> **Note**
>
> Click here to download the full example code

Train Test Performance

This notebooks provides an overview for using and understanding train test performance check.

Structure:

What is the purpose of the check?

This check helps you compare your model's performance between the train and test datasets based on multiple scorers.

The default scorers that are used are F1, Percision, and Recall for Classification and Negative Root Mean Square Error, Negative Mean Absolute Error, and R2 for Regression. RMSE and MAE Scorers are negative because we subscribe to the sklearn convention of defining scoring functions. See scorers documentation

Generate data & model

```
from deepchecks.tabular.datasets.classification.iris import load_data, load_fitted_model

train_dataset, test_dataset = load_data()
model = load_fitted_model()
```

Run the check

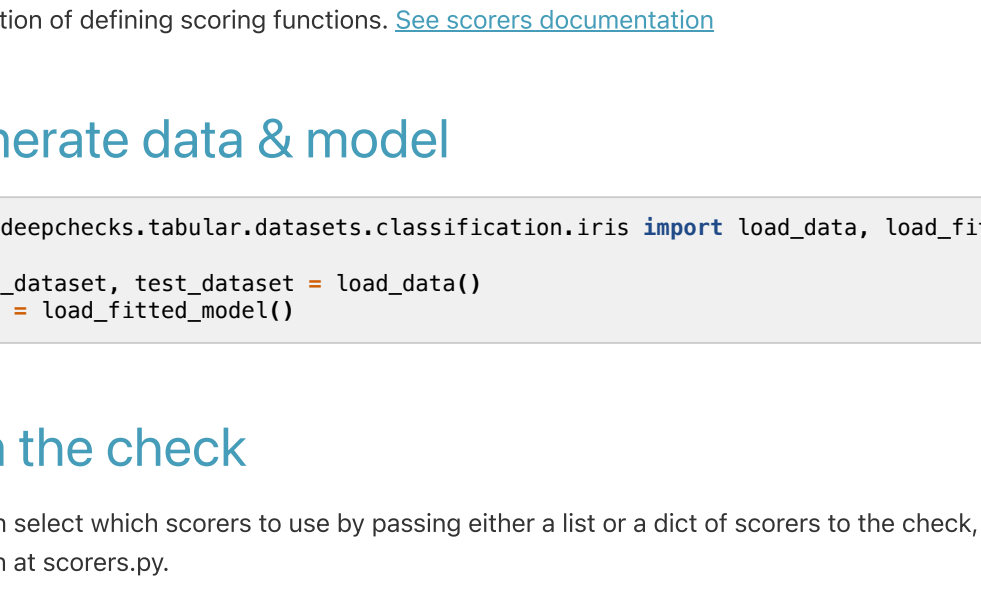
You can select which scorers to use by passing either a list or a dict of scorers to the check, the full list of possible scorers can be seen at scorers.py.

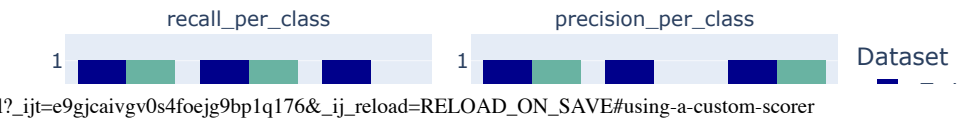
```
from deepchecks.tabular.checks import TrainTestPerformance

check = TrainTestPerformance(scorers=['recall_per_class', 'precision_per_class', 'f1_macro', 'f1_micro'])
result = check.run(train_dataset, test_dataset, model)
result.show()
```

Performance Report

Summarize given model performance on the train and test datasets based on selected scorers.

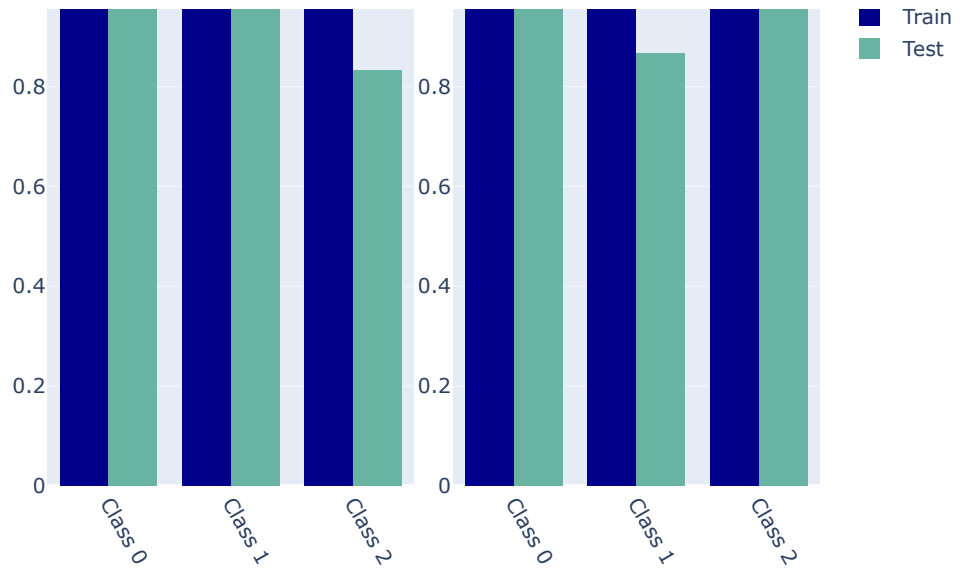
Additional Outputs





Search the docs ...

Train Test Performance



Define a condition

We can define on our check a condition that will validate that our model doesn't degrade on new data.

Let's add a condition to the check and see what happens when it fails:

```
check.add_condition_train_test_relative_degradation_less_than(0.15)
result = check.run(train_dataset, test_dataset, model)
result.show(show_additional_outputs=False)
```

Performance Report

Summarize given model performance on the train and test datasets based on selected scorers.

Conditions Summary

Status	Condition	More Info
×	Train-Test scores relative degradation is less than 0.15	1 scores failed. Found max degradation of 16.67% for metric recall_per_class and class 2.0.

Performance Report

Summarize given model performance on the train and test datasets based on selected scorers.

Conditions Summary

Status	Condition	More Info
×	Train-Test scores relative degradation is less than 0.15	1 scores failed. Found max degradation of 16.67% for metric recall_per_class and class 2.0.

We detected that for class "2" the Recall score result is degraded by more than 15%

Using a custom scorer

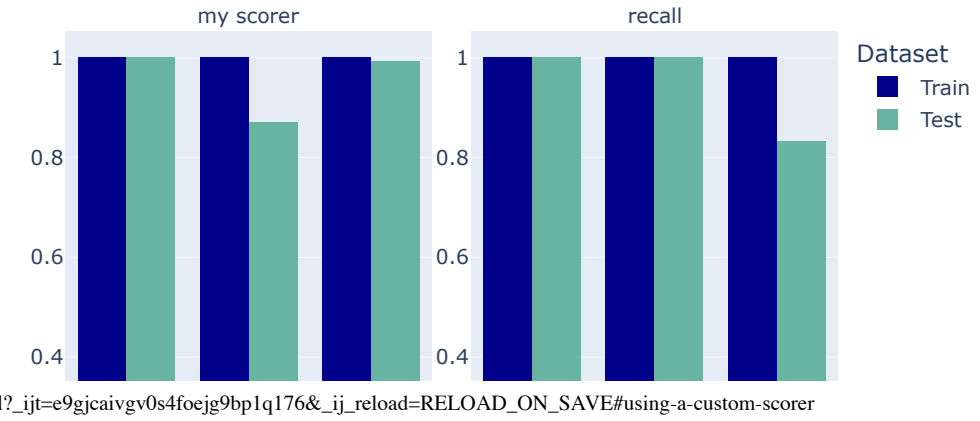
In addition to the built-in scorers, we can define our own scorer based on sklearn api and run it using the check alongside other scorers:

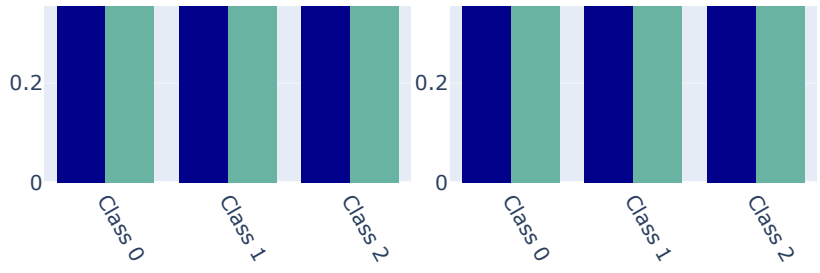
```
from sklearn.metrics import fbeta_score, make_scorer

fbeta_scorer = make_scorer(fbeta_score, labels=[0, 1, 2], average=None, beta=0.2)

check = TrainTestPerformance(scorers={'my scorer': fbeta_scorer, 'recall': 'recall_per_class'})
check.run(train_dataset, test_dataset, model)
```

Performance Report

Summarize given model performance on the train and test datasets based on selected scorers.

Additional Outputs





Total running time of the script: (0 minutes 5.489 seconds)

Download Python source code: plot_test.py

Download Jupyter notebook: plot_test.ipynb

Gallery generated by Sphinx-Gallery

Previous
Plots

Next
Example RST Notebook