CalSYS Lab Senior Project

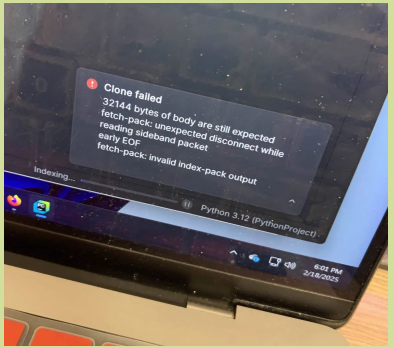
By Sidharth Basam

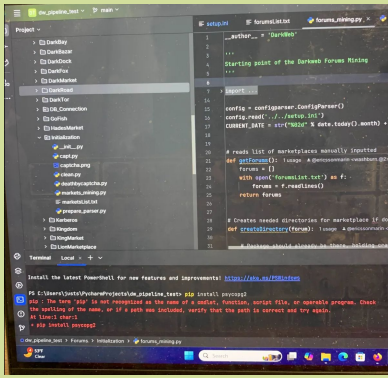
INTRODUCTION

- Environment Setup
- Scraped Pitch and NZ Darknet (New Zealand Darknet)
- Binary Probabilistic Classifier

Environment Setup

- Had to use VMware Fusion instead of Virtualbox since I used a Apple Silicon Mac
- Ran into a few problems with the setup
  - Cloning git error was fixed by restarting VM
  - Certain install commands didn't work because of dependency issues
  - Usually fixed most issues by uninstalling and reinstalling modules

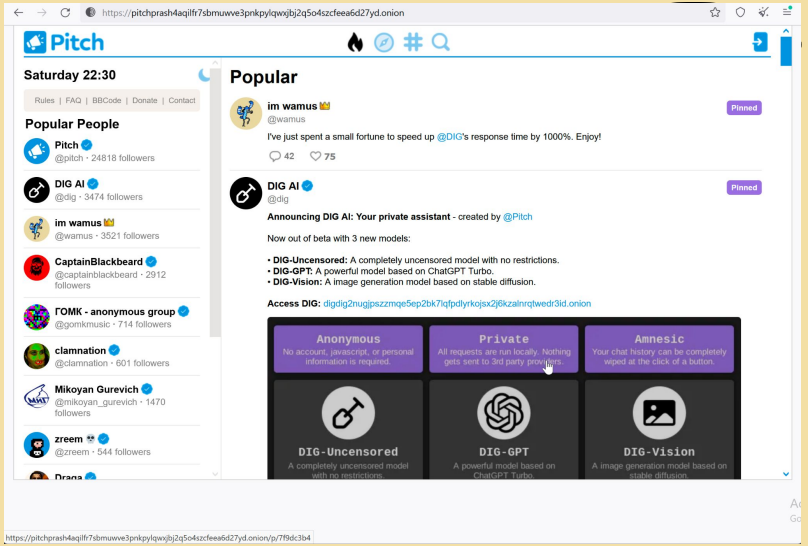


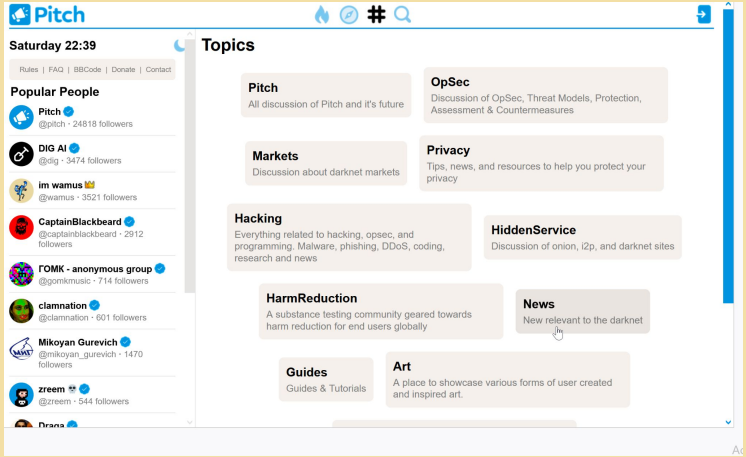


```
Traceback (most recent call last):
  File "C:\Users\justs\PycharmProjects\dw_pipeline_test\Forums\Initialization\forums_mining.py", line 8, in <module>
    from Forums.BestCardingWorld.crawler_selenium import crawler as crawlerBestCardingWorld
  File "C:\Users\justs\PycharmProjects\dw_pipeline_test\Forums\BestCardingWorld\crawler_selenium.py", line 7, in <module>
    from selenium import webdriver
ImportError: cannot import name 'webdriver' from 'selenium' (unknown location)
```

Pitch

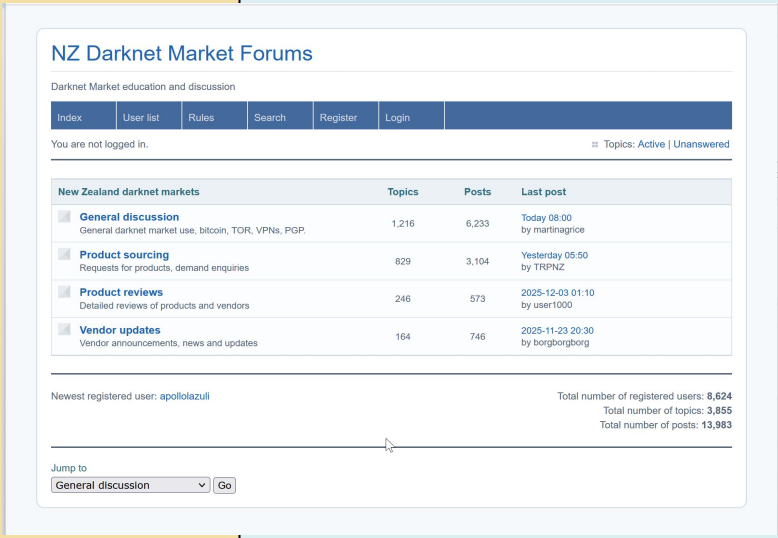
- The entry link into the Pitch forum is the Popular forums tab
- The Pitch forum has many different topics
- I crawled relevant forums like OpSec, Markets, and Hacking





NZ Darknet

- NZ Darknet is a forum for New Zealand's Dark Web Community
- The site have forums not only on illegal software, but also on drug reviews and other drug discussions
- I crawled the General Discussion Forum only since all the other forums on the site discussed only drugs.
- The General Discussion Forum talked more about software than drugs



Binary Classifier

- The first two cells are just for imports and loading the CSV file
- Cells 3 and 4 ensure that the columns are correct and the labels are binary
- I also printed out the distribution of labels so we can see how imbalance the dataset is after modeling
- The classifier needs to be at thread level. This means that all posts that belong to the same thread need to be grouped.

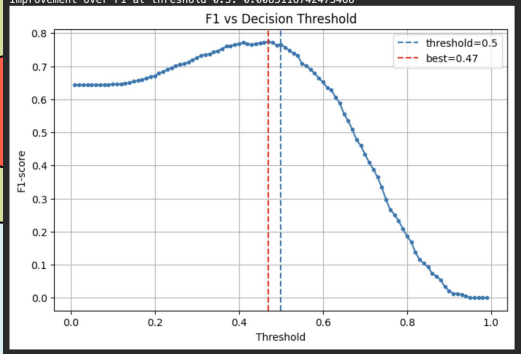
```
!pip install -q scikit-learn pandas matplotlib

import numpy as np
import pandas as pd
import re
import matplotlib.pyplot as plt

from sklearn.model_selection import StratifiedKFold, cross_val_predict
from sklearn.metrics import f1_score, classification_report, confusion_matrix
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import Pipeline
```

```
data_path = "/content/forums_labeled_data.csv"
df = pd.read_csv(data_path)

print(df.head())
print(df.columns)
```

Binary Classifier

- Cell 5 builds the actual text input for the classifier
- For every thread, we clean the title, every post, and concatenate them into a single string
- The result is a list of X_test which is a list of 1000 thread texts and y which is the corresponding labels
- In cell 6, the model is defined. Raw text is turned into numbers by TF-IDF
- In cell 7, I use the 5 fold stratified cross-validation where Each fold uses 80% of the data to train and 20% to test and its repeated 5 times
- If the probability is >= 0.5, then we classify it as positive.
- Cell 8 trains the final model and adds prediction helpers.

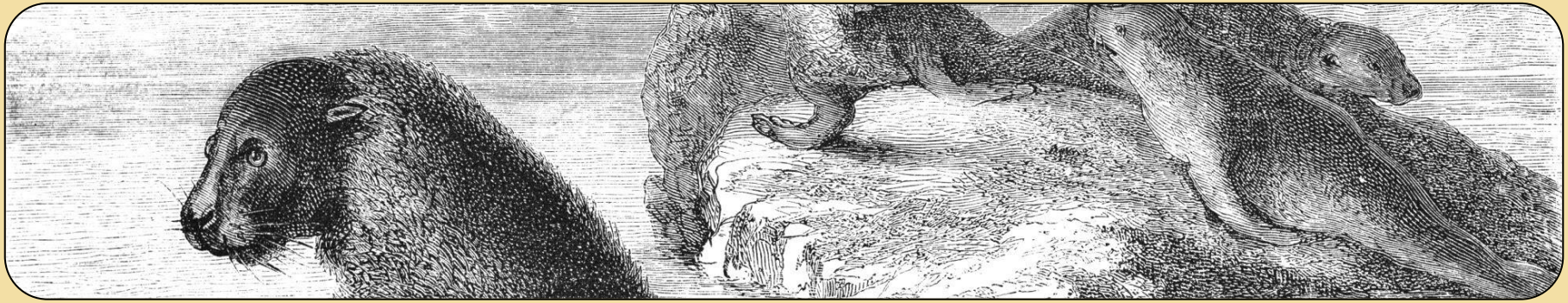
```
Best F1 at that threshold: 0.7736625514403292
Improvement over F1 at threshold 0.5: 0.008316742473468
```




```
Classification Report (Best Threshold):
              precision    recall  f1-score   support

           0     0.8048    0.7681    0.7860       526
           1     0.7550    0.7932    0.7737       474
```

```
        """
        return float(final_pipeline.predict_proba([text])[0, 1])

    def predict_thread_label(text):
        """
        Returns 1 (criminal hacking) or 0 (not)
        using the optimized threshold found in Cell 5.
        """
        prob = predict_thread_probability(text)
        return int(prob >= best_threshold)

    # --- Test with an example ---
    example_title = "Any working exploit packs?"
    example_posts = [
        "Looking for browser exploit kits.",
        "PM me if you have 0day or fresher packs."
    ]

    example_text = example_title + " " + " ".join(example_posts)

    p = predict_thread_probability(example_text)
    label = predict_thread_label(example_text)

    print("Example probability:", p)
    print("Predicted label:", label)
```

```
...  Final model trained on all data.
    Using optimized threshold: 0.47000000000000003
    Example probability: 0.5005176141600262
    Predicted label: 1
```



THANK YOU