The 2021 Belgian Stata Conference
Virtual – Monday, June 7, 2021 (09:00-09:45)

Machine Learning using Stata/Python

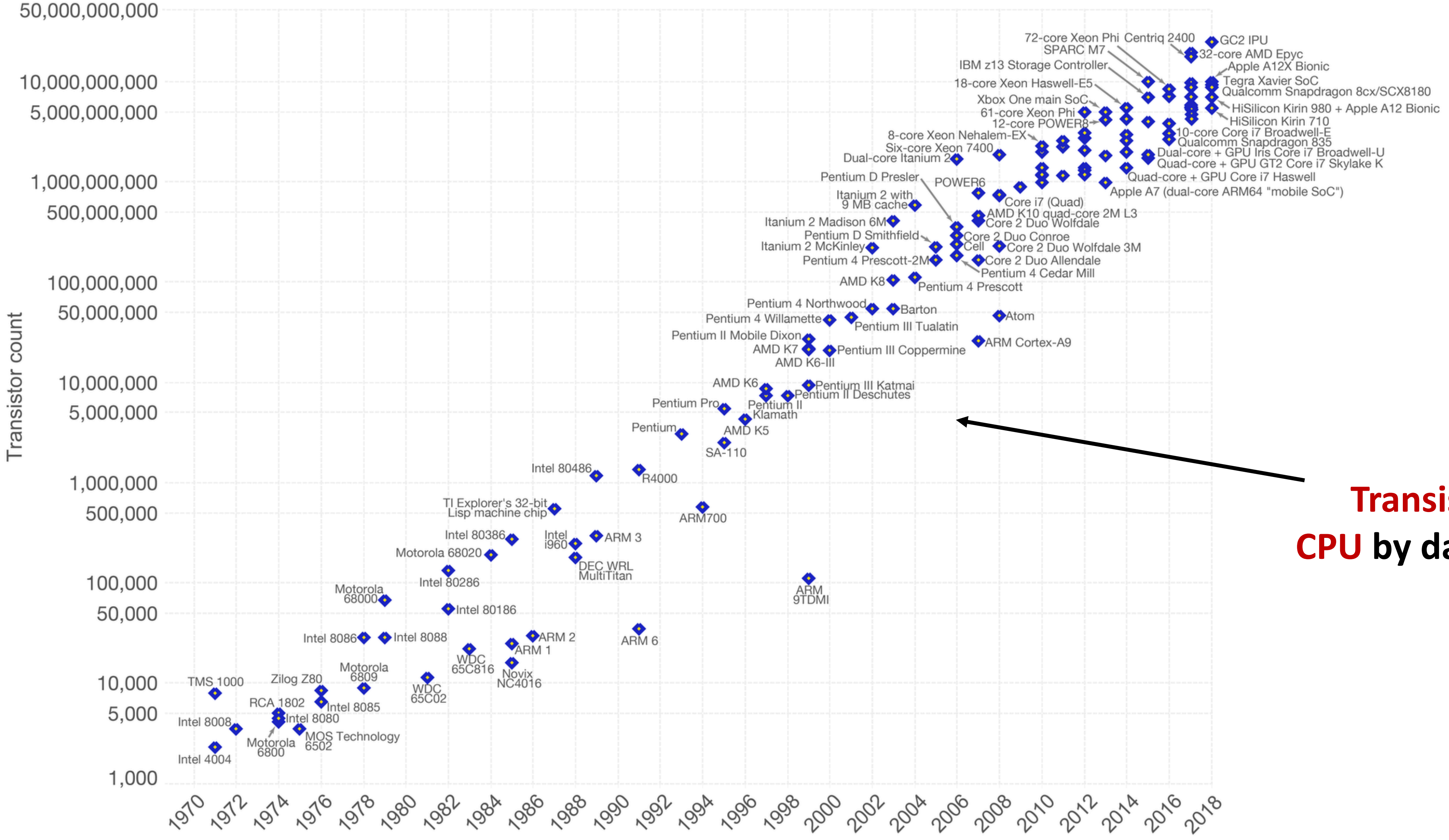
Giovanni Cerulli
IRCrES-CNR
Research Institute on Sustainable Economic Growth
National Research Council of Italy

Consiglio Nazionale delle Ricerche
IRCrES
ISTITUTO di RICERCA sulla CRESCITA ECONOMICA SOSTENIBILE
RESEARCH INSTITUTE on SUSTAINABLE ECONOMIC GROWTH

Machine Learning

Definition, relevance, applications

Growth of computer power (1971 - 2018)



Plot of Transistor counts for CPU by date of introduction

What is Machine Learning ?

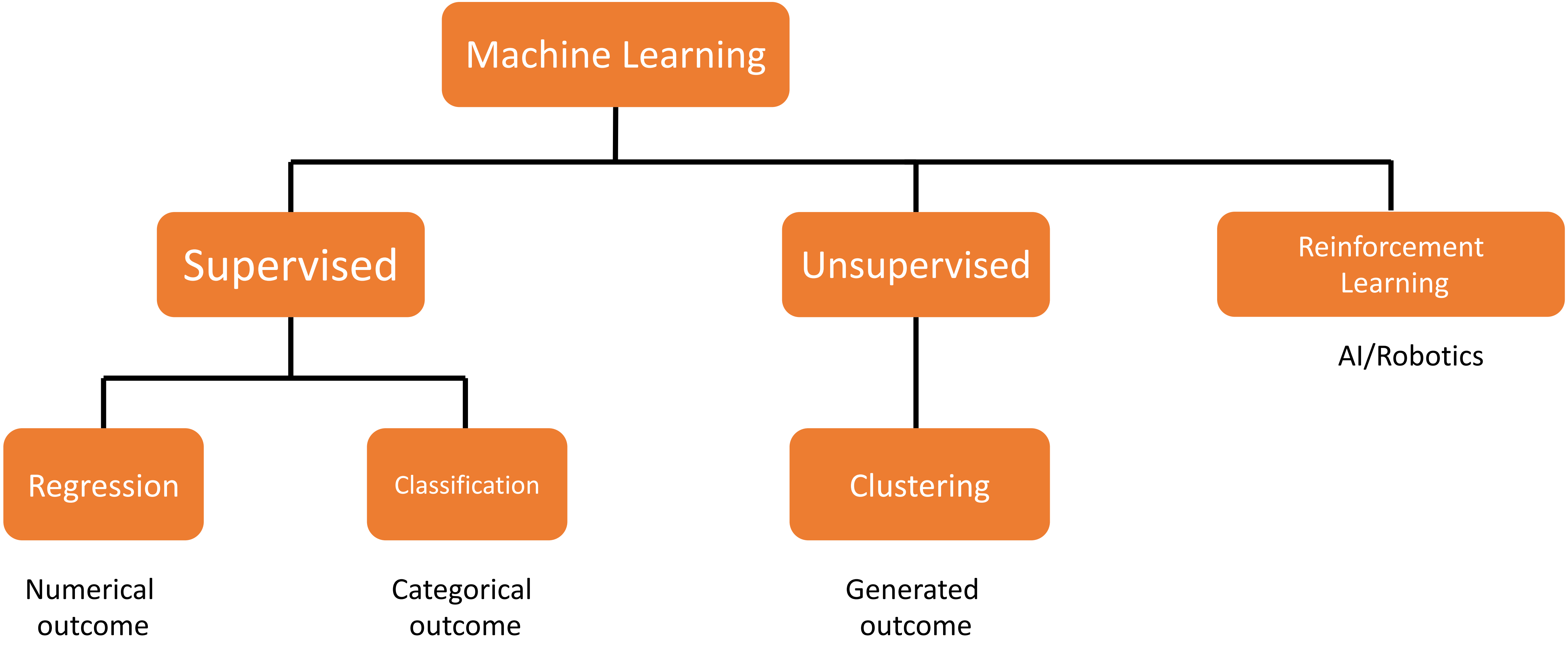
Machine Learning

A relatively new approach to **data analytics**, which places itself in the intersection between **statistics**, **computer science**, and **artificial intelligence**

ML objective

Turning **information** into **knowledge** and **value** by **“letting the data speak”**

Machine learning **taxonomy**



ML purposes

Limiting **prior assumptions**

Model-free philosophy

Based on **algorithm computation, graphics**

Mostly focused on **prediction** than **inference**

Targeted to **Big Data**

Targeted to **complexity reduction**

ML analyses

Prediction

Feature-importance detection

Signal-from-noise extraction

Correct specification via model selection

Model-free classification

Model-free clustering

Machine Learning application examples

- Identifying risk factors for prostate cancer
- Predicting heart attack by demographic, diet and clinical measurements
- Customizing email spam detection system
- Predict stock market price variation
- Self-driving cars
- Classifying pixels in a land-satellite images
- Establishing the relationship between salary and many of its determinants
- Pattern recognition of handwritten symbols
- Automated languages translators (Google Translate)
- Vocal recognition systems (Amazon Alexa)

ML
prediction

Train-MSE vs Test-MSE

Training dataset

N **in-sample** available observations

$$\text{Tr} = \{x_i, y_i\}_1^N$$

$$\text{MSE}_{\text{Tr}} = \text{Ave}_{i \in \text{Tr}}[y_i - \hat{f}(x_i)]^2$$

Overfitting as flexibility increases

Testing dataset

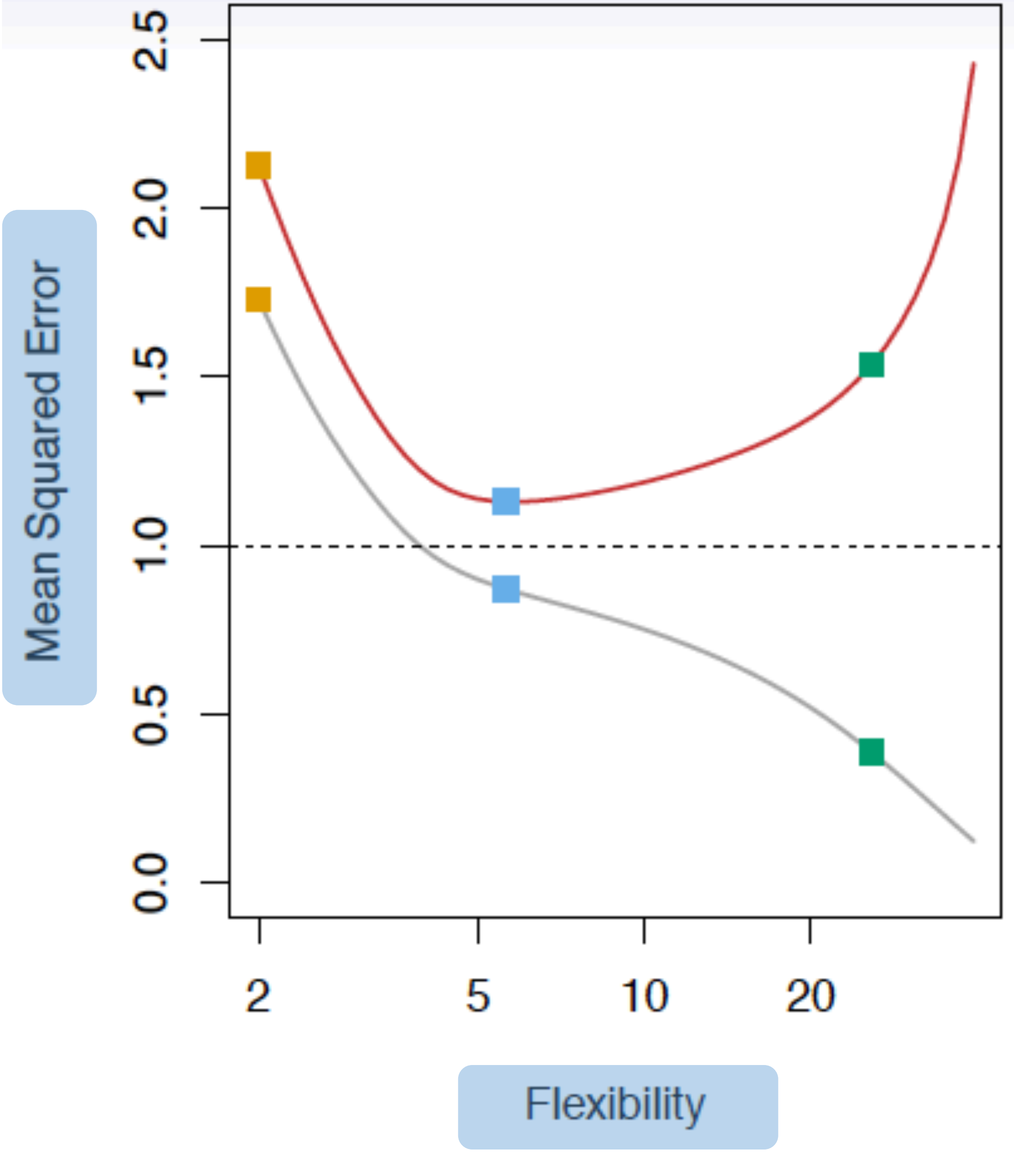
M **out-of-sample** observations

$$\text{Te} = \{x_i, y_i\}_1^M$$

$$\text{MSE}_{\text{Te}} = \text{Ave}_{i \in \text{Te}}[y_i - \hat{f}(x_i)]^2$$

True fitting accuracy

Train-MSE overfitting



Test-MSE

Train-MSE

- As long as model **flexibility** (i.e., degree-of-freedom) increases, the **train-MSE decreases** monotonically.

  This phenomenon is called **overfitting**

- On the contrary, the **test-MSE** first decreases, and then decreases, thus showing a **minimum**

Decomposition of the Test-MSE

Suppose we have fit a model $\hat{f}(x)$ to some training data Tr, and let (x_0, y_0) be a test observation drawn from the population. If the true model is $Y = f(X) + \epsilon$ (with $f(x) = E(Y|X = x)$), then

$$E\left(y_0 - \hat{f}(x_0)\right)^2 = \mathrm{Var}(\hat{f}(x_0)) + [\mathrm{Bias}(\hat{f}(x_0))]^2 + \mathrm{Var}(\epsilon).$$

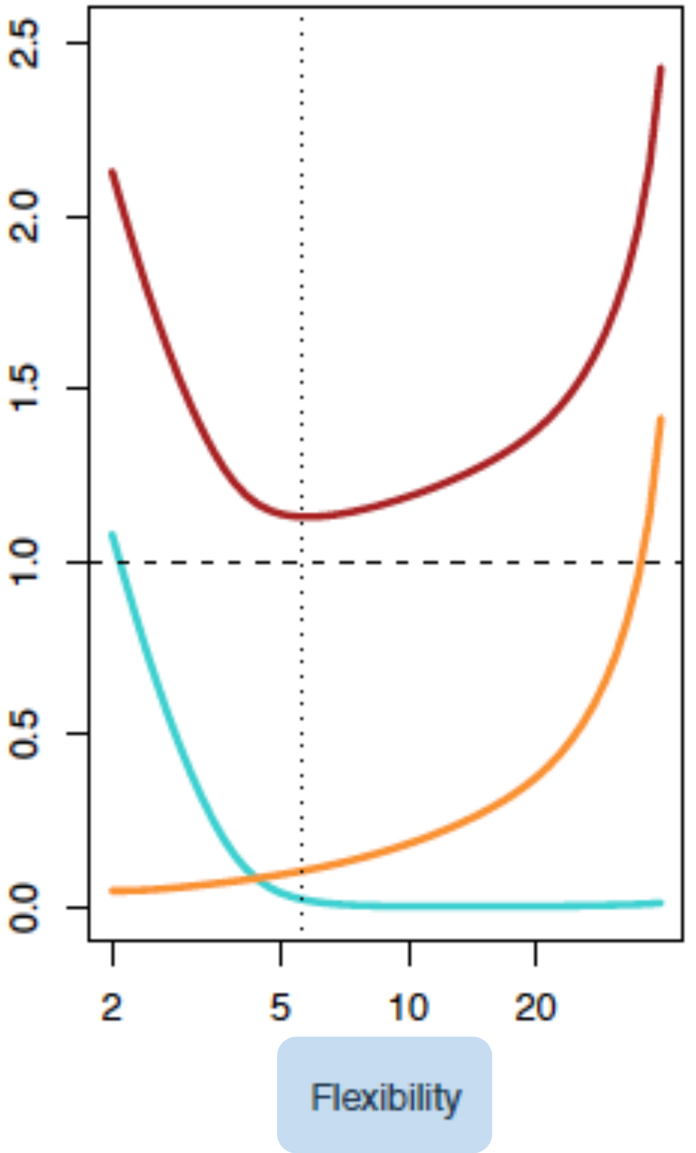
Test-MSE

Variance of the specific ML method

Bias square of the specific ML method

Variance of the Irreducible error term

The variance-bias trade-off



Typically as the *flexibility* of $\hat{f}$ increases, its variance increases, and its bias decreases. So choosing the flexibility based on average test error amounts to a *bias-variance trade-off.*

MSE
Bias
Var
------ Error variance

Observe that the error variance represents a **lower bound** for the **Test-MSE**

Timeline of Machine Learning methods

1936
Discriminant Analysis
(Fisher)

1957
Neural Network
(Rosenblatt)

1967
Nearest Neighbor Algorithm
(Cover & Hart)

1970
Backpropagation
(Linnainmaa)

1976
Classification and Regression Trees
(Breiman and Meisel)

1979
Bootstrap
(Efron)

1983
Subset Selection
(Breiman and Freedman)

1986
Backfitting and GAMs
(Hastie and Tibshirani)

1989
Convolutional Neural Network
(LeCun et al.)

1995
Support Vector Machines
(Cortes and Vapnik)

1995
Boosting
(Freund and Schapire)

1996
Lasso and Ridge Regression
(Tibshirani)

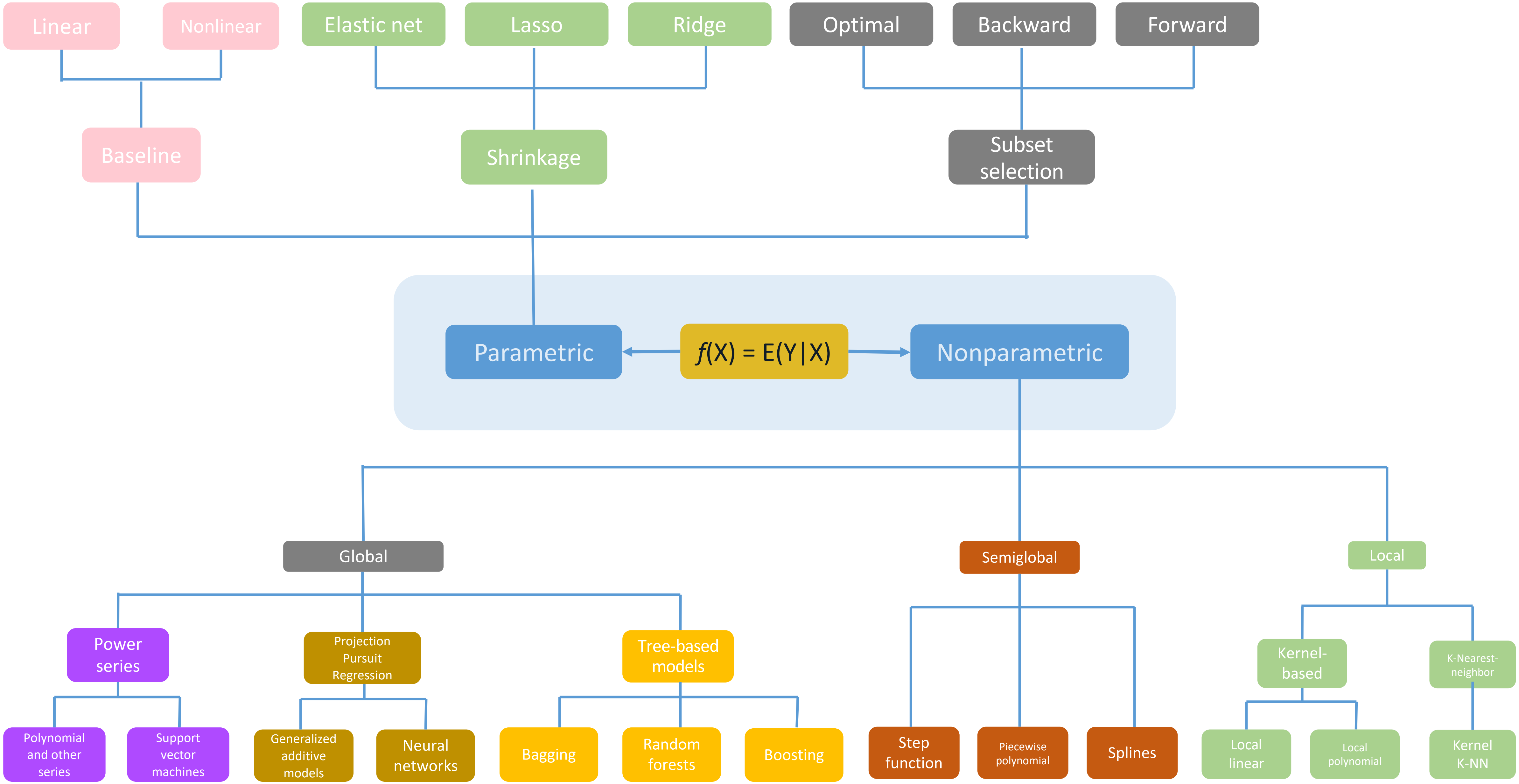
1996
Bagging
(Breiman)

2001
Random Forests
(Breiman)

2006
Deep Learning
(Hinton, Osindero, Teh)

Source: Cerulli (2019)

Machine Learning Methods



Software

Software

General purpose ML platform

Deep Learning platform

Deep Learning platform

scikit learn

TensorFlow

Keras

Software

Stata 16

R

MATLAB

Python/Stata fully integrated platform via the SFI environment

Various ML packages but poor deep learning libraries

Statistics and Machine Learning Toolbox
Deep Learning Toolbox

Python Scikit-learn platform → `c_ml_stata` & `r_ml_stata` (by G. Cerulli, 2020)

Implementation of

r_ml_stata & c_ml_stata

Stata command r_ml_stata

```
r_ml_stata outcome [varlist], mlmodel(modeltype)
           out_sample(filename) in_prediction(name)
           out_prediction(name) cross_validation(name)
           seed(integer) [save_graph_cv(name)]
```

modeltype_options	Description
Model	
elasticnet	Elastic net
tree	Regression tree
randomforest	Bagging and random forests
boost	Boosting
nearestneighbor	Nearest Neighbor
neuralnet	Neural network
svm	Support vector machine

Regression

Stata command c_ml_stata

```
c_ml_stata outcome [varlist], mlmodel(modeltype)
           out_sample(filename) in_prediction(name)
           out_prediction(name) cross_validation(name)
           seed(integer) [save_graph_cv(name)]
```

modeltype_options	Description
Model	
tree	Classification tree
randomforest	Bagging and random forests
boost	Boosting
regularizedmultinomial	Regularized multinomial
nearestneighbor	Nearest Neighbor
neuralnet	Neural network
naivebayes	Naive Bayes
svm	Support vector machine
multinomial	Standard multinomial

Classification

Practical implementation (in 8 steps)

Set the stage

Step 1. Before starting, install Python (from version 2.7 onwards), and the Python packages ***scikit-learn***, ***numpy***, and ***pandas***. If you want suggestion on how to install Python and its packages look here.

Step 2. Once you have Python installed in your machine, you need to install the Stata ML command:

. ssc install r_ml_stata

and look at the documentation file of the command to explore its syntax:

. help r_ml_stata

Step 3. The command requires to provide a dataset with *no missing values*. It is your responsibility to assure this. We can thus load the ***training dataset*** I have prepared for this example:

. use "r_ml_stata_data_example"

This dataset contains one target variable (y) and 13 features (x1, x2, ... , x13). All variables are numerical and thus suitable for running our regression tree.

- Before running the command a **testing dataset** must be provided

- This is a dataset made of the same features of the training one, but with "new" instances. Observe that this dataset must neither contain missing values, nor include the target variable (y)

- Here, we consider a testing dataset called "**r_ml_stata_data_new_example**"

Step 4. We have now all the ingredients to run our regression tree. We simply run these lines of code in Stata:

```
. r_ml_stata y x1-x13 , mlmodel("tree") in_prediction("in_pred")
cross_validation("CV") out_sample("r_ml_stata_data_new_example")
out_prediction("out_pred") seed(10) save_graph_cv("graph_cv")
```

Meaning of the syntax

- **"tree"** tells Stata to run a "tree" regression. Other options are available (see the help-file)
- **"in_pred"** tells Stata to generate a dataset **"in_pred.dta"** containing the in-sample predictions of the estimated model. They are the prediction only for the training dataset
- **"out_pred"** tells Stata to generate a dataset **"out_pred.dta"** containing the out-of-sample predictions of the estimated model. They are predictions only for the testing dataset
- **"r_ml_stata_data_new_example"** tells Stata to use this one as testing dataset
- **"seed(10)"** necessary to replicate the same results and must be an integer
- **"graph_cv"** tells Stata to save the cross-validation results graph in your current directory

Step 5. In order to access the main results, we can look at the command's "**ereturn**" list by typing:

. ereturn list

```
--------------------------------------------------------------------------------
scalars:
e(OPT_LEAVES) = 4
e(TEST_ACCURACY) = .2027650052251946
e(TRAIN_ACCURACY) = .8911061692860425
e(BEST_INDEX) = 3
--------------------------------------------------------------------------------
```

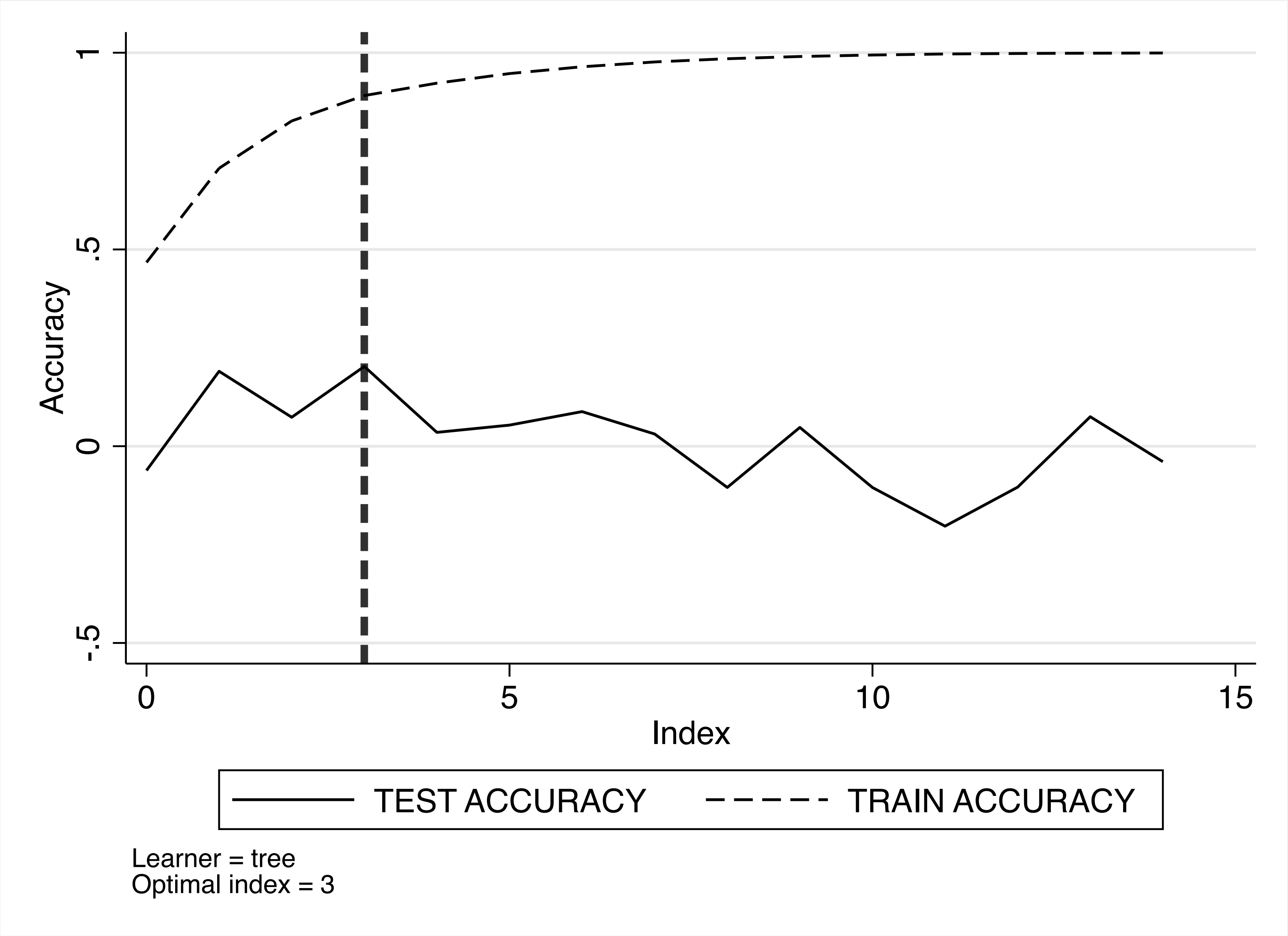
We observe that the cross-validated (CV) optimal number of leaves (namely, the tree optimal final nodes) is 4, the CV optimal train accuracy is 0.89, while the CV optimal test accuracy is much smaller, i.e. 0.20. The accuracy measure is the share of the total outcome variance explained by the model (it is closely similar to an adjusted R-squared)

Step 6. The command provides a graphical representation of the 10-fold cross-validation with the optimal grid search index

At this index, the test accuracy is maximum (over the grid). It is also useful to observe the **overfitting pattern** of the train accuracy going to one (maximum accuracy) as long as the model complexity increases.

This phenomenon justifies ML focus on just the test accuracy which shows, in this graph, a clear variance-bias trade-off.

CV graphical representation



Step 7. We can go even deeper into the understanding of the cross-validation results, by opening the CV results' dataset "CV.dta" and list its content:

```
. use CV , clear

. list
```

	index	mean_tr~e	mean_tes~e	std_tes~e
1.	0	.46707705	-.06167094	.39788509
2.	1	.70630139	.19044095	.4556592
3.	2	.82658573	.0736554	.81239835
4.	3	.89110617	.20276501	.7425186
5.	4	.9226751	.03514619	1.106288
6.	5	.94706553	.07042505	.80642391
7.	6	.9643006	.08797992	.8920612
8.	7	.97651532	.04474171	.90139242
9.	8	.98468366	-.0911493	1.0005497
10.	9	.99037207	-.07410576	.95875368
11.	10	.99419796	.00047517	.99491587
12.	11	.99669345	-.19575196	1.2006688
13.	12	.99800368	-.03616092	.91373416
14.	13	.99881614	-.12245656	.97095934
15.	14	.99929195	-.1099525	.9330861

Results show, by every grid index, the **train accuracy**, the **test accuracy**, and the **standard error** of the test accuracy estimated over the 10-fold runs.

The standard error is important, as it measures the precision we get when estimating the test accuracy. In this example, at the optimal index (i.e., 3), the test accuracy's standard error is 0.74, which should be compared with those obtained from other ML algorithms.

This means that the choice of the ML model to employ for prediction purposes should ponder not only the level of the achieved test accuracy, but also its standard error.

Step 8. Finally, we can have a look at the out-of-sample predictions. This can be done by opening and listing the "out_pred" dataset:

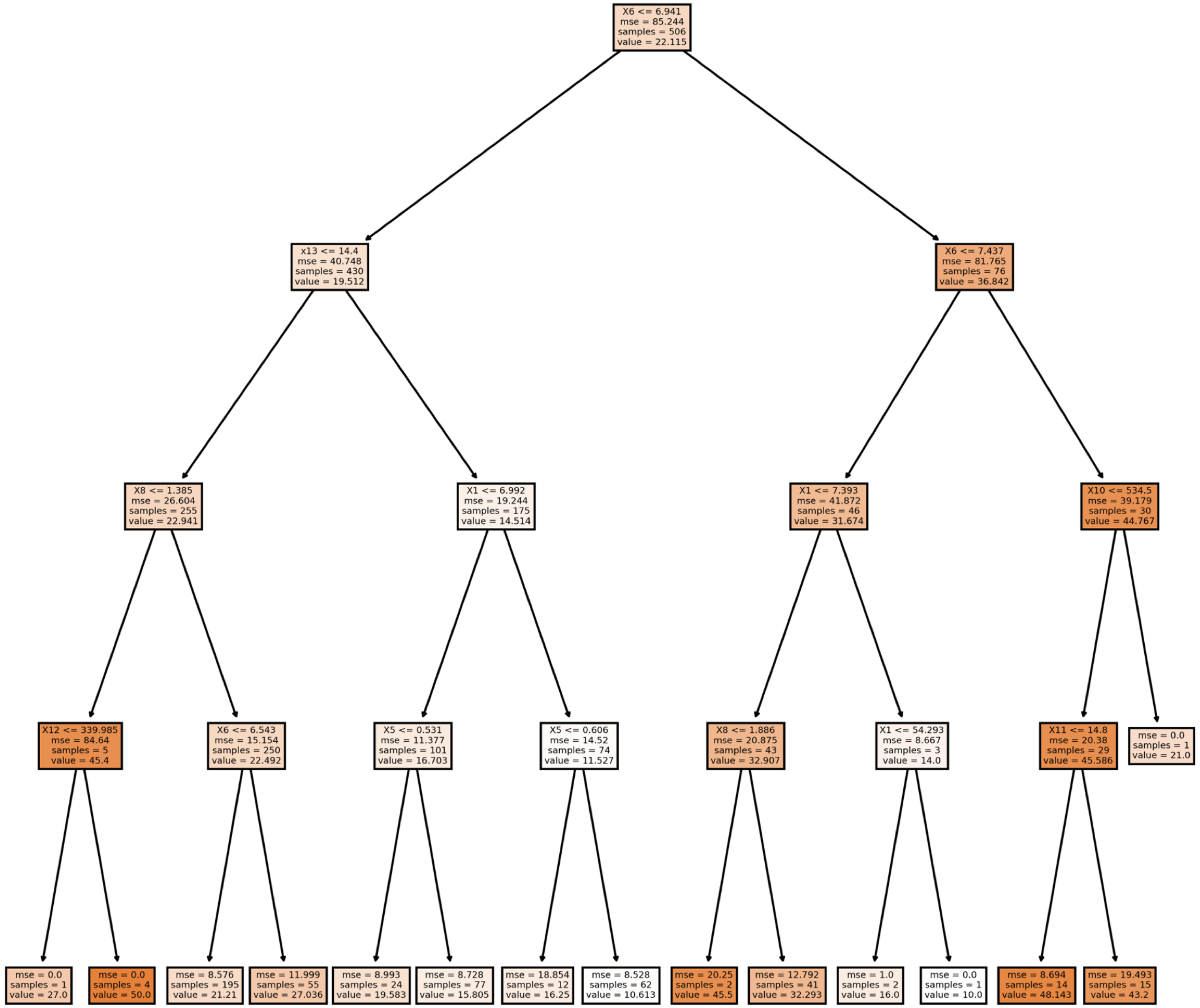
. use out_pred , clear

. list

	index	out_sam~d
1.	0	21.629744
2.	1	16.238961
3.	2	21.629744
4.	3	16.238961
5.	4	16.238961
6.	5	16.238961
7.	6	16.238961
8.	7	16.238961
9.	8	16.238961
10.	9	21.629744
11.	10	27.427273

We observe that the predictions are made of only three values [21.62, 16.23, 27.42] corresponding to three out of the four optimal terminal tree leaves. Graphically it represents a step-function (omitted for the sake of brevity).

Optimal tree (depth = 4)



References

- Cerulli, G. 2020. *C_ML_STATA: Stata module to implement machine learning classification in Stata*. Statistical Software Components, Boston College Department of Economics. Available at: https://econpapers.repec.org/software/bocbocode/s458830.htm
- Cerulli, G. 2020. *R_ML_STATA: Stata module to implement machine learning regression in Stata*. Statistical Software Components, Boston College Department of Economics. Available at: https://econpapers.repec.org/software/bocbocode/s458831.htm
- Cerulli, G. 2020. *A super-learning machine for predicting economic outcomes*, MPRA Paper 99111, University Library of Munich, Germany, 2020
- Cerulli, G. 2020. Improving econometric prediction by machine learning, *Applied Economics Letters*, Forthcoming.
- Gareth, J., Witten, D., Hastie, D.T., Tibshirani, R. 2013. *An Introduction to Statistical Learning : with Application in R*. New York, Springer
- Raschka, S., Mirjalili, V. 2019. *Python Machine Learning*. 3rd Edition, Packt Publishing.