RAPIDS プロジェクト - データ分析フェーズ

フィッシングサイト検出のためのデータ分析結果

Realtime AI-Powered Phishing Detection System

本報告書の目的：

- フィッシングサイトと正常サイトの証明書・ドメイン情報を比較し、**有効な検出モデルを設計するための指標** を導き出す
- 分析対象はCertstreamなどから取得した**新規に発行された証明書情報**をもとに、フィッシングサイトと正常サイトの特徴量を比較検証
- 本レポートの結論をもとに、**リアルタイム検出モデルへの実装**や**継続的なモニタリング** の指針を得る

分析の概要：データセットや目的の整理

各項目の比較分析：証明書認証レベル、レジストラ分布、発行者分布、有効期間など

考察・活用例：分析結果からの示唆やモデル構築への活用方針

まとめ・今後の展望：全体の結論と今後の課題

分析の概要

分析データセット

- **フィッシングサイト（ website_data）**: 12,072件
- **正常サイト（ normal_sites）**: 9,591件
- **データ品質** : 証明書解析成功率99.4%

分析の目的

- フィッシングサイトと正常サイトの**特徴的な差異** を特定
- 効果的な検出モデルのための**特徴量**を選定
- リアルタイム検出に適した**判別基準**を確立

1. 証明書認証レベルの分布

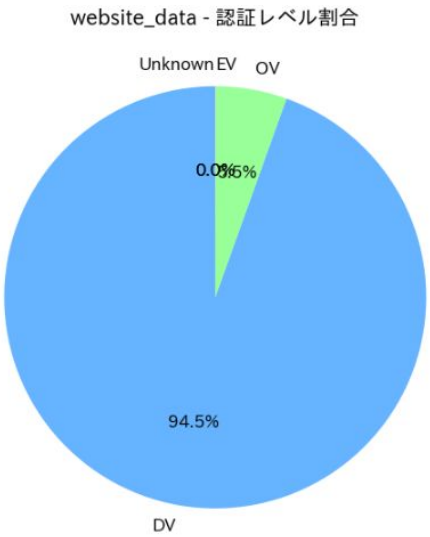
フィッシングサイト: 約94.5%がDV（Domain Validation）。EVはほぼ皆無

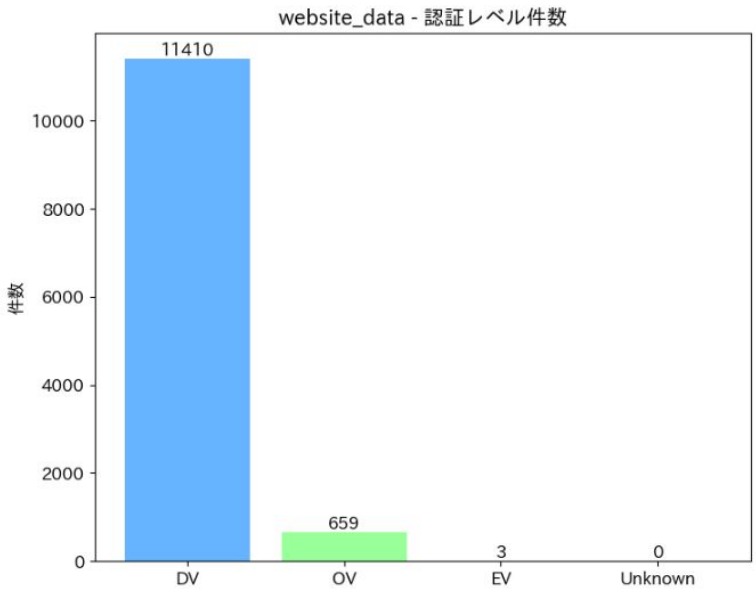
正常サイト: DVが約72.7%でまだ多数だが、OV/EVの利用比率が27.3%ほどある

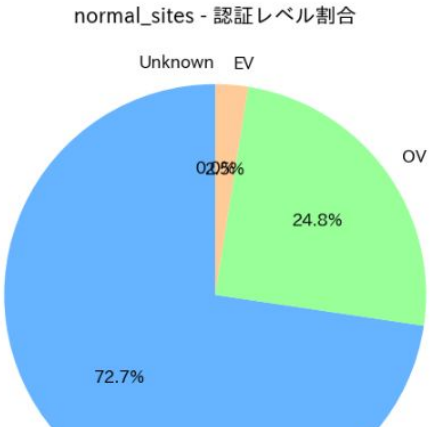
考察:

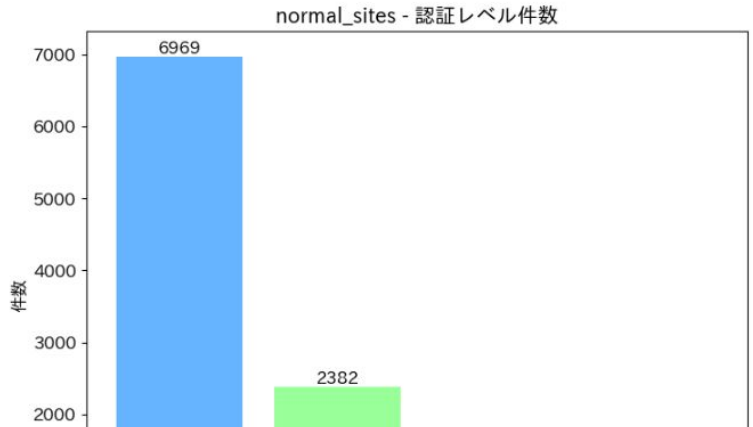
- 高度な証明書（OV/EV）は手間と費用がかかるため、フィッシングサイトではDVが圧倒的に多い。
- しかし、正常サイトでもDVが主流なので、「DVだからすぐフィッシング」とは決め打ちできない。
- **他の特徴量（レジストラ・有効期間など）との併用** が必須。

1.


website_data - 認証レベル割合
Unknown
EV
OV
0.0%
5.6%
94.5%
DV


website_data - 認証レベル件数
件数
11410
659
3
0
DV
OV
EV
Unknown


normal_sites - 認証レベル割合
Unknown
EV
OV
0.0%
2.5%
24.8%
72.7%


normal_sites - 認証レベル件数
件数
6969
2382

IA フィッシングサイトを作成する人がOVの証明書を取得するのは簡単ですか？

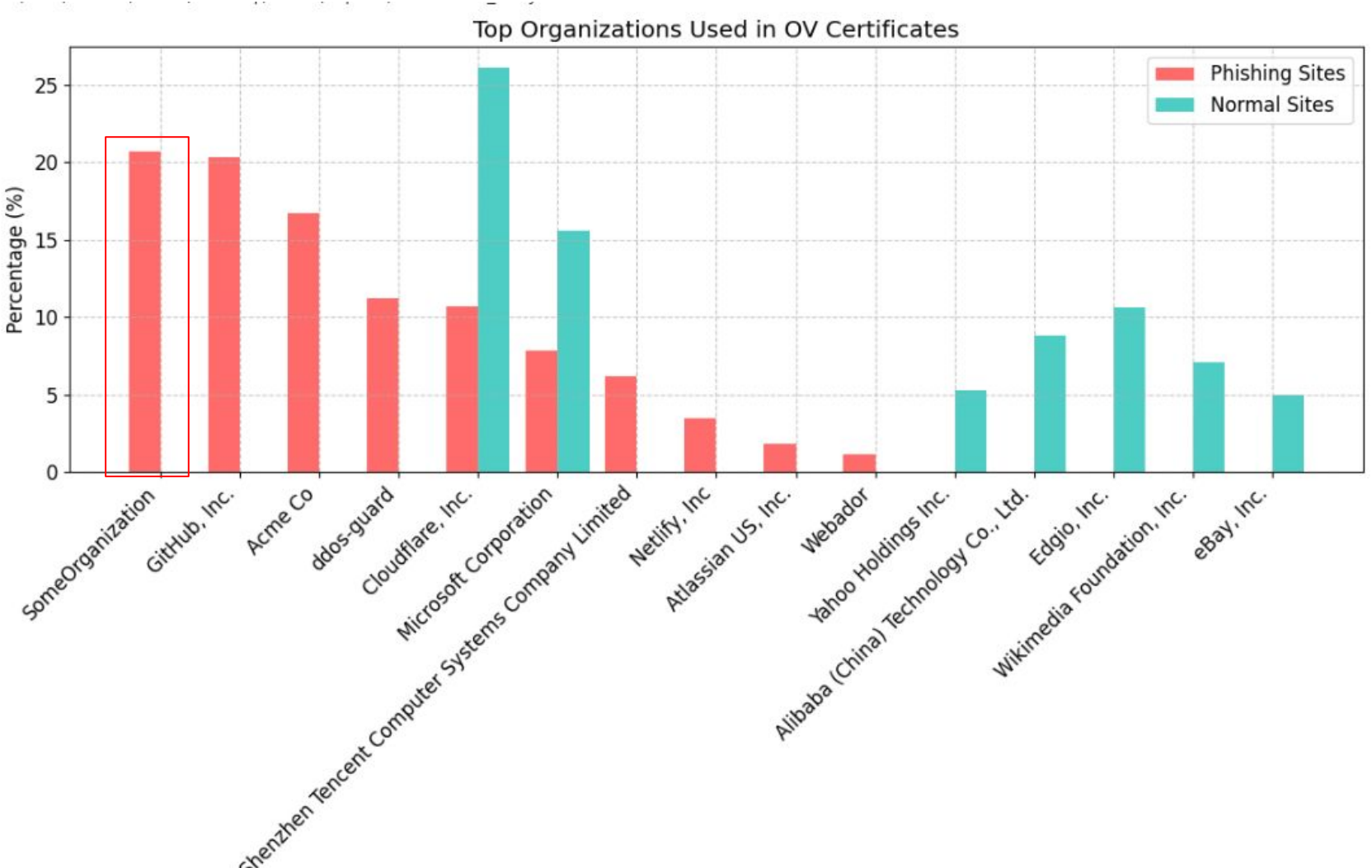
フィッシングサイトを作成する人がOV証明書を取得するのは、一般的には簡単ではありません。その理由はいくつかあります：

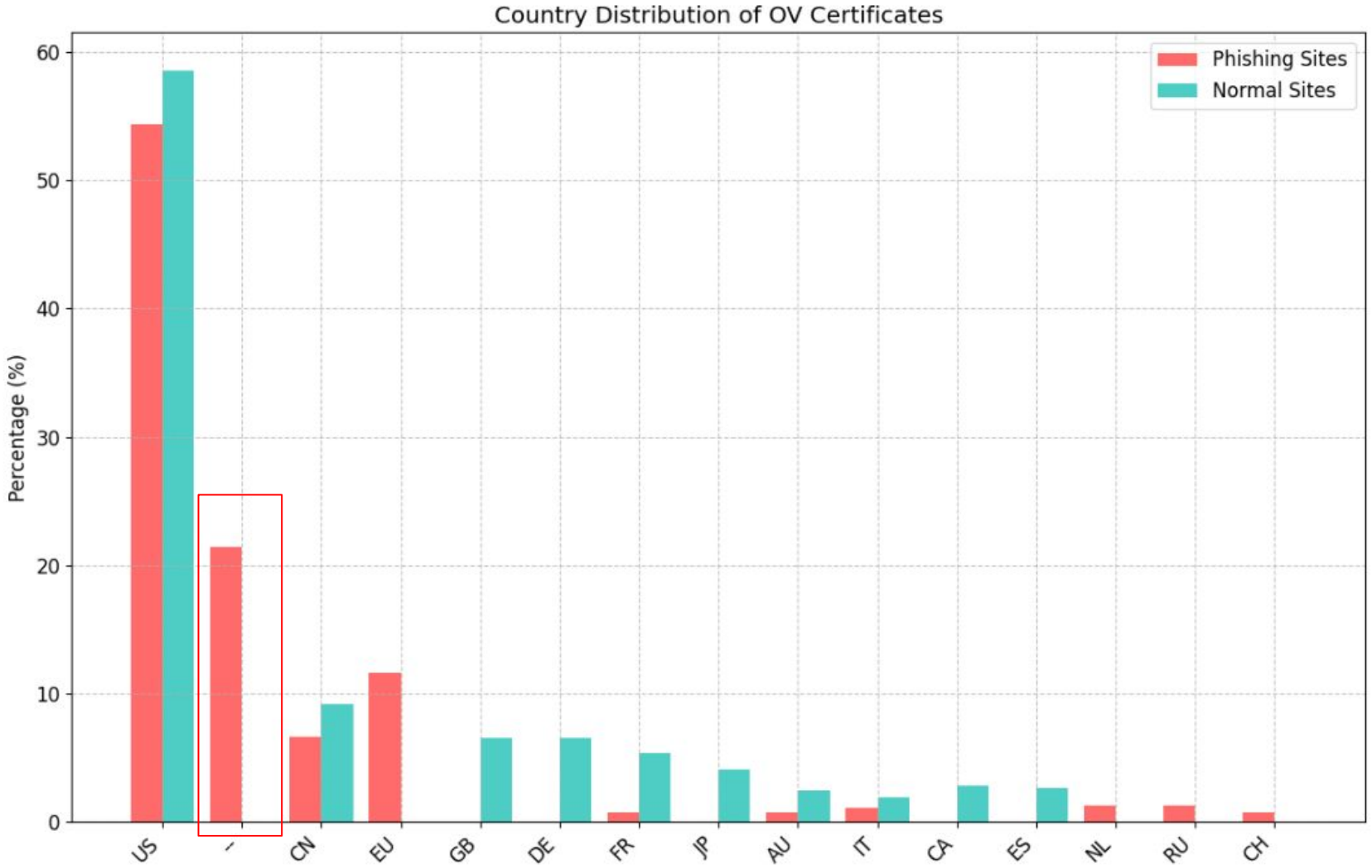
1. **厳格な組織審査** OV証明書の取得には組織の実在性確認が必要で、認証局(CA)は会社登記や法人登録などの公的データベースで組織を確認します。また、登録された電話番号への確認連絡や、提出された書類の検証も行われます。
2. **物理的な所在地の検証** OV証明書では、組織の実際の住所が検証され、郵送による確認や公的記録との照合が行われることがあります。
3. **法的責任** 不正な目的でOV証明書を取得すると詐欺罪や身分詐称などの法的責任が問われる可能性があります。
4. **コストと時間** OV証明書はDV証明書よりも高価で、取得プロセスも1〜3営業日かかります。これは一時的なフィッシング活動には不向きです。
5. **トレーサビリティ** OV証明書の申請には実名や連絡先などの個人情報が必要で、犯罪に使用した場合の追跡可能性が高まります。

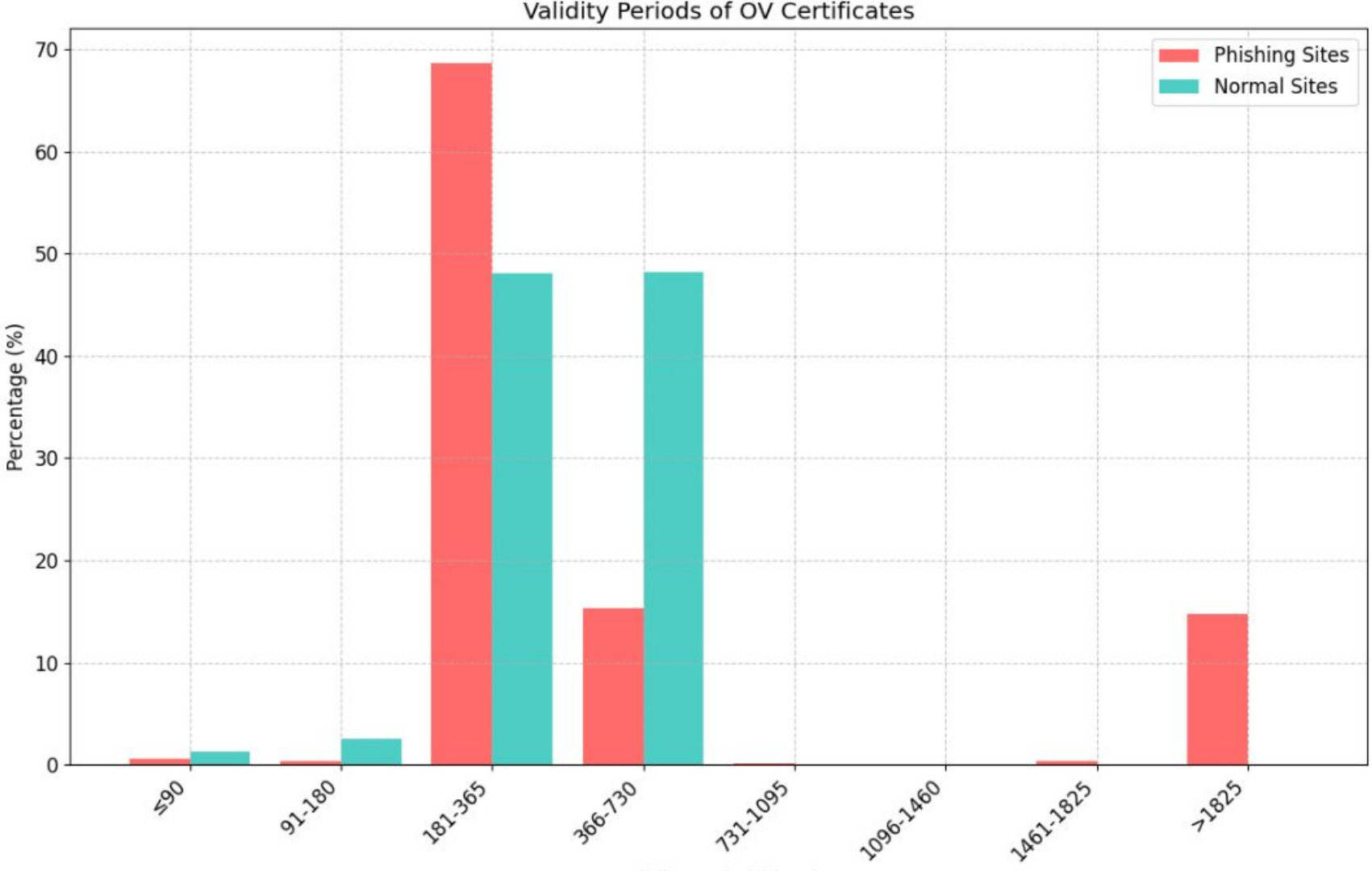
ただし、以下のような方法で悪意ある人がOV証明書を取得しようとする可能性もあります：

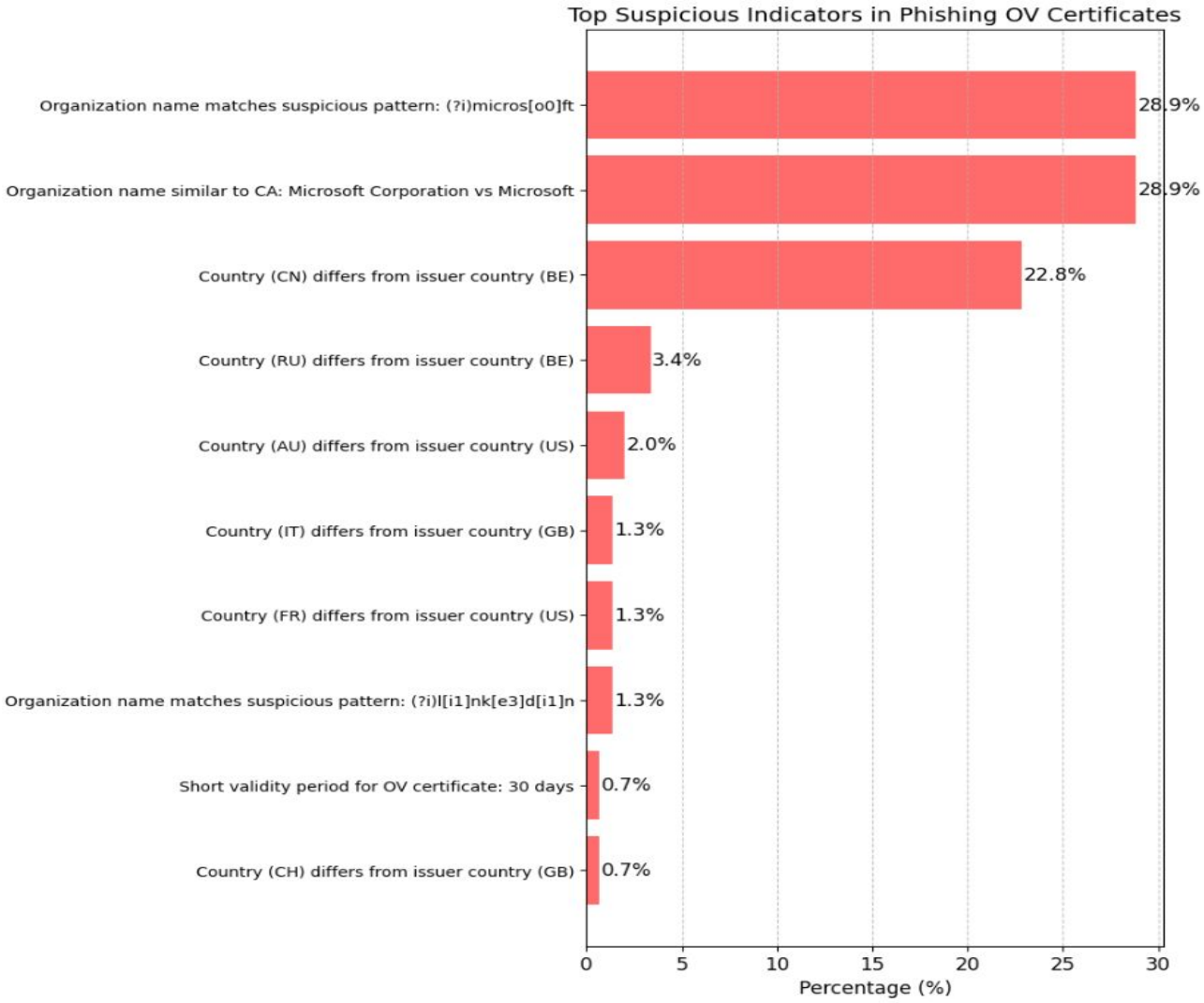
- 実在する正規の会社を設立してから証明書を取得する
- 企業のアイデンティティを盗用する
- すでに取得したOV証明書を不正に転用する

このため、ユーザーはURLだけでなく、証明書の詳細情報（組織名など）も確認することが重要です。多くのフィッシング攻撃はOV証明書よりも取得が容易なDV証明書か、証明書なしのサイトを利用する傾向があります。


Top Organizations Used in OV Certificates
Phishing Sites
Normal Sites
Percentage (%)
0
5
10
15
20
25
SomeOrganization
GitHub, Inc.
Acme Co
ddos-guard
Cloudflare, Inc.
Microsoft Corporation
Shenzhen Tencent Computer Systems Company Limited
Netlify, Inc
Atlassian US, Inc.
Webador
Yahoo Holdings Inc.
Alibaba (China) Technology Co., Ltd.
Edgio, Inc.
Wikimedia Foundation, Inc.
eBay, Inc.


Country Distribution of OV Certificates
Phishing Sites
Normal Sites
Percentage (%)
0
10
20
30
40
50
60
US
–
CN
EU
GB
DE
FR
JP
AU
IT
CA
ES
NL
RU
CH


Validity Periods of OV Certificates
Phishing Sites
Normal Sites
Percentage (%)
0
10
20
30
40
50
60
70
≤90
91-180
181-365
366-730
731-1095
1096-1460
1461-1825
>1825


Top Suspicious Indicators in Phishing OV Certificates
Organization name matches suspicious pattern: (?i)micros[o0]ft
Organization name similar to CA: Microsoft Corporation vs Microsoft
Country (CN) differs from issuer country (BE)
Country (RU) differs from issuer country (BE)
Country (AU) differs from issuer country (US)
Country (IT) differs from issuer country (GB)
Country (FR) differs from issuer country (US)
Organization name matches suspicious pattern: (?i)l[i1]nk[e3]d[i1]n
Short validity period for OV certificate: 30 days
Country (CH) differs from issuer country (GB)
28.9%
28.9%
22.8%
3.4%
2.0%
1.3%
1.3%
1.3%
0.7%
0.7%
0
5
10
15
20
25
30
Percentage (%)

ok

生データ

00_13_12_certificate_analysis_tool_03.ipynb

website_data の認証レベル分析を開始..
取得したレコード数 12072

normal_sites の認証レベル分析を開始..
取得したレコード数 9591

=== website_data の証明書認証レベル分析サマリー ===
総証明書数: 12,072
DV証明書数: 11,410 (94.5%)
OV証明書数: 659 (5.5%)
EV証明書数: 3 (0.0%)

=== normal_sites の証明書認証レベル分析サマリー ===
総証明書数: 9,591
DV証明書数: 6,969 (72.7%)
OV証明書数: 2,382 (24.8%)
EV証明書数: 240 (2.5%)

2. レジストラ分布の特徴

分析結果

- **フィッシングサイト**: 特定レジストラ（Gandi SAS、阿里云など）に**偏り**が大きい
- **正常サイト**: より**多様なレジストラ**を使用
- Gandi SASが約30%を占める一方、正常サイト最大はGoDaddyで16%程度

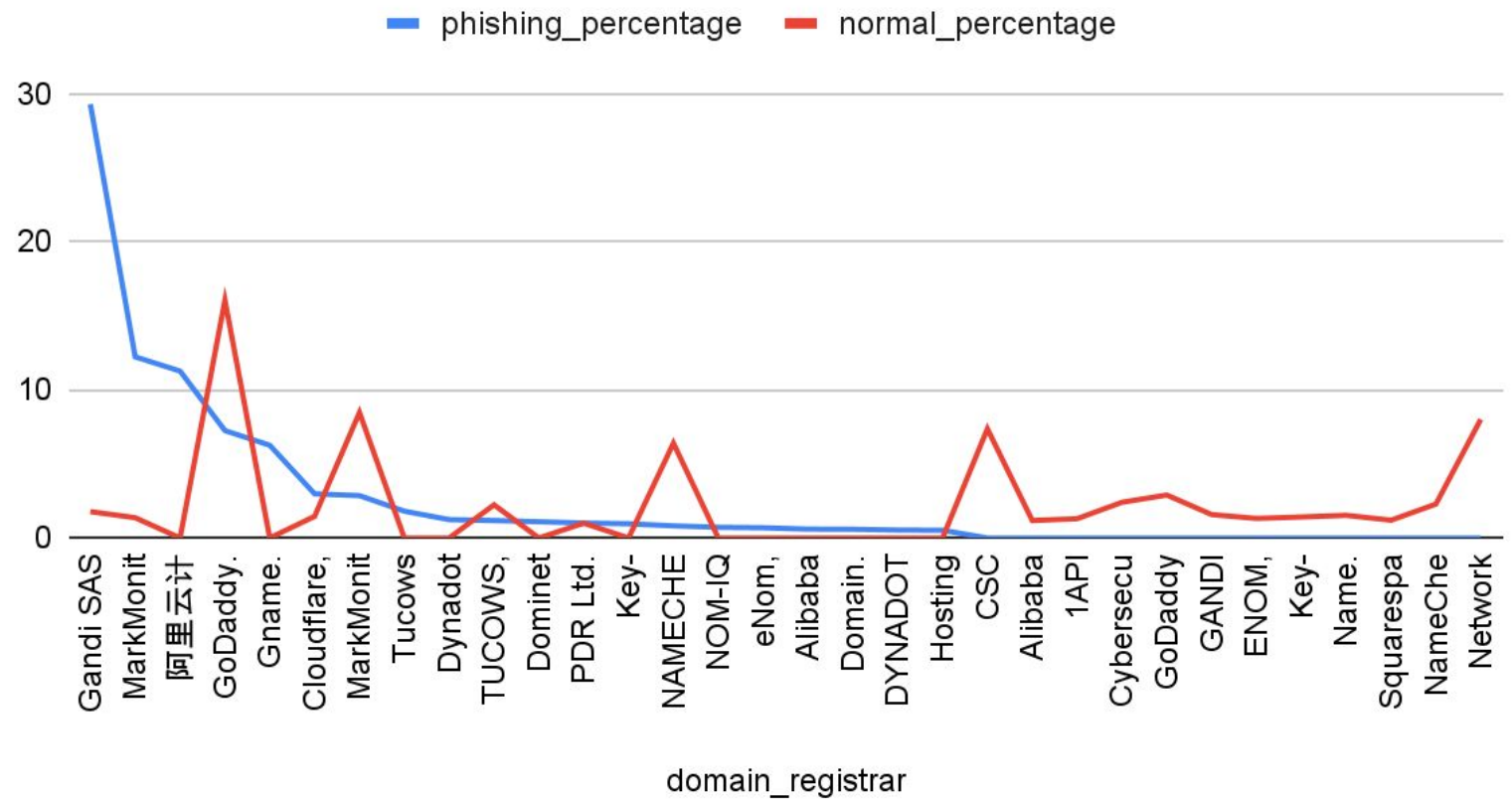
検知指標になるか？

- レジストラ情報は軽量に取得でき、有効なスクリーニング要素。
- フィッシング運営者は設定が簡易・コストが安いレジストラを集中利用する可能性が高い。
- ただし、GandiやAlibaba Cloudも正規利用されるので、**組み合わせ指標**として扱うのが望ましい。
- 短期証明書や多層ドメインとの組み合わせでさらに**精度向上**が期待できる

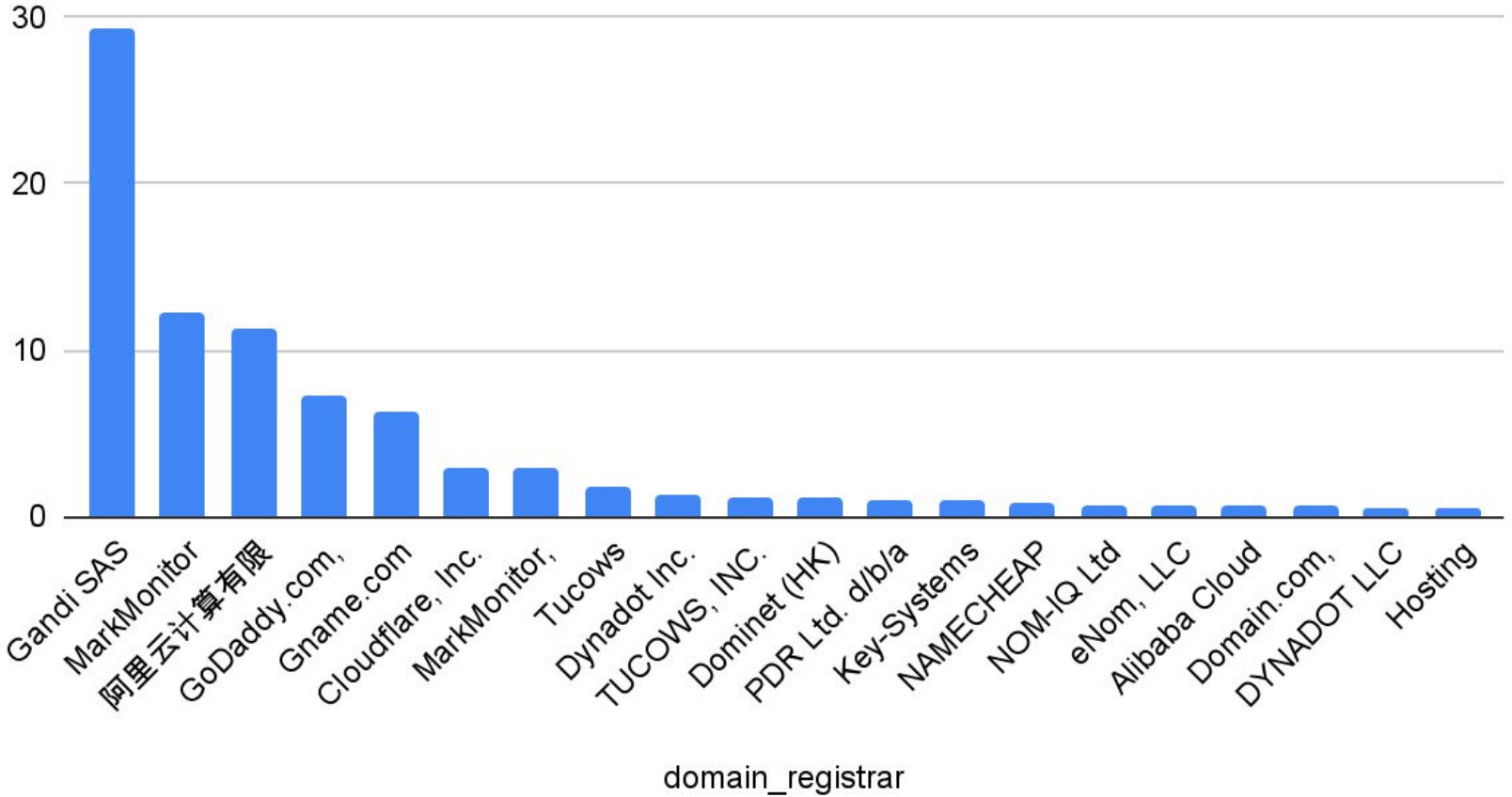
00_02_registrar_distribution_analysis.ipynb

registrar_comparison

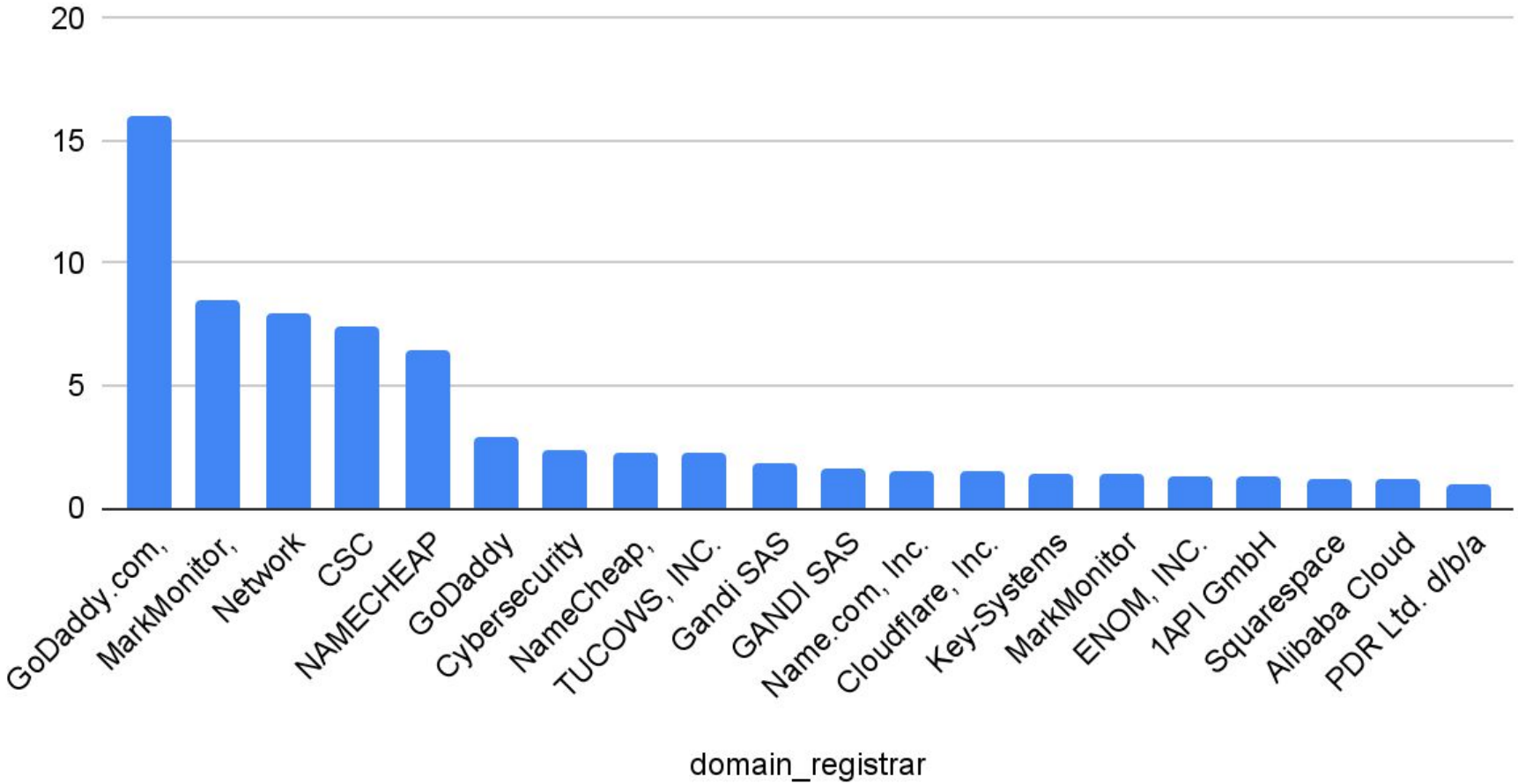
表示の都合上、上位20で区切っています



フィッシングサイトが利用しているレジストラ



一般のサイトが利用しているレジストラ



3. 証明書発行者（Issuer）の分布

分析結果

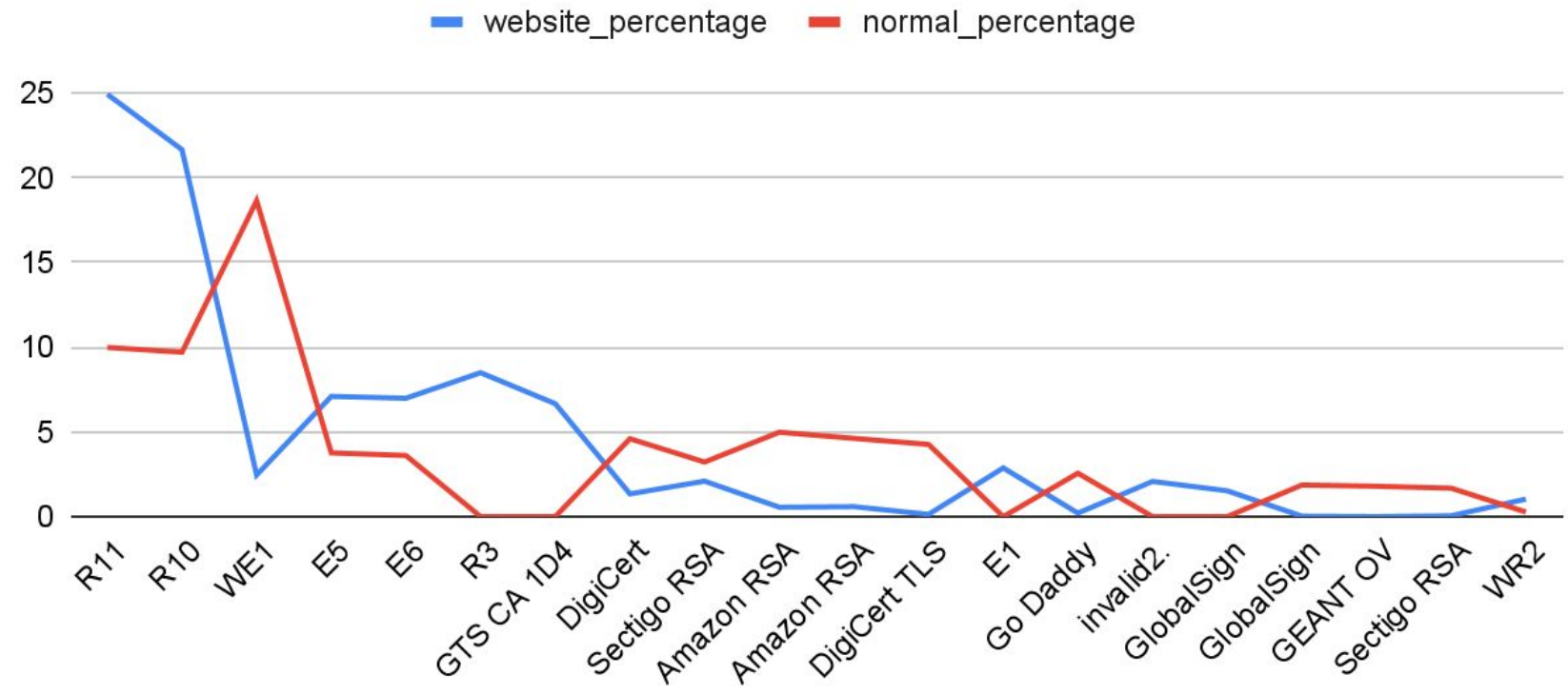
- **フィッシングサイト**: 特定の発行者（R11, R10など）に偏り
- **正常サイト**: より多彩な発行者を利用（191種類）
- フィッシングサイトは上位3発行者で55%以上、正常サイトは約38%

検知指標になるか？

- **発行者情報**は検出モデルの重要な特徴量
- **多様性の低さ**が**リスク要素**となるケースが多い(発行者に**特定パターン**の集中が見られるのは明確な差異。)
- 実際の運用ではLet's Encryptやその派生は正常・フィッシング問わず多いため、**ほかの特徴との組み合わせ**が重要。

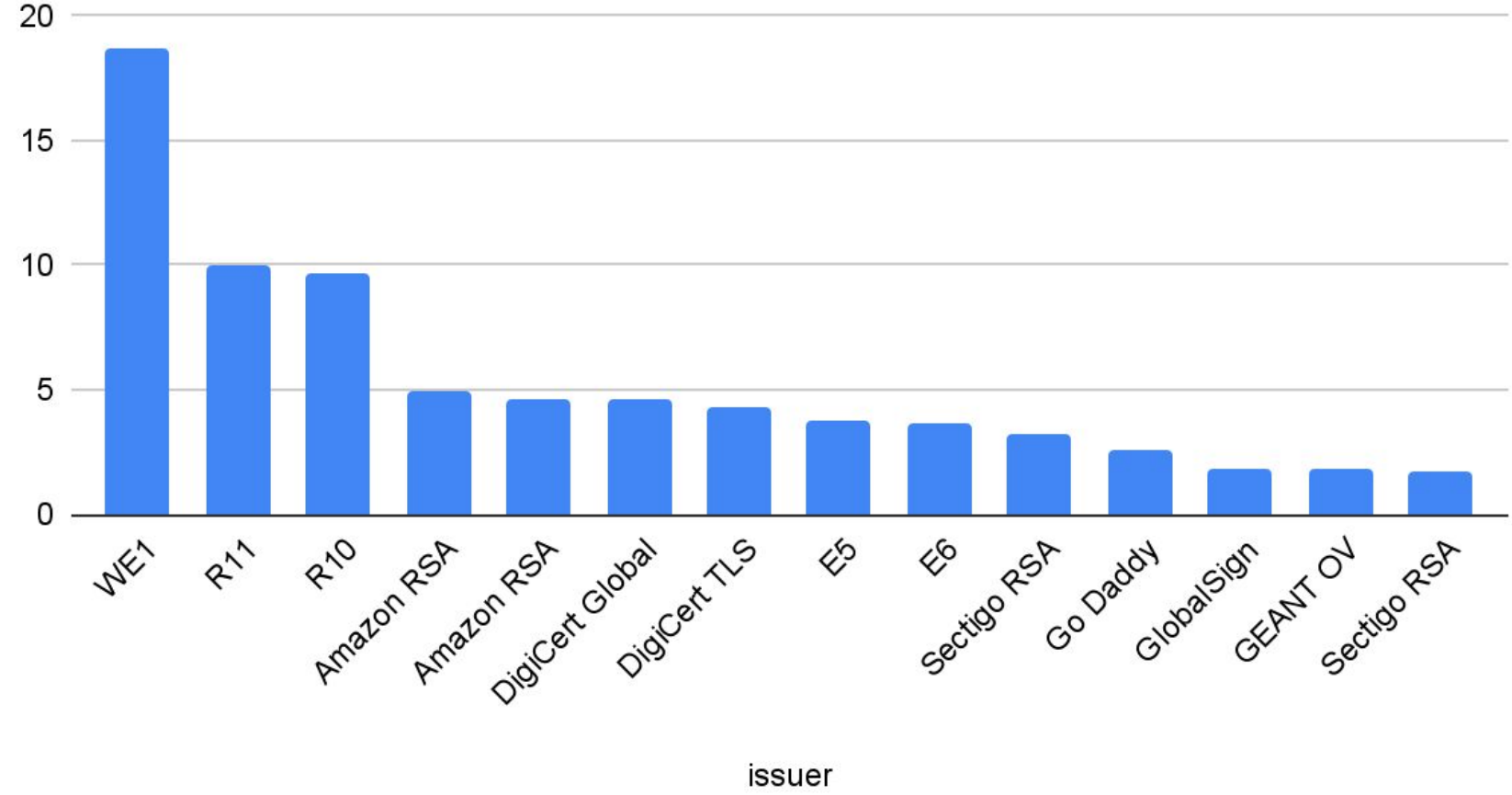
用語補足:「R11」「R10」はLet's Encryptなどの内部管理番号を指す場合あり。詳細は解析対象ログに依存。

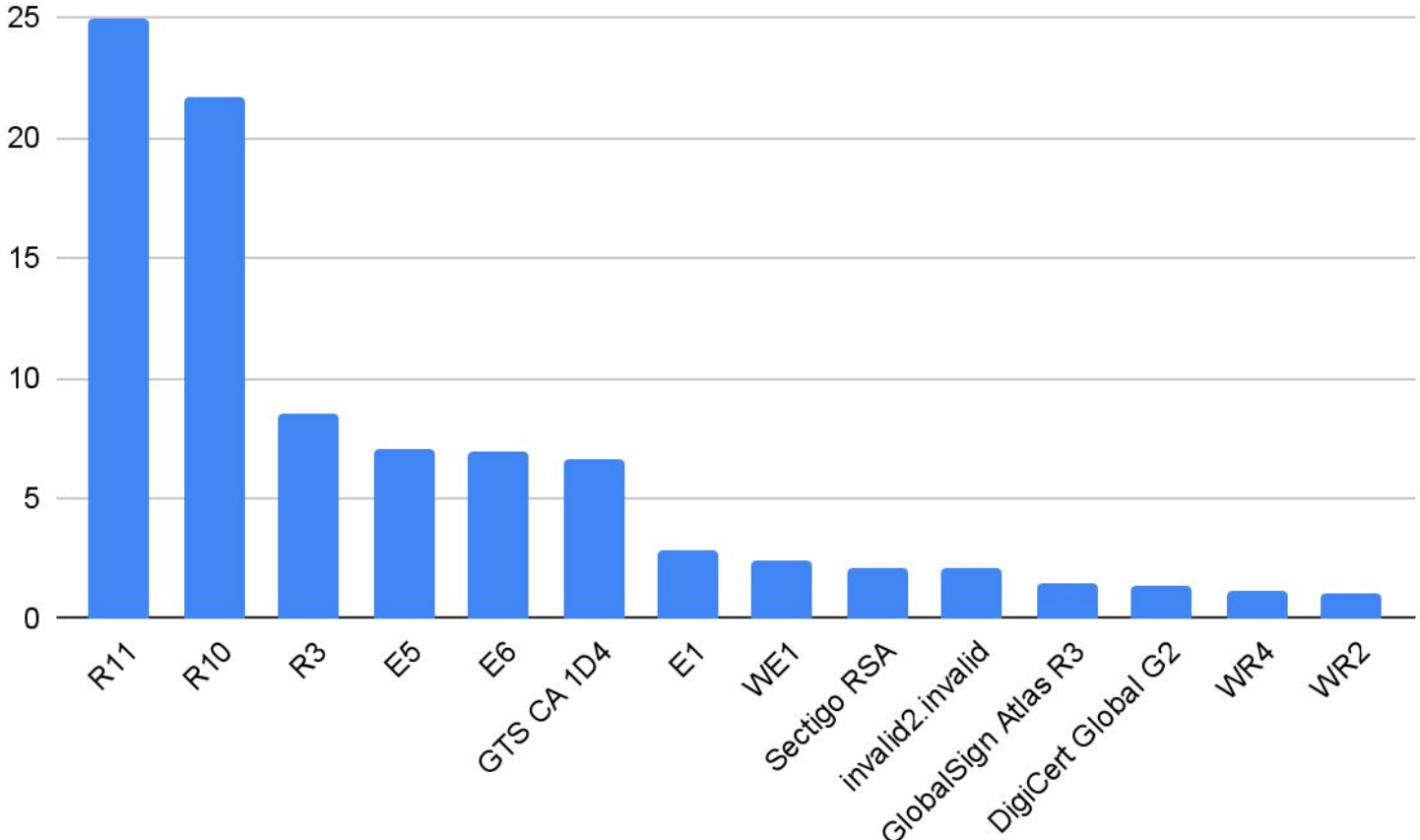
website_count、website_percentage、normal_count、normal_percentage、total_count



/00_12_certificate_analysis_tool_02.ipynb

count と percentage




25
20
15
10
5
0
R11
R10
R3
E5
E6
GTS CA 1D4
E1
WE1
Sectigo RSA
invalid2.invalid
GlobalSign Atlas R3
DigiCert Global G2
WR4
WR2

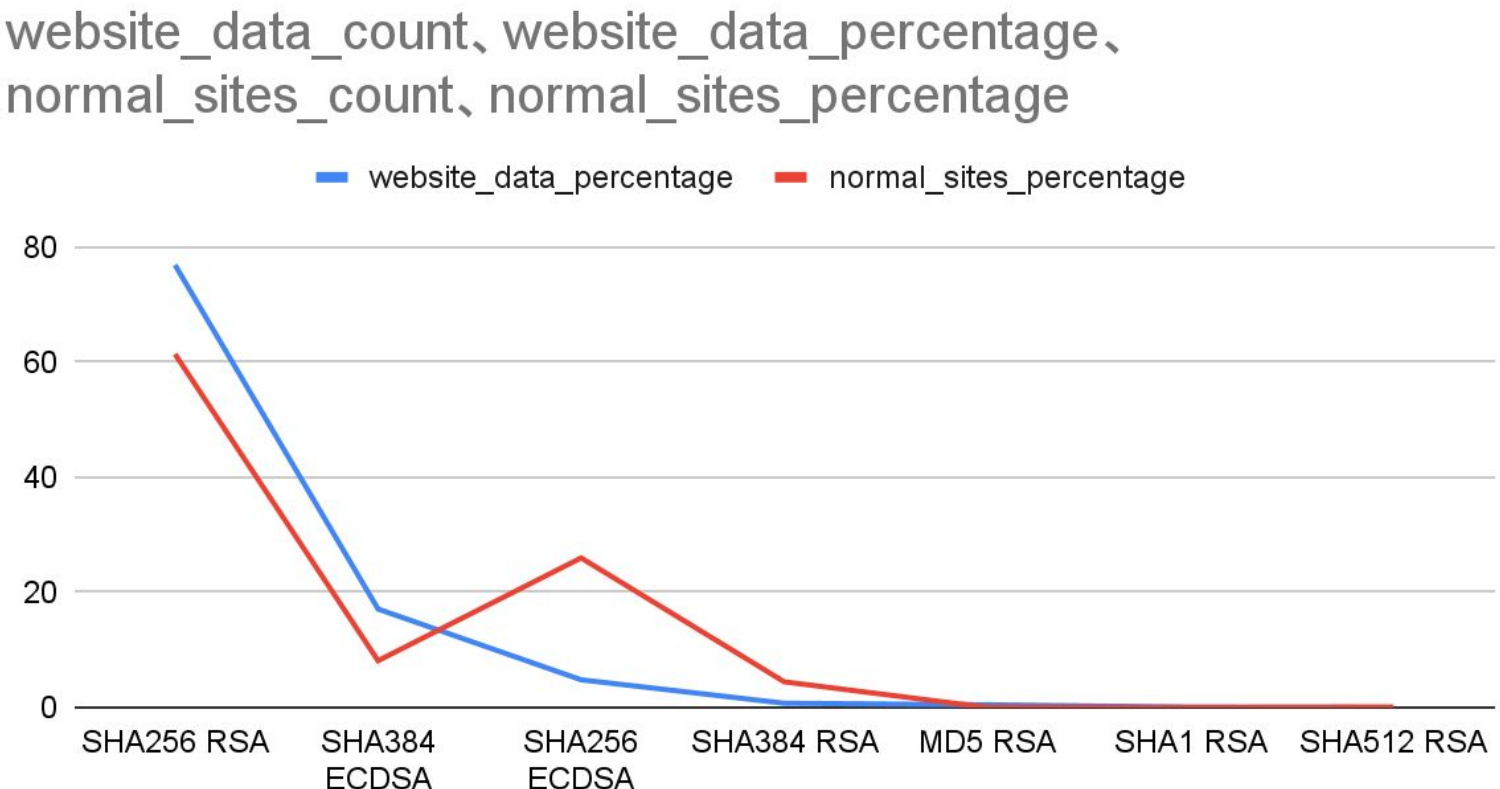
サイト種別ごとの署名アルゴリズム比較

フィッシングサイトではSHA384 ECDSAの利用率が高く、MD5・SHA1といった古いアルゴリズムが少数ながら残存。

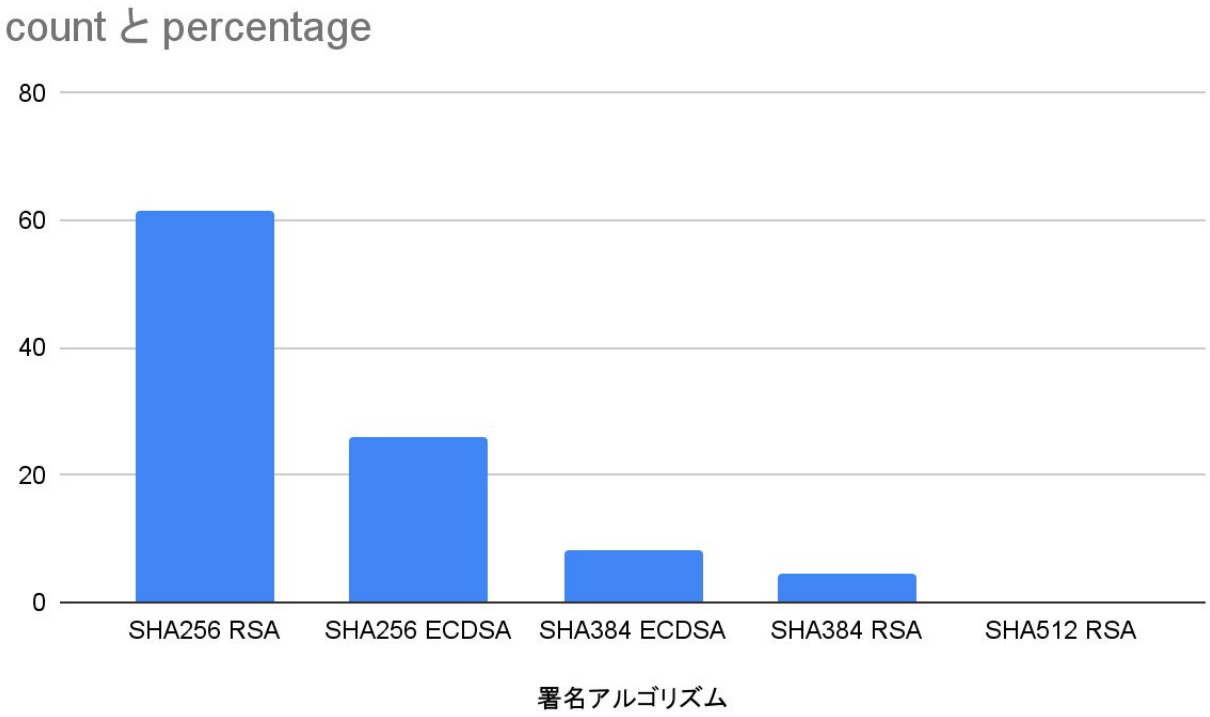
正常サイトではSHA256 ECDSAの比率が高め。

レガシーアルゴリズムの使用はフィッシングを疑う材料になり得るが**大多数は最新アルゴリズム**を利用しているため、これも単独指標での判定は難しい。

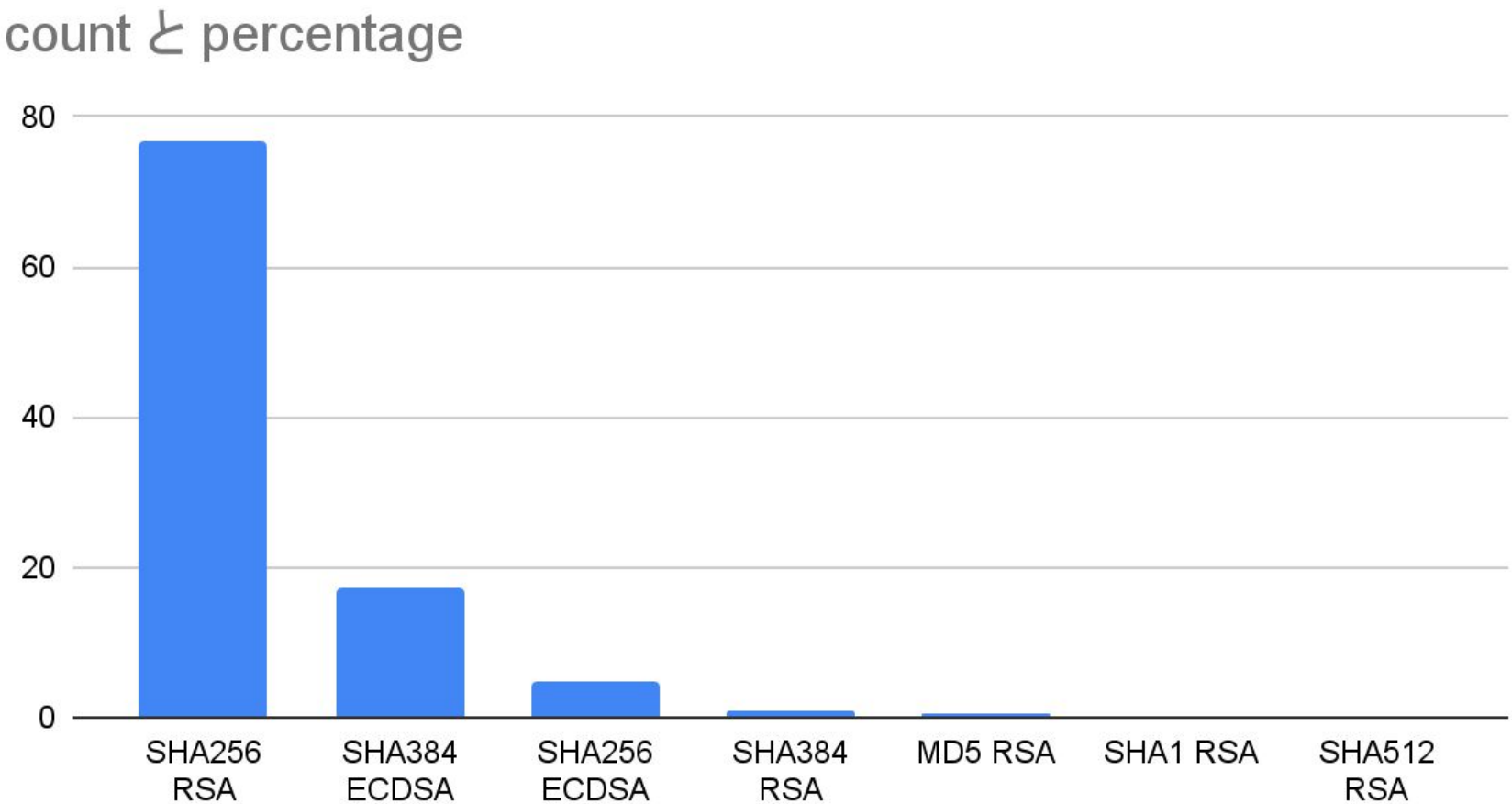
サイト種別ごとの署名アルゴリズム比較



一般のサイト


count と percentage
80
60
40
20
0
SHA256 RSA
SHA256 ECDSA
SHA384 ECDSA
SHA384 RSA
SHA512 RSA
署名アルゴリズム

フィンガープリント


count と percentage
80
60
40
20
0
SHA256 RSA
SHA384 ECDSA
SHA256 ECDSA
SHA384 RSA
MD5 RSA
SHA1 RSA
SHA512 RSA

4. 証明書の有効期間分布

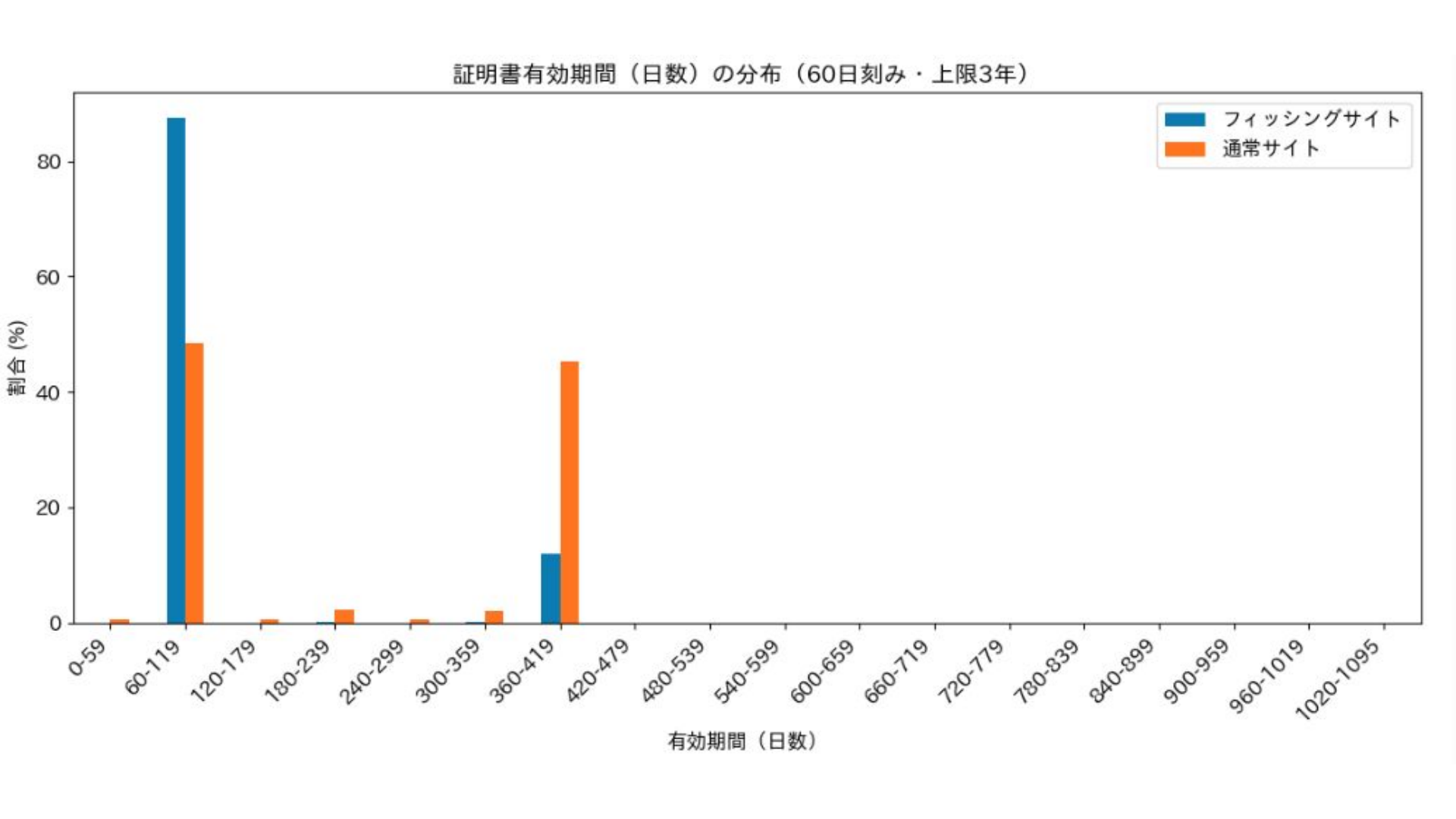
分析結果

- **フィッシングサイト**: 90日以下（短期証明書）が26.0%
- **正常サイト**: 長期（731日以上）が51.5%
- 短期証明書はフィッシングサイトが約2.3倍多い

検知指標になるか？

- **90日以下の短期証明書** はリスク指標として有効
- ただし、Let's Encrypt等の無料CAが短期証明書を発行する場合もあり、**他指標との併用が前提**

/00_12_certificate_analysis_tool_02
.ipynb


証明書有効期間（日数）の分布（60日刻み・上限3年）
フィッシングサイト
通常サイト
割合 (%)
0
20
40
60
80
0-59
60-119
120-179
180-239
240-299
300-359
360-419
420-479
480-539
540-599
600-659
660-719
720-779
780-839
840-899
900-959
960-1019
1020-1095
有効期間（日数）

5. マルチドメイン証明書の特性

分析結果

- **フィッシングサイト**：1ドメイン・2ドメイン・50超の三極化。特に51-100を含むケースが20.7%
- **正常サイト**：2ドメインが52.2%で最大、平均ドメイン数も少なめ（12.09）
- フィッシングサイトは51-100ドメイン証明書が20.7%と突出

統計データ

- **フィッシングサイト**：マルチドメイン率74.2%、平均30.11ドメイン
- **正常サイト**：マルチドメイン率84.6%、平均12.09ドメイン

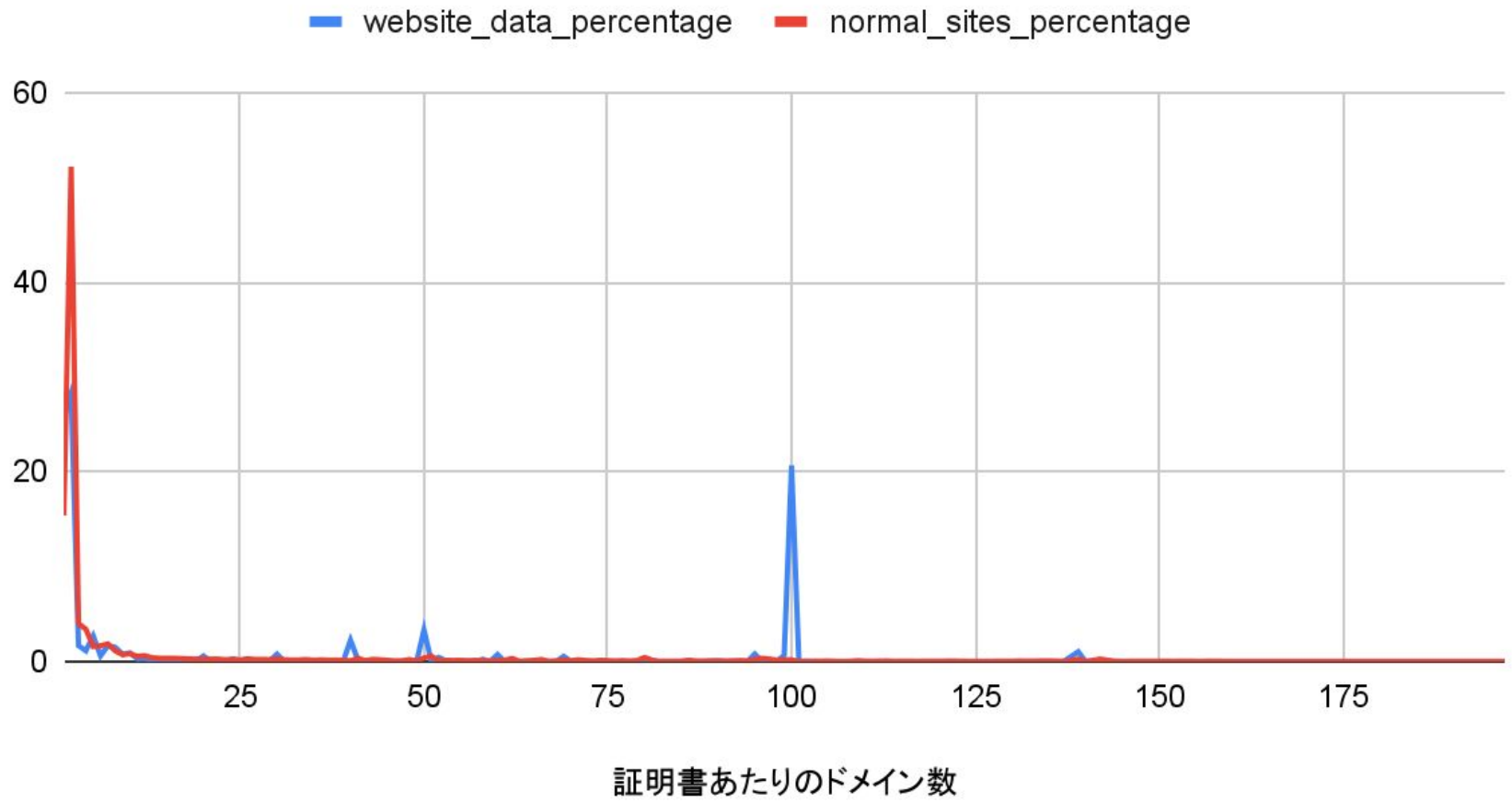
検知指標になるか？

00_11_certificate_analysis_tool-01-Copy1.ipynb

- **大量のドメインを一括管理**　する証明書がフィッシングに多い。
- 一方で、正常サイトも複数サブドメイン運用が一般化しているため、 **極端にドメイン数が多いかどうか**　を基準化すると良い。

ドメイン数の分布

表示の都合上、200で区切っています



6. SSL/TLS実装特性

分析結果

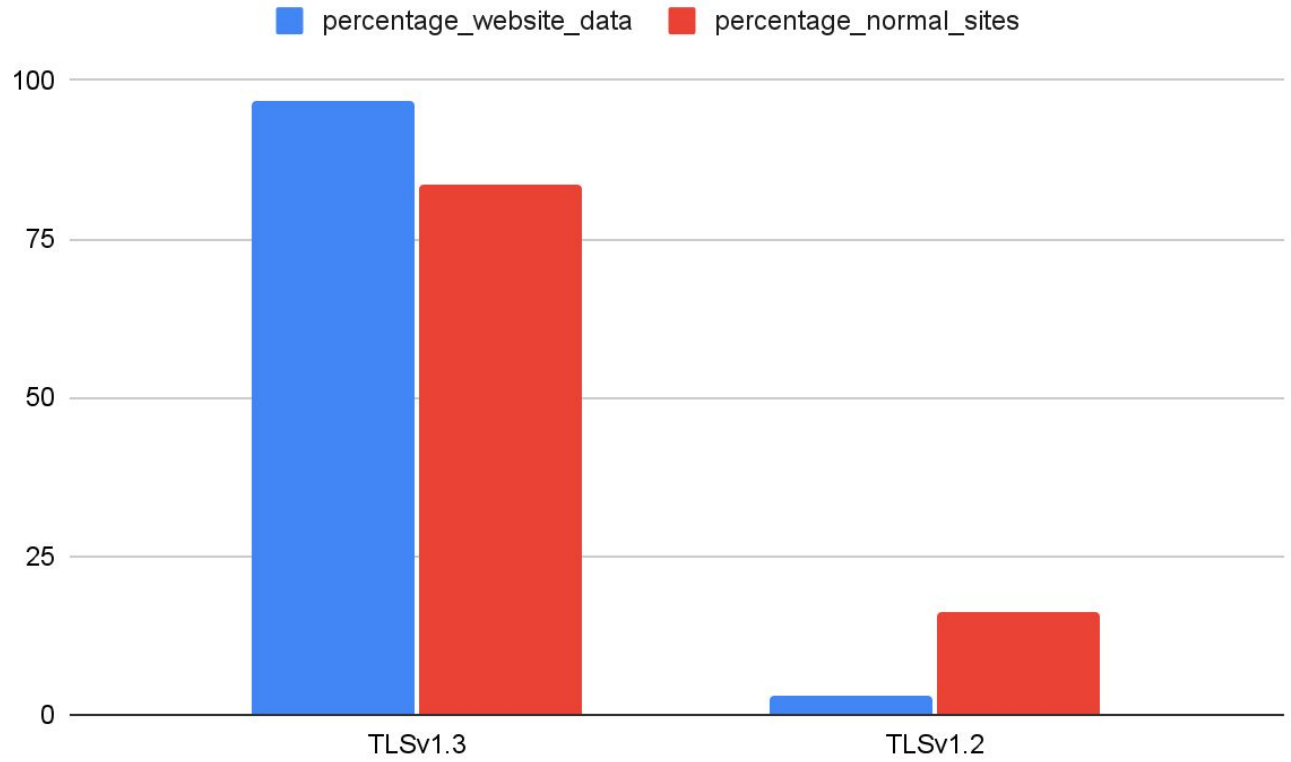
- フィッシングサイト: 96.8%がTLSv1.3
- 正常サイト: 83.7%がTLSv1.3

まとめ/考察

- **最新プロトコル(TLSv1.3)利用率が非常に高い**ため、ここだけでは差異を見つけにくい。
- レガシーTLS（v1.0, v1.1）利用があれば疑わしいが、実際にはv1.3が大多数。

/00-10_certificate_Chain_analysis.ipynb

6. SSL/TLS实装特性



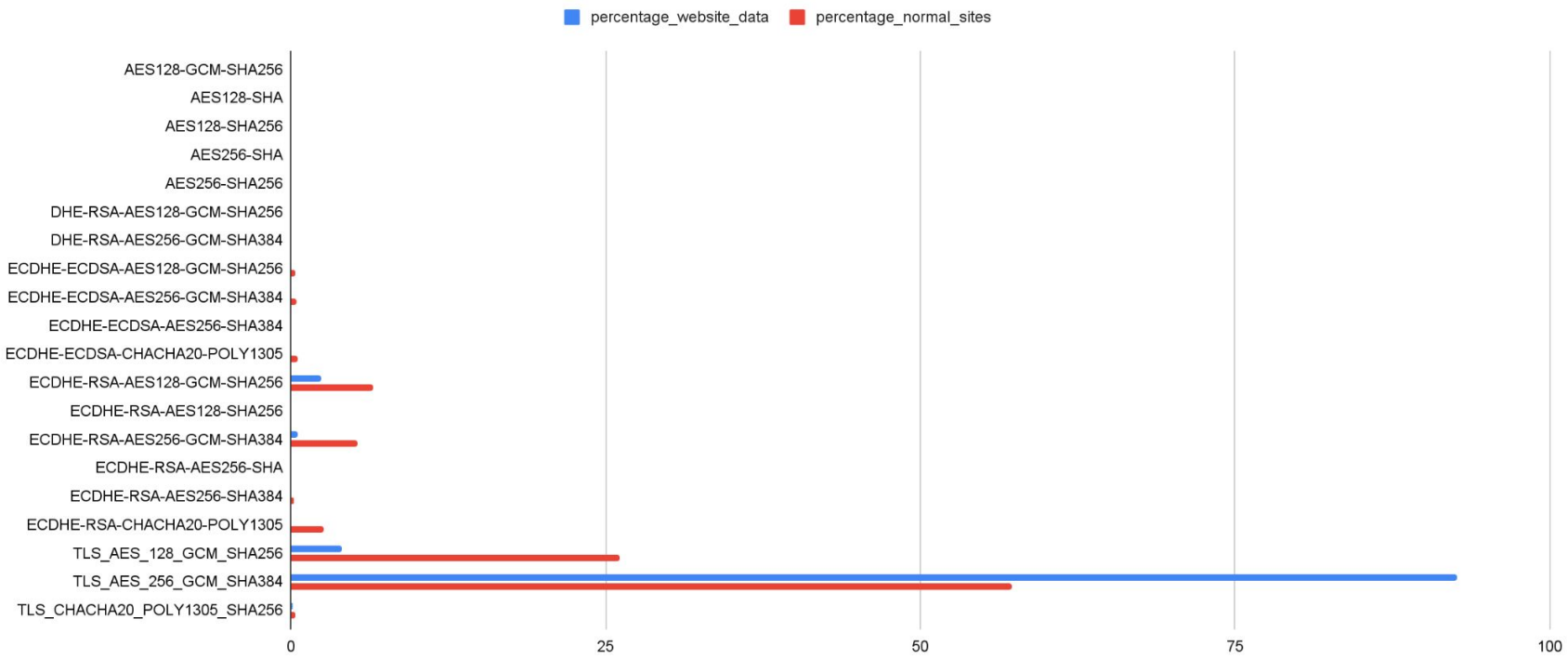
6. SSL/TLS実装特性

- フィッシングサイト / 正常サイトいずれも `TLS_AES_256_GCM_SHA384` が多い。
- 一部で `ECDHE-RSA-AES128-GCM-SHA256` などが使われている。
- レガシーやMD5ベースはごく少数。

まとめ/考察

- 暗号スイートも最新が主流となっており、**暗号スイートのみでの大きな差別化は難しい** 。
- ただし、**極端に旧式** や**不自然なスイート**の利用は注目すべき指標。

暗号スイート分布


percentage_website_data
percentage_normal_sites
AES128-GCM-SHA256
AES128-SHA
AES128-SHA256
AES256-SHA
AES256-SHA256
DHE-RSA-AES128-GCM-SHA256
DHE-RSA-AES256-GCM-SHA384
ECDHE-ECDSA-AES128-GCM-SHA256
ECDHE-ECDSA-AES256-GCM-SHA384
ECDHE-ECDSA-AES256-SHA384
ECDHE-ECDSA-CHACHA20-POLY1305
ECDHE-RSA-AES128-GCM-SHA256
ECDHE-RSA-AES128-SHA256
ECDHE-RSA-AES256-GCM-SHA384
ECDHE-RSA-AES256-SHA
ECDHE-RSA-AES256-SHA384
ECDHE-RSA-CHACHA20-POLY1305
TLS_AES_128_GCM_SHA256
TLS_AES_256_GCM_SHA384
TLS_CHACHA20_POLY1305_SHA256
0
25
50
75
100

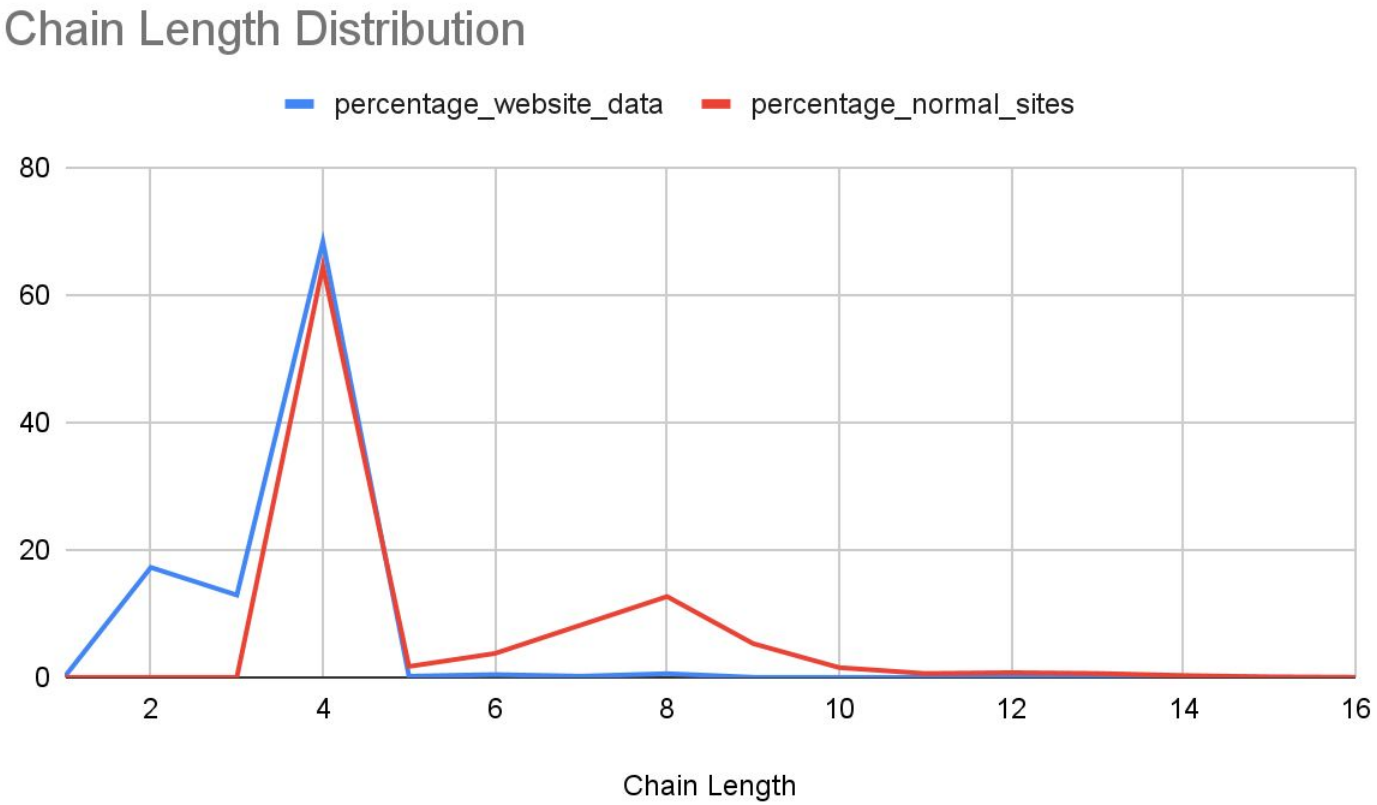
7. 証明書チェーンの特性

- フィッシングサイト: チェーン長が4のケースが68.3%で最多
- 正常サイト: チェーン長4が64.5%と同様に多い
- フィッシングサイトにはチェーン長が1（セルフ署名？）や10以上と極端な例も見られる

まとめ/考察

- 一般的なパターン（チェーン長4前後）は正常/フィッシングどちらも多い。
- **チェーンが異常に短い・長い** のは怪しいケースかもしれないが、全体数としては少ないため見逃しが起こりうる。

7. 証明書チェーンの特性



8. ドメイン構造の特徴

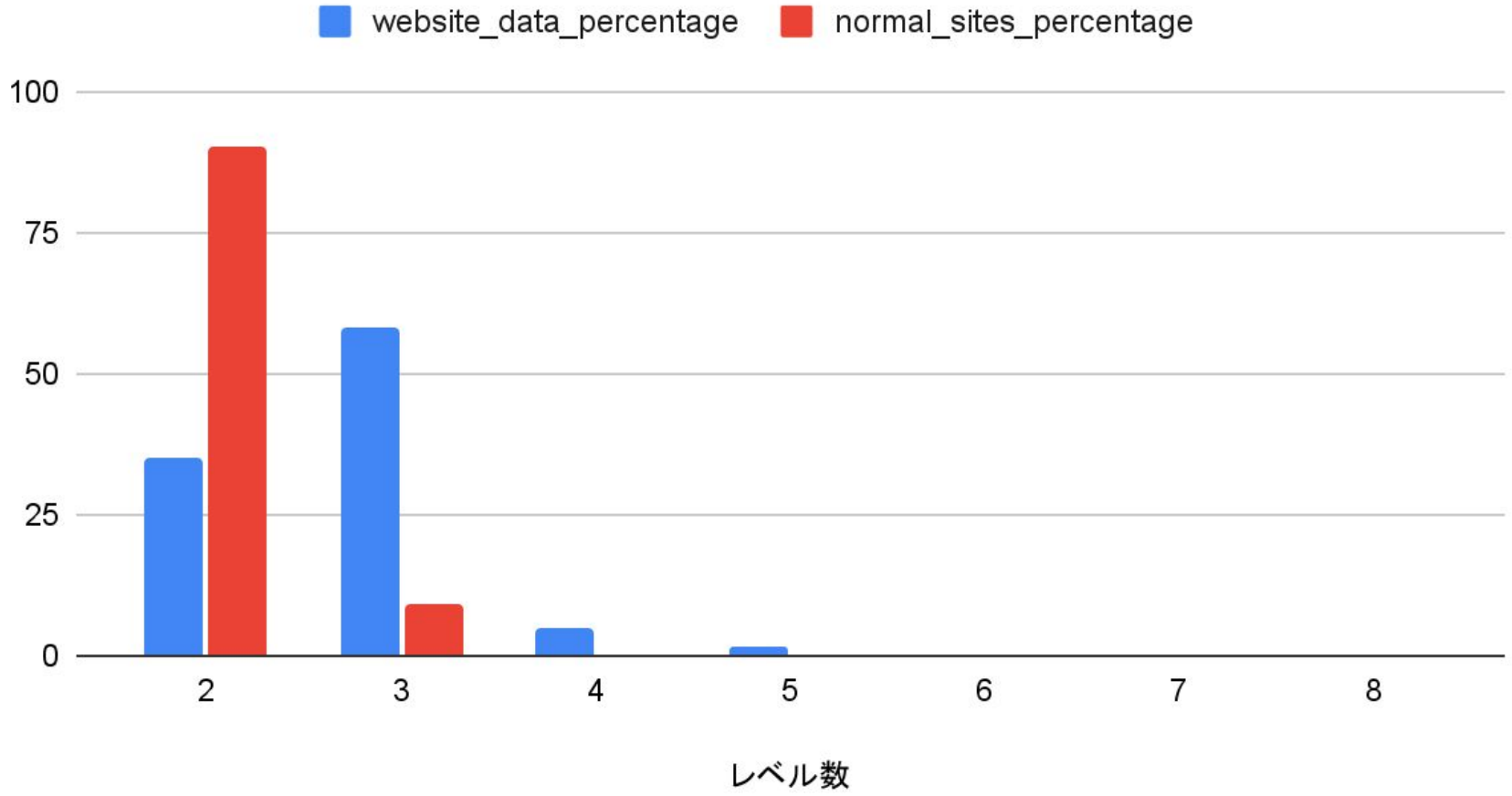
分析結果

- **フィッシングサイト**は多層サブドメインを使用する傾向
- **正常サイト**はシンプルな2レベルドメインが主流
- 4レベル以上の複雑なドメインはフィッシングサイトに多い

意義

- ドメイン構造の複雑さはフィッシング検出の有効な指標
- **ドメイン階層**が多く、www.www.www.○○ のような形態はフィッシングで多用。
- この特徴は比較的優位性があり、**リアルタイム判定**で利用価値が高い。

Domain Level Comparison



9. 時系列パターン

時系列的特徴

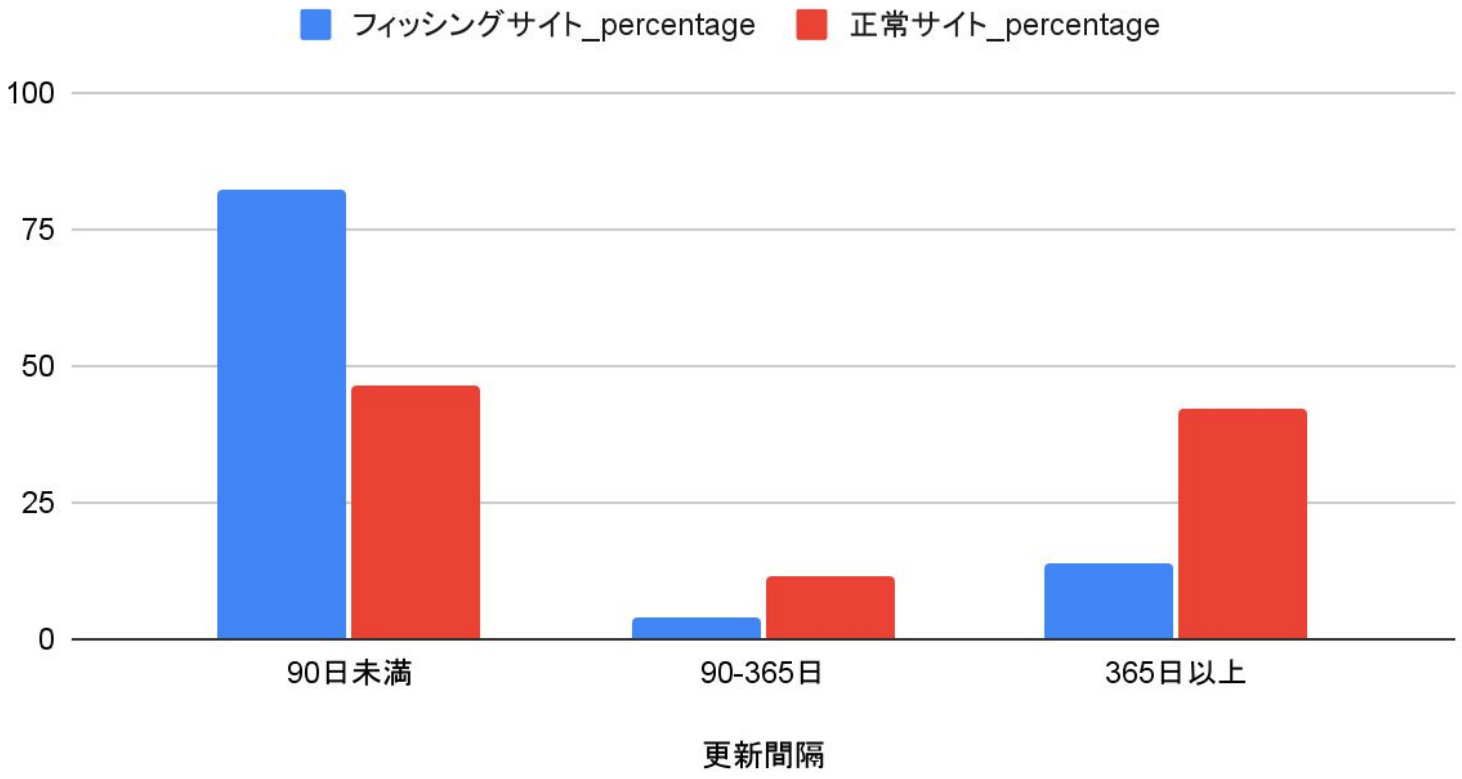
- フィッシングサイト: 短期的に使用、不規則な更新が多い、週末発行が多い
- 正常サイト: 長期的な証明書使用、定期的な更新、平日発行が多い
- 90日未満の更新間隔がフィッシングサイトで82.08%

考察

- **週末や夜間の発行**などの時系列的偏りは、リアルタイム監視において注目すべき要素。
- ただしグローバル運用を考慮すると、時差の問題があり単純な曜日分析は慎重に行う必要がある。

/00_12_certificate_analysis_tool_02.ipynb

時系列パターン


フィッシングサイト_percentage
正常サイト_percentage
100
75
50
25
0
90日未満
90-365日
365日以上
更新間隔

次のステップに向けて

特徴量設計とスコアリングの試作

- これまでの分析で示唆された指標（ DV/OV/EV、レジストラ、署名アルゴリズム、有効期限、ドメイン階層、時系列など）を**一貫したフォーマット** で抽出し、スコアリングモデルをプロトタイプとして実装する。
- 単純なロジスティック回帰や決定木などでまずベースラインを確立し、次に BERT等の高度なモデルに発展させる。

多層サブドメイン解析　+ドメイン文字列 NLP

- ドメインの深さや繰り返し構造を簡易的なルールベースで判定し、**“異常度”** としてスコア化。
- さらにBERTを使い、文字列のタイポスクワッティング（ブランド名との類似）を評価する仕組みを加える。

次のステップに向けて

オンライン学習・リアルタイム実装テスト

- Certstreamからのデータを**リアルタイムに取り込み** 、上記のスコアリングを実行 → 一定スコア以上を“疑わしい”としてログ保存や警告を出すプロトタイプを構築。
- 誤検知と見逃し（False Positive/Negative）を記録し、モデル再学習に活かす。

WHOIS・DNS情報との突合

- 可能であればWHOISやDNSレコードの取得を自動化し、**登録日 /登録者 /リソース IP** を加味してさらに精度を高める。
- 特にWHOISの登録日が近いドメインかつ短期証明書はリスク度を高めるなど、追加ルール化が望ましい。

実運用システムとの連携準備

- 短期的には研究用途の APIやダッシュボードを整備し、中長期的には運用インフラ（クラウドやオンプレ）

次のステップに向けて

実運用システムとの連携準備

- 短期的には研究用途の APIやダッシュボードを整備し、中長期的には運用インフラ（クラウドやオンプレ）で**24/7稼働**を視野に入れる。
- 負荷試験や可用性確保のためのアーキテクチャ設計にも着手する。

まとめ

今回の分析から、フィッシングサイト特有の **DV証明書比率の高さ、特定レジストラへの偏り、短期運用、大量サブドメイン構造、週末に集中する発行タイミング** など、多角的な傾向が確認されました。しかし、正常サイトでも似た特徴を持つ例（DV利用、Let's Encrypt短期証明書など）が増えているため、**複数の指標を組み合わせた総合スコアリング** が鍵となる。

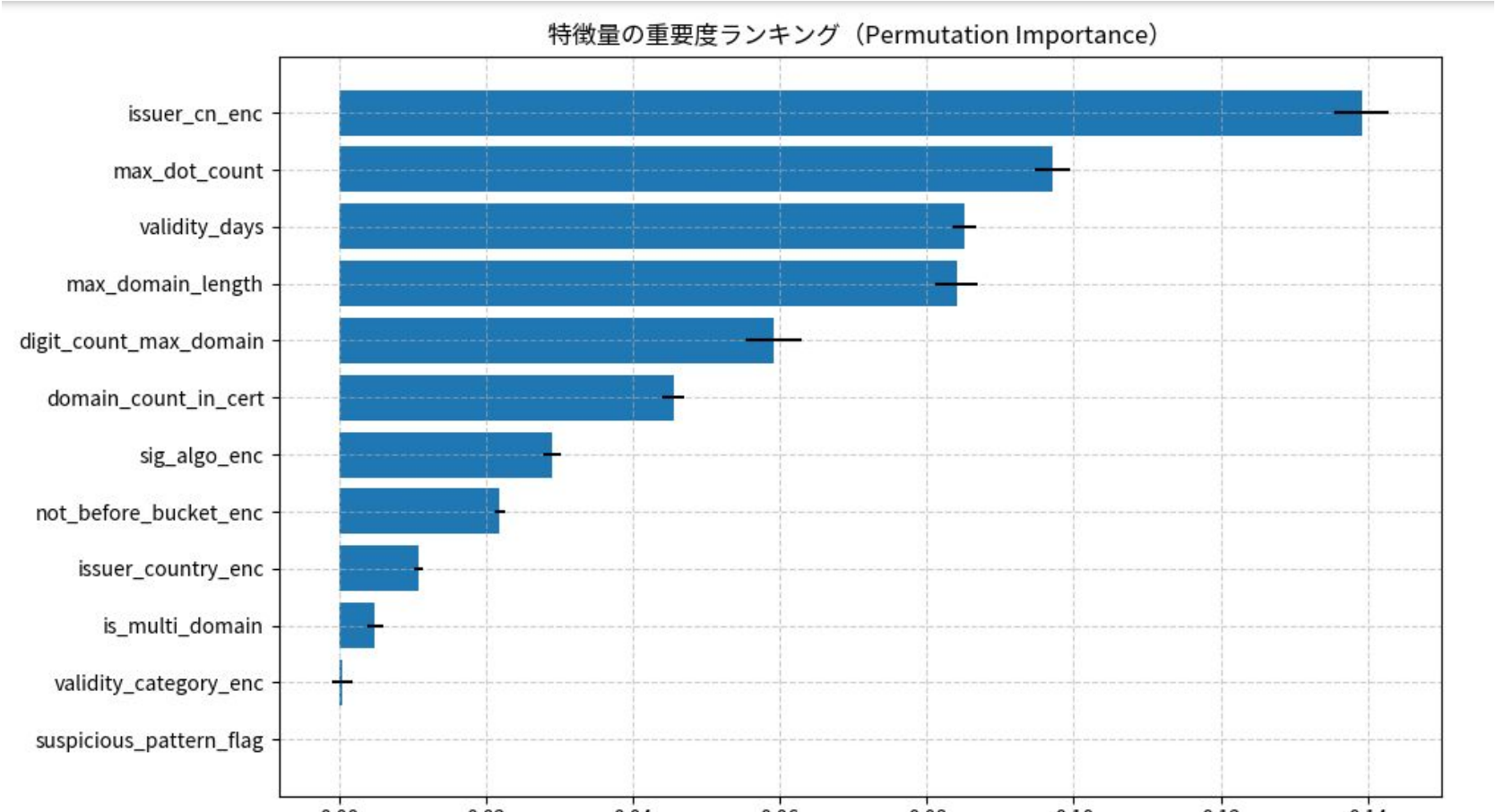
次のステップとしては、これらの **特徴量を活用したモデル開発とリアルタイム実装** に進むことが最も重要です。特に **オンライン学習** や **BERTを活用したドメイン文字列解析** を組み込むことで、現行のフィッシングサイト検出手法を強化することを目標とします。

以下、戸田さんにいただいたアドバイスをもとに実装してみました

次：決定木等 -でデータセットのデータを数値化して、分析　/特徴量を抽出する
–証明書に含まれるデータからやってみる等

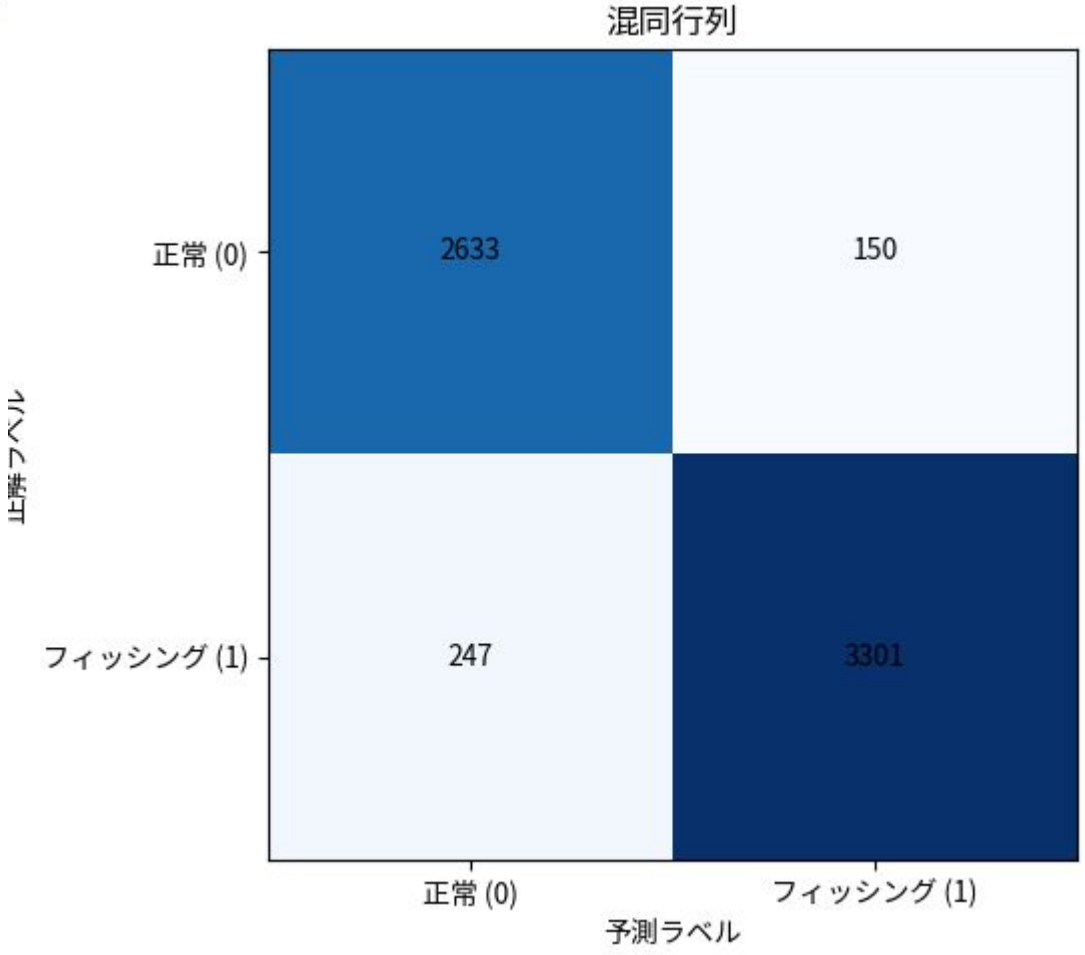
抽出される主な特徴量

SSL証明書の情報を用いて特徴量を抽出し、機械学習モデル（ RandomForest）と**Permutation Importance**を使って、各特徴量の重要度を評価・可視化した。



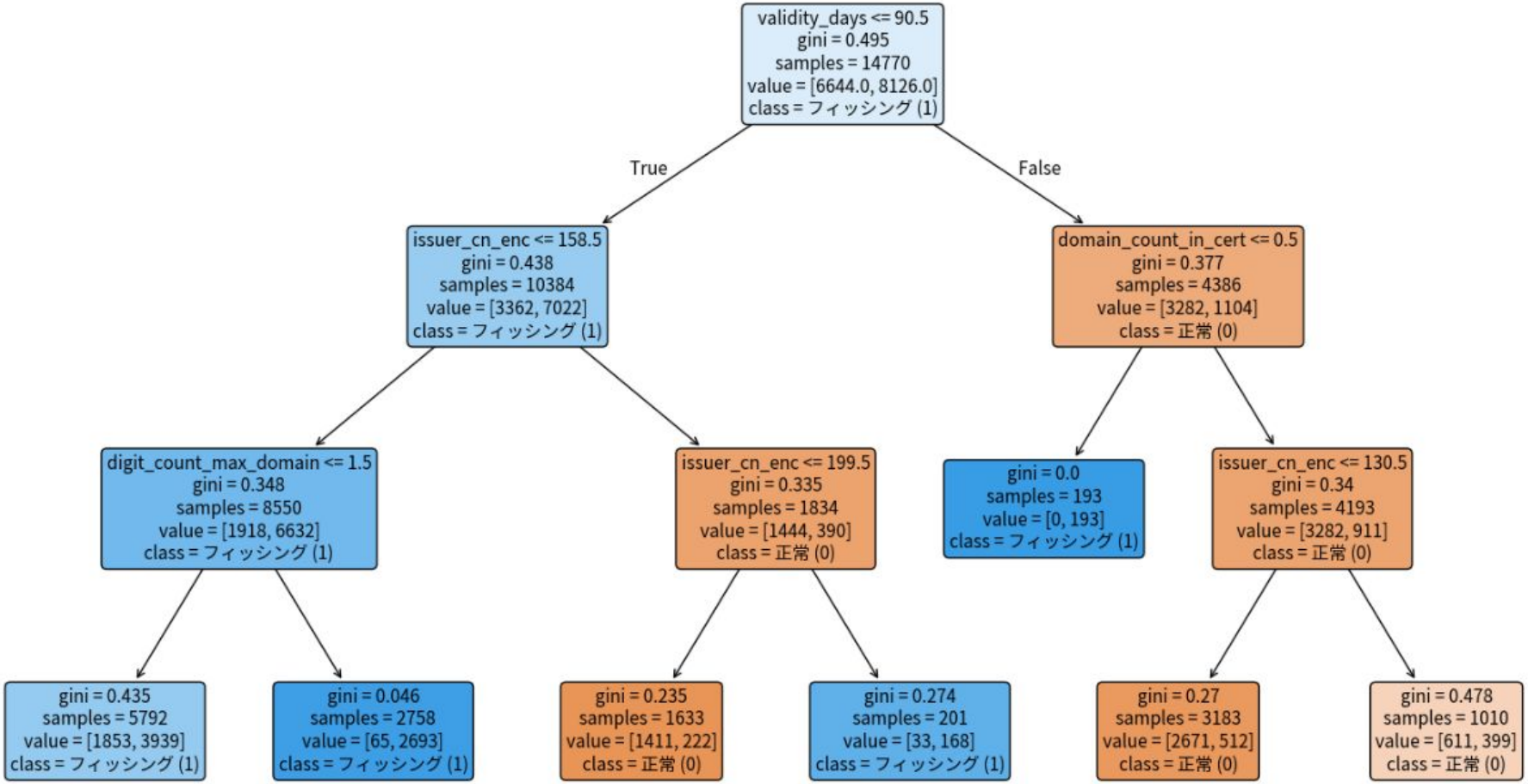
```
=== 選抜モデルの評価結果 ===
              precision    recall  f1-score   s

           0      0.914     0.946     0.930
           1      0.957     0.930     0.943

    accuracy                         0.937
   macro avg      0.935     0.938     0.937
weighted avg      0.938     0.937     0.937

Accuracy: 0.9373
```

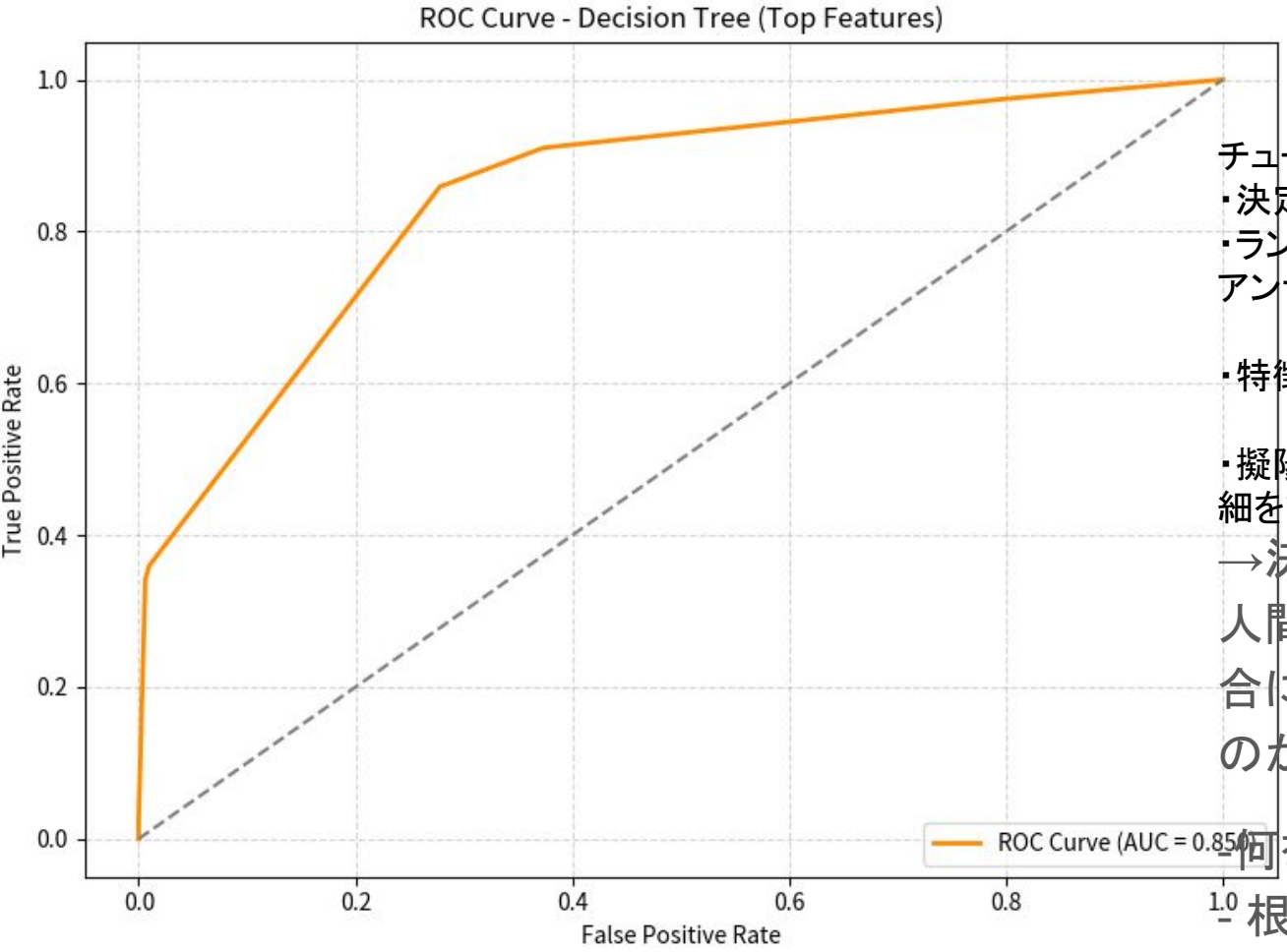
1. ランダムフォレストによるシャッフル重要度（permutation importance）を用いて、予測に有効な特徴量を選抜
2. 1.で選抜した特徴量の上位特徴量のみでモデルを再学習・評価することで、特徴量の選別によるモデル精度や効率性への影響を検証



全体のデータを「学習用（70%）」と「検証用（30%）」に分けた

[シャッフル重要度に基づく特徴選択と分類モデルの再構築]


validity_days <= 90.5
gini = 0.495
samples = 14770
value = [6644.0, 8126.0]
class = フィッシング (1)
True
False
issuer_cn_enc <= 158.5
gini = 0.438
samples = 10384
value = [3362, 7022]
class = フィッシング (1)
domain_count_in_cert <= 0.5
gini = 0.377
samples = 4386
value = [3282, 1104]
class = 正常 (0)
digit_count_max_domain <= 1.5
gini = 0.348
samples = 8550
value = [1918, 6632]
class = フィッシング (1)
issuer_cn_enc <= 199.5
gini = 0.335
samples = 1834
value = [1444, 390]
class = 正常 (0)
gini = 0.0
samples = 193
value = [0, 193]
class = フィッシング (1)
issuer_cn_enc <= 130.5
gini = 0.34
samples = 4193
value = [3282, 911]
class = 正常 (0)
gini = 0.435
samples = 5792
value = [1853, 3939]
class = フィッシング (1)
gini = 0.046
samples = 2758
value = [65, 2693]
class = フィッシング (1)
gini = 0.235
samples = 1633
value = [1411, 222]
class = 正常 (0)
gini = 0.274
samples = 201
value = [33, 168]
class = フィッシング (1)
gini = 0.27
samples = 3183
value = [2671, 512]
class = 正常 (0)
gini = 0.478
samples = 1010
value = [611, 399]
class = 正常 (0)



チューニングの余地はある
・決定木の深さを少しだけ増やす
・ランダムフォレストやXGBoostなどアンサンブル手法を試す

・特徴量の数（TOP_N）を調整する

・擬陽性、偽陰性となったデータの詳細を調べる
→決定木で、はずれたものを人間があらためて判定した場合に、なにが理由ではずれたのか文章化すること

- 何を参照したか
- 根拠となったロジックとかをすべて書き出すこと。

既存研究との差分は必要である
100%に近い値にしたい。ひとつひとつ、見て擬陽性などの理由をみつける。additionalなworkaroundを探すときにllmを用いて分析しても良いかもしれない

判定をする場合にドメイン名をみたり、機械学習モデルでフィッシングサイトを判定するときに60%くらいのときにAIをつかってみる。
LLMエージェントのほうが取り組みやすい。
confidenceが低いサンプルについては、別途AIエージェントが解析する方法
判断の確度を定量化できる手法だと、やりやすいかも。
—--
・分類器を、AIにつくってもらう->入力と出力を与えてAIにお願いする
・判定の確度も教えてもらえるような仕組みをAIにお願いする
—-
新規性は、それでも見落とす部分をどうやって拾っていくかどうか。

Claudeに書くときに、モデルとしては決定木ベース、*boost、ニューラルネットワーク、教師付きxの中から選んで。

判定の角度(confidence)を定量的に表現できるモデル。出力を確率分布をモデル化するものである必要がある。ニューラルネットワークとか、リグレッションベースの決定木とか。

■機械学習モデル単体ではできなかったことを、組み合わせで実現できるようにする
究極的には、メタラーニング？

機械学習モデルの改良と AIエージェント活用に関する提案内容のまとめ

機械学習モデルのチューニング提案

- 決定木の深さを微調整して精度向上を図る
- ランダムフォレストや XGBoostなどのアンサンブル手法を試す
- 使用する特徴量の数（ TOP_N）を最適化する
- 誤判定（擬陽性・偽陰性）となったデータの詳細分析を行う

判定精度向上のアプローチ

- 決定木で誤判定したケースを人間が再検証し、誤判定の理由を文書化する
- 判定の根拠となったロジックをすべて記録し、「ツールボックス」として活用
- 信頼度（confidence）が低いサンプルは、 AIエージェントによる別途解析を実施
- AIエージェントの判断プロセスを説明可能にする

複合アプローチの提案

- 機械学習モデル（決定木ベース、ブースティング手法、ニューラルネットワークなど）と　AIエージェントの組み合わせ
- 判定の確度を定量的に表現できるモデル（確率分布をモデル化できるもの）を選択

やること

いままで提示したプログラムを用いて以下のこと行いたい。

1. 決定木で誤判定したケースを人間が再検証し、誤判定の理由を文書化する。この際、出力したデータに、特徴量の元のデータもいれるようにプログラムを改良してください。

2. 判定の根拠となったロジックをすべて記録し、「ツールボックス」として活用

3. 信頼度（confidence）が低いサンプルは、AIエージェントによる別途解析を実施

4. AIエージェントの判断プロセスを説明可能にする

提示したプログラムを改良してください。

過学習（オーバーフィッティング）を調べるには、基本的に**学習データ上の性能**と**テスト（検証）データ上の性能**を比較します。差が大きいほど過学習の疑いが強くなります。以下のように新しいセルを追加してみてください。