PMLB compare small

January 28, 2021

1 WIP

[47]:
```
import pmlb
import pandas as pd
import feyn
from sklearn.model_selection import train_test_split
```

[48]:
```
#for name in pmlb.regression_dataset_names:
#    df = pmlb.fetch_data(name, local_cache_dir="/tmp/pmlb_data")
#    print(f"('{name}', {len(df)}, {len(df.columns)}),")
```

[49]:
```
datasets = pd.DataFrame([
('1027_ESL', 488, 5),
('1028_SWD', 1000, 11),
('1029_LEV', 1000, 5),
('1030_ERA', 1000, 5),
('1089_USCrime', 47, 14),
('1096_FacultySalaries', 50, 5),
('1191_BNG_pbc', 1000000, 19),
('1193_BNG_lowbwt', 31104, 10),
('1196_BNG_pharynx', 1000000, 11),
('1199_BNG_echoMonths', 17496, 10),
('1201_BNG_breastTumor', 116640, 10),
('1203_BNG_pwLinear', 177147, 11),
('1595_poker', 1025010, 11),
('192_vineyard', 52, 3),
('195_auto_price', 159, 16),
('197_cpu_act', 8192, 22),
('201_pol', 15000, 49),
('207_autoPrice', 159, 16),
('210_cloud', 108, 6),
('215_2dplanes', 40768, 11),
('218_house_8L', 22784, 9),
('225_puma8NH', 8192, 9),
('227_cpu_small', 8192, 13),
('228_elusage', 55, 3),
('229_pwLinear', 200, 11),
```

```
('230_machine_cpu', 209, 7),
('294_satellite_image', 6435, 37),
('344_mv', 40768, 11),
('4544_GeographicalOriginalofMusic', 1059, 118),
('485_analcatdata_vehicle', 48, 5),
('503_wind', 6574, 15),
('505_tecator', 240, 125),
('519_vinnie', 380, 3),
('522_pm10', 500, 8),
('523_analcatdata_neavote', 100, 3),
('527_analcatdata_election2000', 67, 15),
('529_pollen', 3848, 5),
('537_houses', 20640, 9),
('542_pollution', 60, 16),
('547_no2', 500, 8),
('556_analcatdata_apnea2', 475, 4),
('557_analcatdata_apnea1', 475, 4),
('560_bodyfat', 252, 15),
('561_cpu', 209, 8),
('562_cpu_small', 8192, 13),
('564_fried', 40768, 11),
('573_cpu_act', 8192, 22),
('574_house_16H', 22784, 17),
('579_fri_c0_250_5', 250, 6),
('581_fri_c3_500_25', 500, 26),
('582_fri_c1_500_25', 500, 26),
('583_fri_c1_1000_50', 1000, 51),
('584_fri_c4_500_25', 500, 26),
('586_fri_c3_1000_25', 1000, 26),
('588_fri_c4_1000_100', 1000, 101),
('589_fri_c2_1000_25', 1000, 26),
('590_fri_c0_1000_50', 1000, 51),
('591_fri_c1_100_10', 100, 11),
('592_fri_c4_1000_25', 1000, 26),
('593_fri_c1_1000_10', 1000, 11),
('594_fri_c2_100_5', 100, 6),
('595_fri_c0_1000_10', 1000, 11),
('596_fri_c2_250_5', 250, 6),
('597_fri_c2_500_5', 500, 6),
('598_fri_c0_1000_25', 1000, 26),
('599_fri_c2_1000_5', 1000, 6),
('601_fri_c1_250_5', 250, 6),
('602_fri_c3_250_10', 250, 11),
('603_fri_c0_250_50', 250, 51),
('604_fri_c4_500_10', 500, 11),
('605_fri_c2_250_25', 250, 26),
('606_fri_c2_1000_10', 1000, 11),
```

```
('607_fri_c4_1000_50', 1000, 51),
('608_fri_c3_1000_10', 1000, 11),
('609_fri_c0_1000_5', 1000, 6),
('611_fri_c3_100_5', 100, 6),
('612_fri_c1_1000_5', 1000, 6),
('613_fri_c3_250_5', 250, 6),
('615_fri_c4_250_10', 250, 11),
('616_fri_c4_500_50', 500, 51),
('617_fri_c3_500_5', 500, 6),
('618_fri_c3_1000_50', 1000, 51),
('620_fri_c1_1000_25', 1000, 26),
('621_fri_c0_100_10', 100, 11),
('622_fri_c2_1000_50', 1000, 51),
('623_fri_c4_1000_10', 1000, 11),
('624_fri_c0_100_5', 100, 6),
('626_fri_c2_500_50', 500, 51),
('627_fri_c2_500_10', 500, 11),
('628_fri_c3_1000_5', 1000, 6),
('631_fri_c1_500_5', 500, 6),
('633_fri_c0_500_25', 500, 26),
('634_fri_c2_100_10', 100, 11),
('635_fri_c0_250_10', 250, 11),
('637_fri_c1_500_50', 500, 51),
('641_fri_c1_500_10', 500, 11),
('643_fri_c2_500_25', 500, 26),
('644_fri_c4_250_25', 250, 26),
('645_fri_c3_500_50', 500, 51),
('646_fri_c3_500_10', 500, 11),
('647_fri_c1_250_10', 250, 11),
('648_fri_c1_250_50', 250, 51),
('649_fri_c0_500_5', 500, 6),
('650_fri_c0_500_50', 500, 51),
('651_fri_c0_100_25', 100, 26),
('653_fri_c0_250_25', 250, 26),
('654_fri_c0_500_10', 500, 11),
('656_fri_c1_100_5', 100, 6),
('657_fri_c2_250_10', 250, 11),
('658_fri_c3_250_25', 250, 26),
('659_sleuth_ex1714', 47, 8),
('663_rabe_266', 120, 3),
('665_sleuth_case2002', 147, 7),
('666_rmftsa_ladata', 508, 11),
('678_visualizing_environmental', 111, 4),
('687_sleuth_ex1605', 62, 6),
('690_visualizing_galaxy', 323, 5),
('695_chatfield_4', 235, 13),
('706_sleuth_case1202', 93, 7),
```

```
('712_chscase_geyser1', 222, 3),
('banana', 5300, 3),
('titanic', 2201, 4),
],
columns=["name","n","fcount"])
```

[56]:
```
chosen_datasets = datasets[(datasets["n"]>=1000) & (datasets["fcount"]<12) ]
```

2 Winners: 200 samples

Linreg: - 1193_BNG_lowbwt (0.56) - 1199_BNG_echoMonths (0.401) - 529_pollen (0.786)

RF: - banana (.58, but QL did better than LR)

GB: - 215_2dplanes (.927, but QL did better than LR) - 218_house_8L (.491, but QL did better than LR) - 564_fried (.826, but QL did better than LR)

QL: - 1196_BNG_pharynx (.42) - 1203_BNG_pwLinear (.505) - 225_puma8NH (0.59) - 344_mv (.991) - 537_houses (.402, but LR did better after QL feature selection) - titanic (.266, well GB was a tie within rounding error)

None (R2 $<$ 0) - 1201_BNG_breastTumor - 1595_poker

3 Unility functions

[57]:
```
def get_pmlb_data(name, trainsize=250):
    df = pmlb.fetch_data(name, local_cache_dir="pmlb_data")
    return train_test_split(df,train_size=trainsize, random_state=0)

from sklearn import svm, tree, linear_model, ensemble

def X(df):
    return df.iloc[:,:-1]

def y(df):
    return df.iloc[:,-1]

def fit_and_r2_score(model, train, test):
    model.fit(X(train), y(train))
    return model.score(X(train), y(train)), model.score(X(test), y(test))
```

4 Compare to the usual suspects

[58]:
```
#results = pd.DataFrame(columns=["dataset", "model", "train_r2", "test_r2"])
#results = pd.read_csv("results-cache.csv")
```

[137]:
```
models = [
    linear_model.LinearRegression(),
    linear_model.Lasso(alpha=0.01, max_iter=10000),
    linear_model.Lasso(alpha=0.05, max_iter=10000),
    tree.DecisionTreeRegressor(max_depth=1),
    tree.DecisionTreeRegressor(max_depth=2),
    tree.DecisionTreeRegressor(max_depth=4),
    tree.DecisionTreeRegressor(max_depth=6),
    ensemble.RandomForestRegressor(),
    ensemble.GradientBoostingRegressor(),
    ensemble.GradientBoostingRegressor(n_estimators=50),
    ensemble.GradientBoostingRegressor(n_estimators=25)

]
```

[138]:
```
for name in chosen_datasets["name"]:
    print("Dataset:", name, end="")
    train, test = get_pmlb_data(name)
    print(" ... fetched", end="")
    for m in models:
        if ((results["dataset"]==name) & (results["model"]==str(m))).any():
            # Skip if already run
            continue
        r2_train, r2_test = fit_and_r2_score(m, train, test)
        results = results.append({"dataset": name, "model": str(m), "train_r2":␣
 ↪r2_train, "test_r2": r2_test}, ignore_index=True)

    print(" ... and fitted")
```

```
Dataset: 1028_SWD … fetched … and fitted
Dataset: 1029_LEV … fetched … and fitted
Dataset: 1030_ERA … fetched … and fitted
Dataset: 1193_BNG_lowbwt … fetched … and fitted
Dataset: 1196_BNG_pharynx … fetched … and fitted
Dataset: 1199_BNG_echoMonths … fetched … and fitted
Dataset: 1201_BNG_breastTumor … fetched … and fitted
Dataset: 1203_BNG_pwLinear … fetched … and fitted
Dataset: 1595_poker … fetched … and fitted
Dataset: 215_2dplanes … fetched … and fitted
Dataset: 218_house_8L … fetched … and fitted
Dataset: 225_puma8NH … fetched … and fitted
Dataset: 344_mv … fetched … and fitted
Dataset: 529_pollen … fetched … and fitted
Dataset: 537_houses … fetched … and fitted
Dataset: 564_fried … fetched … and fitted
Dataset: 593_fri_c1_1000_10 … fetched … and fitted
Dataset: 595_fri_c0_1000_10 … fetched … and fitted
```

```
Dataset: 599_fri_c2_1000_5 … fetched … and fitted
Dataset: 606_fri_c2_1000_10 … fetched … and fitted
Dataset: 608_fri_c3_1000_10 … fetched … and fitted
Dataset: 609_fri_c0_1000_5 … fetched … and fitted
Dataset: 612_fri_c1_1000_5 … fetched … and fitted
Dataset: 623_fri_c4_1000_10 … fetched … and fitted
Dataset: 628_fri_c3_1000_5 … fetched … and fitted
Dataset: banana … fetched … and fitted
Dataset: titanic … fetched … and fitted
```

[131]:
```
results
```

[131]:
```
                dataset                                          model  train_r2  \
0       1193_BNG_lowbwt                           LinearRegression()  0.601427
1       1193_BNG_lowbwt             Lasso(alpha=0.01, max_iter=10000)  0.601427
2       1193_BNG_lowbwt             Lasso(alpha=0.05, max_iter=10000)  0.601427
3       1193_BNG_lowbwt            DecisionTreeRegressor(max_depth=2)  0.575200
4       1193_BNG_lowbwt            DecisionTreeRegressor(max_depth=6)  0.771070
..                  …                                             …       …
388   612_fri_c1_1000_5  GradientBoostingRegressor(n_estimators=50)  0.973895
389  623_fri_c4_1000_10  GradientBoostingRegressor(n_estimators=50)  0.964780
390   628_fri_c3_1000_5  GradientBoostingRegressor(n_estimators=50)  0.967779
391              banana  GradientBoostingRegressor(n_estimators=50)  0.773114
392             titanic  GradientBoostingRegressor(n_estimators=50)  0.343677

      test_r2
0    0.557403
1    0.557408
2    0.557426
3    0.547522
4    0.377299
..        …
388  0.876299
389  0.844239
390  0.865130
391  0.533523
392  0.268468

[393 rows x 4 columns]
```

5 Fit a qgraph for each data set

[10]:
```
ql = feyn.QLattice()
```

[141]:
```
for name in chosen_datasets["name"]:
    edges = 7
```

```
    criterion = "bic"
    rs = 1
    key = f"QG-r{rs}-{criterion}-{edges}"
    print(key)

    if ((results["dataset"]==name) & (results["model"]==key)).any():
        # Skip if already run
        continue

    train, test = get_pmlb_data(name)
    ql.reset(rs)
    qg = ql.get_regressor(train.columns, train.columns[-1]).filter(feyn.filters.
↪MaxEdges(edges))
    #qg=qg.filter(feyn.filters.
↪Functions(["add","log","inverse","exp","sqrt","squared"]))

    for _ in range(50):
        qg.fit(train, threads=7, criterion=criterion)

        ql.update(qg.best())
        print(name)
        print("Train:\t", qg[0].r2_score(train), "\nTest:\t", qg[0].
↪r2_score(test))

    for _ in range(10000):
        qg[0].fit(train)

    results = results.append({"dataset": name, "model": key, "train_r2": qg[0].
↪r2_score(train), "test_r2": qg[0].r2_score(test)}, ignore_index=True)
```

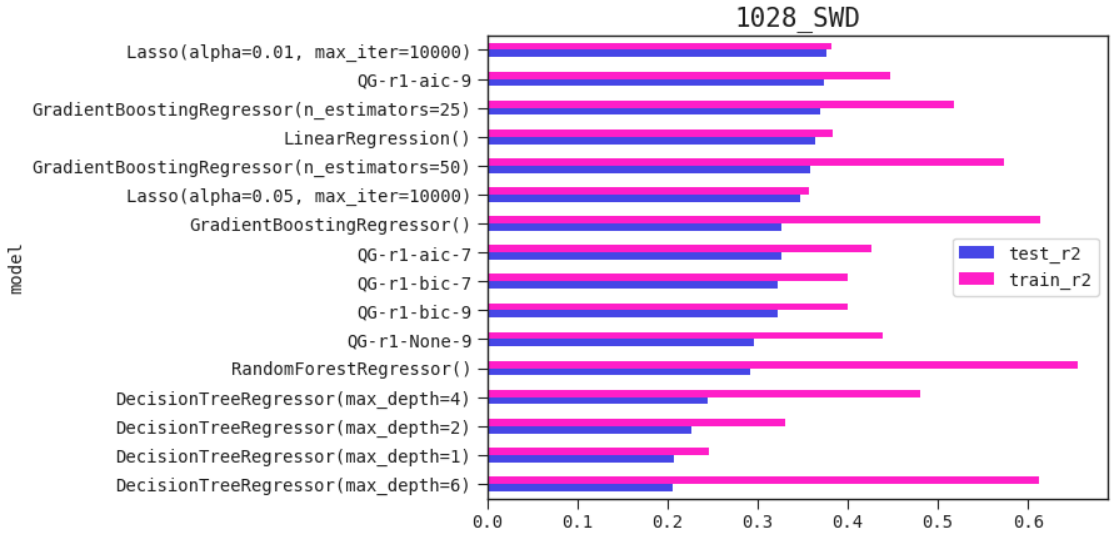
```
<IPython.core.display.HTML object>
```

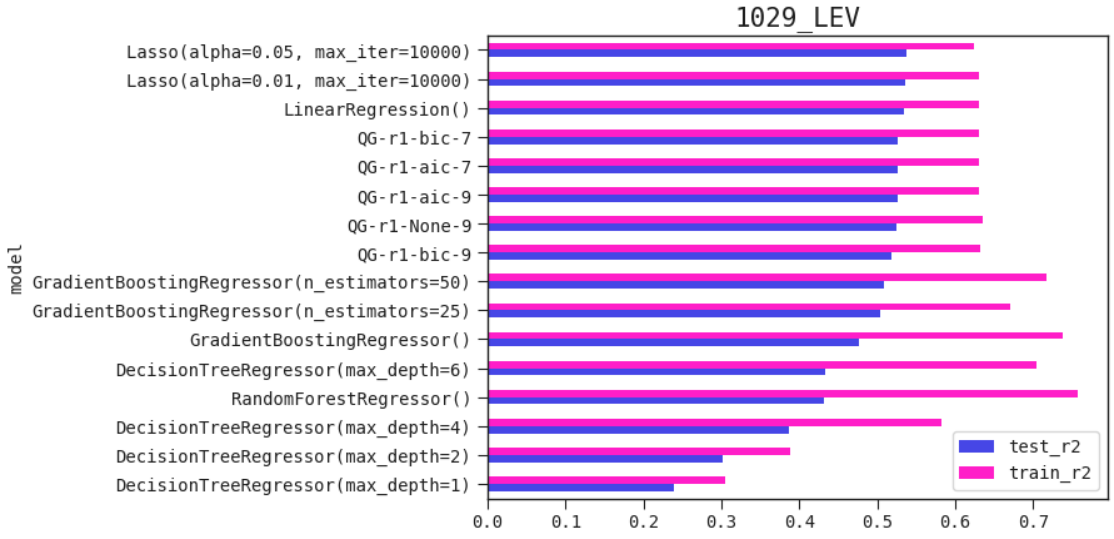
```
628_fri_c3_1000_5
Train:   0.8542383603561037
Test:    0.8292061065823529
QG-r1-bic-7
QG-r1-bic-7
```

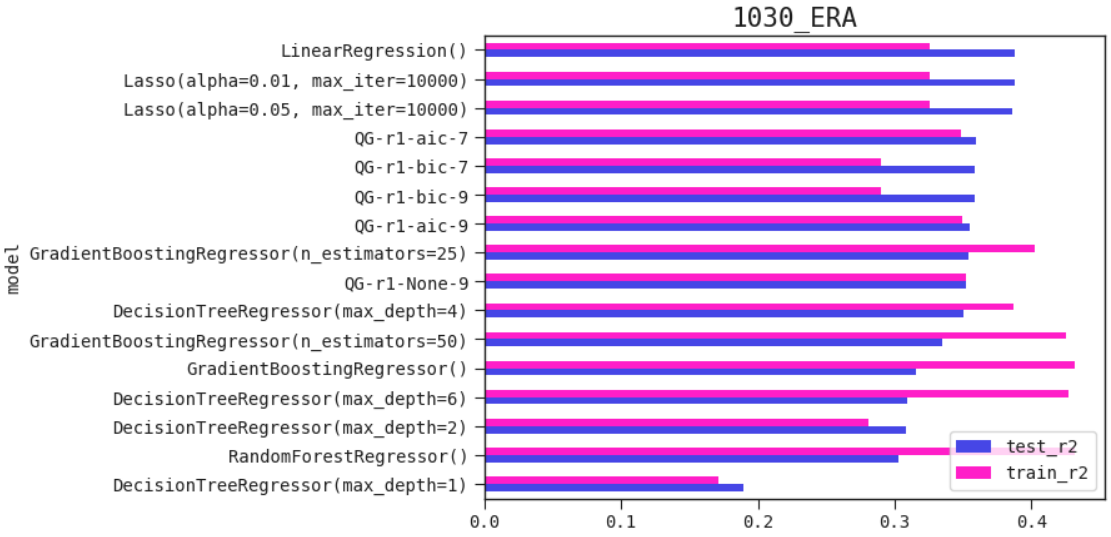
[142]:
```
for name in chosen_datasets["name"]:
    subres = results[(results["dataset"] == name) & (results["test_r2"]>-1) ].
↪sort_values(by="test_r2")
    subres.plot.barh(x="model", y=["test_r2","train_r2"], title=name)
```

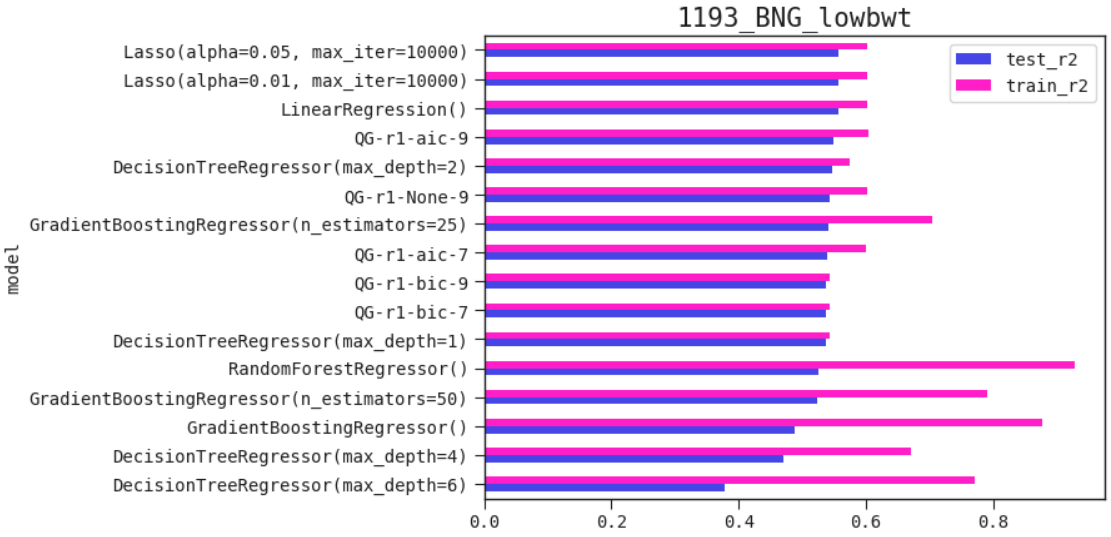
```
/home/cw/.local/share/virtualenvs/feyn-Y-4qiveF/lib/python3.8/site-
packages/pandas/plotting/_matplotlib/core.py:337: RuntimeWarning: More than 20
figures have been opened. Figures created through the pyplot interface
(`matplotlib.pyplot.figure`) are retained until explicitly closed and may
consume too much memory. (To control this warning, see the rcParam
```

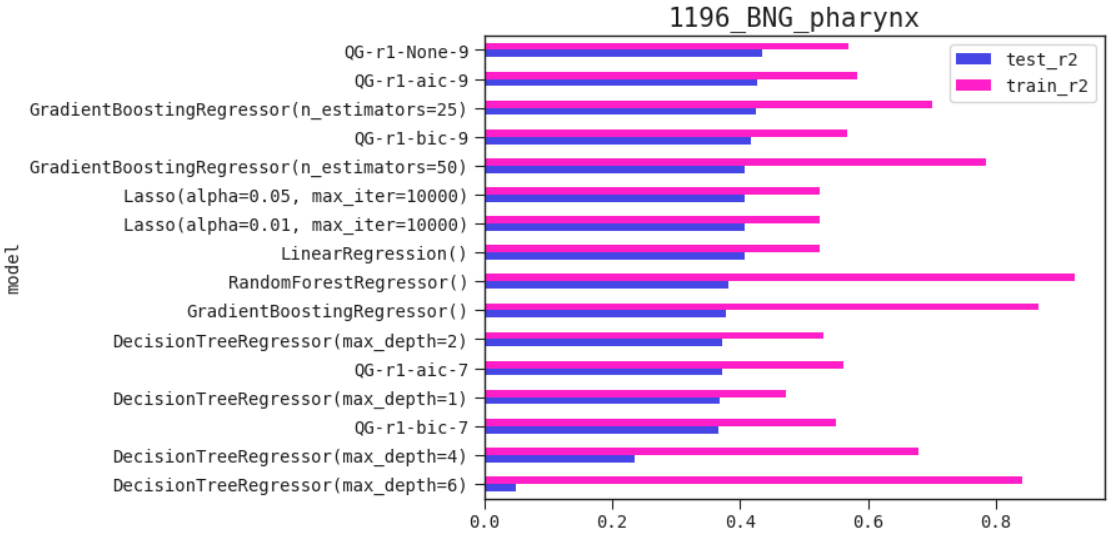
```
`figure.max_open_warning`).
  fig = self.plt.figure(figsize=self.figsize)
```

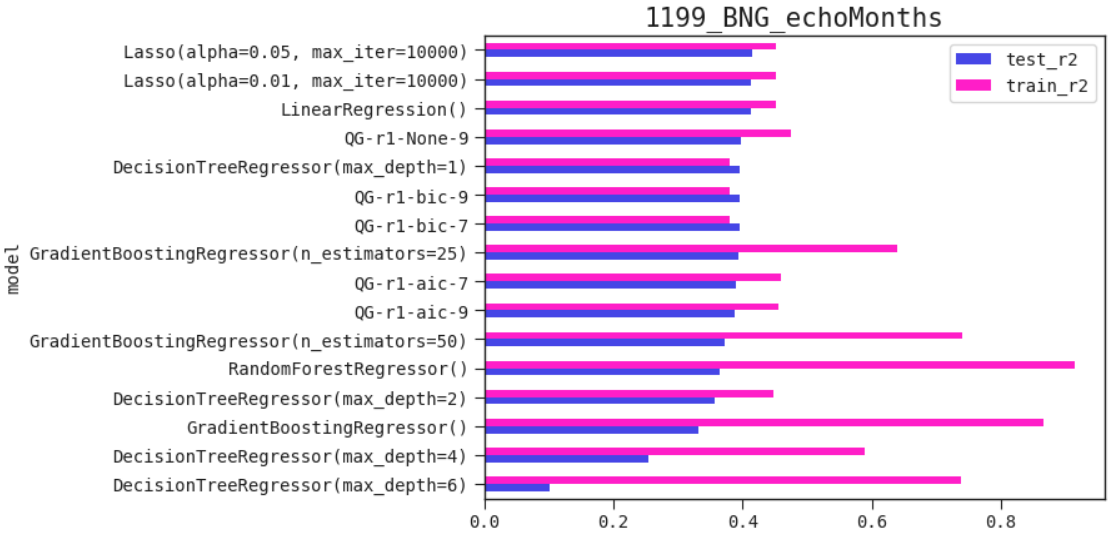


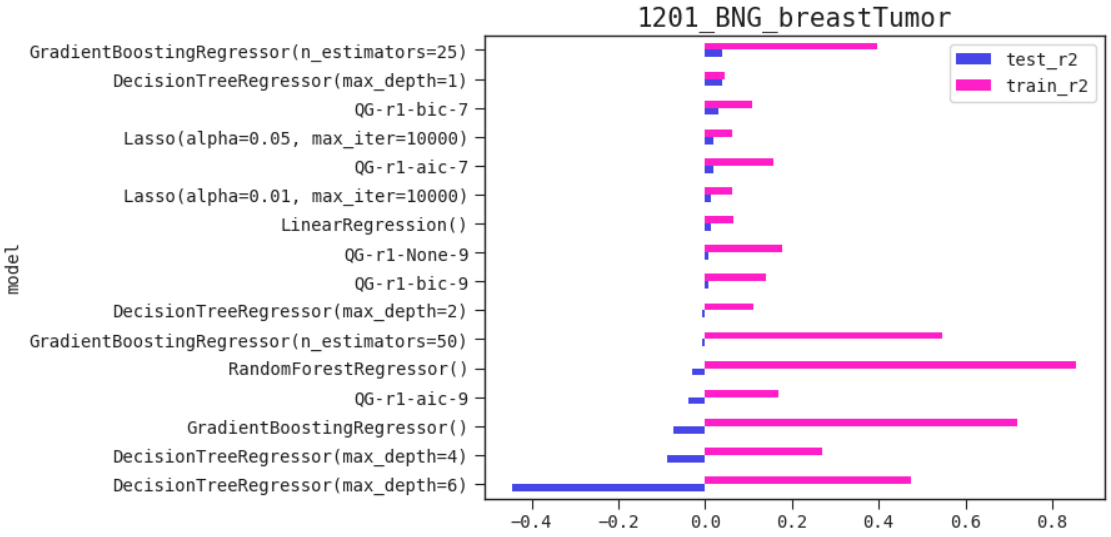


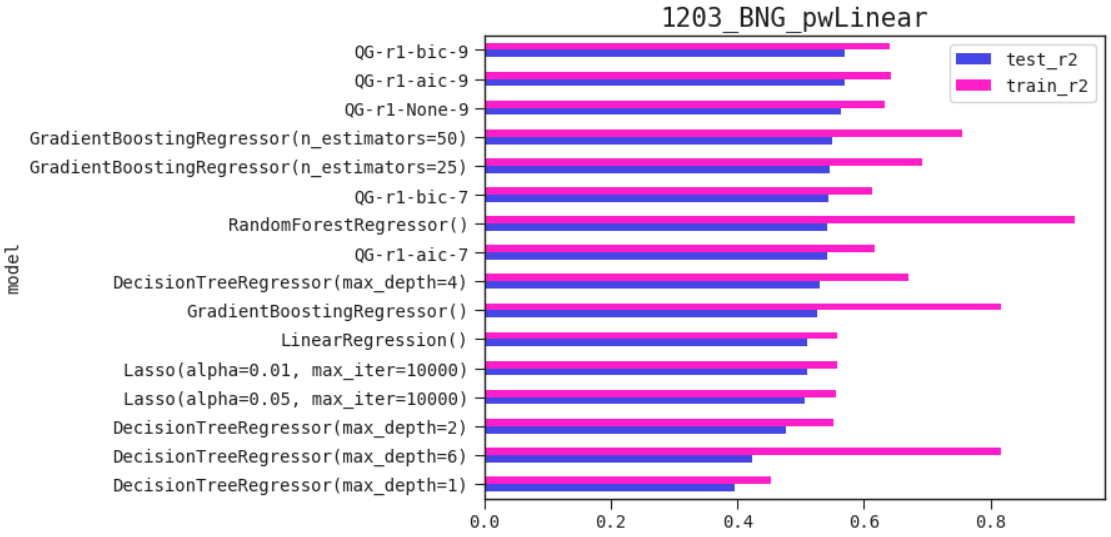

1030_ERA
model
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
QG-r1-aic-7
QG-r1-bic-7
QG-r1-bic-9
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=25)
QG-r1-None-9
DecisionTreeRegressor(max_depth=4)
GradientBoostingRegressor(n_estimators=50)
GradientBoostingRegressor()
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=2)
RandomForestRegressor()
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.1
0.2
0.3
0.4

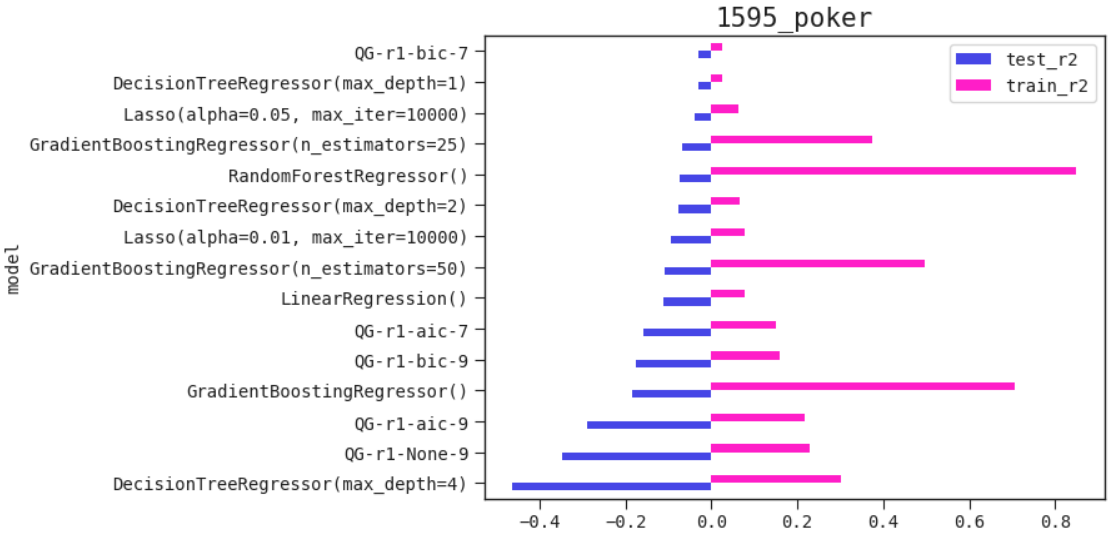

1193_BNG_lowbwt
model
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
QG-r1-aic-9
DecisionTreeRegressor(max_depth=2)
QG-r1-None-9
GradientBoostingRegressor(n_estimators=25)
QG-r1-aic-7
QG-r1-bic-9
QG-r1-bic-7
DecisionTreeRegressor(max_depth=1)
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=50)
GradientBoostingRegressor()
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8

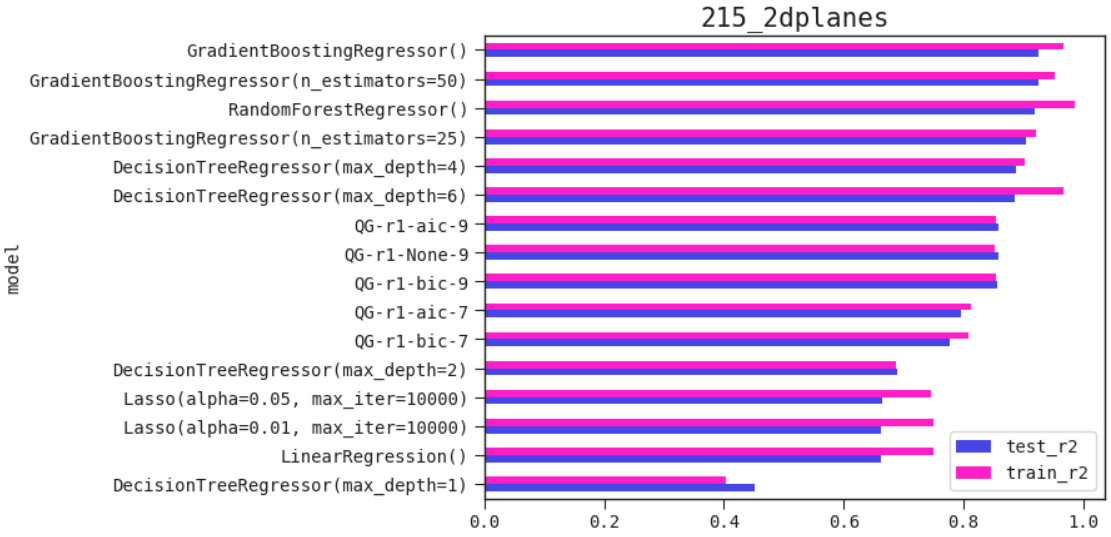

1196_BNG_pharynx
test_r2
train_r2
model
QG-r1-None-9
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=25)
QG-r1-bic-9
GradientBoostingRegressor(n_estimators=50)
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
RandomForestRegressor()
GradientBoostingRegressor()
DecisionTreeRegressor(max_depth=2)
QG-r1-aic-7
DecisionTreeRegressor(max_depth=1)
QG-r1-bic-7
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
0.0
0.2
0.4
0.6
0.8

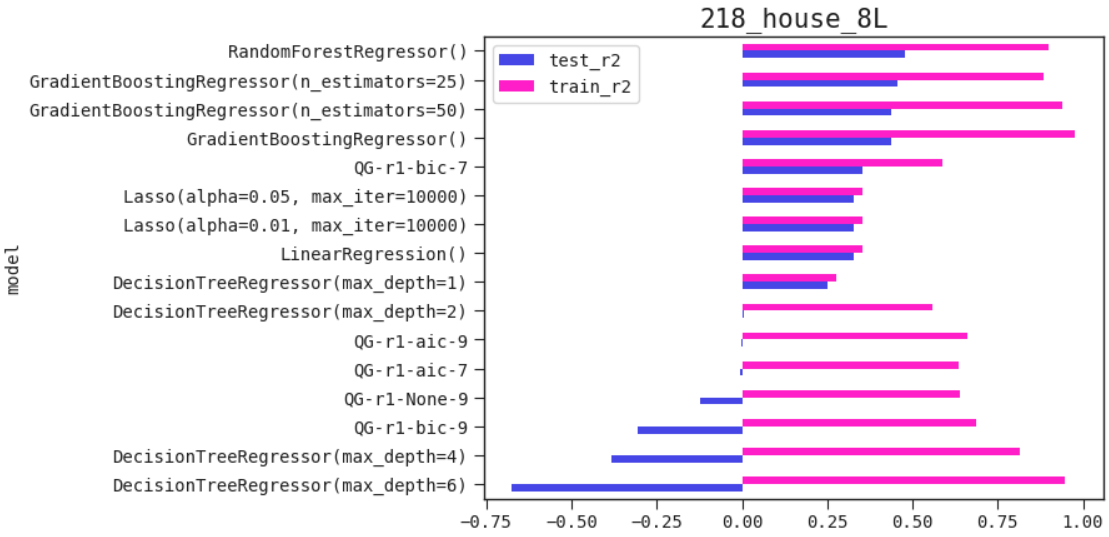

1199_BNG_echoMonths
test_r2
train_r2
model
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
QG-r1-None-9
DecisionTreeRegressor(max_depth=1)
QG-r1-bic-9
QG-r1-bic-7
GradientBoostingRegressor(n_estimators=25)
QG-r1-aic-7
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
DecisionTreeRegressor(max_depth=2)
GradientBoostingRegressor()
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
0.0
0.2
0.4
0.6
0.8

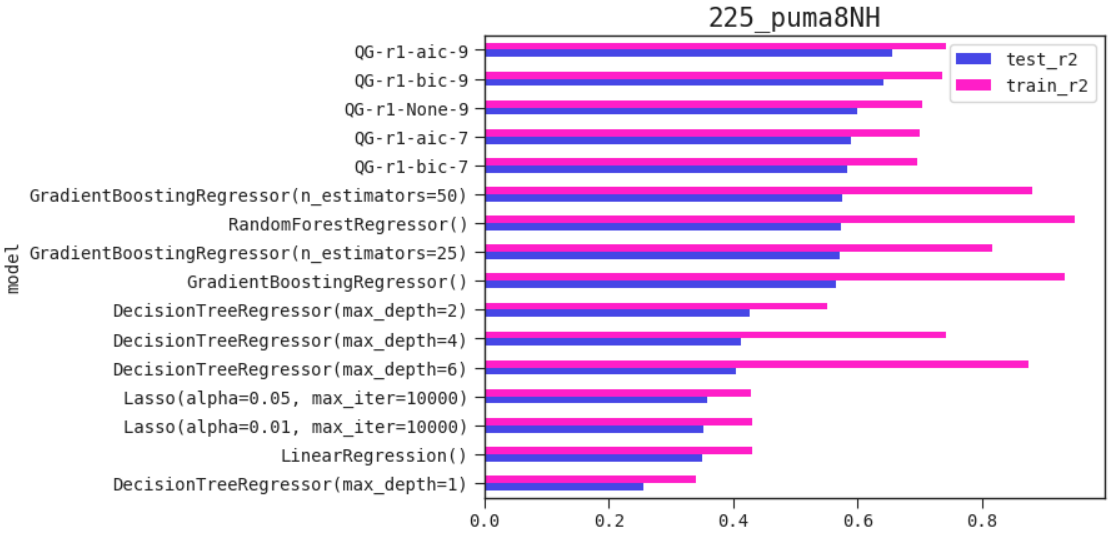

1201_BNG_breastTumor
test_r2
train_r2
model
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=1)
QG-r1-bic-7
Lasso(alpha=0.05, max_iter=10000)
QG-r1-aic-7
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
QG-r1-None-9
QG-r1-bic-9
DecisionTreeRegressor(max_depth=2)
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
QG-r1-aic-9
GradientBoostingRegressor()
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8

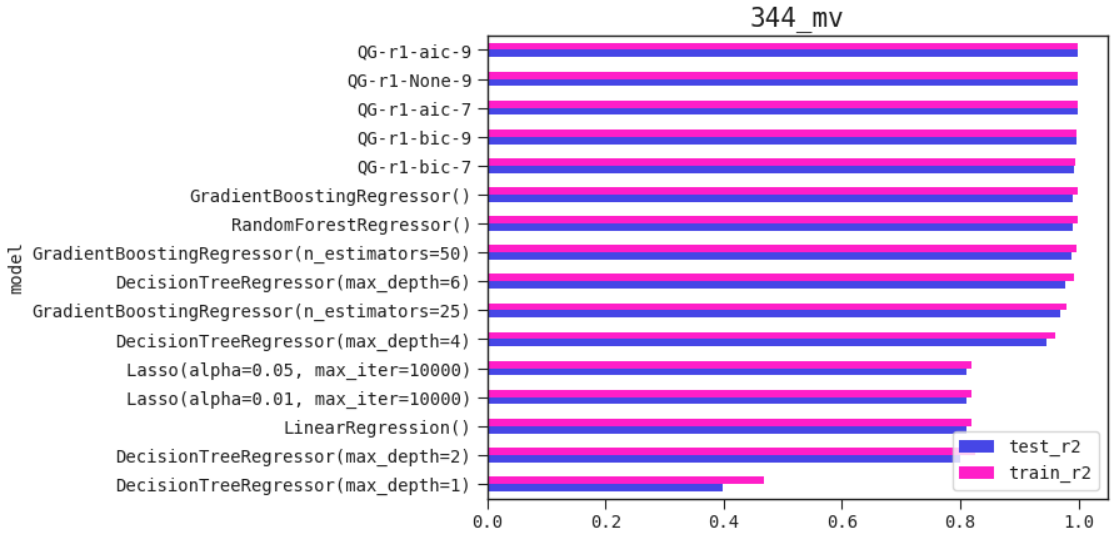

1203_BNG_pwLinear
test_r2
train_r2
model
QG-r1-bic-9
QG-r1-aic-9
QG-r1-None-9
GradientBoostingRegressor(n_estimators=50)
GradientBoostingRegressor(n_estimators=25)
QG-r1-bic-7
RandomForestRegressor()
QG-r1-aic-7
DecisionTreeRegressor(max_depth=4)
GradientBoostingRegressor()
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=1)
0.0
0.2
0.4
0.6
0.8

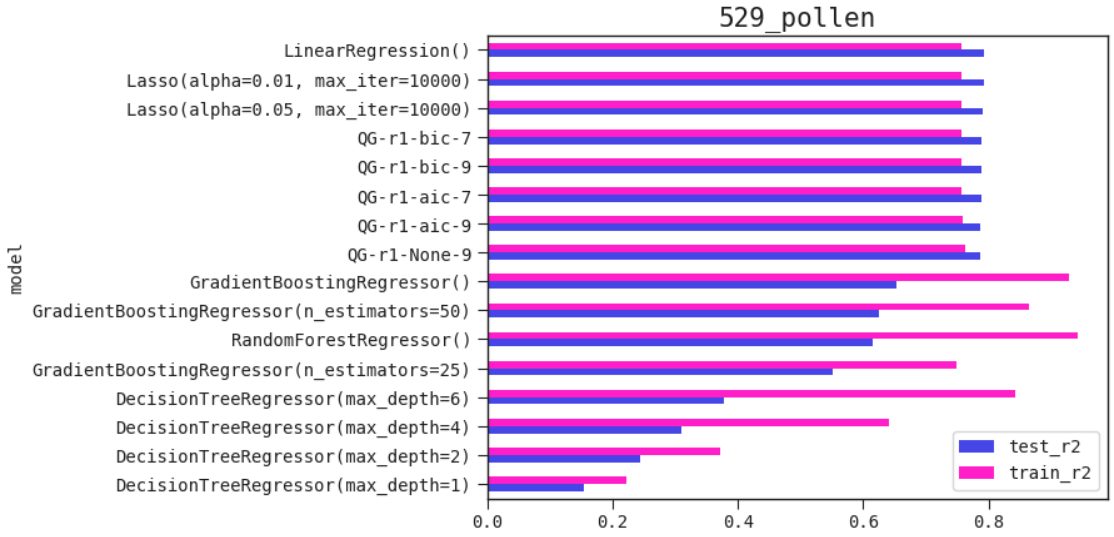

1595_poker
model
QG-r1-bic-7
DecisionTreeRegressor(max_depth=1)
Lasso(alpha=0.05, max_iter=10000)
GradientBoostingRegressor(n_estimators=25)
RandomForestRegressor()
DecisionTreeRegressor(max_depth=2)
Lasso(alpha=0.01, max_iter=10000)
GradientBoostingRegressor(n_estimators=50)
LinearRegression()
QG-r1-aic-7
QG-r1-bic-9
GradientBoostingRegressor()
QG-r1-aic-9
QG-r1-None-9
DecisionTreeRegressor(max_depth=4)
test_r2
train_r2
−0.4
−0.2
0.0
0.2
0.4
0.6
0.8

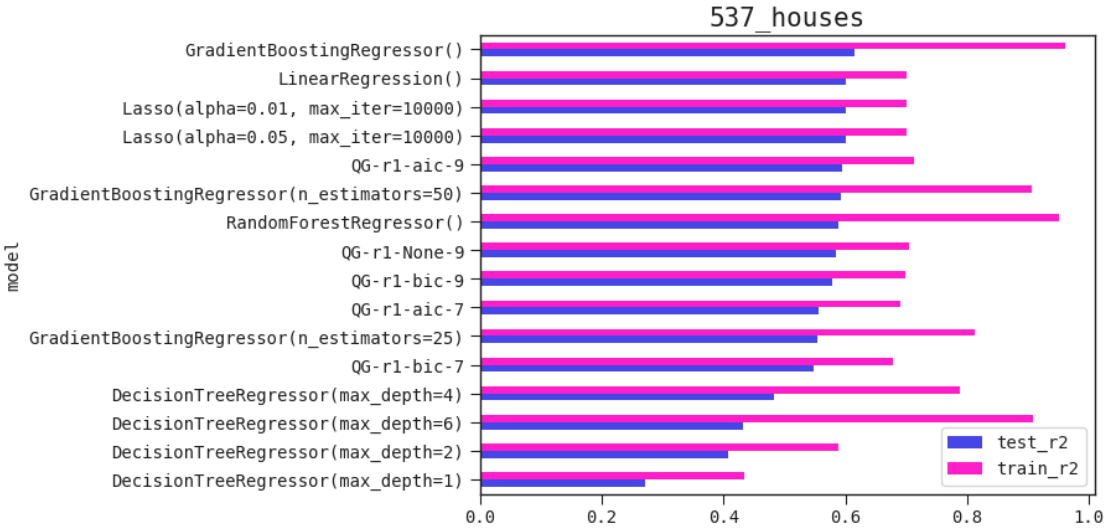

215_2dplanes
model
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
QG-r1-aic-9
QG-r1-None-9
QG-r1-bic-9
QG-r1-aic-7
QG-r1-bic-7
DecisionTreeRegressor(max_depth=2)
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

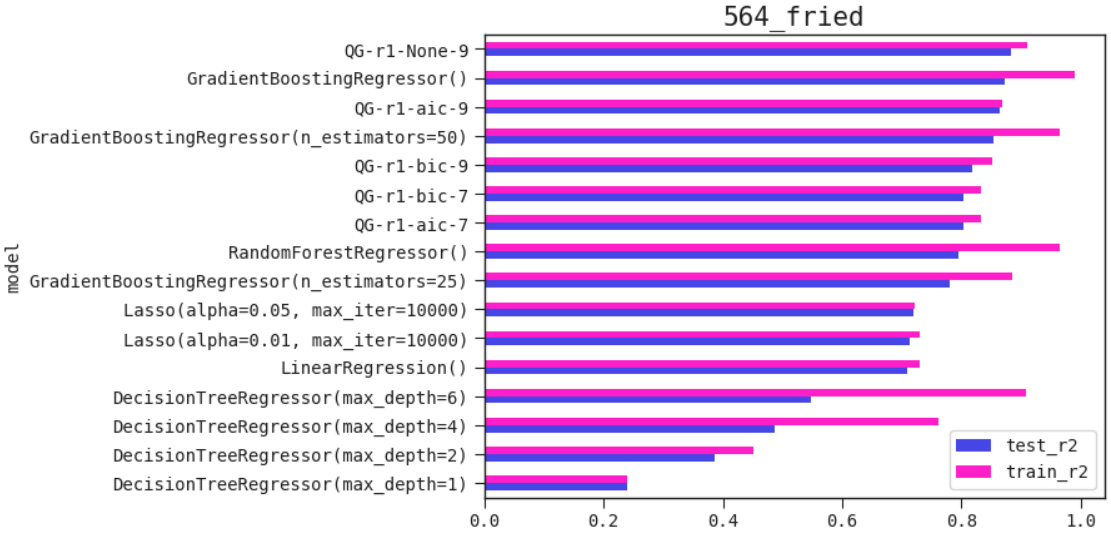

218_house_8L
model
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
GradientBoostingRegressor(n_estimators=50)
GradientBoostingRegressor()
QG-r1-bic-7
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
DecisionTreeRegressor(max_depth=1)
DecisionTreeRegressor(max_depth=2)
QG-r1-aic-9
QG-r1-aic-7
QG-r1-None-9
QG-r1-bic-9
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
test_r2
train_r2
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

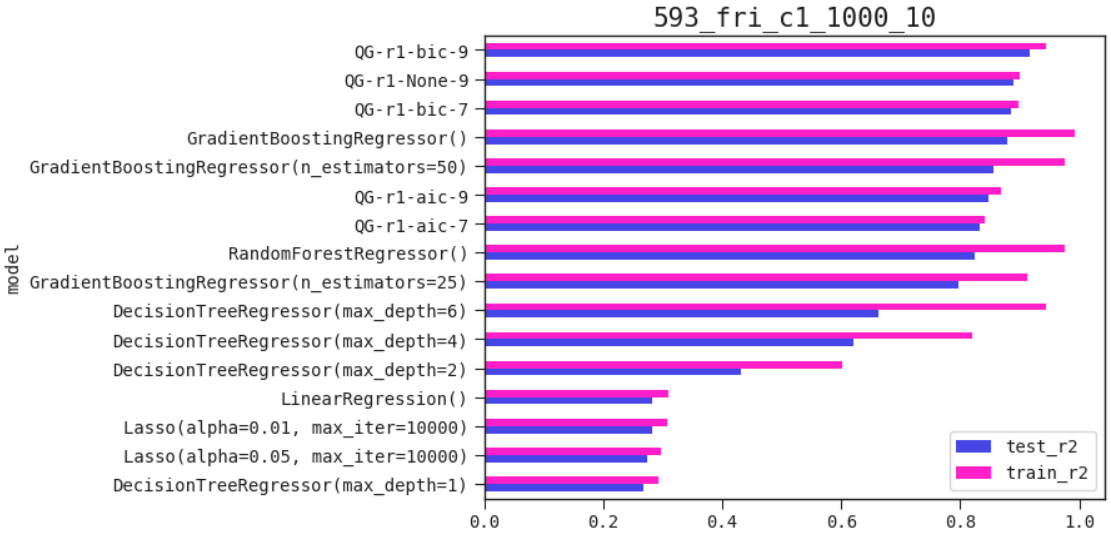

225_puma8NH
model
QG-r1-aic-9
QG-r1-bic-9
QG-r1-None-9
QG-r1-aic-7
QG-r1-bic-7
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
GradientBoostingRegressor()
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8

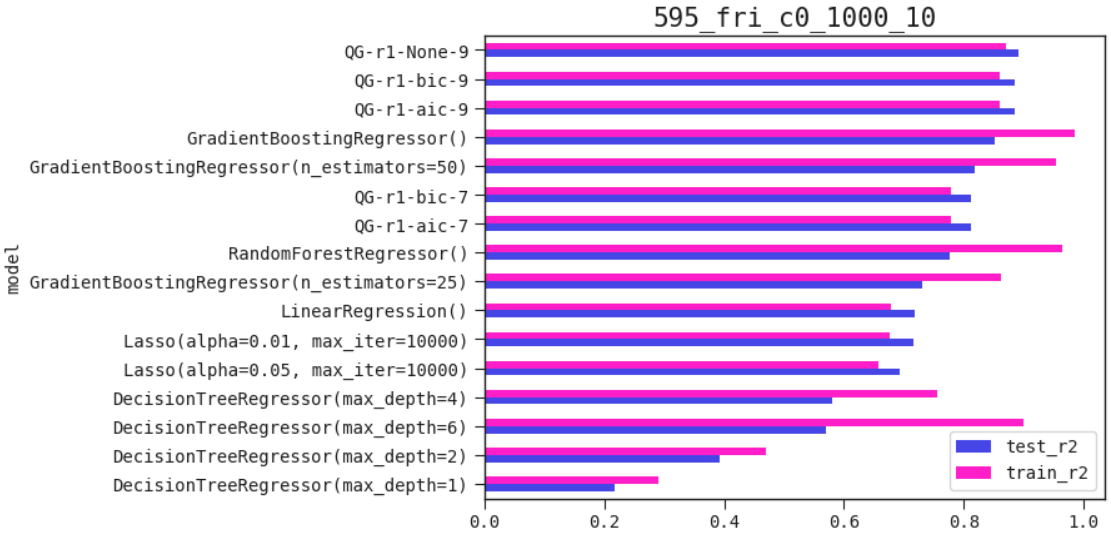

344_mv
model
QG-r1-aic-9
QG-r1-None-9
QG-r1-aic-7
QG-r1-bic-9
QG-r1-bic-7
GradientBoostingRegressor()
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=50)
DecisionTreeRegressor(max_depth=6)
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=4)
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

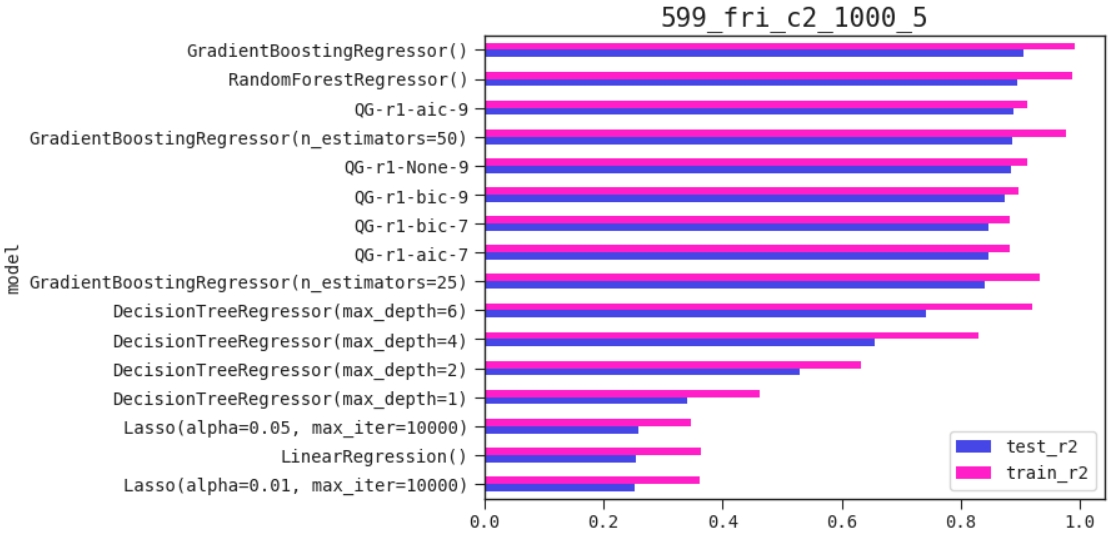

529_pollen
model
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
QG-r1-bic-7
QG-r1-bic-9
QG-r1-aic-7
QG-r1-aic-9
QG-r1-None-9
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8

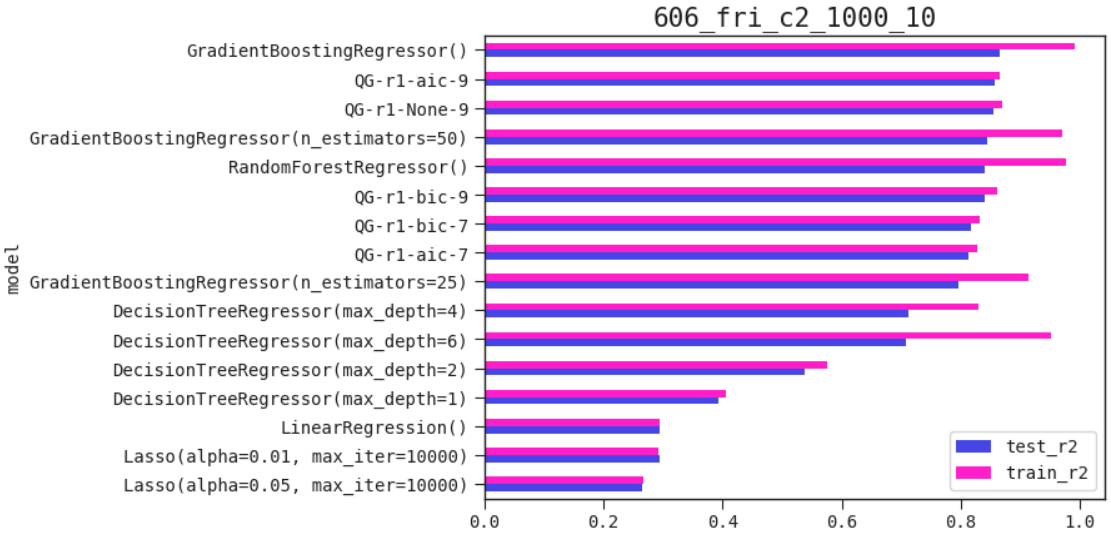

537_houses
GradientBoostingRegressor()
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
QG-r1-None-9
QG-r1-bic-9
QG-r1-aic-7
GradientBoostingRegressor(n_estimators=25)
QG-r1-bic-7
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
model
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

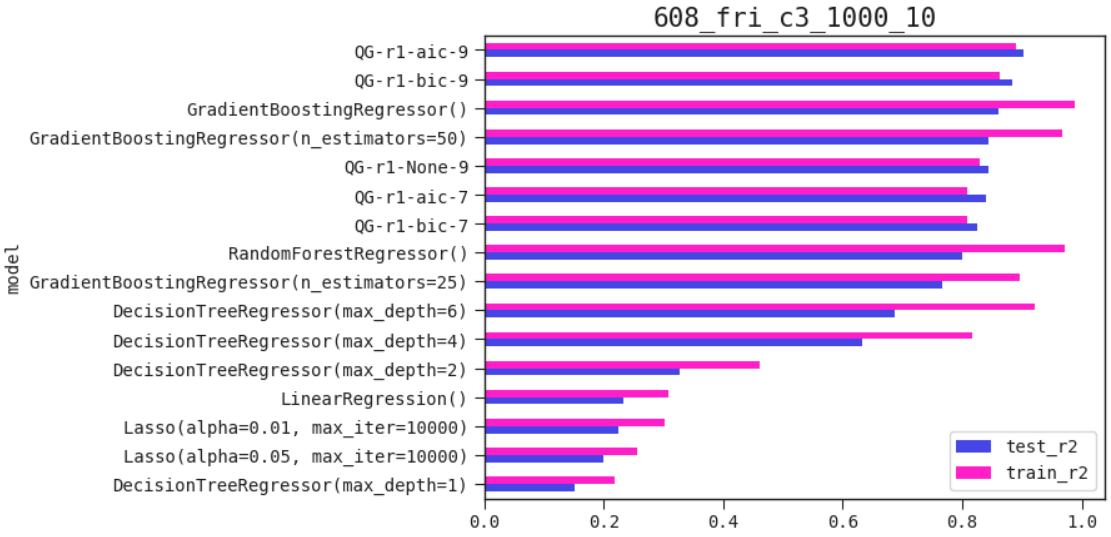

564_fried
QG-r1-None-9
GradientBoostingRegressor()
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=50)
QG-r1-bic-9
QG-r1-bic-7
QG-r1-aic-7
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
model
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

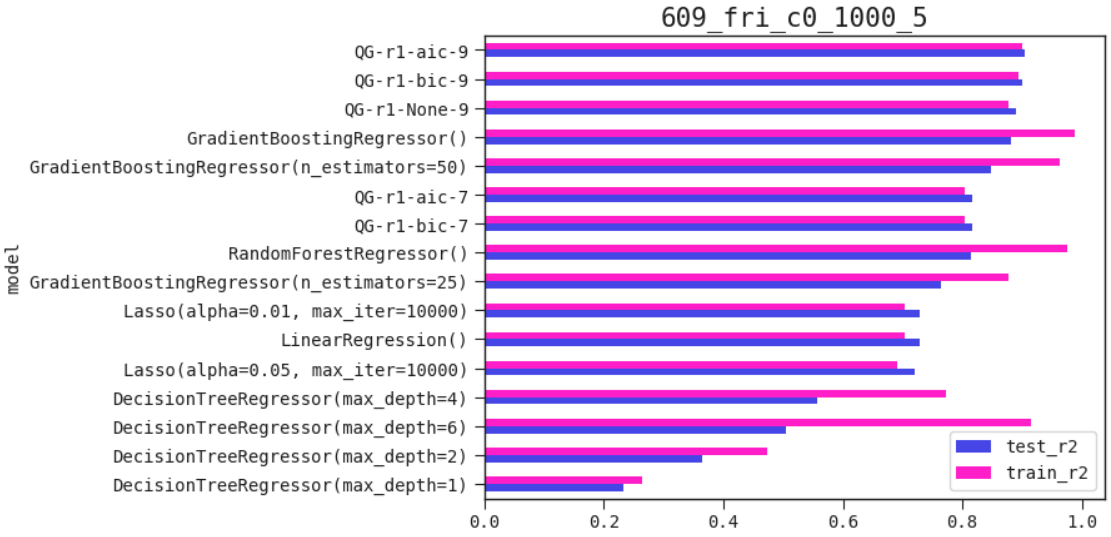

593_fri_c1_1000_10
QG-r1-bic-9
QG-r1-None-9
QG-r1-bic-7
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
QG-r1-aic-9
QG-r1-aic-7
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
DecisionTreeRegressor(max_depth=1)
model
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

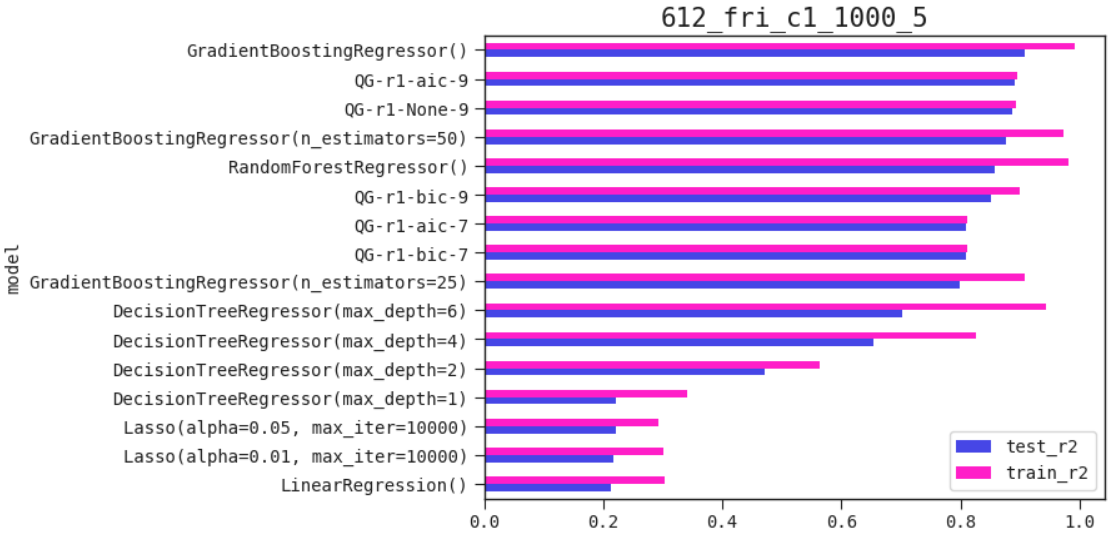

595_fri_c0_1000_10
QG-r1-None-9
QG-r1-bic-9
QG-r1-aic-9
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
QG-r1-bic-7
QG-r1-aic-7
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
model
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

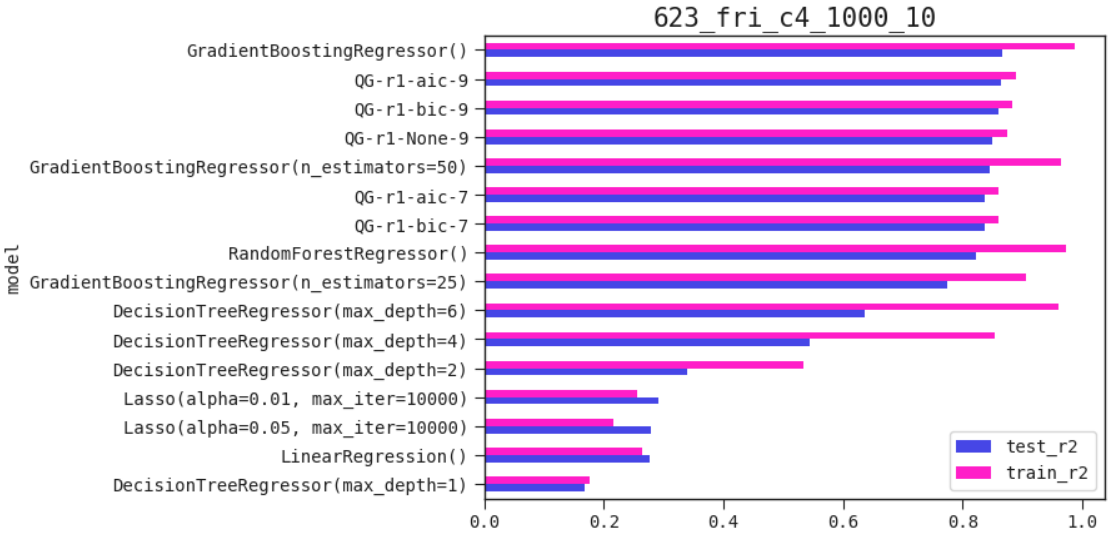

599_fri_c2_1000_5
model
GradientBoostingRegressor()
RandomForestRegressor()
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=50)
QG-r1-None-9
QG-r1-bic-9
QG-r1-bic-7
QG-r1-aic-7
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
Lasso(alpha=0.05, max_iter=10000)
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

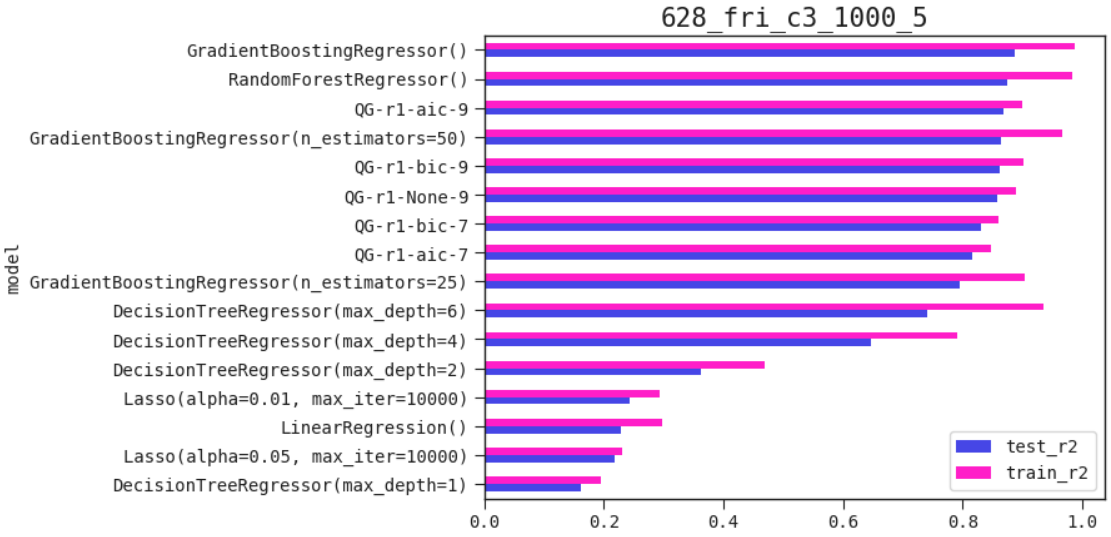

606_fri_c2_1000_10
model
GradientBoostingRegressor()
QG-r1-aic-9
QG-r1-None-9
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
QG-r1-bic-9
QG-r1-bic-7
QG-r1-aic-7
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

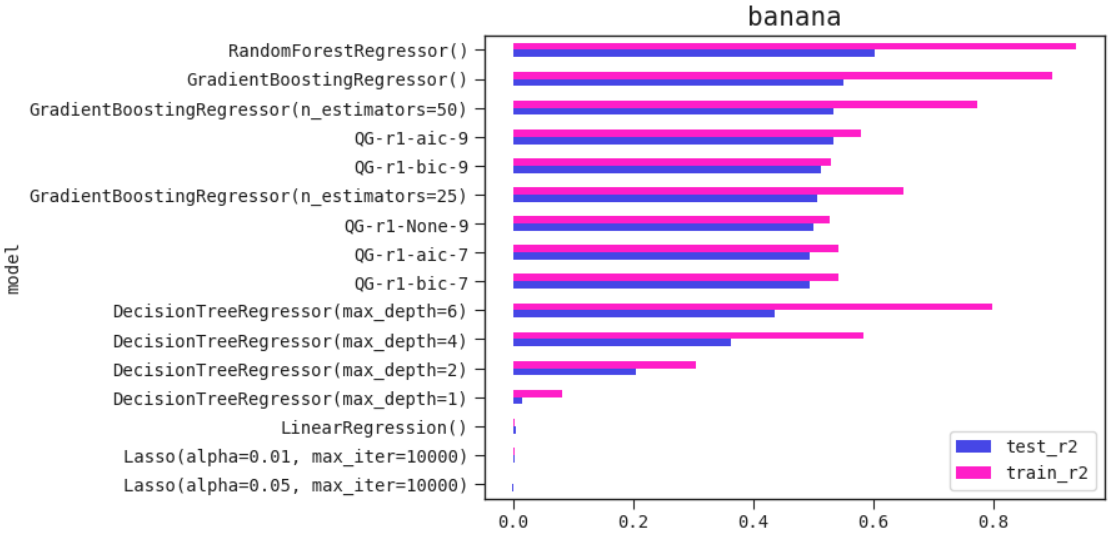

608_fri_c3_1000_10
model
QG-r1-aic-9
QG-r1-bic-9
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
QG-r1-None-9
QG-r1-aic-7
QG-r1-bic-7
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0

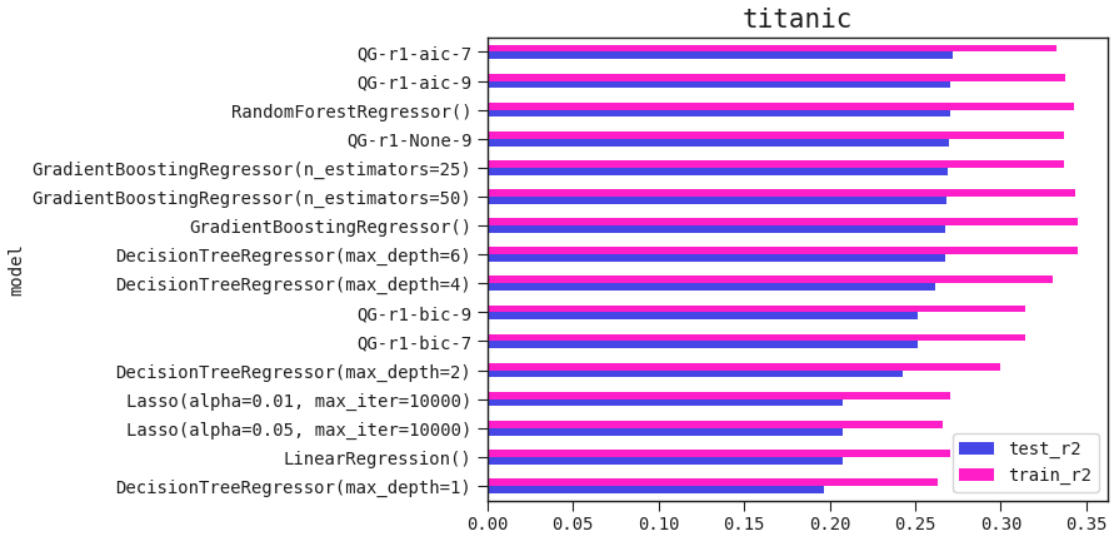

609_fri_c0_1000_5
model
QG-r1-aic-9
QG-r1-bic-9
QG-r1-None-9
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
QG-r1-aic-7
QG-r1-bic-7
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
Lasso(alpha=0.05, max_iter=10000)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0


612_fri_c1_1000_5
GradientBoostingRegressor()
QG-r1-aic-9
QG-r1-None-9
GradientBoostingRegressor(n_estimators=50)
RandomForestRegressor()
QG-r1-bic-9
QG-r1-aic-7
QG-r1-bic-7
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
Lasso(alpha=0.05, max_iter=10000)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
model
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0


623_fri_c4_1000_10
GradientBoostingRegressor()
QG-r1-aic-9
QG-r1-bic-9
QG-r1-None-9
GradientBoostingRegressor(n_estimators=50)
QG-r1-aic-7
QG-r1-bic-7
RandomForestRegressor()
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
LinearRegression()
DecisionTreeRegressor(max_depth=1)
model
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0


628_fri_c3_1000_5
model
GradientBoostingRegressor()
RandomForestRegressor()
QG-r1-aic-9
GradientBoostingRegressor(n_estimators=50)
QG-r1-bic-9
QG-r1-None-9
QG-r1-bic-7
QG-r1-aic-7
GradientBoostingRegressor(n_estimators=25)
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
Lasso(alpha=0.01, max_iter=10000)
LinearRegression()
Lasso(alpha=0.05, max_iter=10000)
DecisionTreeRegressor(max_depth=1)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8
1.0


banana
model
RandomForestRegressor()
GradientBoostingRegressor()
GradientBoostingRegressor(n_estimators=50)
QG-r1-aic-9
QG-r1-bic-9
GradientBoostingRegressor(n_estimators=25)
QG-r1-None-9
QG-r1-aic-7
QG-r1-bic-7
DecisionTreeRegressor(max_depth=6)
DecisionTreeRegressor(max_depth=4)
DecisionTreeRegressor(max_depth=2)
DecisionTreeRegressor(max_depth=1)
LinearRegression()
Lasso(alpha=0.01, max_iter=10000)
Lasso(alpha=0.05, max_iter=10000)
test_r2
train_r2
0.0
0.2
0.4
0.6
0.8



[147]:
```
results.to_csv("results-cache.csv", index=False)
```

[134]:
```
---- EXPERIMENTAL_ZONE
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
<ipython-input-134-3fa28fb7280c> in <module>
----> 1 ---- EXPERIMENTAL_ZONE

NameError: name 'EXPERIMENTAL_ZONE' is not defined
```

[145]:
```
res = results #[(results["model"]!="QG-r1-None-9")]
```

[149]:
```
scores = pd.DataFrame({"model": res["model"].unique(), "score": 0}).
↪set_index("model")

for name in res["dataset"].unique():
    subset = res[res["dataset"]==name].sort_values(by="test_r2",␣
↪ascending=False)
    for rank in range(5):
        m = subset.iloc[rank].model
        r2 = subset.iloc[rank].test_r2
        points = 5-rank
        scores.loc[m,"score"]+=1

scores.sort_values(by="score", ascending=False)
```

[149]:

model	score
QG-r1-aic-9	19
GradientBoostingRegressor(n_estimators=50)	16
QG-r1-None-9	15
GradientBoostingRegressor()	14
QG-r1-bic-9	13
RandomForestRegressor()	9
Lasso(alpha=0.05, max_iter=10000)	8
GradientBoostingRegressor(n_estimators=25)	8
LinearRegression()	7
Lasso(alpha=0.01, max_iter=10000)	7
QG-r1-bic-7	7
QG-r1-aic-7	7
DecisionTreeRegressor(max_depth=1)	3
DecisionTreeRegressor(max_depth=2)	1
DecisionTreeRegressor(max_depth=4)	1
DecisionTreeRegressor(max_depth=6)	0

[ ]:

[154]:
```
scores = pd.DataFrame({"model": res["model"].unique(), "score": 0}).
 ↪set_index("model")
res = results[(results["model"]=="QG-r1-aic-9")
              |(results["model"]=="GradientBoostingRegressor(n_estimators=50)")
              |(results["model"]=="RandomForestRegressor()")
              |(results["model"]=="Lasso(alpha=0.05, max_iter=10000)")
             ]

for name in res["dataset"].unique():
    subset = res[res["dataset"]==name].sort_values(by="test_r2",␣
 ↪ascending=False)
    for rank in range(1):
        m = subset.iloc[rank].model
        r2 = subset.iloc[rank].test_r2
        points = 1-rank
        scores.loc[m,"score"]+=1

scores.sort_values(by="score", ascending=False)
```

[154]:

model	score
QG-r1-aic-9	13
Lasso(alpha=0.05, max_iter=10000)	8
RandomForestRegressor()	4
GradientBoostingRegressor(n_estimators=50)	2

[ ]: