Roots from Trees:

A Machine Learning Approach to Unit Root Detection

Gary Cornwall
Bureau of Economic Analysis

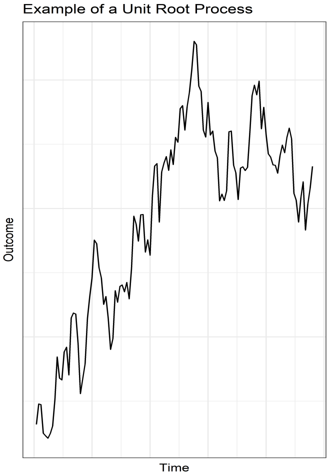
Jeff Chen
Bennett Institute for Public Policy
University of Cambridge

Beau Sauley
Murray State University

October 20, 2021

Motivation

- The [unresolved] presence of a unit root in time series can produce "nonsense regressions" (Granger & Newbold, 1974).
- Dozens of tests have since been developed to detect unit roots under a variety of settings (*e.g.*, panel data, structural breaks, etc.).
- Tests don't always agree but each tells you something about the series, how does a practitioner weigh the evidence?



Research Question(s)

1. Can we draw a link between a single unit root test and a "weak learner".
2. If a test can be considered a weak learner, can we exploit between test variation using modern machine learning algorithms to better identify unit root processes?

Quick Answers

1. Can we draw a link between a single unit root test and a "weak learner". Yes, in fact these are equivalent in both single and two-tailed tests for some $\alpha = \alpha'$.
2. If a test can be considered a weak learner, can we exploit between test variation using modern machine learning algorithms to better identify unit root processes? Yes, since we know how to aggregate weak base learners and create more powerful ensemble prediction methods we can use tools such as random forests and gradient boosting to improve unit root test accuracy.

Why Unit Roots?

- The unit root problem is a difficult time series econometrics problem which has produced nearly five decades of research and many different test statistics.
- The test for unit roots is important because failing to identify a unit root can invalidate all subsequent inferences (Granger & Newbold, 1974).
- Co-integrated relationships between series means you can't just assume everything has a unit root (Granger, 1981; Engle and Granger, 1987).
- The difficulty comes from differentiating unit roots from *near unit roots*, as a result these test statistics have low power (Ng & Perron, 2001)

What is a Unit Root?

Let y_t be an autoregressive time series generated such that,

$$y_t = \phi y_{t-1} + \epsilon_t, \quad t = (1, \ldots T)$$

- We assume $\epsilon_t \sim N(0, \sigma^2)\ \forall\ t$ and that $\sigma_1^2 = \ldots = \sigma_T^2$.
- We can write this as $(1 - \phi L)y_t = \epsilon_t$ such that $Ly_t = y_{t-1}$.
- $(1 - \phi L)$ has a root of $1/\phi$ and if $|\phi| < 1$ then y_t is considered stationary.
- Tests are often using an $H_0 : \phi = 1$ and $H_1 : |\phi| < 1$ structure (*e.g.* Augmented Dickey Fuller test).

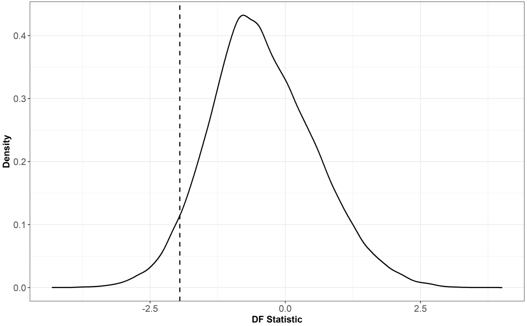
How do we test for a Unit Root?

- The Dickey-Fuller test statistic:

$$\hat{\tau} = (\hat{\phi} - 1)S_e^{-1}\left(\sum_{t=2}^{N} Y_{t-1}^2\right)^{1/2},$$

$$S_e^{-1} = (n-2)^{-1}\sum_{t=2}^{N}(Y_t - \hat{\phi}Y_{t-1})^2,$$

  with limiting distribution outlined in Dickey & Fuller (1979).
- Calculated on first difference of Y with $H_0: \phi = 1$ and $H_1: |\phi| < 1$.



How do we test for a Unit Root?

- Different assumed DGPs result in different null distributions and decision thresholds:[1]

$$y_t = \lambda + \phi y_{t-1} + \delta t + \epsilon_t \rightarrow x_\alpha = -3.45|\alpha = 0.05$$
$$y_t = \lambda + \phi y_{t-1} + \epsilon_t \rightarrow x_\alpha = -2.89|\alpha = 0.05$$
$$y_t = \phi y_{t-1} + \epsilon_t \rightarrow x_\alpha = -1.95|\alpha = 0.05$$

[1] Not everyone even agrees on the decision thresholds for the same test (e.g., Banerjee, et al., 1993 versus Hamilton, 1994 versus MacKinnon, 2010!

How do we test for a Unit Root?

- Different assumed DGPs result in different null distributions and decision thresholds:[1]

$$y_t = \lambda + \phi y_{t-1} + \delta t + \epsilon_t \rightarrow x_\alpha = -3.45 | \alpha = 0.05$$
$$y_t = \lambda + \phi y_{t-1} + \epsilon_t \rightarrow x_\alpha = -2.89 | \alpha = 0.05$$
$$y_t = \phi y_{t-1} + \epsilon_t \rightarrow x_\alpha = -1.95 | \alpha = 0.05$$

- Choice of DGP opens up an additional error path beyond Type I and Type II errors.

[1] Not everyone even agrees on the decision thresholds for the same test (e.g., Banerjee, et al., 1993 versus Hamilton, 1994 versus MacKinnon, 2010!

How do we test for a Unit Root?

- Different assumed DGPs result in different null distributions and decision thresholds:[1]

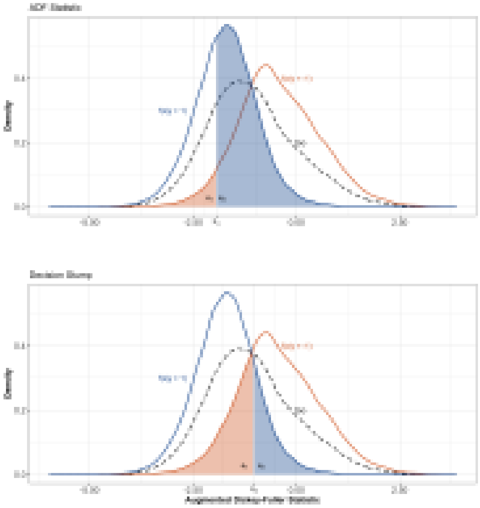
$$
\begin{aligned}
y_t = \lambda + \phi y_{t-1} + \delta t + \epsilon_t &\rightarrow x_\alpha = -3.45 | \alpha = 0.05 \\
y_t = \lambda + \phi y_{t-1} + \epsilon_t &\rightarrow x_\alpha = -2.89 | \alpha = 0.05 \\
y_t = \phi y_{t-1} + \epsilon_t &\rightarrow x_\alpha = -1.95 | \alpha = 0.05
\end{aligned}
$$

- Choice of DGP opens up an additional error path beyond Type I and Type II errors.
- There are many tests that are similarly structured, *e.g.*, ADF (Dickey and Fuller, 1981), PP (Phillips and Perron, 1988), KPSS (Kwiatkowski et al., 1992), PGFF (Pantula et al., 1994), Breit (Breitung, 2002; Breitung and Taylor, 2003), ERS (Elliot et al., 1996), URSP (Schmidt and Phillips, 1992), and URZA (Zivot and Andrews, 2002).

[1] Not everyone even agrees on the decision thresholds for the same test (e.g., Banerjee, et al., 1993 versus Hamilton, 1994 versus MacKinnon, 2010!

Roots are Stumps

- In both cases a decision is being made over a shared support of $\mathcal{X}$.
- For some $\alpha = \alpha'$ it must be the case that $x_0 = x_\alpha$.
- $\Big((h(x) \equiv g(x))|\alpha = \alpha'\Big)$ where $h(x)$ is the decision stump and $g(x)$ is the Unit Root test.
- $x_0 \approx -1.03$ which means $\alpha = \alpha' \approx 0.273$.



A Simple Procedure for Composite Test Construction

1. Simulate a balanced training, validation, and test set containing representative cases of the null and alternative hypotheses
2. Derive transmitters from one or multiple test statistics and attributes of the time series
3. Train a set of supervised classifiers, then select the model that fairs the best in cross-validation
4. Finally, conditional upon some desired Type I error rate, α, or error cost ratio, $c(e_2)/c(e_1)$, return a class prediction for the series in question.

Simulate a balanced, representative data set

For any hypothesis test we can write down a DGP which will satisfy the null, *e.g.* unit roots.

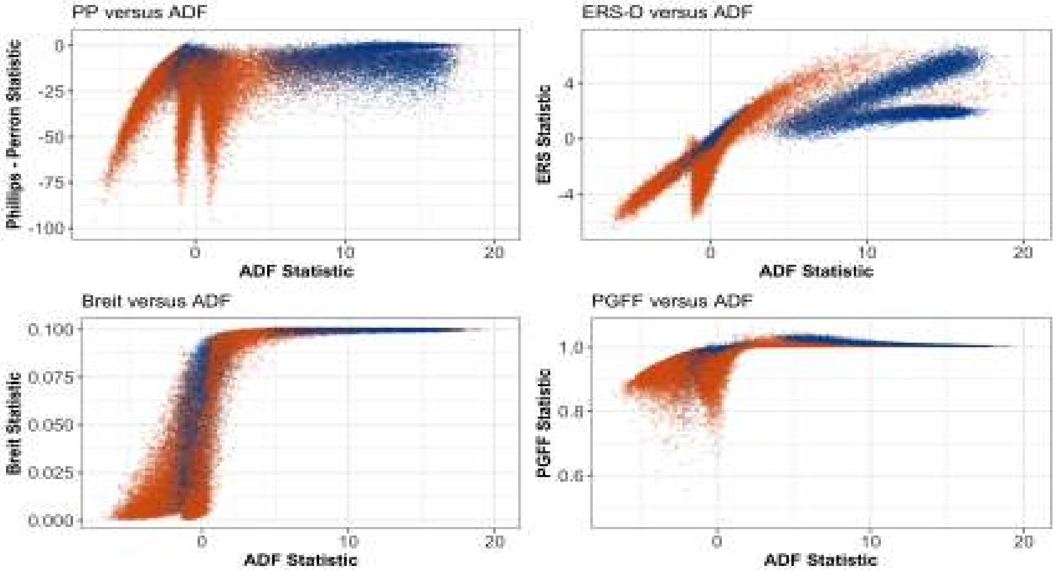
1. Generate $500,000$ time series with $350,000$ for training, $75,000$ for validation, and $75,000$ for testing.
2. A series will contain a unit root, that is $\phi = 1$ with probability 0.50 and $\phi \in \{0.9000, 0.9999\}$ otherwise.
3. Series will be uniformly distributed over the three unit root DGPs mentioned earlier.
4. All noise is Gaussian white noise.

What are the features?

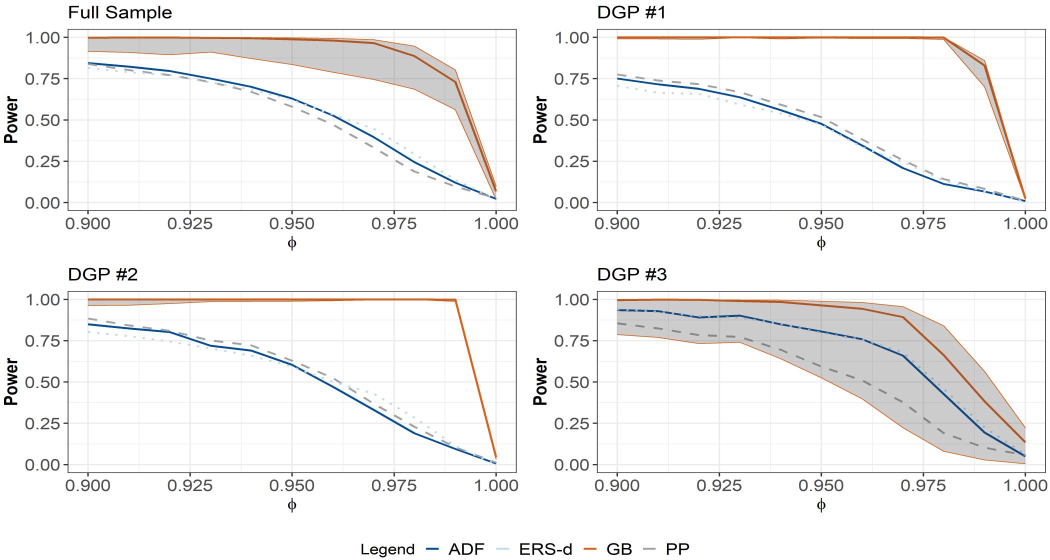
UR Tests	Level and First Difference	STL Decomposed Series	Miscellaneous
ADF	Skewness	TNN Test	Length
PP	Kurtosis	Skewness	Frequency
PGFF	Box Statistic	Kurtosis	$\text{var}(\Delta y)/\text{var}(y)$
KPSS	Lyapunov Exponent	Box Statistic	
ERS (d & p)	TNN Test		
URSP	Hurst Exponent		
URZA	Strength of Trend		
Breit	Strength of Seasonality		

While we generate the data from one of the three possible "cases" outlined in the literature all test statistics are calculated on the most parsimonious DGP assumption possible, *e.g.* no drift or trend for the ADF.

Is there variation in our features?



Power Curves



Empirical Example

- Revisit 14 macro indicators from Nelson & Plosser (1982) which serve as a common benchmarking data set for unit root studies.
- Original paper indicated that, of the 14 series, only Unemployment Rate was stationary.
- Subsequent studies by Perron (1989), Stock (1991), Kwiatkowski, et. al. (1992), Andrews & Chen (1994), and Charles & Darne (2012) all introduced significant disagreement.
- We find that, depending on the desired Type I error rate, between 11 indicators ($\alpha = 0.10$) and 2 indicators ($\alpha = 0.01$) can be considered stationary.

Package Preview

```r
15 out01 <- ml_test(nelson_plosser_data, pvalue = .01)
16 out05 <- ml_test(nelson_plosser_data, pvalue = .05)
17 out10 <- ml_test(nelson_plosser_data, pvalue = .10)
```

```
> out01$verdicts
   series_no pvalue_requested pvalue_returned                   type_of_result verdict_xgbTree
1      gnp.r             0.01            0.01 Using exact p-value threshold       unit root
2      gnp.n             0.01            0.01 Using exact p-value threshold       unit root
3     gnp.pc             0.01            0.01 Using exact p-value threshold      stationary
4         ip             0.01            0.01 Using exact p-value threshold       unit root
5        emp             0.01            0.01 Using exact p-value threshold      stationary
6         ur             0.01            0.01 Using exact p-value threshold       unit root
7      gnp.p             0.01            0.01 Using exact p-value threshold       unit root
8        cpi             0.01            0.01 Using exact p-value threshold       unit root
9       wg.n             0.01            0.01 Using exact p-value threshold       unit root
10      wg.r             0.01            0.01 Using exact p-value threshold       unit root
11         M             0.01            0.01 Using exact p-value threshold       unit root
12       vel             0.01            0.01 Using exact p-value threshold       unit root
13       bnd             0.01            0.01 Using exact p-value threshold       unit root
14        sp             0.01            0.01 Using exact p-value threshold       unit root
```

Package Preview

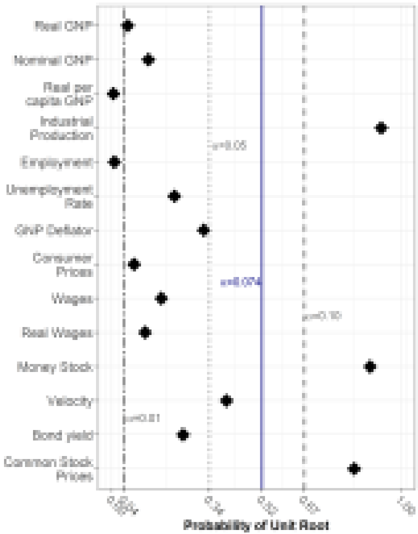
```
> thresholds_display
```

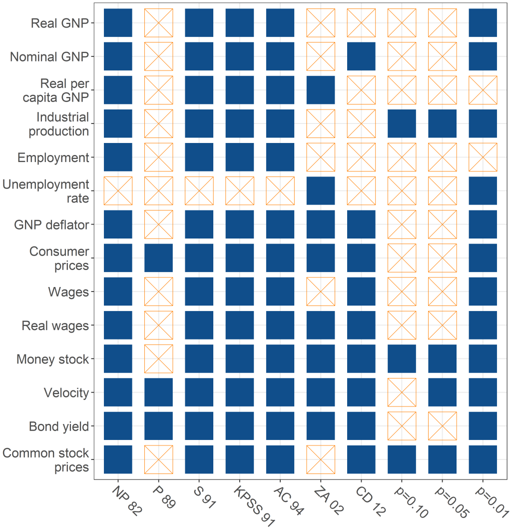
	method	pvalue	threshold	sensitivity	specificity	tp	tn	fp	fn	accuracy
1	xgbTree	best	0.51944008	0.9596049	0.9261502	35847	34864	2780	1509	0.9428133
2	xgbTree	0.1	0.66509449	0.9765232	0.9000106	36479	33880	3764	877	0.9381200
21	xgbTree	0.05	0.33739650	0.9249652	0.9500053	34553	35762	1882	2803	0.9375333
22	xgbTree	0.01	0.04205808	0.6550755	0.9900117	24471	37268	376	12885	0.8231867

```
> out01results
```

	series_no	pvalue_requested	pvalue_returned	type_of_result	score_xgbTree	threshold_xgbTree
1	gnp.r	0.01	0.01	using exact p-value threshold	0.056929197	0.04205808
2	gnp.n	0.01	0.01	using exact p-value threshold	0.126725985	0.04205808
3	gnp.pc	0.01	0.01	using exact p-value threshold	0.005538497	0.04205808
4	ip	0.01	0.01	using exact p-value threshold	0.931120299	0.04205808
5	emp	0.01	0.01	using exact p-value threshold	0.000899303	0.04205808
6	ur	0.01	0.01	using exact p-value threshold	0.216778159	0.04205808
7	gnp.p	0.01	0.01	using exact p-value threshold	0.318520453	0.04205808
8	cpi	0.01	0.01	using exact p-value threshold	0.077048123	0.04205808
9	wg.n	0.01	0.01	using exact p-value threshold	0.171301544	0.04205808
10	wg.r	0.01	0.01	using exact p-value threshold	0.115318656	0.04205808
11	M	0.01	0.01	using exact p-value threshold	0.892714120	0.04205808
12	vel	0.01	0.01	using exact p-value threshold	0.896362881	0.04205808
13	bnd	0.01	0.01	using exact p-value threshold	0.246558752	0.04205808
14	sp	0.01	0.01	using exact p-value threshold	0.837645248	0.04205808

Comparison with previous literature...





Conclusion

- Unit Root tests are weak learners.
- We can aggregate weak learners using gradient boosting to form a pseudo-composite test for unit roots.
- This is [pessimistically] 20 percentage points more accurate and 37 percentage points more powerful than a traditional unit root test.

Thank you!

Main Results Table

Table: Main Results

	ACC	SEN	SPE	PPV	NPV	F^1	MCC
$\mathrm{GB}\vert\alpha = 0.100$	0.937	0.951	0.923	0.925	0.950	0.938	0.874
$\mathrm{GB}\vert\alpha = 0.074^*$	0.941	0.934	0.948	0.947	0.935	0.941	0.882
$\mathrm{GB}\vert\alpha = 0.050$	0.938	0.900	0.976	0.973	0.908	0.935	0.878
$\mathrm{GB}\vert\alpha = 0.010$	0.892	0.785	0.998	0.998	0.823	0.879	0.802
ADF	0.763	0.546	0.980	0.964	0.684	0.697	0.583
PP	0.744	0.512	0.975	0.953	0.667	0.666	0.549
KPSS	0.614	0.250	0.977	0.916	0.567	0.393	0.331
PGFF	0.745	0.499	0.989	0.978	0.665	0.661	0.560
BREIT	0.672	0.361	0.981	0.951	0.607	0.524	0.437
ERSd	0.762	0.545	0.979	0.963	0.683	0.696	0.582
ERSp	0.770	0.564	0.976	0.958	0.692	0.710	0.592
URZA	0.635	0.309	0.959	0.883	0.582	0.458	0.354
URSP	0.727	0.552	0.903	0.850	0.669	0.669	0.485