Evaluate testing data (binary-class) - XGBoost

EVE W.

2019-04-05

Contents

```
## user input
project_home <- "~/EVE/examples"
project_name <- "xgboostR_multi_1"
```

0. Load Data

```
## Warning: `data_frame()` is deprecated, use `tibble()`.
## This warning is displayed once per session.
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   Patient_ID = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## 300 of samples were used
```

```
## 100 of full features
```

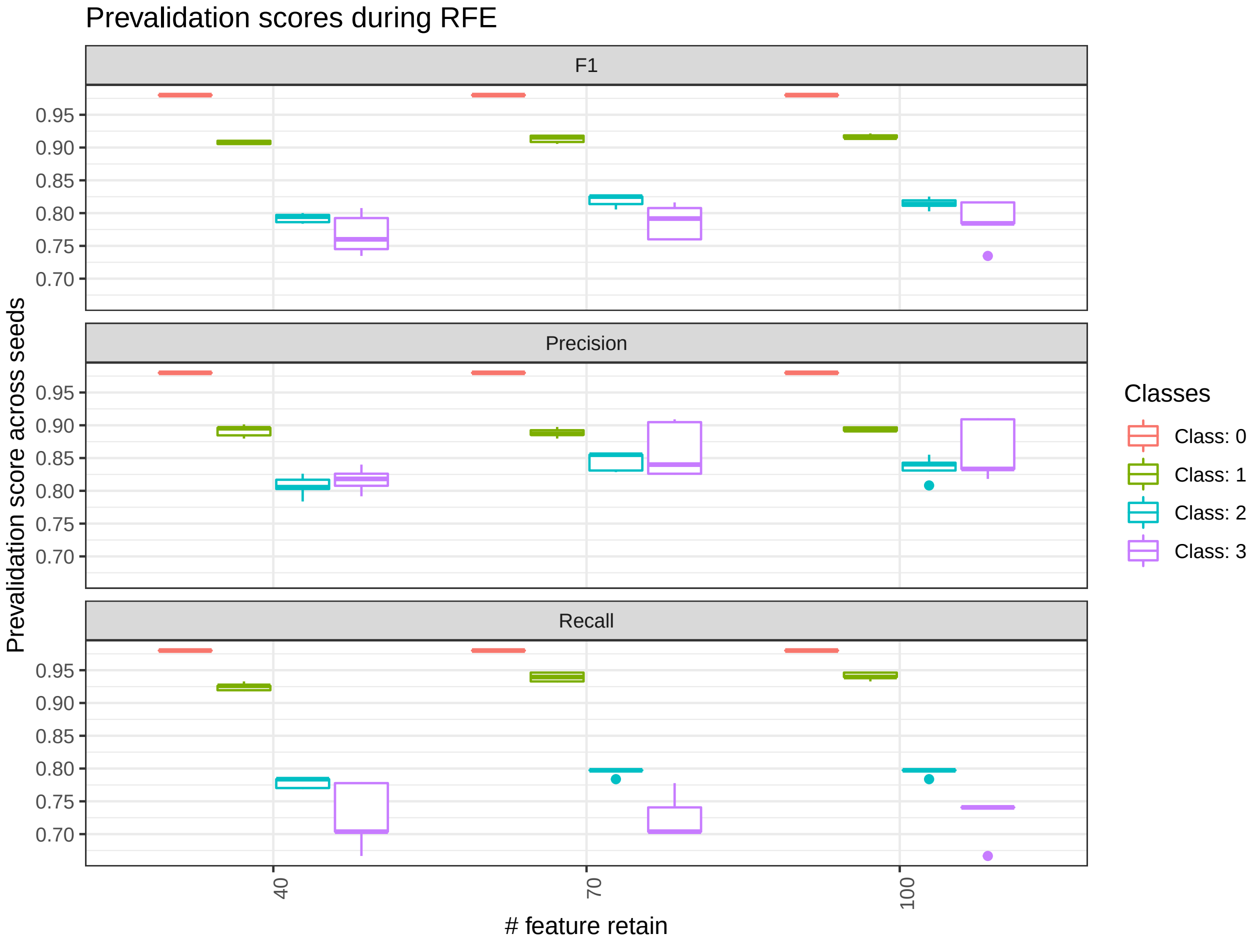
```
## 5 runs, each run contains 300 CVs.
```

```
## Labels:
```

run with XGBoost.r evaluation metric: NA.

1. Scores

1.1 Scores per Class



Confusion Matrix

```
## confusion matrix at feature size = 100
## sum across 5 seeds
```

```
##           Reference
## Prediction   0   1   2   3
##          0 245   5   0   0
##          1   0 701  68  15
##          2   0  36 294  22
##          3   5   3   8  98
```

1.2 Average score

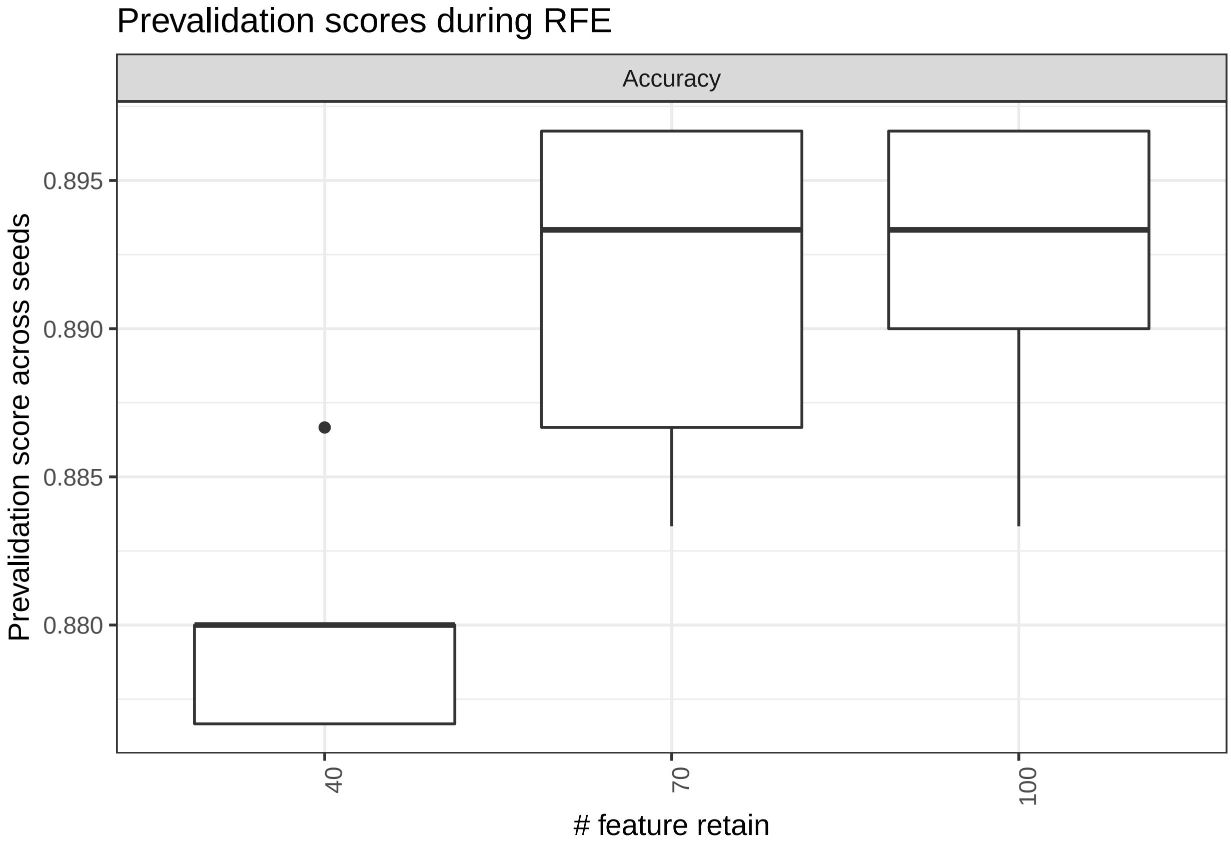
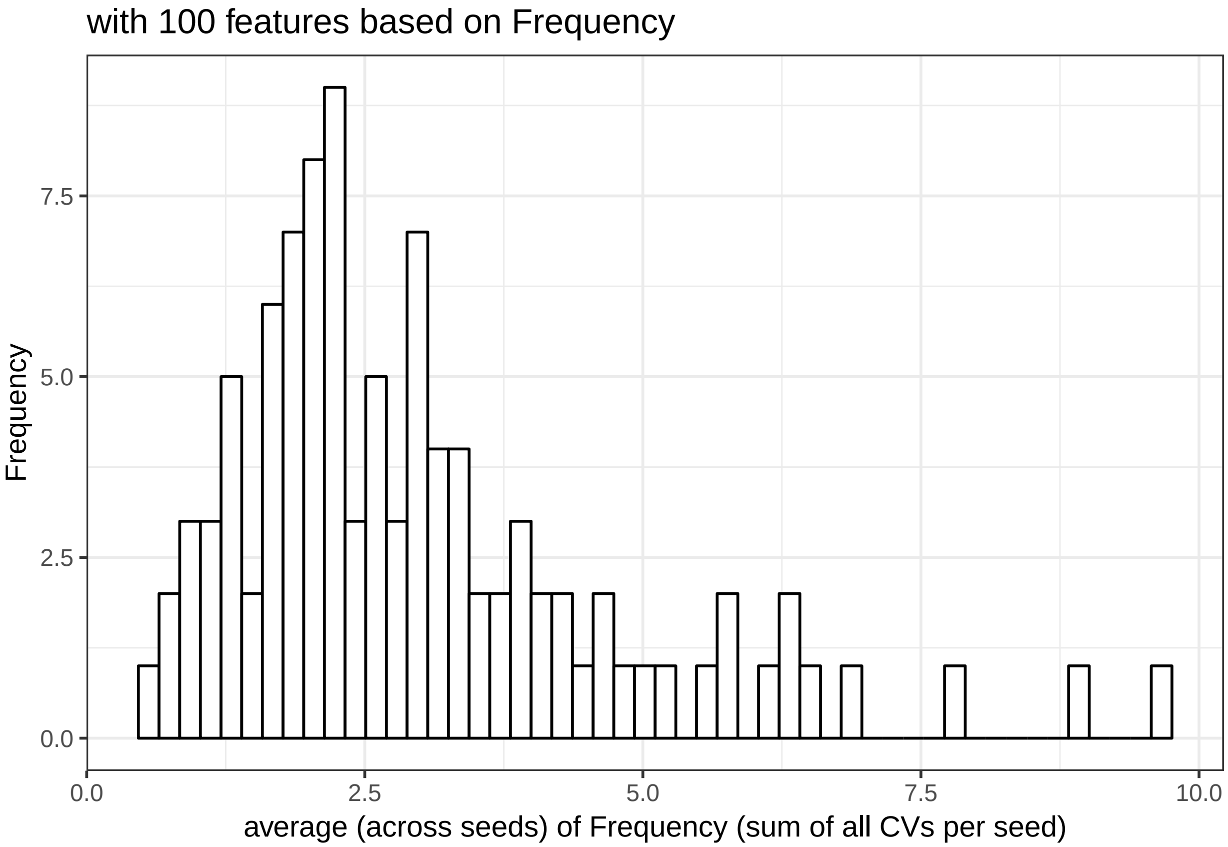
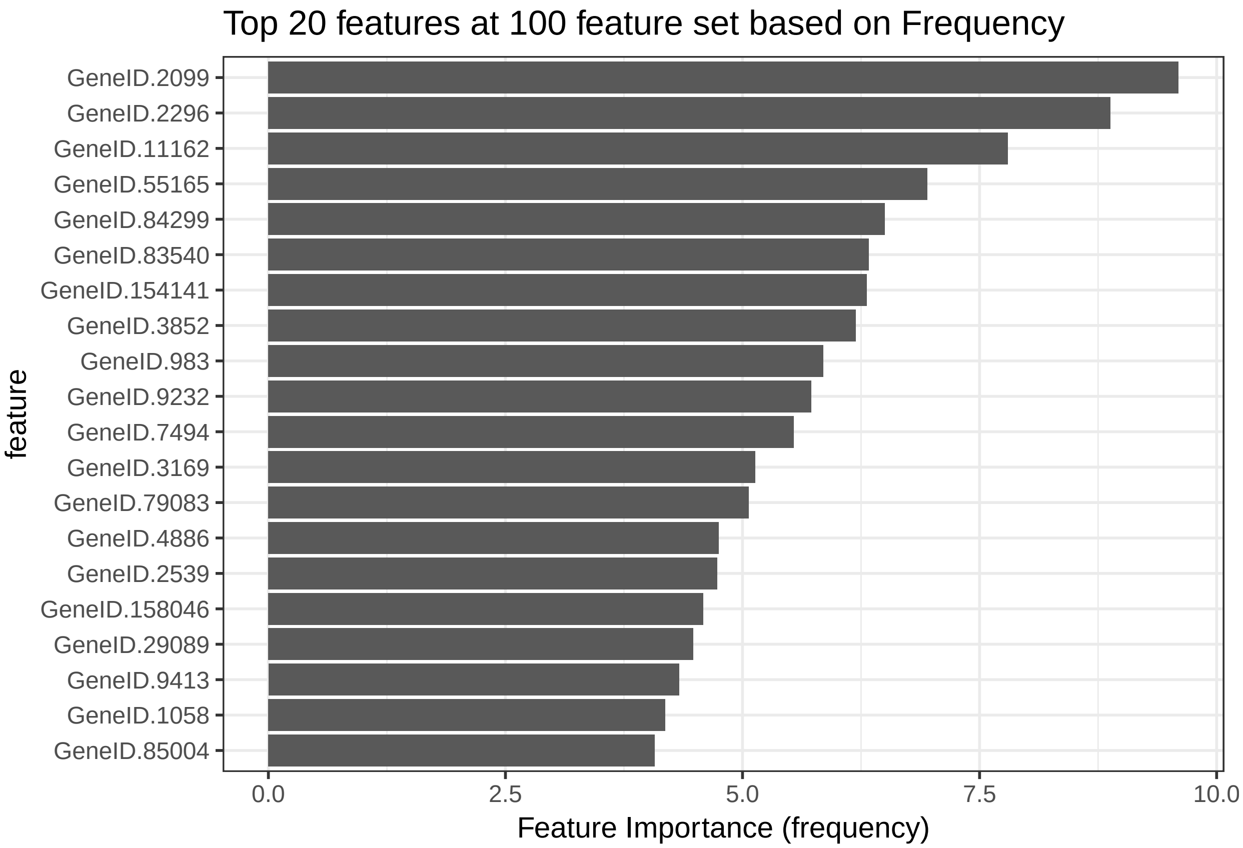


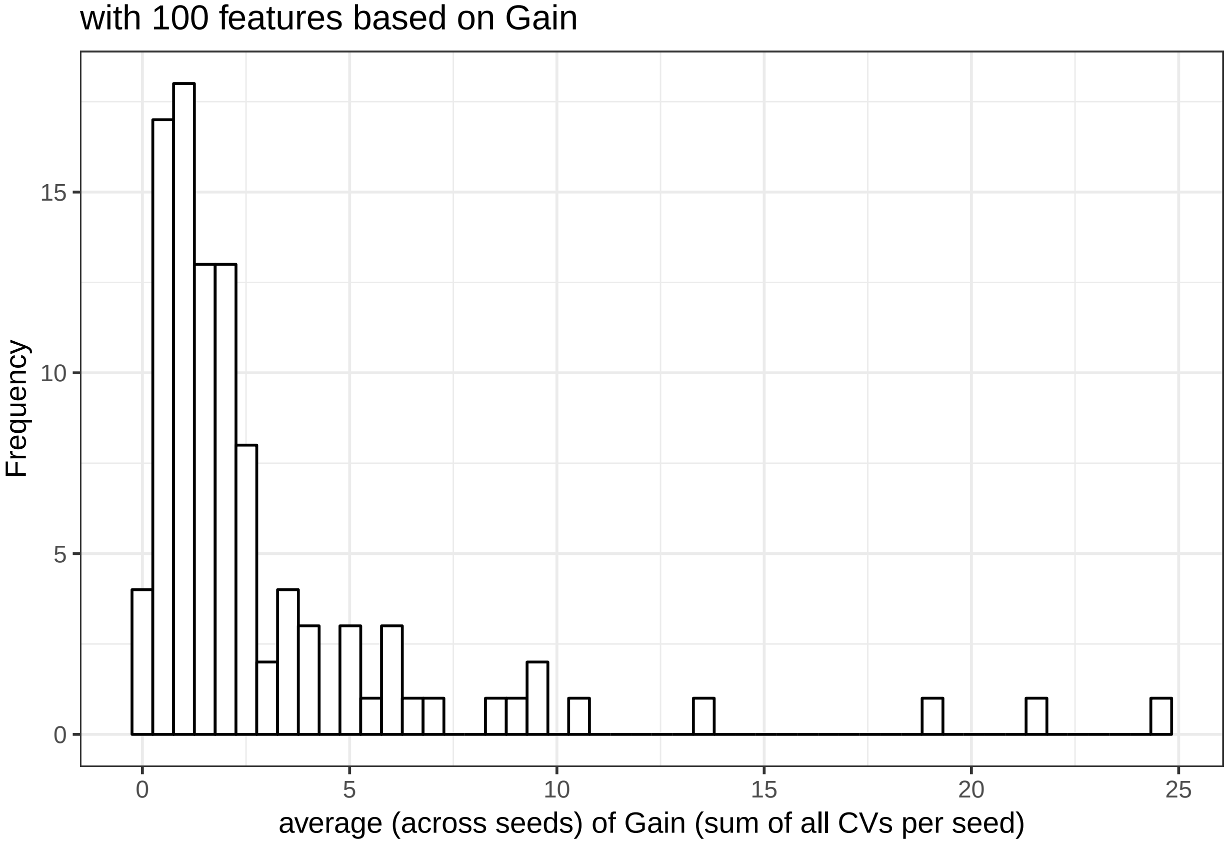
Table 1: best scores

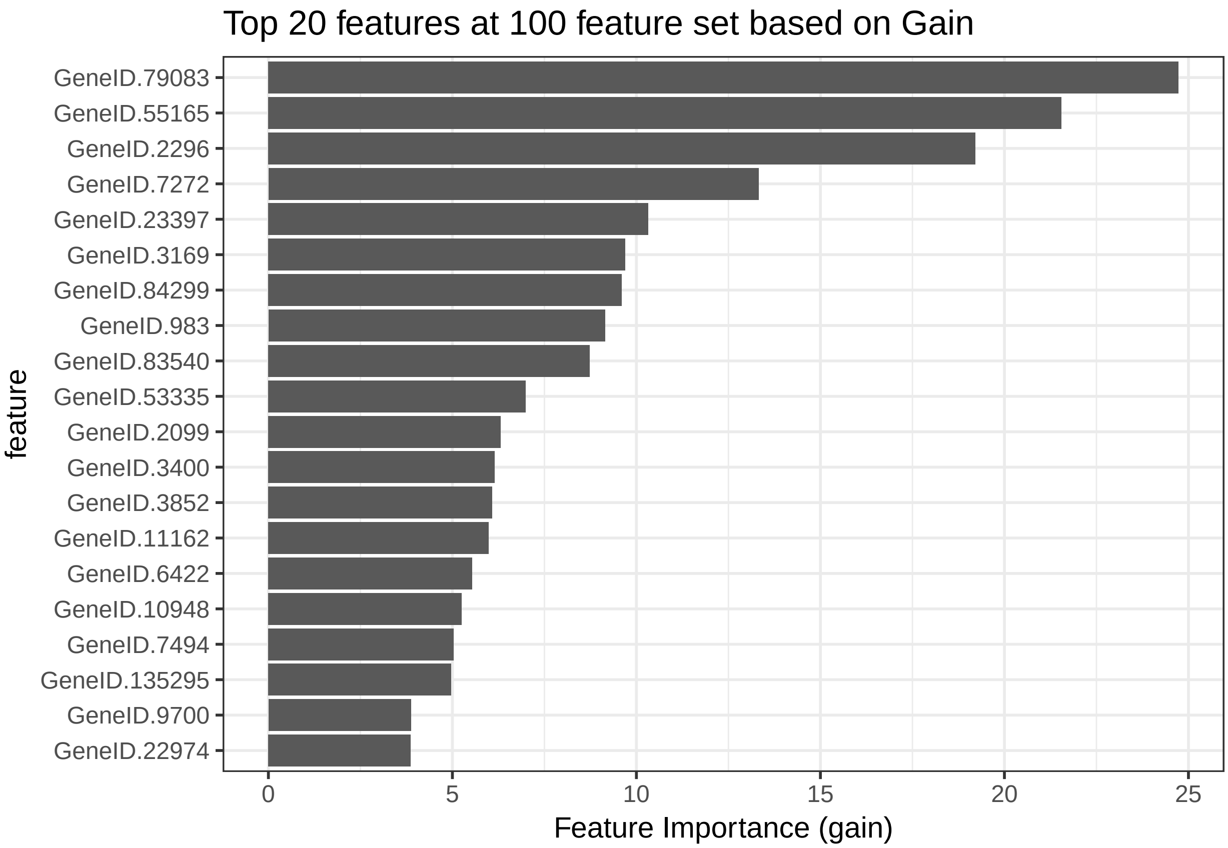
metrics	size.max	median.max	size.min	median.min
Accuracy	70	0.893	40	0.880
F1	100	0.878	40	0.861
Precision	70	0.893	40	0.872
Recall	100	0.864	40	0.847

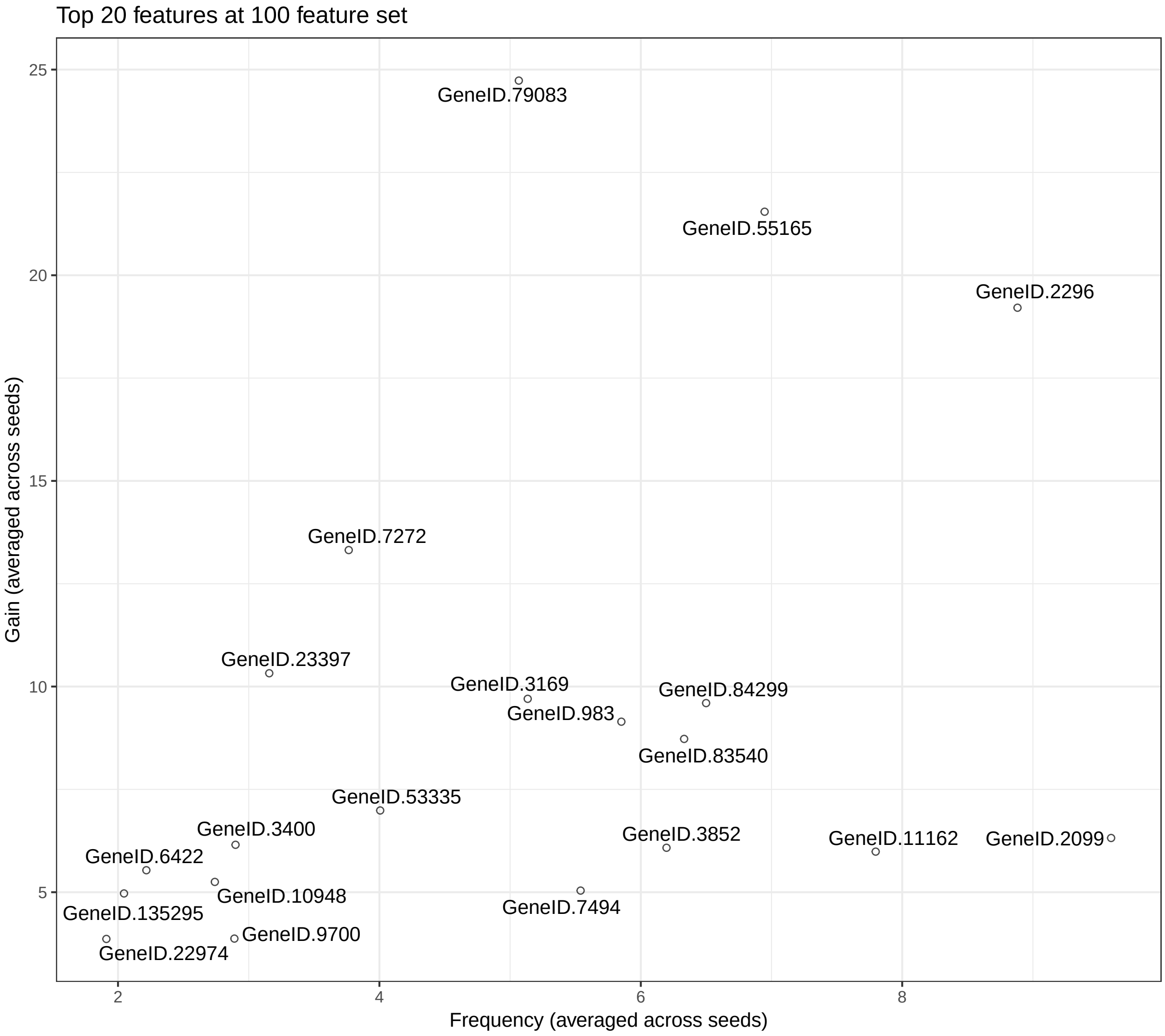
2. Important Features

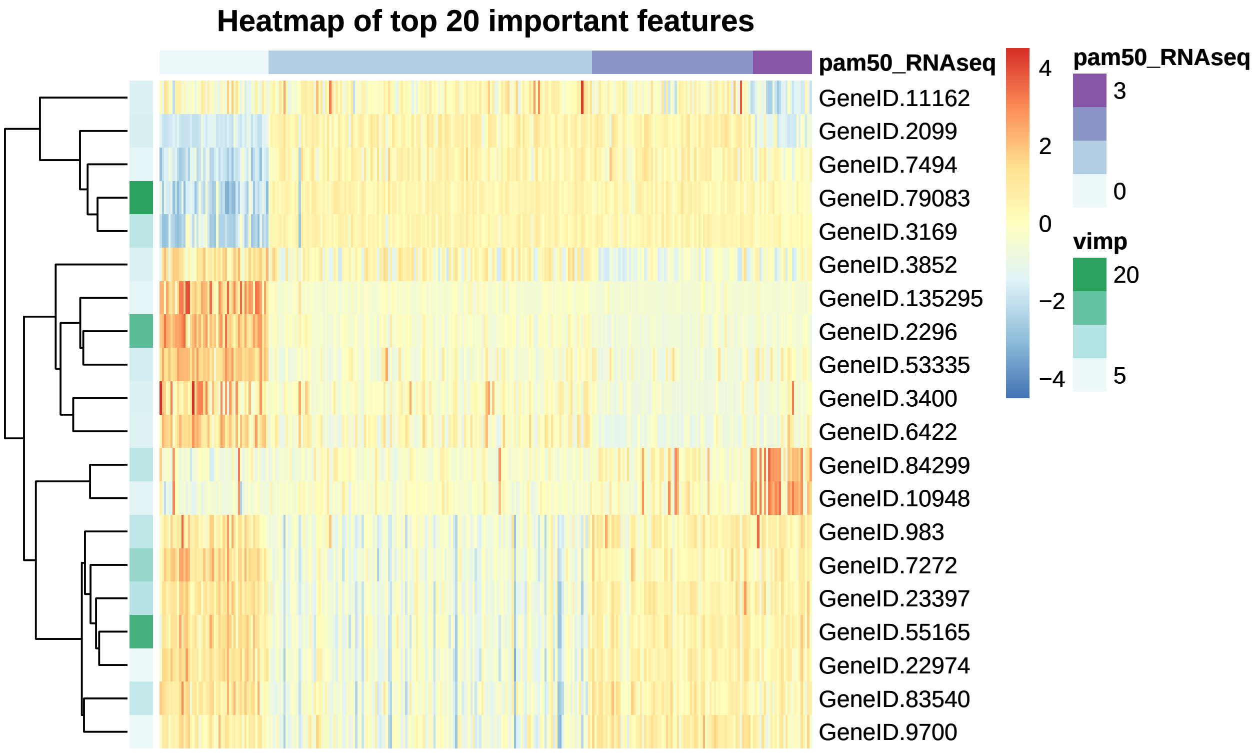



Top 20 features at 100 feature set based on Frequency
feature
GeneID.2099
GeneID.2296
GeneID.11162
GeneID.55165
GeneID.84299
GeneID.83540
GeneID.154141
GeneID.3852
GeneID.983
GeneID.9232
GeneID.7494
GeneID.3169
GeneID.79083
GeneID.4886
GeneID.2539
GeneID.158046
GeneID.29089
GeneID.9413
GeneID.1058
GeneID.85004
0.0
2.5
5.0
7.5
10.0
Feature Importance (frequency)


with 100 features based on Gain
Frequency
0
5
10
15
0
5
10
15
20
25
average (across seeds) of Gain (sum of all CVs per seed)


Top 20 features at 100 feature set based on Gain
feature
GeneID.79083
GeneID.55165
GeneID.2296
GeneID.7272
GeneID.23397
GeneID.3169
GeneID.84299
GeneID.983
GeneID.83540
GeneID.53335
GeneID.2099
GeneID.3400
GeneID.3852
GeneID.11162
GeneID.6422
GeneID.10948
GeneID.7494
GeneID.135295
GeneID.9700
GeneID.22974
0
5
10
15
20
25
Feature Importance (gain)

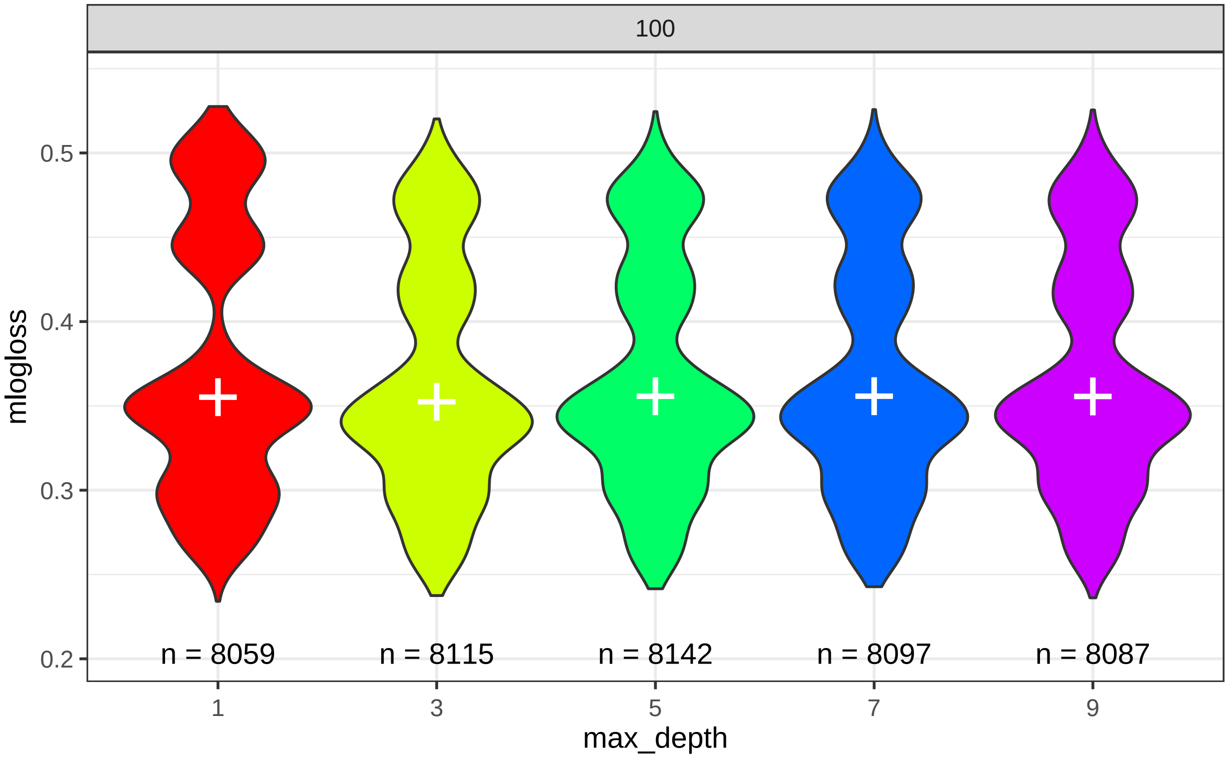

Top 20 features at 100 feature set
Gain (averaged across seeds)
Frequency (averaged across seeds)
25
20
15
10
5
2
4
6
8
GeneID.79083
GeneID.55165
GeneID.2296
GeneID.7272
GeneID.23397
GeneID.3169
GeneID.84299
GeneID.983
GeneID.83540
GeneID.53335
GeneID.3400
GeneID.3852
GeneID.11162
GeneID.2099
GeneID.6422
GeneID.10948
GeneID.135295
GeneID.7494
GeneID.9700
GeneID.22974

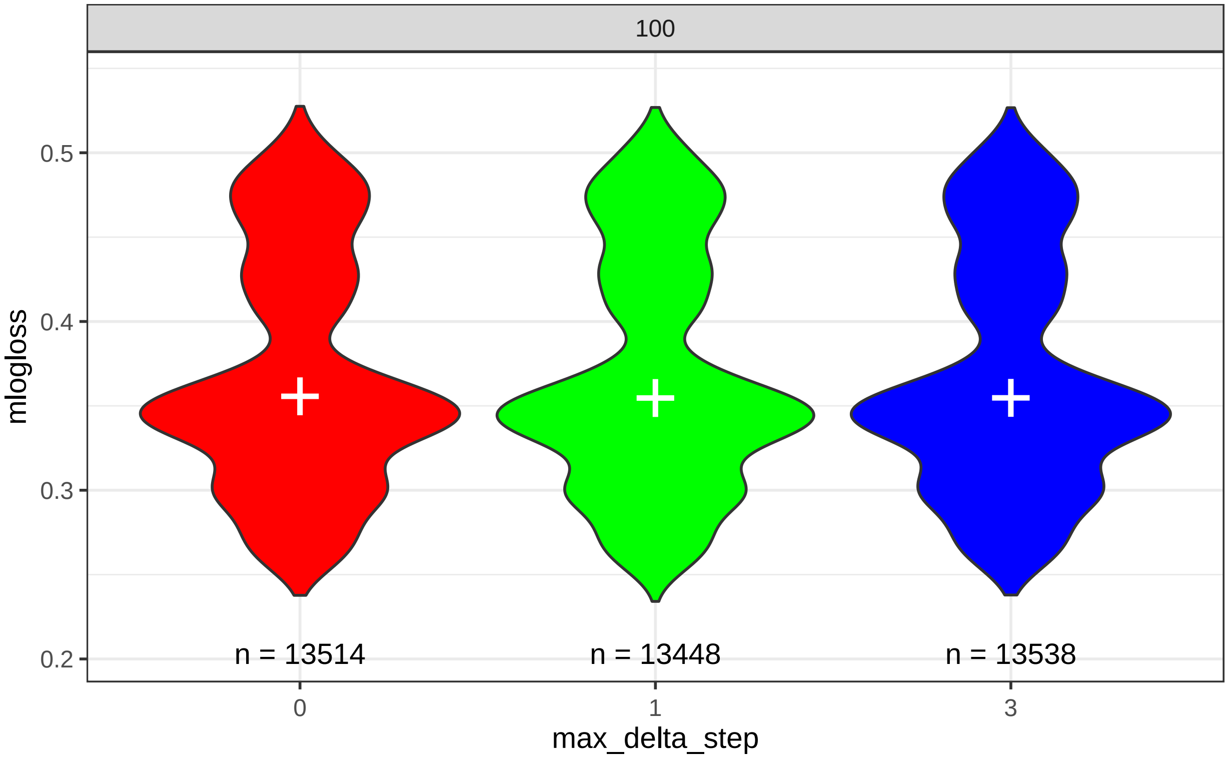


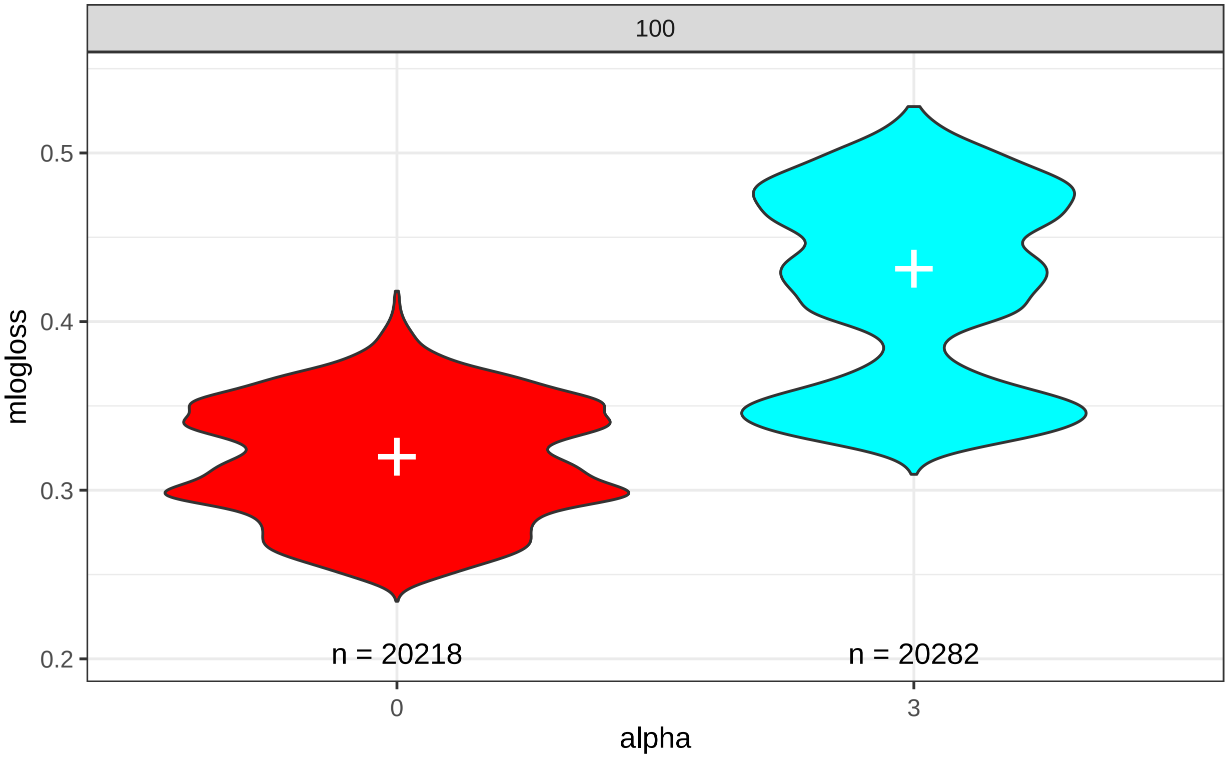
3. Hyper-parameters

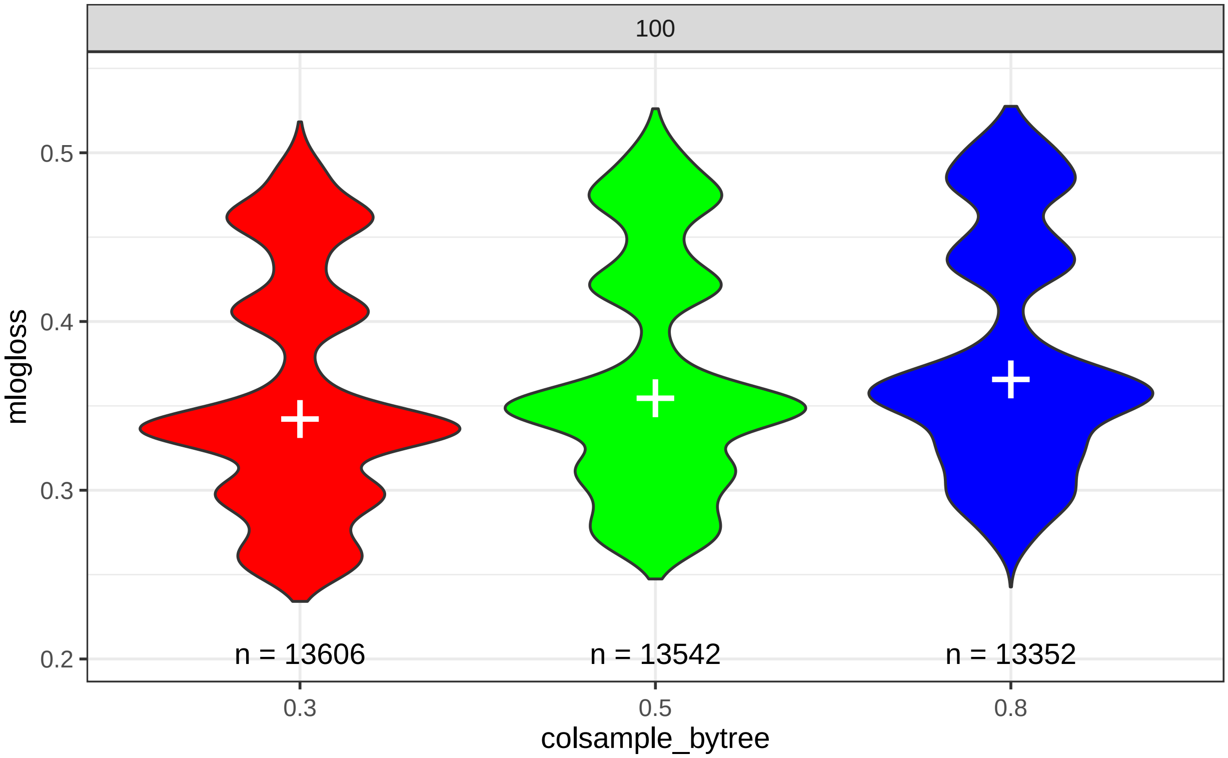
parameter optimization file (40500 records) includes 5 seeds. Each seed generates 300 cv splits. Within each cv split, there is a 1 step RFE (at 100). So 40500 / 5 / 300 / 1 = 27 parameter combinations tried in each cv split.

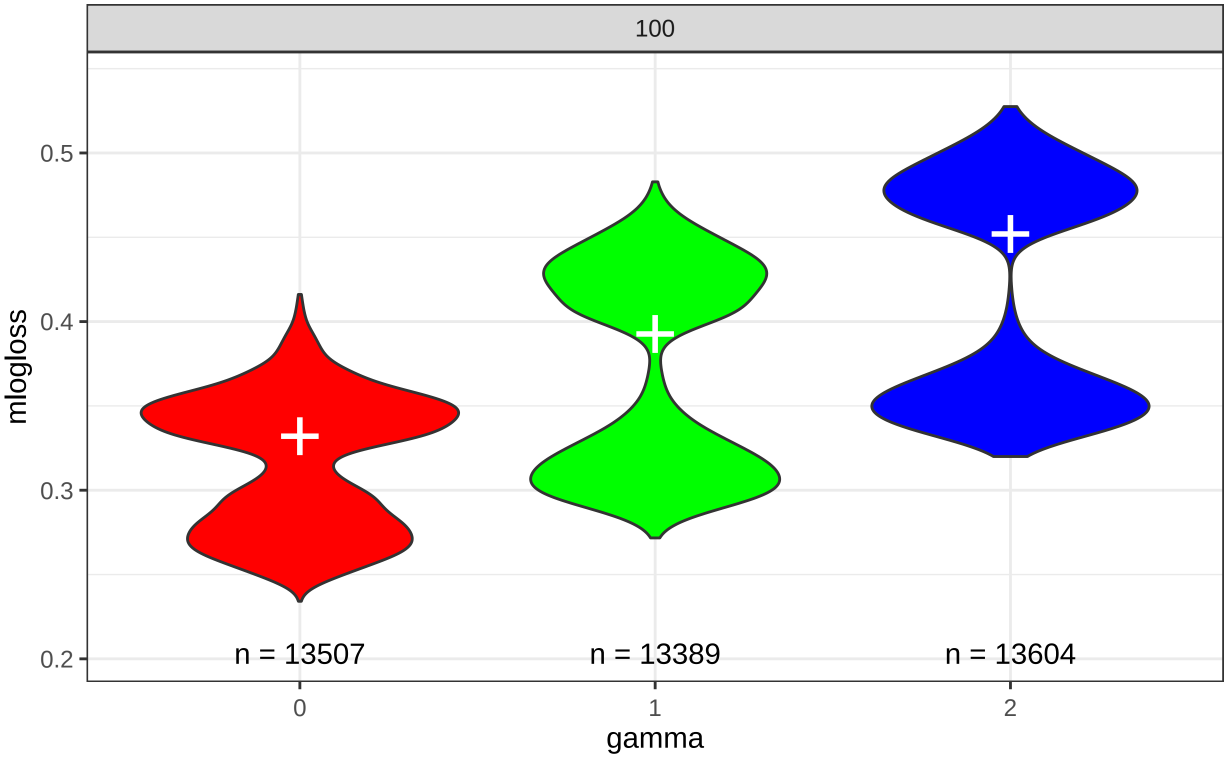
all grid search results

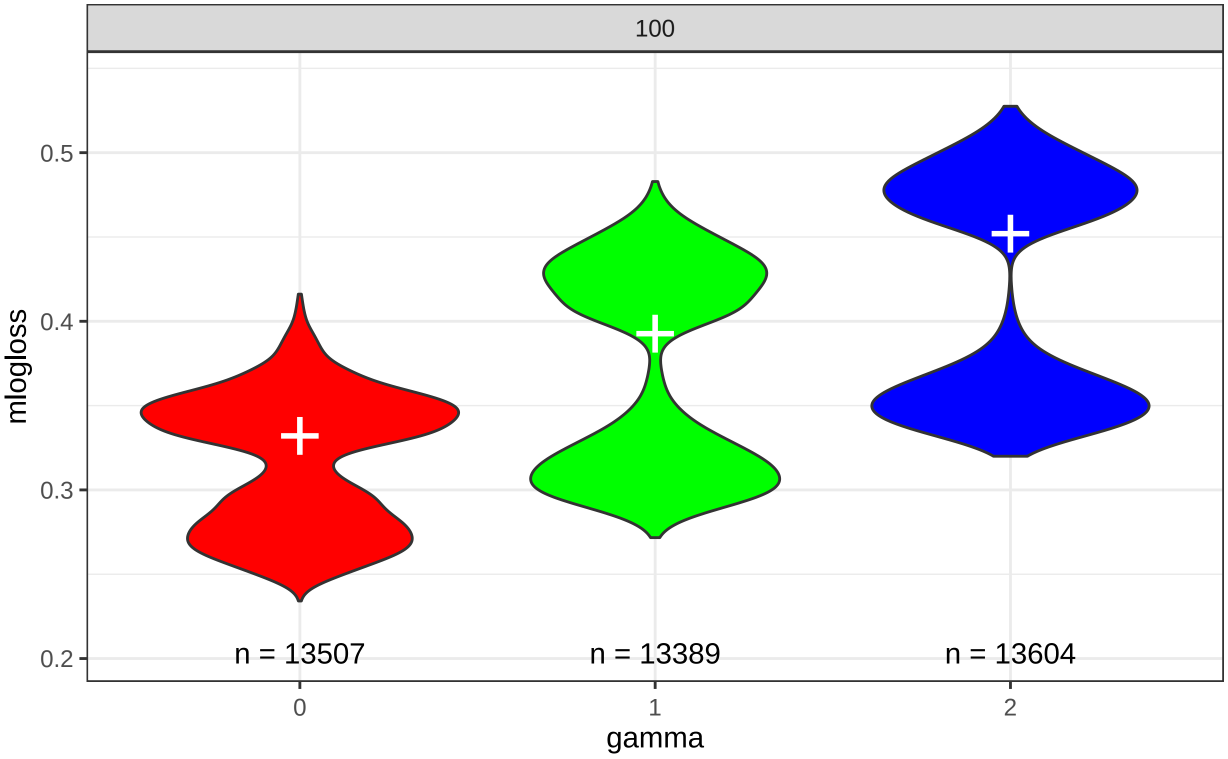


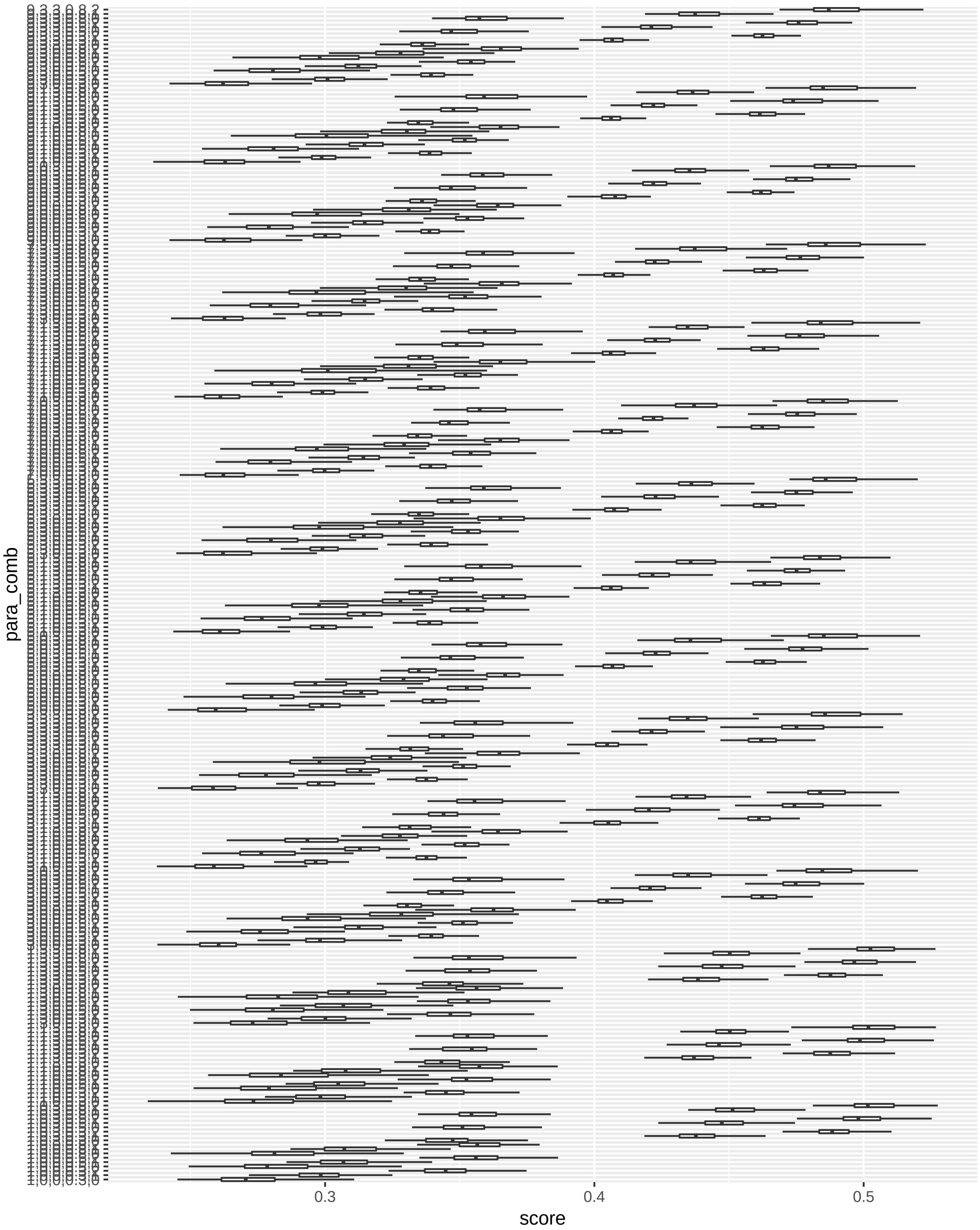



100
mlogloss
0.5
0.4
0.3
0.2
n = 20218
n = 20282
0
3
alpha


100
mlogloss
0.5
0.4
0.3
0.2
n = 13606
n = 13542
n = 13352
0.3
0.5
0.8
colsample_bytree

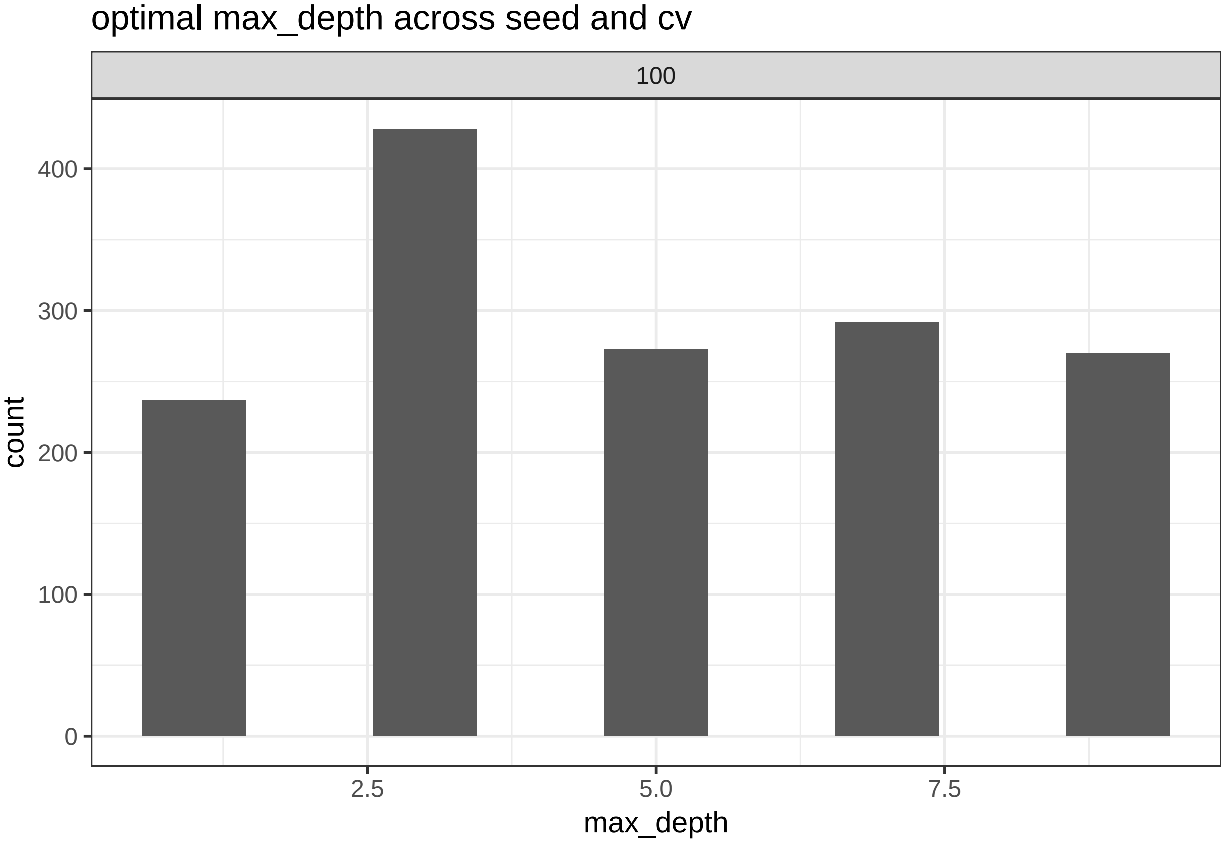

100
0.5
0.4
0.3
0.2
mlogloss
n = 13507
n = 13389
n = 13604
0
1
2
gamma


100
0.5
0.4
0.3
0.2
mlogloss
n = 13507
n = 13389
n = 13604
0
1
2
gamma

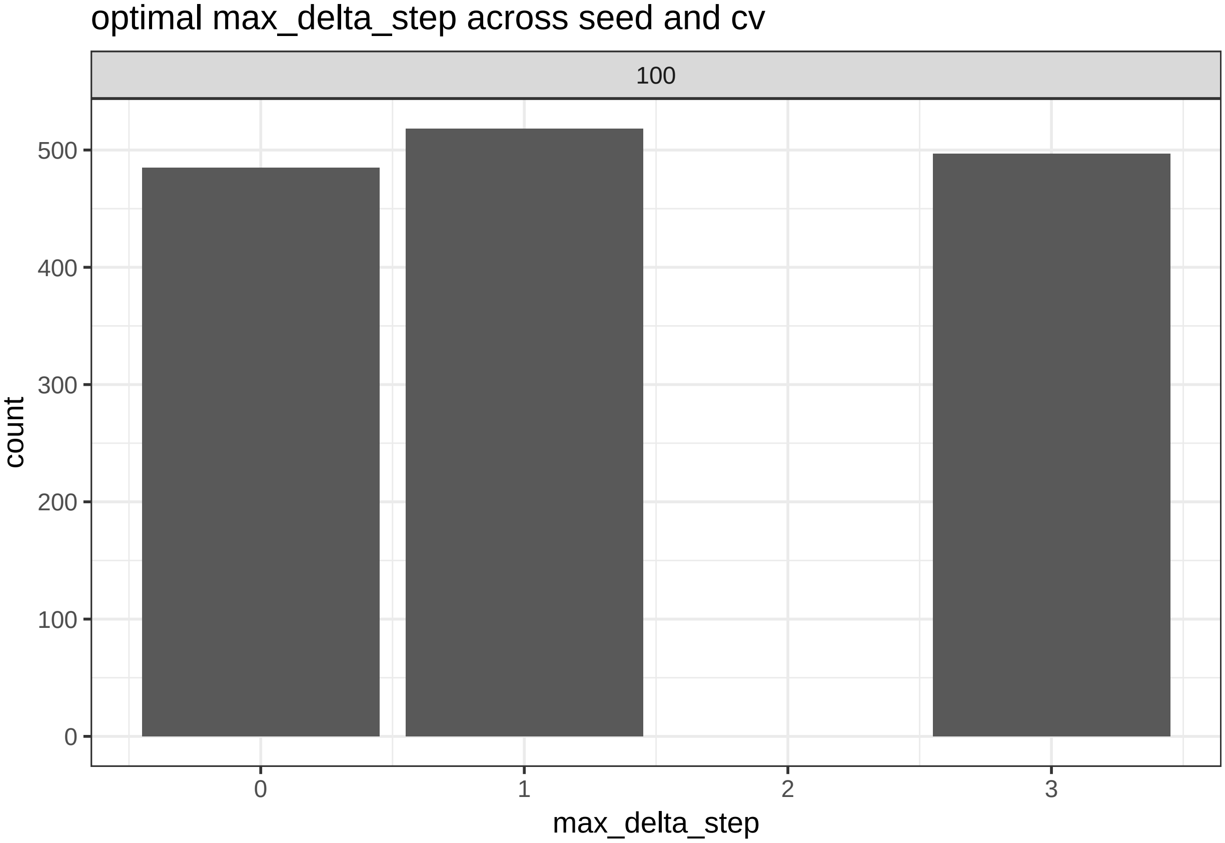


over best parameter combo per cv

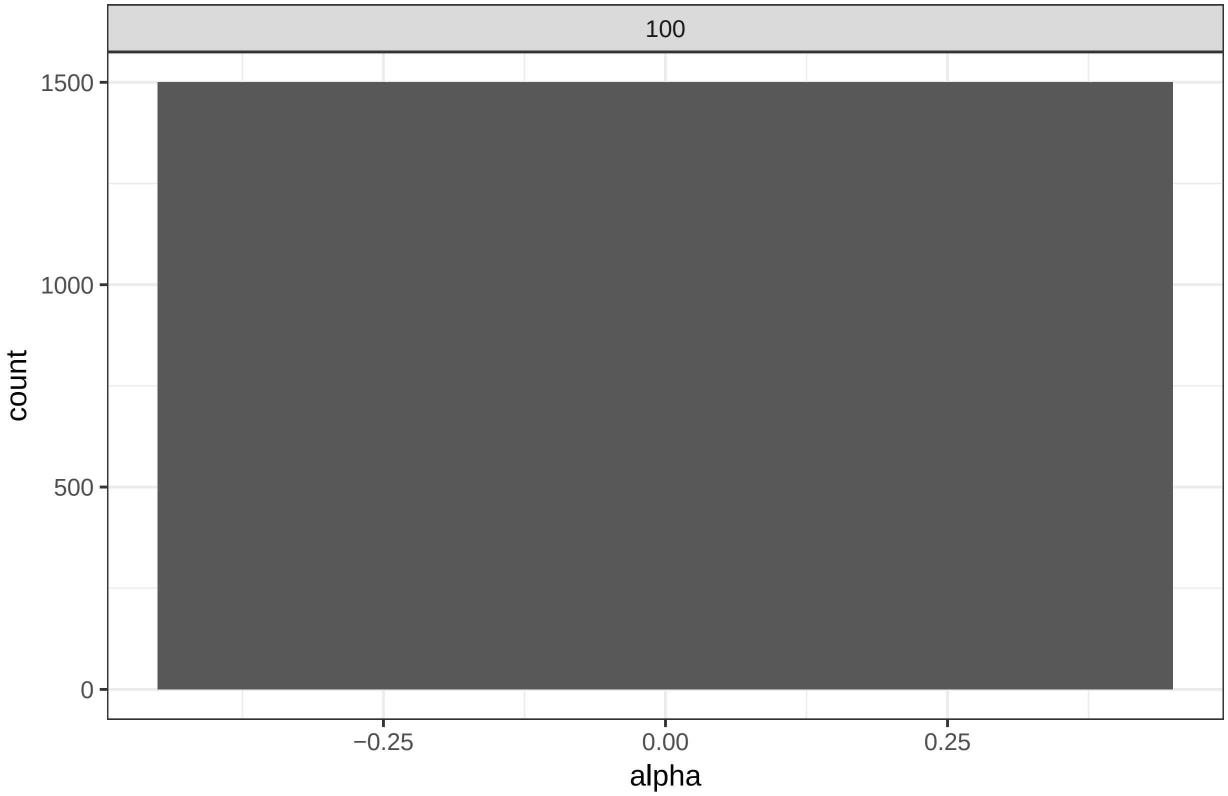
Note the 2nd /3rd best parameter combinations might not be too bad either.



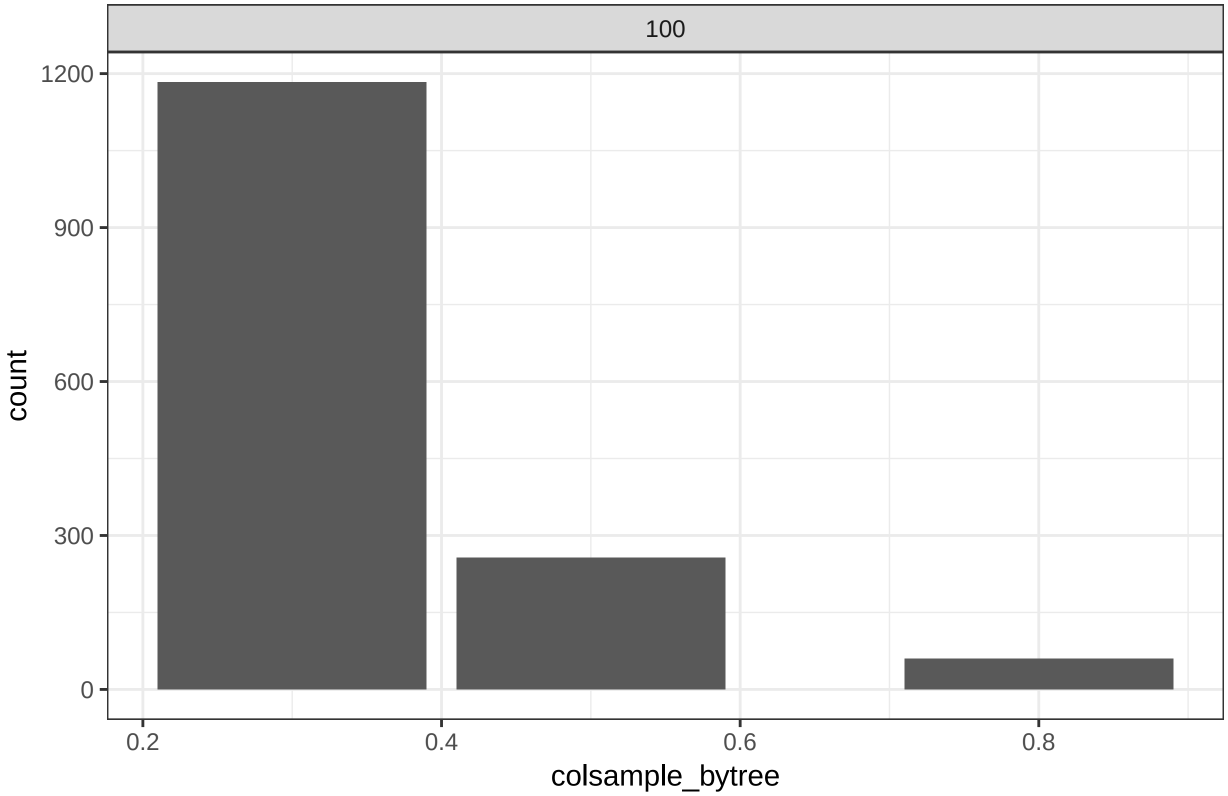
optimal max_depth across seed and cv

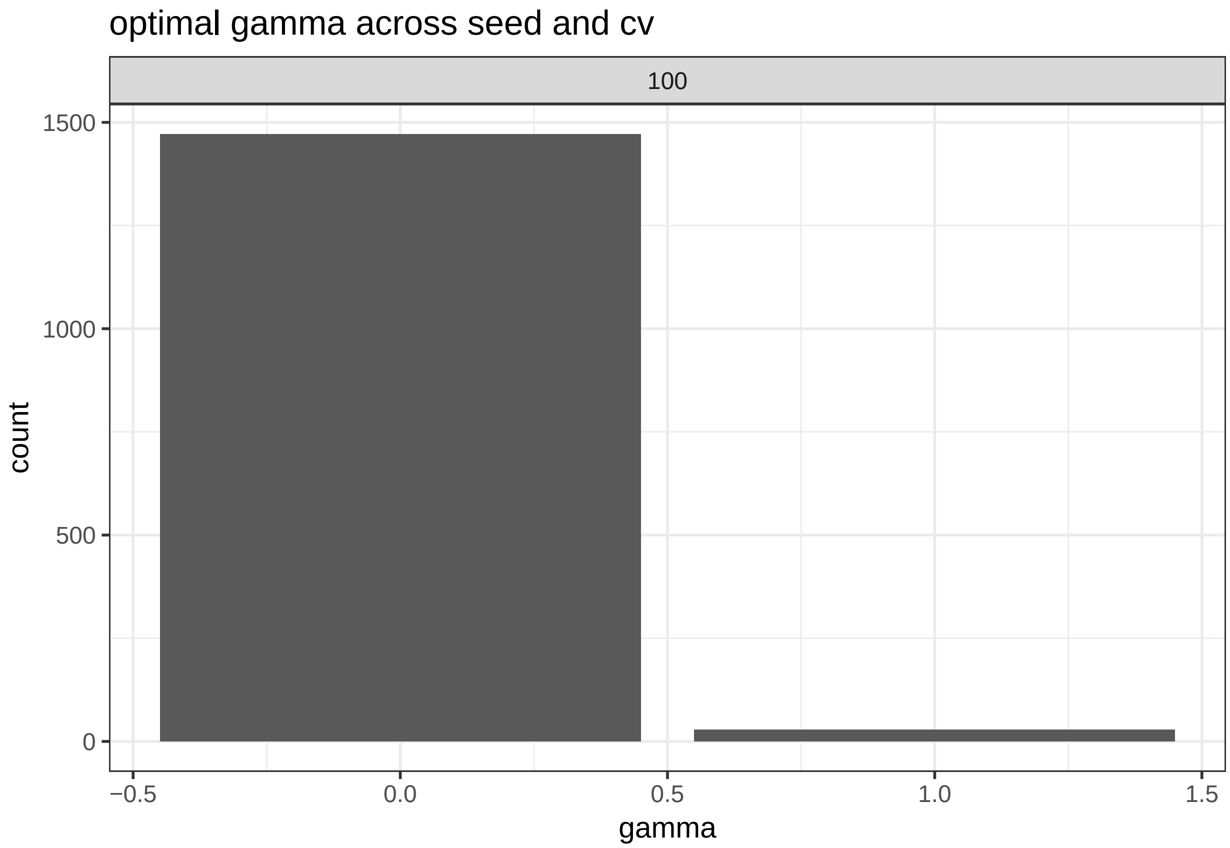


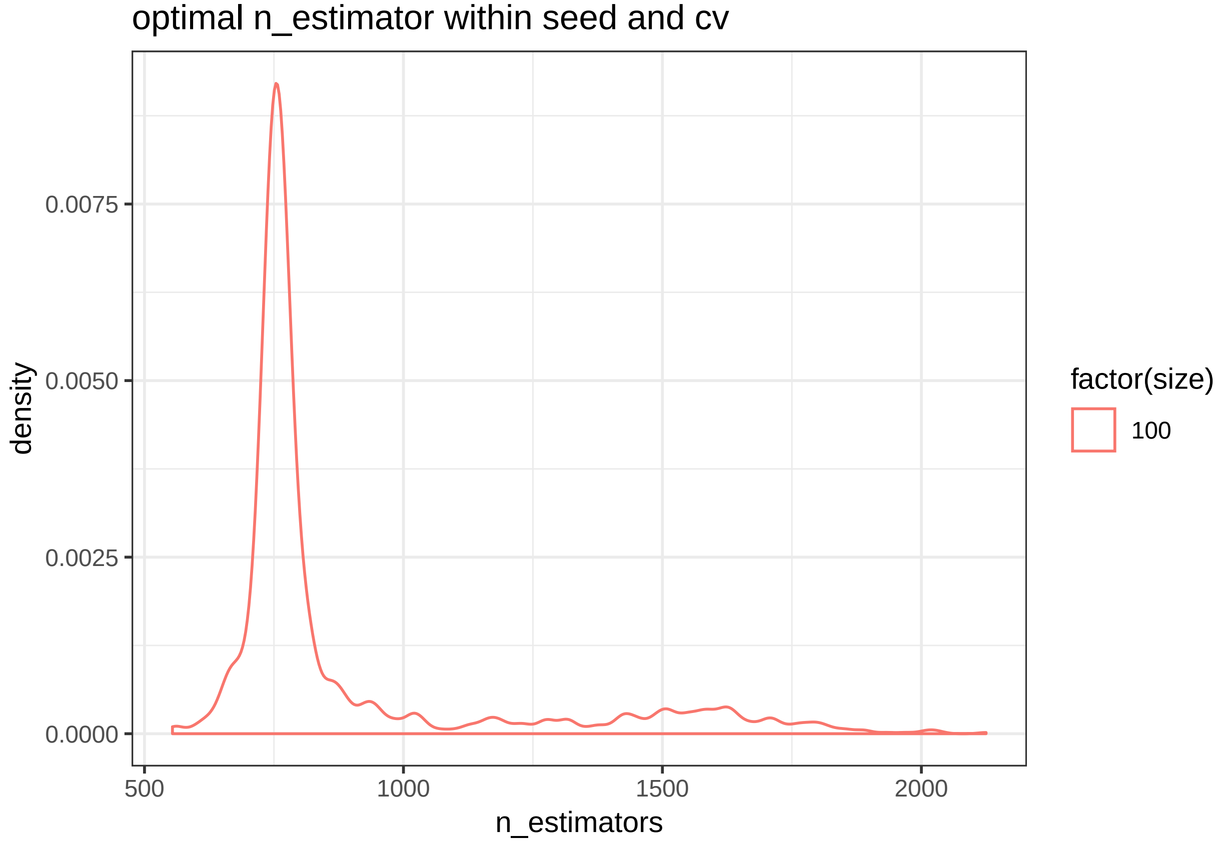
optimal max_delta_step across seed and cv optimal alpha across seed and cv



optimal colsample_bytree across seed and cv




optimal gamma across seed and cv
100
count
1500
1000
500
0
−0.5
0.0
0.5
1.0
1.5
gamma


optimal n_estimator within seed and cv
density
0.0075
0.0050
0.0025
0.0000
500
1000
1500
2000
n_estimators
factor(size)
100

more about the best parameter combination selection

```
select_ft_step <- 100

df1 <- subset(grid_best, size==select_ft_step & max_depth==1 &  max_delta_step == 0 )
print( paste('summary of n estimator at',select_ft_step, 'feature step'))
```

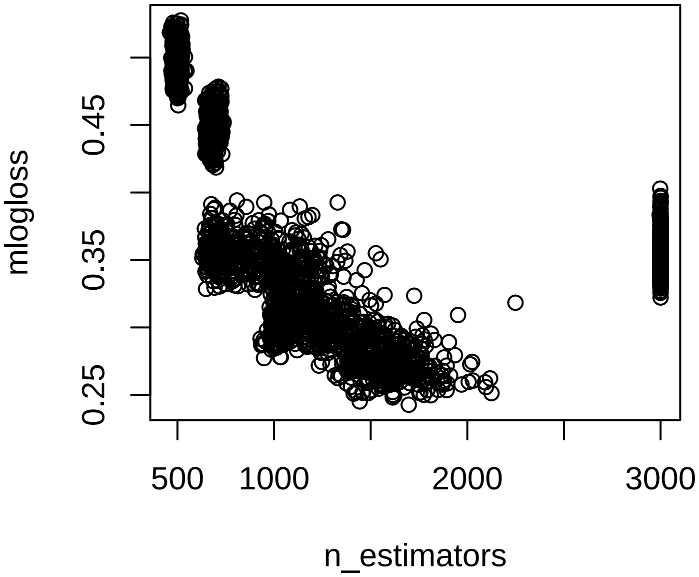
```
## [1] "summary of n estimator at 100 feature step"
```

```
print(summary(df1$n_estimators))
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1025    1353    1538    1510    1630    2125
```

```
df2 <- subset(df.grid, size==select_ft_step & max_depth==1 &  max_delta_step == 0 )

with(df2, plot(x = n_estimators, y=score, ylab=score_label))
```


mlogloss
n_estimators