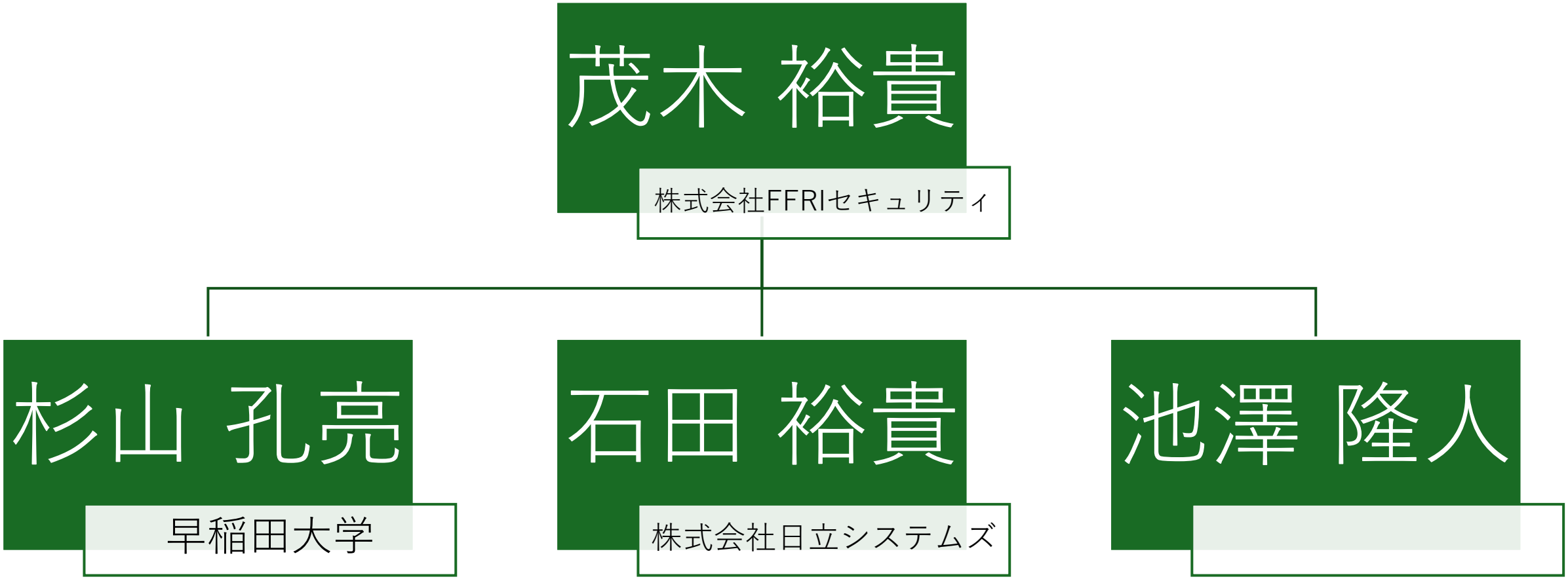
ANTI-MALWARE AND ANTI-CYBERATTACKS ENGINEERING WORKSHOP

MWS Cup 2024
マルウェア分類問題

解説資料

作問チーム紹介



作問方針

Kaggleを使用する

例年と同様のデータ形式を用いる

基本的な知識を問うような問題にする

課題内容

- FFRI Datasetと同形式のデータから特徴量を作成し、ラベルの有無を予測

- Fexrdについて
  - 概要：FFRI Datasetと同形式のデータから特徴量を生成するライブラリ
  - URL：https://github.com/FFRI/FEXRD

データ形式

- FFRI Datasetのデータ形式とほぼ同様
  - dateやハッシュ値が存在しないだけ
  - ラベルデータ（訓練データのみ）
    - MalwareBazzarのDr.Web vxCubeによる分析結果

- 訓練用データ：train.jsonl
- テスト用データ：test.jsonl

ラベルデータに関する注意

- 今回のラベルデータにはMalware BazaarのvxCube解析データをそのまま利用した
  - ラベルデータとしては適切ではない。。

- 本Cupにおけるマルウェア分類問題は、分類に関する知識・技術を問う問題であるため、ラベルの正しさについては考慮していない

- したがって、本Cupの手法を元に論文等に応用する場合は十分に考慮する必要がある

ヒント・模範解答に対する注意

- 他の問題と違ってDSコンペに「解答」はない

- 模範解答も普通はない（KaggleにせよProbSpaceにせよ普通ホストは出さない）
- 今回のヒントや模範解答はあくまで1つのやり方を提示しているに過ぎない
- サンプルやヒントは一切手が付けられないことのないように出しているだけで、この方法に誘導したいわけでは一切ないし、模範解答のやり方が「正しい」訳ではない

- 参加者にはより良い作問のためWrite-upにご協力いただきたいです （本コンペのDiscussionやNotebookの公開）

解説1(PB: 0.567)

- サンプル解答
  - サンプル解答はRandomForestを使用して特定のラベルの有無を予測する

```python
# First Submission
# 単純にRandomForestを使ってみる
from sklearn.ensemble import RandomForestClassifier
X_train = df_train.drop(["label", "ID"], axis=1).to_numpy()
# あくまでEVASIONシグネチャのみを予測して答えるため、0番目のシグネチャの予測を答える
y_train = np.array(df_train["label"].tolist())[:, 0]
cv(X_train, y_train, lambda: RandomForestClassifier(random_state=42))
```

Python

```
CV: 0.5700279385740679
```

```python
X_test = df_test.drop(["ID"], axis=1).to_numpy()
df_solution = pd.read_csv(solution_path)
df_pred = make_prediction(X_train, y_train, X_test, lambda: RandomForestClassifier(random_state=42))
compute_score(df_pred, df_solution)
```

Python

```
LB: 0.5171428571428571
PB: 0.5671204987911949
```

解説2(PB: 0.567 -> 0.621)

- データを確認すると、特徴量として使用しているラベルが存在しないデータが存在することがわかる
- そのため、ラベルがないデータをどう扱うのかが重要となる

```python
# Second Submission
# 特徴量の欠損値を0で埋めてみる
# それなりにスコアが向上する
X_train = df_train.drop(["label", "ID"], axis=1).fillna(0).to_numpy()
cv(X_train, y_train, lambda: RandomForestClassifier(random_state=42))
X_test = df_test.drop(["ID"], axis=1).fillna(0).to_numpy()
df_pred = make_prediction(X_train, y_train, X_test, lambda: RandomForestClassifier(random_state=42))
compute_score(df_pred, df_solution)
```

Python

```
CV: 0.6765958870373222
LB: 0.6771428571428572
PB: 0.621325868431098
```

解説3(PB: 0.621 -> 0.712)

- 機械学習には様々なモデルが存在するが、より強力なモデルとしてLightGBMがある

```python
# Third Submission
# RandomForestからLightGBMに変更してみる
# これもかなり精度が向上する
import lightgbm as lgb
cv(X_train, y_train, lambda: lgb.LGBMClassifier(random_state=42))
df_pred = make_prediction(X_train, y_train, X_test, lambda: lgb.LGBMClassifier(random_state=42))
compute_score(df_pred, df_solution)
```

Python

解説4-1(PB: 0.712 -> 0.727)

- 複数シグネチャを使用して分類を行うことで精度向上が望める
- 複数シグネチャを扱う場合はClassifierChainが使える

- ClassifierChainは以下の流れで予測を行うため、予測の順番も重要となる
  1. あるラベルXを予測する
  2. ラベルXの有無を特徴量に追加して、ラベルYを予測する
  3. ラベルX/Yの有無を特徴量に追加して、ラベルZを予測する

解説4-2

- コード

```python
y_train = np.array(df_train["label"].tolist()) # 先頭だけでなく全てのシグネチャを使う
# 順番を決めるためにCVを実行
cv_multi_sig(X_train, y_train, lambda: MultiOutputClassifier(lgb.LGBMClassifier(random_state=42)))
```

- 実行結果
  - CV: [0.73235932 0.8014239 0.85447183 0.77331016 0.82315805 0.85498788]
- 考察
  - 各シグネチャのCVの結果から、ClassifierChainの順番は5=>2=>4=>1=>3=>0にするのが合理的と考えられる
  - なお、並び順は精度が高い順の方が良いが、そもそもシグネチャを選ぶ際には、ただ予測精度が高いシグネチャを選べば良いとも言えないことに注意

解説4-3

- 最終的に、ClassifierChainとLightGBMを組み合わせて以下の形となる

```python
from sklearn.multioutput import ClassifierChain
# 最後のラベルが予測したいラベル
y_train = np.array(df_train["label"].tolist())[:, np.r_[5,2,4,1,3,0]]

cv_multi_sig(X_train, y_train, lambda: ClassifierChain(lgb.LGBMClassifier(random_state=42)))
```

Python

Towards：次回以降（サステナビリティ）

- 持続可能なMWS Cupの実施のために、
  作問に協力していただけるメンバーを募集しています。

2018-201	• Fexrdの出力結果をそのまま使用する • 前回の模範解答がスタート地点
2022	• Stringsのみを用いた • trainデータとtest データの時間をずらした
2023	• マルウェア検知ではなく分類問題とした • 半教師あり設定とし工夫の余地を増やした
2024	• マルウェアのラベル予測の問題とした • 昨年同様に半教師ありとした

来年度はどうなるか未定です！
ぜひ皆様のご協力を、よろしくお願いいたします！！！！！