Exploit Prediction Scoring System (EPSS)

Jay Jacobs @jayjacobs
Michael Roytman, @mroytman

Jay - @jayjacobs

- Chief Data Scientist, Cyentia Institute
- Co-Author, “Data-Driven Security"
- Previously: Lead Analyst, Verizon DBIR

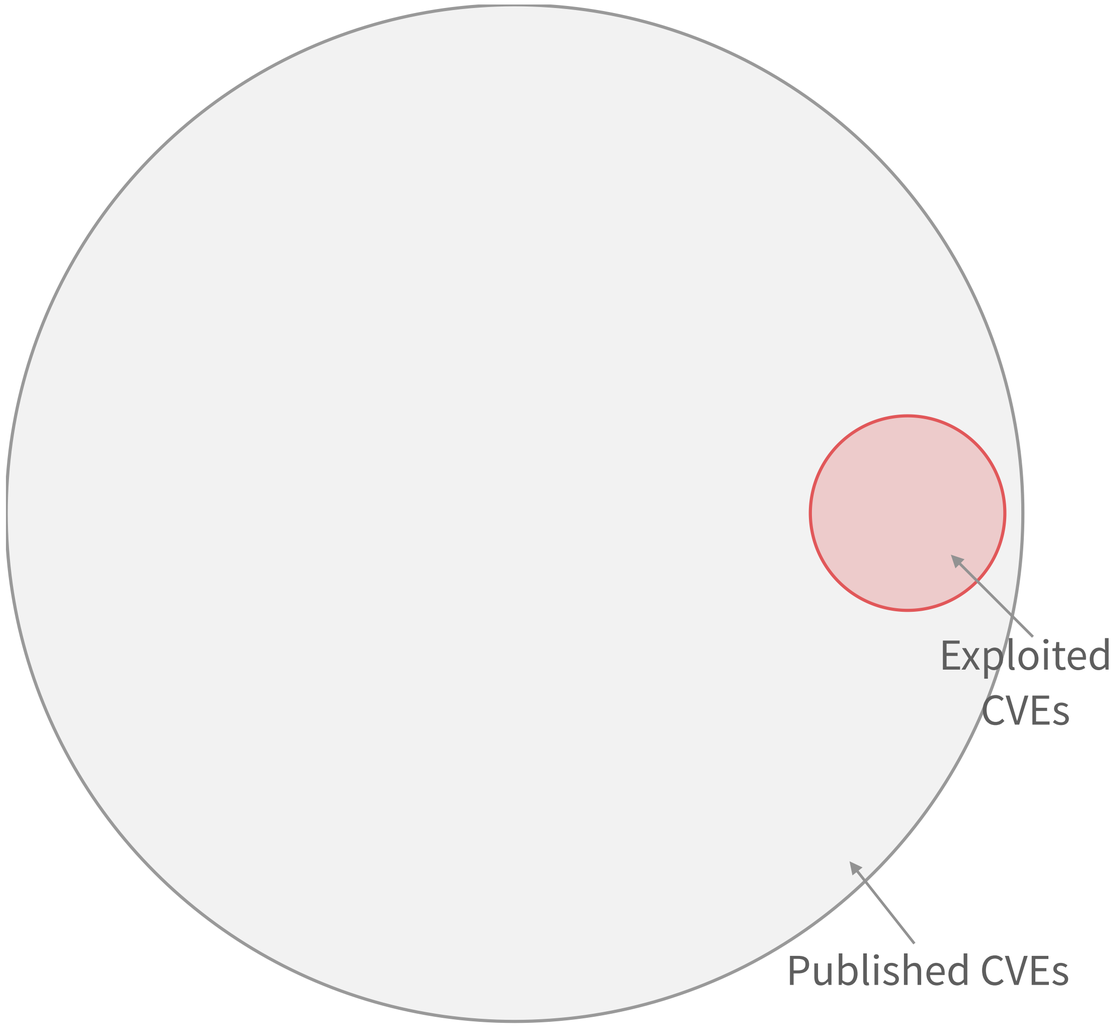
Michael - @mroytman

- Chief Data Scientist, Kenna Security
- Founder Dharma Platform (Forbes 30 under 30)

What are we talking about?

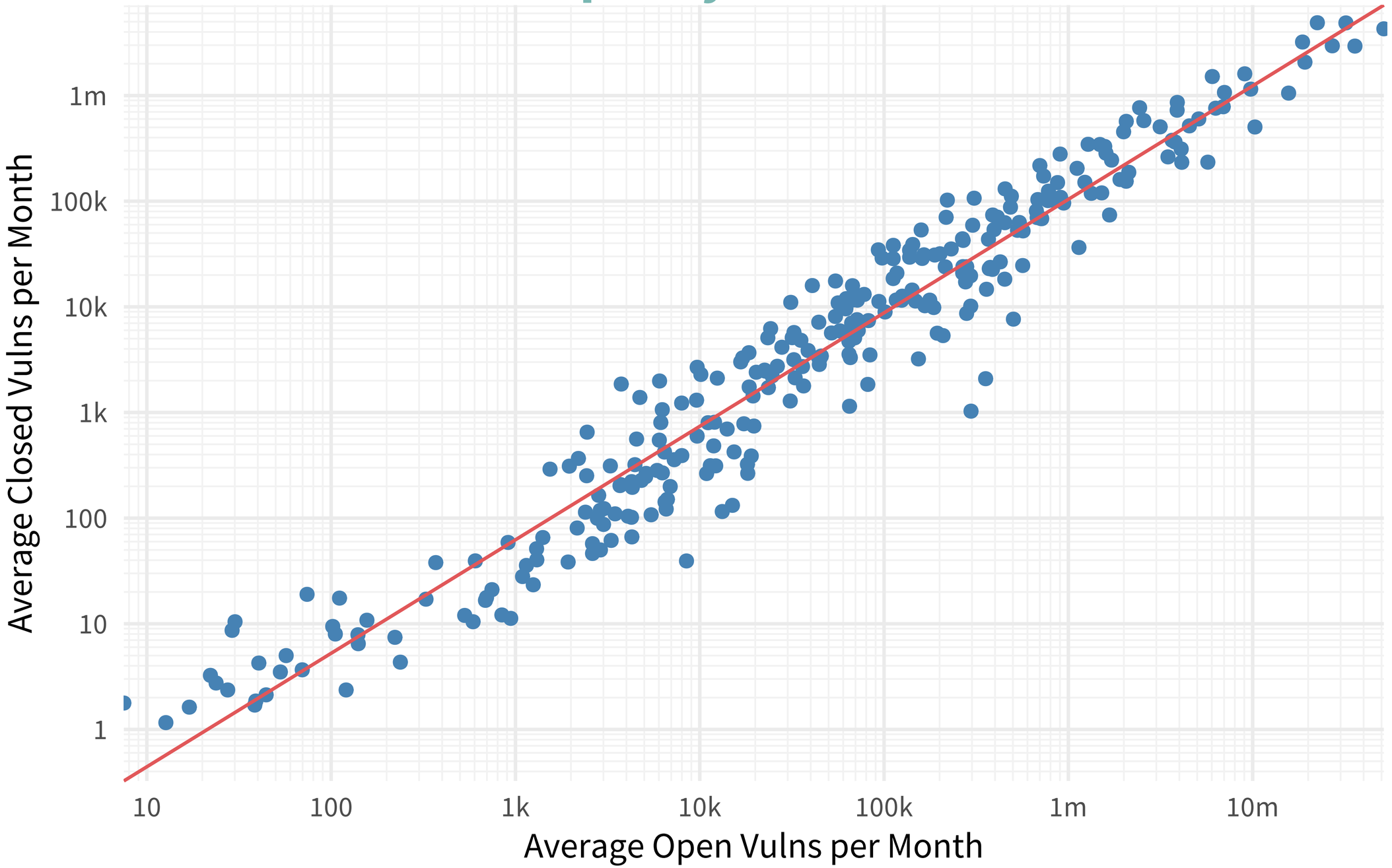
1. The Wicked Problem: Vulnerability Management
2. Measuring Performance
3. A (closed) predictive model
4. An Open Model: Probabilistic Vulnerability Scoring System (PVSS)

Wicked Problem: Vulnerability Management

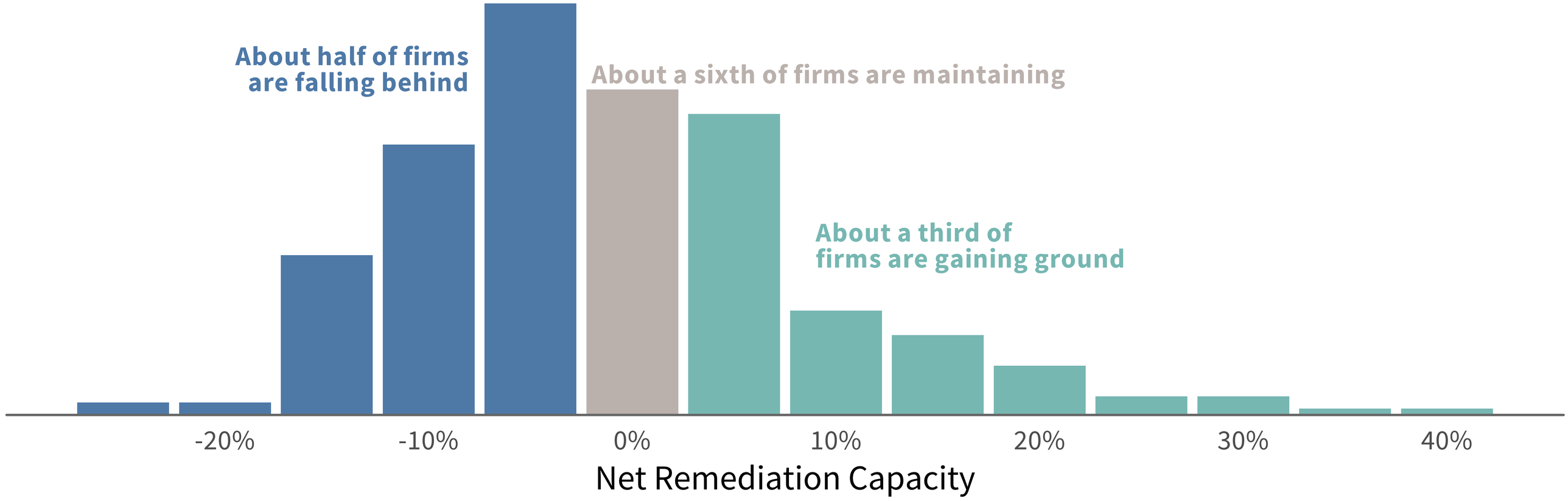


- Too many vulnerabilities to patch them all
- Most vulnerabilities are never used by attackers
- We need to a system to fix what matters

What is the capacity for remediation?



Is remediation keeping up with new vulnerabilities?

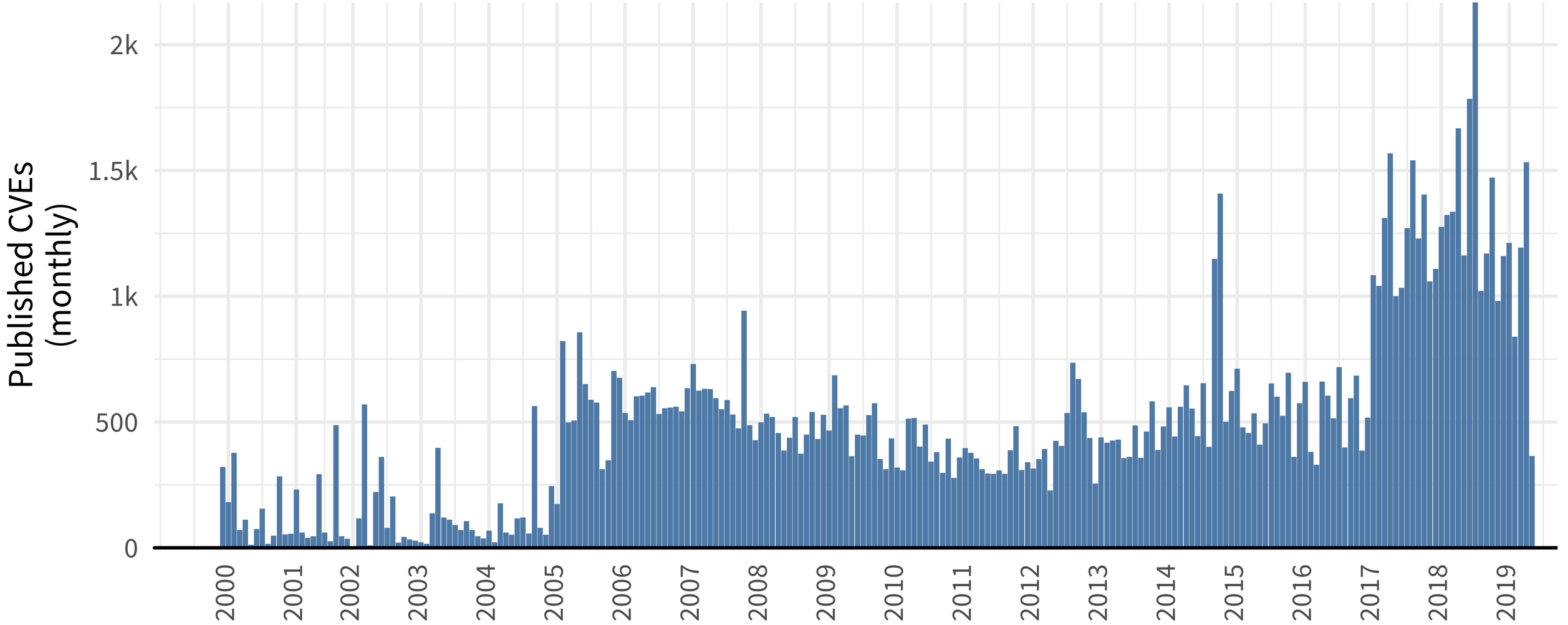


Source: Kenna / Cyentia

A Problem everyone recognizes

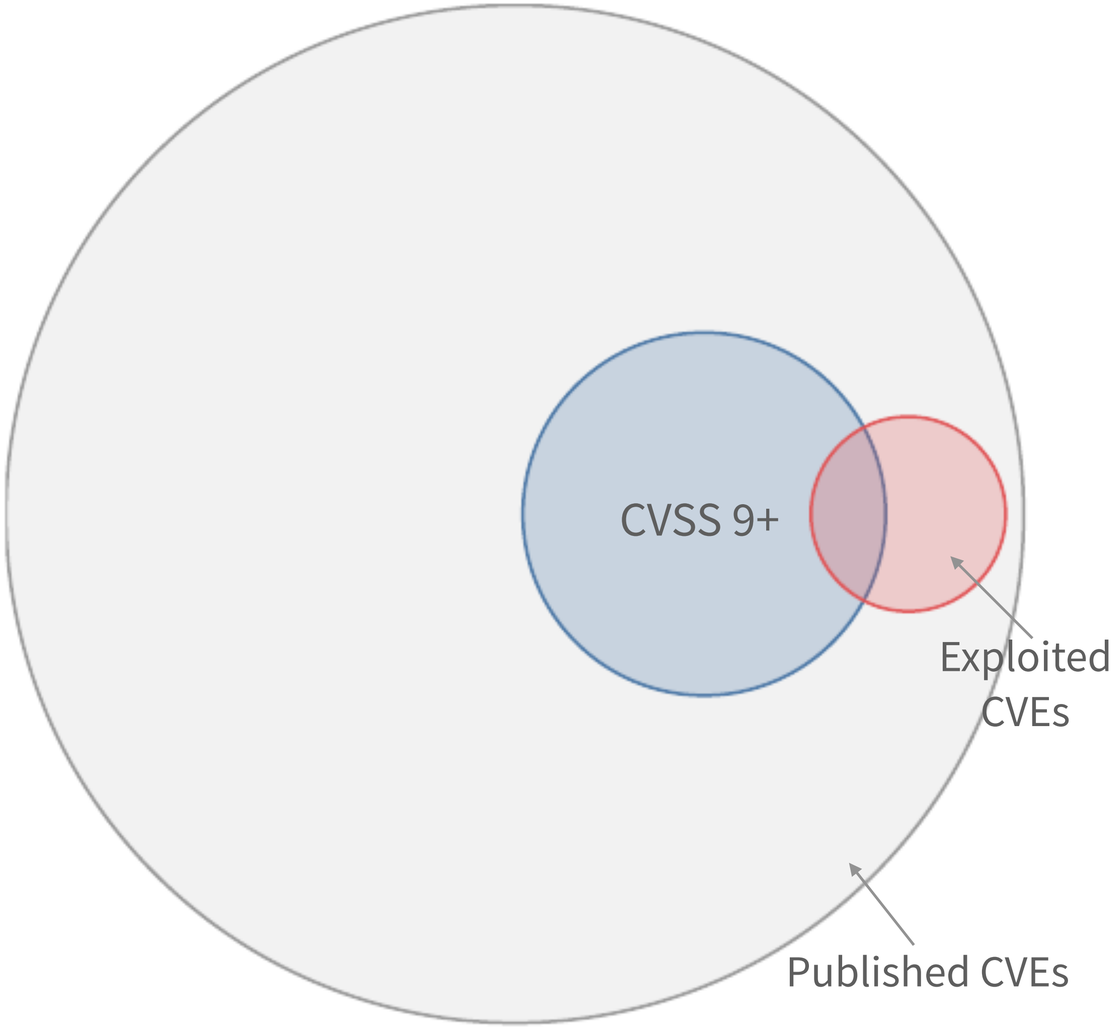
- US CyberCom challenge #2:
  - “It takes time to assess, test, and deploy patches that fix newly identified vulnerabilities. This is as much a resource management issue as it is a technical issue. We lack a rapid patch methodology based on newly released Common Vulnerability Exposures (CVEs).”
- Critiquing a system isn’t enough, must provide an alternative system to test against
- More vulnerabilities every year, more of them critical, same amount of resources/people:

CVEs Over Time



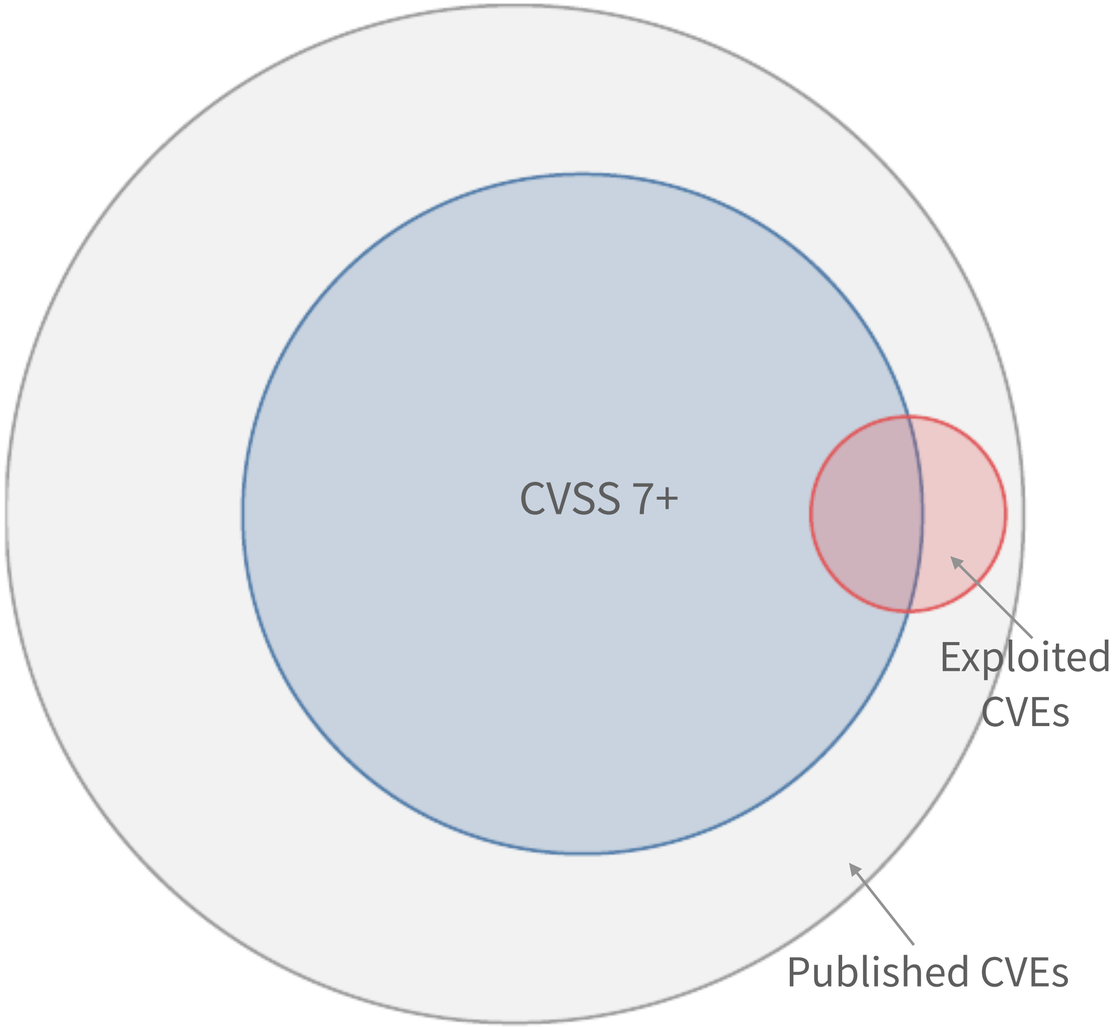
A Wicked Problem…

- Somewhere between 2-5% of CVE's are (detected as) exploited in the wild
- How do we prioritize these?
- How does CVSS help?



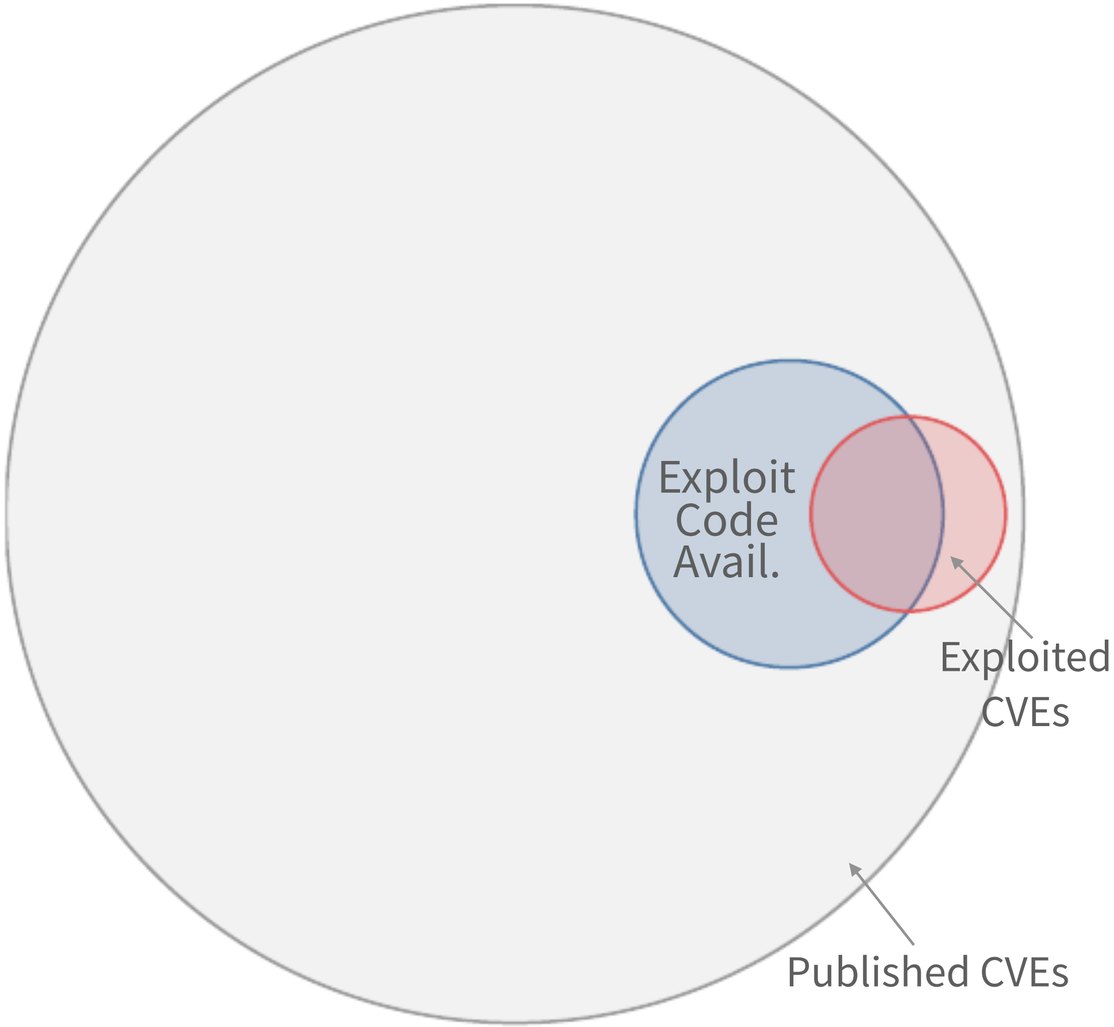
A Wicked Problem…

- Somewhere between 2-5% of CVE's are (detected as) exploited in the wild
- How do we prioritize these?
- How does CVSS help?



A Wicked Problem…

- Somewhere between 2-5% of CVE's are (detected as) exploited in the wild
- How do we prioritize these?
- How does CVSS help?
- What about tracking published exploits?



UNIVERSITY OF TRENTO

SECONOMICS

TENACE

How CVSS is DOSsing your patching policy (and wasting your money)

Luca Allodi, Fabio Massacci
University of Trento, Italy.

black hat USA 2013

JULY 27 - AUGUST 1, 2013
CAESARS PALACE | LAS VEGAS, NV
WWW.BLACKHAT.COM

Conclusion: answer to the CIO

SECONOMICS

black hat USA 2013

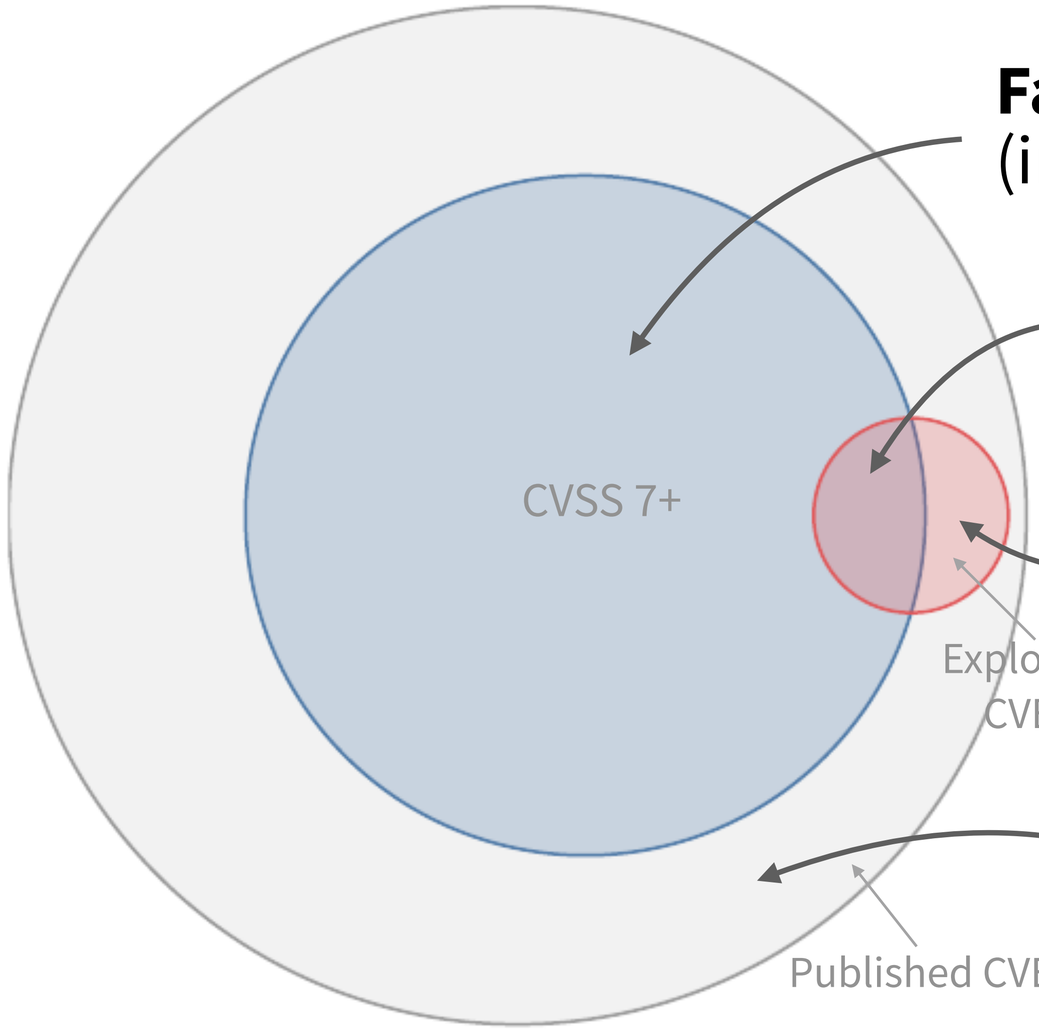
- Is wearing a seat belt any useful?
  - Pr(Death x Safety Belt on) – Pr(Death x Safety Belt off)
  - Yes it is→ 43% improvement of chances of survival
    - L. Evans, Accident Analysis and Prevention 1986
- Is patching HIGH score any useful?
  - Pr(Attack x CVSS High) – Pr(Attack x CVSS Low)
- Finally the figures the CIO wants
  - Patching HIGH/MED and exploit sold in Exploit Kits
    →improves by +62.81% (Buckle up!)
  - Patching fix HIGH/MED and PoC exploit by white hats
    → improves by +19.64% (Up to you)
  - Patching just HIGH/MED
    → improves by +3.2% (Life is too short)

Thanks

black hat USA 2013

Measuring Your Remediation Strategy

Measuring performance



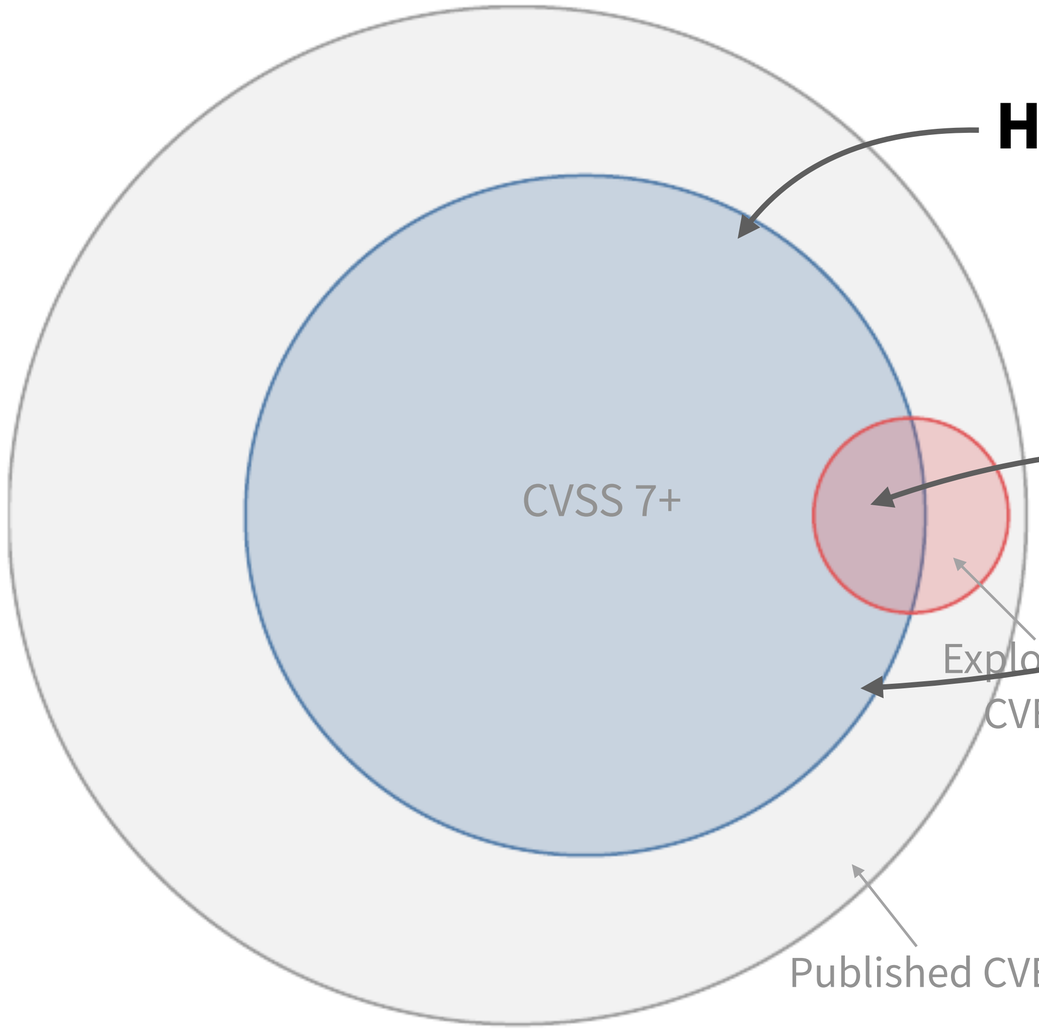
False Positives
(incorrectly prioritized)

True Positives
(correctly prioritized)

False Negatives
(incorrectly delayed)

True Negatives
(correctly delayed)

Measuring performance: Efficiency (precision)

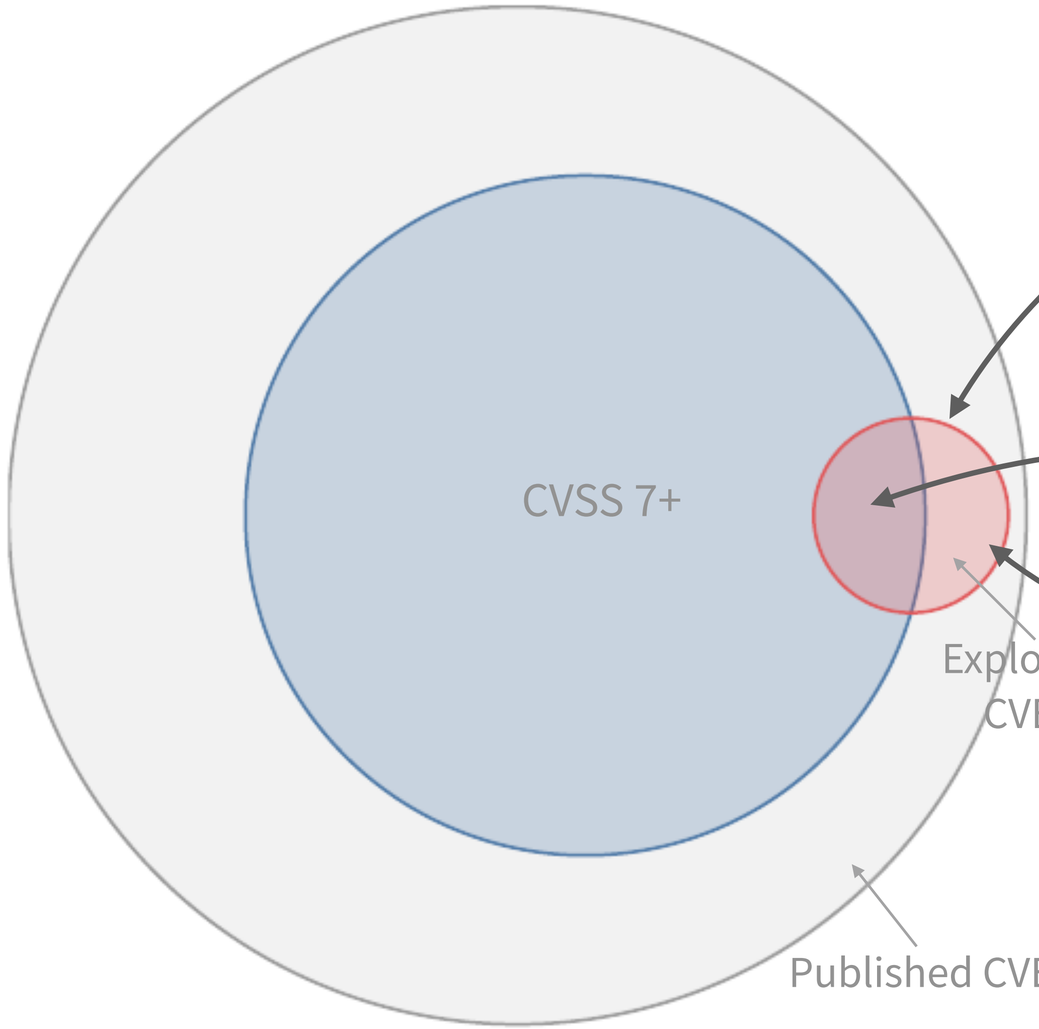


How <u>efficient</u> is the remediation effort?

$$\frac{\text{\# of True Positives}}{\text{\# Prioritized (FP+TP)}}$$

<u>Efficiency</u> for CVSS 7+ is about 5-7%

Measuring performance: Coverage (recall)



How much <u>coverage</u> from the remediation effort?

$$\frac{\text{\# of True Positives}}{\text{\# Should be Prioritized (FN +TP)}}$$

<u>Coverage</u> for CVSS 7+ is about 45-65%

Improving Vulnerability Remediation Through Better Exploit Prediction

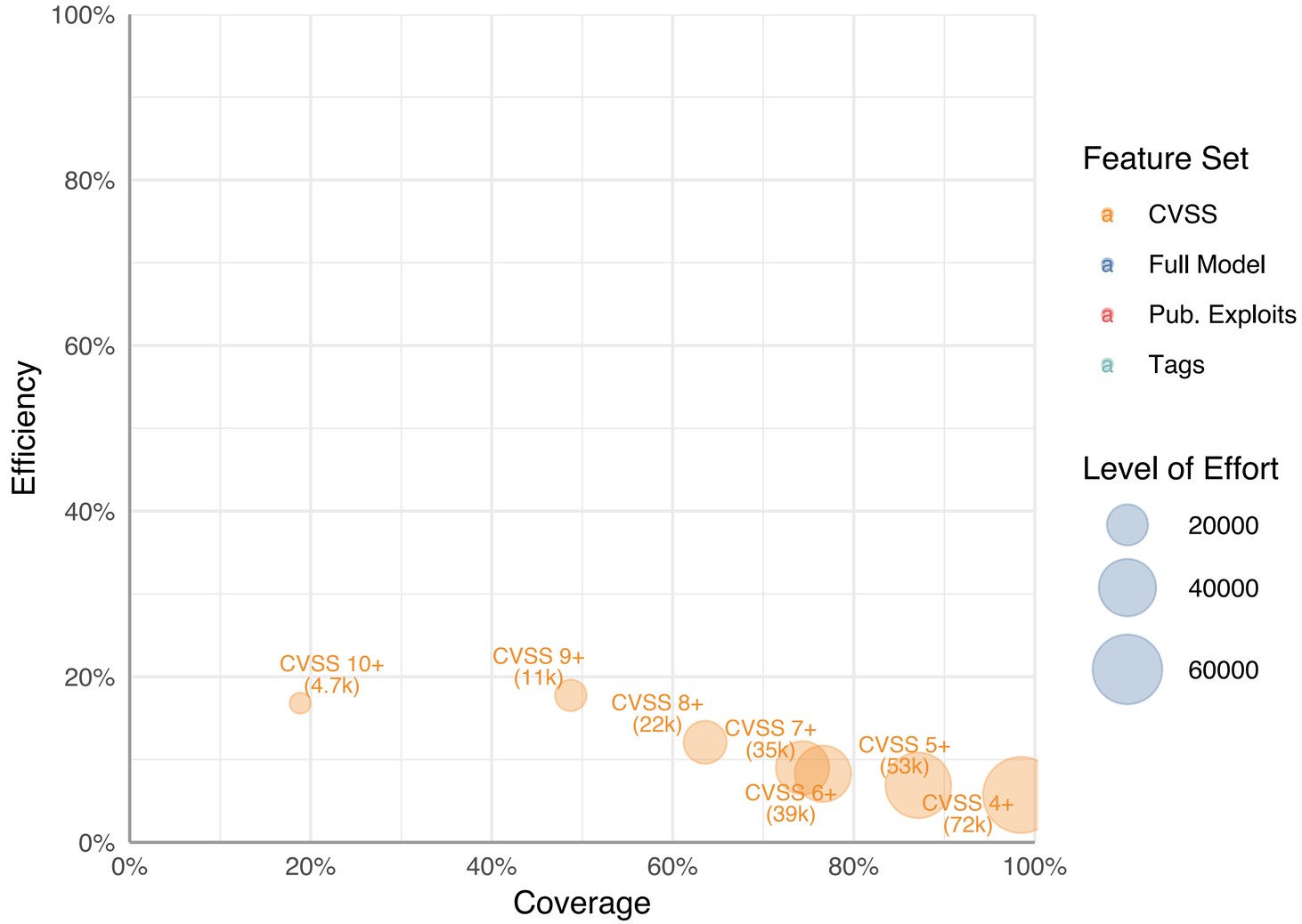
Jay Jacobs, jay@cyentia.com, Cyentia
Sasha Romanosky, sromanos@rand.org, RAND Corporation
Idris Adjerid, iadjerid@vt.edu, Virginia Tech
Wade Baker, wbaker@vt.edu, Virginia Tech

Despite significant innovations in IT security products and research over the past 20 years, the information security field is still immature and struggling. Practitioners lack the ability to properly assess cyber risk, and decision-makers continue to be paralyzed by vulnerability scanners that overload their staff with mountains of scan results. In order to cope, firms prioritize vulnerability remediation using crude heuristics and limited data, though they are still too often breached by known vulnerabilities for which patches have existed for months or years. And so, the key challenge firms face is trying to identify a remediation strategy that best balances two competing forces. On one hand, it could attempt to patch all vulnerabilities on its network. While this would provide the greatest *coverage* of vulnerabilities patched, it would inefficiently consume resources by fixing low-risk vulnerabilities. On the other hand, patching a few high-risk vulnerabilities would be highly *efficient*, but may leave the firm exposed to many other high-risk vulnerabilities. Using a large collection of multiple datasets together with machine learning techniques, we construct a series of vulnerability remediation strategies and compare how each perform in regard to trading off coverage and efficiency. We expand and improve upon the small body of literature that uses predictions of *published exploits*, by instead using *exploits in the wild* as our outcome variable. We implement the machine learning models by classifying vulnerabilities according to high- and low-risk, where we consider high risk vulnerabilities to be those that have been exploited in actual firm networks.

Keywords: vulnerability management, exploited vulnerability, CVSS, security risk management, machine learning, precision, recall

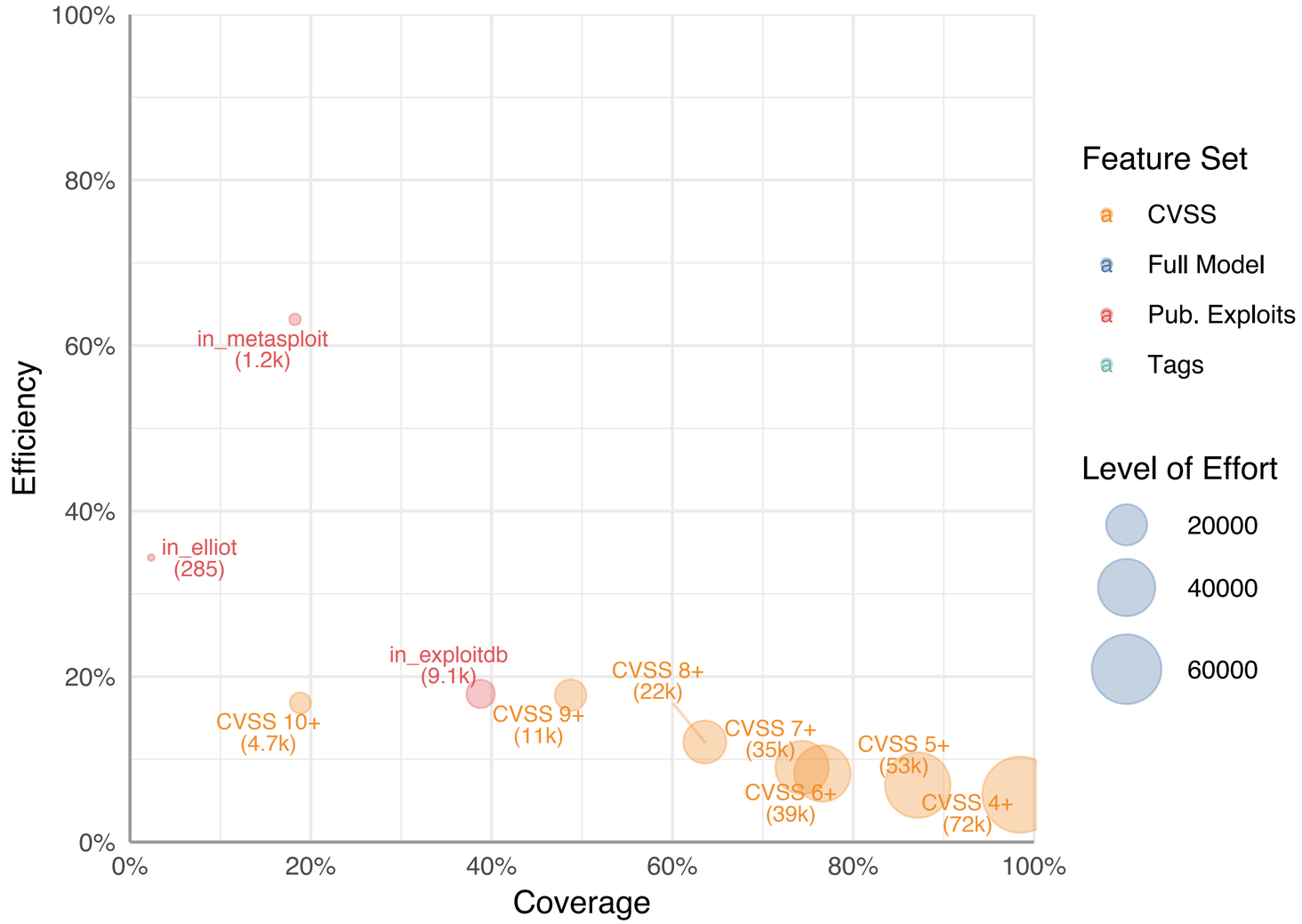
Improving Vulnerability Remediation Through Better Exploit Prediction

- Predicting the binary outcome of exploitation in the wild
- Observed 5.5% exploitation rate
- Compared relatively simple strategies to a predictive model
- eXtreme Gradient Boosted Tree



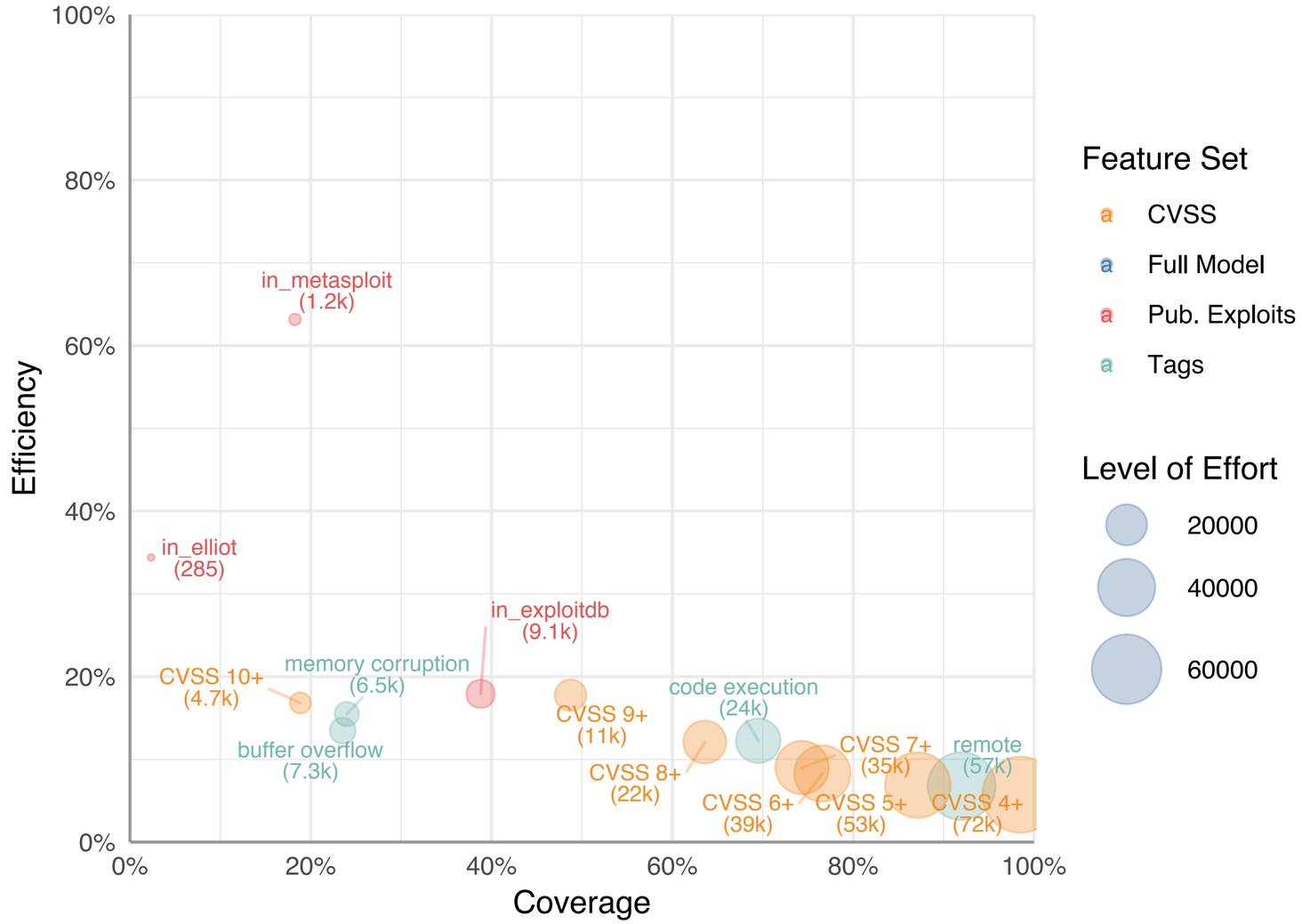
Improving Vulnerability Remediation Through Better Exploit Prediction

- Predicting the binary outcome of exploitation in the wild
- Observed 5.5% exploitation rate
- Compared relatively simple strategies to a predictive model
- eXtreme Gradient Boosted Tree



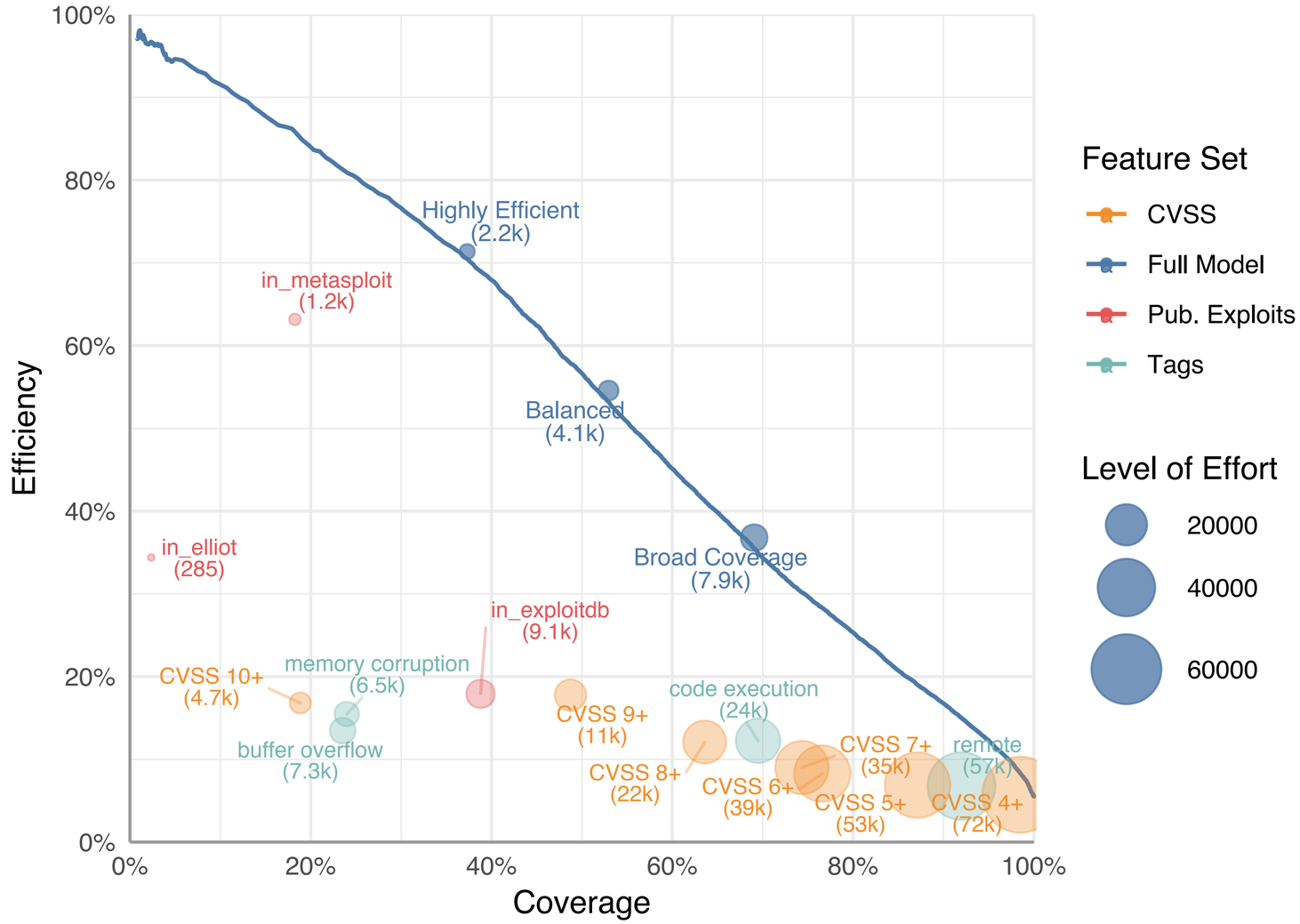
Improving Vulnerability Remediation Through Better Exploit Prediction

- Predicting the binary outcome of exploitation in the wild
- Observed 5.5% exploitation rate
- Compared relatively simple strategies to a predictive model
- eXtreme Gradient Boosted Tree



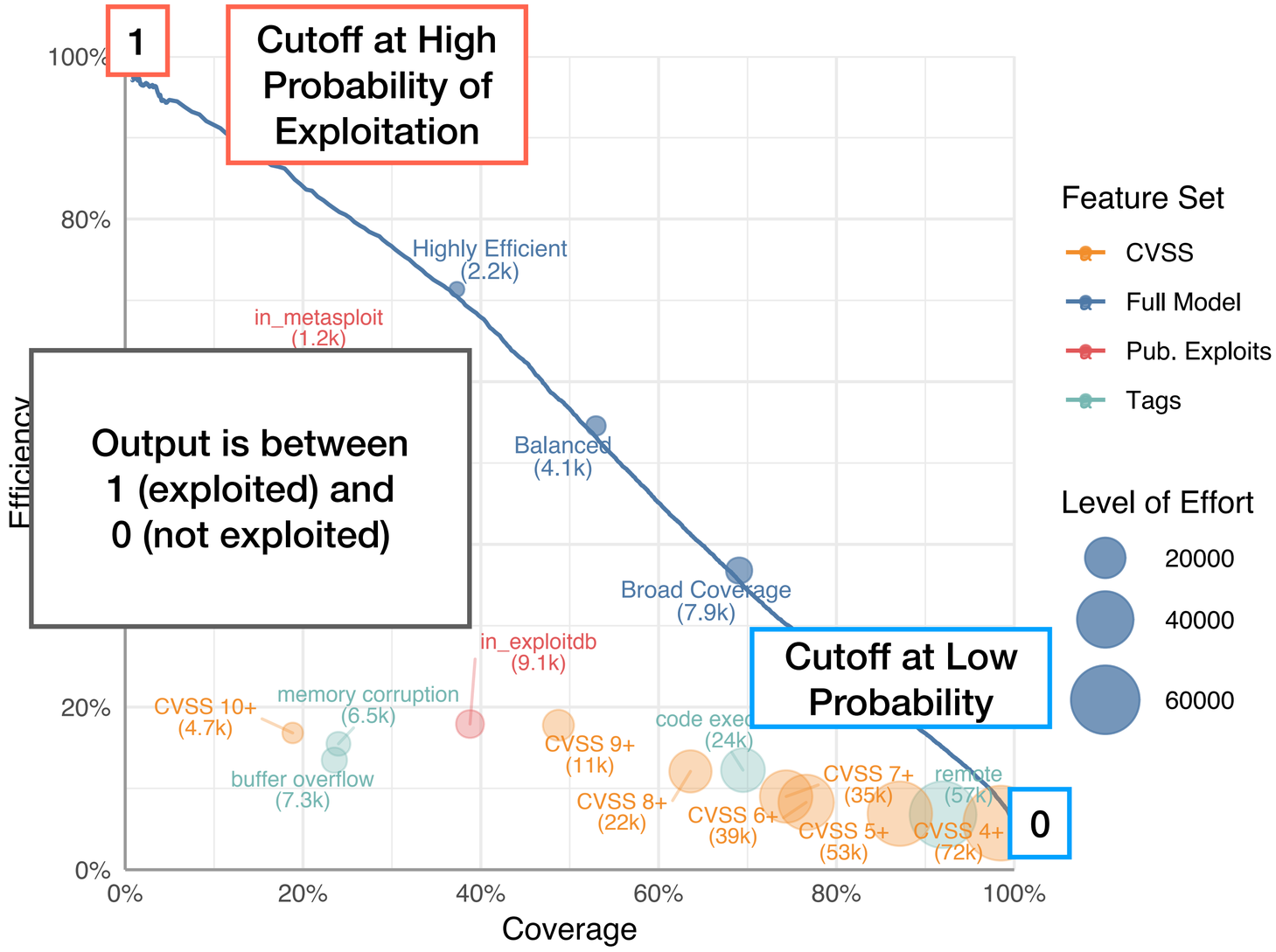
Improving Vulnerability Remediation Through Better Exploit Prediction

- Future predicting past (randomized cross-validation)
- 10 years of CVEs
- 3-5 Years of Exploitations
- Closed Model



Improving Vulnerability Remediation Through Better Exploit Prediction

- Future predicting past (randomized cross-validation)
- 10 years of CVEs
- 3-5 Years of Exploitations
- Closed Model



An Open Prioritization Model

Goals:

- Simple to Implement
- Interpretable
- Parsimonious
- Performant

- Targeting spreadsheet implementation
- Logistic Regression
- Limit variables required, public and objective only
- Better than existing

The data behind the model

- CVEs from Mitre
  - CVE ID, description, references
- CVEs from NVD
  - CVSS, CPE (vendor/platform)
- Public Exploits
  - Metasploit, D2's Elliot, Canvas
  - Exploit DB
- Exploited-in-the-wild
  - Fortinet, Proofpoint, AlienVault, Grey Noise

The data behind the model

- CVEs from Mitre
  - CVE ID, description, references
- CVEs from NVD
  - CVSS, CPE (vendor/platform)
- Public Exploits
  - Metasploit, D2's Elliot, Canvas
  - Exploit DB
- Exploited-in-the-wild
  - Fortinet, Proofpoint, AlienVault, Grey Noise

→ From the References:

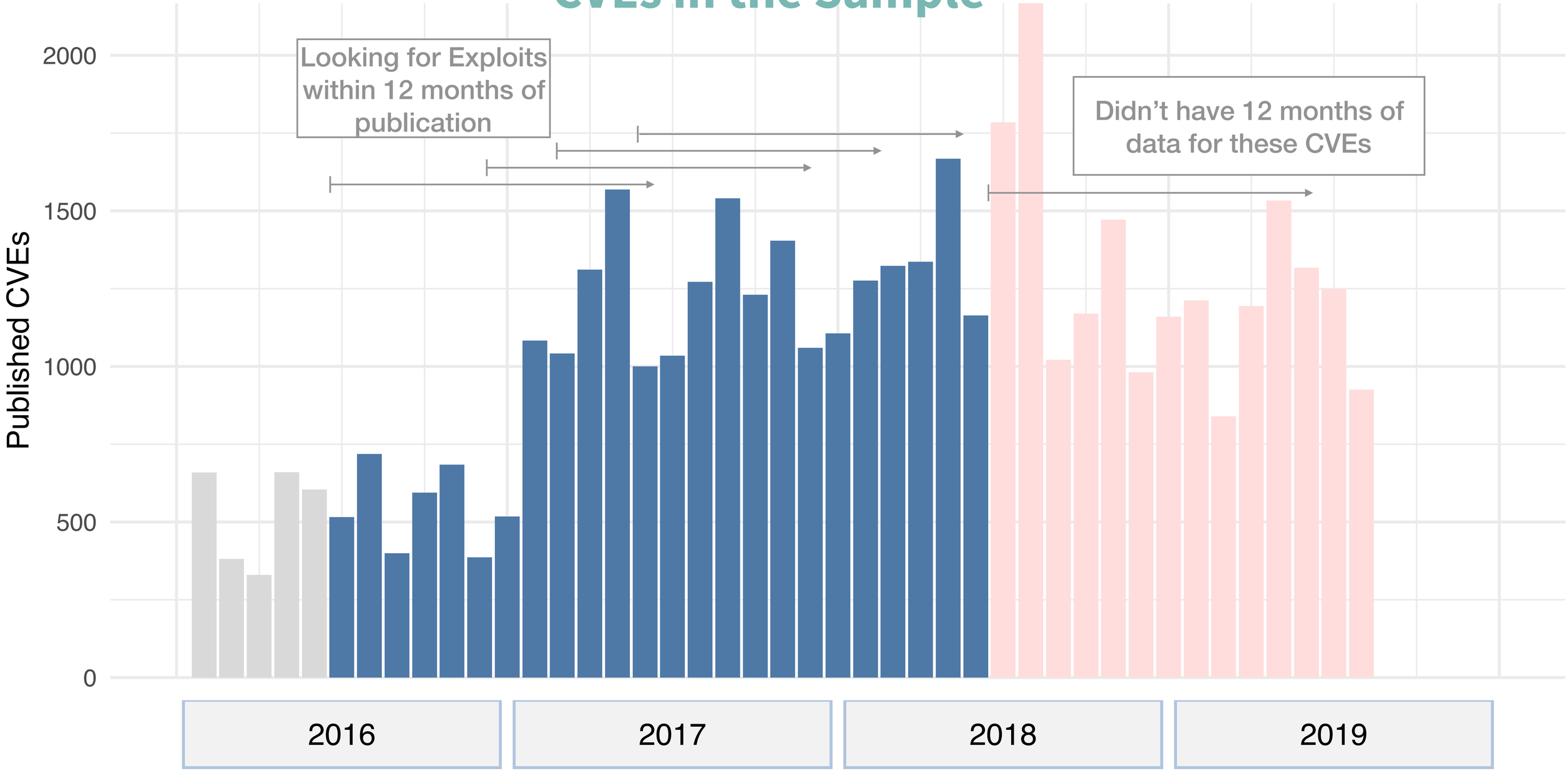
- Scraped the text descriptions
- Extracted multi-word expressions (RAKE)
- Normalized 191 tags

SQLi, SQL injection > "sqli"

Denial of service, ddos, dos > "denial of service"

And so on…

CVEs in the Sample



Data Prep: only impactful variables

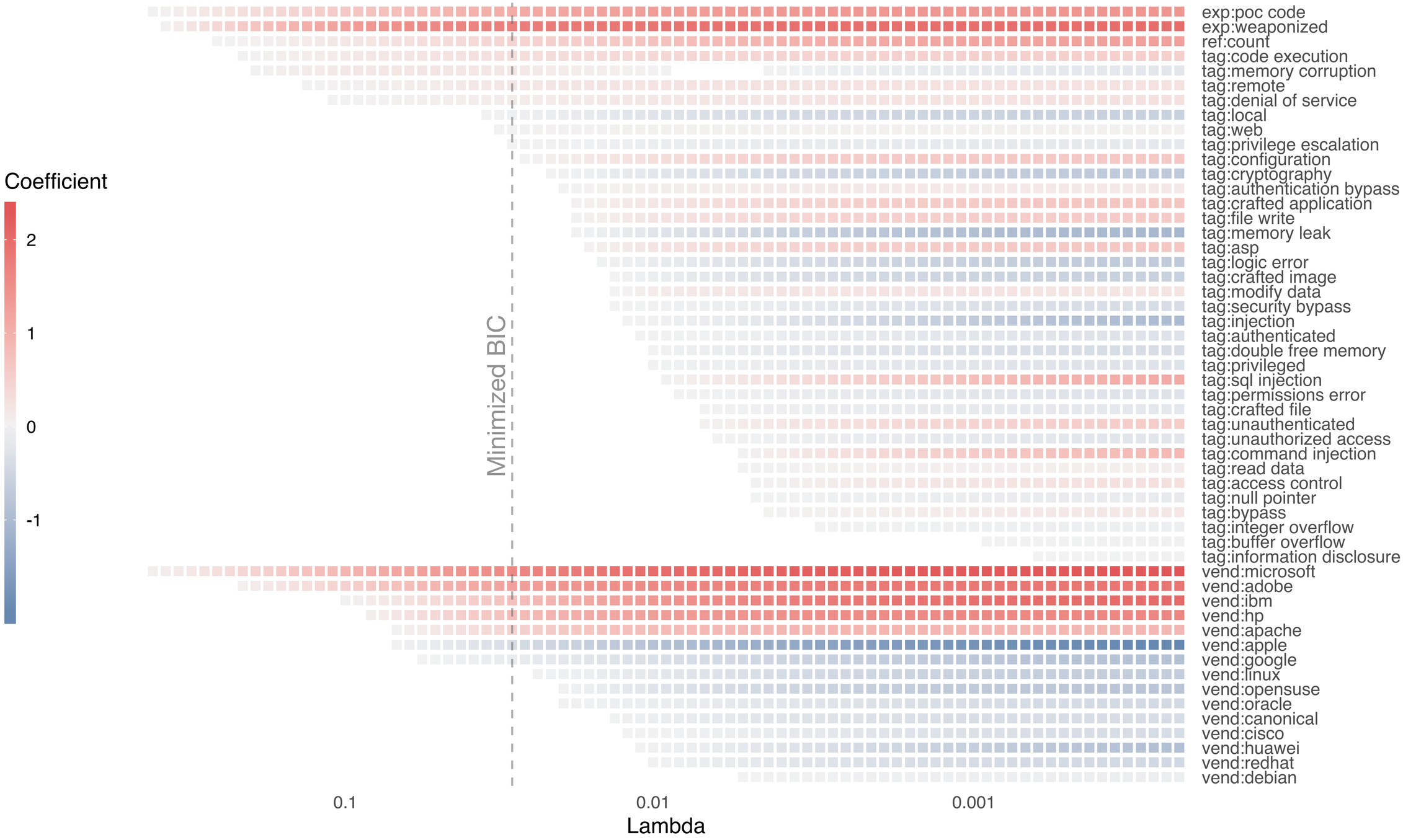
- **Complete Separation**: Remove if variable isn't associated with any/all exploited vulnerabilities
- **Volume**: Must have at least 1% (250) vulnerabilities
- **Expert Review**: where exploits are tough to detect or generic detection (mitm, SQLi, directory traversal, XSS, etc)

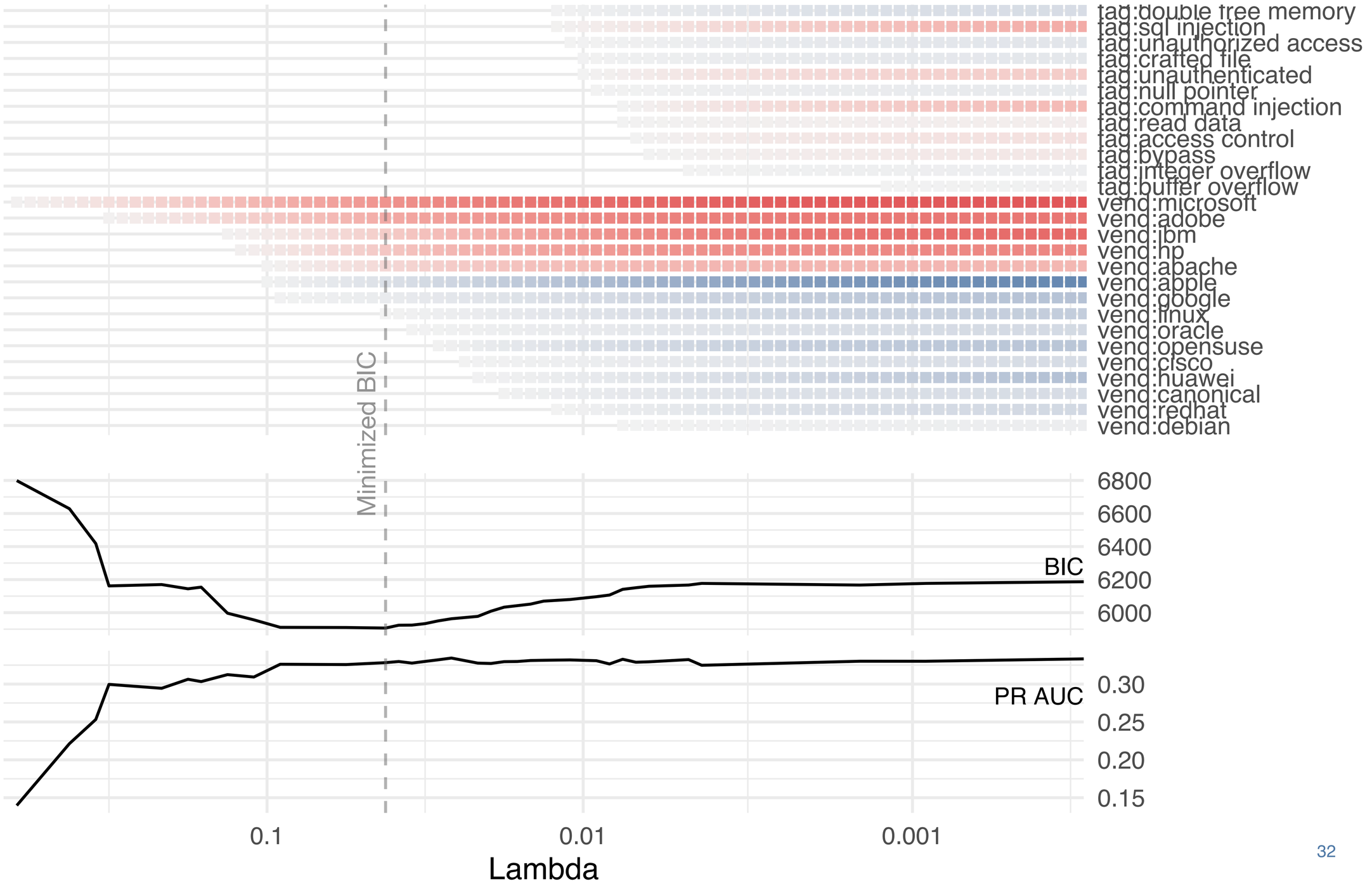
Category of variable	Raw Count	Complete Separation	Volume	Expert Review
Tags	166	111	46	35
Vendors	3374	171	16	15
References	45	28	16	1
Weaponized code	1	1	1	1
POC code	1	1	1	1

ElasticNet

- Applied "Elastic Net” for feature selection
  - A linear combination of L1 and L2 penalty terms on the coefficients
  - Penalizes the size of the coefficients in the model, inducing coefficients to be smaller and less likely to be statistically significant
  - At each linear combination, we tested model performance (BIC, PR AUC)

Zou, Hui, and Trevor Hastie. "Regularization and variable selection via the elastic net." *Journal of the royal statistical society: series B (statistical methodology)* 67, no. 2 (2005): 301-320.

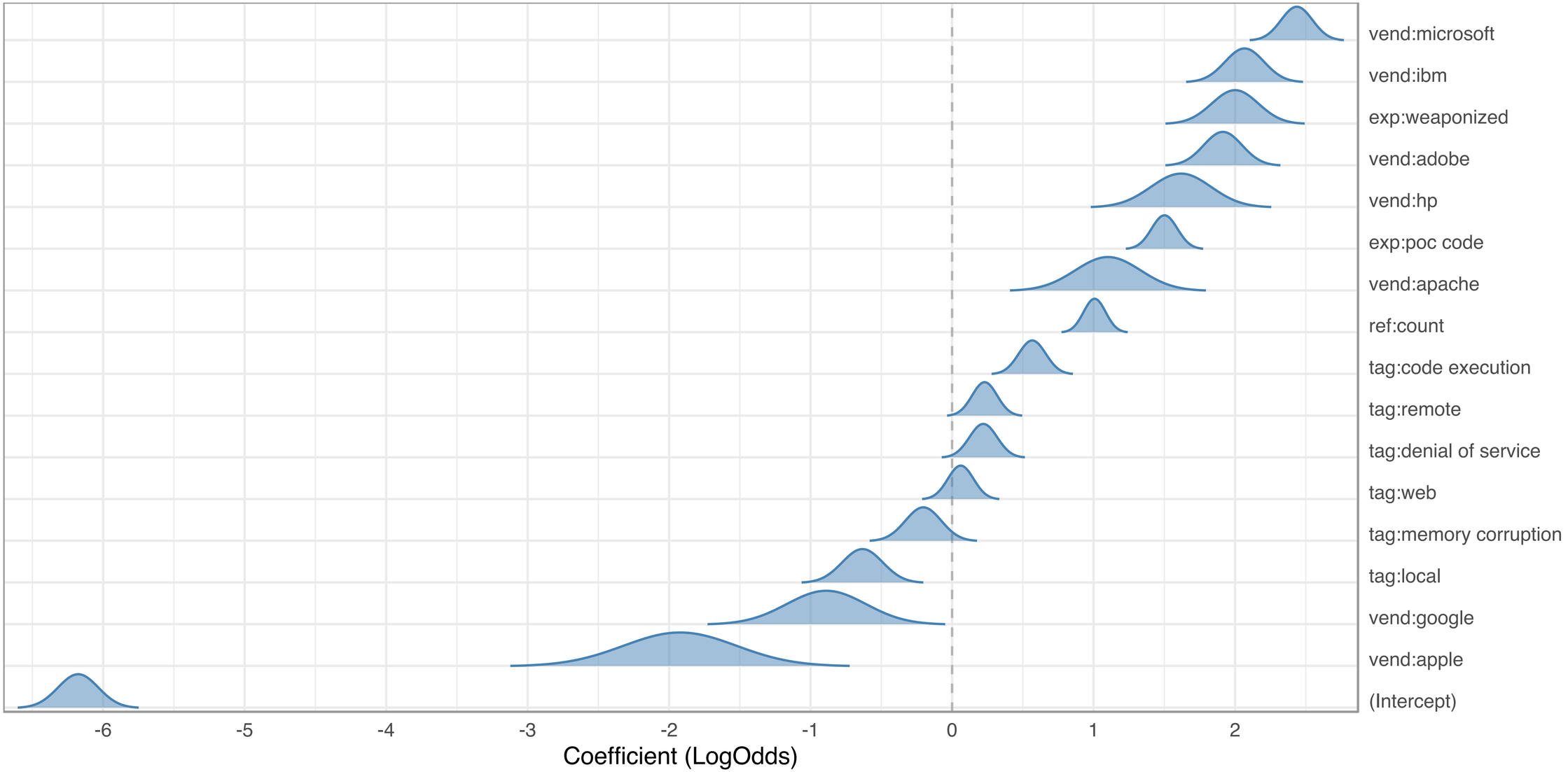

Coefficient
2
1
0
-1
Minimized BIC
exp:poc code
exp:weaponized
ref:count
tag:code execution
tag:memory corruption
tag:remote
tag:denial of service
tag:local
tag:web
tag:privilege escalation
tag:configuration
tag:cryptography
tag:authentication bypass
tag:crafted application
tag:file write
tag:memory leak
tag:asp
tag:logic error
tag:crafted image
tag:modify data
tag:security bypass
tag:injection
tag:authenticated
tag:double free memory
tag:privileged
tag:sql injection
tag:permissions error
tag:crafted file
tag:unauthenticated
tag:unauthorized access
tag:command injection
tag:read data
tag:access control
tag:null pointer
tag:bypass
tag:integer overflow
tag:buffer overflow
tag:information disclosure
vend:microsoft
vend:adobe
vend:ibm
vend:hp
vend:apache
vend:apple
vend:google
vend:linux
vend:opensuse
vend:oracle
vend:canonical
vend:cisco
vend:huawei
vend:redhat
vend:debian
0.1
0.01
0.001
Lambda


tag:double free memory
tag:sql injection
tag:unauthorized access
tag:crafted file
tag:unauthenticated
tag:null pointer
tag:command injection
tag:read data
tag:access control
tag:bypass
tag:integer overflow
tag:buffer overflow
vend:microsoft
vend:adobe
vend:ibm
vend:hp
vend:apache
vend:apple
vend:google
vend:linux
vend:oracle
vend:opensuse
vend:cisco
vend:huawei
vend:canonical
vend:redhat
vend:debian
Minimized BIC
6800
6600
6400
6200
6000
BIC
0.30
0.25
0.20
0.15
PR AUC
0.1
0.01
0.001
Lambda

The Final Variables (prevalence of exploitation)

Variable	Percent Exploited	Variable count
POC Weaponized	37.1%	283
Vend: Microsoft	21.2%	1,333
POC Published	16.5%	2,212
Vend: Adobe	15.9%	747
Memory Corruption	11.7%	1,570
Vend: HP	11.0%	308
Vend: Apache	9.3%	354
Vend: IBM	7.2%	1,202
Code Execution	7.1%	7,278
Denial of Service	5.6%	8,547
Remote	5.4%	10,681
Web	4.9%	5,317
Local	3.5%	2,686
Vend: Apple	0.8%	877
Vend: Google	0.8%	1,852
Ref. Count	Avg Exploited: 5.3% Avg Not Exploited: 3.7%	25,159

Final Variables (Coefficients)



Exploit Prediction Scoring System (EPSS)

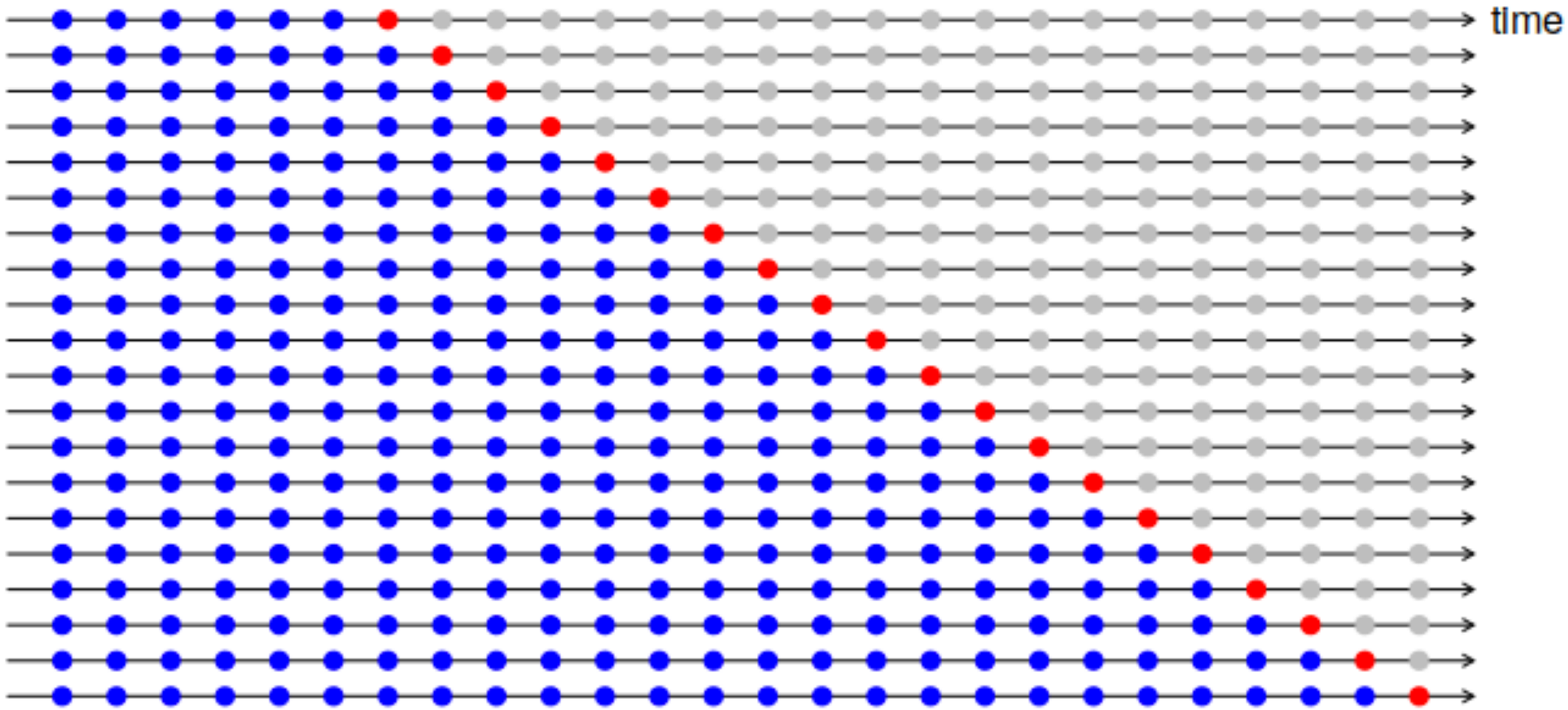
LogOdds = -6.18 +

2.44 * vend:microsoft +
2.07 * vend:ibm +
2.00 * exp:weaponized +
1.91 * vend:adobe +
1.62 * vend:hp +
1.50 * exp:poc code +
1.10 * vend:apache +
1.01 * log(ref:count + 1) +
0.57 * tag:code execution +
0.23 * tag:remote +
0.22 * tag:denial of service +
0.06 * tag:web +
-0.20 * tag:memory corruption +
-0.63 * tag:local +
-0.89 * vend:google +
-1.92 * vend:apple

$$Pr[exploitation] = 1/(1+e^{-LogOdds})$$

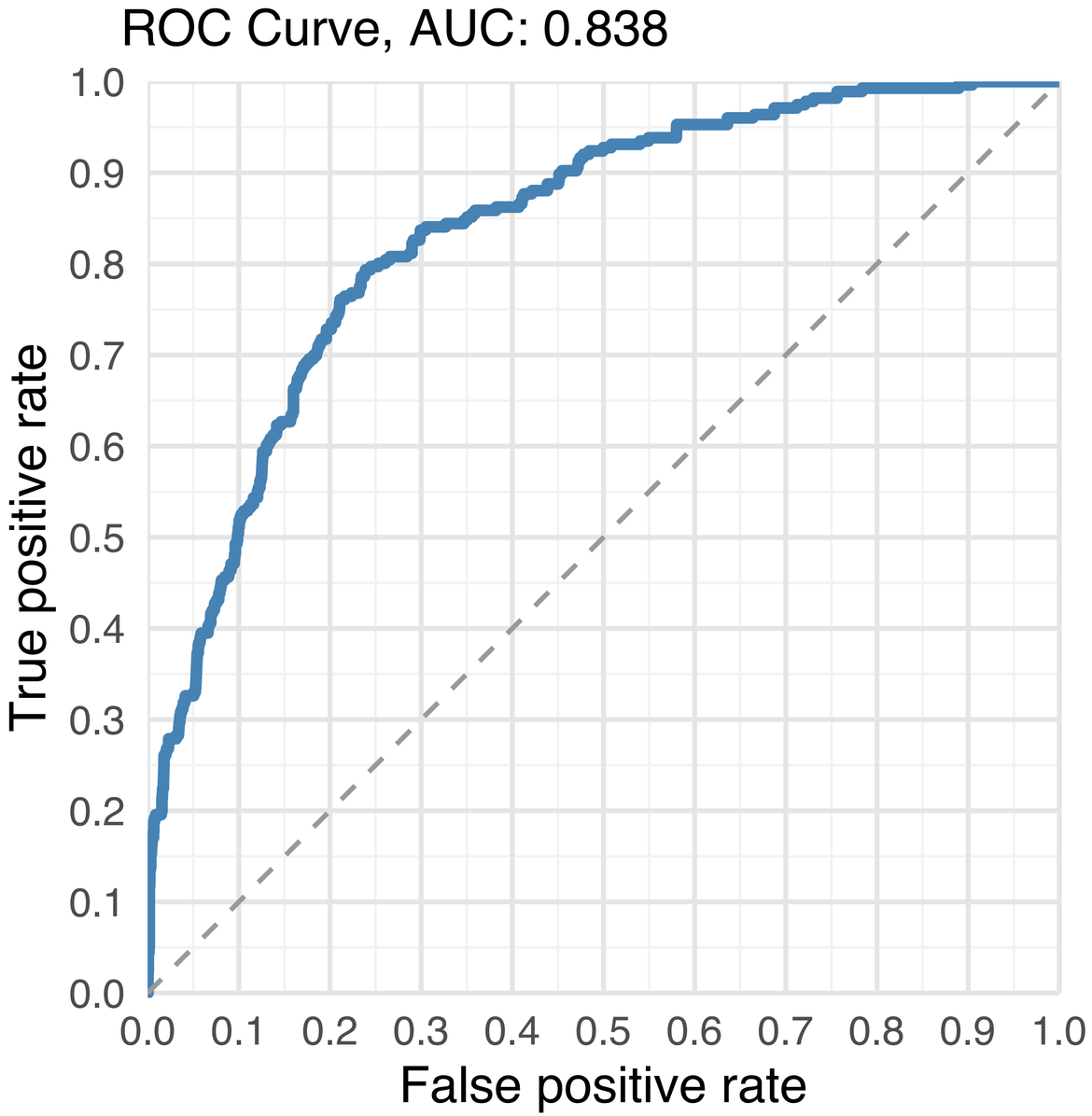
Model Validation

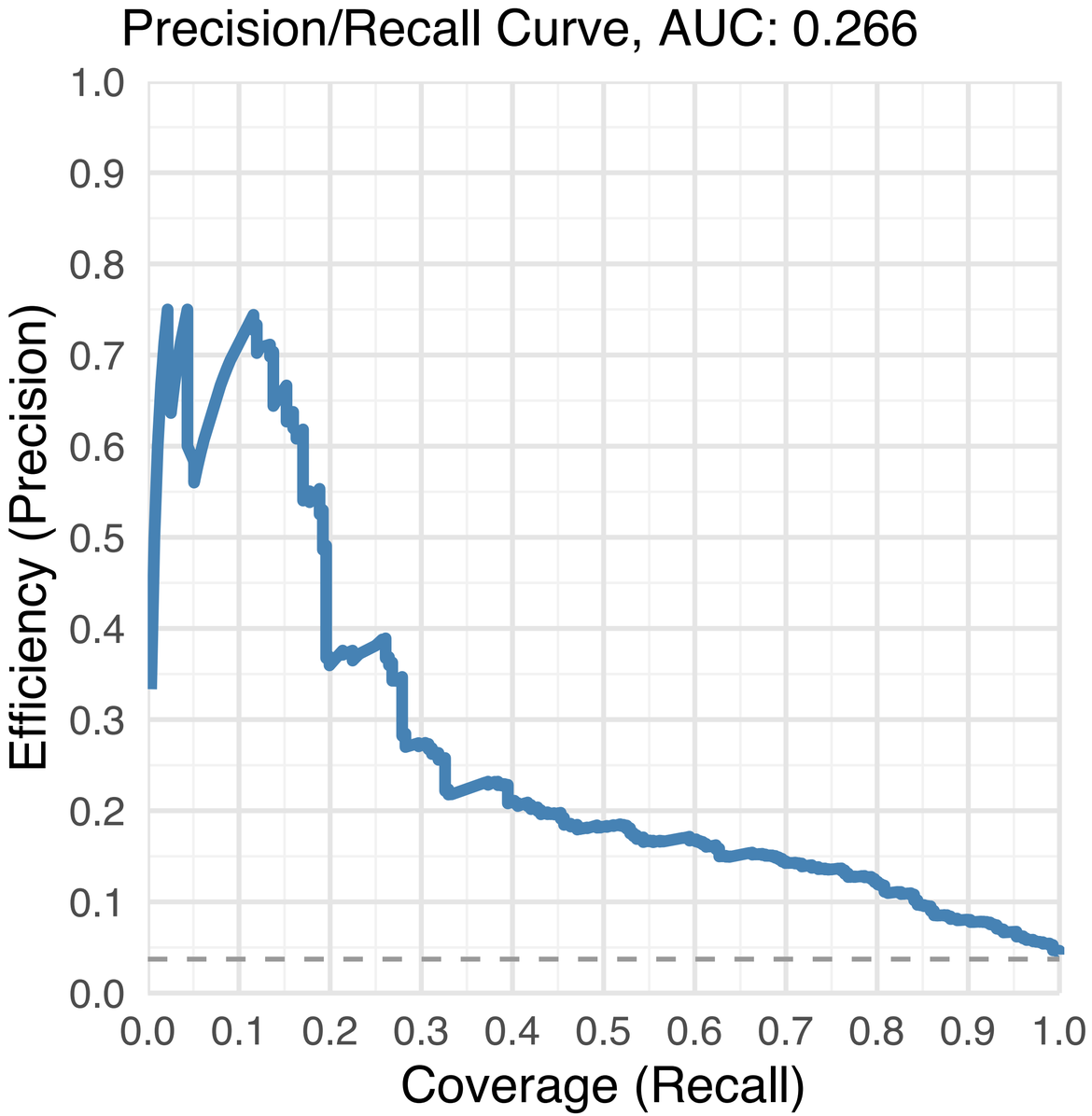
Rolling Forecasting Origin

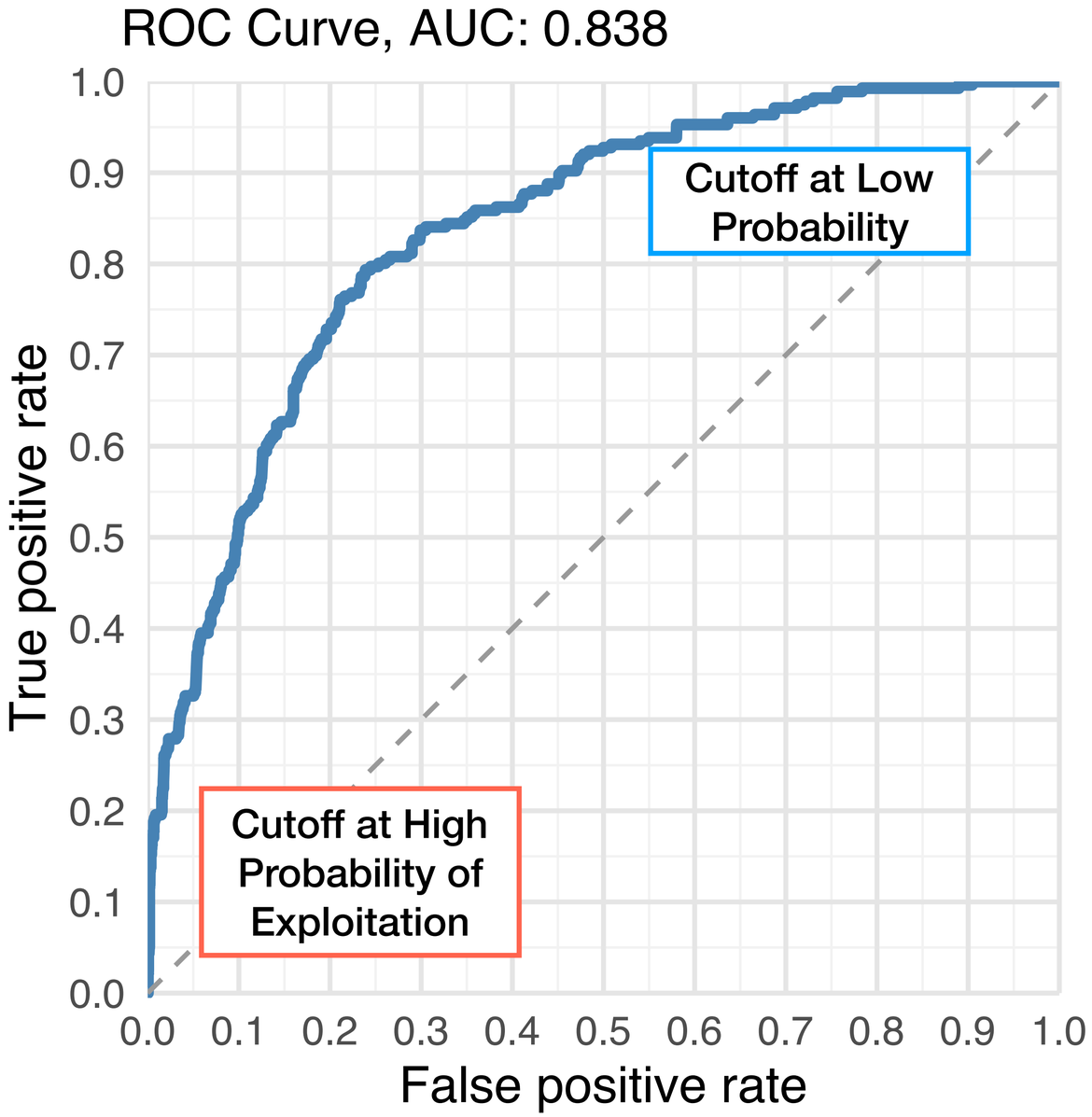


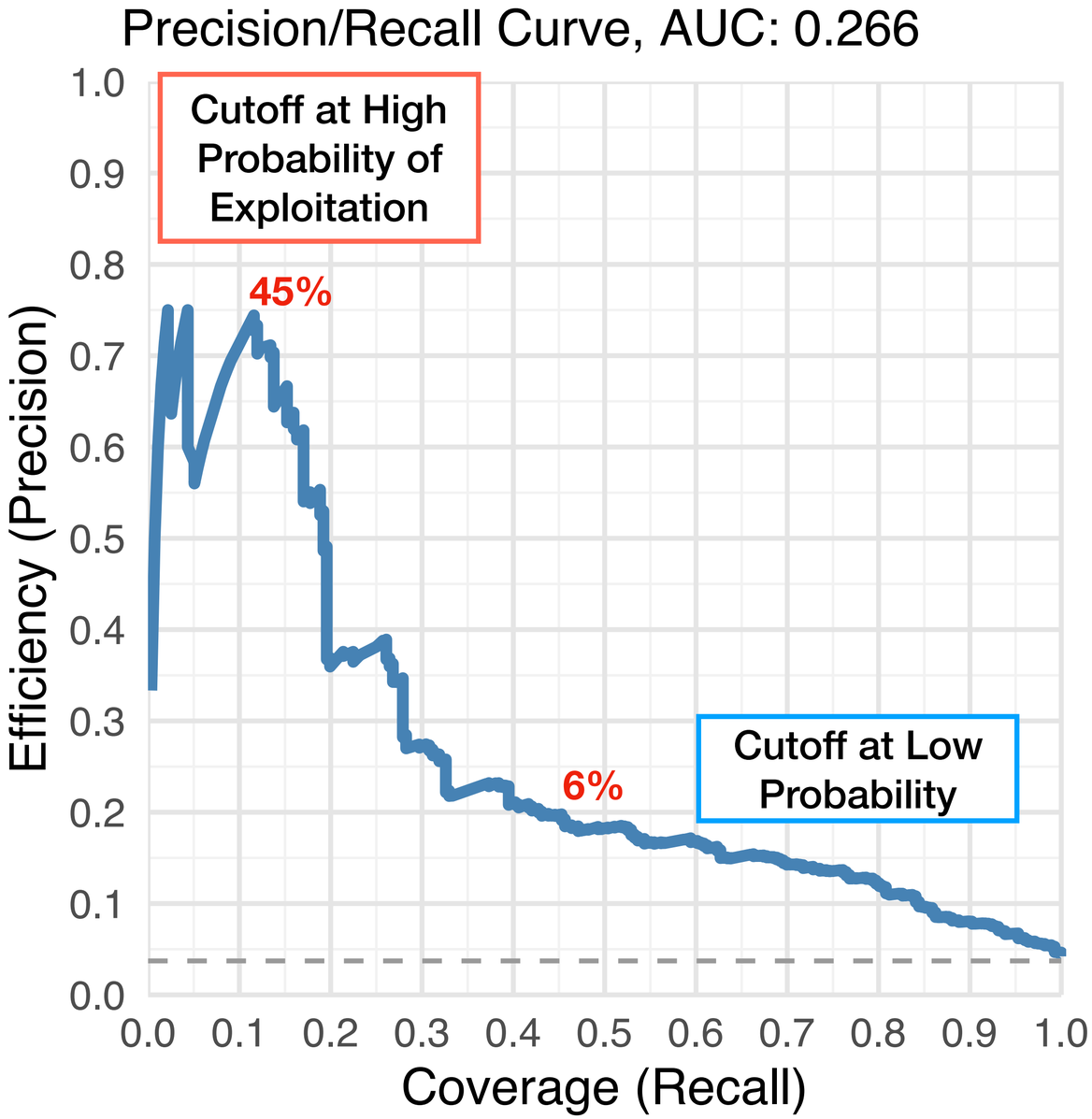
Hyndman, Rob J., and George Athanasopoulos. Forecasting: principles and practice. OTexts, 2013.

https://otexts.com/fpp2/accuracy.html


ROC Curve, AUC: 0.838
True positive rate
False positive rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

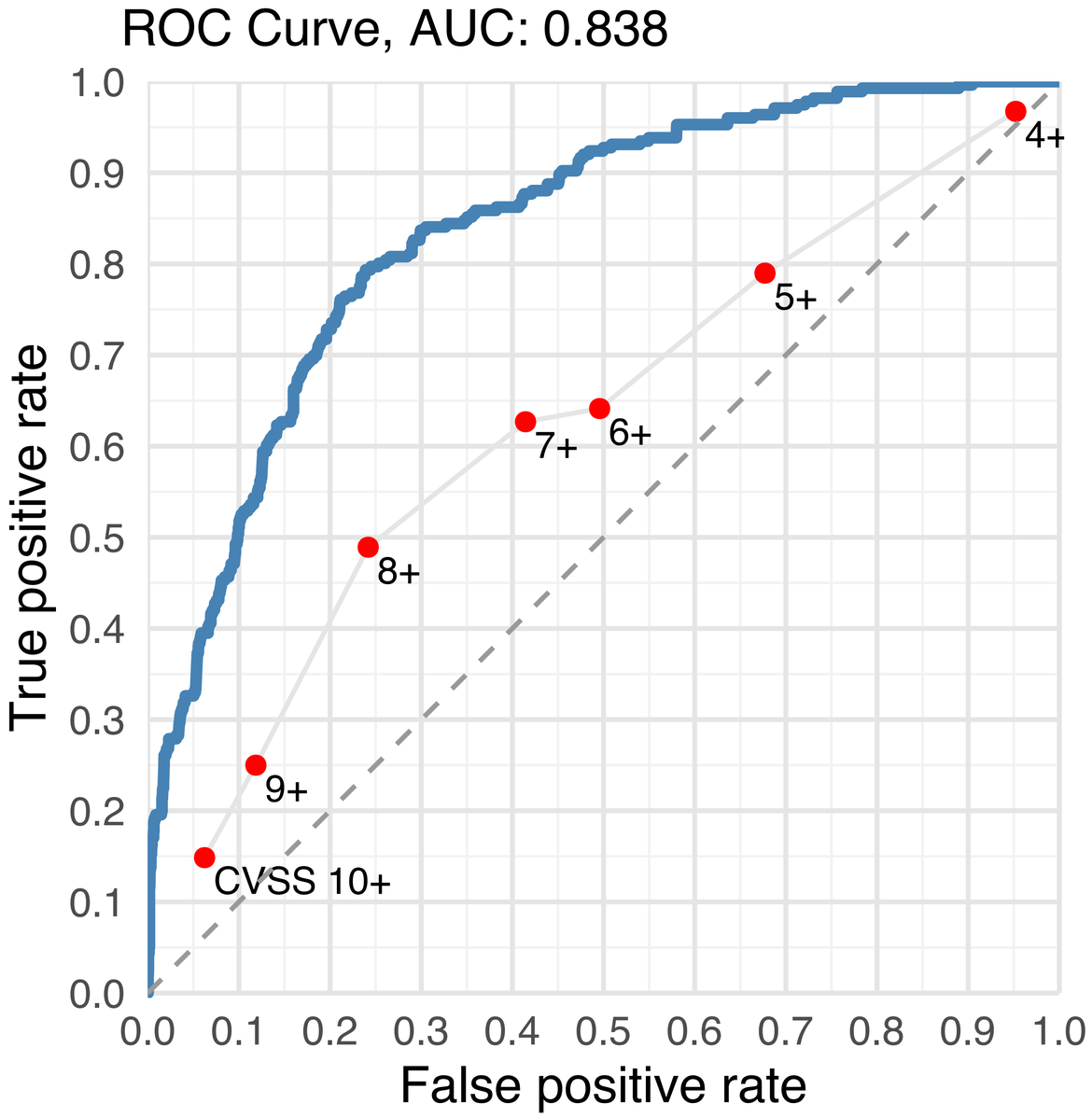

Precision/Recall Curve, AUC: 0.266
Efficiency (Precision)
Coverage (Recall)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

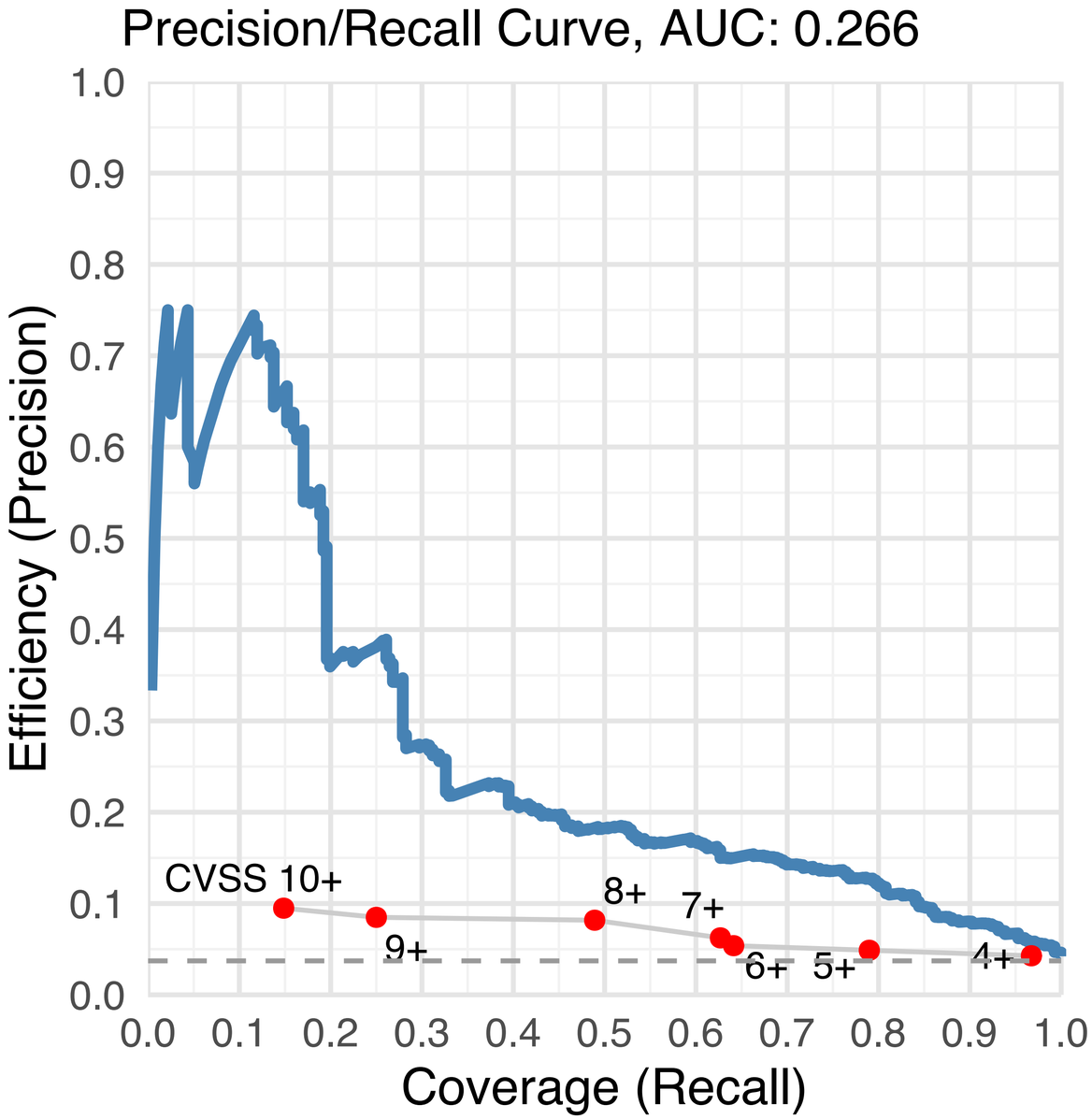

ROC Curve, AUC: 0.838
True positive rate
False positive rate
Cutoff at Low Probability
Cutoff at High Probability of Exploitation
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0


Precision/Recall Curve, AUC: 0.266
Efficiency (Precision)
Coverage (Recall)
Cutoff at High Probability of Exploitation
45%
6%
Cutoff at Low Probability
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

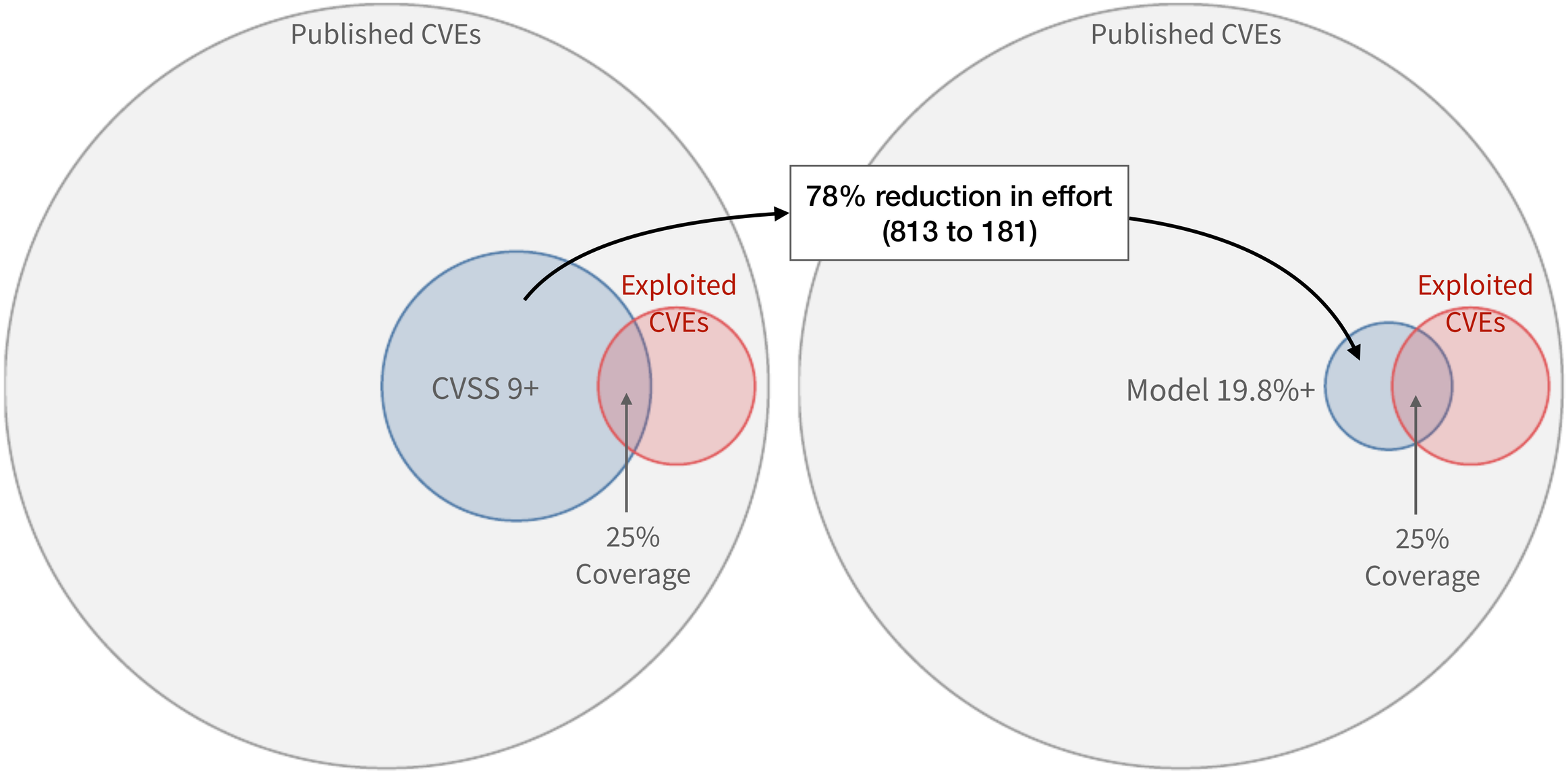
Assessing CVSS Performance

Base Score	False Negatives	False Positives	True Negative	True Positive	**TPR**	**FPR**	**Efficiency (Precision)**	**Coverage (Recall)**
CVSS 10+	235	391	5,883	41	15.1%	4.6%	9.5%	14.9%
CVSS 9+	207	744	5,530	69	31.5%	12%	8.5%	25.0%
CVSS 8+	141	1,517	4,757	135	51%	25.2%	8.2%	48.9%
CVSS 7+	103	2,600	3,674	173	64.9%	43.8%	6.2%	62.7%
CVSS 6+	99	3,110	3,164	177	67.3%	49.9%	5.4%	64.1%
CVSS 5+	58	4,248	2,026	218	79.3%	67.2%	4.9%	79.0%
CVSS 4+	9	5,974	300	267	96.4%	94.8%	4.3%	96.7%

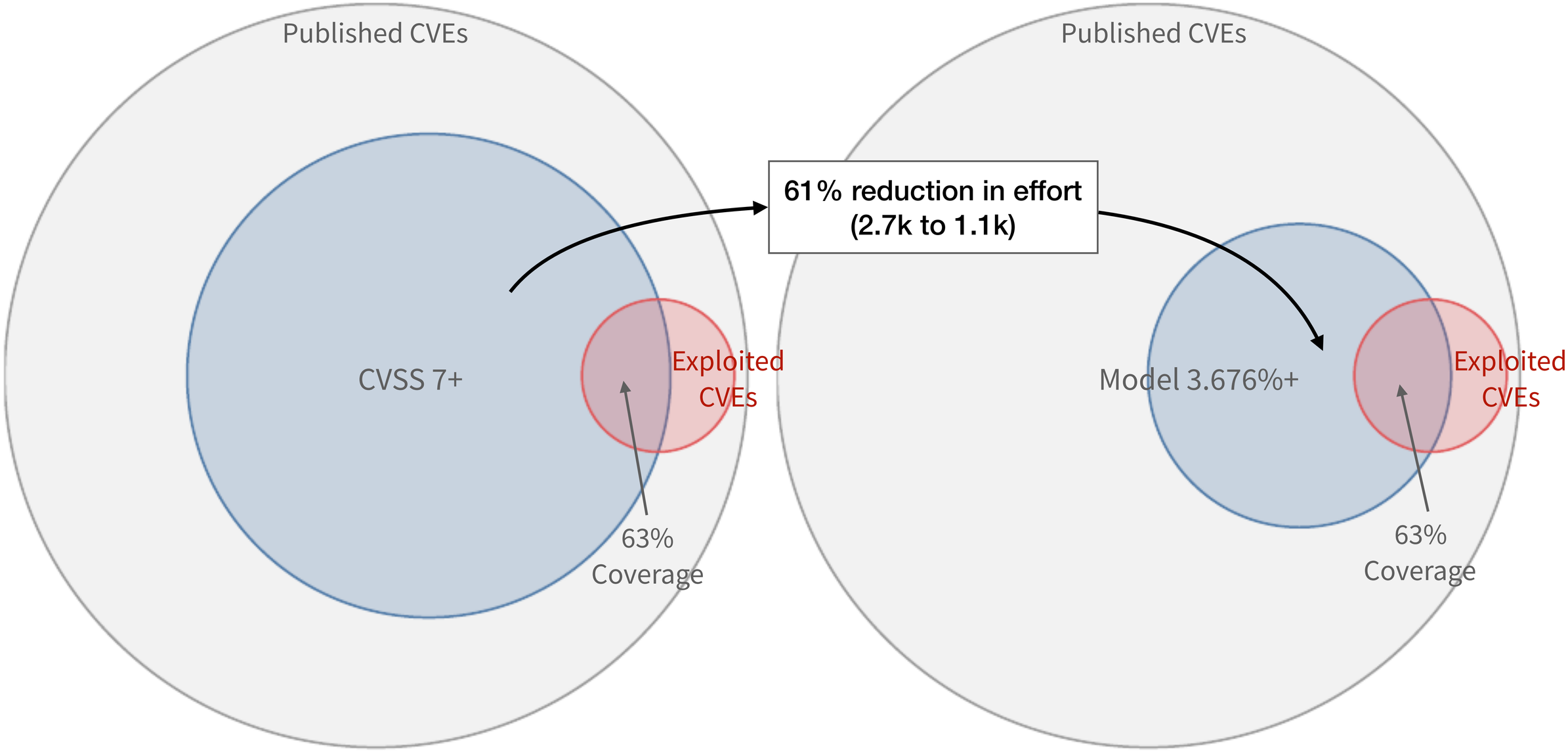

ROC Curve, AUC: 0.838
True positive rate
False positive rate
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
4+
5+
6+
7+
8+
9+
CVSS 10+


Precision/Recall Curve, AUC: 0.266
Efficiency (Precision)
Coverage (Recall)
0.0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
CVSS 10+
9+
8+
7+
6+
5+
4+

Model Performance: Compared to CVSS 9+

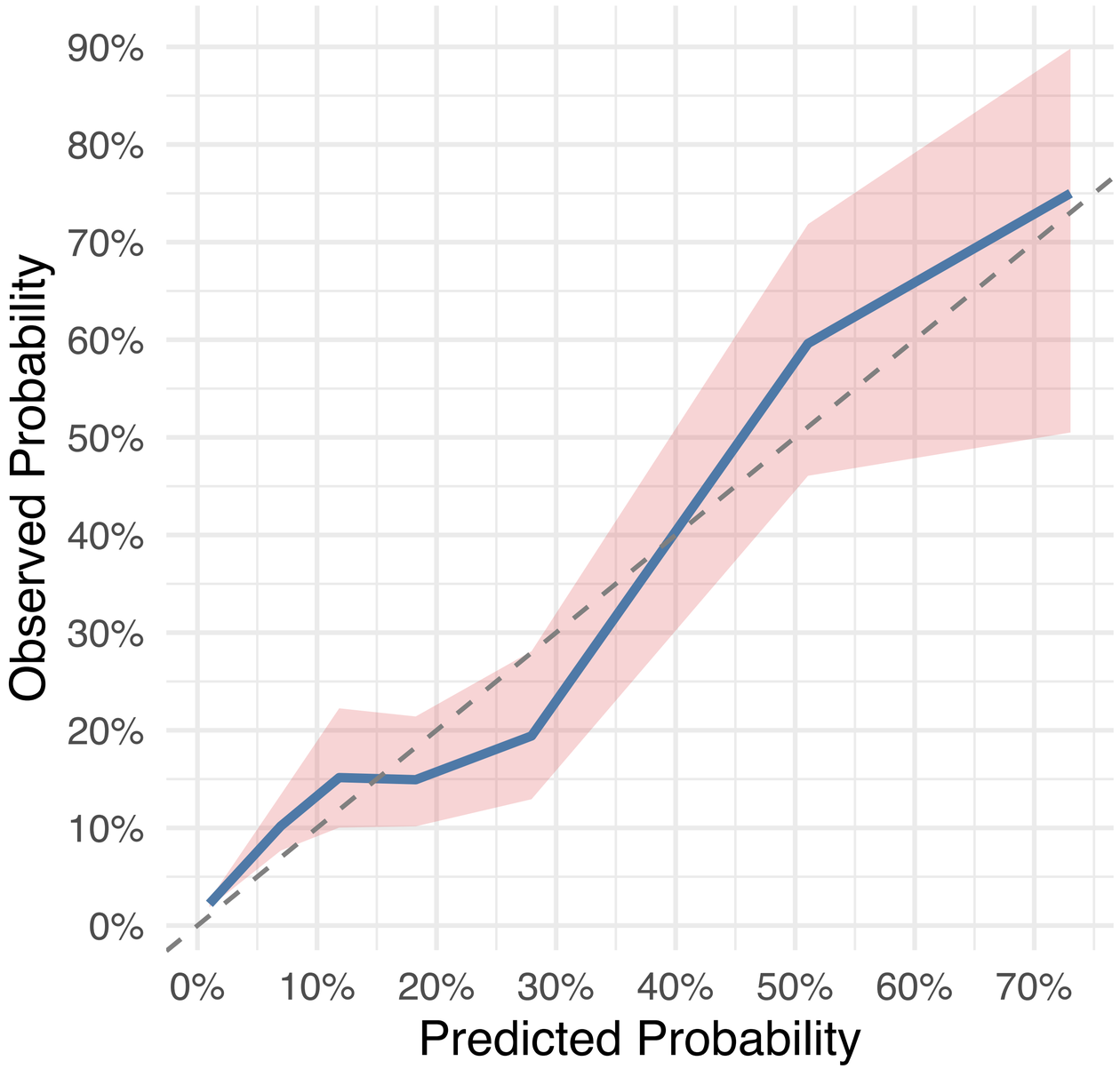


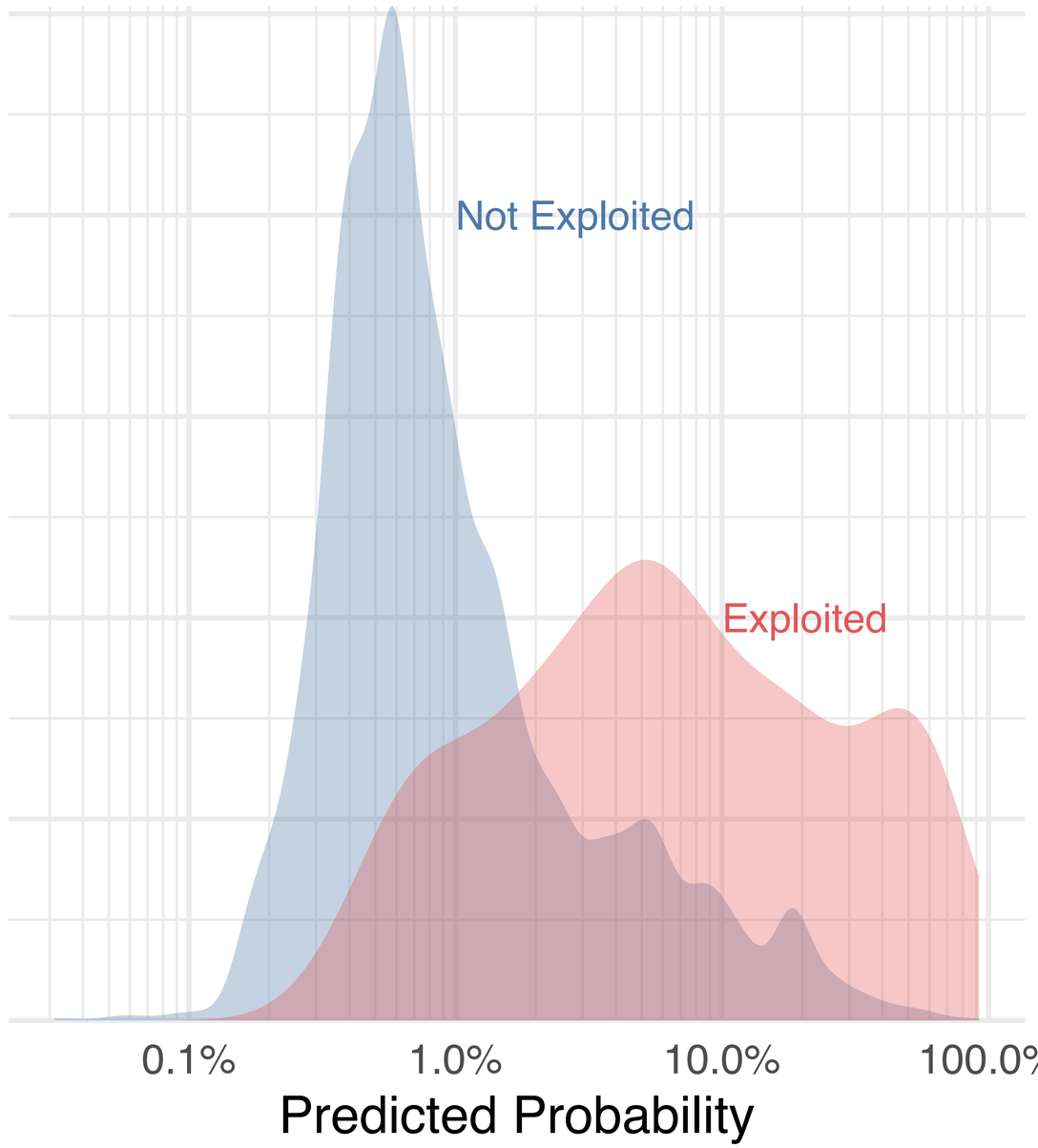
Model Performance: Compared to CVSS 7+

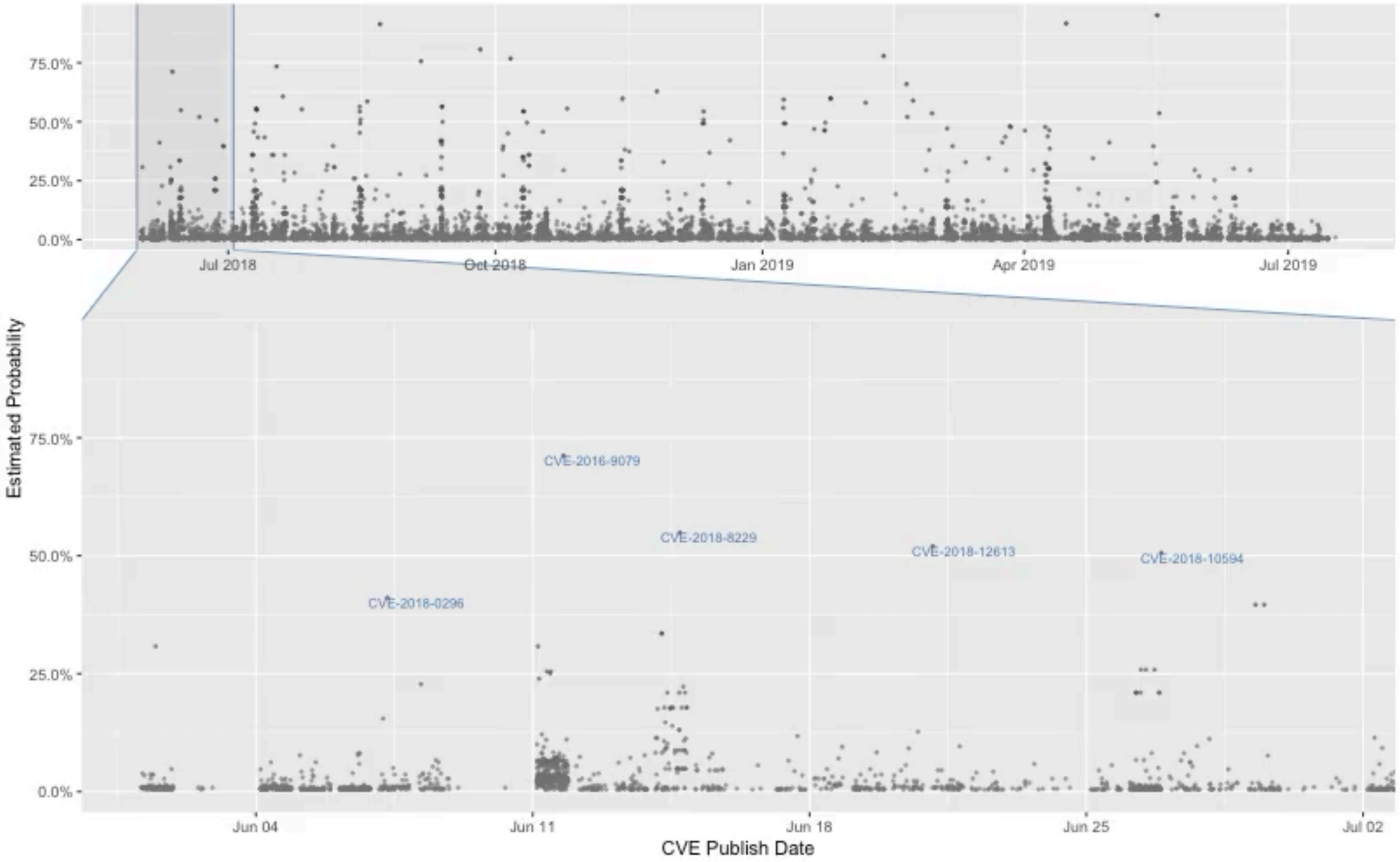


Model Performance: Compared to CVSS 7+

Base Score	Coverage	CVSS Effort	EPSS Effort	EPSS Cutoff	Reduction in Effort
CVSS 10+	14.9%	432	61	44.1%	85.9%
CVSS 9+	25.0%	813	181	19.8%	77.7%
CVSS 8+	48.9%	1,652	738	5.8%	55.3%
CVSS 7+	62.7%	2,773	1,091	3.68%	60.7%
CVSS 6+	64.1%	3,287	1,174	3.35%	64.3%
CVSS 5+	79.0%	4,466	1,712	1.71%	61.7%
CVSS 4+	96.7%	6,241	4,443	0.54%	28.8%


90%
80%
70%
60%
50%
40%
30%
20%
10%
0%
Observed Probability
0%
10%
20%
30%
40%
50%
60%
70%
Predicted Probability


Not Exploited
Exploited
Density
0.1%
1.0%
10.0%
100.0%
Predicted Probability



CVE-2016-9079	71.2%
CVE-2018-8229	54.9%
CVE-2018-12613	52.0%
CVE-2018-10594	50.6%
CVE-2018-0296	41.1%
CVE-2018-12465	39.6%
CVE-2018-12464	39.6%
CVE-2018-8214	33.5%
CVE-2018-8208	33.5%
CVE-2018-11646	30.8%
CVE-2018-5159	30.8%
CVE-2018-10660	25.8%
CVE-2018-10661	25.8%
CVE-2018-10662	25.8%
CVE-2017-5465	25.5%
CVE-2017-5447	25.5%
CVE-2017-5404	25.1%
CVE-2016-9899	24.0%
CVE-2018-4233	22.8%
CVE-2018-0982	22.2%

	Probability	vend:microsoft	vend:ibm	exp:weaponized	vend:adobe	vend:hp	exp:poc code	vend:apache	ref:count	tag:code execution	tag:remote	tag:denial of service	tag:web	tag:memory corruption	tag:local	vend:google	vend:apple
CVE-2019-0708	95.1%	×		×			×		10	×	×						
CVE-2019-0232	91.7%			×			×	×	21	×	×		×				
CVE-2018-11776	91.4%			×			×	×	16	×	×	×	×				
CVE-2018-15961	80.6%			×	×		×		4	×							
CVE-2019-5736	77.9%					×	×	×	49	×	×				×	×	
CVE-2018-17456	76.8%			×			×		16	×	×	×					
CVE-2018-16509	75.7%			×			×		15	×	×	×					
CVE-2018-1612	73.5%		×	×			×		3		×						
CVE-2016-9079	71.2%			×			×		11	×	×	×	×				
CVE-2019-8942	66.0%			×			×		9	×	×	×					
CVE-2018-19518	62.9%			×			×		17		×		×				
CVE-2018-7602	60.7%			×			×		7	×	×	×					
CVE-2019-1653	60.1%			×			×		15		×		×				
CVE-2018-8544	59.9%	×					×		4	×	×	×					
CVE-2019-1652	59.8%			×			×		8	×	×		×				
CVE-2019-0539	59.4%	×					×		5	×	×	×		×			
CVE-2019-6340	59.0%			×			×		6	×	×	×	×				
CVE-2018-15473	58.6%			×			×		15		×						
CVE-2018-20250	58.1%			×			×		8	×		×					
CVE-2018-8466	56.4%	×					×		4	×	×	×	×	×			

Tool and Demo

- http://kennaresearch.com/tools/epss-calculator

Jay Jacobs
@jayjacobs

Michael Roytman
@mroytman

- ✓ Too many vulnerabilities to patch them all
- ✓ Most vulnerabilities are never used by attackers
- ✓ Use Coverage and Efficiency to measure prioritization strategies
- ✓ EPSS outperforms CVSS on every metric
- ✓ More about EPSS is available at **kennaresearch.com**