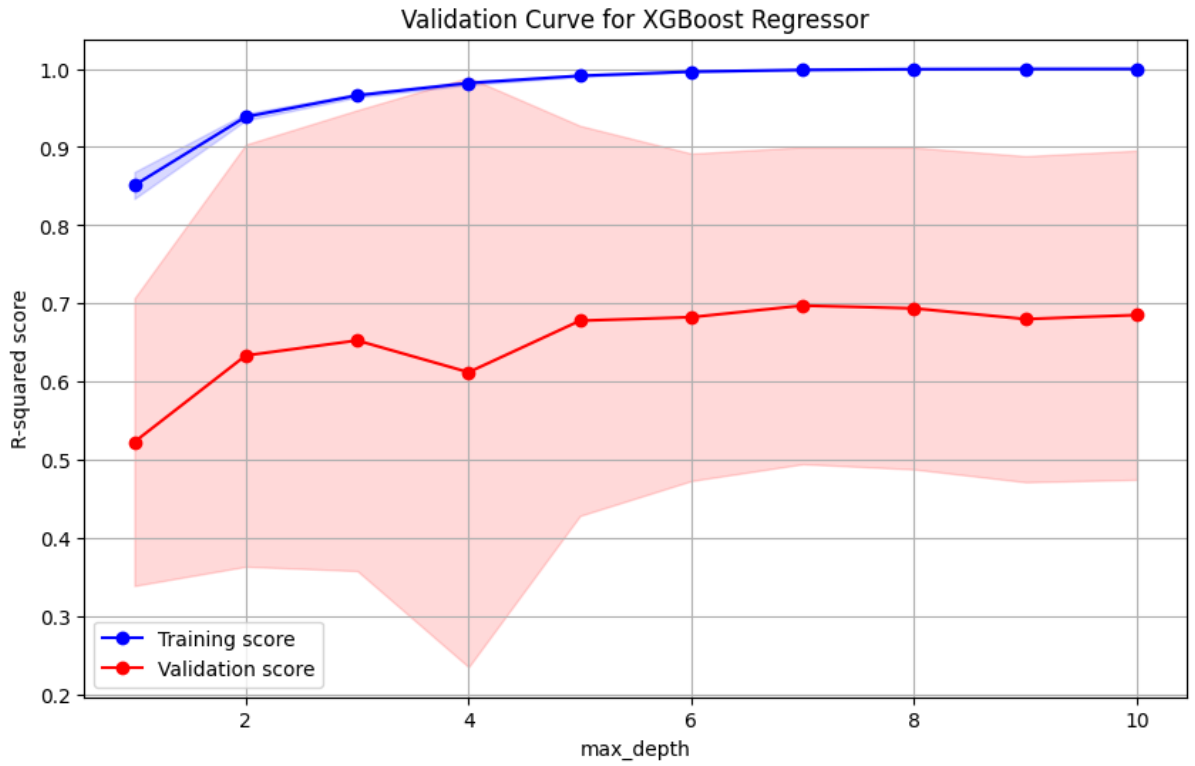
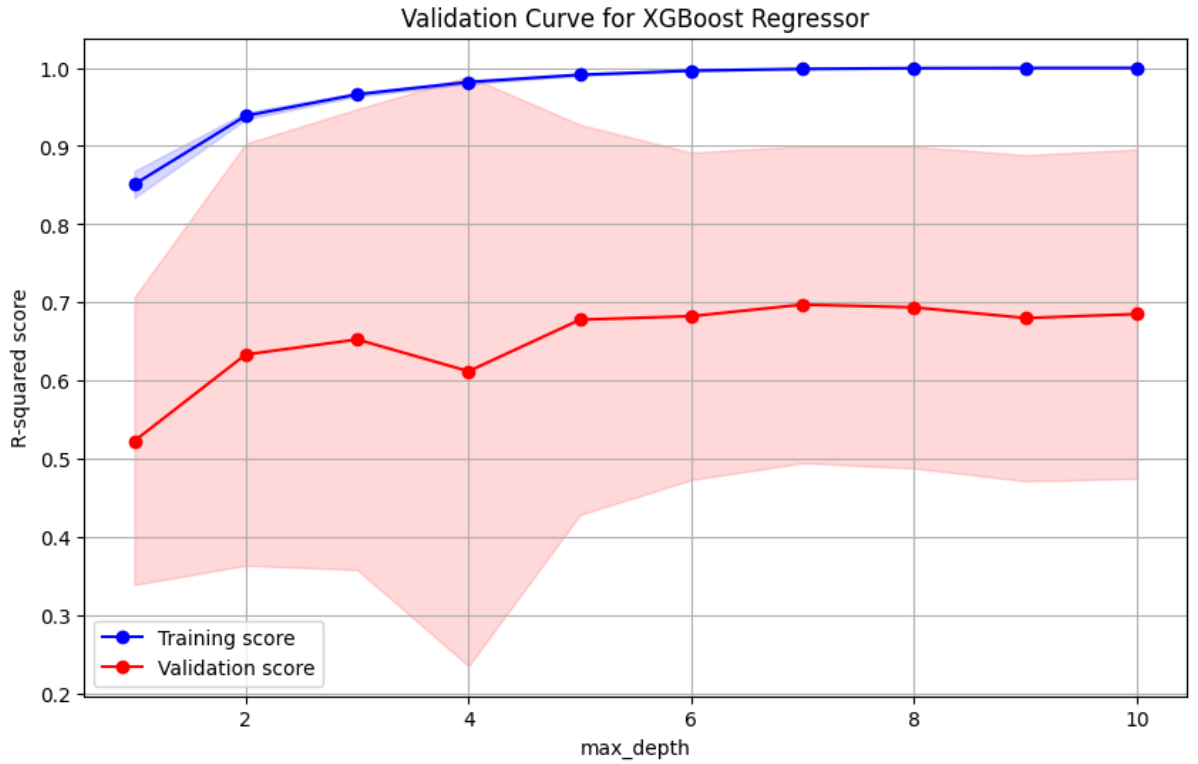
Parameters=

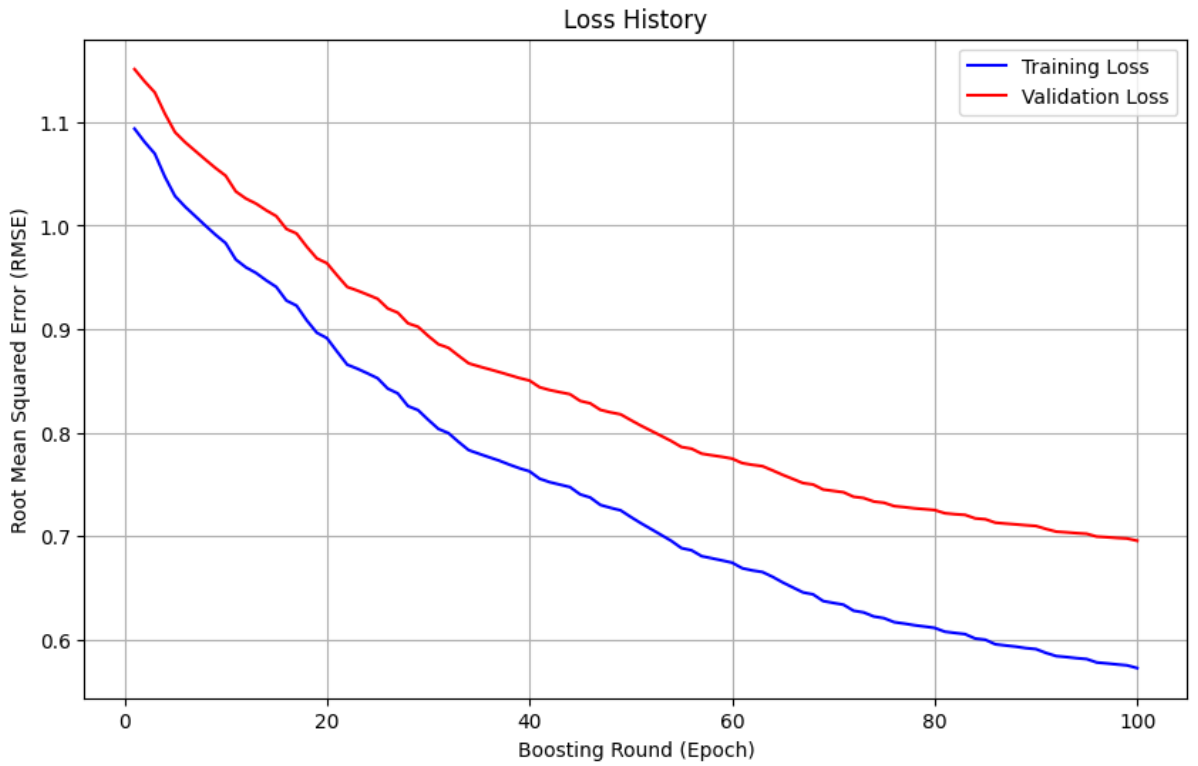
```
param = {
    "booster": "gbtree",
    "learning_rate":0.01,
    "tree_method": "hist",
    "learning_rate": 0.1,
    "max_depth": 5,
    "subsample": 0.5,
    "colsample_bytree": 0.4,
    "gamma": 0.1,
    "reg_alpha": 0.001,
    "reg_lambda": 0.1,
    "objective": "reg:pseudohubererror",
    "min_child_weight": 2,
    "eval_metric": "rmse",
    "early_stopping_rounds": 30,
    "random_state": 452,
    "validate_parameters": 1,
}
xgb_reg = XGB.XGBRegressor(n_estimator=100, **param)
```



Parameters

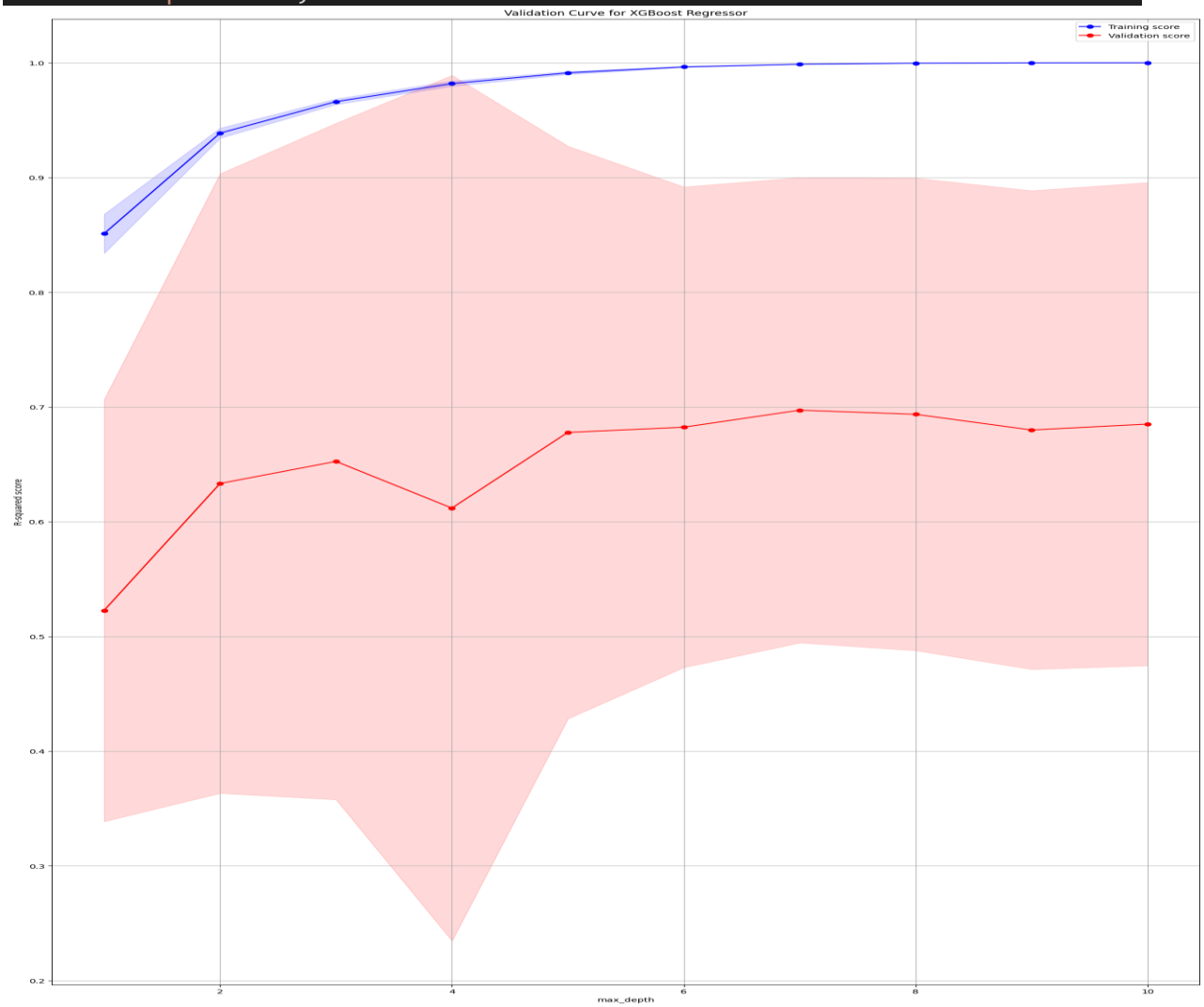
```
    "learning_rate":0.01,
xgb_reg = XGB.XGBRegressor(n_estimator=1000, **param)
```

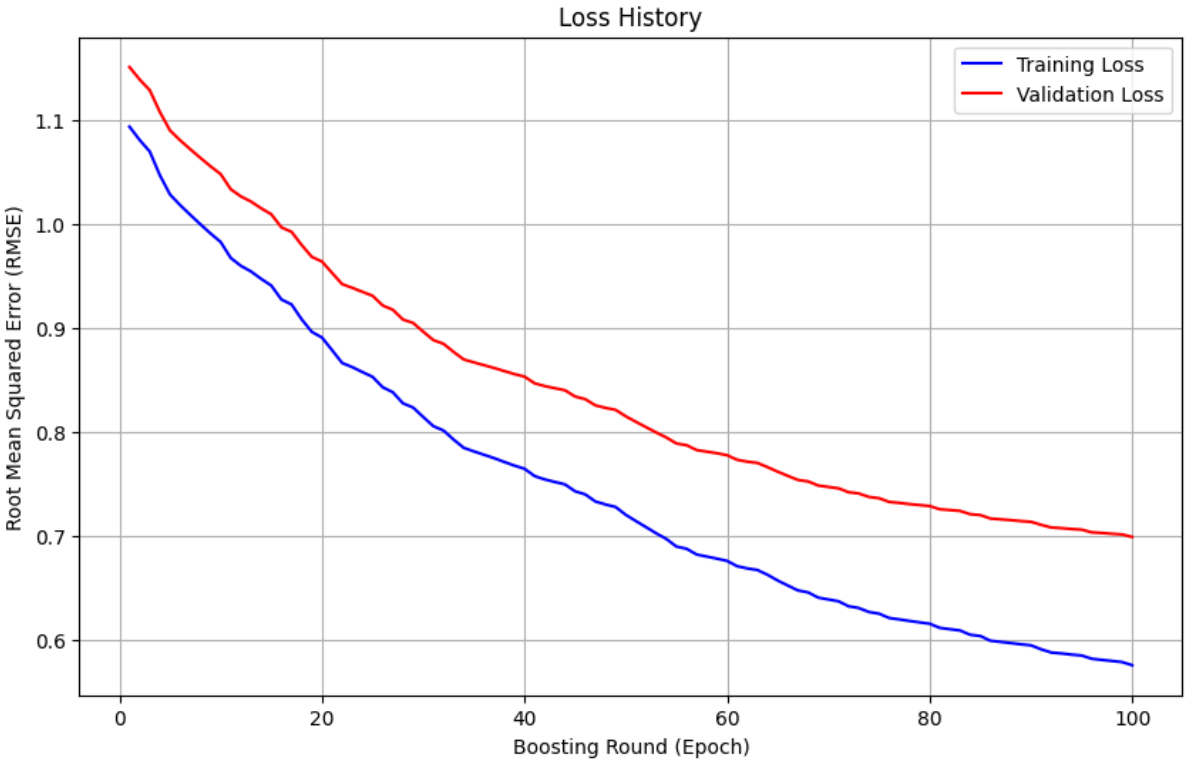




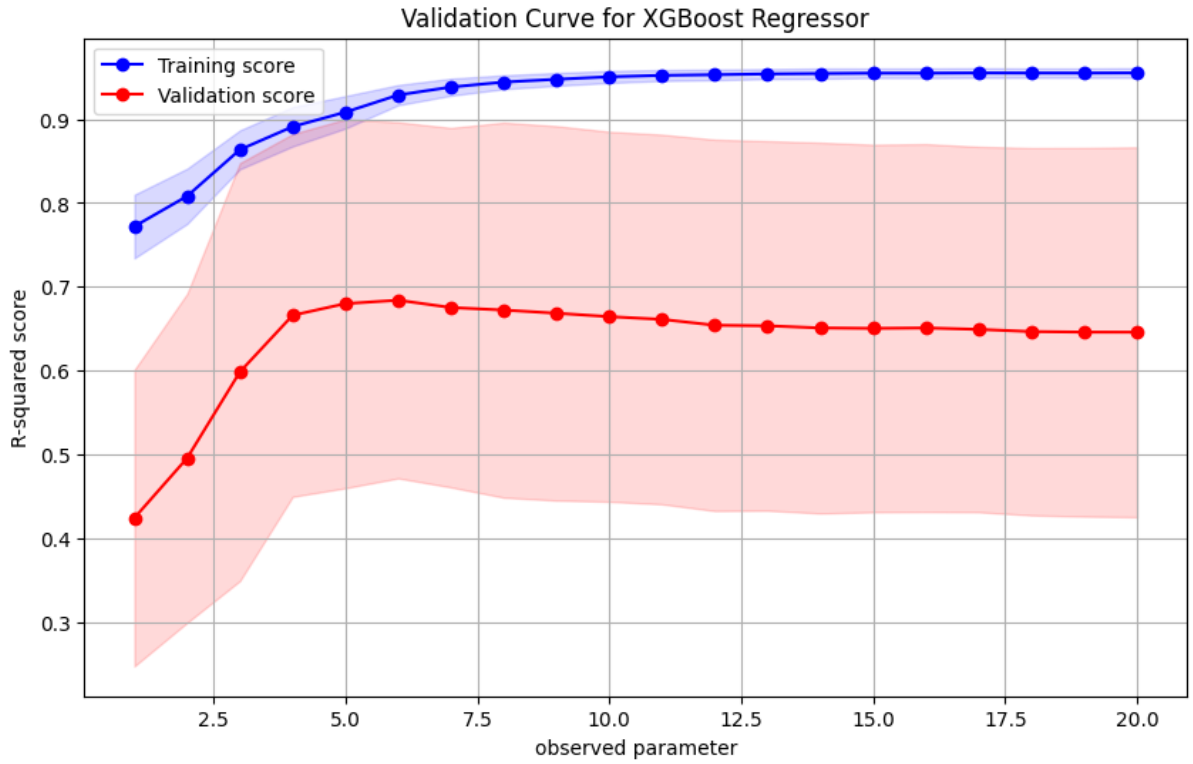
Param

```
    "subsample": 1.0,
```




Loss History
Training Loss
Validation Loss
Root Mean Squared Error (RMSE)
1.1
1.0
0.9
0.8
0.7
0.6
0
20
40
60
80
100
Boosting Round (Epoch)

Max_depth



No scaling

Poly degree 5

```
[0]   validation_0-rmse:135987.55982 validation_1-rmse:1479190.65646
[1]   validation_0-rmse:135932.81363 validation_1-rmse:1479182.47141
```

Eta 0.001

N_iter = 100

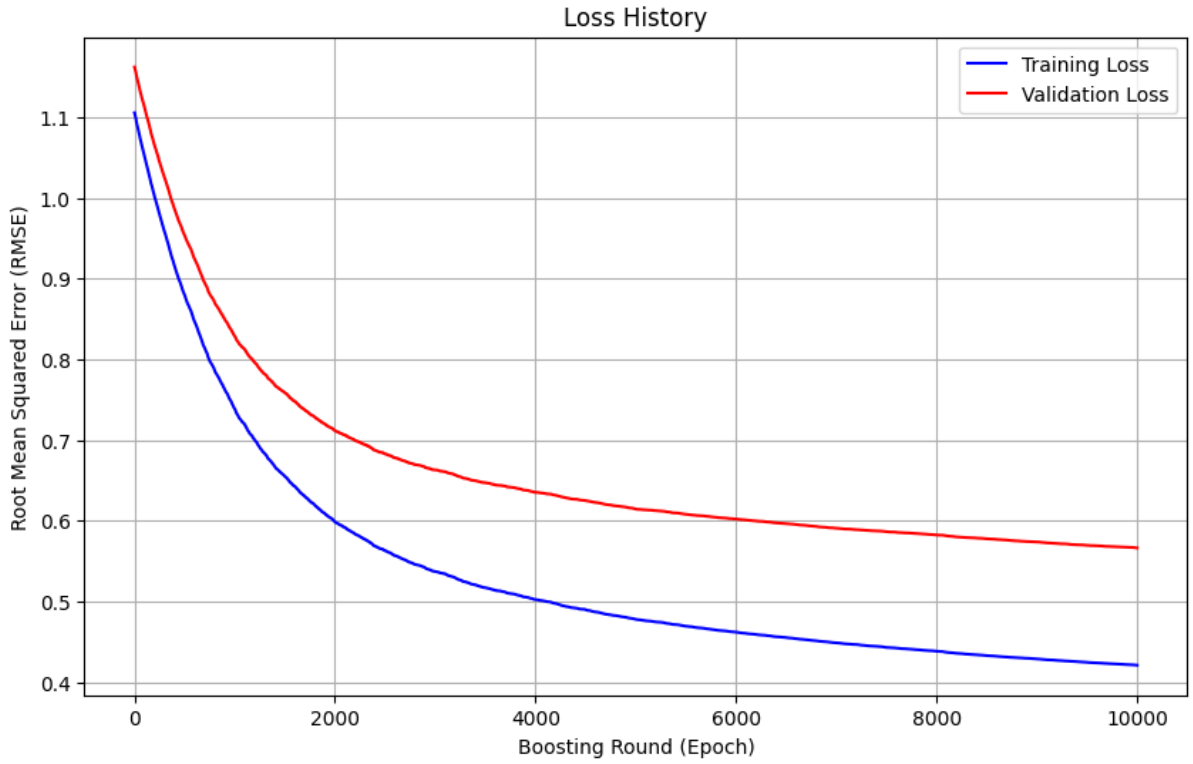
```
[98]  validation_0-rmse:1.10650      validation_1-rmse:1.16304
[99]  validation_0-rmse:1.10650      validation_1-rmse:1.16304
```

```
param = {
    "booster": "gbtree",
    "learning_rate":0.001,
    "tree_method": "hist",
    "max_depth": 5,
    "subsample": 0.1,
    "colsample_bytree": 0.1,
    "gamma": 0,
    "reg_alpha": 0.0,
```

```
    "reg_lambda": 0.0,
    "objective": "reg:squarederror",
    "min_child_weight": 0,
    "eval_metric": "rmse",
    # "early_stopping_rounds": 30,
    "random_state": 452,
    "validate_parameters": 1,
}

xgb_reg = XGB.XGBRegressor(n_estimators=10000, **param)
```

```
[0]   validation_0-rmse:1.10586      validation_1-rmse:1.16246
[1]   validation_0-rmse:1.10523      validation_1-rmse:1.16187
[2]   validation_0-rmse:1.10483      validation_1-rmse:1.16150
[3]   validation_0-rmse:1.10420      validation_1-rmse:1.16090
[4]   validation_0-rmse:1.10379      validation_1-rmse:1.16052
[5]   validation_0-rmse:1.10321      validation_1-rmse:1.15997
[6]   validation_0-rmse:1.10267      validation_1-rmse:1.15947
[7]   validation_0-rmse:1.10223      validation_1-rmse:1.15907
[8]   validation_0-rmse:1.10181      validation_1-rmse:1.15868
[9]   validation_0-rmse:1.10118      validation_1-rmse:1.15809
[10]  validation_0-rmse:1.10072      validation_1-rmse:1.15768
[11]  validation_0-rmse:1.10021      validation_1-rmse:1.15721
[12]  validation_0-rmse:1.09978      validation_1-rmse:1.15681
[13]  validation_0-rmse:1.09909      validation_1-rmse:1.15619
[14]  validation_0-rmse:1.09857      validation_1-rmse:1.15573
[15]  validation_0-rmse:1.09753      validation_1-rmse:1.15472
[16]  validation_0-rmse:1.09712      validation_1-rmse:1.15434
[17]  validation_0-rmse:1.09620      validation_1-rmse:1.15346
[18]  validation_0-rmse:1.09531      validation_1-rmse:1.15260
[19]  validation_0-rmse:1.09480      validation_1-rmse:1.15213
[20]  validation_0-rmse:1.09378      validation_1-rmse:1.15120
[21]  validation_0-rmse:1.09313      validation_1-rmse:1.15061
[22]  validation_0-rmse:1.09274      validation_1-rmse:1.15023
[23]  validation_0-rmse:1.09235      validation_1-rmse:1.14987
[24]  validation_0-rmse:1.09195      validation_1-rmse:1.14950
...
[9996]validation_0-rmse:0.42087      validation_1-rmse:0.56629
[9997]validation_0-rmse:0.42086      validation_1-rmse:0.56628
[9998]validation_0-rmse:0.42086      validation_1-rmse:0.56628
[9999]validation_0-rmse:0.42086      validation_1-rmse:0.56626
```

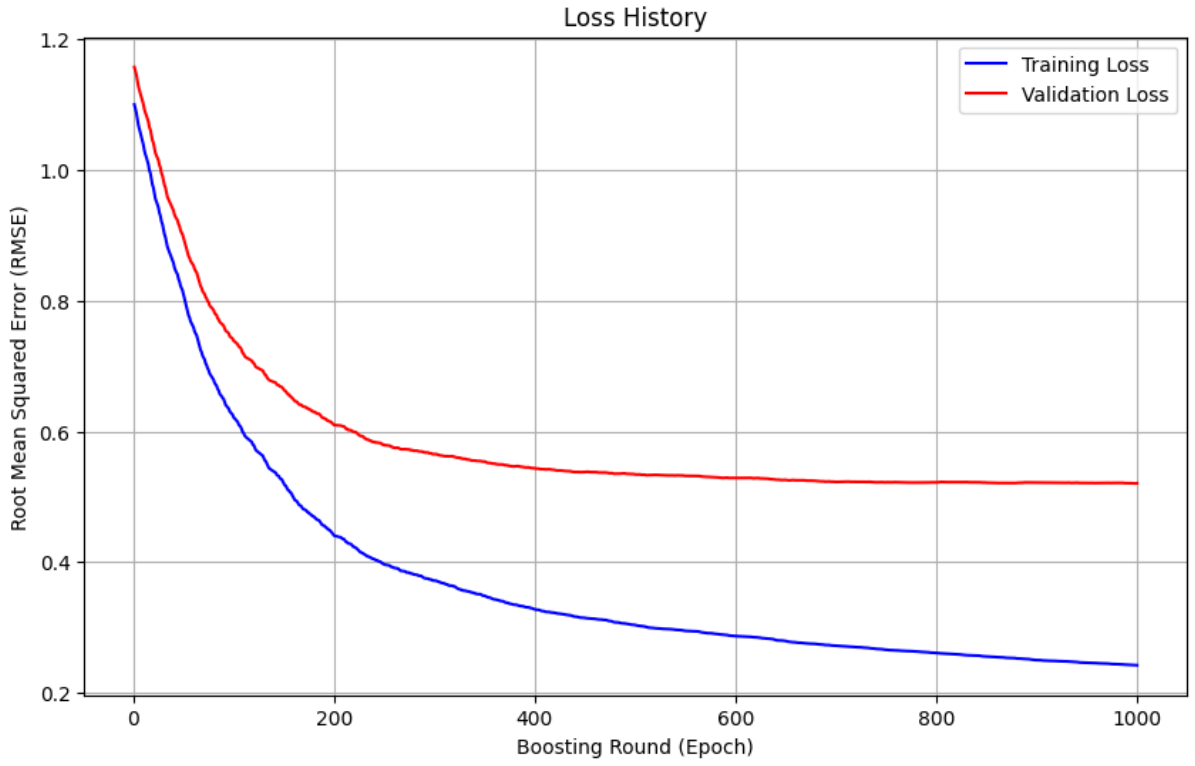


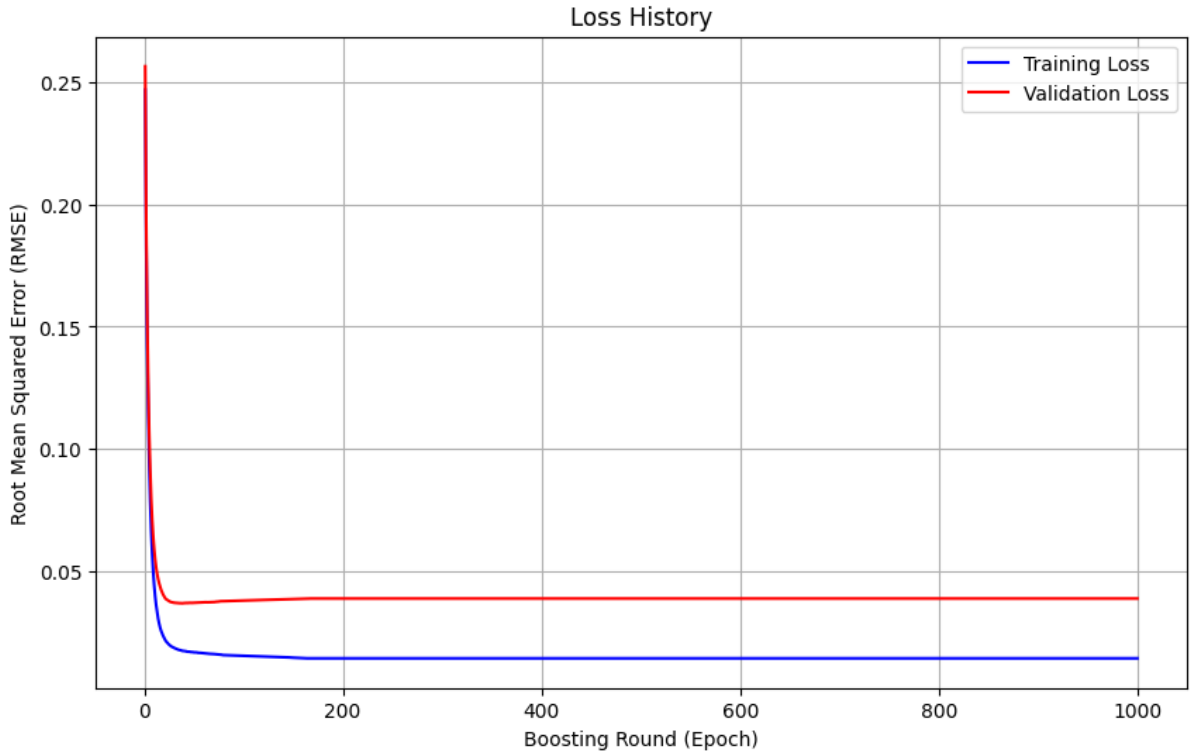

```python
param = {
    "booster": "gbtree",
    "learning_rate": 0.01,
    "tree_method": "hist",
    "max_depth": 5,
    "subsample": 0.5,
    "colsample_bytree": 0.5,
    "gamma": 0.0,
    "reg_alpha": 0.0,
    "reg_lambda": 0.0,
    "objective": "reg:squarederror",
    "min_child_weight": 0,
    "eval_metric": "rmse",
    # "early_stopping_rounds": 30,
    "random_state": 452,
    "validate_parameters": 10,
}

xgb_reg = XGB.XGBRegressor(n_estimators=1000, **param)
```

```
[0]   validation_0-rmse:1.09993      validation_1-rmse:1.15705
[1]   validation_0-rmse:1.09323      validation_1-rmse:1.15106
[2]   validation_0-rmse:1.08721      validation_1-rmse:1.14570
[3]   validation_0-rmse:1.07772      validation_1-rmse:1.13617
[4]   validation_0-rmse:1.06872      validation_1-rmse:1.12789
[5]   validation_0-rmse:1.06249      validation_1-rmse:1.12190
```

```
[6]   validation_0-rmse:1.05646      validation_1-rmse:1.11638
[7]   validation_0-rmse:1.05013      validation_1-rmse:1.11090
[8]   validation_0-rmse:1.04416      validation_1-rmse:1.10566
[9]   validation_0-rmse:1.03829      validation_1-rmse:1.10041
[10]  validation_0-rmse:1.02936      validation_1-rmse:1.09241
[11]  validation_0-rmse:1.02353      validation_1-rmse:1.08722
[12]  validation_0-rmse:1.01903      validation_1-rmse:1.08318
[13]  validation_0-rmse:1.01310      validation_1-rmse:1.07805
[14]  validation_0-rmse:1.00784      validation_1-rmse:1.07344
[15]  validation_0-rmse:0.99910      validation_1-rmse:1.06493
[16]  validation_0-rmse:0.99468      validation_1-rmse:1.06125
[17]  validation_0-rmse:0.98579      validation_1-rmse:1.05363
[18]  validation_0-rmse:0.97725      validation_1-rmse:1.04514
[19]  validation_0-rmse:0.97257      validation_1-rmse:1.04117
[20]  validation_0-rmse:0.96406      validation_1-rmse:1.03314
[21]  validation_0-rmse:0.95544      validation_1-rmse:1.02566
[22]  validation_0-rmse:0.95151      validation_1-rmse:1.02207
[23]  validation_0-rmse:0.94688      validation_1-rmse:1.01826
[24]  validation_0-rmse:0.94220      validation_1-rmse:1.01444
...
[996] validation_0-rmse:0.24250      validation_1-rmse:0.52035
[997] validation_0-rmse:0.24247      validation_1-rmse:0.52033
[998] validation_0-rmse:0.24244      validation_1-rmse:0.52033
[999] validation_0-rmse:0.24236      validation_1-rmse:0.52063
```





Test 1

```
param = {
    "objective": "reg:squarederror",
    "min_child_weight": 2,
    "base_score": 0.4,
    "eval_metric": "rmse",
    "random_state": 452
}
```

The best iteration and the R2 Scores is listed below'

	Results
Best_iteration	99.00
r2 train	0.99
r2 train after validation	0.68
r2 test	0.67
r2 test after validation	0.53

None
Mean Value of MPHE - Training = 0.14692556964733405 Mean Value of MPHE - Validation = 0.6118852138669802

Test 2

```
param = {
    "booster": "gbtree",
    "tree_method": "hist",
    "learning_rate": 0.01,
    "max_depth": 5,
    "subsample": 0.5,
    "sampling_method": "uniform",
    "colsample_bytree": 0.4,
    "gamma": 0.1,
    "reg_alpha": 0.001,
    "reg_lambda": 0.1,
    "objective": "reg:squarederror",
    "min_child_weight": 2,
    "base_score": 0.4,
    "eval_metric": "rmse",
    "random_state": 452,
    "validate_parameters": 1,
}
```

'The best iteration and the R2 Scores is listed below'

	Results
Best_iteration	99.00
r2 train	0.61
r2 train after validation	0.05
r2 test	0.51
r2 test after validation	0.19

None

Mean Value of MPHE - Training = 0.7999582952738752 Mean Value of MPHE - Validation = 0.8906350805804313

Test 3

```
param = {
    "booster": "gbtree",
    "learning_rate":0.01,
    "tree_method": "hist",
    "max_depth": 10,
    "subsample": 0.5,
    "colsample_bytree": 0.5,
    "sampling_method": "uniform",
    "gamma": 0.1,
    "reg_alpha": 5.5,
    "reg_lambda": 10.50,
    "objective": "reg:pseudohubererror",
    "min_child_weight": 54,
    "eval_metric": "mphe",
    "random_state": 15615,
    "huber_slope": np.linspace(start=10.0, stop=15.0, num=1, endpoint=True,
dtype=float).tolist()[0],
    "validate_parameters": 1,
}
```

'The best iteration and the R2 Scores is listed below'

	Results
Best_iteration	999.00
r2 train	0.85
r2 train after validation	0.66
r2 test	0.71
r2 test after validation	0.57

None

Mean Value of MPHE - Training = 0.11555719663035964 Mean Value of MPHE - Validation = 0.16038161183126046

Test 4

```
param = {
    "booster": "gbtree",
    "learning_rate":0.01,
    "tree_method": "hist",
    "max_depth": 20,
    "subsample": 1.0,
    "colsample_bytree": 1.0,
    "gamma": 0.1,
    "reg_alpha": 5.5,
    "reg_lambda": 5.50,
    "objective": "reg:pseudohubererror",
    "min_child_weight": 5,
```

```
    "eval_metric": "mphe",
    "random_state": 0,
    "huber_slope": np.linspace(start=10.0, stop=15.0, num=1, endpoint=True,
dtype=float).tolist()[0],
    "validate_parameters": 1,
}
```

'The best iteration and the R2 Scores is listed below'

Results	
Best_iteration	4,999.00
r2 train	0.95
r2 train after validation	0.68
r2 test	0.71
r2 test after validation	0.56

None

Mean Value of MPHE - Training = 0.02996989604642381 Mean Value of MPHE - Validation = 0.12104693883360358

Test 5

```
param = {
    "booster": "gbtree",
    "learning_rate":0.01,
    "tree_method": "hist",
    "max_depth": 20,
    "subsample": 1.0,
    "colsample_bytree": 1.0,
    "gamma": 0.1,
    "reg_alpha": 5.5,
    "reg_lambda": 10.50,
    "objective": "reg:pseudohubererror",
    "min_child_weight": 10,
    "eval_metric": "mphe",
    "random_state": 42,
    "huber_slope": np.linspace(start=10.0, stop=15.0, num=1, endpoint=True,
dtype=float).tolist()[0],
    "validate_parameters": 1,
}
usecols=["price", "plot_area", "habitable_surface", "bedroom_count",
"land_surface", "room_count"])
```

'The best iteration and the R2 Scores is listed below'

Results	
Best_iteration	999.00
r2 train	0.85

	Results
r2 train after validation	0.66
r2 test	0.71
r2 test after validation	0.57

None
Mean Value of MPHE - Training = 0.11555719663035964 Mean Value of MPHE - Validation = 0.16038161183126046

Test 6

```
param = {
    "booster": "gbtree",
    "learning_rate":0.01,
    "tree_method": "hist",
    "max_depth": 20,
    "subsample": 1.0,
    "colsample_bytree": 1.0,
    "gamma": 0.1,
    "reg_alpha": 10.5,
    "reg_lambda": 15.50,
    "objective": "reg:pseudohubererror",
    "min_child_weight": 10.50,
    "eval_metric": "mphe",
    "random_state": 42,
    "n_jobs":-1,
    "huber_slope": np.linspace(start=10.0, stop=15.0, num=1, endpoint=True,
dtype=float).tolist()[0],
    "validate_parameters": 1,
}

# load fit predict regression model
xgb_reg = XGB.XGBRegressor(n_estimators=5000, **param)
```

'The best iteration and the R2 Scores is listed below'

	Results
Best_iteration	4,999.00
r2 train	0.92
r2 train after 5 validation	0.68
r2 test	0.74
r2 test after 5 after validation	0.54

None

Mean Value of MPHE - Training = 0.0441294333405825 Mean Value of MPHE - Validation = 0.11487400781310625
0.2897039613704993