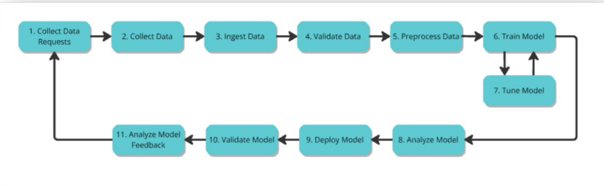
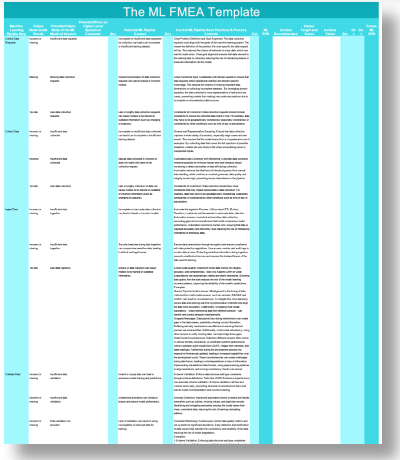
INTRODUCING THE ML FMEA

Paul Schmitt, Torc Robotics
Bodo Seifert, TÜV Rheinland
Jerry Lopez, Torc Robotics
Mario Bijelic, Torc Robotics
Krzysztof Pennar, Torc Robotics
Felix Heide, Torc Robotics

Link to the paper

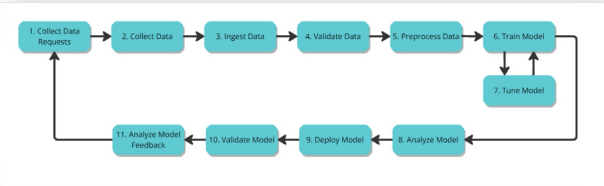
Executive Summary



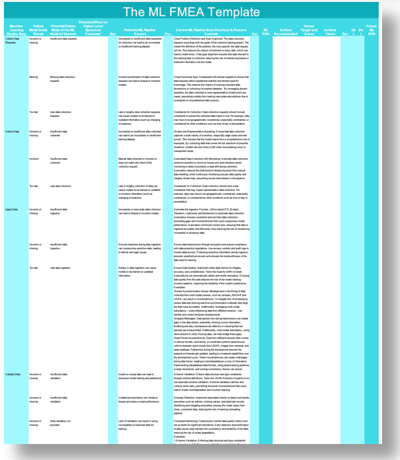

The ML FMEA Template

- Raise awareness of a new approach to identify, prioritize, and communicate machine learning risk and safety.

Executive Summary



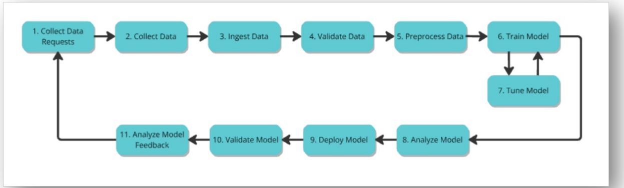
1. Collect Data Requests
2. Collect Data
3. Ingest Data
4. Validate Data
5. Preprocess Data
6. Train Model
7. Tune Model
8. Analyze Model
9. Deploy Model
10. Validate Model
11. Analyze Model Feedback



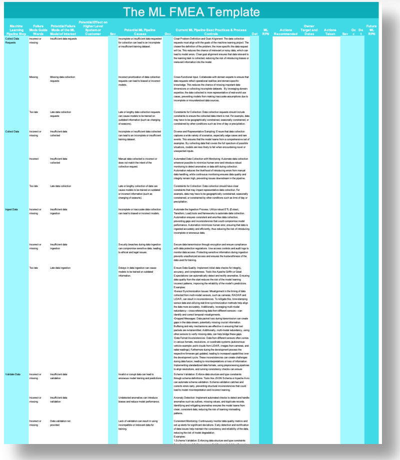
The ML FMEA Template

- Raise awareness of a new approach to identify, prioritize, and communicate machine learning risk and safety.
- Empowers development teams while increasing risk transparency to safety reviewers.

Executive Summary



1. Collect Data Requests
2. Collect Data
3. Ingest Data
4. Validate Data
5. Preprocess Data
6. Train Model
7. Tune Model
8. Analyze Model
9. Deploy Model
10. Validate Model
11. Analyze Model Feedback

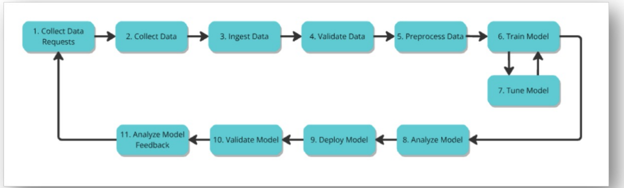


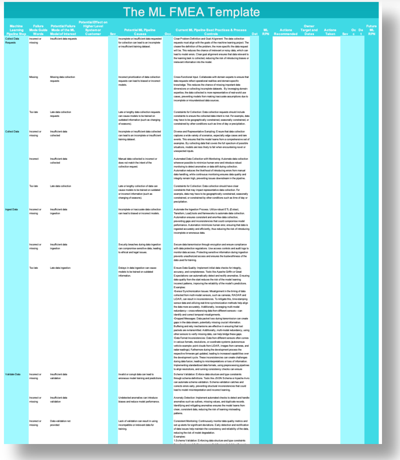
The ML FMEA Template

Machine Learning Pipeline Step	Failure Mode Guide Words	Potential Failure Mode of the ML Model of Interest	Potential Effect on Higher Level System or Customer	Sev	Potential ML Pipeline Causes	Occ	Current ML Pipeline Best Practices & Process Controls	Det	ML RPN	Actions Recommended	Owner Target and Dates	Actions Taken	Sev	Occ	Det	Future ML RPN
Collect Data Requests	Incorrect or missing	Insufficient data requests			Incomplete or insufficient data requested for collection can lead to an incomplete or insufficient training dataset.		Clear Problem Definition and Goal Alignment: The data collection requests must align with the goals of the machine learning project. The clearer the definition of the problem, the more specific the data request will be. This reduces the chance of irrelevant or noisy data, which can lead to model errors. Clear goal alignment ensures that data relevant to the learning task is collected, reducing the risk of introducing biases or irrelevant information into the model.									
	Missing	Missing data collection requests			Incorrect prioritization of data collection requests can lead to biased or incorrect models.		Cross-Functional Input. Collaborate with domain experts to ensure that data requests reflect operational realities and domain-specific knowledge. This reduces the chance of missing important data dimensions or collecting incomplete datasets. By leveraging domain expertise, the data collected is more representative of real-world use cases, preventing models from making inaccurate assumptions due to incomplete or misunderstood data sources.									
	Too late	Late data collection requests			Late or lengthy data collection requests can cause models to be trained on outdated information (such as changing of seasons).		Constraints for Collection: Data collection requests should include constraints to ensure the collected data intent is met. For example, data may have to be geographically constrained, seasonally constrained, or constrained by other conditions such as time of day or precipitation.									
Collect Data	Incorrect or missing	Insufficient data collected			Incomplete or insufficient data collected can lead to an incomplete or insufficient training dataset.		Diverse and Representative Sampling: Ensure that data collection captures a wide variety of scenarios, especially edge cases and rare events. This ensures that the model learns from a comprehensive set of examples. By collecting data that covers the full spectrum of possible situations, models are less likely to fail when encountering novel or unexpected inputs.									
	Incorrect	Insufficient data collected			Manual data collected is incorrect or does not match the intent of the collection request.		Automated Data Collection with Monitoring: Automate data collection wherever possible to minimize human error and introduce robust monitoring to detect anomalies or data drift during collection. Automation reduces the likelihood of introducing errors from manual data handling, while continuous monitoring ensures data quality and integrity remain high, preventing issues downstream in the pipeline.									
	Too late	Late data collection			Late or lengthy collection of data can cause models to be trained on outdated or incorrect information (such as changing of seasons).		Constraints for Collection: Data collection should have clear constraints that may impact representative data collection. For example, data may have to be geographically constrained, seasonally constrained, or constrained by other conditions such as time of day or precipitation.									
Ingest Data	Incorrect or missing	Insufficient data ingestion			Incomplete or inaccurate data collection can lead to biased or incorrect models.		Automate the Ingestion Process. Utilize robust ETL (Extract, Transform, Load) tools and frameworks to automate data collection. Automation ensures consistent and error-free data collection, preventing gaps and inconsistencies that could compromise model performance. Automation minimizes human error, ensuring that data is ingested accurately and efficiently, thus reducing the risk of introducing incomplete or erroneous data.									
	Incorrect or missing	Insufficient data ingestion			Security breaches during data ingestion can compromise sensitive data, leading to ethical and legal issues.		Secure data transmission through encryption and ensure compliance with data protection regulations. Use access controls and audit logs to monitor data access. Protecting sensitive information during ingestion prevents unauthorized access and ensures the trustworthiness of the data used for training.									
	Too late	Late data ingestion			Delays in data ingestion can cause models to be trained on outdated information.		Ensure Data Quality: Implement initial data checks for integrity, accuracy, and completeness. Tools like Apache Griffin or Great Expectations can automatically detect and rectify anomalies. Ensuring data quality from the start reduces the risk of the model learning incorrect patterns, improving the reliability of the model's predictions. Examples: •Sensor Synchronization Issues: Misalignment in the timing of data collected from multi-modal sensors, such as cameras, RADAR and LiDAR, can result in inconsistencies. To mitigate this, time-stamping sensor data and utilizing real-time synchronization methods help align the data more accurately. Additionally, leveraging multi-modal redundancy—cross-referencing data from different sensors—can identify and correct temporal misalignments. •Dropped Messages: Data packet loss during transmission can create gaps in the data stream, potentially missing crucial information. Buffering and retry mechanisms are effective in ensuring that lost packets are re-transmitted. Additionally, multi-modal redundancy, using other sensors to verify missing data, can help bridge these gaps. •Data Format Inconsistencies: Data from different sensors often comes in various formats, resolutions, or coordinate systems (autonomous vehicle example: point clouds from LiDAR, images from cameras, and radar readings). Furthermore during the development process the respective firmware get updated, leading to increased capabilities over the development cycle. These inconsistencies can create challenges during data fusion, leading to misinterpretations or loss of information. Implementing standardized data formats, using preprocessing pipelines to align resolutions, and running consistency checks can ensure									
Validate Data	Incorrect or missing	Insufficient data validation			Invalid or corrupt data can lead to erroneous model training and predictions.		Schema Validation: Enforce data structure and type constraints through schema definitions. Tools like JSON Schema or Apache Avro can automate schema validation. Schema validation catches and corrects errors early, preventing structural inconsistencies that could lead to model misinterpretation and incorrect learning.									
	Incorrect or missing	Insufficient data validation			Undetected anomalies can introduce biases and reduce model performance.		Anomaly Detection: Implement automated checks to detect and handle anomalies such as outliers, missing values, and duplicate records. Identifying and mitigating anomalies ensures the model learns from clean, consistent data, reducing the risk of learning misleading patterns.									
	Incorrect or missing	Data validation not provided			Lack of validation can result in using incompatible or irrelevant data for training.		Consistent Monitoring: Continuously monitor data quality metrics and set up alerts for significant deviations. Early detection and rectification of data issues help maintain the consistency and reliability of the data, reducing the risk of model degradation. Examples: 1.Schema Validation: Enforcing data structure and type constraints									

- Raise awareness of a new approach to identify, prioritize, and communicate machine learning risk and safety.
- Empowers development teams while increasing risk transparency to safety reviewers.
- We see it as a key component of an overall safety case.

Executive Summary




The ML FMEA Template

- Raise awareness of a new approach to identify, prioritize, and communicate machine learning risk and safety.
- Empowers development teams while increasing risk transparency to safety reviewers.
- We see it as a key component of an overall safety case. Across industries:
  - Automotive, ADAS, Autonomous Vehicles
  - Military, Aerospace
  - Medical, Pharmaceutical, …

Agenda

Motivation

Laying the Foundation

ML FMEA Method

ML FMEA Template

Link to the paper

Agenda

Motivation

Laying the Foundation

ML FMEA Method

ML FMEA Template

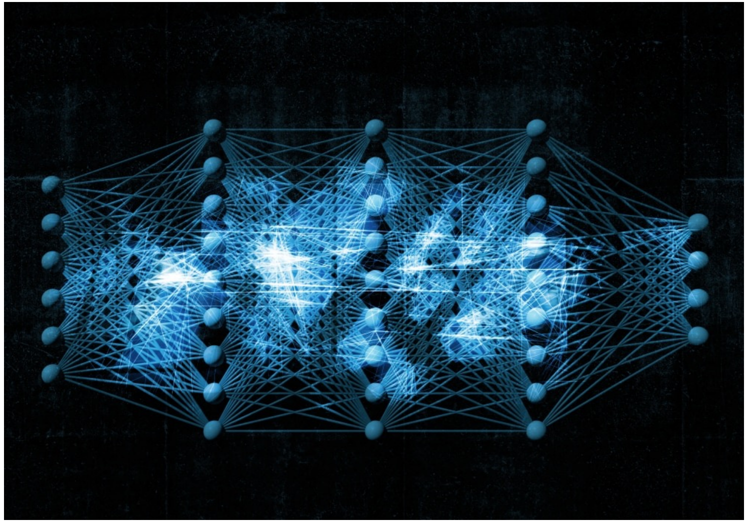
Open Source!

Link to the paper

MOTIVATION

Link to the paper

Motivation



- ML is non-deterministic and opaque
- Challenges traditional safety analyses

State of the Art

Technical Report

9:2024

Publicly Available Specification

ISO/PAS 8800:2024

Road vehicles — Safety and artificial intelligence

Edition 1
2024-12

Reference number
ISO/PAS 8800:2024

© ISO 2025

Several current and upcoming standards:

- ISO 5469
- ISO PAS 8800
- UL 4600
- ISO 21448

Highlight the need to systematically ensure safety of ML

State of the Art

Opportunity Gap: Specific techniques and methodologies

State of the Art

Opportunity Gap: Specific techniques and methodologies

- Holistic: Addresses all ML engineering and V&V activities

State of the Art

Opportunity Gap: Specific techniques and methodologies

- Holistic: Addresses all ML engineering and V&V activities
- Indicates the needed level of rigor

State of the Art

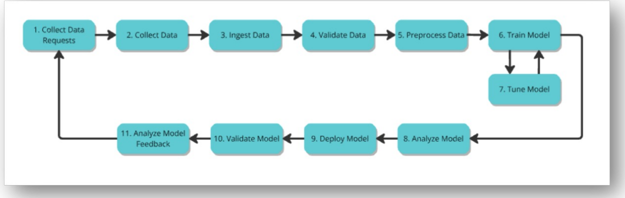
Opportunity Gap: Specific techniques and methodologies

- Holistic: Addresses all ML engineering and V&V activities
- Indicates the needed level of rigor
- Supports a safety argument
  - Provides sufficient evidence
  - Supports the argument that ML algorithm is absent unreasonable risk

CONTRIBUTION

Link to the paper

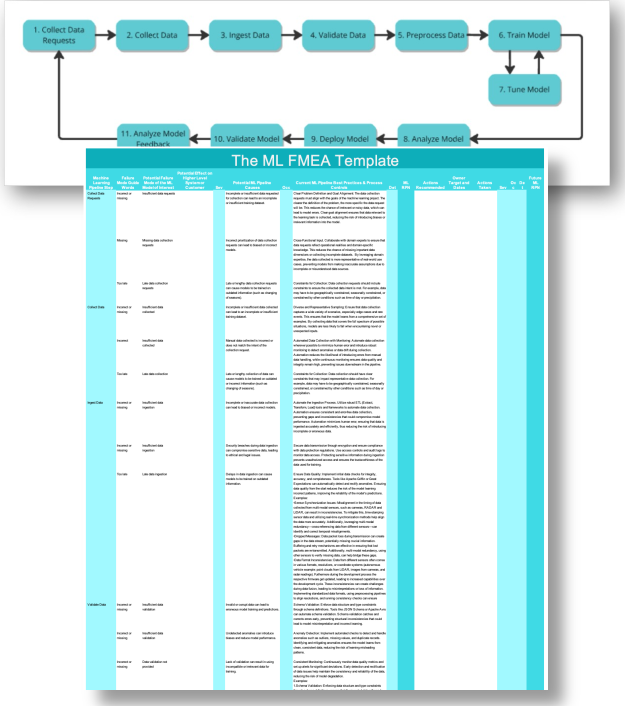
The Contribution



The ML FMEA Method

- Connects ML Pipeline step -> Relevant ML failure modes -> Possible ML causes -> Known ML mitigations

The Contribution



The ML FMEA Template

Machine Learning Pipeline Step	Failure Mode Guide Words	Potential Failure Mode of the ML Model of Interest	Potential Effect on Higher Level System or Customer	Sev	Potential ML Pipeline Causes	Occ	Current ML Pipeline Best Practices & Process Controls	Det	ML RPN	Actions Recommended	Owner Target and Dates	Actions Taken	Sev	Occ	Det	Future ML RPN
Collect Data Requests	Incorrect or missing	Insufficient data requests			Incomplete or insufficient data requested for collection can lead to an incomplete or insufficient training dataset.		Clear Problem Definition and Goal Alignment: The data collection requests must align with the goals of the machine learning project. The clearer the definition of the problem, the more specific the data request will be. This reduces the chance of irrelevant or noisy data, which can lead to model errors. Clear goal alignment ensures that data relevant to the learning task is collected, reducing the risk of introducing biases or irrelevant information into the model.									
	Missing	Missing data collection requests			Incorrect prioritization of data collection requests can lead to biased or incorrect models.		Cross-Functional Input. Collaborate with domain experts to ensure that data requests reflect operational realities and domain-specific knowledge. This reduces the chance of missing important data dimensions or collecting incomplete datasets. By leveraging domain expertise, the data collected is more representative of real-world use cases, preventing models from making inaccurate assumptions due to incomplete or misunderstood data sources.									
	Too late	Late data collection requests			Late or lengthy data collection requests can cause models to be trained on outdated information (such as changing of seasons).		Constraints for Collection: Data collection requests should include constraints to ensure the collected data intent is met. For example, data may have to be geographically constrained, seasonally constrained, or constrained by other conditions such as time of day or precipitation.									
Collect Data	Incorrect or missing	Insufficient data collected			Incomplete or insufficient data collected can lead to an incomplete or insufficient training dataset.		Diverse and Representative Sampling: Ensure that data collection captures a wide variety of scenarios, especially edge cases and rare events. This ensures that the model learns from a comprehensive set of examples. By collecting data that covers the full spectrum of possible situations, models are less likely to fail when encountering novel or unexpected inputs.									
	Incorrect	Insufficient data collected			Manual data collected is incorrect or does not match the intent of the collection request.		Automated Data Collection with Monitoring: Automate data collection wherever possible to minimize human error and introduce robust monitoring to detect anomalies or data drift during collection. Automation reduces the likelihood of introducing errors from manual data handling, while continuous monitoring ensures data quality and integrity remain high, preventing issues downstream in the pipeline.									
	Too late	Late data collection			Late or lengthy collection of data can cause models to be trained on outdated or incorrect information (such as changing of seasons).		Constraints for Collection: Data collection should have clear constraints that may impact representative data collection. For example, data may have to be geographically constrained, seasonally constrained, or constrained by other conditions such as time of day or precipitation.									
Ingest Data	Incorrect or missing	Insufficient data ingestion			Incomplete or inaccurate data collection can lead to biased or incorrect models.		Automate the Ingestion Process. Utilize robust ETL (Extract, Transform, Load) tools and frameworks to automate data collection. Automation ensures consistent and error-free data collection, preventing gaps and inconsistencies that could compromise model performance. Automation minimizes human error, ensuring that data is ingested accurately and efficiently, thus reducing the risk of introducing incomplete or erroneous data.									
	Incorrect or missing	Insufficient data ingestion			Security breaches during data ingestion can compromise sensitive data, leading to ethical and legal issues.		Secure data transmission through encryption and ensure compliance with data protection regulations. Use access controls and audit logs to monitor data access. Protecting sensitive information during ingestion prevents unauthorized access and ensures the trustworthiness of the data used for training.									
	Too late	Late data ingestion			Delays in data ingestion can cause models to be trained on outdated information.		Ensure Data Quality: Implement initial data checks for integrity, accuracy, and completeness. Tools like Apache Griffin or Great Expectations can automatically detect and rectify anomalies. Ensuring data quality from the start reduces the risk of the model learning incorrect patterns, improving the reliability of the model's predictions. Examples: •Sensor Synchronization Issues: Misalignment in the timing of data collected from multi-modal sensors, such as cameras, RADAR and LiDAR, can result in inconsistencies. To mitigate this, time-stamping sensor data and utilizing real-time synchronization methods help align the data more accurately. Additionally, leveraging multi-modal redundancy—cross-referencing data from different sensors—can identify and correct temporal misalignments. •Dropped Messages: Data packet loss during transmission can create gaps in the data stream, potentially missing crucial information. Buffering and retry mechanisms are effective in ensuring that lost packets are re-transmitted. Additionally, multi-modal redundancy, using other sensors to verify missing data, can help bridge these gaps. •Data Format Inconsistencies: Data from different sensors often comes in various formats, resolutions, or coordinate systems (autonomous vehicle example: point clouds from LiDAR, images from cameras, and radar readings). Furthermore during the development process the respective firmware get updated, leading to increased capabilities over the development cycle. These inconsistencies can create challenges during data fusion, leading to misinterpretations or loss of information. Implementing standardized data formats, using preprocessing pipelines to align resolutions, and running consistency checks can ensure									
Validate Data	Incorrect or missing	Insufficient data validation			Invalid or corrupt data can lead to erroneous model training and predictions.		Schema Validation: Enforce data structure and type constraints through schema definitions. Tools like JSON Schema or Apache Avro can automate schema validation. Schema validation catches and corrects errors early, preventing structural inconsistencies that could lead to model misinterpretation and incorrect learning.									
	Incorrect or missing	Insufficient data validation			Undetected anomalies can introduce biases and reduce model performance.		Anomaly Detection: Implement automated checks to detect and handle anomalies such as outliers, missing values, and duplicate records. Identifying and mitigating anomalies ensures the model learns from clean, consistent data, reducing the risk of learning misleading patterns.									
	Incorrect or missing	Data validation not provided			Lack of validation can result in using incompatible or irrelevant data for training.		Consistent Monitoring: Continuously monitor data quality metrics and set up alerts for significant deviations. Early detection and rectification of data issues help maintain the consistency and reliability of the data, reducing the risk of model degradation. Examples: 1.Schema Validation: Enforcing data structure and type constraints									

The ML FMEA Method

- Connects ML Pipeline step -> Relevant ML failure modes -> Possible ML causes -> Known ML mitigations

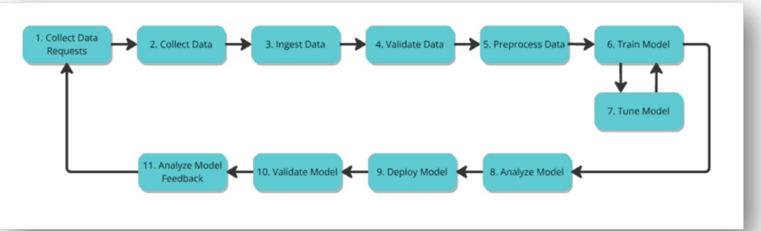
The ML FMEA Template

- PFMEA modified for Machine Learning
- Prepopulated
- Enables ML developers to assess, prioritize, and communicate safety risk

LAYING THE FOUNDATION

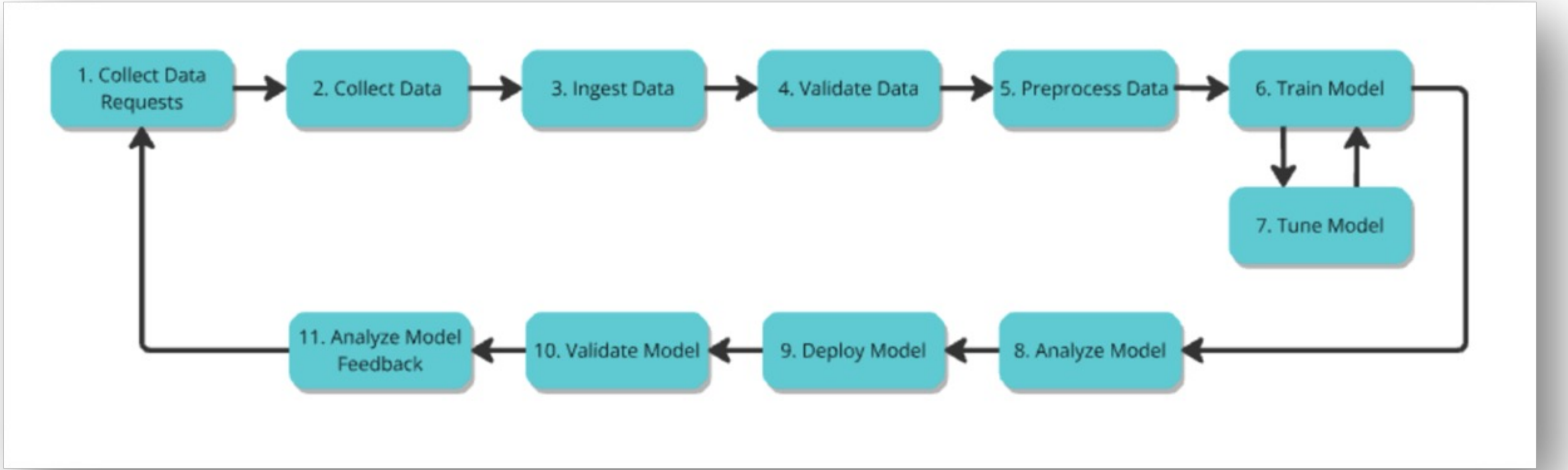
Link to the paper

The ML Pipeline

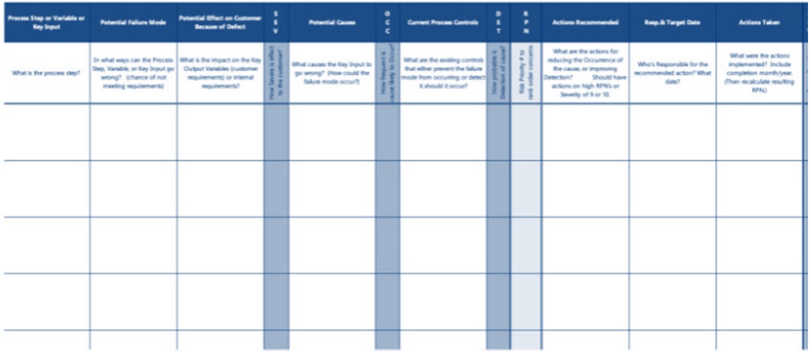


- There are several popular ML pipelines in the literature
- We crafted this one for the paper
- Clear steps
- Clear value addition

The ML Pipeline



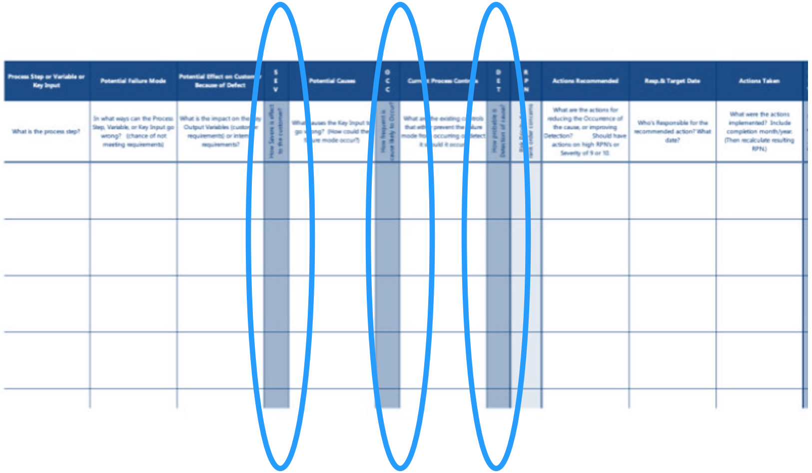
The Process Failure Modes and Effects Analysis



Process Step or Variable or Key Input	Potential Failure Mode	Potential Effect on Customer Because of Defect	SEV	Potential Causes	OCC	Current Process Controls	DET	RPN	Actions Recommended	Resp.& Target Date	Actions Taken
What is the process step?	In what ways can the Process Step, Variable, or Key Input go wrong? (chance of not meeting requirements)	What is the impact on the Key Output Variables (customer requirements) or internal requirements?	How Severe is effect to the customer?	What causes the Key Input to go wrong? (How could the failure mode occur?)	How frequent is cause likely to Occur?	What are the existing controls that either prevent the failure mode from occurring or detect it should it occur?	How probable is Detection of cause?	Risk Priority # to rank order concerns	What are the actions for reducing the Occurrence of the cause, or improving Detection? Should have actions on high RPN's or Severity of 9 or 10.	Who's Responsible for the recommended action? What date?	What were the actions implemented? Include completion month/year. (Then recalculate resulting RPN.)

- Established, systematic method to identify and mitigate risk
- Originally designed for automotive manufacturing processes

The Process Failure Modes and Effects Analysis



Process Step or Variable or Key Input	Potential Failure Mode	Potential Effect on Customer Because of Defect	SEV	Potential Causes	OCC	Current Process Controls	DET	RPN	Actions Recommended	Resp.& Target Date	Actions Taken
What is the process step?	In what ways can the Process Step, Variable, or Key Input go wrong? (chance of not meeting requirements)	What is the impact on the Key Output Variables (customer requirements) or internal requirements?	How Severe is effect to the customer?	What causes the Key Input to go wrong? (How could the failure mode occur?)	How frequent is cause likely to Occur?	What are the existing controls that either prevent the failure mode from occurring or detect it should it occur?	How probable is Detection of cause?	Risk Priority [illegible] rank order concerns	What are the actions for reducing the Occurrence of the cause, or improving Detection? Should have actions on high RPN's or Severity of 9 or 10.	Who's Responsible for the recommended action? What date?	What were the actions implemented? Include completion month/year. (Then recalculate resulting RPN.)

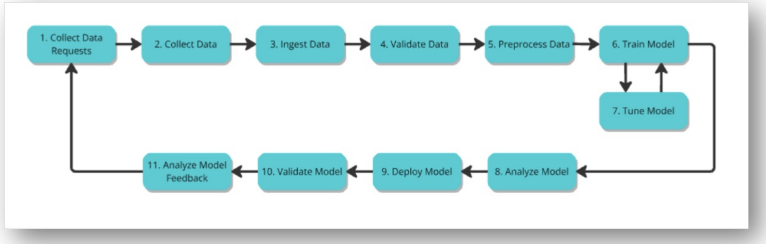
- Established, systematic method to identify and mitigate risk
- Originally designed for automotive manufacturing processes
- Assesses failures for each process step:
  - Severity
  - Occurrence
  - Detectability
- Identify, prioritize, and manage risk

THE ML FMEA:

THE METHOD

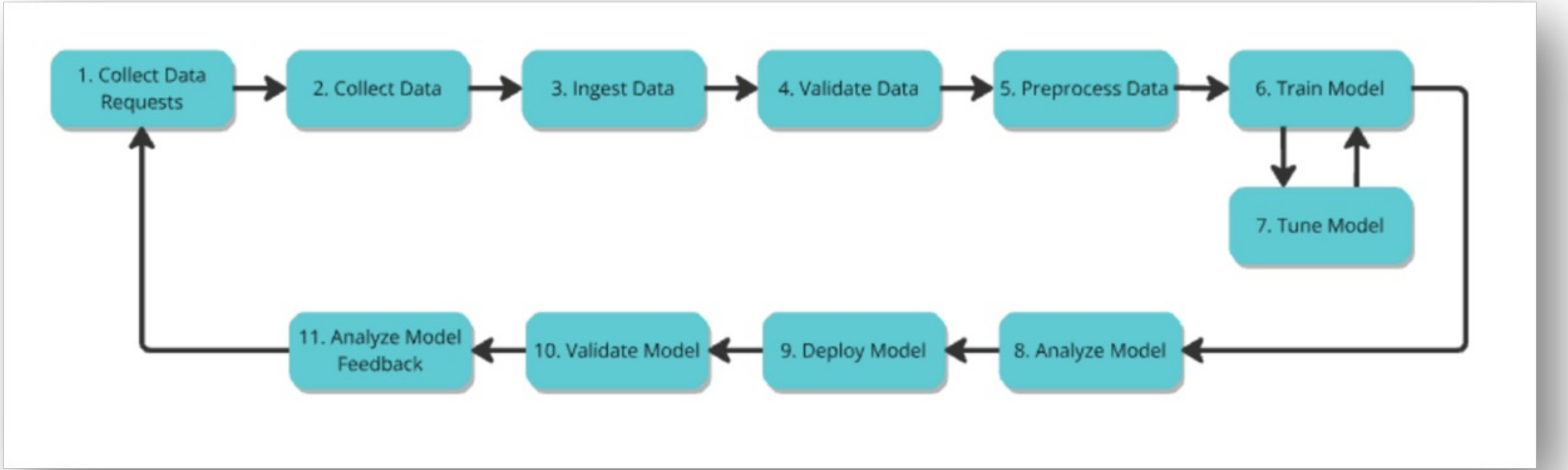
Link to the paper

The ML FMEA Method

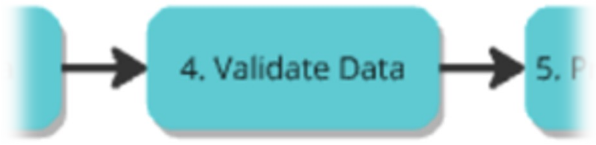


- View the ML Pipeline as a process
- The model is an output of the process
- Method:
  1. ML Value Function
  2. ML Model Failure mode
  3. ML Model Cause
  4. ML Best Practice & Mitigation

The ML FMEA Method



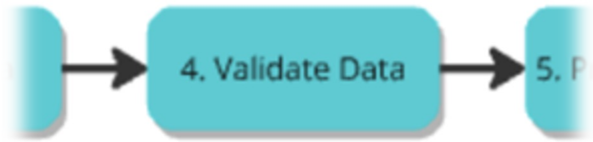
The ML FMEA Method



Example 1: Validate Data

- Verifies that the collected data meets quality standards and is suitable for analysis
- Ensures the data is accurate, complete, and free from significant errors

The ML FMEA Method



Example 1: Validate Data

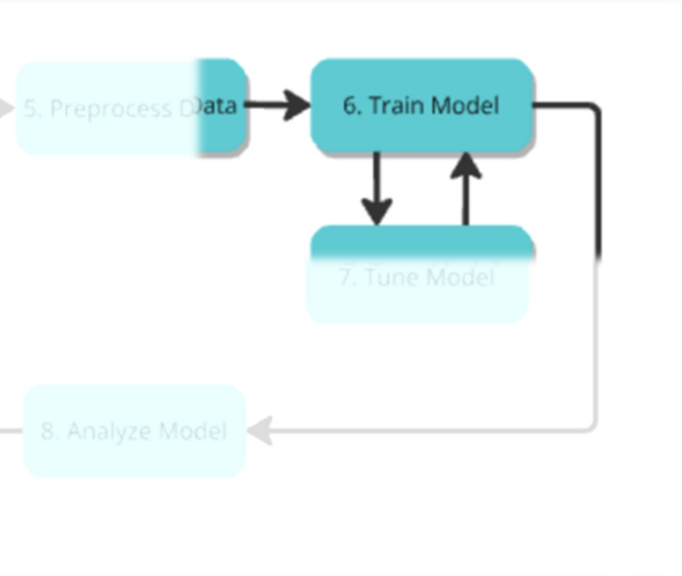
- Potential Failure 1:

  Invalid or corrupt data can lead to erroneous model training and predictions.

- Possible Mitigation:

  Schema Validation. Enforce data structure and type constraints through schema definitions. Tools like JSON Schema or Apache Avro can automate schema validation.

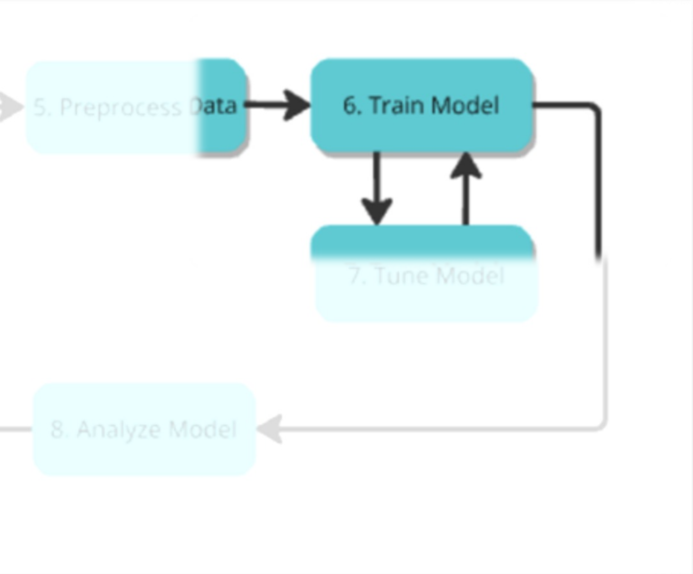
The ML FMEA Method



Example 2: Train Model

- Select appropriate algorithms, configuring model parameters, and fitting the model to the training data
- Train the model

The ML FMEA Method



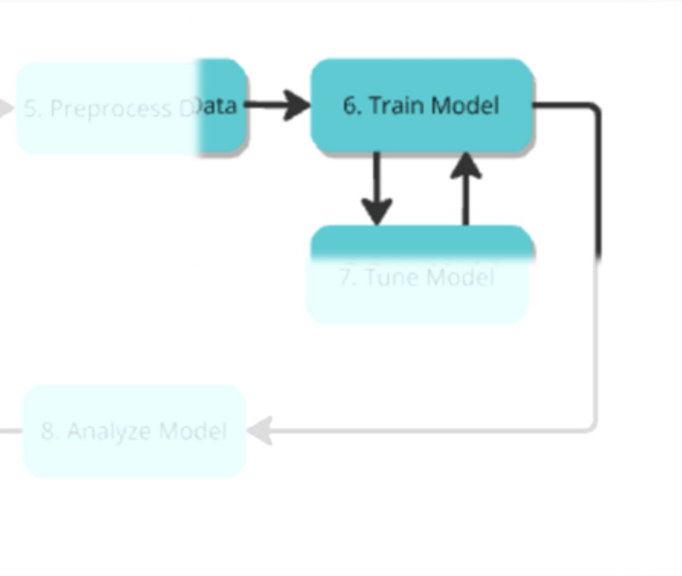
Example 2: Train Model

- Potential Failure 1:

  Incorrect algorithm selection can lead to poor model performance.

- Possible Mitigations:
  1. Ensure understanding of the problem that is being solved.
  2. Clarify and document available data characteristics
  3. Employ multiple models using a consistent training pipeline and compare using cross-validation.

The ML FMEA Method



Example 2: Train Model

- Potential Failure 2:

  Overfitting or underfitting the data

- Possible Mitigation:

  Cross-Validation. Employ techniques like k-fold cross-validation to ensure the model's performance is consistent across different data subsets.

The ML FMEA Method

Example 3: Deploy Model

- Integrates the trained model into a production environment
- Checks reliability and efficiency

The ML FMEA Method

Example 3: Deploy Model

- Potential Failure 1:

  Inadequate infrastructure can lead to performance bottlenecks.

- Possible Mitigation:

  Continuous Integration / Continuous Deployment (CI/CD). Implement CI/CD pipelines to automate the deployment process. CI/CD ensures smooth updates and minimizes manual errors, preventing deployment failures and maintaining model consistency.

The ML FMEA Method

Example 3: Deploy Model

- Potential Failure 2:

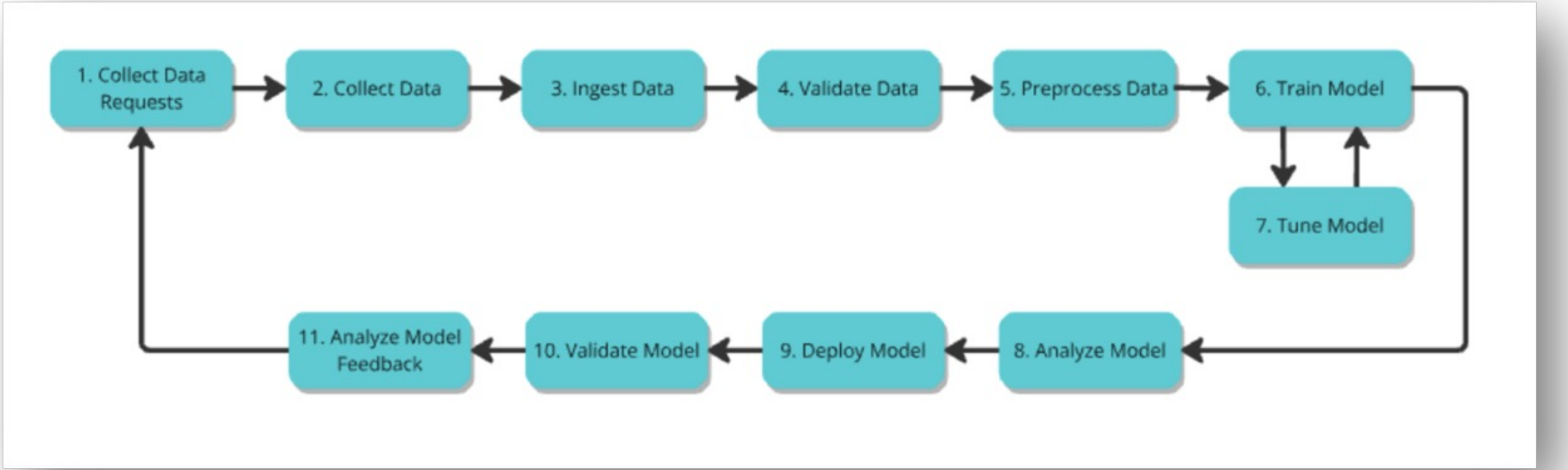
  Poor monitoring can result in undetected performance degradation.

- Possible Mitigations:

  Logging. Continuously monitor the deployed model's performance and log predictions.

  Alerts. Set up alerts and analyze logs to detect issues early.

The ML FMEA Method



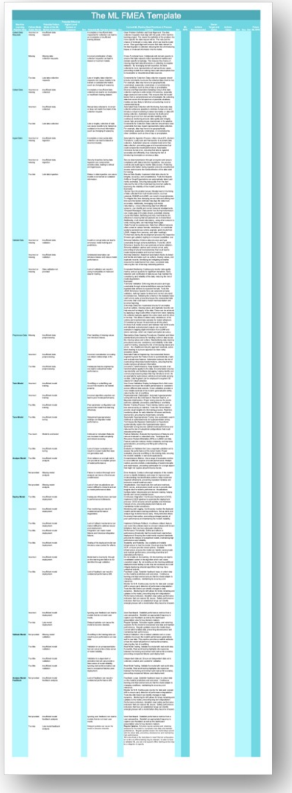
1. Collect Data Requests
2. Collect Data
3. Ingest Data
4. Validate Data
5. Preprocess Data
6. Train Model
7. Tune Model
8. Analyze Model
9. Deploy Model
10. Validate Model
11. Analyze Model Feedback

THE ML FMEA:

THE TEMPLATE

Link to the paper

The ML FMEA: The Template



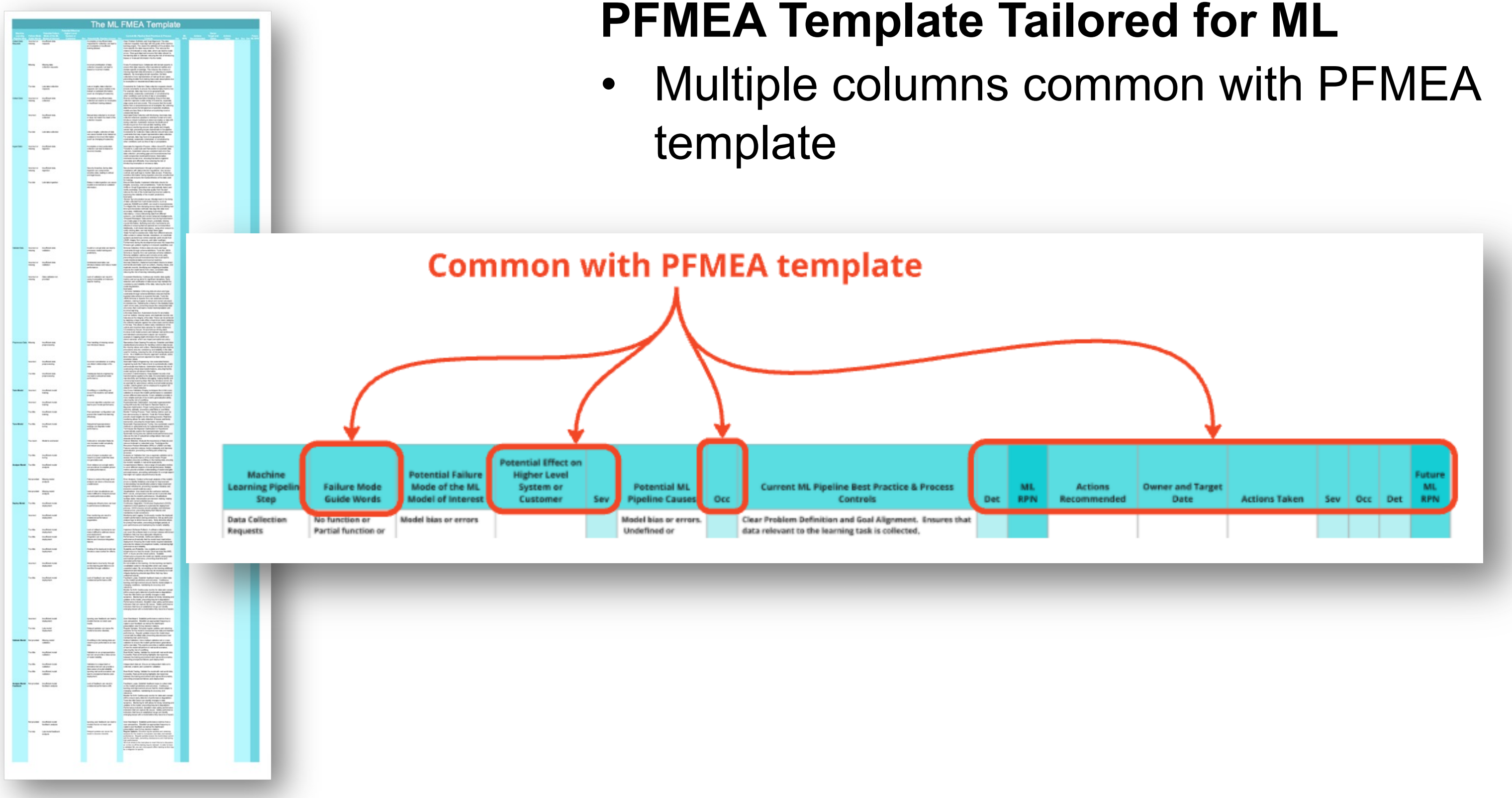
The ML FMEA Template

PFMEA Template Tailored for ML

The ML FMEA: The Template

PFMEA Template Tailored for ML

- Multiple columns common with PFMEA template

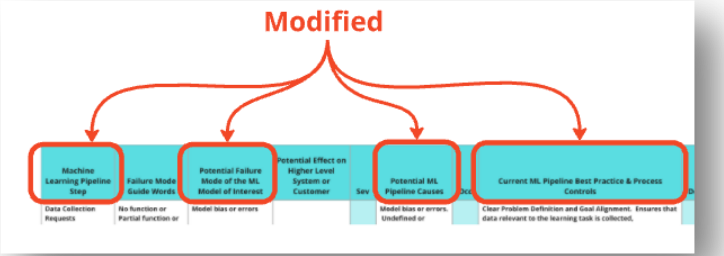


The ML FMEA Template

Common with PFMEA template

Machine Learning Pipelin Step	Failure Mode Guide Words	Potential Failure Mode of the ML Model of Interest	Potential Effect on Higher Level System or Customer	Sev	Potential ML Pipeline Causes	Occ	Current ML Pipeline Best Practice & Process Controls	Det	ML RPN	Actions Recommended	Owner and Target Date	Actions Taken	Sev	Occ	Det	Future ML RPN
Data Collection Requests	No function or Partial function or	Model bias or errors			Model bias or errors. Undefined or		Clear Problem Definition and Goal Alignment. Ensures that data relevant to the learning task is collected.									

The ML FMEA: The Template

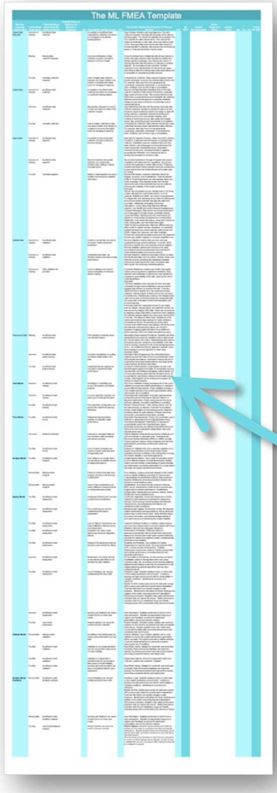


Machine Learning Pipeline Step	Failure Mode Guide Words	Potential Failure Mode of the ML Model of Interest	Potential Effect on Higher Level System or Customer	Sev	Potential ML Pipeline Causes	Occ	Current ML Pipeline Best Practice & Process Controls	D
Data Collection Requests	No function or Partial function or	Model bias or errors			Model bias or errors. Undefined or		Clear Problem Definition and Goal Alignment. Ensures that data relevant to the learning task is collected.	

PFMEA Template Tailored for ML

- Four columns tailored
  1. Pipeline step
  2. Potential Failure Mode of the ML Model of Interest
  3. Potential ML Pipeline Causes
  4. Current ML Pipeline Best Practice & Process Control

The ML FMEA: The Template

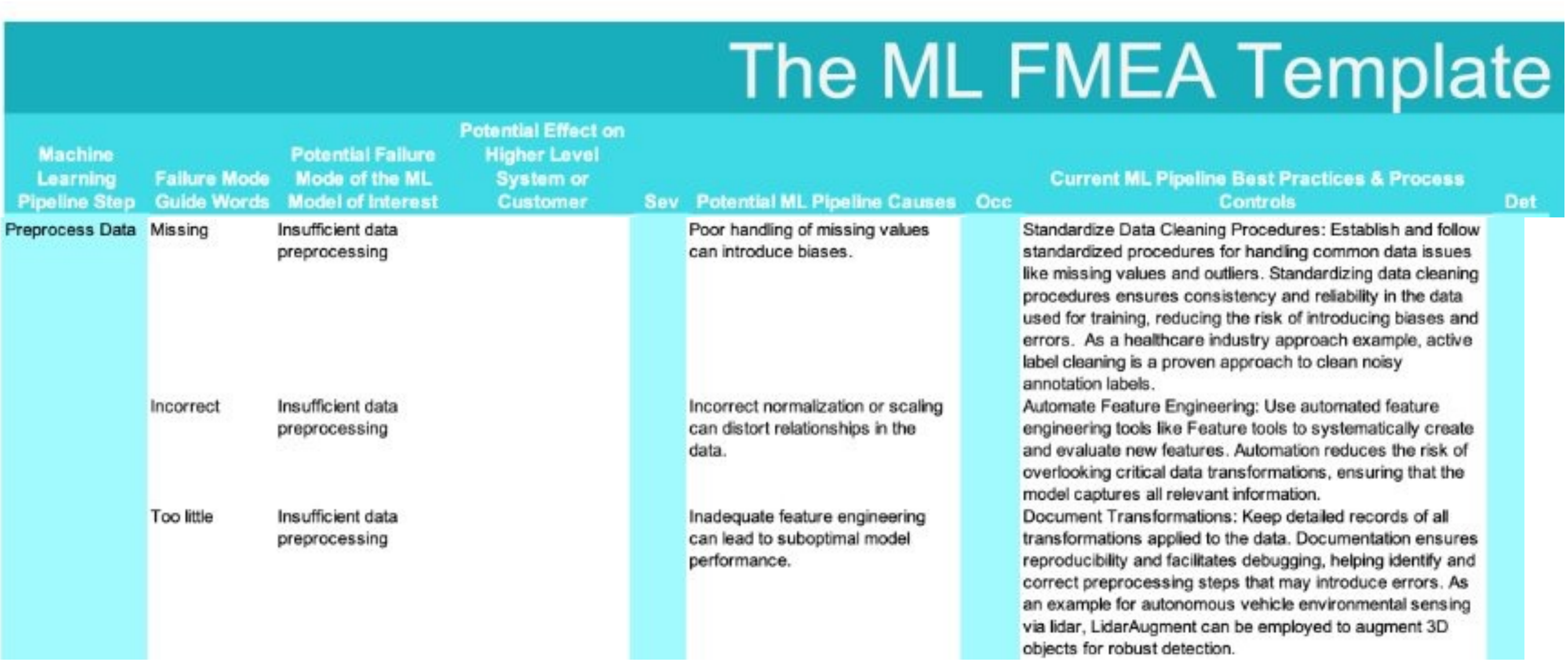


The ML FMEA Template

PFMEA Template Tailored for ML

- Four columns tailored
    1. Pipeline step
    2. Potential Failure Mode of the ML Model of Interest
    3. Potential ML Pipeline Causes
    4. Current ML Pipeline Best Practice and Process Control
- …and populated

The ML FMEA: The Template



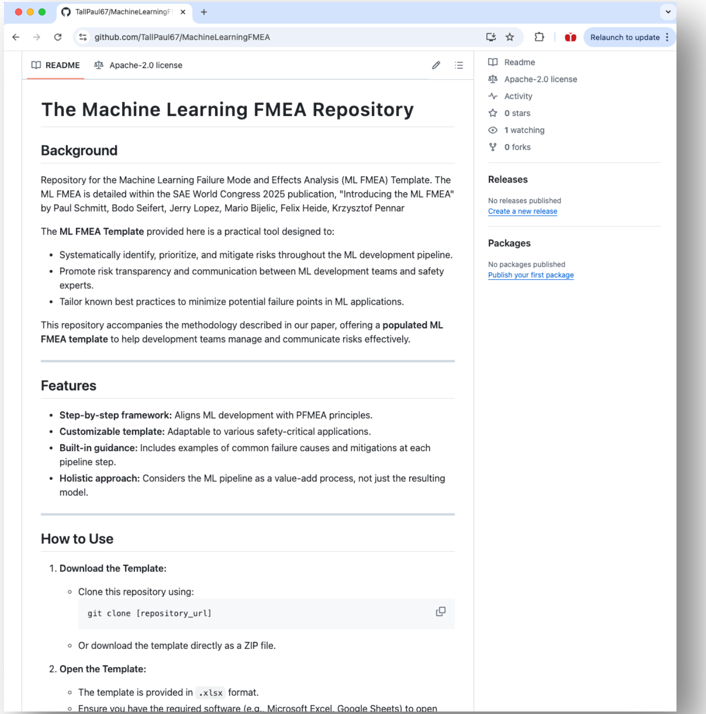
The ML FMEA Template

Machine Learning Pipeline Step	Failure Mode Guide Words	Potential Failure Mode of the ML Model of Interest	Potential Effect on Higher Level System or Customer	Sev	Potential ML Pipeline Causes	Occ	Current ML Pipeline Best Practices & Process Controls	Det
Preprocess Data	Missing	Insufficient data preprocessing			Poor handling of missing values can introduce biases.		Standardize Data Cleaning Procedures: Establish and follow standardized procedures for handling common data issues like missing values and outliers. Standardizing data cleaning procedures ensures consistency and reliability in the data used for training, reducing the risk of introducing biases and errors. As a healthcare industry approach example, active label cleaning is a proven approach to clean noisy annotation labels.	
	Incorrect	Insufficient data preprocessing			Incorrect normalization or scaling can distort relationships in the data.		Automate Feature Engineering: Use automated feature engineering tools like Feature tools to systematically create and evaluate new features. Automation reduces the risk of overlooking critical data transformations, ensuring that the model captures all relevant information.	
	Too little	Insufficient data preprocessing			Inadequate feature engineering can lead to suboptimal model performance.		Document Transformations: Keep detailed records of all transformations applied to the data. Documentation ensures reproducibility and facilitates debugging, helping identify and correct preprocessing steps that may introduce errors. As an example for autonomous vehicle environmental sensing via lidar, LidarAugment can be employed to augment 3D objects for robust detection.	

OPEN SOURCE

Link to the paper

The ML FMEA Template: Open Source



The Machine Learning FMEA Repository

Background

Repository for the Machine Learning Failure Mode and Effects Analysis (ML FMEA) Template. The ML FMEA is detailed within the SAE World Congress 2025 publication, "Introducing the ML FMEA" by Paul Schmitt, Bodo Seifert, Jerry Lopez, Mario Bijelic, Felix Heide, Krzysztof Pennar

The **ML FMEA Template** provided here is a practical tool designed to:

- Systematically identify, prioritize, and mitigate risks throughout the ML development pipeline.
- Promote risk transparency and communication between ML development teams and safety experts.
- Tailor known best practices to minimize potential failure points in ML applications.

This repository accompanies the methodology described in our paper, offering a **populated ML FMEA template** to help development teams manage and communicate risks effectively.

Features

- **Step-by-step framework:** Aligns ML development with PFMEA principles.
- **Customizable template:** Adaptable to various safety-critical applications.
- **Built-in guidance:** Includes examples of common failure causes and mitigations at each pipeline step.
- **Holistic approach:** Considers the ML pipeline as a value-add process, not just the resulting model.

How to Use

1. **Download the Template:**
   - Clone this repository using:
     ```
     git clone [repository_url]
     ```
   - Or download the template directly as a ZIP file.
2. **Open the Template:**
   - The template is provided in `.xlsx` format.
   - Ensure you have the required software (e.g., Microsoft Excel, Google Sheets) to open

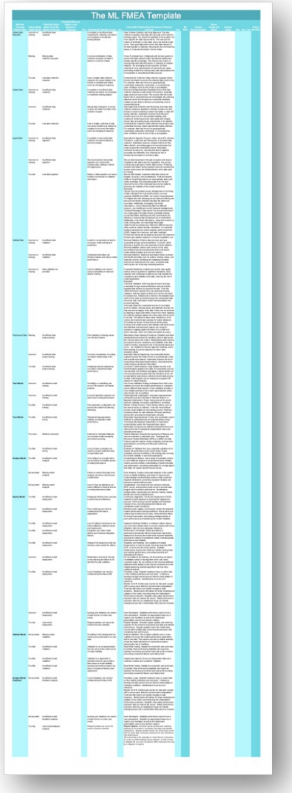
- Available on github.com
- Inviting the community to use and improve the template. Across industries!
- Start here:

DISCUSSION

BENEFITS & LIMITATIONS

Link to the paper

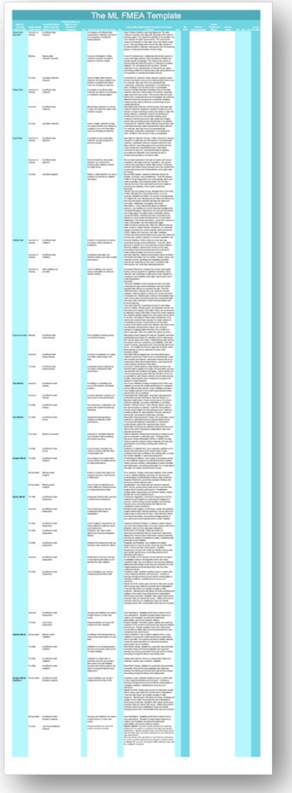
Discussion: The ML FMEA



Limitations

- Not a stand-alone tool
- Part of a comprehensive safety management system and safety case

Discussion: The ML FMEA

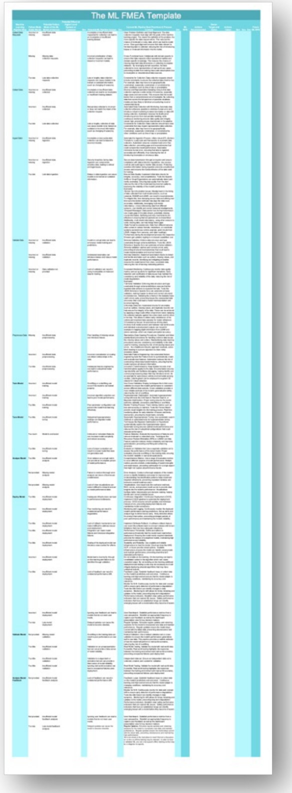


The ML FMEA Template

Template Benefits

1. Provides a systematic way of identifying and documenting functional insufficiencies and mitigations associated with the ML pipeline
2. “Checklist” to ensure that no critical steps were missed in the ML pipeline development

Discussion: The ML FMEA

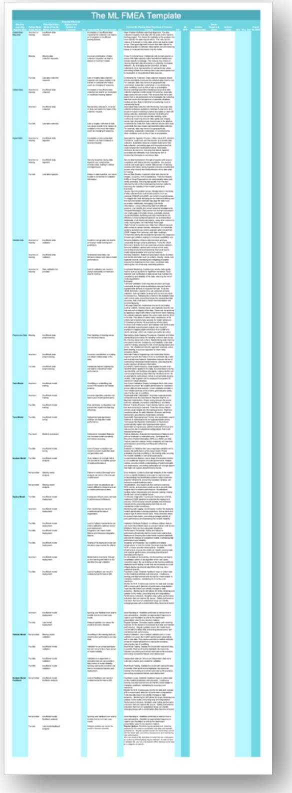


The ML FMEA Template

Template Benefits

3. Identify steps in the ML pipeline that could have the highest impact on safety risk

Discussion: The ML FMEA

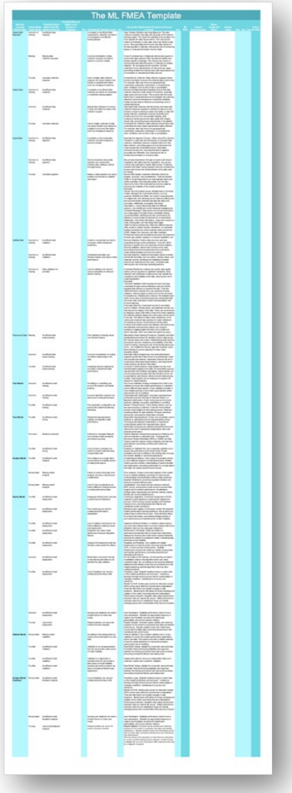


The ML FMEA Template

Template Benefits

3. Identify steps in the ML pipeline that could have the highest impact on safety risk
4. Serve as important evidence artifact supporting a safety case claim that the ML pipeline was developed with the highest level of rigor

Discussion: The ML FMEA



The ML FMEA Template

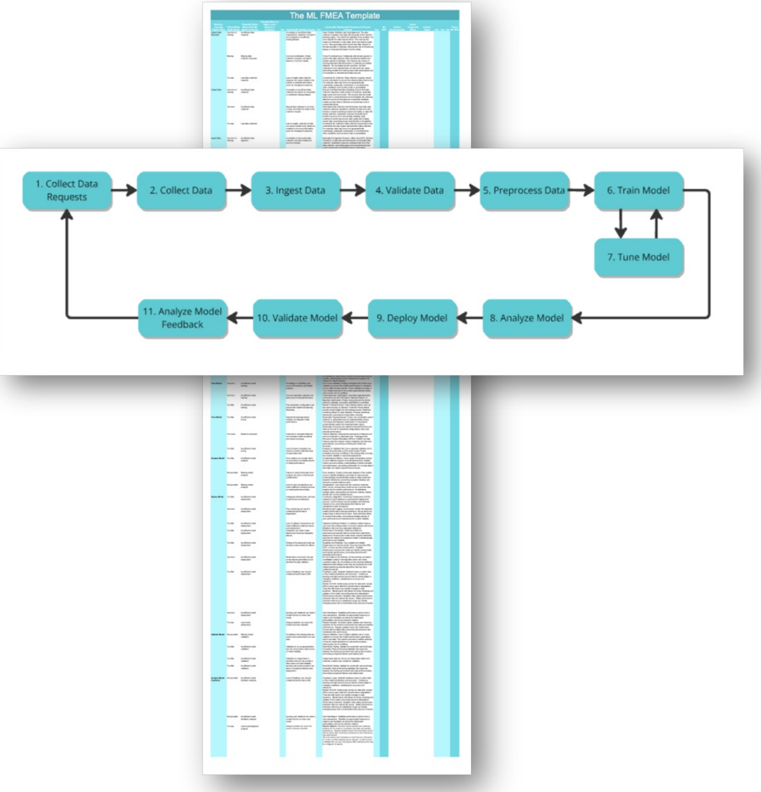
Template Benefits

3. Identify steps in the ML pipeline that could have the highest impact on safety risk
4. Serve as important evidence artifact supporting a safety case claim that the ML pipeline was developed with the highest level of rigor
5. Enables a common language between ML development teams and safety assessors

FUTURE WORK

Link to the paper

Future Work: The ML FMEA

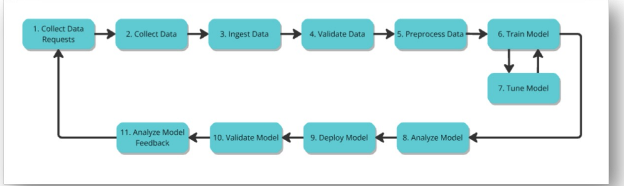


- Conceptual -> Applied
- Open Source Community improvements!

SUMMARY

Link to the paper

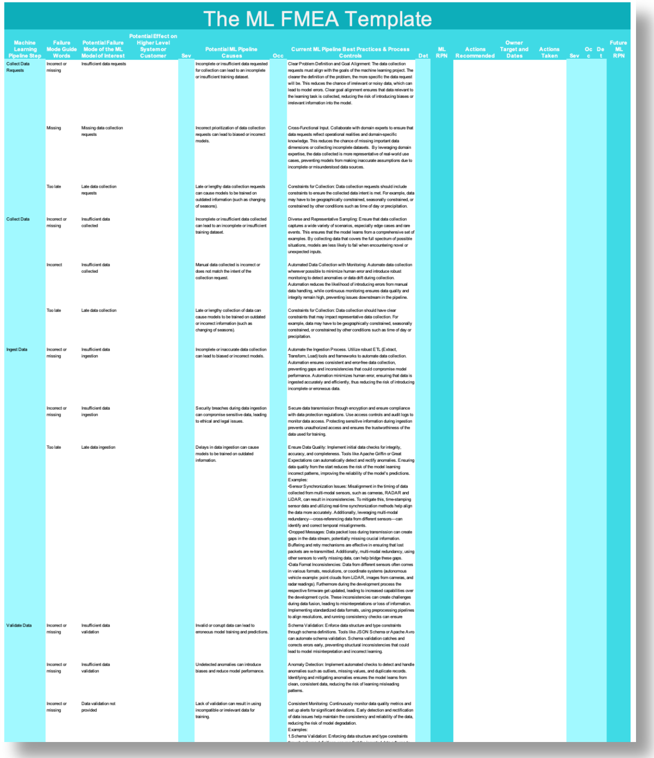
Summary



The ML FMEA Method

1. Analyzes safety risk at each stage of the ML pipeline.
2. Based upon proven failure mode analysis and ML methods

Summary



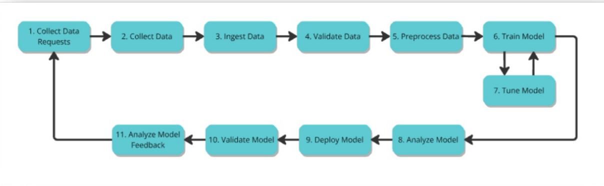
The ML FMEA Template

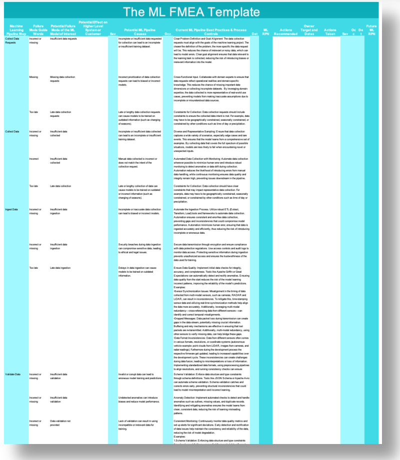
Machine Learning Pipeline Step	Failure Mode Guide Words	Potential Failure Mode of the ML Model of Interest	Potential Effect on Higher Level System or Customer	Sev	Potential ML Pipeline Causes	Occ	Current ML Pipeline Best Practices & Process Controls	Det	ML RPN	Actions Recommended	Owner Target and Dates	Actions Taken	Sev	Oc c	De t	Future ML RPN
Collect Data Requests	Incorrect or missing	Insufficient data requests			Incomplete or insufficient data requested for collection can lead to an incomplete or insufficient training dataset.		Clear Problem Definition and Goal Alignment: The data collection requests must align with the goals of the machine learning project. The clearer the definition of the problem, the more specific the data request will be. This reduces the chance of irrelevant or noisy data, which can lead to model errors. Clear goal alignment ensures that data relevant to the learning task is collected, reducing the risk of introducing biases or irrelevant information into the model.									
	Missing	Missing data collection requests			Incorrect prioritization of data collection requests can lead to biased or incorrect models.		Cross-Functional Input. Collaborate with domain experts to ensure that data requests reflect operational realities and domain-specific knowledge. This reduces the chance of missing important data dimensions or collecting incomplete datasets. By leveraging domain expertise, the data collected is more representative of real-world use cases, preventing models from making inaccurate assumptions due to incomplete or misunderstood data sources.									
	Too late	Late data collection requests			Late or lengthy data collection requests can cause models to be trained on outdated information (such as changing of seasons).		Constraints for Collection: Data collection requests should include constraints to ensure the collected data intent is met. For example, data may have to be geographically constrained, seasonally constrained, or constrained by other conditions such as time of day or precipitation.									
Collect Data	Incorrect or missing	Insufficient data collected			Incomplete or insufficient data collected can lead to an incomplete or insufficient training dataset.		Diverse and Representative Sampling: Ensure that data collection captures a wide variety of scenarios, especially edge cases and rare events. This ensures that the model learns from a comprehensive set of examples. By collecting data that covers the full spectrum of possible situations, models are less likely to fail when encountering novel or unexpected inputs.									
	Incorrect	Insufficient data collected			Manual data collected is incorrect or does not match the intent of the collection request.		Automated Data Collection with Monitoring: Automate data collection wherever possible to minimize human error and introduce robust monitoring to detect anomalies or data drift during collection. Automation reduces the likelihood of introducing errors from manual data handling, while continuous monitoring ensures data quality and integrity remain high, preventing issues downstream in the pipeline.									
	Too late	Late data collection			Late or lengthy collection of data can cause models to be trained on outdated or incorrect information (such as changing of seasons).		Constraints for Collection: Data collection should have clear constraints that may impact representative data collection. For example, data may have to be geographically constrained, seasonally constrained, or constrained by other conditions such as time of day or precipitation.									
Ingest Data	Incorrect or missing	Insufficient data ingestion			Incomplete or inaccurate data collection can lead to biased or incorrect models.		Automate the Ingestion Process. Utilize robust ETL (Extract, Transform, Load) tools and frameworks to automate data collection. Automation ensures consistent and error-free data collection, preventing gaps and inconsistencies that could compromise model performance. Automation minimizes human error, ensuring that data is ingested accurately and efficiently, thus reducing the risk of introducing incomplete or erroneous data.									
	Incorrect or missing	Insufficient data ingestion			Security breaches during data ingestion can compromise sensitive data, leading to ethical and legal issues.		Secure data transmission through encryption and ensure compliance with data protection regulations. Use access controls and audit logs to monitor data access. Protecting sensitive information during ingestion prevents unauthorized access and ensures the trustworthiness of the data used for training.									
	Too late	Late data ingestion			Delays in data ingestion can cause models to be trained on outdated information.		Ensure Data Quality: Implement initial data checks for integrity, accuracy, and completeness. Tools like Apache Griffin or Great Expectations can automatically detect and rectify anomalies. Ensuring data quality from the start reduces the risk of the model learning incorrect patterns, improving the reliability of the model's predictions. Examples: •Sensor Synchronization Issues: Misalignment in the timing of data collected from multi-modal sensors, such as cameras, RADAR and LiDAR, can result in inconsistencies. To mitigate this, time-stamping sensor data and utilizing real-time synchronization methods help align the data more accurately. Additionally, leveraging multi-modal redundancy—cross-referencing data from different sensors—can identify and correct temporal misalignments. •Dropped Messages: Data packet loss during transmission can create gaps in the data stream, potentially missing crucial information. Buffering and retry mechanisms are effective in ensuring that lost packets are re-transmitted. Additionally, multi-modal redundancy, using other sensors to verify missing data, can help bridge these gaps. •Data Format Inconsistencies: Data from different sensors often comes in various formats, resolutions, or coordinate systems (autonomous vehicle example: point clouds from LiDAR, images from cameras, and radar readings). Furthermore during the development process the respective firmware get updated, leading to increased capabilities over the development cycle. These inconsistencies can create challenges during data fusion, leading to misinterpretations or loss of information. Implementing standardized data formats, using preprocessing pipelines to align resolutions, and running consistency checks can ensure									
Validate Data	Incorrect or missing	Insufficient data validation			Invalid or corrupt data can lead to erroneous model training and predictions.		Schema Validation: Enforce data structure and type constraints through schema definitions. Tools like JSON Schema or Apache Avro can automate schema validation. Schema validation catches and corrects errors early, preventing structural inconsistencies that could lead to model misinterpretation and incorrect learning.									
	Incorrect or missing	Insufficient data validation			Undetected anomalies can introduce biases and reduce model performance.		Anomaly Detection: Implement automated checks to detect and handle anomalies such as outliers, missing values, and duplicate records. Identifying and mitigating anomalies ensures the model learns from clean, consistent data, reducing the risk of learning misleading patterns.									
	Incorrect or missing	Data validation not provided			Lack of validation can result in using incompatible or irrelevant data for training.		Consistent Monitoring: Continuously monitor data quality metrics and set up alerts for significant deviations. Early detection and rectification of data issues help maintain the consistency and reliability of the data, reducing the risk of model degradation. Examples: 1.Schema Validation: Enforcing data structure and type constraints									

The ML FMEA Template

1. Connects ML development pipeline failure modes with machine learning best practices as mitigations
2. Enables ML developers to assess, prioritize, and communicate safety risk
3. Empowers safety assessors to ask the right questions

Summary




The ML FMEA Template

- For ML in safety critical applications:
  - Automotive, ADAS, Autonomous Vehicles
  - Military, Aerospace
  - Medical, Pharmaceutical, …

CO-AUTHOR APPRECIATION

Co-authors: Introducing the ML FMEA

Paul Schmitt
Autonomy Senior Manager
Torc Robotics

Jerry Lopez
Sr. Director of
Safety Assurance
Torc Robotics

Krzysztof Pennar
Principal Engineer - Safety
Torc Robotics

Bodo Seifert
Sr. Safety Lead
TÜV Rheinland Group

Mario Bijelic
Machine Learning Lead
Torc Robotics

Felix Heide
AI Foundation
Torc Robotics

Introducing the ML FMEA

Paul Schmitt
TORC Robotics
paul.schmitt@torc.ai

Bodo Seifert
TÜV Rheinland
Bodo.Seifert@us.tuv.com

Link to the paper

Introducing the ML FMEA

Thank you!

Paul Schmitt
TORC Robotics
paul.schmitt@torc.ai

Link to the paper

Bodo Seifert
TÜV Rheinland
Bodo.Seifert@us.tuv.com

INTRODUCING THE MACHINE LEARNING FMEA

Paul Schmitt, Torc Robotics
Bodo Seifert, TÜV Rheinland
Jerry Lopez, Torc Robotics
Mario Bijelic, Torc Robotics
Krzysztof Pennar, Torc Robotics
Felix Heide, Torc Robotics

Link to the paper