Energy Market - Data Analysis

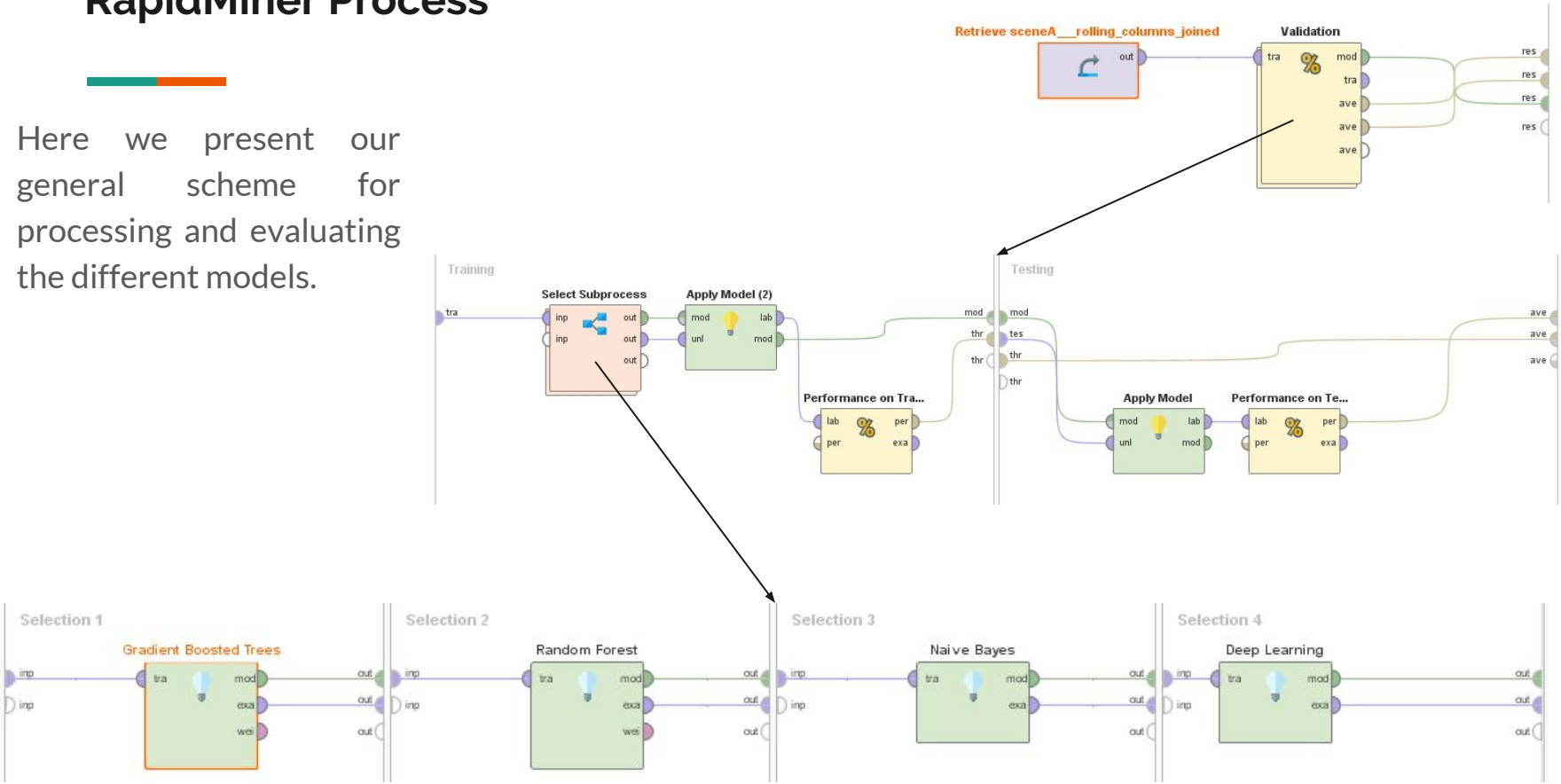
Complement Slides

André Cruz
Edgar Carneiro
Xavier Fontes

Agents and Distributed Artificial Intelligence - AIAD
Faculty of Engineering
University of Porto
2018

RapidMiner Process

Here we present our general scheme for processing and evaluating the different models.



Results from Other Experiments

We also experimented using Scikit-Learn, a machine learning library for Python*.

The scripts are in the repository and what we tested were several models that are also present in RapidMiner but whose configuration is too computationally expensive, causing RapidMiner to crash.

We tested the following models from Scikit-Learn: **Random Forest**, **Gradient Boosting**, **Naive Bayes**, **Decision Tree**, **Multi-Layer Perceptron**.

By Regular consumer we mean the consumer that is driven by the price of the energy.

* https://scikit-learn.org/stable/

Results from Other Experiments - Random Forest

Estimators	100
Criterion	Gini Index
Max Depth	100
Features	All

The configuration used was the one presented on the right and the results are shown below (recall, precision and confusion matrix).

	rolling	**even split**
Recall %	87.34	47.82
Precision %	87.45	47.94
F1 %	87.31	47.46

rolling	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	3663	242	128
True Eco	284	3359	169
True Lazy	300	297	2771

even split	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	473	219	132
True Eco	279	368	133
True Lazy	243	191	256

Results from Other Experiments - Gradient Boosting

The configuration used was the one presented on the right and the results are shown below (recall, precision and confusion matrix).

	rolling	even split
Recall %	66.87	43.59
Precision %	68.70	43.73
F1 %	66.61	42.14

Estimators	20
Loss	Deviance
Criterion	MSE
Learning Rate	0.01
Features	All
Max Depth	100

rolling	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	3259	424	350
True Eco	1113	2419	280
True Lazy	1093	455	1820

even split	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	536	167	121
True Eco	372	275	133
True Lazy	337	164	189

Results from Other Experiments - Naive Bayes

The results are shown below (recall, precision and confusion matrix).

	rolling	**even split**
Recall %	34.22	35.66
Precision %	33.73	35.08
F1 %	32.43	34.60

rolling	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	2211	789	1033
True Eco	2168	690	954
True Lazy	1839	593	936

even split	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	424	258	142
True Eco	381	246	153
True Lazy	312	230	148

Results from Other Experiments - Decision Tree

The configuration used was the one presented on the right and the results are shown below (recall, precision and confusion matrix).

	rolling	even split
Recall %	70.20	43.94
Precision %	70.20	43.98
F1 %	70.20	43.95

Max Depth	85
Splitter	Best
Criterion	Entropy

rolling	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	2894	615	524
True Eco	645	2662	505
True Lazy	548	504	2316

even split	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	363	250	211
True Eco	212	361	207
True Lazy	215	191	284

Results from Other Experiments - Multi-Layer Perceptron

The configuration used was the one presented on the right and the results are shown below (recall, precision and confusion matrix).

Hidden Layers	(32, 16, 8)
Activation	ReLU
Solver	Adam
Learning Rate	Adaptive
Learning Rate Init	0.05

	rolling	**even split**
Recall %	35.97	35.92
Precision %	12.94	12.90
F1 %	19.03	18.99

rolling	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	4033	0	0
True Eco	3812	0	0
True Lazy	3368	0	0

even split	Predicted Regular	Predicted Eco	Predicted Lazy
True Regular	824	0	0
True Eco	780	0	0
True Lazy	690	0	0

Other Observations

- The Multi-Layer Perceptron performed worse than anticipated but we believe this was due to not enough hyper-parameter experimentation and not enough data.

- Overall we had some good results and one of the advantages of Scikit-Learn + Python is its speed and memory consumption, when compared to RapidMiner.

Appendix - Data Correlation