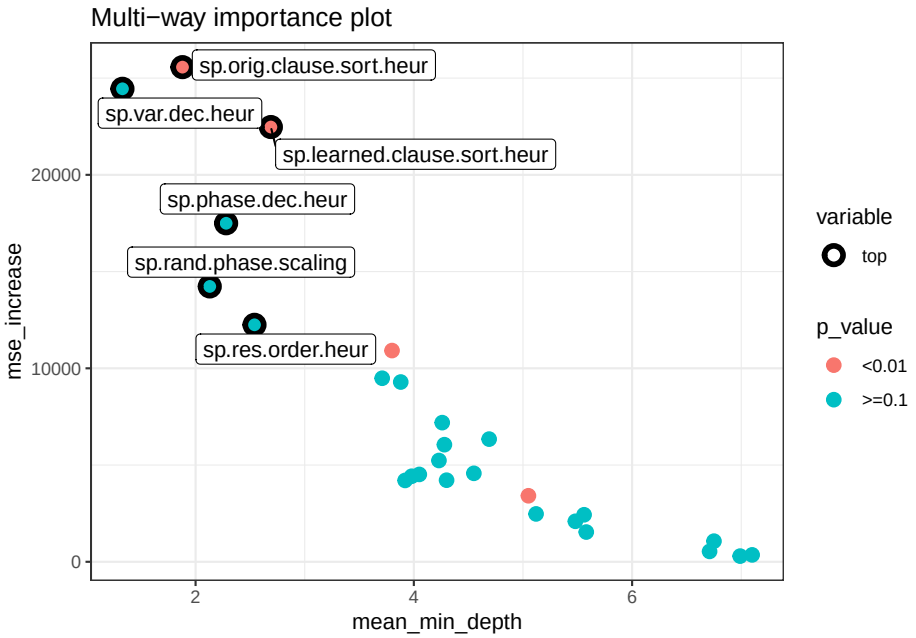
```
## Warning: Using alpha for a discrete variable is not advised.
```



Removing the instance:

```
plot_multi_way_importance(importance_frame[importance_frame[,"variable"]!="instance",], size_measure = "
```

```
## Warning: Using alpha for a discrete variable is not advised.
```



We can also visualize the relationship between importance measures ***TODO: check what this actually means and how can be used***:

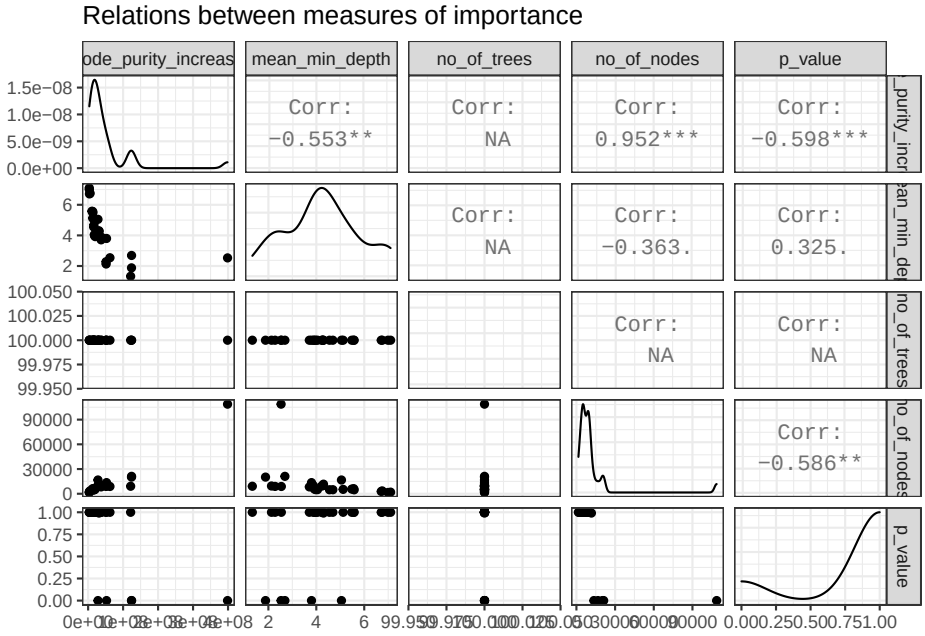
```
plot_importance_ggpairs(importance_frame)
```

```
## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero
```

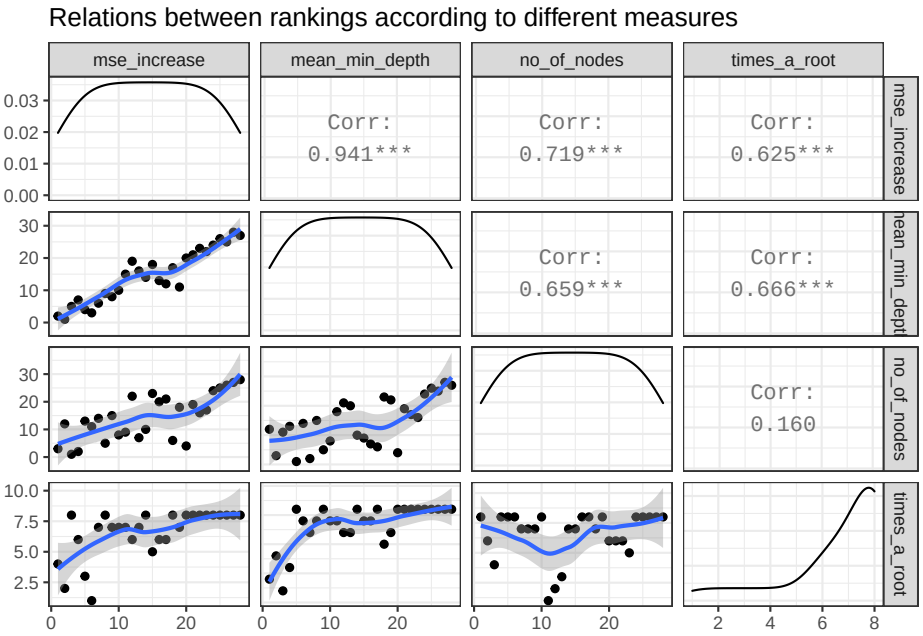
```
## Warning in cor(x, y): the standard deviation is zero

## Warning in cor(x, y): the standard deviation is zero
```




```
plot_importance_rankings(importance_frame, measures=c("mse_increase", "mean_min_depth", "no_of_nodes",
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



After the analysis of conditional parameters the most important 5 parameters are (plus dummy added for interaction analysis):

```
print(important_parameters)
```

```
## [1] "sp.var.dec.heur"          "sp.orig.clause.sort.heur"
## [3] "sp.rand.phase.scaling"    "sp.phase.dec.heur"
## [5] "instance"                 "dummy"
```

None that nnls, which was detected initially as a very important parameter has been removed since its importance seemed to be related to its conditionallity to localsearch. This was confirmed by a the re training process described above.

We run the interaction analysis and find the importance of interactions:

```
kable(model$full_interactions_frame)  %>%  kable_styling(latex_options="scale_down")
```

variable	root_variable	mean_min_depth	occurrences	interaction	uncond_mean_min_depth
dummy	sp.orig.clause.sort.heur	1.6000	100	sp.orig.clause.sort.heur:dummy	2.88
dummy	sp.var.dec.heur	1.9600	100	sp.var.dec.heur:dummy	2.88
instance	instance	1.2100	100	instance:instance	1.99
instance	sp.orig.clause.sort.heur	0.8900	100	sp.orig.clause.sort.heur:instance	1.99
instance	sp.rand.phase.scaling	0.9600	100	sp.rand.phase.scaling:instance	1.99
instance	sp.var.dec.heur	0.6800	100	sp.var.dec.heur:instance	1.99
sp.orig.clause.sort.heur	sp.orig.clause.sort.heur	1.2500	100	sp.orig.clause.sort.heur:sp.orig.clause.sort.heur	1.47
sp.orig.clause.sort.heur	sp.phase.dec.heur	0.8600	100	sp.phase.dec.heur:sp.orig.clause.sort.heur	1.47
instance	sp.phase.dec.heur	1.1076	99	sp.phase.dec.heur:instance	1.99
sp.orig.clause.sort.heur	instance	1.0212	99	instance:sp.orig.clause.sort.heur	1.47
sp.orig.clause.sort.heur	sp.rand.phase.scaling	0.9812	99	sp.rand.phase.scaling:sp.orig.clause.sort.heur	1.47
sp.orig.clause.sort.heur	sp.var.dec.heur	0.9779	99	sp.var.dec.heur:sp.orig.clause.sort.heur	1.47
sp.phase.dec.heur	sp.phase.dec.heur	2.2476	99	sp.phase.dec.heur:sp.phase.dec.heur	1.55
sp.rand.phase.scaling	sp.rand.phase.scaling	2.0912	99	sp.rand.phase.scaling:sp.rand.phase.scaling	1.32
sp.var.dec.heur	instance	0.9612	99	instance:sp.var.dec.heur	1.15
sp.var.dec.heur	sp.orig.clause.sort.heur	0.8800	99	sp.orig.clause.sort.heur:sp.var.dec.heur	1.15
sp.var.dec.heur	sp.phase.dec.heur	0.8976	99	sp.phase.dec.heur:sp.var.dec.heur	1.15
sp.var.dec.heur	sp.rand.phase.scaling	0.9912	99	sp.rand.phase.scaling:sp.var.dec.heur	1.15
sp.var.dec.heur	sp.var.dec.heur	1.3179	99	sp.var.dec.heur:sp.var.dec.heur	1.15
dummy	sp.phase.dec.heur	1.9152	98	sp.phase.dec.heur:dummy	2.88
dummy	sp.rand.phase.scaling	2.2024	98	sp.rand.phase.scaling:dummy	2.88
dummy	instance	1.8636	97	instance:dummy	2.88
instance	dummy	1.4156	96	dummy:instance	1.99
sp.rand.phase.scaling	sp.var.dec.heur	2.0016	96	sp.var.dec.heur:sp.rand.phase.scaling	1.32
sp.orig.clause.sort.heur	dummy	1.5920	95	dummy:sp.orig.clause.sort.heur	1.47
sp.rand.phase.scaling	sp.phase.dec.heur	1.9680	95	sp.phase.dec.heur:sp.rand.phase.scaling	1.32
sp.rand.phase.scaling	sp.orig.clause.sort.heur	2.1600	94	sp.orig.clause.sort.heur:sp.rand.phase.scaling	1.32
dummy	dummy	2.3848	93	dummy:dummy	2.88
sp.phase.dec.heur	sp.rand.phase.scaling	2.5996	92	sp.rand.phase.scaling:sp.phase.dec.heur	1.55
sp.rand.phase.scaling	instance	2.8420	90	instance:sp.rand.phase.scaling	1.32
sp.var.dec.heur	dummy	2.4504	89	dummy:sp.var.dec.heur	1.15
sp.phase.dec.heur	sp.var.dec.heur	4.0006	86	sp.var.dec.heur:sp.phase.dec.heur	1.55
sp.phase.dec.heur	sp.orig.clause.sort.heur	4.0500	85	sp.orig.clause.sort.heur:sp.phase.dec.heur	1.55
sp.phase.dec.heur	instance	4.5276	77	instance:sp.phase.dec.heur	1.55
sp.rand.phase.scaling	dummy	4.5664	74	dummy:sp.rand.phase.scaling	1.32
sp.phase.dec.heur	dummy	5.1692	72	dummy:sp.phase.dec.heur	1.55

Once we filter dummy interactions and aggregate bidirectional interactions we get a matrix where the importance of interactions are summarized:

```
print(model$interactions_frame)
```

```
## [1] variable              root_variable         mean_min_depth
## [4] occurrences           interaction           uncond_mean_min_depth
## <0 rows> (or 0-length row.names)
```

2. Configuration ranking quartile as dependent variable

In this example we use the ranking as the dependent variable when training.

```
load("model_data/model-spear-cat-qranking.Rdata")
importance_frame = model$importance_frame
important_parameters = model$important_parameters
kable(importance_frame[order(importance_frame[,"mean_min_depth"]),])  %>%  kable_styling(latex_options=
```

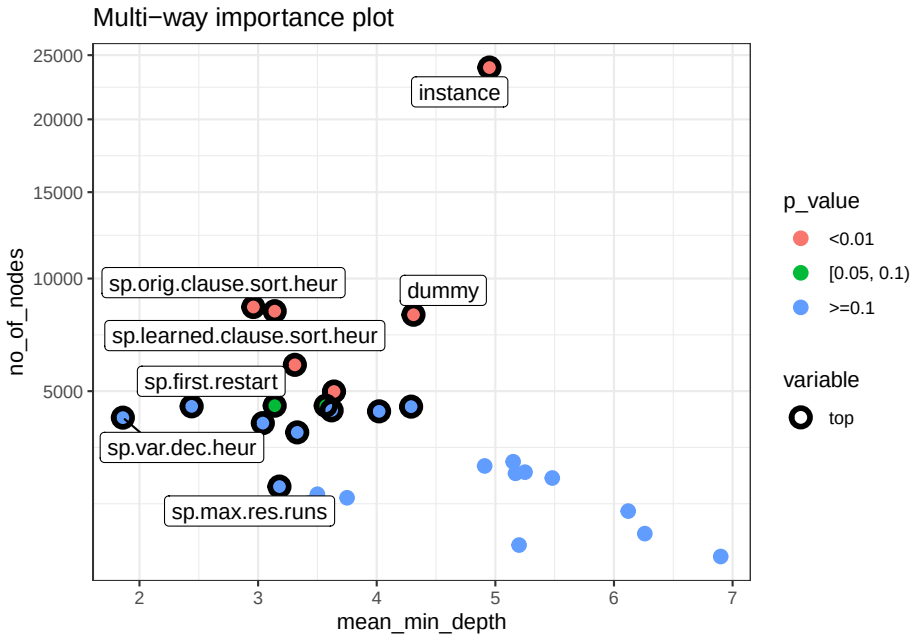
	variable	mean_min_depth	no_of_nodes	accuracy_decrease	gini_decrease	no_of_trees	times_a_root	p_value
27	sp.var.dec.heur	1.86	4073	0.1417939	386.49253	100	19	0.9999994
14	sp.phase.dec.heur	2.44	4457	0.1168833	406.04690	100	7	0.1352682
13	sp.orig.clause.sort.heur	2.96	8575	0.1740708	814.65339	100	2	0.0000000
17	sp.rand.var.dec.freq	3.04	3893	0.0966712	347.48702	100	6	1.0000000
8	sp.learned.clause.sort.heur	3.14	8375	0.1824176	796.15398	100	1	0.0000000
16	sp.rand.phase.scaling	3.14	4480	0.1027298	398.94018	100	6	0.0730551
12	sp.max.res.runs	3.18	2106	0.0806189	186.87331	100	11	1.0000000
7	sp.first.restart	3.31	6015	0.1065683	548.75191	100	0	0.0000000
21	sp.res.order.heur	3.33	3598	0.0949119	336.58265	100	5	1.0000000
11	sp.max.res.lit.inc	3.43	1732	0.0698537	153.73952	100	10	1.0000000
20	sp.res.cutoff.lits	3.50	1927	0.0632773	170.84109	100	8	1.0000000
18	sp.rand.var.dec.scaling	3.57	4479	0.1037946	396.51948	100	1	0.0751979
9	sp.learned.clauses.inc	3.62	4318	0.0879828	381.97991	100	3	0.8497824
15	sp.rand.phase.dec.freq	3.64	4987	0.0789392	449.05001	100	0	0.0000000
19	sp.res.cutoff.cls	3.75	1847	0.0646102	160.48579	100	12	1.0000000
10	sp.learned.size.factor	4.02	4285	0.0802541	374.29384	100	0	0.9389657
23	sp.restart.inc	4.29	4447	0.0699432	395.45278	100	0	0.1713650
1	dummy	4.31	8206	0.0802393	690.34881	100	0	0.0000000
4	sp.clause.decay	4.91	2627	0.0431276	223.32001	100	1	1.0000000
2	instance	4.95	23993	-0.0029934	1218.24155	100	0	0.0000000
28	sp.variable.decay	5.15	2741	0.0451787	234.91977	100	0	1.0000000
26	sp.var.activity.inc	5.17	2434	0.0403607	204.89132	100	2	1.0000000
6	sp.clause.inversion	5.20	941	0.0109864	85.48543	100	0	1.0000000
5	sp.clause.del.heur	5.25	2464	0.0353460	211.01104	100	0	1.0000000
3	sp.clause.activity.inc	5.48	2315	0.0281328	197.88398	100	0	1.0000000
25	sp.use.pure.literal.rule	6.12	1561	0.0225160	133.19032	100	0	1.0000000
24	sp.update.dec.queue	6.26	1132	0.0136616	97.40967	100	0	1.0000000
22	sp.resolution	6.90	767	0.0343973	60.63237	100	6	1.0000000

We can plot some measures using the randomForestExplainer package. In this case, since we are not predicting performance analizing the importance of parameters not including the instance would be an error.

```
plot_multi_way_importance(importance_frame, size_measure = "p_value", y_measure="no_of_nodes", x_measure
```

```
## Warning: Using alpha for a discrete variable is not advised.
```

```
## Warning: ggrepel: 9 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

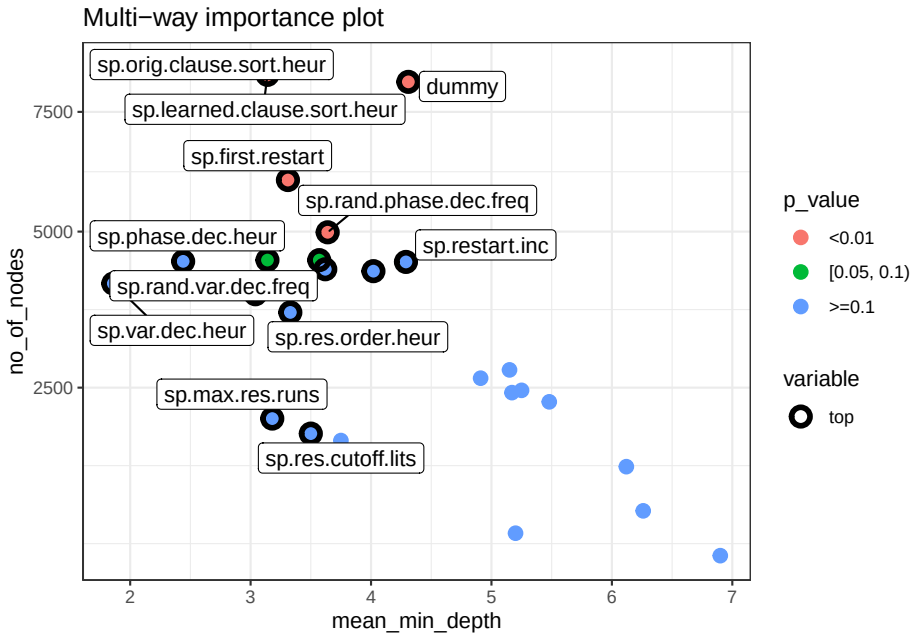


Removing the instance:

```
plot_multi_way_importance(importance_frame[importance_frame[,"variable"]!="instance",], size_measure = "
```

```
## Warning: Using alpha for a discrete variable is not advised.
```

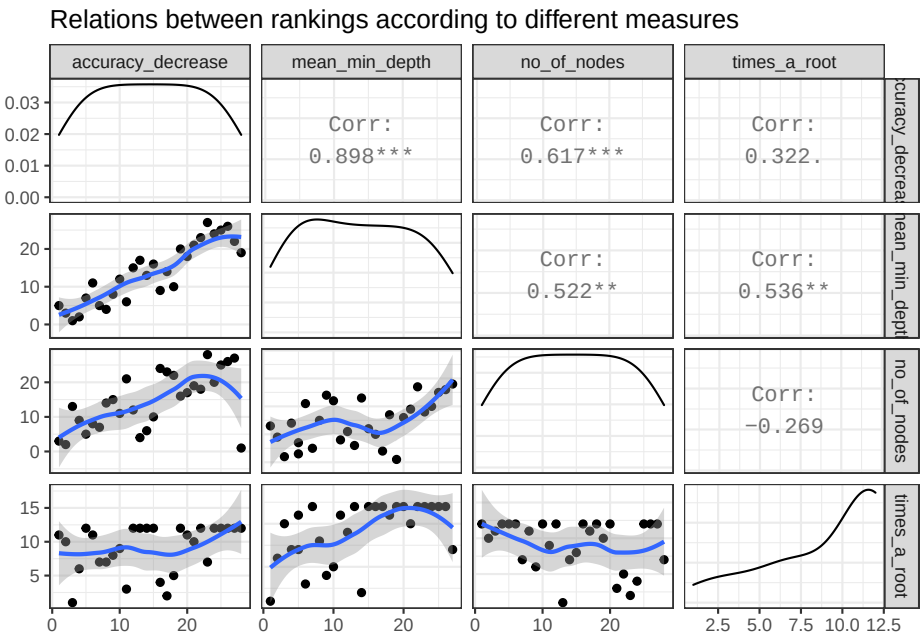
```
## Warning: ggrepel: 4 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



We can also visualize the relationship between importance measures ***TODO: check what this actually means and how can be used***:

```
#plot_importance_ggpairs(importance_frame)
plot_importance_rankings(importance_frame, measures=c("accuracy_decrease", "mean_min_depth", "no_of_node
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



After the analysis of conditional parameters the most important 5 parameters are (plus dummy added for interaction analysis):

```
print(important_parameters)
```

```
## [1] "sp.var.dec.heur"              "sp.phase.dec.heur"
## [3] "sp.orig.clause.sort.heur"     "sp.rand.var.dec.freq"
## [5] "sp.learned.clause.sort.heur" "sp.rand.phase.scaling"
## [7] "dummy"
```

None that nnls, which was detected initially as a very important parameter has been removed since its importance seemed to be related to its conditionallity to localsearch. This was confirmed by a the re training process described above.

We run the interaction analysis and find the importance of interactions:

```
kable(model$full_interactions_frame)  %>%  kable_styling(latex_options="scale_down")
```

variable	root_variable	mean_min_depth	occurrences	interaction	uncond_mean_min_depth
dummy	sp.var.dec.heur	2.1600	100	sp.var.dec.heur:dummy	3.29
sp.orig.clause.sort.heur	sp.phase.dec.heur	1.2653	99	sp.phase.dec.heur:sp.orig.clause.sort.heur	1.88
sp.phase.dec.heur	sp.phase.dec.heur	2.2453	99	sp.phase.dec.heur:sp.phase.dec.heur	1.71
dummy	sp.phase.dec.heur	2.0706	98	sp.phase.dec.heur:dummy	3.29
dummy	sp.rand.phase.scaling	2.2366	98	sp.rand.phase.scaling:dummy	3.29
sp.learned.clause.sort.heur	sp.rand.phase.scaling	1.5566	98	sp.rand.phase.scaling:sp.learned.clause.sort.heur	2.19
sp.orig.clause.sort.heur	sp.rand.phase.scaling	1.4066	98	sp.rand.phase.scaling:sp.orig.clause.sort.heur	1.88
sp.orig.clause.sort.heur	sp.var.dec.heur	1.3528	98	sp.var.dec.heur:sp.orig.clause.sort.heur	1.88
sp.rand.var.dec.freq	sp.phase.dec.heur	1.8306	98	sp.phase.dec.heur:sp.rand.var.dec.freq	1.81
sp.learned.clause.sort.heur	sp.phase.dec.heur	1.5859	97	sp.phase.dec.heur:sp.learned.clause.sort.heur	2.19
sp.learned.clause.sort.heur	sp.var.dec.heur	1.9092	97	sp.var.dec.heur:sp.learned.clause.sort.heur	2.19
sp.rand.phase.scaling	sp.rand.phase.scaling	2.4499	97	sp.rand.phase.scaling:sp.rand.phase.scaling	1.42
sp.var.dec.heur	sp.rand.phase.scaling	1.8199	97	sp.rand.phase.scaling:sp.var.dec.heur	1.22
sp.rand.var.dec.freq	sp.rand.phase.scaling	2.1932	96	sp.rand.phase.scaling:sp.rand.var.dec.freq	1.81
dummy	sp.orig.clause.sort.heur	2.4965	95	sp.orig.clause.sort.heur:dummy	3.29
dummy	sp.rand.var.dec.freq	2.6950	95	sp.rand.var.dec.freq:dummy	3.29
sp.learned.clause.sort.heur	sp.learned.clause.sort.heur	1.8280	95	sp.learned.clause.sort.heur:sp.learned.clause.sort.heur	2.19
sp.learned.clause.sort.heur	sp.rand.var.dec.freq	1.9450	95	sp.rand.var.dec.freq:sp.learned.clause.sort.heur	2.19
sp.orig.clause.sort.heur	sp.learned.clause.sort.heur	1.8680	95	sp.learned.clause.sort.heur:sp.orig.clause.sort.heur	1.88
sp.rand.var.dec.freq	sp.rand.var.dec.freq	3.0850	95	sp.rand.var.dec.freq:sp.rand.var.dec.freq	1.81
sp.rand.var.dec.freq	sp.var.dec.heur	2.4320	95	sp.var.dec.heur:sp.rand.var.dec.freq	1.81
sp.var.dec.heur	sp.phase.dec.heur	2.3065	95	sp.phase.dec.heur:sp.var.dec.heur	1.22
dummy	sp.learned.clause.sort.heur	2.7836	94	sp.learned.clause.sort.heur:dummy	3.29
sp.orig.clause.sort.heur	sp.rand.var.dec.freq	2.3840	94	sp.rand.var.dec.freq:sp.orig.clause.sort.heur	1.88
sp.var.dec.heur	sp.learned.clause.sort.heur	2.4936	94	sp.learned.clause.sort.heur:sp.var.dec.heur	1.22
sp.var.dec.heur	sp.var.dec.heur	2.3784	94	sp.var.dec.heur:sp.var.dec.heur	1.22
sp.learned.clause.sort.heur	sp.orig.clause.sort.heur	2.2524	92	sp.orig.clause.sort.heur:sp.learned.clause.sort.heur	2.19
sp.orig.clause.sort.heur	sp.orig.clause.sort.heur	2.4924	92	sp.orig.clause.sort.heur:sp.orig.clause.sort.heur	1.88
sp.var.dec.heur	sp.rand.var.dec.freq	2.7120	92	sp.rand.var.dec.freq:sp.var.dec.heur	1.22
sp.rand.var.dec.freq	sp.orig.clause.sort.heur	2.9277	91	sp.orig.clause.sort.heur:sp.rand.var.dec.freq	1.81
sp.phase.dec.heur	sp.var.dec.heur	3.0940	90	sp.var.dec.heur:sp.phase.dec.heur	1.71
sp.var.dec.heur	sp.orig.clause.sort.heur	3.3930	90	sp.orig.clause.sort.heur:sp.var.dec.heur	1.22
sp.rand.var.dec.freq	sp.learned.clause.sort.heur	3.2716	89	sp.learned.clause.sort.heur:sp.rand.var.dec.freq	1.81
sp.phase.dec.heur	sp.rand.var.dec.freq	3.5270	87	sp.rand.var.dec.freq:sp.phase.dec.heur	1.71
sp.rand.phase.scaling	sp.var.dec.heur	4.2760	85	sp.var.dec.heur:sp.rand.phase.scaling	1.42
sp.rand.phase.scaling	sp.phase.dec.heur	4.3248	84	sp.phase.dec.heur:sp.rand.phase.scaling	1.42
sp.phase.dec.heur	sp.rand.phase.scaling	4.8594	82	sp.rand.phase.scaling:sp.phase.dec.heur	1.71
sp.rand.phase.scaling	sp.rand.var.dec.freq	4.7510	81	sp.rand.var.dec.freq:sp.rand.phase.scaling	1.42
dummy	dummy	4.2098	77	dummy:dummy	3.29
sp.orig.clause.sort.heur	dummy	3.4124	76	dummy:sp.orig.clause.sort.heur	1.88
sp.phase.dec.heur	sp.learned.clause.sort.heur	4.9744	76	sp.learned.clause.sort.heur:sp.phase.dec.heur	1.71
sp.learned.clause.sort.heur	dummy	3.6950	75	dummy:sp.learned.clause.sort.heur	2.19
sp.phase.dec.heur	sp.orig.clause.sort.heur	5.2325	75	sp.orig.clause.sort.heur:sp.phase.dec.heur	1.71
sp.rand.phase.scaling	sp.orig.clause.sort.heur	5.4678	74	sp.orig.clause.sort.heur:sp.rand.phase.scaling	1.42
sp.rand.phase.scaling	sp.learned.clause.sort.heur	5.6212	73	sp.learned.clause.sort.heur:sp.rand.phase.scaling	1.42
sp.rand.var.dec.freq	dummy	5.1784	66	dummy:sp.rand.var.dec.freq	1.81
sp.var.dec.heur	dummy	5.1284	66	dummy:sp.var.dec.heur	1.22
sp.phase.dec.heur	dummy	4.9962	63	dummy:sp.phase.dec.heur	1.71
sp.rand.phase.scaling	dummy	5.3866	59	dummy:sp.rand.phase.scaling	1.42

Once we filter dummy interactions and aggregate bidirectional interactions we get a matrix where the importance of interactions are summarized:

```
print(model$interactions_frame)
```

```
## [1] variable              root_variable         mean_min_depth
## [4] occurrences           interaction           uncond_mean_min_depth
## <0 rows> (or 0-length row.names)
```

3.Configuration performance as dependent variable

In this example we use the performance as the dependent variable when training.

```
load("model_data/model-spear-cat-performance.Rdata")
importance_frame = model$importance_frame
important_parameters = model$important_parameters
kable(importance_frame[order(importance_frame[,"mean_min_depth"]),])  %>%  kable_styling(latex_options=
```

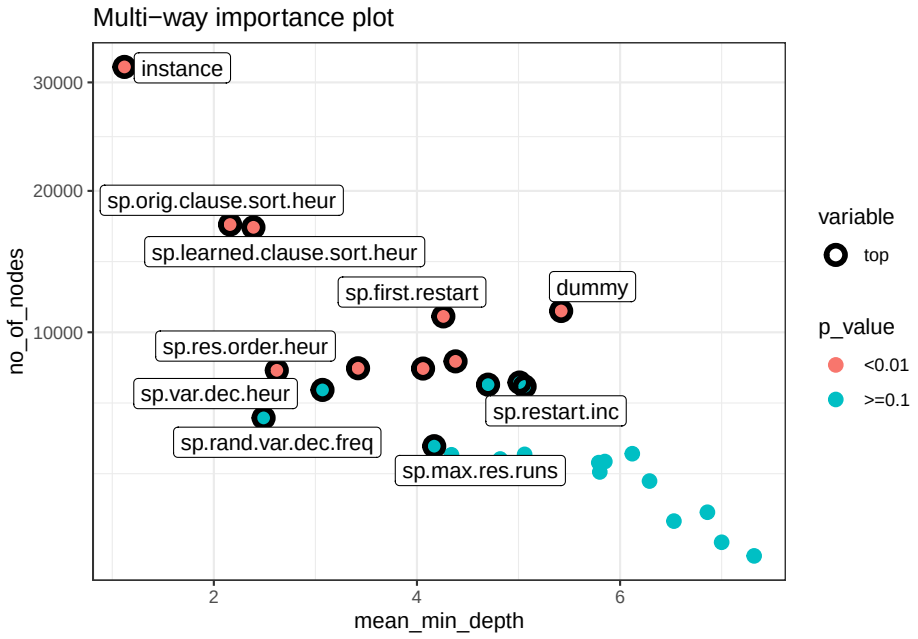
	variable	mean_min_depth	no_of_nodes	mse_increase	node_purity_increase	no_of_trees	times_a_root	p_value
2	instance	1.12	31591	4996.344412	20417672.61	100	27	0.0000000
27	sp.var.dec.heur	1.52	6890	2089.292793	3926199.26	100	23	0.9999972
13	sp.orig.clause.sort.heur	2.16	17309	71.808048	3354998.88	100	10	0.0000000
8	sp.learned.clause.sort.heur	2.39	17095	11.549379	3415857.56	100	7	0.0000000
17	sp.rand.var.dec.freq	2.49	5613	395.985981	1587212.80	100	12	1.0000000
21	sp.res.order.heur	2.62	7896	89.860266	1688270.34	100	3	0.0000000
18	sp.rand.var.dec.scaling	3.07	6908	265.782510	1303888.22	100	6	0.9999923
14	sp.phase.dec.heur	3.42	8004	165.914513	1388640.84	100	4	0.0000000
16	sp.rand.phase.scaling	4.06	7989	62.936127	931604.21	100	0	0.0000000
12	sp.max.res.runs	4.17	4440	40.282389	534401.27	100	4	1.0000000
7	sp.first.restart	4.26	10952	29.280770	1338772.10	100	0	0.0000000
11	sp.max.res.lit.inc	4.34	4111	71.335309	443767.13	100	3	1.0000000
15	sp.rand.phase.dec.freq	4.38	8369	18.236511	949459.90	100	0	0.0000000
10	sp.learned.size.factor	4.70	7168	5.774730	690410.28	100	0	0.8820183
20	sp.res.cutoff.lits	4.82	3954	19.718712	346594.07	100	1	1.0000000
9	sp.learned.clauses.inc	5.01	7274	89.879038	648744.01	100	0	0.4666264
19	sp.res.cutoff.cls	5.06	4127	25.162924	331090.57	100	0	1.0000000
23	sp.restart.inc	5.06	7078	8.497704	621893.47	100	0	0.9883118
1	dummy	5.42	11287	4.031441	678589.70	100	0	0.0000000
4	sp.clause.decay	5.79	3821	-8.402375	267148.00	100	0	1.0000000
5	sp.clause.del.heur	5.80	3491	18.959272	277269.99	100	0	1.0000000
26	sp.var.activity.inc	5.85	3861	21.292059	244921.90	100	0	1.0000000
28	sp.variable.decay	6.12	4151	-4.805710	261104.91	100	0	1.0000000
3	sp.clause.activity.inc	6.29	3183	14.073640	262955.85	100	0	1.0000000
22	sp.resolution	6.53	1997	8.284435	108558.54	100	0	1.0000000
25	sp.use.pure.literal.rule	6.86	2236	1.438529	127524.82	100	0	1.0000000
24	sp.update.dec.queue	7.00	1481	7.300873	156562.63	100	0	1.0000000
6	sp.clause.inversion	7.32	1190	-3.030569	93085.51	100	0	1.0000000

We can plot some measures using the randomForestExplainer package. In this case, since we are not predicting performance analizing the importance of parameters not including the instance would be an error.

```
plot_multi_way_importance(importance_frame, size_measure = "p_value", y_measure="no_of_nodes", x_measure
```

```
## Warning: Using alpha for a discrete variable is not advised.
```

```
## Warning: ggrepel: 6 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```

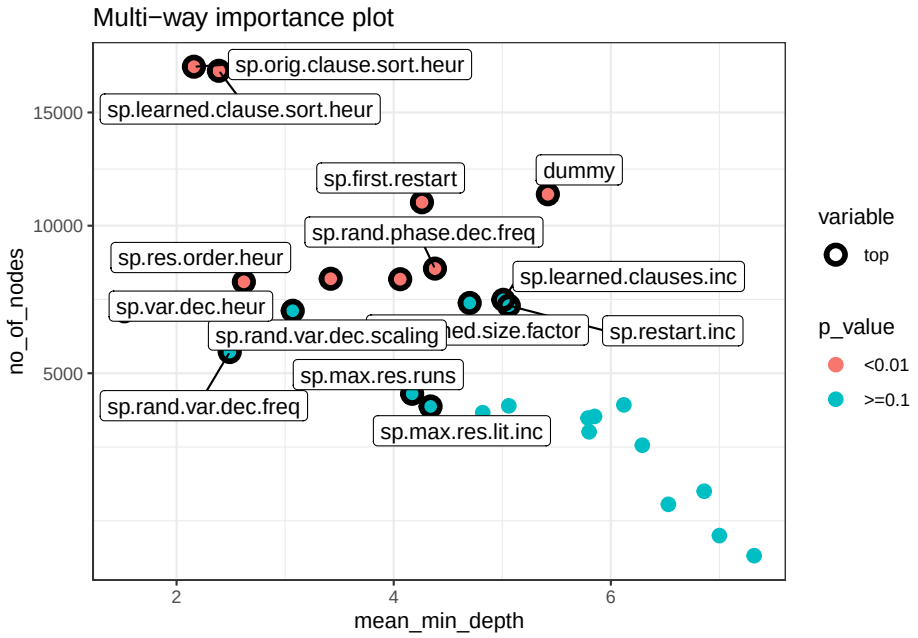


Removing the instance:

```
plot_multi_way_importance(importance_frame[importance_frame[,"variable"]!="instance",], size_measure = "
```

```
## Warning: Using alpha for a discrete variable is not advised.
```

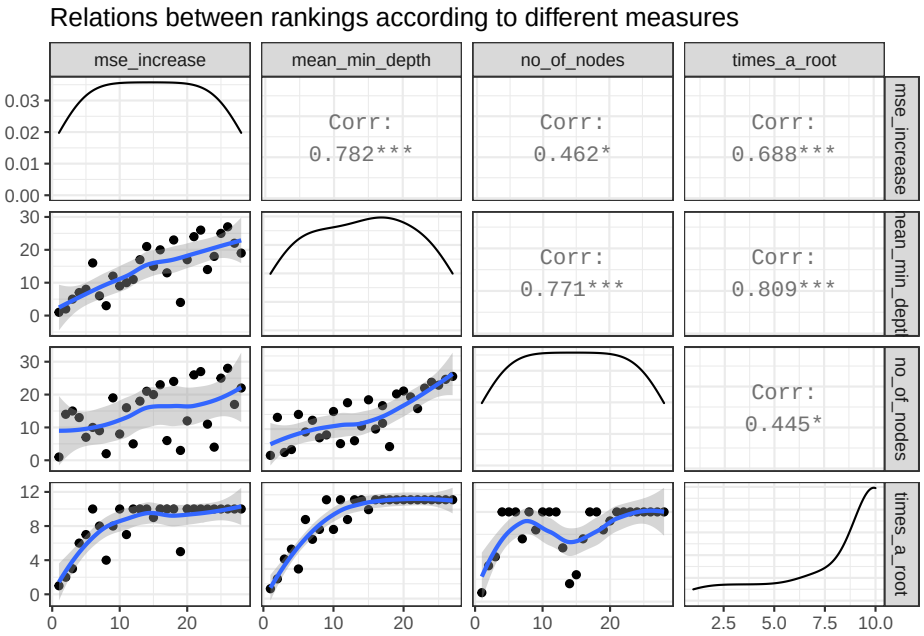
```
## Warning: ggrepel: 2 unlabeled data points (too many overlaps). Consider
## increasing max.overlaps
```



We can also visualize the relationship between importance measures ***TODO: check what this actually means and how can be used***:

```
#plot_importance_ggpairs(importance_frame)
plot_importance_rankings(importance_frame, measures=c("mse_increase", "mean_min_depth", "no_of_nodes", "
```

```
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
## `geom_smooth()` using formula 'y ~ x'
```



After the analysis of conditional parameters the most important 5 parameters are (plus dummy added for interaction analysis):

```
print(important_parameters)
```

```
## [1] "instance"                  "sp.var.dec.heur"
## [3] "sp.orig.clause.sort.heur"  "sp.learned.clause.sort.heur"
## [5] "sp.rand.var.dec.freq"      "dummy"
```

None that nnls, which was detected initially as a very important parameter has been removed since its importance seemed to be related to its conditionallity to localsearch. This was confirmed by a the re training process described above.

We run the interaction analysis and find the importance of interactions:

```
kable(model$full_interactions_frame) %>% kable_styling(latex_options="scale_down")
```

variable	root_variable	mean_min_depth	occurrences	interaction	uncond_mean_min_depth
dummy	sp.learned.clause.sort.heur	1.6900	100	sp.learned.clause.sort.heur:dummy	3.54
dummy	sp.var.dec.heur	2.0000	100	sp.var.dec.heur:dummy	3.54
instance	instance	0.7300	100	instance:instance	0.91
instance	sp.var.dec.heur	0.6000	100	sp.var.dec.heur:instance	0.91
sp.learned.clause.sort.heur	instance	0.9900	100	instance:sp.learned.clause.sort.heur	1.94
sp.learned.clause.sort.heur	sp.orig.clause.sort.heur	1.0300	100	sp.orig.clause.sort.heur:sp.learned.clause.sort.heur	1.94
sp.learned.clause.sort.heur	sp.rand.var.dec.freq	1.0000	100	sp.rand.var.dec.freq:sp.learned.clause.sort.heur	1.94
sp.learned.clause.sort.heur	sp.var.dec.heur	0.8900	100	sp.var.dec.heur:sp.learned.clause.sort.heur	1.94
sp.orig.clause.sort.heur	instance	1.0000	100	instance:sp.orig.clause.sort.heur	1.75
sp.orig.clause.sort.heur	sp.learned.clause.sort.heur	0.8900	100	sp.learned.clause.sort.heur:sp.orig.clause.sort.heur	1.75
sp.orig.clause.sort.heur	sp.orig.clause.sort.heur	1.3400	100	sp.orig.clause.sort.heur:sp.orig.clause.sort.heur	1.75
sp.orig.clause.sort.heur	sp.var.dec.heur	1.0800	100	sp.var.dec.heur:sp.orig.clause.sort.heur	1.75
sp.rand.var.dec.freq	instance	1.3400	100	instance:sp.rand.var.dec.freq	1.73
sp.rand.var.dec.freq	sp.var.dec.heur	1.1000	100	sp.var.dec.heur:sp.rand.var.dec.freq	1.73
sp.var.dec.heur	sp.var.dec.heur	1.5000	100	sp.var.dec.heur:sp.var.dec.heur	1.21
dummy	instance	2.4817	99	instance:dummy	3.54
dummy	sp.orig.clause.sort.heur	2.1496	99	sp.orig.clause.sort.heur:dummy	3.54
dummy	sp.rand.var.dec.freq	1.9683	99	sp.rand.var.dec.freq:dummy	3.54
sp.orig.clause.sort.heur	sp.rand.var.dec.freq	1.0983	99	sp.rand.var.dec.freq:sp.orig.clause.sort.heur	1.75
sp.var.dec.heur	instance	1.0717	99	instance:sp.var.dec.heur	1.21
sp.var.dec.heur	sp.orig.clause.sort.heur	1.2896	99	sp.orig.clause.sort.heur:sp.var.dec.heur	1.21
sp.learned.clause.sort.heur	sp.learned.clause.sort.heur	1.9050	98	sp.learned.clause.sort.heur:sp.learned.clause.sort.heur	1.94
sp.rand.var.dec.freq	sp.orig.clause.sort.heur	1.6992	98	sp.orig.clause.sort.heur:sp.rand.var.dec.freq	1.73
sp.rand.var.dec.freq	sp.rand.var.dec.freq	2.4466	98	sp.rand.var.dec.freq:sp.rand.var.dec.freq	1.73
sp.var.dec.heur	sp.learned.clause.sort.heur	1.3750	98	sp.learned.clause.sort.heur:sp.var.dec.heur	1.21
sp.var.dec.heur	sp.rand.var.dec.freq	1.5766	98	sp.rand.var.dec.freq:sp.var.dec.heur	1.21
instance	sp.orig.clause.sort.heur	1.1788	97	sp.orig.clause.sort.heur:instance	0.91
sp.rand.var.dec.freq	sp.learned.clause.sort.heur	1.9025	97	sp.learned.clause.sort.heur:sp.rand.var.dec.freq	1.73
instance	sp.rand.var.dec.freq	1.3832	96	sp.rand.var.dec.freq:instance	0.91
instance	sp.learned.clause.sort.heur	1.6275	95	sp.learned.clause.sort.heur:instance	0.91
sp.learned.clause.sort.heur	dummy	1.9496	92	dummy:sp.learned.clause.sort.heur	1.94
dummy	dummy	3.0333	91	dummy:dummy	3.54
sp.orig.clause.sort.heur	dummy	2.0833	91	dummy:sp.orig.clause.sort.heur	1.75
instance	dummy	2.6255	85	dummy:instance	0.91
sp.rand.var.dec.freq	dummy	3.2729	83	dummy:sp.rand.var.dec.freq	1.73
sp.var.dec.heur	dummy	3.2066	82	dummy:sp.var.dec.heur	1.21

Once we filter dummy interactions and aggregate bidirectional interactions we get a matrix where the importance of interactions are summarized:

```
print(model$interactions_frame)
```

```
## [1] variable              root_variable         mean_min_depth
## [4] occurrences           interaction           uncond_mean_min_depth
## <0 rows> (or 0-length row.names)
```