Machine Learning using Stata/Python

Giovanni Cerulli
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”
Supervised, Unsupervised, Reinforcement Learning

- Machine learning
  - Supervised learning
    - Classification
      - Outcome Categorical
  - Unsupervised learning
    - Regression
      - Outcome Continuous
    - Clustering
      - No Outcome
  - Reinforcement learning
    - Classification
    - Outcome
      - Reward

Artificial intelligence
Supervised Machine Learning Methods

**Parametric**
- $f(X) = E(Y|X)$

**Nonparametric**

- **Global**
  - Power series
    - Polynomial, additive models
  - Support vector machines
  - Generalized additive models
  - Neural networks
  - Bagging
  - Random forests
  - Boosting
  - Tree-based models

- **Semiglobal**
  - Step function
    - Piecewise polynomial
  - Splines

- **Local**
  - Kernel-based
    - Linear, polynomial
  - K-Nearest-neighbor

- **Subset selection**
  - Optimal
  - Backward
  - Forward

- **Shrinkage**
  - Elastic net
  - Lasso
  - Ridge
  - Shrinkage
  - Elastic net
  - Ridge
  - Lasso

- **Baseline**
  - Linear
  - Nonlinear

- **Projection Pursuit Regression**

- **Generalized additive models**

- **Neural networks**

- **Bagging**

- **Random forests**

- **Boosting**

- **Step function**
  - Piecewise polynomial

- **Splines**

- **Local linear**

- **Local polynomial**

- **Kernel K-NN**
## Hyper-parameter tuning

<table>
<thead>
<tr>
<th>ML method</th>
<th>Parameter 1</th>
<th>Parameter 2</th>
<th>Parameter 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Models and GLS</td>
<td>N. of covariates</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Lasso</td>
<td>Penalization coefficient</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Elastic-Net</td>
<td>Penalization coefficient</td>
<td></td>
<td>Elastic parameter</td>
</tr>
<tr>
<td>Nearest-Neighbor</td>
<td>N. of neighbors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Neural Network</td>
<td>N. of hidden layers</td>
<td>N. of neurons</td>
<td></td>
</tr>
<tr>
<td>Trees</td>
<td>N. of leaves</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boosting</td>
<td>Learning parameter</td>
<td>N. of bootstraps</td>
<td>N. of leaves</td>
</tr>
<tr>
<td>Random Forest</td>
<td>N. of features for splitting</td>
<td>N. of bootstraps</td>
<td>N. of leaves</td>
</tr>
<tr>
<td>Bagging</td>
<td>Tree-depth</td>
<td>N. of bootstraps</td>
<td></td>
</tr>
<tr>
<td>Support Vector Machine</td>
<td>C</td>
<td></td>
<td>Gamma</td>
</tr>
<tr>
<td>Kernel regression</td>
<td>Bandwidth</td>
<td></td>
<td>Kernel function</td>
</tr>
<tr>
<td>Piecewise regression</td>
<td>N. of knots</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Series regression</td>
<td>N. of series terms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Software for ML
Software

General purpose ML platform

Deep Learning platform

Deep Learning platform

Software

scikit learn
TensorFlow
Keras
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)
Classification
Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

Regression
Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

Clustering
Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...
Stata’s Python API documentation

The Stata Function Interface (sfi) module allows users to interact Python’s capabilities with core features of Stata. The module can be used interactively or in do-files and ado-files.

Within the module, classes are defined to provide access to Stata’s characteristics, current dataset, frames, date and time, macros, scalars, matrices, value labels, global Mata matrices, missing values, etc.

Class Summary

- Characteristic (sfi.Characteristic)
- Data (sfi.Data)
- Datetime (sfi.Datetime)
- Frame (sfi.Frame)
- Macro (sfi.Macro)
- Mata (sfi.Mata)
- Matrix (sfi.Matrix)
- Missing (sfi.Missing)
- Platform (sfi.Platform)
- Preference (sfi.Preference)
- Scalar (sfi.Scalar)
- SFIToolkit (sfi.SFIToolkit)
- StrLConnector (sfi.StrLConnector)
- ValueLabel (sfi.ValueLabel)
ML regression and classification with

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)]
```

<table>
<thead>
<tr>
<th>modeltype_options</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Regression</td>
</tr>
<tr>
<td>elasticnet</td>
<td>Elastic net</td>
</tr>
<tr>
<td>tree</td>
<td>Regression tree</td>
</tr>
<tr>
<td>randomforest</td>
<td>Bagging and random forests</td>
</tr>
<tr>
<td>boost</td>
<td>Boosting</td>
</tr>
<tr>
<td>nearestneighbor</td>
<td>Nearest Neighbor</td>
</tr>
<tr>
<td>neuralnet</td>
<td>Neural network</td>
</tr>
<tr>
<td>svm</td>
<td>Support vector machine</td>
</tr>
</tbody>
</table>
Stata command \texttt{c\_ml\_stata}

\texttt{c\_ml\_stata outcome [varlist], mlmodel(modeltype)}

\texttt{out\_sample(filename) in\_prediction(name)}

\texttt{out\_prediction(name) cross\_validation(name)}

\texttt{seed(integer) [save\_graph\_cv(name)]}

\begin{tabular}{ll}
\hline
\textit{modeltype\_options} & Description \\
\hline
Model & \\
\texttt{tree} & Classification tree \\
\texttt{randomforest} & Bagging and random forests \\
\texttt{boost} & Boosting \\
\texttt{regularizedmultinomial} & Regularized multinomial \\
\texttt{nearestneighbor} & Nearest Neighbor \\
\texttt{neuralnet} & Neural network \\
\texttt{naivebayes} & Naive Bayes \\
\texttt{svm} & Support vector machine \\
\texttt{multinomial} & Standard multinomial \\
\end{tabular}

\textbf{Classification}
Practical implementation

Nearest neighbor regression
DESCRIPTION OF THE DATASET

The dataset is available through Hastie et al. (2009) on the authors' website

Training dataset: "prostate.dta"

The following variables are included in the dataset

Predictors (or features)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lpsa</td>
<td>Log(prostate-specific antigen)</td>
</tr>
<tr>
<td>lweight</td>
<td>Log(prostate weight)</td>
</tr>
<tr>
<td>age</td>
<td>Patient age</td>
</tr>
<tr>
<td>lbph</td>
<td>Log(benign prostatic hyperplasia amount)</td>
</tr>
<tr>
<td>svi</td>
<td>Seminal vesicle invasion</td>
</tr>
<tr>
<td>lcp</td>
<td>Log(capsular penetration)</td>
</tr>
<tr>
<td>gleason</td>
<td>Gleason score</td>
</tr>
<tr>
<td>pgg45</td>
<td>Percentage Gleason scores 4 or 5</td>
</tr>
</tbody>
</table>

Outcome (or target)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lcavol</td>
<td>Log(cancer volume)</td>
</tr>
</tbody>
</table>
Clear all
clear all

Set the directory
cd "\Users/giocer/Desktop/output"

Set the "learner"
global learner "nearestneighbor"

Load the dataset
sysuse "prostate.dta", clear

Set "target" (y) and "features" (X)
global y "lcavol"
global X "lpsa lweight age lbph svi lcp gleason pgg45"

Split sample into "training" and "testing" datasets
splitsample, generate(vsplit, replace) split(0.80 0.20) show rseed(1010)
* Form the "training" dataset
preserve
keep if vsplit==1
drop vsplit
save data_train, replace
restore

* Form the "testing" dataset
preserve
keep if vsplit==2
drop $y
drop vsplit
save data_test, replace
restore
* Form a dataset containing only the "y" of the testing dataset

```
preserve
keep if vsplit==2
keep $y
gen index=_n-1
save test_y, replace
restore

* Open the "training" dataset
```

```
use data_train, clear
```
Run a ML regression using "r_ml_stata"
```
r_ml_stata $y$ $X$, mlmodel($learner$) in_prediction("in_pred") ///
cross_validation("CV") out_sample("data_test") ///
out_prediction("out_pred") seed(10) save_graph_cv("graph_cv")
```

Explore the results
```
return list
```

Scalars:
```
e(OPT_NN) = 27
e(TEST_ACCURACY) = -.1116904556751251
e(TRAIN_ACCURACY) = .217652040719986
e(BEST_INDEX) = 52
e(SE_TEST_ACCURACY) = .2502414777390628
```

Macros:
```
e(OPT_WEIGHT) : "uniform"
```
10-fold cross-validation results

Learner = nearestneighbor
Optimal index = 52
* Plot the in-sample predictions
use in_pred, clear
gen id = _n
sort id
tw (line $y id, lc(green)) ///
   (line in_pred id, lc(orange)), ///
xtitle("Units' identifier") ///
legend(order(1 "Actual" 2 "In-sample prediction")) ///
note(LEARNER: $learner) ///
plotregion(style(none)) scheme(s1mono)
In-sample predictions

LEARNER: nearestneighbor
* Plot the out-of-sample predictions

```stata
use out_pred, clear
merge 1:1 index using "test_y"
tw (line $y index, lc(green)) ///
    (line out_sample_pred index, ///
    lc(orange)), xtitle("Units' identifier") ///
legend(order(1 "Actual" 2 "Out-sample prediction")) ///
note(LEARNER: $learner) ///
plotregion(style(none)) scheme(s1mono)
```
Out-of-sample prediction

LEARNER: nearestneighbor
Example

Comparing multiple learners

Guessing whether a “new” car is a “foreign” or “domestic” one based on a series of characteristics, including price, number of repairs, weight, etc
Machine Learning using Stata/Python

Giovanni Cerulli
Cross-validation maximum of the classification test accuracy over a grid of learners' tuning parameters.

Accuracy measure: “error rate”
Comparing learner performance

Forest plot for comparing mean and standard deviation of different learners. Classification setting
Improving econometric prediction by machine learning

Giovanni Cerulli
References


