Machine Learning for Software Dependability

Fabio Palomba
ZEST @ Ifi
University of Zurich

Machine Learning Basics

Definition

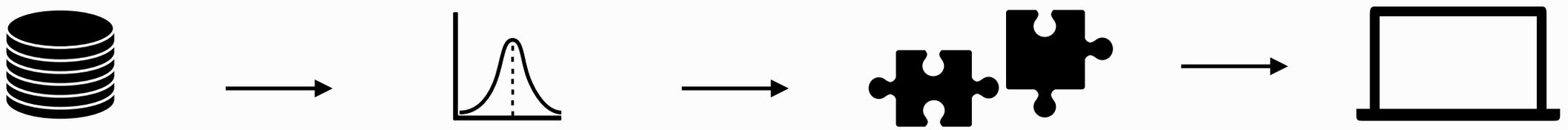
Machine Learning is a branch of Artificial Intelligence, that concerns with the design and development of algorithms that allow computers to evolve behaviors based on empirical data.

Machine Learning Basics

Definition

Machine Learning is a branch of Artificial Intelligence, that concerns with the design and development of algorithms that allow computers to evolve behaviors based on empirical data.

General Process



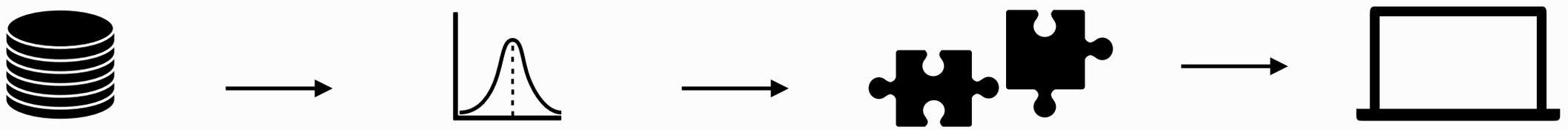
Input data → Analyze data → Find patterns → Predictions

Machine Learning Basics

Definition

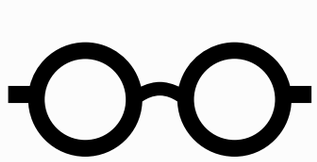
Machine Learning is a branch of Artificial Intelligence, that concerns with the design and development of algorithms that allow computers to evolve behaviors based on empirical data.

General Process



Input data → Analyze data → Find patterns → Predictions

Types of Machine Learning



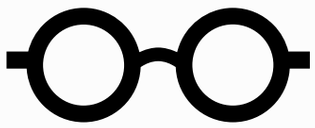
Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

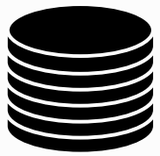
Reinforcement Learning
(autonomous learning)

Machine Learning Basics

Types of Machine Learning



Supervised Learning
(learning from labeled data)



Known data

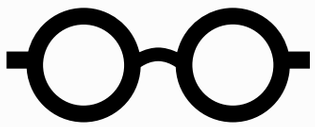
Known response

Unsupervised Learning
(learning from unlabeled data)

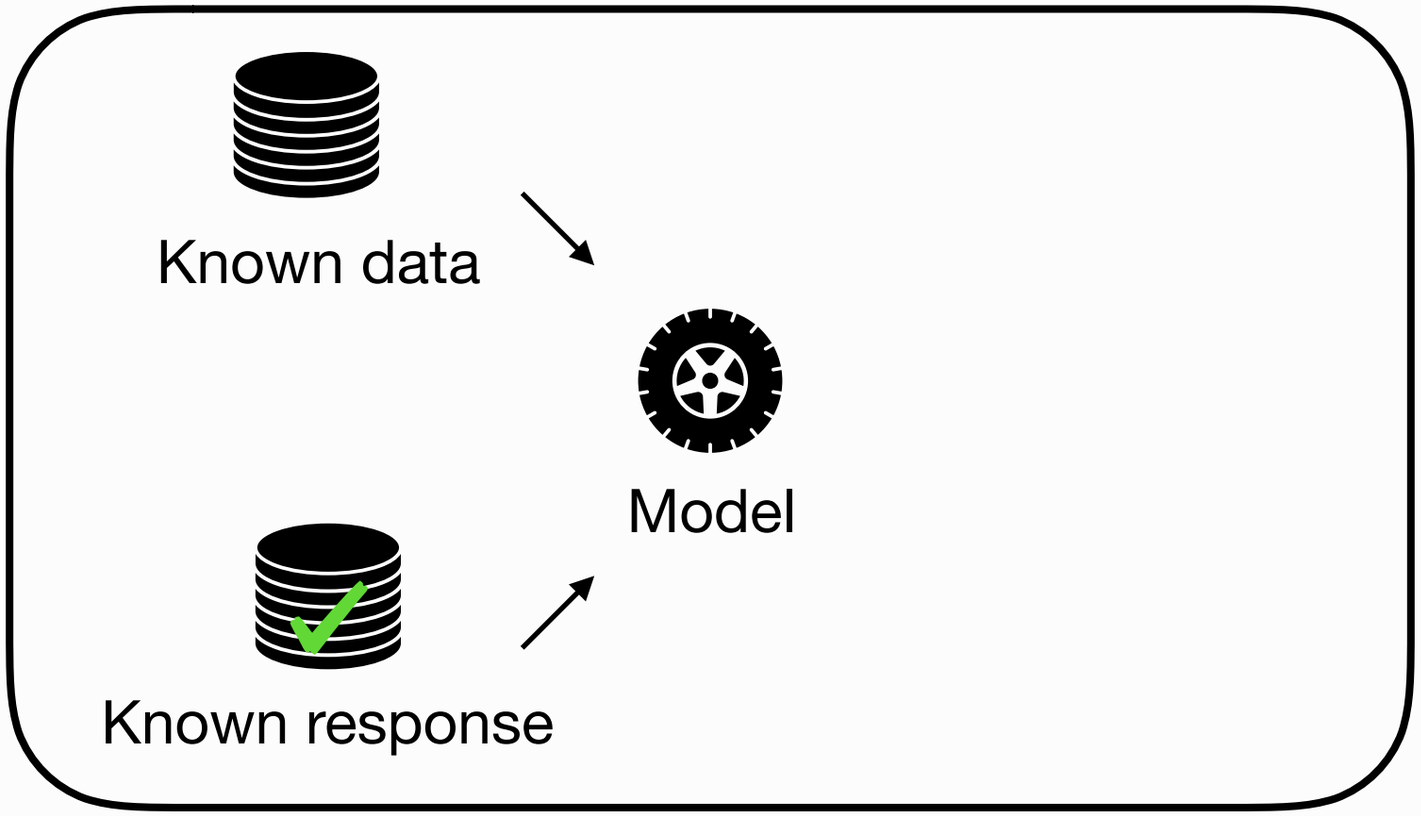
Reinforcement Learning
(autonomous learning)

Machine Learning Basics

Types of Machine Learning



Supervised Learning
(learning from labeled data)

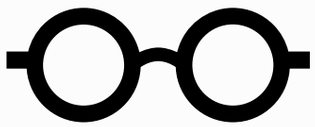


Unsupervised Learning
(learning from unlabeled data)

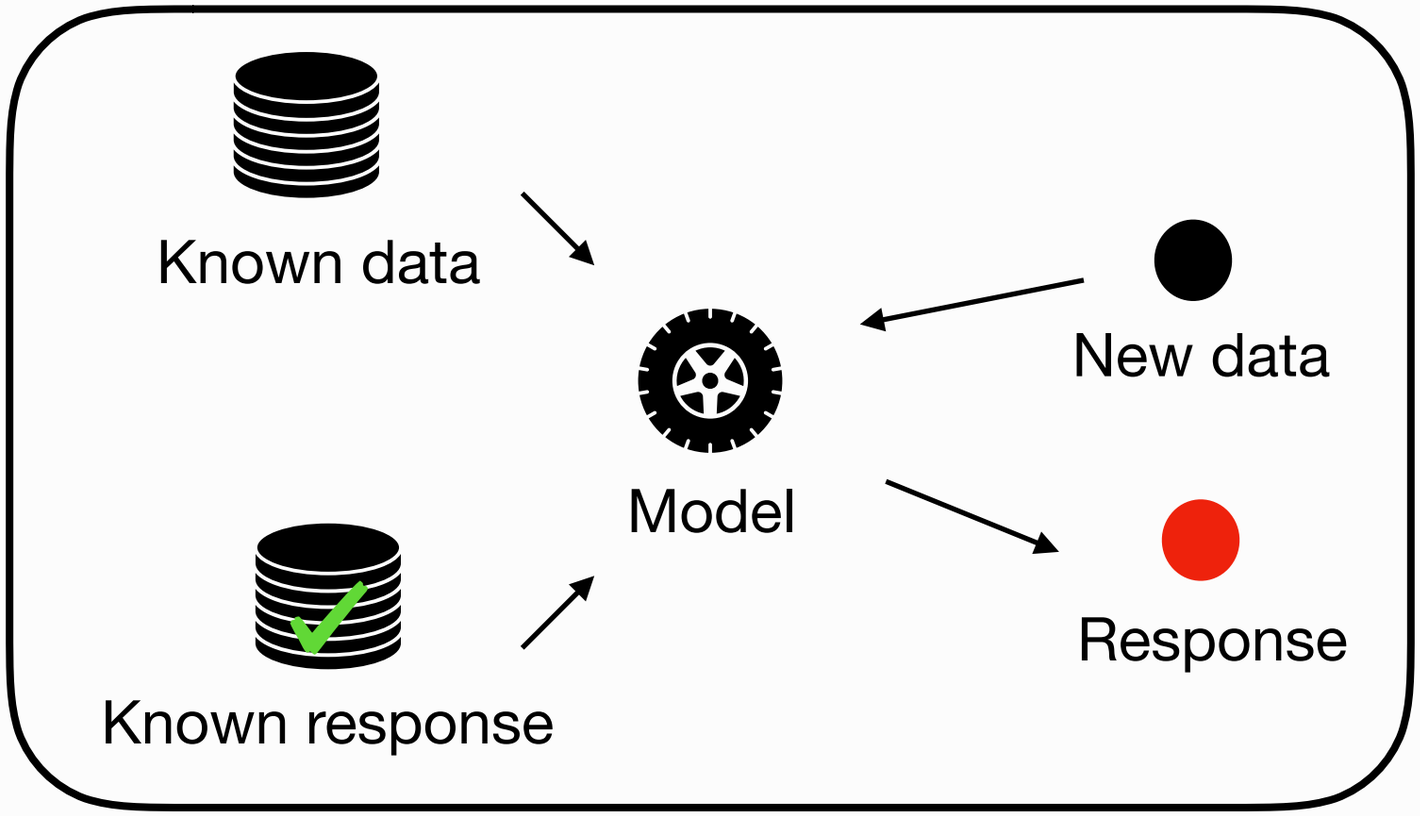
Reinforcement Learning
(autonomous learning)

Machine Learning Basics

Types of Machine Learning



Supervised Learning
(learning from labeled data)



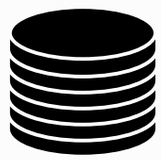
Unsupervised Learning
(learning from unlabeled data)

Reinforcement Learning
(autonomous learning)

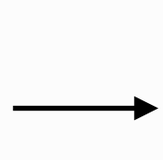
Machine Learning Basics

Types of Machine Learning

Unsupervised Learning
(learning from unlabeled data)



Input data



Model

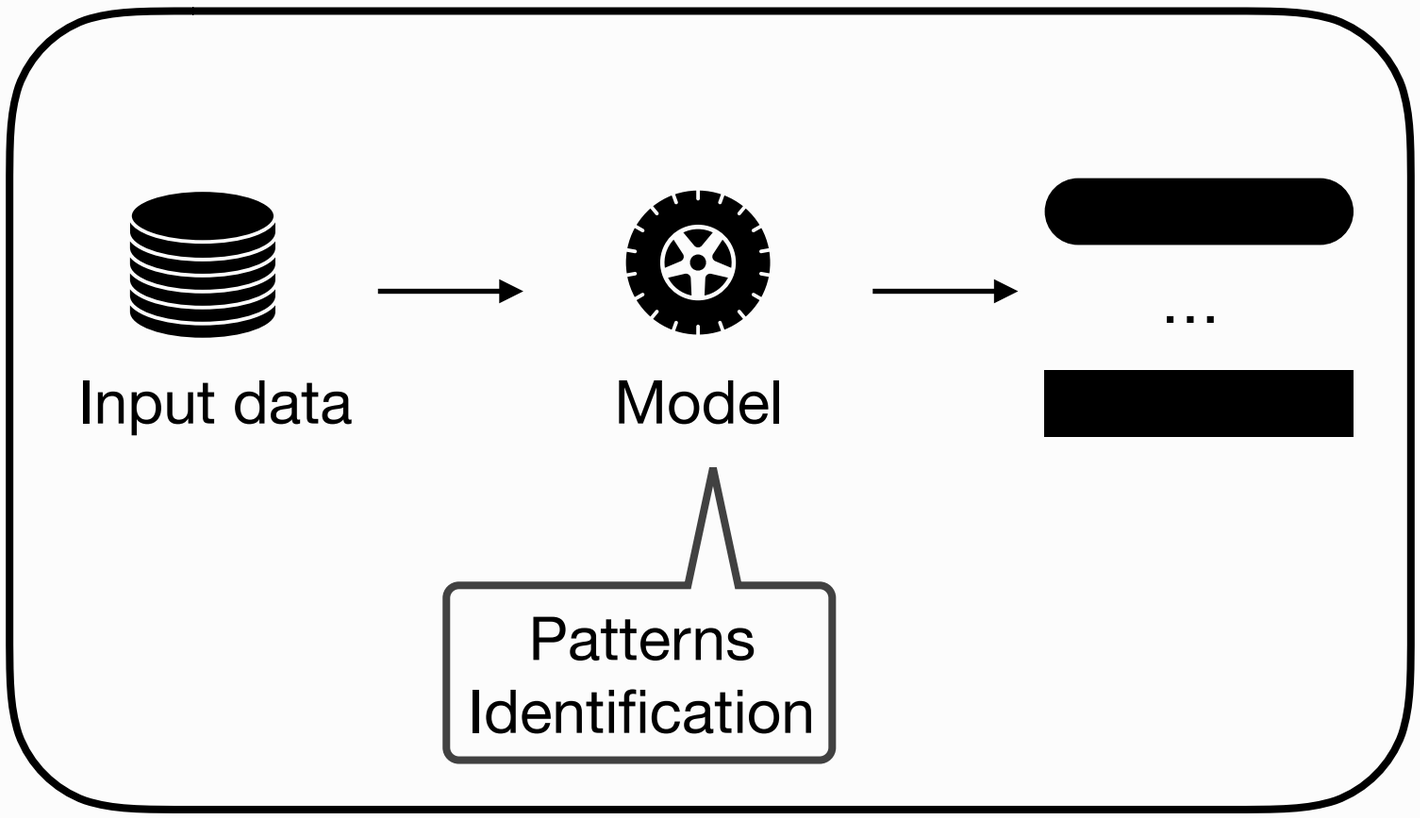
Supervised Learning
(learning from labeled data)

Reinforcement Learning
(autonomous learning)

Machine Learning Basics

Types of Machine Learning

Unsupervised Learning
(learning from unlabeled data)



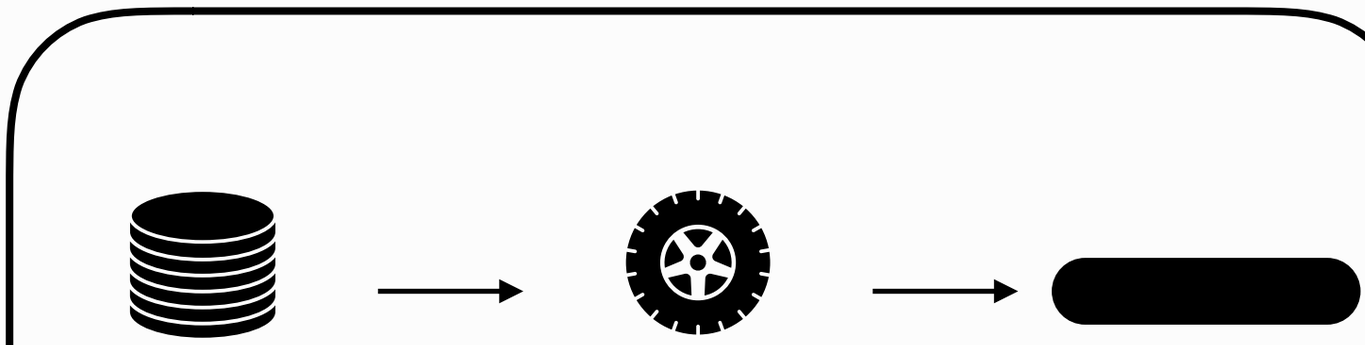
Supervised Learning
(learning from labeled data)

Reinforcement Learning
(autonomous learning)

Machine Learning Basics

Types of Machine Learning

Reinforcement Learning
(autonomous learning)



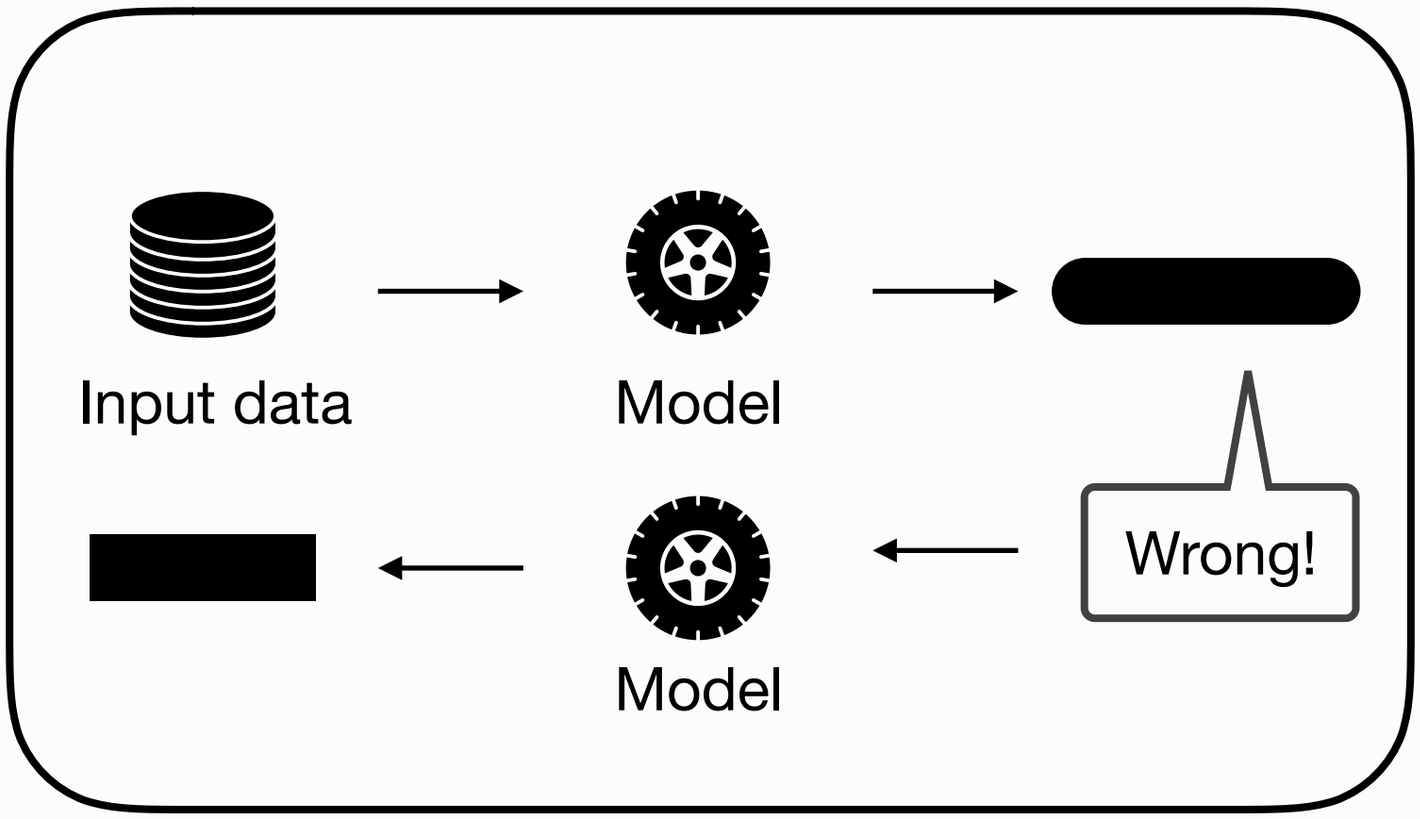
Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

Machine Learning Basics

Types of Machine Learning

Reinforcement Learning
(autonomous learning)



Supervised Learning
(learning from labeled data)

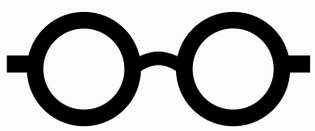
Unsupervised Learning
(learning from unlabeled data)

Machine Learning Basics

Types of Machine Learning - What's the right solution?

It depends on:

1. Problem specification;
2. Size, quantity, and nature of the data;
3. Complexity of the algorithm;
4. Other domain-specific requirements.



Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

Reinforcement Learning
(autonomous learning)

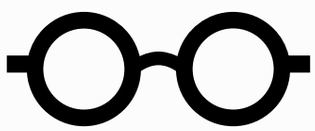
Machine Learning Basics

Types of Machine Learning - What's the right solution?

It depends on:

1. Problem specification;
2. Size, quantity, and nature of the data;
3. Complexity of the algorithm;
4. Other domain-specific requirements.

Classification: Used when the expected output is categorical (e.g., Yes/No);



Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

Reinforcement Learning
(autonomous learning)

Machine Learning Basics

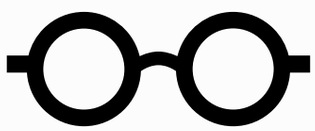
Types of Machine Learning - What's the right solution?

It depends on:

1. Problem specification;
2. Size, quantity, and nature of the data;
3. Complexity of the algorithm;
4. Other domain-specific requirements.

Classification: Used when the expected output is categorical (e.g., Yes/No);

Clustering: Used when the data needs to be organized to find patterns;



Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

Reinforcement Learning
(autonomous learning)

Machine Learning Basics

Types of Machine Learning - What's the right solution?

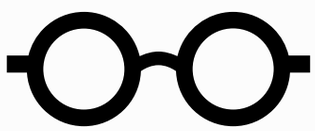
It depends on:

1. Problem specification;
2. Size, quantity, and nature of the data;
3. Complexity of the algorithm;
4. Other domain-specific requirements.

Classification: Used when the expected output is categorical (e.g., Yes/No);

Clustering: Used when the data needs to be organized to find patterns;

Regression: Used when the expected output is numerical (e.g., stock price).



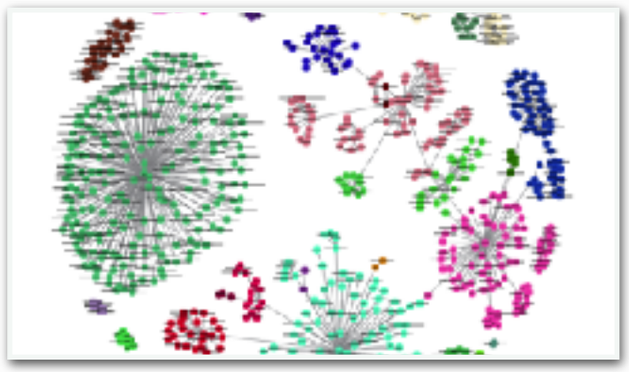
Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

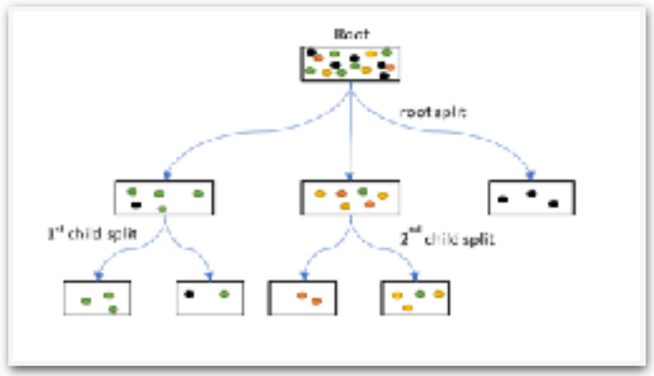
Reinforcement Learning
(autonomous learning)

Machine Learning Basics

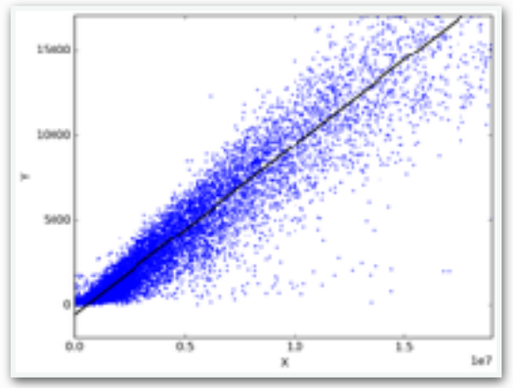
Machine Learning Algorithms



K-nearest neighbors



Decision Tree



Linear Regression

and many others…

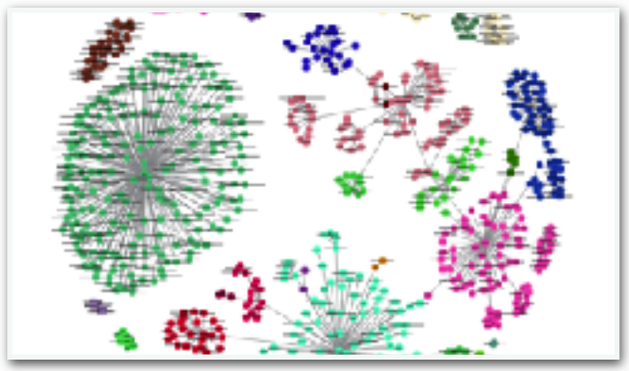
Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

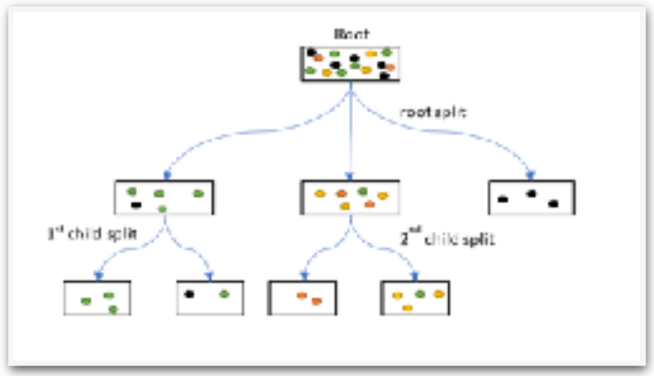
Reinforcement Learning
(autonomous learning)

Machine Learning Basics

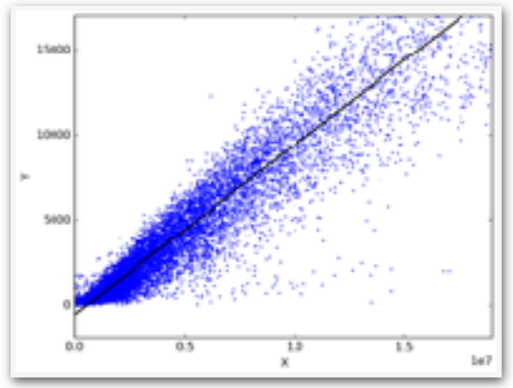
Machine Learning Algorithms



K-nearest neighbors



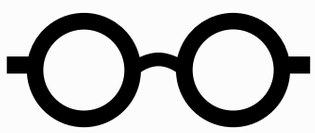
Decision Tree



Linear Regression

and many others…

The algorithm selection basically depends on the distribution of data. For instance, linear regression requires the underlying data distribution to be normal, while Decision Trees do not have this requirement.



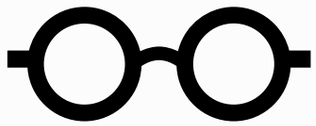
Supervised Learning
(learning from labeled data)

Unsupervised Learning
(learning from unlabeled data)

Reinforcement Learning
(autonomous learning)

Machine Learning for Software Dependability

Machine Learning Algorithms



Supervised Learning
(learning from labeled data)

60%

Unsupervised Learning
(learning from unlabeled data)

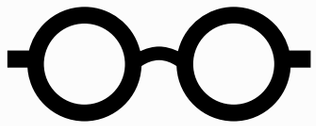
35%

Reinforcement Learning
(autonomous learning)

5%

Machine Learning for Software Dependability

Machine Learning Algorithms



Supervised Learning
(learning from labeled data)

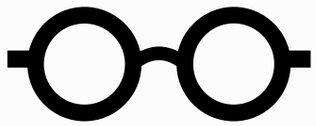
Mainly classification problems, where the goal is to predict the existence of a certain property.

Some examples:

1. Defect prediction;
2. Code smell prediction;
3. Vulnerability prediction;
4. …

Machine Learning for Software Dependability

Machine Learning Algorithms



Supervised Learning
(learning from labeled data)

Mainly classification problems, where the goal is to predict the existence of a certain property.

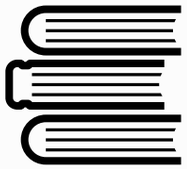
Some examples:

1. Defect prediction;
2. Code smell prediction;
3. Vulnerability prediction;
4. …

A few regression problems, mainly related to effort and cost estimation of specific software maintainability tasks, e.g., how much does a bug fixing operation cost?

Machine Learning for Software Dependability

Machine Learning Algorithms



Unsupervised Learning
(learning from unlabeled data)

Mainly clustering problems, e.g., refactoring, software remodularization, etc.

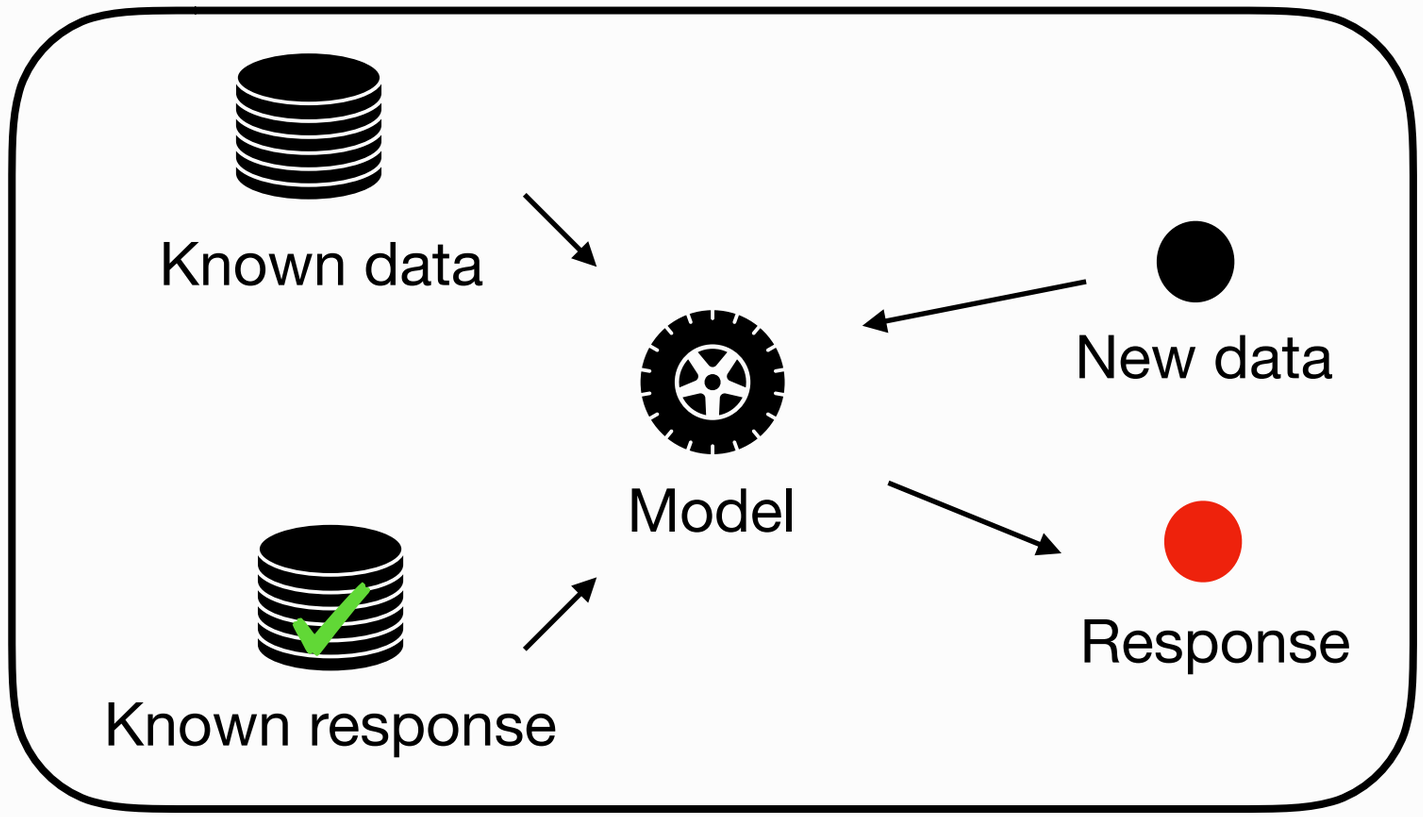
A few alternative applications, e.g., association rule mining.

Reinforcement Learning
(autonomous learning)

A relative new trend in software dependability

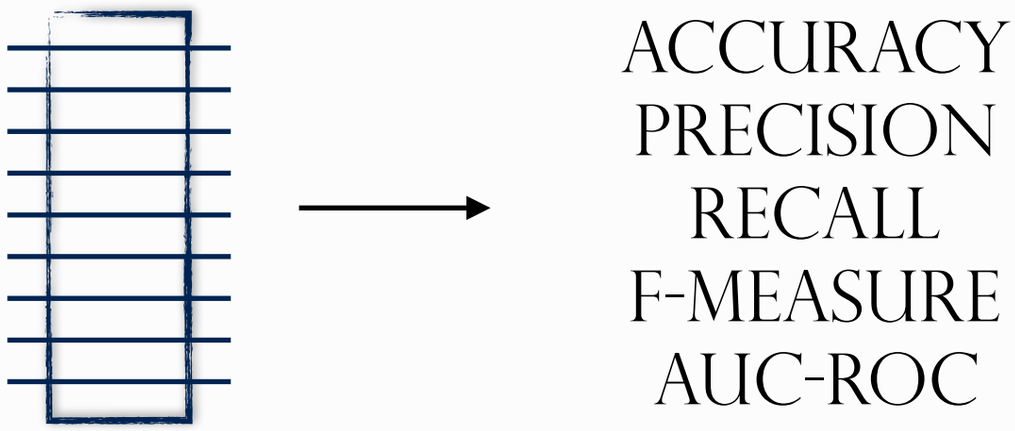
Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. Data cleaning;
3. Ground truth definition;
4. Model settings;
5. Model Construction.

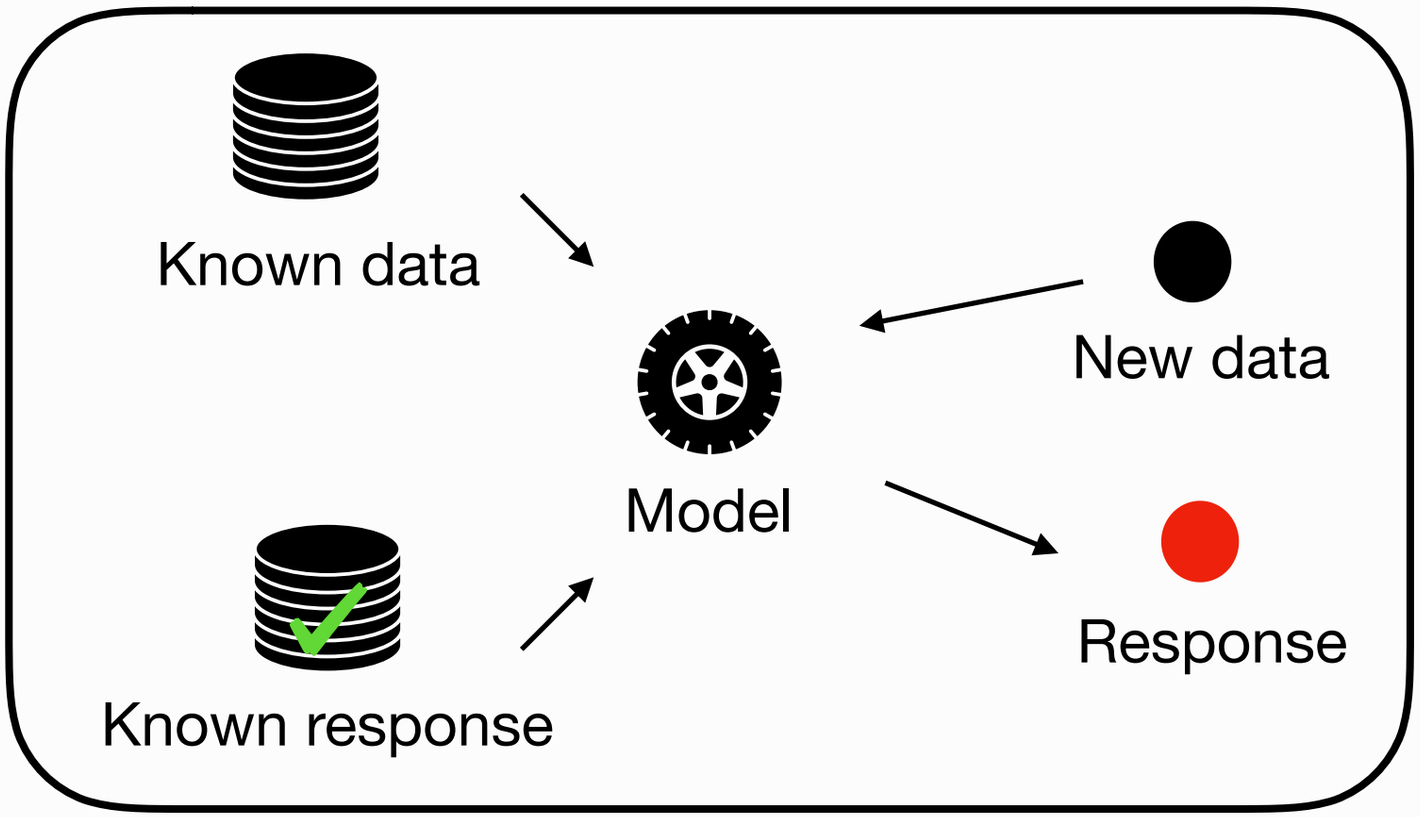


Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

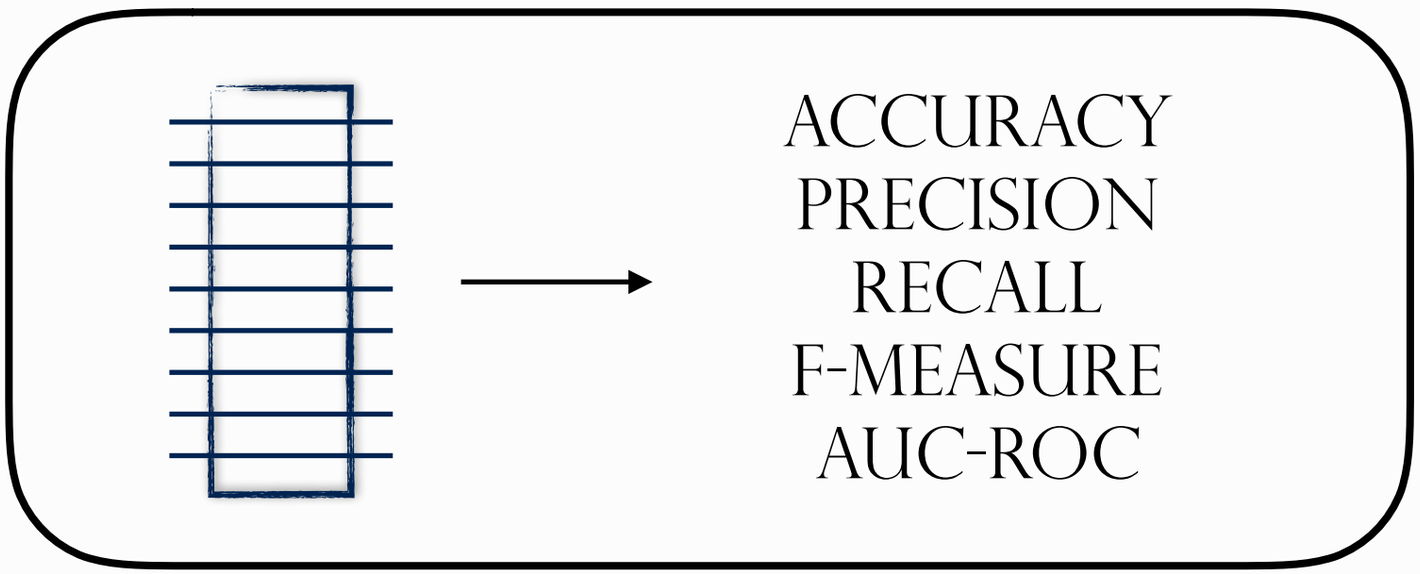
Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. **Data collection;**
2. **Data cleaning;**
3. **Ground truth definition;**
4. **Model settings;**
5. Model Construction.

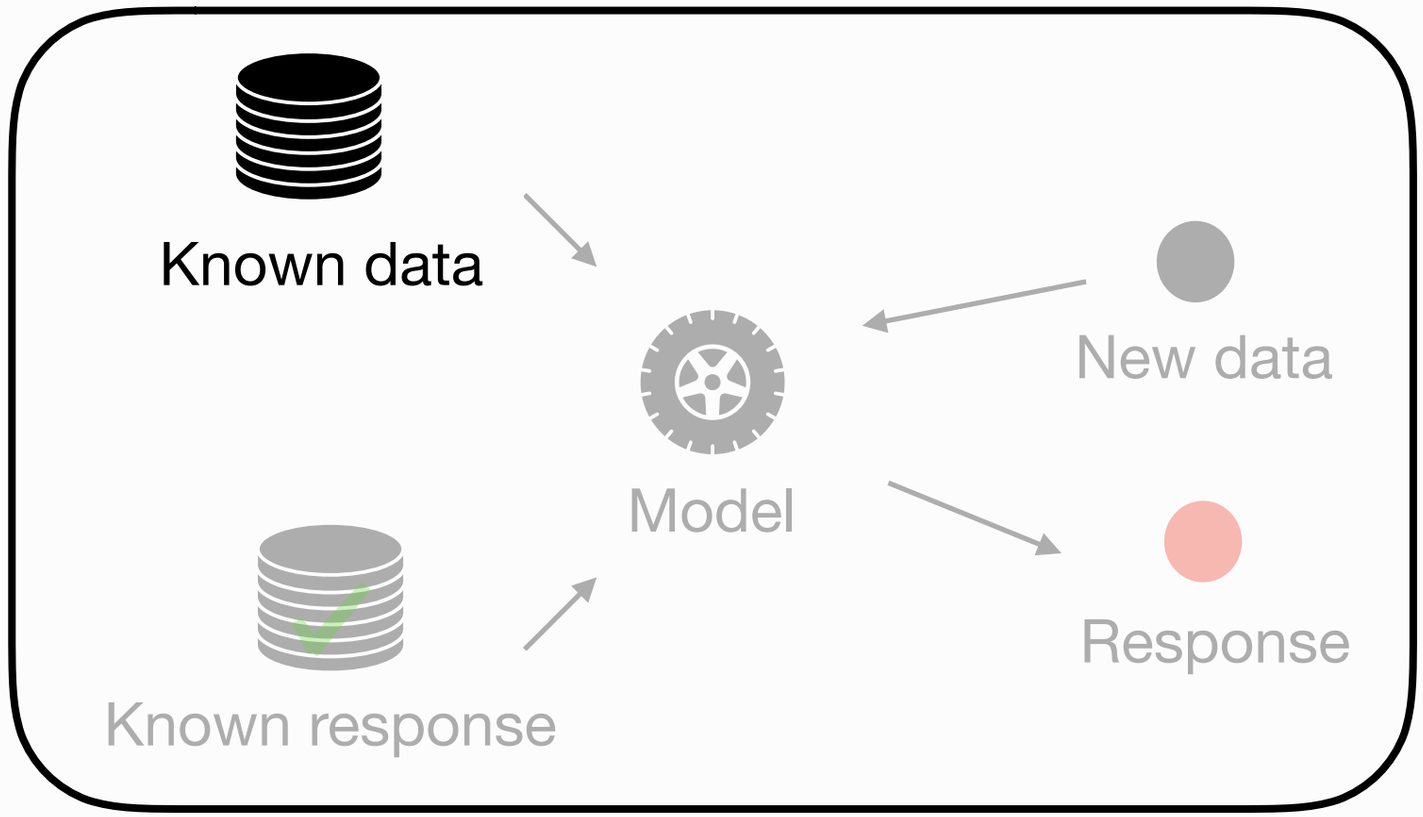


Evaluation phase:

1. **Validation strategy selection;**
2. Results interpretation.

Machine Learning for Software Dependability - A Typical Workflow

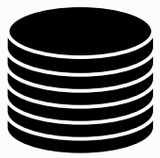
Supervised Methods for Software Dependability



Building phase:

1. **Data collection;**
2. Data cleaning;
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data collection



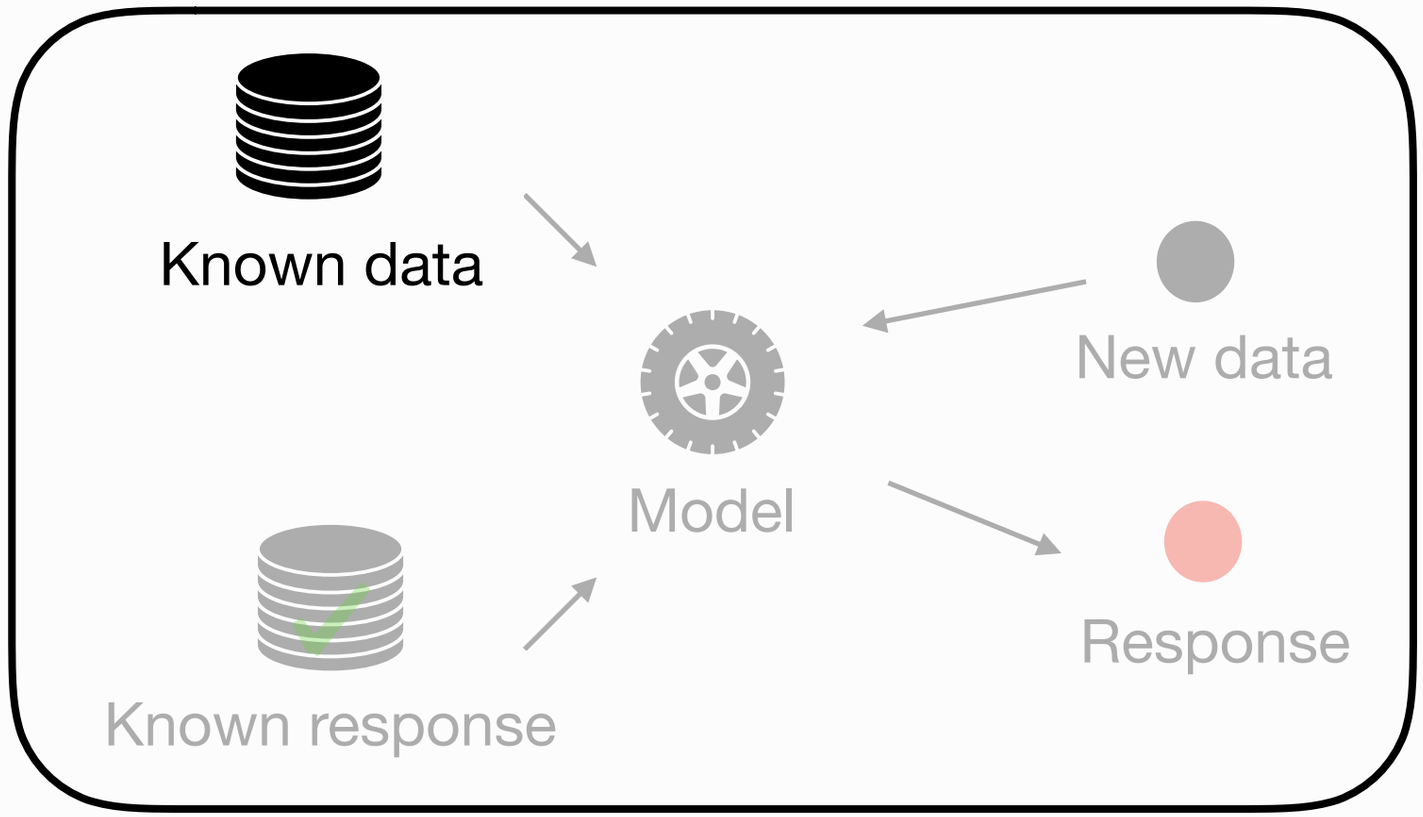
Known data

1. Define scope and goal of your problem:
   For certain problems or in certain contexts, you would need the collection of specific data.

2. Define the features required for the problem of interest:
   Structural, historical, semantic features? Which ones? Can be all extracted? How?

Machine Learning for Software Dependability - A Typical Workflow

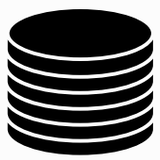
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



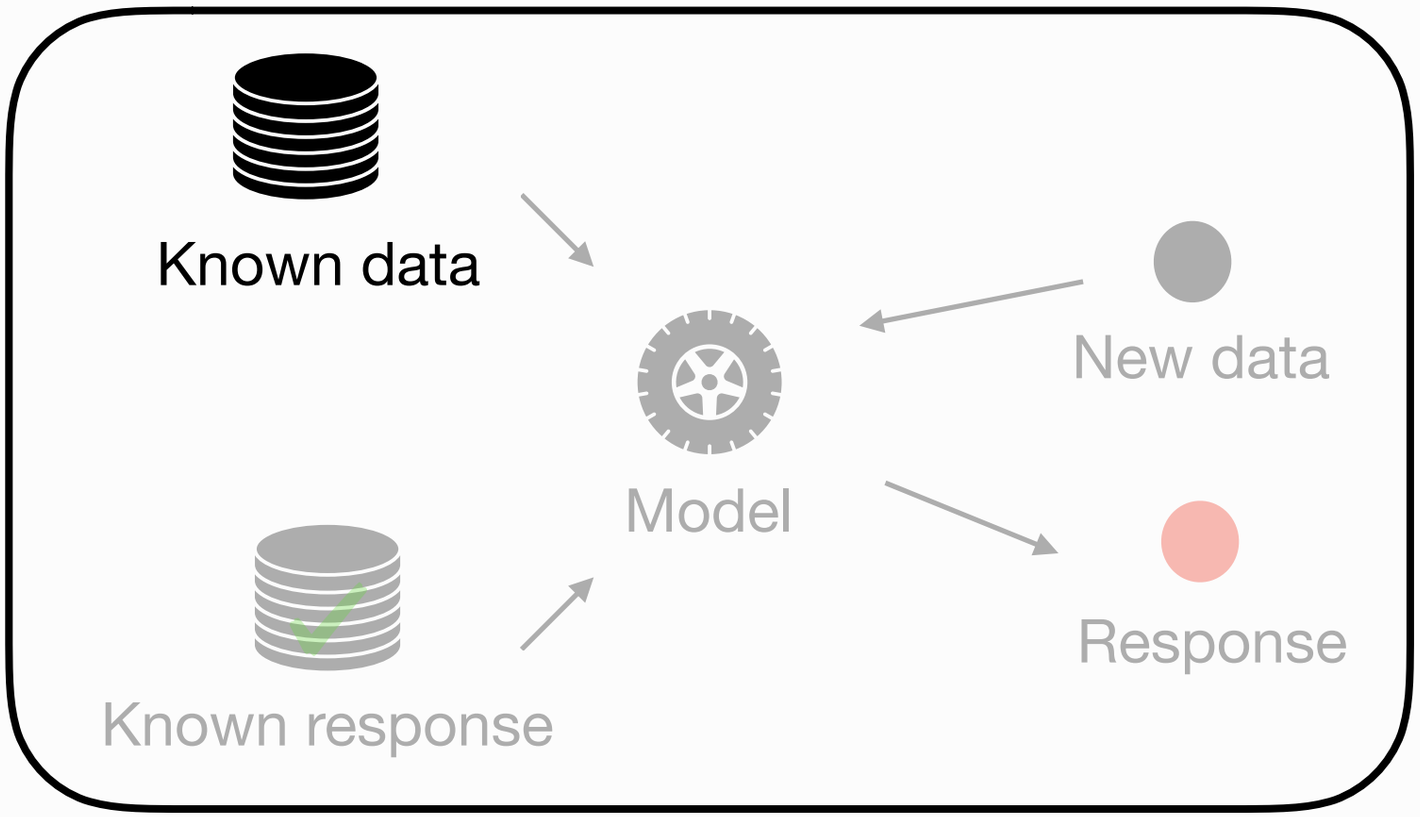
Known data

1. There are some key problems with data quality:
   - Missing data: The data extracted is incomplete, e.g., some values for a certain feature are not available;

How would you solve it?

Machine Learning for Software Dependability - A Typical Workflow

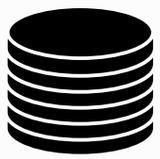
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



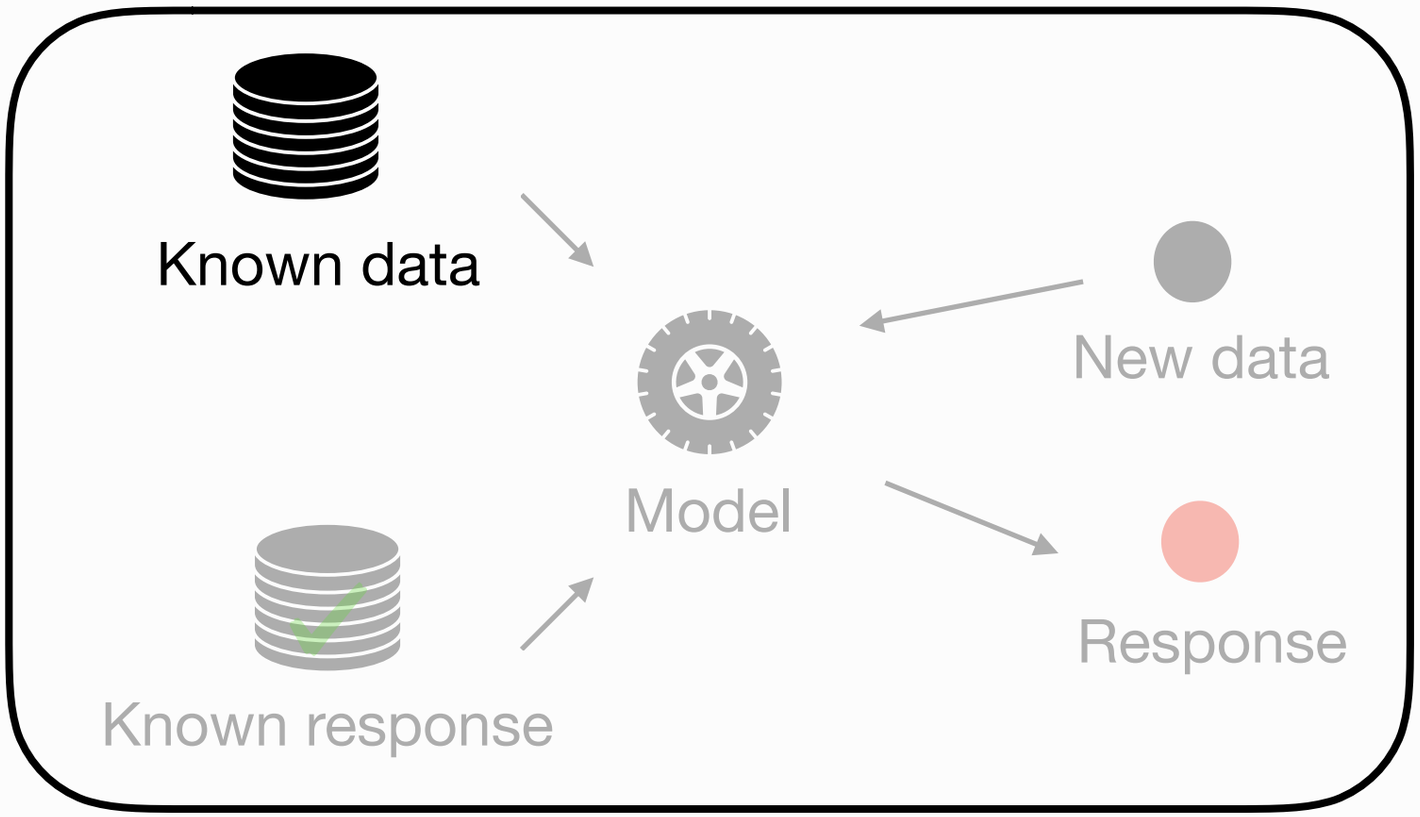
Known data

1. There are some key problems with data quality:
   - Missing data: The data extracted is incomplete, e.g., some values for a certain feature are not available;

Data Imputation Techniques.

Machine Learning for Software Dependability - A Typical Workflow

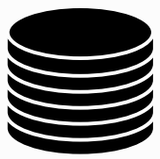
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



Known data

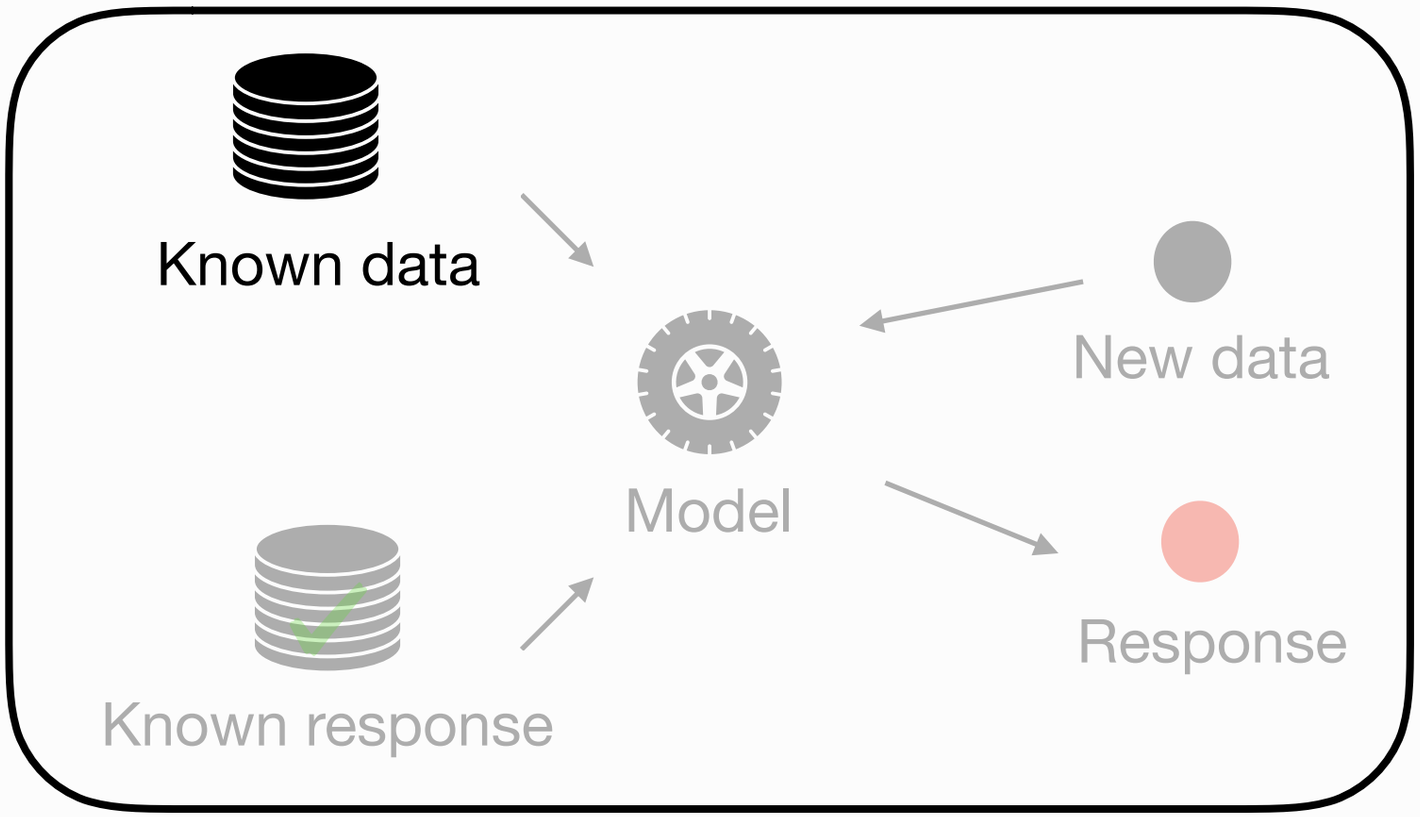
1. There are some key problems with data quality:
   - Missing data: The data extracted is incomplete, e.g., some values for a certain feature are not available;

Data Imputation Techniques.

- *Deductive Imputation*: an imputation rule defined by logical reasoning, as opposed to a statistical rule. For instance, if someone has 2 children in year 1, year 2 has missing values, and 2 children in year 3, we can reasonably impute that they have 2 children in year 2.

Machine Learning for Software Dependability - A Typical Workflow

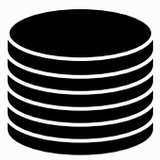
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



Known data

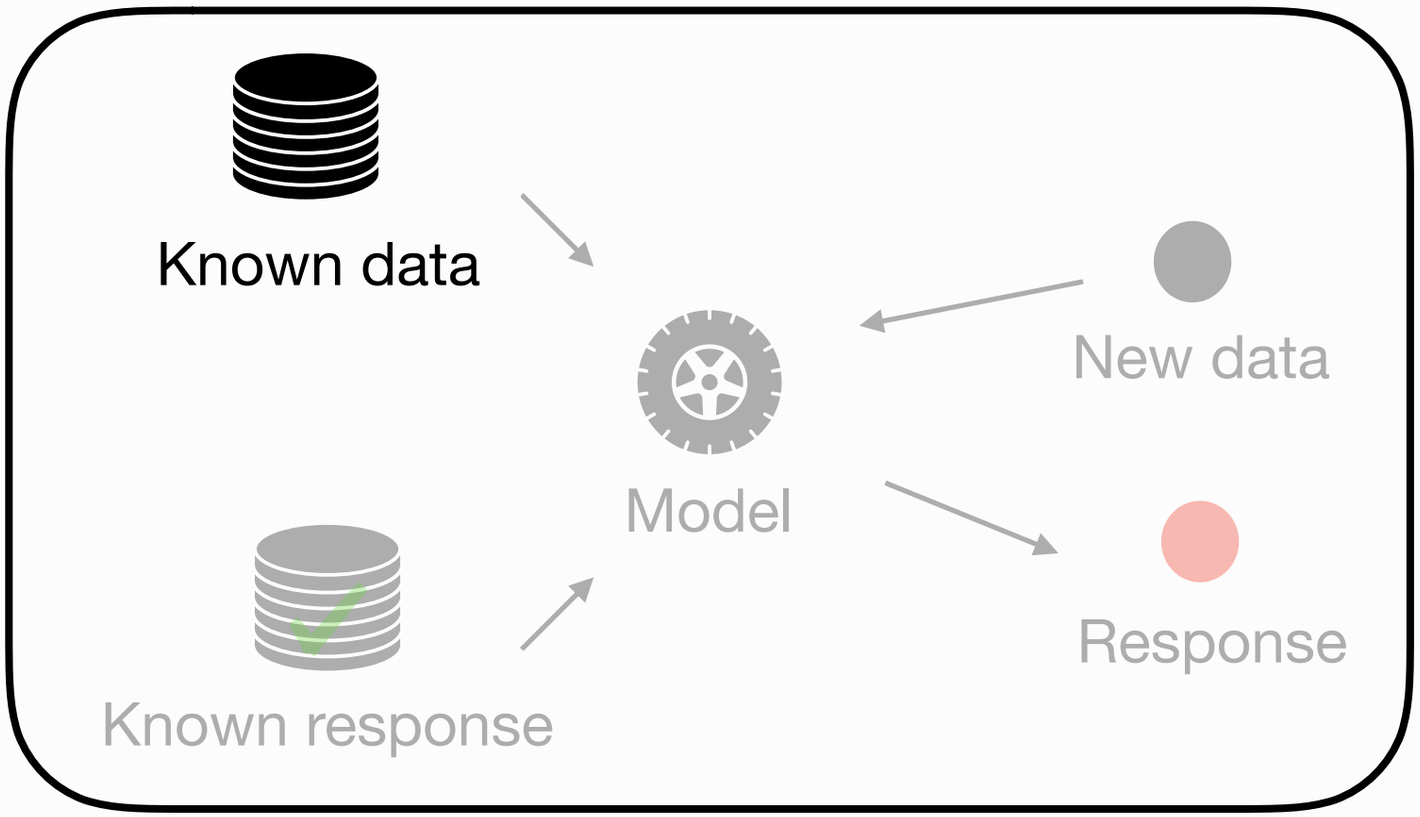
1. There are some key problems with data quality:
   - Missing data: The data extracted is incomplete, e.g., some values for a certain feature are not available;

Data Imputation Techniques.

- *Mean/Median/Mode Imputation*: an imputation rule defined though the analysis of the distribution of an attribute. For instance, a missing value is replaced by the mean of the distribution.

Machine Learning for Software Dependability - A Typical Workflow

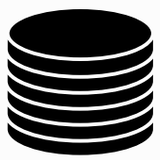
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



Known data

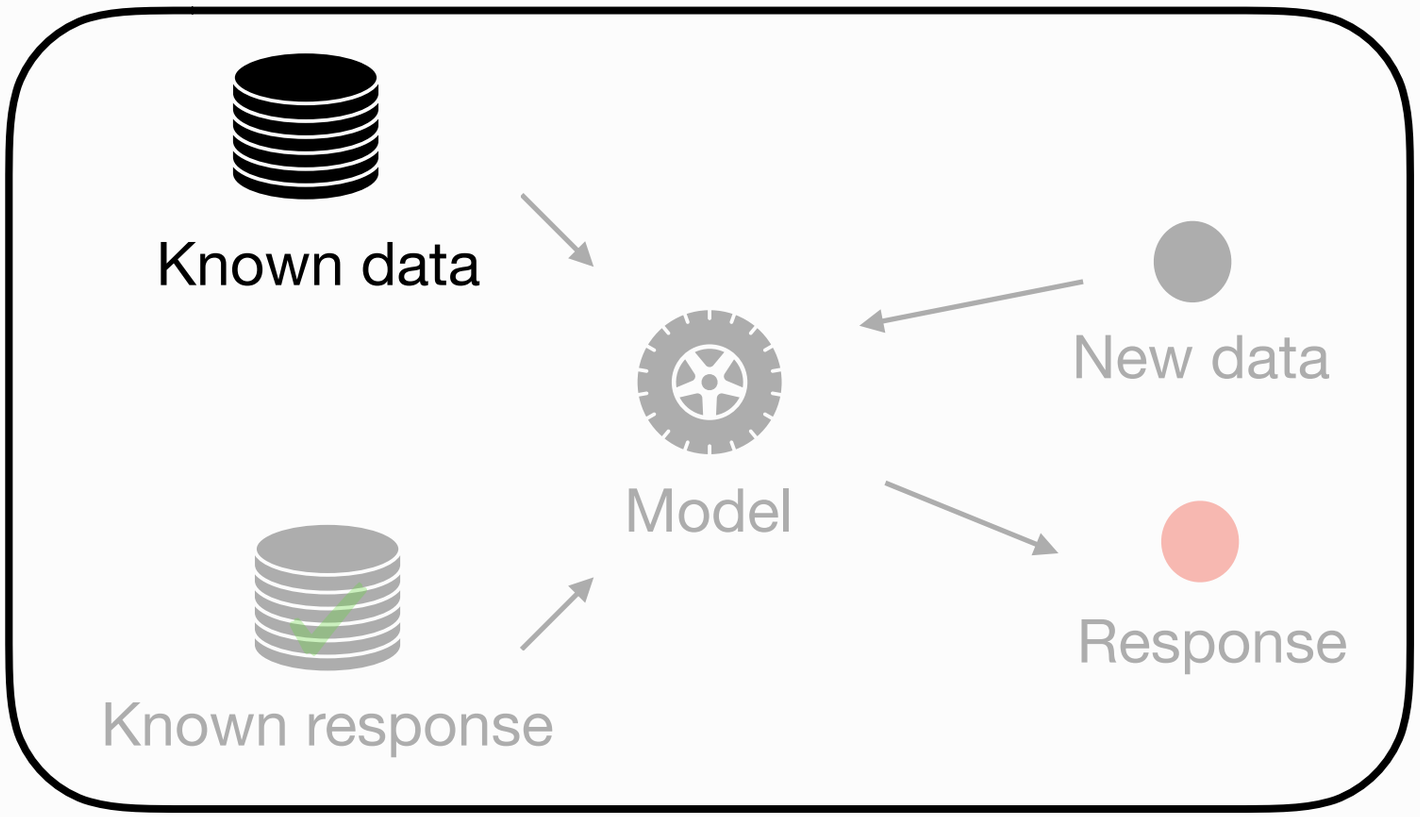
1. There are some key problems with data quality:
   - Variable independence: Each feature should not be too much correlated with the others.

Multicollinearity.

- If two or more variables are strongly related to each other, it means that they give the same information and the model may not be able to attribute a explanatory meaning to them, leading to biased results.

Machine Learning for Software Dependability - A Typical Workflow

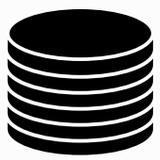
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



Known data

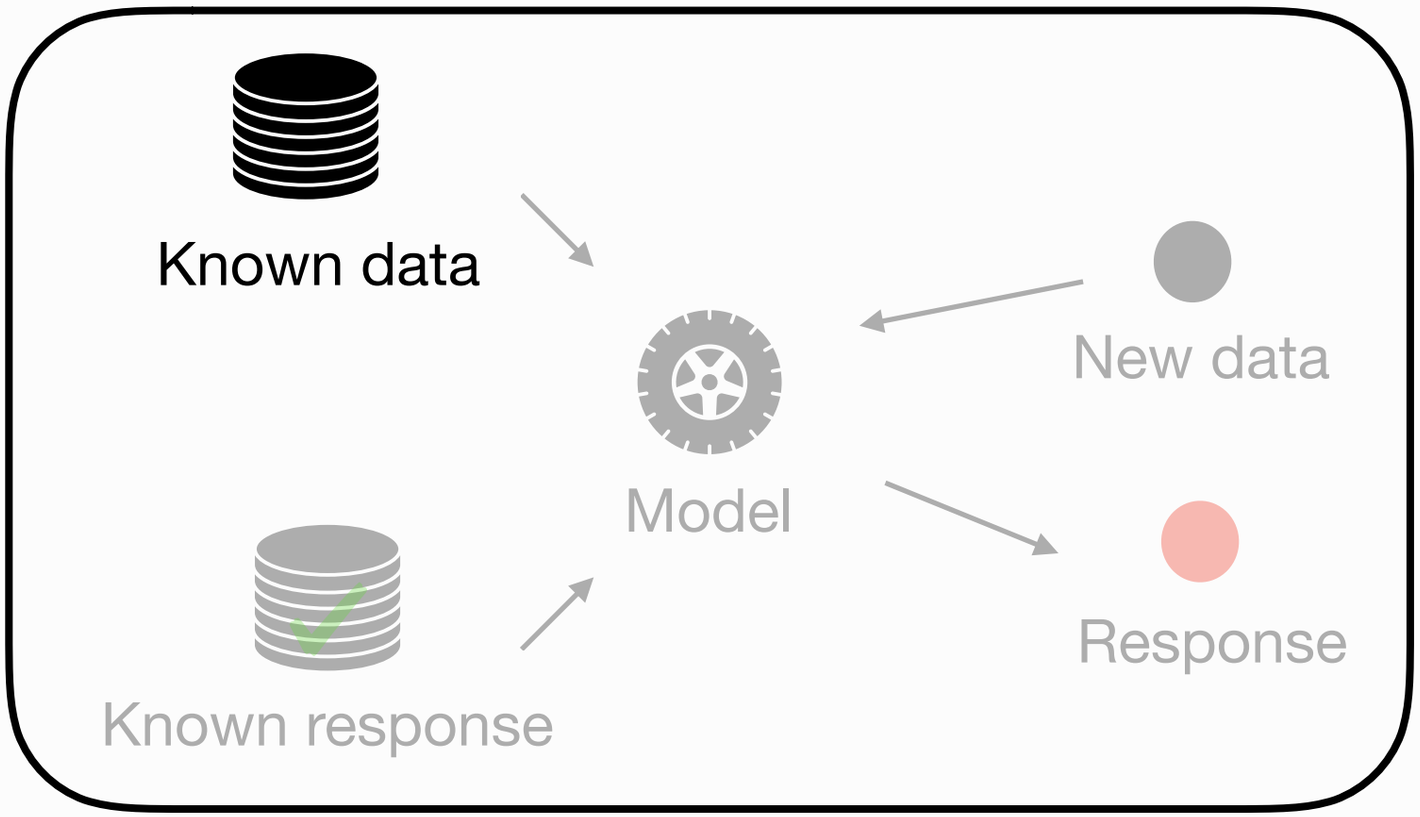
1. There are some key problems with data quality:
   - Variable independence: Each feature should not be too much correlated with the others.

Multicollinearity.

How would you solve it?

Machine Learning for Software Dependability - A Typical Workflow

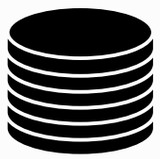
Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. **Data cleaning;**
3. Ground truth definition;
4. Model settings;
5. Model Construction.

Data cleaning



Known data

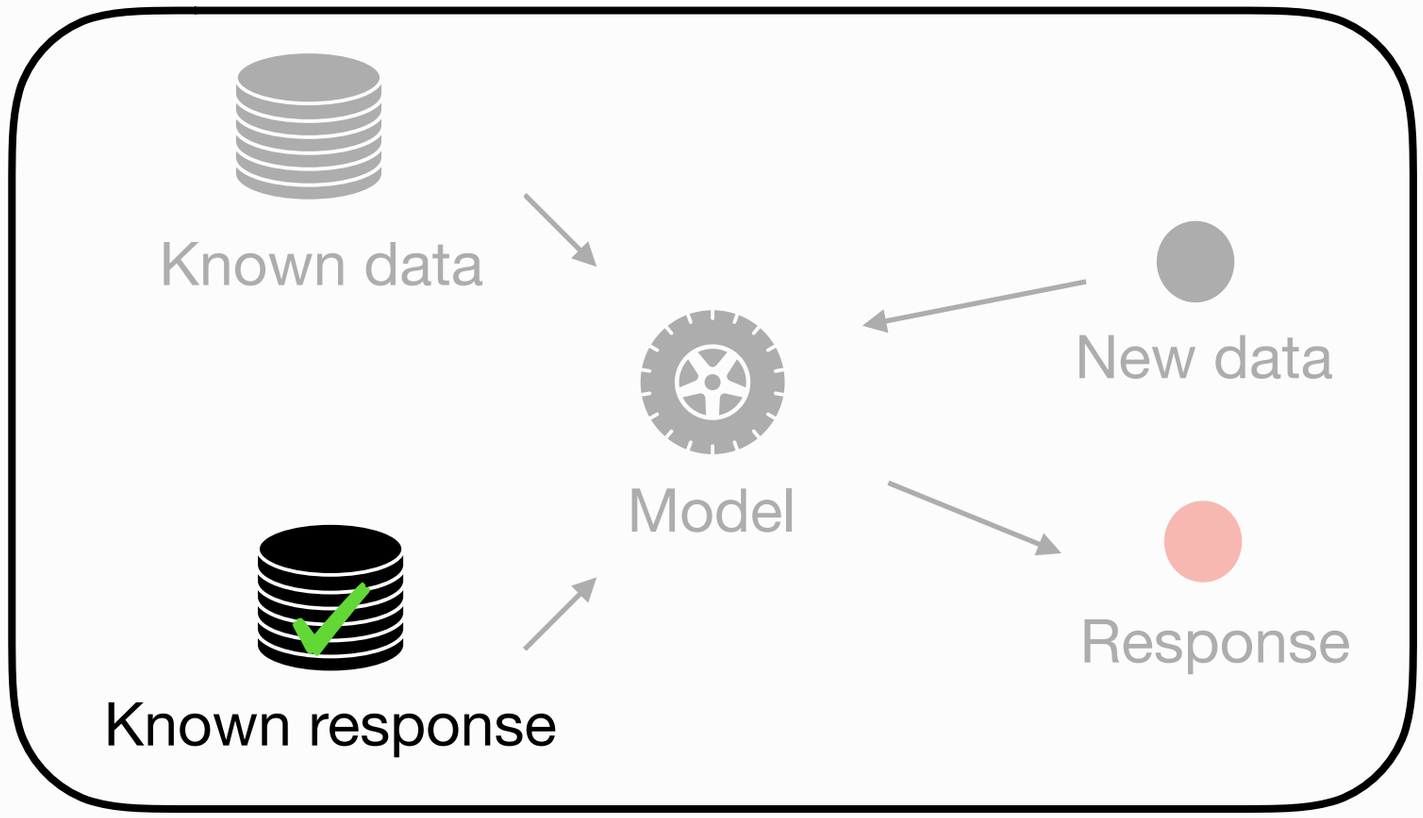
1. There are some key problems with data quality:
   - Variable independence: Each feature should not be too much correlated with the others.

Multicollinearity.

- Different methods available. One of them is correlation analysis: if two variables have a correlation higher than, e.g., 0.6, then one of them should be excluded. Usually, it is discarded the most complicated feature to compute/explain.

Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. Data cleaning;
3. **Ground truth definition;**
4. Model settings;
5. Model Construction.

Defining the oracle

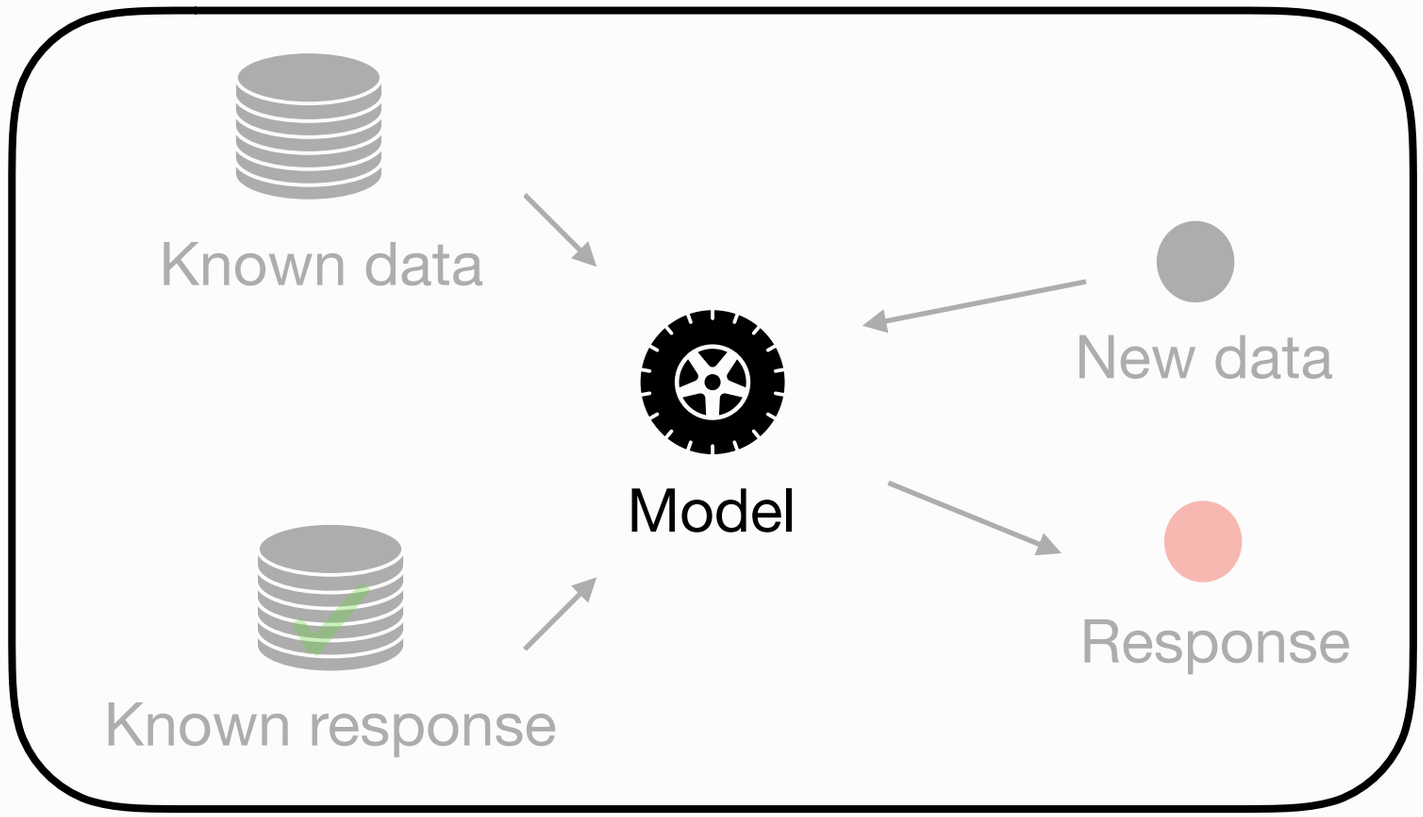
Known response

1. The most challenging part:
   Is previous valid data available? If not, may I use a cross-project strategy? If not, can I do manual labeling?

2. The reliability of the information is crucial:
   For manual analysis, a common solution is to have a pool of experts performing the task.

Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. Data cleaning;
3. Ground truth definition;
4. **Model settings;**
5. Model Construction.

Data collection

Model

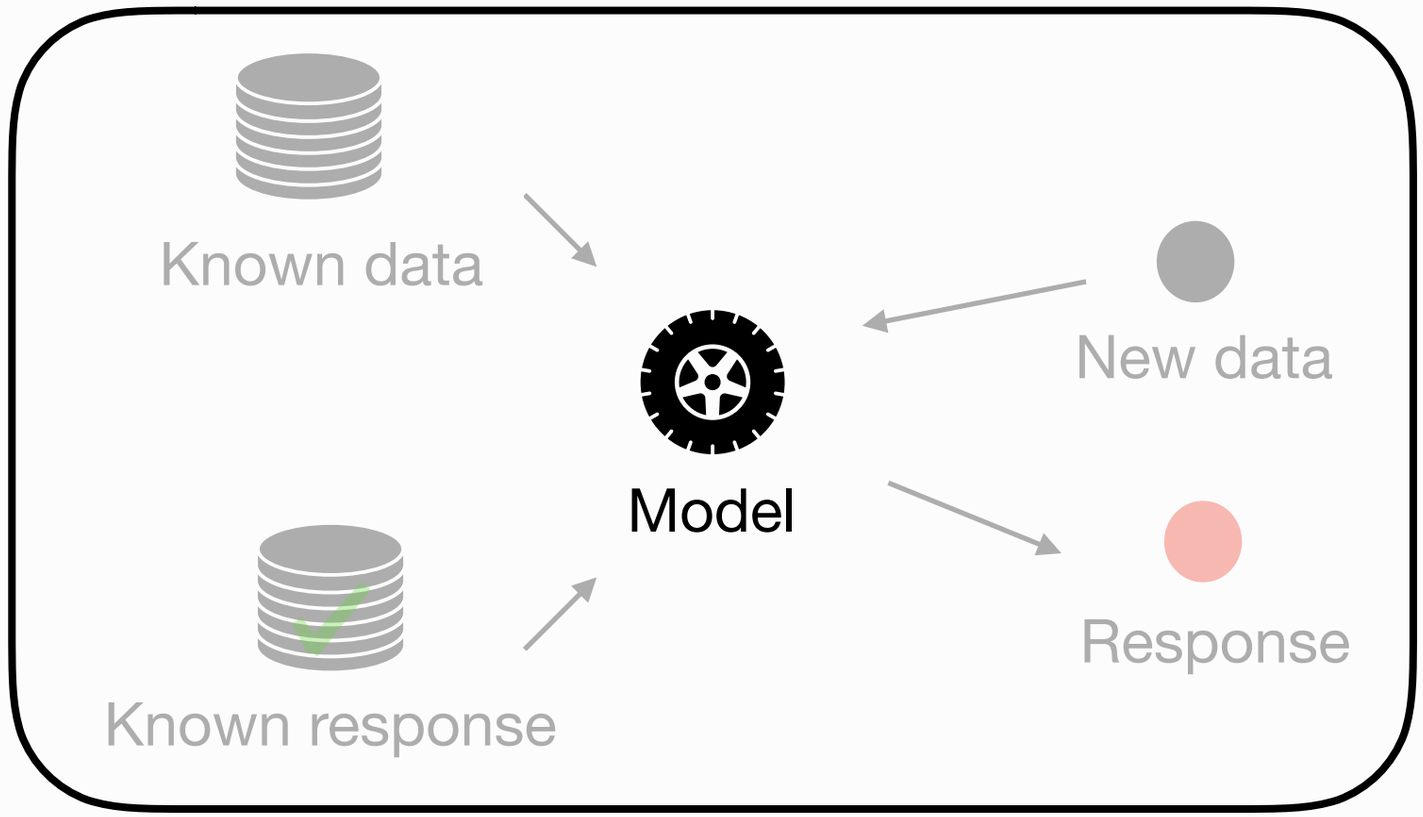
1. Some precautions should be taken:
   - Classifier configuration: Some machine learning algorithms require the specification of hyper-parameters.

Configuring the model.

- Default configuration. While it is often available, it may not properly fit the specific data under consideration.

Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. Data cleaning;
3. Ground truth definition;
4. **Model settings;**
5. Model Construction.

Data collection

Model

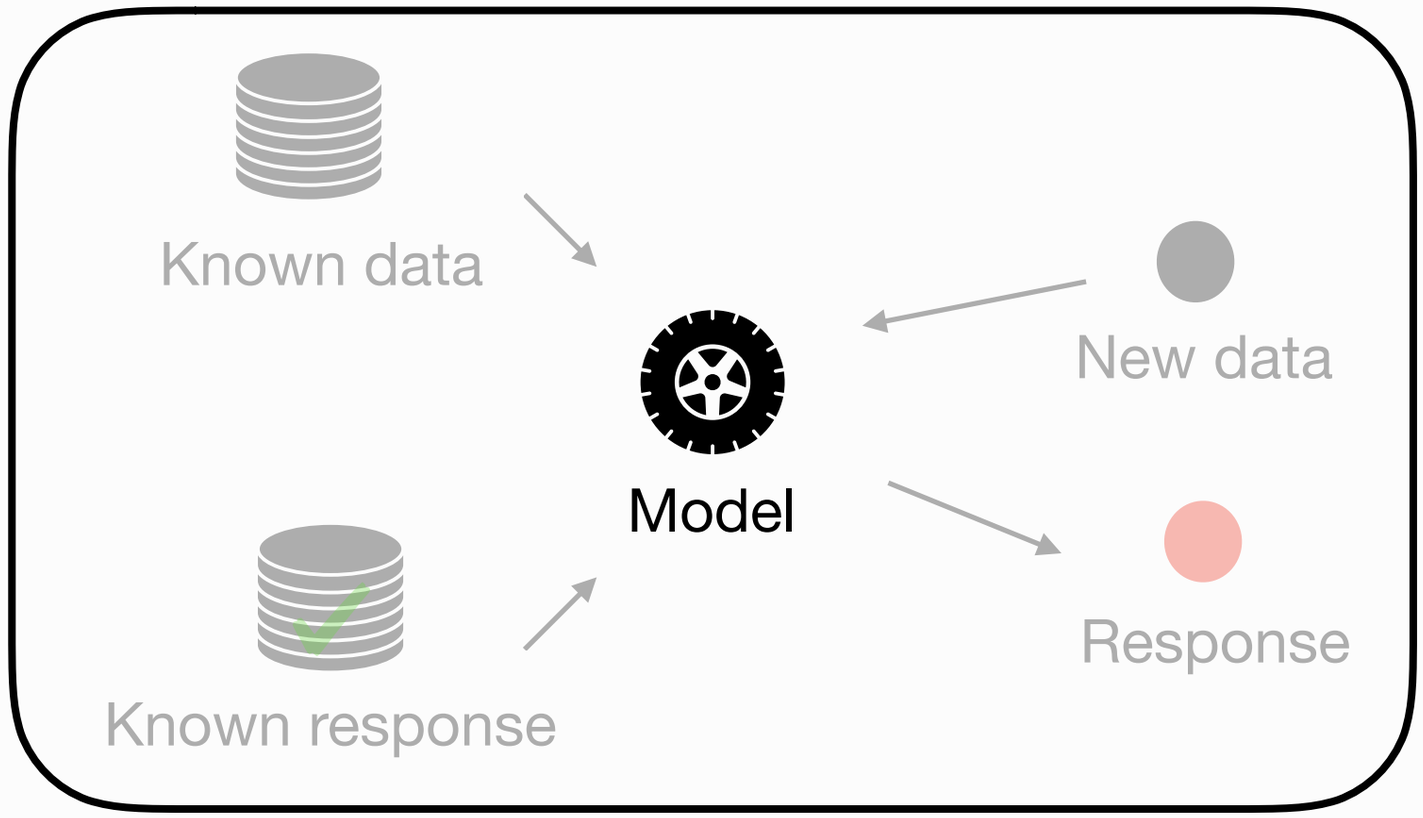
1. Some precautions should be taken:
   - Classifier configuration: Some machine learning algorithms require the specification of hyper-parameters.

Configuring the model.

- Configuration algorithms. There are several algorithms that deal with the problem. The easiest one is the Grid Search, that implements an exhaustive searching approach of the hyper-parameter space.

Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. Data cleaning;
3. Ground truth definition;
4. **Model settings;**
5. Model Construction.

Data collection

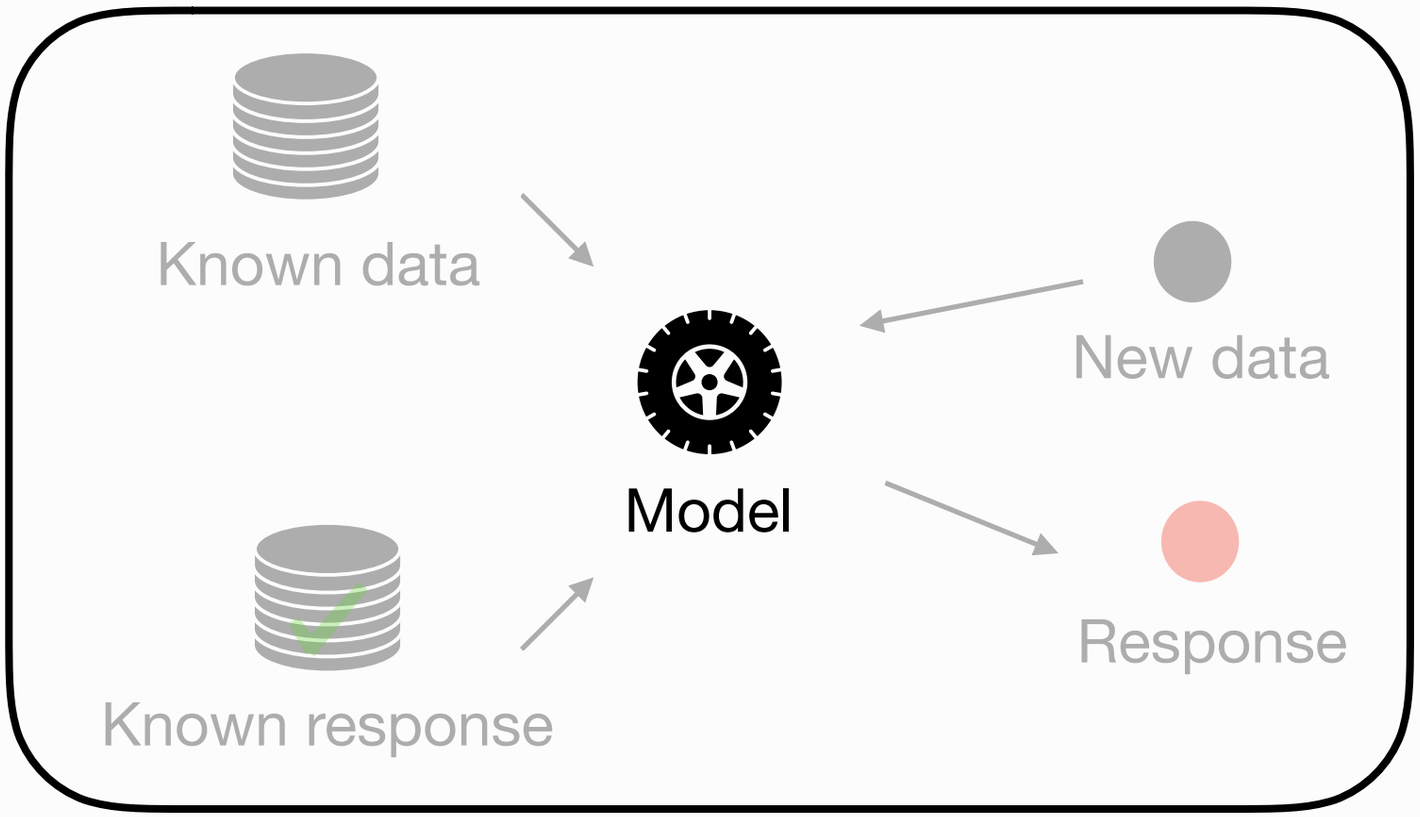
Model

1. Some precautions should be taken:
   - Data Balancing: To use when a classifier does not have enough data to classify the response variable.

How would you solve it?

Machine Learning for Software Dependability - A Typical Workflow

Supervised Methods for Software Dependability



Building phase:

1. Data collection;
2. Data cleaning;
3. Ground truth definition;
4. **Model settings;**
5. Model Construction.

Data collection

Model

1. Some precautions should be taken:
   - Data Balancing: To use when a classifier does not have enough data to classify the response variable.

Balancing (ONLY) the training set.

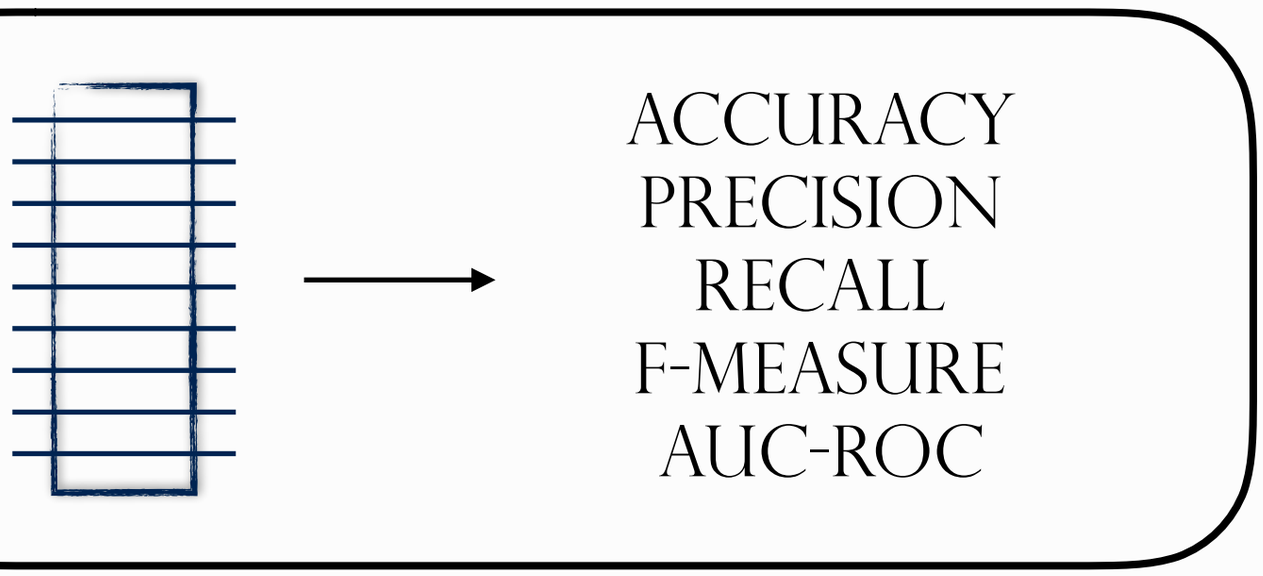
- Balancing algorithms. There are several algorithms that deal with the problem. One of the most famous is SMOTE, that creates synthetic instances of the minority class using statistical methods and analysis of the distribution.

Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Supervised Methods for Software Dependability



Evaluation phase:

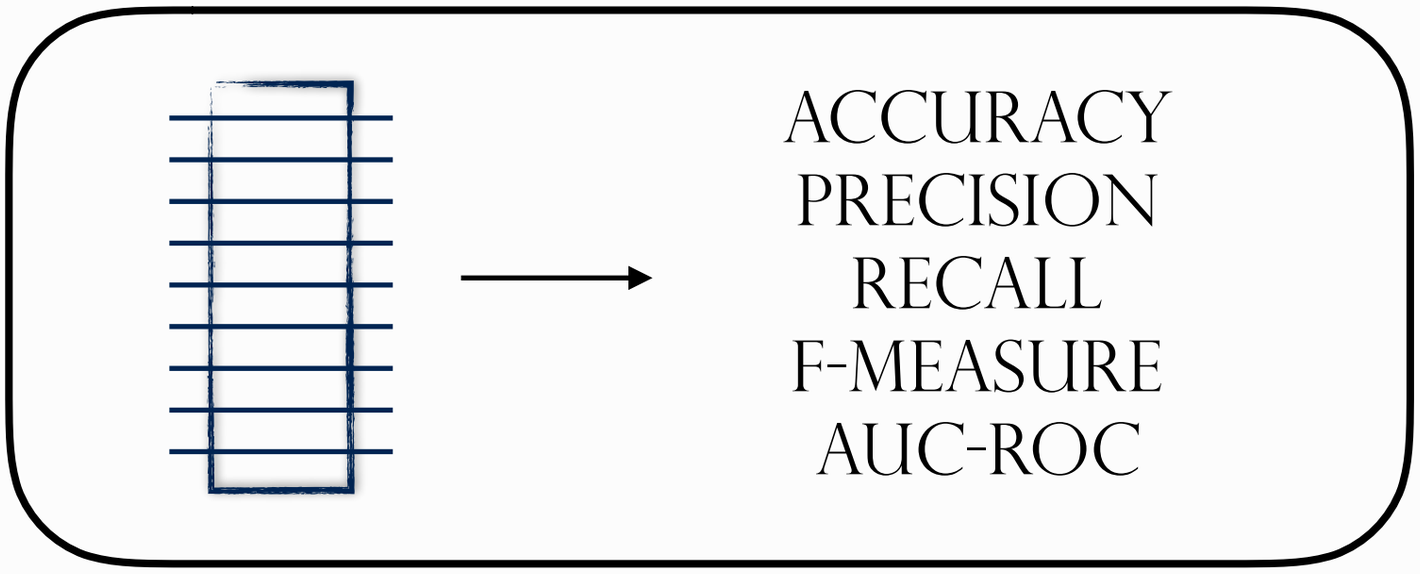
1. Validation strategy selection;
2. Results interpretation.

Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Supervised Methods for Software Dependability



Evaluation phase:

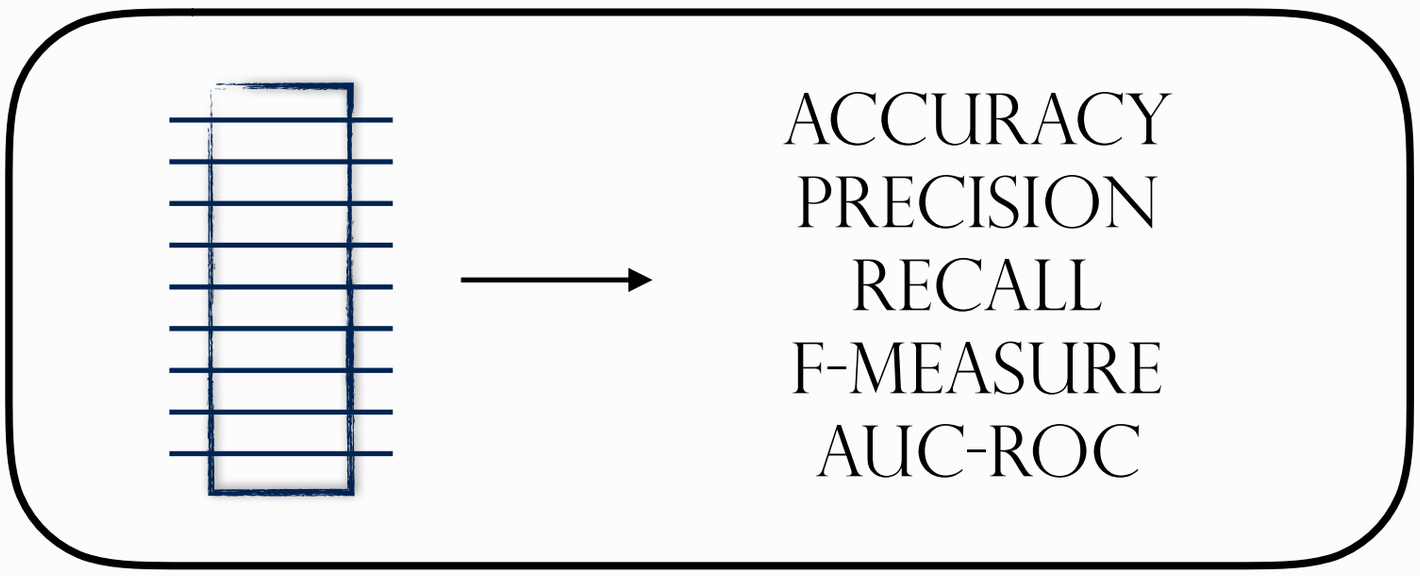
1. Validation strategy selection;
2. Results interpretation.

Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Supervised Methods for Software Dependability

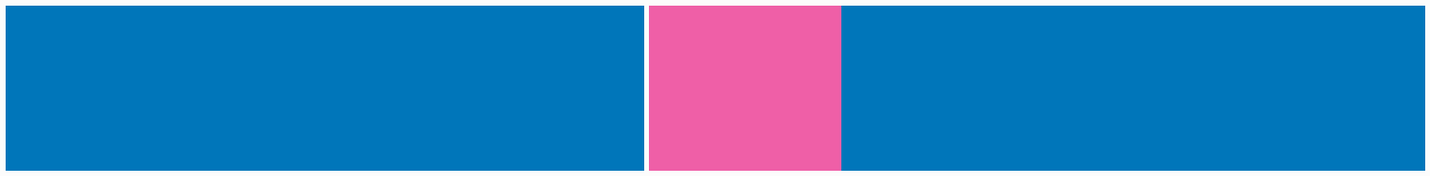


Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

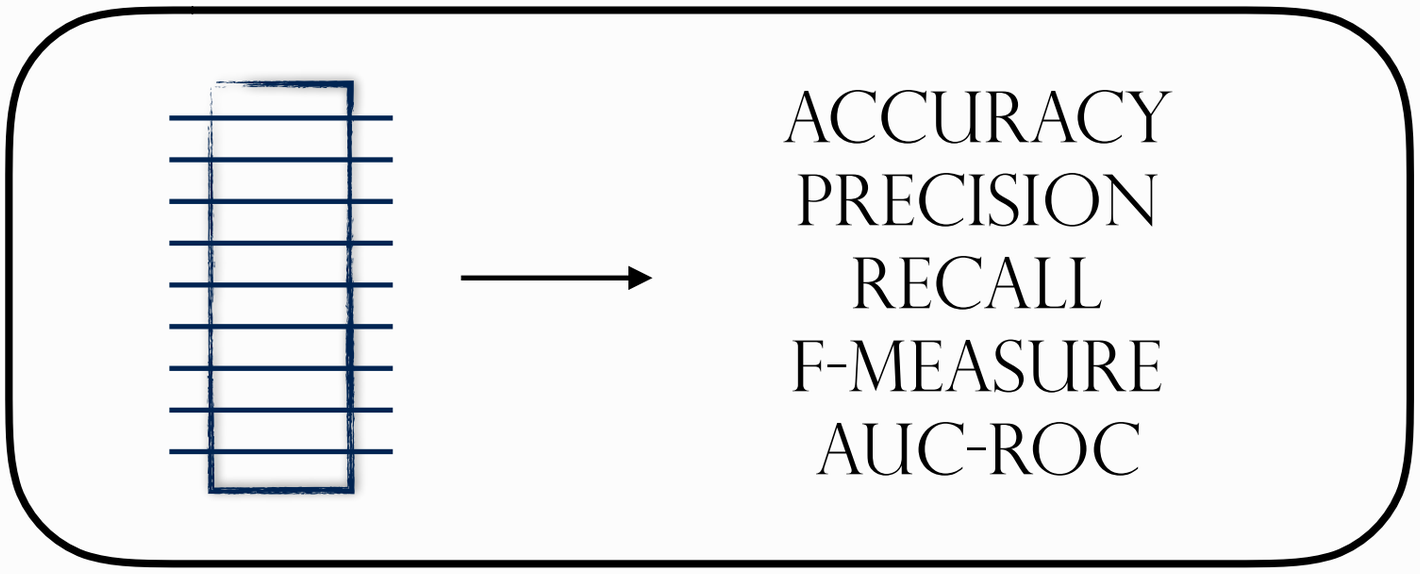
Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.



The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Supervised Methods for Software Dependability



Evaluation phase:

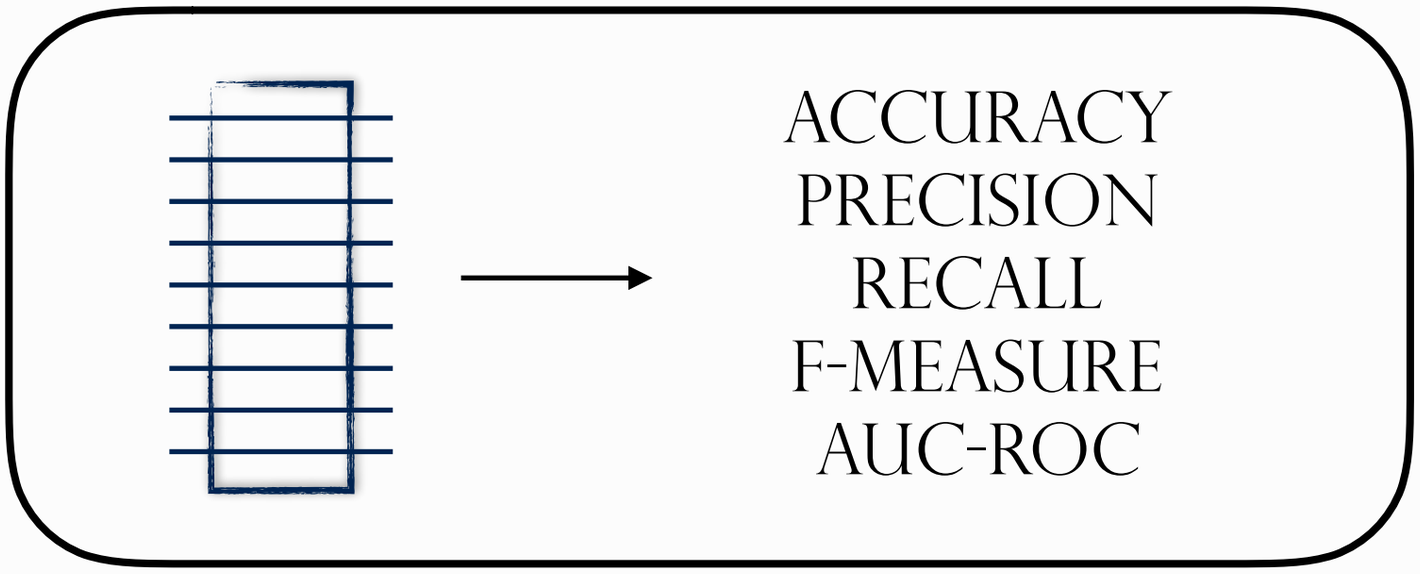
1. Validation strategy selection;
2. Results interpretation.

Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Supervised Methods for Software Dependability



Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

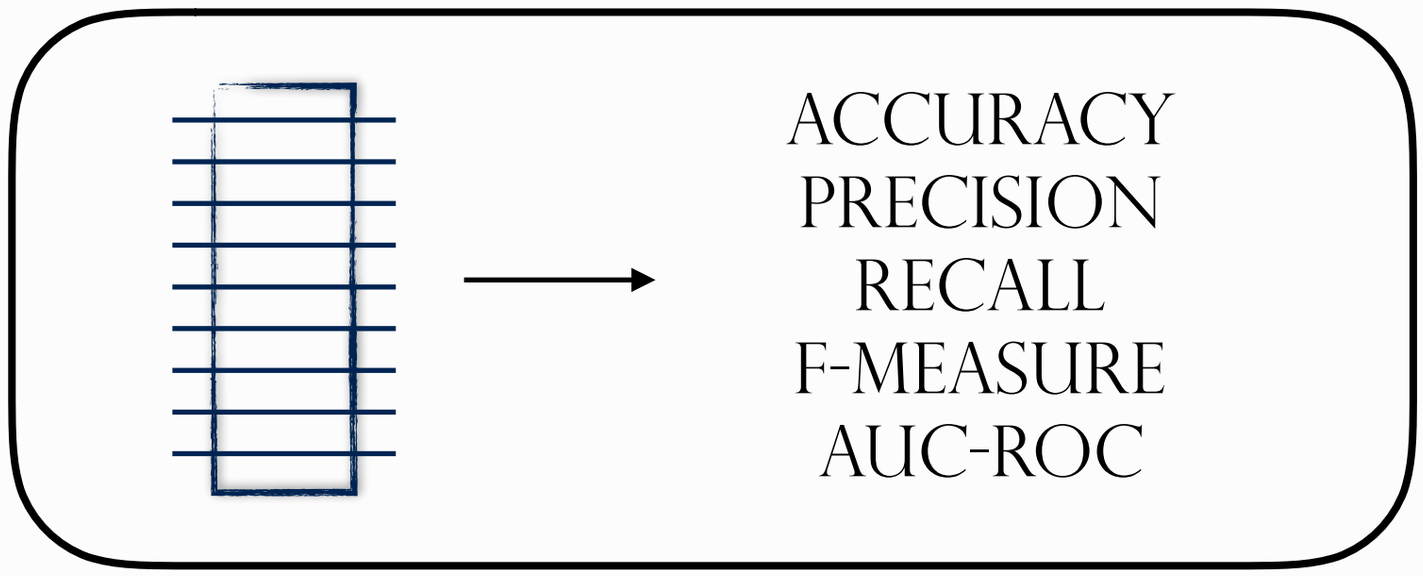
Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

What's wrong with that?

Supervised Methods for Software Dependability



Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

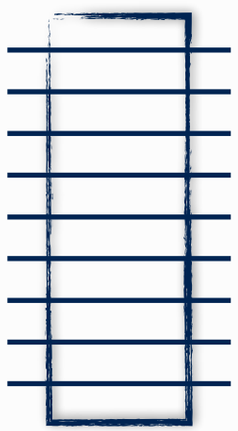
Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Depending on the initial random splitting, the process may lead to over-estimate or under-estimate the real performance of a model.

Supervised Methods for Software Dependability



ACCURACY
PRECISION
RECALL
F-MEASURE
AUC-ROC

Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

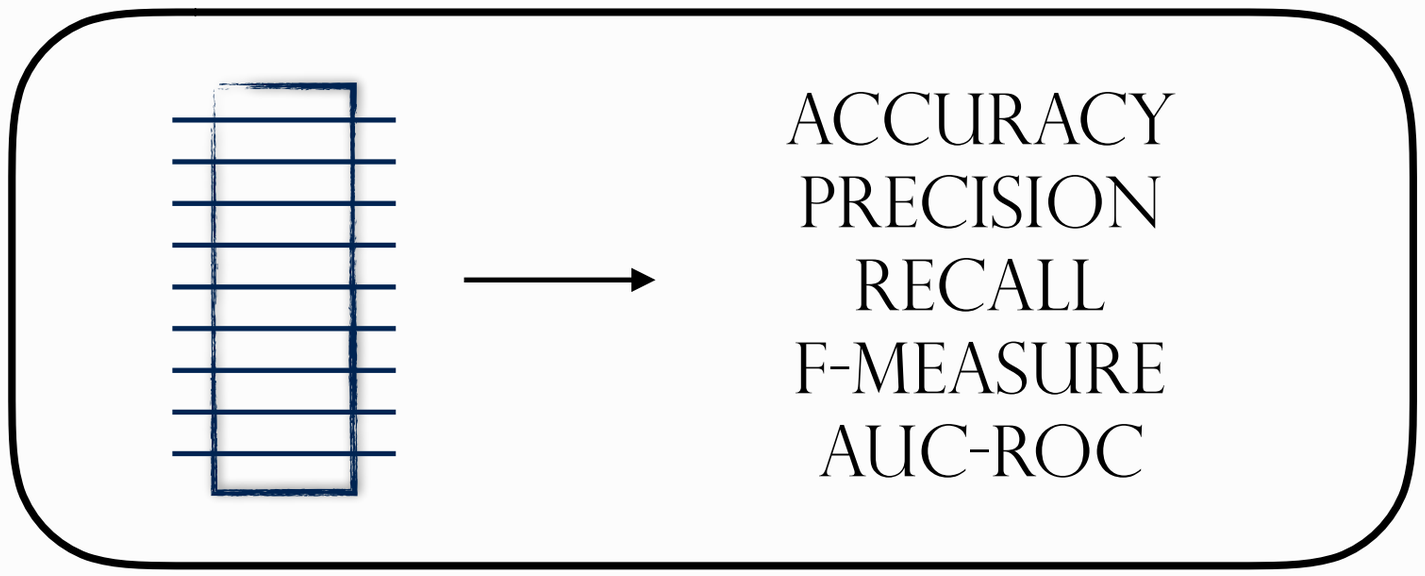
Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

How would you solve it?

Supervised Methods for Software Dependability



Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

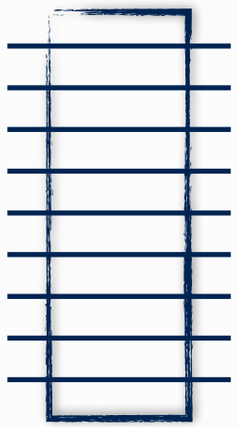
Machine Learning for Software Dependability - A Typical Workflow

Most of the software dependability machine learning models have been assessed using the 10-fold cross validation strategy.

The dataset is randomly partitioned in ten folds. Nine of them are then used as training set, while one is retained as test set. The process is then repeated ten times, so that each fold is used as test once.

Repeat the validation process multiple times, so that the random effect is mitigated. For instance, perform a 10 times 10-fold cross validation.

Supervised Methods for Software Dependability



ACCURACY
PRECISION
RECALL
F-MEASURE
AUC-ROC

Evaluation phase:

1. Validation strategy selection;
2. Results interpretation.

End of the first part

Machine Learning for Software Dependability in Practice

Applying machine learning techniques to detect code smells

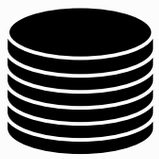
Install the R toolkit - https://www.r-project.org
A Language and environment for statistical computing and graphics

Install the Caret package
A Classification and Regression Training library for R

```
install.packages("caret", dependencies=c(“Depends”, “Suggests”))
```

Doc: https://cran.r-project.org/web/packages/caret/caret.pdf

Known response



Known data

Get the dataset from the Github page of the course