

regression_tree_testing

December 14, 2025

1 Testing Notebook - Regression Tree

TODO: - [] create random regressions datasets for testing - [] run training and prediction on these datasets and collect statistics - [] same as 3 but with the scikit regression tree - [] try a bunch of different settings, most likely use grid search with cross validation - [] visualize the results

```
[40]: from scripts.regression_tree import RegressionTree
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split, cross_validate
from sklearn.datasets import make_regression
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
```

```
RANDOM_STATE = 42
```

1.1 Helper functions

```
[57]: """Generate synthetic regression data using scikit-learn's make_regression_
↪function."""
def generate_regression_data(n_samples=100, n_features=1, noise=0.1,
↪random_state=RANDOM_STATE):
    X, y = make_regression(n_samples=n_samples, n_features=n_features,
↪noise=noise, random_state=random_state)
    print("Generated regression data with {} samples and {} features.".
↪format(n_samples, n_features))
    print("First 5 samples of X:\n", X[:5])
    print("First 5 samples of y:\n", y[:5])
    return X, y

"""Evaluate a regression model using cross-validation and return performance_
↪metrics."""
def evaluate_model(model, X, y, cv=5):
    cv_results = cross_validate(
        model,
        X,
```

```

    y,
    cv=cv,
    scoring=["r2", "neg_mean_squared_error", "neg_mean_absolute_error"],
    return_train_score=False,
)

return {
    "r2_mean": np.mean(cv_results["test_r2"]),
    "r2_std": np.std(cv_results["test_r2"]),
    "mse_mean": -np.mean(cv_results["test_neg_mean_squared_error"]),
    "mae_mean": -np.mean(cv_results["test_neg_mean_absolute_error"]),
}

```

1.2 Definition of testing configurations

```

[58]: TESTING_CONFIGS = [
    {
        "description": "Basic regression data with low noise",
        "n_samples": 100,
        "n_features": 1,
        "noise": 0.1
    },
    {
        "description": "Regression data with higher noise",
        "n_samples": 100,
        "n_features": 1,
        "noise": 10.0
    },
    {
        "description": "Regression data with multiple features",
        "n_samples": 200,
        "n_features": 5,
        "noise": 5.0
    },
    {
        "description": "Larger dataset with moderate noise",
        "n_samples": 5000,
        "n_features": 3,
        "noise": 2.0
    },
    {
        "description": "Large dataset with low noise",
        "n_samples": 10000,
        "n_features": 2,
        "noise": 0.5
    },
    {

```

```

        "description": "Large dataset with high noise",
        "n_samples": 10000,
        "n_features": 2,
        "noise": 20.0
    },
    {
        "description": "Moderate dataset with many features",
        "n_samples": 1000,
        "n_features": 50,
        "noise": 5.0
    }
]

```

```

HYPERPARAM_CONFIGS = [
    {
        "name": "shallow",
        "max_depth": 2,
        "min_samples_leaf": 1,
        "min_samples_split": 2,
    },
    {
        "name": "medium",
        "max_depth": 4,
        "min_samples_leaf": 1,
        "min_samples_split": 2,
    },
    {
        "name": "deep",
        "max_depth": 8,
        "min_samples_leaf": 1,
        "min_samples_split": 2,
    },
    {
        "name": "unrestricted",
        "max_depth": None,
        "min_samples_leaf": 1,
        "min_samples_split": 2,
    },
    {
        "name": "regularized_leaf",
        "max_depth": None,
        "min_samples_leaf": 5,
        "min_samples_split": 2,
    },
    {
        "name": "strongly_regularized",
        "max_depth": None,
    },
]

```

```

        "min_samples_leaf": 10,
        "min_samples_split": 10,
    },
]

```

1.3 Running the tests

For each configuration we will train the scikit-learn's built-in regression tree model as well as our implementation, we collect useful statistics and compare them.

```

[59]: results = []

for config in TESTING_CONFIGS:
    print("=" * 80)
    print(f"Dataset: {config['description']}")
    print("=" * 80)

    X, y = generate_regression_data(
        n_samples=config["n_samples"],
        n_features=config["n_features"],
        noise=config["noise"],
    )

    for hp in HYPERPARAM_CONFIGS:
        print("-" * 80)
        print(f"Hyperparams: {hp['name']}")

        model_specs = [
            (
                "our",
                RegressionTree(
                    max_depth=hp["max_depth"],
                    min_samples_leaf=hp["min_samples_leaf"],
                    min_samples_split=hp["min_samples_split"],
                ),
            ),
            (
                "sklearn",
                DecisionTreeRegressor(
                    max_depth=hp["max_depth"],
                    min_samples_leaf=hp["min_samples_leaf"],
                    min_samples_split=hp["min_samples_split"],
                ),
            ),
        ]

        for model_name, model in model_specs:

```

```

metrics = evaluate_model(model, X, y)

print(
    f"{model_name:8s} | "
    f"R2: {metrics['r2_mean']:.3f} ± {metrics['r2_std']:.3f} | "
    f"MSE: {metrics['mse_mean']:.3f} | "
    f"MAE: {metrics['mae_mean']:.3f}"
)

results.append({
    "dataset": config["description"],
    "hyperparams": hp["name"],
    "model": model_name,
    **metrics,
})

```

=====

Dataset: Basic regression data with low noise

=====

Generated regression data with 100 samples and 1 features.

First 5 samples of X:

```

[[ 0.93128012]
 [ 0.08704707]
 [-1.05771093]
 [ 0.31424733]
 [-0.47917424]]

```

First 5 samples of y:

```

[ 38.9917296   3.4964533 -44.05770173  13.09112657 -19.9786311 ]

```

Hyperparams: shallow

```

our      | R2: 0.885 ± 0.029 | MSE: 159.505 | MAE: 10.251
sklearn  | R2: 0.885 ± 0.029 | MSE: 159.505 | MAE: 10.251

```

Hyperparams: medium

```

our      | R2: 0.987 ± 0.006 | MSE: 19.322 | MAE: 3.020
sklearn  | R2: 0.987 ± 0.006 | MSE: 19.322 | MAE: 3.020

```

Hyperparams: deep

```

our      | R2: 0.994 ± 0.008 | MSE: 10.395 | MAE: 1.420
sklearn  | R2: 0.994 ± 0.008 | MSE: 10.395 | MAE: 1.420

```

Hyperparams: unrestricted

```

our      | R2: 0.994 ± 0.008 | MSE: 10.395 | MAE: 1.421
sklearn  | R2: 0.994 ± 0.008 | MSE: 10.395 | MAE: 1.421

```

Hyperparams: regularized_leaf

```

our      | R2: 0.971 ± 0.015 | MSE: 45.195 | MAE: 4.272
sklearn  | R2: 0.971 ± 0.015 | MSE: 45.195 | MAE: 4.272

```

Hyperparams: strongly_regularized

our | R^2 : 0.914 \pm 0.032 | MSE: 124.116 | MAE: 8.168

sklearn | R^2 : 0.914 \pm 0.032 | MSE: 124.116 | MAE: 8.168
=====

Dataset: Regression data with higher noise
=====

Generated regression data with 100 samples and 1 features.

First 5 samples of X:

[[0.93128012]

[0.08704707]

[-1.05771093]

[0.31424733]

[-0.47917424]]

First 5 samples of y:

[50.77992943 -10.06527016 -34.91839191 10.52674299 -17.73837724]

Hyperparams: shallow

our | R^2 : 0.808 \pm 0.073 | MSE: 299.307 | MAE: 13.587

sklearn | R^2 : 0.808 \pm 0.073 | MSE: 299.307 | MAE: 13.587

Hyperparams: medium

our | R^2 : 0.918 \pm 0.028 | MSE: 127.604 | MAE: 9.026

sklearn | R^2 : 0.918 \pm 0.028 | MSE: 127.604 | MAE: 9.026

Hyperparams: deep

our | R^2 : 0.885 \pm 0.037 | MSE: 177.317 | MAE: 10.900

sklearn | R^2 : 0.885 \pm 0.037 | MSE: 177.317 | MAE: 10.900

Hyperparams: unrestricted

our | R^2 : 0.888 \pm 0.035 | MSE: 174.039 | MAE: 10.736

sklearn | R^2 : 0.888 \pm 0.035 | MSE: 174.039 | MAE: 10.736

Hyperparams: regularized_leaf

our | R^2 : 0.909 \pm 0.027 | MSE: 148.618 | MAE: 9.392

sklearn | R^2 : 0.909 \pm 0.027 | MSE: 148.618 | MAE: 9.392

Hyperparams: strongly_regularized

our | R^2 : 0.855 \pm 0.047 | MSE: 236.005 | MAE: 11.690

sklearn | R^2 : 0.855 \pm 0.047 | MSE: 236.005 | MAE: 11.690
=====

Dataset: Regression data with multiple features
=====

Generated regression data with 200 samples and 5 features.

First 5 samples of X:

[[-0.3853136 0.1990597 -0.60021688 0.46210347 0.06980208]

[0.13074058 1.6324113 -1.43014138 -1.24778318 -0.44004449]

[-0.77300978 0.22409248 0.0125924 -0.40122047 0.0976761]

```

[-0.57677133 -0.05023811 -0.23894805  0.27045683 -0.90756366]
[-0.57581824  0.6141667  0.75750771 -0.2209696 -0.53050115]]
First 5 samples of y:
[ -23.06825766 -124.34606251    6.99640504 -71.77655611   17.06336023]

```

```

-----
Hyperparams: shallow
our      | R2: 0.623 ± 0.079 | MSE: 3325.007 | MAE: 44.673
sklearn  | R2: 0.623 ± 0.079 | MSE: 3325.007 | MAE: 44.673

```

```

-----
Hyperparams: medium
our      | R2: 0.766 ± 0.068 | MSE: 2056.622 | MAE: 36.590
sklearn  | R2: 0.759 ± 0.065 | MSE: 2118.765 | MAE: 36.980

```

```

-----
Hyperparams: deep
our      | R2: 0.836 ± 0.072 | MSE: 1437.223 | MAE: 29.751
sklearn  | R2: 0.852 ± 0.067 | MSE: 1315.800 | MAE: 27.940

```

```

-----
Hyperparams: unrestricted
our      | R2: 0.836 ± 0.066 | MSE: 1435.343 | MAE: 29.745
sklearn  | R2: 0.839 ± 0.078 | MSE: 1414.952 | MAE: 28.763

```

```

-----
Hyperparams: regularized_leaf
our      | R2: 0.811 ± 0.043 | MSE: 1709.692 | MAE: 31.868
sklearn  | R2: 0.811 ± 0.043 | MSE: 1709.692 | MAE: 31.868

```

```

-----
Hyperparams: strongly_regularized
our      | R2: 0.783 ± 0.014 | MSE: 1974.617 | MAE: 34.900
sklearn  | R2: 0.783 ± 0.014 | MSE: 1974.617 | MAE: 34.900
=====

```

Dataset: Larger dataset with moderate noise

Generated regression data with 5000 samples and 3 features.

```

First 5 samples of X:
[[ 0.67796997 -1.28472777 -0.33102433]
 [ 1.03138053  0.3881858  -0.97027133]
 [-1.21689671  1.36337651 -0.60515624]
 [-0.54429615 -0.50442268 -1.5198928 ]
 [ 0.2074888  0.44567791  0.42350787]]
First 5 samples of y:
[ -3.90976845 -46.39345674 -98.15873956 -157.1184525   47.56364341]

```

```

-----
Hyperparams: shallow
our      | R2: 0.728 ± 0.016 | MSE: 2520.708 | MAE: 39.683
sklearn  | R2: 0.728 ± 0.016 | MSE: 2520.708 | MAE: 39.683

```

```

-----
Hyperparams: medium
our      | R2: 0.904 ± 0.007 | MSE: 890.915 | MAE: 23.495
sklearn  | R2: 0.904 ± 0.007 | MSE: 890.915 | MAE: 23.495

```

```
-----
Hyperparams: deep
our      | R2: 0.989 ± 0.001 | MSE: 101.228 | MAE: 7.401
sklearn  | R2: 0.989 ± 0.001 | MSE: 100.775 | MAE: 7.390
-----
```

```
Hyperparams: unrestricted
our      | R2: 0.994 ± 0.001 | MSE: 55.475 | MAE: 4.884
sklearn  | R2: 0.994 ± 0.000 | MSE: 52.423 | MAE: 4.828
-----
```

```
Hyperparams: regularized_leaf
our      | R2: 0.992 ± 0.001 | MSE: 70.037 | MAE: 5.411
sklearn  | R2: 0.992 ± 0.001 | MSE: 70.039 | MAE: 5.411
-----
```

```
Hyperparams: strongly_regularized
our      | R2: 0.989 ± 0.002 | MSE: 99.862 | MAE: 6.621
sklearn  | R2: 0.989 ± 0.002 | MSE: 99.862 | MAE: 6.621
=====
```

Dataset: Large dataset with low noise

=====

Generated regression data with 10000 samples and 2 features.

First 5 samples of X:

```
[[ -0.5691482   1.59040357]
 [  0.73487779   0.49097495]
 [  0.20069869   1.10623156]
 [ -1.07774393  -0.54427443]
 [  0.10507597  -0.88502863]]
```

First 5 samples of y:

```
[ -47.49496686   75.02051971   25.36349671 -108.87452141    5.49344142]
```

```
Hyperparams: shallow
our      | R2: 0.877 ± 0.005 | MSE: 1180.012 | MAE: 27.415
sklearn  | R2: 0.877 ± 0.005 | MSE: 1180.012 | MAE: 27.415
-----
```

```
Hyperparams: medium
our      | R2: 0.987 ± 0.001 | MSE: 128.172 | MAE: 8.694
sklearn  | R2: 0.987 ± 0.001 | MSE: 128.172 | MAE: 8.694
-----
```

```
Hyperparams: deep
our      | R2: 0.999 ± 0.000 | MSE: 10.967 | MAE: 2.547
sklearn  | R2: 0.999 ± 0.000 | MSE: 11.017 | MAE: 2.547
-----
```

```
Hyperparams: unrestricted
our      | R2: 1.000 ± 0.000 | MSE: 2.996 | MAE: 1.118
sklearn  | R2: 1.000 ± 0.000 | MSE: 3.065 | MAE: 1.123
-----
```

```
Hyperparams: regularized_leaf
our      | R2: 0.999 ± 0.000 | MSE: 5.449 | MAE: 1.394
sklearn  | R2: 0.999 ± 0.000 | MSE: 5.442 | MAE: 1.393
```

Hyperparams: strongly_regularized

our | R^2 : 0.999 \pm 0.000 | MSE: 10.863 | MAE: 1.810

sklearn | R^2 : 0.999 \pm 0.000 | MSE: 10.863 | MAE: 1.810
=====

Dataset: Large dataset with high noise
=====

Generated regression data with 10000 samples and 2 features.

First 5 samples of X:

[[-0.5691482 1.59040357]

[0.73487779 0.49097495]

[0.20069869 1.10623156]

[-1.07774393 -0.54427443]

[0.10507597 -0.88502863]]

First 5 samples of y:

[-56.9869474 68.99765941 2.79332014 -93.43973055 7.91252206]

Hyperparams: shallow

our | R^2 : 0.841 \pm 0.004 | MSE: 1592.545 | MAE: 31.659

sklearn | R^2 : 0.841 \pm 0.004 | MSE: 1592.545 | MAE: 31.659

Hyperparams: medium

our | R^2 : 0.947 \pm 0.002 | MSE: 529.733 | MAE: 18.329

sklearn | R^2 : 0.947 \pm 0.002 | MSE: 529.733 | MAE: 18.329

Hyperparams: deep

our | R^2 : 0.955 \pm 0.001 | MSE: 446.888 | MAE: 16.879

sklearn | R^2 : 0.955 \pm 0.001 | MSE: 446.658 | MAE: 16.877

Hyperparams: unrestricted

our | R^2 : 0.920 \pm 0.001 | MSE: 801.002 | MAE: 22.730

sklearn | R^2 : 0.921 \pm 0.002 | MSE: 795.899 | MAE: 22.674

Hyperparams: regularized_leaf

our | R^2 : 0.945 \pm 0.001 | MSE: 552.718 | MAE: 18.864

sklearn | R^2 : 0.945 \pm 0.001 | MSE: 552.655 | MAE: 18.863

Hyperparams: strongly_regularized

our | R^2 : 0.951 \pm 0.001 | MSE: 491.906 | MAE: 17.711

sklearn | R^2 : 0.951 \pm 0.001 | MSE: 491.906 | MAE: 17.711
=====

Dataset: Moderate dataset with many features
=====

Generated regression data with 1000 samples and 50 features.

First 5 samples of X:

[[0.22210537 0.62386678 -0.16605612 0.63463021 0.75376893 1.65859202

-0.70189811 0.18361217 -2.08195404 0.60286295 0.14837528 -1.13521912

0.9986041 -0.49461035 -0.51024062 0.93668094 -0.399035 -0.38356556

```

1.1898542 -0.5212666 -1.16138215 1.72223231 0.36539182 1.33601969
0.78801999 -0.83774995 0.56658811 0.92829461 -0.38674766 0.6208172
0.19797213 1.33836855 0.02365751 -2.52871576 1.75988802 -1.207332
-0.82470391 -0.68112696 0.6151632 0.35782542 -0.7864838 0.10673978
0.69987124 -0.03436237 -0.11283199 -0.36389558 0.10516231 -0.53678647
-0.0572436 -0.30034055]
[ 0.83057101 -1.71669373 -0.15501343 1.11435578 -0.82530382 0.60915961
-0.06801749 -0.71940537 -0.05022968 -1.14378475 0.78259513 -0.10643051
1.68403763 -0.58623557 -0.40156696 -0.59052457 0.96258821 0.79276514
-0.16365711 0.99107132 -1.08776862 -0.43419762 -1.11047912 -1.56341347
-0.20934306 0.19477547 1.77209275 -0.18015897 1.54042895 1.66236667
0.17283733 0.34237509 1.7280638 -0.85290707 -1.17369964 0.55164771
-1.65811325 0.11898642 0.20540077 -0.43031452 2.22904372 -1.44808321
-1.14136697 -0.10875609 -0.51455358 -0.14546159 0.740684 0.85154814
-0.72018801 -0.39993873]
[-0.96335321 0.5437933 -1.11313988 1.21235947 0.55206656 -0.91026463
-0.85609014 0.53390779 0.38681169 0.25747809 0.40567447 -1.38173128
-1.34554208 2.22468763 0.15646961 -1.15238917 -0.11994209 -0.0888014
-1.33626712 -2.46829003 0.73874622 -0.74254137 0.21518837 2.30103361
0.6396049 -1.00437615 0.009012 1.53905609 0.68941732 -0.81846411
-0.33594849 1.13719016 -0.84854273 -0.92413914 -0.1971291 -1.59092134
-0.20260583 1.09189461 -1.43450227 -1.83909427 0.63691302 1.81273542
-0.4661267 0.98463866 -0.82158042 -0.80044475 -1.55415405 -0.79595511
0.85645506 -0.13711049]
[ 2.1498434 -0.10467932 -0.43722006 -0.33944537 1.17827468 0.60539814
0.62604166 0.02681872 -0.59611361 -0.325793 -0.26361207 1.89754169
-0.40314991 -0.14655815 1.29817434 0.3420916 -0.23200561 1.83926104
-1.72979741 2.16414981 1.01110931 -0.68284508 0.84036538 1.90220597
2.31782164 0.93558607 -0.64628821 -0.42569971 -0.40269403 -1.27318569
1.76253455 1.41378331 -0.04727162 1.15564512 -1.09488559 0.69360613
-0.94454315 0.02231656 0.04946641 -2.29909535 -0.15431771 0.0542324
0.31923321 0.39123279 0.5049892 -0.00548364 -0.58436466 0.76151403
-0.75676622 -1.26037829]
[-0.5372907 0.06883836 -0.49145363 -2.14575896 -0.98773148 -0.43352192
0.31588178 0.46436161 -0.79141506 -0.20844714 -0.74615901 -0.11587905
0.32387858 0.28785406 -0.58537497 1.2066765 -0.70211482 0.95639433
-0.69194153 -0.20713186 0.54609752 0.32999556 -0.36007058 0.95665943
0.22027397 1.11085552 0.40401589 0.95472592 -0.27240938 -1.51210178
0.50589761 -0.90487118 -0.41419898 -0.42887986 -2.38069159 -1.0825947
-0.5624822 0.92588024 -0.8635109 0.42003858 -1.82769117 0.95275321
-1.41406661 -0.36089524 1.41201599 1.56501888 1.34935979 -0.75915137
-0.91560975 0.18396914]]

```

First 5 samples of y:

```
[137.83775667 -3.71055967 145.69973099 -9.68172968 -83.67934132]
```

Hyperparams: shallow

our | R^2 : 0.380 \pm 0.039 | MSE: 17106.506 | MAE: 104.267

sklearn | R^2 : 0.380 \pm 0.039 | MSE: 17106.506 | MAE: 104.267

```

-----
Hyperparams: medium
our      | R²: 0.670 ± 0.027 | MSE: 9083.463 | MAE: 76.403
sklearn  | R²: 0.668 ± 0.028 | MSE: 9145.507 | MAE: 76.560
-----

Hyperparams: deep
our      | R²: 0.727 ± 0.026 | MSE: 7523.640 | MAE: 67.167
sklearn  | R²: 0.728 ± 0.029 | MSE: 7475.288 | MAE: 66.524
-----

Hyperparams: unrestricted
our      | R²: 0.705 ± 0.032 | MSE: 8120.275 | MAE: 69.466
sklearn  | R²: 0.685 ± 0.041 | MSE: 8631.939 | MAE: 71.072
-----

Hyperparams: regularized_leaf
our      | R²: 0.756 ± 0.026 | MSE: 6707.463 | MAE: 64.719
sklearn  | R²: 0.757 ± 0.028 | MSE: 6674.830 | MAE: 64.225
-----

Hyperparams: strongly_regularized
our      | R²: 0.758 ± 0.028 | MSE: 6653.061 | MAE: 64.376
sklearn  | R²: 0.758 ± 0.028 | MSE: 6653.061 | MAE: 64.376

```

1.4 Results

```

[64]: df = pd.DataFrame(results)
      paired = (
          df.pivot_table(
              index=["dataset", "hyperparams"],
              columns="model",
              values=["r2_mean", "mse_mean", "mae_mean"]
          )
      )

      paired.columns = ["_".join(col) for col in paired.columns]
      paired = paired.reset_index()

      paired["r2_diff"] = paired["r2_mean_our"] - paired["r2_mean_sklearn"]
      paired["mse_diff"] = paired["mse_mean_our"] - paired["mse_mean_sklearn"]
      paired["mae_diff"] = paired["mae_mean_our"] - paired["mae_mean_sklearn"]

      paired.head()

```

```

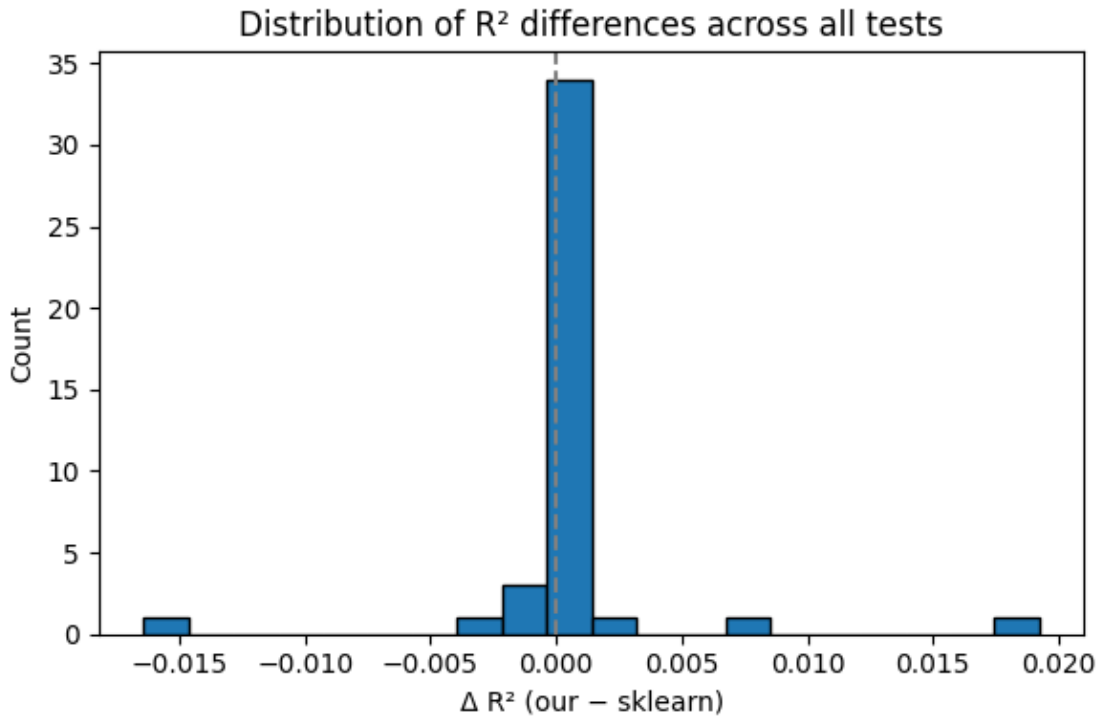
[64]:
      dataset      hyperparams  mae_mean_our \
0  Basic regression data with low noise      deep      1.420082
1  Basic regression data with low noise      medium      3.019620
2  Basic regression data with low noise  regularized_leaf      4.271927
3  Basic regression data with low noise      shallow     10.251087
4  Basic regression data with low noise  strongly_regularized      8.168215

```

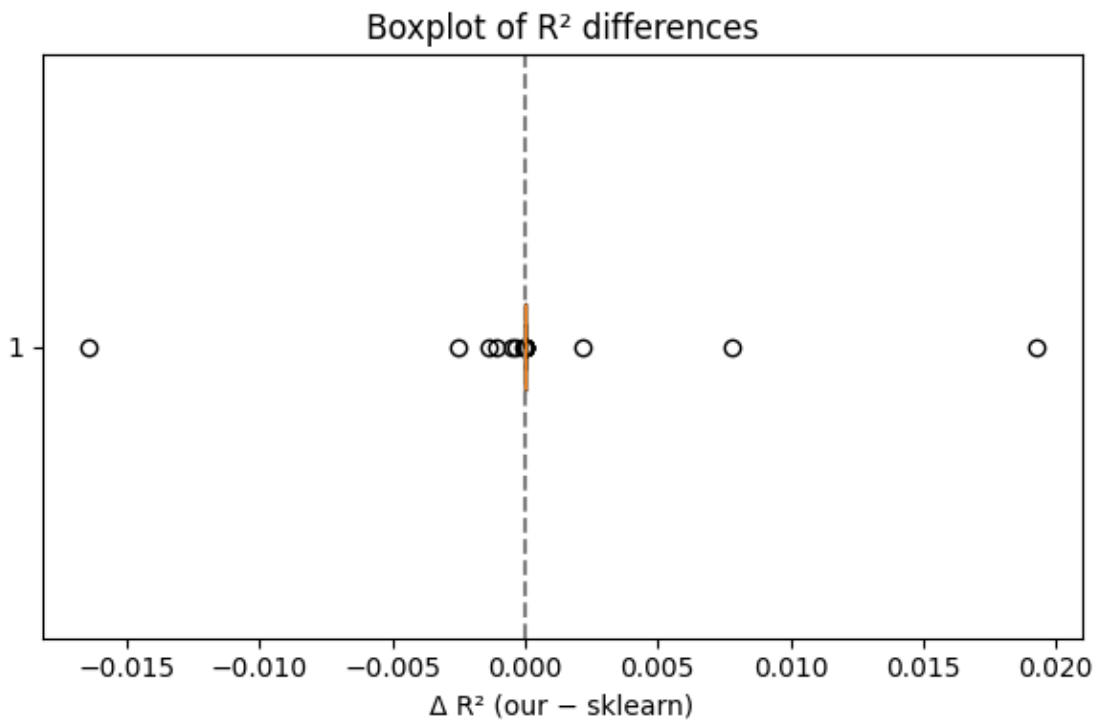
	mae_mean_sklearn	mse_mean_our	mse_mean_sklearn	r2_mean_our \
0	1.420082	10.395394	10.395394	0.993668
1	3.019620	19.322168	19.322168	0.986635
2	4.271927	45.194549	45.194549	0.971219
3	10.251087	159.505327	159.505327	0.884677
4	8.168215	124.116470	124.116470	0.913597

	r2_mean_sklearn	r2_diff	mse_diff	mae_diff
0	0.993668	0.0	0.000000e+00	0.000000e+00
1	0.986635	0.0	0.000000e+00	0.000000e+00
2	0.971219	0.0	2.131628e-14	1.776357e-15
3	0.884677	0.0	-5.684342e-14	1.776357e-15
4	0.913597	0.0	0.000000e+00	0.000000e+00

```
[66]: plt.figure(figsize=(6, 4))
plt.hist(paired["r2_diff"], bins=20, edgecolor="black")
plt.axvline(0, linestyle="--", color="gray")
plt.xlabel(" $\Delta R^2$  (our - sklearn)")
plt.ylabel("Count")
plt.title("Distribution of  $R^2$  differences across all tests")
plt.tight_layout()
plt.show()
```



```
[67]: plt.figure(figsize=(6, 4))
plt.boxplot(
    paired["r2_diff"],
    vert=False,
    showfliers=True
)
plt.axvline(0, linestyle="--", color="gray")
plt.xlabel("Δ R2 (our - sklearn)")
plt.title("Boxplot of R2 differences")
plt.tight_layout()
plt.show()
```



```
[69]: by_hp = (
    paired.groupby("hyperparams")["r2_diff"]
    .agg(["mean", "std"])
    .reset_index()
)

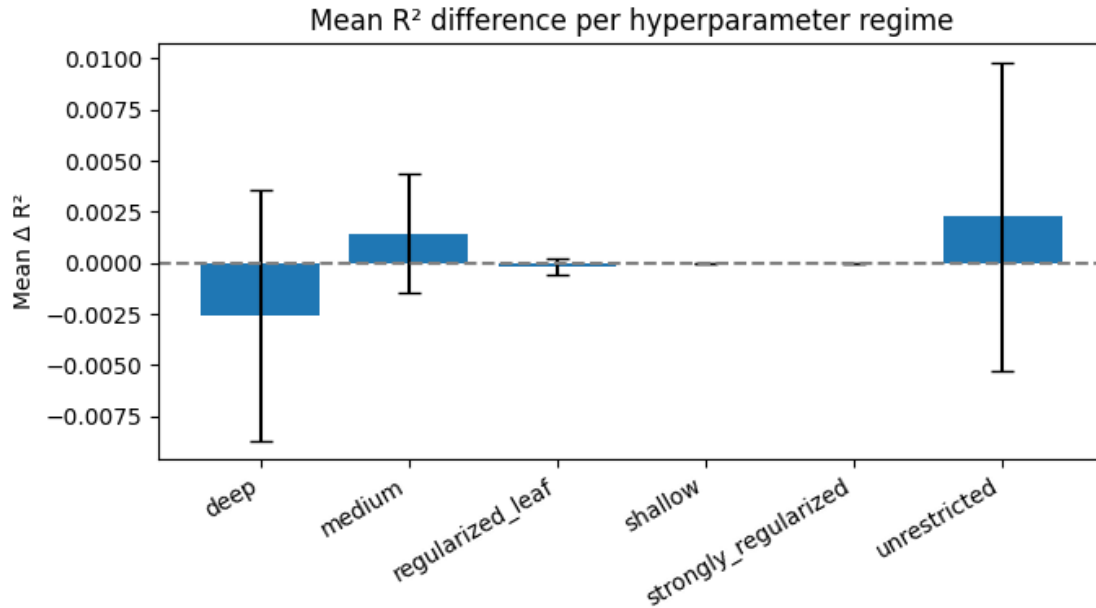
plt.figure(figsize=(7, 4))
plt.bar(
    by_hp["hyperparams"],
    by_hp["mean"],
    yerr=by_hp["std"],

```

```

    capsizes=5
)
plt.axhline(0, linestyle="--", color="gray")
plt.ylabel("Mean  $\Delta R^2$ ")
plt.title("Mean  $R^2$  difference per hyperparameter regime")
plt.xticks(rotation=30, ha="right")
plt.tight_layout()
plt.show()

```

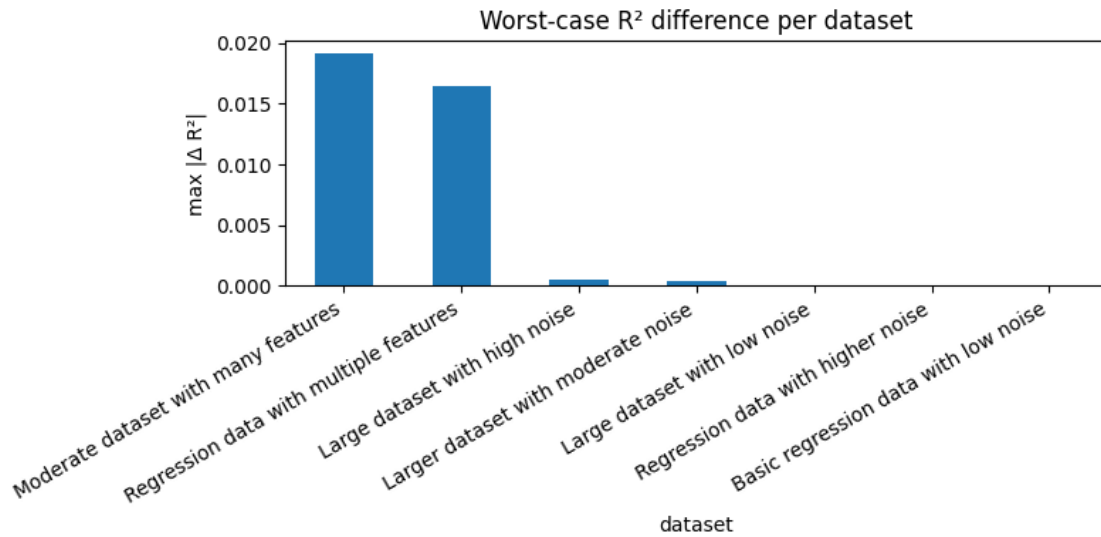


```

[70]: worst_by_dataset = (
    paired.groupby("dataset")["r2_diff"]
    .apply(lambda x: x.abs().max())
    .sort_values(ascending=False)
)

plt.figure(figsize=(8, 4))
worst_by_dataset.plot(kind="bar")
plt.ylabel("max  $|\Delta R^2|$ ")
plt.title("Worst-case  $R^2$  difference per dataset")
plt.xticks(rotation=30, ha="right")
plt.tight_layout()
plt.show()

```



```
[71]: fig, axes = plt.subplots(1, 2, figsize=(10, 4))

axes[0].hist(paired["mse_diff"], bins=20, edgecolor="black")
axes[0].axvline(0, linestyle="--", color="gray")
axes[0].set_title("Δ MSE distribution")

axes[1].hist(paired["mae_diff"], bins=20, edgecolor="black")
axes[1].axvline(0, linestyle="--", color="gray")
axes[1].set_title("Δ MAE distribution")

plt.tight_layout()
plt.show()
```

