# Orange Conformal

September 13, 2021

## 1 Conformal Prediction in Orange

Tomaž Hočevar, tomaz.hocevar@fri.uni-lj.si
Blaž Zupan, blaz.zupan@fri.uni-lj.si

## 2 Orange3-Conformal package

- add-on for the Orange3 data mining software
- provides an extensive toolset for basic conformal prediction
- provided library can be used in Python Scripts (no GUI yet)

Useful links:

- Python package (https://pypi.org/project/Orange3-Conformal/)
- documentation (https://orange3-conformal.readthedocs.io/)
- development (https://github.com/biolab/orange3-conformal)

## 3 Related packages

- nonconformist (Python)
- conformal (R)
- PyCP (Python, unavailable)

## 4 Design

Components of a conformal predictor:

- conformal prediction method (transductive, inductive, cross conformal predictors)
- nonconfromity measure
  - inverse probability, probability margin, SVM distance, KNN fraction, …
  - absolute error, absolute error normalized, error model, average error KNN, …
- classification/regression model

Properties:

- modular and extensible
- wide range of nonconformity measures for experimentation
- evaluation scores and procedures to estimate performance

# 5 Examples

## 5.1 Iris data set

[1]:
```
import Orange
import orangecontrib.conformal as cp

iris = Orange.data.Table('iris')
iris.domain
```

[1]:
```
[sepal length, sepal width, petal length, petal width | iris]
```

[2]:
```
learn, test_instance = iris[:-1], iris[-1]
train, calibrate = learn[0::2], learn[1::2]
test_instance
```

[2]:
```
[5.9, 3.0, 5.1, 1.8 | Iris-virginica]
```

## 5.2 Simple conformal classifier

[3]:
```
learner = Orange.classification.LogisticRegressionLearner()
measure = cp.nonconformity.InverseProbability(learner)
predictor = cp.classification.InductiveClassifier(measure, train, calibrate)

for eps in [0.1, 0.05, 0.01]:
    print(eps, predictor(test_instance, eps))
```

```
0.1 []
0.05 ['Iris-virginica']
0.01 ['Iris-setosa', 'Iris-versicolor', 'Iris-virginica']
```

## 5.3 Evaluation

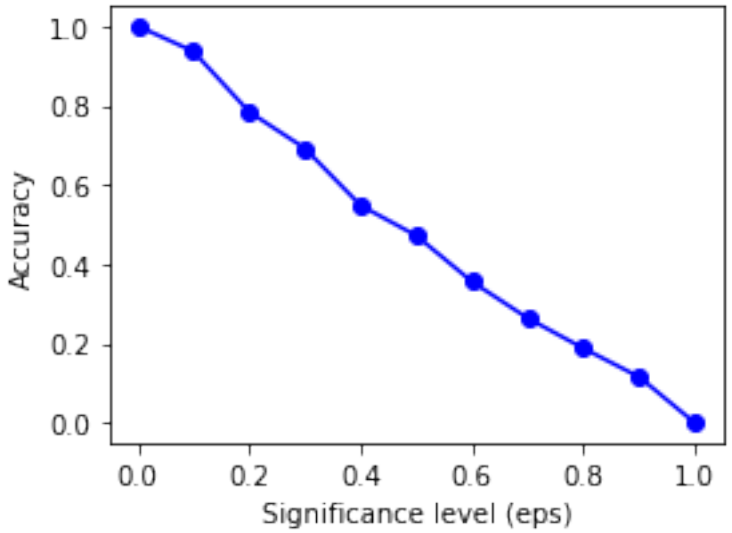
[4]:
```
import numpy as np
np.random.seed(1)
learn, test = next(cp.evaluation.RandomSampler(iris, 1, 1))
train, calibrate = next(cp.evaluation.RandomSampler(learn, 1, 1))
results = cp.evaluation.run_train_test(predictor, 0.1, train, test, calibrate)

print("accuracy: ", results.accuracy())
print("singleton: ", results.singleton_criterion())
print("empty: ", results.empty_criterion())
print("multiple: ", results.multiple_criterion())
```

```
accuracy:  0.8933333333333333
singleton:  0.9066666666666666
empty:  0.09333333333333334
multiple:  0.0
```

### 5.4 Validation

[41]:
```
import matplotlib.pyplot as plt
plt.figure(figsize = (4, 3))
plt.xlabel('Significance level (eps)')
plt.ylabel('Accuracy')
x = np.linspace(0.0, 1.0, 11)
y = [round(results.accuracy(eps = e), 3) for e in x]
plt.plot(x, y, linestyle = '-', marker = 'o', color = 'b')
plt.tight_layout()
```




## 6 Simple conformal regressor

[6]:
```
housing = Orange.data.Table('housing')
housing.domain
```

[6]:
```
[CRIM, ZN, INDUS, CHAS, NOX, RM, AGE, DIS, RAD, TAX, PTRATIO, B, LSTAT | MEDV]
```

[8]:
```
train, test_instance = housing[:-1], housing[-1]
test_instance
```

[8]:
```
[0.04741, 0.0, 11.93, 0, 0.5730, … | 11.9]
```

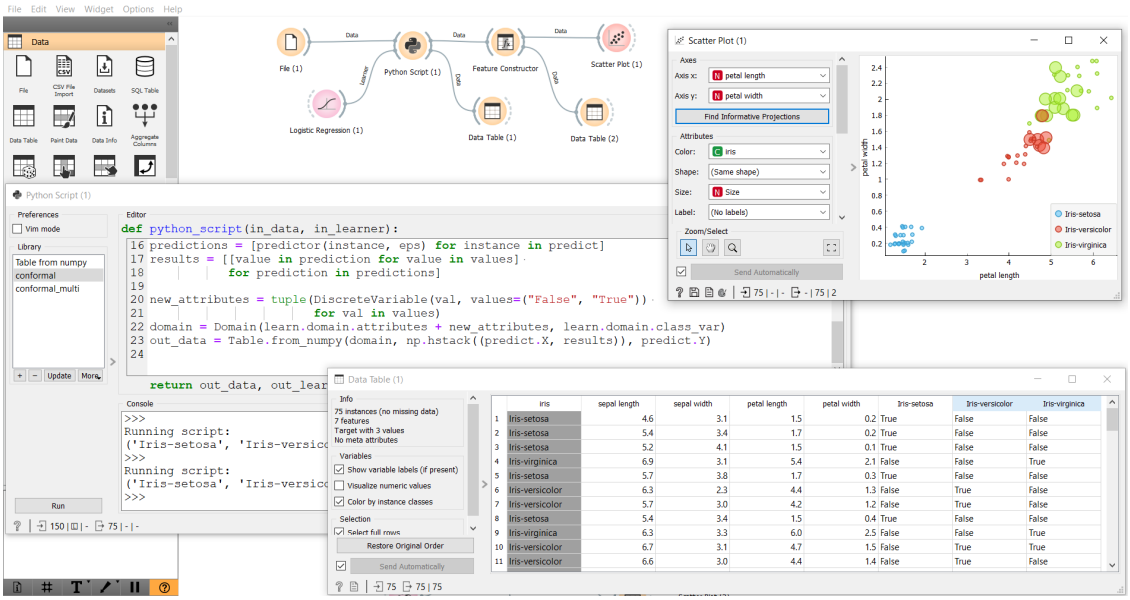
[46]:
```
learner = Orange.regression.LinearRegressionLearner()
measure = cp.nonconformity.AbsError(learner)
predictor = cp.regression.CrossRegressor(measure, 5, train)

for eps in [0.1, 0.05, 0.01]:
```

```
    print(predictor(test_instance, eps))
```

```
(15.726917024823898, 29.398863595167185)
(12.644058384356232, 32.48172223563485)
(4.800611093725575, 40.32516952626551)
```

# 7 Visual conformal prediction with Orange




# 8 Future Work

- standalone widgets for conformal prediction
- custom visualizations for set and range predictions
- missing methods: Venn predictors, predictive distributions, …
- training material and practical showcases
- collaboration with conformal prediction experts