

Advanced Machine Learning For Design

Lecture 6: Train, Evaluate and Integrate Machine
Learning Models (part 1)

Module 3

Evangelos Niforatos

25/10/2023

aml4d-ide@tudelft.nl
<https://aml4design.github.io/>

Admin

Few remarks

- Module #1 Group Assignment has been graded
- Average Peer Assessment #1 scores have been computed
- Module #2 Group Assignment: Deadline extended: Oct. 25, 23:59
- Complete Peer Assessment #2 by Friday, Oct. 27, 2022, 23:59
- Prepare for Tutorial #3 on “Training, Evaluating and Integrating Machine Learning Models”
 - Tutorial #3 is scheduled for Friday, Oct 27
 - This tutorial is more advanced than the previous ones
 - More preparation is required

How to fill in a peer assessment form

	A	B	C	D	E	F	G	H	I	J	K
1	Group Number: <Insert Your Group Nr.>										
2	Criterion	Assessemment (From Rubric)									
3											
4	Helping										
5	Listening										
6	Participating										
7	Persuading										
8	Questioning										
9	Respecting										
10	Sharing										
11											
12											
13	Private Comments										
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											
24											

RUBRIC | <<Your Name>> | <<Name of Member 2>> | <<Name of Member 3>> | <<Name of Member 4>> | <<Name of Member 5>> | + |



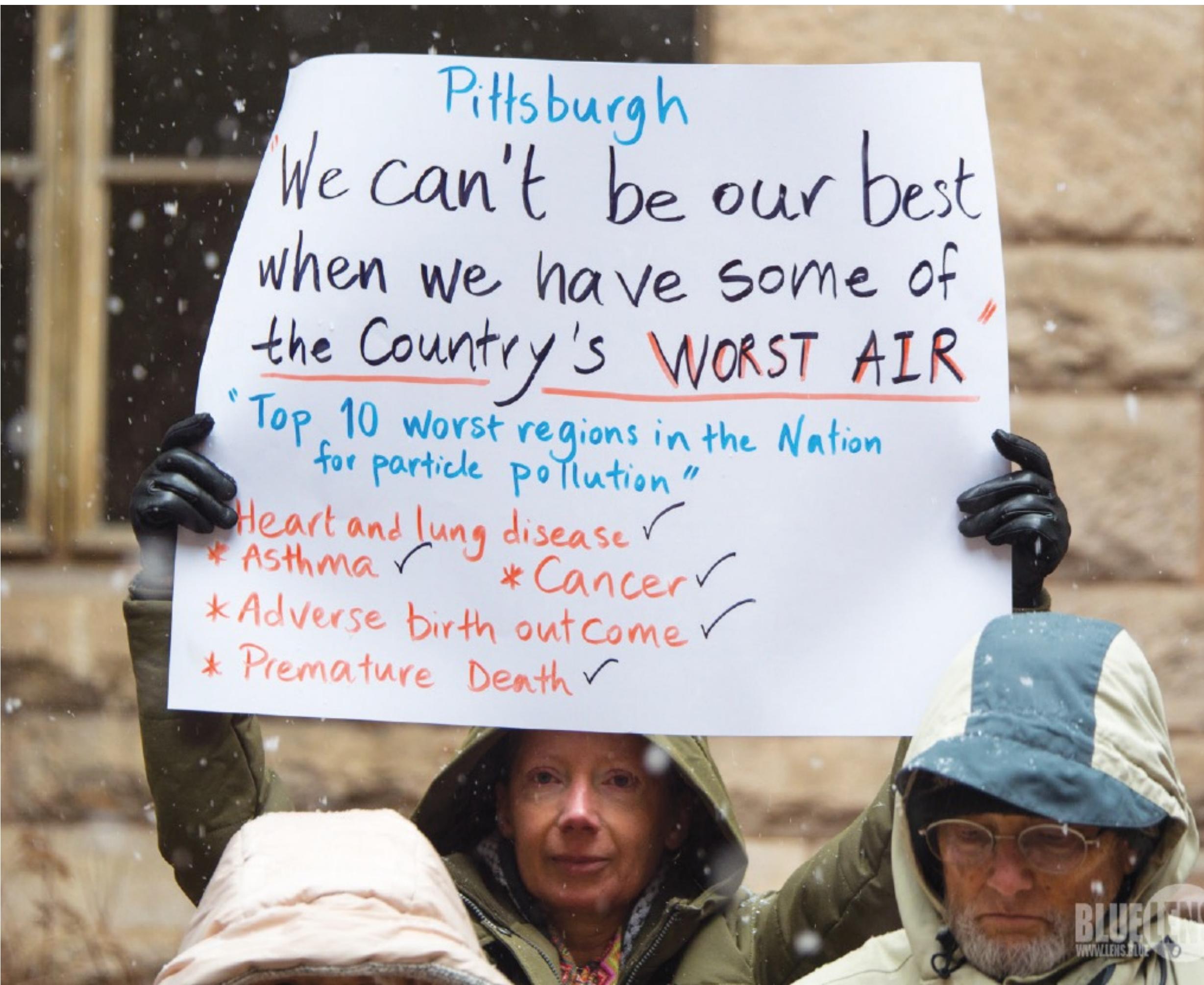
PeerAssessmentForm-GroupAssignment-2_YOUR_NAME_YOUR_GROUP
XLSX



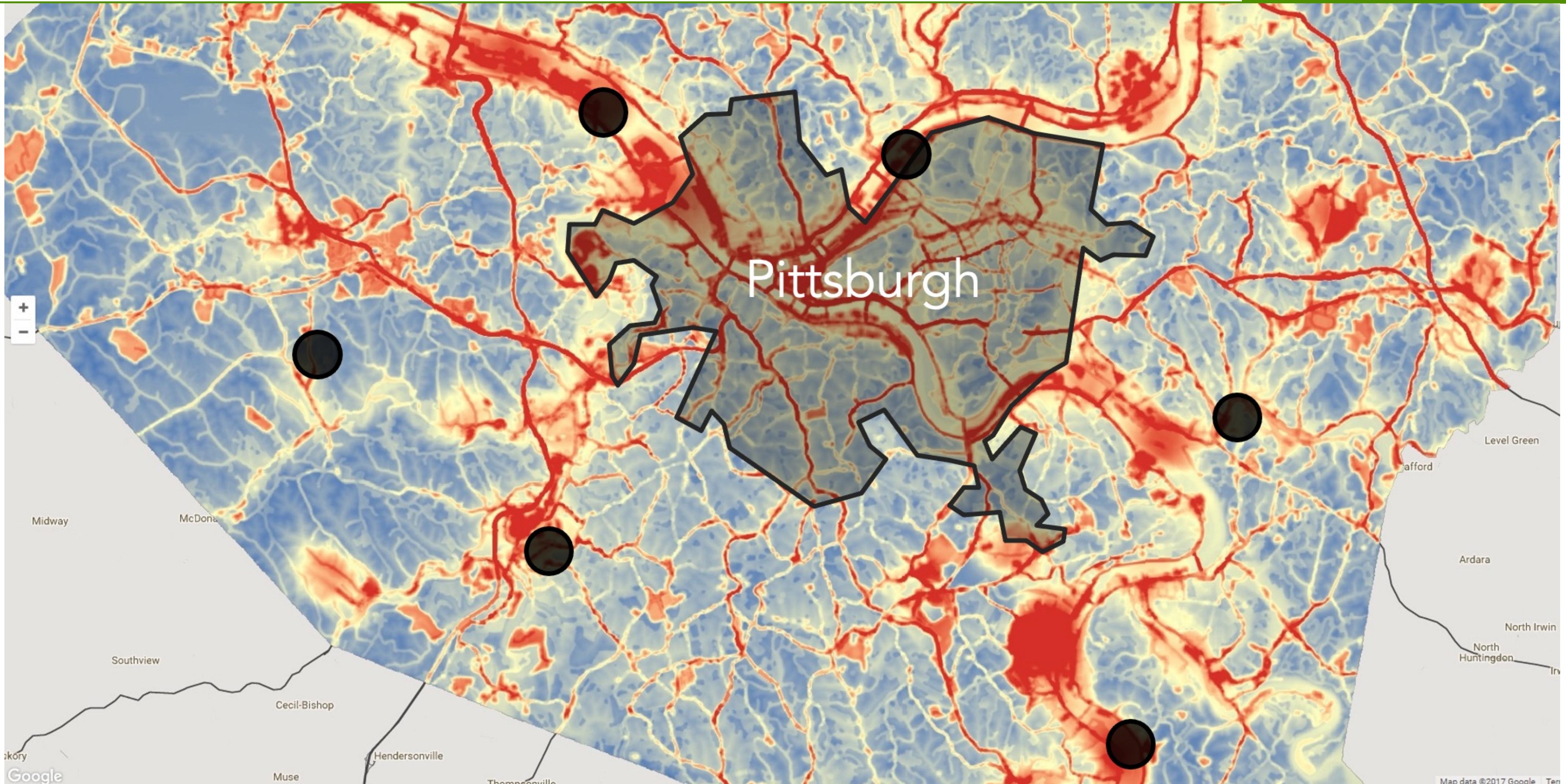
Let's go to Pittsburgh, PA, USA

Tutorial 3 scheduled on Friday 27, 2023

According to the American Lung Association, Pittsburgh is one of the ten most polluted cities (measured by particulate matter) in the United States. Local residents have been fighting against air pollution for decades.

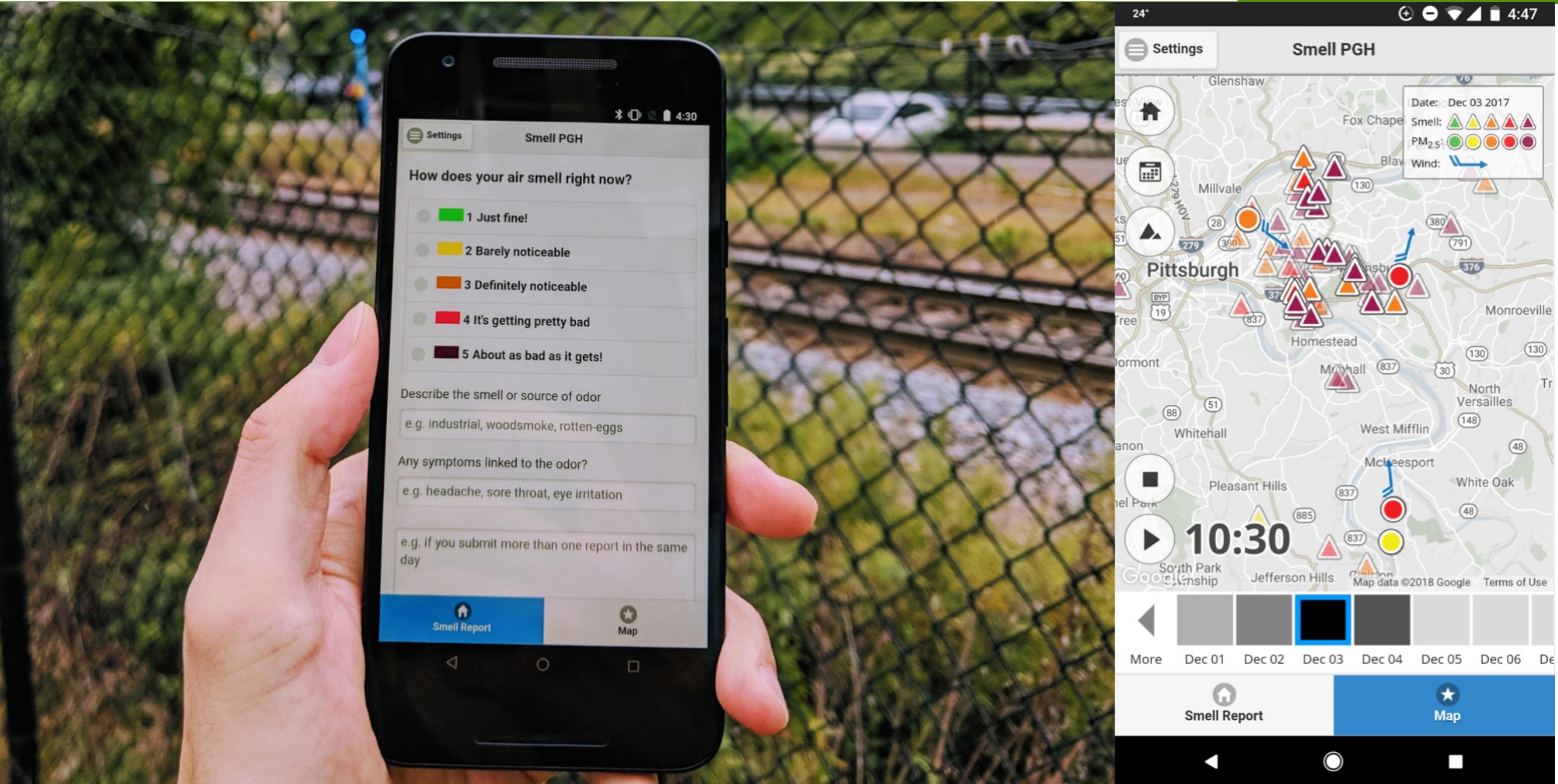


Local people have identified smell as an indicator of air pollution. But, how can we effectively **collect the smell experiences on a city-wide scale** with more than 300,000 residents over many years?



Link to the Pittsburgh pollution map – <https://breatheproject.org/pollution-map/>

Smell Pittsburgh is a mobile application that enables local communities to **contribute odor reports** in real-time (with accurate time and location information) and **visualize air pollution collaboratively**.



Link to the Smell Pittsburgh application – <https://smellpgh.org>

How does your air smell right now?

- 1 Just fine!
- 2 Barely noticeable
- 3 Definitely noticeable
- 4 It's getting pretty bad
- 5 About as bad as it gets!

Describe the smell or source of odor

Any symptoms linked to the odor?

Add a personal note to the Health Department



Smell Report



Map

Smell Pittsburgh **predicts upcoming smell events** (based on the existing data at a certain time point) and **sends push notifications** to inform users while **encouraging engagement** in submitting odor data.



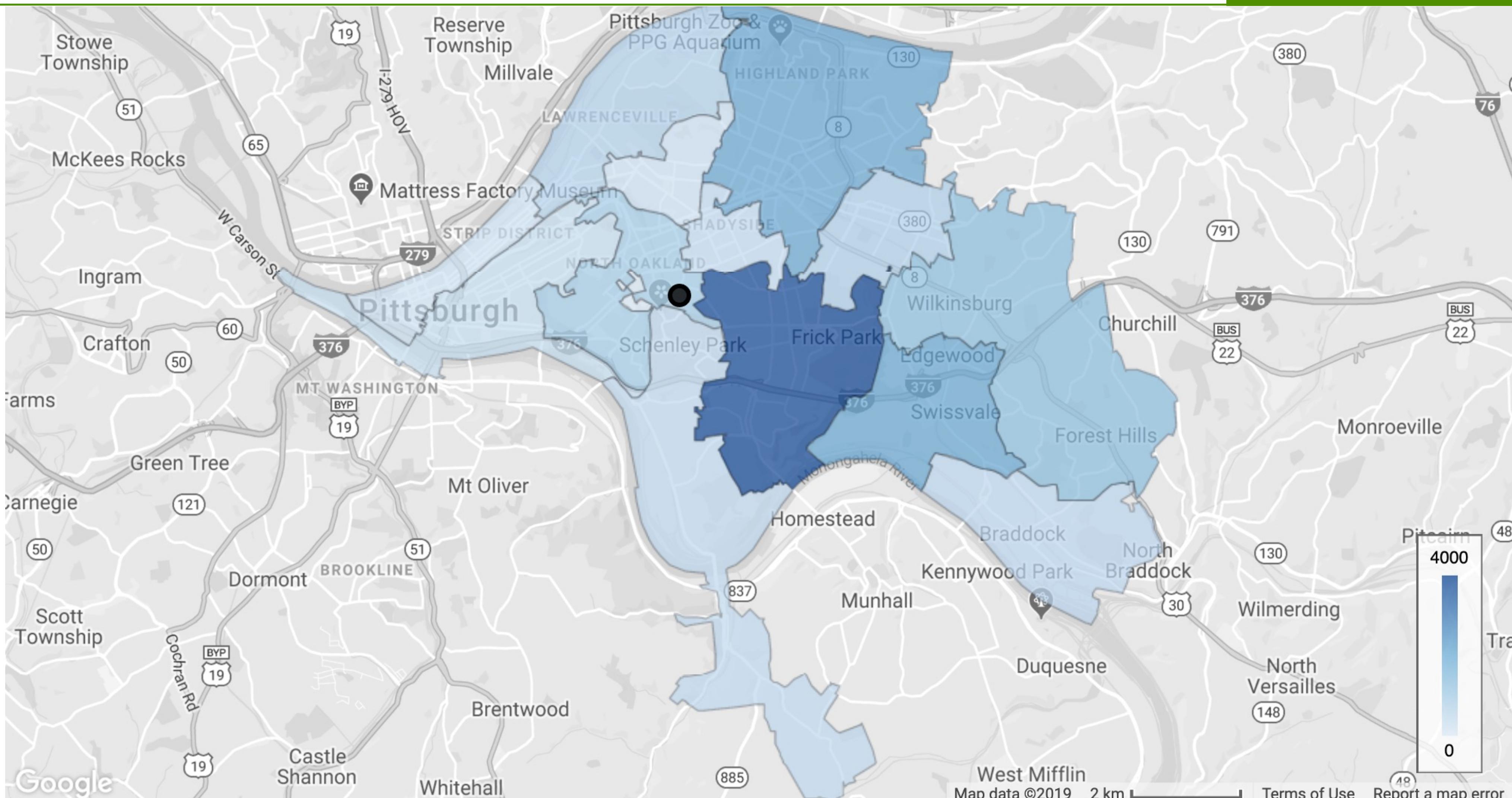
SMELL PGH

Smell Event Alert

Local weather and pollution data indicates there may be a Pittsburgh smell event in the next few hours.

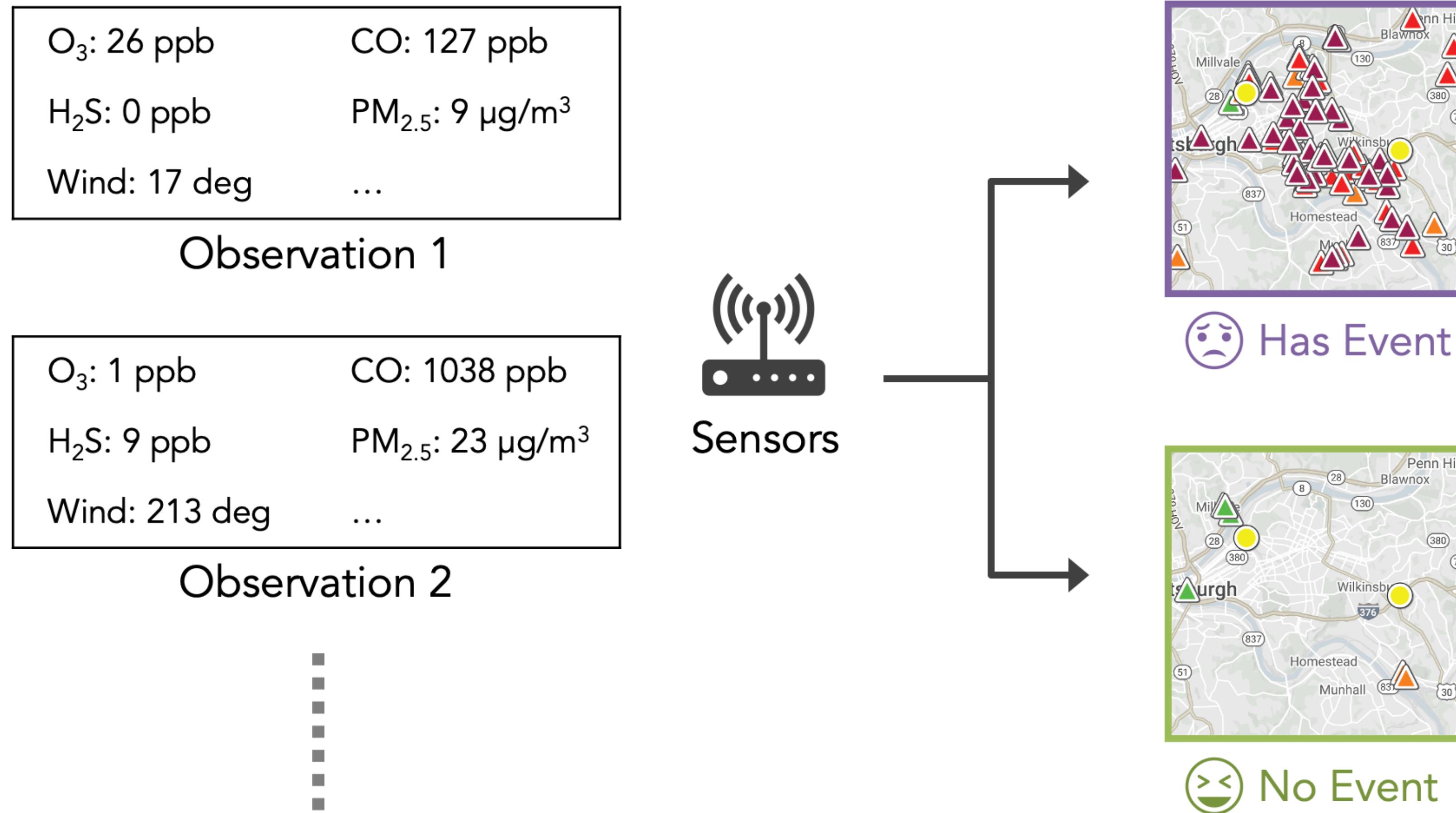
Keep a nose out and report smells you notice!

A geographic region in Pittsburgh is manually selected when predicting the smell events. The black dot in the figure represents the location of Carnegie Mellon University.



Number of smell reports aggregated by zip codes in the dataset.

To predict the presence of bad odor within the next few hours, we need to **estimate a function that can map sensor measurements to smell events** as accurately as possible.



One can technically use if-else rules to predict smell events. But such an approach can be laborious. **Can we do better than manually specifying these if-else rules** while minimizing human efforts?

O_3 : 26 ppb	CO: 127 ppb
H_2S : 0 ppb	$PM_{2.5}$: 9 $\mu\text{g}/\text{m}^3$
Wind: 17 deg	...

Observation 1

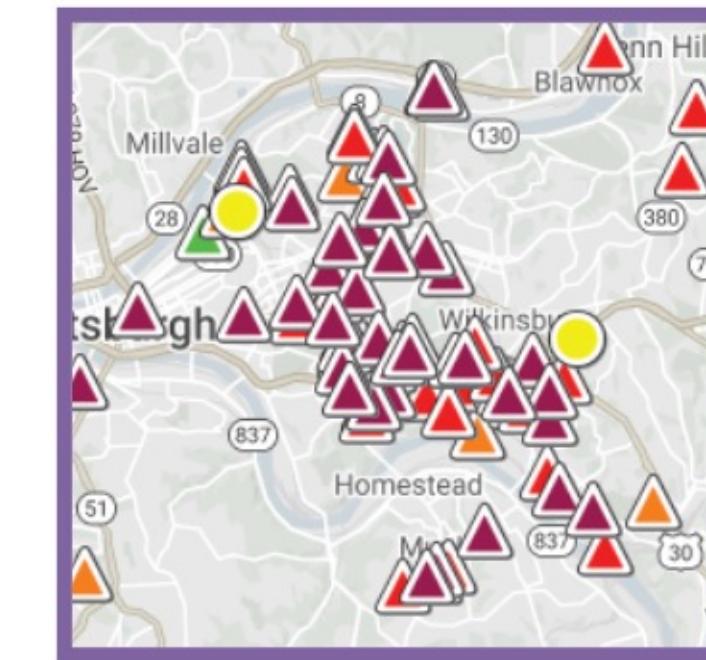
O_3 : 1 ppb	CO: 1038 ppb
H_2S : 9 ppb	$PM_{2.5}$: 23 $\mu\text{g}/\text{m}^3$
Wind: 213 deg	...

Observation 2

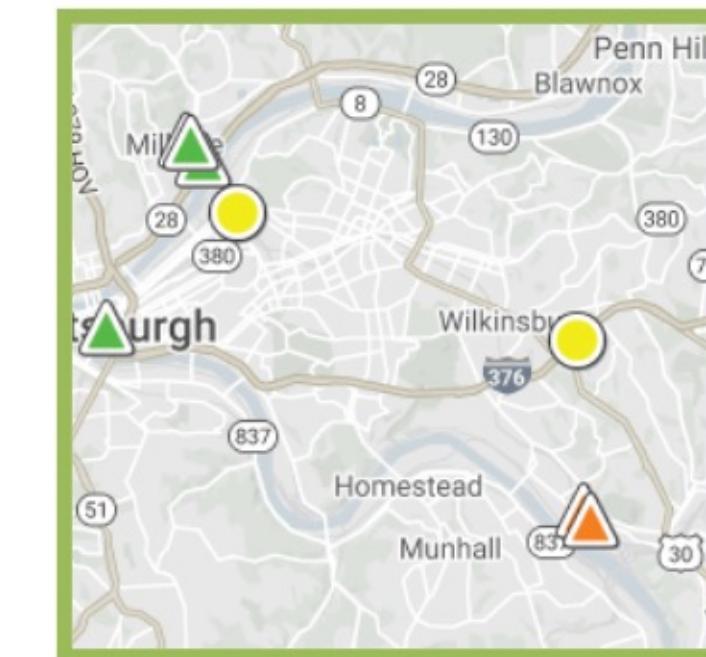
⋮

if $H_2S > ?$
and $CO > ?$
and $PM_{2.5} > ?$
and ...
then has event

else no event



:(Has Event



>: No Event

It turns out that we can use the Smell Pittsburgh dataset to estimate a function (i.e., train a machine learning model) that can predict smell events from sensor measurements.

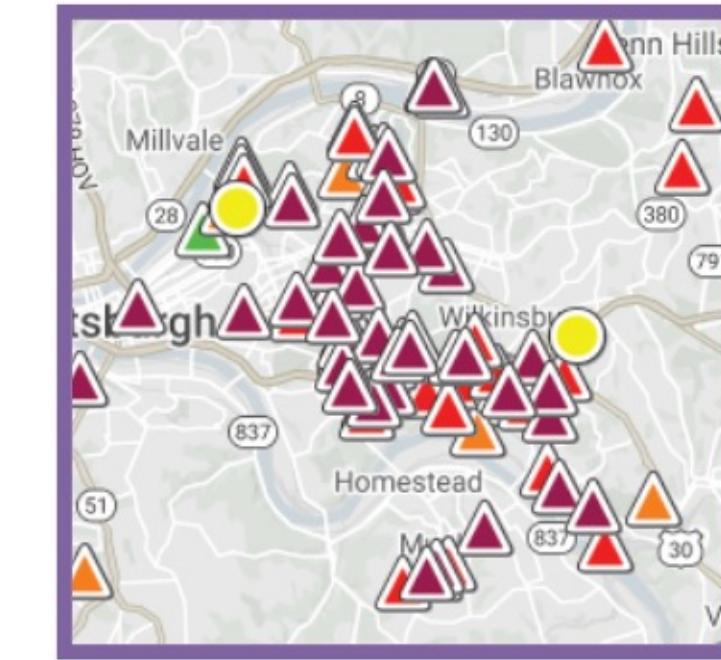
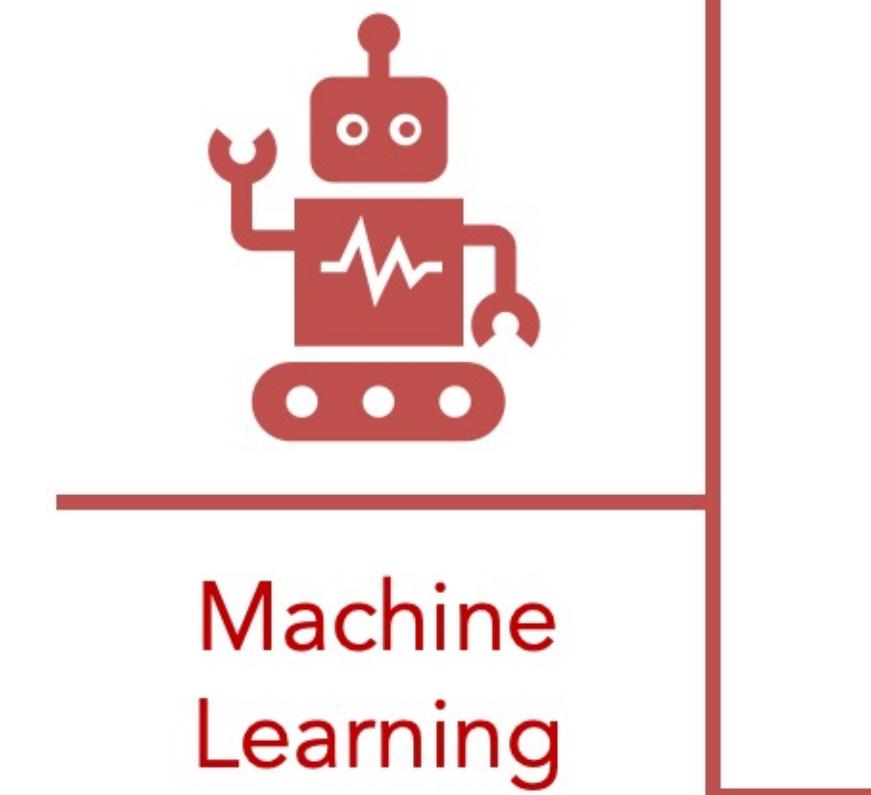
O_3 : 26 ppb	CO: 127 ppb
H_2S : 0 ppb	$PM_{2.5}$: 9 $\mu\text{g}/\text{m}^3$
Wind: 17 deg	...

Observation 1

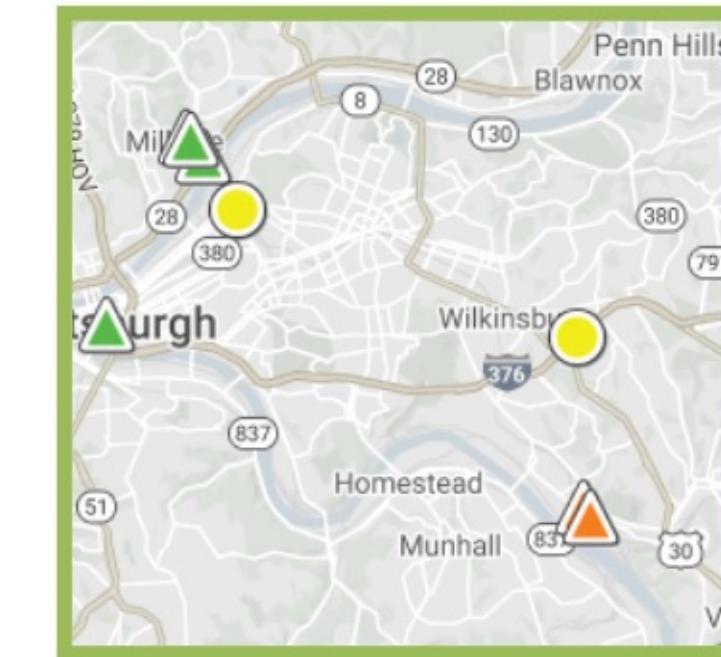
O_3 : 1 ppb	CO: 1038 ppb
H_2S : 9 ppb	$PM_{2.5}$: 23 $\mu\text{g}/\text{m}^3$
Wind: 213 deg	...

Observation 2

⋮



:(Has Event



:> No Event

Researchers collected the **Smell Pittsburgh dataset**, including all the smell reports and sensor measurements (from air quality and weather monitoring stations) from October 31, 2016 to September 30, 2018.

- Samples of Citizen-Contributed Smell Reports

EpochTime	feelings_symptoms	smell_description	smell_value	zipcode
...
1478353854	Headache, sinus, seeping into house even though it is as shut and sealed as possible. Air purifiers are unable to handle it thoroughly.	Industrial, acrid, strong	4	15206
1478354971	...	Industrial	4	15218
...

- Samples of Air Quality Sensor Measurements

EpochTime	3.feed_28.H2S_PPM	3.feed_28.SO2_PPM	3.feed_28.SIGTHETA_DEG	3.feed_28.SONICWD_DEG	3.feed_28.SONICWS MPH
...
1478046600	0,019	0,020	14,0	215,0	3,2
1478050200	0,130	0,033	13,4	199,0	3,4
...



imgflip.com

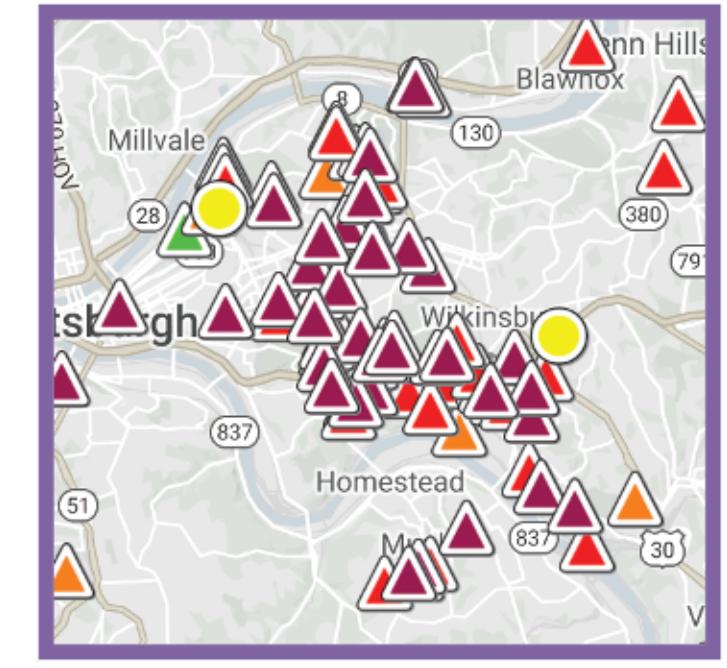
We need to **quantitatively define a smell event** (i.e., the presence of bad odor): whether the sum of smell values within a specific time range is larger than a particular threshold.

- Samples of Citizen-Contributed Smell Reports

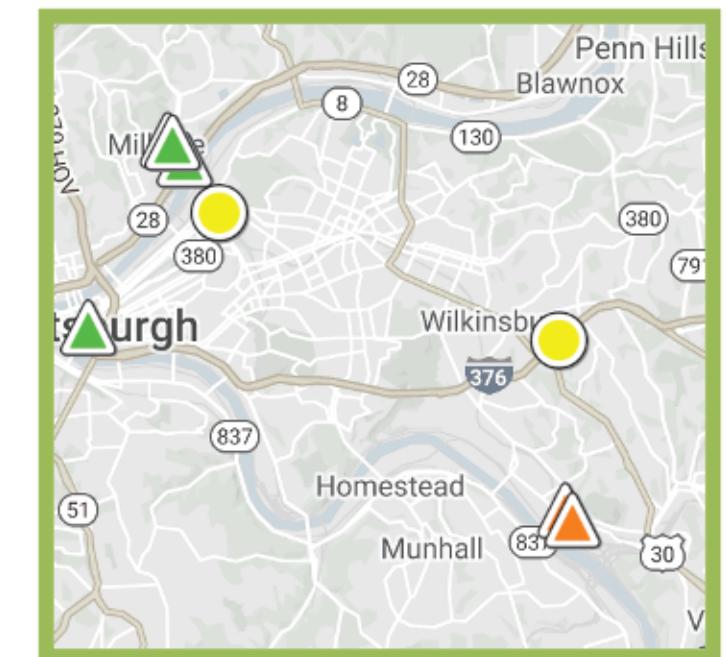
EpochTime	smell_value	zipcode
...
1478353854	4	15206
1478354971	4	15218
1478359473	4	15218
1478371179	3	15207
1478393585	3	15217
1478399011	4	15217
1478432399	4	15218
1478432502	2	15206
1478434105	4	15217
1478435133	4	15206
1478435313	4	15206
1478435748	3	15206
1478435801	5	15218
...

if the sum of smell values
within H hours > V
(need to define H and V)

then there is a smell event
else no event

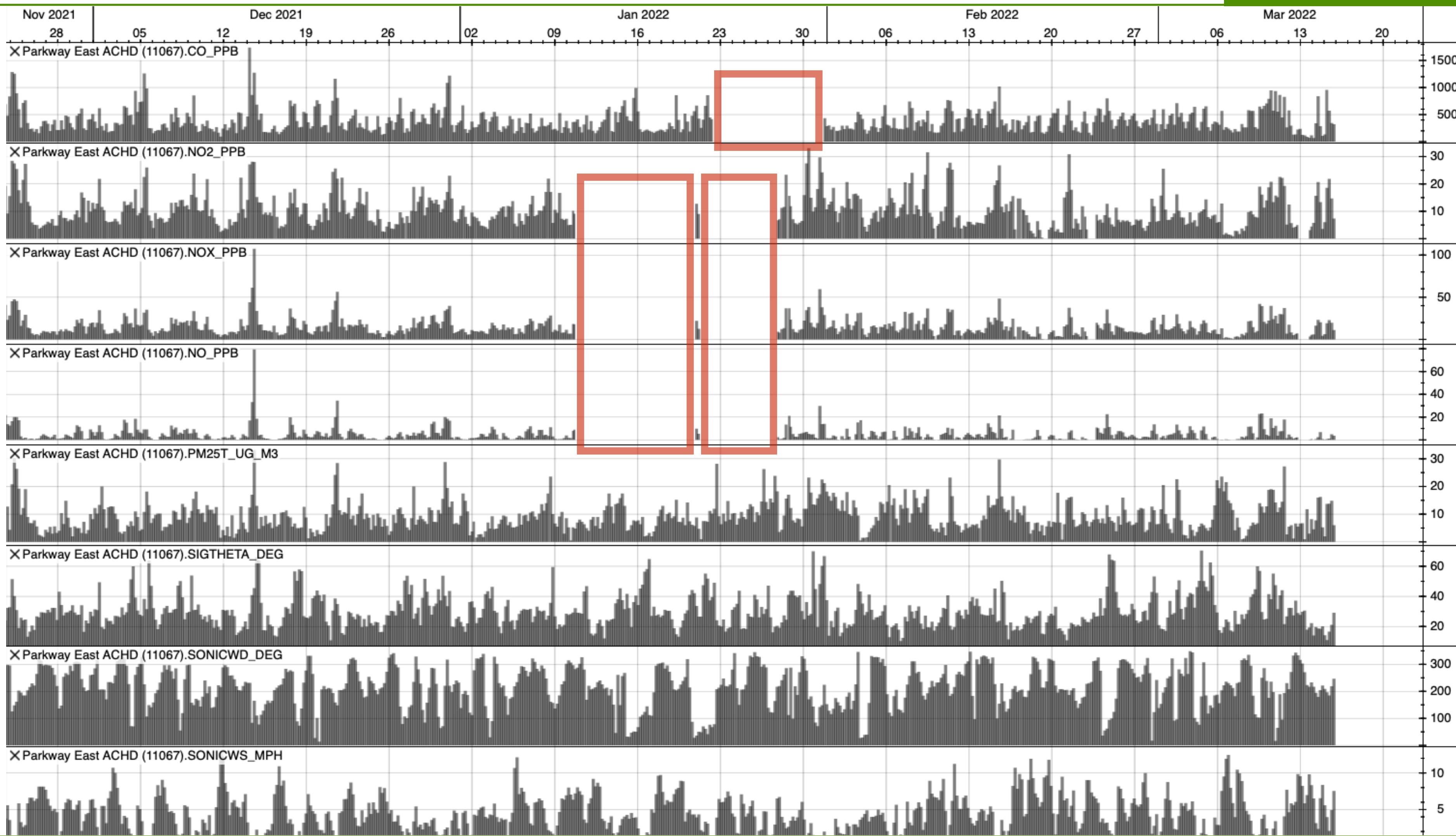


Has Event

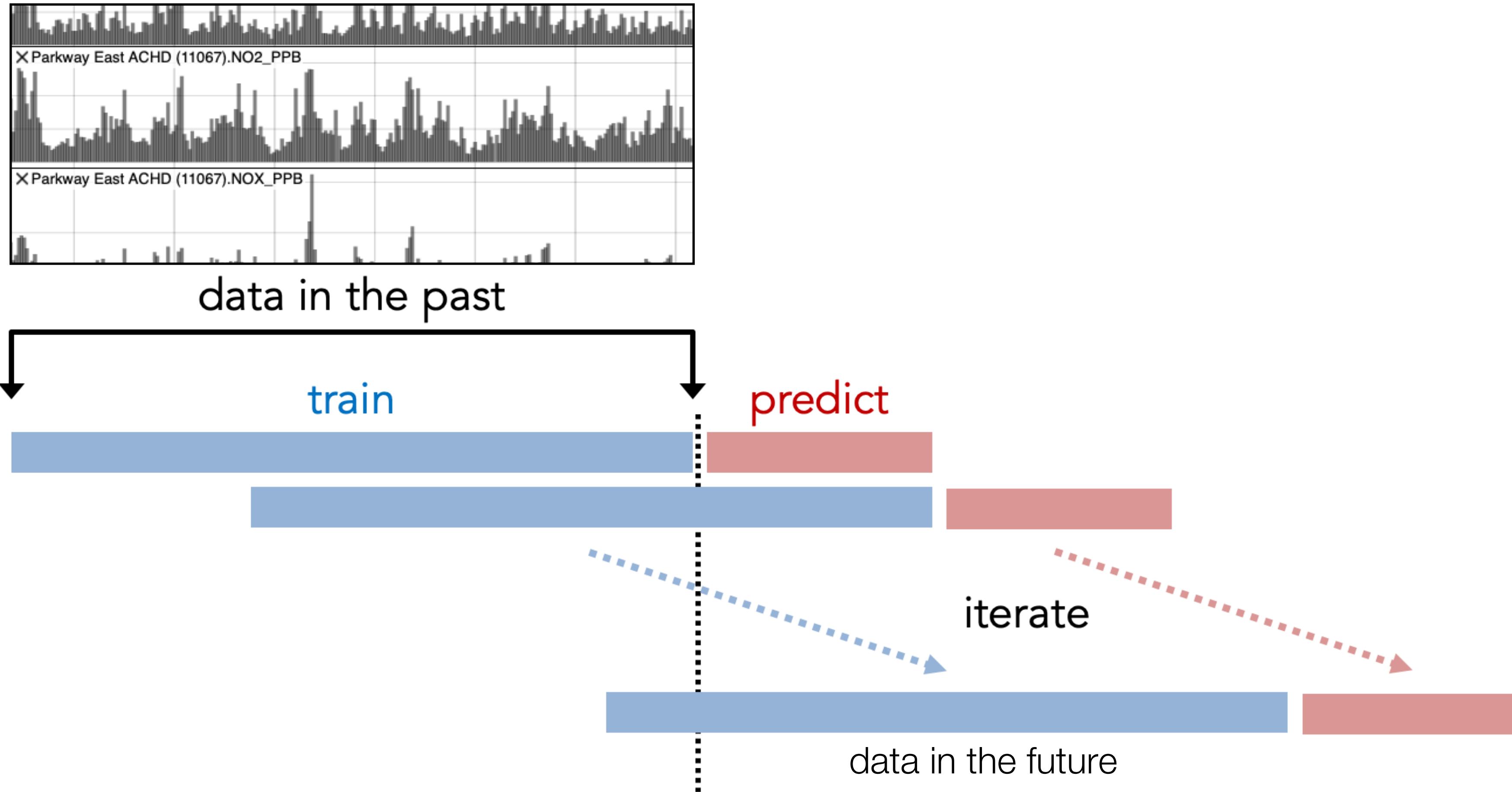


No Event

We need to **treat missing data**. The sensor measurements can be missing during some time periods since some air quality or weather monitoring stations may be down for maintenance.

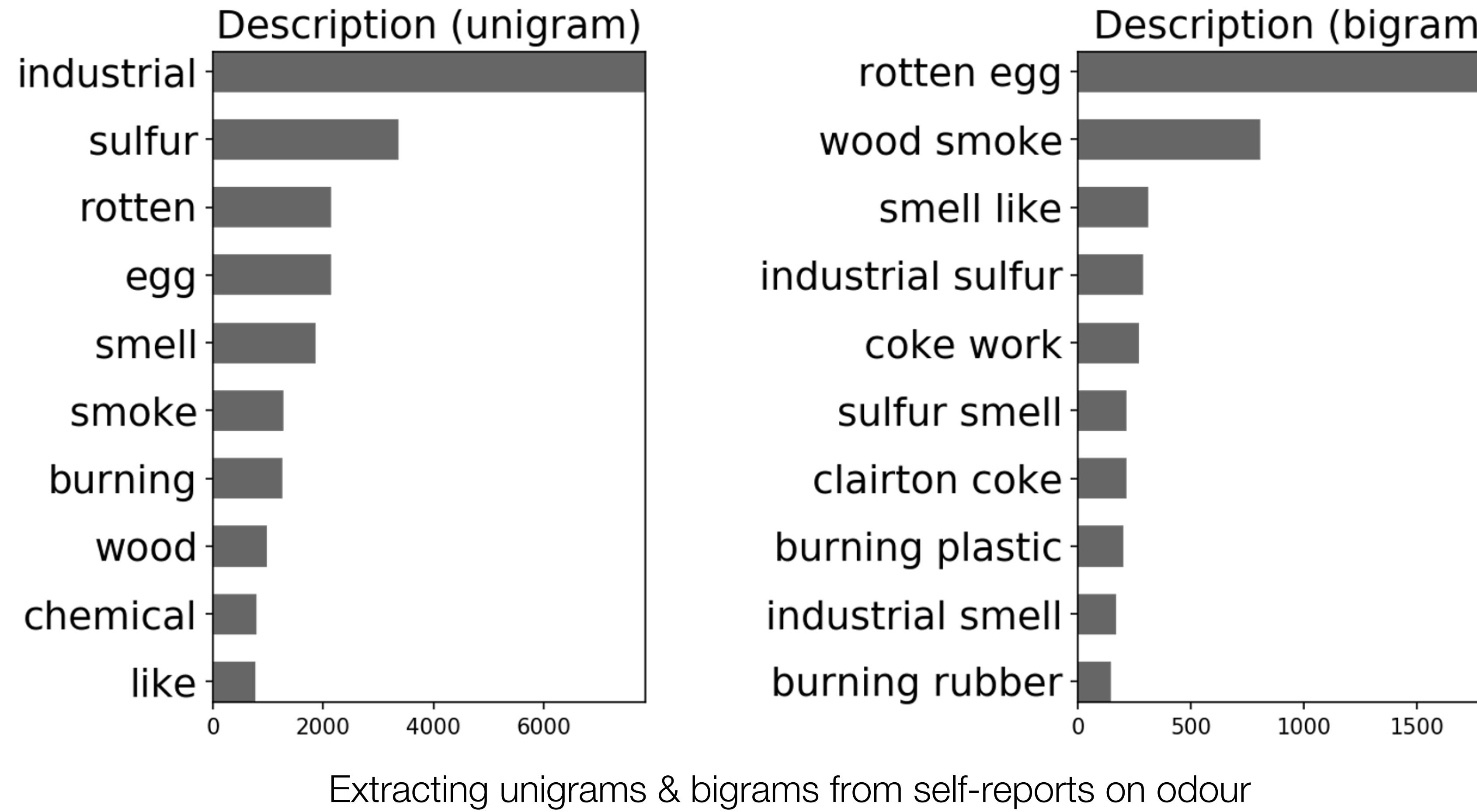


The dataset contains **time-series data**, which means each data point has a timestamp, and we can only use data in the past (i.e., data that exists for a specific time point) to train the model to predict the future.

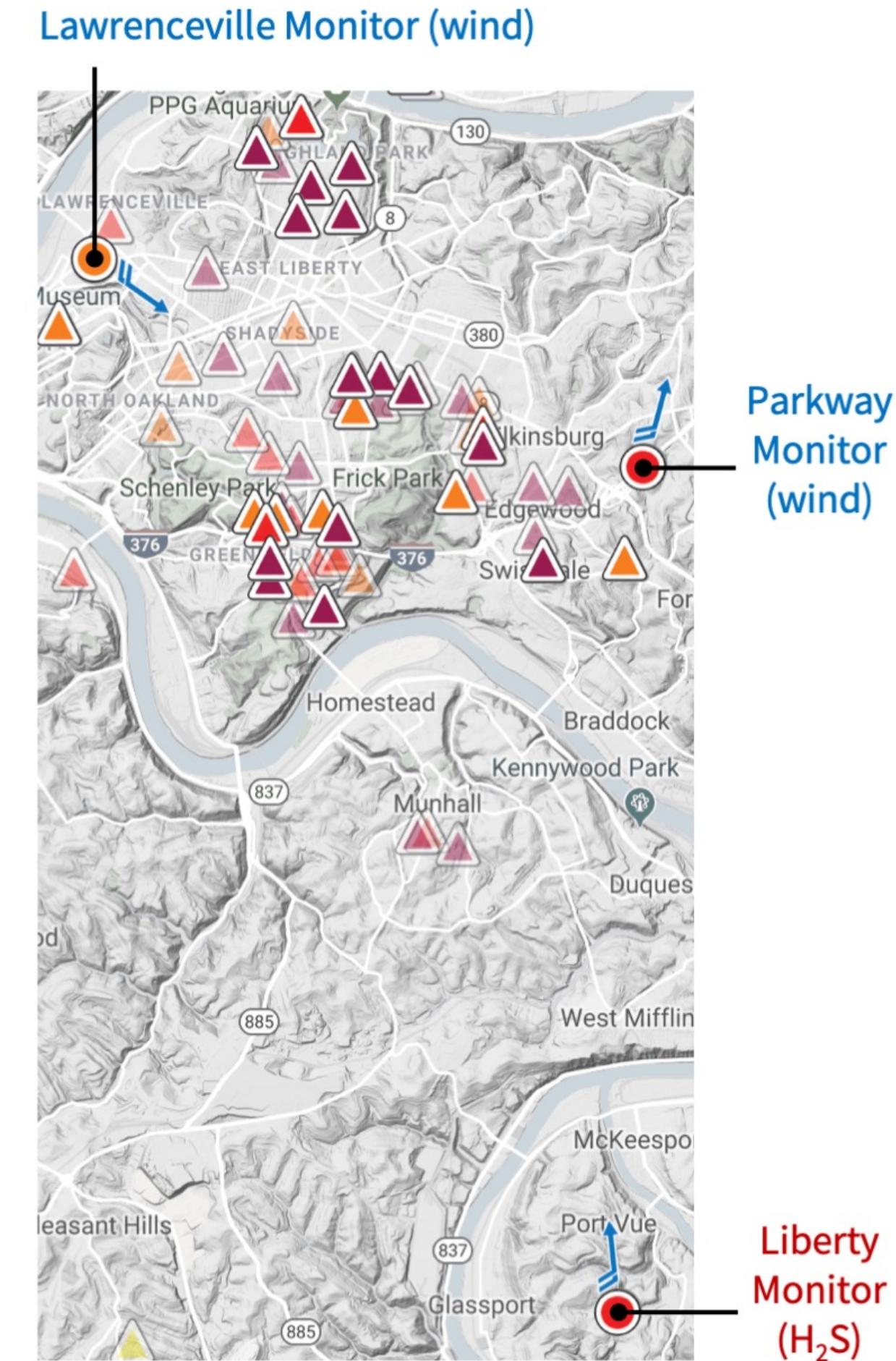
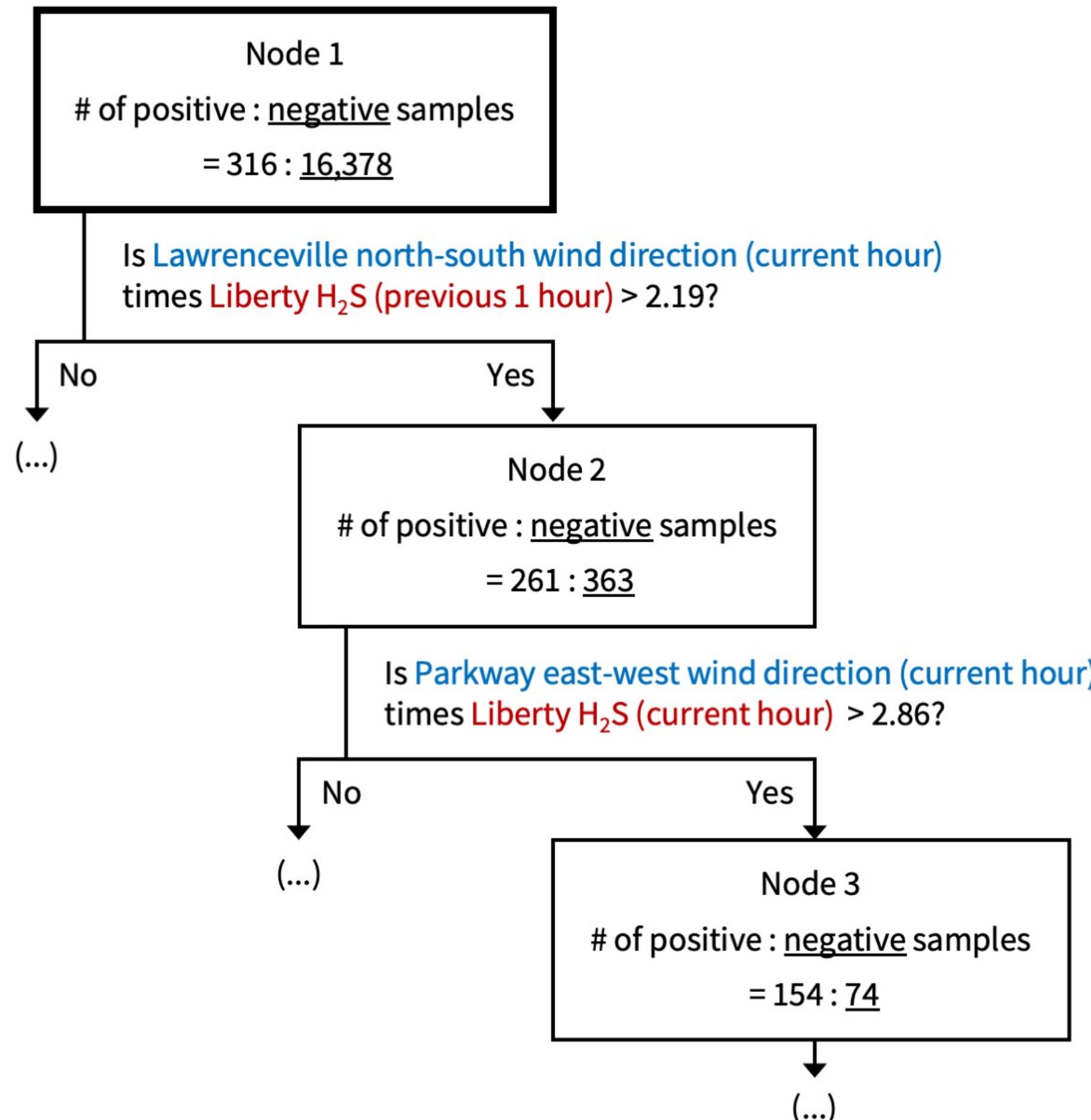


October 25, 2023, 14:30

How do we know **which variables from which monitoring stations** are effective in predicting the presence of bad odor? We can explore the data to get insights or rely on local knowledge of pollution sources.

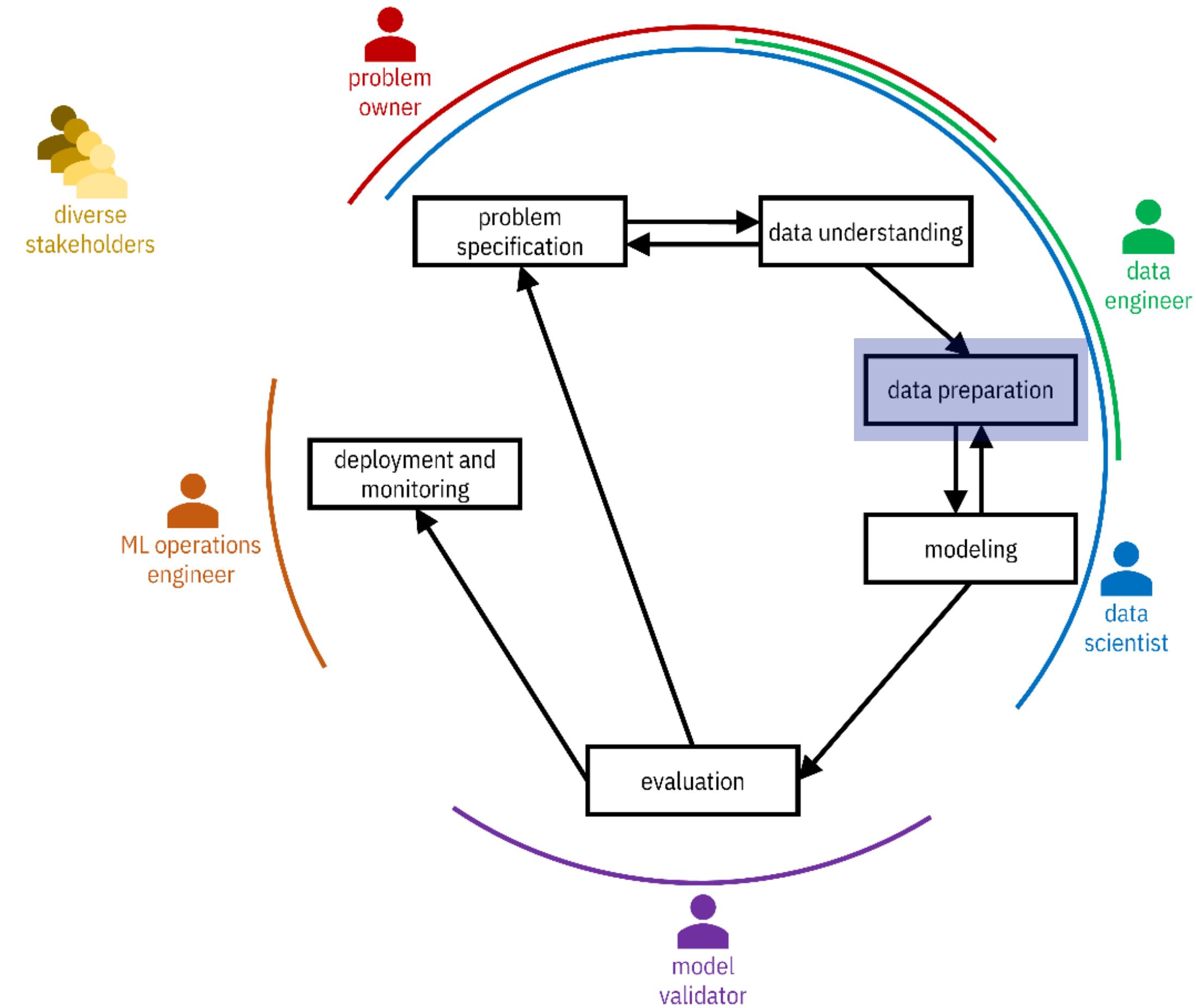


We also need to **extract and decide on the features** that we want to use when training the machine learning model. Such features can help us identify air pollution patterns in the Pittsburgh region.



Data Preparation

Cross-Industry Standard Process for Data Mining (CRISP-DM) Methodology



Types of Feature / Label Values

Categorical

- Named data
- Can take numerical values, but no mathematical meaning

Nominal

- No order
- No direction

- Marital status
- Gender
- Ethnicity

Ordinal

- Order
- Direction

- Letter grades (A, B, C, D)
- Socio-economic status (poor, rich)
- Ratings (dislike, neutral, like)

Numerical

- Measurements
- Take numerical values
 - Discrete or continuous

Interval

- Difference between measurements
- No true zero or fixed beginning

- Temperature (C or F)
- IQ
- Time, Dates

Ratio

- Difference between measurements
- True zero exists

- Temperature (K)
- Age
- Height
- Weight

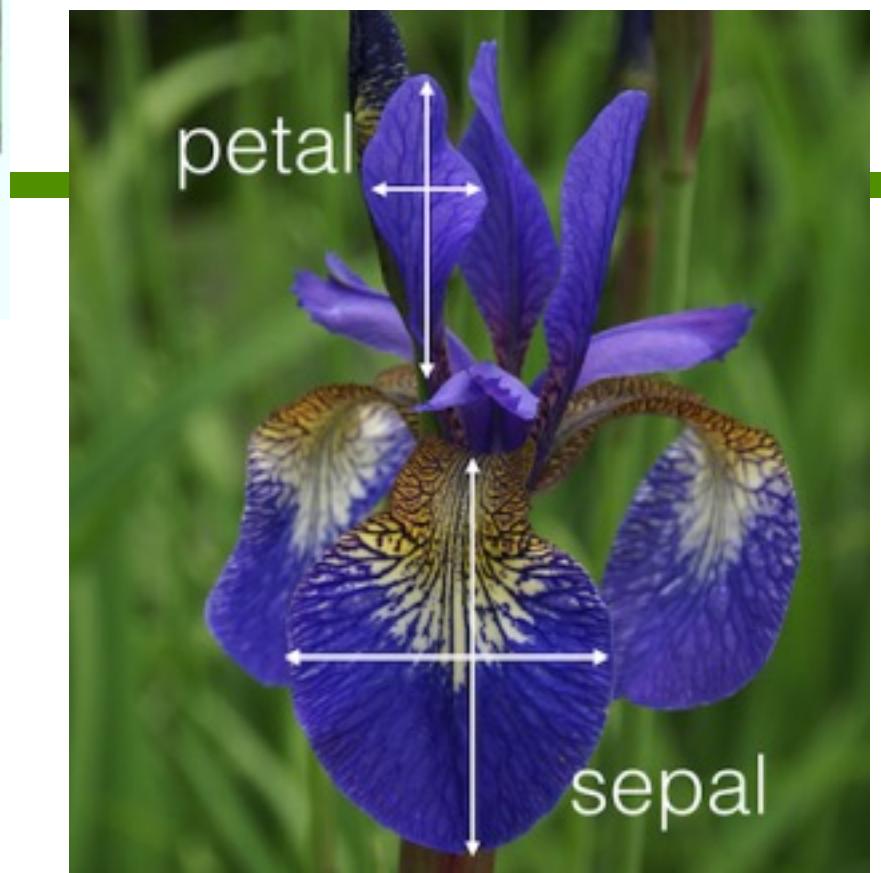
Ideal Data



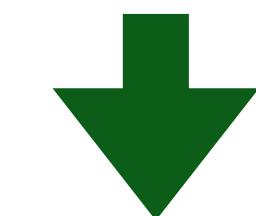
Setosa

Virginica

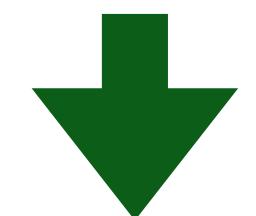
Versicolor



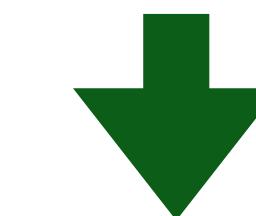
Numerical
Feature



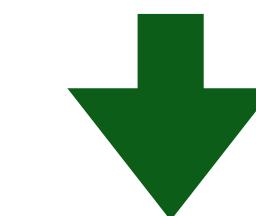
Numerical
Feature



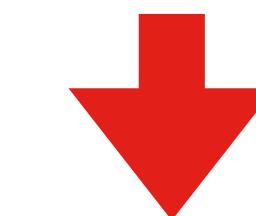
Numerical
Feature



Numerical
Feature



Label



sepal_length	sepal_width	petal_length	petal_width	Class
5.0	3.3	1.4	0.2	Iris-setosa
7.0	3.2	4.7	1.4	Iris-versicolor
5.7	2.8	4.1	1.3	Iris-versicolor
6.3	3.3	6.0	2.5	Iris-virginica

Dataset Size



Dataset Dimensionality

Record / Sample / Data Item

Label Value

Feature Value

Mixed Feature Types

- Data is rarely “clean”
 - Approximately 50-80 % of the time is spent on **data wrangling** - could be an under-estimate
- Having good data with the correct features is critical
- 3 issues to deal with:
 - (1) *Encoding features* as numerical values
 - (2) *Transforming features* to make ML algorithms work better
 - (3) Dealing with *missing feature values*

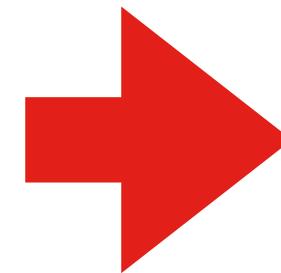
MSSubClass	MSZoning	LtFrontage	LtArea	Street	Alley	LtShape	...	MoSold	YrSold	SaleType	SaleCondition	SalePrice
20	RL	80.0	10400	Pave	NaN	Reg	...	5	2008	WD	Normal	174000
180	RM	35.0	3675	Pave	NaN	Reg	...	5	2006	WD	Normal	145000
60	FV	72.0	8640	Pave	NaN	Reg	...	6	2010	Con	Normal	215200
20	RL	84.0	11670	Pave	NaN	IR1	...	3	2007	WD	Normal	320000
60	RL	43.0	10667	Pave	NaN	IR2	...	4	2009	ConLw	Normal	212000
80	RL	82.0	9020	Pave	NaN	Reg	...	6	2008	WD	Normal	168500
60	RL	70.0	11218	Pave	NaN	Reg	...	5	2010	WD	Normal	189000
80	RL	85.0	13825	Pave	NaN	Reg	...	12	2008	WD	Normal	140000
60	RL	Nan	13031	Pave	NaN	IR2	...	7	2006	WD	Normal	187500

Data excerpt from houses

Easy case: features are already numerical (or Boolean)

- Each feature is assigned its own value in the feature space

IsAdult	Age
FALSE	17
TRUE	21
TRUE	34
FALSE	9

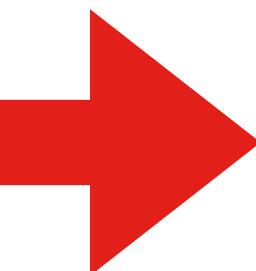


IsAdult	Age
0	17
1	21
1	34
0	9

One-hot encoding of categorical features

- Why not encode each value as an integer?
 - A naive integer encoding would create an ordering of the feature values that do not exist in the original data
 - You can try direct integer encoding if a feature does have a natural ordering (ORDINAL e.g. ECTS grades A–F)
- Each value of a categorical feature gets its own column

Status	Gender
Single	M
Married	F
Single	O
Single	M

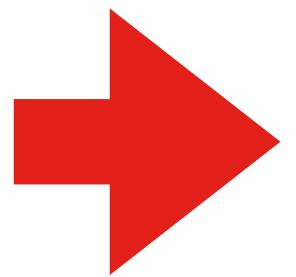


Status	Singl e	StatusMarri ed	GenderM	GenderF	GenderO
Single	1	0	1	0	0
Married	0	1	0	1	0
Single	1	0	0	0	1
Single	1	0	1	0	0

Encoding Ordinal Features

- Convert to a number, preserving the order
 - [low,medium,high] → [1,2,3]
- Encoding may not capture relative differences*

Health Status	Blood Pressure
Good	Very good
Very Good	Excellent
Normal	Good
Bad	Normal



Health Status	Blood Pressure
3	4
4	5
2	3
1	1

Data Issues

- Incorrect feature values
 - Typos
 - e.g., color = {"blue", "green", "gren", "red"}
 - Garbage
 - e.g., color = "wor--Sil"
 - Inconsistent spelling (e.g., "color", "colour") or capitalization
 - Inconsistent abbreviations (e.g., "Oak St.", "Oak Street")
- Missing labels
 - Delete instances if only a few are missing labels
 - Use semi-supervised learning techniques
 - Predict the missing labels via self-supervision

Merging Data

- Data may be split across different files
 - Requires doing a join based on a key to combine data into one table
- Problems During Merge
 - Inconsistent data
 - Same instance key with conflicting labels
 - Data duplication
 - The merged table may be too large for memory
 - Encoding issues
 - Inconsistent data formats or terminology
 - Key aspects mentioned in cell comments or auxiliary files

tracks

	A	B	C	D	E	F	G	H	I
1	id	name	album_id	media_type_id	genre_id	composer	milliseconds	bytes	unit_price
2	1	For Those About To Rock We Will Be Strong	1	1	1	Angus Young	343719	11170334	0.99
3	2	Balls to the Wall	2	2	1		342562	5510424	0.99
4	3	Fast As a Shark	3	2	1	F. Baltes, S. K.	230619	3990994	0.99
5	4	Restless and Wild	3	2	1	F. Baltes, R.A.	252051	4331779	0.99
6	5	Princess of the Night	3	2	1	Deaffy & R.A.	375418	6290521	0.99
7	6	Put The Finger	1	1	1	Angus Young	205662	6713451	0.99
8	7	Let's Get It Under Control	1	1	1	Angus Young	233926	7636561	0.99
9	8	Inject The Venom	1	1	1	Angus Young	210834	6852860	0.99
10	9	Snowballed	1	1	1	Angus Young	203102	6599424	0.99
11	10	Evil Walks	1	1	1	Angus Young	263497	8611245	0.99
12	11	C.O.D.	1	1	1	Angus Young	199836	6566314	0.99
13	12	Breaking The Habit	1	1	1	Angus Young	263288	8596840	0.99
14	13	Night Of The Living Dead	1	1	1	Angus Young	205688	6706347	0.99
15	14	Spellbound	1	1	1	Angus Young	270863	8817038	0.99

albums

	A	B	C	D
1	id	title	artist_id	
2	1	For Those About To Rock We Will Be Strong	1	
3	2	Balls to the Wall	2	
4	3	Restless and Wild	2	
5	4	Let There Be Rock	1	
6	5	Big Ones	3	
7	6	Jagged Little Pill	4	
8	7	Facelift	5	
9	9	Plays Metallica By Four	7	
10	10	Audioslave	8	
11	11	Out Of Exile	8	
12	12	BackBeat Soundtrack	9	
13	13	The Best Of Billy Cobham	10	
14	14	Alcohol Fueled Brewtanica	11	
15	15	Alcohol Fueled Brewtanica	11	

artists

	A	B	C	D
1	id	name		
2	1	AC/DC		
3	2	Accept		
4	3	Aerosmith		
5	4	Alanis Morissette		
6	5	Alice In Chains		
7	7	Apocalyptica		
8	8	Audioslave		
9	9	BackBeat		
10	10	Billy Cobham		
11	11	Black Label Society		
12	12	Black Sabbath		
13	13	Body Count		
14	14	Brave New Day		

Treating missing values

- Delete features with mostly missing values (columns)
- Delete instances with missing features (rows)
 - If rare
- **Feature imputation** methods try to "fill in the blanks"
- Variants:
 - replacing with a constant
 - the mean feature value (numerical)
 - the mode (categorical or ordinal)
 - "flag" missing values using out of range values
 - replacing with a random value
 - predicting the feature value from other features

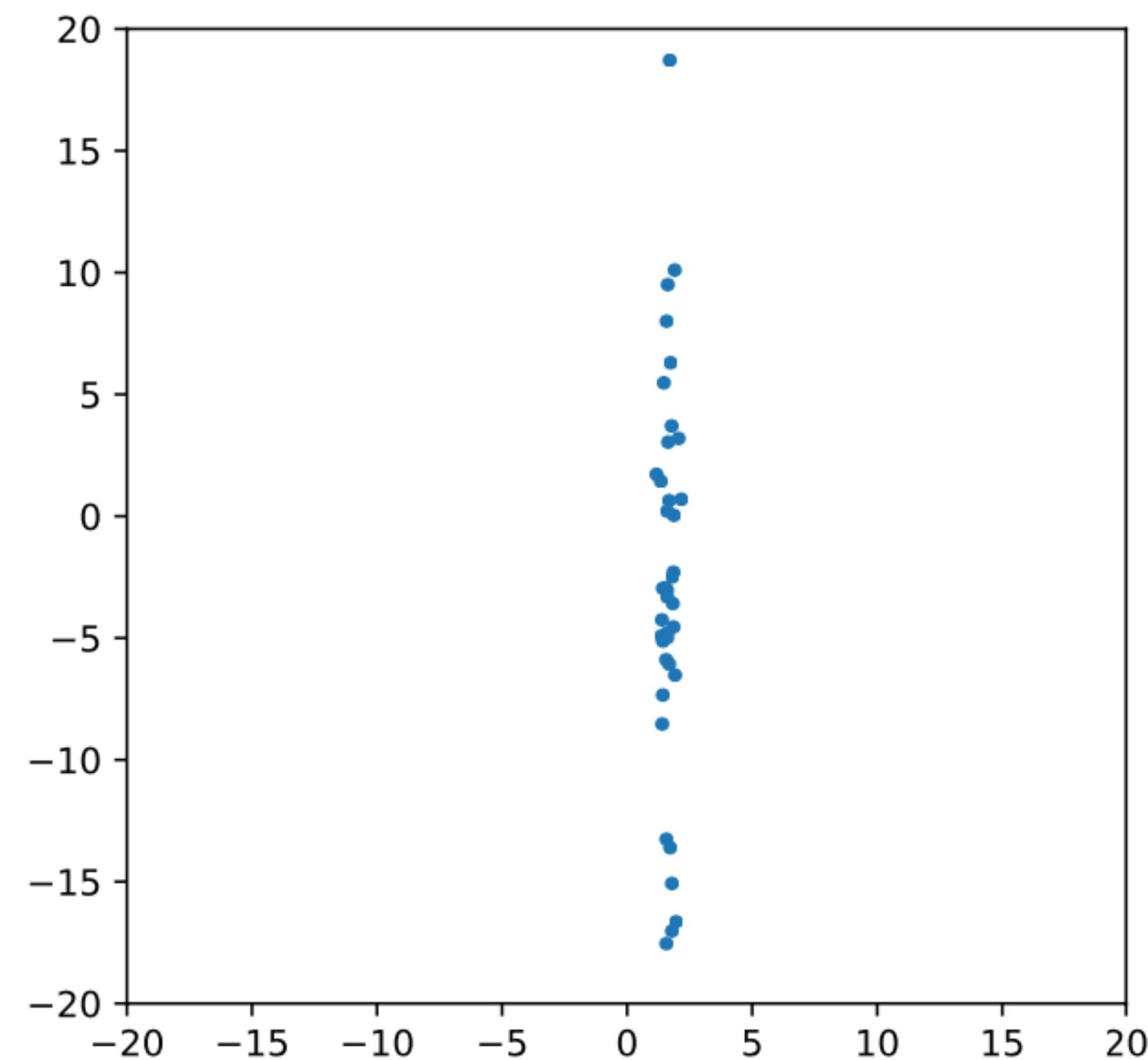
sepal_length	sepal_width	petal_length	petal_width	Class
5.0	3.3	1.4	0.2	Iris-setosa
7.0	NaN	4.7	1.4	Iris-versicolor
5.7	2.8	4.1	1.3	
6.3	NaN	6.0	2.5	Iris-virginica

Data might not be “missing at random” or
due to technical issues

It might be meaningful that instances have
missing features!

What if our features look like this?

- What if the features have different magnitudes?
- Does it matter if a feature is represented as meters or millimeters?
- What if there are outliers?
- Values spread strongly affects many models:
 - linear models (linear SVM, logistic regression, . . .)
 - neural networks
 - models based on distance or similarity (e.g., kNN)
- It does not matter for most tree-based predictors
 - they just consider thresholds of one feature at a time



Feature Normalisation

- Normalization is needed for many algorithms to work properly
 - Or to speed up training

- Min/Max scaling
 - Values scaled between 0 and 1

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- Standard scaling
 - Rescales features to have zero mean and unit variance
 - Outliers can cause problems

$$x_{new} = \frac{x - \mu_x}{\sigma_x}$$

- Scaling to unit length (typically for document)

$$x_{new} = \frac{x}{|x|}$$

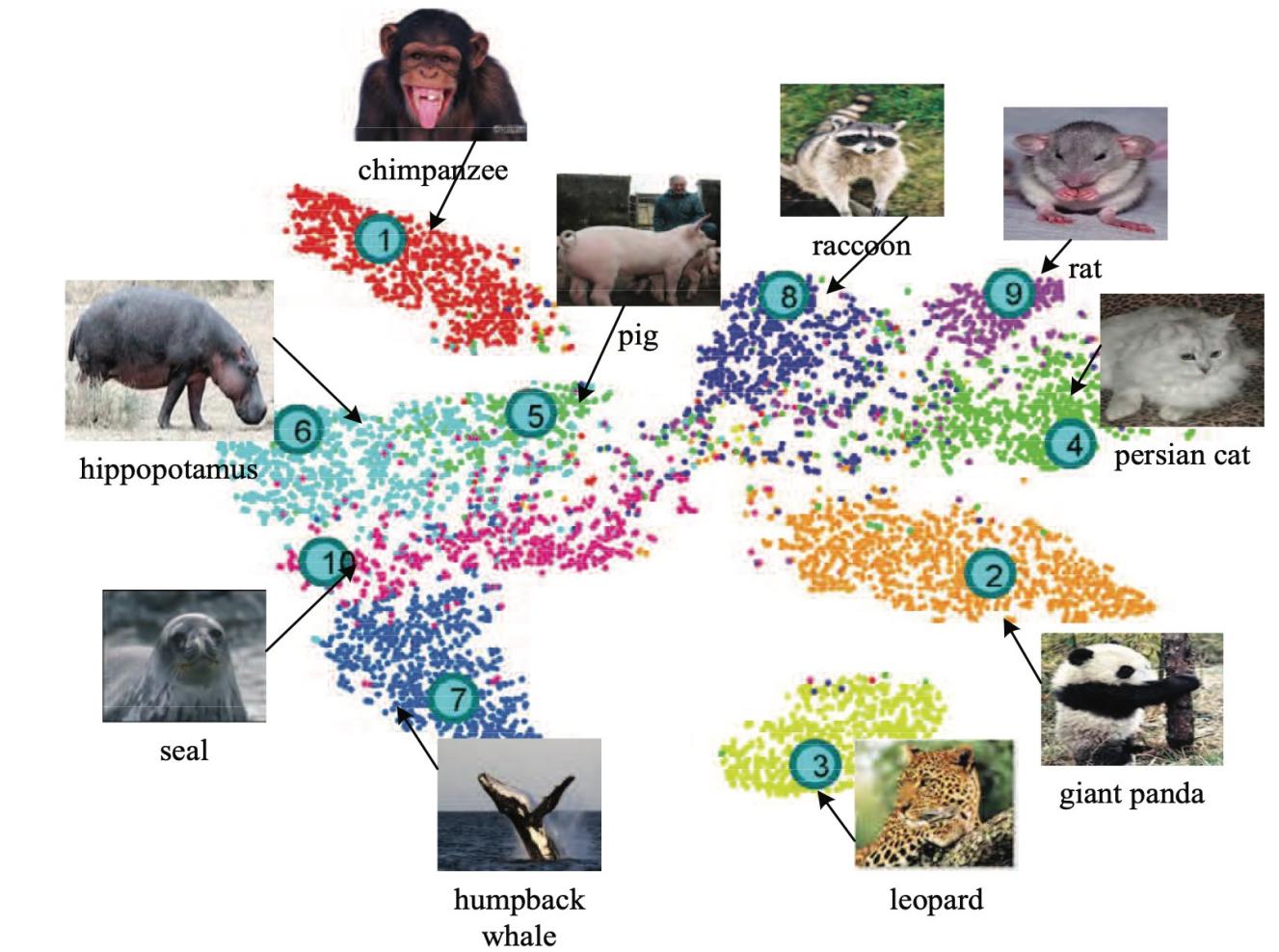
Other feature transformations

- we may try to improve performance by trying other transformations
 - logarithm, square root, . . .
 - TF-IDF (term frequency–inverse document frequency)
- Trial and error, exploration and your intuition

Feature Selection and Removal

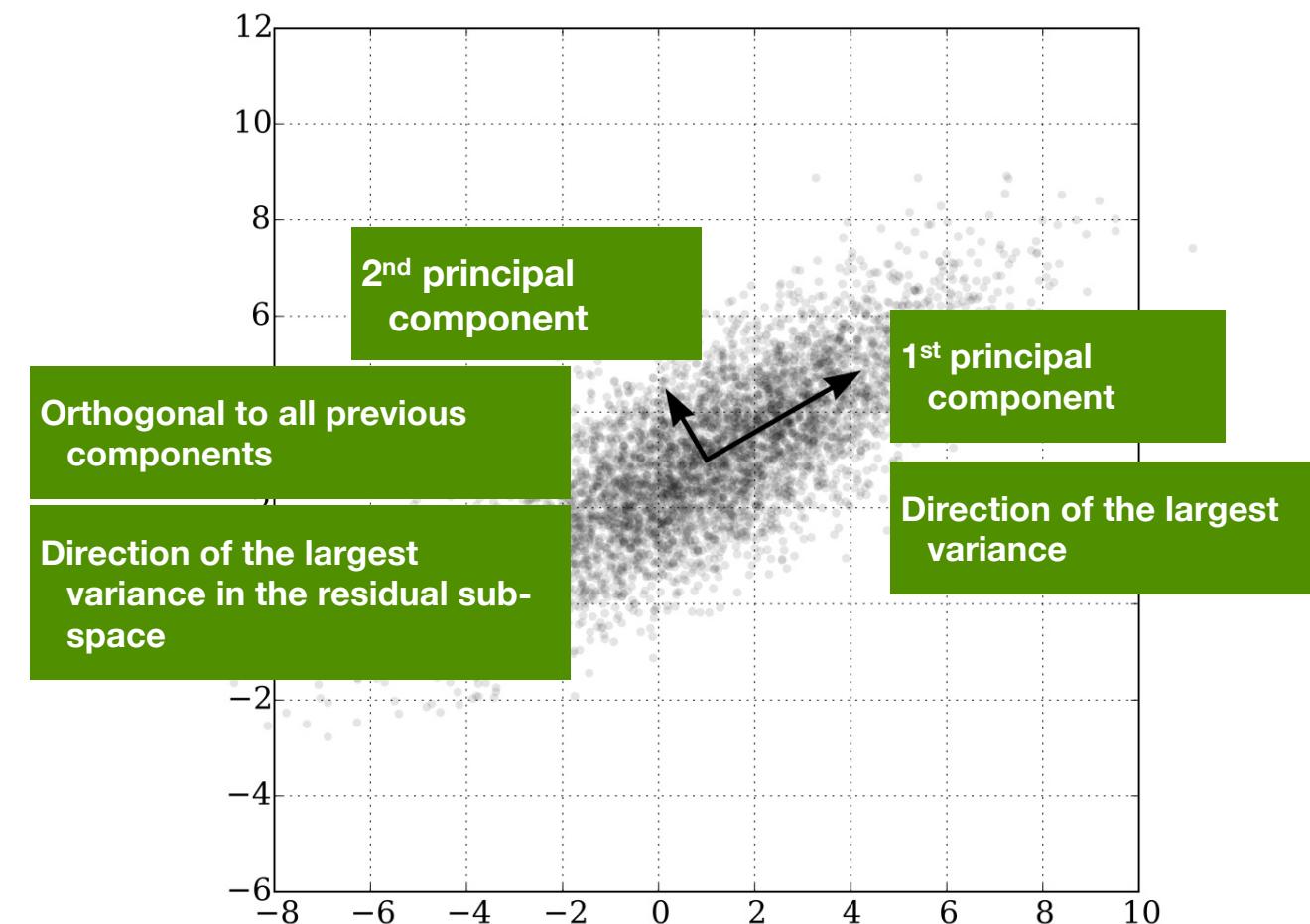
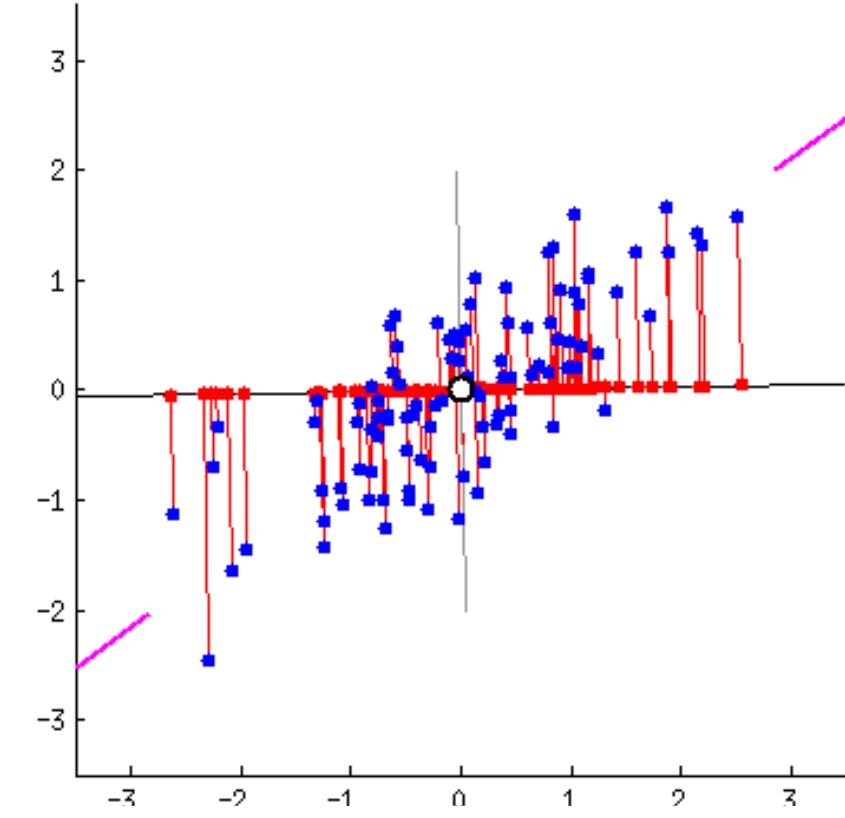
- In some cases, the number of features may be very large leading to several problems:
 - Important information is drowned out
 - Longer model training time
 - More complexity \Rightarrow bad for generalization
- Solution: leave out some features. But which ones?
 - Feature selection methods can find a useful subset
- **Idea:** find a subspace that retains most of the information about the original data
 - Pretty much as we were doing with Word embeddings
 - PRO: fewer dimensions make for datasets that are easier to explore and visualise, and faster training of ML algorithms
 - CONS: drop in prediction accuracy (less information)
- There are many different methods, *Principal Component Analysis* is a classic

Image from: <https://arxiv.org/pdf/1703.08893.pdf>



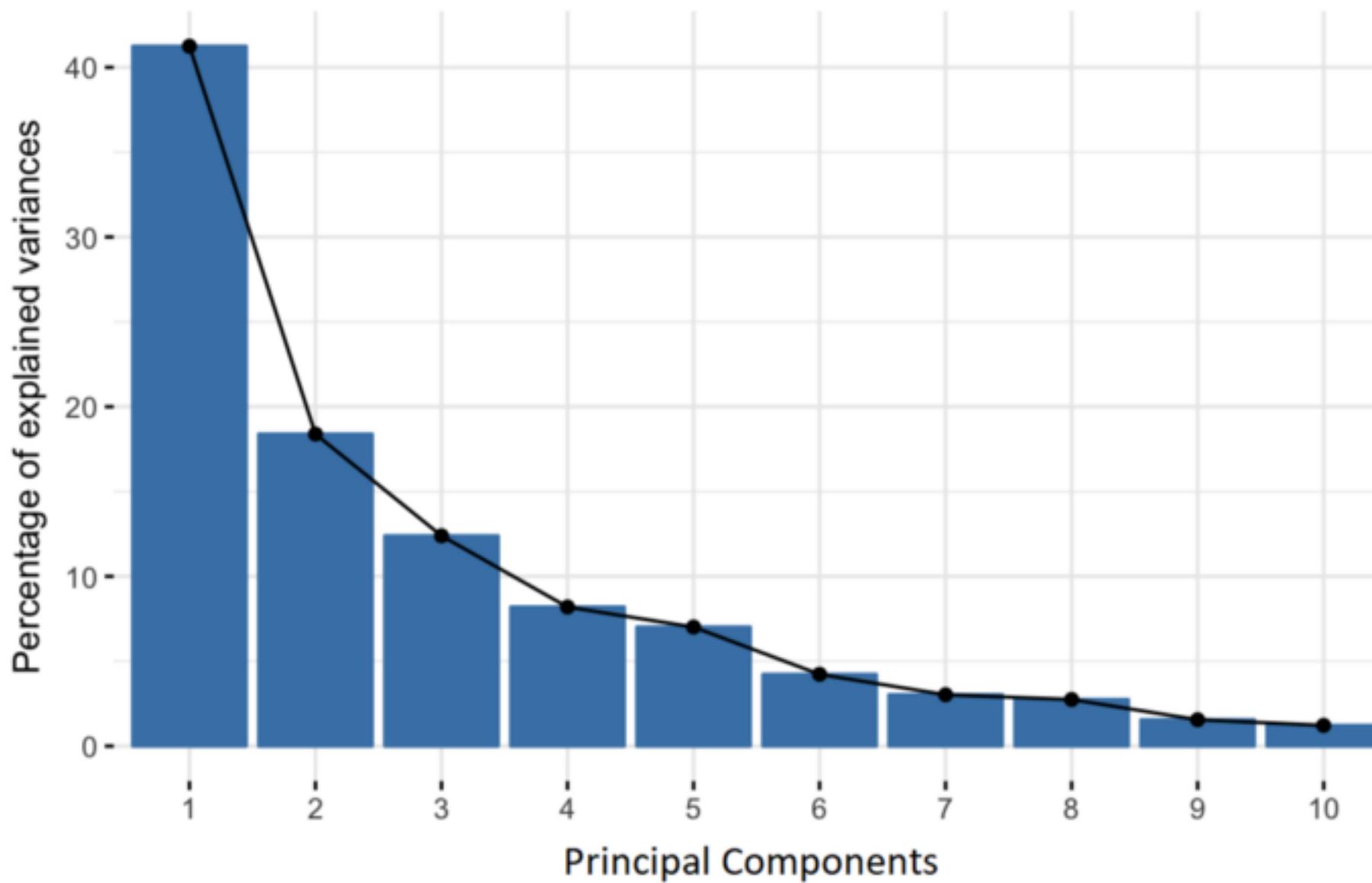
Principal Component Analysis

- Sometimes features are highly correlated with each other, therefore containing redundant information
- **Principal components** are new features that are constructed as linear combinations or mixtures of the initial features
 - Orthogonal projection of data onto lower-dimension linear space that:
 - maximizes the variance of projected data (purple line)
 - minimizes mean squared distance between data point and projections (sum of red lines)
- The new features (i.e., principal components) are uncorrelated
 - Most of the information within the initial features is compressed into the first components



Dimensionality Reduction

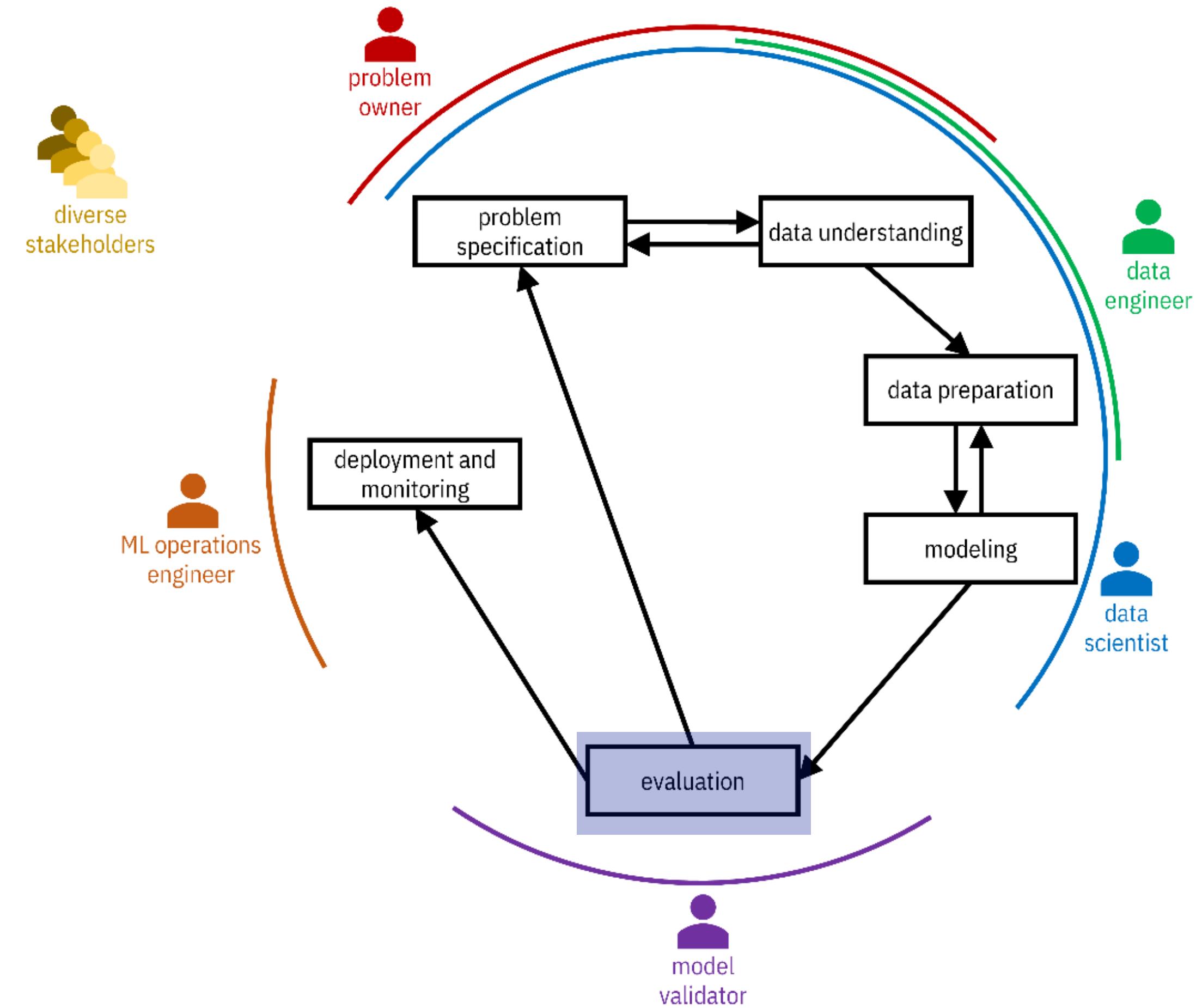
- Use the PCA transformation of the data instead of the original features
 - PCA keeps most of the variance of the data
 - So, we are reducing the dataset to features that retain meaningful variations of the dataset



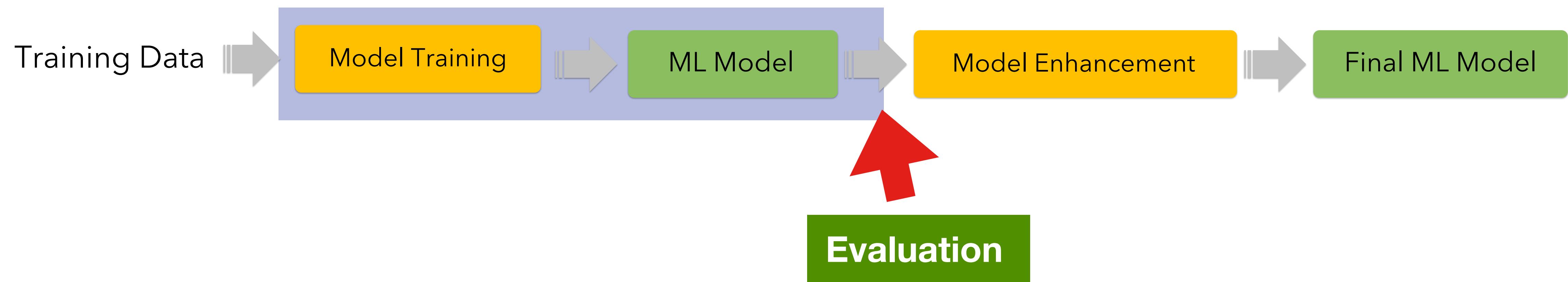
- Ignore the components of less significance (e.g. only pick the first 3 components)

ML Model Evaluation

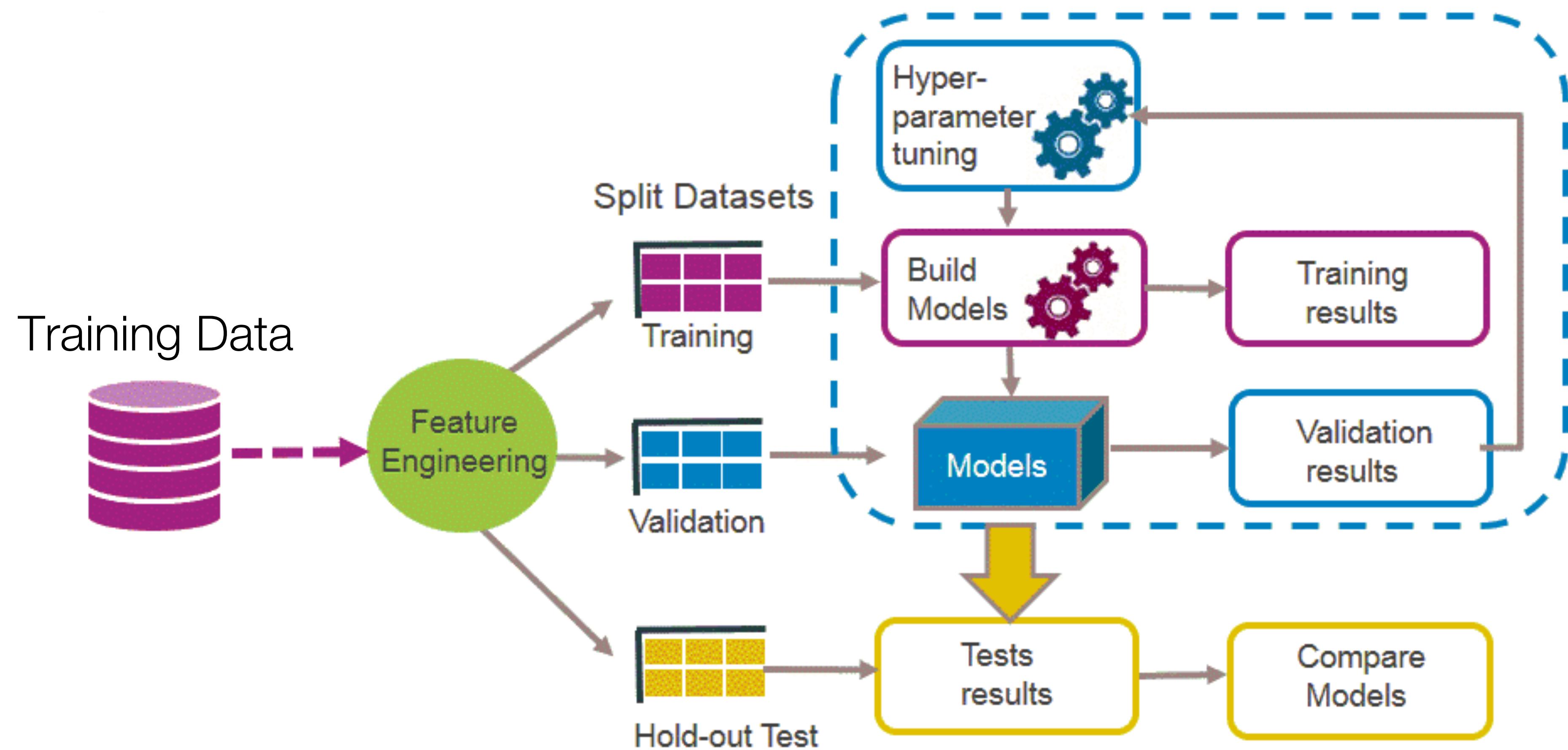
Cross-Industry Standard Process for Data Mining (CRISP-DM) methodology



How do machines learn?

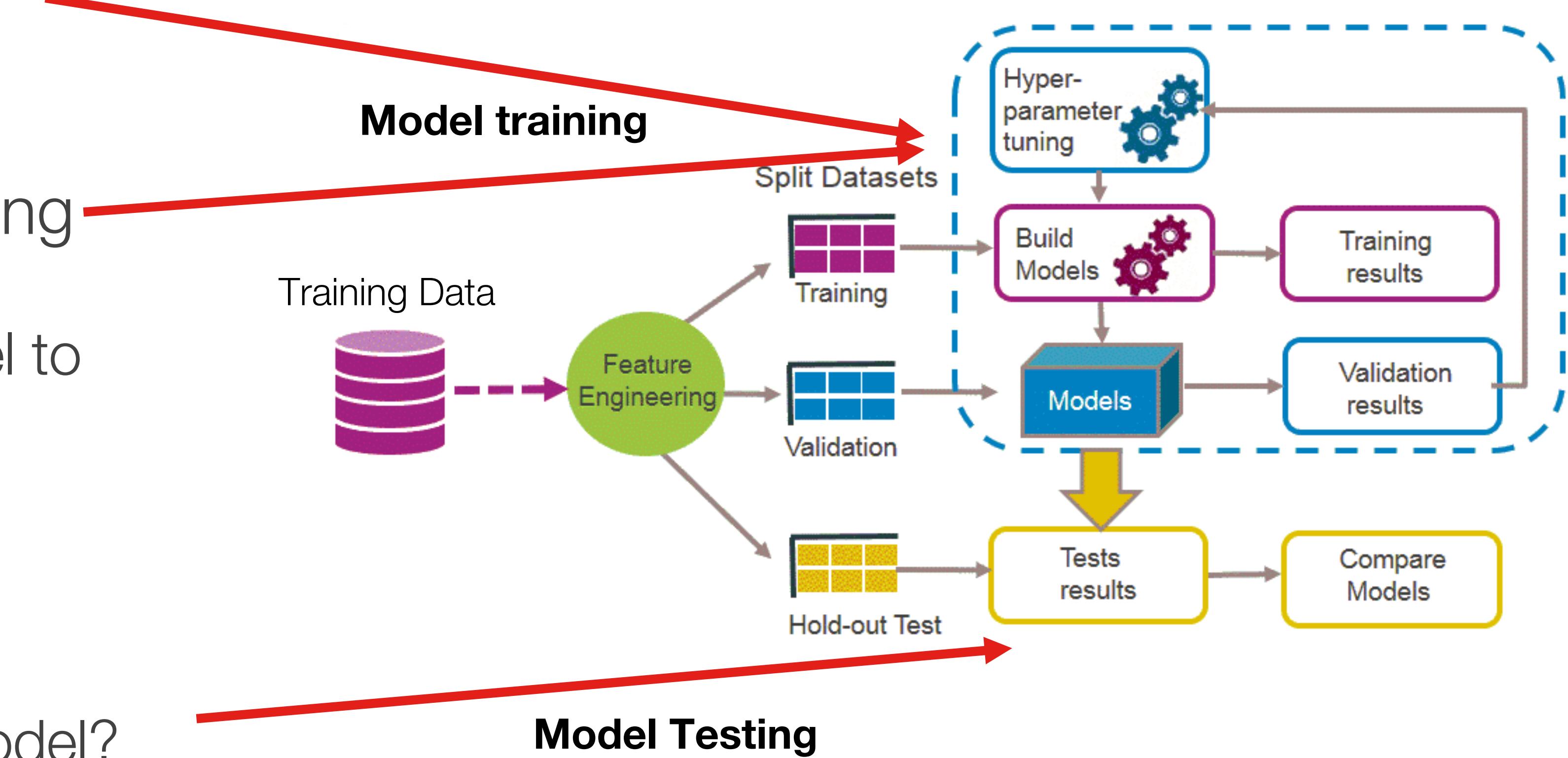


Machine Learning Training and Evaluation Process



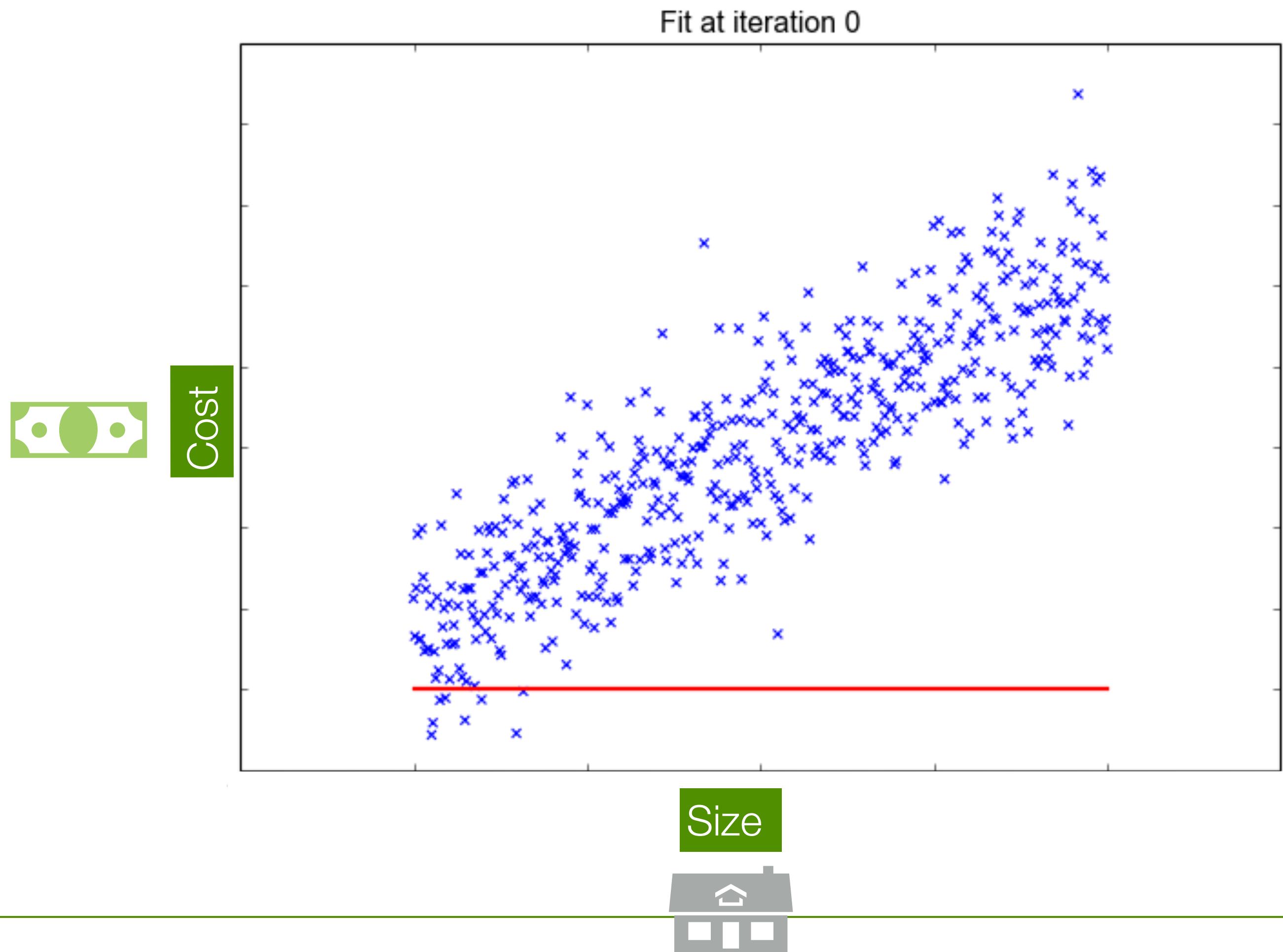
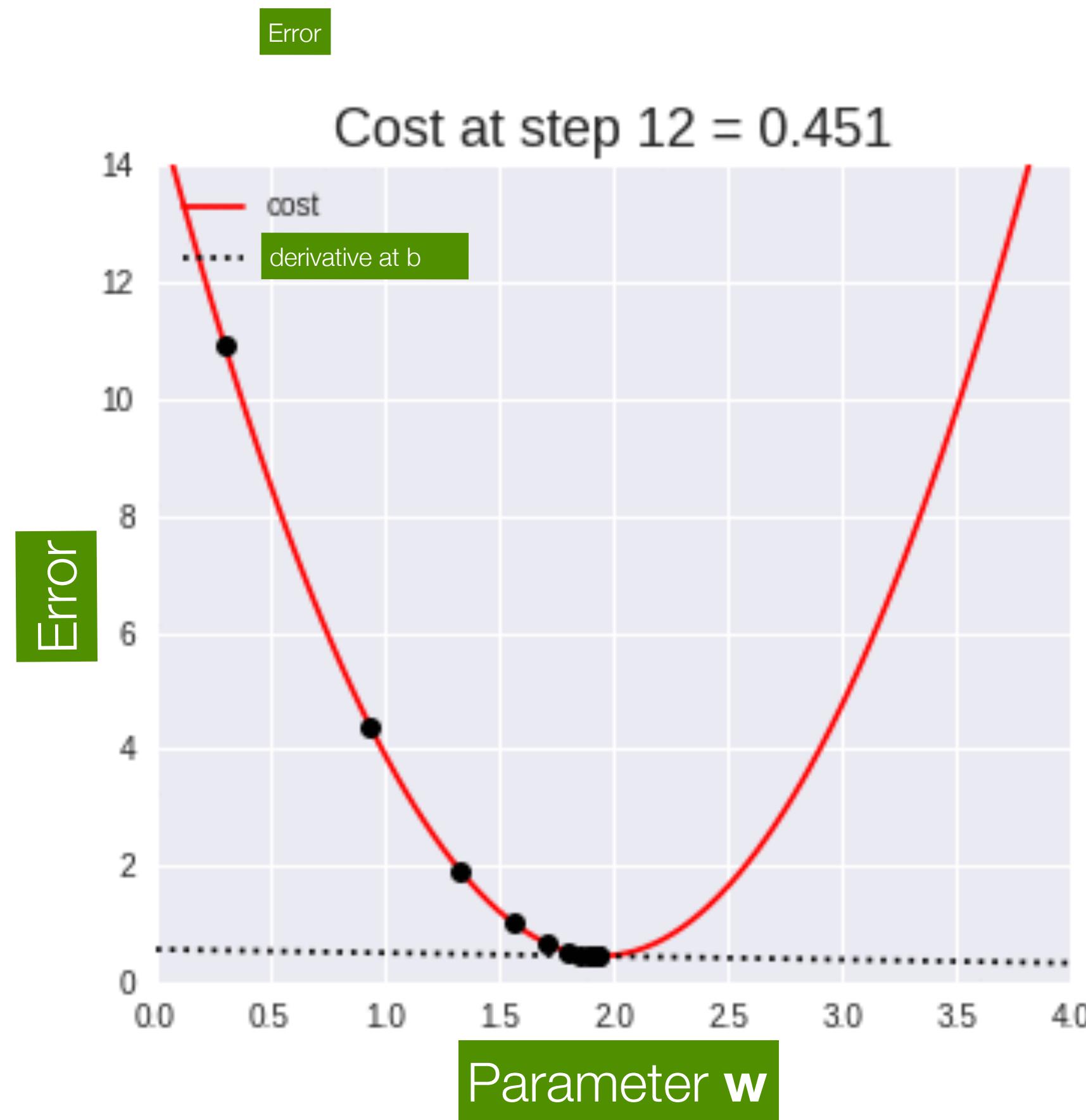
How to Evaluate?

- Metric
 - How to measure errors?
 - Both training and testing
- Training of Machine Learning algorithm
 - How to “help” the ML model to generalise?
- Experiment
 - How to pick the best ML model?



Training the model

■ Gradient descent



Model Training: Error as a Metric

- Errors are almost inevitable!
 - How to measure errors?
- We're generally interested in the following:
 - How often is the prediction wrong?
 - How is the prediction wrong?
 - What is the cost of wrong predictions?
 - How does the cost vary by the type of prediction that was wrong?
 - How can we minimize costs? (or regret?)
- Select an evaluation procedure (a “metric”)
 - **Ok, but which one?**

Regression

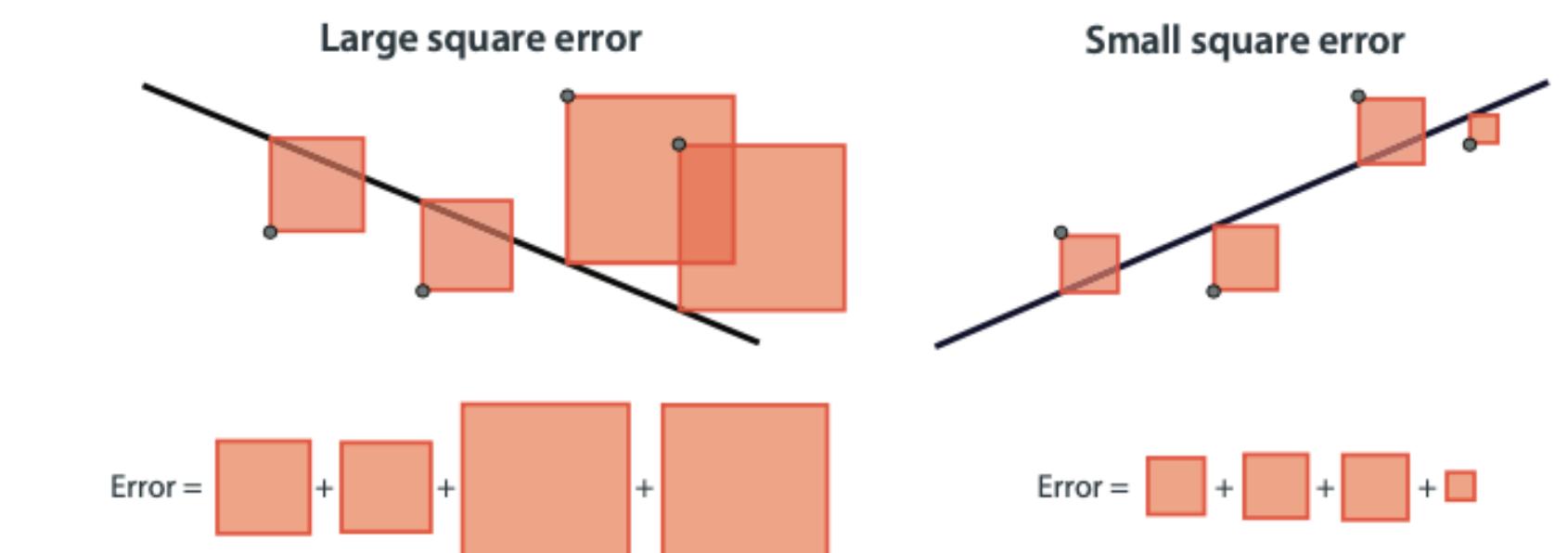
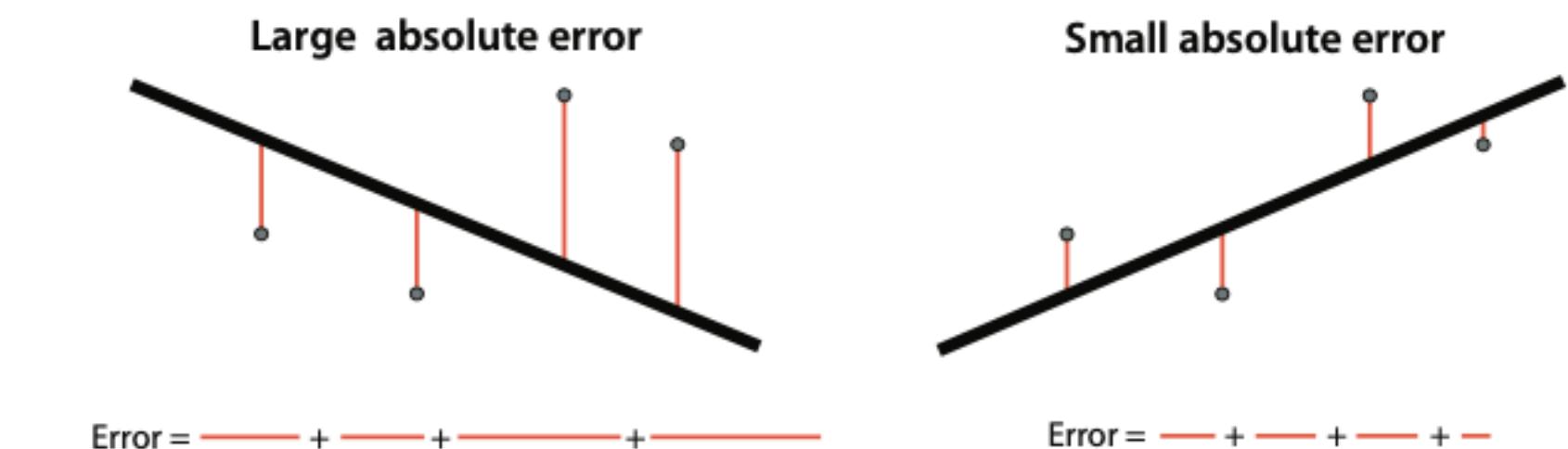
- **Mean (absolute | squared) error**

- **Absolute**: average of the difference between the original values and the predicted value
 - No direction

$$MAE = \frac{1}{N} \sum_{i=1}^N |prediction_i - value_i|$$

- **Squared**: average of the square of the difference between the original values and the predicted value
 - Squared is *nicer* to deal with during the training process (derivative)
 - Larger errors are more pronounced
 - More sensitive to outliers

$$MSE = \frac{1}{N} \sum_{i=1}^N (prediction_i - value_i)^2$$



Classification

■ Accuracy

- The percentage of times that a model is correct
- However, the model with the highest accuracy is not necessarily the best model
- Some errors (e.g. False Negative) may be much more expensive than others
 - Usually due to imbalanced trained datasets

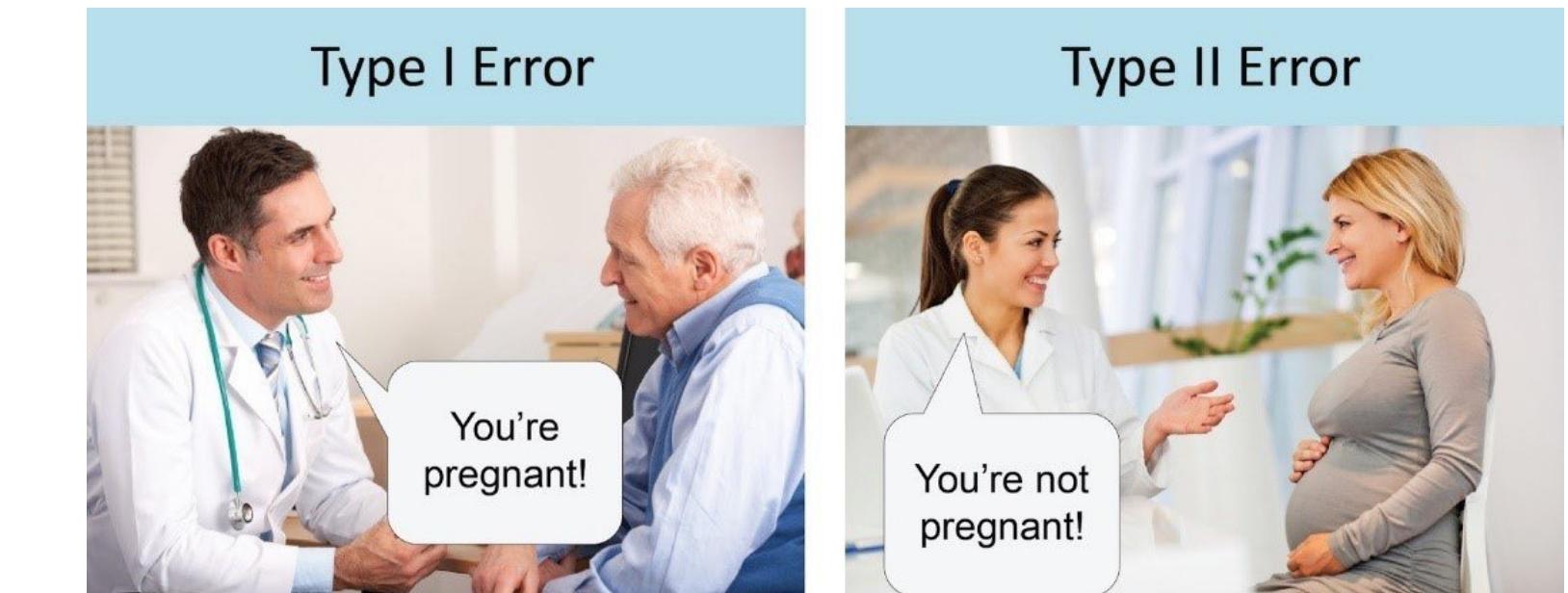
$$\text{Accuracy} = \frac{\# \text{CorrectPredictions}}{\# \text{Predictions}}$$

■ Confusion Matrix

- Describes the complete performance of the model

		Actual Class	
		Yes	No
Predicted Class	True Positive	50	10
	False Positive - <i>false alarm!</i> (Type I Error)	10	
Predicted Class	Yes	40	100
	True Negative	40	

False Negative - *underestimation*
(Type II Error)

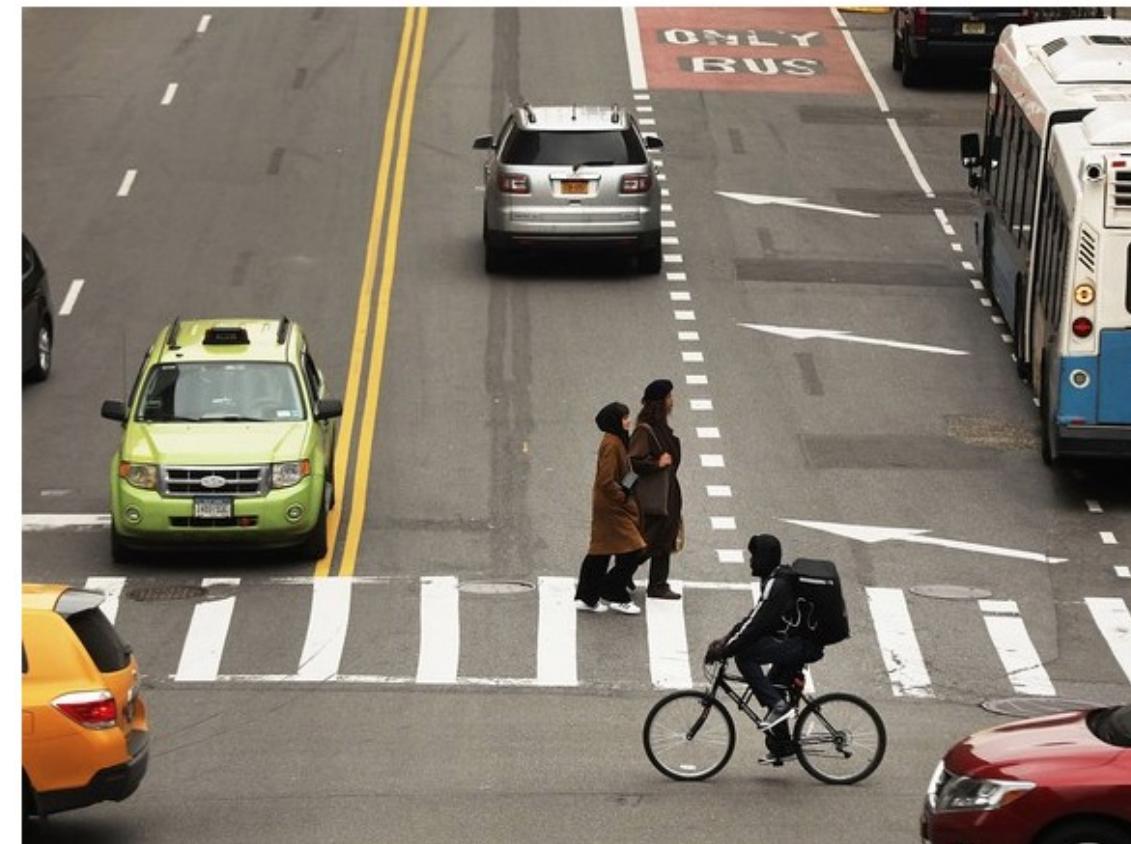


$$\text{Accuracy} = \frac{\# \text{TruePositives} + \# \text{TrueNegatives}}{\# \text{AllPredictions}}$$

Not all errors are equal

- Depending on your task, different errors have different costs
- Pregnancy detection
 - Cost of “false negatives”?
 - Cost of “false positives”?
- Covid testing
 - Cost of “false negatives”?
 - Cost of “false positives”?
- In law enforcement?
- In detecting the “Alexa” command?
- In detecting a person on the road?

FALSE POSITIVES: SELF-DRIVING CARS AND THE AGONY OF KNOWING WHAT MATTERS



According to a preliminary report released by the National Transportation Safety Board last week, Uber’s system detected pedestrian Elaine Herzberg six seconds before striking and killing her. It identified her as an unknown object, then a vehicle, then finally a bicycle. (She was pushing a bike, so close enough.) About a second before the crash, the system determined it needed to slam on the brakes. But Uber hadn’t set up its system to act on that decision, the NTSB explained in the report. The engineers prevented their car from making that call on its own “to reduce the potential for erratic vehicle behavior.” (The company relied on the car’s human operator to avoid crashes, which is a whole separate problem.)

Uber’s engineers decided not to let the car auto-brake because they were worried the system would overreact to things that were unimportant or not there at all. They were, in other words, very worried about false positives.

READ MORE



<https://www.wired.com/story/self-driving-cars-uber-crash-false-positive-negative/>

Tesla autopilot:

https://youtube.com/shorts/tpOg87AQvbo?si=n-F1G0rZuae_pqsV

Classification

- **Precision**

- Among the examples we classified as positive, how many did we correctly classify?

- **Recall (sensitivity)**

- Among the positive examples, how many did we correctly classify?

- **F1-Score**

- The harmonic mean between **precision** (how many instances correctly classified), and **recall** (how many relevant instances are correctly classified)
- What is the implicit assumption about the costs of errors?

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

$$F_1 = 2 * \frac{1}{\frac{1}{Precision} + \frac{1}{Recall}}$$

Classification

- **Sensitivity** (True positive rate)
 - the capacity of the model to identify the positively labelled points
 - Same as recall

$$Sensitivity = \frac{TruePositive}{FalseNegative + TruePositive}$$

- **Specificity** (False positive rate)
 - the capacity of the model to identify the negatively labeled points
 - Not the same as precision

$$Specificity = \frac{TrueNegative}{FalsePositive + TrueNegative}$$

	Predicted positive	Predicted negative
Positive label	True positives	False negatives
Negative label	False positives	True negatives

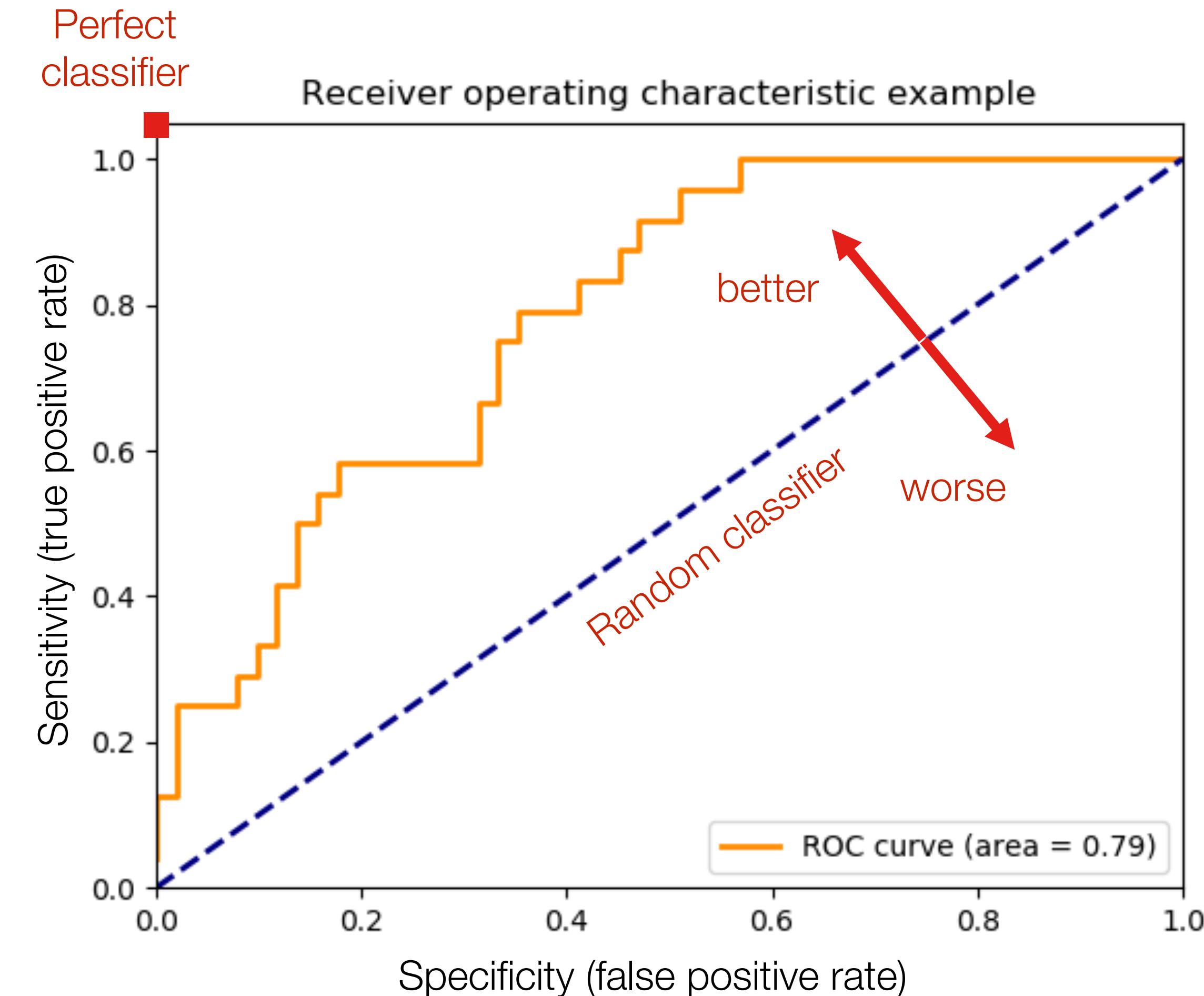
Recall (sensitivity) = $\frac{TP}{TP + FN}$

Specificity = $\frac{TN}{FP + TN}$

Precision = $\frac{TP}{TP + FP}$

The Receiver Operating Characteristic (ROC) curve

- A useful technique to evaluate a model based on its performance on false positives and negatives at the same time
 - based on *sensitivity* and *specificity*
- It also gives us a way to “explore” model performance visually
 - Trade-off specificity and sensitivity by moving the threshold



Some Examples

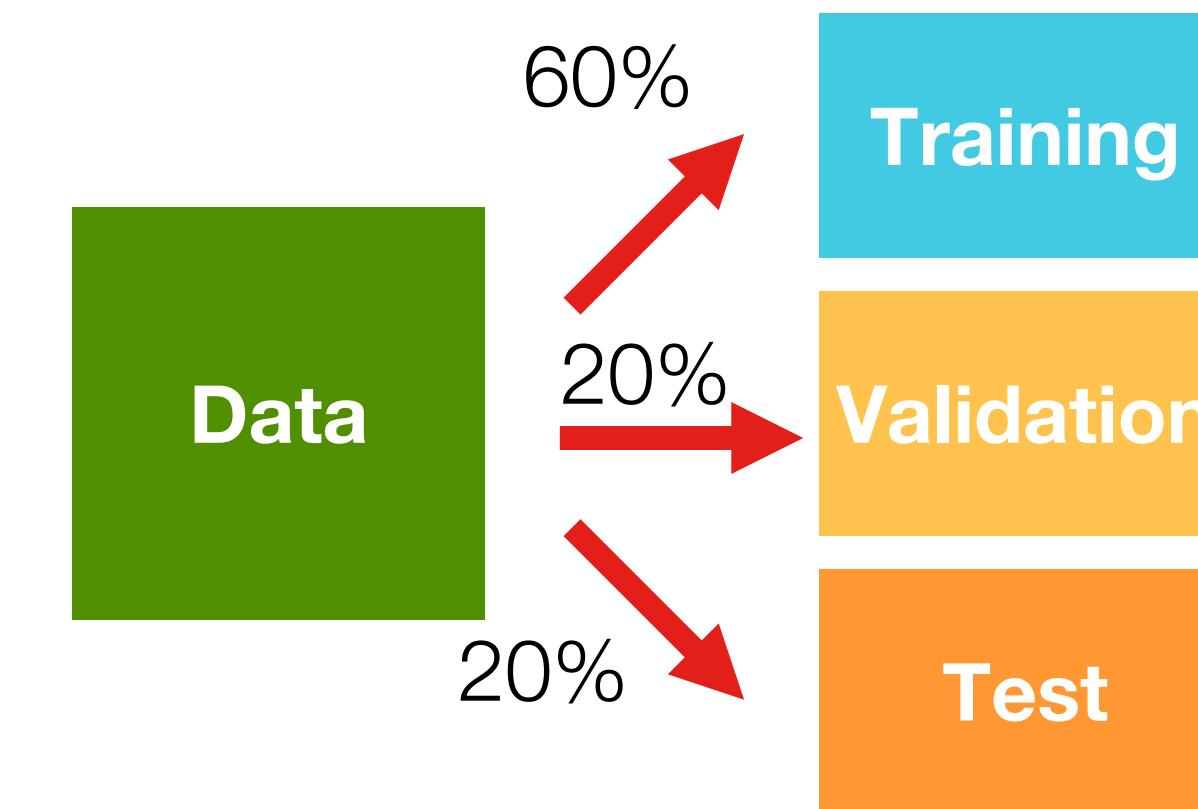
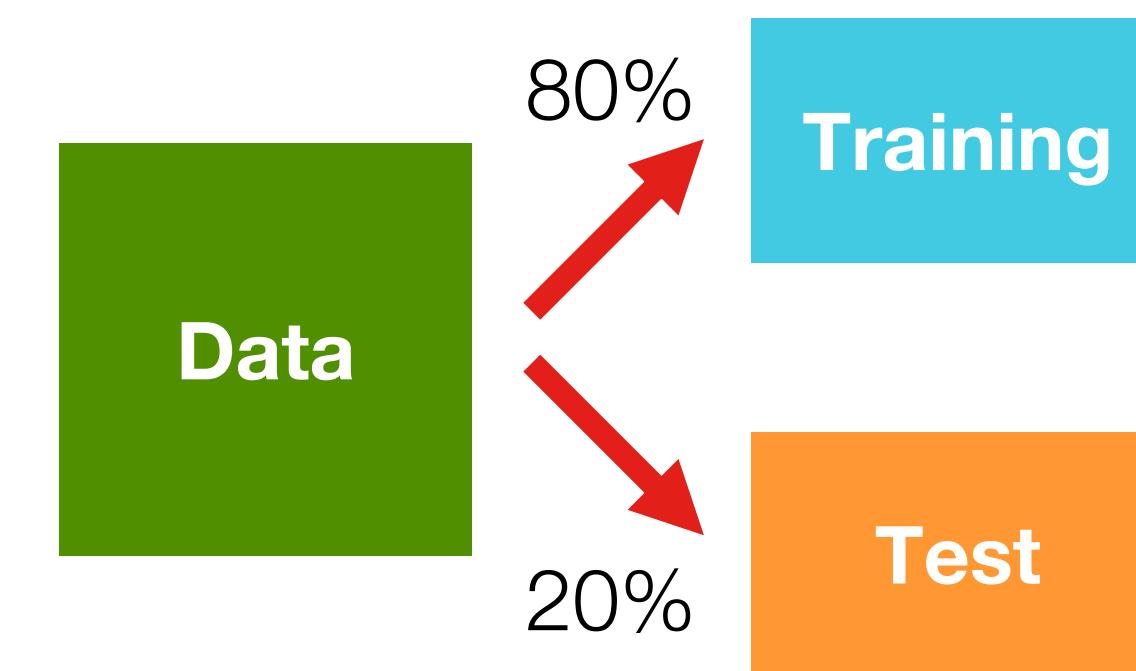
- Medical model:
 - **Recall** and **sensitivity**: among the sick people (positives), how many were correctly diagnosed as sick?
 - **Precision**: among the people diagnosed as sick, how many were actually sick?
 - **Specificity**: among the healthy people (negatives), how many were correctly diagnosed as healthy?
- Email model:
 - **Recall** and **sensitivity**: among the spam emails (positives), how many were correctly deleted?
 - **Precision**: among the deleted emails, how many were actually spam?
 - **Specificity**: among the spam emails (negatives), how many were correctly sent to the inbox?

Choosing Metrics

- When a high precision is a hard constraint:
 - search engine results, grammar correction: Intolerant to FP
 - Metric: Prioritize precision over recall
- When a high recall is a hard constraint:
 - medical diagnosis: Intolerant to FN
 - Metric: Prioritize recall over precision
- When both precision and recall are top priorities:
 - Automated surveillance systems: intolerant to FP and FN
 - Metric: F1 score
- Capacity constrained (by K)
 - Metric: Precision in top-K
 - What is the precision of the top 3 (K=3) recommendations or results
 - Relevant in recommender systems and search engines (e.g., Spotify and Google search)

Model Training: dataset splitting

- Split your data
 - Training set —> to **train** the model
 - (Optional) Validation set —> to decide which model to use
 - Test set —> to evaluate the model



NEVER use the test set for training —> That is CHEATING!

Model Training: Cross-validation

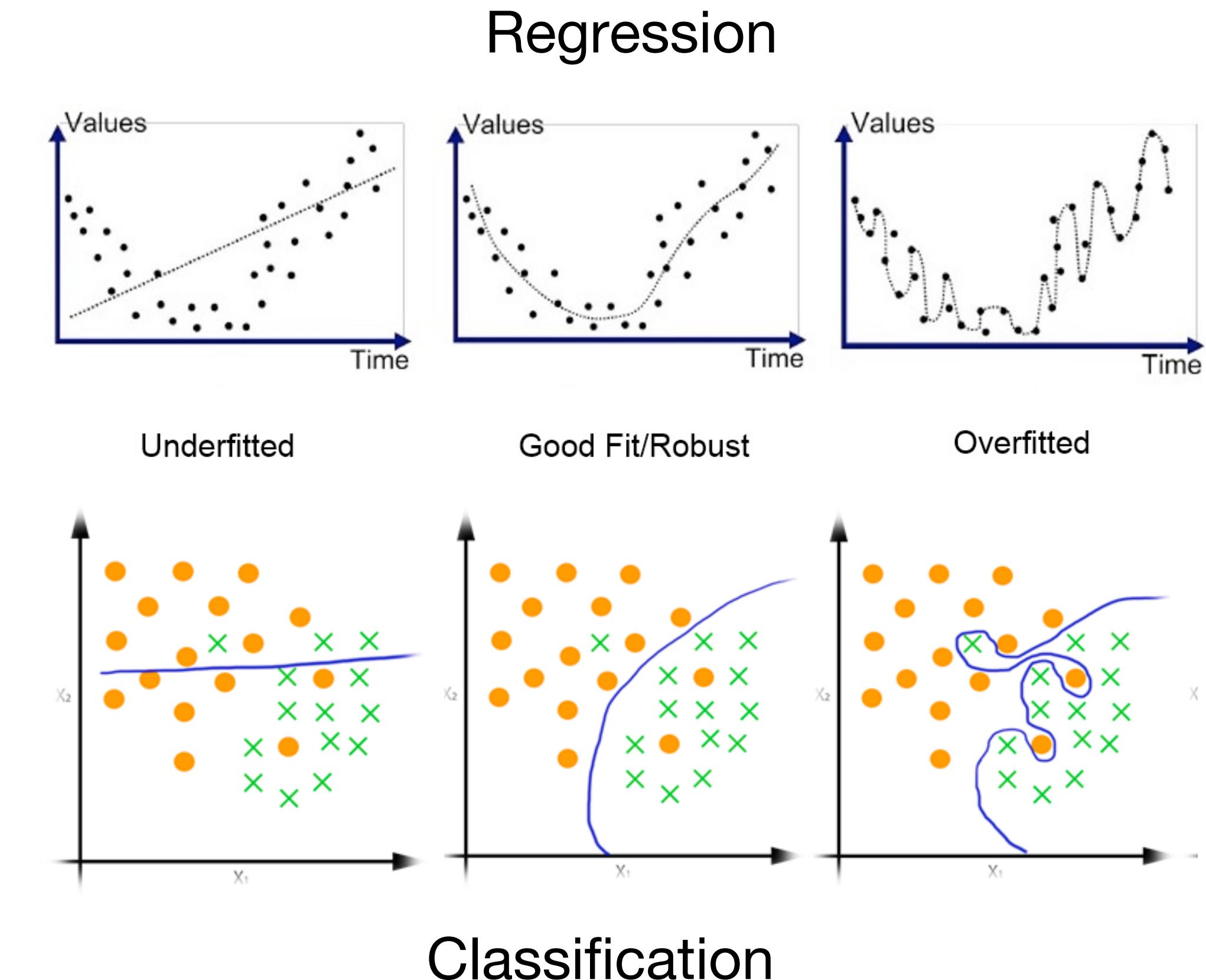
- A way to use all the data for training and testing, by recycling it several times
 - Split the data in n portions
 - Train the model n times using $n-1$ portions for training
- Useful when dataset is small



$$Score = \frac{Score_1 + Score_2 + Score_3 + Score_4}{4}$$

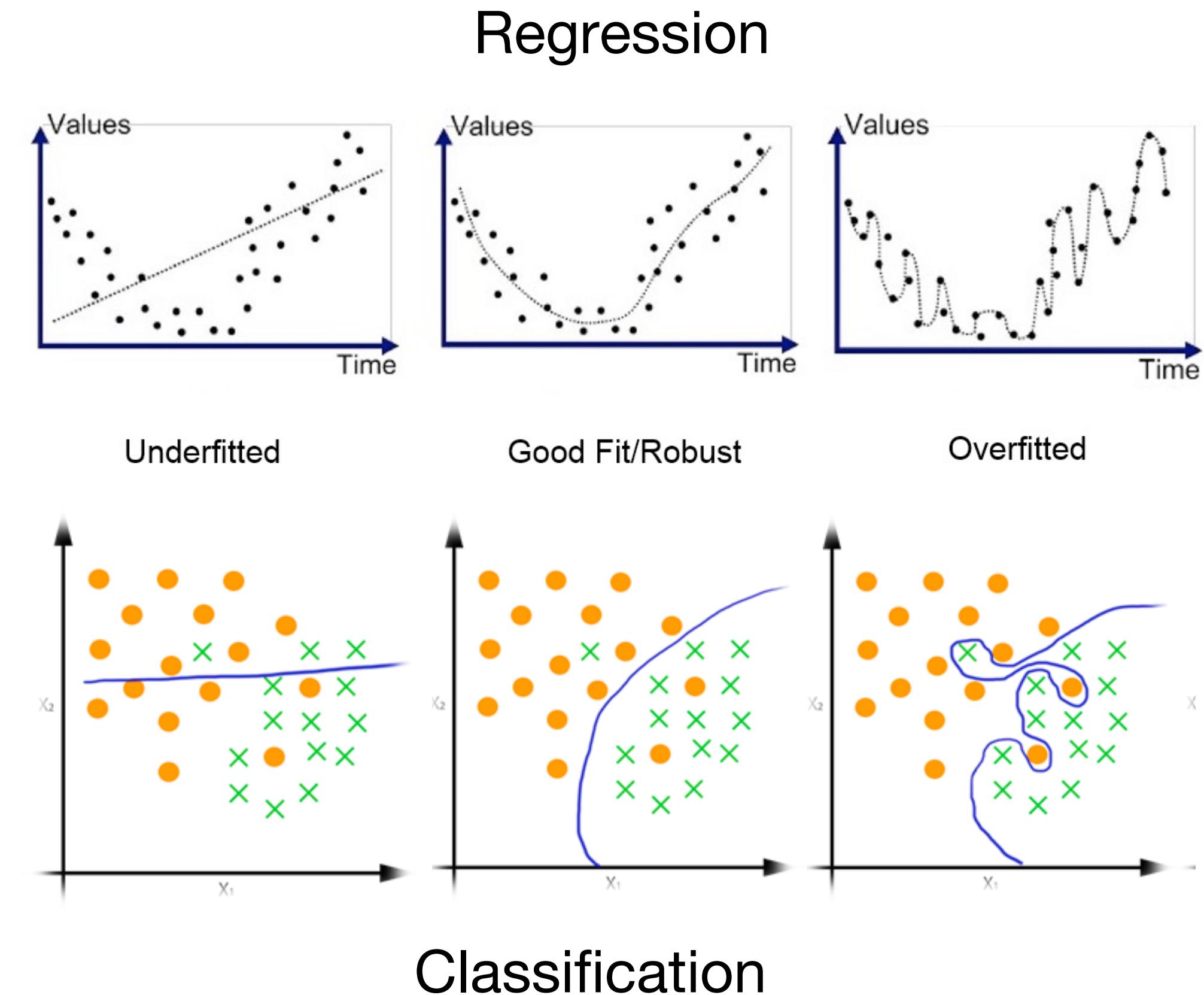
No free lunch

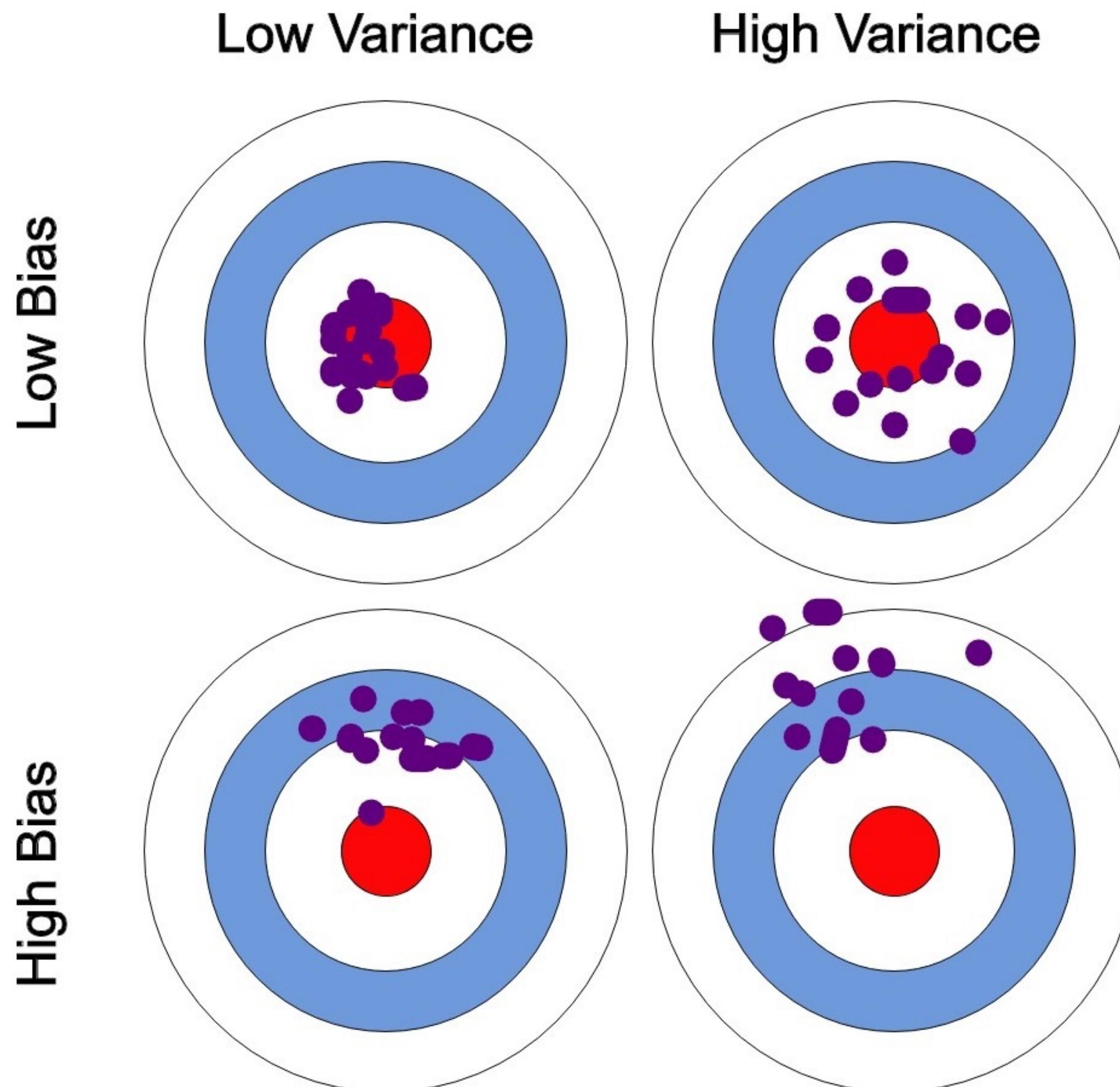
- There is no one best machine learning algorithm for all problems and datasets
- **Generalization**
 - How well does a trained model generalize from the data it was trained on to a new evaluation set?
 - Out-of-sampling or (out-of-distribution) testing



Generalisation

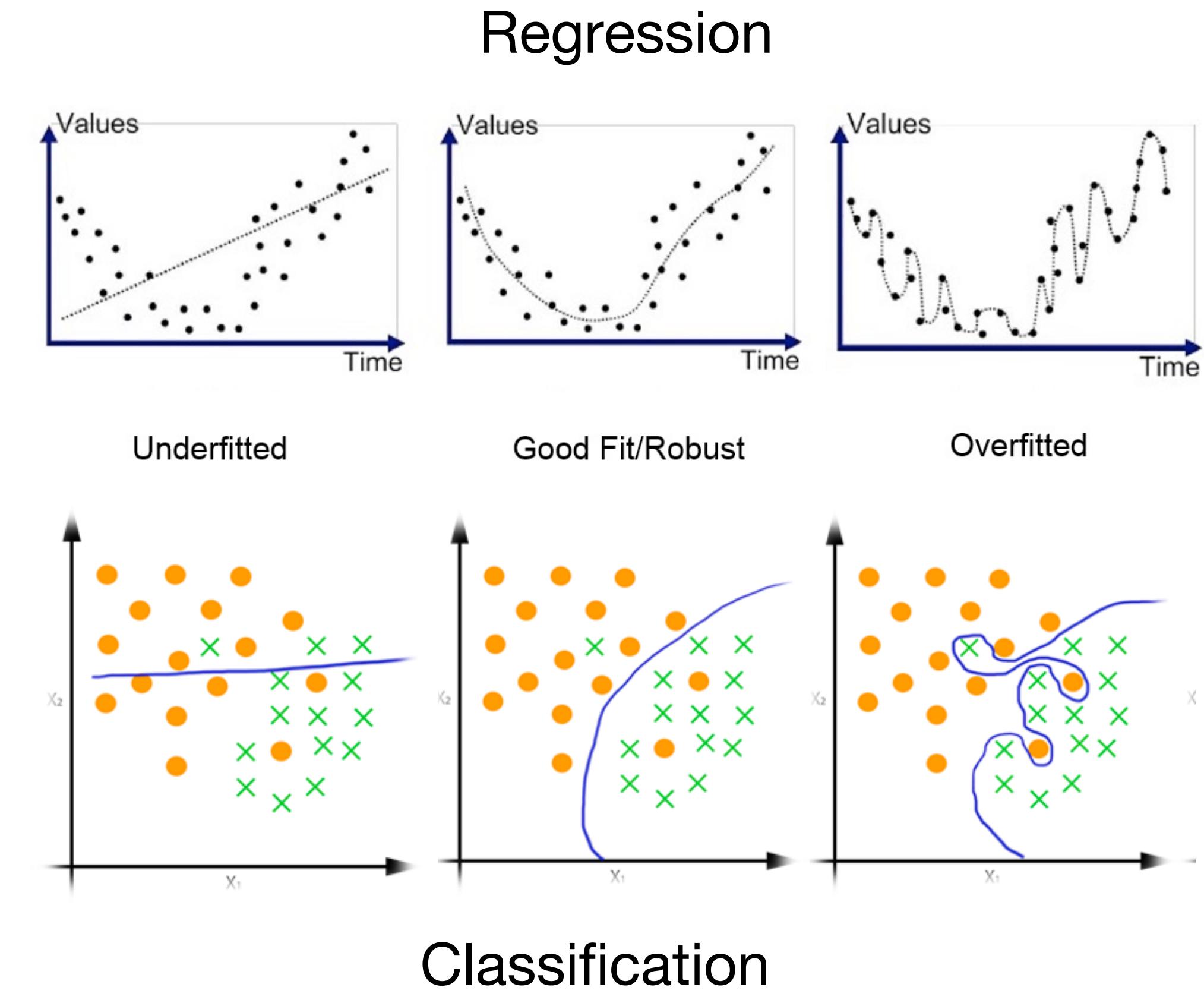
- **Challenge:** achieving good generalization and a small error rate
- Components of expected loss
 - **Noise** in our observations: **unavoidable**
 - **Bias**: how much the average model differs from the true model (i.e., the ideal)
 - Error due to inaccurate assumptions/simplifications made by the model
 - **Variance**: how much models trained on different training sets differ from each other
 - Too much sensitivity to the samples → overfitting





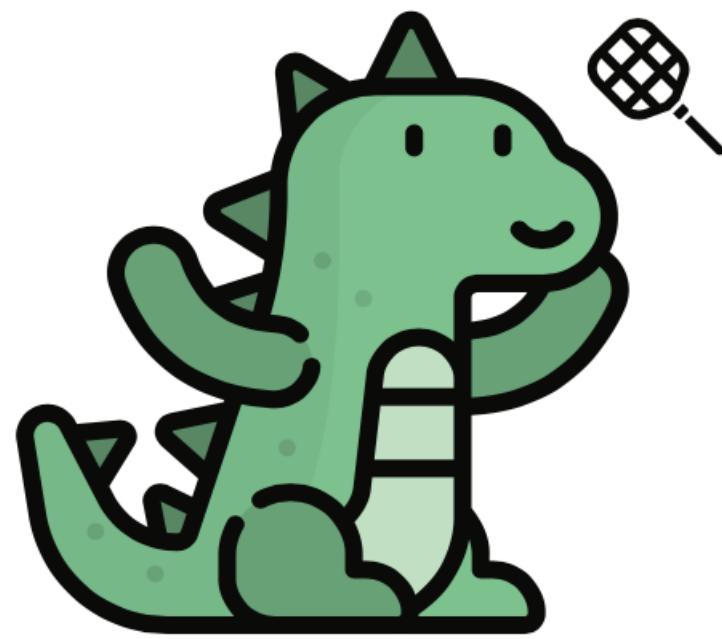
Overfitting vs. Underfitting

- Protect against **overfitting**
 - training a model that too closely matches the idiosyncrasies of the training data
 - model is too “complex” and fits irrelevant characteristics (noise) in the data
 - Low bias and high variance
 - Low training error and high-test error
- Protect against **underfitting**
 - training a model that does not adequately capture the patterns in the training data
 - The model is too “simple” to represent all the relevant class characteristics
 - High bias and low variance
 - High training error and high-test error

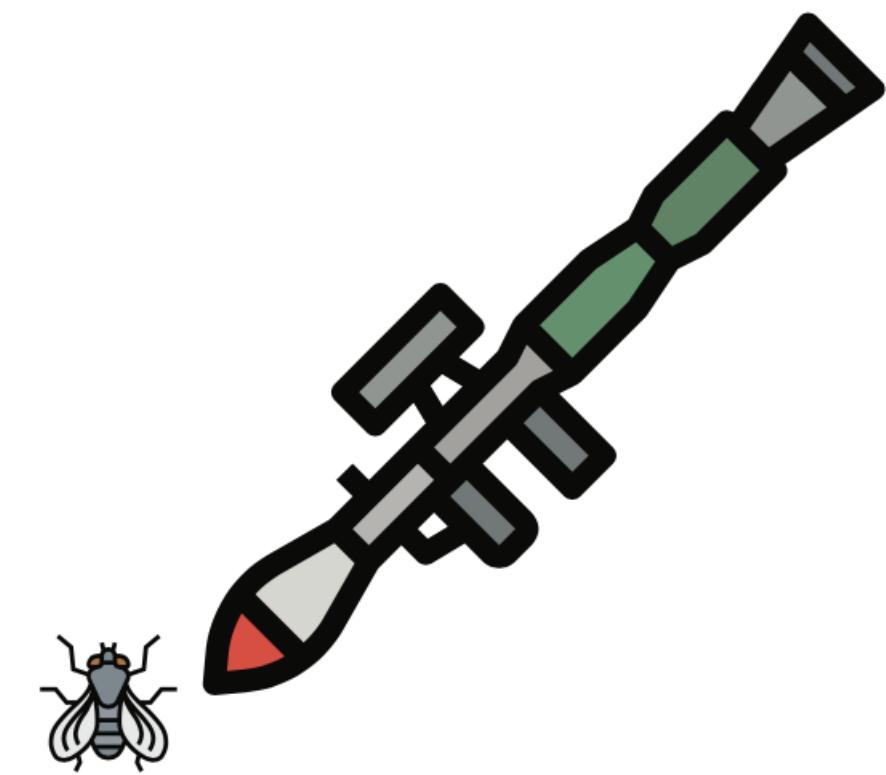


Overfitting vs. Underfitting

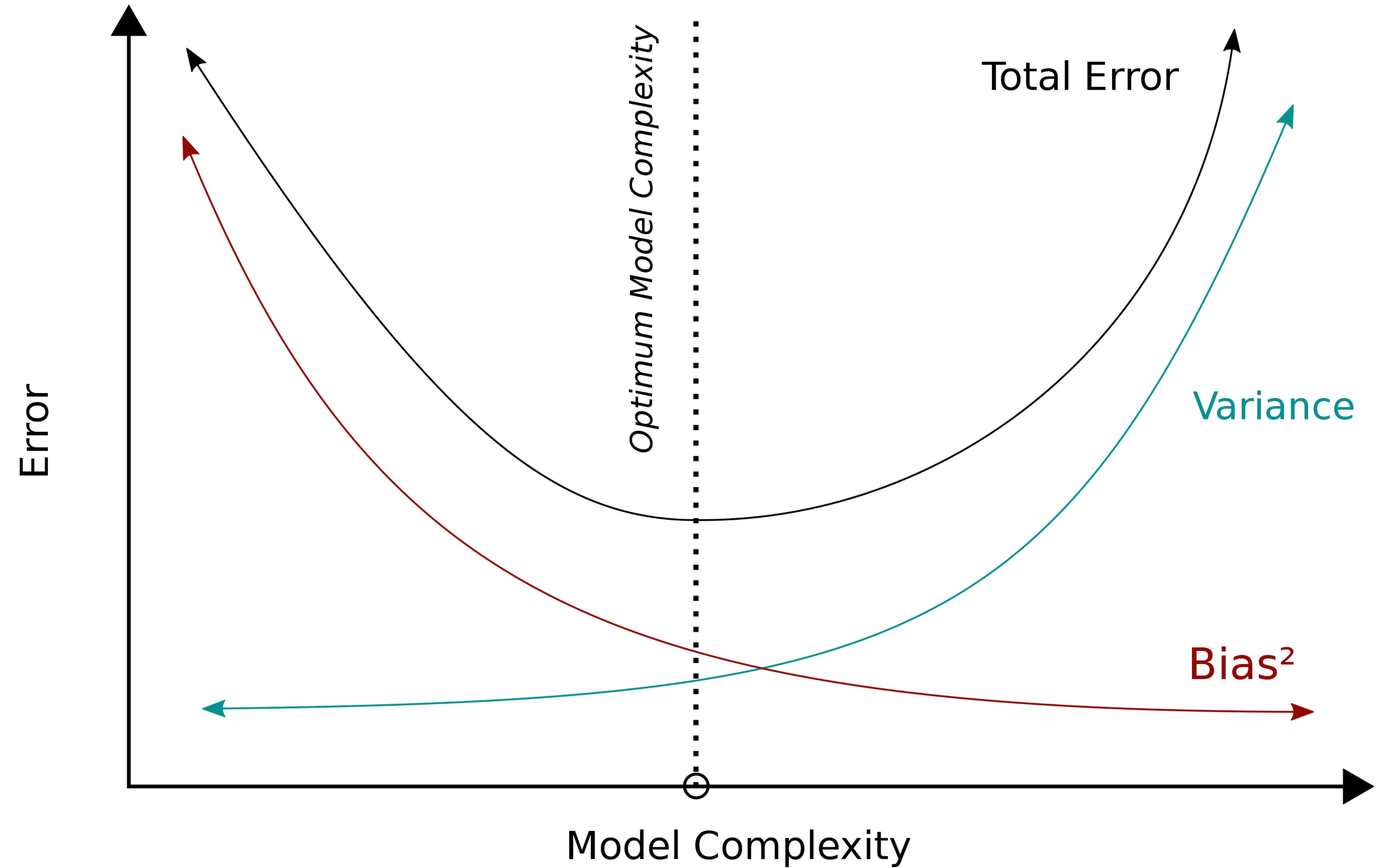
- Protect against **overfitting**
 - training a model that too closely matches the idiosyncrasies of the training data
 - model is too “complex” and fits irrelevant characteristics (noise) in the data
 - Low bias and high variance
 - Low training error and high-test error
- Protect against **underfitting**
 - training a model that does not adequately capture the patterns in the training data
 - The model is too “simple” to represent all the relevant class characteristics
 - High bias and low variance
 - High training error and high-test error

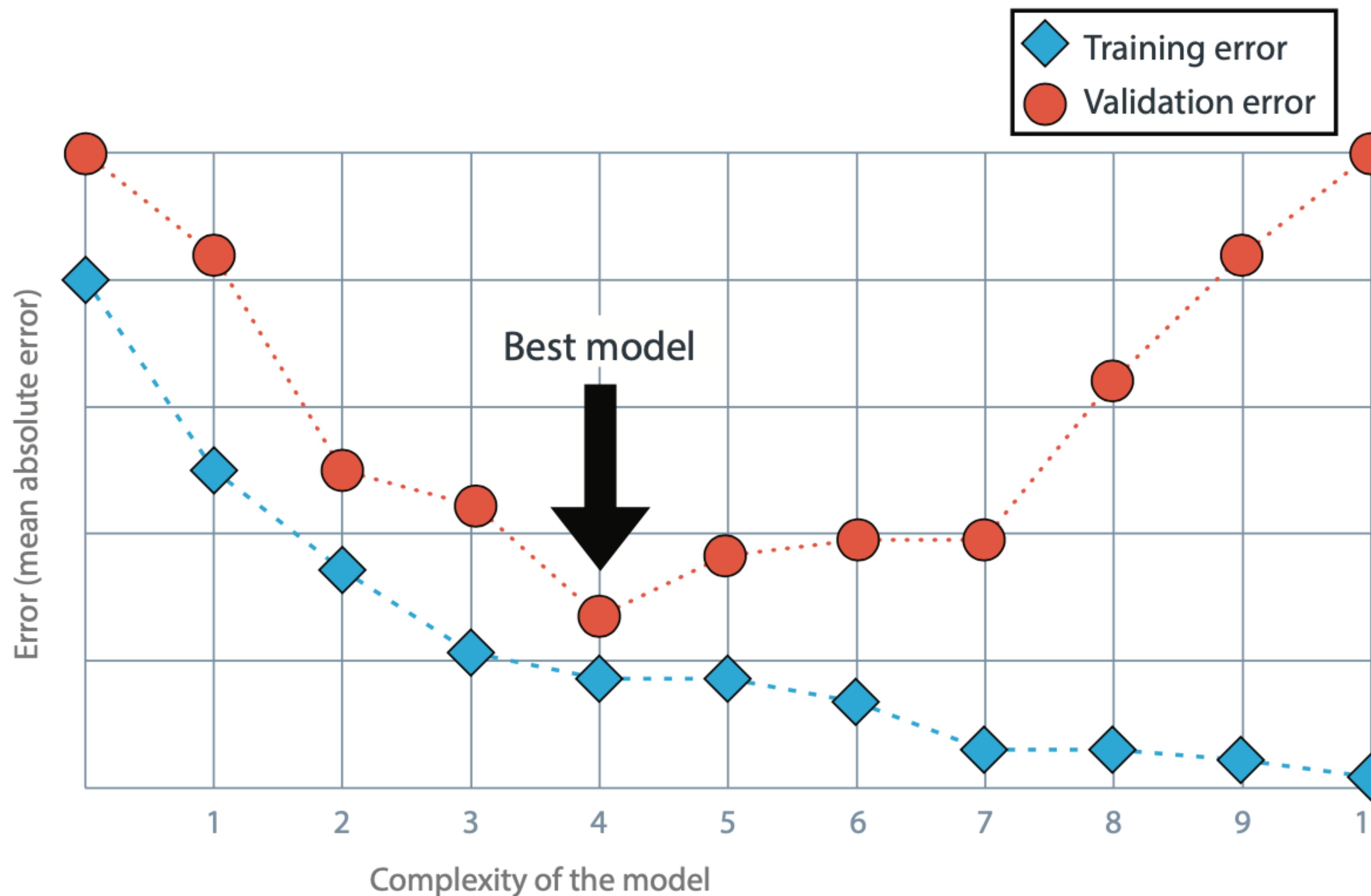


Underfitting



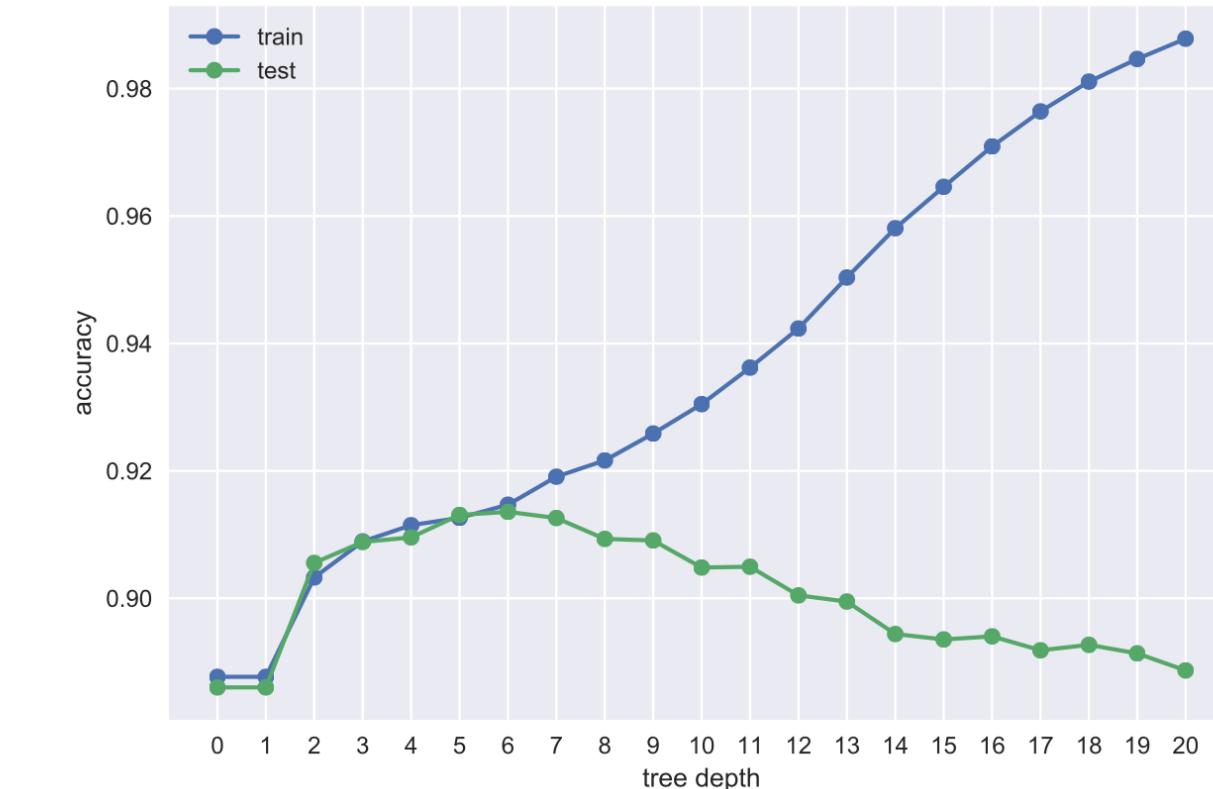
Overfitting



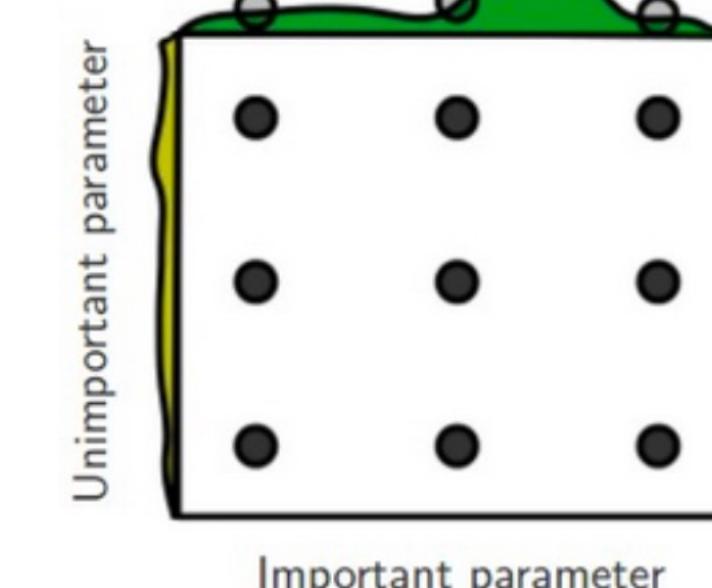


Tuning Hyperparameters

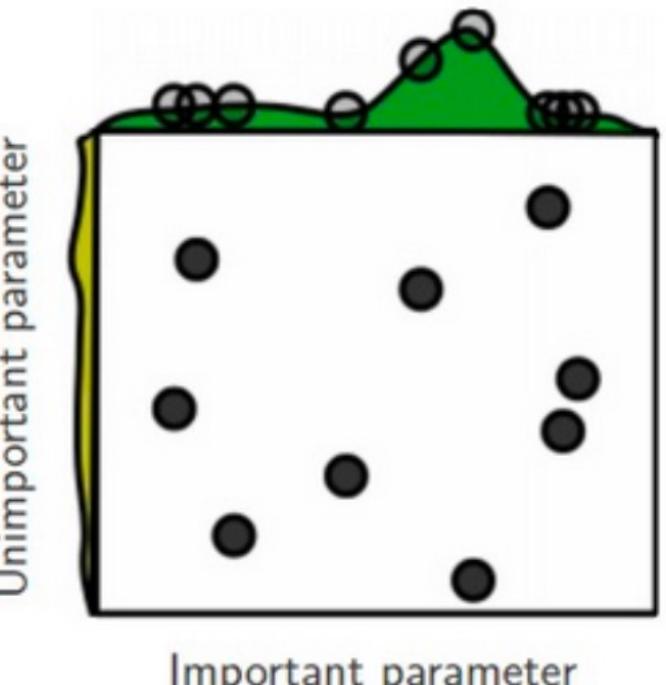
- Inputs to the learning algorithms that control their behaviour
 - Examples:
 - maximum tree depth in decision trees
 - number of neighbours k in *k-nearest neighbour*
 - Neural networks: architecture, learning rate, etc.
- For a model to work well, they often need to be tuned carefully
 - Huge search space! may be inefficient to search exhaustively
- Possible approaches
 - **Grid search**: brute-force exhaustive search among a finite set of hyperparameter settings
 - All combinations are tried, then the best setting selected
 - **Random search**: for each hyperparameter, define a distribution (e.g. normal, uniform)
 - In the search loop, we sample randomly from these distributions



Grid Layout



Random Layout

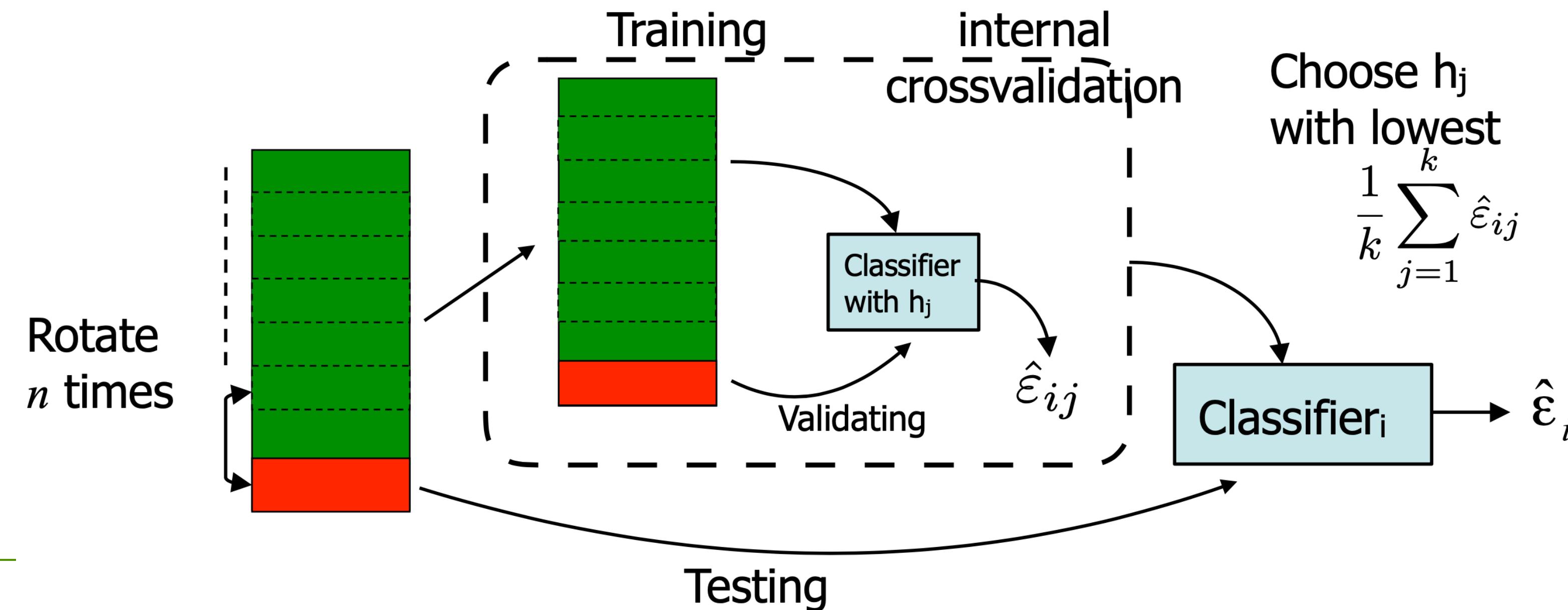


<https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf>

DON'T optimise these numbers by looking at the test set! That is CHEATING!

Double Cross-Validation

- To optimise over the hyperparameter do cross-validation inside another cross-validation
- The minimum error is often not the most interesting. Try to understand the advantages/disadvantages
 - What errors are made? (inspect objects, inspect labels)
 - What classes are problematic? (confusion matrix)
 - Does adding training data help? (learning curve)
 - How robust is the model?



Sources

- CIS 419/519 Applied Machine Learning. Eric Eaton, Dinesh Jayaraman.
<https://www.seas.upenn.edu/~cis519/spring2020/>
- EECS498: Conversational AI. Kevin Leach. <https://dijkstra.eecs.umich.edu/eecs498/>
- CS 4650/7650: Natural Language Processing. Diyi Yang.
https://www.cc.gatech.edu/classes/AY2020/cs7650_spring/
- Natural Language Processing. Alan W Black and David Mortensen. <http://demo.clab.cs.cmu.edu/NLP/>
- IN4325 Information Retrieval. Jie Yang.
- Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Third Edition. Daniel Jurafsky, James H. Martin.
- Natural Language Processing, Jacob Eisenstein, 2018.
- A Step-by-Step Explanation of Principal Component Analysis (PCA). <https://builtin.com/data-science/step-step-explanation-principal-component-analysis>

Advanced Machine Learning For Design

Lecture 6: Train, Evaluate and Integrate Machine
Learning Models (part 1)

Module 3

Evangelos Niforatos

25/10/2023

aml4d-ide@tudelft.nl
<https://aml4design.github.io/>