**NAME: SANDRA DARKSON.**

**COURSE: DSCI-6015-01: ARTIFICIAL INTELLIGENCE AND CYBERSECURITY**

**SPRING 2024**

**MIDSEM PROJECT**

**Machine Learning Model Training and Deployment Report**

1. **Introduction**

This report outlines the process of training a machine learning classifier to identify malicious URLs. It also covers the deployment of the model as an API endpoint on Amazon SageMaker and the development of a Streamlit Python client to interact with the deployed endpoint. The project involves using a dataset of URLs that are labeled as malicious or benign, and utilizing various technologies such as Python, Scikit-Learn, Amazon SageMaker, and Streamlit.

2**. Training a Classifier for Malicious URL Detection**

When implementing machine learning for URL classification, the first step I took was to prepare the data. This involved tokenizing URLs and converting them into numerical features. The data was then divided into training and testing sets. A Random Forest classifier from Scikit-Learn was selected as the model, and it was trained on the training data to classify URLs as malicious or benign. During the evaluation of a model's performance on testing data, metrics such as accuracy, precision, recall, and F1 score are used to determine its ability to distinguish between two types of URLs. In this case, I used TensorFlow and scikit-learn libraries to train a Convolutional Neural Network (CNN) model for detecting malicious URLs. The model achieved an accuracy of roughly 56.16% on the test data. I also saved the trained model as 'cnn\_model.h5’.

3. **Deployment on Amazon SageMaker**

Upon completing the data set training, I immediately proceeded towards deployment and successfully created an S3 bucket in AWS. However, I Faced challenges in deploying the model as an API endpoint on Amazon SageMaker due to configuration issues and permissions. The screenshots below are proofs.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

4. **Streamlit Python Client Development**

To create interactive web applications with Python, Streamlit can be installed by using the command `pip install streamlit`. For client development, a Streamlit Python script is developed to create a user interface that allows users to input a URL. The script uses the AWS SDK or SageMaker SDK to make API calls to the deployed SageMaker endpoint with the user's URL input. The result, which is either a malicious or benign classification, is then displayed on the webpage for user feedback.

**Conclusion**

This project provided valuable insights into the entire process of developing and deploying a machine learning model, as well as creating a client application. The key takeaways from this project include skills related to data preparation, model selection, AWS integration, and web development. The lessons learned from this project range from model evaluation and deployment challenges, to user experience considerations and maintenance requirements. Suggestions for future improvements include data augmentation, model optimization, enhanced UI features, continuous monitoring, and security enhancements. Overall, this project has contributed significantly to a better understanding of real-world machine learning deployment scenarios and best practices.

**References**

- Scikit-Learn Documentation: <https://scikit-learn.org/>

- Amazon SageMaker Documentation: <https://docs.aws.amazon.com/sagemaker/>

- Streamlit Documentation: <https://docs.streamlit.io/>

https://github.com/SandraDarkson/Midterm-AI-Project/settings

**Key Learnings**

To optimize machine learning workflows, there are some key steps that you need to focus on. Firstly, preprocess data to enhance model readiness. Secondly, carefully select algorithms tailored to dataset needs and performance metrics. Finally, leverage AWS services like Amazon SageMaker for seamless deployment and resource management in the cloud. By following these steps, you can ensure that your machine learning workflows are effective and efficient.

**Lessons Learned**

It is essential to conduct comprehensive assessments of machine learning models before deploying them to ensure their effectiveness. Deploying models as API endpoints can present challenges such as resource allocation, scalability, and security considerations. A great user experience is crucial in client applications, emphasizing intuitive UI design and clear result visualization to enhance engagement. Maintenance and updates are critical after deployment, involving tasks like versioning, monitoring, and addressing drift over time.

**Suggestions for Future Improvements.**

To improve the performance of a machine learning model and enhance the user experience, there are various strategies that can be employed. These include utilizing data augmentation techniques to enable better generalization, optimizing models through hyperparameter tuning, enriching the user interface with visualizations and feedback mechanisms, continuously monitoring the model's performance for refinement, and ensuring that security measures are in place to protect the privacy and integrity of data.