

# Vulnerabilities of Machine Learning Algorithms to Adversarial Attacks for Cyber-Physical Power Systems

---

Tapadhir Das

Ph.D. Candidate

Department of Computer Science and Engineering

University of Nevada, Reno

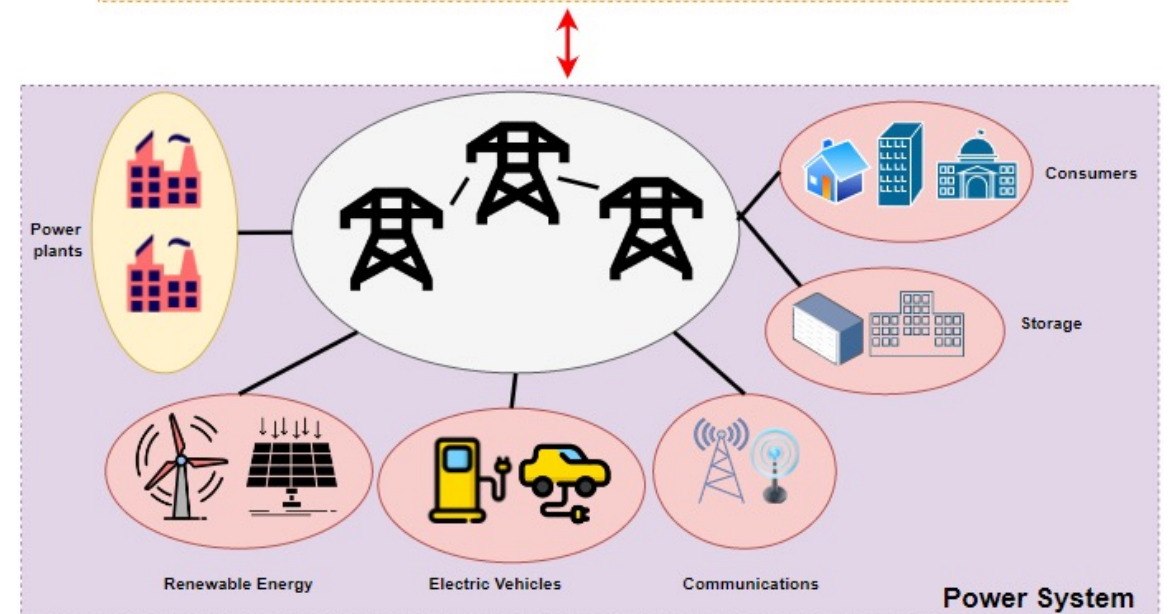
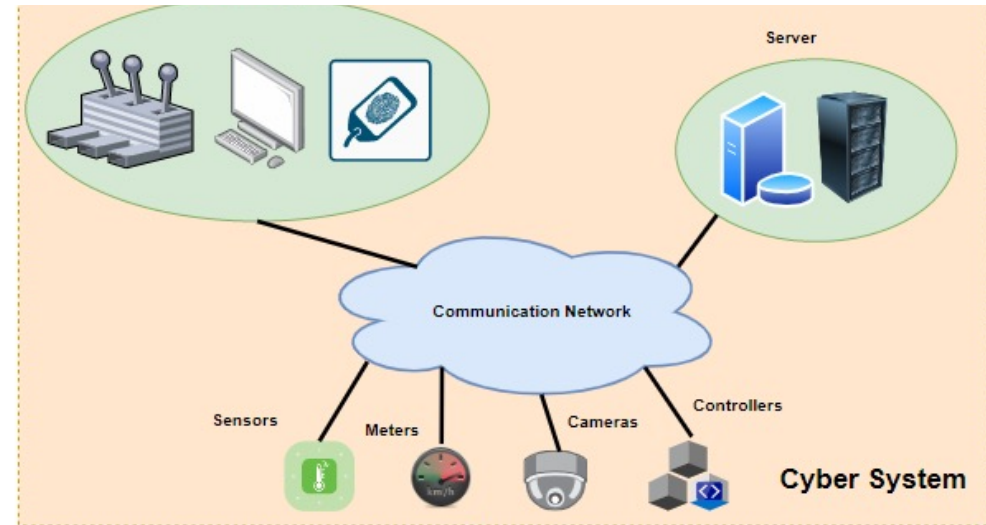


# Overview

- Usage of machine learning in cyber-physical power systems
- Security of machine learning algorithms
- Adversarial attacks
- Vulnerabilities in machine learning algorithms
- Attack models
- Protection Strategies
- Conclusion

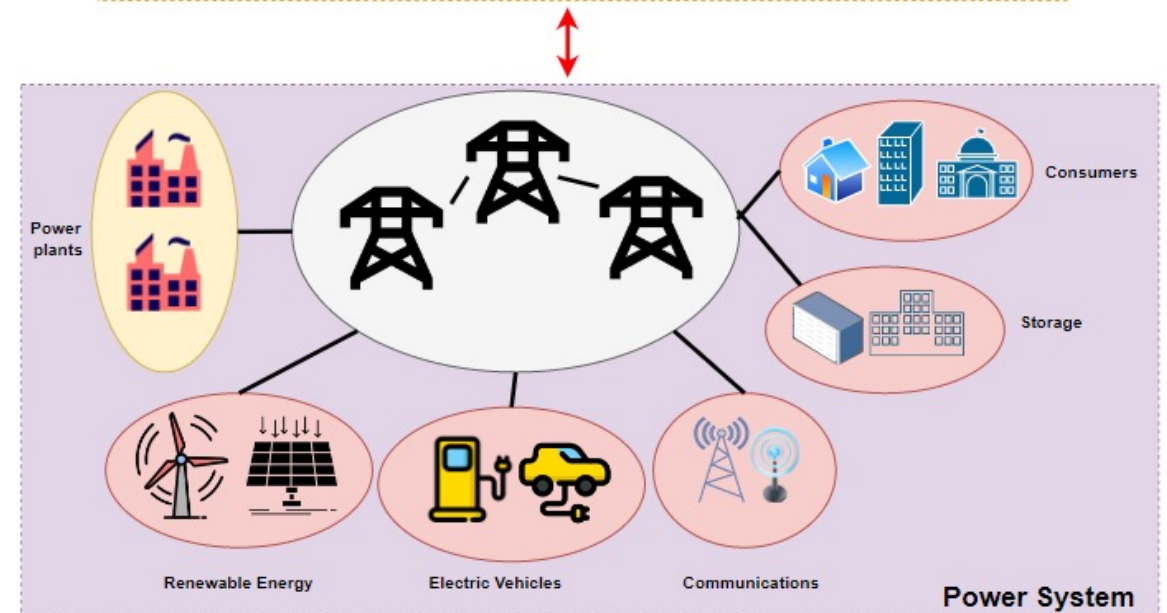
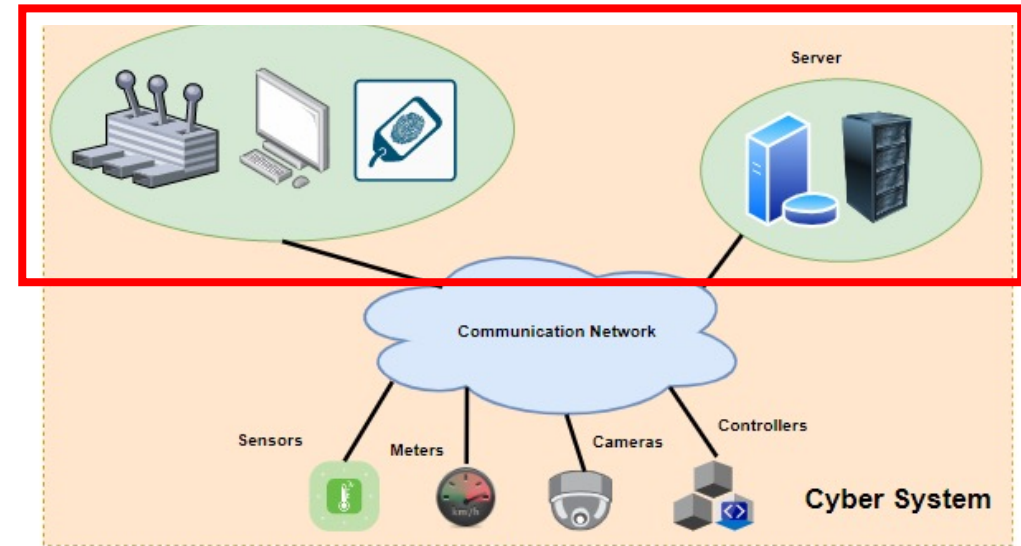
# Cyber-physical Power Systems (CPPS)

- Contemporary power system architectures are cyber-physical in nature.
- These architectures contain 3 layers:
  - Computing: servers, computer
  - Data acquisition: sensors, phasor measurement units
  - Physical [1]
- CPPS help transform how we interact with power systems, making them more connected and observable.



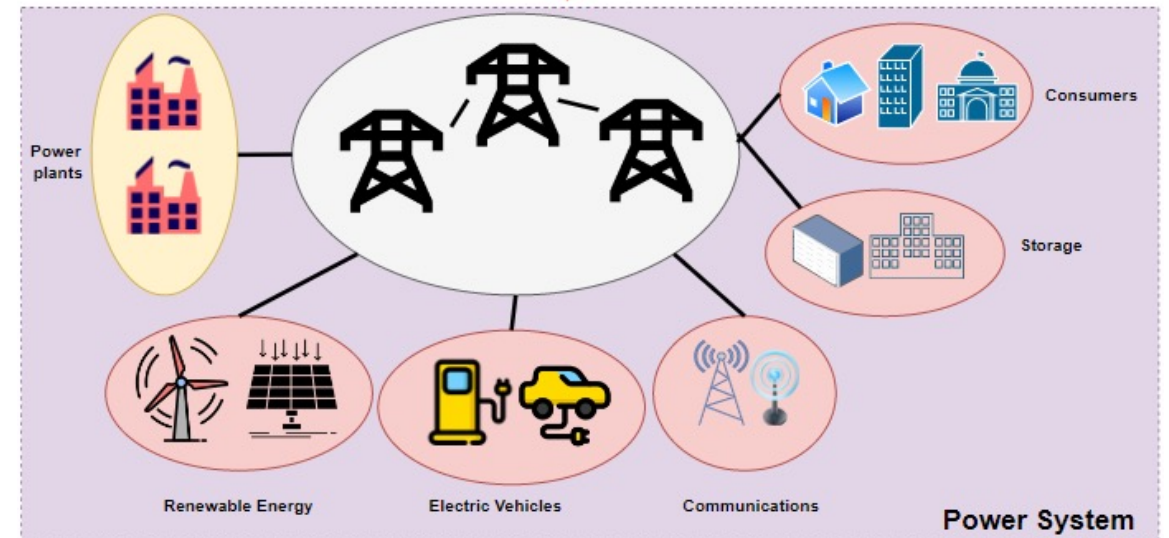
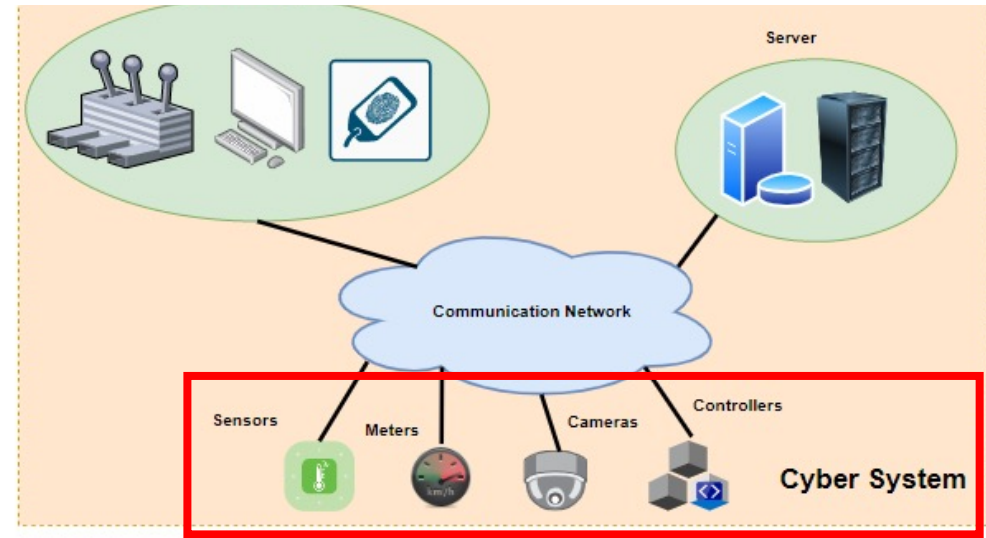
# Cyber-physical Power Systems (CPPS)

- Contemporary power system architectures are cyber-physical in nature.
- These architectures contain 3 layers:
  - Computing: servers, computer
  - Data acquisition: sensors, phasor measurement units
  - Physical [1]
- CPPS help transform how we interact with power systems, making them more connected and observable.



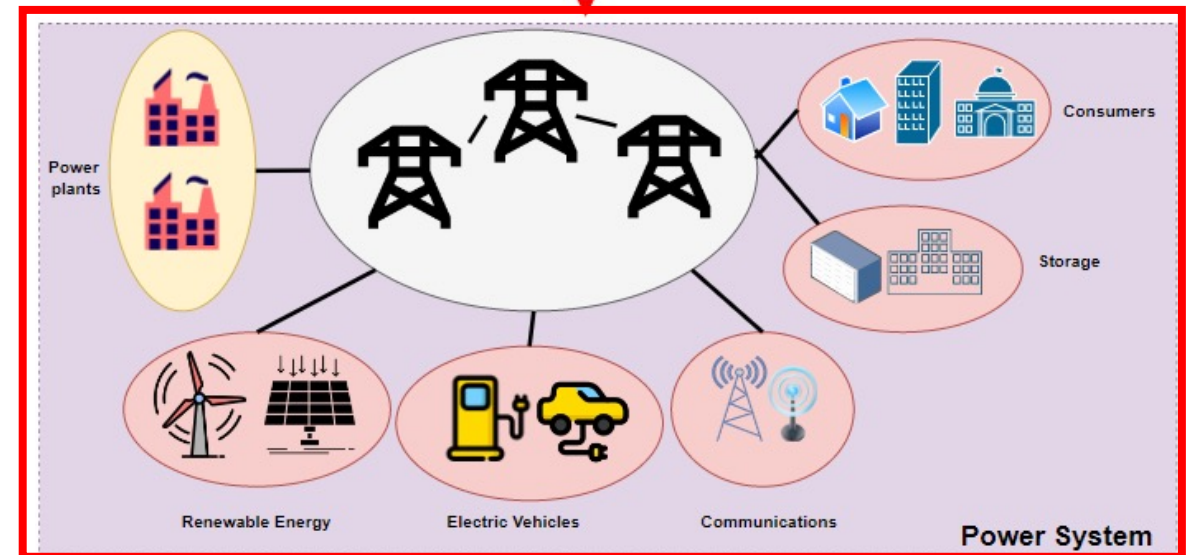
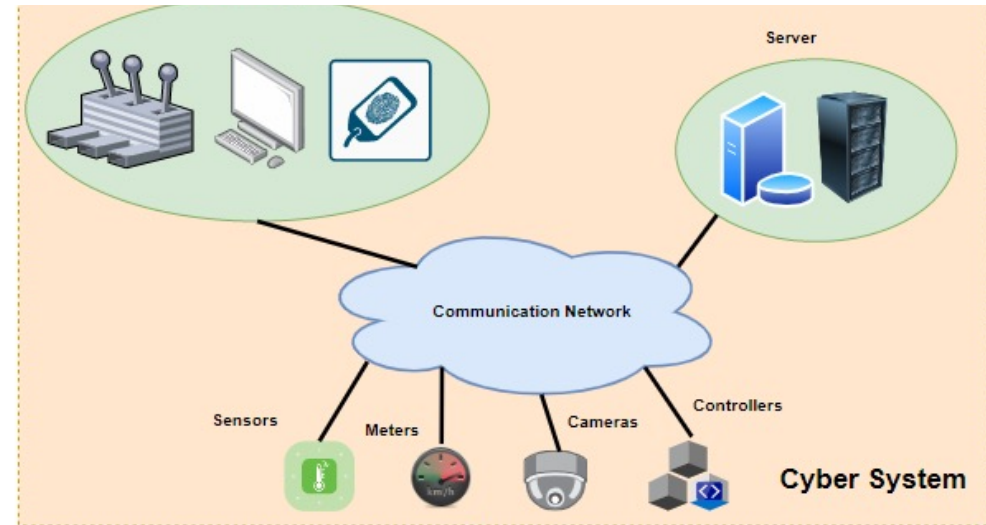
# Cyber-physical Power Systems (CPPS)

- Contemporary power system architectures are cyber-physical in nature.
- These architectures contain 3 layers:
  - Computing: servers, computer
  - Data acquisition: sensors, phasor measurement units
  - Physical [1]
- CPPS help transform how we interact with power systems, making them more connected and observable.



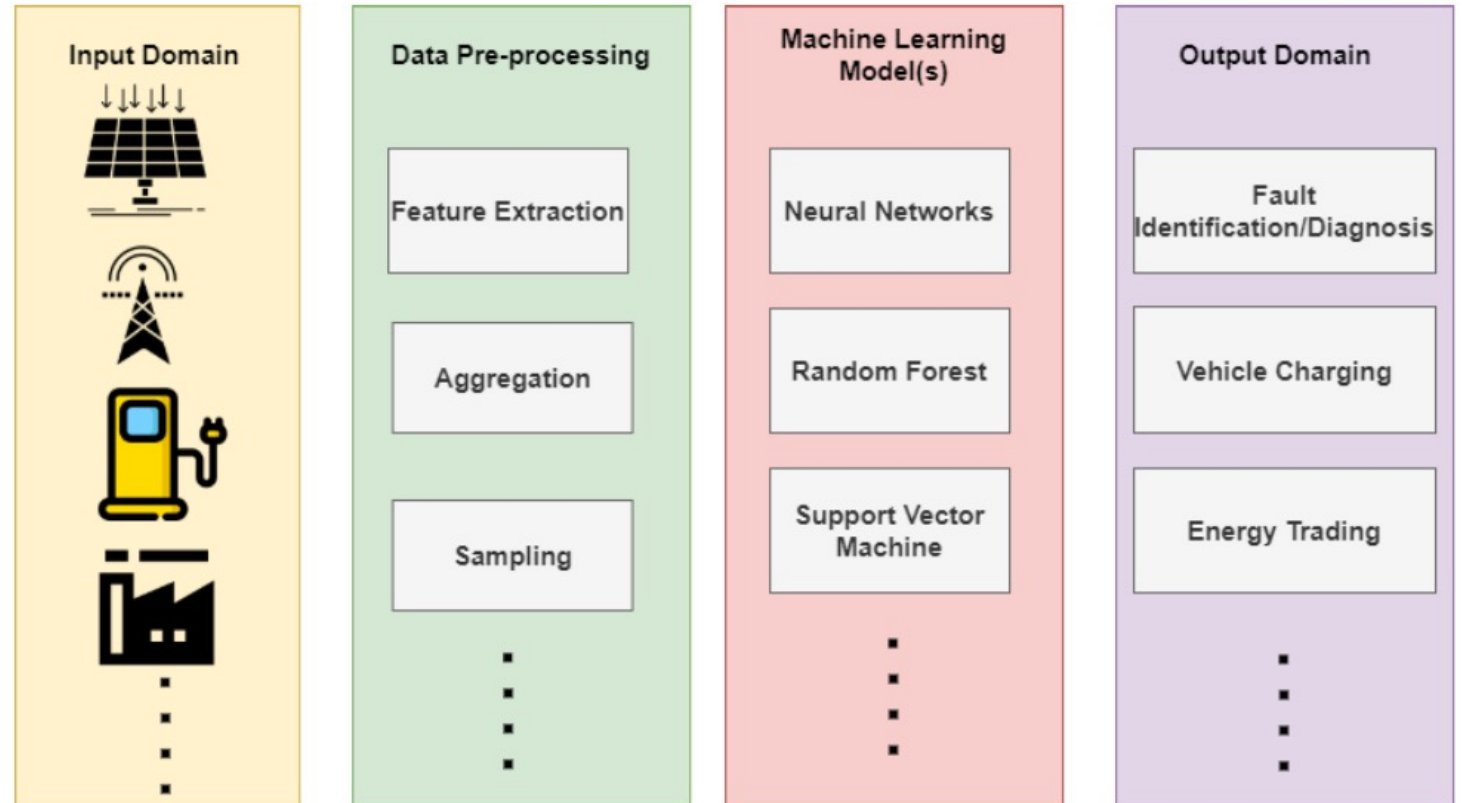
# Cyber-physical Power Systems (CPPS)

- Contemporary power system architectures are cyber-physical in nature.
- These architectures contain 3 layers:
  - Computing: servers, computer
  - Data acquisition: sensors, phasor measurement units
  - Physical [1]
- CPPS help transform how we interact with power systems, making them more connected and observable.



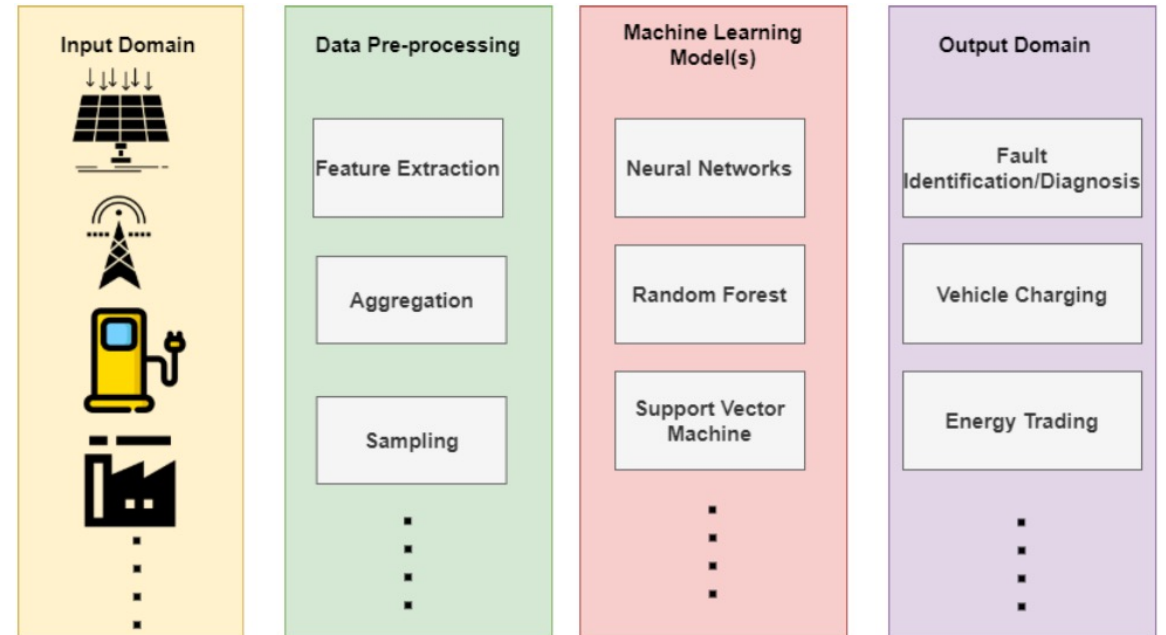
# Machine learning (ML) in CPPS

- Application of ML in CPPS
  - Electric vehicle power predictions
  - Energy trading in distribution systems.
  - Optimal scheduling for battery swapping stations
  - High-performance solar cell creation
  - Performance estimation and monitoring
  - Monitoring energy consumption in smart homes
- ML makes modern CPPS intelligent, efficient, optimal, faster
- ML Pipeline: Input, Pre-process, Model, Output



# ML Security

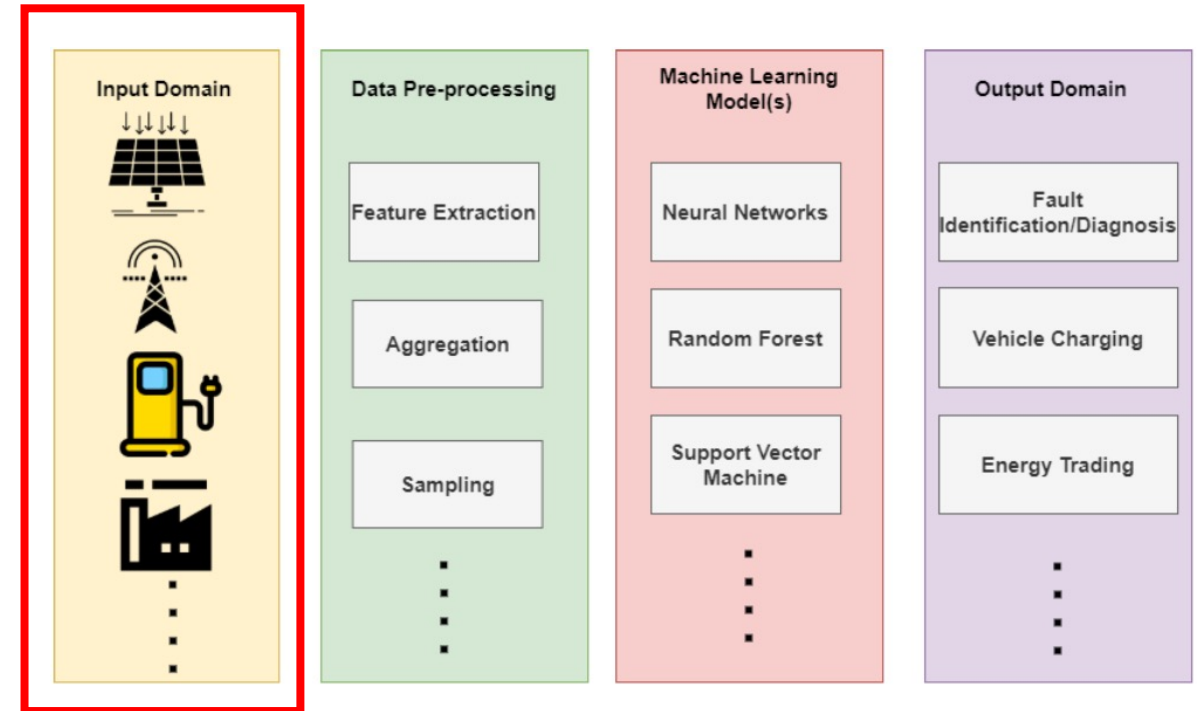
- According to NIST, there are multiple Targets of Attack (TA) in an ML pipeline, for CPPS [2]:
  - Input domain: Malicious tampering with input sensor data or sensor devices.
  - Data Pre-processing: Manipulating collected datasets and maliciously altering them before they get fed to the ML model.
  - ML model: Poisoning the model by creating adversarial examples of data to make misclassifications. Can be in both training and testing.
  - Output domain: Maliciously tampering with output devices, screens to display incorrect results/predictions.





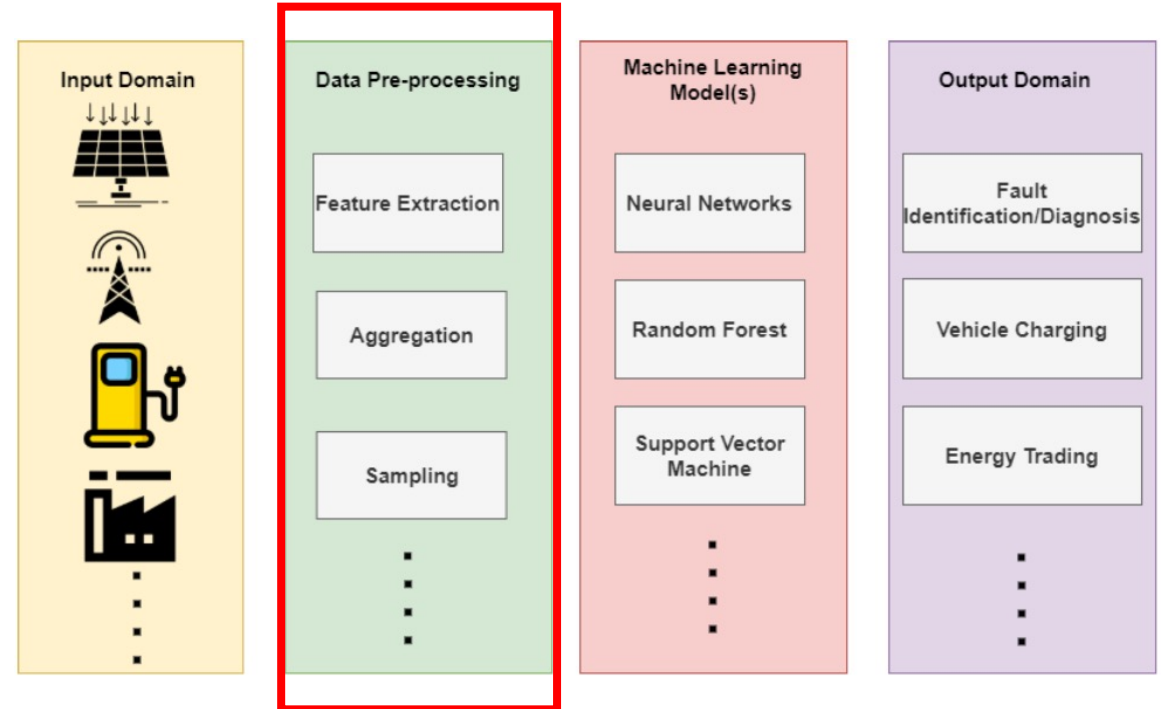
# ML Security

- According to NIST, there are multiple Targets of Attack (TA) in an ML pipeline, for CPPS [2]:
  - Input domain: Malicious tampering with input sensor data or sensor devices.
  - Data Pre-processing: Manipulating collected datasets and maliciously altering them before they get fed to the ML model.
  - ML model: Poisoning the model by creating adversarial examples of data to make misclassifications. Can be in both training and testing.
  - Output domain: Maliciously tampering with output devices, screens to display incorrect results/predictions.



# ML Security

- According to NIST, there are multiple Targets of Attack (TA) in an ML pipeline, for CPPS [2]:
  - Input domain: Malicious tampering with input sensor data or sensor devices.
  - Data Pre-processing: Manipulating collected datasets and maliciously altering them before they get fed to the ML model.
  - ML model: Poisoning the model by creating adversarial examples of data to make misclassifications. Can be in both training and testing.
  - Output domain: Maliciously tampering with output devices, screens to display incorrect results/predictions.



# ML Security

- According to NIST, there are multiple Targets of Attack (TA) in an ML pipeline, for CPPS [2]:
  - Input domain: Malicious tampering with input sensor data or sensor devices.
  - Data Pre-processing: Manipulating collected datasets and maliciously altering them before they get fed to the ML model.
  - ML model: Poisoning the model by creating adversarial examples of data to make misclassifications. Can be in both training and testing.
  - Output domain: Maliciously tampering with output devices, screens to display incorrect results/predictions.



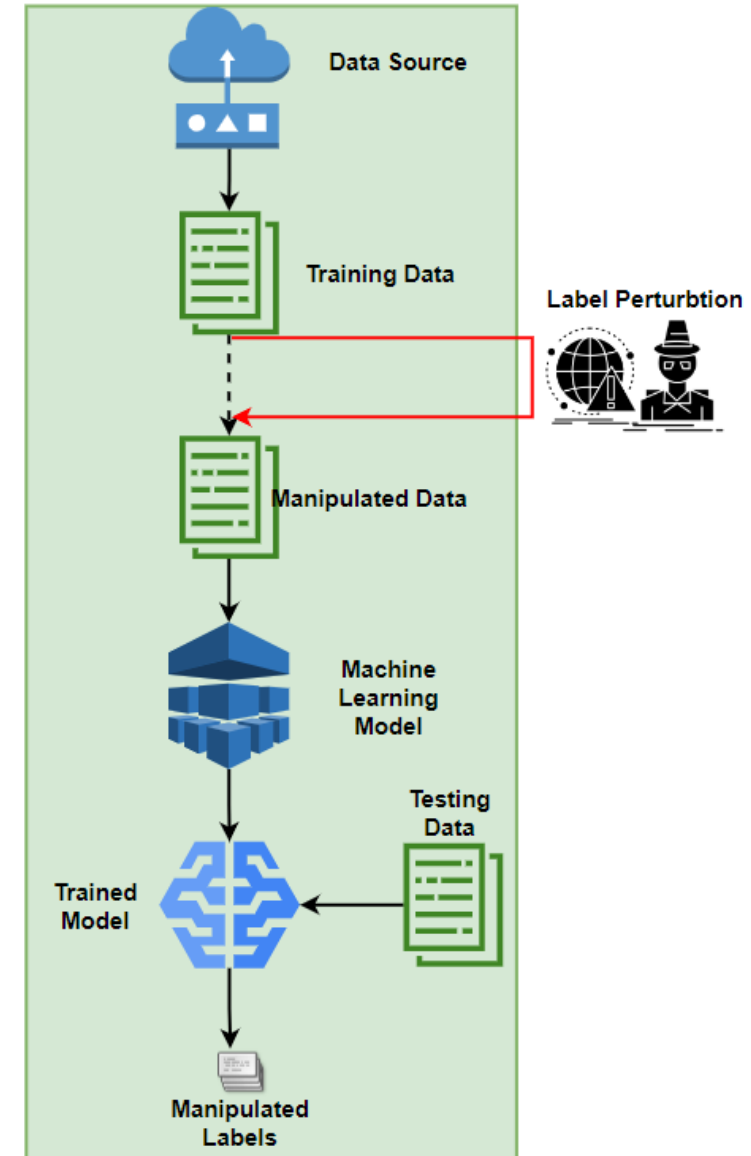
# ML Security

- According to NIST, there are multiple Targets of Attack (TA) in an ML pipeline, for CPPS [2]:
  - Input domain: Malicious tampering with input sensor data or sensor devices.
  - Data Pre-processing: Manipulating collected datasets and maliciously altering them before they get fed to the ML model.
  - ML model: Poisoning the model by creating adversarial examples of data to make misclassifications. Can be in both training and testing.
  - Output domain: Maliciously tampering with output devices, screens to display incorrect results/predictions.



# Adversarial Attacks

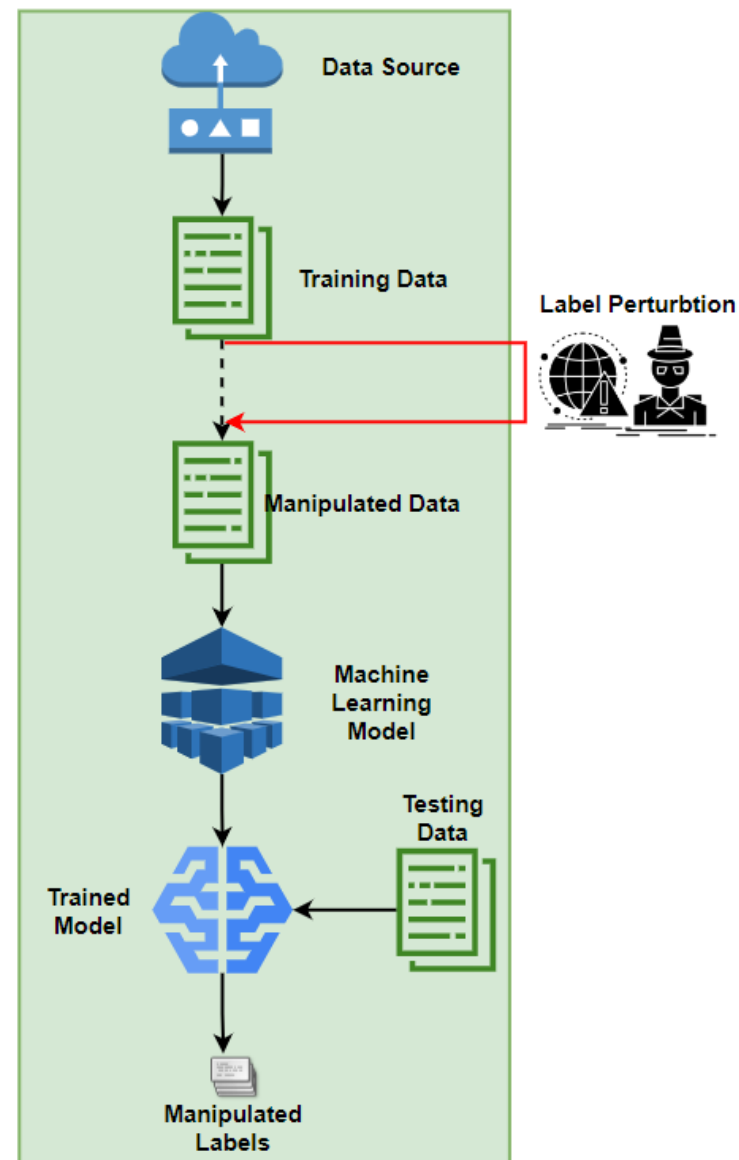
- Cyber attacks on ML pipelines that aim to manipulate data and exploit model sensitivities to affect ML performance.
- Goal: Trigger misclassification, mispredictions, confidence reduction, attack model/data
- In CPPS, adversarial attacks can be detrimental as they can put the public and the environment at risk.



# Adversarial Attack Examples

---

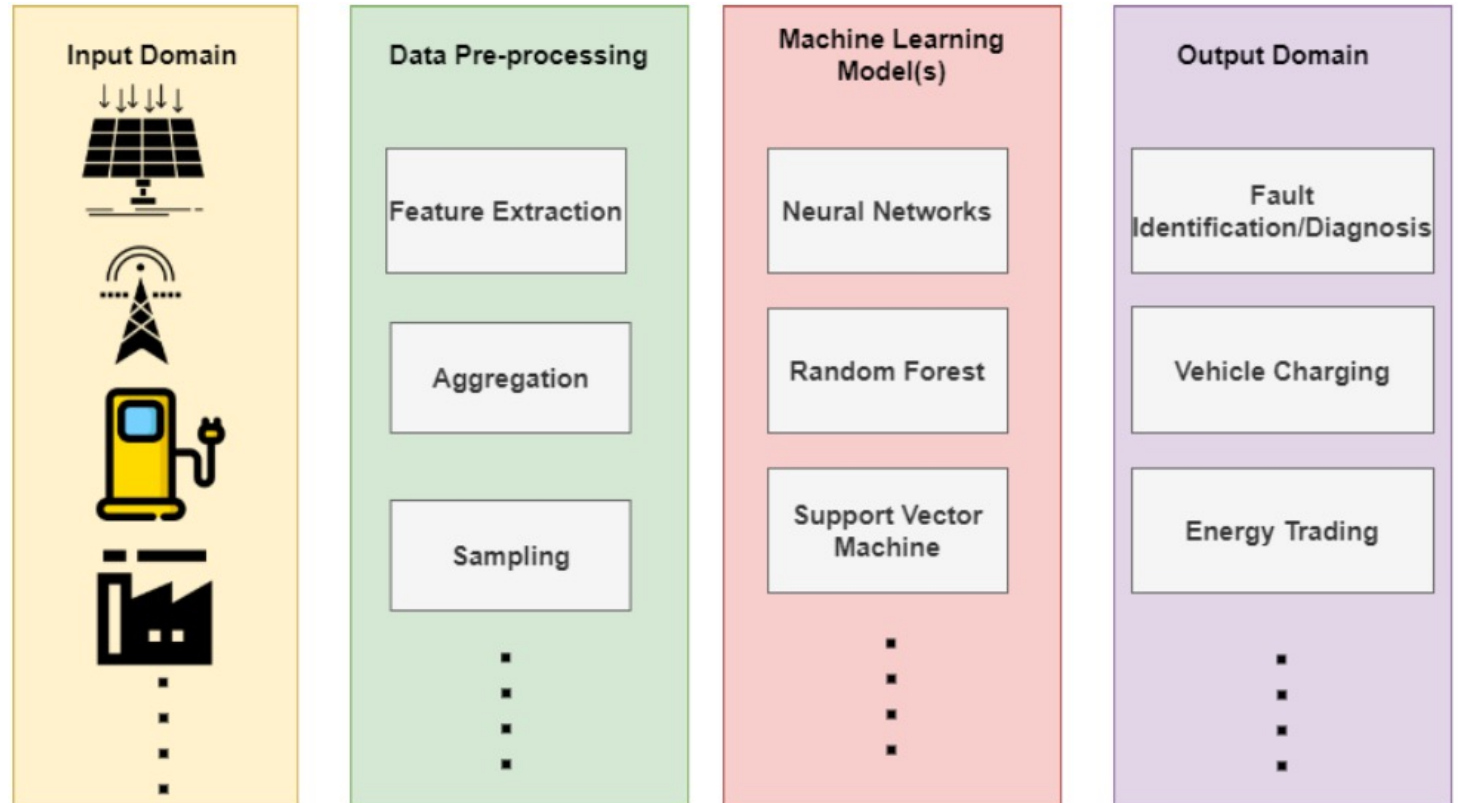
- Data Access attacks
- Data/Model poisoning
- Evasion attacks
- Oracle attacks



# Vulnerabilities in ML algorithms

---

- The four main TAs in an ML pipeline is:
  - Input domain
    - Training Phase
    - Testing Phase
  - Output domain




# Input Domain

- Targets: Transformers, temperature sensors, load leveling sensors, etc.
- Common attacks:
  - **False data injection:**
    - Malicious, yet believable, samples are introduced into data to make ML classifier malfunction.
    - Optimized, making them difficult to detect.
  - **Data Manipulation:**
    - Manipulating the existing training data/labels to corrupt ML classification performance.
    - Difficult to protect against due to the innate nature of manipulation.
  - **Physical device tampering**





# Input Domain

- Targets: Transformers, temperature sensors, load leveling sensors, etc.
- Common attacks:
  - **False data injection:** 
    - Malicious, yet believable, samples are introduced into data to make ML classifier malfunction.
    - Optimized, making them difficult to detect.
  - **Data Manipulation:**
    - Manipulating the existing training data/labels to corrupt ML classification performance.
    - Difficult to protect against due to the innate nature of manipulation.
  - **Physical device tampering**



# Input Domain

- Targets: Transformers, temperature sensors, load leveling sensors, etc.
- Common attacks:
  - **False data injection:**
    - Malicious, yet believable, samples are introduced into data to make ML classifier malfunction.
    - Optimized, making them difficult to detect.
  - **Data Manipulation:** ←
    - Manipulating the existing training data/labels to corrupt ML classification performance.
    - Difficult to protect against due to the innate nature of manipulation.
  - **Physical device tampering**



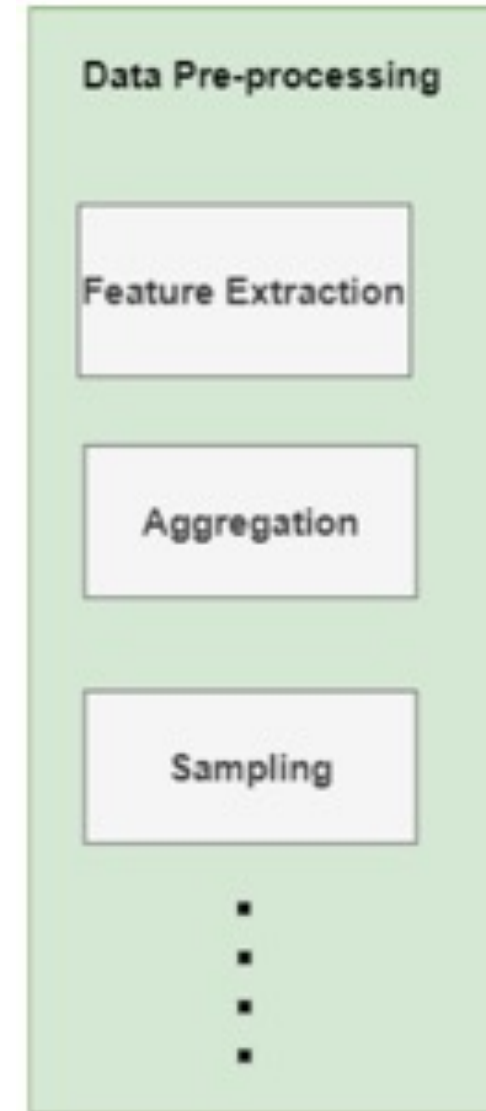
# Input Domain

- Targets: Transformers, temperature sensors, load leveling sensors, etc.
- Common attacks:
  - **False data injection:**
    - Malicious, yet believable, samples are introduced into data to make ML classifier malfunction.
    - Optimized, making them difficult to detect.
  - **Data Manipulation:**
    - Manipulating the existing training data/labels to corrupt ML classification performance.
    - Difficult to protect against due to the innate nature of manipulation.
  - **Physical device tampering** ←



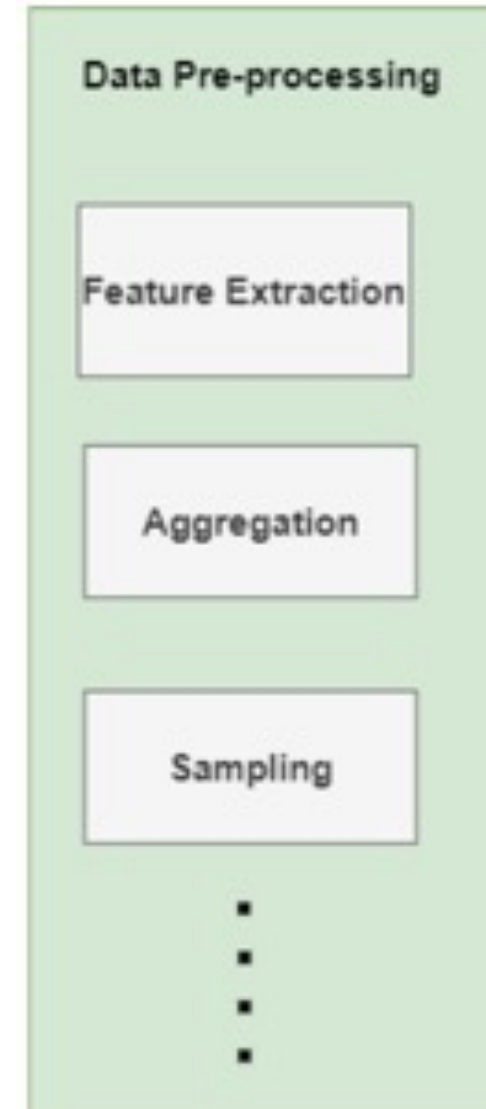
# Data Pre-processing

- Aggregated data from sensors must be pre-processed before continuing.
- Pre-processing techniques: noise removal, frequency-time analysis, dimensionality reduction, sampling, and feature extraction.
- Attacks:
  - **Noise manipulation:** Manipulate noise removal algorithms to add more noise to the data.
  - **Feature extraction:** Introduce malicious magnitudes in data parameters to degrade feature extraction algorithms.
  - **Visual data tampering:** Data captured through CPPS network cameras can be modified through image scaling attacks.



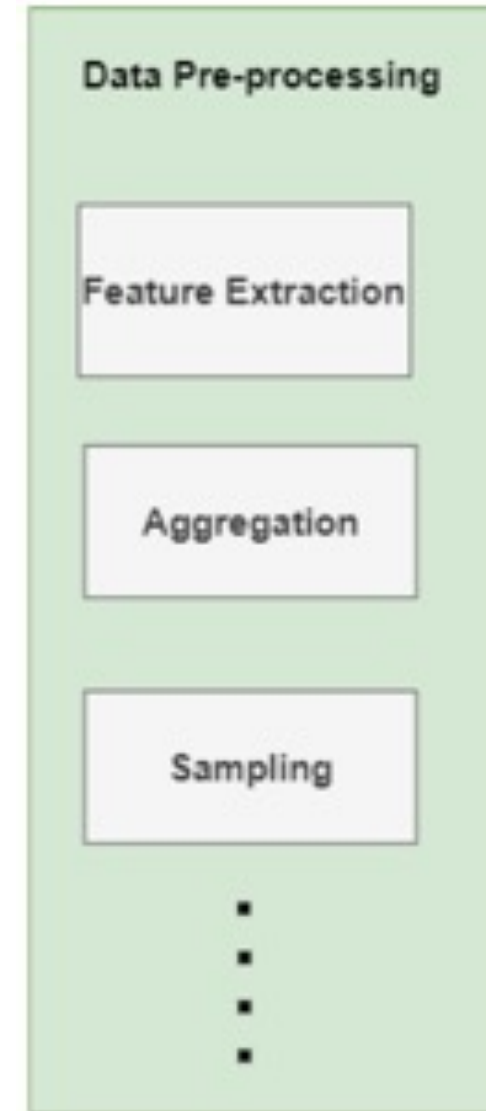
# Data Pre-processing

- Aggregated data from sensors must be pre-processed before continuing.
- Pre-processing techniques: noise removal, frequency-time analysis, dimensionality reduction, sampling, and feature extraction.
- Attacks:
  - • **Noise manipulation:** Manipulate noise removal algorithms to add more noise to the data.
  - **Feature extraction:** Introduce malicious magnitudes in data parameters to degrade feature extraction algorithms.
  - **Visual data tampering:** Data captured through CPPS network cameras can be modified through image scaling attacks.



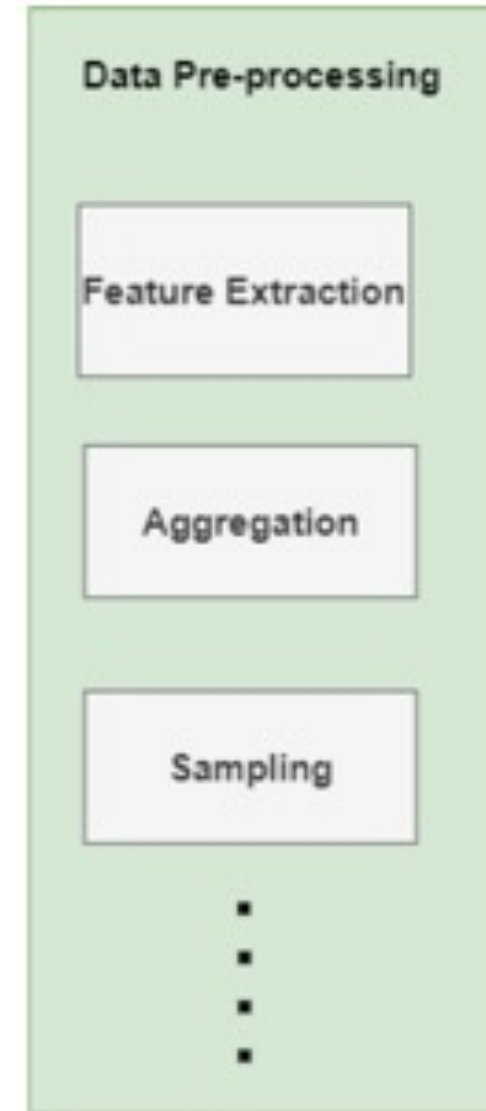
# Data Pre-processing

- Aggregated data from sensors must be pre-processed before continuing.
- Pre-processing techniques: noise removal, frequency-time analysis, dimensionality reduction, sampling, and feature extraction.
- Attacks:
  - **Noise manipulation:** Manipulate noise removal algorithms to add more noise to the data.
  - **Feature extraction:** Introduce malicious magnitudes in data parameters to degrade feature extraction algorithms.
  - **Visual data tampering:** Data captured through CPPS network cameras can be modified through image scaling attacks.



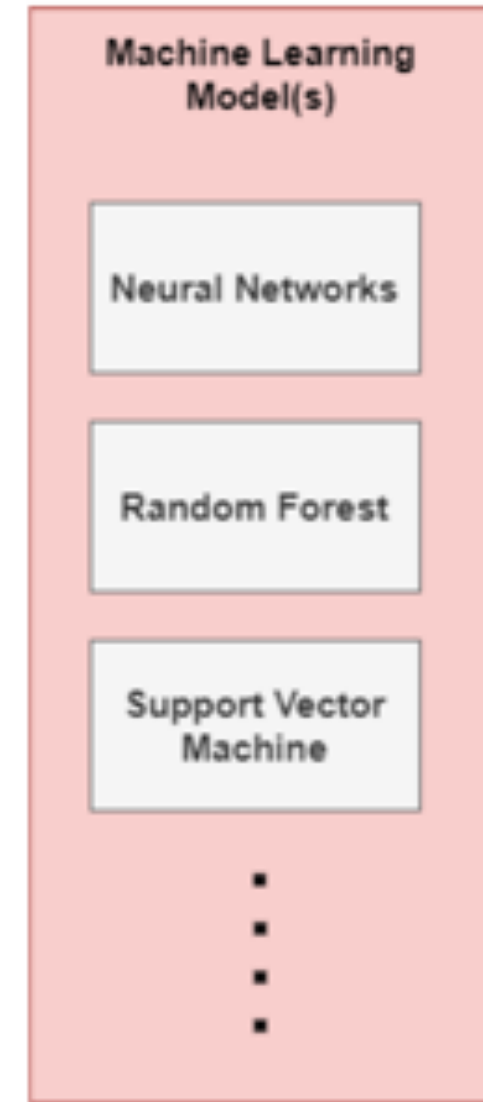
# Data Pre-processing

- Aggregated data from sensors must be pre-processed before continuing.
- Pre-processing techniques: noise removal, frequency-time analysis, dimensionality reduction, sampling, and feature extraction.
- Attacks:
  - **Noise manipulation:** Manipulate noise removal algorithms to add more noise to the data.
  - **Feature extraction:** Introduce malicious magnitudes in data parameters to degrade feature extraction algorithms.
  - **Visual data tampering:** Data captured through CPPS network cameras can be modified through image scaling attacks.



# ML Models – Training Phase

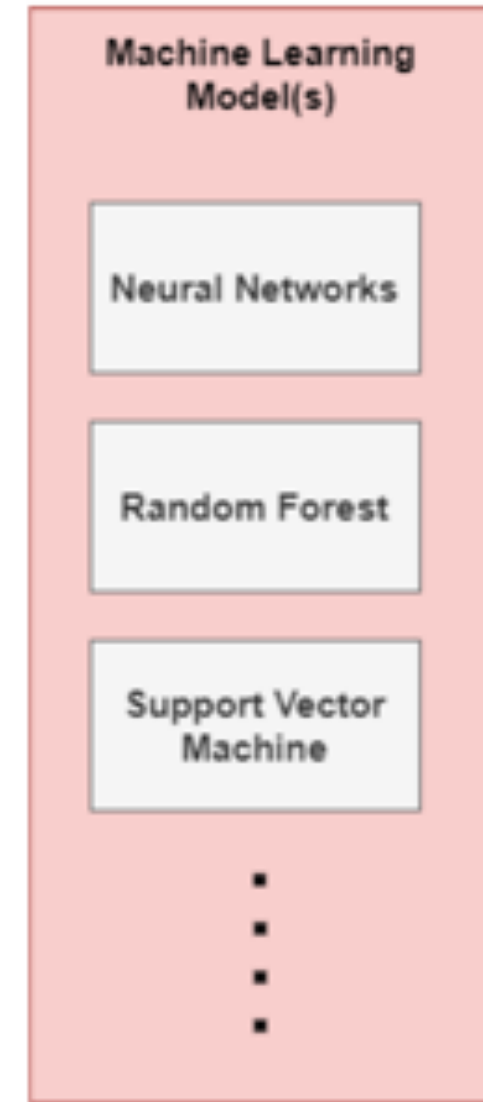
- Training phase attacks are geared to influence the training data or model. Most influential TA.
- **Poisoning attacks:**
  - **Data injection:** The adversary has access to the data. Optimized adversarial inputs are introduced into the original data to trigger misclassifications.
  - **Data manipulation:** The adversary has access to the data. They poison the training data by flipping the input data (Input manipulation), or labels (Label manipulation)
  - **Logic Corruption:** The adversary has access to the model. They can change the learning process and model parameters. Very hard to create counter strategies for this.
- **Data access attacks:** A subset or all the training data is illegally accessed
  - Generate a substitute model that can be used to evaluate the effectiveness of potential inputs before submitting them to testing attacks.





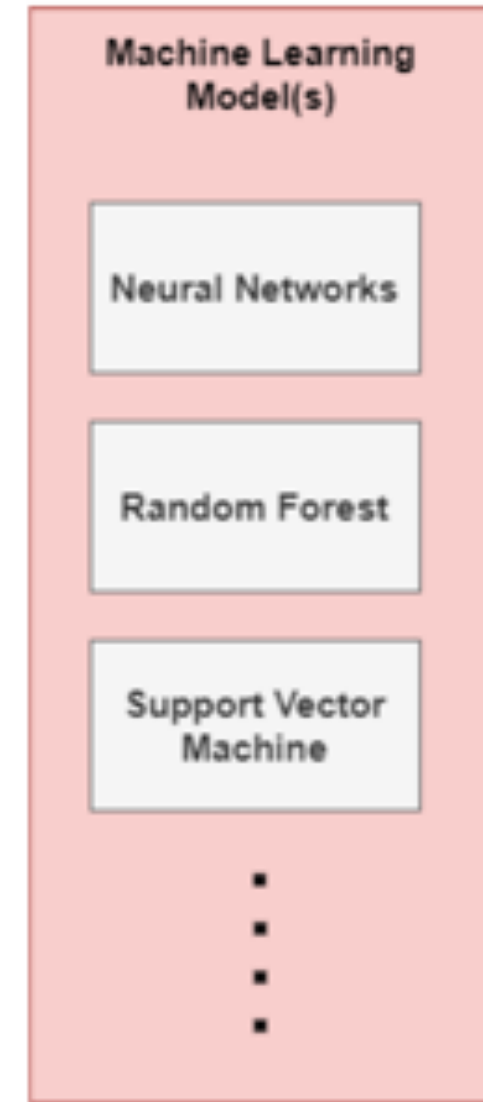
# ML Models – Training Phase

- Training phase attacks are geared to influence the training data or model. Most influential TA.
- **Poisoning attacks:**
  - **Data injection:** The adversary has access to the data. Optimized adversarial inputs are introduced into the original data to trigger misclassifications.
  - **Data manipulation:** The adversary has access to the data. They poison the training data by flipping the input data (Input manipulation), or labels (Label manipulation)
  - **Logic Corruption:** The adversary has access to the model. They can change the learning process and model parameters. Very hard to create counter strategies for this.
- **Data access attacks:** A subset or all the training data is illegally accessed
  - Generate a substitute model that can be used to evaluate the effectiveness of potential inputs before submitting them to testing attacks.



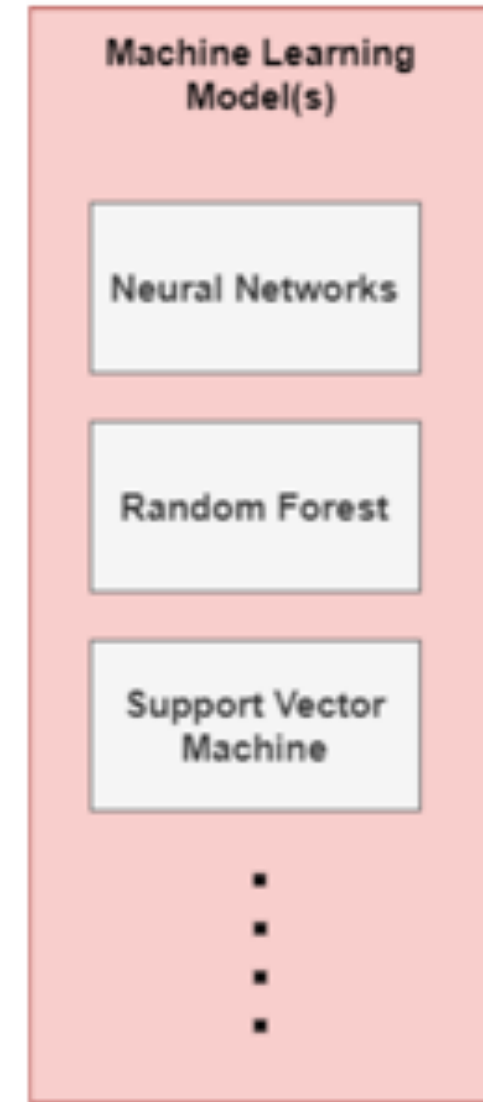
# ML Models – Training Phase

- Training phase attacks are geared to influence the training data or model. Most influential TA.
- **Poisoning attacks:**
  - **Data injection:** The adversary has access to the data. Optimized adversarial inputs are introduced into the original data to trigger misclassifications.
  - **Data manipulation:** The adversary has access to the data. They poison the training data by flipping the input data (Input manipulation), or labels (Label manipulation)
  - **Logic Corruption:** The adversary has access to the model. They can change the learning process and model parameters. Very hard to create counter strategies for this.
- **Data access attacks:** A subset or all the training data is illegally accessed
  - Generate a substitute model that can be used to evaluate the effectiveness of potential inputs before submitting them to testing attacks.



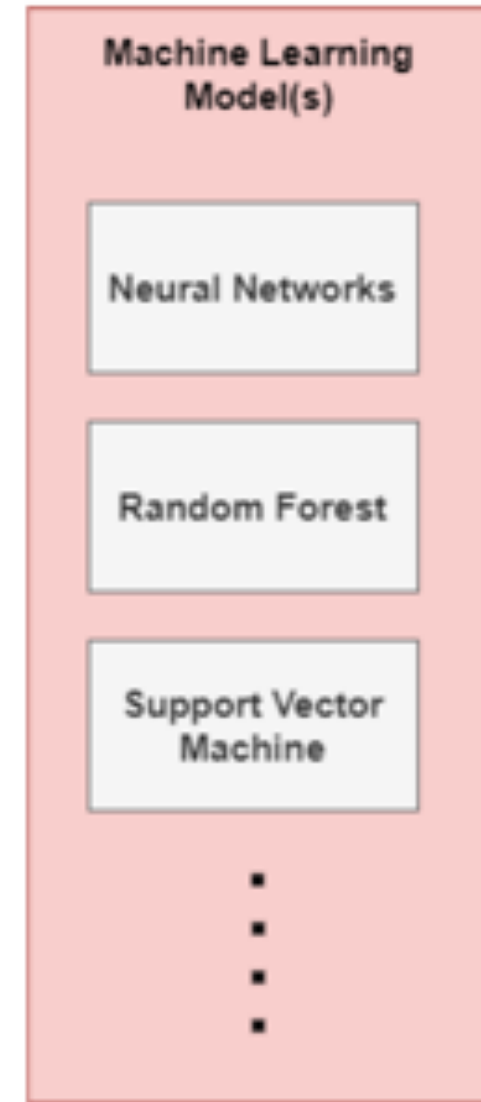
# ML Models – Training Phase

- Training phase attacks are geared to influence the training data or model. Most influential TA.
- **Poisoning attacks:**
  - **Data injection:** The adversary has access to the data. Optimized adversarial inputs are introduced into the original data to trigger misclassifications.
  - **Data manipulation:** The adversary has access to the data. They poison the training data by flipping the input data (Input manipulation), or labels (Label manipulation)
  - **Logic Corruption:** The adversary has access to the model. They can change the learning process and model parameters. Very hard to create counter strategies for this.
- **Data access attacks:** A subset or all the training data is illegally accessed
  - Generate a substitute model that can be used to evaluate the effectiveness of potential inputs before submitting them to testing attacks.



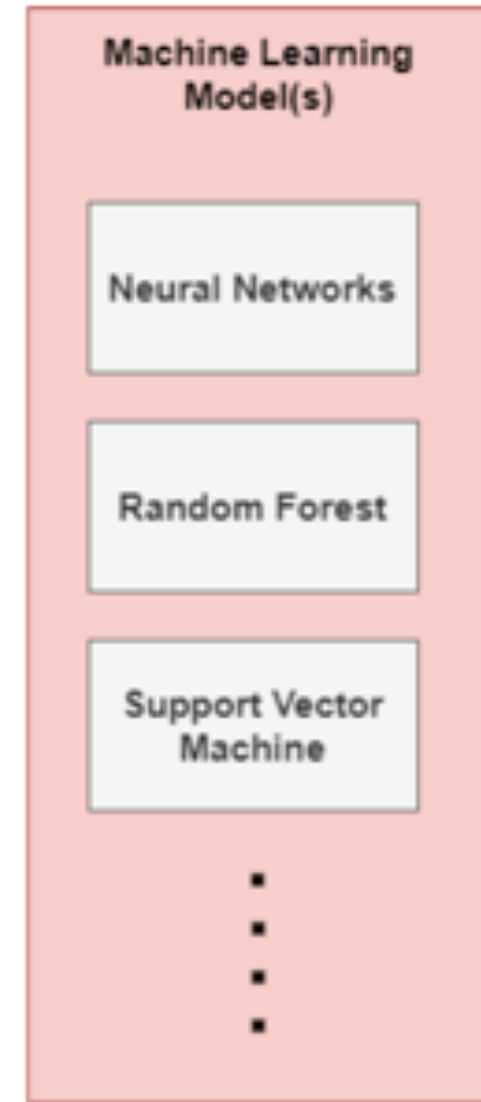
# ML Models – Training Phase

- Training phase attacks are geared to influence the training data or model. Most influential TA.
- **Poisoning attacks:**
  - **Data injection:** The adversary has access to the data. Optimized adversarial inputs are introduced into the original data to trigger misclassifications.
  - **Data manipulation:** The adversary has access to the data. They poison the training data by flipping the input data (Input manipulation), or labels (Label manipulation)
  - **Logic Corruption:** The adversary has access to the model. They can change the learning process and model parameters. Very hard to create counter strategies for this.
- **Data access attacks:** A subset or all the training data is illegally accessed
  - Generate a substitute model that can be used to evaluate the effectiveness of potential inputs before submitting them to testing attacks.



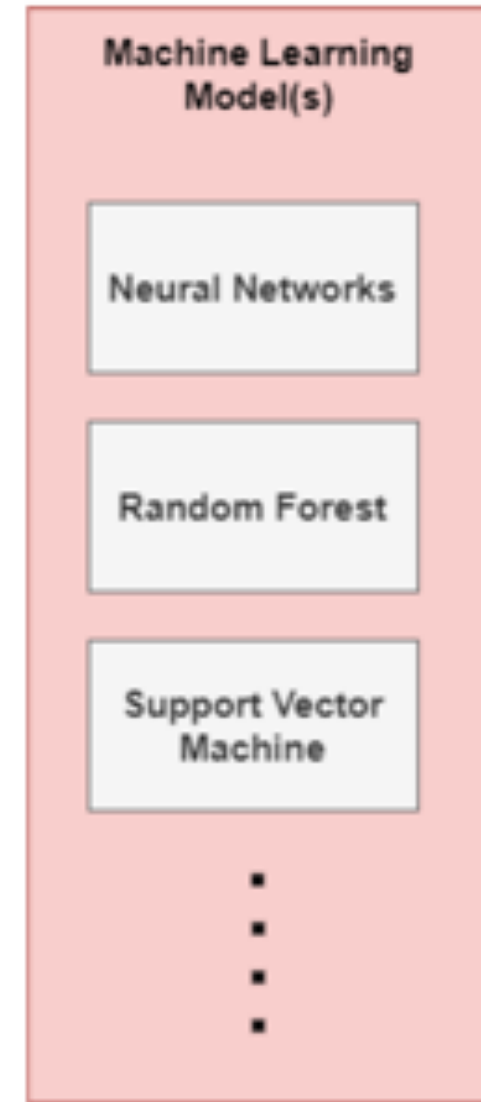
# ML Models – Testing Phase

- Testing phase attacks are exploratory attacks that perform no tampering.
  - Goal is to evade proper classification by the model or collect information about the model or training data
- **Evasion:** The main goal is to solve an optimization problem to find a small input perturbation that creates a large change in loss, to trigger misclassification.
- **Oracle:** A malicious API is used by adversaries to present the model with inputs and to observe the outputs. These are then used to train a substitute model using the input-output mapping.



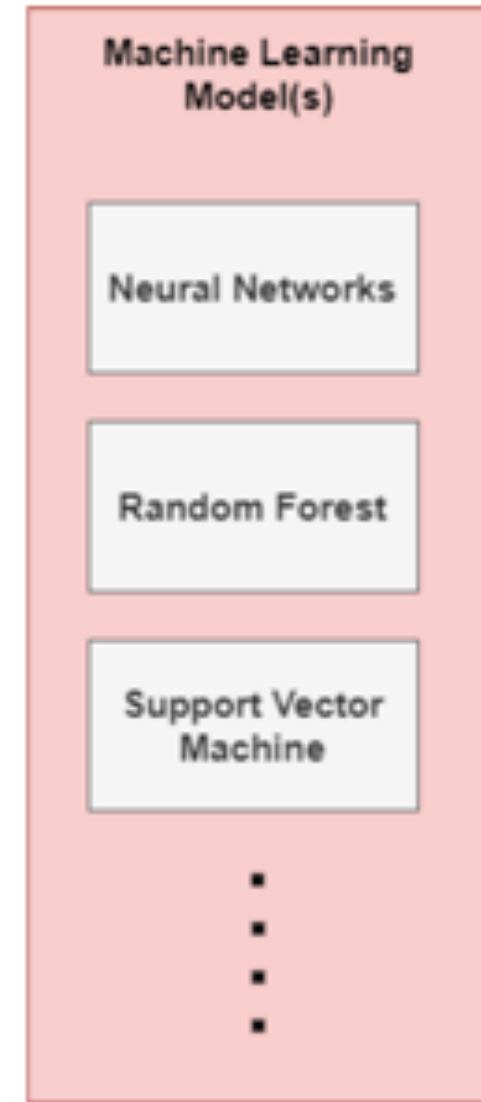
# ML Models – Testing Phase

- Testing phase attacks are exploratory attacks that perform no tampering.
  - Goal is to evade proper classification by the model or collect information about the model or training data
- • **Evasion:** The main goal is to solve an optimization problem to find a small input perturbation that creates a large change in loss, to trigger misclassification.
- **Oracle:** A malicious API is used by adversaries to present the model with inputs and to observe the outputs. These are then used to train a substitute model using the input-output mapping.



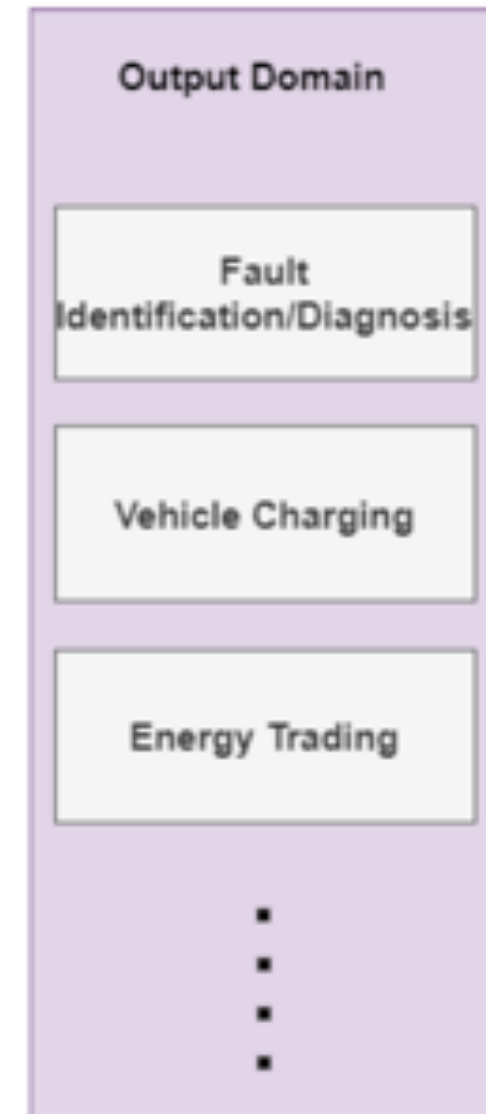
# ML Models – Testing Phase

- Testing phase attacks are exploratory attacks that perform no tampering.
  - Goal is to evade proper classification by the model or collect information about the model or training data
- **Evasion:** The main goal is to solve an optimization problem to find a small input perturbation that creates a large change in loss, to trigger misclassification.
- • **Oracle:** A malicious API is used by adversaries to present the model with inputs and to observe the outputs. These are then used to train a substitute model using the input-output mapping.



# Output Domain

- Includes the output indicators and displays that provide predictions. Least influential TA.
- Uses in CPPS:
  - Forecasting electricity demands
  - Electric transmission ampacity
  - Peak electric load
- Attacks:
  - **Label Manipulation:** Flipping the label at the output device
  - **Regression Manipulation:** Maliciously altering the predicted regression value at the output device
- Can be even more problematic if these ML pipelines are daisy-chained.





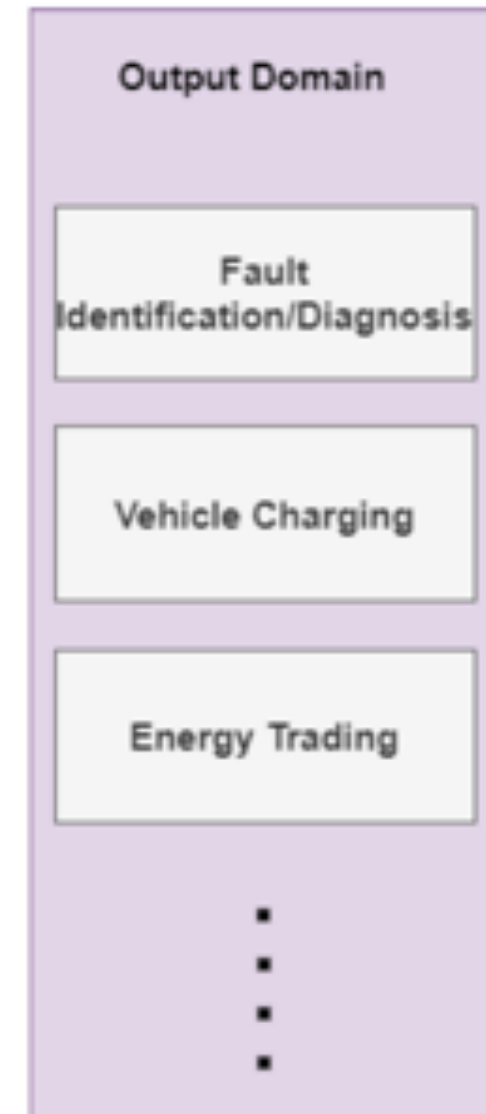
# Output Domain

- Includes the output indicators and displays that provide predictions. Least influential TA.
- Uses in CPPS:
  - Forecasting electricity demands
  - Electric transmission ampacity
  - Peak electric load
- Attacks:
  - • **Label Manipulation:** Flipping the label at the output device
  - **Regression Manipulation:** Maliciously altering the predicted regression value at the output device
- Can be even more problematic if these ML pipelines are daisy-chained.

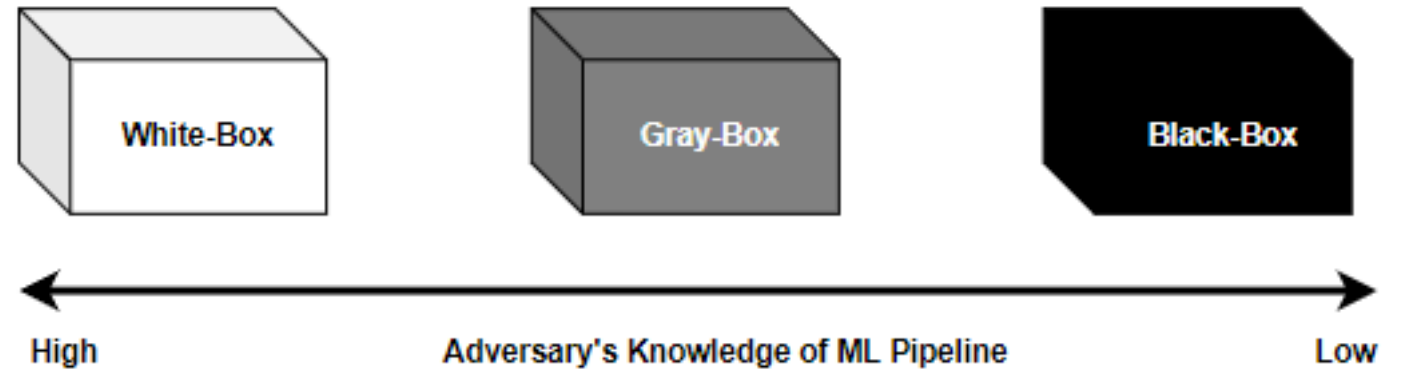


# Output Domain

- Includes the output indicators and displays that provide predictions. Least influential TA.
- Uses in CPPS:
  - Forecasting electricity demands
  - Electric transmission ampacity
  - Peak electric load
- Attacks:
  - **Label Manipulation:** Flipping the label at the output device
  - **Regression Manipulation:** Maliciously altering the predicted regression value at the output device
- Can be even more problematic if these ML pipelines are daisy-chained.

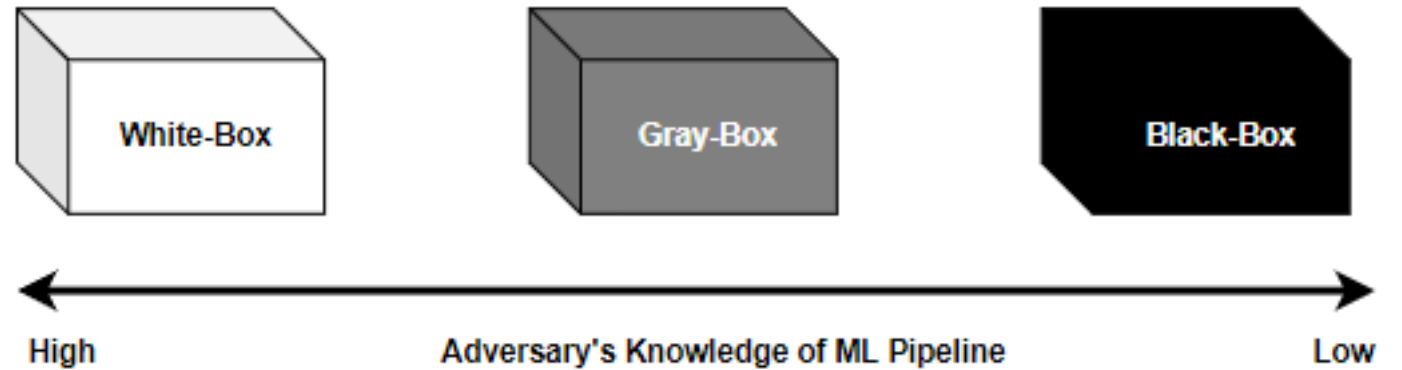


# Attacks Models



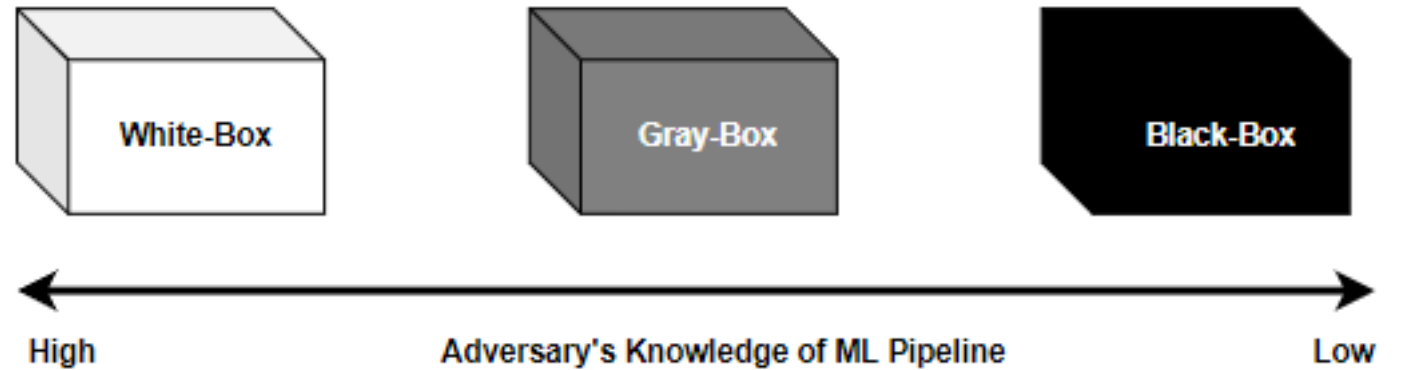
- Adversarial attacks are generated with multiple objectives in mind.
  - **Confidence reduction:** Reduce confidence in a prediction for a target model.
  - **Mis-classification:** Alter output classification to degrade performance.
  - **Target attacks:** Manipulate an input data sample to make the ML classifier predict it as a particular desired class
  - **Non-targeted attacks:** Manipulate an input data sample to make the ML classifier predict it as any other class than the actual class.
- Types of attack models: White box, Gray box, Black box

# Attacks Models



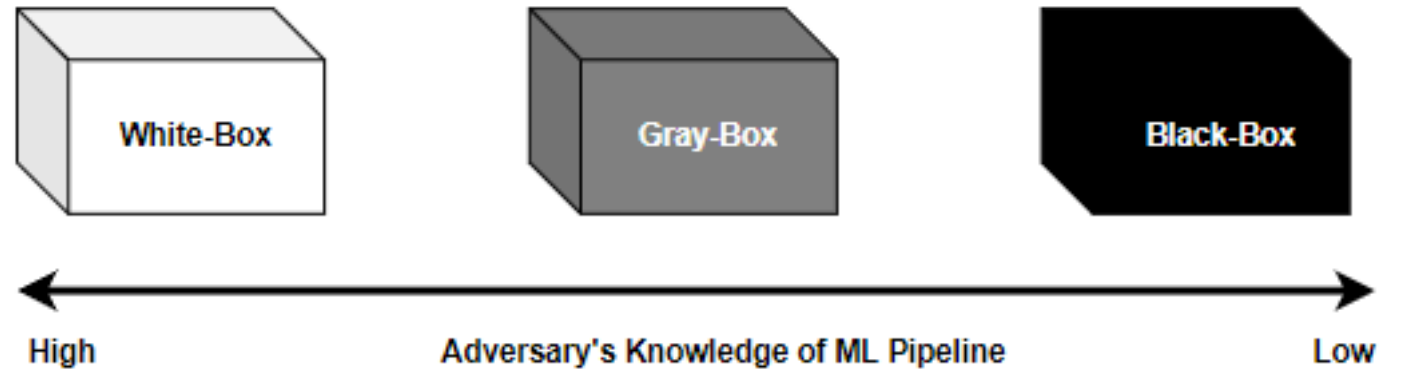
- Adversarial attacks are generated with multiple objectives in mind.
- ➔ • **Confidence reduction:** Reduce confidence in a prediction for a target model.
- **Mis-classification:** Alter output classification to degrade performance.
- **Target attacks:** Manipulate an input data sample to make the ML classifier predict it as a particular desired class
- **Non-targeted attacks:** Manipulate an input data sample to make the ML classifier predict it as any other class than the actual class.
- Types of attack models: White box, Gray box, Black box

# Attacks Models



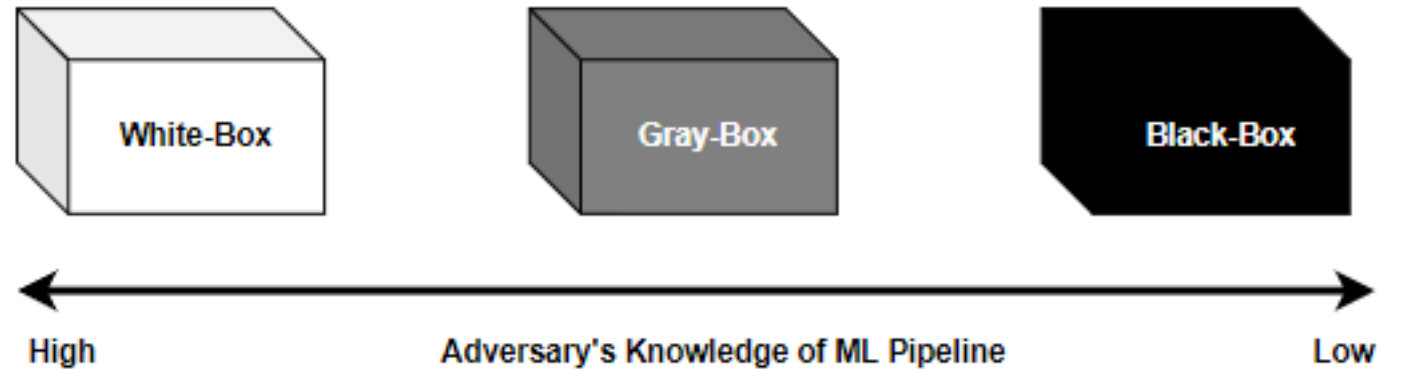
- Adversarial attacks are generated with multiple objectives in mind.
  - **Confidence reduction:** Reduce confidence in a prediction for a target model.
  - ➔ **Mis-classification:** Alter output classification to degrade performance.
  - **Target attacks:** Manipulate an input data sample to make the ML classifier predict it as a particular desired class
  - **Non-targeted attacks:** Manipulate an input data sample to make the ML classifier predict it as any other class than the actual class.
- Types of attack models: White box, Gray box, Black box

# Attacks Models



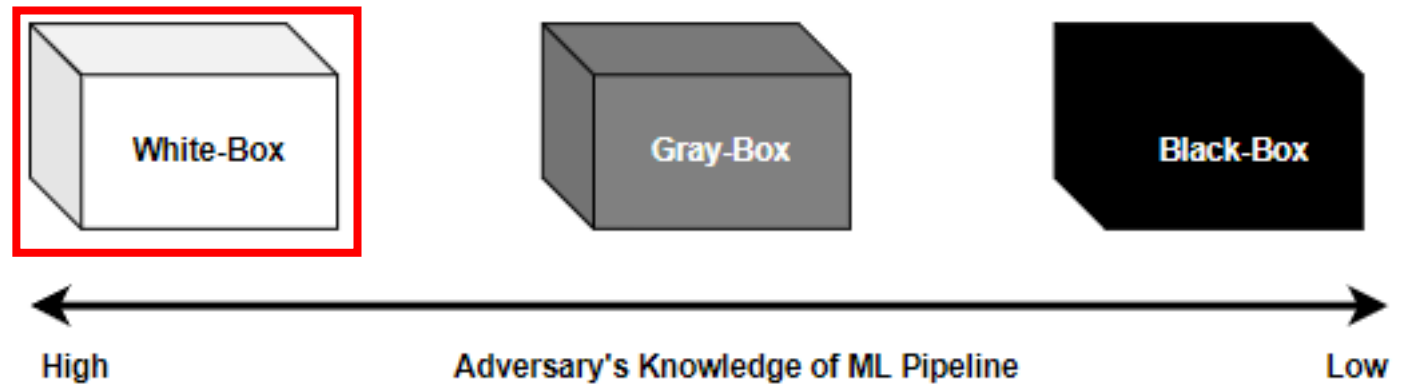
- Adversarial attacks are generated with multiple objectives in mind.
  - **Confidence reduction:** Reduce confidence in a prediction for a target model.
  - **Mis-classification:** Alter output classification to degrade performance.
  - ➔ • **Target attacks:** Manipulate an input data sample to make the ML classifier predict it as a particular desired class
  - **Non-targeted attacks:** Manipulate an input data sample to make the ML classifier predict it as any other class than the actual class.
- Types of attack models: White box, Gray box, Black box

# Attacks Models



- Adversarial attacks are generated with multiple objectives in mind.
  - **Confidence reduction:** Reduce confidence in a prediction for a target model.
  - **Mis-classification:** Alter output classification to degrade performance.
  - **Target attacks:** Manipulate an input data sample to make the ML classifier predict it as a particular desired class
  - **Non-targeted attacks:** Manipulate an input data sample to make the ML classifier predict it as any other class than the actual class.
- Types of attack models: White box, Gray box, Black box

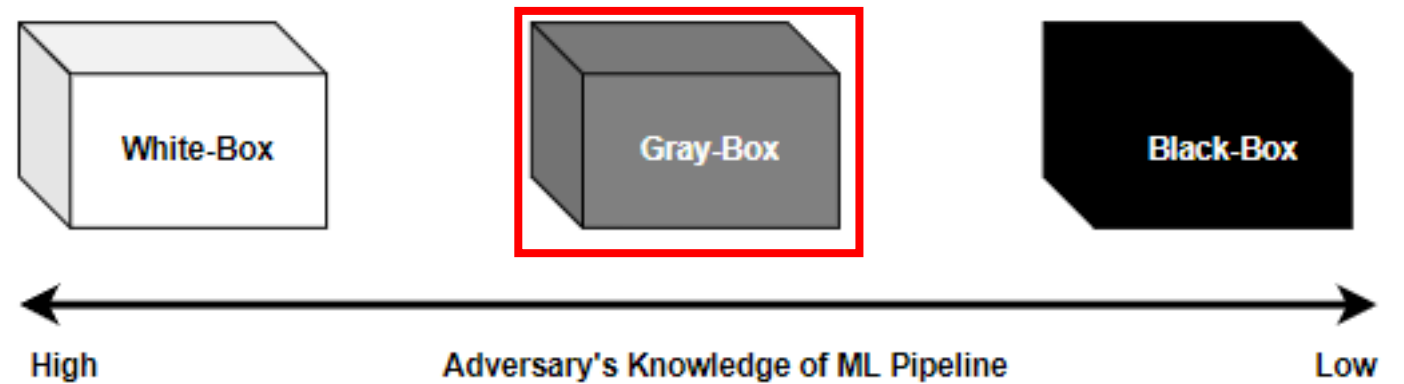
# White Box



- Adversary has complete knowledge of an ML classifier like training algorithm, optimization approach, input data, and distribution
- Adversaries can use this information to analyze the vulnerabilities of the ML pipeline and attack at a point where they will maximize their error rate.
- Very hard to defend against due to the adversary having complete knowledge
- Impractical

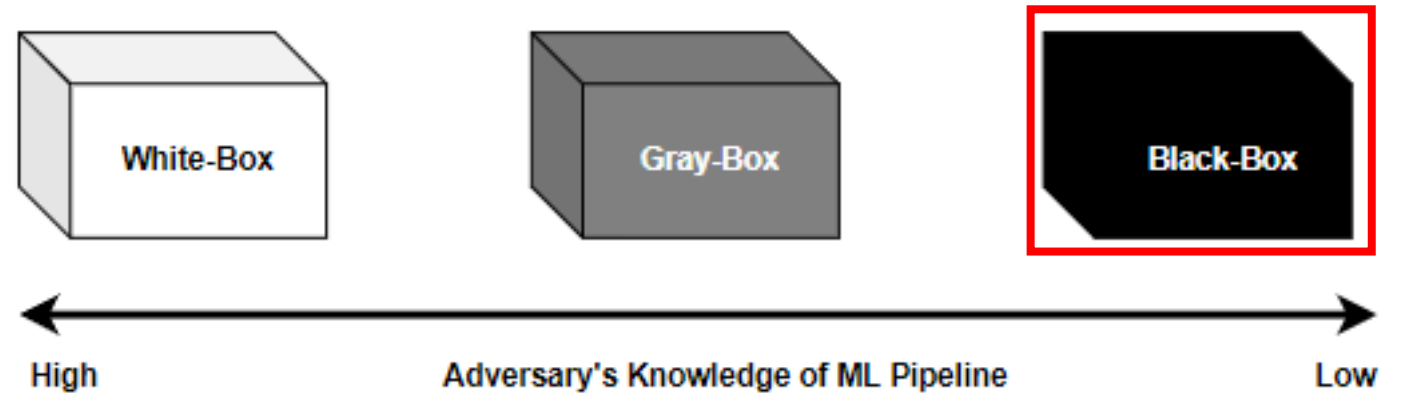


# Gray Box



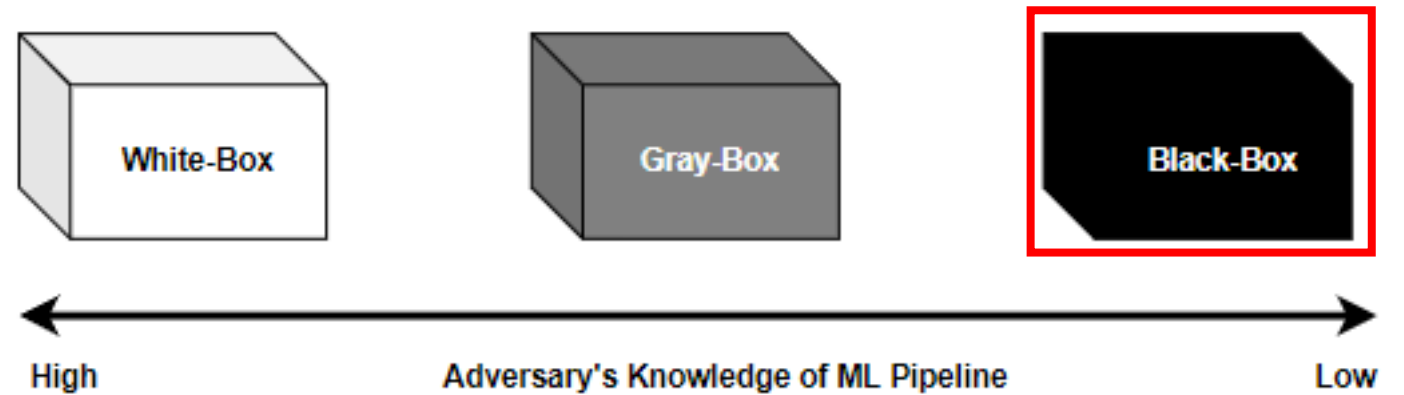
- Attack model where the adversary has partial or limited knowledge of the ML pipeline.
- Can have access to model architecture, model parameters, magnitudes, training methods, loss functions, and data distributions, but not all.
- More practical for adversarial situations than white box attacks.

# Black Box



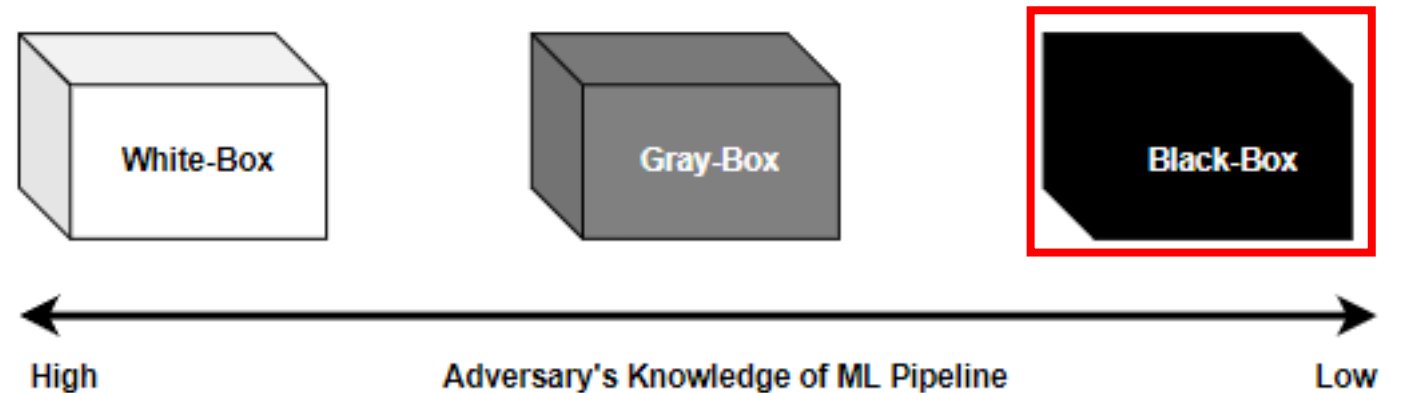
- Attack model where the adversary has no knowledge about the internal functionality of the ML pipeline.
- Utilizes information about settings, inputs, and outputs to attack.
- **Non-adaptive attack:** The adversary can only access training data distribution, to train a local model. Then, approximate the model to the target model to learn its parameters.

# Black Box



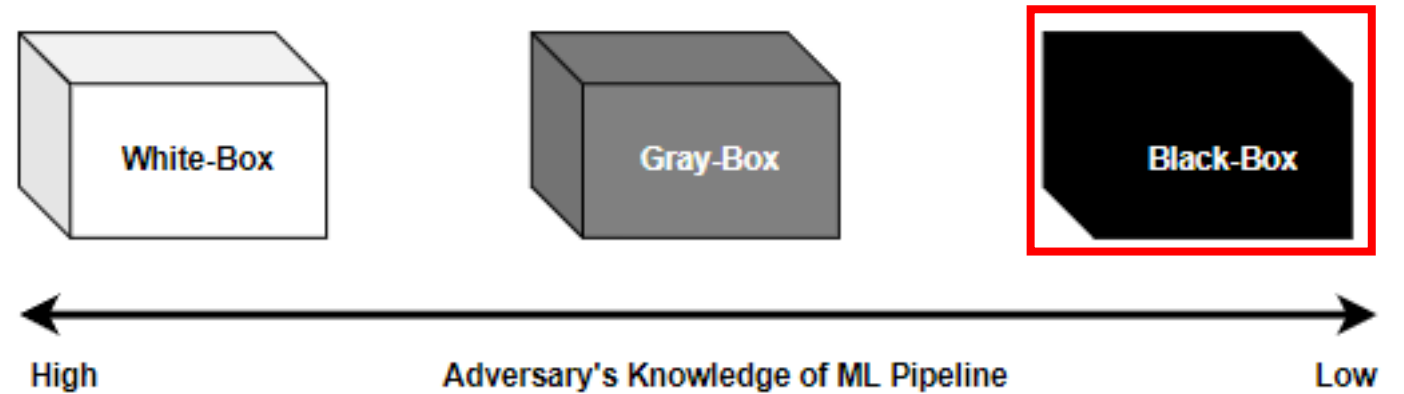
- Attack model where the adversary has no knowledge about the internal functionality of the ML pipeline.
- Utilizes information about settings, inputs, and outputs to attack.
- **Non-adaptive attack:** The adversary can only access training data distribution, to train a local model. Then, approximate the model to the target model to learn its parameters.

# Black Box – Cont.



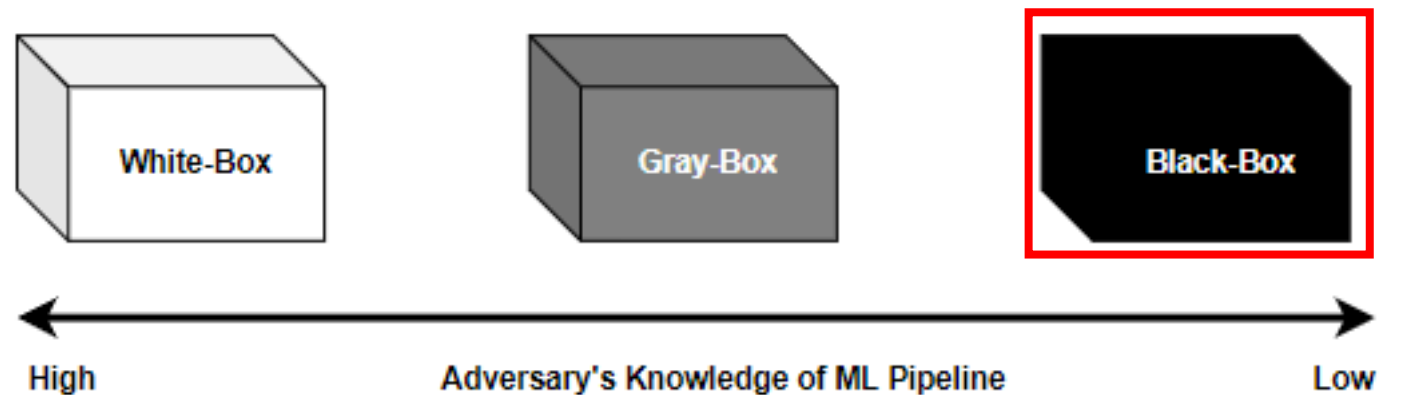
- **Adaptive attack:** The adversary can access an unknown trained model and feed random and adaptive input samples to label a carefully selected dataset. Thereby gaining knowledge of the internal functionality of the unknown trained model.
- **Strict attack:** The adversary does not possess data distribution but can collect input-output pairs to get model mapping. They can't change the input like an adaptive attack.
- Most practical attack model

# Black Box – Cont.



- **Adaptive attack:** The adversary can access an unknown trained model and feed random and adaptive input samples to label a carefully selected dataset. Thereby gaining knowledge of the internal functionality of the unknown trained model.
- **Strict attack:** The adversary does not possess data distribution but can collect input-output pairs to get model mapping. They can't change the input like an adaptive attack.
- Most practical attack model

# Black Box – Cont.

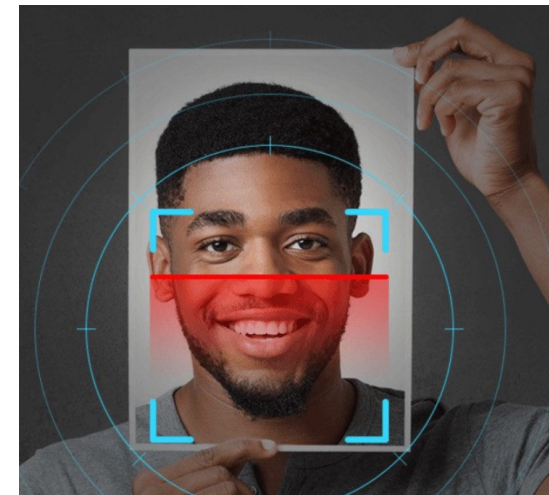


- **Adaptive attack:** The adversary can access an unknown trained model and feed random and adaptive input samples to label a carefully selected dataset. Thereby gaining knowledge of the internal functionality of the unknown trained model.
- **Strict attack:** The adversary does not possess data distribution but can collect input-output pairs to get model mapping. They can't change the input like an adaptive attack.
- Most practical attack model

# Practical Examples of Adversarial Attacks

Reliable sign  
detection  
(Autonomous  
vehicles)

Facial recognition  
systems

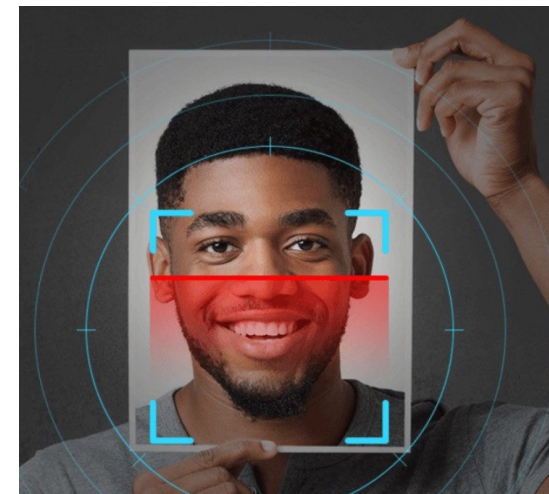


Retrieved from [3]

# Practical Examples of Adversarial Attacks

Reliable sign  
detection  
(Autonomous  
vehicles)

Facial recognition  
systems



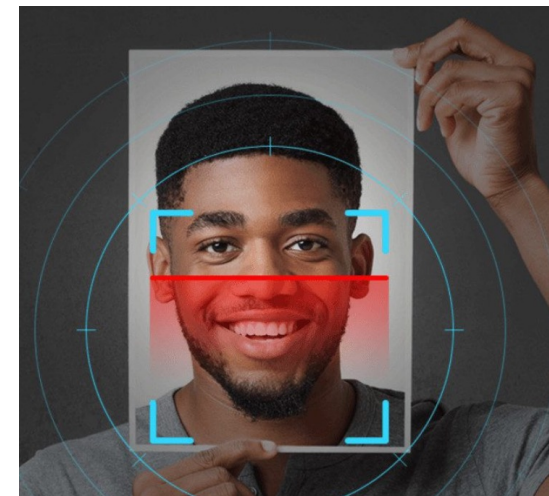
Retrieved from [3]



# Practical Examples of Adversarial Attacks

Reliable sign  
detection  
(Autonomous  
vehicles)

Facial recognition  
systems



Retrieved from [3]

# Reliable Sign Detection



# Examples of Adversarial Application in CPPS

---

- In CPPS Surveillance:
  - Adversaries can manipulate CPPS surveillance sensors (temperature, humidity, state estimation) using injections or physical tampering.
  - Introduce perturbations that make the ML classifier not treat it as an abnormal state when the recovered system state is different from the actual state
- In Electric Vehicle charging
  - Adversaries can damage charging stations or install malware through the USB ports, which can cause data theft or a Denial-of-Service in EV charging.
  - Ex: Adversary can gain access to vulnerable charging stations by disabling the charger, causing a ripple effect on the rest of the power grid.

# Protection against Adversarial Attacks

---

- Adversarial attacks are difficult to detect due to:
  - **Adversarial Examples Crafting Model:** Creating adversarial samples involves complex optimization. Due to the unknown nature of this optimization, defensive mechanisms are typically ineffectual.
  - **ML models require input-output pairs:** Modifying ML models to be resilient against adversarial attacks can change the objective of the model, hampering reliability, performance, and usefulness.

# Current Protection Solutions

---

- **Adversarial Training:** Model robustness is increased by injecting adversarial examples into the training data. The target model is trained with both real and perturbed samples.
- **Gradient hiding:** To protect against attacks that target gradient optimization, like FGSM, essential information about the model's gradient can be hidden or encrypted as a countermeasure.
- **Defensive distillation:** An ML model gets trained to classify samples as either “hard” or “soft” labels. The “soft” outputs get sent to another ML classifier as input, trained on the same input samples. The second “distilled” model provides a smoother output, more robust to adversarial attacks.

# Current Protection Solutions

---

- **Adversarial Training:** Model robustness is increased by injecting adversarial examples into the training data. The target model is trained with both real and perturbed samples.
- **Gradient hiding:** To protect against attacks that target gradient optimization, like FGSM, essential information about the model's gradient can be hidden or encrypted as a countermeasure.
- **Defensive distillation:** An ML model gets trained to classify samples as either "hard" or "soft" labels. The "soft" outputs get sent to another ML classifier as input, trained on the same input samples. The second "distilled" model provides a smoother output, more robust to adversarial attacks.

# Current Protection Solutions

---

- **Adversarial Training:** Model robustness is increased by injecting adversarial examples into the training data. The target model is trained with both real and perturbed samples.
- **Gradient hiding:** To protect against attacks that target gradient optimization, like FGSM, essential information about the model's gradient can be hidden or encrypted as a countermeasure.
- **Defensive distillation:** An ML model gets trained to classify samples as either “hard” or “soft” labels. The “soft” outputs get sent to another ML classifier as input, trained on the same input samples. The second “distilled” model provides a smoother output, more robust to adversarial attacks.

# Current Protection Solutions

---

- **Adversarial Training:** Model robustness is increased by injecting adversarial examples into the training data. The target model is trained with both real and perturbed samples.
- **Gradient hiding:** To protect against attacks that target gradient optimization, like FGSM, essential information about the model's gradient can be hidden or encrypted as a countermeasure.
- **Defensive distillation:** An ML model gets trained to classify samples as either “hard” or “soft” labels. The “soft” outputs get sent to another ML classifier as input, trained on the same input samples. The second “distilled” model provides a smoother output, more robust to adversarial attacks.



# Current Protection Solutions

---

- **Feature Squeezing:** Model hardening technique to reduce the complexity of representing the data, to make adversarial perturbations disappear due to low sensitivity.
- **Defense-GAN:** Provide protection by using a Generative Adversarial Network (GAN) to reduce the efficiency of adversarial perturbations
- **MagNet:** Using ML classifiers as a black box to read the output of the classifier's last layer, without modifying the classifier. Then, use detectors to discern between normal and adversarial samples.

# Current Protection Solutions

---

- • **Feature Squeezing:** Model hardening technique to reduce the complexity of representing the data, to make adversarial perturbations disappear due to low sensitivity.
- **Defense-GAN:** Provide protection by using a Generative Adversarial Network (GAN) to reduce the efficiency of adversarial perturbations
- **MagNet:** Using ML classifiers as a black box to read the output of the classifier's last layer, without modifying the classifier. Then, use detectors to discern between normal and adversarial samples.

# Current Protection Solutions

---

- **Feature Squeezing:** Model hardening technique to reduce the complexity of representing the data, to make adversarial perturbations disappear due to low sensitivity.
- • **Defense-GAN:** Provide protection by using a Generative Adversarial Network (GAN) to reduce the efficiency of adversarial perturbations
- **MagNet:** Using ML classifiers as a black box to read the output of the classifier's last layer, without modifying the classifier. Then, use detectors to discern between normal and adversarial samples.

# Current Protection Solutions

---

- **Feature Squeezing:** Model hardening technique to reduce the complexity of representing the data, to make adversarial perturbations disappear due to low sensitivity.
- **Defense-GAN:** Provide protection by using a Generative Adversarial Network (GAN) to reduce the efficiency of adversarial perturbations
- **MagNet:** Using ML classifiers as a black box to read the output of the classifier's last layer, without modifying the classifier. Then, use detectors to discern between normal and adversarial samples.

# Conclusion

---

- In this presentation, we have illustrated:
  - Vulnerabilities that exist within ML algorithms in CPPS.
  - Adversarial attacks and how they can manipulate the vulnerabilities.
  - Different TAs that exist in the ML pipelines and how they can be compromised using various attacks.
  - Various attack models for adversarial attacks
  - Real-life examples of conducting adversarial attacks
  - Current protection strategies that exist.
- Defense against adversarial attacks is still a wide-open research area.
- Current methods show promising signs but can still be circumvented by adversaries and put our critical CPPS at risk.

# References

- [1] Wu, G., Li, Z.S. Cyber—Physical Power System (CPPS): A review on measures and optimization methods of system resilience. *Front. Eng. Manag.* **8**, 503–518 (2021). <https://doi.org/10.1007/s42524-021-0163-3>
- [2] Tabassi, Elham, et al. "A taxonomy and terminology of adversarial machine learning." *NIST IR* (2019): 1-29.
- [3] <https://mobidev.biz/wp-content/uploads/2020/01/anti-spoofing-techniques-face-recognition-1.jpg>

Thank you!  
Questions?

---

