I. PREDICTING NUMBER OF SATELLITES IN A HALO

In this section, we try to predict the number of satellites above a threshold mass, $m_{\text{above}} = 1.42 \times 10^{10}\ h^{-1}\ M_{\odot}$, a dark matter (DM) halo can have if we are given the halo properties. The strategy to do so is as follows:

1. Train a Random Forest (RF) Regression model on the halo data. The halo data contains halo properties, or *features*, and the number of satellites, or the *ground truth*.

2. Plot feature importances, and then select the top 5-7 features.

3. Plot the correlation matrix of the features, and then select the most important uncorrelated features.

4. Run the trained RF regression model on the entire halo data set to predict number of satellites per halo.

A. Data Preparation

We begin with separating out all satellites that are above m_{above}. We then identify the unique halos in this subset. The rest of the halos in the satellite data must have no satellites above m_{above}. In total, there are about 1.83 million unique halos which contain the satellites in the data. Training this huge dataset is computationally very expensive. We therefore must select a small sample of the halos and then train our machine learning model on it. Sampling the halos is not straightforward. The distribution of halos per mass bin isn't uniform (Figure 1). Therefore, we must stratify the sample based on mass distribution. We created a stratified sample from 10% of the halo data set. The rest 90% of the data forms the test set.

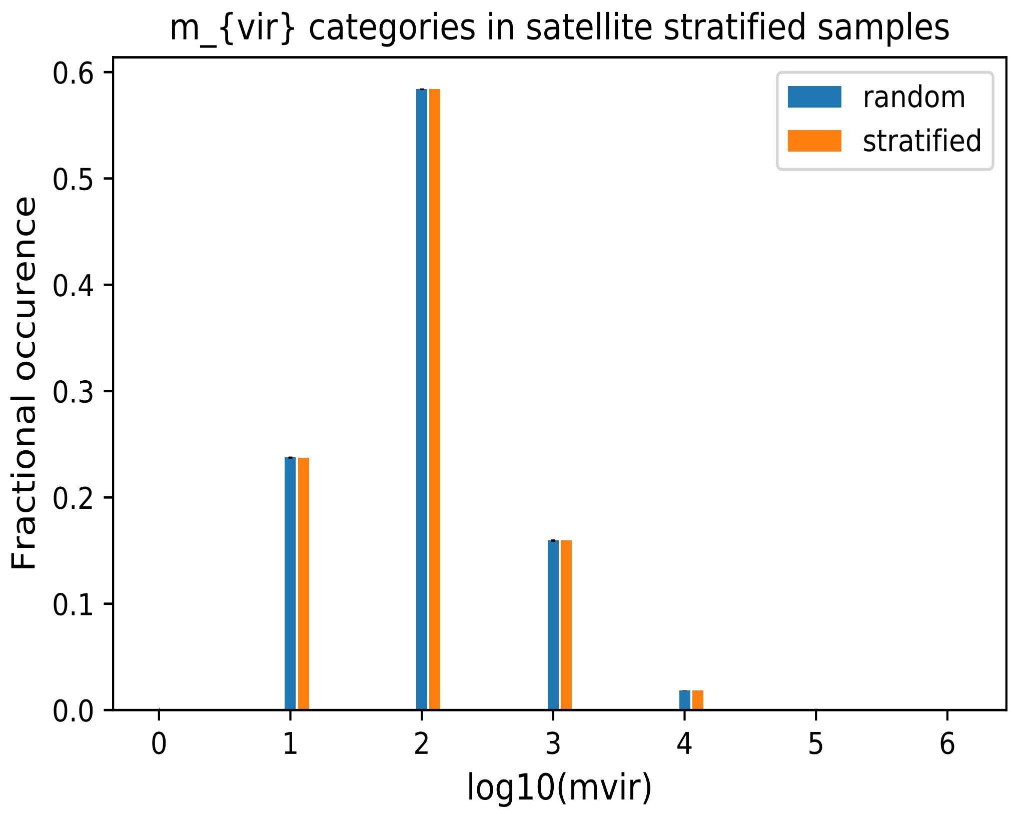


FIG. 1: Distribution of halo masses, $m_{\rm vir}$, in the halo data.

B. Random Forest Regression

We then train a RF regression model on the halo sample. The `RandomForestRegressor` module in scikit-learn has a hyper-parameter called `max_features`. We trained our sample with `max_features = log2` and `1.0`.

C. Feature Importances

Shown below are the bar plots denoting the feature importances of the RF regression model for the two hyper-parameters, `log2` and `1.0` (see Figure 2). We see that in both models, `mvir` is the most important feature. However, in `1.0`, `mvir` is more dominant as compared to other features. In what follows below, we evaluate how good the model `1.0` is.

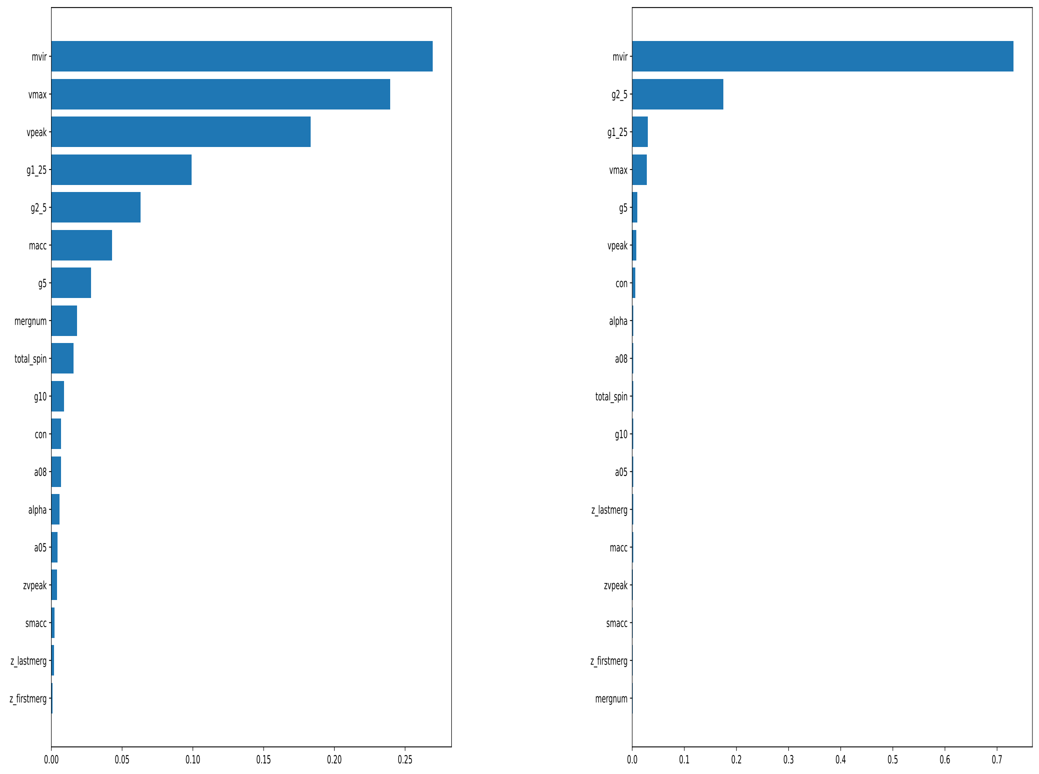


FIG. 2: Feature importances of the RF regression model for satellite galaxies. Left: `max_features = log2`, Right: `max_features = 1.0`. It appears that in `log2`, the feature importances are spread over all halo features, whereas in `1.0`, halo mass `mvir` is by far the most dominant feature followed by `g2_5, g5,` and `g1_25`.

We plot the top 7 feature correlation matrices for the two models in Figure 3. We see that in `log2`, the most important features such as `mvir, vmax` and `vpeak` are correlated to each other: `vmax-vpeak` ≈ 1.0, `vmax/vpeak-mvir` $= 0.6$. On the other hand, in `1.0`, the top 2 features, `mvir` and `g2_5`, aren't correlated to each other (`mvir-g2_5` = 0.25). The next feature `g5` is correlated to `g2_5`, `g2_5-g5` = 0.87. However, its importance is quite low in comparison to `g2_5`. It seems like the `1.0` model is doing much better than the `log2` model since it prioritizes uncorrelated features.

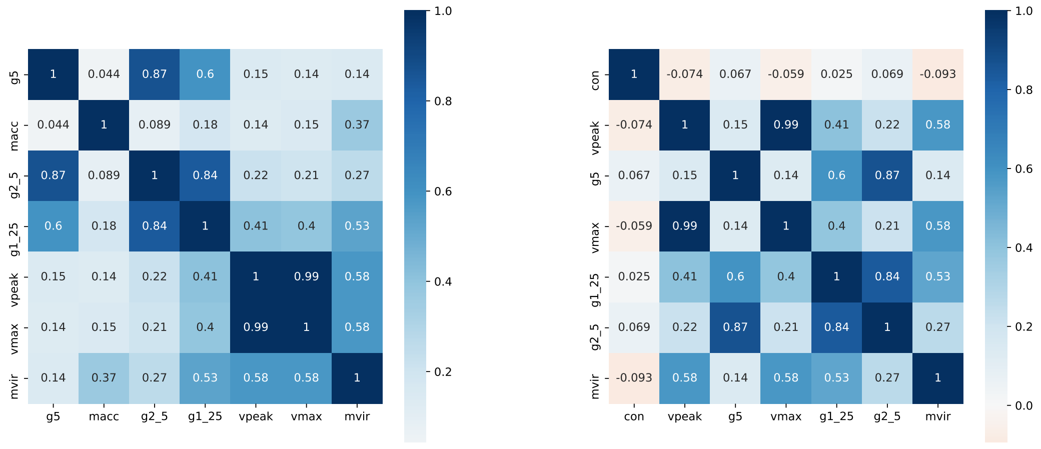


FIG. 3: Correlation matrix or heatmap of the top 7 features for the satellite data. Left: `max_features = log2`, Right: `max_features = 1.0`. The closer the values are to ±1, the more correlated/anti-correlated the features are.

We next compare the performance measure, r^2 score, from the two models. For this step, we select the most important, uncorrelated features from both models. For `log2`, the features are `mvir, vmax, g1_25, g2_5, macc`, and for `1.0`: `mvir, g2_5, g1_25, con`. We then perform 5-fold cross-validation test on both models. After training the models, we predict the satellite numbers in the test set, followed by the entire halo data. The results are as follows, Table III:

We see that the model `1.0` does slightly better than the `log2` model. This leads to the following conclusion: the features `mvir, g2_5, g1_5, con` are good predictors of the number of satellites a halo can occupy.

Model	5-fold CV	Test	Total
`log2`	0.916 ± 0.016	0.875	0.881
`1.0`	0.922 ± 0.014	0.878	0.885
`Gradient Boost`	0.916 ± 0.017	0.887	0.895

TABLE I: Model comparison using r^2 score using both halo and environment properties.

Model	5-fold CV	Test	Total
`log2`	0.846 ± 0.047	0.866	0.867
`1.0`	0.849 ± 0.038	0.867	0.868
`Gradient Boost`	0.842 ± 0.033	0.853	0.860

TABLE II: Model comparison using r^2 score using only halo properties.

II. PREDICTING CENTRAL GALAXY OCCUPATION

The purpose of this section is to predict the occupation of the halo by a central galaxy of mass above m_{above}. The strategy for doing so is nearly the same as that in the previous section for the case of satellites. The only difference is that central occupancy prediction is a classification problem.

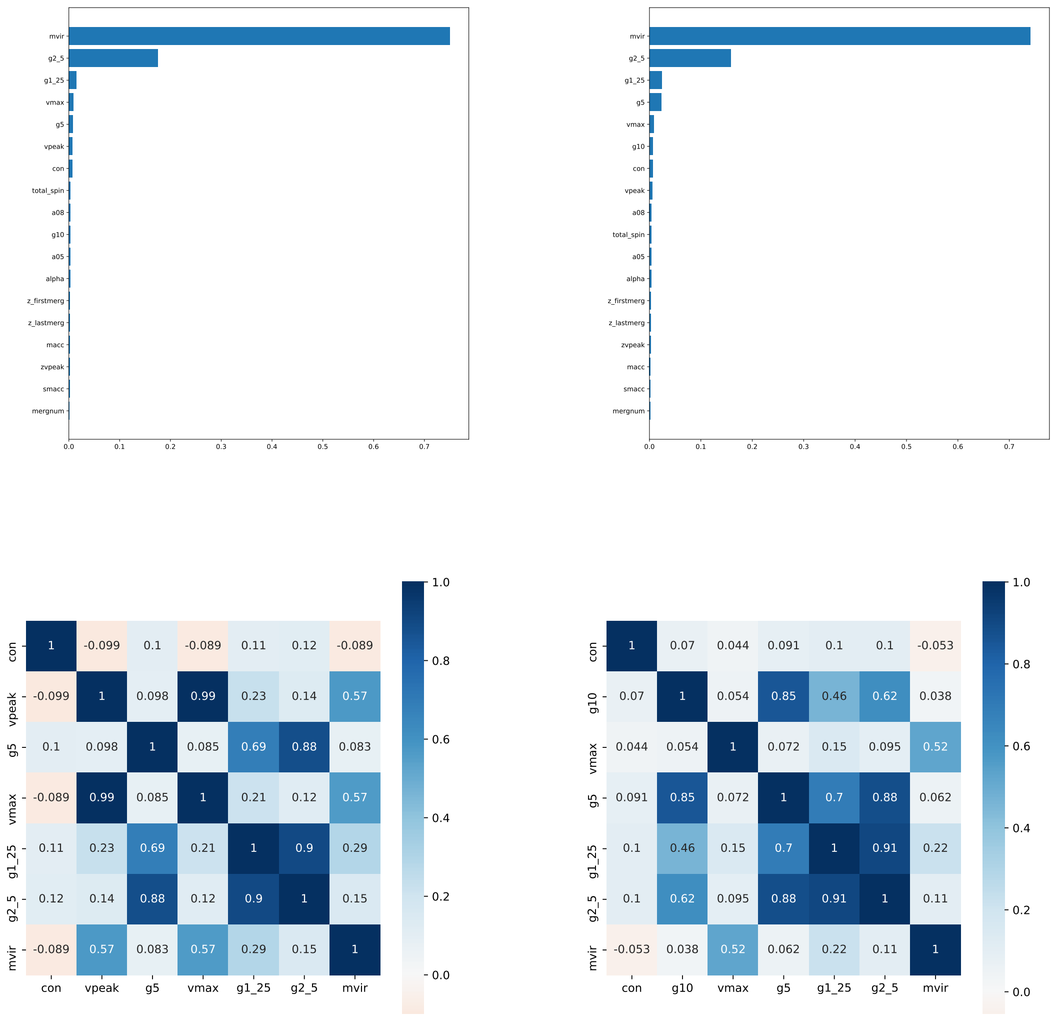


FIG. 4: Feature importances (top) and heatmap (bottom) from random forest. Left: with cut, Right: no cut.

A. Data Preparation

In total, the central galaxy dataset contains 8.76 million unique halos. This means that instead of using the entire dataset to train a classification model, we must create a sample from the original dataset for training. The distribution of halos per `vmax` bin isn't uniform (Figure 21). Therefore, we must stratify the sample based on `vmax` distribution. We created a stratified sample from 10% of the central data set. The rest of the 90% of the central data forms the test set.

B. RF classifier

We trained a RF classifier model on the halo sample with `max_features = log2` and `1.0`.

C. Feature Importances

Below are bar plots denoting the feature importances of the RF classifier model (Figure 22) for `max_features = log2` and `1.0`. From both models, we find that `vmax` is the most important feature. However, the `1.0` model gives `vmax` much greater importance that the `log2` model.

We then select the top 7 features from the above bar plots, and plot the heatmaps for both models (Figure 23).

Again, comparing both heatmaps, we find that the `1.0` model gives more importance to uncorrelated features. We then train our samples with both models keeping only the most important uncorrelated features. For `log2`, we select `vmax, mvir, total_spin, z_firstmerg, macc`, and for `1.0`, we select `vmax, z_lastmerg, a05, total_spin`. We plot the confusion matrices from the validation set (25% instances from the stratified sample) from both models below (Figure 25):

Finally, we compare the various F_1 scores from both models as shown in the table below, Table IV

We see that, the `1.0` model performs equally well as the `log2` model.

Model	5-fold CV	Test	Total
`log2`	0.904 ± 0.005	0.904	0.905
`1.0`	0.905 ± 0.002	0.905	0.905

TABLE III: Model comparison using F_1 score.

Model↓ Features→	Top 4/5	Top 4/5 + env	All
`log2`	0.903/0.916/0.898	0.905/0.919/0.899	0.916/0.930/0.912
`1.0`	0.906/0.920/0.901	0.906/0.919/0.901	0.919/0.929/0.916

TABLE IV: Model comparisons using `max_features` and number of features. The first two scores show F1 and F2 scores. The third score is the F1 = F2 score when conserving the central fraction.

III. PREDICTING BOTH CENTRAL AND SATELLITES

In this section, we predict the total number of galaxies, central and satellites, in a given halo. We can do this in 2 different ways: first, we can combine the predictions of the regressor and classifier as described above. Second, we can run a regressor predicting both central and satellites.

We can compare both methods by plotting the number of galaxies per halo mass for each individual halo. This is shown in Figure 31. A key feature which stands out for both models is that they fail spectacularly at high halo mass regimes, $m_{\text{halo}} > 10^4$. This seems to be a result of low number of halo samples at higher halo masses.

We then plot the galaxy occupation per total number of halos in a given mass bin or the halo occupation function as a function of halo mass, Figure 33. At low masses,

$\log_{10}(m_{\text{halo}}) < 4$, the first method does better. This is so because the RF regressor does a poor job predicting small number of galaxies in a given halo.

Appendix A: R^2 score or coefficient of determination

Let $\bar{y}$ denote the mean of the ground truth values. The total sum of squares is given as

$$SS_{tot} = \sum_i (y_i - \bar{y})^2. \tag{A1}$$

The regression/explained sum of squares is given as

$$SS_{reg} = \sum_i (p_i - \bar{y})^2 \tag{A2}$$

where p_i are the predicted values. The residual sum of squares is given as

$$SS_{res} = \sum_i (p_i - y_i)^2 = \sum_i e_i^2. \tag{A3}$$

The R^2 is given as

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}. \tag{A4}$$

The best model where $p_i = y_i$ will give an R^2 value of 1 whereas the worst model will have an R^2 value of $-\infty$.

Appendix B: Heatmap scores

The scores given in the heatmap cells are the Pearson's r given as

$$r_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}. \tag{B1}$$

Therefore the r score can only take values between -1 and 1.

Appendix C: Decision Tree: Regression

Below is an example of a decision tree for a regressor. How does a decision tree make decisions? The CART (Classification and Regression Tree) algorithm to train decision trees follows the following prescription:

1. When `max_features = 1`, at each node the tree has all features available from which it picks the best feature to split the data.

2. The best feature, k, and the threshold, t_k, is selected by minimizing the loss function $J(k, t_k)$. Eg., $k =$ `mvir`, $t_k = 2400.07$ in the first daughter node on the left. The loss function is given by

$$J(k, t_k) = \frac{m_{left}}{m} MSE_{left} + \frac{m_{right}}{m} MSE_{right} \tag{C1}$$

where

$$MSE_{node} = \sum_i (p_{node} - y_i)^2 \tag{C2}$$
$$p_{node} = \frac{1}{m_{node}} \sum_i y_i$$

3. In each node, `mse` denotes the mean square error, `samples` denotes the number of samples in the node/leaf, and `value` denotes the predicted value of those samples which is equal to the mean value of the ground truth of the samples y_i.

Appendix D: Decision Tree: Classification

Below is an example of a decision tree for a classifier. The algorithm for making decisions is the same as that of the regressor. The only difference is that instead of minimizing the mean square error, the classifier minimizes either the gini impurity (default) or the entropy which are defined as

$$\text{Gini Impurity: } G = 1 - \sum_{k=1}^{n} p_k^2, \tag{D1}$$

$$\text{Entropy: } H = - \sum_{k=1,\ p_k \neq 0}^{n} p_k \ log_2(p_k) \tag{D2}$$

where p_k is the fraction of the instances belonging class k in the node. For example, in the first daughter node on the left, $p_0 = 90816/91244 = 0.995$, and $p_1 = 0.005$ giving us $G = 0.009$. In the nodes/leaves in the picture, `value` denotes the number of instances in the `samples` in that node which belong to class 0 and 1.

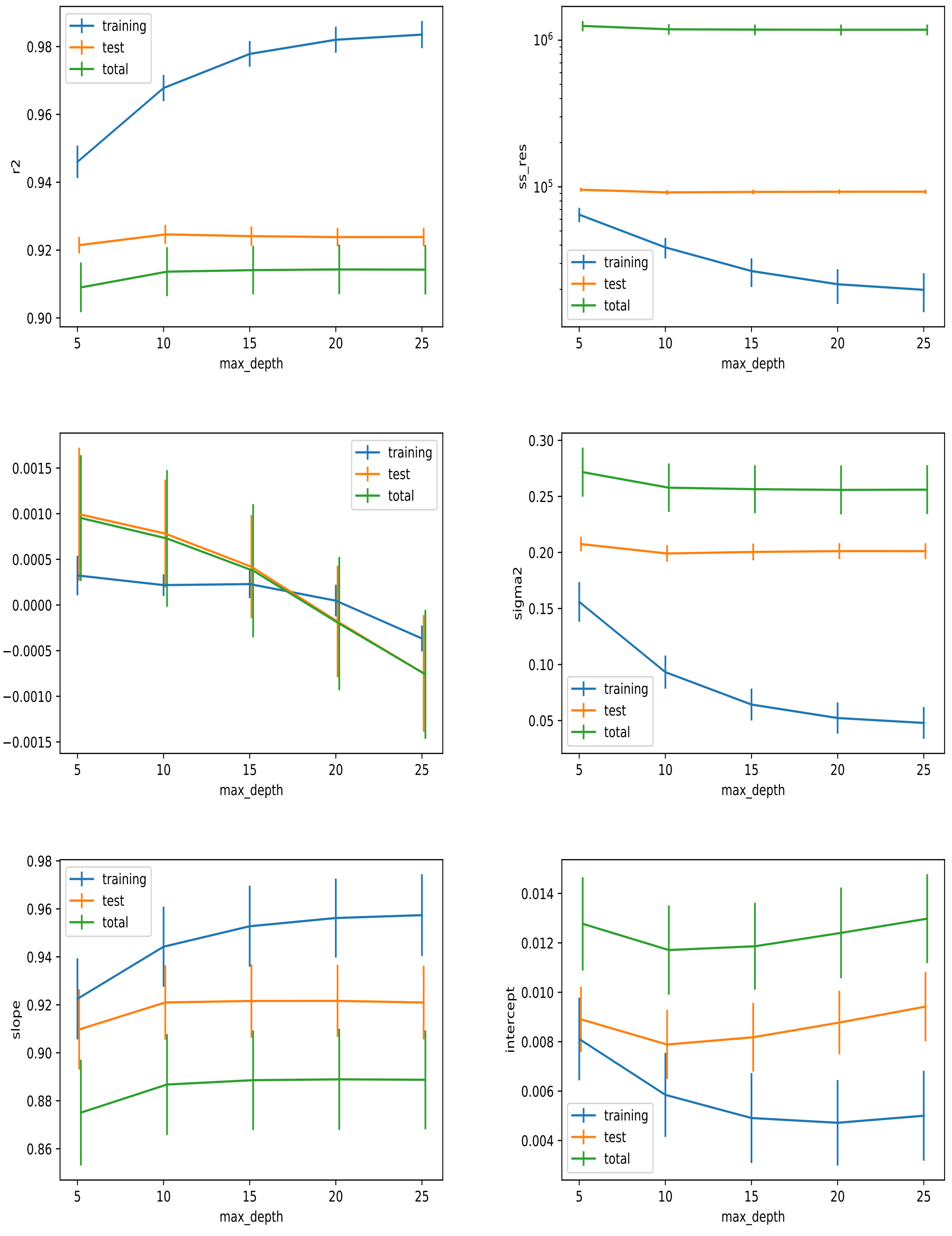


FIG. 5: Learning curve for `max_depth`: `n_estimators = 300`, `min_samples_leaf = 1`. Sample size = 0.1, Features = top.

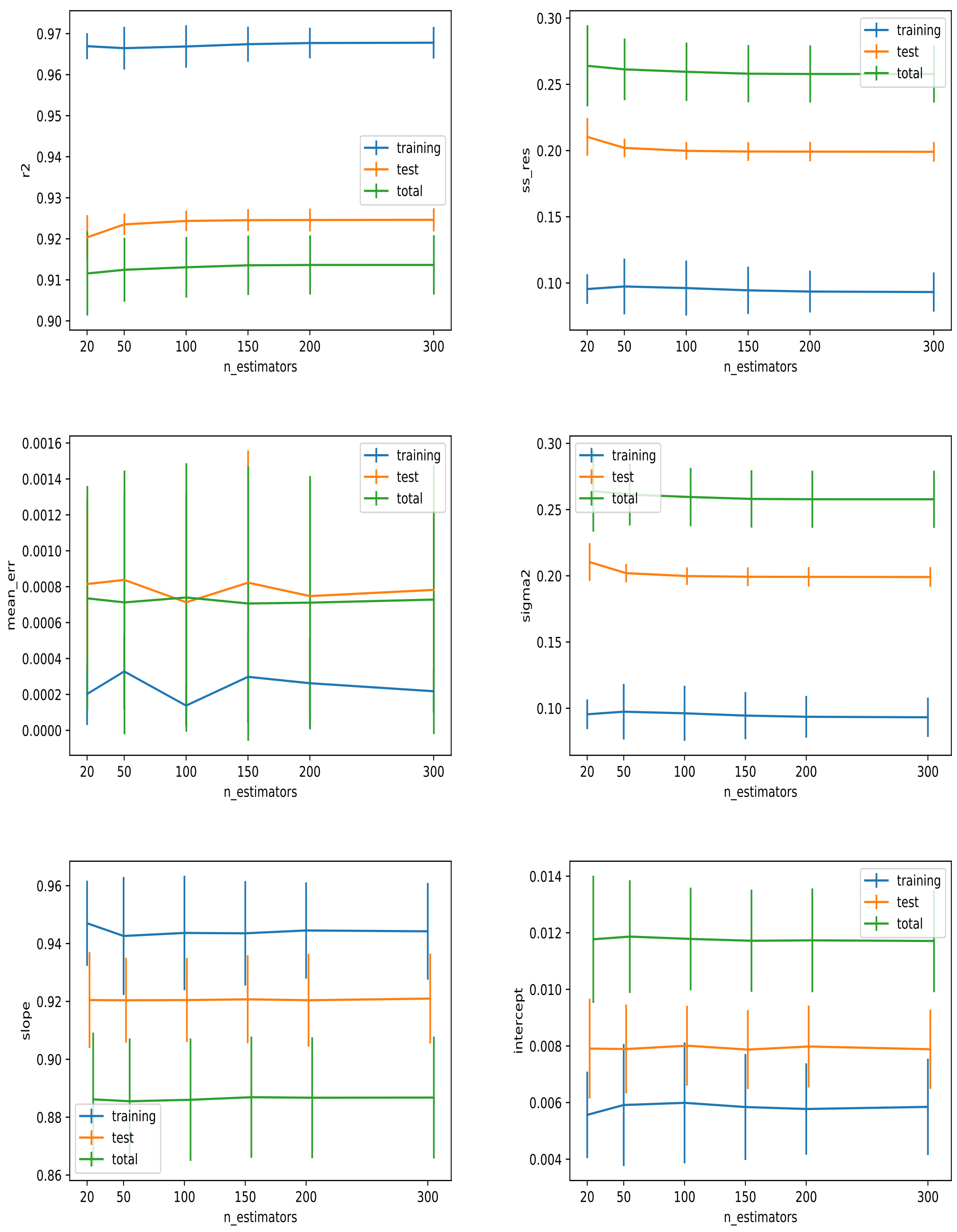


FIG. 6: Learning curve for `n_estimators`: `max_depth = 10`, `min_samples_leaf = 1`. Sample size = 0.1, Features = top.

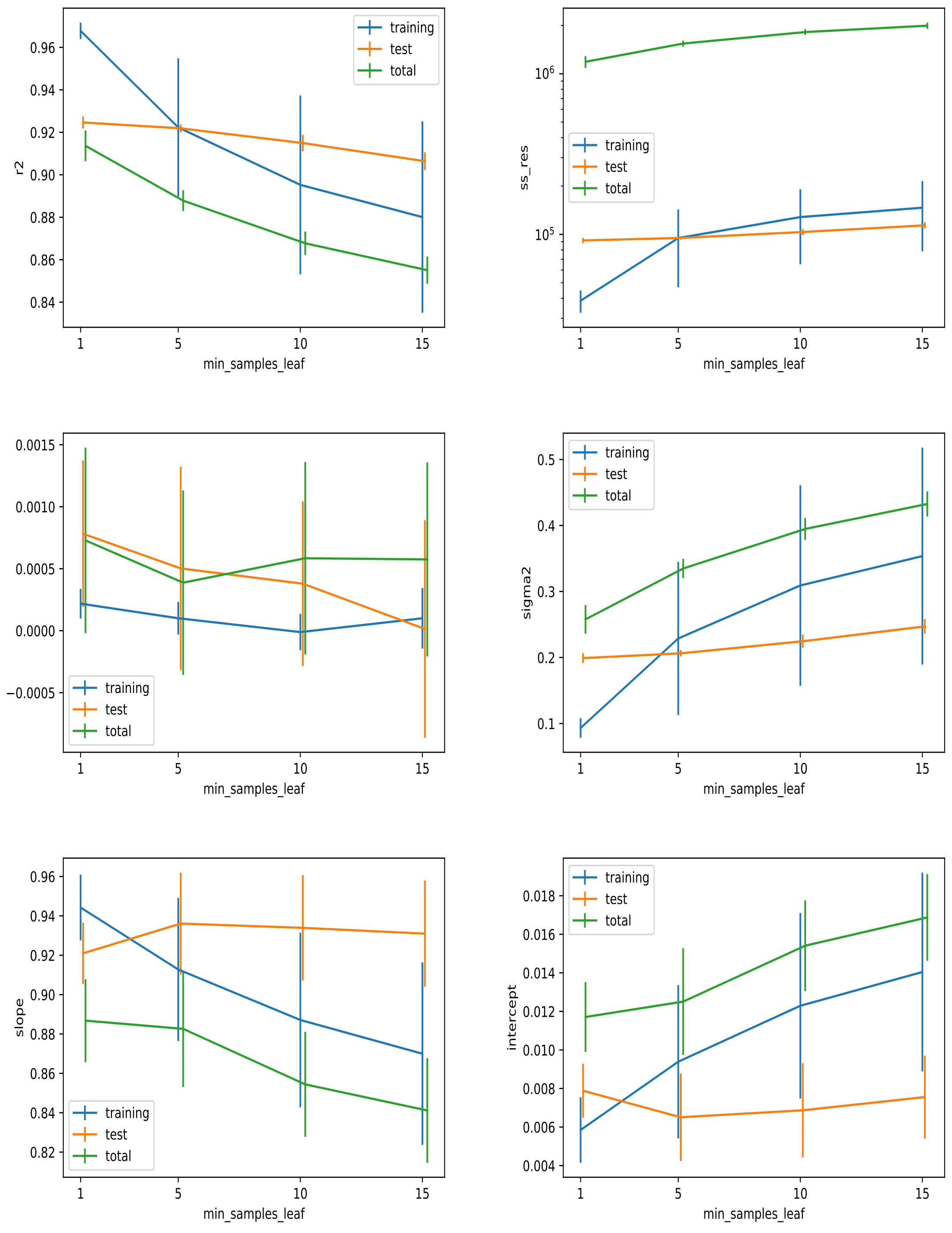


FIG. 7: Learning curve for `min_samples_leaf`: `max_depth = 10`, `n_estimators = 300`. Sample size = 0.1, Features = top.

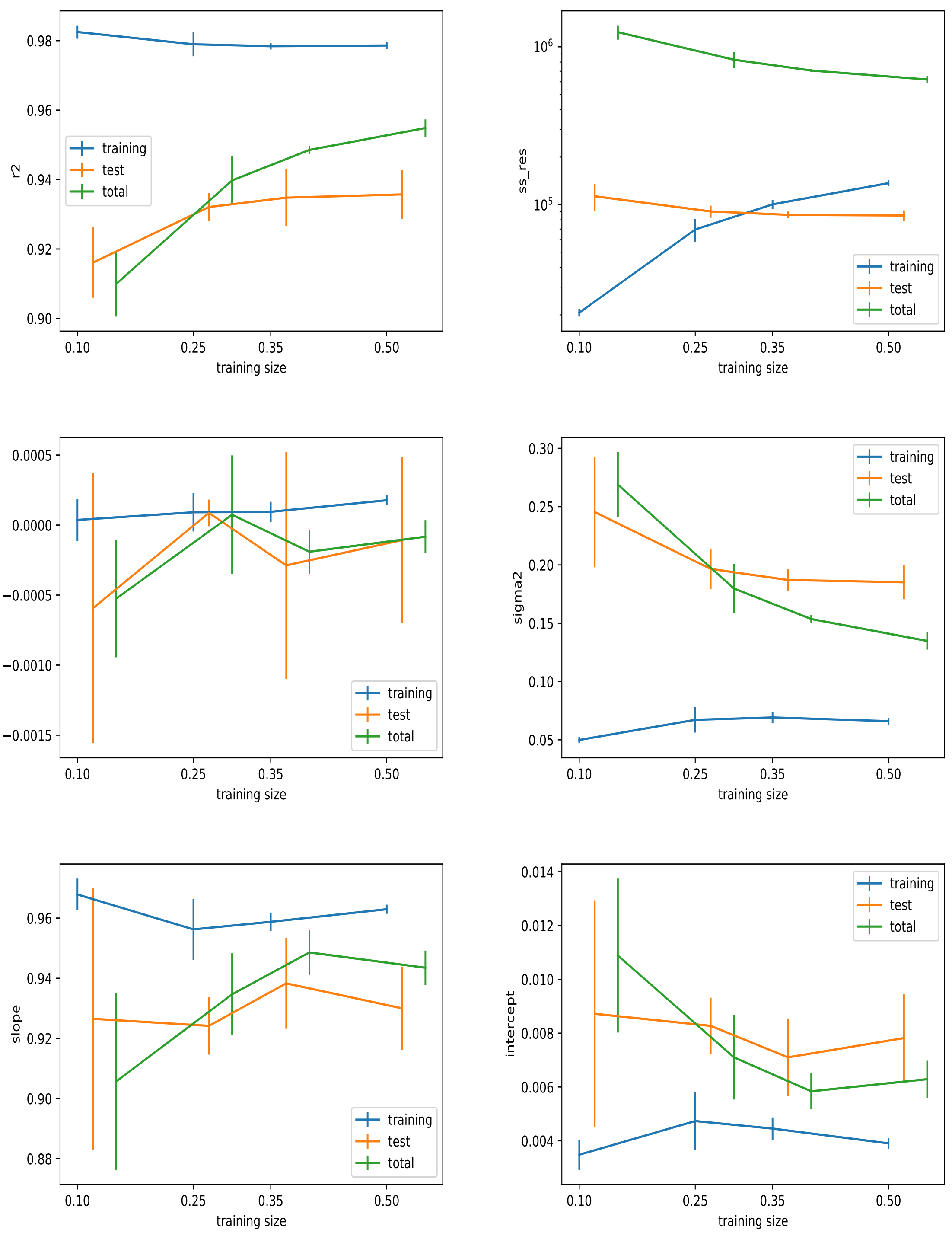


FIG. 8: Learning curve for `training size`: `min_samples_leaf = 1, max_depth = 15, n_estimators = 300.` Sample size = 0.1, Features = all.

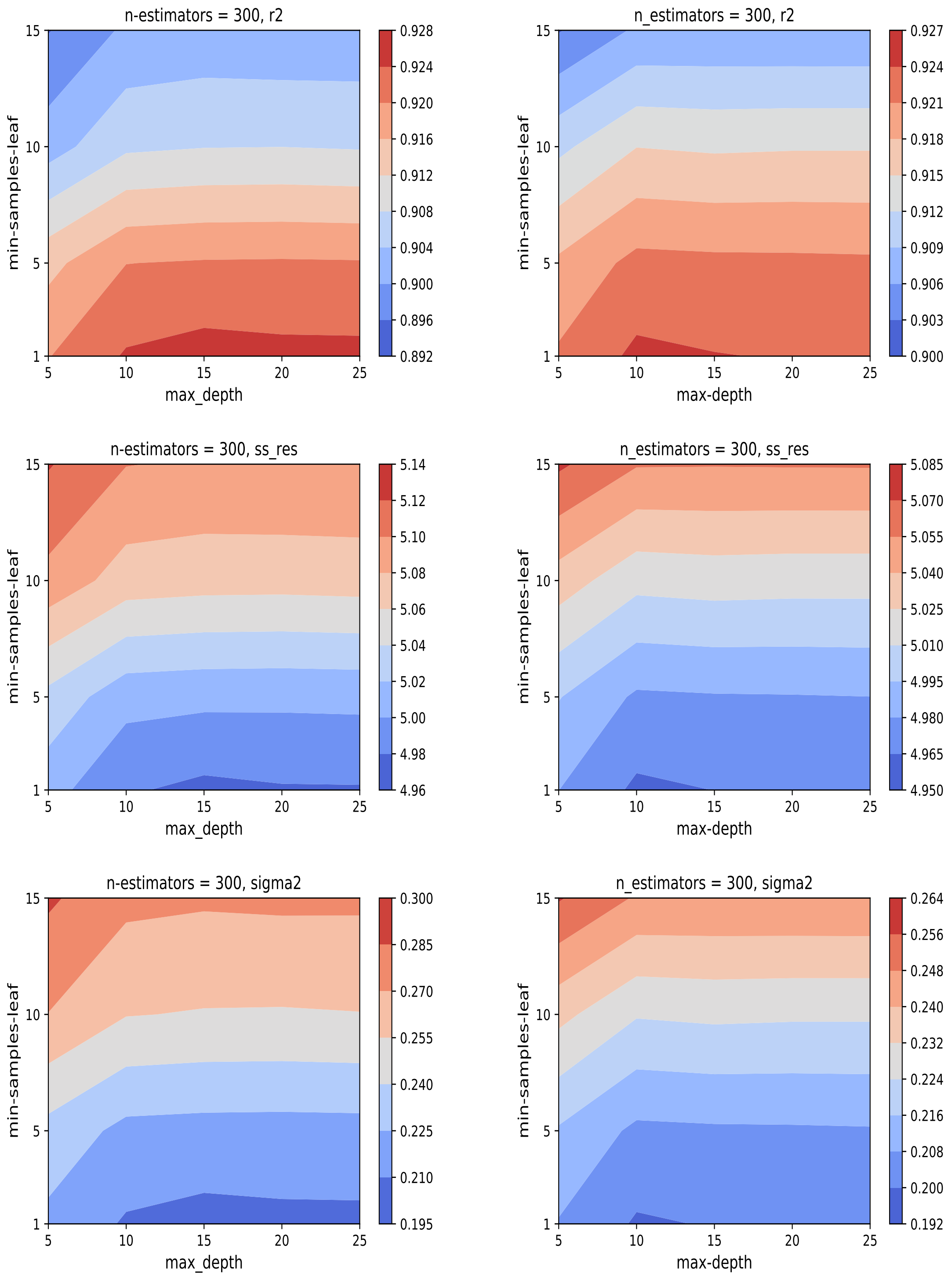


FIG. 9: Test scores after grid search for 10% of satellite data used for training after mass cut. Test size = 10%. Left: All features, Right: Top features, `n_estimators = 300`.

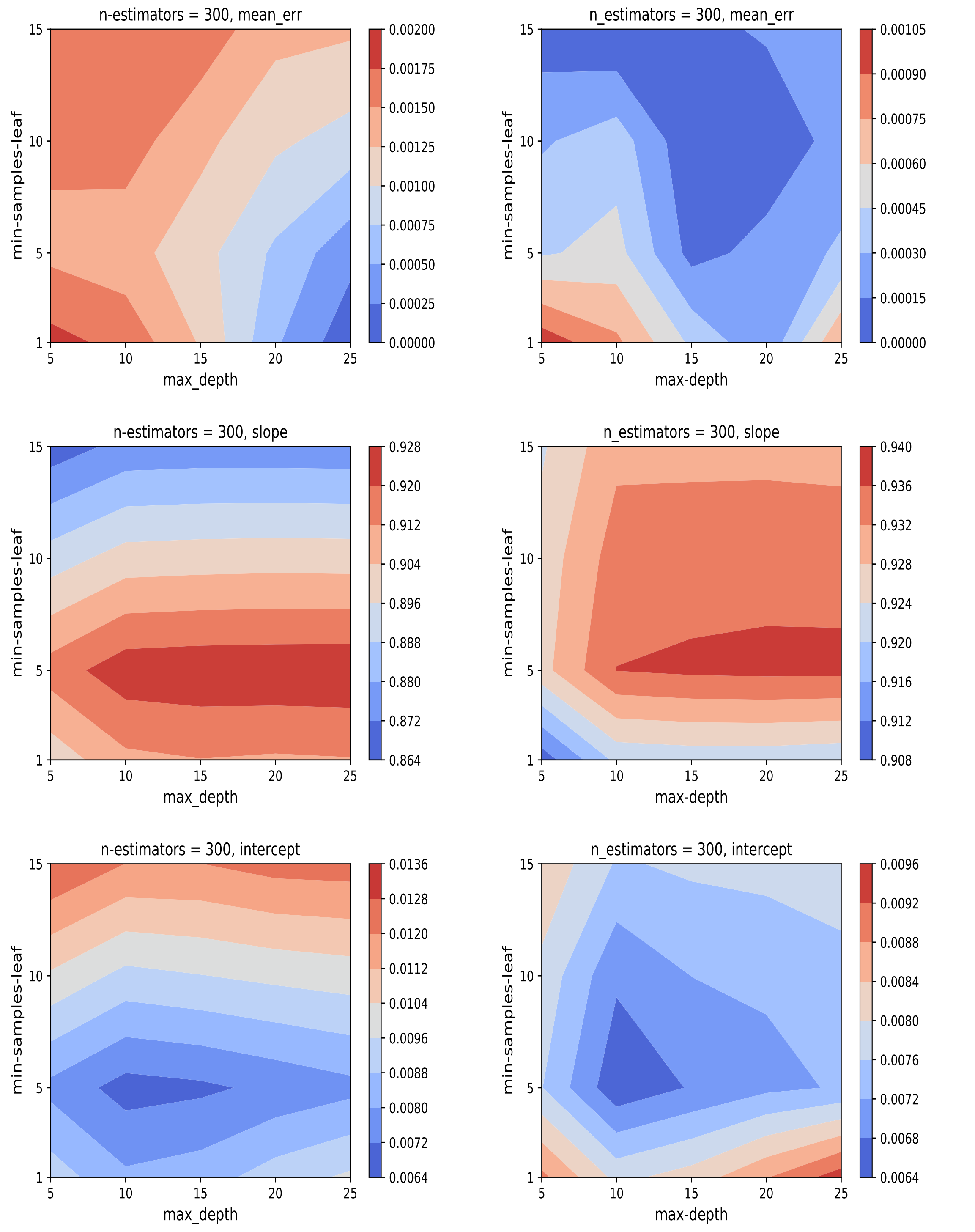


FIG. 10: Test scores after grid search for 10% of satellite data used for training after mass cut. Test size = 10%. Left: All features, Right: Top features, `n_estimators = 300`.

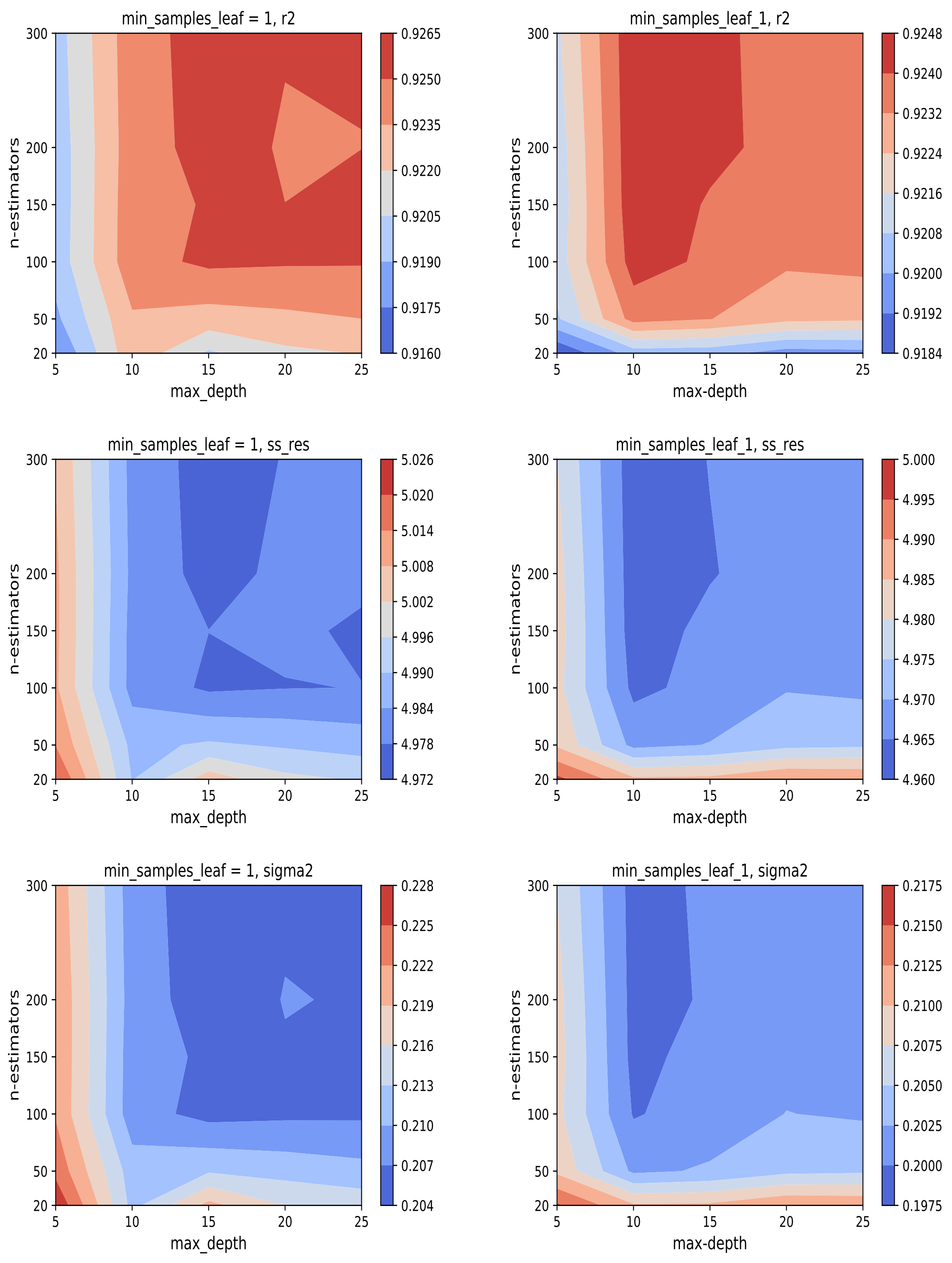


FIG. 11: Test scores after grid search for 10% of satellite data used for training after mass cut. Test size = 10%. Left: All features, Right: Top features, `min_samples_leaf = 1`.

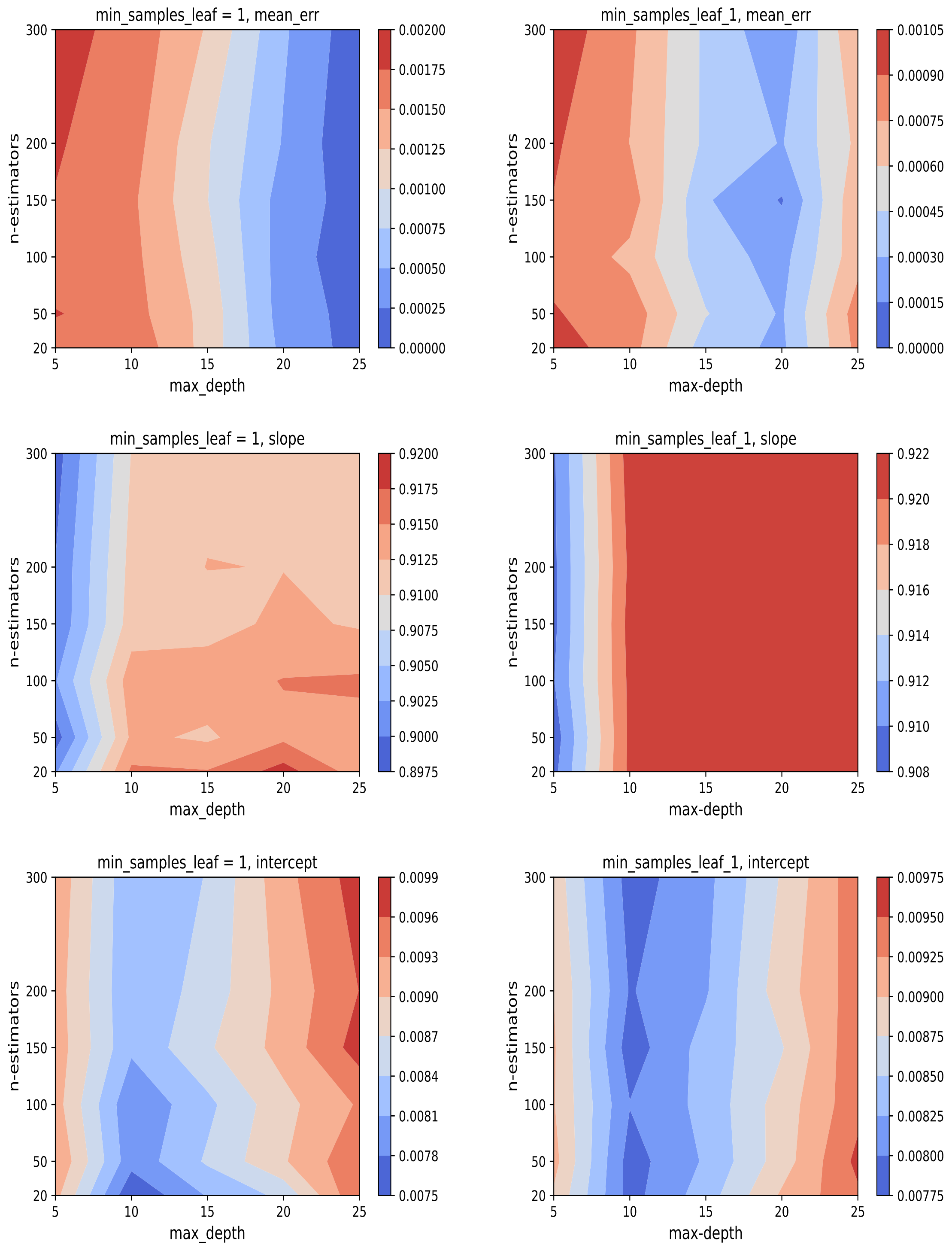


FIG. 12: Test scores after grid search for 10% of satellite data used for training after mass cut. Test size = 10%. Left: All features, Right: Top features, `min_samples_leaf = 1`.

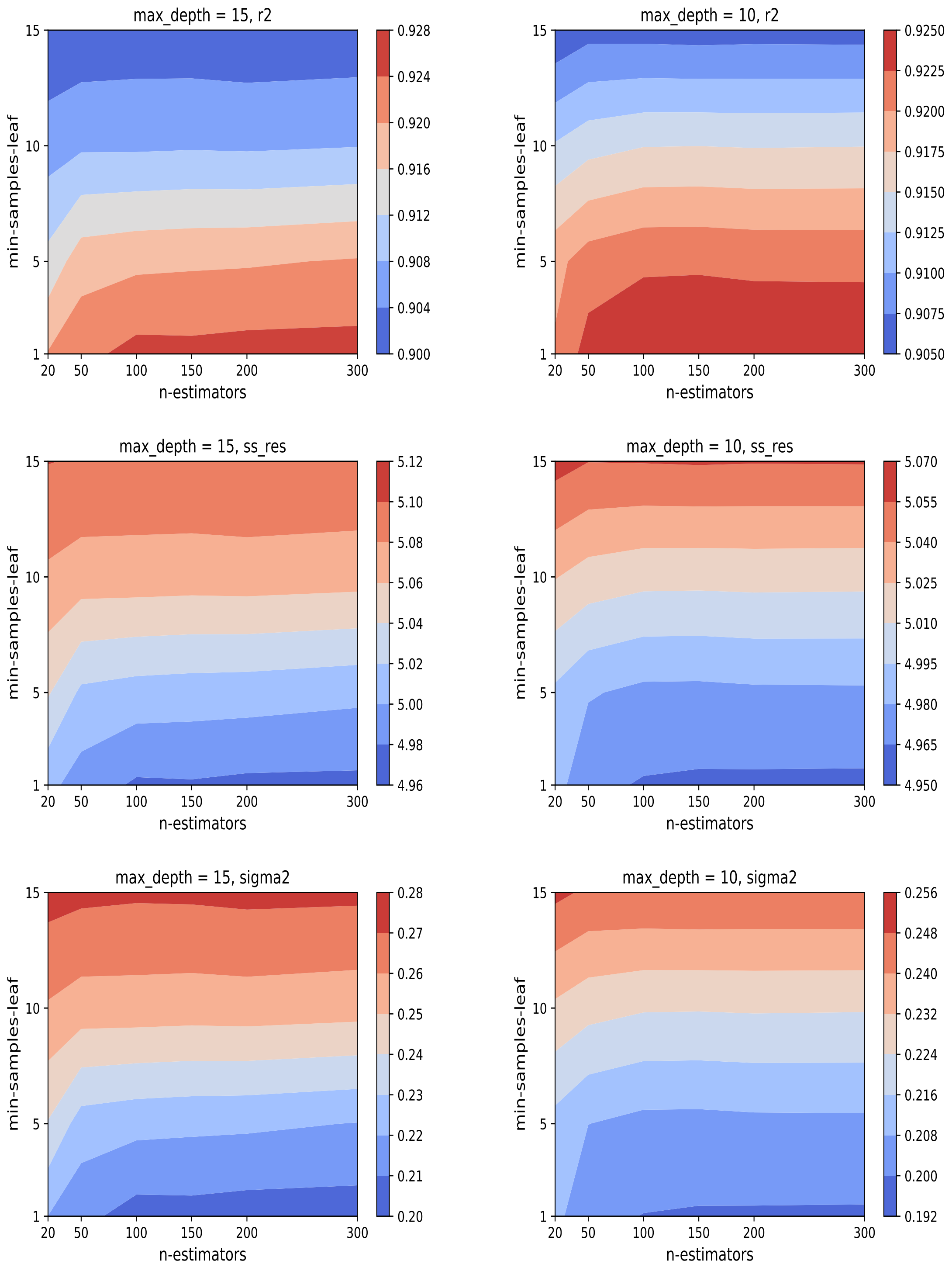


FIG. 13: Test scores after grid search for 10% of satellite data used for training after mass cut. Test size = 10%. Left: All features, Right: Top features, `max_depth = fixed`.

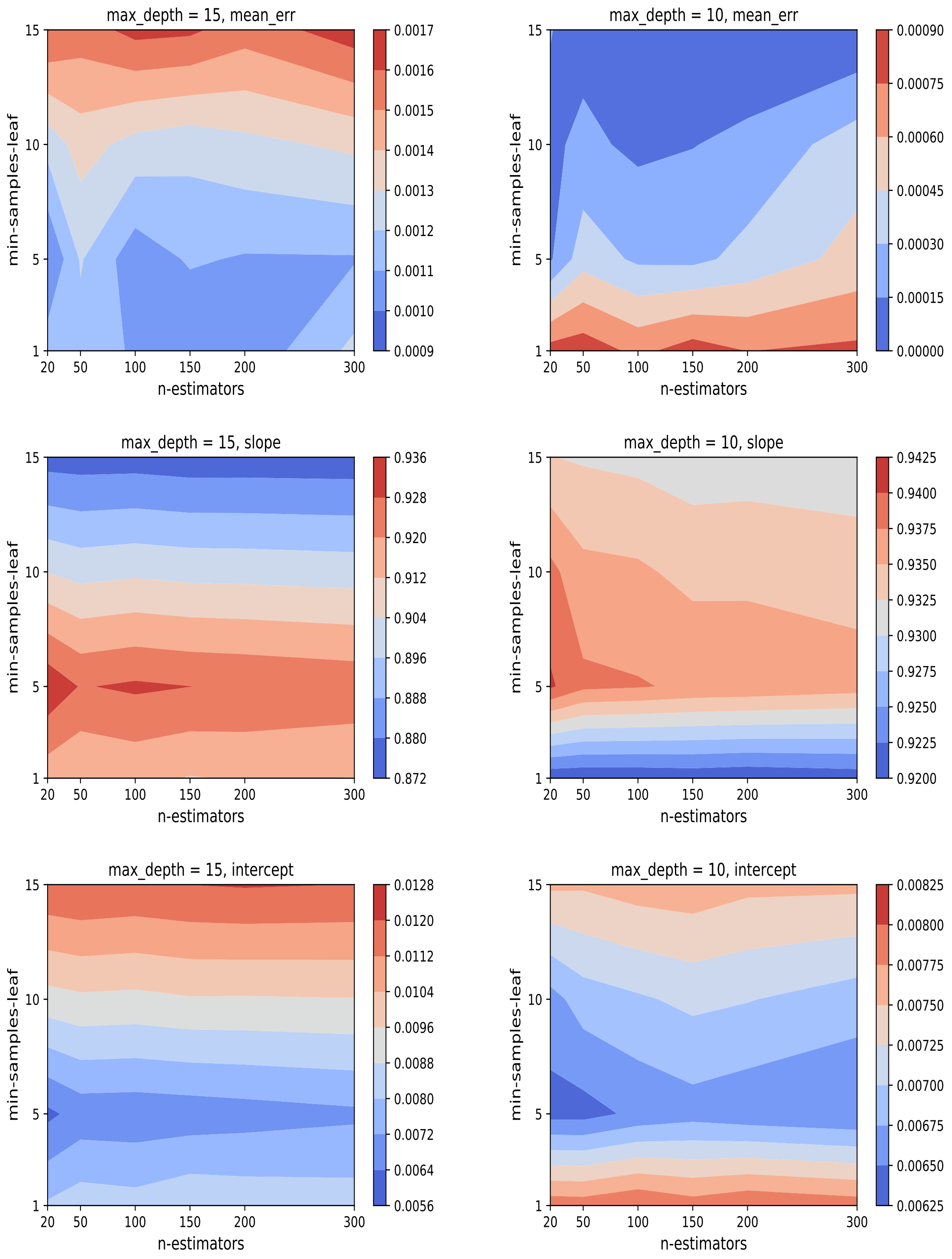


FIG. 14: Test scores after grid search for 10% of satellite data used for training after mass cut. Test size = 10%. Left: All features, Right: Top features, `max_depth = fixed`.

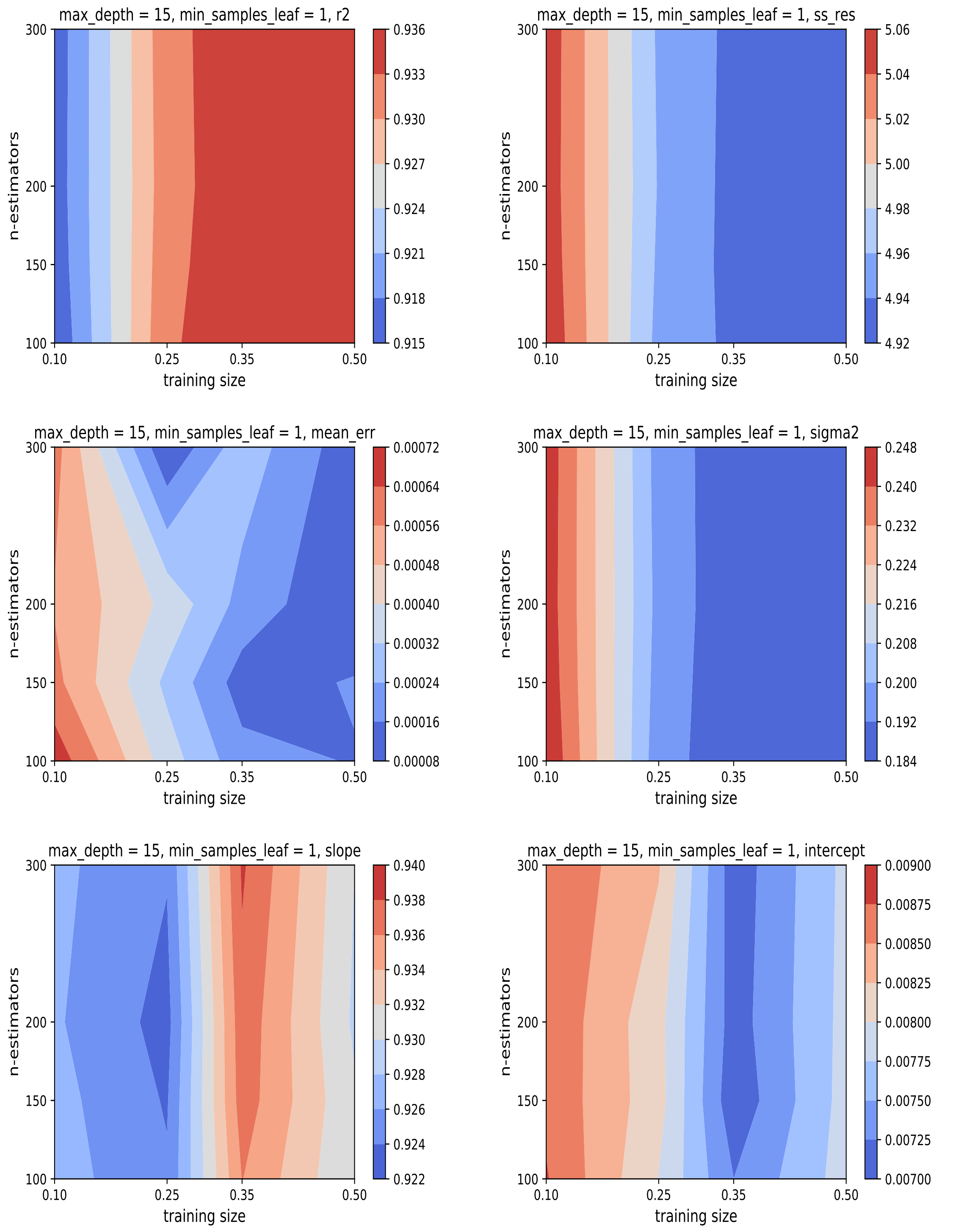


FIG. 15: Test scores after grid search for varying training size of satellite data after mass cut. Test size = 10%. `max_depth = 15, min_samples_leaf = 1`.

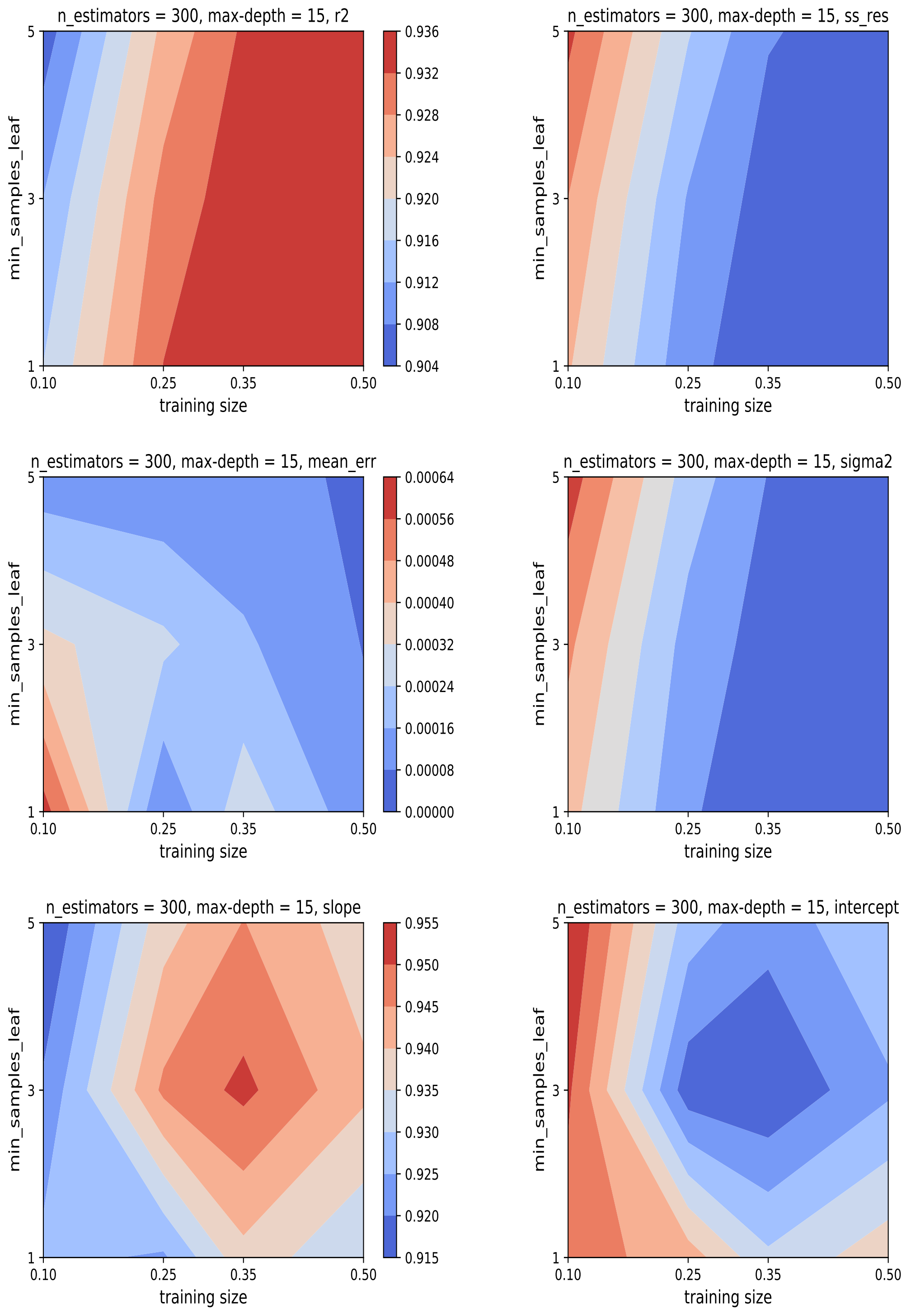


FIG. 16: Test scores after grid search for varying training size of satellite data after mass cut. Test size = 10%. `max_depth = 15, n_estimators = 300`.

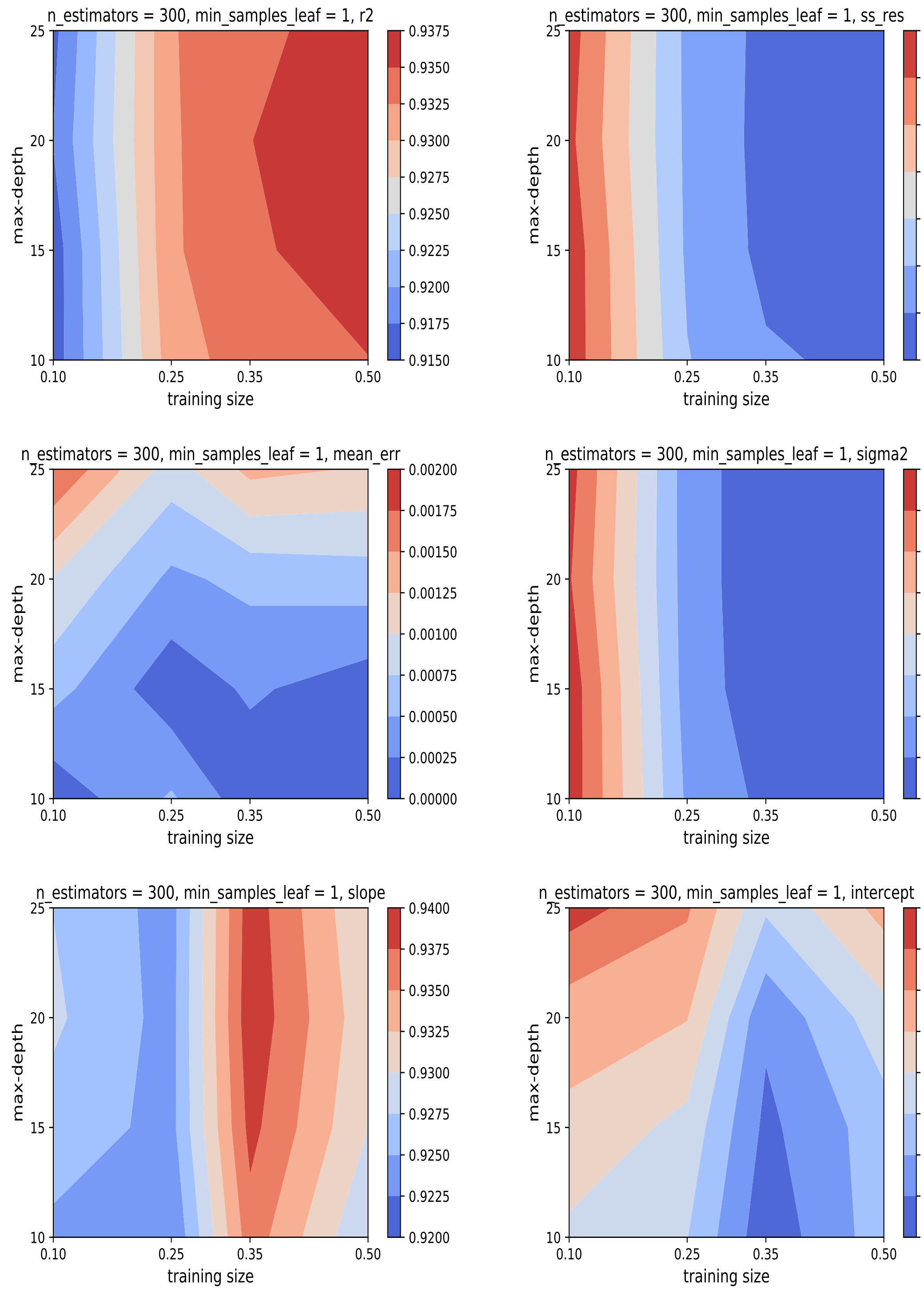


FIG. 17: Test scores after grid search for varying training size of satellite data after mass cut. Test size = 10%. `min_samples_leaf = 1, n_estimators = 300`.

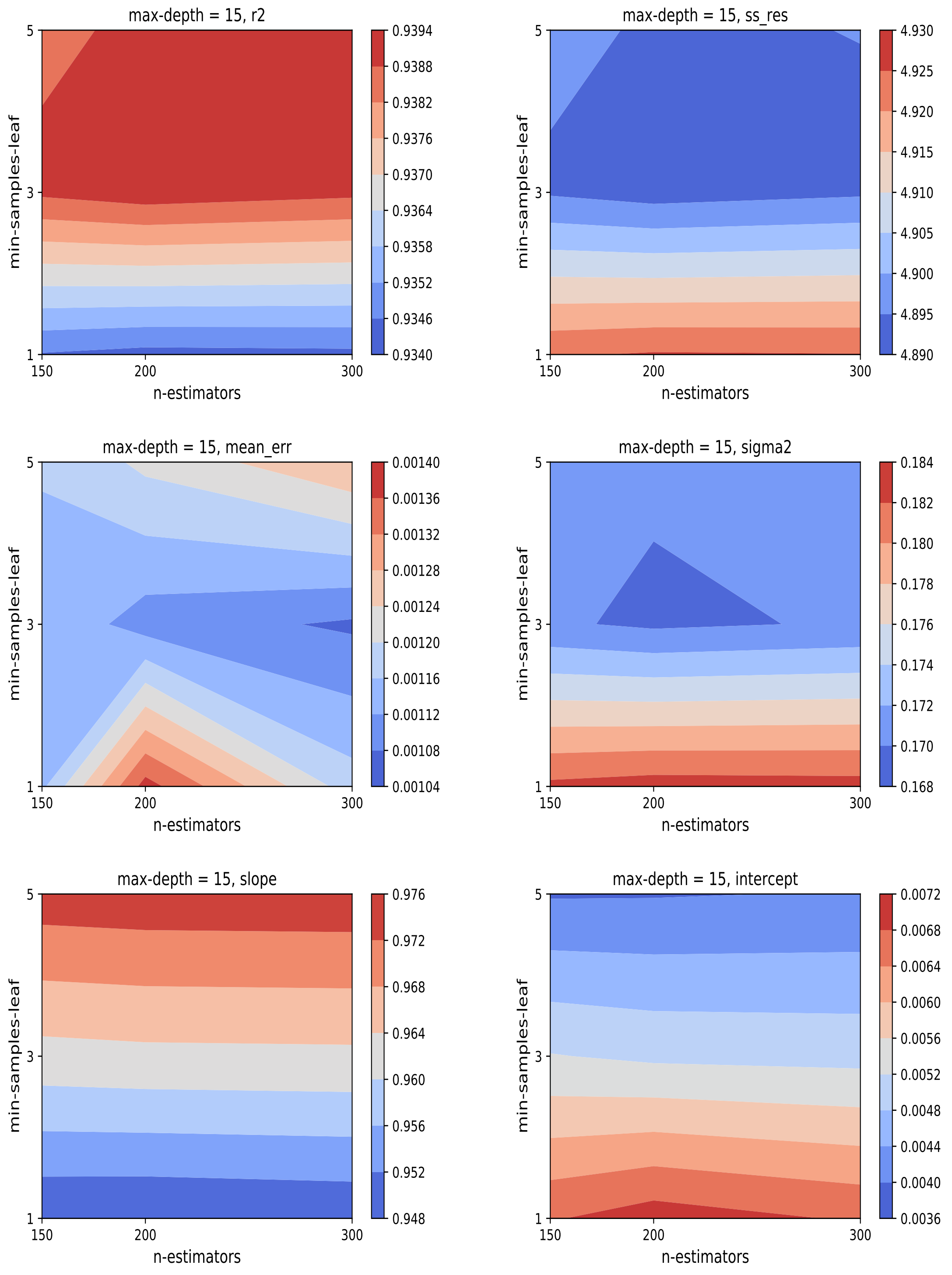


FIG. 18: Test scores after grid search for 80% training size of satellite data after mass cut. Test size = 10%. `max_depth = 15`. All features.

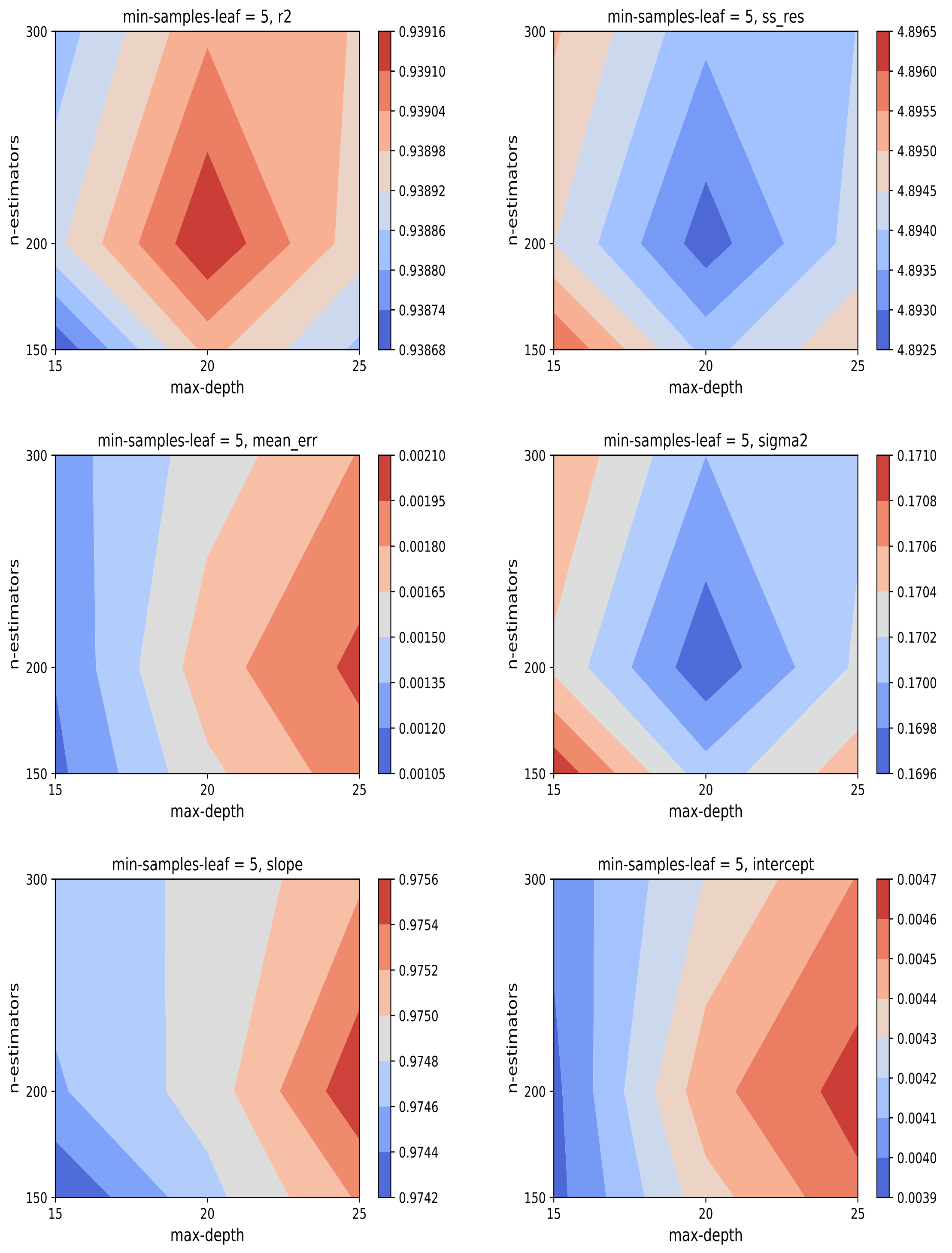


FIG. 19: Test scores after grid search for 80% training size of satellite data after mass cut. Test size = 10%. `min_samples_leaf = 5`.

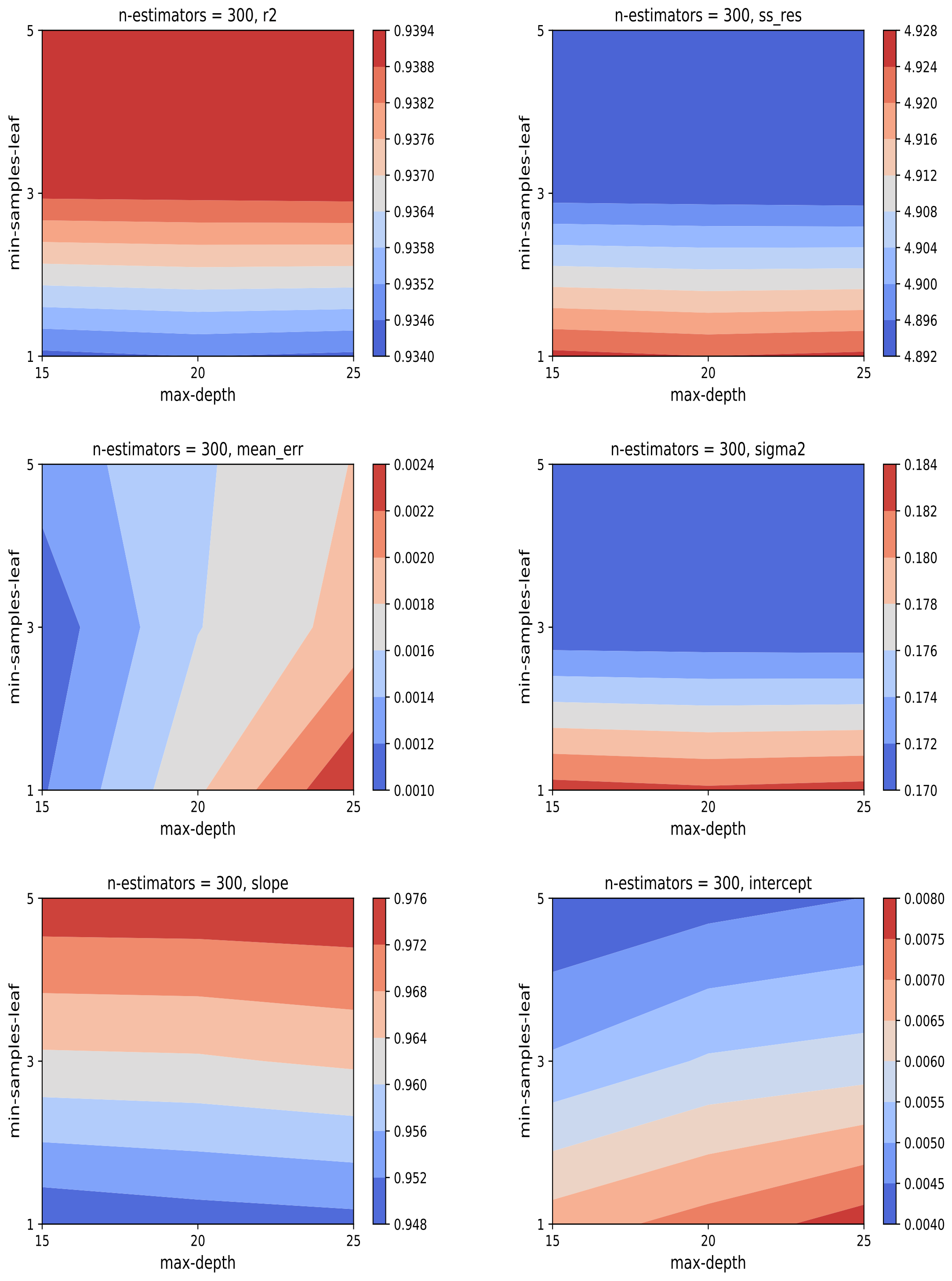


FIG. 20: Test scores after grid search for 80% training size of satellite data after mass cut. Test size = 10%. `n_estimators = 300`.

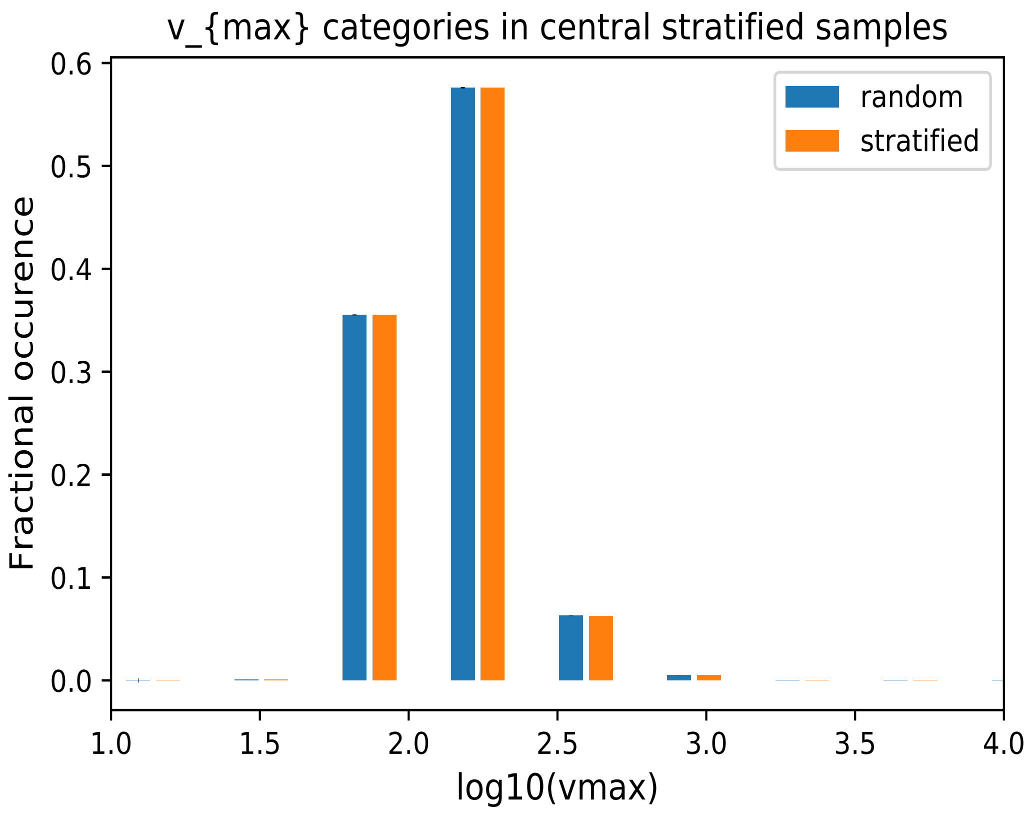


FIG. 21: Distribution of halo `vmax`, v_{max}, in the central data.

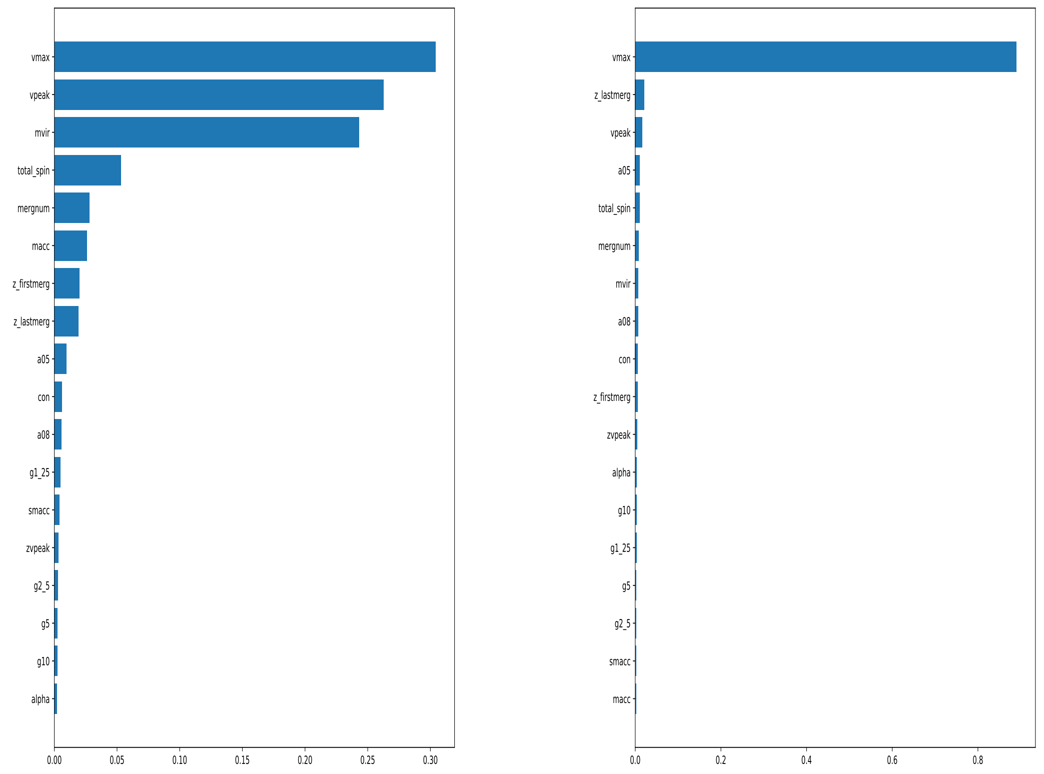


FIG. 22: Feature importances of the RF classifier model for central galaxies. Left: `max_features = log2`, Right: `max_features = 1.0`. It appears that in `log2`, the feature importances are spread over all halo features, whereas in `1.0`, halo mass `vmax` is by far the most dominant feature followed by `z_lastmerg, vpeak,` and `a05`.

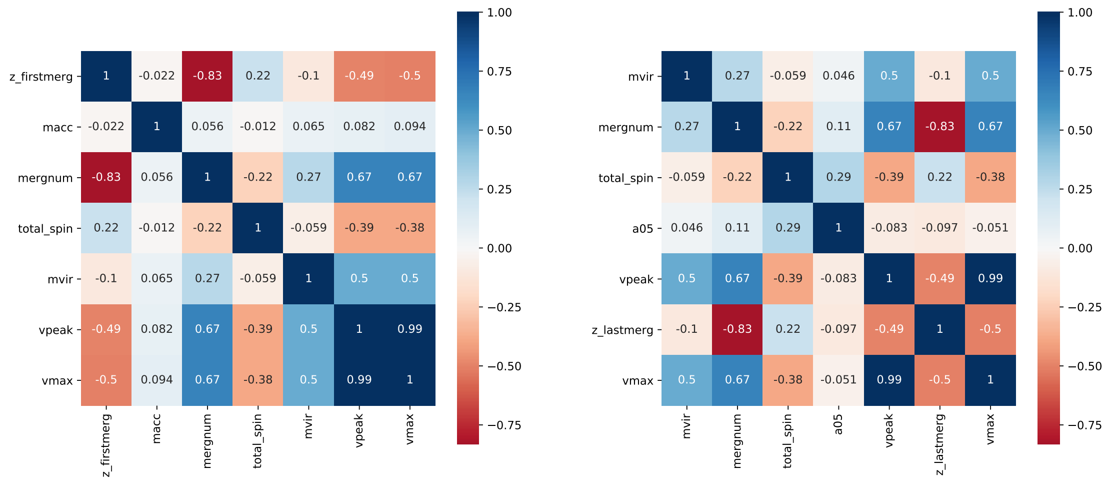


FIG. 23: Correlation matrix or heatmap of the top 7 features for the central data. Left: `max_features = log2`, Right: `max_features = 1.0`.

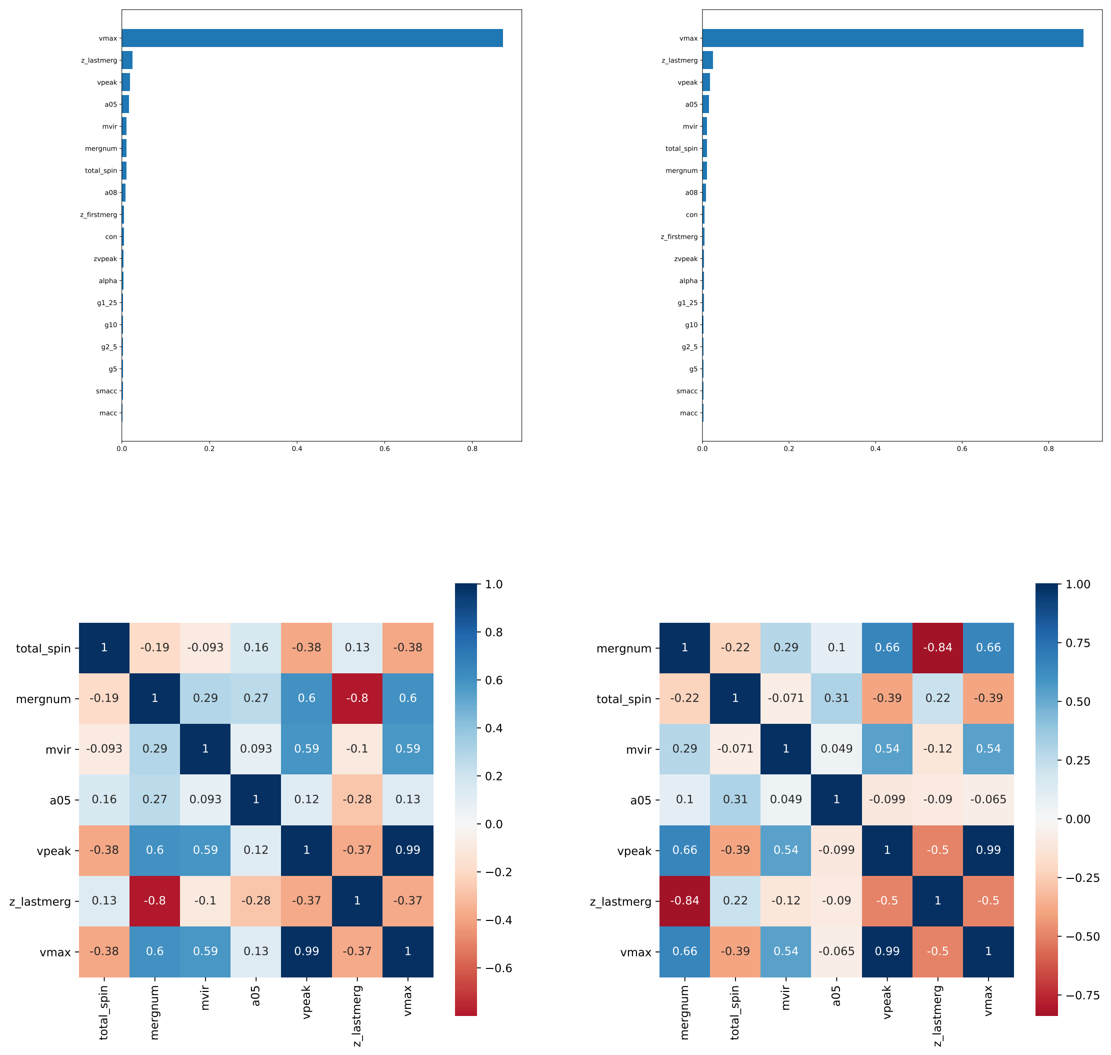


FIG. 24: Feature importances (top) and heatmap (bottom) from random forest classifier. Left: with cut, Right: no cut.

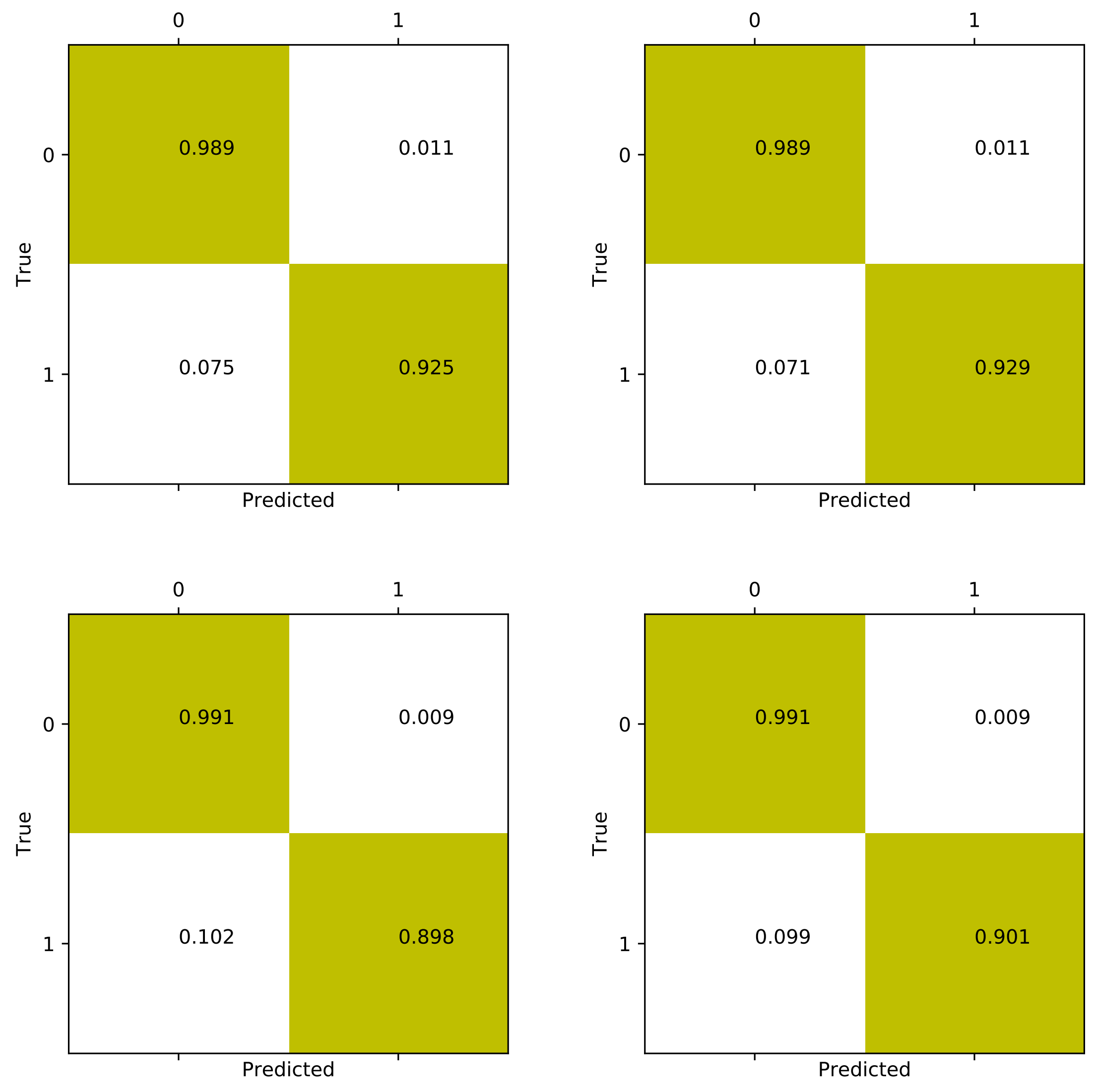


FIG. 25: Confusion matrix of the total set for uncorrelated important features for central galaxies using *top 5* features. Left: `max_features = log2`, Right: `max_features = 1.0`.

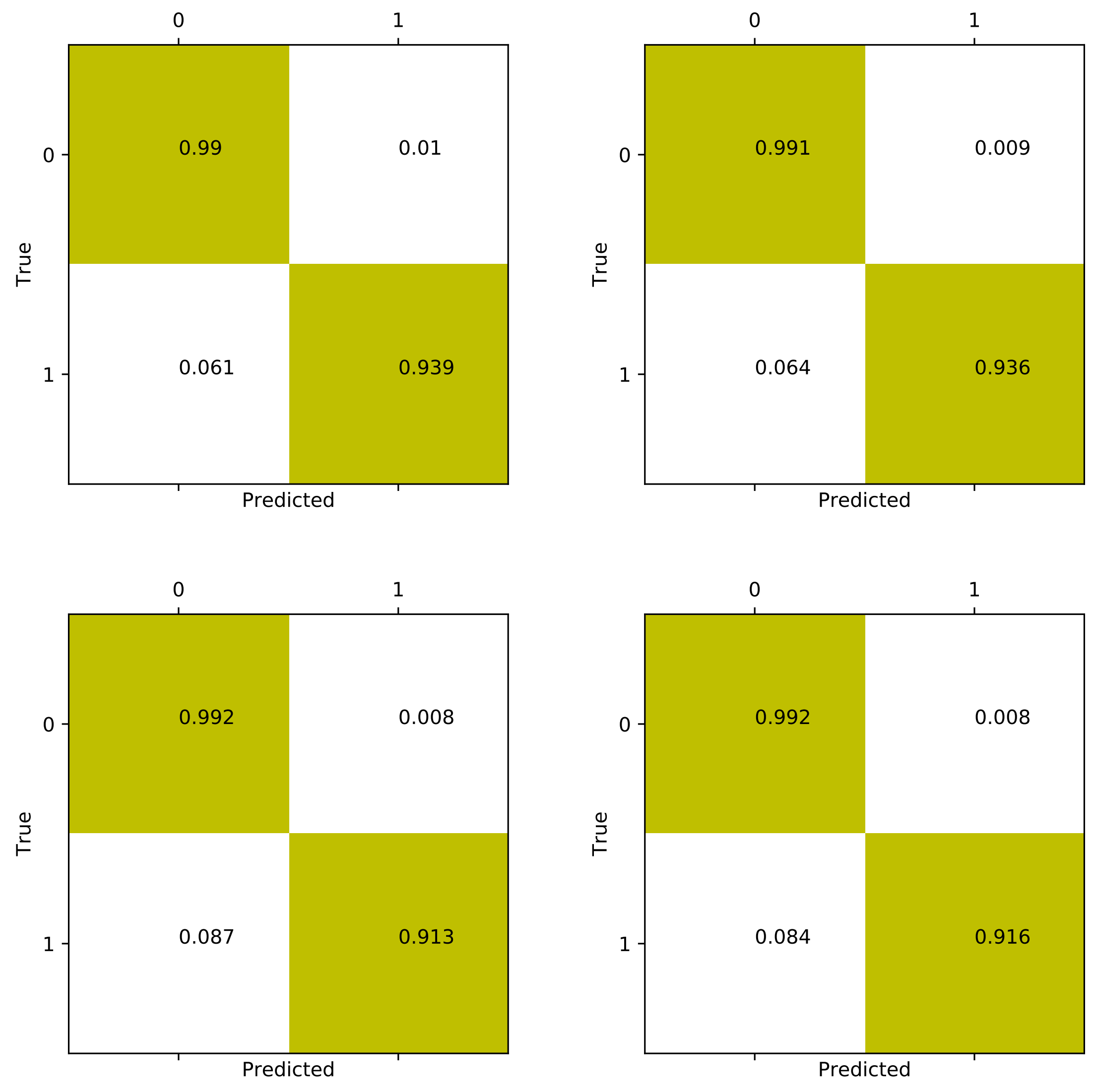


FIG. 26: Confusion matrix of the total set for uncorrelated important features for central galaxies using *all* features. Left: `max_features = log2`, Right: `max_features = 1.0`.

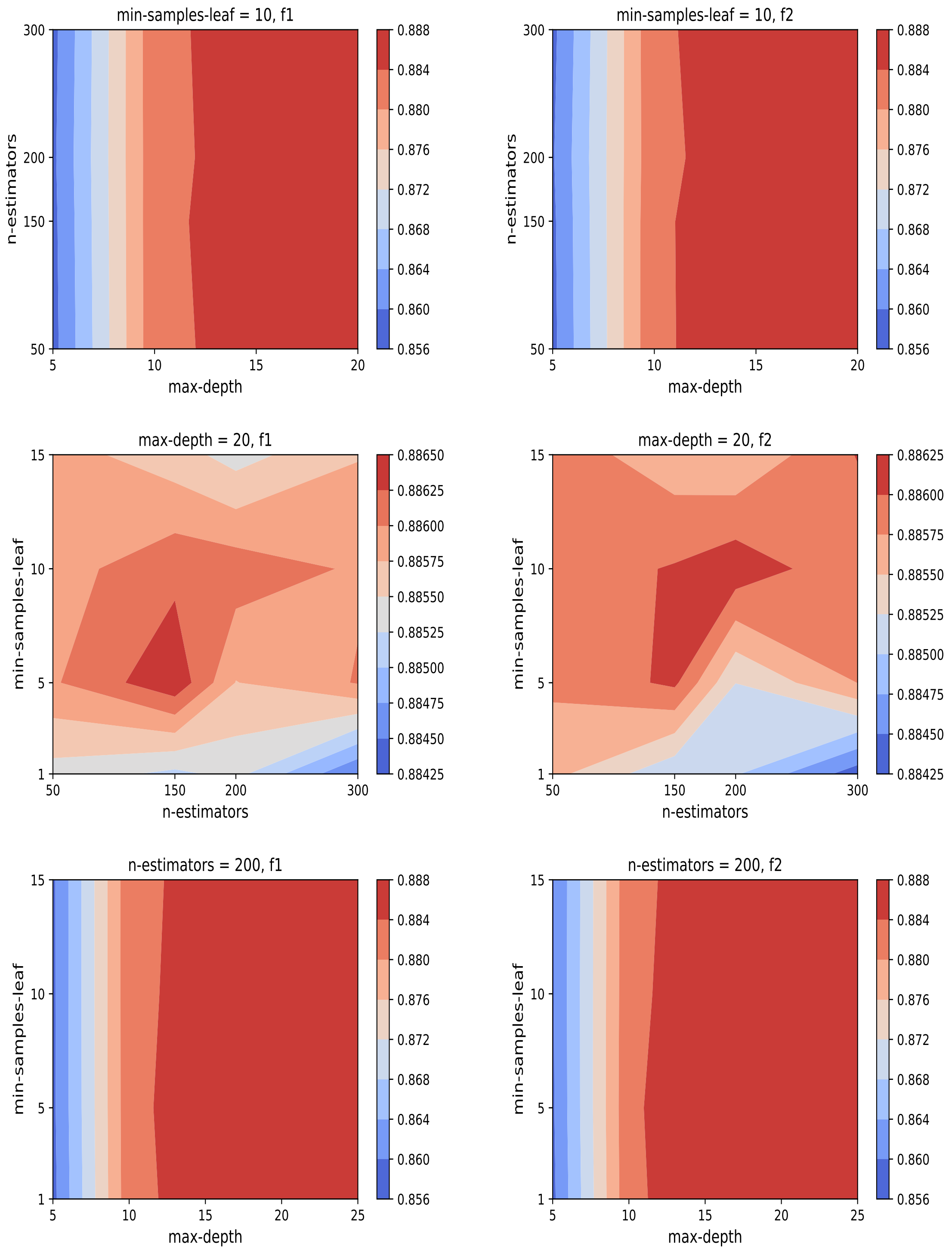


FIG. 27: Features = all. Train size = 10%.

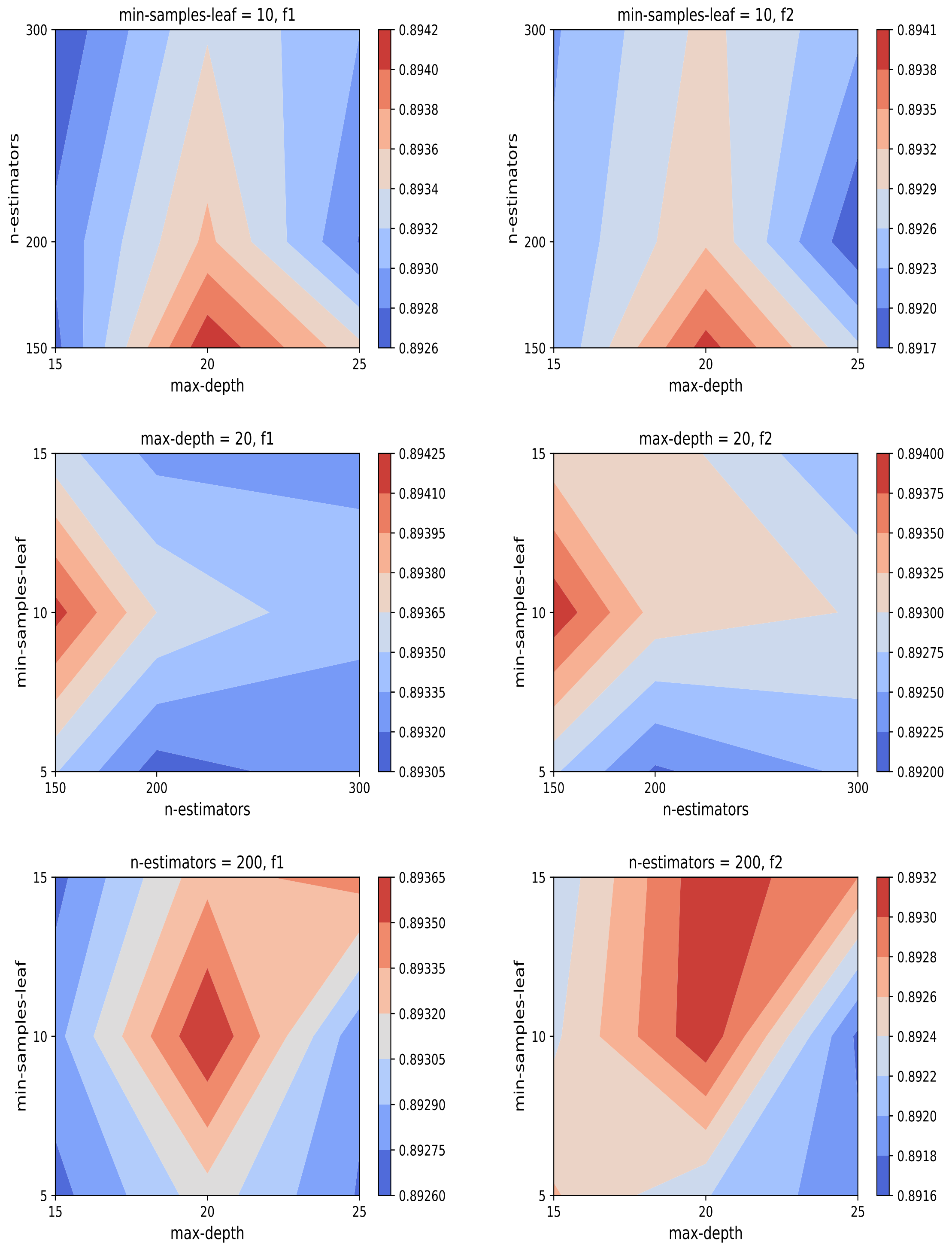


FIG. 28: Features = all. Train size = 80%.

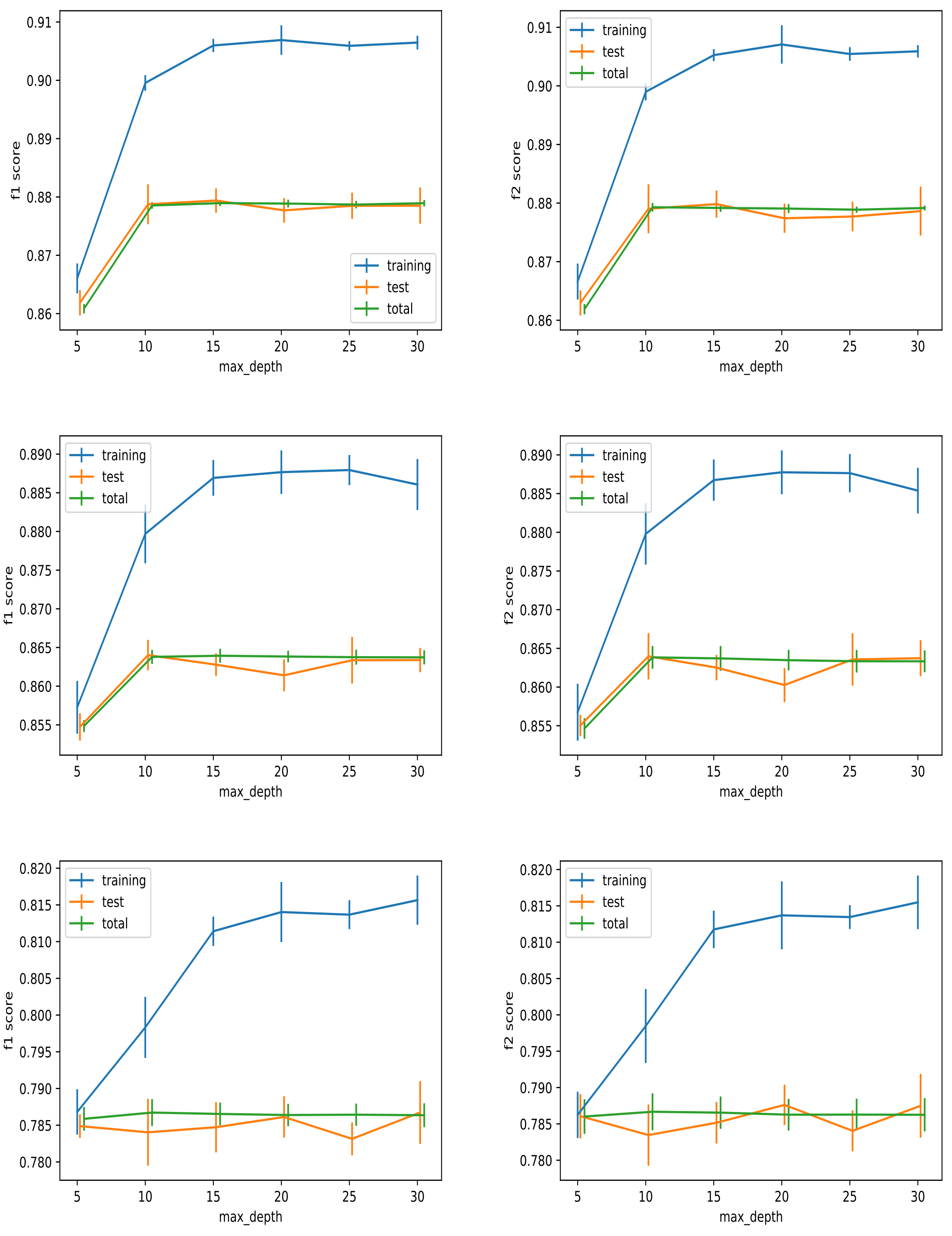


FIG. 29: `n_estimators=200, min_samples_leaf=10`. Top to bottom: All, top+env, `mvir+g1_25`.

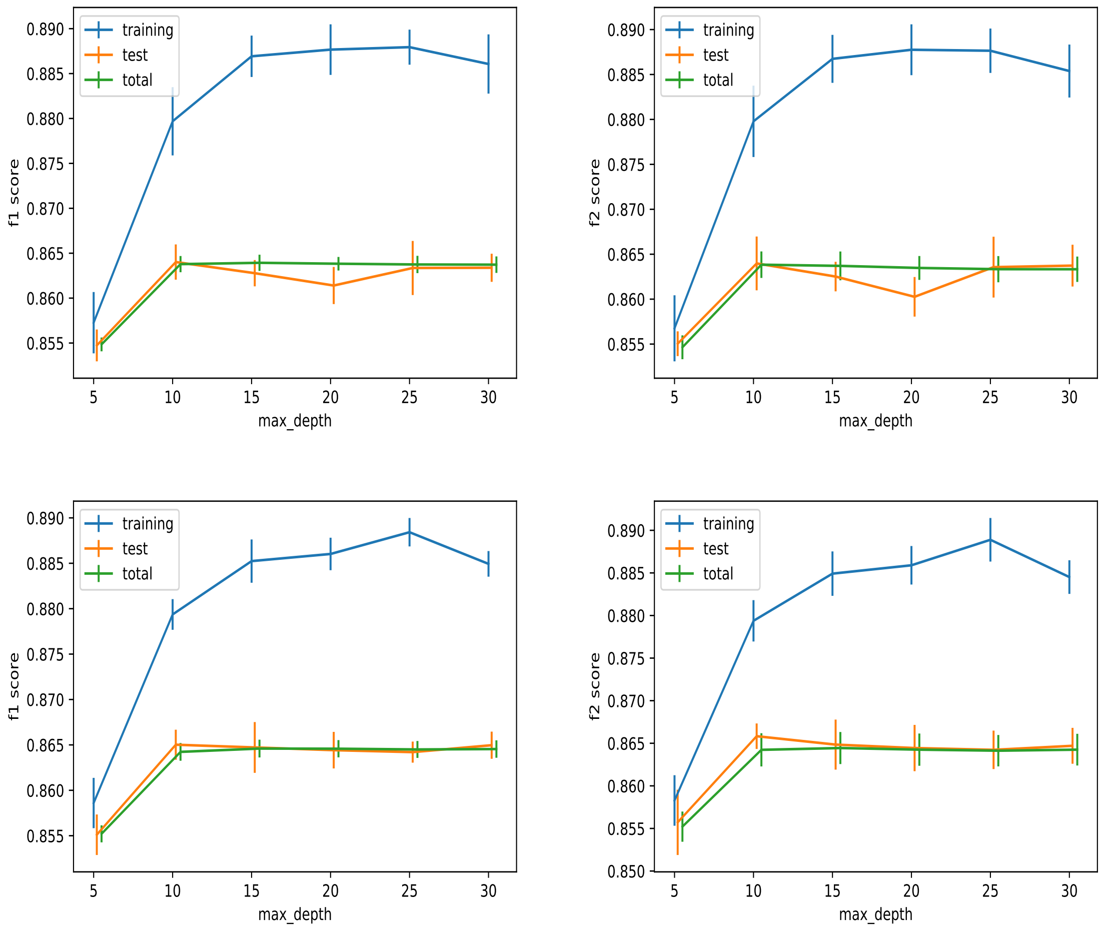
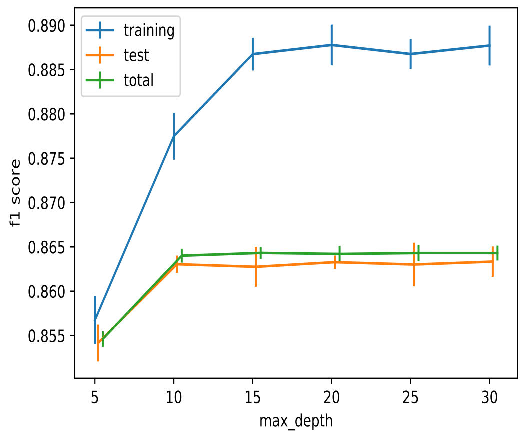
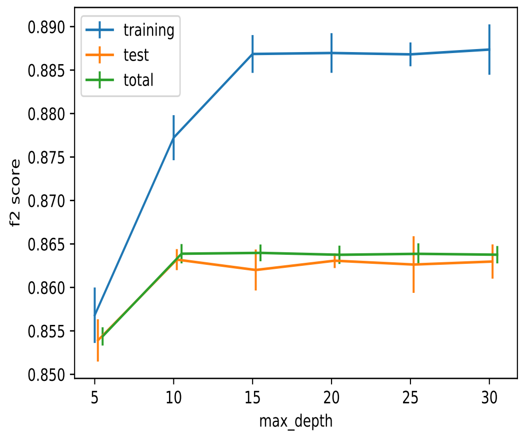
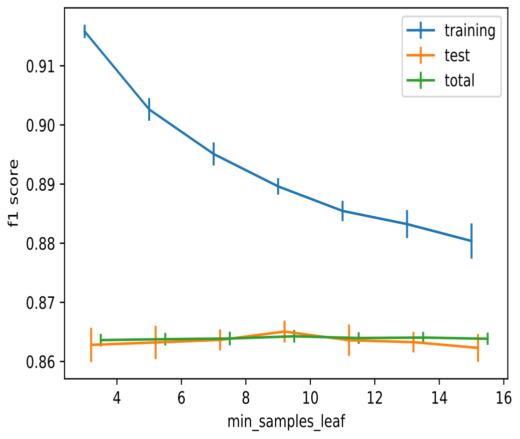
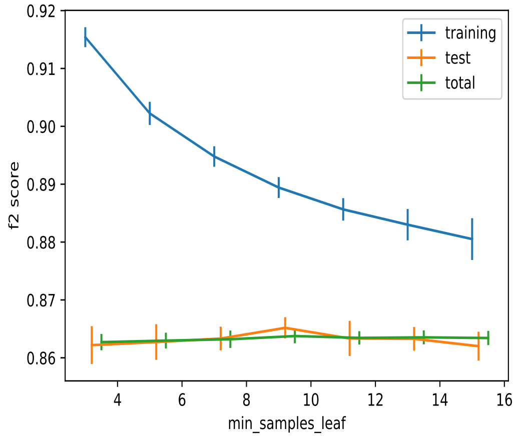


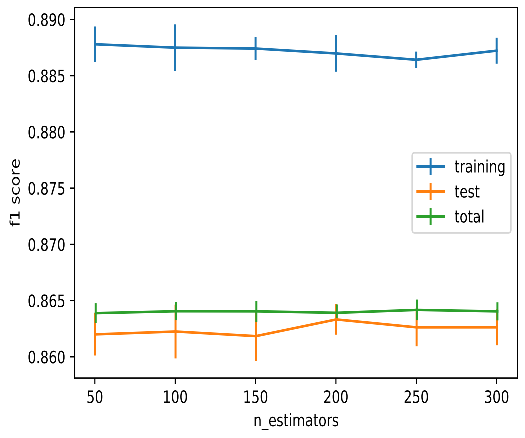
FIG. 30: `n_estimators=200, min_samples_leaf=10`. Top to bottom: top+env, top.

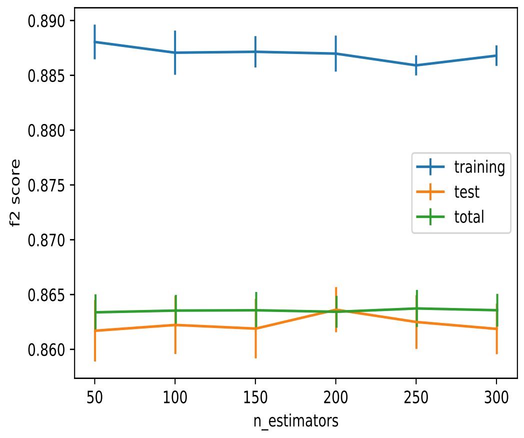

training
test
total
f1 score
0.890
0.885
0.880
0.875
0.870
0.865
0.860
0.855
5
10
15
20
25
30
max_depth

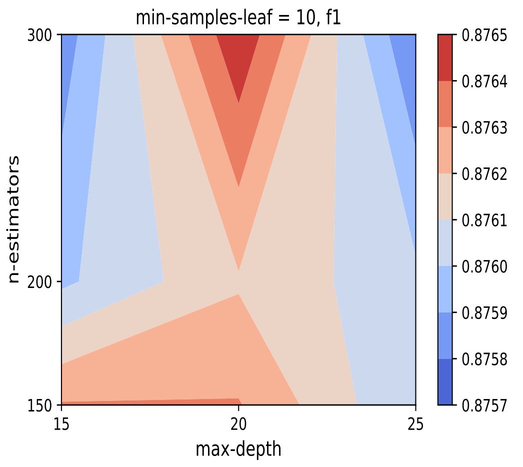

training
test
total
f2 score
0.890
0.885
0.880
0.875
0.870
0.865
0.860
0.855
0.850
5
10
15
20
25
30
max_depth

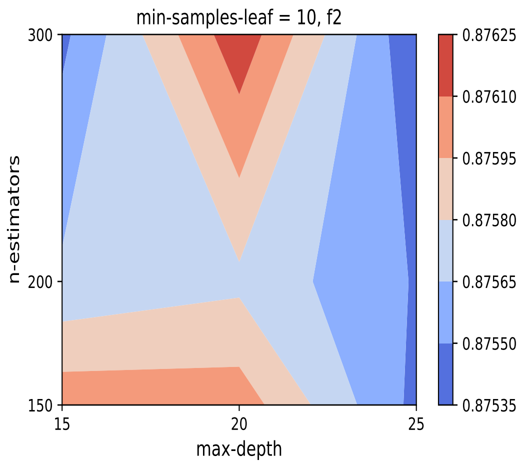

training
test
total
f1 score
0.91
0.90
0.89
0.88
0.87
0.86
4
6
8
10
12
14
16
min_samples_leaf

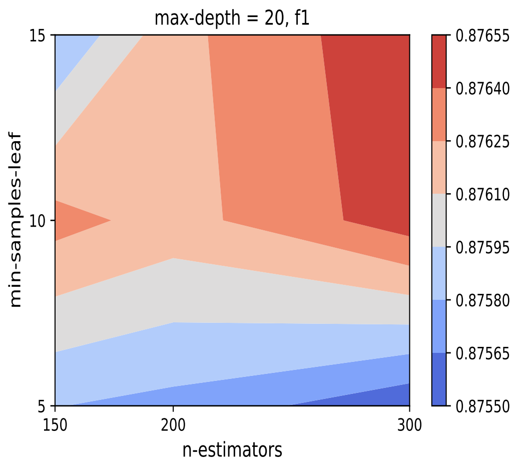

training
test
total
f2 score
0.92
0.91
0.90
0.89
0.88
0.87
0.86
4
6
8
10
12
14
16
min_samples_leaf

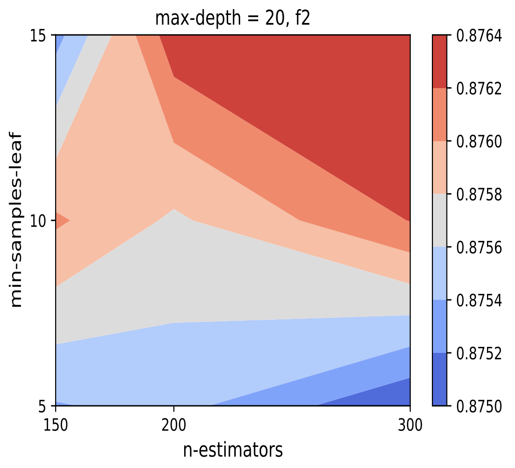

training
test
total
f1 score
0.890
0.885
0.880
0.875
0.870
0.865
0.860
50
100
150
200
250
300
n_estimators

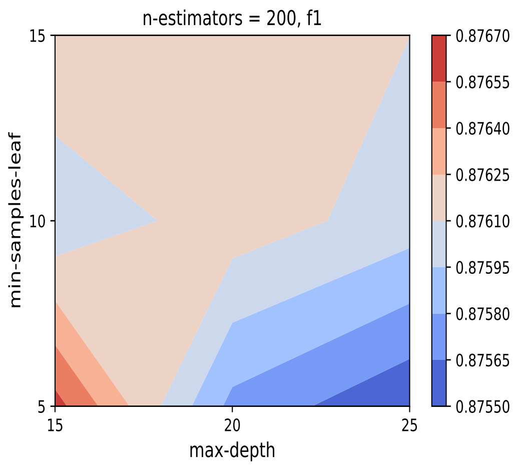

training
test
total
f2 score
0.890
0.885
0.880
0.875
0.870
0.865
0.860
50
100
150
200
250
300
n_estimators

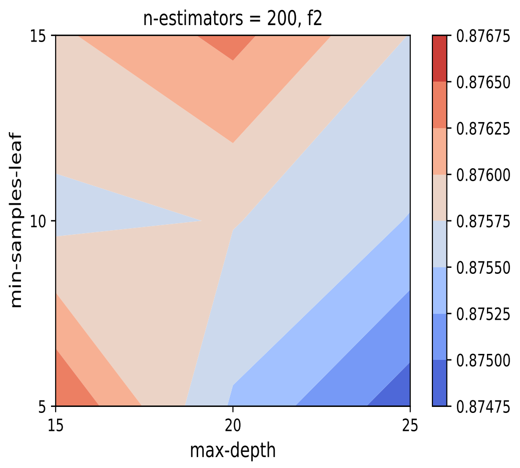

min-samples-leaf = 10, f1
n-estimators
300
200
150
max-depth
15
20
25
0.8765
0.8764
0.8763
0.8762
0.8761
0.8760
0.8759
0.8758
0.8757


min-samples-leaf = 10, f2
n-estimators
300
200
150
max-depth
15
20
25
0.87625
0.87610
0.87595
0.87580
0.87565
0.87550
0.87535


max-depth = 20, f1
min-samples-leaf
15
10
5
n-estimators
150
200
300
0.87655
0.87640
0.87625
0.87610
0.87595
0.87580
0.87565
0.87550


max-depth = 20, f2
min-samples-leaf
15
10
5
n-estimators
150
200
300
0.8764
0.8762
0.8760
0.8758
0.8756
0.8754
0.8752
0.8750


n-estimators = 200, f1
min-samples-leaf
15
10
5
max-depth
15
20
25
0.87670
0.87655
0.87640
0.87625
0.87610
0.87595
0.87580
0.87565
0.87550


n-estimators = 200, f2
min-samples-leaf
15
10
5
max-depth
15
20
25
0.87675
0.87650
0.87625
0.87600
0.87575
0.87550
0.87525
0.87500
0.87475

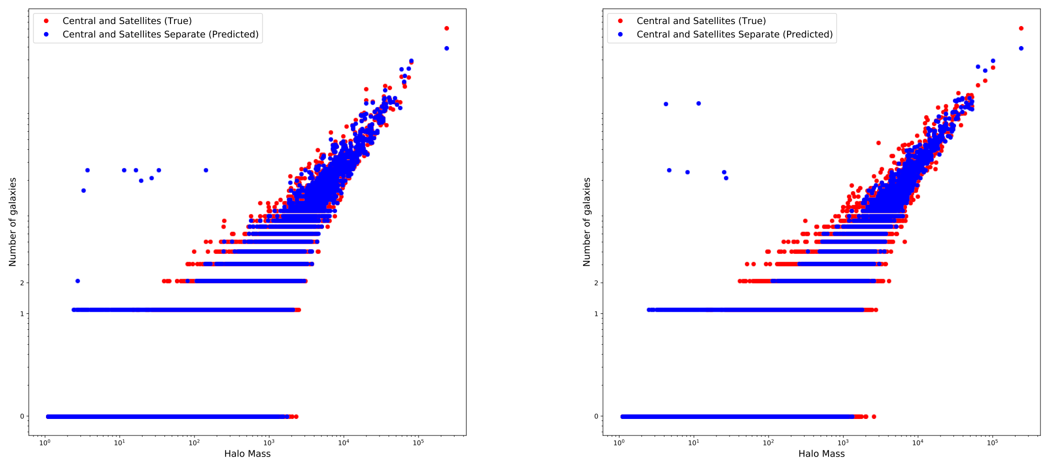


FIG. 31: Number of galaxies vs. halo mass. The halo masses shown are in units of $10^{10}\ h^{-1}\ \mathrm{M}_{\odot}$. Left: Plot for the training set used to train the gradient boost regressor model. Right: Plot for a randomly selected sample.

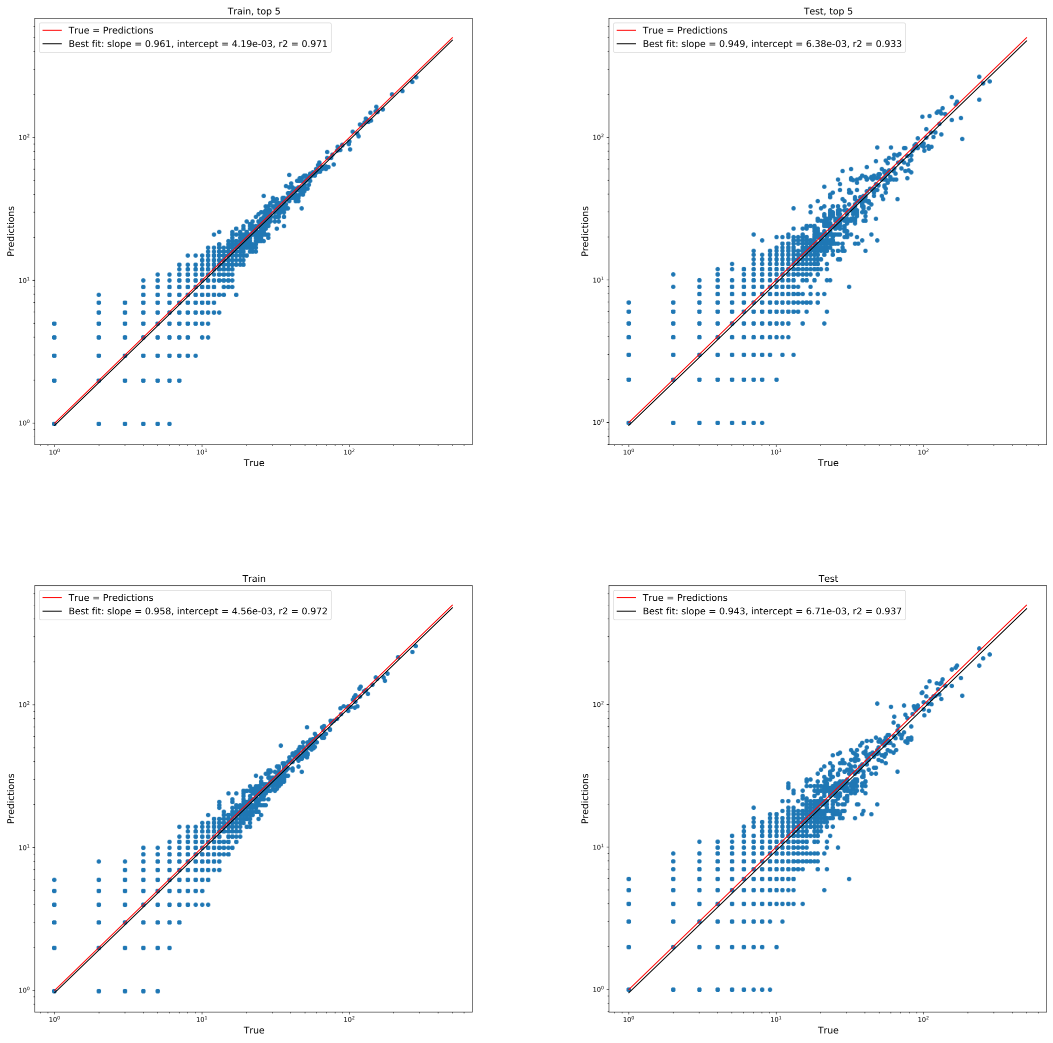
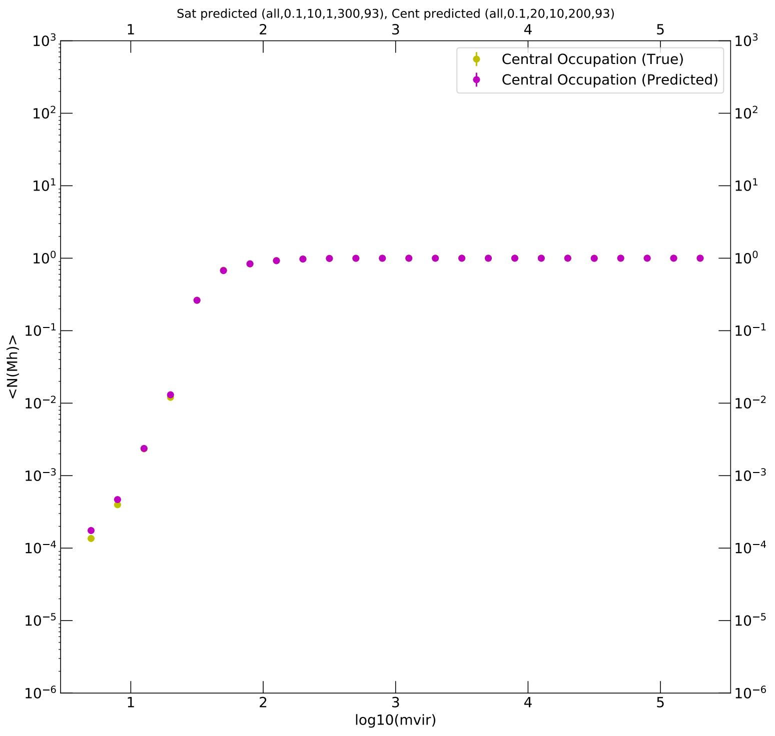


FIG. 32: Prediction vs. true values for the training set (left) and a randomly selected sample (right). Top: Top 5 features, Bottom: All. Hyper parameters used: `max_depth = 10, min_samples_leaf = 1, n_estimators = 300`. Training size = 0.1.



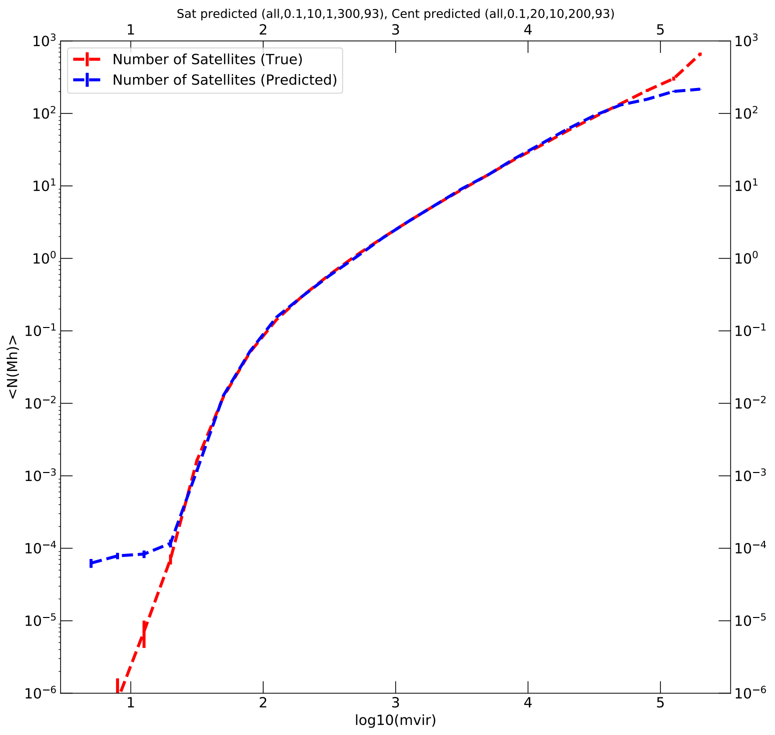


FIG. 33: Halo occupation vs. log10 of halo mass.

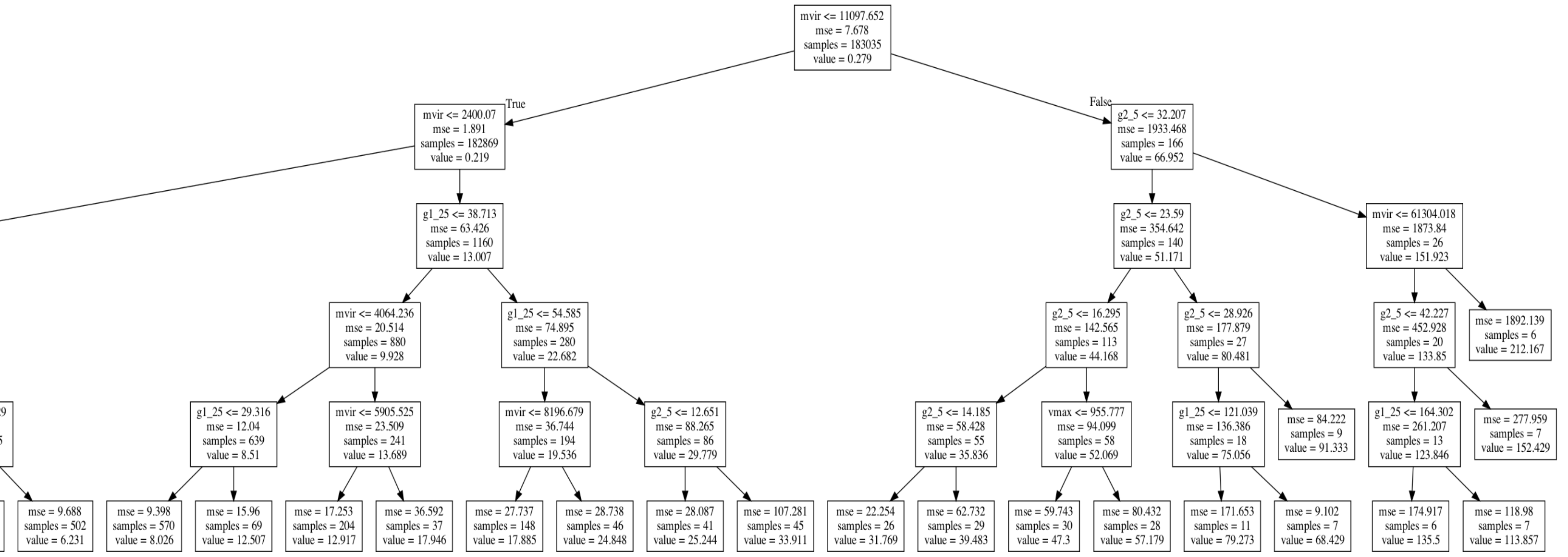

mvir <= 11097.652 mse = 7.678 samples = 183035 value = 0.279
True
False
mvir <= 2400.07 mse = 1.891 samples = 182869 value = 0.219
g2_5 <= 32.207 mse = 1933.468 samples = 166 value = 66.952
g1_25 <= 38.713 mse = 63.426 samples = 1160 value = 13.007
g2_5 <= 23.59 mse = 354.642 samples = 140 value = 51.171
mvir <= 61304.018 mse = 1873.84 samples = 26 value = 151.923
mvir <= 4064.236 mse = 20.514 samples = 880 value = 9.928
g1_25 <= 54.585 mse = 74.895 samples = 280 value = 22.682
g2_5 <= 16.295 mse = 142.565 samples = 113 value = 44.168
g2_5 <= 28.926 mse = 177.879 samples = 27 value = 80.481
g2_5 <= 42.227 mse = 452.928 samples = 20 value = 133.85
mse = 1892.139 samples = 6 value = 212.167
g1_25 <= 29.316 mse = 12.04 samples = 639 value = 8.51
mvir <= 5905.525 mse = 23.509 samples = 241 value = 13.689
mvir <= 8196.679 mse = 36.744 samples = 194 value = 19.536
g2_5 <= 12.651 mse = 88.265 samples = 86 value = 29.779
g2_5 <= 14.185 mse = 58.428 samples = 55 value = 35.836
vmax <= 955.777 mse = 94.099 samples = 58 value = 52.069
g1_25 <= 121.039 mse = 136.386 samples = 18 value = 75.056
mse = 84.222 samples = 9 value = 91.333
g1_25 <= 164.302 mse = 261.207 samples = 13 value = 123.846
mse = 277.959 samples = 7 value = 152.429
mse = 9.688 samples = 502 value = 6.231
mse = 9.398 samples = 570 value = 8.026
mse = 15.96 samples = 69 value = 12.507
mse = 17.253 samples = 204 value = 12.917
mse = 36.592 samples = 37 value = 17.946
mse = 27.737 samples = 148 value = 17.885
mse = 28.738 samples = 46 value = 24.848
mse = 28.087 samples = 41 value = 25.244
mse = 107.281 samples = 45 value = 33.911
mse = 22.254 samples = 26 value = 31.769
mse = 62.732 samples = 29 value = 39.483
mse = 59.743 samples = 30 value = 47.3
mse = 80.432 samples = 28 value = 57.179
mse = 171.653 samples = 11 value = 79.273
mse = 9.102 samples = 7 value = 68.429
mse = 174.917 samples = 6 value = 135.5
mse = 118.98 samples = 7 value = 113.857

FIG. 34: Example of a decision tree with `max_depth` = 5 and `max_features` = 1.

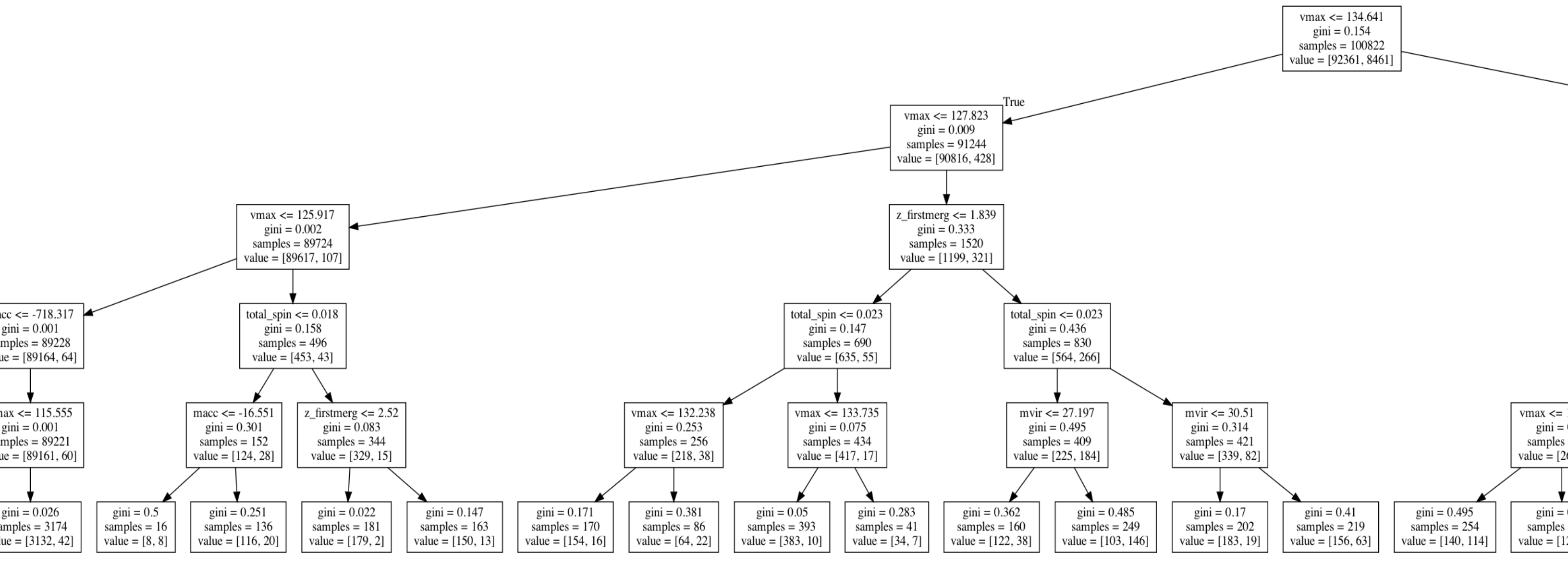

vmax <= 134.641 gini = 0.154 samples = 100822 value = [92361, 8461]
True
vmax <= 127.823 gini = 0.009 samples = 91244 value = [90816, 428]
vmax <= 125.917 gini = 0.002 samples = 89724 value = [89617, 107]
z_firstmerg <= 1.839 gini = 0.333 samples = 1520 value = [1199, 321]
gini = 0.001
total_spin <= 0.018 gini = 0.158 samples = 496 value = [453, 43]
total_spin <= 0.023 gini = 0.147 samples = 690 value = [635, 55]
total_spin <= 0.023 gini = 0.436 samples = 830 value = [564, 266]
gini = 0.001
macc <= -16.551 gini = 0.301 samples = 152 value = [124, 28]
z_firstmerg <= 2.52 gini = 0.083 samples = 344 value = [329, 15]
vmax <= 132.238 gini = 0.253 samples = 256 value = [218, 38]
vmax <= 133.735 gini = 0.075 samples = 434 value = [417, 17]
mvir <= 27.197 gini = 0.495 samples = 409 value = [225, 184]
mvir <= 30.51 gini = 0.314 samples = 421 value = [339, 82]
gini = 0.026
gini = 0.5 samples = 16 value = [8, 8]
gini = 0.251 samples = 136 value = [116, 20]
gini = 0.022 samples = 181 value = [179, 2]
gini = 0.147 samples = 163 value = [150, 13]
gini = 0.171 samples = 170 value = [154, 16]
gini = 0.381 samples = 86 value = [64, 22]
gini = 0.05 samples = 393 value = [383, 10]
gini = 0.283 samples = 41 value = [34, 7]
gini = 0.362 samples = 160 value = [122, 38]
gini = 0.485 samples = 249 value = [103, 146]
gini = 0.17 samples = 202 value = [183, 19]
gini = 0.41 samples = 219 value = [156, 63]
gini = 0.495 samples = 254 value = [140, 114]

FIG. 35: Example of a decision tree with `max_depth` = 5 and `max_features` = 1.