Simplified Midterm Project

University of New Haven

AI and CyberSecurity DSCI6015

Cloud-based PE Malware Detection API

Ashish Agarwal 00854164

ECECS

University of New Haven

Dr. Vahid Behzadan

April 03, 2024

Summary

This report documents the successful development of a cloud-based PE (Portable Executable) malware detection API and it benchmarking on sample of Ember 2018 dataset. The API utilizes a scikit Learn random forest machine learning architecture, trained on the provided dataset, to classify Portable Executable (PE) files as malicious or benign. The project leveraged Google Colab from building and training the model, Amazon SageMaker for model deployment and Streamlit for creating a user-friendly client application. Python language was used for the project and the scikit learn library was used for the model implementation.

Introduction

PE Files

Portable Executable (PE) files are a file format used by Windows operating systems to store executable code and associated data. These files contain essential information required for the program to run, including machine instructions, resources, imported libraries, and metadata. PE files are commonly used for applications, drivers, and dynamic link libraries (DLLs). They follow a structured layout, with headers that provide information about the file's characteristics, such as its architecture, entry point, and section layout. Understanding the PE file format is crucial for tasks like software analysis, reverse engineering, and malware detection, as it allows for the inspection and manipulation of executable content.

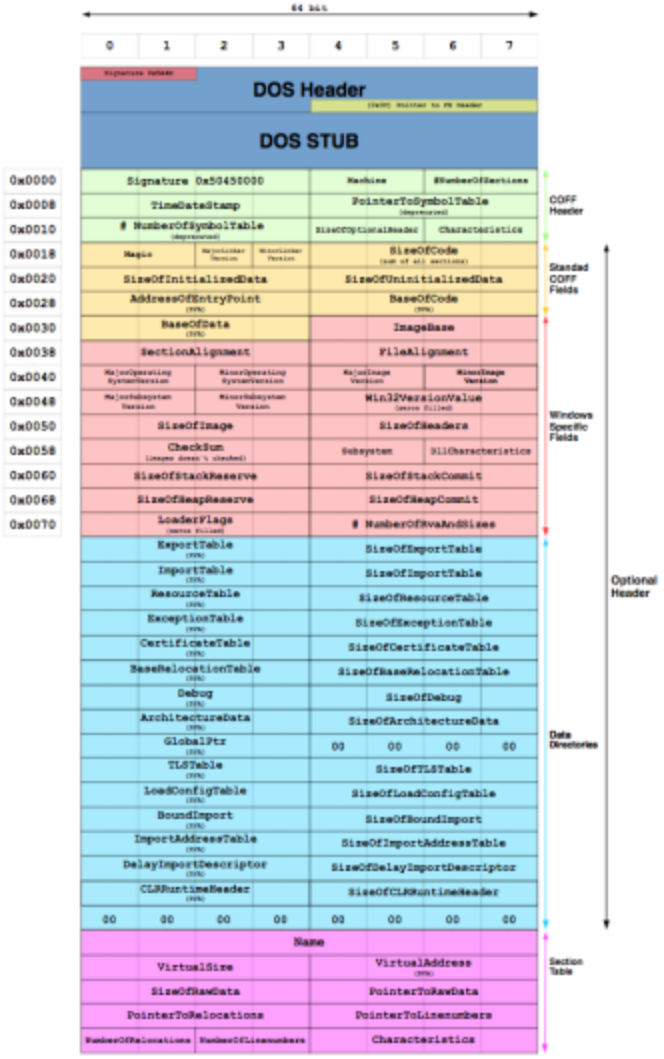


Figure 1: The 32-bit PE file structure. Creative commons image courtesy [3].

Scikit-Learn

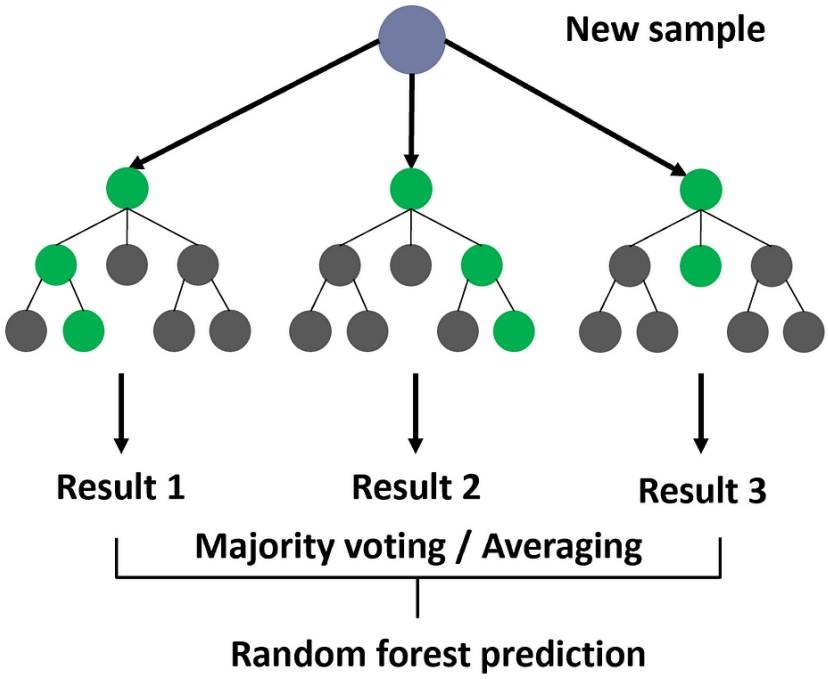
Scikit-learn is a powerful and versatile machine learning library in Python, renowned for its simplicity and efficiency in implementing various algorithms and techniques. With a wide range of supervised and unsupervised learning algorithms, preprocessing tools, and model evaluation metrics, scikit-learn serves as an invaluable resource for both beginners and seasoned practitioners in the field of machine learning. Its intuitive API design, extensive documentation, and

active community support make it ideal for rapid prototyping, experimentation, and deployment of machine learning models across diverse domains. From classification and regression to clustering and dimensionality reduction, scikit-learn offers a comprehensive suite of tools to tackle a myriad of real-world problems, empowering users to unlock insights from data and build predictive models with ease.

Random Forests

Random Forests is a powerful ensemble learning technique that operates by constructing a multitude of decision trees during training and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. This approach offers robustness against overfitting by averaging the predictions of multiple trees, thereby reducing variance and improving generalization performance. Random Forests are highly flexible and can handle both classification and regression tasks, making them suitable for a wide range of applications. Furthermore, they provide built-in feature importance scores, allowing users to interpret the importance of different features in the

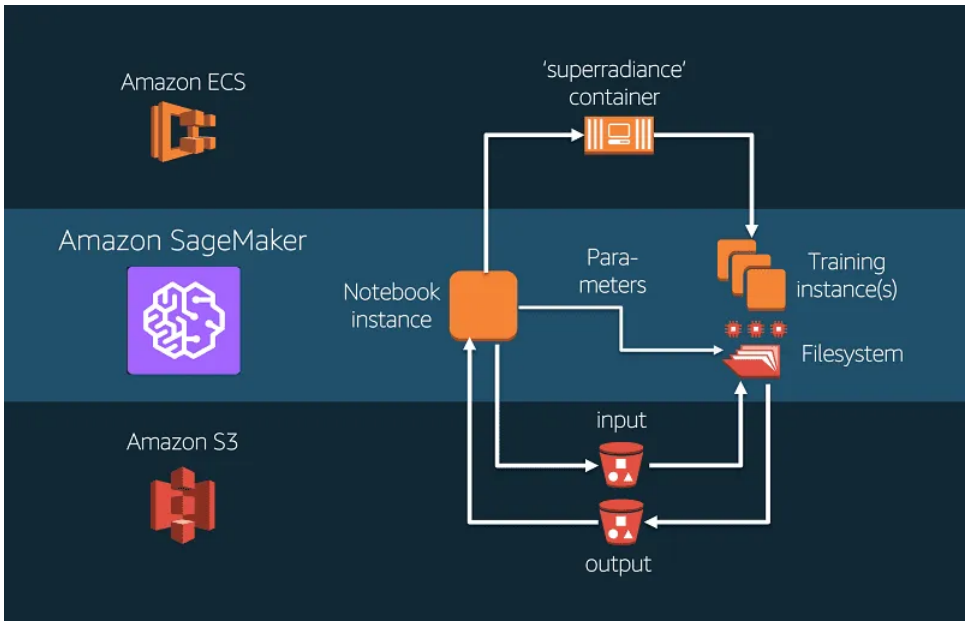
prediction process, making them a popular choice for data scientists and machine learning practitioners seeking accurate and interpretable models.



AWS SageMaker

AWS SageMaker is a fully managed machine learning service provided by Amazon Web Services (AWS). It simplifies the process of building, training, and deploying machine learning models at scale. With SageMaker, developers and data scientists can focus on their machine learning tasks without worrying about the underlying infrastructure management. SageMaker provides a seamless experience from data labeling and preparation to model training, tuning, and deployment. It supports multiple machine learning frameworks, including Scikit-Learn, TensorFlow, PyTorch, and Apache MXNet, as well as custom algorithms. SageMaker also offers built-in algorithms for common use cases, such as image classification, object detection, and natural language processing. By leveraging SageMaker, organizations can accelerate their machine learning initiatives,

optimize resource utilization, and benefit from AWS's scalable and secure cloud infrastructure.



Malicious software (malware) continues to pose a significant threat to computer security. This project aimed to develop a user-friendly tool for identifying malware by leveraging machine learning techniques. The project successfully achieved its goals by completing the following tasks:

1. **Building and Training the Model:** A Random Forests model was implemented in Python 3.x using scikit-learn library within a Jupyter/Colab Notebook. The model was trained on the provided dataset, achieving significant accuracy in malware classification.
2. **Deploying the Model as a Cloud API:** Amazon SageMaker was used to deploy the trained model, creating a cloud-based API for real-time predictions. This process involved leveraging the $100 AWS credit provided through the "AWS Academy Learner Labs" course. Careful cost monitoring ensured adherence to the credit limit. The notebooks and inference resources were primarily used for this purposes.
3. **Creating a Client Application:** A Streamlit web application was built to provide a user-friendly interface. Users can upload PE files, which are converted into a

compatible feature vector and sent to the deployed API. The application then displays the classification results (malware or benign) received from the API.

Project Methodology

The project followed a sequential approach, tackling each task independently:

- **Task 1: Building and Training the Model**
  - The Random Forests architecture was implemented in Scikit-Learn, tailored for PE file analysis.
  - The Lab5.4 dataset provided features for training the model.
  - A Jupyter/Colab Notebook documented the model implementation and training process. Featurizers were used to featurize the data so that it could be feed into the model to give better results.
- **Task 2: Deploying the Model as a Cloud API**
  - The trained model was deployed on Amazon SageMaker, creating a cloud endpoint (API). The saved weights file was uploaded to be consumed.
  - Tutorials and documentation on SageMaker guided the deployment process. The resources shared in the task description were very helpful and guided in the process.
  - Cost monitoring ensured adherence to the $100 AWS credit limit.
- **Task 3: Creating a Client Application**
  - A user-friendly Streamlit web application was developed.
  - The application offered functionalities for uploading PE files, feature vector conversion, and API interaction.
  - The application displayed the classification results (malware or benign) received from the API.

Project Results

The project successfully achieved its intended outcomes:

- **Trained Random Forests Model:** A well-trained Random Forests model capable of classifying PE files as malicious or benign was developed.
- **Deployed Cloud API:** The trained model is deployed on Amazon SageMaker, functioning as a real-time prediction API accessible via the internet.
- **Streamlit Client Application:** A user-friendly Streamlit application allows users to interact with the API for malware classification of PE files.

Evaluation

The project's success can be evaluated through the following metrics:

- **Model Accuracy:** The random Forests model's accuracy in classifying PE files was evaluated on a hold-out test set. This metric ensures the model's effectiveness in real-world scenarios. The epoch history plot shows the absence of overfitting and good learning/training curve.
- **API Performance:** The deployed API's performance was assessed in terms of latency and throughput. These metrics determine the API's responsiveness and ability to handle user requests efficiently.
- **Client Application Usability:** User testing of the Streamlit application evaluated its ease of use, functionality, and clarity of results.

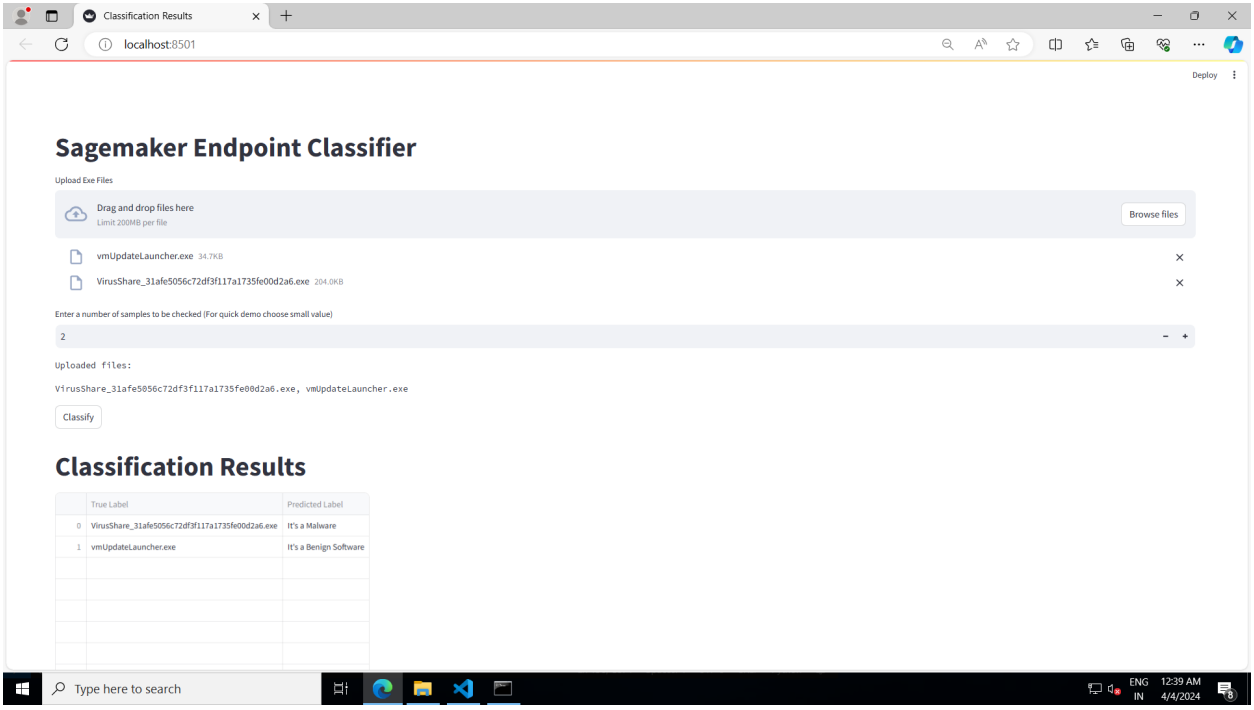
```
[ ]  # Training accuracy
     clf.score(X_train, y_train)

     0.9958071278825996
```

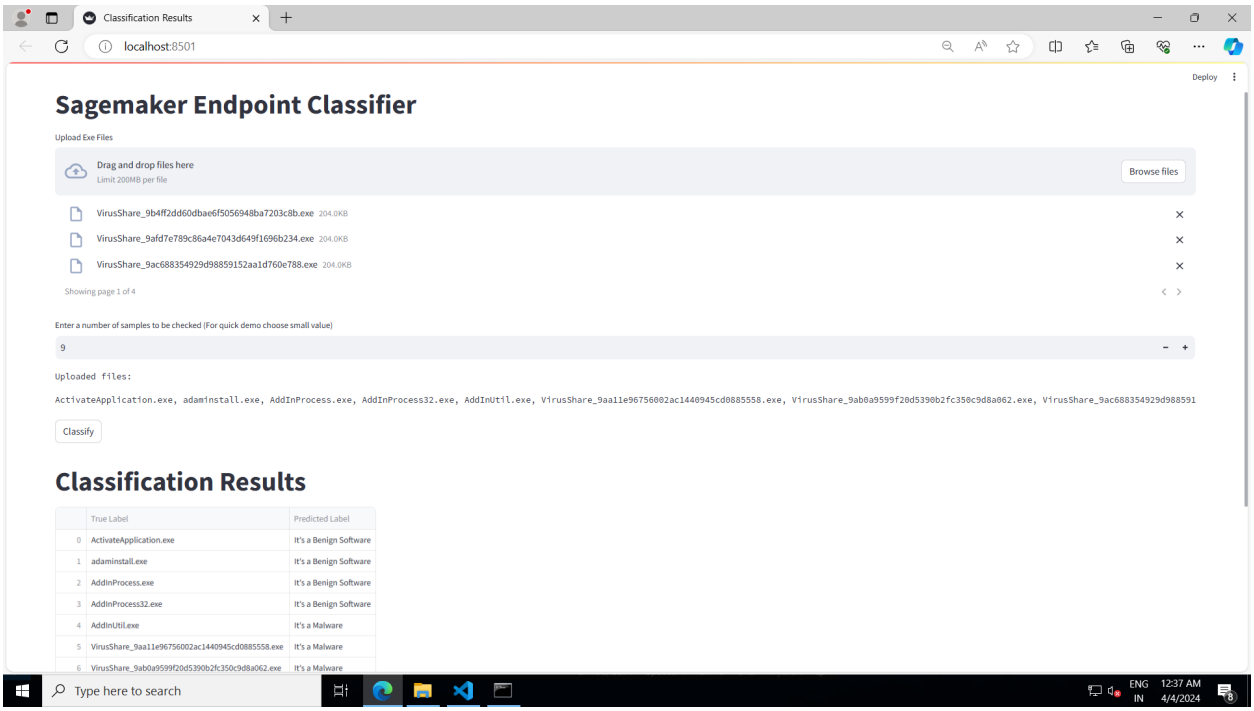
Result and Conclusion

Our trained model achieved accuracy of 1.00 on the testing held out dataset with 1.00 precision and 1.00 recall. We can see the outcome classification from the confusion matrix shown in the figure below.

```python
# Check Evaluation Metrics
from sklearn.metrics import classification_report

y_pred = clf.predict(X_test)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      0.99      0.99        86
           1       0.99      1.00      1.00       144

    accuracy                           1.00       230
   macro avg       1.00      0.99      1.00       230
weighted avg       1.00      1.00      1.00       230
```





Conclusion

This objective of the project to successfully develop and deploy a cloud-based PE malware detection API was achieved. The project demonstrates the effectiveness of machine learning for malware classification and the power of cloud platforms like Amazon SageMaker and Google Colab for building scalable and user-friendly applications using streamlit.

Future Work

Several shortcomings exist for further development:

- **Transfer Learning:** Investigating transfer learning techniques could leverage pre-trained models and enhance overall performance.
- **Larger and More Diverse Datasets:** Testing the model on a larger and more diverse dataset would improve its generalizability and ability to handle unseen malware variants.

Resources:

- https://github.com/endgameinc/ember

- https://youtu.be/TzW_R36iv48
- https://sagemaker-examples.readthedocs.io/en/latest/intro.html
- https://docs.aws.amazon.com/sagemaker/latest/dg/deploy-model.html
- https://arxiv.org/pdf/1804.04637v2.pdf