Securing the ML Lifecycle

2022-03-17

Authors:
Dr. Carmen Kempka
WIBU-SYSTEMS AG
carmen.kempka@wibu.com

Prof. Dr. Andreas Schaad
University of Offenburg
andreas.schaad@hs-offenburg.de

CONTENTS

1 Overview 4
1.1 Introduction 4
1.2 Purpose 4
1.3 Scope 4
1.4 Audience 4
1.5 Terms and Definitions 5
2 Motivation: Secure Machine Learning 5
2.1 Object Recognition in Manufacturing 5
2.2 Medical Image Classification 6
3 The ML Lifecycle 6
3.1 ML Lifecycle 6
3.2 ML Stakeholders 7
4 Attacking the Machine Learning Lifecycle 8
4.1 Attacking the Training Data 8
4.1.1 Poisoning Attack 8
4.1.2 Countermeasures 9
4.2 Attacking the Training Process 9
4.2.1 Observing the Preprocessing 10
4.2.2 Poisoning Attack 10
4.2.3 Configuration Stealing Attack 10
4.2.4 Countermeasures 10
4.3 Attacking the Deployed Model 11
4.3.1 Stealing the Model 11
4.3.2 Evading the Model 11
4.3.3 Unintended Usage 11
4.3.4 Countermeasures 12
4.4 Attacking the Query 12
4.4.1 Query Interception or Modification 12
4.4.2 Countermeasures 12
4.5 Summary 12
5 A Secure Machine Learning Cheat Sheet 13
5.1 Initial ML Security Assessment 13
5.2 ML Security Checklist 14
6 Acknowledgements 16

FIGURES

Figure 3-1: ML Lifecycle, Assets and Stakeholders 7
Figure 5-1: Initial Security Assessment 14

TABLES

Table 5-1: Overall Project Assessment 14

Table 5-2: Training Data Assessment.....15

Table 5-3: Preprocessing / Training Assessment 15

Table 5-4: Model Deployment Assessment.....15

1 OVERVIEW

1.1 INTRODUCTION

Machine Learning (ML) has established itself a widely adopted technique that complements and even supersedes traditional software engineering and development in the industrial context[1]. ML has been applied in countless industrial scenarios[2], such as design and manufacturing optimization, predictive maintenance, or material sourcing optimization. ML can even be used to improve cybersecurity in the IoT context[3]. At the same time, using ML also introduces new attack vectors into IoT and IIoT devices. Due to the highly data-driven nature of ML, specific care must be taken to implement a secure ML lifecycle, aligned with the organization's secure software development lifecycle.

1.2 PURPOSE

The purpose of this document is to draw attention to the fact that ML activities need to be defined in a structured and secure process and couched in a supporting organizational structure. This can differ from standard secure software development and supply chains, as a result of ML's highly data-driven nature.

1.3 SCOPE

This document surveys existing recommendations and approaches to implementing a secure ML lifecycle. An analysis of ML lifecycle models is provided alongside known attacks and countermeasures for each of the lifecycle steps, including specific cases of adversarial attacks. We also provide a discussion of the stakeholders involved in the ML process chain. Specific focus is placed on the protection of training data and trained models, both from a data integrity and a commercial (licensing, IP) perspective. Our intention is not to produce an exhaustive list of all possible attacks, but rather a realistic lifecycle model, identify the stakeholders involved, and present selected attacks and appropriate countermeasures. This should allow decision makers to apply our observations to their organizational context.

1.4 AUDIENCE

Software Architects, Product Management, Operations

---

[1] Yanming Yang, Xin Xia, David Lo, and John Grundy. 2021. A Survey on Deep Learning for Software Engineering. ACM Comput. Surv., December 2021. DOI: *https://doi.org/10.1145/3505243*

[2] Massimo Bertolini, Davide Mezzogori, Mattia Neroni, Francesco Zammori: Machine Learning for industrial applications: A comprehensive literature review. Expert Syst. Appl. 175: 114820 (2021).

[3] Murat Kuzlu, Corinne Fair, and Ozgur Guler: Role of Artificial Intelligence in the Internet of Things (IoT) cybersecurity. Discov Internet Things 1, 7 (2021). *https://doi.org/10.1007/s43926-020-00001-4*

1.5 Terms and Definitions

The following terms and definitions are key to understanding this document:

- ML – Machine Learning
- SE – Software Engineering
- SDLC – Secure Development Lifecycle

2 Motivation: Secure Machine Learning

Machine Learning can be defined as programming a computer so that it can learn from data[3]. Unlike in procedural development, a developer does not simply define the algorithmic approach to a solution, but rather the required steps and parameters to extract patterns from data to produce generalized models[4]. In other words, based on a (large) set of training data, a training process is executed, which leads to a trained model. This could be models for image recognition in radio-diagnostics or industrial maintenance, models for financial prediction, models for malware detection in networks[5], models for speech recognition, or text-to-speech synthesis[6].

These three core artefacts (training data, training process description, trained model) can be susceptible to inadvertent modifications or even intentional attacks. The verification and validation mechanisms used in standard software development (e.g., static code analysis or unit testing) do not suffice to guarantee the overall quality of training data, training code, or trained models.

When discussing the ML Lifecycle (Section 3.1), we also need to understand the stakeholders involved (Section 3.2) before we can discuss the actual threats, attacks, and countermeasures for the individual stages of the ML lifecycle (Section 4). The following two brief case studies will be referred to in the discussion.

2.1 Object Recognition in Manufacturing

A company produces complex machines consisting of several hundred individual parts. An image recognition model has been trained to recognize each part (even if already combined with others) and will display information about it to the engineers during the assembly process. Several dozen parts suppliers provide detailed technical descriptions and images for each part, but the many individual parts are only assembled on the company's premises.

[3] Geron, A. "Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems" O'Reilly, 2019.

[4] Kelleher, J. et al. "Fundamentals of Machine Learning for Predictive Data Analytics" MIT Press, 2020.

[5] Andreas Schaad, Dominik Binder: FEX - A Feature Extractor for Real-Time IDS. ISC 2021.

[6] Vanessa Barnekow, Dominik Binder, Niclas Kromrey, Pascal Munaretto, Andreas Schaad, Felix Schmieder: *Creation and Detection of German Voice Deepfakes.* 14th International Symposium on Foundations & Practice of Security, 2021.

We can already observe that several stakeholders are interacting here, though we left it open who trains the model and how the raw supplier data (images and technical documentation) is transferred into the ML pipeline.

2.2 Medical Image Classification

A hospital is using a pre-trained commercial model to analyze medical images on their network. However, the hospital also wants to use their own databases to improve the model. Data from the radio-diagnostics department is sent to a cloud platform where a commercial provider has set up their training infrastructure. Models are retrained and updated on a regular basis.

Again, at least two stakeholders are involved: the actual end user of the model and the company providing the initial model. The case again leaves it open how the existing model is retrained and how the required data is transferred.

3 The ML Lifecycle

As for any software solution, training an ML model should serve a defined business objective and the non-functional requirements derived from it. Only if these are clearly defined can the required training data be gathered, and the correct training parameters be defined.

3.1 ML Lifecycle

When collecting data, the two core parameters are quantity and quality. This specifically concerns monitoring the balance of data to avoid a later bias (e.g., in medical data[7]). Another concern is where the data is generated. One possibility is that this is done within the full control of the organization that defines the training process. However, it could also be data that is obtained from public repositories or from contractual agreements with third parties or customers.

In an initial data analysis, the collected data is examined with respect to its structure, data types and categories, possible outliers, or possible immediate correlations. This is then usually followed by a preprocessing step to, for example, remove any statistical noise or zero values. The features (column names) used for the later training are identified, and the data is split into a training, validation, and testing set.

The training phase usually starts with a reasoned choice (configuration) of suitable training algorithms and associated parameters (e.g., number of hidden layers, batch size, or learning rate). “Suitable” in this context should imply that the training approach fits the defined business objective. The training data is then fed into this defined training pipeline, resulting in incrementally validated and adjusted training parameters and intermediate models.

[7] Vokinger, K.N., Feuerriegel, S., & Kesselheim, A.S. “Mitigating bias in machine learning for medicine.” Commun Med 1, 25 (2021).

The final model is evaluated against the separate test data. Such a model can then be made available in the context of, for example, a cloud service so that it can be queried (sometimes referred to as an inference model). However, how this is done technically depends on the framework that was used for training (e.g., scikit-learn, keras, or pytorch). Interoperability standards such as ONNX[8] try to address this. The MLOps community[9] also emphasizes that we cannot treat machine learning like traditional software engineering.

Figure 3-1 brings together our observations concerning the ML lifecycle and introduces the assets and stakeholders involved, which will be discussed in the next sections.

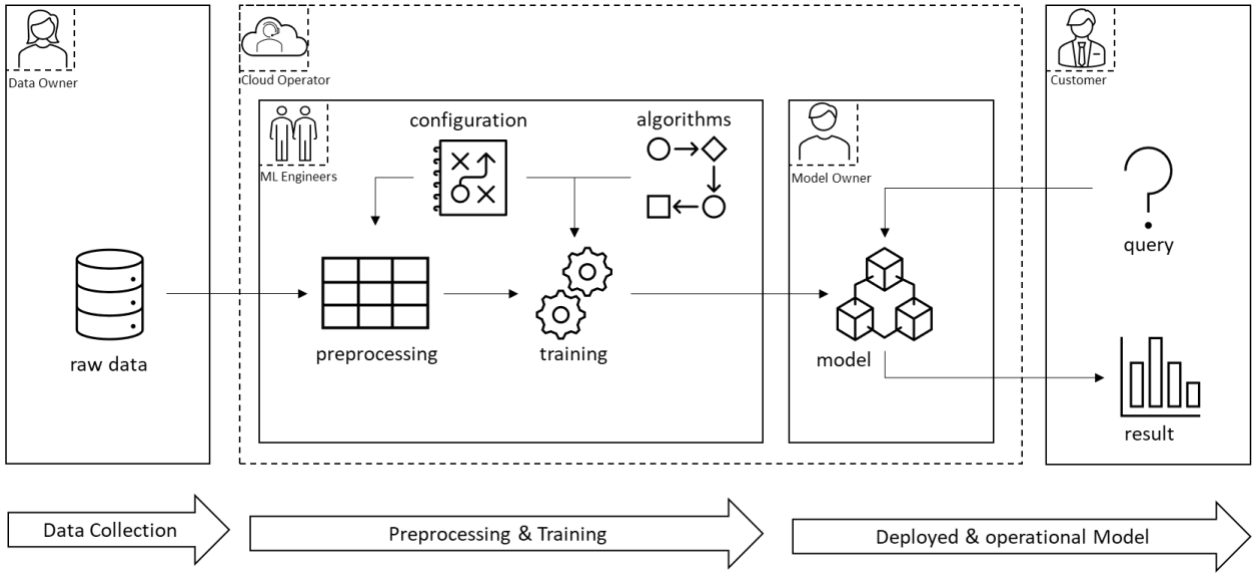

Figure 3-1: ML Lifecycle, Assets and Stakeholders

3.2 ML STAKEHOLDERS

For the purposes of this paper, we need to emphasize that the entire ML process is a result of several stakeholders interacting. This observation is fundamental for our discussion of the security requirements concerning the ML process as well as its technical artifacts.

Several stakeholders may participate in any given machine learning lifecycle[10]. Some entity will be the actual owner of the training data or act as an aggregator to whom ownership-like rights have been transferred as part of some legal agreement. The training code may be owned by some other entity, but it can be as confidential as the training data, as it includes the final training model architecture and selected parameters. The final "trained" (inference) model may be owned by either of the two former entities, but could also have been transferred to a separate,

[8] *https://onnx.ai/*

[9] *https://ml-ops.org/*

[10] Wojciech Ozga, Do Le Quoc, Christof Fetzer*: Perun - Confidential Multi-stakeholder Machine Learning Framework with Hardware Acceleration Support.* DBSec 2021*: 189-208*

distinct model owner. At this point, we already note that full transfer of ownership is different from a licensed model. The actual inference code that is required to query the model may again be owned by a separate actor. Likewise, the data of a query used for inference may be owned by an entity different from all the mentioned actors. Finally, all this data (training data, training code, trained model, inference, …) may be processed in a technical environment not under full control of the owner (i.e., in a cloud service).

4 ATTACKING THE MACHINE LEARNING LIFECYCLE

A variety of attacks has been identified over the last years that could compromise the individual stages and assets of the machine learning lifecycle[11]. We align our observations with several of the taxonomies provided by academic[12], industrial[13], non-profit,[14] and governmental institutions[15]. Note that we do not discuss how machine learning could be used by malicious actors[16]. Again, our intention is not to present a full list of all possible attacks, but rather a realistic lifecycle model, the stakeholders involved, selected attacks, and selected countermeasures.

4.1 ATTACKING THE TRAINING DATA

The quality of the training data is the baseline for a model that can be used for accurate predictions or classifications. If an attacker succeeds in changing existing data or injecting data at will, we consider this a poisoning attack.

4.1.1 POISONING ATTACK

The intention behind such an attack could be to diminish the model's quality or even cause a denial of service. The attacker could also try to influence the model generation process, so that the model would only fail in certain situations to the advantage of the attacker. In this case, we would speak of targeted poisoning attacks, as the attacker wants to have only specific examples misclassified. How such a poisoning attack could even be carried out by trained adversarial model has already been described in a malware detection scenario[17].

In the context of our medical image classification scenario, poisoning the training data could result in a model that will not correctly classify certain images. In fact, it has already been

[11] M. Xue, C. Yuan, H. Wu, Y. Zhang, and W. Liu, "Machine Learning Security: Threats, Countermeasures, and Evaluations," in *IEEE Access*, vol. 8, pp. 74720-74742, 2020, DOI:10.1109/ACCESS.2020.2987435.

[12] Barreno, M., Nelson, B., Joseph, A.D., *et al.* The security of machine learning. *Mach Learn* **81,** 121–148 (2010).

[13] *https://docs.microsoft.com/en-us/security/engineering/failure-modes-in-machine-learning*

[14] *https://atlas.mitre.org/*

[15] *https://www.enisa.europa.eu/publications/artificial-intelligence-cybersecurity-challenges*

[16] *https://www.europol.europa.eu/publications-events/publications/malicious-uses-and-abuses-of-artificial-intelligence*

[17] Sen Chen, Minhui Xue, Lingling Fan, Shuang Hao, Lihua Xu, Haojin Zhu, and Bo Li. 2018. Automated poisoning attacks and defenses in malware detection systems: An adversarial machine learning approach. computers & security 73 (2018), 326–344.

successfully reported that medical training data can be intentionally poisoned to later result in incorrect patient treatment recommendations[18].

Even more sophisticated attacks can be performed if the adversary has knowledge of the algorithms used for preprocessing. Image material used for training, for example, can be altered if the scaling algorithm is known: the unscaled image is edited in such a way that it does not look different from the original, but the scaled image contains some adversarial artifact erroneously used for training or even scales down to a completely different picture. Several works have been published describing such image scaling attacks[19] and countermeasures[20].

4.1.2 COUNTERMEASURES

Conventional access controls and integrity preserving measures can help mitigate this type of attack. Besides secure communication at the network level, confidentiality and integrity preservation at the application layer could be achieved by means of trusted elements[21]. If we think that poisoned data has already been transmitted by the source, more advanced anomaly detection techniques could be used to directly address the data's provenance[22] or try to identify false data at the preprocessing stage[23] by means of statistical techniques[24].

4.2 ATTACKING THE TRAINING PROCESS

Besides the more conventional attempt to just technically disturb the training process, more advanced attacks may again try to influence the chosen algorithms and related libraries (e.g. for GPU interaction). If the training process is running in a cloud, we could also assume the cloud operator to behave in an "honest-but-curious" fashion.

[18] M. Jagielski, et al., "Manipulating Machine Learning: Poisoning Attacks and Countermeasures for Regression Learning," in 2018 IEEE Symposium on Security and Privacy (SP), San Francisco, CA, USA, 2018 pp. 19-35.

[19] Qixue Xiao, Yufei Chen, Yu Chen, Kang Li, "Seeing is Not Believing: Camouflage Attacks on Image Scaling Algorithms", Proceedings of the 28th USENIX Security Symposium, 2019.

[20] Erwin Quiring, David Klein, Daniel Arp, Martin Johns, Konrad Rieck, "Adversarial Preprocessing: Understanding and Preventing Image-Scaling Attacks in Machine Learning", Proceedings of the 29th USENIX Security Symposium, 2020.

[21] Andreas Schaad, Tobias Reski, Oliver Winzenried: Integration of a Secure Physical Element as a Trusted Oracle in a Hyperledger Blockchain. ICETE (2) 2019: 498-503.

[22] N. Baracaldo, B. Chen, H. Ludwig, A. Safavi, and R. Zhang, "Detecting Poisoning Attacks on Machine Learning in IoT Environments," *2018 IEEE International Congress on Internet of Things (ICIOT)*, 2018, pp. 57-64, doi: 10.1109/ICIOT.2018.00015.

[23] *https://blog.f-secure.com/poisoning-attacks-in-a-distributed-learning-environment/*

[24] Benjamin IP Rubinstein, Blaine Nelson, Ling Huang, Anthony D Joseph, Shing-hon Lau, Satish Rao, Nina Taft, and J Doug Tygar. 2009. Antidote: understanding and defending against poisoning of anomaly detectors. In Proceedings of the 9th ACMSIGCOMM Conference on Internet Measurement. 1–14.

4.2.1 Observing the Preprocessing

Observing the preprocessing stage may yield valuable information for the attacker. Specifically, any knowledge about the features used in it could support later adversarial attacks. This also includes knowledge about the features not used for a training model.

4.2.2 Poisoning Attack

An attacker may choose to either perform a targeted poisoning attack or cause byzantine failures during the training process. By failures, we do not necessarily imply technical failures resulting in disrupted training processes. The attacker may rather focus on influencing the model's quality itself. One obvious reason could be that the final model should simply not correctly classify later activities of the attacker (e.g., network intrusion), but more subtle attacks could intend to just diminish the overall model quality to achieve a competitive advantage.

4.2.3 Configuration Stealing Attack

At the training stage, the attacker may also try to derive knowledge about the used configuration and learning parameters. This also includes trying to illicitly obtain knowledge about the steps taken as part of the preprocessing, such as the selected features. Such knowledge could then facilitate later adversarial attacks aimed at evading model prediction or classification.

4.2.4 Countermeasures

Again, conventional access controls can be used against these attacks. However, specifically in shared training environments, possible application layer countermeasures might have to use more fine-grained software protection services[25]. While access to critical parameters of the training process could be controlled, we could even consider shifting[26] parts of the training process to trusted execution environments such as SGX enclaves or secure elements in general.

The possible dilemma in using, but not fully trusting a cloud operator[27] has long been discussed by the database, trusted computing, and secure computation communities. A realistic engineering scenario for shared machine learning using trusted execution environments such as SGX was recently presented[28].

[25] *https://www.wibu.com/us/products/codemeter/codemeter-cloud-server.html*

[26] Wojciech Ozga, Do Le Quoc, *Christof Fetzer: Perun - Confidential Multi-stakeholder Machine Learning Framework with Hardware Acceleration Support.* DBSec 2021*: 189-208.*

[27] B. Alouffi, M. Hasnain, A. Alharbi, W. Alosaimi, H. Alyami, and M. Ayaz, "A Systematic Literature Review on Cloud Computing Security: Threats and Mitigation Strategies," in IEEE Access, vol. 9, pp. 57792-57807, 2021, doi: 10.1109/ACCESS.2021.3073203.

[28] Wojciech Ozga, Do Le Quoc, Christof Fetzer*: Perun - Confidential Multi-stakeholder Machine Learning Framework with Hardware Acceleration Support.* DBSec 2021*: 189-208*

4.3 Attacking the Deployed Model

Attacking the deployed model should be considered from two perspectives, i.e. stealing the model or trying to evade the model.

4.3.1 Stealing the Model

Stealing the model could again be a rather conventional attack, requiring illegitimate access to files and copying them. But more sophisticated attacks may try to duplicate a model just by posing queries. In a similar fashion, the attacker could try to infer knowledge about the underlying training dataset. These latter two means are often referred to as adversarial attacks, as they are conducted by models trained with the purpose of copying other models[29].

4.3.2 Evading the Model

The attacker may also try and generate adversarial models that work to construct malicious input and yield erroneous model output, while appearing unmodified to human observers[30]. In fact, we are aware that there are many examples of published models used for supporting IT security purposes (e.g., phishing detection) that were immediately attacked by corresponding adversarial models[31] and that successfully evaded the originally proposed and trained classifier.

4.3.3 Unintended Usage

Models may be made available to customers just like regular software. There are active discussions about how to license trained models as well as training parameters with respect to open-source usage. Technical solutions to embed watermarks in model output also appear to be available[32]. However, we are not aware of any true technical license enforcement embedded in models[33]. Our assumption is that, given the current state of the art, such enforcement can only realistically be done in the operational environment.

---

[29] Ishai Rosenberg, Asaf Shabtai, Yuval Elovici, and Lior Rokach. 2021. Adversarial Machine Learning Attacks and Defense Methods in the Cyber Security Domain. ACM Comput. Surv. 54, 5, Article 108 (June 2022).

[30] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. 2017. Practical Black-Box Attacks against Machine Learning. ASIA CCS '17

[31] Hossein Shirazi, Bruhadeshwar Bezawada, *Indrakshi Ray,* Chuck Anderson*: Directed adversarial sampling attacks on phishing detection.* J. Comput. Secur. 29(1)*: 1-23 (2021)*

[32] Huili Chen, Bita Darvish Rouhani, Cheng Fu, Jishen Zhao, and Farinaz Koushanfar. 2019. DeepMarks: A Secure Fingerprinting Framework for Digital Rights Management of Deep Learning Models. In Proceedings of the 2019 on International Conference on Multimedia Retrieval (ICMR '19).

[33] *https://www.mayerbrown.com/-/media/files/news/2019/01/expert-qanda-on-artificial-intelligence-ai-licensing-w0219801.pdf*

4.3.4 COUNTERMEASURES

Besides standard engineering and access controls[34], we acknowledge that significant work[35] in the form of publicly available software libraries (e.g., ART, CleverHans) has been done to test models for their resistance to adversarial attacks[36].

4.4 ATTACKING THE QUERY

4.4.1 QUERY INTERCEPTION OR MODIFICATION

Though not under full control of the model owner or cloud operator, the query by a customer as well as the result delivered to that customer may contain sensitive data. This could make it possible to not only intercept immediate business data, but to even reconstruct parts of the actual data used for training. Given a data record and black-box access to a model, it has been shown that this can be used to determine whether a record was in a model's training dataset[37].

4.4.2 COUNTERMEASURES

Besides protecting the query content by means of network or application-level encryption, the problem of protecting against membership inference appears to be a classical trade-off between data protection and utility. In fact, current ML frameworks already provide support for performing such inference attacks as part of testing the robustness of the model[38]. One current avenue pursued by researchers to counter inference attacks is that of Differential Privacy[39].

4.5 SUMMARY

We have now identified the stages of the ML lifecycle, associated assets, and stakeholders as well as possible attacks and some possible (albeit selected) countermeasures. We argued that some of these countermeasures require comparatively easy engineering interventions, whilst others are still subject to academic discussions. One key observation was that, as far as we are aware of there seem to be no major technical solutions to support the licensing of models resulting from machine learning processes. To be more specific, whilst it appears feasible to harness the supporting environment with access-control features to enable licensing, the open research question is whether such licensing could become an intrinsic feature of a trained model. We

[34] *https://docs.microsoft.com/en-us/security/engineering/threat-modeling-aiml*

[35] *https://adversarial-robustness-toolbox.readthedocs.io/en/latest/*

[36] *https://github.com/cleverhans-lab/cleverhans*

[37] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership Inference Attacks Against Machine Learning Models," *2017 IEEE Symposium on Security and Privacy (SP)*, 2017, pp. 3-18, doi: 10.1109/SP.2017.41.

[38] *https://github.com/tensorflow/privacy/tree/master/tensorflow_privacy/privacy/privacy_tests*

[39] *https://cacm.acm.org/magazines/2021/7/253460-the-limits-of-differential-privacy-and-its-misuse-in-data-release-and-machine-learning/fulltext*

explicitly did not discuss how to increase trustworthiness in ML, although our proposed secure machine learning lifecycle can be considered a fundamental prerequisite.

5 A Secure Machine Learning Cheat Sheet

We propose a simple set of artifacts to support an initial security analysis of a machine learning project.

5.1 Initial ML Security Assessment

Our previous discussions emphasized that, at its core, the four main assets in a machine learning lifecycle are the training data, training configuration, resulting trained model, and the final queries to the model and generated results. The usual Confidentiality, Integrity, and Availability triad (CIA) still holds true with respect to these assets.

However, the threats differ. Stealing training or configuration data or even the trained model corresponds to some extent to what we know from conventional information security. Yet, the idea of injecting data (poisoning) to impact the integrity of the trained model or even causing such a poisoned model to not correctly classify data may not appear immediately apparent. The same applies to an attacker's attempts to infer knowledge about the involved assets.

Figure 5-1 can be used to support such a discussion and allow us to ask questions such as:

- "Is stealing a confidentially trained model possible in our ML pipeline?"
- "Can an attacker poison the training data to reduce the integrity of the trained model?"
- "Can an adversarial model evade our classifier due to leaked confidential knowledge about the features used for training?"

We acknowledge that the very simple matrix we propose is heavily asset-centric, and a different approach may be to focus only on data flows. We also assume that any of these attacks could be performed by an adversarial attacker model. To offer simple mnemonic support, we use the acronym SPIE (Stealing, Poisoning, Inference, Evasion) to indicate which phase of the ML lifecycle may be primarily (though not exclusively) subject to a certain type of attack.

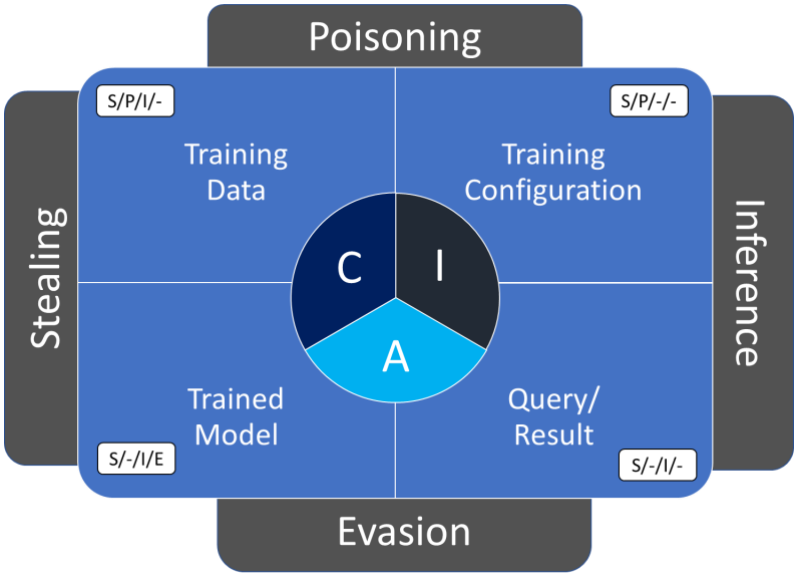

Figure 5-1: Initial Security Assessment

5.2 ML Security Checklist

Based on our discussions of (selected) attacks and countermeasures, we now present a checklist with sample questions that could be used (and should be added to in any given organization) to support initial discussions about securing a machine learning project.

Questions labelled "standard" should be asked in any software project, while "advanced" questions require in-depth technical knowledge of the ML pipeline.

Overall Project		
Question / Action Item	**Standard**	**Advanced**
Do you know your data assets?	✓	
Do you know all involved stakeholders and where your ML pipeline is executed?	✓	
Can you clearly articulate your business objective (needed to validate your model)?	✓	
What interest could an attacker have in downgrading or evading your model?	✓	
Do your project and the involved data have to meet any regulatory demands or compliance criteria?	✓	

Table 5-1: Overall Project Assessment

Training Data		
Question / Action Item	**Standard**	**Advanced**
Where does your raw data come from, and has it been securely transmitted?	✓	
Can you prove data provenance at the application layer?		✓
Can an attacker poison the training data to reduce the integrity of the trained model?		✓
Could the training data itself be considered a trade secret?	✓	

Table 5-2: Training Data Assessment

Preprocessing / Training		
Question / Action Item	**Standard**	**Advanced**
Have you secured your training parameters and overall setup?	✓	
Are you using attested training algorithms and related software (e.g., signed GPU drivers)?	✓	
Are you using adversarial robustness testing techniques?		✓
Could any of the preprocessing data (e.g., selected features) be of interest to an attacker?		✓

Table 5-3: Preprocessing / Training Assessment

Model Deployment / Operation		
Question / Action Item	**Standard**	**Advanced**
Where is your trained model deployed?	✓	
Are standard access controls for accessing the model enforced?	✓	
Is there any need for commercial license-based access controls?		✓
Are your queries and answers secured?	✓	

Table 5-4: Model Deployment Assessment

6 ACKNOWLEDGEMENTS

The views expressed in the IIC Journal of Innovation are the author's views and do not necessarily represent the views of their respective employers nor those of the Industry IoT Consortium®.

- Return to *IIC Journal of Innovation landing page* for more articles and past editions

.