


M5 / M5' model trees in python with `m5py`

compliant with scikit-learn

PyConDE PyData Berlin – April 12th, 2022

Sylvain Marié - Senior Group Expert
Schneider Electric Artificial Intelligence Hub

Introduction

Offices, schools, hotels…
Energy efficiency tracking
Microgrid campus

Power distribution devices
Plug’n play network protocols
Service-oriented components

Critical processes
Condition monitoring
Predictive maintenance

Industry, Manufacturing
Advanced process control
Energy efficient processes

Water distribution networks
Demand forecasting

We are hiring
data scientists
& data engineers !

Life Is On Schneider Electric

M5 / M5' ? Once upon a time…

v1 (2012-2016, RIP)

Version control	
Analytics language	
Machine Learning	
Web service wrapper	
Model storage	
Cloud host	 (VMs = IaaS)

Our first analytics-as-a-service cloud platform !

M5P model was here ☺

M5 / M5' ? Once upon a time…

v1 (2012-2016, RIP)

Version control	
Analytics language	
Machine Learning	
Web service wrapper	
Model storage	
Cloud host	 (VMs = IaaS)

Our first analytics-as-a-service cloud platform !

M5P model was here ☺

…but not there ☹

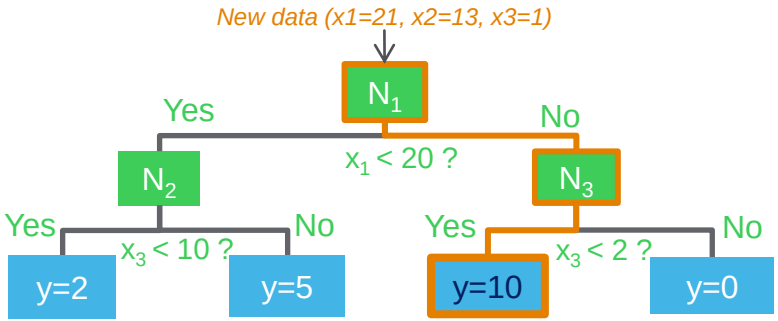
v2 (2016-2020)

M5

Quinlan J. R. (1992). Learning with continuous classes. Proceedings of the Australian Joint Conference on Artificial Intelligence. 343--348. World Scientific, Singapore.

Principles (1):

- Grow a regression tree model (as in CART) by splitting the dataset ("divide-and-conquer"), using `impurity = std(y)`. Stop the splitting process when impurity or nb samples is too small.
- Start by placing a constant predictor at each leave, as in CART.

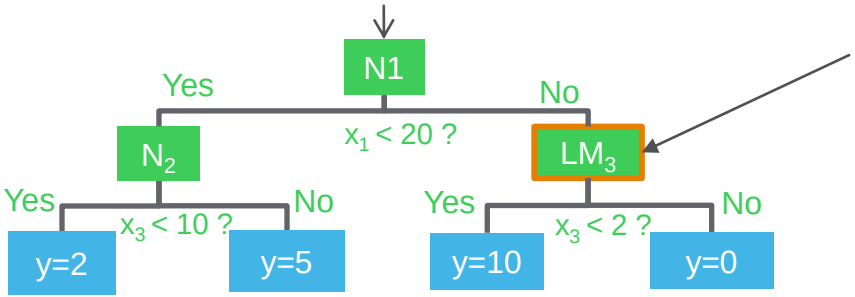


M5

Quinlan J. R. (1992). Learning with continuous classes. Proceedings of the Australian Joint Conference on Artificial Intelligence. 343--348. World Scientific, Singapore.

Principles (2):

- For all nodes in the tree, train a linear model on the samples reaching that node.
- Simplify it by greedily removing parameters until the performance-vs-complexity tradeoff (an "adjusted" MAE) stops decreasing.
- Prune from bottom-up: everytime a linear model at a node is better than the combination of its two children, cut the subtree



$$LM: y = a_1 * x_1 + b \ ?$$

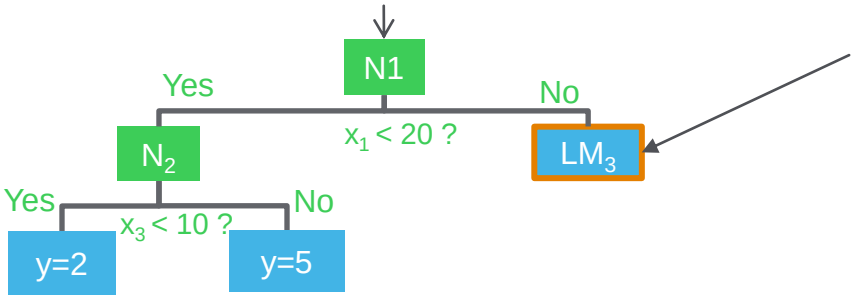
$$err_{LM} = MAE_{LM} * \frac{n + v}{n - v}$$

M5

Quinlan J. R. (1992). Learning with continuous classes. Proceedings of the Australian Joint Conference on Artificial Intelligence. 343--348. World Scientific, Singapore.

Principles (2):

- For all nodes in the tree, train a linear model on the samples reaching that node.
- Simplify it by greedily removing parameters until the performance-vs-complexity tradeoff (an "adjusted" MAE) stops decreasing.
- Prune from bottom-up: everytime a linear model at a node is better than the combination of its two children, cut the subtree



$$LM: y = a_1 * x_1 + b \ ?$$

$$err_{LM} = MAE_{LM} * \frac{n + v}{n - v}$$

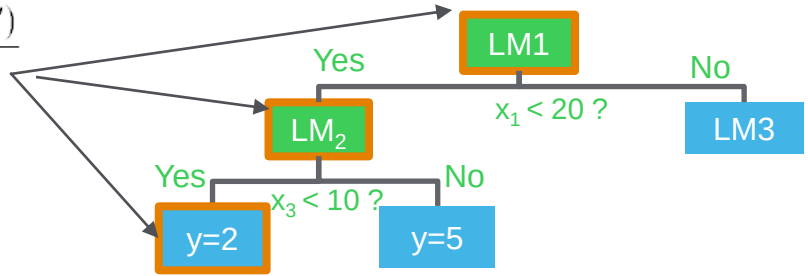
M5

Quinlan J. R. (1992). Learning with continuous classes. Proceedings of the Australian Joint Conference on Artificial Intelligence. 343--348. World Scientific, Singapore.

Principles (3):

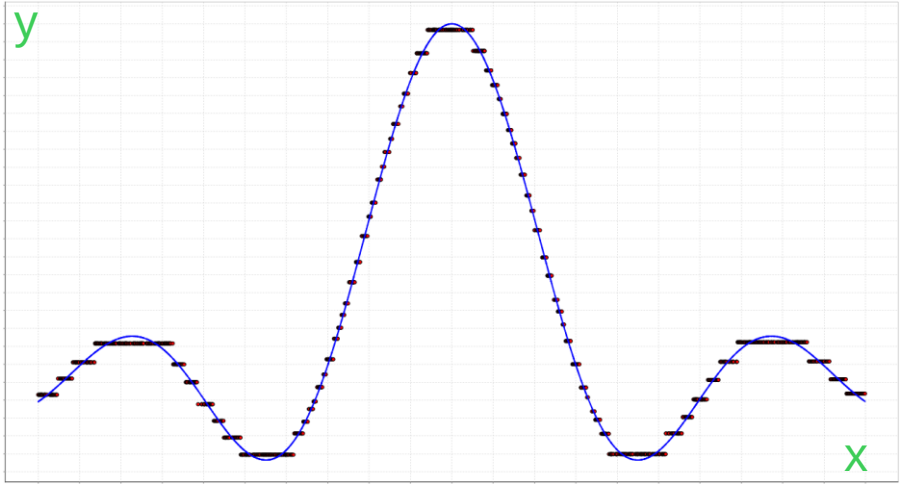
- Smooth recursively when predicting, in order for nodes with few samples to be corrected by their parent(s)

$$PV(S) = \frac{n_i \times PV(S_i) + k \times M(S)}{n_i + k}$$



M5 – why ?

Piecewise linear: model complex nonlinear relationships while preserving interpretability



Constant leaves

Linear model leaves

M5’ (prime)

Wang, Y and Witten, I. H. (1997). Induction of model trees for predicting continuous classes. Proceedings of the poster papers of the European Conference on Machine Learning. University of Economics, Faculty of Informatics and Statistics, Prague.

- Brings clarification + a reference implementation to the initial idea
- 2 improvements: handling categorical attributes (enumerations) and missing values
- Results show how the two improvements behave on datasets with many categorical/missing
- An additional improvement seems to be provided (“pruning factor”) but without explanation in the paper, apart from “we introduce a modification which allows the tree size to be reduced dramatically”. From the code implemented in Weka, it could be this new PF (default=2) :

$$err_{LM} = MAE_{LM} * \frac{n + PF * v}{n - v}$$

m5py

Timeline

- First implementation in 2016 (internal project)
- PR opened at scikit-learn in 2019 (scikit-learn#13732)
  - Stuck since, mostly because of overall complexity and documentation quality.
  - Still, several community members explicitly stated their interest.
  - It seems to me that this is interesting for educational purposes too ?

Today's proposal

- I put something here https://smarie.github.io/python-m5p/
- And made a first release (0.3.0) ☺

Implementing `m5py` - caveats

M5/M5' use non-generalized tree concepts (e.g. « std » instead of « impurity ») – 'translation' is needed

Pre-computing (weka's « installing ») the smoothed models for speed at prediction time requires to reformulate the smoothing formula using linear model coefficients.

Some tree growing requirements are not available in sklearn's `BaseDecisionTree`

- Minimum impurity (std) for splitting = `5% * total`

The Weka implementation slightly differs from the M5' paper

- RMSE is used to evaluate the models, instead of MAE
- new « Pruning factor » constant ?

Not implemented, and new-but-not-in-the-paper

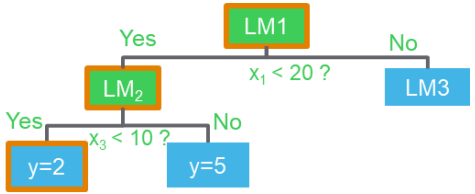
Not (yet) implemented

- The greedy linear model simplification process
- M5' enumeration & missing values handling

Benefits from using `sklearn` framework: the implementation can support any kind of models thanks to sklearn's "clone". However

- Not tested yet ☹
- Some features are disabled (e.g. smoothing can not be pre-installed)

Conclusion / Perspectives



`m5py` is an implementation of M5 / M5' in python, available now from PyPi.

- It « unlocks » the pending scikit-learn#13732 so that the community can use it and polish it.
- Many improvements can still be done, do not hesitate to contribute if you are interested !

Yet, M5' is just one of the many regression tree algorithms out there! For example:

- Yu Shi, Jian Li, and Zhize Li. 2019. Gradient boosting with piece-wise linear regression trees. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI'19)*. AAAI Press, 3432–3438. https://arxiv.org/pdf/1802.05640.pdf
- Haozhe Zhang, Dan Nettleton, and Zhengyuan Zhu. 2019. Regression-Enhanced Random Forests. https://arxiv.org/abs/1904.10416
- Igor Ilic, Berk Gorgulu, Mucahit Cevik, Mustafa Gokce Baydogan. Explainable boosted linear regression for time series forecasting. https://arxiv.org/abs/2009.09110

2 python implementations: LightGBM and linear-tree

Life Is On | Schneider Electric

Towards Sustainable and Reproducible Results

How to make sure that our AI solutions have the appropriate level of quality?

Evaluation and benchmarking

- Reference datasets: private + public
- Datasets access tools: `odsclient`, `cloudpathlib`…
- Reproducible benchmarks: `pytest`, `pytest-cases`…
- Simulation of relearning strategies

> ENSIMAG MoSIG DSAI Master student: William Templier

Actionable documentation

- Reproducible doc: `sphinx-gallery`, `mkdocs-gallery`…
- API helpers `azmlclient`, `pydantic-tables`…

“Open data” challenges

- 5 DRIVENDATA “power laws” competitions
- Microgrid energy management benchmark

Contributions to community efforts

- Scientific libraries: `scikit-learn`, `pandas`, etc.
- Tools: `pytest`, `nox`, `requests`, etc.
- Knowledge: stackoverflow, stackexchange..

Open source - Highlights

Contributions to reference scientific libraries

scikit-learn, improved R^2 and explained variance scores through a new `force_finite` parameter, 2022

pandas, improved documentation for `to_datetime` – in particular about timezone handling, 2022

scikit-learn, added a new randomized SVD solver to KernelPCA, 2021

scikit-learn, fixed Kernel PCA numerical consistency between 32-64bits, 2020

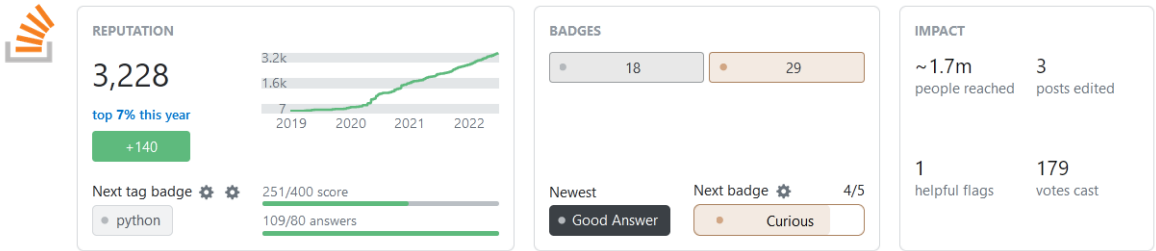
scikit-learn, statistical uniformity tests of the new libsvm/liblinear pseudo-random number generator, 2020

scikit-learn, fixed pseudo-random number generator for libsvm and liblinear, 2020

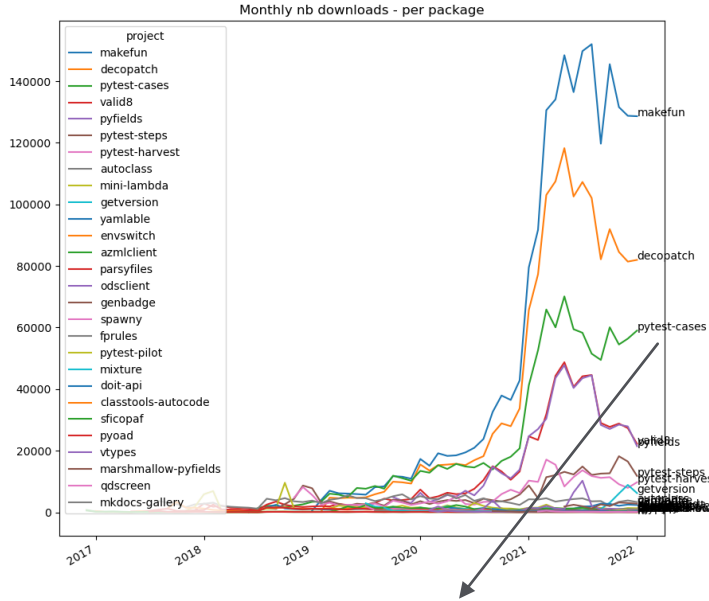
scikit-learn, added checks for eigenvalue decomposition numerical/conditioning issues in KernelPCA, 2019

scikit-learn, fixed transform issue in KernelPCA when zero eigenvalues are present and not removed, 2019

Stackoverflow / StackExchange knowledge sharing



Popularity of published new packages



Example users:

pyuvdata, ProbNum, OPENCTI, TOKERN, 3scale by RED HAT, Labelbox

Life Is On Schneider Electric