UPDATES

July 18, 2024

Notes

- new TGUS equation in place
- new data in place - 391 usable tests

parameter	c	s	s	r	h	a	c	t	s	s	r	s	c	tgus
Metolachlor OA	6...	3...	3...	A...	17	1...	65	0...	2	4...	0...	2...	1	18.5
Metolachlor ESA	7...	4...	3...	A...	9	13	65	0...	2	4...	0...	2...	1	17.6
Metolachlor ESA	7...	4...	3...	A...	9	13	65	0...	2	4...	0...	2...	1	12.9
Metolachlor OA	6...	3...	3...	A...	17	1...	65	0...	2	4...	0...	2...	1	20.9
Oxadiazon	1...	5...	3...	A...	3...	1...	4...	1...	0	0...	0...	2...	1	2.1
Metolachlor OA	6...	3...	3...	A...	17	1...	4...	0...	2	4...	0...	2...	1	22.5
Metolachlor OA	6...	3...	3...	A...	17	1...	65	0...	2	4...	0...	2...	1	18.5
Metolachlor OA	6...	3...	3...	A...	17	1...	4...	0...	2	4...	0...	2...	1	20
Metolachlor ESA	7...	4...	3...	A...	9	13	4...	0...	2	4...	0...	2...	1	19.6
Metolachlor ESA	7...	4...	3...	A...	9	13	4...	0...	2	4...	0...	2...	1	14.8
Metolachlor ESA	7...	4...	3...	A...	9	13	65	0...	2	4...	0...	2...	1	12.9
Carbaryl	2...	16	2...	E...	3...	1...	2...	1...	1	2...	0...	2...	1	0.5
Atrazine	2...	75	2...	C..	1...	3...	1...	1...	2	3...	0...	2...	1	6.6
Carbaryl	2...	16	2...	E...	3...	1...	75	0...	1	2...	0...	2...	1	0.3
Metolachlor ESA	7...	4...	3...	A...	9	13	4...	0...	2	4...	0...	2...	1	19.6
Simazine	2...	60	2...	C..	1...	1...	2...	1...	2	3...	0...	2...	1	4.4

```python
def tgus(shl, app, det, om, bulk, koc):
    '''
    calculates tgus equation from soil and pesticide parameters
    ---
    returns: tgus, result from equation
    ---
    args: shl, soil halflife for pesticide [m3/Mg]
          app, pesticide application rate [mg/m2]
          det, pesticide detection limit [mg/m3]
          om, soil organic matter fraction
          bulk, soil bulk density [Mg/m3]
          koc, partitioning coefficient for pesticide
    '''
    t = 100 # current assumption, [days]
    pf = 0.004 # current assumption
    phi = app / (det * 0.01)
    foc = 0.6 * om / 100
    xi = (pf * phi) / (foc * bulk)
    tgus = round(0.025 * shl * (np.log10(xi/koc)) - 0.0075*t,1)

    return tgus
```

New dataset, more complex TGUS equation

Results - binary classification accuracy (detected or nondetected)

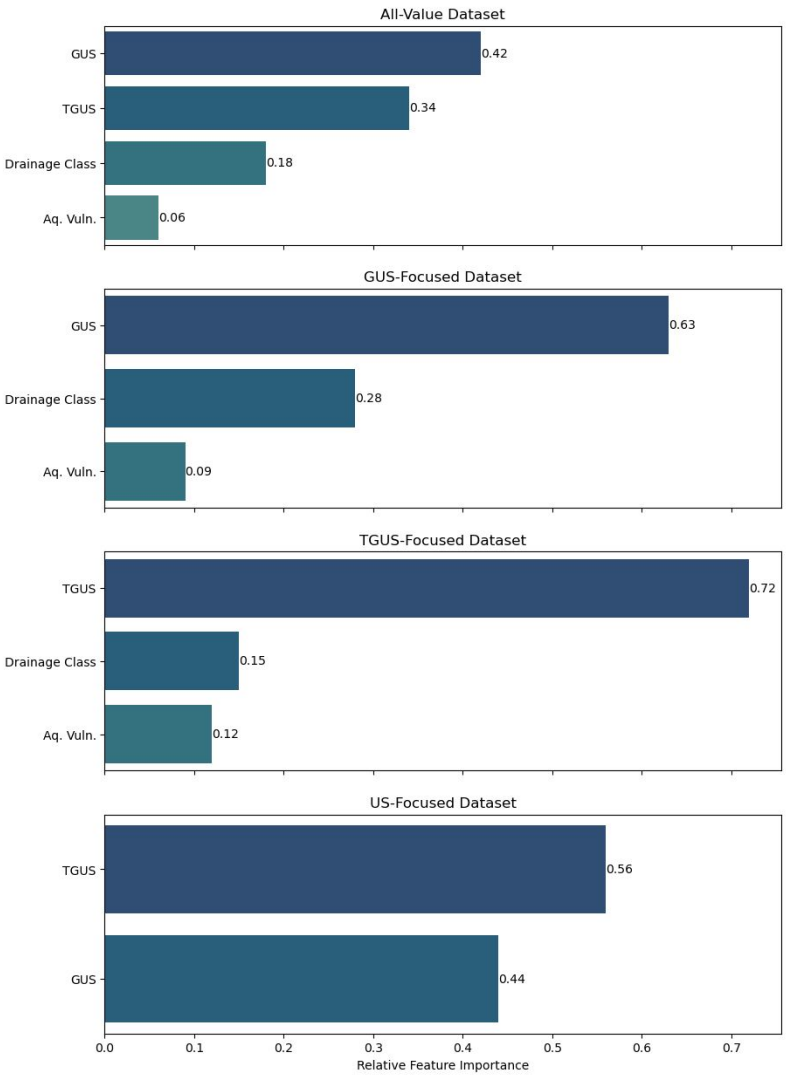
- **All**: GUS, TGUS, drainage class, aquifer vulnerability
- **GUS Focus**: GUS, drainage class, aquifer vulnerability
- **TGUS Focus**: TGUS, drainage class, aquifer vulnerability
- **US Focus**: GUS, TGUS

	Avg. Train %	Avg. Validation %	Avg. Test %	Best Sc. Train %	Best Sc. Validation %	Best Sc. Test %
All	97.3	95.7	94.2	96.2	97.4	98.7
GUS Focus	94.5	95	92.4	93.6	93.6	97.5
TGUS Focus	97.3	95.4	92.8	96.6	94.9	97.5
US Focus	97.1	96.5	93.2	97.4	93.6	98.7

- **All** is statistically best performer (t-tests), otherwise no differences

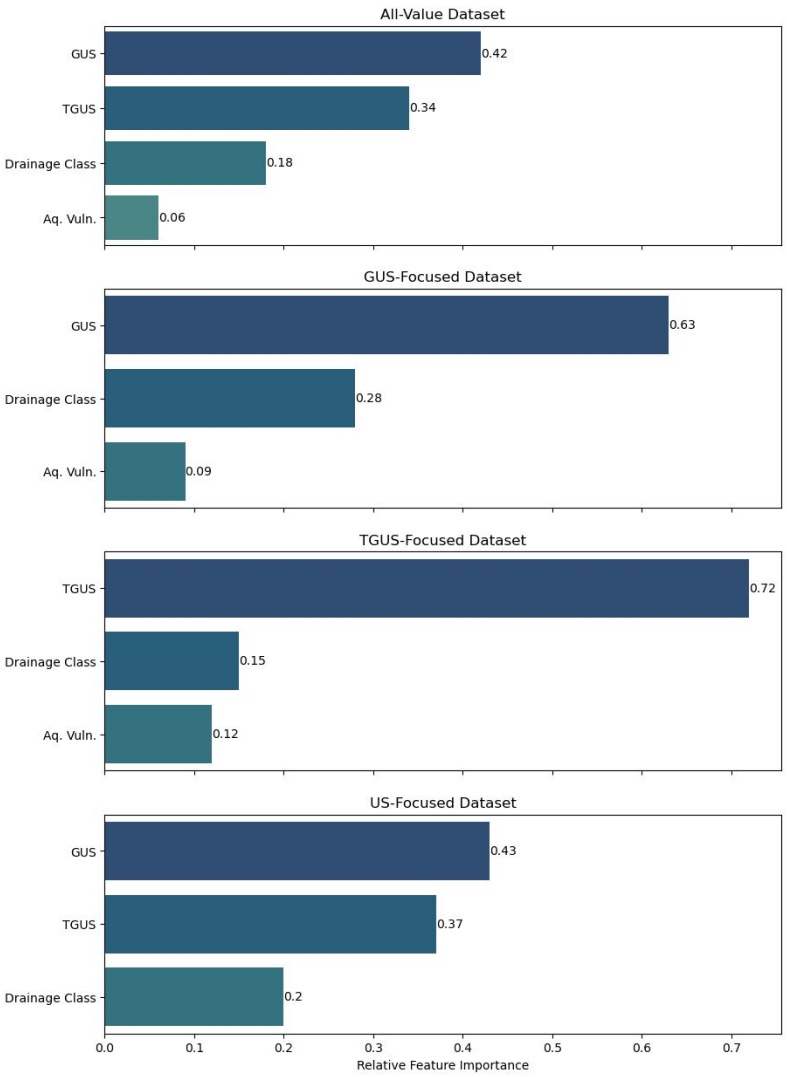
Feature Importances (FI)

- averaged over 50 iterations -> consistent pattern
- GUS vs TGUS
  - TGUS has more FI alone
  - TGUS has more FI with just GUS
  - TGUS has less FI with all values…confusing
- if we look further…



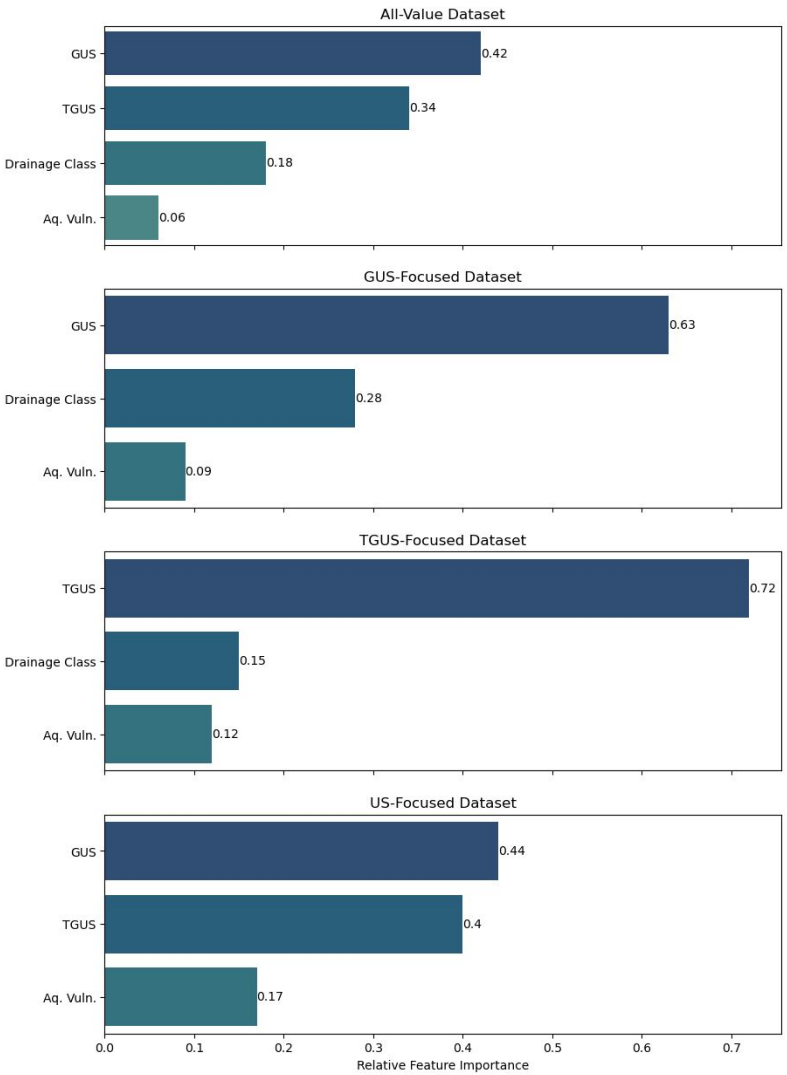
Add in Drainage Class

- TGUS shares predictive info with drainage class
- testing accuracy roughly same: 94.2%
    - need to investigate further

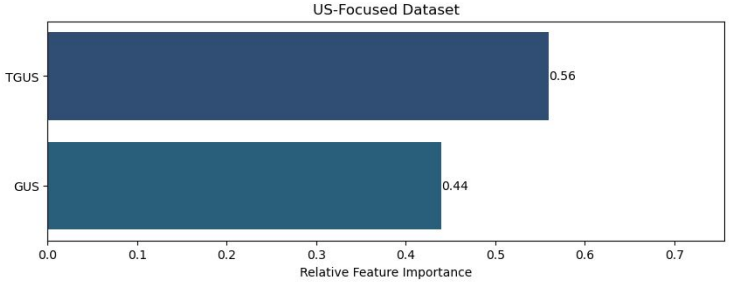


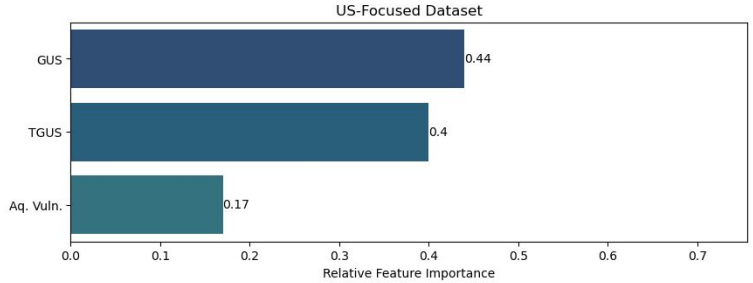
Add in Aquifer Vulnerability

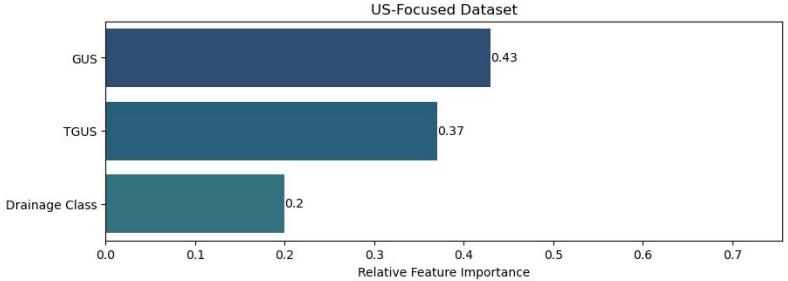
- TGUS shares predictive info with aquifer vulnerability
- testing accuracy roughly same: 93.6%
    - need to investigate further

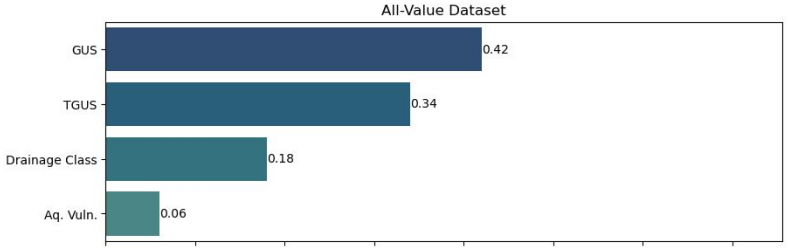


What this tells us









- GUS remains unchanged, ~0.42-0.44
- **TGUS capturing more of drainage class and aquifer vulnerability**
  - …how much??

TGUS and GUS alone

Results - binary classification accuracy (detected or nondetected)

	Avg. Train %	Avg. Validation %	Avg. Test %	Best Sc. Train %	Best Sc. Validation %	Best Sc. Test %
GUS Alone	90.7	92.8	89.5	90.2	89.7	93.7
TGUS Alone	97	95.8	93.1	95.7	96.2	98.7

- TGUS performs just as well alone *without* drainage class or aquifer vulnerability
- GUS does not…

Moving Forward

- better understand what TGUS is not capturing
- find the best model is
- verify no errors in results