University of New Haven

Midterm Project

AI and CyberSecurity

DSCI6015

Simplified Midterm

Varshitha Edla

00888582

University of New Haven

Dr. Vahid Behzadan

April 01, 2024

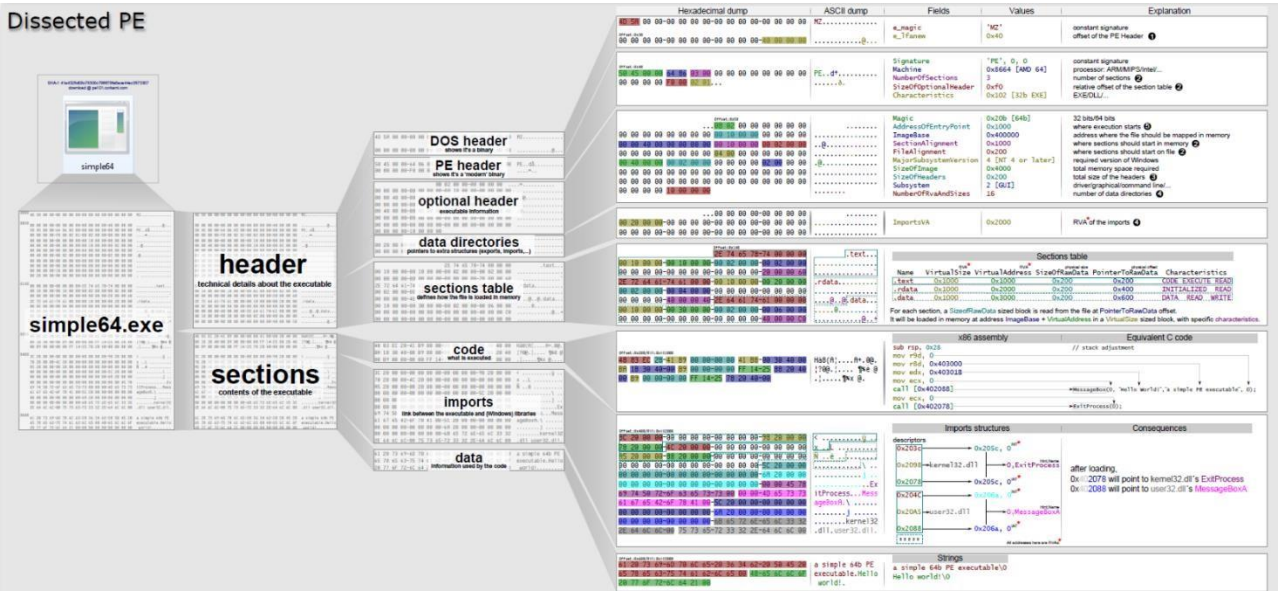
Summary

This article describes the successful use of AWS Sagemaker in the development of a cloud-based Portable Executable (PE) static malware detection API. Using a labeled dataset of binary feature vectors as training data, the Random Forest binary classifier—which the API trained—classifies PE files as safe or harmful. Throughout the project, model building, training, and deployment were done using AWS Sagemaker. To facilitate remote file uploads for threat identification, an intuitive web application was also created. The machine learning tools utilized for creating and executing the models were nltk, pefile, and sklearn. Python served as the primary programming language.

Introduction

PE Files

PE files are widely used by Windows operating systems to hold executable code and related data. These files contain metadata, imported libraries, machine instructions, and other essential information required to run the program. They are commonly used by drivers, applications, and dynamic link libraries (DLLs). They have headers listing the file's architecture, entry point, and section order and are logically arranged. The PE file format is crucial for tasks like malware detection, reverse engineering, and software analysis because it allows the examination and modification of executable code.



Random Forest Classifier

For applications involving both regression and classification, the Random Forest classifier is a versatile and trustworthy machine learning technique. Because it generates predictions by combining multiple models, this technique is categorized as ensemble learning. The various models that make up the Random Forest architecture are decision trees. In Random Forest, the word "forest" refers to a collection of decision trees, each of which was built individually and relies its findings on a particular set of input features.

The main goal of Random Forests is to promote a wide range of decision trees by adding randomization to the training and prediction stages. During training, a random subset of the training data and a random feature selection procedure are used to build each tree.

While managing multi-attribute, high-dimensional data, random forests provide various benefits. Compared to more complex models, they require far less hyperparameter tinkering and demonstrate tolerance to noise and outliers within the dataset. Moreover, Random Forests provide insightful information about feature significance, allowing users to identify the traits that most

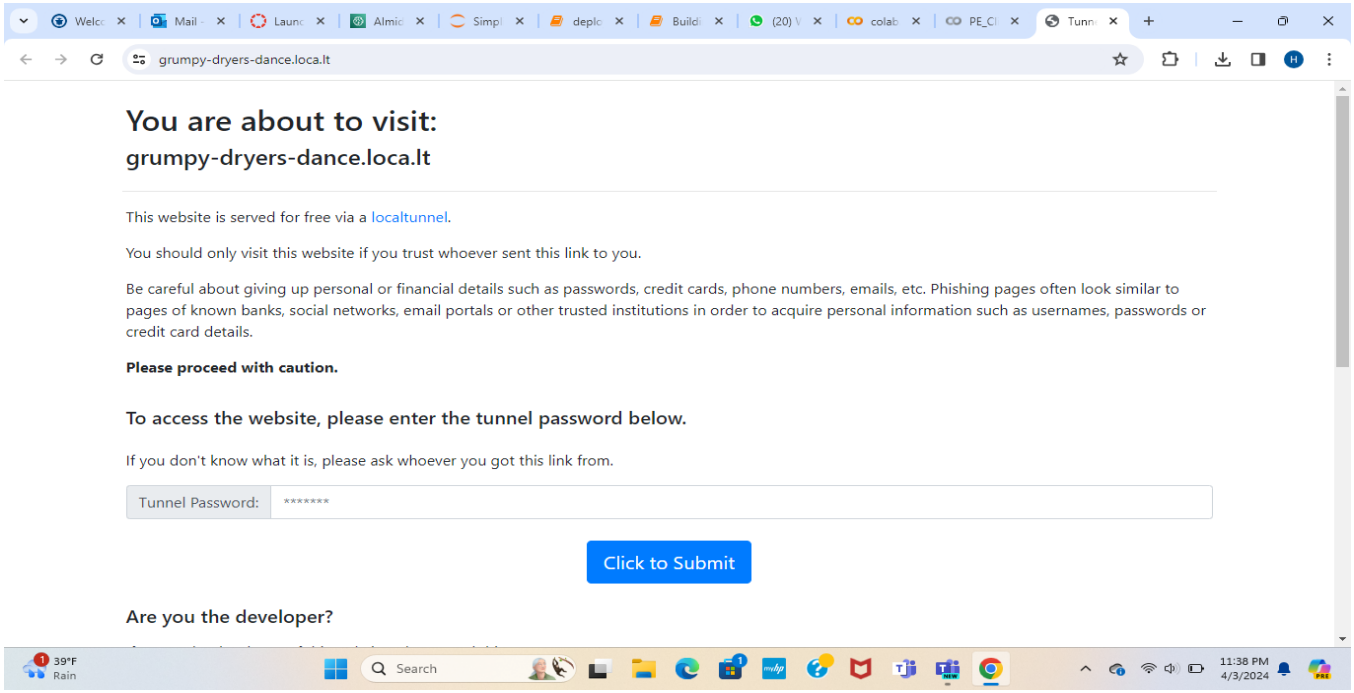
influence the model's predictions. As a result, Random Forests are extensively utilized across numerous industries.

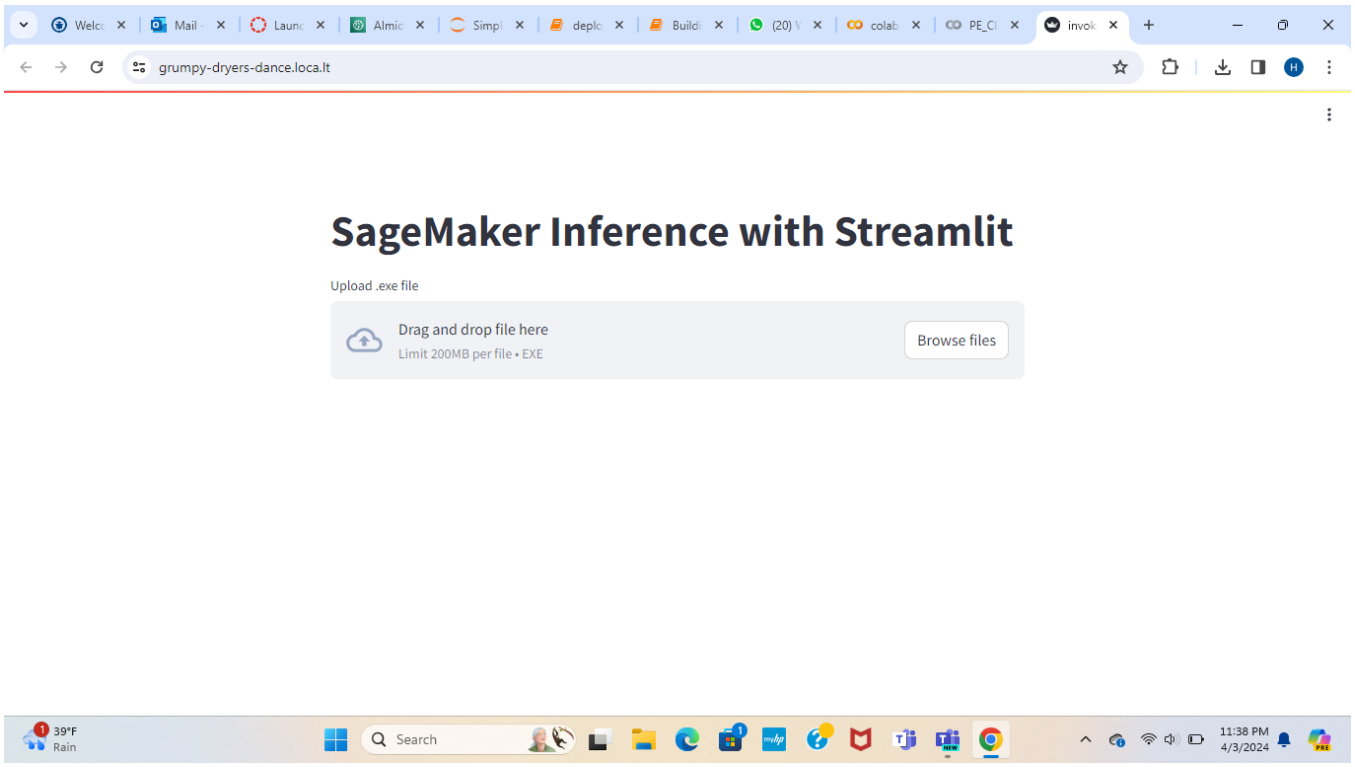
Task Approach:

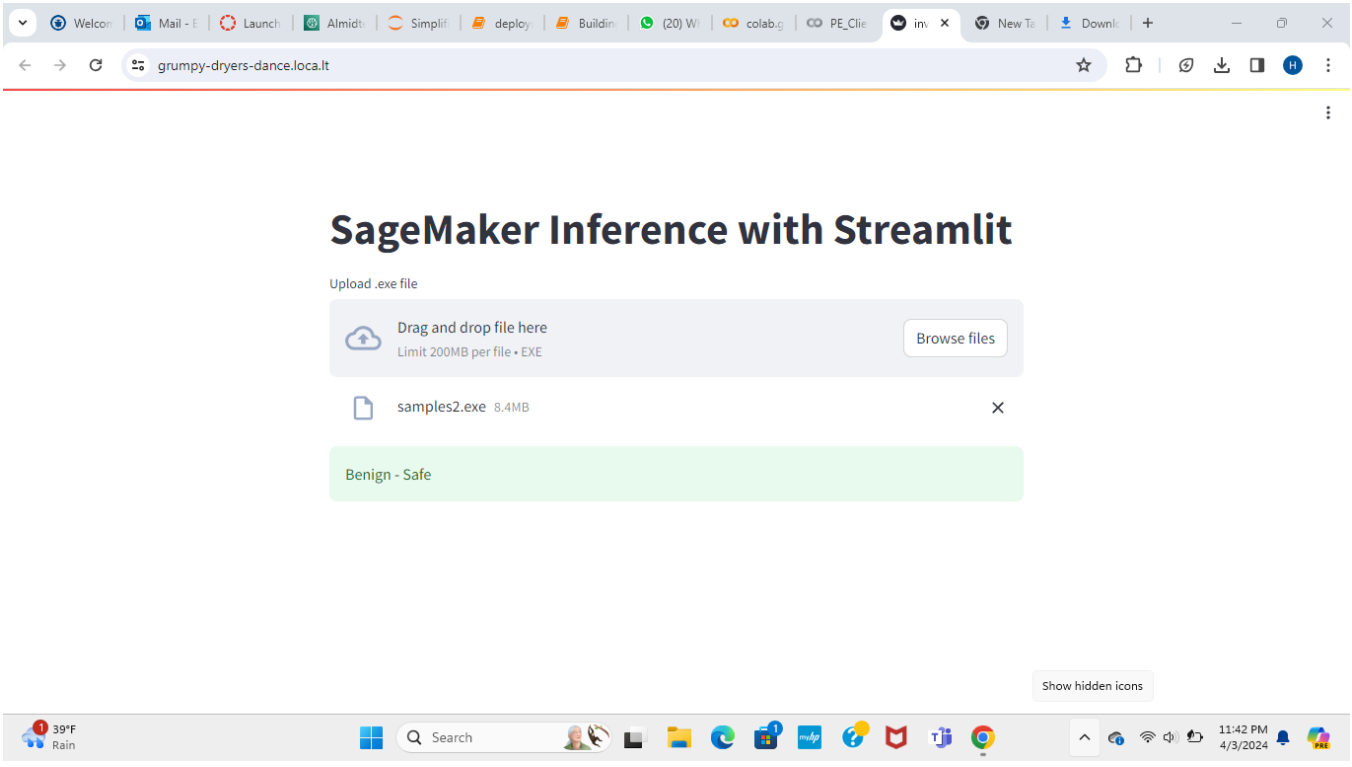
Building and Training the Model The version of Scikit-learn 1.2.1 used in AWS Sagemaker is meant for educational and research purposes. A dataset of binary feature vectors that had been correctly classified was used to train the Random Forest binary classifier. Next, a joblib file containing the trained model is dropped in.

Deploying the Model as a Cloud API: The trained model was used to establish an endpoint for a cloud-based real-time prediction API using Amazon Sagemaker. After the model has been loaded using the joblib file that has been stored, it may be deployed using Sagemaker.SKLearn module. An established and configured endpoint receives the model deployment.

Creating a Client Application: The user interface of a sophisticated online application was designed. Users have the ability to upload executable (.exe) files to the webclient. The webclient uses pefile lib and additional pretrained data to extract the necessary features from the.exe file. The features are delivered to the deployed API once they have been translated to JSON. The classification results (Malware - Danger or Benign - Safe) are subsequently shown by the program. I opened the well-lit application using Google Colab and utilized it for the client deployment.




grumpy-dryers-dance.loca.lt
SageMaker Inference with Streamlit
Upload .exe file
Drag and drop file here
Limit 200MB per file • EXE
Browse files
39°F
Rain
Search
11:38 PM
4/3/2024


grumpy-dryers-dance.loca.lt
SageMaker Inference with Streamlit
Upload .exe file
Drag and drop file here
Limit 200MB per file • EXE
Browse files
samples2.exe 8.4MB
Benign - Safe
Show hidden icons
39°F
Rain
Search
11:42 PM
4/3/2024

Result

My models' accuracy is demonstrated by the data at the top. For the exe file type, the model's prediction accuracy is 97.47%, or 0.9979035639412998.

Conclusion

When a cloud-based PE static malware detection API was created and made accessible, the project was all but completed. The study demonstrates how cloud computing platforms like Google Colab and Amazon Sagemaker can be utilized to build scalable and user-friendly applications, as well as how machine learning can be applied to malware detection with effectiveness.

Resources:

- https://github.com/endgameinc/ember
- https://github.com/endgameinc/ember/tree/master/malconv
- https://github.com/UNHSAILLab/S24-AISec/tree/main/Midterm%20Tutorial
- https://sagemaker-examples.readthedocs.io/en/latest/intro.html
- https://sagemakerexamples.readthedocs.io/en/latest/frameworks/pytorch/get_started_mnist_train_outputs.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/deploy-model.html
- https://github.com/RamVegiraju/Pre-Trained-Sklearn-SageMaker
- https://youtu.be/ueI9Vn747x4?si=KSFTvR9hBnU0u0DO
- https://youtu.be/oOqqwYI60FI?si=3WKd-iDz93mm1Vbe
- https://youtu.be/g6kQl_EFn84?si=9MHbO9I52AS2pPjx