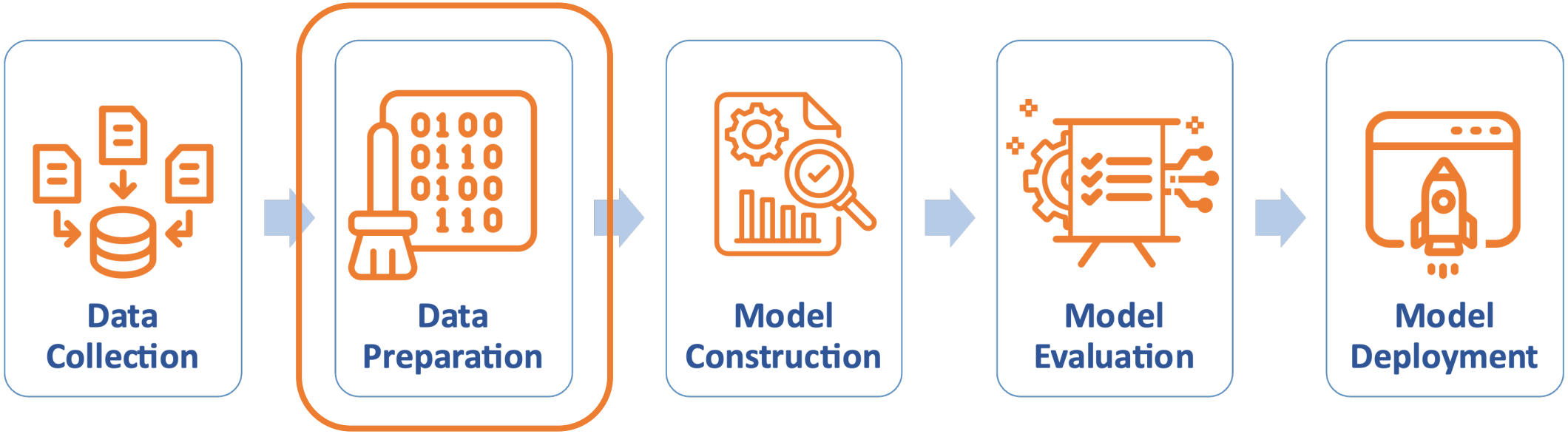
Certain and Approximately Certain Models for Statistical Learning

Cheng Zhen, Nischal Aryal, **Arash Termehchy**, Amandeep Singh Chabada

Oregon State University

Data preparation is important for ML but expensive



Most data scientists spend ~ 80% of their time preparing data

Cleaning missing data is important in data preparation for ML

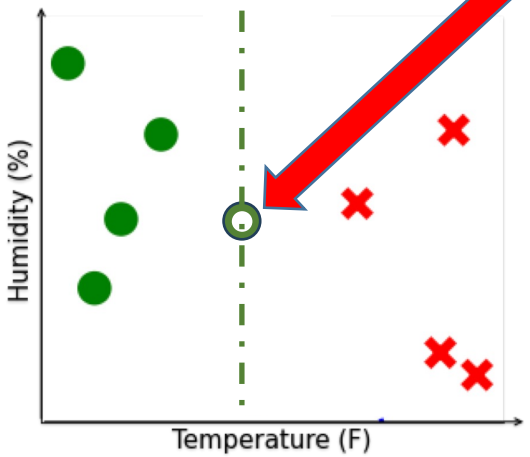
Missing Data

City	Temperature (F)	Humidity (%)	Rain (1) or no rain (-1)
Seattle	65	80	1
Portland	null	30	-1
San Diego	60	null	1

Predict whether rains or not

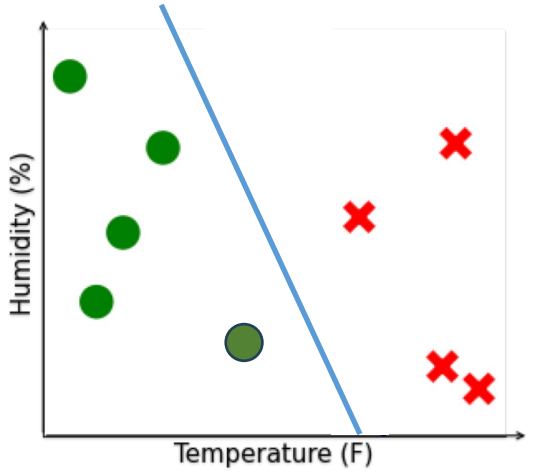
Rain No rain

Data point with missing humidity

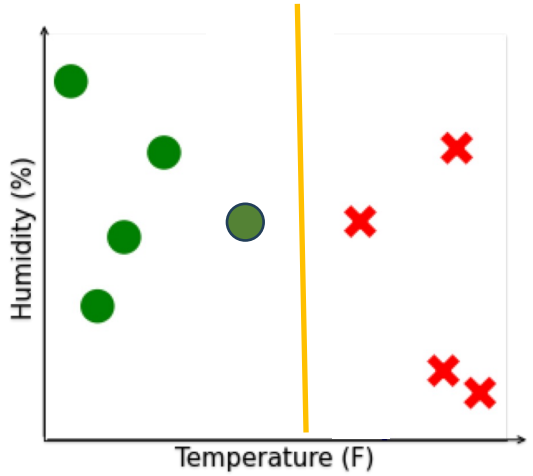


raw data

Need to clean missing data to learn an accurate ML model



correct cleaning → correct model

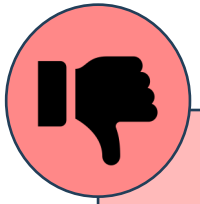


incorrect cleaning → incorrect model

Deleting records with missing values

City	Temperature (F)	Humidity (%)	Rain (1) or no rain (-1)
Seattle	65	80	1
Portland	null	30	-1
San Francisco	54	45	-1
San Diego	60	null	1

City	Temperature (F)	Humidity (%)	Rain (1) or no rain (-1)
Seattle	65	80	1
San Francisco	54	45	-1



- Loss of valuable information
- Might introduce bias

Data imputation

City	Temperature (F)	Humidity (%)	Rain (1) or no rain (-1)
Seattle	65	80	1
Portland	**null**	30	-1
San Francisco	54	45	-1
San Diego	60	**null**	1

City	Temperature (F)	Humidity (%)	Rain (1) or no rain (-1)
Seattle	65	80	1
Portland	50	30	-1
San Francisco	54	45	-1
San Diego	60	70	1

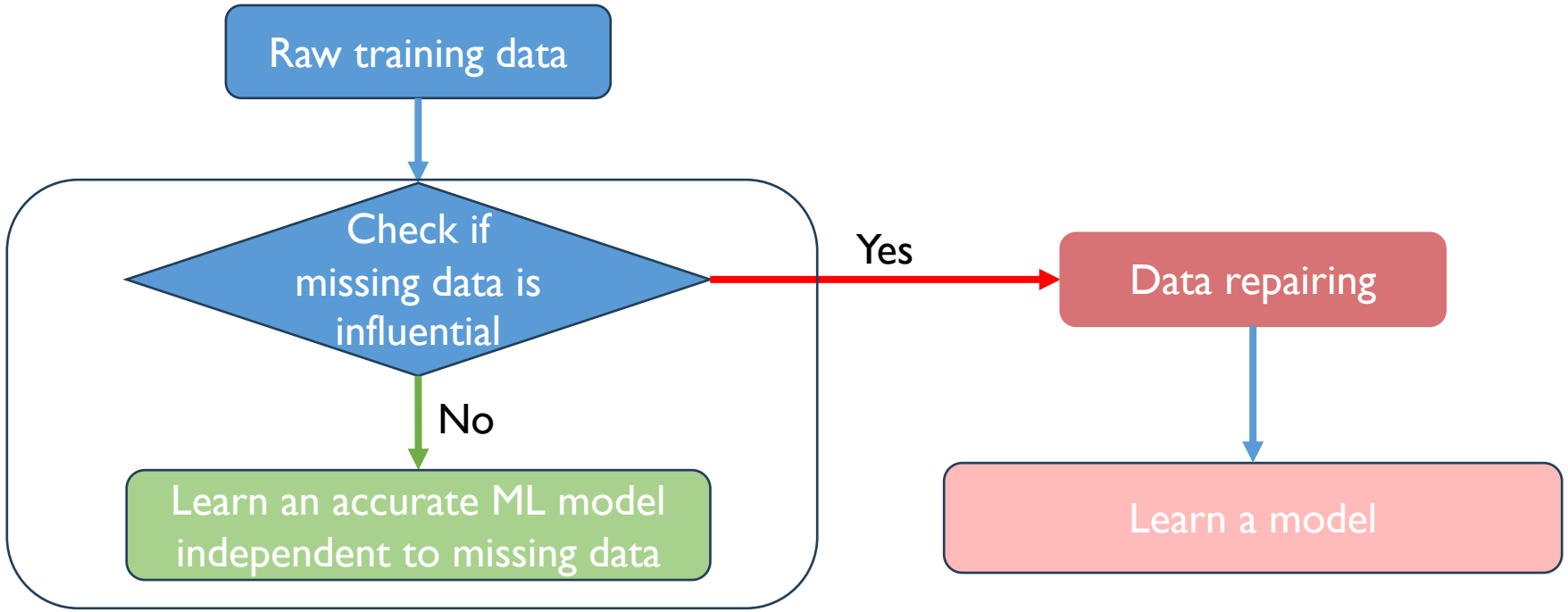
Repaired data

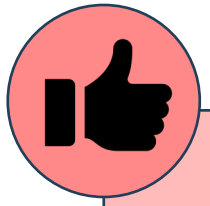
- High cost: resources to impute data or train an imputation model
- Not clear if repaired data is accurate

Our question:

When can we learn accurate ML models without cleaning missing data ?

Our proposal



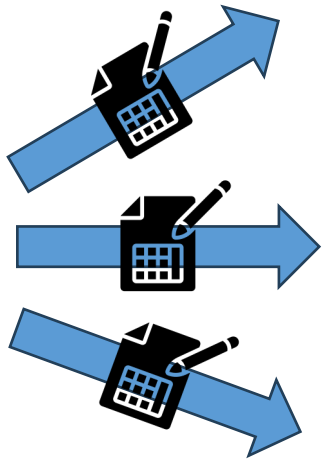


➢ Save significant data cleaning costs

Repairs of a dataset with missing values

Repair: a complete data set that replaces "null" values with data values

City	Temperature (F)	Humidity (%)
Seattle	65	80
Portland	**null**	30
San Francisco	54	90



There are many possible repairs

City	Temperature (F)	Humidity (%)
Seattle	65	80
Portland	60	30
San Francisco	54	90

Repair 1

•••

City	Temperature (F)	Humidity (%)
Seattle	65	80
Portland	80	30
San Francisco	54	90

Repair n

We propose the concept of certain models: optimal for all repairs

City	Temperature (F)	Humidity (%)
Seattle	65	80
Portland	null	30
San Francisco	54	90

Raw data

City	Temperature (F)	Humidity (%)
Seattle	65	80
Portland	60	30
San Francisco	54	90

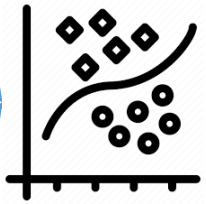
Repair 1

City	Temperature (F)	Humidity (%)
Seattle	65	80
Portland	80	30
San Francisco	54	90

Repair n

Model training

Model training

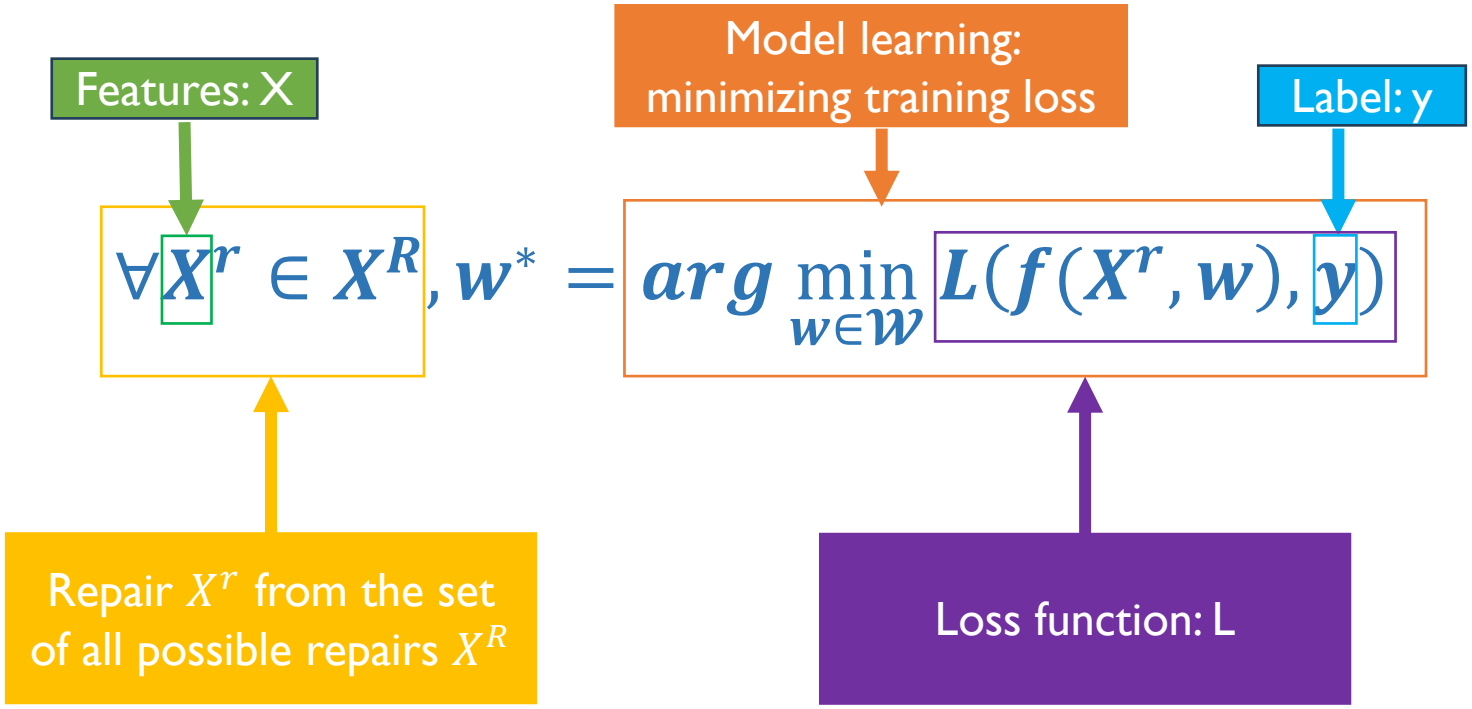


optimal model (certain model)

A certain model exists → *Imputation is unnecessary*

Certain model definition

A model w^* is a certain model if:

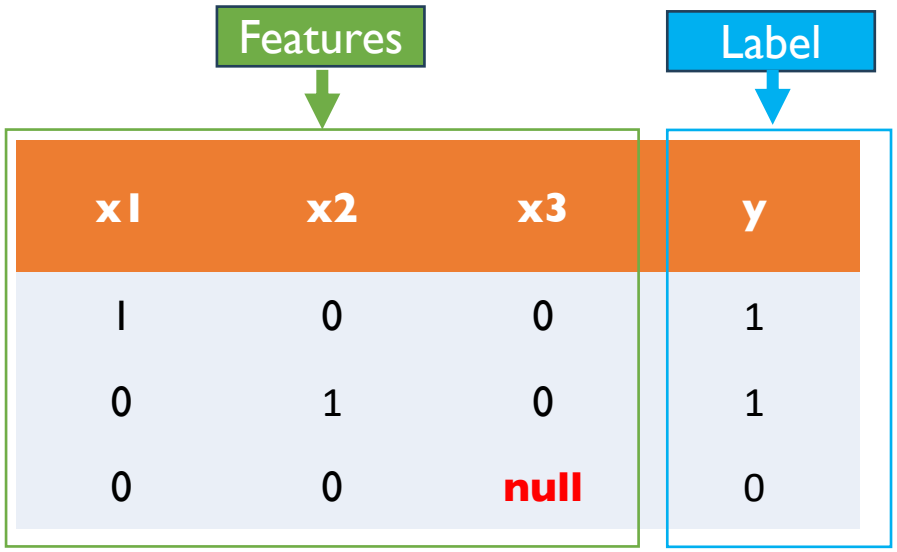


Certain model minimizes training loss for all repairs

Certain model definition

A model w^* is a certain model if:

$$\forall X^r \in X^R, w^* = arg \min_{w \in \mathcal{W}} L(f(X^r, w), y)$$

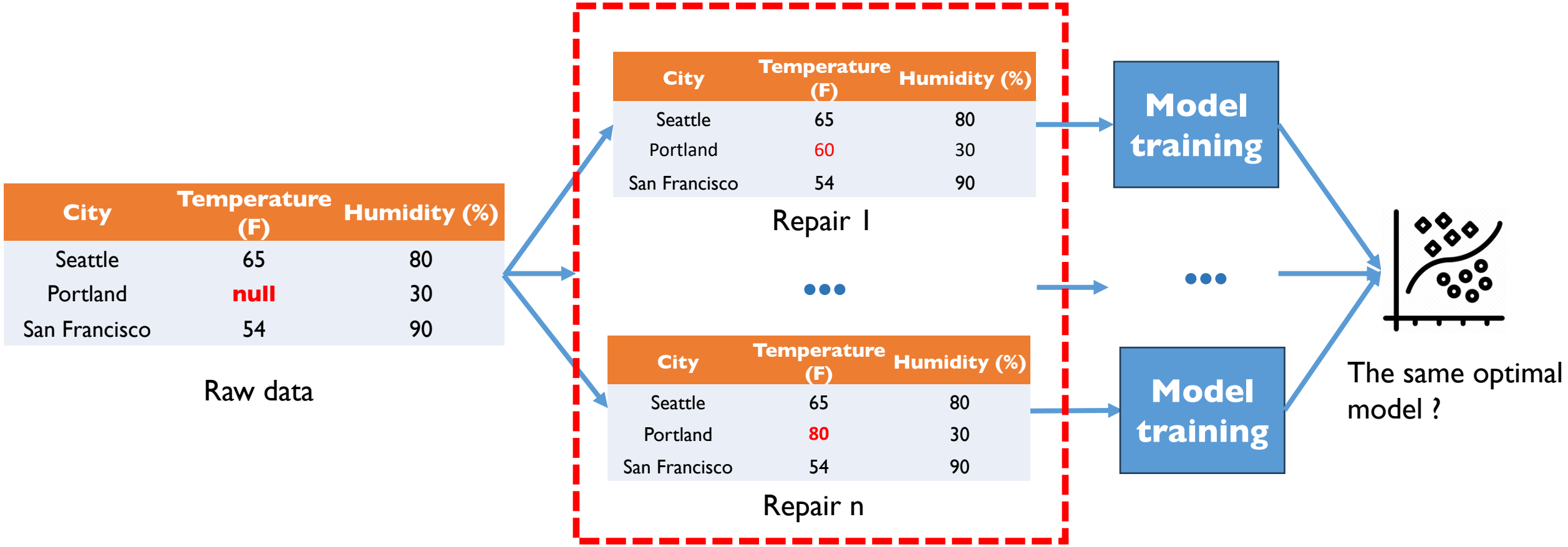


Example: linear regression

Certain model $w^* = [1, 1, 0]$ minimizes training loss in all repairs:

$\forall\ x_3^r \in x_3^R,\ 1 * x_1 + 1 * x_2 + 0 * x_3^r$ - **y = 0**

Checking existence of certain models is challenging



This is incredibly slow because there are numerous repairs

We propose efficient algorithms for checking and learning certain models

- **Linear Regression**
- **Linear Support Vector Machine**
- **Support Vector Machine with Kernels: polynomial kernel, RBF Kernel**
- **Neural Network: feed-forward neural network (approximated)**

We have proved the correctness of the algorithms (proof in the paper)

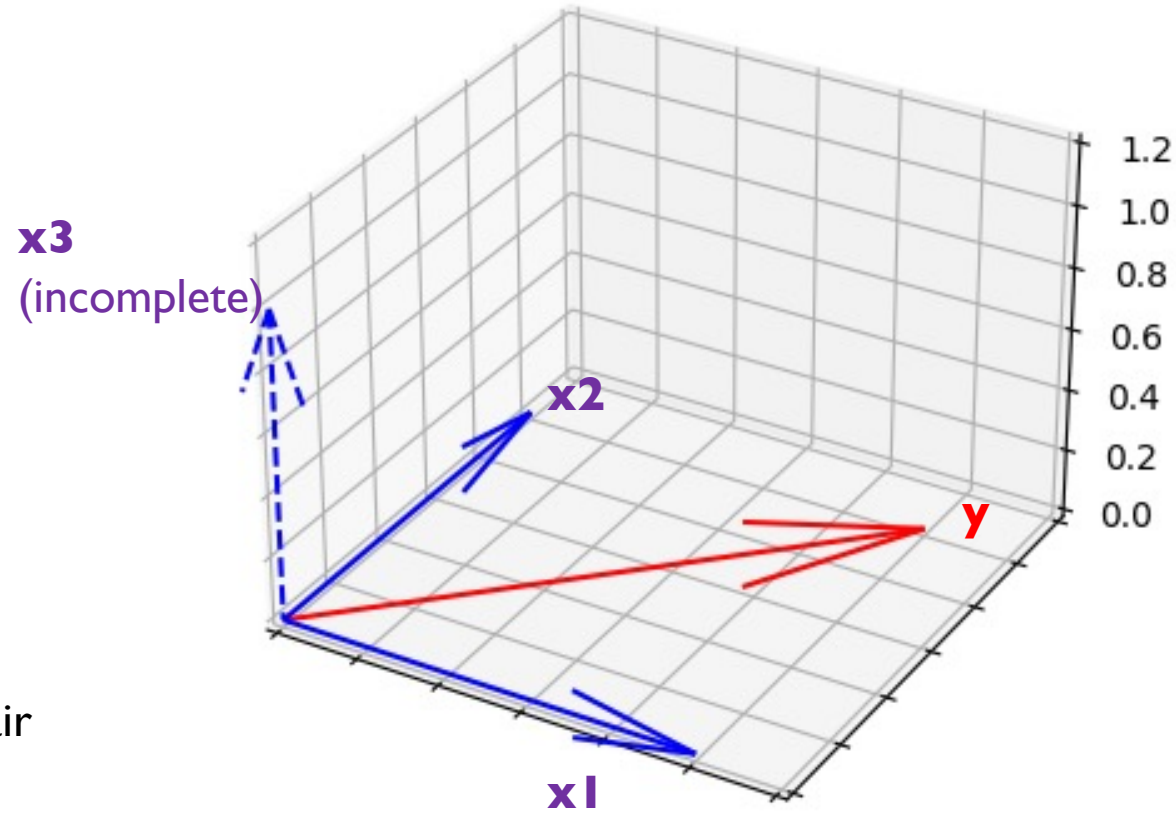
Example:
Checking certain models efficiently for linear regression

x3 ⊥ the regression residue between the label and complete features (a zero vector) since y ∈ col (x1, x2)

Features

Label

x1	x2	x3	y
1	0	0	1
0	1	0	1
0	0	**Null**	0

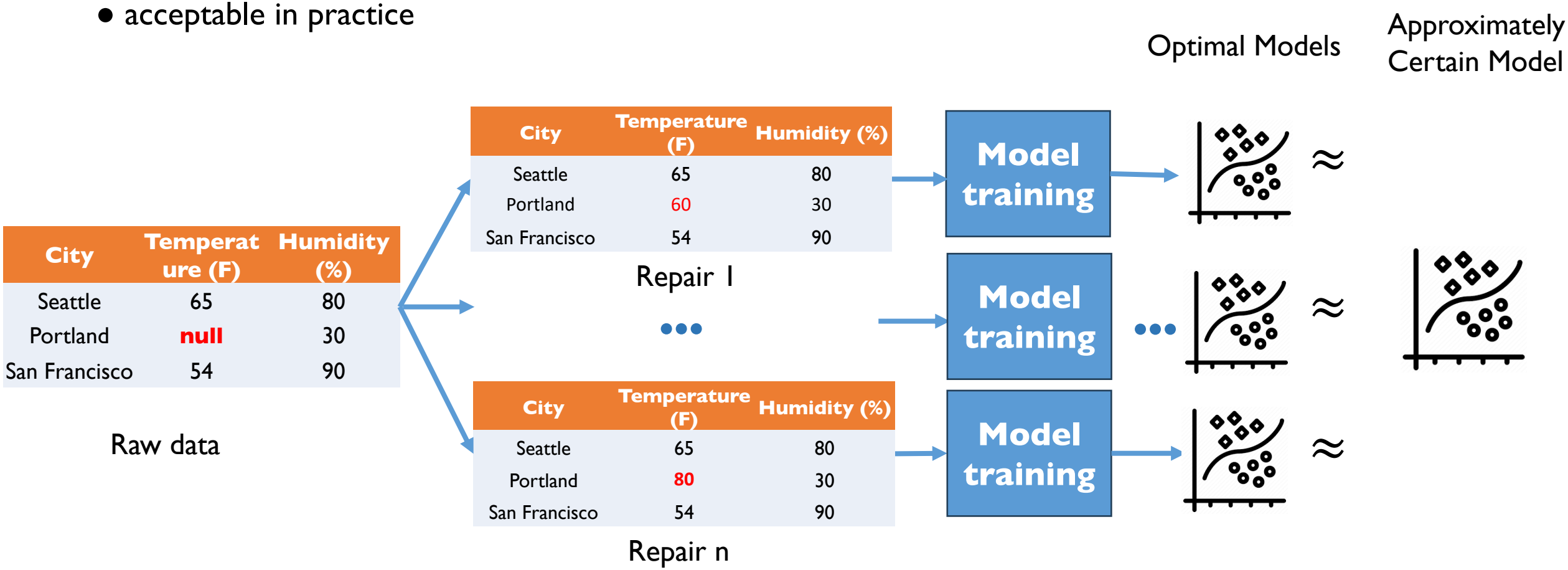


- **x3** does not contribute to minimizing loss in any repair
- Our algorithm checks if the incomplete feature vectors are orthogonal to the residue vector in all repairs

Certain model conditions are often too strict → we propose relaxed version

Approximately certain model ($w^{\approx}$):

- not optimal but sufficiently close in all repairs
- acceptable in practice



Approximately certain model definition

A model $\boldsymbol{w}^{\approx}$ is an approximately certain model (ACM) if:

$$\forall \boldsymbol{X}^{\boldsymbol{r}} \in \boldsymbol{X}^{\boldsymbol{R}}, \boldsymbol{L}(\boldsymbol{f}(\boldsymbol{X}^{\boldsymbol{r}}, \boldsymbol{w}^{\approx}), \boldsymbol{y}) - \min_{\boldsymbol{w} \in \boldsymbol{\mathcal{W}}} \boldsymbol{L}(\boldsymbol{f}(\boldsymbol{X}^{\boldsymbol{r}}, \boldsymbol{w}), \boldsymbol{y}) \leq \boldsymbol{\varepsilon}$$

- Repairs
- Loss with ACM
- Minimal training loss
- User-defined threshold

We propose efficient algorithms for checking and learning approximately certain models

We have proposed algorithms for all ML models with convex loss function:

For example:

- **Linear Regression**
- **Linear Support Vector Machine**
- **Logistic Regression**

We have proved the correctness of the algorithms (proof in the paper)

Experimental setup

- **Real world datasets containing missing values**

Data Set	Task	Features	Training Examples	Missing Factor
Breast Cancer	Classification	10	559	1.97%
Intel-Sensor	Classification	11	1850945	4.05%
NFL	Regression	34	34302	9.04%
Water-Potability	Classification	9	2620	39.00%
Online Education	Classification	36	7026	35.48%
COVID	Regression	188	60229	53.67%
Air-Quality	Regression	12	7192	90.99%
Communities	Regression	1954	1595	93.67%

Missing Factor: # of examples with at least one missing value / total # of examples

Methods in our experiments

- **No Imputation (NI): Delete all incomplete examples**

- **Imputations:**
  1) Mean Imputation (MI): Impute by a simple mean of the feature values
  2) KNN-Imputer (KI): Impute through a KNN classifier
  3) MIWAE (DI): A deep-learning-based imputation framework featured by high imputation quality

- **On-demand Cleaning:**
  ActiveClean (AC): a data cleaning framework also aims to avoid unnecessary data cleaning

- **Our Approaches:**
  1) Certain Model (CM)
  2) Approximately Certain Model (ACM)

Comparing in terms of imputation costs (# of examples imputed), and learning (+ imputation) time (sec)

When certain model exists

Saving cleaning costs

Data Set	Imputations	On-demand Cleaning	Deletion	Our Method
	MI/KI/DI	AC	NI	CM
NFL	3101	12.0	0	0
COVID	32325	33.6	0	0

Number of Examples Cleaned

Computational overhead is small

Data Set	Imputation	On-demand Cleaning	Deletion	Our Method
	DI	AC	NI	CM
NFL	394.18	49.91	0.13	7.11
COVID	1944.10	100.59	0.28	438.79

Learning (+imputation) Time (sec)

Guarantee optimal model

Data Set	Imputation	On-demand Cleaning	Deletion	Our Method
	DI	AC	NI	CM
NFL	0.00	0.02	0.00	0.00
COVID	0.00	2.07	0.00	0.00

Regression MSE

When CM does not exist, but ACM exists

Saving cleaning costs

Data Set	Imputations	On-demand Cleaning	Deletion	Our Method	
	MI/KI/DI	AC	NI	CM	ACM
Communities	1494	319.6	0	0	0

Number of Examples Cleaned

Computational overhead is small

Data Set	Imputation	On-demand Cleaning	Deletion	Our Method	
	DI	AC	NI	CM	ACM
Communities	4088.46	2.10	0.08	1.45	3.74

Learning (+imputation) Time (sec)

Guaranteeing approximately optimal model

Data Set	Imputation	On-demand Cleaning	Deletion	Our Method
	DI	AC	NI	ACM
Communities	0.35	0.06	2.30	0.03

Regression MSE

When neither exists

Computational overhead of CM and ACM is small

Data Set	Imputation	On-demand Cleaning	Deletion	Our Method	
	DI	AC	NI	CM	ACM
Air Quality	111.08	0.87	0.01	0.01	4.62

Learning (+imputation) Time (sec)

High variability among cleaning methods

Data Set	Imputation	On-demand Cleaning	Deletion
	DI	AC	NI
Air Quality	2.11	28.22	3.47

Regression MSE

Conclusion

- We propose CM/ACM to learn accurate model without cleaning
- We propose efficient algorithms for learning CM & ACM for a wide variety of ML models
- Our algorithms learn accurate models efficiently over real-world datasets

Ongoing work

- When CM does not exist, our algorithms propose a set of tuples to repair to get CM
- We work on extending algorithms to find minimal sets of repairs

Scan this code to connect with the lead-author Cheng Zhen

Contact:

zhenc@oregonstate.edu

THANK YOU!

Oregon State
University